



Data Federation Administration Tool Guide

- SAP Business Objects Business Intelligence platform 4.1 Support Package 2

2013-11-21

Copyright

© 2013 SAP AG or an SAP affiliate company. All rights reserved. No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice. Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors. National product specifications may vary. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty. SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries. Please see <http://www.sap.com/corporate-en/legal/copyright/index.epx#trademark> for additional trademark information and notices.

2013-11-21

Contents

Chapter 1	What's new in the Data Federation Administration Tool Guide.....	7
Chapter 2	Introduction to administration and tuning of the data federation service.....	9
2.1	Introduction to administration and tuning of the data federation service.....	9
Chapter 3	Using the data federation administration tool.....	11
3.1	The purpose of the data federation administration tool	11
3.2	Starting the data federation administration tool.....	12
3.3	Adding users with administration rights for the data federation administration tool.....	12
3.4	Configuring the data federation administration tool for Microsoft Active Directory authentication.....	12
3.5	Logging out of a data federation administration tool session.....	13
3.6	Examining running queries on the data federation query engine.....	13
3.7	Testing SQL queries on the data federation query server.....	13
3.8	Viewing how the data federation query engine plans queries.....	13
3.9	Browsing the history of queries executed on the data federation query server.....	14
3.10	Querying metadata.....	14
3.11	Cancelling a query.....	14
3.11.1	Cancelling a query.....	15
3.12	The Query Panel tab in data federation administration tool.....	15
3.13	The Query Monitoring tab in data federation administration tool.....	18
3.14	The System Parameters tab in data federation administration tool.....	19
3.15	The Connector Configuration tab in data federation administration tool.....	21
3.16	The Statistics Tab in Data Federation Administration Tool.....	23
3.17	Showing the Properties view in the data federation administration tool.....	25
3.18	Connecting from the data federation administration tool to a server configured for SSL.....	25
Chapter 4	Optimizing queries.....	27
4.1	Tuning the performance of data federation queries.....	27
4.2	Using system parameters to optimize the use of memory.....	27
4.2.1	Operators that consume memory.....	29
4.3	Using statistics to let the application choose the best algorithms for querying sources.....	29

4.3.1	About column cardinality.....	30
4.3.2	About the fanout value of relationships between columns.....	30
4.3.3	Filtering the recorded statistics to compute only those needed to optimize reports	31
4.4	Optimizing query plans.....	31
4.4.1	The Query Plan view in the data federation administration tool.....	32
4.4.2	The Explain Statistics Command.....	33
4.4.3	Using the explain query feature to get feedback to tune a query.....	33
4.4.4	Checking if an operator was pushed using the data federation administration tool.....	34
4.4.5	Guidelines for using system parameters to optimize queries on small tables joined to large tables.....	35
4.4.6	Guidelines for using system parameters to optimize queries on large tables with data that can be sorted.....	37
4.4.7	Using system parameters to control activation of order-based operators.....	38
4.4.8	Forcing parallel execution of data source sub-queries.....	39
4.4.9	Semi-join execution strategies.....	39
4.5	Optimizing specific connectors.....	40
4.5.1	Increasing concurrency of callbacks for parallel queries to SAP NetWeaver BW.....	40
4.5.2	Changing the size of response packages from queries to SAP NetWeaver BW.....	40
4.6	Promoting optimizations settings made for the data federation service.....	41
Chapter 5	Configuring connectors to sources of data.....	43
5.1	Viewing the information for a connector in the data federation administration tool.....	43
5.2	Changing the properties of a connector in the data federation administration tool.....	43
5.3	Configuring connectors for relational data sources.....	43
5.3.1	List of common connector properties for relational data sources.....	43
5.3.2	List of specific connector properties for MySQL data sources.....	47
5.3.3	List of specific connector properties for Teradata data sources.....	48
5.3.4	List of specific connector properties for Sybase ASE data sources.....	48
5.3.5	List of specific connector properties for SQL Server data sources.....	49
5.3.6	List of specific connector properties for Generic ODBC or JDBC data sources	50
5.3.7	List of specific connector properties for Oracle data sources	52
5.3.8	List of specific connector properties for SAP HANA data sources.....	53
5.3.9	List of specific connector properties for MaxDB data sources.....	53
5.4	Configuring connectors for SAS.....	53
5.4.1	List of connector properties for SAS data sources.....	54
5.4.2	Optimizing SAS queries by ordering tables in the from clause by their cardinality	62
5.5	Configuring connectors for SAP NetWeaver BW.....	63
5.5.1	List of connector properties for SAP NetWeaver BW data sources.....	63
5.5.2	Manually setting the callback ID that SAP NetWeaver BW uses to contact the data federation service.....	68
5.5.3	Cleaning the IDs of callbacks for SAP NetWeaver BW connections.....	70

5.5.4	Leveraging SAP analysis authorizations to filter data automatically.....	70
5.5.5	Architecture of the SAP NetWeaver BW connection in multi-source universes.....	72
5.5.6	Callback sequence of the SAP NetWeaver BW connection in multi-source universes.....	73
5.6	Setting the capabilities of relational and SAS connectors using the data federation administration tool.....	74
5.7	Complete list of connector capabilities for relational data sources.....	75
Chapter 6	Managing system and session parameters.....	79
6.1	About system and session parameters.....	79
6.2	Changing a system parameter using the data federation administration tool.....	79
6.3	Changing a session parameter using the data federation administration tool.....	79
6.4	Setting the capabilities of relational and SAS connectors using the data federation administration tool.....	80
6.5	List of system parameters.....	80
6.6	List of session parameters.....	97
6.7	Collation in the data federation application.....	98
6.7.1	Supported Collations in the data federation application.....	99
6.7.2	How the data federation application decides how to push queries to sources when using binary collation.....	100
6.7.3	Setting string sorting and string comparison behavior for data federation SQL queries.....	100
Chapter 7	SQL syntax reference.....	103
7.1	The query language for the data federation query engine	103
7.1.1	Identifiers and naming conventions.....	103
7.1.2	Data types used in the data federation query engine.....	105
7.1.3	Statements.....	108
7.1.4	Expressions.....	110
7.1.5	Comments.....	114
7.2	Grammar for the SELECT clause.....	114
Chapter 8	Glossary.....	119
8.1	Terms and descriptions.....	119
Chapter 9	Troubleshooting.....	121
9.1	About logging of the data federation service.....	121
9.2	For SAP NetWeaver BW datasources, long-running queries cause the connection to close..	121
9.3	For SAP NetWeaver BW connector, error NoClassDefFoundError: CpicDriver.....	122
Index		123

What's new in the Data Federation Administration Tool Guide

Links to information about the new features and documentation changes for the data federation administration tool for each version of SAP BusinessObjects BI platform.

SAP BusinessObjects BI platform 4.1 Support Package 2 - November 2013

What's new	Link to more information
System parameter FORCE_ASYNC_SUBMIT_ON_BW_SOURCES that forces queries to be submitted asynchronously for SAP NetWeaver BW data sources. This allows queries on SAP NetWeaver BW to be cancelled.	List of system parameters

Introduction to administration and tuning of the data federation service

2.1 Introduction to administration and tuning of the data federation service

To administer or tune the data federation service, you use the data federation administration tool.

Administration

You use the data federation administration tool when you need to administer aspects of the data federation service that are specific to the way data is treated by the service. These aspects include managing properties of connectors to specific data sources, configuring memory, or setting parameters that affect queries on the data federation query engine.

With the data federation administration tool, you can browse and manage connectors, browse data sources and run queries against them, manage “statistics”, and view the lists of past queries and running queries. You may want to view lists of past or running queries because in your production system, reporting applications will generate the queries and send them to query server without human intervention. Viewing the queries that have been generated lets you verify that your system is doing what you expect.

For general administration, such as management of user accounts or logging, use the tools of the platform where the data federation service is installed.

Tuning

You use the data federation administration tool for tuning when you want to adapt your connectors or your queries to the data in your data sources.

Tuning involves setting capabilities of each connector to make it pass as much work as possible to each data source, setting appropriate “statistics” for each data source, and configuring parameters to optimize each query that is sent to the server. Optimization typically means making your data sources do as much processing as possible, and sending as little data over the network as possible. The data federation service has multiple options for “pushing” work to sources and reducing data transfer, as well as tools that help you understand how the system is processing your queries.

Using the data federation administration tool

3.1 The purpose of the data federation administration tool

The data federation administration tool is a rich client application that offers easy-to-use features to manage your data federation service.

Tightly integrated in the SAP BusinessObjects Enterprise platform, the data federation service enables multi-source universes by distributing queries across disparate data sources, and lets you federate data through a single data foundation.

The data federation administration tool lets you optimize data federation queries and fine-tune the data federation query engine for the best possible performance.

You use the data federation administration tool to do the following:

- Test your SQL queries.
- Visualize optimization plans which detail how federated queries are distributed to each source.
- Compute “statistics” and set system parameters to fine-tune the data federation services and get the best possible performance.
- Manage properties to control how queries are executed in each data source at the connector level.
- Monitor running SQL queries
- Browse the history of executed queries.

Related Topics

- [Examining running queries on the data federation query engine](#)
- [Testing SQL queries on the data federation query server](#)
- [Viewing how the data federation query engine plans queries](#)
- [Browsing the history of queries executed on the data federation query server](#)
- [About system and session parameters](#)
- [Viewing the information for a connector in the data federation administration tool](#)
- [Using statistics to let the application choose the best algorithms for querying sources](#)

3.2 Starting the data federation administration tool

1. Click **Start > Programs > BusinessObjects Data Federator XI Release 4 > data federation administration tool**.
2. Enter the name of your system, your user name and password, then click **OK**.

3.3 Adding users with administration rights for the data federation administration tool

In the SAP BusinessObjects Enterprise server, the user group called Data Federation Administrators has rights to administer the data federation service.

See the *SAP BusinessObjects Enterprise Administrator's Guide* for details on adding users to a group.

3.4 Configuring the data federation administration tool for Microsoft Active Directory authentication

To configure the data federation administration tool for Active Directory authentication, you must edit the initialization file for the data federation administration tool. In this file, you must point to two configuration files: a login configuration file and a kerberos configuration file.

1. Edit the file: `install_dir\SAP BusinessObjects\SAP BusinessObjects Enterprise XI 4.0\win32_x86\DFAdministrationTool.ini`.

Add the following lines to the end of the file:

```
-Djava.security.auth.login.config=path-to-bsclogin\bscLogin.conf  
-Djava.security.krb5.conf=path-to-kerberos\krb5.ini
```

For example:

```
-Djava.security.auth.login.config=C:\WINNT\bscLogin.conf  
-Djava.security.krb5.conf=C:\WINNT\krb5.ini
```

2. Make sure the two files `bscLogin.conf` and `krb5.ini` are configured for Active Directory authentication with Kerberos.

For details, see the section *Using Kerberos authentication for Windows AD* in the *SAP BusinessObjects Enterprise Administrator's Guide*.

3.5 Logging out of a data federation administration tool session

- Click the Logout button at the top left side of the toolbar.

3.6 Examining running queries on the data federation query engine

1. Start the data federation administration tool.
2. Click the **Query Monitoring** tab.
3. Click Refresh.

The "Running queries" pane shows the queries that are running.

3.7 Testing SQL queries on the data federation query server

1. Start the data federation administration tool.
2. Click the **Query Panel** tab.
3. Enter your query.
4. Click **Run** to execute the query.

The query is run and the results displayed in the **Query Results** panel.

3.8 Viewing how the data federation query engine plans queries

The data federation query engine analyzes your SQL queries and decides how to translate them so that you get the correct data from multiple sources as fast as possible. In order to perform this analysis, the query engine distributes as much work as possible among the data sources and writes sub-queries to fetch as little data over the network as is needed to produce the final result.

You can see how the query has been distributed among the source by using the explain tool.

1. Start the data federation administration tool.
2. Click the **Query Panel** tab.
3. Type the query you want to view.
4. Click the arrow beside **Run**, then click **Explain query**.

Your query appears as a plan that the query engine generated.

Related Topics

- [The Query Plan view in the data federation administration tool](#)

3.9 Browsing the history of queries executed on the data federation query server

If you or your applications have already sent queries to the data federation query server, you can see the list of those queries using the data federation administration tool.

1. Start the data federation administration tool.
2. Click the **Query Monitoring** tab.

The "Executed queries" pane shows the queries that have been executed.

3.10 Querying metadata

Dynamic applications that are not hard-coded to work with a specific set of tables must have a mechanism for determining the structure and attributes of the objects in any database to which they connect. These applications may require information such as the following.

- the number and names of the tables in the targets and datasources
- the number of columns in a table together with the name, data type, scale, and precision of each column
- the keys that are defined for a table

Applications based on the data federation query engine can access the information in the system catalogs by using the following stored procedures:

```
CALL getTables 'name-of-catalog', '%', '%'  
CALL getColumns 'name-of-catalog', 'name-of-schema', 'name-of-table', '%'  
CALL getKeys 'name-of-catalog', 'name-of-schema', 'name-of-table'
```

3.11 Cancelling a query

When using data federation, a command lets you cancel all running queries, or a specific running query.

The cancel command is asynchronous. Therefore, in some cases, when you cancel a query, your client application may see the query as cancelled while the data federation query engine may have not yet completed the cancel.

Note:

The system parameter `FORCE_ASYNC_SUBMIT_ON_BW_SOURCES` needs to be set to true in order to cancel queries on SAP NetWeaver BW data sources.

Related Topics

- [Changing a system parameter using the data federation administration tool](#)

3.11.1 Cancelling a query

1. Click the **Query Monitoring** tab.
2. Right-click the query that you want to cancel.
3. Click **Cancel**.

3.12 The Query Panel tab in data federation administration tool

Panels

Panel	Description
SQL Text	<p>Where you can type your SQL query.</p> <p>You can insert elements into your query by double-clicking or dragging and dropping from the Catalogs, Operators and Functions panels.</p> <p>Controls</p> <ul style="list-style-type: none">• Max Rows: the maximum number of rows to retrieve• Show total number of rows: specifies whether or not to show the total number of rows in the result, even if you do not retrieve all of them
Catalogs	shows all existing catalogs on the data federation service
Operators	shows list of possible operators
Functions	shows list of available functions grouped by categories

Panel	Description
Query Results	container for query results; displayed when you run a query by clicking Run
Raw Data	shows raw data results of last run query; displayed when you click Run or Execute
Auto Charts	simple chart (Pie) presentation of query results; displayed when you click Run or Execute
Query Plan	<p>shows query plan without executing query; displayed when you click Explain Query</p> <p>contains two internal panels</p> <ul style="list-style-type: none"> • Plan: shows the plan structure as a tree view • Details: shows details about selected node in Plan panel <p>You can find more details about selected nodes in the Properties view.</p>
Query Statistics	shows impacted elements of the current query with their “statistics”; displayed when you click Explain Statistics

Buttons

Button Label	Description
Run	<p>a pull down button with menu items</p> <ul style="list-style-type: none"> • Default action: executes the query that is currently in the SQL Text panel • Execute Query action: same as default action • Explain Query action: explains query plan • Explain Statistics action: shows impacted elements with their “statistics” and lets you update cardinalities <p>Results are displayed in Query Results panel.</p>
Undo last change	reverts the last change in SQL text panel
Redo last change	repeats last change in the SQL Text panel
Refresh Catalogs	refreshes the Catalogs panel

Button Label	Description
Show/Hide Catalogs	shows or hides the Catalogs panel
Show/Hide Operators	shows or hides the Operators panel
Show/Hide Functions	shows or hides the Functions panel
Display only source queries	displays only source queries nodes by filtering intermediate nodes

Contextual Menu

Menu Item	Description
Compute	<p>a pull down menu with sub menu items</p> <ul style="list-style-type: none"> • Selection Only computes only selected nodes • Selection and children computes selected nodes and their children within this query context • Not computed Only (Children included) computes only selection with their children when Current Cardinality column is unknown
Current Cardinality	<p>a pull down menu with sub menu items</p> <ul style="list-style-type: none"> • Use User Cardinality: forces the data federation service to use cardinality set by the user for query optimization on selected objects; This action is enabled when you select only tables or columns. After the action completes, the current cardinality is equal to User Cardinality. • Use Source Cardinality: forces the data federation service to use cardinality retrieved from the data source for query optimization on selected objects; This action is enabled when you select only tables or columns. After the action completes, the current cardinality is equal to Cardinality from Datasource.

Related Topics

- [The Query Plan view in the data federation administration tool](#)
- [The Explain Statistics Command](#)

3.13 The Query Monitoring tab in data federation administration tool

Table

Column Name	Description
Query	The ID of the query or sub-query Different icons <ul style="list-style-type: none">• Running icon: the query is running• Closed success icon: the query is closed and succeed• Closed failed icon: the query is closed and failed; You can use Properties view to see the exception.
Start Time	start time of execution
End Time	end time of execution
Execution Time	the elapsed time between start and end time of execution
Rows	number of rows extracted by query
Status	the Query statuses <ul style="list-style-type: none">• Analyzing: The query is getting analyzed by federation engine.• Executing: The query is getting executed by federation engine.• Closed: The query is closed either if an exception occurs or not.
Server Name	the server name which handles the query
User Name	the user name who launched the query
SQL Text	SQL text of the query

Filters

Filter Label	Description
Filter	filter on text of available columns
Status	filter on queries statuses <ul style="list-style-type: none"> • All Queries • Running Queries • Executed Queries
Type	filter on queries types <ul style="list-style-type: none"> • All Queries • SQL • Commands • Procedures
Connection	filter on connections <ul style="list-style-type: none"> • All Connection • Current Connection: shows only queries of current data federation administration tool connection

Buttons

Button Tooltip	Description
Save monitoring information as XML	saves monitoring information as XML
Refresh	gets fresh monitoring information from server

3.14 The System Parameters tab in data federation administration tool**Tabs**

You can use the System Parameters tab to manage system and session parameters and properties.

Tab Label	Description
System Parameters	<p>lets you manage the system parameters</p> <p>Columns</p> <ul style="list-style-type: none"> • Parameter: the name of the parameter. • Current value: the value that the parameter currently has; You can enter a new value here. Those parameters that are read-only have a gray background. • Default value: the value that the parameter had at system startup; You can use this value as a reference if you have changed the current value and you want to revert it. • Category: the category of the parameter. • Description: the description of the parameter; You can also see the full list of parameters with their descriptions in the Data Federation Administration Tool Guide.
Session Parameters	<p>lets you manage the session parameters</p> <p>Columns</p> <ul style="list-style-type: none"> • Parameter: the name of the parameter. • Current value: the value that the parameter currently has; You can enter a new value here. • Description: the description of the parameter; You can also see the full list of parameters with their descriptions in the Data Federation Administration Tool Guide.
System Properties	<p>shows the system properties</p> <p>Columns</p> <ul style="list-style-type: none"> • Parameter: the name of the parameter. • Current value: the value that the parameter currently has.
Startup Parameters	<p>shows the startup parameters</p> <p>Columns</p> <ul style="list-style-type: none"> • Parameter: the name of the parameter. • Current value: the value that the parameter currently has.

Tab Label	Description
Install Parameters	<p>shows the install parameters</p> <p>Columns</p> <ul style="list-style-type: none"> • Component: the name of component of the parameter. • Parameter: the name of the parameter. • Current value: the value that the parameter currently has. • Default value: the value that the parameter had at system startup. • Origin: the origin of the value of the parameter. One of: ORIGIN_DEFAULT, ORIGIN_SERVER_PROPERTIES, ORIGIN_SYSTEM_PROPERTIES.

Display Contextual Menu

Menu Item	Description
System and Session parameters	shows only System and Session parameters
All Parameters	Show all tabs

Related Topics

- [List of system parameters](#)

3.15 The Connector Configuration tab in data federation administration tool

Panels

Panel	Description
Connectors tree	<p>shows the list of connectors</p> <p>To see the configuration of any connector, double-click it in the Connectors tree .</p>
General Information tab	<p>shows general information about the current connector</p> <p>To see general information about any connector, double-click it in the Connectors tree.</p>

Panel	Description
Capabilities tab	shows the capabilities of the current connector To see the capabilities of any connector, double-click it and select the Capabilities tab.
Configuration Properties tab	shows the configuration properties of the current connector To see the configuration properties of any connector, expand it and double-click the Configuration node.

Buttons

Button Label	Description
Collapse All	collapses the connectors list
Expand All	expands the connectors list
Show/Hide search bar	shows or hides the search bar you can use this to search for connectors by their name
Refresh	refreshes the connectors list
Save	saves configuration properties

Contextual Menu

Menu Item	Description
Create configuration	creates a new configuration for a connector Lets you set new values for configuration properties.
Edit configuration	lets you edit the configuration of a connector Alternatively, just double-click the connector.
Delete configuration	deletes the configuration of a connector When a configuration is deleted, default values are used.

3.16 The Statistics Tab in Data Federation Administration Tool

Table

Column Name	Description
Catalogs	<p>Can contain:</p> <ul style="list-style-type: none"> • a catalog name • a schema name • a table name • a column name • an error description • a wait message <p>The sibling columns are filled only if the object is a table or a column.</p>
Last Compute Date	Last time a compute action was made on the object or Not computed if none.
Number of Requests	Number of queries run on the data federation service against the object or No cached record if none.
Current Cardinality	Cardinality currently used by the data federation service to optimize its query plans or Unknown if none.
Cardinality From Source	Cardinality returned by the data source after a compute action is run against the object or Unknown if none.
User Cardinality	<p>Cardinality forced by the user that the data federation service will use to optimize its query plans or Unset if none.</p> <p>This column is editable. To edit the value just click on the cell, enter an integer and then press return or click somewhere else. To discard editing, press escape.</p>
- All columns	When an action is currently executing on an item (Compute, Refresh...) the item is displayed in italics.

Buttons

Button Label	Description
Refresh	<p>Updates all the data currently displayed from the data federation service.</p> <p>This action can be time consuming when numerous objects are displayed. For long refresh operations, the progression of the action that is run in background can be monitored in the Progress view.</p>
Compute	<p>Asks the data federation service to retrieve cardinalities of selected objects from the data sources they belong to. After the action completes, User Cardinality and Last Compute Date are updated, and Current Cardinality is set to User Cardinality.</p> <p>This action is enabled when selection only contains tables or columns.</p>

Contextual Menu

Menu Item	Description
Compute	<p>Asks the data federation service to retrieve cardinalities of selected objects from the data sources they belong to. After the action completes, User Cardinality and Last Compute Date are updated, and Current Cardinality is set to User Cardinality.</p> <p>This action is enabled when selection only contains tables or columns.</p>
Use User Cardinality	<p>Forces the data federation service to use cardinality set by the user for query optimization on selected objects. This action is enabled when selection only contains tables or columns. After the action completes, the current cardinality is equal to User Cardinality.</p>
Use Source Cardinality	<p>Forces the data federation service to use cardinality retrieved from the datasource for query optimization on selected objects. This action is enabled when selection only contains tables or columns. After the action completes, the current cardinality is equal to Cardinality from Source.</p>

Filters

Column	Description
Catalogs	<p>Make a filter on all checked catalogs. Click OK to validate the selection or click elsewhere to discard the selection.</p>

Column	Description
- All other filters	<p>These filters let you filter the displayed tables and columns.</p> <p>If a table does not satisfy a filter condition it can nevertheless be displayed if one of its columns satisfies all the filter conditions.</p> <p>These filters do not let you hide catalogs or schemas. Catalogs and schemas are displayed even if none of their objects satisfies the filter conditions. To hide whole catalogs, use the Catalogs filter.</p>

Related Topics

- [Filtering the recorded statistics to compute only those needed to optimize reports](#)

3.17 Showing the "Properties" view in the data federation administration tool

The "Properties" view in the data federation administration tool shows you supplementary information about various elements of the interface.

- Click **Window > Other > Admin > Properties**.

3.18 Connecting from the data federation administration tool to a server configured for SSL

Often it is necessary to connect the data federation administration tool to a server configured for SSL to increase security.

1. Edit the file `DFAdministrationTool.ini` in the directory `boe-install-dir/win32_x86`.
2. Add following JVM arguments:

```
-Dbusinessobjects.ora.oci.protocol=ssl
-DcertDir=C:\SSLCert
-DtrustedCert=cacert.der
-DsslCert=servercert.der
-DsslKey=server.key
-Dpassphrase=passphrase.txt
```


Optimizing queries

4.1 Tuning the performance of data federation queries

You can tune the performance of your data federation queries. The strategy to use to tune performance is as follows.

1. Use system parameters to optimize the use of memory.
2. Use “statistics” to let the application choose the best algorithms for querying sources.
3. If the application did not automatically activate the semi-join operator, verify if you can change the parameters to activate the semi-join.
4. If the semi-join is not appropriate, verify if you can change parameters to activate the “merge join”.
5. If your data supports capabilities that are disabled by default, activate the capabilities in your connector.

For example, while DB2 does not support predictable ordering of null values, if you know that your data has no nulls, you can still use a “merge join”. In this case, set the capabilities of the source to force it to perform an order by.

Note:

When you want to promote a data foundation to another system, and you have changed system parameters to optimize the queries against the data foundation, you must also promote the system parameters. You can use the Lifecycle management console for SAP BusinessObjects Enterprise to do this.

Related Topics

- [Using system parameters to optimize the use of memory](#)
- [Guidelines for using system parameters to optimize queries on small tables joined to large tables](#)
- [Guidelines for using system parameters to optimize queries on large tables with data that can be sorted](#)
- [Promoting optimizations settings made for the data federation service](#)

4.2 Using system parameters to optimize the use of memory

You can use the following strategies to optimize how the application uses memory.

- Set the amount of memory used by the Java virtual machine (JVM) that is running the application. For details, see the documentation about changing server properties in the *SAP BusinessObjects Enterprise Administrator's Guide*.

Adjust the default value depending on the speed of the application and the amount of memory you have available.

- Set the server parameter `EXECUTOR_TOTAL_MEMORY`.

This parameter lets you configure the amount of memory used for query execution.

Set this parameter either as a percentage of the memory used by the JVM, or as a fixed value with a suffix indicating the units (for example, 512M, 512m, 1024K or 1024k). If you enter a fixed value, it must be lower than the value given to the JVM.

- Set the server parameter `EXECUTOR_STATIC_MEMORY`.

This parameter lets you set the minimal amount of memory allocated to operators upon initialization. You can set a percentage of the memory used by the executor, or as a fixed value. If you enter a fixed value, it must be lower than the value given to query execution.

- Set the server parameter `MAX_CONCURRENT_MEMORY_CONSUMING_QUERIES`.

Defines the number of queries that consume memory that can run concurrently. Other queries are not affected.

Enter a small value here, if you have many large queries.

Enter a large number if you have many small queries.

- `MAX_CONCURRENT_MEMORY_CONSUMING_OPERATORS`

This parameter limits how many operators that consume memory run in parallel.

Decrease this number if the operators in your queries are consuming too much memory.

You can approximate the average size and number of operators in your queries by counting the number of large tables in different datasources accessed. For example, four large tables in different datasources in one mapping rule result in three joins that consume memory.

For example, set the JVM memory to 1000M to allocate 1000 megabytes of memory for the JVM.

Then, set `EXECUTOR_TOTAL_MEMORY` to 80% to allocate 800 megabytes of memory for the query execution.

Then, set `EXECUTOR_STATIC_MEMORY` to 25% to allocate 200 megabytes of memory for each operator.

Then, set `MAX_CONCURRENT_MEMORY_CONSUMING_QUERIES` to 2 to limit concurrent operators to two.

With the example settings above, two queries will be able to run concurrently, each will have 100 megabytes of minimal memory, and each will be able to access a dynamic pool of 600 megabytes of memory.

To audit your system's memory use, use the statement `info buffermanager`.

Note:

When you want to promote a data foundation to another system, and you have changed system parameters to optimize the queries against the data foundation, you must also promote the system parameters. You can use the Lifecycle management console for SAP BusinessObjects Enterprise to do this.

Related Topics

- [Changing a system parameter using the data federation administration tool](#)
- [Operators that consume memory](#)
- [Promoting optimizations settings made for the data federation service](#)

4.2.1 Operators that consume memory

The following are the operators that cause the data federation service to consume memory when you use them in your queries.

- join
- cartesian product
- orderby
- groupby
- groupby when you have a lot of different values in the group (a large group set)

The data federation query engine does not use a significant amount of memory when it performs scans of tables, projections, filters, function evaluation or when it “pushes” the operations down to the sources.

4.3 Using “statistics” to let the application choose the best algorithms for querying sources

“Statistics” are used internally by the data federation query engine to optimize queries.

“Statistics” are not refreshed continuously. The idea is to wait until the system is deployed in production, then run “statistics” at some sample time. Then, “statistics” are gathered and taken into consideration to generate subsequent query plans.

The “statistics” subsystem is actually made of two parts:

- a tool that computes cardinalities from the measures that are known at the data source level
- a recorder that counts the number of times a table or attribute is requested when a query is executed

You can override cardinalities with manual values to influence their usage in optimizing the query plans.

Related Topics

- [About column cardinality](#)
- [Filtering the recorded statistics to compute only those needed to optimize reports](#)

4.3.1 About column cardinality

Cardinality is the number of rows in a column.

You can measure cardinality on other elements, too. It is possible to measure cardinality for a table, for a schema that contains tables, or for an entire catalog. In each case, talking about the cardinality of the object is a shortcut for talking about the cardinalities of all the objects it contains. For example, if we say that the cardinality of a schema is 1000, then we mean that most columns in most tables of the schema have 1000 rows.

When you are working with data federation, the system can optimize its queries better the more precisely it knows the cardinality of the columns in the sources of data. For this reason, the data federation query engine can estimate the cardinalities of the sources of data, and it lets you set the cardinalities if you know them better.

Estimating and setting cardinalities is part of an optimization task called setting “statistics”.

Related Topics

- [Using statistics to let the application choose the best algorithms for querying sources](#)

4.3.2 About the “fanout” value of relationships between columns

Estimating and setting “fanout” values is part of an optimization task called setting “statistics”.

The “fanout” measures an association between the data in two columns. If there are two columns, then for each distinct value in the first column, the “fanout” is the average number of columns in the second column. For example, if one column lists countries, and another column lists cities, then the “fanout” can measure the average number of cities for each country.

When you are working with data federation, the query engine can optimize its queries better the more precisely it knows the “fanout” of the columns in the sources of data. For this reason, the data federation query engine lets you set the “fanout” of the columns in your sources.

Related Topics

- [Using statistics to let the application choose the best algorithms for querying sources](#)

4.3.3 Filtering the recorded “statistics” to compute only those needed to optimize reports

You can compute the “statistics” for all your data sources at once, but this operation may take a long time. The following procedure shows you how to compute only those “statistics” that are needed by your queries in order to speed up this process.

This procedure is based on the example of getting “statistics” generated by the refresh of an SAP BusinessObjects Interactive Analysis document, but could be adapted to any other situation.

Computing cardinalities can be done at any time and does not require any activation.

1. In SAP BusinessObjects Interactive Analysis, open the report in the "Edit Query" panel.
2. Open the SQL text area in the "Query Panel", copy the SQL of the query and close the text area.
3. In the data federation administration tool, paste the SQL in the text area of the **Query Panel** tab.
4. Click **Run**.
5. In the **Statistics** tab, click the Refresh statistics from server button.

The tables and columns that are used to optimize your query are recorded in the **Number of Requests** column.

6. In the **Statistics** tab, ensure the following:
 - Make sure that the value of the filter in the **Number of Requests** column is set to **Recorded**.
7. Ctrl click to select all the rows with a value in the **Number of Requests** column, then click the Compute button.

The data federation administration tool computes only those “statistics” that are useful to your query.

8. Run the actual query by refreshing the query in the SAP BusinessObjects Interactive Analysis report.

The data federation query engine will now use the gathered “statistics” and generate an optimal plan.

Related Topics

- [The Statistics Tab in Data Federation Administration Tool](#)

4.4 Optimizing query plans

4.4.1 The "Query Plan" view in the data federation administration tool

Definition

When you click **Explain Query**, the "Query Plan" view shows the result of the query optimization. The "Query Plan" view has three panes:

- "Plan" pane: displays the query plan in a tree structure
- "Details" pane: displays the details of the highlighted item in the "Plan" pane
- "Properties" pane: displays the properties of the highlighted item in "Plan" pane and "Details" pane.

The "Plan" pane displays a query plan in a tree structure with leaves representing connector queries sent to connectors. The intermediate nodes are Projection, Order By, Group By, Aggregation, Union, Full Outer Join, Calculation (filter, join) etc.

This document only describes general information about the query and connector queries. This is the information (without intermediate nodes) shown to the user by default.

1. Overall information for a query:

- a. In the "Properties" pane:
 - a. Memory Used: the estimated memory required for the query.
 - b. Number of Concurrent Memory Consuming Operators : the maximum number of memory-consuming operators that are executed concurrently in the query plan.
- b. In the "Details" pane:
 - a. **Statistics**
 - a. **Table cardinality**: the estimated number of rows returned by this query

2. Information for Connector Query:

- a. In the "Properties" pane:
 - a. **id**: the identifier of the connector query
 - b. Data Federation SQL: the connector query represented in the SQL syntax used by the multi-connector query engine
 - c. Native Connector Query: the connector query represented in native syntax (supported by the connector)
 - d. Connector Name: the name of the connector
- b. In the "Details" pane:
 - a. **Schema**: the list of projected columns of the connector query
 - b. **Keys**: derived keys (key deduced from table's keys)
 - c. **Statistics**: the statistics used by the optimizer and their respective estimated values
 - a. **Table cardinality**
 - b. **Column cardinality**
 - d. **Capabilities**: this is a list of operations that the connector can perform
 - e. **Semi-Joins**: the list of semi-joins
 - a. **Filtered columns**: this is the list of columns used in semi-joins

- a. **Dependent Columns:** the columns used to filter this (filtered) column
- b. **Dependent Source Queries:** the list of connector queries that provide the values for the semi-join
- c. **Strategies:** the list of execution strategies for the semi-join operator in order of preference
- d. **Reduction Factor:** the ratio between the number of rows returned without semi-join and the number of rows returned with semi-join
- f. **Data Federator SQL:** the connector query represented in the SQL syntax used by the query engine
- g. **Native Connector Query:** the connector query represented in native syntax (supported by the connector)

4.4.2 The Explain Statistics Command

Description

The **Explain Statistics** command lists all the “statistics” needed by the query engine to optimize an SQL Query. When the command is executed for a query, a tree-like structure is returned. This view lets you see for each source what tables are used in the query, which “statistics” are required and if they are updated. In this view you can:

1. Refresh all the “statistics” needed by the query in one click.
2. Refresh the “statistics” of a particular table or column.
3. Set the “statistics” of a particular table or column.
4. Ensure that all the needed “statistics” for generating the best plan are available.
5. See which “statistics” are used: From the source or the ones set by the user.

The result of the command has 6 columns:

- **Catalogs:** The tree-view where the user can browse the source and tables/columns.
- **Last compute date:** The last time when the “statistics” were calculated from the source.
- **Number of Requests:** The number of times the distinct value of the column (the cardinality of table) has been requested in the system (not just for this query).
- **Current Cardinality:** There are two types of cardinalities possible: the cardinality from the source and the cardinality from the user (admin). Depending upon which policy is used, the appropriate cardinality is shown as the current cardinality.
- **Cardinality from Source:** The cardinality from the data source.
- **User Cardinality:** If the user sets a different statistic for a particular table or column, it is shown here.

4.4.3 Using the explain query feature to get feedback to tune a query

You can use the “explain” query feature as feedback to tune a query. The following query performs a join between two tables from two different data sources. *T1* is from data source *S1* and is a small table; *T2* is from data source *S2* and is a large table.

1. In the "Query Panel" enter `Select * From T1, T2 where T1.C1 = T2.C2`
2. Click **Explain query**.
3. Click the source queries *S1* [*T1*], *S2* [*T2*] in the "Plan" panel

The detailed information will display in the **Details** panel. Looking at the details, you can see that both source queries for *S1* and *S2* are full table scans. But since you know that *T1* is a small table, you would expect a semi-join to be generated on *S2*. To investigate why a semi-join is not generated, you can look at the “statistics” of both source queries: you can see that the optimizer is trying to use:

- The cardinality of *T1*
- The cardinality of *T1.C1*
- The cardinality of *T2*
- The cardinality of *T2.C2*

But all these “statistics” are marked as unknown.

4. Click **Explain Statistics**

The "Query Statistics" tab is displayed.

In the "Query Statistics" tab, you can set the “statistics” to the following values:

- Cardinality(*T1*)=25
- Cardinality(*T1.C1*)=25
- Cardinality(*T2*)=100000
- Cardinality(*T2.C2*)=100000

5. Click **Explain query** again

You get a different plan: a semi-join is generated for *S2*.

Related Topics

- [The Query Plan view in the data federation administration tool](#)
- [Guidelines for using system parameters to optimize queries on small tables joined to large tables](#)

4.4.4 Checking if an operator was “pushed” using the data federation administration tool

Your queries are generally more efficient when your database systems, instead of the data federation query engine, evaluate operators.

You can check if an operator is being “pushed” in the **Query Monitoring** tab of the data federation administration tool.

1. In the data federation administration, tool open the **Query Monitoring** tab.
2. Click the **Refresh** button to see the most recent queries.
3. Find your query, and look in its subqueries to see if your operators are being “pushed”.
 - If your operator is listed in a subquery, it means it is being “pushed” to the source of data.
 - If your operator is only listed in the top query, it means it is not being “pushed”.

To force data federation query engine to “push” the operator to the source of data, you can try setting the capabilities of the connector to your source of data so that it accepts the operator.

Related Topics

- [Tuning the performance of data federation queries](#)
- [The Query Monitoring tab in data federation administration tool](#)
- [Setting the capabilities of relational and SAS connectors using the data federation administration tool](#)

4.4.5 Guidelines for using system parameters to optimize queries on small tables joined to large tables

While optimizing queries, the data federation optimizer attempts to reduce data transfer from data sources to the query engine. One way to achieve this is to generate semi-joins while accessing large tables in data sources. The optimizer only attempts to generate semi-joins when there is an estimated performance gain.

The generation and execution of semi-joins are governed by the following system parameters and connector properties:

- **ACTIVATE_SEMI_JOIN_RULE**

Whether the semi-join generation rule is activated. The optimizer attempts to generate semi-joins only if this parameter is set to true.

- **MIN_SOURCE_CARDINALITY_THRESHOLD_FOR_SEMI_JOIN_RULE**

The minimum cardinality of the source query for which the optimizer attempts to generate a semi-join. The optimizer only attempts to generate semi-joins for source queries that return a large amount of data. If the estimated cardinality of the source query is less than this parameter, the optimizer does not attempt to generate semi-joins for this source query.

- **MIN_ACTIVATION_THRESHOLD_FOR_SEMI_JOIN_RULE**

The purpose of a semi-join is to reduce data transfer from the data sources to the query engine. This parameter is the minimum reduction of data transfer for which the optimizer generates a semi-join. The ratio calculated by: (Number of rows without semi-join / number of rows with semi-join) is called the reduction factor. If the reduction is greater than this parameter, a semi-join is generated; otherwise no semi-join is generated.

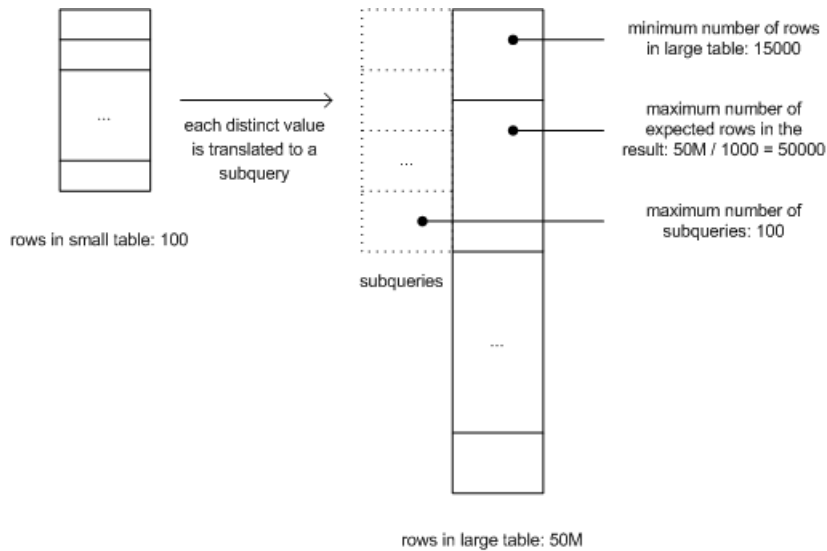


Figure 4-1: How the data federation query engine decides to activate a “semi-join” with parameters `ACTIVATE_SEMI_JOIN_RULE=true`, `MIN_SOURCE_CARDINALITY_THRESHOLD_FOR_SEMI_JOIN_RULE=15000`, and `MIN_ACTIVATION_THRESHOLD_FOR_SEMI_JOIN_RULE=1000`

Note:

When you want to promote a data foundation to another system, and you have changed system parameters to optimize the queries against the data foundation, you must also promote the system parameters. You can use the Lifecycle management console for SAP BusinessObjects Enterprise to do this.

Example: Activating a “semi-join” on a query with a small table and a very large table

This example shows how to set system and session parameters to activate the “semi-join”, when you have a small table containing 100 rows and a large table with 50M rows. We also assume that when the values of the small table are used to filter the values in the large table, 10000 rows will be returned.

Refresh the “statistics” once your data federation project has been deployed. You can refresh “statistics” in the data federation administration tool.

Set `MIN_SOURCE_CARDINALITY_THRESHOLD_FOR_SEMI_JOIN_RULE` to 15000. The number of rows in the large table exceeds 15000, so this value will allow the query engine to use a “semi-join”.

Set `MIN_ACTIVATION_THRESHOLD_FOR_SEMI_JOIN_RULE` to 1000. This is a good default value. It is used as follows.

The number of rows in the large table is divided by this number to calculate a threshold. In this case, the threshold is 50000 ($50M / 1000 = 50000$). The query engine then checks the “statistics”, which show that the “semi-join” will return about 10000 rows. This is under the threshold of 50000 and therefore allows data federation application to use the “semi-join”.

If you set this value too low, the query engine will use a “semi-join” when it is not efficient. For example, if you set this value to 1, the query engine will use a “semi-join” even when the number of rows returned by the “semi-join” is 50M ($50M / 1 = 50M$). This is equivalent to doing a full table scan.

If you set this value to 2, the query engine will use a “semi-join” when the number of rows returned by the “semi-join” is half of that returned by a table scan. This is not a sufficient gain over a full table scan.

If you set this value too high, the query engine will not use a “semi-join” when it would be efficient. For example, if you set this value to 50M, the query engine will only use the “semi-join” if the number of rows returned by the “semi-join” is 1 ($50M / 50M = 1$).

Setting this value to 1000 is generally equivalent to requesting that the “semi-join” be activated when its result is 1000 times smaller than a table scan.

With these settings, the query engine should be able to perform a “semi-join” and thus run your query with optimal speed and use of memory.

Related Topics

- [Promoting optimizations settings made for the data federation service](#)

4.4.6 Guidelines for using system parameters to optimize queries on large tables with data that can be sorted

When your queries return large tables, and the data in those tables can be sorted, the application can use order-based operators to speed up operation. The order-based operators are “merge join” and order-based group by.

A “merge join” “pushes” an order by operator on the sources and then uses the ordered results to perform a join on the fly.

This technique avoids storing the results that need to be joined. It is thus faster than applying a join on unordered results.

Checking when “merge join” is useful

Nevertheless, the “merge join” is only useful if all of the following conditions are met:

- if a semi-join is not possible
- if your query returns big tables to join
- if the source of data supports the order by operator, or if your data is appropriate for using the order by operator
 - You can check the capabilities of your sources of data to learn if they support the order by operator.

For example, DB2 does not support predictable ordering of null values.

- Also, on some sources, order by is not supported because the collation settings are not predictable.

For example, while DB2 does not support predictable ordering of null values, if you know that your data has no nulls, you can still use a “merge join”. In this case, set the capabilities of the source to force it to perform an order by.

Checking when to change “merge join” parameters

“Merge join” is activated on large tables by default. You can use system parameters to control activation of a “merge join”.

You may need to set the parameters under certain conditions, as follows:

- You have big tables, but the size is distributed among big rows. Queries do not return more than the required minimum rows.
- You have small tables, but you want to use a “merge join” anyway.

Verifying that “merge join” is activated

To verify that the “merge join” is working, use the data federation administration tool to look at your query history, and check that your subqueries include the order by operator.

Related Topics

- [Using system parameters to control activation of order-based operators](#)
- [Guidelines for using system parameters to optimize queries on small tables joined to large tables](#)
- [Setting the capabilities of relational and SAS connectors using the data federation administration tool](#)

4.4.7 Using system parameters to control activation of order-based operators

You can use the following parameters to trigger order-based operators:

- Set the server parameter `ACTIVATE_ORDER_BASED_OPTIMIZATION_RULE` to true in order to activate optimizer rules to detect advantageous use of order-based operators.
- Set the server parameters `MIN_STORE_CARDINALITY_THRESHOLD_FOR_ORDER_BASED_JOIN_RULE` and `MIN_TRANSFER_CARDINALITY_THRESHOLD_FOR_MERGE_JOIN_RULE`. Those numbers define minimal cardinality (number of rows) of input operands to choose a merge join operator. A merge join can be chosen only if one operand has a cardinality over `minStoreCardForMergeJoin` and the other operand has a cardinality over `minTransferCardForMergeJoin`.
- Set the server parameter `MIN_CARDINALITY_THRESHOLD_FOR_GROUP_BY_TRANSFORMATION_RULE`. This number defines minimal cardinality of input operand to choose an `orderBased GroupBy` operator.

Note:

When you want to promote a data foundation to another system, and you have changed system parameters to optimize the queries against the data foundation, you must also promote the system

parameters. You can use the Lifecycle management console for SAP BusinessObjects Enterprise to do this.

Related Topics

- [Promoting optimizations settings made for the data federation service](#)

4.4.8 Forcing parallel execution of data source sub-queries

By default, the data federation application submits the execution of a sub-query to a data source only when the data federation application is ready to consume the result of the sub-query. Doing this, the data federation application reduces the time the query result has to be cached by the underlying database and avoids timeout effects when this time is too large.

However it is possible to force early submission of data source queries:

Set the server parameter `ACTIVATE_MULTI_THREADED_UNION_OPERATOR` to true in order to activate parallel submission of data source sub-queries which are operands of a union operator.

4.4.9 Semi-join execution strategies

Introduction

When the data federation service applies the “semi-join” operator to optimize a join between a small table and a large table, it can use one of the following strategies to reduce the number of rows from the large table.

Each of these strategies creates a relatively small list of values and joins the rows in the large table against this list. The execution strategy is simply the technical means of creating this list. Not all sources of data support the same techniques.

You can use the parameter `SEMI_JOIN_EXECUTION_STRATEGIES` to activate or deactivate these strategies, or to change their order of preference.

Strategy	Description
IN	The data federation query engine constructs the list of values using the IN keyword.
Temporary tables	The data federation query engine constructs the list of values by creating a temporary table on the source of data.

Strategy	Description
Prepared statement	The data federation query engine constructs the list of values using SQL prepared statements, with each value in the list passed as a parameter to the prepared statement.

4.5 Optimizing specific connectors

4.5.1 Increasing concurrency of callbacks for parallel queries to SAP NetWeaver BW

You can use the resource property called `jcoServerProperties` to increase the number of threads that the query server will provide for callbacks from SAP NetWeaver BW.

1. Open the data federation administration tool, and log on using a user account with administration rights.
2. Use the **Connector Configuration** tab to edit the SAP NetWeaver BW connector.
3. Set the connector property named `jcoServerProperties` to the value `jco.server.connection_count=10`.

The default value of this property is 2. The maximum recommended value is 10, except if system parameter `MAX_CONCURRENT_MEMORY_CONSUMING_QUERIES` is above 10. In this case, the number of threads should be higher than the system parameter value to avoid a famine situation.

Note:

The connector property is named `jcoServerProperties`. You must set its value to the entire string `jco.server.connection_count=10`.

Related Topics

- [List of connector properties for SAP NetWeaver BW data sources](#)

4.5.2 Changing the size of response packages from queries to SAP NetWeaver BW

You can use the resource property called `packageSize` to change the size of packages of data that are returned in query responses from SAP NetWeaver BW. The size of packages is measured by the number of rows per package.

When you increase the package size, you can get more speed, but you will use more memory.

Conversely, when you decrease the package size, you can get less speed, but you will save memory.

1. Open the data federation administration tool, and log on using a user account with administration rights.
2. Right click the **Connector Configuration** tab to edit the SAP NetWeaver BW connector.
3. Edit the property `packageSize` and enter the desired number of rows per package as the value of this property.

For details, see the description of the property `packageSize` in the list of SAP NetWeaver BW connector properties.

Related Topics

- [List of connector properties for SAP NetWeaver BW data sources](#)

4.6 Promoting optimizations settings made for the data federation service

When you migrate your data foundations from one system to another, for example from a development to a testing system, this task is called promoting.

If you have made optimization changes to the system parameters for the data federation service, you must promote the system parameters while promoting your data foundation.

You can do this using the Lifecycle management console for SAP BusinessObjects Enterprise.

1. Open the Lifecycle management console for SAP BusinessObjects Enterprise.
2. Expand the folder **Folders and Objects**, then click **Data Federation**.
3. Promote the object **Parameters**.

See the *User Guide* for Lifecycle management console for SAP BusinessObjects Enterprise for details.

Configuring connectors to sources of data

5.1 Viewing the information for a connector in the data federation administration tool

1. Start the data federation administration tool.
2. Click the **Connector Configuration** tab.
3. Double-click a connector in the tree list.
4. Click the **General information** to view the settings, or **Capabilities** to view the capabilities of the connector.

5.2 Changing the properties of a connector in the data federation administration tool

1. Start the data federation administration tool.
2. Click the **Connector Configuration** tab.
3. Right-click a connector in the tree list, then click **Create configuration**.
4. Double-click the **configuration** node that appears.
5. In the **Configuration Properties** tab, double-click a property to edit it, change the value, then click the Save data icon to save your changes.

5.3 Configuring connectors for relational data sources

5.3.1 List of common connector properties for relational data sources

The table below lists the common properties that you can configure for relational data sources.

Property	Description
capabilities	<p>A list of all capabilities supported by the database. Elements are separated by the character ';' (no space between elements).</p> <p>Example</p> <pre>capabilities=fullSQL\=true;outer join\=false;rightouterjoin\=true</pre>
compCollationCompatible	<p>True/Yes or False/No</p> <p>Tells if the collation for comparison operations in the data source is compatible with the current setting on the data federation service. When set to <code>true</code>, the server can ignore the collation of comparison operations and predicates can be safely pushed on the source. Defaults to <code>false</code>.</p> <p>Example</p> <pre>compCollationCompatible=true</pre>
sortCollationCompatible	<p>True/Yes or False/No</p> <p>Tells if the collation for sort operations (<code>ORDER BY</code>) in the data source is compatible with the current setting on the data federation service. When set to <code>true</code>, the server can ignore the collation of sort operations and (<code>ORDER BY</code>) expressions can be safely pushed on the source. Defaults to <code>false</code>.</p> <p>Example</p> <pre>sortCollationCompatible=true</pre>
longVarCharMaxSize	Limits the size for longvarchar data types (like text data type).
varCharMaxSize	Limits the size for varchar data types. The default value is -1 which mean no truncation.
arrayFetchBufferSize	Defines the maximal size in bytes of the buffer dedicated to each array fetch. The default value is 65536.
enableArrayFetchSizeOptimization	Enables or disables the optimization of the array fetch size. The default value is <code>true</code> .

Property	Description
maxConnectionIdleTime	<p>The maximum time an idle connection is kept in the pool of connections. Unit is milliseconds. -1 means no limit. If this parameter is not set, then the default connection server pool time is used (10mn) 100000.</p> <p>Possible values are:</p> <ol style="list-style-type: none"> 1. -1: no timeout, connection kept in the pool for the entire wrapper lifetime. 2. 0: connection not managed by the pool. 3. > 0: value is the maximum time a connection can stay idle (in milliseconds).
enableUpdateQueries	<p>True/Yes or False/No</p> <p>Tells if the execution of update queries is enabled.</p> <p>Defaults to True.</p>
enableTemporaryTableQueries	<p>True/Yes or False/No</p> <p>Tells if the execution of temporary table queries is enabled.</p> <p>Defaults to True.</p>
maxValuesInInClause	<p>Specifies the maximum number of values in the IN clause.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>
semiJoinMaxQueries	<p>Specifies the maximum number of queries that a semi-join operator can execute.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>
semiJoinTempTableMinCardinality	<p>Specifies the minimum cardinality of the dimension in the temporary table strategy of the semi-join operator.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>

Property	Description
<code>semiJoinTempTableMaxCardinality</code>	<p>Specifies the maximum cardinality of the dimension in the temporary table strategy of the semi-join operator.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>
<code>semiJoinExecutionStrategies</code>	<p>Specifies the list of execution strategies for the semi-join operator in order of preference.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> • <code>DEFAULT</code> <p><code>DEFAULT</code> means that the value of the system parameter <code>SEMI_JOIN_EXECUTION_STRATEGIES</code> is used.</p> <ul style="list-style-type: none"> • A combination of <code>I,T,P</code> separated by commas in order of preference, where <code>I</code> stands for <code>IN</code> query execution strategy, <code>T</code> for temporary table execution strategy, and <code>P</code> for parameterized query execution strategy. <p>Examples:</p> <ul style="list-style-type: none"> • <code>T,P,I</code> • <code>I,T</code> • <code>P</code> • for no strategy <p>If one of <code>I,T,P</code> is missing then the corresponding execution strategy is not supported by the wrapper.</p> <p>Note: <code>NONE</code> means that no execution strategy is supported by the wrapper.</p> <p>The returned value cannot be <code>null</code> nor equal to empty string.</p>
<code>allowPartialResults</code>	<p>True/Yes or False/No</p> <p>This parameter is used in combination with the parameter <code>maxRows</code>. When <code>maxRows</code> is set to a positive value, and a query returns more rows than the specified limit, an exception is thrown by default. This behavior can be changed setting the parameter <code>allowPartialResults</code> to <code>true</code>. Default value is <code>false</code>.</p>

Property	Description
maxRows	<p>Defines the maximum number of rows to return. This parameter is used in combination with the parameter <code>allowPartialResults</code>.</p> <p>When <code>maxRows</code> is set to a positive value and a query returns more rows than the specified limit, an exception is thrown by default.</p> <p>This behavior can be changed setting the parameter <code>allowPartialResults</code> to <code>true</code>. Default value is <code>0</code>, meaning no limit.</p>
maxLevelOfFunctionNesting	<p>Specifies the maximum number of levels supported for nesting functions. The default value is <code>0</code> which means that there is no limit or the limit is unknown.</p>

Related Topics

- [Complete list of connector capabilities for relational data sources](#)

5.3.2 List of specific connector properties for MySQL data sources

The table below lists the specific properties that you can configure in MySQL connectors.

Property	Description
datasourceCompCollation	<p>The source collation to use for comparisons (except <code>LIKE</code> / <code>NOT LIKE</code> and function evaluations). It is used for SQL Server and MySQL to add collate clause in queries. If unset, no collate clause is generated for these operations. Unset by default.</p>
datasourceSortCollation	<p>The source collation to use for sort operations (<code>ORDER BY</code>). It is used for SQL Server and MySQL to add collate clause in queries. If unset, no collate clause is generated for these operations. Unset by default.</p>

Property	Description
<code>datasourceBinaryCollation</code>	The source collation to use for comparisons that needs to be evaluated with a binary collation (<code>LIKE / NOT LIKE</code> and function evaluations). It is used for SQL Server and MySQL to add collate clause in queries where binary collation semantics is required. If unset, no collate clause is generated for these operations. Unset by default.
<code>unicodeStrings</code>	True/Yes or False/No. Specifies whether the Unicode syntax should be used for string constants pushed to the database. Defaults to <code>False</code> .

Related Topics

- [Complete list of connector capabilities for relational data sources](#)

5.3.3 List of specific connector properties for Teradata data sources

The table below lists the specific property that you can configure in Teradata connectors.

Property	Description
<code>sampleSize</code>	Defines to maximum number of rows to return using <code>SAMPLE</code> operator.

Related Topics

- [Complete list of connector capabilities for relational data sources](#)

5.3.4 List of specific connector properties for Sybase ASE data sources

The table below lists the specific properties that you can configure in Sybase ASE connectors.

Property	Description
setQuotedIdentifier	<p>True/Yes or False/No</p> <p>If setQuotedIdentifier=true then quote string identifier is forced to ".</p>

Related Topics

- [Complete list of connector capabilities for relational data sources](#)

5.3.5 List of specific connector properties for SQL Server data sources

The table below lists the specific properties that you can configure in SQL Server connectors.

Property	Description
datasourceCompCollation	<p>The source collation to use for comparisons (except <code>LIKE</code> / <code>NOT LIKE</code> and function evaluations). It is used for SQL Server and MySQL to add collate clause in queries. If unset, no collate clause is generated for these operations. Unset by default.</p> <p>Example</p> <pre>datasourceCompCollation=Latin1_general_ci_ai</pre>
datasourceSortCollation	<p>The source collation to use for sort operations (<code>ORDER BY</code>). It is used for SQL Server and MySQL to add collate clause in queries. If unset, no collate clause is generated for these operations. Unset by default.</p> <p>Example</p> <pre>datasourceSortCollation=Latin1_general_ci_as</pre>

Property	Description
<code>datasourceBinaryCollation</code>	<p>The source collation to use for comparisons that needs to be evaluated with a binary collation (<code>LIKE</code> / <code>NOT LIKE</code> and function evaluations). It is used for SQL Server and MySQL to add collate clause in queries where binary collation semantics is required. If unset, no collate clause is generated for these operations. Unset by default.</p> <p>Example</p> <pre>datasourceBinaryCollation=Latin1_general_bin</pre>
<code>unicodeStrings</code>	<p>True/Yes or False/No. Specifies whether the Unicode syntax should be used for string constants pushed to the database. Defaults to <code>False</code>.</p>

Related Topics

- [Complete list of connector capabilities for relational data sources](#)

5.3.6 List of specific connector properties for Generic ODBC or JDBC data sources

The table below lists the specific properties that you can configure in Generic connectors.

Property	Description
sqlDialect	<p>Identifies the SQL dialect supported by the database. One of:</p> <ul style="list-style-type: none"> • sql92 • sql99 (reserved for future usage) • jdbc3 (JDBC syntax is used for outer joins) • odbc • oracle • sqlserver • ids (Informix Dynamic Server) • teradata • maxdb • greenplum • postgresql <p>Defaults to the SQL dialect supported by the source as identified by the parameter <code>sourceType</code>. If <code>sourceType</code> is undefined, then default to <code>sql92</code>.</p>
supportsCatalog	Tells if the connector supports the notion of catalog. Defaults to <code>true</code> .
supportsSchema	Tells if the connector supports the notion of schema. Defaults to <code>true</code> .
supportsBoolean	<p>True/Yes or False/No</p> <p>False if the JDBC driver or database does not support booleans as first class objects. The default value for this parameter depends on the database. If this is a one of the supported source type, this parameter is already set to its correct value. However, it can be override. Defaults to <code>false</code>.</p>
useIndexInOrderBy	<p>Specifies if index (column position) should be used instead of alias (column name) in the <code>ORDER BY</code> clause of submitted queries. Defaults to <code>false</code> (except for databases which does not handle well aliases in <code>ORDER BY</code> clause).</p> <p>Example</p> <p>If we order by column 2 and 3, we will generate <code>ORDER BY 2, 3</code> instead of <code>ORDER BY C2, C3</code>.</p>

Property	Description
escapeIdentifierQuoteString	Defines the string used to escape the identifier quote string (as returned by <code>java.sql.DatabaseMetaData#getIdentifierQuoteString</code>) when it appears inside an identifier. By default, this escape string is set to the identifier quote string itself. If set to "", no escape will be done.
ignoreKeys	True/Yes or False/No No if the wrapper should not query the JDBC driver to get keys/foreign keys metadata. (sun jdbc-odbc bridge does not support such calls and this option should be set to <code>true</code>). Default: No.
supportsTemporaryTables	True/Yes or False/No False if the source does not support temporary tables (or we want to disable the generation of the temporary tables). Defaults to <code>false</code> .
supportsTableCardinality	True/Yes or False/No False if the source cannot compute table cardinality. Defaults to <code>true</code> .
supportsColumnCardinality	True/Yes or False/No False if the source cannot compute column cardinality. Defaults to <code>true</code> .

Related Topics

- [Complete list of connector capabilities for relational data sources](#)

5.3.7 List of specific connector properties for Oracle data sources

The table below lists the specific properties that you can configure in Oracle connectors.

Property Type	Property Value	Property Description
unicode Strings	True/Yes or False/No	Specifies whether the Unicode syntax should be used for string constants pushed to the database. Defaults to <code>False</code> .

5.3.8 List of specific connector properties for SAP HANA data sources.

The table below lists the specific properties that you can configure in SAP HANA connectors.

Property Type	Property Value	Property Description
unicode Strings	True/Yes or False/No	Specifies whether the Unicode syntax should be used for string constants pushed to the database. Defaults to <code>False</code> .

5.3.9 List of specific connector properties for MaxDB data sources

The table below lists the specific properties that you can configure in MaxDB connectors.

Property Type	Property Value	Property Description
unicode Strings	True/Yes or False/No	Specifies whether the Unicode syntax should be used for string constants pushed to the database. Defaults to <code>False</code> .

5.4 Configuring connectors for SAS

Before configuring connectors for SAS, you must install SAS middleware and drivers.

For details on configuring middleware and drivers for SAS, see the *Data Access Guide*.

5.4.1 List of connector properties for SAS data sources

The table below lists the specific properties that you can configure in SAS connectors.

Property	Description
maxConnections	The maximum number of simultaneous connections to the underlying database. 0 means no limit. Defaults to 0.
maxConnectionIdleTime	The maximum time an idle connection is kept in the pool of connections. Unit is milliseconds. 0 means no limit. Defaults to 60000 (60 seconds).
maxPoolSize	The maximum number of idle (free) connections to keep in the pool. 0 means no limit. Defaults to 32.
maxIdlePools	The maximum number of pools that can be kept idle. If this value is reached, the oldest unused pool is closed and removed. 0 means no limit. Defaults to 24.
connectionTestQuery	<p>The SQL test query that can be used to check if connections to the underlying database are valid. Caution: this query should be "cheap" to execute. An empty string means no test query. Defaults to empty string.</p> <p>Example</p> <p>An example of a test query could be <code>SELECT 1 FROM DUAL</code>.</p>
connectionFailureDetectionOnError	A keyword telling the kind of connection failure detection that should be done when a <code>SQLException</code> is thrown by the underlying database.

Property	Description
connectionFailureSQLStates	<p>The list of specific SQLState codes that can be used to detect a connection failure when an <code>SQLException</code> is thrown by the underlying database. Standard codes for connection failures (starting with the two character class 08) do not need to be specified here. An example of specific code for Oracle can be 61000: (ORA-00028: your session has been killed). Elements are separated by the character ; (no space between elements). Defaults to empty.</p>
driverProperties	<p>A list of driver properties. You must separate the properties with the character ; (no space between properties).</p> <p>The properties you can enter are the same as those that are available for the driver that you are using to connect to the database. See the document for the driver for a list of the properties.</p> <p>Example</p> <pre>driverProperties=selectMethod\=cursor;connectionRetryCount\=2</pre>
sessionProperties	<p>A list of session properties set on the database. You must separate the properties with the character ; (no space between properties).</p> <p>The properties you can enter are the same as those that are available for the database to which you are connecting. See the document for the database for a list of the properties.</p> <p>Example</p> <pre>sessionProperties=selectMethod\=cursor;connectionRetryCount\=2</pre>
capabilities	<p>A list of all capabilities supported by the database. Elements are separated by the character ; (no space between elements).</p> <p>Example</p> <pre>capabilities=fullSQL\=true;outerjoin\=false;rightouterjoin\=true</pre>

Property	Description
useParameterInlining	When set to <code>true</code> , the JDBC wrapper do not use <code>java.sql.PreparedStatement</code> objects to execute a parameterized query but uses <code>java.sql.Statement</code> objects. The parameterized query is inlined, replacing placeholder with constant values. This option is useful for JDBC drivers that do not support well prepared statements. Defaults to <code>false</code> .
castColumnType	<p>A list of <code>databasetype=jdbctype</code> type mappings. This is useful when the default mapping done by the driver is incorrect or incomplete. Note: for our officially supported databases the type mappings are implicitly set but it can be overwritten by a user.</p> <p>Example</p> <p>For oracle JDBC driver <code>castColumnType=FLOAT\=FLOAT;BLOB\=BLOB</code></p>
enableUpdateQueries	<p><code>True/Yes</code> or <code>False/No</code></p> <p>Tells if the execution of update queries is enabled.</p> <p>Defaults to <code>True</code>.</p>
enableTemporaryTableQueries	<p><code>True/Yes</code> or <code>False/No</code></p> <p>Tells if the execution of temporary table queries is enabled.</p> <p>Defaults to <code>True</code>.</p>
defaultFetchSize	<p>The default fetch size to set when creating <code>java.sql.Statement</code>. 0 means that the fetch size is not set.</p> <p>Gives the connector a hint as to the number of rows that should be fetched from the database when more rows are needed.</p> <p>Default: 0 (fetch size is not set)</p>

Property	Description
compCollationCompatible	<p>True/Yes or False/No</p> <p>Tells if the collation for comparison operations in the data source is compatible with the current setting on the data federation service. When set to <code>true</code>, the server can ignore the collation of comparison operations and predicates can be safely pushed on the source. Defaults to <code>false</code>.</p> <p>Example</p> <pre>compCollationCompatible=true</pre>
sortCollationCompatible	<p>True/Yes or False/No</p> <p>Tells if the collation for sort operations (ORDER BY) in the data source is compatible with the current setting in the data federation query service. When set to <code>true</code>, the server can ignore the collation of sort operations and (ORDER BY) expressions can be safely pushed on the source. Defaults to <code>false</code>.</p> <p>Example</p> <pre>sortCollationCompatible=true</pre>
datasourceCompCollation	<p>The source collation to use for comparisons (except LIKE / NOT LIKE and function evaluations). It is used for SQL Server and MySQL to add collate clause in queries. If unset, no collate clause is generated for these operations. Unset by default.</p> <p>Example</p> <pre>datasourceCompCollation=Latin1_general_ci_ai</pre>
datasourceSortCollation	<p>The source collation to use for sort operations (ORDER BY). It is used for SQL Server and MySQL to add collate clause in queries. If unset, no collate clause is generated for these operations. Unset by default.</p> <p>Example</p> <pre>datasourceSortCollation=Latin1_general_ci_as</pre>

Property	Description
<code>datasourceBinaryCollation</code>	<p>The source collation to use for comparisons that needs to be evaluated with a binary collation (<code>LIKE / NOT LIKE</code> and function evaluations). It is used for SQL Server and MySQL to add collate clause in queries where binary collation semantics is required. If unset, no collate clause is generated for these operations. Unset by default.</p> <p>Example</p> <pre>datasourceBinaryCollation=Latin1_general_bin</pre>
<code>sqlDialect</code>	<p>Identifies the SQL dialect supported by the database. One of:</p> <ul style="list-style-type: none"> • <code>sql92</code> • <code>sql99</code> (reserved for future usage) • <code>oracle</code> • <code>sqlserver</code> • <code>jdbc3</code> (JDBC syntax is used for outer joins) • <code>sas</code> <p>Defaults to the SQL dialect supported by the source as identified by the parameter <code>sourceType</code>. If <code>sourceType</code> is undefined, then default to <code>sql92</code>.</p>
<code>useIndexInOrderBy</code>	<p>Specifies if index (column position) should be used instead of alias (column name) in the <code>ORDER BY</code> clause of submitted queries. Defaults to <code>false</code> (except for databases which does not handle well aliases in <code>ORDER BY</code> clause).</p> <p>Example</p> <p>If we order by column 2 and 3, we will generate <code>ORDER BY 2, 3</code> instead of <code>ORDER BY C2, C3</code>.</p>
<code>escapeIdentifierQuoteString</code>	<p>Defines the string used to escape the identifier quote string (as returned by <code>java.sql.DatabaseMetaData#getIdentifierQuoteString</code>) when it appears inside an identifier. By default, this escape string is set to the identifier quote string itself. If set to <code>""</code>, no escape will be done.</p>

Property	Description
ignoreKeys	<p>True/Yes or False/No</p> <p>No if the wrapper should not query the JDBC driver to get keys/foreign keys metadata. (sun jdbc-odbc bridge does not support such calls and this option should be set to <code>true</code>). Default: No.</p>
transactionIsolation	<p>The transaction isolation level. One of:</p> <ul style="list-style-type: none"> TRANSACTION_READ_COMMITTED TRANSACTION_READ_UNCOMMITTED TRANSACTION_REPEATABLE_READ TRANSACTION_SERIALIZABLE <p>Default: not set.</p>
setFetchForwardDirection	<p>True/Yes or False/No</p> <p>True if fetch forward should be explicitly set. Default: False.</p>
setReadOnly	<p>True/Yes or False/No</p> <p>False if setting the connection to read only should not be done. Default: False.</p>
metadataFetchMode	<p>The metadata fetch mode used for SAS datasources only.</p> <p>One of:</p> <ul style="list-style-type: none"> <code>eager</code>: all metadata is fetched in one shot. <code>lazy</code>: metadata is fetched on demand. <p>Default: <code>lazy</code>.</p>

Property	Description
sasWeights	<p>A mapping between table name and his weight used to order the tables in the <code>FROM</code> clause when generating a query in the SAS dialect. Tables in the <code>FROM</code> clause are ordered according to weights, in descending order. The weight is by default set to the table cardinality but it can be overridden using this parameter. This ordering is done only for inner joins.</p> <p>A table name here is the name as exported by the wrapper. A weight is a long value.</p> <p>If this parameter is not specified, or if no weight is defined for a given table, then the weight is by default the cardinality of the table (as set on the data federation service).</p> <p>If a table name is unknown, it is simply ignored.</p> <p>This parameter is taken into account only when the parameter <code>sqlDialect="sas"</code>.</p> <p>Example</p> <pre>sasWeights=EMPLOYEE\=16;DEPARTMENT\=4</pre> <p>Using this setting, the <code>EMPLOYEE</code> table will appear before the <code>DEPARTMENT</code> table when pushing a query on SAS with a join of this two tables.</p>
addCompensationPredicates	<p>True/Yes or False/No</p> <p>False if we want to disable the generation of the compensation predicates.</p> <p>The compensation predicates are <code>IS NOT NULL</code> conditions added to enforce the SQL semantics for null values (SAS is not SQL compliant).</p> <p>These extra conditions can impact the performance. If a user does not care about enforcing the SQL semantics for null values, he can set this parameter to <code>false</code>.</p> <p>This parameter is taken into account only when the parameter <code>sqlDialect="sas"</code>.</p> <p>Defaults to <code>True</code>.</p>

Property	Description
trimTrailingSpaces	<p>True/Yes or False/No</p> <p>Some JDBC drivers are returning metadata padded with blank spaces. Setting this parameter to Yes will ensure that extra spaces in catalog, schema, table, column, key and foreign key names will be removed. The default value for this parameter is No.</p>
maxValuesInInClause	<p>Specifies the maximum number of values in the IN clause.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>
semiJoinMaxQueries	<p>Specifies the maximum number of queries that a semi-join operator can execute.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>
semiJoinTempTableMinCardinality	<p>Specifies the minimum cardinality of the dimension in the temporary table strategy of the semi-join operator.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>
semiJoinTempTableMaxCardinality	<p>Specifies the maximum cardinality of the dimension in the temporary table strategy of the semi-join operator.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>

Property	Description
semiJoinExecutionStrategies	<p>Specifies the list of execution strategies for the semi-join operator in order of preference.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> • DEFAULT <p>DEFAULT means that the value of the system parameter <code>SEMI_JOIN_EXECUTION_STRATEGIES</code> is used.</p> <ul style="list-style-type: none"> • A combination of I, T, P separated by commas in order of preference, where I stands for IN query execution strategy, T for temporary table execution strategy, and P for parameterized query execution strategy. <p>Examples:</p> <ul style="list-style-type: none"> • T,P,I • I,T • P • for no strategy <p>If one of I, T, P is missing then the corresponding execution strategy is not supported by the wrapper.</p> <p>Note: NONE means that no execution strategy is supported by the wrapper.</p> <p>The returned value cannot be <code>null</code> nor equal to empty string.</p>

Related Topics

- [Complete list of connector capabilities for relational data sources](#)

5.4.2 Optimizing SAS queries by ordering tables in the from clause by their cardinality

SAS is sensitive to the ordering of tables in the from clause. For the fastest response from the SAS/Share server, the table names in the from should appear in descending order with respect to their cardinalities.

You can ensure that the data federation application generates tables in this order by keeping the “statistics” in the data federation application accurate. You can do this using the data federation administration tool.

To control the order of tables manually, you can also set the `sasWeights` resource property for the SAS JDBC connector.

5.5 Configuring connectors for SAP NetWeaver BW

5.5.1 List of connector properties for SAP NetWeaver BW data sources

The table below lists the specific properties that you can configure in SAP NetWeaver BW connectors.

Property	Description
<code>packageSize</code>	<p>Package size for callbacks.</p> <p>This is the number of rows returned by SAP NetWeaver BW to the data federation query engine, per package.</p> <p>The default value is 200.</p> <p>Example</p> <pre>packageSize=300</pre>

Property	Description
programIDMapping	<p>Defines the program IDs for the callback that SAP NetWeaver BW uses to contact Data Federator. The IDs are provided as a list of mappings: server name => program ID. This list is formatted as a string containing key/values separated by ';'. The key represents the server name and the value represents a program ID. Each ID must match the name of an RFC destination created on SAP NetWeaver BW.</p> <p>If this property is not defined, Data Federator will automatically create an RFC destination. This RFC destination will be created using a program ID identical to the name of the RFC destination.</p> <p>The format of the created RFC destination is: <i>DF_JCO_</i> + <i>hostname</i> + <i>_</i> + <i>counter</i>. The <i>hostname</i> is the name of the local host and the <i>counter</i> goes from 0 to 9. However the maximal length of <i>hostname</i> is 23. If the name of local host is bigger than 23 only a prefix of 23 characters is used in the RFC destination name.</p> <p>If the property is defined but there is no mapping listed for the current server, an error is returned.</p> <p>There is not default value for this property (which means that the automatic mode is used).</p> <p>Example 1</p> <pre>MySIA.AdaptiveProcessingServer=RFC1</pre> <p>Example 2</p> <pre>MySIA.DFServer1=RFC1;MySIA.DFServer2=RFC2;...</pre>
useBinaryXML	<p>When set to true, the data exchanged between the connector and the SAP Netweaver BW server is compressed in binary XML format instead of plain text format. This improves performance. By default the value is true.</p> <p>Example</p> <pre>useBinaryXML=true</pre>

Property	Description
checkUnits	<p>When set to true, a query using a measure without its unit is refused, and an error is raised. By default the value is false, and use of a measure without its unit is allowed.</p> <p>Example</p> <pre>checkUnits=false</pre>
forcedCapabilities	<p>Lets you artificially limit the capabilities of the SAP NetWeaver BW connector, if you want the data federation query engine to perform operations instead of SAP NetWeaver BW.</p> <p>Enter the capabilities that you want the data federation query engine to delegate to SAP NetWeaver BW.</p> <p>The valid values are as follows:</p> <ul style="list-style-type: none"> • SCAN_ONLY the data federation query engine only delegates scans to SAP NetWeaver BW. • PROJECTIONS_ONLY the data federation query engine only delegates projections to SAP NetWeaver BW. • empty - the data federation query engine delegates all valid operations to SAP NetWeaver BW. <p>Example</p> <pre>forcedCapabilities=SCAN_ONLY</pre>
jcoDestinationProperties	<p>A list of JCO destination properties. Use the character ; to separate properties (do not type spaces).</p> <p>The properties you can use are documented in the API reference for the JCo API, in the interface DestinationDataProvider.</p>
jcoServerProperties	<p>A list of JCO server properties. Use the character ; to separate properties (do not type spaces).</p> <p>The properties you can use are documented in the API reference for the JCo API, in the interface ServerDataProvider.</p>

Property	Description
authorityCheck	<p>Sets the parameter in SAP NetWeaver BW that indicates whether or not to check authorizations.</p> <p>The parameter in SAP NetWeaver BW specifies whether SAP NetWeaver BW should do the following.</p> <p>Should it check whether the user account is authorized to see the requested data (read) or should authorizations not be checked at all (none)?</p> <ul style="list-style-type: none"> • true (default): SAP NetWeaver BW checks read authorizations. • false: SAP NetWeaver BW checks no authorizations.
pingTimeout	<p>The value of the timeout in milliseconds used when pinging the SAP server; The default value is 10000 milliseconds.</p> <p>Example</p> <pre>pingTimeout=60000</pre>
maxValuesInInClause	<p>Specifies the maximum number of values in the <code>IN</code> clause.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>
semiJoinMaxQueries	<p>Specifies the maximum number of queries that a semi-join operator can execute.</p> <p>The default value is 0 which means that there is no limit or the limit is unknown.</p>

Property	Description
semiJoinExecutionStrategies	<p>Specifies the list of execution strategies for the semi-join operator in order of preference.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> • DEFAULT <p>DEFAULT means that the value of the system parameter <code>SEMI_JOIN_EXECUTION_STRATEGIES</code> is used.</p> <ul style="list-style-type: none"> • A combination of I, T, P separated by commas in order of preference, where I stands for IN query execution strategy, T for temporary table execution strategy, and P for parameterized query execution strategy. <p>Examples:</p> <ul style="list-style-type: none"> • T,P,I • I,T • P • for no strategy <p>If one of I, T, P is missing then the corresponding execution strategy is not supported by the wrapper.</p> <p>Note: NONE means that no execution strategy is supported by the wrapper.</p> <p>The returned value cannot be <code>null</code> nor equal to empty string.</p> <p>Note: T and P execution strategies are not supported by the connector SAP NetWeaver BW.</p>
enableAuthorizationsFiltering	<p>When set to true, authorizations filtering is enabled: the connector will ensure that filters are automatically added to satisfy the SAP BI authorizations defined for the current user. When the authorization filtering is disabled, the user will get an error if he tries to get unauthorized data. By default the value is false.</p> <p>Example</p> <pre>enableAuthorizationsFiltering=true</pre>

Property	Description
debugReportPrefix	<p>String of maximal length 11 that is included in the name of the ABAP report program. The name of the generated program is Z_RSDRI_DF_TXT_\${debugReportPrefix}_ID or Z_RSDRI_DF_DBG_\${debugReportPrefix}_ID where ID is a numerical value of 3 digits generated on the wrapper side. The generated program can be used by SAP specialists to reproduce a bug of the DF Facade.</p> <p>If not set, then no program is generated.</p> <p>Example</p> <p>MY_HOSTNAME</p>
gatewayHostname	<p>The name of the machine hosting the SAP NetWeaver BW gateway.</p> <p>If not specified, an RFC is executed to let SAP NetWeaver BW to choose the value.</p> <p>Example</p> <p>gatewayHostname=server.wdf.sap.corp</p>
gatewayServiceName	<p>Name or port number of the SAP NetWeaver BW gateway service.</p> <p>If not specified, an RFC is executed to let SAP NetWeaver BW to choose the value.</p> <p>Example</p> <p>gatewayServiceName=sapgw50</p> <p>Example</p> <p>gatewayServiceName=3350</p>

5.5.2 Manually setting the callback ID that SAP NetWeaver BW uses to contact the data federation service

SAP NetWeaver BW uses a callback ID in order to contact the data federation service. A callback is registered automatically when the first query on the SAP NetWeaver BW connector is executed, but you may want to change it, for example to comply with your organization's security policy.

1. Open SAP Logon and log on to the SAP system.
2. Enter se37 in the transaction text field and click execute .
3. Enter function module RSDRI_DF_CONFIGURE click the **TestExecute** icon.

This opens the parameter panel.

4. Set the parameters as follows.

I_ONLY_CHECK	" (empty)
I_RFC_DESTINATION	DF_JCO_ <i>some-hostname</i> _ <i>some-sid</i>
I_REMOVE_CONFIGURATION	" (empty)

For the second parameter, replace *some-hostname* by the hostname of the machine where you installed the server running the data federation service.

Use *_some-sid* as a unique system identifier, to differentiate between potential multiple connections with the same *some-hostname* value.

In this case, `DF_JCO_MYHOST` is a unique identifier that you must reuse in the data federation administration tool.

5. Execute the module.

Note:

It is acceptable to receive the message: RFC Destination already exists.

Make sure you that you clear the **I_ONLY_CHECK** check box.

6. Click **System**, then **Logoff**.
7. Open the data federation administration tool, and log on using a user account with administration rights.
8. Use the **Connector Configuration** tab to edit the SAP NetWeaver BW connector.
9. In the property `programIDMapping`, add a mapping between your server and the string **I_RFC_DESTINATION** (also known as program ID) that you used in SAP NetWeaver BW.

In this case, the value of the property `programIDMapping` is `MySIA.AdaptiveProcessingServer=DF_JCO_MYHOST`.

For details, see the description of the property `programIDMapping` in the list of SAP NetWeaver BW connector properties.

10. Test that the data are available by running a query on a table.

5.5.3 Cleaning the IDs of callbacks for SAP NetWeaver BW connections

Currently, the maximum number of callbackProgramIDs is ten. During normal execution (the server running the data federation service is not stopped abruptly), the callbackProgramIDs are automatically cleaned from the SAP server.

An error occurs when you cannot generate any more callbackProgramIDs on the server (you used them all). The following is a procedure to delete callback names if, due to an abrupt system stop, they were not deleted automatically.

1. Log in to the SAP NetWeaver BW server.
2. Enter the transaction sm59.
3. Click **TCP/IP Connections**.
4. Click on each corresponding connection (*DF_JCO_MYHOST_0* to *DF_JCO_MYHOST_9*) then click the delete icon.

5.5.4 Leveraging SAP analysis authorizations to filter data automatically

A query executed in SAP NetWeaver BW always selects a set of data from the database. If authorization-relevant characteristics are part of this data, you have to make sure that the user who is executing the query has sufficient authorization for the complete selection. Otherwise, an error message is returned indicating that the authorization is not sufficient.

You can instruct the data federation services to leverage analysis authorizations and add automatically filters for the authorized values on each characteristic that is part of your cube. This feature is especially useful when you are using the single sign on authentication mode to connect to SAP NetWeaver BW: you can easily filter data based on each user connecting to the SAP Business Intelligence platform.

Activating the feature

You can activate this feature setting the following SAP NetWeaver BW connector resource property:

- enableAuthorizationsFiltering: true / false

Use the data federation administration tool to configure the resource used by your datasource.

Prerequisites

This feature relies on a component in SAP NetWeaver BW called the “Data Federator facade”.

To check the pre-requisites on SAP NetWeaver BW side, see SAP Note 1500945.

Principle

All characteristics declared as authorization-relevant in SAP NetWeaver BW Data Warehousing Workbench are taken into consideration to compute the set of filters to add automatically.

If a query contains an explicit filter, then no extra authorization filter is added.

If a query does not contain a filter for a given characteristic (column) and this characteristic is authorization-relevant, then filters are automatically added to return authorized values.

Hierarchy authorizations

The data federation services do not expose hierarchies. However, analysis authorizations on hierarchies are taken into account to filter data and return authorized values.

Colon authorizations

If aggregation authorizations (also called colon authorizations) are defined in addition to value based authorizations, the data federation services will always take into account the value based authorizations, even if the column is not in the SELECT clause of the SQL query.

For more information about aggregation authorization logic, see SAP Note 1140831.

Authorizations on multiple hierarchies

When a characteristic has multiple hierarchies on its values, with authorizations on each of these hierarchies, a merge will be performed to return all authorized values from all hierarchies.

Comparison with SAP Business Explorer (BEx)

The SAP NetWeaver BW desktop application BEx Query Designer allows you to define queries and add filters based on analysis authorizations. However, there are some differences between BEx Query Designer and SAP BI data federation services:

- Using BEx Query Designer, the query designer can selectively specify on which characteristic to apply an authorization filter (using authorization variables). With data federation services, authorizations filtering is applied on all characteristics declared as authorization-relevant in SAP NetWeaver BW Data Warehousing Workbench.
- Using the data federation services, when an SQL query contains explicitly a filter on a given characteristic, the authorization filtering is not done on this characteristic and only the explicit filter is used. With BEx Query Designer, you can combine an authorization filter and an explicit filter added by the user.

SAP Notes for SAP NetWeaver BW

See SAP Note 1578089:DBIF: Adding authorizations to filter despite aggregation (<https://service.sap.com/sap/support/notes/1578089>)

Example: Authorization Filtering

An infocube ZCUBE1 in BW contains only 6 rows and the datasource BW_ZCUBE1 is configured in SAP Business Intelligence platform to access it with SSO enabled. When a user with full authorization is used to read fact table, he executes this SQL query:

```
select ZCHA1, ZCHA2, ZCHA3, ZKYF1
from /DF_PROJECT/sources/BW_ZCUBE1/IZCUBE1
```

And the result is:

ZCHA1	ZCHA2	ZCHA3	ZKYF1
A	69226	2001	250.0
A	69226	2000	300.0
B	69190	2001	150.0
B	69190	2000	450.0
C	69115	2001	200.0
C	69115	2000	100.0

Now suppose another user JOE has full authorization on ZCHA1 and ZCHA3 but has access to a single value 69190 for characteristic ZCHA2. If he executes the same SQL query without activating the feature, he will get an error User JOE does not have authorization for InfoProvider ZCUBE1.

When the feature is activated, SAP Business Intelligence platform will return all authorized data and thus the result of the SQL query above is:

ZCHA1	ZCHA2	ZCHA3	ZKYF1
B	69190	2001	150.0
B	69190	2000	450.0

Actually the result is exactly the same as the result of SQL with explicit filter on ZCHA2:

```
select ZCHA1, ZCHA2, ZCHA3, ZKYF1
from /DF_PROJECT/sources/BW_ZCUBE1/IZCUBE1 where ZCHA2 = 69190
```

5.5.5 Architecture of the SAP NetWeaver BW connection in multi-source universes

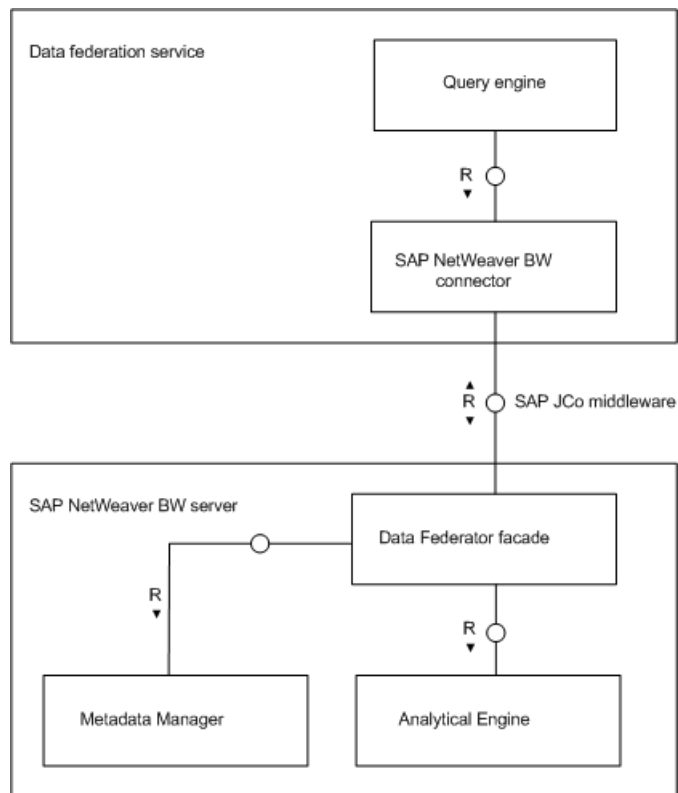


Figure 5-1: Architecture of the SAP NetWeaver BW connection in multi-source universes

5.5.6 Callback sequence of the SAP NetWeaver BW connection in multi-source universes

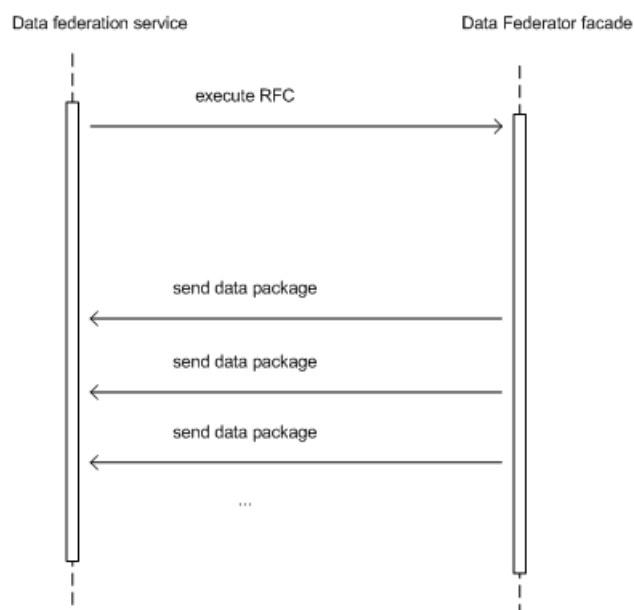


Figure 5-2: Callback sequence of the SAP NetWeaver BW connection in multi-source universes

5.6 Setting the capabilities of relational and SAS connectors using the data federation administration tool

The capabilities of a connector include such things as what kind of operators the source of data supports.

You can set the capabilities of a connector to let the data federation query engine choose to execute operations itself, or delegate them to the source of data.

Generally, it is more efficient to delegate operations to database systems, but not all database systems support the same operators. The list of capabilities tells the data federation query engine to which sources of data it can delegate each operator. This delegation of operators is usually called “pushing”.

Note:

You can only set capabilities for relational or SAS connectors.

1. In the data federation administration tool, click the **Connector Configuration** tab.
2. Right-click your connector in the tree list, then click **Create configuration**.
3. In the **Configuration properties** tab, click the "Value" cell in the "Capabilities" row, and enter your capability in the form my-capability=true;.

Make sure multiple capabilities are separated by a semicolon (;). You can use the values true or false for most capabilities.

Related Topics

- [Complete list of connector capabilities for relational data sources](#)

5.7 Complete list of connector capabilities for relational data sources

The following table lists the capabilities of a connector. You can use these when configuring the resource property named capabilities.

Note:

fullsql is a special capability that allows you to set all capabilities to true by default. Individual capabilities can then be set to false separately if required.

Capability	Comments
fullsql	allows you to set all capabilities to true by default. Individual capabilities can then be set to false separately if required
project	specifies if the connector supports projection operations
orderby	specifies if the connector supports order by operations
orderbystrings	specifies if the connector supports order by operations on string columns
distinct	specifies if the connector supports distinct operations
union	specifies if the connector supports union distinct operations
unionall	specifies if the connector supports union all operations
join	specifies if the connector supports join operations
outerjoin	specifies if the connector supports full outer join operations
leftouterjoin	specifies if the connector supports left outer join operations
rightouterjoin	specifies if the connector supports right outer join operations

Capability	Comments
aggregate	specifies if the connector supports aggregation
aggregatedistinct	specifies if the connector supports aggregation with distinct clause
minaggregate	specifies if the connector supports min aggregate functions
maxaggregate	specifies if the connector supports max aggregate functions
countaggregate	specifies if the connector supports count aggregate functions
avgaggregate	specifies if the connector supports average aggregate functions
sumaggregate	specifies if the connector supports sum aggregate functions
minaggregatedistinct	specifies if the connector supports min aggregate functions with distinct clause
maxaggregatedistinct	specifies if the connector supports max aggregate functions with distinct clause
countaggregatedistinct	specifies if the connector supports count aggregate functions with distinct clause
avgaggregatedistinct	specifies if the connector supports average aggregate functions with distinct clause
sumaggregatedistinct	specifies if the connector supports sum aggregate functions with distinct clause
equalitypredicate	specifies if the connector supports equality predicates
comparisonpredicate	specifies if the connector supports inequality predicates
likepredicate	specifies if the connector supports like predicates

Capability	Comments
nullpredicate	specifies if the connector supports is null predicates
inpredicate	specifies if the connector supports in predicates
arithmeticevaluation	specifies if the connector supports arithmetic operations
booleanevaluation	specifies if the connector supports boolean operations
constantevaluation	specifies if the connector supports literals
emptystringevaluation	specifies if the connector supports empty string literals
cancel	specifies if the connector supports cancelling a query execution
shareconcurrentstatements	specifies if the connector can share can share multiple queries (statements) on a single connection
functionevaluation	specifies if the connector supports function evaluation

Managing system and session parameters

6.1 About system and session parameters

There are two levels of parameters in Data Federator: system and session.

System parameters are shared by a running instance of the data federation query engine.

Session parameters are defined for one connection. The value of these parameters can be different among connections.

Each session parameter takes its default value from the system parameter of the same name. When you change the value of a system parameter corresponding to a session parameter, the new value is only taken into account on new sessions.

You can use system and session parameters to configure various aspects of the data federation query engine, such as the following.

- use of memory
- use of network
- the order of execution of queries
- optimizations

6.2 Changing a system parameter using the data federation administration tool

1. To access the data federation administration tool interface for managing parameters, login to the data federation administration tool, and click the **System Parameters** tab.
2. In the row containing the parameter, type the new value in the **Current value** box, and press **enter**.

6.3 Changing a session parameter using the data federation administration tool

1. To access the data federation administration tool interface for managing parameters, login to the data federation administration tool, and click the **System Parameters** tab, then click **Session Parameters**.

2. In the row containing the parameter, type the new value in the **Current value** box, and press **enter**.

6.4 Setting the capabilities of relational and SAS connectors using the data federation administration tool

The capabilities of a connector include such things as what kind of operators the source of data supports.

You can set the capabilities of a connector to let the data federation query engine choose to execute operations itself, or delegate them to the source of data.

Generally, it is more efficient to delegate operations to database systems, but not all database systems support the same operators. The list of capabilities tells the data federation query engine to which sources of data it can delegate each operator. This delegation of operators is usually called “pushing”.

Note:

You can only set capabilities for relational or SAS connectors.

1. In the data federation administration tool, click the **Connector Configuration** tab.
2. Right-click your connector in the tree list, then click **Create configuration**.
3. In the **Configuration properties** tab, click the "Value" cell in the "Capabilities" row, and enter your capability in the form my-capability=true;.

Make sure multiple capabilities are separated by a semicolon (;). You can use the values true or false for most capabilities.

Related Topics

- [Complete list of connector capabilities for relational data sources](#)

6.5 List of system parameters

System Parameter	Description
ACCEPT_MORE_CONCURRENT_QUERIES	<p>Boolean indicating that we should keep a list of waiting queries (true) instead of throwing an exception if we have a maximum of concurrent queries (A new value of this parameter takes effect when there are no queries registered in the BufferManager.)</p> <p>type: boolean</p> <p>needs restart? no</p> <p>default value: true</p>
ACTIVATE_FREEZE_WHEN_OUT_OF_MEMORY	<p>Special parameter to set the action to be taken if an OutOfMemory is intercepted. If set the system will freeze all managed threads. Some actions may still work through non managed threads, but the system's state can't be trusted. If not set the system will just exit, stopping the java process.</p> <p>type: boolean</p> <p>needs restart? no</p> <p>default value: false</p>
ACTIVATE_JOIN_DISTRIBUTION_RULE	<p>If set to to true, activates the join distribution rule.</p> <p>needs restart? no</p> <p>default value: true</p>
ACTIVATE_JOIN_ELIMINATION_RULE	<p>If set to to true, activates the useless join elimination rule.</p> <p>type: boolean</p> <p>needs restart? no</p> <p>default value: false</p>

System Parameter	Description
ACTIVATE_MULTI_THREADED_UNION_OPERATOR	<p>If set to true, uses the multi-threaded implementation of the Union operator.</p> <p>type: boolean</p> <p>needs restart? no</p> <p>default value: false</p>
ACTIVATE_OPTIMIZED_PREPARED_STATEMENTS	<p>Boolean indicating if the query is optimized for all executions or for each execution of prepared statement. if set to true, use the same query plan for all executions of prepared statement. If set to false, re-optimizes the query for each execution of prepared statement</p> <p>type: boolean</p> <p>needs restart? no</p> <p>default value: false</p>
ACTIVATE_ORDER_BASED_OPTIMIZATION_RULE	<p>If set to to true, activates all rules doing order-based optimization.</p> <p>type: boolean</p> <p>needs restart? no</p> <p>default value: true</p>
ACTIVATE_PROFITABILITY_BASED_JOIN_ORDERING_RULE	<p>If set to to true, activates the order join rule that tries to build bushy trees based on profitability.</p> <p>type: boolean</p> <p>needs restart? no</p> <p>default value: true</p>

System Parameter	Description
ACTIVATE_SEMI_JOIN_DIMENSION_RUNTIME_CARDINALITY_LIMIT	<p>Whether to activate cardinality limitation computed at runtime for semi-join. If this option is activated, the runtime cardinality of a semi-join dimension is compared to the expected value. If the runtime value is greater than the expected value, the dimension is discarded.</p> <p>type: boolean</p> <p>needs restart? no</p> <p>default value: true</p>
ACTIVATE_SEMI_JOIN_RULE	<p>If set to true, activates the rule that generates semi-joins.</p> <p>type: boolean</p> <p>needs restart? no</p> <p>default value: true</p>
AVG_SIZE_OF_BUFFER_ROW	<p>An estimate of the average size of one row. (A new value of this parameter takes effect when there are no queries registered in the BufferManager.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 256</p>
CLUSTER_SYNCHRONIZE_DELAY	<p>Defines the time in seconds between two synchronize events. The fault tolerance module uses synchronize events to identify possible modifications in repositories that were not notified to other cluster members because of a server failure in the middle of a resource modification action.</p> <p>type: long</p> <p>needs restart? no</p> <p>default value: 3600</p>

System Parameter	Description
DEFAULT_COMP	<p>Defines the comp collation. This system parameter is the default value for session parameter COMP.</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: binary</p>
DEFAULT_DECIMAL_PRECISION	<p>The value that is reported by the data federation query server for the decimal precision of a column if the connector does not return a value for the column. Under normal circumstances, the connector always supplies this value.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 27</p>
DEFAULT_DECIMAL_SCALE	<p>The value that is reported by data federation query server for the decimal scale of a column if the connector does not return a value for the column. Under normal circumstances, the connector always supplies this value.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 6</p>
DEFAULT_LOCALE	<p>Defines the ISO locale code for the locale. This system parameter is the default value for session parameter LOCALE.</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: en_US</p>

System Parameter	Description
DEFAULT_SORT	<p>Defines the sort collation. This system parameter is the default value for session parameter SORT.</p> <p>.</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: binary</p>
DEFAULT_STRING_SIZE	<p>The value that is reported by the data federation query engine for the string size of a column if the connector does not return a value for the column. Under normal circumstances, the connector always supplies this value.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 40</p>
EXECUTOR_BUFFER_OVERHEAD	<p>This parameter represents the memory overhead that can be generated during query execution. (A new value of this parameter takes effect when there are no queries registered in the BufferManager.)</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: 10%</p>
EXECUTOR_BUFFER_SIZE	<p>The size of one page parameter, in number of rows. (A new value of this parameter takes effect when there are no queries registered in the BufferManager.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 128</p>

System Parameter	Description
EXECUTOR_STATIC_MEMORY	<p>This parameter represents the minimum memory space allocated to operators upon initialization. It can be expressed in one of two ways:</p> <ul style="list-style-type: none"> As an exact value, for example EXECUTOR_STATIC_MEMORY=50M. <p>The value should be less than the memory space allocated to the executor. See the EXECUTOR_TOTAL_MEMORY parameter.</p> <ul style="list-style-type: none"> As the percentage of the executor memory size, for example EXECUTOR_STATIC_MEMORY=25%. <p>A new value of this parameter takes effect when there are no queries registered in the BufferManager.</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: 25%</p>
EXECUTOR_TOTAL_MEMORY	<p>This parameter represents the memory space allocated to the executor. It can be expressed in one of two ways:</p> <ul style="list-style-type: none"> As the value of memory size, for example EXECUTOR_TOTAL_MEMORY=256M. As the percentage of the memory size allocated by the JVM, for example EXECUTOR_TOTAL_MEMORY=80%. <p>A new value of this parameter takes effect when there are no queries registered in the BufferManager.</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: 80%</p>

System Parameter	Description
FORCE_ASYNCCH_SUBMIT_ON_BW_SOURCES	<p>This parameter, when set to true, forces queries to be submitted asynchronously for SAP NetWeaver BW data sources. This allows queries on SAP NetWeaver BW to be cancelled. The parameter has no effect on any other data sources.</p> <p>default value: false</p>
MAX_ACTIVATION_LIMIT_FOR_PUSH_AGGREGATE_RULE	<p>This is the fraction of rows returned if the "Group-By" operator is pushed on the source compared to the initial cardinality without pushing "Group-By" on a source, for Data Federation to consider that the pushing of "Group-By" is useful. If you have to retrieve too many values, the pushing of "Group-By" becomes less useful. For example, if the parameter is set to = 80% (0.80), and the new cardinality due to pushing "Group-By" is more than 80% of the initial cardinality, "Group-By" is not pushed to the source. Raise this value if you want to push the "Group-By" operator more often. Lower this value if you want to push the "Group-By" operator less often.</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: 1.00</p>
MAX_BUFFER_SHARE_PER_OPERATOR	<p>The maximum share of maximum dynamic buffers. (A new value of this parameter takes effect when there are no queries registered in the BufferManager.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 80</p>

System Parameter	Description
MAX_CARDINALITY_FOR_HOP_STORE_IN_HASH_JOIN_OPERATOR	<p>This parameter defines the maximum cardinality for a store of a HOP algorithm.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 5000</p>
MAX_CONCURRENT_MEMORY_CONSUMING_OPERATORS	<p>The maximum number of memory-consuming concurrent operators. (A new value of this parameter should take effect when there are no queries registered in the BufferManager. Currently, you must restart the server)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 5</p>
MAX_CONCURRENT_MEMORY_CONSUMING_QUERIES	<p>The maximum number of parallel queries. (A new value of this parameter takes effect when there are no queries registered in the BufferManager)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 2</p>
MAX_CONJUNCTIONS	<p>Maximum number of conjunctions in a predicate.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 512</p>

System Parameter	Description
MAX_DECIMAL_PRECISION	<p>The maximum value that is reported by the data federation query server for the decimal precision of a column.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 40</p>
MAX_ESTIMATED_SIZE_FOR_STRINGS_OR_DECIMALS	<p>The maximum estimated size in bytes for a string or a decimal value. (A new value of this parameter takes effect on subsequent queries.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 512</p>
MAX_EXECUTIONS_PER_SUBQUERY_IN_HISTORY	<p>The maximum number of executions of a subquery kept in history.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 10</p>
MAX_NUMBER_OF_FRACTIONAL_DIGITS_FOR_TOSTRING_DOUBLE_IN_LOCALE	<p>The maximum number of fractional digits in the string representation of a double when using the locale-sensitive function toStringL(double, varchar).</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 20</p>
MAX_ORDERING_LIMIT_FOR_ORDER_JOINS_RULE	<p>Parameter for inference rule OrderJoinsRule. This is the maximum number of join orderings to produce.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 1</p>

System Parameter	Description
MAX_PARTITIONS_FOR_HASH_OPERATORS	<p>The maximum number of first level partitions to produce for the hash algorithms. (A new value of this parameter takes effect on subsequent queries.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 1987</p>
MAX_QUEUE_SIZE_LIMIT_FOR_ORDER_JOINS_RULE	<p>Parameter for inference rule OrderJoinsRule. This is the maximum size of the priority queue: It defines the maximum size of the search space.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 1024</p>
MAX_STRING_SIZE	<p>The maximum value that is reported by the data federation query engine for the string size of a column.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 255</p>
MAX_SUBQUERIES_IN_HISTORY	<p>The maximum number of subqueries per query kept in history.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 100</p>
MAX_TEMPORARY_TABLES	<p>Defines the maximum number of unique temporary tables generated by one connector.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 1000</p>

System Parameter	Description
MAX_THREADS_IN_UNION_OPERATOR	<p>The maximum number of active threads used by the operator UNION. (A new value of this parameter takes effect on subsequent queries.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 2</p>
MIN_ACTIVATION_THRESHOLD_FOR_SEMI_JOIN_RULE	<p>This parameter defines the fraction of tuples returned by a semi-join compared to a full table scan in order for the data federation service to consider that the semi-join is useful. If you have to retrieve too many values, the semi-join becomes less useful and the data federation service executes a table scan instead. For example, if you have a table with 10M rows, and you set the minimum activation threshold to 1000, then $10M / 1000 = 10\,000$. The data federation service uses the semi-join operator if it calculates that it will fetch less than 10 000 rows to execute the semi-join. Raise this value if you want to use semi-joins less often. Lower this value if you want to use semi-joins more often.</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: 100</p>
MIN_BUFFER_PAGES_PER_OPERATOR	<p>The minimum number of pages to return to an operator. (A new value of this parameter takes effect when there are no queries registered in the BufferManager.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 8</p>

System Parameter	Description
MIN_CARDINALITY_FOR_ASYNC_PREFETCH	<p>This parameter defines the minimum cardinality to determine an asynchronous prefetch. The value -1 means that no asynchronous prefetch is allowed</p> <p>type: long</p> <p>needs restart? no</p> <p>default value: 50000</p>
MIN_CARDINALITY_THRESH OLD_FOR_GROUP_BY_TRANSFORMATION_RULE	<p>The minimum cardinality of distinct values used to decide to eliminate GroupBy nodes by using order of sources <p>. The value 0 means that Group by elimination should always be done.</p> <p>type: long</p> <p>needs restart? no</p> <p>default value: 300</p>
MIN_DECIMAL_SCALE	<p>The minimum value that is reported by the data federation query engine for the decimal scale of a column.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 6</p>
MIN_SIZE_FOR_BUFFER_HASH_TABLE	<p>The minimum size of the buffer Hash Table used for Hash Join/Distinct (the secondary hash for each entry). (A new value of this parameter takes effect on subsequent queries.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 200</p>

System Parameter	Description
MIN_SOURCE_CARDINALITY_THRESH OLD_FOR_SEMI_JOIN_RULE	Specifies the cardinality threshold on the large table required to activate the semi-join operator. type: long needs restart? no default value: 15000
MIN_STORE_CARDINALITY_THRESH OLD_FOR_ORDER_BASED_JOIN_RULE	This parameter defines the minimum cardinality of store size that justifies the use of an ordered Merge Join. type: long needs restart? no default value: 10000
MIN_TRANSFER_CARDINALITY_THRESH OLD_FOR_MERGE_JOIN_RULE	This parameter defines the minimum cardinality of transfer that justifies the use of an ordered Merge Join. type: long needs restart? no default value: 30000
NUM_PARTITIONS_FOR_DISTINCT_OPERATOR	The optimal number of first level partitions to produce for the distinct operator. (A new value of this parameter takes effect when there are no queries registered in the BufferManager.) type: integer needs restart? no default value: 300

System Parameter	Description
NUMBER_OF_PARTITIONS_FOR_HASH_JOIN_OPERATOR	<p>The estimated optimal number of first level partitions for HashJoin/HashOuterJoin algorithms (A new value of this parameter takes effect on subsequent queries.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 300</p>
NUMBER_OF_PARTITIONS_FOR_MERGE_AGGREGATE_RULE	<p>The number of partitions to use in the MergeBasedGroupByAggregate algorithm. (A new value of this parameter takes effect on subsequent queries.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 300</p>
NUMBER_OF_PARTITIONS_FOR_ORDER_AGGREGATE_RULE	<p>The number of partitions to use in the OrderBasedGroupByAggregate algorithm (A new value of this parameter takes effect on subsequent queries.)</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 1987</p>
OPTIMIZER_COMPUTE_BINDINGS_PARAMETER	<p>Defines the parameter for ResolveBindings rule properties:</p> <ul style="list-style-type: none"> • The value 0 means deactivate the rule • The value 1 means resolve with bind join • The value 2 means resolve with cache node <p>type: integer</p> <p>needs restart? no</p> <p>default value: 2</p>

System Parameter	Description
QUERY_HISTORY_SIZE	<p>The maximum size of the history for the repository of executed queries.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 10</p>
SCALE_FOR_MAX_DECIMAL_PRECISION	<p>The value that is reported by the data federation query engine for the decimal scale of a column with maximum precision.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 6</p>
SEMI_JOIN_DIMENSION_CACHE_MEMORY_SIZE	<p>The amount of memory (Kb) allocated to one dimension cache for a semi-join.</p> <p>type: integer</p> <p>needs restart? no</p> <p>default value: 1024</p>

System Parameter	Description
SEMI_JOIN_EXECUTION_STRATEGIES	<p>Specifies the list of execution strategies for the semi-join operator in order of preference. The value is a combination of I, T, P separated by commas in order of preference. I stands for IN query execution strategy, T for temporary table execution strategy, and P for parameterized query execution strategy. The value NONE means that no execution strategy is supported by the wrapper. Empty string or null values are not supported.</p> <p>Examples:</p> <ul style="list-style-type: none">• T,P,I• I,T• P• NONE <p>If one of I, T, P is missing then the corresponding execution strategy is not supported by the wrapper.</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: I,T,P</p>

System Parameter	Description
THREADPOOL_ACTION_ON_OUT_OF_MEMORY	<p>Special parameter to set the action to be taken when an imminent Out of Memory condition is found by the Memory Tracker. Possible values are:</p> <ul style="list-style-type: none"> • freeze: Freezes all threads that execute managed queries. This lets you study the virtual machine state with a special external tool. • kill&freeze: Kills the managed queries currently running (their current thread is killed and queries are canceled and closed). This solution frees some more memory for the profiler to work, but may leave the server in an inconsistent state so that no more queries can run. After the queries are canceled, the server is frozen (no more managed queries can be run). • cancel running: Cancels all queries currently managed and for which there is currently an action executed in server. This lets you recover memory and keep the server running. • cancel all: Cancels all queries. Cancel may free up memory only if a currently running query is the cause of the problems and not an internal server error. • none <p>Note: A managed query is any query sent from a ThinDriver or remote server connection. The Administration console and Simple text consoles don't use managed queries and so they are not explicitly frozen.</p> <p>type: string</p> <p>needs restart? no</p> <p>default value: cancel all</p>

6.6 List of session parameters

Session parameter	Description
CATALOG	Defines the current catalog, used if no catalog is given in a query.
SCHEMA	Defines the current schema, used if no schema is given in a query.
COMP	Defines the collation used for comparing strings. It is used to define how strings will be compared in SQL queries. The value of this parameter is either one of the supported collation values, or the keyword LINGUISTIC: In this case the collation used is the collation defined by the parameter SORT. The default is BINARY. The default value can be changed with the system parameter DEFAULT_COMP.
SORT	Defines the collation used for sorting strings. It is used to define how strings will be sorted in SQL queries. The value of this parameter is one of the supported collation values. The default is BINARY. The default value can be changed with the system parameter DEFAULT_SORT.
LOCALE	Defines the ISO code for the locale. The default is en_US. The default value can be changed with the system parameter DEFAULT_LOCALE.
DATA_LOCALE	Defines the locale to use for data. This parameter is used by the connectors that can return localized data (currently SAP NetWeaver BW connector).

Related Topics

- [Collation in the data federation application](#)
- [List of system parameters](#)

6.7 Collation in the data federation application

Collation is a set of rules that determines how data is sorted and compared.

The data federation application and the database systems that it accesses sort and compare character data using rules that define the correct sequence of characters. For most database systems, you can configure options to specify whether the database system should consider upper or lower case, accent marks, character width or types of kana characters.

Case sensitivity

If a system treats the character M the same as the character m, then the system is case-insensitive. A computer treats M and m differently because it uses ASCII codes to differentiate the input. The ASCII value of M is 77, while m is 109.

Accent sensitivity

If a system treats the character `a` the same as the character `á`, then the system is accent-insensitive. A computer treats `a` and `á` differently because it uses ASCII codes for differentiating the input. The ASCII value of `a` is 97 and `á` is 225.

Kana Sensitivity

When Japanese kana characters Hiragana and Katakana are treated differently, it is called Kana sensitive.

Width sensitivity

When a single-byte character (half-width) and the same character when represented as a double-byte character (full-width) are treated differently then it is width sensitive

Related Topics

- [Supported Collations in the data federation application](#)
- [How the data federation application decides how to push queries to sources when using binary collation](#)

6.7.1 Supported Collations in the data federation application

The following collations are supported in DF:

binary

Unicode binary sort (or compatible with Unicode binary sort - for example, sort on ASCII charset is compatible with sort on Unicode charset)

locale_AI_CI

locale, accent insensitive, case insensitive

locale_AI_CS

locale, accent insensitive, case sensitive

locale_AS_CI

locale, accent sensitive, case insensitive

locale_AS_CI

locale, accent sensitive, case insensitive

locale_AS_CS

locale, accent sensitive, case sensitive

where locale is defined as `LN_CY` with

- `LN` - ISO language code (for example, `en`)
- `CY` - ISO country code (for example, `US`)

Note:

All the DF collations are kana-insensitive and width-insensitive

Example:

en_US_AS_CI - English, US, accent sensitive, case insensitive

Related Topics

- [Collation in the data federation application](#)

6.7.2 How the data federation application decides how to “push” queries to sources when using binary collation

The optimizer in the data federation query engine performs pushdown analysis to decide if an SQL operation can be “pushed” down to a data source.

When collations are binary, the query engine decides whether or not to “push” a subquery on a particular datasource by examining only the SQL capabilities of the source of data.

Thus, in the general case, the query engine assumes that the underlying source of data is using a default collation that is compliant with the binary collation in the data federation application.

For SQLServer, MySQL and Oracle only, it is possible to force the data federation query engine to use binary collation even if the default collation on the source is not compliant with binary collation. (see MySQL, SQLserver, Oracle for details of how to configure resource parameters for binary collation).

Related Topics

- [Collation in the data federation application](#)
- [Setting string sorting and string comparison behavior for data federation SQL queries](#)
- [Supported Collations in the data federation application](#)

6.7.3 Setting string sorting and string comparison behavior for data federation SQL queries

You can use the sort and comp parameters to set how the data federation query engine treats sorting and comparison for strings.

The sort parameter is used to define how strings will be sorted by the data federation query engine. The value of the sort parameter is one of the supported collation values. The default is binary.

The comp parameter is used to define how strings will be compared in SQL queries . The value of the comp parameter is either

- one of the supported collation values
- the keyword Linguistic: In this case the collation used is the collation defined by the sort parameter.

The sort and comp parameters can be defined as a session parameter, system parameter or as a property of a user account.

- If the sort or comp parameter is defined in session parameters, this value will be used for the current connection.
- If not defined in session parameters, the sort or comp property of the user account will be used for current connection.
- If not defined as a property of the current user account, the sort or comp system parameter will be used for current connection.

The values for sort and comp parameters have an impact on the result of SQL operations applied on string values. An operation might be a function, a SQL operator such as GROUP BY, ORDER BY or a filter expression such as $T.A < e$. The figure below summarizes the SQL operators which are sensitive to comp and sort parameters:

SQL Expressions	Sensitivity
=, !=, >, <=, >=	comp sensitive
BETWEEN, NOTBETWEEN	comp sensitive
CASE	comp sensitive
DISTINCT	comp sensitive
GROUP BY	comp sensitive
HAVING	comp sensitive
IN, NOTIN	comp sensitive
LIKE, NOTLIKE	insensitive: binary only
ORDER BY	sort sensitive
UNION ALL	insensitive

SQL Functions	Sensitivity
MAX, MIN	comp sensitive
data federation string functions	insensitive: binary only

Example:

```
SELECT LASTNAME, count(*)
FROM EMPLOYEE E
WHERE SALARY < 5000 AND DEPARTMENT_NAME =
      Sales
GROUP BY LASTNAME
```

Table 6-4: Employee table

LASTNAME	FIRSTNAME	SALARY	DEPARTMENT_NAME
Smith	John	6000	Sales
Sm Ith	Jo	4000	Sales
Smith	John	2000	Sa Les
Smith	Albert	7000	Sales

When the comp parameter is: *en_US_AS_CS*, the result is:

Smith	3
Sm Ith	1

When the comp parameter is: *en_US_AI_CI*, the result is:

Smith	4
-------	---

SQL syntax reference

7.1 The query language for the data federation query engine

Whenever possible, the data federation application aligns to the standard SQL-92 syntax. It is important to understand, however, how some elements are used by, or influence, statements in the data federation query engine. This section describes elements of SQL-92 implemented by the data federation application including object management, data types, select and expressions.

7.1.1 Identifiers and naming conventions

You refer to tables by giving the catalog and schema containing the table. The catalog, schema and table must be separated by dots (.).

Example: Defining the name of a table

A qualified name must be used to reference a table. It is composed of catalog name, schema name and the table name.

- `c.s.t`
- `"c"."s"."t"`

If there is a catalog or schema defined by default, you can omit the name of the catalog or schema in the reference to the table.

Catalogs

A catalog is a named group of schemas. The catalog's name qualifies the names of schemas that belong to it. You either state the catalog name explicitly in the query, or set a default catalog.

Schemas

An SQL schema is a named group of tables or views. Schemas are dependent on a catalog. The schema name must be unique within the catalog to which it belongs.

Schema identifiers are absolute paths when no default catalog is set, or a relative path from the default catalog directory:

A default schema can be set via session parameters in the data federation administration tool.

Tables

A table is attached to one schema. The table name must be unique within a schema to which it belongs.

A table must be identified by: a catalog name, a schema name and the table name. In standard SQL syntax, the table identifier is constructed by concatenating the catalog name, the schema name and the table name separated by a . (period).

When default catalog and/or default schemas are set, catalog names and schema names can be omitted in the table identifier.

Columns

A table is described by a set of columns. A column name must be unique within a table to which it belongs. In standard SQL syntax, the column identifier is constructed by concatenating the table identifier, with the column name separated by a period ".".

Default catalogs and schemas

You can specify a default catalog or schema via the session parameters in the data federation administration tool. Specifying a default catalog allows you to send queries without fully qualifying table names.

To reference the table	if the default catalog is	and if the default schema is	then use the qualified name
c.s.t	c		s.t
"c1".s.t	"c1"		s.t
c.s.t	c	s	t

Using double quoted delimiters

To avoid misinterpretation of identifiers by the parser, you must use double quoted delimiters for catalog, schema, table and column names when those names contain non alpha-numeric characters.

Correct	"c1/c2"."sche+ma"."Tab-le1".col1
Incorrect	/c1/c2.sche+ma.Tab-le1.col1

Related Topics

- [Changing a system parameter using the data federation administration tool](#)
- [Expressions](#)

7.1.2 Data types used in the data federation query engine

In the data federation query engine, each column, local variable, expression, and parameter has a related data type. A data type is a definition of the size and structure of data that the object can hold, such as: integer data, character data, date and time data or decimal data.

A data type that is associated to an object defines three attributes of the object:

- data type: the kind of data contained by the object
- length and size: the length or size of the value
- scale and precision: the scale and precision of the number (numeric data types only)

In a traditional database, length, precision and scale are set when creating a column since they define the properties of the stored value. The data federation query engine is a virtual database and does not store any values. Thus length, precision and scale are not defined at schema definition time. Their values are dynamically inferred from the contributing source tables.

Known data types

The data federation query engine supports the standard SQL types defined in `java.sql.Types`. Here is a list of the data types that are supported:

- BIT
- DATE
- TIMESTAMP
- TIME
- INTEGER
- DOUBLE
- DECIMAL
- VARCHAR
- NULL

Since all databases do not use the same data types or interpret them in the same way, the query engine has standardized a mapping between the common database types and the query engine.

Mapping data federation query engine types to JDBC data types

The following table details the correspondence between the internal data types used in the data federation query engine and the JDBC data types returned by the data federation JDBC driver.

Data federation data type	JDBC data type
BIT	BIT
DATE	DATE
TIMESTAMP	TIMESTAMP
TIME	TIME
INTEGER	INTEGER
DOUBLE	DOUBLE
DECIMAL	DECIMAL
VARCHAR	VARCHAR
NULL	NULL

Mapping from JDBC data types to data federation data types

When accessing a JDBC data source, the data federation query engine does the mapping of the JDBC types returned by the JDBC driver to the internal data federation application data types. The following table details the correspondence between the JDBC data types and the data federation type that is used for the mapping.

JDBC data type	Data federation data type
TINYINT, SMALLINT, INTEGER, DECIMAL with precision ≤ 10 and scale = 0	INTEGER
BIT	BIT
REAL, FLOAT, DOUBLE	DOUBLE
BIGINT, DECIMAL, NUMERIC	DECIMAL

JDBC data type	Data federation data type
VARCHAR, LONGVARCHAR, CHAR	VARCHAR
DATE	DATE
TIME	TIME
TIMESTAMP	TIMESTAMP
NULL and all other JDBC types	NULL

Date and time conversion

The data federation query engine converts TIME data into a TIMESTAMP data by setting the date to '1970-01-01'

For example, the conversion of a time into a timestamp:

TIME '12:01:01' is converted to TIMESTAMP '1970-01-01 12:01:01.0'

The data federation query engine converts DATE data to TIMESTAMP by adding the time: 00:00:00.000000000.

For example, the conversion of a date to a timestamp:

DATE '1999-01-01' is converted to TIMESTAMP '1999-01-01 00:00:00.000000000'

Type inference in expressions

When two expressions have different data types, the data type of the result of an expression that combines these two expressions with an arithmetic operator is determined by applying precedence of data types.

The data federation query engine uses the following precedence order between types:

```

NULL
VARCHAR
INTEGER
DOUBLE
DECIMAL

```

Scale and precision

The length, scale and precision of the result of an expression is inferred from the type of the result. In the case of VARCHAR or DECIMAL result types, the length, scale and precision are inferred from the scale and precision of the input expressions and the function and operator that combined them.

The table below gives the vector (length, precision, scale) for all data federation expressions.

Column type	Fixed limit (length, precision, scale)
BIT	(1, 1, 0)
INTEGER	(11, 10, 0)
DOUBLE	(22, 15, 0)
DATE	(10, 0, 0)
TIMESTAMP	(29, 9, 0)
TIME	(8, 0, 0)
NULL	(0, 0, 0)
DECIMAL	Inferred
VARCHAR	Precision and scale are always (0, 0) Length is Inferred

7.1.3 Statements

You can write SQL queries to retrieve or manipulate data that is stored on the data federation query engine. A query can be issued in several forms:

- The data federation administration tool, a graphical user interface (GUI) on top of the data federation query engine.
- a command line SQL application
- another compatible utility that can issue a SELECT statement
- a client or middle tier-based application, such as a Microsoft Visual Basic application; These can map the data from an SQL Server table into a bound control, such as a grid.

7.1.3.1 SELECT Statement

Although queries have various ways of interacting with a user, they all accomplish the same task: They present the result set of a SELECT statement to the user.

The SELECT statement retrieves data from the data federation query engine and returns it to the user in one or more result sets. A result set is a tabular arrangement of the data from the SELECT. Like an SQL table, the result set is made up of columns and rows.

The full syntax of the SELECT statement is complex, but most SELECT statements describe four primary properties of a result set:

- the number and attributes of the columns in the result set
- the names of the tables which provide the data
- the conditions that the rows in the source tables must meet to qualify for the SELECT; Rows that do not meet the conditions are ignored.
- the sequence in which the rows of the result set are ordered

Example: **SELECT** statement

The following SELECT statement finds the product ID, name, and list price of any products whose unit price exceeds \$40.

```
SELECT ProductID, Name, ListPrice
FROM Production.Product
WHERE ListPrice > $40
ORDER BY ListPrice ASC
```

- **SELECT** clause

The column names listed after the SELECT keyword (*ProductID*, *Name*, and *ListPrice*) form the select list. This list specifies that the result set has three columns, and each column has the name, data type, and size of the associated column in the table that is given in the FROM clause (the *Product* table). Because the FROM clause specifies only one table, all column names in the SELECT statement refer to columns in that

- **FROM** clause

The FROM clause lists the *Product* table as the one table from which the data is to be retrieved.

- **WHERE** clause

The WHERE clause specifies the condition that the only rows in the *Product* table that qualify for this SELECT statement are those rows in which the value of the *ListPrice* column is more than \$40.

- **ORDER BY** clause

The ORDER BY clause specifies that the result set is to be sorted in ascending sequence (ASC) based on the value in the *ListPrice* column.

7.1.3.2 SQL-92 statements supported by the data federation query engine

The data federation query engine supports the Data Manipulation language (DML) and a list of procedures and commands. A particular set of SELECT statements are supported and unless noted, the entire standard SQL-92 syntax. In particular both the SQL-92 grammar for outerjoin and JDBC syntax for outerjoins is supported.

Related Topics

- [Grammar for the SELECT clause](#)

7.1.4 Expressions

This section details the expressions in data federation SQL syntax.

Functions in expressions

To see the latest list of functions, see *SAP BusinessObjects SQL function reference for multisource-enabled universes* in the book *Information Design Tool User Guide*.

Operators in expressions

Operators in expressions combine one or more simple expressions to form a more complex expression.

Operator name	Description
+ (Addition)	An arithmetic operator that means "Addition" for numeric types and "Concatenation" for VARCHAR type.
- (Subtract)	An arithmetic operator that means "Subtraction".
* (Multiply)	An arithmetic operator that means "Multiplication".

Operator name	Description
/ (Divide)	An arithmetic operator that means "Division".
% (Modulo)	An arithmetic operator. It returns the integer remainder of a division. For example, $12 \% 5 = 2$ because the remainder of 12 divided by 5 is 2.
** (Power)	An arithmetic operator. It returns the value of the given expression to the specified power.
= (Equals)	A comparison operator that means "Equal to".
> (Greater)	A comparison operator that means "Greater than".
< (Less)	A comparison operator that means "Less than"
>= (Greater Than Or Equal To)	A comparison operator that means "Greater than or equal to".
<= (Less Than Or Equal To)	A comparison operator that means "Less than or equal to"
<> (Not Equal To)	Comparison operator meaning "Not equal to".
ALL	A logical operator that is TRUE if all of a set of comparisons are TRUE.
AND	A logical operator that is TRUE if both BOOLEAN expressions are TRUE.
ANY	A logical operator that is TRUE if any one of a set of comparisons are TRUE.
BETWEEN	A logical operator that is TRUE if the operand is within a range.

Operator name	Description
EXISTS	A logical operator that is TRUE if a subquery contains any rows.
IN	A logical operator that is TRUE if the operand is equal to one of a list of expressions.
LIKE	A logical operator that is TRUE if the operand matches a pattern.
NOT	A logical operator that inverts the value of any other BOOLEAN operator.
OR	A logical operator that is TRUE if either BOOLEAN expression is TRUE.
SOME	A logical operator that is TRUE if some of a set of comparisons are TRUE.
+ (Positive)	A unary operator where the numeric value is positive.
- (Negative)	A unary operator where the numeric value is negative.

Operator precedence levels

When a complex expression has multiple operators, operator precedence determines the sequence in which the operations are performed. The order of execution can significantly affect the resulting value.

Operators have these precedence levels. An operator on a higher level is evaluated before an operator on a lower level:

- + (Positive), - (Negative)
- * (Multiply), / (Division), % (Modulo), ** (Power)
- + (Add), (+ Concatenate), - (Subtract)
- =, >, <, >=, <=, <> (Comparison operators)
- NOT
- AND

- OR
- ALL, ANY, BETWEEN, IN, LIKE, SOME

Object identifiers and numeric constants

Names of identifiers and constants must start with a letter and use only letters and underscores. You can, however, use any characters in your identifier / constant name if you enclose it in double-quotes: `"`.

For example, identifier names could be `ABC_12` or `"!%any name you like$#%"`.

The following table describes the data federation syntax for identifiers and numeric constants:

To type a	Use this definition	For example
Integer	INTEGER: nnn (only digits - one or more)	12 14 15
Double or Decimal	DOUBLE/ DECIMAL: nn.nn (one or more digits, followed by a point, followed by one or more digits)	12.3 13.222 11.3
Date	DATE: {d 'yyyy-mm-dd'}	{d '2005-03-28'}
Time	TIME: {t 'hh:mm:ss'}	{t '01:10:12'}
Timestamp	TIMESTAMP: {ts 'yyyy-mm-dd hh:mm:ss.ffff'}	{ts '2005-03-28 01:11:34.23222'}
String or Varchar	any string inside single quotes	'asdasdas'
Simple identifier	any string starting with a letter and followed by any combination of letters, digits and underscores	ABC_12
Identifier with special characters	any string inside double-quotes	"!%any name you like\$#%"

7.1.5 Comments

To add comments to the SQL Statements precede the text with a double hyphen (--), or with a pound sign (#). Comments terminate at the end of the line.

7.2 Grammar for the SELECT clause

The following section details the complete SQL Select clause grammar used with the data federation query engine.

```

start      := ( query ) ( ";" )? <EOF>

query      := ( <WITH> withListElement ( "," withListElement ) * )?
              SQLSelectFromWhere (
                ( <UNION> | <INTERSECT> | <EXCEPT> ) ( <DISTINCT> | <ALL> )?
                SQLSelectFromWhere QueryExpression )?
                ( <ORDER> <BY> orderByTerms ( "," orderByTerms ) * )?

QueryExpression :=
  ( ( <UNION> | <INTERSECT> | <EXCEPT> ) ( <DISTINCT> | <ALL> )?
    SQLSelectFromWhere ) *

withListElement := anyIdentifier <AS> ( WITHView | nativeQuery )

WITHView      := "(" query ")"

nativeQuery := <NATIVE> "("
              dataSourceIdentifier ","
              nativeQueryStatement ","
              columnSpecificationList
              ( "," paramSpecificationList )? ")"

dataSourceIdentifier := anyIdentifier

nativeQueryStatement := quotedString

columnSpecificationList := columnSpecification ( "," ( columnSpecification ) ) *

paramSpecificationList := paramSpecification ( "," ( paramSpecification ) ) *

columnSpecification := anyIdentifier columnDataType

paramSpecification := ( ( ( <DATE_LITERAL> | <TIME_LITERAL> | <TIMESTAMP_LITERAL> ) )
                        | quotedString ) columnDataType
                    | <NULL_LITERAL>

columnDataType := identifier ( "(" integerLiteral ( "," integerLiteral )? ")" )?

integerLiteral := <INT_LITERAL>

SQLSelectFromWhere :=
  <SELECT> ( <DISTINCT> )? ( selectExpression ( "," selectExpression ) * | ( <MULT> ) )
  ( fromClause
    ( <WHERE> disjunction )?
    ( <GROUP> <BY> ( additiveTerm ) ( "," additiveTerm ) * )?
    ( <HAVING> disjunction )? )

fromClause := ( <FROM> tableReferenceList )

tableReferenceList := ( tableReference ( "," tableReference ) * )

```

```

tableReference := tableReferenceAtomicTerm ( qualifiedJoinPart ) *
tableReferenceAtomicTerm := ( tablePrimary )
    | jdbcOuterJoin
    | "(" query ")" ( ( <AS> )? ( identifier | delimitedIdentifier ) )?
    | "(" tableReference ")" ( ( <AS> )?
        identifier ( "(" projectAlias ( "," projectAlias ) * ")" )? )?
tablePrimary := ( table ( ( <AS> )? ( tableAlias ) )? )
table := ( anyIdentifier ( "." anyIdentifier ( "." anyIdentifier )? )? )
qualifiedJoinPart := ( ( <NATURAL> )? ( joinType )?
    <JOIN> tableReferenceAtomicTerm ( joinSpecification )? )
jdbcOuterJoin := "{" <OUTER_JOIN_JDBC> jdbcOuterJoinPart "}"
jdbcOuterJoinPart := tableReferenceAtomicTerm
    ( outerJoinType <OUTER> <JOIN> ( jdbcOuterJoinPart ) joinSpecification )?
joinType := ( ( <INNER> ) | ( <CROSS> ) | ( outerJoinType ( <OUTER> )? ) )
outerJoinType := ( <LEFT> | <RIGHT> | <FULL> )
joinSpecification := ( joinCondition | namedColumnsJoin )
joinCondition := ( <ON> disjunction )
namedColumnsJoin := ( <USING> "(" addUsing ( "," addUsing ) * ")" )
addUsing := columnName
projectAlias := ( anyIdentifier )
selectExpression := ( ( tableStar )
    | ( disjunction ( ( <AS> )? anyIdentifier )? ) )
tableStar := table "." <MULT>
functionTermJdbc := ( "{" <FUNCTION_JDBC> (
    ( identifier )
    | ( <LEFT> )
    | ( <RIGHT> ) ) "(" ( disjunction ( "," disjunction ) * )? ")" "}" )
functionTerm := ( (
    ( identifier ) |
    ( <LEFT> )
    | ( <RIGHT> ) )
    "(" ( ( <DISTINCT> | <ALL> )?
        ( disjunction ( "," disjunction ) * | <MULT> ) )? ")" )
analyticFunctionPart := ( <OVER> "("
    ( <PARTITION> <BY> ( variable ) ( "," variable ) * )?
    <ORDER> <BY> ( ( variable ( <ASC> | <DESC> )? ) )
    ( "," ( variable ( <ASC> | <DESC> )? ) ) * ")" )
disjunction := ( conjunction ( <OR> conjunction ) * )
conjunction := ( negationTerm ( <AND> negationTerm ) * )
escapeChar := quotedString
quotedString := <QUOTED_STRING_LITERAL>
anyIdentifier := <IDENTIFIER>
    | <DELIMITED_IDENTIFIER>
delimitedIdentifier := <DELIMITED_IDENTIFIER>
identifier := <IDENTIFIER>
columnName := anyIdentifier
negationTerm := ( <NOT> )? ( ( comparisonTerm ) | ( <EXISTS> "(" query ")" ) )
comparisonTerm := additiveTerm ( <COMPARISON_OPERATOR> (
    ( additiveTerm )
    | ( ( ( <ANY> ) | ( <SOME> ) | ( <ALL> ) ) "(" query ")" ) )
    | ( <BETWEEN> additiveTerm <AND> additiveTerm )
    | ( inValuesOrQuery )

```

```

| <LIKE> additiveTerm ( <ESCAPE> escapeChar )?
| <IS> ( <NULL_LITERAL> | <NOT> <NULL_LITERAL> )
| <NOT> (
    <BETWEEN> additiveTerm <AND> additiveTerm
    | <LIKE> additiveTerm ( <ESCAPE> escapeChar )? ) )?

nativeExpression    := <NATIVE> <EXPRESSION> "("
    dataSourceIdentifier ","
    columnDataType ","
    quotedString bindingArgumentList ")"

bindingArgumentList := ( "," additiveTerm ) *

inValuesOrQuery := ( ( <NOT> )? <IN> "(" ( ( inValues ) | ( query ) ) ")" )

inValues    := ( signedConstant ( "," signedConstant ) * )

additiveTerm := ( factor ( ( <PLUS> | <MINUS> ) factor ) * )

factor := unaryTerm ( (
    <MULT>
    | <DIVIDE>
    | <POWER>
    | <INT_DIVIDE>
    | <MOD> ) unaryTerm ) *

unaryTerm := atomicTerm
    | <PLUS> atomicTerm
    | <MINUS> atomicTerm

variable := ( anyIdentifier
    ( "." anyIdentifier
        ( "." anyIdentifier
            ( "." anyIdentifier )? )? )? )

variableFullName := anyIdentifier (
    "." anyIdentifier
    ( "." anyIdentifier
        ( "." anyIdentifier )? )? )?

constant := <BOOL_LITERAL>
    | <INT_LITERAL>
    | <FLOAT_LITERAL>
    | <SCIENTIFIC_NOTATION_LITERAL>
    | <DATE_LITERAL>
    | <TIMESTAMP_LITERAL>
    | <TIME_LITERAL>
    | <NULL_LITERAL>
    | quotedString
    | <PARAMETER>

signedConstant := <BOOL_LITERAL>
    | ( <PLUS> | <MINUS> )? ( <INT_LITERAL> | <FLOAT_LITERAL> )
    | <SCIENTIFIC_NOTATION_LITERAL>
    | <DATE_LITERAL>
    | <TIMESTAMP_LITERAL>
    | <TIME_LITERAL>
    | <NULL_LITERAL>
    | quotedString
    | <PARAMETER>

atomicTerm := functionTerm ( analyticFunctionPart )?
    | functionTermJdbc
    | variable
    | constant
    | "(" disjunction ")"
    | caseExpression
    | coalesceExpression
    | castExpression
    | convertFunction
    | nativeExpression

caseExpression := ( <CASE> ( ( additiveTerm ( (
    <WHEN> additiveTerm <THEN> additiveTerm ) + )
    | ( ( <WHEN> disjunction <THEN> additiveTerm ) + ) )
    ( <ELSE> additiveTerm )? <END> )

coalesceExpression := ( <COALESCE> "(" additiveTerm ( ( "," additiveTerm ) + ")" )

castExpression := ( <CAST> "(" disjunction <AS> identifier ")" )

```

```

convertFunction := ( <CONVERT> "(" disjunction "," identifier ")" )
tableAlias      := ( delimitedIdentifier | identifier )
orderByTerms    := ( variableFullName | integerLiteral ) ( <ASC> | <DESC> )?
bindingFunction := ( variable <COMPARISON_OPERATOR> additiveTerm )
startStoredProcedure := ( procedureCall ) ( ";" )? <EOF>
procedureCall   := <CALL> anyIdentifier ( ( "(" procedureArguments ")" )
      | ( procedureArguments ) )
procedureArguments := ( procedureArgument ( "," procedureArgument ) * )?
procedureArgument := ( procedureConstant )
      | ( <CAST> "(" procedureConstant <AS> identifier ")" )
procedureConstant := (
  <BOOL_LITERAL>
  | <INT_LITERAL>
  | <FLOAT_LITERAL>
  | <SCIENTIFIC_NOTATION_LITERAL>
  | <DATE_LITERAL>
  | <TIMESTAMP_LITERAL>
  | <TIME_LITERAL>
  | <NULL_LITERAL>
  | quotedString
  | <PARAMETER> )

<DEFAULT> TOKEN [IGNORE_CASE] : {
  <FROM: "from">
  | <SELECT: "select">
  | <DISTINCT: "distinct">
  | <WHERE: "where">
  | <GROUP: "group">
  | <ORDER: "order">
  | <BY: "by">
  | <HAVING: "having">
  | <DESC: "desc">
  | <ASC: "asc">
  | <AS: "as">
  | <UNION: "union">
  | <INTERSECT: "intersect">
  | <EXCEPT: "except">
  | <WITH: "with">
  | <USING: "using">
  | <ON: "on">
  | <MERGE: "merge">
  | <MERGING: "merging">
  | <NATIVE: "native">
  | <EXPRESSION: "expression">
  | <NATURAL: "natural">
  | <JOIN: "join">
  | <CROSS: "cross">
  | <INNER: "inner">
  | <OUTER: "outer">
  | <LEFT: "left">
  | <RIGHT: "right">
  | <FULL: "full">
  | <ESCAPE: "escape">
  | <OUTER_JOIN_JDBC: "oj">
  | <FUNCTION_JDBC: "fn">
  | <OVER: "over">
  | <PARTITION: "partition">
  | <CASE: "case">
  | <WHEN: "when">
  | <THEN: "then">
  | <ELSE: "else">
  | <END: "end">
  | <COALESCE: "coalesce">
  | <CALL: "call">
  | <CAST: "cast">
  | <CONVERT: "convert">
}

<DEFAULT> TOKEN [IGNORE_CASE] : {
  <NULL_LITERAL: "null">
}

```

```

<DEFAULT> TOKEN [IGNORE_CASE] : {
    <BOOL_LITERAL: "true" | "false">
}

<DEFAULT> TOKEN [IGNORE_CASE] : {
<AND: "and">
| <OR: "or">
| <IN: "in">
| <ANY: "any">
| <SOME: "some">
| <ALL: "all">
| <EXISTS: "exists">
| <BETWEEN: "between">
| <COMPARISON_OPERATOR: ">" | ">=" | "<" | "<=" | "=" | "<>">
| <LIKE: "like">
| <NOT: "not">
| <MULT: "*">
| <PLUS: "+">
| <MINUS: "-">
| <DIVIDE: "/">
| <INT_DIVIDE: "//">
| <POWER: "**">
| <MOD: "%">
| <IS: "is">
| <PARAMETER: "?">
}

<DEFAULT> SPECIAL : {
    <SINGLE_LINE_COMMENT: ("#" | "--") (~["\n", "\r"])* (" \n" | " \r" | " \r\n")*>
}

<DEFAULT> TOKEN : {
<INT_LITERAL: ("0"-"9")+>

| <FLOAT_LITERAL: ("0"-"9")+ "." ("0"-"9")+>

| <SCIENTIFIC_NOTATION_LITERAL: ("-" | "+")? ((["0"-"9"])+ ("." ("0"-"9"))?)
  | ("." ("0"-"9"))+ ("e" | "E") ("-" | "+")? (["0"-"9"])+>

| <DATE_LITERAL: "{" (" ")* "d" (" ")* "\" <DIGIT> <DIGIT> <DIGIT> <DIGIT>
  "-" <DIGIT> <DIGIT> "-" <DIGIT> <DIGIT> "\" (" ")* "}">

| <TIME_LITERAL: "{" (" ")* "t" (" ")* "\" <DIGIT> <DIGIT>
  ":" <DIGIT> <DIGIT> ":" <DIGIT> <DIGIT> "\" (" ")* "}">

| <TIMESTAMP_LITERAL: "{" (" ")* "ts" (" ")* "\"
  <DIGIT> <DIGIT> <DIGIT> <DIGIT> "-" <DIGIT> <DIGIT> "-" <DIGIT> <DIGIT> " "
  <DIGIT> <DIGIT> ":" <DIGIT> <DIGIT> ":" <DIGIT> <DIGIT>
  ("." (<DIGIT>)*)? "\" (" ")* "}">

| <DELIMITED_IDENTIFIER: "\" (~["\"", "\n", "\r"] | "\"\"")* "\">

| <QUOTED_STRING_LITERAL: "\"" (~["'"] | "\"'\"")* "\">

| <IDENTIFIER: <LETTER> (<LETTER> | <DIGIT>)*>

| <#URLCHAR: [":", "?", ".", "/", "@", "_", "-", "+", "%", "!"]>

| <#LETTER: ["$", "A"-"Z", " ", "a"-"z",
  "\u00c0"-" \u00d6", "\u00d8"-" \u00f6", "\u00f8"-" \u00ff", "\u0100"-" \u024f",
  "\u0370"-" \u052f", "\u0530"-" \u05ff", "\u0600"-" \u06ff", "\u0900"-" \u10ff",
  "\u1100"-" \u11ff", "\u1e00"-" \u1eff", "\u0100"-" \u1fff", "\u3040"-" \u319f",
  "\u3200"-" \u32fe", "\u3300"-" \u33fe", "\u3400"-" \u3d2d", "\u4e00"-" \u9fff",
  "\uac00"-" \ud7a3", "\uf900"-" \ufa2d", "\ufb00"-" \ufb4f", "\ufb50"-" \ufdbf",
  "\ufe70"-" \ufefc", "\uff00"-" \uffff"]>

| <#DIGIT: ["0"-"9", "\u0660"-" \u0669", "\u06f0"-" \u06f9", "\u0966"-" \u096f",
  "\u09e6"-" \u09ef", "\u0a66"-" \u0a6f", "\u0ae6"-" \u0aef", "\u0b66"-" \u0b6f",
  "\u0be7"-" \u0bef", "\u0c66"-" \u0c6f", "\u0ce6"-" \u0cef", "\u0d66"-" \u0d6f",
  "\u0e50"-" \u0e59", "\u0ed0"-" \u0ed9", "\u1040"-" \u1049"]>
}

```

Glossary

8.1 Terms and descriptions

This section lists the terms used in the data federation application and documentation.

Term or phrase	Definition
connector	a driver that allows the data federation query engine to connect to a source of data
“fanout”	In a relationship between columns, the average number of entries in the second column that are related to each entry in the first column.
“merge join”	An operation used in data federation where two large tables of data are sorted before they are joined, in order to reduce the time it takes to join the tables.
“push” (verb)	To request a source database to perform some operation (instead of performing the operation within the data federation engine; performing operations on source databases is generally more efficient than in the data federation engine).
“semi-join”	An operation between two tables that returns the rows of the first table that match at least one row in the second table. In other words, the first table is filtered with the rows of the second table.

Term or phrase	Definition
"statistics"	Numeric information about data stored in sources that are used for data federation, including such things as estimated number of entries in a table, estimated number of distinct values in a column, or average number of relationships between each value in a column and another column.

Troubleshooting

9.1 About logging of the data federation service

The data federation service is hosted by an Adaptive Processing Server on the SAP BusinessObjects Enterprise platform.

You can find the logs for your data federation service in the Adaptive Processing Server that is hosting the service.

See the documentation about logging for SAP BusinessObjects Enterprise servers in the *SAP BusinessObjects Enterprise Administrator's Guide*.

9.2 For SAP NetWeaver BW datasources, long-running queries cause the connection to close

When running queries that take over 10 minutes on SAP NetWeaver BW datasources, the connection is closed without a message.

Cause

The default timeout value on SAP NetWeaver BW is too short to run the query.

Action

1. Increase the timeout value as follows.
2. Log on to the SAP system.
3. Enter rz11 in the transaction text field and execute it.
4. Display the parameter rdisp/max_wprun_time.
5. Click **Change Value** and set the parameter to a value greater than 600 to allow your reports to run.

The value is set in seconds.

9.3 For SAP NetWeaver BW connector, error NoClassDefFoundError: CpdcDriver

I get the exception: `NoClassDefFoundError: com.sap.conn.rfc.driver.CpdcDriver`.

Cause

You may get this exception because a dependency of SAP Java Connector (JCo) is not installed on your host. JCo is the middleware used by the data federation query engine to connect to SAP NetWeaver. The missing dependency is the set of Microsoft Visual Studio 2005 C/C++ runtime libraries.

Action

Install the Microsoft Visual Studio 2005 C/C++ runtime libraries. For details, see <https://service.sap.com/sap/support/notes/684106>.

Index

C

- callback
 - SAP NetWeaver BW 68
- cancelling
 - queries 14, 15
- capabilities
 - list of 75
- comments 114
- connections
 - SAP NetWeaver BW 70
- connector properties 43, 47, 48, 49, 54, 63
- CpicDriver error
 - SAP NetWeaver BW 122

D

- datasources
 - SAP NetWeaver BW 121

M

- memory
 - optimizing 28
- merge join
 - optimizing 37

O

- optimization
 - using statistics 31
 - using statistics to choose best query algorithms 29
- optimizing
 - memory 28
 - queries
 - merge join 37
 - semi-join 35
 - using order-based operators 38, 39
 - SAP NetWeaver BW 40
 - using order-based operators 37, 38, 39
- order-based operators 37, 38, 39

P

- privileges
 - in SQL statements 108

Q

- queries
 - cancelling 14, 15
 - SAP NetWeaver BW
 - parallel 40

- query optimization
 - large tables 35, 37
 - merge join 37
 - parameters 35, 37
 - semi-join 35

S

- SAP Business Explorer 70
- SAP NetWeaver BW
 - callback 40, 68
 - configuring connectors 122
 - connections 70
 - CpicDriver error 122
 - datasources 121
 - optimizing 40
- SAP NetWeaver BW Data Warehousing Workbench 70
- SAS
 - optimizing queries 62
 - order of tables in from clause 62
- SELECT statement 109
- semi-join
 - optimizing 35
- SQL-92 statements 110
- statistics
 - purpose of 29
 - recording 31
 - using to optimize reports 31

