SAP BusinessObjects

Data Access Guide

- SAP BusinessObjects Business Intelligence platform 4.1 Support Package 2

2013-11-21

SAP

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 Document Version History....13

Chapter 2 Introduction to the Data Access Guide....17

2.1 About This Guide....17
2.2 Audience....17
2.3 Key Tasks....17
2.4 Conventions in This Guide....18

Chapter 3 Introduction to Data Access....19

3.1 About Connection Server....19
3.2 Components of a Connection....19
3.2.1 System Architecture....20
3.2.2 Data Access Drivers....20
3.3 Data Access Configuration Files....21
3.3.1 About the cs.cfg Global Configuration File....22
3.3.2 About the Driver Configuration Files....22
3.3.3 About the OlapClient.cfg Configuration File....24
3.4 Deployment Mode....24
3.5 Connectivity Services....24
3.6 OLAP Connections....26

Chapter 4 Learning Data Access Specifics....27

4.1 64-bit Operating System Support....27
4.1.1 64-bit UNIX Support....27
4.1.2 64-bit Microsoft Windows Support....28
4.1.3 Installation of ODBC Drivers....29
4.2 Single Sign-On Support....29
4.3 Stored Procedures....31
4.3.1 Supported Database Capabilities....31
4.3.2 Oracle Stored Procedures....32
4.3.3 To Create a Cursor Inside a Package....32
4.3.4 To Create an Oracle Stored Procedure....32

4.3.5 SAP HANA Stored Procedures....33
4.3.6 Teradata Macros....34
4.4 Connection Pools....34
4.4.1 About Connections of the Pool....35
4.4.2 Connection Pool Mode....35
4.4.3 Checking Middleware Connection Pooling....36
4.5 Load Balancing....36
4.5.1 Finding the Best Server....36
4.5.2 Load Balancing Logic....37
4.5.3 Backward Compatibility....38
4.6 Memory Allocation....38
4.6.1 To Enable HOARD....38
4.6.2 To Disable HOARD....39
4.7 Activity in CA Wily Introscope Workstation....39

Chapter 5 Creating a Connection....41

5.1 Connection Requirements....41
5.2 Checking Connection Configuration....41
5.2.1 Displaying Help on the cscheck Tool....42
5.2.2 To Run the cscheck Tool....43
5.2.3 Check Tool—Function Overview....43
5.2.4 Check Tool—list....44
5.2.5 Check Tool—driverssearch....45
5.2.6 Check Tool—find....46
5.2.7 Check Tool—middleware....47
5.2.8 Check Tool—accessdriver....48
5.2.9 Check Tool—connectivity....50
5.2.10 Check Tool—ping....51
5.2.11 Check Tool—CMS Information....53
5.3 Creating JDBC Connections....54
5.3.1 To Create a JDBC Connection with the SBO File....55
5.3.2 JDBC SBO Example File Structure....56
5.3.3 To Create a JDBC Connection with Extensions....56
5.3.4 To Create a Generic JDBC Connection....57
5.3.5 JAR File Location Reference....58
5.3.6 To Find the Version of a JDBC Driver....61
5.4 Creating JavaBean Connections....62
5.4.1 To Create a JavaBean Connection....63
5.4.2 JavaBean SBO Example File Structure....64
5.4.3 To Create a JavaBean Connection with Extensions....64
5.5 Creating ODBC Connections....65

5.5.1 To Create a Generic ODBC Connection....66
5.5.2 To Create a Generic ODBC3 Connection....67
5.6 To Set the Locale in a Multiple Data Sources Workflow....68

Chapter 6 Data Access Driver Reference....71

6.1 Data Access Drivers....71
6.2 CSV OpenDriver....71
6.2.1 CSV OpenDriver Capabilities....72
6.2.2 CSV OpenDriver - Table Mapping....72
6.2.3 CSV File Location....74
6.2.4 CSV Schema Detection....74
6.3 OData Driver....76
6.3.1 OData Driver Capabilities....76
6.3.2 OData Service Location....77
6.3.3 OData Service Example....78
6.3.4 OData Driver - Mapping Namespace, Owner and Qualifier....79
6.3.5 OData Driver - Mapping Tables....79
6.3.6 OData Driver - Mapping Columns....80
6.3.7 OData Driver - Mapping Keys....81
6.3.8 OData Driver - Mapping Documentation Elements....83
6.3.9 OData Driver - Mapping Entities of Derived Types....83
6.3.10 Monitoring the OData Driver Performance....84
6.3.11 OData Driver - Nontrivial Behaviors and Restrictions....86
6.3.12 OData Driver - Configuring Column Maximum Size....87
6.3.13 Authenticating OData Sources....88
6.3.14 Tracing the OData Driver Activity....88
6.4 SAP ERP Driver....88
6.4.1 SAP ERP Driver Capabilities....89
6.4.2 SAP ERP Driver - Access to InfoSets and SAP Queries....90
6.4.3 SAP ERP Driver - Access to ABAP Functions....92
6.4.4 SAP ERP Driver Restrictions....94
6.5 XML Driver....95
6.5.1 XML Driver Capabilities....96
6.5.2 XML File Location....96
6.5.3 XML Document and Schema Example....97
6.5.4 XML Driver - Mapping Multiple Files....99
6.5.5 XML Driver - Mapping Tables....99
6.5.6 XML Driver - Mapping Columns....100
6.5.7 XML Driver - Mapping Primary and Foreign Keys....101
6.5.8 XML Driver - Mapping Mixed Elements....103
6.5.9 XML Driver - Mapping Recursive Elements....104

6.5.10 XML Driver - Mapping any and anyAttribute....104
6.5.11 XML Driver - Mapping anyType and simpleType....106
6.5.12 XML Driver - Configuring Column Maximum Size....107
6.6 Web Service Driver....107
6.6.1 Web Service Driver Capabilities....108
6.6.2 Web Service Location....108
6.6.3 Web Service Definition Example....109
6.6.4 Mapping Rules for Web Services....110
6.6.5 Web Service Driver - Configuring Column Maximum Size....113
6.7 Local Disk Used as a Cache for Sorting Operations....114

Chapter 7 Connection Reference....115

7.1 HIVE Connections....115
7.1.1 To Create a Connection to Apache Hadoop HIVE....115
7.1.2 To Make HIVE Connections Work After Platform Update....117
7.1.3 To Create a Connection to Amazon EMR HIVE....118
7.2 IBM DB2 Connections - Reference Keys Mapped to Null....119
7.3 IBM Informix Connections....119
7.3.1 To Set the JVM Timezone for IBM Informix Connections....119
7.3.2 To Set the Transaction Mode for IBM Informix Connections....120
7.4 MS Analysis Services Connections....120
7.5 MS SQL Server Connections....121
7.5.1 Synonym Support for OLE DB Connections to MS SQL Server....121
7.5.2 To Set the JVM Options for Connections to MS SQL Server on UNIX....121
7.6 Oracle Connections....122
7.6.1 Connections to Clusters of Oracle Servers....122
7.6.2 To Set the Oracle CURSOR_SHARING Parameter Value....122
7.7 Oracle EBS Connections....123
7.8 Oracle Essbase Connections....124
7.9 Oracle RAC Connections....125
7.10 salesforce.com Connections - To Configure the Environment....125
7.10.1 To Make salesforce.com Connections Work in the Information Design Tool....126
7.10.2 To Make salesforce.com Connections Work in the Universe Design Tool....126
7.11 SAP BW Connections....127
7.11.1 To Enable SAP BW 64-bit Connections....127
7.12 SAP ERP Connections - Driver Failed to Load....127
7.13 SAP HANA Connections....128
7.13.1 Creating an SAP HANA Connection....129
7.13.2 Editing an SAP HANA Connection....130
7.13.3 Before Configuring Single Sign-On for SAP HANA Connections....130
7.13.4 To Configure Single Sign-On to SAP HANA for the Information Design Tool....131

7.13.5 To Configure Single Sign-On to SAP HANA for Web Intelligence....133
7.13.6 To Configure Single Sign-On to SAP HANA for Web Intelligence Rich Client....133
7.13.7 To Configure the Java Virtual Machine for Instrumentation with SAP HANA Connections....134
7.14 SAP MaxDB Connections....134
7.15 SAP NetWeaver BW Connections....135
7.15.1 Requirements for Connecting Data Federator to SAP NetWeaver BW....135
7.15.2 Requirements for Making SAP NetWeaver BW Connections Work in the Information Design Tool....136
7.15.3 Requirements for Accessing Multisource-Enabled Universes on SAP NetWeaver BW....136
7.16 SAS Connections....136
7.16.1 Installing Drivers for SAS Connections....137
7.17 Teradata Connections - Mapping Teradata Database to Current Owner....137

Chapter 8 Creating a Connection to Data Federator XI 3.0 Query Server....139

8.1 About Data Federator XI 3.0 Query Server Connections....139
8.2 Configuring the Connection Wizard for a Data Federator JDBC or ODBC Connection....140
8.3 Configuring Data Federator ODBC Connections....140
8.3.1 Configuring the Data Federator ODBC Middleware....141
8.3.2 Configuring Connection Server for a Data Federator ODBC Connection....141
8.4 Configuring Web Intelligence Rich Client Connections Using Data Federator ODBC Middleware....142
8.4.1 Configuring the Data Federator ODBC Middleware for a Connection to Web Intelligence Rich Client....143
8.4.2 Configuring Connection Server for a Web Intelligence Rich Client Connection to Data Federator....143
8.4.3 Setting the Windows RichClient Registry Key....144
8.4.4 Configuring Connection Server for Connections of Web Intelligence Rich Client or Universe Design Tool to Data Federator....144

Chapter 9 Configuring Data Access Global Parameters....147

9.1 About Global Parameters....147
9.2 About the cs.cfg Configuration File....147
9.3 To View and Edit the cs.cfg File....148
9.4 Configuring the Global Settings Parameters....148
9.4.1 Charset List Extension....149
9.4.2 Config File Extension....149
9.4.3 Description Extension....149
9.4.4 Ignore Driver Load Failure....150
9.4.5 Load Drivers On Startup....150
9.4.6 Max Pool Time....151
9.4.7 Setup File Extension....152

9.4.8 SQL External Extension....152
9.4.9 SQL Parameter Extension....152
9.4.10 Strategies Extension....153
9.4.11 Validate Configuration Files....153
9.4.12 Validate XML Streams....154
9.5 Setting the Deployment Mode....154
9.6 Configuring the Deployment Mode....155
9.7 Configuring the Drivers to Load....155
9.7.1 Setting One Connectivity per Machine....156
9.8 Configuring the CORBA Access Protocols....156
9.9 Activating Logs and Traces of Connection Server and Drivers....157
9.9.1 About the _trace.ini Configuration File....157
9.9.2 Activating Logs and Traces in the cs.cfg File....160
9.9.3 To activate Logs and Traces in Library Mode....161
9.9.4 To activate Logs and Traces in Server Mode....162
9.9.5 Reading Logs and Traces....162
9.10 Activating Logs and Traces for the OLAP Client....163
9.10.1 Example of Log....164

Chapter 10 Configuring Data Access Driver Parameters....165

10.1 Configuring Driver Parameters....165
10.1.1 Data Access Configuration Files....165
10.1.2 Installed SBO Files....166
10.1.3 To View and Edit SBO Files....171
10.1.4 To Customize SBO Files....171
10.1.5 To Check Connections Dynamically....172
10.1.6 JDBC Driver Properties....173
10.2 About DataDirect ODBC Drivers....173
10.2.1 To Enable the Use of DataDirect Branded Drivers....174

Chapter 11 SBO Parameter Reference....177

11.1 SBO File Structure....177
11.2 SBO Parameter Description....178
11.3 Common SBO Parameters....179
11.3.1 Array Bind Available....179
11.3.2 Array Bind Size....180
11.3.3 Array Fetch Available....180
11.3.4 Array Fetch Size....181
11.3.5 BigDecimal Max Display Size....182
11.3.6 Binary Max Length....182

11.3.7 Bucket Split Size....183
11.3.8 Catalog Separator....184
11.3.9 CharSet Table....184
11.3.10 Description File....184
11.3.11 Dictionary Transaction Mode....185
11.3.12 Driver Capabilities....185
11.3.13 Driver Name....186
11.3.14 Escape Character....186
11.3.15 Extensions....187
11.3.16 Family....187
11.3.17 Force Execute....188
11.3.18 Identifier Case....188
11.3.19 Identifier Quote String....189
11.3.20 Include Synonyms....189
11.3.21 Integer Max Length....190
11.3.22 Introscope Available....190
11.3.23 Max Rows Available....191
11.3.24 Native Int64 Available....191
11.3.25 Optimize Execute....192
11.3.26 Owners Available....192
11.3.27 Qualifiers Available....193
11.3.28 Query TimeOut Available....193
11.3.29 Quote Identifiers....194
11.3.30 Skip SAML SSO....194
11.3.31 SQL External File....194
11.3.32 SQL Parameter File....195
11.3.33 SSO Available....195
11.3.34 Strategies File....196
11.3.35 String Max Length....196
11.3.36 Temp Data Dir....197
11.3.37 Transactional Available....198
11.3.38 Type....199
11.3.39 Unicode....199
11.3.40 URL Format....199
11.3.41 XML Max Size....200
11.4 JavaBean SBO Parameters....200
11.4.1 JavaBean Class....201
11.5 JCO SBO Parameters....201
11.5.1 ERP Max Rows....201
11.6 JDBC SBO Parameters....202
11.6.1 Connection Shareable....202

11.6.2 Escape Character Available....203
11.6.3 ForeignKeys Available....203
11.6.4 Get Extended Column....203
11.6.5 JDBC Class....204
11.6.6 PrimaryKey Available....204
11.6.7 PVL Available....205
11.6.8 Shared Connection....205
11.6.9 JDBC ResultSet Type....206
11.6.10 JDBC ResultSet Concurrency....206
11.7 OData SBO Parameters....207
11.7.1 Enforce Max Protocol Version....207
11.8 ODBC SBO Parameters....208
11.8.1 CharSet....208
11.8.2 Connection Status Available....208
11.8.3 Cost Estimate Available....209
11.8.4 Empty String....209
11.8.5 ODBC Cursors....210
11.8.6 SQLDescribeParam Available....210
11.8.7 SQLMoreResults Available....211
11.8.8 Use DataDirect OEM Driver....211
11.8.9 V5toV6DriverName....212
11.9 OLE DB SBO Parameters....212
11.9.1 Enumerator CLSID....212
11.9.2 Provider CLSID....213
11.10 OLE DB OLAP SBO Parameters....213
11.10.1 MSOlap CLSID....213
11.11 Sybase SBO Parameters....213
11.11.1 Driver Behavior....214
11.11.2 Password Encryption....214
11.11.3 Quoted Identifier....215
11.11.4 Recover Errors....215
11.11.5 Text Size....215
11.12 Teradata SBO Parameters....216
11.12.1 Replace Current Owner With Database....216

Chapter 12 Configuring Database Capability Parameters....219

12.1 About Database Capability Parameters....219
12.2 About PRM Files....219
12.2.1 PRM Parameter File Structure....220
12.3 To View and Edit PRM Files....221
12.4 To Verify and Add Analytic Function Support to PRM Files....222

12.5 To View and Edit a Function Help Text File....223
12.6 To Edit the Help Text for a PRM Function....223

Chapter 13 PRM Parameter Reference....225

13.1 PRM file Configuration Reference....225
13.1.1 ANALYTIC_CLAUSE....225
13.1.2 ANALYTIC_FUNCTIONS....226
13.1.3 CALCULATION_FUNCTION....226
13.1.4 CONSTANT_SAMPLING_SUPPORTED....227
13.1.5 DISTINCT....227
13.1.6 EXT_JOIN....228
13.1.7 FULL_EXT_JOIN....228
13.1.8 GROUP_BY....229
13.1.9 GROUP_BY_SUPPORTS_COLUMN_INDEX....229
13.1.10 GROUP_BY_SUPPORTS_COMPLEX....230
13.1.11 GROUP_BY_SUPPORTS_CONSTANT....230
13.1.12 HAVING....231
13.1.13 INTERSECT....231
13.1.14 INTERSECT_ALL....232
13.1.15 INTERSECT_IN_SUBQUERY....232
13.1.16 JOIN....232
13.1.17 LEFT_EXT_JOIN....233
13.1.18 LEFT_OUTER....233
13.1.19 LIKE_SUPPORTS_ESCAPE_CLAUSE....234
13.1.20 MINUS....234
13.1.21 MINUS_ALL....235
13.1.22 MINUS_IN_SUBQUERY....235
13.1.23 ORDER_BY....236
13.1.24 ORDER_BY_REQUIRES_SELECT....236
13.1.25 ORDER_BY_SUPPORTS_COLUMN_INDEX....236
13.1.26 PERCENT_RANK_SUPPORTED....237
13.1.27 RANK_SUPPORTED....237
13.1.28 RIGHT_EXT_JOIN....238
13.1.29 RIGHT_OUTER....238
13.1.30 SEED_SAMPLING_SUPPORTED....239
13.1.31 SELECT_SUPPORTS_NULL....239
13.1.32 SUBQUERY_IN_FROM....240
13.1.33 SUBQUERY_IN_IN....240
13.1.34 SUBQUERY_IN_WHERE....240
13.1.35 TECHNICAL_COLUMN_NAME_PATTERN....241
13.1.36 UNION....241

13.1.37 UNION_ALL....242
13.1.38 UNION_IN_SUBQUERY....242

Chapter 14 Data Type Conversion Reference....243

14.1 Data Type Conversion....243
14.1.1 CSV File Data Types....244
14.1.2 JDBC Data Types....245
14.1.3 ODBC Data Types....248
14.1.4 OData Data Types....250
14.1.5 OLE DB Data Types....251
14.1.6 Oracle OCI Data Types....253
14.1.7 SAP ERP Data Types....254
14.1.8 SAP HANA Data Types....255
14.1.9 Sybase CTL Data Types....256
14.1.10 XML Data Types....258
14.2 Large Variable Length Data Restriction....260
14.3 Data Type Mapping for Multisource-Enabled Universes....260

Index 261

Document Version History

The following table provides an overview of the most important document changes.

Version	Date	Changes
SAP BusinessObjects Business Intelligence platform 4.1	May 2013	New OData, XML and Web Service drivers, see Data Access Drivers. Support of single sign-on for Teradata 13, Teradata 14, and Sybase IQ, see Single Sign-On Support. Support of SAP HANA stored procedures, see SAP HANA Stored Procedures. For SAP ERP connections, simplified mapping of optional input columns, see SAP ERP Driver - Access to ABAP Functions. For SAP HANA connections, support of new SAP HANA SPS 05 release, OLAP connections, single sign-on through SAML protocol, SSL protocol, and 64-bit UNIX platforms through ODBC, see SAP HANA Connections. Support of HIVE 0.9 connections through JDBC, see HIVE Connections. New logger for Connection Server activity, see Activating Logs and Traces of Connection Server and Drivers. How to activate OLAP Client logs, see Activating Logs and Traces for the OLAP Client. Support of DataDirect ODBC 7.0 drivers, see About DataDirect ODBC Drivers. New SBO parameter for any data access driver, see Dictionary Transaction Mode. New SBO parameters for MySQL connections, see JDBC ResultSet Type and JDBC ResultSet Concurrency. New SBO parameter for Teradata connections, see Replace Current Owner With Database. New SBO parameter for skipping SAML with SAP HANA connections, see Skip SAML SSO.

Version	Date	Changes
SAP BusinessObjects Business Intelligence platform 4.1 Support Package 1	August 2013	New connectivities are available, see the *Product Availability Matrix* for more information. Description of the connection pool mode, see Connection Pool Mode. To check connections stored on the CMS , see Check Tool—CMS Information. To find the JDBC driver version, see To Find the Version of a JDBC Driver. Tuning the OData connection timeout to improve the performance, see Monitoring the OData Driver Performance. Support of connections to different versions of the HIVE database, see HIVE Connections. UNIX settings for MS SQL Server connections, see To Set the JVM Options for Connections to MS SQL Server on UNIX. To set the `CURSOR_SHARING` parameter value, see Oracle Connections. Support of 64-bit connections to SAP BW for `.unv` universes, see SAP BW Connections. If the SAP ERP driver fails to load, see SAP ERP Connections - Driver Failed to Load. Support of SAP HANA database 1.0 SPS 06, see SAP HANA Connections. Security requirements for SAP NetWeaver BW connections, see SAP NetWeaver BW Connections. `cs.cfg` file example of Connection Server in server mode, see Setting the Deployment Mode. Date format to be used in queries to SAP ERP systems, see SAP ERP Data Types. SAP HANA data types and their equivalent in the data foundations, see SAP HANA Data Types.

Version	Date	Changes
SAP BusinessObjects Business Intelligence platform 4.1 Support Package 2	November 2013	New connectivities are available, see the *Product Availability Matrix* for more information. Update of the 64-bit Microsoft Windows support section, see 64-bit Microsoft Windows Support. Support of locale in native connections of multisource-enabled universes, see To Set the Locale in a Multiple Data Sources Workflow. To install the Apache Hadoop HIVE driver after a platform update, see To Make HIVE Connections Work After Platform Update. Synonym columns of MS SQL Server through OLE DB not supported, see MS SQL Server Connections. Update of the Oracle Essbase section, see Oracle Essbase Connections. Update of the SAP BW Connections section, see SAP BW Connections. Support of SAP HANA database 1.0 SPS 07, see SAP HANA Connections. To support new analytic functions, see To Verify and Add Analytic Function Support to PRM Files.

Introduction to the Data Access Guide

2.1 About This Guide

The Data Access Guide helps you learn about the Connection Server functionality and how to configure Connection Server to enable connections of SAP BusinessObjects Business Intelligence platform 4.1 SP1 to production databases.

The Data Access Guide provides the following information:

- Learning Connection Server fundamentals
- Learning about data access drivers for CSV, OData, SAP ERP, and XML data sources
- Creating a JDBC, ODBC, or Javabean connection
- Configuring data access parameters

Note:
It also provides information for configuring some relational connections that rely on the data federation service, and OLAP connections.

2.2 Audience

The Data Access Guide is intended for the following audience:

- SAP BusinessObjects application users in charge of creating connections to data sources
- System administrators who are responsible for configuring, managing, and maintaining a BI platform installation

2.3 Key Tasks

The Data Access Guide provides you with key information for managing configuration parameters and establishing connections. For each of these following tasks, refer to the appropriate section below:

- How to set the server working mode?
- How to choose the drivers you want to load?

- How to configure data access drivers?
- How to check connection configuration?
- How to create JDBC connections?
- How to create SAP HANA connections?

Note:
For administrative tasks such as starting and stopping Connection Server, managing properties and metrics, refer to the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

Related Topics

- Configuring the Deployment Mode
- Configuring the Drivers to Load
- To View and Edit SBO Files
- To Run the cscheck Tool
- Creating JDBC Connections
- SAP HANA Connections

2.4 Conventions in This Guide

In this guide, the variable *connectionserver-install-dir* is the installation root path for the data access files used by SAP BusinessObjects client tools. On Microsoft Windows, the default *connectionserver-install-dir* stands for `C:\Program Files\SAP Business Objects\SAP BusinessObjects Enterprise XI 4.0\dataAccess`.

The variable *bip-install-dir* is the installation root path of the BI platform or Client Tools. On MS Windows (64-bit), it stands for the `C:\Program Files (x86)\SAP Business Objects\SAP BusinessObjects Enterprise XI 4.0` directory.

Caution:
In the data access configuration files, use the escape sign `\` with the backslash `\` in file paths if you deploy the BI platform on Microsoft Windows.

Introduction to Data Access

3.1 About Connection Server

Connection Server is the data access software that manages the connection between an SAP BusinessObjects application and a data source.

Connection Server allows applications such as universe design tool, information designer tool and SAP BusinessObjects Web Intelligence to connect to and run queries against a data source.

Connection Server does not have a user interface. You create and administer connections from the user interface of these applications, or by editing Connection Server configuration files.

- Creating connections

  You create connections using the connection wizard of the BI platform Client Tools.

  Note:
  The connection wizard of the universe design tool is the New Connection wizard. In the information design tool, it can be either the New Relational Connection wizard or the New OLAP Connection wizard. See the application user guides to learn how to use the connection wizard.

- Optimizing data access

  You can optimize the way that data is passed through Connection Server by modifying data access configuration files. These files are in XML format, and are installed with Connection Server. You can set parameter values to apply to a specific data access driver, or to all installed data access drivers.

3.2 Components of a Connection

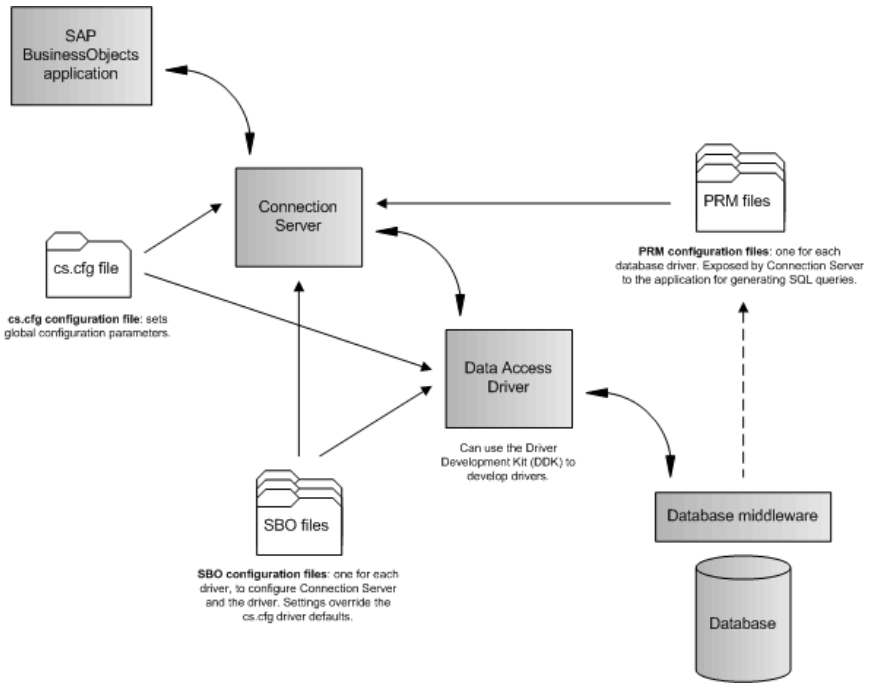
A data access connection consists of the following components:

- Connection Server is the software that manages the connection between the application and the data source. For example, Connection Server handles requests for data from the application.
- A data access driver is the database-specific software component that manages the connection between Connection Server and the database middleware.
- Configuration files define parameters to configure the connection between the following systems:
  - The application and Connection Server

- The application and the data access driver
- Connection Server and the data access driver

3.2.1 System Architecture

The diagram below details where Connection Server and data access drivers fit into an SAP BusinessObjects configuration.



3.2.2 Data Access Drivers

Data access drivers provide the connection between Connection Server and a data source. A database requires a data access driver in order that it can be accessed by an SAP BusinessObjects application.

SAP BusinessObjects applications include data access drivers that you can use to configure connections to your databases. The data access drivers that are included can depend on your licence.

Before you can create a connection to a database for which you do not have a driver, you need to obtain the required drivers. The following options are available for obtaining a driver:

- Contact your SAP representative to determine if there is a driver available, and if you are licenced to use it.
- Use the Driver Development Kit (DDK) to develop a driver to use. Contact your SAP representative for details.

When you create a new connection, you select the appropriate data access driver for the target data source. For example, if you access an Oracle 10g database, you must install the appropriate middleware (Oracle 10g Client), then the SAP BusinessObjects Oracle data access driver.

Caution:
Excel bean (`bean_excel.jar`) and CSV (`dbd_open_sample.jar`) data access drivers are driver samples. You should not use them as is, but as starting points for developing more complex drivers by using the DDK.

For an up-to-date list of supported data access drivers, check the SAP Service Marketplace at http://service.sap.com/bosap-support, or contact your SAP representative.

For more information about the DDK, refer to the *Data Access Driver Java SDK Developer Guide* at http://boc.sdn.sap.com.

3.3 Data Access Configuration Files

Data access configuration files come with installation of the BI platform. They can be divided into the following levels:

- Global level

  The `cs.cfg` configuration file applies to all connections.
- Driver level

  The SBO configuration files apply to specific drivers.
- OLAP connections

  The `OlapClient.cfg` configuration file applies to OLAP connections for `.unx` universes.

In addition to the configuration files that control a connection, each data access driver has an associated PRM configuration file. These files control the way in which an application generates SQL depending on the database software capabilities. They are used by applications such as the information design tool.

Related Topics

- OLAP Connections

  The Data Access layer allows the BI platform to connect to OLAP data sources.
- About the cs.cfg Configuration File
- Data Access Configuration Files

3.3.1 About the cs.cfg Global Configuration File

The `cs.cfg` global configuration file used by all data access drivers is installed in the following location:

- *connectionserver-install-dir*\connectionServer

The `cs.cfg` file contains parameters that apply to all installed data access drivers.

Related Topics

- About Global Parameters

3.3.2 About the Driver Configuration Files

The configuration files used by data access drivers are installed in the following path:

- On a Microsoft Windows system:

  connectionserver-install-dir\connectionServer*RDBMS*

- On a UNIX system:

  connectionserver-install-dir/connectionServer/*RDBMS*

where *RDBMS* is either the name of the network layer or the database middleware that uses the configuration file.

The files listed below have parameters that apply to installed data access drivers.

Driver Specific File	Can Be Edited	Description	Example
<driver>.sbo	Yes	Each data access driver has a SBO file. Defines the specific connectivity configuration for each driver and target database.	oracle.sbo

Driver Specific File	Can Be Edited	Description	Example
`<driver>.prm`	Yes	Each data access driver has a PRM file. Defines parameters that affect the way that an application generates SQL.	`oracle.prm`
`<driver><language>.cod`	No	Each data access driver has a COD file. Stores information related to connection definitions. Defines the fields that appear when you create a new connection. **Note:** Do not modify these files.	`oracleen.cod`
`<driver>.rss`	No	Each data access driver has a RSS file. Stores predefined SQL sentences, which are used by Connection Server.	`oracle.rss`
`<driver>.stg`	No	The data access driver can have a strategy file. See SBO file parameter reference for more information.	`oracle.stg`

Related Topics

- SBO Parameter Description
- PRM file Configuration Reference
- About Database Capability Parameters

3.3.3 About the `OlapClient.cfg` Configuration File

On Microsoft Windows, the `OlapClient.cfg` file is stored in the following location:

- *bip-install-dir*\win32_x86

In the `OlapClient.cfg` file, you can configure parameters of the `OlapClient` section only.

Related Topics

- Activating Logs and Traces for the OLAP Client

3.4 Deployment Mode

Connection Server can run in the following deployment modes:

- Library mode (in-proc)

  Connection Server is included in the client process. Most SAP BusinessObjects applications use Connection Server in library mode.

- Server mode

  Connection Server is a CORBA server and is accessed remotely. Connection Server serves the CORBA and HTTP clients to address the 2-tier and web tier deployment modes respectively.

See the *SAP BusinessObjects Business Intelligence platform Administrator Guide* for more information on the deployment scenarios.

Related Topics

- Setting the Deployment Mode
- Configuring the Deployment Mode
- Configuring the Drivers to Load

3.5 Connectivity Services

Three Connection Server server instances come with the default BI platform installation. They are grouped under **Connectivity Services** in the Central Management Console (CMC).

Connection Server servers host the following services:

- Native Connectivity Service (64-bit)
- Native Connectivity Service (32-bit on MS Windows only)

The Adaptive Processing Server hosts the Adaptive Connectivity Service, which allows user applications to access Java-based data sources remotely.

When starting up, the Connectivity Services advertise the list of the data sources they support on the BI platform cluster, so that SAP BusinessObjects applications are able to look up and use the appropriate server instance. Applications look for data sources through Connection Server frst in library mode, then in the server mode.

Connection vs. Service

When Connection Server is used in library mode, the list of available data sources is defined by the data access drivers and middleware installed on the local machine. When Connection Server is used in server mode, the list of data sources also includes those supported by each server instance that runs on the back-end system of the BI platform.

Each server instance supports a subset of the data sources supported by the Data Access layer. The subset depends on the following parameters:

- The CS server implementation technology (C++ or Java)
- The host operating system (UNIX flavors or MS Windows)
- The drivers you can select when installing the BI platform
- The active data sources you can select for each server instance in the CMC

The following table describes which Connectivity Service each kind of connection can use.

Connection	Connectivity Service	Description
All 64-bit native data sources	Native Connectivity Service	Support of ODBC, OLE DB, OCI, and so on.
All 32-bit native data sources	Native Connectivity Service	Support of data sources available in 32-bit only. This service is only available on MS Windows.
Java-based data sources	Adaptive Connectivity Service	Support of all Java-based middleware. This service is a Java implementation based on the Platform Java Service (PJS) framework.

Example:

- A 64-bit native data source is Oracle database through Oracle OCI.
- A 32-bit native data source is MS Excel 2007 through ODBC.
- A Java-based data source is MS SQL Server 2008 R2 through JDBC.

3.6 OLAP Connections

The Data Access layer allows the BI platform to connect to OLAP data sources.

The `.unv` universes based on OLAP data sources use connections managed by the 32-bit Connection Server. The present release allows SAP BW connections to use either the 32-bit or the 64-bit Connection Server.

The `.unx` universes based on OLAP data sources use connections managed by the OLAP Client component of the BI platform.

For the list of the supported OLAP data sources, see the *Product Availability Matrix*. For creating OLAP connections, see the *Information Design Tool User Guide*.

Related Topics

- 64-bit Microsoft Windows Support
- SAP BW Connections

The Data Access layer allows the BI platform to access the SAP BW data source.

Learning Data Access Specifics

4.1 64-bit Operating System Support

SAP BusinessObjects provides releases of the BI platform for the following operating systems:

- 32-bit versions of Microsoft Windows
- 64-bit versions of Microsoft Windows and UNIX flavors

The Data Access layer then provides data access drivers that are able to run either in a 32-bit or a 64-bit environment.

The following sections give details on what this implies for database middleware support in 32-bit or 64-bit environments.

For the complete list of data sources supported on 64-bit operating systems, see the *Product Availability Matrix*.

Note:
The DDK provides data access driver samples that are able to run either in a 32-bit or in a 64-bit environment. For more information about the DDK, see the *Data Access Driver Java SDK Developer Guide*.

4.1.1 64-bit UNIX Support

Connection Server comes with the BI platform and then supports installation on 64-bit UNIX environments as in-proc or as a remote server.

Caution:
You must make sure you install 64-bit middleware to run database connections through Connection Server.

Some vendors do not provide 64-bit middleware for the following databases and network layers on UNIX. They are only available on Microsoft Windows:

- DB2 UDB for iSeries V5 with Client Access AS400
- DB2 UDB for iSeries V6 with Client Access AS400
- DB2 for i v6.1 with Client Access AS400
- DB2 for i v7.1 with Client Access AS400

- Generic data source through OLE DB
- MS Access 2010 and 2013 through ODBC
- MS Excel 2010 and 2013 through ODBC
- MS SQL Server 2008, 2008 R2, and 2012 through OLE DB

4.1.2 64-bit Microsoft Windows Support

Remember:
This section is only about databases used for `.unv` universes.

Connection Server comes with the BI platform and then supports installation on both 32-bit (as in-proc) and 64-bit MS Windows environments (as in-proc and remote server).

Vendors do not provide 64-bit middleware for the following databases through ODBC on Microsoft Windows:

- Ingres database 9
- MS Access 2007 and MS Excel 2007
- PostgreSQL 8
- Text files
- Data Federator XI 3.0 Query Server
- Salesforce.com

64-bit Microsoft Windows also does not support the following OLAP databases and their middleware:

- MS Analysis Services database through OLE DB for OLAP
- Oracle Essbase 9 and 11 through Oracle Essbase Client

For all these data sources, the Data Access layer is able to work with 32-bit middleware on 64-bit versions of Microsoft Windows. This functionality is handled through a specific deployment that consists of two Connection Servers running simultaneously in server mode. The first one is 32-bit and runs connections to data sources that cannot handle 64-bit middleware, while the second one is 64-bit and runs connections to the other data sources. As the `cs.cfg` file is common to both servers, they share the same configuration.

Restriction:
This architecture implies you cannot run the same data access driver on both servers. However, you can establish a generic ODBC connection on either 32-bit Connection Server or 64-bit Connection Server, but not on both servers at the same time.

Related Topics

- Oracle Essbase Connections

  The Data Access layer allows the BI platform to access the Oracle Essbase data source.

- SAP BW Connections

The Data Access layer allows the BI platform to access the SAP BW data source.

- Array Fetch Size

4.1.3 Installation of ODBC Drivers

For installations of the information design tool or universe design tool on a 32-bit Windows operating system, the drivers for any ODBC data sources that need to be created, tested, and accessed in the tool should be defined using the ODBC Data Source Administrator that can be found at the following location on the physical machine where the tool is installed:

- `C:\Windows\System32\odbcad32.exe`

For installations of the information design tool or universe design tool on a 64-bit Windows operating system, the drivers for any ODBC data sources that need to be created, tested, and accessed in the tool should be defined using the 32-bit version of ODBC Data Source Administrator that can be found at the following location on the physical machine where the tool is installed:

- `C:\Windows\SysWOW64\odbcad32.exe`

For the information design tool, the ODBC data sources that need to be used by the server components are defined using the ODBC Data Source Administrator that can be found at the following location on the physical machine where the server components are installed:

- 64-bit version: `C:\Windows\System32\odbcad32.exe` (recommended to use when 64-bit database middleware exists)
- 32-bit version: `C:\Windows\SysWOW64\odbcad32.exe` (use when only 32-bit database middleware exists)

Note:
If the information design tool or universe design tool is using ODBC DSN for universe data sources, DSN with the same names should be created on the physical machine where server components are installed for the SAP BusinessObjects reporting applications that use the published universes.

For information, refer to *SAP BusinessObjects Business Intelligence platform Installation Guide*.

4.2 Single Sign-On Support

The BI platform provides single sign-on (SSO) authentication when installed on the following platforms and with the following connectivities:

Data Source	Description
MS Analysis Services on Microsoft Windows	Single sign-on to BI platform is provided by Windows AD with Kerberos.
MS SQL Server through ODBC or OLE DB on Microsoft Windows	Single sign-on to BI platform is provided by Windows AD with Kerberos.
Oracle through OCI on Microsoft Windows	Single sign-on to BI platform is provided by LDAP.
Oracle EBS through OCI on all platforms	You enable SSO by installing and configuring the authentication plugin that comes with the platform. Users log into the BI platform from an SAP BusinessObjects application by using their EBS credentials (username and password).
SAP BW through OLAP BAPI on all platforms	You enable SSO by installing and configuring SAP Authentication. Users log into the BI platform from an SAP BusinessObjects application by using their SAP BW credentials.
SAP ERP systems through SAP Java Connectivity (JCo) 3.x on all platforms	You enable SSO by installing and configuring SAP Authentication. Users log into the BI platform from an SAP BusinessObjects application by using their SAP ERP credentials.
SAP HANA database 1.0 SPS 07 through ODBC, JDBC on all platforms and OLAP on Microsoft Windows, Linux and AIX	The SSO provided is SSO to database and uses either Windows AD with Kerberos (in ODBC, JDBC and OLAP) or SAML protocol (in JDBC and OLAP). Users log into the BI platform from an SAP BusinessObjects application by using their Windows AD credentials.
Sybase IQ database through ODBC on Microsoft Windows	Single sign-on to BI platform is provided by Windows AD with Kerberos.
Teradata 13 and Teradata 14 through ODBC on Microsoft Windows	Single sign-on to BI platform is provided by Windows AD with Kerberos.

For more information about SSO, refer to the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

Related Topics

- Oracle EBS Connections

- SAP ERP Driver
- SAP HANA Connections

4.3 Stored Procedures

Connection Server is able to manage data from data sources that result either from SQL query or stored procedure execution.

Stored procedures are SQL scripts that are stored as executable code in an RDBMS. They can receive arguments and return data.

Stored procedures for the following databases and network layers are supported in the BI platform:

- DB2 UDB and iSeries through CLI driver
- Sybase Adaptive Server through CTLIB
- Javabean
- DB2 UDB, Derby, HSQL DB, Informix, MS SQL Server, MySQL 5, Oracle, SAP HANA and Sybase, all through JDBC
- Oracle through OCI
- DB2 iSeries, Informix, MS SQL Server, SAP HANA, Sybase ASIQ, Sybase SQL Anywhere, and Teradata (only macros) all through ODBC
- MS SQL Server through OLE DB

4.3.1 Supported Database Capabilities

Connection Server only supports stored procedures that return data as result sets, that is, tables. This means that the stored procedure cannot return integers, strings, or cursors and must always contain `SELECT` statements. Plus, supported stored procedures must not contain `OUT` or `IN/OUT` parameters, but only `IN` parameters. In addition, `COMPUTE`, `PRINT`, `OUTPUT` or `STATUS` statements contained in stored procedures are not executed.

Caution:
These restrictions are not valid for Oracle stored procedures. Refer to next section for learning about supported Oracle stored procedures.

Connection Server supports Oracle stored procedures inside a package. The package name is returned as the catalog name. This behavior is valid for Oracle CI and JDBC network layers.

For more information about using stored procedures, refer to the *Universe Design Tool User Guide*.

4.3.2 Oracle Stored Procedures

The supported Oracle stored procedures are the following:

- Any PL/SQL procedure that returns result sets through a REF Cursor
- PL/SQL stored procedures that have one `IN/OUT` REF cursor variable parameter and no `OUT` parameter

Note:
The other `IN/OUT` cursor parameters of the procedure are ignored.

The unsupported Oracle stored procedures are the following:

- Any PL/SQL procedure that does not return result sets through a REF CURSOR parameter
- Any PL/SQL procedures having at least one `OUT` parameter
- Any PL/SQL function
- Any PL/SQL procedure having one `IN/OUT` parameter of a type different than a REF CURSOR, for example, `VARRAY`
- Any PL/SQL Table function

To access Oracle stored procedures, you must do a number of tasks at the server end to allow the BI platform to connect to a stored procedure. This is explained in the next sections.

4.3.3 To Create a Cursor Inside a Package

In Oracle databases, a package is a database object that contains related PL/SQL types, objects, and subprograms. You must first create a cursor inside a package before creating an Oracle stored procedure using the defined cursor.

- In the Oracle database administration system, use the following statement:

```
CREATE or REPLACE PACKAGE catalog_data AS
 TYPE CatCurTyp IS REF CURSOR RETURN
 all_objects%ROWTYPE;
END catalog_data;
```

The BI platform does not support packaged stored procedures, only standalone.

4.3.4 To Create an Oracle Stored Procedure

In the following procedure, you use `catcurtyp` cursor that you previously created in the package, and `catalog_data.catcurtyp`.

- Do one of the following:
  a. Write the following statement:

```
CREATE or REPLACE PROCEDURE get_allobjects(cat_cv IN OUT
 catalog_data.catcurtyp) AS
BEGIN
 OPEN cat_cv FOR SELECT * FROM all_objects;
END;
```

  b. Write the following statement with several parameters:

```
CREATE or REPLACE PROCEDURE get_ownerobjects(owner_name IN
varchar2, cat_cv IN OUT catalog_data.catcurtyp) AS
BEGIN
 OPEN cat_cv FOR SELECT * FROM all_objects WHERE
 owner=owner_name;
END;
```

Note:
See your Oracle documentation for more information on how to create packages and stored procedures.

4.3.5 SAP HANA Stored Procedures

Connection Server supports SAP HANA stored procedures through JDBC and ODBC.

You create SAP HANA stored procedures with the following syntax:

```
CREATE PROCEDURE <procedure_name>
{LANGUAGE <lang>} {SQL SECURITY <mode>}
{READS SQL DATA {WITH RESULT VIEW <view_name>}} AS
 BEGIN
   select ... from <table_name>;
 END
```

`LANGUAGE`, `SQL SECURITY` and `READS SQL DATA` are not mandatory. The default language is SQLScript. See the *SAP HANA Database SQLScript Guide* for more information.

Example:

Simple stored procedure:

```
CREATE PROCEDURE Proc
LANGUAGE SQLSCRIPT READS SQL DATA AS
 BEGIN
   select * from CUSTOMER;
 END
```

Stored procedure with a parameter and a result view:

```
CREATE PROCEDURE ProcWithResultView(IN id int)
LANGUAGE SQLSCRIPT READS SQL DATA WITH RESULT VIEW ProcView AS
 BEGIN
   select * from CUSTOMER where CUST_ID = :id;
 END
```

4.3.6 Teradata Macros

Connection Server only supports Teradata macros when the connection established uses ODBC.

Caution:
It does not support Teradata stored procedures in ODBC, because stored procedures do not return any result sets. It also does not support macros and stored procedures in JDBC.

You create Teradata macros with the following syntax:

```
create macro <macro_name> as (select * from <table_name>;);
```

See the Teradata documentation for more information.

Example:

Simple macro:

```
create macro GUEST95 as (select * from guest where TYear='FY95';);
```

Macro with a parameter:

```
create macro MGUESTIN (inyear VARCHAR(12)) as (select * from GUEST where TYear=:inyear;);
```

Macro with multiple result sets:

```
create macro MGUEST as
(
 select * from guest where TYear='FY95';
 select count(*) from guest;
);
```

4.4 Connection Pools

A driver opens a connection to the database in order to access data. The following are two methods that you can use to connect to a database:

- Each time that Connection Server requires information, the data access driver opens a connection to the database, retrieves the data, then closes the connection.
- Connection Server keep available connections open and maintain their details in a connection pool. Each time that Connection Server requires information from the data source, the data access driver checks the connection pool to see if it contains an unused, suitable connection. If an existing connection is available, the connection is used. If all connections are being used, Connection Server creates a new connection and adds it to the pool. This method uses system resources more efficiently.

Remember:
Connection Server does not add a connection that uses single sign-on to the pool.

4.4.1 About Connections of the Pool

Connections that are available in the connection pool can be exclusive or shareable.

- Exclusive connections can be allocated to one user only at a time. When an exclusive connection is allocated, it is no longer available in the pool. Then it cannot be allocated to any other requesters. When the connection is no longer required, the custom driver releases the connection so that it can be reallocated.
- Shareable connections can be allocated to multiple users at a time. When a connection is allocated, the connection remains in the pool so that it is available for other requesters.

You can choose to create an exclusive or shareable connection with the **Connection Pool Mode** parameter of the connection wizard.

Related Topics

- Connection Shareable
- Shared Connection
- Max Pool Time

4.4.2 Connection Pool Mode

Connection Pool Mode Value	Connection Pool Mode Description
Disconnect after each transaction	Disconnects from the database after a query is completed. The next time a query is run, the connection is re-created.
Keep the connection active for	This is the connection pooling option. If a query is completed before the time specified in **Pool Timeout** (10 minutes is the default), then the connection can be re-used. All the users share the connection.
Keep the connection active during the whole session (local mode only)	The connection is closed when the user exits the application. This option does not use the connection pooling.

Choosing the Connection Pool Mode

The connection pool mode may depend on the availability of your marchine resources.

- If the memory is low, then choose the **Disconnect after each transaction** option.
- If the memory is high and the performance is critical, then choose the **Keep the connection active during the whole session (local mode only)** option.
- To control the behavior more precisely, choose the **Keep the connection active for** option.

4.4.3 Checking Middleware Connection Pooling

Data source middleware often provide their own connection pool mechanism. You must make sure the middleware connection pool setting is compatible with the one set in the design tool to obtain the connection performance that you expect.

4.5 Load Balancing

In server mode, you can run several Connection Server instances on the BI platform. Connection Server implements a load-balancing mechanism in this deployment mode. It chooses the best server in terms of resources that can support new client requests.

Load balancing helps to address scalability issues by increasing the number of server instances on the BI platform.

Caution:
This load-balancing mechanism is different from the load balancing implemented by the Central Management Server.

Load balancing is performed at the following levels:

- Client proxies in a 2-tier deployment, on the client tier
- Server bridges in a web tier deployment, on the web tier

Connection Server also provides an application-level lookup mechanism, which helps it to choose the right server instance for a job. Connection Server binds the client to the appropriate server, depending on the type of data source requested by the client and on what network layer and database pair the server supports.

Remember:
Load balancing is performed after the lookup. It is available on all platforms.

4.5.1 Finding the Best Server

Connection Server scalability issues may come from the following:

- The number of opened database connections

  Issues dealing with the number of database connections are already handled by the connection pool mechanism.

- The CPU used by Connection Server

  Connection Server may use a noticeable amount of CPU when converting data or when writing and sending back big response buffers to the client. However, Connection Server is not an intensive CPU consumer.

- The number of jobs running

  The number of jobs that run on a server increases its workload.

- The amount of memory allocated by either Connection Server or the middleware it embeds

  The amount of memory used depends on the SQL queries executed. For example, a single SQL statement against a large table may have a bigger impact on scalability than a series of small SQL queries.

Connection Server implements load balancing with the help of the most relevant indicator of server workload, which is in this case the amount of allocated memory. The less memory a Connection Server server uses, the healthier it is.

Note:
Connection Server may also use the number of jobs as health indicator in future releases.

The following formula calculates the HEALTH of the server:

```
HEALTH = (available memory / max memory) * constant
```

where:

- `max memory` is the maximum amount of allocatable memory, for example 2GB on a 32-bit MS Windows platform
- `available memory` is the difference between `max memory` and the current amount of allocated memory
- `constant` is the maximum HEALTH (set to `10000` for Connection Server).

4.5.2 Load Balancing Logic

1. Each candidate server reports its HEALTH to the load balancer.

   A candidate server is a Connection Server server that supports the requested network layer and database pair. Assuming that all the server instances support the same set of data sources, they are all potential candidates for serving a client request.

2. The load balancer determines which server to send the client request to by identifying and ranking the healthiest servers. It then identifies the best server among those. If the set of servers only contains one server, then this is the target server.

The HEALTH is updated during the lifecycle of the server. HEALTH is recalculated when one of the following operations is performed:

- A job is created or destroyed
- A job is prepared or executed
- A data fetch operation is performed

4.5.3 Backward Compatibility

The load-balancing mechanism is supported on all platforms starting from the SAP BusinessObjects Business Intelligence platform 4.0 Feature Pack 3 release. In the case of incremental deployments or limited upgrades on previous releases, backward compatibility is supported. Any server which fails to report its HEALTH to the load balancer is considered as healthy, that is, at its maximum health.

Note:
Load balancing is also available for some platforms in some Service Packs of previous releases. Contact your SAP representative for more information.

4.6 Memory Allocation

The Native Connectivity Service can use the HOARD memory-allocation functionality on 64-bit MS Windows. HOARD is a scalable and optimized memory allocator for C++ components that allows the service to have better performance, better scalability, and less memory fragmentation.

The Data Access layer provides the following binaries:

- `ConnectionServer.exe`, which is the default binary. HOARD is not enabled.
- `ConnectionServerOptimized.exe`, which is the HOARD-enabled binary

4.6.1 To Enable HOARD

1. Open the CMC.
2. Under "Connectivity Services", stop the Native Connectivity Service hosted by the ConnectionServer server.
3. Locate the Connection Server executable.

It is located in the *bip-install-dir*\win64_x64 directory.

4. Make a backup copy of the ConnectionServer.exe file.
5. Rename ConnectionServerOptimized.exe to ConnectionServer.exe.
6. Restart the service.

You have enabled HOARD in the Native Connectivity Service.

4.6.2 To Disable HOARD

1. Open the CMC.
2. Under "Connectivity Services", stop the Native Connectivity Service hosted by the ConnectionServer server.
3. Restore the default ConnectionServer.exe file.
4. Restart the service from the CMC.

You have disabled HOARD in the Native Connectivity Service.

4.7 Activity in CA Wily Introscope Workstation

Workflow activity involving Connection Server is traced in CA Wily Introscope workstation. Connection Server functions can be analyzed through the different views proposed by the tool, which are the following:

- Dashboard and summary view for overall information
- Trace view, where errors are highlighted and described by messages
- Tree view, where successive function calls of a specific workflow are displayed and time spent on a function is highlighted to track the time-consuming activities easily

For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

Related Topics

- To Configure the Java Virtual Machine for Instrumentation with SAP HANA Connections

Creating a Connection

5.1 Connection Requirements

This section covers the requirements for creating a connection.

- Ensure that your platform conforms to the platforms supported for SAP connections.
- Ensure that the database middleware is installed correctly, and that you can access your database through either your computer or a server.
- Ensure that you have all the information necessary to access your database, for example the database login name and password.
- Install the SAP BusinessObjects solution that you will use, including the appropriate data access driver.
- Check that all of the services have started successfully.
- Refer to the Readme notice that comes with your SAP BusinessObjects solution to check for any configuration changes that your environment and software might require.
- Refer to the current Data Access release notice to check for any configuration changes that could affect your environment.

Note:
You can use the `cscheck` tool to check your infrastructure and determine if it is suitable for use with SAP BusinessObjects applications.

Related Topics

- Checking Connection Configuration

5.2 Checking Connection Configuration

The Connection Server software includes a command line utility that you can use to check your data source connection infrastructure. You can use the `cscheck` tool to check your client middleware and the installed data access drivers at any time.

Note:
The results of all checks apply to your local machine, from which you run the tool.

The `cscheck` tool is installed in the *`boe-install-dir\platform_dir`* where *`boe-install-dir`* is the BOE installation directory and *`platform_dir`* is `win32x_86`, `win64_x64`, and so on.

You run the `cscheck` tool from a command console (DOS or shell). The output is displayed on the screen. You can specify that the output is generated as XML format, or you can suppress output, to use the tool in a script.

The `cscheck` tool can perform the following functions on your local machine:

- Return details of all the connectivities, that is network layers and databases, that the installation can support
- Return details of the data access drivers installed on your local machine
- Return details of the connectivities installed on your local machine
- Check for a valid middleware installation for a supplied network layer and database client
- Check for a valid data access driver installation for a supplied network layer and database client
- Check if a connection can be esbablished to a given database

Related Topics

- Check Tool—Function Overview

5.2.1 Displaying Help on the cscheck Tool

The `cscheck` tool provides functionality to do the following:

- Display general help on the `cscheck` utility
- Display help on each available `cscheck` function

The help can be displayed in any language selected when you installed your SAP BusinessObjects solution.

To display general help on `cscheck`, use the following syntax:

Figure 5-1: Command help syntax

`cscheck --help|h --language|l` { language }

To display help on a function, use the following syntax, where *`functionName`* is the name of the function for which you want help, and *`language`* is the language in which to display the help:

Figure 5-2: Function help syntax

`cscheck --help|h` { functionName } `--language|l` { language }

Example:

To display help in English on the `cscheck` tool, use the following command:

```
cscheck --help
```

To display help in French on the `connectivity` function, use the following command:

```
cscheck --language fr --help connectivity
```

5.2.2 To Run the cscheck Tool

You can run the cscheck tool at any time after you have installed your SAP BusinessObjects solution.

1. Open a command console.
2. Change directory to the path where the tool is installed.
3. Enter `cscheck` with the correct parameters to find the information that you want.
4. Review the returned information.

Related Topics

- Check Tool—Function Overview

5.2.3 Check Tool—Function Overview

From a command console, you use the `cscheck` command with the appropriate function and its arguments to return the results that you want.

`cscheck` commands have the following structure. Some of the parameters are optional.

Figure 5-3: cscheck syntax

`cscheck --language|l` {output language} `--xml|x --mute|m` function name function options

The first part of the command controls the output format:

- *`output language`* or `l` followed by the language specified in ISO639-1 standard. This is optional. The default language is English.
- `--xml` or `x` specifies that the output is in XML format. This is optional. The default output is text displayed on the screen.
- `--mute` specifies that the output is not generated. You would use this switch if you were using the tool in a script that checked the returned status. This is optional. The default is that output is generated.

The remaining part of the command consists of the function and its option arguments.

function name can take the following values. Each function has a short version that you can use in place of the full function name:

- `list` or `lt`
- `driverssearch` or `ds`
- `find` or `fd`
- `middleware` or `mw`
- `accessdriver` or `ad`
- `connectivity` or `ct`
- `ping` or `pg`

Related Topics

- Check Tool—accessdriver
- Check Tool—connectivity
- Check Tool—driverssearch
- Check Tool—find
- Check Tool—list
- Check Tool—middleware
- Check Tool—ping

5.2.4 Check Tool—list

This function returns a list of the supported network layers and database engines. For example you could use it to determine the correct values to use with other check tool functions.

Note:
This function returns the full list of supported data access drivers and middleware, including those that are not necessarily installed on your machine.

Figure 5-4: list syntax

```
cscheck |list| |lt|
```

Example:

The following command lists all network layers and database engines supported by the SAP BusinessObjects solution installed on the current machine.

```
cscheck list
```

The following is an excerpt of the result list:

```
Oracle Client
 Oracle 10
 Oracle 11
Sybase Open Client
 Sybase Adaptive Server 15.5
Informix ODBC Driver
 Informix Dynamic Server 11
Teradata ODBC Driver
 Teradata 12
 Teradata 13
 Teradata 14
ODBC Drivers
 Generic ODBC Datasource
 Generic ODBC3 Datasource
 ...
```

Related Topics

- Check Tool—Function Overview
- Displaying Help on the cscheck Tool

5.2.5 Check Tool—driverssearch

This function returns a list of the installed data access drivers.

Figure 5-5: driverssearch syntax

```
cscheck |driverssearch| |ds|
```

Example:

The following command lists all data access drivers installed on the machine.

```
cscheck driverssearch
```

The following is an excerpt of the result list:

```
This access driver is installed: Oracle OCI access driver
Client layer: Oracle Client
Database engine(s):
 Oracle 10
 Oracle 11
This access driver is installed: Sybase Open Client access driver
Client layer: Sybase Open Client
Database engine(s):
 Sybase Adaptive Server 15.5
This access driver is installed: Informix ODBC access driver
Client layer: Informix ODBC Driver
Database engine(s):
 Informix Dynamic Server 11
This access driver is installed: Teradata ODBC access driver
Client layer: Teradata ODBC Driver
Database engine(s):
 Teradata 12
 Teradata 13
 Teradata 14
...
```

Related Topics

- Check Tool—Function Overview
- Displaying Help on the cscheck Tool

5.2.6 Check Tool—find

This function lists the available connectivity types, that is middleware and database clients, that are available from the local machine. This includes:

- connectivity types available on the local machine
- connectivity types available using the CORBA communication layer
- connectivity types available using the HTTP communication layer
- Java connectivity types available on the local machine

Figure 5-6: find syntax

`cscheck |find| |fd| -m` { Connection Server access mode }

Table 5-1: Function input parameters

Connection Server access mode *(-m)*	The mode in which the client application accesses Connection Server: • `local`: lists connectivity types available on the local machine. • `corba`: lists connectivity types available using CORBA. • `http`: lists connectivity types available using HTTP. • `java`: lists Java connectivity types available on the local machine. • `extended`: lists local, java, and CORBA connectivity types.

Example: Finding local connectivities

The following command returns a list of the data access drivers on the local machine that can be loaded by Connection Server.

```
cscheck find -m local
```

The following is an excerpt of the result list:

```
Local Library Mode
 IBM DB2 Client
  DB2 10 for LUW
  DB2 10 for z/OS
  DB2 for i v6
  DB2 for i v7
  DB2 for z/OS v9
  DB2 UDB for iSeries v5
  DB2 v9
 Essbase Provider
  Hyperion Essbase 7
  Hyperion Essbase 9
 Informix ODBC Driver
  Informix Dynamic Server 11
 ODBC Drivers
  Generic ODBC datasource
  Generic ODBC3 datasource
  MS SQL Server 2008
  MS SQL Server 2012
  Sybase SQL Anywhere 10
  ...
```

Example: Finding CORBA server connectivities

The following command returns a list of the data access drivers available from a CORBA server.

```
cscheck find -m corba
```

Related Topics

- Check Tool—Function Overview
- Displaying Help on the cscheck Tool
- Check Tool—CMS Information

5.2.7 Check Tool—middleware

For a supplied network layer and database client, this function checks for a valid installation of the client middleware. To check both the middleware and data access driver for a supplied network layer and database client, you can use the `connectivity` function.

Figure 5-7: middleware syntax

`cscheck |middleware| |mw| -c` { network layer } `-d` { database client }

Table 5-2: Function input parameters

network layer (`-c`)	The network layer that the database middleware uses, as returned by the `find` function.
database client (`-d`)	The database to check, as returned by the `find` function.

Example:

The following command checks for a valid installation of the Oracle Client 10g middleware on the local machine. It creates an XML file of the output: `c:\result.xml`

```
cscheck --xml middleware -c "Oracle Client" -d "Oracle 9" > c:\result.xml
```

If the middleware is not correctly installed, the result will be the following:

```
Starting to check the middleware component installation...
 Begin AND operator...
  ORACLE_HOME... The environment setting does not exist.
 End AND operator: failure.
The middleware is not correctly installed.
```

Related Topics

- Check Tool—Function Overview
- Displaying Help on the cscheck Tool
- Check Tool—connectivity
- Check Tool—accessdriver

5.2.8 Check Tool—accessdriver

For a supplied network layer and database client, this function checks for a valid data access driver installation. To check both the middleware and data access driver for a supplied network layer and database client, you can use the `connectivity` function.

Figure 5-8: accessdriver syntax

`cscheck |accessdriver| |ad| -c` { network layer } `-d` { database client }

Table 5-3: Function input parameters

network layer (`-c`)	The network layer that the database middleware uses, as returned by the `find` function.
database client (`-d`)	The database to check, as returned by the `find` function.

Example:

The following command checks for a valid installation of an Oracle 10 data access driver, and displays the output in French:

```
cscheck -l fr accessdriver -c "Oracle Client" -d "Oracle 10"
```

If the French language is not installed, the result will be the following:

```
The language specified is not installed. Please use an installed language. English ([en]).
```

Example:

The following command checks for a valid installation of an Oracle 10 data access driver:

```
cscheck ad -c "Oracle Client" -d "Oracle 10"
```

The result is the following:

```
Starting to check the access driver component installation...
 Begin AND operator...
  Config Directory... success.
  %SharedRoot%\ConnectionServer\Network Layers\Oracle OCI... success.
  Directory... success.
  /connectionserver-install-dir/connectionServer//oracle... success.
  Library... success.
  /connectionserver-install-dir/connectionServer//libdbd_oci10.so... success.
  /connectionserver-install-dir/connectionServer//libdbd_oci11.so... success.
  Data File Name... success.
  /connectionserver-install-dir/connectionServer//oracle/oracle.sbo... success.
 End AND operator: success.
The access driver is installed.
```

Related Topics

- Check Tool—Function Overview
- Displaying Help on the cscheck Tool
- Check Tool—list

5.2.9 Check Tool—connectivity

For the supplied network layer and database client, this function checks that both the installed middleware and the data access driver are valid.

You can check each individually using the `middleware` and the `accessdriver` functions. You can use the `ping` function to check if you can connect to a specific database.

Figure 5-9: connectivity syntax

`cscheck |connectivity| |ct| -c` { network layer } `-d` { database client }

Table 5-4: Function input parameters

network layer (`-c`)	The network layer that the database middleware uses, as returned by the `find` function.
database client (`-d`)	The database to check, as returned by the `find` function.

Example:

The following command checks the installed Oracle client middleware, and the Oracle 10 data access driver. The command writes the output to a text file: `c:\result.txt`.

```
cscheck -l en connectivity -c "Oracle Client" -d "Oracle 10">c:\result.txt
```

If the middleware is not correctly installed, the result will be the following:

```
Starting to check the middleware component installation...
 Begin AND operator...
  ORACLE_HOME... The environment setting does not exist.
 End AND operator: failure.
The middleware is not correctly installed.
Starting to check the access driver component installation...
 Begin AND operator...
  Config Directory... success.
  %SharedRoot%\ConnectionServer\Network Layers\Oracle OCI... success.
  Directory... success.
  /connectionserver-install-dir/connectionServer//oracle... success.
  Library... success.
  /connectionserver-install-dir/connectionServer//libdbd_oci10.so... success.
  /connectionserver-install-dir/connectionServer//libdbd_oci11.so... success.
  Data File Name... success.
  /connectionserver-install-dir/connectionServer//oracle/oracle.sbo... success.
 End AND operator: success.
The access driver is installed.
```

Related Topics

- Check Tool—Function Overview

- Displaying Help on the cscheck Tool
- Check Tool—find
- Check Tool—accessdriver
- Check Tool—middleware
- Check Tool—ping

5.2.10 Check Tool—ping

This function attempts to access a given database using the supplied details.

Figure 5-10: ping syntax

`cscheck ping|pg| -m` { Connection Server access mode } `-c` { network layer } `-d` { database client } `-u` { user name } `-p` { password } `-s` { data source } `-t` { database } `-r` { host name } `-j` { PID }

Table 5-5: Function input parameters

Connection Server access mode *(-m)*	The mode in which the client application accesses Connection Server: • `local`: Connection Server is running on the local machine. • `corba`: Connection Server is running on a CORBA server. • `http`: Connection Server is running on a HTTP server. • `java`: Connection Server uses a Java data access driver on the local machine.
network layer *(-c)*	The database middleware for the connection to check, as returned by the `find` function.
database client *(-d)*	The database type, as returned by the `find` function.
user name *(-u)*	A valid user name for the database.
password *(-p)*	The password for the user name.

data source *(-s)*	The server on which the database is running.
database *(-t)*	The database server.
host name *(-r)*	For CORBA mode, the computer hosting Connection Server.
PID *(-i)*	For CORBA mode, the process number of the Connection Server to ping through.

Example: **Pinging an Oracle database**

The following command checks access for:

- Connection Server access mode: `local`, that is, the database runs on the local machine.
- Network layer: `Oracle Client`
- Database: `Oracle 10g`
- Data source: `Harlaxton`
- User name: `efashion`
- Password: `X2345`

```
cscheck ping -m local -c "Oracle Client" -d "Oracle 10" -u "efashion" -p
"X2345" -s "Harlaxton"
```

Example: **Pinging a Sybase database using CORBA**

The following command checks access for:

- Connection Server access mode: `CORBA`, that is, Connection Server runs on a CORBA server.
- Network layer: `Sybase`
- User name: `syadmin`
- Password: `password`
- Data source: `Sybase Adaptive Server 15`
- Database: `SY1`
- Database host: `sybasehost`
- Process ID: `456`

```
cscheck ping -m corba -c "Sybase Open Client" -d syb15 -u "syadmin" -p
"password" -s "Sybase Adaptive Server 15.5" -t "SY1" -r "sybasehost" -i 456
```

Related Topics

- Check Tool—Function Overview
- Check Tool—find
- Check Tool—CMS Information

5.2.11 Check Tool—CMS Information

The following parameters allow you to specify the CMS information that you can use with the `find` or `ping` function of the `cscheck` tool.

Figure 5-11: CMS syntax

`cscheck --ce_cluster` { CMS server } `--ce_user` { User name } `--ce_pass` { Password } `--ce_auth` { Authentication } `find -m corba`

Table 5-6: Function Input Parameters

CMS server (*`--ce_cluster`*)	The host and port to access the CMS.
User name (*`--ce_user`*)	The user name to access the CMS.
Password (*`--ce_pass`*)	The password to access the CMS.
Authentication (*`--ce_auth`*)	The method used to authenticate the user's login credentials when accessing the CMS.

Example:

The following command returns a list of connections that are stored on the CMS and available using CORBA.

```
cscheck --ce_cluster localhost --ce_user Administrator --ce_pass Password1
--ce_auth SecEnterprise find -m corba
```

Remember:

The parameters must be placed before the `find` or `ping` function in the command line. They all are mandatory.

Related Topics

- Check Tool—find
- Check Tool—ping

5.3 Creating JDBC Connections

A set of data access drivers are installed when you install the BI platform. You can use these data access drivers to create connections to databases. They are located in the *connectionserver-install-dir*`\connectionServer\drivers\java` directory.

Note:
JDBC connectivity is available for SAP BusinessObjects Enterprise XI 3.0 and higher. Web Intelligence Rich Client supports JDBC connectivity in 3-tier mode in the SAP BusinessObjects Business Intelligence platform 4.0 and higher.

SAP BusinessObjects software also includes configuration files for using JDBC drivers to access your databases. To use these drivers, you do the following:

1. Obtain the java driver software from your database supplier.
2. Specify JAR file paths by one of the following ways:
   - Set the `ClassPath` element in the SBO configuration file of the data access driver with the fully qualified path of the JAR file.
   - Store JAR files into directories you create from the `Extensions` parameter values of the SBO file.

You can use simultaneously these two ways of specifying JAR file paths. However, JAR files specified in the SBO file take precedence over JAR files stored in your own directories.

Note:
SAP Visual Intelligence allows users to select JAR files directly from the connection creation box of the application. See *SAP Visual Intelligence User Guide* for information.

For an up-to-date list of supported JDBC drivers, check the SAP Service Marketplace at http://service.sap.com/bosap-support, or contact your SAP representative.

Note:
The Data Access layer provides the Generic JDBC connectivity to create a connection to a data source that the BI platform does not support explicitly.

Related Topics

- To Create a JDBC Connection with the SBO File
- To Create a JDBC Connection with Extensions
- To Create a Generic JDBC Connection
- JDBC Driver Properties

5.3.1 To Create a JDBC Connection with the SBO File

- Obtain the necessary JDBC driver software for the database, and copy the files to your system. These files are available from the database vendor. The driver software consists typically of one or more `.jar` files. Note the installation path details for these files.
- Ensure that you have the database access details to hand, for example the login and password details.

1. Navigate to the directory that contains the SBO file you want to use.

   For example, on Microsoft Windows, the JDBC configuration files are located in the *`connection server-install-dir`*`\connectionServer\jdbc` directory.
2. Use an XML editor to open the SBO file for editing.
3. Add the required `.jar` file details to the `ClassPath` area. Include the fully qualified path names when specifying these files, for example:

   ```
   <Path>C:\\JDBC Drivers\\MSSQLSERVER2008\\msutil.jar</Path>
   ```

   Note:
   These files need to be installed on the machine running the application. Make sure the JDBC driver path is correct.
4. Locate the `Driver Capabilities` parameter, and check that it is set to either `Procedure`, `Queries`, or both.

   Note:
   In the last case, settings are separated by a comma.

   Caution:
   If it is not set to one of these values, the JDBC driver is unavailable from the connection wizard.
5. Save and close the SBO file.
6. Run the connection wizard.

   The JDBC driver that you have configured appears in the list of available connections.
7. Select the JDBC driver and use the wizard to configure the connection.

When you complete this task, the connection is available for use.

Related Topics

- Connection Requirements
- Data Access Configuration Files

5.3.2 JDBC SBO Example File Structure

This shows an example of the section of the `sqlsrv.sbo` file that you need to modify. This SBO file is for Microsoft SQL Server 2008.

```
<DataBase Active="Yes" Name="MS SQL Server 2008">
 ...
 <JDBCDriver>
  <ClassPath>
   <Path>C:\\JDBC Drivers\\MSSQLSERVER2008\\msbase.jar</Path>
   <Path>C:\JDBC Drivers\MSSQLSERVER2008\\msutil.jar</Path>
   <Path>C:\\JDBC Drivers\\MSSQLSERVER2008\\mssqlserver.jar</Path>
  </ClassPath>
 ...
 </JDBCDriver>
 ...
</DataBase>
```

5.3.3 To Create a JDBC Connection with Extensions

- Obtain the necessary JDBC driver software for the database, and copy the files to your system. These files are available from the database vendor. The driver software consists typically of one or more JAR files.
- Ensure that you have the database access details to hand, for example the login and password details.

1. Go to the next section of the guide to find the `Extensions` parameter values.
2. Use one or more of the `Extensions` parameter values to create your own driver directories.

   For example, `Extensions` parameter values for MS SQL Server 2008 middleware are `sqlsrv2008`, `sqlsrv` and `jdbc` in the `sqlsrv.sbo` file. You can create any of the following directories:

   - *connectionserver-install-dir*\connectionServer\jdbc\drivers\sqlsrv2008
   - *connectionserver-install-dir*\connectionServer\jdbc\drivers\sqlsrv
   - *connectionserver-install-dir*\connectionServer\jdbc\drivers\jdbc
3. Copy the JAR files into the directories of your choice.
4. Run the connection wizard.

   The JDBC driver that you have configured appears in the list of available connections.
5. Select the JDBC driver and use the wizard to configure the connection.

To load JDBC drivers, Connection Server searches for JAR files in each directory from the most specific to the least specific until it finds them. The connection is then available for use.

Example:

For example, if you store JAR files in *connectionserver-install-dir*\connectionServer\jdbc\drivers\sqlsrv only, then Connection Server first searches for drivers in sqlsrv2008 directory, finds it empty, then it searches in sqlsrv directory, finds the JAR files and loads the driver.

Note:
Because sqlsrv is Extensions value of all MS SQL Server target databases, JAR files specified in this directory are loaded for all MS SQL Server databases.

Related Topics

- To Create a JDBC Connection with the SBO File
- Data Access Configuration Files

5.3.4 To Create a Generic JDBC Connection

- Obtain the necessary JDBC driver software for the database. The .jar file need to be installed on the machine running the SAP BusinessObjects application.
- Ensure that you have the database access details ready, for example the login and password details.

1. Navigate to the directory that contains the jdbc.sbo and jdbc.prm files.

   For example, on Microsoft Windows, the configuration files are located in the *connectionserver-install-dir*\connectionServer\jdbc directory.
2. Copy the required .jar file to the *connectionserver-install-dir*\connectionServer\jdbc\drivers\jdbc directory.

   You have to create the directory if it does not exist.
3. Run the connection wizard.

   The JDBC driver appears in the list of available connections under Generic.
4. Select the JDBC driver and use the wizard to configure the connection with the following details:
   - JDBC URL
   - JDBC class
   - user name
   - password

When you complete this task, the connection to the data source is available for use through JDBC.

Note:
The jdbc.prm file only contains information about generic functions of a JDBC database. You can add or update any information specific to a custom database in the file. The modifications to jdbc.prm file apply to all generic JDBC connections that are created or to be created.

Related Topics

- Data Access Configuration Files
- About PRM Files

5.3.5 JAR File Location Reference

The following table describes the names of the folders where you must place middleware JAR files to enable JDBC connections. The first column lists the database vendor names as shown in the connection wizard. The second column lists the names of the databases that support JDBC connections, as described in the `DataBase` section of the corresponding SBO file. The third column lists the names of the folders you can create, as described by the `Extensions` parameter values of the SBO file.

The `Extensions` parameter is a child element of the `DataBase` element that corresponds to the targeted database middleware in SBO files. If it is not under `DataBase`, it means the parameter value is valid for all middleware configured in the SBO file. Then refer to the `Defaults` section of the file. On MS Windows, JDBC configuration files are located in the *`connectionserver-install-dir`*`\connectionServer\jdbc` directory.

Note:
The Data Federator JDBC drivers are installed as part of the BI platform in the *`connectionserver-install-dir`*`\connectionServer\jdbc\drivers\datafederator` directory. The SAP HANA 1.0 SPS 07 JDBC drivers are installed in the *`connectionserver-install-dir`*`\connectionServer\jdbc\drivers\newdb` directory. Consequently, you do not require to perform any additional configuration to create a connection to either Data Federator XI 3.0 Query Server or SAP HANA database.

Vendor	Database	Extensions Parameter Values
Apache	Derby 10 Embedded	derby10, derby, jdbc
	Apache Hadoop Hive 0.7	apache, hive, hive07, jdbc
	Apache Hadoop Hive 0.8	apache, hive, hive08, jdbc
	Apache Hadoop Hive 0.9	apache, hive, hive09, jdbc
	Apache Hadoop Hive 0.10	apache, hive, hive010, jdbc
	Amazon EMR Hive 0.7	amazon, hive, emrhive07, jdbc
	Amazon EMR Hive 0.8	amazon, hive, emrhive08, jdbc
Greenplum	PostgreSQL 8	postgresql8, postgresql, jdbc
	GreenPlum4, PostgreSQL 9	postgresql9, postgresql, jdbc
Hewlett Packard	HP Neoview	neoview, jdbc
	HP Vertica 6.1	vertica, jdbc
HSQLDB	HSQLDB 1.8 Embedded	hsqldb18, hsqldb, jdbc
IBM	DB2 v9	db2v9, db2udb, db2, jdbc
	DB2 10 for z/OS	db2mvs10, db2mvs, db2, jdbc
	DB2 10 for LUW	db2v10, db2udb, db2, jdbc
	DB2 10.5 for LUW	db2v10_5, db2udb, db2, jdbc
	Informix Dynamic Server 11	ids11, informix, jdbc
Ingres	Ingres Database 9	ingres9, ingres, jdbc

Vendor	Database	Extensions Parameter Values
Microsoft	MS SQL Server 2008	sqlsrv2008, sqlsrv, jdbc
	MS SQL Server 2012	sqlsrv2012, sqlsrv, jdbc
Netezza	Netezza Server 4	netezza4, netezza, jdbc
	Netezza Server 5	netezza5, netezza, jdbc
	Netezza Server 6	netezza6, netezza, jdbc
	Netezza Server 7	netezza7, netezza, jdbc
Oracle	MySQL 5	mysql5, mysql, jdbc
	Oracle 10	oracle10, oracle, jdbc
	Oracle 11	oracle11, oracle, jdbc
	Oracle Exadata	oracleexadata, oracle11, oracle, jdbc
SAP	Data Federator XI R3	datafederator3, datafederator, jdbc
	Data Federator XI R4	datafederator4, datafederator, jdbc
	MaxDB 7.7	maxdb7.7, maxdb, jdbc
	SAP HANA database 1.0	newdb, jdbc

Vendor	Database	Extensions Parameter Values
Sybase	Sybase Adaptive Server Enterprise 15.5	sybase15, sybase, jdbc
	Sybase IQ 15	iq15, asiq, jdbc
	Sybase IQ 16	iq16, asiq, jdbc
	Sybase SQL Anywhere 11	ssa11, ssa, jdbc
	Sybase SQL Anywhere 12	ssa12, ssa, jdbc
	Sybase SQL Anywhere 16	ssa16, ssa, jdbc
Teradata	Teradata 12	teradata12, teradata, jdbc
	Teradata 13	teradata13, teradata, jdbc
	Teradata 14	teradata14, teradata, jdbc

Related Topics

- SAP HANA Connections
- About Data Federator XI 3.0 Query Server Connections

5.3.6 To Find the Version of a JDBC Driver

This section describes how to find the version of the driver in your JDBC connection.

You can find the version of the JDBC driver used in the `META-INF/MANIFEST.MF` file contained in the driver JAR file.

- Do one of the following:

Option	Description
To extract the file by using WinRAR	a. Launch the WinRAR application. b. Drag the JAR file and drop it into the application window. c. Expand the **META-INF** folder. d. Drag the `MANIFEST.MF` file from WinRAR and drop it into a local folder. e. Open the file and locate the `Bundle-Version` value.

To extract the file from a command prompt	a. Open a command prompt.
	b. Go to the folder where you should extract the file:
	`cd driver_path`
	c. Run the following command:
	`JAR_path\jar.exe -xf driver_path\driver_name.jar META-INF/MANIFEST.MF`
	d. Open the file and locate the `Bundle-Version` value.
	For example, run the following commands to retrieve the version of the SAP HANA JDBC driver:
	`cd C:\Program Files (x86)\SAP BusinessObjects\SAP BusinessObjects Enterprise XI 4.0\dataAccess\connectionServer\jdbc\drivers\newdb`
	`"C:\Program Files (x86)\SAP BusinessObjects\SAP BusinessObjects Enterprise XI 4.0\win64_x64\jdk\bin\jar.exe" -xf ngdbc.jar META-INF/MANIFEST.MF`
	Note:
	Put the path of the JAR file into double quotes if it contains spaces.

5.4 Creating JavaBean Connections

A data access driver that uses a JavaBean is installed when you install the BI platform. It is located in the *connectionserver-install-dir*\connectionServer\drivers\java directory.

Developers can also create JavaBeans that provide access to data sources. You can create connections using these JavaBeans. In order to create a JavaBeans connection, the developers who create the JavaBean will supply the following information:

- The required JAR files
- Any other files that the JavaBean requires
- Any specific configuration details that the JavaBean driver requires

As for JDBC connections, you can also create a JavaBean connection by using the `Extensions` functionality.

Note:
Within a JavaBean driver, data-retrieval procedures are configured as stored procedures. When creating a JavaBean connection from the connection wizard, you must check **Filter Stored Procedures Network Layers** on **Database Middleware Selection** screen. If you do not, the connection wizard does not display the JavaBean drivers that are available.

Related Topics

- Creating JDBC Connections
- To Create a JavaBean Connection
- To Create a JavaBean Connection with Extensions

5.4.1 To Create a JavaBean Connection

1. Use an XML editor to open the `javabean.sbo` file for editing.

   For example, on Microsoft Windows, the configuration files are located in the *`connectionserver-install-dir`*`\connectionServer\javabean` directory.

2. Add the required `.jar` file details to the `ClassPath` area. Include the fully qualified path names when specifying these files.

   Note:
   These files need to be installed on the machine running the SAP BusinessObjects application.

   Refer to the information on the `javabean.sbo` example file structure for details.

3. Save and close the file.
4. Perform any other configuration tasks specified by the JavaBeans developer.
5. Run the connection wizard.

   The JavaBeans data source that you have configured should appear in the list of available connections.

6. Select the JavaBeans data source and use the wizard to configure the connection.

When you complete this task, the connection is available for use with the application.

Related Topics

- Data Access Configuration Files

5.4.2 JavaBean SBO Example File Structure

This section contains an example of a JavaBeans SBO file.

```
<DataBase Active="Yes" Name="Excel Spreadsheet">
  <JavaBean>
    <ClassPath>
      <Path>$ROOT$\\beans\\bean_excel.jar</Path>
    </ClassPath>
    <Parameter Name="JavaBean Class">com.businessobjects.beans.excel.Excel</Parameter>
    <Parameter Name="URL Format">$DATASOURCE$</Parameter>
  </JavaBean>
  <Parameter Name="Family">Java Beans</Parameter>
    <Parameter Name="Description File">bean_excel</Parameter>
    <Parameter Name="Authentication Mode">Bypass</Parameter>
    <Parameter Name="Extensions">bean_excel,javabean</Parameter>
  </DataBase>
</DataBases>
```

5.4.3 To Create a JavaBean Connection with Extensions

1. Navigate to the directory that contains the `javabean.sbo` file.

   For example, on Microsoft Windows, the file is located in the *`connectionserver-install-dir`*`\connectionServer\javabean` directory.
2. Open the SBO file for viewing.
3. Locate `<Parameter Name="Extensions">` element in the `Defaults` section.

   Note:
   If you develop a JavaBean driver using the DDK, locate the `Extensions` parameter in the `<DataBase Active="Yes" Name="Excel Spreadsheet">` element.
4. Use one or more of the `Extensions` parameter values to create your own driver directories.

   For example, `Extensions` parameter value is `javabean` in the `Defaults` section of the file. You can then create the *`connectionserver-install-dir`*`\connectionServer\javabean\drivers\javabean` directory.
5. Copy the JAR files into the directories of your choice.
6. Close the SBO file.
7. Run the connection wizard.

   The JavaBean driver that you have configured appears in the list of available connections.
8. Select the JavaBean driver and use the wizard to configure the connection.

To load JavaBean drivers, Connection Server searches for JAR files in each directory from the most specific to the least specific until it finds them. The connection is then available for use.

Related Topics

- To Create a JavaBean Connection
- Data Access Configuration Files

5.5 Creating ODBC Connections

A set of data access drivers are installed when you install the BI platform. You can use these data access drivers to create connections to databases. They are located in the *connectionserver-install-dir*\connectionServer\drivers\lib32 or *connectionserver-install-dir*\connectionServer\drivers\lib64 directory.

SAP BusinessObjects software also includes configuration files for using ODBC drivers to access your databases. To use these drivers, you do the following:

1. Obtain the ODBC driver software from your database supplier.
2. Modify the supplied configuration files.

Most of ODBC drivers are available in Unicode and non-Unicode.

For an up-to-date list of supported ODBC drivers, check the SAP Service Marketplace at http://service.sap.com/bosap-support, or contact your SAP representative.

Note:
The Data Access layer provides the Generic ODBC connectivity to create a connection to a data source that the BI platform does not support explicitly.

About Generic ODBC Connections on UNIX

The Microsoft Windows operating system comes with a standard ODBC driver manager. In contrast, UNIX does not provide any standard way to manage drivers. SAP BusinessObjects software allows you to use either DataDirect or unixODBC driver managers for generic ODBC connections on UNIX.

Before creating a generic ODBC connection on UNIX for a specific database, you must identify the following:

- The version of the ODBC driver
- If the driver is able to work with DataDirect driver manager or unixODBC

You then modify the supplied configuration files and configure the relevant data source to enable the connection.

Related Topics

- To Create a Generic ODBC Connection
- To Create a Generic ODBC3 Connection

5.5.1 To Create a Generic ODBC Connection

The following procedure shows how to configure a generic ODBC connection to a database with the following assumptions:

- The driver is ODBC2
- The driver supports the unixODBC driver manager

1. Navigate to the directory that contains the `odbc.sbo` and `odbc.prm` files.

   The configuration files are located in the *`connectionserver-install-dir`*`/connectionServer/odbc` directory.

2. Use an XML editor to open the `odbc.sbo` file for editing.
3. Locate the following section:

```
<DataBases>
 <DataBase Active="Yes" Name="Generic ODBC datasource">
 <Libraries>
  ...
  <Library Platform="Unix">dbd_wddodbc</Library>
  <Library Platform="Unix">dbd_ddodbc</Library>
  <!--Library Platform="Unix">dbd_wuxodbc</Library-->
  <!--Library Platform="Unix">dbd_uxodbc</Library-->
 </Libraries>
 <Parameter Name="Charset Table" Platform="Unix">datadirect</Parameter>
...
</DataBases>
```

4. Comment out the first two rows for DataDirect and uncomment one of the next two rows. Make sure you put the row you uncomment on top of the section, so that it can be read first. Comment out the `<Parameter Name="CharSet Table" Platform="Unix">` element.

   Note:

   - `dbd_wddodbc` specifies the DataDirect-based ODBC2 Unicode driver
   - `dbd_ddodbc` specifies the DataDirect-based ODBC2 non-Unicode driver
   - `dbd_wuxodbc` specifies the unixODBC-based ODBC2 Unicode driver
   - `dbd_uxodbc` specifies the unixODBC-based ODBC2 non-Unicode driver

5. Save and close the `odbc.sbo` file.
6. Use an XML editor to open the `odbc.prm` file for editing.
7. Add or update any information specific to the database.

   Note:
   The `odbc.prm` file may not contain information about specific functions of the database.

8. Save and close the `odbc.prm` file.
9. Install the relevant ODBC drivers on your UNIX machine.
10. Configure the data source by editing the `odbc.ini` file.
11. Save and close the `odbc.ini` file.
12. Run the connection wizard.

The ODBC driver that you have configured appears in the list of available connections under Generic.

13. Select the ODBC driver and use the wizard to configure the connection.

When you complete this task, the connection to the data source is available for use through generic ODBC with unixODBC driver manager.

Related Topics

- Data Access Configuration Files

5.5.2 To Create a Generic ODBC3 Connection

The following procedure shows how to configure a generic ODBC connection to a database with the following assumptions:

- The driver is ODBC3
- The driver supports the unixODBC driver manager

1. Navigate to the directory that contains the `odbc.sbo` and `odbc.prm` files.

   The configuration files are located in the *`connectionserver-install-dir`*`/connectionServer/odbc` directory.

2. Use an XML editor to open the `odbc.sbo` file for editing.
3. Locate the following section:

```
<DataBases>
 <DataBase Active="Yes" Name="Generic ODBC3 datasource">
 <Libraries>
  ...
  <Library Platform="Unix">dbd_wddodbc3</Library>
  <Library Platform="Unix">dbd_ddodbc3</Library>
  <!--Library Platform="Unix">dbd_wuxodbc3</Library-->
  <!--Library Platform="Unix">dbd_uxodbc3</Library-->
  <!--Library Platform="Unix64">dbd_wux32odbc3</Library-->
  <!--Library Platform="Unix64">dbd_ux32odbc3</Library-->
 </Libraries>
 <Parameter Name="Charset Table" Platform="Unix">datadirect</Parameter>
...
</DataBases>
```

4. Comment the first two rows for DataDirect and uncomment one of the next four rows. Make sure you put the row you uncomment on top of the section, so that it can be read first. Comment out the `<Parameter name="CharSet Table" Platform="Unix">` element.

   Note:

   - `dbd_wddodbc3` specifies the DataDirect-based ODBC3 Unicode driver
   - `dbd_ddodbc3` specifies the DataDirect-based ODBC3 non-Unicode driver
   - `dbd_wuxodbc3` specifies the unixODBC-based ODBC3 Unicode driver
   - `dbd_uxodbc3` specifies the unixODBC-based ODBC3 non-Unicode driver
   - `dbd_wux32odbc3` specifies the unixODBC-based ODBC3 Unicode driver with 32-bit API rather than 64-bit (available only on 64-bit platforms)

- `dbd_ux32odbc3` specifies the unixODBC-based ODBC3 non-Unicode driver with 32-bit API rather than 64-bit (available only on 64-bit platforms)

5. Save and close the `odbc.sbo` file.
6. Use an XML editor to open the `odbc.prm` file for editing.
7. Add or update any information specific to the database.

   Note:
   The `odbc.prm` file may not contain information about specific functions of the database.

8. Save and close the `odbc.prm` file.
9. Install the relevant ODBC drivers on your UNIX machine.
10. Configure the data source by editing the `odbc.ini` file.
11. Save and close the `odbc.ini` file.
12. Run the connection wizard.
    The ODBC driver that you have configured appears in the list of available connections under Generic.
13. Select the ODBC driver and use the wizard to configure the connection.

When you complete this task, the connection to the data source is available for use through ODBC3 with unixODBC driver manager.

Related Topics

- Data Access Configuration Files

5.6 To Set the Locale in a Multiple Data Sources Workflow

In the designing and reporting tools, queries to BigDecimal columns of a multisource-enabled universe that is using a native connection may return an error (`java.lang.NumberFormatException`). To avoid this problem, add the `Locale` parameter with the appropriate language and country values to the SBO file of the database driver on the BIP platform server.

The procedure uses the French locales as examples.

1. Stop the Adaptive Connectivity Service.
2. Open the SBO file for editing.
   It is located in the *`bip-install-dir`*`\dataAccess\connectionServer\`*`connection_type`* directory, where *`connection_type`* is for example `db2`, `odbc`, `oracle`, or `sybase`.
3. Add one of the following lines under the appropriate `Database` section:
   - `<Parameter Name="Locale">fr</Parameter>`
   - `<Parameter Name="Locale">fr_FR</Parameter>`
4. Save the file.
5. Restart the service and application.

When you complete this task, native connections in a multisource workflow run properly.

Data Access Driver Reference

6.1 Data Access Drivers

This chapter provides detailed information on some data access drivers, which enable connections to data sources using OData or XML documents, and specific databases like ABAP-based SAP ERP systems. It mainly describes mapping rules between nontrivial data models and the relational model used in the Data Access layer.

This chapter only relates to connections to be created in the information design tool.

Related Topics

- CSV OpenDriver
- OData Driver
- SAP ERP Driver
- XML Driver
- Web Service Driver

6.2 CSV OpenDriver

The Data Access layer allows the BI platform to connect to Comma-Separated Value (CSV) files through BusinessObjects OpenConnectivity network layer. It provides a data access driver called CSV OpenDriver to access the CSV files.

The connection wizard provides a workflow to enter the necessary information to connect to CSV files. CSV data sources show up under `Text Files` DBMS and `BusinessObjects OpenConnectivity` NetworkLayer.

Backward compatibility

Backward compatibility between SAP BusinessObjects Enterprise XI 3.x and SAP BusinessObjects Business Intelligence platform 4.1 releases is only supported for documents created using an ODBC connectivity.

When a report user refreshes in Web Intelligence Rich Client a document based on a BusinessObjects OpenConnectivity connectivity in 3-tier mode and created by using the XI 3.x release, the following exception is raised:

```
Database error: (CS) "Specified network layer is invalid : BO OC". (IES 10901) (WIS 10901)
```

Since JDBC connectivity in 3-tier mode is not supported in XI 3.x releases, BusinessObjects OpenConnectivity is not recognized as a network layer when the user tries to refresh a document in the 4.1 release.

6.2.1 CSV OpenDriver Capabilities

Since text files do not provide any database operations, the data access driver implements the following querying capabilities:

- Basic operations for SELECT statements (WHERE, ORDER BY, GROUP BY, and AS)
- Basic operators within the WHERE clause (=, <>, <, <=, >, >=, BETWEEN, NOT BETWEEN, LIKE, NOT LIKE, IS NULL, IS NOT NULL, AND, OR)
- Use of wildcards ("?") within the WHERE clause
- DISTINCT clause within the SELECT statement
- UNION and UNION ALL

The data access driver also supports the `MIN`, `MAX`, `AVG`, `SUM`, and `COUNT` grouping functions in SELECT statements.

Driver Restrictions

The CSV OpenDriver does not implement any functionality for altering the state of the CSV file, such as INSERT, UPDATE and ALTER. The data access driver also does not support indexes or JOIN operations on CSV files.

Related Topics

- Local Disk Used as a Cache for Sorting Operations

  The CSV OpenDriver and the OData, SAP ERP, XML and Web Service drivers can use the local disk as a cache for sorted rows.

6.2.2 CSV OpenDriver - Table Mapping

The data access driver exposes CSV files as tables to the information design tool.

You can connect to multiple files if the **Filepath or Pattern** parameter of the connection wizard is set to a pattern, for example `report_*.csv`, which is matched by files such as `report_2010-09-22.csv` and `report_2010-09-21.csv`.

Remember:
You cannot connect to multiples files located on an HTTP server. The HTTP protocol does not support the use of wildcards in patterns.

The driver applies one of the following processes:

- If the **Merge Files into One** parameter is selected in the connection wizard, all CSV files are mapped to one single table. It is assumed they all have the same schema. This is the default behavior of the data access driver.
- If the parameter is not selected, each CSV file is mapped to a different table.

Mapping All Files to One Table

The table name is the name of the pattern, for example `report_*.csv`. The table has an additional column named `sourcefile`, which contains the name of the source file for each table row.

If you use an SQLDDL file to provide the schema, the table name must also be the pattern.

If the files have different column names, the column names of the resulting table are the ones of the first file analyzed by the driver, knowing that the files are analyzed in alphabetical order.

Mapping One File to One Table

The name of each table corresponds to the file name of its data source.

If you use an SQLDDL file to provide the schema, the name of the table in the `CREATE` statement must be the file name.

Independently of the parameter value, the file path is mapped to the qualifier of the table.

Example: Column Names

If the `report_1.csv` file has the `col1`, ..., and `col10` columns, while the `report_2.csv` file has the `column1`, ..., and `column10` columns, then the columns of the resulting table are `col1`, ..., and `col10`.

Example: Qualifier

If `C:\reports\report_1.csv` is the data source, then the qualifier is `C:\reports\` and the table name is `report_1.csv`.

Related Topics

- CSV File Location
- CSV Schema Detection

6.2.3 CSV File Location

CSV files used as data sources can be either local or remote. For files on remote systems, the HTTP, FTP, and SMB (also known as CIFS) protocols are supported. The present release also supports the HTTPS and FTPS encrypted protocols based on one certificate. CSV files can be located on the Central Management Server (CMS) of the BI platform.

Note:
SMB is the standard file sharing protocol on MS Windows. Since the jCIFS library provides access to the protocol, you must install the jCIFS JAR file in the Connection Server directory, that is *`connection server-install-dir`*`\connectionServer\sharedLibraries\jcifs`. The version to install is 1.3.15. It is available at http://jcifs.samba.org/.

You can set whether the data source is local or remote through the **Location Type** parameter of the connection wizard. If the data source is remote, you must set the protocol used through the **Protocol** parameter of the wizard. The **Protocol** value must match the protocol specified in the **Filepath or Pattern** and **Schema File** parameter values.

Example: CSV File Paths

- `C:\csv\report.csv` or `file:///C:/csv/report.csv` for a local file
- `http://server:8080/path/report.csv` for a file accessible through HTTP
- `ftp://server/path/` for files accessible through FTP
- `smb://server:port/myshare/mydirectory` for files accessible through SMB

6.2.4 CSV Schema Detection

The Data Access layer provides the following methods for the schema detection of a CSV file:

- No detection
- Automatic detection
- Use of a Data Definition Language (DDL) file
- Use of an SQLDDL file

You choose the method to apply to the files using the **Schema Detection** parameter in the connection wizard. You can provide the schema by using schema files (DDL or SQLDDL). This can be helpful for large and complex files.

Files are analyzed in alphabetical order.

Note:
The first lines of a CSV file may contain comments and can be skipped using the **Number of Comment Lines in the Beginning** parameter in the wizard.

No Detection

The data access driver skips comment lines, analyzes the first line, and determines the number of columns, but not the column types. All values are considered as strings, and column size is set to 255 characters, which corresponds to the standard length of the `VARCHAR` data type. A value that exceeds 255 is truncated. The columns can contain null values.

Automatic Detection

Remember:
You cannot apply this method to CSV files with fixed-length columns.

The data access driver analyzes well-formed files and obtains the following information:

- column names

  In most cases, the first line contains column names. Otherwise, the CSV OpenDriver generates column names in the format `column_0`, `column_1`, ..., `column_n`.

- Column types

  The driver finds column types by using predefined patterns and user settings for numerical, date and time datatypes. If a column contains values of different types, the driver then considers column values as strings.

- Column sizes

  The column size depends on its type. For numerical values, the column size is the length of the highest value in the range of the type. For the other values, this is the length of the longest string value found during the detection.

  Caution:
  Only **Probe Rows** set to **all** allows the driver to find the longest string.

- Column nullability

  Nullability is true if there is a null value in the column, false if all values are filled.

Unless you select all rows, you choose the number of lines the driver has to analyze by setting the **Number of Probe Rows** parameter in the connection wizard.

See the *Information Design Tool User Guide* for recommendations.

Use of DDL Files

The data access driver first analyzes the DDL file to convert SQL data types into driver data types.

The DDL file follows this pattern:

```
ColumnName[:ColumnType[(Length)]];
```

For example:

```
col1:VARCHAR(20);
col2:DATE;
col3:INTEGER;
col4:INTEGER;
```

The DDL file can only define one table schema. The driver assigns the same schema to every CSV file used as data source.

Use of SQLDDL Files

The data access driver first analyzes the SQLDDL file to convert SQL data types into driver data types.

The SQLDDL file follows this pattern:

```
CREATE TABLE <Filename> (
 (<ColumnName> <ColumnType> [NOT] NULL,)*
 (<ColumnName> <ColumnType> [NOT] NULL)
);
```

For example:

```
CREATE TABLE Clients (
 id INTEGER NOT NULL,
 name CHAR(20) NULL,
 date DATE NULL,
PRIMARY KEY (id) )
```

The SQLDDL file can define several table schemas. The driver can assign a schema to several tables.

The data access driver analyzes `CREATE` statements and ignores the other ones, if any.

If the DDL or SQLDDL file does not define a column type, the driver then considers column values as strings of 255 characters. A value that exceeds 255 is truncated. If the DDL or SQLDDL file provides a type but not its length, the driver then uses standard lengths, such as 10 for integers. Precision and scale are mandatory for the `DECIMAL` data type.

6.3 OData Driver

The Data Access layer allows the BI platform to connect to data sources that use the Open Data (OData) protocol. It provides a data access driver called OData driver to access online data sources on the Internet or intranets. The OData driver supports the OData 2.0 protocol, whose schema is described with the Conceptual Schema Definition Language (CSDL) 2.0.

The OData driver allows the BI platform to connect to the OData services that SAP NetWeaver Gateway 2.0 exposes. SAP NetWeaver Gateway 2.0 allows application users to access data from SAP Business Suite systems such as ERP and CRM through HTTP.

The connection wizard provides a workflow to enter the necessary information to connect to OData data sources. OData data sources show up under `Generic OData 2.0` DBMS and `OData Connector` NetworkLayer.

You can find the driver configuration files in the *`connectionserver-install-dir`*`\connectionServer\odata` directory. The `odata.prm` file lists the data access driver capabilities in terms of database operations and functions.

6.3.1 OData Driver Capabilities

The data access driver supports the following querying capabilities:

- Basic database operations (SELECT, WHERE, ORDER BY, structured joins)
- Basic filtering operators within the WHERE clause (=, !=, <, <=, >, >=, AND, OR, LIKE, NOT LIKE, BETWEEN, NOT BETWEEN, IS NULL, IS NOT NULL)
- Nested queries within the FROM clause

The data access driver supports the AS, DISTINCT and GROUP BY operations, although they cannot be expressed in the OData 2.0 protocol. The driver performs these operations internally.

Restriction:

The present version of the OData driver does not support the following capabilities:

- UNION and UNION ALL operations
- HAVING clause
- DISTINCT in aggregate functions
- Subqueries in the WHERE clause

Database Functions

The data access driver supports the `AVG`, `MIN`, `MAX`, and `SUM` database functions, although they cannot be expressed in the OData 2.0 protocol. The driver performs these operations internally. The OData 2.0 protocol only supports the `COUNT` function, which corresponds to the `$count` operator.

Related Topics

- OData Driver - Mapping Keys
- OData Driver - Nontrivial Behaviors and Restrictions
- Local Disk Used as a Cache for Sorting Operations

  The CSV OpenDriver and the OData, SAP ERP, XML and Web Service drivers can use the local disk as a cache for sorted rows.

6.3.2 OData Service Location

You can connect to OData services through HTTP. The present release also supports the HTTPS encrypted protocols based on one certificate.

Example:

The following links refer to examples of OData services through HTTP and HTTPS and to an example of an SAP NetWeaver Gateway service.

Related Topics

- http://services.odata.org/Northwind/Northwind.svc/
- https://api.datamarket.azure.com/Data.ashx/data.gov/Crimes/

- http://gw.esworkplace.sap.com/sap/opu/odata/sap/SALESORDERS/

6.3.3 OData Service Example

An OData data source exposes its data according to a schema based on the Entity Data Model (EDM). The schema is described with the help of CSDL. CSDL is an XML format that describes the structure and semantics of Entity Data Model schemas.

The following examples illustrate the structure of a schema based on CSDL. This schema defines metadata of an OData data source that provides the Northwind service. The service URI is http://services.odata.org/Northwind/Northwind.svc/. To learn more about the concepts that rule an entity data model, see the CSDL specification.

Remember:
The present version of OData driver does not support FunctionImports, AnnotationElements, ReferentialConstraints, and dynamic properties.

Example: Northwind Service Schema

You access the schema from http://services.odata.org/Northwind/Northwind.svc/$metadata.

Note:
The fictive `CustomerClothing` ComplexType and `CustomerClothes` Property have been added to the example for documentation purposes.

```
<Schema Namespace ="NorthwindModel">
 ...

 <EntityType Name="Customer">
  <Key>
   <PropertyRef Name="CustomerID"/>
  </Key>
  <Property Name="CustomerID" Type="Edm.String" Nullable="false" MaxLength="5" Unicode="true"
FixedLength="true"/>
  <Property Name="CompanyName" Type="Edm.String" Nullable="false" MaxLength="40" Unicode="true"
FixedLength="false"/>
  <Property Name="ContactName" Type="Edm.String" Nullable="true" MaxLength="30" Unicode="true"
FixedLength="false"/>
  <Property Name="ContactTitle" Type="Edm.String" Nullable="true" MaxLength="30" Unicode="true"
FixedLength="false"/>
  <Property Name="CustomerClothes" Type="NorthwindModel.CustomerClothing" Nullable="false">
  ...
  <NavigationProperty Name="Orders" Relationship="NorthwindModel.FK_Orders_Customers" FromRole="Customers"
 ToRole="Orders"/>
  <NavigationProperty Name="CustomerDemographics" Relationship="NorthwindModel.CustomerCustomerDemo" From
Role="Customers" ToRole="CustomerDemographics"/>
 </EntityType>

 <ComplexType Name="CustomerClothing">
  <Property Name="Suit" Type="Edm.String" Nullable="false">
  <Property Name="Shoes" Type="Edm.String" Nullable="true">
  ...
 </ComplexType>

 <Association Name="FK_Orders_Customers">
  <End Role="Customers" Type="NorthwindModel.Customer" Multiplicity="0..1"/>
  <End Role="Orders" Type="NorthwindModel.Order" Multiplicity="*"/>
  <ReferentialConstraint>
```

```
  ...
  </ReferentialConstraint>
 </Association>
 ...
</Schema>

<Schema Namespace="ODataWeb.Northwind.Model">
 <EntityContainer Name="NorthwindEntities" p7:LazyLoadingEnabled="true" m:IsDefaultEntityContainer="true">

  ...
  <EntitySet Name="Customers" EntityType="NorthwindModel.Customer"/>
  <EntitySet Name="Employees" EntityType="NorthwindModel.Employee"/>
  <EntitySet Name="Order_Details" EntityType="NorthwindModel.Order_Detail"/>
  <EntitySet Name="Orders" EntityType="NorthwindModel.Order"/>
  <EntitySet Name="Products" EntityType="NorthwindModel.Product"/>
  ...
  <AssociationSet Name="FK_Orders_Customers" Association="NorthwindModel.FK_Orders_Customers">
   <End Role="Customers" EntitySet="Customers"/>
   <End Role="Orders" EntitySet="Orders"/>
  </AssociationSet>
  <AssociationSet Name="FK_Employees_Employees" Association="NorthwindModel.FK_Employees_Employees">
   <End Role="Employees" EntitySet="Employees"/>
   <End Role="Employees1" EntitySet="Employees"/>
  </AssociationSet>
  ...
 </EntityContainer>
</Schema>
```

6.3.4 OData Driver - Mapping Namespace, Owner and Qualifier

The `Namespace` attribute of the schema is mapped to the model namespace.

The EntityContainer is mapped to the database qualifier. The default qualifier is the EntityContainer with the attribute `m:IsDefaultEntityContainer` set to `true`.

EntityContainer elements are children of Schema elements. Since different Schemas can contain the same EntityContainer, the database qualifier value is *<Schema_Namespace>.<EntityContainer_Name>*.

The present version of the driver exposes only one owner, which is called `entities`.

Example:

The namespace of the Northwind service is `ODataWeb.Northwind.Model`.

The database qualifier of the Northwind service is `ODataWeb.Northwind.Model.NorthwindEntities`.

The database owner of the Northwind service is `entities`.

6.3.5 OData Driver - Mapping Tables

The data access driver exposes OData feeds as tables to the information design tool.

An EntitySet is mapped to a table. The `Name` attribute of the EntitySet is mapped to the table name.

In the OData schema, an AssociationSet links two EntitySets. An AssociationSet is mapped to a table, if the multiplicity of both EntitySets is `*`. The `Name` attribute of the AssociationSet is mapped to the table name. If at least one of the multiplicities of the EntityTypes is `1` or `0..1`, the AssociationSet is mapped to a foreign key.

Example: Mapping EntitySets

The following EntitySets of the Northwind service are mapped to tables:

- `Customers`
- `Orders`
- `Products`

Example: Mapping AssociationSets

The following AssociationSets of the Northwind service are also mapped to tables:

- `CustomerCustomerDemo`
- `EmployeeTerritories`

The `CustomerCustomerDemo` AssociationSet is mapped to the `CustomerCustomerDemo` table, because the `CustomerCustomerDemo` Association links both `CustomerDemographic` and `Customer` EntityTypes with `*` multiplicities.

```
<Association Name="CustomerCustomerDemo">
 <End Role="CustomerDemographics" Type="NorthwindModel.CustomerDemographic" Multiplicity="*"/>
 <End Role="Customers" Type="NorthwindModel.Customer" Multiplicity="*"/>
</Association>
```

Related Topics

- OData Driver - Mapping Keys

6.3.6 OData Driver - Mapping Columns

An EntitySet is defined by an EntityType. An EntityType consists of a series of Properties. A Property of primitive type is mapped to a column of the table that represents the EntitySet. The `Name` attribute of the Property is mapped to the column name.

A Property of complex type is mapped to a set of table columns. One column corresponds to one subproperty. The column name is the concatenation of the names of the properties of complex type and primitive type, separated by a slash (`/`). This mapping also applies to nested properties of complex type.

Example:

The `Orders` table has the `OrderID` column, because the `Order` EntityType has the `OrderID` Property.

The `Customers` table has the `Address`, `CustomerClothes/Suit` and `CustomerClothes/Shoes` columns, because:

- The `Address` Property of the `Customer` EntityType has a primitive type.
- The `CustomerClothes` Property of the `Customer` EntityType has a complex type, which is made of the `Suit` and `Shoes` Properties of primitive type.

6.3.7 OData Driver - Mapping Keys

Mapping Primary Keys

The Key of an EntityType is mapped to the primary key of the table that maps the corresponding EntitySet. This can be one or more columns.

Example:

The `Customers` table has a primary key on the `CustomerID` column, because the `Customer` EntityType has a Key that refers to the `CustomerID` Property.

Mapping Foreign Keys

The OData protocol uses NavigationProperties to expose the join paths of a relational model. NavigationProperties use Roles defined in AssociationSets to link EntitySets according to their multiplicity.

The OData driver represents bidirectional joins as foreign keys. A bidirectional join can be created between tables if EntityTypes of the EntitySets that represent these tables contain NavigationProperties that use the same AssociationSet to link EntitySets. This means that both ends of an AssociationSet must have NavigationProperties in both EntityTypes. Associations that do not meet this condition are not exposed as foreign keys.

If the multiplicity of both EntitySets is `*`, the AssociationSet is mapped to a table. If at least one of the multiplicities of the EntityTypes is `1` or `0..1`, the AssociationSet is mapped to a foreign key in the table that represents the EntitySet of `*` multiplicity. The pattern of a foreign key name is `-idref-`*`<NavigationProperty>-<Primary Key>`*.

Remember:
Foreign keys are hidden columns in data foundation tables by default, because they do not contain business data. However, you can display them by editing the table and column properties. See the *Information Design Tool User Guide*.

Caution:
SAP recommends application users not to query `-idref` columns, because it may cause a driver performance decrease.

Example: Foreign Keys of an EntitySet

The `Orders` table has `-idref-Customer-CustomerID`, `-idref-Employee-EmployeeID`, and `-idref-Shipper-ShipperID` foreign keys because:

- The `Order` EntityType has the following NavigationProperties.

```
<EntityType Name="Order">
...
 <NavigationProperty Name="Customer" Relationship="NorthwindModel.FK_Orders_Customers" FromRole="Orders"
 ToRole="Customers"/>
 <NavigationProperty Name="Employee" Relationship="NorthwindModel.FK_Orders_Employees" FromRole="Orders"
 ToRole="Employees"/>
 <NavigationProperty Name="Shipper" Relationship="NorthwindModel.FK_Orders_Shippers" FromRole="Orders"
 ToRole="Shippers"/>
</EntityType>
```

- These NavigationProperties refer to the `FK_Orders_` Associations, which link the `Order` EntityType to `Customer`, `Employee` and `Shipper` EntityTypes. There is one `0..1` multiplicity in each Association. This generates three foreign keys in the table that represents the `Orders` EntitySet.

```
<Association Name="FK_Orders_Employees">
 <End Role="Employees" Type="NorthwindModel.Employee" Multiplicity="0..1"/>
 <End Role="Orders" Type="NorthwindModel.Order" Multiplicity="*"/>
</Association>
```

- The `CustomerID` in `-idref-Customer-CustomerID` comes from the primary key of the `Customers` table.
- The `EmployeeID` in `-idref-Employee-EmployeeID` comes from the primary key of the `Employees` table.
- The `ShipperID` in `-idref-Shipper-ShipperID` comes from the primary key of the `Shippers` table.

Example: Foreign Keys of an AssociationSet

The `CustomerCustomerDemo` table has the `-idref-CustomerDemographics-CustomerTypeID` and `-idref-Customers-CustomerID` foreign keys because:

- The `Customer` EntityType has a `CustomerDemographics` NavigationProperty.

```
<EntityType Name="Customer">
...
 <NavigationProperty Name="CustomerDemographics" Relationship="NorthwindModel.CustomerCustomerDemo"
FromRole="Customers" ToRole="CustomerDemographics"/>
</EntityType>
```

- The `CustomerDemographic` EntityType has a `Customers` NavigationProperty.

```
<EntityType Name="CustomerDemographic">
...
 <NavigationProperty Name="Customers" Relationship="NorthwindModel.CustomerCustomerDemo" FromRole="Cus
tomerDemographics" ToRole="Customers"/>
</EntityType>
```

- The NavigationProperties refers to the `CustomerCustomerDemo` Association, which links `Customer` and `CustomerDemographic` EntityTypes. This generates two foreign keys in the table that represents the `CustomerCustomerDemo` AssociationSet.

```
<Association Name="CustomerCustomerDemo">
 <End Role="CustomerDemographics" Type="NorthwindModel.CustomerDemographic" Multiplicity="*"/>
 <End Role="Customers" Type="NorthwindModel.Customer" Multiplicity="*"/>
</Association>
```

- The `CustomerID` in `-idref-Customers-CustomerID` comes from the primary key of the `Customers` table.
- The `CustomerTypeID` in `-idref-CustomerDemographics-CustomerTypeID` comes from the primary key of the `CustomerDemographics` table.

6.3.8 OData Driver - Mapping Documentation Elements

Depending on availability, either Summary or LongDescription subelements of Documentation are mapped to descriptions of relational model entities.

A Documentation subelement of EntitySet and AssociationSet is displayed as the table description in the information design tool. A Documentation subelement of Property or NavigationProperty is displayed as the column description. If there is no Summary or LongDescription, then the Description field remains empty.

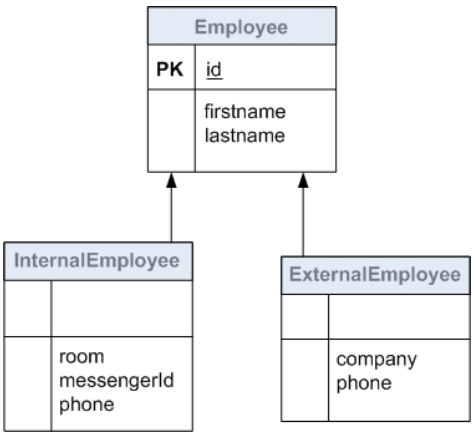
Descriptions are not localized.

6.3.9 OData Driver - Mapping Entities of Derived Types

The following example illustrates an EntitySet of type `Employee`.

```
<EntitySet Name="Employees" EntityType="Employee" />
```

The following diagram shows the inheritance concept between one base type `Employee`, and two derived types, `ExternalEmployee` and `InternalEmployee`. The derived types inherit from all properties of the base type. They also define additional properties, which are called direct properties.



For a specific EntitySet, the result schema is the result of a merge to one general type. The following diagram shows the Employee table as represented in the relational model.

Employee	
PK	id
	firstname lastname InternalEmployee_room InternalEmployee_messengerId InternalEmployee_phone ExternalEmployee_company ExternalEmployee_phone

The `Employee` table of the relational model will contain information from both entities, that is, columns of the parent type and of the derived types. To prevent naming conflicts, columns declared in derived types are prefixed with the name of the EntityType. They can also contain the null value.

6.3.10 Monitoring the OData Driver Performance

You may consider the following behaviors of the OData driver to improve the performance of the connection at query execution.

Executing Operations on Client or Provider Side

When creating a connection, you can choose to execute some operations of the SQL query at the driver or provider level. The following table shows the parameters of the connection wizard that control this behavior.

If a parameter is selected, the OData service provider handles the corresponding operation if it can support it. This is the default behavior. If unselected, the OData driver handles the operation.

Caution:
SAP recommends that you do not use the data access driver to execute these operations, because it can reduce the connection performance. Since the full data set needs to be transferred to the client, response time increases. Use it only if the OData service provider does not support or partially supports the operation.

Parameter	Operation	Impact on Performance
Column Selection	SELECT	If the parameter is unselected, the OData driver handles the SQL projection, which increases the data traffic significantly.

Parameter	Operation	Impact on Performance
Supported Filter Conditions	WHERE	You can execute some filtering operations at the driver level if the OData standards do not support them. However, filters on the provider usually reduce query execution time. The Data Access layer applies the following logic to improve performance: any filtering condition compatible with OData standards is pushed to the OData provider. The OData driver executes any other incompatible filter that the WHERE clause may contain. See the example below.
Sorting	ORDER BY	The OData provider executes the ORDER BY operation if no DISTINCT or GROUP BY expression is used in the query. If a query contains either a DISTINCT or a GROUP BY expression, then the driver has to perform the sorting operation, which increases the execution time.

Note:

- The ORDER BY operation depends on the `Bucket Split Size` SBO parameter when it is performed by the driver.
- The BETWEEN operator is mapped to OData comparison functions, then executed by the OData provider.
- The LIKE and NOT LIKE operators are usually executed on the client. However, they are translated into the `startsWith` and `endsWith` OData functions, if only one `%` pattern is placed at the end or at the beginning of the filter value respectively.

Example: Supporting Filter Expressions

The following query contains two filter conditions in the WHERE clause:

```
SELECT * FROM T1 WHERE col1='San Francisco' AND col2 LIKE 'S%n'
```

The OData provider can handle the first part of the clause, but not the second one. The OData driver then executes `col2 LIKE 'S%n'`, while the provider executes `col1='San Francisco'`.

Example: Mapping the startsWith and endsWith Functions

The following SQL query returns the Sales employees who live on boulevards.

```
SELECT * FROM EMPLOYEES WHERE EMPLOYEES."Title" LIKE 'Sales %' AND EMPLOYEES."Address" LIKE '% Blvd.'
```

It corresponds to the URI below:

```
http://services.odata.org/Northwind/Northwind.svc/Employees?$filter=startswith(Title,'Sales')&endswith(Address,'Blvd.')
```

Executing Aggregate Functions

The `COUNT` database function is the only aggregate function that the OData 2.0 protocol supports. Other aggregate functions and grouping triggered by the GROUP BY clause are executed on the client. However, if the driver performs calculations that modify the result before the count is computed, the `COUNT` function must also be executed on the client. This can decrease the connection performance.

Setting the Connection Timeout Parameter

You can choose the value of the **Connection Timeout** parameter in the connection wizard to fine-tune the driver performance. It specifies the time in seconds a connection remains active in case of no response from the data source.

You have to choose the best parameter value for the best performance. For example, if you connect to a data source with a high response time, you may choose a high value for **Connection Timeout** to wait for the server response and not to raise an error too early. If your data source is usually fast, you may choose a low value, not to wait too long for an error if the server does not respond.

Using the Cache Metamodel

The **Cache MetaModel** parameter is selected by default in the connection wizard. It allows you to retrieve the metadata model only once for the duration of the connection in the connection pool, instead of doing it at each query execution. This results in a performance increase of the connection.

Note:
You must enable the connection pooling if the metadata model is cached.

Related Topics

- OData Driver Capabilities

6.3.11 OData Driver - Nontrivial Behaviors and Restrictions

Sorting Algorithms

Sorting algorithms may differ from one service provider to another. This can lead to different results when ORDER BY is used in the query.

Since the OData driver may execute sorting operations using its own algorithm, the use of combined operations such as ORDER BY, GROUP BY and DISTINCT may also lead to different sort orders.

Mapping the Max Rows Parameter

The `$top` OData function is mapped to the `Max Rows` parameter of the Query Panel, only if the query operations do not modify the number of rows. Query results can be erroneous if a filter is applied, which removes some rows from the expected `$top` result.

Example:

The following OData query returns the first 15 rows from the `Orders` table:

```
http://services.odata.org/Northwind/Northwind.svc/Orders?$top=15
```

The following query returns the first 15 rows where the shipment city is "Reims". This returns only 5 rows:

```
http://services.odata.org/Northwind/Northwind.svc/Orders?$top=15&$filter=ShipCity eq 'Reims'
```

Restrictions to the COUNT Function

A row count on a table in the information design tool is semantically counting rows that contain non-null values for the defined columns. Some Azure services return only the total number of rows in one table. This may cause a discrepancy between expected and actual query results.

Restrictions to the SAP NetWeaver Gateway Services

If the NetWeaver Gateway service provider does not support filters on all columns, you must not select **Supported Filter Conditions** when creating the connection. If it does not support sorting on all columns, you must not select **Sorting** when creating the connection.

The schema may indicate these filtering and sorting restrictions with the help of the `sap:filterable` and `sap:sortable` metadata. In the present version, the OData driver does not map these attributes.

Some providers may not offer a full table scan capability on all tables. This can lead to runtime errors, for example if one of the tables involved in a JOIN cannot be scanned.

SAP recommends that you create connections to the SAP ERP system directly to avoid issues with ad hoc queries.

6.3.12 OData Driver - Configuring Column Maximum Size

For some properties, the `MaxLength` parameter may have the value `Max`. For example:

```
<Property Name="Synopsis" Type="Edm.String" Nullable="true" MaxLength="Max" Unicode="true"
FixedLength="false"/>
```

Since the OData driver cannot recognize this nonnumeric value, it has to be configured. The `cs.cfg` configuration file provides the following parameters:

- `Binary Max Length` to set the maximum size of table columns whose type is binary
- `String Max Length` to set the maximum size of table columns whose type is string

Related Topics

- Binary Max Length
- String Max Length

6.3.13 Authenticating OData Sources

Access to some OData data sources requires authentication. An application can authenticate to an OData data source using the following methods:

- HTTP basic authentication, which uses a username and a password. Credentials are not encrypted.
- A GET access token, that is, a {key, value} pair appended to the service URI. The token key and value can vary according to the service. Set the token through custom parameters by using **Custom Authentication Parameters** in the connection wizard. Use the following pattern: **<***key***>=<***value***>**.

Note:
The Data Access layer does not provide traces for custom authentication parameters, because they contain security information. They are replaced with `<CUSTOM_AUTH_PARAMS>` in the OData driver traces. However, consider using other tracing levels such as the **Enable Job Tracing** option in the CMC to display the parameters.

Example: **Access Token**

`http://wine.cloudapp.net/Regions?apikey=<api key value>.`

Authenticating Microsoft Azure Data Sources

You connect to a Microsoft Azure data set using the HTTP basic authentication. In the connection wizard, enter the account key as password. You can leave the username blank.

6.3.14 Tracing the OData Driver Activity

The Data Access layer traces the OData driver with the PATH and DEBUG severity levels. The PATH traces contain the entries and exits of each method used in query planning and execution. It allows you to find out whether the OData driver or the OData provider has executed a query operation and where the cause of a problem can be.

The information contained in the URI (the data source credentials, the proxy server credentials and the custom authentication parameters) is blanked out in driver traces.

6.4 SAP ERP Driver

The Data Access layer allows the BI platform to connect to SAP ERP systems through the SAP Java Connector (JCo) 3.x network layer. It provides a data access driver that allows access to the following ERP objects:

- InfoSets
- SAP Queries
- ABAP functions

For more information about supported SAP ERP systems, see the *Product Availability Matrix*.

SAP ERP connectivities support single sign-on (SSO) on all platforms. For more information about authentication, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

SAP ERP connectivities also support Preferred Viewing Locale (PVL) for all authentication modes. You deactivate the use of PVL by checking **Save Language** when creating the connection.

You can find the driver configuration files in the `connectionserver-install-dir\connectionServer\jco` directory. The `jco.prm` file lists the data access driver capabilities in terms of database operations.

The connection wizard provides a workflow to enter the necessary information to connect to SAP ERP systems.

Related Topics

- Single Sign-On Support

6.4.1 SAP ERP Driver Capabilities

The data access driver supports the following querying capabilities:

- Basic operations (SELECT, WHERE, ORDER BY, GROUP BY, and AS)
- Basic operators (=, !=, <, <=, >, >=, BETWEEN, NOT BETWEEN, IN, NOT IN, AND, OR) whose operands can be either column names or constant values
- DISTINCT clause within the SELECT statement
- Constant values within the SELECT statement
- LIKE and NOT LIKE accepts escape characters
- Nested queries within the FROM clause

The Data Access layer performs the sorting operations internally.

Note:
The data access driver does not allow ordering columns by index in SQL statements. Only ORDER BY with column names is a valid clause. The `ORDER_BY_SUPPORTS_COLUMN_INDEX` parameter is set to `NO` in the `jco.prm` file.

Related Topics

- Local Disk Used as a Cache for Sorting Operations

The CSV OpenDriver and the OData, SAP ERP, XML and Web Service drivers can use the local disk as a cache for sorted rows.

- ORDER_BY_SUPPORTS_COLUMN_INDEX

6.4.2 SAP ERP Driver - Access to InfoSets and SAP Queries

About InfoSets and SAP Queries

In the SAP ERP system, InfoSets and SAP Queries are organized within either a local work area or a global work area. Objects within the local work area can only be accessed by the SAP ERP client used, whereas objects of the global work area can be accessed by all clients. An SAP Query is created in a user group and belongs to it.

For more information about work areas and user groups, refer to your SAP ERP documentation.

Object Mapping

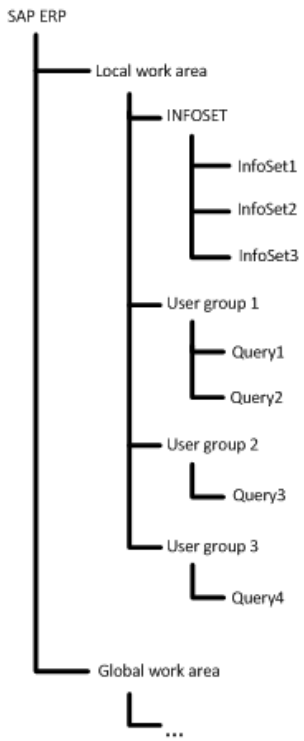
The data access driver exposes InfoSets and SAP Queries as tables to the information design tool. InfoSets and SAP Queries are made of a series of fields, which are mapped to table columns. Fields are used in SELECT statements.

At driver level, work areas are mapped to qualifiers. The default qualifier corresponds to the Local work area.

User groups are mapped to owners. User groups of a work area are then placed beneath a qualifier that represents this work area.

InfoSets of a work area are placed beneath a qualifier that represents this work area and beneath a fictive owner called `INFOSET`. SAP Queries of a work area are placed beneath the qualifier that represents this work area. SAP Queries of a user group are placed beneath the owner that represents this user group.

The organization of InfoSets and SAP Queries in a work area is depicted in the figure below:



Fields are mapped to table columns. An InfoSet or an SAP Query contains fields that are both projection and selection fields. It also contains selection fields that are not used for projection. When you create a connection with the connection wizard, you choose whether these selection fields are represented as table columns. The **Map Selection Fields into Table Columns** parameter addresses this functionality. If selected, then the field is mapped as an optional input column. If unselected, the selection field is ignored, and no filtering is possible on this field.

Note:
There is no specific mapping for obligatory selection fields. The application user must query these fields with filters.

Input Columns

Input columns of a table are columns on which an equal condition (= operator) must be given when the application user queries the table. Only equal conditions are allowed on input columns. `IN` conditions are forbidden.

In the information design tool, the user can view input columns as mandatory or optional. A mandatory input column is a column for which it is mandatory to set a value. An optional input column is a column for which it is not mandatory to set a value.

If the user does not set any value to an optional input column in a query, then Connection Server returns one of the following:

- No value if the optional input column has no default value
- The default value if the optional input column has a default value

Driver Capabilities

When the application user queries an InfoSet, some database operations are passed to the SAP ERP system (SELECT, WHERE, sorting only with ORDER BY columns, and AS), while others are handled internally by the driver (other filtering with WHERE, GROUP BY, DISTINCT and sorting only with ORDER BY aggregate functions).

When the application user queries an SAP Query, some operations are also passed to the ERP system. These are SELECT, WHERE, ORDER BY (only if the SAP query already contains the same sorting) and AS.

For more information on the operations directly managed by the system, see your SAP ERP system documentation.

6.4.3 SAP ERP Driver - Access to ABAP Functions

Remember:
The SAP ERP driver allows you to create connections to released and unreleased ABAP functions.

ABAP Function Concepts

ABAP functions are made of a series of parameters that can either be input parameters used in function calls, or output parameters, which are contained in call responses. In SAP terminology, input parameters are called import, whereas output parameters are called export. Some parameters can be both input and output. They are called changing parameters. Besides import, export and changing parameters, there are also table parameters, which can be considered as a type of changing parameters.

An ABAP function parameter can have the following type:

- Field of primitive ABAP type
- Structure, which is a sequence of several fields
- Table, which can either be made of fields, structures, or even tables

Object Mapping

The data access driver exposes ABAP functions as tables to the information design tool. At driver level, ABAP function groups are mapped to owners and ABAP functions are mapped to a qualifier called `ABAPFunction`. This means each ABAP function is placed beneath a specific owner, which corresponds to a function group in the SAP ERP system.

Import, changing, and table parameters used as input are mapped to input columns. Export, changing, and table parameters used as output are mapped into tables. Input columns can either be mandatory or optional. The Data Access layer considers these parameters as primary key attributes, which are not nullable.

The driver maps ABAP function parameters as follows:

- A mandatory parameter is mapped to a mandatory input column. The user has to set a value to this column.

- An optional parameter that has a default value is mapped to an optional input column. The user does not have to set a value to this column. The value returned to Connection Server by the data source in that case is the default value.
- An optional parameter with no default value is also mapped to an optional input column. Connection Server returns the following values of string data type, depending on the SQL type of the input column:

SQL Type	Value
SQL_Integer	`0`
SQL_Char, SQL_VarChar, SQL_Binary	An empty string
SQL_Double, SQL_Float, SQL_Numeric	`0.0`
SQL_Date	`00010101`
SQL_Time	`000000`
Any other type	An empty string

Note:
The driver appends a `<column name>-ID` column to the table which the ABAP function is mapped into and another `<column_name>-IDREF` to the table if it contains a reference to another table.

Column Renaming

Column names are renamed at driver level. The following table shows the prefixes added to the columns.

Input or Output	Parameter	Prefix
Input	Import	`-IMPORT_`
Input	Changing	`-IMPORT_CH_`
Input	Table	`-IMPORT_CH_`
Output	Changing	`-EXPORT_CH_`
Output	Table	`-EXPORT_CH_`

Note:
When an output table parameter (A) is made of another table (B), both tables are renamed `-EXPORT_CH_A` and `-EXPORT_CH_A.B` respectively.

Driver Capabilities

All database operations (SELECT, WHERE, ORDER BY, GROUP BY, AS, etc.) are handled internally by the driver, while equal conditions on input columns are passed to the SAP ERP system.

Remember:
The driver only accepts tables with one single row as input parameters. They are equivalent to input structures.

- When you create a connection in the connection wizard, you set whether a changing parameter of table type or a table parameter is considered as both input and output. The **Map Table Parameters into Input Columns** parameter addresses this functionality. If checked, the parameter is both input and output. In this case, it can only be a structure as input. If not checked, the parameter is only considered as an output parameter.
- The driver considers fields of an optional input structure as optional input columns. Input columns are considered optional as a whole, that is, if a column has a value, the others must also have a value. The application user must ensure all columns have a value.
- Some ABAP function parameters may have values whose size is reported to be equal to zero by the SAP ERP system. The driver uses the `String Max Length` parameter of the `cs.cfg` file to set a string maximum length to the table columns that are mapped to these parameters.

Related Topics

- String Max Length

6.4.4 SAP ERP Driver Restrictions

- You cannot query ERP objects based on ABAP programs that involve a SAPGUI processing, because they cannot be accessed using Remote Function Calls.
- The JCo API does not support array fetch. This means the result data is returned all at once. To restrict the number of resulting rows and to avoid an out-of-memory issue, the application user has to filter data appropriately. Regardless of this restriction, Connection Server still applies array fetch.

  Note:
  The application user can also set a value to `Max Rows` in the Query Panel. However, the SAP ERP system does not take `Max Rows` into account if the universe defined is multisource or if the query requires computations after data is fetched (SELECT DISTINCT, ORDER BY, and GROUP BY).

- The `ERP Max Rows` parameter of the `jco.sbo` configuration file allows the application user to query without filter an InfoSet or an SAP Query table that only contains numerical data type columns. The user can also query a table containing numerical and DATS data type columns. This parameter is also used when `Max Rows` is not passed to the SAP ERP system. `ERP Max Rows` can be used with single-source and multisource universes. Set `ERP Max Rows` to an appropriate value to avoid out-of-memory issues.

  Note:
  In the case of a multisource universe, you can also set a value to the `Max Rows` parameter of the Query Panel of the data federation administration tool. See the *Data Federator Administration Tool Guide* for more information.

Related Topics

- Array Fetch Size
- ERP Max Rows

6.5 XML Driver

The Data Access layer allows the BI platform to connect to XML documents as data sources.

XML documents are based on XML schemas. An XML schema is an XSD document that defines the metadata of an XML data source.

The connection wizard provides a workflow to enter the necessary information to connect to XML data sources. XML data sources show up under `XML Files` DBMS and `XML File Connector` NetworkLayer.

You can find the driver configuration files in the *`connectionserver-install-dir`*`\connection Server\xml` directory. The `xml.prm` file lists the data access driver capabilities in terms of database operations and functions.

6.5.1 XML Driver Capabilities

The data access drivers support the following querying capabilities:

- Basic database operations (SELECT, WHERE, ORDER BY, GROUP BY, DISTINCT)
- Column aliases within the ORDER BY clause (AS)
- Nested queries within the FROM clause
- Filtering operators (=, !=, <, <=, >, >=, AND, OR, BETWEEN, NOT BETWEEN, IN, NOT IN, LIKE, NOT LIKE, IS NULL, IS NOT NULL)

The data access driver does not support JOIN operations.

The data access driver supports the `MIN`, `MAX`, `AVG`, `SUM`, and `COUNT` grouping functions in SELECT statements.

Related Topics

- Local Disk Used as a Cache for Sorting Operations

  The CSV OpenDriver and the OData, SAP ERP, XML and Web Service drivers can use the local disk as a cache for sorted rows.

6.5.2 XML File Location

XML documents used as data sources can be either local or remote. For files on remote systems, the HTTP, FTP, and SMB protocols are supported. The present release also supports the HTTPS and FTPS encrypted protocols based on one certificate.

Note:
SMB is the standard file sharing protocol on MS Windows. Since the jCIFS library provides access to the protocol, you must install the jCIFS JAR file in the Connection Server directory, that is *`connection server-install-dir`*`\connectionServer\sharedLibraries\jcifs`. The version to install is 1.3.15. It is available at http://jcifs.samba.org/.

You can set whether the data source is local or remote using the **Location Type** parameter of the connection wizard. If the data source is remote, you can also set the protocol used through the **Protocol** parameter of the wizard. The **Protocol** value must match the protocol specified in the **Filepath or Pattern** parameter value.

Example: XML File Paths

- `C:\xml\report.xml` or `file:///C:/xml/report.xml` for a local file
- `http://server:8080/path/report.xml` for a file accessible through HTTP

- `ftp://server/path/` for files accessible through FTP
- `smb://server:port/myshare/mydirectory` for files accessible through SMB

Related Topics

- CSV File Location

6.5.3 XML Document and Schema Example

The following examples illustrate the structure of an XML document called `clubdemo.xml` and based on the `clubdemo.xsd` schema.

Example: **clubdemo.xml Document**

```
<?xml version="1.0" encoding="UTF-8"?>
<ClubDemo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="clubdemo.xsd">


   <Countries>
      <Country name="France"/>
      <Country name="US"/>
   </Countries>

   <Customers>
      <Customer ID="204">
         <Name type="first">Christine</Name>
         <Name type="last">Martin</Name>
         <Age>25</Age>
         <Address>12, allee Victor Hugo</Address>
         <ZIP>75016</ZIP>
         <City>Paris</City>
         <Country name="France"/>
         <Invoice>1234</Invoice>
      </Customer>
      <Customer ID="103">
         <Name type="first">Peter</Name>
         <Name type="last">Travis</Name>
         <Age>34</Age>
         <Address>7835 Hartford Drive</Address>
         <ZIP>06108</ZIP>
         <City>Springfield</City>
         <Country name="US"/>
         <Invoice>23204</Invoice>
      </Customer>
      <Customer ID="106">
         <Name type="first">William</Name>
         <Name type="last">Baker</Name>
         <Age>64</Age>
         <Address>2890 Grant Avenue</Address>
         <ZIP>20020</ZIP>
         <City>Washington DC</City>
         <Country name="US"/>
         <Invoice>35306</Invoice>
         <Invoice>44106</Invoice>
       </Customer>
    </Customers>
</ClubDemo>
```

Example: **clubdemo.xsd Schema**

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="ClubDemo">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Countries"/>
            <xs:element ref="Customers"/>
         </xs:sequence>
      </xs:complexType>

      <xs:key name="countkey">
         <xs:selector xpath="Countries/Country"/>
         <xs:field xpath="@name"/>
      </xs:key>

      <xs:key name="custkey">
         <xs:selector xpath="Customers/Customer"/>
         <xs:field xpath="@ID"/>
      </xs:key>

      <xs:keyref name="custkeyref" refer="countkey">
         <xs:selector xpath="Customers/Customer"/>
         <xs:field xpath="Country/@name"/>
      </xs:keyref>
   </xs:element>

   <xs:element name="Countries">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Country" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="Country">
      <xs:complexType>
         <xs:attribute name="name" type="xs:string"/>
      </xs:complexType>
   </xs:element>

   <xs:element name="Customers">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="Customer" maxOccurs="unbounded"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

   <xs:element name="Customer">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="Name" maxOccurs="2">
               <xs:complexType>
                  <xs:simpleContent>
                     <xs:extension base="xs:string">
                        <xs:attribute name="type" type="xs:string"/>
                     </xs:extension>
                  </xs:simpleContent>
               </xs:complexType>
            </xs:element>
            <xs:element name="Age" type="xs:integer"/>
            <xs:element name="Address" type="xs:string"/>
            <xs:element name="ZIP" type="xs:string"/>
            <xs:element name="City" type="xs:string"/>
            <xs:element name="Country">
               <xs:complexType>
                  <xs:attribute name="name" type="xs:string"/>
               </xs:complexType>
            </xs:element>
            <xs:element name="Invoice" type="xs:integer" maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="ID" type="xs:integer"/>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

6.5.4 XML Driver - Mapping Multiple Files

You can connect to multiple files if the **Filepath or Pattern** parameter of the connection wizard is set to a pattern, for example `clubdemo_*.xml`, which is matched by files such as `clubdemo_1.xml` and `clubdemo_2.xml`.

Remember:
You cannot connect to multiples files located on an HTTP server. The HTTP protocol does not support the use of wildcards in patterns.

If the **Merge Files into One** parameter is selected in the connection wizard, identical tables from multiple XML files are merged into one table. The name of the table qualifier is the full path of the pattern, for example `C:\files\xml\clubdemo_*.xml`. It is assumed that all files have the same schema.

If the paremeter is not selected, tables are not merged as they do not belong to the same qualifier. Every XML document is then mapped to a database qualifier. The name of the qualifier is the full path of the XML file, for example `C:\files\xml\clubdemo_1.xml`. In that case, each XML file can have its own schema.

No data is mapped to database owners.

6.5.5 XML Driver - Mapping Tables

The data access driver exposes XML files as tables to the information design tool.

The XML document root is mapped to a table.

Elements that have `maxOccurs` value greater than `1` are mapped to tables.

The name of the table mapped to an element is the concatenation of all the element names between the root and the element, separated with a slash (`/`).

Example:

The `clubdemo.xml` document is mapped to the following tables:

- `ClubDemo`
- `ClubDemo/Countries/Country`
- `ClubDemo/Customers/Customer`
- `ClubDemo/Customers/Customer/Invoice`
- `ClubDemo/Customers/Customer/Name`

Related Topics

- XML Document and Schema Example

6.5.6 XML Driver - Mapping Columns

The following objects under an XML element are mapped to columns of the table that represents this element:

- Attributes
- Elements of either `simpleType` or `complexType` with a `simpleContent` element
- Texts which are in a mixed element, that is, whose type defines `mixed=true`
- `any` and `anyAttribute`

If the column represents an attribute, its name is the concatenation of the names of all the elements starting with (but not including) the element with `maxOccurs` greater than `1`, up to and including the element hosting this attribute. The attribute name is prefixed with `@`, to avoid name conflicts with other elements.

If the column represents an element, its name is the concatenation of the names of all the elements starting with (but not including) the element with `maxOccurs` greater than `1`.

If the column represents the same element as its table, its name is the element name.

The separator used in concatenations is the slash (`/`).

Element names are prefixed with a hyphen (`-`) when they represent parent elements used in paths to build foreign keys.

The following mapping examples refer to the topic on the example of XML document and schema.

Example:

The `ClubDemo` table has the following column:

- `-id`

Example:

The `ClubDemo/Countries/Country` table has the following columns:

- `@name`
- `-Countries/-ClubDemo/-id`

Example:

The `ClubDemo/Customers/Customer` table has the following columns:

- `Age`
- `Address`
- `ZIP`
- `City`
- `Country/@name`
- `@ID`
- `-Customers/-ClubDemo/-id`

Example:

The `ClubDemo/Customers/Customer/Invoice` table has the following columns:

- `-id`
- `Invoice`
- `-Customers/@ID`

Example:

The `ClubDemo/Customers/Customer/Name` table has the following columns:

- `-id`
- `Name`
- `@type`
- `-Customer/@ID`

In the XML schema, `name`, `type`, and `ID` are attribute names of the `Country`, `Name`, and `Customer` elements respectively. The `Invoice` and `Name` columns represent the `Invoice` and `Name` elements respectively. See the topic on mapping primary and foreign keys to learn about the meaning of the other column names.

Related Topics

- XML Document and Schema Example
- XML Driver - Mapping Primary and Foreign Keys
- XML Driver - Mapping anyType and simpleType

6.5.7 XML Driver - Mapping Primary and Foreign Keys

Primary Keys

The following rules apply to mapping primary keys:

- The driver generates the primary key of the table that maps the document root. The associated column is called `-id` and the column value is the document name.

- An attribute or element of type "xs:ID" is mapped to a column that represents a primary key.

If the XML schema defines a key with <xs:key>, the following rules apply:

- If the key is defined at the root level and if the selector contains an element that has a maxOccurs value greater than 1, then the key is mapped to the primary key of the corresponding table.
- If the key is not defined at the root level, then the primary key of the table is the combination of the primary key of the parent element and the field of the key.

If the schema does not define a primary key for a selector element that has a maxOccurs value greater than 1, a column with name -id is added to the table as primary key.

Example: **Document Root**

The ClubDemo element is the root of clubdemo.xml document. The -id column is created and defined as the primary key of the ClubDemo table. The -id column value is clubdemo.xml.

Example: **Explicit Key**

The following key set at the root level defines the @name column of the ClubDemo/Countries/Country table as primary key:

```
<xs:key name="countkey">
   <xs:selector xpath="Countries/Country"/>
   <xs:field xpath="@name"/>
</xs:key>
```

The ClubDemo/Customers/Customer table has the @ID column as primary key, because the schema defines the custkey explicit key:

```
<xs:key name="custkey">
   <xs:selector xpath="Customers/Customer"/>
   <xs:field xpath="@ID"/>
</xs:key>
```

Example: **No Explicit Key**

The ClubDemo/Customers/Customer/Invoice table has an -id column as primary key, because the schema does not define a key for this element.

Foreign Keys

If the XML schema defines a key reference with <xs:keyref>, and if the key reference is defined at the same level as the key it references, then the key reference is mapped to a foreign key of the corresponding table.

For each table that corresponds to an element which is not the root, the foreign key is built from the primary key columns of its parent table and the reverse path from the element to its parent. This enables the mapping of the relationships between parent and child elements.

Restriction:
Although data foundations expose foreign keys, the XML driver does not support JOIN operations.

Example: Keyref

The `ClubDemo/Customers/Customer` table has the following foreign keys:

- `Country/@name`, which is the field of the following keyref:

```
<xs:keyref name="custkeyref" refer="countkey">
   <xs:selector xpath="Customers/Customer"/>
   <xs:field xpath="Country/@name"/>
</xs:keyref>
```

  Note that `countkey` is the explicit key defined for a table which is not a parent.
- `-Customers/-ClubDemo/-id`, which is built from the reverse path to its parent table and the primary key of the parent table

---

Example: No Keyref

The `ClubDemo/Customers/Customer/Invoice` table has `-Customer/@ID` column defined as a foreign key, which is built from the primary key column of the parent table and the reverse path to its parent table.

Related Topics

- XML Document and Schema Example
- XML Driver - Mapping Columns

6.5.8 XML Driver - Mapping Mixed Elements

The whole content of a mixed element is mapped to a table column. This includes its children and all data between the children.

Attributes of a mixed element are mapped to columns, independently of how the element is mapped.

Example:

In this example, the `documentation` element has a `complexType` made of three child elements (`name`, `datatype` and `location`).

```
<documentation>
 The <name>XML driver</name> allows you to retrieve <datatype>XML</datatype> data
 from documents located on a <location>remote server</location>.
</documentation>
```

The Data Access layer maps such elements as follows:

- The `documentation` element is mapped to a table column.

- The `documentation` element content becomes the column data, including its children and all data between the children. In the example, the column contains the following row:

```
The <name>XML driver</name> allows you to retrieve <datatype>XML</datatype> data from documents located
 on a <location>remote server</location>.
```

6.5.9 XML Driver - Mapping Recursive Elements

The following schema defines recursive elements:

```
<xs:element name="parent">
   <xs:complexType>
      <xs:sequence>
         <xs:element ref="child" minOccurs="0" maxOccurs="5"/>
      </xs:sequence>
   </xs:complexType>
</xs:element>

<xs:element name="child">
   <xs:complexType>
      <xs:sequence>
         <xs:element ref="parent" minOccurs="0" maxOccurs="2"/>
      </xs:sequence>
  </xs:complexType>
</xs:element>
```

The XML driver maps the first `parent` element and the `child` element to tables and ignores the `parent` child.

6.5.10 XML Driver - Mapping any and anyAttribute

Example: XSD Schema with any

```
<xs:element name="documentation">
 <xs:complexType>
   <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="datatype" type="xs:string"/>
      <xs:any minOccurs="0" maxOccurs="1"/>
   </xs:sequence>
 </xs:complexType>
</xs:element>
```

- The `name` and `datatype` are mapped to columns
- The `any` element is mapped to the `-any` column

Example: XML File with any

```
<documentation>
   <name>Data Access Guide</name>
   <datatype>XML</datatype>
```

```
   <audience>
      <type>External</type>
      <name>Administrator</name>
   </audience>
</documentation>
```

The `-any` column data is the element name and its content. In the example, it contains the following row:

```
<audience><type>External</type><name>Administrator</name></audience>
```

Restriction:

If the `any` element has `maxOccurs` greater than `1`, the XML driver only maps to the `-any` column the first element name encountered when parsing the XML document. Other element names are ignored.

In the following example, `<format>PDF</format>` is ignored.

```
<documentation>
   <name>Data Access Guide</name>
   <datatype>XML</datatype>
   <audience>
      <type>External</type>
      <name>Administrator</name>
   </audience>
   <format>PDF</format>
</documentation>
```

Example: XSD Schema with anyAttribute

```
<xs:element name="documentation">
 <xs:complexType>
   <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="datatype" type="xs:string"/>
   </xs:sequence>
   <xs:anyAttribute/>
 </xs:complexType>
</xs:element>
```

- The `name` and `datatype` are mapped to columns
- The `anyAttribute` element is mapped to the `-@anyAttribute` column

Example: XML File with anyAttribute

```
<documentation myAttribute="ConnectionServer">
   <name>Data Access Guide</name>
   <datatype>XML</datatype>
</documentation>
<documentation myAttribute="DataFederator">
   <name>Data Federator User Guide</name>
   <datatype>any</datatype>
</documentation>
```

The `-@anyAttribute` column data is the element attribute name and its value. In the example, the column data contains the two following rows:

```
myAttribute="ConnectionServer"
myAttribute="DataFederator"
```

6.5.11 XML Driver - Mapping anyType and simpleType

The following sections show how `anyType` and specific arrangements of `simpleType` in XML schemas are mapped.

Mapping of anyType Type

The whole content of an element of `anyType` type is mapped to a table column.

Example: **anyType**

In the Club Demo example, the following `Address` element is mapped to the `Address` column.

```
<xs:element name"Address" type="xs:anyType"/>
```

Mapping of Lists of Values

Independently of the base data type of the value, a list of values is always mapped to VARCHAR.

Example: **List of Data Types**

The base data type is an integer. The data type of the list of integers is VARCHAR.

```
<intvalues>100 34 56 -23 1567</intvalues>
```

Mapping of Restrictions to a Type

The value of an element can be restricted to a certain range. This does not impact the data type conversion.

Example: **Restriction**

```
<xs:element name="age">
 <xs:simpleType>
  <xs:restriction base="xs:integer">
   <xs:minInclusive value="0"/>
   <xs:maxInclusive value="100"/>
  </xs:restriction>
 </xs:simpleType>
</xs:element>
```

Mapping of Unions of Types

The XML data type to which the union is mapped is string, that is, CHAR or VARCHAR data types in data foundations.

Example: **Union of Types**

```
<xs:element name="jeans_size">
 <xs:simpleType>
```

```
  <xs:union memberTypes="xs:string xs:int"/>
 </xs:simpleType>
</xs:element>
```

6.5.12 XML Driver - Configuring Column Maximum Size

The `cs.cfg` configuration file provides the following parameters:

- `Binary Max Length` to set the maximum size of table columns whose data type is binary
- `Integer Max Length` to set the maximum size of table columns whose data type is integer, nonPositiveInteger, negativeInteger, noNegativeInteger, and positiveInteger
- `String Max Length` to set the maximum size of table columns whose data type is string or any of the following: anyURI, QName, NOTATION, duration, gYearMonth, gYear, gMonthDay, gDay, gMonth, TYPE_NORMALIZEDSTRING, token, language, Name, NCName, XSD_TYPE_ID, IDREF, IDREFS, ENTITY, ENTITIES

Related Topics

- Binary Max Length
- Integer Max Length
- String Max Length

6.6 Web Service Driver

The Data Access layer allows the BI platform to connect to services available on the Internet over HTTP or FTP. It provides a data access driver called Web Service driver to communicate with web services using SOAP 1.1 messages.

Web services are defined in documents which describe data types, messages, and bindings with the Web Services Description Language (WSDL) 1.1.

The Web Service driver supports web services of SOAP 1.1 binding and `document` or `rpc` style with `literal` body. For example:

```
<wsdl:operation ...>
   <soap:operation ... style="document"/>
   <wsdl:input>
      <soap:body use="literal"/>
   </wsdl:input>
   ...
</wsdl:operation>
```

or

```
<wsdl:operation ...>
   <soap:operation ... style="rpc"/>
```

```
   <wsdl:input>
      <soap:body use="literal"/>
   </wsdl:input>
   ...
</wsdl:operation>
```

Remember:
The driver does not support other SOAP protocol versions and services of styles other than `document` and `rpc`.

The connection wizard provides a workflow to enter the necessary information to connect to web services. Web services show up under `Web Services (WSDL 1.1)` DBMS and `Web Services Connector` NetworkLayer.

You can find the driver configuration files in the *`connectionserver-install-dir`*`\connectionServer\xml` directory. The `xml.prm` file lists the data access driver capabilities in terms of database operations and functions.

6.6.1 Web Service Driver Capabilities

The XML and Web Service drivers share the same capabilities.

Related Topics

- XML Driver Capabilities

6.6.2 Web Service Location

You set the Web Service location using the **Web Service URL** parameter in the connection wizard, with the following information:

- User name and password if required for authentication
- Proxy server address
- Proxy user name and password if required for authentication

The present release also supports the HTTPS and FTPS encrypted protocol based on one certificate.

Remember:
The Web Service driver only supports the web service definition document and the associated services hosted on the same HTTP server.

Example:

http://wsf.cdyne.com/WeatherWS/Weather.asmx?WSDL

6.6.3 Web Service Definition Example

The following example illustrates the `ClubDemo.wsdl` document which defines the CustomerService web service of `document` style. The sample URL of this web service is `http://ws.myexample.com/ClubDemo/ClubDemo.wsdl`.

Example:

The WSDL document defines the following details:

- The `CustomerService` service
- The `Customer` port
- The `GetCustomer` binding
- The `CustomerType` portType
- The `GetCustomerList` operation
- The `GetCustomerListByCountryRequest` input message, which contains the `Country` element
- The `GetCustomerListByCountryResponse` output message, which contains the `GetCustomerListReturn` element of `Customer` complex type

```
<wsdl:definitions targetNamespace="http://ws.myexample.com/ClubDemo/"
                  xmlns:apachesoap="http://xml.apache.org/xml-soap"
                  xmlns:impl="http://ws.myexample.com/ClubDemo/"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                  xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
                  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">

   <wsdl:types>
      <schema targetNamespace="http://ws.myexample.com/ClubDemo/">

         <complexType name="Customer">
            <sequence>
               <element name="Name" maxOccurs="2">
                  <complexType>
                     <simpleContent>
                        <extension base="xsd:string">
                           <attribute name="type" type="xsd:string"/>
                        </extension>
                     </simpleContent>
                  </complexType>
               </element>
               <element name="Age" type="xsd:integer"/>
               <element name="Address" type="xsd:string"/>
               <element name="ZIP" type="xsd:string"/>
               <element name="City" type="xsd:string"/>
               <element name="Country">
                  <complexType>
                     <attribute name="name" type="xsd:string"/>
                  </complexType>
               </element>
               <element name="Invoice" type="xsd:integer" maxOccurs="unbounded"/>
            </sequence>
            <attribute name="ID" type="xsd:integer"/>
         </complexType>

         <element name="GetCustomerListByCountry">
            <element name="Country">
               <complexType>
                  <attribute name="name" type="xsd:string"/>
               </complexType>
            </element>
```

```
        </element>

        <element name="GetCustomerListResponse">
           <complexType>
              <sequence>
                 <element name="GetCustomerListReturn" type="impl:Customer" maxOccurs="unbounded"/>
              </sequence>
           </complexType>
           <key name="custkey">
              <selector xpath="GetCustomerListReturn"/>
              <field xpath="@ID"/>
           </key>
        </element>
     </schema>

   </wsdl:types>

   <wsdl:message name="GetCustomerListByCountryRequest">
      <wsdl:part name="request" element="impl:GetCustomerListByCountry"></wsdl:part>
   </wsdl:message>

   <wsdl:message name="GetCustomerListByCountryResponse">
      <wsdl:part name="response" element="impl:GetCustomerListResponse"></wsdl:part>
   </wsdl:message>

   <wsdl:portType name="CustomerType">
      <wsdl:operation name="GetCustomerList">
         <wsdl:documentation>Gets Information on ClubDemo Customers</wsdl:documentation>
         <wsdl:input message="impl:GetCustomerListByCountryRequest"/>
         <wsdl:output message="impl:GetCustomerListByCountryResponse"/>
      </wsdl:operation>
   </wsdl:portType>

   <wsdl:binding name="GetCustomer" type="impl:CustomerType">
      <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
      <wsdl:operation name="GetCustomerList">
         <soap:operation soapAction="http://ws.myexample.com/ClubDemo/" style="document"/>
         <wsdl:input>
            <soap:body use="literal"/>
         </wsdl:input>
         <wsdl:output>
            <soap:body use="literal"/>
         </wsdl:output>
      </wsdl:operation>
   </wsdl:binding>

   <wsdl:service name="CustomerService">
      <wsdl:port name="Customer" binding="impl:GetCustomer">
         <soap:address location="http://ws.myexample.com/ClubDemo/"/>
      </wsdl:port>
   </wsdl:service>

</wsdl:definitions>
```

6.6.4 Mapping Rules for Web Services

The Web Service driver uses the same table and column mapping rules as the XML driver. The driver also supports the following, specific rules to web services.

Qualifiers

The Data Access layer can connect to one web service at a time. The service is mapped to the database qualifier. In the ClubDemo example, qualifier name is `CustomerService` service name.

Owners

The `port` attribute is mapped to the database owner. In the example, owner is `Customer`.

Tables

In the Data Access layer, every operation is represented as a root. Every output message represents a child element of an operation. The Web Service driver exposes output messages as tables.

Table names are the concatenation of the names of the operation, the output message, and the XML elements contained in the response, separated with a slash (`/`).

Example:

In the ClubDemo example, the tables are the following:

- `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse`
- `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse/GetCustomerListReturn`
- `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse/GetCustomerListReturn/Name`
- `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse/GetCustomerListReturn/Invoice`

Columns and Input Columns

The XML mapping rules for columns also apply to web services.

The Web Service driver only accepts tables with a single row as input messages. Each entry of that table is mapped to an input column of the result table. The driver maps all input messages, even if they are defined as optional in the WSDL document.

Restriction:

The driver does not support `any` and `anyAttribute` in input columns.

In the following examples, the `GetCustomerListByCountryRequest/GetCustomerListByCountry` column is an input column. The other column names result from the mapping rules of XML elements. See the XML driver section for more information.

Example:

The columns of the `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse` table are the following:

- `-id`
- `GetCustomerListByCountryRequest/GetCustomerListByCountry`

Example:

The columns of the `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse/GetCustomerListReturn` table are the following:

- `Age`

- `Address`
- `ZIP`
- `City`
- `Country/@name`
- `@ID`
- `-GetCustomerListResponse/-id`
- `GetCustomerListByCountryRequest/GetCustomerListByCountry`

Example:

The columns of the `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse/GetCustomerListReturn/Name` table are the following:

- `-id`
- `Name`
- `@type`
- `-GetCustomerListReturn/@ID`
- `GetCustomerListByCountryRequest/GetCustomerListByCountry`

Example:

The columns of the `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse/GetCustomerListReturn/Invoice` table are the following:

- `-id`
- `Invoice`
- `-GetCustomerListReturn/@ID`
- `GetCustomerListByCountryRequest/GetCustomerListByCountry`

Primary Keys and Foreign Keys

The XML mapping rules for primary and foreign keys also apply to web services. In addition, an input column is always part of the definition of a primary key.

Restriction:
Although data foundations expose foreign keys, the Web Service driver does not support JOIN operations.

Example: Primary Keys

The `-id` column of the `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse` table is generated because the schema does not define any primary key.

The schema defines an explicit primary key for the `@ID` column of the `GetCustomerList/GetCustomerListByCountryResponse/GetCustomerListResponse/GetCustomerListReturn` table.

The `GetCustomerListByCountryRequest/GetCustomerListByCountry` input column is used as primary key.

Example: Foreign Keys

The `-GetCustomerListResponse/@ID` and `-GetCustomerListReturn/@ID` columns are built from the reverse path to the parent table and the primary key of the parent table.

Array Data Types

Web services support data types in the format of an array made of a series of primitive or complex types. An element of this type is mapped to a one-row table. It is mapped as part of column or table names as `-arrayElement`. The array data type does not define the element name and accepts any name.

See the link below for more information on array data types.

Example: Array Data Types

The `arrayType="xsd:int[2]"` is a data type of an element that has two occurrences and whose value is an integer. For example, the following complex type applies to the `CustomerIDs` element and defines it as parent of a `CustomerID` element that accepts integer values:

```
<complexType name="ArrayOfIntegers">
   <complexContent>
      <restriction base="soapenc:Array">
         <attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:int[2]"/>
      </restriction>
   </complexContent>
</complexType>

<element name="CustomerIDs" type="ArrayOfIntegers">
   <CustomerID>103</CustomerID>
   <CustomerID>204</CustomerID>
</element>
```

Related Topics

- Details on arrays in SOAP 1.1 note
- XML Driver - Mapping Columns
- XML Driver - Mapping Primary and Foreign Keys

6.6.5 Web Service Driver - Configuring Column Maximum Size

Like the XML driver, the Web Service driver also uses the `Binary Max Length`, `Integer Max Length` and `String Max Length` parameters in the `cs.cfg` file to deal with large values of these data types.

Related Topics

- Binary Max Length
- Integer Max Length

- String Max Length

6.7 Local Disk Used as a Cache for Sorting Operations

The CSV OpenDriver and the OData, SAP ERP, XML and Web Service drivers can use the local disk as a cache for sorted rows.

The data access drivers perform the sorting operations (ORDER BY, GROUP BY and DISTINCT) as follows:

- In the main memory if there is a small amount of data to be sorted
- By writing the data in a temporary directory on the local disk

The sorted rows are written on the disk either when the free main memory is too small, or when the number of rows defined using the `Bucket Split Size` SBO parameter is reached.

The local disk is used as a cache and the data is written in the default temporary folder of your machine. You can configure the directory path in the `cs.cfg` file using the `Temp Data Dir` parameter when the operating system does not specify a temporary folder.

Related Topics

- CSV OpenDriver
- OData Driver
- SAP ERP Driver
- XML Driver
- Web Service Driver
- Bucket Split Size
- Temp Data Dir

Connection Reference

7.1 HIVE Connections

The following sections describe how to create connections to HIVE databases. In this release, you can create connections to different versions of the HIVE databases simultaneously.

7.1.1 To Create a Connection to Apache Hadoop HIVE

The Data Access layer allows the BI platform to connect to the Apache Hadoop HIVE 0.7.1, 0.8.0, 0.9.0, and 0.10 databases through JDBC on all platforms.

1. To create a connection to the Apache Hive Thrift server, place the following sets of JAR files in the *connectionserver-install-dir*\connectionServer\jdbc\drivers*hadoop_version* directory, where *hadoop_version* is hive07, hive08, hive09 or hive010.

Apache Hadoop HIVE Database Version	JAR Files
0.7.1	hadoop-0.20.1-core.jar or hadoop-core-0.20.2.jar hive-exec-0.7.1.jar hive-jdbc-0.7.1.jar hive-metastore-0.7.1.jar hive-service-0.7.1.jar libfb303.jar log4j-1.2.16.jar commons-logging-1.0.4.jar slf4j-api-1.6.1.jar slf4j-log4j12-1.6.1.jar

Apache Hadoop HIVE Database Version	JAR Files
0.8.0	hadoop-0.20.1-core.jar or hadoop-core-0.20.2.jar hive-exec-0.8.0.jar hive-jdbc-0.8.0.jar hive-metastore-0.8.0.jar hive-service-0.8.0.jar libfb303.jar log4j-1.2.16.jar commons-logging-1.0.4.jar slf4j-api-1.6.1.jar slf4j-log4j12-1.6.1.jar
0.9.0	hadoop-0.20.1-core.jar or hadoop-core-0.20.2.jar hive-exec-0.9.0.jar hive-jdbc-0.9.0.jar hive-metastore-0.9.0.jar hive-service-0.9.0.jar libfb303.jar log4j-1.2.16.jar commons-logging-1.0.4.jar slf4j-api-1.6.1.jar slf4j-log4j12-1.6.1.jar

Apache Hadoop HIVE Database Version	JAR Files

0.10.0	`hadoop-0.20.1-core.jar` or `hadoop-core-0.20.2.jar`
	`hive-exec-0.10.0.jar`
	`hive-jdbc-0.10.0.jar`
	`hive-metastore-0.10.0.jar`
	`hive-service-0.10.0.jar`
	`libfb303.jar`
	`log4j-1.2.16.jar`
	`commons-logging-1.0.4.jar`
	`slf4j-api-1.6.1.jar`
	`slf4j-log4j12-1.6.1.jar`

2. Run the connection wizard.
3. Select the Apache Hadoop HIVE driver version you want to use.
4. Use the wizard to configure the connection.

You have created a connection to the Apache Hadoop HIVE database.

Note:
The connections created from the BI platform 4.1 continue to work in this release. The `Apache Hadoop HIVE` database name is an alias of `Apache Hadoop Hive 0.7`. The `hive` folder for JAR files is still supported.

Related Topics

- JAR File Location Reference

7.1.2 To Make HIVE Connections Work After Platform Update

As a prerequisite, your production environment runs an SAP BusinessObjects BI platform 4.0 release up to Support Package 8 on a Microsoft Windows or UNIX operating system.

After the update of the platform servers or client tools to a 4.1 release, the Apache Hadoop HIVE driver has been uninstalled. You must modify the installation of the platform to make Apache Hadoop HIVE connections work.

The following procedure shows the steps to perform on Microsoft Windows.

1. In the Control Panel, locate the latest full installation program of the BI platform.

   Remember:
   You cannot perform the next step on a program update of the platform.

2. Right-click and select **Uninstall/Change**.
3. In the "Application Maintenance" screen of the "SAP BusinessObjects BI platform setup" dialog box, select **Modify** and click **Next**.
4. In the "Select Features" screen, select the **Hadoop HIVE** option under **Database Access and Security**.
5. Click Next and complete the installation.

You have installed the Apache Hadoop HIVE driver on the platform. You can now create connections to the HIVE databases.

7.1.3 To Create a Connection to Amazon EMR HIVE

The Data Access layer allows the BI platform to connect to the Amazon Elastic MapReduce (EMR) HIVE 0.7 and 0.8 databases through JDBC on all platforms.

1. Set up an SSH tunnel to the Amazon master node.

   See the Amazon documentation for more information.

2. Download the required JAR files from the Amazon website and place them in the `connectionserver-install-dir\connectionServer\jdbc\drivers\emrhive_version` directory, where `emrhive_version` is either `emrhive07` or `emrhive08`.
3. In the connection wizard, specify localhost for server and 10002 for port in the **Server (host:port)** field.

You have created a connection to the Amazon EMR HIVE database.

Note:
The connections created from the BI platform 4.1 continue to work in this release. The `Amazon EMR HIVE` database name is an alias of `Amazon EMR Hive 0.7`. The `hive` folder for JAR files is still supported.

Related Topics

- Creating an SSH tunnel to the master node
- HIVE 0.7.1 JAR file download website
- HIVE 0.8.1 JAR file download website
- JAR File Location Reference

7.2 IBM DB2 Connections - Reference Keys Mapped to Null

Restriction:
Due to a database driver restriction, reference key names of IBM DB2 10 for z/OS database tables are mapped to NULL.

7.3 IBM Informix Connections

The following sections describe the actions to perform when problems with IBM Informix connections happen.

Related Topics

- To Set the JVM Timezone for IBM Informix Connections
- To Set the Transaction Mode for IBM Informix Connections

7.3.1 To Set the JVM Timezone for IBM Informix Connections

In the information design tool, queries to IBM Informix Dynamic Server 11 using JDBC may return wrong date type values if the client tool and server are located in different timezones. To avoid this problem, you must set the timezone value of the JVM to the same timezone set at the database server level.

1. Exit the information design tool.
2. Go to the *bip-install-dir*`\win32_x86` directory.
3. Open the `InformationDesignTool.ini` for editing.
4. Add the following line at the end of the file:

   ```
   -Duser.timezone=server_timezone_value
   ```

   where *`server_timezone_value`* is the timezone value set on the database server. For example: `GMT`.
5. Save the file.
6. Restart the application.

When you complete this task, connections to an IBM Informix database using JDBC are properly configured.

7.3.2 To Set the Transaction Mode for IBM Informix Connections

The Data Access layer allows SQL operations to be run as a transaction block by default. If the Informix Dynamic Server does not support transactions, you may expect connection failure. To avoid this problem, specify in the driver configuration file that the transaction mode is not available.

1. Exit the client tool.
2. Open the `informix.sbo` file for editing.

   It is located in the *`connectionserver-install-dir`*`\connectionServer\jdbc` directory.
3. Add the following line under the appropriate `Database` section:

```
<Parameter Name="Transactional Available">No</Parameter>
```

4. Save the file.
5. Restart the service and application.

When you complete this task, connections to IBM Informix are properly configured.

Related Topics

- Transactional Available

7.4 MS Analysis Services Connections

Caution:
Connections to MS Analysis Services through XMLA do not use Connection Server.

This section only relates to connections to be created in the information design tool.

In the information design tool, users can create connections to MS Analysis Services over HTTP through an XMLA driver.

To configure these connections, you first need to enable the HTTP access. For more information, refer to the Microsoft TechNet website.

Related Topics

- http://technet.microsoft.com/en-us/library/cc917711.aspx
- http://technet.microsoft.com/en-us/library/cc917712.aspx

7.5 MS SQL Server Connections

The following sections provide information for MS SQL Server connections.

Related Topics

- Synonym Support for OLE DB Connections to MS SQL Server

  Synonym columns do not display in reporting tools.

- To Set the JVM Options for Connections to MS SQL Server on UNIX

  After you install the JDBC driver for MS SQL Server on UNIX and configure the `jdbc.sbo` file, you set the JVM information in the `cs.cfg` file.

7.5.1 Synonym Support for OLE DB Connections to MS SQL Server

Synonym columns do not display in reporting tools.

MS SQL Server Native Client middleware does not support metadata of Synonym columns. Consequently, Synonym columns from the MS SQL Server database do not display in the universe design tool or information design tool when connecting through OLE DB.

Related Topics

- SQL Server Native Client (OLE DB)

7.5.2 To Set the JVM Options for Connections to MS SQL Server on UNIX

After you install the JDBC driver for MS SQL Server on UNIX and configure the `jdbc.sbo` file, you set the JVM information in the `cs.cfg` file.

1. Open the `cs.cfg` file for editing.
2. Add the following file path to the `JavaVM` section in order to overwrite the default JVM configuration:

```
<LibraryName JNIVersion="JNI_VERSION_1_4">$BOBJEDIR/jdk/jre/lib/i386/server/libjvm.so</LibraryName>
```

3. Add one of the following options to the `JavaVM` section:

Option	Description

Solaris or Linux	`<Options>`
	`  <Option>-Xmx512m</Option>`
	`</Options>`

AIX	`<Options>`
	`  <Option>-Xmx511m</Option>`
	`</Options>`

Note:
You can allow 512 MB or higher to the Java Virtual Machine on Solaris or Linux, but no more than 511 MB on AIX.

4. Save the file.

7.6 Oracle Connections

The following sections provide configuration information for Oracle connections.

Related Topics

- Connections to Clusters of Oracle Servers
- To Set the Oracle CURSOR_SHARING Parameter Value

7.6.1 Connections to Clusters of Oracle Servers

Connections to clusters of Oracle servers that work in failover mode is supported through JDBC. The driver chooses one of these servers, and if it is not available, the next server is used. A connection error raises if all servers are unavailable.

In the connection wizard, enter the server host and port separated by a semi-colon in **Server(s) (host:port{,host:port})** to create a JDBC connection. Enter information for either one server or multiple servers if you want to take advantage of the failover mechanism.

7.6.2 To Set the Oracle CURSOR_SHARING Parameter Value

The databases Oracle 10 and higher use the `CURSOR_SHARING` parameter. The BI platform and SAP BusinessObjects reporting tools support only the `EXACT` value for `CURSOR_SHARING`.

If you use the `FORCE` value, this may cause some performance issues of the reporting tools.

- Do one of the following:

Option	Description
To set the `CURSOR_SHARING` value to the universe connection	a. Open the universe connection for editing. b. Click **Next** to go to the "Custom Parameters" screen. c. Click **Add Parameter**. d. Enter ConnectInit and click **OK**. e. Enter "alter session set CURSOR_SHARING=EXACT" as `ConnectInit` value. f. Click **Finish**.
To set the `CURSOR_SHARING` value to all Oracle connections	a. Browse to the directory that stores the `oracle.prm` file: *`connectionserver-install-dir`*`\connectionServer\Oracle`. b. Open the file in an XML editor. c. Under the `Configuration` section, add the following parameter: `<Parameter Name="CURSOR_SHARING">EXACT</Parameter>` d. Save the file. e. Restart the connectivity and reporting services from the CMC.

Note:
For the definition of `CURSOR_SHARING`, see the Oracle documentation.

7.7 Oracle EBS Connections

The Data Access layer provides a new driver for connecting SAP BusinessObjects applications to Oracle E-Business Suite (EBS) through Oracle Call Interface (OCI). It allows applications to access data from EBS views and stored procedures. You can only use OCI to connect the driver to Oracle EBS.

When creating a connection in the universe design tool or information design tool, you first choose the authentication mode, which is either user-supplied if the application user provides user name and password when logging in to the BI platform, or single sign-on if the user logs in with Oracle EBS

credentials. You then choose the Oracle EBS application, responsibility and security group, which define the data source of your universe.

The BI platform provides SSO for Oracle EBS with the help of an authentication plugin. SSO is enabled when the plugin is installed and configured properly. SSO allows the application user to login into the BI launch pad with Oracle EBS credentials. Same credentials are used by Connection Server to access Oracle EBS data source. For more information about the authentication plugin, refer to *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

7.8 Oracle Essbase Connections

The Data Access layer allows the BI platform to access the Oracle Essbase data source.

Connections to Oracle Essbase are OLAP connections.

You can create `.unv` universes based on Oracle Essbase 9 or 11 through the Essbase Client middleware on 32-bit MS Windows or a 64-bit UNIX operating system . These connections work with 32-bit or 64-bit Connection Server. On 64-bit MS Windows, the Data Access layer allows you to work with 32-bit Connection Server. See the *64-bit Microsoft Windows Support* section.

You can create `.unx` universes based on Oracle Essbase 11 through ESSJAPI or the Native API in a 32-bit or 64-bit environment. These connections work with the OLAP Client.

Caution:

Because of the restriction in the ESSJAPI middleware for `.unv` universes, a specific deployment of the BI platform server is required to support both `.unv` and `.unx` universes based on Oracle Essbase 11. To deploy a server that supports both `.unv` and `.unx` Essbase universes, configure a system with the two following clusters on two separate machines:

- In a 64-bit environment, install the following software on the first cluster (Cluster1): the 64-bit BI platform, the Central Management Server, Web Intelligence servers, a 64-bit Connection Server, and all the 64-bit middleware drivers.
- In a 32-bit environment, install the following software on the second cluster (Cluster2): the 32-bit Connection Server, and the 32-bit middleware drivers (including the 32-bit Essbase driver for `.unv` universes).

At runtime, Connection Server uses Cluster2 for `.unv` universes. The OLAP Client, which serves `.unx` universes, uses the 64-bit middleware available in Cluster1.

Related Topics

- 64-bit Microsoft Windows Support

7.9 Oracle RAC Connections

The Data Access layer allows the BI platform to connect to Oracle Real Application Clusters (RAC) through JDBC.

To create a connection from your application using the connection wizard, you need to provide the Oracle RAC data source entry, which has the following format:

```
<host>:<port>,<host>:<port>,...,<host>:<port>
```

The number of host and port pairs depends on the number of machines involved in the cluster.

Example:

```
pmrac1.us.oracle.com:1521,pmrac2.us.oracle.com:1521
```

7.10 salesforce.com Connections - To Configure the Environment

You must first configure your environment to make salesforce.com connections work either in the universe design tool or in the information design tool.

1. Stop the Connectivity Service from the CMC and the client application.
2. Run `regedit.exe` in the MS Windows command line to open the Registry Editor.
3. Go to the Salesforce ODBC data source under **HKEY_LOCAL_MACHINE** > **SOFTWARE** > **ODBC** > **ODBC.INI** > ***<SALESFORCE_DSN_NAME>***, where *SALESFORCE_DSN_NAME* is the salesforce.com data source name.
4. Right-click the data source name and select **New** > **String Value**.
5. Enter CheckJVMChanged.
6. Double-click the property and enter 0.
7. Close the Registry Editor.

You must then follow the steps below to complete the configuration.

Related Topics

- To Make salesforce.com Connections Work in the Information Design Tool
- To Make salesforce.com Connections Work in the Universe Design Tool

7.10.1 To Make salesforce.com Connections Work in the Information Design Tool

1. Exit the information design tool.
2. Copy the `sforce.jar` driver file to the *`bip-install-dir`*`\win32_x86\jdk\jre6\lib\ext` directory.
3. Go to the *`bip-install-dir`*`\win32_x86` directory.
4. Open the `InformationDesignTool.ini` for editing.
5. Add the following line at the end of the file:

```
-Dosgi.parentClassloader=ext
```

6. Save the file.
7. Restart the application.

When you complete this task, connections to salesforce.com are properly configured.

7.10.2 To Make salesforce.com Connections Work in the Universe Design Tool

1. Exit the universe design tool.
2. Open the `cs.cfg` file for editing.
3. In the `JavaVM` section, add the following path:

```
<ClassPath>
   <Path>C:\Program Files (x86)\Progress\DataDirect\Connect_for_ODBC_61\java\lib\sforce.jar</Path>
</ClassPath>
```

4. Save the file.
5. Open the `openaccess.sbo` file for editing.

   It is located in the *`connectionserver-install-dir`*`\connectionServer\odbc` directory.
6. Add the following line under the `Database` section:

```
<Parameter Name="Transactional Available">No</Parameter>
```

7. Save the file.
8. Restart the service and application.

When you complete this task, connections to salesforce.com are properly configured.

Related Topics

- Transactional Available

7.11 SAP BW Connections

The Data Access layer allows the BI platform to access the SAP BW data source.

Connections to SAP BW through BAPI are OLAP connections.

This present release allows you to create `.unv` universes based on SAP BW on 32-bit or 64-bit MS Windows, and 64-bit UNIX flavors.

SAP BW connections can work either with 32-bit Connection Server for 32-bit operating systems or with 64-bit Connection Server for 64-bit operating systems.

7.11.1 To Enable SAP BW 64-bit Connections

You can use SAP BW connections with a 64-bit Connection Server to build `.unv` universes.

You can install the SAP BW 64-bit driver on an SAP BusinessObjects BI platform on 64-bit Microsoft Windows.

- Follow the steps below according to the type of installation you perform:
  - If you perform a full installation of this release, SAP BW connections will use the 64-bit Connection Server.
  - If you perform a patch installation over an existing platform, SAP BW connections will continue to use the 32-bit Connection Server. To use the 64-bit Connection Server, modify the installation of the platform:

  a. In the Control Panel, locate the latest full installation program of the BI platform.

     Remember:
     You cannot perform the next step on a program update of the platform.

  b. Right-click and select **Uninstall/Change**.
  c. In the "Application Maintenance" screen of the "SAP BusinessObjects BI platform setup" dialog box, select **Modify** and click **Next**.
  d. In the "Select Features" screen, select the **SAPBW64** option under **Database Access**.
  e. Click **Next** and complete the installation.

You have installed the SAP BW 64-bit driver on the platform. You can now create SAP BW connections that use the 64-bit Connection Server on 64-bit Microsoft Windows.

7.12 SAP ERP Connections - Driver Failed to Load

The following error message may appear in SAP BusinessObjects Web Intelligence:

```
Database error: (CS) "Java Exception : java.lang.NoClassDefFoundError:
 com/sap/bip/jco/JCoManager : cannot initialize class because prior initialization attempt failed"
```

It means the SAP ERP driver fails to load, because the Web Intelligence Processing Server cannot find the `sapjco3.dll` file.

To make SAP ERP connections work, you must copy the file from the BI platform server directory (*`bip-install-dir`*`\win64_x64`) to the MS Windows directory (`C:\WINDOWS\system32`). You then restart the Web Intelligence Processing Server.

7.13 SAP HANA Connections

The Data Access layer allows the BI platform to connect SAP HANA database 1.0 SPS 07 using ODBC and JDBC on all platforms. The present release also supports OLAP connections to the SAP HANA database on MS Windows, Linux and AIX.

Array Fetch Size

The `Array Fetch Size` default value is `1000` for SAP HANA connections.

Caution:
Because a high number for `Array Fetch Size` requires more memory, this configuration may affect your system performance.

Failover

Connections to clusters of SAP HANA servers that work in failover mode are supported through ODBC and JDBC. The driver chooses one of these servers, and if it is not available, the next server is used. A connection error occurs if all servers are unavailable.

Preferred Viewing Locale

SAP HANA connectivities support Preferred Viewing Locale (PVL) for all authentication modes through JDBC and ODBC. The PVL allows the user to retrieve localized data if the database stores data in different languages.

Single sign-on to database

The BI platform provides single sign-on (SSO) to database using either Windows AD with Kerberos or the SAML protocol on MS Windows and Linux platforms. SSO with Kerberos is available for JDBC and ODBC connections. SSO with SAML is available in JDBC, but not in ODBC.

If both Kerberos and SAML implementations are available in your environment, the platform uses SAML in priority. If any problem occurs using SAML, the platform uses Kerberos instead. This logic is transparent to the application user. To deactivate the use of SAML, set the `Skip SAML SSO` parameter to `True` in the appropriate `newdb.sbo` configuration file.

The BI platform also provides SSO to database using either Windows AD with Kerberos or the SAML protocol in OLAP connections.

Once users are logged on to the BI platform from an SAP BusinessObjects application, they can perform actions that require database access, such as refreshing Web Intelligence documents, without providing their database credentials.

UNIX support

Starting from the 4.1 release, the Data Access layer allows the BI platform to connect to the SAP HANA database through ODBC on 64-bit UNIX platforms.

Use of SSL

Relational and OLAP connections to SAP HANA can use the SSL protocol.

Related Topics

- Skip SAML SSO

7.13.1 Creating an SAP HANA Connection

Creating a JDBC Connection

The SAP HANA 1.0 SPS 07 JDBC drivers are installed as part of the BI platform in the *connection server-install-dir*`\connectionServer\jdbc\drivers\newdb` directory. Consequently, you do not need to perform any additional configuration to create a connection to the SAP HANA database.

Select one of the following options in the connection wizard to create a JDBC connection:

- **Single Server**

  Enter the HANA host name and instance number in the **Host Name** and **Instance Number** fields.

- **Multiple Servers (Failover)**

  Enter the server host and port separated by a semi-colon in **Server (host:port{;host:port})**. Enter information for either one server or multiple servers if you want to take advantage of the failover mechanism.

The convention for port number is as follows:

```
3##15
```

where ## is the HANA instance number.

Example:

If you are connecting to instance `0`, type either the instance number `00` or the port number `30015` in the wizard. If you are connecting to instance `1`, type either the instance number `01` or the port number `30115`.

Using SSL

Select **Use SSL** in the connection wizard to create an OLAP or JDBC connection that is using the SSL protocol to connect to the SAP HANA server.

Note:
To use SSL in ODBC on MS Windows, open the ODBC Data Source Administrator and select **SSL** when you configure the DSN.

Related Topics

- Creating JDBC Connections

7.13.2 Editing an SAP HANA Connection

An SAP HANA connection opened for editing can show up under **Single Server** or **Multiple Servers (Failover)** in the connection wizard. To edit the connection, do one of the following:

- If it shows up in **Single Server**, modify the host name and instance number of the data source.
- If it shows up in **Multiple Servers (Failover)**, modify the server host and port.

7.13.3 Before Configuring Single Sign-On for SAP HANA Connections

Remember:
This section is about configuring single sign-on through Windows AD with Kerberos for SAP HANA connections.

Before setting up the necessary Java VM options, you must create the JAAS `bscLogin.conf` and Kerberos `krb5.ini` configuration files to prepare your application for Windows AD authentication. See the *SAP BusinessObjects Business Intelligence platform Administrator Guide* for more information.

Related Topics

- To Configure Single Sign-On to SAP HANA for the Information Design Tool
- To Configure Single Sign-On to SAP HANA for Web Intelligence
- To Configure Single Sign-On to SAP HANA for Web Intelligence Rich Client
- Before Configuring Single Sign-On for SAP HANA Connections

7.13.4 To Configure Single Sign-On to SAP HANA for the Information Design Tool

The following steps enable single sign-on to the SAP HANA database through JDBC from the information design tool.

- Do one of the following:

Option	Description

To configure SSO for local connections through JDBC	a. Exit the information design tool.
	b. Open the `InformationDesignTool.ini` file for editing.
	It is located at the *`bip-install-dir`*`\win32_x86` directory.
	c. Add the following lines:
	`-Djava.security.auth.login.config=C:\<location>\bscLogin.conf`
	`-Djava.security.krb5.conf=C:\<location>\Krb5.ini`
	where *`<location>`* is the configuration file directory on the machine where the Connection Server is running.
	d. Save the file.
	e. Restart the information design tool.

To configure SSO for connections located on the CMS through JDBC	The objective is to configure the Adaptive Connectivity Service.
	a. Open the CMC.
	b. Under "Connectivity Services", stop the Adaptive Connectivity Service hosted by the Adaptive Processing Server.
	c. Go to the "Properties" page.
	d. Add the following options to the "Command Line Parameters" property:
	`-Djava.security.auth.login.config=C:\<location>\bscLogin.conf`
	`-Djava.security.krb5.conf=C:\<location>\Krb5.ini`
	where *`<location>`* is the configuration file directory on the machine where the Connection Server is running.
	e. Click **Save**.
	f. Restart the service from the CMC.
	Remember:
	You also need to configure the Adaptive Connectivity Service to enable SSO with Web Intelligence Rich Client in Connected mode. These steps also allow you to configure SSO for all other Java services, such as the Data Federation services. You use Data Federation services when querying on a multisource-enabled universe built on an SAP HANA connection.

7.13.5 To Configure Single Sign-On to SAP HANA for Web Intelligence

The following steps enable single sign-on to the SAP HANA database through JDBC for SAP BusinessObjects Web Intelligence.

Note:
This procedure applies to the Java or HTML interface of SAP BusinessObjects Web Intelligence that the application user launches from the BI launch pad.

Remember:
This configuration only addresses Web Intelligence reporting services hosted by the Web Intelligence Processing Server.

1. Open the CMC.
2. Under "Web Intelligence Services", stop the reporting service hosted by the Web Intelligence Processing Server.
3. Open the `cs.cfg` file for editing.
4. In the `JavaVM` section, add the following Java VM options:

```
<Options>
 <Option>-Djava.security.auth.login.config=C:\<location>\bscLogin.conf</Option>
 <Option>-Djava.security.krb5.conf=C:\<location>\Krb5.ini</Option>
<Options>
```

   where *<location>* is the configuration file directory on the machine where the Connection Server is running.
5. Save the file.
6. Restart the service from the CMC.

7.13.6 To Configure Single Sign-On to SAP HANA for Web Intelligence Rich Client

The following steps enable single sign-on to the SAP HANA database through JDBC from the Web Intelligence Rich Client application.

- Follow one of the procedures below:
  - If you connect to Web Intelligence Rich Client from a WID document or from the Windows Start menu:
    a. Close your document and exit from Web Intelligence Rich Client.
    b. Create the following environment variables:
       - `java.security.auth.login.config=C:\<location>\bscLogin.conf`
       - `java.security.krb5.conf=C:\<location>\Krb5.ini`

       where *<location>* is the configuration file directory on the machine where Connection Server is running.
    c. Restart Web Intelligence Rich Client.

- If you use Web Intelligence Rich Client in Connected mode from the BI launch pad (also known as HTTP mode), you must configure the Adaptive Connectivity Service. Follow the steps described to configure single sign-on for the information design tool.

Related Topics

- To Configure Single Sign-On to SAP HANA for the Information Design Tool

7.13.7 To Configure the Java Virtual Machine for Instrumentation with SAP HANA Connections

You can monitor the activity of SAP HANA drivers if you install CA Wily Introscope with the BI platform. This instrumentation provides end-to-end tracing support for SAP HANA connections through ODBC and JDBC.

SAP HANA tracing is enabled by default in the platform. However, you also have to configure the Java Virtual Machine to use the JNI bridge.

1. Open the `cs.cfg` file for editing.
2. Add the following JAR file paths to the `JavaVM` section in order to load the necessary classes:

```
<ClassPath>
 <Path>"bip-install-dir\java\lib\TraceLog.jar;bip-install-dir\java\lib\external\com.sap.js.passport.api.jar"

 </Path>
</ClassPath>
```

3. Add the following options to the `JavaVM` section:

```
<Options>
 <Option>-javaagent:bip-install-dir\java\wily\Agent.jar</Option>
 <Option>-Dcom.wily.introscope.agentProfile=bip-install-dir\java\wily\IntroscopeAgent_CSJNI.profile</Option>

 <Option>-Dcom.wily.introscope.agent.agentName=CSJNIEngine</Option>
</Options>
```

4. Save the file.
5. Open the `IntroscopeAgent_CSJNI.profile` for editing.

   It is located in the *`bip_install_dir`*`\java\wily` directory.
6. Replace `localhost` with the host name of the Introscope Agent in the following line:

```
introscope.agent.enterprisemanager.transport.tcp.host.DEFAULT=localhost
```

7. Save the file.

7.14 SAP MaxDB Connections

On ODBC, make sure to use the SAP MaxDB ODBC driver version 7.7.07 (build number 07 or higher). SAP MaxDB provides ASCII and Unicode drivers for both MS Windows and UNIX. The ASCII version of the ODBC driver always connects to the database kernel with ASCII. The Unicode version of the ODBC driver connects to ASCII database kernels via ASCII and to Unicode database kernels with UCS2.

SAP MaxDB does not require a specific driver manager on UNIX. However, it can be configured to work with the following driver managers if needed:

- unixODBC 2.0.9 or higher
- iODBC 3.0.5 or higher

On JDBC, make sure to use the latest version of the `sapdbc.jar` driver. For more information about SAP MaxDB JDBC driver, refer to `maxdb.sbo` configuration file.

7.15 SAP NetWeaver BW Connections

Relational connections to SAP NetWeaver BW do not use Connection Server. These connections go through a dedicated connector and use a specific facade in SAP NetWeaver BW.

For details on how to configure and tune connections to SAP NetWeaver BW, see the *Data Federator Administration Tool Guide*.

7.15.1 Requirements for Connecting Data Federator to SAP NetWeaver BW

To connect to SAP NetWeaver BW, you must have a compatible release of SAP NetWeaver BW, as well as the appropriate SAP notes:

- The minimum required release is SAP NetWeaver BI 7.01 SP06.

  Note:
  The official name SAP NetWeaver BW changed between releases. Before release 7.3, it was called SAP NetWeaver BI.

- The required SAP note is: https://service.sap.com/sap/support/notes/1460273.

For details on supported versions of SAP NetWeaver BW, see the *Product Availability Matrix*.

7.15.2 Requirements for Making SAP NetWeaver BW Connections Work in the Information Design Tool

An external application such as the information design tool can connect to SAP NetWeaver BW if SAP Gateway permits.

To make connections work in the information design tool, ensure the SAP Gateway security settings for external programs are configured properly. For more information, see the SAP Gateway online documentation.

Related Topics

- Security Parameters - Connectivity - SAP Library

7.15.3 Requirements for Accessing Multisource-Enabled Universes on SAP NetWeaver BW

Application users require authorizations to access multisource universes based on the SAP NetWeaver BW database.

For information on the authorizations necessary for users of query and reporting applications to access multisource universes on SAP NetWeaver BW, see the SAP Note #1465871.

Related Topics

- SAP Note #1465871

7.16 SAS Connections

Connections to SAS do not use Connection Server. They use an adapted JDBC connector to SAS/SHARE data sets.

To configure these connections, you need to install the compatible JDBC driver.

For more information about configuring SAS connectors, refer to the *Data Federator Administration Tool Guide*.

7.16.1 Installing Drivers for SAS Connections

In order to use a SAS connector, you must install a driver that lets the data federation query engine connect to a SAS/SHARE server.

A SAS/SHARE server is a server that allows you to connect to SAS data sets. For more information about SAS/SHARE, see the SAS website.

The directory where you should copy the SAS JDBC driver jars on the machine where you installed the BI platform is `boe-install-dir/java/pjs/services/DataFederatorService/resources/drivers/sas`.

You must create the `drivers/sas` directories under the `resources` directory.

For details on supported versions of SAS, see the *Product Availability Matrix*.

Related Topics

- http://www.sas.com/products/share/index.html

7.17 Teradata Connections - Mapping Teradata Database to Current Owner

The following section provides configuration information for Teradata connections.

A Teradata database supports table owners, but not qualifiers. The Connection Server returns the current user as current owner for Teradata data sources. However, the data source may be configured to map a user to the database of another user. In this case, you can configure the Teradata driver to map the current owner to the database with the `Replace Current Owner With Database` parameter. You can only use this parameter with ODBC connections.

Related Topics

- Replace Current Owner With Database

Creating a Connection to Data Federator XI 3.0 Query Server

8.1 About Data Federator XI 3.0 Query Server Connections

You can create connections to tables deployed on Data Federator XI 3.0 Query Server, in order to use these tables with an SAP BusinessObjects application.

This chapter describes configuration settings you must do on Data Federator XI 3.0 Query Server and Connection Server to create connections.

You can only create connections to Data Federator XI 3.0 Query Server by using the universe design tool. This chapter also indicates the configuration that must be made to the connection wizard to be able to create connections.

JDBC Connections

There is no additional settings for creating JDBC connections. The Data Federator JDBC driver comes with the SAP BusinessObjects Business Intelligence platform 4.0 and is configured to run seamlessly with Data Federator XI 3.0 Query Server.

ODBC Connections

With ODBC connections, the configuration depends on the SAP BusinessObjects application that you use. The process differs if you are creating a connection to use with Web Intelligence Rich Client.

Caution:

SAP recommends you use a JDBC connectivity to connect SAP BusinessObjects applications to Data Federator XI 3.0 Query Server. JDBC connectivity is available on all platforms (Microsoft Windows, UNIX flavors, and Linux).

The Data Federator ODBC middleware can only be used on Microsoft Windows and because of the use of an OpenAccess ODBC-JDBC bridge, has impacts on performance.

Related Topics

- Configuring the Connection Wizard for a Data Federator JDBC or ODBC Connection
- Configuring Data Federator ODBC Connections
- Configuring Web Intelligence Rich Client Connections Using Data Federator ODBC Middleware

8.2 Configuring the Connection Wizard for a Data Federator JDBC or ODBC Connection

In order to create a connection to Data Federator XI 3.0 Query Server, you need the following information. See your Data Federator administrator to obtain these details:

- The server name and port on which the Data Federator server is running
- The name of the catalog on Data Federator Query Server

  In the connection wizard, you enter this as the name of the database to which you are connecting.
- The authentication details for the installation of Data Federator Query Server that serves the catalog to which you are connecting

In the connection wizard, from the **Database Middleware Selection** screen, you use either the **SAP BusinessObjects**, **Data Federator Server**, **JDBC Drivers** or **ODBC Drivers** middleware to create the connection.

To configure an ODBC connection to Data Federator XI 3.0 Query Server, there are some additional configurations that you must make. If you are using Web Intelligence Rich Client, the configuration changes required are different to those used by other SAP BusinessObjects applications.

8.3 Configuring Data Federator ODBC Connections

This section contains Data Federator XI 3.0 Query Server additional settings and Connection Server configuration changes for ODBC connections to all SAP BusinessObjects applications except Web Intelligence Rich Client.

The configuration details in this section refer to the following paths:

- *`data-federator-drivers-install-dir`*`\OaJdbcBridge`: the root installation directory for the Data Federator ODBC middleware. Your administrator chose this directory when running the Data Federator drivers installer.
- *`data-federator-drivers-install-dir`*`\JdbcDriver`: the root installation directory for the Data Federator JDBC middleware. Your administrator chose this directory when running the Data Federator installer.
- *`bo-install-dir`*: the root installation directory for your SAP BusinessObjects applications.

Note:
Edit the files in an XML editor to make sure your files are well-formed. After you make the configuration changes shown below, re-start your system for the changes to take effect.

Related Topics

- Configuring the Data Federator ODBC Middleware
- Configuring Connection Server for a Data Federator ODBC Connection

8.3.1 Configuring the Data Federator ODBC Middleware

Note:
This topic applies to all SAP BusinessObjects applications that use Connection Server except Web Intelligence Rich Client.

To configure the Data Federator ODBC middleware, you need to modify the `openrda.ini` file. This file is installed in the following directory:

- *data-federator-drivers-install-dir*\OaJdbcBridge\bin\iwinnt

Set parameters in the `[JavaIp]` section, as follows:

- CLASSPATH=*data-federator-drivers-install-dir*\OaJdbcBridge\oajava\oasql.jar;*data-federator-drivers-install-dir*\JdbcDriver\thindriver.jar;*bo-install-dir*\SAP BusinessObjects Enterprise XI 4.0\java\lib\ConnectionServer.jar
- JVM_DLL_NAME=*bo-install-dir*\javasdk\jre\bin\client\jvm.dll
- JVM_OPTIONS=-DODBCMode=true -Dbusinessobjects.connectivity.directory=*bo-install-dir*\SAP BusinessObjects Enterprise XI 4.0\dataAccess\connectionServer

  Note:
  Check the `openrda.ini` file to ensure that this path is not set using the `Djava.endorsed.dirs` parameter. If it is, then you must remove the path from the `Djava.endorsed.dirs` parameter.

8.3.2 Configuring Connection Server for a Data Federator ODBC Connection

Note:
This topic applies to all SAP BusinessObjects applications that use Connection Server except Web Intelligence Rich Client.

To configure Connection Server, you need to change the configuration file: *connectionserver-install-dir*\connectionServer\cs.cfg

To configure the `cs.cfg` file, set parameters under the `JavaVM` tag, as follows:

```
<ClassPath>
  <Path>\\data-federator-drivers-install-dir\\OaJdbcBridge\\oajava\\oasql.jar</Path>
  <Path>\\data-federator-drivers-install-dir\\JdbcDriver\\thindriver.jar</Path>
</ClassPath>
```

8.4 Configuring Web Intelligence Rich Client Connections Using Data Federator ODBC Middleware

When creating Web Intelligence Rich Client connections that use Data Federator ODBC middleware, you need to make the configuration changes described in this section. Without the correct configuration amendments, the connection generates errors. This type of connection is supported in a Windows environment only.

Remember:
The details below relate to Web Intelligence Rich Client only.

To create connections that use Data Federator ODBC middleware, you need to modify the configurations for:

- The ODBC middleware
- Connection Server
- The Windows RichClient registry key

The configuration details in this section refer to the following paths:

- *data-federator-drivers-install-dir*`\OaJdbcBridge`: the root installation directory for the Data Federator ODBC middleware. Your administrator chose this directory when running the Data Federator drivers installer.
- *data-federator-drivers-install-dir*`\JdbcDriver`: the root installation directory for the Data Federator JDBC middleware. Your administrator chose this directory when running the Data Federator installer.
- *bo-install-dir*: the root installation directory for your SAP BusinessObjects applications.

Note:
Edit the files in an XML editor to make sure your files are well-formed.

Related Topics

- Configuring the Data Federator ODBC Middleware for a Connection to Web Intelligence Rich Client
- Configuring Connection Server for a Web Intelligence Rich Client Connection to Data Federator
- Setting the Windows RichClient Registry Key
- Configuring Connection Server for Connections of Web Intelligence Rich Client or Universe Design Tool to Data Federator

8.4.1 Configuring the Data Federator ODBC Middleware for a Connection to Web Intelligence Rich Client

To configure the Data Federator ODBC middleware, you need to modify the `openrda.ini` file. This file is installed in the following directory:

- *data-federator-drivers-install-dir*\OaJdbcBridge\bin\iwinnt

To configure the `openrda.ini` file, set parameters in the `[JavaIp]` section, as follows:

- CLASSPATH=*data-federator-drivers-install-dir*\OaJdbcBridge\oajava\oasql.jar;*data-federator-drivers-install-dir*\JdbcDriver\thindriver.jar;*bo-install-dir*\SAP BusinessObjects Enterprise XI 4.0\java\lib\ConnectionServer.jar
- JVM_DLL_NAME=*bo-install-dir*\javasdk\jre6\bin\client\jvm.dll

  Note:
  Web Intelligence Rich Client requires JDK 6.

- JVM_OPTIONS=-DODBCMode=true -Dbusinessobjects.connectivity.directory=*bo-install-dir*\SAP BusinessObjects Enterprise XI 4.0\dataAccess\connectionServer

  Note:
  Check the `openrda.ini` file to ensure that this path is not set using the `Djava.endorsed.dirs` parameter. If it is, then you must remove the path from the `Djava.endorsed.dirs` parameter.

8.4.2 Configuring Connection Server for a Web Intelligence Rich Client Connection to Data Federator

To configure Connection Server for a Web Intelligence Rich Client connection to Data Federator, you need to change the configuration file: *connectionserver-install-dir*\connectionServer\cs.cfg

To configure the `cs.cfg` file, set parameters under the `JavaVM` tag, as follows:

```
<ClassPath>
  <Path>\\data-federator-drivers-install-dir\\OaJdbcBridge\\oajava\\oasql.jar</Path>
  <Path>\\data-federator-drivers-install-dir\\JdbcDriver\\thindriver.jar</Path>
</ClassPath>
```

8.4.3 Setting the Windows RichClient Registry Key

To modify the Windows RichClient registry key, use a tool such as regedit.

1. In the registry, locate the `HKEY_LOCAL_MACHINE\SOFTWARE\SAP BusinessObjects\Suite XI 4.0\default\WebIntelligence\RichClient` key.
2. In this key, add the following values to the `classpath` entry. As with all registry key values, the values must be separated with a semicolon.
   - *data-federator-drivers-install-dir*`\OaJdbcBridge\oajava\oasql.jar`
   - *data-federator-drivers-install-dir*`\JdbcDriver\thindriver.jar`
3. In the `RichClient\JVMOptions` key, add the following value to the `6` entry (if five entries already exist):
   - `ODBCMode=true`

8.4.4 Configuring Connection Server for Connections of Web Intelligence Rich Client or Universe Design Tool to Data Federator

It is possible to use a single configuration for creating an ODBC connection to Data Federator from the universe design tool as well as Web Intelligence Rich Client. In addition to the configuration settings described previously, do one of the following sets of instructions:

- Do the following:
  1. Run regedit tool.
  2. In the registry, locate the `HKEY_LOCAL_MACHINE\SOFTWARE\SAP BusinessObjects\Suite XI 4.0\default\ConnectionServer\Configuration` key.
  3. In this key, add the following value to the `JVM Library` entry: *bo-install-dir*`\javasdk\jre6\bin\client\jvm.dll`.
- Do the following:
  1. Open the `cs.cfg` file for editing.
  2. Locate the `JavaVM` tag.
  3. Set `LibraryName` to the same JVM directory path specified in the `openrda.ini` file:

```
...
<JavaVM>
 <LibraryName JNIVersion="JNI_VERSION_1_4">\\bo-install-dir\\javasdk\\jre6\\bin\\client\\jvm.dll</Li
braryName>
</JavaVM>
```

Caution:
The universe design tool and OpenEdge bridge must specify the same JVM directory path.

Configuring Data Access Global Parameters

9.1 About Global Parameters

You can configure the global parameter values that apply to all connections. You can do this to improve performance, or to resolve issues with the connection that arise.

Data access global parameters are maintained in the `cs.cfg` file. This is an XML file that contains Connection Server configuration parameters, and default configuration parameters that apply to all data access drivers.

To override these global settings, you can edit settings in the SBO configuration file of each driver.

Related Topics

- Configuring Driver Parameters

9.2 About the `cs.cfg` Configuration File

On Microsoft Windows, the `cs.cfg` file is stored in the following location:

- *connectionserver-install-dir*\connectionServer

In the `cs.cfg` file, you can configure parameters in the following sections only:

- `Capabilities`

  This section defines parameters that allow you to specify the use of local or remote Connection Server.

- `Settings`

  This section defines Connection Server global configuration parameters, including which drivers to load at startup in library mode.

- `JavaVM`

  This section defines the default library of the Java Virtual Machine (JVM) used by the data access layer.

- `DriverDefaults`

  This section defines parameters that apply to all data access drivers. They can be overridden for a specific driver by corresponding settings in the *<driver>*`.sbo` configuration files, where *<driver>* is the name of the data access driver to which the SBO file relates.

- `Traces`

  This section defines parameters that allow the recording of connection activity through Connection Server in log files.

The remaining `Locales` section defines the operating system charset for each available language. You must not modify the parameters in this section.

9.3 To View and Edit the `cs.cfg` File

1. Browse to the directory that stores the `cs.cfg` file. For example, on a Microsoft Windows system:

   connectionserver-install-dir`\connectionServer\cs.cfg` where *connectionserver-install-dir* is the path where your Connection Server software is installed.

2. Open `cs.cfg` in an XML editor.
3. Expand sections as required.
4. Set parameters by either adding new parameters and values, or modifying existing parameter values.
5. Check that the document is valid against the DTD, then save and close the file.

Remember:
You must restart Connection Server after editing the `cs.cfg` file.

9.4 Configuring the Global Settings Parameters

The `Settings` section of `cs.cfg` defines settings that apply to all drivers, and cannot be customized for individual data access drivers.

To view or edit parameters, open `cs.cfg` in an XML editor and go to the `Settings` section. In the file, each parameter is defined in the following tag:

```
<Parameter Name="parameter">value</Parameter>
```

where *parameter* is the name of the parameter, and *value* is the value to which the parameter is set.

Each parameter is shown with the following information:

- Example of how the parameter appears in the XML file

- Description of the parameter
- Possible values that can be set for the parameter (where applicable)
- Default value for the parameter

Note:
Some settings can only be modified from the Central Management Console (CMC). For more information about it, refer to the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

9.4.1 Charset List Extension

```
<Parameter Name="CharSet List Extension">crs</Parameter>
```

Description	**Note:** Do not change this setting. Sets the file extension for character set files.
Default	`crs`

9.4.2 Config File Extension

```
<Parameter Name="Config File Extension">sbo</Parameter>
```

Description	**Note:** Do not change this setting. Sets the file extension for general configuration files.
Default	`sbo`

9.4.3 Description Extension

```
<Parameter Name="Description Extension">cod</Parameter>
```

Description	**Note:** Do not change this setting. Sets the file extension for the connection description files.
Default	`cod`

9.4.4 Ignore Driver Load Failure

```
<Parameter Name="Ignore Driver Load Failure">Yes</Parameter>
```

Description	Determines action taken when a driver fails to load. The parameter lets you choose whether you want a usable connection possibly without all drivers operating, or a fatal error and no functionality when a driver fails to load. **Note:** This parameter is ignored in server deployment mode.
Values	`Yes`: Connection Server generates a warning message when a driver fails to load. `No`: Connection Server generates a fatal error when a driver fails to load.
Default	`Yes`

Related Topics

- Load Drivers On Startup

9.4.5 Load Drivers On Startup

```
<Parameter Name="Load Drivers on Startup">No</Parameter>
```

Caution:
`Load Drivers On Startup` only applies to library mode.

Description	Determines how driver libraries are loaded.
Values	• `Yes`: all installed drivers are loaded during the initialization phase. • `No`: drivers are loaded on demand.
Default	`No`

9.4.6 Max Pool Time

```
<Parameter Name="Max Pool Time">-1</Parameter>
```

Caution:
`Max Pool Time` is available for library mode only.

You can change connection pool timeout for server mode on the "Properties" page of the server in the CMC. See the *SAP BusinessObjects Business Intelligence platform Administrator Guide*. Value applies to Connection Server stand alone server installations. For information on deploying Connection Server on a dedicated node, see the *SAP BusinessObjects Business Intelligence platform Planning Guide*.

Description	Determines the maximum length of time that an unused connection can remain idle in the connection pool. This represents an upper bound for connection lifetimes. A connection pool is a mechanism used by data access drivers to re-use database connections, to make the best use of system resources. You can set `Max Pool Time` for the library deployment mode only. Value applies to nodes that have Connection Server installed with server products.
Values	`-1`: No timeout, keep alive for the whole session. `0`: Connection not managed by the pool. `>0`: Idle lifetime (in minutes).
Default	`-1`

Related Topics

- Connection Pools

9.4.7 Setup File Extension

```
<Parameter Name="Setup File Extension">setup</Parameter>
```

Description	**Note:** Do not change this setting. Sets the extension for data access driver setup files.
Default	`setup`

9.4.8 SQL External Extension

```
<Parameter Name="SQL External Extension">rss</Parameter>
```

Description	**Note:** Do not change this setting. Sets the file extension for external SQL files.
Default	`rss`

9.4.9 SQL Parameter Extension

```
<Parameter Name="SQL Parameter Extension">prm</Parameter>
```

Description	**Note:** Do not change this setting. Sets the file extension for SQL parameter files.
Default	`prm`

9.4.10 Strategies Extension

```
<Parameter Name="Strategies Extension">stg</Parameter>
```

Description	**Note:** Do not change this setting. Sets the extension for strategy files.
Default	`stg`

9.4.11 Validate Configuration Files

```
<Parameter Name="Validate Configuration Files">No</Parameter>
```

Description	Triggers the validation of the configuration files (SBO, COD, and PRM) against the XML schema. **Note:** You can only validate configuration files when using Java-based connectivities.
Default	`No`

9.4.12 Validate XML Streams

```
<Parameter Name="Validate XML Streams">No</Parameter>
```

Description	Triggers the validation of the XML streams parsed within Connection Server implementation against the XML schema (for example: connection definition). **Note:** You can only validate XML streams when using Java-based connectivities.
Default	`No`

9.5 Setting the Deployment Mode

The `Capabilities` section helps you specify the deployment mode of Connection Server to be used at runtime.

Beneath `Capabilities` section, `Local` name means connectivity service is locally provided, as an in-proc library embedded in the client process. `Remote` name means Connection Server is provided on a remote server.

You can enable these deployment modes independently by setting the `Active` attribute. Specific to `Local` capability, the `EnableJNI` attribute enables the use of Java Native Interface (JNI).

Note:
A two-way JNI embedded in Connection Server allows APIs to work with Connection Server core developed in a different language. This means you can use the Java API to work with the native core and vice-versa.

Example: **Default Values**

The following configuration allows the library deployment mode with JNI and the server deployment mode.

```
<Capabilities>
 <Capability Name="Local" EnableJNI="Yes" Active="Yes"/>
 <Capability Name="Remote" Active="Yes"/>
</Capabilities>
```

Example: Connection Server in Server Mode

The following configuration allows you to use the Connection Server in server mode only.

```
<Capabilities>
 <Capability Name="Local" EnableJNI="Yes" Active="No"/>
 <Capability Name="Remote" Active="Yes"/>
</Capabilities>
```

In this mode, only the drivers installed on the back-end system of the BI platform can be used to create connections.

If a server instance is running, you can create a local connection based on a driver installed on the back-end system of the platform. If you stop the server instance, you cannot create any connections, because this setting does not allow you to use a driver installed on the client machine.

Related Topics

- Connectivity Services

9.6 Configuring the Deployment Mode

Parameters defined in the `Settings` section of the `cs.cfg` file control the deployment mode.

Library mode

Parameters in the `Library` section control library mode.

Server mode

You manage this mode from the CMC of your SAP BusinessObjects Enterprise installation. Parameters displayed on the "Properties" page of the server control the CORBA access. See the *SAP BusinessObjects Business Intelligence platform Administrator Guide* for more information.

9.7 Configuring the Drivers to Load

The default behavior of Connection Server is to load all available drivers. However, you can select the data sources to be exposed by Connection Server.

Library mode

Uncomment the `ActiveDataSources` section beneath `Library` in the `cs.cfg` file and specify the network layers and databases you want to load. For example:

```
<Library>
 <ActiveDataSources>
  <NetworkLayer Name="ODBC">
```

```
   <DataBase Name="MS SQL Server.*$"/>
  </NetworkLayer>
  <NetworkLayer Name="Oracle OCI">
   <DataBase Name="Oracle 10"/>
  </NetworkLayer>
 </ActiveDataSources>
</Library>
```

Note:
Database names can be regular expressions if they are pure ASCII. Patterns use GNU regexp syntax. Use the `.*` pattern to match any character. For more information about regular expressions, refer to the PERL website at http://www.perl.com/doc/manual/html/pod/perlre.html#Regular_Expressions.

Server mode

Select the data sources in the **Active Data Sources** section of the server properties in the CMC.

Caution:
To avoid connection failures, you must make sure your environment provides all the necessary middleware, otherwise you must set the network layers and databases you only need in the **Active Data Sources** section.

By allowing server specialization, this setting also addresses complex deployment scenarios involving multiple Connection Servers in server mode. For more information about complex deployment scenarios, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

9.7.1 Setting One Connectivity per Machine

In complex deployment scenarios, you may want to set up one connectivity per type of machine, for example you want to connect applications that come with the BI platform to an MS SQL Server database installed on Microsoft Windows and to an Oracle database installed on a UNIX machine.

To avoid connection failures, you need to do one of the following:

- When doing a custom installation of the BI platform, specify the connectivities you want to deploy for each type of database you require. You do this when selecting features you would like to install.
- When configuring Connection Server in the `cs.cfg` file, set the `ActiveDataSources` section for the library mode (`Library` parent element) as you do for the server mode in the CMC. Because applications first try to establish connections through drivers installed locally, you should configure the driver filter for both modes in the same way.

9.8 Configuring the CORBA Access Protocols

You set the CORBA access protocols in the CMC of your BI platform installation. Protocols define values that Connection Server uses to process requests coming from CORBA clients or HTTP clients.

For more infomation about the CMC, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

9.9 Activating Logs and Traces of Connection Server and Drivers

SAP differentiates between two types of messages:

- A trace message analyzes the system in detail from a developer's point of view as an exceptional procedure.
- A log message is a permanent record of the events and status of a system.

Log messages are intended for system administrators, while traces are for developers. You can enable logs and traces for all connectivities supported by the Data Access layer (for example, ODBC, JDBC, Javabean, OLE DB, OCI, and JCo).

The Connection Server logging and tracing process relies on the SAP BusinessObjects TraceLog logging and tracing facility. It helps to log and trace the following processes:

Logs	• Startup and stop of Connection Server server instances • Single sign-on initialization • Driver loading
Traces	• Connection Server initialization and configuration • Job and dictionary creation and execution • Driver loading and activity • Startup and stop of Connection Server server instances • Initialization and activity of services of Connection Server server instances • Single sign-on initialization

9.9.1 About the `_trace.ini` Configuration File

You set the logging and tracing levels in the `_trace.ini` configuration file. For example, you can set the following information in the `BO_trace.ini` file:

```
sap_log_level = log_info;
sap_trace_level = trace_debug;

if
(process == "cms")
   {
   log_level = error; // but only log errors for the CMS
   }
```

File content

In the example above, the tags defined are the following:

- `sap_log_level`, which defines the logging level
- `sap_trace_level`, which defines the tracing level
- A piece of code to specify the logging or tracing level for a process. The code is C-based or Java-based and can contain expressions, simple statements, and if-else statements.

The following tables describe the configuration values accepted by the `sap_log_level` and `sap_trace_level` tags.

Table 9-14: SAP log level

Severity	Configuration Values
INFO	`log_info` or `log_information`
WARNING	`log_warn` or `log_warning`
ERROR	`log_error`
FATAL	`log_fatal`
NONE	`log_none`

All messages of the set severity level and higher appear in the log file. For example, if you set log severity to WARNING, all messages of severity WARNING, ERROR and FATAL are logged. The default log severity is ERROR.

Table 9-15: SAP trace level

Severity	Configuration Values
DEBUG	`trace_debug`
PATH	`trace_path`
INFO	`trace_info` or `trace_information`
ERROR	`trace_error`

Severity	Configuration Values
NONE	`trace_none`

All messages of the set severity level and higher appear in the log file. For example, if you set trace severity to INFO, all messages of severity INFO and ERROR are logged. The default trace severity is ERROR.

Configuration options

You can also add the following options to the file:

Option	Values	Description
`always_close`	`true` or `false`	Closes the log file after each write. Default value is `false`.
`append`	`true` or `false`	Adds the log information to the existing log files. Changes the naming pattern to exclude the PID and timestamp, and instead uses a rolling log file number. Default value is `false`.
`keep_num`	an integer	Specifies the number of log files to keep. Default value is `0`. A negative integer means all log files are kept.
`log_dir`	a string	Specifies the log file directory. Default value is the value of the `BO_TRACE_LOGDIR` environment variable. A hyphen (-) means the log information is directed to `stdout`.
`log_level`	`none`, `low`, `medium` or `high`	Defines a shortcut to specify both `sap_log_level` and `sap_trace_level` values. See the table below.
`scope_only`	`true` or `false`	Specifies if only messages associated with scopes appear in log files. The default value is `false`. **Note:** A scope marks the entry and exit out of a code block.

Option	Values	Description
size	a number in KB	Maximal size of a log file. Default value is 10000.

log_level option values

The following table provides the sap_log_level and sap_trace_level values set when the corresponding log_level value is specified in the configuration file.

log_level Value	sap_log_level Value	sap_trace_level Value
none	log_error	trace_error
low	log_error	trace_info
medium	log_warn	trace_path
high	log_info	trace_debug

Parsing errors

Errors and warnings that may occur when parsing the _trace.ini file are written to an ERR file in the same directory as the INI file. The error file name is made of the INI file name.

If the same INI file is shared by several processes, then you do not know which process generates the error. To generate more informative error file names, add the following lines at the top of the _trace.ini file:

```
error_file = config_file + "_" + process + ".err";
output_file = config_file + "_" + process + ".out";
```

9.9.2 Activating Logs and Traces in the cs.cfg File

You can find Connection Server logs and traces with those of SAP BusinessObjects applications, such as SAP BusinessObjects Web Intelligence. You can also log and trace Connection Server separately by configuring the cs.cfg file.

The Traces section of the file helps you activate traces for Connection Server jobs and drivers. You activate job traces by setting the Active attribute of Job and JobLevel to Yes. You activate driver traces by setting the Active attribute of the Middleware to Yes. You can activate traces for a specific driver by setting the Active attribute of the driver to Yes.

Example: Default values

```
<Traces Active="Yes">
 <Logger Implementation="C" ClassID="csTRACELOG">
 <Logger Implementation="Java" ClassID="com.sap.connectivity.cs.logging.TraceLogLogger"/>
 <Trace Name="JobLevel" Active="Yes">
   <Trace Name="Job" Active="Yes"/>
 </Trace>
 <Trace Name="MiddleWare" Active="Yes">
   <Trace Name="Oracle" Active="Yes"/>
   <Trace Name="Sybase" Active="Yes"/>
   <Trace Name="DB2" Active="Yes"/>
   ...
   <Trace Name="JCO" Active="Yes"/>
   <Trace Name="BO OC" Active="Yes"/>
 </Trace>
</Traces>
```

This configuration allows by default the logging and tracing of Connection Server and drivers.

9.9.3 To activate Logs and Traces in Library Mode

1. Stop the Connection Server.
2. Create and set values to the following environment variables:
   - `BO_TRACE_CONFIGDIR`, to set the folder name of configuration files for logs, for example: `C:\BOTraces\config`
   - `BO_TRACE_CONFIGFILE`, to set the name of the configuration file, for example `BO_trace.ini`
   - `BO_TRACE_LOGDIR`, to set the folder name for logs, for example: `C:\BOTraces`

   Note:
   These environment variables are commonly used to configure tracing of all BI platform activity.

3. Create the `BO_trace.ini` configuration file as follows:

   ```
   sap_log_level = log_info;
   sap_trace_level = trace_debug;
   ```

   Note:
   The `INI` file name is case-sensitive.

4. Add the `logging.jar` and `tracelog.jar` directory paths to the `CLASSPATH` environment variable. For example: `C:\BOTraces\lib\logging.jar` and `C:\BOTraces\lib\tracelog.jar`.
5. Restart the Connection Server.

The logs and traces are written into a file whose default name is `TraceLog_<pid>_<timestamp>_trace.log`. If you want to set another name to the log file, update the `BO_TRACE_PROCESS` environment variable with the new name.

9.9.4 To activate Logs and Traces in Server Mode

You enable traces for jobs and drivers from the CMC. This allows you to let the server run while enabling traces.

1. Start the CMC.
2. Go to the **Properties** page of either ConnectionServer server (for Native Connectivity Service) or Adaptive Processing Server (for Adaptive Connectivity Service).
3. In the "Low Level Tracing" section, select:
   - **Enable Job Tracing** to enable job traces
   - **Enable Middleware Tracing** to enable middleware traces

   Note:
   The **TraceLog** level must be set to `High` in the "TraceLog Service" section. This level is different from the SAP log level defined through the `BO_Trace.ini` file.

   Remember:
   The **Enable Middleware Tracing** setting allows you to trace all middleware. If you want to trace only specific middleware, you then need to configure the `cs.cfg` file and restart the server.

9.9.5 Reading Logs and Traces

Connection Server traces and logs are tagged with `|CS|` in trace files. Connection Server logs are also identified by the `THIS IS A LOG` mention. Each function has its own trace. Traces contain the following information:

- `ENTER` traces, which identify the middleware API call. `EXIT` traces identify the returned call. The middleware API does not return any call if there is no `EXIT` after `ENTER`.
- Call exceptions, if `|E|` is displayed
- API errors, if the return code `-1` is displayed
- Error or log severity (For example, `Debug`, `Path`, or `Information`)
- Traced network layer and database in the `[Network Layer|Database|ID]` section of the trace
- The log and error messages

Example: Trace

The following is an excerpt of a trace with DEBUG and PATH severities obtained when the Connection Server was used in library mode:

```
|BE42CC9D40484416BADEF01039652F620|2012 04 12 11:41:27.422|+0200|Debug| |<<| | |TraceLog| 5256|   1|main
           | |||||||||||||||||||||CS||[unknown|unknown|ID:0]CS Internal Configuration:
|BE42CC9D40484416BADEF01039652F621|2012 04 12 11:41:27.534|+0200|Debug| |<<| | |TraceLog| 5256|   1|main
           | |||||||||||||||||||||CS||[unknown|unknown|ID:0]
```

```
...
|BE42CC9D40484416BADEF01039652F628|2012 04 12 11:41:27.920|+0200|Path| |  | | |TraceLog| 5256|   1|main
          | ||||||||||||||||||||CS||ENTER [JDBC|Oracle 11|ID:0]com.sap.connectivity.cs.java.ConnectionMan
ager{com.businessobjects.connectionserver.ConnectionManager}.close
|BE42CC9D40484416BADEF01039652F629|2012 04 12 11:41:27.920|+0200|Path| |  | | |TraceLog| 5256|   1|main
         | ||||||||||||||||||||CS||EXIT [JDBC|Oracle 11|ID:0]com.sap.connectivity.cs.java.ConnectionManag
er{com.businessobjects.connectionserver.ConnectionManager}.close
```

Example: Trace

The following is an excerpt of a trace with INFO severity obtained when the Connection Server was used in server mode:

```
|6E606102224D49849A2C17D8691FAC785011|2012 04 26 19:02:43.075|+0200|Information| |==| | |aps_MySIA.csjava|
 7088|  53|service builder-4| ||||||||||||||||||||CS||[unknown|unknown|ID:0]Starting CORBA NetworkLayer
service...
```

Example: Log

The following is an excerpt of a log obtained when the Connection Server was used in server mode:

```
|6E606102224D49849A2C17D8691FAC7B5114|2012 04 26 19:02:43.805|+0200| |Information|  | | |aps_MySIA.csjava|
 7088|  61|service builder-1| ||||||||||||||||||||CS||THIS IS A LOG [unknown|unknown|ID:0]Loaded driver:
NetworkLayer=JDBC, Database=Generic JDBC datasource

|6E606102224D49849A2C17D8691FAC7B5116|2012 04 26 19:02:43.817|+0200| |Information|  | | |aps_MySIA.csjava|
 7088|  61|service builder-1| ||||||||||||||||||||CS||THIS IS A LOG [unknown|unknown|ID:0]Loaded driver:
NetworkLayer=JDBC, Database=MaxDB 7.7

...
|43eabdad-d3e4-ec14-89aa-0c9a9fba3101|2012 04 26 19:02:20:883|+0200| |Information|  | | |connectionserv
er_MySIA.ConnectionServer|1576|7816|| |0|94|0|2|-|-|-|-|-|-|||||||||||CS||THIS IS A LOG [unknown|un
known|ID:0]ConnectionServer is now started
```

9.10 Activating Logs and Traces for the OLAP Client

You can find OLAP Client logs and traces with those of SAP BusinessObjects applications, such as SAP BusinessObjects Web Intelligence. For example, you can find them with traces of the Web Intelligence Processing Server. You can also log and trace OLAP Client separately by configuring the `OlapClient.cfg` file.

You activate logs by setting the `UseLog` attribute to `yes` in the `OlapClient` section of the file. You specify the log file path in the `LogFileName` attribute. You must not change the `UseProcessName` value.

You do not need to stop the Web Intelligence Processing Server to change these settings. You can activate the logs and traces while it is running.

Example:

```
[OlapClient]
UseLog =yes
UseProcessName=false
```

```
LogFileName =c:\OlapClient.log
; Factory Mode can be: Lib (inproc) or Proxy (Remoting)
FactoryMode=Lib
...
```

This configuration allows the logging and tracing for the OLAP Client.

Related Topics

- About the OlapClient.cfg Configuration File

9.10.1 Example of Log

The following example is an excerpt of a log file for the OLAP Client obtained for an MSAS connection.

```
...
[2012/11/05 - 12:04:54:025][0] XmlaConnectionWinhttp::XmlaConnectionWinhttp() -> Using Windows HTTP Services
 version 6.1.7600.16385
[2012/11/05 - 12:04:54:025][0] HTTPClient::connect() -> Using direct access
[2012/11/05 - 12:04:54:040][0] HTTPClient::connect() -> Using Windows Proxy Automatic Discovery (WPAD)
[2012/11/05 - 12:04:54:040][0] HTTPClient::connect() -> Using proxy settings: WPAD url=http://proxy:8083,
Proxy= ProxyBypass=
[2012/11/05 - 12:04:54:040][0] HTTPClient::connect() -> Connecting using WinHttpConnect(): Hostname='olap-
wxp' Port=80
[2012/11/05 - 12:04:54:040][0] XmlaConnectionWinhttp::Open() -> Connected.
[2012/11/05 - 12:04:54:040][0] XmlaClient::discover() Call 'DISCOVER_DATASOURCES'...
[2012/11/05 - 12:04:54:040][0] SoapEnvelope::updateProperties() BeginRange=-1 EndRange=-1
[2012/11/05 - 12:04:54:040][0] XmlaConnectionWinhttp::SendRequest() Authentication Mode = Credentials.
[2012/11/05 - 12:04:54:040][0] XmlaConnectionWinhttp::SendRequest() Setting basic credential to HttpRequest.
[2012/11/05 - 12:04:54:040][0] XmlaConnection::getTimeout() No timeout was specified in ConnectionSpecifica
tion, defaulting to 100(s).
[2012/11/05 - 12:04:54:056][0] XmlTextReader::XmlTextReader() Using libXML version 2.06.32
[2012/11/05 - 12:04:54:056][0] XmlaClient::discover() Rowset contains 1 rows
[2012/11/05 - 12:04:54:056][0] XmlaClient::discover() Call 'DISCOVER_DATASOURCES' ElapseTime=15ms
[2012/11/05 - 12:04:54:056][0] XmlaClient::createOlapEntities() -> Rowset=[DISCOVER_DATASOURCES], 1 entities
 retrieved Elapse=15ms
[2012/11/05 - 12:04:54:056][0] XmlaClient::discover() Call 'DISCOVER_SCHEMA_ROWSETS'...
[2012/11/05 - 12:04:54:072][0] SoapEnvelope::updateProperties() BeginRange=-1 EndRange=-1
[2012/11/05 - 12:04:54:072][0] XmlaConnectionWinhttp::SendRequest() Authentication Mode = Credentials.
[2012/11/05 - 12:04:54:072][0] XmlaConnectionWinhttp::SendRequest() Setting basic credential to HttpRequest.
[2012/11/05 - 12:04:54:072][0] XmlaConnection::getTimeout() No timeout was specified in ConnectionSpecifica
tion, defaulting to 100(s).
[2012/11/05 - 12:04:54:072][0] XmlTextReader::XmlTextReader() Using libXML version 2.06.32
[2012/11/05 - 12:04:54:072][0] XmlaClient::discover() Rowset contains 54 rows
...
```

Configuring Data Access Driver Parameters

10.1 Configuring Driver Parameters

To configure data access for a particular data access driver, you can either edit the driver configuration files to adjust the parameter settings, or create your own database entries if you need connections for specific databases of your environment.

Note:
For each SAP BusinessObjects application that uses Connection Server, the associated Readme file contains information on command line utilities that you can use to check your RDBMS and data access driver configuration. These utilities can create log files that trace interactive analysis server activity. Refer to the readme file for the release for instructions on the use of these utilities.

Related Topics

- About Global Parameters

10.1.1 Data Access Configuration Files

The following configuration files control data access driver configurations for each defined connection:

- `cs.cfg`

  This file defines global parameters that apply to all connections. It is located in the *connection server-install-dir*\connectionServer directory.

- *<driver>*.sbo

  This file is specific to each data access driver. The *<driver>* placeholder stands for the data source to which the configuration file applies. Each SBO file is in a subdirectory of the connectionServer directory, where the subdirectory is named after the database network layer or middleware, for example

  connectionserver-install-dir\connectionServer\oracle for Oracle databases.

Note:
The parameters set in the `DriverDefaults` section of `cs.cfg` are overridden by corresponding settings in the SBO files.

- *`<driver>`*`.setup`

  This file defines the SBO file name, directory and the database network layer or middleware that relates to the driver. This file is required to make the driver usable. A driver without any setup file cannot be used. All the files are in the

  `connectionserver-install-dir``\connectionServer\setup` directory.

  As an example, the following `oracle_jdbc.setup` file defines the `oracle.sbo` configuration file for Oracle data access drivers that are used to establish JDBC connections:

```
...
<Driver>
 <NetworkLayer Name="JDBC"></NetworkLayer>
 <Directory>jdbc</Directory>
 <DataFileName>oracle</DataFileName>
</Driver>
...
```

Related Topics

- About Global Parameters
- Installed SBO Files

10.1.2 Installed SBO Files

The following *`<driver>`*`.sbo` files are installed by default on Microsoft Windows.

For an up-to-date list of supported drivers, check the SAP Service Marketplace website at service.sap.com/bosap-support, or contact your SAP representative.

Subdirectory	Database Technology	SBO File
`db2`	IBM DB2	`db2.sbo`
`essbase`	Oracle Essbase	`essbase.sbo`
`javabean`	Javabean	`javabean.sbo`
`jco`	SAP ERP	`jco.sbo`

Subdirectory	Database Technology	SBO File
jdbc	Data Federator Server	datafederator.sbo
	IBM DB2	db2.sbo
	Derby	derby.sbo
	Greenplum and PostgreSQL	greenplum.sbo
	HIVE	hive.sbo
	HSQLDB	hsqldb.sbo
	IBM Informix	informix.sbo
	Ingres	ingres.sbo
	Generic JDBC	jdbc.sbo
	MaxDB	maxdb.sbo
	MySQL	mysql.sbo
	HP Neoview	neoview.sbo
	Netezza	netezza.sbo
	SAP HANA	newdb.sbo
	Oracle	oracle.sbo
	Microsoft SQL Server	sqlsrv.sbo
	Sybase	sybase.sbo

Subdirectory	Database Technology	SBO File
	Teradata	`teradata.sbo`
	HP Vertica 6.1	`vertica.sbo`
`odata`	OData 2.0	`odata.sbo`

Subdirectory	Database Technology	SBO File
odbc	MS Access 2007	access.sbo
	MS Access 2010 and 2013	access2010.sbo
	Data Federator Server	datafederator.sbo
	IBM DB2 iSeries	db2iseries.sbo
	PostgreSQL 8	greenplum.sbo
	Greenplum 4 and PostgreSQL 9	greenplum4.sbo
	IBM Informix	informix.sbo
	Ingres	ingres.sbo
	MaxDB	maxdb.sbo
	MS Excel 2007	msexcel.sbo
	MySQL	mysql.sbo
	HP Neoview	neoview.sbo
	Netezza	netezza.sbo
	SAP HANA	newdb.sbo
	Generic ODBC and Generic ODBC3	odbc.sbo
	OpenAccess for Salesforce	openaccess.sbo
	MS Excel 2010, 2013 and text files	personalfiles.sbo

Subdirectory	Database Technology	SBO File
	Microsoft SQL Server	`sqlsrv.sbo`
	Sybase	`sybase.sbo`
	Teradata	`teradata.sbo`
	HP Vertica 6.1	`vertica.sbo`
`oledb`	Generic OLE DB	`oledb.sbo`
	Microsoft SQL Server	`sqlsrv.sbo`
`oledb_olap`	Microsoft Analysis Services	`slqsrv_as.sbo`
`open`	CSV files	`open.sbo`
`oracle`	Oracle	`oracle.sbo`
	Oracle EBS	`oracle_ebs.sbo`
`sap`	SAP BW	`sap.sbo`
`sybase`	Sybase	`sybase.sbo`
`xml`	XML	`xml.sbo`
	Web services	`webservices.sbo`

`jdbc` subdirectory contains configuration files for managing connections to databases through JDBC network layer. Check the SAP Service Marketplace website or the series of SBO files for details of the databases supported for JDBC connections.

`odbc` subdirectory contains the `mysql.sbo` configuration file for managing connections to MySQL database through ODBC network layer. MySQL 5 database is available on all platforms, with Unicode

support. Make sure you use the MySQL Connector/ODBC 5.1.4 or higher to connect to this database. If you use an older version of the driver on UNIX, you will encounter errors at runtime.

`odbc` subdirectory contains the `greenplum4.sbo` and `access2010.sbo` configuration files for managing connections to Greenplum 4 and PostgreSQL 9 on one hand, and MS Access 2010 and 2013 on the other hand through ODBC network layer. These files are different from the `greenplum.sbo` and `access.sbo` files, because they deploy the configuration on both 32-bit and 64-bit MS Windows. Greenplum 4 and PostgreSQL 9 databases are also available on 64-bit UNIX flavors.

`odbc` subdirectory also contains the `msexcel.sbo` configuration file for managing connections to MS Excel 2007 through ODBC network layer. This file deploys the configuration on 32-bit MS Windows only. The `personalfiles.sbo` configuration file deploys configuration for MS Excel 2010 and 2013 through ODBC on both 32-bit and 64-bit MS Windows.

`open` subdirectory contains the `open.sbo` configuration file for managing connections to CSV files through the CSV OpenDriver. When developing a CSV driver based on the CSV Open driver sample using the Java Driver Development Kit, you must locate all your configuration files in this directory. For more information about this driver, refer to the *Data Access Driver Java SDK Developer Guide*.

10.1.3 To View and Edit SBO Files

Caution:
Before opening an SBO file, make a backup copy of the file. Some configuration parameters must not be edited. If you change or delete them it could affect the operation of your SAP BusinessObjects applications.

1. Browse to the directory that stores the SBO file for your target data access driver.
2. Open the SBO file in an XML editor.
3. Expand sections as required.
4. Locate the appropriate tag for the value to change, and change the value.

   Parameters appear in the format: `<Parameter Name="parameter">value</Parameter>` where *`parameter`* is the name of the parameter, and *`value`* is the value attributed to the parameter.
5. Check that the file is valid against the DTD, save and close the file.

10.1.4 To Customize SBO Files

Caution:
When installing a new driver, you may want to customize SBO files with your own database entries and drivers. In order to avoid possible errors when editing SBO files, SAP BusinessObjects recommends you to create a separate SBO file that specifies the databases targeted by your customization and the

libraries used by the driver. Beforehand you also need to create a setup file for the SBO file definition. You do not need to modify registries.

1. Create a setup file in the *connectionserver-install-dir*\connectionServer\setup directory by using an XML editor.

   You specify the SBO file name, its directory and the database network layer that is used for the connection.
2. Check that the file is valid against the DTD, save and close the file.
3. Either browse to the subdirectory where you want to store the SBO file or create your own subdirectory as specified in the setup file.
4. Create the SBO file with your target database and driver libraries by using an XML editor.
5. Check that the file is valid against the DTD, save and close the file.

Note:
Apart from customizing the SBO file, you also need to create the COD, PRM and RSS files for each driver newly installed.

10.1.5 To Check Connections Dynamically

You can customize your driver to make it validate at runtime that a connection of the connection pool can be used. This validation process consists of executing an SQL query without side effect when the connection is retrieved from the pool. It means that the connection can be used if the SQL query is executed without error. If it is not, the connection is discarded.

Remember:
This functionality is available for Generic ODBC, Generic OLE DB and Generic JDBC connections. All other supported connectivities handle this connection check already, then no customization is required.

1. Stop Connection Server.
2. Open the SBO file of your driver.
3. Locate the `DataBase` section that is appropriate to your connection.

   For example, for a Generic ODBC connection:

```
<DataBases>
 <DataBase Active="Yes" Name="Generic ODBC datasource">
  <Libraries>
   ...
  </Libraries>
  <Parameter Name="CharSet Table" Platform="Unix">datadirect</Parameter>
 </DataBase>
 ...
```

4. Add the following row in the `DataBase` section:

```
<Parameter Name="Connection Check">SQL query</Parameter>
```

   where *SQL query* is the query you use to check the validity.
5. Save the SBO file.
6. Restart Connection Server.

Connection Server validates that the connection can be used before executing the data request.

10.1.6 JDBC Driver Properties

You can add JDBC driver properties by doing one of the following:

- In the connection wizard of your application, at connection creation, set the "JDBC Driver Properties (key=value,key=value):" parameter. It is optional.
- Set driver properties in the appropriate SBO file after you stop Connection Server.

If the property is both set in the SBO file and through the wizard, only the value set through the wizard is taken into account by the application.

Example:

This shows the section of an `oracle.sbo` file modified with two driver properties.

```
<DataBase Active="Yes" Name="Oracle 10">
 <Class JARFile="dbd_jdbc,dbd_oracle">
com.businessobjects.connectionserver.java.drivers.jdbc.oracle.OracleDriver</Class>
 <JDBCDriver>
  <Parameter Name="JDBC Class">oracle.jdbc.OracleDriver</Parameter>
  <Parameter Name="URL Format">jdbc:oracle:thin:@$DATASOURCE$:$DATABASE$</Parameter>
  <Properties>
   <Property Name="oracle.jdbc.defaultNChar">true</Property>
   <Property Name="defaultNChar">true</Property>
  </Properties>
  ...
 </JDBCDriver>
 ...
</DataBase>
```

Related Topics

- To Create a JDBC Connection with the SBO File

10.2 About DataDirect ODBC Drivers

The Data Access layer allows the use of DataDirect ODBC 7.0 drivers for MS SQL Server databases on all UNIX platforms. These drivers can be either non-branded or branded drivers.

DataDirect branded drivers are provided as part of the BI platform and can only be used with SAP BusinessObjects applications such as SAP BusinessObjects Web Intelligence. You can find them in *boe-install-dir*/enterprise_xi40/*platform-name*/odbc/lib directory, where *boe-install-dir* stands for the BI platform installation directory and *platform-name* for the UNIX platform name.

The MS SQL Server databases can work with either a DataDirect ODBC 7.0 driver or a DataDirect ODBC 7.0 branded driver. They can also work with the 6.0 SP2 version of the driver. However, this version does not come with the present platform release.

The Connection Server default settings allow the non-branded ODBC driver to work seamlessly with your current configuration settings. If you have already deployed a driver in your environment, you will be able to install the non-branded driver without any configuration change.

10.2.1 To Enable the Use of DataDirect Branded Drivers

To use the branded driver, you must make sure the data access is configured correctly.

1. Navigate to the directory that contains the `sqlsrv.sbo` file.

   On UNIX, this configuration file is located in the *connectionserver-install-dir*/connectionServer/odbc directory.

2. Use an XML editor to open the `sqlsrv.sbo` file for editing.
3. Locate the `Defaults` section.

   The `Use DataDirect OEM Driver` parameter is set to `No` by default. This means the data access is configured by default to work for non-branded drivers.

4. Set the `Use DataDirect OEM Driver` parameter to `Yes` and save the file.
5. Add the following path to `LD_LIBRARY_PATH` environment variable:

   boe-install-dir/enterprise_xi40/*platform-name*/odbc/lib

6. Configure the environment by editing the `env.sh` file in the *boe-install-dir*/setup directory and source it.

   For example:

```
DEFAULT_ODBCFILE="${BOBJEDIR}enterprise_xi40/"odbc.ini
Export DEFAULT_ODBCFILE
ODBC_HOME="${BINDIR}odbc"
Export ODBC_HOME
ODBCINI="${BOBJEDIR}enterprise_xi40/"odbc.ini
export ODBCINI
```

   Note:

   `DEFAULT_ODBCFILE` can point to any file which holds the connection details for the branded drivers.

7. Configure the data source by editing the `odbc.ini` file.

   For example:

```
[ODBC Data Sources]
sql2008=sql=DataDirect 7.0 SQL Server Native Wire Protocol
Driver=/.../enterprise_xi40/linux_x64/odbc/lib/CRsqls24.so
Description=DataDirect 7.0 SQLServer Wire Protocol Driver
Address=10.180.0.197,1433
Database=bodb01
```

8. Start Connection Server from the CMC.

Connection Server is able to create a connection to MS SQL Server databases with DataDirect ODBC branded drivers.

Example:

The following is an excerpt of the default `sqlsrv.sbo` file.

```
<Defaults>
 <Parameter Name="Family">Microsoft</Parameter>
 <Parameter Name="SQL External File">sqlsrv</Parameter>
 <Parameter Name="SQL Parameter File">sqlsrv</Parameter>
 <Parameter Name="Description File">sqlsrv</Parameter>
 <Parameter Name="Strategies File">sqlsrv</Parameter>
 ...
 <Parameter Name="Use DataDirect OEM Driver" Platform="Unix">No</Parameter>
</Defaults>
...
```

SBO Parameter Reference

11.1 SBO File Structure

There is a `<driver>.sbo` file for each supported data access driver. Each `<driver>.sbo` file is divided into the following sections:

File Section	Description
`Defaults`	This section contains the default configuration parameters that apply to all database middleware that uses the data access driver. These parameters override any corresponding values set in the database middleware.

File Section	Description
`Databases`	This section contains a `DataBase` child element for each database middleware that is supported by the data access driver. Each `DataBase` element can contain the following elements or attributes: • `Active`: this attribute specifies if middleware support is activated or not. Its value is `YES` or `NO`. • `Name`: this attribute specifies the name of the middleware supported by the data access driver. The middleware name values set here appear in the Database Middleware page of the connection wizard. • `Alias`: this element specifies the name of the older middleware versions no longer officially supported by the data access driver, but that are still in use. You can add an alias element for an older middleware version so that existing connections use the current data access driver instead. You can set configuration parameters specific to the old middleware as parameters of the new alias. You can create new connections using the alias. • `Parameter`: this element has a `Name` attribute and a value that applies specifically to a middleware. Values set for parameters listed here override the values set for the same parameters in the `Defaults` section. **Note:** Boolean parameters accept both `true`/`false` and `yes`/`no` as values. Values do not depend on case.

11.2 SBO Parameter Description

The configuration parameters are listed according to the following:

- Common

  This section describes SBO file parameters shared by different database technologies. The other categories describe parameters that are specific to the database technology or network layer they represent.

- JavaBean
- JCO
- JDBC

- OData
- ODBC
- OLE DB
- OLE DB for OLAP
- Sybase ASE/CTL
- Teradata

Each parameter is shown with the following information:

- Example of how the parameter appears in the XML file
- Description of the parameter
- Possible values that can be set for the parameter
- Default value of the parameter

Related Topics

- Common SBO Parameters
- JavaBean SBO Parameters
- JCO SBO Parameters
- JDBC SBO Parameters
- OData SBO Parameters
- ODBC SBO Parameters
- OLE DB SBO Parameters
- OLE DB OLAP SBO Parameters
- Sybase SBO Parameters
- Teradata SBO Parameters

11.3 Common SBO Parameters

These SBO parameters are mostly common to all data access drivers. They are defined in the `Defaults` section of the files. Some of these SBO parameters are also defined in the `cs.cfg` file. Values set in the `Defaults` section override values set in the `cs.cfg` file.

11.3.1 Array Bind Available

```
<Parameter Name="Array Bind Available">No</Parameter>
```

Description	Specifies whether the database supports array binding. Array binding helps you optimize performance of SQL update queries.
Values	`Yes`: the database supports array binding. `No`: the database does not support array binding.
Default	The value set in the `cs.cfg` file.

11.3.2 Array Bind Size

Restriction:
SAP BusinessObjects BI platform Client Tools do not use this parameter.

```
<Parameter Name="Array Bind Size">5</Parameter>
```

Description	Specifies the number of rows exported with each `INSERT` query.
Values	The number of rows that are exported with each `INSERT` (integer).
Default	The value set in the `cs.cfg` file.

11.3.3 Array Fetch Available

```
<Parameter Name="Array Fetch Available">No</Parameter>
```

Description	Specifies whether Connection Server enables array fetching. Array fetching helps you optimize performance by retrieving SQL results per slice.

Values	`Yes`: array fetching is supported. `No`: array fetching is not supported.
Default	The value set in the `cs.cfg` file.

11.3.4 Array Fetch Size

```
<Parameter Name="Array Fetch Size">10</Parameter>
```

Description	Specifies the number of rows of data retrieved per slice. Connection Server provides array fetch capability in any deployment mode. **Remember:** The `Array Fetch Size` value is propagated to the database middleware if it supports array fetch. The optimal number depends on the performance of your system: • If the number is low, the system retrieves small amounts of data many times. This can affect performance. • If the number is high, the system performs fewer retrieval operations, but it requires more memory for each one. **Caution:** Make sure the `Array Fetch Size` value is appropriate because it can affect the performance of your system, especially in remote access, for example when connections to SAP ERP system are established in a web tier deployment mode. For OLAP connections in remote access (MS Analysis Services, SAP BW, and Essbase data sources through 32-bit Connection Server), set the array fetch size to an optimal value depending on the number of columns in reports to be created (for example `100` if the number of columns is high, and `250` if the number is low). In a web tier deployment mode, the **HTTP Chunk Size** parameter can also help you to fine-tune the performance by reducing the number of data calls between the client and the server. See the *SAP BusinessObjects Business Intelligence platform Administrator Guide* for more information about HTTP chunking.

Values	The number of rows retrieved per slice (integer). `1` specifies that array fetching is disabled. `0` specifies that the driver decides which array fetch size to use. This value is only valid for JDBC drivers.
Default	The value set in the `cs.cfg` file.

Related Topics

- SAP ERP Driver Restrictions

11.3.5 BigDecimal Max Display Size

```
<Parameter Name="BigDecimal Max Display Size">128</Parameter>
```

Description	Specifies the maximum display size of data retrieved with BigDecimal character type.
Values	The display size (integer in bytes).
Default	No default value.

11.3.6 Binary Max Length

```
<Parameter Name="Binary Max Length">32768</Parameter>
```

Description	Specifies the maximum length of table columns whose type is: • binary and `MaxLength` value is `Max` or no fixed length is provided for OData data sources • base64Binary and hexBinary for XML data sources Data returned is truncated if larger than specified.

Default	`32768`

Related Topics

- OData Driver
- XML Driver
- Web Service Driver

11.3.7 Bucket Split Size

```
<Parameter Name="Bucket Split Size">25000</Parameter>
```

Description	Specifies the number of records which are sorted in memory before they are written to the local disk. The following data access drivers use `Bucket Split Size` when they perform the ORDER BY, GROUP BY or DISTINCT operations: • CSV OpenDriver • OData driver • SAP ERP driver • XML and Web Service drivers **Note:** You can configure the directory path in the `cs.cfg` file using the `Temp Data Dir` parameter when the operating system does not specify a temporary folder. This can happen with Linux operating systems. `Bucket Split Size` affects the memory consumption. If the main memory size is too small, the parameter value is ignored.
Default	`25000`

Related Topics

- Local Disk Used as a Cache for Sorting Operations

  The CSV OpenDriver and the OData, SAP ERP, XML and Web Service drivers can use the local disk as a cache for sorted rows.

- Temp Data Dir

11.3.8 Catalog Separator

```
<Parameter Name="Catalog Separator">.</Parameter>
```

Description	Specifies the separator character that is used between elements of database identifiers (qualifiers, owners, tables and columns). For example, *database_name.table_name.column_name*.
Values	The separator character to use. Usually a period.
Default	If not specified, Connection Server uses the separator specified in the database middleware.

11.3.9 CharSet Table

```
<Parameter Name="CharSet Table">sybase</Parameter>
```

Description	Specifies the name of the table used for character set mapping between the operating system and the middleware.
Values	The name of the CRS file.
Default	No default.

11.3.10 Description File

```
<Parameter Name="Description File">oracle</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies the name of the COD file, which holds the connection wizard input field labels.

11.3.11 Dictionary Transaction Mode

```
<Parameter Name="Dictionary Transaction Mode">Transactional</Parameter>
```

Description	Specifies that the data access driver queries metadata in transaction mode. **Note:** The data access drivers query metadata in AutoCommit mode by default. To change the configuration of a driver to transactional, add the parameter to the appropriate section of the SBO configuration file (either `Defaults` or any `Database` section).
Values	`Transactional`

11.3.12 Driver Capabilities

```
<Parameter Name="Driver Capabilities">Procedures,Query</Parameter>
```

Description	The capabilities of the driver, that is whether it can access stored procedures and queries available in the database software. This parameter is set using the connection wizard. You can include both values in the parameter. **Note:** This parameter must be set to `Procedures` for a JavaBean driver. The functionality of a JavaBean driver is defined as stored procedures as far as SAP BusinessObjects applications are concerned.

Values	`Procedures`: the driver can use procedures stored in the database to retrieve data. `Query`: the driver can use a query language such as SQL to retrieve data.
Default	`Query`

11.3.13 Driver Name

```
<Parameter Name="Driver Name">Adaptive Server IQ</Parameter>
```

Description	Specifies the name of the driver that displays in **Drivers** tab of ODBC Data Source Administrator on Microsoft Windows. This parameter is ODBC-specific. It helps you filter drivers in the ODBC Data Source Name (DSN) list.
Values	The name of the driver. **Note:** You can use a regular expression based on the GNU regexp syntax from PERL.
Default	No default value.

11.3.14 Escape Character

```
<Parameter Name="Escape Character">/</Parameter>
```

Description	Specifies the character to use to escape strings of special characters, for example patterns.

Values	The character to use as the escape character.
Default	If not specified, Connection Server retrieves the value from the middleware.

11.3.15 Extensions

```
<Parameter Name="Extensions">oracle10,oracle,jdbc</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies the list of possible names for PRM and RSS files of data access drivers. This list also specifies the possible names for directories where you can store JAR files.

Related Topics

- To Create a JDBC Connection with Extensions

11.3.16 Family

```
<Parameter Name="Family">Sybase</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies the family of the database engine that is displayed in the "Database Middleware Selection" page of the connection wizard. The set of middleware that corresponds to your license is displayed on this page in a tree view.

11.3.17 Force Execute

```
<Parameter Name="Force Execute">Never</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies whether the SQL query is executed before retrieving result descriptions.
Values	`Never`: SQL query is never executed before retrieving result descriptions. `Procedures`: SQL query is executed before retrieving result descriptions, but only for stored procedures. `Always`: SQL query is always executed before retrieving result descriptions.
Default	`Never`

11.3.18 Identifier Case

```
<Parameter Name="Identifier Case">LowerCase</Parameter>
```

Description	Specifies how the database handles the case behavior of simple identifiers.

Values	`CaseSensitive`: The database treats mixed case SQL identifiers as case sensitive. `LowerCase`: The database treats mixed case SQL identifiers as case insensitive and stores them in lower case. `MixedCase`: The database treats mixed case SQL identifiers as case insensitive and stores them in mixed case. `UpperCase`: The database treats mixed case SQL identifiers as case insensitive and stores them in upper case.

11.3.19 Identifier Quote String

```
<Parameter Name="Identifier Quote String">&quot;</Parameter>
```

Description	Specifies the character used to quote database identifiers.
Values	The character used to quote database identifiers. Usually quotes (`"`).
Default	If not specified, Connection Server retrieves the information from the database middleware.

11.3.20 Include Synonyms

```
<Parameter Name="Include Synonyms">False</Parameter>
```

Description	Specifies if columns for Oracle Synonyms are retrieved from the Oracle database. This is valid for Oracle connections through JDBC or Oracle OCI network layers.
Values	`True`: columns for Oracle Synonyms are retrieved and display as table columns. `False`: columns for Oracle Synonyms are not retrieved.

Default	`False`

11.3.21 Integer Max Length

```
<Parameter Name="Integer Max Length">18</Parameter>
```

Description	Specifies the maximum length of table columns whose XML built-in type is integer, nonPositiveInteger, negativeInteger, noNegativeInteger, and positiveInteger. Data returned is truncated if larger than specified. `Integer Max Length` applies to XML data sources and web services.
Default	`18`

Related Topics

- XML Driver - Configuring Column Maximum Size
- Web Service Driver - Configuring Column Maximum Size

11.3.22 Introscope Available

```
<Parameter Name="Introscope Available">True</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies if performance monitoring through CA Wily Introscope is activated for the driver.
Values	`True`: the monitoring of the driver is activated. `False`: the monitoring of the driver is not activated.

Default	The parameter is set to `False` for all the drivers in the `cs.cfg` file. The monitoring is actually activated for SAP HANA connections only in the `newdb.sbo` file.

11.3.23 Max Rows Available

```
<Parameter Name="Max Rows Available">No</Parameter>
```

Description	Specifies if the driver can limit the maximum number of rows that can be retrieved from a data source.
Values	`Yes`: the maximum number of rows can be limited. `No`: the maximum number of rows cannot be limited.
Default	`No`

11.3.24 Native Int64 Available

```
<Parameter Name="Native Int64 Available">False</Parameter>
```

Description	**Note:** Do not edit this parameter. Indicates if 64-bit integers can be handled directly by the middleware.
Values	`True`: 64-bit integers can be handled by the middleware. `False`: the Data Access Layer emulates the Int64 methods.
Default	`False`

11.3.25 Optimize Execute

```
<Parameter Name="Optimize Execute">False</Parameter>
```

Description	Specifies whether Connection Server optimizes the execution of SQL queries. This parameter is supported by Oracle and ODBC drivers only.
Values	`True`: SQL queries are optimized on execution wherever possible. `False`: SQL queries are not optimized for execution.
Default	`False`

11.3.26 Owners Available

```
<Parameter Name="Owners Available">No</Parameter>
```

Description	Specifies whether data access drivers handle database owners. **Note:** To set table owners manually in the information design tool, you must set this parameter to `Yes`.
Values	`Yes`: owners are supported. `No`: owners are not supported.
Default	Not specified. Connection Server retrieves this information from the database middleware.

11.3.27 Qualifiers Available

```
<Parameter Name="Qualifiers Available">No</Parameter>
```

Description	Specifies whether data access drivers handle database qualifiers. **Note:** To set table qualifiers manually in the information design tool, you must set this parameter to `Yes`.
Values	`Yes`: qualifiers are supported. `No`: qualifiers are not supported.
Default	Not specified. Connection Server retrieves this information from the database middleware.

11.3.28 Query TimeOut Available

```
<Parameter Name="Query TimeOut Available">False</Parameter>
```

Description	Specifies whether a query timeout is supported by the database middleware, that is if a query that is running can be cancelled after a time period has expired.
Values	`True`: the database middleware handles query timeouts. `False`: the database middleware does not handle query timeouts.
Default	`False`

11.3.29 Quote Identifiers

```
<Parameter Name="Quote Identifiers">True</Parameter>
```

Description	Specifies whether the identifier of the stored procedure supports quotes.
Values	`True`: quotes are supported. `False`: quotes are not supported.
Default	`True`

11.3.30 Skip SAML SSO

```
<Parameter Name="Skip SAML SSO">False</Parameter>
```

Description	Specifies whether the SAML implementation of single sign-on (SSO) is skipped when connecting to the SAP HANA database. For more information about single sign-on, refer to *SAP BusinessObjects Business Intelligence platform Administrator Guide*.
Values	`True`: SAML is skipped. `False`: SAML is used first.
Default	`False`

11.3.31 SQL External File

```
<Parameter Name="SQL External File">filename</Parameter>
```

Description	**Note:** Do not edit this parameter. The SQL external file holds configuration details used by the data access layer.

11.3.32 SQL Parameter File

```
<Parameter Name="SQL Parameter File">oracle</Parameter>
```

Description	The name of the file that stores database parameters. The extension of this file is `.prm`. You must ensure that this file is located in the same directory as the SBO configuration file.
Values	See the list of values in the SBO file.
Default	The listed values.

11.3.33 SSO Available

```
<Parameter Name="SSO Available">False</Parameter>
```

Description	Specifies whether single sign-on (SSO) is supported. For more information about single sign-on, refer to *SAP BusinessObjects Business Intelligence platform Administrator Guide*.
Values	`True`: single sign-on is supported. `False`: single sign-on is not supported.

Default	`False`

11.3.34 Strategies File

```
<Parameter Name="Strategies File">oracle</Parameter>
```

Description	Specifies the name, with no extension, of the Strategy file (`.stg`). This file contains the external strategies that universe design tool uses for automatic universe creation. Strategy files are stored in the same directory as the SBO file.
Values	`db2` for IBM DB2 data access drivers `informix` for IBM Informix `oracle` for Oracle `sqlsrv` for MS SQL Server `sybase` for Sybase `teradata` for Teradata
Default	The listed values.

11.3.35 String Max Length

```
<Parameter Name="String Max Length">32768</Parameter>
```

Description	Specifies the string maximum length of table columns mapped to ABAP function parameters whose length of value is equal to zero. Also specifies the maximum length of table columns whose type is the following: • String and `MaxLength` value is `Max` for OData data sources • String and any of the following for XML data sources: anyURI, QName, NOTATION, duration, gYearMonth, gYear, gMonthDay, gDay, gMonth, TYPE_NORMALIZEDSTRING, token, language, Name, NCName, XSD_TYPE_ID, IDREF, IDREFS, ENTITY, ENTITIES Data returned is truncated if larger than specified.
Default	`32768`

Related Topics

- OData Driver
- SAP ERP Driver - Access to ABAP Functions
- XML Driver
- Web Service Driver

11.3.36 Temp Data Dir

```
<Parameter Name="Temp Data Dir">C:\temp\</Parameter>
```

Description	Specifies the directory path for the temporary data written on the disk. Use this parameter if the operating system does not specify any default temporary folder, for example: `C:\Users\`*`myName`*`\AppData\Local\Temp`. The following data access drivers can use `Temp Data Dir`: • CSV OpenDriver • OData driver • SAP ERP driver • XML and Web Service drivers When the driver performs an ORDER BY, GROUP BY or DISTINCT operation, the data is written to this folder if the number of records to be sorted exceeds the `Bucket Split Size` parameter value.

Default	The parameter is commented out. Remove the comment syntax to enable the parameter.

Related Topics

- Local Disk Used as a Cache for Sorting Operations

  The CSV OpenDriver and the OData, SAP ERP, XML and Web Service drivers can use the local disk as a cache for sorted rows.

- Bucket Split Size

11.3.37 Transactional Available

```
<Parameter Name="Transactional Available">Yes</Parameter>
```

Description	Specifies if SQL operations run against the database are run as block transactions or individually. This parameter is not listed by default in the SBO file. Add it to the SBO file if your data access driver does not support transactional mode.
Values	`Yes`: operations against the database are run as a block when committed. `No`: each SQL statement is immediately committed, that is, `Autocommit` is deactivated. **Note:** Do not use a driver with `Transactional Available=No` to access the SAP BusinessObjects repository.
Default	`Yes`. This is set in the `cs.cfg` file.

Related Topics

- To Set the Transaction Mode for IBM Informix Connections
- To Make salesforce.com Connections Work in the Universe Design Tool

11.3.38 Type

```
<Parameter Name="Type">Relational</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies the data source type.

11.3.39 Unicode

```
<Parameter Name="Unicode">UTF8</Parameter>
```

Description	Specifies if the access driver can benefit from the Unicode configuration of the client middleware. This parameter appears as a driver default in the `cs.cfg` file. Its value applies to all data access drivers. It is not listed by default in the SBO file. If you want to override the default value, you must add it to the `Defaults` section of the SBO file for the target data access driver.
Values	`UTF8`: 8-bit UCS/Unicode Transformation Format coding. `CharSet`: Character Set coding. `UCS2`: 2-byte Universal Character Set coding
Default	The value set in the `cs.cfg` file.

11.3.40 URL Format

```
<Parameter Name="URL Format ">string</Parameter>
```

Description	Specifies the URL Format.
	The JDBC specification does not specify the format of the connection string that it requires. Vendors use different kinds of URL format, for example:
	• MySQL vendor:
	`jdbc:mysql://$DATASOURCE$/$DATABASE$`
	• Oracle vendor:
	`jdbc:oracle:thin:@$DATASOURCE$:$DATABASE$`
	Note:
	This parameter is supported by JDBC and JavaBean driver files only.

Values	The URL Format.
Default	No default value.

11.3.41 XML Max Size

```
<Parameter Name="XML Max Size">65536</Parameter>
```

Description	Specifies the maximum size of data retrieved in XML format.
Values	The maximum allowed XML size (in bytes).
Default	This varies depending on the database.

11.4 JavaBean SBO Parameters

These parameters apply to the JavaBean SBO file. They are used to configure a JavaBean connection.

These parameters are defined in the `\\connectionserver-install-dir\connectionServer\javabean\javabean.sbo` file.

11.4.1 JavaBean Class

```
<Parameter Name="JavaBean Class">string</Parameter>
```

Description	Defines the entry point of the JavaBean that the SAP BusinessObjects application uses. The entry point is the definition of a java class extending from the `Bean` interface specified through the `com.businessobjects` package.
Values	A fully qualified JavaBean class name.
Default	No default value.

11.5 JCO SBO Parameters

These parameters apply to the SAP ERP database technology. They are used to configure a connection to a SAP ERP system.

These parameters are defined in the `\\connectionserver-install-dir\connectionServer\jco\jco.sbo` file.

11.5.1 ERP Max Rows

```
<Parameter Name="ERP Max Rows">2147483647</Parameter>
```

Description	Specifies the maximum number of rows that can be returned by a query to an InfoSet or SAP Query with no filter.

Values	An integer equal or lower than 2147483647.
Default	`2147483647`

Related Topics

- SAP ERP Driver

11.6 JDBC SBO Parameters

These parameters apply to the JDBC SBO file. They are used to configure a JDBC connection.

These parameters are defined in the SBO files of `\\connectionserver-install-dir\connectionServer\jdbc` directory.

11.6.1 Connection Shareable

```
<Parameter Name="Connection Shareable">False</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies if a connection of a connection pool is shareable between different requesters. Operates in conjunction with the `Shared Connection` parameter.
Values	`True`: the connection can be shared between multiple users. `False`: the connection cannot be shared between users.
Default	`False`

Related Topics

- Shared Connection

11.6.2 Escape Character Available

```
<Parameter Name="Escape Character Available">True</Parameter>
```

Description	Specifies whether the JDBC network layer handles an escape clause after the `like` expression of the SQL query. This clause helps you to specify a character to escape specific characters such as underscores (_).
Values	`True`: an escape clause is supported. `False`: no escape clause is supported.
Default	`True`

11.6.3 ForeignKeys Available

```
<Parameter Name="ForeignKeys Available">True</Parameter>
```

Description	Specifies if foreign keys of database tables can be retrieved.
Values	`True`: foreign keys can be retrieved. `False`: foreign keys cannot be retrieved.
Default	`True`

11.6.4 Get Extended Column

```
<Parameter Name="Get Extended Column">No</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies if the Data Access layer uses the `getExtendedColumns` stored procedure provided by Data Federator Query Server to retrieve input columns.
Values	`Yes`: `getExtendedColumns` is used. `No`: `getExtendedColumns` is not used.
Default	`No`

11.6.5 JDBC Class

```
<Parameter Name="JDBC Class">string</Parameter>
```

Description	The fully qualified Java class of JDBC driver.
Values	Depends on the vendor or data source, for example: • `oracle.jdbc` for OracleDriver for Oracle • `com.ibm.db2.jcc.DB2Driver` for DB2
Default	No default value.

11.6.6 PrimaryKey Available

```
<Parameter Name="PrimaryKey Available">True</Parameter>
```

Description	Specifies whether the primary keys of database tables can be retrieved.
Values	`True`: primary keys can be retrieved. `False`: primary keys cannot be retrieved.
Default	`True`

11.6.7 PVL Available

```
<Parameter Name="PVL Available">True</Parameter>
```

Description	Specifies whether the connection supports Preferred Viewing Locale (PVL) functionality. It is only supported by SAP HANA connections in this release.
Values	`True`: PVL is supported. `False`: PVL is not supported.

11.6.8 Shared Connection

```
<Parameter Name="Shared Connection">False</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies whether the requested connection of a connection pool is shared. Operates in conjunction with the `Connection Shareable` parameter. Is not taken into account if the `Max Pool Time` parameter of `cs.cfg` is set to `0`.

Values	`True`: the connection is shared. `False`: the connection is not shared.
Default	`False`

Related Topics

- Connection Shareable
- Max Pool Time

11.6.9 JDBC ResultSet Type

```
<Parameter Name="JDBC ResultSet Type">1003</Parameter>
```

Description	**Note:** You must use this parameter along with `JDBC ResultSet Concurrency`. It is not mandatory. Specifies the `resultSetType` argument value of the `createStatement` Java method. This method belongs to the `java.sql.Connection` interface. You add this parameter to create a default `Statement` object and fine-tune your connection performance.
Values	You set either standard Java values or database-specific values. Values must be integers.

Related Topics

- createStatement method
- ResultSet interface

11.6.10 JDBC ResultSet Concurrency

```
<Parameter Name="JDBC ResultSet Concurrency">1007</Parameter>
```

Description	**Note:** You must use this parameter along with `JDBC ResultSet Type`. It is not mandatory. Specifies the `resultSetConcurrency` argument value of the `createStatement` Java method. This method belongs to the `java.sql.Connection` interface. You add this parameter to create a default `Statement` object and fine-tune your connection performance.
Values	You set either standard Java values or database-specific values. Values must be integers.

Related Topics

- createStatement method
- ResultSet interface

11.7 OData SBO Parameters

These parameters are used to configure connections to data sources that are using the OData protocol.

These parameters are defined in the `\\`*`connectionserver-install-dir`*`\connectionServer\odata\odata.sbo` file.

11.7.1 Enforce Max Protocol Version

```
<Parameter Name="Enforce Max Protocol Version">Yes</Parameter>
```

Description	Specifies if the driver sends the supported version of the OData protocol to the OData provider. If it does, the OData provider can decide whether it answers the driver in the given protocol. This parameter can be helpful when the BI platform is connecting to a provider that is using the OData 3.0 protocol.

Values	`Yes`: the driver sends the supported version of the OData protocol. `No`: the driver does not send the supported version of the OData protocol.
Default	`Yes`

11.8 ODBC SBO Parameters

These parameters apply to the ODBC network layer. They are used to configure an ODBC connection.

These parameters are defined in the SBO files of `\\`*`connectionserver-install-dir`*`\connectionServer\odbc` subdirectory.

11.8.1 CharSet

```
<Parameter Name="CharSet">ISO88591</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies the character set of the data returned by the database middleware.
Values	`ISO88591`: the default character set value specified for HP Neoview on UNIX.
Default	If not specified, Connection Server uses the character set specified in the database middleware.

11.8.2 Connection Status Available

```
<Parameter Name="Connection Status Available">True</Parameter>
```

Description	Specifies whether the middleware can detect a bad connection (ping function).
Values	`True`: the middleware can detect a bad connection. `False`: the middleware cannot detect a bad connection.
Default	The value set in the middleware.

11.8.3 Cost Estimate Available

```
<Parameter Name="Cost Estimate Available">False</Parameter>
```

Description	Specifies if the database middleware supports cost estimation of the SQL query execution. This parameter is only used with Teradata database.
Values	`True`: the middleware supports cost estimation. `False`: the middleware does not support cost estimation.
Default	`False`

11.8.4 Empty String

```
<Parameter Name="Empty String">EmptyString</Parameter>
```

Description	Specifies that certain functions, for example SQL tables, receive either an empty string or a null pointer to replace missing parameters.

Values	`NullString`: a null string is used. `EmptyString`: an empty string used.
Default	`EmptyString`

11.8.5 ODBC Cursors

```
<Parameter Name="ODBC Cursors">No</Parameter>
```

Description	Specifies if the ODBC cursor library is used by the data access driver. It can help you to improve the performance of your system.
Values	`Yes`: the ODBC cursor library is used by the data access driver. `No`: the ODBC cursor library is not used by the data access driver.
Default	The value set in the `cs.cfg` file.

11.8.6 SQLDescribeParam Available

```
<Parameter Name="SQLDescribeParam Available">True</Parameter>
```

Description	Specifies whether the middleware handles the `SQLDescribeParam` ODBC function. This function helps you describe the parameters of a stored procedure. This parameter is only used for IBM Informix database.
Values	`True`: the `SQLDescribeParam` function is available. `False`: the `SQLDescribeParam` function is not available.
Default	The value set in the database middleware.

11.8.7 SQLMoreResults Available

```
<Parameter Name="SQLMoreResults Available">True</Parameter>
```

Description	Specifies whether the middleware handles the `SQLMoreResults` ODBC function. This function helps you to retrieve more result sets from SQL execution, if any. This parameter is supported by ODBC drivers only.
Values	`True`: the `SQLMoreResults` function is supported. `False`: the `SQLMoreResults` function is not supported.
Default	The value set in the middleware.

11.8.8 Use DataDirect OEM Driver

```
<Parameter Name="Use DataDirect OEM Driver">No</Parameter>
```

Description	Specifies whether connections to MS SQL Server databases can use DataDirect ODBC branded drivers.
Values	`Yes`: the connection can use the branded driver. `No`: the connection cannot use the branded driver.
Default	`No`

Related Topics

- About DataDirect ODBC Drivers

11.8.9 V5toV6DriverName

```
<Parameter Name="V5toV6DriverName">{Informix 3.34 32 BIT}</Parameter>
```

Description	Specifies the conversion rule from `Informix Connect` to `Informix ODBC`. The value of this parameter determines which Informix Driver is used to define the ODBC Data Source Name (DSN) without the connection string. This parameter is only used for IBM Informix database.
Values	The exact name of the Informix driver installed on the machine.
Default	The value set in the `cs.cfg` file.

11.9 OLE DB SBO Parameters

These parameters apply to the OLE DB database technology. They are used to configure an OLE DB connection.

These parameters are defined in the `\\`*`connectionserver-install-dir`*`\connectionServer\oledb\oledb.sbo` and `\sqlsrv.sbo` files.

11.9.1 Enumerator CLSID

```
<Parameter Name="Enumerator CLSID">MSDASQL Enumerator</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies class ID of OLE DB enumerator. This parameter is used with OLE DB only.

11.9.2 Provider CLSID

```
<Parameter Name="Provider CLSID">MSDASQL</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies class ID of OLE DB provider. This parameter is used with OLE DB only.

11.10 OLE DB OLAP SBO Parameters

These parameters apply to the OLE DB OLAP database technology. They are used to configure an OLE DB connection for OLAP.

These parameters are defined in the `\\connectionserver-install-dir\connectionServer\oledb_olap\slqsrv_as.sbo` file.

11.10.1 MSOlap CLSID

```
<Parameter Name="MSOlap CLSID">msolap.4</Parameter>
```

Description	**Note:** Do not edit this parameter. Specifies class ID of OLE DB provider. This parameter is used with OLE DB for OLAP only.

11.11 Sybase SBO Parameters

These parameters apply to the Sybase ASE/CTLIB SBO file. These parameters are used to configure a Sybase ASE/CTLIB connection.

These parameters are defined in the `\\connectionserver-install-dir\connectionServer\sybase\sybase.sbo` file.

11.11.1 Driver Behavior

```
<Parameter Name="Driver Behavior">Dynamic</Parameter>
```

Description	Specifies which one of the Sybase drivers is used.
Values	`Dynamic`: the ct_dynamic driver is used. Any other value enables the use of the CTLib driver.
Default	`Dynamic`.

11.11.2 Password Encryption

```
<Parameter Name="Password Encryption">True</Parameter>
```

Description	Specifies if you want to use the encryption password mechanism specified in the middleware for the password entered in the **Connection details** dialog box. This parameter is used only with Sybase. It is included in the `Defaults` section for future compatibility.
Values	`True`: the encryption password mechanism of the middleware is used. `False`: the encryption password mechanism of the middleware is not used.
Default	The value set in the `cs.cfg` file.

11.11.3 Quoted Identifier

```
<Parameter Name="Quoted Identifier">False</Parameter>
```

Description	Specifies whether quoted identifiers are supported. This parameter is supported by Sybase middleware only.
Values	`True`: quoted identifiers are supported. `False`: quoted identifiers are not supported.
Default	`False`

11.11.4 Recover Errors

```
<Parameter Name="Recover Errors">True</Parameter>
```

Description	Specifies whether the Client Library driver is usable after ct_dynamic driver failure.
Values	`True`: the Client Library driver is used. `False`: the Client Library driver is not used.
Default	`True`

11.11.5 Text Size

```
<Parameter Name="Text Size">32768</Parameter>
```

Description	**Note:** This parameter is not mandatory. Specifies the maximum size of large variable length binary or character data supported. Large variable length binary or character data retrieved from Sybase CTL database is truncated if larger than 32 KBytes. You add this parameter to the appropriate `Database` section of the configuration file to avoid the data truncation.
Default	`32768`

11.12 Teradata SBO Parameters

These parameters apply to the Teradata SBO files. These parameters are used to configure a Teradata connection through JDBC or ODBC.

These parameters are defined in the `\\`*`connectionserver-install-dir`*`\connectionServer\jdbc\teradata.sbo` and `\\`*`connectionserver-install-dir`*`\connectionServer\odbc\teradata.sbo` files.

11.12.1 Replace Current Owner With Database

```
<Parameter Name="Replace Current Owner With Database">Yes</Parameter>
```

Description	Specifies if the driver returns the current database name when the data source is queried for the current owner. Only used with ODBC connections.
Values	`Yes`: the current owner is mapped to the Teradata database name. `No`: the current owner is mapped to the current user name.
Default	`No`

Related Topics

- Teradata Connections - Mapping Teradata Database to Current Owner

The following section provides configuration information for Teradata connections.

Configuring Database Capability Parameters

12.1 About Database Capability Parameters

Database capability parameters describe capabilities of databases used as sources of data for the contents of a universe. You can set these parameters to operate at the following levels:

- Universe level

  You set these parameters when you create or modify a universe.

- Database level

  You set these parameters in the PRM file of the database. They are overridden by any corresponding setting at universe level.

Note:
To view the operators and functions available for your data access driver, open the `<driver>.prm` file in an XML editor.

12.2 About PRM Files

PRM files provide you with parameters that describe capabilities of databases used as sources of data for SAP BusinessObjects applications. They allow database-dependent factors to control what SQL can be used with the universe, based on the connection and the database that it is attached to. There is a PRM file corresponding to each database driver.

You can configure some database capability parameters from inside a universe. They override PRM file settings consequently.

PRM files are located in the following folders:

- *connectionserver-install-dir*`\connectionServer\`*`<RDBMS>`* directories, where *`<RDBMS>`* is the network layer or middleware name.
- *connectionserver-install-dir*`\connectionServer\`*`<RDBMS>`*`\extensions\qt` directories. These PRM files are called extended files. For more information about parameters of extended files, see the SQL and MDX reference chapter in the *Information Design Tool User Guide*.

Note:
Help text files in other languages are also available in the same directories.

12.2.1 PRM Parameter File Structure

There is a PRM file for each supported database driver. Each file is divided into sections, which contain specific parameters. The following table describes the content and meaning of each PRM file section.

File Section	Description
`Configuration`	Parameters used to describe capabilities of databases used as source of data for universes, for example `EXT_JOIN`, `ORDER_BY`, and `UNION`. They are not directly available to any SAP BusinessObjects product. These parameters can be edited to optimize queries run against universes using the target data access driver. **Note:** They are described in the next chapter.
`DateOperations`	Date operators available to universe design tool and information design tool, for example `YEAR`, `QUARTER`, `MONTH`.
`Operators`	Operators available to universe design tool and information design tool, for example `ADD`, `SUBSTRACT`, `MULTIPLY`.

File Section	Description

Functions	Functions available to universe design tool and information design tool, for example `Average`, `Sum`, `Variance`. See the *Information Design Tool User Guide* for more information.
	
	Help text that appears when functions in this section are selected in universe designer tool and information design tool is listed in the file `<driver><language>.prm`, for example, `oracleen.prm`. This file is found in the same directory as the `<driver>.prm` file. You can open it to view descriptions of all the functions available in the `<driver>.prm` file.
	
	The `Functions` section has the following child elements:
	• `Group`: whether the usage of this function in a query generates a `group by` clause.
	  • `True` sets that the query generates a `group by` clause.
	  • `False` sets that the query does not generate a `group by` clause.
	• `ID`: this is the name that appears in the "User Objects" function list in Desktop Intelligence. Deprecated in this release.
	• `InMacro`: if this value is `True`, then the function is listed in the "User Objects" function list in Desktop Intelligence. Deprecated in this release.
	• `Type`: function data type.
	• `Arguments`: arguments accepted by the function. A function can have a maximum of only four arguments, and any additional arguments are not taken into account.
	• `SQL`: the SQL syntax for the function.

Related Topics

- To View and Edit a Function Help Text File

12.3 To View and Edit PRM Files

1. Browse to the directory that stores the PRM file for your target data access driver.
   PRM files are stored in `connectionserver-install-dir\connectionServer\<RDBMS>`directories.
2. Open a `<driver>.prm` file in an XML editor.

3. Expand sections as required.
4. Set values by entering the value in the appropriate tag.
5. Save and close the file.

12.4 To Verify and Add Analytic Function Support to PRM Files

The PRM files may not contain all the analytic functions available in the targeted database. Before using an analytic function, you should verify that it is present in the file, and if necessary, add it to the list.

You need to update the PRM file only if the function is both analytic and aggregate, such as SUM.

1. Browse to the directory that stores the extended PRM file for your target data access driver.
2. Open the `<driver>.prm` file in an XML editor.
3. Under the `Configuration` section, verify that the following parameter is present:

   ```
   <Parameter Name="OVER_CLAUSE">Y</Parameter>
   ```

   For a definition of this parameter, see the *Information Design Tool User Guide*.
4. Browse to the directory that stores the PRM file for your target data access driver.
5. Open the `<driver>.prm` file in an XML editor.
6. Under the `Configuration` section, verify that the ID of the analytic function is listed in the `ANALYTIC_FUNCTIONS` parameter value.
7. If not present, type the ID of the function at the end of the list.
8. Under `Functions`, add the `Function` section and specify the ID, name, argument types and SQL definition of the analytic function.

   ```
   <Function Group="False" ID="Function_ID" InMacro="False" Name="Function_Name" Type ="All">
    <Arguments>
     <Argument Type="Argument_Type"></Argument>
     ...
    </Arguments>
    <SQL>SQL_Definition</SQL>
   </Function>
   ```

   Remember:
   Give different names and IDs to a function that is both analytic and aggregate. For example, the name of SUM as aggregate function is SUM. The name of SUM as analytic function is SUM_OVER.
9. Save any modifications and close the file.

You have added support of an analytic function to the information design tool.

You need to restart the information design tool for any changes to the PRM file to take effect.

Related Topics

- About PRM Files

12.5 To View and Edit a Function Help Text File

The Help text that appears under each function when selected in the universe design tool or information design tool, is maintained in a separate XML file. You can edit and add text to describe a function by editing the `<driver><language>.prm` file. There is a Help text file for each language version of SAP BusinessObjects products installed.

When you add a function to the PRM file, you need to add the Help text for the new function to the appropriate `<driver><language>.prm` file, for example, if you add a function to the `oracle.prm` file, then you also add the function name and the Help text for the function to the `oracleen.prm` file, if you are working with the English version of the tool. See the ISO639-1 standard.

12.6 To Edit the Help Text for a PRM Function

1. Browse to the directory that stores the PRM language file for your target data access driver.

   PRM language files are stored in `connectionserver-install-dir\connectionServer\<RDBMS>` directories.
2. Open a `<driver><language>.prm` file in an XML editor.
3. Expand the `Messages` section.
4. To add Help for a new function do the following:
   - Add a new section for a function. The easiest way to do this is to copy an existing function entry and copy it into the `Function` section. You then edit the new function text.
   - Enter Help text for the function.
5. To view or edit existing function Help text, do the following:
   - Expand the `Function` section.
   - Expand the `Message` section for a function.
   - Edit Help text as required.
6. Save and close the file.

PRM Parameter Reference

13.1 PRM file Configuration Reference

The database capability parameters are listed alphabetically. To view functions, date operators, and other operators available, open a `<driver>.prm` file in an XML editor, each parameter is defined in the following tag:

```
<Parameter Name="parameter">value</Parameter>
```

where *parameter* is the name of the parameter, and *value* is the value attributed to the parameter.

Each parameter is shown with the following information:

- Example of how the parameter appears in the XML file
- Description of the parameter
- Possible values that can be set for the parameter
- Default value for the parameter if any

Note:
Certain configuration parameters must not be edited. These parameters have values set for use internally within SAP BusinessObjects solutions. These parameters are described in this section but contain a warning not to edit the value. You must not edit these parameters. Before editing any other `PRM` file parameter, you should make a backup copy of the `PRM` file.

13.1.1 ANALYTIC_CLAUSE

```
<Parameter Name="ANALYTIC_CLAUSE">WHEN</Parameter>
```

Description	Specifies which SQL keyword must be used if a function specified in the `ANALYTIC_FUNCTIONS` parameter is used in the SQL statement.

Values	`WHEN`: indicates a WHEN clause is used. This is the default value for IBM Red Brick databases. `QUALIFY`: indicates a QUALIFY clause is used. This is the default value for Teradata databases. **Note:** Refer to your documentation database to find how it implements the SQL clause.
Default	See values above.

Related Topics

- ANALYTIC_FUNCTIONS

13.1.2 ANALYTIC_FUNCTIONS

```
<Parameter Name="ANALYTIC_FUNCTIONS">RANK,SUM,AVG,COUNT,MIN,MAX</Parameter>
```

Description	Lists the analytic functions supported by the database.
Values	See values in the PRM files.
Default	The listed values.

13.1.3 CALCULATION_FUNCTION

```
<Parameter Name="CALCULATION_FUNCTION">YES</Parameter>
```

Description	Specifies if the database supports calculation functions.

Values	`YES`: the database supports calculation functions. If so, the information design tool is able to create calculation columns. `NO`: the database does not support calculation functions.

13.1.4 CONSTANT_SAMPLING_SUPPORTED

```
<Parameter Name="CONSTANT_SAMPLING_SUPPORTED">YES</Parameter>
```

Description	Specifies if the database supports random sampling. Random sampling consists in extracting random rows from a dataset.
Values	`YES`: the database supports random sampling. `NO`: the database does not support random sampling.
Default	`YES`

13.1.5 DISTINCT

```
<Parameter Name="DISTINCT">YES</Parameter>
```

Description	Specifies if the database supports the DISTINCT keyword in SQL statements. This parameter is used with MS Access.

Values	`YES`: the database supports the DISTINCT keyword. `NO`: the database does not support the DISTINCT keyword. This behavior disables the following features: • The **Distinct Values** option that appears when you click the **View Values** button in the **Quick Design** wizard of the universe design tool. • The `Countdistinct` function that appears when you create a condition with the Calculation operand in the **Query Panel**.
Default	`YES`

13.1.6 EXT_JOIN

```
<Parameter Name="EXT_JOIN">YES</Parameter>
```

Description	Specifies if the database supports outer joins. This parameter is used by all data access drivers.
Values	`YES`: the database supports outer joins. `NO`: the database does not support outer joins. The **Outer join** check boxes in the **Edit Join** dialog box of the universe design tool are greyed.
Default	`YES`

13.1.7 FULL_EXT_JOIN

```
<Parameter Name="FULL_EXT_JOIN">YES</Parameter>
```

Description	Specifies if the database supports full outer joins.

Values	`YES`: the database supports full outer joins. `NO`: the database does not support full outer joins.
Default	`YES`

13.1.8 GROUP_BY

```
<Parameter Name="GROUP_BY">NO</Parameter>
```

Description	Specifies if the database supports the GROUP BY clause in the SQL statements.
Values	`YES`: the database supports the GROUP BY clause. `NO`: the database does not support the GROUP BY clause.

13.1.9 GROUP_BY_SUPPORTS_COLUMN_INDEX

```
<Parameter Name="GROUP_BY_SUPPORTS_COLUMN_INDEX">NO</Parameter>
```

Description	Specifies if the database supports the use of column indexes from the SELECT statement in the GROUP BY clause.
Values	`YES`: the database supports the use of column indexes from the SELECT statement instead of column names. `NO`: the database does not support the use of column indexes from the SELECT statement instead of column names.
Default	`NO`

13.1.10 GROUP_BY_SUPPORTS_COMPLEX

```
Parameter Name="GROUP_BY_SUPPORTS_COMPLEX">NO</Parameter>
```

Description	Specifies if the database supports complex expressions in the GROUP BY clause. Complex expressions mean anything than column names or indexes present in the SELECT statement, for example functions or columns absent from the SELECT statement. This parameter is used with IBM DB2. **Note:** If you run a query containing measure objects and complex objects, that is, objects using functions and concatenation, your application displays the following error message: `Your database does not allow you to do aggregations with the <object name> object.`
Values	`YES`: the database supports complex expressions in the GROUP BY clause. `NO`: the database does not support complex expressions in the GROUP BY clause.
Default	`NO`

13.1.11 GROUP_BY_SUPPORTS_CONSTANT

```
<Parameter Name="GROUP_BY_SUPPORTS_CONSTANT">YES</Parameter>
```

Description	Specifies if the database supports the use of constant objects in the GROUP BY clause. This parameter is used by IBM DB2 and Microsoft SQL Server databases.

Values	`YES`: the database supports the use of constant objects in the GROUP BY clause. `NO`: the database does not support the use of constant objects in the GROUP BY clause.
Default	`YES`

13.1.12 HAVING

```
<Parameter Name="HAVING">NO</Parameter>
```

Description	Specifies if the database supports the HAVING clause in SQL statements.
Values	`YES`: the database supports the HAVING clause. `NO`: the database does not support the HAVING clause.
Default	`NO`

13.1.13 INTERSECT

```
<Parameter Name="INTERSECT">INTERSECT</Parameter>
```

Description	Specifies the keyword supported by the database for the INTERSECT set operation.
Values	`INTERSECT`: the keyword supported by the database is INTERSECT. No value: the database does not support any keyword for the INTERSECT set operation. In this case, two queries are generated.

Default	INTERSECT

13.1.14 INTERSECT_ALL

```
<Parameter Name="INTERSECT_ALL">YES</Parameter>
```

Description	Specifies if the database supports the INTERSECT ALL set operation.
Values	YES: the database supports the INTERSECT ALL set operation. NO: the database does not support the INTERSECT ALL set operation.
Default	YES

13.1.15 INTERSECT_IN_SUBQUERY

```
<Parameter Name="INTERSECT_IN_SUBQUERY">YES</Parameter>
```

Description	Specifies if the database supports the INTERSECT set operation in subqueries.
Values	YES: the database supports the INTERSECT set operation in subqueries. NO: the database does not support the INTERSECT set operation in subqueries.

13.1.16 JOIN

```
<Parameter Name="JOIN">YES</Parameter>
```

Description	Specifies if the database supports a JOIN operation between two tables.
Values	`YES`: the database supports joins between any columns of two tables. `STRUCTURE_JOIN`: the database supports joins between two tables that are related with referential constraints. Is identical to `NO` in this release. `NO`: the database does not support joins between two tables.

13.1.17 LEFT_EXT_JOIN

```
<Parameter Name="LEFT_EXT_JOIN">YES</Parameter>
```

Description	Specifies if the database supports left external joins.
Values	`YES`: the database supports left external joins. `NO`: the database does not support left external joins.

13.1.18 LEFT_OUTER

```
<Parameter Name="LEFT_OUTER">$(+)</Parameter>
<Parameter Name="LEFT_OUTER">$*</Parameter>
```

Description	Specifies the syntax to be used for left outer join expressions.

Values	`$(+)`: this syntax is used with Oracle. `$*`: this syntax is used with Sybase, MS SQL Server and IBM Red Brick. **Note:** `$` represents a join expression.
Default	See values above.
Example	If `table1.col1` is joined to `table2.col2` in Oracle, the expression generated is then : `table1.col1(+) = table2.col2`.

13.1.19 LIKE_SUPPORTS_ESCAPE_CLAUSE

```
<Parameter Name="LIKE_SUPPORTS_ESCAPE_CLAUSE">YES</Parameter>
```

Description	Specifies if the database supports the use of an ESCAPE clause within the LIKE condition of the SQL statement.
Values	`YES`: the database supports the ESCAPE clause in the LIKE condition. `NO`: the database does not support the ESCAPE clause in the LIKE condition.
Default	If this setting is not specified, Connection Server retrieves the information from the database middleware.

13.1.20 MINUS

```
<Parameter Name="MINUS">MINUS</Parameter>
```

Description	Specifies the keyword supported by the database for the MINUS set operation.

Values	`MINUS`: the database supports the MINUS set operator. `EXCEPT`: the database supports the EXCEPT set operator. No value: the database does not support any keyword for the MINUS set operation. In this case, two queries are generated.
Default	`MINUS`

13.1.21 MINUS_ALL

```
<Parameter Name="MINUS_ALL">Yes</Parameter>
```

Description	Specifies if the database supports the MINUS ALL set operation.
Values	`YES`: the database supports the MINUS ALL set operation. `NO`: the database does not support the MINUS ALL set operation.

13.1.22 MINUS_IN_SUBQUERY

```
<Parameter Name="MINUS_IN_SUBQUERY ">YES</Parameter>
```

Description	Specifies if the database supports the MINUS set operation in sub-queries.
Values	`YES`: the database supports the MINUS set operation in subqueries. `NO`: the database does not support the MINUS set operation in sub-queries.

13.1.23 ORDER_BY

```
<Parameter Name="ORDER_BY">YES</Parameter>
```

Description	Specifies if the database supports the ORDER BY clause.
Values	`YES`: the database supports the ORDER BY clause. `NO`: the database does not support the ORDER BY clause.

13.1.24 ORDER_BY_REQUIRES_SELECT

```
<Parameter Name="ORDER_BY_REQUIRES_SELECT">NO</Parameter>
```

Description	Specifies if the database requires columns used in the ORDER BY clause to be referenced in the SELECT statement.
Values	`YES`: users are not allowed to sort on columns if they are not included in the SELECT statement. In this case, the **Manage Sorts** button is greyed in the **Query** Panel of the universe design tool. `NO`: users are allowed to sort on columns even if they are not included in the SELECT statement.
Default	`NO`

13.1.25 ORDER_BY_SUPPORTS_COLUMN_INDEX

```
<Parameter Name="ORDER_BY_SUPPORTS_COLUMN_INDEX">YES</Parameter>
```

Description	Specifies if the database supports the use of column indexes from the SELECT statement in the ORDER BY clause.
Values	`YES`: the database supports the use of column indexes from the SELECT statement instead of column names. `NO`: the database does not support the use of column indexes from the SELECT statement instead of column names.

13.1.26 PERCENT_RANK_SUPPORTED

```
<Parameter Name="PERCENT_RANK_SUPPORTED">YES</Parameter>
```

Description	Specifies if the database supports the Percent Rank analytical function. Refer to your database documentation to find how it implements percent rank.
Values	`YES`: the database supports Percent Rank. `NO`: the database does not support Percent Rank.
Default	`YES`

13.1.27 RANK_SUPPORTED

```
<Parameter Name="RANK_SUPPORTED">YES</Parameter>
```

Description	Specifies if the database supports the Rank analytical function in SQL statements.
Values	`YES`: the database supports Rank. `NO`: the database does not support Rank.

Default	`YES`

13.1.28 RIGHT_EXT_JOIN

```
<Parameter Name="RIGHT_EXT_JOIN">YES</Parameter>
```

Description	Specifies if the database supports right external joins.
Values	`YES`: the database supports right external joins. `NO`: the database does not support right external joins.

13.1.29 RIGHT_OUTER

```
<Parameter Name="RIGHT_OUTER">$(+)</Parameter>
```

```
<Parameter Name="RIGHT_OUTER">*$</Parameter>
```

Description	Specifies the syntax to be used for right outer join expressions.
Values	`$(+)`: this syntax is used with Oracle. `*$`: this syntax is used with Sybase, MS SQL Server and IBM Red Brick. **Note:** `$` represents a join expression.
Default	See values above.

13.1.30 SEED_SAMPLING_SUPPORTED

```
<Parameter Name="SEED_SAMPLING_SUPPORTED">YES</Parameter>
```

Description	Specifies if the database supports seed sampling. Seed sampling is a variation of random sampling in which the random seed is provided by the user.
Values	`YES`: the database supports seed sampling. `NO`: the database does not support seed sampling.
Default	`NO`

13.1.31 SELECT_SUPPORTS_NULL

```
<Parameter Name="NULL_IN_SELECT_SUPPORTED">YES</Parameter>
```

Description	Specifies if the database supports NULL as a column in the SELECT statement.
Values	`YES`: the database supports NULL as a column in the SELECT statement. `NO`: the database does not support NULL as a column in the SELECT statement.
Default	`YES` `NO` for Teradata, IBM DB2, IBM Informix and IBM Red Brick databases, which do not support the NULL value as column.

13.1.32 SUBQUERY_IN_FROM

```
<Parameter Name="SUBQUERY_IN_FROM">YES</Parameter>
```

Description	Specifies if the database supports the use of subqueries inside the FROM clause.
Values	`YES`: the database supports the use of subqueries inside the FROM clause. `NO`: the database does not support the use of subqueries inside the FROM clause.

13.1.33 SUBQUERY_IN_IN

```
<Parameter Name="SUBQUERY_IN_IN">YES</Parameter>
```

Description	Specifies if the database supports the use of subqueries inside the IN clause.
Values	`YES`: the database supports the use of subqueries inside the IN clause. `NO`: the database does not support the use of subqueries inside the IN clause.

13.1.34 SUBQUERY_IN_WHERE

```
<Parameter Name="SUBQUERY_IN_WHERE">YES</Parameter>
```

Description	Specifies if the database supports the use of subqueries inside the WHERE clause.

Values	`YES`: the database supports the use of subqueries inside the WHERE clause. `NO`: the database does not support the use of subqueries inside the WHERE clause.

13.1.35 TECHNICAL_COLUMN_NAME_PATTERN

```
<Parameter Name="TECHNICAL_COLUMN_NAME_PATTERN">^(-idref)(.)*</Parameter>
```

Description	Specifies that all columns beginning with `-idref` pattern are not displayed in the connections, data foundations and business layers of universes based on OData data sources. Consequently, the information designer cannot build queries with these columns. **Note:** Do not edit this parameter.
Value	`^(-idref)(.)*`

13.1.36 UNION

```
<Parameter Name="UNION">UNION</Parameter>
```

Description	Specifies the keyword supported by the database for the UNION set operation.
Values	`UNION`: the keyword supported by the database is UNION. No value : the database does not support any ekyword for the UNION set operation. In this case, two queries are generated.
Default	`UNION`

13.1.37 UNION_ALL

```
<Parameter Name="UNION_ALL">YES</Parameter>
```

Description	Specifies if the database supports the UNION ALL set operation.
Values	`YES`: the database supports the UNION ALL set operation. `NO`: the database does not support the UNION ALL set operation.

13.1.38 UNION_IN_SUBQUERY

```
<Parameter Name="UNION_IN_SUBQUERY">YES</Parameter>
```

Description	Specifies if the database supports the use of the UNION set operation in subqueries.
Values	`YES`: the database supports the UNION set operation in subqueries. `NO`: the database does not support the UNION set operation in subqueries.

Data Type Conversion Reference

14.1 Data Type Conversion

In the information design tool, data foundations expose tables from one or more relational databases, which are the basis for business layers. The data type associated with each table column is displayed in a data foundation with other column details. Business layers expose table columns as metadata objects, like dimensions and hierarchies, and display the data type associated with each object. The following table describes the mapping between these two sets of data types:

Data Type Shown in the Data Foundation	Data Type Shown in the Business Layer
BINARY, LONGVARBINARY, VARBINARY	BLOB
BIT	Boolean
DATE	Date
TIME, TIMESTAMP	DateTime
LONGVARCHAR	Long Text
BIGINT, DECIMAL, DOUBLE, FLOAT, INTEGER, NUMERIC, REAL, SMALLINT, TINYINT	Numeric
CHAR, VARCHAR, XML	String
UNDEFINED	Unknown

Note:
From any database, a time is managed as a DateTime in the BI platform. The date part of the DateTime should be normally displayed as the current date in end-user reports.

The Data Access layer manages the conversion of data types exposed by network layers into Connection Server data types, which are then mapped to data types exposed in data foundations.

This section provides conversion tables between generic network layer data types (JDBC and ODBC) and data foundation data types. It also provides conversion tables for specific network layers such as OLE DB, Oracle OCI, and Sybase CTL, and describes nontrivial conversions and exceptions for specific databases, such as CSV files, SAP ERP systems, and the SAP HANA database.

Related Topics

- CSV File Data Types
- JDBC Data Types
- ODBC Data Types
- OData Data Types
- OLE DB Data Types
- Oracle OCI Data Types
- SAP ERP Data Types
- SAP HANA Data Types
- Sybase CTL Data Types
- XML Data Types

14.1.1 CSV File Data Types

The following table lists the data types declared in DDL files for schema detection of CSV files, and their equivalent in data foundations.

Type Declared in the DDL File	Data Foundation Data Type
BIT, BOOLEAN	BIT
DATE	DATE
TIME	TIME
TIMESTAMP	TIMESTAMP
BIGINT, DECIMAL	DECIMAL
FLOAT, DOUBLE, REAL	DOUBLE

Type Declared in the DDL File	Data Foundation Data Type
INTEGER, INT, SMALLINT	INTEGER
NUMBER, NUMERIC	NUMERIC
VARCHAR	VARCHAR

Related Topics

- CSV Schema Detection

14.1.2 JDBC Data Types

The following table lists the data types that show up through JDBC and their equivalent in data foundations.

JDBC Data Type	Data Foundation Data Type
BINARY	BINARY
VARBINARY	VARBINARY
BLOB, LONGVARBINARY	LONGVARBINARY
BIT, BOOLEAN	BIT
CLOB, NCLOB	LONGVARCHAR
DATE	DATE
TIME	TIME
TIMESTAMP	TIMESTAMP

JDBC Data Type	Data Foundation Data Type
DOUBLE	DOUBLE
FLOAT	FLOAT
REAL	REAL
SMALLINT	SMALLINT
CHAR, NCHAR	CHAR
LONGNVARCHAR, LONGVARCHAR, NVARCHAR, ROWID, VARCHAR	VARCHAR
SQLXML	XML

IBM Informix

The following table lists the data types that could show up in results when the user queries an IBM Informix database through JDBC, and their equivalent in data foundations. These data types overwrite the generic ones.

Informix JDBC Data Type	Data Foundation Data Type
TIMESTAMP	It can be mapped to the following type: • DATE if the Informix data type follows the pattern `DATETIME\\s+(HOUR|MINUTE|SECOND)\\s+TO\\s+(HOUR|MINUTE|SECOND)` • TIME if the Informix data type follows the pattern `DATETIME\\s+(YEAR|MONTH|DAY)\\s+TO\\s+(YEAR|MONTH|DAY)` • TIMESTAMP in the other cases
BOOLEAN	SMALLINT

MS SQL Server

The following table lists the data types that could show up in results when the user queries an MS SQL Server database through JDBC, and their equivalent in data foundations. The data type mapping also

depends on the SQL type name associated with the network layer data type. These data types overwrite the generic ones.

MS SQL Server JDBC Data Type	SQL Type Name	Data Foundation Data Type
LONGVARCHAR	any other value than "xml"	VARCHAR
LONGVARCHAR	xml	XML

Oracle

The following table lists the data types that could show up in results when the user queries an Oracle database through JDBC, and their equivalent in data foundations. The data type mapping also depends on the SQL type name associated with the network layer data type. These data types overwrite the generic ones.

Oracle JDBC Data Type	SQL Type Name	Data Foundation Data Type
BFILE	any value	LONGVARBINARY
any value	BFILE	LONGVARBINARY
DATE	any value	TIMESTAMP
any value	TIMESTAMP	TIMESTAMP
BINARY_DOUBLE	any value	DOUBLE
OTHER	FLOAT	DOUBLE
BINARY_FLOAT	any value	REAL
any value	NCHAR	CHAR
any value	NVARCHAR2, ROWID, UROWID	VARCHAR

14.1.3 ODBC Data Types

The following table lists the data types that show up through ODBC and their equivalent in data foundations.

ODBC Data Type	Data Foundation Data Type
SQL_BINARY	BINARY
SQL_VARBINARY	VARBINARY
SQL_LONGVARBINARY	LONGVARBINARY
SQL_BIT	BIT
SQL_DATE, SQL_TYPE_DATE	DATE
SQL_DATETIME, SQL_TIME, SQL_TIMESTAMP, SQL_TYPE_TIME, SQL_TYPE_TIMESTAMP	TIMESTAMP
SQL_LONGVARCHAR, SQL_WLONGVARCHAR	LONGVARCHAR
SQL_BIGINT	BIGINT
SQL_DECIMAL	DECIMAL
SQL_DOUBLE	DOUBLE
SQL_FLOAT	FLOAT
SQL_INTEGER	INTEGER
SQL_NUMERIC	NUMERIC
SQL_REAL	REAL

ODBC Data Type	Data Foundation Data Type
SQL_SMALLINT	SMALLINT
SQL_TINYINT	TINYINT
SQL_CHAR, SQL_GUID, SQL_WCHAR	CHAR
SQL_VARCHAR, SQL_WVARCHAR	VARCHAR

IBM DB2

The following table lists the data types that could show up in results when the user queries an IBM DB2 database through ODBC, and their equivalent in data foundations. These data types overwrite the generic ones.

DB2 ODBC Data Type	Data Foundation Data Type
SQL_BLOB	LONGVARBINARY
SQL_CLOB, SQL_DBCLOB, SQL_LONGVAR-GRAPHIC	LONGVARCHAR
SQL_DECFLOAT	DOUBLE
SQL_GRAPHIC	CHAR
SQL_VARGRAPHIC	VARCHAR
SQL_XML	XML

IBM Informix

The following table lists the data types that could show up in results when the user queries an IBM Informix database through ODBC, and their equivalent in data foundations. These data types overwrite the generic ones.

Informix ODBC Data Type	Data Foundation Data Type
SQL_BIT	SMALLINT
SQL_INFX_UDT_BLOB	LONGVARBINARY
SQL_TIMESTAMP, SQL_TYPE_TIMESTAMP	It can be mapped to the following type: • DATE if the Informix data type follows the pattern `DATETIME\\s+(HOUR|MINUTE|SECOND)\\s+TO\\s+(HOUR|MINUTE|SECOND)` • TIME if the Informix data type follows the pattern `DATETIME\\s+(YEAR|MONTH|DAY)\\s+TO\\s+(YEAR|MONTH|DAY)` • TIMESTAMP in the other cases
SQL_INFX_UDT_CLOB	LONGVARCHAR

MS SQL Server

The following table lists the data types that could show up in results when the user queries an MS SQL Server database through ODBC, and their equivalent in data foundations. These data types overwrite the generic ones.

MS SQL Server ODBC Data Type	Data Foundation Data Type
SQL_SS_TIME2, SQL_SS_TIMESTAMPOFFSET	TIMESTAMP
SQL_SS_XML	XML

14.1.4 OData Data Types

The following table lists the Entity Data Model (EDM) data types internal to OData data sources and their equivalent in data foundations.

OData Data Type	Data Foundation Data Type
Edm.Binary	BINARY
Edm.Boolean	BIT
Edm.DateTime, Edm.Time	TIMESTAMP **Note:** Nanoseconds are not mapped.
Edm.Int64	BIGINT
Edm.Decimal	DECIMAL
Edm.Double	DOUBLE
Edm.Float, Edm.Single	FLOAT
Edm.Int32	INTEGER
Edm.Byte, Edm.Int16, Edm.SByte	SMALLINT
Edm.DateTimeOffset, Edm.Guid, Edm.String	VARCHAR

14.1.5 OLE DB Data Types

The following table lists the data types that show up through OLE DB and their equivalent in data foundations.

Note:
For some data types, the mapping depends on the values of `DBCOLUMNFLAGS_ISLONG` and `DBCOLUMNFLAGS_ISFIXEDLENGTH` parameters associated with each type.

OLE DB Data Type	Data Foundation Data Type
DBTYPE_BYTES	VARBINARY
DBTYPE_BYTES if `DBCOLUMNFLAGS_ISLONG=true`	LONGVARBINARY
DBTYPE_BYTES if `DBCOLUMNFLAGS_IS FIXEDLENGTH=true`	BINARY
DBTYPE_BOOL	BIT
DBTYPE_DBDATE	DATE
DBTYPE_DBTIME	TIME
DBTYPE_DBTIMESTAMP	TIMESTAMP
DBTYPE_DECIMAL	DECIMAL
DBTYPE_I1, DBTYPE_UI1	TINYINT
DBTYPE_I2, DBTYPE_UI2	SMALLINT
DBTYPE_I4, DBTYPE_UI4	INTEGER
DBTYPE_I8, DBTYPE_UI8	BIGINT
DBTYPE_R4	REAL
DBTYPE_R8	DOUBLE
DBTYPE_CY, DBTYPE_NUMERIC	NUMERIC
DBTYPE_STR, DBTYPE_WSTR	VARCHAR

OLE DB Data Type	Data Foundation Data Type
DBTYPE_STR, DBTYPE_WSTR if `DBCOLUMN FLAGS_ISLONG=true`	LONGVARCHAR
DBTYPE_STR, DBTYPE_WSTR if `DBCOLUMN FLAGS_ISFIXEDLENGTH=true`	CHAR

MS SQL Server Data Types

The following table shows the data types that could show up in results when the user queries an MS SQL Server database through OLE DB, and their equivalent in data foundations. These data types overwrite the previous ones.

MS SQL Server OLE DB Data Type	Data Foundation Data Type
DBTYPE_DBTIME2, DBTYPE_DBTIMESTAM-POFFSET	TIMESTAMP
DBTYPE_GUID	CHAR
DBTYPE_XML	XML

14.1.6 Oracle OCI Data Types

The following table lists the data types that show up through Oracle OCI and their equivalent in data foundations.

Oracle OCI Data Type	Data Foundation Data Type
SQLT_BIN	BINARY
SQLT_BFILE, SQLT_BLOB, SQLT_LBI	LONGVARBINARY

Oracle OCI Data Type	Data Foundation Data Type
SQLT_DAT, SQLT_DATE, SQLT_TIME, SQLT_TIMESTAMP, SQLT_TIMESTAMP_LTZ, SQLT_TIMESTAMP_TZ	TIMESTAMP
SQLT_CLOB, SQLT_LNG	LONGVARCHAR
SQLT_FLT, SQLT_IBDOUBLE	DOUBLE
SQLT_IBFLOAT	REAL
SQLT_NUM	NUMERIC
SQLT_AFC	CHAR
SQLT_CHR, SQLT_RDD, SQLT_RID	VARCHAR

14.1.7 SAP ERP Data Types

The following table lists the ABAP data types used by SAP ERP systems and their equivalent in data foundations.

ABAP Data Type	Data Foundation Data Type
HEXADECIMAL	BINARY
DATE	DATE
TIME	TIME
FLOAT	FLOAT

ABAP Data Type	Data Foundation Data Type
INTEGER	INTEGER
NUMERIC TEXT, PACKED NUMBER	NUMERIC
TEXT	CHAR
VARIABLE LENGTH STRING	VARCHAR

Date Format in SQL Expressions

The date format that the application user must use when building queries in the SQL Expression Editor is DATE'yyyy-mm-dd', for example: DATE'2013-04-10'.

The format is specified in the `jco.prm` extended PRM file with the `USER_INPUT_DATE_FORMAT` parameter. See the *Information Design Tool User Guide*.

Related Topics

- About PRM Files

14.1.8 SAP HANA Data Types

The following table lists the data types used in the SAP HANA database and their equivalent in data foundations. For more information on SAP HANA data types, see the SAP HANA Data Type Reference.

SAP HANA Data Type	Data Foundation Data Type
BLOB	LONGVARBINARY
VARBINARY	VARBINARY
DATE	DATE
TIME	TIME

SAP HANA Data Type	Data Foundation Data Type
LONGDATE, SECONDDATE, TIMESTAMP	TIMESTAMP **Note:** SECONDDATE is converted into TIMESTAMP without milliseconds.
DECIMAL, REAL, SMALLDECIMAL	FLOAT
BIGINT, INTEGER, SMALLINT, TINYINT	INTEGER
DOUBLE	DOUBLE
CLOB, NCLOB, TEXT	LONGVARCHAR
ALPHANUM, NVARCHAR, SHORTTEXT, VARCHAR	VARCHAR

Related Topics

- SAP HANA Data Type Reference

14.1.9 Sybase CTL Data Types

The following table lists the data types that show up through Sybase CTL and their equivalent in data foundations.

Note:
Sybase CTL data types are values of `datatype` field of `CS_DATAFMT` structure, which is actually exposed by the network layer. The `usertype` field value of the structure is also provided for some data types and used for mapping by the Data Access layer.

Sybase CTL datatype	Sybase CTL usertype	Data foundation Data Type
CS_BINARY_TYPE	CS_BINARY_USERTYPE	BINARY

Sybase CTL datatype	Sybase CTL usertype	Data foundation Data Type
CS_BINARY_TYPE	CS_VARBINARY_USERTYPE or no value	VARBINARY
CS_LONGBINARY_TYPE	Any value	VARBINARY
CS_IMAGE_TYPE	Any value	LONGVARBINARY
CS_BIT_TYPE	Any value	BIT
CS_DATE_TYPE	Any value	DATE
CS_BIGDATETIME_TYPE, CS_BIGTIME_TYPE, CS_DATETIME_TYPE, CS_DATETIME4_TYPE, CS_TIME_TYPE	Any value	TIMESTAMP
CS_LONGCHAR_TYPE, CS_TEXT_TYPE	Any value	LONGVARCHAR
CS_BIGINT_TYPE, CS_UBIG-INT_TYPE	Any value	BIGINT
CS_DECIMAL_TYPE, CS_MONEY_TYPE, CS_MON-EY4_TYPE	Any value	DECIMAL
CS_FLOAT_TYPE	Any value	DOUBLE
CS_INT_TYPE , CS_UINT_TYPE	Any value	INTEGER
CS_NUMERIC_TYPE	Any value	NUMERIC
CS_REAL_TYPE	Any value	REAL

Sybase CTL datatype	Sybase CTL usertype	Data foundation Data Type
CS_SMALLINT_TYPE, CS_USMALLINT_TYPE	Any value	SMALLINT
CS_TINYINT_TYPE	Any value	TINYINT
CS_CHAR_TYPE	CS_CHAR_USERTYPE or CS_NCHAR_USERTYPE or CS_UNICHAR_USERTYPE	CHAR
CS_CHAR_TYPE	CS_VARCHAR_USERTYPE or CS_NVARCHAR_USERTYPE or CS_UNIVARCHAR_USERTYPE or CS_SYSNAME_USERTYPE or no value	VARCHAR
CS_UNICHAR_TYPE	USER_UNICHAR_TYPE	CHAR
CS_UNICHAR_TYPE	USER_UNIVARCHAR_TYPE or no value	VARCHAR

14.1.10 XML Data Types

The following table lists the XML built-in data types and their equivalent in data foundations.

XML Built-in Data Type	Data Foundation Data Type
boolean	BIT
base64Binary, hexBinary	• BINARY if a length is defined • VARBINARY
date	DATE

XML Built-in Data Type	Data Foundation Data Type
time	TIME
dateTime	TIMESTAMP
float	FLOAT
double, decimal	DOUBLE
integer, negativeInteger, nonPositiveInteger, noNegativeInteger, positiveInteger	DECIMAL
long, unsignedLong	BIGINT
int, unsignedInt	INTEGER
short, unsignedShort	SMALLINT
byte, unsignedByte	TINYINT
string	• CHAR if a length is defined • VARCHAR **Note:** Data types derived from string are mapped to CHAR or VARCHAR.
anySimpleType, anyType, anyURI, duration, gDay, gMonth, gMonthDay, gYear, gYearMonth, NOTATION, QName	VARCHAR

Mapping of Column Medatata

- Signed or unsigned is determined from the data type.
- Nullable is determined from the `nullable` and `minOccurs` attributes.
- Column size, decimal digits, and display size are determined from the data type. If the type does not give any indication, a user-configurable maximum length is returned.

14.2 Large Variable Length Data Restriction

Restriction:
The following only relates to Crystal Reports applications.

Due to the Data Access layer implementation, the maximum size of a column for large variable length binary and character data is limited to 16MB for the following data sources:

- IBM DB2 through ODBC
- MS SQL Server through ODBC
- Sybase CTL

If an application user inserts large variable length data larger than 16MB into a report, the system performance may be affected.

14.3 Data Type Mapping for Multisource-Enabled Universes

In the case of a multisource-enabled data foundation, the Connection Server data types are converted into the data types exposed through the Data Federation Service.

Data Federation data types are the following:

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

They are mapped to the corresponding data foundation data types.

Caution:
BINARY, VARBINARY, and LONGVARBINARY types are not supported and values of these types are mapped to NULL.

Index

_trace.ini file 157

64-bit support
  about 27
  Microsoft Windows support 28
  UNIX environment 27

A

ABAP functions 92
about the guide 17
ActiveDataSources setting 155
Adaptive Connectivity service 24
Amazon EMR HIVE connections 118
ANALYTIC_CLAUSE 225
ANALYTIC_FUNCTIONS 226
Apache Hadoop HIVE connections 115
  to install the driver 117
architecture 20
Array Bind Available 179
Array Bind Size 180
Array Fetch Available 180
Array Fetch Size 95, 181
audience of the guide 17

B

backward compatibility 38, 71
BigDecimal Max Display Size 182
Binary Max Length 87, 107, 113, 182
branded drivers 173
Bucket Split Size 183

C

CALCULATION_FUNCTION 226
Catalog Separator 184
CharSet 208
Charset List Extension 149
CharSet Table 184
ClassPath
  creating a JavaBean connection 63
  creating a JDBC connection 55
CMC 148
Config File Extension 149
configuration files
  about 21
  driver-specific files 22
  global file 22
configuration files *(continued)*
  installed SBO files 166
  JavaBean SBO example 64
Connection Check 172
connection pools
  about 34
  about connections of 35
Connection Server
  64-bit support 27
  about global settings 147
  choosing the deployment mode 154
  configuring data access for DataDirect branded drivers 174
  configuring deployment mode 155
  configuring for complex deployment 155
  configuring for ODBC connection to Data Federator 141
  configuring global settings 148
  defining protocols in server mode 156
  definition 19
  deployment mode 24
  memory 38
  reading logs and traces 162
  stored procedures 31
  tracing in library mode 161
  tracing in server mode 162
  workflow activity 39
Connection Shareable 202
Connection Status Available 209
connections
  components of 19
  configuring JDBC connections 173
  creating a JavaBean connection 63
  creating Data Federator XI 3.0 139
  creating generic JDBC connections 57
  creating generic ODBC connections 66
  creating generic ODBC3 connections 67
  creating JavaBean connections using Extensions 64
  creating JDBC connections using Extensions 56
  creating JDBC connections using SBO files 55
  requirements for creating 41
Connectivity services 24
connectors
  SAP NetWeaver BW 135
  SAS 137
CONSTANT_SAMPLING_SUPPORTED 227
conventions 18
converting data types 243
Cost Estimate Available 209
COUNT function 86
Crystal Reports 260
cs.cfg file 160
  about 22, 147
  Capabilities section 154
  editing 148
  Library section 155
  settings parameters reference 148
  viewing 148
cscheck
  accessdriver 48
  connectivity 50
  driverssearch 45
  find 46
  function list 43
  getting help 42
  list 44
  middleware 47
  ping 51
  running 43
  syntax 43
CSV files 71
  data types 244

D

data access drivers
  about 20
  checking availability on a machine 48
  configuring parameters 165
  CSV files 71
  files used by 22
  PRM files 219
  SAP ERP systems 88
  searching for installed 45
  selecting 155
  setup files 165
data access guide
  about 17
  audience 17
  conventions 18

data access guide *(continued)*
key tasks 17
Data Federator XI 3.0
about creating connections 139
installation directory 140
required version 139
data types
CSV files 244
JDBC 245
OData 250
ODBC 248
OLE DB 251
Oracle OCI 253
SAP ERP 254
Sybase 256
web services 258
XML 258
databases
capability parameters 219
checking access 51
configuring access 155
listing supported 44
selecting access to 156
supporting stored procedures 31
DataDirect 173
DDL files 74
deployment mode 24, 155
Description Extension 150
Description File 184
Dictionary Transaction Mode 185
DISTINCT 227
Driver Behavior 214
Driver Capabilities 185
Driver Name 186
DriverDefaults 147
dsn entry 141

E

Empty String 209
Enforce Max Protocol Version 207
Enumerator CLSID 212
ERP Max Rows 95, 201
Escape Character 186
Escape Character Available 203
Essbase connections 124
EXT_JOIN 228
Extensions 187
creating a JavaBean connection 64
creating a JDBC connection 56

F

Failover 122, 128
Family 187
Force Execute 188
ForeignKeys Available 203
FULL_EXT_JOIN 228

G

generic JDBC data source 57
generic ODBC data source 28, 66
generic ODBC3 data source 67
Get Extended Column 204
GROUP_BY 229
GROUP_BY_SUPPORTS_COLUMN_INDEX 229
GROUP_BY_SUPPORTS_COMPLEX 230
GROUP_BY_SUPPORTS_CONSTANT 230

H

HAVING 231
HEALTH 37
help
cscheck 42
PRM files 223
HOARD 38
disabling 39
enabling 38

I

IBM DB2
GROUP_BY_SUPPORTS_COMPLEX 230
GROUP_BY_SUPPORTS_CONSTANT 230
SELECT_SUPPORTS_NULL 239
IBM DB2 connections 119
IBM Informix
SELECT_SUPPORTS_NULL 239
SQLDescribeParam Available 210
V5toV6DriverName 212
IBM Informix connections 119
IBM Red Brick
ANALYTIC_CLAUSE 225
LEFT_OUTER 233
RIGHT_OUTER 238
SELECT_SUPPORTS_NULL 239
Identifier Case 188
Identifier Quote String 189
Ignore Driver Load Failure 150
Include Synonyms 189
information design tool 29
InfoSets 90
input columns 90
installation
checking data access driver 48
Data Federator directory 140
files installed 21
Integer Max Length 107, 113, 190
INTERSECT 231
INTERSECT_ALL 232
INTERSECT_IN_SUBQUERY 232
Introscope 39
Introscope Available 190

J

Java drivers 54
JavaBean
about connections 62
JavaBean Class 201
URL Format 199
JavaBean Class 201
JavaBean connections
creating 63
creating with Extensions 64
JavaBean SBO example 64
javabean.sbo 63
JDBC
about connections 54
Connection Shareable 202
creating connections with Extensions 56
creating connections with SBO files 55
data types 245
Escape Character Available 203
ForeignKeys Available 203
generic 57
Include Synonyms 189
JDBC Class 204
JDBC ResultSet Concurrency 207
JDBC ResultSet Type 206
PrimaryKey Available 204
Shared Connection 205
URL Format 199
JDBC Class 204
JDBC connections
adding driver properties 173
JDBC ResultSet Concurrency 207
JDBC ResultSet Type 206
jdbc.sbo 57
JNI usage 154
JOIN 233

K

key tasks 17

L

LEFT_EXT_JOIN 233
LEFT_OUTER 233
library mode 155, 156, 161
LIKE_SUPPORTS_ESCAPE_CLAUSE 234
load balancing 36
Load Drivers On Startup 150
Locales 147
logs 157, 163

M

macros 34
Max Pool Time 151
Max Rows 86, 95
Max Rows Available 191
memory allocation 38
memory usage 37
Microsoft Azure data sources 88
Microsoft Windows
  64-bit 28
middleware
  checking availability on a machine 47
  checking connectivity with network layer 50
  listing supported 46
MINUS 234
MINUS_ALL 235
MINUS_IN_SUBQUERY 235
MS Access
  DISCTINCT 227
MS SQL Server 173
MS SQL Server Connections 121
MSOlap CLSID 213
multisource-enabled universes 260

N

native connections
  Locale setting 68
Native Connectivity service 24
Native Int64 Available 191
Neoview
  CharSet 208
network layers
  checking connectivity with middleware 50
  listing supported 44
  selecting 155

O

OData connections 76, 86
  data types 250
  driver capabilities 77
  HTTPS 77
  performance 84
  schema 78
  tracing 88
ODBC
  about connections 65
  Connection Status Available 209
  DataDirect drivers 173
  Empty String 209
  exposing database layer in server mode 156
  generic ODBC2 66
  generic ODBC3 67
  ODBC Cursors 210
  SQLMoreResults Available 211
  Use DataDirect OEM Driver 211
ODBC Cursors 210
ODBC data types 248
ODBC drivers 29
OLAP connections 124, 127
OLAP Connections 26
OlapClient.cfg file 163
  about 24
OLE DB
  data types 251
  Enumerator CLSID 212
  Provider CLSID 213
OLE DB OLAP
  MSOlap CLSID 213
openrda.ini file 141
Optimize Execute 192
Oracle
  connectivity check example 51
  data types 253
  Identifier Quote String 189
  Include Synonyms 189
  LEFT_OUTER 233
  Optimize Execute 192
  RIGHT_OUTER 238
  stored procedures 32
Oracle EBS connections 123
Oracle JDBC connections 122
Oracle RAC connections 125
ORDER_BY 236
ORDER_BY_REQUIRES_SELECT 236
ORDER_BY_SUPPORTS_COLUMN_INDEX 236
Owners Available 192

P

Password Encryption 214
PERCENT_RANK_SUPPORTED 237
performance 84
PrimaryKey Available 204
PRM files
  about 219
  about function help text 223
  editing 221
  editing help text 223
  structure 220
  viewing 221
Provider CLSID 213
PVL 88, 128
PVL Available 205

Q

Qualifiers Available 193
Query TimeOut Available 193
Quote Identifiers 194
Quoted Identifier 215

R

RANK_SUPPORTED 237
reading logs and traces 162
Recover Errors 215
Replace Current Owner With Database 216
requirements
  connections 41
  Data Federator minimum version 139
RIGHT_EXT_JOIN 238
RIGHT_OUTER 238

S

salesforce.com connections 125
SAML 29, 128
sample drivers 20
SAP BW connections 127
SAP ERP connections 88, 128, 201
  data types 254
SAP HANA connections 128, 194
  stored procedures 33
SAP MaxDB connections 135
SAP NetWeaver BW connections 135
SAP NetWeaver Gateway 76, 86
SAP Queries 90
SAS connections 137

SBO files
configuring a JavaBean connection using Extensions 64
configuring a JDBC connection using ClassPath 55
configuring a JDBC connection using Extensions 56
Connection Check 172
customizing 171, 172
Databases parameters 177
Defaults parameters 177
editing 171
installed 166
structure 177
viewing 171
SEED_SAMPLING_SUPPORTED 239
SELECT_SUPPORTS_NULL 239
server mode 155, 162
Settings 147
Setup File Extension 152
setup files 165
Shared Connection 205
single sign-on 29, 131, 133
Skip SAML SSO 194
SMB 74
SQL External Extension 152
SQL External File 195
SQL Parameter Extension 152
SQL Parameter File 195
SQL Server
GROUP_BY_SUPPORTS_CONSTANT 230
LEFT_OUTER 233
RIGHT_OUTER 238
SQL Server Analysis Services connections 120
SQLDDL files 74
SQLDescribeParam Available 210
SQLMoreResults Available 211
SSL 128
SSO
configuring for the information design tool 131
SSO *(continued)*
configuring for Web Intelligence 133
configuring for Web Intelligence Rich Client 133
data sources supported 29
SSO Available 195
stored procedures
about 31
about JavaBean connections 62
creating Oracle 32
limitations 31
Oracle 32
SAP HANA 33
Teradata 34
Strategies Extension 153
Strategies File 196
String Max Length 87, 107, 113, 196
SUBQUERY_IN_FROM 240
SUBQUERY_IN_IN 240
SUBQUERY_IN_WHERE 240
supported connectivities 46
supported databases 44
supported network layers 44
Sybase
connectivity check example 51
data types 256
Driver Behavior 214
LEFT_OUTER 233
Password Encryption 214
Quoted Identifier 215
Recover Errors 215
RIGHT_OUTER 238
system architecture 20

T

TECHNICAL_COLUMN_NAME_PATTERN 241
Temp Data Dir 197
Teradata
ANALYTIC_CLAUSE 225
Cost Estimate Available 209
Teradata *(continued)*
macros 34
SELECT_SUPPORTS_NULL 239
stored procedures 34
Text Size 215
traces 157, 163
Traces 147
Transactional Available 120, 126, 198
Type 199

U

Unicode 199
UNION 241
UNION_ALL 242
UNION_IN_SUBQUERY 242
universe design tool 29
UNIX
64-bit 27
unv universes 28
URL Format 199
Use DataDirect OEM Driver 211

V

V5toV6DriverName 212
Validate Configuration Files 153
Validate XML Streams 154
validating connections at runtime 172

W

Web Service connections 107
web service data types 258
workflow activity 39

X

XI 3.x releases 54
XML connections 95
XML data types 258
XML Max Size 200