Information Builders

WebFOCUS

Developing Reporting Applications

Release 8.2 Version 01M

February 24, 2021

Active Technologies, FOCUS, Hyperstage, Information Builders, the Information Builders logo, iWay, iWay Software, Omni-Gen, Omni-HealthData, Parlay, RStat, Table Talk, WebFOCUS, WebFOCUS Active Technologies, and WebFOCUS Magnify are registered trademarks, and DataMigrator, and ibi are trademarks of Information Builders, Inc.

Adobe, the Adobe logo, Acrobat, Adobe Reader, Flash, Adobe Flash Builder, Flex, and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Due to the nature of this material, this document refers to numerous hardware and software products by their trademarks. In most, if not all cases, these designations are claimed as trademarks or registered trademarks by their respective companies. It is not this publisher's intent to use any of these names generically. The reader is therefore cautioned to investigate all claimed trademark rights before using any of these names other than to refer to the product described.

Copyright © 2017. TIBCO Software Inc. All Rights Reserved.

Contents

Preface 23
Conventions 24
Related Publications 25
Information You Should Have 25
User Feedback 26
Information Builders Consulting and Training 26

1. WebFOCUS Application Logic 27
Three-Tier Application Logic 27
Distributing Processing Among Platforms. 31
Publishing an Application 31
WebFOCUS Components 31
Section 508 Accessibility in WebFOCUS 36

2. Managing Applications 37
Managing Applications 37
What Is an Application? 37
Generating Samples and Tutorials 40
Procedures and Metadata on the Application Tree 41
Managing Applications and Paths 56
Creating and Mapping Applications. 57
Using an SQL Database to Store Application Contents. 63
Nested Application Directories. 69
Home Application Directories for Users. 73
Configuring the Application Path. 78
Filtering the Application Tree. 81
Searching for Files. 87
Sorting the Application Tree. 88
Managing Application Files. 88
Application Commands Overview 90
Search Path Management Commands 93
APP PATH. 94
APP PREPENDPATH. 94

APP APPENDPATH. 95
APP MAP. 96
APP SET METALOCATION_SAME. 98
APP ? METALOCATION_SAME. 98
APP SHOWPATH. 98
Application and File Management Commands 99
APP CREATE. 99
APP COPY. 102
APP COPYF[ILE]. 102
APP MOVE. 103
APP MOVEF[ILE]. 104
APP DELETE. 105
APP DELETEF[ILE]. 105
APP PROPERTY CODEPAGE. 106
APP RENAME. 107
APP RENAMEF[ILE]. 107
Designating File Types for APP Commands. 108
Output Redirection Commands 112
APP HOLD. 114
APP HOLDDATA. 115
APP HOLDMETA. 115
APP FI[LEDEF]. 116
Application Metadata Commands and Catalog Metadata 116
Retrieving Basic Information. 116
STATE. 117
APP LIST. 118
APP QUERY. 119
Retrieving Extended Catalog Information. 121
catalog/sysapps. 121
catalog/sysfiles. 122
Help Information: APP HELP 124
Restricting the Use of APP Commands 124
Accessing Metadata and Procedures 126

Search Rules. 126
Creation Rules for Procedure Files. 127
Locating Master Files and Procedures. 127
Accessing Existing Data Files. 128
Creation Rules for Data Files. 128
Data Set Names. 132
Allocating Temporary Files 133
Temporary Space Usage and Location 136
Temporary Disk Space Usage for Non-PDS Deployment 136
Application Tools 139
EX Procedure and Amper Variables. 139
EX EDAMAIL. 140

3. Coding a User Interface 147

Which Tools Can You Use? 147
Choosing Search Path Logic. 148
The WebFOCUS Client 150
Using the Servlet 151
Using a Dynamic Multiselect Drop-Down List 157
Enabling Ad Hoc Reporting 162
Validating a Form With JavaScript 165
WebFOCUS Autoprompt Facility 168
Autoprompt Configuration. 169
Responsive Autoprompt 171
Responsive Autoprompt Page Components. 172
Selection Lists. 173
Simple Filter 179
Calendar Control. 180
Responsive Sample Code (Parameter_Type.fex). 182
HTML Autoprompt 184
HTML Autoprompt Page Components. 184
Autoprompt Considerations 187
Defining Parameter-Based Filters 190

Adding a Simple Filter to an Autoprompt Form. 190
Adding a Variable Description to a Filter. 191
Specifying a Format for a Variable. 192
Setting a Default Variable Value. 193
Setting a Hidden Variable. 195
Adding a Single-Select List of Values. 196
Adding a Multiselect List of Values. 200
Adding Static Lists of Display and Sort Fields. 208
Adding Dynamic Lists of Display and Sort Fields. 214
Customizing the Autoprompt Facility 222
Specifying the HTML Template In a Procedure. 222
Customizing the HTML Autoprompt Facility. 222
Specifying the HTML Template In a Procedure. 223
Specifying the Prompting Level in a URL to Run a Report. 224
Adding a Description of the Procedure to the Autoprompt Internal XML. 225
Displaying a Report on the Default WebFOCUS Page 225
Designing an HTML Page for Report Display 231
Displaying an Accordion By Row Report Using HOLD FORMAT HTMTABLE and -HTMLFORM. 248
Displaying an HTML HFREEZE Report Using HOLD FORMAT HTMTABLE and -HTMLFORM. 252
Displaying an Active Technologies Report Using HOLD format AHTMLTAB and -HTMLFORM. 256
Using HOLD FORMAT XML and -HTMLFORM. 257

4. Enhancing a User Interface 259

Displaying a Report in a Helper Application 259
Controlling Multiple Reports 266
Adding JavaScript for Drill-Down Reporting 279
Facilitating Report Manipulation 286
Navigating a Report With the WebFOCUS Viewer. 288
Opening and Closing the WebFOCUS Viewer. 290
Controlling Button Display on the WebFOCUS Viewer. 292
Using the Viewer Control Panel. 293

Searching a Report. 294
Printing With On-Demand Paging. 297
Using a Cascading Style Sheet to Standardize Display 298
Displaying a Previously Run Report 304
Passing a User ID From HTML for a Custom Menu 305
Customizing a Menu. 306

5. Managing Flow of Control in an Application 329

Uses for Dialogue Manager 330
Dialogue Manager Processing 332
Creating a Dialogue Manager Procedure 335
Commenting a Procedure. 335
Customizing a Procedure With Variables 336
Supplying a Default Variable Value. 339
Supplying Variable Values in an Expression. 340
Supplying Variable Values From Another Procedure. 343
Reading Variable Values From and Writing Variable Values to an External File. 346
Closing an External File. 352
Supplying Dates as Variable Values. 352
Using Variables With Cross-Century Dates. 352
Performing a Calculation on a Variable. 353
Changing a Variable Value With the DECODE Function. 353
Validating a Variable Value. 354
Verifying the Presence, Value, Length, or Type of User-Supplied Values. 354
Validating Variable Values Without Data File Access: REGEX. 360
Verifying User-Supplied Values Against a Set of Format Specifications. 362
Verifying User-Supplied Values Against a Value Range. 363
Counting With an Indexed Variable. 364
Creating a Standard Quote-Delimited String 365
Creating and Working With Variables 369
Creating and Working With Local and Global Variables. 370
Concatenating Variables. 373
Removing Trailing Blanks From a Variable. 373

Displaying the Value of a Variable. 375
Assigning the Value of a Variable to a Parameter. 376
Working With System and Statistical Variables. 377
Using Numeric Amper Variables in Functions 384
Determining Amper Variable Data Type. 384
Manipulating Amper Variables. 385
Using an Amper Variable in an Expression. 386
Using Amper Variables as Function Parameters. 387
Using a Numeric Amper Variable as a Numeric Function Parameter. 387
Using a Numeric Amper Variable as an Alphanumeric Function Parameter. 388
Controlling the Execution of a Procedure 389
Executing Stacked Commands and Continuing the Procedure. 389
Executing Stacked Commands and Exiting the Procedure. 390
Canceling the Execution of a Procedure. 391
Navigating a Procedure 392
Performing Unconditional Branching. 392
Performing Conditional Branching. 393
Performing a Compound -IF Test. 397
Looping in a Procedure. 397
Calling Another Procedure With -INCLUDE. 401
Using Fully Qualified Names With the -Include Command. 403
Nesting Procedures With -INCLUDE. 405
Calling Another Procedure With EXEC. 406
Enhancing an HTML Webpage With a Procedure 407
Referring to an External Webpage. 407
Embedding HTML Commands in a Procedure. 408
Embedding a Procedure in an HTML Webpage. 410
Including Variables in an HTML Webpage. 415
Issuing Operating System Commands 417
Change Directory Operating System Command. 418
Executing Multiple Operating System Commands on a Single Line. 418
Reviewing Command Output. 419
Controlling Passwords With Dialogue Manager 420

Sending a Message to the Application 420
Testing and Debugging a Dialogue Manager Procedure 421
Viewing Messages for Debugging an Application. 426
Dialogue Manager Syntax Reference 428
-* Command. 428
-? Command. 428
-CLOSE. 429
-DEFAULT. 429
-DEFAULTH. 430
-DOS. 430
-EXIT. 430
-GOTO. 431
-HTMLFORM. 431
-IF. 432
-INCLUDE. 433
-label. 433
-MVS. 434
-PASS. 434
-QUIT. 435
-QUIT FOCUS. 435
-READ. 436
-READFILE. 436
-REPEAT. 437
-RUN. 438
-SET. 438
-TSO. 439
-TYPE. 439
-UNIX. 440
-VMS. 440
-WINNT. 440
-WRITE. 441

6. Testing and Debugging a Procedure 443

Debugging Your Application With Query Commands 444
Displaying Combined Structures 446
Displaying Virtual Fields 447
Displaying the Currency Data Source in Effect 448
Displaying Available Fields 448
Displaying the File Directory Table 449
Displaying Field Information for a Master File 451
Displaying Data Source Statistics 451
Displaying Current ddnames Assigned With FILEDEF 453
Displaying Defined Functions 454
Displaying HOLD Fields 454
Displaying JOIN Structures 455
Displaying National Language Support 456
Displaying Explanations of Error Messages 457
Displaying the Current Search Path 457
Displaying the Release Number 458
Displaying the Values of a Remote Server 459
Displaying Parameter Settings 459
Displaying Graph Parameters 461
Displaying the Site Code of the Connected Server 462
Displaying Command Statistics 463
Displaying StyleSheet Parameter Settings 464
Displaying Information About the SU Machine 466
Displaying Data Sources Specified With USE 467
Displaying Global Variable Values 467
Identifying the Files Being Used 468
Reporting Dynamically From System Tables 470
Overview of System Table Synonyms 470
SYSAPPS: Reporting on Applications and Application Files 472
SYSCOLUM: Reporting on Tables and Their Columns 473
SYSDEFFN: Reporting on DEFINE FUNCTIONS 474
SYSERR: Reporting on Error Message Files 475
SYSFILES: Reporting on Metadata or Procedure Directory Information 476

SYSIMP: Reporting on Impact Analysis Information. 478
SYSINDEX: Reporting on Index Information. 479
SYSKEYS: Reporting on Key Information. 480
SYSRPDIR: Reporting on Stored Procedures. 481
SYSSET: Reporting on SET Parameters. 482
SYSSQLOP: Reporting on Function Information. 482
SYSTABLE: Reporting on Table Information. 483
Reporting on Data Types. 484

7. Accessing a FOCUS Data Source 487

The USE Command 487
Identifying a FOCUS Data Source 488
Using Alternative File Specifications 492
Identifying a New Data Source 494
Protecting a Data Source 495
Concatenating Data Sources 496
Displaying the Current USE Options 499
Clearing the USE Options 500

8. Customizing Your Environment 501

When Do You Use the SET Command? 501
Ways to Issue a SET Command 502
Coding a SET Command 503
Types of SET Parameters 507
Calculations. 507
Data and Metadata. 508
Date Manipulation Tasks. 510
WebFOCUS-Specific Tasks. 512
Graph Tasks. 512
Memory Setup and Optimization Tasks. 515
Report Code, Content, and Processing Tasks. 516
Report Layout and Display Tasks. 519
Security Tasks. 524
SET Parameter Syntax 525

3D. 529
ACCBLN. 529
ACCESSHTML. 530
ACCESSIBLE. 530
ACCESSPDF. 531
ACROSSLINE. 531
ACROSSPRT. 532
ACROSSTITLE. 532
ACRSVRBTITL. 533
ALL 534
ALLOWCVTERR. 535
AREXPIRE. 536
ARGRAPHENGIN. 536
ARPASSWORD. 536
ASNAMES. 537
AUTODRILL. 538
AUTOFIT. 539
AUTOINDEX. 539
AUTOPATH. 540
AUTOSTRATEGY. 540
AUTOTABLEF. 541
AUTOTICK. 541
BARNUMB. 541
BASEURL. 542
BINS. 542
BLANKINDENT. 542
BOTTOMMARGIN. 543
BUSDAYS. 544
BYDISPLAY. 544
BYPANEL. 545
CACHE. 545
CARTESIAN. 546
CDN. 547

CENT-ZERO. 548
CNOTATION. 548
COLLATION. 549
COMPMISS. 550
COMPOUND. 550
COMPUTE. 551
COUNTWIDTH. 552
CSSURL. 552
CURRENCY_DISPLAY. 552
CURRENCY_ISO_CODE. 553
CURRENCY_PRINT_ISO. 553
CURRSYMB. 554
CURSYM_D. 554
CURSYM_E. 555
CURSYM_F. 555
CURSYM_G. 556
CURSYM_L. 556
CURSYM_Y. 556
DATE_ORDER. 557
DATE_SEPARATOR. 557
DATEDISPLAY. 558
DATEFNS. 558
DATEFORMAT. 559
DATETIME. 559
DBAJOIN. 560
DB_INFILE. 560
DBACSENSITIV. 561
DBASOURCE. 561
DEFCENT. 562
DEFECHO. 563
DEFINES. 563
DIRECTHOLD. 564
DMH_LOOPLIM. 564

DMH_STACKLIM. 565
DMPRECISION. 565
DROPBLNKLINE. 566
DTSTRICT. 566
DUPLICATECOL. 567
EMBEDHEADING. 567
EMPTYREPORT. 568
EQTEST. 568
ERROROUT. 569
ESTRECORDS. 569
EUROFILE. 570
EXCELSERVURL. 570
EXL2KLANG. 571
EXL2KTXTDATE. 572
EXPANDABLE. 572
EXPANDBYROW. 572
EXTAGGR. 573
EXTHOLD. 573
EXTENDNUM. 574
EXTRACT. 574
EXTSORT. 575
FIELDNAME. 575
FILECASE. 576
FILECOMPRESS. 576
FILE[NAME]. 577
FILTER. 577
FIXRET[RIEVE] 578
FLOATMAPPING. 578
FOC144. 579
FOCEXURL. 579
FOCFIRSTPAGE. 579
FOCHTMLURL. 580
FOCSTACK. 580

FOCTRANSFORM. 581
FORMULTIPLE. 581
GRAPHDEFAULT. 582
GRAPHEDIT. 582
GRAPHENGINE. 582
GRAPHSERVURL. 583
GRID. 583
GRMERGE. 584
GRMULTIGRAPH. 585
GRWIDTH. 585
GTREND. 585
HAUTO. 586
HAXIS. 586
HCLASS. 586
HDAY. 587
HIDENULLACRS. 587
HISTOGRAM. 588
HLDCOM_TRIMANV. 588
HMAX. 589
HMIN. 589
HNODATA. 589
HOLDATTR[S]. 590
HOLDFORMAT. 591
HOLDLIST. 591
HOLDMISS. 592
HOLDSTAT. 592
HSTACK. 593
HTICK. 593
HTMLARCHIVE. 593
HTMLCSS. 594
HTMLEMBEDIMG. 594
HTMLENCODE. 595
INDEX. 595

JOIN_LENGTH_MODE (JOINLM). 596
JOINOPT. 597
JPEGENCODE. 597
JPEGQUALITY. 598
JSURLS. 598
KEEPDEFINES. 599
KEEPFILTERS. 599
LANG[UAGE] 600
LAYOUTGRID. 602
LAYOUTRTL. 602
LEADZERO. 603
LEFTMARGIN. 603
LINES. 603
LOOKGRAPH. 604
MATCHCOLUMNORDER. 604
MAXDATAEXCPT. 605
MAXLRECL. 605
MDICARDWARN. 606
MDIENCODING. 606
MDIPROGRESS. 607
MESSAGE. 607
MISSINGTEST. 608
MULTIPATH. 608
NODATA. 609
NULL. 609
OFFLINE-FMT. 610
OLAPGRMERGE. 610
OLDSTYRECLEN. 610
ONFIELD. 611
ONLINE-FMT. 611
ORIENTATION. 612
OVERFLOWCHAR. 612
PAGE[-NUM]. 613

PAGESIZE. 613
PANEL. 615
PASS. 615
PCOMMA. 616
PDFLINETERM. 617
PERMPASS 618
PHONETIC_ALGORITHM. 618
POPUPDESC. 619
PPTXGRAPHTYPE. 619
PRINT. 620
PRINTDST. 620
PRINTPLUS. 621
PSPAGESETUP. 621
QUALCHAR. 621
QUALTITLES. 622
RANK. 623
RECAP-COUNT. 623
RECORDLIMIT. 623
RIGHTMARGIN. 624
RPAGESET. 624
SAVEDMASTERS. 625
SAVEMATRIX. 625
SHADOW. 625
SHIFT. 626
SHORTPATH. 626
SHOWBLANKS. 627
SPACES. 628
SQLTOPTTF. 628
SQUEEZE. 629
%STRICTMATH. 629
STYLEMODE. 630
STYLE[SHEET]. 630
SUBTOTALS. 631

SUMMARYLINES. 632
SUMPREFIX. 632
TARGETFRAME. 633
TEMP. 633
TEMPERASE. 633
TESTDATE. 634
TIME_SEPARATOR. 634
TITLELINE. 635
TITLES. 635
TOPMARGIN. 636
UNITS. 636
USER. 637
USERFCHK. 637
USERFNS. 638
VAUTO. 639
VAXIS. 639
VCLASS. 639
VGRID. 640
VISBARORIENT. 640
VMAX. 641
VMIN. 641
VTICK. 641
VZERO. 642
WARNING. 642
WEBARCHIVE. 642
WEBVIEWALLPG. 643
WEBVIEWCLOSE. 643
WEBVIEWCLMSG. 644
WEBVIEWER. 644
WEBVIEWHELP. 645
WEBVIEWHOME. 645
WEBVIEWTARG. 646
WEBVIEWTITLE. 646

WEEKFIRST. 647
WPMINWIDTH. 648
XRETRIEVAL. 648
YRTHRESH. 649

9. Defining and Allocating WebFOCUS Files 651

Allocating WebFOCUS Files 651
Dynamically Defining a File Under Windows and UNIX. 654
Assigning a Logical Name With the FILEDEF Command. 654
Displaying Current ddnames Assigned With FILEDEF. 657
Clearing Allocations. 658
Application Files Under Windows 658
Master Files Under Windows. 658
Locating a Master File Under Windows. 659
Access Files Under Windows. 659
Procedures Under Windows. 660
FOCUS Data Sources Under Windows. 660
External Indexes for FOCUS Data Sources Under Windows. 661
Other Supported Data Sources Under Windows. 661
Sequential Data Sources Under Windows. 661
WebFOCUS StyleSheet Files Under Windows. 662
Library of Functions Under Windows. 662
Profiles Under Windows. 662
Webpages Under WebFOCUS Under Windows. 662
Extract Files Under Windows 663
HOLD Files Under Windows. 663
SAVB Files Under Windows. 664
SAVE Files Under Windows. 664
HOLDMAST Files Under Windows. 665
Work Files Under Windows 665
Determining If A File Exists Under Windows 666
WebFOCUS Files Under MVS 667
Referencing Files Under MVS. 667

Dynamically Defining Files Under MVS. 668
Application Files Under MVS 669
Master Files Under MVS. 669
Access Files Under MVS. 670
Procedures Under MVS. 670
FOCUS Data Sources Under MVS. 671
Using FOCUS Data Sources Under MVS. 671
External Indexes for FOCUS Data Sources Under MVS. 672
Other Supported Data Sources Under MVS. 672
WebFOCUS StyleSheet Files Under MVS. 672
Profiles Under MVS. 673
Extract Files Under MVS 673
HOLD Files Under MVS. 673
SAVB Files Under MVS. 674
SAVE Files Under MVS. 674
HOLDMAST Files: Temporary Master Files Under MVS. 675
HOLDSTAT Files Under MVS. 675
Work Files Under MVS 676
Reviewing Attributes of Allocated Files Under MVS 677
Displaying Allocated Files With ? TSO DDNAME Under MVS. 677
Displaying File Attributes With ? TSO DDNAME ddname Under MVS. 678
Displaying File Attributes With ? TSO DDNAME ddname. 678
Placing File Attributes in Dialogue Manager Commands. 679
Determining If a Data Source Exists Under MVS. 681
Estimating Data Set Sizes to Determine Available Space Under MVS. 682
Application Files Under UNIX 683
Master Files Under UNIX. 683
Access Files Under UNIX. 684
Procedures Under UNIX. 684
FOCUS Data Sources Under UNIX. 685
External Indexes for FOCUS Data Sources Under UNIX. 686
Sequential Data Sources Under UNIX. 686
WebFOCUS StyleSheets Under UNIX. 686

Profiles Under UNIX. 687
Extract Files Under UNIX 687
HOLD Files Under UNIX. 687
SAVB Files Under UNIX. 688
SAVE Files Under UNIX. 688
HOLDMAST Files Under UNIX. 688
Work Files Under UNIX 689
Determining If A File Exists Under UNIX 690

10. Euro Currency Support 691

Integrating the Euro Currency 691
Converting Currencies 692
Creating the Currency Data Source 693
Identifying Fields That Contain Currency Data 696
Activating the Currency Data Source 698
Processing Currency Data 699
Querying the Currency Data Source in Effect 703
Punctuating Numbers 703
Selecting an Extended Currency Symbol 705

Preface

This content describes how to use WebFOCUS to create and deploy self-service report applications on the Internet or corporate intranets. It is intended for application developers. This documentation is part of the WebFOCUS documentation set.

How This Manual Is Organized

This manual includes the following chapters:

	Chapter/Appendix	Contents
1	WebFOCUS Application Logic	Describes the underlying logic of WebFOCUS that exists from the process of developing your application to the time a user launches a report request in a web browser and sees the results return to the browser.
2	Managing Applications	
3	Coding a User Interface	Describes how to code a user interface with HTML and JavaScript.
4	Enhancing a User Interface	Describes coding capabilities that extend the functionality and usability of an interface, including the display of reports in popular formats and the display of multiple reports on a single launch page.
5	Managing Flow of Control in an Application	Describes how to control the processing of a procedure with Dialogue Manager.
6	Testing and Debugging a Procedure	Describes how to diagnose problems encountered when running a procedure.
7	Accessing a FOCUS Data Source	Describes how to assign a logical name to a FOCUS data source.
8	Customizing Your Environment	Describes how to control your WebFOCUS Environment with the SET command.
9	Defining and Allocating WebFOCUS Files	Provides file allocation requirements for WebFOCUS operating systems, and describes how to assign a logical name to a file or device.

	Chapter/Appendix	Contents
10	Euro Currency Support	Describes how to create and use a currency data source to convert to and from the new euro currency.

Conventions

The following table describes the conventions that are used in this manual.

Convention	Description

`THIS TYPEFACE`
or
`this typeface`
	Denotes syntax that you must enter exactly as shown.

`this typeface`	Represents a placeholder (or variable) in syntax for a value that you or the system must supply.
`underscore`	Indicates a default setting.
this typeface	Represents a placeholder (or variable), a cross-reference, or an important term. It may also indicate a button, menu item, or dialog box option that you can click or select.
Key + Key	Indicates keys that you must press simultaneously.
{ }	Indicates two or three choices. Type one of them, not the braces.
[ ]	Indicates a group of optional parameters. None is required, but you may select one of them. Type only the parameter in the brackets, not the brackets.
|	Separates mutually exclusive choices in syntax. Type one of them, not the symbol.
...	Indicates that you can enter a parameter multiple times. Type only the parameter, not the ellipsis (...).

Convention	Description
. . .	Indicates that there are (or could be) intervening or additional commands.

Related Publications

Visit our Technical Content Library at *http://documentation.informationbuilders.com*. You can also contact the Publications Order Department at (800) 969-4636.

Information You Should Have

To help our consultants answer your questions effectively, be prepared to provide the following information when you call:

- Your six-digit site code (*xxxx.xx*).
- Your WebFOCUS configuration:
  - The front-end software you are using, including vendor and release.
  - The communications protocol (for example, TCP/IP or HLLAPI), including vendor and release.
  - The software release.
  - Your server version and release. You can find this information using the Version option in the Web Console.
- The stored procedure (preferably with line numbers) or SQL statements being used in server access.
- The Master File and Access File.
- The exact nature of the problem:
  - Are the results or the format incorrect? Are the text or calculations missing or misplaced?
  - Provide the error message and return code, if applicable.
  - Is this related to any other problem?

- Has the procedure or query ever worked in its present form? Has it been changed recently? How often does the problem occur?
- What release of the operating system are you using? Has it, your security system, communications protocol, or front-end software changed?
- Is this problem reproducible? If so, how?
- Have you tried to reproduce your problem in the simplest form possible? For example, if you are having problems joining two data sources, have you tried executing a query containing just the code to access the data source?
- Do you have a trace file?
- How is the problem affecting your business? Is it halting development or production? Do you just have questions about functionality or documentation?

User Feedback

In an effort to produce effective documentation, the Technical Content Management staff welcomes your opinions regarding this document. You can contact us through our website, *http://documentation.informationbuilders.com/connections.asp*.

Thank you, in advance, for your comments.

Information Builders Consulting and Training

Interested in training? Information Builders Education Department offers a wide variety of training courses for this and other Information Builders products.

For information on course descriptions, locations, and dates, or to register for classes, visit our website (*http://education.informationbuilders.com*) or call (800) 969-INFO to speak to an Education Representative.

Chapter 1

WebFOCUS Application Logic

To develop self-service reporting applications for the web, you need software components that manage the flow from the time you begin to develop your project to the time a user launches a report request in a web browser and sees the results returned to the browser. A project is a WebFOCUS application being developed on the local server. After building and testing a project, you can publish it as a web application.

In this chapter:

- Three-Tier Application Logic
- Publishing an Application
- WebFOCUS Components
- Section 508 Accessibility in WebFOCUS

Three-Tier Application Logic

The three tiers of application logic in a published application are designed to exploit the capabilities of web processing and enhance the performance, scalability, and maintenance of an application. Partitioning an application into tiers is a means of classifying the functionality of the application. You can publish partitioned components across many different platforms. See *Distributing Processing Among Platforms* on page 31.

The three tiers of application logic are:

- **Presentation layer,** which is commonly known as the user interface, or the front end of an application. In a WebFOCUS reporting application, the presentation layer consists of webpages (including HTML files, Cascading Style Sheets, JavaScript™, Java classes, and images).

  The presentation layer resides on the web server and is handled by the web server and the web browser of the end user.

- **User interface logic,** which is any application logic other than presentation or data access logic. User interface logic allows you to create and manipulate the components of a report. The WebFOCUS Servlets program is configured to provide the user interface logic. An example of user interface logic is the WebFOCUS Servlet that powers OLAP.

This logic is provided by Information Builders, and resides on the Application Server or servlet engine plug-in to the web server.

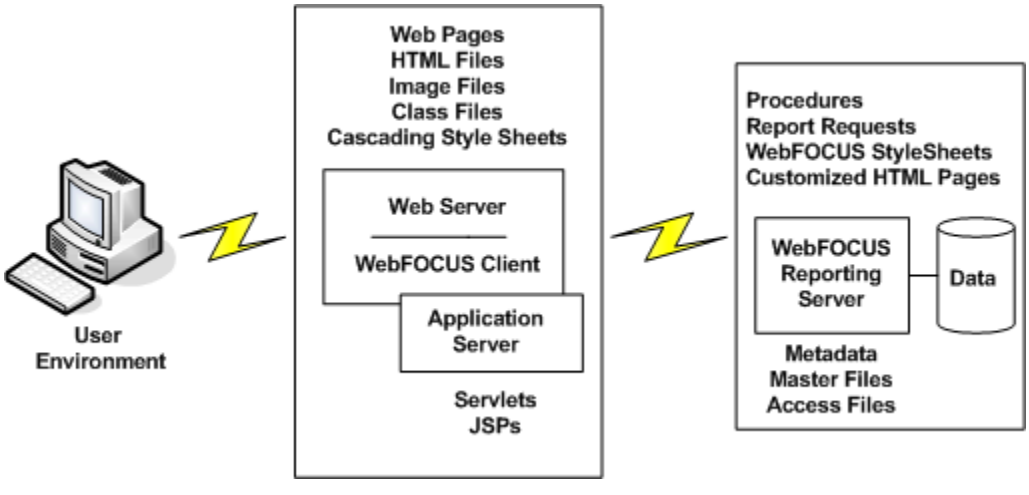
- **Data access and formatting logic,** which retrieves information from a data source. Data access logic contains the metadata that defines the data sources your applications access. It consists of Master Files and Access Files. Formatting logic consists of the procedures (focexecs), WebFOCUS StyleSheets, certain customized HTML files and embedded graphics, developed by knowledge workers using WebFOCUS. These application objects display reports or graphs, or perform more complicated processing based on values that the end user supplies.

  Data access logic usually resides on the same server as the data itself (either the WebFOCUS Reporting Server or a sub-server). A WebFOCUS reporting application takes advantage of the data access logic of Information Builders middleware technology.

For an illustration, see *Partitioned Application Files and Servers* on page 28.

Reference: Partitioned Application Files and Servers

The following diagram illustrates how all user interface logic and data access logic can be configured behind multiple firewalls for maximum security.



The web server processes the webpages that provide the presentation logic for a WebFOCUS reporting application. The web server accesses HTML files, graphical image files, and Java class files, and Cascading Style Sheets. For more information about how to specify file locations on the web server, see the WebFOCUS installation guide for your platform.

The application server provides the user interface logic for your WebFOCUS tools, and is the location for servlets, JSPs, and Java Beans. Having an application server to process application logic behind the user interface provides performance and security benefits.

Information Builders does not provide an application server. You can use a third-party server, such as WebLogic, WebSphere, Tomcat, Apache, or Netscape.

The Application server resides separately or together with the web server.

The WebFOCUS Reporting Server processes the procedures containing the user interface logic for reports, as well as the files that contain metadata for the data sources. Stored on the WebFOCUS Reporting Server are report requests, WebFOCUS StyleSheets, customized HTML pages called from WebFOCUS procedures, and Master and Access Files.

The WebFOCUS Reporting Server also accesses the data sources used by an application.

Reference: **WebFOCUS File Types**

A WebFOCUS reporting application can publish and use all of the following types of files. The following chart includes file extensions used by the Windows and UNIX operating systems. See *Defining and Allocating WebFOCUS Files* on page 651 for comparable information for z/OS and other supported operating environments.

File Type	Description	Location After Publishing	File Extension (Windows/UNIX)
Webpage	Includes files displayed for the end user through a web browser, such as HTML files, graphical images, Java executable objects (class files), and Cascading Style Sheets.	In web server home directory or in web server alias	.htm .html .jpg .gif .css .js .class .jar
User interface logic	Includes servlets and JSPs.	Application server	.jsp .class

File Type	Description	Location After Publishing	File Extension (Windows/UNIX)
Procedure	Includes files that contain the executable functions of an application: report requests, WebFOCUS StyleSheets, and customized HTML called from WebFOCUS procedures using the Dialogue Manager command -HTMLFORM. For details about this command, see *Enhancing an HTML Webpage With a Procedure* on page 407.	On the path of the WebFOCUS Reporting server	.fex .sty .htm
Master File Access File	Includes all Master Files and Access Files.	On the path of the WebFOCUS Reporting server	.mas .acx
Data source	Includes all supported data sources types.	On the platform with the WebFOCUS Reporting Server or sub-servers	n/a
Temporary file	Includes data extracts, temporary files that your application creates during processing, and temporary work files that WebFOCUS uses internally.	In the EDA temporary directory (default) or a user-defined EDA location	.ftm or other requested extension

Distributing Processing Among Platforms

WebFOCUS applications are capable of distributing processing over many platforms, with the following advantages:

- **You can access data on multiple platforms.** This allows you to form relationships among disparate data sources.
- **Each component runs in the most suitable environment.** For example, a WebFOCUS HTML front-end may run on a Windows or UNIX platform managed by local departmental administrators who authorize and group users, while the data access routines run on a z/OS machine capable of securing the data and aggregating it quickly.
- **You can speed up your applications.** Procedures that access data can run on the platform where the data resides, thus ensuring that any aggregation or screening takes place immediately. This means that your application is not shipping large quantities of data across a network to be aggregated or screened somewhere else. Less network traffic means increased application speed.

For a more detailed discussion of this topic, see the WebFOCUS installation guide for your platform.

Publishing an Application

After you have developed a WebFOCUS reporting application that is logically partitioned into project components, you must publish those components to the web server and the WebFOCUS Reporting Server. Publishing is the process of copying or transferring the files to the appropriate server. All of your files must be placed in the proper paths so that the appropriate server can find them.

You can use an external tool, such as FTP, to transfer them to the appropriate server and platform. In this case, you will need to provide path information to ensure that these files can be located for processing.

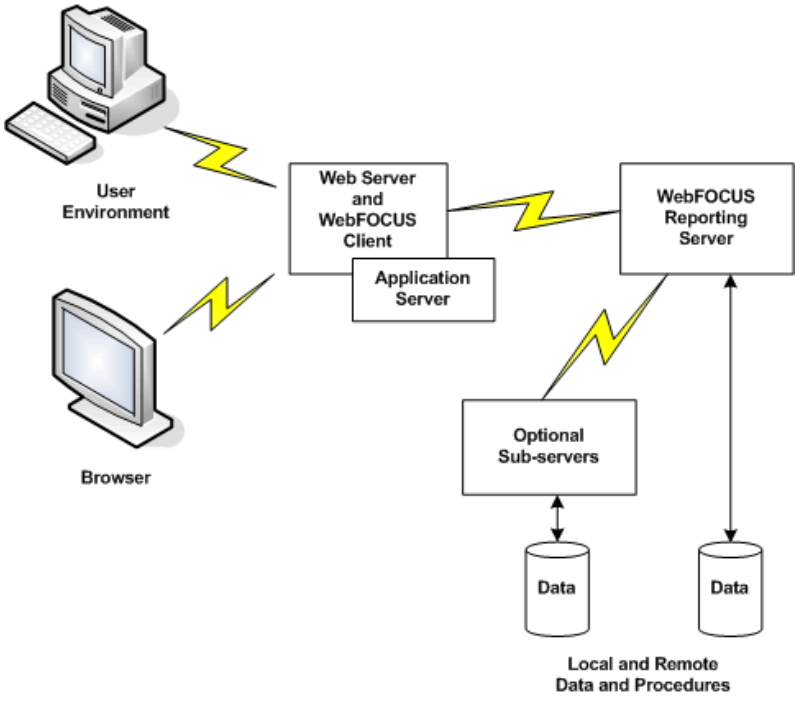
WebFOCUS Components

For development and publishing of reporting applications to the web, you need software components that manage the flow. A WebFOCUS system comprises several components, including:

- **Standard web components.** A web server and a web browser that manages the display of webpages.
- **WebFOCUS product components.** These include App Studio, and the WebFOCUS client and server components that manage processing of data and requests.

- **WebFOCUS application components.** These include user-created procedures and data.

More detailed descriptions of each component follow the diagram.



App Studio, the recommended WebFOCUS development environment, enables you to build, test, and publish reporting applications using Windows-based graphical tools, optionally supplemented by manual coding for maximum customization. App Studio tools handle a wide range of tasks, including report, graph, and form design.

The web browser is responsible for displaying web pages that may include reports, and graphs returned by the query. WebFOCUS works with Microsoft® Internet Explorer®.

The web server handles requests by fetching HTML files from and returning them to the browser. When webpages contain calls to the WebFOCUS client, the web server launches the WebFOCUS client, which collects variables and sends the request to the WebFOCUS Reporting Server.

The application server processes user interface logic, runs servlets and compiles JSPs.

The WebFOCUS client resides on the web server. It is implemented as Java servlets. WebFOCUS works with any standard web server that supports the selected option. These include Microsoft, IBM®, and NCSA-compatible web servers.

The WebFOCUS Reporting Server is responsible for accessing data, processing business logic, and generating fully styled output. It stores report procedures, WebFOCUS StyleSheets, and metadata (data source descriptions). You need one or more WebFOCUS Reporting Servers to run procedures and to access data sources.

Optional WebFOCUS sub-servers access data on remote platforms sources.

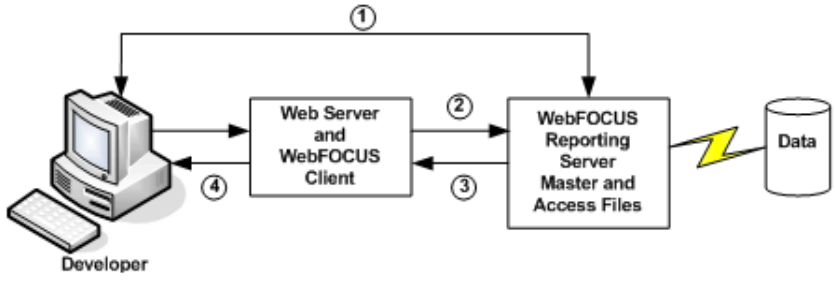
Data. A single server can access multiple types of data. For each type of DBMS, a data adapter must be installed.

Procedures. Procedures can reside on a WebFOCUS Reporting Server or a sub-server. At run time, the procedure requested by the WebFOCUS client is accessed by the WebFOCUS Reporting Server and executed. Procedures may call other procedures. Called procedures also reside on a WebFOCUS Reporting Server or sub-server.

Three additional diagrams will help you visualize how these components work together, in three integrated phases, to form a total development to publishing solution.

- *Phase 1: Develop and Test a Project* on page 33.
- *Phase 2: Partition and Publish Project Components* on page 34
- *Phase 3: Run the Published Application From the Web* on page 35.

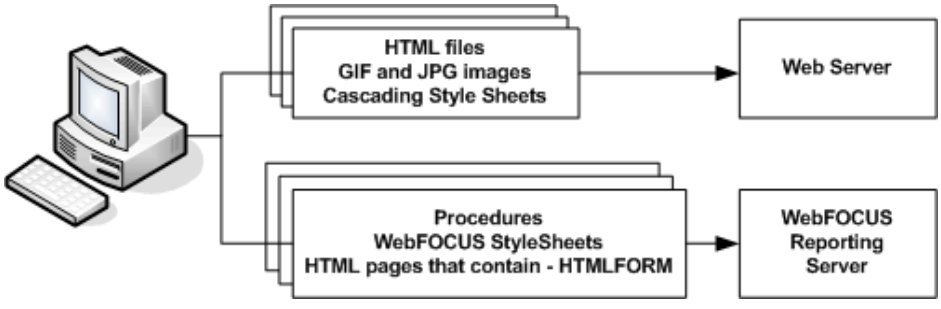
Reference: **Phase 1: Develop and Test a Project**



1. The application developer creates metadata (synonyms) for data sources on the server.
2. The application developer creates an application in App Studio, using graphical tools (supplemented by code, if desired). The application consists primarily of *reporting procedures* and *HTML forms* from which the procedures can be launched. The procedures and HTML files are stored on a server.

3. The developer runs and tests procedures and HTML forms. With each test, the request passes to a web server, through the WebFOCUS client, to a WebFOCUS Reporting Server, which retrieves the data and processes the request. During the testing process, WebFOCUS uses the Master File and Access File on the server to interpret and access the data.
4. The WebFOCUS Reporting Server then sends the results back to the WebFOCUS client, which returns it to the web server for display.

Reference: **Phase 2: Partition and Publish Project Components**

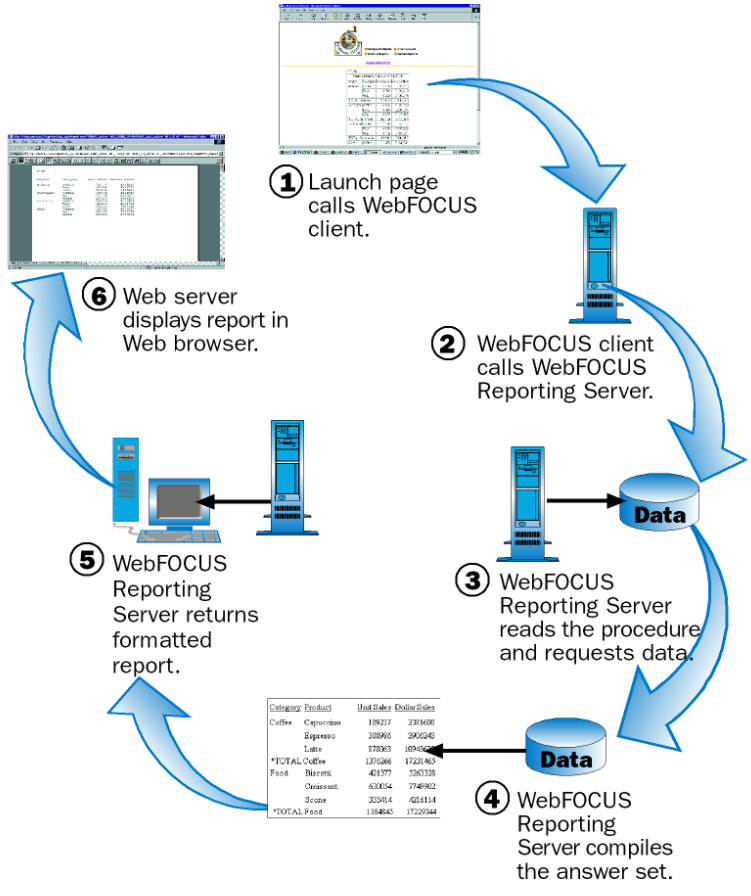


After testing has been completed (in phase 1), the developer publishes the application.

Presentation logic resides on the web server, and user interface logic resides on the WebFOCUS Reporting Server.

The developer can manually add other project components, such as image files, class files, and Cascading Style Sheets to the appropriate server.

Reference: Phase 3: Run the Published Application From the Web



1. A user opens a launch page in a web browser and selects a report. The launch page, through the web server, calls the WebFOCUS client. The request specifies the report procedure to be executed, as well as any parameters and values required by the procedure. Optionally, HTTP environment parameters set by the user connection to the web server may be passed to the procedure. The procedure name and all parameters are passed to the WebFOCUS client, which resides on the web server of your company.
2. The WebFOCUS client instructs the WebFOCUS Reporting Server to open an agent process for the request. The agent locates and executes the procedure.
3. The report procedure requests data from the data source (for example, Oracle). The data source may be local to or remote from the WebFOCUS Reporting Server. If it is remote, it is accessed through an intermediate sub-server, not shown in this simplified diagram.
4. The WebFOCUS Reporting Server compiles the answer set from the data source.

5. The WebFOCUS Reporting Server formats the answer set in the requested format (for example, HTML, PDF, Excel 2000), and returns the report output to the web server.
6. The web server passes the report output to the user browser. The browser displays the report in an HTML page or calls the appropriate desktop product, based on the file format, to display the output.

Section 508 Accessibility in WebFOCUS

WebFOCUS provides accessibility that meets the requirements set by Section 508. WebFOCUS complies with accessibility requirements as a result of the following features:

- The ACCESSIBLE and ACCESSHTML SET parameters generate HTML report output that are Section 508 compliant. The ACCESSHTML parameter supersedes the ACCESSIBLE parameter (which is still supported) because it will allow for future expansion to support additional accessibility standards. This HTML code automatically turns off pagination and column titles, page headings display only at the beginning of the report, and page footings are displayed only at the end of the report. For more information on this parameter, see *Customizing Your Environment* on page 501 and Technical Memo 4505: WebFOCUS HTML Report Accessibility Support.
- The ACCESSPDF SET parameter generates PDF report output that is Section 508 compliant.
- The SUMMARY WebFOCUS StyleSheet attribute maps reports and graphs to the HTML <TABLE SUMMARY> tag and places a description of the report or graph into a summary object inside the HTML table. For details on this attribute, see *Laying Out the Report Page* in the *Creating Reports With WebFOCUS Language* manual.
- The ALT WebFOCUS StyleSheet attribute is for use with a drill-down report you specify in a WebFOCUS StyleSheet. It maps to the HTML ALT tag on a hyperlink in a report. It can also add descriptive text to an embedded image in an HTML report. For details on this attribute, see *Laying Out the Report Page* in the *Creating Reports With WebFOCUS Language* manual.

Chapter 2

Managing Applications

In this chapter:

- Managing Applications
- What Is an Application?
- Procedures and Metadata on the Application Tree
- Managing Applications and Paths
- Application Commands Overview
- Search Path Management Commands
- Application and File Management Commands
- Output Redirection Commands
- Application Metadata Commands and Catalog Metadata
- Help Information: APP HELP
- Restricting the Use of APP Commands
- Accessing Metadata and Procedures
- Allocating Temporary Files
- Temporary Space Usage and Location
- Temporary Disk Space Usage for Non-PDS Deployment
- Application Tools

Managing Applications

An application is a platform-independent repository for a group of related components, such as procedures, Master and Access Files, data files, HTML files, PDF files, and image files.

You can use a variety of application (APP) commands to control the application environment, including the application itself, its component files, and its search paths.

What Is an Application?

An application is a platform-independent repository for a group of related components, such as procedures, Master and Access Files, data files, HTML files, PDF files, and image files. It provides an area on the server that both confers a unique identity on the application components and facilitates the sharing of components across applications in an organized manner. This construct also simplifies the process of moving a user application from one platform to another and of deploying PC-developed applications.

These components are physically grouped together on an application-by-application basis for run-time execution. This physical grouping can be within an application under a common root or a mapping to an application anywhere in the file system. The physical application or mapped name is referred to as the application name in this document. A comprehensive set of application (APP) commands are provided to control/manipulate the application components, as well as to facilitate applications that can be written and deployed to any platform.

The physical location of an application and its components is determined by a configuration parameter called approot. This parameter is set at installation time and stored in the server configuration file, edaserve.cfg. The default value is dependent on the platform, relative to the install ID home directory, where applicable, as indicated in the following chart.

Application directories can be nested. A nested application directory is an application created within a higher-level application. For more information. see *Nested Application Directories* on page 69.

You can also create a home application directory for each user. Providing a user home application gives each user a place where he has full control to create, change, and run his applications. For more information, see *Home Application Directories for Users* on page 73.

The various operating systems on which the product runs have their own behaviors for physical files and how directories or components are referenced. In addition, some are case sensitive and some are not. For example, on Windows, the files abc and ABC refer to the same file, regardless of how it is stored on disk (aBc, for example). However, on UNIX they are all different files. Additionally, z/OS under PDS deployment and OpenVMS ODS2 make file names uppercase when they are actually saved but they can generally also be referenced in lowercase or mixed case after they have been created. Some operating systems also allow spaces in file names and some do not. To co-exist within the various platforms on which the product runs, the product rules are to either always use APP commands or product tools such as the Web Console to create apps for application files (which will create them in the appropriate case), or to use lowercase names with external tools (such as mkdir myapp and vi mytest.fex), as they will save appropriately and work within the APP framework. Additionally, spaces in file names are not allowed. The use of external tools is also discouraged, as they may not act in the same way as internal tools.

Note:

- Where directories are referenced in the chart, lowercase application directories are created below the approot value. Actual application names must be lowercase and must not contain spaces. For z/OS PDS deployment, data sets are created for each component type using the approot value as the high-level qualifier.

- Some server features will not work without nested applications enabled. Among those are home application and file upload.

Platform	Default Value for approot
z/OS PDS Deployment	*approotvalue.appname.component_type*
z/OS HFS Deployment	.../ibi/apps
UNIX	.../ibi/apps
Linux	.../ibi/apps
Windows	...\ibi\apps
IBM i (formerly known as i5/OS)	.../ibi/apps
OpenVMS	[.IBI.APPS]

Two applications are provided during installation: a default application called *baseapp* and an application in which you can generate legacy sample files called *ibisamp*. In addition, when you connect to the server, a temporary directory called foccache is added as the first directory in the search path. When you want to be able to reuse data within the same browser session, you can store the data in the form of a HOLD, SAVE, or SAVB file in the foccache directory. As long as the browser session remains active, the files stored in the foccache directory can be referenced in requests.

Access to a particular application component can be explicit or implicit. Implicit access is dependent upon the search path in effect at the time of execution. The search path always includes the default application, baseapp. There is no need to explicitly declare this application.

You can change the search path from the Web Console, from the Data Management Console, or from within your application code. You can also change the search path temporarily to add application names to the beginning or end of an existing search path. APP commands are described briefly in *Application Commands Overview* on page 90 and in detail later in this chapter.

In addition to explicit APP names under APPROOT or an APP MAP command, which might look like *myapp/myproc*, there are also special reference names for internal locations that may be useful. They are:

Reference Name	Description
_edatemp	Current directory location. For server use this is the EDACONF edatemp agent directory (that is, ts000001, where 000001 is tscomid of the agent). For PDS Deployment, this is a temporary HFS location. For edastart -t, -x and -f uses, this is the current user directory.
_edahome	EDAHOME installation location.
_edaconf	EDACONF configuration location.

While the EDAHOME and EDACONF catalog locations are internally already on a server search path, special handles such as these allow you to explicitly reference names of the form _edahome/catalog/sysapps, which is one of a number of internal catalog tables.

Note: For platforms that support the Universal Naming Convention (UNC), you can specify a network drive for approot. The UNC must be:

- Minimally one folder below the initial shared location.
- Not contain spaces. For example:

```
\\mynode\myshare\accnting
```

Generating Samples and Tutorials

You can generate several types of sample files and save them in an application in order to run sample procedures and test features and configurations.

The Web Console Applications page has a button named *Tutorials*, which you can click to open the *Create Tutorial Framework* page. You can also right-click an application folder and select *New*, then *Tutorials*, from the context menu.

Following is the list of available tutorials.

- WebFOCUS - Retail Demo
- WebFOCUS - Express Demo

- WebFOCUS - State Population Demo
- WebFOCUS - Star Schema with Variable Data
- WebFOCUS - Custom SQL Security Provider
- DataMigrator - General
- DataMigrator - Iterator
- DataMigrator - File Listener
- DataMigrator - Star Schema
- Create Legacy Sample Tables and Files

In the past, a static application directory named IBISAMP was created and populated with a number of files. Additionally, there were Web Console and Data Management Console (DMC) folder options for selecting and creating items to be used with those products.

The IBISAMP folder is still created as part of the installation process, but it is not populated. You can create those legacy sample files by choosing *Create Legacy Sample Tables and Files* from the Tutorial drop-down list.

Because the change is implemented at the folder level, you can choose to continue using IBISAMP as the location for creating the legacy files, or you can select any other folder. The drop-down option for Legacy samples creates most, but not all, of the sample files that used to be in IBISAMP. Some of the prior files were actually DataMigrator-related, and those have been moved to their own tutorials.

Under z/OS PDS Deployment there are some additional restrictions. The z/OS PDS JSCOM Listener option must have been selected at installation time. The PDS JSCOM option also co-installs some HFS source files that are used by the various create options. If the HFS files for a z/OS PDS Deployment are not present, request to create a tutorial will produce a specific *HFS files not found* message. Note that not all tutorials are implemented for z/OS, and those will produce a *not implemented* message, if selected.

Procedures and Metadata on the Application Tree

The Applications menu groups all application files, including procedures, synonyms, HTML files, data flows, user functions, and other files on a single application tree. By default, files in an application are listed in order of file type. The file type is indicated by a unique icon for each type that displays to the left of the file name in the tree. However, you can sort or filter the list to generate custom views of each application.

Right-clicking an item brings up a context menu. The items you see on the context menu depend on your role and the type of file. Some items are only available to administrators or other users with administrator privileges. Some of the items on the context menu provide shortcuts to items on the *Workspace* or *My Console* menu.

Reference: Application Tree Icons

The following table lists the icons displayed for each type of file on the application tree.

Icon	Type of File
	Closed application directory
	Opened application directory
Metadata	
	Synonym (.mas)
	Cluster synonym (.mas)
	Business View synonym (.mas)
Procedures	
	Stored Procedure (.fex)
	DataMigrator Flow (.fex)
	DataMigrator Flow with IUD (.fex)
	DBMS SQL Flow (.fex)
	Direct Load Flow (.fex)
	Direct Load Flow with IUD (.fex)
	User Function (.fex)

Icon	Type of File
	Scheduled Only (.fex)
Documents	
	HTML file (.htm, .html, .shtml, .htt, .mht, .mhtml, .cfm, .tpl, .hta, .htb)
	Microsoft Excel® Document (.xls, .xlsx, .xlsb, .xht, .xltx, .xlsm, .xltm)
	Adobe Acrobat Document (.pdf, .ai)
	XML Document (.xml, .wsd, .xsd, .wsdl, .mxml, .gcl, .xul, .dtd, .xsl, .xslt, .axl)
	JavaScript Object Notation (.json)
	Microsoft PowerPoint® Document (.ppt, .pptx, .pptm)
	Microsoft Word® Document (.doc, .docx, .docm, .dot, .dotx, .dotm)
Other	
	SQL Script (.sql)
	Data File (.foc, .ftm, .dat, .txt, .csv, .tab, .bdat, .data, .tmp)
	Custom Page (.wcpg)
	Graphics File (.jpg, .jpeg, .jpe, .svg, .jfif, .tif, .tiff, .ico, .gif, .bmp, .png)
	Style File (.sty, .focstyle, .css)
	Archive File (.zip, .rar, .tar, .jar, .war)
	Script File (.js, .jcs, .omi, .cbl, .hti, .jji, .vbs, .sh, .ctl, .bat, .t3i, .jcl, .ps)
	Log/Trace File (.log, .trc, .hto, .t3o, .sta, .msg)

Icon	Type of File
	Configuration File (.cfg, .ini, .prf, .err, .nls)
	Language File (.lng)
	Other Document (.prn, .dif, .fmu, .wp, .ifp, .lzx, .syl, .dmc, .tdl, .bst, .adr, .eps, .swf, .as, .trf, .idx)
	MAINTAIN File (.mpt, .mnt, .fcm, .wfm, .wri, .wxi)

Reference: **Context Menu Options for the Application Directories Tree**

The following options are available when you right-click the top level of the Application Directories tree:

- **New.** Enables you to create new application files. The available context menu options are:
  - **Procedure.** Opens a text box in which you can create a new FOCEXEC as text.
  - **Synonym (Create or Update).** Opens a window that enables you to configure an adapter or create a synonym.
  - **Custom Page.** Enables you to create pages that monitor the various services available from the *Workspace* folder, as well as run procedures.
  - **File.** Opens a text box in which you can create a new file as text.
  - **Application Directory.** Enables you to create a new application directory folder.
  - **Upload Data.** Enables you to upload a file to an application directory folder.
  - **Upload Image Files.** Enables you to upload image files to an application directory
- **Filter.** Enables you to customize the items that display on the application tree. Selecting any item on the context menu opens the Filter Applications Tree screen. For more information, see *Filtering the Application Tree* on page 81.
- **Search Files.** Enables you to find files bases on a range of search criteria. For more information, see *Searching for Files* on page 87.

- **Sort by.** Enables you to sort application files by Name, Size, Type (the default), or date Modified. For more information, see *Sorting the Application Tree* on page 88.

  If you select Sort by Size, the size in bytes of each file is indicated next to its name on the Application tree.

  If you select Sort by Modified, the date each file was last modified is indicated next to its name on the Application tree.

- **Application Preferences.** Enables you to customize the tree display. The available options are:

  - **Show sub-directories last.** Shows subdirectories at the bottom of the tree.

  - **Show Applications not in Path.** Adds an Inactive Directories tree listing all of the applications not in the path to the Application Tree. When the Inactive Directories tree is displayed, the context menu option changes to Hide Applications not in Path.

    Note that this option is only available for the server administrator and users with server administrator privileges. Users with these privileges can manage (add, delete, copy, move, or modify files) directories that are not in the Application Path.

  - **Show Statistics on Applications Tree.** Shows the number of files in the directory when the mouse hovers over an application directory.

  - **Show Descriptions on Applications Tree.** Shows the file description when the mouse hovers over a file.

  - **Show Application Profile.** Shows the current application profile.

  - **Number of files per page on Application Tree.** Enables you to separate the contents of a folder into pages, displaying a *Next* icon to page to additional files. The default is zero (0), which does not separate the contents into pages.

- **Application Settings.** Opens the Application Settings pane.

- **Manage External Repository.** Has the following options.

  - **Manage SQL Repository.** Provides options to create and manage SQL repositories, allowing you to store the contents of an application folder in an SQL database.

  - **Manage REST Repository.** Provides options to create or list and delete a REST repository.

- **Application Path.** Opens the Application Path Configuration page under the My Console menu while leaving the Application Directories tree open in the left pane.

- **Create My Home.** Creates the home application directory of the user. (DataMigrator or Managed Reporting license required.)
- **Schedule/E-mail.** Enables you to view or manage events while leaving the Application tree open in the left pane. You can select Scheduler Agents or Scheduled Events.
- **Log and Statistics.** Enable you to view various logs and statistics while leaving the Application tree open in the left pane.
- **Impact Analysis.** Reports on your use of synonyms, columns, and procedures. The choices are Synonyms by Procedure, Procedures by Synonym, Columns by Procedure, and Procedures by Column. In addition, you can select a Summary or Detailed flow report.
- **Tutorials.** Opens the Create Tutorial Framework page where you can create samples tables and metadata.
- **Data Assist.** Opens a dialog box for selecting a synonym to open in the Data Assist tool that lets you edit and enhance the synonym.
- **SQL Wizard.** Opens the Connect to Data wizard.
- **Upload Wizard.** Opens the Upload wizard for uploading a file to the server and preparing its metadata.

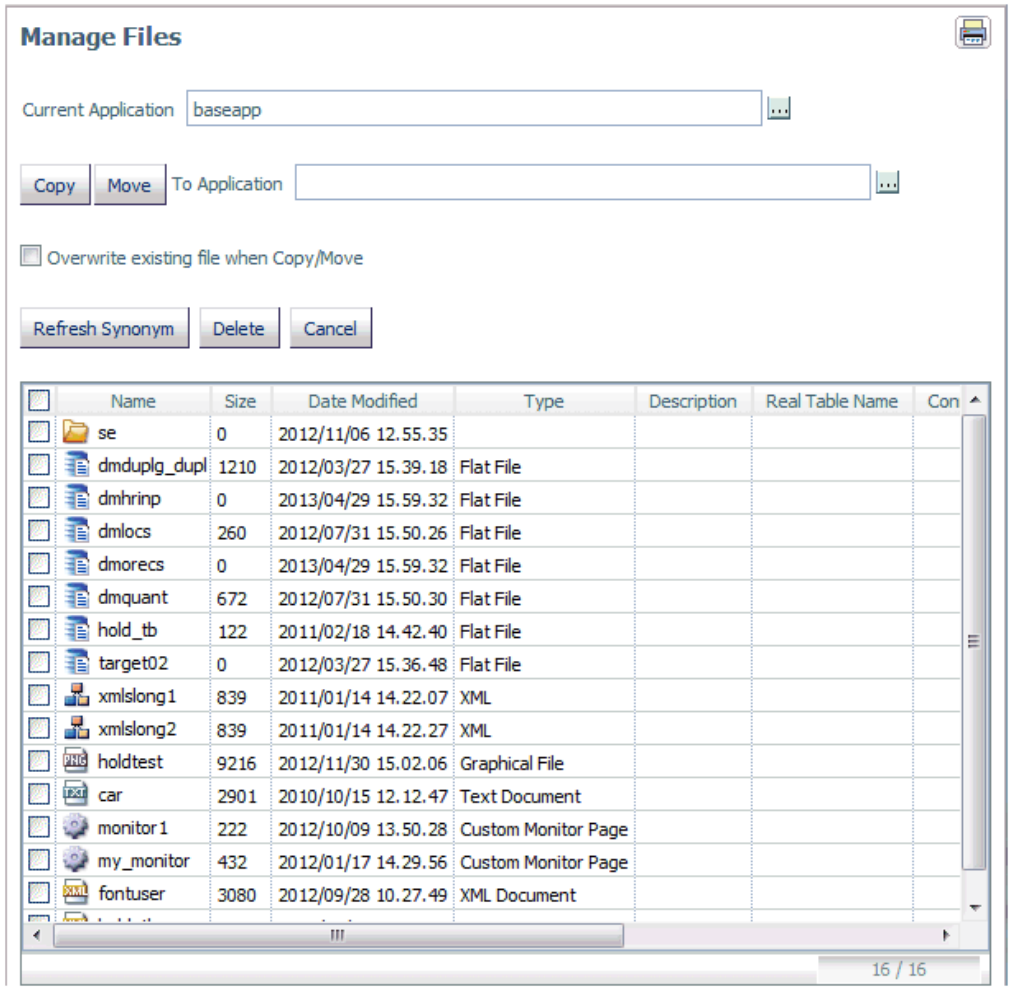
Reference: Context Menu Options for an Application Directory

Like the top level of the Application Directories tree, application directory folders include New, Schedule and E-mail, and Impact Analysis options, as described in *Context Menu Options for the Application Directories Tree* on page 44.

Application directory folders also include the following right-click options:

- **Refresh.** Refreshes the list of files in the application.
- **New.** In the application directory enables you to create a new procedure, synonym, synonym using Data Assist, custom page, text file, application directory , upload data, upload image files, or tutorials.
- **Quick Copy.** Enables you to copy data when you have configured SQL adapters.

- **Manage Files.** Opens the Manage Files pane that lists all files and applications in the selected application with their sizes, date modified, adapter type for synonym files, description, and table name, if applicable:



You can select one or more files and applications and click the following buttons:

- **Copy.** Copies the files to another application directory. Select the target application from the drop-down list.
- **Move.** Moves the files to another application directory. Select the target application from the drop-down list.

- **Refresh Synonym.** Updates synonyms.
- **Delete.** Deletes the files from the application directory.
- **Cancel.** Closes the Manage Files pane.

Note: If you want to overwrite existing files of the same name in the target application, check the *Overwrite existing file when Copy/Move* check box. Copy, move and delete also allow you to include the subfolders of an application. Subfolders are copied, moved, or deleted with all their files and subfolders.

- **Log and Statistics.** Opens the Log and Statistics page where you can view a range of log and statistics reports.
- **Schedule and E-Mail.** Opens the Scheduler Agents or Scheduled Events page for the application.
- **Impact Analysis.** Lets you run impact analysis and flow reports for the application.
- **Copy.** Copies the application directory.
- **Delete.** Deletes the application directory and is only available for application directories that you created.
- **Cut.** Cuts the application directory and is only available for application directories you created.
- **Paste.** Pastes the application directory.
- **Rename.** Opens a dialog box for renaming the application directory.
- **Privileges.** Opens the Manage Privileges pane for the selected application:

- **Properties.** Opens the Properties pane that shows the location of the application directory, number of files in the directory, the date the directory was last modified, and the description. You can also edit the description using the text box provided.

Reference: Context Menu Options for All Files in an Application Directory

All files include the standard right-click options of Cut, Copy, Paste, and Delete. They include the Privileges option, as described in *Context Menu Options for an Application Directory* on page 46.

Files also include the following right-click options:

- **Open.** Opens the file in the editor associated with that type of file (not available for Quick Query, HTML, and Synonym).

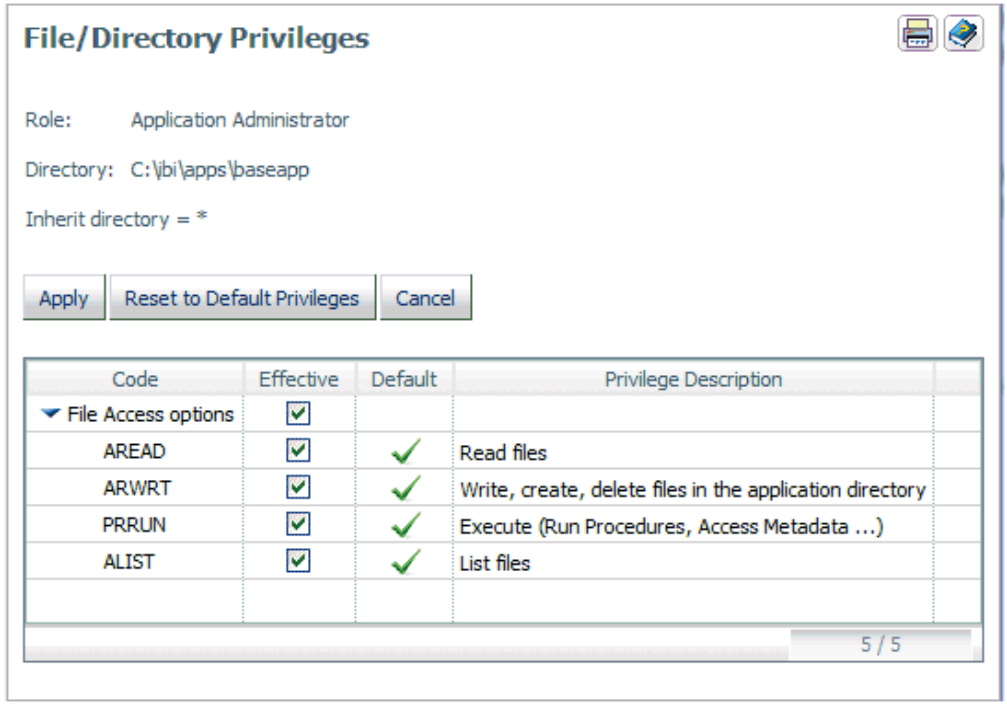
- **Properties.** Opens the Properties pane that shows the location of the file, the date it was last modified, the Job Type, the description, and the privileges associated with the file.
- **Download.** When this option is selected, the file is transferred to the Downloads folder of the user. On Windows, this folder is, by default, the following directory:

  ```
  C:\users\username\Downloads
  ```

  If you right-click a synonym, the shortcut menu gives you a choice of Master File or Access File.
- **Copy.** Copies the application file.
- **Delete.** Deletes the application file and is only available for application directories that you created.
- **Cut.** Cuts the application file and is only available for application directories you created.
- **Rename.** Opens a dialog box for renaming the application file.
- **Privileges.** Opens the Manage Privileges pane for the selected file.

  Each user is listed under the appropriate roles, and the privileges for the role and the individual users are indicated by check marks in the relevant columns.

Right-click a user name to choose *Edit Privileges*. The File/Directory Privileges pane opens.



Check or uncheck the box in the *Effective* column to add or remove the indicated privileges for this user, and click *Apply* to save the changes.

Click *Reset to Default Privileges* to reestablish the default privileges based on the role this user was assigned.

- **Properties.** Opens the Properties pane that shows the location of the application file, the date the file was last modified, the description, and the privileges for the connected user. You can also edit the description using the text box provided.

Reference: Context Menu Options for Stored Procedures

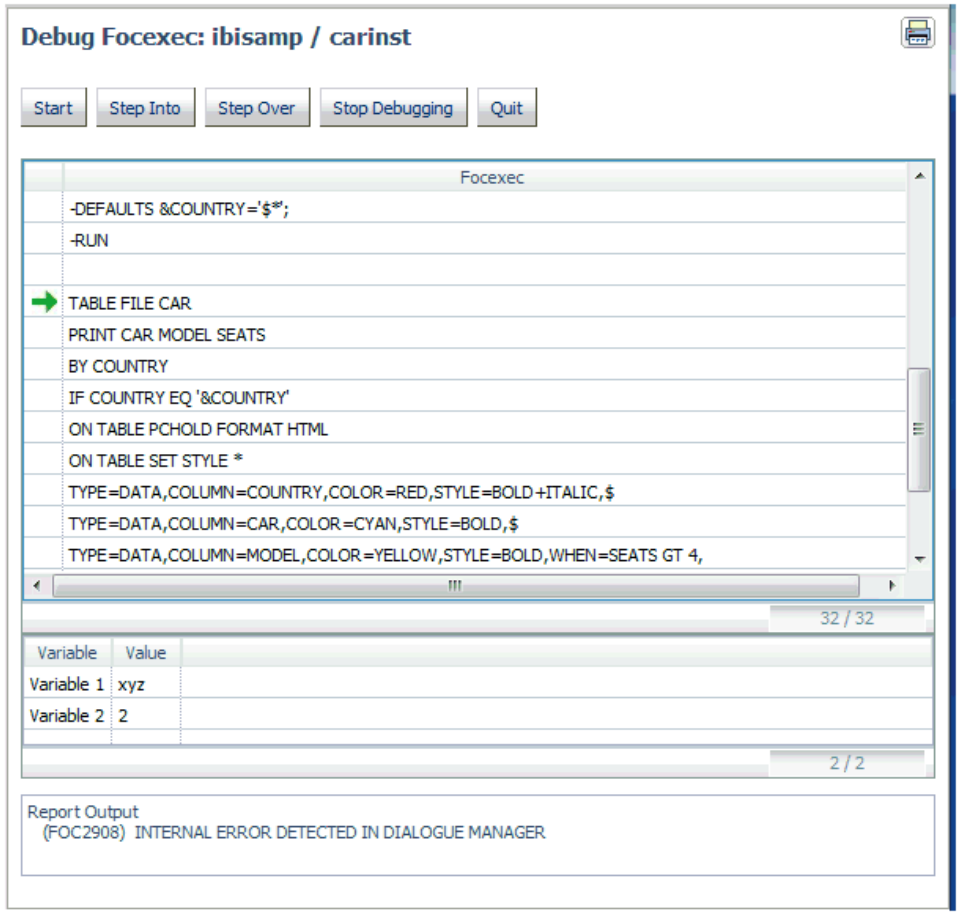
Stored procedures include the Schedule and E-mail, Impact Analysis, and Privileges options, as described in *Context Menu Options for the Application Directories Tree* on page 44.

Stored procedures also include the following right-click options:

- **Run.** Runs the file if it is an executable file.

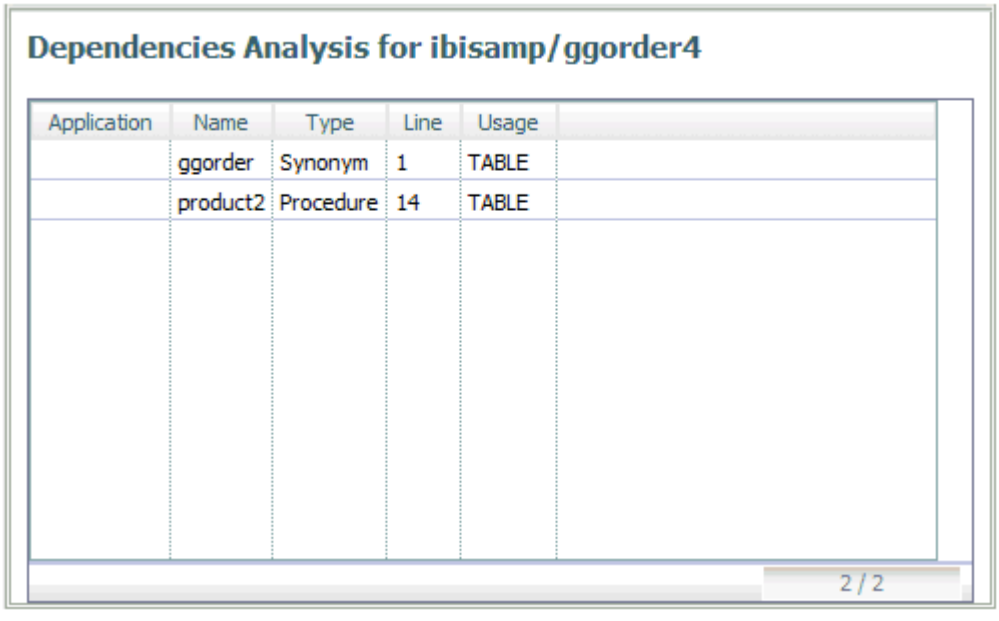
- **Run Changing Defaults.** Lets you enter values for variables and then runs the file if it is an executable file.
- **Run Advanced.** Opens a menu containing the following options:
  - **Submit with E-Mail.** Enables you to set up email notification when the procedure starts and/or completes, and to change default values of amper variables.
  - **Run Stress.** Opens a pane that enables you to set the number of threads, the interval, and the time to keep alive the procedure when it runs. You can also choose to generate statistics and comparisons for this run.
  - **Run Stress Deferred.** Opens a pane that enables you to set the number of threads, the interval, and the time to keep alive the procedure when it runs deferred. You can also choose to generate statistics and comparisons for this run.

- **Debug.** Enable you to run the procedure in debug mode, as shown in the following image. You can step through the code, viewing the result for each line.



- **Logs.** Provides the following options:
  - **Last Log.** Displays the log from the last time the procedure was run.
  - **Last Output.** Displays the output from the last time the procedure was run.
  - **Log and Statistics.** Opens the Logs and Statistics page where you can view a range of log and statistics reports.
- **Analysis Reports.** Provides the following options:
  - **Impact Analysis.** Reports where this procedure is found and its properties.

- **Dependencies Analysis.** Displays the Dependencies Analysis page, as shown in the following image. It enables you to see the synonym(s) and other procedures that are used by the procedure.



- **Data Lineage.** Reports on the fields in the procedure and their properties.

Reference: Context Menu Options for DataMigrator and Direct Load Flows

DataMigrator and Direct Load Flows include the Schedule and E-mail and Impact Analysis options, as described in *Context Menu Options for the Application Directories Tree* on page 44. They include the Run, Run Advanced, Logs, and Dependencies Analysis options, as described in *Context Menu Options for Stored Procedures* on page 51.

DataMigrator and Direct Load Flows also include the following options:

- **Submit.** Submits the flow for processing.
- **Flow Report.** Displays details about the flow, including source and target information, load options, SQL select statements, transformations, execution properties, and record logging.

Reference: Context Menu Options for Quick Queries

Quick Queries include the Schedule and E-mail and Impact Analysis options, as described in *Context Menu Options for the Application Directories Tree* on page 44. They also include the Run, Run Advanced, Submit with Options, and Logs options, as described *Context Menu Options for Stored Procedures* on page 51.

Reference: Context Menu Options for User Functions

In addition to the standard file options (Cut, Copy, Paste, Delete, and Properties), User Functions include the Test right-click option, which tests the function.

Reference: Context Menu Options for HTML Files

In addition to the standard file options (Cut, Copy, Paste, Delete, and Properties), HTML files include the Run and Logs options, as described in *Context Menu Options for Stored Procedures* on page 51.

HTML files also include the following options:

- **Edit.** Enables you to edit the HTML code.
- **View.** Enables you to view how the file displays in a browser.

Reference: Context Menu Options for Custom Pages

Custom Pages include the standard file options (Open, Cut, Copy, Paste, Delete, Privileges, and Properties). Custom Pages also include an Edit option.

Reference: Context Menu Options for Synonyms

Synonyms include the Impact Analysis option, as described in *Context Menu Options for the Application Directories Tree* on page 44. They include the Quick Copy option, as described in *Context Menu Options for an Application Directory* on page 46. They also include the Dependencies Analysis option, as described *Context Menu Options for Stored Procedures* on page 51.

Synonyms also include the following options:

- **Open.** Opens the synonym in Data Assist.
- **Edit as Text.** Opens a Master File in a text editor.
- **Edit Access File as Text.** Opens an Access File in a text editor.
- **Sample Data.** Runs a sample report against a synonym.

- **Data Profiling.** Runs a report against a synonym listing its segments, fields, and field information, with a drill-down report to field values.
- **Metadata Management.** Provides options to create a cluster synonym and to prepare translation files for the metadata in the synonym.
- **Data Management.** Enables you to recreate a DBMS table, drop a table, insert sample data, delete all data, and show/modify data.
- **Quick Copy.** Opens the Quick Copy page where you can specify the target parameters for the new synonym and table.
- **Custom Copy.** Opens Data Assist for editing and enhancing the synonym.
- **Impact Analysis.** Reports where this synonym is found and its properties.
- **Dependencies Analysis.** Displays the Dependencies Analysis page. It enables you to the applications and synonyms that contain or have references to this synonym.

Managing Applications and Paths

Path management tasks are available from the Applications page, which is accessed by clicking *Applications* on the Web Console menu bar. The Application Directories tree displays is on the navigation pane. The ribbon includes a set of icons that provide the quickest way to initiate path management tasks. Some tasks are available from the menu bar at the top of the application tree pane.

A single click on one of the following icons is enough to get you started.

- - Hide and Show tree

  Toggles the navigation pane on and off, allowing one or both panes to occupy the screen.

  Note that these icons are displayed in any context in which you might wish to toggle the navigation pane on and off.

-

  Opens a Find text box with a Search button (magnifying glass icon), Forward (>) and Backward (<) buttons, and a Search Options button. The search options are Case Sensitive, Contains search item, Starts with search item, Ends with search item, and Show only matched items.

- - Refresh

Depending upon the selected context in the navigation pane, this icon refreshes the synonyms or procedures and HTML files listed in the application tree.

Note that this icon is displayed in any context in which you might wish to refresh the display of objects on the navigation pane.

Note: You can also right-click the Application Directories folder in both the Web Console and the Data Management Console to access the *Application Path* option.

Creating and Mapping Applications

Applications are designed to group related components.

- When you create a new physical application, it becomes available for addition to the search path in a selected profile.
- When you map an application name to an existing physical location outside of approot, the mapped application becomes available for inclusion in the search path of a selected profile.

Applications can be created and mapped in either the Web Console or the Data Management Console.

Application directory names must comply with the following rules:

- The maximum length is 64 characters.
- For multi-part Application directories, the total length is limited to 512 characters.

- Names can contain any valid alphanumeric character except the following:

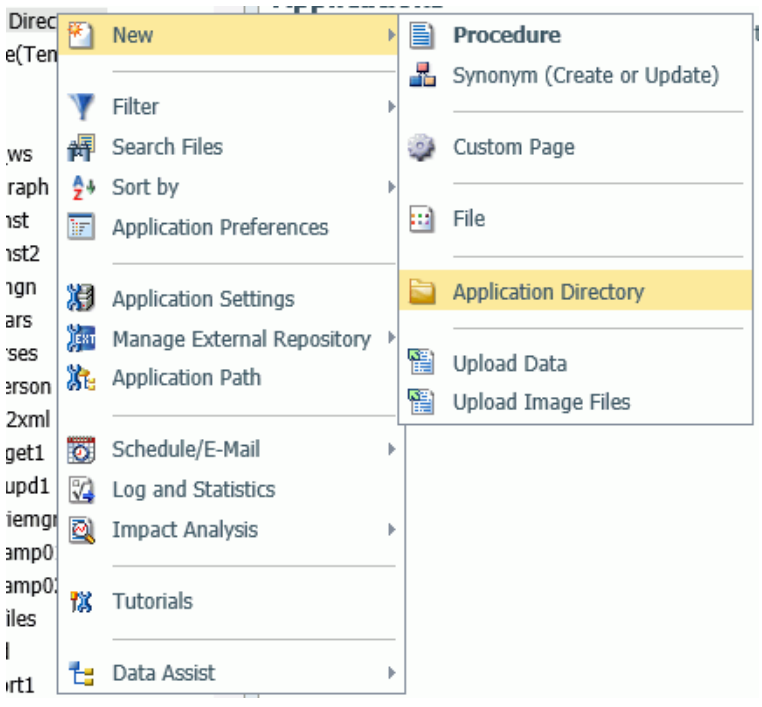
' '	space character	,	comma
*	star character	.	dot
?	question mark	:	colon
<	less than sign	"	double quotation mark
>	greater than sign	'	single quotation mark
&	ampersand	\t	tab
\	backslash	\0	null terminator character
|	vertical bar	/	slash
=	equal sign		

Procedure: How to Create an Application

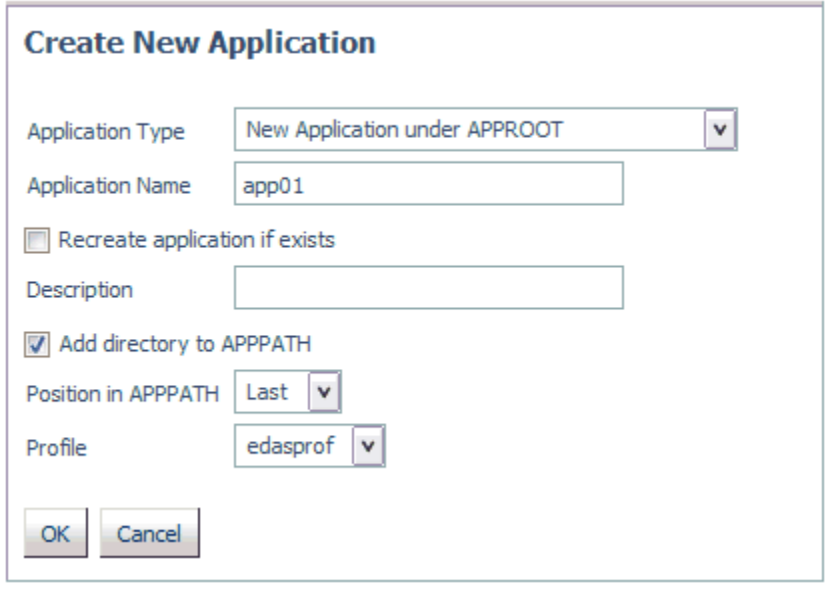
1. From the Web Console menu bar, click *Applications*, or from the Data Management Console, expand the Server node folder.

   On the Web Console, the Applications page opens.

2. Right-click the *Application Directories* folder, select *New*, and then *Application Directory*.



The Create New Application page opens.



3. Use the default *Application Type*, *New Application under APPROOT*.
4. Enter a name in the *Application Name* field.
5. Optionally, select the *Recreate application if exists* check box.

   Warning: Choosing this option will overwrite the existing application and any content in it.
6. Enter a description in the *Description* field.
7. The *Add directory to APPPATH* option is the default. Optionally, you can decide not to add the directory.
8. Select a position from the *Position in APPPATH* drop-down menu. The options are *Last* and *First*. The default is *Last*.
9. Select a profile from the *Profile* drop-down menu. For Server Administrators, the default is *edasprof*. For all other users, the default is the user profile.
10. Click *OK*.

The application is added to the *Application Directories* folder.

Procedure: How to Map an Application to a Physical Directory

You can map an application name to a physical directory anywhere in the file system. This application name can be then used in APP commands.

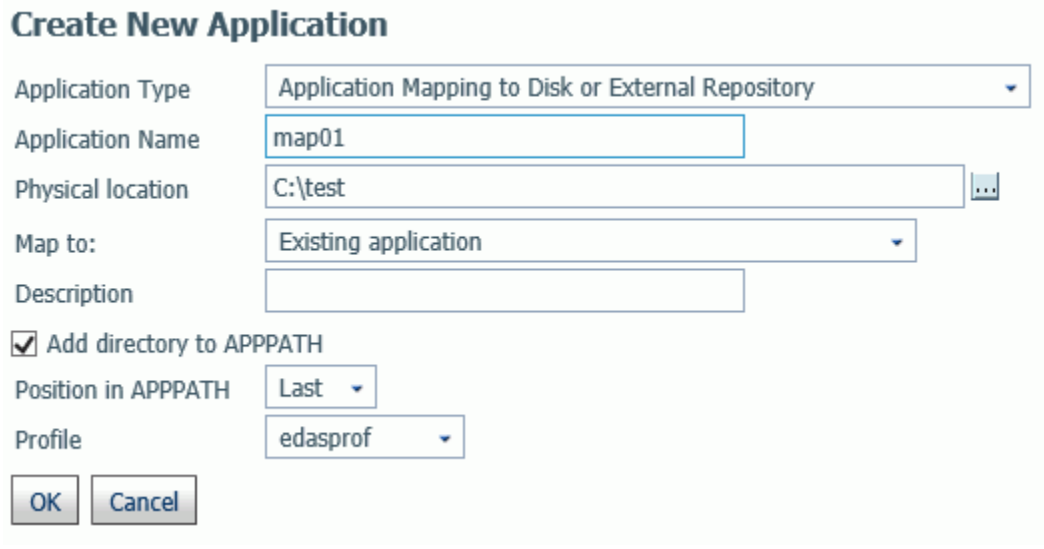
Application mappings can be added and deleted based on profiles from either the Web Console or the DMC.

1. From the Web Console menu bar, click *Applications*, or from the Data Management Console, expand the Server node folder.

   On the Web Console, the Applications page opens.
2. Right-click the *Application Directories* folder, select *New*, and then *Application Directory*.

   The Create New Application page opens.

3. Select *Application Mapping to Disk or SQL Repository* from the *Application Type* drop-down menu.



4. Enter a name in the *Application Name* field.
5. Accept the default Physical location, enter a different location, or click the selector button (...) and navigate to a directory on your file system.

   For platforms other than z/OS HFS Deployment, in addition to using the selector button, you can enter the full path of the physical directory to be mapped, in the format required on your platform. (If there are spaces in the directory you are mapping, you must enclose the entire path in double quotation marks.)

   For z/OS HFS and PDS servers, in addition to using the selector button, you can enter values using the following formats:

   `ext=//DD:ddname[;ext2=//DD:ddname2][...][;extn=//DD:ddnamen]`

   where *extn* are file type extensions.

   `/dir/subdir`

   entered manually or using the selector button.

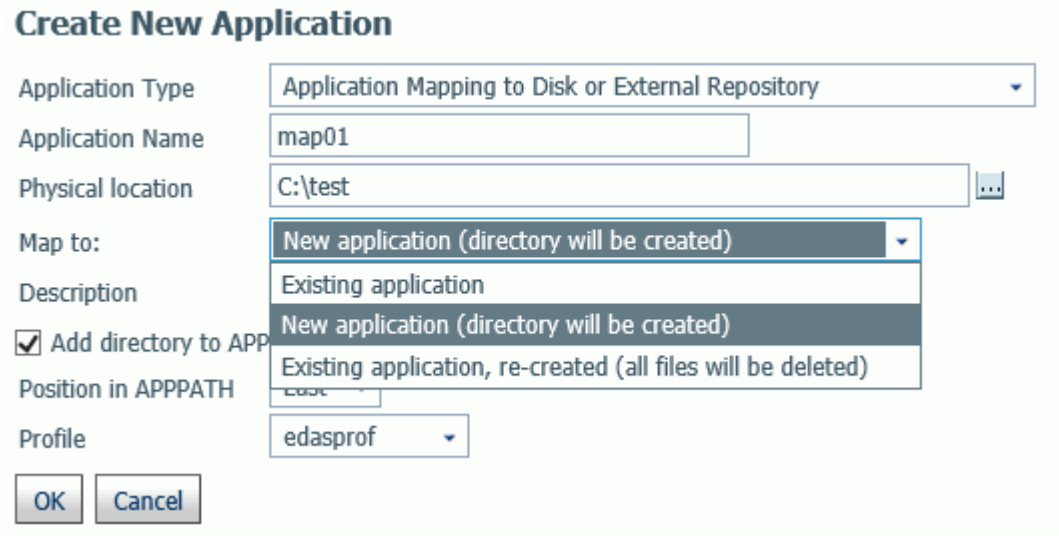
   For z/OS PDS servers, you can also inform the high-level qualifiers of the data set collection that comprise this application. Here is an example of the format:

   `iADMIN.SRV77.MAPAPP`

   where the user has the following datasets (not in approot):

```
IADMIN.SRV77.MAPAPP.FOCEXEC.DATA
IADMIN.SRV77.MAPAPP.MASTER.DATA
IADMIN.SRV77.MAPAPP.ACCESS.DATA
```

6. Select *New application (directory will be created)* from the *Map to* drop-down menu, as shown in the following image.



7. Optionally, enter a description in the *Description* field.
8. *Add directory to APPPATH* is the default. Optionally, you can decide not to add the directory.
9. Select a position from the *Position in APPPATH* drop-down menu. The options are *Last* and *First*. The default is *Last*.
10. Select a profile from the *Profile* drop-down menu.

    For Server Administrators, the default is *edasprof*. For all other users, the default is the user profile.
11. Click *OK*.

The mapping is added to the *Application Directories* folder.

Procedure: How to Delete an Application or the Application Mapping

Applications and application mappings can be deleted from either the Web Console or the DMC.

1. From the Web Console menu bar, click *Applications*, or from the Data Management Console, expand the Server node folder.

On the Web Console, the Applications page opens.

2. Right-click the application or application mappings and select *Delete* or *Delete Mapping*, respectively.

   A confirmation dialog box opens.

3. Click *OK* to delete the application or application mapping.

Note: Deleting an application mapping will delete it from any APP PATH commands that reference it, if the APP PATH command is in the same profile as APP MAP.

Using an SQL Database to Store Application Contents

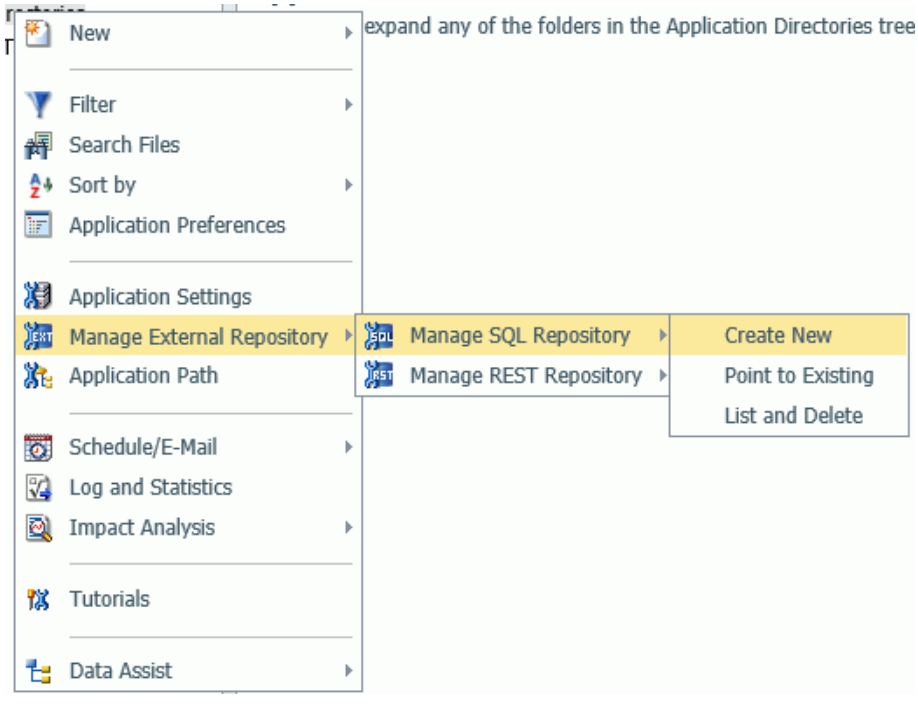
In addition to using a physical location, the contents of an application folder can also be stored in an SQL database. To use an SQL database, you must first create a new SQL Repository. After the repository is created, you can create applications mapped to the SQL Repository and store files there.

Warning: It is advisable to create a separate SQL Connection to use in creating an SQL Repository. If an existing SQL Connection is used, only Server Administrators or users with WSCFG privileges will be able to use the synonyms from this SQL connection in the application. This provides protection for the SQL Repository, preventing unauthorized users (without administrator privileges) from accessing the contents of the SQL Repository through a synonym.

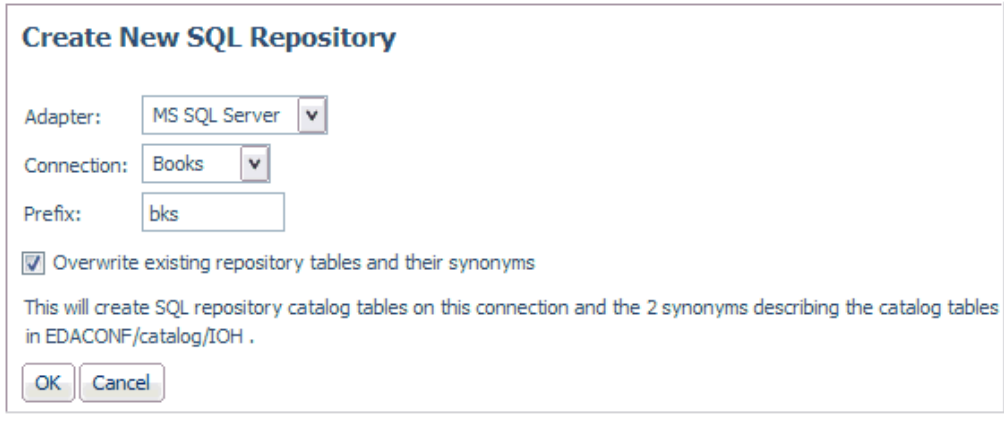
Procedure: How to Create an SQL Repository to Store Applications

You must have an adapter connection configured to an SQL database.

1. From the ribbon, click the *Manage External Repository* icon, select *Manage SQL Repository*, and then select *Create New*, or right-click the *Application Directories* folder in the navigation pane, select *Manage External Repository*, then *Manage SQL Repository*, and then *Create New*, as shown in the following image.

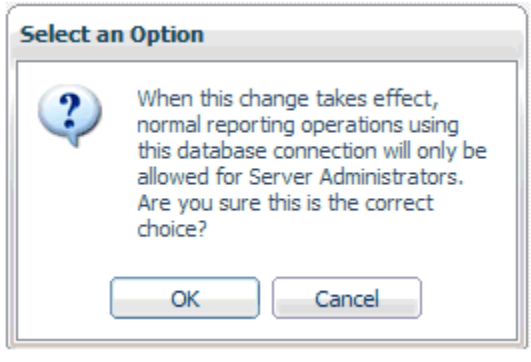


The Create New SQL Repository page opens, as shown in the following image.



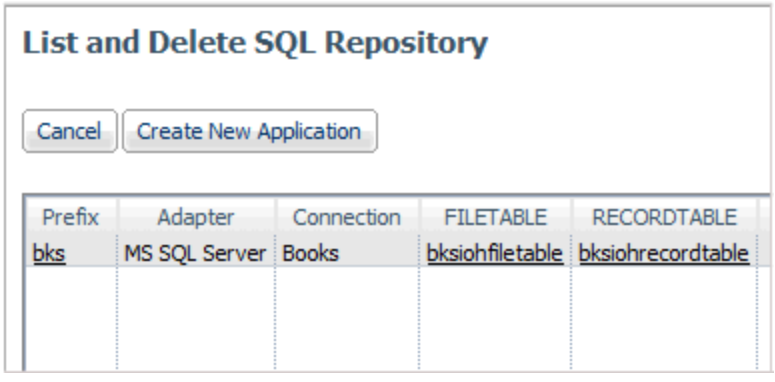
2. Select a configured adapter from the Adapter drop-down menu.
3. Select a connection from the Connection drop-down menu.
4. Enter a prefix in the *Prefix* field.
5. Optionally, select the *Overwrite existing repository tables and their synonyms* check box.
6. Click *OK*.

   A warning message is displayed, as shown in the following image.



7. Click *OK*.

Two SQL Repository catalog tables are created with this connection, as shown in the following image.



The tables are:

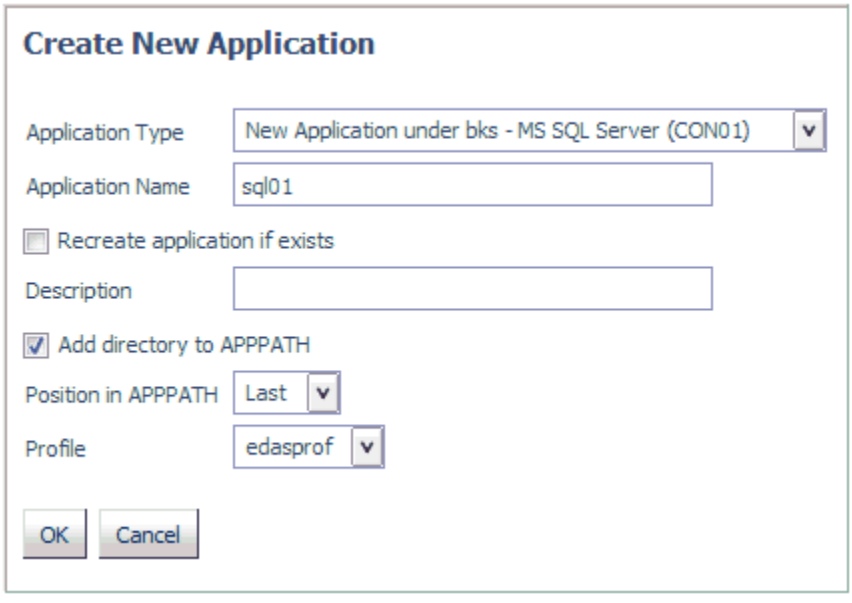
*prefix*IOHFILETABLE

*prefix*IOHRECORDTABLE

Two synonyms describing the catalog tables are also created in EDACONF/catalog/IOH.

8. Optionally, click *Create New Application*.

The Create New Application page opens, as shown in the following image.



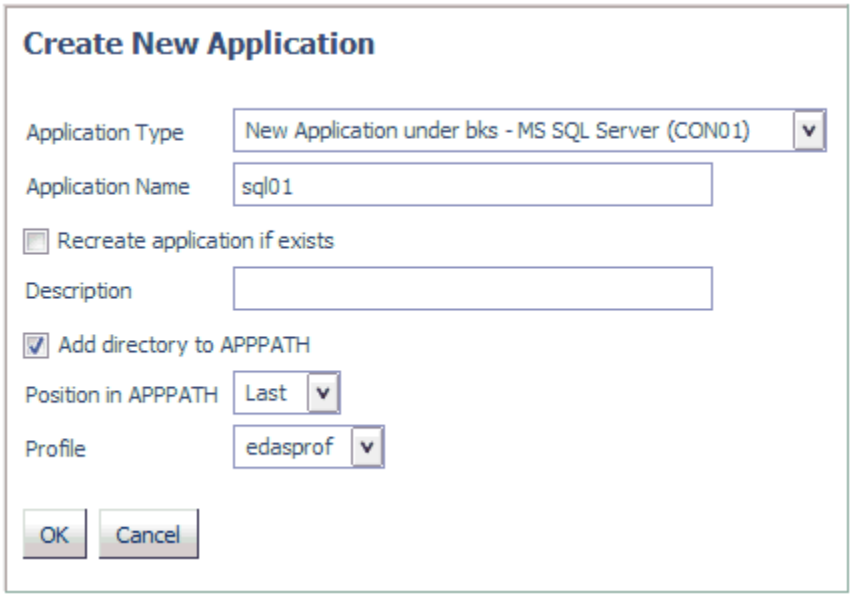
Procedure: How to Create an Application With SQL Content

You must have created an SQL Repository.

1. From the Applications page, right-click the *Application Directories* folder in the navigation pane, select *New*, then *Application Directory*.

   The Create New Application page opens.

2. Select *New Application under repository - DBMS (Connection name)* from the Application Type drop-down menu, as shown in the following image.



Note: The choices will include the repository, adapter type, and connection name. In this example, they are the bks repository, Adapter for Microsoft SQL Server, and CON01 connection.

3. Enter a name in the *Application Name* field.
4. Optionally, select the *Recreate application if exists* check box.

   Warning: Choosing this option will overwrite the existing application and any content in it.
5. Optionally, enter a description in the corresponding field.
6. Optionally, deselect the *Add directory to APPPATH* check box. The application is added to the APPPATH by default.
7. Select a position for the application from the *Position in APPPATH* drop-down menu. The choices are *Last* or *First*. The default value is *Last*.
8. Select a profile from the *Profile* drop-down menu. For server administrators, the default value is *edaprof*. For non-administrators, their user profile is the default value.
9. Click *OK*.

The application is added to the navigation tree. You can now use this application to store procedures, synonyms, data files, and other content.

Note: Repository procedures can be executed, and Master and Access Files stored in a repository can be accessed, during server, group, and user profile execution after all DBMS connections and an APP MAP command to the repository have been executed.

Nested Application Directories

A nested application directory is one created within a higher-level application. The server allows five levels of nested application directories by default. The server must be configured for deeper or unlimited levels.

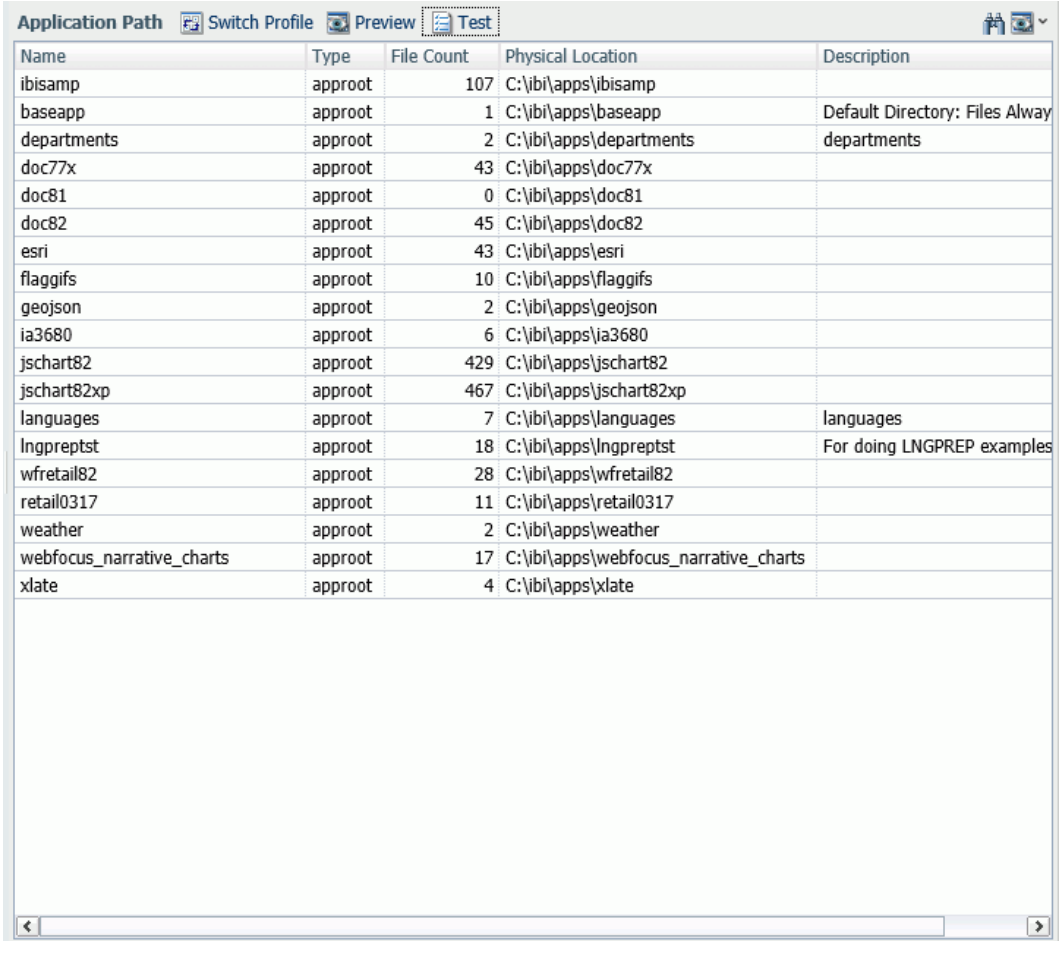
Nested application directories are implicitly added to the application path if the parent directory is on the application path.

For example, the following application tree has a directory named new1 that has a child directory named new2.

The APP PATH command explicitly places new1 on the application path:

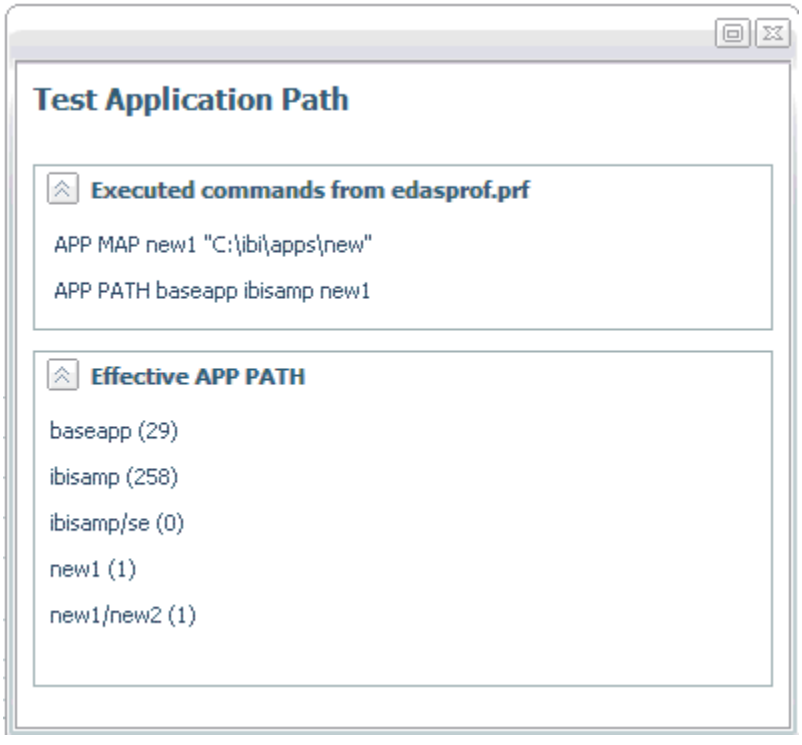
```
APP MAP new1 "C:\ibi\apps\new"
APP PATH baseapp ibisamp new1
```

However, you can test the path to see all of the implicitly added directories by right-clicking the top level of Applications Directories tree and selecting *Application Path*. When the Application Path page opens, click *Test* in the Application Path frame:



Name	Type	File Count	Physical Location	Description
ibisamp	approot	107	C:\ibi\apps\ibisamp	
baseapp	approot	1	C:\ibi\apps\baseapp	Default Directory: Files Alway
departments	approot	2	C:\ibi\apps\departments	departments
doc77x	approot	43	C:\ibi\apps\doc77x	
doc81	approot	0	C:\ibi\apps\doc81	
doc82	approot	45	C:\ibi\apps\doc82	
esri	approot	43	C:\ibi\apps\esri	
flaggifs	approot	10	C:\ibi\apps\flaggifs	
geojson	approot	2	C:\ibi\apps\geojson	
ia3680	approot	6	C:\ibi\apps\ia3680	
jschart82	approot	429	C:\ibi\apps\jschart82	
jschart82xp	approot	467	C:\ibi\apps\jschart82xp	
languages	approot	7	C:\ibi\apps\languages	languages
lngpreptst	approot	18	C:\ibi\apps\lngpreptst	For doing LNGPREP examples
wfretail82	approot	28	C:\ibi\apps\wfretail82	
retail0317	approot	11	C:\ibi\apps\retail0317	
weather	approot	2	C:\ibi\apps\weather	
webfocus_narrative_charts	approot	17	C:\ibi\apps\webfocus_narrative_charts	
xlate	approot	4	C:\ibi\apps\xlate	

The effective application path also includes *new1/new2*:

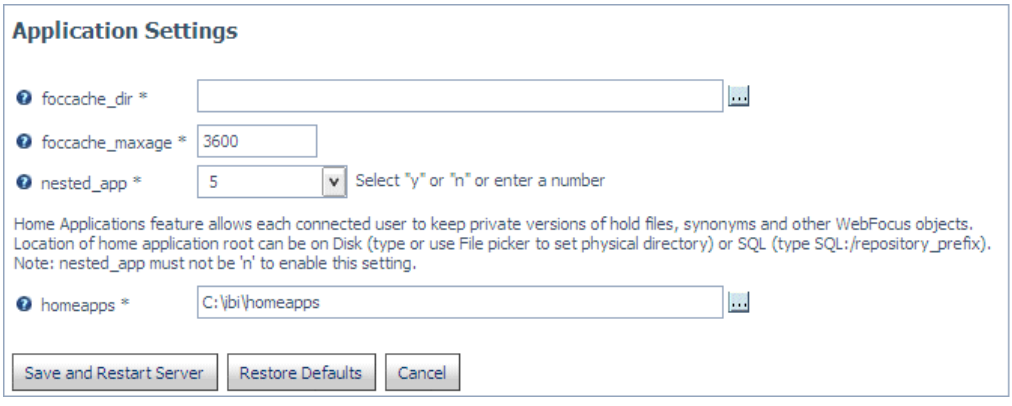


Procedure: How to Set the Level of Nested Application Directories

1. From the menu bar, select *Applications*.
2. From the ribbon, click the *Application Settings* icon, or right-click the *Applications* folder, and select *Application Settings*.

   The Application Settings page opens.

3. Enter the level of nested applications in the *nested_app* field or select *y* from the drop-down menu. The default value is 5. Selecting *y* allows unlimited levels of nested applications.



   Note: For z/OS servers, this setting is only applicable to directory-style applications. It is not applicable to PDS-style applications or to applications mapped as a collection of ddnames.

4. Click the *Save and Restart Server* button.

   After the server restarts, you can create a new application subdirectory by right-clicking an application folder and selecting *New* and then *Application Directory* from the context menu.

Note: Nested applications must be in effect in order to create user home application directories. Some other server features also require nested application directories. For example, when you Upload a data file, the file is initially staged in a temporary upload directory under foccache while you make any edits or enhancements needed to the synonym. Tutorials use nested applications as well.

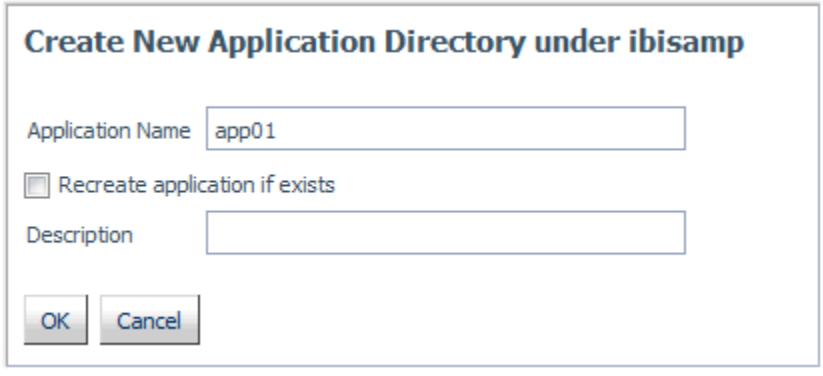
Procedure: How to Create a Nested Application Directory

1. From the Web Console menu bar, click *Applications*, or from the Data Management Console, expand the Server node folder.

   On the Web Console, the Applications page opens.

2. Right-click an application, select *New*, and then *Application Directory*.

The Create New Application Directory page opens.



3. Enter a name in the *Application Name* field.
4. Optionally, enter a description in the *Description* field.
5. Click *OK*.

The nested application is added to the application tree under its parent application folder.

Home Application Directories for Users

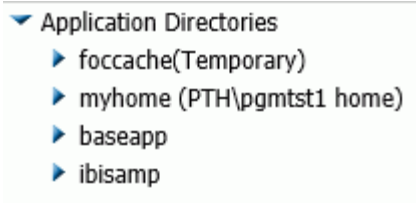
The server administrator can configure the server to allow each user to have a home application directory. Providing a user home application directory gives each user a directory where he has full control to create, change, and run his applications.

The home application for any user who is not a server administrator appears in two places on the Web Console application tree. Both of these applications point to the same physical location, so that they can be referenced in two ways. The two applications are:

- Application *myhome*, which is prepended to APPPATH.
- Application *homeapps*, which is appended to APPPATH and can be expanded to see the user home application with the user ID name, such as *pgmtst1*. The homeapps folder on the application tree is only available to users registered under the server administrator role.

Server administrator users have the myhome application prepended and the homeapps application appended to APPPATH, The homeapps folder can be expanded to show the home applications for all users.

The following image shows both the myhome application and the homeapps application for user *pgmtst1*:



The files created in home applications can be referenced in procedures as myhome/*procname*.fex and homeapps/*pgmtst1*/*procname*.fex

The first type of reference can be ported easily to any user. It enables you to create a common application that utilizes data stored in the home applications of users as myhome/data, so that each user can run the same procedure but get a report based on the data stored in the home application of that user.

The second type of reference enables you to run applications specific to a user, referring to the data and procedure for each user as homeapps/*pgmtst1*/proc1.fex and homeapps/*pgmtst1*/data. This type of reference can be used for testing applications before moving them to common application folders.

Home application directories should be enabled only on secured servers. If the server runs with security OFF, all users have total control of files in all applications, and the home directories will not work as designed. Nested applications must be enabled in order to create user home application directories.

The Server Administrator can monitor and manage home application directories for all users and, therefore, all user home directories are visible and in the path on the Web Console application tree when the connected user has Server Administrator privileges.

When the server is enabled for home applications, user home application directories are not created automatically. Users can create them from the Web Console Application page, or ask their server admin to create them.

Home applications can also be stored in an SQL Repository. You must first create an SQL Repository and configure the application settings *homeapps* parameter to point to the SQL Repository, as described in *Using an SQL Database to Store Application Contents* on page 63.

After a home application directory is created for a user:

- The home application is added to the users application path by the server implicitly. A user cannot remove it from the path using APP commands.

- The home application directory is always first in the users active application path.
- Non-server administrator users will only see their own home application on the Application tree. The server admin or a user with server admin privileges will see all user home applications. The server admin has full privileges to manage user home applications.

Note: This feature is available only for customers that are licensed for Managed Reporting or DataMigrator.

Procedure: How to Manage Home Application Directories

In order to set up home directories for individual users, nested applications must be enabled. They are enabled to five levels by default. The home directory under which user application directories will be created is set during installation. The default is *homeapps*. The homeapps directory can be changed to a different physical location on the Application Setting page. All user home directories will be nested under the home directory.

1. From the menu bar, select *Applications*.
2. From the ribbon, click the *Application Settings* icon, or right-click the *Applications* folder, and select *Application Settings*.

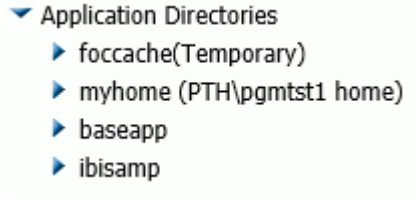
   The Application Settings page opens.
3. Check the location of the *homeapps* directory. You can set the *homeapps* parameter to point to a physical directory or to an existing SQL Repository.
4. Click the *Save and Restart Server* button to implement these changes.

   When the server restarts, go to the *Applications* menu. For a user with Server Administrator privileges, a *users home* folder will appear under the Application Directories tree and will have all of the user home directories.

   Users who do not have server administrator privileges can see their home directory under application *myhome*, which is prepended to APPPATH.

   Server administrator users have the myhome application prepended and the homeapps application appended to APPPATH, The homeapps folder can be expanded to show the home applications for all users.

   The following image shows the myhome application for user *pgmtst1*:



The files created in home applications can be referenced in procedures as myhome/*procname*.fex and homeapps/*pgmtst1*/*procname*.fex

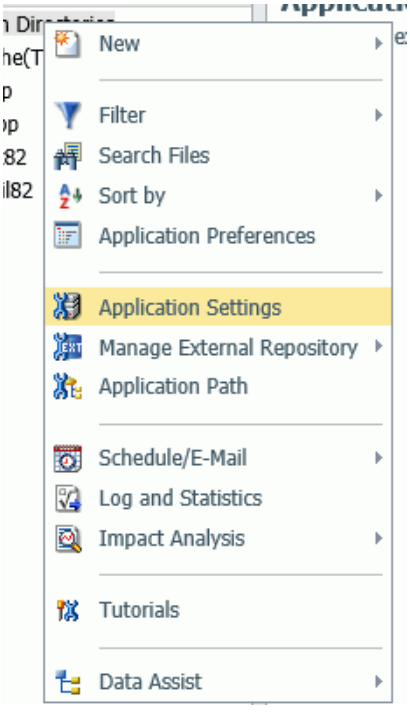
The first type of reference can be ported easily to any user. It enables the server administrator to create a common application that utilizes data stored in the home applications of users as myhome/data, so that each user can run the same procedure but get a report based on the data stored in the home application of that user.

The second type of reference enables the server administrator to run applications specific to a user, referring to the data and procedure for each user as homeapps/*pgmtst1*/proc1.fex and homeapps/*pgmtst1*/data. This type of reference can be used for testing applications before moving them to common application folders.

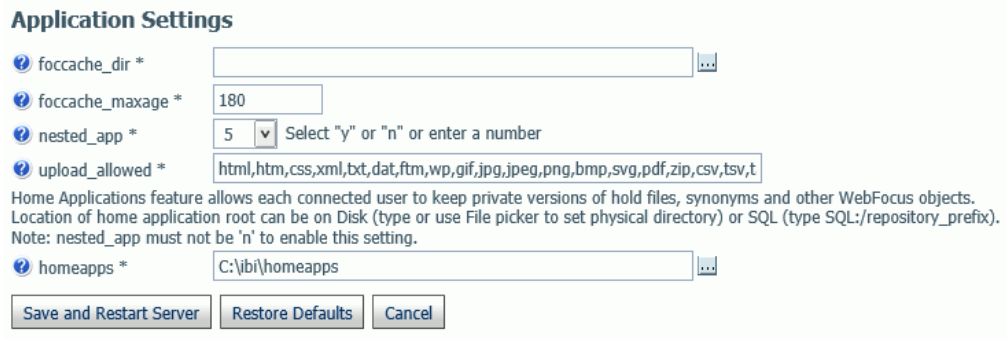
Procedure: How to Store Home Applications in an SQL Repository

You must first create an SQL Repository. For information, see *How to Create an SQL Repository to Store Applications* on page 64.

1. From the ribbon, click the *Application Settings* icon, or right-click the *Application Directories* folder in the navigation pane, and select *Application Settings*, as shown in the following image.



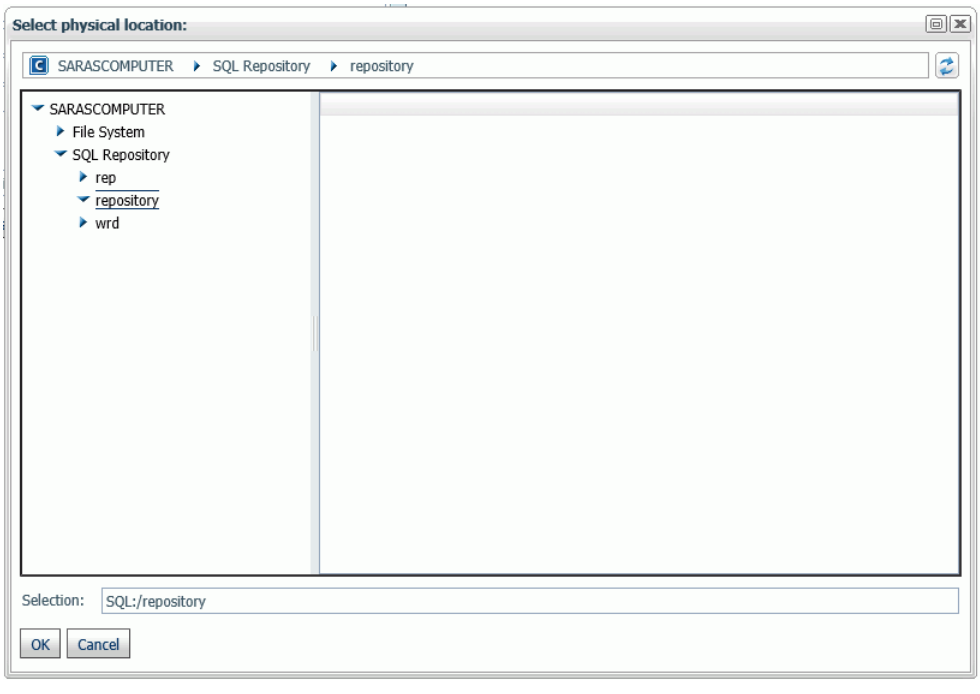
The Application Settings page open, as shown in the following image.



2. Click the selector button (...) next to the homeapps field.

   The Select physical location dialog box opens.

3. Select *SQL Repository*, as shown in the following image.



4. Select a subfolder, and click *OK*.

The subfolder is entered in the *homeapps* field, as shown in the following image.

5. Click *Save and restart Server*.

Home applications that are created will now be stored in the SQL database.

Configuring the Application Path

The applications available for inclusion in the search path are identified by name, type, and physical location. If the *Add directory to APPPATH* check box was selected when the application was created, it was automatically added to the search path. If not, you must *explicitly* add it to your search path.

Note: You can also create profiles from the Application Path page. Profiles are the locations in which the search path is saved.

Procedure: How to Configure the Application Path

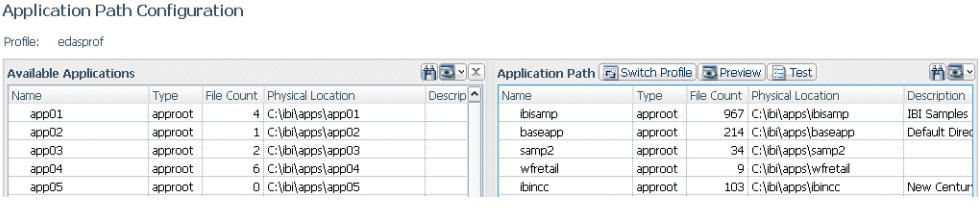
You can configure the Application Path to add or remove applications or mappings from either the Web Console or the DMC.

The User Interface for configuring the application path uses a double list.

You configure the application path from the Applications tab on the Web Console.

1. Click *Application Path* on the ribbon, or right-click the Application Directories tree and select *Application Path* from the shortcut menu.

   The Application Path Configuration page opens as a double-list table. The left pane shows a list of all available applications, the right shows all applications in the active Application Path, as shown in the following image.



2. To add an application to the application path, drag it from the Available Applications list to any position on the Application Path list.

   You can move applications up or down on the path by dragging them up or down on the list.

3. To remove an application from the application path, drag it from the Application Path list to the Available Applications list.

4. When you have finished, Click *Save*.

On both lists, you can

- Search for applications using the Binocular Search.
- Customize the columns displayed, or reset the display to its default, using the View icon.

On the Application Path list, you can:

- Select a different profile.
- Preview the commands in the selected profile.
- Click the *Test* button to view the APP PATH command and the effective application path.
- Click *Save* to save your changes.

The navigation pane is updated.

Note: You can create a new profile on the Application path pane.

Procedure: How to Configure the Application Path in a User, Group, or Role Profile

The Application Path can be configured from either the Web Console or the DMC.

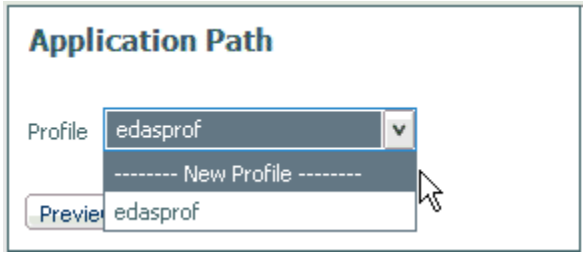
1. From the Web Console menu bar, click *Applications*, or from the Data Management Console, expand the Server node folder.

   On the Web Console, the Applications page opens.

2. From the ribbon, click the *Application Path* icon, or right-click the *Application Directories* folder in the navigation pane, and select *Application Path*.

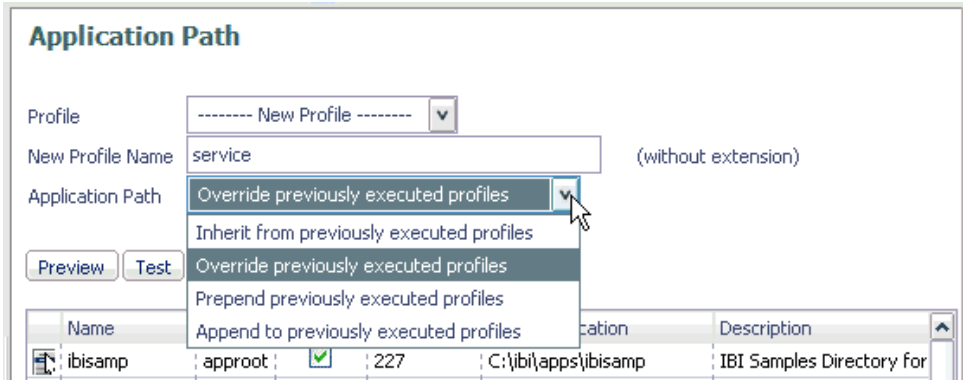
   The Application Path page opens.

3. From the *Profile* drop-down menu, select *New Profile*.



4. Enter a name in the *New Profile Name* field.

5. Select an option from the *Application Path* drop-down menu.



The options are:

- *Inherit from previously executed profiles*
- *Override previously executed profiles*
- *Prepend previously executed profiles*
- *Append to previously executed profiles*

6. Optionally, click *Preview* to see the profile.

7. Click *Save*.

Procedure: How to Edit a Profile

1. From the menu bar, select *Workspace*.
2. On the navigation pane, open the *Configuration Files* and *User/Group Profiles* folders.
3. Right-click the profile and select *Edit*.

   The profile opens in a text editor with its current path displayed.
4. Edit the path information and click the *Save* icon.

Tip: You can also edit a profile search path by selecting and saving configuration options. Follow instructions for configuring the application path.

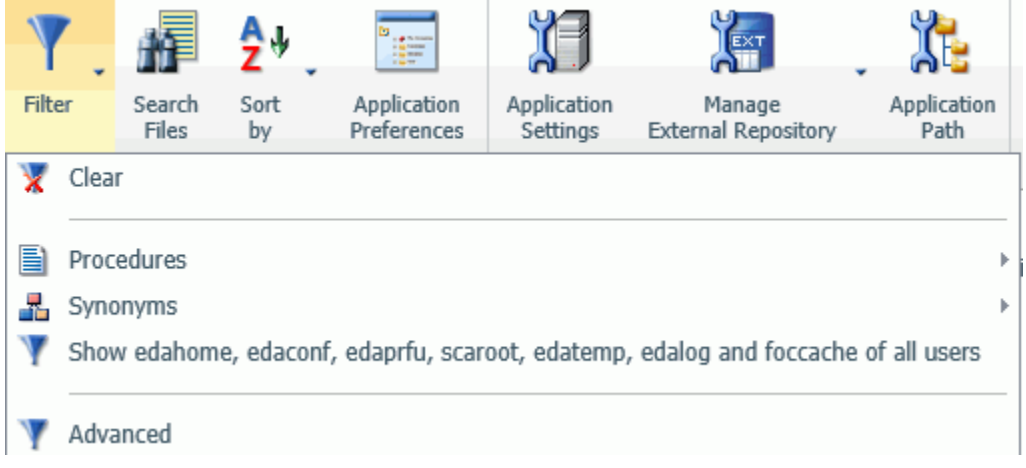
Filtering the Application Tree

Filtering enables you to customize the file listings on the Application tree, displaying only the files you choose. It can be based on file name, location, statistics, type, or any combination of items matching a number of criteria.

Procedure: How to Filter Items on the Application Tree

You can customize the items that display on the Application Tree by filtering. Your filtering selections apply to all applications displayed in the navigation pane.

1. From the Web Console menu bar, click *Applications*.
2. Click the *Filter* icon above the navigation pane, and select *Procedures*, *Synonyms*, or *Advanced*, as shown in the following image.

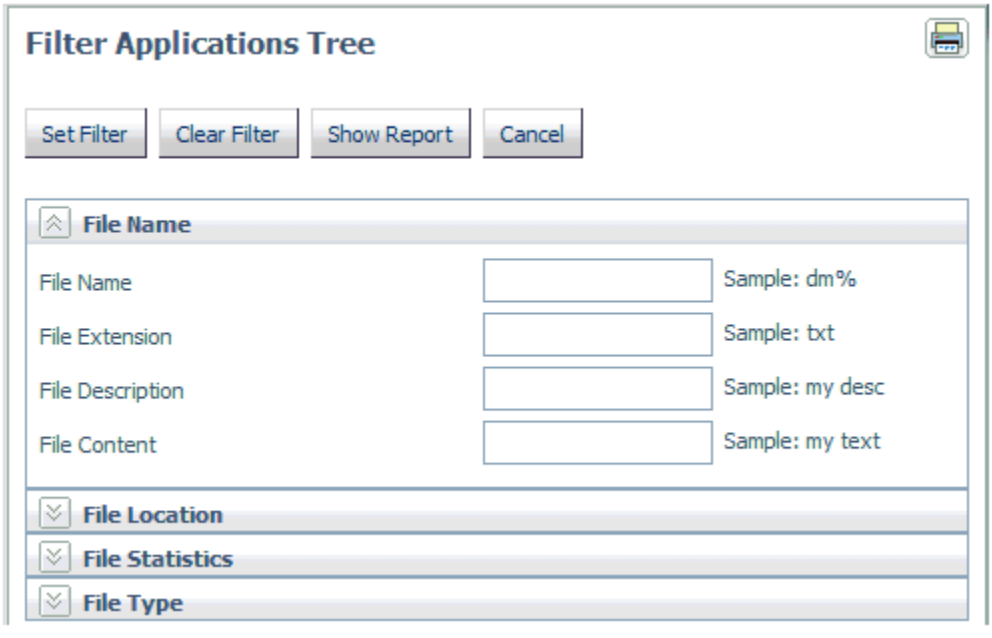


   a. If you choose *Procedures*, you can display all procedures or only scheduled ones.
   b. If you choose *Synonyms*, you can display all synonyms or only cluster or business views.

c. If you choose *Show edahome, edaconf, edaprfu, scaroot, edatemp and foccache of all users*, the server administrator and users with administrator privileges will see these internal locations as applications for all users.

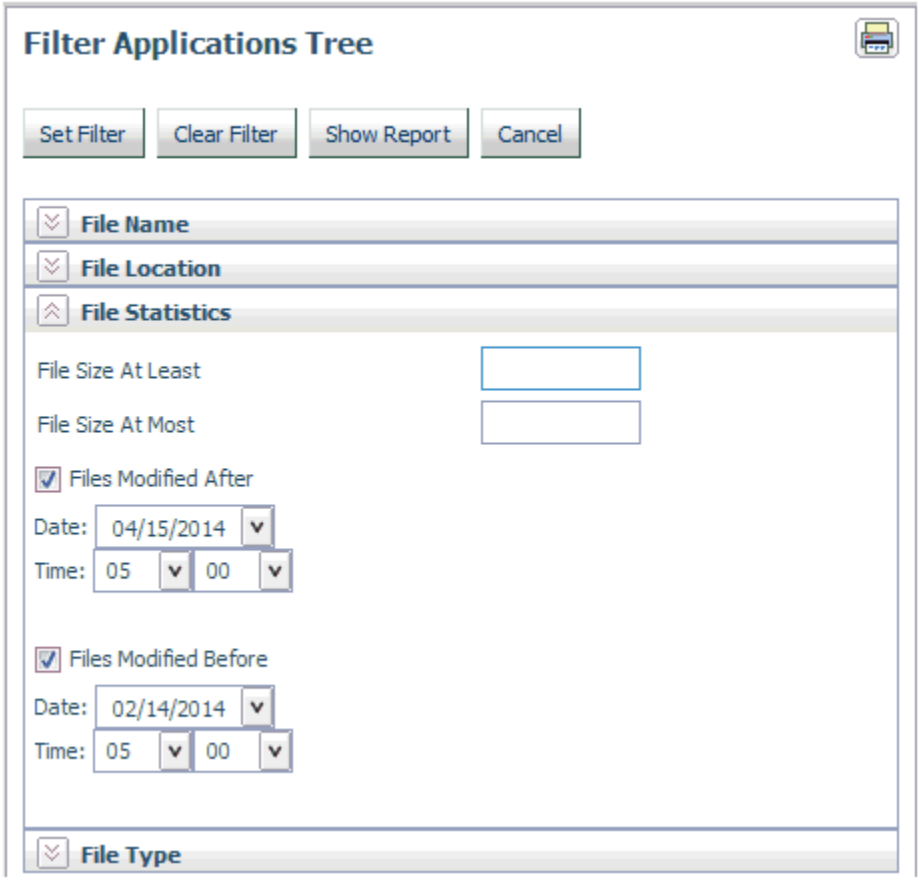
d. If you choose *Advanced*, The Filter Applications Tree page opens

Using the File Name section, you can filter by name, extension, description and content, as shown in the following image. To filter the name, you can use the percent sign (%) as a wildcard character. Specifying e% displays all files whose name begins with the letter *e*.

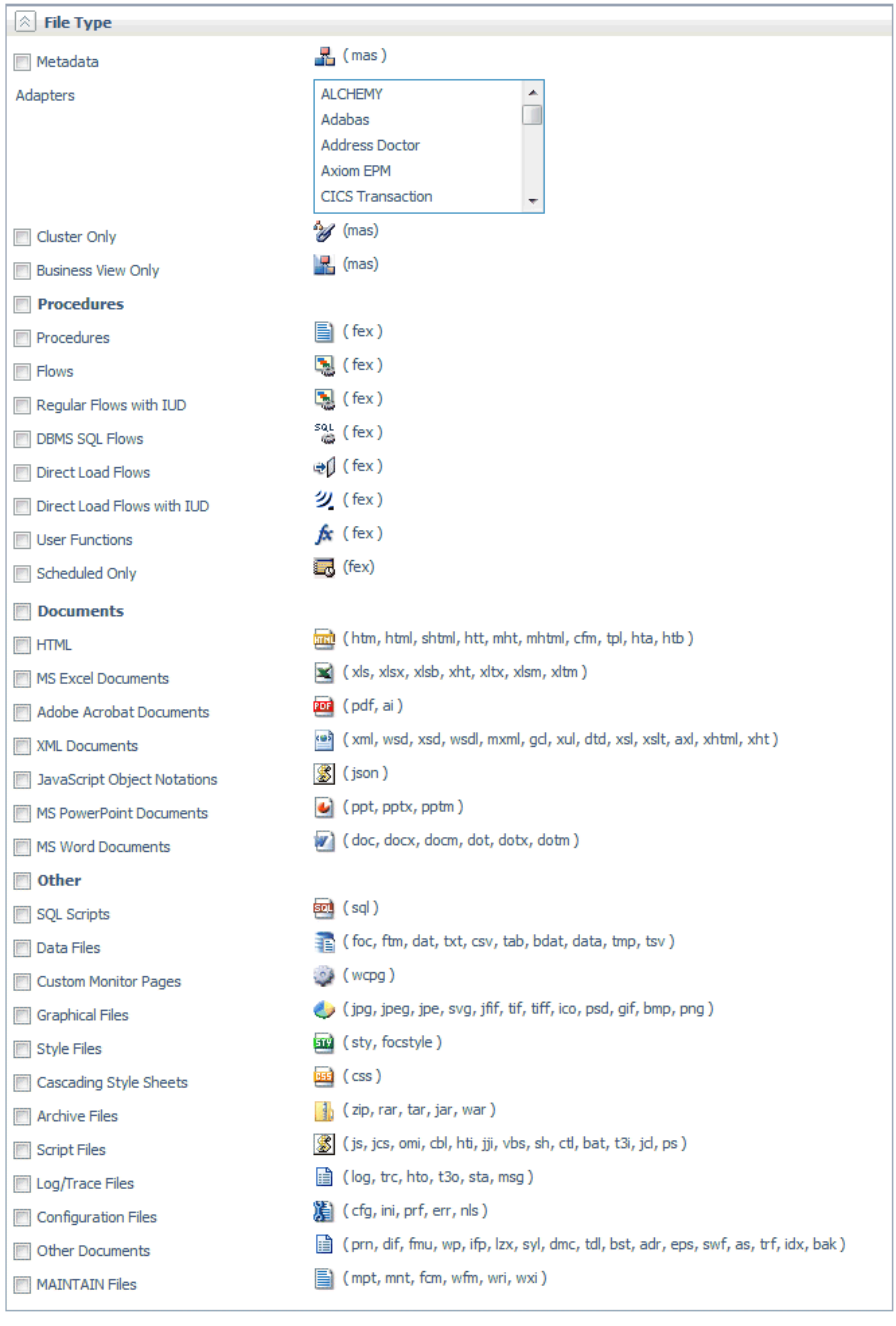


3. Using the File Location section, you can filter by Application Directory, Application Path, Mapped Applications, and Applications under APPROOT, as shown in the following image.

4. Using the File Statistics section, you can filter by file size and modified date, as shown in the following image.

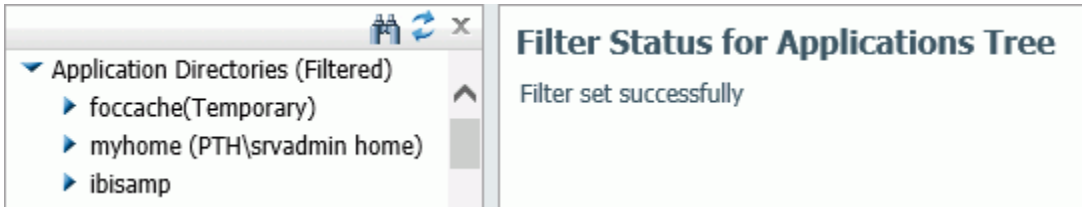


5. Using the File Type section, you can filter by type of file to be included.



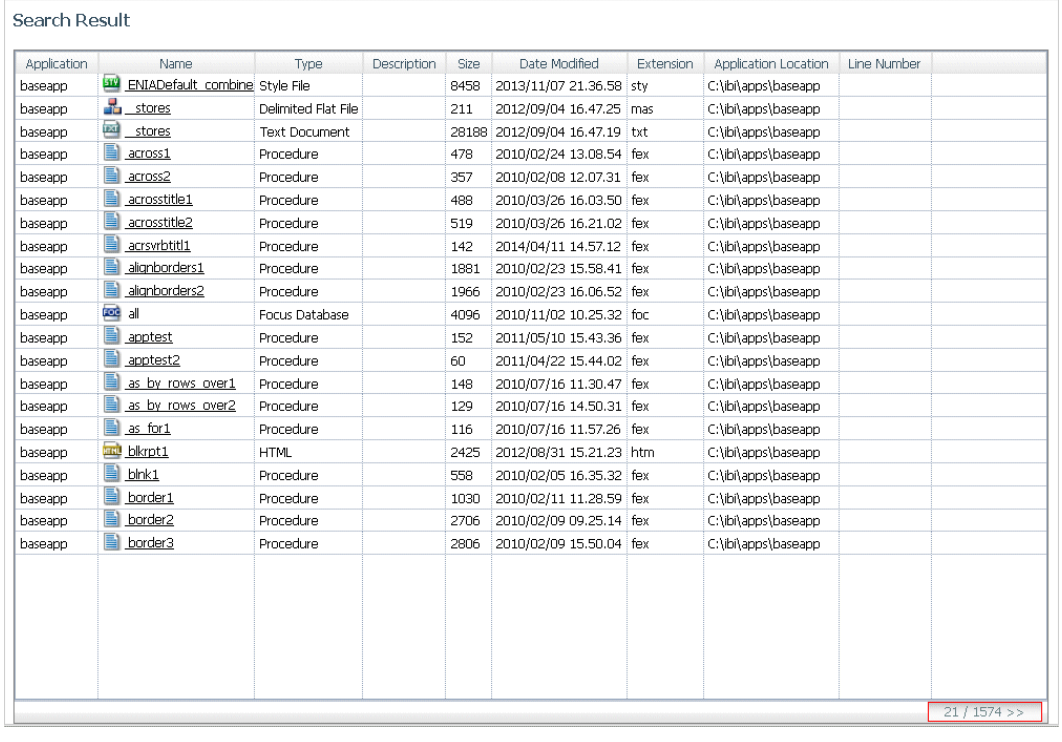
6. Optionally, select an adapter from the *Adapters* drop-down menu. Only synonyms created with that adapter that match the filtering criteria will appear in the tree.
7. Click *Set Filter*. The Filter Status page confirms that the filter was set.

When a filter is applied, the *Application Directories* tree label includes *(Filtered)*, as shown in the following image.



Note: You can remove the filter by clicking the *Clear Filter* button.

You can see the filter results by clicking *Show Report*, as shown in the following image.



Searching for Files

The Search Files tool provides a search function on the Application Tree. It can search for files using a variety of criteria, including file name, location, description, contents, statistics, and type, in an application directory or across the entire server.

To view the file search page, click the *Search Files* button on the Applications page ribbon, or right-click the *Application Directories* folder and select *Search Files* on the context menu.

The Search Files page opens, as shown in the following image.

Enter your search criteria, and click *Search*.

The report returned allows you to open procedures, Master Files, and displayable data in the editor.

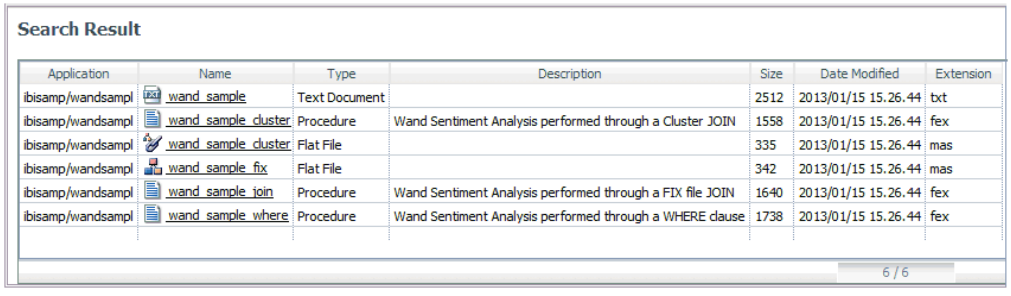
Procedure: How to Search for Files

The Search Files page provides a wide range of search criteria.

1. From the Web Console menu bar, click *Applications*.
2. Click the *Search Files* icon.

   The Search Files page opens.
3. Select the search criteria from the *File Name*, *File Location*, *File Statistics*, and *File Type* sections.
4. Click the *Search* button.

The Search Results are displayed, as shown in the following image.



Sorting the Application Tree

Sorting enables you to change the order in which items are listed on the application tree.

Procedure: How to Sort the Application Tree

1. From the Web Console menu bar, click *Applications*.

   The Applications page opens.

2. From the ribbon, click the Sort by icon, and select a sort method.

   You can sort by *Name, Size, Type,* or *Modified* date. Type is the default.

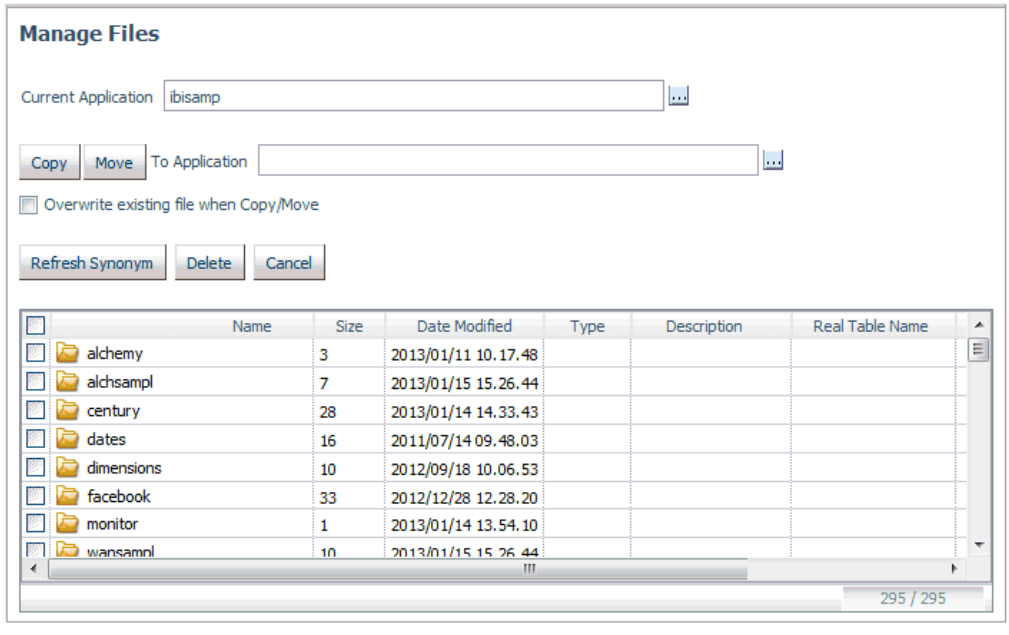
Managing Application Files

You can copy and move files between applications, delete them from applications, and refresh synonyms in applications. You can also copy, move, or delete subfolders from one application to another. The subfolders will be copied, moved, or deleted with all of its files and all of its own subfolders.

Procedure: How to Manage Application Files

1. From the Web Console menu bar, click *Applications*.
2. Right-click an application folder and select *Manage Files*.

The Manage Files page opens.



3. Select the check box next to one or more files. To select all files, click *Select All*.
4. Enter an application folder in the *Current Application* field.

   The selector (...) button opens the Select current application dialog box which allows you to navigate to the application.
5. Enter an application folder in the *To Application* field or click the selector button (...) and navigate to one.
6. Click *Copy* or *Move*.

   The file(s) are copied or moved to the selected application.
7. To delete files, select their check boxes and click *Delete*.

Procedure: How to Refresh Synonyms in an Application

1. From the Web Console menu bar, click *Applications*.
2. Right-click an application folder and select *Manage Files*.

   The Manage Files page opens.
3. Select the check box next to one or more synonyms.

   To refresh all listed synonyms in the selected application, click *Select All*.

4. Click *Refresh Synonym*.

   This is also available by right-clicking an application ,selecting *New Synonym (Create or Update)*, choosing the *Update* option and *Refresh Synonym* on the Master File menu.

Application Commands Overview

This topic lists the platform-independent application (APP) commands that enable you to control the application environment.

You can use the following wildcard characters in the file name and file type references for the APP commands COPYFILE, MOVEFILE, DELETEFILE, and RENAMEFILE.

- An asterisk (*) replaces any combination of characters of any length (including zero).

  Note that an asterisk can also be used to replace the entire filename or filetype parameter.

- A question mark (?) replaces zero or one character.

Reference: APP Commands Quick Reference

Click any command in the following charts to access detailed information, including the required syntax.

Search Path Management Commands

Command	Description
APP PATH	Sets or resets the application search path.
APP PREPENDPATH	Temporarily adds application names to the beginning of an existing APP PATH search path.
APP APPENDPATH	Temporarily adds application names to the end of an existing APP PATH search path.
APP MAP	Defines a virtual application that points to a physical location outside of the approot structure. This command makes the virtual application available for addition to the search path. It does not automatically add it to the APP PATH.
APP SET METALOCATION_SAME	Indicates whether corresponding Master and Access files must be in the same application directory.

Command	Description
APP ? METALOCATION_SAME	Retrieves the value of APP SET METALOCATION_SAME.
APP SHOWPATH	Lists all the currently active applications in the search path.

Application Management Commands

Command	Description
APP COPY	Copies the contents of one application to a second application.
APP CREATE	Creates an application under the approot location.
APP DELETE	Deletes an application.
APP MOVE	Moves the contents of one application to a second application.
APP PROPERTY CODEPAGE	Specifies a code page for files in an application.
APP RENAME	Renames an application.

File Management Commands

Command	Description
APP COPYF[ILE]	*Copies a single component or component type from one application to another.
APP MOVEF[ILE]	*Moves a single component or component type from one application to another.
APP RENAMEF[ILE]	*Renames a single component or component type in an application.
APP DELETEF[ILE]	*Deletes a single component or component type from an application.

* The shortened form of the APP commands is used in the remainder of this document.

Output Redirection Commands

Command	Description
APP HOLD	Controls where output files are created for any HOLD, SAVE, SAVB, CREATE SYNONYM, or APP QUERY HOLD process in the application, unless a FILEDEF command has been used to allocate data files.
APP HOLDDATA	Designates an application as the location for temporary data files created with the HOLD, SAVE, or SAVB command.
APP HOLDMETA	Designates an application as the location for temporary Master and Access Files created with the HOLD command.
APP FI[LEDEF]	This command has been deprecated and aliased to FILEDEF.

Help Commands

Command	Description
Help Information: APP HELP	Displays a list of APP commands with a brief description of each.

Reference: Application Metadata Commands and Metadata Tables

Click any command in the following chart to access detailed information, including the required syntax.

Command	Description
STATEapp/file.extension	Check file existence

Command	Description
APP LIST app [HOLD]	Lists the applications under approot. If the HOLD option is used, it lists the applications under approot and writes the output to a temporary file called focappl.ftm, which you can then use in a report request.
APP QUERY app [HOLD]	Lists all files in the application. If the HOLD option is used, it lists all files in the application and writes the output to a temporary file called focappq.ftm, which you can then use in a report request.
catalog/sysfiles	Table, list of accessible app name objects on path for a given type (default MASTER).
catalog/sysdirs	Table, recursive list of physical files under a physical directory.
catalog/sysapps	Table, metadata for physical objects on path.
catalog/systables	Table, app name of tables (and related metadata) on path.

Search Path Management Commands

The server has a default search path for application and system components. You can supplement this search path by using one or more of the following APP commands:

- APP PATH
- APP PREPENDPATH
- APP APPENDPATH
- APP SET METALOCATION_SAME
- APP ? METALOCATION_SAME

Generally, these commands add applications to the beginning of the default search path. The exception is temporary components that are created in the current session. These temporary components are searched first, before the user defined path.

You can issue the APP PATH command manually or set the application search path from the Web Console or the Data Management Console. When you configure the application path from the Web Console or the Data Management Console, the APP PATH command is stored in a selectable profile (global, group, or user). The global profile, edasprof, is the default.

APP PATH

The APP PATH command sets the search path to a designated list of application names that refer to applications under the approot value. You can specify multiple application names to extend the search path.

Syntax: How to Add an Application to the Search Path Manually

```
APP PATH app1[/] [app2[/] ...]
    [appn[/]]
```

where:

app1...appn

Are application names. If you follow an application name with a slash (/), nested applications (the subtree of applications below the named application) will not be in the search path. If you do not follow the application name with a slash, the *nested_app* parameter in the edaserve.cfg file determines whether nested applications are searched for files referenced in a procedure, and to what level. If you need to specify more application names than can fit on one line, add the continuation character (-) at the end of the first line, and code more application names on the next line.

Note:

- You can use the APP PATH command without an application name to reset the search path to the initial list.
- APP PATH does not validate the application list.

APP PREPENDPATH

The APP PREPENDPATH command enables you to temporarily add application names to the beginning of an existing APP PATH search path.

If you wish to use this command to alter the search path, you must code it manually in your application.

Syntax: How to Add Application Names to the Beginning of a Search Path

```
APP PREPENDPATH app1[/] [app2[/]] ...
   [appn[/]]
```

where:

app1...appn

Are application names. If you follow an application name with a slash (/), nested applications (the subtree of applications below the named application) will not be in the search path. If you do not follow the application name with a slash, the *nested_app* parameter determines whether nested applications are searched for files referenced in a procedure, and to what level. If you need to specify more application names than can fit on one line, add the continuation character (-) at the end of the first line, and code more application names on the next line.

APP APPENDPATH

The APP APPENDPATH command enables you to temporarily add application names to the end of an existing APP PATH search path.

If you wish to use this command to alter the search path, you must code it manually in your application.

Syntax: How to Add Application Names to the End of a Search Path

```
APP APPENDPATH app1[/] [app2[/]] ... [appn[/]]
```

where:

app1...appn

Are application names. If you follow an application name with a slash (/), nested applications (the subtree of applications below the named application) will not be in the search path. If you do not follow the application name with a slash, the *nested_app* parameter determines whether nested applications are searched for files referenced in a procedure, and to what level. If you need to specify more application names than can fit on one line, add the continuation character (-) at the end of the first line, and code more application names on the next line.

APP MAP

The APP MAP command allows you to assign an application name to a non-approot application anywhere in the file system. This application name becomes a virtual application under approot, which can be referenced in an APP PATH command and any other APP command that takes an application name as a parameter.

Note that mapping does *not* automatically add a directory to the path, it simply makes it available for addition to the search path.

Syntax: How to Map a Physical File Location Outside of APPROOT Manually

```
APP MAP virtualname real_location
```

where:

virtualname

Is an application name of up to 64 characters that can later be used in an APP PATH command.

real_location

Is a real full path name or DDname in the native style of the given operating system. On UNIX, Linux, i5 and z/OS HFS Deployment, the location may be a mixed case name, but the MAP virtualname itself is always handle insensitive when used (that is, EX mymap/mytest).

Note that if the real location contains spaces, it must be surrounded by double quotation marks.

Note: On IBM i, the APP MAP command can only be used to map an IFS directory and not a QSYS library.

Example: Sample APP MAP Commands

Basic example for Windows:

```
APP MAP test c:\temptest\
```

Note that if a path name contains spaces, the name must be surrounded by double quotation marks. For example:

```
APP MAP test "c:\temp test\"
```

Syntax: How to Map DDNAME Allocations

For the Unified Server (HFS and PDS deployments), the syntax for this type of mapping is

```
APP MAP appname file_extension=//dd:ddname;file_extension=//
dd:ddname;...
```

where:

appname

Is the name of the application used to reference this mapping in an APP PATH, APP APPENDPATH, or APP PREPENDPATH command.

file_extension

Is one of the following valid server file extensions:

```
.mas
.fex
.acx
.htm
.sty
.gif
.psb
```

ddname

Is the ddname of the allocation you wish to map. The allocation can be performed using JCL code or a DYNAM command.

Example: Mapping DDNAME Allocations

```
DYNAM ALLOC FILE MYMAS DA EDAARH.MASTER.DATA SHR REU
APP MAP APP1 MAS=//DD:MYMAS;
APP APPENDPATH APP1
```

By default, the server has an APP MAP command in the edasprof.prf file to map the application MVSAPP to the allocations FOCEXEC, MASTER, ACCESS, HTML, FOCSTYLE, GIF, FOCPSB. While allocations of these ddnames are not required for the APP MAP command to be valid, once the ddnames are allocated by JCL or DYNAM commands, they become available for use.

Reference: APP MAP With Universal Naming Convention (UNC)

On platforms that support Universal Naming Convention (UNC), you must use the UNC to designate a network drive to access APP directories. The UNC must:

- Be at least one folder below the initial shared location.
- Not contain spaces unless enclosed in single quotation marks. For example,

```
\\mynode\myshare\accnting
'\\mynode\my share\accnting'
```

APP SET METALOCATION_SAME

The APP SET METALOCATION_SAME command identifies whether Master Files and their corresponding Access Files must be in the same location.

Syntax: How to Control the Location of Synonym Files

```
APP SET METALOCATION_SAME {ON|OFF}
```

where:

ON

Specifies that Master Files and their corresponding Access Files must reside in the same application directory. ON is the default value.

OFF

Specifies that once the Master File for a request is located, the server will use the active search path to find the corresponding Access File.

APP ? METALOCATION_SAME

The APP ? METALOCATION_SAME command queries whether Master Files and their corresponding Access Files must be in the same location.

Syntax: How to Query Whether Synonym Files Must Reside in the Same Location

```
APP ? METALOCATION_SAME
```

If the result of this query command is ON, the server expects to find corresponding Master and Access Files in the same application directory. If the result is OFF, the server uses the active search path to find the Access File that corresponds to a given Master File.

APP SHOWPATH

The APP SHOWPATH command lists all the currently active applications in the search path, including baseapp, which is always last. This list mirrors the list of applications displayed in the applications tree on the navigation pane.

Syntax: How to List Active Applications

```
APP SHOWPATH
```

Example: Listing Active Applications in the Search Path

The Server is generally installed with two default applications: ibisamp (contains sample files), and baseapp (which can contain any files you create).

The APP SHOWPATH command generates the following output is:

```
ibisamp
baseapp
```

Application and File Management Commands

The APP commands in this section provide management options for applications and their component files.

APP CREATE

In general, the APP CREATE command creates an application under the approot location. The exception is a PDS deployment on a Unified Server, where an application is a physical entity and each of its component file types is stored in a separate PDS.

The APP CREATE command can create any number of applications with one command.

Syntax: How to Create an Application Manually

```
APP CREATE app1[/app1a...] [app2[/app2a...] ...
   [appn[/appna...] ] [DROP]
```

where:

app1...appn

Are application names under approot. The application name can be up to 64 characters.

app1a...appna

Are nested application directories, allowed when nested applications are configured. In order to create a nested application, the parent application must already exist.

DROP

Deletes an application if one already exists with the same name as the one to be created, and then creates a new application with that name. Note that any files in the pre-existing application are deleted. Without the DROP option, a message will be generated, and the pre-existing application will not be deleted or changed.

The application name may not contain spaces. If the name contains spaces, each section is understood to be a separate application. If you require a name with spaces, you must create it using another mechanism, such as the Windows Explorer. You can then use the APP MAP command to add it to APPROOT.

If you need to specify more application names than can fit on one line, add the continuation character (-) at the end of the first line, and code more application names on the next line.

The word HOLD cannot be used as an application name.

Syntax:

How to Change Default Characteristics of Component File Types (PDS Deployment Only)

If you are working on a Unified Server in PDS deployment, you can change the default characteristics of individual component file types by issuing a DYNAM SET APP command. This command controls the types of component files that are generated for the application when an APP CREATE command is issued. By default, all component file types are generated.

The syntax is

```
DYNAM SET APP FOR filetype [SKIP|CREATE] [POSTFIX aaa.bbb] [parms]
```

where:

filetype

Are the component types that may be affected by this command, in uppercase: FOCEXEC, MASTER, ACCESS, HTML, GIF, FOCSTYLE, MAINTAIN, WINFORMS, ETG. You must issue a separate command for each component type you wish to affect.

SKIP

Indicates that the designated file type should not be created when the APP CREATE command is issued.

CREATE

Creates the designated file type when the APP CREATE command is issued. This is the default setting.

POSTFIX

Specifies the lower level qualifier of the DSN (data set name) for the component type. The APPROOT value is used to complete the full DSN, which is expressed as

approotvalue.appname.component_type

The default value for component_type is *filetype.DATA*.

parms

Are the allocation parameters you can set. The default parameter values are:

File Type	Parameter
FOCEXEC	`RECFM VB TRKS LRECL 4096 BLKSIZE 27998 SPACE 50 50 DIR 50`
MASTER	`RECFM FB TRKS LRECL 80 BLKSIZE 22000 SPACE 50 50 DIR 50`
ACCESS	`RECFM FB TRKS LRECL 80 BLKSIZE 22000 SPACE 50 50 DIR 50`
HTML	`RECFM VB TRKS LRECL 4096 BLKSIZE 27998 SPACE 50 50 DIR 50`
GIF	`RECFM VB TRKS LRECL 4096 BLKSIZE 27998 SPACE 50 50 DIR 50` The GIF file type creates libraries for GIF and JPG files.
FOCSTYLE	`RECFM FB TRKS LRECL 1024 BLKSIZE 27648 SPACE 50 50 DIR 50`
MAINTAIN	`RECFM VB TRKS LRECL 4096 BLKSIZE 27998 SPACE 50 50 DIR 50`
WINFORMS	`RECFM VB TRKS LRECL 4096 BLKSIZE 27998 SPACE 50 50 DIR 50`
ETG	`RECFM FB TRKS LRECL 80 BLKSIZE 22000 SPACE 50 50 DIR 50`

Example: **Changing Default Characteristics of an Application (PDS Deployment)**

The following command indicates that GIF files should not be created when the APP CREATE command is issued.

```
DYNAM SET APP FOR GIF SKIP
```

The following command indicates that Procedures (FOCEXECs) should be created when APP CREATE is issued.

```
DYNAM SET APP FOR FOCEXEC TRKS SP 10 20 DIR 30
```

APP COPY

The APP COPY command copies the entire contents of one application to another. The target application must already exist.

Syntax: How to Copy an Application

```
APP COPY app1[/app1a...] app2[/app2a...]
```

where:

app1[/*app1a...*]

Is the application being copied. It can be a nested application name.

app2[/*app2a...*]

Is the application to which the contents of the first application are being copied. It can be a nested application name.

APP COPYF[ILE]

The APP COPYF[ILE] command copies one or more components or component types from one application to another.

Note that if you copy a component manually, you can, optionally, rename it in the process.

If you copy a Master File, the corresponding Access File is also copied. However, copying an Access File (file type FOCSQL) does not automatically copy the corresponding Master File.

Syntax: How to Copy an Application Component Manually

```
APP COPYF[ILE] app1[/app1a...]
  {filename1|*} filetype1   app2 [/app2a...]
  {filename2|*} {filetype2|*} [IFEXIST] DROP
```

where:

app1[/*app1a...*]

Is the application that contains the component to be copied. It can be a nested application name.

filename1

Are the components to be copied. Use an asterisk (*) to copy all components of file type *filetype1*.

You can use the following wildcard characters in the file name and file type references.

- An asterisk (*) replaces any combination of characters of any length (including zero).

  Note that an asterisk can also be used to replace the entire filename or filetype parameter.

- A question mark (?) replaces zero or one character.

filetype1

Is the file type, in uppercase, of the component to be copied.

app2[*/app2a...*]

Is the application to which the named component is being copied. It can be a nested application name.

filename2

Is the component name in the target application, after the copy process. Use an asterisk (*) to propagate the file names from the source application to the target application.

filetype2

Is the component type, in uppercase, in the target application after the copy process. Use an asterisk (*) to propagate the file types from the source application to the target application.

IFEXIST

Ignores any component in the source application that does not exist.

DROP

Overwrites any component already in the target application with the same name and file type as a component being copied.

For a full list of the types of files you can copy with APP commands, see *Designating File Types for APP Commands* on page 108.

APP MOVE

The APP MOVE command moves the entire contents of one application to another. The target application must already exist.

Syntax: How to Move an Application

APP MOVE *app1*[*/app1a...*] *app2*[*/app2a...*]

where:

app1[/*app1a...*]

Is the application being moved. It can be a nested application name.

app2[/*app2a...*]

Is the application to which the contents of the first application are being moved. It can be a nested application name.

APP MOVEF[ILE]

The APP MOVEF[ILE] command moves one or more components or component types from one application to another.

Note that if you move a component manually, you can, optionally, rename it in the process.

If you move a Master File, the corresponding Access File is also moved. However, moving an Access File (file type FOCSQL) does not automatically move the corresponding Master File.

Syntax: How to Move an Application Component Manually

```
APP MOVEF[ILE] app1[/app1a...]
  {filename1|*} filetype1  app2 [/app2a...]
  {filename2|*} {filetype2|*} [IFEXIST] [DROP]
```

where:

app1[/*app1a...*]

Is the application that contains the component to be moved. It can be a nested application name.

filename1

Is the name of the component to be moved. Use an asterisk (*) to move all components of file type *filetype1*.

You can use the following wildcard characters in the file name and file type references.

- An asterisk (*) replaces any combination of characters of any length (including zero).

  Note that an asterisk can also be used to replace the entire filename or filetype parameter.

- A question mark (?) replaces zero or one character.

filetype1

Is the file type, in uppercase, of the component to be moved.

app2[*/app2a...*]

Is the application to which the named component is being moved. It can be a nested application name.

filename2

Is the component name in the target application, after the move process. Use an asterisk (*) to propagate the file names from the source application to the target application.

filetype2

Is the component type, in uppercase, in the target application after the move process. Use an asterisk (*) to propagate the file types from the source application to the target application.

IFEXIST

Ignores any component in the source application that does not exist.

DROP

Overwrites any component already in the target application with the same name and file type as a component being moved.

For a full list of the types of files you can move with APP commands, see *Designating File Types for APP Commands* on page 108.

APP DELETE

The APP DELETE command deletes applications under approot.

Syntax: How to Delete an Application Manually

```
APP DELETE app1[/app1a...] [app2[/app2a...] ...
  [appn[/appna...]]
```

where:

app1[*/app1a...*]... [*appn*[*/appna...*]

Are application names. Nested application names are supported. If you need to specify more application names than can fit on one line, add the continuation character (-) at the end of the first line, and enter additional application names on the next line.

APP DELETEF[ILE]

The APP DELETEF[ILE] command deletes one or more components or component types from an application.

If you delete a Master File, the corresponding Access File is also deleted. However, deleting an Access File (file type FOCSQL) does not automatically delete the corresponding Master File.

Syntax: How to Delete an Application Component Manually

```
APP DELETEF[ILE] app[/appna...] {filename|*} filetype
```

where:

appn[/appa...]

Is the application from which the component or component type is being deleted. Nested application names are supported.

filename

Is the name of the component to be deleted. Use an asterisk (*) to delete all files of type *filetype*.

You can use the following wildcard characters in the file name and file type references.

- An asterisk (*) replaces any combination of characters of any length (including zero).

  Note that an asterisk can also be used to replace the entire filename or filetype parameter.

- A question mark (?) replaces zero or one character.

filetype

Is the component type, in uppercase, of the component to be deleted.

For a full list of the types of files you can use with APP commands, see *Designating File Types for APP Commands* on page 108.

APP PROPERTY CODEPAGE

The APP PROPERTY appname CODEPAGE command identifies the codepage to be used for non-data files in the application directory.

Syntax: How to Specify a Code Page for an Application

```
APP PROPERTY app[/appa...] CODEPAGE number
```

where:

app[/appa...]

Is an application name. Nested application names are supported.

number

Is the code page number for non-data files in the application.

APP RENAME

The APP RENAME command renames an existing application.

Note: You cannot rename an application if it is active in the search path.

Syntax: How to Rename an Application

```
APP RENAME app1[/app1a...] app2[/app2a...]
```

where:

app1[/*app1a...*]

Is the application name to be renamed. It can be a nested application name.

app2[/*app2a...*]

Is the new application name of up to 64 characters. It can be a nested application name.

Example: Renaming an Application

The following shows app1 being renamed to app2.

```
APP RENAME app1 app2
```

APP RENAMEF[ILE]

The APP RENAMEF[ILE] command renames one or more components in an application.

If you rename a Master File, the corresponding Access File is also renamed. However, renaming an Access File (file type FOCSQL) does not automatically rename the corresponding Master File.

Syntax: How to Rename an Application Component

```
APP RENAMEF[ILE] app[/appa... ] filename1
filename2 filetype [DROP]
```

where:

app[/*appa...*]

Is the name of the application that contains the component being renamed. It can be a nested application name

filename1

Is the file name of the component to be renamed.

You can use the following wildcard characters in the file name and file type references.

- An asterisk (*) replaces any combination of characters of any length (including zero).

  Note that an asterisk can also be used to replace the entire filename or filetype parameter.

- A question mark (?) replaces zero or one character.

filename2

Is the new name for the component. The component name may be up to 64 characters.

filetype

Is the file type, in uppercase, of the component to be renamed.

DROP

Overwrites an existing component with the same file name and file type.

For a full list of the types of files you can use with APP commands, see *Designating File Types for APP Commands* on page 108.

Designating File Types for APP Commands

The APP COPYF, APP MOVEF, APP DELETEF, and APP RENAMEF commands enable you to perform their actions on a wide variety of file types.

Reference: APP Commands and File Types

The following is a comprehensive list of the file types you can use with APP commands and the file extensions associated with the on-disk names for hierarchical file systems.

Note that the file types must be coded in uppercase in any APP command that requires it.

Note: This list reflects file types supported across all Information Builders products and release levels. Particular file types may not be supported in particular releases or with every product.

File Type	File Extension
ACX	.acx
ADR	.adr

File Type	File Extension
AFM	.afm
BMP	.bmp
BST	.bst
cascading style sheet	.css
CONTROL	.ctl
DATA	.dat
DDS	.DDS
DEFAULT	The APP *filename* value is used to derive the physical extension for the APP command, so that unknown user-defined extensions may be supported in an APP command (for example, APP COPYFILE BASEAPP MYFILE.FOO DEFAULT BASEAPP MYFILE FOCEXEC).
DTD	.dtd
EDANLS	.nls
EDAPRFU	.prf
EDAPROF	.prf
EDAPSB	.psb
EPS	.eps
ERRORS	.err
ETG	.etg
ETL	.etl
EXCEL	.xls
FMU	.fmu

File Type	File Extension
FOCCOMP	.fcm
FOCDEF	.def
FOCEXEC OR FEX	.fex
FOCFTMAP	.fmp
FOCPSB	.psb
FOCSQL	.acx
FOCSTYLE	.sty
FOCTEMP	.ftm
FOCUS	.foc
GIF	.gif
HLI	.hli
HTML	.htm
IBICPG	.sl
JPG	.jpg
JS	.js
LSN	.lsn
MAINTAIN	.mnt
MASTER OR MAS	.mas MASTER has a special behavior that any matching Access File (.acx) is also operated upon by the APP command. This is so metadata is operated upon as a matched pair. Use MAS if it is strictly desired to only operate on the Master File and not the Access File.

File Type	File Extension
MHT	.mht
Microsoft Access database	.mdb
MNTPAINT	.mpt
OMI	.omi
PDF	.pdf
PFA	.pfa
PFB	.pfb
PNG	.png
PS	.ps
SMARTLIB	.knb
SQL	.sql
SVG	.svg
TABS	.txt
TDL	.tdl
TRF	.trf
TTEDIT	.tte
TXT	.txt
WINFORMS	.wfm
WSDL	.wsd
XHT	.xht
XLSM	.xlsm
XLSX	.xlsx

File Type	File Extension
XLTM	.xltm
XLTX	.xltx
XML	.xml
XSD	.xsd
XSL	.xsl

Output Redirection Commands

Three APP commands (APP HOLD, APP HOLDDATA, and APP HOLDMETA) along with the FILEDEF and DYNAM commands comprise a class of commands that control where output is stored. In order to redirect output as you wish, it is important to understand the interactions among these commands.

Note: When the same behavior applies for APP HOLD, APP HOLDDATA, and APP HOLDMETA, these commands are referred to collectively as APP HOLD*. Note also that although DYNAM (USS only) and FILEDEF are not members of the APP family of commands, these file allocation commands interact with the APP HOLD* commands. Therefore, where appropriate, these commands are also included in this discussion. APP FI[LEDEF] has been deprecated and aliased to FILEDEF.

The most straightforward of these commands is APP HOLD, which allows you to relocate all output to a particular application. You can use this command with operations that produce output files, such as HOLD, SAVE and SAVB, as well as with CREATE SYNONYM and APP QUERY HOLD. (For details about HOLD, SAVE, and SAVB commands, see the *Creating Reports With WebFOCUS Language* manual.)

The APP HOLD* commands are particularly helpful when you are creating permanent files for other applications to use. However, if a command is used at an inappropriate point in the application or if it remains in effect when further steps are performed within the application, the target application may be flooded with intermediate and unintended files. Understanding the behavior of each command and the interactions among them will help you avoid this situation.

Reference: **Interactions Among Output Redirection Commands**

This chart describes the behavior associated with each redirection command and the interactions among them if multiple commands are used.

Command	Stand Alone	Notes
`APP HOLD`	Redirects all Agent output from HOLD, SAVE, SAVB, CREATE SYNONYM, and APP QUERY HOLD commands to the designated application.	When issued without a specific *appname,* APP HOLD has the effect of turning off the command.
`APP HOLDDATA`	Redirects the *data* from HOLD, SAVE, and SAVB operations to the designated application, but does not redirect the associated *metadata* (see Note 1 after chart).	Overrides APP HOLD.
`APP HOLDMETA`	Redirects the *metadata* from HOLD, SAVE, and SAVB operations to the designated application, but does not redirect the associated *data* (see Note 1 after chart).	Overrides APP HOLD.
`FILEDEF` *ddname* `DYNAM ALLOC` *ddname*	Redirects the *data* from specific HOLD, SAVE, and SAVB operations to the designated target, but does not redirect the associated *metadata* (see Note 1 after chart). The AS phrase must match the ddname. When there is no AS phrase, the *ddname* must match a predefined default name: HOLD for HOLD output files; SAVE for SAVE output files; and SAVB for SAVB output files.	Overrides APP HOLD and APP HOLDDATA.

Command	Stand Alone	Notes
`DYNAM ALLOC HOLDMAST`	Redirects the *metadata* from HOLD, SAVE, and SAVB operations to the designated target (using the HOLDMAST *ddname*), but does not redirect the associated *data* (see Note 1 after chart). The recommended practice is to use this command on a request-by-request basis to avoid overriding previous output. If used as a global setting, previously held output will be overwritten with the same name.	Overrides APP HOLD and APP HOLDMETA.

Note:

- Not all formats have associated metadata. For example, the HOLD FORMAT PDF command does not produce metadata, therefore, there is no metadata to redirect.
- The use of the APP HOLD command to redirect CREATE SYNONYM output is neither necessary nor desirable since the CREATE SYNONYM command directly supports application names using the syntax:

  ```
  CREATE SYNONYM appname/synonym ...
  ```

APP HOLD

The APP HOLD command defines an application in which to hold output data files (and associated Master and Access Files, if applicable) created by a HOLD, SAVE, or SAVB process in the application.

APP HOLD is intended to be used to refresh files that are common for all users of the application. It should not be used for private files since it points to an application area that is used by multiple users. If the same hold name (HOLD or AS *name*, for example) is used, conflicts between users could result.

For related information, see *Interactions Among Output Redirection Commands* on page 113.

Syntax: How to Designate a Storage Location for Temporary Files

```
APP HOLD appname[/appnamea...]
```

where:

appname[/*appnamea*...]

Is the application in which you wish to store output files. It can be a nested application name.

Note: Issuing APP HOLD without an *appname* turns off the effects of the command.

APP HOLDDATA

The APP HOLDDATA command designates an application as the location for storing data files created with the HOLD command. For related information, see *Interactions Among Output Redirection Commands* on page 113.

Syntax: How to Designate a Storage Location for Data Files

```
APP HOLDDATA appname[/appnamea...]
```

where:

appname[/*appnamea*...]

Is the name of the location for the data files created by any write process in the application. It can be a nested application name

APP HOLDMETA

The APP HOLDMETA command designates an application directory as the location for storing Master and Access Files created in the application. For related information, see *Interactions Among Output Redirection Commands* on page 113.

Syntax: How to Designate a Storage Location for Master and Access Files

```
APP HOLDMETA appname[/appnamea...]
```

where:

appname[/*appnamea*...]

Is the name of the location for the Master and Access Files created in the application. It can be a nested application name.

APP FI[LEDEF]

The APP FI[LEDEF] command has been deprecated and aliased to FILEDEF. For information, see the *Stored Procedure Reference* manual.

Application Metadata Commands and Catalog Metadata

Developers may want to write applications that check application metadata and decide a course of action. For example, they may want to check the existence of a file or a file date, and decide on the need for another step, such as recreation of the file. There are multiple ways to accomplish a simple check for file existence or some other attribute, that have evolved over the release history of the product. However, some of these methods have limitations. A good example of this is the STATE command, which uses a native path name for UNIX. This type of path name would not match a Windows or OpenVMS file path and, therefore, would require IF THEN ELSE or GOTO logic to issue the correct version of the command for the operating environment, that might be quite cumbersome, depending on how often it is needed.

To solve part of this problem, commands such as STATE, FILEDEF, and DYNAM have been extended to support APP names (that is, issue APP MAP then use STATE mymap/myproc.fex). To deal with more complex issues, such as retrieving a list of available applications (APP names) and files within a particular application, a series of APP commands were developed (APP LIST and APP QUERY). However, as features such as nested applications (sub-directories) were implemented, it became apparent that a much more extended ecosystem for accessing application metadata was needed.

To satisfy this need for extended information, various internal tables were extended or created. Today the catalog/sysapps table is the primary method for accessing application metadata using standard TABLE or SELECT syntax. This is what is used in most internal applications. That is not to say that the prior methods are no longer supported. At times they can provide quick and simple coding for a specific need, but they have limitations (as noted). More complex situations require the use of the newer methods to access information. Additionally, tables such as catalog/systables and catalog/syscolum can provide additional information that is table specific, such as what DBMS a table is using and the data specification of particular columns, but they are beyond the scope of this section. It should also be noted that the newer methods occasionally overlap on how to accomplish a task. For example, a number of the catalog/sys* tables can be used to answer the question of whether a file exists. However, the tables differ from each other in the more detailed information, such as physical or application locations and attributes.

Retrieving Basic Information

The following commands return basic information about files and applications.

STATE

The STATE command lets you check for the existence of a file. The file reference you supply can be the full path native operating system file name, or a file name prefaced with an APP name. This section only described the use of APP name prefaced files. When an APP name is used, it does not matter if the name was natively created under the APPROOT location of the server or as an APP MAP name.

If the file does not exist, the STATE command displays a message to that effect. After issuing the STATE command, the &RETCODE system variable contains the value zero (0) if the file exists, or a non-zero value if the file does not exist.

Syntax: How to Check File Existence

```
STATE appname/filename.filetype -TYPE RETCODE &RETCODE
```

where:

appname

Is the application under which the file is stored.

filename

Is the name of the file.

filetype

Is the file type or extension of the file.

If the file exists, the &RETCODE value will be 0 (zero). Otherwise, it will be non-zero and can be used to further direct the logic of the application, typically in a -SET or a -IF command. The STATE command will also output a *not found* message. To suppress this message, use the SET TRMOUT={OFF|ON} command.

For example, the following STATE command checks the existence of the file *myproc.fex* in the *baseapp* application. The STATE command displays a message if the file does not exist. The -TYPE command displays the value zero (0) if the file exists or the value -1 if the file does not exist.

```
STATE baseapp/myproc.fex
-TYPE RETCODE &RETCODE
```

Example: Checking the Existence of a File With the STATE Command

The following partial example suppresses the message returned by the STATE command, issues the STATE command to check if the file *myproc.fex* exists in the baseapp application, checks the return code, and creates the file if it does not exist, before continuing with the next step in the application. If the file does exist, the code immediately transfers to the next step in the application, (-RESUME label):

```
SET TRMOUT=OFF
STATE baseapp/myproc.fex
SET TRMOUT=ON
-IF &RETCODE EQ 0 THEN GOTO RESUME;
...
* Some code to create the file goes here
...
-RESUME
```

APP LIST

The APP LIST command alphabetically lists the applications available under the application root, APPROOT, or under an APP MAPped location. It does not care if the APP is on the current application map or not, as it is a raw list of available applications.

Syntax: How to List the Applications in APPROOT

```
APP LIST [HOLD]
```

If the HOLD option is used, the output is written to a temporary file called focappl.ftm, (FOCAPPL on PDS Deployment), which can, in turn, be used in a request to drive a report or take an action using the catalog/focappl Master File.

Limitations:

- APP LIST does not display nested application names.
- On operating systems that use case sensitive file names (such as UNIX), uppercase physical directory names are not valid (so are not returned by APP LIST). APP names are case insensitive, but they are created on disk as lowercase, which may in turn be upper-cased by the native operating system (that is, OpenVMS ODS/2 disks and PDS uppercase file names). However, APP LIST returns them in lowercase, to be homogenous across operating systems.

Example: Using APP LIST to List and Work with Applications

The following request lists applications

```
APP LIST
```

The APP LIST output is:

```
BEGIN-APP-LIST
15/02/2000  13.36.38   baseapp
15/02/2000  13.36.38   ggdemo
15/02/2000  13.36.38   ncp
15/02/2000  13.36.38   template
END-APP-LIST
```

The following request lists applications that have been stored using the HOLD option

```
APP LIST HOLD
SQL SELECT DATE, TIME, APPNAME FROM FOCAPPL;
END
```

The APP LIST output is:

```
DATE        TIME      APPNAME
----        ----      -------
15/02/2000  13.36.38   baseapp
15/02/2000  13.36.38   ggdemo
15/02/2000  13.36.38   ncp
15/02/2000  13.36.38   template
```

The following practical example of using the APP LIST HOLD command issues a TABLE request against the HOLD file to check if any files exist in the application *myapp*. If no lines are returned, the application does not exist, so it is created, and the application continues. Otherwise, the application continues without creating the application.

```
APP LIST HOLD
TABLE FILE FOCAPPL
PRINT * ON TABLE HOLD WHERE APPNAME = 'myapp'
END
-IF &LINES GT 0 THEN GOTO RESUME
APP CREATE myapp
-RESUME
```

APP QUERY

The APP QUERY command lists files within a given application. Applications and specific nested applications can be queried.

Syntax: How to List Components

```
APP QUERY app1[/app1a...]  [app2[/app2a]...] ...
  [appn[/appna]] [HOLD]
```

where:

app1[*/app1a...appn*[*/appna*]

Are application names. They can be nested application names. If you need to specify more application names than can fit on one line, add the continuation character (-) at the end of the first line, and continue more application names on the next line.

If the HOLD option is used, the output is written to a temporary file called focappq.ftm (FOCAPPQ on PDS Deployment), which can, in turn, be used in a request to drive a report or take an action using the catalog/focappq Master File.

Limitations: All files within an APP are listed. On systems like UNIX, this may include files of any case, so files such as MYPROC.FEX and myproc.fex may appear in a listing, but only the lowercase version would be accessed in a request.

Example: Listing Application Files

The following request lists application files.

```
APP QUERY abc
```

The APP QUERY output is:

```
BEGIN-APP-QUERY: abc
24/10/2014 21.38.28        4 F myproc1.fex
24/10/2014 21.38.35        4 F myproc1.fex
24/10/2014 21.37.49        4 F myapp1
24/10/2014 21.32.36        0 D myapp2
END-APP-QUERY
```

The following request lists files that have been stored using the HOLD option.

```
APP QUERY ABC HOLD
SQL SELECT DATE, TIME, GMTTIME, SIZE, OTYPE, FILENAME, APPNAME FROM
FOCAPPQ ;
END
```

The APP QUERY output is:

```
DATE        TIME      GMTTIME     SIZE  OTYPE FILENAME     APPNAME
----        ----      -------     ----  ----- --------     -------
24/10/2014  21.38.28  1414201108  4     F     myproc1.fex  abc
24/10/2014  21.38.35  1414201115  4     F     myproc2.fex  abc
24/10/2014  21.37.49  1414201069  4     D     myapp1       abc
24/10/2014  21.32.36  1414200756  0     D     myapp2       abc
```

Note that APP QUERY ... HOLD returns a slightly extended type of information. Whitespace has selectively been removed from the above output for readability (the FILENAME column is actually 70 characters wide).

The following practical example of using the APP QUERY HOLD command checks the existence of the file *myproc1.fex* in application *abc*. If the file does not exist, the procedure exits. If the file does exist, the procedure continues.

```
APP QUERY abc HOLD
TABLE FILE FOCAPPQ
PRINT * ON TABLE HOLD
WHERE APPNAME = 'abc'
WHERE FILENAME = 'myproc1.fex'
END
-IF &LINES GT 0 THEN GOTO RESUME
-TYPE Procedure Not Found ... exiting!
-EXIT
-RESUME
```

Retrieving Extended Catalog Information

This section provides basic information about querying the server catalogs.

catalog/sysapps

The catalog/sysapps table contains metadata for physical objects on path.

This section only touches on basic uses typically needed by a developer. The Master File on disk robustly describes more attributes than are described here. You can directly study the Master File in order to understand other uses. The catalog/sys* group of files are subject to change (and are usually upwardly compatible). You should never write applications that have specific dependencies (typically on object size), which tend to cause upward compatibility issues.

Example: Listing Files in an APP

The following request lists the application name, application location, file names, and file extensions in the application named *abc*.

```
TABLE FILE SYSAPPS
PRINT APPNAME APPLOC FNAME FEXT
WHERE APPNAME EQ 'abc' ;
END
```

The output (with whitespace selectively removed for readability) is:

```
APPNAME     APPLOC                    FNAME        FEXT
-------     ------                    -----        ----
abc         /usr/wf/ibi/apps/abc      myproc1      fex
abc         /usr/wf/ibi/apps/abc      myproc2      fex
```

The following practical example of using the SYSAPPS table to check file existence checks the existence of the file *myproc1.fex* in the application *abc*. If it does not exist, the procedure exits. If the file does exist, the procedure transfers to the next step in order to continue:

```
TABLE FILE SYSAPPS
PRINT * ON TABLE HOLD
WHERE APPNAME = 'abc' ;
WHERE FNAME =  'myproc1' ;
WHERE FEXT = 'fex' ;
END
-IF &LINES GT 0 THEN GOTO RESUME
-TYPE Procedure Not Found ... exiting!
-EXIT
-RESUME
```

catalog/sysfiles

The catalog/sysapps table contains metadata for app name objects on a path for a select object type. The default is for file type MASTER (Master Files), but is settable for other types. Unless limited in some way, all objects (of the selected type) are displayed.

This section only touches on basic uses typically needed by a developer. The Master File on disk robustly describes more attributes than are described here. You can directly study it in order to understand other uses. The catalog/sys* group of files are subject to change (and are usually upwardly compatible). You should never write applications that have specific dependencies (typically on object size), which tend to cause upward compatibility issues.

Example: **Listing APP MASTER Objects**

The following request lists file names, file names with their application paths, and extensions of files with file type MASTER (the default):

```
TABLE FILE SYSFILES
PRINT FILENAME LGNAME PHNAME EXTENSION
END
```

The output (with some records and whitespace selectively removed for readability) is:

```
FILENAME         LGNAME    PHNAME                          EXTENSION
--------         ------    ------                          ---------
...
mydata           MASTER    baseapp/mydata.mas              mas
mdschema         MASTER    _edahome/catalog/mdschema.mas   mas
```

Example: **Listing APP FOCEXEC Objects**

The following request sets the file type to FOCEXEC and then prints the file names, file names with their application paths, and extensions of files with file type FOCEXEC:

```
SQL FMI SET SYSFILES FOCEXEC
TABLE FILE SYSFILES
PRINT FILENAME LGNAME PHNAME EXTENSION
END
```

The output (with some records and whitespace selectively removed for readability) is:

```
FILENAME         LGNAME    PHNAME                          EXTENSION
--------         ------    ------                          ---------
...
myproc1          FOCEXEC   baseapp/myproc1                 fex
myproc2          FOCEXEC   baseapp/myproc2                 fex
...
```

Note: The value for LGNAME will switch to DEFAULT if the data is limited and only one object returns.

A valid value for the SQL FMI SET SYSFILES command is any valid server file type. Some examples are FOCUS, FOCEXEC, STY, PDF. or ACCESS. For a full list of valid file types, see *Designating File Types for APP Commands* on page 108.

Example: Using the SYSFILES Table to Check File Existence

The following practical example of using the SYSFILES table to check file existence prints the filename *myproc1* with extension *fex* (with the file type set to FOCEXEC). If no lines are returned, the file does not exist and the procedure exits. If the file exists, the procedure transfers to the point at which processing continues.

```
SQL FMI SET SYSFILES FOCEXEC
TABLE FILE SYSFILES
PRINT FILENAME ON TABLE HOLD
WHERE FILENAME = 'myproc1' ;
WHERE EXTENSION = 'fex' ;
END
-IF &LINES GT 0 THEN GOTO RESUME
-TYPE Procedure Not Found ... exiting!
-EXIT
-RESUME
```

Help Information: APP HELP

The APP HELP command provides help information for all of the APP commands.

Syntax: How to Request Help for an APP Command

```
APP HELP command parameters
```

where:

command

Is any valid APP command.

parameters

Are parameters that are available to or required by the command.

Restricting the Use of APP Commands

Server administrators can restrict the ability of other classes of users to change the APP environment. This setting is configurable for Application Administrators and Basic Users.

When this restriction is set, all user interfaces affected by this server setting, such as the Web Console, the Data Management Console, and Managed Reporting, display only applications that are in the effective application search path, instead of displaying all applications defined by the server approot setting. In addition, the use of certain APP commands that users might otherwise issue to bypass the intended controls is restricted.

To ensure full administrative capabilities, the restriction is dynamically switched off when a user who has server administration rights logs on from client software or to the Web Console.

This capability is supported with all security settings.

Procedure: How to Restrict the Use of APP Commands

The server administrator can restrict the use of APP commands by other users.

1. From the *Workspace* menu, select *Access Control*.
2. Expand the *Roles* folder for *Application Administrator* or *Basic* user.
3. Right-click a *user* or *group* and choose *Properties*.
4. Select the *General Privileges* tab.
5. By default, *Change Own Application Path (No applock)* is selected. Deselect this check box to restrict the current user or group ability to use the APP commands listed below.
6. Click *Apply* to confirm the setting.

Once modification of the application path is blocked, use of the following APP commands is restricted for the designated user or group:

- The following commands generate an error message:
  - APP CREATE
  - APP DELETE
  - APP RENAME
- Although no error message is explicitly generated, the following commands are restricted if they reference an application outside of the current application path:
  - APP APPENDPATH
  - APP HOLD
  - APP HOLDDATA
  - APP HOLDMETA
  - APP MAP
  - APP PATH
  - APP PREPENDPATH
  - APP COPY
  - APP COPYF[ILE]
  - APP DELETEF[ILE]
  - APP MOVEF[ILE]
  - APP RENAMEF[ILE]

Accessing Metadata and Procedures

Permanent files include metadata and procedures that were either created before the session by another application or remain after the session is over for use by another application.

Search Rules

Unless a file name is fully qualified with the application name, the search sequence is:

1. Current directory of the agent, which is edatemp/ts*nnnnn*.

2. Applications set using APP HOLDMETA for metadata files, and APP HOLDDATA for hold data files.
3. Applications set in APP PATH (including MVSAPP for z/OS).
4. The *baseapp* application.
5. The EDAHOME/catalog.
6. For stored procedures only: if the file is not found, the server checks to see if the file was allocated with a FILEDEF or DYNAM command, and if so, tries to execute it.

Example: **Search Paths**

The following commands follow the search path, starting with the application set by the APP HOLDMETA command:

```
APP HOLDMETA APP1
```

When a procedure is executed, and referred to by a one-part name

```
EX ABC
```

the following is executed

```
profile.fex in APP1 application
```

followed by

```
EX APP1/ABC
```

If the procedure ABC is not found in APP1, the server follows the standard search path for procedures to find and execute it.

Creation Rules for Procedure Files

Unless a file name is fully qualified or redirected to another location using an APP HOLD, APP HOLDMETA, APP HOLDDATA, FILEDEF, or DYNAM command, it is created in the temporary application area of the agent and disappears after the agent is released.

For example, on z/OS if DYNAM allocation for HOLDMAST or HOLDACC is present, the metadata files are created in the corresponding PDSs (for example, for a CREATE SYNONYM or TABLE FILE file with HOLD).

For related information, see *Output Redirection Commands* on page 112.

Locating Master Files and Procedures

Once your path is set, you can locate Master Files and procedures using the WHENCE command.

Syntax: How to Locate Master Files and Procedures

To locate a Master File or procedure, issue the following command

```
WHENCE filename filetype
```

where:

filename

Is the name of the file you are trying to locate.

filetype

Is the type of file you are trying to locate.

Accessing Existing Data Files

You can allocate existing data files using the following methods:

- DATASET keyword in the Master File.
- FILEDEF command for non-FOCUS data sources (FIXED, RMS, VSAM, XML).
- USE command for FOCUS data sources.
- For the z/OS Server, native operating system services, when supported.
- DYNAM command as a USS equivalent for the MVS ALLOCATE command.
- Superseded by JCL DD card.

It is recommended that you use only one method for each allocation.

Creation Rules for Data Files

For a newly created data file, the location is determined as follows:

1. An application set by APP HOLDDATA applies to all HOLD files.
2. For FILEDEF command, one for each data file.
3. For z/OS, native operating system allocations when supported.

The request that caused the file to be created determines the file DCB parameters, such as record length, record format, and so on.

For related information, see *Output Redirection Commands* on page 112.

Example: Sample Allocations by JCL

The following table contains sample allocations by JCL.

VSAM	`//VSAM01 DD DISP=SHR, DSN=qualif.DATA.VSAM` This type of allocation requires the szero = y parameter in the edaserve.cfg file to support sharing of BufferPool Zero.
Fixed	`//FIX01 DD DISP=SHR,DSN=qualif.FIXED.DATA`
PDS	`//MASTER DD DISP=SHR,DSN=qualif.MASTER.DATA`
FOCUS	`//CAR DD DISP=SHR,DSN=qualif.CAR.FOCUS`

Example: Sample DYNAM Commands

The following table contains samples of the DYNAM command.

VSAM	`DYNAM ALLOC FILE QVASM DA qualif.QVSAM.VSAM SHR REUSE`
Fixed	`DYNAM ALLOC FILE FILE1 DA qualif.FILE1.DATA SHR REUSE`
PDS	`DYNAM ALLOC FILE MASTER DA qualif.MASTER.DATA SHR REUSE`
FOCUS	`DYNAM ALLOC FILE CAR DA qualif.CAR.FOCUS SHR REU`

Syntax: How to Issue a FILEDEF Command

```
FI[LEDEF] filedes DISK app/[appa.../]physfile.ftm
```

where:

filedes

Is a file designation (ddname).

app/[appa...]

Is an application name. It can be a nested application name.

physfile.ftm

Is a physical file located in the application.

Syntax: How to Issue a FILEDEF Command to Concatenate Files

```
FI[LEDEF] concatname DISK [app1/]filename1.ext
FI[LEDEF] name2 DISK [app2/]filename2.ext
...
FI[LEDEF] namen DISK [appn/]filenamen.ext
FI[LEDEF] concatname CONCAT name2 ... namen
```

where:

concatname

Is the ddname for one of the files and the name for the concatenated files. Use this name in a request. The individual ddnames will not be available once they are used in a FILEDEF CONCAT command.

name2 ... namen

Are ddnames for the files that will be added to the concatenation.

app1 ... appn

Are application names. They can be nested application names.

filename1.ext ...filenamen.ext

Are the physical file names.

Example: Concatenating Files Using FILEDEF

The following request creates three files, file1.ftm, file2.ftm, and file3.ftm.

```
APP HOLD app1
TABLE FILE WF_RETAIL_LITE
SUM COGS_US REVENUE_US
BY STATE_PROV_NAME
WHERE STATE_PROV_NAME LE 'F'
WHERE COUNTRY_NAME EQ 'United States'
ON TABLE HOLD AS file1 FORMAT ALPHA
END
-RUN
TABLE FILE WF_RETAIL_LITE
SUM COGS_US REVENUE_US
BY STATE_PROV_NAME
WHERE STATE_PROV_NAME GT 'F' AND STATE_PROV_NAME LE 'M'
WHERE COUNTRY_NAME EQ 'United States'
ON TABLE HOLD AS file2  FORMAT ALPHA
END
-RUN
TABLE FILE WF_RETAIL_LITE
SUM COGS_US REVENUE_US
BY STATE_PROV_NAME
WHERE STATE_PROV_NAME GT 'M'
WHERE COUNTRY_NAME EQ 'United States'
ON TABLE HOLD AS file3  FORMAT ALPHA
END
```

The following commands concatenate the three files.

```
FILEDEF FILE1 DISK app1/file1.ftm
FILEDEF FILE2 DISK app1/file2.ftm
FILEDEF FILE3 DISK app1/file3.ftm
FILEDEF FILE1 CONCAT FILE2 FILE3
```

The following procedure issues a request against the concatenated files.

```
TABLE FILE FILE1
SUM COGS_US REVENUE_US
BY STATE_PROV_NAME
ON TABLE SET PAGE NOPAGE
ON TABLE SET STYLE *
GRID=OFF, SIZE=8,$
END
```

Syntax: How to Issue a FILEDEF Command for a Native MVS Data Set

```
FI filedes DISK "//'NATIVE.MVS.DATASET'"
```

where:

filedes

Is a file designation.

NATIVE.MVS.DATASET

Is a Native MVS data set. It can contain any number of qualifiers, up to 44 characters long.

Syntax: How to Issue a USE command

The USE command can be issued instead of an allocation command for FOCUS data sources. The USE command is the only mechanism for accessing files on the sink machine.

Example: Sample USE Commands

The USE command supports renaming of Master Files and concatenation of data sets. The USE command is the only mechanism for accessing files on the sink machine.

Renaming a Master File

```
USE
 CAR1 AS CAR
END
```

Concatenating Master Files

```
USE
 CAR1 AS CAR
 CAR2 AS CAR
END
```

Accessing Files on a Sink Machine

```
USE
 CAR1 ON FOCSU01
END
```

Data Set Names

If a data set name satisfies one of the following conditions, the server assumes that it is an MVS file name:

- Data set name starts with "//".

- Data set name contains no "/" and contains at least one "."

In all other cases, the name is interpreted as an HFS file name.

Syntax: How to Define a Data Set

The following syntax is supported:

```
DATASET=APP1/physfile.ftm
DATASET='qualif.car.data'
DATASET=qualif.car.data
```

In addition, on z/OS, you can use the following:

GDG files	`FILENAME=CARGDG,SUFFIX=FOCUS,` `DATASET='qualif.CARGDG.FOCUS(0)'`
PDS members	`FILENAME=CARMEMB,SUFFIX=FOCUS,` `DATASET=qualif.CARPDS.DATA(CARMEMB)`
FOCUS, VSAM, Fixed	`FILENAME=CAR,SUFFIX=FOCUS,` `DATASET=//'qualif.CAR.FOCUS'`

Allocating Temporary Files

Temporary files are transient files that disappear after you end a session. By default, all temporary data files (for HOLD and FOCSORT files) and temporary metadata files, such as temporary Master Files and Access Files, are created in the temporary area of an agent, which corresponds to your TSCOM ID. For example, if your TSCOM ID is TS000001, your temporary files are located in //edatemp/ts000001.

For z/OS, you can control the size and location of these temporary metadata files and data files. You can specify that the temporary files reside in the hierarchical file system, MVS data sets, or in hiperspace.

Syntax: How to Allocate Temporary Files

To specify the allocation of your temporary files, issue the following command

```
DYNAM SET TEMP[ALLOC] {HFS|MVS|HIPER}
```

where:

HFS

Allocates temporary files to the hierarchical file system. HFS is the default value.

`MVS`

Allocates temporary files to MVS data sets.

`HIPER`

Allocates temporary files to hiperspace.

Reference: Usage Notes for Allocating Temporary Files

For z/OS, temporary metadata files can be allocated using a similar procedure to allocating permanent metadata files:

- If DYNAM allocation for HOLDMAST or HOLDACC is present, temporary files are stored in the designated PDSs.
- If DYNAM SET TEMP[ALLOC] MVS is issued; in the default temporary PDSs.
- If DYNAM SET TEMP[ALLOC] HIPER is issued; in the HIPERSPACE.

Syntax: How to Allocate Temporary Files to MVS Data Sets

To alter the default allocation parameters for temporary files for MVS data sets, issue the following command

```
DYNAM SET TEMP[ALLOC] FOR type dynam_parms
```

where:

`type`

Is one of the following: HOLDACC, HOLDMAST, HOLD SAVE, REBUILD, FOCUS, FOCSORT, OFFLINE, or FOC$HOLD.

`dynam_parms`

Are regular DYNAM ALLOC parameters to be used as default for that type. Note that DCB parameters, if provided here, will be ignored, since they must be compatible with the file type being written.

This is similar to the functionality of IBITABLA in the SSCTL Server. The defaults should be overwritten for all cases when, in older versions, a private copy of IBITABLA existed containing different values.

Reference: System Defaults for Allocating Temporary Files to MVS Data Sets

System defaults for HOLDMAST and HOLDACC are:

```
TRKS 5  5 DSORG PO DIR 36 NEW REU
```

System defaults for all other types are:

```
CYLS 5 10 DSORG PS NEW REU
```

Syntax: **How to Support Long Synonym Names Using DYNAM SET LONGSYNM**

The server supports synonym names up to 64 characters. However, PDS member names cannot exceed eight characters. The server accounts for this operating environment limitation with the command DYNAM SET LONGSYNM.

A synonym comprises a Master File and, usually, an Access File. When you create a synonym with a name exceeding eight characters, the LONGSYNM setting currently in effect determines how the long name of the Master File and of the Access File will be handled.

You can issue DYNAM SET LONGSYNM anywhere SET commands are valid, including the global server profile (edasprof.prf) and a stored procedure (FOCEXEC).

DYNAM SET LONGSYNM, on servers running on z/OS, corresponds functionally to the server configuration file keyword LONGSYNM on servers running on MVS.

The syntax is

```
DYNAM SET LONGSYNM {HFS|MVS|MATCH}
```

where:

HFS

Specifies that each synonym whose name is longer than eight characters will be created in an HFS directory. This is the default.

MVS

Specifies that when you save a synonym with a name exceeding eight characters, the server truncates the name, preserving up to the first six characters, followed by a left curly brace ({) and a suffix number that ensures the name is unique. (The server preserves the original long name within the synonym files.)

For example, if you create a Master File named VERYLONGNAMETEST, it will be saved as VERYLO{0. If you then create a Master File named VERYLONGNAMEPROD, it will be saved as VERYLO{1.

The server chooses a suffix number by taking the next unused number in the sequence for that truncation of a Master File or Access File name. If the next number available for the Master File is different than that available for the Access File, the files will be created with different numbers. For example, if the highest Master File name truncated to VERYLO is VERYLO{8, and the highest Access File name truncated to VERYLO is VERYLO{5, and you create a synonym specifying the name VERYLONGNAMEAGAIN, the new Master File will be saved as VERYLO{9, and the new Access File will be saved as VERYLO{6.

`MATCH`

Works the same as the MVS setting, except that it ensures that the truncated names of a Master File and Access File synonym will always match. That is, they will be named using the same suffix number.

In the example provided for the MVS setting, if SET LONGSYNM had instead been set to MATCH, both the new Master File and the new Access File would have been named VERYLO{9.

Matching names may be a convenience for some people if they manually manage synonym files. It is less efficient than the MVS setting, however.

Temporary Space Usage and Location

The server uses temp space for the various processes. This space may be used for anything from logs and traces to transient work files (such as HOLD files). This temp space uses real disk space, but is recycled and cleaned up based on server configuration (for example, secured servers clean agent directories upon disconnect). Ultimately, a server restarts also cleans up this temporary space. If a server has been turned off, and a restart is not planned (possibly because it was a test configuration), the space may be recovered by the server admin with the following command:

```
edastart -cleardir
```

Temporary Disk Space Usage for Non-PDS Deployment

Temporary space, by default, is created under the configuration directory (ffs, wfs, or dm) in the file system used to install the product. The edatemp directory is two levels deep.

The first level directory is used for trace files if tracing is active. By default, the directory is named edatemp and is also generally referred to as the edatemp directory of EDACONF.

The subdirectory is used for temporary files and traces created by various processes, such as the TCP and HTTP listeners, the Workspace Manager, and the end-user agent processes. Each connecting user is assigned to a private data agent directory to isolate their work from other users, and that directory is cleaned up upon disconnect. The agent subdirectories are named ts*nnnnnn*, where *nnnnnn* is the ID number assigned to the data agent.

The edatemp directory can be configured to use a separate file system by setting the EDATEMP variable to point at the alternate location. The steps and syntax vary by platform, and may be done as an operating system variable or as a variable in the server configuration file (edaserve.cfg) before server start up.

If you are using the server configuration file, the best method is to edit edaserve.cfg from the Web Console *Workspace* folder, adding the line:

```
edatemp = directory
```

where

directory

Is a full path directory name, using the appropriate platform syntax. After saving, restart the server.

Note: Editing the server configuration file (edaserve.cfg) is the only method available for Windows.

Syntax: How to Specify the EDATEMP Variable

For UNIX and Linux, the edatemp variable can be coded in the edastart.sh file:

```
export EDATEMP=/u/iway/edatemp
```

Note: This also applies to IBM i, VMS, and Windows.

For OpenVMS, the edatemp variable can be coded in the edastart.com file:

```
DEFINE EDATEMP TMP:[TMP.EDATEMP]
```

For UNIX System Services, it can be coded in the ISTART JCL member under the EDAENV dd statement:

```
//EDAENV      DD   *
EDATEMP=/u/iway/edatemp
```

For z/OS UNIX System Services only, you can set the temporary area to MVS by using the following command in edasprof.prf or user profile.

```
DYNAM SET TEMPALLOC MVS
```

The above are global settings which affect temporary file allocations for all users.

It can be specified in the edaserve.cfg file.

```
edatemp=/u1/home/temp
```

Syntax: How to Pre-Allocate Temporary Files

You can pre-allocate an individual file for a user, using the following techniques:

- **For UNIX and Linux:**

  ```
  FILEDEF XXX DISK /u/another/area/xxx.dat
  ```

  where:

  `/u/another/area`

  Has enough free space to hold the file.

- **For Windows:**

  ```
  FILEDEF XXX DISK C:\tmp\another\area\xxx.dat
  ```

  where:

  `tmp\another\area`

  Has enough free space to hold the file.

- **For OpenVMS:**

  ```
  FILEDEF XXX DISK TMP:[ANOTHER.AREA]XXX.DAT
  ```

  where:

  `TMP:[ANOTHER.AREA]`

  Has enough free space to hold the file.

- **For z/OS UNIX System Services:**

  You can use the same filedef syntax indicated for UNIX, or you can use the DYNAM command to direct the temporary file to MVS.

  ```
  DYNAM ALLOC FILE XXX SPACE
  ```

  Note that, for the DYNAM command, you must to specify the amount of space required.

- **For FOCUS files:**

You can use the FILEDEF and USE commands to create a FOCUS file outside the edatemp area.

```
FILEDEF NAME DISK /{pathname}{filename}.foc ......
USE NAME NEW
END
```

Syntax: **How to Dynamically Allocate FOCUS Files on z/OS**

You can dynamically allocate FOCUS files on z/OS with the USE command. The command is

```
DYNAM ALLOC FILE ddname SPACE
USE ddname AS masterfile
END
```

where:

ddname

Is the DDNAME.

masterfile

Is the Master File name.

If the DDNAME and Master File name are the same, use just the command:

```
DYNAM ALLOC
```

Application Tools

An application program may be as simple as a TABLE FILE ... END request or as elaborate as a series of replaceable variables (known as amper variables) controlled by dynamic inputs, GOTO statements, and/or looping logic (known collectively as Dialogue Manager). While Dialogue Manager is fully detailed in the *Stored Procedure Reference* manual, a limited set of these commands and tools are also noted here due to their integral use in APP management.

EX Procedure and Amper Variables

The EX procedure executes another procedure, as found on the application path. The procedure extension (on non-PDS platforms) is not required, and is assumed to be .fex. The full syntax is:

```
EX[EX] [appname/]procedure[.fex] [parameter=value[,parameter=value][,...]
```

Parameters may be names (such as LASTNAME) or numeric (such as 1), and may be mixed within the list. A specific APP name can be referenced and does not require the APP to be on the application path prior to execution. Numerics can also be assumed positionally, as in this example:

```
EX mytest DETAIL=ALL,QUARTERLY,PERSON=JAMES MADISON,5=MANAGER
```

and the procedure would contain amper variables (parameters) for &DETAIL, &2, and &PERSON, and would use the context of:

```
TABLE FILE ...
WHERE PERSON IS '&PERSON' AND PERIOD EQ '&2'
...
END
```

There are two types of amper variables, single and double. Single amper variables are only active within the procedure they are created in. Double amper variables are good for the life of the session. Additionally, there are predefined, single amper variables, such as &LINES (records returned from a request), &FOCUSER (current user ID), &FOCFOCEXEC (name of the current executing procedure), &FOCCODEPAGE (the server NLS code page), and many others. See the *Stored Procedure Reference* manual for a detailed listing of other pre-defined amper variables.

EX EDAMAIL

EDAMAIL is an internal procedure that allows you to send emails from a procedure with the content of a specific file included in the email body or as an attachment. This is useful for sending email alerts for events ranging from special error conditions to simple report completion. EDAMAIL requires a configured (and working) email SMTP Server node for the server workspace.

There are two forms of syntax for EDAMAIL:

- The original form uses positional parameters, and is limited to the original specification.
- The extended form uses non-positional, name-value pair parameters. The extended form supports additional SMTP email client features such as Reply To and Importance.

The extended form is assumed if the first parameter contains an equal sign (=) indicating a name-value pair is being passed.

The basic form of email address (that is, foo@foo.com or <foo@foo.com>) may be used in either the original or the extended form of EDAMAIL. Extended forms of email addressing (such as My Support <support@foo.com>, "My Support" <support@foo.com>, "My Support <support@foo.com>", and "Support, Me <support@foo.com>") are only supported in the extended form of EDAMAIL.

Multiple addresses may be use for parameters that support multiple addresses (for example, *to*) when properly formed and delimited.

For multiple addresses and basic EDAMAIL, the separator must be a semi-colon (;), and the overall string may be, optionally, enclosed in double quotation marks, but remember that basic EDAMAIL only supports simple addresses.

For multiple addresses and extended EDAMAIL, the separator is normally a semi-colon (with exceptions where a comma may be used), plus the overall string must be enclosed in single quotation marks if any of the addresses contains a comma (for example, Support, Me <support@foo.com>). A comma may be used as separator if the email addresses are all simple addresses (that is, foo@foo.com or <foo@foo.com>), plus the overall string must be enclosed in single or double quotation marks.

The actual use and display of the email headers created by EDAMAIL are a function of the SMTP Server in use (and any intervening SMTP hops), plus the user email client (such as Outlook, Gmail, Hotmail, and so on). They are not directly controlled by the sender email headers. Therefore, an email may not always exhibit the expected behavior in all email client environments. For example, older email clients may not recognize newer header types, and may ignore Reply To. Any functional issues should first be investigated by an experienced SMTP/Email administrator to determine if they are client-related.

Syntax:

How to Use EX EDAMAIL in the Positional Parameter Form

The syntax is:

```
EX EDAMAIL to, cc, bcc,from,  subject, [flag], filetype, data
```

where:

to

Is the email recipient. Multiple addresses are allowed, using a semi-colon as the address separator.

cc

Is the carbon copy recipient. Multiple addresses are allowed, using a semi-colon as the address separator.

bcc

Is the blind carbon copy recipient. Multiple addresses are allowed, using a semi-colon as the address separator.

from

Is the email sender.

subject

Is the email subject string. If the subject string contains a comma, the string must be enclosed in single or double quotation marks.

flag

If set to a or A, the file is sent as an attachment. Otherwise, it is included as the body of the email. All other values are ignored. The value is also ignored if the filetype parameter is blank.

filetype

Defines the data file type for an email message body that uses a file. Any application file type is valid, including MASTER, FOCEXEC, HTML, TEXT, and so on. Leave the parameter empty to use the inline email message body feature.

data

Is the inline data stream for the email message body or the [app/]filename file containing the email message body. The inline data stream feature requires an empty filetype parameter. The EDAMAIL feature can also spread an inline email message body on to multiple lines using the -LINES {n|*} feature.

If an inline data stream message body is spread across multiple lines in the procedure, the resulting email is a single line of output. Multi-line message bodies are respected when the message body from a file option is used.

Example: Mailing an HTML File as Message Body

```
TABLE FILE file1
PRINT A B C
ON TABLE HOLD AS MYFILE FORMAT HTML
END
EX EDAMAIL  user1@corp1.com, user2@corp1.com, File1 Report,,HTML, MYFILE
```

Example: Mailing an HTML File as a Message Attachment

```
TABLE FILE file1
PRINT A B C
ON TABLE HOLD AS MYFILE FORMAT HTML
END
EX EDAMAIL  user1@corp1.com, user2@corp1.com, File1 Report,a,HTML, MYFILE
```

Example: Mailing a Multi-line Inline Message

```
...
EX -LINES * EDAMAIL user1@corp1.com, user2@corp1.com, &SUBJECT,,,
    Run result for &TESTNAME is:
&RESULT
EDAMAIL*
```

Syntax: How to Use EX EDAMAIL in the Extended Form

The EDAMAIL extended form using name-value pairs is activated by the detection of an equal sign (=) in the first parameter. Parameter names are not case sensitive and may be in any order, but the message parameter (if used) must be last.

The syntax is:

```
EX EDAMAIL to=addresslist,
[toaddr= {addresslist|%[app/]addresslist.fex}, ]
[cc=cadrlist1,]
[ccaddr=cadrlist2,]
[bccaddr=bcadrlist,] [from=address,]
[fromaddr=address,] [replyto=address,] [importance=low|normal|high,]
[subject=string,] [flags=value,] [filetype=extension,]
[filename=file,] [message=body message]
```

where:

`to=addresslist`

Is the email recipient displayed by the user email client (and used in an email client Reply To All, if a toaddr header is not supplied). Multiple addresses are allowed, using a semi-colon as the address separator. A comma is also allowed as a separator, if the overall address string is enclosed in single quotation marks.

If the to parameter is used in conjunction with toaddr, the value of *to* may be an arbitrary string, such as *Group Managers*, which most email clients will display as a pseudo title in the To field, and will not display the actual addresses used in the toaddr parameter. A forced blank can be supped for the To field by using *to*="".

`toaddr=addresslist`

Is the email recipient. If not supplied, SMTP servers will use the *to* header. Email clients will use the toaddr value in an email client Reply To All. Multiple addresses are allowed, using a semi-colon as the address separator. A comma is also allowed as a separator, if the overall address string is enclosed in single quotation marks.

`toaddr=%[app/]addresslist.fex`

Is a FOCEXEC that generates an email recipient list at run time. Email clients will use the toaddr value in an email client Reply To All. The actual FOCEXEC may go against any data source for the addresses, but must PRINT a single column and use ON TABLE PCHOLD FORMAT COMT FORMATTED syntax to format the data (as one email address enclosed in double quotation marks per row).

The actual FOCEXEC extension must be .fex in the specification. This feature also only works in a running server context. It is not supported in any other mode.

`cc=cadrlist1`

Is the carbon copy recipient displayed by the user email client (and used in an email client Reply To All, if a ccaddr header is not supplied). Multiple addresses are allowed, using a semi-colon as the address separator. A comma (,) is also allowed as a separator, if the overall address string is enclosed in single quotation marks (').

If the cc parameter is used in conjunction with ccaddr, the value of cc may be an arbitrary string, such as *Group Managers*, which most email clients will display as a pseudo title in the Cc field, without displaying the actual addresses used in the ccaddr parameter. A forced blank can be supped for the Cc field by using *to=""*.

`ccaddr=cadrlist2`

Is the carbon copy recipient. If not supplied, SMTP servers will use the cc header. Email clients will use the ccaddr value in an email client Reply To All. Multiple addresses are allowed, using a semi-colon as the address separator. A comma (,) is also allowed as a separator, if the overall address string is enclosed in single quotation marks (').

`bccaddr=bcadrlist`

Is the blind carbon copy recipient. Email clients will use the bccaddr value in an email client Reply To All. Multiple addresses are allowed, using a semi-colon as the address separator. A comma (,) is also allowed as a separator, if the overall address string is enclosed in single quotation marks (').

`from=address`

Is the email sender displayed by the email client (and used in an email client Reply, unless overridden by a fromaddr or Reply To header). If the from parameter is used in conjunction with fromaddr, the from value may be an arbitrary string, such as *The Boss*, which most email clients will display as a pseudo title in the From field, and will not display the actual address used in the fromaddr parameter.

`fromaddr=`*`address`*

Is the email sender. If not supplied, most email clients will use the From header when doing a reply.

`replyto=`*`address`*

Is the email sender. If not supplied, most email clients will use the fromaddr or from parameter value.

`importance=`*`low|normal|high`*

Is the email importance level for email clients that support importance flags. Valid values are high, normal, and low.

`subject=`*`string`*

Is the email subject string. If the subject string contains a comma, the string must be enclosed in single or double quotation marks.

`flags=`*`value`*

If set to a or A, the file is sent as an attachment. Otherwise, it is included as the body of the email.

`filetype=`*`extension`*

Defines the data file type for an email message body that uses a file as the body. Any application file type is valid, including MASTER, FOCEXEC, HTML, TEXT, and so on. Leave the parameter out to use the inline email message body feature.

`filename=`*`file`*

Defines the data file for an email message body that uses a file as the body. Leave the parameter out to use the inline email message body feature.

`message=`*`body message`*

Is the inline data stream containing the email message body. If used, it must be the last parameter in the EDAMAIL command. To use the inline data stream feature, the filetype and filename parameters cannot be supplied. The data stream may also be spread onto multiple lines if EDAMAIL is used with the EX -LINES {n|*} feature.

If an inline data stream message body is spread across multiple lines in the procedure, the resulting email is a single line of output. Multi-line message bodies are respected when the message body from a file option is used.

Example: Mailing an HTML File as a Message Body Using Multiple Addresses and Extended Form

```
TABLE FILE file1
PRINT A B C
ON TABLE HOLD AS MYFILE FORMAT HTML
END
EX EDAMAIL to=Managers,toaddr=user1@corp1.com;user2@corp1.com,
from=support1@corp1.com,subject=File1 Report, filetype=HTML,
filename=MYFILE
```

Chapter 3

Coding a User Interface

This topic describes how to code a user interface for self-service applications. You will see how to call WebFOCUS from a hyperlink or form on a launch page to run a report. A form may contain one or more controls that prompt for values required by a procedure.

JavaScript adds dynamic features to a user interface. This topic illustrates the use of JavaScript to check for a required entry on a form.

You will also see how to use the Dialogue Manager command -HTMLFORM to embed report output on a display page that you design.

User interface templates are supplied with your WebFOCUS software. They include a variety of launch pages that call different types of reports. You can modify these templates for your application.

Additional coding capabilities are described in *Enhancing a User Interface* on page 259.

In this chapter:

- Which Tools Can You Use?
- The WebFOCUS Client
- Using the Servlet
- Using a Dynamic Multiselect Drop-Down List
- Enabling Ad Hoc Reporting
- Validating a Form With JavaScript
- WebFOCUS Autoprompt Facility
- Responsive Autoprompt
- HTML Autoprompt
- Autoprompt Considerations
- Defining Parameter-Based Filters
- Customizing the Autoprompt Facility
- Displaying a Report on the Default WebFOCUS Page
- Designing an HTML Page for Report Display

Which Tools Can You Use?

You can manually code an HTML-based user interface page using:

- **HTML Canvas.** This allows you to graphically create an HTML page that incorporates WebFOCUS forms, reports, graphs, and web objects.
- The App Studio text editor.

- **A standard text editor of your choice.** With this option, you can copy or move a user interface page to a project directory in App Studio.

Choosing Search Path Logic

You can set either EDAPATH or APP PATH logic in the Reporting Server global profile (EDASPROF.PRF), a user profile, or a procedure. For either kind of logic, make sure that the following files are accessible by the Reporting Server:

- Procedures.
- WebFOCUS StyleSheets.
- Custom HTML display pages called by the -HTMLFORM command.
- Images embedded in report output.

The directories in which the preceding files reside must be included on EDAPATH or APP PATH, depending on the logic used.

Procedure: How to Use APP PATH Logic

If you create a user interface page with a text editor, do the following.

In the interface page, set the variable IBIAPP_app as follows:

```
<INPUT TYPE="HIDDEN" NAME="IBIAPP_app" VALUE="app_directory">
```

where:

app_directory

Is the directory on the Reporting Server to which procedures are deployed.

For a full description of all commands related to APP PATH logic, see *Managing Applications* on page 37.

Example: Coding APP PATH Logic

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. In a project named SALESDIR, create a procedure named SALESSE. It reports on total unit and dollar sales in the Southeast for a category requested by the user.

   Procedure: SALESDIR.FEX

```
TABLE FILE GGSALES
HEADING
"&CATEGORY Sales for Southeast"
SUM UNITS AND DOLLARS
WHERE (CATEGORY EQ '&CATEGORY') AND (REGION EQ 'Southeast');
END
```

2. Using a third-party text editor such as Notepad, manually code a launch page named LAUNCHSE. It contains a form that uses the Servlet to call WebFOCUS to run SALESSE. A one-line text box prompts the user for a value for CATEGORY.

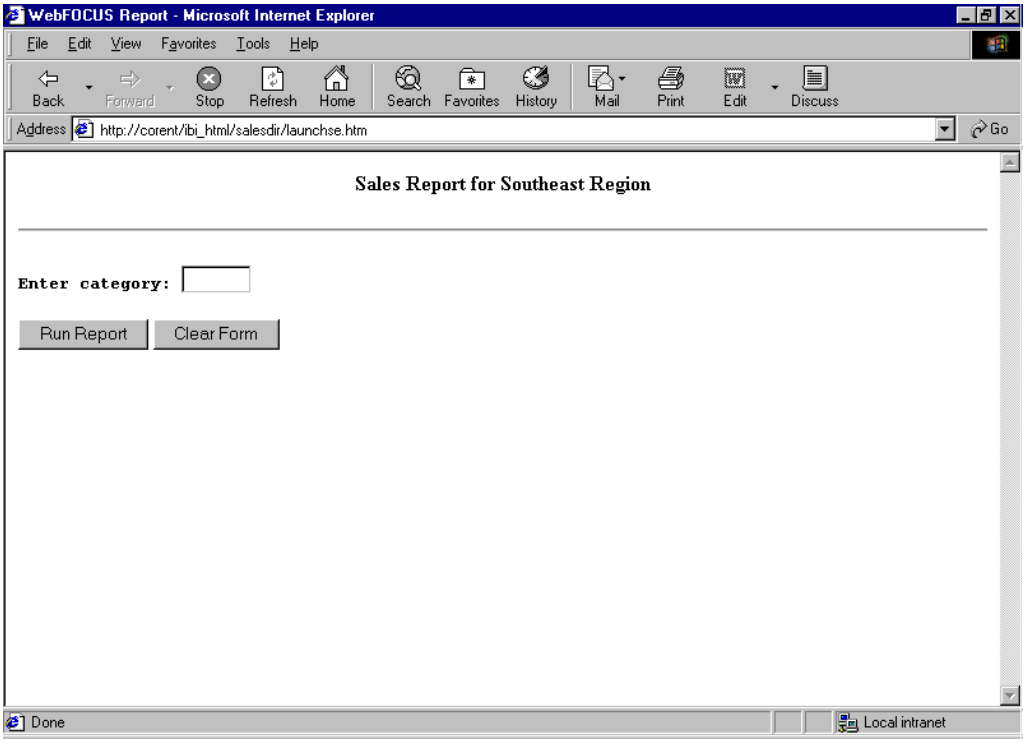
   The launch page contains code that sets the variable IBIAPP_app to the directory (SALESDIR) in which the procedure (SALESSE) resides after publishing.

   Launch Page: LAUNCHSE.HTML

```
<HTML>
<HEAD>
<TITLE> WebFOCUS Report </TITLE>
</HEAD>
<BODY>
<H4 ALIGN=CENTER>Sales Report for Southeast Region</H4>
<HR>
<FORM ACTION="/ibi_apps/WFServlet" METHOD="get">

<INPUT TYPE="HIDDEN" NAME="IBIAPP_app" VALUE="SALESDIR">

<INPUT NAME="IBIF_ex" VALUE="salesse" TYPE="hidden">
<P ALIGN=LEFT NOWRAP><PRE>
<B>Enter category: </B><INPUT NAME="CATEGORY" TYPE="text" SIZE="6">
</PRE></P>
<INPUT NAME="submit" TYPE=SUBMIT VALUE="Run Report">
<INPUT NAME="reset" TYPE=RESET VALUE="Clear Form">
</FORM>
</BODY>
</HTML>
```

3. Publish the procedure and launch page using App Studio.

4. Run the launch page in your browser. It will look similar to this:



5. In the Enter category field, type:

   Food

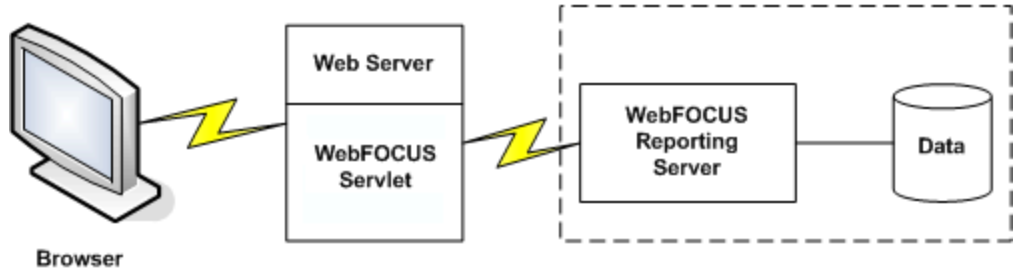
6. Click *Run Report* to receive the report.

PAGE 1	
Food Sales for Southeast	
Unit Sales	Dollar Sales
349829	4308731

The WebFOCUS Client

When WebFOCUS is installed, it is configured to invoke the WebFOCUS client using the WebFOCUS Servlet:

The WebFOCUS Servlet resides on the same machine as the web server and routes requests from that server to the WebFOCUS Reporting Server, as the following diagram illustrates:



This configuration includes complete support for WebFOCUS features and accepts variables from the standard WebFOCUS script (WFS) files, installed by default in the WebFOCUS Client.

Using the Servlet

If you are using the Servlet to run a dynamic report and pass variable values, you can call it:

- From an HTML hyperlink to run a procedure without passed values, or with fixed values.
- From an HTML form to run a procedure with user-supplied values. Provide controls on the form, such as radio buttons and check boxes, which prompt for the values.

Reference: Requirements for WebFOCUS Servlet Configuration

Your web server must be servlet-enabled and have servlet support. The following types of servlet support are acceptable:

- Web server with native servlet support (for example, iPlanet™ or WebSphere®).
- Web server with a servlet engine plug-in (for example, ServletExec™ or JRun™).
- Web Application Server (for example, WebSphere, WebLogic®, or SilverStream®).

Syntax: How to Call the Servlet From a Hyperlink

```
<A HREF="/alias/WFServlet?
IBIF_ex=procedure[&var=value[&var=value]...]">text</A>
```

where:

alias

Points to the directory in which the WebFOCUS Servlet is located. A web server uses an alias to provide a logical name for a physical directory. The default alias is ibi_apps. It is set during WebFOCUS installation and configuration.

To call the WebFOCUS Servlet on another web server, specify a fully qualified URL, such as:

```
http://web_server/ibi_apps/WFServlet...
```

procedure

Is the name of the procedure to run.

var=value

Is a variable and its corresponding value. The call can include both HTTP environment variables and application variables passed to the procedure.

You can pass more than one variable-value pair, but do not include a space between pairs. Use an ampersand (&) as a delimiter to separate each variable-value pair.

If a value contains an embedded blank, substitute a plus sign (+) or the characters %20 for the blank.

text

Is the text on the launch page that serves as the hyperlink that runs the procedure.

Note: The maximum length of the URL can be 4K or 4,096 bytes. There is no limit on the number of the parameters, but their total length cannot exceed 4K.

Example: Calling the Servlet From a Hyperlink

This example creates a launch page with four hyperlinks. Each hyperlink calls the Servlet to pass:

- The name of a procedure to run.
- Two values that the procedure requires.

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create a procedure named CSALES. It generates a dynamic report that shows total unit and dollar sales for a category and region.

   Procedure: CSALES.FEX

```
TABLE FILE GGSALES
HEADING
"&CATEGORY Sales for &RGN Region"
SUM UNITS AND DOLLARS
WHERE (CATEGORY EQ '&CATEGORY') AND (REGION EQ '&RGN');
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
```

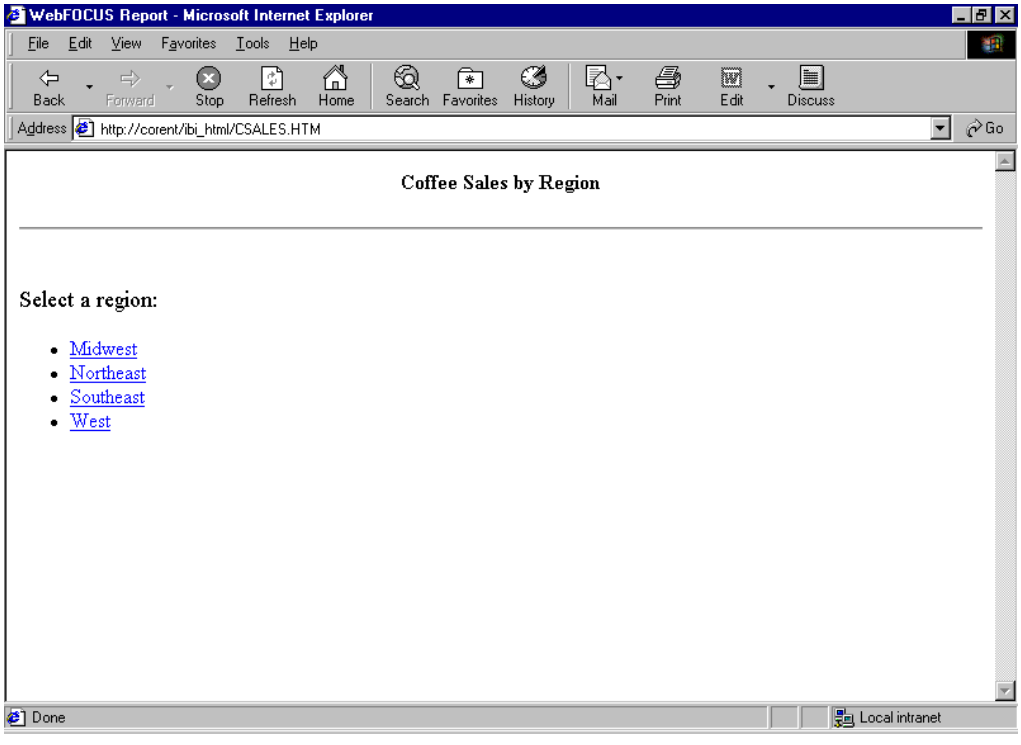
2. Create a launch page named CSALES. Each instance of the A HREF code calls the WebFOCUS Servlet and passes it the name of the procedure, CSALES. Each instance also passes the value Coffee for CATEGORY, and a value for RGN (Midwest, Northeast, Southeast, or West). The hyperlinks on the launch page prompt the user for the region of interest.

   Launch Page: CSALES.HTM

```
<HTML>
<HEAD>
<TITLE> WebFOCUS Report </TITLE>
</HEAD>
<BODY>
<H4 ALIGN=CENTER>Coffee Sales by Region</H4>
<HR>
<P><FONT SIZE=+2></FONT><BR>
<FONT SIZE=+1>Select a region:</FONT>
</P>
<UL TYPE=SQUARE>
<LI>

<A HREF="/ibi_apps/WFServlet?IBIF_ex=csales&CATEGORY=
Coffee&RGN=Midwest">Midwest</A>

<LI>

<A HREF="/ibi_apps/WFServlet?IBIF_ex=csales&CATEGORY=
Coffee&RGN=Northeast">Northeast</A>

<LI>

<A HREF="/ibi_apps/WFServlet?IBIF_ex=csales&CATEGORY=
Coffee&RGN=Southeast">Southeast</A>

<LI>

<A HREF="/ibi_apps/WFServlet?IBIF_ex=csales&CATEGORY=
Coffee&RGN=West">West</A>
```

```
</UL>
</BODY>
</HTML>
```

3. Run the launch page in your browser. It will look similar to this:



4. Click the hyperlink *Midwest* to receive the report:

Coffee Sales for Midwest Region

Unit Sales	Dollar Sales
332777	4178513

Syntax: How to Call the Servlet From a Form

```
<FORM ACTION="/ibi_apps/WFServlet" METHOD="get">
<INPUT NAME="IBIF_ex" VALUE="procedure" TYPE="hidden">
    .
    .
    .
</FORM>
```

where:

procedure

Is the name of the procedure to run. The call can include both HTTP environment variables and application variables passed to the procedure.

Reference: Conventions for Creating a Form

- To begin and end a form, use the HTML <FORM tag.
- To specify the name of the procedure to run, use the HTML <INPUT tag.
- Include the desired controls on the form, such as text boxes, check boxes, or drop-down lists.
- Include a Submit button to execute the Servlet and pass the values collected on the form to the procedure.

Example: Calling the Servlet From a Form

This example creates a launch page with a form that calls the Servlet to pass:

- The name of a procedure to run.
- Two values that the procedure requires. The form prompts the user for the first value with a one-line text box, and for the second value with radio buttons.

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. If you have not done so in a previous example, create a procedure named CSALES. It generates a dynamic report that shows total unit and dollar sales for the category and region requested by the user.

   Procedure: CSALES.FEX

```
TABLE FILE GGSALES
HEADING
"&CATEGORY Sales for &RGN Region"
SUM UNITS AND DOLLARS
WHERE (CATEGORY EQ '&CATEGORY') AND (REGION EQ '&RGN');
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
```

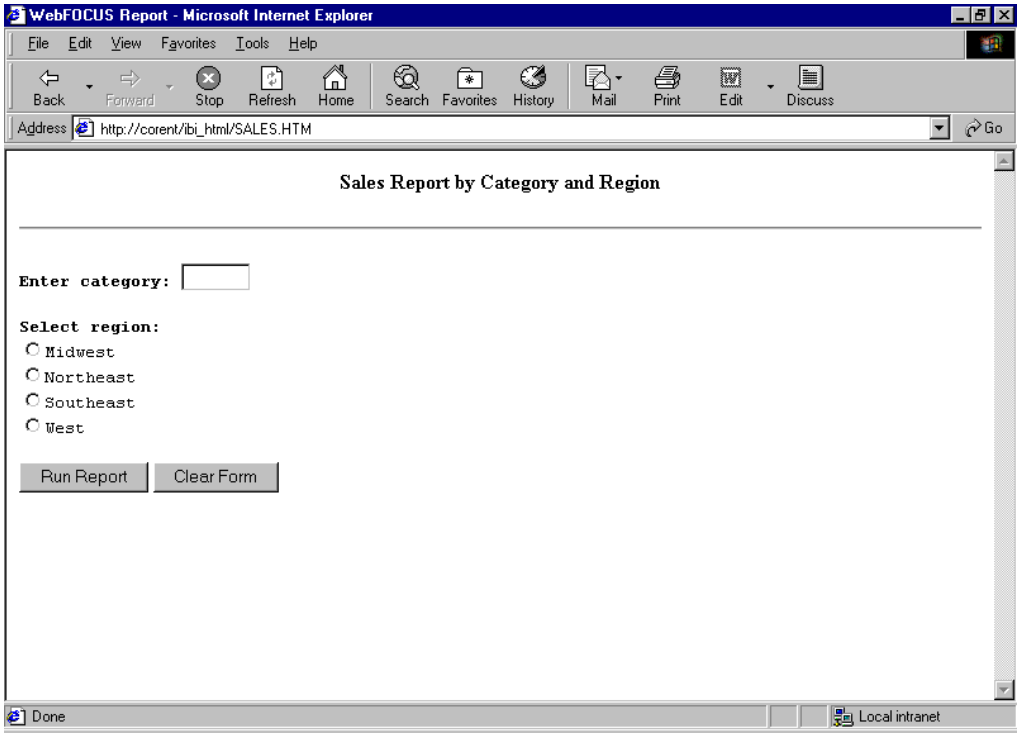
2. Create a launch page named SALES.HTM. It contains a form that calls the Servlet to pass the name of the procedure, CSALES. A one-line text box prompts the user for a value for CATEGORY, and radio buttons prompt for a value for RGN.

   The form also contains a Submit button (labeled Run Report) and a Reset button (labeled Clear Form). When you prompt for user input on a form, you must include a Submit button.

   Launch Page: SALES.HTM

```
<HTML>
<HEAD>
<TITLE> WebFOCUS Report </TITLE>
</HEAD>
<BODY>
<H4 ALIGN=CENTER>Sales Report by Category and Region</H4>
<HR>

<FORM ACTION="/ibi_apps/WFServlet" METHOD="get">
<INPUT NAME="IBIF_ex" VALUE="csales" TYPE="hidden">
<P ALIGN=LEFT NOWRAP><PRE>
<B>Enter category: </B><INPUT NAME="CATEGORY" TYPE="text" SIZE="6">
</PRE></P>
<P><PRE><B>Select region: </B>
<INPUT NAME="RGN" TYPE=RADIO VALUE=Midwest>Midwest
<INPUT NAME="RGN" TYPE=RADIO VALUE=Northeast>Northeast
<INPUT NAME="RGN" TYPE=RADIO VALUE=Southeast>Southeast
<INPUT NAME="RGN" TYPE=RADIO VALUE=West>West
</PRE></P>
<P>
<INPUT NAME="submit" TYPE=SUBMIT VALUE="Run Report">
<INPUT NAME="reset" TYPE=RESET VALUE="Clear Form">
</P>
</FORM>

</BODY>
</HTML>
```

3. Run the launch page in your browser. It will look similar to this:



4. In the Enter category field, type:

   ```
   Coffee
   ```

5. Select the *Southeast* radio button.
6. Click *Run Report* to receive the report.

Coffee Sales for Southeast Region

Unit Sales	Dollar Sales
350948	4415408

Using a Dynamic Multiselect Drop-Down List

You can add a dynamic multiselect drop-down list whose values are populated by field values retrieved live from a data source. The data source is allocated dynamically with no fixed maximum record length.

Procedure: How to Add a Dynamic Multiselect Drop-Down List

This procedure uses the Dialogue Manager command –HTMLFORM to populate a drop-down list with values from a data source. Use this technique when selection values change frequently, to ensure that you always derive a list of current valid values.

For details on -HTMLFORM, see *Designing an HTML Page for Report Display* on page 231.

1. Create a procedure that populates a drop-down list:
   a. Allocate a text file that will contain the dynamic values.

      For example, on Windows, the command is

      ```
      FILEDEF textfile DISK APP/textfile.TXT
      ```

      where:

      textfile
      Is the name of the file that contains the values.

      APP

      Is the name of the application, under APPROOT, that contains the physical file.

      See *Defining and Allocating WebFOCUS Files* on page 651 for details on platform-specific commands.

   b. Include a DEFINE command to create a temporary field that identifies the values.
   c. Include the following command to save the values to the allocated file.

      ```
      ON TABLE HOLD FORMAT ALPHA as textfile
      ```

      where:

      textfile

      Is the name of the file. The name can be from 1 to 8 characters.

   d. Include the following command to send the report output to the launch page that displays the values.

      ```
      -HTMLFORM valuespg
      ```

      where:

      valuespg

      Is the name of the launch page.

2. Create the launch page. Use a comment to indicate where the values display

   ```
   !IBI.FIL.textfile;
   ```

or

```
<!--WEBFOCUS TABLE textfile-->
```

where:

textfile

Is the name of the file that contains the values.

Note: The HTML comment line must be closed with the —> characters or a single > character and should not have any other HTML tags within it.

Note that the text file is allocated dynamically in memory and has no fixed length limit.

The Reporting Server must be able to locate the launch page using APP PATH or EDAPATH. See *Choosing Search Path Logic* on page 148 for details on search paths.

3. Create a procedure that accepts multiple values to generate a dynamic report.

Example: Adding a Dynamic Multiselect Drop-Down List

This example runs on WebFOCUS for Windows. If you are using another platform, substitute the appropriate platform-specific command for the FILEDEF command. For information on the FILEDEF command and where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create a procedure named DYNAMMUL, which populates a drop-down list from the field COUNTRY in the data source SHORT.

   Procedure: DYNAMMUL.FEX

   ```
   FILEDEF DYNAMLST DISK BASEAPP/DYNAMLST.TXT
   DEFINE FILE SHORT
   OPTCOUNTRY/A40 = '<option>'|COUNTRY;
   END
   TABLE FILE SHORT
   SUM OPTCOUNTRY
   BY COUNTRY NOPRINT
   ON TABLE HOLD FORMAT ALPHA AS DYNAMLST
   END
   -RUN
   -HTMLFORM DYNAMIC2
   -RUN
   ```

2. Create a launch page named DYNAMIC2.HTM. The Reporting Server must be able to locate the page in APP PATH or EDAPATH. See *WebFOCUS Application Logic* on page 27 for details on search paths.

   The sample launch page uses the Servlet.

   Launch Page: DYNAMIC2.HTM

```
<HTML>
<TITLE> DYNAMIC DROP-DOWN LIST REPORT </TITLE>
<H4> PROJECTED RETURN BY COUNTRY </H4>
<FORM ACTION="/ibi_apps/WFServlet" METHOD="GET">
<INPUT TYPE="HIDDEN" NAME="IBIF_ex" VALUE="MULRPT">
<SELECT NAME="COUNTRY" SIZE="3" MULTIPLE>
<!--WEBFOCUS TABLE DYNAMLST-->
</SELECT>
<BR><BR><INPUT TYPE="SUBMIT" VALUE="RUN REPORT!"></FORM>
</BODY>
</HTML>
```

Note: The HTML comment line must be closed with the –-> characters or a single > character and should not have any other HTML tags within it.

3. Create a procedure named MULRPT, which displays projected returns for a selected country or countries. (If no countries are selected, projected returns for all countries in the data source will be displayed.)

   Procedure: MULRPT.FEX

```
-SET &COUNTER=1;
TABLE FILE SHORT
SUM PROJECTED_RETURN
BY COUNTRY
-IF &COUNTRY.EXISTS THEN GOTO LOOP1 ELSE GOTO DONE;
-LOOP1
WHERE COUNTRY EQ '&COUNTRY'
-IF &COUNTRY0.EXISTS NE 1 THEN GOTO OUTLOOP;
-REPEAT OUTLOOP FOR &COUNTER FROM 2 TO &COUNTRY0;
OR '&COUNTRY.&COUNTER'
-OUTLOOP
-DONE
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
```

   Note: At run time, &COUNTRY0 contains the number of countries selected in the multiselect drop-down list. For example, if three countries are selected, &COUNTRY0 is set to 3, &COUNTRY and &COUNTRY1 are set to the first country selected, &COUNTRY2 is set to the second country selected, and &COUNTRY3 is set to the third country selected.

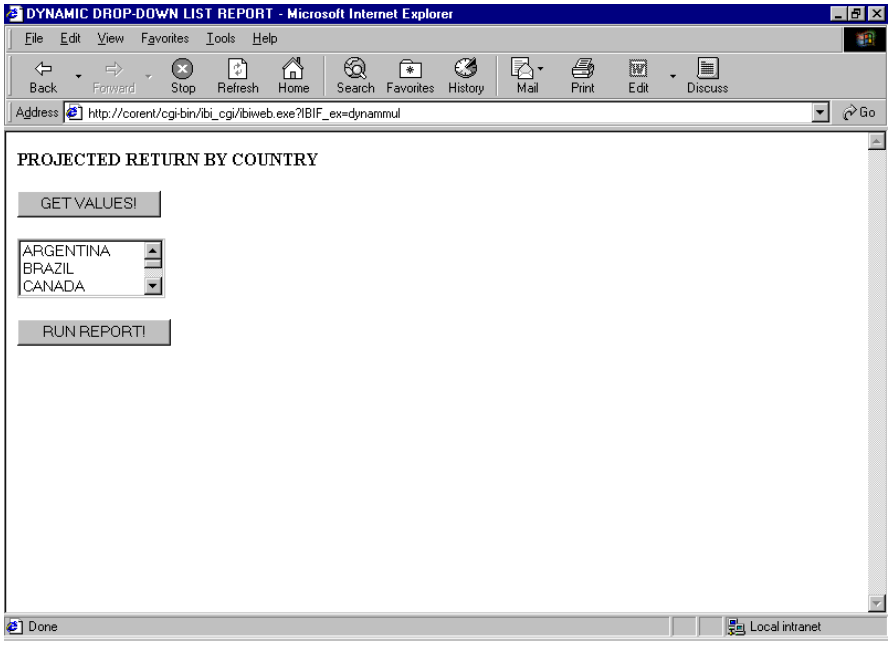
   This example is for WebFOCUS environment with the Amper Auto-prompting feature not enabled (WF Client config variable IBIF_wfdescribe=OFF). For more information on the configuration parameter IBIF_wfdescribe, see the *WebFOCUS Security and Administration* manual.

If environment has the Amper Auto-prompting feature enabled, the following parameter can be added to the form to turn off the Amper Auto-prompting feature for the request being submitted from this form.

```
<INPUT TYPE="HIDDEN" NAME="IBIF_wfdescribe" VALUE="OFF">
add the above line to example file DYNAMIC2.HTM after line
<FORM ACTION="/ibi_apps/WFServlet" METHOD="GET">
```

4. Run the procedure DYNAMMUL. You can also use JavaScript to run a procedure when a launch page opens.

   A page similar to the following displays:



5. Select *CANADA*, *HONG KONG*, and *MEXICO*. Click *RUN REPORT!* to receive the report:

Country	Projected Annualized Return
CANADA	3.990
HONG KONG	4.550
MEXICO	5.040

Enabling Ad Hoc Reporting

A text area enables a user to enter an ad hoc report request. The request can include WebFOCUS report commands (for example, TABLE FILE *filename*) and Dialogue Manager commands and variables. When the user submits the request, it is assigned in its entirety to the variable associated with the text area and is passed to WebFOCUS.

Caution: Before enabling ad hoc reporting, make sure that you consider all the possible consequences. A careless user, for instance, could enter a request which uses so many resources that response time is adversely affected for all users.

A site may disable ad hoc reporting in the file IBIDIR.WFS, which is located by default as follows:

Windows: *install_drive*:\ibi\WEBFOCUS82\client\wfc\etc

UNIX: /ibi/WEBFOCUS82/client/wfc/etc

z/OS: /ibi/WEBFOCUS82/client/wfc/etc

If the following line is commented (the default), you can implement ad hoc reporting:

```
# <SET> IBIF_adhocfex(protect)
```

If the line is uncommented, you cannot implement ad hoc reporting. Uncommented, it prohibits the passing of the variable IBIF_adhocfex to the WebFOCUS Client from a browser.

The variable is generated internally. Managed Reporting uses it for every procedure.

Syntax: How to Enable Ad Hoc Reporting

You can add optional attributes to the syntax shown.

```
<TEXTAREA NAME="IBIF_adhocfex" VALUE="value" ROWS=rows COLS=cols>
</TEXTAREA>
```

where:

value

Is a default request that displays in the text area. To display an empty text area, type:

```
VALUE=""
```

rows

Is the number of rows in the text area.

cols

Is the number of columns in the text area.

Example: Enabling Ad Hoc Reporting

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create a launch page named ADHOC, which contains a text area that prompts for a report request. This page must be accessible to the web server.

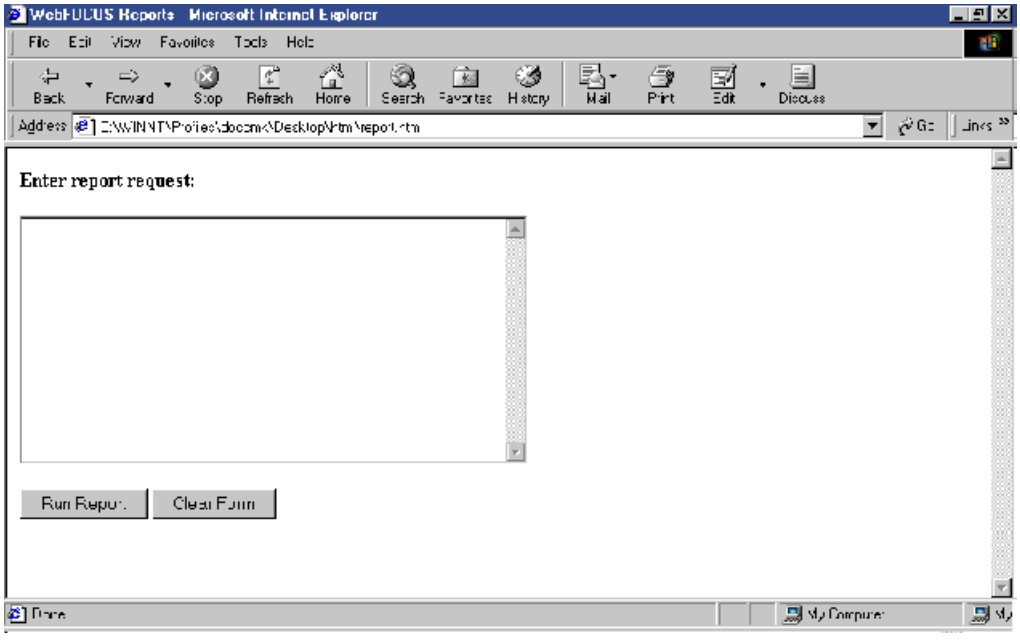
   The sample launch page uses the Servlet.

   Launch Page: ADHOC.HTM

```
<HTML>
<HEAD>
<TITLE> WebFOCUS Report </TITLE>
</HEAD>
<BODY>
<H4>Enter report request:</H4>

<FORM METHOD="get" ACTION="/ibi_apps/WFServlet">
<P ALIGN=LEFT NOWRAP><PRE>
<TEXTAREA NAME="IBIF_adhocfex" VALUE="" ROWS=12 COLS=48 ALIGN=LEFT>
</TEXTAREA>
</PRE></P>
<P>
<INPUT NAME="submit" TYPE=SUBMIT VALUE="Run Report">
<INPUT NAME="reset" TYPE=RESET VALUE="Clear Form">
</P>
</FORM>
```

```
</BODY>
</HTML>
```

2. Run the launch page, which will look similar to this:



3. Enter the following request in the text area:

```
TABLE FILE CENTORD
SUM QUANTITY BY HIGHEST 1 ORDER_DATE
BY PRODNAME
END
```

4. Click *Run Report* to receive the report.

PAGE 1		
Date Of Order:	Product Name:	Quantity:
2000/12/30	110 VHS-C Camcorder 20 X	1
	120 VHS-C Camcorder 40 X	1
	150 8MM Camcorder 20 X	2
	2 Hd VCR LCD Menu	1
	250 8MM Camcorder 40 X	1
	650DL Digital Camcorder 150 X	1
	AR2 35MM Camera 8 X	1
	Combo Player - 4 Hd VCR + DVD	2
	DVD Upgrade Unit for Cent. VCR	2
	QX Portable CD Player	1
	R5 Micro Digital Tape Recorder	2
	ZC Digital PDA - Standard	1
	ZT Digital PDA - Commercial	2

Validating a Form With JavaScript

JavaScript is an object-oriented, interpretive language used with HTML to provide dynamic capabilities to otherwise static webpages.

A JavaScript function included in an HTML page is processed and executed by the web browser. Integrating a JavaScript function into an HTML page that calls WebFOCUS, or a page that displays a report, enables you to:

- Validate entries on a form before submitting them to WebFOCUS. Validating entries prevents a user from requesting restricted or unavailable data. This topic describes how to code a JavaScript function that checks for a user entry on a form.
- Perform calculations on data returned in a WebFOCUS report. JavaScript offloads data processing demands from the Reporting Server to the web browser. See *Enhancing a User Interface* on page 259 for an example of a drill-down report illustrating this feature.

You have many additional design capabilities with JavaScript. See your JavaScript documentation for details on JavaScript techniques and syntax.

Example: Checking For a User-Supplied Value on a Form

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create a launch page named VALIDATE, which calls a JavaScript function named checkInput. This procedure must be accessible to the web server.

   The sample launch page uses the Servlet.

   Launch Page: VALIDATE.HTM

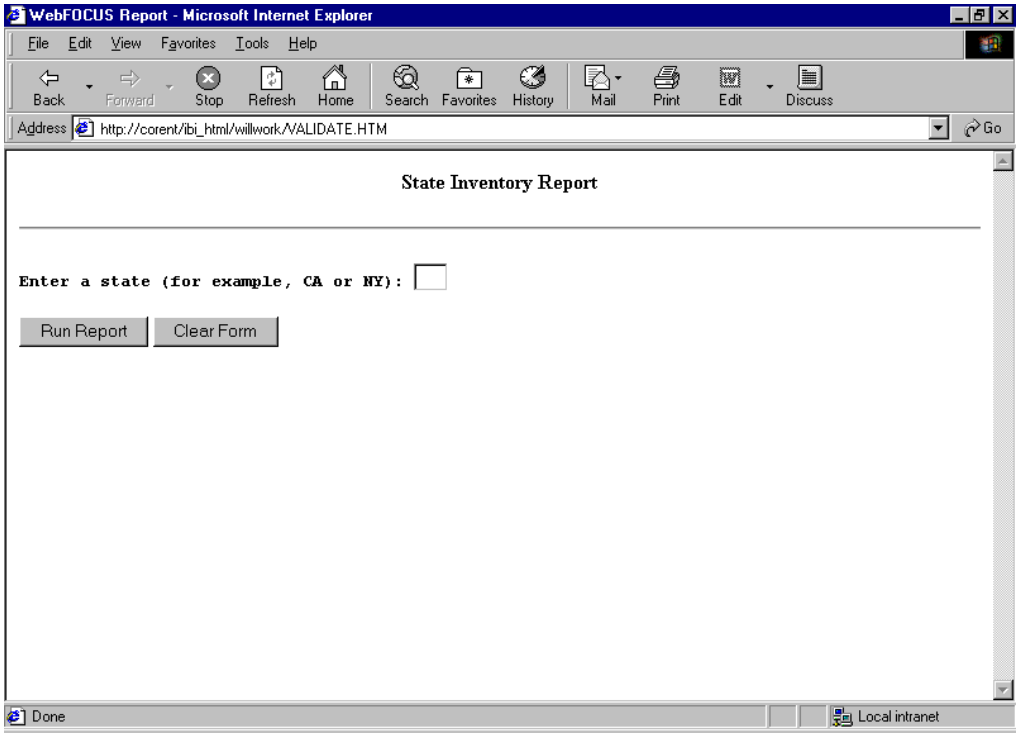
```
<HTML>
<HEAD>
<TITLE> WebFOCUS Report </TITLE>


<SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">
function checkInput() {
  if(document.form2.STATE.value == "") {
    alert("You must enter a value for the state!");
    return false;
  }
    else {
      return true;
    }
  document.form2.submit();
}
</SCRIPT>


</HEAD>
<BODY>
<H4 ALIGN=CENTER>State Inventory Report</H4>
<HR>
<FORM ACTION="/ibi_apps/WFServlet" METHOD="get"
TARGET="_blank" NAME="form2" onSubmit="return checkInput()">
<INPUT NAME="IBIF_ex" VALUE="stsales" TYPE="hidden">
<P ALIGN=LEFT NOWRAP><PRE>
<B>Enter a state (for example, CA or NY): </B><INPUT NAME="STATE"
TYPE="text" SIZE="2">
</PRE></P>
<P><PRE></PRE></P>
<P>
<INPUT NAME="submit" TYPE=SUBMIT VALUE="Run Report">
<INPUT NAME="reset" TYPE=RESET VALUE="Clear Form">
</P>
</FORM>
</BODY>
</HTML>
```

2. Create a procedure named STSALES, which generates an inventory report for eMart, in the state the user requests.

   Procedure: STSALES.FEX

```
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STATE BY STORENAME BY PRODNAME
WHERE STATE EQ '&STATE'
WHERE STORENAME EQ 'eMart'
END
```

3. Run the launch page.



4. Without entering a value for the state, click *Run Report*. The following message displays:

WebFOCUS Autoprompt Facility

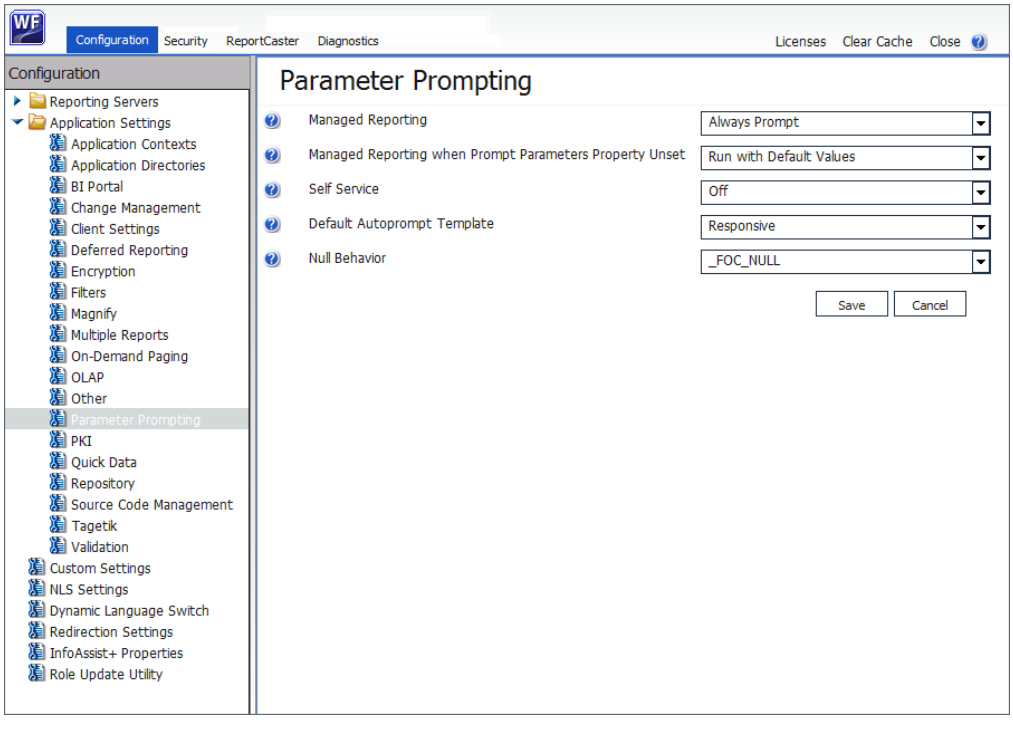
The WebFOCUS Autoprompt facility dynamically creates a form that prompts users for the parameters (variables) necessary to execute a procedure (FEX). The parameter values entered or selected by users on the Autoprompt form can be used as field values, or as the objects of the display and sort commands in the report request.

There are two Autoprompt user interface templates:

- Responsive Autoprompt. For more information, see *Responsive Autoprompt* on page 171.
- HTML Autoprompt. For more information, see *HTML Autoprompt* on page 184.

Autoprompt Configuration

The Autoprompt facility is configured using the WebFOCUS Administration Console, from the Configuration Application Settings Parameter Prompting dialog box. The Responsive Autoprompt template is the default in Release 8.2.01. The Release 8.1 HTML Autoprompt template is available and can be configured by selecting *HTML_Top* or *HTML_Top_Checked* in the WebFOCUS Administration Console Parameter Prompting Configuration settings, as shown in the following image.



The Parameter Prompting Settings determine parameter prompting behavior in the WebFOCUS Client.

Managed Reporting (IBIMR_PROMPTING)

Enables or disables parameter prompting for all Managed Reporting requests. Possible values are:

- **Off.** Turns off parameter prompting at the site level.
- **Run with Default Values (XMLRUN).** Prompts for variables created with the -DEFAULT command and any other variable that does not have a value.

- **Always Prompt (XMLPROMPT).** Prompts for variables created with the -DEFAULT command when there is another variable that does not have a value assigned. This is the default value.

Managed Reporting when Prompt Parameters Property Unset (IBIMR_PROMPTINGUNSET)

Enables or disables parameter prompting for Managed Reporting procedures (FEXes) when Managed Reporting (IBIMR_PROMPTING) is set to Always Prompt (XMLPROMPT) or Run with Default Values (XMLRUN), and the Prompt for Parameters check box is not selected in the FEX Properties dialog box. Possible values are:

- **Off.** Turns off parameter prompting.
- **Run with Default Values (XMLRUN).** Prompts for variables created with the -DEFAULT command and any other variable that does not have a value. This is the default value.
- **Always Prompt (XMLPROMPT)**. Prompts for variables created with the -DEFAULT command when there is another variable that does not have a value assigned.

Self Service (IBI_WFDESCRIBE_DEFAULT)

Enables or disables Autoprompt for self-service reporting. Possible values are:

- **Off.** Turns off Autoprompt. This is the default value.
- **Run with Default Values (XMLRUN).** Prompts for variables created with the -DEFAULT command and for any other variable that does not have a value.
- **Always Prompt (XMLPROMPT).** Only prompts for variables created with the -DEFAULT command when there is another variable that does not have a value assigned and, therefore, will be prompted for.
- **Display XML (Debug with syntax error checking) (XML).** Displays the XML document describing the variables in the browser. This setting is used internally, and is recommended for debugging and syntax error checking purposes only.
- **Display XML (Debug) (XMLCHECK).** Displays the XML document describing the variables in the browser. This setting is used internally, and is recommended for debugging purposes only.

Note: Managed Reporting uses a separate variable setting, which is IBIMR_PROMPTING.

Default Autoprompt Template (IBI_DESCRIBE_TEMPLATES)

Specifies the template that defines the layout of the Autoprompt facility. Possible values are:

- **Responsive.** Specifies the use of the responsive jQuery-based implementation and the autoprompt.jsp template. This is the default value.

- **HTML_Top.** Specifies the use of the HTML-based implementation and the autoprompt_top.html template, which displays parameters horizontally at the top of the page.
- **HTML_Top_Checked**. Specifies the use of the HTML-based implementation and the autoprompt_top_checked.html template. In this template, the Run in a new window check box is pre-selected, specifying that all reports open in a new window, by default.

Null Behavior (IBIF_DESCRIBE_NULL)

Specifies the value (_FOC_NULL or FOC_NONE) that the client assigns (in a -SET command) to the variable when the dynamic multiselect list All Values option is selected. The default value is _FOC_NULL. For more information on _FOC_NULL, see *Internal Processing of _FOC_NULL* on page 205. For more information on FOC_NONE, see *Internal Processing of FOC_NONE* on page 205.

Responsive Autoprompt

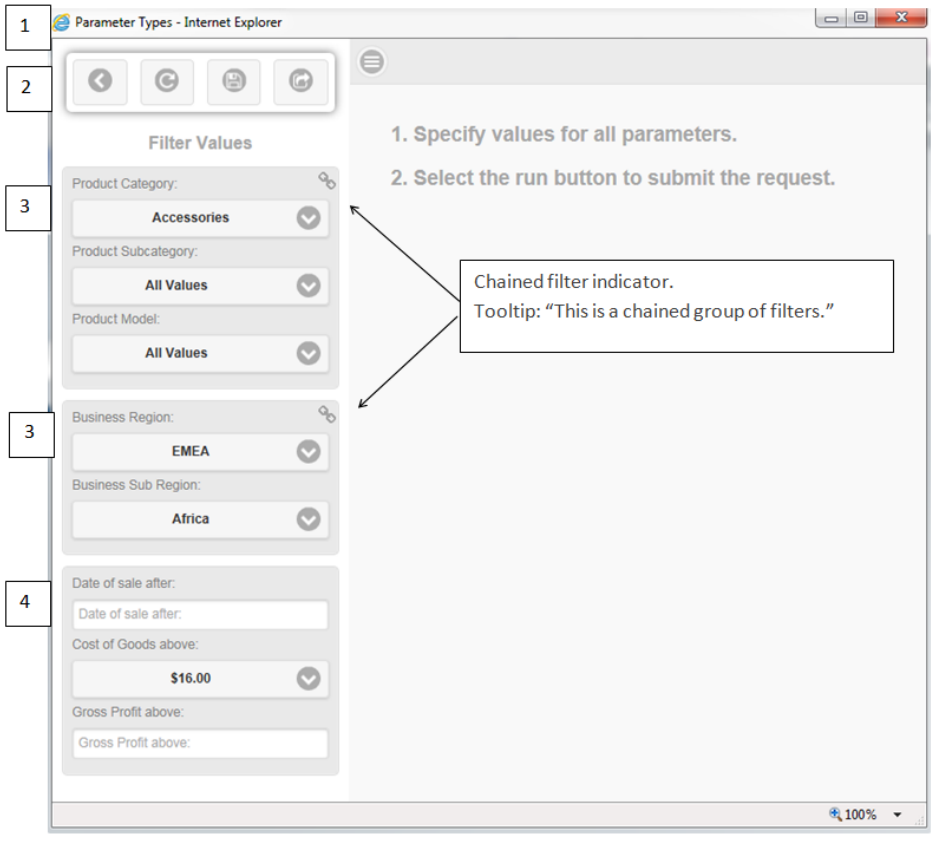
The Responsive Autoprompt facility provides a modern user interface design with responsive mobile support, chaining of dynamic lists for fields in a dimension hierarchy, and a calendar control for simple filters to select a date for a date field of YYMD format with modifiers. The date format must have all components (year, month, day) and not be a Date-Time field format.

Note:

- Responsive Autoprompt is the default for the Autoprompt facility. For information on how to configure the Autoprompt facility, using the Administration Console, see *Autoprompt Configuration* on page 169.
- The Responsive Autoprompt facility does not provide 508 accessibility support.

Responsive Autoprompt Page Components

The Responsive Autoprompt page displays filters vertically on the left and report output on the right, as shown in the following image.



1. The window title is the title that displays in the tree when the report or chart is run from the tree or a request that obtains the procedure code from the repository. When the request is run from within InfoAssist a number is appended to the end of the report output tab within InfoAssist. When the request has not been previously saved and is run from a tool or the text editor the title is AdHocFex.
2. Options bar enables the user to:

   ❑ Close (hide) the filter panel.

   ❑ Reset the filter values to the values shown on initial display of the page.

- Save the filter values. This option is available when the request is running the procedure directly from the repository and the user is authorized to create Save Parameter reports. This option does not display when the request is run from InfoAssist, the Text Editor, or App Studio Report Canvas because the request code is sent by the tool and may not have been saved to the repository.

  Note:

  - The maximum length of a title value for a file is 256 characters.
  - If the title of your file has the same name of an existing file in the folder, you will receive a message asking if you wish to replace the file.

- Run with filter values runs the request with the values selected or entered. The filter pane closes so that the report output is fully visible which is important when using small devices.

3. The Filter Values section displays the parameter controls in a chained group for each set of parameters for fields in a single path dimension hierarchy. Parameters for fields that are not in a dimension hierarchy are in an unchained group. Chained groups display before the unchained group.

   Chaining will populate controls based on the selected value from the prior control in the chain. Chaining requires using a dynamic filter for all fields in the chain.

   A chained grouping is identified by an image displayed in the upper-right corner of the group, as shown in the Responsive Autoprompt page example.

4. Unchained groups do not display an image in the upper-right corner.

For the request code used for the Responsive Autoprompt page, see *Responsive Sample Code (Parameter_Type.fex)* on page 182.

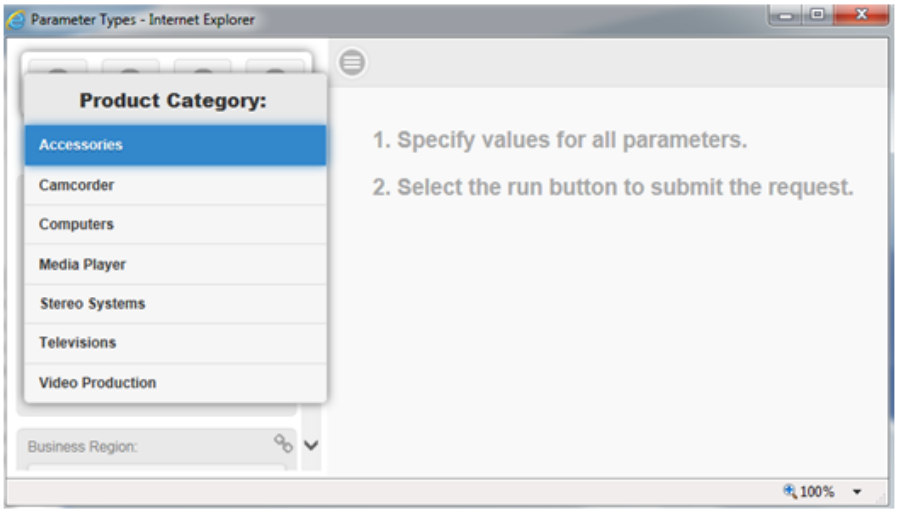
The Filter Panel can be closed (hidden) or displayed by selecting the Show filter panel image located in the upper-left corner of the report output panel.

Selection Lists

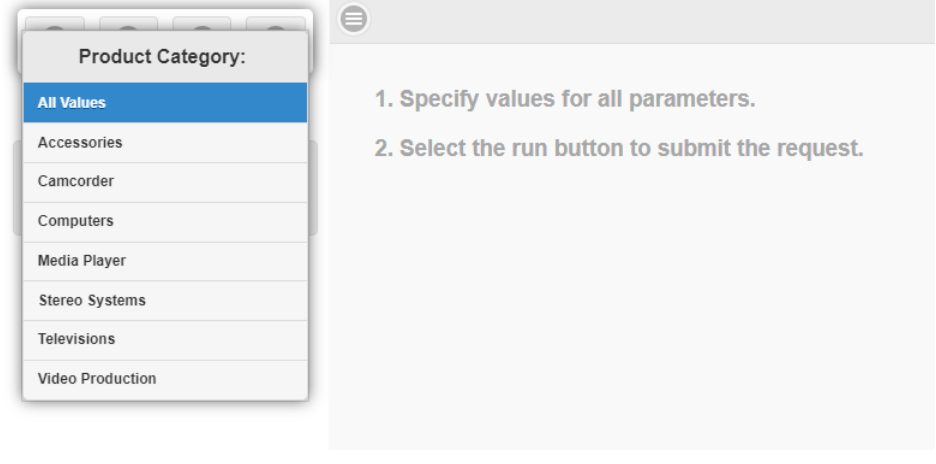
A selection list will display when the parameter field is selected. The selection list display varies depending on the number of values in the selection list and the device type. A selection list is dismissed (closed) by selecting the option in the upper-left corner or selecting any area outside of the selection list.

When there are approximately seven (7) or fewer values, depending on the device size, the selection list overlays the Filter Panel display.

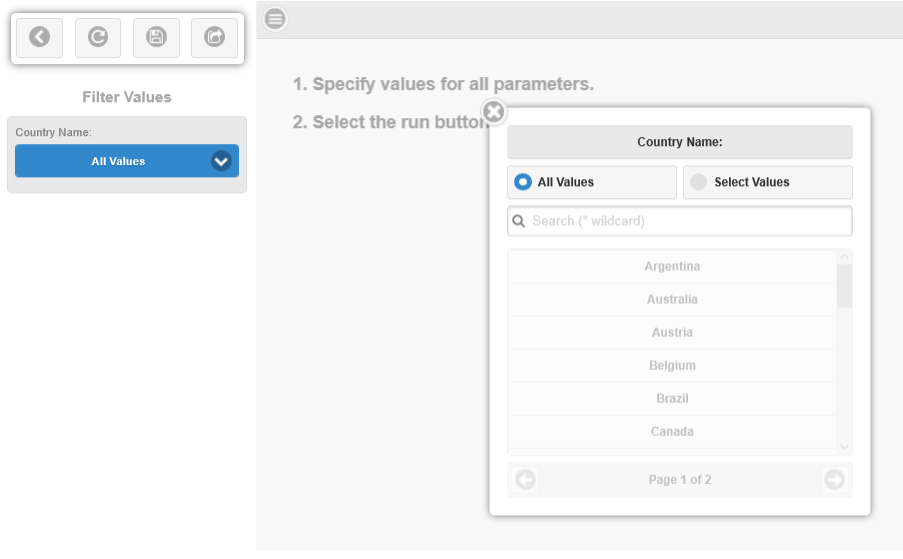
A single select list will default to the first value in the list when no default value has been specified for the parameter.



When a filter is identified as optional by assigning _FOC_NULL to be the default value, the first value displayed in the list is *All Values*, and it is selected, by default, as shown in the following image.



Larger selection lists overlay the report output panel, as shown in the following image.

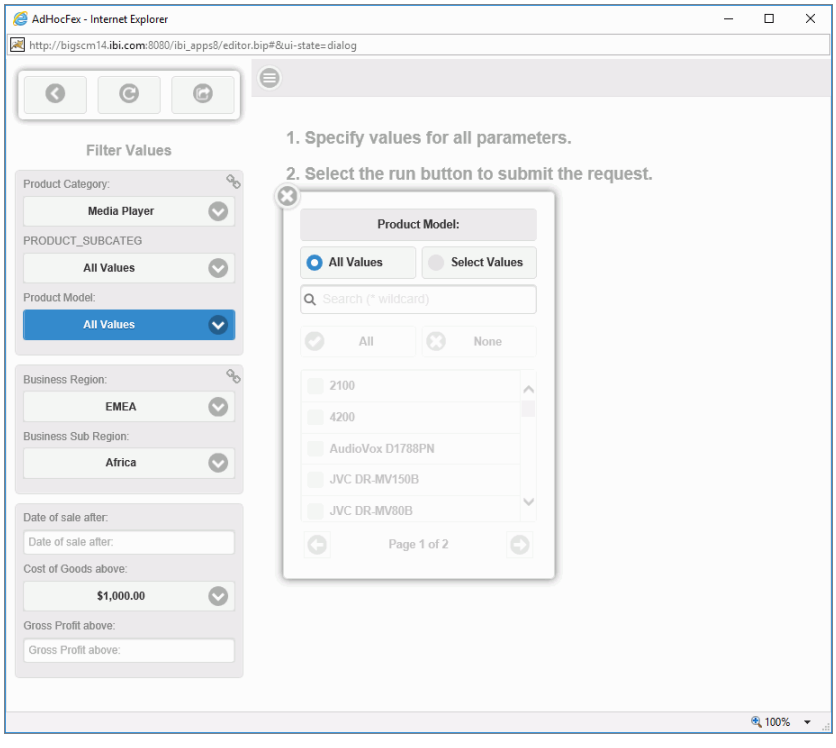


When _FOC_NULL is defined as the default value for a selection list, the *All Values* option appears as the default selection, regardless of the size of the list.

The appearance of other options in the values selection dialog box depends upon the filter type in which the parameter is referenced.

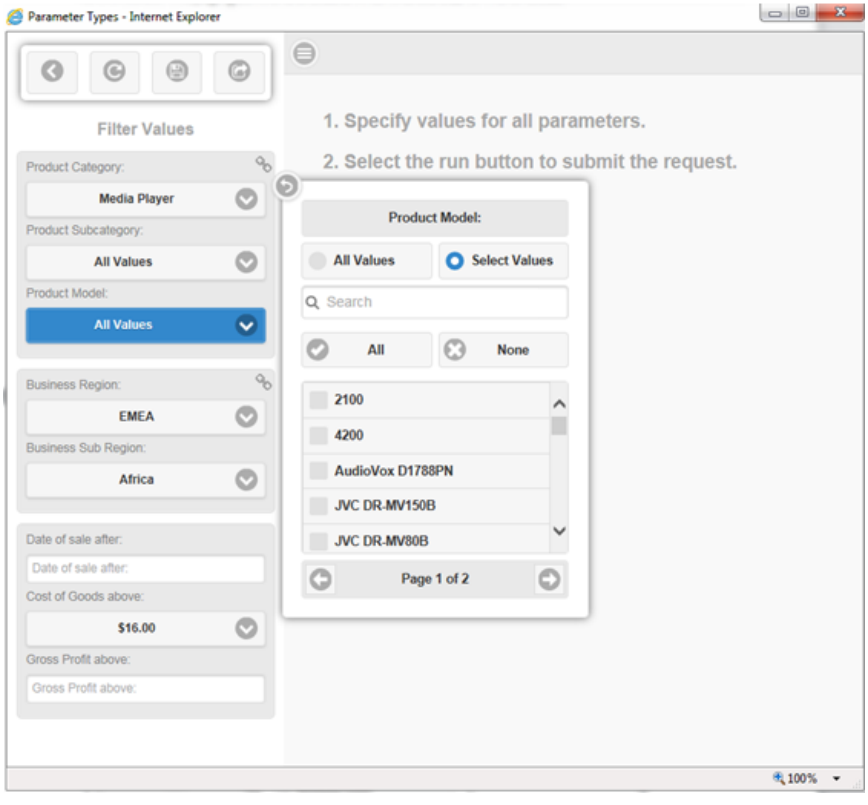
- The *Select Values* option is available with the *All Values* option when the filter is a static or dynamic multiselect list. It is also available in a static single select list whose list of items is longer than can be displayed on a single screen.
- The *All* and *None* options are also available when the filter is a dynamic or static multiselect list.

The following image shows the values selection dialog box options when the parameter filter type is a dynamic multiselect list and the parameter is not assigned a default value. When a default value is not assigned to a parameter in a dynamic multiselect filter, the default value is *All Values*. When *All Values* is selected, the filter is not applied, resulting in all values of the field being included in the report or chart.

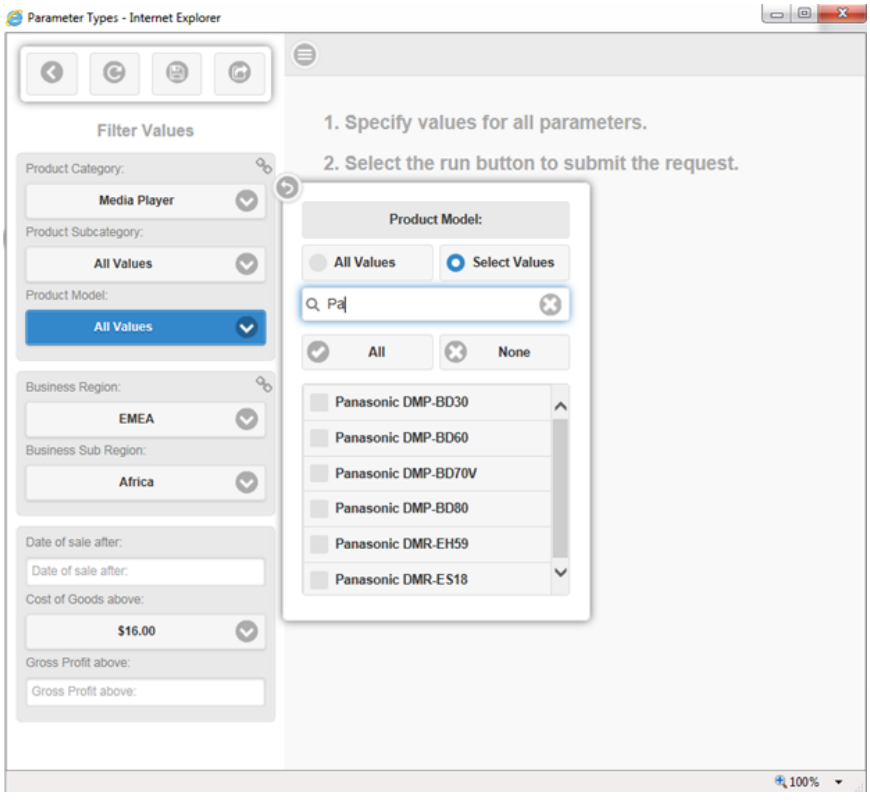


Dynamic list values that display in Responsive Autoprompt are organized in a case-insensitive sort order. Dynamic list values that display in HTML Autoprompt are organized in the sort order returned by the Reporting Server, which is determined by the operating system of the host machine.

Selecting the *Select Values* option enables the Search, All, None, individual values, and paging controls. The paging control is available when there are 25 or more values.



The *Search* option filters the values in the list as characters are entered. The search is applied to the value that displays in the selection list.



Selecting *All* checks the check box for all values in the selection list. When a search has been applied, only the values listed for the search result are checked. After selecting *All*, individual values can be unchecked.

Selecting *None* clears all values that are checked.

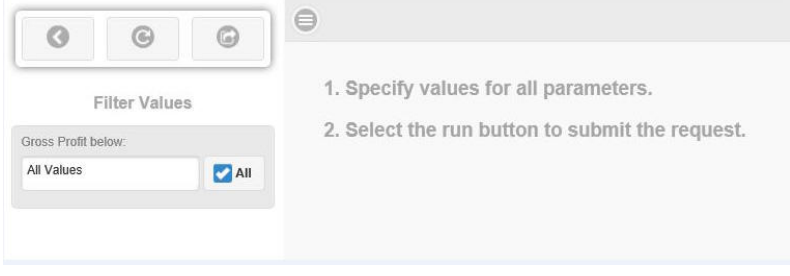
Note: When a single select filter is assigned a value and the filter is selected to display the Values Selection dialog box, the list of values will display with the assigned value visible and not selected (highlighted). You can view the values assigned to a parameter by placing the cursor over the parameter field in the Filter panel located on the left side of the Autoprompt page.

Simple Filter

You can add a simple filter to prompt the user for the value of a variable. When a variable is assigned the default value _FOC_NULL and used in a simple filter, the Autoprompt simple filter control will default to *All Values* and have the *All* option checked, by default. When a value is entered, the *All* check box is unchecked. If you want to specify to include all values of the field the variable is filtering on, select the *All* option. Entering *All Values* in the parameter field will filter on the value All Values, not return all values for the field.

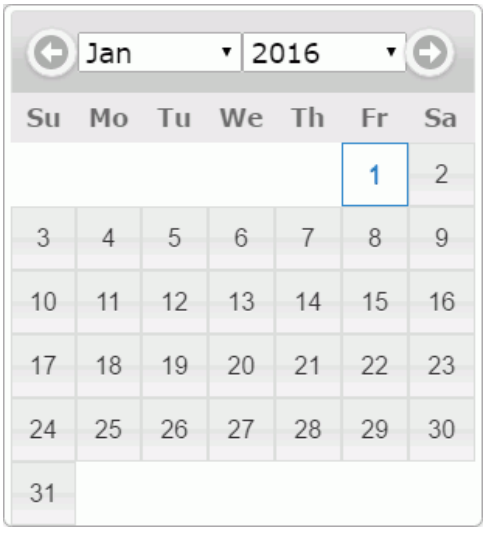
For example:

```
-DEFAULT &GROSS_PROFIT_US = _FOC_NULL;
TABLE FILE wfretail82/wf_retail
HEADING CENTER
"Product Models with Gross Profit is below &GROSS_PROFIT_US"
SUM WF_RETAIL.WF_RETAIL_SALES.GROSS_PROFIT_US
BY WF_RETAIL.WF_RETAIL_PRODUCT.PRODUCT_CATEGORY
BY WF_RETAIL.WF_RETAIL_PRODUCT.PRODUCT_SUBCATEG
BY WF_RETAIL.WF_RETAIL_PRODUCT.MODEL
WHERE WF_RETAIL.WF_RETAIL_SALES.GROSS_PROFIT_US LT &GROSS_PROFIT_US.(|
FORMAT=D20.2M).Gross Profit below:.QUOTEDSTRING;
ON TABLE SET STYLE *
INCLUDE=IBFS:/FILE/IBI_HTML_DIR/javaassist/intl/EN/combine_templates/
ENWarm.sty,$
ENDSTYLE
END
```



Calendar Control

The calendar control will display for parameters that are simple filters for a field that is a combination of YYMD date format with any of the supported modifiers. The date format must have all components (year, month, day). Date-Time field formats are not supported.

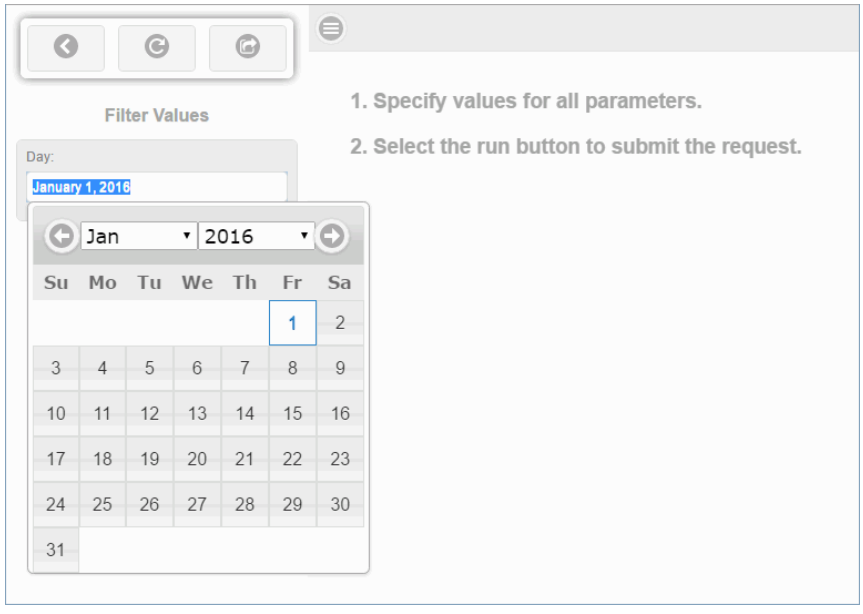


Note:

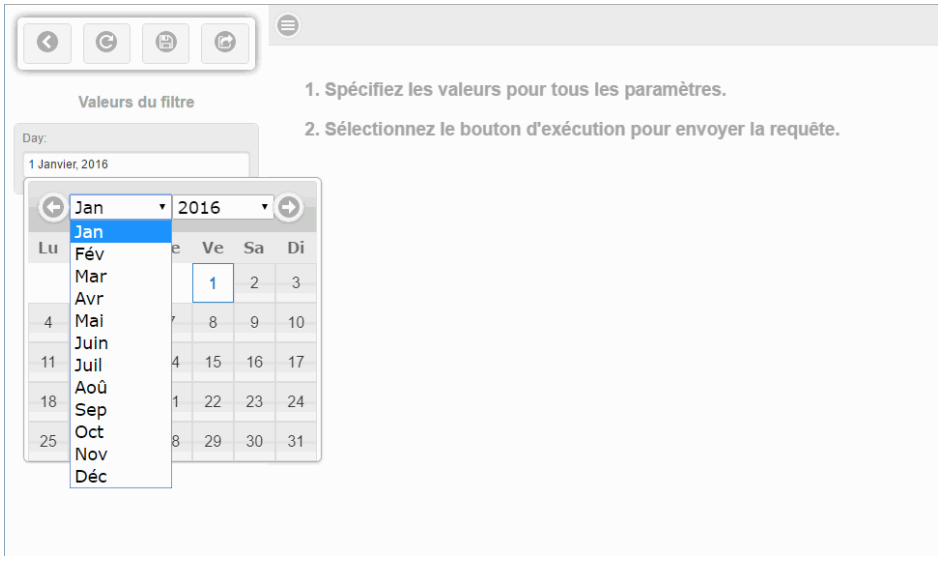
- The default value is the current date.
- The default value can be specified with an English month, two-digit day, and four-digit year. For example:

```
-DEFAULT &STARTDATE='January 01 2016';
```

The following image shows the calendar control with the default date of January 01, 2016.



The default date will display in the calendar control in the language that the user used at sign in. The following image shows the filter pane with the default value in French and the calendar control with the month expanded to show the month values are French.



Responsive Sample Code (Parameter_Type.fex)

You can use the following request to generate the Responsive Autoprompt page shown in *Responsive Autoprompt Page Components* on page 172.

1st Chained Group

Product Category: Media Player *(Single select small value list)*

Product Subcategory: All Values *(Multiselect small value list)*

Product Models: All Values *(Multiselect list with options and paging control - more than 25 values)*

2nd Chained Group *(Easy to show chaining of value)*

Business Region: North America (Single select small value list)

Business Subregion: Northeast (Single select small value list)

Non-chained grouping (*select run without entering or selecting value to see validation message)*

Date of Sale: January/1/2016

Cost of Goods above: $16.00 (minimum value)

Gross Profit above: 1000 (type in an alpha value and select Run to see validation message)

```
SET EMPTYREPORT=ON
TABLE FILE retail_samples/wf_retail
HEADING CENTER
"Product Models sold after &TIME_DATE in &BUSINESS_REGION
&BUSINESS_SUB_REGION with:"
"Cost of Goods is above &COGS_US"
"Gross Profit is above &GROSS_PROFIT_US"

SUM WF_RETAIL.WF_RETAIL_SALES.COGS_US
WF_RETAIL.WF_RETAIL_SALES.GROSS_PROFIT_US
BY WF_RETAIL.WF_RETAIL_PRODUCT.PRODUCT_CATEGORY
BY WF_RETAIL.WF_RETAIL_PRODUCT.PRODUCT_SUBCATEG
BY WF_RETAIL.WF_RETAIL_PRODUCT.MODEL

-** 1ST CHAINED GROUPING **
-*Single Select Dynamic List (7 values)
WHERE WF_RETAIL.WF_RETAIL_PRODUCT.PRODUCT_CATEGORY EQ
'&PRODUCT_CATEGORY.(FIND WF_RETAIL.WF_RETAIL_PRODUCT.PRODUCT_CATEGORY
IN WF_RETAIL |FORMAT=A40V).Product Category:.';
```

```
-*Multi-select Select Dynamic List (fewer than 10 values for
-*each PRODUCT_CATEGORY value) WHERE
WF_RETAIL.WF_RETAIL_PRODUCT.PRODUCT_SUBCATEG EQ
'&PRODUCT_SUBCATEG.(OR(FIND WF_RETAIL.WF_RETAIL_PRODUCT.PRODUCT_SUBCATEG
IN WF_RETAIL |FORMAT=A50V,WITHIN=PRODUCT_CATEGORY)).Product
Subcategory:.';


-* Chained Multi-select Dynamic List (All Values and # of values
-* dependent on chained selection.  Selecting;
-* Product Category (Media Player) and Product Subcategory (Blu Ray)
-* more than 25 values.
-* Product Category (Media Player) and Product Subcategory
-* DVD Players) more than 10 values and fewer than 25 values.
WHERE WF_RETAIL.WF_RETAIL_PRODUCT.MODEL EQ &MODEL.(OR(FIND
WF_RETAIL.WF_RETAIL_PRODUCT.MODEL IN WF_RETAIL
|FORMAT=A50V,WITHIN=PRODUCT_SUBCATEG)).Product Model:.;


-** 2ND CHAINED GROUPING **
-* Single Multiselect Static List (4 values)
WHERE WF_RETAIL.WF_RETAIL_GEOGRAPHY_STORE.BUSINESS_REGION EQ
'&BUSINESS_REGION.(FIND
WF_RETAIL.WF_RETAIL_GEOGRAPHY_STORE.BUSINESS_REGION IN WF_RETAIL
  |FORMAT=A15V).Business Region:.';


-* Chained Single Dynamic List (10 or fewer values for each
-* Business Region selection)
WHERE WF_RETAIL.WF_RETAIL_GEOGRAPHY_STORE.BUSINESS_SUB_REGION EQ
'&BUSINESS_SUB_REGION.(FIND
WF_RETAIL.WF_RETAIL_GEOGRAPHY_STORE.BUSINESS_SUB_REGION IN WF_RETAIL
|FORMAT=A25V,WITHIN=BUSINESS_REGION).Business Sub Region:.';


-** NOT CHAINED GROUPING **
-*Dynamic Single Select List (Over 25 values)
WHERE WF_RETAIL.WF_RETAIL_SALES.COGS_US GT '&COGS_US.(FIND
WF_RETAIL.WF_RETAIL_SALES.COGS_US IN WF_RETAIL |FORMAT=D20.2M).Cost of
Goods above:.';


-*Simple Prompt (Enter numeric value)
WHERE WF_RETAIL.WF_RETAIL_SALES.GROSS_PROFIT_US GT
&GROSS_PROFIT_US.(|FORMAT=D20.2M).Gross Profit above:.QUOTEDSTRING;


-*Simple Prompt - Calendar (Select date value)
WHERE WF_RETAIL.WF_RETAIL_TIME_SALES.TIME_DATE GT
&TIME_DATE.(|FORMAT=YYMD).Date of sale after:.QUOTEDSTRING;
ON TABLE SET STYLE *
INCLUDE=IBFS:/FILE/IBI_HTML_DIR/javaassist/intl/EN/combine_templates/
ENWarm.sty,$
ENDSTYLE
END
```

HTML Autoprompt

The HTML Autoprompt facility enables you to create a form that prompts users for the variables necessary to execute a procedure (FEX). The variable values entered or selected by users on the Autoprompt form can be used as field values, or as the objects of the display and sort commands in the report request.

Note:

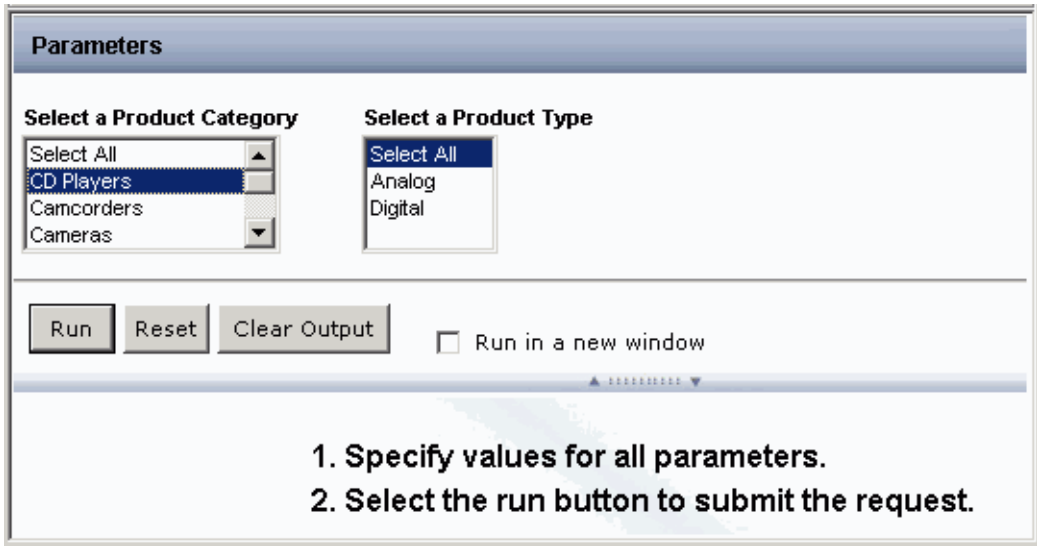
- HTML Autoprompt is not the default for the Autoprompt facility. Responsive Autoprompt is the default. For information on how to configure the Autoprompt facility, using the Administration Console, see *Autoprompt Configuration* on page 169.
- The HTML Autoprompt facility provides 508 accessibility support.
- HTML Autoprompt does not include chaining support. For more information on chaining, see *Chaining Values* on page 200.
- HTML Autoprompt does not include a calendar control for date fields used in a simple filter. For more information on calendar controls, see *Calendar Control* on page 180.
- Dynamic list values that display in HTML Autoprompt are organized in the sort order returned by the Reporting Server, which is determined by the operating system of the machine. Dynamic list values that display in Responsive Autoprompt are organized in a case-insensitive sort order.

HTML Autoprompt Page Components

When creating a report request (FEX) with variables to be prompted for with an Autoprompt form, the descriptions to display in the title, heading, and for each variable being prompted for can be specified. You can also specify whether users enter or select values from a static or dynamic list and whether the supplied parameter values are validated against a format specification or range of values.

The Autoprompt *Run in new window* option enables users to run a report while still being able to view and change their parameter selections or to display the report in a new window. Users can hide the parameters area to increase the size available to display the report.

The following image shows an Autoprompt form that prompts you to select parameter values that display horizontally across the top of the page.



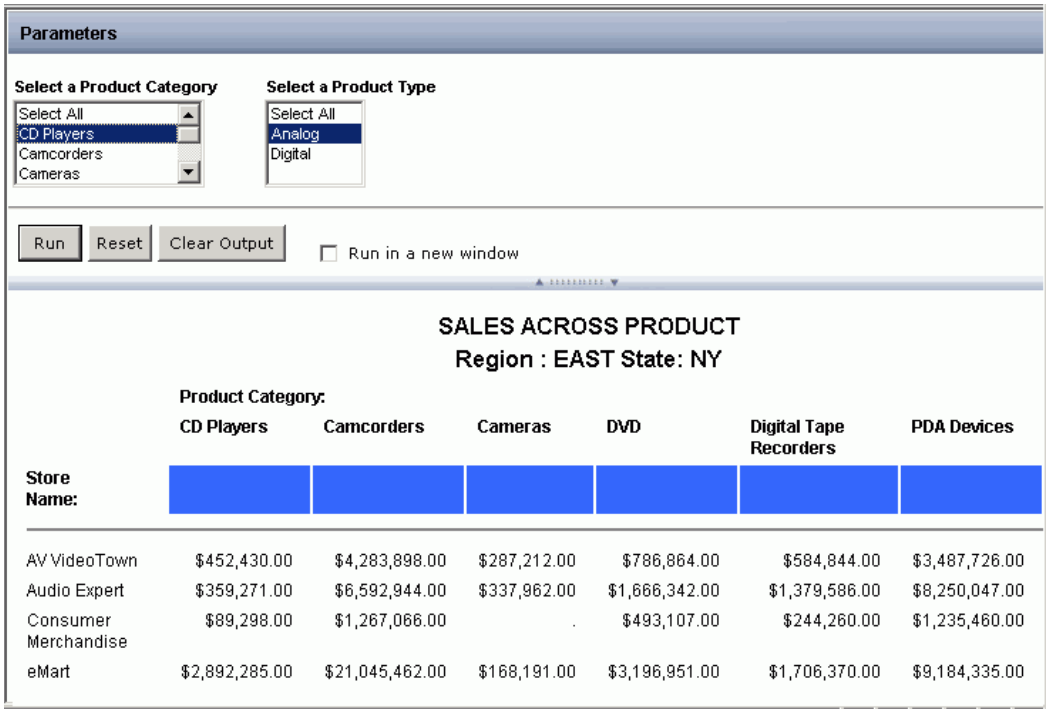
Note: The list controls in the Autoprompt form display in the order the variables are coded in the procedure (FOCEXEC).

Under the Parameters in the Autoprompt form, you have the following options:

Option	Description
Run	Click this button to run the report.
Reset	Click this button to reset the parameter selections.

Option	Description
Save	The Save button displays when the report (FEX) stored in the repository is run and you are authorized to create Save Parameter Reports. This will not be available when the request is run from InfoAssist+, the Text Editor, and the HTML Report canvas. **Note:** ❑ The maximum length of a title value for a file is 256 characters. ❑ If the title of your file has the same name of an existing file in the folder, you will receive a message asking if you wish to replace the file.
Clear Output	Click this button to clear the report output area.
Run in a new window	Select this check box to open the report in a new browser window.
 Hide/Show Parameters	Double-click the splitter bar to hide parameters for full-screen report view. Double-click the splitter bar again to return to the original parameters and report view.

After you run the report, the output appears as shown in the following image:



Autoprompt Considerations

The following are important HTML Autoprompt issues to consider:

- When a Managed Reporting procedure issues a –INCLUDE statement of another Managed Reporting procedure, the procedure referenced by the –INCLUDE statement inherits the Prompt for Parameters property setting in the main procedure.
- If a description value has an open parenthesis character, it must have a space before it. The ampersand (&) character is supported within the description for a variable by specifying the pipe character (|) immediately after the & character in the description. For example, see the Select phrase in the following request:

```
TABLE FILE MOVIES
HEADING
PRINT
MOVIES.MOVINFO.TITLE
MOVIES.MOVINFO.COPIES
BY MOVIES.MOVINFO.RATING
BY MOVIES.MOVINFO.CATEGORY
WHERE MOVIES.MOVINFO.RATING EQ &RATING.(OR(<General Audience,G>,
<Not Rated,NR>,<Parental Guidance,PG>,<PG Over 13,PG13>,<R Over
8,R>)).Select Favorite &| Desired Rating.;
ON TABLE SET PAGE-NUM OFF
ON TABLE NOTOTAL
ON TABLE PCHOLD FORMAT HTML
ON TABLE SET HTMLCSS ON
ON TABLE SET STYLE *
UNITS=IN,
SQUEEZE=ON,
ORIENTATION=PORTRAIT,$
TYPE=REPORT,
GRID=OFF,
FONT='ARIAL',
SIZE=9,$
TYPE=TITLE,
STYLE=BOLD,$
TYPE=HEADING,
SIZE=12,
STYLE=BOLD,$
ENDSTYLE
END
```

- All Dialogue Manager variables referenced within the -HTMLFORM BEGIN and -HTMLFORM END commands are evaluated, even within comment tags. If you do not want any variables to be evaluated, you can specify the NOEVAL option (-HTMLFORM BEGIN NOEVAL). If you want to mix-and-match WebFOCUS variables (which you want to prompt for, such as &COUNTRY) and HTML *special* variables, do not use the NOEVAL option. Either assign each variable a default value or escape the ampersand (&) symbol using a pipe character (|) so that the ampersand (&) symbol is not interpreted as the start of a variable name. For example, in the following sample, the ampersand (&) is escaped using the pipe (|) character so that the variable definition in the comment is not evaluated:

  ```
  -HTMLFORM BEGIN
  <HTML>
  <BODY>
  HELLO &|nbsp &COUNTRY
  </BODY>
  </HTML>
  -HTMLFORM END
  ```

- For dynamic lists, the WebFOCUS Client constructs the request to obtain the values using the specified data source. All environmental commands, such as SET commands, needed to obtain the values from the specified data source must be issued in the WebFOCUS Server profile, user profile, or WebFOCUS Client profile.
- Dynamic list values that display in Responsive Autoprompt are organized in a case-insensitive sort order. Dynamic list values that display in HTML Autoprompt are organized in the sort order returned by the Reporting Server, which is determined by the operating system of the machine.
- Dynamic list values are obtained by the Autoprompt facility prior to running the procedure. Therefore, you cannot use a TABLE request to create a HOLD file to be used by a subsequent TABLE request in the same procedure to populate a dynamic list.
- Autoprompt does not support the following:
  - Parameters in an INCLUDE file when it is coded as -INCLUDE &FILENAME.
  - A parameter coded in a procedure that is referenced by an EX or EXEC command is not processed by parameter prompting. If you must reference a parameter in a procedure, use a -INCLUDE statement.
  - A parameter prompt string containing the ampersand (&) character.
  - Variable names that exceed 12 characters when the variable is included between the -HTMLFORM BEGIN and -HTMLFORM END statements. (The variable is not resolved.)
  - Alphanumeric MISSING data value. Numeric MISSING data value is supported.
- Autoprompt ignores:
  - Variables created with the -SET command. A value has been explicitly specified.
  - Variables created with the –DEFAULTH command. For more information about the –DEFAULTH command, see *Setting a Hidden Variable* on page 195.
  - Global amper variables (&&NAME). This is supported with global variables defined in a Master File. For more information, see *Describing Data With WebFOCUS Language*. If you want to be prompted for the global variable value each time the report is run, make sure the Persistent Amper Variable setting is not selected on the Configuration tab, under the Application Settings, Client Settings page in the Administration Console. For more information, see *Security and Administration*.

- Browsers running in Standards Mode displaying output in an iframe will not display XML. When running a WebFOCUS request with PCHOLD FORMAT XML, select the *Run in new window* option to display the result in a new browser window.

Defining Parameter-Based Filters

The following topics describe how to define parameter-based filters to be displayed in an Autoprompt form.

Adding a Simple Filter to an Autoprompt Form

You can add a simple filter to prompt the user for the value of a variable.

Syntax: How to Add a Simple Filter

```
'&variable'
```

where:

&variable

Is the variable name, including the ampersand (&), for which the value is being prompted. If the variable value will be compared to an alphanumeric field, you must enclose the variable in single quotation marks ('). Character strings must be enclosed in single quotation marks to be handled by most database engines.

If the variable value contains a single quotation mark ('), use the QUOTEDSTRING suffix on the variable, instead of enclosing the value in single quotation marks.

Example: Adding a Simple Filter

The following request prompts the user for a Quantity in Stock value. The variable &QTYSTOCK is not enclosed within single quotation marks (') because the QTY_IN_STOCK field is a numeric format.

```
TABLE FILE ibinccen/centord
SUM CENTORD.INVSEG.QTY_IN_STOCK
BY CENTORD.INVSEG.PRODNAME
WHERE CENTORD.INVSEG.QTY_IN_STOCK LT &QTYSTOCK
ON TABLE PCHOLD FORMAT HTML
END
```

Adding a Variable Description to a Filter

You can add a description for a variable that will replace the variable name in the form that prompts the user for a value. The description is appended to the variable name in the filter (WHERE) expression of the report request.

Syntax: How to Add a Variable Description to the Filter

'&variable.description.'

where:

&variable

Is the variable name, including the ampersand (&), for which the value is being prompted. If the variable value will be compared to an alphanumeric field, you must enclose the variable in single quotation marks ('). Character strings must be enclosed in single quotation marks to be handled by most database engines.

If the variable value contains a single quotation mark ('), use the QUOTEDSTRING suffix on the variable, instead of enclosing the value in single quotation marks.

description

Is a description of the variable that replaces the variable name in the prompt.

Note: The following are usage limitations for description values:

- The ampersand (&) character is supported within the description for a variable by specifying the pipe character (|) immediately after the & character in the description.
- The period (.) cannot be used in a description because a period is the delimiter to specify the beginning and end of the description value.
- The open parenthesis character must have a space before it.

Example: Adding a Variable Description

The following request provides a more descriptive name for the QTYSTOCK field within the Autoprompt form. The variable description will precede the field controls on the Autoprompt form. The variable &QTYSTOCK is not enclosed within single quotation marks (') because the QTY_IN_STOCK field is a numeric format.

```
TABLE FILE ibinccen/centord
SUM CENTORD.INVSEG.QTY_IN_STOCK
BY CENTORD.INVSEG.PRODNAME
WHERE CENTORD.INVSEG.QTY_IN_STOCK LT &QTYSTOCK.Quantity In Stock:.;
ON TABLE PCHOLD FORMAT HTML
END
```

Syntax: How to Add a Range of Values List

```
&variable.(FROM Range1 TO Range2).
```

where:

&variable

Is the numeric variable, including the ampersand (&), for which you are supplying a list of values.

Range1

Is the starting numeric value of the range of the list of values.

Range2

Is the ending numeric value of the range of the list of values.

Example: Adding a Range of Values List

The following request provides a list of numeric values that are valid for the QTY_IN_STOCK field. This list is populated with values from the CENTORD data source that fall within the range.

```
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STATE BY STORENAME BY PRODNAME
ON TABLE SUBHEAD
"Inventory Report"
WHERE QTY_IN_STOCK GT &STOCK.(FROM 5000 TO 10000).
END
```

Specifying a Format for a Variable

You can specify a format for a variable to specify how to evaluate the variable for validation and sorting.

Syntax: How to Specify a Format

```
&variable.(|FORMAT=format)
```

where:

&variable

Is the variable name, including the ampersand (&), for which the value is being prompted. If the variable value will be compared to an alphanumeric field, you must enclose the variable in single quotation marks ('). Character strings must be enclosed in single quotation marks to be handled by most database engines.

If the variable value contains a single quotation mark ('), use the QUOTEDSTRING suffix on the variable, instead of enclosing the value in single quotation marks.

format

Is the format of the field. The default value is D12.2. For information on field formats, see *Describing Data With WebFOCUS Language*.

Example: Specifying a Format

The following request specifies a seven digit integer format (I7) for the QTYSTOCK field.

```
TABLE FILE ibinccen/centord
SUM CENTORD.INVSEG.QTY_IN_STOCK
BY CENTORD.INVSEG.PRODNAME
WHERE CENTORD.INVSEG.QTY_IN_STOCK LT &QTYSTOCK.(|FORMAT=I7).Quantity In
Stock:;
ON TABLE PCHOLD FORMAT HTML
END
```

Note: If a non-numeric value is typed, a validation message displays.

Setting a Default Variable Value

You can set a default variable value. This value will be used in the report if one is not supplied by the user. The code that specifies a default variable value must be added before the report request.

Syntax: How to Set a Default Variable Value

```
-DEFAULT &variable=value
```

where:

&variable

Is the variable name, including the ampersand (&), for which the value is being prompted. If the variable value will be compared to an alphanumeric field, you must enclose the variable in single quotation marks ('). Character strings must be enclosed in single quotation marks to be handled by most database engines.

If the variable value contains a single quotation mark ('), use the QUOTEDSTRING suffix on the variable, instead of enclosing the value in single quotation marks.

value

Is the default value for the variable. Embedded single quotation marks are indicated by two contiguous single quotation marks (' '). Quotation marks are required around variables containing delimiters, which include spaces and commas (,).

To specify multiple default values for a variable that will be used with a multiselect filter, each value must be enclosed in two single quotation marks (' ') and include the operation used in the filter the variable is referenced in (for example, OR) between each value. In addition, the entire string must be enclosed in single quotation marks ('). For example, in the following example, each of the values within the string are enclosed in two single quotation marks (' ') and the entire string is enclosed in quotation marks, such that the beginning and ending of the string has three (3) quotation marks.

```
-DEFAULT &parm1 = '''value1'' OR ''value2'' OR ''value3''';
```

The default value _FOC_NULL allows users to make a parameter-based filter selection optional by including the *All Values* option in selection lists based on the Responsive Autoprompt template or the *Select All* option in selection lists based on the HTML Autoprompt template. When users select either of these options, internal processing selectively removes WebFOCUS filtering phrases (such as BY and WHERE) from that parameter before passing the procedure to the core WebFOCUS engine to run the request.

```
-DEFAULT &parm1 = _FOC_NULL;
```

For more information on _FOC_NULL, see *Internal Processing of _FOC_NULL* on page 205.

The default value FOC_NONE also allows users to make a parameter-based filter selection optional by including the *All Values* option in selection lists based on the Responsive Autoprompt template or the *Select All* option in selection lists based on the HTML Autoprompt template. In this case, when users select either of these options, internal processing removes all non-Dialogue Manager lines (those lines that do not begin with a '-' dash) that contain the value FOC_NONE. However, this process may not remove an entire filtering phase (such as BY or WHERE) that begins on a non-Dialogue Manager line, if that phrase is coded across multiple lines.

```
-DEFAULT &parm1 = FOC_NONE;
```

For more information on FOC_NONE, see *Internal Processing of FOC_NONE* on page 205.

Example: Setting a Default Variable Value

The following sets a default value of NY for the STATE field. Note that there must be an ampersand in front of the field name in the -DEFAULT command for the variable to contain a default attribute in the Autoprompt form.

```
-DEFAULT &STATE=NY
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STATE BY STORENAME BY PRODNAME
ON TABLE SUBHEAD
"Inventory Report"
WHERE STATE EQ '&STATE.2 letters for US State.'
WHERE STORENAME EQ '&STORENAME.Store Name.'
WHERE PRODNAME EQ '&PRODNAME.Product Name.'
END
```

Setting a Hidden Variable

Dialogue Manager variables can be given default values using the -DEFAULT command. These variables are returned in the XML describe information used for WebFOCUS parameter prompting features (HTML Autoprompt, HTML Canvas, and ReportCaster Scheduling).

You can initialize a variable value and prevent it from being used for WebFOCUS parameter prompting by using the -DEFAULTH command. Variables initialized with -DEFAULTH are not used for parameter prompting. Since these variables are not displayed by the parameter prompting features, they are hidden from the user.

The code that specifies a default variable value must be added before the report request.

Syntax: How to Define and Initialize a Hidden Variable

```
-DEFAULTH &variable=value
```

where:

&variable

Is the name of the hidden variable, including the ampersand (&), for which the value is being prompted. If the variable value will be compared to an alphanumeric field, you must enclose the variable in single quotation marks ('). Character strings must be enclosed in single quotation marks to be handled by most database engines.

If the variable value contains a single quotation mark ('), use the QUOTEDSTRING suffix on the variable, instead of enclosing the value in single quotation marks.

value

Is the initial value for the variable. Embedded single quotation marks are indicated by two contiguous single quotation marks (' '). Quotation marks are required around variables containing delimiters, which include spaces and commas (,).

To specify multiple default values for a variable that will be used with a multiselect filter, each value must be enclosed in two single quotation marks (' ') and include the operation used in the filter the variable is referenced in (for example, OR) between each value. In addition, the entire string must be enclosed in single quotation marks ('). For example, in the following example, each of the values within the string are enclosed in two single quotation marks (' ') and the entire string is enclosed in quotation marks, such that the beginning and ending of the string has three (3) quotation marks.

```
-DEFAULTH &parm1 = '''value1'' OR ''value2'' OR ''value3''';
```

Adding a Single-Select List of Values

You can add a single-select list of values. The values in the list can be static or dynamic.

Syntax: How to Add a Static Single-Select List of Values

```
'&variable.(value,value2[,value3][,value4]...
[|FORMAT=format] [,WITHIN=within]).[description.]'
```

where:

&variable

Is the variable, including the ampersand (&), for which you are supplying a list of values.

value, value2, value3, value4...

Are the values comprising the list of selectable variable values.

format

Specifies how to evaluate the variable for validation and sorting. The FORMAT attribute is for all filter types.

within

The WITHIN attribute is for chaining supported with the Responsive Autoprompt template. The dynamic or static filter type is required for the first field in a chained grouping that is created for fields within the same dimension hierarchy. The second and subsequent fields in the chained group must be a dynamic filter The filters should be in the order that the fields are referenced in the Master File in a dimension hierarchy. The parameter names must be the same name as the field name in the Master File.

description

Is an optional description of the variable. For details and restrictions, see *Adding a Variable Description to a Filter* on page 191.

Note: A static value cannot contain the comma character (,) when specified within the WHERE statement syntax because the comma character (,) is the delimiter character for specifying a display value. When there is a comma (,) in one or more static values, put the values in a file and use the dynamic list (FIND) functionality.

Example: Adding a Static Single-Select List of Values

The following provides a list of values that are valid for the STORENAME field. The user can select only one value from the list.

```
-DEFAULT &STATE=NY
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STATE BY STORENAME BY PRODNAME
ON TABLE SUBHEAD
"Inventory Report"
WHERE STATE EQ '&STATE.2 letters for US State.'
WHERE STORENAME EQ '&STORENAME.(eMart,TV City,Web Sales).Store Name.'
WHERE PRODNAME EQ '&PRODNAME.Product Name.'
END
```

Syntax: How to Add a Dynamic Single-Select List of Values

```
'&variable.(FIND return_fieldname [,display_fieldname] IN datasource
[|SORT=sortoption] [|FORMAT=format] [,WITHIN=within]).[description.]'
```

where:

&variable

Is the variable, including the ampersand (&), for which you are supplying a list of values.

return_fieldname

Is the name of the field containing the possible variable values that are returned to the FOCEXEC.

display_fieldname

Is the name of the field containing the possible variable values that are displayed in the Autoprompt form.

Note: Both *return_fieldname* and *display_fieldname* can be a DEFINE field in a Master File but not a DEFINE field in a procedure.

datasource

Is the data source that contains the fields specified in *return_fieldname* and *display_fieldname*. If the fields reside in a cross-reference file of a data source used in a join, use the data source name that contains the fields.

Note:

- For dynamic lists, the WebFOCUS Client constructs the request to obtain the values using the specified data source. All environmental commands, such as SET commands, needed to obtain the values from the specified data source must be issued in the WebFOCUS Server profile, user profile, or WebFOCUS Client profile.
- Dynamic list values that display in Responsive Autoprompt are organized in a case-insensitive sort order. Dynamic list values that display in HTML Autoprompt are organized in the sort order returned by the Reporting Server, which is determined by the operating system of the machine.

sortoption

Provides the ability to specify how to sort the values returned for a dynamic list. Valid values are:

- **ASCENDING**, which sorts in low to high order. This is the default value when sort processing is not specified.
- **DESCENDING**, which sorts in high to low order.

format

Specifies how to evaluate the variable for validation and sorting when the sort option is not specified. The FORMAT attribute is for all filter types.

within

The WITHIN attribute is for chaining supported with the Responsive Autoprompt template. The dynamic or static filter type is required for the first field in a chained grouping that is created for fields within the same dimension hierarchy. The second and subsequent fields in the chained group must be a dynamic filter. The filters should be in the order that the fields are referenced in the Master File in a dimension hierarchy. The parameter names must be the same name as the field name in the Master File.

description

Is an optional description of the variable. For details and restrictions, see *Adding a Variable Description to a Filter* on page 191.

Example: Adding a Dynamic Single-Select List of Values

The following provides a list of values that are valid for the product name field. This list is populated with values from the CENTORD data source. The user can select only one value from the list.

```
-DEFAULT &STATE=NY
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STATE BY STORENAME BY PROD_NUM
ON TABLE SUBHEAD
"Inventory Report"
WHERE STATE EQ '&STATE.2 letters for US State.'
WHERE STORENAME EQ '&STORENAME.(eMart,TV City,Web Sales).Store Name.'
WHERE PROD_NUM EQ '&PROD_NUM.(FIND PROD_NUM,PRODNAME IN CENTORD).Product
Name.'
END
```

Example: Adding a Dynamic Single-Select List of Values in Descending Sort Order

The following provides a list of values that are valid for the product name field. This list is populated with values from the CENTORD data source. The user can select only one value from the list.

```
-DEFAULT &STATE=NY
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STATE BY STORENAME BY PROD_NUM
ON TABLE SUBHEAD
"Inventory Report"
WHERE STATE EQ '&STATE.2 letters for US State.'
WHERE STORENAME EQ '&STORENAME.(eMart,TV City,Web Sales).Store Name.'
WHERE PROD_NUM EQ '&PROD_NUM.(FIND PROD_NUM,PRODNAME IN CENTORD|
SORT=DESCENDING).Product Name.'
END
```

Example: Chaining Values

Chained fields limit the values that are available to select from, based on the fields referenced in the Master File dimension hierarchy. The WITHIN attribute specifies to chain the filters. The first field in a chained group must be a dynamic or static list and the second and subsequent filters in the chain must be a dynamic filter. The following provides a list of chained values for the Product Type, Product Category, and Product Name fields. For example, the values listed for the Product Category field are those for the values selected for the Product Type field.

```
TABLE FILE ibinccen/centord
SUM CENTORD.INVSEG.QTY_IN_STOCK
CENTORD.INVSEG.PRICE
BY CENTORD.INVSEG.PRODTYPE
BY CENTORD.INVSEG.PRODCAT
BY CENTORD.INVSEG.PRODNAME
WHERE CENTORD.INVSEG.PRODTYPE EQ &PRODTYPE.(OR(FIND CENTORD.INVSEG.PRODTYPE
IN ibinccen/CENTORD |FORMAT=A19)).PRODTYPE:.;
WHERE CENTORD.INVSEG.PRODCAT EQ &PRODCAT.(OR(FIND CENTORD.INVSEG.PRODCAT IN
ibinccen/CENTORD |FORMAT=A22 ,WITHIN=PRODTYPE)).PRODCAT:.;
WHERE CENTORD.INVSEG.PRODNAME EQ &PRODNAME.(OR(FIND CENTORD.INVSEG.PRODNAME
IN ibinccen/CENTORD |FORMAT=A30 ,WITHIN=PRODCAT)).Product Name:.;
END
```

Adding a Multiselect List of Values

You can add a multiselect list of values. The values in the list can be static or dynamic.

Syntax: How to Add a Static Multiselect List of Values

```
&variable.(operation (<displayvalue1,value1>,
<displayvalue2,value2>,...<displayvalueN,valueN>)
[|FORMAT=format] [,WITHIN=within]).[description.]
```

where:

&variable

Is the variable, including the ampersand (&), for which you are supplying a list of values.

operation

Specifies how to evaluate multiple values. May contain the value OR, AND, or a comma (,). If omitted, the default value is OR.

displayvalue, displayvalue2, displayvalue3, displayvalue4...

Are the values comprising the list of selectable variable values that you can select.

value, value2, value3, value4...

Are the values comprising the list of selectable variable values passed to the reporting server.

format

Specifies how to evaluate the variable for validation and sorting. The FORMAT attribute is for all filter types.

within

The WITHIN attribute is for chaining supported with the Responsive Autoprompt template. The dynamic or static filter type is required for the first field in a chained grouping that is created for fields within the same dimension hierarchy. The second and subsequent fields in the chained group must be a dynamic filter The filters should be in the order that the fields are referenced in the Master File in a dimension hierarchy. The parameter names must be the same name as the field name in the Master File.

description

Is an optional description of the variable. For details and restrictions, see *Adding a Variable Description to a Filter* on page 191.

Note: A static value cannot contain the comma character (,) when specified within the WHERE statement syntax because the comma character (,) is the delimiter character for specifying a display value. When there is a comma (,) in one or more static values, put the values in a file and use the dynamic list (FIND) functionality.

Example: Adding a Static Multiselect List of Values

The following request provides a list of values that are valid for the STORENAME field. The user can select more than one value from the list.

```
-DEFAULT &STATE=NY
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STATE BY STORENAME BY PRODNAME
ON TABLE SUBHEAD
"Inventory Report"
WHERE STATE EQ '&STATE.2 letters for US State.'
WHERE STORENAME EQ &STORENAME.(OR(eMart,TV City,Web Sales)).Store Name.
WHERE PRODNAME EQ '&PRODNAME.Product Name.'
END
```

Syntax: How to Add a Dynamic Multiselect List of Values

```
&variable.(operation (FIND return_fieldname [,display_fieldname] IN
datasource[|SORT=sortoption] [|FORMAT=format] [,WITHIN=within])).
[description.]
```

where:

&variable

Is the variable, including the ampersand (&), for which you are supplying a list of values.

operation

Specifies how to evaluate multiple values. May contain the value OR, AND, or a comma (,). If omitted, the default value is OR.

return_fieldname

Is the name of the field containing the possible variable values that are returned to the FOCEXEC.

display_fieldname

Is the name of the field containing the possible variable values that are displayed in the Autoprompt form.

Note: Both *return_fieldname* and *display_fieldname* can be a DEFINE field in a Master File but not a DEFINE field in a procedure.

datasource

Is the data source that contains the fields specified in *return_fieldname* and *display_fieldname*. If the fields reside in a cross-reference file of a data source used in a join, use the data source name that contains the fields.

Note:

- For dynamic lists, the WebFOCUS Client constructs the request to obtain the values using the specified data source. All environmental commands, such as SET commands, needed to obtain the values from the specified data source must be issued in the WebFOCUS Server profile, user profile, or WebFOCUS Client profile.
- Dynamic list values that display in Responsive Autoprompt are organized in a case-insensitive sort order. Dynamic list values that display in HTML Autoprompt are organized in the sort order returned by the Reporting Server, which is determined by the operating system of the machine.

sortoption

Provides the ability to specify how to sort the values returned for a dynamic list. Valid values are:

- **ASCENDING**, which sorts in low to high order. This is the default value when sort processing is not specified.
- **DESCENDING**, which sorts in high to low order.

format

Specifies how to evaluate the variable for validation and sorting when the sort option is not specified. The FORMAT attribute is for all filter types.

within

The WITHIN attribute is for chaining supported with the Responsive Autoprompt template. The dynamic or static filter type is required for the first field in a chained grouping that is created for fields within the same dimension hierarchy. The second and subsequent fields in the chained group must be a dynamic filter. The filters should be in the order that the fields are referenced in the Master File in a dimension hierarchy. The parameter names must be the same name as the field name in the Master File.

description

Is an optional description of the variable. For details and restrictions, see *Adding a Variable Description to a Filter* on page 191.

Example: Adding a Dynamic Multiselect List of Values

The following request provides a list of values that are valid for the state field. This list is populated with values from the CENTORD data source. The user can select more than one value from the list.

```
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STORE_CODE BY STATE BY PRODNAME
ON TABLE SUBHEAD
"Inventory Report"
WHERE STORE_CODE EQ &STORE_CODE.(OR(FIND STORE_CODE,STORENAME IN
CENTORD)).Store Name.
WHERE STATE EQ &STATE.(OR(CA,IL,MA,NY,NJ,FL,TX)).2 letters for US State.
WHERE PRODNAME EQ '&PRODNAME.(FIND PRODNAME IN CENTORD).Product Name.'
END
```

Example: Adding a Dynamic Multiselect List of Values in Descending Sort Order

The following request provides a list of values that are valid for the state field. This list is populated with values from the CENTORD data source. The user can select more than one value from the list.

```
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STORE_CODE BY STATE BY PRODNAME
ON TABLE SUBHEAD
"Inventory Report"
WHERE STORE_CODE EQ &STORE_CODE.(OR(FIND STORE_CODE,STORENAME IN CENTORD|
SORT=DESCENDING)).Store Name.
WHERE STATE EQ &STATE.(OR(CA,IL,MA,NY,NJ,FL,TX)).2 letters for US State.
WHERE PRODNAME EQ '&PRODNAME.(FIND PRODNAME IN CENTORD).Product Name.'
END
```

Reference: Rules for a Multiselect List of Values

- When creating a multiselect list of values, do not add single quotation marks (') around individual values. WebFOCUS automatically encloses each value in single quotation marks (') for multiselect lists. If you add your own single quotation marks ('), an error will result. For example, WHERE COUNTRY EQ '&COUNTRY' will generate an error. Use &COUNTRY without quotation marks.
- When using the result of a multiselected list of values as a parameter in a drill down, use QUOTEDSTRING to handle the embedded blanks that are added as separators between the parameters.

  The following example is correct:

  ```
  FOCEXEC=DRILLDOWN(COUNTRY=&COUNTRY.QUOTEDSTRING OTHER=xxx)
  ```

  The next example is not correct, and the embedded blanks within the &COUNTRY parameter will terminate the drill down.

  ```
  FOCEXEC=DRILLDOWN(COUNTRY=&COUNTRY OTHER=xxx)
  ```

  For more information, see *Creating a Standard Quote-Delimited String* on page 365.

Reference: Selecting Values for Multiselect Variables

- If your report contains a static multiselect list of values and an HTML Autoprompt template is configured, you can select the *Select All* option to select all values in the parameters list. Internally, selecting the Select All option results in the parameter being set to all the values in the multiselect list separated by the qualifier, AND, OR, or the comma character (,) in the WHERE statement. For more information, see *Internal Processing of Select All* on page 206.
- If your report contains a dynamic multiselect list of values, you can choose the *All Values* option to not select any values from the parameter list. The All Values option is available with the Responsive and HTML Autoprompt templates. Internally, either the _FOC_NULL value or the FOC_NONE value is passed for the dynamic multiselect list *All Values* option. By default, _FOC_NULL is passed. However, you can control which of these values is passed for the *All Values* option with the *IBIF_describe_null* variable that configures the value the WebFOCUS Client will send.
- Individual phrases and expressions that contain the value _FOC_NULL are removed from the procedure prior to the procedure being run by the reporting server. For more information about _FOC_NULL, see *Internal Processing of _FOC_NULL* on page 205.

- Non-Dialogue manager lines (do not begin with a '-' dash) that contain the value FOC_NONE are removed from the procedure prior to the procedure being run by the reporting server. For more information about FOC_NONE, see *Internal Processing of FOC_NONE* on page 205.

Reference: Internal Processing of _FOC_NULL

When you select or manually code the *All Values* option within a multiselect list, the parameter value sent to the WebFOCUS Server is _FOC_NULL, by default. Internal processing is then performed by the WebFOCUS Server Describe layer to search the procedure and selectively remove WebFOCUS phrases (such as BY and WHERE) or expressions that contain _FOC_NULL. The procedure is then passed to the core WebFOCUS engine to run the request.

The following is a WHERE example for selecting or manually coding the *All Values* option that generates the _FOC_NULL value for dynamic multiselect lists:

```
WHERE CATEGORY EQ &CATEGORY.(OR(FIND CATEGORY IN MOVIES)).CATEGORY. AND
DIRECTOR EQ &DIRECTOR.(OR(FIND DIRECTOR IN MOVIES)).DIRECTOR.;
```

If *All Values* is returned for CATEGORY, the test for DIRECTOR will still be processed.

The following is a WHERE example for selecting or manually coding the *All Values* option that generates the _FOC_NULL value for a static multiselect list and a dynamic multiselect list:

```
WHERE COPIES EQ
&COPIES.(<All Values,_FOC_NULL>,<1,1>,<2,2>,<3,3>).COPIES.
AND DIRECTOR EQ &DIRECTOR.(OR(FIND DIRECTOR IN MOVIES)).DIRECTOR.;
```

Each list must be coded completely on a single line in the procedure.

When coding a HEADING, FOOTING, SUBHEAD, or SUBFOOT, you should have a minimum of one line that does not contain a parameter that could be assigned the value of _FOC_NULL.

Reference: Internal Processing of FOC_NONE

When you select or manually code the *All Values* option within a multiselect list, the parameter value that may be sent to the WebFOCUS Server is FOC_NONE. Internal processing is then performed by the WebFOCUS Server Describe layer to search the procedure and remove all non-Dialogue Manager lines (do not begin with a dash '-') that contain FOC_NONE. The procedure is then passed to the core WebFOCUS engine to run the request.

The following are WHERE examples for selecting or manually coding the All Values option that generates the FOC_NONE value:

- Dynamic multiselect

```
WHERE CATEGORY EQ &CATEGORY.(OR(FIND CATEGORY IN MOVIES)).CATEGORY.;
```

- Static multiselect with FOC_NONE (coded on a single line in the procedure (FEX))

```
WHERE COPIES EQ &COPIES.(<All Values,FOC_NONE>,<1,1>,<2,2>,<3,3>).
COPIES.;
```

The following are FOC_NONE coding considerations:

- When coding a HEADING, FOOTING, SUBHEAD, or SUBFOOT, you should have a minimum of one line that does not contain a parameter that could be assigned the value of FOC_NONE.
- If the All Values option is used within a dynamic multiselect list, or if a manually coded WHERE statement passes the FOC_NONE parameter value:
  - Code a WHERE statement that passes the FOC_NONE value on a single line in the procedure (FOCEXEC). This prevents the WHERE statement from being partially removed from the request and avoids a FOC002 error.
  - Do not code a complex WHERE statement on a single line when FOC_NONE is passed for a field. This prevents the entire complex WHERE statement from being removed from the request. Note that a complex WHERE statement has selection tests for more than one field using AND or OR operators.

Reference: Internal Processing of Select All

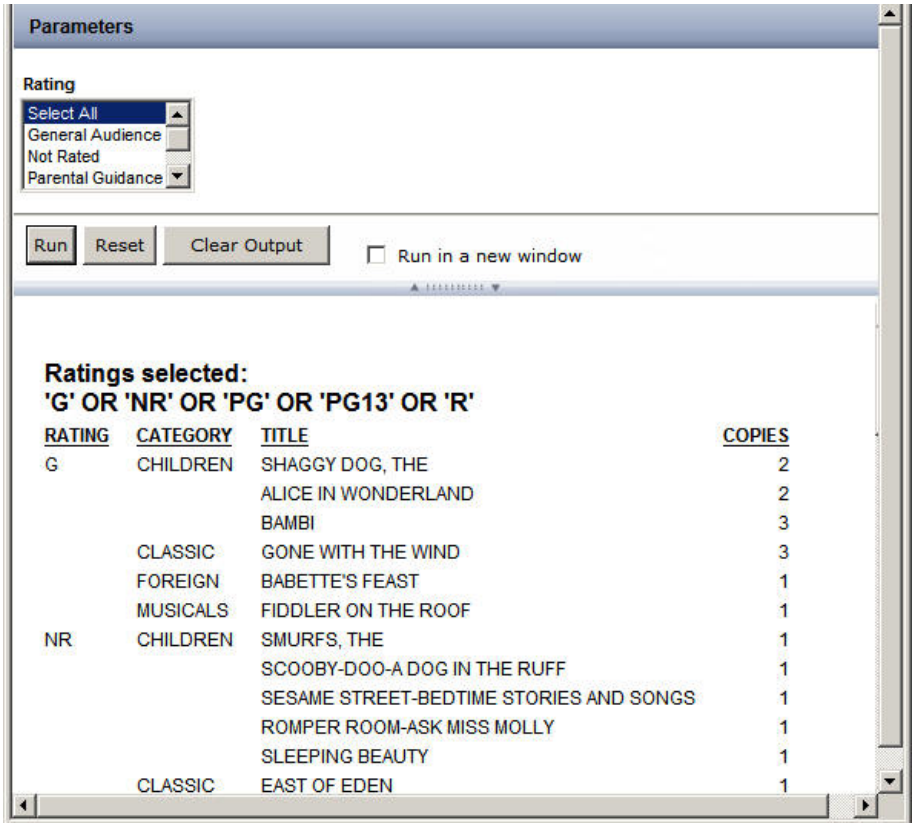
When you run a request with a static multiselect list, the HTML Autoprompt facility dynamically adds the *Select All* option to the list of values that, when selected, assigns all the values in the static multiselect list to the parameter. When the Select All and individual values are selected, the Select All value is ignored.

Internal processing is then done by the Autoprompt facility to assign the parameter the select list values separated by the qualifier (for example, AND or OR) specified in the WHERE statement that references the parameter.

The following example shows a WHERE statement using an OR qualifier with a static multiselect list for the parameter &RATING. The parameter &RATING is referenced in the WHERE statement and in the HEADING. The HEADING shows how the parameter value is set to all values in the multiselect list separated by the OR qualifier.

```
TABLE FILE MOVIES
HEADING
" Ratings selected: "
" &RATING "
PRINT
     'MOVIES.MOVINFO.TITLE'
     'MOVIES.MOVINFO.COPIES'
BY 'MOVIES.MOVINFO.RATING'
BY 'MOVIES.MOVINFO.CATEGORY'
HEADING
""
FOOTING
""
WHERE MOVIES.MOVINFO.RATING EQ &RATING.(OR(<General Audience,G>,<Not
Rated,NR>,<Parental Guidance,PG>,<PG Over 13,PG13>,<R Over 18,R>)).Rating.;
ON TABLE SET PAGE-NUM OFF
ON TABLE NOTOTAL
ON TABLE PCHOLD FORMAT HTML
ON TABLE SET HTMLCSS ON
ON TABLE SET STYLE *
     UNITS=IN,
     SQUEEZE=ON,
     ORIENTATION=PORTRAIT,
$
TYPE=REPORT,
     GRID=OFF,
     FONT='ARIAL',
     SIZE=9,
$
TYPE=TITLE,
     STYLE=BOLD,
$
TYPE=HEADING,
     SIZE=12,
     STYLE=BOLD,
$
ENDSTYLE
END
```

The following image shows the HTML Autoprompt form and report when *Select All* is selected.



The WebFOCUS Client assigns &RATING the list of values by creating and adding the following syntax to the request sent to the Reporting Server:

```
-SET &RATING='''G'' OR ''NR'' OR ''PG'' OR ''PG13'' OR ''R''';
```

Adding Static Lists of Display and Sort Fields

In a static list of fields, you specify the field names and display values in the select list for the display or sort command.

Syntax: How to Add a Static Single-Select List of Sort or Display Fields

```
cmd &var.(<dsply1[,val1>],<dsply2[,val2>],...<dsplyN[,valN>]).[desc.]
```

where:

cmd

Is the command to which the list of fields will apply. Valid values are PRINT, COUNT, SUM, WRITE, ADD, BY, and ACROSS.

&var

Is the variable, including the ampersand (&), for which you are supplying a list of field values.

dsply1, dsply2, ...

Are the entries that display on the Autoprompt form. Selecting an entry automatically selects the corresponding field.

val1, val2, ...

Are the field names passed to the Reporting Server. They can be qualified field names and can be omitted if the display value is identical to the field name.

desc

Is an optional description of the variable.

Note: The entire command, up to and including the description, if there is one, must appear on one line of the report request.

Example: Adding Single-Select Lists of Sum Fields and Sort Fields

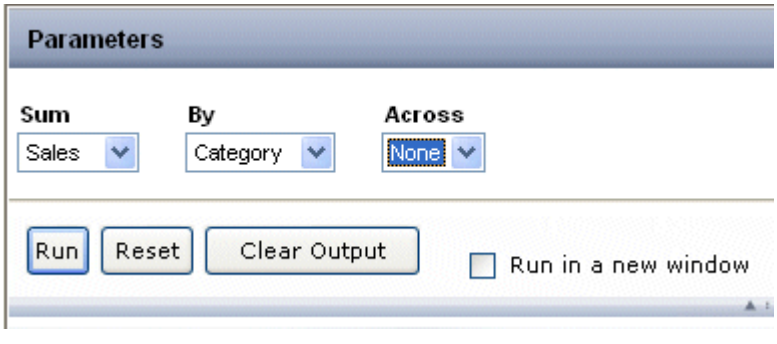
The following request against the GGSALES data source enables the user to select one of the following from each list when running the procedure:

- Sum: DOLLARS, BUDDOLLARS, UNITS.
- BY fields: CATEGORY, PRODUCT.
- ACROSS fields: REGION, ST, None.

  Note: For information about the All Values option, see *Internal Processing of FOC_NONE* on page 205.

```
TABLE FILE GGSALES
HEADING
"Sales Report Summary for: "
"Fields: &SumFlds"
"Sort: &SortBy"
"Across: &Acrs"
" "
SUM &SumFlds.(<Sales,DOLLARS>,<Budget,BUDDOLLARS>,<Units,UNITS>).Sum.
BY &SortBy.(<Category,GGSALES.SALES01.CATEGORY>,<PRODUCT>).By.
ACROSS &Acrs.(<Area,REGION>,<State,ST>,<None,FOC_NONE>).Across.
WHERE REGION NE 'West'
ON TABLE SET PAGE NOPAGE
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF, FONT=ARIAL,SIZE=10,SQUEEZE=ON,$
TYPE=HEADING, FONT=ARIAL, STYLE=BOLD, JUSTIFY=LEFT,$
TYPE=HEADING, LINE=1, FONT=ARIAL, STYLE=BOLD, JUSTIFY=LEFT,$
TYPE=TITLE, BACKCOLOR=BLACK, COLOR=WHITE,$
TYPE=ACROSSTITLE, BACKCOLOR=LIGHT BLUE, COLOR=BLUE,$
TYPE=ACROSSVALUE, BACKCOLOR=LIGHT GRAY, COLOR=GREEN,$
END
```

Running the request opens the following Autoprompt form on which the selection for the REGION sort field displays as Area, the selection for the DOLLARS field displays as Sales, the selection for the CATEGORY sort field and the BUDDOLLARS and UNITS fields display in mixed-case, and the display value for PRODUCT is the same as the sort field name. Only one selection is allowed from each list.



Selecting *Sales*, *Category*, and *None* produces the following report:

Sales Report Summary for:
Fields: DOLLARS
Sort: GGSALES.SALES01.CATEGORY

Category	Dollar Sales
Coffee	12757938
Food	13026996
Gifts	8718410

Syntax: How to Add a Static Multiselect List of Sort Fields

```
cmd &var.(cmd(<dsply1[,val1>],<dsply2[,val2>],...<dsplyN[,srtN>])).
[desc.]
```

where:

cmd

Is the command to which the list of fields will apply. Valid values are BY and ACROSS. Adding the command around the select list creates a multiselect list.

&var

Is the variable, including the ampersand (&), for which you are supplying a list of field values.

dsply1, dsply2, ...

Are the entries that display on the Autoprompt form. Selecting an entry automatically selects the corresponding sort field.

val1, val2, ...

Are the field names passed to the Reporting Server. They can be qualified field names and can be omitted if the display value is identical to the field name.

desc

Is an optional description of the variable.

Note: The entire command, up to and including the description, if there is one, must appear on one line of the report request.

Syntax: How to Add a Static Multiselect List of Display Fields

```
cmd &var.(AND(<dsply1[,val1>],<dsply2[,val2>],...<dsplyN[,srtN>])).[desc.]
```

where:

cmd

Is the command to which the list of fields will apply. Valid values are PRINT, COUNT, SUM, WRITE, and ADD.

&var

Is the variable, including the ampersand (&), for which you are supplying a list of field values.

AND

Is the connector that creates a multiselect list.

dsply1, dsply2, ...

Are the entries that display on the Autoprompt form. Selecting an entry automatically selects the corresponding sort field.

val1, val2, ...

Are the field names passed to the Reporting Server. They can be qualified field names and can be omitted if the display value is identical to the field name.

desc

Is an optional description of the variable.

Note: The entire command, up to and including the description, if there is one, must appear on one line of the report request.

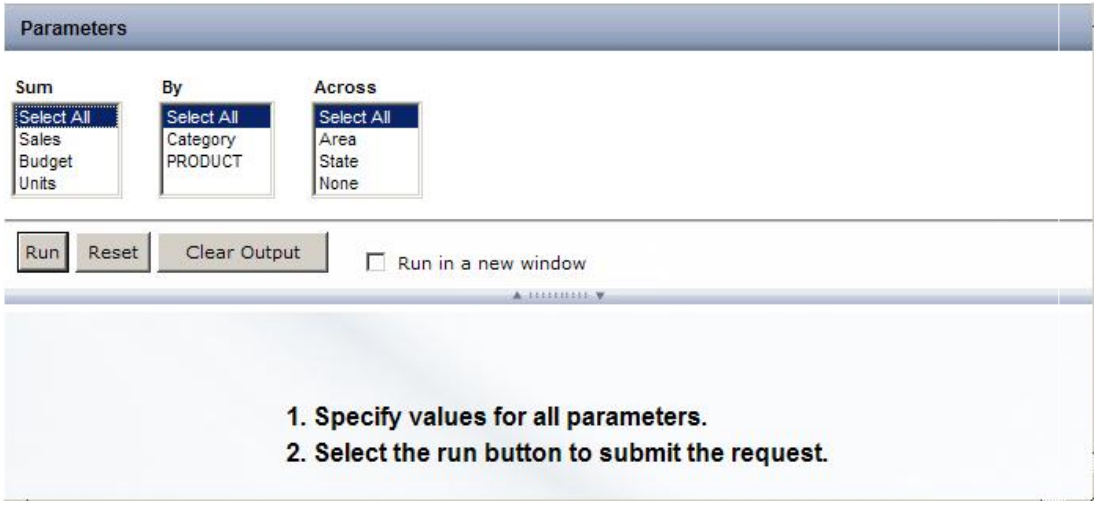
Example: Adding Multiselect Lists of Sum Fields and Sort Fields

The following request against the GGSALES data source enables the user to select one or more of the following when running the procedure:

- Sum: DOLLARS, BUDDOLLARS, UNITS.
- BY fields: CATEGORY, PRODUCT.
- ACROSS fields: REGION, ST, None.

```
TABLE FILE GGSALES
HEADING
"Sales Report Summary for: "
"Fields: &SumFlds"
"Sort: &SortBy"
"Across: &Acrs"
" "
SUM &SumFlds.(AND(<Sales,DOLLARS>,<Budget,BUDDOLLARS>,<Units,UNITS>)).Sum.
BY &SortBy.(BY(<Category,GGSALES.SALES01.CATEGORY>,<PRODUCT>)).By.
ACROSS &Acrs.(ACROSS(<Area,REGION>,<State,ST>,<None,FOC_NONE>)).Across.
WHERE REGION NE 'West'
ON TABLE SET PAGE NOPAGE
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF, FONT=ARIAL,SIZE=10,SQUEEZE=ON,$
TYPE=HEADING, FONT=ARIAL, STYLE=BOLD, JUSTIFY=CENTER,$
TYPE=HEADING, LINE=1, FONT=ARIAL, STYLE=BOLD, JUSTIFY=CENTER,$
TYPE=TITLE, BACKCOLOR=BLACK, COLOR=WHITE,$
TYPE=ACROSSTITLE, BACKCOLOR=LIGHT BLUE, COLOR=BLUE,$
TYPE=ACROSSVALUE, BACKCOLOR=LIGHT GRAY, COLOR = GREEN,$
END
```

Running the request opens the following Autoprompt form on which the selection for the REGION sort field displays as Area, the selection for the DOLLARS field displays as Sales, the selection for the CATEGORY sort field and the BUDDOLLARS and UNITS fields display in mixed-case, and the display value for PRODUCT is the same as the sort field name:



Selecting *Sales*, *Units*, *Category*, and *Area* produces the following report:

Sales Report Summary for:
Fields: DOLLARS AND UNITS
Sort: GGSALES.SALES01.CATEGORY
Across: REGION

	Region					
	Midwest		Northeast		Southeast	
Category	Dollar Sales	Unit Sales	Dollar Sales	Unit Sales	Dollar Sales	Unit Sales
Coffee	4178513	332777	4164017	335778	4415408	350948
Food	4338271	341414	4379994	353368	4308731	349829
Gifts	2883881	230854	2848289	227529	2986240	234455

Adding Dynamic Lists of Display and Sort Fields

In a dynamic list of fields, you use the FIND command in the select list to retrieve field names and display values from a WebFOCUS data source.

Reference: Creating Data Sources Containing Field Names

The following examples create three (3) data sources to be used in the examples that create dynamic field lists:

- GGDETFLD, which contains names of fields from the GGSALES data source that will be used as display fields in select lists.
- GGBYFLD, which contains names of fields from the GGSALES data source that will be used as BY fields in select lists.
- GGACRFLD, which contains names of fields from the GGSALES data source that will be used as ACROSS fields in select lists.

Example: Creating the GGDETFLD Data Source

The following request creates the FOCUS data source named GGDETFLD:

```
APP HOLD IBISAMP
DEFINE FILE SYSCOLUM
FLDDESC/A20=IF NAME EQ 'DOLLARS' THEN 'Dollars'
  ELSE IF NAME EQ 'BUDDOLLARS' THEN 'Budget'
  ELSE IF NAME EQ 'UNITS' THEN 'Units' ELSE 'N/A';
END
TABLE FILE SYSCOLUM
PRINT FLDDESC
BY NAME
WHERE TBNAME EQ 'GGSALES'
WHERE NAME EQ 'DOLLARS' OR 'BUDDOLLARS' OR 'UNITS'
ON TABLE HOLD AS GGDETFLD FORMAT FOCUS
END
```

Example: Creating the GGBYFLD Data Source

The following request creates the FOCUS data source named GGBYFLD:

```
APP HOLD IBISAMP
DEFINE FILE SYSCOLUM
FLDDESC/A20=IF NAME EQ 'CATEGORY' THEN 'Category'
  ELSE IF NAME EQ 'PRODUCT' THEN 'Product' ELSE 'N/A';
END
TABLE FILE SYSCOLUM
PRINT FLDDESC
BY NAME
WHERE TBNAME EQ 'GGSALES'
WHERE NAME EQ 'CATEGORY' OR 'PRODUCT'
ON TABLE HOLD AS GGBYFLD FORMAT FOCUS
END
```

Example: Creating the GGACRFLD Data Source

The following request creates the FOCUS data source named GGACRFLD:

```
APP HOLD IBISAMP
DEFINE FILE SYSCOLUM
FLDDESC/A20=IF NAME EQ 'REGION' THEN 'Region'
  ELSE IF NAME EQ 'ST' THEN 'State' ELSE 'N/A';
END
TABLE FILE SYSCOLUM
PRINT FLDDESC
BY NAME
WHERE TBNAME EQ 'GGSALES'
WHERE NAME EQ 'REGION' OR 'ST'
ON TABLE HOLD AS GGACRFLD FORMAT FOCUS
END
```

Syntax: How to Add a Dynamic Single-Select List of Display or Sort Fields

```
cmd &var.(FIND return_fieldname [,display_fieldname] IN datasource).
[description.]
```

where:

cmd

Is the command to which the list of fields will apply. Valid values are PRINT, COUNT, SUM, WRITE, ADD, BY, and ACROSS.

&var

Is the variable, including the ampersand (&), for which you are supplying a list of field values.

return_fieldname

Is the name of the field containing the possible variable values that are returned to the FOCEXEC.

display_fieldname

Is the name of the field containing the possible variable values that are displayed in the Autoprompt form.

Note: Both *return_fieldname* and *display_fieldname* can be a DEFINE field in a Master File, but not a DEFINE field in a procedure.

datasource

Is the data source that contains the fields specified in *return_fieldname* and *display_fieldname*. If the fields reside in a cross-reference file of a data source used in a join, use the data source name that contains the fields.

Note:

- For dynamic lists, the WebFOCUS Client constructs the request to obtain the values using the specified data source. All environmental commands, such as SET commands, needed to obtain the values from the specified data source must be issued in the WebFOCUS Server profile, user profile, or WebFOCUS Client profile.
- Dynamic list values that display in Responsive Autoprompt are organized in a case-insensitive sort order. Dynamic list values that display in HTML Autoprompt are organized in the sort order returned by the Reporting Server, which is determined by the operating system of the machine.

description

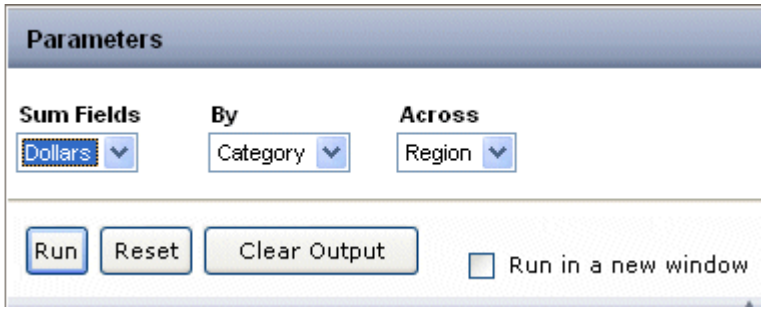
Is an optional description of the variable.

Example: **Creating Dynamic Single-Select Lists of Display and Sort Fields**

The following request against the GGSALES data source creates three (3) single-select dynamic lists by retrieving field names from the FOCUS data sources GGDETFLD, GGBYFLD, and GGACRFLD:

```
TABLE FILE GGSALES
HEADING
"Sales Report Summary for: "
"Fields: &SumFlds"
"Sort: &SortBy"
"Across: &Acrs"
" "
SUM &SumFlds.(FIND NAME, FLDDESC IN GGDETFLD).Sum Fields.
BY &SortBy.(FIND NAME, FLDDESC IN GGBYFLD).By.
ACROSS &Acrs.(FIND NAME, FLDDESC IN GGACRFLD).Across.
WHERE REGION NE 'West'
ON TABLE SET PAGE NOPAGE
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF, FONT=ARIAL,SIZE=10,SQUEEZE=ON,$
TYPE=HEADING, FONT=ARIAL, STYLE=BOLD, JUSTIFY=CENTER,$
TYPE=HEADING, LINE=1, FONT=ARIAL, STYLE=BOLD, JUSTIFY=CENTER,$
TYPE=TITLE, BACKCOLOR=BLACK, COLOR=WHITE,$
TYPE=ACROSSTITLE, BACKCOLOR=LIGHT BLUE, COLOR=BLUE,$
TYPE=ACROSSVALUE, BACKCOLOR=LIGHT GRAY, COLOR = GREEN,$
END
```

Running the request opens the following Autoprompt form with single-select lists for the SUM, BY, and ACROSS commands:



Selecting *Dollars* on the Sum Fields drop-down list, *Category* on the By drop-down list, and *Region* on the Across drop-down list produces the following report:

Sales Report Summary for:
Fields: DOLLARS
Sort: CATEGORY
Across: REGION

	Region		
	Midwest	Northeast	Southeast
Category			
Coffee	4178513	4164017	4415408
Food	4338271	4379994	4308731
Gifts	2883881	2848289	2986240

Syntax: How to Add a Dynamic Multiselect List of Sort Fields

```
cmd &var.(cmd(FIND return_fieldname [,display_fieldname]
IN datasource)).[description.]
```

where:

cmd

Is the command to which the list of fields will apply. Valid values are BY and ACROSS. Adding the command around the select list creates a multiselect list of values.

&var

Is the variable, including the ampersand (&), for which you are supplying a list of field values.

return_fieldname

Is the name of the field containing the possible variable values that are returned to the FOCEXEC.

display_fieldname

Is the name of the field containing the possible variable values that are displayed in the Autoprompt form.

Note: Both *return_fieldname* and *display_fieldname* can be a DEFINE field in a Master File, but not a DEFINE field in a procedure.

datasource

Is the data source that contains the fields specified in *return_fieldname* and *display_fieldname*. If the fields reside in a cross-reference file of a data source used in a join, use the data source name that contains the fields.

Note:

- For dynamic lists, the WebFOCUS Client constructs the request to obtain the values using the specified data source. All environmental commands, such as SET commands, needed to obtain the values from the specified data source must be issued in the WebFOCUS Server profile, user profile, or WebFOCUS Client profile.
- Dynamic list values that display in Responsive Autoprompt are organized in a case-insensitive sort order. Dynamic list values that display in HTML Autoprompt are organized in the sort order returned by the Reporting Server, which is determined by the operating system of the machine.

description

Is an optional description of the variable.

Syntax: How to Add a Dynamic Multiselect List of Display Fields

```
cmd &var.(AND(FIND return_fieldname [,display_fieldname]
IN datasource)).[description.]
```

where:

cmd

Is the command to which the list of fields will apply. Valid values are PRINT, COUNT, SUM, WRITE, and ADD.

&var

Is the variable, including the ampersand (&), for which you are supplying a list of field values.

AND

Is the connector that creates a multiselect list.

return_fieldname

Is the name of the field containing the possible variable values that are returned to the FOCEXEC.

display_fieldname

Is the name of the field containing the possible variable values that are displayed in the Autoprompt form.

Note: Both *return_fieldname* and *display_fieldname* can be a DEFINE field in a Master File, but not a DEFINE field in a procedure.

datasource

Is the data source that contains the fields specified in *return_fieldname* and *display_fieldname*. If the fields reside in a cross-reference file of a data source used in a join, use the data source name that contains the fields.

Note:

- For dynamic lists, the WebFOCUS Client constructs the request to obtain the values using the specified data source. All environmental commands, such as SET commands, needed to obtain the values from the specified data source must be issued in the WebFOCUS Server profile, user profile, or WebFOCUS Client profile.
- Dynamic list values that display in Responsive Autoprompt are organized in a case-insensitive sort order. Dynamic list values that display in HTML Autoprompt are organized in the sort order returned by the Reporting Server, which is determined by the operating system of the machine.

description

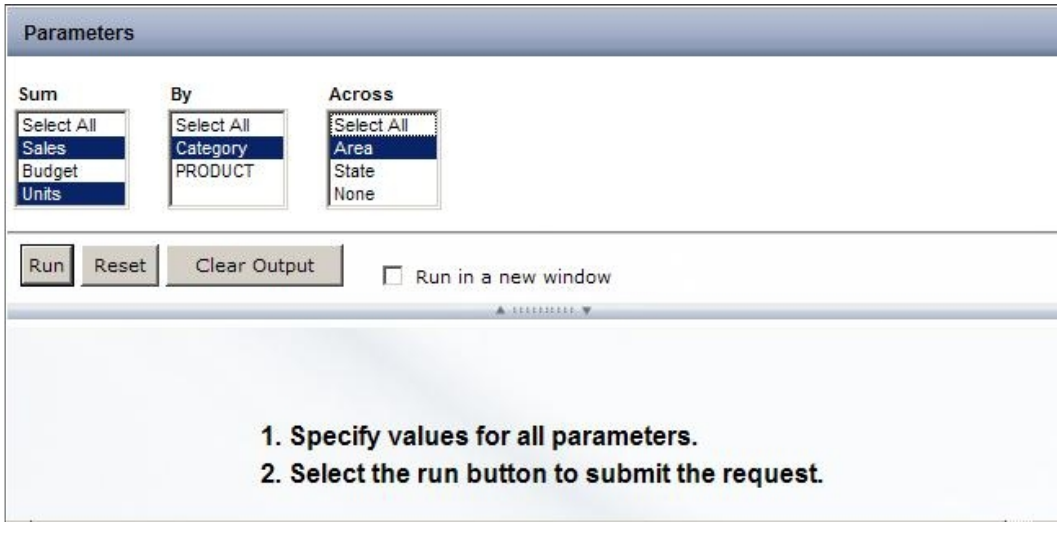
Is an optional description of the variable.

Example: Creating Dynamic Multiselect Lists of Display and Sort Fields

The following request against the GGSALES data source creates three multiselect dynamic lists by retrieving field names from the FOCUS data sources GGDETFLD, GGBYFLD, and GGACRFLD:

```
TABLE FILE GGSALES
HEADING
"Sales Report Summary for: "
"Fields: &SumFlds"
"Sort: &SortBy"
"Across: &Acrs"
" "
SUM &SumFlds.(AND(FIND NAME, FLDDESC IN GGDETFLD)).Sum Fields.
BY &SortBy.(BY(FIND NAME, FLDDESC IN GGBYFLD)).By.
ACROSS &Acrs.(ACROSS(FIND NAME, FLDDESC IN GGACRFLD)).Across.
WHERE REGION NE 'West'
ON TABLE SET PAGE NOPAGE
ON TABLE PCHOLD FORMAT PDF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF, FONT=ARIAL,SIZE=10,SQUEEZE=ON,$
TYPE=HEADING, FONT=ARIAL, STYLE=BOLD, JUSTIFY=CENTER,$
TYPE=HEADING, LINE=1, FONT=ARIAL, STYLE=BOLD, JUSTIFY=CENTER,$
TYPE=TITLE, BACKCOLOR=BLACK, COLOR=WHITE,$
TYPE=ACROSSTITLE, BACKCOLOR=LIGHT BLUE, COLOR=BLUE,$
TYPE=ACROSSVALUE, BACKCOLOR=LIGHT GRAY, COLOR = GREEN,$
END
```

Running the request opens the following Autoprompt form with multiselect lists for the SUM, BY, and ACROSS commands.



Selecting *Dollars*, *Units*, *Category*, and *Region* produces the following report:

Sales Report Summary for:
Fields: DOLLARS AND UNITS
Sort: CATEGORY
Across: REGION

	Region					
	Midwest		Northeast		Southeast	
Category	Dollar Sales	Unit Sales	Dollar Sales	Unit Sales	Dollar Sales	Unit Sales
Coffee	4178513	332777	4164017	335778	4415408	350948
Food	4338271	341414	4379994	353368	4308731	349829
Gifts	2883881	230854	2848289	227529	2986240	234455

Customizing the Autoprompt Facility

The topics in this section provide information on how you can customize the Autoprompt run-time settings and the look and feel of the Autoprompt form.

Specifying the HTML Template In a Procedure

If you want to specify the HTML template to use when a report is run, include the following code at the beginning of the procedure (FEX):

```
-<describe_html>template</describe_html>
```

where:

template

Is set to one of the following template values:

- **autoprompt_top.** Displays the parameters horizontally at the top of the page and is the default template value.
- **autoprompt_top_checked.** Is the same as autoprompt_top, but the *Run in a new window check box* is preselected.

Customizing the HTML Autoprompt Facility

You can customize the look and feel of the HTML Autoprompt facility by editing the template file. The template file (autoprompt_top.css) is located in the ibi\WebFOCUS82\ibi_html \javaassist\ibi\html\describe directory. Make a backup copy of the autoprompt_top.css file before making any changes to the file.

If you want to customize the banner, create an image, save it in the describe directory, and change the background-image property, which is shown in bold type in the following Cascading Style Sheet (CSS) code:

```
#idBannerDiv {
height:41px;
background-image:url(style/logo_banner_TOP.gif);
background-position:top left;
background-repeat:no-repeat;
margin:0px;
margin-top:0px;
cursor:pointer; }
```

The option to select different templates can be set in the WebFOCUS Administration Console using the Parameter Prompting selection under Application Settings. The setting to use to select the Autoprompt page depends upon the type of Autoprompt implementation you select in the Default Autoprompt Template (IBI_DESCRIBE_TEMPLATES) setting. For more information, see *Autoprompt Configuration* on page 169.

- **HTML_Top**. Specifies the use of the HTML-based implementation and the autoprompt_top.html template, which displays parameters horizontally at the top of the page.
- **HTML_Top_Checked**. Specifies the use of the HTML-based implementation and the autoprompt_top_checked.html template. In this template, the Run in a new window check box is pre-selected, specifying that all reports open in a new window, by default.

Specifying the HTML Template In a Procedure

If you want to specify the HTML template to use when a report is run, include the following code at the beginning of the procedure (FEX):

```
-<describe_html>template</describe_html>
```

where:

template
Is set to one of the following template values:

- **autoprompt_top.** Displays the parameters horizontally at the top of the page and is the default template value.
- **autoprompt_top_checked.** Is the same as autoprompt_top, but the *Run in a new window check box* is preselected.

Specifying the Prompting Level in a URL to Run a Report

You can specify Autoprompt settings directly in a URL to run a report located in an application directory on the Reporting Server. The WebFOCUS Client configuration setting that you can specify for parameter prompting is:

```
IBIF_wfdescribe
```

The possible values for this setting are:

- **OFF.** Turns Autoprompt off. This is the default configuration value for self-service requests.
- **XMLRUN.** Only prompts for variables created with -DEFAULT when there is another variable that does not have a value assigned and, therefore, will be prompted for.
- **XMLPROMPT.** Prompts for variables created with -DEFAULT and for any other variable that does not have a value.
- **XML.** The XML document describing the variables is displayed in the browser. This setting is used internally by the WebFOCUS tools and is recommended for debugging purposes only.

When a request is sent to the WebFOCUS Client to run a report, the WebFOCUS Client adds the following parameter to the request sent to the Reporting Server:

```
WFDESCRIBE=value
```

where:

`WFDESCRIBE`
Communicates to the Reporting Server the required level of parameter prompting evaluation.

value
Is the value of the WebFOCUS Client configuration parameter prompting setting, which can be specified in the URL sent to the WebFOCUS Client to run the report request, using the IBIF_wfdescribe setting.

You can use the WebFOCUS Client variable *IBIC_server* to specify the server node for the report. For more information about IBIC_server, see *WebFOCUS Security and Administration*. The following is an example of a URL used to run a WebFOCUS Reporting Server procedure named *salessummary* on the Reporting Server node EDASERVE with IBIF_wfdescribe set to XMLRUN:

```
http://hostname[:port]/ibi_apps/WFServlet?IBIF_ex=salessummary
&IBIC_server=EDASERVE&IBIF_wfdescribe=XMLRUN
```

Note: Changes to the parameter prompting configuration setting should be evaluated in a development or test environment, and the evaluation should include running existing reports and applications.

For more information about IBIF_wfdescribe, see *WebFOCUS Security and Administration*.

Adding a Description of the Procedure to the Autoprompt Internal XML

You can add a description of the procedure to be included in the internal XML used by the Autoprompt facility. This provides information about the Autoprompt form that can be useful for application integration, such as a procedure that is part of a web service application. The code that adds a description must be added before the report request.

Syntax: How to Add a Description of the Procedure

```
-<description>text</description>
```

where:

text
Is the description of the procedure. If the description requires multiple lines, you must repeat the <description> and </description> tags.

Example: Adding a Description of the Procedure

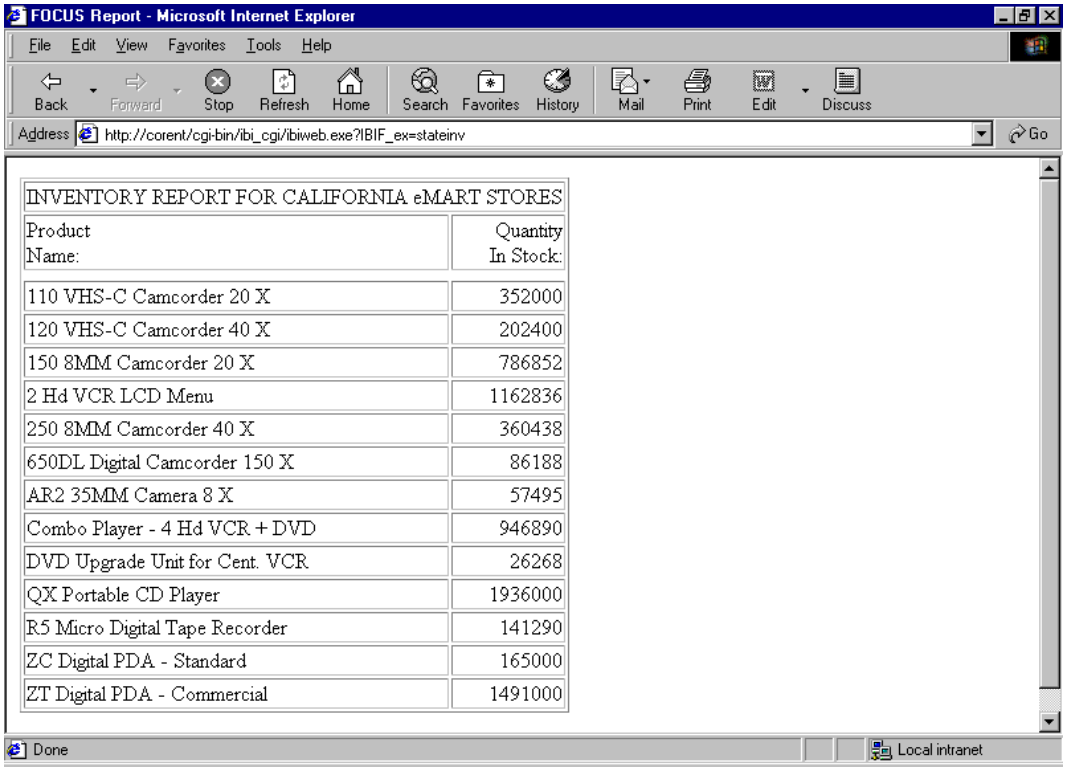
The following provides a description of the procedure. Note that this tag is only visible in the returned XML code.

```
-<description>This procedure reports on Inventory </description>
-<description>by state, storename and product name. </description>
TABLE FILE CENTORD
SUM QTY_IN_STOCK BY STATE BY STORENAME BY PRODNAME
ON TABLE SUBHEAD
"Inventory Report"
WHERE STATE EQ '&STATE'
WHERE STORENAME EQ '&STORENAME'
WHERE PRODNAME EQ '&PRODNAME'
END
```

Displaying a Report on the Default WebFOCUS Page

The simplest way to display a report is to use the default WebFOCUS page. You do not need to design the page on which the report will appear. WebFOCUS automatically generates an HTML page that contains the necessary tagging. The report output is formatted on the page using the HTML <TABLE tag.

The following image shows a report displayed on the default WebFOCUS page.



INVENTORY REPORT FOR CALIFORNIA eMART STORES	
Product Name:	Quantity In Stock:
110 VHS-C Camcorder 20 X	352000
120 VHS-C Camcorder 40 X	202400
150 8MM Camcorder 20 X	786852
2 Hd VCR LCD Menu	1162836
250 8MM Camcorder 40 X	360438
650DL Digital Camcorder 150 X	86188
AR2 35MM Camera 8 X	57495
Combo Player - 4 Hd VCR + DVD	946890
DVD Upgrade Unit for Cent. VCR	26268
QX Portable CD Player	1936000
R5 Micro Digital Tape Recorder	141290
ZC Digital PDA - Standard	165000
ZT Digital PDA - Commercial	1491000

If you save the output from a dynamic report using the command ON TABLE HOLD FORMAT HTML in a procedure, you create a *static report*, which can be called and displayed.

In applications where users do not require real-time information, static reports can substantially reduce overhead on the system. Reports can then be refreshed during off-peak times.

Procedure: How to Display a Dynamic Report on the Default WebFOCUS Page

1. Create a procedure that generates a dynamic report.
2. Create a launch page from which a user can run the report.
3. Run the report from the launch page. WebFOCUS returns the report to the browser as a complete HTML file. You can view it using browser options, for example, *View/Source* in Microsoft Internet Explorer.

Example: Displaying a Dynamic Report on the Default WebFOCUS Page

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create a procedure named STATEINV, which generates a dynamic report. This report must be accessible to the WebFOCUS Reporting Server.

   Procedure: STATEINV.FEX

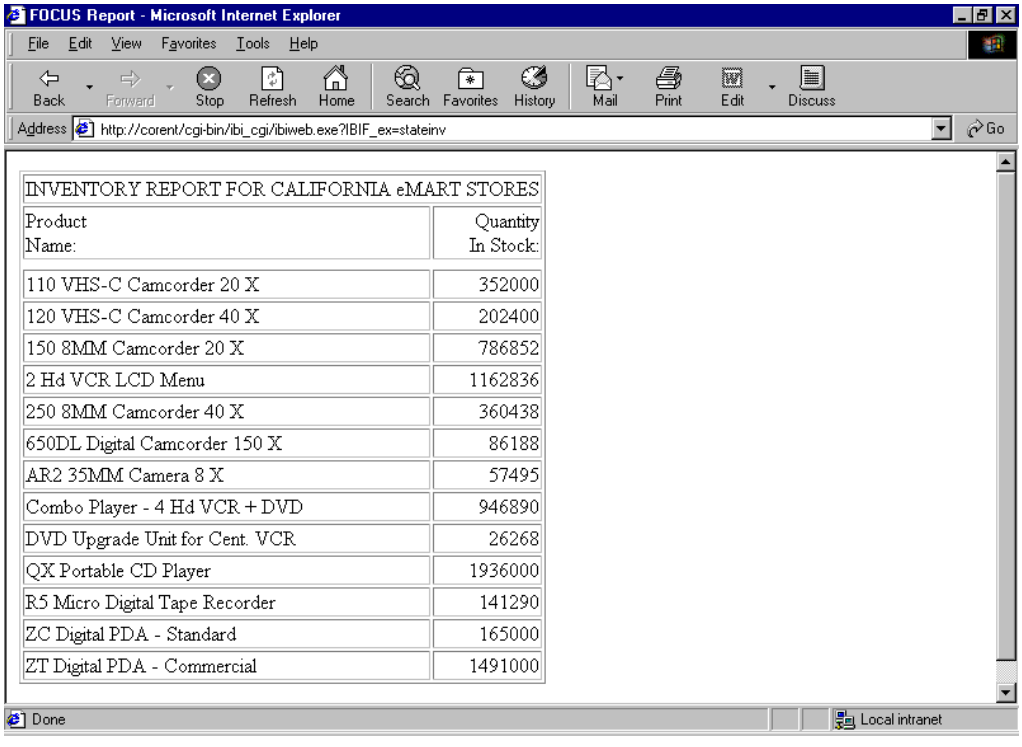
   ```
   SET PAGE-NUM = OFF
   TABLE FILE CENTORD
   SUM QTY_IN_STOCK BY PRODNAME
   ON TABLE SUBHEAD
   "INVENTORY REPORT FOR CALIFORNIA eMART STORES"
   WHERE STATE EQ 'CA'
   WHERE STORENAME EQ 'eMart'
   END
   ```

2. Create a launch page from which a user can run the report. The following is a simple launch page named RUNDYNAM. It uses the WebFOCUS Servlet:

   Launch Page: RUNDYNAM.HTM

   ```
   <HTML>
   <BODY>
   <A HREF="http://server1/servlet/WFServlet?IBIF_ex=stateinv">Click here
   to run an inventory report for eMart in California.</A>
   </BODY>
   </HTML>
   ```

3. Run the launch page, and click the link. The dynamic report displays in the browser on the default WebFOCUS page:



INVENTORY REPORT FOR CALIFORNIA eMART STORES	
Product Name:	Quantity In Stock:
110 VHS-C Camcorder 20 X	352000
120 VHS-C Camcorder 40 X	202400
150 8MM Camcorder 20 X	786852
2 Hd VCR LCD Menu	1162836
250 8MM Camcorder 40 X	360438
650DL Digital Camcorder 150 X	86188
AR2 35MM Camera 8 X	57495
Combo Player - 4 Hd VCR + DVD	946890
DVD Upgrade Unit for Cent. VCR	26268
QX Portable CD Player	1936000
R5 Micro Digital Tape Recorder	141290
ZC Digital PDA - Standard	165000
ZT Digital PDA - Commercial	1491000

Procedure: How to Display a Static Report on the Default WebFOCUS Page

1. Create a procedure:

   a. Identify the HTML file that will contain the report output and save it to the designated location. For example, on Windows, use a FILEDEF command. See *Defining and Allocating WebFOCUS Files* on page 651 for details on platform-specific commands.

      The web server must be able to locate the file for browser display, so specify a location under a web server alias.

   b. Include the following command

      ```
      ON TABLE HOLD FORMAT HTML AS report
      ```

      where:

      report is the name of a virtual file that contains the report output. The name can be from 1 to 8 characters. Do not include an extension. This file is not saved to disk.

2. Run the procedure to create the report as an HTML file. There are several ways to do this, including:

   - Call the WebFOCUS Client. For an example, see *Displaying a Static Report on the Default WebFOCUS Page* on page 229.
   - Schedule the report to run with the WebFOCUS ReportCaster. See your ReportCaster documentation for details.
   - In WebFOCUS, access the following default directory

     Windows: *install_drive*:\ibi\srv82\wfs\bin

     UNIX: /ibi/srv82/wfs/bin

     z/OS: /ibi/srv82/wfs/bin

     and issue the following command:

     ```
     edastart -t
     ```

     Execute the procedure from the WebFOCUS prompt >>:

     ```
     ex procedure_name
     ```

     To return to the WebFOCUS directory, type:

     ```
     fin
     ```

3. Create a launch page with a hyperlink that links to the report.

Example: **Displaying a Static Report on the Default WebFOCUS Page**

This example runs on WebFOCUS for Windows. If you are using another platform, substitute the appropriate platform-specific command for the FILEDEF command. For details, see *Defining and Allocating WebFOCUS Files* on page 651.

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create a procedure that saves a report as an HTML file. The following sample procedure is named STATDATA. The letters on the left correspond to the notes explaining the code.

   Procedure: STATDATA.FEX

```
a. FILEDEF STATRPT DISK APP/STATRPT.HTM
   TABLE FILE GGORDER
   HEADING
   "Orders Summary"
   SUM QUANTITY AS 'Units Ordered'
   BY ORDER_DATE AS 'Order Date'
   BY PRODUCT_DESCRIPTION
   WHERE ORDER_DATE EQ '01/01/96'
b. ON TABLE HOLD FORMAT HTML AS STATRPT
   END
```

a. This command allocates the file STATRPT and saves it to a designated directory.

b. This command saves the report output as an HTML file named STATRPT.HTM.

2. Run the procedure STATDATA to create and save STATRPT. One way to do this is to call the WebFOCUS Client.

   The following simple HTML page uses the Servlet to run STATDATA. An administrator can use this kind of page to create the static reports that users will access.

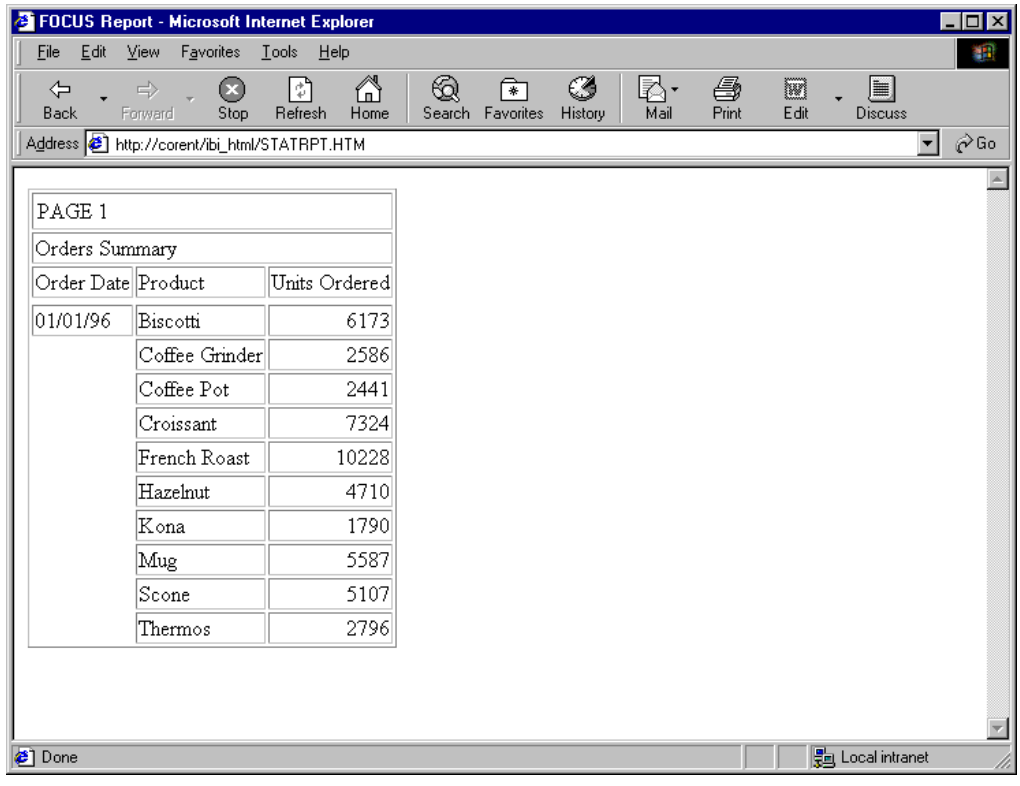
```
<HTML>
<BODY>
<A HREF="/ibi_apps/WFServlet?IBIF_ex=statdata">Click here
to run STATDATA and create a static report stored as STATRPT.HTM.</A>
</BODY>
</HTML>
```

   WebFOCUS displays a message indicating that an HTML file was saved.

3. Create a launch page from which a user can run the report. The following is a simple launch page named STATIC:

```
<HTML>
<BODY>
<A HREF="STATRPT.HTM">Click here to view the Orders Summary report</A>
</BODY>
</HTML>
```

4. Run the launch page, and click the link. The static report displays in the browser on the default WebFOCUS page on which it was saved:



Designing an HTML Page for Report Display

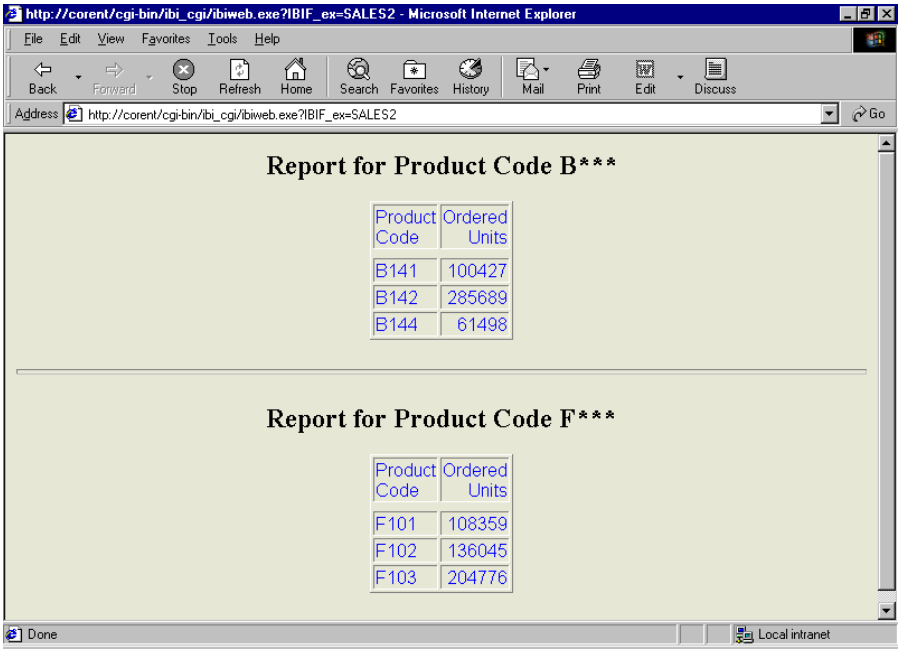
Use the Dialogue Manager command -HTMLFORM to design the HTML page on which a report displays. You can control page style and format, and imaginatively enhance a report. A custom HTML page can include HTML elements such as frames, tables, and graphic images. It must be a complete HTML page with all required HTML tags. It also contains a special HTML comment that indicates where to insert the WebFOCUS report output.

You can do one of the following:

- Call an external, existing HTML file that defines the display page. This method uses the command -HTMLFORM *filename* in a procedure. The procedure is in one file, and the HTML page is in another file.
- Create a custom HTML page within the procedure, using the commands -HTMLFORM BEGIN and -HTMLFORM END. In this topic, this type of page is called *inline*.

You can display one or many reports on a page. A report can be dynamic or static.

The following image shows multiple reports displayed on a custom HTML page.



For more information on -HTMLFORM and its capabilities, see *Enhancing an HTML Webpage With a Procedure* on page 407.

Reference: Usage Considerations for -HTMLFORM

- Custom styling code specified within -HTMLFORM that creates an HTML page on which to display an HTML report can interfere with multi-drill, HFREEZE, Accordion, BYTOC, and On-demand Paging JavaScript-based reporting features. App Studio HTML Canvas is the recommended tool for creating an HTML page that displays one or more WebFOCUS interactive JavaScript-based features within a frame that requires specific styling, such as sizing of the area in which the report displays or scroll bar positioning.

Procedure: How to Display a Dynamic Report on an Existing HTML Page

1. Create the HTML page that will incorporate the report output. Use the following comment to indicate where the output will display

   ```
   !IBI.FIL.report;
   ```

   or

```
<!--WEBFOCUS TABLE report-->
```

where:

report

Is the name of a virtual file that holds the report output. That file is generated in step 2.

Note: The HTML comment line must be closed with the —> characters or a single > character and should not have any other HTML tags within it.

The Reporting Server must be able to locate the page using APP PATH or EDAPATH.

When the page displays, the comment is replaced by report output.

2. Create a procedure:
   a. Include the following command

      ```
      ON TABLE HOLD FORMAT HTMTABLE AS report
      ```

      where:

      report is the name of a virtual file that contains the report output. The name can be from 1 to 8 characters. Do not include an extension. This file is not saved to disk.

   b. At the end of the procedure, include the Dialogue Manager command

      ```
      -HTMLFORM filename
      ```

      where:

      filename is the name of the HTML page that will incorporate the report output. You create this page in step 1.

3. Create a launch page from which a user can run the report. WebFOCUS incorporates the report output into the HTML page you designed, and returns a complete HTML file to the browser.

Example: Displaying a Dynamic Report on an Existing HTML Page

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create an HTML page named RPTPG.HTM. The Reporting Server must be able to locate the page using APP PATH or EDAPATH. See *WebFOCUS Application Logic* on page 27 for details on search paths.

   The letters on the left correspond to the notes explaining the code.

   HTML Page: RPTPG.HTM

```
    <HTML>
    <BODY BGCOLOR="#CCCCCC">
    <FONT FACE="Arial" COLOR="Black">
    <H2>Orders Summary</H2>
a.  !IBI.FIL.ORDERS;
    </BODY>
    </HTML>
```

a. WebFOCUS reads the HTML comment and replaces it with the report output held in ORDERS (step 2).

2. Create a procedure named ORDERS. The letters on the left correspond to the notes explaining the code.

Procedure: ORDERS.FEX

```
   SET PAGE-NUM = OFF
   TABLE FILE GGORDER
   SUM QUANTITY AS 'Units Ordered'
   BY HIGHEST 1 ORDER_DATE AS 'Order Date'
   BY PRODUCT_DESCRIPTION
a. ON TABLE HOLD FORMAT HTMTABLE AS ORDERS
   END
b. -HTMLFORM RPTPG
```

a. This command saves the report output to a virtual file in HTML format. The file is named ORDERS.

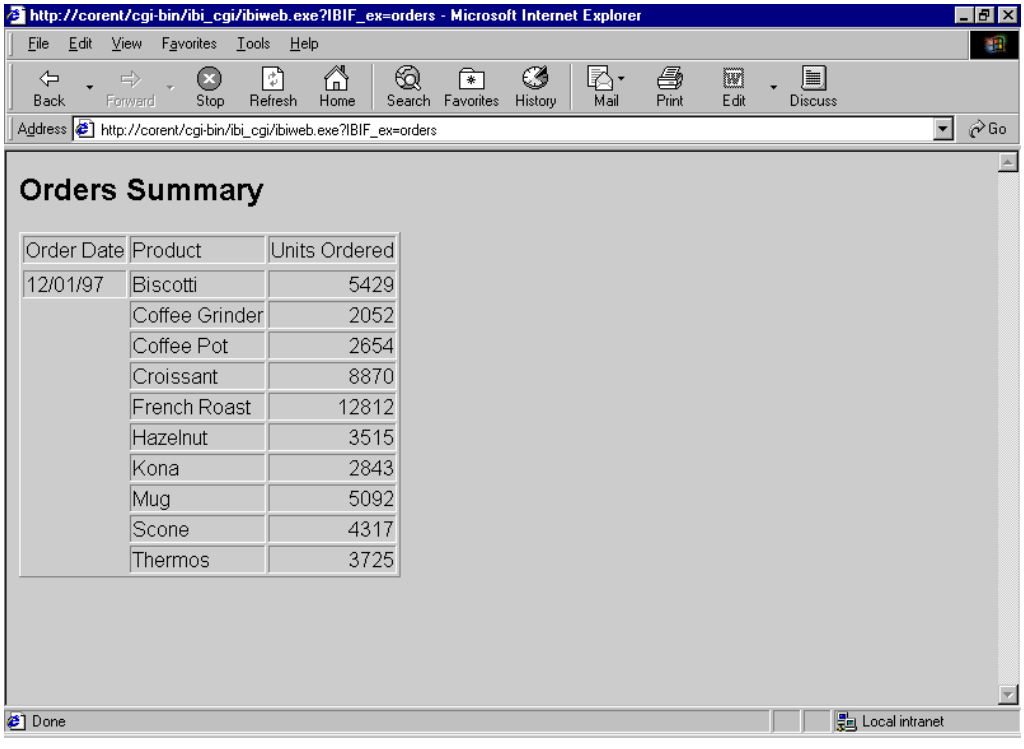
b. This command sends the report output held in ORDERS to the HTML page named RPTPG.HTM (step 1).

3. Create a launch page from which a user can run the report. The following simple launch page is named RUNRPT.HTM and it uses the Servlet.

Launch Page: RUNRPT.HTM

```
<HTML>
<BODY>
<A HREF="/ibi_apps/WFServlet?IBIF_ex=orders">Click here for the Orders
Summary report.</A>
</BODY>
</HTML>
```

4. Run the launch page, and click the link. The dynamic report displays on the custom HTML page.



Syntax: How to Display a Report on an Inline HTML Page

```
-HTMLFORM BEGIN
html_code
html_code
html_code
.
.
.-HTMLFORM END
```

where:

html_code

Is a line of standard HTML code. It cannot exceed 80 characters.

Example: Displaying a Dynamic Report on an Inline HTML Page

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create a procedure named ORDERS2. The letters on the left correspond to the notes explaining the code.

   Procedure: ORDERS2

```
    SET PAGE-NUM = OFF
    TABLE FILE GGORDER
    SUM QUANTITY AS 'Units Ordered'
    BY HIGHEST 1 ORDER_DATE AS 'Order Date'
    BY PRODUCT_DESCRIPTION
a. ON TABLE HOLD FORMAT HTMTABLE AS ORDERS2
    END
b. -HTMLFORM BEGIN
   <HTML><HEAD><STYLE>TD {FONT-FAMILY: ARIAL; COLOR:
    BLUE;}</STYLE></HEAD>
    <BODY BGCOLOR="FAEBD7"><DIV align="left">
c. !IBI.FIL.ORDERS2;
    </DIV></BODY></HTML>
b. -HTMLFORM END
```

   a. This command saves the report output to a virtual file in HTML format. The file is named ORDERS2.

   b. These commands delimit the inline HTML page that defines style and format. The lines in-between the commands are standard HTML code. Each line is less than 80 characters.

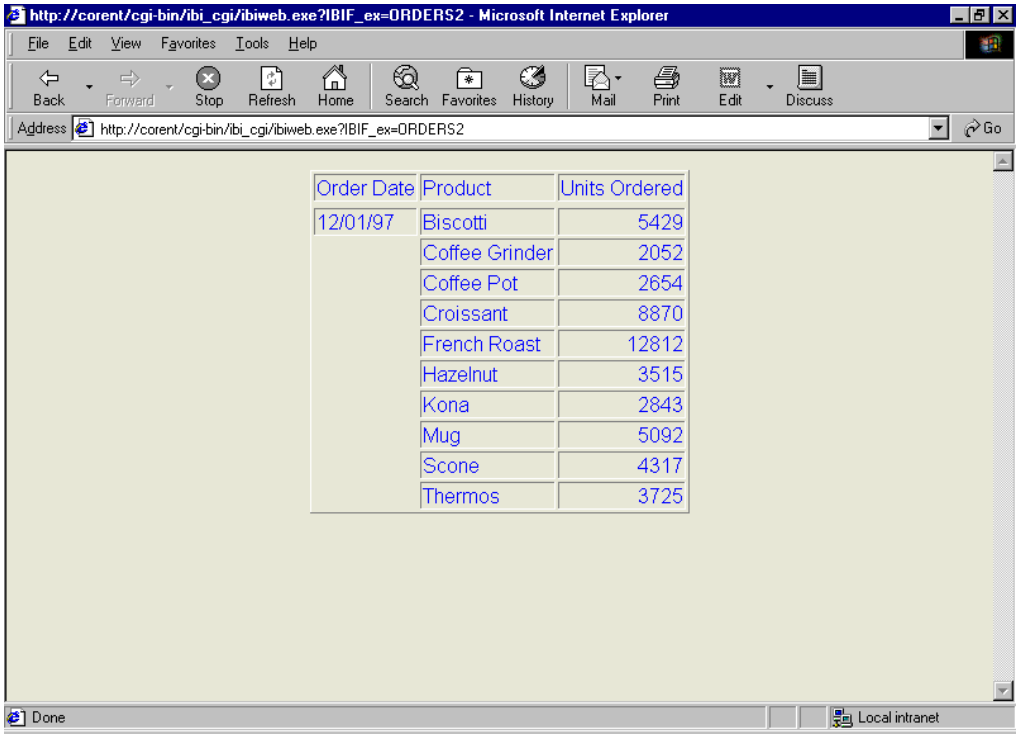
   c. WebFOCUS reads this HTML comment and replaces it with the report output held in ORDERS2.

2. Create a launch page from which a user can run the report. The following simple launch page is named RUNRPT.HTM and it uses the Servlet.

   Launch Page: RUNRPT.HTM

```
<HTML>
<BODY>
<A HREF="/ibi_apps/WFServlet?IBIF_ex=orders2">Click here for the
Orders Summary report.</A>
</BODY>
</HTML>
```

3. Run the launch page, and click the link. The dynamic report displays on the custom inline HTML page.



Example: Displaying Two Dynamic Reports on an Existing HTML Page

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create an HTML page named FIRST.HTM. The Reporting Server must be able to locate the page using APP PATH or EDAPATH.

   The letters on the left correspond to the notes explaining the code.

HTML Page: FIRST.HTM

```
     <HTML>
     <BODY BGCOLOR="#CCCCCC">
a.   <H2>Report for Product Code B***</H2>
b.   <!--WEBFOCUS TABLE UPPER-->
     <HR SIZE=5>
a.   <H2>Report for Product Code F***</H2>
b.   <!--WEBFOCUS TABLE LOWER-->
     </BODY>
     </HTML>
```

a. This text identifies the two reports on the HTML page.

b. WebFOCUS reads these HTML comments. UPPER and LOWER are the names assigned to the virtual files in step 2. On the HTML page, WebFOCUS replaces the comments with the designated report output.

   Note: The HTML comment line must be closed with the —> characters or a single > character and should not have any other HTML tags within it.

2. Create a procedure named TWOSALES. The letters on the left correspond to the notes explaining the code.

   Procedure: TWOSALES.FEX

```
   SET PAGE-NUM = OFF
   TABLE FILE GGORDER
   SUM QUANTITY BY PCD
   IF PCD EQ 'B$$$'
a. ON TABLE HOLD FORMAT HTMTABLE AS UPPER
   END
   TABLE FILE GGORDER
   SUM QUANTITY BY PCD
   IF PCD EQ 'F$$$'
a. ON TABLE HOLD FORMAT HTMTABLE AS LOWER
   END
b. -RUN
c. -HTMLFORM FIRST
```

   a. These commands generate and save the report output from each request to virtual files in HTML format. The files are named UPPER and LOWER.

   b. This command runs the report requests.

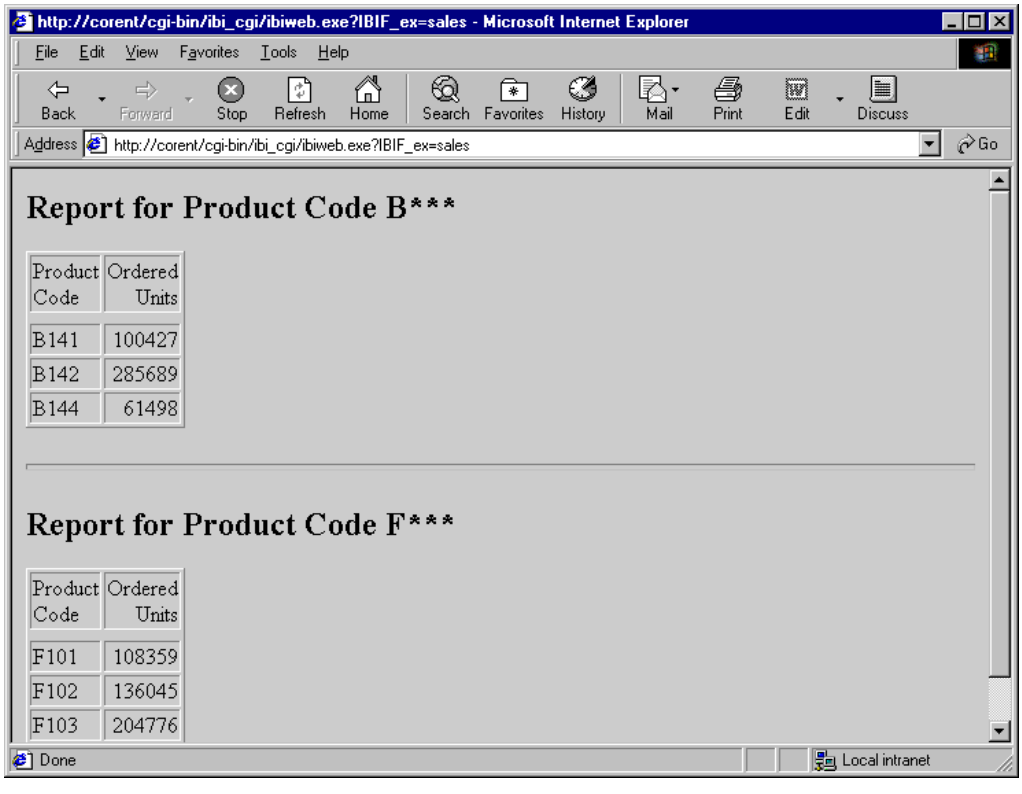
   c. This command sends the report output to the HTML page named FIRST.HTM. You create this page in step 1.

3. Create a launch page from which a user runs the reports. The following simple launch page is called MULTIRPT.HTM and it uses the Servlet.

Launch Page: MULTIRPT.HTM

```
<HTML>
<BODY>
<A HREF="/ibi_apps/WFServlet?IBIF_ex=twosales">Click here for sales
reports for two product codes.</A>
</BODY>
</HTML>
```

4. Run the launch page, and click the link. The two dynamic reports display on the custom HTML page.



Example: Displaying Two Dynamic Reports on an Inline HTML Page

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

1. Create a procedure named SALES2. The letters on the left correspond to the notes explaining the code.

Procedure: SALES2.FEX

```
     SET PAGE-NUM = OFF
     TABLE FILE GGORDER
     SUM QUANTITY BY PCD
     IF PCD EQ 'B$$$'
a.   ON TABLE HOLD FORMAT HTMTABLE AS UPPER
     END
     TABLE FILE GGORDER
     SUM QUANTITY BY PCD
     IF PCD EQ 'F$$$'
a.   ON TABLE HOLD FORMAT HTMTABLE AS LOWER
     END
b.  -RUN
c.  -HTMLFORM BEGIN
    <HTML><HEAD><STYLE>TD {FONT-FAMILY: ARIAL; COLOR:
     BLUE;}</STYLE></HEAD>
    <BODY BGCOLOR="FAEBD7"><DIV align="left">
    <H2>Report for Product Code B***</H2>
d.  <!--WEBFOCUS TABLE UPPER-->
    <HR SIZE=5>
    <H2>Report for Product Code F***</H2>
d.  <!--WEBFOCUS TABLE LOWER-->
    </DIV></BODY></HTML>
c.  -HTMLFORM END
```

a. These commands generate and save the report output from each request to virtual files in HTML format. The files are named UPPER and LOWER.

b. This command runs the report requests.

c. These commands delimit the inline HTML page that defines style and format. The lines in-between the commands are standard HTML code. Each line is less than 80 characters.

d. WebFOCUS reads these HTML comments. UPPER and LOWER are the names assigned to the virtual files. On the HTML page, WebFOCUS replaces the comments with the designated report output.

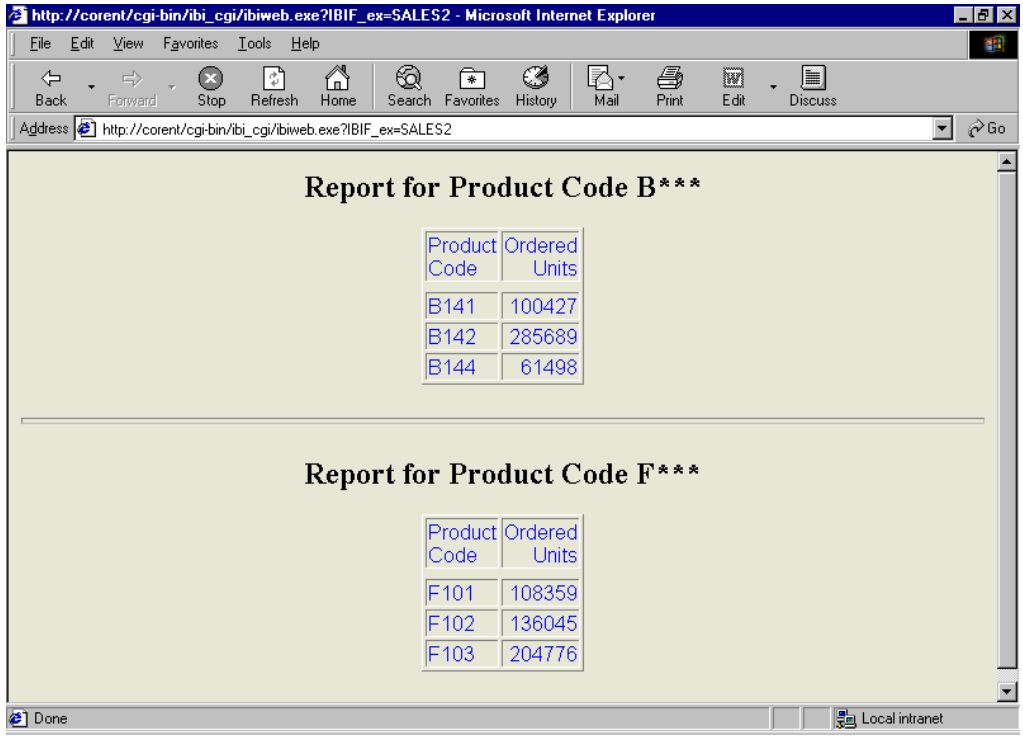
   Note: The HTML comment line must be closed with the —> characters or a single > character and should not have any other HTML tags within it.

2. Create a launch page from which a user can run the reports. The following simple launch page is called MULTIRPT.HTM and it uses the Servlet.

Launch Page: MULTIRPT.HTM

```
<HTML>
<BODY>
<A HREF="/ibi_apps/WFServlet?IBIF_ex=sales2">Click here for sales
reports for two product codes.</A>
</BODY>
</HTML>
```

3. Run the launch page, and click the link. The two dynamic reports display on the custom inline HTML page.



Procedure: How to Display a Static Report on an Existing HTML Page

To create a procedure:

1. Allocate the HTML file that will combine the report output and the custom HTML page.

   For example, on Windows, the command is

   ```
   FILEDEF htmlpage DISK app/htmlpage.HTM
   ```

   where:

   htmlpage

   Is the name of the combined report output and custom HTML page file. The file name can be from 1 to 8 characters.

   app\

   Is the application directory in which to save the file.

For other platform-specific commands, see *Defining and Allocating WebFOCUS Files* on page 651.

2. Include the following command

   ```
   ON TABLE HOLD FORMAT HTMTABLE AS report
   ```

   where:

   report

   Is the name of a virtual file that contains the report output. The name can be from 1 to 8 characters. Do not include an extension. This file is not saved to disk.

3. At the end of the procedure, include the Dialogue Manager command

   ```
   -HTMLFORM filename SAVE AS htmlpage
   ```

   where:

   filename

   Is the name of the custom HTML page that will incorporate the report output. You create this page in step 2. The Reporting Server must be able to locate the page using APP PATH or EDAPATH.

   htmlpage

   Is the name of the HTML file that will combine the report output and the custom HTML page. The web server must be able to locate this file.

4. Create the custom HTML page. Add the following comment to indicate where the output will display

   ```
   !IBI.FIL.report;
   ```

   or

   ```
   <!--WEBFOCUS TABLE report-->
   ```

   where:

   report

   Is the name of the virtual file that holds the report output. That file was created in step 1.

   Note: The HTML comment line must be closed with the —> characters or a single > character and should not have any other HTML tags within it.

   The Reporting Server must be able to locate the page using APP PATH or EDAPATH.

   On the HTML page, WebFOCUS replaces the comment with the report output.

5. Run the procedure from step 1 to create the static report on the custom HTML page.
6. Create a launch page with a hyperlink that links to the report.

Example: Displaying Static Reports on an Existing HTML Page

This example runs on WebFOCUS for Windows. If you are using another platform, substitute the appropriate platform-specific command for the FILEDEF command. For more information, see *Defining and Allocating WebFOCUS Files* on page 651.

Note: For information on where to store the files created in this example, see *Defining and Allocating WebFOCUS Files* on page 651.

Displaying One Static Report on an Existing HTML Webpage

1. Create a procedure named CONTACTS. The letters on the left correspond to the notes explaining the code.

   Procedure: CONTACTS.FEX

```
a. FILEDEF WEBPAGE DISK APPDIR/WEBPAGE.HTM
   SET PAGE-NUM = OFF
   TABLE FILE GGSTORES
   PRINT STORE_NAME ADDRESS1
   BY CITY
b. ON TABLE HOLD FORMAT HTMTABLE AS AREPORT
   END
c. -HTMLFORM DATAOUT SAVE AS WEBPAGE
```

   a. This command allocates the file WEBPAGE.HTM and saves it to a directory accessible to the web server. It will combine the report output and the custom HTML page.

   b. This command saves the report output to a virtual file in HTML format. The file is named AREPORT. It is not saved on disk.

   c. This command merges the report output with the custom HTML page named DATAOUT, which you create in step 2. It also specifies a second HTML file, WEBPAGE, in which the combined HTML from the report output and the custom page are saved.

2. Create a custom HTML page named DATAOUT.HTM. The Reporting Server must be able to locate the page using APP PATH or EDAPATH.

   HTML Page: DATAOUT.HTM

```
   <HTML>
   <BODY BGCOLOR="#CCCCCC">
a. <H2>Store Contacts</H2>
b. <!--WEBFOCUS TABLE AREPORT-->
   </BODY>
   </HTML>
```

   a. This text identifies the report.

b. WebFOCUS reads the HTML comment and replaces it with the report output held in AREPORT.

Note: The HTML comment line must be closed with the –> characters or a single > character and should not have any other HTML tags within it.

3. Run the procedure CONTACTS to create the static report on the custom HTML page. One way to do this is to call the WebFOCUS Client:

```
<HTML>
<BODY>
<A HREF="/ibi_apps/WFServlet?IBIF_ex=contacts">Click here
to run CONTACTS and create a static report.</A>
</BODY>
</HTML>
```

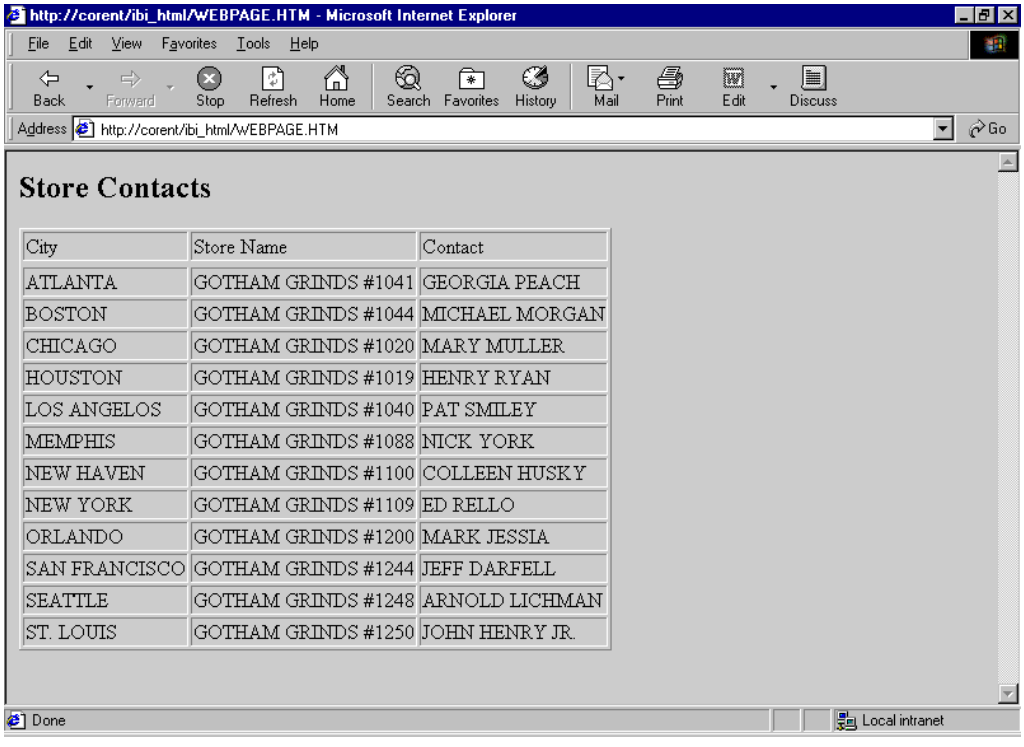
The message Done in your browser indicates that file was created.

4. Create a launch page from which a user can run the report. This simple launch page is named ONESTAT.HTM:

Launch Page: ONESTAT.HTM

```
<HTML>
<BODY>
<A HREF="WEBPAGE.HTM">Click here to view store contacts.</A>
</BODY>
</HTML>
```

5. Run the launch page, and click the link. The static report displays on the custom HTML page.



Displaying Two Static Reports on an Existing HTML Webpage

1. Create a procedure named DEMOG. The letters on the left correspond to the notes explaining the code.

   Procedure: DEMOG.FEX

```
a. FILEDEF RPTPAGE DISK APPDIR/RPTPAGE.HTM
   SET PAGE-NUM = OFF
   TABLE FILE GGDEMOG
   SUM HH AS 'Number of,Households' AVGHHSZ98 AS 'Avg.,Size'
   MEDHHI98 AS 'Avg.,Income'
   BY ST
   WHERE ST EQ 'CA'
b. ON TABLE HOLD FORMAT HTMTABLE AS UPPER
   END
   TABLE FILE GGDEMOG
   SUM HH AS 'Number of,Households' AVGHHSZ98 AS 'Avg.,Size'
   MEDHHI98 AS 'Avg.,Income'
   BY ST
   WHERE ST EQ 'NY'
c. ON TABLE HOLD FORMAT HTMTABLE AS LOWER
   END
   -RUN
d. -HTMLFORM RPTOUT SAVE AS RPTPAGE
```

a. This command allocates the file RPTPAGE.HTM and saves it to a directory accessible to the web server. It will combine the report output and the custom HTML page.

b. This command saves the output from the first request to a virtual file named UPPER.

c. This command saves the output from the second request to a virtual file named LOWER.

d. This command merges the report output with the custom HTML page named RPTOUT, which you create in step 2. It also specifies a second HTML file, RPTPAGE, in which the combined HTML from the report output and the custom page are saved.

2. Create a custom HTML page named RPTOUT.HTM. The Reporting Server must be able to locate the page using APP PATH or EDAPATH.

HTML Page: RPTOUT.HTM

```
   <HTML>
   <BODY BGCOLOR="#CCCCCC">
a. <H3>Demographic Report for California</H3>
b. <!--WEBFOCUS TABLE UPPER-->
   <HR SIZE=5>
a. <H3>Demographic Report for New York</H3>
b. <!--WEBFOCUS TABLE LOWER-->
   </BODY>
   </HTML>
```

a. This text identifies the two reports.

b. WebFOCUS reads these HTML comments and replaces UPPER and LOWER with the report output.

Note: The HTML comment line must be closed with the –-> characters or a single > character and should not have any other HTML tags within it.

3. Run the procedure DEMOG to create the static reports on the custom HTML page. One way to do this is to call the WebFOCUS Client:

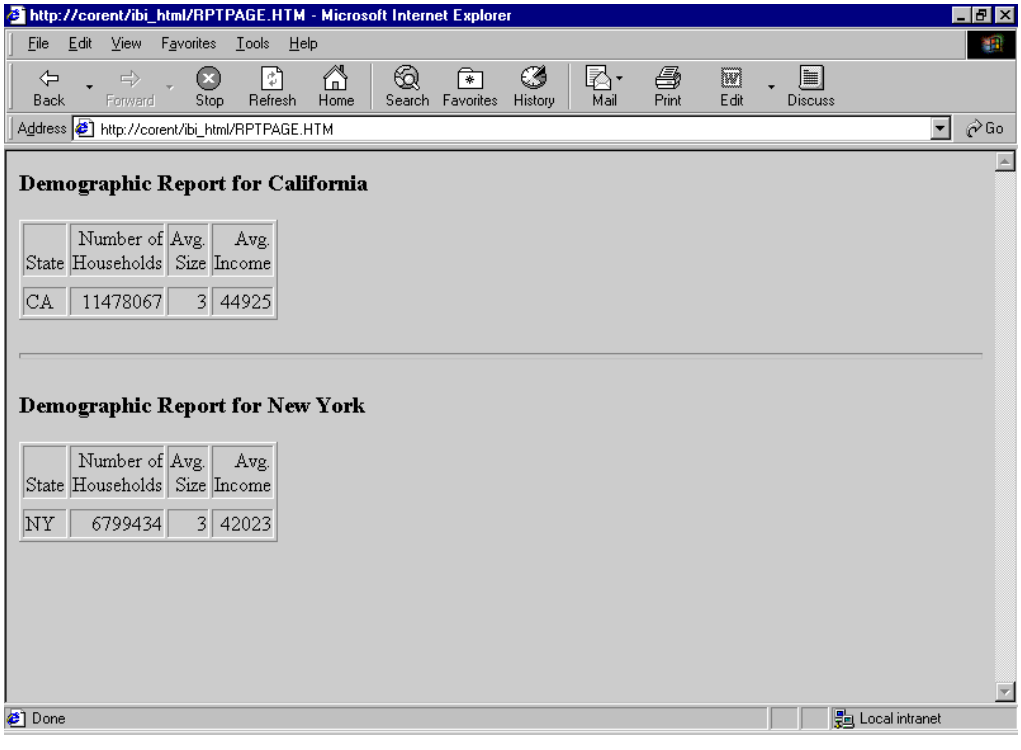
```
<HTML>
<BODY>
<A HREF="/ibi_apps/WFServlet?IBIF_ex=demog">Click here
to run DEMOG and create two static reports.</A>
</BODY>
</HTML>
```

4. Create a launch page from which a user can run the reports. This simple launch page is called TWOSTATS.HTM:

   Launch Page: TWOSTATS.HTM

```
<HTML>
<BODY>
<A HREF="RPTPAGE.HTM">Click here to view the demographic reports.</A>
</BODY>
</HTML>
```

5. Run the launch page, and click the link. The static reports display on the custom HTML page:



Displaying an Accordion By Row Report Using HOLD FORMAT HTMTABLE and -HTMLFORM

The HOLD FORMAT HTMTABLE command in conjunction with -HTMLFORM functionality supports the display of Accordion By Row (SET EXPANDBYROW) reports to create an HTML page. Accordion by Column reports (SET EXPANDABLE) are not supported with -HTMLFORM.

Syntax: How to Display an Accordion Report Using HOLD FORMAT HTMLTABLE and -HTMLFORM

You can display one or more Accordion By Row reports on a custom HTML page by creating the report with the SET EXPANDBYROW command, specifying HOLD FORMAT HTMTABLE, and then using the Dialogue Manager command -HTMLFORM.

Include the following commands in the procedure:

```
ON TABLE SET EXPANDBYROW ON
ON TABLE HOLD FORMAT HTMLTABLE AS report
```

where:

report

Is the 1-character to 8-character name of a virtual file that contains the report output.

Running this report creates an output file that contains only the report data. In order to display the output as an HTML Accordion By Row report, the following JavaScript code must be included in the HTML by using the following syntax and placement:

`IBI.OBJ.EXPBYROWJS`

Is the JavaScript for HTML Accordion report functionality. Must be coded in the -HTMLFORM after the <HTML tag> and before the <BODY> tag.

`<div id="IBI_popupHere"></div>`

Is required prior to display of the first Accordion report (using !IBI.FIL.*filename*) for display of pop-up column title description.

`!IBI.OBJ.IBIGBLONLOAD`

Is the JavaScript that requests the load of the global environment setup for WebFOCUS HTML reports. Must be coded within the <BODY> tag before the comment line (! IBI.FIL.*report*) that indicates where to display the first HTML report as:

```
<BODY ONLOAD='!IBI.OBJ.IBIGBLONLOAD';>
```

`!IBI.OBJ.IBIGBLJS`

Is the JavaScript for global environment setup for WebFOCUS HTML reports. Must be coded after the comment line (!IBI.FIL.*report*) that indicates where to display the last HTML report, and before the closing </HTML> tag.

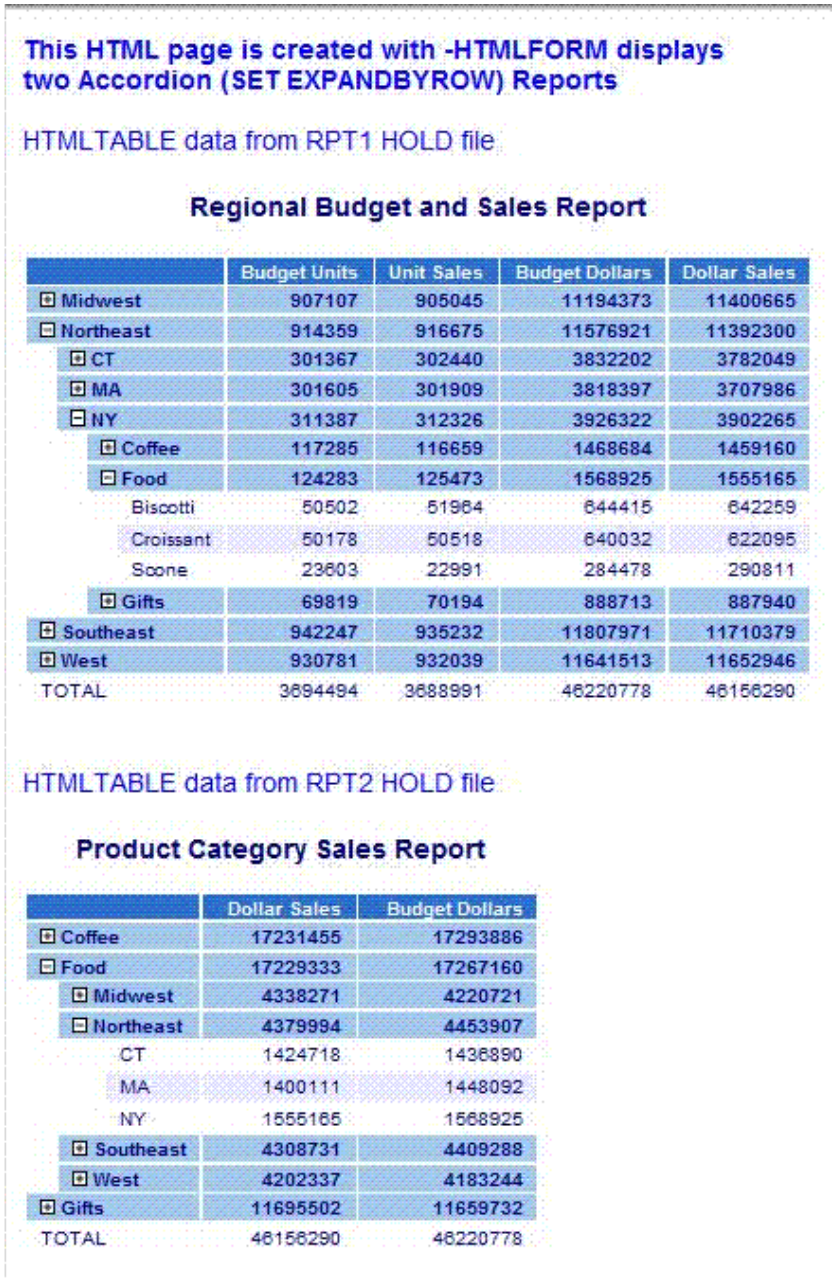
Example: Displaying Two HTML Accordion Reports on an HTML Webpage Created by -HTMLFORM

The following request contains two Accordion By Row reports, RPT1 and RPT2, saved as format HTMTABLE. The -HTMLFORM block contains all of the JavaScript code to display them on an HTML page.

```
TABLE FILE GGSALES
HEADING
"Regional Budget and Sales Report"
" "
SUM BUDUNITS UNITS BUDDOLLARS DOLLARS
BY REGION
BY ST
BY CATEGORY
BY PRODUCT
ON TABLE HOLD AS RPT1 FORMAT HTMTABLE
ON TABLE SET EXPANDBYROW ON
ON TABLE SET HTMLCSS ON
ON TABLE SET DROPBLNKLINE ALL
ON TABLE SET STYLE *
TYPE=REPORT,COLOR=NAVY,FONT='ARIAL',SIZE=9,GRID=OFF,$
TYPE=HEADING,LINE=1,STYLE=BOLD,SIZE=12,JUSTIFY=CENTER,$
TYPE=TITLE,BACKCOLOR=RGB(45 111 205),COLOR=WHITE,STYLE=-UNDERLINE+BOLD, $
TYPE=DATA,BACKCOLOR=(WHITE RGB(235 235 255)),$
TYPE=SUBTOTAL,BACKCOLOR=RGB(163 200 236),STYLE=BOLD,$
END
```

```
TABLE FILE GGSALES
HEADING
"Product Category Sales Report"
" "
SUM GGSALES.SALES01.DOLLARS
GGSALES.SALES01.BUDDOLLARS
BY GGSALES.SALES01.CATEGORY
BY GGSALES.SALES01.REGION
BY GGSALES.SALES01.ST
ON TABLE HOLD AS RPT2 FORMAT HTMTABLE
ON TABLE SET EXPANDBYROW ON
ON TABLE SET HTMLCSS ON
ON TABLE SET PAGE-NUM NOLEAD
ON TABLE SET SQUEEZE ON
ON TABLE SET EMPTYREPORT ON
ON TABLE SET DROPBLNKLINE ALL
ON TABLE SET STYLE *
TYPE=REPORT,COLOR=NAVY,SQUEEZE=ON,FONT='ARIAL',SIZE=9,GRID=OFF,$
TYPE=HEADING,LINe=1,STYLE=BOLD,SIZE=12,JUSTIFY=CENTER,$
TYPE=TITLE,BACKCOLOR=RGB(45 111 205),COLOR=WHITE,STYLE=-UNDERLINE+BOLD, $
TYPE=DATA,BACKCOLOR=(WHITE RGB(235 235 255)),$
TYPE=SUBTOTAL,BACKCOLOR=RGB(163 200 236),STYLE=BOLD,$
END

-HTMLFORM BEGIN
<HTML>!IBI.OBJ.EXPBYROWJS;
<BODY ONLOAD='!IBI.OBJ.IBIGBLONLOAD;'>
<font face="arial" color="blue"><b>This HTML page is created with -HTMLFORM
and displays <br> two Accordion By Row (SET EXPANDBYROW) Reports</b>
<br></br>
</font>
<div id="IBI_popupHere"></div>
!IBI.FIL.RPT1;
<br></br>
!IBI.FIL.RPT2;
!IBI.OBJ.IBIGBLJS;
</BODY>
</HTML>
-HTMLFORM END
```

The output is:



This HTML page is created with -HTMLFORM displays two Accordion (SET EXPANDBYROW) Reports

HTMLTABLE data from RPT1 HOLD file

Regional Budget and Sales Report

	Budget Units	Unit Sales	Budget Dollars	Dollar Sales
⊞ Midwest	907107	905045	11194373	11400665
⊟ Northeast	914359	916675	11576921	11392300
⊞ CT	301367	302440	3832202	3782049
⊞ MA	301605	301909	3818397	3707986
⊟ NY	311387	312326	3926322	3902265
⊞ Coffee	117285	116659	1468684	1459160
⊟ Food	124283	125473	1568925	1555165
Biscotti	50502	51964	644415	642259
Croissant	50178	50518	640032	622095
Scone	23603	22991	284478	290811
⊞ Gifts	69819	70194	888713	887940
⊞ Southeast	942247	935232	11807971	11710379
⊞ West	930781	932039	11641513	11652946
TOTAL	3694494	3688991	46220778	46156290

HTMLTABLE data from RPT2 HOLD file

Product Category Sales Report

	Dollar Sales	Budget Dollars
⊞ Coffee	17231455	17293886
⊟ Food	17229333	17267160
⊞ Midwest	4338271	4220721
⊟ Northeast	4379994	4453907
CT	1424718	1436890
MA	1400111	1448092
NY	1555165	1568925
⊞ Southeast	4308731	4409288
⊞ West	4202337	4183244
⊞ Gifts	11695502	11659732
TOTAL	46156290	46220778

Displaying an HTML HFREEZE Report Using HOLD FORMAT HTMTABLE and -HTMLFORM

You can display one or more HTML HFREEZE reports on a custom HTML page by creating the report with the HFREEZE=ON styling attribute, adding the HOLD FORMAT HTMTABLE command, and then using the Dialogue Manager command -HTMLFORM.

Syntax: How to Display an HTML HFREEZE Report Using HOLD FORMAT HTMTABLE and -HTMLFORM

Include the following command in the report request:

```
ON TABLE HOLD FORMAT HTMLTABLE AS report
```

Also, include the following in the StyleSheet section of the request:

```
ON TABLE SET STYLE *
TYPE=REPORT, HFREEZE=ON,$
```

where:

report

Is the 1 to 8 character name of a virtual file that will contain the report output.

Running this report creates an output file that contains only the report data. In order to display the output as an HTML HFREEZE report, the following JavaScript code must be included in the HTML by using the following syntax and placement:

`!IBI.FIL.`*report*

Is required as a placeholder to indicate the positioning or placement of the report on the HTML page.

`!IBI.OBJ.IBIHEADJS`

Is placed in the <HEAD> tag of the HTML page.

Example: **Displaying Two HTML HFREEZE Reports Using HOLD FORMAT HTMTABLE and -HTMLFORM**

The following FOCEXEC creates and displays two HFREEZE reports in an HTML page created with -HTMLFORM . The first two TABLE requests create the HFREEZE reports (styling code specifies HFREEZE=ON) that are saved to the Reporting Server with ON TABLE HOLD FORMAT HTMLTABLE:

```
-*Example: Displaying Two HTML HFREEZE Reports on an HTML Web Page
-*
TABLE FILE GGSALES
HEADING CENTER
"Regional Sales Report"
SUM DOLLARS BUDDOLLARS
BY CATEGORY
BY PRODUCT
BY REGION
BY ST
BY CITY
ON REGION SUBTOTAL
FOOTING CENTER
"*** Scroll To See All Data ***"
ON TABLE HOLD AS TOPRPT FORMAT HTMTABLE
ON TABLE SET HTMLCSS ON
ON TABLE SET PAGE-NUM NOLEAD
ON TABLE SET SQUEEZE ON
ON TABLE SET EMPTYREPORT ON
ON TABLE SET BYDISPLAY ON
ON TABLE SET STYLE *
TYPE=REPORT,COLOR=NAVY,SQUEEZE=ON,FONT='ARIAL',SIZE=9,GRID=OFF,$
TYPE=REPORT,HFREEZE=ON,SCROLLHEIGHT=2.25, $
TYPE=HEADING,STYLE=BOLD,SIZE=12,JUSTIFY=CENTER,$
TYPE=FOOTING,STYLE=BOLD,SIZE=10,JUSTIFY=CENTER,$
TYPE=TITLE,BACKCOLOR=RGB(45 111 205),COLOR=WHITE,STYLE=-UNDERLINE+BOLD, $
TYPE=DATA,BACKCOLOR=(WHITE RGB(235 235 255)),$
TYPE=SUBTOTAL,BACKCOLOR=RGB(163 200 236),STYLE=BOLD,$
TYPE=SUBTOTAL,BORDER-TOP=LIGHT,BORDER-TOP-COLOR=RGB(111 155 246),$
END
```

```
TABLE FILE GGSALES
HEADING CENTER
"Regional Sales Report"
SUM GGSALES.SALES01.DOLLARS
GGSALES.SALES01.BUDDOLLARS
BY GGSALES.SALES01.CATEGORY
BY GGSALES.SALES01.ST
BY GGSALES.SALES01.REGION
ON REGION SUBTOTAL
FOOTING CENTER
"*** Scroll To See All Data ***"
ON TABLE HOLD AS BOTRPT FORMAT HTMTABLE
ON TABLE SET HTMLCSS ON
ON TABLE SET PAGE-NUM NOLEAD
ON TABLE SET SQUEEZE ON
ON TABLE SET EMPTYREPORT ON
ON TABLE SET BYDISPLAY ON
ON TABLE SET STYLE *
TYPE=REPORT,COLOR=NAVY,SQUEEZE=ON,FONT='ARIAL',SIZE=9,GRID=OFF,$
TYPE=REPORT,HFREEZE=ON,SCROLLHEIGHT=2.25, $
TYPE=HEADING,STYLE=BOLD,SIZE=12,JUSTIFY=CENTER,$
TYPE=FOOTING,STYLE=BOLD,SIZE=10,JUSTIFY=CENTER,$
TYPE=TITLE,BACKCOLOR=RGB(45 111 205),COLOR=WHITE,STYLE=-UNDERLINE+BOLD, $
TYPE=DATA,BACKCOLOR=(WHITE RGB(235 235 255)),$
TYPE=SUBTOTAL,BACKCOLOR=RGB(163 200 236),STYLE=BOLD,$
TYPE=SUBTOTAL,BORDER-TOP=LIGHT,BORDER-TOP-COLOR=RGB(111 155 246),$
END

-HTMLFORM BEGIN
<HTML>
<HEAD>!IBI.OBJ.IBIHEADJS;
</HEAD>
<BODY>
<font face="arial" color="blue" ><b>This HTML page is created with -
HTMLFORM displaying two HFREEZE reports</b>
<br></br>
HTMLTABLE data from TOPRPT HOLD file </font>
!IBI.FIL.TOPRPT;
<font face="arial" color="blue">HTMLTABLE data from BOTRPT HOLD file</font>
!IBI.FIL.BOTRPT;
</BODY>
</HTML>
-HTMLFORM END
```

The generated HTML page with the two reports is:

This HTML page is created with -HTMLFORM displaying two HFREEZE reports

HTMLTABLE data from TOPRPT HOLD file

Regional Sales Report

Category	Product	Region	State	City	Dollar Sales	Budget Dollars
Coffee	Capuccino	Northeast	CT	New Haven	158995	141574
Coffee	Capuccino	Northeast	MA	Boston	174344	192747
Coffee	Capuccino	Northeast	NY	New York	208756	227170
*TOTAL REGION Northeast					542095	561491
Coffee	Capuccino	Southeast	FL	Orlando	317027	285194
Coffee	Capuccino	Southeast	GA	Atlanta	352161	384281
Coffee	Capuccino	Southeast	TN	Memphis	274812	287186
*TOTAL REGION Southeast					944000	956661
Coffee	Capuccino	West	CA	Los Angeles	306468	281701
Coffee	Capuccino	West	CA	San Francisco	279830	294884
Coffee	Capuccino	West	WA	Seattle	309197	300719
*TOTAL REGION West					895495	877304
TOTAL					46156290	46220778

*** Scroll To See All Data ***

HTMLTABLE data from BOTRPT HOLD file

Regional Sales Report

Category	State	Region	Dollar Sales	Budget Dollars
Coffee	CA	West	2937886	2960640
*TOTAL REGION West			2937886	2960640
Coffee	CT	Northeast	1364420	1395283
*TOTAL REGION Northeast			1364420	1395283
Coffee	FL	Southeast	1463453	1408190
*TOTAL REGION Southeast			1463453	1408190
Coffee	GA	Southeast	1576915	1656902
*TOTAL REGION Southeast			1576915	1656902
Coffee	IL	Midwest	1398779	1366264
*TOTAL REGION Midwest			1398779	1366264
Coffee	MA	Northeast	1340437	1388495
*TOTAL REGION Northeast			1340437	1388495
TOTAL			46156290	46220778

*** Scroll To See All Data ***

Displaying an Active Technologies Report Using HOLD format AHTMLTAB and -HTMLFORM

You can display one or more HTML Active Technologies Reports on a custom HTML page using HOLD FORMAT AHTMLTAB and the Dialogue Manager command -HTMLFORM.

Include the following command in the procedure:

```
ON TABLE HOLD FORMAT AHTMLTAB AS report
```

where:

report

Is the name of a virtual file that contains the report output. The name can be from 1 to 8 characters.

Running this report creates an output file that contains only data and parameters used in the HTML active report. In order to display the output as a complete HTML active report, active report JavaScript code must be included in the HTML BODY by using the following syntax:

```
<BODY>
!IBI.OBJ.ACTIVEREPORTJS;
```

Example: Displaying HTML Active Technologies Reports on an HTML Webpage

The HTML code to include active technologies report JavaScript is shown in boldface type in the following example:

```
TABLE FILE GGSALES
SUM
  DOLLARS
  UNITS
BY REGION
BY ST
BY CITY
HEADING
"Regional Sales Summary"
ON TABLE HOLD AS REPORT1 FORMAT AHTMLTAB
END
```

```
-*
TABLE FILE GGSALES
SUM
  DOLLARS
  UNITS
BY CATEGORY
BY PRODUCT
HEADING
"Production Order Summary"
ON TABLE HOLD AS REPORT2 FORMAT AHTMLTAB
END
-*
-HTMLFORM BEGIN
<!DOCTYPE html>
<HTML>
<HEAD>
<TITLE>Displaying HTML Active Technologies Reports on an HTML Web Page</
TITLE>
</HEAD>

<BODY>
!IBI.OBJ.ACTIVEREPORTJS;

<TABLE BORDER='1'>
 <TR>
  <TD valign=top>
!IBI.FIL.REPORT1;
  </TD>
  <TD valign=top>
!IBI.FIL.REPORT2;
  </TD>
 </TR>
</TABLE>
</BODY>
</HTML>
-HTMLFORM END
```

Using HOLD FORMAT XML and -HTMLFORM

Running a report which creates a HOLD FORMAT XML that is called by -HTMLFORM requires additional code to avoid generating an empty HOLD file and an error condition. Add two lines of FILEDEF code, shown in boldface type in the following example:

```
TABLE FILE GGORDER
SUM QUANTITY
ON TABLE HOLD FORMAT XML
END
-RUN
FILEDEF HOLD CLEAR
FILEDEF HOLD DISK HOLD.XML
-HTMLFORM BEGIN
!IBI.FILE.HOLD;
-HTMLFORM END
```

Chapter 4

Enhancing a User Interface

A WebFOCUS StyleSheet enables you to specify a hyperlink to a JavaScript function. You will apply this feature to perform data calculations in a drill-down report.

How you enable report manipulation is an important part of a user interface. This topic describes how to use the WebFOCUS Viewer in an application for navigation of a long report. You will also see how to use an external Cascading Style Sheet to add scrolls bars to report display.

You can implement advanced design features on a launch page. An example is a menu with customized options for an individual user.

This topic describes coding capabilities that extend the functionality and usability of an interface.

Report presentation is not limited to HTML. You will see how to display:

- A report in a format other than HTML, such as PDF (Portable Document Format) or PostScript, using a helper application.
- Multiple reports in different formats from a single launch page.

In this chapter:

- Displaying a Report in a Helper Application
- Controlling Multiple Reports
- Adding JavaScript for Drill-Down Reporting
- Facilitating Report Manipulation
- Using a Cascading Style Sheet to Standardize Display
- Displaying a Previously Run Report
- Passing a User ID From HTML for a Custom Menu

Displaying a Report in a Helper Application

A helper application or plug-in is a desktop program used by a browser to open a file with a format other than standard HTML. An example of a helper application is Adobe® Acrobat® Reader, which opens a PDF file.

When a web server sends a special type of file to a browser, it includes MIME (Multipurpose Internet Mail Extensions) information, indicating the file format. WebFOCUS determines which helper application to run based on MIME type. You must associate the content (MIME) type of a file with a helper application using Windows Explorer.

Procedure: How to Specify a MIME Type in Microsoft Windows Explorer

1. Open Windows Explorer.
2. Select *Folder Options* from the Tools menu.
3. Select the *File Types* tab.
4. Select a file type in the Registered File Types list box, then click the *New Type* button.

   The Add New File Type dialog box opens.
5. Enter the following information.
   - In the Description of Type field, enter a description of your choice.
   - In the Associated Extension field, enter the extension for the file type.
   - In the Content Type (MIME) field, enter the MIME type for the file type and extension.
6. Click *OK* to save your changes.

Reference: Supported MIME Types

WebFOCUS supports the following MIME types.

MIME Type	Application	File Extension
application/postscript	GhostView	.PS
application/x-prn	Printer (using PCL printer driver)	.PRN
application/x-dif	Microsoft Excel® or other spreadsheet	.DIF
application/x-doc	Microsoft Word	.DOC
application/x-xls application/vnd.ms-excl	Microsoft Excel	.XLS, .E97, .WK1, . XHT
application/pdf	Adobe Acrobat Reader	.PDF
text/plain	Microsoft Notepad or other text editor	.WP or .HTS

MIME Type	Application	File Extension
text/html	Native browser format	.HTML
text/htm	Native browser format	.HTM
image/gif	Native browser format	.GIF
image/jpeg	Native browser format	.JPG, .JPEG
XML	Native browser format	.XML

Procedure: How to Display a Report in a Helper Application

1. Create a procedure that includes the command

   ```
   ON TABLE PCHOLD FORMAT fmt
   ```

   where:

   fmt

   Is the file format for the data.

   For details on formats and complete syntax, see the *Creating Reports With WebFOCUS Language* manual.

   Caution: Keep in mind that a valid format for browser display in a helper application requires a supported MIME type.

2. Create a launch page from which a user can run the report.
3. Publish the procedure and launch page using App Studio. When you run the launch page, WebFOCUS returns the data to the browser in the format specified. The browser displays it using the appropriate helper application.

Example: Displaying a Report In Excel and Offering a Choice of Display Formats

The following are examples of displaying a report in an Excel spreadsheet and offering a choice of display formats.

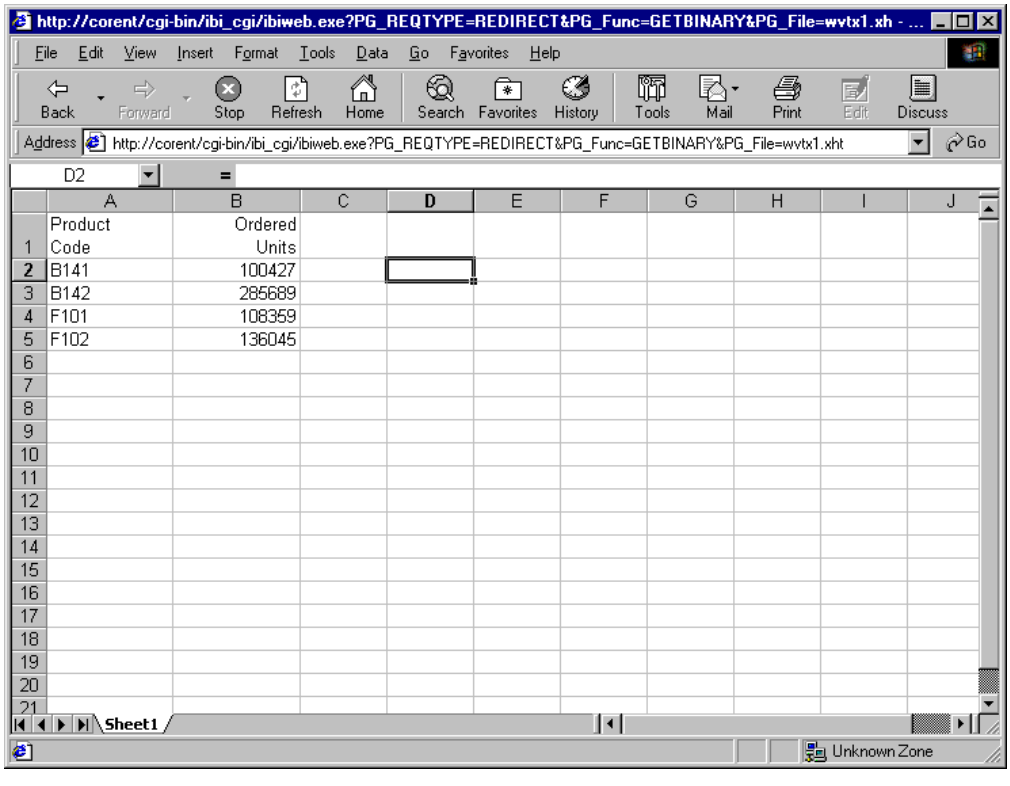
Displaying a Report in an Excel 2000 Spreadsheet

The following is an example of displaying a report in an Excel 2000 spreadsheet.

1. Create a procedure named HELPER, which generates a report on product orders:

```
TABLE FILE GGORDER
SUM QUANTITY BY PCD
WHERE PCD IS 'B141' OR 'B142' OR 'F101' OR 'F102'
ON TABLE PCHOLD FORMAT EXL2K
END
```

2. Create a launch page from which a user can run the report.
3. Publish the procedure and launch page using App Studio.
4. Run the launch page, and click the link. WebFOCUS returns the data to the browser in EXL2K format. The browser calls the helper application, Microsoft Excel 2000, and displays the report:



Offering a Choice of Display Formats

The following is an example of offering a choice of display formats.

1. Create a procedure named QASTATUS, which generates a quality assurance report on a plant (PLANT) of interest, in a format (FMT) of choice:

```
TABLE FILE CENTQA
ON TABLE SET PAGE-NUM OFF
SUM CNT.PROBNUM AS 'Total Number,of Problems'
SUM CNT.PROBNUM AS 'Problem by,Product' BY PLANT NOPRINT BY PRODNAME
WHERE PLANT EQ '&PLANT'
HEADING CENTER
"QA Report for &PLANT"
ON TABLE PCHOLD FORMAT &FMT
END
```

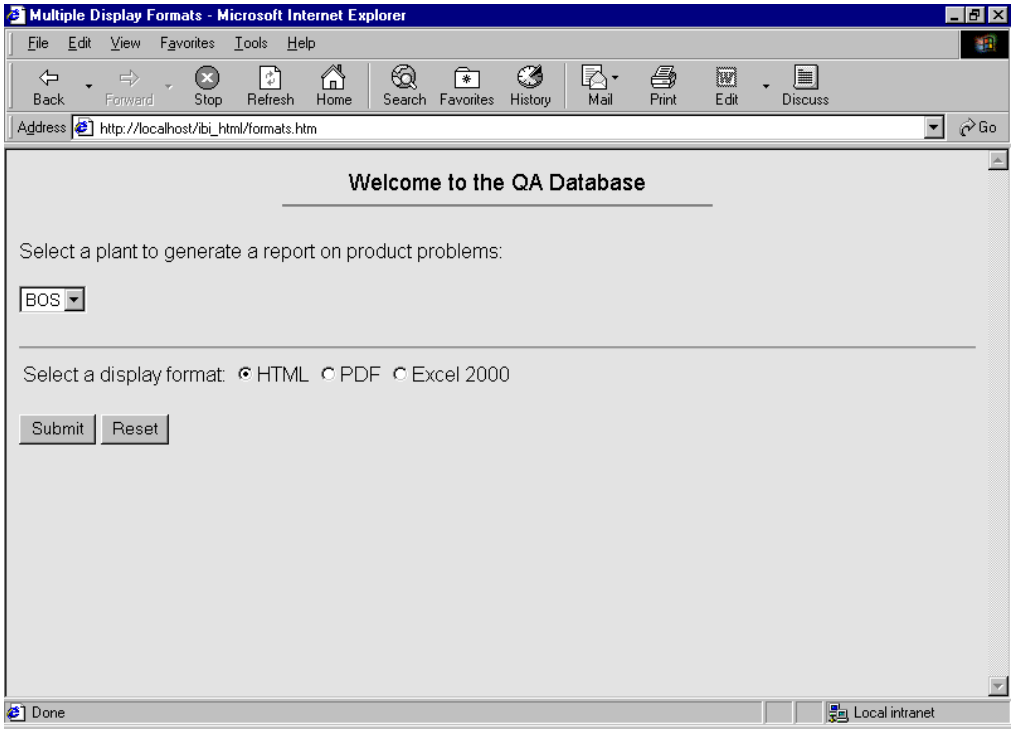
2. Create a launch page from which a user can run the report. The following sample launch page is named FORMATS. The letter on the left corresponds to the note explaining the code.

```
<HTML>
<HEAD>
<TITLE>Multiple Display Formats</TITLE>
</HEAD>
<BODY BGCOLOR="#E3E3E3">
<FONT FACE="Arial">
<CENTER>
<FONT SIZE="+1">Welcome to the QA Database</FONT>
<HR WIDTH="45%" NOSHADE>
</CENTER>
<FORM ACTION="/ibi_apps/WFServlet" METHOD="get">
<INPUT NAME="IBIF_ex" VALUE="qastatus" TYPE="hidden">
<P>
Select a plant to generate a report on product problems:
</P>
<SELECT NAME="PLANT">
<OPTION>BOS
<OPTION>DAL
<OPTION>LA
<OPTION>ORL
<OPTION>SEA
<OPTION>STL
</SELECT>
<P>
<HR>
a. <TABLE CELLPADDING="2">
<TR>
<TD CLASS="LABEL">Select a display format:</TD>
<TD><INPUT TYPE="RADIO" NAME="FMT" VALUE="HTML" CHECKED>HTML</TD>
<TD><INPUT TYPE="RADIO" NAME="FMT" VALUE="PDF">PDF</TD>
<TD><INPUT TYPE="RADIO" NAME="FMT" VALUE="EXL2K">Excel 2000</TD>
</TR>
</TABLE>
<P>
<INPUT TYPE="SUBMIT" NAME="SUBMIT" VALUE="Submit">
<INPUT TYPE="RESET" NAME="RESET" VALUE="Reset">
</FORM>
</BODY>
</HTML>
```

a. Radio buttons in a table prompt the user for the format (FMT), and the value is passed to the procedure.

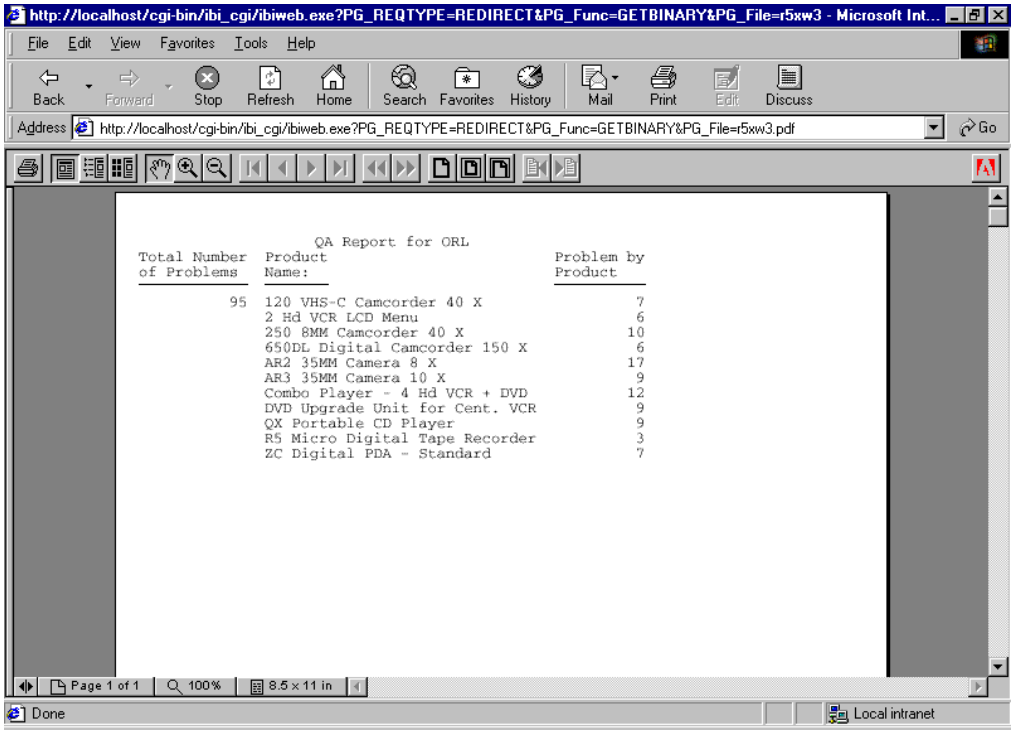
3. Publish the procedure and launch page using App Studio.

4. Run the launch page.



5. Select *ORL* from the drop-down list. Select *PDF* for the format. Click *Submit*.

WebFOCUS returns the data to the browser in PDF. The browser calls the helper application, Adobe Acrobat Reader, and displays the report:



Controlling Multiple Reports

WebFOCUS Client variables give you control over the display of multiple reports on the same HTML page with multiple frames without coding HTML FRAME syntax. It also allows you to link multiple reports with a single Table of Contents (TOC) that calls them. For example, you can use these variables to display different report formats, such as HTML and PDF, on a single launch page.

Note:

- Table of Contents (TOC) functionality is not supported for ReportCaster distribution because the WebFOCUS Client is controlling the display of multiple reports.

You can:

- Pass the variables on a call to the WebFOCUS Client from a hyperlink.

- Set the variables in a procedure with the Dialogue Manager -TYPE command. If you use this method, test your application in WebFOCUS. Either create the procedure with a text editor in WebFOCUS, or create it in App Studio, publish it, and then run it.

Use the variables described in this topic with procedures that generate more than one report. See *Managing Flow of Control in an Application* on page 329 for details on the use of -RUN, -INCLUDE, and EX to code multiple report requests.

Reference: Formats of Multiple Reports

Two formats of multiple reports are supported:

- Multiple reports displayed in a frame set.
- Table of content reports where all reports in the procedure are listed in a menu on the left side, and the first report is displayed on the right. When the user clicks on a report in the menu, it is executed and appears by default on the right side of the window.

Reference: WebFOCUS Client Variables

The following are the variables that control multiple reports.

IBIWF_mreports

Controls the multiple report option and creates either a TOC or a frameset with one report for each frame in the browser.

The syntax is

```
IBIWF_mreports = {OFF|INDEX|FRAME}
IBIWF_mrcolumns = {1|n}
IBIWF_mrrows = nIBIWF_mprefix = {Report|text}
IBIWF_morder = {FORWARD|REVERSE}
IBIWF_mframename = {MREPORT|text}
IBIWF_index = {ON|OFF}
```

where:

`OFF`

Disables the multiple report option. OFF is the default value.

`INDEX`

Creates a TOC that lists all reports in a procedure. Default sequence numbers are from 1 (for the first report generated) to *n* (for the last report generated). Used with IBIWF_mprefix.

FRAME

Creates a default frameset. The number of reports in the frameset is determined by the IBIWF_mrcolumns variable.

IBIWF_mrcolumns

Is the number of side-by-side reports from left to right across the page when IBIWF_mreports is set to FRAME. If this variable is not set, reports are displayed top to bottom.

The syntax is

```
IBIWF_mrcolumns = {1|n}
```

where:

n

Is the number of reports. The default value is 1. The maximum value is 9.

IBIWF_mrrows

Is the number of vertically stacked reports when IBIWF_mreports is set to FRAME.

The syntax is

```
IBIWF_mrrows = n
```

where:

n

Is the number of reports desired from top to bottom.

IBIWF_mprefix

Is descriptive text that precedes a sequence number and identifies a report on a TOC.

WebFOCUS appends the number 1 (for the first report generated) to *n* (for the last report generated), as set by the index value on IBIWF_mreports.

Do not use this variable if IBIWF_mreports = FRAME.

The syntax is

```
IBIWF_mprefix = {Report|text}
```

where:

text

Is a character string, up to 50 characters long. The maximum length does not include the number appended by WebFOCUS. The default value is Report.

IBIWF_morder

Is the order in which reports display in the browser. Applies only when IBIWF_mreports = FRAME. Ignored when IBIWF_mreports = INDEX.

The syntax is

```
IBIWF_morder = {FORWARD|REVERSE}
```

where:

`FORWARD`

Displays reports in the order in which they were coded and executed. This value is the default.

`REVERSE`

Displays reports in the reverse order in which they were coded and executed. This is especially useful if the last report is a summary report you would like to display on the webpage first.

IBIWF_mframename

Is a name for a frame when IBIWF_mreports = FRAME. If you do not code this variable, WebFOCUS internally names the frames MREPORT1 through MREPORT*n*, which may conflict with other HTML code.

The syntax is

```
IBIWF_mframename = {MREPORT|text}
```

where:

text

Is a character string, up to 20 characters long.

IBIWF_index

Controls whether a sequence number (1, 2,...*n*) is appended to the end of the names on the TOC when IBIWF_mreports = INDEX.

The syntax is

```
IBIWF_index = {ON|OFF}
```

where:

ON

Appends a sequence number of 1 (for the first report generated) to *n* (for the last reported generated). ON is the default value.

OFF

Omits a sequence number. Only the text specified by IBIWF_mprefix applies.

Syntax: How to Control Multiple Reports From a Hyperlink

```
<A HREF="/alias/WFServlet?IBIF_ex=report1[&var=value[&var=value]...]">
text</A>
<A HREF="/alias/WFServlet?IBIF_ex=report2[&var=value[&var=value]...]">
text</A>
```

where:

alias

Points to the directory in which the WebFOCUS Client is located. An application or web server uses an alias to provide a logical name for a physical directory. The WebFOCUS default alias is ibi_apps. It is customizable during the WebFOCUS Client installation from which it is then configured..

To call WebFOCUS on another web server, specify a fully qualified URL that includes the server name and port of the application or web server, For example

```
<A HREF="http://servername:port/alias/WFServlet?
IBIF_ex=report1[&var=value[&var=value]...]"> text</A>
<A HREF="http://servername:port/alias/WFServlet?
IBIF_ex=report2[&var=value[&var=value]...]"> text</A>
```

servername

Is the name of the application or web server on which WebFOCUS is installed.

port

Is the port on which the server is listening

report1, *report2*

Is the name of the procedure to run.

var=value

Is a WebFOCUS Client variable and its corresponding value.

You can pass more than one variable-value pair, but do not include a space between pairs. Use an ampersand (&) as a delimiter to separate each variable-value pair. A value can be a maximum of 80 characters long.

If a value contains an embedded blank, substitute a plus sign (+) or the characters %20 for the blank.

For a list of variables and valid values, see *WebFOCUS Client Variables* on page 267.

text

Is the text on the launch page that serves as the hyperlink that runs the procedure.

Example: Displaying Two Reports With an Index Value

The following is an example of displaying two reports with an index value.

1. Create a procedure named TWORPTS, which consists of two requests. The first generates a report on total dollar sales; the second, on total unit and dollar sales.

   Procedure:

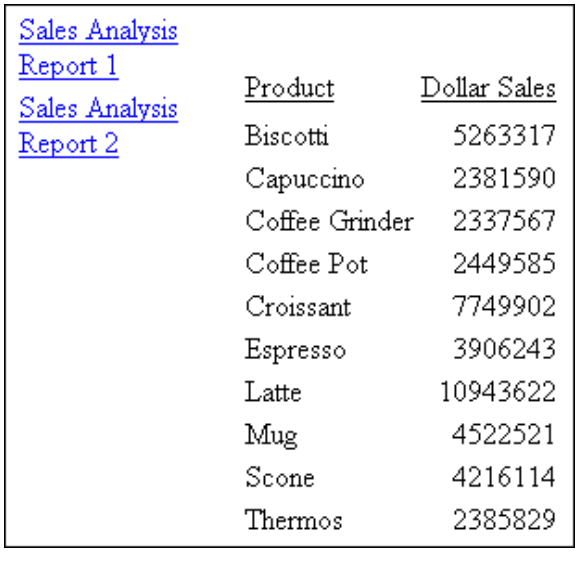
```
TABLE FILE GGSALES
SUM DOLLARS BY PRODUCT
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
-RUN

TABLE FILE GGSALES
SUM UNITS DOLLARS BY PRODUCT
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
-RUN
```

2. Create a launch page named TWOLNCH. It contains a hyperlink that calls the WebFOCUS Servlet and passes it the name of the procedure to run. It also sets the variables that provide an identifier (Sales Analysis Report) and sequence number (1 and 2) to each report.

   Launch Page:

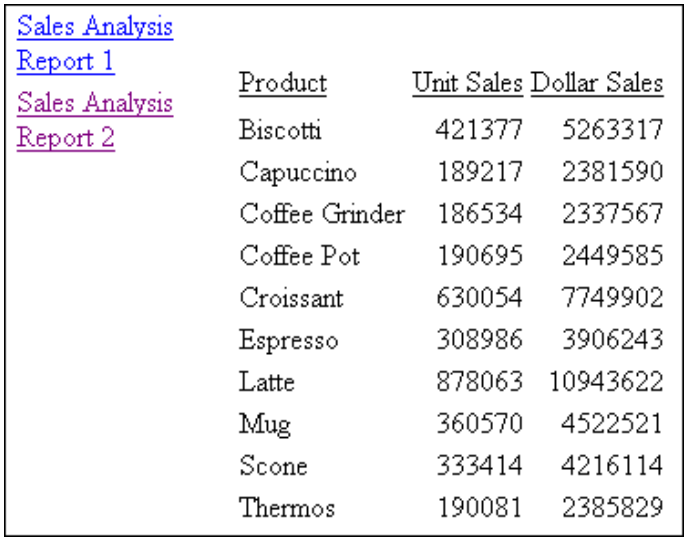
```
<HTML>
<BODY>
<A HREF="/ibi_apps/WFServlet?IBIF_ex=tworpts
&IBIWF_mreports=index&IBIWF_mprefix=Sales+Analysis+Report">
Run report.</A>
</BODY>
</HTML>
```

3. Publish the procedure and launch page using App Studio.
4. Run the launch page in the browser, and click *Run report* to receive the report on total dollar sales:



Product	Dollar Sales
Biscotti	5263317
Capuccino	2381590
Coffee Grinder	2337567
Coffee Pot	2449585
Croissant	7749902
Espresso	3906243
Latte	10943622
Mug	4522521
Scone	4216114
Thermos	2385829

Click *Sales Analysis Report 2* for the report on total unit and dollar sales:



Syntax: How to Control Multiple Reports From a Procedure

```
-TYPE WEBFOCUS CGIVAR var=value
.
.
.
-RUN
```

where:

var=value

Is a WebFOCUS Client variable and its corresponding value. You can include only one variable-value pair per each -TYPE command.

For a list of variables and valid values, see *WebFOCUS Client Variables* on page 267.

Note: Include the command -RUN at the end of each request to execute the previous set of -TYPE commands.

Example: Displaying Side-By-Side Reports

The following is an example of displaying side-by-side reports.

1. Create a procedure named SIDERPTS, which consists of two requests. The first generates a report on sales by store. The second generates a report on sales by product.

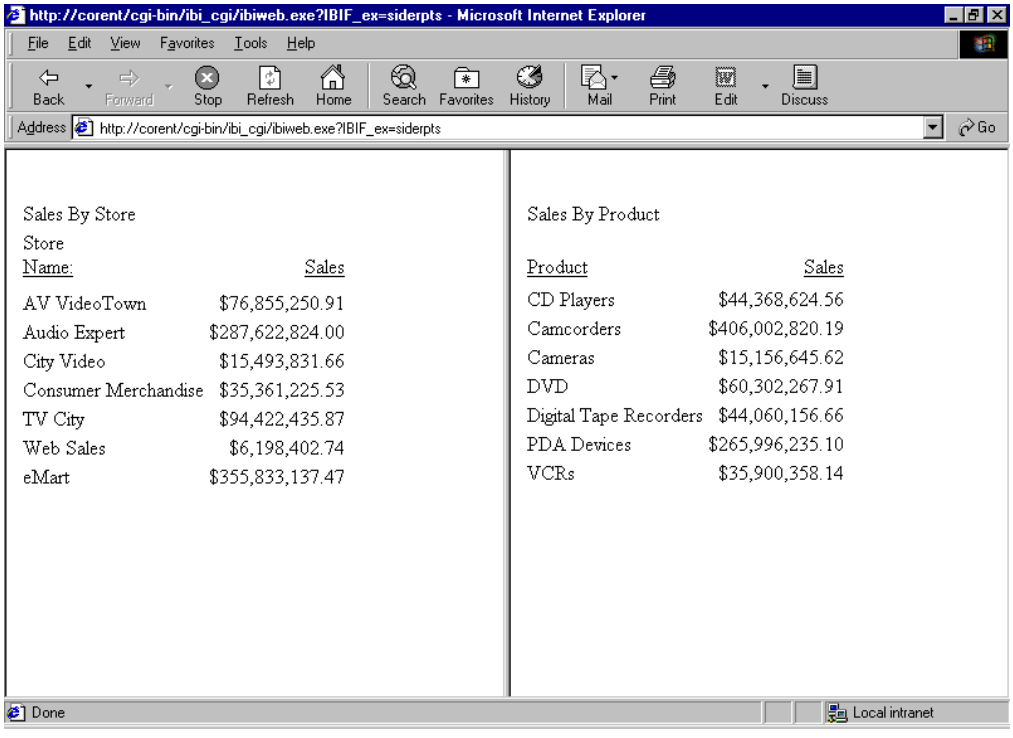
-TYPE commands create a two-frame page on which the reports display side-by-side.

Procedure:

```
-TYPE WEBFOCUS CGIVAR IBIWF_mreports=FRAME
-TYPE WEBFOCUS CGIVAR IBIWF_mrcolumns=2
TABLE FILE CENTORD
HEADING
"Sales By Store"
SUM LINEPRICE AS 'Sales'
BY SNAME
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
-RUN
TABLE FILE CENTORD
HEADING
"Sales By Product"
" "
SUM LINEPRICE AS 'Sales'
BY PRODCAT AS 'Product'
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
```

2. Create a launch page that runs the procedure.
3. Publish the procedure and launch page using App Studio.

4. Access the launch page in the browser, and run the reports:



Example: Displaying Two Reports With Descriptive Names and Sequence Numbers

The following is an example of displaying two reports with descriptive names and sequence numbers.

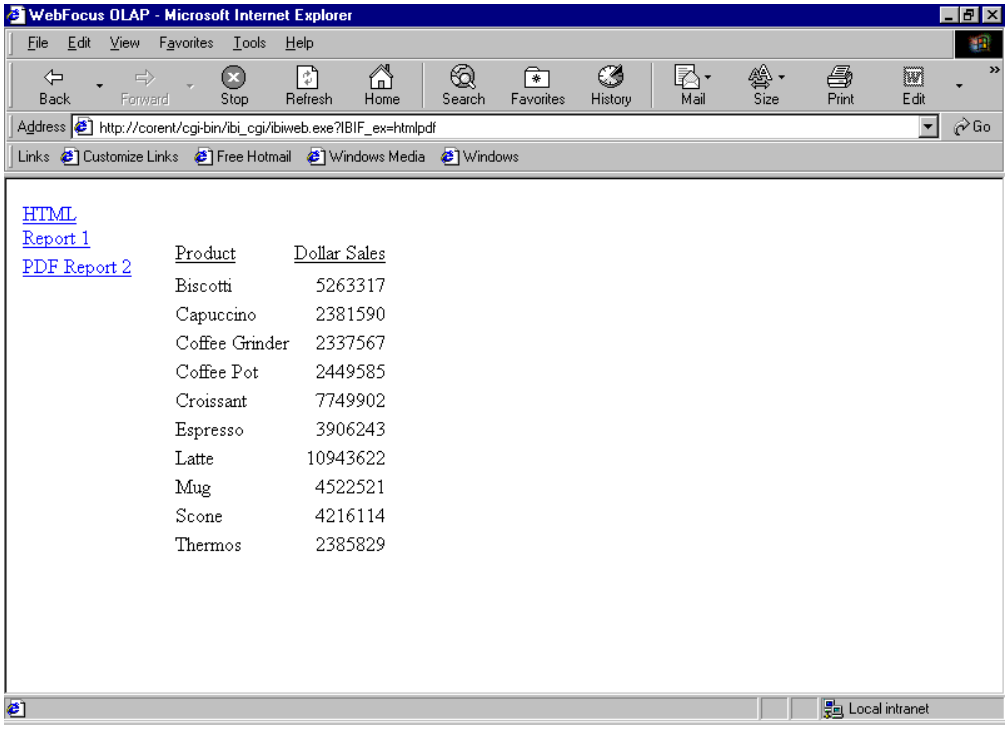
1. Create a procedure named HTMPDF1, which consists of two requests. The first generates a report on total dollar sales in HTML format; the second, on total dollar sales in PDF format.

   -TYPE commands set the variables that provide an identifier (HTML Report and PDF Report) and sequence number (1 and 2) to each report. The command that creates the TOC is required only once at the beginning of the procedure.

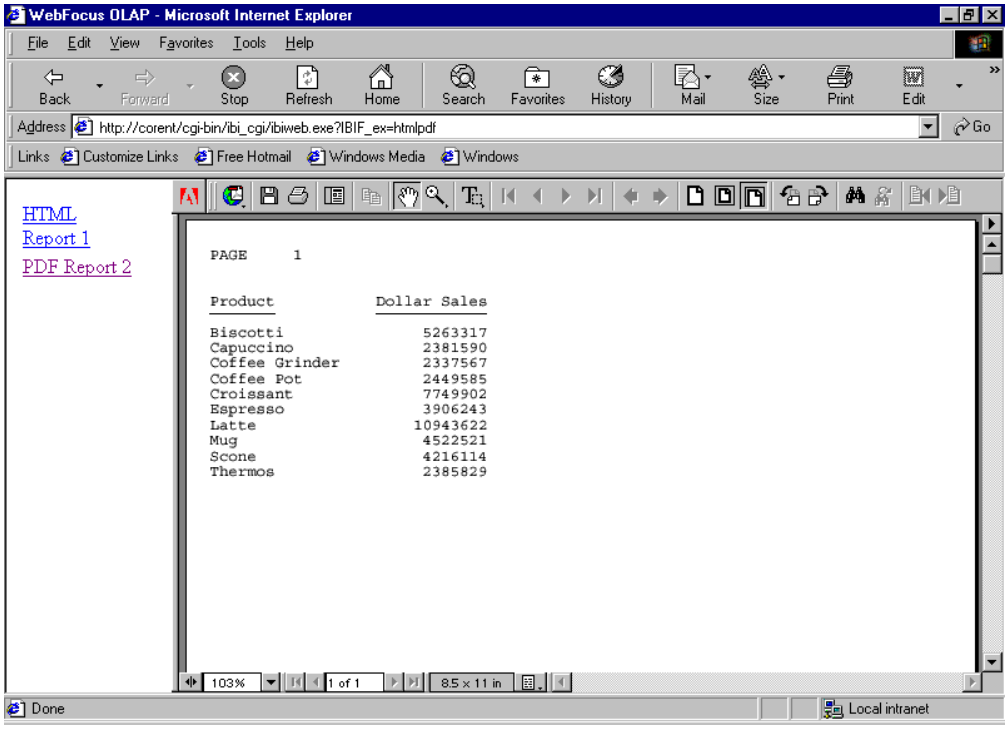
   Procedure:

```
-TYPE WEBFOCUS CGIVAR IBIWF_mreports=INDEX
-TYPE WEBFOCUS CGIVAR IBIWF_mprefix=HTML Report
TABLE FILE GGSALES
SUM DOLLARS BY PRODUCT
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
-RUN

TABLE FILE GGSALES
-TYPE WEBFOCUS CGIVAR IBIWF_mprefix=PDF Report
SUM DOLLARS BY PRODUCT
ON TABLE PCHOLD FORMAT PDF
END
```

2. Create a launch page that runs the procedure.
3. Publish the procedure and launch page using App Studio.
4. Run the launch page in the browser. The report in HTML format displays.



5. Click *PDF Report 2* for the report in PDF using the Acrobat Reader.



Example: Displaying Two Reports With Descriptive Names Only

The following is an example of displaying two reports with descriptive names only.

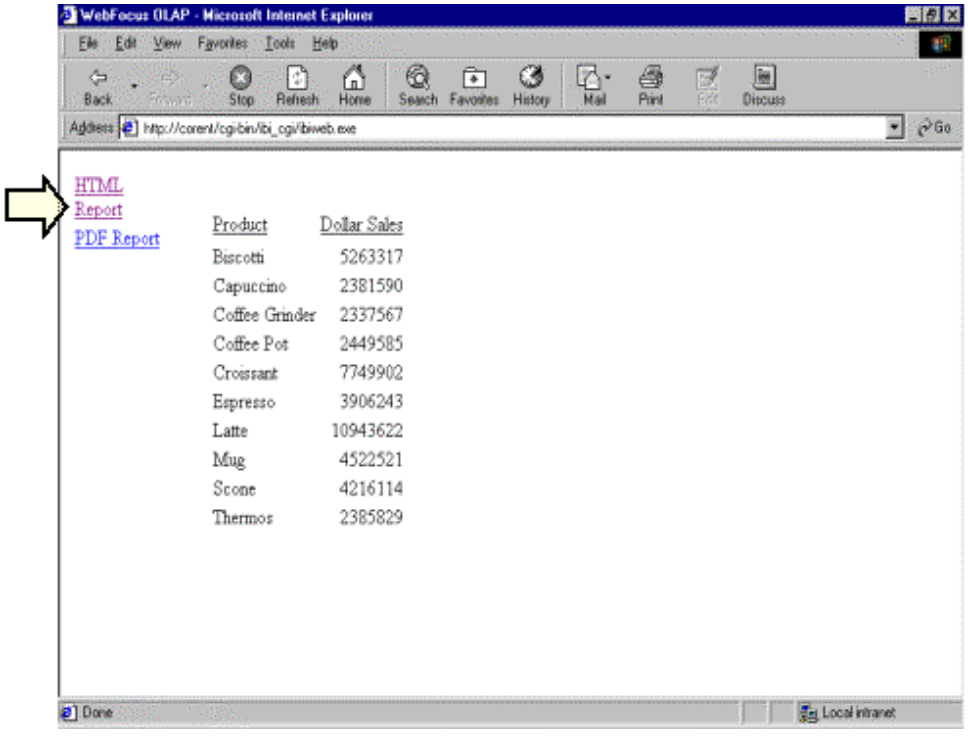
1. Create a procedure named HTMPDF2, which consists of two requests. The command -TYPE WEBFOCUS CGIVAR IBIWF_index=OFF omits sequence numbers from the names in the TOC.

Procedure:

```
-TYPE WEBFOCUS CGIVAR IBIWF_mreports=INDEX
-TYPE WEBFOCUS CGIVAR IBIWF_index=OFF
-TYPE WEBFOCUS CGIVAR IBIWF_mprefix=HTML Report
TABLE FILE GGSALES
SUM DOLLARS BY PRODUCT
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
-RUN
```

```
-TYPE WEBFOCUS CGIVAR IBIWF_mprefix=PDF Report
TABLE FILE GGSALES
SUM DOLLARS BY PRODUCT
ON TABLE PCHOLD FORMAT PDF
END
```

2. Create a launch page that runs the procedure.
3. Publish the procedure and launch page using App Studio.
4. Run the launch page in the browser. The reports have descriptive names without sequence numbers.



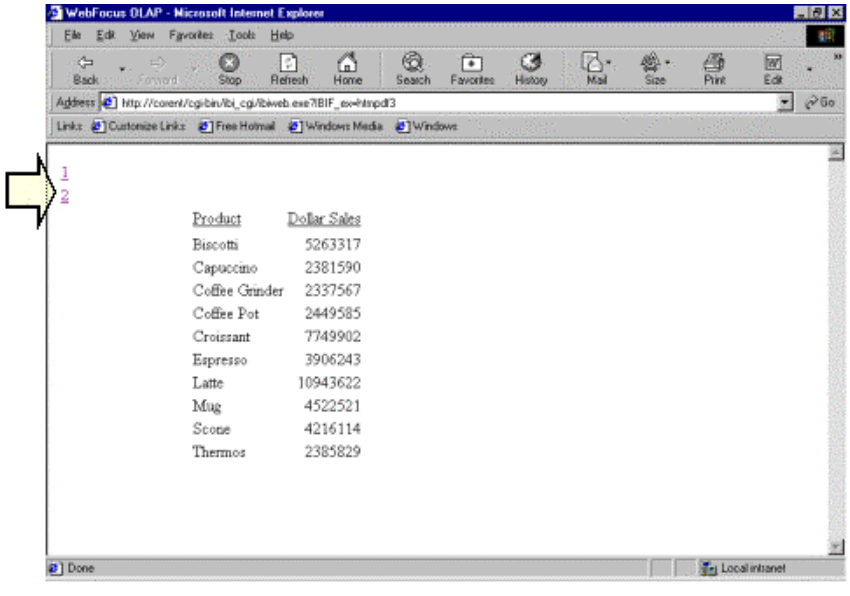
Example: Displaying Two Reports With Sequence Numbers Only

The following is an example of displaying two reports with sequence numbers only.

1. Create a procedure named HTMPDF3, which consists of two requests. The command -TYPE WEBFOCUS CGIVAR IBIWF_mprefix= omits descriptive names for the reports.

```
-TYPE WEBFOCUS CGIVAR IBIWF_mreports=INDEX
-TYPE WEBFOCUS CGIVAR IBIWF_mprefix=
TABLE FILE GGSALES
SUM DOLLARS BY PRODUCT
ON TABLE SET PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
ENDSTYLE
END
-RUN

TABLE FILE GGSALES
SUM DOLLARS BY PRODUCT
ON TABLE PCHOLD FORMAT PDF
END
```

2. Create a launch page that runs the procedure.
3. Publish the procedure and launch page using App Studio.
4. Run the launch page in the browser. The reports have sequence numbers without descriptive names:



Adding JavaScript for Drill-Down Reporting

This topic illustrates the use of JavaScript to create a drill-down report. It describes how to call a JavaScript function and pass values to it from the summary component of the report, to dynamically determine the content of the detailed component.

You will see how to specify a hyperlink to a JavaScript function in a procedure's StyleSheet. Once a hyperlink is defined, a user can select the associated object in the report to execute the function.

For more information on StyleSheets, see the *Creating Reports With WebFOCUS Language* manual.

For details on JavaScript capabilities and syntax, see your JavaScript documentation.

Syntax: How to Specify a Hyperlink to a JavaScript Function

```
TYPE=type, {COLUMN|ACROSSCOLUMN}=fieldname, JAVASCRIPT=func[(value)],$
```

where:

type

Is the report component that serves as the hyperlink. The TYPE attribute and its value must be first in the StyleSheet.

For example, use TYPE=DATA to set up a hyperlink from a data field in a report, or use TYPE=REPORT to set up a hyperlink from any component in a report.

fieldname

Is the name of the field in the data source whose value, when selected, executes the hyperlink.

func

Is the name of the JavaScript function.

The maximum length of the code for JAVASCRIPT=*func*, including any passed values, is 800 characters. The code can span more than one line. If you split it across a line, use a backslash at the end of the first line as the continuation character. If you split a line at a point at which a space is required, the space must be before the backslash, or must be the first character on the next line.

In this example,

```
JAVASCRIPT=myfunc(COUNTRY\CAR MODEL),$
```

the code correctly spans two lines.

value

Is a value or values passed to the JavaScript function. Specify a value in one of the following ways:

- As the name of a report column.
- As a constant. You must enclose the value in single quotation marks.
- As a Dialogue Manager amper variable. You can only specify an amper variable in a StyleSheet that is part of the procedure (inline).

An amper variable is typically used to pass a constant value, in which case it must be enclosed in single quotation marks, for example, '&ABC'.

Example: Creating a Drill-Down Report With JavaScript

In this example, the summary component of a drill-down report displays orders per month for each store code. When the user selects a particular store code, a hyperlink calls a JavaScript function that performs calculations on the data and displays detailed information for the selected store.

1. Create a procedure named JDRILL. The letters on the left correspond to the notes explaining the code.

```
   SET MESSAGE = OFF
   TABLE FILE GGORDER
   SUM QUANTITY BY STORE_CODE ACROSS ORDER_DATE
   IF ORDER_DATE GT 12/31/96
a. ON TABLE HOLD
   END
   -RUN
   TABLE FILE HOLD
   HEADING CENTER
   "Store Sales Analysis Using JavaScript"
b. PRINT E01 AS 'Store,Code' E02 AS 'Jan' E03 AS 'Feb' E04 AS 'Mar'
                             E05 AS 'Apr' E06 AS 'May' E07 AS 'Jun'
                             E08 AS 'Jul' E09 AS 'Aug' E10 AS 'Sep'
                             E11 AS 'Oct' E12 AS 'Nov' E13 AS 'Dec'
   FOOTING CENTER
   "Please click on the store code to summarize the store's data."
c. ON TABLE HOLD AS JAVATEMP FORMAT HTMTABLE
d. ON TABLE SET STYLE *
   TYPE=DATA,COLUMN=STORE_CODE,$
   JAVASCRIPT=conprint(E01 E02 E03 E04 E05 E06 E07 E08 E09 E10 E11 \
                       E12 E13),$
   COLOR=GREEN,STYLE=ITALIC,$

   TYPE=TITLE,COLOR=RED,STYLE=BOLD,$
   TYPE=HEADING,COLOR=BLUE,STYLE=ITALIC,SIZE=11,$
   TYPE=FOOTING,COLOR=BLUE,STYLE=ITALIC,$
   END STYLE
   END
e. -HTMLFORM BEGIN
   <HTML>
```

```
f. <SCRIPT LANGUAGE="JavaScript">
    var spacer = "..............................."; var pos=0;
    var aaaa;
    function conprint(aaax,lsyr,aa1,aa2,aa3,aa4,aa5,aa6,aa7,
                      aa8,aa9,aa10,aa11,aa12)
    {
    var spacer="     ";

    lsyrave=parseFloat(lsyr);
    a1=parseFloat(aa1);
    a2=parseFloat(aa2);
    a3=parseFloat(aa3);
    a4=parseFloat(aa4);
    a5=parseFloat(aa5);
    a6=parseFloat(aa6);
    a7=parseFloat(aa7);
    a8=parseFloat(aa8);
    a9=parseFloat(aa9);
    a10=parseFloat(aa10);
    a11=parseFloat(aa11);
    a12=parseFloat(aa12);
    gototа=eval(a1 + a2 + a3 + a4 + a5 + a6 + a7 + a8 + a9 + a10 +
                a11 + a12);
    goavea=gototа/12;
    gotot=Math.round(gototа);
    goave=Math.round(goavea);
    diffavea=goavea-lsyrave;
    diffave=Math.round(diffavea);
g.  document.form1.text1.value=gotot;
    document.form1.text2.value=goave;
    document.form1.store.value=aaax;
    arraygo = new Array(13);
    arraygo[1]=a1;
    arraygo[2]=a2;
    arraygo[3]=a3;
    arraygo[4]=a4;
    arraygo[5]=a5;
    arraygo[6]=a6;
    arraygo[7]=a7;
    arraygo[8]=a8;
    arraygo[9]=a9;
    arraygo[10]=a10;
    arraygo[11]=a11;
    arraygo[12]=a12;
    minval=100000;
    maxval=0;
    mnmmax=0;
    mnmmin=0;
```

```
for(i = 1; i <= 12; i++)
{
if(arraygo[i] > maxval)
{
mnmmax=I;
maxval = arraygo[i];
}
if(arraygo[i] < minval)
{
mnmmin=i;
minval = arraygo[i];
}
}
rng1=maxval - minval;
rng=Math.round(rng1);
mnms=new Array(13);
mnms[1]="January";
mnms[2]="February";
mnms[3]="March";
mnms[4]="April";
mnms[5]="May";
mnms[6]="June";
mnms[7]="July";
mnms[8]="August";
mnms[9]="September";
mnms[10]="October";
mnms[11]="November";
mnms[12]="December";
document.form1.themax.value=maxval;
document.form1.themin.value=minval;
document.form1.range.value=rng;
document.form1.mnmmax.value=mnms[mnmmax];
document.form1.mnmmin.value=mnms[mnmmin]; }
</SCRIPT>
<BODY>
<FORM name="form1">
h.  <!--WEBFOCUS TABLE JAVATEMP>
<INPUT TYPE=text NAME="text1" SIZE="10"> Units sold for store
<INPUT TYPE=text NAME="store" SIZE="10"> Monthly average of
<INPUT TYPE=text NAME="text2" SIZE="10"><BR><BR> Top selling
      month is
<INPUT TYPE=text NAME="mnmmax" SIZE="10"> with
<INPUT TYPE=text NAME="themax" SIZE="10">units<BR> Slowest
      month was
<INPUT TYPE=text NAME="mnmmin" SIZE="10"> with
<INPUT TYPE=text NAME="themin" SIZE="10">units<BR> Range
      between best and slowest months
<INPUT TYPE=text NAME="range" SIZE="10">units </FORM>
</BODY>
</HTML>
I. -HTMLFORM END
-EXIT
```

a. This command saves the report output with 1997 data to a temporary file named HOLD in native machine format. Since this is the only data necessary for the report, this server extract file is created to speed subsequent processing.

b. This code formats the report, providing descriptive column titles. The store code is the first column. The following columns contain total monthly quantity for each store.

c. This command saves the report output to a temporary file in HTML format. The file is named JAVATEMP. It will be merged with the HTML page created later (see item e).

d. The StyleSheet specifies a hyperlink to a JavaScript function named conprint. The code passes the store code and monthly values to the function.

e. The Dialogue Manager command -HTMLFORM BEGIN indicates the start of an HTML page in which the JavaScript function is defined. The report output will be embedded on this page.

f. The HTML code declares the JavaScript function and passes values to it.

g. JavaScript assigns variable names to values displayed on the HTML page.

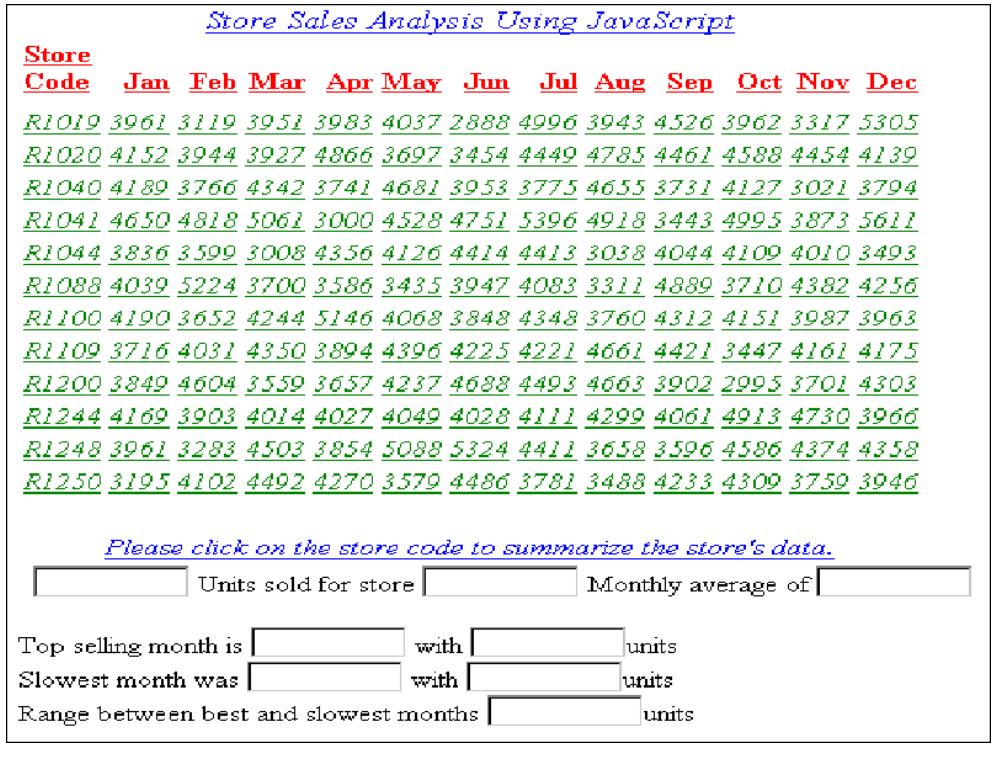
h. WebFOCUS reads the HTML comment and replaces it with the report output held in JAVATEMP.

Note: The HTML comment line must be closed with the –> characters or a single > character and should not have any other HTML tags within it.

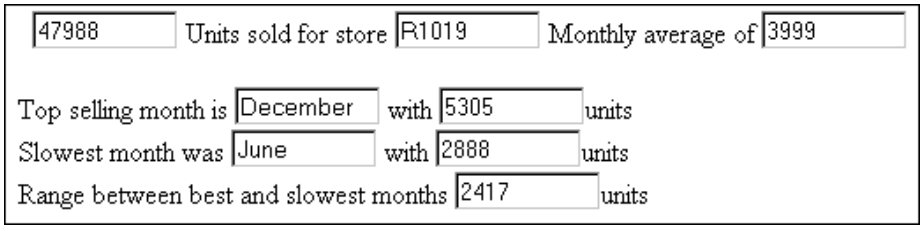
i. The Dialogue Manager command -HTMLFORM END indicates the end of the HTML page.

2. Create a launch page from which a user can run the report.

3. Run the launch page, and click the link. The summary component displays.



4. Click store code *R1019* for the detail component.



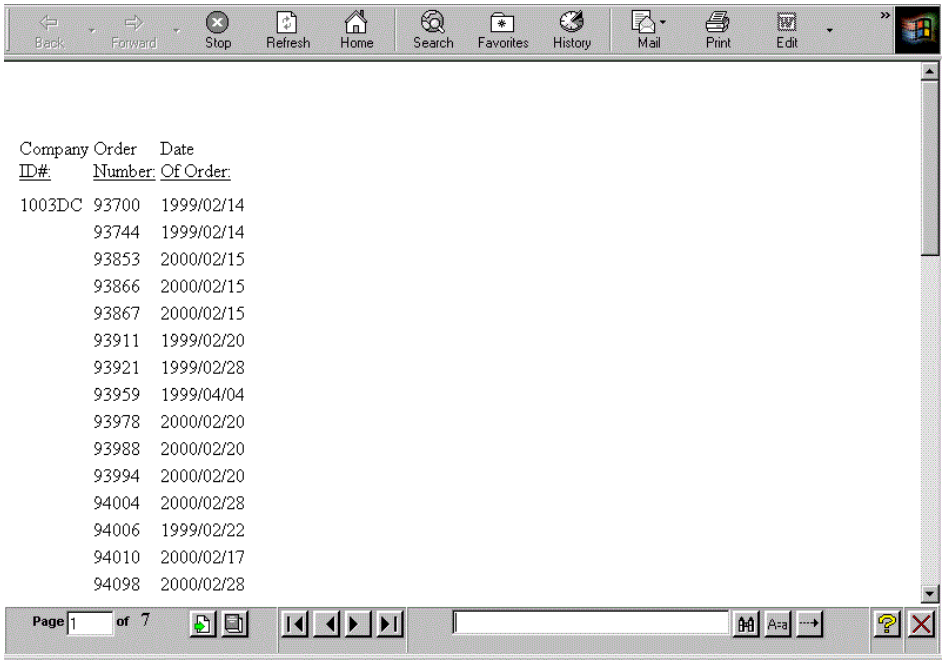
Facilitating Report Manipulation

Normally, a web server returns an entire report to a browser. The browser waits until it receives all of the report before displaying it.

On-demand paging allows you to download one page of a report that is up to 99999 pages long to a browser instead of downloading the entire report at once. The web server holds the remaining pages until the user requests them. This feature shortens the time the user waits to see the first page of a report. It is especially effective for long reports.

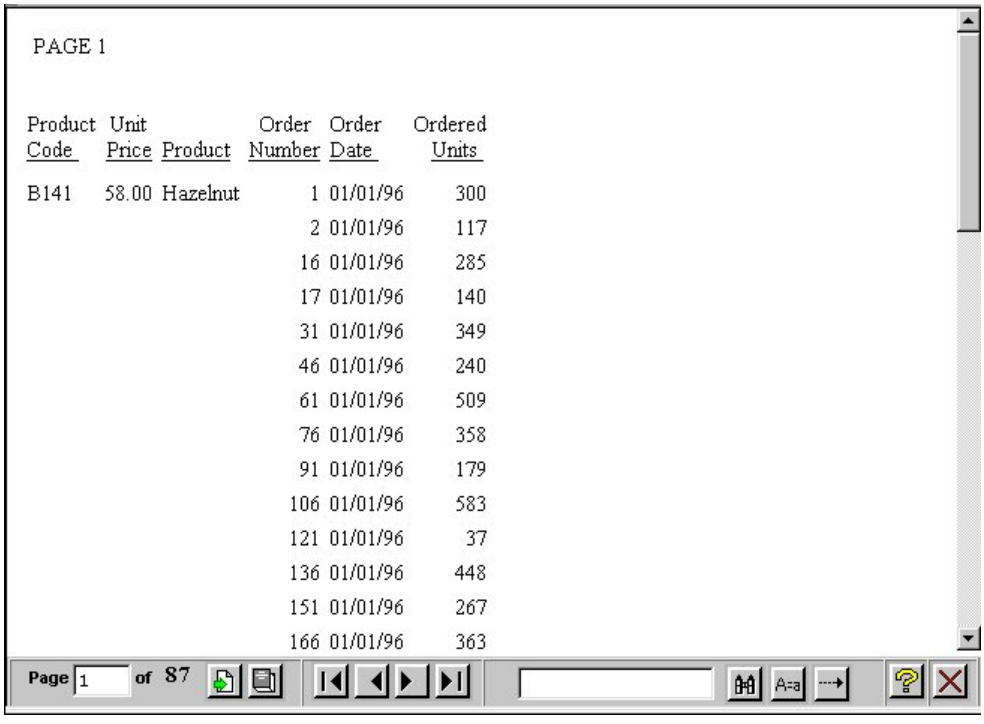
On-demand paging is implemented in the WebFOCUS Viewer. It requires that report output be formatted as HTML, which is the default setting for a request submitted through the WebFOCUS Client.

The following is a report displayed in the WebFOCUS Viewer.



Navigating a Report With the WebFOCUS Viewer

When you run a report designated for On-Demand Paging, the WebFOCUS Viewer opens automatically and displays the first page of the report. The Viewer consists of two panes: the Report Pane and the Viewer Control Panel, as shown in the following image.



The Report Pane is the larger pane and contains one page of report output. When you first run a report, the Report Pane contains the first page of report output. The Viewer Control Panel contains the controls that allow you to display specific pages, deliver the entire report to your web server, and search your document for particular strings of information.

Syntax: How to Enable the WebFOCUS Viewer

```
SET WEBVIEWER = {OFF|ON}
```

or

```
ON TABLE SET WEBVIEWER {OFF|ON}
```

where:

OFF

Disables on-demand paging. WebFOCUS downloads the entire report to a standard browser window. OFF is the default value.

ON

Enables on-demand paging. WebFOCUS downloads the first page of a report to the browser in the Viewer. The number of lines displayed at one time depends on Windows desktop settings (resolution).

Example: Enabling the WebFOCUS Viewer

The procedure and launch page for this example are run in WebFOCUS. They must be tested and run in this environment.

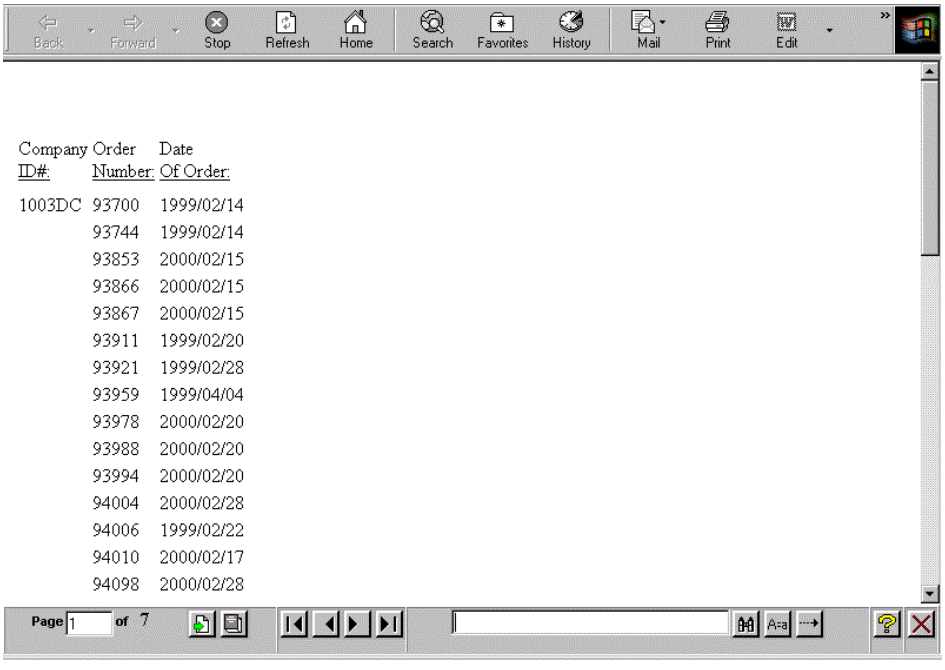
1. Create a procedure named ONDEMAND, which displays an order report for a store in the Viewer.

   Procedure:

   ```
   SET WEBVIEWER=ON
   TABLE FILE CENTORD
   PRINT ORDER_NUM ORDER_DATE
   BY STORE_CODE
   WHERE STORE_CODE EQ '1003DC'
   ON TABLE SET PAGE-NUM OFF
   ON TABLE SET STYLE *
   TYPE=REPORT, GRID=OFF,$
   ENDSTYLE
   END
   ```

2. Create a launch page from which a user can run the report.

3. Run the launch page, and click the link. The report displays in the Viewer.



Opening and Closing the WebFOCUS Viewer

You can specify a target frame in which to open the WebFOCUS Viewer, and a home page that displays when you close the Viewer.

Syntax: How to Open the WebFOCUS Viewer in a Target Frame

```
SET WEBVIEWTARG = {target_frame|OFF}
```

where:

target_frame

Is the name of an existing frame in the browser or one of the following reserved HTML target frames:

`_blank`

Opens the Viewer in a new browser window. This is the default for reports that do not have accessibility enabled.

`_self`

Opens the Viewer in the same frame as the anchor.

`_parent`

Opens the Viewer in the immediate parent frame that contains the anchor.

`_top`

Opens the Viewer in the current browser window.

`OFF`

Opens the Viewer in the frame from which you ran the report. This is the default for reports that have accessibility enabled.

Note: For more information on enabling accessibility, see *ACCESSIBLE* on page 530.

Syntax: **How to Display a Home Page When You Close the WebFOCUS Viewer**

```
SET WEBVIEWHOME = {home_URL|OFF}
```

where:

home_URL

Is a valid URL that displays an HTML page when you close the Viewer.

`OFF`

Displays a blank browser window when you close the Viewer. You must enter another URL to run another report. OFF is the default value.

Reference: **Closing the WebFOCUS Viewer**

The Close button, located on the Control Frame, closes the Viewer and removes the report from the web server. The page the browser displays next depends on the WEBVIEWTARG and the WEBVIEWHOME settings, as follows:

- If you set WEBVIEWTARG to _blank, the window that contained the Viewer is removed. The browser does not display any page in any frame, and the WEBVIEWHOME setting has no effect.
- If you set WEBVIEWTARG to any other value, the result of clicking *Close* depends on the WEBVIEWHOME setting:
  - If you set WEBVIEWHOME to a URL, the browser displays the page associated with the URL in the frame that the Viewer occupied.

- If you set WEBVIEWHOME to OFF, the browser displays a blank page.

Controlling Button Display on the WebFOCUS Viewer

You can issue commands that specify whether the Viewer displays the Close, Help, and All Pages buttons.

Syntax: How to Control Whether The Close Button Displays

```
SET WEBVIEWCLOSE = {OFF|ON}
```

or

```
ON TABLE SET WEBVIEWCLOSE {OFF|ON}
```

where:

ON

Displays the Close button. ON is the default value.

OFF

Does not display the Close button.

Syntax: How to Control Whether The Help Button Displays

```
SET WEBVIEWHELP = {OFF|ON}
```

or

```
ON TABLE SET WEBVIEWHELP {OFF|ON}
```

where:

ON

Displays the Help button. ON is the default value.

OFF

Does not display the Help button.

Syntax: How to Control Whether The All Pages Button Displays

```
SET WEBVIEWALLPG = {OFF|ON}
```

or

```
ON TABLE SET WEBVIEWALLPG {OFF|ON}
```

where:

ON

Displays the All Pages button. ON is the default value.

OFF

Does not display the All Pages button.

Procedure: How to Enable the Goto Last Page Button

1. From Internet Explorer, click the *Tools* menu and select *Internet Options*.
2. The Internet Options dialog box opens.
3. From the Temporary Internet files box, click *Settings*.

   The Settings dialog box opens.
4. Click the *Every visit to the page* radio button.
5. Click *OK* to apply the change and exit the Settings dialog box.
6. Click *OK* to exit the Internet Options dialog box.

Using the Viewer Control Panel

The Viewer Control Panel, as shown in the following image, (located at the bottom of the window) contains the controls you use to navigate through the report and to search for a string in the report. The Viewer Control Panel navigational controls allow you to display the next or previous page, the first or last page, or a specific page. You use the searching function to have the Viewer locate a search string you specify within all report pages.

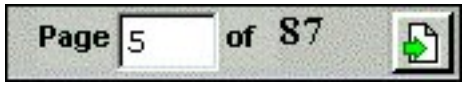
Note: When specifying a search string, you must specify the actual number of spaces between characters because HTML displays a single space, even when multiple spaces are used between characters.

Procedure: How to Navigate Through a Report

The Viewer Control Panel offers several ways to view pages in your report:

❑ To display a specific page:

1. Enter a page number in the Page input box, as shown in the following image.



2. Click Go to Page, as shown in the following image.

- To display the previous or the next page in sequence, click *Previous* or *Next*, as shown in the following image.

- To display the first or last page of the report, click *First Page* or *Last Page*, as shown in the following image.

- To download the entire report to the browser as a single document, click *All Pages*, as shown in the following image.

- To close the Viewer, click *Close*, as shown in the following image.

Searching a Report

The Viewer Control Panel contains controls that offer several ways to search your report. Using the Viewer search controls, you can select a string of information, such as a phrase that occurs in your report or a group of numbers, and search for each occurrence of that string. You can further customize your search by matching capitalization of words exactly (a case-sensitive search) or by controlling the direction of your search (either forward or backward from your starting point in the report). Use these controls to search your report:

- To perform a case-sensitive search, click *Match Case*, as shown in the following image.

- To search backward in a report, click *Search Backward*, as shown in the following image.

- To locate a specific string, type the string you want to search for and click *Find*, as shown in the following image.

Procedure: How to Search the Report

1. Enter the string in the Search input box.
2. Click *Match Case* if you want to perform a case-sensitive search.

   Notice that the Viewer displays the Match Case button with a red line across it to indicate that it is active.
3. To begin your search, click:

   - *Search Backward* to search for the string from the current page back to the first page.

     Or

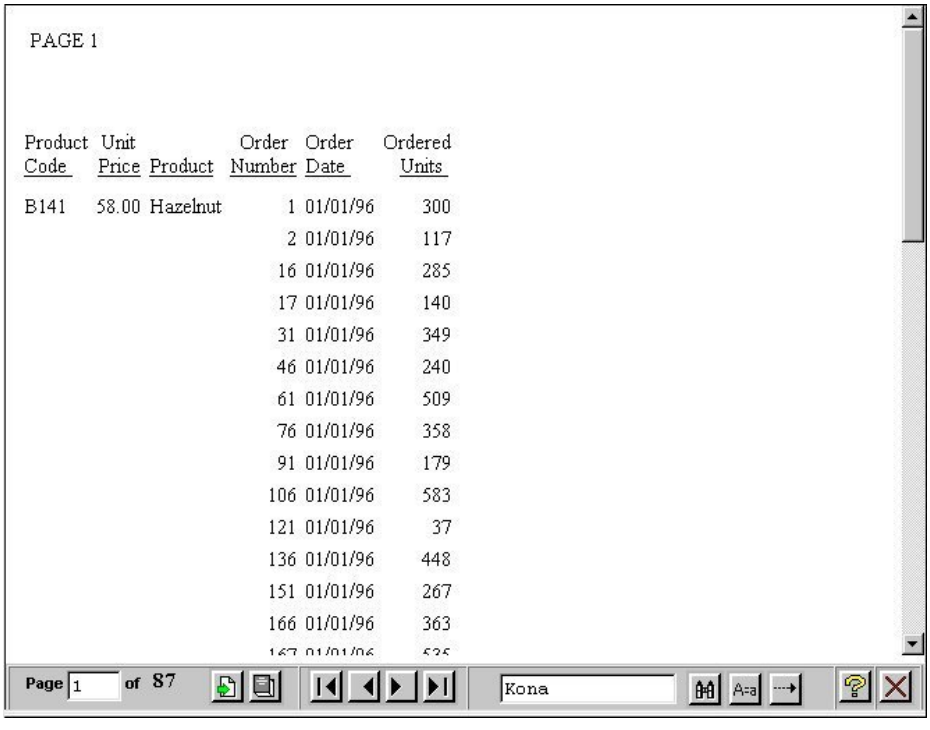
   - *Find* to search from the current page to the end of your report.

   The Viewer searches the report, underlines the first occurrence of the string, and opens the display to the top of the page on which it appears. If the underlined occurrence of the string is not visible, scroll down the page until it appears in the window of the Viewer.
4. Click *Find* again to search for another occurrence of the string.

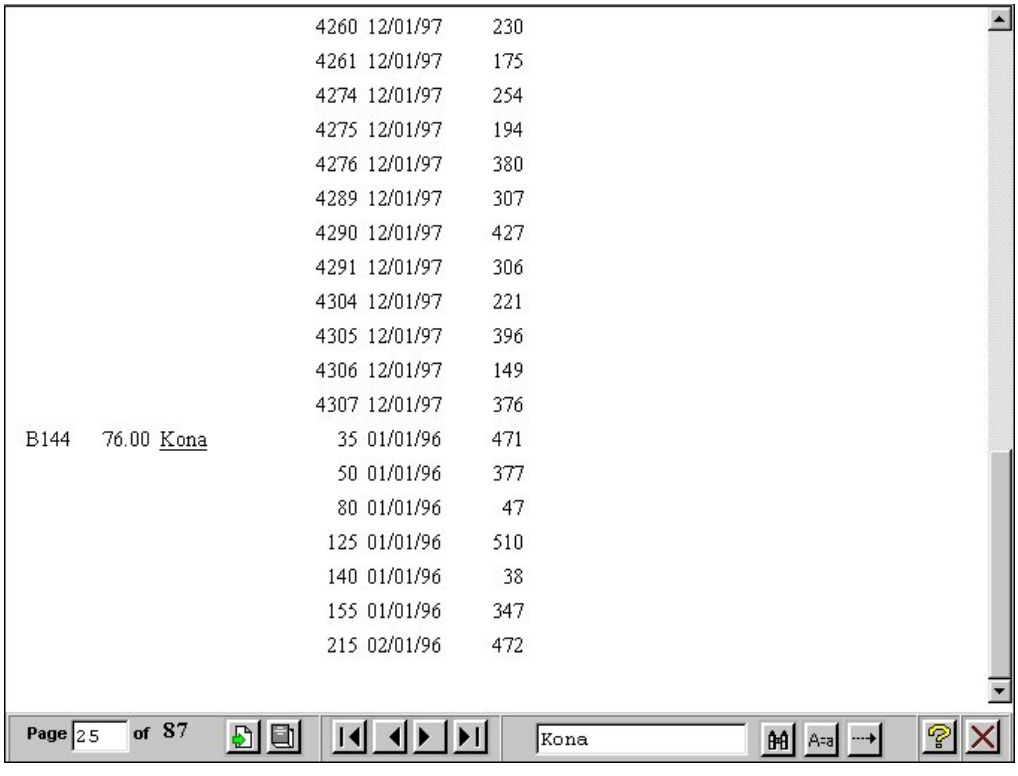
Example: Using the Viewer Control Panel to Search

You want to use the Viewer Control Panel to navigate a long report called Coffee Sales to find occurrences of the string "Kona," a type of coffee that you sell. After you run the report, WebFOCUS displays the first page of the report in the Viewer.

1. To search for sales of Kona, type Kona in the input box and click *Find*, as shown in the following image.



The Viewer returns your report with the first occurrence of your search string underlined, as shown in the following image.



2. Click *Find* again to locate the next occurrence of Kona.

Printing With On-Demand Paging

You must clear the browser cache before printing a report locally using the Print button on the browser toolbar. You must also activate the window by clicking it before using the Print button.

Use the procedure that applies to your browser to clear the cache.

Procedure: How to Clear the Cache in Microsoft Internet Explorer

1. Select *Internet Options* from the Tools menu.
2. On the General tab, for Temporary Internet Files, select *Delete Files*.

Using a Cascading Style Sheet to Standardize Display

A Cascading Style Sheet is an extension to HTML that allows you to specify formatting for an HTML page. A Cascading Style Sheet can reside either in the HTML page that it formats, or in a separate file (with the extension .CSS), which can be shared by multiple pages. When it is in a separate file, it is known as an *external* Cascading Style Sheet.

This topic illustrates the use of an external Cascading Style Sheet to add scroll bars to reports when necessary and set standard fonts, font sizes, colors, and other display characteristics.

For details on Cascading Style Sheets, see the *Creating Reports With WebFOCUS Language* manual.

Example: Adding Scroll Bars to a Report

The following is an example of adding scroll bars to a report.

1. Create a procedure named SCROLL, which consists of two report requests and two graph requests. Each request uses a WebFOCUS StyleSheet to add individual styling features. The WebFOCUS StyleSheets for the reports call an external Cascading Style Sheet to add standardized application styling. See the *Creating Reports With WebFOCUS Language* manual for details on WebFOCUS StyleSheets and graph formatting options.

   The letters on the left correspond to the notes explaining the code.

   Procedure: SCROLL

```
   SET PAGE-NUM=OFF

   TABLE FILE CENTORD
   HEADING
   "Sales By Store"
   SUM LINEPRICE AS 'Sales'
   BY SNAME
a. ON TABLE SET STYLE *
b. TYPE=HEADING, CLASS=HEAD, $
```

```
b. TYPE=TITLE, CLASS=TITLE, $
   TYPE=REPORT, GRID=OFF, $
   TYPE=DATA,COLUMN=SNAME,COLOR=RED,STYLE=BOLD,
   WHEN=LINEPRICE LT 10000000,$
   TYPE=DATA,COLUMN=LINEPRICE,COLOR=RED,STYLE=BOLD,
   WHEN=LINEPRICE LT 10000000,$
   ENDSTYLE
   ON TABLE HOLD AS CREPORT1 FORMAT HTMTABLE
   END

    GRAPH FILE CENTORD
    SUM LINEPRICE
    ACROSS SNAME
a.  ON GRAPH SET STYLE *
    UNITS=IN, LEFTMARGIN=0.250000,
        RIGHTMARGIN=0.250000, TOPMARGIN=0.250000,
        BOTTOMMARGIN=0.250000, SQUEEZE=ON, ORIENTATION=PORTRAIT, $
    DEFMACRO=COND0001, MACTYPE=RULE, WHEN=N1 LE    100000.00, $
    TYPE=REPORT, FONT='VERDANA', SIZE=10, BACKCOLOR=NONE,
        STYLE=NORMAL, $
    TYPE=DATA, ACROSSCOLUMN=N1, COLOR=RGB(144 24 24), $
    TYPE=DATA, ACROSSCOLUMN=N1, COLOR=YELLOW, MACRO=COND0001, $
    ENDSTYLE
    ON GRAPH SET LOOKGRAPH 3D_BAR
    ON GRAPH SET GRAPHEDIT OFF
    ON GRAPH SET GRAPHSTYLE *
    setConnectLineMarkers(true);
    setO1LabelDisplay(true);
    setO1AxisSide(0);
    setO1MajorGridDisplay(false);
    setO1MinorGridDisplay(false);
    setY1LabelDisplay(true);
    setY1AxisSide(0);
    setY1MajorGridDisplay(false);
    setY1MinorGridDisplay(false);
    setPieFeelerTextDisplay(1);
    setPieLabelDisplay(0);
    setTextFormatPreset(getPieSliceLabel(),1);
    setLegendDisplay(true);
    setFootnoteString("Store Sales");
    setTextJustHoriz(getFootnote(),1);
    setFontStyle(getFootnote(),2);
    ENDSTYLE
    ON GRAPH SET BARNUMB OFF
    ON GRAPH SET 3D ON
    ON GRAPH SET GRID ON
    ON GRAPH SET VAXIS 200
    ON GRAPH SET HAXIS 300
    ON GRAPH HOLD AS CGRAPH1 FORMAT HTMTABLE
    END
```

```
     SET LOOKGRAPH=PIE
     SET GRAPHEDIT=OFF
     SET GRID=ON
     SET BARNUM=ON
     SET 3D=ON
     SET VAXIS=250
     SET HAXIS=250
     GRAPH FILE CENTORD
     SUM LINEPRICE
     BY PRODCAT
a.   ON GRAPH SET GRAPHSTYLE *
     setLegendDisplay(true);
     setTitleDisplay(true);
     setTextRotation(getO1Label(),0);
     setY1LabelFormat(10);
     setO1LabelAutofit(false);
     setO1LabelStagger(false);
     setTextRotation(getO1Label(),0);
     setFontSizeVC(getO1Label(),1500);
     setY1LabelAutofit(false);
     setFontSizeVC(getY1Label(),1800);
     setTextWrap(getLegendText(0),true);
     setRect(getLegendArea(), new Rectangle(9000, -8000,8000, 15000));
     setAutofit(getLegendText(0),false);
     setFontSizeVC(getLegendText(0),850);
     setGroupLabel(0,"Sales By Product");
     setPieTilt(45);
     ENDSTYLE
     ON GRAPH HOLD AS CGRAPH2 FORMAT HTMTABLE
     END

     TABLE FILE CENTORD
     HEADING
     "Sales By Product"
     SUM LINEPRICE AS 'Sales'
     BY PRODCAT AS 'Product'
a.   ON TABLE SET STYLE *
b.   TYPE=HEADING, CLASS=HEAD, $
b.  TYPE=TITLE, CLASS=TITLE, $
     TYPE=REPORT, GRID=OFF, $
     ENDSTYLE
     ON TABLE HOLD AS CREPORT2 FORMAT HTMTABLE
     END
c.   -HTMLFORM scrollpg
```

a. These commands indicate the start of a WebFOCUS StyleSheet.

b. The CLASS attribute in the WebFOCUS StyleSheet refers to a set of styling characteristics in an external Cascading Style Sheet (SCROLLSS.CSS, which you create in step 3).

c. This command calls an HTML display page named SCROLLPG, which you create in step 2. It will incorporate the report output.

2. Create an HTML display page named SCROLLPG.HTM. The WebFOCUS Reporting Server must be able to locate this page using APP PATH or EDAPATH. For details on search paths, see *WebFOCUS Application Logic* on page 27.

   In Windows and UNIX, an HTML file called by a -HTMLFORM command must have the extension .HTM.

   The display page links to an external Cascading Style Sheet you create in step 3.

   HTML Display Page: SCROLLPG.HTM

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title>Cent Corp Demo KPI Template</title>
<link rel="stylesheet"
   href="/ibi_apps/ibi_html/NewCentCorp/scrollss.css"
    type="text/css">
</head>
<body>
<div>!IBI.FIL.CREPORT1;</div>
<div>!IBI.FIL.CGRAPH1;</div>
<div>!IBI.FIL.CGRAPH2;</div>
<div>!IBI.FIL.CREPORT2;</div>
<div id="toolbar">
<img src="/ibi_apps/ibi_html/NewCentCorp/images/wflogo.gif" width="96"
    height="14" border="0" alt="" style="position: relative;
    top: 2px;">
 
</div>
</body>
</html>
```

3. Create a Cascading Style Sheet named SCROLLSS, which positions the reports and graphs on the display page, creates the scroll bars for report display, and sets text and background colors. The web server must be able to locate the Cascading Style Sheet.

   The letters on the left correspond to the notes explaining the code.

Cascading Style Sheet: SCROLLSS.CSS

```
a.  { color: Navy;
      font-weight : bold;
    }
    /* the following rule controls the default styling for fonts in
    the application */
    body, td {
      font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
      font-size : 12px;
      font-style : normal;
      font-variant : normal;
      font-weight : normal;
    /*  width : 10%; */
    /* the rule below allows the scroll bars to be customized */
    scrollbar-base-color : Blue;
    scrollbar-arrow-color : white;
    }
      #toolbar {
        position: relative;
        top: 500;
        left: 0;
        width: 430px;
        height: 20pt;
        padding: 6px;
        background: 0033ff;
        border : thin outset;
    }
    /* the following defines position and properties of top left
       report */
    .report1 { position: absolute;
a.          top: 30;
            left: 60;
            width: 200;
            height: 200;
            border: thin outset;
b.          overflow: auto;
    }
    /* the following defines position and properties of top right
       graph */
    .graph1 { position: absolute;
```

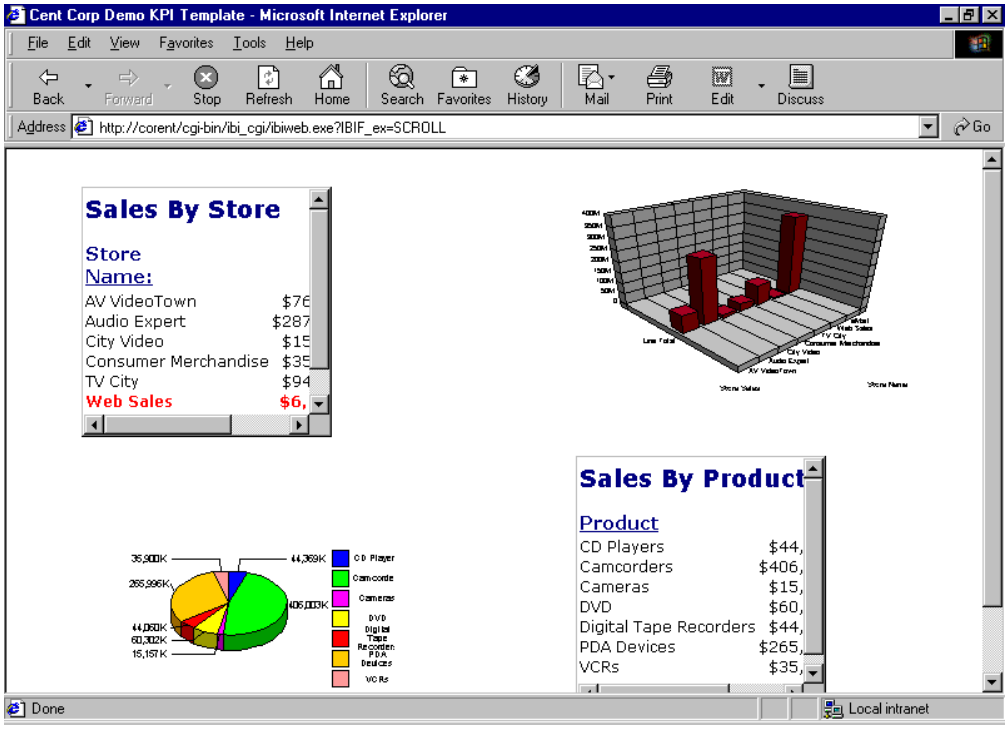
```
c.      top: 0;
        left: 440;
        overflow: auto;
    }
    /* the following defines position and properties of bottom left
       graph */
    .graph2 {  position: absolute;
        top: 246;
        left: 60;
        overflow: auto;
        }
    /* the following defines position and properties of bottom right
       report */
   .report2 { position: absolute;
        top: 245;
        left: 455;
        width: 200;
        height: 200;
        border: thin outset;
        overflow: auto;
   }
   /* this class sets the styling for a WebFOCUS heading */
   .HEAD {
d.    font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
      font-size: 14pt;
      font-weight: bold;
      color: #333366;
      position: relative;
      top: -13px;
   }
   /* this class sets the styling for WebFOCUS column titles */
   .TITLE {
e.    font-family : Verdana, Geneva, Arial, Helvetica, sans-serif;
      font-size : 11pt;
      font-weight : bold;
      color:  Navy;
      /*position: relative;
      top : 3px; */
   }
   /* this rule controls the defaults for all tables */
   TABLE {
     border-collapse: collapse;
     /*  table-layout : fixed; */
   }
   /* the two rules below controls styling for various headings */
   h2 {  font-size: 14pt; color: gold; }
   h3 {  font-size: 14pt; color: white; }
   /* this rule sets the style for the mouse over effect on an
      anchor */
   a:hover {  color: red; }
```

a. Position the first report on the HTML page.

b. Create scroll bars if the report cannot be entirely displayed at one time in a frame.

c. Position the first graph on the HTML page.
d. Set styling characteristics that are applied to WebFOCUS report headings.
e. Set styling characteristics that are applied to WebFOCUS column titles.

4. Create a launch page that runs the procedure SCROLL to generate the following:



Displaying a Previously Run Report

You can display a previously run report in a browser without re-executing the request. WebFOCUS holds a report in a cache for a set period of time. If the output for a new request is the same as the output for a previous request, and it is still in the cache, the browser displays the previous report with the *Back*, *Refresh*, or *Reload* button.

The EXPIRE_REPORTS parameter sets the period of time for which a report is held in the cache. It is customizable for a WebFOCUS installation and affects all WebFOCUS output. You can change the EXPIRE_REPORTS parameter in the following ways:

- Set the EXPIRE_REPORTS value in the CGIVARS.WFS file, which is located by default as follows:

  Windows: *install_ drive:*\ibi\WEBFOCUS82\client\conf\etc

UNIX: /ibi/WEBFOCUS82/client/conf/etc

z/OS: /ibi/WEBFOCUS82/client/conf/etc

- Add EXPIRE_REPORTS and its revised value to the SITE.WFS file, which overrides values in other .WFS files.

Syntax: How to Set Report Duration

```
EXPIRE_REPORTS = {n|300}
```

where:

n

Is the number of seconds for which a report is held in the cache. The default value is 300.

To ensure that a report is re-executed, set EXPIRE_REPORTS to 1.

To view the cached output of a browser using the *Back*, *Refresh*, or *Reload* button, set EXPIRE_REPORTS to a large number, such as 4,000,000,000 seconds.

Passing a User ID From HTML for a Custom Menu

You can capture a user ID on an HTML logon page and pass it to a procedure. Use this technique to create menus that display only those application functions that a user is permitted to execute. For instance, one user may access menu options that apply to a particular role in the organization, and another user may access certain sensitive capabilities. The choice of menu options is based on the user ID.

The procedure in this topic runs on WebFOCUS for Windows.

Customizing a Menu

The following is a summary of steps that you will complete in the example. You can modify the steps for your own application requirements. The specified file locations are the Windows file locations used in this procedure.

Step		File Name	File Location
1	Create an HTML logon page. A cookie is created after successful logon, containing the user ID (IBIC_user) entered on the page.	SIGNON.HTM	WebFOCUS82\ibi_apps\ibi_html
2	Create a home page with a frameset that displays custom menu options and a report when run.	HOME.HTM	WebFOCUS82\ibi_apps\ibi_html
3	Create the frame pages.	MENU.HTMLOADMEM.HTMWELCOME.HTM	WebFOCUS82\ibi_apps\ibi_html
4	Modify the file CGIVARS.WFS to assign the value of the user ID to a Dialogue Manager amper variable.	CGIVARS.WFS	ibi\WEBFOCUS82\client\conf\etc

Step		File Name	File Location
5	Create a user data source that contains user names and their allowed menu options.	USERLST.MASUSERLST.DAT	srv82\ggdemo
6	Create a menu data source that identifies and supplies a label for all available menu options, the location of the file accessed by an option, and the target frame for the output of an option. You can customize the contents of this file for your application.	MENULST.MASMENULST.DAT	srv82\ggdemo
7	Create a procedure that generates a custom menu for a user.	FHSUB.FEX	srv82\ggdemo
8	Create an HTML display page on which the custom menu is displayed.	SUBMENU.HTM	srv82\ggdemo
9	Create the forms that call the report selections on the menu.	1MYGRAPH.THM2MYGRAP.HTM	WebFOCUS82\ibi_apps\ibi_html

Step		File Name	File Location
10	Create the procedures that generate the user-specific reports.	FHSUB.FEX	srv82\ggdemo

Note:

- Procedures used by the WebFOCUS Reporting Server to create output, and HTM files in which the output is embedded, must be accessible to the WebFOCUS Reporting Server.
- Launch pages, menus that request reports, and images are stored in the WebFOCUS Client on the web server.

Procedure: How to Create the HTML Logon Page (Step 1)

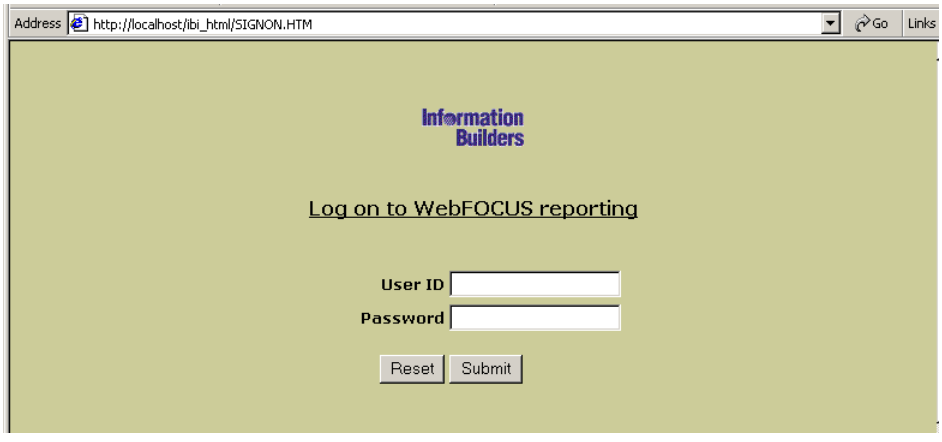
The following file is named SIGNON.HTM. Modify the lines in bold to apply to the web server at your site.

SIGNON.HTM:

```
<html>
<head>
<meta http-equiv="Content-Language" content="nl-be">
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1252">
<meta name="GENERATOR" content="Microsoft FrontPage 4.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<title>Please Identify yourself!</title>
</head>
<body stylesrc="submenu.htm" bgcolor="#CECF9C">
<p> </p>
<form method="POST" action="/ibi_apps/WFServlet"
    name="Signon">
  <table border="0" width="100%" height="112">
    <tr>
      <td width="66%" height="19" align="right" colspan="2">
        <p align="center"><img border="0"
           src="http://localhost/ibi_apps/ibi_html/IB_LOGO.GIF"></p>
        <p align="center"> </td>
    </tr>
```

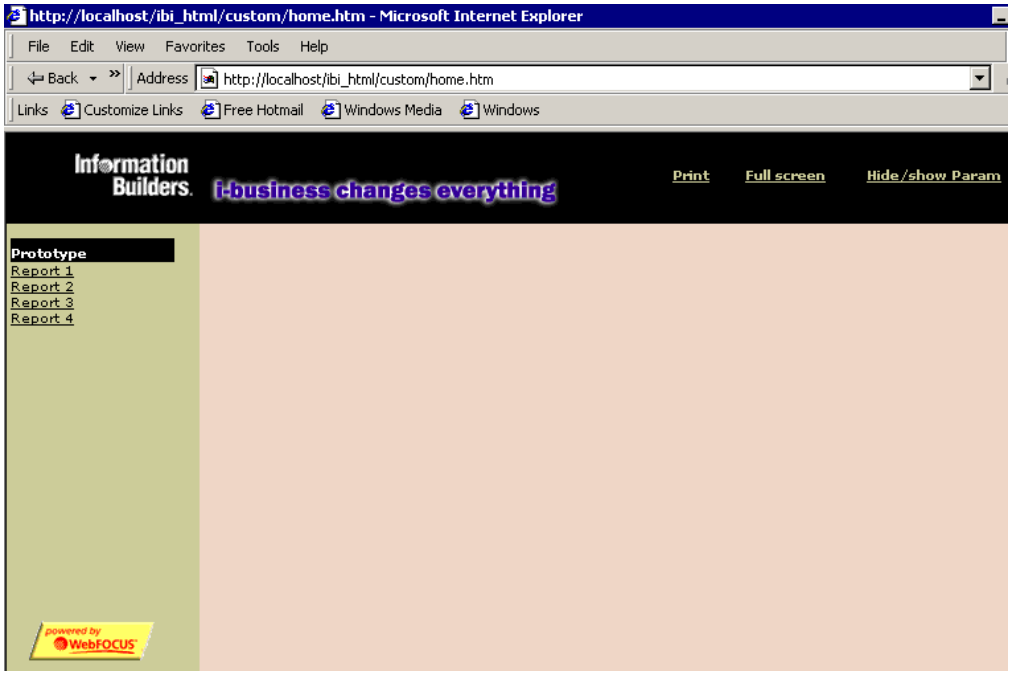
```
    <tr>
      <td width="66%" height="19" align="right" colspan="2">
       <p align="center"><b><u><font face="Verdana" size="3">Log on to
          WebFOCUS reporting</font></u></b></p>
        <p align="center"> </td>
    </tr>
    <tr>
      <td width="31%" height="19" align="right"><font face="Verdana"
        size="2"><b>User ID</b></font></td>
      <td width="35%" height="19"><input type="text" name="IBIC_user"
        size="20"></td>
    </tr>
    <tr>
      <td width="31%" height="18" align="right"><font face="Verdana"
        size="2"><b>Password</b></font></td>
      <td width="35%" height="18"><input type="password"
        name="IBIC_pass" size="20"></td>
    </tr>

  <tr>
      <td width="31%" height="57">
        <p align="right"><input type="reset" value="Reset"
          name="B2"></td>
      <td width="35%" height="57"><input type="submit" value="Submit"
        name="B1"></td>
    </tr>
  </table>

  <p> </p>
  <p> </p>
  <p> </p>
  <p> </p>
  <p> </p>
  <input type="hidden" name="IBIWF_action" value="WF_SIGNON">
  <input type="hidden" name="IBI_random" value=''>
  <input type="hidden" name="WF_SIGNON_MESSAGE"
  value="http://localhost/ibi_apps/IBI_HTML/HOME.HTM">
 </form>
<p> </p>
</body>
</html>
```

The following is the SIGNON.HTM page when accessed in the web browser.



Procedure: How to Create the Home Page (Step 2)

In this example, you create the home page with a frameset. The following HTML files supply the content for each frame used: MENU.HTM (banner frame), LOADMEN.HTM (contents frame), and WELCOME.HTM (query frame).



The following file is named HOME.HTM. Modify the lines in bold to apply to the web server at your site.

HOME.HTM:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
   charset=windows-1252">
<title>Webfocus Reporting</title>
<meta name="GENERATOR" content="Microsoft FrontPage 4.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<meta name="Microsoft Border" content>
<meta base href="http://localhost/">
</head>

<frameset id="header" rows="68,*" framespacing="0" border="0"
   frameborder="0">
<frame name="banner" scrolling="no" noresize target="contents"
   src="MENU.HTM" marginwidth="0" marginheight="0" NOBORDER>
 <frameset id="bodypart" cols="146,*">
 <frame name="contents" target="main" src="LOADMEN.HTM"
    scrolling="auto" marginwidth="3" marginheight="11">
   <frameset id="querypart" rows="80,*">
   <frame name="QueryFrame" src="WELCOME.HTM" target="bottom"
      marginwidth="0" marginheight="0" scrolling="no" noresize>
   <frame name="ReportFrame" src="WELCOME.HTM" scrolling="auto"
      noresize marginwidth="12" marginheight="1" target="_self">
   </frameset>
 </frameset>
 <noframes>
  <body>
  <p>This page uses frames, but your browser doesn't support them.</p>
  </body>
 </noframes>
</frameset>
</html>
```

Procedure: How to Create the Frame Pages (Step 3)

1. The following file is named MENU.HTM. It supplies the content for the top frame (banner). Modify the lines in bold to apply to the web server at your site.

MENU.HTM:

```
<HTML>
<HEAD>
<meta base href="http://localhost/ibi_apps/ibi_html" target="contents">
<TITLE>The JavaScript Menu Object</TITLE>
<script language="Javascript"><!---
function doprint()
{
 parent.document.frames.bodypart.all("Reportframe").focus();
 window.print();
}
function Show_query()
{
 if (parent.frames('QueryFrame').document.title == 'Query')
 {
  x=parent.frames('QueryFrame').document.all('Query').offsetHeight-10;
  parent.document.all("header").all("bodypart").all("QueryPart")
  .rows=x+",*";
 }
}

function Hide_A_query()
{
 if (parent.document.all("header").all("bodypart").all("QueryPart")
  .rows =='0,*')
 {
  resize()
 }
 else
 {
  parent.document.all("header").all("bodypart").all("QueryPart")
  .rows='0,*';
 }
}

function resize(){
var
x=parent.frames('QueryFrame').document.all('Query').offsetHeight-10;
parent.document.all("header").all("bodypart").all("QueryPart")
.rows=x+",*";
}
```

The following code validates the query.

```
function dovalidation(Checktype){
  var boodschap=''
  var frm=parent.frames('QueryFrame').frminput

//Check Required !!!
  for (x=0;x<Checktype.length;x++) {
  if (Checktype[x][2]=='R') {
     if (frm.elements(Checktype[x][0]).value == '') {
       boodschap=boodschap + '"'+Checktype[x][1]+'" is mandatory ! \n'
   }  }   }
//End Check Required
  if (boodschap != '') {return boodschap};
//Check Type !!!
  for (x=0;x<Checktype.length;x++)
  {
    if (Checktype[x][3]=='N')
    {
      var anum=/(^\d+$)|(^\d+\,\d+$)/
      if (anum.test(frm.elements(Checktype[x][0]).value))
      {
    if ((Checktype[x][4] > frm.elements(Checktype[x][0]).value) ||
       (Checktype[x][5] < frm.elements(Checktype[x][0]).value))
    {
    boodschap=boodschap+'"'+Checktype[x][1]+'" must be within range
    ('+Checktype[x][4]+'-'+Checktype[x][5]+')! \n';
    }
    else
    {}
      }
      else
      {
      boodschap=boodschap+'"'+Checktype[x][1]+'" must be a number !
       \n';
      }
    }
    else
    {
      if (Checktype[x][3]=='T')
      {
    if (frm.elements(Checktype[x][0]).value.length < Checktype[x][4]
     ||
      frm.elements(Checktype[x][0]).value.length > Checktype[x][5])
```

```
{
boodschap=boodschap+'"'+Checktype[x][1]+'"length must be within
    range ('+Checktype[x][4]+'-'+Checktype[x][5]+')! \n';
    }

      }
      else
      {

if((Checktype[x][3]=='D')&&(frm.elements(Checktype[x][0]).value.length
> 0))
    {
    var dateStr = frm.elements(Checktype[x][0]).value;
    var datePat = /^(\d{1,2})(\/|-)(\d{1,2})\2(\d{2}|\d{4})$/;
    // To require a 4 digit year entry, use this line instead:
    // var datePat = /^(\d{1,2})(\/|-)(\d{1,2})\2(\d{4})$/;

    var matchArray = dateStr.match(datePat);
     if (matchArray == null) {
       boodschap=boodschap+'"'+Checktype[x][1]+'" is not a valid date!
         \n';
       return boodschap;
     }
     day = matchArray[1];
     month = matchArray[3];
     year = matchArray[4];
     if (month < 1 || month > 12) {
       boodschap=boodschap+'"'+Checktype[x][1]+'" :month must be
         between 1 and 12.';
       return boodschap;
     }
     if (day < 1 || day > 31) {
       boodschap=boodschap+'"'+Checktype[x][1]+'" :day must be between
         1 and 31.';
       return boodschap;
     }
     if ((month==4 || month==6 || month==9 || month==11) && day==31) {
       boodschap=boodschap+'"'+Checktype[x][1]+'" : month '+month+'
         does not have 31 days!';
       return boodschap;
     }
     if (month == 2) {
       var isleap = (year % 4 == 0 && (year % 100 != 0 || year % 400
         == 0));
       if (day>29 || (day==29 && !isleap)) {
       boodschap=boodschap+'"'+Checktype[x][1]+'" : February ' + year
         + 'does not have ' + day + ' days!';
       return boodschap;
     }
```

```
    }
      }
    }
  }
 }
  return boodschap
}

function breakout() {
  mywindow=window.open("","Titel","scrollbars=yes,status=no");
  mywindow.location.href=parent.frames('ReportFrame').location.href;
}
//-->
</script>
<style fprolloverstyle>A:hover {color: #FFFF00; font-family: Verdana;
 font-size:8pt; font-weight: bold}
</style>
<base target="contents">
</HEAD>

<BODY style="font-family: Verdana; font-size: 8pt"
  bgcolor="#000000" link="#CECF9C" vlink="#CECF9C" alink="#CECF9C">

<table border="0" width="600" bgcolor="#000000" cellspacing="0"
  cellpadding="0" bordercolor="#000000" height="62">
  <tr>
    <td> </td>

    <td width="57" align="center" height="1">
      <p align="center"><font color="#CECF9C" size="1">
        <b>
        <a href='javascript:void 0' onclick='doprint();'>Print</a>
        </b></font>
    </td>

    <td width="86" height="1" align="center">
      <p align="center"><font color="#CECF9C" size="1">
      <b><a href='javascript:void 0' onclick='breakout();'>
         Full screen</a></b></font>
    </td>

    <td width="138" align="center" height="1">
      <p align="center"><font color="#CECF9C" size="1">
      <b><a href='javascript:void 0'onclick='Hide_A_query();'>
          Hide/show Param</a></b></font>
    </td>
  </tr>
</table>
</BODY>
```

2. The following file is named LOADMEN.HTM. It supplies the content for the left-hand frame (contents). Modify the lines in bold to apply to the web server at your site.

   LOADMEN.HTM:

```
<html>
<head>
<title>Welcome text</title>
<SCRIPT LANGUAGE="JavaScript">
<!--
function loadmenu() {
IBI_random=Math.random()*Math.random()*Math.random()*100 ;
loadurl="http://localhost/ibi_apps/
WFServlet?IBIF_ex=fhsub&IBI_random="+IBI_random
parent.frames[1].location.href=loadurl
}
// -->
</SCRIPT>
</head>
<body bgcolor="#E0E0BE" onload="loadmenu()">
<p align="center">Loading Menu ....</p>
<p align="center">Stand by ..</font></p>
</SCRIPT>
</body>
</html>
```

3. The following file is named WELCOME.HTM. It provides the initial blank frame that will be replaced by report output. No modification is required.

   WELCOME.HTM:

```
<html>
<head>
<meta http-equiv="Content-Language" content="nl-be">
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1252">
<meta name="GENERATOR" content="Microsoft FrontPage 4.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<title>Welcome text</title>
<base target="_self">
</head>
<body bgcolor="#E0E0BE"></body>
</html>
```

Procedure: How to Modify the SITE.WFS Files (Step 4)

The following is an example of modifying the SITE.WFS files.

1. Open the file SITE.WFS, located by default as follows

   Windows: drive:\ibi\WEBFOCUS82\client\conf\etc

   UNIX: /ibi/WEBFOCUS82/client/conf/etc

   z/OS: /ibi/WEBFOCUS82/client/conf/etc

   Note: At many locations, developers will not have access to the .WFS files on the web server. Contact your systems administrator or other appropriate staff.

2. In IBICOMMD.WFS, add the following lines after the code <ifndef>EDACS3... <endif>:

```
<sendvar>
RUSER=&IBIC_user
<endsendvar>
```

   If you are modifying SITE.WFS, add these lines after the comment.

   This step will send the user ID to a Dialogue Manager variable named &RUSER, which will be used for selecting menu options in the FHSUB procedure.

3. Save the file and exit.

Procedure: How to Create the User Data Source (Step 5)

The following Master File for the user data source is named USERLST.MAS. This file must be created in the WebFOCUS Reporting Servers search path, either the APP PATH or EDAPATH in EDASPROF.PRF. For details, see *WebFOCUS Application Logic* on page 27.

```
FILENAME=USERLST, SUFFIX=FIX,
 SEGNAME=USERLST,SEGTYPE=S0,$
   FIELD=USER ,ALIAS=USER, USAGE=A8  , ACTUAL=A8  ,$  user ID
   FIELD=MKEY ,ALIAS=MKEY, USAGE=A2  , ACTUAL=A2  ,$  menu key
   FIELD=DUMM ,ALIAS=DUMM, USAGE=A70 , ACTUAL=A70 ,$  filler
```

The data source is USERLST.DAT:

```
CEO     01
ADMIN   02
```

The user ID CEO has access only to menu option 01. The user ID ADMIN has access to menu option 02.

Procedure: How to Create the Menu Data Source (Step 6)

The following Master File for the menu data source is named MENULST.MAS:

```
FILENAME=MENULST, SUFFIX=FIX,
 SEGNAME=TCSTAB04,SEGTYPE=S0,$
  FIELD=MKEY , ALIAS=MKEY,  USAGE=A2  ,  ACTUAL=A2  , INDEX=I ,$ menu key
   FIELD=RAP  , ALIAS=RAP ,  USAGE=A25 , ACTUAL=A25 ,$  menu text
   FIELD=HTM  , ALIAS=HTM ,  USAGE=A35 , ACTUAL=A35 ,$  HTML page
   FIELD=FRM  , ALIAS=FRM ,  USAGE=A12 , ACTUAL=A12 ,$  target frame
   FIELD=DUMM , ALIAS=DUMM ,  USAGE=A6  , ACTUAL=A6  ,$   filler
```

The data source is MENULST.DAT:

```
01REPORT1   ibi_apps/ibi_html/1myqrap.htm    QueryFrame
02REPORT2   ibi_apps/ibi_html/2myqrap.htm    QueryFrame
```

Procedure: How to Create the Procedure (Step 7)

The following file is named FHSUB.FEX. The JOIN command associates users with their permitted menu options. The DEFINE command dynamically defines the hyperlinks that will appear on the custom menu.

The value for the Dialogue Manager variable &RUSER is passed to the procedure by the <sendvar> block in the file SITE.WFS.

Modify the lines in bold to apply to your site.

```
-*********************************************************************
-* Calling from : LOADMEN.htm
-* Files called : sub.htm
-* Used &VARS   : &RUSER -> Web user ID (IBIC_user or equivalent)
-* Files used   : MENULIST.DAT -> fixed file with menu items
-*                USERLST.DAT  -> fixed file with users and privileges
-* Files created: rlist.htm -> temporary file with menu list
-*                H1.FOC    -> temporary file used in JOIN
-*********************************************************************
-*********************************************************************
-* Change these for customization:
-* &HOMEURL : web server URL
-* &LSTDIR  : Location of userlst.dat and menulst.dat
-* &LSTDIR  : Location of rlist.htm
-*********************************************************************
-*SET &ECHO=ALL;
-SET &HOMEURL='HTTP://localhost/';
-SET &LSTDIR='drive\IBI\APPS\GGDEMO\';
-SET &EDAHTMDIR='drive\IBI\APPS\GGDEMO\';
-SET &USERDAT='&LSTDIR.EVAL' || 'userlst.dat' ;
-SET &MENUDAT='&LSTDIR.EVAL' || 'menulst.dat' ;
-SET &LISTLOC='&EDAHTMDIR.EVAL' || 'rlist.htm' ;
-SET &RUSERL=&RUSER.LENGTH;
-SET &OLENGTH='A' || &RUSERL.EVAL;
-SET &UPUSER=UPCASE(&RUSERL.EVAL,'&RUSER.EVAL','&OLENGTH.EVAL');
FILEDEF USERLST DISK &USERDAT.EVAL
FILEDEF MENULST DISK &MENUDAT.EVAL
FILEDEF RLIST   DISK &LISTLOC.EVAL
-RUN
-*********************************************************************
-* This code creates temporary file containing all menu items
-* with index on MKEY fields needed in JOIN.
-*********************************************************************
TABLE FILE MENULST
PRINT *
ON TABLE HOLD AS H1 FORMAT FOCUS INDEX MKEY
END
-RUN
-*********************************************************************
-* This will build the HTML syntax needed for dynamic user menu
-* with selection on user ID.
-*********************************************************************
```

```
JOIN MKEY IN USERLST TO ALL MKEY IN H1 AS A
DEFINE FILE USERLST
TAGP1/A25='<TR> <TD WIDTH="100%" >' ;
TAGP2/A150=
 '<A HREF='  || '&HOMEURL.EVAL' || HTM ||
 ' TARGET="' || FRM || '" ONCLICK="Prepare();" >' ;
TAGP3/A80='<FONT SIZE="1" COLOR="#000000" >'|| RAP ||
'</FONT></A></TD></TR>';
END
TABLE FILE USERLST
PRINT TAGP1 TAGP2 TAGP3
WHERE USER EQ '&UPUSER.EVAL';
ON TABLE HOLD AS RLIST FORMAT ALPHA
END
-RUN
-*************************************************************************
-* This will call the HTM file that contains the !IBI.FIL syntax.
-*************************************************************************
-HTMLFORM SUBMEN
```

Procedure: How to Create the HTML Display Page (Step 8)

The HTML display page is called by the procedure, and must be accessible by the WebFOCUS Reporting Server. In Windows and UNIX, the extension is always .HTM.

The following file is named SUBMEN.HTM. It is merged with the report output generated by the procedure FHSUB. Modify the lines in bold to apply to the web server at your site.

```
<html>
<head>
<meta http-equiv="Content-Language" content="nl-be">
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1252">
<meta name="GENERATOR" content="Microsoft FrontPage 4.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<title>Submenu</title>

<SCRIPT LANGUAGE='JavaScript'>
<!-- shields up!

function Prepare () {
parent.document.all("header").all("bodypart").all("QueryPart")
 .all("Reportframe").src="welcome.htm"
parent.document.IBI_random=Math.floor((Math.random()*100000));
}

function Hide_query()
{
parent.document.all("header").all("bodypart").all("QueryPart")
 .rows="26,*";
parent.document.all("header").all("bodypart").all("QueryPart")
 .all('QueryFrame').src='welcome.htm';
}

function set_rand()
{
parent.document.IBI_random=Math.floor((Math.random()*100000));
}
//  -->
</SCRIPT>

<style fprolloverstyle>A:hover {color: #808000; font-size: 10pt;
  font-family: Verdana; font-weight: bold}
</style>
<meta base href="http://localhost/ibi_apps/ibi_html" target="main">
<base target="main">
</head>

<body style="font-family: Verdana; font-size: 8pt" bgcolor="#CECF9C"
  onload="set_rand()">
```

```
<table border="0" width="100%" cellspacing="0" cellpadding="0">
  <tr>
    <td width="100%" bgcolor="#000000">
      <p align="left"><b><font size="1" color="#FFFFFF">Prototype
        </font></b>
     </td>
  </tr>
!IBI.FIL.RLIST;
  <tr>
    <td width="100%"><font size="1" color="#CECF9C">.</font></td>
  </tr>
</table>

<p> </p>
<p> </p>
<p> </p>
<p> </p>
<p> </p>
<p> </p>
<p align="center"><img border="0"
   src="http://localhost/ibi_apps/ibi_html/template/powered.gif"
   align="baseline"></p>
</body>
</html>
```

Procedure: How to Create the Forms That Call the Report Selections on the Menu (Step 9)

1. The following file is named 1MYQRAP.HTM. No modification is required. The file must be stored in the WebFOCUS Client under the directory ibi\WebFOCUS82\ibi_apps\ibi_html.

   1MYQRAP.HTM:

   ```
   <html>
   <head>
   <title>Query</title>
   <meta name="Microsoft Border" content>
   </head>
   <body name="test" topmargin="1" style="font-family: Verdana;
     font-size:8pt" bgcolor="#D8D9B0" onload="resize()">

   <DIV  ID="Query" style="width: 713; height: 159">
   <script Language="JavaScript">
   <!--
   ```

```
function Form1_Validator(theForm){
  var boodschap = ''
  var terug = true
//
//Checktype[[Fieldname,Text, Required, Type,rangefrom,rangeto]]
//Type=(T,N,D), T -> Rangefrom & rangeto = length, N > range= real
     range !!! - Daterange not yet supported
//
  var Checktype=[[],[]];
//
// End of Validation definition
//
//
  parent.frames("Reportframe").document.write('<br>
  <p align="center"><b><font face="Verdana" size="2">Please
    Wait...<BR>WebFocus is working for you</font></b></p>');

fun1()
  return (terug)
}

function Hide_query()
{
parent.document.all("header").all("bodypart").all("QueryPart")
  .rows="26,*";
}

function resize(){
var x=document.all("Query").offsetHeight-10;
parent.document.all("header").all("bodypart").all("QueryPart")
  .rows=x+",*";
}

function checkaction(){
  if
  (parent.document.all("header").all("bodypart").all("QueryPart").rows
    == '26,*') {
    resize()
  }
  else {
  parent.document.all("header").all("bodypart").all("QueryPart")
    .rows='26,*';
  }
}

function fun1()
{
document.frminput.IBI_random.value=Math.floor((Math.random()*100000));
}
//--></script>
```

```
<form method="GET" action="/ibi_apps/WFServlet"
  style="font-weight:bold" name="frminput"
  onsubmit="return Form1_Validator(this)" target="ReportFrame">
<input type="hidden" name="IBIF_ex" value="REPORT1">
<input type="hidden" name="IBI_random" value="">

<table name="tble" border="0" width="759" cellspacing="1"
  style="font-family:Verdana; font-size: 8pt" height="94">
    <tr>
      <td width="759" align="right" bgcolor="#CFD09F" colspan="4"
          height="11">
      <p align="left"><font face="Verdana" size="1"><u><b>
        Please provide Parameters for Report 1 </b></u></font>
      </td>
    </tr>
    <tr>
      <td width="82" align="right" bgcolor="#D8D9B0" height="25">
        <font face="Verdana" size="1">Variable 1  </font>
      </td>
      <td width="212" bgcolor="#D8D9B0" height="25">
        <font face="Verdana" size="1">
        <select size="1" name="AMP_VAR1" style="font-family:Verdana;
           font-size: 8pt">
        <option value="ACTION">ACTION</option>
        <option value="MYSTERY">MYSTERY</option>
        <option value="CLASSIC">CLASSIC</option>
        <option value="CHILDREN">CHILDREN</option>
        <option value="MUSICALS">MUSICALS</option>
        </select></font>
      </td>
    <tr>
      <td width="70" align="right" bgcolor="#D8D9B0" height="1">
        <p></p>
      </td>
      <td width="104" bgcolor="#D8D9B0" height="27">
         <font face="Verdana" size="1">
         <input type="submit" value="Run" name="B1"></font>
      </td>
    </tr>
  </table>
</form>
</DIV>
</body>
</html>
```

2. The following file is named 2MYQRAP.HTM. No modification is required.

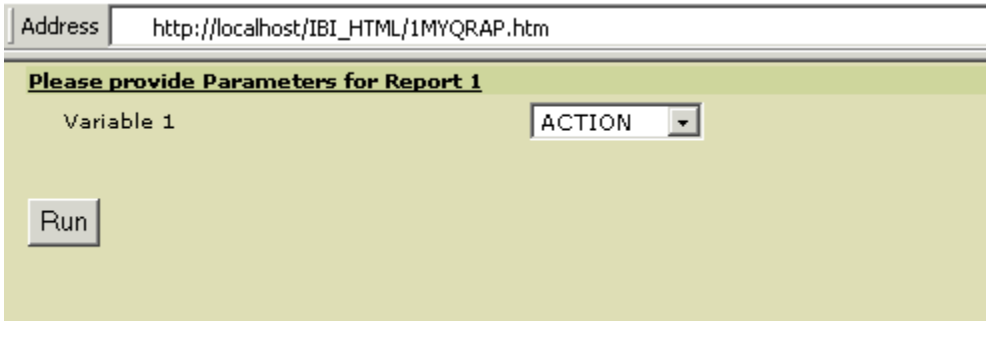
2MYQRAP.HTM:

```
<html>
<head>
<title>Query</title>
<meta name="Microsoft Border" content>
</head>

<body name="test" topmargin="1" style="font-family:Verdana;
  font-size:8pt" bgcolor="#D8D9B0" onload="resize()">

<DIV ID="Query" style="width: 713; height: 159">
<script Language="JavaScript">
<!--
function Form1_Validator(theForm){
  var boodschap = ''
  var terug = true
//
//Checktype[[Fieldname,Text, Required, Type,rangefrom,rangeto]]
//Type=(T,N,D), T -> Rangefrom & rangeto = length, N > range= real
    range !!! - Daterange not yet supported
//
  var Checktype=[[],[]];

 // End of Validation definition

  parent.frames("Reportframe").document.write('<br>
  <p align="center"><b><font face="Verdana" size="2">Please
    Wait...<BR>WebFocus is working for you</font></b></p>');

fun1()
  return (terug)
}

function Hide_query()
{
parent.document.all("header").all("bodypart").all("QueryPart")
  .rows="26,*";
}

function resize(){
var x=document.all("Query").offsetHeight-10;
parent.document.all("header").all("bodypart").all("QueryPart")
  .rows=x+",*";
}

function checkaction()
{
  if (parent.document.all("header").all("bodypart").all("QueryPart")
  .rows =='26,*') {
  resize()
  }
```

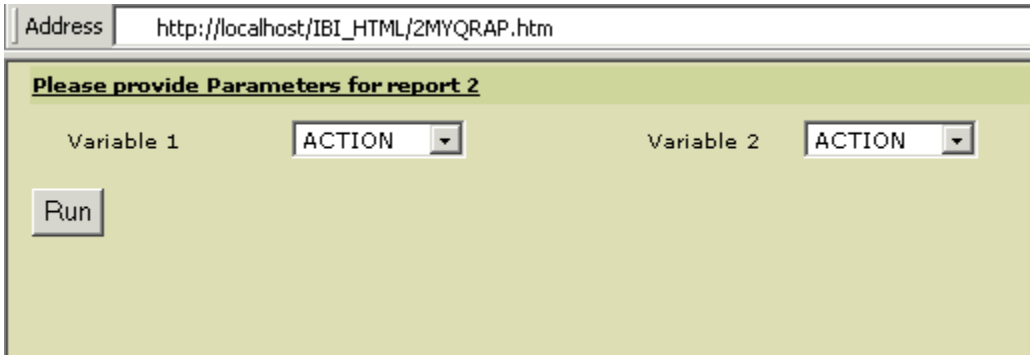
```
  else {
  parent.document.all("header").all("bodypart").all("QueryPart")
   .rows='26,*';
  }
}

function fun1()
{
document.frminput.IBI_random.value=Math.floor((Math.random()*100000));
}
//--></script>
<form method="GET" action="/ibi_apps/WFServlet"
  style="font-weight: bold" name="frminput"
  onsubmit="return Form1_Validator(this)" target="ReportFrame">
  <input type="hidden" name="IBIF_ex" value="REPORT2">
  <input type="hidden" name="IBI_random" value="">

<table name="tble" border="0" width="759" cellspacing="1"
  style="font-family:Verdana; font-size: 8pt" height="94">
   <tr>
      <td width="759" align="right" bgcolor="#CFD09F" colspan="4"
         height="11">
        <p align="left"><font face="Verdana" size="1">
           <u><b>Please provide Parameters for Report 2</b></u></font>
      </td>
    </tr>
    <tr>
      <td width="82" align="right" bgcolor="#D8D9B0" height="25">
      <font face="Verdana" size="1">Variable 1  </font></td>
```

```
        <td width="212" bgcolor="#D8D9B0" height="25">
           <font face="Verdana" size="1">
           <select size="1" name="AMP_VAR1" style="font-family:Verdana;
              font-size: 8pt">
            <option value="ACTION">ACTION</option>
            <option value="MUSICALS">MUSICALS</option>
            <option value="CLASSIC"> CLASSIC</option>
            <option value="CHILDREN">CHILDREN</option>
            <option value="FOREIGN">FOREIGN</option>
            <option value="MYSTERY">MYSTERY</option>
           </select></font></td>

        <td width="82" align="right" bgcolor="#D8D9B0" height="1">
        <font face="Verdana" size="1">Variable 2</font>
        <td width="212" bgcolor="#D8D9B0" height="25">
            <font face="Verdana" size="1">
        <select size="1" name="AMP_VAR2" style="font-family:Verdana;
            font-size: 8pt">
         <option value="ACTION">ACTION</option>
         <option value="MUSICALS">MUSICALS</option>
         <option value="CLASSIC"> CLASSIC</option>
         <option value="CHILDREN">CHILDREN</option>
         <option value="FOREIGN">FOREIGN</option>
         <option value="MYSTERY">MYSTERY</option>
        </select></font></td>
      </tr>
     <tr>
        <td width="70" align="right" bgcolor="#D8D9B0" height="1">
           <p></p>
         </td>
         <td width="104" bgcolor="#D8D9B0" height="27">
            <font face="Verdana" size="1">
           <input type="submit" value="Run" name="B1"></font></td>
      </tr>
     </table>
   </form>
   </DIV>
   </body>
   </html>
```

The following is the 1MYQRAP.HTM page when accessed in the web browser.



The following is the 2MYQRAP.HTM page when accessed in the web browser.



Procedure: How to Create the Procedures (Step 10)

The following file is named REPORT1.FEX.

```
DEFINE FUNCTION SUBTRACT (VAL1/D8, VAL2/D8)
SUBTRACT/D8.2 = VAL1 - VAL2;
END
TABLE FILE MOVIES
PRINT TITLE LISTPR IN 35 WHOLESALEPR AND COMPUTE
PROFIT/D8.2 = SUBTRACT(LISTPR,WHOLESALEPR);
BY CATEGORY
WHERE CATEGORY EQ '&AMP_VAR1';
ON TABLE HOLD FORMAT HTMTABLE AS REPORT1
END

-HTMLFORM BEGIN
<HTML>
<BODY>
<H2>MOVIE SALES PROFIT</H2>
<!--WEBFOCUS TABLE REPORT1>
<HR>
<P>Parameters selected </P>
<UL>
<LI> CATEGORY: &AMP_VAR1
<LI> CURRENT DATE : &DATEDMYY
<LI> CURRENT TIME : &TOD
<LI> CURRENT USER : &RUSER
</UL>
</BODY>
</HTML>
-HTMLFORM END
```

Note: The HTML comment line must be closed with the –> characters or a single > character and should not have any other HTML tags within it.

Chapter 5

Managing Flow of Control in an Application

Dialogue Manager is the part of the WebFOCUS language that controls the execution of your application components. You can dynamically control the execution of procedures, giving you flexibility in application design. Dialogue Manager also enables you to use variables in your procedures and supply values for them at run time.

This topic contains information for the following operating systems: Windows, z/OS, and UNIX.

In this chapter:

- Uses for Dialogue Manager
- Dialogue Manager Processing
- Creating a Dialogue Manager Procedure
- Customizing a Procedure With Variables
- Creating a Standard Quote-Delimited String
- Creating and Working With Variables
- Using Numeric Amper Variables in Functions
- Controlling the Execution of a Procedure
- Navigating a Procedure
- Enhancing an HTML Webpage With a Procedure
- Issuing Operating System Commands
- Controlling Passwords With Dialogue Manager
- Sending a Message to the Application
- Testing and Debugging a Dialogue Manager Procedure
- Dialogue Manager Syntax Reference

Uses for Dialogue Manager

Dialogue Manager is a part of the WebFOCUS language that allows you to control the flow of your application with the use of commands and variables. The following are some of the ways you can use Dialogue Manager:

- **Customizing a procedure with variables.** You can dynamically change the execution of a procedure by including variables, or retrieve information from the system. For details, see *Customizing a Procedure With Variables* on page 336. For information on creating the variables used in a procedure, see *Creating and Working With Local and Global Variables* on page 370.
- **Controlling the execution of a procedure.** You can control a procedure by controlling the order in which to execute commands, closing external files, and exiting or canceling a procedure. For details, see *Controlling the Execution of a Procedure* on page 389.
- **Navigating a procedure.** You can conditionally execute requests, repeat execution with program loops, or call another procedure. For details, see *Navigating a Procedure* on page 392.
- **Enhance an HTML webpage.** You can increase the functionality of your webpage by including HTML commands in your procedure. For details, see *Enhancing an HTML Webpage With a Procedure* on page 407.
- **Issuing operating system commands.** You can issue an operating system command to set up an environment in which a request must run. For details on issuing operating system commands, see *Issuing Operating System Commands* on page 417.
- **Controlling passwords.** You can directly assign and change passwords. For details, see *Controlling Passwords With Dialogue Manager* on page 420.
- **Sending a message to the application.** You can send a message to an application while a procedure is processing to explain the purpose of the procedure, display results, or present useful information. For details, see *Sending a Message to the Application* on page 420.

Reference: **Summary of Dialogue Manager Commands**

The following table summarizes commands available in Dialogue Manager. For detailed information about these commands, see the individual topics for each command.

Command	Description
`-*`	Signals a comment.
`-?`	Displays the value of local variables.
`-CLOSE`	Closes an external file on z/OS opened for reading or writing. An external file is a sequential file in the file system of the operating system.
`-DEFAULT` `-DEFAULTS`	Sets a variable to an initial value.
`-DEFAULTH`	Sets a hidden variable to an initial value.
`-DOS`	Executes a DOS operating system command.
`-EXIT`	Executes stacked commands and terminates the procedure.
`-GOTO`	Transfers control to a label.
`-HTMLFORM`	Sends one or more reports to the browser.
`-IF`	Determines the execution flow of a procedure based on the evaluation of an expression.
`-INCLUDE`	Calls another Dialogue Manager procedure.
`-label`	Is the target of a -GOTO or -IF.
`-MVS`	In WebFOCUS, executes a z/OS operating system command. Only works with the RUN command.
`-PASS`	Issues and controls passwords.
`-QUIT`	Terminates the procedure without executing stacked commands.

Command	Description
-QUIT FOCUS	Terminates a procedure and exits FOCUS.
-READ	Reads data from an external file.
-READFILE	Reads a file by first reading its Master File and creating Dialogue Manager amper variables based on the ACTUAL formats for each field in the Master File.
-REPEAT	Executes a loop.
-RUN	Executes stacked commands.
-SET	Sets a variable to a literal value or to a value computed in an expression.
-? SET *parameter* &myvar	Captures the value of a settable parameter in &myvar.
-TSO	In WebFOCUS, executes a TSO operating system command. Only works with the RUN command.
-TYPE	Sends a message to a client application.
-UNIX	In WebFOCUS, executes a UNIX operating system command.
-VMS	In WebFOCUS, executes a VMS operating system command.
-WINNT	Executes a Windows operating system command.
-WRITE	Writes data to an external file.

Dialogue Manager Processing

You can modify your application at run time with user input and environment conditions by using Dialogue Manager procedures, which include commands and variables. It is important to understand how Dialogue Manager processes commands and variables of an application, using a procedure.

Dialogue Manager executes a procedure as follows:

1. Dialogue Manager reads each line of the procedure one by one, and substitutes values for variables when encountered.

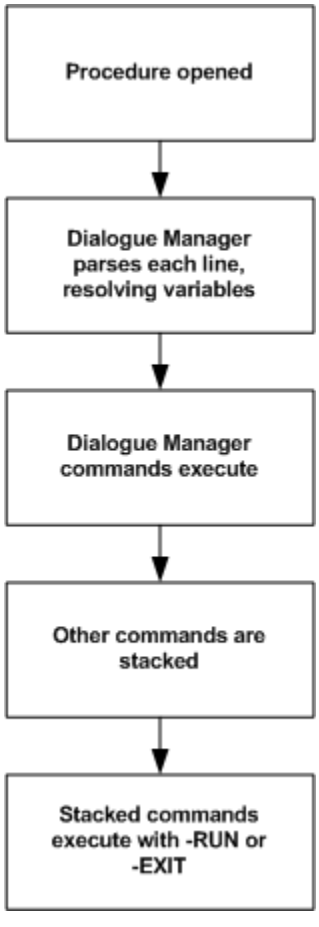
   You must make sure to assign values or identify defaults for all variables. Otherwise, you will receive an error message indicating a missing value.

2. All Dialogue Manager commands execute as soon as Dialogue Manager reads them.

   Other WebFOCUS commands are temporarily stored for subsequent execution, which are called *stacked* commands.

3. The -RUN and -EXIT commands execute any stacked commands.

The following diagram illustrates how a Dialogue Manager procedure is processed:



Example: A Procedure and Its Execution Process

The following is an example of a procedure, with an explanation of the way it processes.

The procedure contains the variable &PLANT. Assume that when the procedure executes it includes the variable value of BOS for &PLANT. The numbers on the left correspond to the notes explaining the code.

```
1. -IF &PLANT EQ 'DONE' THEN GOTO GETOUT;

2. TABLE FILE CENTHR
   HEADING
    "SALARY REPORT FOR LOCATION: &PLANT"
   PRINT SALARY POSITION
   BY LNAME
   WHERE PLANT IS '&PLANT'
   END

3. -RUN

4. -EXIT

   -GETOUT
   -TYPE NO PROCESSING DONE: EXITING
```

The procedure processes as follows:

1. The values for &PLANT is passed to the procedure before the first line executes. This can be done by entering values in the HTML interface, or by setting the values previously with other Dialogue Manager commands. Dialogue Manager substitutes the value BOS for the variable &PLANT in the first line, and tests for the value DONE. The test fails, so Dialogue Manager proceeds to the next line.

   If the value were DONE instead of BOS, control would pass to the label -GETOUT, and the message NO PROCESSING DONE: EXITING would be generated. Dialogue Manager would skip the report request.

2. The next six lines are the report request. Dialogue Manager scans them for the presence of variables, substituting BOS for &PLANT. As each line is processed, it is placed on a stack to be executed later.

   After Dialogue Manager processes the END command, the stacked commands look like this:

   ```
   TABLE FILE CENTHR
   HEADING
   "SALARY REPORT FOR LOCATION: BOS"
   PRINT SALARY POSITION
   BY LNAME
   WHERE PLANT IS 'BOS'
   END
   ```

The next line is then processed by Dialogue Manager.

To display populated variable values like those contained in this step, include the &ECHO variable in your procedure. For information on the &ECHO variable, see *Testing and Debugging a Dialogue Manager Procedure* on page 421.

3. -RUN causes the stacked commands to be executed. They are sent to be processed.
4. -EXIT terminates the procedure.

For complete information on using variables in procedures, see *Customizing a Procedure With Variables* on page 336.

Creating a Dialogue Manager Procedure

You can create a procedure with a system editor. When creating a procedure, you must follow these rules:

- Dialogue Manager commands begin with a hyphen in column 1. Other commands do not.
- At least one space must be inserted between the Dialogue Manager command and other text.
- If a Dialogue Manager command exceeds one line, the following line must begin with a hyphen (-) in the first position. The continuation line can begin immediately after the hyphen, or you may insert a space between the hyphen and the rest of the line.

Commenting a Procedure

It is good practice to include comments in a procedure for the benefit of others who may use it. It is particularly recommended that you use comments in a procedure heading to supply the date, the version, and other relevant information. A hyphen and an asterisk (-*) mark the beginning of a comment.

Reference: Adding a Comment in a Procedure

You can place a comment in a procedure in the following ways:

- Include text after the -*, optionally with a space before the text.
- Place a comment at the beginning or end of a procedure, or between commands. A comment cannot be on the same line as a command. The following is invalid:

```
-RUN -*Version 3 06/10/00
```

Example: Placing a Comment in a Procedure

The following example places comments at the beginning of a procedure.

```
-* Version 1  08/26/02  HRINFO  Procedure
TABLE FILE CENTHR
   .
   .
   .
```

Customizing a Procedure With Variables

You can customize a procedure and control its execution with variables. This is done dynamically by supplying a value of a variable at run time. You can supply a value of a variable in the following ways:

- Directly in a procedure. The commands -DEFAULT, -SET, and -READ enable you to supply values directly in a procedure.
- When calling a procedure from another procedure. You can include the variable names and their corresponding values as parameters in an EXEC command that calls one procedure from another.
- When calling a procedure from a webpage. The webpage calls the WebFOCUS script from either a form or a hyperlink. If you use a form, the end user enters the values in the form. If you use a hyperlink, you include the name of the procedure and the variable values directly in the HTML <A HREF code. The value for each variable is included in the call to the WebFOCUS script.
- In a drill-down report. You can use a WebFOCUS StyleSheet to pass parameters.
- From an external file. You can supply values from an external file with the -READ command.

Dialogue Manager variables cannot be used to customize commands in a WebFOCUS Report StyleSheet.

For information on the types of variables and how to use them, see *Creating and Working With Local and Global Variables* on page 370.

Reference: Rules for Supplying Variable Values

The following rules apply to values for variables:

- The maximum length of a variable value is 32,000 characters when using the -TYPE or -WRITE command.

- If a value contains an embedded comma, equal sign, or blank, you must enclose the variable name in single quotation marks when you use it in an expression. For example, if the value for &CITY is NY, NY, you must refer to the variable as '&CITY' in any expression.
- After a value is supplied for a local or global variable, it is used throughout the procedure unless it is changed with a command, such as -SET or -READ.
- Variables that are SET with an IF THEN command require an explicit ELSE clause.
- You can populate a global variable each time a procedure is executed by including the variable in a WebFOCUS Reporting Server profile, such as EDASPROF.PRF.

Example: Customizing a Procedure With Variable Values

This example illustrates the use of the -DEFAULT and -SET commands to supply values for variables. The end user supplies the value B10 for &CODE1, B20 for &CODE2, and SMITH for &REGIONMGR as prompted by an HTML form.

Note: In this example, the output was formatted to eliminate page numbering and grids. For information on formatting your output, see the *Creating Reports With WebFOCUS Language* manual.

The numbers to the left of the example apply to the notes that follow:

```
1. -DEFAULT &VERB=SUM
2. -SET &CITY=IF &CODE1 GT 'B09' THEN 'STAMFORD' ELSE 'UNIONDALE';
3. -TYPE REGIONAL MANAGER FOR &CITY
    SET PAGE=OFF
4.      TABLE FILE SALES
        HEADING CENTER
        "MONTHLY REPORT FOR &CITY"
        "PRODUCT CODES FROM &CODE1 TO &CODE2"
        " "
        &VERB UNIT_SOLD AND RETURNS AND COMPUTE
        RATIO/D5.1 = 100 * (RETURNS/UNIT_SOLD);
        BY PROD_CODE
        IF PROD_CODE IS-FROM &CODE1 TO &CODE2
        FOOTING CENTER
5.      "REGION MANAGER: &REGIONMGR"
        "CALCULATED AS OF &DATE"
        ON TABLE SET STYLE *
        TYPE=REPORT, GRID=OFF,$
        END
6. -RUN
```

The procedure executes as follows:

1. The -DEFAULT command sets the value of &VERB to SUM.

2. The -SET command supplies the value for &CITY depending on the value the end user entered in the form for &CODE1. Because the end user entered B10 as the value for &CODE1, &CITY becomes STAMFORD.
3. When the end user runs the report, WebFOCUS writes a message that incorporates the value for &CITY:

   ```
   REGIONAL MANAGER FOR STAMFORD
   ```

   The output from -TYPE is not displayed on the HTML page. Instead, it is written to the HTML source file. To display the message, the end user must view the source file of the document.
4. The stack contains the following lines:

   ```
   TABLE FILE SALES
   HEADING CENTER
   "MONTHLY REPORT FOR STAMFORD"
   "PRODUCT CODES FROM B10 TO B20"
   " "
   SUM UNIT_SOLD AND RETURNS AND COMPUTE
   RATIO/D5.1 = 100 * (RETURNS/UNIT_SOLD);
   BY PROD_CODE
   IF PROD_CODE IS-FROM B10 TO B20
   FOOTING CENTER
   "REGION MANAGER: SMITH"
   "CALCULATED AS OF 07/08/93"
   END
   ```

5. The end user supplied the value for &REGIONMGR on the form. WebFOCUS supplies the current date at run time.
6. The -RUN command causes execution of all commands in the stack. The output from the report request is as follows:

   ```
   MONTHLY REPORT FOR STAMFORD
   PRODUCT CODES FROM B10 TO B20

   PROD_CODE                          UNIT_SOLD         RETURNS       RATIO

   B10                                      103              13        12.6

   B12                                       69               4         5.8

   B17                                       49               4         8.2

   B20                                       40               1         2.5

   REGION MANAGER: SMITH
   CALCULATED AS OF 07/08/93
   ```

Supplying a Default Variable Value

-DEFAULT commands set default values for local or global variables. Supplying default values to variables in a stored procedure helps ensure that it runs correctly.

Note that -DEFAULTS is a synonym for -DEFAULT.

You can issue multiple -DEFAULT commands for a variable. If the variable is global, these -DEFAULT commands can be issued in separate FOCEXECs. At any point before another method is used to establish a value for the variable, the most recently issued -DEFAULT command will be in effect.

However, as soon as a value for the variable is established using any other method (for example, by issuing a -SET command, retrieving a value input by the user, or reading a value from a file), subsequent -DEFAULT commands issued for that variable are ignored.

If you want to initialize a variable and prevent it from being used for WebFOCUS parameter prompting, you can use the -DEFAULTH command to initialize the variable. Variables initialized with -DEFAULTH, are not returned in the XML describe information used for parameter prompting. Since these variables are not displayed by the parameter prompting features, they are hidden from the user.

Syntax: How to Supply a Default Value

```
-DEFAULT[S] &[&]name=value , [&[&]name=value] [;]
```

where:

&name

Is the name of the variable.

value

Is the default value assigned to the variable.

;

Is an optional punctuation character.

Note: -DEFAULTS is a synonym for -DEFAULT.

Syntax: How to Supply a Default Value for a Hidden Variable

```
-DEFAULTH &[&]name=value , [&[&]name=value]
```

where:

&name

Is the name of the variable. This variable is not returned in the XML describe information used for parameter prompting and is, therefore, hidden from the user.

value

Is the default value assigned to the variable.

Example: Supplying a Default Value

In the following example, -DEFAULT sets a default value for &PLANT:

```
-DEFAULT &PLANT=BOS
TABLE FILE CENTHR
     .
     .
     .
```

Supplying Variable Values in an Expression

You can assign a value to the variable by computing the value in an expression or assigning a literal value to a variable with the -SET command. You can also use the IN FILE phrase to test whether a character value exists in a file and populate a variable with the result. The value of the variable is set to 1 if the test value exists in the file and 0 (zero) if it does not.

If you want to set a variable value to a number, the only supported characters you can use are numeric digits, a leading minus sign, and a period to represent following decimal places. These are the only valid characters that Dialogue Manager supports in a number, regardless of EDIT options or the value of CDN.

Syntax: How to Assign a Value in an Expression

```
-SET &[&]name= {expression|value};

-SET &[&]var3= &var1 IN FILE filename1 [OR &var2 IN FILE filename2 ...];
```

where:

&name

Is the name of the variable.

expression

Is a valid expression. Expressions can occupy several lines, so you should end the command with a semicolon.

value

Is a literal value, or arithmetic or logical expression assigned to the variable. If the literal value contains commas or embedded blanks, you must enclose the value in single quotation marks. If the value contains a single quote, place two single quotes where you want one to appear.

&[&]*var3*

Is a variable that is populated with the value 1 if the result of the expression on the right side of the equal sign is true, or with the value 0 if the result is false.

&*var1*

Is the variable that contains the value to be searched for in filename1.

&*var2*

Is the variable that contains the value to be searched for in filename2.

Reference: Usage Notes for IN FILE

- The result of the IN FILE phrase is an alphanumeric value (1 or 0) that can be used in a logical expression connected with AND and OR operators within same –SET command. This value cannot be used as an argument in an alphanumeric operation such as concatenation within the same -SET command.
- In order for IN FILE to return the value 1, the values in the file and the search string must match exactly, starting with the leftmost byte in the file.
- The file can be in any order and have duplicate values. The search stops when either the first match is found or the end of the file is reached. If the file is allocated but does not exist, the value 0 is returned. If the file is not allocated, a FOC351 message displays in the View Source.
- The length of the variable used in the IN FILE phrase determines the number of bytes from the beginning of each record in the file used for comparison. Only an exact match on that number of bytes will return a 1. Trailing blanks in the variable will require the same number of trailing blanks in the file in order to match. For example, the following will match only the value 'ABC ' (ABC with three trailing blanks):

```
-SET &VAR1 = 'ABC   ';
-SET &VAR2 = &VAR1 IN FILE FILE1;
```

Syntax: How to Specify Precision for Dialogue Manager Calculations

The DMPRECISION setting enables Dialogue Manager -SET commands to display and store accurate numeric variable values.

Without this setting, results of numeric calculations are returned as integer numbers, although the calculations themselves employ double-precision arithmetic. To return a number with decimal precision without this setting, you have to enter the calculation as input into function FTOA, where you can specify the number of decimal places returned.

The SET DMPRECISION command gives users the option of either accepting the default truncation of the decimal portion of output from arithmetic calculations, or specifying up to nine decimal places for rounding.

```
SET DMPRECISION = {OFF|n}
```

where:

OFF

Specifies truncation without rounding after the decimal point. OFF is the default value.

n

Is a positive number from 0-9, indicating the point of rounding. Note that n=0 results in a rounded integer value.

Note:

- When using SET DMPRECISION, you must include -RUN after the SET DMPRECISION command to ensure that it is set prior to any numeric -SET commands.
- As the actual conversion to double precision follows the rules for the operating system, the values may vary from platform to platform.

Example: Using SET DMPRECISION

The following table below shows the result of dividing 20 by 3 with varying DMPRECISION (DMP) settings:

SET DMPRECISION =	Result
OFF	6
0	7

SET DMPRECISION =	Result
1	6.7
2	6.67
9	6.666666667

Example: Setting a Variable Value in an Expression

In the following example, -SET assigns the value 14Z or 14B to the variable &STORECODE, as determined by the logical IF expression. The value of &CODE is supplied by the end user.

```
-SET &STORECODE = IF &CODE GT C2 THEN '14Z' ELSE '14B';
   TABLE FILE SALES
   SUM UNIT_SOLD AND RETURNS
   BY PROD_CODE
   IF PROD_CODE GE &CODE
   BY STORE_CODE
   IF STORE_CODE IS &STORECODE
   END
```

Example: Testing Whether a Variable Value Is in a File

The following FOCEXEC creates an alphanumeric HOLD file called COUNTRY1 with the names of countries from the CAR file. It then sets the variable &C equal to FRANCE. The IN FILE phrase returns the value 1 to &IN1 if FRANCE is in the HOLD file and 0 if it is not:

```
TABLE FILE CAR
PRINT COUNTRY
ON TABLE HOLD AS COUNTRY1 FORMAT ALPHA
END
-RUN
-SET &C = 'FRANCE';
-SET &IN1 = &C IN FILE COUNTRY1;
-TYPE THE VALUE IS &IN1
```

The output shows that FRANCE is in the file COUNTRY1:

```
THE VALUE IS 1
```

Supplying Variable Values From Another Procedure

You can supply values for variables in an EXEC command used in a procedure to call another procedure. See *Calling Another Procedure With EXEC* on page 406 for more information on the use of this technique to control application flow.

Values are assigned using a name and value pair, and multiple pairs must be separated by commas. If the list of pairs exceeds the maximum width of the line, insert a comma as the last character on the line and continue the list on the following line. You do not have to enter the pairs in the order in which they are encountered in the procedure.

With EXEC you can supply values for some, but not all, variables used in the procedure. In that case, values not provided in the EXEC command must be supplied using another Dialogue Manager command. For an example, see *Supplying Variable Values From Another Procedure* on page 343.

If the variable is positional (that is, if it is a numbered variable), you do not need to specify the variable name in the EXEC command. WebFOCUS matches the EXEC values to the positional variables as they are encountered in the procedure. Therefore, it is critical to enter the appropriate value for each variable in order.

Reference: Rules for Using Named and Positional Variables With EXEC

You can mix named and positional variables freely in the EXEC command by following these rules:

- Names must be associated with values for named variables.
- Values for positional variables must be supplied in the order that those variables are numbered within the procedure.

The following command is valid for named variables and positional variables:

```
EX HRINFO SALARY=72000, 5, BOS, STATUS=EMPLOYED
```

Syntax: How to Supply a Variable Value With Another Procedure

```
EX[EC] procedure name=value
```

where:

procedure

Is the name of the procedure that will contain the name/value values.

name

Is the variable name.

value

Is the value you are giving to the variable.

Note: When EXEC is used in Managed Reporting, it is important to note that there is a difference between EX and EXEC. EX statements coded in a procedure are processed by the WebFOCUS Client, which looks for the procedure in the Managed Reporting repository. Procedures that are referenced with an EXEC statement are not processed by the WebFOCUS Client, they are only passed to the WebFOCUS Reporting Server for processing, and these procedures are not looked for in the Managed Reporting repository.

Example: Supplying Values With Another Procedure

Consider the following procedure named SLRPT:

```
TABLE FILE SALES
HEADING CENTER
"MONTHLY REPORT FOR &CITY"
SUM UNIT_SOLD AND RETURNS AND COMPUTE
RATIO/D5.2 = 100 * (RETURNS/UNIT_SOLD);
BY PROD_CODE
IF PROD_CODE IS-FROM &CODE1 TO &CODE2
BY CITY
IF CITY EQ &CITY
END
```

You can call this procedure from another procedure and supply values for the variables as parameters using the EX command as follows:

```
EX SLRPT CITY=STAMFORD, CODE1=B10, CODE2=B20
```

Example: Using Positional Variables

Consider the following example:

```
TABLE FILE SALES
HEADING CENTER
"MONTHLY REPORT FOR &1"
SUM UNIT_SOLD AND RETURNS AND COMPUTE
RATIO/D5.2 = 100 * (RETURNS/UNIT_SOLD);
BY PROD_CODE
IF PROD_CODE IS-FROM &2 TO &3
BY CITY
IF CITY EQ &1
END
```

The EX command that calls the procedure is as follows:

```
EX SLRPT STAMFORD, B10, B20
```

This command will substitute STAMFORD for the first positional variable, B10 for the second, and B20 for the third.

Reading Variable Values From and Writing Variable Values to an External File

You can read variable values from an external file, or write variable values to an external file with the -READ and -WRITE commands.

- You can supply variable values with the -READ command. For example, an external file may contain the start and end dates of a reporting period. Dialogue manager can read these values from an external file and use them a variable in a WHERE command that limits the range of data selected in a report request.
- You can save variable values in an external file with the -WRITE command. For example, a request can store the summed total of sales for the day in an external file so that it can be compared to the total sales of the following day.

The external file can be a fixed-format file (in which the data is in fixed columns) or a free-format file (in which the data is comma delimited).

When using a -READ or -WRITE command, the external file must be included in a FILEDEF command in your procedure. A -RUN command must then separate the FILEDEF command and the -READ or -WRITE command.

You can also read a file using the -READFILE command. The -READFILE command reads a file by first reading its Master File and creating Dialogue Manager amper variables based on the ACTUAL formats for each field in the Master File. It then reads the file and, if necessary, converts the fields from numeric values to alphanumeric strings before returning them to the created variables. Display options in the USAGE formats are not propagated to the variables. The names of the amper variables are the field names prefixed with an ampersand (&).

Syntax: How to Retrieve a Variable Value From an External File

```
-READ filename[,] [NOCLOSE] &name[.format.][,][&name][.format.]
```

where:

filename[,]

Is the name of the external file, which must be defined to the operating system. A space after *filename* denotes a fixed-format file, while a comma denotes a free-format file.

- On UNIX and Windows platforms, a FILEDEF for the external file is required.
- On z/OS, the external file must be allocated in the JCL or dynamically allocated by WebFOCUS with the ALLOCATE command.

`NOCLOSE`

Keeps the external file open until the -READ operation is complete. Files kept open with NOCLOSE can be closed using the command -CLOSE *filename* or a subsequent -WRITE command.

`&name``[,]`

Is the variable name. For free-format files, you may separate the variable names with commas. If the list of variables is longer than one line, end the first line with a comma and begin the next line with a dash followed by a blank. For fixed-format files, including comma-delimited files, begin the next line with a dash, a blank, and a comma.

Free-format

```
-READ EXTFILE, &CITY, &CODE1,- &CODE2
```

Fixed-format

```
-READ EXTFILE &CITY.A8. &CODE1.A3.,- ,&CODE2.A3.
```

`.format.`

Is the format of the variable. For a free-format file, specifying this value is optional. For a fixed-format file, format is the length or the type and the length. The type is either A (alphanumeric), which is the default, or I (numeric). The format value must be delimited by periods. The format is ignored for comma-delimited files.

Note: Instead of using *.format.*, you can specify the length of a variable using -SET and enclosing the corresponding number of blanks in single quotes. For example:

```
-SET &CITY='         ';
-SET &CODE1='    ';
-SET &CODE2='    ';
```

Syntax: How to Write a Variable Value to an External File

```
-WRITE filename [NOCLOSE] text
```

where:

filename
Is the name of the file being written to or read from. On UNIX and Windows platforms, *filename* must specify the absolute path and file name for the called procedure.

NOCLOSE
Keeps the external file open until the -WRITE operation is complete. A file kept open with NOCLOSE can be closed using the command -CLOSE *filename*.

text

Is any combination of variables and text.

Example: Reading a Value From an External File

Assume that EXTFILE is a fixed-format file containing the following data:

```
STAMFORDB10B20
```

To detect the end of a file, the following code tests the system variable &IORETURN. When no records remain to be read, a value equal to zero will not be found.

```
-READ EXTFILE &CITY.A8. &CODE1.A3. &CODE2.A3.
-IF &IORETURN NE 0 GOTO RESUME;
   TABLE FILE SALES
   SUM UNIT_SOLD
   BY CITY
   IF CITY IS &CITY
   BY PROD_CODE
   IF PROD_CODE IS-FROM &CODE1 TO &CODE2
    END
-RESUME
 .
 .
 .
```

Example: Reading From and Writing to an External File

The following example reads from and writes to text files and uses FILEDEF commands in a procedure. The numbers on the left apply to the notes that follow. In the example, the user supplies a value for &CITY in an HTML form:

```
1. -TOP
2.    FILEDEF PASS DISK D:PASS.DAT
   -RUN
3. -WRITE PASS &CITY
      TABLE FILE SALES
      HEADING CENTER
      "LOWEST MONTHLY SALES FOR &CITY"
      " "
      PRINT DATE PROD_CODE
      BY LOWEST 1 UNIT_SOLD
      BY STORE_CODE
      BY CITY
      WHERE CITY EQ '&CITY'
      FOOTING CENTER
      "CALCULATED AS OF &DATE"
      ON TABLE SAVE AS INFO
      END
4. -RUN

5.  FILEDEF LOG DISK D:LOG.DAT
   -RUN
      MODIFY FILE SALES
      COMPUTE
      TODAY/I6=&YMD;
      CITY='&CITY';
      FIXFORM X5 STORE_CODE/A3 X15 DATE/A4 PROD_CODE/A3
      MATCH STORE_CODE DATE PROD_CODE
      ON MATCH TYPE ON LOG
      "<STORE_CODE><DATE><PROD_CODE><TODAY>"
      ON MATCH DELETE
      ON NOMATCH REJECT
      DATA ON INFO
      END
6. -RUN
      EX SLRPT3
7. -RUN
8. -GOTO TOP
9. -QUIT
```

The procedure SLRPT3, which is invoked from the calling procedure, contains the following lines:

```
10. -READ PASS &CITY.A8.
      TABLE FILE SALES
      HEADING CENTER
      "MONTHLY REPORT FOR &CITY"
      "LOWEST SALES DELETED"
      " "
      PRINT PROD_CODE UNIT_SOLD RETURNS DAMAGED
      BY STORE_CODE
      BY CITY
      WHERE CITY EQ '&CITY'
      FOOTING CENTER
      "CALCULATED AS OF &DATE"
      END
11. -RUN
```

The procedure processes as follows:

1. -TOP marks the beginning of the procedure.
2. The FILEDEF command defines a file named PASS.
3. The -WRITE command writes the value of &CITY to the text file named PASS. In this case, assume that the value written is STAMFORD.
4. The -RUN command executes the stacked report request. In this case, a text file named INFO is created with the SAVE command. This is a sequential file, containing the result of the report request.
5. The FILEDEF command defines a log file for the subsequent MODIFY request.
6. The -RUN statement executes the stacked MODIFY request. The data comes directly from the INFO file created in the prior report request and is entered using the FIXFORM statement. The product with the lowest value in UNIT_SOLD is deleted from the data source and logged to a log file.
7. The next -RUN command executes the procedure called SLRPT3.
8. The -GOTO TOP statement passes control to the start of the procedure.
9. -QUIT ends processing.
10. The -READ command reads the value for &CITY from the text file PASS. In this case, the value passed is STAMFORD.
11. The -RUN command executes the report request and control is returned to the calling procedure.

Syntax: How to Read Master File Fields Into Dialogue Manager Variables

```
-READFILE [app/]mastername
```

where:

app
Is the application directory in which the file resides.

mastername
Is the name of the Master File to be read.

Reference: Usage Notes for -READFILE

- A -RUN command does not close the file. You must issue a -CLOSE command to close the file. You must be careful not to delete, change, or re-allocate the file before closing it.
- If multiple fields have the same field name, only one variable is created, and it contains the value of the last field in the Master File.
- -READFILE does not work if the Master File contains DBA restrictions. The following message is generated:

```
(FOC339) DIALOGUE MANAGER -READ FAILED: CHECK FILEDEF OR ALLOCATION FOR:
-READFILE filename
```

- -READFILE is not supported with text fields. The following message is generated:

```
(FOC702) THE OPTION SPECIFIED IS NOT AVAILABLE WITH TEXT FIELDS:
fieldname
```

- -READFILE cannot read XFOCUS data sources. The file to be read must have an associated Master File.

Example: Reading Fields From a Data Source Into Dialogue Manager Variables Using -READFILE

The following request creates a binary HOLD file, then uses -READFILE to read the first record from the HOLD file and type the values that were retrieved into Dialogue Manager variables. Note that the names of the variables are the field names prefixed with an ampersand:

```
TABLE FILE EMPLOYEE
PRINT LAST_NAME FIRST_NAME DEPARTMENT CURR_SAL
BY EMP_ID
ON TABLE HOLD AS READF1 FORMAT BINARY
END
-RUN
-READFILE READF1
-TYPE LAST_NAME  IS &LAST_NAME
-TYPE FIRST_NAME IS &FIRST_NAME
-TYPE DEPARTMENT IS &DEPARTMENT
-TYPE CURR_SAL   IS &CURR_SAL
-TYPE EMP_ID     IS &EMP_ID
```

The output is:

```
>   NUMBER OF RECORDS IN TABLE=       12  LINES=     12

 HOLDING BINARY FILE...
LAST_NAME  IS STEVENS
FIRST_NAME IS ALFRED
DEPARTMENT IS PRODUCTION
CURR_SAL   IS     11000.00
EMP_ID     IS 071382660
```

Closing an External File

The -CLOSE command closes an external file opened with the -READ or -WRITE command with the NOCLOSE option. The NOCLOSE option keeps a file open until the -READ or -WRITE operation is complete.

Syntax: How to Close an External File

```
-CLOSE filename
```

where:

filename

Is a symbolic name associated with a physical file known to the operating system.

Supplying Dates as Variable Values

You can use variables to supply dates to Dialogue Manager. A date supplied to Dialogue Manager cannot be more than 20 characters long, including spaces. Dialogue Manager variables only accept full-format dates (that is, MDY or MDYY, in any order).

Example: Setting the Difference Between Two Dates

In the following example, the variable &DELAY is set to the difference in days between &LATER and &NOW and the result is returned to your browser.

```
-SET &NOW = 'JUN 30 2002';
-SET &LATER = '2002 25 AUG';
-SET &DELAY = &LATER - &NOW;
-TYPE &DELAY
```

Using Variables With Cross-Century Dates

If you are working with cross-century dates that do not include a four-digit year, variables are available which allow you to identify the century. For details, see the legacy repository.

Performing a Calculation on a Variable

You can use -SET to define a value for a substituted variable based on the results of a logical or arithmetic expression or a combination.

Syntax: How to Perform a Calculation on a Variable

```
-SET &name = expression;
```

where:

&name

Is a user-supplied variable that has its value assigned with the expression.

expression

Is an expression following the rules outlined in the *Creating Reports With WebFOCUS Language* manual, but with limitations as defined in this topic. The semicolon (;) after the expression is required to terminate the -SET command.

Example: Altering a Variable Value

The following example demonstrates the use of -SET to alter variable values based on tests.

```
-SET &OPERATOR = IF &CHOICE EQ A THEN 'GT' ELSE 'LT';
   TABLE FILE SALES
   PRINT PROD_CODE UNIT_SOLD RETAIL_PRICE
   BY STORE_CODE BY DATE
   IF RETAIL_PRICE &OPERATOR 2.00
   END
```

Assuming the user enters the letter A, -SET assigns the string value GT to &OPERATOR. Then, the value GT is passed to the &OPERATOR variable in the procedure, so that the expanded FOCUS command at execution time is:

```
IF RETAIL_PRICE GT 2.00
```

Changing a Variable Value With the DECODE Function

You can use the DECODE function to change a variable to an associated value.

Example: Changing the Value of a Variable

In this example, the variable refers to a movie rating:

```
&SELECT. ENTER A,B,C,D,E.
-SET &RATING = DECODE &SELECT (A G B PG13 C R D NR E PG);
TABLE FILE MOVIES
PRINT TITLE BY RATING
IF RATING EQ &RATING
END
```

For more information on DECODE, see the *Using Functions* manual.

Validating a Variable Value

To ensure that a supplied value is valid and being used properly in a procedure, you can test it for presence, type, and length. You can also verify that it has an acceptable format and that it falls within a specific range of values. For example, you would not want to perform a numeric computation on a variable for which alphanumeric data has been supplied.

Verifying the Presence, Value, Length, or Type of User-Supplied Values

You screen a value by adding a suffix to the variable value:

- The .EXIST suffix tests the presence of a value.
- The .EVAL suffix replaces a variable with its value.
- The .LENGTH suffix tests the length of a value.
- The .TYPE suffix tests the type of a value.
- The .DATE_LOCALE suffix applies the DATE_SEPARATOR, DATE_ORDER, and TIME_SEPARATOR parameter values to date system amper variables.

Reference: Usage Notes for .EVAL

A Dialogue Manager variable is a placeholder for a value that will be substituted at run time. In some situations, the value of the variable may not be resolved at the point where the command containing the variable is encountered, unless evaluation is forced by using the .EVAL operator. One example where .EVAL Is required is in a -IF statement, when the variable is embedded in a label (for example, GOTO AB&*label*.EVAL). The .EVAL operator is also required any time a variable is included within single quotation marks (').

Syntax: How to Verify the Presence, Value, Length, or Type of Variable Values

```
-IF &name{suffix} expression GOTO label...;
```

where:

&name

Is a user-supplied variable.

suffix

Is one of the following:

- **.EXIST,** which tests for the presence of a value. If a value is not present, a zero (0) is passed to the expression. Otherwise, a non-zero value is passed.
- **.LENGTH,** which tests for the length of a value. If a value is not present, a zero (0) is passed to the expression. Otherwise, the number of characters in the value is passed.
- **.TYPE,** which tests for the type of a value. The letter N (numeric) is passed to the expression if the value can be interpreted as a number up to 10^9-1 and stored in four bytes as a floating point format. In Dialogue Manager, the result of an arithmetic operation with numeric fields is truncated to an integer after the whole result of an expression is calculated. If the value could not be interpreted as numeric, the letter A (alphanumeric) is passed to the expression.
- **.EVAL**, which replaces a variable with its value. The .EVAL operator enables you to evaluate the value of a variable immediately, making it possible to change a procedure dynamically. The .EVAL operator is particularly useful in modifying code at run time. When the command procedure is executed, the expression is replaced with the value of the specified variable before any other action is performed. Without the .EVAL operator, a variable cannot be used in place of some commands.

expression

Is a valid expression that uses *&name* with the specified suffix as a variable.

GOTO *label*

Specifies a label to branch to.

Example: Testing for the Presence of a Variable

If a value is not supplied in the following example, &OPTION.EXIST is equal to zero and control is passed to the label -CANTRUN. The -HTMLFORM displays a message that the procedure cannot run. If a value is supplied, control passes to the label -PRODSALES.

```
-IF &OPTION.EXIST GOTO HRINFO ELSE GOTO CANTRUN;
     .
     .
     .
-HRINFO
   TABLE FILE CENTHR
     .
     .
     .
   END
-EXIT
-CANTRUN
-HTMLFORM BEGIN
<HTML>
<BODY>
TOTAL REPORT CAN'T BE RUN WITHOUT AN OPTION.
</BODY>
</HTML>
-HTMLFORM END
-EXIT
```

Example: Testing for the Length of a Variable With an HTML Format Message

If the length of the value supplied for &OPTION is more than one character, control passes to the label -FORMAT. Control then continues to the command -HTMLFORM, which displays a message, informing the user that only a single character can be entered.

```
-IF &OPTION.LENGTH GT 1 GOTO FORMAT ELSE
-GOTO PRODSALES;
     .
     .
     .
-PRODSALES
   TABLE FILE SALES
     .
     .
     .
   END
-EXIT
-FORMAT
-HTMLFORM BEGIN
<HTML>
<BODY>
ONLY A SINGLE CHARACTER IS ALLOWED.
</BODY>
</HTML>
-HTMLFORM END
```

Example: Testing for the Length of a Variable

If the length of &OPTION is greater than one character, control passes to the label -FORMAT. The -HTMLFORM command then displays a message informing the user that only a single character can be entered.

```
-TOP
-IF &OPTION.LENGTH GT 1 GOTO FORMAT ELSE
-GOTO PRODSALES;
     .
     .
     .
-PRODSALES
   TABLE FILE SALES
     .
     .
     .
   END
-EXIT
-FORMAT
-HTMLFORM BEGIN
<HTML>
<BODY>
PLEASE ENTER ONLY ONE CHARACTER.
</BODY>
</HTML>
-HTMLFORM
END
```

Example: Testing for the Type of a Value

If &OPTION is not alphanumeric, control passes to the label -NOALPHA. -HTMLFORM displays an HTML page that informs the user that only alphanumeric characters are allowed.

```
-IF &OPTION.TYPE NE A GOTO NOALPHA ELSE
- GOTO HRINFO;
      .
      .
      .
-HRINFO
   TABLE FILE CENTHR
      .
      .
      .
   END
-EXIT
-NOALPHA
-HTMLFORM BEGIN
<HTML>
<BODY>
ENTER A LETTER ONLY.
</BODY>
</HTML>
-HTMLFORM
```

Example: **Using .EVAL to Allow WebFOCUS to Interpret a Variable**

The code

```
-SET &A='-TYPE';
&A HELLO
```

produces an error message which shows that WebFOCUS does not recognize the value of &A:

```
(FOC1517) UNRECOGNIZED COMMAND -type HELLO
```

Appending the .EVAL operator to the &A variable makes it possible for WebFOCUS to interpret the variable correctly. Adding the .EVAL operator as follows

```
-SET &A='-TYPE';
&A.EVAL HELLO
```

produces the following output:

```
HELLO
```

Example: Evaluating a Variable Immediately

The following example illustrates how to use the .EVAL operator in a record selection expression. The numbers to the left apply to the notes that follow the procedure:

```
1. -SET &R='IF SALARY GT 60000';
2. -IF &Y EQ 'YES' THEN GOTO START;
3. -SET &R = '-*';
   -START
4.   TABLE FILE CENTHR
     PRINT SALARY
     BY PLANT BY LNAME
5.   &R.EVAL
     END
```

The procedure executes as follows:

1. The procedure sets the value of &R to 'IF SALARY GT 60000'.
2. If &Y is YES, the procedure branches to the START label, bypassing the second -SET command.
3. If &Y is NO, the procedure continues to the second -SET command, which sets &R to '-*', which is a comment.

   The report request is stacked.
4. The procedure evaluates the value of &R. If the end user wanted a record selection test, the value of &R is 'IF SALARY GT 60000' and this line is stacked.
5. If the end user did not want a record selection test, the value of &R is '-*' and this line is ignored.

Reference: Applying Date and Time Locale Parameters to System Date Variables

The DATE_SEPARATOR parameter defines a date separator character and the DATE_ORDER parameter defines the order of components for dates, based on your locale. The TIME_SEPARATOR parameter defines the time separator character for the &TOD variable. To use these settings with the Dialogue Manager system variables, (for example, &DATE, &TOD, &YMD, &DATEfmt, and &DATXfmt) append the suffix .DATE_LOCALE to the system variable. This allows system variables that are localized to coexist with non-localized system variables.

Example: Setting Date and Time Parameters for System Variables

The following applies the DATE_ORDER and DATE_SEPARATOR parameters to the &DATE system variable.

```
SET DATE_SEPARATOR = DASH
SET DATE_ORDER = DMY
-TYPE NON-LOCALIZED: &DATE
-TYPE LOCALIZED: &DATE.DATE_LOCALE
```

The output is:

```
NON-LOCALIZED: 04/07/17
LOCALIZED: 07-04-17
```

Validating Variable Values Without Data File Access: REGEX

You can validate a parameter value without accessing the data by using the REGEX mask. The REGEX mask specifies a regular expression to be used as the validation string. A regular expression is a sequence of special characters and literal characters that you can combine to form a search pattern.

Many references for regular expressions exist on the web. For a basic summary, see the section *Summary of Regular Expressions* in Chapter 2, *Security*, of the *Server Administration* manual.

The following messages display in case of an error:

```
(FOC2909) INVALID REGULAR EXPRESSION:
(FOC2910) RESPONSE DOES NOT MATCH THE REGULAR EXPRESSION:
```

Syntax: How to Validate a Variable Value Using a REGEX Mask

```
&variable.(|VALIDATE=REGEX,REGEX='regexpression').
```

where:

&variable

Is the variable to validate.

regexpression

Is the regular expression that specifies the acceptable values.

Example: Using a REGEX Mask to Validate a Social Security Number

The following request validates a Social Security number in either *xxxxxxxxx* or *xxx-xx-xxxx* format:

```
-REPEAT NEXTFMT FOR &FMTCNT FROM 1 TO 2
-SET &EMPID1=DECODE &FMTCNT(1 '071382660' 2 '818-69-2173');
-SET &EMPID=IF
       &EMPID1.(|VALIDATE=REGEX,REGEX='^\d{3}\-?\d{2}\-?\d{4}$').Employee
ID. CONTAINS '-'
-       THEN EDIT(&EMPID1,'999$99$9999') ELSE &EMPID1;
TABLE FILE EMPLOYEE
HEADING
" "
"Testing EMPID = &EMPID1</
1"
PRINT EID CSAL
WHERE EID EQ '&EMPID.EVAL'
ON TABLE SET PAGE NOPAGE
ON TABLE SET STYLE *
GRID=OFF,$
END
-RUN
-NEXTFMT
```

The output is

```
Testing EMPID = 071382660


EMP_ID CURR_SAL
071382660 $11,000.00

Testing EMPID = 818-69-2173


EMP_ID CURR_SAL
818692173 $27,062.00
```

Example: Using REGEX With an Incorrect Value

In the following request, the second value for &EMPID1 is invalid because it does not conform to the REGEX mask:

```
-REPEAT NEXTFMT FOR &FMTCNT FROM 1 TO 2
-SET &EMPID1=DECODE &FMTCNT(1 '071382660' 2 '818-69-2173');
-TYPE EMPID1 = &EMPID1
-SET &EMPID=&EMPID1.(|VALIDATE=REGEX,REGEX='^\d{3}\d{2}\d{4}$').Employee
ID.;
-TYPE EMPID = &EMPID
-NEXTFMT
```

The FOC2910 message in the output shows that the second value for &EMPID1 was rejected:

```
EMPID1 = 071382660
EMPID = 071382660
EMPID1 = 818-69-2173
 ERROR AT OR NEAR LINE      7  IN PROCEDURE __WCFEX FOCEXEC *
(FOC2910)  RESPONSE DOES NOT MATCH THE REGULAR EXPRESSION: 818-69-2173
 ERROR AT OR NEAR LINE      7  IN PROCEDURE __WCFEX FOCEXEC *
(FOC295) A VALUE IS MISSING FOR: &EMPID1
```

Example: Using REGEX With an Invalid Regular Expression

In the following request, the REGEX mask is not a valid regular expression:

```
-SET &EMPID1='071382660';
-SET &EMPID=&EMPID1.(|VALIDATE=REGEX,REGEX='^\d{3}\d{2)}\d{4}$').Employee
ID.;
```

The FOC2909 message in the output shows that the regular expression is not valid:

```
ERROR AT OR NEAR LINE      5  IN PROCEDURE __WCFEX FOCEXEC *
(FOC2909)  INVALID REGULAR EXPRESSION: ^\d{3}\d{2)}\d{4}$
 ERROR AT OR NEAR LINE      5  IN PROCEDURE __WCFEX FOCEXEC *
(FOC295) A VALUE IS MISSING FOR: &EMPID1
```

Verifying User-Supplied Values Against a Set of Format Specifications

You can specify that a variable must adhere to a set of format conditions against which entered values can be compared. If the entered value does not have the specified format, a message displays informing the user that there are too many characters. The user can acknowledge the message and can try again to enter a valid value.

Reference: Format Specifications for Variables

Alphanumeric formats are described by the letter A, followed by the number of characters. The number of characters can be from 1 to 3968.

Numeric formats are described by the letter I, followed by the number of digits to be entered. The number of digits can be from 1 to 10 (value must be less than 2^{31}-1), and the value supplied for the number can contain a decimal point.

The description of the format must be enclosed within periods (.).

If you test field names against input variable values, specify formats of the input variables. If you do not, and the supplied value exceeds the format specification from the Master File, the procedure is ended and error messages are displayed. To continue, the procedure must be executed again. However, if you do include the format, and the supplied value exceeds the format, the entered value will be rejected and the user is prompted again.

Note: Parameter values are internally stored in memory as alphanumeric codes. To perform arithmetic operations, the parameter value is converted to double precision floating point decimal and then the result is converted back to alphanumeric codes, dropping the decimal places. For this reason, do not perform tests that look for the decimal places in the numeric codes.

Example: Using a Format Specification to Verify User Input

Consider the following format specification for the STATE and QTY_IN_STOCK fields.

```
TABLE FILE CENTORD
BY QTY_IN_STOCK
BY STATE
BY PROD_NUM
BY PRODNAME
ON TABLE SUBHEAD
"Inventory Report"
WHERE STATE EQ '&STATE.A2.US State:.'
WHERE QTY_IN_STOCK LT '&PRODNUM.I7.Quantity in Stock below:.'
END
```

If, in the above example, the user enters more than two alphanumeric characters for the STATE value it is rejected, the message *There are too many characters in your entry.* is displayed, and the user is prompted again.

If, in the above example the user enters more than seven numeric characters for the QTY_IN_STOCK, the value is rejected, the message *There are too many digits in your entry: (format is 'I7')* is displayed, and the user is prompted again.

Verifying User-Supplied Values Against a Value Range

You can specify that a variable value must fall within a range of values against which entered values can be compared. If the entered value is not within the specified value range, a message displays informing the user that there are too many characters. The user can acknowledge the message and can try again to enter a valid value.

Reference: Value Range Specifications for Variables

A value range is enclosed within parenthesis specifying (FROM *value1* TO *value2*), where *value1* is a numeric value less than *value2*. For example,

```
&QTY_IN_STOCK.(FROM 1 TO 10000)
```

Note: Parameter values are internally stored in memory as alphanumeric codes. To perform arithmetic operations, the parameter value is converted to double precision floating point decimal, and then the result is converted back to alphanumeric codes, dropping the decimal places. For this reason, do not perform tests that look for the decimal places in the numeric codes.

Example: Using a Value Range Specification to Verify User Input

Consider the following value range specification for the &QTY_IN_STOCK variable.

```
TABLE FILE CENTINV
PRINT QTY_IN_STOCK
BY PROD_NUM
BY PRODNAME
ON TABLE SUBHEAD
"Inventory Report"
WHERE QTY_IN_STOCK LT &QTY_IN_STOCK.(FROM 1 TO 10000).Quantity in Stock
below:.
END
```

If, in the above example, the user enters a value less than 1 and more than 10000 for the *Quantity in Stock below* parameter value, the value is rejected and the message *value entered is not within range 1 - 10000* is displayed.

Counting With an Indexed Variable

You can append the value of one variable to the value of another variable, creating an *indexed variable*. This feature applies to both local and global variables.

If the indexed value is numeric, the effect is similar to that of an array in traditional computer programming languages. For example, if the value of index &K varies from 1 to 10, the variable &AMOUNT.&K refers to one of ten variables, from &AMOUNT1 to &AMOUNT10.

A numeric index can be used as a counter, and it can be set, incremented, and tested in a procedure.

Syntax: How to Create an Indexed Variable

```
-SET &name.&index[.&index...] = expression;
```

where:

&name

Is a variable.

.&index

Is a numeric or alphanumeric variable whose value is appended to *&name*. The period is required.

When more than one index is used, all index values are concatenated and the string appends to the name of the variable.

For example, &V.&I.&J.&K is equivalent to &V1120 when &I=1, &J=12, and &K=0.

expression

Is a valid expression. For information on the kinds of expressions you can write, see the *Creating Reports With WebFOCUS Language* manual.

Example: Using an Indexed Variable in a Loop

An indexed variable can be used in a loop. The following example creates the equivalent of a DO loop used in traditional programming languages:

```
-SET &N = 0;
-LOOP
-SET &N = &N+1;
-IF &N GT 12 GOTO OUT;
-SET &MONTH.&N=&N;
-TYPE &MONTH.&N
-GOTO LOOP
-OUT
```

In this example, &MONTH is the indexed variable and &N is the index. The value of the index is supplied through the command -SET. The first -SET initializes the index to 0, and the second -SET increments the index each time the procedure goes through the loop.

If the value of an index is not defined prior to reference, a blank value is assumed. As a result, the name and value of the indexed variable will not change.

Indexed variables are included in the system limit of 384 variables.

Creating a Standard Quote-Delimited String

Character strings must be enclosed in single quotation marks to be handled by most database engines. In addition, embedded single quotation marks are indicated by two contiguous single quotation marks. Quotation marks are required around variables containing delimiters, which include spaces and commas.

The QUOTEDSTRING suffix on a Dialogue Manager variable applies the following two conversions to the contents of the variable:

- Any single quotation mark embedded within a string is converted to two single quotation marks.

- Single quotation marks are added around the string.

Dialogue Manager commands differ in their ability to handle character strings that are not enclosed in single quotation marks and contain embedded blanks. An explicit or implied -PROMPT command can read such a string. The entire input string is then enclosed in single quotation marks when operated on by .QUOTEDSTRING.

Note: When using the -SET command to reference a character string, ensure the character string is enclosed in single quotes to prevent errors.

Syntax: How to Create a Standard Quote-Delimited Character String

```
&var.QUOTEDSTRING
```

where:

&var

Is a Dialogue Manager variable.

Example: Creating a Standard Quote-Delimited Character String

The following example shows the results of the QUOTEDSTRING suffix on input strings.

```
-SET &A = ABC;
-SET &B = 'ABC';
-SET &C = O'BRIEN;
-SET &D = 'O'BRIEN';
-SET &E = 'O''BRIEN';
-SET &F = O''BRIEN;
-SET &G = OBRIEN';
-TYPE  ORIGINAL = &A QUOTED = &A.QUOTEDSTRING
-TYPE  ORIGINAL = &B QUOTED = &B.QUOTEDSTRING
-TYPE  ORIGINAL = &C QUOTED = &C.QUOTEDSTRING
-TYPE  ORIGINAL = &D QUOTED = &D.QUOTEDSTRING
-TYPE  ORIGINAL = &E QUOTED = &E.QUOTEDSTRING
-TYPE  ORIGINAL = &F QUOTED = &F.QUOTEDSTRING
-TYPE  ORIGINAL = &G QUOTED = &G.QUOTEDSTRING
```

The output is:

```
ORIGINAL = ABC         QUOTED = 'ABC'
ORIGINAL = ABC         QUOTED = 'ABC'
ORIGINAL = O'BRIEN     QUOTED = 'O''BRIEN'
ORIGINAL = O'BRIEN     QUOTED = 'O''BRIEN'
ORIGINAL = O'BRIEN     QUOTED = 'O''BRIEN'
ORIGINAL = O''BRIEN    QUOTED = 'O''''BRIEN'
ORIGINAL = OBRIEN'     QUOTED = 'OBRIEN'''
```

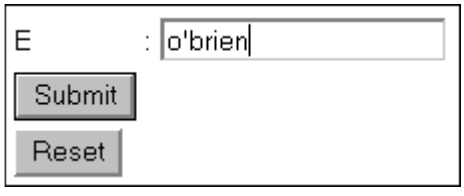
Note: The -SET command will remove single quotes around a string. Notice in the example above that the result of -SET &B = 'ABC' was changed to ORIGINAL = ABC (as shown in the output), prior to the QUOTEDSTRING conversion.

Example: Converting User Input to a Standard Quote-Delimited Character String

The following -TYPE command accepts quoted or unquoted input and displays quoted output.

```
-TYPE THE QUOTED VALUE IS: &E.QUOTEDSTRING
```

The output is:




```
THE QUOTED VALUE IS: 'o''brien'
```

Example: Using Quote-Delimited Strings With Relational Data Adapters

The following procedure creates an Oracle table named SQLVID from the VIDEOTRK data source.

```
TABLE FILE VIDEOTRK
SUM CUSTID EXPDATE PHONE STREET CITY STATE ZIP
   TRANSDATE PRODCODE TRANSCODE QUANTITY TRANSTOT
BY LASTNAME BY FIRSTNAME
WHERE LASTNAME NE 'NON-MEMBER'
ON TABLE HOLD
END
-RUN
CREATE FILE SQLVID
-RUN
MODIFY FILE SQLVID
FIXFORM FROM HOLD
DATA ON HOLD
END
```

Consider the following SQL Translator request:

```
SET TRACEUSER = ON
SET TRACEON = STMTRACE//CLIENT
SQL
SELECT *
FROM SQLVID WHERE LASTNAME  = &1.QUOTEDSTRING;
END
```

When this request is executed, you must enter a last name, in this case, O'BRIEN:

```
PLEASE SUPPLY VALUES REQUESTED
1=
O'BRIEN
```

In the generated SQL request, the character string used for the comparison is correctly enclosed in single quotation marks, and the embedded single quote is doubled:

```
SELECT SQLCOR01.CIN , SQLCOR01.LN , SQLCOR01.FN ,
SQLCOR01.EXDAT , SQLCOR01.TEL , SQLCOR01.STR , SQLCOR01.CITY ,
SQLCOR01.PROV , SQLCOR01.POSTAL_CODE , SQLCOR01.OUTDATE ,
SQLCOR01.PCOD , SQLCOR01.TCOD , SQLCOR01.NO , SQLCOR01.TTOT
FROM SQLVID SQLCOR01  WHERE SQLCOR01.LN  = 'O''BRIEN';
```

The following input variations are translated to the correct form of quoted string demonstrated in the trace.

```
'O'BRIEN'
'O''BRIEN'
```

Any other variation results in:

- A valid string that does not match the database value and does not return any rows. For example, O''''BRIEN becomes 'O''''''''BRIEN' in the WHERE predicate.
- An invalid string that produces one of the following messages:

  Error - Semi-colon or END expected

  Error - Missing or Misplaced quotes

  Error - (value entered) is not a valid column

  Error - Syntax error on line ... Unbalanced quotes

Strings without embedded single quotation marks can be entered without quotes or embedded in single quotation marks, either SMITH or 'SMITH'.

If you use &1 without the QUOTEDSTRING suffix in the request, acceptable input strings that retrieve O'Brien's record are:

```
'''O'''BRIEN'''
'''O''''BRIEN'''
```

Using &1 without the QUOTEDSTRING suffix, the acceptable form of a string without embedded single quotation marks is '''SMITH'''.

To make a string enclosed in single quotation marks acceptable without the QUOTEDSTRING suffix, use '&1' in the request. In this case, in order to retrieve the record for O'Brien, you must enter the string that would have resulted from the QUOTEDSTRING suffix:

```
'O''''BRIEN'
```

To enter a string without embedded single quotation marks using '&1', you can either omit the surrounding single quotation marks or include them: SMITH or 'SMITH'.

Note: The form '&1.QUOTEDSTRING' is not supported.

Reference: Usage Notes for Quote-Delimited Character Strings

- QUOTEDSTRING must be used when passing parameter strings to drill-downs if the parameter contains multi-selected values. For an example, see *Rules for a Multiselect List of Values* on page 204.
- An unmatched single quotation mark at the beginning of a character string is treated as invalid input and generates the following message:

  ```
  (FOC257) MISSING QUOTE MARKS:  value;
  ```

Creating and Working With Variables

You can use variables to customize a procedure, as described in *Customizing a Procedure With Variables* on page 336. Variables fall into two categories:

- Local and global variables, whose user-defined values must be supplied at run time. For details, see *Creating and Working With Local and Global Variables* on page 370.
- System and statistical variable values are predefined and automatically supplied by the system when a procedure references them. For details, see *Naming Conventions for Local and Global Variables* on page 370.

Reference: Variable Characteristics

The following features apply to all variables:

- A variable stores numbers or a string of text and can be placed anywhere in a procedure.

  Note: A Dialogue Manager variable contains only alphanumeric data. If a function or expression returns a numeric value to a Dialogue Manager variable, the value is truncated to an integer and converted to alphanumeric format before being stored in the variable.
- A variable can refer to a command, a data source field, a verb, or a phrase.
- The total length of all variable values cannot exceed 32K.

- A variable can be supplied directly in the command, when you execute the procedure, or through the -DEFAULT, -SET, or -READ commands in the procedure.

Creating and Working With Local and Global Variables

Local and global variables are variables whose user-defined values must be supplied at run time.

- A local variable retains its values during the execution of one procedure. Values are lost after the procedure finishes processing. Values are *not* passed to other procedures that contain the same variable name.

  A local variable is identified by a single ampersand (&) followed by the variable name.

- A global variable retains its value for the duration of your connection to the WebFOCUS Reporting Server and are passed from the execution of one procedure to the next.

  Because WebFOCUS creates a new WebFOCUS session on the WebFOCUS Reporting Server each time it submits a request, values for global variables are not retained between report requests. This means that you can use the same global variable in more than one procedure as long as these procedures are called in the same request.

  If you want a global value of the variable to be in effect for every procedure, add the variable to a WebFOCUS Reporting Server profile, such as EDASPROF.PRF.

  A global variable is identified by a double ampersand (&&) followed by the variable name.

Note: Values with embedded commas (',') are not allowed.

Reference: Naming Conventions for Local and Global Variables

Local and global variable names are user-defined, while system and statistical variables have predefined names. The following rules apply to the naming of local and global variables:

- A local variable name is always preceded by an ampersand (&).
- A global variable name is always preceded by a double ampersand (&&).
- Embedded blanks are not permitted in a variable name.
- If a value for a variable might contain an embedded blank, comma, or equal sign, enclose the variable in single quotation marks when you refer to it.
- A variable name may be any combination of the characters A through Z, 0 through 9, and the underscore. The first character of the name should be a letter.

- You can assign a number instead of a name to a variable to create a positional variable.
- The underscore may be included in a variable name, but the following special characters are not permitted: plus sign (+), minus sign (-), asterisk (*), slash (/), period (.), ampersand (&), and semicolon (;).

Syntax: How to Specify a Variable Name

```
&[&]name
```

where:

name

Is the variable name. A single ampersand (&) denotes a local variable, while a double ampersand (&&) denotes a global variable. A single ampersand followed by a numeric string denotes a positional variable.

The name you assign must follow the rules outlined in *Naming Conventions for Local and Global Variables* on page 370.

Example: Variable Names

The following variables are properly named:

```
&WHICHPRODUCT
&WHICH_CITY
'&CITY'
&&CITY
```

The following variables are improperly named for the reason given:

```
&WHICH CITY
```

Contains an embedded blank.

```
&WHICH -CITY
```

Contains a hyphen.

```
WHICHCITY
```

Leading ampersand(s) missing.

Example: Using Local Variables

Consider the following procedure, SALESREPORT, in which &CITY, &CODE1, and &CODE2 are local variables.

```
TABLE FILE SALES
HEADING CENTER
"MONTHLY REPORT FOR &CITY"
"PRODUCT CODES FROM &CODE1 TO &CODE2"
" "
SUM UNIT_SOLD AND RETURNS AND COMPUTE
RATIO/D5.2 = 100 * (RETURNS/UNIT_SOLD);
BY CITY
IF CITY EQ &CITY
BY PROD_CODE
IF PROD_CODE IS-FROM &CODE1 TO &CODE2
END
```

Assume you supply the following values when you call the procedure:

```
EX SALESREPORT CITY=STAMFORD, CODE1=B10, CODE2=B20
```

Dialogue Manager substitutes the values for the variables as follows:

```
TABLE FILE SALES
HEADING CENTER
"MONTHLY REPORT FOR STAMFORD"
"PRODUCT CODES FROM B10 TO B20"
" "
SUM UNIT_SOLD AND RETURNS AND COMPUTE
RATIO/D5.2 = 100 * (RETURNS/UNIT_SOLD);
BY CITY
IF CITY EQ STAMFORD
BY PROD_CODE
IF PROD_CODE IS-FROM B10 TO B20
END
```

After the procedure executes and terminates, the values STAMFORD, B10, and B20 are lost.

Example: Using Global Variables

The following example illustrates the use of three global variables: &&CITY, &&CODE1, &&CODE2. The values are substituted in the first procedure, PROC1, and the values are retained and passed to the second procedure, PROC2.

```
TABLE FILE SALES
HEADING CENTER
"MONTHLY REPORT FOR &&CITY"
SUM UNIT_SOLD AND RETURNS AND COMPUTE
RATIO/D5.2 = 100 * (RETURNS/UNIT_SOLD);
BY CITY
IF CITY EQ &&CITY
BY PROD_CODE
IF PROD_CODE IS-FROM &&CODE1 TO &&CODE2
END
EX PROC2
```

```
TABLE FILE SALES
HEADING CENTER
"MONTHLY REPORT FOR &&CITY AND PRODUCT &&CODE1"
PRINT UNIT_SOLD AND RETURNS AND COMPUTE
RATIO/D5.2 = 100 * (RETURNS/UNIT_SOLD);
BY CITY
IF CITY EQ &&CITY
IF PROD_CODE EQ &&CODE1
END
```

Concatenating Variables

You can append a variable to a character string, or you can combine two or more variables and/or literals to concatenate a variable. See the *Creating Reports With WebFOCUS Language* manual for details on concatenation. When using variables, it is important to separate each variable from the concatenation symbol with a space.

Syntax: How to Concatenate a Variable

```
-SET &name3 = &name1 || &name2;
```

where:

&name3

Is the name of the concatenated variable.

&name1 || &name2

Are the variables, separated by the concatenation symbol and a space on both sides.

Removing Trailing Blanks From a Variable

You can remove trailing blanks form a variable with the Dialogue Manager TRUNCATE function. TRUNCATE can be used only with Dialogue Manager commands that support functions, such as -SET and -IF commands. It cannot be used in a -TYPE command or in arguments passed to procedures.

The TRUNCATE can act only on one variable at a time. If you attempt to use the Dialogue Manager TRUNCATE function with more than one argument, the following error message is generated:

```
(FOC03665) Error loading external function 'TRUNCATE'
```

Note: A user-written function of the same name can exist without conflict.

Syntax: How to Remove Trailing Blanks

```
-SET &var2 = TRUNCATE(&var1);
```

where:

&var2

Is the Dialogue Manager variable to which the truncated string is returned. The length of this variable is the length of the original string or variable without the trailing blanks. If the original variable consisted of only blanks, a single blank with a length of one is returned.

&var1

Is a Dialogue Manager variable or a literal string enclosed in single quotation marks (‘). System variables and statistical variables are allowed, as well as user-created local and global variables.

Example: Removing Trailing Blanks

In the following example, TRUNCATE removes trailing blanks.

```
-SET &LONG = 'ABC   ' ;
-SET &RESULT =  TRUNCATE(&LONG);
-SET &LL = &LONG.LENGTH;
-SET &RL = &RESULT.LENGTH;
-HTMLFORM BEGIN
<HTML>
<BODY>
<P>LONG   =  &LONG     LENGTH =  &LL</P>
<P>RESULT =  &RESULT     LENGTH =  &RL</P>
</BODY>
</HTML>
-HTMLFORM END
```

The output is:

```
LONG = ABC LENGTH = 06

RESULT = ABC LENGTH = 03
```

Example: Removing Trailing Blanks in a String of All Blanks

In the following example, when TRUNCATE removes the trailing blanks, since the string consists of all blanks, a string with the length of 1 is returned.

```
-SET &LONG = '      ' ;
-SET &RESULT =  TRUNCATE(&LONG);
-SET &LL = &LONG.LENGTH;
-SET &RL = &RESULT.LENGTH;
-HTMLFORM BEGIN
<HTML>
<BODY>
<P>LONG   =  &LONG     LENGTH =  &LL</P>
<P>RESULT =  &RESULT     LENGTH =  &RL</P>
</BODY>
</HTML>
-HTMLFORM END
```

The output is:

LONG = LENGTH = 06

RESULT = LENGTH = 01

Example: Using the TRUNCATE Function as an Argument for a Function

In the following example, TRUNCATE is an argument for the EDIT function.

```
-SET &LONG = 'ABC   ' ;
-SET &RESULT =  EDIT(TRUNCATE(&LONG)|'Z','9999');
-SET &LL = &LONG.LENGTH;
-SET &RL = &RESULT.LENGTH;
-HTMLFORM BEGIN
<HTML>
<BODY>
<P>LONG   =  &LONG     LENGTH =  &LL</P>
<P>RESULT =  &RESULT     LENGTH =  &RL</P>
</BODY>
</HTML>
-HTMLFORM END
```

The output is:

LONG = ABC LENGTH = 06

RESULT = ABCZ LENGTH = 04

Displaying the Value of a Variable

You can display the value of a variable by issuing a query.

Syntax: How to Display the Value of a Local Variable

```
-? &[string]
```

where:

string

Is a variable name. If this parameter is not specified, the current values of all local, global, system, and statistical variables are displayed.

Syntax: How to Display the Value of Global Variables

```
? &&
```

Assigning the Value of a Variable to a Parameter

You can capture the value of a SET parameter value in a local variable.

In WebFOCUS, the result is returned in your browser window, or as a comment in the HTML file if there is other HTML output from the request. Use the applicable web browser functions to view the HTML comments (for example, View Source in Microsoft Internet Explorer).

Syntax: How to Assign the Value of a Parameter to a Variable

```
-? SET parameter variable
```

where:

parameter

Is a SET parameter.

variable

Is the name of the variable where the value is to be stored.

Example: Assigning the Value of a Variable to a Parameter

Entering the following stores the value of ASNAMES as the value for &ABC.

```
-? SET ASNAMES &ABC
```

If you omit &ABC from the command, then a variable called &ASNAMES is created that contains the value of ASNAMES.

Working With System and Statistical Variables

System and statistical variable values are predefined and automatically supplied by the system when a procedure references them. System and statistical variables have names that begin with a single ampersand (&). Examples of these variables are &LINES, which indicates how many lines of output were produced, and &DATE, which indicates the current date.

- System values can be used to control processing. For example, if no records are read for a report request because no records fit the selection criteria, processing may halt or the selection criteria may be changed.

  For a list of system variables, see *WebFOCUS System Variables* on page 377.

- A statistical variable can be used in a procedure to derive information about the last request run. Dialogue Manager automatically supplies values for statistical variables.

  For a list of statistical variables, see *WebFOCUS Statistical Variables* on page 383.

Reference: WebFOCUS System Variables

You can override WebFOCUS system-supplied values by assigning a value to the variable explicitly with a -SET or -DEFAULT command. However, we recommend that you do not override these values.

The following table lists the system variables available in WebFOCUS.

System Variable	Description	Format or Value
`&APPROOT`	Contains directories and data. By default, this is the Application Root directory (APPROOT directory) in which WebFOCUS looks for project files. Sample files are provided in the \ibinccen and \ibisamp directories.	`d:\ibi\apps`

System Variable	Description	Format or Value
&AUTOINDEX	Retrieves data faster by automatically taking advantage of indexed fields in most cases where TABLE requests contain equality or range tests on those fields. Applies only to FOCUS data sources. AUTOINDEX is never performed when the TABLE request contains an alternate file view, for example, TABLE FILE *filename.filename*. Indexed retrieval is not performed when the TABLE request contains BY HIGHEST or BY LOWEST phrases and AUTOINDEX is ON.	No
&DATE	Returns the current date.	MM/DD/YY
&DATE*fmt* &DATX*fmt*	Returns the current date or date-time value, where *fmt* can be any valid date or date-time format. &DATE*fmt* retains trailing blanks in the returned value. &DATX*fmt* suppresses trailing blanks in the returned value. **Note:** Using the concatenation symbol (|) to remove punctuation between components is not supported. To return a value without punctuation between the components, use &YYMD or &DATEHYYMDN. For information about date and date-time formats, see Chapter 4, *Describing an Individual Field*, in the *Describing Data With WebFOCUS Language* manual.	Returns the current date or date-time value, where *fmt* can be any valid date or date-time format. Because many date format options can be appended to the prefix *DATE* to form one of these variable names, you should avoid using DATE as the prefix when creating a variable name.
&DMY	Returns the current date.	DDMMYY

System Variable	Description	Format or Value
&DMYY	Returns the current (four-digit year) date.	DDMMCCYY
&ECHO	Displays command lines as they execute in order to test and debug procedures.	ON,OFF, ALL, or NONE
&FOCCODEPAGE	Returns the code page being used by the server.	An integer value.
&FOCEXURL	Runs and executes drill downs remotely. The drill down program can be on your local machine or on a remote machine.	/ibi_apps/WFServlet?IBIF_webapp=/ibi_apps&IBIC_server
&FOCFEXNAME	Returns the name of the FOCEXEC running even if it was executed using an EX command or a -INCLUDE command from within another FOCEXEC. This variable differs from the &FOCFOCEXEC variable because &FOCFOCEXEC returns the name of the calling FOCEXEC only.	
&FOCFOCEXEC	Manages reporting operations involving many similarly named requests that are executed using the EX command. &FOCFOCEXEC enables you to easily determine which procedure is running by returning the fully qualified path for the procedure. &FOCFOCEXEC can be specified within a request or a Dialogue Manager command to display the name of the current procedure.	

System Variable	Description	Format or Value
&FOCHTMLURL	Allows you to access resources with an alias other than /ibi_apps/ibi_html. To generate an alias other than /ibi_html on the WebFOCUS Reporting Server, use the SET FOCHTMLURL command to set the alias that will be generated instead of /ibi_apps/ibi_html. This command will most likely be used in a server profile (EDASPROF.PRF) or in one of the WFS files (site.wfs) to establish a default setting for the installation.	/ibi_apps/ibi_html
&FOCINCLUDE	Manages reporting operations involving many similarly named requests that are included using the -INCLUDE command. &FOCINCLUDE can be specified within a request or a Dialogue Manager command to display the name of the current procedure.	
&FOCLANGCODE	Holds the three-character language code, for example, GER for GERMAN or FRE for FRENCH. This variable is blank when the language is English, either AME or ENG.	
&FOCMODE	Identifies the operating environment.	CMS, CRJE, MSO, OS, or TSO.
&FOCNEXTPAGE	Is a variable whose value is determined by the last page number used by the last report. Its value is one more than the last page number used in the last report.	0

System Variable	Description	Format or Value
&FOCQUALCHAR	Returns the character used to separate the components of qualified field names.	. : ! % | \
&FOCREL	Identifies the FOCUS release number.	The release number.
&FOCSECUSER	Returns the user ID of the user connected to the running agent on a Reporting Server running with security ON.	
&FOCSECGROUP	Returns the primary group ID of the user ID stored in &FOCSECUSER.	
&FOCSECGROUPS	Returns list of group IDs of the user ID stored in &FOCSECUSER. The length limit of the returned list is 4000 characters.	
&MDY	Returns the current date. Useful for numeric comparisons.	MMDDYY
&MDYY	Returns the current (four-digit year) date.	MMDDCCYY
&RETCODE	Is the value returned after a server command is executed. &RETCODE executes all stacked commands, like the command -RUN.	Any value defined by the server command.
&SETFILE	Contains the value from the SET FILE command.	

System Variable	Description	Format or Value
&TOD	Returns the current time. When you enter FOCUS, this variable is updated to the current system time only when you execute a MODIFY, SCAN, or FSCAN command. To obtain the exact time during any process, use the HHMMSS function.	HH.MM.SS
&YMD	Returns the current date.	YYMMDD
&YYMD	Returns the current (four-digit year) date.	CCYYMMDD

Example: Retrieving the Date With &DATE

The following example incorporates the system variable &DATE into a request. The footing uses the system variable &DATE to insert the current system date at the bottom of the report.

```
TABLE FILE SALES
SUM UNIT_SOLD
BY PROD_CODE
FOOTING
"CALCULATED AS OF &DATE"
END
```

Example: Retrieving the Procedure Name With &FOCFOCEXEC

The following example incorporates the system variable &FOCFOCEXEC in a report request to display the name of the current procedure.

```
SET PAGE=OFF
TABLE FILE EMPLOYEE
"REPORT: &FOCFOCEXEC -- EMPLOYEE SALARIES"
PRINT CURR_SAL BY EMP_ID
ON TABLE SET STYLE *
TYPE=REPORT, SIZE=10, GRID=OFF,$
END
```

If the request is stored as a FOCEXEC called SALPRINT, the output is:

```
REPORT: SALPRINT -- EMPLOYEE SALARIES

EMP_ID                                                    CURR_SAL
```

```
071382660                                                    $11,000.00
112847612                                                    $13,200.00
115360218                                                         $.00
117593129                                                    $18,480.00
119265415                                                     $9,500.00
119329144                                                    $29,700.00
121495681                                                         $.00
123764317                                                    $26,862.00
126724188                                                    $21,120.00
219984371                                                    $18,480.00
326179357                                                    $21,780.00
451123478                                                    $16,100.00
543729165                                                     $9,000.00
818692173                                                    $27,062.00
```

Reference: **WebFOCUS Statistical Variables**

The following statistical variables are available in WebFOCUS.

Statistical Variable	Description
&BASEIO	Is the number of input/output operations.
&FOCDISORG	Is the percentage of disorganization for a FOCUS file.
&FOCERRNUM	Is the last error number, in the format FOCnnnn, displayed after the execution of a procedure. If multiple errors occurred, this variable holds the number of the most recent error. If no error occurred, the variable has a value of 0.
&INDEXIO	Is the number of indexed input/output operations.
&LINES	Is the number of lines printed in last answer set.
&READS	Is the number of physical reads from an external file.

Statistical Variable	Description
&RECORDS	Is the number of records retrieved in last answer set.
&SORTIO	Is the number of sorted input/output operations.

Example: **Counting the Number of Printed Lines With &LINES**

In the following example, the system calculates the value of the statistical variable &LINES. If &LINES is 0, control passes to the TABLE FILE EMPLOYEE request identified by the label -RPT2. If the value is not 0, control passes to the label -REPTDONE, and processing is terminated.

```
TABLE FILE SALES
HEADING CENTER
"MONTHLY REPORT FOR &CITY"
SUM UNIT_SOLD AND RETURNS AND COMPUTE
RATIO/D5.2 = 100 * (RETURNS/UNIT_SOLD);
BY CITY
IF CITY EQ &CITY
BY PROD_CODE
IF PROD_CODE IS-FROM &CODE1 TO &CODE2
END
-RUN
-IF &LINES EQ 0 GOTO RPT2 ELSE GOTO REPTDONE;
-RPT2
TABLE FILE EMPLOYEE
   .
   .
   .
END
-RUN
-QUIT
-REPTDONE
-EXIT
```

Using Numeric Amper Variables in Functions

WebFOCUS stores all amper variables as strings in alphanumeric format whether they contain alphanumeric or numeric data or a mixture of the two. There are only two data types available to amper variables: alphanumeric and numeric.

Determining Amper Variable Data Type

Data typing for amper variables is determined by the data content only. As a result, using quotation marks around a numeric value in a -SET command has no effect on the data type of the amper variable.

For example, the following request stores numeric data in variables &A, &B, and &C:

```
-SET &A=12345;
-SET &B='12345';
-SET &C=123.45
-TYPE &A &B &C
-TYPE &A.TYPE &B.TYPE &C.TYPE
```

The output shows that &A, &B, and &C all have the numeric data type:

```
12345 12345 123.45
N  N  N
```

Manipulating Amper Variables

When an amper variable is displayed, substituted, concatenated, or appended, there is no transformation of the value contained in the amper variable.

Substitution

```
-SET &C=123.45
IF RETAIL_COST EQ &C
```

becomes

```
IF RETAIL_COST EQ 123.45
```

Also, consider the following:

```
-SET &D= &C;
-TYPE &D &D.TYPE
```

The output shows that &D has the same value as &C and is also numeric:

```
123.45  N
```

Concatenation

The amper variable &F is created by concatenating &A and &C:

```
-SET &F =  &A | &C;
-TYPE &F &F.TYPE
```

The output shows that the value of &F is the value of &A followed by the value of &C, and that the type is numeric:

```
12345123.45  N
```

The following example creates the amper variable &E by embedding an ampersand in the string. The ampersand is not recognized as the start of a variable name and is treated as an alphanumeric symbol in a string:

```
-SET &E = 1234&C;
-TYPE &E &E.TYPE
```

The output shows that the variable is of type alphanumeric, not numeric. It is not the concatenation of the string '1234' with the variable &C:

```
1234&C  A
```

This same behavior can be produced with concatenation:

```
-SET &G = AT&|T;
-TYPE &G &G.TYPE
```

The output is:

```
AT&T  A
```

Using an Amper Variable in an Expression

When an amper variable is used in an expression, conversion may be required in order to process the expression. The amper variable used in the expression is generally seen as a literal, and its value is substituted in before the expression is processed. Under these circumstances, data conversion necessary to process the expression is performed. Numerics contained in amper variables are seen as integers. If the expression can be evaluated as integer, it will be.

In the following example, &C is set to 123.55. Then an expression creates &D by adding 100 to &C :

```
-SET &C=123.55;
-SET &D=&C + 100;
-TYPE &D &D.TYPE
```

The output shows that &D is numeric and its value is 123.55+100 truncated to the integer 223 because integer arithmetic is used:

```
223  N
```

The following expression requires conversion to double precision, as the numeric literal (100.49) in the expression is not an integer:

```
-SET &C=123.55;
-SET &D=&C + 100.49;
-TYPE &D &D.TYPE
```

The output shows that while the arithmetic was done by converting the value of &C to double precision, the result is truncated before being returned to &D:

```
224  N
```

If you want the result to retain the decimal places, you can set the DMPRECISION parameter to the number of decimal places you want returned to the resulting amper variable.

For example:

```
SET DMPRECISION=2
-RUN
-SET &C=123.55;
-SET &D=&C + 100.49;
```

Now the result retains the decimal places:

```
224.04 N
```

Using Amper Variables as Function Parameters

How you treat numeric amper variables when passing them to a function depends on the data type of the function parameter.

Using a Numeric Amper Variable as a Numeric Function Parameter

When using a numeric amper variable as a numeric parameter in a function call, the amper variable is treated as a field. Since as a field, it has no specified type in either the Master File or the FOCEXEC, it takes the default data type of double precision.

Note that when the result is returned to an output variable, its type is determined by its content. If it has only numbers and a decimal point, it is numeric. If it contains other symbols, it is alphanumeric.

For example, the FTOA function converts a double or single precision number (D or F) to an alphanumeric string with the format specified within the parentheses of the second parameter:

```
FTOA (number_to_convert, '(format)', 'alpha_output_format' )
```

The following example sets &C to 123.55 and passes it to the FTOA function to be converted to an alphanumeric string with a dollar sign:

```
-SET &C=123.55;
-SET &G=FTOA(&C,'(D7.2M)','A11');
-TYPE &G &G.TYPE
```

The output shows that the string $123.55 has been returned to &G. Since it has a symbol other than numeric digits and a decimal point, its type is alphanumeric:

```
$123.55    A
```

In the following example, the format returned does not specify a dollar sign:

```
-SET &A=12345;
-SET  &G=FTOA(&A/100,'(D7.2)','A11');
-TYPE &G &G.TYPE
```

Since the returned string contains only numeric digits and a decimal point, its type is numeric:

```
123.45    N
```

Note that if the number had another digit, it would be returned with a comma, and its type would be alphanumeric:

```
-SET &A=123456;
-SET  &G=FTOA(&A/100,'(D7.2)','A11');
-TYPE &G &G.TYPE
```

The output is:

```
1,234.56    A
```

Using a Numeric Amper Variable as an Alphanumeric Function Parameter

When using a numeric amper variable as an alphanumeric parameter in a function call, you must convert the numeric value to an alphanumeric string before using it in order to avoid failure due to a format error. You can do this using one of the functions designed to convert numerics to alphanumeric, or you can concatenate an alphanumeric character to the numeric value in order to assign it an alphanumeric data type.

For example, the following converts &C to a string and returns the string to the variable &G. It then passes &G to the RJUST function, which right justifies the value and returns it to the variable &H:

```
-SET &C=123.55;
-SET  &G=FTOA(&C,'(D7.2M)','A11');
-SET &H = RJUST(11,&G,'A11');
-TYPE &G &G.TYPE
-TYPE &H &H.TYPE
```

The output is:

```
$123.55   A
  $123.55 A
```

Controlling the Execution of a Procedure

You can use Dialogue Manager commands to manage the execution of a procedure. The commands used for this purpose are -RUN, -EXIT, -QUIT, and -QUIT FOCUS.

- -RUN causes immediate execution of all stacked commands, closes any external files, and continues the procedure. For more information, see *Executing Stacked Commands and Continuing the Procedure* on page 389.
- -EXIT forces the execution of stacked commands, and closes the procedure. For more information, see *Executing Stacked Commands and Exiting the Procedure* on page 390.
- -QUIT cancels execution of any stacked commands and causes an immediate exit from the procedure. For more information, see *Canceling the Execution of a Procedure* on page 391.
- -QUIT FOCUS terminates a procedure and exits FOCUS. For more information, see *Canceling the Execution of a Procedure* on page 391.

Executing Stacked Commands and Continuing the Procedure

You can execute stacked commands and continue the procedure with the -RUN command.

The -RUN command causes immediate execution of all stacked commands and closes any external files opened with -READ or -WRITE. Following execution of the stacked commands, processing of the procedure continues with the line that follows -RUN.

Example: **Executing Stacked Commands and Continuing the Procedure**

The following illustrates the use of -RUN to execute stacked code and then return to the procedure. The numbers to the left correspond to the notes explaining the code.

```
1. TABLE FILE SALES
    PRINT PROD_CODE UNIT_SOLD
    BY CITY
   END
2. -RUN
   TABLE FILE EMPLOYEE
    PRINT LAST_NAME FIRST_NAME
    BY DEPARTMENT
   END
```

The procedure processes as follows:

1. The first four lines are the report request. Each line is placed on a stack to be executed later.

2. -RUN causes the stacked commands to be executed. They are sent to the WebFOCUS Reporting Server for processing. Processing continues with the line following -RUN.

Executing Stacked Commands and Exiting the Procedure

You can execute stacked commands and then exit a procedure with the -EXIT command. -EXIT forces the execution of stacked commands as soon as it is encountered.

- In WebFOCUS, -EXIT closes all external files and terminates the procedure. If the procedure was called by another procedure, control returns to the calling procedure.

Example: Executing Stacked Commands and Exiting the Procedure

In the following, the first report request or the second report request will execute, but not both.

```
1. -SET &PROC = 'SALES'
2. -IF &PROC EQ 'EMPLOYEE' GOTO EMPLOYEE;
   -SALES
3. TABLE FILE SALES
   SUM UNIT_SOLD
    BY PROD_CODE
   END
4. -EXIT
   -EMPLOYEE
   TABLE FILE EMPLOYEE
   PRINT LAST_NAME
    BY DEPARTMENT
   END
```

The procedure processes as follows:

1. Dialogue Manager passes SALES to &PROC.
2. An -IF test is done, and since the value for &PROC is not EMPLOYEE, the test fails and control is passed to the next line, -SALES.

   If the value for &PROC had been EMPLOYEE, control would pass to -EMPLOYEE.
3. The FOCUS code is processed, and stacked to be executed later.
4. -EXIT executes the stacked commands. The output is sent to the WebFOCUS Client application and the procedure is terminated.

   The request under the label -EMPLOYEE is not executed.

This example also illustrates an implicit exit. If the value of &PROC was EMPLOYEE, control would pass to the label -EMPLOYEE after the -IF test, and the procedure would never encounter -EXIT. The TABLE FILE EMPLOYEE request would execute and the procedure would automatically terminate.

Canceling the Execution of a Procedure

You can cancel the execution of a procedure with the -QUIT command. -QUIT cancels execution of any stacked commands and causes an immediate exit from the procedure. Control returns directly to the WebFOCUS application regardless of whether the procedure was called by another procedure. This command is useful if tests or computations generate results that make additional processing unnecessary.

Syntax: How to Cancel the Execution of a Procedure

```
-QUIT
```

Syntax: How to Cancel the Execution of a Procedure and Exit FOCUS

```
-QUIT FOCUS [n]
```

where:

n

Is the operating system return code number. It can be a constant or variable. A variable should be an integer. If you do not supply a value or if you supply a non-integer value, a default return code is posted to the operating system, indicating abnormal termination.

A major function of user-controlled return codes is to detect processing problems. The return code value determines whether to continue or terminate processing. This is particularly useful for batch processing.

Example: Canceling the Execution of a Procedure

The following example illustrates the use of -QUIT to cancel execution based on the results of an -IF test:

```
1. -DEFAULT &CODE='B11';
2. -IF &CODE EQ '0' OR &CODE EQ 'DONE' GOTO QUIT;
3. TABLE FILE SALES
   SUM UNIT_SOLD
   WHERE PROD_CODE EQ &CODE
   END
4. -QUIT
```

The procedure processes as follows:

1. The -DEFAULT command sets the default value for &CODE to B11.
2. The value B11 is passed to &CODE.
3. The FOCUS code is processed and stacked to be executed later.
4. -QUIT cancels the execution of stacked commands and exits the procedure.

Navigating a Procedure

You can navigate a procedure in the following ways:

- **Unconditional branching.** Transfers control to a label.
- **Conditional branching.** Transfers control to a label depending on the outcome of a test.
- **Looping.** Performs a function repeatedly in your procedure.
- **Calling another procedure.** Incorporates a whole or partial procedure into your procedure.

Performing Unconditional Branching

You can perform unconditional branching which transfers control to a label with the -GOTO command.

The first time through a procedure, Dialogue Manager notes the addresses of all of the labels it encounters so that they can be found immediately if needed again. However, If it encounters a -GOTO command that bypasses certain labels, those labels do not go on the list. When Dialogue Manager processes a -GOTO command, it first checks its list of labels. If the target label is already on the list, control is transferred to that label. If the target label is not on the list, Dialogue Manager searches forward through the procedure for the target label. If it reaches the end without finding the label, it continues the search from the beginning of the procedure. Dialogue Manager takes no action on labels that do not have a corresponding -GOTO.

If a -GOTO does not have a corresponding label, execution halts and an error message is displayed.

Note: It is recommended that you not use Dialogue Manager commands as labels.

Syntax: How to Perform Unconditional Branching

```
-GOTO label
  .
  .
  .
-label [TYPE text]
```

where:

-label

Is a user-defined name of up to 64 characters. Do not use embedded blanks or the name of any other Dialogue Manager command except -QUIT or -EXIT. Do not use arithmetic or logical operations, words that can be confused with functions, or reserved words.

Note: The word CONTINUE can be used as a label in a -GOTO that is not part of a -IF command, but CONTINUE will not be recognized as a label in a -IF command, where it always transfers to the command immediately following the -IF.

The *label* text may precede or follow the -GOTO command in the procedure.

Note: When the *label* follows the -GOTO command, a dash does not precede it.

TYPE *text*

Sends a message to a client application.

Example: Performing Unconditional Branching

The following example comments out all the FOCUS code using an unconditional branch rather than -* in front of every line:

```
-GOTO DONE
TABLE FILE SALES
PRINT UNIT_SOLD RETURNS
BY PROD_CODE,CITY
END
-RUN
-DONE
```

Performing Conditional Branching

Conditional branching performs a test of the values of variables and, based on the test, transfers control to a label in the procedure with the -IF... GOTO command. This helps control the execution of requests and builds a dynamic procedure by choosing to execute or not execute parts of a procedure.

For example, you can check whether an extract file was created from a production data source. If the extract file exists, the program runs a set of reports against the extract. If it does not exist, the program branches around the reports and writes a message to a log file.

Note: An -IF test does not require that each test specify a target label.

Syntax: How to Perform Conditional Branching

```
-IF expression [THEN] {GOTO label1|CONTINUE} [ELSE IF...;] [ELSE {GOTO
label2|CONTINUE}];
```

where:

expression

Is a valid expression. Literals do not need to be enclosed in single quotation marks unless they contain embedded blanks or commas.

THEN

Is an optional command that increases readability of the command.

label1

Is a user-defined name of up to 64 characters to which to pass control if the -IF test is true. Do not use embedded blanks or the name of any other Dialogue Manager command except -QUIT or -EXIT. Do not use arithmetic or logical operations, words that can be confused with functions or reserved words. The word CONTINUE can be used as a label in a -GOTO that is not part of a -IF command, but CONTINUE will not be recognized as a label in a -IF command, where it always transfers to the command immediately following the -IF.

The *label* text may precede or follow the -IF criteria in the procedure.

CONTINUE

Continues to the command that follows the semicolon of the -IF command.

Note: CONTINUE cannot be used as a label in a -IF statement.

ELSE IF

Specifies a compound -IF test. The command -IF must end with a semicolon to signal that all logic has been specified. For more information, see *Performing a Compound -IF Test* on page 397.

ELSE GOTO

Passes control to *label2* when the -IF test fails.

If a command spans more than one line, continuation lines must begin with a hyphen and one or more spaces.

Example: Performing Conditional Branching

The following example passes control to the label -PRODSALES if &OPTION is equal to S. Otherwise, control passes to the label -PRODRETURNS, the next line in the procedure.

```
-IF &OPTION EQ 'S' GOTO PRODSALES;
-PRODRETURNS
TABLE FILE SALES
PRINT PROD_CODE UNIT_SOLD
BY STORE_CODE
END
-EXIT
-PRODSALES
TABLE FILE SALES
SUM UNIT_SOLD
BY PROD_CODE
END
-EXIT
```

The following command specifies both transfers explicitly:

```
-IF &OPTION EQ 'S' GOTO PRODSALES ELSE
                - GOTO PRODRETURNS;
```

Notice that the continuation line begins with a hyphen and includes a space after the hyphen.

Example: Testing System and Statistical Variables

In the following example, if data (&LINES) is retrieved with the request, then the procedure branches to the label -PRODSALES. Otherwise, it terminates.

```
TABLE FILE SALES
SUM UNIT_SOLD
BY PROD_CODE BY CITY
WHERE TOTAL UNIT_SOLD GE 50
ON TABLE HOLD
END
-IF &LINES NE 0 GOTO PRODSALES;
-EXIT
-PRODSALES
TABLE FILE SALES
SUM UNIT_SOLD
BY PROD_CODE ACROSS CITY
END
```

Example: Executing Stacked Commands and Exiting the Procedure

In the following example, the first report request or the second report request, but not both, will execute. Assume the report is launched by an HTML form that uses a text input variable named &PROC to prompt the user for the procedure to run. The user may enter SALES or EMPLOYEE.

```
1. -IF &PROC EQ 'EMPLOYEE' GOTO EMPLOYEE;
2. -SALES
   TABLE FILE SALES
   SUM UNIT_SOLD
    BY PROD_CODE
   END
3. -EXIT
   -EMPLOYEE
   TABLE FILE EMPLOYEE
   PRINT LAST_NAME
    BY DEPARTMENT
   END
```

The procedure processes as follows:

1. Dialogue Manager passes SALES to &PROC. An -IF test is done, and since the value for &PROC is not EMPLOYEE, the test fails and control is passed to the next line, -SALES.

   If the value for &PROC had been EMPLOYEE, control would pass to -EMPLOYEE.

2. The FOCUS code is processed, and stacked to be executed later.
3. -EXIT executes the stacked commands. The output is sent to the WebFOCUS Client application, and the procedure is terminated.

   The request under the label -EMPLOYEE is not executed.

This example also illustrates an implicit exit. If the value of &PROC was EMPLOYEE, control would pass to the label -EMPLOYEE after the -IF test, and the procedure would never encounter -EXIT. The TABLE FILE EMPLOYEE request would execute and the procedure would automatically terminate.

Example: Canceling the Execution of a Procedure With Conditional Branching

The following example illustrates the use of -QUIT to cancel execution based on the results of an -IF test. Assume the report is launched by an HTML form that uses a text input variable named &SELECT to prompt the user for the procedure to run.

```
-IF &SELECT EQ 'DONE' GOTO QUIT;
-REPORT
   TABLE FILE SALES
   SUM UNIT_SOLD
   BY PROD_CODE
   WHERE AREA EQ '&SELECT';
   END
-RUN
-QUIT
```

The user can run sequential reports by hitting the Back button in the browser and entering a new value for SELECT and resubmitting the form. If a product code is selected, the procedure continues to the next line and the request executes before encountering -QUIT. When the user enters DONE, control passes to -QUIT. The procedure is exited.

Performing a Compound -IF Test

A compounded -IF test is a series of -IF tests nested within each other. You can use a compound -IF test if each test specifies a target label.

Example: **Using a Compound -IF Test**

In the following example, if the value of &OPTION is neither R nor S, the procedure terminates (-GOTO QUIT). -QUIT serves both as a target label for the GOTO and as an executable command. Assume the report is launched by an HTML form that uses a text input variable named &OPTION to prompt the user for the procedure to run.

```
-IF &OPTION EQ 'R' THEN GOTO PRODRETURNS ELSE IF
- &OPTION EQ 'S' THEN GOTO PRODSALES ELSE
- GOTO QUIT;
-PRODRETURNS
TABLE FILE SALES
PRINT PROD_CODE UNIT_CODE
BY STORE_CODE
END
-EXIT
-PRODSALES
TABLE FILE SALES
SUM UNIT_SOLD
BY PROD_CODE
END
-QUIT
```

Looping in a Procedure

You can perform a function repeatedly by using looping in your procedure with the -REPEAT command. Looping can be used for many tasks. For example, files can be named and renamed by embedding operating system calls within a Dialogue Manager loop. Indexed variables can also be populated using a loop, or the output of a request can be used for a second request.

A process loop can be executed a designated number of times or until a condition is met. A loop ends when any of the following occurs:

- It is executed in its entirety.
- A -QUIT or -EXIT command is issued.
- A -GOTO is issued to a label outside of the loop.

  Note: If you issue another -GOTO later in the procedure to return to the loop, the loop proceeds from the point it left off.

You can also limit the repetition of a loop by incrementing a variable with the -SET command.

Syntax: How to Specify a Loop

```
-REPEAT label n TIMES
```

or

```
-REPEAT label WHILE condition;
```

or

```
-REPEAT label FOR &variable [FROM fromval] [TO toval] [STEP s]
```

where:

label

Identifies the code to be repeated (the loop). A label can include another loop if the label for the second loop has a different name than the first.

n TIMES

Specifies the number of times to execute the loop. The value of *n* can be a local variable, a global variable, or a constant. If it is a variable, it is evaluated only once, so you cannot change the number of times to execute the loop. The loop can only be ended early using -QUIT or -EXIT.

WHILE *condition*

Specifies the condition under which to execute the loop. The condition is any logical expression that can be true or false. The loop executes if the condition is true.

Note: The condition must be followed by a semi-colon.

&variable

Is a variable that is tested at the start of each execution of the loop. It is compared with the value of *fromval* and *toval*, if supplied. The loop is executed only if *&variable* is less than or equal to *toval* when s is positive, or greater than or equal to *toval* when *s* is negative.

fromval

Is the minimum value of *&variable* that will execute a loop. 1 is the default value.

toval

Is the maximum value of *&variable* that will execute a loop. 1,000,000 is the default value.

STEP *s*

Increments the value of *&variable* by a constant, s. It may be positive or negative. The default increment is 1.

Note: The parameters FROM, TO, and STEP can appear in any order.

Example: Repeating a Loop

These examples illustrate each syntactical element of -REPEAT.

```
-REPEAT label n TIMES
```

For example:

```
-REPEAT LAB1 2 TIMES
-TYPE INSIDE
-LAB1 TYPE OUTSIDE
```

The output is:

```
INSIDE
INSIDE
OUTSIDE
```

```
-REPEAT label WHILE condition;
```

For example:

```
-SET &A = 1;
-REPEAT LABEL WHILE &A LE 2;
-TYPE &A
-SET &A = &A + 1;
-LABEL TYPE END: &A
```

The output is:

```
1
2
END: 3
```

```
-REPEAT label FOR &variable FROM fromval TO toval STEP s
```

For example:

```
-REPEAT LABEL FOR &A STEP 2 TO 4
-TYPE INSIDE &A
-LABEL TYPE OUTSIDE &A
```

The output is:

```
INSIDE 1
INSIDE 3
OUTSIDE 5
```

Example: Controlling Loops

The following example illustrates the use of -SET to control a loop:

```
1. -DEFAULT &N=0
2. -START
3. -SET &N=&N+1;
4.   EX SLRPT
   -RUN
5. -IF &N GT 5 GOTO NOMORE;
6. -GOTO START
5. -NOMORE TYPE EXCEEDING REPETITION LIMIT
   -EXIT
```

The procedure executes as follows:

1. The -DEFAULT command gives &N the initial value of 0.
2. -START begins the loop. This is also the target of an unconditional -GOTO.
3. The -SET command increments the value of &N by one each time the loop executes.
4. The FOCUS command EX SLRPT is stacked. -RUN then executes the stacked command.
5. The -IF command tests the current value of the variable &N. If the value is greater than 5, control passes to the label -NOMORE, which displays a message for the end user and forces an exit. If the value of &N is 5 or less, control goes to the next Dialogue Manager command.
6. -GOTO passes control to the -START label, and the loop continues.

Calling Another Procedure With -INCLUDE

You can call a procedure from another procedure with the -INCLUDE command, which can incorporate a whole or partial procedure. A partial procedure might contain heading text, or code that should be included at run time based on a test in the calling procedure. It executes immediately when encountered.

A calling procedure cannot branch to a label in a called procedure, and vice versa. When a procedure is included using the -INCLUDE command, the procedure being included (called) has full access to variables defined in the calling procedure.

Procedures called using the -INCLUDE command must be in the search path of the WebFOCUS Reporting Server.

Reference: Uses for -INCLUDE

The -INCLUDE command can be used for the following:

- Controlling the environment. For example, the called procedure may set variables such as server name or user name before the calling procedure continues execution.
- As a security mechanism. The included procedure can be encrypted and a direct password set.
- Shortening the code when there are several possible procedures that may be called. For example, the command -INCLUDE &NEWLINES could be used to determine the called procedure, reducing the number of GOTO commands.
- Continuing sections of code used throughout the application such as standard headings and footings. This enables changes made in a single module effect the entire application.

Syntax: How to Call Another Procedure With -INCLUDE

```
-INCLUDE filename
```

where:

filename

Is the name of the called procedure:

- On UNIX and Windows platforms, the procedure must have the extension .fex.
- On UNIX, *filename* must specify the absolute path and file name for the called procedure. For example, F:\\ibi\apps\sales\headings.fex.

- On z/OS, the included file must reside in the EDARPC library.

Example: Calling Another Procedure With -INCLUDE

In the following example, Dialogue Manager searches for a procedure named DATERPT as specified on the -INCLUDE command.

```
-IF &OPTION EQ 'S' GOTO PRODSALES
-ELSE GOTO PRODRETURNS;
   .
   .
   .
-PRODRETURNS
-INCLUDE DATERPT
-RUN
   .
   .
   .
```

Assume that DATERPT contains the following code, which Dialogue Manager incorporates into the original procedure. Dialogue Manager substitutes a value for the variable &PRODUCT as soon as the -INCLUDE is encountered. -RUN executes the request.

```
TABLE FILE SALES
PRINT PROD_CODE UNIT_SOLD
WHERE PROD_CODE = '&PRODUCT';
END
```

Example: Calling a Procedure With a Heading

The following incorporates a heading, which is stored as a procedure:

```
TABLE FILE SALES
-INCLUDE SALEHEAD
SUM UNIT_SOLD AND RETURNS AND COMPUTE
   .
   .
   .
```

The file SALEHEAD contains:

```
HEADING
"THE ABC CORPORATION"
"RETAIL SALES DIVISION"
"MONTHLY SALES REPORT"
```

This heading is included in the report request.

Example: Calling a Procedure for a Virtual Field

The following incorporates a virtual field from a procedure:

```
-INCLUDE DEFRATIO
   TABLE FILE SALES
-INCLUDE SALEHEAD
   SUM UNIT_SOLD AND RETURNS AND RATIO
   BY CITY
   .
   .
   .
```

The file DEFRATIO creates a virtual field:

```
DEFINE FILE SALES
RATIO/D5.2=(RETURNS/UNIT_SOLD);
END
```

This virtual field will be dynamically included before the report request executes.

Syntax: How to Call Another Procedure in a Different Application

If the procedure (FOCEXEC) that is included in another procedure is in a different application, issue the command:

```
-INCLUDE APPNAME/FOCEXEC
```

Using Fully Qualified Names With the -Include Command

The -INCLUDE Dialogue Manager command can accept fully qualified file names so that files outside of the standard search path can be inserted into procedures. This technique reduces the time it takes to search for a specific procedure, providing a performance benefit.

Relative paths are not supported in the -INCLUDE command and must use the fully qualified platform-specific file name. The limit for a fully qualified platform-specific file name is defined by the operating system (as long as it is within the 32K length maximum for a FOCEXEC line).

The fully qualified name applies only to the -INCLUDE command in which it is specified. It is not inherited by other -INCLUDE commands.

Syntax: How to Use a Fully Qualified File Name on UNIX

```
-INCLUDE /path/filename.ext
```

where:

path

Is the fully qualified path to the file that contains the FOCEXEC.

filename

Is the name of the file that contains the FOCEXEC.

ext

Is the extension of the file that contains the FOCEXEC.

Syntax: How to Use a Fully Qualified File Name on Microsoft Windows

```
-INCLUDE drive:\path\filename.ext
```

where:

drive

Is the drive that contains the path to the FOCEXEC.

path

Is the fully qualified path to the file that contains the FOCEXEC.

filename

Is the name of the file that contains the FOCEXEC.

ext

Is the extension of the file that contains the FOCEXEC.

Example: Using a Fully Qualified Name in a -INCLUDE Command

Assume the FOCEXEC named HEADINGS contains the following heading text for the request:

```
HEADING
"THIS IS THE INCLUDED HEADING FILE"
" "
```

Microsoft Windows example:

```
TABLE FILE EMPLOYEE
-INCLUDE c:\ibi\srv82\mydataarea\headings.fex
PRINT CURR_SAL BY LAST_NAME BY FIRST_NAME
WHERE DEPARTMENT EQ 'MIS'
END
```

UNIX example:

```
TABLE FILE EMPLOYEE
-INCLUDE /u2/prog/headings.data
PRINT CURR_SAL BY LAST_NAME BY FIRST_NAME
WHERE DEPARTMENT EQ 'MIS'
END
```

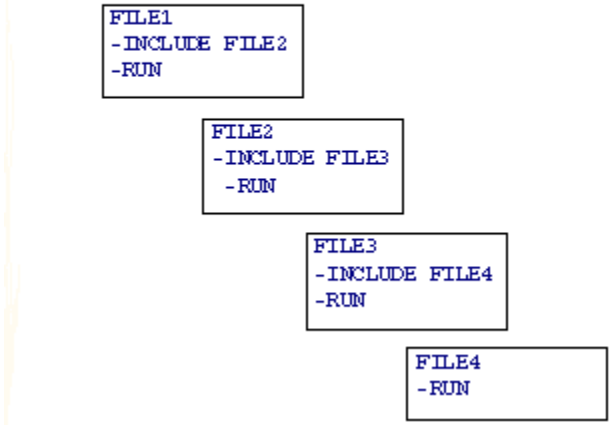
The output is:

```
THIS IS THE INCLUDED HEADING FILE
LAST NAME      FIRST NAME  CURR SAL
BLACKWOOD      ROSEMARIE   $21,780.00
CROSS          BARBARA     $27,062.00
GREENSPAN      MARY         $9,000.00
JONES          DIANE       $18,480.00
MCCOY          JOHN        $18,480.00
SMITH          MARY        $13,200.00
```

Nesting Procedures With -INCLUDE

Any number of different procedures can be invoked from a single calling procedure. You can also nest -INCLUDE commands within each other as seen here. There is no limit to the number of -INCLUDE commands that can be nested.

```
-PRODSALES
-INCLUDE FILE1
-RUN
```



Files one through four are incorporated into the original procedure. All of the included files are viewed as part of the original procedure.

Calling Another Procedure With EXEC

You can call a procedure from another procedure with the EXEC command. The called procedure must be fully executable. It behaves as a completely separate procedure, with its own content. It cannot use any local variables (&variables) defined by the *calling* procedure (unless they are explicitly passed to the *called* procedure on the command line). However, the executed (called) procedure can use any global variables (&&variables) that have been defined in the calling procedure.

When an EXEC command is encountered, it is stacked and executed when the appropriate Dialogue Manager command is encountered. The called procedure must be fully executable.

Procedures called using the -INCLUDE command must be in the search path of the WebFOCUS Reporting Server.

Syntax: How to Call a Procedure With the EXEC Command

```
EX[EC] procedure
```

where:

procedure

Is the name of the procedure.

Note: When EXEC is used in Managed Reporting, it is important to note that there is a difference between EX and EXEC. EX statements coded in a procedure are processed by the WebFOCUS Client, which looks for the procedure in the Managed Reporting repository. Procedures that are referenced with an EXEC statement are not processed by the WebFOCUS Client, they are only passed to the WebFOCUS Reporting Server for processing, and these procedures are not looked for in the Managed Reporting repository.

Example: Calling a Procedure With EXEC

In the following example, a procedure calls DATERPT:

```
-IF &OPTION EQ 'S' GOTO PRODSALES ELSE GOTO PRODRETURNS;
    .
    .
    .
-PRODRETURNS
    EX DATERPT
    .
    .
    .
-RUN
```

Enhancing an HTML Webpage With a Procedure

You can enhance the functionality of your webpage by enabling you to include HTML commands in your procedures using the Dialogue Manager -HTMLFORM command. -HTMLFORM supports all standard HTML elements, including character styling, hyperlinks, graphic images, tables, forms, and frames. You can combine a procedure and HTML commands in the following ways:

- Refer to a webpage file that is merged with your report output from a procedure.
- Embed HTML commands in a procedure. To do this, enter the appropriate syntax in your procedure using an editor.
- Embed a procedure in HTML commands.
- Include local and global variables, identified with WebFOCUS escape sequences, in your webpage.

Referring to an External Webpage

You can include HTML commands in your procedure by specifying an HTML webpage in your procedure.

Syntax: How to Refer to an External Webpage From a Procedure

```
-HTMLFORM filename
```

where:

filename

Is the HTML file that contains the target webpage. The name of the HTML file must be in uppercase letters.

- In UNIX and Windows, this is the 8-character file name of the HTML file that contains the webpage. The file extension must be .HTM, not .HTML.
- In Windows and UNIX, the file extension must be .HTM.
- In z/OS, this is either the member name of a file in the PDS allocated to ddname WWWHTM, or it is the ddname of a sequential file. The file is variable blocked (VB).
- An HTML file called by -HTMLFORM must reside in a path defined by the WebFOCUS Reporting Server.

If you use separate HTML files, you cannot use Dialogue Manager commands and variables in the file. However, the WebFOCUS escape sequences, described in *Including Variables in an HTML Webpage* on page 415, can be used.

Embedding HTML Commands in a Procedure

You can include HTML commands in a procedure by embedding those commands in a procedure.

Note:

- The -RUN command cannot be included in the lines between the -HTMLFORM BEGIN and -HTMLFORM END commands.

Syntax: How to Embed HTML Commands in a WebFOCUS Procedure

```
-HTMLFORM BEGIN
        .
        .
        .
-HTMLFORM END
```

where:

`-HTMLFORM BEGIN`

Indicates the beginning of HTML commands.

`-HTMLFORM END`

Indicates the end of HTML commands.

Note: All lines that are not Dialogue Manager commands are assumed to be HTML.

All Dialogue Manager variables referenced within the `-HTMLFORM BEGIN` and `-HTMLFORM END` commands are evaluated, even within comment tags, so you should either assign them a default value or escape the ampersand symbol using a pipe character (|) so that it is not interpreted as the start of a variable name. For example, in the following sample, the ampersand is escaped using the pipe character so that the variable definition in the comment is not evaluated:

```
-HTMLFORM BEGIN
<HTML>
<HEAD>
<SCRIPT>
/* escaping the ampersand in a comment: &|x=y */
</SCRIPT>
</HEAD>
<BODY>
Test variable in Javascript comments
</BODY>
</HTML>
-HTMLFORM END
```

Example: Including a Webpage in a Procedure

The following procedure contains a report request and the -HTMLFORM command, which customizes the webpage the report is displayed on. The report is centered in the browser window, and displays with the Arial font in blue.

```
TABLE FILE SHORT
SUM PROJECTED_RETURN
BY COUNTRY
ON TABLE HOLD AS SHORT FORMAT HTMTABLE
END

-HTMLFORM BEGIN
<HTML><HEAD><STYLE>TD {FONT-FAMILY: ARIAL; COLOR: BLUE; }</STYLE></HEAD>
<BODY><DIV ALIGN="CENTER">
!IBI.FIL.SHORT;
</DIV></BODY></HTML>
-HTMLFORM END
```

The following webpage is produced:

PAGE 1

Country	Projected Annualized Return
ARGENTINA	4.200
BRAZIL	4.200
CANADA	3.990
CZECH REP	12.110
ENGLAND	4.060
FRANCE	3.920
GUATEMALA	4.760
HONDURAS	4.760
HONG KONG	4.550
ISRAEL	3.780
JAPAN	4.550
MEXICO	5.040
SAUDI ARABIA	4.200
SKLARVIA	4.040
UNITED STATES	3.430

Embedding a Procedure in an HTML Webpage

You can embed one or more reports or HTML files into your webpage using -HTMLFORM commands.

Procedure: How to Embed a Report in an HTML Webpage

To embed a report into a webpage created with -HTMLFORM commands:

1. Create a report procedure that retrieves and formats the report data, then saves the output from *each* report in a separate temporary file. The following syntax is used in the report procedure:

   ```
   ON TABLE HOLD FORMAT HTMTABLE AS reportname
   ```

Note: The temporary file created for the output is not available to the user.

2. If you are using an external HTML file to display the webpage that embeds the report, the report procedure must specify the file using the following command:

   ```
   -HTMLFORM filename
   ```

   Note: If you are not using an external HTML file, the HTML commands for the webpage can reside in the same file using -HTMLFORM BEGIN and -HTMLFORM END to separate the HTML commands from the rest of the procedure.

3. In the HTML commands for the webpage, specify the location where *each* report or HTML file is to be displayed, using an HTML escape code. The escape code syntax is

   ```
   !IBI.FIL.reportname
   ```

   where:

   reportname

   Is the name of the report file to embed in the Web page.

   Note: The designated report name can be up to 512 characters.

   Files embedded with the !IBI.FIL escape sequence can contain other escape sequences. If reference is made to a file that does not exist, an error message is returned.

Example: Embedding Multiple Reports Into One HTML Webpage

This example shows how to embed multiple HTML reports into one HTML webpage. The following code generates three reports, each consisting of the same automotive data that is sorted three different ways; by country, by body type, and by manufacturer.

```
TABLE FILE CAR
"REPORT 1 - BY COUNTRY"
PRINT CAR MODEL BODYTYPE
BY COUNTRY
ON TABLE HOLD FORMAT HTMTABLE AS REPORT1
ON TABLE SET STYLE *
TYPE=HEADING,STYLE=BOLD,COLOR=RED,$
ENDSTYLE
END
TABLE FILE CAR
"REPORT 2 - BY BODY TYPE"
PRINT CAR MODEL COUNTRY
BY BODYTYPE
ON TABLE HOLD FORMAT HTMTABLE AS REPORT2
ON TABLE SET STYLE *
TYPE=HEADING,STYLE=BOLD,COLOR=BLUE,$
ENDSTYLE
END
TABLE FILE CAR
"REPORT 3 - BY MANUFACTURER"
PRINT MODEL BODYTYPE COUNTRY
BY CAR
ON TABLE HOLD FORMAT HTMTABLE AS REPORT3
ON TABLE SET STYLE *
TYPE=HEADING,STYLE=BOLD,COLOR=GREEN,$
ENDSTYLE
END
-HTMLFORM BEGIN
!IBI.FIL.REPORT1;
!IBI.FIL.REPORT2;
!IBI.FIL.REPORT3;
-HTMLFORM END
```

The following images show the three reports that are embedded into one HTML document:

PAGE 1

REPORT 1 - BY COUNTRY

COUNTRY	CAR	MODEL	BODYTYPE
ENGLAND	JAGUAR	V12XKE AUTO	CONVERTIBLE
	JAGUAR	XJ12L AUTO	SEDAN
	JENSEN	INTERCEPTOR III	SEDAN
	TRIUMPH	TR7	HARDTOP
FRANCE	PEUGEOT	504 4 DOOR	SEDAN
ITALY	ALFA ROMEO	2000 4 DOOR BERLINA	SEDAN
	ALFA ROMEO	2000 GT VELOCE	COUPE
	ALFA ROMEO	2000 SPIDER VELOCE	ROADSTER
	MASERATI	DORA 2 DOOR	COUPE
JAPAN	DATSUN	B210 2 DOOR AUTO	SEDAN
	TOYOTA	COROLLA 4 DOOR DIX AUTO	SEDAN
W GERMANY	AUDI	100 LS 2 DOOR AUTO	SEDAN
	BMW	2002 2 DOOR	SEDAN
	BMW	2002 2 DOOR AUTO	SEDAN
	BMW	3.0 SI 4 DOOR	SEDAN
	BMW	3.0 SI 4 DOOR AUTO	SEDAN
	BMW	530I 4 DOOR	SEDAN
	BMW	530I 4 DOOR AUTO	SEDAN

PAGE 1

REPORT 2 - BY BODY TYPE

BODYTYPE	CAR	MODEL	COUNTRY
CONVERTIBLE	JAGUAR	V12XKE AUTO	ENGLAND
COUPE	ALFA ROMEO	2000 GT VELOCE	ITALY
	MASERATI	DORA 2 DOOR	ITALY
HARDTOP	TRIUMPH	TR7	ENGLAND
ROADSTER	ALFA ROMEO	2000 SPIDER VELOCE	ITALY
SEDAN	JAGUAR	XJ12L AUTO	ENGLAND
	JENSEN	INTERCEPTOR III	ENGLAND

Embedding Variables Into One HTML Webpage

This example generates a report consisting of product codes and quantities, and displays the current time and date found in Dialogue Manager system variables. The following explains how the elements required to display this report are combined. The numbers to the left of the example apply to the notes that follow.

Step 1 - First.htm

Set up the webpage that will hold the report. This webpage must be saved in a directory within EDAPATH or APPPATH in order for WebFOCUS to find it. The following file is called first.htm:

```
   <HTML>
   <BODY>
2. Time:
3. !IBI.FIL.&TOD <BR>
2. Date:
3. !IBI.FIL.&DATE <BR>
   <HR SIZE=5>
4. <!--WEBFOCUS TABLE UPPER-->
   </BODY>
   </HTML>
```

Note: The HTML comment line must be closed with the –> characters or a single > character and should not have any other HTML tags within it.

Step 2 - Amper.fex

Create your report request:

```
    TABLE FILE GGORDER
    SUM QUANTITY BY PCD
    IF PCD EQ 'B$$$'
1. ON TABLE HOLD FORMAT HTMTABLE AS UPPER
    END
5. -RUN
6. -HTMLFORM first
```

Step 3 - Launch.htm

Create a webpage, which launches the report procedure:

```
<HTML>
<BODY>
<P>
<P>
<A HREF="/ibi_apps/WFServlet?IBIF_ex=amper">Click here.</A>
</BODY>
</HTML>
```

When the procedure and both webpages have been created, the webpage that launches the procedure can be called from the browser. When the procedure is launched, the report is executed:

1. The HOLD command extracts report output into a temporary file, from which it can be sent to a webpage.
2. The code Time: and Date: specify text that will appear above the report on the webpage.
3. The HTML comments are read by WebFOCUS and identify the variables to be printed on the webpage.
4. The HTML comment is read by WebFOCUS and identifies the report to be displayed. This table name is the name assigned in the HOLD command in the procedure. On the webpage, WebFOCUS substitutes the designated report for the corresponding code.
5. The -RUN command runs the report.
6. The Dialogue Manager command merges the report output with the contents of first.htm, the HTML file that contains your webpage.

The following webpage is generated:

Time: 10.29.36
Date: 08/30/01

PAGE 1	
Product Code	Ordered Units
B141	100427
B142	285689
B144	61498

Including Variables in an HTML Webpage

In WebFOCUS, you can use Dialogue Manager variables to insert variable values in a webpage. This is done by specifying the variable name using a WebFOCUS escape code. You can use the value of a variable either as text on an HTML webpage or as a value in an HTML form input field. You can include a variable that you have created, or system variables like &DATE (today's date) and &TOD (time of day). A single webpage can contain many variable substitution tags.

To use a variable in a webpage created with the -HTMLFORM commands, the variable must first be declared with -SET or -DEFAULT. See *Customizing a Procedure With Variables* on page 336 for information on using variables.

Syntax: How to Return the Value of a Variable in an HTML Webpage

```
<!--WEBFOCUS VAR [&]&variable-->
```

or

```
!IBI.{AMP|GLB}.[&]&variable;
```

where:

`AMP`

Returns the value of a local variable.

`GLB`

Returns the value of a global variable.

`variable`

Is a local or global variable.

Note: The HTML comment line must be closed with the –> characters or a single > character and should not have any other HTML tags within it.

Syntax: How to Return the Length of a Variable in an HTML Webpage

```
!IBI.{AML|GLL}.[&]&variable;
```

where:

`AML`

Returns the length of a local variable.

`GLL`

Returns the length of a global variable.

Syntax: How to Use a Variable in a Webpage

The syntax for a local variable in a form input field is

```
<INPUT TYPE=TEXT NAME=variable     VALUE="!IBI.AMP.variable;">
```

where:

variable

Is the variable name.

When making substitutions, the full string is replaced, starting with the exclamation point (!) and ending with the semicolon (;). You cannot include variables whose values include internal escape sequences in your command stream.

Issuing Operating System Commands

You can issue an operating system command to set up an environment in which a request must run. For example, a program may change directories, rename files, unzip files, copy files, or perform other operations before executing a request. After the environment is customized, the program can begin the report and graph requests.

Syntax: How to Execute an Operating System Command

```
op_system command
```

where:

op_system

Specifies the operating system.

`-DOS` or `DOS` specifies the DOS operating system.

`-MVS` or `MVS` specifies the MVS operating system.

`-TSO` or `TSO` specifies the TSO operating system.

`-UNIX` or `UNIX` specifies the UNIX operating system.

`-WINNT` or `WINNT` specifies the Windows operating system.

`-VMS` or `VMS` specifies the OpenVMS operating system.

`-AS/400` or `AS/400` specifies the IBM i operating system, which is also known as AS400 or OS400.

`-SYSTEM` or `!` specifies any operating system.

command

Is an operating system command.

Note:

- The difference between prefixing the operating system with or without a "-" (dash) is that dash prefixed operating system commands can run within a stream of dialogue manager commands without the need for a -RUN to force execution. Additionally, a dash prefixed operating system command is ignored (bypassed) if the command is not actually run on that operating system. For example, if an application runs on several different operating systems and you want to display the current directory, which requires a different command for each operating system, you can simply use the following code:

```
-UNIX pwd
-WINNT chdir
-VMS SHOW DEFAULT
-AS/400 pwd
```

- The -SYSTEM and ! operating system commands have the advantage of not checking which operating system is in use, so when using multiple operating systems where a particular command is the same, for example, the DIR command is the same on Windows and OpenVMS, the -SYSTEM or ! operating system command can simplify coding effort.

Change Directory Operating System Command

The change directory command, which is the cd command on all platforms except OpenVMS (where it is the SET DEFAULT command), may not appear to operate as expected if a second request issues a -UNIX pwd command. An example is when -UNIX cd /tmp seems to be ignored when issued. Operating system commands operate as separate processes, so when the process exits, environmental properties such as variables no longer exist and any change of directory is undone. This behavior is consistent with the non-persistent behavior of server environments that protect one execution from another.

Due to this behavior, a change directory command is typically used as part of a request to execute multiple operating system commands on a single line. The cd command is issued first, then the desired command is issued, so the commands execute as a group. Applications ported from releases prior to Version 7 Release 1.5 or ported from FOCUS may have been coded previously based on persistent behavior and must now be coded to combine the change directory and one or more operating system commands onto a single line.

Executing Multiple Operating System Commands on a Single Line

Most operating systems allow more than one command on a single line by using special separators or surrounding syntax. Each method varies depending on the operating system. The following are examples using the required syntax for executing (stacking) multiple operating system commands on a single line:

For Windows:

```
! cmd /c "cd \tmp && chdir && if exist mydata.txt echo got it"
! cmd /c "cd \tmp & chdir & if exist mydata.txt echo got it "
! cd \tmp & chdir & if exist mydata.txt echo got it
! dir mydata & if exist mydata.txt echo got it
```

For UNIX:

```
! cd /tmp ; pwd ; if [ -f mydata.txt ] ; then echo got it ; fi
! ls mydata.txt ; if [ -f mydata.txt ] ; then echo got it ; fi
```

For OpenVMS:

```
! PIPE DIR /SIZE/DATE mydata.txt | SEARCH SYS$INPUT MYDATA
! PIPE DELETE mydata.txt;* /Noconfirm > NL: 2> NL:
```

For IBM i:

```
! cd /tmp ; pwd ; if [ -f mydata.txt ] ; then echo got it ; fi
! system "DSPLIB QGPL" ; if [ -f mydata.txt ] ; then echo got it ; fi
! ls mydata.txt ; if [ -f mydata.txt ] ; then echo got it ; fi
```

Note: Native IBM i commands can be stacked with QShell commands if the native portion uses the QShell system command with the native command portion enclosed in double quotes.

Reviewing Command Output

Some operating system commands perform a function and do not display additional output. For example, the following commands do not display any output when the command is executed:

```
-UNIX cp a.fex b.fex
-DOS copy a.fex b.fex
```

Other commands return output when the command is executed. For example, the following commands display a list of files in the current directory that ends in fex:

```
-UNIX ls -l grep fex
-DOS dir *.fex
```

WebFOCUS does not display operating system command results in the browser. To review the results of an operating system command, you must redirect the output to another file.

For example, the following commands redirect the command output to a file named OUTPUT (which you can review later in an editor):

```
-UNIX ls -l grep fex > /tmp/output
-DOS dir *.fex > c:\tmp\output
```

Controlling Passwords With Dialogue Manager

You can issue and control passwords with the -PASS command. This is especially useful for specifying a particular file or set of files that a given user can read to or write from. Passwords have detailed sets of functions associated with them through the DBA facility.

The procedure that sets passwords can be encrypted so that it and the passwords that it sets cannot be typed and made known.

A variable can also be associated with -PASS so that you can prompt for and assign a password value.

Syntax: How to Specify a Password

```
-PASS password
```

where:

password

Is a password or a variable containing a password.

Sending a Message to the Application

You can send a message to an application while a procedure is processing with the -TYPE command. You can use a message to do the following:

- Explain the purpose of the procedure.
- Display the results of a procedure or calculation during testing of a procedure.
- Present useful information.
- Pass CGI variables to WebFOCUS. For details, see *Controlling Multiple Reports* on page 266.

In WebFOCUS, depending on the execution of a procedure, the message may be displayed in the browser, or as a comment in the HTML file. To view an HTML file comment, use the appropriate web browser option, for example, View Source in Microsoft Internet Explorer.

Syntax: How to Send a Message

```
-TYPE text-label TYPE text
```

or

```
-label TYPE text
```

where:

text

Is the message to be sent. If you include quotation marks around the text, they are displayed in the message. The length of the message can be up to 495 characters.

In WebFOCUS, the message is included in the HTML source file.

-label

Is the target of a -GOTO or -IF.

Note: Labels are the only Dialogue Manager commands that can be placed on the same line as another Dialogue Manager command.

Example: **Sending a Message**

The following example illustrates the use of -TYPE to inform a client application about the contents of a report:

```
-* Version 1  06/26/00  SLRPT  Procedure
-* Component of Retail Sales Reporting Module
-TYPE This report calculates percentage of returns.
TABLE FILE SALES
 .
 .
 .
END
```

Testing and Debugging a Dialogue Manager Procedure

You can test and debug your procedure with the following.

- The &ECHO variable controls the display of command lines as they execute so you can test and debug procedures.
- The &STACK variable enables you to test the logic of Dialogue Manager commands. Setting this variable to OFF lets you run the procedure while preventing the execution of stacked commands. This gives you the ability to view the sequence of commands and see how the variable values are resolved.
- The &RETCODE variable returns a code after a procedure is executed. If the procedure results in normal output or no records are retrieved, the value of &RETCODE is one. If an error occurs while parsing the procedure, the value of &RETCODE is eight.

&RETCODE can be used to test the result of an operating system command. This retrieves the return code from the operating system.

- The &IORETURN variable tests the result of Dialogue Manager -READ and -WRITE commands. After a -READ or -WRITE operation, a non-zero return code indicates an error, such as end-of-file being reached.

  &IORETURN can be used to test the result of the following:

  - A -READ command. If &IORETURN equals zero, a value was successfully read from the external file.
  - A -WRITE command. If &IORETURN equals zero, a value was successfully written to the external file.

Syntax: How to Test and Debug a Procedure

```
{-DEFAULT|-SET|EX} &ECHO = {ON|ALL|OFF|NONE}
```

where:

`ON`

Displays WebFOCUS commands that are expanded and stacked for execution.

`ALL`

Displays Dialogue Manager commands and WebFOCUS commands that are expanded and stacked for execution.

`OFF`

Suppresses the display of both stacked commands and Dialogue Manager commands. This value is the default.

`NONE`

Prevents procedure code from being displayed (echoed). Once the value of &ECHO has been set to NONE, it cannot be changed during the session or connection.

By default, any procedure that does not explicitly set the &ECHO variable executes with the value OFF. You can change this default value for &ECHO with the SET DEFECHO command, as described in *How to Establish a Default Value for the &ECHO Variable* on page 423.

Syntax: How to Establish a Default Value for the &ECHO Variable

```
SET DEFECHO = {OFF|ON|ALL|NONE}
```

where:

OFF
Establishes OFF as the default value for &ECHO. OFF is the default value.

ON
Establishes ON as the default value for &ECHO. Displays WebFOCUS commands that are expanded and stacked for execution.

ALL
Establishes ALL as the default value for &ECHO. ALL displays Dialogue Manager commands and WebFOCUS commands that are expanded and stacked for execution.

NONE
Prevents procedure code from being displayed (echoed). Once the value of DEFECHO or &ECHO has been set to NONE, it cannot be changed during the session or connection.

Reference: Usage Notes for SET DEFECHO = NONE

- If you issue the SET DEFECHO=NONE command in a FOCEXEC, the setting does not affect &ECHO in that routine. It takes effect as the value of &ECHO in the next executed (EX) procedure after which it may not be changed.
- If you attempt to reset &ECHO within the duration of its NONE value, the value you attempted to set will display if you issue a -TYPE command, but the value will not actually change.

Example: Preventing Procedure Code From Being Displayed

The following procedure queries the value of the DEFECHO parameter and issues a TABLE request against the EMPLOYEE data source:

```
? SET DEFECHO
-RUN
-TYPE ECHO = &ECHO
TABLE FILE EMPLOYEE
PRINT CURR_SAL CURR_JOBCODE
BY LAST_NAME BY FIRST_NAME
END
-RUN
```

The query command output shows that DEFECHO is OFF (the default value):

```
 DEFECHO                  OFF
```

The -TYPE command shows that the value of &ECHO is OFF (the default):

```
ECHO = OFF
```

Because &ECHO is OFF the TABLE commands do not display as the procedure executes:

```
 NUMBER OF RECORDS IN TABLE=       12  LINES=     12
 PAUSE.. PLEASE ISSUE CARRIAGE RETURN WHEN READY
PAGE     1
LAST_NAME        FIRST_NAME         CURR_SAL  CURR_JOBCODE
---------        ----------         --------  ------------
BANNING          JOHN             $29,700.00  A17
BLACKWOOD        ROSEMARIE        $21,780.00  B04
CROSS            BARBARA          $27,062.00  A17
GREENSPAN        MARY              $9,000.00  A07
IRVING           JOAN             $26,862.00  A15
JONES            DIANE            $18,480.00  B03
MCCOY            JOHN             $18,480.00  B02
MCKNIGHT         ROGER            $16,100.00  B02
ROMANS           ANTHONY          $21,120.00  B04
SMITH            MARY             $13,200.00  B14
                 RICHARD           $9,500.00  A01
STEVENS          ALFRED           $11,000.00  A07
                      END OF REPORT
```

Now, set DEFECHO=ON and re-run the procedure.

The query command output shows that DEFECHO is ON:

```
 DEFECHO                  ON
```

The -TYPE command shows that the value of &ECHO has been changed to ON:

```
ECHO = ON
```

Because &ECHO is ON, the TABLE commands display as the procedure executes:

```
TABLE FILE EMPLOYEE
PRINT CURR_SAL CURR_JOBCODE
BY LAST_NAME BY FIRST_NAME
END
```

The output displays next:

```
 NUMBER OF RECORDS IN TABLE=       12  LINES=     12
 PAUSE.. PLEASE ISSUE CARRIAGE RETURN WHEN READY
PAGE     1
LAST_NAME        FIRST_NAME         CURR_SAL  CURR_JOBCODE
---------        ----------         --------  ------------
BANNING          JOHN             $29,700.00  A17
BLACKWOOD        ROSEMARIE        $21,780.00  B04
CROSS            BARBARA          $27,062.00  A17
GREENSPAN        MARY              $9,000.00  A07
IRVING           JOAN             $26,862.00  A15
JONES            DIANE            $18,480.00  B03
MCCOY            JOHN             $18,480.00  B02
MCKNIGHT         ROGER            $16,100.00  B02
ROMANS           ANTHONY          $21,120.00  B04
SMITH            MARY             $13,200.00  B14
                 RICHARD           $9,500.00  A01
STEVENS          ALFRED           $11,000.00  A07
                      END OF REPORT
```

Now, issue the SET DEFECHO = NONE command and rerun the procedure:

```
SET DEFECHO = NONE
```

The query command output shows that the value of DEFECHO has been changed to NONE:

```
 DEFECHO                 NONE
```

The -TYPE command shows that the value of &ECHO is NONE:

```
ECHO = NONE
```

Because DEFECHO has the value NONE, the TABLE commands do not display as the procedure executes. The output is:

```
 NUMBER OF RECORDS IN TABLE=       12  LINES=     12
 PAUSE.. PLEASE ISSUE CARRIAGE RETURN WHEN READY
PAGE     1
LAST_NAME        FIRST_NAME         CURR_SAL  CURR_JOBCODE
---------        ----------         --------  ------------
BANNING          JOHN             $29,700.00  A17
BLACKWOOD        ROSEMARIE        $21,780.00  B04
CROSS            BARBARA          $27,062.00  A17
GREENSPAN        MARY              $9,000.00  A07
IRVING           JOAN             $26,862.00  A15
JONES            DIANE            $18,480.00  B03
MCCOY            JOHN             $18,480.00  B02
MCKNIGHT         ROGER            $16,100.00  B02
ROMANS           ANTHONY          $21,120.00  B04
SMITH            MARY             $13,200.00  B14
                 RICHARD           $9,500.00  A01
STEVENS          ALFRED           $11,000.00  A07
                      END OF REPORT
```

Once the value of DEFECHO has been set to NONE, it cannot be changed. The following SET command attempts to change the value to ON, but the query command output shows that it is still NONE:

```
SET DEFECHO=ON
? SET DEFECHO
 DEFECHO                 NONE
```

Syntax: How to Test Dialogue Manager Command Logic

```
{-DEFAULT|-SET|EX} &STACK = {ON|OFF}
```

where:

ON

Executes stacked commands normally. This value is the default.

OFF

Prevents the execution of stacked commands. In addition, system variables (for example, &RECORDS or &LINES) are not set. Dialogue Manager commands are executed so you can test the logic of the procedure.

Example: Using the &RETCODE Variable to Test the Result of a Command

The following command tests the existence of the C:\MYDATA\WEEK27.DAT file. If it exists, the value of &RETCODE is 0, and the file is erased.

```
DOS STATE C:\MYDATA\WEEK27.DAT
-IF &RETCODE NE 0 GOTO START;
DOS ERASE C:\MYDATA\WEEK27.DAT
-START
```

Viewing Messages for Debugging an Application

The Message Viewer enables you to see messages including error messages, informational messages, and Dialogue Manager commands such as -TYPE, -DEFAULT, and -SET &ECHO. These messages appear in a separate frame below the report output and serve as a good resource for debugging an application.

Syntax: How to Use the Message Viewer

Add the following WFServlet variable to your URL call for WebFOCUS:

```
IBIWF_msgviewer=option
```

where:

option

Is one of the following:

OFF

Disables the Message Viewer. When this option is disabled, you can still view informative messages or error messages for HTML reports from the view source option in your browser.

ON

Displays messages in a separate frame below the report output.

ECHOON

Displays messages and lines, in a separate frame below the report output, that are expanded and stacked for execution.

ECHOALL

Displays messages, lines, and all Dialogue Manager commands, in a separate frame below the report output, that are expanded and stacked for execution.

Note: The ON, ECHOON, and ECHOALL options are available for all output formats including HTML and PDF.

Example: Using the Message Viewer

The following illustrates how to use the Message Viewer with WebFOCUS self-service applications. This example shows the parameter with the ECHOALL option.

```
http://srv:port/ibi_apps/WFServlet?IBIF_ex=region&IBIWF_msgviewer=ECHOALL
```

The Message Viewer output is:

```
-INCLUDE region
-* Region.fex
-DEFAULT &VALUE='Northeast';
TABLE FILE GGSALES
SUM UNITS BY REGION
WHERE REGION EQ 'Northeast';
END
0 NUMBER OF RECORDS IN TABLE= 1080 LINES= 1
WebFOCUS Version 7.1.1 compiled and linked on Sat Aug 20 21:25:07 EDT 2005 (Gen WEB711:82)
```

Dialogue Manager Syntax Reference

This topic describes all the Dialogue Manager commands.

-* Command

-* signals the beginning of a comment line.

Any number of comment lines can follow one another, but each must begin with -*. A comment line may be placed at the beginning or end of a procedure, or in between commands. However, it cannot be on the same line as a command.

Use comment lines to document a procedure so that its purpose and history are clear to others.

The syntax is:

```
-* text
```

where:

text

Is a comment. A space is not required between -* and *text*.

-? Command

-? displays the current value of a local variable.

The syntax is:

```
-? &[string]
```

where:

string

Is a variable name. If this parameter is not specified, the current values of all local, global, and defined system and statistical variables are displayed, except for those whose values have been hidden using the -DEFAULTH command.

-CLOSE

-CLOSE closes an external file opened with the -READ or -WRITE command using the NOCLOSE option. The NOCLOSE option keeps a file open until the -READ or -WRITE operation is complete.

The syntax is:

```
-CLOSE filename
```

where:

filename

Is a symbolic name associated with a physical file known to the operating system.

-DEFAULT

-DEFAULT commands set default values for local or global variables. Supplying default values to variables in a stored procedure helps ensure that it runs correctly.

You can issue multiple -DEFAULT commands for a variable. If the variable is global, these -DEFAULT commands can be issued in separate FOCEXECs. At any point before another method is used to establish a value for the variable, the most recently issued -DEFAULT command will be in effect.

However, as soon as a value for the variable is established using any other method (for example, by issuing a -SET command, retrieving a value input by the user, or reading a value from a file), subsequent -DEFAULT commands issued for that variable are ignored.

The syntax is:

```
-DEFAULT &[&]name=value [...]
```

where:

name

Is a name of a variable.

value

Is the default value assigned to the variable.

-DEFAULTH

-DEFAULTH commands set default values for hidden variables.

You can initialize a variable and prevent it from being used for WebFOCUS parameter prompting by using the -DEFAULTH command. Variables initialized with -DEFAULTH, are not returned in the XML describe information used for parameter prompting. Since these variables are not displayed by the parameter prompting features and are not displayed by the -? command, they are hidden from the user.

The syntax is:

```
-DEFAULTH &[&]name=value [...]
```

where:

name

Is a name of a variable. This variable is not used for parameter prompting and is hidden from the user.

value

Is the default value assigned to the variable.

-DOS

-DOS executes a DOS operating system command from a procedure.

The syntax is:

```
-DOS command
```

where:

command

Is a DOS command.

-EXIT

-EXIT forces a procedure to end. All stacked commands are executed and the procedure exits. If the procedure was called by another one, the calling procedure continues processing.

Use -EXIT for terminating a procedure after processing a final branch that completes the desired task. The last line of a procedure is an implicit -EXIT.

The syntax is:

```
-EXIT
```

-GOTO

-GOTO transfers control to a specified label.

If Dialogue Manager finds the label, processing continues with the line following it. If Dialogue Manager does not find the label, processing ends and an error message is displayed.

The syntax is:

```
-GOTO label
      .
      .
-label [TYPE text]
```

where:

label
: Is a user-defined name of up to 64 characters that specifies the target of the -GOTO action.

  Do not use embedded blanks or the name of any other Dialogue Manager command except -QUIT or -EXIT. Do not use words that can be confused with functions, arithmetic and logical operations, and so on.

TYPE *text*
: Sends a message to the client application.

-HTMLFORM

-HTMLFORM sends report output to the HTML file you create for a webpage and displays that page in the browser. The output can also be saved as an HTML file that can be displayed later.

The syntax is:

```
-HTMLFORM filename [SAVE AS htmlpage]
```

or

```
-HTMLFORM BEGIN
.
.
.
-HTMLFORM END
```

where:

filename

Is the HTML file that contains placeholders for the report or reports WebFOCUS creates with the command:

```
ON TABLE HOLD HTMTABLE AS report
```

SAVE

Indicates that the HTML page, created when the HTML file *filename* and the report or reports generated by WebFOCUS are combined, is to be saved.

AS *htmlpage*

Is the file name for the HTML page that is created when the HTML file *filename* and the report or reports generated by WebFOCUS are combined. This file is saved so that it can be displayed later.

-HTMLFORM BEGIN

Indicates the beginning of an inline HTML form in a procedure.

-HTMLFORM END

Indicates the end of an inline HTML form in a procedure.

-IF

-IF routes execution of a procedure based on the evaluation of the specified expression.

An -IF command without an explicitly specified ELSE whose expression is false continues processing with the line immediately following it.

The syntax is:

```
-IF expression [THEN] GOTO label1; [ELSE GOTO label2;]
                                   [ELSE IF...;]
```

where:

label1...label2

Is a user-defined name of up to 64 characters that specifies the target of the GOTO action.

Do not use embedded blanks or the name of any other Dialogue Manager command except -QUIT or -EXIT. Do not use words that can be confused with functions, arithmetic or logical operations, and so on.

expression

Is a valid expression. Literals do not need to be enclosed in single quotation marks unless they contain embedded blanks or commas.

THEN

Is optional code that increases readability of the command.

ELSE GOTO

Passes control to *label2* when the -IF test fails.

ELSE IF

Specifies a compound -IF test.

The semicolon is required at the end of the command, and continuation lines must begin with a hyphen.

-INCLUDE

-INCLUDE enables one procedure to call another one. A procedure can call any number of other procedures. Up to four -INCLUDE commands can be nested.

The called procedure can contain fully executable or partial code. The calling procedure cannot branch to a label in the called procedure and vice versa.

Note: When calling the WebFOCUS CGI, the CGI uses an -INCLUDE command. Therefore, the end user can include only three nested -INCLUDE commands. When using edastart, this does not occur.

The syntax is:

```
-INCLUDE filename
```

where:

filename

Is the name of the called procedure.

For information on syntax for using fully qualified file names, see *Using Fully Qualified Names With the -Include Command* on page 403.

-label

-label is the target of a -GOTO command or -IF criteria.

The syntax is:

```
-label [TYPE message]
```

where:

label

Is a user-supplied name of up to 64 characters that identifies the target for a branch.

Do not use embedded blanks or the name of any other Dialogue Manager command except -QUIT or -EXIT. Do not use words that can be confused with functions, arithmetic or logical operations, and so on.

TYPE *message*

Sends a message to the client application.

-MVS

In WebFOCUS, executes an MVS operating system command from a procedure.

The syntax is:

```
-MVS command
```

where:

command

Is an MVS command.

-PASS

-PASS directly issues and controls passwords. This feature is especially useful for specifying a particular file or set of files that a given user can read or write. Passwords have detailed sets of functions associated with them through the DBA facility.

The procedure that sets passwords can be encrypted so that it and the passwords that it sets cannot be typed and made known.

A variable can be associated with -PASS so that you can prompt for and assign a password value.

The syntax is:

```
-PASS password
```

where:

password

Is a literal FOCUS password or a variable containing a password.

-QUIT

-QUIT forces an immediate exit from a procedure. Stacked commands are not executed.

If the procedure was called by another one, control returns directly to the client application, not to the calling procedure.

-QUIT can also be the target of a branch.

The syntax is:

```
-QUIT
```

-QUIT FOCUS

-QUIT FOCUS terminates the procedure and exits FOCUS. The user returns to the operating system, or the calling program.

Upon exiting FOCUS, the value is returned to the calling program, if one exists. This is useful when FOCUS is called from another Windows application to return a status code indicating the state at which FOCUS was exited.

-QUIT FOCUS can also be made the target of a branch, with the same results as those already described.

The syntax is:

```
-QUIT FOCUS [n]
```

where:

n

Is the application return code value. It can be a constant or an integer variable.

If you do not supply a value, or if you supply a non-integer value for *n*, the return code is 0 (the default value).

-READ

-READ enables the reading of data from an external file that is defined to the operating system.

The length of the variable list must be known before -READ is encountered. Use -DEFAULT to establish the number of characters expected for each variable.

If the list of variables is longer than one line, end the first line with a comma and begin the next line with a hyphen if you are reading a free-format file:

```
-READ EXTFILE, &CITY, &CODE1,
 - &CODE2
```

If you are reading a fixed-format file, begin the next line with a hyphen and comma as follows:

```
-READ EXTFILE &CITY.A8. &CODE1.A3.,
-, &CODE2.A3.
```

The syntax is:

```
-READ filename[,] [NOCLOSE] &name[.format.][,]...
```

where:

filename[,]

Is the name of an external file to read, which must be defined to the operating system. A space after *filename* denotes a fixed-format file, while a comma after *filename* denotes a free-format file.

NOCLOSE

Keeps the external file open until the -READ operation is complete. Files kept open with NOCLOSE can be closed using the command -CLOSE *filename*.

&name[,]...

Is a list of variables. For free-format files, you may but are not required to separate the variable names with commas.

.format.

Is the format of the variable. It may be Alphanumeric (A) or Numeric (I). Note that format must be delimited by periods. The format is ignored for comma-delimited files.

-READFILE

-READFILE reads Master File fields into Dialogue Manager variables.

The syntax is:

```
-READFILE [app/]mastername
```

where:

app
Is the application directory in which the file resides.

mastername
Is the name of the Master File to be read.

-REPEAT

-REPEAT allows looping in a stored procedure. A loop ends when any of the following occurs:

- It is executed in its entirety.
- A -QUIT or -EXIT command is issued.
- A -GOTO is issued to a label outside of the loop. If a -GOTO is later issued to return to the loop, the loop proceeds from the point it left off.

The syntax is:

```
-REPEAT label n TIMES
```

or

```
-REPEAT label WHILE condition;
```

or

```
-REPEAT label FOR &variable [FROM fromval] [TO toval] [STEP s]
```

where:

label
Identifies the code to be repeated (the loop). A label can include another loop if the label for the second loop has a different name than the first.

n TIMES
Specifies the number of times to execute the loop. The value of n can be a local variable, a global variable, or a constant. If it is a variable, it is evaluated only once, so you cannot change the number of times to execute the loop. The loop can only be ended early using -QUIT or -EXIT.

WHILE *condition*;
Specifies the condition under which to execute the loop. The condition is any logical expression that can be true or false. The loop is run if the condition is true.

`FOR &variable`

Is a variable that is tested at the start of each execution of the loop. It is compared with the value of *fromval* and *toval* (if supplied). The loop is executed only if *&variable* is less than or equal to *toval* when STEP is positive, or greater than or equal to *toval* when STEP is negative. If *toval* is not used, another option must be substituted, such as *n* TIMES.

`FROM fromval`

Is a constant that is compared with *&variable* at the start of each execution of the loop. 1 is the default value.

`TO toval`

Is a value that is compared with *&variable* at the start of each execution of the loop. The default value is 1,000,000.

`STEP s`

Is a constant used to increment *&variable* at the end of each execution of the loop. It may be positive or negative. 1 is the default value.

Note: The parameters FROM, TO, and STEP can appear in any order.

-RUN

-RUN causes immediate execution of all stacked commands.

Following execution, processing of the stored procedure continues with the line that follows -RUN.

-RUN is commonly used to do the following:

- Generate results from a request that can then be used in testing and branching.
- Close an external file opened with -READ or -WRITE. When a file is closed, the line pointer is placed at the beginning of the file for a -READ. The line pointer for -WRITE is positioned depending on the allocation and definition of the file.

The syntax is:

```
-RUN
```

-SET

-SET assigns a literal value, or a value that is computed in an arithmetic or logical expression, to a variable.

Single quotation marks around a literal value are optional unless it contains an embedded blank, comma, or equal sign, in which case you must include them.

The syntax is:

```
-SET &[&]name= {expression|value};
```

where:

&name

Is the name of a variable whose value will be set.

expression

Is a valid expression. Expressions can occupy several lines, so end the command with a semicolon (;).

value

Is a literal value assigned to the variable. If the literal value contains commas or embedded blanks, you must enclose the value in single quotation marks. If the value contains a single quote, place two single quotes where you want one to appear.

-TSO

In WebFOCUS, -TSO executes a TSO operating system command from a procedure. This is only supported when the command is RUN.

The syntax is:

```
-TSO command
```

where:

command

Is a TSO RUN command.

-TYPE

-TYPE sends a message to a client application.

Any number of -TYPE commands can follow one another, but each must begin with -TYPE.

Variables may be embedded in the message. The values currently assigned to each variable will be displayed.

The syntax is:

```
-TYPE text
```

where:

text

Is a message that will be sent to a client application, followed by a line feed. If you include quotation marks around *text*, they will be displayed as part of the message.

The length of *text* can be up to 256 bytes.

-UNIX

In WebFOCUS, -UNIX executes a UNIX operating system command from a procedure.

The syntax is:

```
-UNIX command
```

where:

command

Is a UNIX command.

-VMS

In WebFOCUS, -VMS executes a VMS operating system command from a procedure.

The syntax is:

```
-VMS command
```

where:

command

Is a VMS command.

-WINNT

-WINNT executes a Windows operating system command from a procedure.

The syntax is:

```
-WINNT command
```

where:

command

Is a Windows command.

-WRITE

-WRITE writes data to an external file.

If the command continues over several lines, put a comma at the end of the line and a hyphen at the beginning of each subsequent line.

Unless you specify the NOCLOSE option, an opened file is closed upon termination of the procedure with -RUN, -EXIT, or -QUIT.

The syntax is:

```
-WRITE filename [NOCLOSE] text
```

where:

filename

Is the name of a physical external file being written to. The *filename* must be known to the operating system.

NOCLOSE

Keeps the external file open until the -WRITE operation is complete. Files kept open with NOCLOSE can be closed with the command -CLOSE *filename*.

text

Is any combination of variables and text.

Chapter 6

Testing and Debugging a Procedure

You can debug a WebFOCUS application by querying your environment to display information, such as release, server information, and joins, as well as by identifying files you are using. Other debugging methods are available, including specialized Dialogue Manager variables and traces performed from the Web Console.

In this chapter:

- Debugging Your Application With Query Commands
- Displaying Combined Structures
- Displaying Virtual Fields
- Displaying the Currency Data Source in Effect
- Displaying Available Fields
- Displaying the File Directory Table
- Displaying Field Information for a Master File
- Displaying Data Source Statistics
- Displaying Current ddnames Assigned With FILEDEF
- Displaying Defined Functions
- Displaying HOLD Fields
- Displaying JOIN Structures
- Displaying National Language Support
- Displaying Explanations of Error Messages
- Displaying the Current Search Path
- Displaying the Release Number
- Displaying the Values of a Remote Server
- Displaying Parameter Settings
- Displaying Graph Parameters
- Displaying the Site Code of the Connected Server
- Displaying Command Statistics
- Displaying StyleSheet Parameter Settings
- Displaying Information About the SU Machine
- Displaying Data Sources Specified With USE
- Displaying Global Variable Values
- Identifying the Files Being Used
- Reporting Dynamically From System Tables

Debugging Your Application With Query Commands

You can use query commands to display information about your metadata, data sources, language environment, and WebFOCUS. This is useful for determining any aspects of your application that may be preventing the proper behavior of your application. You can query your environment with a query command executed in WebFOCUS.

The result of a query command is returned in your browser window or as a comment in the HTML file. Use the applicable web browser functions to view the HTML comments (for example, View Source in Microsoft Internet Explorer).

Syntax: How to Issue a Query Command

```
? query [filename]
```

where:

query

Is the subject of the query.

filename

Is the name of the file that is the subject of the query. This parameter applies to only some queries.

To list the query commands, type a question mark in a procedure.

Reference: Query Command Summary

The following is a list of query commands. This topic contains a detailed description of each.

Query Command	Description
`? COMBINE`	Displays a list of combined file structures.
`? DEFINE`	Displays currently active virtual fields created by the DEFINE command or attribute.
`? SET EUROFILE`	Lists fields currently available to you.
`?F`	Lists fields currently available.
`? FDT`	Displays physical attributes of a FOCUS data source.

Query Command	Description
?FF	Lists field names, aliases, and format information for an active Master File.
? FILE	Displays the number of segment instances in a FOCUS data source and the last time the data sources was changed.
? FILEDEF	Displays the current logical names assigned by the FILEDEF command.
? FUNCTION	Displays functions created with the DEFINE command.
? HOLD	Displays fields described in a HOLD Master File.
? JOIN	Displays JOIN structures that exist between data sources.
? LANG	Displays information about National Language Support.
? MDI	Generates statistics and descriptions for multi-dimensional indexes.
? *n*	Displays an explanation of an error message (*n* represents the number of the error message).
? PATH	Displays the current search path.
? RELEASE	Displays the release number of your product.
? REMOTE	Displays the values of the remote servers.
? SET	Displays parameter settings that control WebFOCUS.
? SET GRAPH	Displays parameter settings that control graphs produced with the GRAPH command.
? SET NOT	Produces a list of SET commands that cannot be set in a specific area.
? SITECODE	Retrieves the site code of the connected server.
? STAT	Displays statistics about the last command executed.

Query Command	Description
? STYLE	Displays the current settings for StyleSheet parameters.
? SU	Is communication available to the SU machine.
? USE	Displays data sources specified with the USE command.
? &&	Displays values of global variables.

Displaying Combined Structures

The ? COMBINE command displays files that are in the current combined structures.

Syntax: How to Display Combined Structures

```
? COMBINE
```

Example: Displaying Combined Structures

Issuing the command

```
? COMBINE
```

produces information similar to the following:

```
FILE=EMPLJOB COMBINED FROM EMPLOYEE JOBFILE
```

Reference: ? COMBINE Query Information

The list of file structures has the following form

```
FILE=combine_name COMBINED FROM
file_1
file_2
   .
   .
   .
file_n
```

where:

combine_name

Is the name of the COMBINE structure.

file_1...file_n

Are the names of FOCUS data sources that comprise the combined structure.

Displaying Virtual Fields

The? DEFINE command lists the active virtual fields used in a request. The fields can be created by either the DEFINE command or DEFINE attribute in the Master File. The command displays field names of up to 32 characters. If a name exceeds 32 characters, a symbol in the 32nd position indicates a longer field name. In WebFOCUS, this symbol is an ampersand (&).

Syntax: How to Display Virtual Fields

```
? DEFINE [filename]
```

where:

filename

Is the data source containing the virtual fields. If *filename* is omitted, the command displays all virtual fields.

Example: Displaying Virtual Fields

Assume that you created a virtual field named NEW_DATE in a request against the EMPLOYEE database.

Issuing

```
? DEFINE
```

produces the following information:

```
FILE FIELD NAME FORMAT SEGMENT VIEW TYPE EMPLOYEE NEW_DATE YYMD 2
```

Reference: ? DEFINE Query Information

The following information is listed for each virtual field created with DEFINE:

Option	Description
FILE	Is the name of the data source containing the virtual field.
FIELD NAME	Is the name of the virtual field.

Option	Description
FORMAT	Is the format of the virtual field. The notation is the same as that used for the FORMAT attribute in a Master File.
SEGMENT	Is the number of the segment in the Master File containing the virtual field. During reporting, your application treats the virtual field as a field in this segment. To relate segment numbers to segment names, use ? FDT.
VIEW	Is the root segment of DEFINE that specifies an alternate view. For example: `DEFINE FILE EMPLOYEE.JOBCODE`
TYPE	Indicates whether the virtual field is created by the DEFINE attribute in the Master File, or by a DEFINE command, identified by MASTER or a blank, respectively.

Displaying the Currency Data Source in Effect

The ? SET EUROFILE command displays the currency data source in effect.

Syntax: How to Display the Currency Data Source in Effect

To display the currency data source in effect, issue the command:

```
? SET EUROFILE
```

Example: Displaying the Currency Data Source in Effect

Issuing the command

```
? SET EUROFILE
```

produces information similar to the following:

```
EUROFILE CURRCODE
EUROFILE GBP
```

Displaying Available Fields

The ?F command displays the fields that are currently available.

?F displays entire 66 character field names.

Syntax: How to Display Available Fields

```
?F filename
```

where:

filename

Is the name of a data source.

Example: Displaying Available Fields

Issuing the command

```
?F EMPLOYEE
```

produces the following information:

```
FILENAME = EMPLOYEE    EMP_INFO.EMP_ID LAST_NAME FIRST_NAME HIRE_DATE
DEPARTMENT CURR_SAL CURR_JOBCODE ED_HRS   BANK_NAME BANK_CODE BANK_ACCT
EFFECT_DATE   DAT_INC PCT_INC SALARY PAYINFO.JOBCODE   TYPE ADDRESS_LN1
ADDRESS_LN2 ADDRESS_LN3 ACCTNUMBER   PAY_DATE GROSS   DED_CODE DED_AMT
JOBSEG.JOBCODE JOB_DESC   SEC_CLEAR   SKILLS SKILLS_DESC   DATE_ATTEND
ATTENDSEG.EMP_ID   COURSE_CODE COURSE_NAME
```

Displaying the File Directory Table

The ? FDT command displays the file directory table, which lists the physical characteristics of a FOCUS data source.

Each segment and index (those fields designated by the keyword FIELDTYPE=I in the Master File) occupies an integral number of pages. The file directory table shows the amount of space occupied by each segment instance in a page, the starting and ending page numbers, and the number of pages in between for each segment and index.

Syntax: How to Display a File Directory Table

```
? FDT filename
```

where:

filename

Is the name of the data source.

Example: Displaying a File Directory Table

Issuing the command

```
? FDT EMPLOYEE
```

produces the following information:

```
   DIRECTORY:C:\ibi\WEBFOCUS\...\employee.foc ON 09/25/2006 AT 09.50.28
DATE/TIME OF LAST CHANGE: 03/30/2005 16.19.22 SEGNAME LENGTH PARENT
START END PAGES LINKS TYPE 1 EMPINFO 22 1 1 1 6 2 FUNDTRAN 10 1 2 2 1 2
3 PAYINFO 8 1 3 3 1 3 4 JOBSEG 11 3 4 5 SECSEG 4 4 2 6 SKILLSEG 11 4 2 7
ADDRESS 19 1 4 4 1 2 8 SALINFO 6 1 5 5 1 3 9 DEDUCT 5 8 6 8 3 2 10
ATTNDSEG 7 1 3 11 COURSEG 11 10 2
```

Reference: ? FDT Query Information

The following information is listed in the file directory table:

SEGNAME	Is the name of each segment in the file. The segments are also numbered consecutively down the left of the table. Unnumbered entries at the foot of the table are indexes, which belong to fields having the attribute FIELDTYPE=I in the Master File.
LENGTH	Is the length in words (units of four bytes) of each segment instance. Divide this number into 992 to get the number of instances that fit on a page.
PARENT	Is the parent segment. Each number refers to a segment name in the SEGNAME column.
START	Is the page number on which the segment or index begins.
END	Is the page number on which the segment or index ends.
PAGES	Is the number of pages occupied by the segment or index.
LINKS	Is the length, in words, of the pointer portion in each segment instance. Every segment instance consists of two parts, data and pointers. Pointers are internal numbers used to find other instances.
TYPE	Is the type of index. NEW indicates a binary index. OLD indicates a hash index. Segments of type KU, LM, DKU, DKM, KL, and KLU are not physically in this file. Therefore, this information is omitted from the table.

Displaying Field Information for a Master File

The ?FF command displays field names, aliases, and format information for an active Master File.

Syntax: How to Display Field Information for a Master File

```
?FF filename [string]
```

where:

filename

Is the name of the Master File.

string

Is a character string up to 66 characters long. The command displays information only for fields beginning with the specified character string. If you omit this parameter, the command displays information for all fields in the Master File.

Example: Displaying Field Information for a Master File

Issuing the command

```
?FF EMPLOYEE
```

produces the following information:

```
FILENAME= EMPLOYEE EMPINFO.EMP_INFO EMPINFO.EID A9 LAST_NAME LN A15
FIRST_NAME FN A10 HIRE_DATE HDT 16YMD DEPARTMENT DPT A10 CURR_SAL CSAL
D12.2M CURR_JOBCODE CJC A3 ED_HRS OJT F6.2 BANK_NAME BN A20 BANK_CODE BC
I6S BANK_ACCT BA I9S EFFECT_DATE EDATE 16YMD
```

Displaying Data Source Statistics

The ? FILE command displays information, such as the number of segment instances in a FOCUS data source and when the data source was last changed.

Syntax: How to Display Data Source Statistics

```
? FILE filename
```

where:

filename

Is the name of the data source.

Example: Displaying Data Source Statistics

Issuing the command

```
? FILE EMPLOYEE
```

produces statistics similar to the following:

```
    STATUS OF FOCUS FILE: D:\ibi\apps\...\employee.foc ON 07/14/2000 AT
14.59.14 ACTIVE DELETED DATE OF TIME OF LAST TRANS SEGNAME COUNT COUNT
LAST CHG LAST CHG NUMBER EMPINFO 12 06/14/2000 11.04.01 FUNDTRAN 6
03/31/2000 11.16.15 6 PAYINFO 19 06/05/2000 16.07.49 ADDRESS 21
03/31/2000 11.16.15 21 SALINFO 70 03/31/2000 11.16.16 448 DEDUCT 448
03/31/2000 11.16.16 448 TOTAL SEGS 576 TOTAL CHARS 8984 TOTAL PAGES 8
LAST CHANGE 06/14/2000 11.01.01
```

Reference: ? FILE Query Information

The following data source statistics are listed:

SEGNAME	Is the name of each segment in the data source. After the segments, the indexes are listed, if applicable. Indexes are those fields specified by the attribute FIELDTYPE=I in the Master File.
ACTIVE COUNT	Is the number of instances of each segment.
DELETED COUNT	Is the number of segment instances deleted, for which the space is not reused.
DATE OF LAST CHG	Is the date on which data in a segment instance or index was last changed.
TIME OF LAST CHG	Is the time of day, on a 24-hour clock, when the last update of a file was made for that segment or index.
LAST TRANS NUMBER	Is the number of transactions performed by the last update request to access the segment. If the data source was changed under Simultaneous Usage mode, this column refers to the REF NUMB column of the CR HLIPRINT file.
TOTAL SEGS	Is the total number of segment instances in the file (shown under ACTIVE COUNT), and the number of segments deleted when the file was last changed (shown under DELETED COUNT).

TOTAL CHARS	Is the number of characters of data in the file.
TOTAL PAGES	Is the number of pages in the data source. Pages are physical records in FOCUS data sources.
LAST CHANGE	Is the date and time the data source was last changed.

If a data source is disorganized by more than 29%, that is, the physical placement of data in the data source is considerably different from its logical or apparent placement, the following message appears

```
FILE APPEARS TO NEED THE -REBUILD- UTILITY
REORG PERCENT IS A MEASURE OF FILE DISORGANIZATION
0 PCT IS PERFECT -- 100 PCT IS BAD
REORG PERCENT IS x%
```

where:

x

Is a percentage between 30 and 100.

The variable &FOCDISORG also indicates the level of disorganization. Following is an example of how to use &FOCDISORG in a Dialogue Manager -TYPE command:

```
-TYPE THE AMOUNT OF DISORGANIZATION OF THIS FILE IS: &FOCDISORG
```

This command, depending on the amount of disorganization, produces a message similar to the following:

```
THE AMOUNT OF DISORGANIZATION OF THIS FILE IS: 10
```

When using a -TYPE command with &FOCDISORG, a message is displayed even if the percentage of disorganization is less than 30%.

Displaying Current ddnames Assigned With FILEDEF

The ? FILEDEF command displays the logical names (ddnames) assigned for various files, input and output.

During a session, depending on the types of operations you are doing, WebFOCUS can create many FILEDEFs. Four of the system-created FILEDEFs are apparent using the ? FILEDEF command. You can have up to 250 FILEDEFs including both user-created and system-created FILEDEFs. For related information see *Defining and Allocating WebFOCUS Files* on page 651.

Syntax: How to Display Current ddnames

To display current ddnames, issue the command:

```
? FILEDEF
```

Example: Displaying Current ddnames

Issuing the command

```
? FILEDEF
```

produces information similar to the following:

```
Lname Device Lrecl Recfm Append Expl Filename
============================================================ HOLD2 DISK
0 V N Y C:\VM\SMALL\HOLD2.FTM
```

Displaying Defined Functions

The ? FUNCTION command displays all defined functions and the parameters.

Syntax: How to Display DEFINE Functions

To display defined functions, issue the command:

```
? FUNCTION
```

Example: Displaying DEFINE Functions

Issuing the command

```
? FUNCTION
```

produces information similar to the following:

```
FUNCTIONS CURRENTLY ACTIVE

Name          Format     Parameter      Format
SUBTRACT      D8.2       VAL1           D8
                         VAL2           D8
```

Displaying HOLD Fields

The ? HOLD command lists fields described in a Master File created by the ON TABLE HOLD command. The list displays the field names, the aliases, and the formats as defined by the FORMAT (USAGE) attribute. The ? HOLD command displays field names up to 32 characters. If a field name exceeds 32 characters, a symbol in the 32nd position indicates a longer field name. In WebFOCUS, this symbol is an ampersand (&).

The ? HOLD command displays fields of a HOLD Master File created by the current request.

Syntax: How to Display HOLD Fields

```
? HOLD [filename]
```

where:

filename

Is the name assigned in the AS phrase in the ON TABLE HOLD command. If you omit the file name, it defaults to HOLD.

Example: Displaying HOLD Fields

Issuing the command

```
? HOLD
```

produces information similar to the following:

```
DEFINITION OF HOLD FILE FIELDNAME ALIAS FORMAT LAST_NAME E01 A15
FIRST_NAME E02 A10
```

Displaying JOIN Structures

The ? JOIN command lists the JOIN structures currently in effect. The command displays field names up to 12 characters. If a field name exceeds 12 characters, a symbol in the twelfth position indicates a longer field name. In WebFOCUS, this symbol is an ampersand (&).

Syntax: How to Display JOIN Structures

To display JOIN structures, issue the command:

```
? JOIN
```

Example: Displaying JOIN Structures

Issuing the command

```
? JOIN
```

produces information similar to the following:

```
JOINS CURRENTLY ACTIVE
HOST                    CROSSREFERENCE
FIELD     FILE      TAG   FIELD           FILE     TAG   AS  ALL   WH
-----     ----      ---   -----           ----     ---   --  ---   --
JOBCODE   EMPLOYEE        JOBCODE         JOBFILE            N     N
```

Reference: ? JOIN Query Information

The following JOIN information is listed:

HOST FIELD	Is the name of the host field that is joining the data sources.
FILE	Is the name of the host data source.
TAG	Is a tag name used as a unique qualifier for field names in the host data source.
CROSSREFERENCE FIELD	Is the name of the cross-referenced field used to join the data sources.
FILE	Is the name of the cross-referenced data source.
TAG	Is a tag name used as a unique qualifier for field names in the cross-referenced data source.
AS	Is the name of the joined structure.
ALL	Displays Y for a non-unique join and N for a unique join.
WH	Specifies whether the join is a conditional join or an equi-join.

Displaying National Language Support

The ? LANG command displays information about National Language Support.

Syntax: How to Display Information About National Language Support

To display information about National Language Support:

```
? LANG
```

Example: Displaying Information About National Language Support

Issuing the command

```
? LANG
```

produces information similar to the following:

```
        NATIONAL LANGUAGE INFORMATION Language 001/AMENGLISH ( ) Code
Page 437 client Code Page 437 Dollar 24($) Lowcase alphabet YES Decimal
notation OFF(.) Currency symbol $ Date/Time format EDA NLS sort NO NLS
upcase/lowcase NO DBCS Flag OFF(SBCS)
```

Displaying Explanations of Error Messages

The ? *n* command displays a detailed explanation of an error message, providing assistance in correcting the error.

Error messages generated by certain data adapters, such as the DB2 and MODEL 204 data adapters, are also accessible through this feature.

Syntax: How to Display Explanations of Error Messages

```
? n
```

where:

n

Is the error message number.

Example: Displaying Explanations of Error Messages

If you receive the message

```
(FOC125) RECAP CALCULATIONS MISSING
```

issuing the command

```
? 125
```

produces the following message:

```
(FOC125) RECAP CALCULATIONS MISSING The word RECAP is not
followed by a calculation. Either the RECAP should be removed,
or a calculation provided.
```

Displaying the Current Search Path

The ? PATH query displays your current search path. The search for files extends over several directories. The order that these directories are searched is determined for each user with the SET DEFAULT command and the values for the FOC$DIR logicals.

Syntax: How to Display the Current Search Path

```
? PATHquery commands:? PATH;? PATH command;
```

Example: Displaying the Current Search Path

Issuing the command

```
? PATH
```

produces information similar to the following if the WebFOCUS Reporting Server is using EDAPATH:

```
EDAPATH = D:\ibi\apps\ggdemo\ D:\ibi\apps\ncp\ D:\ibi\apps\template\ C:
\VM\SMALL\ TEMPDIR = D:\ibi\srv82\wfs\edatemp\ts000019\ CURRDIR = D:\ibi
\srv82\wfs\edatemp\ts000019\ EDASYN = D:\ibi\srv82\wfs\catalog\ EDASYNR
= D:\ibi\srv82\wfs\catalog\
```

If the WebFOCUS Reporting Server is using APP PATH, ? PATH produces information similar to the following:

```
BASEAPP = C:\ibi\apps\baseapp; APPHOLD = C:\ibi\srv82\wfs\edatemp
\ts000001; APPPATH = C:\ibi\apps\session; C:\ibi\apps\session
```

Displaying the Release Number

The ? RELEASE command displays the number of the currently installed release of your product.

Syntax: How to Display the Release Number

To display the release number, issue the command:

```
? RELEASE
```

Example: Displaying the Release Number

Issuing the command

```
? RELEASE
```

produces information similar to the following:

```
EDA999 Release * Current Software EDA999 Release RELEASE = R729999B
EDA999 Release GEN_NUM = 000.223968 EDA999 Release SOURCE_DATE =
10/16/2002 18:13:18 EDA999 Release BUILD_DATE = 10/17/2002 02:59:24
EDA999 Release INSTALLATION_DATE = 10-17-2002 16:53:16
```

Displaying the Values of a Remote Server

The ? REMOTE command displays information about remote servers. By issuing the ? REMOTE command while connected to a WebFOCUS server, you can obtain information about your remote destination, service, user ID, EDA node, servlet, report server node, and the web server user ID and password.

Syntax: How to Display the Values of a Remote Server

```
? REMOTE
```

Example: Displaying the Values of a Remote Server

Issuing the command

```
? REMOTE
```

produces information similar to the following:

```
 Remote Destination --> LOOPBACK Remote User ID is not set. Remote User
Password is not set. There is no conversations active.
```

Displaying Parameter Settings

The ? SET command lists the parameter settings that control your WebFOCUS environment. Your application sets default values for these parameters, but you can change them with the SET command.

SET parameters are described in *Customizing Your Environment* on page 501.

Syntax: How to Display Parameter Settings

```
? SET [ALL|[FOR] parameter]
```

where:

`ALL`

Displays all possible parameter settings.

`parameter`

Is a SET parameter. This displays the setting for the specific parameter.

`FOR`

Includes where the parameter can be set from in addition to the parameter setting.

Example: Displaying Parameter Settings

Issuing the command

```
? SET
```

produces information similar to the following:

```
                          PARAMETER SETTINGS
ALL.                  OFF    FOC2GIGDB              ON    PAUSE                 OFF
ASNAMES             FOCUS    FOCALLOC              OFF    POOL                  OFF
AUTOINDEX              ON    FOCCREATELOC          OFF    PRINT              ONLINE
AUTOPATH               ON    FOCSTACK SIZE           8    PRINTPLUS             OFF
BINS                   64    FOCTRANSFORM           ON    QUALCHAR                .
BLKCALC               NEW    FORMFEED            ASCII    QUALTITLES            OFF
BUSDAYS          _MTWTF_     HDAY                         REBUILDMSG           1000
BYPANELING            OFF    HIPERFOCUS            OFF    RECAP-COUNT           OFF
CACHE                   0    HOLDATTRS           FOCUS    SAVEMATRIX            OFF
CARTESIAN             OFF    HOLDLIST              ALL    SCREEN                OFF
CDN                   OFF    HOLDSTAT              OFF    SHADOW PAGE           OFF
COLUMNSCROLL          OFF    HOTMENU               OFF    SMARTMODE             OFF
DATEDISPLAY           OFF    IBMLE                  ON    SPACES               AUTO
DATEFNS                ON    INDEX TYPE            NEW    SQLENGINE
DATETIME  STARTUP/RESET    LANGUAGE          AMENGLISH    SUMPREFIX             LST
DEFCENT                19    LINES/PAGE            66    TCPIPINT              OFF
DEFECHO               OFF    LINES/PRINT           57    TEMP DISK D:\ibi\srv8..
DUPLICATECOL           ON    MERGEOPT               Y    TERMINAL         IBM3270
EMPTYREPORT           OFF    MESSAGE                ON    TESTDATE           TODAY
EQTEST           WILDCARD    MODE                WINNT    TITLES                 ON
EXCELRELEASE         2000    MORE                  OFF    VIEWNAMESIZE           60
EXL2KLANG       AMENGLISH    MULTIPATH        COMPOUND    WIDTH                 130
EXTAGGR                ON    NFOC                  OFF    WINPFKEY              OLD
EXTHOLD                ON    NODATA                  .    XFBINS     64 (passive)
EXTRACT               OFF    ONFIELD               ALL    XFC                    ON
EXTSORT               OFF    PAGE-NUM               ON    XRETRIEVAL             ON
FIELDNAME             NEW    PANEL                   0    YRTHRESH                0
```

Some parameters are listed differently from the way you specify them in the SET command. These include:

SET Parameters	Description
FOCSTACK SIZE	Is the same as the FOCSTACK parameter.
INDEX TYPE	Is the same as the INDEX parameter.
LINES/PAGE	Is the same as the PAPER parameter.
LINES/PRINT	Is the same as the LINES parameter.

SET Parameters	Description
SHADOW PAGES	Is the same as the SHADOW parameter.

Example: Displaying a Single Parameter Setting

Issuing the command

```
? SET ONLINE-FMT
```

produces the following if the parameter is set to its default value:

```
ONLINE-FMT HTML
```

Example: Displaying Where a Parameter Can Be Set

Issuing the command

```
? SET FOR EXTSORT
```

produces the following information:

```
EXTSORT ON

  ------------------------------------------------
  SETTABLE FROM COMMAND LINE                  : YES
  SETTABLE ON TABLE                           : YES
  SETTABLE FROM SYSTEM-WIDE PROFILE           : YES
  SETTABLE FROM HLI PROFILE                   : YES
```

Displaying Graph Parameters

The ? SET GRAPH command lists the parameter settings that control graphs produced with the GRAPH command. These parameters are described further in *Customizing Your Environment* on page 501.

Syntax: How to Display Graph Parameters

To display graph parameters, issue the command:

```
? SET GRAPH
```

Example: Displaying Graph Parameters

Issuing the command

```
? SET GRAPH
```

produces information similar to the following:

```
                        GRAPH PARAMETER SETTINGS

3DGRAPH              ON    GTREND              OFF    PIE                 OFF
AUTOTICK             ON    GVIEWERMAX          OFF    TERMINAL        IBM3270
BARNUMB             OFF    GVIEWERPAUSE         ON    VAUTO                ON
BARSPACE              0    HAUTO                ON    VAXIS                66
BARWIDTH              1    HAXIS               130    VCLASS              .00
BSTACK              OFF    HCLASS              .00    VGRID               OFF
FORMATGRAPH        HTML    HISTOGRAM            ON    VMAX                .00
GMISSING            OFF    HMAX                .00    VMIN                .00
GMISSVAL            .00    HMIN                .00    VTICK               .00
GPROMPT             OFF    HSTACK              OFF    VZERO               OFF
GRIBBON             OFF    HTICK               .00
GRID                OFF    LOOKGRAPH          NONE
```

Displaying the Site Code of the Connected Server

The edaserve.cfg file has an optional keyword called site_code that identifies the site code associated with the server. You can populate this entry during installation of the server or manually after installation using the License option on Configuration pane of the Web Console.

Once the site code has been installed, you can retrieve its value by issuing the ? SITECODE query command. If you issue this query command in a request, you will retrieve the site code of the server you are connected to for that request. If the site code has not been installed, you will get a message indicating that the site code is not available.

Syntax: How to Retrieve the Site Code of the Connected Server

```
? SITECODE
```

Example: Querying the Site Code of the Connected Server

Assume the edaserve.cfg file has the following entry:

```
site_code = A527O9b
```

Issue the following query command:

```
? SITECODE
```

The output is:

```
SITE CODE A527O9b
```

If the site code is not installed, the ? SITECODE query returns the following message:

```
SITE CODE NOT AVAILABLE
```

Displaying Command Statistics

The ? STAT command lists statistics for the most recently executed command.

Each statistic applies only to a certain command. If another command is executed, the statistic is either 0 or does not appear in the list at all. When you execute commands in stored procedures, these statistics are automatically stored in Dialogue Manager statistical variables. See your Dialogue Manager documentation for details.

Syntax: How to Display Command Statistics

To display command statistics, issue the command:

```
? STAT
```

Example: Displaying Command Statistics

Issuing the command

```
? STAT
```

produces information similar to the following:

```
                    STATISTICS OF LAST COMMAND
RECORDS         =      0           SEGS DELTD      =      0
LINES           =      0           NOMATCH         =      0
BASEIO          =      0           DUPLICATES      =      0
SORTIO          =     47           FORMAT ERRORS   =      0
SORT PAGES      =      0           INVALID CONDTS  =      0
READS           =      0           OTHER REJECTS   =      0
TRANSACTIONS    =      0           CACHE READS     =      0
ACCEPTED        =      0           MERGES          =      0
SEGS INPUT      =      0           SORT STRINGS    =      0
SEGS CHNGD      =      0           INDEXIO         =      0

INTERNAL MATRIX CREATED:   YES     AUTOINDEX USED: NO
SORT USED:               FOCUS     AUTOPATH USED:  NO
AGGREGATION BY EXT.SORT:    NO     HOLD FROM EXTERNAL SORT: NO
```

Reference: ? STAT Query Information

The following information displays:

RECORDS
: Is for TABLE, TABLEF, and MATCH commands. It indicates the number of data source records used in the report. The meaning of a record depends on the type of data source used.

LINES
Is for TABLE and TABLEF commands. It indicates the number of lines displayed in a report.

BASEIO
Is for TABLE and TABLEF commands. It indicates the number of I/O operations performed on the data source.

SORTIO
Is for TABLE, TABLEF, and MATCH commands. It indicates the number of I/O operations performed on the FOCSORT file, which is a work file invisible to the end user.

SORTPAGES
Is for TABLE and TABLEF commands. It indicates the number of physical records in the FOCSORT file.

CACHE READS
Is the number of cache reads performed. For details, see CACHE in *Customizing Your Environment* on page 501.

MERGES
Is the number of times that merge routines were invoked.

SORT STRINGS
Is the number of times that the sort capacity was exceeded.

INTERNAL MATRIX CREATED
Indicates how report sorting was handled. If an external sort handled it entirely, the value is NO. If both the application and an external sort handled it, the value is Y.

SORT USED
Is the type of sort facility used. It can have a value of FOCUS, EXTERNAL, SQL, or NONE. NONE means that the report did not require sorting.

AGGREGATION BY EXT. SORT
Uses external sorts to perform aggregation.

AUTOINDEX USED
Automatically takes advantage of indexed fields to speed data retrieval.

AUTOPATH USED
Selects an optimal retrieval path for accessing a data source.

HOLD FROM EXTERNAL SORT
Creates hold files with an external sort.

Displaying StyleSheet Parameter Settings

The ? STYLE command displays the current settings for StyleSheet parameters.

Syntax: **How to Display StyleSheet Parameter Settings**

```
? [SET] STYLE
```

Example: **Displaying StyleSheet Parameter Settings**

Issuing the command

```
? STYLE
```

produces information similar to the following:

```
                  STYLESHEET PARAMETER SETTINGS
ONLINE-FMT       HTML
OFFLINE-FMT      STANDARD
STYLESHEET       ON
SQUEEZE          OFF
LABELPROMPT      OFF
STYLEMODE        FULL
ORIENTATION      PORTRAIT
UNITS            INCHES
TOPMARGIN        .250
BOTTOMMARGIN     .250
LEFTMARGIN       .250
RIGHTMARGIN      .250
TARGETFRAME      ????
FOCEXURL         \cgi-bin\ibi_cgi\ibiweb.exe?IBIMR_drill=X,corp/corp.htm
BASEURL          OFF
```

Reference: **? STYLE Query Information**

The following StyleSheet information is listed:

`ONLINE-FMT`	Is the format of report output. Can produce output as HTML (the default), PDF, Excel 2000, Excel 97, PostScript, or as unstyled character-based output.
`OFFLINE-FMT`	Is the format of report output. Can produce output as HTML or as unstyled character-based output.
`STYLESHEET`	Rejects or accepts StyleSheet parameters that specify formatting options, such as page size, orientation, and margins.
`LABELPROMPT`	Specifies on which label of the first page to begin printing a multi-pane report, such as a mailing label report.

STYLEMODE	Speeds the retrieval of large report output by displaying output in multiple HTML tables where each table is a separate report page.
ORIENTATION	Is the page orientation for styled reports. Can be either portrait or landscape.
UNITS	Is the unit of measure for PostScript and PDF report output, as inches, centimeters, or points.
TOPMARGIN	Is the top boundary for a page of report output.
BOTTOMMARGIN	Is the bottom boundary for a page of report output.
LEFTMARGIN	Is the left boundary for a page of report output.
RIGHTMARGIN	Is the right boundary for a page of report output.
TARGETFRAME	Is a frame to which all drill-down hyperlinks are directed.
FOCEXURL	Is the location of a procedure and the method used to execute it from the WebFOCUS Client.
BASEURL	Is the default location where the browser searches for relative URLs specified in the HTML documents created by your application.

Displaying Information About the SU Machine

The ? SU command displays the communication available to the FOCUS Database Server.

Syntax: How to Display Information About the FOCUS Database Server

```
? SU [userid|ddname]
```

where:

userid

Is a sync machine user ID.

ddname

Is a valid ddname.

Example: Displaying Information About the FOCUS Database Server

Issuing the command

```
? SU SYNCA
```

produces the following information:

```
USERID      FILEID      QUEUE
WIBMLH      QUERY
WIBJBP      CAR
```

Displaying Data Sources Specified With USE

The ? USE command displays data sources specified with the USE command.

Syntax: How to Display Data Sources Specified With USE

To display data sources specified with the USE, issue the command:

```
? USE
```

Example: Displaying Data Sources Specified With USE

Issuing the command

```
? USE
```

produces information similar to the following:

```
 DIRECTORIES IN USE ARE: C:\IBI\apps\ibisamp\CENTINV.FOC AS CENTINV C:
\IBI\apps\ibisamp\CENTHR.FOC AS CENTHR
```

Displaying Global Variable Values

The ? && command lists Dialogue Manager global variables and the current values. Global variables maintain the values for the duration of the connection to the WebFOCUS Reporting Server.

Note: You can query all Dialogue Manager variables (local, global, system, and statistical) from a stored procedure by issuing:

```
-? &
```

See your Dialogue Manager documentation for details.

Syntax: How to Display Global Variable Values

```
? &&
```

Your site may replace the ampersand (& or &&) indicating Dialogue Manager variables, with another symbol. In that case, use the replacement symbol in your query command. For example, if your installation uses the percent sign (%) to indicate Dialogue Manager variables, list global variables by issuing:

```
? %%
```

Example: Displaying Global Variable Values

Issuing the command

```
? &&
```

produces information similar to the following:

```
 &&CITY 'STAMFORD' &&CODE1 'B10' &&CODE2 'B20'
```

Identifying the Files Being Used

You can debug your application by identifying the files you are using. You can identify the file name and location of a procedure, data source, Master File, Access File, or StyleSheet. WebFOCUS searches on the path of the WebFOCUS Reporting Sever and identifies the file using the WHENCE command.

The maximum length of the file name is 64 characters when using the WHENCE command.

Syntax: How to Query the Procedure Being Used

```
WHENCE procedure FOCEXEC
```

where:

procedure

Is a procedure.

Example: Querying the Procedure Being Used

Issuing the command

```
WHENCE FINANCE FOCEXEC
```

produces output similar to the following:

```
C:\ibi\apps\session\layout.fex
```

Syntax: How to Query the FOCUS Data Source Being Used

```
WHENCE focusfile FOCUS
```

where:

focusfile

Is a FOCUS data source.

Example: Querying the FOCUS Data Source Being Used

Issuing the command

```
WHENCE CENTINV FOCUS
```

produces output similar to the following:

```
C:\ibi\apps\session\centinv.foc
```

Syntax: How to Query the Master File Being Used

```
WHENCE masterfile MASTER
```

where:

masterfile

Is a Master File.

Example: Querying the Master File Being Used

Issuing the command

```
WHENCE CENTINV MASTER
```

produces output similar to the following:

```
C:\ibi\apps\session\centinv.mas
```

Syntax: How to Query the Access File Being Used

```
WHENCE accessfile ACCESS
```

where:

accessfile

Is an Access File.

Example: Querying the Access File Being Used

Issuing the command

```
WHENCE GKEIVP ACCESS
```

produces output similar to the following:

```
C:\ibi\srv\home\catalog\gkeivp.acx
```

Syntax: How to Query the StyleSheet Being Used

```
WHENCE stylesheet FOCSTYLE
```

where:

stylesheet

Is a StyleSheet.

Example: Querying the StyleSheet Being Used

Issuing the command

```
WHENCE GGORDERS FOCSTYLE
```

produces output similar to the following:

```
C:\ibi\apps\session\classic.sty
```

Reporting Dynamically From System Tables

You can issue report requests against a set of synonyms that dynamically gather information about your environment, including its applications, files, columns, directories, tables, indexes, and keys. You can also retrieve information about functions, SET parameters, and error files. These synonyms reside in the catalog directory under the server home directory and have the suffix FMI (FOCUS Metadata Interface).

Overview of System Table Synonyms

Each FMI synonym retrieves information about a specific set of files in your environment. If you examine an FMI Master File, the REMARKS and DESCRIPTION attributes document what data will be returned by each system table and each column within the system table.

Note: The system table synonyms may change in future releases. Therefore, you should not design applications that depend on the structure of the system table synonyms.

For example, following is a version of the SYSFILES Master File, which, by default, retrieves information about Master Files in your application path.

```
$-------------------------------------------------------------------------------$
$ Copyright (c) 2013 Information Builders, Inc. All rights reserved. @MFSM_NOPROLOG@ $
$-------------------------------------------------------------------------------$
$--CAN BE USED TO RETRIEVE DIRECTORY INFO - USE THE FOLLOWING TO DEFINE DIRECTORY
$--SQL FMI SET SYSFILES EDASYNM
$--SQL FMI SET SYSFILES FOCEXEC
FILE=SYSFILES, SUFFIX=FMI, REMARKS='Metadata: Directory information', $
SEGMENT=FILE,SEGTYPE=S0,$
  FIELD=FILENAME  ,      ,A64  ,A64B,    DESC='MEMBER  NAME
',$
  FIELD=LGNAME    ,      ,A8   ,A8B ,    DESC='LOGICAL NAME
',$
  FIELD=PHNAME    ,      ,A80  ,A80B,    DESC='PHYSICAL NAME  1ST PART
',$
  FIELD=PHNAME2   ,      ,A80  ,A80B,    DESC='PHYSICAL NAME  2ND PART
',$
  FIELD=PHNAME3   ,      ,A80  ,A80B,    DESC='PHYSICAL NAME  ETC..
',$
  FIELD=PHNAME4   ,      ,A80  ,A80B,    DESC='PHYSICAL NAME  The whole length up to
512 bytes                    ',$
  FIELD=PHNAME5   ,      ,A80  ,A80B,    DESC='PHYSICAL NAME  The last part is 32 but
can be up                    ',$
  FIELD=PHNAME6   ,      ,A80  ,A80B,    DESC='PHYSICAL NAME  to 44 because of nlscut
of ',$
  FIELD=PHNAME7   ,      ,A80  ,A80B,    DESC='PHYSICAL NAME  previous 6 parts (2
bytes per part)',$
  FIELD=VERSION   ,      ,I4   ,I1,     DESC='MF:VERSION
',$
  FIELD=MOD       ,      ,I4   ,I1,     DESC='MF:MODIFICATION NUMBER
',$
  FIELD=LINECNT   ,      ,I4   ,I2,     DESC='MF:CURRENT LINE COUNTER.
',$
  FIELD=DATE      ,      ,A8   ,A8B,     DESC='IBI DATE (DD/MM/YY)
',$
  FIELD=TIME      ,      ,A8   ,A8B,     DESC='IBI TIME (HH.MM.SS)
',$
  FIELD=USERID    ,      ,A100 ,A100B,   DESC='MF: LAST USER WHO CHANGED. UNIX/
NT:owner',$
  FIELD=SIZE      ,      ,I11  ,I4,     DESC='UNIX/NT:SIZE IN BYTES.
',$
  FIELD=EXTENSION ,      ,A3   ,A3B,     DESC='ACCEPTED SHORT EXTENSION FOR FILE
',$
```

A list of some of the most useful FMI synonyms follows. You can generate a list of system table synonyms by issuing a request against the systable synonym.

- **SYSAPPS.** Retrieves information about applications and the files within them.

- **SYSCOLUM.** Retrieves information about tables and their columns.
- **SYSDEFFN.** Retrieves DEFINE FUNCTION information.
- **SYSERR.** Retrieves information about error message files.
- **SYSFILES.** Retrieves directory information.
- **SYSIMP.** Retrieves impact analysis information.
- **SYSINDEX.** Retrieves index information.
- **SYSKEYS.** Retrieves information about keys.
- **SYSRPDIR.** Retrieves information about all available FOCEXECs in your application path.
- **SYSSET.** Retrieves information about SET commands and global Dialogue Manager variables.
- **SYSSQLOP.** Retrieves function information.
- **SYSTABLE.** Retrieves table information.
- **SYSVDTP.** Retrieves data type information for Relational Adapters.

SYSAPPS: Reporting on Applications and Application Files

The sysapps synonym retrieves information about applications and the files within them.

Example: Retrieving Application and File Information

The following request retrieves the application name and path and the file name, extension, suffix, keys, and number of segments for Master Files in the ibisamp application, where the file names start with the letters a through g.

```
TABLE FILE SYSAPPS
PRINT APPNAME AS App APPLOC AS Path
FNAME AS 'File,Name' SUFFIX AS 'File,Type' KEYS
NUMSEG AS '# of,Segments'
WHERE APPNAME EQ 'ibisamp'
WHERE FEXT EQ 'mas'
WHERE FQNAME LT 'C:\ibi\apps\ibisamp\hday'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

App	Path	File Name	File Type	KEYS	# of Segments
ibisamp	C:\ibi\apps\ibisamp	brokers	FOC	BROKER_ID	1
ibisamp	C:\ibi\apps\ibisamp	car	FOC	COUNTRY CAR MODEL BODYTYPE WARRANTY STANDARD	7
ibisamp	C:\ibi\apps\ibisamp	carolap	FOC	COUNTRY CAR MODEL BODYTYPE WARRANTY STANDARD	7
ibisamp	C:\ibi\apps\ibisamp	cashflow	FOC	CASH_DATE	1
ibisamp	C:\ibi\apps\ibisamp	course	FOC	COURSECODE	1
ibisamp	C:\ibi\apps\ibisamp	courses	FOC	COURSE_CODE	1
ibisamp	C:\ibi\apps\ibisamp	educfile	FOC	COURSE_CODE DATE_ATTEND EMP_ID	2
ibisamp	C:\ibi\apps\ibisamp	empdata	FOC	PIN	1
ibisamp	C:\ibi\apps\ibisamp	employee	FOC	EMP_ID DAT_INC TYPE PAY_DATE DED_CODE SKILLS DATE_ATTEND	11
ibisamp	C:\ibi\apps\ibisamp	experson	FOC	SOC_SEC_NO	1
ibisamp	C:\ibi\apps\ibisamp	filemntr	FIX		1
ibisamp	C:\ibi\apps\ibisamp	finance	FOC	YEAR ACCOUNT	1
ibisamp	C:\ibi\apps\ibisamp	ggdemog	FOC	ST	1
ibisamp	C:\ibi\apps\ibisamp	ggorder	FOC	ORDER_NUMBER	2
ibisamp	C:\ibi\apps\ibisamp	ggprods	FOC	PRODUCT_ID	1
ibisamp	C:\ibi\apps\ibisamp	ggsales	FOC	SEQ_NO	1
ibisamp	C:\ibi\apps\ibisamp	ggstores	FOC	STORE_CODE	1

SYSCOLUM: Reporting on Tables and Their Columns

The syscolum synonym retrieves table information, including table names creator names, segment names and numbers, segment roles in a dimension view or business view, column names, and column data types. Use it to report on data sources referenced in a Master File.

Example: Retrieving Table and Column Information

The following request retrieves table and column information from tables whose table names start with the characters *wf_*.

```
TABLE FILE SYSCOLUM
PRINT TBNAME AS Table TBTYPE AS Suffix NAME AS Field,Name COLTYPE AS
Data,Type ACTUAL AS Format
WHERE TBNAME LIKE 'wf_%'
WHERE RECORDLIMIT EQ 20
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

Table	Suffix	Field Name	Data Type	Format
wf_retail	SQLMSS	ID_SALES	INTEGER	I4
wf_retail	SQLMSS	ID_STORE	INTEGER	I4
wf_retail	SQLMSS	ID_CURRENCY	INTEGER	I4
wf_retail	SQLMSS	ID_CUSTOMER	INTEGER	I4
wf_retail	SQLMSS	ID_DISCOUNT	INTEGER	I4
wf_retail	SQLMSS	ID_PRODUCT	INTEGER	I4
wf_retail	SQLMSS	ID_TIME	INTEGER	I4
wf_retail	SQLMSS	COGS_LOCAL	DOUBLE	D8
wf_retail	SQLMSS	COGS_US	DOUBLE	D8
wf_retail	SQLMSS	DISCOUNT_LOCAL	DOUBLE	D8
wf_retail	SQLMSS	DISCOUNT_US	DOUBLE	D8
wf_retail	SQLMSS	GROSS_PROFIT_LOCAL	DOUBLE	D8
wf_retail	SQLMSS	GROSS_PROFIT_US	DOUBLE	D8
wf_retail	SQLMSS	MSRP_LOCAL	DOUBLE	D8
wf_retail	SQLMSS	MSRP_US	DOUBLE	D8
wf_retail	SQLMSS	QUANTITY_SOLD	INTEGER	I4
wf_retail	SQLMSS	REVENUE_LOCAL	DOUBLE	D8
wf_retail	SQLMSS	REVENUE_US	DOUBLE	D8
wf_retail	SQLMSS	SALE_UNITY	INTEGER	
wf_retail	SQLMSS	ID_SHIPFACT	INTEGER	I4

SYSDEFFN: Reporting on DEFINE FUNCTIONS

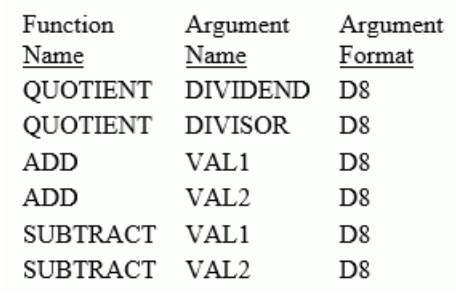
The sysdeffn synonym retrieves information about DEFINE FUNCTIONs, including function names, arguments, argument formats, function fields, and descriptions.

Example: Retrieving DEFINE FUNCTION Information

The following request retrieves DEFINE FUNCTION names, arguments, and argument formats.

```
TABLE FILE SYSDEFFN
PRINT DFNAME AS Function,Name ARGNAME AS Argument,Name ARGFORMAT AS
Argument,Format
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.



Function Name	Argument Name	Argument Format
QUOTIENT	DIVIDEND	D8
QUOTIENT	DIVISOR	D8
ADD	VAL1	D8
ADD	VAL2	D8
SUBTRACT	VAL1	D8
SUBTRACT	VAL2	D8

SYSERR: Reporting on Error Message Files

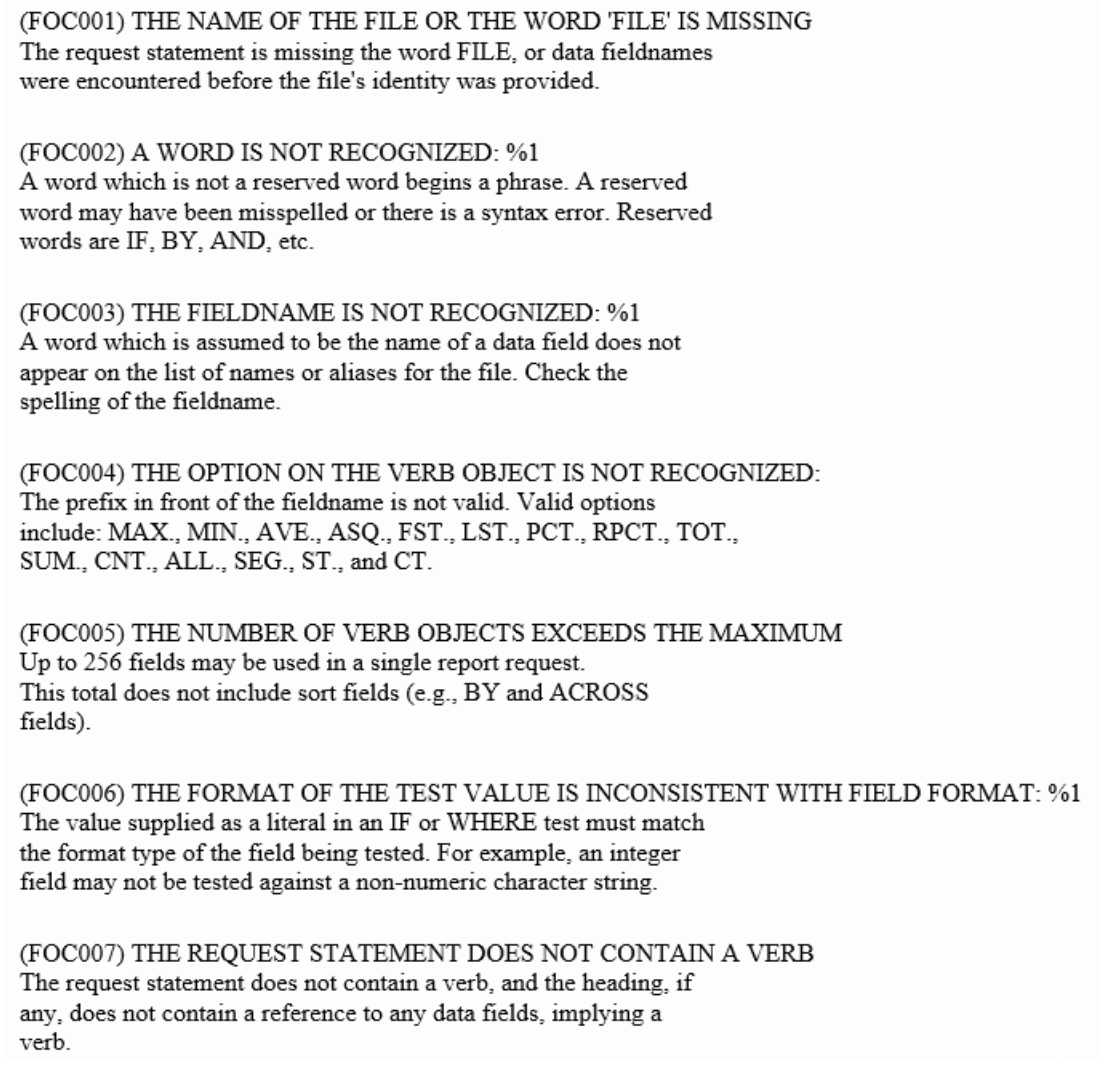
The syserr synonym retrieves error file names, the lowest and highest message numbers in each file, message and explanation text, message number, whether the message is a warning, whether the message is informational, and whether the line number is displayed in a procedure.

Example: Retrieving Error Message File Information

The following request retrieves message text and explanations.

```
TABLE FILE SYSERR
BY ERRNUM NOPRINT SUBHEAD
" <ERRTEXT  "
 "<ERRLINE1 "
 "<ERRLINE2 "
 "<ERRLINE3 "
 "<ERRLINE4 "
WHERE RECORDLIMIT EQ 7
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.



SYSFILES: Reporting on Metadata or Procedure Directory Information

The sysfiles synonym retrieves Master Files or FOCEXEC files in your application path. By default, sysfiles retrieves a list of Master Files and their properties. The SET SYSFILES command determines which type of files are retrieved.

The syntax is

```
SQL FMI SET SYSFILES {EDASYNM|FOCEXEC}
```

where:

EDASYNM
: Retrieves information about Master Files. This is the default value.

FOCEXEC
: Retrieves information about procedure files.

Example: **Retrieving Master File Information**

The following request retrieves the file name, extension, and path for the Master File names that start with the letters a through h in the ibisamp application directory.

```
TABLE FILE SYSFILES
PRINT FILENAME AS File
EXTENSION AS Extension
PHNAME AS Path
WHERE PHNAME LIKE 'ibisamp/%'
WHERE FILENAME LT 'i'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

File	Extension	Path
brokers	mas	ibisamp/brokers.mas
car	mas	ibisamp/car.mas
carolap	mas	ibisamp/carolap.mas
cashflow	mas	ibisamp/cashflow.mas
course	mas	ibisamp/course.mas
courses	mas	ibisamp/courses.mas
educfile	mas	ibisamp/educfile.mas
empdata	mas	ibisamp/empdata.mas
employee	mas	ibisamp/employee.mas
experson	mas	ibisamp/experson.mas
filemntr	mas	ibisamp/filemntr.mas
finance	mas	ibisamp/finance.mas
ggdemog	mas	ibisamp/ggdemog.mas
ggorder	mas	ibisamp/ggorder.mas
ggprods	mas	ibisamp/ggprods.mas
ggsales	mas	ibisamp/ggsales.mas
ggstores	mas	ibisamp/ggstores.mas
hday	mas	ibisamp/hday.mas

SYSIMP: Reporting on Impact Analysis Information

The sysimp synonym retrieves information about where files reside and where they are referenced. Sysimp contains a segment for caller information and a child segment for the files called by each caller.

Example: Retrieving Impact Analysis Information

The following request retrieves the names, types, and descriptions of caller files whose names start with the letters s through z in the ibisamp application and the names, types, and descriptions of the files they called.

```
TABLE FILE SYSIMP
PRINT CTYPE AS Caller,Type CDESCRIPTION AS Description
RFILE AS Called,Name
RPT_TYPE AS Called,Type
RLINENUM AS Line RUSAGE AS 'Used In'
REXTENSION AS Extension RDESCRIPTION AS Description
BY CFILE/A15  AS Caller,Name
WHERE CAPPLICATION EQ 'ibisamp'
WHERE CFILE GE 's'
ON TABLE SET PAGE NOLEAD
ON TABLE SET SHOWBLANKS ON
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The partial output is shown in the following image.

Caller Name	Caller Type	Description	Called Name	Called Type	Line	Used In	Extension	Description
subfoot1	Procedure		car	MFD	2	TABLE	mas	Legacy Metadata Sample: car
wf_retail	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_sales	MFD	4	CRFILE	mas	Sales Fact
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_customer	MFD	42	CRFILE	mas	Customer Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_age	MFD	211	CRFILE	mas	Age Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_time	MFD	967	CRFILE	mas	Time Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_time	MFD	935	CRFILE	mas	Time Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_industry	MFD	872	CRFILE	mas	Industry Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_occupation	MFD	344	CRFILE	mas	Occupation Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_marital_status	MFD	339	CRFILE	mas	Marital Status Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_income	MFD	326	CRFILE	mas	Income Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_geography	MFD	231	CRFILE	mas	Geography Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_education	MFD	220	CRFILE	mas	Education Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_currency	MFD	202	CRFILE	mas	Currency Dimension
	Synonym	Cluster Join of Fact Tables Sales, Shipments and Labor for Demo Database	wf_retail_time	MFD	170	CRFILE	mas	Time Dimension

SYSINDEX: Reporting on Index Information

The sysindex synonym retrieves information about indexes defined in a synonym.

Example: Retrieving Index Information

The following request retrieves index field names for files that start with the characters *gg*.

```
TABLE FILE SYSINDEX
PRINT NAME AS Index
BY TBNAME AS File
WHERE TBNAME LIKE 'gg%'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

File	Index
ggdemog	ST
ggorder	PRODUCT_ID
	VENDOR_CODE
ggprods	PRODUCT_ID
	VENDOR_CODE
ggsales	CATEGORY
	PCD
	REGION
	ST
	STCD
ggstores	STORE_CODE
	STATE

SYSKEYS: Reporting on Key Information

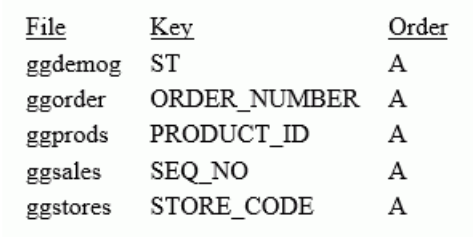
The syskeys synonym retrieves information about keys defined in a synonym.

Example: Retrieving Key Information

The following request retrieves key field names and sort order for files that start with the characters *gg*.

```
TABLE FILE SYSKEYS
PRINT IXNAME AS Key ORDERING AS Order
BY TBNAME AS File
WHERE TBNAME LIKE 'gg%'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.



File	Key	Order
ggdemog	ST	A
ggorder	ORDER_NUMBER	A
ggprods	PRODUCT_ID	A
ggsales	SEQ_NO	A
ggstores	STORE_CODE	A

SYSRPDIR: Reporting on Stored Procedures

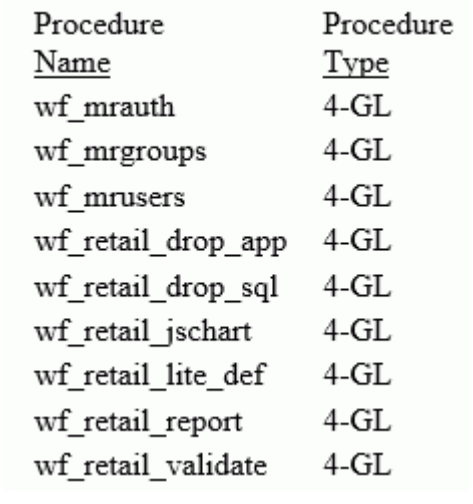
The sysrpdir synonym retrieves all available FOCEXECs in your application path.

Example: Retrieving Stored Procedure Information

The following request retrieves procedures that start with the characters *wf_*.

```
TABLE FILE SYSRPDIR
PRINT RPC_TYPE AS Procedure,Type
BY RPC_NAME AS Procedure,Name
WHERE RPC_NAME LIKE 'wf_%'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.



Procedure Name	Procedure Type
wf_mrauth	4-GL
wf_mrgroups	4-GL
wf_mrusers	4-GL
wf_retail_drop_app	4-GL
wf_retail_drop_sql	4-GL
wf_retail_jschart	4-GL
wf_retail_lite_def	4-GL
wf_retail_report	4-GL
wf_retail_validate	4-GL

SYSSET: Reporting on SET Parameters

The sysset synonym retrieves information about SET parameters, their allowed values, and their default values.

Example: Retrieving Information About SET Parameters

The following request displays SET parameters that start with the letter D, along with their descriptions and values.

```
TABLE FILE SYSSET
PRINT SETDESC CURR_VALUE VALUE IS_DEFAULT
BY SETNAME AS Set,Name
WHERE SETNAME GE 'D' AND SETNAME LT 'E'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The partial output is shown in the following image.

Set Name	Description	Current Value	Available Value	Default
DATEDISPLAY	Date display value when date functions evaluate as zero.	OFF	OFF	1
	Date display value when date functions evaluate as zero.	OFF	ON	0
	Date display value when date functions evaluate as zero.	OFF	COMP	0
DATEFNS	Activates year 2000-compliant versions of date subroutines.	ON	OFF	0
	Activates year 2000-compliant versions of date subroutines.	ON	ON	1
DATEFORMAT	Format of input string in DT() date/time function.	MDY	MDY	1
	Format of input string in DT() date/time function.	MDY	YMD	0
	Format of input string in DT() date/time function.	MDY	DMY	0
	Format of input string in DT() date/time function.	MDY	MYD	0
DATEOUTPUT	Enable locale-sensitive Date Format	DEFAULT	.	.
DATETIME	Sets time and date in reports and controls the format in which CREATE SYNONYM creates date-time columns in a Master File.	STARTUP	STARTUP	0
	Sets time and date in reports and controls the format in which CREATE SYNONYM creates date-time columns in a Master File.	STARTUP	STARTUP/RESET	0
	Sets time and date in reports and controls the format in which CREATE SYNONYM creates date-time columns in a Master File.	STARTUP	RESET	0
	Sets time and date in reports and controls the format in which CREATE SYNONYM creates date-time columns in a Master File.	STARTUP	CURRENT	0
	Sets time and date in reports and controls the format in which CREATE SYNONYM creates date-time columns in a Master File.	STARTUP	CURRENT/NOW	0
	Sets time and date in reports and controls the format in which CREATE SYNONYM creates date-time columns in a Master File.	STARTUP	NOW	0
DBACSENSITIV	Controls whether password validation is case-sensitive.	OFF	OFF	0
	Controls whether password validation is case-sensitive.	OFF	ON	0
DBAJOIN	Controls how to add VALUE restriction as JOIN condition or to TABLE request.	OFF	OFF	1
	Controls how to add VALUE restriction as JOIN condition or to TABLE request.	OFF	ON	0
DBASOURCE	Controls the source of access restrictions in a multi-file structure.	HOST	HOST	0
	Controls the source of access restrictions in a multi-file structure.	HOST	ALL	0

SYSSQLOP: Reporting on Function Information

The syssqlop synonym retrieves information about functions, their descriptions, parameters, syntax, and adapter category.

Example: Retrieving Function Descriptions and Syntax

The following request retrieves the names, descriptions, and syntax for the legacy functions whose names begin with the letters A and B.

```
TABLE FILE SYSSQLOP
SUM FUNCTION_DESC FUNCTION_SYNTAX
BY FUNCTION
WHERE CATEGORY LIKE 'L%'
WHERE FUNCTION LE 'C'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

Function name	Function description	Function syntax
ARGLEN	Returns the length of a character string, excluding trailing blanks. Use CHAR_LENGTH instead.	ARGLEN(length, string, output_format)
ATODBL	Converts a character string to double-precision format	ATODBL(string, length, output_format)
AYMDI	Add or subtract days to or from a date	AYMDI(arg1, arg2, arg3, arg4)
AYMI	Add or subtract months to or from dates	AYMI(arg1, arg2, arg3, arg4)
BAR	Produce a bar chart	BAR(barlength, infield, maxvalue, 'char', output_format)
BITSON	Determine if a bit is ON or OFF	BITSON(bitnumber, string, output_format)
BITVAL	Evaluate a bit string as a binary integer	BITVAL(string, startbit, number, output_format)
BNYSRC	Get the number of FOR value	BNYSRC(arg1)
BYTVAL	Translates a character to a decimal value	BYTVAL(character, output_format)

SYSTABLE: Reporting on Table Information

The systable synonym retrieves information about synonyms in your path, including type of synonym, creator, number of columns, keys, record length, and description.

Example: Retrieving A List of FMI Synonyms

The following request retrieves the names, descriptions, and attributes of the system table synonyms.

```
TABLE FILE SYSTABLE
PRINT TBTYPE REMARKS COLCOUNT RECLENGTH KEYCOLUMNS
BY NAME
WHERE NAME LIKE 'sys%'
WHERE TBTYPE EQ 'FMI'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

NAME	TBTYPE	REMARKS	COLCOUNT	RECLENGTH	KEYCOLUMNS
sysacfg	FMI	Metadata: Server admin information	14	2282	0
sysapps	FMI	Metadata: Applications and Files Information	123	51307	0
syscnv	FMI	SQL Adapter: Type Conversion Information	8	846	0
syscolum	FMI	Metadata: Column Information	121	8376	0
syscube	FMI	Metadata: Cube Information - Hierarchical	66	8763	0
syscube2	FMI	Metadata: Cube Information - Parent/Child	29	2478	0
sysdeffn	FMI	Metadata: DEFINE FUNCTION Information	17	3738	0
sysdirs	FMI	Metadata: File System Directory and File Information	17	1361	0
sysentty	FMI	Metadata: Entity Information - OBSOLETE	10	1100	0
syserr	FMI	Metadata: Error Files Information	12	826	0
sysfiles	FMI	Metadata: Directory information	17	774	0
sysfkeys	FMI	Metadata: Foreign Key Information	11	324	0
sysflow	FMI	Metadata: FOCEXEC Flow Information	91	9997	0
sysimp	FMI	Metadata For FMI Impact	31	3515	0
sysindex	FMI	Metadata: Index Information	42	986	0
syskeys	FMI	Metadata: Key Information	11	630	0
sysrpdir	FMI	Metadata: Available Stored Procedures	2	22	0
sysscha	FMI	Scheduler: Agent Information	21	2846	0
syssche	FMI	Scheduler: Event Information	7	704	0
sysset	FMI	Metadata: SET Information	34	2779	0
syssqlop	FMI	Functions Optimization Information for SQL Adapters	23	4090	0
systable	FMI	Metadata: Table Information	40	1149	0

Reporting on Data Types

The sysvdtp synonym retrieves data types and their corresponding USAGE and ACTUAL formats for the SQL Adapters and for fixed sequential data sources. It is not an FMI synonym, but retrieves the data type information from a delimited sequential file.

Example: Retrieving Data Types for the Adapter for MySQL

The following request retrieves data type information for the Adapter for MySQL

```
TABLE FILE SYSVDTP
PRINT DATA_TYPE_CATEGORY
VENDOR_DATA_TYPES
TYPE_RANGE
SERVER_USAGE_CATEGORY
SERVER_USAGE
SERVER_ACTUAL
REMARKS
BY ADAPTER
WHERE SUFFIX EQ 'SQLMYSQL'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

Adapter	Data Type Category	Vendor Data Types	Type Range	Server Data Type	Server USAGE	Server ACTUAL	Remarks
MySQL	Date-Time	DATE		Date-Time	YYMD	DATE	
	Date-Time	TIME		Date-Time	HHIS	HHIS	
	Date-Time	TIMESTAMP/DATETIME		Date-Time	HYYMDS	HYYMDS	
	Date-Time	TIMESTAMP WITH TIME ZONE			N/A	N/A	
	Date-Time	YEAR		Numeric	I6	I4	
	Numeric	TINYINT/SMALLINT		Numeric	I6	I2	
	Numeric	TINYINT(1)/BIT/BOOL		Numeric	I11	I2	
	Numeric	INTEGER/MEDIUMINT		Numeric	I11	I4	
	Numeric	INTEGER UNSIGNED		Numeric	P11	P8	
	Numeric	BIGINT		Numeric	P20	P11	
	Numeric	DECIMAL(p,s)/NUMERIC(p,s)	p=1..31,s=0	Numeric	Pn	Pk	n=p÷1,k=(p/2)÷1
	Numeric	DECIMAL(p,s)/NUMERIC(p,s)	p=1..31,s>0	Numeric	Pn.m	Pk	n=p÷2,m=min(s,31),k=(p/2)÷1
	Numeric	DECIMAL(p,s)/NUMERIC(p,s)	p>32,s=0	Numeric	P32	P16	
	Numeric	DECIMAL(p,s)/NUMERIC(p,s)	p>32,s>0	Numeric	P33.m	P16	m=min(s,31)
	Numeric	REAL/FLOAT/DOUBLE PRECISION		Numeric	D20.2	D8	
	LOB and Other	TEXT/MEDIUMTEXT/LONGTEXT		Text	TX50	TX	
	LOB and Other	TINYTEXT		Alphanumeric	A255V	A255V	
	LOB and Other	BLOB/TINYBLOB/LONGBLOB		BLOB	BLOB	BLOB	
	LOB and Other	VARBINARY(n)	n>16383	BLOB	BLOB	BLOB	
	LOB and Other	ENUM		Alphanumeric	An	An	
	LOB and Other	SET		Alphanumeric	An	An	MySQL JDBC Connector version has to be 5.1.7 or higher
	LOB and Other	GEOMETRY		Text	TX50	TX	Server supports this type as read-only via spatial properties:
	LOB and Other			Text			- Master File: GEOGRAPHIC_ROLE=GEOMETRY_AREA;
	LOB and Other			Text			- Access File: SQL_FLD_OBJ_TYPE=OPAQUE,
	LOB and Other			Text			SQL_FLD_OBJ_PROP=GEOMETRY_SHAPE,
	LOB and Other			Text			SQL_FLD_OBJ_EXPR='DB_EXPR()'
	LOB and Other			Text			where DB_EXPR() are vary depends on DBMS
	Character	CHAR(n)		Alphanumeric	An	An	
	Character	VARCHAR(n)		Alphanumeric	AnV	AnV	n>32765 will be truncated
	Character	BINARY(n)		Alphanumeric	Am	Am	m=2*n
	Character	VARBINARY(n)	n<=16383	Alphanumeric	Am	Am	m=2*n
	Character, Unicode	CHAR(n)		Alphanumeric	An	An	
	Character, Unicode	VARCHAR(n)		Alphanumeric	AnV	AnV	n>32765 will be truncated

Chapter 7

Accessing a FOCUS Data Source

This topic describes how to access a FOCUS data source with the USE command. Some of the functionality of the USE command can be performed with the DATASET attribute in the Master File. For information on the DATASET attribute, see the *Describing Data With WebFOCUS Language* manual.

This topic contains information and examples for the following operating systems: Windows, z/OS, and UNIX.

You can also use Universal Concatenation to retrieve data from unlike data sources in a single request. See the *Creating Reports With WebFOCUS Language* manual for more information.

You must issue a USE command in order to work with an external index.

In this chapter:

- The USE Command
- Identifying a FOCUS Data Source
- Using Alternative File Specifications
- Identifying a New Data Source
- Protecting a Data Source
- Concatenating Data Sources
- Displaying the Current USE Options
- Clearing the USE Options

The USE Command

When you issue a command to access a FOCUS data source, such as TABLE FILE filename, WebFOCUS searches for a Master File with the specified file name, and then searches for a data source with the same name. The USE command specifies the name and location of a FOCUS data source. This is useful under the following conditions:

- The default naming convention is not used.

- The data source cannot be found on the standard search path.
- An explicit extra option is desired.

When you identify a FOCUS data source with the USE command, a USE directory is created, which is a list of data source definitions. When a USE directory is in effect, WebFOCUS locates a data source using the information in the directory instead of searching for it using the default name and search path. A USE directory enables you to access up to 1022 data sources and applies only to FOCUS data sources. You can simplify the creation of a USE directory in the following ways:

- Include all USE specifications in a single file that you call with the -INCLUDE command at the beginning of each procedure that needs to access the USE directory. For details on the -INCLUDE command, see *Calling Another Procedure With -INCLUDE* on page 401.
- Include all USE specifications in a WebFOCUS Reporting Server profile. When the profile is invoked, the USE commands will be active.

The DATASET attribute in the Master File can perform some, but not all, of the same tasks as the USE command. For information on DATASET, see the *Describing Data With WebFOCUS Language* manual.

Reference: Additional File Specifications for FOCUS Data Sources

On some operating systems, WebFOCUS also uses additional file specifications to identify a FOCUS data source:

- On **Windows** and **UNIX**, the FOCUS data source has a default file extension of .FOC. For example, the command TABLE FILE EMPLOYEE uses EMPLOYEE.FOC.
- On **z/OS**, additional specifications are not required. The command TABLE FILE EMPLOYEE uses the data source allocated to ddname EMPLOYEE.

Identifying a FOCUS Data Source

You can identify a FOCUS data source with the USE command (see *How to Identify a Data Source With the USE Command* on page 489).

You can use a single USE command to specify one or several data sources. For examples, see *Identifying a Single Data Source* on page 491 and *Identifying Multiple Data Sources* on page 491.

Syntax: How to Identify a Data Source With the USE Command

```
USE {ADD|CLEAR|REPLACE}
fileid [READ|NEW] [AS mastername] [ON resource]
[fileid [READ|NEW] [AS mastername] [ON resource]]
.
.
.
END
```

where:

ADD

Appends one or more new file IDs to the present directory. If you issue the USE command without the ADD option, the list of data sources you specify replaces the existing USE directory.

CLEAR

Erases the USE directory. Any other specified options are ignored. The END command is not required with this option.

REPLACE

Replaces an existing file ID in the USE directory. This option enables you to change the specification for the file ID and the options following the ID.

fileid

Is any valid file name for the specific operating system.

The following table lists and describes file names for each operating system:

Operating System	Description
Windows:	Any valid file name in *drive*:*path**filename.ext* format. The drive and path specifications are optional. **Note:** When using Application Namespace technology, you do not have to specify the complete path. Use any valid file name in the *app_name*[*/app_name_a*]*/filename.ext* format (for example, ibisamp/ggsales.foc).
z/OS:	A *ddname* allocated to the full z/OS specification of the file.

Operating System	Description
UNIX:	Any valid file name in */directory/filename.ext* format. The directory specification is optional. **Note:** When using Application Namespace technology, you do not have to specify the complete path. Use any valid file name in the *app_name*[*/app_name_a*]*/filename.ext* format (for example, ibisamp/ggsales.foc).

READ

Restricts the data source to read-only access.

NEW

Indicates that the data source is new.

AS *mastername*

Specifies the name of the Master File associated with the file ID.

Both the READ and NEW option can be used with the AS option. Any other combination of options after the file ID is *not* valid.

Note: When using the AS phrase for files on the FOCUS Database Server with ON, the READ parameter is required. Write access to these files is prevented, as they are being accessed with Master Files other than those with which they were created. Only READ access is allowed in this case

ON *resource*

Specifies a specially configured WebFOCUS Reporting Server that synchronizes FOCUS data sources for use by multiple end users.

Note: When using the AS phrase for files on the FOCUS Database Server with ON, the READ parameter is required. Write access to these files is prevented, as they are being accessed with Master Files other than those with which they were created. Only READ access is allowed in this case

Example: Identifying a Single Data Source

The following table shows sample USE commands you can use to specify a data source.

Operating System	Use Command Example

Windows:	`USE`
	`EMP026.FOC AS PRODUCTS`
	`END`

z/OS:	`USE`
	`EMP026 AS PRODUCTS`
	`END`

UNIX:	`USE`
	`emp026.foc AS PRODUCTS`
	`END`

Example: Identifying Multiple Data Sources

You can specify several data sources in one USE command, each with different parameters. The following table includes sample USE commands.

Operating System	Use Command Example

Windows:	`USE`
	`EMP024.FOC ON MULTID`
	`EMP025.FOC AS PRODUCTS`
	`EMP026.FOC READ AS ACCOUNTS`
	`END`

z/OS:	`USE`
	`EMP024 ON MULTID`
	`EMP025 AS PRODUCTS`
	`EMP026 READ AS ACCOUNTS`
	`END`

UNIX:	`USE`
	`emp024.foc ON MULTID`
	`emp025.foc AS PRODUCTS`
	`emp026.foc READ AS ACCOUNTS`
	`END`

Using Alternative File Specifications

Issue the USE command to specify an alternate file specification for a FOCUS data source if:

- The data source has a name different from the associated Master File.
- The data source has a type or extension different from the default.
- The data source is located in a directory outside the standard search path.

Note: A data source named by the LOCATION attribute in a Master File assumes default values unless you issue a USE command. If the physical file has a specification or location other than the default, it must be in the USE directory.

Syntax: How to Use an Alternative File Specification

```
USE
fileid AS mastername END
```

where:

fileid

Is any valid file name for the specific operating system.

mastername

Is the name of the Master File associated with the file ID.

Example: Specifying a Different File Name

To read the EMP026 data source described by the Master File EMPLOYEE, issue the USE command, then you can read EMP026 with the command TABLE FILE EMPLOYEE.

Operating System	Use Command Example

Windows:	`USE`
	`EMP026.FOC AS EMPLOYEE`
	`END`

z/OS:	`USE`
	`EMP026 AS EMPLOYEE`
	`END`

Operating System	Use Command Example

UNIX:	`USE`
	`emp026.foc AS EMPLOYEE`
	`END`

Example: Specifying a Different File Type or Extension

Windows:

To read the EMP026 data source with an extension of .DAT, described by the Master File EMPLOYEE, issue the command:

```
USE
EMP026.DAT AS EMPLOYEE
END
```

z/OS:

To read the emp026 data source with an extension of .dat, described by the Master File EMPLOYEE, issue the command:

```
USE
EMP026 AS EMPLOYEE
END
```

UNIX:

To read the emp026 data source with an extension of .dat, described by the Master File EMPLOYEE, issue the command:

```
USE
emp026.dat AS EMPLOYEE
END
```

After issuing the USE command, you can read EMP026 with the command TABLE FILE EMPLOYEE.

Example: Specifying a Different File Location

Windows:

To read the EMP026 data source located in the C:\ACCOUNTING\ directory, described by the Master File EMPLOYEE, issue the command:

```
USE
C:\ACCOUNTING\EMP026.FOC AS EMPLOYEE
END
```

UNIX:

To read the emp026 data source located in the /accounting/ directory, described by the Master File EMPLOYEE, issue the command:

```
USE
/accounting/emp026.foc AS EMPLOYEE
END
```

Identifying a New Data Source

You can identify a data source that does not exist yet. When you identify a new data source, you can accept the default file specification conventions or specify different ones.

Note: For z/OS, you must allocate the file before issuing the USE command. You can do this using the z/OS command ALLOCATE or the WebFOCUS command DYNAM. You must also issue the z/OS command CREATE to physically create the file. This is not a requirement for platforms that dynamically allocate files.

Syntax: How to Identify a New Data Source

```
USE
fileid NEW
END
```

where:

fileid

Is any valid file specification for the operating system. The file ID will be assigned to the data source when it is created.

NEW

Specifies that the data source does not yet exist. If you omit the NEW parameter, a message is returned stating that the data source cannot be found. The USE command is not executed.

Example: Identifying a New Data Source

The following command creates the data source WAGES using the WAGES Master File.

Windows:

```
USE
C:\DATA\WAGES.FOC NEW
END
CREATE FILE WAGES
```

z/OS:

```
USE
WAGES NEW
END
CREATE FILE WAGES
```

UNIX:

```
USE
wages.foc NEW
END
CREATE FILE WAGES
```

Protecting a Data Source

You can protect a data source from changes by including the READ parameter when issuing a USE command. A protected data source can be read by various tools and commands, but cannot be changed.

Syntax: How to Protect a Data Source

```
USE
fileid READ
END
```

where:

fileid

Is any valid file specification for the operating system.

READ

Restricts the data source to read-only access.

Example: Protecting a Data Source

The following command protects the EMPLOYEE data source from modification.

Windows:

```
USE
EMPLOYEE.FOC READ
END
```

z/OS:

```
USE
EMPLOYEE READ
END
```

UNIX:

```
USE
/usr/mydata/employee.foc READ
END
```

Concatenating Data Sources

If several FOCUS data sources are described by the same Master File, you can read all of them in one request by issuing a USE command that concatenates them.

You can also concatenate multiple files that cross-reference a common file. This is done by specifying the host files in the USE command, then specifying the cross-reference file.

Note: You cannot specify a concatenated file as a cross-reference file in a JOIN command.

Syntax: How to Concatenate Data Sources

```
USE
fileid-1 AS mastername
fileid-2 AS mastername
.
.
fileid-n AS mastername
END
```

where:

fileid-1...fileid-n

Are any valid file specifications for the files being concatenated.

mastername

Is the name of the Master File that describes the data sources.

Example: Concatenating Data Sources

The following command concatenates the FOCUS data sources EMP024, EMP025, and EMP026, all described by the Master File EMPLOYEE. You can then read all three data sources with a single TABLE FILE EMPLOYEE command.

Windows:

```
USE
C:\DATA\EMP024.FOC AS EMPLOYEE
C:\DATA\EMP025.FOC AS EMPLOYEE
C:\DATA\EMP026.FOC AS EMPLOYEE
END
```

z/OS:

```
USE
EMP024 AS EMPLOYEE
EMP025 AS EMPLOYEE
EMP026 AS EMPLOYEE
END
```

UNIX:

```
USE
/usr/mydata/emp024.foc AS EMPLOYEE
/usr/mydata/emp025.foc AS EMPLOYEE
/usr/mydata/emp026.foc AS EMPLOYEE
END
```

Example: Specifying Multiple Concatenations

The following command concatenates the EMP01 and EMP02 data sources described by the Master File EMPLOYEE, and concatenates the SALES01 and SALES02 data sources, described by the Master File SALES.

Windows:

```
USE
EMP01.FOC AS EMPLOYEE
EMP02.FOC AS EMPLOYEE
SALES01.FOC AS SALES
SALES02.FOC AS SALES
END
```

z/OS:

```
USE
EMP01 AS EMPLOYEE
EMP02 AS EMPLOYEE
SALES01 AS SALES
SALES02 AS SALES
END
```

UNIX:

```
USE
emp01.foc AS EMPLOYEE
emp02.foc AS EMPLOYEE
sales01.foc AS SALES
sales02.foc AS SALES
END
```

You can read the EMP01 and EMP02 data sources with the TABLE FILE EMPLOYEE command. You can read the SALES01 and SALES02 data sources with the TABLE FILE SALES command.

Syntax: How to Concatenate Data Sources and a Single Cross-Reference File

```
USE
fileid-1 AS mastername
fileid-2 AS mastername
.
.
fileid-n AS mastername
reffileEND
```

where:

fileid-1...fileid-n

Are any valid file specifications for the files being concatenated.

mastername

Is the name of the Master File that describes the data sources.

reffile

Is the cross-reference file for the files being concatenated.

Example: Concatenating Data Sources and a Single Cross-Reference File

EMPLOYEE is made up of the two files EMP01 and EMP02, both of which cross-reference the common file EDUCFILE. To read the files together, issue the USE command.

Windows:

```
USE
EMP01.FOC AS EMPLOYEE
EMP02.FOC AS EMPLOYEE
EDUCFILE.FOC
END
```

z/OS:

```
USE
EMP01 AS EMPLOYEE
EMP02 AS EMPLOYEE
EDUCFILE
END
```

UNIX:

```
USE
/usr/mydata/emp01.foc AS EMPLOYEE
/usr/mydata/emp02.foc AS EMPLOYEE
/usr/mydata/educfile.foc
END
```

Example: Concatenating Data Sources and Multiple Cross-Reference Files

If EMPLOYEE consists of two files, EMP01 and EMP02, and each has its own cross-reference file, ED01 and ED02, you can read all four files in one command. Issue the USE command in which each host file is followed by its cross-reference file.

Windows:

```
USE
EMP01.FOC AS EMPLOYEE
ED01.FOC AS EDUCFILE
EMP02.FOC AS EMPLOYEE
ED02.FOC AS EDUCFILE
END
```

z/OS:

```
USE
EMP01 AS EMPLOYEE
ED01  AS EDUCFILE
EMP02 AS EMPLOYEE
ED02  AS EDUCFILE
END
```

UNIX:

```
USE
/usr/mydata/emp01.foc AS EMPLOYEE
/usr/mydata/ed01.foc AS EDUCFILE
/usr/mydata/emp02.foc AS EMPLOYEE
/usr/mydata/ed02.foc AS EDUCFILE
END
```

Displaying the Current USE Options

To display the USE options in effect, issue the ? USE query. This query displays a list of data sources specified with the USE command and the options currently in effect.

Syntax: How to Display the Current USE Options

```
? USE
```

Example: Displaying Current USE Options

A sample response to the ? USE command is:

```
 DIRECTORIES IN USE ARE:
C:\IBI\apps\ibisamp\CENTINV.FOC AS CENTINV
C:\IBI\apps\ibisamp\CENTHR.FOC AS CENTHR
```

Clearing the USE Options

You can clear the USE directory with the USE command.

Syntax: How to Erase the USE Directory

```
USE CLEAR
```

Chapter 8

Customizing Your Environment

You can use the SET command to change parameters that govern the WebFOCUS environment. SET command parameters control the content of reports and graphs, and the appearance of reports and graphs on the screen and when printed. These parameters also control system responses to end user requests, and data retrieval characteristics that affect performance. SET commands also help you set up metadata and manipulate information, such as dates.

The SET command governs many important aspects of your work.

In this chapter:

- When Do You Use the SET Command?
- Ways to Issue a SET Command
- Coding a SET Command
- Types of SET Parameters
- SET Parameter Syntax

When Do You Use the SET Command?

If you are an application developer, use the SET command to:

- Help you work efficiently and meet your testing and debugging needs.
- Provide a uniform and appropriate run-time environment for your end users with desirable defaults that do not need customization.

If you are creating your own ad hoc reports, use SET commands when you need to tailor the report presentation or content to meet your individual needs.

Ways to Issue a SET Command

You can issue a SET command by using the SET tool to generate the command in the stored procedure, or by coding it in a stored procedure. If you use the SET tool, a SET command will be generated for the report you are developing. If you code the SET command, you have the following options for its placement:

- In a supported profile on a WebFOCUS Reporting Server. When issued in a profile, the setting applies to the server environment during the connection for executing the current procedure.
- In a stored procedure:
  - Before the report request.
  - With ON TABLE SET or ON GRAPH SET in a report or graph request. SET used in this manner applies only to the report or graph request in which it is issued, even though there may be multiple report or graph requests in the stored procedure.
- In a type Other reporting object, in the Managed Reporting administrator environment. A setting issued in a reporting object (type Other) applies to a single report template.
- In the WebFOCUS script file site.wfs. When issued in site.wfs, the setting applies to all connections to the WebFOCUS Reporting Server from the WebFOCUS Client.

Reference: Precedence of the SET Command

If you issue more than one SET command for the same parameter, the precedence is as follows:

WebFOCUS

1. SET in a stored procedure.
2. SET in the site.wfs or ibidir.wfs file.
3. SET in a profile on the WebFOCUS Reporting Server.
   a. In a global profile.
   b. In a group profile.
   c. In a user profile.

Coding a SET Command

The following guidelines apply to SET command syntax:

- You can set several parameters in one command by separating each with a comma.
- You can include as many parameters as you can fit on one line. If you exceed one line, repeat the SET command for each new line.
- You can set many, but not all, parameters using ON TABLE SET or ON GRAPH SET within a request. Parameters that cannot be set in this way are specified in the detailed description.

For the specific syntax of a parameter with its valid values, see *SET Parameter Syntax* on page 525.

Syntax: How to Set Parameters

```
SET parameter = option[, parameter = option,...]
```

where:

parameter

Is the setting you wish to change.

option

Is a valid value for the parameter.

Example: Setting a Single Parameter

In the following example, the PAGE-NUM parameter suppresses default page numbering.

```
SET PAGE-NUM = OFF

TABLE FILE EMPLOYEE
PRINT LAST_NAME FIRST_NAME
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
END
```

The output is:

```
LAST_NAME                       FIRST_NAME
```

```
STEVENS                         ALFRED
SMITH                           MARY
JONES                           DIANE
SMITH                           RICHARD
BANNING                         JOHN
IRVING                          JOAN
ROMANS                          ANTHONY
MCCOY                           JOHN
BLACKWOOD                       ROSEMARIE
MCKNIGHT                        ROGER
GREENSPAN                       MARY
CROSS                           BARBARA
```

Note: In this example, the output was formatted to eliminate grids. For information on formatting your output, see the *Creating Reports With WebFOCUS Language* manual.

Example: Setting Multiple Parameters

The following example sets two parameters in one command in a stored procedure. The first parameter, NODATA, changes the default character for missing data from a period to the word NONE. The second parameter, PAGE-NUM, suppresses default page numbering.

```
SET NODATA = NONE, PAGE-NUM = OFF
TABLE FILE EMPLOYEE
PRINT CURR_SAL BY EMP_ID
ACROSS DEPARTMENT
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
END
```

In the output, NONE appears when there is no salary information for a specific employee because that employee does not work in the department that is referenced. There is no page number at the top of the output.

The output is:

```
                        DEPARTMENT
EMP_ID          MIS                PRODUCTION
071382660             NONE         $11,000.00
112847612      $13,200.00                NONE
117593129      $18,480.00                NONE
119265415             NONE          $9,500.00
119329144             NONE         $29,700.00
123764317             NONE         $26,862.00
126724188             NONE         $21,120.00
219984371      $18,480.00                NONE
326179357      $21,780.00                NONE
451123478             NONE         $16,100.00
543729165       $9,000.00                NONE
818692173      $27,062.00                NONE
```

Note: In this example, the output was formatted to eliminate grids. For information on formatting your output, see the *Creating Reports With WebFOCUS Language* manual.

Syntax: How to Set Parameters in a Report Request

```
ON TABLE SET parametervalue [AND parametervalue ...]
```

where:

parameter
Is the setting you wish to change.

value
Is a valid value for the parameter.

Example: Setting Parameters in a Report Request

In the following example, the command ON TABLE SET changes the default character for missing data from a period to the word NONE and suppresses default page numbering.

```
TABLE FILE EMPLOYEE
PRINT CURR_SAL BY EMP_ID
ACROSS DEPARTMENT
ON TABLE SET NODATA NONE AND PAGE-NUM OFF
ON TABLE SET STYLE *
TYPE=REPORT, GRID=OFF,$
END
```

In the output, NONE appears when there is no salary information for a specific employee. There is no page number at the top of the output.

The output is:

```
                    DEPARTMENT
EMP_ID          MIS             PRODUCTION

071382660             NONE       $11,000.00
112847612       $13,200.00             NONE
117593129       $18,480.00             NONE
119265415             NONE        $9,500.00
119329144             NONE       $29,700.00
123764317             NONE       $26,862.00
126724188             NONE       $21,120.00
219984371       $18,480.00             NONE
326179357       $21,780.00             NONE
451123478             NONE       $16,100.00
543729165        $9,000.00             NONE
818692173       $27,062.00             NONE
```

Note: In this example, the output was formatted to eliminate grids. For information on formatting your output, see the *Creating Reports With WebFOCUS Language* manual.

Syntax: How to Set Parameters in a Graph Request

```
ON GRAPH SET parametervalue [AND parametervalue ...]
```

where:

parameter

Is the setting you wish to change.

value

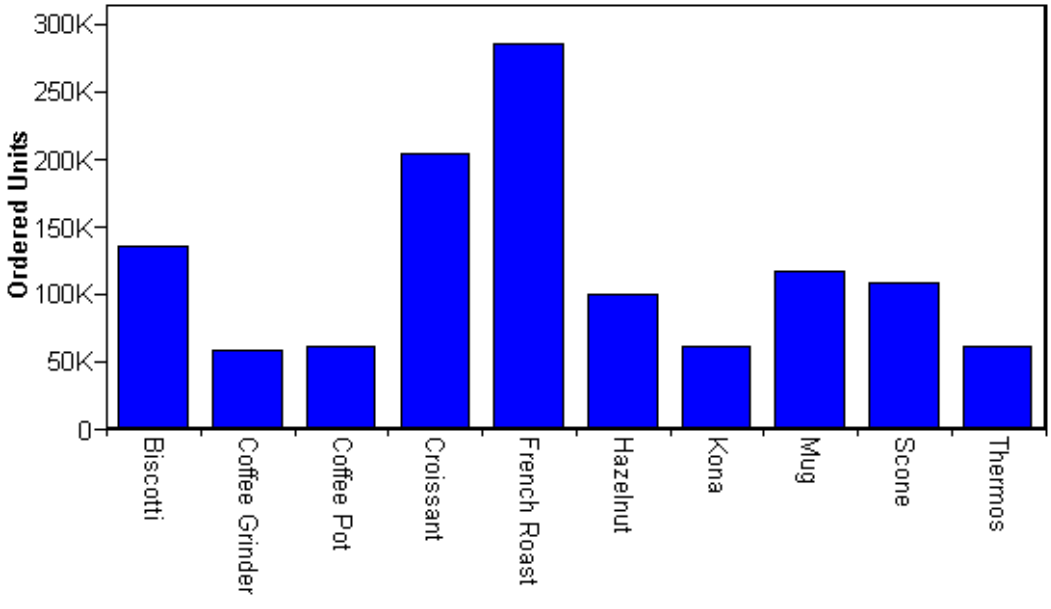
Is a valid value for the parameter.

Example: Setting Parameters in a Graph Request

In the following example, the command ON GRAPH SET changes the default setting for the 3D parameter to OFF.

```
GRAPH FILE GGORDER
SUM QUANTITY
ACROSS PRODUCT_DESC
ON GRAPH SET 3D OFF
END
```

The output is:



Types of SET Parameters

This topic lists the types of tasks that can be accomplished, and the SET parameters that allow you to perform these tasks. If a single parameter applies to more than one activity, it appears in more than one category. For more detailed descriptions, as well as the syntax for each parameter, see *SET Parameter Syntax* on page 525.

The following are the types of tasks performed with SET parameters.

Calculations	Affects the way calculations are performed in FOCUS.
Data and Metadata	Determines the way data is stored and processed.
Date Manipulation Tasks	Controls the way dates are processed and displayed in reports.
WebFOCUS-Specific Tasks	Affects the behavior of web-specific tasks in your application.
Graph Tasks	Controls the processing and display of graphs.
Memory Setup and Optimization Tasks	Affects the memory and optimization of your application.
Report Code, Content, and Processing Tasks	Determines the content and processing of a request.
Report Layout and Display Tasks	Affects the display of a report.
Security Tasks	Controls user access to data sources and procedures.

Calculations

The following parameters control the behavior of calculations in WebFOCUS.

CDN	Specifies the punctuation used in numeric notation.
COMPUTE	Controls the compilation of expressions.
DMPRECISION	Specifies precision of numeric values in Dialogue Manager -SET commands to calculate accurate numeric variable values.

FLOATMAPPING	Takes advantage of decimal-based precision numbers for all numeric processing for floating point numbers.
MISSINGTEST	Determines whether the IF expression in IF-THEN-ELSE tests is checked for missing values.
USERFCHK	Controls the level of verification applied to DEFINE FUNCTION arguments and Information Builders-supplied function arguments.
USERFNS	Determines whether an Information Builders-supplied function or a locally-written function with the same name used.
PACKPOS	In WebFOCUS, defines the positive character value for packed fields.
%STRICTMATH	In WebFOCUS, determines the behavior of a calculation when dividing by zero.

Data and Metadata

The following parameters determine the way data is stored and processed.

ACCBLN	Accepts blank or zero values for fields with ACCEPT commands in the Master File.
ASNAMES	Controls the FIELDNAME attribute in a HOLD Master File.
COUNTWIDTH	Expands the default format of COUNT fields from a five byte integer to a nine byte integer.
DATEFORMAT	Specifies the order of the date components (month/day/year) when date-time values are entered in a formatted-string or translated-string format.
DEFINES	Compiles virtual fields into machine code to improve performance.
DIRECTHOLD	Controls whether HOLD Files in FOCUS format are created directly.
DTSTRICT	Controls the use of strict processing for date-time fields.

EQTEST	Controls whether the characters $ and $* are treated as wildcard characters or normal characters in selection criteria.
EUROFILE	Activates the data source that contains information for the currency you want to convert.
FIELDNAME	Controls the use of long and qualified field names.
FOCHTMLURL	Allows you to access resources with an alias other than /ibi_apps/ibi_html.
FOCTRANSFORM	Translates field and segment names in a business view to their corresponding names in the original Master File for request processing.
HLDCOM_TRIMANV	Controls whether trailing blanks are retained when the output is held in a delimited format.
HNODATA	Controls missing values propagated to a HOLD file.
HOLDFORMAT	Determines the default format for HOLD files.
HOLDLIST	Determines what fields in a report request are included in the HOLD file.
HOLDMISS	Distinguishes between missing data and default data (zeros or blanks) in a HOLD file.
HOLDSTAT	Determines if comments and DBA information are included in HOLD Master Files.
HTMLARCHIVE	Packages HTML and DHTML reports together with image files into a single web archive document (.mht file).
HTMLENCODE	Encodes data within HTML output.
KEEPDEFINES	Controls whether a virtual field created for a host or joined structure is retained after a JOIN command is run.
MAXDATAEXCPT	Enables you to change the number of data exceptions allowed before the session is terminated.
MDICARDWARN	Displays a warning message when the cardinality of a dimension exceeds a specified value.

MDIENCODING	Enables retrieval of output from an MDI file without reading the data source.
MDIPROGRESS	Displays messages about the progress of an MDI build.
NULL	Enables creation of a variable-length comma or tab delimited HOLD file that differentiates between a missing value and a blank string or zero value.
OLDSTYRECLEN	Determines whether the record length, LRECL, is set to the current setting of LRECL=0, or the older setting of LRECL=512.
PCOMMA	Enables retrieval of comma delimited files created by a PC application or HOLD FORMAT COM command.
QUALCHAR	Specifies the qualifying character to be used in qualified field names.
RANK	Determines how rank numbers are assigned in a request when multiple data values fall into the same rank category.
SAVEDMASTERS	Saves a Master File to memory after it has been used in a request.
SHADOW	Activates the Absolute File Integrity feature.
SHIFT	Controls the use of *shift* strings.
TEMPERASE	Determines if the temporary files created during a WebFOCUS connection are retained after the connection is closed.
WEBARCHIVE	Packages multiple EXL2K files into a single file.
WEEKFIRST	Specifies what day of the week is the start of the week.
WPMINWIDTH	Specifies a minimum width for format WP output files.
XRETRIEVAL	Controls the retrieval of data when previewing a report.

Date Manipulation Tasks

The following parameters control the way dates are processed and displayed in reports.

BUSDAYS	Specifies which days are considered business days and which are not.
DATE_ORDER	Specifies the order of date components.
DATE_SEPARATOR	Specifies the separator for date components.
DATEDISPLAY	Controls the display of date format fields that contain the value zero.
DATEFNS	Activates year 2000-compliant versions of date subroutines.
DATETIME	Sets date and time in reports and controls the format in which CREATE SYNONYM creates date-time columns in a Master File.
DEFCENT	Defines a default century for your application.
EXL2KTXTDATE	Controls whether translated dates are sent as date values with format masks instead of text values.
FILECASE	Both UNIX and WebFOCUS support uppercase, lowercase, and mixed-case file names and directories. Since lowercase is the default in UNIX environments, WebFOCUS offers a way to automatically convert file names and directories to lowercase.
HDAY	Specifies the holiday file from which to retrieve dates that are considered holidays.
LEADZERO	Avoids the truncation of leading zeros.
TIME_SEPARATOR	Specifies the separator for time components for the &TOD variable.
TESTDATE	Temporarily alters the system date in order to test a dynamic window.
YRTHRESH	Defines the start of a 100-year window.

WebFOCUS-Specific Tasks

The following parameters affect the behavior of web-specific tasks in WebFOCUS.

&&APP_PERMIT	Enables you to create and update applications and files under the APPROOT directory.
BASEURL	Specifies a default location where your browser searches for relative URLs referenced in the HTML documents created by WebFOCUS.
FOCEXURL	Runs and executes drill downs remotely.
JSURLS	Includes JavaScript or VBScript files in an HTML report.
WEBVIEWALLPG	Controls whether the WebFOCUS Viewer displays an All Pages button.
WEBVIEWCLOSE	Controls whether the WebFOCUS Viewer displays a Close button.
WEBVIEWCLMSG	Controls whether the WebFOCUS Viewer Close message displays when the Close option is clicked
WEBVIEWER	In WebFOCUS, enables on-demand paging, and invokes the WebFOCUS Viewer.
WEBVIEWHELP	Controls whether the WebFOCUS Viewer displays a Help button.
WEBVIEWHOME	Allows an HTML page to be displayed when the WebFOCUS Viewer is closed.
WEBVIEWTARG	Allows the user to open the WebFOCUS Viewer in a target frame.
WEBVIEWTITLE	Defines the title text for the WebFOCUS Viewer window.

Graph Tasks

The following parameters control the processing and display of graphs.

3D	Sets a chart to either three dimensions or two dimensions.
ARGRAPHENGIN	Sets the graph engine used for rendering graphs in active technologies reports. it only has an effect when used with FORMAT AHTML or other active formats (APDF or FLEX).

AUTOFIT	Controls resizing of HTML5 graphs to fit their containers.
AUTOTICK	Sets the tick mark intervals for graphs.
BARNUMB	Places summary numbers at the end of bars on bar charts, or slices on pie charts.
BSTACK	Specifies whether bar chart bars are stacked or placed side by side.
EMBEDHEADING	Controls whether the heading or footing text is rendered within, or outside, the chart image.
GRAPHDEFAULT	Controls whether WebFOCUS default settings or the graph API default settings are in effect.
GRAPHEDIT	As of WebFOCUS 8.0, this parameter has been deprecated. You can use InfoAssist to edit charts. For information, see the *WebFOCUS InfoAssist User's Manual*.
GRAPHENGINE	Determines the version of the WebFOCUS graph engine that is used.
GRAPHSERVURL	Creates a GIF file of the graph by sending an HTTP request to a servlet.
GRMERGE	Specifies whether to create multiple graphs or a single graph when a request has multiple sort fields.
GRMULTIGRAPH	Specifies how many sort fields are used to generate multiple graphs when GRMERGE is set to ADVANCED.
GRID	Draws a grid of parallel horizontal lines at the vertical class marks on the graph.
GRWIDTH	Places an output graph in an HTML table of a specified width, in cells.
GTREND	Specifies the use of basic linear regression to alter the X and Y axis values in a SCATTER graph.
HAUTO	Performs automatic scaling of the horizontal axis for the given values.
HAXIS	Specifies the width, in characters, of the horizontal axis.
HCLASS	Specifies the horizontal interval mark when AUTOTICK is OFF.

HISTOGRAM	Draws a histogram instead of a curve when the values on the horizontal axis are not numeric.
HMAX	Sets the maximum value on the horizontal axis when automatic scaling is not used (HAUTO=OFF).
HMIN	Sets the minimum value on the horizontal axis when automatic scaling is not used (HAUTO=OFF).
HSTACK	Stacks the bars on a histogram instead of placing them side by side.
HTICK	Sets the horizontal axis interval mark when AUTOTICK is OFF.
LOOKGRAPH	Specifies a graph style.
OLAPGRMERGE	When a graph is rendered as OLAP, generates a three-dimensional graph that reflects all output data from a request containing both a BY and an ACROSS phrase.
VAUTO	Performs automatic scaling of the vertical axis for the given values.
VAXIS	Specifies the length of the vertical axis, in lines.
VCLASS	Specifies the vertical interval mark when AUTOTICK is OFF.
VGRID	Draws a grid at the horizontal and vertical class marks of the graph.
VMAX	Sets the maximum value on the vertical axis when automatic scaling is not used (VAUTO=OFF).
VMIN	Sets the minimum value on the vertical axis when automatic scaling is not used (VAUTO=OFF).
VTICK	Sets the vertical axis interval mark when AUTOTICK is OFF.
VZERO	Treats missing values on the vertical axis as zeros.

Memory Setup and Optimization Tasks

The following parameters control the memory and optimization of your application.

AUTOINDEX	Retrieves data faster by automatically taking advantage of indexed fields in a FOCUS data source.
AUTOPATH	Dynamically selects an optimal retrieval path.
AUTOSTRATEGY	Determines when WebFOCUS stops the search for a key field specified in a WHERE or IF test.
BINS	Specifies the number of pages of memory used for data source buffers.
CACHE	Stores FOCUS data source pages in memory and buffers between the data source and BINS.
COMPUTE	Controls the compilation of expressions.
DEFINES	Compiles virtual fields into machine code to improve performance.
DMH_LOOPLIM	Controls the number of loop iterations allowed in Dialogue Manager.
DMH_STACKLIM	Controls the number of lines allowed in FOCSTACK.
ESTRECORDS	Passes the estimated number of records to be sorted in the request.
FIXRET[RIEVE]	Enables keyed retrieval from a fixed format sequential file, such as a HOLD file.
FOCSTACK	Specifies the amount of space, in thousands of bytes, used by FOCUS commands waiting for execution.
RECORDLIMIT	Limits the number of records retrieved.
SQLTOPTTF	Enables the SQL Translator to generate TABLEF commands instead of TABLE commands.
TEMP	In WebFOCUS, assigns temporary files to a specific directory.

Report Code, Content, and Processing Tasks

The following parameters affect the content or processing of a report.

ALL	Handles missing segment instances in a report.
ALLOWCVTERR	Controls the display of a row of data that contains an invalid date format.
AREXPIRE	Specifies the date when an HTML active report expires.
ARPASSWORD	Specifies the password required to view HTML active report output.
ASNAMES	Controls the FIELDNAME attribute in a HOLD Master File.
AUTOTABLEF	Avoids creating the internal matrix based on the features used in the query.
BUSDAYS	Specifies which days are considered business days.
CARTESIAN	Generates a report containing all combinations of non-related data instances in a multi-path request containing a PRINT or LIST command.
CDN	Specifies punctuation used in numeric notation.
CENT-ZERO	Displays a leading zero in decimal-only numbers.
COLLATION	Controls ordering of alphanumeric values.
COMPMISS	Controls whether the missing attribute is propagated to reformatted fields in a report request.
COMPUTE	Controls the compile of expressions.
DATEDISPLAY	Controls the display of date format fields that contain the value zero.
DATEFNS	Activates year 2000-compliant versions of date subroutines.
DATETIME	Sets date and time in a report and controls the format in which CREATE SYNONYM creates date-time columns in a Master File.

DBAJOIN	Controls whether DBA restrictions are treated as report filters or are added to the join conditions.
DB_INFILE	Controls whether the expression generated by the DB_INFILE function for use against a relational data source is optimized.
DEFCENT	Defines a default century for your application.
DEFECHO	Defines a default value for the &ECHO variable for your application.
EQTEST	Controls the output generated when a report request retrieves zero records.
ERROROUT	Terminates a request and returns a message when an error is encountered.
ESTRECORDS	Passes the estimated number of records to be sorted in the request.
EXCELSERVURL	Specifies the application server to be used to zip the file components that comprise an EXCEL 2007 file (.xlsx).
EXL2KTXTDATE	Specifies the language used for Microsoft® Excel requests. This language must be the same as the language of Excel on the browser machine.
EXTAGGR	Enables aggregation in an external sort.
EXTHOLD	Enables you to use an external sort to create HOLD files.
EXTRACT	Activates Structured HOLD Files for a request.
EXTSORT	Activates the external sorting feature.
FIELDNAME	Controls the use of long and qualified field names.
FILECOMPRESS	The SET FILECOMPRESS=ON command compresses PDF output files. The compressed PDF file is 25% of the original PDF file size.
FILE[NAME]	Specifies a file to be used, by default, in commands.
FILTER	Activates declared filters.

FOC144	Suppresses warning message FOC144, which reads *Warning Testing in Independent sets of Data*.
FORMULTIPLE	Allows you to include the same value of a FOR field in multiple rows of the FML matrix.
HNODATA	Controls missing values propagated to a HOLD file.
HOLDATTR[S]	Includes the TITLE and ACCEPT attributes from the original Master File in the HOLD Master File.
HLDCOM_TRIMANV	Controls whether strict equality is required or partial key joins are supported for record-oriented adapters. JOINLM is a synonym for JOIN_LENGTH_MODE.
JOINOPT	Ensures proper alignment of report output by correcting for lagging (missing) values. Also enables joins between fields with different numeric data types.
JSURLS	Includes JavaScript or VBScript files in an HTML report.
KEEPDEFINES	Controls whether a virtual field created for a host or joined structure is retained after a JOIN command is run.
LANG[UAGE]	The LANG[UAGE] parameter specifies the National Language Support (NLS) environment. It sets the language of server error messages and can also be used to set the language of report titles if the Master File Description contains alternate language TITLE attributes.
LEADZERO	Avoids the truncation of leading zeros.
MULTIPATH	Controls whether a parent segment is included in report output when selection tests are done on independent paths.
NODATA	Determines the character string that indicates missing data in a report.
ONFIELD	Controls whether ON phrases are ignored for fields not referenced in a request.

PDFLINETERM	Determines if an extra space is appended to each record of a PDF output file to facilitate proper file transfer between Windows and UNIX.
PHONETIC_ALGORITHM	Sets the phonetic algorithm to use with the PHONETIC function.
PPTXGRAPHTYPE	Enhances the quality of charts embedded into PowerPoint (PPTX) slides.
PRINTDST	Controls processing of reports that use the PRINT command in conjunction with multiple DST operators.
QUALCHAR	Specifies the qualifying character to be used in qualified field names.
SAVEMATRIX	Saves the matrix from your request to protect it from being overwritten when using Dialogue Manager commands.
SHORTPATH	Controls how tests against missing cross-referenced segment instances are processed in a left outer join.
SUMMARYLINES	Permits the combination of fields with and without prefix operators on summary lines in one request.
SUMPREFIX	Allows users to choose the answer set display order when using an external sort to perform aggregation of alphanumeric or smart date formats.
TITLES	Uses predefined column titles in the Master File as column titles in report output.
WARNING	Turns off warning messages.

Report Layout and Display Tasks

The following parameters affect the layout and display of a report.

ACCESSHTML	Generates HTML report output that is Section 508 compliant.

ACCESSIBLE	Generates report output in HTML code that is Section 508 compliant.
ACCESSPDF	Generates PDF report output that is Section 508 compliant.
ACROSSLINE	Controls underlining of column titles on report output. TITLELINE is a synonym.
ACROSSPRT	Reduces the number of report lines within each sort group when a request uses the PRINT command and an ACROSS phrase.
ACROSSTITLE	Controls whether ACROSS titles display above or to the left of ACROSS values.
ACRSVRBTITL	Controls the display of ACROSS column titles when there is only one displayed field for an ACROSS group.
AUTODRILL	Enables you to automatically drill down to the next level of a dimension within the body of a report.
AUTOFIT	Controls resizing of HTML report output to fit its window.
BASEURL	Specifies a default location where your browser searches for relative URLs referenced in the HTML documents created by WebFOCUS.
BLANKINDENT	Clarifies relationships within an FML hierarchy by indenting the captions (titles) of values at each level.
BOTTOMMARGIN	Sets the bottom boundary for report contents on a page in a styled report.
BYDISPLAY	Displays a sort field value on every row, column, or both in a report.
BYPANEL	Controls the repetition of BY fields on panels.
CENT-ZERO	Displays a leading zero in decimal-only numbers.
COMPOUND	Enables you to combine multiple reports into a single PDF or PS file to create a compound report.

CSSURL	Links an HTML report to an external cascading style sheet (CSS) file in order to style the report.
CURRENCY_DISPLAY	Defines the position of the currency symbol relative to the monetary number.
CURRENCY_ISO_CODE	Defines the ISO code for the currency symbol to use.
CURRENCY_PRINT_ISO	Defines what will happen when the currency symbol cannot be displayed by the code page in effect.
CURRSYMB	Sets a currency symbol to display on the report output when a numeric format specification uses the M or N display options.
CURSYM_D	Sets the characters to display on the report output when a numeric format specification uses the :D or :d display options.
CURSYM_E	Sets the characters to display on the report output when a numeric format specification uses the :E or :e display options.
CURSYM_F	Sets the characters to display on the report output when a numeric format specification uses the :F display option.
CURSYM_G	Sets the characters to display on the report output when a numeric format specification uses the :G display option.
CURSYM_L	Sets the characters to display on the report output when a numeric format specification uses the :L or :l display options.
CURSYM_Y	Sets the characters to display on the report output when a numeric format specification uses the :Y or :y display options.
DROPBLNKLINE	Eliminates blank lines from the report output.
DUPLICATECOL	Controls whether columns for multiple display commands are spread out or stacked on top of each other.
EXPANDABLE	Creates an Accordion report that expands by column.

EXPANDBYROW	Creates an Accordion report that expands by row.
EXTENDNUM	Prevents visual overflow on reports.
FOCFIRSTPAGE	Assigns a page number to the first page of output.
HIDENULLACRS	Hides ACROSS columns containing only null values.
HTMLCSS	Creates an inline Cascading Style Sheet command in the HTML page that displays the report output.
HTMLEMBEDIMG	Determines whether to embed images and graphs directly into an HTML or DHTML .htm file.
LANG[UAGE]	Specifies the National Language Support (NLS) environment. Sets the language of server error messages. Can also be used to set the language of report titles if the Master File Description contains alternate language TITLE attributes.
LAYOUTGRID	Displays a grid in the report output, which enables you to evaluate the correct placement of data and objects during your report design. This option is applicable only when using the PDF, PS, or DHTML report output.
LAYOUTRTL	Displays report output from right to left for supported formats.
LEFTMARGIN	Sets the left boundary for report contents on a page in a styled report.
LINES	Sets the maximum number of lines of printed output that appear on a page, from the heading at the top to the footing on the bottom. The OFFLINE-FMT parameter determines the format of printed report output generated from a request.
OFFLINE-FMT	The OFFLINE-FMT parameter determines the format of printed report output generated from a request.
ONLINE-FMT	Determines the format of report output. (Applies to WebFOCUS only.)
ORIENTATION	Specifies the page orientation for styled reports.

OVERFLOWCHAR	Changes the character that displays in a numeric report column when the column does not have enough space for the value.
PAGE[-NUM]	Controls the numbering of output pages.
PANEL	Sets the maximum line width of a report panel.
PRINTPLUS	Specifies printing enhancements.
POPUPDESC	Enables a pop-up field description in an HTML report when your mouse pointer is positioned over a column title in the report output.
PSPAGESETUP	Coordinates the paper source used by a PostScript printer with the PAGESIZE parameter setting.
QUALTITLES	Uses qualified column titles in report output when duplicate field names exist in a Master File.
RECAP-COUNT	Includes lines containing a value created with RECAP when counting the number of lines per page for printed output.
RIGHTMARGIN	Sets the right boundary for report contents on a page.
SHOWBLANKS	Preserves leading and internal blanks in HTML and EXL2K report output.
SPACES	Sets the number of spaces between columns in a report.
SQUEEZE	Determines the column width in report output.
STYLEMODE	For large report output, displays output in multiple HTML tables where each table is a separate report page.
STYLE[SHEET]	Controls the format of report output by accepting or rejecting StyleSheet parameters.
SUBTOTALS	Controls whether summary lines display above or below the data.

TARGETFRAME	Includes the HTML code <BASE TARGET="*framename*"> in the heading of the HTML file that is displayed in your browser.
TITLELINE	Controls underlining of column titles. ACROSSLINE is a synonym.
TOPMARGIN	Sets the top boundary on a page for report output.
UNITS	Specifies the unit of measure for page margins, column positions, and column widths.
VISBARORIENT	Specifies the orientation of visualization bars for ACROSS columns.

Security Tasks

The following parameters specify user access to data sources and procedures.

&&APP_PERMIT	Enables you to create and update applications and files under the APPROOT directory.
DBACSENSITIV	Controls whether password validation is case-sensitive.
DBASOURCE	Controls the source of access restrictions in a multi-file structure.
PASS	Enables user access to a data source or stored procedure protected by Information Builders security.
PERMPASS	The PERMPASS parameter establishes a user password that remains in effect throughout a session or connection.
USER	In WebFOCUS, enables user access to a data source or stored procedure protected by Information Builders security.

SET Parameter Syntax

This topic alphabetically lists the SET parameters that control the environment with a description and the syntax.

A	B	C	D
3D	*BARNUMB*	*CACHE*	*DATE_ORDER*
ACCBLN	*BASEURL*	*CARTESIAN*	*DATE_SEPARATOR*
ACCESSHTML	*BINS*	*CDN*	*DATEDISPLAY*
ACCESSIBLE	*BLANKINDENT*	*CENT-ZERO*	*DATEFNS*
ACCESSPDF	*BOTTOMMARGIN*	*COLLATION*	*DATEFORMAT*
ACROSSLINE	*BUSDAYS*	*CNOTATION*	*DATETIME*
ACROSSPRT	*BYDISPLAY*	*COMPMISS*	*DBAJOIN*
ACROSSTITLE	*BYPANEL*	*COMPOUND*	*DB_INFILE*
ACRSVRBTITL		*COMPUTE*	*DBACSENSITIV*
ALL		*COUNTWIDTH*	*DBASOURCE*
ALLOWCVTERR		*CSSURL*	*DEFCENT*
AREXPIRE		*CURRENCY_DISPLAY*	*DEFECHO*
ARPASSWORD		*CURRENCY_ISO_CODE*	*DEFINES*
ASNAMES		*CURRENCY_PRINT_ISO*	*DIRECTHOLD*
AUTODRILL		*CURRSYMB*	*DMH_LOOPLIM*
AUTOINDEX		*CURSYM_D*	*DMH_STACKLIM*
AUTOPATH		*CURSYM_E*	*DMPRECISION*
AUTOSTRATEGY		*CURSYM_F*	*DROPBLNKLINE*
AUTOTABLEF		*CURSYM_G*	*DTSTRICT*
ARGRAPHENGIN		*CURSYM_L*	*DUPLICATECOL*
AUTOFIT		*CURSYM_Y*	
AUTOTICK			

E	F	G	H
EMBEDHEADING	*FIELDNAME*	*GRAPHDEFAULT*	*HAUTO*
EMPTYREPORT	*FILECASE*	*GRAPHEDIT*	*HAXIS*
EQTEST	*FILECOMPRESS*	*GRAPHENGINE*	*HCLASS*
ERROROUT	*FILE[NAME]*	*GRAPHSERVURL*	*HDAY*
ESTRECORDS	*FILTER*	*GRID*	*HIDENULLACRS*
EUROFILE	*FIXRET[RIEVE]*	*GRMERGE*	*HISTOGRAM*
EXCELSERVURL	*FLOATMAPPING*	*GRMULTIGRAPH*	*HMAX*
EXL2KLANG	*FOC144*	*GRWIDTH*	*HMIN*
EXL2KTXTDATE	*FOCEXURL*	*GTREND*	*HLDCOM_TRIMANV*
EXPANDABLE	*FOCFIRSTPAGE*		*HNODATA*
EXPANDBYROW	*FOCHTMLURL*		*HOLDATTR[S]*
EXTAGGR	*FOCSTACK*		*HOLDFORMAT*
EXTHOLD	*FOCTRANSFORM*		*HOLDLIST*
EXTENDNUM	*FORMULTIPLE*		*HOLDMISS*
EXTRACT			*HOLDSTAT*
EXTSORT			*HSTACK*
			HTICK
			HTMLARCHIVE
			HTMLCSS
			HTMLEMBEDIMG
			HTMLENCODE

I	J	K	L
INDEX	*JOIN_LENGTH_MODE (JOINLM)* *JOINOPT* *JPEGENCODE* *JPEGQUALITY* *JSURLS*	*KEEPDEFINES* *KEEPFILTERS*	*LANG[UAGE]* *LAYOUTGRID* *LAYOUTRTL* *LEADZERO* *LEFTMARGIN* *LINES* *LOOKGRAPH*

M	N	O	P
MATCHCOLUMNORDER *MAXLRECL* *MAXDATAEXCPT* *MDICARDWARN* *MDIENCODING* *MDIPROGRESS* *MESSAGE* *MISSINGTEST* *MULTIPATH*	*NODATA* *NULL*	*OFFLINE-FMT* *OLAPGRMERGE* *OLDSTYRECLEN* *ONFIELD* *ONLINE-FMT* *ORIENTATION* *OVERFLOWCHAR*	*PAGE[-NUM]* *PAGESIZE* *PANEL* *PASS* *PCOMMA* *PDFLINETERM* *PERMPASS* *PHONETIC_ALGORITHM* *POPUPDESC* *PPTXGRAPHTYPE* *PRINTDST* *PRINTPLUS* *PSPAGESETUP*

Q	R	S	T
QUALCHAR	*RANK*	*SAVEDMASTERS*	*TARGETFRAME*
QUALTITLES	*RECAP-COUNT*	*SAVEMATRIX*	*TEMP*
	RECORDLIMIT	*SHADOW*	*TEMPERASE*
	RIGHTMARGIN	*SHIFT*	*TESTDATE*
	RPAGESET	*SHORTPATH*	*TIME_SEPARATOR*
		SHOWBLANKS	*TITLELINE*
		SPACES	*TITLES*
		SQLTOPTTF	*TOPMARGIN*
		SQUEEZE	
		%STRICTMATH	
		STYLEMODE	
		STYLE[SHEET]	
		SUBTOTALS	
		SUMMARYLINES	
		SUMPREFIX	

U	V	W	X-Y-Z
UNITS	*VAUTO*	*WARNING*	*XRETRIEVAL*
USER	*VAXIS*	*WEBARCHIVE*	*YRTHRESH*
USERFCHK	*VCLASS*	*WEBVIEWALLPG*	
USERFNS	*VGRID*	*WEBVIEWCLOSE*	
	VMAX	*WEBVIEWCLMSG*	
	VISBARORIENT	*WEBVIEWER*	
	VMIN	*WEBVIEWHELP*	
	VTICK	*WEBVIEWHOME*	
	VZERO	*WEBVIEWTARG*	
		WEBVIEWTITLE	
		WEEKFIRST	
		WPMINWIDTH	

3D

The 3D parameter sets a chart to either three dimensions or two dimensions.

The syntax is:

```
SET 3D = {ON|OFF}
```

where:

ON

Produces a three-dimensional chart. ON is the default.

OFF

Produces a two-dimensional chart.

ACCBLN

The ACCBLN parameter accepts blank or zero values for fields with ACCEPT commands in the Master File (see the *Describing Data With WebFOCUS Language* manual).

The syntax is:

```
SET ACCBLN = {ON|OFF}
```

where:

ON

Accepts blank and zero values for fields with ACCEPT commands unless blank or zero values are explicitly coded in the list of acceptable values. ON is the default value.

OFF

Does not accept blank and zero values for fields with ACCEPT commands unless blank or zero values are explicitly coded in the list of acceptable values.

ACCESSHTML

The ACCESSHTML parameter generates HTML report output that is compliant with Section 508-level accessibility requirements. The ACCESSHTML supersedes the ACCESSIBLE parameter (which is still supported) because it allows for future expansion to support additional accessibility standards.

HTML code that is Section 508 compliant automatically turns off pagination. Column titles and page headings are displayed only at the beginning of the report and page footings only at the end of the report.

The syntax is:

```
SET ACCESSHTML = {508|OFF}
```

where:

508

Generates HTML report output that is compliant with Section 508-level accessibility requirements.

OFF

Does not generate report output that is compliant with Section 508-level accessibility requirements. OFF is the default.

ACCESSIBLE

The ACCESSIBLE parameter generates report output in HTML code that is compliant with Section 508-level accessibility requirements.

HTML code that is Section 508 compliant automatically turns off pagination. Column titles and page headings are displayed only at the beginning of the report and page footings only at the end of the report.

The syntax is:

```
SET ACCESSIBLE = {508|OFF}
```

where:

508

Generates report output in HTML code that is compliant with Section 508-level accessibility requirements.

OFF

Does not generate report output that is compliant with Section 508-level accessibility requirements. OFF is the default.

ACCESSPDF

The ACCESSPDF parameter generates PDF report output that is compliant with Section 508-level accessibility requirements.

The syntax is:

```
SET ACCESSPDF = {508|OFF}
```

where:

508

Generates PDF report output that is compliant with Section 508-level accessibility requirements.

OFF

Does not generate report output that is compliant with Section 508-level accessibility requirements. OFF is the default.

ACROSSLINE

The ACROSSLINE parameter controls underlining of column titles on report output. TITLELINE is a synonym for ACROSSLINE.

The syntax is:

```
SET {ACROSSLINE|TITLELINE} = {ON|OFF|SKIP}
```

where:

ON

Underlines column titles on report output. ON is the default value.

OFF

Replaces the underline with a blank line.

SKIP

Specifies no underline and no blank line.

ACROSSPRT

The ACROSSPRT parameter reduces the number of report lines within each in a request that uses the PRINT command and an ACROSS phrase.

The PRINT command generates a report that has a single line for each record retrieved from the data source after screening out those that fail IF or WHERE tests. When PRINT is used in conjunction with an ACROSS phrase, many of the generated columns may be empty. Those columns display the missing data symbol.

To avoid printing such a sparse report, you can use the SET ACROSSPRT command to compress the lines in the report. The number of lines is reduced within each sort group by swapping non-missing values from lower lines with missing values from higher lines, and then eliminating any lines whose columns all have missing values.

Because data may be moved to different report lines, row-based calculations, such as ROW-TOTAL and ACROSS-TOTAL in a compressed report are different from those in a non-compressed report. Column calculations are not affected by compressing the report lines.

The syntax is:

```
SET ACROSSPRT = {NORMAL|COMPRESS}
```

where:

NORMAL

Does not compress report lines.

COMPRESS

Compresses report lines by promoting data values up within a sort group.

ACROSSTITLE

In a report that uses the ACROSS sort phrase to sort values horizontally across the page, by default, two lines are generated on the report output for the ACROSS columns. The first line displays the name of the sort field (ACROSS title), and the second line displays the values for that sort field (ACROSS value). The ACROSS field name is left justified above the first ACROSS value.

The ACROSSTITLE parameter enables you to display both the ACROSS title and the ACROSS values on one line in PDF, HTML, EXL2K, or EXL07 report output. You can issue the SET ACROSSTITLE = SIDE command. This command places ACROSS titles to the left of the ACROSS values. The titles are right justified in the space above the BY field titles. The heading line that is created, by default, to display the ACROSS title will not be generated.

This feature is designed for use in requests that have both ACROSS fields and BY fields. For requests with ACROSS fields but no BY fields, the set command is ignored, and the ACROSS titles are not moved.

The syntax is:

```
SET ACROSSTITLE = {ABOVE|SIDE}
```

where:

ABOVE

Displays ACROSS titles above their ACROSS values. ABOVE is the default value.

SIDE

Displays ACROSS titles to the left of their ACROSS values, above the BY columns.

ACRSVRBTITL

Using the SET ACRSVRBTITL command, you can control the display of an ACROSS column title in an ACROSS group. The behavior of the title is determined by the number of verb columns in the ACROSS group. The field count is affected by the following features, which add internal matrix columns to the report:

- Fields in a heading or footing.
- Fields whose display is suppressed with the NOPRINT phrase.
- Reformatted fields (which are normally counted twice).
- A COMPUTE command referencing multiple fields.

The syntax is:

```
SET ACRSVRBTITL = {HIDEONE|ON|OFF}
```

```
ON TABLE SET ACRSVRBTITL {HIDEONE|ON|OFF}
```

where:

HIDEONE

Suppresses the title when there is only one display field, or there is only one display field and the request contains one or more of the features that add internal matrix columns to the report. This value is the default.

ON

Always displays the title even if there is only one display field.

OFF

Suppresses the title when there is only one display field. Displays the title when there is only one display field and the request contains one or more of the features that add internal matrix columns to the report. This is legacy behavior.

ALL

The ALL parameter handles missing segment instances in a report.

The command SET ALL = ON specifies a left outer join. With a left outer join, all records from the host file display on the report output. If a cross-referenced segment instance does not exist for a host segment instance, the report output displays missing values for the fields from the cross-referenced segment.

If there is a screening condition on the dependent segment, those dependent segment instances that do not satisfy the screening condition are omitted from the report output, and so are their corresponding host segment instances.

The syntax is:

```
SET ALL = {ON|OFF|PASS}
```

where:

ON

Includes missing segment instances in a report when fields in the segment are not screened by WHERE or IF criteria in the request. The missing field values are denoted by the NODATA character, set with the NODATA parameter (for more information, see *NODATA* on page 609).

OFF

Omits missing segment instances from a report. OFF is the default value.

PASS

Includes missing segment instances in a report, regardless of WHERE or IF criteria in the request. This option is not supported when MULTIPATH = COMPOUND (see *MULTIPATH* on page 608).

ALLOWCVTERR

The ALLOWCVTERR parameter applies to non-FOCUS data sources when converting from the way the date is stored (ACTUAL attribute) to the way it is formatted (FORMAT or USAGE attribute).

It controls the display of a row of data that contains an invalid date format (formerly called a smart date). When it is set to ON, the invalid date format is returned as the base date or a blank, depending on the settings for the MISSING and DATEDISPLAY parameters.

Note: The ALLOWCVTERR parameter is not supported for virtual fields.

The syntax is:

```
SET ALLOWCVTERR = {ON|OFF}
```

where:

ON

Displays a row of data that contains an invalid date format. When ALLOWCVTERR is set to ON, the display of invalid dates is determined by the settings of the MISSING attribute and DATEDISPLAY command.

The results are explained in the following table:

DATEDISPLAY	MISSING	RESULT
OFF	OFF	A blank is returned.
	ON	The value of the NODATA character (a period, by default) is returned. (See *NODATA* on page 609).
ON	OFF	The base date is returned (December 31, 1900, for dates with YMD or YYMD format; or January 1901, for dates with YM, YYM, YQ, or YYQ format).
	ON	The value of the NODATA character (a period, by default) is returned.

OFF

Does not display a row of data that contains an invalid date format and generates an error message. OFF is the default value.

AREXPIRE

The AREXPIRE parameter enables you to set the date when an HTML active report expires and the report output can no longer be displayed.

The use of AREXPIRE is optional and by default, there is no expiration date set for an HTML active report. To reset the expiration date, rerun the report to create new output which has no expiration date.

The syntax is:

```
SET AREXPIRE = {yymmdd|xxxDAYS}
```

where:

yymmdd

Is the expiration date in the format of year, month, and day. For example, if you want the report to expire on January 1, 2009, use 090101.

xxx

Is the number of days from the current date that the report expires. Valid values are 1 to 999.

The command can also be issued from within a request using:

```
ON TABLE SET AREXPIRE {yymmdd|xxxDAYS}
```

ARGRAPHENGIN

Sets the graph engine used for rendering graphs in active technologies reports. it only has an effect when used with FORMAT AHTML or other active formats (APDF or FLEX)..

The syntax is:

```
SET ARGRAPHENGIN = JSCHART
```

ARPASSWORD

The ARPASSWORD parameter enables you to set a password that is required to view HTML active report output. Prior to opening the report output, the user is prompted to enter a password to unlock the report.

The use of ARPASSWORD is optional and by default, there is no password required to access an HTML active report.

The syntax is:

```
SET ARPASSWORD = password
```

where:

password

Is any character string up to 32 characters in length.

The command can also be issued from within a request using:

```
ON TABLE SET ARPASSWORD password
```

ASNAMES

The ASNAMES parameter controls the FIELDNAME attribute in a HOLD Master File. When an AS phrase is used in a TABLE request, the specified literal is used as a field name in a HOLD file. It also controls how field names are specified for the values of an ACROSS field when a HOLD file is created.

The syntax is:

```
SET ASNAMES = {ON|OFF|MIXED|FOCUS|FLIP}
```

where:

OFF

Does not use the literal specified in an AS phrase as a field name in HOLD files, and does not affect the way ACROSS fields are named.

ON

Uppercases the literal specified in an AS phrase and propagates it as the field name in the HOLD Master File. Creates names for ACROSS fields that consist of the AS name value concatenated to the beginning of the ACROSS field value and controls the way ACROSS fields are named in HOLD files of any format.

MIXED

Uses the literal specified in an AS phrase for the field name, retaining the case of the AS name, and creates names for ACROSS fields that consist of the AS name value concatenated to the beginning of the ACROSS field value.

FOCUS

Uses the literal specified in an AS phrase as the field name and controls the way ACROSS fields are named only in HOLD files in FOCUS format. FOCUS is the default value.

FLIP

Propagates the field names in the original Master File to the alias names in the HOLD Master File and the alias names in the original Master File to the field names in the HOLD Master File.

AUTODRILL

The AUTODRILL parameter applies to OLAP.

It enables the end user to automatically drill down to the next level of a dimension within the body of the current report. That is, the server automatically generates a hyperlink (the <A HREF tag) in the HTML code for every output data item described by the Master File WITHIN attribute. This occurs when AUTODRILL is set to ON.

If there is no other level, and the selected field is part of a BY command in the request (that is, it is a vertical sort field), the report is generated without a column for that dimension. If there is no other level, and the selected field is part of an ACROSS command in the request (it is a horizontal sort field), the report is generated without a row for that dimension. AUTODRILL must act on a sort field (that is also defined as a dimension in the Master File).

If the OLAP Control Panel is open, the output data item selected on the screen is added to the Selection Criteria list in the panel, enabling the end user to choose multiple items before running the drill-down report.

If the OLAP Control Panel is not open, the drill-down report runs automatically, based on the output data item selected on the screen.

The syntax is:

```
SET AUTODRILL = {ON|OFF|ALL}
```

where:

ON

Automatically generates a hyperlink in the HTML source code for every output data item described by the Master File WITHIN attribute. ON is the default.

OFF

Disables the automatic generation of hyperlinks.

ALL

Allows you to drill down on any value in the report by creating every value in the report as a hyperlink, including Measures. Drilling on a measure generates a more detailed report based on the lowest drill down (BY) field.

AUTOFIT

The AUTOFIT parameter automatically resizes HTML report output to fit its window or frame and HTML5 graphs to fit their containers.

The syntax is:

```
SET AUTOFIT = {OFF|ON|RESIZE}

ON {GRAPH|TABLE} SET AUTOFIT {OFF|ON|RESIZE}
```

where:

OFF

Respects the dimensions specified by the data and styles for TABLE or by the HAXIS and VAXIS parameters for HTML5 graphs.

ON

Always resizes HTML report output to fit its window or frame and HTML5 graph output to fit its container.

RESIZE

Applies to HTML5 graphs only. Respects the dimensions specified by the HAXIS and VAXIS parameters initially, but resizes the graph output if the container is resized.

AUTOINDEX

The AUTOINDEX parameter speeds data retrieval by automatically taking advantage of indexed fields in most cases where TABLE requests contain equality or range tests on those fields. This applies only to FOCUS data sources.

AUTOINDEX is never performed when the TABLE request contains an alternate file view, for example, TABLE FILE *filename.fieldname*. Indexed retrieval is not performed when the TABLE request contains BY HIGHEST or BY LOWEST phrases and AUTOINDEX is ON.

The syntax is:

```
SET AUTOINDEX = {ON|OFF}
```

where:

ON

Uses indexed retrieval when possible. ON is the default value.

OFF

Uses indexed retrieval only when explicitly specified using an indexed view, for example, TABLE FILE *filename.indexed-fieldname*.

AUTOPATH

The AUTOPATH parameter dynamically selects an optimal retrieval path for accessing a FOCUS data source by analyzing the data source structure and the fields referenced, and choosing the lowest possible segment as the entry point. Use AUTOPATH only if your field is not indexed.

The syntax is:

```
SET AUTOPATH = {ON|OFF}
```

where:

ON

Dynamically selects an optimal retrieval path. ON is the default value.

OFF

Uses sequential data retrieval. The end user controls the retrieval path through *filename.segname*.

AUTOSTRATEGY

The AUTOSTRATEGY parameter determines when FOCUS stops the search for a key field specified in a WHERE or IF test. When set to ON, the search ends when the key field is found, optimizing retrieval speed. When set to OFF, the search continues to the end of the data source.

The syntax is:

```
SET AUTOSTRATEGY = {ON|OFF}
```

where:

ON

Stops the search when a match is found. ON is the default value.

OFF

Searches the entire data source.

AUTOTABLEF

The AUTOTABLEF parameter avoids creating the internal matrix based on the features used in the query. Avoiding internal matrix creation reduces internal overhead costs and yields better performance.

The syntax is:

```
SET AUTOTABLEF = {ON|OFF}
```

where:

ON

Does not create an internal matrix. ON is the default value.

OFF

Creates an internal matrix.

AUTOTICK

The AUTOTICK parameter automatically sets the tick mark intervals for graphs. (See also *HTICK* on page 593 and *VTICK* on page 641.)

The syntax is:

```
SET AUTOTICK = {ON|OFF}
```

where:

ON

Automatically sets the tick mark intervals. ON is the default.

OFF

Does not set the tick mark intervals. You must set them with parameters including HTICK and VTICK.

BARNUMB

The BARNUMB parameter places summary values at the end of bars on bar charts, or slices on pie charts.

The syntax is:

```
SET BARNUMB = {ON|OFF}
```

where:

ON

Places the summary values at the ends of bars on bar charts, or slices on pie charts.

OFF

Does not include summary values. OFF is the default.

BASEURL

The BASEURL parameter specifies a default location where your browser searches for relative URLs referenced in the HTML documents created by WebFOCUS. This allows you to hyperlink to files, images, and Java files using only the file names rather than the full URLs.

The syntax is:

```
SET BASEURL = url
```

where:

url

Is the fully qualified directory in which additional HTML files, graphics files, and Java applet class files reside. If the URL represents a web server address, it must begin with http:// and end with a slash (/).

BINS

The BINS parameter specifies the number of pages of memory (blocks of 4,096 bytes) used for data source buffers.

The syntax is:

```
SET BINS = n
```

where:

n

Is the number of core pages used for data source buffers. Valid values are 13 to 64. The default value is two-thirds of the core remaining after you start, about 64 pages.

BLANKINDENT

To clarify relationships within an FML hierarchy, the captions (titles) of values are indented at each level. You can use the BLANKINDENT parameter in an HTML, PDF, or PostScript report to specify the indentation between each level the hierarchy. You can use the default indentation for each hierarchy level or choose your own indentation value. To print indented captions in an HTML report, you must set the BLANKINDENT parameter to ON or to a number.

In PDF and PS reports, you may need to adjust the widths of columns to accommodate the indentations.

The syntax is:

```
SET BLANKINDENT = {ON|OFF|n}
```

where:

`ON`

Indents FML hierarchy captions 0.125 units for each space normally displayed before the caption. For child levels in an FML hierarchy, it indents 0.125 units for each space that would normally display between this line and the line above it.

`OFF`

Turns off indentations for FML hierarchy captions in an HTML report. For other formats, uses the default indentation of two spaces. OFF is the default value.

n

Is an explicit measurement in the unit of measurement defined by the UNITS parameter. This measurement is multiplied by the number of spaces that would normally display before the caption. For child levels in an FML hierarchy, it indents *n* units for each space that would normally display between this line and the line above it. The default number of spaces is two. Zero (0) produces the same report output as OFF. Negative values for n are not supported. They generate the following message, and the request processes as if BLANKINDENT=OFF:

```
VALID VALUES ARE OFF, ON OR A POSITIVE NUMBER (IN CURRENT UNITS)
```

BOTTOMMARGIN

The BOTTOMMARGIN parameter sets the StyleSheet bottom boundary for report contents on a page.

This parameter applies only to PostScript and PS report formats.

The syntax is:

```
SET BOTTOMMARGIN = {n|.250}
```

where:

n

Is the bottom margin, in inches, for report contents on a page. 0.25 inches is the default value.

BUSDAYS

The BUSDAYS parameter specifies which days are considered business days and which days are not if, your business does not follow the traditional Monday through Friday week.

The syntax is:

```
SET BUSDAYS = {week|_MTWTF_}
```

where:

week

Is SMTWTFS, representing the days of the week. Any day that you do not want to designate as a business day must be replaced with an underscore in the designated place for that day.

If a letter is not in its correct position, or if you replace a letter with a character other than an underscore, you receive an error message. _MTWTF_ is the default value.

BYDISPLAY

Within a sort group, the sort field value displays only on the first line of the rows or the leftmost column of the columns for its sort group. However, you can display the appropriate BY or ACROSS field on every row or column in a report using the SET BYDISPLAY command. Although SET BYDISPLAY is supported for all output formats, it is especially important for making your report output more usable by Excel, which cannot sort columns properly when they have blank values in some rows.

This feature may enable you to avoid specifying the sort field twice, once as a display field and once for sorting (with the NOPRINT option).

The syntax is:

```
SET BYDISPLAY = {OFF|ON|BY|ACROSS|ALL}
```

where:

OFF

Displays a BY field value only on the first line or column of the report output for the sort group and on the first line or column of a page. OFF is the default value.

`ON or BY`

Displays the associated BY field value on every line of report output produced. BY is a synonym for ON.

`ACROSS`

Displays the relevant ACROSS field value on every column of report output produced.

`ALL`

Displays the relevant BY field value on every line of report output and the relevant ACROSS field value on every column of report output.

BYPANEL

The BYPANEL parameter enables display of By fields in the left portion of each panel of a multi-panel report.

The syntax is:

`SET BYPANEL = option`

where:

option

Is one of the following:

`ON` repeats the sort field values on each report panel.

`OFF` does not repeat sort field values on each report panel. Fields are displayed only on the first panel, and columns may split between panels. This value is the default.

`0` does not repeat sort field values on each report panel, and columns do not split between panels.

n repeats *n* columns of sort fields on each report panel. The value for *n* can be equal to or less than the total number of sort fields specified in the request.

CACHE

The CACHE parameter controls the number of cache pages to be allocated. This command cannot be used with ON TABLE SET.

Stores 4K FOCUS data source pages in memory and buffers them between the data source and BINS.

When a procedure calls for a read of a data source page, FOCUS first searches BINS, then cache memory, and then the data source on disk. If the page is found in cache, FOCUS does not have to perform an I/O to disk.

When a procedure calls for a write of a data source page, the page is written from BINS to disk. The updated page is also copied into cache memory so that the cache and disk versions remain the same. Unlike reads, cache memory does not save disk I/Os for write procedures.

FOCSORT pages are also written to cache. When the cache becomes full, they are written to disk. For optimal results, set cache to hold the entire data source plus the size of FOCSORT for the request. To estimate the size of FOCSORT for a given request, issue the ? STAT command, then add the number of SORTPAGES listed to the number of data source pages in memory. Issue a SET CACHE command for that amount. If cache is set to 50, 50 4K pages of contiguous storage are allocated to cache. The maximum number of cache pages can be set at installation. For more information, see Technical Memo 7838.1, *Setting the Maximum Number of Cache Pages.*

To clear the CACHE setting, issue a SET CACHE = *n* command. This command flushes the buffer (everything in cache memory is lost).

The syntax is:

```
SET CACHE = {0|n}
```

where:

`0`

Allocates no space to cache, which is inactive. 0 is the default value.

n

Is the number of 4K pages of contiguous storage allocated to cache memory. The minimum is two pages. The maximum is determined by the amount of memory available.

CARTESIAN

The CARTESIAN parameter applies to requests containing PRINT or LIST.

It generates a report containing all combinations of non-related data instances in a multi-path request. ACROSS cancels this parameter.

The syntax is:

```
SET CARTESIAN = {ON|OFF}
```

where:

`ON`

Generates a report with non-related records.

OFF

Disables the Cartesian product. OFF is the default value.

CDN

The CDN parameter specifies punctuation used in numeric notation.

Continental Decimal Notation (CDN) is supported for output in TABLE requests. It is not supported in DEFINE or COMPUTE commands.

The syntax is:

```
SET CDN = option
```

where:

option

Is one of the following:

DOTS_COMMA or ON uses CDN. Sets the decimal separator as a comma and the thousands separator as a period. For example, the number 3,045,000.76 is represented as 3.045.000,76. ON should be used for Germany, Denmark, Italy, Spain, and Brazil.

Note: Numeric parameters that use CDN ON must be separated by a comma followed by a space in calls to functions.

COMMAS_DOT or OFF turns CDN off. For example, the number 3,045,000.76 is represented as 3,045,000.76. OFF is the default value. OFF should be used for the USA, Canada, Mexico, and the United Kingdom.

SPACES_COMMA or SPACE sets the decimal point as a comma, and the thousands separator as a space. For example, the number 3,045,000.76 is represented as 3 045 000,76. SPACE should be used for France, Norway, Sweden, and Finland.

QUOTES_COMMA or QUOTE sets the decimal point as a comma and the thousands separator as an apostrophe. For example, the number 3,045,000.76 is represented as 3'045'000,76. QUOTE should be used for Switzerland.

QUOTES_DOT or QUOTEP sets the decimal point as a period and the thousands separator as an apostrophe. For example, the number 3,045,000.76 is represented as 3'045'000.76.

Note: If the display format of a report is Excel 2000 or later, Continental Decimal Notation is controlled by the settings on the computer. That is, numbers in report output are formatted according to the convention of the locale (location) set in regional or browser language options.

CENT-ZERO

The CENT-ZERO parameter displays a leading zero in decimal-only numbers. The setting of CDN determines whether a decimal point or comma is the decimal separator.

The syntax is:

```
SET CENT-ZERO = {ON|OFF}
```

where:

ON

Displays fractions with a leading zero. The fraction is preceded by either a decimal point or comma, depending on the CDN setting.

OFF

Does not display a leading zero. The fraction is preceded by either a decimal point or comma, depending on the CDN setting. OFF is the default value.

CNOTATION

Column notation assigns a sequential column number to each column in the internal matrix created for a report request. You can use column notation in COMPUTE and RECAP commands to refer to these columns in your request.

Because column numbers refer to columns in the internal matrix, they are assigned after retrieval and aggregation are completed. Columns not actually displayed on the report output may exist in the internal matrix. For example, calculated values used in the request generate one or more columns in the internal matrix. Fields with the NOPRINT option take up a column in the internal matrix, and a reformatted field generates an additional column for the reformatted value. Certain RECAP calculations, such as FORECAST or REGRESS generate multiple columns in the internal matrix.

BY fields are not assigned column numbers but, by default, every other column in the internal matrix is assigned a column number, which means that you have to account for all of the internally generated columns if you want to refer to the appropriate column value in your request. You can change this default column assignment behavior with the SET CNOTATION=PRINTONLY command, which assigns column numbers only to columns that display on the report output, or the SET CNOTATION=EXPLICIT command, which assigns column numbers to columns that are referenced in the request.

The syntax is:

```
SET CNOTATION={ALL|PRINTONLY|EXPLICIT}
```

where:

ALL

Assigns column reference numbers to every column in the internal matrix. ALL is the default value.

PRINTONLY

Assigns column reference numbers only to columns that display on the report output.

EXPLICIT

Assigns column reference numbers to all fields referenced in the request, whether it is displayed or not.

Note: This setting is not supported in an ON TABLE phrase.

COLLATION

The COLLATION parameter controls the ordering and matching of all language elements that involve comparison of two alphanumeric values.

The syntax is:

```
SET COLLATION = {BINARY|SRV_CI|SRV_CS|CODEPAGE}
```

where:

BINARY
Bases the collation sequence on binary values.

SRV_CI
Bases collation sequence on the LANGUAGE setting, and is case-insensitive.

SRV_CS
Bases collation sequence on the LANGUAGE setting, and is case-sensitive.

CODEPAGE
Bases collation sequence on the code page in effect, and is case-sensitive. CODEPAGE is the default value.

In most cases, CODEPAGE is the same as BINARY. The only differences are for Danish, Finnish, German, Norwegian, and Swedish in an EBCDIC environment.

COMPMISS

When a field is reformatted in a request (for example, SUM field/format), an internal COMPUTE field is created to contain the reformatted field value and displayed on the report output. If the original field has a missing value, that missing value can be propagated to the internal field by setting the COMPMISS parameter ON. If the missing value is not propagated to the internal field, it displays a zero (if it is numeric) or a blank (if it is alphanumeric). If the missing value is propagated to the internal field, it displays the missing data symbol on the report output.

The syntax is:

```
SET COMPMISS = {ON|OFF}
```

where:

ON

Propagates a missing value to a reformatted field. ON is the default value.

OFF

Displays a blank or zero for a reformatted field.

COMPOUND

The COMPOUND parameter, which is used to create compound reports, combines multiple reports into a single PDF or PostScript (PS) file. Using COMPOUND enables you to concatenate reports with styled report formats (PDF, HTML, active report, Power Point, Excel). You can also embed image files, including graphs saved as images, in a compound report.

For more information about creating compound reports, see the *Creating Reports With WebFOCUS Language* manual.

For a compound report that may contain different report types, the syntax is:

```
SET COMPOUND = {OPEN|CLOSE} [NOBREAK]
```

or

```
ON TABLE SET COMPOUND {OPEN|CLOSE}
```

Note that when you are using this syntax, you must also include the following code to identify the display format of each of the different reports to be concatenated:

```
ON TABLE {PCHOLD|HOLD|SAVE} [AS name] FORMAT formatname
```

If all of the reports in the compound set are of the same type, either PDF or PS, the syntax is:

```
ON TABLE {PCHOLD|HOLD|SAVE} [AS name] FORMAT {PDF|PS} {OPEN|CLOSE} [NOBREAK]
```

where:

name

Is the name of the generated file. The name is taken from the first request in the compound report. If no name is specified in the first report, the name HOLD is used.

formatname

Is the name of the styled report format. Valid formats include PDF, PS, HTML, AHTML, PPT, and EXL2K.

OPEN

Is specified with the first report, and begins the concatenation process. A report that contains the OPEN attribute must be in PDF or PS format.

CLOSE

Is specified with the last report, and ends the concatenation process.

NOBREAK

Is an optional phrase that suppresses page breaks. By default, each report is displayed on a separate page. You can use NOBREAK selectively in a request to control which reports are displayed on the same page.

Note:

- You can save or hold the output from a compound report.
- Compound reports cannot be nested.
- Multi-pane reports cannot be used in a compound report.

COMPUTE

The COMPUTE parameter controls the compilation of calculations when a request is executed.

The syntax is:

```
SET COMPUTE = {NATV|NEW|OLD}
```

where:

NATV

Compiles calculations that are referenced at run time using native arithmetic. NATV is the default value.

NEW

Compiles COMPUTE calculations when a request is parsed.

OLD

Does not compile calculations when a request is executed.

COUNTWIDTH

The COUNTWIDTH parameter expands the default format of COUNT fields from a five-byte integer to a nine-byte integer.

The syntax is:

```
SET {COUNTWIDTH|LISTWIDTH} = {ON|OFF}
```

where:

ON

Expands the default format of COUNT fields from a five-byte integer to a nine-byte integer.

OFF

Does not expand the default format of COUNT fields from a five-byte integer to a nine-byte integer. OFF is the default value.

CSSURL

The CSSURL parameter links an HTML report to an external cascading style sheet (CSS) file in order to style the report.

The syntax is:

```
SET CSSURL = link
```

where:

link

Is the URL location of the CSS file. This can be an absolute or relative link.

CURRENCY_DISPLAY

This parameter defines the position of the currency symbol relative to the monetary value.

The syntax is:

```
SET CURRENCY_DISPLAY = pos
```

where:

pos

Defines the position of the currency symbol relative to a number. The default value is *default*, which uses the position for the format and currency symbol in effect. Valid values are:

- **LEFT_FIXED.** The currency symbol is left-justified preceding the number.
- **LEFT_FIXED_SPACE.** The currency symbol is left-justified preceding the number, with at least one space between the symbol and the number.
- **LEFT_FLOAT.** The currency symbol precedes the number, with no space between them.
- **LEFT_FLOAT_SPACE.** The currency symbol precedes the number, with one space between them.
- **TRAILING.** The currency symbol follows the number, with no space between them.
- **TRAILING_SPACE.** The currency symbol follows the number, with one space between them.

CURRENCY_ISO_CODE

This parameter defines the ISO code for the currency symbol to use.

The syntax is:

```
SET CURRENCY_ISO_CODE = iso
```

where:

iso

Is a standard three-character currency code such as USD for US dollars or JPY for Japanese yen. The default value is *default*, which uses the currency code for the configured language code.

CURRENCY_PRINT_ISO

This parameter defines what will happen when the currency symbol cannot be displayed by the code page in effect.

The syntax is:

```
SET CURRENCY_PRINT_ISO = {DEFAULT|ALWAYS|NEVER}
```

where:

DEFAULT
: Replaces the currency symbol with its ISO code, when the symbol cannot be displayed by the code page in effect. This is the default value.

ALWAYS
: Always replaces the currency symbol with its ISO code.

NEVER
: Never replaces the currency symbol with its ISO code. If the currency symbol cannot be displayed by the code page in effect, it will not be printed at all.

Using a Unicode environment allows the printing of all currency symbols, otherwise this setting is needed.

CURRSYMB

The CURRSYMB parameter specifies a symbol used to represent currency when a numeric format specification uses the M or N display options. The default currency symbol depends on the code page being used.

The syntax is:

```
SET CURRSYMB = symbol
```

where:

symbol

Is any printable character or a supported currency code.

Note: In order to specify a dollar sign as the character, you must enclose it in single quotation marks (').

- USD or '$' specifies U.S. dollars.
- GBP specifies the British pound.
- JPY specifies the Japanese yen.
- EUR specifies the Euro.
- NIS specifies the Israeli new shekel.

CURSYM_D

The CURSYM_D parameter specifies the characters used to represent currency when a numeric format specification uses the !D, :D, !d, or :d display options which, by default, display a floating (D) or fixed (d) dollar sign to the left of the number.

The syntax is:

```
SET CURSYM_D = currsym
```

where:

currsym

Specifies up to four printable characters. If the characters include a blank space, they must be enclosed in single quotation marks.

CURSYM_E

The CURSYM_E parameter specifies the characters used to represent currency when a numeric format specification uses the !E, :E, !e, or :e display options which, by default, display a floating (E) or fixed (e) euro symbol to the left of the number.

The syntax is:

```
SET CURSYM_E = currsym
```

where:

currsym

Specifies up to four printable characters. If the characters include a blank space, they must be enclosed in single quotation marks.

CURSYM_F

The CURSYM_F parameter specifies the characters used to represent currency when a numeric format specification uses the !F or :F display option which, by default, places a floating euro symbol to the right of the number. This command supports adding a blank space between the number and the currency symbol.

The syntax is:

```
SET CURSYM_F = currsym
```

where:

currsym

Specifies up to four printable characters. If the characters include a blank space, they must be enclosed in single quotation marks.

CURSYM_G

The CURSYM_G parameter specifies the characters used to represent currency when a numeric format specification uses the !G or :G display option which, by default, places a floating dollar sign to the right of the number. This command supports adding a blank space between the number and the currency symbol.

The syntax is:

```
SET CURSYM_G= currsym
```

where:

currsym

Specifies up to four printable characters. If the characters include a blank space, they must be enclosed in single quotation marks.

CURSYM_L

The CURSYM_L parameter specifies the characters used to represent currency when a numeric format specification uses the !L, :L, !l, or :l display options which, by default, display a floating (L) or fixed (l) British pound symbol to the left of the number.

The syntax is:

```
SET CURSYM_L = currsym
```

where:

currsym

Specifies up to four printable characters.

CURSYM_Y

The CURSYM_Y parameter specifies the characters used to represent currency when a numeric format specification uses the !Y, :Y, !y, or :y display options which, by default, display a floating (Y) or fixed (y) Japanese yen or Chinese yuan symbol to the left of the number.

The syntax is:

```
SET CURSYM_Y = currsym
```

where:

currsym

Specifies up to four printable characters.

DATE_ORDER

This parameter defines the order of date components for display.

The syntax is:

```
SET DATE_ORDER = {DEFAULT|DMY|MDY|YMD}
```

where:

DEFAULT
Respects the original order of date components. This is the default value.

DMY
Displays all dates in day/month/year order.

MDY
Displays all dates in month/day/year order.

YMD
Displays all dates in year/month/day order.

Note:

- DATE_ORDER and DATE_SEPARATOR override the specified date order for all date and date-time displays unless they include a translation display option (T,Tr, t, or tr), in which case the specified order is produced. To limit the scope to a request, use the ON TABLE SET phrase.
- To use these settings with the Dialogue Manager system variables, (for example, &DATE, &TOD, &YMD, &DATEfmt, and &DATXfmt) append the suffix .DATE_LOCALE to the system variable. This allows system variables that are localized to coexist with non-localized system variables.

DATE_SEPARATOR

This parameter defines the separator for date components for display.

The syntax is:

```
SET DATE_SEPARATOR = separator
```

where:

separator
Can be one of the following values.

- **DEFAULT.** Respects the separator defined by the USAGE format of the field. This is the default value.

- **SLASH.** Uses a slash (/) to separate date components.
- **DASH.** Uses a dash (-) to separate date components.
- **BLANK.** Uses a blank to separate date components.
- **DOT.** Uses a dot (.) to separate date components.
- **NONE.** Does not separate date components.

Note:

- DATE_ORDER and DATE_SEPARATOR override the specified date order for all date and date-time displays unless they include a translation display option (T,Tr, t, or tr), in which case the specified order is produced. To limit the scope to a request, use the ON TABLE SET phrase.
- To use these settings with the Dialogue Manager system variables, (for example, &DATE, &TOD, &YMD, &DATEfmt, and &DATXfmt) append the suffix .DATE_LOCALE to the system variable. This allows system variables that are localized to coexist with non-localized system variables.

DATEDISPLAY

The DATEDISPLAY parameter controls the display of date format fields that contain the value zero. When set to on, the base date is displayed for those fields. By default, the value zero in a date format field, such as YYMD, is returned as a blank.

The syntax is:

```
SET DATEDISPLAY = {ON|OFF|COMP}
```

where:

`ON`

Returns the base date for a date format field containing the value zero.

`OFF`

Returns a blank for a date format field containing the value zero. OFF is the default.

`COMP`

Returns a z/OS compatible blank for a date format field containing the value zero. The COMP value is required to return a blank in portable code.

DATEFNS

The DATEFNS parameter activates year 2000-compliant versions of date functions.

The syntax is:

```
SET DATEFNS = {ON|OFF}
```

where:

ON
: Activates the date functions that support year-2000 dates. ON is the default value.

OFF
: Deactivates the date functions that support year-2000 dates.

DATEFORMAT

The DATEFORMAT parameter specifies the order of the date components (month/day/year) when date-time values are entered in the formatted string and translated string formats. It makes the input format of a value independent of the format of the variable to which it is being assigned.

The syntax is:

```
SET DATEFORMAT = datefmt
```

where:

datefmt
: Can be one of the following: MDY, DMY, YMD, or MYD. MDY is the default value for the U.S. English format.

DATETIME

The DATETIME parameter sets time and date in reports and controls the format in which CREATE SYNONYM creates date-time columns in a Master File.

The syntax is:

```
SET DATETIME = option
```

where:

option
: Is one of the following:

- **STARTUP,** which is the time and date when you began your session. STARTUP is the default value.

- **CURRENT|NOW,** which changes each time it is interrogated. For example, if your batch job starts before midnight at 11:59 P.M., it will not complete until the next day. If DATETIME is set to NOW|CURRENT, any reference to the variable gives the current date, not the date when the job started.
- **RESET,** which freezes the date and time of the current run for the rest of the session or until another SET DATETIME command is issued.

DBAJOIN

The DBAJOIN parameter controls where DBA restrictions are treated as WHERE conditions in the report request or are added as join conditions.

```
SET DBAJOIN = {OFF|ON}
```

where:

`OFF`

Treats DBA restrictions as WHERE filters in the report request. OFF is the default value.

`ON`

Treats DBA restrictions as join conditions.

DB_INFILE

The SET DB_INFILE command controls whether the expression generated by the DB_INFILE function for use against a relational data source is optimized.

The syntax is:

```
SET DB_INFILE = {DEFAULT|EXPAND_ALWAYS|EXPAND_NEVER}
```

where:

`DEFAULT`

Enables DB_INFILE to create a subquery if its analysis determines that it is possible. This is the default value.

`EXPAND_ALWAYS`

Prevents DB_INFILE from creating a subquery and, instead, expands the expression into IF and WHERE clauses in memory.

EXPAND_NEVER

Prevents DB_INFILE from expanding the expression into IF and WHERE clauses in memory and, instead, attempts to create a subquery. If this is not possible, a FOC32585 message is generated and processing halts.

DBACSENSITIV

When a DBA or user issues the SET USER, SET PERMPASS or SET PASS command, this user ID is validated before they are given access to any data source whose Master File has DBA attributes. The password is also checked when encrypting or decrypting a FOCEXEC.

The SET DBACSENSITIV command determines whether the password is converted to uppercase prior to validation.

The syntax is:

```
SET DBACSENSITIV = {ON|OFF}
```

where:

ON

Does not convert passwords to uppercase. All comparisons between the password set by the user and the password in the Master File or FOCEXEC are case-sensitive.

OFF

Converts passwords to uppercase prior to validation. All comparisons between the password set by the user and the password in the Master File or FOCEXEC are not case-sensitive. OFF is the default value.

DBASOURCE

The DBASOURCE parameter determines which security attributes are used to grant access to multi-file structures. By default, access restrictions are based on the host file in a JOIN structure or the last file in a COMBINE structure. If you set the DBASOURCE parameter to ALL, access restrictions from all files in a JOIN or COMBINE structure will be enforced.

All files in the JOIN or COMBINE structure must have the same DBA password. If the DBA attributes are not the same, there will be no way to access the structure.

The SET DBASOURCE command can only be issued one time in a session or connection. Any attempt to issue the command additional times will be ignored. If the value is set in a profile, no user can change it at any point in the session.

When DBASOURCE=ALL:

- In a TABLE request against a JOIN structure, access to a cross-reference file or segment is allowed only if the user has at least read access to each file in the structure.
- In a MODIFY COMBINE structure, the user must have a minimum of READ access to all files in the structure. The user requires WRITE, UPDATE, or READ/WRITE access to a file in the structure when an INCLUDE, DELETE, or UPDATE request is issued against that file.

When DBASOURCE=HOST:

- In a TABLE request, the user needs read access to the host file in the JOIN structure. All security limitations come from the host file. Note that you can create and activate a DBAFILE in order to enforce security restrictions from all files in the structure.
- In a MODIFY procedure, the user needs write access to the last file in the COMBINE structure. All security limitations come from the restrictions in the last file in the structure. Note that you can create and activate a DBAFILE in order to enforce security restrictions from all files in the structure.

The syntax is:

```
SET DBASOURCE = {HOST|ALL}
```

where:

HOST

Enforces access restrictions only from the host file in a JOIN structure or the last file in a COMBINE structure unless a DBAFILE is used to enforce access restrictions to other files in the structure. HOST is the default value.

ALL

Requires the user to have read access to every file in a JOIN or COMBINE structure. The user needs W, U, or RW access to a file in a COMBINE structure when an INCLUDE, UPDATE, or DELETE command is issued against that file.

DEFCENT

The DEFCENT parameter defines a default century globally or on a field-level for an application that does not contain an explicit century. DEFCENT is used in conjunction with YRTHRESH to interpret the current century according to the given values. When assigned globally, the time span created by these parameters applies to every 2-digit year used by the application unless you specify file-level or field-level values. (See *YRTHRESH* on page 649.)

Note: This same result can be achieved by including the FDEFCENT and FYRTHRESH attributes in the Master File.

The syntax is:

```
SET DEFCENT = {cc|19}
```

where:

cc

Is the default century. 19 is the default value if one is not supplied. The value *cc* defaults to 19, for the twentieth century.

DEFECHO

The DEFECHO parameter defines a default value for the &ECHO variable.

The syntax is:

```
SET DEFECHO = {OFF|ON|ALL|NONE}
```

where:

OFF

Establishes OFF as the default value for &ECHO. OFF is the default value.

ON

Establishes ON as the default value for &ECHO.

ALL

Establishes ALL as the default value for &ECHO.

NONE

Prevents procedure code from being displayed (echoed). Once the value of DEFECHO or &ECHO has been set to NONE, it cannot be changed during the session or connection.

DEFINES

The DEFINES parameter increases the speed of calculations in virtual fields by compiling virtual fields into machine code.

This parameter applies to Windows, z/OS, and UNIX operating systems only.

The syntax is:

```
SET DEFINES = {COMPILED|OLD}
```

where:

COMPILED

Implements expression compilation at request run time, compiling only those DEFINEs that are used in the request. COMPILED is the default value.

OLD

The value OLD has been deprecated and functions as COMPILED.

DIRECTHOLD

The DIRECTHOLD parameter creates a HOLD file in FOCUS format directly, without an internally generated MODIFY procedure and an intermediate sequential file.

This parameter applies to Windows and UNIX operating systems only.

The syntax is:

```
SET DIRECTHOLD = {ON|OFF}
```

where:

ON

Creates a FOCUS HOLD file directly without an intermediate sequential file and MODIFY procedure. ON is the default value and the only value supported. OFF is allowed for backward syntax compatibility, but it operates the same way as ON.

OFF

OFF is allowed for backward syntax compatibility, but it operates the same way as ON.

DMH_LOOPLIM

The DMH_LOOPLIM parameter sets the maximum number of Dialogue Manager loop iterations allowed, using -REPEAT or -GOTO commands.

The syntax is:

```
SET DMH_LOOPLIM = n
```

where:

n

Sets the maximum number of loop iterations allowed. The default value is zero (0), which does not limit the number of loop iterations.

DMH_LOOPLIM should be set high enough to run your existing reports and procedures without error for your entire session. It is recommended that if you set this parameter, you set it in a profile.

DMH_STACKLIM

The DMH_STACKLIM parameter sets the maximum number of lines allowed in FOCSTACK.

The syntax is:

```
SET DMH_STACKLIM = n
```

where:

n

Sets the maximum number of lines allowed in FOCSTACK. The default value is zero (0), which does not limit the number of stacked commands.

DMH_STACKLIM should be set high enough to run your existing reports and procedures without error for your entire session. It is recommended that if you set this parameter, you set it in a profile.

DMPRECISION

The DMPRECISION parameter specifies numeric precision in Dialogue Manager -SET commands.

Without this setting, results of numeric calculations are returned as integer numbers, although the calculations themselves employ double-precision arithmetic. To return a number with decimal precision without this setting, you have to enter the calculation as input into subroutine FTOA, where you can specify the number of decimal places returned.

The syntax is:

```
SET DMPRECISION = {OFF|n}
```

where:

OFF

Specifies truncation without rounding after the decimal point. OFF is the default value

n

Is a positive number from 0-9, indicating the point of rounding. Note that n=0 results in a rounded integer value.

DROPBLNKLINE

The DROPBLNKLINE parameter suppresses blank lines around subtotals, subheadings, and subfootings when formatting a report for output. In addition, certain data lines may be blank and appear as blank lines on the report output. You can eliminate these blank lines from the report output using the SET DROPBLNKLINE=ON command.

This setting does not apply to the following output formats: HOLD/PCHOLD/SAVE formats ALPHA, INTERNAL, BINARY, COM, COMT, COMMA, TAB, TABT, FIX, DFIX, all DBMS, VSAM, LOTUS, SYLK, DIF, FOCUS, and XFOCUS.

The syntax is:

```
SET DROPBLNKLINE = {OFF|ON|BODY|HEADING|ALL}
```

where:

`OFF`

Inserts system-generated blank lines as well as empty data lines. OFF is the default value.

`ON|BODY`

Removes system-generated blank lines within the body of the report, for example, before and after subheads. In addition, certain data lines that may be blank and appear as blank lines on the report output will be removed from the output. BODY is a synonym for ON.

`HEADING`

Removes the blank lines between headings and titles and between the report body and the footing. Works in positioned formats (PDF, PS, DHTML, PPT, PPTX) when a request has a border or backcolor StyleSheet attribute anywhere in the report.

`ALL`

Provides both the ON and HEADING behaviors.

DTSTRICT

The DTSTRICT parameter controls the use of strict processing. Strict processing checks date-time values when they are input by an end user, read from a transaction file, displayed, or returned by a subroutine to ensure that they represent a valid date and time. For example, a numeric month must be between 1 and 12, and the day must be within the number of days for the specified month.

The syntax is:

```
SET DTSTRICT = {ON|OFF}
```

where:

ON

Invokes strict processing. ON is the default value.

OFF

Does not invoke strict processing. Date-time components can have any value within the constraint of the number of decimal digits allowed in the field. For example, if the field is a two-digit month, the value can be 12 or 99, but not 115.

DUPLICATECOL

The DUPLICATECOL parameter reformats report requests that use multiple display commands, placing aggregated fields in the same column above the displayed field.

The syntax is:

```
SET DUPLICATECOL = {ON|OFF}
```

where:

ON

Displays the report with each field as a column. ON is the default value.

OFF

Displays the report with common fields as a row.

EMBEDHEADING

The EMBEDHEADING parameter controls whether the Heading and Footing text is rendered within or outside the chart image.

The syntax is:

```
SET EMBEDHEADING = {ON|OFF}
```

where:

ON

Renders the heading and footing text as part of the chart image.

OFF

Renders the heading and footing as text elements that are separate from the chart image. This is the default value.

EMPTYREPORT

The EMPTYREPORT parameter controls the output generated when a TABLE request retrieves zero records.

EMPTYREPORT is not supported with TABLEF or Excel. When a TABLEF or Excel request retrieves zero records, an empty report is generated.

Note: Using the IF TOTAL or WHERE TOTAL phrases when EMPTYREPORT is set to OFF may produce an empty report if there is no data that satisfies the TOTAL condition. This occurs because the test for report lines for EMPTYREPORT is applied before the TOTAL condition is applied.

The syntax is:

```
SET EMPTYREPORT={ANSI|ON|OFF}
```

where:

`ANSI`

Produces a single-line report and displays the missing data character or a zero if a COUNT is requested. In each case, &RECORDS will be 0, and &LINES will be 1.

If the SQL Translator is invoked, ANSI automatically replaces OFF as the default setting for EMPTYREPORT.

`ON`

Produces an empty report (column headings with no content). This was the default behavior in prior releases.

`OFF`

Produces no report output. OFF is the default value except for SQL Translator requests. When the SQL Translator is invoked, ANSI replaces OFF as the default setting for the EMPTYREPORT parameter, so the results are the same as for the ANSI setting.

The command can also be issued from within a request using:

```
ON TABLE SET EMPTYREPORT ANSI
```

EQTEST

The EQTEST parameter controls whether the characters $ and $* are treated as wildcard characters or normal characters in IF tests and WHERE tests that can be converted to one or more IF tests.

The syntax is:

```
SET EQTEST = {WILDCARD|EXACT}
```

where:

WILDCARD

Treats the $ and $* characters as wildcard characters. WILDCARD is the default value.

EXACT

Treats the $ and $* characters as normal characters, not wildcards, in IF tests and in WHERE tests that can be translated to IF tests.

ERROROUT

The ERROROUT parameter terminates a job and returns a message when an error is encountered in a request.

The syntax is:

```
SET ERROROUT = {ON|OFF|OVERRIDE/filename}
```

where:

ON

Terminates the procedure and returns a message to the client in the event of an error on the server. ON is the default.

OFF

Continues processing when an error is encountered.

OVERRIDE

Establishes an override file for running a batch application. Has the effect of setting ERROROUT=OFF and applying whatever changes are coded in the named override file.

filename

The named file contains specific override entries with the syntax:

```
Message number W|E.
```

For example: 200 E would set ERROROUT=ON and specify that message number 200 is an error rather than whatever the FOC200 flag previously indicated.

ESTRECORDS

The ESTRECORDS parameter passes the estimated number of records to be sorted in the request.

ESTRECORDS can only be set with the ON TABLE SET command within the TABLE, MATCH, or GRAPH request.

The syntax is:

```
ON TABLE SET ESTRECORDS n
```

where:

n

Is the estimated number of records to be sorted.

EUROFILE

The EUROFILE parameter activates the data source that contains information for the currency you want to convert. This setting can be changed during a session to access a different currency data source. This parameter cannot be issued in a report request.

Note: You cannot set any additional parameters on the same line as EUROFILE. FOCUS ignores any other parameters specified on the same line.

The syntax is:

```
SET EUROFILE = {ddname|OFF}
```

where:

ddname

Is the name of the Master File for the currency data source you want to use. The ddname must refer to a read-only data source accessible by FOCUS. There is no default value.

OFF

Deactivates the current currency data source and removes it from memory.

EXCELSERVURL

The EXCELSERVURL parameter specifies the application server to be used to zip the file components that comprise an EXCEL 2007 file (.xlsx). The machine opening the file must have Excel 2007 or Excel 2003 with the Microsoft Office conversion utility installed. For Excel 2003, you must have the XLSX file type defined and should change the *Save Report* redirection setting in the WebFOCUS Administration Console to *yes*. For additional information and instructions for changing your redirection settings, see the *WebFOCUS Security and Administration* manual.

The syntax is:

```
SET EXCELSERVURL = url
```

where:

url

Is the URL (up to 256 characters) of the context root of the WebFOCUS application on the application server where the WebFOCUS Client is installed.

For example,

```
http://hostname/ibi_apps//IBIEXCELSERVURL
```

In this example,

hostname

Is the name of the application server where the WebFOCUS Client is installed.

EXL2KLANG

When included in the nlscfg.err file, the EXL2KLANG parameter specifies the language used for Microsoft® Excel requests. This language must be the same as the language of Excel on the browser machine in order to correctly display output.

You can code the SET EXL2KLANG command in a profile or procedure to override the setting in the errors file.

For additional information on setting this width, see the *National Language Support for International Computing* manual.

The syntax is:

```
EXL2KLANG = {language|ENG}
```

where:

language

Is the Excel language. Valid values are:

- **ENG** for English. ENG is the default value.
- **FRE** for French.
- **GER** for German.
- **JPN** for Japanese.
- **KOR** for Korean.
- **SPA** for Spanish.

EXL2KTXTDATE

The EXL2KTXTDATE parameter allows you to specify that translated dates should be sent as date values with format masks instead of text values.

The syntax is:

```
SET EXL2KTXTDATE = {TEXT|VALUE}
```

where:

TEXT

Passes date values that contain text to Excel 2000 as formatted text. TEXT is the default value.

VALUE

Passes the types of translated date values that contain text and are supported Excel date formats to Excel 2000 as standard date values with text format masks applied.

EXPANDABLE

Generates an Accordion Report that expands by column.

The syntax is:

```
SET EXPANDABLE = {OFF|ON}
```

where:

OFF

Does not create an Accordion Report. OFF is the default value.

ON

Creates an Accordion report that expands by column.

EXPANDBYROW

Generates an Accordion Report that expands by row.

The syntax is:

```
SET EXPANDBYROW = {OFF|ON|ALL|n}
```

where:

OFF

Does not create an Accordion Report. OFF is the default value.

ON

Creates an Accordion Report which initially displays only the highest sort field level. To see rows on lower levels, click the plus sign next to one of the displayed sort field values.

ALL

Creates an Accordion Report in which all sort field levels are initially expanded. To roll up a sort field level, click the minus sign next to one of the sort field values on that level.

n

Creates an Accordion Report in which *n* sort field levels are initially expanded. To roll up an expanded sort field level, click the minus sign next to one of the sort field values on that level.

EXTAGGR

The EXTAGGR parameter uses external sorts to perform aggregation.

The syntax is:

```
SET EXTAGGR = {ON|OFF|NOFLOAT}
```

where:

ON

Allows aggregation by an external sort. ON is the default.

OFF

Does not allow aggregation by an external sort.

NOFLOAT

Allows aggregation if there are no floating data fields present.

EXTHOLD

The EXTHOLD parameter enables you to create a HOLD file using an external sort.

The syntax is:

```
SET EXTHOLD = {ON|OFF}
```

where:

ON

Creates HOLD files using an external sort. ON is the default value.

OFF

Does not create HOLD files using an external sort.

EXTENDNUM

The EXTENDNUM parameter controls whether asterisks (*) display on report output when the value to be displayed does not fit in the allotted space on report output or whether the report column is extended to display the number.

The syntax is:

```
SET EXTENDNUM = {ON|OFF|AUTO}
```

where:

ON

Displays all numbers in full, regardless of the USAGE format defined.

OFF

Displays asterisks when the value does not fit in the space allotted by the USAGE format. This is the legacy behavior.

AUTO

Applies an ON or OFF setting based on output format and SQUEEZE settings, as shown in the following table.

Format	SQUEEZE Setting	EXTENDNUM
PDF, PS, DHTML, PPT, PPTX	ON	ON
	OFF	OFF
HTML, EXL2K, XLSX	N/A	ON
BINARY, ALPHA	N/A	OFF
WP, other delimited formats	N/A	OFF

AUTO is the default value.

EXTRACT

The EXTRACT parameter activates Structured HOLD Files for a request.

This parameter is only supported in a TABLE or TABLEF request using an ON TABLE phrase.

The syntax is:

```
ON TABLE SET EXTRACT = {ON|*|OFF}
```

where:

ON

Activates Structured HOLD Files for this request and extracts all fields mentioned in the request.

*

Activates Structured HOLD Files for this request and indicates that a block of extract options follows. For example, you can exclude specific fields from the Structured HOLD File.

OFF

Deactivates Structured HOLD files for this request. OFF is the default value.

EXTSORT

The EXTSORT parameter activates an external sorting feature for use with the TABLE, MATCH, and GRAPH commands.

If the report can be processed entirely in memory, external sorting does not occur.

In order to determine if the report can be processed in memory, issue the ? STAT query after the TABLE, MATCH, or GRAPH command, and check the value of the SORT USED parameter.

When StyleSheets are being used, an external sort does not work.

The syntax is:

```
SET EXTSORT = {ON|OFF}
```

where:

ON

Enables the selective use of an external sorting product to sort report. ON is the default value.

OFF

Uses the internal sorting procedure to sort reports.

FIELDNAME

The FIELDNAME parameter controls whether long and qualified field names are supported.

This command cannot be used with ON TABLE SET.

The syntax is:

```
SET FIELDNAME = {NEW|NOTRUNC|OLD}
```

where:

NEW

Supports long and qualified field names. NEW is the default value.

NOTRUNC

Supports long and qualified field names, but not unique truncations.

OLD

Limits field names to 12 characters. Qualified field names are not supported.

FILECASE

Both UNIX and WebFOCUS support uppercase, lowercase, and mixed case file names and directories. Since lowercase is the default in UNIX environments, WebFOCUS offers a way to automatically convert file names and directories to lowercase.

The syntax is:

```
SET FILECASE = {ACTUAL|LOWER}
```

where:

ACTUAL

Retains file names in uppercase, lowercase, or mixed-case, exactly as they are entered. When this setting is used, WebFOCUS looks for file names exactly as they are entered (uppercase, lowercase, or mixed-case).

LOWER

Converts uppercase or mixed-case names to lowercase. When this setting is used, WebFOCUS looks for file names in lowercase.

FILECOMPRESS

The SET FILECOMPRESS=ON command compresses PDF output files.

The syntax is:

```
SET FILECOMPRESS = {ON|OFF}
```

where:

ON

Compresses PDF output files.

OFF

Does not compress PDF output files. OFF is the default value.

This command applies to PDF output only. It is ignored by all other output formats, such as HTML and Excel.

FILE[NAME]

The FILE[NAME] parameter specifies a file to be used, by default, in commands. When you set a default file name, you can use that file without specifying its name.

The syntax is:

```
SET FILE[NAME] = filename
```

where:

filename

Is a default file to be used in commands.

FILTER

The FILTER parameter assigns screening conditions to a data source for reporting purposes. It activates and deactivates filters.

The SET FILTER command is limited to one line. To activate more filters to fit on one line repeat the SET FILTER command.

The syntax is:

```
SET FILTER= {*|xx[yy zz]} IN {file|*} {ON|OFF}
```

where:

*

Denotes all declared filters. * is the default value.

xx, yy, zz

Are the names of filters as declared in the NAME = syntax of the FILTER FILE command.

file

Is the name of the data source you are assigning screening conditions to.

ON

Activates all (*) or specifically named filters for the data source or all data sources (*). The maximum number of filters you can activate for a data source is limited by the number of WHERE/IF phrases the filters contain, not to exceed the limit of WHERE/IF criteria in any single report request.

OFF

Deactivates (*) or specifically named filters for the data source or all data sources (*). OFF is the default value.

FIXRET[RIEVE]

The FIXRETRIEVE parameter enables keyed retrieval from a fixed format sequential file, such as a HOLD file. That is, the retrieval process stops when an equality or range test on a key holds true.

The syntax is:

```
SET FIXRET[RIEVE] = {OFF|ON}
```

where:

ON

Enables keyed retrieval. ON is the default.

OFF

Disables keyed retrieval.

FLOATMAPPING

SET FLOATMAPPING enables you to take advantage of decimal-based precision numbers available in DB2 and Oracle, and extends that functionality to all numeric processing for floating point numbers. With this processing, you gain both precision, including improved rounding, and enhanced performance.

The syntax is

```
SET FLOATMAPPING = {D|M|X}
```

where:

D

Uses the standard double-precision processing. This is the default value.

M

Uses a new internal format that provides decimal precision for double-precision floating point numbers up to 16 digits.

X

Uses a new internal format that provides decimal precision for double-precision floating point numbers up to 34 digits.

Note: If the field is passed to a HOLD file, the internal data types X or M data type will be propagated to the USAGE and ACTUAL formats in the HOLD Master File.

FOC144

The FOC144 parameter suppresses warning message FOC144, which reads:

```
"Warning: Testing in Independent sets of Data."
```

The syntax is:

```
SET FOC144 = {NEW|OLD}
```

where:

NEW

Displays the FOC144 warning message. NEW is the default value.

OLD

Suppresses the FOC144 warning message.

FOCEXURL

The FOCEXURL parameter runs and executes drill downs remotely. The drill down program can be on your local machine or on a remote machine.

The syntax is:

```
SET FOCEXURL = path
```

where:

path

Is the location of the WebFOCUS Servlet. The default path for Servlet is /ibi_apps/WFServlet.

FOCFIRSTPAGE

The FOCFIRSTPAGE parameter assigns a page number to the first page of output.

The syntax is:

```
SET FOCFIRSTPAGE = {n|1|&FOCNEXTPAGE}
```

where:

n

Is the number to be assigned to the first page of output. Valid values are integers with one to six characters. 1 is the default value.

&FOCNEXTPAGE

Is a variable whose value is determined by the last page number used by the last report. Its value is one more than that number.

FOCHTMLURL

The FOCHTMLURL parameter allows you to access resources with an alias other than /ibi_apps/ibi_html.

To generate an alias other than /ibi_apps/ibi_html on the WebFOCUS Reporting Server, use the SET FOCHTMLURL command to set the alias that will be generated instead of /ibi_apps/ibi_html.

This command will most likely be used in a server profile (EDASPROF.PRF) or in one of the WFS files (site.wfs) to establish a default setting for the installation.

The syntax is:

```
SET FOCHTMLURL=/my_html
```

where:

my_html

Is the alias other than /ibi_apps/ibi_html that is generated on the WebFOCUS Reporting Server. Can be defined with up to 260 characters.

FOCSTACK

This setting is no longer needed, but has been left in the product so that existing applications that still include it continue to work. The FOCSTACK parameter specified the amount of memory, in thousands of bytes, used by FOCSTACK, the stack of FOCUS commands awaiting execution.

This command cannot be used with ON TABLE SET.

The syntax is:

```
SET FOCSTACK [SIZE] = {n|8}
```

where:

n

Is the maximum amount, in thousands of bytes, that can be used by FOCSTACK. The maximum value depends on your region size.

8

Allows 8000 bytes to be used by FOCSTACK. 8 is the default value.

FOCTRANSFORM

The FOCTRANSFORM parameter translates field and segment names in a business view to their corresponding names in the original Master File for request processing.

This command cannot be used with ON TABLE SET.

The syntax is:

```
SET FOCTRANSFORM  = {OFF|ON}
```

where:

OFF

Does not convert field and segment references.

ON

Converts all the field and segment references to the corresponding entities in the real Master File. ON is the default value.

FORMULTIPLE

You can use the same value of a FOR field in many separate rows whether alone, as part of a range, or in a calculation by including the following syntax before or within an FML request:

The syntax is:

```
SET FORMULTIPLE = {ON|OFF}
```

where:

ON

Enables you to reference the same value of a FOR field in more than one row in an FML request.

With FORMULTIPLE set to ON, a value retrieved from the data source is included on every line in the report output for which it matches the tag references.

OFF

Does not enable you to include the same value in multiple rows. OFF is the default value.

With FORMULTIPLE set to OFF, multiple tags referenced in any of these ways (OR, TO, *) are evaluated first for an exact reference or for the end points of a range, then for a mask, and finally within a range. For example, if a value is specified as an exact reference and then as part of a range, the exact reference is displayed. Note that the result is unpredictable if a value fits into more than one row whose tags have the same priority (for example, an exact reference and the end point of a range.)

GRAPHDEFAULT

In WebFOCUS, the GRAPHDEFAULT parameter controls whether the WebFOCUS default graph styles (such as default colors and data label placement) or the graph API default graph styles are used for rendering the graph.

The syntax is:

```
SET GRAPHDEFAULT = {ON|OFF}
```

where:

ON

Uses WebFOCUS graph default styles. ON is the default value.

OFF

Uses the graph API default styles.

GRAPHEDIT

As of WebFOCUS 8.0, this parameter has been deprecated. You can use InfoAssist to edit charts. For information, see the *WebFOCUS InfoAssist User's Manual*.

The syntax is:

```
SET GRAPHEDIT = SERVER
```

where:

SERVER

Generates a server-side graph. A GIF graph is created on the web server and the image is returned to the client in your browser. SERVER is the default.

GRAPHENGINE

The GRAPHENGINE parameter determines which version of the Java-based WebFOCUS graph engine is used. The syntax is:

```
SET GRAPHENGINE = {GRAPH53|NEW}
```

where:

GRAPH53|NEW

Is the WebFOCUS graph engine in Version 5 Release 3 and higher. GRAPH53 is the default value. NEW is a synonym for GRAPH53.

The graph engine is configured with the IBIF_graphengine setting in the Client Settings Graph panel in the Configuration area of the WebFOCUS Administration Console. The graph engine can also be configured manually in the cgivars.wfs file, located in the *install_drive:*\ibi\WebFOCUS82\client\wfc\etc directory.

GRAPHSERVURL

The GRAPHSERVURL parameter creates an image file of the graph by sending an HTTP request to a servlet. The image file is saved on the WebFOCUS Reporting Server, where it can be used as input for an embedded image in a PDF report. It does not require running Java on the WebFOCUS Reporting Server, only on the remote machine that runs the servlet. This will work on any platform that has TCP/IP support, including z/OS.

For more details, see the *Creating Reports With WebFOCUS Language* manual.

The syntax is:

```
SET GRAPHSERVURL = url
```

where:

url

Is the URL (up to 256 characters) of the graph servlet that creates the GIF file.

For example,

```
http://hostname/ibi_apps/IBIGraphServlet
```

In this example,

hostname

Is the name of the host where the server resides.

GRID

The GRID parameter draws a grid on the graph at the horizontal and vertical class marks.

The syntax is:

```
SET GRID = {ON|OFF}
```

where:

ON

Draws a grid of lines at the horizontal and vertical class marks on the graph.

OFF

Does not draw a grid of lines at the horizontal and vertical class marks on the graph. OFF is the default.

GRMERGE

The GRMERGE parameter generates a three-dimensional graph that reflects all output data from a request containing multiple sort fields, or creates separate graphs that collectively reflect all output data. When GRMERGE is OFF, a request containing both a BY and an ACROSS phrase generates two graphs. When it is ON, one graph results.

The syntax is:

```
SET GRMERGE = {ON|OFF|ADVANCED}
```

where:

ON

Generates a single three-dimensional graph that reflects all output data. ON is the default.

OFF

Generates separate graphs that collectively reflect all output data.

ADVANCED

Turns on the advanced merge option. This option uses three parameters to determine how to merge the graphs:

- **GRMULTIGRAPH,** which specifies how many sort fields to use to create multiple graphs.
- **GRLEGEND,** which specifies how many sort fields to place on the graph legend.
- **GRXAXIS,** which specifies how many sort fields to display on the X-axis. GRXAXIS must be at least 1 in order to plot the graph. A value greater than one creates nested X-axes.

Note: The sum of the sort fields used by GRMULTIGRAPH, GRLEGEND, and GRXAXIS must equal the number of sort fields in the graph request.

GRMULTIGRAPH

The GRMULTIGRAPH parameter specifies how many sort fields to use to create multiple graphs when GRMERGE is set to ADVANCED.

The syntax is:

```
SET GRMULTIGRAPH = n
```

where:

n

Specifies how many sort fields (0 through 2) to use to break the output into multiple graphs. The outermost sort fields are used to separate the graphs. When *n* is greater than zero, this is similar to GRMERGE=OFF, but allows an additional sort field.

GRWIDTH

The GRWIDTH parameter applies to OLAP graphs and regular report requests. However, the parameter is not under user control in the OLAP Control Panel.

The syntax is:

```
SET GRWIDTH = {n|0}
```

where:

n

Is the width, in cells, of an HTML table in which a graph is displayed. Valid values are between 0 and 512. If the parameter is set to 0, a Java applet is generated displaying multiple graphs on one page, one beneath another. 0 is the default value.

GTREND

The GTREND parameter specifies the use of basic linear regression to alter the X and Y axis values in a SCATTER graph. Basic linear regression involves the average of the summation of X and Y axis values to determine a linear equation that expresses the trend of the scatter diagram.

The syntax is:

```
SET GTREND={ON|OFF}
```

where:

ON

Uses basic linear regression to alter the X and Y axis values.

OFF

Does not alter the X and Y axis values. OFF is the default.

HAUTO

The HAUTO parameter performs automatic scaling of the horizontal axis for the given values. If HAUTO is OFF, the horizontal axis must be scaled using the HMAX and HMIN parameters. (See HMAX and HMIN.)

The syntax is:

```
SET HAUTO = {ON|OFF}
```

where:

ON

Scales the horizontal axis automatically. ON is the default.

OFF

Does not scale the horizontal axis automatically. The end user must supply values for HMAX and HMIN.

HAXIS

The HAXIS parameter specifies the width, in characters, of the horizontal axis. This parameter applies to graphs generated offline. HAXIS is ignored for online displays since WebFOCUS automatically adjusts the width of the graph to the width of the page.

The syntax is:

```
SET HAXIS = {n|770}
```

where:

n

Is the width, in characters, of the horizontal axis. Acceptable values are integers between 20 and 770. 770 is the default value.

HCLASS

The HCLASS parameter specifies the horizontal interval mark when AUTOTICK is OFF. See also *HTICK* on page 593.

The syntax is:

```
SET HCLASS = {n|.00}
```

where:

n

Is a number specifying the horizontal interval mark. The default value is .00.

HDAY

The HDAY parameter specifies the holiday file from which to retrieve dates that are designated as holidays for use with the date functions DATEDIF, DATEMOV, DATECVT, and DATEADD. The file must be named HDAY, followed by two to four characters.

To clear the holiday file, use

```
SET HDAY = OFF
```

The syntax is:

```
SET HDAY = xxxx
```

where:

xxxx

Are the letters in the name of the holiday file, named HDAY*xxxx*. This string must be between two and four characters long.

The default is no setting for this parameter.

HIDENULLACRS

The HIDENULLACRS parameter hides the display of ACROSS groups containing only null columns.

Hiding null ACROSS columns is supported for all styled output formats except for the EXL2K PIVOT and EXL2K FORMULA options. It is not supported for Active Technologies.

The syntax is:

```
SET HIDENULLACRS = {ON|OFF}
```

where:

ON

Hides columns with missing data in ACROSS groups within a BY-generated page break.

OFF

Does not hide columns. OFF is the default value.

HISTOGRAM

The HISTOGRAM parameter draws a histogram instead of a curve when the values on the horizontal axis are not numeric.

The syntax is:

```
SET HISTOGRAM = {ON|OFF}
```

where:

ON

Draws a histogram instead of a curve when the values on the horizontal axis are not numeric. ON is the default.

OFF

Draws a curve when the values on the horizontal axis are not numeric.

HLDCOM_TRIMANV

The HLDCOM_TRIMANV parameter controls whether trailing blanks are retained in AnV fields in delimited output files.

The syntax is:

```
SET HLDCOM_TRIMANV = {OFF|ON}
```

where:

OFF

Retains trailing blanks in AnV fields when the output is held in a delimited format. OFF is the default value.

ON

Removes trailing blanks in AnV fields when the output is held in a delimited format.

HMAX

The HMAX parameter sets the maximum value on the horizontal axis when automatic scaling is not used (HAUTO=OFF).

The syntax is:

```
SET HMAX = {nnn|.00}
```

where:

nnn
Is the maximum value on the horizontal axis when HAUTO=OFF. The default value is .00.

HMIN

The HMIN parameter sets the minimum value on the horizontal axis when automatic scaling is not used (HAUTO=OFF).

The syntax is:

```
SET HMIN = {nnn|.00}
```

where:

nnn

Is the minimum value on the horizontal axis when HAUTO=OFF. The default value is .00.

HNODATA

The HNODATA parameter controls the missing data characters that are propagated to fields with the MISSING=ON attribute in HOLD FORMAT ALPHA files. Missing values in fields that do not have the MISSING=ON attribute are propagated to a HOLD file as blank (for alphanumeric fields) or zero (for numeric fields).

The syntax is:

```
SET HNODATA = {charstring|,$}
```

where:

charstring

Is a string of up to 12 characters propagated to a HOLD FORMAT ALPHA file for missing values in a field with the MISSING=ON attribute. A period (.) is the default value.

If the string is longer than the length of the field, the value stored in:

- An alphanumeric field is the leftmost character of the string.

- A numeric field is a blank string.

When an alphanumeric string other than the default value (the period) is used to populate a missing numeric field, a blank is inserted in the held field to prevent a format error when displaying the data. If you use the default HNODATA value, it is inserted in numeric fields. In this way, a request against the HOLD file can recognize missing data that was propagated to the HOLD file.

If a number with decimal places is specified for HNODATA and the field with missing data is integer, the value is rounded to a whole number and inserted. In a numeric field that supports decimal places, it is rounded and inserted with the correct number of decimal digits.

`,$`

Indicates that nothing should be placed in the field when there is missing data. This setting can be used to support null values in non-FOCUS data sources.

HOLDATTR[S]

The HOLDATTR[S] parameter controls whether the TITLE and ACCEPT attributes from the original Master File are used in the HOLD Master File. This setting does not affect the way fields are named in the HOLD Master File.

The syntax is:

```
SET HOLDATTR = {ON|OFF|FOCUS}
```

where:

`ON`

Includes the TITLE attribute from the original Master File in HOLD Master Files for HOLD files of any format. The ACCEPT attribute is included in the HOLD Master File when the HOLD file is in FOCUS format.

`OFF`

Does not include the TITLE or ACCEPT attributes from the original Master File in the HOLD Master File.

`FOCUS`

Includes the TITLE and ACCEPT attributes in HOLD Master Files when the HOLD file is in FOCUS format. FOCUS is the default value.

HOLDFORMAT

The HOLDFORMAT parameter determines the default format for HOLD files. This value can be overridden for an individual HOLD file by issuing the ON TABLE SET HOLD FORMAT command in a request.

The syntax is:

```
SET HOLDFORMAT = {BINARY|ALPHA}
```

where:

BINARY

Creates HOLD files in binary format. BINARY is the default value.

ALPHA

Creates HOLD files in ALPHA format.

HOLDLIST

The HOLDLIST parameter controls whether only displayed fields or all fields are included in the HOLD or PCHOLD file.

The syntax is:

```
SET HOLDLIST = {PRINTONLY|ALL|ALLKEYS|EXPLICIT}
```

where:

PRINTONLY

Includes only those fields in the HOLD or PCHOLD file that are specified in the report request.

ALL

Includes all fields referenced in a request in the HOLD or PCHOLD file, including both computed fields and fields referenced in a COMPUTE command. ALL is the default value. (OLD may be used as a synonym for ALL.)

Note: Vertical sort (BY) fields specified in the request with the NOPRINT option are not included in the HOLD file, even with SET HOLDLIST=ALL.

ALLKEYS

Includes all fields in the HOLD or PCHOLD file, including NOPRINTed BY fields.

EXPLICIT

Includes fields in the HOLD or PCHOLD file that are explicitly omitted from the report output using the NOPRINT option in the request, but does not include fields that are implicitly NOPRINTed. For example, if a field is reformatted in the request, two versions of the field exist, the one with the new format and the one with the original format, which is implicitly NOPRINTed.

HOLDMISS

The HOLDMISS parameter enables you to distinguish between missing data and default values of blank (for character data) or zero (for numeric data) in a HOLD file.

The syntax is:

```
SET HOLDMISS = {OFF|ON}
```

where:

OFF

Does not allow you to store missing data in a HOLD file. OFF is the default value.

ON

Enables you to store missing data in a HOLD file. When TABLE generates a default value for data not found, it generates missing values.

HOLDSTAT

The HOLDSTAT parameter includes comments and DBA information in HOLD and PCHOLD Master Files. This information can be from the HOLDSTAT ERRORS file supplied by Information Builders, or a file specified by the user.

The syntax is:

```
SET HOLDSTAT = {ON|OFF|name}
```

where:

ON

Derives comments and DBA information from the holdstat.mas or errors.mas file in UNIX and Windows. In z/OS, this information is derived from the member HOLDSTAT in the PDS allocated to the ddname MASTER or ERRORS.

OFF

Does not include information from the HOLDSTAT file in the HOLD or PCHOLD Master File. OFF is the default value.

name

Specifies a HOLDSTAT file, created by the end user, whose information is included in the HOLD or PCHOLD Master File.

HSTACK

The HSTACK parameter stacks the bars on a histogram instead of placing them side by side.

The syntax is:

```
SET HSTACK = {ON|OFF}
```

where:

ON

Stacks the bars on a histogram.

OFF

Places the bars on a histogram side by side. OFF is the default.

HTICK

The HTICK parameter sets the horizontal axis interval mark when AUTOTICK is OFF. See also *HCLASS* on page 586.

The syntax is:

```
SET HTICK = {nnn|.00}
```

where:

nnn

Is the horizontal axis interval mark when AUTOTICK=OFF. The default value is .00.

HTMLARCHIVE

The HTMLARCHIVE parameter packages HTML or DHTML reports together with image files into a single web archive document (.mht file). The only browser that supports this format for HTML is Internet Explorer.

The syntax is:

```
SET HTMLARCHIVE = {ON|OFF}
```

where:

ON

Packages HTML or DHTML reports together with image files into a single web archive document (.mht file).

OFF

Does not package multiple files into a single document. OFF is the default value.

HTMLCSS

The HTMLCSS parameter creates an internal Cascading Style Sheets command in the HTML display page.

The syntax is:

```
SET HTMLCSS = {ON|OFF}
```

where:

ON

Creates an internal CSS command in the HTML page that displays the report output.

OFF

Does not create an internal CSS command in the HTML page that displays the report output. OFF is the default value.

HTMLEMBEDIMG

The HTMLEMBEDIMG parameter activates an encoding mechanism that embeds images and graphs directly into an HTML or DHTML .htm file to ensure that all WebFOCUS reports can be accessed from any browser.

The syntax is:

```
SET HTMLEMBEDIMG =  {OFF|ON|AUTO}
```

where:

OFF

Does not affect the default behavior. If HTMLARCHIVE is set ON, .mht files are generated.

ON

Encodes images within the .htm file.

AUTO

Determines how to handle images based on the browser of the calling client. For clients in Internet Explorer, HTMLARCHIVE will be used to embed the images into an .mht file. For all other browsers, HTMLEMBEDIMG will encode the image information into an .htm file. If the displaying browser is unknown, as in reports generated using ReportCaster, AUTO will use the HTMLARCHIVE setting that is in effect.

HTMLENCODE

The HTMLENCODE parameter controls whether HTML tags are encoded when these tags are stored within the actual data or created using a DEFINE or COMPUTE command.

In a WebFOCUS report, HTMLENCODE=ON causes any text set in a string to be encrypted for transportation, and then decrypted to be displayed as written on a report. This is both for security and to ensure that special characters are displayed correctly.

The syntax is:

```
SET HTMLENCODE {ON|OFF}
```

where:

ON

Encodes the HTML output that is data. This setting disables the rendering of HTML tags within a browser when these tags are stored within the actual data or created using a DEFINE or COMPUTE command.

OFF

Disables HTML encoding. OFF is the default value.

Note: Because of the new format of the zipped XLSX files, native HTML symbols, such as a caret (<), cannot be supported as tag characters. For XLSX, unlike other output formats, HTMLENCODE defaults to ON. HTMLENCODE set to OFF will cause any data containing HTML tag characters to be omitted from the cell.

INDEX

The INDEX parameter determines the indexing scheme used for indexes. Indexes are fields specified with FIELDTYPE=I keywords in the Master Files. This parameter applies to indexes created in FOCUS data sources using early WebFOCUS releases.

The syntax is:

```
SET INDEX[TYPE] = {NEW|OLD}
```

where:

NEW

Creates a binary tree index. NEW is the default value.

OLD

Creates a hash index.

JOIN_LENGTH_MODE (JOINLM)

The JOIN_LENGTH_MODE (JOINLM) parameter controls processing of equality joined field pairs for the record oriented Adapters (such as VSAM, DFIX, and FIX). There are two supported modes of handling compatible but not identical joined fields:

- **SQL compliance.** The JOIN command processor assures strict value equality of joined fields. Detected truncation of significant characters during host to cross-referenced conversion raises a *target not found* condition. A shorter host field value is extended to the length of cross-referenced field with non-significant characters according to the data type.
- **FOCUS reporting.** The JOIN command processor assures partial value equality of joined fields.
  - When joining a shorter to a longer field, a search range is created to find all cross-referenced values that are prefixed with the host value (partial key join).
  - When joining a longer to a shorter field, the host value is unconditionally truncated to the cross-referenced field length.

The syntax is:

```
SET JOIN_LENGTH_MODE = {SQL|RANGE}
```

where:

SQL

Sets SQL compliant mode. Assures strict equality of host and cross-referenced fields. This is the default value.

RANGE

Sets FOCUS reporting mode. Supports partial key joins.

JOINOPT

The JOINOPT parameter has two functions:

- **Correcting for lagging values with a unique join.** If a parent segment has two or more unique child segments so that each has multiple children, the report may incorrectly display a missing value. The remainder of the child values may then be misaligned in the report. These misaligned values are called lagging values. The JOINOPT parameter ensures proper alignment of your output by correcting for lagging values.
- **Enabling joins with data type conversion.** You can join two or more data sources containing different numeric data types. For example, you can join a field with a short packed decimal format to a field with a long packed decimal format, or a field with an integer format to a field with a packed decimal format. This provides enormous flexibility for creating reports from joined data sources.

The syntax is:

```
SET JOINOPT = {NEW|GNTINT|OLD}
```

where:

`NEW`

Corrects lagging values when a parent segment has multiple unique children. Also, enables joins with data type conversion.

`GNTINT`

Corrects lagging values when a parent segment has multiple unique children. Also, enables joins with data type conversion.

`OLD`

Does not correct lagging values or support joins with data type conversion. This is the default value.

JPEGENCODE

The SET JPEGENCODE command allows you to select the compression methodology.

The syntax is:

```
SET JPEGENCODE=[FLATE|DCT]
```

where:

`FLATE`

Fixed quality defined

Is a compression method, introduced in WebFOCUS Release 7.7 Version 03, which allows images within PDF files to be individually compressed. This method is the default for WebFOCUS Release 7.7 Version 03 through WebFOCUS Release 7.7 Version 06.

`DCT`
Discrete Cosine Transform

Is a compression method, introduced in WebFOCUS Release 8.2 Version 01, which allows for the designation of the percentage of quality to retain. This method is the default in WebFOCUS Release 8.2 Version 01 and higher.

Note: In some images, quality loss is not noticeable when higher degrees of compression are applied. Generally, the higher the compression applied, the lower the image quality. Use SET JPEGQUALITY with the DCT setting to set the quality value.

JPEGQUALITY

The SET JPEGQUALITY command allows you to select image quality.

The syntax is:

```
SET JPEGQUALITY=n
```

where:

n
Is the percentage of quality used with the DCT setting. This value can be from 1-100. The default value is 100, where no quality is lost and minimal compression is applied.

JSURLS

The JSURLS parameter includes JavaScript or VBScript files in an HTML report (TABLE request) or graph request that generates HTML output. This allows you to customize the display of WebFOCUS HTML reports or graphs with any JavaScript or VBScript functions. The JavaScript and VBScript files are the last files loaded, and are loaded in the order they are listed, allowing complete customization of the HTML page.

In addition, when a WebFOCUS report or graph is run, a set of pre-defined JavaScript functions is invoked. Some of these functions apply to all WebFOCUS reports, others are specific to OLAP, Web Services and Table of Contents reports. Using JSURLS you can disable or modify these default functions. To view the full set of pre-defined JavaScript functions, see /ibi/WebFOCUSxx/ibi_apps/ibi_html/javaassist/ibi/html/js/ibigl.js.

The syntax is:

```
SET JSURLS='/file1 [/file2] [/file3]...'
```

where:

/file1 [/file2] [/file3]...

Are the files that contain JavaScript or VBScript. If there is more than one js file, the delimiter is a blank and the values must be enclosed in single quotes. Files must be in a location that is accessible by the web server.

You can reference files with a URL.

KEEPDEFINES

The KEEPDEFINES parameter controls whether a virtual field created for a host or joined structure is retained after a JOIN command is run. This parameter applies when a DEFINE command precedes the JOIN command.

The syntax is:

```
SET KEEPDEFINES = {ON|OFF}
```

where:

ON

Retains the virtual field after a JOIN command is run.

OFF

Clears the virtual field after a JOIN command is run. OFF is the default value.

KEEPFILTERS

By default, filters defined on the host data source are cleared by a JOIN command. However, filters can be maintained when a JOIN command is issued, by issuing the SET KEEPFILTERS=ON command.

Setting KEEPFILTERS to ON reinstates filter definitions and their individual declared status after a JOIN command. The set of filters and virtual fields defined prior to each join is called a context (see your documentation on SET KEEPDEFINES and on DEFINE FILE SAVE for information about contexts as they relate to virtual fields). Each new JOIN or DEFINE FILE command creates a new context.

If a new filter is defined after a JOIN command, it cannot have the same name as any previously defined filter unless you issue the FILTER FILE command with the CLEAR option. The CLEAR option clears all filter definitions for that data source in all contexts.

When a JOIN is cleared, each filter definition that was in effect prior to the JOIN command and that was not cleared, is reinstated with its original status. Clearing a join by issuing the JOIN CLEAR join_name command removes all of the contexts and filter definitions that were created after the JOIN join_name command was issued.

The syntax is:

```
SET KEEPFILTERS = {OFF|ON}
```

where:

OFF

Does not preserve filters issued prior to a join. This is the default value.

ON

Preserves filters across joins.

LANG[UAGE]

The LANG[UAGE] parameter specifies the National Language Support (NLS) environment. It sets the language of server error messages and can also be used to set the language of report titles if the Master File contains alternate language TITLE attributes. For more information, see the *Describing Data With WebFOCUS Language* manual.

The syntax is:

```
SET LANG[UAGE] = [LNG|ln]
```

where:

LNG

Is the 3-letter abbreviation used to specify a language.

ln

Is the 2-letter ISO code used to specify a language.

The abbreviations and ISO codes used to specify a language are shown in the following table.

Language Name (Code)	Displayed Language (GUI)	Language Abbreviation	Language ISO code
AMENGLISH or ENGLISH or UKENGLISH	English	AME or ENG or UKE	en

Language Name (Code)	Displayed Language (GUI)	Language Abbreviation	Language ISO code
ARABIC	Arabic	ARB	ar
BALTIC	Lithuanian	BAL	lt
CZECH	Czech	CZE	cs
DANISH	Danish	DAN	da
DUTCH	Dutch	DUT	nl
FINNISH	Finnish	FIN	fi
FRENCH	French - Standard or Canadian	FRE	fr fc
GERMAN	German - Standard or Austrian	GER	de at
GREEK	Greek	GRE	el
HEBREW	Hebrew	HEB or HEW	iw
ITALIAN	Italian	ITA	it
JAPANESE	Japanese-JIS or EUC	JPN or JPE	ja or je
KOREAN	Korean	KOR	ko
POLISH	Polish	POL	po
PORTUGUESE	Portuguese- Brazil or Portugal	POR	br pt
RUSSIAN	Russian	RUS	ru
S-CHINESE	Chinese-Simplified GB	PRC	zh

Language Name (Code)	Displayed Language (GUI)	Language Abbreviation	Language ISO code
SPANISH	Spanish	SPA	es
SWEDISH	Swedish	SWE	sv
T-CHINESE	Chinese-Traditional Big-5	ROC	tw
THAI	Thai	THA	th
TURKISH	Turkish	TUR	tr

LAYOUTGRID

Displays a grid in the report output, which enables you to evaluate the correct placement of data and objects during your report design. This option is applicable only when using the PDF, PS, or DHTML report output.

The syntax is:

```
SET LAYOUTGRID = {ON|OFF}
```

where:

ON

Displays a grid in the report output.

OFF

Turns off the grid in the report output. OFF is the default value.

LAYOUTRTL

Displays report output from right to left for supported formats.

The syntax is:

```
SET LAYOUTRTL = {ON|OFF}
```

where:

ON

Displays report output from right to left.

OFF

Displays report output from left to right. OFF is the default value.

Note: Activating the right-to-left setting will shift the table contents to align to the right of the page, but will not move images placed in headings or footings.

LEADZERO

Leading zeros are truncated in Dialogue Manager strings. The functions in WebFOCUS, when called in Dialogue Manager, may return a numeric result. If the format of the result is YMD and contains a 00 for the year, the 00 is truncated.

The syntax is:

```
SET LEADZERO = {ON|OFF}
```

where:

ON

Allows the display of leading zeros if present.

OFF

Truncates leading zeros if present. OFF is the default value.

LEFTMARGIN

The LEFTMARGIN parameter sets the StyleSheet left boundary for report contents on a page. This parameter applies to PostScript and PDF reports.

The syntax is:

```
SET LEFTMARGIN = {value|.250}
```

where:

value

Is the left boundary of report contents on a page. 0.250 inches is the default value.

LINES

The LINES parameter sets the maximum number of lines of printed output that appear on a page, from the heading to the footing. It sets the maximum number of lines that appear on a logical page, from the heading at the top to the footing on the bottom. The value of LINES can range between 1 and 999999 (specify 999999 for continuous forms). WebFOCUS will generate HEADING and FOOTING text after each group of lines as defined with the SET parameter.

The syntax is:

```
SET LINES = {n|57}
```

where:

n
Is the maximum number of lines of output that appear on a logical page. 57 is the default value.

LOOKGRAPH

The LOOKGRAPH parameter specifies a graph style. This can be a number of variations of area, bar, histogram, and pie charts, among other graph styles.

The syntax is:

```
SET LOOKGRAPH = option
```

where:

option

Specifies a graph style. An extensive number of graph styles is available for WebFOCUS. For a complete list, see the *Creating Reports With WebFOCUS Language* manual.

MATCHCOLUMNORDER

The MATCHCOLUMNORDER parameter controls how fields in MATCH FILE requests are sequenced in the MATCH output. The new default method groups fields across files with their sort field values in the resulting HOLD file regardless of their position within the MATCH request. The old legacy method appends them to the HOLD file record as they are referenced in the request.

The syntax is:

```
SET MATCHCOLUMNORDER = {GROUPED|UNGROUPED}
```

where:

GROUPED
Groups verb objects with their highest-level common sort keys. This can result in the fields being propagated to the HOLD file in a different order from the legacy process. This can affect a subsequent request against the MATCH results if that request uses the default alias names generated in the HOLD Master File. This typically occurred when fields with the same name, not used as keys, were merged together with MATCH. The advantage of the new technique is that it supports HOLD file formats such as FORMAT FOCUS that generate an associated Master File.

UNGROUPED
: Does not group verb objects with their sort keys across files when laying out the resulting HOLD file record. Fields are appended to the HOLD file record as they are referenced in the request.

MAXDATAEXCPT

Data exceptions occur when data that is supposed to contain a numeric value is manipulated in ways unsupported by the architecture of the operating environment. You can change the number of data exceptions allowed before the session is terminated using the SET MAXDATAEXCPT command.

Note: SET MAXDATAEXCPT is functional on mainframe platforms only. All other platforms allow the syntax, but do not support the functionality.

If this command is issued in a TABLE request using the ON TABLE SET phrase, a new count is established for that request. The running session count is saved and is restored after the request executes.

The syntax is:

```
SET MAXDATAEXCPT={0|maxexcpt}
```

where:

maxexcpt
: Is a one to four-digit number that represents how many data exceptions can occur before the session is terminated. The value zero (0) allows an unlimited number of data exceptions, and is the default value. The value one (1) terminates the session at the first data exception.

  If MAXDATAEXCPT is changed in a request, a new count is established and the session counter is saved and then restored after the request executes. If you issue the command outside of a TABLE request, the running counter is reset to zero.

MAXLRECL

The MAXLRECL parameter does not apply to FOCUS data sources.

It specifies the maximum length of a record described with the Master File OCCURS attribute.

The syntax is:

```
SET MAXLRECL = {n|12288}
```

where:

n

Is the maximum record length for a record with the Master File OCCURS attribute. 12288 is the default value. The maximum value allowed depends on your platform.

MDICARDWARN

The MDICARDWARN parameter displays a warning message every time the cardinality in a dimension exceeds a specified value, offering you the chance to study the MDI build. When the number of equal values of the data in a dimension reaches a specified percent, a warning message is issued. In order for MDICARDWARN to be reliable, the data source should contain at least 100,000 records.

Note: In addition to the warning message, a number displays in brackets. This number is the least number of equal values for the dimension mentioned in the warning message text.

The syntax is:

```
SET = MDICARDWARN = n
```

where:

n

Is a percentage value from 0 to 50.

MDIENCODING

The MDIENCODING parameter enables retrieval of output from the MDI file without reading the data source.

FOCUS encodes indexed values any time a field or dimension of an MDI has a MAXVALUES attribute specified or is involved in a parent-child relationship. Encoded values are stored in the MDI file at rebuild time and can be retrieved and decoded with a TABLE request that specifies the MDIENCODING command. The MDIENCODING command allows the user to get output from the MDI file itself without having to read the data source.

The following rules apply to fields in a TABLE request that uses MDIENCODING:

- Only one MDI can be referred to at a time.
- Only dimensions that are part of the same parent-child hierarchy can be used simultaneously in a request. A dimension that is not part of a parent-child relationship can be used as the field in a request if it has a MAXVALUES attribute.

The syntax is:

```
SET MDIENCODING = {ON|OFF}
```

where:

ON

Enables retrieval of output from the MDI file without reading the data source.

OFF

Requires access of the data source to allow retrieval of MDI values.

Note: This command can only be issued in an ON TABLE phrase. It has no default value.

MDIPROGRESS

The MDIPROGRESS parameter displays messages about the progress of an MDI build. The messages show the number of data records accumulated for every *n* records inserted into the MDI as it is processed.

The syntax is:

```
SET MDIPROGRESS = {n|0}
```

where:

n

Is an integer greater than 1000, which displays a progress message for every *n* records accumulated in the MDI build. 100,000 is the default value.

0

Disables progress messages.

MESSAGE

The MESSAGE parameter displays or suppresses informational messages in the view source of your web browser. This parameter cannot be used with ON TABLE SET.

The syntax is:

```
SET {MESSAGE|MSG} = {ON|OFF}
```

where:

ON

Displays informational messages. ON is the default value.

OFF

Suppresses both informational messages and carets that appear when FOCUS executes commands in procedures. Error messages and the carets that prompt for input are still displayed.

MISSINGTEST

By default, when an IF-THEN-ELSE expression is used to calculate a result and the IF expression evaluates to zero (for numeric expressions) or blank (for alphanumeric expressions), the left hand side is checked to see if it has MISSING ON. If it does, the result of the expression will be MISSING, not true or false, and the outcome returned will be MISSING, not the result of evaluating the THEN or ELSE expression, if the field only needs some missing values. You can use the SET MISSINGTEST command to eliminate the missing test for the IF expression so that either the THEN expression or the ELSE expression will be evaluated and returned as the result.

The syntax is:

```
SET MISSINGTEST = {OLD|SPECIAL}
```

where:

OLD

Includes the IF expression in the missing values evaluation. If the IF expression evaluates to MISSING, the result is also MISSING, if the missing field only needs some missing values. This is the default value.

SPECIAL

Excludes the IF expression from the missing values evaluation so that it results in either true or false, not MISSING. If it evaluates to true, the THEN expression is used to calculate the result. If it evaluates to false, the ELSE expression is used to calculate the result.

MULTIPATH

The MULTIPATH parameter controls whether a parent segment is included in report output when selection tests are done on independent paths. That is, it determines whether WHERE or IF tests on separate paths are treated as if an AND or OR operator connects them.

The syntax is:

```
SET MULTIPATH = {SIMPLE|COMPOUND}
```

where:

SIMPLE

Treats selection tests as if connected by an OR operator. The parent segment displays if the condition is true for one of the tests. The following message also displays:

```
Warning. Testing in independent sets of data
```

COMPOUND

Treats selection tests as if connected by an AND operator. The parent segment displays only if the conditions are true for all tests. COMPOUND is the default.

NODATA

The NODATA parameter determines the character string that indicates missing data in a report.

The syntax is:

```
SET {NODATA|NA} = {string|.}
```

where:

string

Is the character string that indicates missing data in reports. A period (.) is the default value.

NULL

The NULL parameter enables you to create a variable-length comma or tab delimited HOLD file that differentiates between a missing value and a blank string or zero value.

The HOLD formats supported for SET NULL=ON are COM, COMT, TAB, and TABT. Missing values in a record are denoted by two consecutive delimiters. A record that starts with a missing value has a delimiter in the first position, and a record that ends with a missing value has a delimiter in the last position.

The syntax is:

```
SET NULL = {ON|OFF}
```

where:

ON

Propagates missing values to a delimited HOLD file when the field has MISSING=ON in the Master File.

OFF

Propagates the value zero for a missing numeric value and blank ("") for a missing alphanumeric value to a delimited HOLD file. OFF is the default value.

OFFLINE-FMT

The OFFLINE-FMT parameter determines the format of printed report output generated from a request.

The syntax is:

```
SET OFFLINE-FMT = option
```

where:

option

Is one of the following:

STYLED formats the report according to the active StyleSheet.

STANDARD produces the report as unstyled character-based output. STANDARD is the default value.

OLAPGRMERGE

For an OLAP graph that has multiple BY fields, or a BY and ACROSS field, determines whether these graphs are merged into a single graph.

The syntax is:

```
SET OLAPGRMERGE={ON|OFF}
```

where:

ON

Turns on the merge graph option. With this setting, AUTODRILL is disabled for the graph.

OFF

Turns off the merge graph option and creates a separate graph for every value of the outer sort field. OFF is the default value.

OLDSTYRECLEN

The OLDSTYRECLEN parameter determines whether the record length, LRECL, is set to the current setting of LRECL=0, or the older setting of LRECL=512 that was used prior to WebFOCUS Version 5 Release 3.4.

The syntax is:

```
SET OLDSTYRECLEN = {ON|OFF}
```

where:

ON

Determines that LRECL=512.

OFF

Determines that LRECL=0. OFF is the default value.

ONFIELD

The ONFIELD parameter determines whether ON phrases that refer to fields not present in the request are ignored or cause the request to terminate. Allowing ON phrases for absent fields enables user selections at run time to determine which elements are included in each execution of the request.

Note that any field used must be present in the Master File for the data source or the following message is generated and execution terminates:

The syntax is:

```
SET ONFIELD = {ALL|IGNORE}

ON TABLE SET ONFIELD {ALL|IGNORE}
```

where:

ALL

Issues a message and terminates execution when a field referenced in an ON phrase is not present in the request. ALL is the default value.

IGNORE

Ignores ON phrases that reference fields that are not present in the request as well as ON phrases that include options not supported by the type of field specified.

ONLINE-FMT

The ONLINE-FMT parameter determines the format of report output. (Applies to WebFOCUS only.)

The syntax is:

```
SET ONLINE-FMT = option
```

where:

option

Is one of the following:

HTML specifies that the report displays as an HTML page. HTML is the default.

PDF specifies that the report displays as a PDF document (Adobe Acrobat Portable Document Format).

Excel 2000 (EXL2K) specifies that the report displays as an Excel 2000 worksheet.

Excel 97 (EXL97) specifies that the report displays as an Excel 97worksheet.

STANDARD specifies that the report will be displayed using a legacy character-based and line-based layout and a monospaced font.

POSTSCRIPT (PS) specifies that the report displays, according to the current StyleSheet, as a PostScript document. You must have installed a third-party tool capable of displaying PS.

ORIENTATION

The ORIENTATION parameter specifies the page orientation for reports styled with StyleSheets.

The syntax is:

```
SET ORIENTATION = {PORTRAIT|LANDSCAPE}
```

where:

PORTRAIT

Displays the page in portrait style. PORTRAIT is the default value.

LANDSCAPE

Displays the page in landscape style.

OVERFLOWCHAR

The OVERFLOWCHAR parameter controls the characters displayed in a numeric report column when the column does not provide enough space to display its value. The number of overflow characters displayed is the same as the length assigned to the field. By default, the displayed overflow character is the asterisk (*).

The syntax is

```
SET OVERFLOWCHAR = 'char'
```

where:

char

Is a single byte displayable character. Depending on the character specified, it may not need to be enclosed in single quotation marks (').

The following characters are not supported as the overflow character: numeric digits, comma, period, apostrophe, percent sign, minus sign, space, current currency symbol, dollar sign, Yen symbol, Pound Sterling sign, and Euro symbol. In addition, other symbols may have significance in your operating environment.

PAGE[-NUM]

The PAGE[-NUM] parameter controls the numbering of output pages.

The syntax is:

```
SET PAGE[-NUM] = option
```

where:

option

Is one of the following:

ON displays the page number on the upper left-hand corner of the page. ON is the default value.

OFF suppresses page numbering.

NOPAGE suppresses page breaks, causing the report to be printed as a continuous logical page. When PAGE is set to NOPAGE, the LINES parameter controls where column headings are printed. You can use NOLEAD in place of NOPAGE.

PAGESIZE

The PAGESIZE parameter specifies the paper size for report pages.

The syntax is:

```
SET PAGESIZE = size
```

where:

size

Specifies the page size. If the actual paper size does not match the PAGESIZE setting, your report is either cropped or contains extra blank space.

The page size options are:

LETTER sets the page size to 8.5 x 11 inches.
ENVELOPE-PERSONAL sets the page size to 3.625 x 6.5 inches.
ENVELOPE-MONARCH sets the page size to 3.875 x 7.5 inches.
ENVELOPE-9 sets the page size to 3.875 x 8.875 inches.
ENVELOPE-10 sets the page size to 4.125 x 9.5 inches.
ENVELOPE-12 sets the page size to 4.5 x 11 inches.
ENVELOPE-DL sets the page size to 4.3 x 8.6 inches.
ENVELOPE-ITALY sets the page size to 4.3 x 9.1 inches.
ENVELOPE-B4 sets the page size to 9.8 x 13.9 inches.
ENVELOPE-B5 sets the page size to 6.9 x 9.8 inches.
ENVELOPE-B6 sets the page size to 6.9 x 4.9 inches.
ENVELOPE-C3 sets the page size to 12.75 x 18 inches.
ENVELOPE-C4 sets the page size to 9 x 12.75 inches.
ENVELOPE-C5 sets the page size to 6.4 x 9 inches.
ENVELOPE-C6 sets the page size to 4.5 x 6.375 inches.
ENVELOPE-C65 sets the page size to 4.5 x 9 inches.

STATEMENT sets the page size to 5.5 x 8.5 inches.
EXECUTIVE sets the page size to 7.5 x 10.5 inches.
GERMAN-STANDARD-FANFOLD sets the page size to 8.5 x 12 inches.
GERMAN-LEGAL-FANFOLD sets the page size to 8.5 x 13 inches.
FOLIO sets the page size to 8.5 x 13 inches.
LEGAL sets the page size to 8.5 x 14 inches.
10X14 sets the page size to 10 x 14 inches.
TABLOID sets the page size to 11 x 17 inches.

A3 sets the page size to 11.7 x 16.8 inches.
A4 sets the page size to 8.25 x 11.7 inches.
A5 sets the page size to 5.8 x 8.25 inches.
B4 sets the page size to 9.8 x 13.9 inches.
B5 sets the page size to 7.2 x 10.1 inches.
C sets the page size to 17 x 22 inches.
D sets the page size to 22 x 34 inches.
E sets the page size to 34 x 44 inches.
US-STANDARD-FANFOLD sets the page size to 14.875 x 11 inches.
LEDGER sets the page size to 17 x 11 inches.
QUARTO sets the page size to 8.5 x 10.8 inches.

PANEL

The PANEL parameter sets the maximum line width, in characters, of a report panel for a screen or printer. If report output exceeds this value, the output is partitioned into several panels. For example, if you set PANEL to 80, the first 80 characters of a record appear on the first panel, the second 80 characters appear on the second panel, and so on.

When printing a report to your screen, the ideal value for the PANEL parameter is the width of your screen (usually 80). When printing to your printer, the ideal value for PANEL is the print width of your printer (usually 132). If PANEL is larger or set to 0, long report lines wrap around the screen or page.

When the BYPANEL parameter is OFF, a report can be divided into a maximum of 4 panels. If SET BYPANEL has a value other than OFF, the report may be divided into 99 panels.

When the STYLESHEET parameter is in effect, PANEL is ignored.

The syntax is:

```
SET PANEL = {0|n}
```

where:

n

Is the maximum line width, in characters, of a report panel.

0

Does not divide the report into panels. Long report lines wrap around the screen or page. 0 is the default value.

Note: The PANEL parameter affects only character-based output format. All other output formats that are not character-based ignore the PANEL parameter (HTML, AHTML, EXL2K) or perform their own paneling (PDF). To be effective, the PANEL parameter requires that SET ONLINE-FMT=STANDARD or SET ONLINE-FMT=STYLED.

PASS

The PASS parameter enables user access to a data source or stored procedure protected by Information Builders security.

This command cannot be used with ON TABLE SET.

The syntax is:

```
SET PASS = password [IN filename]
```

where:

password

Is the password that allows access to data sources protected by Information Builders database security.

filename

Is a specific FOCUS data source or stored procedure protected by security.

PCOMMA

The PCOMMA parameter controls the retrieval of comma-delimited files.

By default, when a Master File specifies SUFFIX=COM, incoming alphanumeric values are not enclosed in double quotation marks, and each record is terminated with a comma and dollar sign (,$) character combination. This format does not support retrieval of most comma-delimited files produced by a PC application.

The syntax is:

```
SET PCOMMA = option
```

where:

option

Can be one of the following:

- **ON**, which enables the retrieval of comma-delimited data sources created by a PC application, in which alphanumeric data is enclosed in double quotation marks and each record is completely contained on one line and is terminated with a carriage return and line feed. It can also retrieve comma-delimited data sources in which alphanumeric data is not enclosed in double quotation marks and each record is terminated with a comma and dollar sign.
- **OFF**, which does not enable the retrieval of comma-delimited data sources created by a PC application. It indicates that alphanumeric data is not enclosed in double quotation marks and each record is terminated with a comma and dollar sign. OFF is the default value.
- **DFIX**, which causes delimited files with SUFFIX=COM, COMT, TAB, and TABT to be processed through the Adapter for DFIX. This processing provides more complete and meaningful messages and some changes to the processing of missing values when two delimiters in a row are encountered. With DFIX processing, a missing value is assigned to the field.

In order to be eligible for DFIX processing, the delimited file must satisfy the following requirements.

- Each record must be completely contained on one line and terminated with the crlf (carriage return/linefeed) character combination.
- The ENCLOSURE can be only in the first position after the delimiter for COM (new) and COMT records. Otherwise, it will not be recognized.
- The number of fields on a line cannot exceed the number of fields defined in the Master File.

PDFLINETERM

The PDFLINETERM parameter determines if an extra space is appended to each record of a PDF output file to facilitate proper file transfer between Windows and UNIX.

In Windows systems, the end of each PDF file has a table containing the byte offset, including two line termination characters, a carriage return, and a line feed. In UNIX, files are terminated by only one character, a line feed. Transferring files between Windows and UNIX systems requires the proper use of the PDFLINETERM parameter.

The syntax is:

```
SET PDFLINETERM = {STANDARD|SPACE}
```

where:

STANDARD

Creates a PDF file without any extra characters. This file will be a valid PDF file if transferred in text mode to a Windows machine, but not to a UNIX machine. If subsequently transferred from a UNIX machine to a Windows machine in text mode, it will be a valid PDF file on the Windows machine.

SPACE

Creates a PDF file with an extra space character appended to each record. This file will be a valid PDF file if transferred in text mode to a UNIX machine, but not to a Windows machine. If subsequently transferred from an ASCII UNIX machine to a Windows machine in binary mode, it will be a valid PDF file on the Windows machine.

PERMPASS

The PERMPASS parameter establishes a user password that remains in effect throughout a session or connection. You can issue this setting in any supported profile but is most useful when established for an individual user by setting it in a user profile. It cannot be set in an ON TABLE phrase. It is recommended that it not be set in EDASPROF because it would then apply to all users.

All security rules established in the DBA sections of existing Master Files are respected when PERMPASS is in effect. The user cannot issue the SET PASS or SET USER command to change to a user password with different security rules. Any attempt to do so generates the following message:

```
permanent PASS is in effect. Your PASS will not be honored.
VALUE WAS NOT CHANGED
```

Only one permanent password can be established in a session. After it is set, it cannot be changed within the session.

The syntax is:

```
SET PERMPASS=userpass
```

where:

userpass

Is the user password used for all access to data sources with DBA security rules established in their associated Master Files.

PHONETIC_ALGORITHM

The PHONETIC_ALGORITHM parameter sets a phonetic algorithm to use with the PHONETIC function, which calculates an index for alphanumeric values such as names, based on their pronunciation, so that words that have variations in spelling can be grouped together.

The syntax is:

```
SET PHONETIC_ALGORITHM = {METAPHONE|SOUNDEX}
```

where:

METAPHONE

Uses the Metaphone algorithm for indexing. Metaphone is suitable for use with most English words, not just names. Metaphone algorithms are the basis for many popular spell checkers. METAPHONE is the default algorithm, except on z/OS.

Note: Metaphone is not optimized in the SQL sent to a relational DBMS. Therefore, if you need to optimize the request for an SQL DBMS, the SOUNDEX value should be used.

SOUNDEX
: Soundex is a legacy phonetic algorithm for indexing names by sound, as pronounced in English. SOUNDEX is the default algorithm on z/OS.

POPUPDESC

The POPUPDESC parameter enables you to view a pop-up field description in an HTML report when your mouse pointer is positioned over a column title in the report output.

The syntax is:

```
SET POPUPDESC = {ON|OFF}
```

where:

ON
: Enables pop-up field descriptions.

OFF
: Deactivates pop-up field descriptions. OFF is the default value.

PPTXGRAPHTYPE

The PPTXGRAPHTYPE attribute enhances the quality of charts embedded into PowerPoint (PPTX) slides. You can use the PNG output format to enhance the image and text quality and support transparency. This is useful for a number of important scenarios, including use of templates with background color and for overlapping a chart with other components and drawing objects.

The syntax is:

```
SET PPTXGRAPHTYPE={PNG|PNG_NOSCALE|JPEG}
```

where:

PNG
: Scales the PNG image to twice its dimensions to get significantly improved quality. This may cause problems if you have non-scalable items in the chart, such as text with absolute point sizes (including embedded scales headings). The output file is also larger due to the larger bitmap. Text within the chart is noticeably sharper than the legacy JPEG format.

  PNG preserves font sizes in the chart when it is internally rescaled for increased resolution. It converts absolute font sizes set in the StyleSheet (*GRAPH_SCRIPT) to sizes expressed in virtual coordinates (which are relative to the dimensions of the chart) and generates font sizes for embedded headings and footings in virtual coordinates.

PNG_NOSCALE

Renders in PNG, but does not scale. This produces slightly better quality than JPEG. Going from JPEG to PNG_NOSCALE makes the chart sharper, but has only a slight effect on the text.

JPEG

Indicates legacy format. This is the default value.

PRINT

The PRINT parameter specifies the report output destination.

It determines whether report output is sent to your screen or to the printer.

You can enter ONLINE and OFFLINE as separate commands that have the same effect as specifying ONLINE and OFFLINE as PRINT settings.

The syntax is:

```
SET PRINT = {ONLINE|OFFLINE}
```

where:

ONLINE

Sends report output to a browser. ONLINE is the default value.

OFFLINE

Sends report output to the system printer or to a file. Output can also be sent to a file labeled with the online ddname using the FILEDEF command.

PRINTDST

The handling of DST operators has been improved to support multiple DST operators in the same request, and the ability to use DST with ACROSS.

With these improvements, you can control the behavior of requests that use the PRINT command with multiple DST operators to achieve independent DST values. To implement this functionality, set the PRINTDST parameter to NEW.

The syntax is:

```
SET PRINTDST = {OLD|NEW}
```

where:

OLD

Processes multiple DST operators in a PRINT request as nested BY fields, making them dependent on each other. OLD is the default value.

NEW

Processes multiple DST operators in a PRINT request as totally independent objects.

PRINTPLUS

The PRINTPLUS parameter introduces printing enhancements. PRINTPLUS is not supported with StyleSheets.

The syntax is:

```
SET {PRINTPLUS|PRTPLUS} = {ON|OFF}
```

where:

ON

Handles page breaks internally to provide the correct spacing of pages. You can perform RECAPs in cases where pre-specified conditions are met. Additionally, a SUBFOOT prints above the footing instead of below it.

OFF

Does not support StyleSheets. OFF is the default value.

PSPAGESETUP

The PSPAGESETUP parameter causes the paper source used by a PostScript printer to match the PAGESIZE parameter setting. It enables the PAGESIZE and ORIENTATION parameter settings, which determine the size and orientation of the paper. If PSPAGESETUP is set to OFF, the settings for PAGESIZE and ORIENTATION are ignored.

The syntax is:

```
SET PSPAGESETUP = {OFF|ON}
```

where:

OFF

Does not include PostScript code for the selection of a PostScript printer paper source. OFF is the default value.

ON

Includes PostScript code that automatically tells a PostScript printer to set its paper source to the size specified by PAGESIZE.

QUALCHAR

The QUALCHAR parameter specifies the qualifying character to be used in qualified field names.

The syntax is:

```
SET QUALCHAR = {character|.}
```

where:

character

Is a valid qualifying character. They include:

.	period	(hex 4B)
:	colon	(hex 7A)
!	exclamation point	(hex 5A)
%	percent sign	(hex 6C)
¦	broken vertical bar	(hex 6A)
\	backslash	(hex E0)

A period (.) is the default value. The use of the other qualifying characters listed above is restricted and should not be used with 66-character field names.

If the qualifying character is a period, you can use any of the other characters listed above as part of a field name. If you change the default qualifying character to a character other than the period, then you cannot use that character in a field name.

QUALTITLES

The QUALTITLES parameter uses qualified column titles in report output when duplicate field names exist in a Master File. A qualified column title distinguishes between identical field names by including the segment name.

The syntax is:

```
SET QUALTITLES = {ON|OFF}
```

where:

ON

Uses qualified column titles when duplicate field names exist and FIELDNAME is set to NEW.

OFF

Disables qualified column titles. OFF is the default value.

RANK

The RANK parameter determines how rank numbers are assigned when a request contains the [RANKED] BY [HIGHEST|LOWEST] *n* phrase and multiple data values fall into the same rank category. If the rank number for the next group of values is the next sequential integer, the ranking method is called *dense*. If the rank number for the next group of values is the previous rank number plus the number of multiples, the ranking method is called *sparse*.

The syntax is:

```
SET RANK = {DENSE|SPARSE}
```

where:

DENSE

Specifies dense ranking. With this method, each rank number is the next sequential integer, even when the same rank is assigned to multiple data values. DENSE is the default value.

SPARSE

Specifies sparse ranking. With this method, if the same rank number is assigned to multiple data values, the next rank number will be the previous rank number plus the number of multiples.

RECAP-COUNT

The RECAP-COUNT parameter includes lines containing a value created with RECAP when counting the number of lines per logical page for printed output.

The number of lines per page is determined by the LINES parameter.

The syntax is:

```
SET RECAP-COUNT = {ON|OFF}
```

where:

ON

Counts lines containing a value created with RECAP.

OFF

Does not count lines containing a value created with RECAP. OFF is the default value.

RECORDLIMIT

The RECORDLIMIT parameter limits the number of records retrieved.

The syntax is:

```
SET RECORDLIMIT = {n|RECORDLIMIT}
```

where:

n

Is the maximum number of records to be retrieved.

RECORDLIMIT

Respects explicit RECORDLIMIT values only. RECORDLIMIT is the default.

Note: Using SET RECORDLIMIT disables AUTOINDEX.

RIGHTMARGIN

The RIGHTMARGIN parameter sets the StyleSheet right boundary for report contents on a page. This parameter applies to PostScript and PDF reports.

The syntax is:

```
SET RIGHTMARGIN = {value|.250}
```

where:

value

Is the right boundary of report contents on a page. 0.250 inches is the default value.

RPAGESET

The RPAGESET parameter controls how the number of lines per logical page are determined when output contains text created with SUBFOOT and a field value created with RECAP.

The syntax is:

```
SET RPAGESET = {NEW|OLD}
```

where:

NEW

Sets the number of lines per logical page equal to the LINES value plus two plus the number of the highest BY field with a SUBFOOT.

OLD

Sets the number of lines per logical page equal to the value of the LINES parameter. See the LINES parameter for details. OLD is the default.

SAVEDMASTERS

The SAVEDMASTERS parameter saves a Master File in memory after it is used in a request. Saving a Master File prevents re-parsing the Master File when referenced in subsequent requests, resulting in performance improvement.

Up to 99 Master Files can be saved to memory.

This parameter cannot be set in the ON TABLE SET command.

The syntax is:

```
SET SAVEDMASTERS = n
```

where:

n

Is an integer between 0 and 99 that specifies the maximum number of Master Files on the SAVEDMASTERS list. 10 is the default value.

Note that the most recently used Master File is always stored in memory, even with SAVEDMASTERS set to zero. However, the zero setting does not generate the list of saved Master Files.

SAVEMATRIX

The SAVEMATRIX parameter saves the matrix from your request to protect it from being overwritten when using Dialogue Manager commands.

The syntax is:

```
SET SAVEMATRIX = {ON|OFF}
```

where:

ON

Saves the internal matrix from the last report request, preventing it from being overwritten.

OFF

Overwrites the internal matrix for each request. In WebFOCUS, OFF is the default.

SHADOW

The SHADOW parameter is only useful for activating the Absolute File Integrity feature for FOCUS database files.

The syntax is:

```
SET SHADOW [PAGE] = {ON|OFF|OLD}
```

where:

ON

Activates the FOCUS Absolute File Integrity feature. The maximum number of pages shadowed is 256K.

OFF

Deactivates the Absolute File Integrity feature. OFF is the default value.

OLD

Indicates that your FOCUS file was created before Version 7.0. This means that the maximum number of pages shadowed is 63,551.

SHIFT

The SHIFT parameter controls the use of *shift* strings.

The syntax is:

```
SET SHIFT = {ON|OFF}
```

where:

ON

Specifies a shift string for Hebrew or DBCS (double-byte character support).

OFF

Indicates that SHIFT is not in effect. OFF is the default value.

SHORTPATH

The SHORTPATH parameter controls how screening conditions against missing cross-referenced segment instances are processed in a left outer join.

In WebFOCUS, the command SET ALL = ON or JOIN LEFT_OUTER specifies a left outer join. With a left outer join, all records from the host file display on the report output. If a cross-referenced segment instance does not exist for a host segment instance (called a *short path*), the report output displays missing values for the fields from the cross-referenced segment. However, the fields are not assigned missing values for testing purposes.

If there is a screening condition on the dependent segment, those dependent segment instances that do not satisfy the screening condition are omitted from the report output, and so are their corresponding host segment instances. With missing segment instances, tests for missing values fail because the fields in the segment have not been assigned missing values.

When a relational engine performs a left outer join, it processes host records with missing cross-referenced segment instances slightly differently from the way WebFOCUS processes those records when both of the following conditions apply:

- There is a screening condition on the cross-referenced segment.
- A host segment instance does not have a corresponding cross-referenced segment instance.

When these two conditions are true, WebFOCUS omits the host record from the report output, while relational engines supply null values for the fields from the dependent segment and then apply the screening condition. If the missing values pass the screening condition, the entire record is retained on the report output. This type of processing is useful for finding or counting all host records that do not have matching records in the cross-referenced file or for creating a DEFINE-based join from the cross-referenced segment with the missing instance to another dependent segment.

If you want WebFOCUS to assign null values to the fields in a missing segment instance when a left outer join is in effect, you can issue the command SET SHORTPATH=SQL.

```
SET SHORTPATH = {FOCUS|SQL}
```

where:

FOCUS

Omits a host segment from the report output when it has no corresponding cross-referenced segment and the report has a screening condition on the cross-referenced segment.

SQL

Supplies missing values for the fields in a missing cross-referenced segment in an outer join. Applies screening conditions against this record and retains the record on the report output if it passes the screening test.

Note: There must be an outer join in effect, either as a result of the SET ALL=ON command or a JOIN LEFT_OUTER command (either inside or outside of the Master File).

SHOWBLANKS

The SHOWBLANKS parameter preserves leading and internal blanks in HTML and EXL2K report output.

The syntax is:

```
SET SHOWBLANKS = {OFF|ON}
```

where:

OFF

Removes leading and internal blanks in HTML and EXL2K report output. OFF is the default value.

ON

Preserves leading and internal blanks in HTML and EXL2K report output.

SPACES

The SPACES parameter sets the number of spaces between columns in a report.

This parameter does not work with HTML, PDF, or styled reports.

The syntax is:

```
SET SPACES = {AUTO|n}
```

where:

AUTO

Automatically places either one to two spaces between columns. AUTO is the default value.

n

Is the number of spaces to place between columns of a report. Valid values are integers between one and eight.

SQLTOPTTF

The SQLTOPTTF parameter enables the SQL Translator to generate TABLEF commands instead of TABLE commands.

The syntax is:

```
SET SQLTOPTTF = {ON|OFF}
```

where:

ON

Generates TABLEF commands when possible. For example, a TABLEREF command is generated if there is no JOIN or GROUP BY command. ON is the default value. ON is the default value.

OFF

Always generates TABLE commands.

SQUEEZE

The SQUEEZE parameter applies only to the StyleSheet feature.

It determines the column width in report output. The column width is based on the size of the data value or column title, or on the field format defined in the Master File.

The syntax is:

```
SET SQUEEZE = {ON|OFF|n}
```

where:

ON

Assigns column widths based on the widest data value or widest column title, whichever is longer.

OFF

Assigns column widths based on the field format specified in the Master File. This value pads the column width to the length of the column title or field format descriptions, whichever is greater. OFF is the default value.

n

Represents a specific numeric value, based on the UNITS parameter setting, to which the column width can be set (valid only in PDF and PS).

%STRICTMATH

In WebFOCUS, the %STRICTMATH parameter determines the behavior of a calculation when dividing by zero.

The syntax is:

```
SET %STRICTMATH = {'OLD'|'NEW'}
```

where:

OLD

Returns zero when a value is divided by zero. This value must be enclosed in single quotation marks (').

NEW

Returns zero when a value is divided by zero. The following error message is produced in the HTML source of the report output.

```
(FOC201) INTERRUPT. DIVISION BY ZERO
```

NEW is the default and must be enclosed in single quotation marks (').

STYLEMODE

The STYLEMODE parameter determines the type of HTML generated for report output.

Note: The SQUEEZE parameter must be set to OFF to align data columns from one page to the next.

The syntax is:

```
SET STYLEMODE = {FULL|FIXED|PAGED}
```

where:

FULL

Displays normal HTML output. FULL is the default value.

FIXED

Generates <PRE tag in HTML to indicate preformatted text which is not to be treated as HTML.

PAGED

In WebFOCUS, this displays a report output in multiple HTML tables where each table is a separate report page. These smaller HTML files are retrieved from the web server quicker than a single large file.

STYLE[SHEET]

The STYLE[SHEET] parameter controls the format of report output by accepting or rejecting StyleSheet parameters. The parameters specify formatting options, such as page size, orientation, and margins.

The syntax is:

```
SET STYLE[SHEET] = {stylesheet|ON|OFF}
```

where:

stylesheet

Is the name of the StyleSheet file. For UNIX and Windows, this is the name of the StyleSheet file without the file extension .sty. For z/OS, this is the member name in the PDS allocated to ddname FOCSTYLE.

For a PDF or PostScript report, it uses the page layout settings for UNITS, TOPMARGIN, BOTTOMMARGIN, LEFTMARGIN, RIGHTMARGIN, PAGESIZE, ORIENTATION, and SQUEEZE. The settings for LINES, PAPER, PANEL, and WIDTH are ignored.

ON

Creates an HTML table using the default proportional font defined in the browser for the end user. ON is the default.

For a PDF or PostScript report, uses the page layout settings for UNITS, TOPMARGIN, BOTTOMMARGIN, LEFTMARGIN, RIGHTMARGIN, PAGESIZE, ORIENTATION, and SQUEEZE; the settings for LINES and WIDTH are ignored.

OFF

Creates a preformatted report using the default fixed font defined in the browser for the end user.

For a PDF or Postscript report, this uses the settings for LINES and WIDTH. The settings for UNITS, TOPMARGIN, BOTTOMMARGIN, LEFTMARGIN, RIGHTMARGIN, PAGESIZE, ORIENTATION, and SQUEEZE are ignored.

This is used when the STYLE parameter is set in an ON TABLE command. It indicates that StyleSheet commands follow and are embedded in the procedure.

SUBTOTALS

The SUBTOTALS parameter specifies whether summary lines are displayed above or below the detail lines in a report. The summary commands affected include SUBTOTAL, SUB-TOTAL, RECOMPUTE, SUMMARIZE, COMPUTE, RECAP, and COLUMN-TOTAL.

The syntax is:

```
SET SUBTOTALS {ABOVE|BELOW}
```

where:

ABOVE

Places summary lines above the detail lines and displays the sort field values on every detail line of the report output.

BELOW

Places summary lines below the detail lines. BELOW is the default value.

SUMMARYLINES

The SUMMARYLINES parameter allows users to combine fields with and without prefix operators on summary lines in one request. Prefix operator processing is used for all summary lines. Fields without prefix operators are processed as though they were specified with the operator SUM.

This command cannot be used with ON TABLE SET.

The syntax is:

```
SET SUMMARYLINES = {NEW|OLD|EXPLICIT}
```

where:

NEW

Propagates all summary operations to the grand total line. Uses prefix operator processing for all summary commands (all summary fields without prefix operators are processed as though they had a SUM. operator). Fields listed in a summary command are populated only on summary lines created by that summary command and on summary lines created by propagation of that summary command. Supports display of alphanumeric fields on summary lines. NEW is the default value.

OLD

This value is no longer supported. It processes as NEW.

EXPLICIT

Does not propagate SUBTOTAL and RECOMPUTE to the grand total line. Uses prefix operator processing for all summary commands (all summary fields without prefix operators are processed as though they had a SUM. operator). Fields listed in a summary command are populated only on summary lines created by that summary command and on summary lines created by propagation of that summary command. Supports display of alphanumeric fields on summary lines.

Note: This command is not supported in a request using the ON TABLE SET syntax.

SUMPREFIX

The SUMPREFIX parameter allows users to choose the answer set display order when using an external sort to perform aggregation on alphanumeric or smart date formats.

The syntax is:

```
SET SUMPREFIX = {FST|LST}
```

where:

FST

Displays the first value when alphanumeric or smart date data types are aggregated.

LST

Displays the last value when alphanumeric or smart date data types are aggregated. LST is the default value.

TARGETFRAME

The TARGETFRAME parameter includes the HTML code <BASE TARGET="*framename*"> in the heading of the HTML file that is displayed in your browser. All drill-down hyperlinks from the base report or graph are directed to the specified frame unless overridden by the TARGET attribute in the StyleSheet.

The syntax is:

```
SET TARGETFRAME = framename
```

where:

framename

Is the frame on the webpage in which the output from the drill-down hyperlink (either a FOCEXEC or URL) is displayed. Possible values include standard HTML frame names, such as, _blank, _self, _parent, _top, or a user-defined name.

TEMP

In WebFOCUS, the TEMP parameter assigns temporary files to a specific directory.

The syntax is:

```
SET TEMP = directory
```

where:

directory

Is the directory to which all temporary files are assigned.

TEMPERASE

The TEMPERASE parameter determines if the temporary files created during a WebFOCUS connection are retained after the connection is closed. Applies to HOLD files and other files that may be created during the session.

The syntax is:

```
SET TEMPERASE = {ON|OFF}
```

where:

ON

Erases any temporary files after the WebFOCUS connection is closed. ON is the default.

OFF

Keeps any temporary files created during a WebFOCUS connection.

TESTDATE

The TESTDATE parameter temporarily alters the system date in order to test a dynamic window allowing you to simulate clock settings to determine the behavior of your program. Only use TESTDATE for testing purposes with test data. The value of TESTDATE affects all reserved variables that retrieve the current date from the system. Setting TESTDATE also affects anywhere in FOCUS that a date is used (such as CREATE, MODIFY, MAINTAIN) but does not affect the date referenced directly from the system.

TESTDATE can either be equal to TODAY or a date in the format YYYYMMDD. If anything else is entered the following message is displayed:

```
TESTDATE MUST BE YYYYMMDD OR TODAY
```

The syntax is:

```
SET TESTDATE = {yyyymmdd|TODAY}
```

where:

yyyymmdd

Is an 8-digit date in the format YYYYMMDD.

TODAY

Is the current date. TODAY is the default value.

TIME_SEPARATOR

This parameter defines the separator for time components for the &TOD system variable.

The syntax is:

```
SET TIME_SEPARATOR = {DOT|COLON}
```

where:

DOT
: Uses a dot (.) to separate time components. This is the default value.

COLON
: Uses a colon (:) to separate time components.

TITLELINE

The TITLELINE parameter controls underlining of column titles on report output.

The syntax is:

```
SET {TITLELINE|ACROSSLINE} = {ON|OFF|SKIP}
```

where:

ON
: Underlines column titles on report output. ON is the default value.

OFF
: Replaces the underline with a blank line.

SKIP
: Specifies no underline and no blank line.

TITLES

The TITLES parameter controls whether to use pre-defined column titles in the Master File as column titles in report output.

The syntax is:

```
SET TITLES = {ON|OFF|NOPREFIX}

ON TABLE SET TITLES {ON|OFF|NOPREFIX}
```

where:

ON
: Displays the value of the TITLE attribute as the column heading on the report output, if a TITLE attribute exists in the Master File. If the field has a prefix operator in the report request, creates the column heading using both the prefix operator and the TITLE attribute. If there is no TITLE attribute, the field name is used instead. ON is the default value.

OFF

Displays the field name as the column heading on the report output. If the field has a prefix operator in the report request, creates the column heading using both the prefix operator and the field name.

NOPREFIX

Displays the value of the TITLE attribute as the column heading on the report output, if a TITLE attribute exists in the Master File. If there is no TITLE attribute, the field name is used instead. If the field has a prefix operator in the report request, creates the column heading using both the prefix operator and the field name.

TOPMARGIN

The TOPMARGIN parameter sets the top StyleSheet boundary for report contents on a page.

This parameter applies to PostScript and PDF reports.

The syntax is:

```
SET TOPMARGIN = {value|.250}
```

where:

value

Is the top boundary on a page for report output. 0.250 inches is the default value.

UNITS

The UNITS parameter applies to PostScript and PDF reports.

It specifies the unit of measure for page margins, column positions, and column widths.

The syntax is:

```
SET UNITS = {INCHES|CM|PTS}
```

where:

INCHES

Uses inches as the unit of measure. INCHES is the default value.

CM

Uses centimeters as the unit of measure.

PTS

Uses points as the unit of measurement. (One inch = 72 points, one cm = 28.35 points).

USER

The USER parameter enables user access to a data source or stored procedure protected by Information Builders security.

The syntax is:

```
SET USER = user
```

where:

user

Is the user name that, with a password, enables access to a data source or stored procedure protected by Information Builders security.

USERFCHK

The USERFCHK parameter controls the level of verification applied to DEFINE FUNCTION arguments and Information Builders-supplied function arguments. It does not affect verification of the number of parameters. The correct number must always be supplied.

Note that the USERFNS=SYSTEM setting must be in effect. For details, see *USERFNS* on page 638.

Issue the following command in a profile or procedure.

```
SET USERFCHK = setting
```

where:

setting

Can be one of the following:

ON verifies parameters in requests, but does not verify parameters for functions used in Master File DEFINEs. If a parameter has an incorrect length, an attempt is made to fix the problem. If such a problem cannot be fixed, a message is generated and the evaluation of the affected expression is terminated. ON is the default value.

Because parameters are not verified for functions specified in a Master File, no errors are reported for those functions until the DEFINE field is used in a subsequent request when, if a problem occurs, the following message is generated:

```
(FOC003) THE FIELDNAME IS NOT RECOGNIZED
```

OFF does not verify parameters except in the following cases:

- If a parameter that is too long would overwrite the memory area in which the computational code is stored, the size is automatically reduced without issuing a message.

  Note: The OFF setting will be deprecated in a future release.

- If an alphanumeric parameter is too short, it is padded with blanks to the correct length.

FULL is the same as ON, but also verifies parameters for functions used in Master File DEFINEs.

ALERT verifies parameters in a request without halting execution when a problem is detected. It does not verify parameters for functions used in Master File DEFINEs. If a parameter has an incorrect length and an attempt is made to fix the problem behind the scenes, the problem is corrected with no message. If such a problem cannot be fixed, a warning message is generated. Execution then continues as though the setting were OFF.

USERFNS

If your site has a locally written function with the same name as an Information Builders-supplied function, the USERFNS parameter determines which function is used.

Parameter verification can be enabled for DEFINE FUNCTIONs and functions supplied by Information Builders.

The syntax is:

```
SET USERFNS= {SYSTEM|LOCAL}
```

where:

SYSTEM

Gives precedence to functions supplied by Information Builders and to those created with the DEFINE FUNCTION command. SYSTEM is the default value.

This setting is required to enable parameter verification. For details, see *USERFCHK* on page 637.

LOCAL

Gives precedence to locally written functions. Parameter verification is not performed with this setting in effect.

VAUTO

The VAUTO parameter performs automatic scaling of the vertical axis for the given values. If VAUTO is OFF, the vertical axis must be scaled using the VMAX and VMIN parameters. (See *VMAX* on page 641 and *VMIN* on page 641.)

The syntax is:

```
SET VAUTO = {ON|OFF}
```

where:

ON

Scales the vertical axis automatically. ON is the default.

OFF

Does not scale the horizontal axis automatically. The end user must supply values for VMAX and VMIN.

VAXIS

The VAXIS parameter applies to graphs generated offline.

It specifies the length of the vertical axis, in lines. VAXIS is ignored for online displays since WebFOCUS automatically adjusts the length of the graph to the height of the terminal.

The syntax is:

```
SET VAXIS = {n|405}
```

where:

n

Is the page length, in lines. Valid values are integers between 20 and 405. The default value is 405.

VCLASS

The VCLASS parameter specifies the vertical interval mark when AUTOTICK is OFF. See also *VTICK* on page 641.

The syntax is:

```
SET VCLASS = {nnn|0}
```

where:

nnn

Is the vertical interval mark.

0

Specifies zero as the vertical interval mark. The default value is 0.

VGRID

The VGRID parameter draws a grid at the horizontal and vertical class marks of the graph. See also *GRID* on page 583.

The syntax is:

```
SET VGRID = {ON|OFF}
```

where:

ON

Draws a grid at the horizontal and vertical class marks of the graph.

OFF

Does not draw a grid. OFF is the default.

VISBARORIENT

The VISBARORIENT parameter enables you to set horizontal or vertical orientation for visualization bars for ACROSS columns.

The syntax is:

```
SET VISBARORIENT = {H|V}
```

where:

H

Indicates horizontal bar orientation for visualization bars.

V

Indicates vertical bar orientation for visualization bars. This is the default value.

VMAX

The VMAX parameter sets the maximum value on the vertical axis when automatic scaling is not used (VAUTO=OFF).

The syntax is:

```
SET VMAX = {nnn|.00}
```

where:

nnn

Is the maximum value on the vertical axis when VAUTO is OFF. The default value is .00.

VMIN

The VMIN parameter sets the minimum value on the vertical axis when automatic scaling is not used (VAUTO=OFF).

The syntax is:

```
SET VMIN = {nnn|.00}
```

where:

nnn

Is the minimum value on the vertical axis when VAUTO is OFF. The default value is .00.

VTICK

The VTICK parameter sets the vertical axis interval mark when AUTOTICK is OFF. See also *VCLASS* on page 639.

The syntax is:

```
SET VTICK = {nnn|.00}
```

where:

nnn

Is the vertical axis interval mark. The default value is .00.

VZERO

The VZERO parameter treats missing values on the vertical axis as zeros. When VZERO is off, these values are ignored.

The syntax is:

```
SET VZERO = {ON|OFF}
```

where:

ON

Treats missing values as zeros. ON is the default.

OFF

Ignores missing values.

WARNING

The WARNING parameter suppresses (FOC441) warnings. The file exists already. Create will overwrite it.

The syntax is:

```
SET WARNING = {ON|OFF}
```

where:

ON

Turns on warning messages. On is the default.

OFF

Turns off warning messages.

WEBARCHIVE

The WEBARCHIVE parameter packages EXL2K reports together with associated files (such as the pivot cache for the EXL2K PIVOT format) into a single web archive document (.xmh file). This format is only available for Excel 2002 and later.

The syntax is:

```
SET WEBARCHIVE = {ON|OFF}
```

where:

ON

Packages EXL2K reports together with associated files into a single web archive document (.xmh file). ON is the default value.

OFF

Does not package multiple files into a single document (creates an EXL2K report as an .xht file).

WEBVIEWALLPG

The WEBVIEWALLPG parameter controls whether the WebFOCUS Viewer displays the All Pages button.

The syntax is:

```
SET WEBVIEWALLPG = {OFF|ON}
```

or

```
ON TABLE SET WEBVIEWALLPG {OFF|ON}
```

where:

ON

Displays the All Pages button. ON is the default value.

OFF

Does not display the All Pages button.

WEBVIEWCLOSE

The WEBVIEWCLOSE parameter specifies whether the Close option is displayed on the WebFOCUS Viewer control bar.

The syntax is:

```
SET WEBVIEWCLOSE = {ON|OFF}
```

where:

ON

Displays the Close option. ON is the default value.

OFF

Does not display the Close option.

WEBVIEWCLMSG

The WEBVIEWCLMSG parameter controls whether the WebFOCUS Viewer Close message displays when the Close option is clicked.

The syntax is:

```
SET WEBVIEWCLMSG = {ON|OFF|CLOSE}
```

where:

ON

Displays the Close message string in WebFOCUS language translation file (*XX*webfoc_strings.lng), where *XX* is the two character language abbreviation.

For example, ENwebfoc_strings.lng is the English language translation file. ON is the default value.

OFF

Does not display the Close message. The display page is specified by the WEBVIEWHOME parameter, which will display a blank page if it is not set.

CLOSE

Does not display the Close message and closes the browser window displaying the ODP report.

WEBVIEWER

The WEBVIEWER parameter enables on-demand paging, and invokes the WebFOCUS Viewer. The first page of report output is displayed in the WebFOCUS Viewer while the remaining pages remain on the web or application server until requested by the user.

The syntax is:

```
SET WEBVIEWER = {ON|OFF}
```

where:

ON

Enables on-demand paging. The first page of report output is displayed in the WebFOCUS Viewer, while the remaining pages remain on the web/application server until requested by the user.

OFF

Sends all report output to your browser. The report output displays in a standard browser window. OFF is the default value.

WEBVIEWHELP

The WEBVIEWHELP parameter controls whether the WebFOCUS Viewer displays the Help button.

The syntax is:

```
SET WEBVIEWHELP = {OFF|ON}
```

or

```
ON TABLE SET WEBVIEWHELP {OFF|ON}
```

where:

ON

Displays the Help button. ON is the default value.

OFF

Does not display the Help button.

WEBVIEWHOME

The WEBVIEWHOME parameter allows an HTML page to be displayed when the WebFOCUS Viewer is closed.

The syntax is:

```
SET WEBVIEWHOME = {home_URL|OFF}
```

where:

home_URL

Is a valid URL that displays an HTML page when you close the WebFOCUS Viewer.

OFF

Displays a blank browser window when you close the WebFOCUS Viewer. You must enter another URL to run another report. OFF is the default value.

Note: For more information on enabling accessibility, see *ACCESSPDF* on page 531.

WEBVIEWTARG

The WEBVIEWTARG parameter allows the user to open the WebFOCUS Viewer in a target frame.

The syntax is:

```
SET WEBVIEWTARG = {target_frame|OFF}
```

where:

target_frame

Is the name of an existing frame in the browser or one of the following reserved HTML target frames:

`_blank` opens the WebFOCUS Viewer in a new browser window. This is the default for reports that do not have accessibility enabled.

`_self` opens the WebFOCUS Viewer in the same frame as the anchor.

`_parent` opens the WebFOCUS Viewer in the immediate parent frame that contains the anchor.

`_top` opens the WebFOCUS Viewer in the frame from which you ran the report.

`OFF`

Opens the WebFOCUS Viewer in the frame from which you ran the report. This is the default for reports that have accessibility enabled.

Note: When SET ACCESSIBLE = 508, the default for ODP is WEBVIEWTARG = OFF. For more information on enabling accessibility, see *ACCESSHTML* on page 530 and *Section 508 Accessibility in WebFOCUS* on page 36.

WEBVIEWTITLE

The WEBVIEWTITLE parameter defines the title text for the window containing the WEBVIEWER instance.

The syntax is:

```
SET WEBVIEWTITLE='text'
```

or

```
ON TABLE SET WEBVIEWTITLE 'text'
```

where:

text
Is the title text for the Web Viewer window, enclosed in single quotation marks (').

WEEKFIRST

The WEEKFIRST parameter specifies a day of the week as the start of the week. This is used in week computations by the HDIFF, HNAME, HPART, HYYWD, and HSETPT functions, described in the *Using Functions* manual.

The HPART and HNAME subroutines can extract a week number from a date-time value. To determine a week number, they can use ISO 8601 standard week numbering, which defines the first week of the year as the first week in January with four or more days. Any preceding days in January belong to week 52 or 53 of the preceding year.

Depending on the value of WEEKFIRST, these functions can also define the first week of the year as the first week in January with seven days.

The WEEKFIRST parameter does not change the day of the month that corresponds to each day of the week, but only specifies which day is considered the start of the week.

The syntax is:

```
SET WEEKFIRST = {value|7}
```

where:

value
Can be:

1 through 7, representing Sunday through Saturday with non-standard week numbering.

or

ISO1 through ISO7, representing Sunday through Saturday with ISO standard week numbering.

Note: ISO is a synonym for ISO2.

The ISO standard establishes Monday as the first day of the week, so to be fully ISO compliant, the WEEKFIRST parameter should be set to ISO or ISO2.

WPMINWIDTH

If you need the report width for a format WP output file to remain fixed across releases for later processing of the output file, you can set the width you need using the SET WPMINWIDTH command. This parameter specifies the minimum width of the output file. It will be automatically increased if the width you set cannot accommodate the fields propagated to the output file in the request. On z/OS, The LRECL of the output file will be four bytes more than the report width because the file is variable length and needs an additional four bytes to hold the actual length of each record instance. In other operating environments, the length of the record is the value of WPMIDWIDTH.

The syntax is:

```
SET WPMINWIDTH = {0|nnn}

ON TABLE SET WPMINWIDTH {0|nnn}
```

where:

nnn

Is the minimum width of the output file. On z/OS, the LRECL will automatically be *nnn* + 4 bytes. If you specify zero (0) for *nnn*, the width will be calculated automatically based on the report request. If the width you specify cannot accommodate the fields propagated to the output file, it will be automatically increased enough to accommodate them.

XRETRIEVAL

The XRETRIEVAL parameter previews the format of a report without actually accessing any data. This parameter enables you to perform TABLE, TABLEF, or MATCH requests and produce HOLD Master Files without processing the report.

The syntax is:

```
SET XRETRIEVAL = {ON|OFF}
```

where:

ON

Performs retrieval when previewing a report. ON is the default value.

OFF

Specifies that no retrieval is to be performed.

YRTHRESH

The YRTHRESH parameter defines the start of a 100-year window globally or on a field-level. Used with DEFCENT, interprets the current century according to the given values. Two-digit years greater than or equal to YRTHRESH assume the value of the default century. Two-digit years less than YRTHRESH assume the value of one more than the default century. (See *DEFCENT* on page 562.)

Note: This same result can be achieved by including the FDEFCENT and FYRTHRESH attributes in the Master File.

The syntax is:

```
SET YRTHRESH = {[-]yy|0}
```

where:

yy

Is the year threshold for the window. 0 is the default value.

If *yy* is a positive number, that number is the start of the 100-year window. Any 2-digit years greater than or equal to the threshold assume the value of the default century. Two-digit years less than the threshold assume the value of one more than the default century.

If *yy* is a negative number (*-yy*), the start date of the window is derived by subtracting that number from the current year, and the default century is automatically calculated. The start date is automatically incremented by one at the beginning of each successive year.

Chapter 9

Defining and Allocating WebFOCUS Files

You will come across a variety of files when you use WebFOCUS. These files can be application files that you create, extract files that WebFOCUS creates as requested by your application, or work files that WebFOCUS creates as required by your application.

This topic describes WebFOCUS file allocation requirements and describes how to assign a logical name, or ddname, to a file or device for operating systems that support this feature.

In this chapter:

- Allocating WebFOCUS Files
- Application Files Under Windows
- Extract Files Under Windows
- Work Files Under Windows
- Determining If A File Exists Under Windows
- WebFOCUS Files Under MVS
- Application Files Under MVS
- Extract Files Under MVS
- Work Files Under MVS
- Reviewing Attributes of Allocated Files Under MVS
- Application Files Under UNIX
- Extract Files Under UNIX
- Work Files Under UNIX
- Determining If A File Exists Under UNIX

Allocating WebFOCUS Files

WebFOCUS files fall into three categories: application files, extract files, and work files. This topic contains information about files and ways to define them in the Windows and UNIX environments, and all portable platforms.

Some files are automatically allocated by WebFOCUS, but there are times when you must explicitly define a file and its location. A FILEDEF command generates platform independent file paths for all portable platforms by creating FILEDEF syntax with application names.

Note: In order to use spaces in a field or directory name within a FILEDEF command, you must surround the entire path with double quotation marks ("). For example:

```
FILEDEF TEMPFLE DISK "C:\dir1\dir2\directory space\tempfle.ftm"
```

Syntax: How to Reference a File Under Windows

filename.filetype

where:

filename

Is the name of the file.

filetype

Is the type of file. This is a three-letter extension for an installation of Windows based on the File Allocation Table (FAT) file system.

Reference: WebFOCUS Files Under Windows and UNIX

The following tables describe the WebFOCUS files that you will use under Windows and UNIX. WebFOCUS uses file extensions to distinguish different file types.

Application Files

Extension	Description
.mas	Master File.
.acx	Access File.
.fex	Procedure (FOCEXEC).
.foc	FOCUS data source and external index.
.dat	Sequential data source.
.sty	WebFOCUS StyleSheet file.
.err	Error messages file or help text.
.exe or .dll	Library of functions in the Windows environment.
.prf	Profile.

Extension	Description
.htm .html .jpeg .gif .css .js .class .jar	Files displayed in a web browser.

Extract Files

File	Description
HOLD	Contains data saved using the HOLD command.
SAVB	Contains data saved using the SAVB command.
SAVE	Contains data saved using the SAVE command.
HOLDMAST	Temporary Master File for HOLD files.
.FTM	Contains data saved using the HOLD, SAVB, or SAVE command.

Note: Dialogue Manager output files must be allocated using the FILEDEF command in system or user profiles.

Work Files

File	Description
FOCSTACK	Contains resolved Dialogue Manager procedures.
FOCSORT	Used during sorting.
FOCPOST	Sequential output file saved using the POST command. The PICKUP command reads it back in.
FOCSML	Used by Financial Modeling Language (FML).

File	Description
OFFLINE	Used when the SET PRINT parameter is OFFLINE.
SYSIN	Directs input.
SYSPRINT	Directs output to the screen.

Dynamically Defining a File Under Windows and UNIX

You do not have to explicitly define most of your application files prior to referring to them. WebFOCUS dynamically allocates certain application files. Additionally, WebFOCUS dynamically defines all extract files and temporary work files to the operating system during a session.

WebFOCUS defines the following extract, output, and work files:

- HOLD, SAVB, and SAVE files.
- FOCUS data sources.
- FOCSORT files.
- SYSIN to input from keyboard. It is assigned to TERM. This is equivalent to the command FILEDEF SYSIN TERM.
- SYSPRINT to output to be displayed on the screen. It is assigned to TERM. This is equivalent to the command FILEDEF SYSPRINT TERM.
- OFFLINE to output sent to the printer. It is assigned to PRINTER. This is equivalent to the command FILEDEF OFFLINE PRINTER.
- FOCSTACK to the file FOCSTACK.FTM. This is equivalent to the command FILEDEF FOCSTACK DISK FOCSTACK.FTM. For more information on FOCSTACK, see *Managing Flow of Control in an Application* on page 329.

Assigning a Logical Name With the FILEDEF Command

For a file managed by the operating system, such as an ISAM or comma-delimited data file, the physical file name is the actual name of a file as it appears to the operating system. A logical name (or ddname) is a shorthand name that points to the physical file name. Logical names simplify code by allowing short names to be used in place of the longer physical file name.

The FILEDEF command assigns a logical name to a physical file name and specifies file attributes. You can explicitly define a file and its location to WebFOCUS using the FILEDEF command. This generates platform independent file paths for all portable platforms by creating FILEDEF syntax with application names. An Allocation can be issued in a procedure and lasts for a single request.

It is recommended that instead of including an Allocation in each procedure, you include all FILEDEF commands in a single file that you call with the -INCLUDE command at the beginning of each procedure. This enables you to make changes to your FILEDEF commands globally instead of changing the Allocation information in each procedure.

The FILEDEF command is typically used in the following ways in operating systems that support this command:

- **Identifying data sources.** You must create FILEDEF assignments (or Use directories, in the case of FOCUS data sources) for all data sources you wish to use.
- **Redirecting input and output.** Three ddnames are used for input and output. By reassigning these ddnames, you can redirect input or output:
  - SYSIN for input.
  - SYSPRINT for output displayed on the screen.
  - OFFLINE for output sent to the printer.

You can also use the Universal Naming Convention (UNC) to assign logical names to files that are located on a server. In order to take advantage of the UNC you must first attach to the server you want to use. For information on attaching to a server or mapping network drives, consult your Network Administrator.

Syntax: How to Assign a Logical Name With a FILEDEF Command

```
FILEDEF ddname DISK appname[/appnamea...]/filename [(APPEND] [LRECL n]
[RECFM F]
```

or

```
FILEDEF SYSIN TERM [LOWER]
```

or

```
FILEDEF ddname PRINTER
```

where:

ddname

Is the logical name for the file, input, or output. The *ddname* can be from one to eight characters. When used to associate a data source with a Master File, the *ddname* must match the name of the Master File.

DISK

Associates the specified *ddname* with a file.

appname[/*appnamea*...]

Is the name of the application under APPROOT, or a nested application name, that contains the physical file.

filename

Is the physical name of the file under the appname.

APPEND

Appends records to the end of the file. Without this option, the file is overwritten.

LRECL *n*

Specifies the record length. *n* is an integer.

You must specify an LRECL value if the file is a SAVB file or fixed-format transaction file used in data maintenance (FIXFORM files) that contains binary values.

RECFM F

Specifies fixed length records.

You must specify an RECFM value if the file is a SAVB file or fixed-format transaction file used in data maintenance (FIXFORM files) that contains binary values.

TERM

Specifies the keyboard and monitor as the input source and output destination.

LOWER

Sends keyboard input to WebFOCUS as entered.

PRINTER

Specifies the printer as the output destination.

FOCUS data sources (files with the .foc extension) that do not conform to the default naming convention are identified with the USE command, not FILEDEF. For information on the USE command, see *Accessing a FOCUS Data Source* on page 487.

Example: Assigning a Logical Name to a File Located on a Server

Assign a logical name to a file located on a server using the Universal Naming Convention:

```
FILEDEF DATFILE DISK \\SERVER2\DISK1\MAYSPLES.DAT
```

Example: Redirecting Output

To send output to the LPT1 port (provided that the machine is configured properly):

```
FILEDEF OFFLINE DISK LPT1
```

Example: Setting a Search Path

To search all directories on the search path for the NEW_EMPS.DAT file:

```
FILEDEF SYSIN DISK *:NEW_EMPS.DAT
```

Example: Appending a Report Extract to Other Content

To append a report extract from the LIBRARY data source to the current content of the file LIB03.FTM:

```
FILEDEF SAVE DISK C:\LIBRARY\LIB03.FTM (APPEND
```

Example: Reading Standard Text Editor Files With LRECL

You can specify an LRECL equal to or greater than the implied LRECL for the request. For example, if the length of the longest line in the text file is seven characters, issue this command:

```
FILEDEF BIGLINE DISK BIGLINE.FTM (LRECL 7
```

Displaying Current ddnames Assigned With FILEDEF

The ? FILEDEF command displays the ddnames assigned for various files, input and output.

Syntax: How to Display Current ddnames

```
? FILEDEF
```

Example: Displaying Current ddnames

Issuing the command

```
? FILEDEF
```

produces information similar to the following:

```
Lname   Device   Lrecl   Recfm   Append   Expl   Filename
============================================================
HOLD2   DISK       0     V       N        Y      C:\VM\SMALL\HOLD2.FTM
```

Clearing Allocations

You can clear Allocations by using WebFOCUS syntax.

Syntax: How to Clear a Logical Name With Syntax

```
FILEDEF ddname CLEAR
```

where:

ddname

Is the logical name. It may contain one to eight alphanumeric characters.

CLEAR

Clears the specified ddname.

Application Files Under Windows

You can use WebFOCUS to reference and search for your application files.

When you do not change the default file type, you can prepare a report without issuing a FILEDEF command or other communication. When you change the default file type of a FOCUS data source, or the data source does not reside in a standard search path, you must issue the USE command. The USE command is discussed in *Accessing a FOCUS Data Source* on page 487.

Master Files Under Windows

A Master File contains metadata about a data source, and is identified with a file name and extension, for example, *filename*.mas. It consists of attributes that describe a data source. A Master File and the data source that it describes usually have the same name. A Master File must consist of 80-byte, fixed length records.

The description of a data source and all files it cross-references must be available whenever you refer to the data source.

For more information about how to generate Master Files, see the *Describing Data With WebFOCUS Language* manual.

Locating a Master File Under Windows

WebFOCUS searches for a Master File on the path of the WebFOCUS Reporting Server, using either EDAPATH or APP PATH logic.

Note: WebFOCUS searches the current temporary directory, where HOLD Master Files are created, before searching EDAPATH or APP PATH.

Reference: Naming a Master File Under Windows

The following rules apply:

- The file name can be up to 64 characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are therefore not recommended or supported.
- The file extension must be .mas for a Master File, since WebFOCUS searches for a Master File with that extension by default.

Access Files Under Windows

For some data sources, an Access File supplements a Master File. An Access File includes additional information that completes the description of the data source. For example, it includes the full data source name and location. You need one Master File, and for some data sources, one Access File, to describe a data source.

WebFOCUS searches for an Access File on the path of the WebFOCUS Reporting Server using either EDAPATH or APP PATH logic.

Reference: Naming an Access File Under Windows

The following rules apply:

- The file name can be up to eight characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are therefore not recommended or supported.
- The file extension must be .acx since WebFOCUS searches for an Access File with that extension by default.

- The file name should have the same name as the corresponding Master File.

Procedures Under Windows

A procedure is an executable text file that contains your report request. It can have up to 3024 characters per line. A procedure is identified with a file name and extension, for example, *filename*.fex.

WebFOCUS searches for a procedure on the path of the WebFOCUS Reporting Server, using either EDAPATH or APP PATH logic.

Reference: Naming a Procedure Under Windows

The following rules apply:

- The file name can be up to 64 characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are therefore not recommended or supported.
- The file extension must be .fex since WebFOCUS searches for a procedure with that extension by default.

FOCUS Data Sources Under Windows

FOCUS data sources contain data written in FOCUS format. All FOCUS data sources have a record length of 4096 and a fixed length record format. The maximum size of a FOCUS data source is 1G or 256K pages.

A FOCUS data source is identified with a file name and extension, for example, *filename*.foc. The name of a FOCUS data source usually matches the name of the corresponding Master File. For example, if the Master File is ledger.mas, then the FOCUS data source is ledger.foc. You can override this default with the USE command, described in *Accessing a FOCUS Data Source* on page 487.

WebFOCUS searches for a FOCUS data source on the platform with the WebFOCUS Reporting Server or subserver.

Reference: Naming a FOCUS Data Source Under Windows

The following rules apply:

- The file name can be up to 64 characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are therefore not recommended or supported.
- The file extension must be .foc since WebFOCUS searches for a FOCUS data source with that extension by default.

External Indexes for FOCUS Data Sources Under Windows

An external index is a FOCUS data source that contains index, field, and segment information for one or more specified FOCUS data sources. The external index is independent of its associated FOCUS data source and is used to improve retrieval performance. In Windows, the external index is automatically allocated as a permanent file when it is created using REBUILD.

Other Supported Data Sources Under Windows

WebFOCUS supports many data source types. For details, see the *Adapter Administration* manual.

WebFOCUS searches for a data source on the platform with the WebFOCUS Reporting Server or subserver.

Reference: Naming a Supported Data Source Under Windows

WebFOCUS supports standard data source naming conventions.

Sequential Data Sources Under Windows

WebFOCUS recognizes sequential data sources formatted in the following ways:

- Fixed format, in which each field occupies a predefined position in the record.
- Free-format, also known as comma-delimited, in which fields can occupy any position in a record.

For details, see the *Describing Data With WebFOCUS Language* manual.

WebFOCUS StyleSheet Files Under Windows

WebFOCUS StyleSheets enable you to style and produce reports that highlight key information. With StyleSheets, you can specify various stylistic characteristics of a report, style report components individually, and define a hyperlink from any reporting object. You can also create StyleSheets externally or within a report request. For details, see the *Creating Reports With WebFOCUS Language* manual.

Note that WebFOCUS searches for a WebFOCUS StyleSheet on the path of the WebFOCUS Reporting Server, using either EDAPATH or APP PATH logic.

Reference: Naming a StyleSheet File Under Windows

The following rules apply:

- The file name can be up to 64 characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are therefore not recommended or supported.
- The file extension must be .sty since WebFOCUS searches for a StyleSheet with that extension by default.

Library of Functions Under Windows

External function libraries are located outside of WebFOCUS. For more information on function libraries, see the *Using Functions* manual.

Profiles Under Windows

WebFOCUS supports the following levels of profiles to provide flexibility in designing and running applications.

- A **global profile** is a startup file that is automatically created during the installation and configuration of the server. It contains the default environment settings that are required for the proper operation of the server.
- A **user profile** is available based on a user ID. A user profile is a file used by the server to specify settings that apply to a server environment, but only for a specific user ID.

Webpages Under WebFOCUS Under Windows

Webpages and related files provide the presentation logic for an application. They include files displayed for the user in a browser, such as HTML files, graphical images, Java class and archive files, JavaScript files, and Cascading Style Sheets.

The web server searches for a webpage or related files in the web server home directory or web server alias. An HTML file called by -HTMLFORM must reside in a path defined by the WebFOCUS Reporting Server.

Reference: Naming a Webpage Under Windows

WebFOCUS supports standard file naming conventions. Webpages must have an extension of .htm.

Extract Files Under Windows

WebFOCUS creates all extract files in a temporary directory on the WebFOCUS Reporting Server, or in a user-defined location on the WebFOCUS Reporting Server.

Reference: Locating Extract Files Under Windows

WebFOCUS creates all extract files in a temporary directory under the directory specified by the EDATEMP environment variable in EDASERVE.CFG.

HOLD Files Under Windows

A HOLD extract file is a sequential data source that contains the results of a report request. It may have a corresponding HOLD Master File. The extension for an extract file is .ftm unless you specify the FORMAT option. If a Master File is created, it has the same name as the extract file, with the extension .mas.

If you specify the FORMAT FOCUS option with HOLD, WebFOCUS creates an extract file and a Master File, each with the file name FOC$HOLD. These files are then used as input to the procedure that creates the final FOCUS file. The new FOCUS and Master Files are created in the user temporary directory.

Syntax: How to Create a HOLD File Under Windows

```
ON TABLE HOLD [AS filename]
```

where:

filename

Is a name for the HOLD file. If you do not specify a file name, HOLD is used as the default name. Since each subsequent HOLD command overwrites the previous HOLD file, it is useful practice to code a distinct file name in each request to direct the extracted data to a separate file, preventing it from being overwritten.

For the complete syntax with all options, see the *Creating Reports With WebFOCUS Language* manual.

SAVB Files Under Windows

A SAVB file is an extract file containing the results of a report request in internal format. That is, all numeric fields are stored in binary, and all character fields are padded with spaces to a multiple of four bytes. The file cannot be printed.

When a SAVB file is created, WebFOCUS does not create a Master File.

Syntax: How to Create a SAVB File Under Windows

```
ON TABLE SAVB [AS filename]
```

where:

filename

Is the name of the file. The default is SAVB. The default extension is .ftm.

For the complete syntax with all options, see the *Creating Reports With WebFOCUS Language* manual.

SAVE Files Under Windows

A SAVE file is an extract file that saves the data of a report, but does not save headings or subtotals, or create a Master File. It is a simple sequential character data file that can be used by other programs, or merged into another data file using a data maintenance request. The default format is simple character, although you can specify formats compatible with many other software products. WebFOCUS saves all columns in the report in printable, character format with no spaces between columns.

A SAVE file contains the external character format equivalent to SAVB. The command syntax and allocations are the same as SAVB. However, the numbers are printable Extended Binary Coded Decimal Interchange Code (EBCDIC) characters and no padding takes place.

Syntax: How to Create a SAVE File Under Windows

```
ON TABLE SAVE [AS filename]
```

where:

filename

Is the name of the file. The default is SAVE. The default extension is .ftm.

HOLDMAST Files Under Windows

A HOLDMAST file is a temporary Master File. It is created with the HOLD command. The APP HOLDMETA command specifies its location.

Syntax: How to Specify the Location of a Temporary Master File Under Windows

```
APP HOLDMETA appname
```

where:

appname

Is a valid application name.

Example: Specifying the Location of a Temporary Master File Under Windows

The following example specifies a location for a temporary Master File, and assigns a logical name to a physical data source.

```
1. APP HOLDMETA app1
2. FILEDEF MKTSALES DISK C:\TMP\MKTSALES.FTM
    TABLE FILE GGSALES
    SUM DOLLARS BY REGION
    ON TABLE HOLD AS MKTSALES FORMAT ALPHA
    END
```

The numbers listed next to the code correspond with the following notes:

1. Specifies the location of the Master File in the app1 application directory. WebFOCUS uses the same logical name, MKTSALES, specified in the second line, to name the temporary Master File.
2. Specifies MKTSALES as the logical name for the physical data source. The HOLD command creates the temporary Master File mktsales.mas and the data source mktsales.ftm.

Work Files Under Windows

WebFOCUS creates work files as required by your application. WebFOCUS creates all extract files in a temporary directory on the WebFOCUS Reporting Server, or in a user-defined location on the WebFOCUS Reporting Server.

The available work files are:

- **FOCSTACK** files contain resolved Dialogue Manager procedures.
- **FOCSORT**, allocated as FOCSORT.FOC, may be required for sorting with the TABLE, TABLEF, and MATCH commands. The maximum size of a FOCSORT file is 1G of 256K pages.

- **FOCPOST** files hold sequential output, which is saved by the POST command and read by the PICKUP command.
- **FOCSML** files are used by the Financial Modeling Language (FML) facility.

Example: Sending Output to a File With OFFLINE Under Windows

The following request sends report output to the file EMPL.OUT, which is stored on the C: drive in the \tmp directory:

```
OFFLINE
FILEDEF OFFLINE DISK C:\TMP\EMPL.OUT

TABLE FILE MOVIES
PRINT TITLE BY DIRECTOR
WHERE CATEGORY IS 'ACTION'
END
```

Determining If A File Exists Under Windows

The STATE command sets the system variable &RETCODE to a value that indicates whether a specified file exists.

Syntax: How to Determine If a File Exists Under Windows

```
[opsys] STATE filename
```

where:

opsys

Identifies the operating environment. For Windows, the value can be WINNT or DOS.

filename

Is the file whose existence you want to determine. It can be a file name in the current path, a full path name, or an application name and filename in the form appname/filename.

After issuing the command, the variable &RETCODE has the value zero (0) if the file exists and the value -1 if the file does not exist. If the file does not exist, the following message is generated:

```
File doesn't exist
```

WebFOCUS Files Under MVS

You will come across a variety of files in WebFOCUS. These files fall into three categories: application files, extract files, and work files. This topic contains information about files and ways to define them in the MVS environment.

Some files are automatically allocated by WebFOCUS, but there are times when you must explicitly define a file and its location to WebFOCUS. You can do this by using the DYNAM and ALLOCATE commands. For details on these commands see the *Stored Procedures Reference Manual*.

Referencing Files Under MVS

Reference files by ddnames rather than by their fully qualified data set names. Data set names are arbitrary but must conform to the naming standards of your installation.

Reference: WebFOCUS File Types Under MVS

WebFOCUS uses ddnames to distinguish different file types.

The following tables describe the WebFOCUS files used under MVS:

Application Files

File	Description
MASTER	Master File.
ACCESS	Access file.
FOCEXEC	Procedure.
ddname	FOCUS data source and external index.
ddname	Other supported data sources.
FOCSTYLE	WebFOCUS StyleSheet file.

Extract Files

File	Description
*HOLD	Contains data saved using the HOLD command.
*SAVB	Contains data saved using the SAVB command.
*SAVE	Contains data saved using the SAVE command.
HOLDMAST	Temporary Master File for HOLD files.
HOLDSTAT Files	Documentation and/or DBA information for extract files.

Note: There are also extract files that you must allocate if used: LOG, POST, and Dialogue Manager output files.

Work Files

File	Description
FOCSTACK	Contains resolved Dialogue Manager procedures.
FOCSORT	Used during sorting.
*FOCPOST	Holds sequential output.
FOCSML	Used by Financial Modeling Language (FML).

*The AS phrase renames files marked with an asterisk to allow more than one file of that type in a single session.

Dynamically Defining Files Under MVS

You do not have to explicitly define most of your files. WebFOCUS automatically allocates many application files on an as-needed basis, as long as they follow the data set naming conventions. WebFOCUS allocates five primary cylinders and five secondary cylinders for dynamically allocating files. WebFOCUS dynamically allocates work files, but does not automatically define application files. You need to define these files.

WebFOCUS will allocate some or all of the following output or work files as temporary data sets:

- HOLD, SAVB, and SAVE files.
- FOCUS data sources created by the CREATE FILE command.
- FOCSTACK, FOCSORT, and HOLDMAST files.
- FOCSML files.
- OFFLINE, SYSIN, and SYSPRINT files.

You can explicitly allocate files in the server JCL. For more information about allocating files, see your Reporting Server documentation.

Note: Work files should not be allocated in the server JCL.

Application Files Under MVS

This topic describes how WebFOCUS references and searches for your application files.

Master Files Under MVS

The DCB attributes of the MASTER are: RECFM=FB, LRECL=80, with BLKSIZE a multiple of LRECL.

Master Files are comma-delimited files that contain metadata about data sources. When WebFOCUS needs a Master File, it searches for the ddname MASTER. If it does not find the member there, it next searches the ddname HOLDMAST, followed by EDASYNM. If it does not find the member in HOLDMAST either, WebFOCUS returns an error message.

The partitioned data set (PDS) member names must correspond to the WebFOCUS names of Master Files. For example, the command TABLE FILE CAR requires a member CAR in the MASTER file PDS.

Several partitioned data sets of Master Files can be concatenated under the ddname MASTER, provided that they have identical LRECL and RECFM attributes.

To create and maintain Master Files under TSO, use the ISPF editor.

Example: Allocating a Master File Under MVS

A typical allocation for a Master File is:

```
//MASTER DD DSN=qualif.EDAMFD.DATA,DISP=SHR
```

The entire PDS (and not a particular member) must be referenced in the allocation.

Access Files Under MVS

For some data sources, an Access File supplements a Master File. An Access File includes additional information that completes the description of the data source. For example, it includes the full data source name and location. You need one Master File, and for some data sources, one Access File, to describe a data source.

An Access File is allocated to ddname ACCESS. The DCB attributes of the ACCESS PDS are: RECFM=FB, LRECL=80, with BLKSIZE a multiple of LRECL.

The PDS member names must correspond to the name of the Master File. For example, the FOCUS command TABLE FILE CAR requires a member CAR in the ACCESS file PDS.

Example: Allocating an Access File Under MVS

A typical allocation for an ACCESS file is:

```
//ACCESS DD DSN=prefix.ACCESS.DATA,DISP=SHR
```

The entire PDS must be referenced in the allocation, not a particular member.

Procedures Under MVS

A procedure can be stored in one or more partitioned data sets allocated to ddname FOCEXEC. The name of the procedure corresponds to the member name.

The DCB attributes: RECFM=FB, LRECL=80, with BLKSIZE a multiple of LRECL.

In TSO, you can also create a procedure using the ISPF editor. In batch mode, you can create procedures with the utility programs IEBPTPCH and IEBUPDTE.

Several partitioned data sets of EDARPC procedures may be concatenated under the ddname of FOCEXEC.

Example: Allocating a Procedure Under MVS

A sample allocation for a procedure is:

```
//FOCEXEC DD DSN=qualif.EDARPC.DATA,DISP=SHR
```

FOCUS Data Sources Under MVS

FOCUS databases contain data written in FOCUS format. The maximum size of a FOCUS database is 2G. Each data source is allocated to a ddname that matches the member name for the Master File in the PDS that is allocated to ddname MASTER. For example, if the Master File for the data source has the member name LEDGER, then the data source is allocated to ddname LEDGER. If you want to override this default, you can issue the USE command, or use the DATASET attribute in the Master File or Access File. For more information, see *Accessing a FOCUS Data Source* on page 487.

FOCUS data sources are formatted in 4096-byte pages. We strongly recommend that you allocate FOCUS data sources in cylinders, if the file is big enough to justify this.

Example: Allocating a FOCUS Data Source

The following example applies to a new FOCUS data source:

```
//CAR   DD DSNAME=CAR.FOCUS,UNIT=SYSDA,DISP=(NEW,CTLG),
//   VOL=SER=MYVOL,SPACE=(CYL,(5,5)), LRECL=4096, RECFM=F, BLKSIZE=4096
```

Using FOCUS Data Sources Under MVS

Existing FOCUS data sources can always be allocated as SHR even when being modified. Concurrent destructive updates are prevented by WebFOCUS itself, by reserving exclusive use of the data source, for update purposes, for the duration of the MODIFY command. At the conclusion of the MODIFY, the user which had possession of the data source relinquishes control, making the data source available to the other users.

While only one user is allowed to update the data source, multiple users may have the data source open for read-only purposes, concurrently. This scheme offers the same protection as exclusive allocation (DISP=OLD), but it is more flexible, because with DISP=OLD, the data source is reserved for the duration of the allocation. (For a batch job, this is the entire time from initiation to termination.)

If a user attempts to modify a FOCUS data source currently being modified by another user or batch job, or if the data source is controlled by a sink machine, the result is a message stating that the data source is in use elsewhere. The MODIFY command flushes to the END statement allowing you to perform other tasks. On the other hand, if a batch job encounters the same situation, it will wait until the data source is free (that is, until the MODIFY currently in control ends or the sink machine is terminated).

When TABLE and TABLEF commands are run for the purpose of locating cross-referenced segments (segment types KU and KM), you must allocate the data source with DISP=OLD or NEW. This is so their addresses may be written into the data source from which they are being cross-referenced. This process, called pointer resolution, does not take place if the disposition of the data source is SHR or if the data source resides on a sink machine.

External Indexes for FOCUS Data Sources Under MVS

An external index is a FOCUS data source that contains index, field, and segment information for one or more specified FOCUS data sources. The external index is independent of its associated FOCUS data source and is used to improve retrieval performance.

The external index is automatically allocated as a temporary file when it is created using REBUILD. To create a permanent external index, you must allocate it before you issue the REBUILD command.

SORTOUT is a work file that is used during the process to create an external index. You must allocate the work file with the proper amount of space, minus DCB parameters. To estimate the amount of space, use this formula:

```
bytes = (field_length + 20) * number_of_occurrences
```

Other Supported Data Sources Under MVS

You can use the FOCUS query language to read external files that are not in FOCUS format. WebFOCUS can read tables and files for QSAM, ISAM, VSAM, IMS, CA-IDMS/DB, ADABAS, TOTAL, SYSTEM 2000, MODEL 204, Millennium, Supra, DB2, SQL/DS, CA-DATACOM/DB, Teradata, and Oracle.

The ddname under which you should allocate the external file depends on the type of file. For more information about external files, see your Reporting Server documentation.

WebFOCUS StyleSheet Files Under MVS

WebFOCUS StyleSheets enable you to style and produce reports that highlight key information. With StyleSheets, you specify various stylistic characteristics of a report and style report components individually. You can also use StyleSheets to define a hyperlink from any reporting object.

You can create StyleSheets externally or within a report request. For details, see the *Creating Reports With WebFOCUS Language* manual.

The FOCSTYLE data set stores any external StyleSheets used by FOCUS procedure StyleSheet declarations.

Example: Allocating Space for ddname FOCSTYLE Under MVS

The following example shows how to allocate space for the FOCSTYLE data set:

```
//FOCSTYLE DD DSN=qualif.STY.DATA,DISP=SHR
```

where:

qualif

Is the high-level qualifier for your site.

Profiles Under MVS

WebFOCUS supports the following levels of profiles to provide flexibility in designing and running applications.

- A **global profile** is a startup file that is automatically created during the installation and configuration of the server. It contains the default environment settings that are required for the proper operation of the server.
- A **group profile** is a file used by the server to specify settings that apply to a server environment, but only for a user of a specific security group.
- A **user profile** is available based on a user ID. A user profile is a file used by the server to specify settings that apply to a server environment, but only for a specific user ID.

For details on profiles, see your Reporting Server documentation.

Extract Files Under MVS

Extract files save lines of output during a WebFOCUS request. They contain data taken from existing data sources.

HOLD Files Under MVS

A HOLD extract file is a sequential data source that contains the results of a report request. It may have a corresponding HOLD Master File. The ddname is HOLD unless you specify the FORMAT option. WebFOCUS provides the DCB parameters in accordance with the record length of the report it is about to store. The DCB BLKSIZE parameter is automatically calculated.

If you specify the FORMAT FOCUS option with HOLD, WebFOCUS creates an extract file and a Master File, each with the file name FOC$HOLD. These files are then used as input to the procedure that creates the final FOCUS file. The new FOCUS and Master Files are created in the user temporary directory.

Syntax: How to Create a HOLD File Under MVS

```
ON TABLE HOLD [AS ddname]
```

where:

ddname

Is a name for the HOLD file. If you do not specify a file name, HOLD is used as the default name. Since each subsequent HOLD command overwrites the previous HOLD file, it is useful practice to code a distinct file name in each request to direct the extracted data to a separate file, preventing it from being overwritten.

SAVB Files Under MVS

A SAVB file is an extract file containing the results of a report request in internal format. That is, all numeric fields are stored in binary, and all character fields are padded with spaces to a multiple of four bytes. The file cannot be printed.

The server dynamically allocates a temporary sequential data set under ddname SAVB or a ddname you supply with the AS ddname option. WebFOCUS allocates five tracks each for primary and secondary space. Record format is variable blocked with record length and blocksize dependent on the record size. To keep the file, use the DYNAM COPY command.

Syntax: How to Create a SAVB File Under MVS

```
ON TABLE SAVB [AS ddname]
```

where:

ddname

Is name under which WebFOCUS allocates a temporary sequential data set.

SAVE Files Under MVS

A SAVE file is an extract file that saves the data of a report, but does not save headings or subtotals, or create a Master File. It is a simple sequential character data file that can be used by other programs, or merged into another data file using a data maintenance request. The default format is simple character, although you can specify formats compatible with many other software products. WebFOCUS saves all columns in the report, in printable character format, with no spaces between columns.

A SAVE file contains the external character format equivalent to SAVB. The command syntax and allocations are the same as SAVB. However, the numbers are printable Extended Binary Coded Decimal Interchange Code (EBCDIC) characters and no padding takes place.

Syntax: How to Create a SAVE File Under MVS

```
ON TABLE SAVE [AS filename]
```

where:

`AS filename`

Specifies a file name for your SAVE file. If you do not specify a file name, it defaults to SAVE. If you do not specify a file name, subsequent creations of HOLD files will overwrite each other.

HOLDMAST Files: Temporary Master Files Under MVS

When HOLD files are created either under the default name HOLD, or a specified name (for example, HOLD AS MYNAME), the description is written into the PDS whose ddname is HOLDMAST. This PDS is exactly like MASTER except that WebFOCUS creates the members. It is usually a temporary file.

The data control block (DCB) attributes of the HOLDMAST are RECFM=FB, LRECL=80, with BLKSIZE a multiple of LRECL. DCB parameters must not be supplied by the user. The server will create the HOLDMAST file with the current DCB.

If HOLDMAST is not allocated when a HOLD file is created, the server will allocate HOLDMAST as a temporary data set with five primary and five secondary tracks.

Whenever WebFOCUS needs the description of a file, it first searches the ddname MASTER. If the member is not found, it then searches the ddname HOLDMAST.

If you want to retain the HOLDMAST file, give it a name and a DISP parameter.

HOLDSTAT Files Under MVS

HOLDSTAT files enable you to include DBA information and environmental comments in HOLD and PCHOLD Master Files. Member HOLDSTAT in the distributed library EDAMSG.DATA is the default. Alternately, you may create your own HOLDSTAT or another user-specified member in your ERRORS or MASTER PDSs.

The contents of a HOLDSTAT file are included in HOLD and PCHOLD Master Files when the SET HOLDSTAT command is specified to ON or to a member name. For information about the SET HOLDSTAT command, see the *Creating Reports With WebFOCUS Language* manual.

The HOLDSTAT file may exist as a member in the ERRORS or the MASTER library.

A HOLDSTAT file may contain environmental comments like a file header, or the DBA attribute, or both. The supplied HOLDSTAT member contains the following file header with Dialogue Manager system variables:

```
$================================================================$
$      HOLD file created on &DATE at &TOD by FOCUS &FOCREL       $
$              Database records retrieved= &RECORDS              $
$              Records in the HOLD file = &LINES                 $
$================================================================$
```

In the HOLD Master File, the comments appear after the FILE and SUFFIX attributes and the DBA information is appended to the end.

Reference: HOLDSTAT File Rules Under MVS

If you create your own HOLDSTAT file, the following rules apply:

- Each line of comments must begin with a dollar sign ($) in column 1.
- Comments may not include user-defined variables.
- List the DBA information after any comments. On separate lines, specify the keywords $BOTTOM and END, beginning in column 1, followed by the DBA attribute. The syntax is:

  ```
  $BOTTOM
  END
  DBA=attribute,$
  ```

  where:

  attribute

  Is an attribute such as password, user, access, restrict, name, or value.

Work Files Under MVS

This topic describes characteristics of temporary work files used in the MVS environment. The available work files are:

- **FOCSTACK** files contain resolved Dialogue Manager procedures.
- **FOCSORT**, allocated as FOCSORT.FOC, may be required for sorting with the TABLE, TABLEF, and MATCH commands. The maximum size of a FOCSORT file is 1G of 256K pages.
- **FOCPOST** files hold sequential output, which is saved by the POST command and read by the PICKUP command.
- **FOCSML** files are used by the Financial Modeling Language (FML) facility.

Reviewing Attributes of Allocated Files Under MVS

You can check on existing file allocations using the command ? TSO DDNAME. This can be done in the following ways:

Command	Function
`? TSO DDNAME`	Lists allocated files. Produces a listing on file SYSPRINT even if you issue the OFFLINE command.
`? TSO DDNAME ddname`	Lists file attributes. Produces a listing on file SYSPRINT even if you issue the OFFLINE command.
`-? TSO DDNAME ddname`	Places file attribute information into Dialogue Manager variables. Attributes are returned as values for Dialogue Manager variables.

Displaying Allocated Files With ? TSO DDNAME Under MVS

The ? TSO DDNAME command lists all the currently allocated files by ddname, shows the number of occurrences for each and lists their corresponding data set names. It displays currently allocated files.

Syntax: How to Display Allocated Files With ? TSO DDNAME

```
? TSO DDNAME
```

Example: Displaying Allocated Files With ? TSO DDNAME

The following is sample output from a ? TSO DDNAME command.

```
DDNAME              OCCURRENCES       DSNAME

STEPLIB                   2         EDA.V3R2M01.EDALIB.LOAD
                                    INPDQ.PROC.PROCLIB
MASTER                    1         EDA.V3R2M01.EDAMFD.DATA
FOCEXEC                   1         EDA.V3R2M01.EDAMFD.DATA
ERRORS                    1         EDA.V3R2M01.EDAMSG.DATA
FOCSORT                   1         SYS88229.TO95540.RA000.INPDQ.R0000002
OFFLINE                   1         TSOINFOC.TSOINFOC.OFFLINE
SYSUT1                    1         SYS88229.TO95058.RA000.INPDQ.SYSUT1
SYSEDIT                   1         SYS88229.T095058.RA000.INPDO.EDIT
HOLD                      1         SYS88229.TO95058.RA000.INPDQ.R0000003
HOLDMAST                  1         SYS88229.T095044.RA000.INPDQ.R0000004
CAR                       1         EDA.V3R2M01.CAR.FOCUS
```

The DDNAME column lists all allocated file names (ddnames). The OCCURRENCES column shows how many data sets are allocated to the corresponding ddname. This number will be one unless two or more data sets are concatenated under the ddname. The DSNAME column shows the fully qualified data set name (DSN) corresponding to the ddname. For concatenated data sets, all the data set names are shown.

Displaying File Attributes With ? TSO DDNAME ddname Under MVS

The ? TSO DDNAME ddname command displays attributes of the queried file name (ddname). If the specified ddname was not allocated, the command displays the attributes as either blanks or zeroes.

Syntax: How to Display File Attributes With ? TSO DDNAME ddname

```
? TSO DDNAME ddname
```

where:

ddname

Is the ddname whose attributes you want to display. You may specify up to eight characters. You may also create a general list of allocated files by ddname by specifying a wildcard character (* or ?) with part of the ddname.

Displaying File Attributes With ? TSO DDNAME ddname

The following is sample output from a ? TSO DDNAME *ddname* command.

```
DDNAME                =              CAR
DSNAME                =              EDA.VSR2M00.CAR.FOCUS
DISP                  =              OLD
DEVICE                =              DISK
VOLSER                =              TSOPAK
DSORG                 =              PS
RECFM                 =              F
SECONDARY             =                1
ALLOCATION            =              TRACKS
BLKSIZE               =              4096
LRECL                 =              4096
TRKTOT                =                2
EXTENTSUSED           =                1
BLKSPERTRK            =               10
TRKSPERCYL            =               15
CYLSPERDISK           =              886
BLKSWRITTEN           =               20
FOCUSPAGES            =                8
```

Placing File Attributes in Dialogue Manager Commands

The -? TSO DDNAME ddname command places file attribute information in Dialogue Manager variables instead of displaying the information on the terminal. These Dialogue Manager TSO system variables have the same names as the attributes returned by the ? TSO DDNAME ddname command. For a complete list of these variables, see *Dialogue Manager TSO System Variables* on page 679.

If there is no information for a variable, the variable will contain blanks if it is an alphanumeric variable. If it is a numeric variable, the variable will contain zeroes.

Reference: Dialogue Manager TSO System Variables

TSO System Variable	Description
&DDNAME	Queried file name (ddname).
&DSNAME	Fully qualified data set name. For concatenated data sets, only the first data set name is returned.
&DISP	Disposition: OLD, NEW, MOD, or SHR.
&DEVICE	DISK, TAPE, TERM, or READER-PRINTER.
&VOLSER	Disk or tape volume serial number.
&DSORG	Data set organization: PS (sequential), IS (indexed sequential), PO (partitioned), U (undefined), or DA (direct).
&RECFM	Record format, for example, F, FB, V, VB.
&SECONDARY	Quantity specified for secondary allocations.
&ALLOCATION	Unit of secondary allocation: TRACKS, BLOCKS, or CYLINDER.
&BLKSIZE	Blocksize.
&LRECL	Record length, in bytes.

TSO System Variable	Description
&TRKTOT	Total number of currently allocated tracks, spanning both primary and secondary allocation.
&EXTENTSUSED	Total number of currently allocated extents (max 16).
&BLKSPERTRK	The number of blocks (of BLKSIZE bytes) that can be written onto 1 track of the device.
&TRKSPERCYL	The number of tracks per cylinder for the device.
&CYLSPERDISK	The number of cylinders per disk for the device.
&BLKSWRITTEN	The total number of blocks in the data set, assuming that all blocks are BLKSIZE long.
&FOCUSPAGES	Posted only for FOCUS data sources. This number is the total number of 4096-byte pages in the data source. This is the same as the total number of pages shown by the ? FILE file name command. It is also the same as the highest page number shown by the ? FDT file name command. It does not include SHADOW pages and directory.

Example: **Using TSO ddname Variables**

```
1. -? TSO DDNAME &DD.ENTER DDNAME.
2. -IF &DSNAME EQ ' ' GOTO ALLOCATE;
   -TYPE DATASET &DSNAME ALLOCATED TO &DD
   -EXIT
   -ALLOCATE
    .
    .
    .
```

The process is as follows:

1. The command prompts you for the ddname. The information that is entered from the terminal is then placed in the variable &DD. Depending on the value of &DD, the system supplies the information for the variable &DSNAME. If no data set is allocated to the ddname that was supplied, the variable &DSNAME will contain a blank since it is alphanumeric.
2. The command tests if the variable &DSNAME is a blank. If it is blank (that is, no data set was found for the ddname entered by the operator), the procedure jumps to the label -ALLOCATE and continues. If it is not blank (that is, a data set was found for the ddname entered by the operator), the procedure prints a message stating the name of the data set and exits.

For detailed information on Dialogue Manager commands and variables, see *Managing Flow of Control in an Application* on page 329.

Reference: Issuing the -? TSO DDNAME Command From a Windows Server

If you attempt to issue the -? TSO DDNAME command remotely on MVS EDA from a WebFOCUS Windows Server, the following error message is generated:

```
(FOC303) CONTROL LINE NOT RECOGNIZED IN FOCEXEC: -? TSO DDNAME MASTER
```

When sending the -? TSO DDNAME command from a Windows Server to MVS, you must place -? TSO DDNAME and a -TYPE command (to display the results in Windows), in a procedure (FOCEXEC) between -REMOTE BEGIN and -REMOTE END, as follows:

```
-REMOTE BEGIN
ex Remote.fex
-REMOTE END
```

The Remote.fex procedure, which will execute on MVS and send the results back to windows, must contain the following:

```
-? TSO DDNAME
-TYPE &DDNAME &DISP ...
```

You can run Remote.fex from another procedure, or the Command Console.

Determining If a Data Source Exists Under MVS

Since you can logically erase an existing data set by issuing a CREATE command against it, it is necessary to have a means of testing for the existence of an Operating System data set before issuing the CREATE command.

Syntax: How to Test for a Data Set

```
? {MVS|TSO} DSNAME datasetname
```

where:

datasetname

Is the data set name. If you supply only the unqualified data set name, the prefix from the profile is taken to create a fully qualified data set name. Do not specify member names.

Syntax: How to Test for a Data Set in Dialogue Manager

```
-? TSO DSNAME dsname
```

where:

dsname

Is the data set name.

In this case, there is no output message. The system variable &RETCODE is set and must be tested for the outcome. The possible results follow:

&RETCODE Value	Equivalent FOCUS Code/Message
0	(FOC488) Dataset is in catalog:
4	(FOC489) Dataset is in catalog, but not on volume indicated:
8	(FOC490) Dataset is not in catalog:

Estimating Data Set Sizes to Determine Available Space Under MVS

You can use the file attribute information returned by the ? TSO ddname command to determine the number of records in the data set and to estimate how much available space is left on the currently allocated tracks.

Note: The estimates obtained from the following formulas are approximations and do not take into account space reused after it was logically vacated by deleted segments.

- FOCUS data source formulas:

```
Available pages (without additional extents) = BLKSWRITTEN -
FOCUSPAGES
BLKSWRITTEN = BLKSPERTRK x TRKTOT
```

- Fixed-block data set formulas:

```
RECSPERBLK = BLKSIZE/LRECL
No. of records = BLKSWRITTEN x RECSPERBLK
No. of free blocks = (TRKTOT x BLKSPERTRK) - BLKSWRITTEN
```

- Variable-length blocked data sets:

  The formulas for fixed-block data sets (that is, SAVE files) usually apply to variable-length blocked (VB) data sets as well. This is true because VB data sets usually have the same length blocks even though the description implies otherwise.

- If SET SHADOW=ON, use the following formula:

```
(FOCUSPAGES x 2) + 3
```

Application Files Under UNIX

This topic describes how WebFOCUS references and searches for application files.

When you do not change the default file type, you can prepare a report without issuing a FILEDEF command or other communication. When you change the default file type of a FOCUS data source, or the data source does not reside in a standard search path, you must issue the USE command. The USE command is discussed in *Accessing a FOCUS Data Source* on page 487.

Master Files Under UNIX

A Master File contains metadata about a data source, and is identified with a file name and extension, for example, *filename*.mas. It consists of attributes that describe a data source. A Master File and the data source that it describes usually have the same name. A Master File must consist of 80-byte, fixed length records.

The description of a data source and all files it cross-references must be available whenever you refer to the data source.

For more information about how to generate Master Files, see the *Describing Data With WebFOCUS Language* manual.

WebFOCUS searches for a Master File on the path of the WebFOCUS Reporting Sever, using either EDAPATH or APP PATH logic.

Reference: Naming a Master File Under UNIX

The following rules apply:

- The file name can be up to 64 characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are therefore not recommended or supported.
- The file extension must be .mas for a Master File, since WebFOCUS searches for a Master File with that extension by default.

Access Files Under UNIX

For some data sources, an Access File supplements a Master File. An Access File includes additional information that completes the description of the data source. For example, it includes the full data source name and location. You need one Master File, and for some data sources one Access File, to describe a data source.

WebFOCUS searches for an Access File on the path of the WebFOCUS Reporting Server using either EDAPATH or APP PATH logic.

Reference: Naming an Access File Under UNIX

The following rules apply:

- The file name can be up to 64 characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are therefore not recommended or supported.
- The file extension must be .acx since WebFOCUS searches for a FOCUS data source with that extension by default.

Procedures Under UNIX

A procedure contains your report requests and can contain up to 80 bytes on a single line, but cannot have control characters. A procedure is identified with a file name and extension, for example, *filename*.fex.

Procedures can be simultaneously accessed and executed by all users. The device and directory must precede the procedure file name and extension for the server to locate the procedure. If you specify a disk and a directory, you must include both the file name and the file extension (.fex) when calling the procedure.

WebFOCUS searches for a procedure on the path of the WebFOCUS Reporting Server, using either EDAPATH or APP PATH logic.

Reference: Naming a Procedure Under UNIX

The following rules apply:

- File names can be up to 64 characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are not recommended or supported.
- The file extension must be .fex, since WebFOCUS searches for a procedure with that extension by default.

FOCUS Data Sources Under UNIX

FOCUS data sources contain data written in FOCUS format. All FOCUS data sources have a record length of 4096 and a fixed length record format. The maximum size of a FOCUS data source is 1G or 256K pages.

A FOCUS data source is identified with a file name and extension, for example, *filename*.foc. The name of a FOCUS data source usually matches the name of the corresponding Master File. For example, if the Master File is ledger.mas, then the FOCUS data source is ledger.foc. You can override this default with the USE command, described in *Accessing a FOCUS Data Source* on page 487.

WebFOCUS searches for a FOCUS data source on the platform with the WebFOCUS Reporting Server or subserver.

Reference: Naming a FOCUS Data Source Under UNIX

The following rules apply:

- The file name can be up to 64 characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are therefore not recommended or supported.
- The file extension must be .foc since WebFOCUS searches for an Access File with that extension by default.

External Indexes for FOCUS Data Sources Under UNIX

An external index is a FOCUS data source that contains index, field, and segment information for one or more specified FOCUS data sources. The external index is independent of its associated FOCUS data source and is used to improve retrieval performance. In UNIX, the external index is automatically allocated as a permanent file when it is created using REBUILD.

For more information, see the EDA Server documentation for UNIX.

Sequential Data Sources Under UNIX

WebFOCUS recognizes sequential data sources formatted in the following ways:

- Fixed-format, in which each field occupies a predefined position in the record.
- Free-format, also known as comma-delimited, in which fields can occupy any position in a record.

For details see the *Describing Data With WebFOCUS Language* manual.

WebFOCUS StyleSheets Under UNIX

WebFOCUS StyleSheets enable you to style and produce reports that highlight key information. With StyleSheets, you specify various stylistic characteristics of a report and style report components individually. You can also use StyleSheets to define a hyperlink from any reporting object.

You can create StyleSheets externally or within a report request. For details, see the *Creating Reports With WebFOCUS Language* manual.

WebFOCUS searches for a WebFOCUS StyleSheet on the path of the WebFOCUS Reporting Server, using either EDAPATH or APP PATH logic.

Reference: Naming a StyleSheet Under UNIX

The following rules apply:

- File names can be up to eight characters long. The first character should be a letter, and the remaining characters can be any combination of letters, numbers, and underscores. Other special characters may be used, but could cause problems and are not recommended or supported.
- The file extension must be .sty since WebFOCUS searches for a StyleSheet with that extension by default.

Profiles Under UNIX

WebFOCUS supports the following levels of profiles to provide flexibility in designing and running applications.

- A **global profile** is a startup file that is automatically created during the installation and configuration of the server. It contains the default environment settings that are required for the proper operation of the server.
- A **group profile** is a file used by the server to specify settings that apply to a server environment, but only for a user of a specific security group.
- A **user profile** is available, based on a user ID. A user profile is a file used by the server to specify settings that apply to a server environment, but only for a specific user ID.

Extract Files Under UNIX

WebFOCUS creates all extract files in a temporary directory on the WebFOCUS Reporting Server, or in a user-defined location on the WebFOCUS Reporting Server.

HOLD Files Under UNIX

A HOLD extract file is a sequential data source that contains the results of a report request. It may have a corresponding HOLD Master File. The extension for a HOLD file is .ftm unless you specify the FORMAT option. If a Master File is created, it has the same name as the extract file, with the extension .mas.

If you specify the FORMAT FOCUS option with HOLD, WebFOCUS creates an extract file and a Master File, each with the file name FOC$HOLD. These files are then used as input to the procedure that creates the final FOCUS file. The new FOCUS and Master Files are created in the user temporary directory.

Syntax: How to Create a HOLD File Under UNIX

```
ON TABLE HOLD [AS filename]
```

where:

filename

Is a name for the HOLD file. If you do not specify a file name, HOLD is used as the default name. Since each subsequent HOLD command overwrites the previous HOLD file, it is useful practice to code a distinct file name in each request to direct the extracted data to a separate file, preventing it from being overwritten.

For the complete syntax with all options, see the *Creating Reports With WebFOCUS Language* manual.

SAVB Files Under UNIX

A SAVB file is an extract file containing the results of a report request in internal format. That is, all numeric fields are stored in binary, and all character fields are padded with spaces to a multiple of four bytes. The file cannot be printed.

Syntax: How to Create a SAVB File Under UNIX

```
ON TABLE SAVB [AS filename]
```

where:

filename

Is the name of the file. The default is SAVB. The default extension is .ftm.

SAVE Files Under UNIX

A SAVE file is an extract file that saves the data of a report, but does not save headings, or subtotals, or create a Master File. It is a simple sequential character data file and can be used by other programs, or merged into another data file using a data maintenance request. The default format is simple character, although you can specify formats compatible with many other software products. WebFOCUS saves all columns in the report in printable, character format with no spaces between columns.

A SAVE file contains the external character format equivalent to SAVB. The command format and allocations are the same as SAVB. However, the numbers are printable Extended Binary Coded Decimal Interchange Code (EBCDIC) characters and no padding takes place.

Syntax: How to Create a SAVE File Under UNIX

```
ON TABLE SAVE [AS filename]
```

where:

filename

Is the name of the file. The default is SAVE. The default extension is .ftm.

HOLDMAST Files Under UNIX

A HOLDMAST file is a temporary Master File. It is created with the HOLD command. The APP HOLDMETA command specifies its location.

If the HOLD file was created under the default name HOLD, the FILEDEF command can be used in conjunction with the APP HOLDMETA syntax to designate a logical name to the HOLD file and respective Master File.

Syntax: How to Specify the Location for a HOLDMAST File Under UNIX

```
APP HOLDMETA appname
```

where:

appname

Is a valid application name.

Example: Allocating a Logical Name and Location for a HOLDMAST File Under UNIX

The following example sets a logical name with FILEDEF and specifies a location for a HOLDMAST file:

```
1. APP HOLDMETA app1
2. FILEDEF SALES DISK /tmp/sales.ftm
    TABLE FILE GGSALES
    PRINT GGSALES
    ON TABLE HOLD AS SALES
    END
```

The numbers listed next to the code correspond with the following notes:

1. Specifies the location of the Master File in the app1 application directory. WebFOCUS uses the same logical name, SALES, specified in the second line, to name the temporary Master File.
2. Specifies SALES as the logical name for the physical data source. The HOLD command creates the temporary Master File sales.mas and the data source sales.ftm.

Work Files Under UNIX

WebFOCUS creates work files as required by your application. WebFOCUS creates all extract files in a temporary directory on the WebFOCUS Reporting Server, or in a user-defined location on the WebFOCUS Reporting Server.

The available work files are:

- **FOCSTACK** files contain resolved Dialogue Manager procedures.
- **FOCSORT**, allocated as FOCSORT.FOC, may be required for sorting with the TABLE, TABLEF, and MATCH commands. The maximum size of a FOCSORT file is 1G of 256K pages.

- **FOCPOST** files hold sequential output, which is saved by the POST command and read by the PICKUP command.
- **FOCSML** files are used by the Financial Modeling Language (FML) facility.
- **OFFLINE** sends output to a file when the PRINT parameter is set to OFFLINE. Allocate the file to ddname OFFLINE using a FILEDEF command after the offline has been closed with the OFFLINE CLOSE command.

Example: Sending Output With the OFFLINE Command Under UNIX

The following command sends further report output to the file EMPL OUTPUT A with a fixed length record format and a record length of 80 bytes:

```
OFFLINE
FILEDEF OFFLINE DISK TMP/EMPL.OUT (RECFM F LRECL 80)
```

Determining If A File Exists Under UNIX

The STATE command sets the system variable &RETCODE to a value that indicates whether a specified file exists.

Syntax: How to Determine If a File Exists Under UNIX

```
[opsys] STATE filename
```

where:

opsys

Identifies the operating environment. For UNIX, the value is UNIX.

filename

Is the file whose existence you want to determine. It can be a file name in the current path, a full path name, or an application name and filename in the form appname/filename.

After issuing the command, the variable &RETCODE has the value zero (0) if the file exists and the value -1 if the file does not exist. If the file does not exist, the following message is generated:

```
File doesn't exist
```

Chapter 10

Euro Currency Support

The following topics describe how to create and use a currency data source to convert to and from the new euro currency.

In this chapter:

- Integrating the Euro Currency
- Converting Currencies
- Creating the Currency Data Source
- Identifying Fields That Contain Currency Data
- Activating the Currency Data Source
- Processing Currency Data
- Querying the Currency Data Source in Effect
- Punctuating Numbers
- Selecting an Extended Currency Symbol

Integrating the Euro Currency

With the introduction of the euro currency, businesses need to maintain books in two currencies, add new fields to the data source designs, and perform new types of currency conversions. You can perform currency conversions according to the rules specified by the European Union. To do this:

1. Create a currency data source with the currency IDs and exchange rates you will use. See *Creating the Currency Data Source* on page 693.
2. Identify fields in your data sources that represent currency data. See *Identifying Fields That Contain Currency Data* on page 696.
3. Activate your currency data source. See *Activating the Currency Data Source* on page 698.
4. Perform currency conversions. See *Processing Currency Data* on page 699.

Note: As the euro symbol becomes available to operating systems, Information Builders will support it.

Converting Currencies

Euro currency was introduced in Euroland on January 1, 2002, and on July 1, 2002 it became the only legal tender. All monetary transactions now occur in euro currency.

The European Union has set fixed exchange rates between the euro and the traditional national currency in each of the 12 adopting member nations. Although 12 or more currencies in the European Union use the euro, more than 100 currencies have a recognized status worldwide. In addition, you may need to define custom currencies for other applications.

While the exchange rates within Euroland remain fixed, exchange rates between the euro and non-euro countries continue to vary freely and, in fact, several rates may be in use at one time (for example, actual and budgeted rates).

You identify your currency codes and rates by creating a currency data source. For more information, see *Creating the Currency Data Source* on page 693.

Reference: Currency Conversion Rules

The European Union has established the following rules for currency conversions:

- The exchange rate must be specified as a decimal value, *r*, with six significant digits.

  This rate will establish the following relationship between the euro and the particular national currency:

  1 euro = *r* national units

- To convert from the euro to the national unit, multiply by *r* and round the result to two decimal places.
- To convert from the national currency to the euro, divide by *r* and round the result to two decimal places.
- To convert from one national currency to another, first convert from one national unit to the euro, rounding the result to three decimal places (your application rounds to exactly three decimal places). Then convert from the euro to the second national unit, rounding the result to two decimal places. This two-step conversion process is *triangulation*.

Example: Performing Triangulation

The following example illustrates triangulation. In this case, 10 US dollars (USD) are converted to French francs (FRF). The exchange rate for USD to euros (EUR) is 1.17249. The exchange rate for FRF to euros is 6.55957.

- The 10 USD are converted to EUR by dividing the 10 USD by the EUR exchange rate of 0.8840:

```
EUR = 10 / 0.8840
```

This results in 11.3122 euros.

- The euros are converted to FRF by multiplying the above result by the exchange rate of FRF for euros (6.55957):

```
FRF = 11.3122 * 6.55957
```

The result is 74.26. FRF. This means 74.26 FRF are equivalent to 10 USD.

Creating the Currency Data Source

Supply values for each type of currency you need.

You must supply the following values in your currency data source:

- A three-character code to identify the currency, such as USD for US dollars or BEF for Belgian francs. (For a partial list of recognized currency codes, see *Sample Currency Codes* on page 694.)
- One or more exchange rates for the currency.

There is no limit to the number of currencies you can add to your currency data source, and the currencies you can define are not limited to official currencies. Therefore, the currency data source can be fully customized for your applications.

The currency data source can be any type of data source your application can access (for example, FOCUS, FIX, DB2, or VSAM). The currency Master File must have one field that identifies each currency ID you will use and one or more fields to specify the exchange rates.

We strongly recommend that you create a separate data source for the currency data rather than adding the currency fields to another data source. A separate currency data source enhances performance and minimizes resource utilization because the currency data source is loaded into memory before you perform currency conversions.

Syntax: How to Create a Currency Data Source

```
FILE = name, SUFFIX = suffix,$
FIELD  = CURRENCY_ID,, FORMAT =  A3,                    [ACTUAL = A3 ,]$
FIELD  = rate_1,,      FORMAT = {D12.6|numeric_format1},[ACTUAL = A12,]$
   .
   .
   .
FIELD  = rate_n,,      FORMAT = {D12.6|numeric_formatn}, [ACTUAL = A12,]$
```

where:

name

Is the name of the currency data source.

suffix

Is the suffix of the currency data source. The currency data source can be any type of data source your application can access.

CURRENCY_ID

Is the required field name. The values stored in this field are the three-character codes that identify each currency, such as USD for U.S. dollars. Each currency ID can be a universally recognized code or a user-defined code.

Note: The code EUR is automatically recognized. You should *not* store this code in your currency data source. See *Sample Currency Codes* on page 694 for a list of common currency codes.

rate_1...rate_n

Are types of rates (such as BUDGET, FASB, ACTUAL) to be used in currency conversions. Each rate is the number of national units that represent one euro.

numeric_format1...numeric_formatn

Are the display formats for the exchange rates. Each format must be numeric. The recommended format, D12.6, ensures that the rate is expressed with six significant digits as required by the European Union conversion rules. Do not use Integer format (I).

ACTUAL A*n*

Is required only for non-FOCUS data sources.

Note: The maximum number of fields in the currency data source must not exceed 255 (that is, the CURRENCY_ID field plus 254 currency conversion fields).

Reference: Sample Currency Codes

On January 1, 1999, Euroland set exchange rates between the euro and other currencies. Countries included in Euroland as of that date are marked with an asterisk (*). The rates are fixed and will not change, although the rates for other countries change over time.

Currency Name	Currency Code	Rate
American dollar	USD	.974298

Currency Name	Currency Code	Rate
Austrian schilling	ATS	13.7603
Belgian franc*	BEF	40.3399
British pound	GBP	.625152
Canadian dollar	CAD	1.54504
Danish krone	DKK	7.42659
Dutch guilder*	NLG	2.20371
Deutsche mark*	DEM	1.95583
Euro	EUR	1
Finnish markka	FIM	5.94573
French franc*	FRF	6.55957
Greek drachma*	GRD	340.750
Irish pound*	IEP	0.787564
Italian lira*	ITL	1936.27
Japanese yen or Chinese yuan	JPY	118.377
Luxembourg franc*	LUF	40.3399
Norwegian kroner	NOK	7.34864
Portuguese escudo*	PTE	200.482
Spanish peseta*	ESP	166.386
Swedish krona	SEK	9.20906
Swiss franc	CHF	1/4634

Example: Specifying Currency Codes and Rates in a Master File

The following Master File for a comma-delimited currency data source specifies two rates for each currency, ACTUAL and BUDGET:

```
FILE = CURRCODE, SUFFIX = COM,$
FIELD = CURRENCY_ID,,    FORMAT = A3,    ACTUAL = A3 ,$
FIELD = ACTUAL, ALIAS =, FORMAT = D12.6, ACTUAL = A12 ,$
FIELD = BUDGET, ALIAS =, FORMAT = D12.6, ACTUAL = A12 ,$
```

The following is sample data for the currency data source defined by this Master File:

```
FRF,  6.55957,  6.50000,$
USD,  0.974298,  1.00000,$
BEF,  40.3399, 41.00000,$
```

Identifying Fields That Contain Currency Data

After you have created your currency data source, you must identify the fields in your data sources that represent currency values. To designate a field as a currency-denominated value (a value that represents a number of units in a specific type of currency) add the CURRENCY attribute to one of the following:

- The FIELD specification in the Master File.
- The left side of a DEFINE or COMPUTE.

Syntax: How to Identify a Currency Value

Use the following syntax to identify a currency-denominated value.

In a Master File

```
FIELD = currfield,, FORMAT = numeric_format, ..., CURR =
{curr_id|codefield} ,$
DEFINE currfield/numeric_format CURR curr_id = expression ;$
DEFINE FILE filename
currfield/numeric_format CURR curr_id = expression ;
END
COMPUTE currfield/numeric_format CURR curr_id = expression ;
```

In a DEFINE in the Master File

```
DEFINE currfield/numeric_format CURR curr_id = expression ;$
```

In a DEFINE FILE command

```
DEFINE FILE filename
currfield/numeric_format CURR curr_id = expression ;
END
```

In a COMPUTE command

```
COMPUTE currfield/numeric_format CURR curr_id = expression ;
```

where:

currfield

Is the name of the currency-denominated field.

numeric_format

Is a numeric format. Depending on the currency denomination involved, the recommended number of decimal places is either two or zero. Do not use I or F format.

CURR

Indicates that the field value represents a currency-denominated value. CURR is an abbreviation of CURRENCY, which is the full attribute name.

curr_id

Is the three-character currency ID associated with the field. In order to perform currency conversions, this ID must either be the value EUR or match a CURRENCY_ID value in your currency data source.

codefield

Is the name of a field, qualified if necessary, that contains the currency ID associated with *currfield*. The code field should have format A3 or longer and is interpreted as containing the currency ID value in its first three bytes. For example:

```
FIELD = PRICE,, FORMAT = P12.2C, ..., CURR = TABLE.FLD1,$
   .
   .
   .
FIELD = FLD1,, FORMAT = A3, ...,$
```

The field named FLD1 contains the currency ID for the field named PRICE.

filename

Is the name of the file for which this field is defined.

expression

Is a valid expression.

Example: Identifying a Currency-Denominated Field

The following Master File contains the description of a field named PRICE that is denominated in U.S. dollars.

```
FILE=CURRDATA,SUFFIX=COM,$
FIELD=PRICE, FORMAT=P17.2 , ACTUAL=A5, CURR=USD,$
.
.
.
```

Activating the Currency Data Source

Before you can perform currency conversions, you must specify the currency data source by setting the EUROFILE parameter with the SET command. By default, the EUROFILE parameter is not set.

Issue the SET command in a procedure or any supported profile. It cannot be set within a report request.

After a data source is activated, you can access a different currency data source by reissuing the SET command.

Note: The EUROFILE parameter must be set alone. For example, appending an additional SET parameter will cause the additional parameter setting to be lost.

Syntax: How to Activate Your Currency Data Source

```
SET EUROFILE = {ddname|OFF}
```

where:

ddname

Is the name of the Master File for the currency data source. The ddname must refer to a data source known to and accessible by your application in read-only mode.

OFF

Deactivates the currency data source and removes it from memory.

Reference: EUROFILE Error Messages and Notes

- Issuing the SET EUROFILE command when the currency data source Master File does not exist generates the following error message:

  ```
  (FOC205) THE DESCRIPTION CANNOT BE FOUND FOR FILE NAMED: ddname
  ```

- Issuing the SET EUROFILE command when the currency Master File specifies a FOCUS data source and the associated FOCUS data source does not exist generates the following error message:

```
(FOC036) NO DATA FOUND FOR THE FOCUS FILE NAMED: name
```

Processing Currency Data

After you have created your currency data source, identified the currency-denominated fields in your data sources, and activated your currency data source, you can perform currency conversions.

Each currency ID in your currency data source generates a virtual conversion function whose name is the same as its currency ID. For example, if you added BEF to your currency data source, a virtual BEF currency conversion function will be generated.

The euro function, EUR, is supplied automatically with your application. You do not need to add the EUR currency ID to your currency data source.

The result of a conversion is calculated with very high precision, 31 to 36 significant digits, depending on platform. The precision of the final result is always rounded to two decimal places. In order to display the result to the proper precision, its format must allow at least two decimal places.

Syntax: How to Process Currency Data

In a procedure

```
DEFINE FILE filename
result/format [CURR curr_id] = curr_id(infield, rate1 [,rate2]);
END
```

or

```
COMPUTE result/format [CURR curr_id] = curr_id(infield, rate1 [,rate2]);
```

In a Master File

```
DEFINE result/format [CURR curr_id] = curr_id(infield, rate1 [,rate2]);$
```

where:

filename

Is the name of the file for which this field is defined.

result

Is the converted currency value.

format

Is a numeric format. Depending on the currency denomination involved, the recommended number of decimal places is either two or zero. The result will always be rounded to two decimal places, which will display if the format allows at least two decimal places. Do not use an Integer or Floating Point format.

curr_id

Is the currency ID of the result field. This ID must be the value EUR or match a currency ID in your currency data source. Any other value generates the following message:

```
(FOC263)  EXTERNAL FUNCTION OR LOAD MODULE NOT FOUND: curr_id
```

Note: The CURR attribute on the left side of the DEFINE or COMPUTE identifies the result field as a currency-denominated value which can be passed as an argument to a currency function in subsequent currency calculations. Adding this attribute to the left side of the DEFINE or COMPUTE does not invoke any format or value conversion on the calculated result.

infield

Is a currency-denominated value. This input value will be converted from its original currency to the *curr_id* denomination. If the *infield* and *result* currencies are the same, no calculation is performed and the *result* value is the same as the *infield* value.

rate1

Is the name of a rate field from the currency data source. The *infield* value is divided by the *rate1* value of the currency to produce the equivalent number of euros.

If *rate2* is not specified in the currency calculation and triangulation is required, this intermediate result is then multiplied by the *result* currency *rate1* value to complete the conversion.

In certain cases, you may need to provide different rates for special purposes. In these situations you can specify any field or numeric constant for *rate1* as long as it indicates the number of units of the *infield* currency denomination that equals one euro.

rate2

Is the name of a rate field from the currency data source. This argument is only used for those cases of triangulation in which you need to specify different rate fields for the *infield* and *result* currencies. It is ignored if the euro is one of the currencies involved in the calculation.

The number of euros that was derived using *rate1* is multiplied by the *result* currency *rate2* value to complete the conversion.

In certain cases, you may need to provide different rates for special purposes. In these situations you can specify any field or numeric constant for *rate2* as long as it indicates the number of units of the *result* currency denomination that equals one euro.

Reference: Currency Calculation Error Messages

Issuing a report request against a Master File that specifies a currency code not listed in the active currency data source generates the following message:

```
(FOC1911) CURRENCY IN FILE DESCRIPTION NOT FOUND IN DATA
```

A syntax error or undefined field name in a currency conversion expression generates the following message:

```
(FOC1912)  ERROR IN PARSING CURRENCY STATEMENT
```

Example: Using the Currency Conversion Function

Assume that the currency data source contains the currency IDs USD and BEF, and that PRICE is denominated in Belgian francs as follows:

```
FIELD = PRICE, ALIAS=, FORMAT = P17.2, CURR=BEF,$
```

- The following example converts PRICE to euros and stores the result in PRICE2 using the BUDGET conversion rate for the BEF currency ID:

  ```
  COMPUTE PRICE2/P17.2 CURR EUR = EUR(PRICE, BUDGET);
  ```

- This example converts PRICE from Belgian francs to US dollars using the triangulation rule:

  ```
  DEFINE PRICE3/P17.2 CURR USD = USD(PRICE, ACTUAL);$
  ```

  First PRICE is divided by the ACTUAL rate for Belgian francs to derive the number of euros rounded to three decimal places. Then this intermediate value is multiplied by the ACTUAL rate for US dollars and rounded to two decimal places.

- The following example uses a numeric constant for the conversion rate:

  ```
  DEFINE PRICE4/P17.2 CURR EUR = EUR(PRICE,5);$
  ```

- The next example uses the ACTUAL rate for Belgian francs in the division and the BUDGET rate for US dollars in the multiplication:

  ```
  DEFINE PRICE5/P17.2 CURR USD = USD(PRICE, ACTUAL, BUDGET);$
  ```

Example: Converting U.S. Dollars to Euros, French Francs, and Belgian Francs

The following is an example of converting U.S. dollars to Euros, French Francs, and Belgian Francs.

1. Create a currency data source that identifies the currency and one or more exchange rates. (See *Creating the Currency Data Source* on page 693 for details.) The following sample data source is named CURRCODE:

```
FILE = CURRCODE, SUFFIX = COM,$
FIELD = CURRENCY_ID,,    FORMAT = A3,    ACTUAL = A3 ,$
FIELD = ACTUAL, ALIAS =, FORMAT = D12.6, ACTUAL = A12 ,$
FIELD = BUDGET, ALIAS =, FORMAT = D12.6, ACTUAL = A12 ,$
```

2. Create a data source that contains the values to be converted. (See *Identifying Fields That Contain Currency Data* on page 696 for details.) The following sample data source is named CURRDATA:

```
FILE=CURRDATA,SUFFIX=COM,$
FIELD=PRICE, FORMAT=P17.2 , ACTUAL=A5, CURR=USD,$
```

3. Create a request that uses the currency data source to convert the currency values contained in the data source containing these values. The following procedure converts PRICE to euros, French francs, and Belgian francs. The numbers on the left correspond to the notes explaining the code.

```
   -* THE FOLLOWING ALLOCATIONS ARE FOR RUNNING UNDER z/OS
1. -* DYNAM ALLOC FILE CURRCODE DA USER1.FOCEXEC.DATA(CURRCODE) SHR REU
2. -* DYNAM ALLOC FILE CURRDATA DA USER1.FOCEXEC.DATA(CURRDATA) SHR REU
    -* THE FOLLOWING ALLOCATIONS ARE FOR RUNNING UNDER WINDOWS NT
1.     FILEDEF CURRCODE DISK GGDEMO/CURRCODE.COM
2.    FILEDEF CURRCODE DISK GGDEMO/CURRDATA.COM
3. SET EUROFILE = CURRCODE
   DEFINE FILE CURRDATA
4. PRICEEUR/P17.2 CURR EUR = EUR(PRICE, ACTUAL);
    END
    TABLE FILE CURRDATA
    PRINT PRICE PRICEEUR AND COMPUTE
5. PRICEFRF/P17.2 CURR FRF = FRF(PRICE, ACTUAL);
    PRICEBEF/P17.2 CURR BEF = BEF(PRICE, ACTUAL);
    END
```

The report request executes as follows:

1. The FILEDEF or DYNAM command informs the operating system of the location of the CURRCODE data source.
2. The FILEDEF command informs the operating system of the location of the CURRDATA data source.

3. The SET command specifies the currency data source as CURRCODE.
4. This line calls the EUR function, which converts U.S. dollars to euros.
5. The next two lines are the conversion functions that convert euros into the equivalent in French and Belgian Francs.

The output is:

```
PRICE          PRICEEUR          PRICEFRF          PRICEBEF
-----          --------          --------          --------
 5.00              4.26             27.97            172.01
 6.00              5.12             33.57            206.42
40.00             34.12            223.78           1376.20
10.00              8.53             55.95            344.06
```

You cannot use the derived euro value PRICEEUR in a conversion from USD to BEF. PRICEEUR has two decimal places (P17.2), not three, as the triangulation rules require.

Querying the Currency Data Source in Effect

You can issue a query to determine what currency data source is in effect. To do this, issue ? SET ALL or ? SET EUROFILE.

Syntax: How to Determine the Currency Data Source in Effect

```
? SET EUROFILE
```

Example: Determining the Currency Data Source in Effect

Assume the currency data source is named CURRCODE.

If you issue the following commands

```
SET EUROFILE = CURRCODE
? SET EUROFILE
```

the output is:

```
EUROFILE              CURRCODE
```

Punctuating Numbers

Countries differ in how they punctuate numbers, and you can reflect these differences in your reports using Continental Decimal Notation (CDN) which is specified with the CDN SET parameter. The CDN SET allows you to choose to punctuate numbers with a combination of commas, decimals, spaces, and single quotation marks.

The CDN SET parameter can be used in a report request but is not supported in DEFINE or COMPUTE commands.

Note: The punctuation specified by the CDN parameter also determines the punctuation used in numbers affected by the CENT-ZERO SET parameter.

Syntax: How to Determine the Punctuation of Large Numbers

```
SET CDN = option
```

where:

option

Determines the punctuation used in numeric notation. The options are:

- **ON,** which uses CDN. For example, the number 3,045,000.76 is represented as 3.045.000,76.
- **OFF,** which turns CDN off. For example, the number 3,045,000.76 is represented as 3,045,000.76. This value is the default.
- **SPACE,** which separates groups of three significant digits with a space instead of a comma, and marks a decimal position with a comma instead of a period. For example, the number 3,045,000.76 is represented as 3 045 000,76.
- **QUOTE,** which separates groups of three significant digits with a single quotation mark (') instead of a comma, and marks a decimal position with a comma instead of a period. For example, the number 3,045,000.76 is represented as 3'045'000,76.
- **QUOTEP,** which separates groups of three significant digits with a single quotation mark (') instead of a comma, and marks a decimal position with period. For example, the number 3,045,000.76 is represented as 3'045'000.76.

Example: Displaying Numbers Using Continental Decimal Notation

The following table shows how 1234.56 is displayed, depending on the setting of CDN.

CDN Setting	Result
OFF	1,234.56
ON	1.234,56
SPACE	1 234,56
QUOTE	1'234,56

CDN Setting	Result
QUOTEP	1'234.56

Example: **Determining the Punctuation of Large Numbers**

In the following request, CDN is set to ON which punctuates numbers using a period to separate thousands, and a comma to separate decimals.

```
SET CDN = ON
TABLE FILE EMPLOYEE
PRINT LAST_NAME FIRST_NAME SALARYEND
```

The output is:

```
LAST_NAME       FIRST_NAME          SALARY
STEVENS         ALFRED          $11.000,00
SMITH           MARY            $13.200,00
JONES           DIANE           $18.480,00
JONES           DIANE           $17.750,00
BANNING         JOHN            $29.700,00
IRVING          JOAN            $26.862,00
IRVING          JOAN            $24.420,00
ROMANS          ANTHONY         $21.120,00
MCCOY           JOHN            $18.480,00
BLACKWOOD       ROSEMARIE       $21.780,00
MCKNIGHT        ROGER           $16.100,00
MCKNIGHT        ROGER           $15.000,00
CROSS           BARBARA         $27.062,00
CROSS           BARBARA         $25.775,00
```

Selecting an Extended Currency Symbol

You can select a currency symbol for display in report output regardless of the default currency symbol configured for National Language Support (NLS). Use the extended currency symbol format in place of the floating dollar (M) or non-floating dollar (N) display option. When you use the floating dollar (M) or non-floating dollar (N) display option, the currency symbol associated with the default code page is displayed. For example, when you use an American English code page, the dollar sign is displayed.

Note: You can use the SET CURRSYMB command to control which symbol displays for the M and N options.

The extended currency symbol format allows you to display a symbol other than the dollar sign. For example, you can display the symbol for a United States dollar, a British pound, a Japanese yen, or the euro. Extended currency symbol support is available for numeric formats (I, D, F, and P).

Use the following character combinations as the final two characters in any numeric display format:

Display Option	Description	Example
!d	Fixed dollar sign.	D12.2!d
:d	Fixed dollar sign.	D12.2:d
!D	Floating dollar sign.	D12.2!D
:D	Floating dollar sign.	D12.2:D
!e	Fixed euro symbol.	F9.2!e
:e	Fixed euro symbol.	F9.2:e
!E	Floating euro symbol on the left side.	F9.2!E
:E	Floating euro symbol on the left side.	F9.2:E
!F	Floating euro symbol on the right side.	F9.2!F
:F	Floating euro symbol on the right side.	F9.2:F
!l	Fixed British pound sign.	D12.1!l
:l	Fixed British pound sign.	D12.1:l
!L	Floating British pound sign.	D12.1!L
:L	Floating British pound sign.	D12.1:L
!y	Fixed Japanese yen symbol.	I9!y
:y	Fixed Japanese yen symbol.	I9:y
!Y	Floating Japanese yen symbol.	I9!Y
:Y	Floating Japanese yen symbol.	I9:Y

Note: The colon (:) is equivalent to the exclamation point (!), however, only the colon is invariant across code pages, so using the colon is recommended.

Reference: **Extended Currency Symbol Formats**

The following guidelines apply:

- A format specification cannot be longer than eight characters.
- The extended currency option must be the last option in the format.
- The extended currency symbol format cannot include the floating (M) or non-floating (N) display option.
- A non-floating currency symbol is displayed only on the first row of a report page. If you use field-based reformatting (as in the example that follows) to display multiple currency symbols in a report column, only the symbol associated with the first row is displayed. In this case, do not use non-floating currency symbols.
- Lowercase letters are transmitted as uppercase letters by the terminal I/O procedures. Therefore, the fixed extended currency symbols can only be specified in a procedure.
- Extended currency symbol formats can be used with fields in floating point, decimal, packed, and integer formats. Alphanumeric, dynamic, and variable character formats cannot be used.

Syntax: **How to Select an Extended Currency Symbol**

numeric_format{:|!}*option*

where:

numeric_format

Is a valid numeric format (data type I, D, F, or P).

{:|!}

Either : or ! is required, but only the colon is invariant across code pages.

option

Determines the currency symbol that is displayed, and whether the symbol is floating or non-floating. Possible values are:

d displays a non-floating dollar sign.

D displays a floating dollar sign.

e displays a non-floating euro symbol.

E displays a floating euro symbol on the left side.

F displays a floating euro symbol on the right side.

l displays a non-floating British pound sterling symbol.

L displays a floating British pound sterling symbol.

y displays a non-floating Japanese yen symbol.

Y displays a floating Japanese yen symbol.

Example: Displaying Extended Currency Symbols

The following request displays the euro symbol:

```
SET PAGE-NUM = OFF
TABLE FILE CENTORD
PRINT PRODNAME QUANTITY PRICE/D10.2:E
BY ORDER_DATE
WHERE QUANTITY GT 700;
ON TABLE SET STYLE *
TYPE = REPORT, GRID = OFF,$
ENDSTYLE
END
```

The output is:

Date Of Order:	Product Name:	Quantity:	Price:
2001/10/16	R5 Micro Digital Tape Recorder	726	€89.00
	ZT Digital PDA - Commercial	726	€499.00
2002/03/20	R5 Micro Digital Tape Recorder	751	€89.00
	ZT Digital PDA - Commercial	751	€499.00
2002/04/03	ZC Digital PDA - Standard	751	€299.00
2002/06/07	R5 Micro Digital Tape Recorder	751	€89.00
	ZT Digital PDA - Commercial	751	€499.00
	ZC Digital PDA - Standard	751	€299.00
	R5 Micro Digital Tape Recorder	702	€89.00
	ZT Digital PDA - Commercial	702	€499.00
	ZC Digital PDA - Standard	702	€299.00
2002/06/18	R5 Micro Digital Tape Recorder	751	€89.00
	ZT Digital PDA - Commercial	751	€499.00
2002/10/16	R5 Micro Digital Tape Recorder	798	€89.00
	ZT Digital PDA - Commercial	798	€499.00
2002/12/19	2 Hd VCR LCD Menu	701	€179.00

Index

-? command *428*

-? SET SETCOMMAND &myvar *332*

-? TSO DDNAME command *681*

-* command *428*

-CLOSE command *352*, *429*

-CMS command *417*

-DEFAULT command *339*, *429*

-DEFAULTH *195*

-DEFAULTH command *339*, *430*

-DOS command *417*, *430*

-EXIT command *389*, *390*, *430*

-GOTO command *392*, *393*, *431*

-HTMLFORM command *231*, *235*, *407*–*411*, *431*

    embedding files *410*

-IF ... GOTO command *393*–*397*

-IF command *432*

-IF tests *354*–*357*

-INCLUDE command *401*–*403*, *433*

    nesting procedures *405*

-label command *433*

-MVS command *417*, *434*

-PASS command *420*, *434*

-QUIT command *389*, *391*, *435*

-QUIT FOCUS command *389*, *391*, *435*

-READ command *346*, *436*

-READFILE command *351*, *436*

-REPEAT command *397*–*399*, *437*

-RUN command *389*, *438*

-SET IN FILE phrase *340*

-TSO command *417*, *439*

-TYPE command *420*, *421*, *439*

-UNIX command *417*, *440*

-VMS command *417*, *440*

-WINNT command *417*, *440*

-WRITE command *346*, *441*

? && command *467*, *468*

? COMBINE command *446*

? DEFINE command *447*

? FDT command *449*, *450*

? FILE command *451*, *452*

? FILEDEF command *453*, *454*, *657*

? FUNCTION command *454*

? HOLD command *454*, *455*

? JOIN command *455*

? n command *457*

? NLS command *456*

? PATH command *457*, *458*

? RELEASE command *458*

? REMOTE command *459*

? SET command *459*

? SET EUROFILE command *448*, *703*

? SET GRAPH command *461*

? STAT command *463*

? STYLE command *464*

? SU command *466*

? TSO DDNAME command *677*

? TSO DDNAME ddname command *678*, *679*, *681*

? USE command *467*, *499*

?F command *448*, *449*
?FF command *451*
.EVAL suffix *354*
.EXIST suffix *354*
.LENGTH suffix *354*
.TYPE suffix *354*
&ECHO variable *421*, *422*, *563*
&FOCCODEPAGE variable *379*
&FOCLANGCODE variable *380*
&IORETURN variable *421*
&myvar *332*
&RETCODE variable *421*, *426*
&STACK variable *421*, *426*
%STRICTMATH parameter *508*, *629*

3D parameter *512*

A

ACCBLN parameter *508*, *529*
ACCEPT attribute *590*
Access Files *29*, *659*
    identifying *468*, *470*
    MVS *670*
    UNIX *684*
    WHENCE command *468*, *470*
    Windows *659*
ACCESSHTML parameter *36*, *519*, *530*
ACCESSIBLE parameter *36*, *520*, *530*
accessing APP directories on Windows *97*
accessing FOCUS data sources *489*, *491*
accessing resources *580*
ACCESSPDF parameter *36*, *520*, *531*
ACROSSLINE parameter *520*, *531*
ACROSSPRT parameter *519*, *532*
ACROSSTITLE parameter *520*, *533*
ACRSVRBTITL parameter *533*
activating currency data source *698*
activating filters *577*
ad hoc reports *162*
    enabling *163*
aggregation *573*
ALL parameter *516*, *534*
allocating data files *128*
allocating files in MVS *677*
    Access Files *670*
    data sources *671*, *672*
    displaying *677*
    external indexes *672*
    Master Files *669*
    WebFOCUS StyleSheets *672*, *673*
allocating temporary files *133*
allocating WebFOCUS files *651*
ALLOWCVTERR parameter *516*, *535*
alternate file specifications *492*, *493*
    USE command *492*
    Use tool *492*
amper variables in functions *384*
amper variables
    quote-delimited *365*, *366*
APP ? METALOCATION_SAME command *98*

APP APPENDPATH command *95*
APP application management commands *91*
APP commands *90*
    APP ? METALOCATION_SAME *98*
    APP APPENDPATH *93*, *95*
    APP COPY *102*
    APP COPYF *102*
    APP CREATE *91*
    APP DELETE *91*, *105*
    APP DELETEF *105*
    APP FI *116*
    APP HELP *124*
    APP HOLD *114*
    APP LIST *118*
    APP MAP *96*
    APP MOVE *103*
    APP MOVEF *104*
    APP PATH *93*, *94*
    APP PREPENDPATH *93*, *94*
    APP PROPERTY CODEPAGE *106*
    APP QUERY *119*
    APP RENAME *91*, *107*
    APP RENAMEF *107*
    APP SET METALOCATION_SAME *98*
    APP SHOWPATH *98*
    FI *112*
    file types *108*
    HOLD *112*
    HOLDDATA *112*, *115*
    HOLDMETA *112*, *115*
APP COPY command *102*
APP COPYF command *102*, *108*
APP DELETE command *105*
APP DELETEF command *105*, *108*
APP FI command *112*, *116*
APP HELP command *92*, *124*
APP HOLD command *112*, *114*
APP HOLDDATA command *112*, *115*
APP HOLDMETA command *112*, *115*
APP LIST command *92*, *118*
APP MAP command *96*
APP MOVE command *103*
APP MOVEF command *104*, *108*
APP PATH command *78*, *94*
    configuring *94*
    search paths *94*
APP PATH logic *148*
APP PREPENDPATH command *94*
APP PROPERTY CODEPAGE command *106*
APP QUERY command *92*, *119*
APP RENAME command *107*
APP RENAMEF command *107*, *108*
APP reporting and help commands *92*
APP SET METALOCATION_SAME command *98*
APP SHOWPATH command *98*
appending applications to search path *95*
application (APP) *38*
    physical location of *38*
application components *37*, *38*
    listing *119*

application directories *60*, *78*, *81*, *96*, *99*
creating *99*
filtering in Web Console *81*
mapping from Web Console *60*
mapping manually *96*
application files *652*, *658*
Access Files *659*, *684*
data sources *660*, *661*, *685*, *686*
external indexes *661*, *672*, *686*
function libraries *662*
Master Files *658*
MVS *667*, *669*
procedures *660*, *684*
profiles *662*, *673*, *687*
UNIX *651*, *683*
WebFOCUS StyleSheets *662*, *672*, *686*
webpages *662*
Windows *658*
application logic *27*
Application Namespace *489*
application partitioning *28*
application paths *93*
APP PATH *78*
APP PATH command *94*
configuring *93*
application processing *31*
application repositories *37*, *38*
applications *27*, *105*
controlling memory *515*
creating *33*
applications *27*, *105*
debugging *33*
deleting *105*
managing flow of control *329*
optimizing *515*
partitioning *28*, *31*
processing *31*
WebFOCUS components *31*
approot parameter *38*
APPROOT variable *377*
AREXPIRE parameter *516*, *536*
ARGRAHENGIN parameter *512*, *536*
ARGRAPHENGIN parameter *536*
ARPASSWORD parameter *516*, *537*
AS phrase *537*
ASNAMES parameter *508*, *516*, *537*
AUTODRILL parameter *520*, *538*
AUTOFIT parameter *513*, *520*, *539*
AUTOINDEX parameter *515*, *539*
AUTOINDEX variable *378*
automatic scaling *639*
AUTOPATH parameter *515*, *540*
Autoprompt *222*
launch page templates *222*
AUTOSTRATEGY parameter *515*, *540*
AUTOTABLEF parameter *516*, *541*
AUTOTICK *586*
AUTOTICK parameter *512*, *541*

B

bar chart summary values *541*
BARNUMB parameter *512*, *541*
baseapp directory *39*
BASEURL parameter *512*, *519*, *542*
basic linear regression *585*
BINS parameter *515*, *542*
blank field values *529*
blank lines, suppressing *566*
BLANKINDENT parameter *519*, *542*
BOTTOMMARGIN parameter *519*, *543*
branching procedures *392*, *393*
    conditional *393*
    unconditional *392*
BSTACK parameter *512*
BUSDAYS parameter *510*, *516*, *544*
BYDISPLAY parameter *519*, *544*
BYPANEL parameter *519*, *545*

C

cache *297*
    clearing *297*
calculations *507*
    controlling with SET parameters *507*
CARTESIAN parameter *516*, *546*
Cascading Style Sheets (CSS) *298*, *552*, *594*
case conversion *576*
case sensitive searches *294*, *295*
CDN (Continental Decimal Notation) *703–705*
    punctuating large numbers *703*, *704*
CDN parameter *507*, *516*, *547*
CENT-ZERO parameter *516*, *519*, *548*
clearing a logical name *658*
CNOTATION parameter *548*
codepage, application *106*
collapsing PRINT with ACROSS *532*
COLLATION parameter *516*, *549*
column notation *548*
column spacing *628*
columns, reporting on *473*
combining fields *632*
comma-delimited files *616*
command statistics *463*
    ? STAT *463*
commands *329*, *501*
    in Dialogue Manager *329*, *331*
    ON GRAPH SET *506*
    operating system *417*, *419*
    query commands *444*
    stacked *332*
    testing results *421*, *426*
comments *335*
    adding *335*
COMPMISS parameter *550*
compound -IF tests *397*
COMPOUND parameter *519*, *550*
COMPUTE parameter *507*, *515*, *516*
concatenating FOCUS data sources *496*
    USE command *496–499*
    Use tool *496–499*

concatenating variables *373*
SET command *373*
conditional branching *393*
-IF ... GOTO command *393–396*
compound -IF tests *397*
screening values *354*
configuration files *38*
configuring application paths *93*
Continental Decimal Notation (CDN) *703–705*
punctuating large numbers *703*, *704*
controlling calculations *507*
controlling memory *515*
converting case *576*
converting currencies *692*, *699*, *701*, *702*
currency data source *692*
error messages *701*
converting currency *570*
COUNT fields *552*
COUNTWIDTH parameter *508*, *552*
creating a currency data source *694*
creating data files *128*
creating forms *155*
creating GIF files *583*
creating procedure files *127*
CSS (Cascading Style Sheets) *298*, *552*
CSSURL parameter *519*, *552*
currencies *691*
converting *692*
CURRENCY attribute *696*, *697*
currency codes *694*
currency conversion function *701*
currency conversions *570*, *692*
currency data source *448*, *692*
activating *698*
creating *693*, *694*, *696*
CURRENCY attribute *696*, *697*
currency codes *693*, *694*, *696*
querying *703*
currency symbols *705*
extended currency symbols *707*
CURRENCY_DISPLAY parameter *552*
CURRENCY_ISO_CODE parameter *553*
CURRENCY_PRINT_ISO parameter *553*
currency-denominated fields *697*
currency-denominated values *696*
CURRSYMB parameter *554*
CURSYM_D parameter *554*
CURSYM_E parameter *555*
CURSYM_F parameter *555*
CURSYM_G parameter *556*
CURSYM_L parameter *556*
CURSYM_Y parameter *556*
customizing WebFOCUS environment *501*

D

data *508*
access logic *27*
processing *508*
storing *508*

date format fields *558*
    zero values *558*
date functions *558*
DATE variable *378*
DATE_ORDER parameter *557*
DATE_SEPARATOR parameter *557*
date-time parameters *559*
DATEDISPLAY parameter *510*, *516*, *558*
DATEfmt variable *378*
DATEFNS parameter *510*, *516*, *558*
DATEFORMAT parameter *508*, *559*
dates *510*
DATETIME parameter *510*, *516*, *559*
DB_INFILE parameter *517*, *560*
DBA security rules *618*
DBACSENSITIV parameter *561*
DBAJOIN parameter *560*
DBASOURCE parameter *524*, *561*
ddnames *453*, *654*
    displaying *453*, *454*, *657*
deactivating filters *577*
debugging procedures *444*
    query commands *444*
default file names *577*
default format for HOLD files *591*
default pages *225*
default variable values *193*, *339*
    -DEFAULT command and *193*, *195*, *339*
DEFCENT parameter *510*, *516*, *562*
DEFECHO parameter *423*, *516*, *563*
DEFINE FUNCTIONs, reporting on *474*
defined functions *454*
DEFINES parameter *508*, *515*, *563*
defining files *654*, *668*
    MVS *668*
    Windows *654*
defining WebFOCUS files *651*
dense ranking *623*
designating a network drive on Windows *97*
designing user interfaces *147*
    coding *147*
Dialogue Manager *147*, *329*, *330*
    calling procedures *401*
    comments *335*
    creating procedures *335*
    debugging procedures *421*
    looping procedures *397*
    messages *420*
    operating system commands *417*, *419*
    passwords *420*
    reading files *346*
    stacked commands *332*, *390*
    testing for data sets *682*
    variables *369*
    webpages *407*
    writing files *346*
DIRECTHOLD parameter *564*
display formats *259*, *262*
displaying command statistics *463*
displaying currency data source *448*

displaying data sources with USE *467*
displaying dates *510*
displaying defined functions *454*
displaying Dialogue Manager values *467*
displaying extended currency symbols *708*
displaying fields *446*
? COMBINE command *446*
? DEFINE command *447*
?F command *448*
displaying graph parameters *461*
displaying grid for object placement evaluation *602*
displaying leading zeros *548*, *603*
displaying multiple graphs *585*
displaying parameter settings *459*
displaying product release number *458*
displaying remote server values *459*
displaying reports *232*
multiple reports on an existing HTML page *237*, *245*
multiple reports on an inline HTML page *239*
on an existing HTML page *232*, *233*, *241*
on an inline HTML page *235*
distributed processing *31*
DMH_LOOPLIM parameter *564*
DMH_STACKLIM parameter *565*
DMPRECISION parameter *342*, *507*, *565*
DMPRECISION
rounding *342*
DMY variable *378*
DMYY variable *379*
drill-down reports *279*, *281*
drilling down *538*
drilling down remotely *579*
drop-down lists *157*, *159*
dynamic *157*
DROPBLNKLINE parameter *566*
DTSTRICT parameter *508*, *566*
DUPLICATECOL parameter *519*, *521*
DYNAM command *129*
DYNAM SET LONGSNYM *135*
dynamic drop-down lists *157*
dynamically defining files *654*
MVS *668*

E

ECHO variable *379*
edaserve.cfg configuration file *38*
embedding images in .htm file *594*
EMPTYREPORT parameter *516*, *568*
encoding HTML data *595*
EQTEST parameter *509*, *568*
error files, reporting on *475*
error messages *457*, *569*
? n command *457*
displaying explanations *457*
ERROROUT parameter *516*, *569*
ESTRECORDS parameter *515*, *516*, *570*
euro currency *691*
converting *692*, *699*, *701*, *702*

EUROFILE parameter *508*, *570*, *698*
error messages *698*
restrictions *698*
evaluating variables *355*, *358*, *359*
EX procedures *139*
Excel 2000 *261*
Excel requests *571*
EXCELSRVURL parameter *517*, *570*
EXEC command *343*, *406*
calling procedures *406*
Execute component *343*, *406*
EXL2K output *627*
SHOWBLANKS *627*
EXL2KLANG parameter *571*
EXL2KTXTDATE parameter *511*, *572*
EXPANDABLE parameter *521*, *572*
EXPANDBYROW parameter *522*, *572*
EXPIRE_REPORTS variable *304*, *305*
EXTAGGR parameter *517*, *573*
extended currency symbols *705*, *707*
displaying *708*
formats *707*
EXTENDNUM parameter *574*
external Cascading Style Sheets *552*
external files *346*
closing *352*
reading from *346*
writing to *346*
external indexes for FOCUS data sources *661*
MVS *672*
external indexes for FOCUS data sources *661*
UNIX *686*
Windows *661*
external sorting *575*
external sorts *573*, *632*
EXTHOLD parameter *517*, *573*
extract files *652*, *663*
HOLD *663*, *673*, *687*
HOLDMAST *665*, *675*, *688*
HOLDSTAT *675*
MVS *667*, *673*
SAVB *664*, *674*, *688*
SAVE *664*, *674*, *688*
temporary Master Files *665*, *675*, *688*
UNIX *651*, *687*
Windows *663*
EXTRACT parameter *574*
EXTSORT parameter *517*, *575*

F

field name attributes *537*
FIELDNAME parameter *508*, *516*, *575*
fields *446*
?FF command *451*
displaying *446*
FILCASE parameter *576*
file attributes *654*
Dialogue Manager commands and *679*
displaying *678*
under MVS *677*

file directory table *449*
? FDT command *449*
file locations *29*
file names in -INCLUDE *403*
FILE parameter *577*
file specifications *488*
file types *29*
FILECOMPRESS parameter *517*, *576*
FILEDEF command *128*, *129*, *132*, *654*, *656*, *657*
clear *658*
clearing assignments *658*
displaying assignments *657*
Windows *654*
FILEDEF
CONCAT *130*
FILENAME parameter *516*, *577*
files *29*
files, reporting on *476*
FILTER parameter *516*, *577*
filtering application directories in Web Console *81*
filters *599*
maintaining across joins *599*
Financial Modeling Language (FML) *665*, *689*
FOCSML files and *665*, *689*
FIXRETRIEVE parameter *515*, *578*
FLOATMAPPING parameter *578*
FMI *470*
FML (Financial Modeling Language) *665*, *689*
FOCSML files and *665*, *689*
FML rows *581*
FOC$HOLD files *687*
FOC144 parameter *518*, *579*
FOC441 warnings *642*
foccache directory *39*
FOCEXURL parameter *512*, *579*
FOCEXURL variable *379*
FOCFEXNAME variable *379*
FOCFIRSTPAGE parameter *519*, *579*
FOCFOCEXEC variable *379*
FOCHTMLURL parameter *509*, *580*
FOCHTMLURL variable *380*
FOCINCLUDE variable *380*
FOCMODE variable *380*
FOCNEXTPAGE variable *380*
FOCPOST files *665*, *676*, *689*
Windows *665*
FOCQUALCHAR variable *381*
FOCREL variable *381*
FOCSECGROUP variable *381*
FOCSECGROUPS variable *381*
FOCSECUSER variable *381*
FOCSML files *665*, *689*
FML and *665*, *689*
MVS *676*
Windows *665*
FOCSORT files *665*, *676*, *689*
Windows *665*
FOCSTACK files *665*, *676*, *689*
Dialogue Manager and *665*, *676*, *689*
Windows *665*

FOCSTACK parameter *515*, *580*
FOCTRANSFORM parameter *509*, *581*
FOCUS data sources *487*, *660*, *661*, *672*, *686*
  accessing *487–489*, *491*
  alternative file specifications *492*, *493*
  concatenating *496–499*
  file specifications *488*
  identifying *488*
  MVS *671*
  new *494*
  protecting *495*
  querying *499*
  UNIX *685*
  USE command and *494*
  Use tool and *494*
  Windows *660*
FOCUS Database Server *466*
FOCUS HOLD files *564*
FOCUS Metadata Interface *470*
formatting currency *705*
formatting report output *611*, *615*
formatting reports *298*
  Cascading Style Sheets and *298*
forms *155*
FORMULTIPLE parameter *516*, *518*, *581*
function libraries *662*
  Windows *662*
functions and amper variables *384*
functions, reporting on *482*

G

GIF files *583*
global profiles *662*, *673*, *687*
global variable values *467*
global variables *369*, *370*, *372*
  naming *370*, *371*
graph parameters *461*
graph request *506*
graph style *604*
graph width *585*
GRAPHDEFAULT parameter *582*
GRAPHEDIT parameter *512*, *582*
GRAPHENGINE parameter *512*, *582*
graphs *512*
  displaying *512*
  processing *512*
  tick mark intervals *541*
GRAPHSERVURL parameter *512*, *583*
GRID parameter *512*, *583*
GRMERGE parameter *513*, *584*
GRMULTIGRAPH parameter *513*, *585*
group profiles *673*, *687*
GRTREND parameter *585*
GRWIDTH parameter *513*, *585*

H

HAUTO parameter *512*, *586*
HAXIS parameter *512*, *586*
HCLASS parameter *512*, *586*
HDAY parameter *510*, *587*

HIDENULLACRS parameter *522*, *587*
HISTOGRAM parameter *512*, *588*
histograms *593*
HLDCOM_TRIMANV parameter *588*
HMAX parameter *512*, *589*
HMIN parameter *512*, *589*
HNODATA parameter *516*, *589*
HOLD fields *454*
HOLD files *573*, *663*, *673*
  comments *592*
  DBA information *592*
  default format *591*
  displaying fields *591*
  MVS *673*
  UNIX *687*
  Windows *663*
HOLD Master File *537*, *590*
HOLDATTR parameter *516*, *590*
HOLDFORMAT parameter *508*, *591*
HOLDLIST parameter *508*, *591*
HOLDMAST files *665*, *675*
  MVS *675*
  UNIX *688*
  Windows *665*
HOLDMISS parameter *508*, *592*
HOLDSTAT files *675*, *676*
  MVS *675*, *676*
HOLDSTAT parameter *508*, *592*
holiday file *587*
horizontal axis *589*
horizontal axis width *586*
horizontal interval mark *586*
horizontal scaling *586*
HSTACK parameter *512*, *593*
HTICK parameter *512*, *593*
HTML active reports *536*
  expiration date *536*
  passwords *537*
HTML Autoprompt *184*, *222*, *225*
  customizing launch pages *222*
  descriptions *225*
  dynamic multiselect values *201*, *203*
  dynamic single-select values *197*
  launch page templates *222*
  multiselect values *200*, *204–206*
  processing All Values *205*
  processing Select All *205*
  QUOTEDSTRING *204*
  range of values *192*
  setting default variables *193*
  setting hidden variables *195*
  static multiselect values *201*
  static single-select values *196*
  variable descriptions *191*
HTML Autoprompti
  multiselect values *205*
HTML Autoprompting *187*
HTML display page *594*
HTML display pages
  creating *319*

HTML files *231*
displaying multiple reports *237*, *239*, *245*
inline pages *235*, *239*
HTML forms
creating *321*
HTML output *627*
SHOWBLANKS *627*
STYLEMODE parameter *630*
HTML pages *298*
Cascading Style Sheets and *298*
HTML pop-up descriptions *619*
HTML reports *411*
embedding multiple files in *411*
JavaScript files *598*
VBScript files *598*
HTML variables *184*, *222*
adding quotes *204*
generating FOC_NONE *205*
generating Select All *206*
selecting FOC_NONE *205*
HTML5 graphs *536*
HTMLARCHIVE parameter *509*, *593*
HTMLCSS parameter *519*, *594*
HTMLEMBEDIMG parameter *594*
HTMLENCODE parameter *595*
HTMTABLE format *410*
hyperlinks *147*, *150*, *270*

I

ibisamp directory *39*
IBIWF_mframename variable *267*
IBIWF_mprefix variable *267*
IBIWF_mreports variable *267*
IBIWF_msgviewer parameter *426*
identifying currency values *696*
identifying files *468*
WHENCE command *468–470*
identifying FOCUS data sources *487–489*, *491*, *494*, *499*
using Application Namespace *489*
images
embedding in .htm file *594*
impact analysis, reporting on *478*
IN FILE in Dialogue Manager *340*
Include component *401*
INDEX parameter *595*
indexed fields *539*
indexed variables *364*, *365*
creating *364*
indexes, reporting on *479*
indexing *595*
integrating euro currency *691*
internal blanks *627*
internal Cascading Style Sheets *594*
invalid date formats *535*

J

JavaScript *147*, *165*, *279*, *280*
JavaScript files *598*
JavaScript functions *280*

job termination *569*
JOIN CLEAR command *599*
JOIN command *599*
    maintaining filters *599*
join structures *455*
    displaying *455*
JOIN_LENGTH_MODE parameter *518*, *596*
joining data sources *597*
joining fields *597*
JOINLM parameter *518*, *596*
JOINOPT parameter *516*, *597*
JSURL parameter *516*, *598*

K

KEEPDEFINES parameter *508*, *516*, *599*
KEEPFILTERS SET parameter *599*
key fields *540*
keyed retrieval *578*
keys, reporting on *480*

L

lagging values *597*
LANG parameter *516*, *600*
launch pages *222*, *259*
    coding a FOCEXEC *222*
    customizing *259*
    settings in Administration Console *222*
LAYOUTGRID parameter *519*, *602*
LAYOUTRTL parameter *602*
leading blanks *627*
leading zeros *548*, *603*
LEADZERO parameter *510*, *516*, *603*
LEFTMARGIN parameter *519*, *603*
limiting record retrieval *623*
linear regression *585*
LINES parameter *289*, *519*, *603*
LIST requests *546*
local variables *369–371*, *376*
    naming *370*, *371*
locating a Master File *659*
LOCATION attribute *492*
logical names *654*, *658*
    FILEDEF command and *654*, *656*
    Windows *654*
long field names *575*
LONGSYNM *135*
LOOKGRAPH parameter *512*, *604*
looped procedures *397*
    -REPEAT command *397–399*
    SET command *397*, *400*
LRECL *610*

M

manipulating dates *510*
mapping application directories *60*
    manually *96*
Master Files *28*, *29*, *132*, *492*, *658*
    comments *592*
    DBA information *592*
    HOLDMAST files *665*, *675*, *688*

Master Files *28*, *29*, *132*, *492*, *658*
identifying *468*, *469*
locating *659*
MVS *669*
naming *659*
OCCURS attribute *605*
SAVEDMASTERS parameter *625*
TITLE parameter *635*
UNIX *683*
WHENCE command *468*, *469*
Windows *658*
MATCHCOLUMNORDER parameter *604*
maximum report panel line width *615*
MAXLRECL parameter *605*
MDI builds *607*
MDI files *606*
MDICARDWARN parameter *509*, *606*
MDIENCODING parameter *510*, *606*
MDIPROGRESS parameter *510*, *607*
MDY variable *381*
MDYY variable *381*
memory *515*
controlling *515*
menu options *305*
customizing *305*, *306*, *308*, *310*, *311*, *316*, *317*
MESSAGE parameter *607*
Message Viewer *426*
messages *420*
sending *420*, *421*
metadata *126*, *508*
accessing *126*
processing *508*
storing *508*
Microsoft Internet Explorer (IE) *260*
MIME (Multipurpose Internet Mail Extensions) *259*, *260*
missing data characters *589*
missing field values *550*
missing report data *609*
missing segment instances *534*
MISSING=ON attribute *589*
MISSINGTEST parameter *508*, *608*
MULTIPATH parameter *516*, *608*
multiple reports *266*, *267*
Multipurpose Internet Mail Extensions (MIME) *259*, *260*
multiselect list boxes *157*, *159*
MVS *667*
application files *667*, *669*
data sets *681*
data sources *672*
dynamically defining files *668*
estimating data set size *682*
external indexes *672*
extract files *667*, *673*
profiles *673*
referencing files *667*
temporary Master Files *675*
WebFOCUS files *667*

MVS *667*
WebFOCUS StyleSheets *672*
work files *667*, *676*

N

National Language Support (NLS) *456*
? NLS command *456*
navigating procedures *392*
branching *392*, *393*
NEW parameter *494*
NLS (National Language Support) *456*
? NLS command *456*
NODATA parameter *516*, *609*
non-data files *128*
NULL parameter *508*, *609*
numbering output pages *613*
numeric amper variables in functions *384*
numeric data types *597*
numeric precision *565*
numerical notation *547*

O

OCCURS attribute *605*
OFFLINE files *665*
UNIX *689*
Windows *665*
OFFLINE-FMT parameter *522*, *610*
OLAP *538*
OLAPGRMERGE parameter *514*, *610*
OLDSTYRECLEN parameter *610*
ON GRAPH SET command *506*
ON TABLE SET command *570*
on-demand paging *286*, *289*, *293*, *297*, *644*
displaying a specific page *293*
navigating in reports *293*
viewing reports *288*
ONFIELD parameter *518*, *611*
ONLINE-FMT parameter *522*, *611*
operating system commands *417*, *419*, *421*
-DOS command *430*
-MVS command *434*
-TSO command *439*
-UNIX command *440*
-VMS command *440*
-WINNT command *440*
optimizing retrieval paths *540*
ORIENTATION parameter *519*, *612*
OS/390 *494*
output files *114*, *115*
creating *114*, *115*
output page numbering *579*, *613*
OVERFLOWCHAR parameter *519*, *612*

P

PACKPOS parameter *508*
page numbering *579*
page orientation *612*
PAGE parameter *519*
PAGE-NUM parameter *519*, *613*
PAGESIZE parameter *519*, *613*

PANEL parameter *523*, *615*
parameter settings *459*
parameters *494*, *525*
    assigning values to variables *376*
    NEW *494*
    READ *495*
PASS parameter *524*, *615*
passwords *420*
    permanent *618*
    unchangeable *618*
PCHOLD file *591*
    displaying fields *591*
PCOMMA parameter *508*, *616*
PDF compression *576*
PDF display format *259*
PDFLINETERM parameter *617*
PERMPASS parameter *524*, *618*
PF keys *618*
PHONETIC_ALGORITHM parameter *618*
physical file location *116*
    mapping to *96*
pie chart summary values *541*
platforms *31*
    distributing processing *31*
pop-up field descriptions *619*
POPUPDESC parameter *523*, *619*
positional variables *343*
PostScript reports *636*
PPTXGRAPHTYPE parameter *619*
prepending applications to search path *94*
presentation logic *27*
PRINT parameter *620*
PRINT requests *546*
PRINTDST parameter *519*, *620*
printed output *603*
printing report output *610*
PRINTPLUS parameter *523*, *621*
procedures *127*, *273*, *275*, *335*, *524*, *660*
    altering variable values *353*
    branching *392*, *393*
    calling *401*, *402*, *406*
    canceling *391*, *396*
    comments in *335*
    controlling *329*
    creating *273*, *275*, *277*, *278*, *318*, *327*, *335*
    debugging *421*, *444*
    Dialogue Manager *329*
    exiting *390*
    HTML webpages *407*
    identifying *468*
    in Dialogue Manager *335*
    looping *397*
    navigating *392*, *393*
    nesting *405*, *406*
    processing *334*
    running *332*, *334*, *389*, *391*
    screening values in *354*
    variable values *336*, *355*–*357*
    WHENCE command *468*
processing data *508*

product release number *458*
profiles *93*
applications and *93*
program logic *27*
protecting FOCUS data sources *495*
PSPAGESETUP parameter *519*, *621*
punctuating large numbers *703–705*

Q

QUALCHAR parameter *508*, *516*, *621*
qualified column titles *622*
qualified field names *575*, *621*
qualified file names in -INCLUDE *403*
QUALTITLES parameter *519*, *622*
query commands *444*, *462*, *499*
? FILEDEF *453*, *657*
-? TSO DDNAME *681*
? && *467*, *468*
? COMBINE *446*
? COMBINE command *446*
? DEFINE *447*
? FDT *449*, *450*
? FILE *451*, *452*
? FUNCTION *454*
? HOLD *454*, *455*
? JOIN *455*
? n *457*
? NLS *456*
? PATH *457*, *458*
? RELEASE *458*
query commands *444*, *462*, *499*
? REMOTE *459*
? SET *459*
? SET EUROFILE *448*, *703*
? SET GRAPH *461*
? STAT *463*
? STYLE *464*
? SU *466*
? USE *467*, *499*
?F *448*, *449*
?FF *451*
SITECODE *462*
quote-delimited *365*, *366*
quote-delimited string *365*, *366*
QUOTEDSTRING *365*, *366*

R

RANK parameter *623*
READ parameter *495*
RECAP-COUNT parameter *519*, *623*
RECORDLIMIT parameter *515*, *623*
release number *458*
remote drill-downs *579*
remote servers *459*
report formatting *298*
Cascading Style Sheets and *298*
Report Frame *288*
report output formatting *610*
report types
drill-down *279*, *281*

reporting application logic *27*
reports *225*, *259*, *516*
ad hoc *162*
customizing *231*
displaying *225*, *226*, *228*, *231–233*, *235*, *259*, *261*, *519*
displaying in multiple formats *259*, *266*, *267*, *270*, *271*, *273*
formatting *262*, *298*
on-demand paging *286*, *289*
searching for *294*, *296*
viewing *304*
resizing graphs *539*
RETCODE variable *381*
retrieval paths *540*
retrieving comma-delimited files *616*
retrieving data *539*
retrieving records *623*
RIGHTMARGIN parameter *519*, *624*
rounding using DMPRECISION *342*
RPAGESET parameter *519*, *624*

S

SAVB files *664*
MVS *674*
UNIX *688*
Windows *664*
SAVE files *664*
MVS *674*
UNIX *688*
SAVE files *664*
Windows *664*
SAVEDMASTERS parameter *508*, *625*
SAVEMATRIX parameter *516*, *625*
SCATTER graphs *585*
screening data sources *577*
search paths *94*
adding names to *94*, *95*
appending applications manually *95*
commands *90*
displaying *457*
prepending applications manually *94*
searching for key fields *540*
security *524*
self-service applications *426*
viewing messages *426*
self-service reporting applications *27*
sequential data sources *661*, *686*
UNIX *686*
Windows *661*
servlets support *151*
SET command *289*, *340*, *343*, *373*, *374*, *376*, *397*, *400*, *438*, *501–503*, *505*, *506*
parameters *507*
precedence *502*
SET DROPBLNKLINE *566*
SET parameter types *507*
SET parameters *289*, *503–506*, *525*
%STRICTMATH *629*
ACCBLN *529*

SET parameters *289*, *503–506*, *525*
ACCESSHTML *36*, *530*
ACCESSIBLE *36*, *530*
ACCESSPDF *36*, *531*
ACROSSLINE *531*
ACROSSPRT *532*
ACROSSTITLE *533*
ACRSVRBTITL *533*
ALL *534*
ALLOWCVTERR *535*
AREXPIRE *536*
ARGRAPHENGIN *536*
ARPASSWORD *537*
ASNAMES *537*
AUTODRILL *538*
AUTOFIT *539*
AUTOINDEX *539*
AUTOPATH *540*
AUTOSTRATEGY *540*
AUTOTABLEF *541*
AUTOTICK *541*
BARNUMB *541*
BASEURL *542*
BINS *542*
BLANKINDENT *542*
BOTTOMMARGIN *543*
BUSDAYS *544*
BYDISPLAY *544*
BYPANEL *545*
CACHE *545*

SET parameters *289*, *503–506*, *525*
CARTESIAN *546*
CDN *547*
CENT-ZERO *548*
CNOTATION *548*
COLLATION *549*
COMPMISS *550*
COMPOUND *550*
COUNTWIDTH *552*
CSSURL *552*
CURRENCY_DISPLAY *552*
CURRENCY_ISO_CODE *553*
CURRENCY_PRINT_ISO *553*
CURRSYMB *554*
CURSYM_D *554*
CURSYM_E *555*
CURSYM_F *555*
CURSYM_G *556*
CURSYM_L *556*
CURSYM_Y *556*
DATE_ORDER *557*
DATE_SEPARATOR *557*
DATEDISPLAY *558*
DATEFNS *558*
DATEFORMAT *559*
DATETIME *559*
DB_INFILE *560*
DBACSENSITIV *561*
DBAJOIN *560*
DBASOURCE *561*

SET parameters *289*, *503–506*, *525*
DEFCENT *562*
DEFECHO *423*, *563*
DEFINES *563*
DIRECTHOLD *564*
displaying settings *459*
DMH_LOOPLIM *564*
DMH_STACKLIM *565*
DMPRECISION *565*
DTSTRICT *566*
EMPTYREPORT *568*
EQTEST *568*
ERROROUT *569*
ESTRECORDS *570*
EUROFILE *570*, *698*
EXCELSRVURL *570*
EXL2KLANG *571*
EXL2KTXTDATE *572*
EXPANDABLE *572*
EXPANDBYROW *572*
EXTAGGR *573*
EXTENDNUM *574*
EXTHOLD *573*
EXTRACT *574*
EXTSORT *575*
FIELDNAME *575*
FILCASE *576*
FILE *577*
FILECOMPRESS *576*
FILENAME *577*

SET parameters *289*, *503–506*, *525*
FILTER *577*
FIXRETRIEVE *578*
FLOATMAPPING *578*
FOC144 *579*
FOCEXURL *579*
FOCFIRSTPAGE *579*
FOCHTMLURL *580*
FOCSTACK *580*
FOCTRANSFORM *581*
FORMULTIPLE *518*, *581*
GRAPHDEFAULT *582*
GRAPHEDIT *582*
GRAPHENGINE *582*
GRAPHSERVURL *583*
GRID *583*
GRMERGE *584*
GRMULTIGRAPH *585*
GRTREND *585*
GRWIDTH *585*
HAUTO *586*
HAXIS *586*
HCLASS *586*
HDAY *587*
HIDENULLACRS *587*
HISTOGRAM *588*
HLDCOM_TRIMANV *588*
HMAX *589*
HMIN *589*
HNODATA *589*

SET parameters *289*, *503–506*, *525*
HOLDATTR *590*
HOLDFORMAT *591*
HOLDLIST *591*
HOLDMISS *592*
HOLDSTAT *592*
HSTACK *593*
HTICK *593*
HTMLARCHIVE *593*
HTMLCSS *594*
HTMLEMBEDIMG *594*
HTMLENCODE *595*
INDEX *595*
JOIN_LENGTH_MODE *518*, *596*
JOINLM *518*, *596*
JOINOPT *597*
JSURL *598*
KEEPDEFINES *599*
KEEPFILTERS *599*
LANG *600*
LAYOUTGRID *602*
LAYOUTRTL *602*
LEADZERO *603*
LEFTMARGIN *603*
LINES *289*, *603*
LOOKGRAPH *604*
MATCHCOLUMNORDER *604*
MAXLRECL *605*
MDICARDWARN *606*
MDIENCODING *606*
SET parameters *289*, *503–506*, *525*
MDIPROGRESS *607*
MESSAGE *607*
MISSINGTEST *608*
MULTIPATH *608*
NODATA *609*
NULL *609*
OFFLINE-FMT *610*
OLDSTYRECLEN *610*
ONFIELD *611*
ONLINE-FMT *611*
ORIENTATION *612*
OVERFLOWCHAR *612*
PAGE-NUM *613*
PAGESIZE *613*
PANEL *615*
PASS *615*
PCOMMA *616*
PDFLINETERM *617*
PERMPASS *618*
PHONETIC_ALGORITHM *618*
POPUPDESC *619*
PPTXGRAPHTYPE *619*
PRINT *620*
PRINTDST *620*
PRINTPLUS *621*
PSPAGESETUP *621*
QUALCHAR *621*
QUALTITLES *622*
RANK *623*

SET parameters *289*, *503–506*, *525*
RECAPCOUNT *623*
RECORDLIMIT *623*
RIGHTMARGIN *624*
RPAGESET *624*
SAVEDMASTERS *625*
SAVEMATRIX *625*
SHADOW *625*
SHIFT *626*
SHOWBLANKS *627*
SPACES *628*
SQLTOPTIF *628*
SQUEEZE *629*
STYLEMODE *630*
STYLESHEET *630*
SUBTOTAL *631*
SUMMARYLINES *632*
SUMPREFIX *632*
TARGETFRAME *633*
TEMP *633*
TEMPERASE *633*
TESTDATE *634*
TIME_SEPARATOR *634*
TITLE *635*
TITLELINE *635*
TOPMARGIN *636*
UNITS *636*
USER *637*
USERFCHK *637*
USERFNS *638*
SET parameters *289*, *503–506*, *525*
VAUTO *639*
VAXIS *639*
VCLASS *639*
VGRID *640*
VISBARORIENT *640*
VMAX *641*
VMIN *641*
VTICK *641*
VZERO *642*
WARNING *642*
WEBARCHIVE *642*
WEBVIEWER *644*
WEBVIEWHOME *290*
WEEKFIRST *647*
XRETRIEVAL *648*
YRTHRESH *649*
SET SQUEEZE parameter *629*
SET WPMINWIDTH *648*
SETFILE variable *381*
setting parameters *503*, *505*, *506*
setting multiple parameters *504*
SHADOW parameter *510*, *625*
SHIFT parameter *508*, *626*
SHORTPATH parameter *519*, *626*
SHOWBLANKS parameter *627*
sink machine *132*
SITECODE command *462*
sort groups *544*
sorting externally *575*

sorting fields *544*
sorting records *570*
SPACES parameter *519*, *628*
sparse ranking *623*
specifying business days *544*
specifying currency symbols *554–556*
specifying graph style *604*
specifying punctuation *547*
specifying qualifying characters *621*
SQL Translator *628*
SQLTOPTTF parameter *515*, *628*
SQUEEZE parameter *519*
stacked commands *332*, *396*
STATE command *666*, *690*
statistical variables *369*, *377*, *384*
  BASEIO *383*
  INDEXIO *383*
  LINES *383*
  READS *383*
  RECORDS *383*
  SORTIO *383*
  testing *395*
stored procedures, reporting on *481*
storing data *508*
strict processing *566*
strings *365*, *366*
structured HOLD files *574*
style sheets *28*, *298*
  Cascading Style Sheets (CSS) *298*
  WebFOCUS StyleSheets *28*
STYLEMODE parameter *519*, *630*
StyleSheet bottom boundary *543*
StyleSheet margins *603*
STYLESHEET parameter *519*, *630*
StyleSheet parameters *624*
StyleSheets *464*
  ? STYLE *464*
  formatting *630*
  identifying *468*, *470*
  page orientation *612*
  SQUEEZE parameter *629*
  WHENCE command *468*, *470*
styling HTML reports *552*
styling reports *612*
SU machine *466*
sub-servers *31*
SUBTOTAL parameter *523*
SUBTOTAL SET parameter *631*
summary values *541*
SUMMARYLINES SET parameter *632*
SUMPREFIX parameter *516*, *632*
supplying variable values *504*
suppressing blank lines *566*
sync machines *466*
synonym locations *98*
synonyms *135*
  long names on z/OS *135*
sysapps *472*
syscolum *473*
sysdeffn *474*

syserr *475*
sysfiles *476*
sysimp *478*
sysindex *479*
syskeys *480*
sysrpdir *481*
sysset *482*
syssqlop *482*
systable *483*
system tables *470*
    sysapps *472*
    syscolum *473*
    sysdeffn *474*
    syserr *475*
    sysfiles *476*
    sysimp *478*
    sysindex *479*
    syskeys *480*
    sysrpdir *481*
    sysset *482*
    syssqlop *482*
    systable *483*
    sysvdtp *484*
system variables *369*, *377*, *421*
    &FOCCODEPAGE *379*
    &FOCLANGCODE *380*
    APPROOT *377*
    AUTOINDEX *378*
    DATE *378*, *382*
    DATEfmt *378*
system variables *369*, *377*, *421*
    DMY *378*
    DMYY *379*
    ECHO *379*
    FOCEXURL *379*
    FOCFEXNAME *379*
    FOCFOCEXEC *379*, *382*
    FOCHTMLURL *380*
    FOCINCLUDE *380*
    FOCMODE *380*
    FOCNEXTPAGE *380*
    FOCQUALCHAR *381*
    FOCREL *381*
    FOCSECGROUP *381*
    FOCSECGROUPS *381*
    FOCSECUSER *381*
    MDY *381*
    MDYY *381*
    RETCODE *381*
    SETFILE *381*
    testing *395*
    TOD *382*
    YMD *382*
    YYMD *382*
sysvdtp *484*

T

TABLE requests *568*
TABLEF commands *628*
tables, reporting on *483*

TARGETFRAME parameter *519*, *633*
TEMP parameter *515*, *633*
TEMPERASE parameter *510*, *633*
temporary files *29*, *133*, *633*
temporary Master Files *665*, *675*, *688*
    MVS *675*
    Windows *665*
terminating a job *569*
TESTDATE parameter *510*, *634*
text areas *162*
three-tier application logic *27*
tick mark intervals *541*
TIME_SEPARATOR parameter *634*
TITLE attribute *590*
TITLE parameter *516*, *635*
TITLELINE parameter *524*, *635*
TOD variable *382*
TOPMARGIN parameter *519*, *636*
trailing blanks *373*
    deleting *374*
translating field names *581*
translating segment names *581*
TRUNCATE function *373*, *374*
truncating leading zeros *603*
TSO system variables *679*

U

UNC (Universal Naming Convention) *657*
unconditional branching *392*
    -GOTO command *392*, *393*
UNITS parameter *519*, *636*
Universal Naming Convention (UNC) *97*, *657*
UNIX procedures *684*
UNIX
    Access Files *684*
    application files *651*, *683*
    data sources *685*, *686*
    external indexes *686*
    extract files *651*, *687*
    Master Files *683*
    profiles *687*
    WebFOCUS files *651*
    WebFOCUS StyleSheets *686*
    work files *651*
URL location *542*
USE command *128*, *132*, *487*, *489*
    accessing FOCUS data sources *487*
    alternate file specifications *492*
    concatenating data sources *496*
    new FOCUS data sources and *494*
    NEW parameter and *494*
    protecting FOCUS data sources *495*
    READ parameter and *495*
    using Application Namespace *489*
USE directory *487*
USE options *499*
    clearing *500*
    displaying *499*
USE query *499*

Use tool *487*, *499*
accessing FOCUS data sources *487*
alternate file specifications *492*
concatenating data sources *496*
identifying new FOCUS data sources *494*
protecting FOCUS data sources *495*
user interfaces *147*
coding *147*
templates *147*
USER parameter *524*, *637*
user profiles *662*, *673*, *687*
USERFCHK parameter *637*
USERFNS parameter *638*

V

variable values *376*
-DEFAULT command and *336*, *339*
-READ command *336*
-SET command *336*
? && *467*
assigning *376*
dates *352*
default values *339*
displaying *375*, *376*
querying *467*
supplying *336*, *337*, *339*, *340*, *343*, *504*, *505*
variables *336*, *365*
concatenating *373*
quote-delimited *365*, *366*
variables *336*, *365*
supplying values *504*, *505*
VAUTO parameter *512*, *639*
VAXIS parameter *512*, *639*
VBScript files *598*
VCLASS parameter *512*, *639*
vertical axis length *639*
vertical axis maximum value *641*
vertical axis minimum value *641*
vertical axis missing values *642*
VGRID parameter *512*, *640*
Viewer *288*
closing *291*
home pages *290*, *291*
navigating *293*
opening *290*
options *292*
viewing messages *426*
viewing source code *607*
virtual fields *447*, *563*, *599*
? DEFINE command *447*
displaying *447*
VISBARORIENT parameter *524*, *640*
visual overflow *574*
VMAX parameter *512*, *641*
VMIN parameter *512*, *641*
VTICK parameter *512*, *641*
VZERO parameter *512*, *642*

W

warning message *606*
MDICARDWARN *606*
WARNING parameter *642*
web archive documents *593*, *642*
web browsers *31*
Web Console *94*
application path *78*
application paths *94*
Metadata tree *81*
web pages *29*
web servers *28*, *31*, *151*
WEBARCHIVE parameter *510*, *642*
WebFOCUS Client *31*, *150*
WebFOCUS Client variables *266*, *267*, *273*
IBIWF_mframename *267*
IBIWF_mprefix *267*
IBIWF_mreports *267*
WebFOCUS files *651*
application files *658*, *669*
dynamically defining *654*, *668*
extract files *663*, *673*, *687*
MVS *667*, *677*
referencing *652*, *667*
types *29*, *651*, *652*, *667*
UNIX *651*
Windows *651*, *654*
work files *665*
WebFOCUS Reporting Server *28*, *31*
WebFOCUS Servlet *151*
requirements *151*
WebFOCUS StyleSheets *28*, *259*, *662*, *672*
MVS *672*, *673*
UNIX *686*
Windows *662*
WebFOCUS Viewer *286*, *644*
webpages *28*, *165*, *225*, *407*, *410*, *662*
referring from a procedure *407*
-HTMLFORM command *407–409*
customizing *231*
default *225*, *226*, *228*
embedding files in *410*
embedding multiple files in *411*
JavaScript in *165*, *166*
variables *414–416*
variables and *410*
WEBVIEWALLPG parameter *292*
WEBVIEWCLOSE parameter *292*
WEBVIEWER parameter *289*, *512*, *644*
WEBVIEWHELP parameter *292*, *643*, *645*
WEBVIEWHOME parameter *290*, *291*
WEBVIEWTARG parameter *290*
WEEKFIRST parameter *508*, *647*
WHENCE command *468–470*
Windows *97*, *651*
Access Files *659*
accessing a network drive *97*
application files *658*
data sources *660*, *661*

Windows *97*, *651*
defining files *654*
external indexes *661*
extract files *663*
FILEDEF command *654*
function libraries *662*
HOLD files *663*
HOLDMAST files *665*
locating files in *651*
logical names *654*
Master Files *658*
naming files in *651*
OFFLINE files *665*
procedures *660*
profiles *662*
referencing files *652*
SAVB files *664*
SAVE files *664*
temporary Master Files *665*
WebFOCUS files *651*
WebFOCUS StyleSheets *662*
webpages *662*
work files *665*
work files *652*, *665*
FOCPOST *665*, *676*, *689*
work files *652*, *665*
FOCSML *665*, *676*, *689*
FOCSORT *665*, *676*, *689*
FOCSTACK *665*, *676*, *689*
MVS *667*, *676*
OFFLINE *665*, *689*
UNIX *651*
Windows *665*
WPMINWIDTH parameter *510*, *648*

X

XRETRIEVAL parameter *508*, *648*

Y

Y2K compliance *558*
year-2000 compliance *558*
YMD variable *382*
YRTHRESH parameter *510*, *649*
YYMD variable *382*

Z

zero field values *529*

Feedback

Customer success is our top priority. Connect with us today!

Information Builders Technical Content Management team is comprised of many talented individuals who work together to design and deliver quality technical documentation products. Your feedback supports our ongoing efforts!

You can also preview new innovations to get an early look at new content products and services. Your participation helps us create great experiences for every customer.

To send us feedback or make a connection, contact Sarah Buccellato, Technical Editor, Technical Content Management at *Sarah_Buccellato@ibi.com.*

To request permission to repurpose copyrighted material, please contact Frances Gambino, Vice President, Technical Content Management at *Frances_Gambino@ibi.com.*

WebFOCUS

Developing Reporting Applications

Release 8.2 Version 01M

DN4501669.0321

Information Builders

Information Builders, Inc.
Two Penn Plaza
New York, NY 10121-2898