

WebFOCUS



Server New Features

WebFOCUS Reporting Server Release 8205

DataMigrator Server Release 7709

Active Technologies, EDA, EDA/SQL, FIDEL, FOCUS, Information Builders, the Information Builders logo, iWay, iWay Software, Parlay, PC/FOCUS, RStat, Table Talk, Web390, WebFOCUS, WebFOCUS Active Technologies, and WebFOCUS Magnify are registered trademarks, and DataMigrator and Hyperstage 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 © 2019, by Information Builders, Inc. and iWay Software. All rights reserved. Patent Pending. This manual, or parts thereof, may not be reproduced in any form without the written permission of Information Builders, Inc.

Contents

1. Server Enhancements	7
Data Management Console: Data Type Icons	8
Data Management Console: Group and Replace	8
Data Management Console: Additional Functions Available	10
Installation: New Language Options on Windows	10
Setting Defaults for Upload Target Parameters	10
2. Adapter Enhancements	13
SQL Adapters	13
SQL Adapters: Improved Optimization of MATCH Logic.....	13
SQL Adapters: Support for SET HOLDATTR for Formats SAME_DB and SQL_SCRIPT.....	13
SQL Adapters: Optimization of Simplified Statistical Functions.....	14
Adapter for Microsoft SQL Server OLE DB: Additional Connection String.....	14
Adapter for MongoDB: Support for Connector for Business Intelligence JDBC Adapter.....	15
Adapter for Oracle: Support for Version 18c.....	15
Adapter for Snowflake Cloud Data Warehouse.....	15
Adapter for Stratio Crossdata.....	15
OLAP Adapters	15
Adapter for Microsoft SQL Server Analysis Services Tabular Data Model.....	15
ERP Adapters	16
Adapter for Salesforce.com: Support for Bulk Retrieval.....	16
Procedures Adapters	17
OAuth 2.0 Authentication Support for the Adapter for REST.....	17
Adapter for REST: Support for an XML String in a GET Request Parameter.....	19
Sequential and Indexed Adapters	20
Adapter for Delimited Files: Skip Rows Before Header	20
Adapter for Excel: Creating a Synonym for Crosstabbed Data	21
Statistics Adapters	26
Python Support.....	26
Python Functions Distributed With the Adapter for Python.....	36
3. DataMigrator Enhancements	39
Adapters	39

Adapter for Excel: Creating a Synonym for Crosstabbed Data	39
Adapter for Salesforce.com: Support for Bulk Retrieval.....	45
Adapter for Snowflake Cloud Data Warehouse.....	45
Adapter for Stratio Crossdata.....	46
Calculators	46
Additional Aggregate Functions Available.....	46
GET_TOKEN Function Now Available.....	46
INITCAP Function Now Available.....	46
New FOCUS Functions.....	46
Data Management Console	47
Data and Process Flows	47
Formatted Files With Excel Format Now Use .XLSX.....	47
Enhancements to Join Profiling.....	47
4. Resource Analyzer and Resource Governor Enhancements	49
New Audit Report	49
New Repository Load Type Options	49
New Scheduling Report Option	50
5. Reporting Language Enhancements	51
Expanded Functionality With BY TOTAL and ACROSS	51
COALESCE: Returning the First Non-Missing Value	52
COMPACTFORMAT: Displaying Numbers in an Abbreviated Format	54
FPRINT: Displaying a Value in a Specified Format	55
GET_TOKEN: Extracting a Token Based on a String of Delimiters	57
GIS_REVERSE_COORDINATE: Returning a Geographic Component	58
INITCAP: Capitalizing the First Letter of Each Word in a String	60
NULLIF: Returning a Null Value When Parameters Are Equal	61
Statistical Python Functions	63
BLR_CLASSIFY: Binary Logistic Regression.....	64
KNN_CLASSIFY: K-Nearest Neighbors Classification.....	66
KNN_REGRESS: K-Nearest Neighbors Regression.....	68
POLY_REGRESS: Polynomial Regression.....	71
RF_CLASSIFY: Random Forest Classification.....	73

RF_REGRESS: Random Forest Regression.....	75
Using IF-THEN-ELSE Logic in an Expression	77
Generating Descriptive Column Titles for Prefixed Fields	81
Format Display Options for Abbreviating Numeric Values	82
Support for C-Style In-Line Comments	84
SET PCTFORMAT: Displaying a Percent Sign for Percent Prefix Operators	85

Server Enhancements

















The server provides a wide range of capabilities and tools for adapter configuration, metadata creation, application and path management, security control, communications configuration, and for monitoring, tuning, and troubleshooting server performance. Authorized users can perform most server administration tasks from a graphical Web Console. The server supports WebFOCUS reporting functions, extraction, load and transformation functions, and analysis and data access control functions.

In this chapter:

- ☐ [Data Management Console: Data Type Icons](#)
 - ☐ [Data Management Console: Group and Replace](#)
 - ☐ [Data Management Console: Additional Functions Available](#)
 - ☐ [Installation: New Language Options on Windows](#)
 - ☐ [Setting Defaults for Upload Target Parameters](#)
-

Data Management Console: Data Type Icons

In the DMC, field names are shown with an icon that identifies the data type of the field. For example:

 I23	TRIPDURATION	I9	Integer
 STARTTIME		HYYMDS	Date and Time
 STOPTIME		HYYMDS	Date and Time
 I23	START_STATION_ID	I6	Integer
 Abc	START_STATION_NAME	A56V	Character
 START_STATION_LATITUDE		D20.15	Numeric
 START_STATION_LONGITUDE		D20.14	Numeric
 I23	END_STATION_ID	I6	Integer
 Abc	END_STATION_NAME	A56V	Character
 END_STATION_LATITUDE		D20.15	Numeric
 END_STATION_LONGITUDE		D20.14	Numeric
 I23	BIKEID	I7	Integer
 Abc	NAME_LOCALIZEDVALUE0	A63V	Character
 Abc	USERTYPE	A12V	Character
 Abc	BIRTH_YEAR	A5V	Character
 I23	GENDER	I3	Integer

Data Management Console: Group and Replace

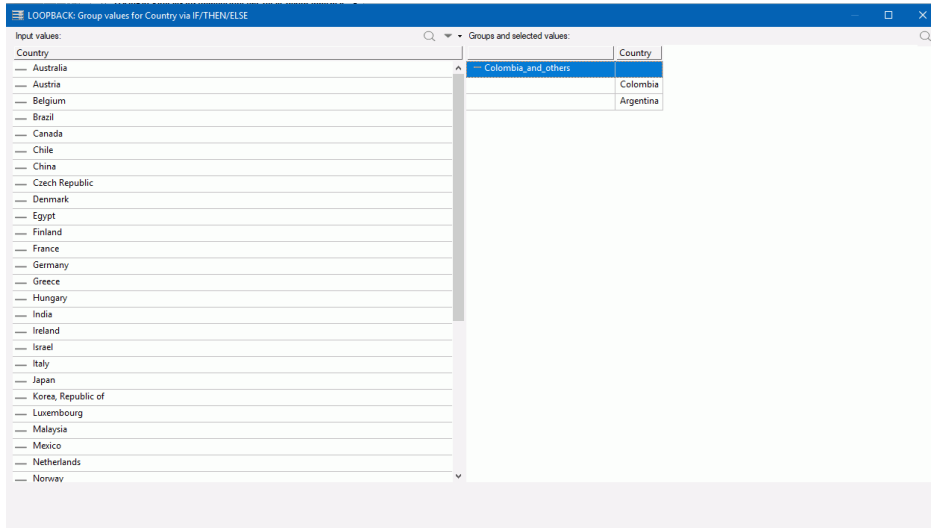
In the synonym editor, the *Group and Replace* option provides a new user interface to simplify adding a field that groups together similar related values.

To implement Group and Replace, right-click a field in the synonym editor, point to *New Expression*, and click *Group and Replace*.

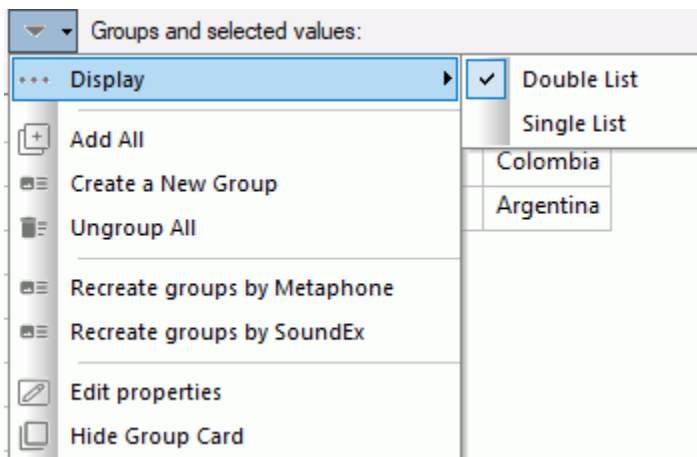
A group field is created with all values ungrouped.

You can create new subgroups either by selecting *Create a new group* from the pull-down menu, or by right-clicking a value on the list and clicking *Make a new group*.

Once you have created a subgroup, you can drag values into that subgroup, as shown in the following image. To facilitate this operation, *Double List* was selected from the *Display* option on the pull-down menu.



The options available on the pull-down menu are shown in the following image.



You can perform the following operations using this menu.

- ☐ Display the group frame as a single list or a double list.

- ☐ Add all of the values to a group.
- ☐ Ungroup all of the values.
- ☐ Recreate groups phonetically using either the Metaphone or Soundex algorithm. Soundex calculates a code for alphanumeric values so that if they sound the same but are spelled differently, they will still have the same code. Metaphone is an improved version of Soundex that takes into account anomalies in English spelling and pronunciation.
- ☐ Edit the properties of the group field, such as its name.
- ☐ Hide the group card.

Data Management Console: Additional Functions Available

The following functions are now available from the DMC:

- ☐ Aggregate functions Median and Mode are now available in the SQL object from the *Aggregate* drop-down menu.
- ☐ Character function SPLIT.
- ☐ Date and Date-Time functions DTADD and DTDIFF.
- ☐ Statistical Functions CORRELATION, STANDEV_POP, and STANDEV_SAMP.

Installation: New Language Options on Windows

The following languages are now available as options when installing the server on Windows:

- ☐ Brazilian Portuguese.
- ☐ Italian.
- ☐ Korean.
- ☐ Traditional Chinese.

Setting Defaults for Upload Target Parameters

New parameters have been added to the Settings for Web Console Preferences page to enable you to implement default targets for the Upload Wizard, Quick Copy, and Custom Copy.

To set your target parameters, go to the Web Console Workspace page, click *Settings*, point to *FOCUS Sets and Info*, then click *Settings for Web Console Preferences*.

The Settings for Web Console Preferences page opens. The new section is named Target Defaults, as shown in the following image.

You can set values for the following parameters.

- ☐ **ETL-TRG-DBMS.** Select a target adapter and connection from the drop-down list for target tables in the Upload Wizard, Upload, Quick Copy and Custom Copy.
- ☐ **ETL-TRG-SQL.** Select Yes or No to enable or disable SQL targets. The default value is Yes.
- ☐ **ETL-TRG-APP.** Enter the name of the target application directory for target tables in the Upload Wizard, Upload, Quick Copy and Custom Copy, or click the ellipsis (. . .) to select an application directory.
- ☐ **ETL-TRG-BULK.** Select Yes or No to use or disable bulk load in the Upload Wizard, Upload, Quick Copy and Custom Copy. The default value is Yes for any DBMS that supports bulk load.
- ☐ **ETL-TRG-PROMPT.** Select Yes or No to enable load option prompts. The default value is Yes.

You can also issue these settings in any supported profile, using the following syntax.

```
ENGINE INT SET ETL-TRG-DBMS dbms/connection
```

```
ENGINE ITN SET ETL-TRG-SQL {YES|NO}
```

```
ENGINE ITN SET ETL-TRG-APP appname
```

```
ENGINE ITN SET ETL-TRG-BULK {YES|NO}
```

```
ENGINE ITN SET ETL-TRG-PROMPT {YES|NO}
```

For example:

```
ENGINE INT SET ETL-TRG-DBMS SQLMSS/CON03
ENGINE INT SET ETL-TRG-APP mycompany
ENGINE INT SET ETL-TRG-PROMPT NO
```


Adapter Enhancements

This section describes new adapter features. All adapters can be used for WebFOCUS Reporting and SQL requests, and as sources for DataMigrator flows.

The server supports adapters designed to access a wide variety of data sources. Using the graphical Web Console, you can configure these adapters and create the metadata you need to seamlessly access the corresponding type of data.

On the Web Console, adapters are grouped as SQL, XML-based, ERP, OLAP, Procedures, Sequential and Indexed, DBMS, and Social Media.

In this chapter:

- ☐ [SQL Adapters](#)
 - ☐ [OLAP Adapters](#)
 - ☐ [ERP Adapters](#)
 - ☐ [Procedures Adapters](#)
 - ☐ [Sequential and Indexed Adapters](#)
 - ☐ [Statistics Adapters](#)
-

SQL Adapters

This section provides descriptions of new features for SQL adapters.

SQL Adapters: Improved Optimization of MATCH Logic

Streamlined SQL is now generated for MERGE requests against those adapters, such as Spark, whose INSERT statement does not support a list of columns. The generated SQL now does not issue redundant SELECT statements.

SQL Adapters: Support for SET HOLDATTR for Formats SAME_DB and SQL_SCRIPT

The SET HOLDATTR command propagates TITLE and DESCRIPTION attributes from a source Master File to a HOLD Master File. This command is now supported for formats SQL_SCRIPT and SAME_DB.

Example: Propagating TITLE and DESCRIPTION Attributes to a HOLD File in Format SQL_SCRIPT

The following request propagates TITLE and DESCRIPTION attributes to a HOLD file generated as format SQL_SCRIPT.

```
APP HOLD ibisamp
SET HOLDATTR=ON
TABLE FILE wf_retail_lite
SUM BUSINESS_REGION STATE_PROV_CODE_ISO_3166_2
BY BUSINESS_REGION NOPRINT
BY STATE_PROV_CODE_ISO_3166_2 NOPRINT
WHERE BUSINESS_REGION EQ 'North America' OR 'EMEA'
WHERE STATE_PROV_CODE_ISO_3166_2 EQ 'AR' OR 'IA' OR 'KS' OR 'KY'
OR 'WY' OR 'CT' OR 'MA' OR '04' OR '11' OR '14' OR 'NJ'
OR 'NY' OR 'RI'
ON TABLE HOLD AS HOLDATTR2 FORMAT SQL_SCRIPT
END
```

The generated Master File, holdattr2.mas follows. The TITLE and DESCRIPTION attributes were propagated from the wf_retail_lite Master File.

```
FILENAME=HOLDATTR2, SUFFIX=DATREC , IOTYPE=BINARY, $
  SEGMENT=HOLDATTR, SEGTYPE=S0, $
    FIELDNAME=BUSINESS_REGION, ALIAS=E01, USAGE=A15V, ACTUAL=A15V,
MISSING=ON,
      TITLE='Customer,Business,Region', DESCRIPTION='Business Region', $
    FIELDNAME=STATE_PROV_CODE_ISO_3166_2, ALIAS=E02, USAGE=A5V,
ACTUAL=A5V,      MISSING=ON,
      TITLE='Customer,State,Province,ISO-3166-2,Code', DESCRIPTION='The
ISO-3166-2 Code for the State or Province', $
    FIELDNAME=NULLFLAG, ALIAS=__NULLFLAG__, USAGE=A2, ACTUAL=A2B,
ACCESS_PROPERTY=(INTERNAL), $
```

SQL Adapters: Optimization of Simplified Statistical Functions

Simplified statistical functions can now be optimized to SQL for those adapters that support statistical functions. To determine which adapters support optimization of the simplified statistical functions, run the SQL Optimization Report from the Web Console Connect to Data page.

Adapter for Microsoft SQL Server OLE DB: Additional Connection String

The configuration page for the Adapter for Microsoft SQL Server OLE DB now has a text box for adding additional connection keywords.

Adapter for MongoDB: Support for Connector for Business Intelligence JDBC Adapter

Support has been added for the MongoDB Enterprise Server, with the Business Intelligence Connector that provides SQL access to data and added view support in MongoDB with a MySQL client and JDBC Driver. The Business Intelligence Connector is distributed by MongoDB and supports Version 3.4 and up of MongoDB. This new adapter replaces the Adapter for MongoDB available in prior releases.

Adapter for Oracle: Support for Version 18c

The Adapter for Oracle now has support for Version 18c.

Adapter for Snowflake Cloud Data Warehouse

The Adapter for Snowflake Cloud Data Warehouse is new in this release. It can be found in the SQL category.

This adapter accesses data stored in the Snowflake Cloud through SQL query operations.

While Snowflake provides both ODBC and JCBC drivers, in the initial release, only the JDBC version is certified and provides Extended Bulk Load functionality.

Adapter for Stratio Crossdata

A new adapter for Crossdata provides access to data using the Stratio JDBC driver. This adapter is in the SQL category.

OLAP Adapters

This section contains descriptions of features for OLAP adapters.

Adapter for Microsoft SQL Server Analysis Services Tabular Data Model

The Adapter for Microsoft SQL Server Analysis Services Tabular Data Model is new in this release. It can be found under OLAP adapters on the New Adapter drop-down list.

The adapter provides access to Analysis Services data sources that are deployed using the Microsoft SQL Server Analysis Services Tabular protocol, whose underlying architecture is an in-memory columnar database. The main metadata object of a Tabular database is a Model, which is a set of joined tables. A Perspective is a subset of a model.

Note that the Adapter for Microsoft SQL Server Analysis Services Tabular Data Model is for use only with Analysis Services data sources that are deployed in tabular mode. For access to Analysis Services data sources that are deployed in Multidimensional mode, use the Adapter for SQL Server Analysis Services (SSAS) available in prior releases.

The underlying language, Data Analysis Expressions (DAX), is radically different from the Multidimensional Expressions (MDX) language used with the multidimensional model.

Data within the model is not pre-aggregated, as it is with the multidimensional cube. A tabular model always contains tables and columns, while OLAP elements (such as measures, KPIs, and hierarchies) are optional. When you generate a synonym for a tabular model or perspective, the synonym describes tables as segments and columns (as well as optional measures and KPIs) as fields. If the model contains hierarchies, their structure is reflected in the synonym by properly organizing the fields that describe hierarchy levels. WebFOCUS requests against these synonyms are similar to requests against a relational data source in the sense that DAX queries generated by the adapter reference tables and columns rather than cubes and their OLAP elements (such as dimension attributes), and data selection is expressed in terms of filter predicates rather than sets of dimension members.

ERP Adapters

This section describes new features for ERP adapters.

Adapter for Salesforce.com: Support for Bulk Retrieval

In prior releases, the Adapter for Salesforce.com used a REST API to issue SOQL (Salesforce Object Query Language) commands to retrieve data. This technique is acceptable for direct reporting against Salesforce or for migrating modest data volumes.

However, Salesforce.com recommends using the BULK API capability *bulk query* to efficiently query large data sets and reduce the number of API requests

Bulk Query can now be enabled from the settings panel for the adapter under *Bulk Services* by selecting ON from the new BULKQUERY pull-down menu. Additional options allow you to enable PK (Primary Key) chunking and to set the chunk size.

Bulk query can also be enabled or disabled with the following command:

```
ENGINE SFDC SET BULKQUERY {ON|OFF}
```

When this option is selected, queries that meet the salesforce.com requirements (such as no aggregation) use bulk query.

To extract extremely large data sets, the Bulk API capability *PK Chunking* can be enabled with the following command:

```
ENGINE SFDC SET PKCHUNKING {ON|OFF}
```

When this option is enabled, the Primary Key of the object is used to retrieve data in chunks, allowing retrieval of larger data volumes.

When PK chunking is enabled, the chunk size can be set using the following command:

```
ENGINE SFDC SET PKCHUNKSIZE {n|100000}
```

where:

n

Where *n* is an integer between zero (0) and 250000. When PKCHUNKSIZE is set to zero or not set, the default chunk size (100,000) is used.

Procedures Adapters

This section provides descriptions of new features for procedures adapters.

OAuth 2.0 Authentication Support for the Adapter for REST

The OAuth 2.0 authentication protocol for granting access to REST APIs is now supported for the Adapter for REST.

Prior to configuring OAuth 2.0 authentication, developers must establish the development environment for the application to obtain the client ID and client secret used for grant authorization code and grant client credentials.

The Adapter for REST is under Procedures on the New Datasource drop-down list of the Web Console Connect to Data page. Right-click the adapter and click *Configure*.

The configuration parameters are shown in the following image.

Connect parameters

? Connection Name

? Base Url Sample: <http://search.twitter.com/>

? Security

? OAuth Grant Type

? Chained Authentication ☐

? Client ID

? Client Secret

? Token URL

Advanced HTTP connection options

Environment

? Select profile (type in a new one or select one from the list)

Enter or select values for the following parameters.

Connection Name

Is the logical name used to identify this particular set of connection attributes. The default is CON01.

Base URL

Is the part of the URL that is common for calling the various functions within a specific REST API.

Security

Select *OAuth* from the Security drop-down list.

The OAuth Grant Type drop-down list appears.

OAuth Grant Type

Can be one of the following three grant types.

☐ **Authorization Code.**

This is used for obtaining an Access Token and Refresh Token using an Authorization Code returned from an Authorization request. The Access Token is automatically renewed using the configured Refresh Token when a request is run.

☐ **Client Credentials.** This is used when the OAuth implementation requires only a Client ID and Client Secret. Once it is selected, the Client ID, Client Secret, and Token URL fields appear. The values for these fields come from the application development environment created as a prerequisite to configuring the adapter. For example, OAuth configuration for the Twitter API is done through the Twitter Developers environment accessed from <https://developer.twitter.com>.

☐ **Password.** This is used when the OAuth implementation requires only a User ID and Password

The relevant parameters display when you select the Grant Type. Enter or select values for the relevant parameters. The list of OAuth parameters follows.

Service Provider

Is the web service provider. Select *custom*.

Service URL

Is the URL to the web service when not configured as a REST adapter connection.

Client ID

Is the Client ID provided by the OAuth implementation of the application being accessed.

Client Secret

Is the Client Secret provided by the OAuth implementation of the application being accessed.

Authorization URL

Is the URL used for OAuth authorization to a specific application.

For example, the Authorization URL for the Google set of APIs is `https://accounts.google.com/o/oauth2/auth`.

Token URL

Is the URL used for obtaining an Access Token to a specific application using the Authorization Code obtained from the OAuth Authorization request.

For example, the Token URL for the Google set of APIs is `https://accounts.google.com/o/oauth2/token`.

Additional Authentication Parameters

Are additional parameters needed for the OAuth Authorization request to a specific application.

Additional Token Parameters

Are additional parameters needed for the OAuth Token request to a specific application.

Access Token

Is the Access Token returned from the OAuth Token request.

Refresh Token

Is the Refresh Token returned from the OAuth Token request.

User

Is the user ID used to perform OAuth authentication using the Authorization URL.

Password

Is the password used to perform OAuth authentication using the Authorization URL.

When you have configured the OAuth authentication parameters, click *Test* to test the connection.

When the test shows that the connection is configured correctly, click *Configure*.

Adapter for REST: Support for an XML String in a GET Request Parameter

An XML string that is constructed from WHERE statements can now be passed as a parameter value in a GET request for those web services that need to pass XML strings within a query parameter of a GET request.

Sequential and Indexed Adapters

This section provides new feature descriptions for sequential and indexed adapters.

Adapter for Delimited Files: Skip Rows Before Header

When you upload or create a synonym for a delimited file, you can indicate that a header row exists that provides column names. Sometimes, delimited files have other rows above the header row. These rows may be blank or may contain general heading information that do not provide attributes that should be included in the synonym. The synonym creation parameters screen for delimited files now provides an option for indicating the number of rows that need to be skipped before the header row, as shown in the following image.

Create Synonym for Delimited Files (CSV/TAB)

?

Create Synonym options

?

Scan All rows

☐

?

Header row

☒

?

Number of first rows to skip

?

Field Delimiter

.,(comma)

(Select or type in a delimiter)

?

Field Enclosure

"(double quote)

(Select or type in an enclosure)

?

CODEPAGE

1252 - Windows (Latin 1) - Server Default

(Select file codepage)

?

CDN

OFF - Server Default

(Select file Continental Decimal Notation)

Enter the number of rows to skip in the field labeled *Number of first rows to skip*.

For example, in the following image, the header row is row 5.

	A	B	C	D	E	F
1	Data Source	World Development Indicators				
2						
3	Last Updated	8/28/2018				
4						
5	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961
6	Aruba	ABW	Population, total	SP.POP.TOTL	54211	55438
7	Afghanistan	AFG	Population, total	SP.POP.TOTL	8996351	9166764
8	Angola	AGO	Population, total	SP.POP.TOTL	5643182	5753024
9	Albania	ALB	Population, total	SP.POP.TOTL	1608800	1659800
10	Andorra	AND	Population, total	SP.POP.TOTL	13411	14375
11	Arab World	ARB	Population, total	SP.POP.TOTL	92490932	95044497
12	United Arab Emirates	ARE	Population, total	SP.POP.TOTL	92634	101078
13	Argentina	ARG	Population, total	SP.POP.TOTL	20619075	20953077
14	Armenia	ARM	Population, total	SP.POP.TOTL	1874120	1941491
15	American Samoa	ASM	Population, total	SP.POP.TOTL	20013	20486

Entering 4 in the Number of first rows to skip field adds the following attribute in the Access File.

`SKIP_ROWS=4`

This number is used both when creating the synonym and reading the data. The *Show File* option that is available when creating a synonym now shows all lines, including blank lines, to aid you in determining how many lines should be skipped.

Adapter for Excel: Creating a Synonym for Crosstabbed Data

Synonyms can now be automatically created in a crosstab format for data in Excel Worksheets.

When you create a synonym using the Adapter for Excel, there is now a check box labeled Crosstab for each Worksheet, as shown in the following image.

Select the Excel worksheet(s) to upload by selecting the corresponding checkbox(s).

Create Synonym for Excel

? Scan All rows ☐

? Number of header rows (Enter number of topmost rows containing column headers)

Advanced

Customize data type mappings

Workbook: ibisamp/retail_cross.xlsx

<input type="checkbox"/> Default Name (editable)	Worksheet Name	Named Range	Rows/Columns	Crosstab	Multiple Measures
<input checked="" type="checkbox"/> sheet1	Sheet1		93/22	<input type="checkbox"/>	<input checked="" type="checkbox"/>

When selected, the header rows, up to the number specified, are un-pivoted to become data values.

The following shows a partial Excel Spreadsheet open in the Data Management Console. The headers include Product Category, a value for Product Category, Product Subcategory, and a value for Product Subcategory.

ROWID	REPORT_RETAIL_WF_RETAIL_LITE_RPT_FEX	FIELD_2	FIELD_3	FIELD_4	FIELD_5	FIELD_6	FIELD_7	FIELD_8
1	2	Product,Category
2	3	Accessories	.	.	Camcorder	.	.	Computers
3	4	Product,Subcategory
4	5	Charger	Headphones	Universal Remote Controls	Handheld	Professional	Standard	Smartphone
5	6	Store Name
6	7	Amsterdam	2724	5738	4471	6276	339	4649 4881
7	8	Anchorage	575	1269	966	1309	72	1039 1169
8	9	Arlington	420	857	750	1037	67	774 849
9	10	Athens	39	81	47	98	4	65 47
10	11	Atlanta	820	1708	1356	1923	97	1475 1674
11	12	Bangalore	79	187	144	196	7	168 121
12	13	Bangkok	1	3	4	8	2	20 7
13	14	Barcelona	92	239	145	204	10	164 130
14	15	Beijing	3	12	7	11	.	8 2
15	16	Belfast	1	6	5	2	.	6 1
16	17	Berlin	2477	5286	4370	6118	292	4558 4901
17	18	Boise	163	249	182	305	18	229 205
18	19	Boston	397	1022	723	1026	55	771 899

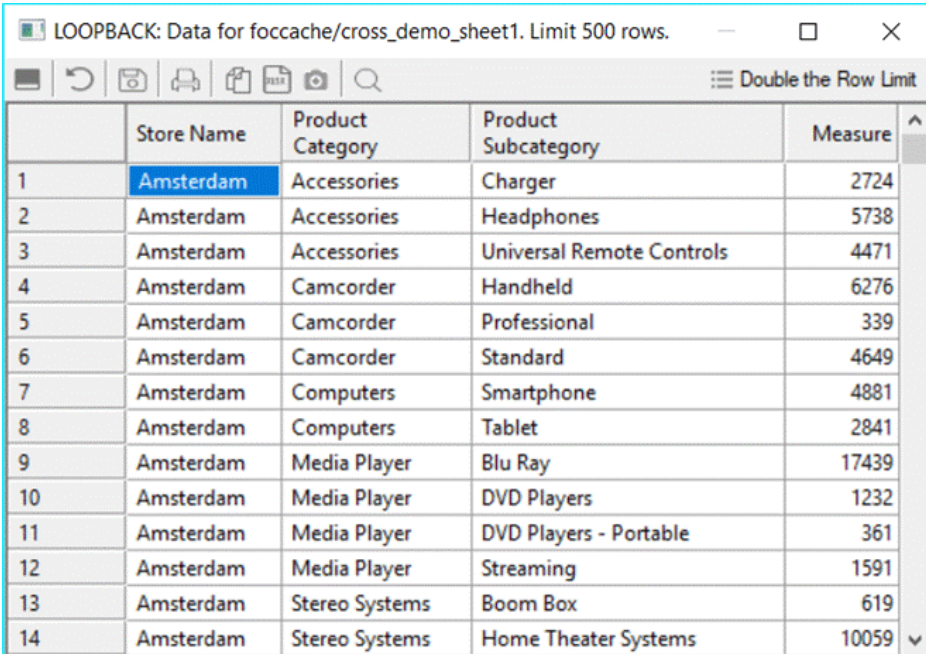
With Crosstab not checked, the resulting Master File has a field for column on the Spreadsheet. The following is a partial list of fields for the Spreadsheet:

```

FIELDNAME=PRODUCT_CATEGORY_ACCESSORIES_PRODUCT_SUBCATEGORY,
ALIAS='Product,Category_Accessories_Product,Subcategory', USAGE=A8V,
ACTUAL=A8V,
MISSING=ON,
TITLE='Product,Category_Accessories_Product,Subcategory', $
FIELDNAME=FIELD_3, ALIAS=FIELD_3, USAGE=A12V, ACTUAL=A12V,
MISSING=ON,
TITLE='FIELD_3', $
FIELDNAME=FIELD_4, ALIAS=FIELD_4, USAGE=A31V, ACTUAL=A31V,
MISSING=ON,
TITLE='FIELD_4', $
FIELDNAME=CAMCORDER, ALIAS=Camcorder, USAGE=A10V, ACTUAL=A10V,
MISSING=ON,
TITLE='Camcorder', $
FIELDNAME=FIELD_6, ALIAS=FIELD_6, USAGE=A15V, ACTUAL=A15V,
MISSING=ON,
TITLE='FIELD_6', $
FIELDNAME=FIELD_7, ALIAS=FIELD_7, USAGE=A10V, ACTUAL=A10V,
MISSING=ON,
TITLE='FIELD_7', $
FIELDNAME=COMPUTERS, ALIAS=Computers, USAGE=A12V, ACTUAL=A12V,
MISSING=ON,
TITLE='Computers', $
FIELDNAME=FIELD_9, ALIAS=FIELD_9, USAGE=A7V, ACTUAL=A7V,
MISSING=ON,
TITLE='FIELD_9', $
FIELDNAME=MEDIA_PLAYER, ALIAS='Media Player', USAGE=A8V, ACTUAL=A8V,
MISSING=ON,
TITLE='Media Player', $

```

With Crosstab checked, multiple measures unchecked, and five header rows specified, PRODUCT_CATEGORY becomes one dimension field, PRODUCT_SUBCATEGORY becomes a second dimension field, and the numeric values in the Spreadsheet become a measure field. The first column (containing store names) becomes an alphanumeric field. For example:



	Store Name	Product Category	Product Subcategory	Measure
1	Amsterdam	Accessories	Charger	2724
2	Amsterdam	Accessories	Headphones	5738
3	Amsterdam	Accessories	Universal Remote Controls	4471
4	Amsterdam	Camcorder	Handheld	6276
5	Amsterdam	Camcorder	Professional	339
6	Amsterdam	Camcorder	Standard	4649
7	Amsterdam	Computers	Smartphone	4881
8	Amsterdam	Computers	Tablet	2841
9	Amsterdam	Media Player	Blu Ray	17439
10	Amsterdam	Media Player	DVD Players	1232
11	Amsterdam	Media Player	DVD Players - Portable	361
12	Amsterdam	Media Player	Streaming	1591
13	Amsterdam	Stereo Systems	Boom Box	619
14	Amsterdam	Stereo Systems	Home Theater Systems	10059

The resulting Master File follows:

```
FILENAME=RETAIL_CROSS_CROSS_ON, SUFFIX=DIREXCEL,
DATASET=ibisamp/retail_cross.xlsx, BV_NAMESPACE=OFF, $
SEGMENT=RETAIL_CROSS_CROSS_ON, SEGTYPE=S0, $
  FIELDNAME=DHL1, ALIAS=DHL1, USAGE=A20V, ACTUAL=A20V,
  MISSING=ON,
  TITLE='DHL1', $
  FIELDNAME=DVL1_PRODUCT_CATEGORY, ALIAS='Product,Category',
  USAGE=A20V, ACTUAL=A20V,
  MISSING=ON,
  TITLE='Product,Category', $
  FIELDNAME=DVL2_PRODUCT_SUBCATEGORY, ALIAS='Product,Subcategory',
  USAGE=A31V, ACTUAL=A31V,
  MISSING=ON,
  TITLE='Product,Subcategory', $
  FIELDNAME=REPORT_RETAIL_WF_RETAIL_LITE_RPT_FEX,
  ALIAS='Report retail/wf_retail_lite_rpt.fex',
  USAGE=I9, ACTUAL=A11V,
  MISSING=ON,
  TITLE='Report retail/wf_retail_lite_rpt.fex', $
FOLDER=MG_RETAIL_CROSS_CROSS_ON, PARENT=.,
  DV_ROLE=MEASURE,
  DESCRIPTION='Measure Groups', $
FOLDER=RETAIL_CROSS_CROSS_ON, PARENT=MG_RETAIL_CROSS_CROSS_ON,
  DESCRIPTION='Retail_cross_cross_on', $
  FIELDNAME=REPORT_RETAIL_WF_RETAIL_LITE_RPT_FEX,
  BELONGS_TO_SEGMENT=RETAIL_CROSS_CROSS_ON, $
FOLDER=DIM_RETAIL_CROSS_CROSS_ON, PARENT=.,
  DV_ROLE=DIMENSION,
  DESCRIPTION='Dimensions', $
FOLDER=RETAIL_CROSS_CROSS_ON1, PARENT=DIM_RETAIL_CROSS_CROSS_ON,
  DESCRIPTION='Retail_cross_cross_on', $
  FIELDNAME=DHL1, BELONGS_TO_SEGMENT=RETAIL_CROSS_CROSS_ON, $
FOLDER=DVL, PARENT=RETAIL_CROSS_CROSS_ON1,
  DESCRIPTION='DVL', $
  FIELDNAME=DVL1_PRODUCT_CATEGORY,
  BELONGS_TO_SEGMENT=RETAIL_CROSS_CROSS_ON,
  DV_ROLE=LEVEL, $
  FIELDNAME=DVL2_PRODUCT_SUBCATEGORY,
  BELONGS_TO_SEGMENT=RETAIL_CROSS_CROSS_ON,
  DV_ROLE=LEVEL, $
```

You can now issue a request similar to the following:

```
TABLE FILE RETAIL_CROSS_CROSS_ON
PRINT REPORT_RETAIL_WF_RETAIL_LITE_RPT_FEX AS Measure
BY DHL1 AS Store
BY DVL1_PRODUCT_CATEGORY AS Category
BY DVL2_PRODUCT_SUBCATEGORY AS Subcategory
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```


Partial output is shown in the following image.

<u>Store</u>	<u>Category</u>	<u>Subcategory</u>	<u>Measure</u>
Amsterdam	Accessories	Charger	2724
		Headphones	5738
		Universal Remote Controls	4471
	Camcorder	Handheld	6276
		Professional	339
		Standard	4649
	Computers	Smartphone	4881
		Tablet	2841
	Media Player	Blu Ray	17439
		DVD Players	1232
		DVD Players - Portable	361
		Streaming	1591
	Stereo Systems	Boom Box	619
		Home Theater Systems	10059
		Receivers	3833
		Speaker Kits	6257
		iPod Docking Station	7939
	Televisions	CRT TV	333
		Flat Panel TV	2293
		Portable TV	517
	Video Production	Video Editing	4897
Anchorage	Accessories	Charger	575
		Headphones	1269
		Universal Remote Controls	966
	Camcorder	Handheld	1309
		Professional	72
		Standard	1039
	Computers	Smartphone	1169
		Tablet	962
	Media Player	Blu Ray	3588
		Streaming	343
	Stereo Systems	Home Theater Systems	2191
		Receivers	805
		Speaker Kits	1441
		iPod Docking Station	1651
	Televisions	Flat Panel TV	459
	Video Production	Video Editing	1066

Statistics Adapters

This section contains descriptions of features for statistics adapters.

Python Support

Python is a high-level, easy to use, powerful, interpreted programming language suitable for scripting, as well as complex programming.

The Python standard library, an extensive collection of modules, implements the Python "batteries included" philosophy that gives programmers immediate access to sophisticated and robust capabilities that make it easy to write your own Python functions to be used in WebFOCUS.

The Adapter for Python defines a connection to the Python interpreter for executing user-written Python scripts that generate calculated fields (WebFOCUS Computes). These fields can be used in WebFOCUS Workbooks, WebFOCUS InfoGraphics, charts, reports, dashboards, and WebFOCUS portals.

The Adapter for Python also comes with a collection of statistical functions, implemented in Python, that you can call from WebFOCUS.

Reference: Guidelines for Writing Python Scripts to be Used With WebFOCUS

The following Python script, named `arithmetic_example.py` contains a function named `adder` that adds two numbers and conforms to the requirements described in this section.

```
# arithmetic_example.py

import csv

def adder(csvin, csvout):

    with open(csvin, 'r', newline='') as file_in,\
        open(csvout, 'w', newline='') as file_out:

        fieldnames = ['addition']

        reader = csv.DictReader(file_in, quoting=csv.QUOTE_NONNUMERIC)
        writer = csv.DictWriter(file_out, quoting=csv.QUOTE_NONNUMERIC,
                                fieldnames=fieldnames)
        writer.writeheader()

        for row in reader:
            addition = row['a_number'] + row['another_number']

            writer.writerow({'addition': addition})
```

A Python script must conform to the following requirements in order to be compatible with WebFOCUS.

- ❑ **csv module requirement.** The script must import the csv module because WebFOCUS sends data to the Python script using an automatically generated, temporary .csv file. The name of the file is stored in a global Python variable named `csvin`. The global Python variable `csvout` contains the name of the temporary .csv results file to be returned to WebFOCUS. WebFOCUS sets the values for `csvin` and `csvout`, and they should not be changed by the programmer. The temporary files are removed immediately and cannot be viewed by the user. The format of both files is the same. The default delimiter is a comma (,) and cannot be changed. All non-numeric fields are enclosed in double quotation marks.

The global variables `csvin` and `csvout` are defined by WebFOCUS for reading and writing .csv files.

- ❑ **csvin and csvout format parameter.** The script must open `csvin` and `csvout` with the following format parameter:

```
newline=''
```

If `newline=""` is not specified, newlines embedded inside quoted fields will not be interpreted correctly, and on platforms that use `\r\n` line endings on write, an extra `\r` will be added.

- ❑ **reader and writer format parameter.** The following format parameter should be included in the reader and writer statements, whether you are using reader and writer or DictReader and DictWriter, to enclose non-numeric values in double quotation marks and make it clear which values are numeric (they will be converted to floating point values):

```
quoting=csv.QUOTE_NONNUMERIC
```

This indicates that non-numeric (alphanumeric, text, string, and date) values are enclosed in double quotation marks and that numeric values are not. When `csvin` is read, all WebFOCUS numeric values will be automatically converted to Python floating-point numbers. If the WebFOCUS COMPUTE defines the returned field as integer, the decimal point and any decimal places will be truncated. In the `csvin` file, if a non-numeric field contains the double quotation character, it will be doubled by WebFOCUS. Python will correctly parse this because the `Dialect.doublequote` format parameter of the Python csv module defaults to `True`.

- ❑ **Output.** The function can return multiple output fields. However, each WebFOCUS COMPUTE command can only retrieve a single output field. To retrieve multiple output fields, issue multiple COMPUTE commands. In the call to the PYTHON function, the output argument (the last argument) must match the name of a field in the OUTPUT_DATA segment of the synonym generated for the Python script.

For example, in the following Master File, the output argument is called ADDITION:

```
SEGMENT=OUTPUT_DATA, SEGTYPE=U, PARENT=INPUT_DATA, $  
FIELDNAME=ADDITION, ALIAS=addition, USAGE=D7.1,  
ACTUAL=STRING, MISSING=ON, TITLE='Addition', $
```

Therefore, the last (output) argument name in the call to PYTHON must be ADDITION:

```
COMPUTE Anyname/I5 = PYTHON(synonym_name, anyarg1, anyarg2, ADDITION);
```

The output written to csvout must be a sequence, for example, a list, even for a single field.

For a list containing a single field, the correct syntax is:

```
writer.writerow([result])
```

The following syntax is incorrect and will return incorrect values for strings and raise an exception for numeric fields:

```
writer.writerow(result)
```

- ❑ **Functions.** The Python script can contain one or more user-defined functions. When the metadata object (synonym) is created for the script, one of these functions must be selected as the starting point for execution of the script. The definition of the user-written function used as the starting point must contain csvin and csvout as the first positional arguments.

Note: Because the Python script will be imported, the following Python programming idiom will be ignored.

```
if __name__ == '__main__':
```

However, including it may be useful for testing outside of WebFOCUS.

- ❑ **Headers.** Using a header record listing the field names (instead of using positional index numbers) is not required in the sample input data file when creating the synonym for the Python script or when sending data to and retrieving data from the Python script. However, using header records, and, therefore, field names, in the Python script makes it more readable. The following syntax shows a sample of how to implement headers. In the csv.DictReader statement, use of a header record is implied:

```
fieldnames = ['addition']
reader = csv.DictReader(file_in, quoting=csv.QUOTE_NONNUMERIC)
writer = csv.DictWriter(file_out, quoting=csv.QUOTE_NONNUMERIC,
                        fieldnames=fieldnames)
writer.writeheader()
```

The recommendation is to use header records in the input and output .csv files. If you use sample data without a header, the field names in the generated metadata will be of the form FIELD_1 through FIELD_n.

The Adapter for Python also comes with a set of predefined statistical Python functions that you can easily invoke in WebFOCUS.

Reference: Prerequisites for Using the Adapter for Python

You can access the complete list of prerequisites when you configure the adapter by right-clicking the adapter name and clicking *Prerequisites*.

In the WebFOCUS Total Access Cloud, Python will be pre-installed and configured, and the required variables will be set.

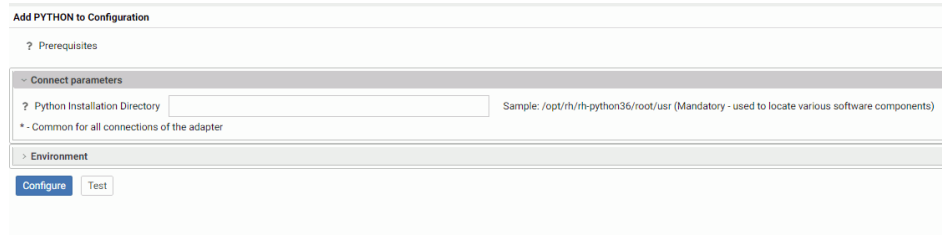
- ☐ Python must be installed on the same machine as the WebFOCUS Reporting Server, using the same bit size (32 or 64-bit).
- ☐ The Adapter for Python is available on Windows, UNIX, and Linux.
- ☐ The required Python release level is 3.6.x. Do not deselect the pip option in the install.
- ☐ The following packages must be installed in the following order (instructions are on the Prerequisites page):
 - ☐ numpy.
 - ☐ scipy (needs numpy).
 - ☐ scikit-learn (needs both of them and will add additional packages).
 - ☐ pandas.
- ☐ The system variables you may need to set are described on the Prerequisites page for the adapter.

Procedure: How to Configure the Adapter for Python

1. In the Reporting Server Web Console, go to the Connect to Data page by clicking *Connect to Data* on the sidebar.

2. Click *New Datasource* on the menu bar, the right-click *PYTHON*. Note that PYTHON can be found in the Statistics category on the *Available* drop-down list.

The Add PYTHON to Configuration page opens, as shown in the following image.



Add PYTHON to Configuration

? Prerequisites

Connect parameters

? Python Installation Directory Sample: /opt/rh/rh-python36/root/usr (Mandatory - used to locate various software components)

* - Common for all connections of the adapter

Environment

Configure Test

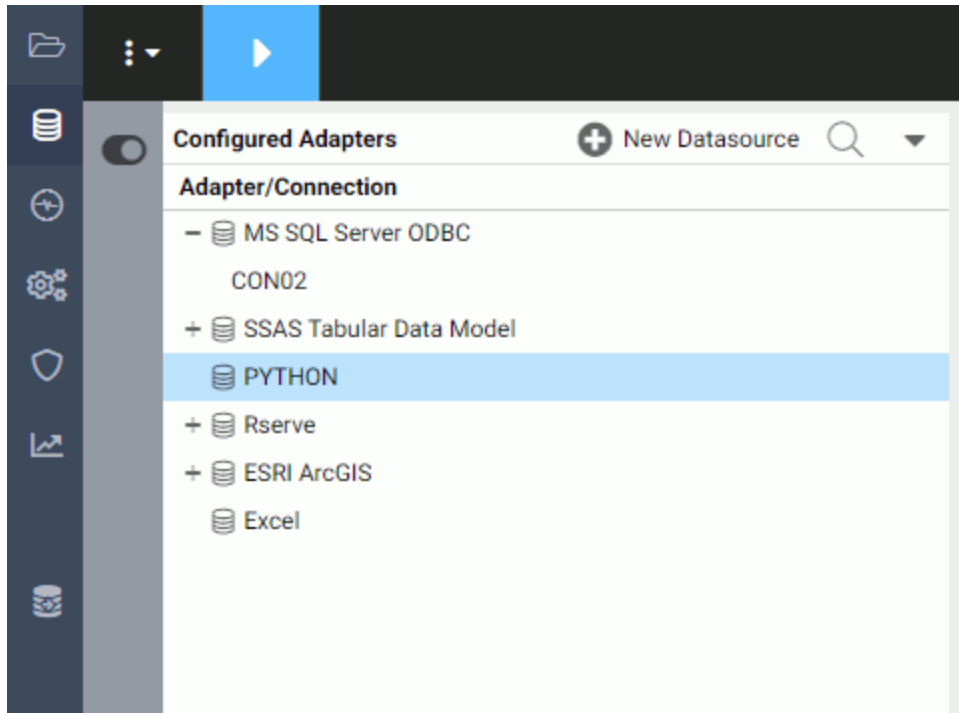
3. Enter the path to your Python installation directory, and click *Test*.

The following message displays for a successful configuration.

`Successful test for the Python environment`

4. Click *Configure*

The Adapter for Python is added to the list of Configured Adapters, as shown in the following image.



Procedure: How to Create a Synonym for a Python Function

Each Python script used with the Adapter for Python must have a synonym (metadata object) that describes the input fields and output fields of the script. If a Python script contains multiple user written functions, and you want to be able to use more than one function within the script as a starting point, you must create a separate synonym for each function within the script.

The synonym will be created using a sample file that contains only the fields that are input parameters for the script. A few rows of sample data are sufficient for the Adapter for Python to determine the appropriate data types and lengths of the parameters. The sample file must be a .csv file. The data in the file does not have to contain actual data, but it should represent the highest values for numeric fields and the longest lengths for alphanumeric values that will appear the actual data. The Master File will contain the list of input fields and output fields. The Access File will contain information about the script file and sample input file.

1. Right-click PYTHON on the list of configured adapters, and click *Create metadata objects* on the context menu.

The Create Synonym for Python frame opens, as shown in the following image.

Note that a metadata object is a synonym. The synonym for a Python function will consist of a Master File (which describes the input fields and output fields needed for running the function) and an Access File (which contains information about the sample data file and the script file).

2. Enter or select values for the following parameters.

PYTHON Script

Is the Python script. Enter an application directory name and script name, or click the ellipsis (...) to navigate to an application directory and select a script, then click *OK*.

The Python script will have the extension .py.

Function Name

Select the name of the (starting) function in the script file for which to create a synonym.

For example, the Python script named `arithmetic_example.py` contains the definition for the function `adder`:

```
def adder(csvin, csvout):
```


Select file with sample input data for the PYTHON Script

Open the file picker (...) to select the application directory and file that contains the sample data for creating the synonym. Click *OK*.

This file is used to determine the field names, data types, and lengths for the data sent to the Python script in *csvin*. If the sample file has no header record, the field names will be FIELD_1 through FIELD_*n*.

CSV files with header

If the input .csv file does not have a header row, uncheck *Input*. If the output .csv file should not have a header row, uncheck *Output*. The header requirements are contained in the function code.

For example, in the following sample code the input file contains no header record (fieldnames) and the fieldnames object is defined at runtime using the fieldnames argument for the reader. The output file will contain a header record, as defined in the fieldnames argument for the writer, and is written to the file using the statement `writer.writeheader()`:

```
with open(csvin, 'r', newline='') as file_in, \
    open(csvout, 'w', newline='') as file_out:
    reader = csv.DictReader(file_in, fieldnames=['input_field'],
                           quoting=csv.QUOTE_NONNUMERIC)
    writer = csv.DictWriter(file_out, fieldnames=['output_field'],
                           quoting=csv.QUOTE_NONNUMERIC)
    writer.writeheader()
```

Application

Enter the name of the application directory in which to create the synonym, or click the ellipsis (...) to navigate to an application directory, then click *OK*.

Synonym Name

Enter a name for the resulting synonym, or accept the default name.

3. Click the Create Synonym button on the ribbon.

The synonym is created in the specified application directory.

Note: You can generate sample Python scripts and data files in the Reporting Server Web Console. Create an application directory to contain the sample files, right-click the application folder, point to *New*, and click *Tutorials*. On the Tutorials page, select the *WebFOCUS - Retail Demo* tutorial, make sure *Create Python Example* is checked and that *Large* or *Medium* is selected for *Tutorial Data Volume Limit*. Click *Create* to create the demo files.

Syntax: How to Run a User-Written Python Function

```
PYTHON([app/]synonym, input1 [, input2 ...], output)
```

where:

`[app/]synonym`

Is the application and synonym name for the Python script.

`input1 [, input2 ...]`

Are the input arguments.

`output`

Is the output argument. This argument must match the name of a field in the OUTPUT_DATA segment in the Master File.

Example: Running a Python Script

The following is the `arithmetic_example_multiple_computes.py` Python script, which calculates four output fields, ADDITION, SUBTRACTION, MULTIPLICATION, and DIVISION in the function named *arithmetic*. All of the files for this example reside in an application directory named *python*.

```
# arithmetic_example_multiple_computes.py

import csv
import time

def arithmetic(csvin, csvout):

    with open(csvin, 'r', newline='') as file_in,\
        open(csvout, 'w', newline='') as file_out:

        fieldnames = ['addition', 'subtraction',
                      'multiplication', 'division']

        reader = csv.DictReader(file_in, quoting=csv.QUOTE_NONNUMERIC)
        writer = csv.DictWriter(file_out, quoting=csv.QUOTE_NONNUMERIC,
                                fieldnames=fieldnames)
        writer.writeheader()

        for row in reader:
            addition      = row['a_number'] + row['another_number']
            subtraction   = row['a_number'] - row['another_number']
            multiplication = row['a_number'] * row['another_number']
            division      = row['a_number'] / row['another_number']

            writer.writerow({'addition':      addition,
                             'subtraction':   subtraction,
                             'multiplication': multiplication,
                             'division':      division})
```

The .csv file with the sample data, arithmetic_sample_input.csv, has a header record and two data records to be used to determine the data types and lengths for the input arguments.

```
"a_number", "another_number"
1,1
100000,100000
```

The synonym creation frame for this script is shown in the following image.

Create Synonym for PYTHON

Create Synonym options

? PYTHON Script: python/arithmetic_example_multiple_computes.py ...

? Function Name: arithmetic

? File with sample input data for the PYTHON Script: python/arithmetic_sample_input.csv ...

? CSV files with header: ☒ Input ☒ Output

Customize data type mappings

? Application: bisamp ... ? Prefix: ? Suffix:

? Synonym Name: arithmetic_example_syn

The generated Master File (arithmetic_example_syn.mas) follows:

```
FILENAME=ARITHMETIC_EXAMPLE_SYN, SUFFIX=PYTHON , $
  SEGMENT=INPUT_DATA, SEGTYPE=S0, $
    FIELDNAME=A_NUMBER, ALIAS=a_number, USAGE=I11, ACTUAL=STRING,
      MISSING=ON,
      TITLE='a_number', $
    FIELDNAME=ANOTHER_NUMBER, ALIAS=another_number, USAGE=I11,
  ACTUAL=STRING,
    MISSING=ON,
    TITLE='another_number', $
  SEGMENT=OUTPUT_DATA, SEGTYPE=U, PARENT=INPUT_DATA, $
    FIELDNAME=ADDITION, ALIAS=addition, USAGE=D10.1, ACTUAL=STRING,
      MISSING=ON,
      TITLE='addition', $
    FIELDNAME=SUBTRACTION, ALIAS=subtraction, USAGE=D5.1, ACTUAL=STRING,
      MISSING=ON,
      TITLE='subtraction', $
    FIELDNAME=MULTIPLICATION, ALIAS=multiplication, USAGE=D15.1,
  ACTUAL=STRING,
    MISSING=ON,
    TITLE='multiplication', $
    FIELDNAME=DIVISION, ALIAS=division, USAGE=D5.1, ACTUAL=STRING,
      MISSING=ON,
      TITLE='division', $
```

The generated Access File (arithmetic_example_syn.acx) follows:

```
SEGNAME=INPUT_DATA,
MODNAME=python/arithmetic_example_multiple_computes.py,
FUNCTION=arithmetic,
PYTHON_INPUT_SAMPL=python/arithmetic_sample_input.csv,
INPUT_HEADER=YES,
OUTPUT_HEADER=YES, $
```

The following WebFOCUS procedure, sales_multiple_computes.fex calls the arithmetic function four times in order to get a value returned for each of the four outputs:

```
-* sales_multiple_computes.fex

TABLE FILE GGSales
SUM
  DOLLARS
  UNITS

COMPUTE Addition/D7      = PYTHON(python/arithmetic_example_syn,
  DOLLARS, UNITS, ADDITION);
COMPUTE Subtraction/D7   = PYTHON(python/arithmetic_example_syn,
  DOLLARS, UNITS, SUBTRACTION);
COMPUTE Multiplication/D16 = PYTHON(python/arithmetic_example_syn,
  DOLLARS, UNITS, MULTIPLICATION);
COMPUTE Division/D7.2    = PYTHON(python/arithmetic_example_syn,
  DOLLARS, UNITS, DIVISION);

WHERE RECORDLIMIT EQ 100
HEADING
  "Arithmetic Example, Multiple Computes"
  " "

ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

Arithmetic Example, Multiple Computes

<u>Dollar Sales</u>	<u>Unit Sales</u>	<u>Addition</u>	<u>Subtraction</u>	<u>Multiplication</u>	<u>Division</u>
1343927	105860	1,449,787	1,238,067	142,268,112,220	12.70

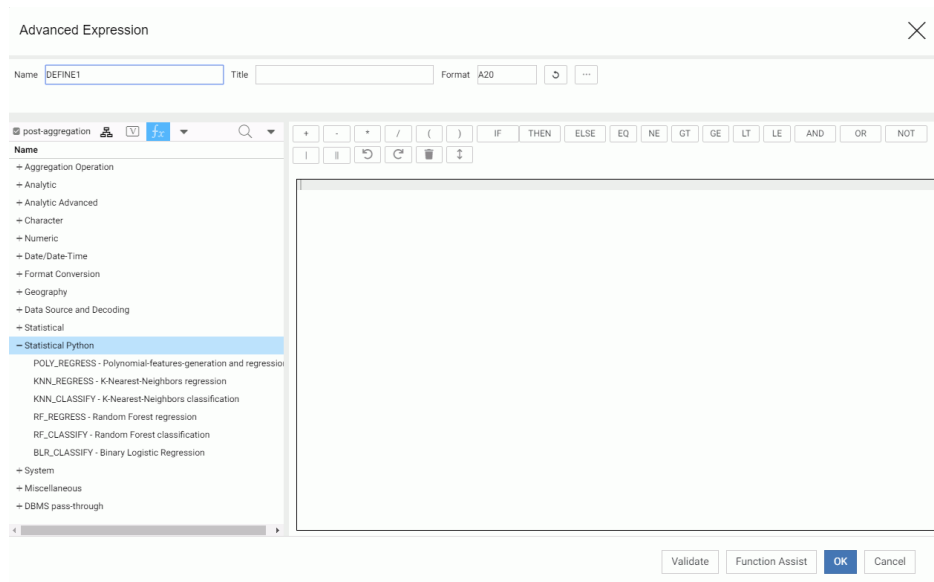
Python Functions Distributed With the Adapter for Python

The following python statistical functions are distributed with the Adapter for Python.

☐ BLR_CLASSIFY: Binary Logistic Regression

- ☐ KNN_CLASSIFY: K-Nearest-Neighbors Classification
- ☐ KNN_REGRESS: K-Nearest-Neighbors Regression
- ☐ POLY_REGRESS: Polynomial Regression
- ☐ RF_CLASSIFY: Random Forest Classification
- ☐ RF_REGRESS: Random Forest Regression

In the Reporting Server Synonym Editor, these functions are available when you create an expression. Right-click a measure field, point to *New Expression*, and click *Apply Function* or *Advanced Expression*. Click *post-aggregation* to gain access to the Python Statistical functions, as shown in the following image.



Double-click a function name to add the function to the expression and open a dialog box for entering parameters.

For more information about these functions, see the *Using Functions* manual.

DataMigrator Enhancements

This section describes the new features for DataMigrator.

DataMigrator represents a broad category of tools designed to facilitate and automate the extraction and integration of data. From source extraction through target load, data is transformed through the application of business rules. Once the transformation is complete, the data is loaded into table structures that have been optimized for a particular application.

For more information on any of these new features, see the *DataMigrator User's Guide*.

In this chapter:

- ☐ [Adapters](#)
 - ☐ [Calculators](#)
 - ☐ [Data Management Console](#)
 - ☐ [Data and Process Flows](#)
-

Adapters

The following section provides descriptions of new features for adapters.

Adapter for Excel: Creating a Synonym for Crosstabbed Data

Synonyms can now be automatically created in a crosstab format for data in Excel Worksheets.

When you create a synonym using the Adapter for Excel, there is now a check box labeled Crosstab for each Worksheet, as shown in the following image.

Select the Excel worksheet(s) to upload by selecting the corresponding checkbox(s).

Create Synonym for Excel

? Scan All rows

☐

? Number of header rows

1

(Enter number of topmost rows containing column headers)

Advanced

Customize data type mappings

Workbook: ibisamp/retail_cross.xlsx

<input type="checkbox"/> Default Name (editable)	Worksheet Name	Named Range	Rows/Columns	Crosstab	Multiple Measures
<input checked="" type="checkbox"/> sheet1	Sheet1		93/22	<input type="checkbox"/>	<input checked="" type="checkbox"/>

When selected, the header rows, up to the number specified, are un-pivoted to become data values.

The following shows a partial Excel Spreadsheet open in the Data Management Console. The headers include Product Category, a value for Product Category, Product Subcategory, and a value for Product Subcategory.

LOOPBACK: Related									
	ROWID	REPORT_RETAIL_WF_RETAIL_LITE_RPT_FEX	FIELD_2	FIELD_3	FIELD_4	FIELD_5	FIELD_6	FIELD_7	FIELD_8
1	2	.	Product,Category
2	3	.	Accessories	.	.	Camcorder	.	.	Computers
3	4	.	Product,Subcategory
4	5	.	Charger	Headphones	Universal Remote Controls	Handheld	Professional	Standard	Smartphone
5	6	Store Name
6	7	Amsterdam	2724	5738	4471	6276	339	4649	4881
7	8	Anchorage	575	1269	966	1309	72	1039	1169
8	9	Arlington	420	857	750	1037	67	774	849
9	10	Athens	39	81	47	98	4	65	47
10	11	Atlanta	820	1708	1356	1923	97	1475	1674
11	12	Bangalore	79	187	144	196	7	168	121
12	13	Bangkok	1	3	4	8	2	20	7
13	14	Barcelona	92	239	145	204	10	164	130
14	15	Beijing	3	12	7	11	.	8	2
15	16	Belfast	1	6	5	2	.	6	1
16	17	Berlin	2477	5286	4370	6118	292	4558	4901
17	18	Boise	163	249	182	305	18	229	205
18	19	Boston	397	1022	723	1026	55	771	899

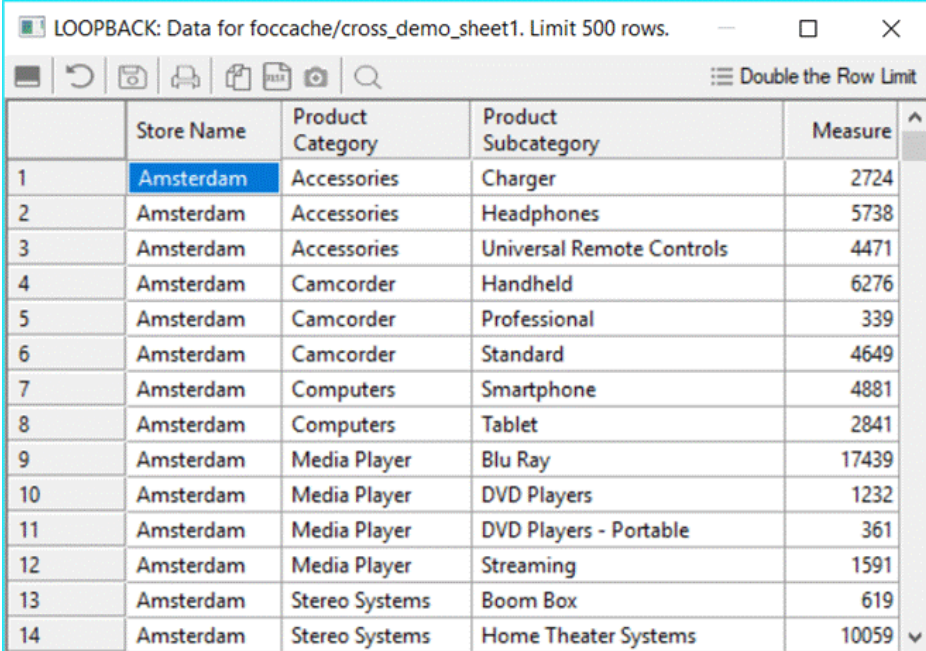
With Crosstab not checked, the resulting Master File has a field for column on the Spreadsheet. The following is a partial list of fields for the Spreadsheet:

```

FIELDNAME=PRODUCT_CATEGORY_ACCESSORIES_PRODUCT_SUBCATEGORY,
ALIAS='Product,Category_Accessories_Product,Subcategory', USAGE=A8V,
ACTUAL=A8V,
    MISSING=ON,
    TITLE='Product,Category_Accessories_Product,Subcategory', $
FIELDNAME=FIELD_3, ALIAS=FIELD_3, USAGE=A12V, ACTUAL=A12V,
    MISSING=ON,
    TITLE='FIELD_3', $
FIELDNAME=FIELD_4, ALIAS=FIELD_4, USAGE=A31V, ACTUAL=A31V,
    MISSING=ON,
    TITLE='FIELD_4', $
FIELDNAME=CAMCORDER, ALIAS=Camcorder, USAGE=A10V, ACTUAL=A10V,
    MISSING=ON,
    TITLE='Camcorder', $
FIELDNAME=FIELD_6, ALIAS=FIELD_6, USAGE=A15V, ACTUAL=A15V,
    MISSING=ON,
    TITLE='FIELD_6', $
FIELDNAME=FIELD_7, ALIAS=FIELD_7, USAGE=A10V, ACTUAL=A10V,
    MISSING=ON,
    TITLE='FIELD_7', $
FIELDNAME=COMPUTERS, ALIAS=Computers, USAGE=A12V, ACTUAL=A12V,
    MISSING=ON,
    TITLE='Computers', $
FIELDNAME=FIELD_9, ALIAS=FIELD_9, USAGE=A7V, ACTUAL=A7V,
    MISSING=ON,
    TITLE='FIELD_9', $
FIELDNAME=MEDIA_PLAYER, ALIAS='Media Player', USAGE=A8V, ACTUAL=A8V,
    MISSING=ON,
    TITLE='Media Player', $

```

With Crosstab checked, multiple measures unchecked, and five header rows specified, PRODUCT_CATEGORY becomes one dimension field, PRODUCT_SUBCATEGORY becomes a second dimension field, and the numeric values in the Spreadsheet become a measure field. The first column (containing store names) becomes an alphanumeric field. For example:



	Store Name	Product Category	Product Subcategory	Measure
1	Amsterdam	Accessories	Charger	2724
2	Amsterdam	Accessories	Headphones	5738
3	Amsterdam	Accessories	Universal Remote Controls	4471
4	Amsterdam	Camcorder	Handheld	6276
5	Amsterdam	Camcorder	Professional	339
6	Amsterdam	Camcorder	Standard	4649
7	Amsterdam	Computers	Smartphone	4881
8	Amsterdam	Computers	Tablet	2841
9	Amsterdam	Media Player	Blu Ray	17439
10	Amsterdam	Media Player	DVD Players	1232
11	Amsterdam	Media Player	DVD Players - Portable	361
12	Amsterdam	Media Player	Streaming	1591
13	Amsterdam	Stereo Systems	Boom Box	619
14	Amsterdam	Stereo Systems	Home Theater Systems	10059

The resulting Master File follows:

```
FILENAME=RETAIL_CROSS_CROSS_ON, SUFFIX=DIREXCEL,
DATASET=ibisamp/retail_cross.xlsx, BV_NAMESPACE=OFF, $
SEGMENT=RETAIL_CROSS_CROSS_ON, SEGTYPE=S0, $
  FIELDNAME=DHL1, ALIAS=DHL1, USAGE=A20V, ACTUAL=A20V,
  MISSING=ON,
  TITLE='DHL1', $
  FIELDNAME=DVL1_PRODUCT_CATEGORY, ALIAS='Product,Category',
  USAGE=A20V, ACTUAL=A20V,
  MISSING=ON,
  TITLE='Product,Category', $
  FIELDNAME=DVL2_PRODUCT_SUBCATEGORY, ALIAS='Product,Subcategory',
  USAGE=A31V, ACTUAL=A31V,
  MISSING=ON,
  TITLE='Product,Subcategory', $
  FIELDNAME=REPORT_RETAIL_WF_RETAIL_LITE_RPT_FEX,
  ALIAS='Report retail/wf_retail_lite_rpt.fex',
  USAGE=I9, ACTUAL=A11V,
  MISSING=ON,
  TITLE='Report retail/wf_retail_lite_rpt.fex', $
FOLDER=MG_RETAIL_CROSS_CROSS_ON, PARENT=.,
  DV_ROLE=MEASURE,
  DESCRIPTION='Measure Groups', $
FOLDER=RETAIL_CROSS_CROSS_ON, PARENT=MG_RETAIL_CROSS_CROSS_ON,
  DESCRIPTION='Retail_cross_cross_on', $
  FIELDNAME=REPORT_RETAIL_WF_RETAIL_LITE_RPT_FEX,
  BELONGS_TO_SEGMENT=RETAIL_CROSS_CROSS_ON, $
FOLDER=DIM_RETAIL_CROSS_CROSS_ON, PARENT=.,
  DV_ROLE=DIMENSION,
  DESCRIPTION='Dimensions', $
FOLDER=RETAIL_CROSS_CROSS_ON1, PARENT=DIM_RETAIL_CROSS_CROSS_ON,
  DESCRIPTION='Retail_cross_cross_on', $
  FIELDNAME=DHL1, BELONGS_TO_SEGMENT=RETAIL_CROSS_CROSS_ON, $
FOLDER=DVL, PARENT=RETAIL_CROSS_CROSS_ON1,
  DESCRIPTION='DVL', $
  FIELDNAME=DVL1_PRODUCT_CATEGORY,
  BELONGS_TO_SEGMENT=RETAIL_CROSS_CROSS_ON,
  DV_ROLE=LEVEL, $
  FIELDNAME=DVL2_PRODUCT_SUBCATEGORY,
  BELONGS_TO_SEGMENT=RETAIL_CROSS_CROSS_ON,
  DV_ROLE=LEVEL, $
```

You can now issue a request similar to the following:

```
TABLE FILE RETAIL_CROSS_CROSS_ON
PRINT REPORT_RETAIL_WF_RETAIL_LITE_RPT_FEX AS Measure
BY DHL1 AS Store
BY DVL1_PRODUCT_CATEGORY AS Category
BY DVL2_PRODUCT_SUBCATEGORY AS Subcategory
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

Partial output is shown in the following image.

<u>Store</u>	<u>Category</u>	<u>Subcategory</u>	<u>Measure</u>
Amsterdam	Accessories	Charger	2724
		Headphones	5738
		Universal Remote Controls	4471
	Camcorder	Handheld	6276
		Professional	339
		Standard	4649
	Computers	Smartphone	4881
		Tablet	2841
	Media Player	Blu Ray	17439
		DVD Players	1232
		DVD Players - Portable	361
		Streaming	1591
	Stereo Systems	Boom Box	619
		Home Theater Systems	10059
		Receivers	3833
		Speaker Kits	6257
		iPod Docking Station	7939
	Televisions	CRT TV	333
		Flat Panel TV	2293
		Portable TV	517
	Video Production	Video Editing	4897
Anchorage	Accessories	Charger	575
		Headphones	1269
		Universal Remote Controls	966
	Camcorder	Handheld	1309
		Professional	72
		Standard	1039
	Computers	Smartphone	1169
		Tablet	962
	Media Player	Blu Ray	3588
		Streaming	343
	Stereo Systems	Home Theater Systems	2191
		Receivers	805
		Speaker Kits	1441
		iPod Docking Station	1651
	Televisions	Flat Panel TV	459
	Video Production	Video Editing	1066

Adapter for Salesforce.com: Support for Bulk Retrieval

In prior releases, the Adapter for Salesforce.com used a REST API to issue SOQL (Salesforce Object Query Language) commands to retrieve data. This technique is acceptable for direct reporting against Salesforce or for migrating modest data volumes.

However, Salesforce.com recommends using the BULK API capability *bulk query* to efficiently query large data sets and reduce the number of API requests

Bulk Query can now be enabled from the settings panel for the adapter under *Bulk Services* by selecting ON from the new BULKQUERY pull-down menu. Additional options allow you to enable PK (Primary Key) chunking and to set the chunk size.

Bulk query can also be enabled or disabled with the following command:

```
ENGINE SFDC SET BULKQUERY {ON|OFF}
```

When this option is selected, queries that meet the salesforce.com requirements (such as no aggregation) use bulk query.

To extract extremely large data sets, the Bulk API capability *PK Chunking* can be enabled with the following command:

```
ENGINE SFDC SET PKCHUNKING {ON|OFF}
```

When this option is enabled, the Primary Key of the object is used to retrieve data in chunks, allowing retrieval of larger data volumes.

When PK chunking is enabled, the chunk size can be set using the following command:

```
ENGINE SFDC SET PKCHUNKSIZE {n|100000}
```

where:

n

Where *n* is an integer between zero (0) and 250000. When PKCHUNKSIZE is set to zero or not set, the default chunk size (100,000) is used.

Adapter for Snowflake Cloud Data Warehouse

The Adapter for Snowflake Cloud Data Warehouse is new in this release. It can be found in the SQL category.

This adapter accesses data stored in the Snowflake Cloud through SQL query operations.

While Snowflake provides both ODBC and JCBC drivers, in the initial release, only the JDBC version is certified and provides Extended Bulk Load functionality.

Adapter for Stratio Crossdata

A new adapter for Crossdata provides access to data using the Stratio JDBC driver. This adapter is in the SQL category.

Calculators

The following section provides descriptions of new features for calculators.

Additional Aggregate Functions Available

Two additional aggregate functions, Median and Mode, are now available from the Data Management Console. The functions can be found in the Aggregate drop-down menu for the Column Selection dialog box. To access this dialog box, right-click the SQL object in a flow and click *Column Selection*.

GET_TOKEN Function Now Available

The GET_TOKEN function is now available from the Data Management Console calculators.

GET_TOKEN extracts a token (substring) based on a string that can contain multiple characters, each of which represents a single-character delimiter.

GET_TOKEN can be optimized if there is a single delimiter character, not a string containing multiple delimiter characters.

INITCAP Function Now Available

The INITCAP function is now available from the Data Management Console calculators. INITCAP capitalizes the first letter of each word in an input string and makes all other letters lowercase. A word starts at the beginning of the string, after a blank space or after a special character.

New FOCUS Functions

Two FOCUS functions, COALESCE and NULLIF, are now available in calculators in the Data Management Console.

COALESCE evaluates the arguments in order and returns the first argument that is not MISSING. NULLIF returns a missing value when its parameters are equal. If they are not equal, it returns the first value.

Data Management Console

The following section provides descriptions of new features for the Data Management Console (DMC).

Data and Process Flows

The following section provides descriptions of new features for data and process flows.

Formatted Files With Excel Format Now Use .XLSX

As of Release 7709, DataMigrator customers that create a formatted file with a specified format of EXL2K (excel) can now create Excel files in the .xlsx format that is found in Excel 2007 and later. You can create an Excel worksheet by adding a new target to a data flow, selecting Formatted File as the adapter, and selecting EXL2K from the Format drop-down menu.

Enhancements to Join Profiling

As of Release 7709, along with the Join Profiling report option, there is a new Join Profiling Chart available. This chart shows a horizontal bar chart that represents the number of rows that would be returned from each table when a Left Outer, Right Outer, and Inner Join is run. You can access this new chart by clicking the *Join Editor* button in the Join Editor dialog box

Resource Analyzer and Resource Governor Enhancements

This section describes the new features for Resource Analyzer.

Resource Analyzer provides Information Systems (IS) organizations with the ability to manage the growing volume and unpredictable nature of ad hoc data access.

Resource Governor controls monitoring, system configuration parameters, and governing rules. It provides preemptive governing for requests issued to both relational and non-relational data sources.

Together, Resource Analyzer and its partner product, Resource Governor, are designed specifically to help IS organizations analyze and control end user data access.

In this chapter:

- ☐ [New Audit Report](#)
 - ☐ [New Repository Load Type Options](#)
 - ☐ [New Scheduling Report Option](#)
-

New Audit Report

A new audit report, entitled ACI Summary, has been added to Resource Management. This report provides information about changes to server configuration files and user application files. You can run the report based on user name, the type of file access allowed, and by file category. The ACI Summary report can be accessed by expanding the *Reports* folder in your Resource Management environment and running *ACI Monitoring*.

New Repository Load Type Options

Resource Management now allows you to indicate the load method that will be used to add log data to the data repository. This new setting lets you choose between loading data into a temporary on your RDBMS server using the connection adapter that was selected for the repository via either an available bulk load utility, or the TABLE command. After the data is loaded into the temporary table, regardless of which method is used, the MERGE command is used to merge the data from the temporary table into the data repository. This option is set while configuring your Resource Management environment, and is accessed from the Configure Resource Management pane.

New Scheduling Report Option

Resource Management now allows you to schedule a report to run and be automatically distributed. You can schedule the report to run at various intervals, ranging from just once to recurring on specific days. Once a scheduled report is run, you have the ability to either download or view the report content. This option is accessed by right-clicking any report and pointing to *Schedule and E-Mail*.

Reporting Language Enhancements

WebFOCUS is a complete information control system with comprehensive features for retrieving and analyzing data. It enables you to create reports quickly and easily. It also provides facilities for creating highly complex reports, but its strength lies in the simplicity of the request language. You can begin with simple queries and progress to complex reports as you learn about additional facilities.

In this chapter:

- ❑ [Expanded Functionality With BY TOTAL and ACROSS](#)
- ❑ [COALESCE: Returning the First Non-Missing Value](#)
- ❑ [COMPACTFORMAT: Displaying Numbers in an Abbreviated Format](#)
- ❑ [FPRINT: Displaying a Value in a Specified Format](#)
- ❑ [GET_TOKEN: Extracting a Token Based on a String of Delimiters](#)
- ❑ [GIS_REVERSE_COORDINATE: Returning a Geographic Component](#)
- ❑ [INITCAP: Capitalizing the First Letter of Each Word in a String](#)
- ❑ [NULLIF: Returning a Null Value When Parameters Are Equal](#)
- ❑ [Statistical Python Functions](#)
- ❑ [Using IF-THEN-ELSE Logic in an Expression](#)
- ❑ [Generating Descriptive Column Titles for Prefixed Fields](#)
- ❑ [Format Display Options for Abbreviating Numeric Values](#)
- ❑ [Support for C-Style In-Line Comments](#)
- ❑ [SET PCTFORMAT: Displaying a Percent Sign for Percent Prefix Operators](#)

Expanded Functionality With BY TOTAL and ACROSS

The BY TOTAL phrase is now supported for a calculated value (generated using the COMPUTE command) in a request that contains an ACROSS phrase.

Example: Using BY TOTAL on a Calculated Value With an ACROSS Phrase

The following request creates the calculated value PROFIT and uses it in the BY TOTAL phrase. The request also has an ACROSS RATING phrase.

```
TABLE FILE MOVIES
SUM LISTPR WHOLESALPR
COMPUTE
PROFIT = LISTPR - WHOLESALPR;
BY CATEGORY
BY TOTAL PROFIT
ACROSS RATING
WHERE RATING NE 'NR' OR 'R'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *GRID = OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

CATEGORY	RATING G		PG		PG13		PROFIT	
	LISTPR	WHOLESALPR	LISTPR	WHOLESALPR	LISTPR	WHOLESALPR	LISTPR	WHOLESALPR
ACTION	13.92	-	34.90	20.98	13.92	-	-	-
CHILDREN	47.40	101.89	47.40	-	-	-	-	-
CLASSIC	48.96	89.95	48.96	-	-	-	-	-
COMEDY	58.12	-	74.83	46.70	28.13	59.99	30.00	29.99
FOREIGN	64.63	19.98	119.90	62.00	57.90	-	-	-
MUSICALS	25.93	29.95	19.98	13.99	5.99	14.98	9.99	4.99
MYSTERY	32.94	-	39.96	18.00	21.96	19.98	9.00	10.98
SCI/FI	18.94	-	-	-	54.93	35.99	18.94	-

COALESCE: Returning the First Non-Missing Value

Given a list of arguments, COALESCE returns the value of the first argument that is not missing. If all argument values are missing, it returns a missing value if MISSING is ON. Otherwise it returns a default value (zero or blank).

Syntax: How to Return the First Non-Missing Value

```
COALESCE(arg1, arg2, ...)
```

where:

arg1, arg2, ...

Any field, expression, or constant. The arguments should all be either numeric or alphanumeric.

Are the input parameters that are tested for missing values.

The output data type is the same as the input data types.

Example: Returning the First Non-Missing Value

This example uses the SALES data source with missing values added. The missing values are added by the following procedure named SALEMISS:

```
MODIFY FILE SALES
  FIXFORM STORE/4 DATE/5 PROD/4
  FIXFORM UNIT/3 RETAIL/5 DELIVER/3
  FIXFORM OPEN/3 RETURNS/C2 DAMAGED/C2
  MATCH STORE
    ON NOMATCH REJECT
    ON MATCH CONTINUE
  MATCH DATE
    ON NOMATCH REJECT
    ON MATCH CONTINUE
  MATCH PROD_CODE
    ON NOMATCH INCLUDE
    ON MATCH REJECT
DATA
14Z 1017 C13 15 1.99 35 30 6
14Z 1017 C14 18 2.05 30 25 4
14Z 1017 E2 33 0.99 45 40
END
```

The following request uses COALESCE to return the first non-missing value:

```
TABLE FILE SALES
PRINT DAMAGED RETURNS RETAIL_PRICE
COMPUTE
COAL1/D12.2 MISSING ON = COALESCE(DAMAGED, RETURNS, RETAIL_PRICE);
BY STORE_CODE
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image. The value of DAMAGED is returned, if it is not missing. If DAMAGED is missing, the value of RETURNS is returned, if it is not missing. If they are both missing, the value of RETAIL_PRICE is returned.

<u>STORE CODE</u>	<u>DAMAGED</u>	<u>RETURNS</u>	<u>RETAIL PRICE</u>	<u>COAL1</u>
14B	6	10	\$.95	6.00
	3	3	\$1.29	3.00
	1	2	\$1.89	1.00
	0	3	\$1.99	.00
	4	5	\$2.39	4.00
	0	0	\$2.19	.00
	4	9	\$.99	4.00
	9	8	\$1.09	9.00
14Z	3	2	\$.85	3.00
	1	2	\$1.89	1.00
	1	0	\$1.99	1.00
	6	.	\$1.99	6.00
	.	4	\$2.05	4.00
	0	0	\$2.09	.00
	2	3	\$2.09	2.00
	7	4	\$.89	7.00
77F	.	.	\$.99	.99
	2	4	\$1.09	2.00
	1	1	\$2.09	1.00
K1	0	0	\$2.49	.00
	0	1	\$1.49	.00
	1	1	\$.99	1.00

COMPACTFORMAT: Displaying Numbers in an Abbreviated Format

COMPACTFORMAT displays numbers in a compact format where:

- ☐ K is an abbreviation for thousands.
- ☐ M is an abbreviation for millions.
- ☐ B is an abbreviation for billions.

❑ T is an abbreviation for trillions.

COMPACTFORMAT computes which abbreviation to use, based on the order of magnitude of the largest value in the column. The returned value is an alphanumeric string. Attempting to output this value to a numeric format will result in a format error, and the value zero (0) will be displayed.

Syntax: How to Display Numbers in an Abbreviated Format

`COMPACTFORMAT(input)`

where:

input

Is the name of a numeric field.

Example: Displaying Numbers in an Abbreviated Format

The following example uses the COMPACTFORMAT function to abbreviate the display of the summed values of the DAYSDELAYED, QUANTITY_SOLD, and COGS_US fields.

```
TABLE FILE WF_RETAIL_LITE
SUM DAYSDELAYED QUANTITY_SOLD COGS_US
COMPUTE
CDAYS/A30= COMPACTFORMAT(DAYSDELAYED);
CQUANT/A30= COMPACTFORMAT(QUANTITY_SOLD);
CCOGS/A30= COMPACTFORMAT(COGS_US);
ON TABLE SET PAGE NOPAGE
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

<u>Days</u> <u>Delayed</u>	<u>Quantity</u> <u>Sold</u>	<u>Cost of Goods</u>	<u>CDAYS</u>	<u>CQUANT</u>	<u>CCOGS</u>
5,355	13,923	\$2,950,358.00	5,355	14K	\$3M

FPRINT: Displaying a Value in a Specified Format

Given an output format, the simplified conversion function FPRINT converts a value to alphanumeric format for display in that format.

Note: A legacy FPRINT function also exists and is still supported. The legacy function has an additional argument for the name or format of the returned value.

Syntax: **How to Display a Value in a Specified Format**

```
FPRINT(value, 'out_format') 
```

where:

value

Any data type

Is the value to be converted.

'out_format'

Fixed length alphanumeric

Is the display format.

Example: **Displaying a Value in a Specified Format**

The following request displays COGS_US as format 'D9M', and TIME_DATE as format 'YYMtrD', by converting them to alphanumeric using FPRINT.

```
DEFINE FILE WF_RETAIL_LITE
COGS_A/A25 = FPRINT(COGS_US, 'D9M');
DATE1/A25 = FPRINT(TIME_DATE, 'YYMtrD');
END
TABLE FILE WF_RETAIL_LITE
PRINT LST.COGS_US COGS_A DATE1
BY TIME_DATE
WHERE RECORDLIMIT EQ 10
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```


The output is shown in the following image.

<u>Sale</u>	<u>LST</u>		
<u>Date</u>	<u>Cost of Goods</u>	<u>COGS_A</u>	<u>DATE1</u>
01/03/2009	\$234.00	\$234	2009, January 3
	\$46.00	\$46	2009, January 3
	\$380.00	\$380	2009, January 3
	\$374.00	\$374	2009, January 3
	\$310.00	\$310	2009, January 3
	\$83.00	\$83	2009, January 3
	\$312.00	\$312	2009, January 3
	\$548.00	\$548	2009, January 3
	\$400.00	\$400	2009, January 3
	\$131.00	\$131	2009, January 3

GET_TOKEN: Extracting a Token Based on a String of Delimiters

GET_TOKEN extracts a token (substring) based on a string that can contain multiple characters, each of which represents a single-character delimiter.

GET_TOKEN can be optimized if there is a single delimiter character, not a string containing multiple delimiter characters.

Syntax: How to Extract a Token Based on a String of Delimiters

`GET_TOKEN(string, delimiter_string, occurrence)`

where:

string

Alphanumeric

Is the input string from which the token will be extracted. This can be an alphanumeric field or constant.

delimiter_string

Alphanumeric constant

Is a string that contains the list of delimiter characters. For example, '; , ' contains three delimiter characters, semi-colon, blank space, and comma.

occurrence

Integer constant

Is a positive integer that specifies the token to be extracted. A negative integer will be accepted in the syntax, but will not extract a token. The value zero (0) is not supported.

Example: **Extracting a Token Based on a String of Delimiters**

The following request defines an input string and two tokens based on a list of delimiters that contains the characters comma (,), semicolon (;), and slash (/).

```
DEFINE FILE EMPLOYEE
InputString/A20 = 'ABC,DEF;GHI/JKL';
FirstToken/A20 WITH DEPARTMENT = GET_TOKEN(InputString, ',;/', 1);
FourthToken/A20 WITH DEPARTMENT = GET_TOKEN(InputString, ',;/', 4);
ENDTABLE FILE EMPLOYEE
PRINT InputString FirstToken FourthToken
WHERE READLIMIT EQ 1
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID = OFF,$
END
```

The output is shown in the following image. The first token was extracted using the comma (,) as the delimiter. The fourth token was extracted using the slash (/) as the delimiter.

<u>InputString</u>	<u>FirstToken</u>	<u>FourthToken</u>
ABC,DEF;GHI/JKL	ABC	JKL

GIS_REVERSE_COORDINATE: Returning a Geographic Component

Given longitude and latitude values and the name of a geographic component, GIS_REVERSE_COORDINATE returns the specified geographic component values associated with those coordinates.

Syntax: **How to Return a Geographic Component**

```
GIS_REVERSE_COORDINATE(longitude, latitude, component)
```

where:

longitude

Numeric

Is the longitude of the component to return.

latitude

Numeric

Is the latitude of the component to return.

latitude

Keyword

Is one of the following components:

- ☐ MATCH_ADDRESS, which returns the matching address.
- ☐ METROAREA, which returns the metro area name.
- ☐ REGION, which returns the region name.
- ☐ SUBREGION, which returns the subregion name.
- ☐ CITY, which returns the city name.
- ☐ POSTAL, which returns the postal code.

The value is returned as text and can be assigned to a field with text or alphanumeric (fixed or variable length) format.

Example: Returning Geographic Components Associated With Coordinates

The following request uses city longitude and city latitude to return the matching address, postal code, region, and subregion.

```
TABLE FILE WF_RETAIL_GEOGRAPHY
SUM FST.CITY_LONGITUDE AS Longitude FST.CITY_LATITUDE AS Latitude
COMPUTE MatchingAddress/A250 = GIS_REVERSE_COORDINATE(CITY_LONGITUDE,
CITY_LATITUDE, MATCH_ADDRESS);
       PostalCode/A250 = GIS_REVERSE_COORDINATE(CITY_LONGITUDE,
CITY_LATITUDE, POSTAL);
       Region/A250 = GIS_REVERSE_COORDINATE(CITY_LONGITUDE, CITY_LATITUDE,
REGION);
       Subregion/A250 = GIS_REVERSE_COORDINATE(CITY_LONGITUDE,
CITY_LATITUDE, SUBREGION);
BY CITY_NAME AS City
WHERE COUNTRY_NAME EQ 'United States'
WHERE TOTAL PostalCode NE ' '
WHERE RECORDLIMIT EQ 20
ON TABLE SET PAGE NOLEAD
END
```

INITCAP: Capitalizing the First Letter of Each Word in a String

The output is shown in the following image.

City	Longitude	Latitude	MatchingAddress	PostalCode	Region	Subregion
Annapolis	-76.54540000	38.98790000	Annapolis Mall, Annapolis, Maryland, 21401	21401	Maryland	Anne Arundel County
Baton Rouge	-91.09780000	30.44990000	233 E Parkland Dr, Baton Rouge, Louisiana, 70806	70806	Louisiana	East Baton Rouge Parish
Cincinnati	-84.45690000	39.16200000	2070-2098 Elm Ave, Cincinnati, Ohio, 45212	45212	Ohio	Hamilton County
Daytona Beach	-81.04740000	29.19310000	511 S Clyde Morris Blvd, Daytona Beach, Florida, 32114	32114	Florida	Volusia County
Detroit	-83.05980000	42.34630000	133 Davenport St, Detroit, Michigan, 48201	48201	Michigan	Wayne County
Harrington Park	-73.98330000	40.99070000	247 Lynn St, Harrington Park, New Jersey, 07640	07640	New Jersey	Bergen County
Johnston	-93.72040000	41.70310000	Camp Dodge	50131	Iowa	Polk County
Lake Mary	-81.33970000	28.75780000	127-129 E Plantation Blvd, Lake Mary, Florida, 32746	32746	Florida	Seminole County
Laredo	-99.50350000	27.51330000	1501-1599 Santa Ursula Ave, Laredo, Texas, 78040	78040	Texas	Webb County
Latham	-73.78040000	42.75260000	1 Lear Jet Ln, Latham, New York, 12110	12110	New York	Albany County
Louisville	-85.69180000	38.20850000	2714 Lamont Rd, Louisville, Kentucky, 40205	40205	Kentucky	Jefferson County
Medley	-80.38390000	25.85880000	33178, Miami, Florida	33178	Florida	Miami-Dade County
North Kansas City	-94.56220000	39.13000000	1201-1499 Quebec St, Kansas City, Missouri, 64116	64116	Missouri	Clay County
Rochester	-77.61560000	43.15480000	1-35 Plymouth Ave S, Rochester, New York, 14614	14614	New York	Monroe County
Waco	-97.13820000	31.55100000	1200 Austin Ave, Waco, Texas, 76701	76701	Texas	McLennan County

INITCAP: Capitalizing the First Letter of Each Word in a String

INITCAP capitalizes the first letter of each word in an input string and makes all other letters lowercase. A word starts at the beginning of the string, after a blank space or after a special character.

Syntax: How to Capitalize the First Letter of Each Word in a String

```
INITCAP(input_string)
```

where:

input_string

Alphanumeric

Is the string to capitalize.

Example: Capitalizing the First Letter of Each Word in a String

The following request changes the last names in the EMPLOYEE data source to initial caps and capitalizes the first letter after each blank or special character in the NewName field.

```
TABLE FILE EMPLOYEE
PRINT LAST_NAME AND COMPUTE
Caps1/A30 = INITCAP(LAST_NAME);
NewName/A30 = 'abc,def!ghi' jKL MNO';
Caps2/A30 = INITCAP(NewName);
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

<u>LAST_NAME</u>	<u>Caps1</u>	<u>NewName</u>	<u>Caps2</u>
STEVENS	Stevens	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
SMITH	Smith	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
JONES	Jones	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
SMITH	Smith	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
BANNING	Banning	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
IRVING	Irving	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
ROMANS	Romans	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
MCCOY	Mccoy	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
BLACKWOOD	Blackwood	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
MCKNIGHT	Mcknight	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
GREENSPAN	Greenspan	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno
CROSS	Cross	abc,def!ghi'jKL MNO	Abc,Def!Ghi'Jkl Mno

NULLIF: Returning a Null Value When Parameters Are Equal

NULLIF returns a null (missing) value when its parameters are equal. If they are not equal, it returns the first value. The field to which the value is returned should have MISSING ON.

Syntax: How to Return a Null Value for Equal Parameters

`NULLIF(arg1, arg2)`

where:

arg1, *arg2*

Any type of field, constant, or expression.

Are the input parameters that are tested for equality. They must either both be numeric or both be alphanumeric.

The output data type is the same as the input data types.

Example: **Testing for Equal Parameters**

The following request uses NULLIF to test the DAMAGED and RETURNS field values for equality.

```
DEFINE FILE SALES
NULL1/I4 MISSING ON = NULLIF(DAMAGED, RETURNS);
END
TABLE FILE SALES
PRINT DAMAGED RETURNS NULL1
BY STORE_CODE
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
END
```

The output is shown in the following image.

<u>STORE CODE</u>	<u>DAMAGED</u>	<u>RETURNS</u>	<u>NULL1</u>
14B	6	10	6
	3	3	.
	1	2	1
	0	3	0
	4	5	4
	0	0	.
	4	9	4
	9	8	9
14Z	3	2	3
	1	2	1
	1	0	1
	0	0	.
	2	3	2
	7	4	7
	2	4	2
77F	1	1	.
	0	0	.
K1	0	1	0
	1	1	.

Statistical Python Functions

The Statistical Python functions are FOCUS functions implemented as Python scripts. These Python scripts take advantage of Python packages such as scipy, numpy, scikit-learn, and pandas, which extends the Python capabilities to machine learning.

Before running the functions, you must configure the Adapter for Python. For information about configuring the adapter, see the *Adapter Administration* manual.

The statistical Python functions perform regression and classification using a variety of machine learning methods. The Python scripts perform a sequence of conventional machine learning tasks including scaling of the data where appropriate. They are built around a grid search with cross-validation. That is, some hyper-parameters (parameters that influence the learning process, but that are not model parameters) are identified, and models are built using a number of values for each hyper-parameter, in order to determine the optimal values. To determine optimality, cross-validation is used, which ensures that the performance is measured on a validation-subset of the data that is distinct from the training-subset. Rows with missing values in the target-column are not used for training and validation, but a predicted value is computed by the trained model.

BLR_CLASSIFY: Binary Logistic Regression

Binary logistic regression finds the best linear separation between two classes in the space spanned by the predictors. The target variable has two values (0 and 1), corresponding to the two classes. The predicted value is either a class assignment (0 or 1) or the probability of belonging to class 1.

***Syntax:* How to Calculate a Binary Logistic Regression**

```
BLR_CLASSIFY(options, proba_flag,  
             predictor_field1[, predictor_field2, ...] target_field)
```

where:

options

Reserved for future use.

proba_flag

Keyword

Indicates whether to display probabilities. Valid values are PROBA, to display the probability of belonging to class 1, and NO_PROBA, to display class assignment.

- ☐ When *proba_flag* has the value NO_PROBA, this function returns a binary-class prediction for Y (Y=0 or 1).
- ☐ When *proba_flag* has the value PROBA, this function returns the probability Y ($0 \leq Y \leq 1$), that (X0, X1, . . .) belongs to class 1.

```
predictor_field1[, predictor_field2, ...]
```

Numeric

Are one or more predictor field names.

target_field

Numeric

Is the target field.

Example: Using BLR_CLASSIFY to Predict Education Code

The DEFINE FILE command in the following request creates two education levels, doctorate level (value 1) and no college education (value 0). It also creates other virtual fields with numeric formats that can be used by the function. The TABLE request uses BLR_CLASSIFY to compute the probability that the education level belongs to class 1, doctorate level education, using predictors age, income, population range, and gender.

```
DEFINE FILE WF_RETAIL
  BINARY_EDUC_LEVEL/I2 =
  DECODE WF_RETAIL_EDUCATION_CUSTOMER.EDUCATION_DEGREE_TYPE (
    'Professional Doctorates' 1,
    'Doctorates' 1,
    'High School' 0,
    'No High School' 0,
    ELSE -1 );
  POP_CODE/I2 =
  DECODE WF_RETAIL_GEOGRAPHY_CUSTOMER.CITY_POPULATION_RANGE (
    'H: 100,001 - 250,000' 1,
    'I: 250,001 - 1,000,000' 2,
    'J: 1,000,001 - 10,000,000' 3,
    'K: 10,000,001 - 50,000,000' 4,
    ELSE 0 );
  GENDER_CODE/I2 =
  DECODE WF_RETAIL_CUSTOMER.GENDER (
    'M' 1, 'F' 0 );
END

TABLE FILE WF_RETAIL
PRINT
  ID_CUSTOMER
  BINARY_EDUC_LEVEL
  COMPUTE PREDICTED_PROB/D6.4=BLR_CLASSIFY(' ',PROBA,
                                           AGE,
                                           INCOME,
                                           POP_CODE,
                                           GENDER_CODE,
                                           BINARY_EDUC_LEVEL);

WHERE BINARY_EDUC_LEVEL NE -1
WHERE POP_CODE NE 0
WHERE INCOME GT 12001.00
WHERE OUTPUTLIMIT EQ 12
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The partial output is shown in the following image.

<u>ID Customer</u>	<u>BINARY_EDUC_LEVEL</u>	<u>PREDICTED_PROB</u>
99444	1	.9985
237172	1	.7626
59823	1	.9640
121373	1	.9989
106534	1	.9996
104894	1	.9989
6721	0	.0915
216378	0	.0120
227169	0	.0276
133681	1	.9823
137550	0	.2026
1896	0	.1329

KNN_CLASSIFY: K-Nearest Neighbors Classification

K-nearest neighbors classification is a method for assigning a class membership to a data point in the space spanned by the predictors. The classification is done by assigning the class most common among its k nearest neighbors. This method requires having a distance definition in this space.

***Reference:* Calculate a K-Nearest Neighbors Classification**

```
KNN_CLASSIFY(options, neighbors, power,  
             predictor_field1[, predictor_field2, ...] target_field)
```

where:

options

Reserved for future use.

neighbors

Integer

Is the number of nearest neighbors to participate in the prediction.

power

Integer

Is the power (p) of the L^p-distance.

- ☐ power=1 calculates the distance as the sum of the absolute values of the differences between the coordinates (Manhattan distance).
- ☐ power=2 calculates the distance as the square root of the sum of the squares of the differences between the coordinates (Euclidean distance).

predictor_field1[, predictor_field2, ...]

Numeric

Are one or more predictor field names.

target_field

Numeric

Is the target field.

Example: Predicting Education Level Using KNN_CLASSIFY

The following request uses KNN_CLASSIFY to predict education level using the 20 nearest neighbors and Euclidean distance, with predictors age, income, population range, and gender. The DEFINE FILE command creates virtual fields with correct numeric formats for use in the function. The data set includes rows with missing target values.

```

DEFINE FILE WF_RETAIL
POP_CODE/I2 =
  DECODE WF_RETAIL_GEOGRAPHY_CUSTOMER.CITY_POPULATION_RANGE (
    'H: 100,001 - 250,000'      1,
    'I: 250,001 - 1,000,000'    2,
    'J: 1,000,001 - 10,000,000' 3,
    'K: 10,000,001 - 50,000,000' 4,
    ELSE 0 );
GENDER_CODE/I2 =
  DECODE WF_RETAIL_CUSTOMER.GENDER (
    'M' 1, 'F' 0 );
END

```

```

TABLE FILE WF_RETAIL
PRINT ID_CUSTOMER AGE INCOME DEGREE_M EDUC_LEVEL_M
COMPUTE ED_PRED/I2=KNN_CLASSIFY(' ',20,2,
                                AGE,
                                INCOME,
                                POP_CODE,
                                GENDER_CODE,
                                EDUC_LEVEL_M);

WHERE POP_CODE NE 0;
WHERE OUTPUTLIMIT IS 8;
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END

```

Partial output is shown in the following image.

<u>ID Customer</u>	<u>Age</u>	<u>Income</u>	Customer Degree <u>w/Missings</u>	Customer Education Level <u>w/Missings</u>	<u>ED_PRED</u>
230444	41	117,209.96	Master's Degrees	6	6
11162	56	102,553.99	Bachelor's Degree	5	5
12673	66	97,259.44	Associate Degree	4	4
13567	66	86,283.38	Non Traditional Degree	1	3
13902	46	140,601.15	Master's Degrees	6	6
99444	62	187,736.31	.	.	8
237172	27	69,740.65	Professional Doctorates	8	8
207543	68	129,006.18	Master's Degrees	6	7

KNN_REGRESS: K-Nearest Neighbors Regression

K-nearest neighbors regression is a method for predicting a target value for a data point in the space spanned by the predictors. The prediction is the average of the target values of its k nearest neighbors. This method requires having a distance definition in this space.

Reference: Calculate a K-Nearest Neighbors Regression

```

KNN_REGRESS(options, neighbors, power,
            predictor_field1[, predictor_field2, ...] target_field)

```

where:

options

Reserved for future use.

neighbors

Integer

Is the number of nearest neighbors to participate in the prediction.

power

Integer

Is the power (p) of the L^p -distance.

- ☐ power=1 calculates the distance as the sum of the absolute values of the differences between the coordinates (Manhattan distance).
- ☐ power=2 calculates the distance as the square root of the sum of the squares of the differences between the coordinates (Euclidean distance).

predictor_field1[, predictor_field2, ...]

Numeric

Are one or more predictor field names.

target_field

Numeric

Is the target field.

Example: Predicting Income Using KNN_REGRESS

The following request uses KNN_REGRESS to predict income using the 10 nearest neighbors and Euclidean distance, with predictors age, education, population range, and gender. The DEFINE FILE command creates virtual fields with correct numeric formats for use in the function.

```

DEFINE FILE WF_RETAIL
POP_CODE/I2 =
  DECODE WF_RETAIL_GEOGRAPHY_CUSTOMER.CITY_POPULATION_RANGE (
    'H: 100,001 - 250,000'      1,
    'I: 250,001 - 1,000,000'    2,
    'J: 1,000,001 - 10,000,000' 3,
    'K: 10,000,001 - 50,000,000' 4,
    ELSE 0 );
GENDER_CODE/I2 =
  DECODE WF_RETAIL_CUSTOMER.GENDER (
    'M' 1, 'F' 0 );
END

```

```

TABLE FILE
WF_RETAIL
PRINT

ID_CUSTOMER
EDUC_LEVEL_M
POP_CODE
GENDER_CODE
INCOME_M
COMPUTE PRED_INCOME/D12.2 = KNN_REGRESS(' ',10,2,
AGE,
EDUC_LEVEL_M,
POP_CODE,
GENDER_CODE,
INCOME_M) ;
WHERE INCOME GT
12001.00
WHERE OUTPUTLIMIT EQ 12
WHERE POP_CODE NE 0
WHERE EDUC_LEVEL_M NE 0
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END

```

Partial output is shown in the following image.

<u>ID Customer</u>	Customer Education Level <u>w/Missings</u>	<u>POP CODE</u>	<u>GENDER CODE</u>	Income <u>w/Missings</u>	<u>PRED INCOME</u>
230444	6	3	0	117,209.96	109,000.21
11162	5	3	0	.	118,418.98
12673	4	4	1	.	109,226.55
13567	1	3	0	.	86,180.19
13902	6	3	1	140,601.15	129,186.74
237172	8	3	1	69,740.65	86,064.81
207543	6	3	0	129,006.18	107,767.61
118562	4	4	0	103,810.97	90,417.75
59823	8	3	1	91,152.13	91,785.86
121373	8	3	1	160,477.48	172,215.45
106534	8	3	1	188,232.28	171,015.29
60202	4	3	0	46,248.75	45,179.21

POLY_REGRESS: Polynomial Regression

Polynomial regression fits the target column to a polynomial expression of the predictor columns. The degree of the polynomial is specified as an input argument to the function.

Reference: Calculate a Polynomial Regression Column

```
POLY_REGRESS(options, degree, terms_generated,  
             predictor_field1, predictor_field2, [...], target_field)
```

where:

options

Reserved for future use.

degree

Integer

Is the degree of the polynomial. Low degree polynomials are recommended.

terms_generated

Keyword

Controls the terms that are generated in the polynomial equation.

- ☐ ALL_TERMS generates the most general polynomial of degree *degree* based on the predictor fields.
- ☐ INTERACTION_ONLY generates only terms linear in each predictor X_0, X_1, \dots , and cross-terms of the form $X_0 * X_1 * X_2$ of at most *degree* predictors are used.

```
predictor_field1, predictor_field2, [...],
```

Numeric

Are at least two predictor field names.

```
target_field
```

Numeric

Is the target field.

Example: Using POLY_REGRESS to Predict Income

The following scatter plot shows the predicted income values using polynomial regression of degree 4, with predictors education level, and age. The DEFINE FILE command creates virtual fields with correct numeric formats for use in the function.

```

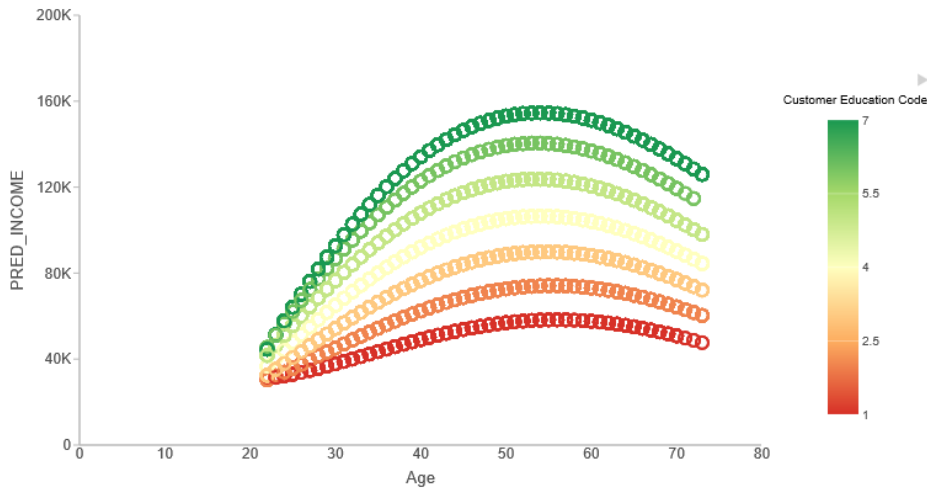
DEFINE FILE wf_retail
  EDUC_CODE/I2 TITLE 'Education Code'= DECODE
  WF_RETAIL_EDUCATION_CUSTOMER.EDUCATION_DEGREE_TYPE (
    'Professional Doctorates' 7,
    'Doctorates' 6,
    'Master's Degrees' 5,
    'Bachelor's Degree' 4,
    'Associate Degree' 3,
    'High School' 2,
    'No High School' 1,
    ELSE 0);
END

GRAPH FILE wf_retail
  PRINT
  AGE
  EDUC_CODE
  COMPUTE PRED_INCOME/D20.2 =POLY_REGRESS('', 4, ALL_TERMS,
                                          AGE,
                                          EDUC_CODE,
                                          INCOME);

  WHERE EDUC_CODE GT 0;
  WHERE INCOME GT 12001.00;
  WHERE AGE GT 21
  ON TABLE PCHOLD
  FORMAT JSCHART
  ON TABLE SET LOOKGRAPH SCATTER
  ON TABLE SET STYLE *
  INCLUDE=IBFS:/FILE/IBI_HTML_DIR/ibi_themes/Warm.sty,$
  TYPE=REPORT, CHART-ORIENTATION=HORIZONTAL, $
  TYPE=DATA, COLUMN=N3, BUCKET=Y-AXIS, /*PRED_INCOME*/
  $
  TYPE=DATA, COLUMN=N1, BUCKET=X-AXIS, /*AGE*/
  $
  TYPE=DATA, COLUMN=N2, BUCKET=COLOR, /*EDUC_CODE*/
  $
  ENDSTYLE
END

```


The output is shown in the following image. Predicted income versus age is plotted, with education level as a parameter visualized by color.



RF_CLASSIFY: Random Forest Classification

RF_CLASSIFY creates a random forest, which is an ensemble of decision trees. Each decision tree produces an independent classification prediction, and the prediction of the forest is the majority vote of the individual predictions.

Syntax: How to Calculate a Random Forest Classification

```
RF_CLASSIFY(options, number_of_trees,
            predictor_field1[, predictor_field2, ...] target_field)
```

where:

options

Reserved for future use.

number_of_trees

Integer

Is the number of decision trees in the forest.

predictor_field1[, *predictor_field2*, ...]

Numeric

Are one or more predictor field names.

target_field

Numeric

Is the target field.

Example: Predicting Education Level Using RF_CLASSIFY

The following procedure uses RF_CLASSIFY to predict education level, using a random forest with 100 decision trees, with predictors age, income, population range, and gender. The DEFINE FILE command creates virtual fields with correct numeric formats for use in the function.

```

DEFINE FILE WF_RETAIL
  POP_CODE/I2 =
    DECODE WF_RETAIL_GEOGRAPHY_CUSTOMER.CITY_POPULATION_RANGE (
      'H: 100,001 - 250,000'      1,
      'I: 250,001 - 1,000,000'    2,
      'J: 1,000,001 - 10,000,000' 3,
      'K: 10,000,001 - 50,000,000' 4,
      ELSE 0 );
  GENDER_CODE/I2 =
    DECODE WF_RETAIL_CUSTOMER.GENDER (
      'M' 1, 'F' 0 );
END

TABLE FILE WF_RETAIL
PRINT ID_CUSTOMER AGE INCOME DEGREE_M EDUC_LEVEL_M
COMPUTE ED_PRED/I2=RF_CLASSIFY(' ',100,
    AGE,
    INCOME,
    POP_CODE,
    GENDER_CODE,
    EDUC_LEVEL_M);

WHERE POP_CODE NE 0;
WHERE OUTPUTLIMIT IS 12;
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END

```

Partial output is shown in the following image.

<u>ID Customer</u>	<u>Age</u>	<u>Income</u>	<u>Customer Degree w/Missings</u>	<u>Customer Education Level w/Missings</u>	<u>ED</u> <u>PRED</u>
230444	41	117,209.96	Master's Degrees	6	6
11162	56	102,553.99	Bachelor's Degree	5	4
12673	66	97,259.44	Associate Degree	4	4
13567	66	86,283.38	Non Traditional Degree	1	4
13902	46	140,601.15	Master's Degrees	6	6
99444	62	187,736.31	.	.	8
237172	27	69,740.65	Professional Doctorates	8	8
207543	68	129,006.18	Master's Degrees	6	6
118562	40	103,810.97	Associate Degree	4	4
1447	69	106,104.18	.	.	6
59823	28	91,152.13	Professional Doctorates	8	8
121373	49	160,477.48	Professional Doctorates	8	8

RF_REGRESS: Random Forest Regression

RF_REGRESS creates a random forest, which is an ensemble of decision trees. Each decision tree produces an independent regression prediction, and the prediction of the forest is the average of the individual predictions.

Syntax: How to Calculate a Random Forest Regression

```
RF_REGRESS(options, number_of_trees,
           predictor_field1[, predictor_field2, ...] target_field)
```

where:

options

Reserved for future use.

number_of_trees

Integer

Is the number of decision trees in the forest.

predictor_field1[, *predictor_field2*, ...]

Numeric

Are one or more predictor field names.

target_field

Numeric

Is the target field.

Example: Predicting Income Using RF_REGRESS

The following procedure uses RF_REGRESS to predict income, using a random forest with 100 decision trees, with predictors age, education level, population range, and gender. The DEFINE FILE command creates virtual fields with correct numeric formats for use in the function.

```

DEFINE FILE WF_RETAIL
POP_CODE/I2 =
  DECODE WF_RETAIL_GEOGRAPHY_CUSTOMER.CITY_POPULATION_RANGE (
    'H: 100,001 - 250,000'      1,
    'I: 250,001 - 1,000,000'    2,
    'J: 1,000,001 - 10,000,000' 3,
    'K: 10,000,001 - 50,000,000' 4,
    ELSE 0 );
GENDER_CODE/I2 =
  DECODE WF_RETAIL_CUSTOMER.GENDER (
    'M' 1, 'F' 0 );
END

TABLE FILE WF_RETAIL
PRINT
ID_CUSTOMER
EDUC_LEVEL_M
POP_CODE
GENDER_CODE
INCOME_M
COMPUTE PRED_INCOME/D12.2 = RF_REGRESS(' ',100,
AGE,
EDUC_LEVEL_M,
POP_CODE,
GENDER_CODE,
INCOME_M);

WHERE EDUC_LEVEL_M NE 0
WHERE POP_CODE NE 0
WHERE INCOME GT 12001.00
WHERE OUTPUTLIMIT IS 12;
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END

```

Partial output is shown in the following image.

<u>ID Customer</u>	Customer Education Level <u>w/Missings</u>	<u>POP CODE</u>	<u>GENDER CODE</u>	Income <u>w/Missings</u>	<u>PRED INCOME</u>
230444	6	3	0	117,209.96	111,863.06
11162	5	3	0	.	112,104.75
12673	4	4	1	.	109,006.67
13567	1	3	0	.	88,624.32
13902	6	3	1	140,601.15	131,893.68
237172	8	3	1	69,740.65	87,870.05
207543	6	3	0	129,006.18	111,237.87
118562	4	4	0	103,810.97	91,134.26
59823	8	3	1	91,152.13	93,017.87
121373	8	3	1	160,477.48	171,263.54
106534	8	3	1	188,232.28	172,922.55
60202	4	3	0	46,248.75	43,194.98

Using IF-THEN-ELSE Logic in an Expression

IF-THEN-ELSE logic is supported as part of an expression, including arithmetic, character, and logical expressions.

Example: Using IF-THEN-ELSE Logic in an Arithmetic Expression

The following request uses IF-THEN-ELSE logic in an arithmetic expression to determine how much to add to LISTPR to calculate NEWPRICE.

```
TABLE FILE MOVIES
SUM COPIES
LISTPR
COMPUTE
NEWPRICE = LISTPR + (IF COPIES GT 10 THEN 0.00 ELSE 25.00);
BY CATEGORY
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image. Where there are more than 10 copies, the NEWPRICE equals LISTPR, otherwise NEWPRICE is \$25.00 greater than LISTPR.

<u>CATEGORY</u>	<u>COPIES</u>	<u>LISTPR</u>	<u>WHOLESALEPR</u>	<u>NEWPRICE</u>
ACTION	14	94.82	55.46	94.82
CHILDREN	12	201.67	105.87	201.67
CLASSIC	21	404.71	201.79	404.71
COMEDY	19	154.80	90.45	154.80
DRAMA	2	19.98	10.00	44.98
FOREIGN	6	269.76	146.24	294.76
MUSICALS	5	84.89	52.97	109.89
MYSTERY	21	235.81	116.97	235.81
SCI/FI	7	114.84	79.52	139.84
TRAIN/EX	10	119.87	60.98	144.87

Example: Using IF-THEN-ELSE Logic in a Character Expression

The following request uses IF-THEN-ELSE logic to determine what characters to concatenate to MOVIECODE in order to compute NEWCODE.

```
TABLE FILE MOVIES
PRINT COPIES
MOVIECODE
COMPUTE
NEWCODE/A20 = MOVIECODE | (IF MOVIECODE CONTAINS 'DIS' THEN 'NEY;' ELSE
';');
BY CATEGORY
WHERE CATEGORY EQ 'CHILDREN'
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image. If MOVIECODE contains the characters 'DIS', NEWCODE is generated by concatenating the characters 'NEY;', otherwise NEWCODE is generated by concatenating the character ';'.

<u>CATEGORY</u>	<u>COPIES</u>	<u>MOVIECODE</u>	<u>NEWCODE</u>
CHILDREN	1	031KKV	031KKV;
	2	043DIS	043DISNEY;
	1	093WOR	093WOR;
	2	306DIS	306DISNEY;
	1	309RAN	309RAN;
	1	387PLA	387PLA;
	1	476DIS	476DISNEY;
	3	732DIS	732DISNEY;

Example: Using IF-THEN-ELSE Logic in a WHERE Clause

The following request uses IF-THEN-ELSE logic in a WHERE clause to select records based on values of WHOLESALEPR where the values used for selection vary depending on the value of LISTPR in that record.

```
TABLE FILE MOVIES
PRINT COPIES
LISTPR
WHOLESALEPR
BY CATEGORY
WHERE WHOLESALEPR GT (IF LISTPR GT 20.00 THEN 15.00 ELSE 11.00)
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image. In the selected records, WHOLESalePR is greater than \$15.00 if LISTPR is greater than \$20.00. WHOLESalePR is greater than \$11.00 if LISTPR is less than or equal to \$20.00.

<u>CATEGORY</u>	<u>COPIES</u>	<u>LISTPR</u>	<u>WHOLESalePR</u>
ACTION	3	19.98	11.50
	4	19.99	11.99
CHILDREN	2	44.95	29.99
	1	29.95	15.99
CLASSIC	3	39.99	20.00
	1	29.95	15.99
	3	89.95	40.99
COMEDY	2	89.99	40.99
	4	19.95	12.55
	2	19.95	12.55
	3	59.99	30.00
	4	19.98	13.75
FOREIGN	4	19.98	13.75
	1	19.98	13.25
	1	59.95	30.00
	1	29.95	15.99
	1	39.98	25.00
	1	59.95	32.00
MUSICALS	1	59.95	30.00
	1	19.98	13.99
MYSTERY	1	19.98	13.99
	4	29.98	15.99
	2	25.99	15.99
SCI/FI	4	59.99	30.00
	1	19.98	14.99
	1	19.98	14.55
	1	19.95	13.99
	1	24.95	16.00
	3	29.98	19.99

Generating Descriptive Column Titles for Prefixed Fields

By default, the column titles for prefixed fields have the name of the prefix operator added above or below the field name. Using the SET PRFTITLE command, you can generate longer, more descriptive column titles. These descriptive column titles use longer descriptions of the prefix operators, which will also be translated into any standard language for which the server is configured.

Syntax: How to Generate Descriptive Column Titles for Prefixed Fields

```
SET PRFTITLE = {SHORT|LONG}
```

```
ON TABLE SET PRFTITLE {SHORT|LONG}
```

where:

SHORT

Places the prefix operator name above the field name to generate the column title.

LONG

Generates descriptive column titles for prefixed fields that can be translated to other languages.

Example: Generating Descriptive Column Titles for Prefixed Fields

The following request uses prefix operators with the default value for SET PRFTITLE.

```
TABLE FILE WF_RETAIL_LITE
SUM COGS_US CNT.COGS_US AVE.COGS_US CNT.DST.COGS_US MIN.COGS_US MAX.COGS_US
MDN.COGS_US
BY PRODUCT_CATEGORY
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

Product	Cost of Goods	COUNT	AVE	COUNT	MIN	MAX	MEDIAN
Category	Cost of Goods	COUNT	Cost of Goods	Cost of Goods	Cost of Goods	Cost of Goods	Cost of Goods
Accessories	\$342,877.00	1484	\$231.05	50	\$16.00	\$2,850.00	\$174.00
Camcorder	\$453,205.00	1417	\$319.83	46	\$60.00	\$8,610.00	\$180.00
Computers	\$109,281.00	600	\$182.13	8	\$81.00	\$668.00	\$167.00
Media Player	\$779,593.00	2249	\$346.64	77	\$36.00	\$1,640.00	\$310.00
Stereo Systems	\$857,042.00	3338	\$256.75	100	\$48.00	\$2,700.00	\$166.00
Televisions	\$227,820.00	267	\$853.26	22	\$275.00	\$9,750.00	\$550.00
Video Production	\$180,540.00	645	\$279.91	21	\$78.00	\$1,880.00	\$190.00

The following version of the request sets PRFTITLE to LONG.

```
SET PRFTITLE = LONG
TABLE FILE WF_RETAIL_LITE
SUM COGS_US CNT.COGS_US AVE.COGS_US CNT.DST.COGS_US MIN.COGS_US MAX.COGS_US
MDN.COGS_US
BY PRODUCT_CATEGORY
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image.

Product		Count	Average	Count of Distinct	Minimum	Maximum	Median
Category	Cost of Goods	Cost of Goods	Cost of Goods	Cost of Goods	Cost of Goods	Cost of Goods	Cost of Goods
Accessories	\$342,877.00	1484	\$231.05	50	\$16.00	\$2,850.00	\$174.00
Camcorder	\$453,205.00	1417	\$319.83	46	\$60.00	\$8,610.00	\$180.00
Computers	\$109,281.00	600	\$182.13	8	\$81.00	\$668.00	\$167.00
Media Player	\$779,593.00	2249	\$346.64	77	\$36.00	\$1,640.00	\$310.00
Stereo Systems	\$857,042.00	3338	\$256.75	100	\$48.00	\$2,700.00	\$166.00
Televisions	\$227,820.00	267	\$853.26	22	\$275.00	\$9,750.00	\$550.00
Video Production	\$180,540.00	645	\$279.91	21	\$78.00	\$1,880.00	\$190.00

Format Display Options for Abbreviating Numeric Values

Numeric values can be displayed in abbreviated form in terms of thousands, millions, billions, or trillions, using the format options k, m, b, t, and a.

- ☐ Format option **k** displays numeric values in terms of thousands. For example, 12345 displays as 12.35K.
- ☐ Format option **m** displays numeric values in terms of millions. For example, 1234567 displays as 1.23M.
- ☐ Format option **b** displays numeric values in terms of billions. For example, 1234567890 displays as 1.23B.
- ☐ Format option **t** displays numeric values in terms of trillions. For example, 1234567890000 displays as 1.23T.
- ☐ Format option **a** calculates the appropriate abbreviated option to use depending on the magnitude of the number, and uses that option for display. This option uses the appropriate abbreviation for the specific value on the current row and, therefore, each row may have a different abbreviation. For example, 1234567890 displays as 1.23B, while 1234567890000 displays as 1.23T.

Note:

- ❑ The abbreviated value displays with the number of decimal places specified in the format of the field to which it is returned.
- ❑ The format options do not change how a value is stored, just how it is displayed.
- ❑ Numbers are rounded prior to display in an abbreviated format. Therefore, when a packed or integer formatted number is displayed in abbreviated form using the EXTENDNUM ON setting, overflow values can be mistaken for correct results.
- ❑ These display options are supported with all numeric formats and are compatible with additional display options such as -, B, R, C, c, M, and N.

Example: Using Display Options for Abbreviating Numeric Values

The following request computes numeric fields that use the display options k, m, b, t, and a.

```
TABLE FILE WF_RETAIL_LITE
SUM COGS_US
COMPUTE
COGS_K/I10k = COGS_US;
COGS_M/D20.2m = COGS_US *2000;
COGS_B/D20.3b = COGS_US *300000;
COGS_T/D20.2t = COGS_US *400000000;
COGS_A/D20.1a = IF COGS_US LT 100000 THEN COGS_US
                ELSE IF COGS_US LT 200000 THEN COGS_US *2000
                ELSE IF COGS_US LT 500000 THEN COGS_US *300000
                ELSE COGS_US *400000000;

BY BRAND
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image. COGS_K has format I, which displays with no decimal places. COGS_M, COGS_B, and COGS_T have format D20.2, which displays with two decimal places. COGS_A has format D20.1, which displays with one decimal place.

<u>Brand</u>	<u>Cost of Goods</u>	<u>COGS_K</u>	<u>COGS_M</u>	<u>COGS_B</u>	<u>COGS_T</u>	<u>COGS_A</u>
Audio Technica	\$38,000.00	38K	76.00M	11.400B	15.20T	38.0K
BOSE	\$152,126.00	152K	304.25M	45.638B	60.85T	304.3M
Canon	\$110,219.00	110K	220.44M	33.066B	44.09T	220.4M
Denon	\$25,970.00	26K	51.94M	7.791B	10.39T	26.0K
Grado	\$21,930.00	22K	43.86M	6.579B	8.77T	21.9K
Harman Kardon	\$78,350.00	78K	156.70M	23.505B	31.34T	78.3K
JVC	\$180,049.00	180K	360.10M	54.015B	72.02T	360.1M
LG	\$161,800.00	162K	323.60M	48.540B	64.72T	323.6M
Logitech	\$61,432.00	61K	122.86M	18.430B	24.57T	61.4K
Niles Audio	\$73,547.00	74K	147.09M	22.064B	29.42T	73.5K
Onkyo	\$125,438.00	125K	250.88M	37.631B	50.18T	250.9M
Panasonic	\$292,871.00	293K	585.74M	87.861B	117.15T	87.9B
Philips	\$43,354.00	43K	86.71M	13.006B	17.34T	43.4K
Pioneer	\$233,330.00	233K	466.66M	69.999B	93.33T	70.0B
Polk Audio	\$64,575.00	65K	129.15M	19.372B	25.83T	64.6K
Roku	\$10,248.00	10K	20.50M	3.074B	4.10T	10.2K
Samsung	\$218,754.00	219K	437.51M	65.626B	87.50T	65.6B
Sanyo	\$43,878.00	44K	87.76M	13.163B	17.55T	43.9K
Sennheiser	\$78,113.00	78K	156.23M	23.434B	31.25T	78.1K
Sharp	\$89,570.00	90K	179.14M	26.871B	35.83T	89.6K
Sony	\$657,576.00	658K	1,315.15M	197.273B	263.03T	263.0T
Thomson Grass Valley	\$89,344.00	89K	178.69M	26.803B	35.74T	89.3K
Toshiba	\$5,214.00	5K	10.43M	1.564B	2.09T	5.2K
Yamaha	\$94,670.00	95K	189.34M	28.401B	37.87T	94.7K

Support for C-Style In-Line Comments

C-style comments can now be used in WebFOCUS procedures.

This type of comment is enclosed within an opening /* tag (slash followed by asterisk) and a closing */ tag (asterisk followed by slash). A C-style comment can appear anywhere and span multiple lines.

Example: Placing C-Style Comments in a Procedure

The following example places C-style comments in a procedure.

```
TABLE FILE WF_RETAIL /* this is a multi-line comment
that will not interfere with processing and will be ignored
until the comment is closed with */
SUM /* Another comment */ COGS_US
.
.
.
```

SET PCTFORMAT: Displaying a Percent Sign for Percent Prefix Operators

The SET PCTFORMAT command controls whether fields prefixed with the operators PCT., RPCT., and PCT.CNT. display with a percent sign or with the format associated with the original field.

The syntax is:

```
SET PCTFORMAT = {OLD|PERCENT}
```

where:

OLD

Displays columns prefixed with PCT., RPCT., and PCT.CNT. with the format associated with the original field.

PERCENT

Displays columns prefixed with PCT., RPCT., and PCT.CNT. with a percent sign. It also allows the prefixed fields to be reformatted. This is the default value.

PCT.CNT.*field* will always display with two decimal places, unless reformatted. For PCT.*field* and RPCT.*field*, with SET PCTFORMAT = PERCENT, if the original field has a:

- ☐ Precision-based format (F, D, M, X), the column will display with length 7 and two decimal places.
- ☐ Packed format, the column will display with its original number of decimal places.
- ☐ Integer format, the column will display with no decimal places.

Example: **Displaying Columns for Fields Prefixed With PCT., RPCT., and PCT.CNT.**

The following request displays columns that use the prefix operators PCT. and PCT.CNT against fields formatted to show currency symbols. The value of the PCTFORMAT parameter is set to OLD.

```
DEFINE FILE GGSales
DOLL1/D12.4M = DOLLARS;
UNIT1/D12.6M = UNITS;
DOLL2/P12.4M = DOLLARS;
UNIT2/I7M = UNITS;
END
TABLE FILE GGSales
SUM DOLLARS UNITS PCT.UNIT2 PCT.DOLL1 PCT.DOLL2 PCT.CNT.UNIT1
BY CATEGORY
ON TABLE SET PCTFORMAT OLD
ON TABLE SET PAGE NOLEAD
ON TABLE SET STYLE *
GRID=OFF,$
ENDSTYLE
END
```

The output is shown in the following image. The columns that display prefixed fields, except for the one using the PCT.CNT. prefix, display currency symbols instead of percent signs because the formats of the original fields are used.

<u>Category</u>	<u>Dollar Sales</u>	<u>Unit Sales</u>	PCT <u>UNIT2</u>	PCT <u>DOLL1</u>	PCT <u>DOLL2</u>	PCT.CNT <u>UNIT1</u>
Coffee	17231455	1376266	\$37	\$37.3328	\$37.3328	33.36
Food	17229333	1384845	\$37	\$37.3282	\$37.3282	33.36
Gifts	11695502	927880	\$25	\$25.3389	\$25.3389	33.29

Changing SET PCTFORMAT to PERCENT produces the following output, in which the currency symbols have been removed and all prefixed columns display percent signs.

<u>Category</u>	<u>Dollar Sales</u>	<u>Unit Sales</u>	PCT <u>UNIT2</u>	PCT <u>DOLL1</u>	PCT <u>DOLL2</u>	PCT.CNT <u>UNIT1</u>
Coffee	17231455	1376266	37%	37.33%	37.3328%	33.36%
Food	17229333	1384845	37%	37.33%	37.3282%	33.36%
Gifts	11695502	927880	25%	25.34%	25.3389%	33.29%



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

/ Server New Features

WebFOCUS Reporting Server Release 8205

DataMigrator Server Release 7709

DN4501691.0219

Information Builders, Inc.
Two Penn Plaza
New York, NY 10121-2898

