# Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 1</td>
<td>13</td>
</tr>
<tr>
<td>Chapter 2</td>
<td>17</td>
</tr>
<tr>
<td>2.1</td>
<td>17</td>
</tr>
<tr>
<td>2.2</td>
<td>18</td>
</tr>
<tr>
<td>2.3</td>
<td>21</td>
</tr>
<tr>
<td>2.4</td>
<td>21</td>
</tr>
<tr>
<td>2.4.1</td>
<td>23</td>
</tr>
<tr>
<td>2.5</td>
<td>23</td>
</tr>
<tr>
<td>2.5.1</td>
<td>25</td>
</tr>
<tr>
<td>2.5.2</td>
<td>25</td>
</tr>
<tr>
<td>2.5.3</td>
<td>26</td>
</tr>
<tr>
<td>2.5.4</td>
<td>27</td>
</tr>
<tr>
<td>2.5.5</td>
<td>27</td>
</tr>
<tr>
<td>2.5.6</td>
<td>28</td>
</tr>
<tr>
<td>2.6</td>
<td>28</td>
</tr>
<tr>
<td>Chapter 3</td>
<td>31</td>
</tr>
<tr>
<td>3.1</td>
<td>31</td>
</tr>
<tr>
<td>3.2</td>
<td>34</td>
</tr>
<tr>
<td>3.3</td>
<td>39</td>
</tr>
<tr>
<td>3.4</td>
<td>39</td>
</tr>
<tr>
<td>3.5</td>
<td>42</td>
</tr>
<tr>
<td>3.6</td>
<td>43</td>
</tr>
<tr>
<td>3.7</td>
<td>44</td>
</tr>
<tr>
<td>3.7.1</td>
<td>45</td>
</tr>
<tr>
<td>Chapter 4</td>
<td>47</td>
</tr>
<tr>
<td>4.1</td>
<td>47</td>
</tr>
<tr>
<td>4.2</td>
<td>47</td>
</tr>
<tr>
<td>4.3</td>
<td>49</td>
</tr>
<tr>
<td>4.4</td>
<td>54</td>
</tr>
</tbody>
</table>

**Chapter 1**

- **Document History**

**Chapter 2**

- **Getting started with the information design tool**
  - About the information design tool
  - About resources in the information design tool
  - Starting the information design tool
  - About the information design tool interface
  - Resetting the user interface display
  - Setting preferences in the information design tool
  - Setting preferences for check integrity
  - Setting preferences for the Data Foundation Editor
  - Setting languages used by the information design tool
  - Setting a link for online tutorials
  - Setting middleware for secured relational connections
  - Setting preferences for showing values
  - How to get help using the information design tool

**Chapter 3**

- **Creating Universes**
  - How to create a universe with the information design tool
  - Universes on SAP NetWeaver BW
  - Universes on SAP ERP
  - Universes on Microsoft Analysis Services (MSAS)
  - Universes on Essbase
  - Universes on SAS
  - Multilingual universes
  - Translating universe metadata

**Chapter 4**

- **Converting .unv universes**
  - About .unv and .unx universes
  - About converting .unv universes
  - Features supported when converting .unv universes
  - Tips for resolving check integrity errors after converting .unv universes
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.8.1</td>
<td>Creating and editing a business layer view</td>
<td>203</td>
</tr>
<tr>
<td>10.8.2</td>
<td>Filtering by business layer view</td>
<td>204</td>
</tr>
<tr>
<td>10.9</td>
<td>About parameters</td>
<td>206</td>
</tr>
<tr>
<td>10.9.1</td>
<td>Inserting and editing a parameter</td>
<td>206</td>
</tr>
<tr>
<td>10.10</td>
<td>About lists of values</td>
<td>207</td>
</tr>
<tr>
<td>10.10.1</td>
<td>Inserting or editing a list of values</td>
<td>210</td>
</tr>
<tr>
<td>10.10.2</td>
<td>List of values column properties</td>
<td>210</td>
</tr>
<tr>
<td>10.10.3</td>
<td>Associating a list of values with a business object</td>
<td>210</td>
</tr>
<tr>
<td>10.10.4</td>
<td>Associating a list of values with a prompt defined in the business layer</td>
<td>211</td>
</tr>
<tr>
<td>10.11</td>
<td>About navigation paths for objects</td>
<td>212</td>
</tr>
<tr>
<td>10.11.1</td>
<td>Inserting a navigation path object into a business layer</td>
<td>213</td>
</tr>
<tr>
<td>10.11.2</td>
<td>About queries in a business layer</td>
<td>213</td>
</tr>
<tr>
<td>10.12</td>
<td>About aggregate awareness</td>
<td>214</td>
</tr>
<tr>
<td>10.13</td>
<td>Setting up aggregate awareness</td>
<td>214</td>
</tr>
<tr>
<td>10.13.1</td>
<td>Setting aggregate navigation</td>
<td>215</td>
</tr>
<tr>
<td>10.14</td>
<td>About refreshing business layers</td>
<td>216</td>
</tr>
<tr>
<td>10.14.1</td>
<td>Refreshing an OLAP business layer</td>
<td>217</td>
</tr>
<tr>
<td>10.14.2</td>
<td>Inserting candidate objects</td>
<td>217</td>
</tr>
<tr>
<td>10.15</td>
<td>About computing statistics for optimized query execution</td>
<td>218</td>
</tr>
<tr>
<td>10.15.1</td>
<td>Computing statistics for a multisource-enabled universe</td>
<td>218</td>
</tr>
<tr>
<td>11.1</td>
<td>How to build a query</td>
<td>221</td>
</tr>
<tr>
<td>11.2</td>
<td>About the Member Selector</td>
<td>222</td>
</tr>
<tr>
<td>11.2.1</td>
<td>About selecting hierarchy members</td>
<td>223</td>
</tr>
<tr>
<td>11.2.2</td>
<td>Opening the Member Selector in the Query Panel</td>
<td>224</td>
</tr>
<tr>
<td>11.2.3</td>
<td>Selecting hierarchy members</td>
<td>224</td>
</tr>
<tr>
<td>11.2.4</td>
<td>Selecting members by hierarchy relationship</td>
<td>225</td>
</tr>
<tr>
<td>11.2.5</td>
<td>Selecting hierarchy members by level</td>
<td>226</td>
</tr>
<tr>
<td>11.2.6</td>
<td>Selecting named sets</td>
<td>227</td>
</tr>
<tr>
<td>11.2.7</td>
<td>Selecting calculated members</td>
<td>227</td>
</tr>
<tr>
<td>11.2.8</td>
<td>Searching for hierarchy members</td>
<td>227</td>
</tr>
<tr>
<td>11.2.9</td>
<td>Excluding hierarchy members</td>
<td>228</td>
</tr>
<tr>
<td>11.2.10</td>
<td>Defining a prompt to select members</td>
<td>229</td>
</tr>
<tr>
<td>11.2.11</td>
<td>Showing selected members in the Member Selector</td>
<td>229</td>
</tr>
<tr>
<td>11.2.12</td>
<td>Sorting hierarchy members</td>
<td>230</td>
</tr>
<tr>
<td>11.2.13</td>
<td>Setting display options</td>
<td>230</td>
</tr>
<tr>
<td>11.2.14</td>
<td>Showing estimated child count</td>
<td>230</td>
</tr>
<tr>
<td>11.3</td>
<td>Filtering data in the Query Panel</td>
<td>231</td>
</tr>
<tr>
<td>11.3.1</td>
<td>How to build a business filter</td>
<td>231</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>11.3.2</td>
<td>Filtering data using prompts</td>
<td>233</td>
</tr>
<tr>
<td>11.4</td>
<td>Setting query properties</td>
<td>235</td>
</tr>
<tr>
<td>11.5</td>
<td>Viewing and editing the query script</td>
<td>236</td>
</tr>
<tr>
<td>11.6</td>
<td>Profiling column values in the query panel</td>
<td>237</td>
</tr>
<tr>
<td>Chapter 12</td>
<td>Checking integrity</td>
<td>239</td>
</tr>
<tr>
<td>12.1</td>
<td>Running check integrity</td>
<td>239</td>
</tr>
<tr>
<td>12.2</td>
<td>Reviewing check integrity problems</td>
<td>240</td>
</tr>
<tr>
<td>Chapter 13</td>
<td>Showing and profiling values in a data source</td>
<td>241</td>
</tr>
<tr>
<td>Chapter 14</td>
<td>Showing dependencies between resources</td>
<td>243</td>
</tr>
<tr>
<td>14.1</td>
<td>About resource dependencies</td>
<td>243</td>
</tr>
<tr>
<td>14.2</td>
<td>Showing local dependencies</td>
<td>245</td>
</tr>
<tr>
<td>14.3</td>
<td>Showing repository dependencies</td>
<td>246</td>
</tr>
<tr>
<td>Chapter 15</td>
<td>Publishing resources</td>
<td>247</td>
</tr>
<tr>
<td>15.1</td>
<td>About publishing resources</td>
<td>247</td>
</tr>
<tr>
<td>15.2</td>
<td>Publishing a universe</td>
<td>248</td>
</tr>
<tr>
<td>15.2.1</td>
<td>Selecting a repository folder</td>
<td>249</td>
</tr>
<tr>
<td>15.2.2</td>
<td>Selecting a local folder</td>
<td>249</td>
</tr>
<tr>
<td>15.3</td>
<td>Publishing a local connection to the repository</td>
<td>249</td>
</tr>
<tr>
<td>Chapter 16</td>
<td>Managing security</td>
<td>251</td>
</tr>
<tr>
<td>16.1</td>
<td>About universe security</td>
<td>251</td>
</tr>
<tr>
<td>16.2</td>
<td>About securing resources in the information design tool</td>
<td>252</td>
</tr>
<tr>
<td>16.3</td>
<td>CMC rights for information design tool users</td>
<td>253</td>
</tr>
<tr>
<td>16.4</td>
<td>About the Security Editor</td>
<td>255</td>
</tr>
<tr>
<td>16.5</td>
<td>How to secure a universe using security profiles</td>
<td>256</td>
</tr>
<tr>
<td>16.6</td>
<td>Opening the Security Editor</td>
<td>258</td>
</tr>
<tr>
<td>16.7</td>
<td>Inserting and editing a Data Security Profile</td>
<td>258</td>
</tr>
<tr>
<td>16.7.1</td>
<td>Data Security Profile settings</td>
<td>259</td>
</tr>
<tr>
<td>16.7.2</td>
<td>Data Security Profile Connections settings</td>
<td>260</td>
</tr>
<tr>
<td>16.7.3</td>
<td>Data Security Profile Controls settings</td>
<td>260</td>
</tr>
<tr>
<td>16.7.4</td>
<td>Data Security Profile SQL settings</td>
<td>261</td>
</tr>
<tr>
<td>16.7.5</td>
<td>Data Security Profile Rows settings</td>
<td>262</td>
</tr>
<tr>
<td>16.7.6</td>
<td>Data Security Profile Tables settings</td>
<td>263</td>
</tr>
<tr>
<td>16.8</td>
<td>Changing Data Security Profile priority</td>
<td>263</td>
</tr>
<tr>
<td>16.9</td>
<td>Inserting and editing a Business Security Profile</td>
<td>264</td>
</tr>
<tr>
<td>Appendix A</td>
<td>More Information</td>
<td>325</td>
</tr>
<tr>
<td>------------</td>
<td>------------------</td>
<td>-----</td>
</tr>
<tr>
<td>Index</td>
<td></td>
<td>327</td>
</tr>
</tbody>
</table>

Index

More Information
The following table summarizes the documentation added for what is new in the information design tool for each version of SAP BusinessObjects BI Suite.
<table>
<thead>
<tr>
<th>Version</th>
<th>New in this version</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.0 Support Package 2</td>
<td>Data foundation commands not supported when the Search Panel is open are now grayed. See <a href="#">Searching for tables and columns in the data foundation</a>.</td>
</tr>
<tr>
<td>March, 2011</td>
<td>Procedure to recover local projects after workspace corruption, see <a href="#">Recovering local projects after workspace corruption</a>.</td>
</tr>
<tr>
<td>4.0 Feature Package 3</td>
<td></td>
</tr>
<tr>
<td>March, 2012</td>
<td></td>
</tr>
<tr>
<td>Version</td>
<td>New in this version</td>
</tr>
<tr>
<td>---------</td>
<td>---------------------</td>
</tr>
<tr>
<td></td>
<td>SAP ERP relational connections, see Universes on SAP ERP.</td>
</tr>
<tr>
<td></td>
<td>Essbase OLAP connections, see Universes on Essbase.</td>
</tr>
<tr>
<td></td>
<td>Parameters for CSV file connections; see Login and schema parameters for CSV file connections.</td>
</tr>
<tr>
<td></td>
<td>Including User Attributes in object definitions with @Variable, see About @Variable.</td>
</tr>
<tr>
<td></td>
<td>Using @Variable in Query Panel filters, see How to build a business filter.</td>
</tr>
<tr>
<td></td>
<td>New K data type for @Prompt function to handle unquoted strings for prompts, see About @Prompt.</td>
</tr>
<tr>
<td></td>
<td>Editing input columns, see About input columns.</td>
</tr>
<tr>
<td></td>
<td>Table and column name delimitation, see About tables in the data foundation.</td>
</tr>
<tr>
<td></td>
<td>Selecting an OLAP hierarchy for a dimension, see Inserting and editing dimensions.</td>
</tr>
<tr>
<td></td>
<td>Setting the aggregation function for measures, see Turning a dimension into a measure.</td>
</tr>
<tr>
<td></td>
<td>Searching for tables in a connection, see Searching for tables in the Connection pane.</td>
</tr>
<tr>
<td></td>
<td>Filtering the list of tables in a connection, see Filtering the tables in the connection by table type.</td>
</tr>
<tr>
<td></td>
<td>New properties pane in the data foundation editor, see About the Data Foundation Editor.</td>
</tr>
<tr>
<td></td>
<td>Selecting hierarchy members up to a level in the Query Panel, see Selecting hierarchy members.</td>
</tr>
<tr>
<td></td>
<td>A cheat sheet to help you create your first universe, see How to get help using the information design tool.</td>
</tr>
<tr>
<td></td>
<td>A link to the information design tool online tutorials, see How to get help using the information design tool.</td>
</tr>
<tr>
<td></td>
<td>Using the local middleware driver for secured connections, see About secured connections.</td>
</tr>
</tbody>
</table>
Getting started with the information design tool

2.1 About the information design tool

The information design tool is an SAP BusinessObjects metadata design environment that enables a designer to extract, define, and manipulate metadata from relational and OLAP sources to create and deploy SAP BusinessObjects universes.

A universe is an organized collection of metadata objects that enable business users to analyze and report on corporate data in a non-technical language. These objects include dimensions, measures, hierarchies, attributes, pre-defined calculations, functions, and queries. The metadata object layer, called the business layer, is built on a relational database schema or an OLAP cube, so the objects map directly to the database structures via SQL or MDX expressions. A universe includes connections identifying the data sources so queries can be run on the data.

The role of the universe is to provide the business user with semantically understandable business objects. The user is free to analyze data and create reports using relevant business language regardless of the underlying data sources and structures.

Universes created using the information design tool can be used by the following SAP BusinessObjects data analysis and reporting applications starting with version BI 4:

- SAP BusinessObjects Web Intelligence
- SAP Crystal Reports for Enterprise
- SAP BusinessObjects Explorer
- SAP BusinessObjects Dashboard Design

To enable the designer to create universes, the information design tool provides the resources necessary to do the following:

- Create connections to data sources.
- Extract a complete OLAP cube schema.
- Extract tables and joins to build a relational schema called a data foundation.
- Create metadata objects from the cube or the data foundation. These objects are contained and organized in a business layer. The SQL and MDX expressions within objects can be validated and queries run against the target databases to test the business layer.
- Share resources to allow multiple designers to work on the same resources concurrently.
- Publish a universe, which compiles the business layer, the data foundation, and the connections into a single universe file (.unx):
  - Publish a universe to a repository to be implemented in deployments of SAP BusinessObjects data analysis and reporting applications.
• Publish a universe locally, to be implemented by client applications in standalone mode (for example Web Intelligence Rich Client).
• Create security profiles to define user access to universe data and metadata.

Who uses the information design tool?
The universe designer may be a database administrator, an applications manager or developer, a project manager, or a report creator who has acquired enough technical skills to create universes for other users. A security administrator also uses the information design tool to define universe security profiles.

There can be more than one universe designer in a company. The number of universe designers depends on the company’s data requirements. For example, one universe designer could be appointed for each application, project, department or functional area.

Related Topics
• About resources in the information design tool
• Starting the information design tool
• How to create a universe with the information design tool

2.2 About resources in the information design tool

The information design tool provides the following design resources to extract metadata and build universes.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
</table>
| Project | A project is a named local workspace that contains the resources used to build one or more universes. A project can be shared so that multiple designers can work on the same resources.

A project can contain any number of independent resources, for example data foundations, business layers, and connections. All resources contained within a project can be used interchangeably, for example a connection can be used by several data foundations within the same project.

Projects and their resources are displayed in the Local Projects View. To open a resource in the editor, double-click the resource in the Local Projects View. |
<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Connection</strong></td>
<td>A connection is a named set of parameters that define how a universe can access a relational or OLAP data source. A universe is always associated with at least one connection. A connection is an independent resource and can be used by several universes. You can build a multisource-enabled universe that references one or more relational connections. Connections can be local (stored in a local file) or secured (an object in a shared repository that is referenced by a connection shortcut). Local connections are stored in the local project as .cnx files.</td>
</tr>
<tr>
<td><strong>OLAP</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Relational</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Connection shortcut</strong></td>
<td>A connection shortcut is an object in the local project that references a secured connection in a repository. You use a connection shortcut to refer to secure connections when creating data foundations and business layers based on secure connections. Connection shortcuts are stored in the local project as .cns files.</td>
</tr>
<tr>
<td><strong>Data foundation</strong></td>
<td>A data foundation is a schema that defines the relevant tables and joins from one or more relational databases. You enhance the data foundation by adding derived tables, alias tables, calculated columns, additional joins, contexts, prompts, lists of values, and other SQL definitions. The data foundation becomes the basis of one or more business layers. Data foundations are stored in the local project as .dfx files.</td>
</tr>
<tr>
<td><strong>Business layer</strong></td>
<td>A business layer is a collection of metadata objects that provides an abstraction of relational database entities or OLAP cubes, understandable by a business user. Objects map via SQL expressions to an underlying data foundation, or via MDX expressions to an underlying OLAP cube. These objects include dimensions, hierarchies, measures, attributes, and predefined conditions. You can add dimensions, hierarchies, measures, attributes, and other objects as the universe design requires. You can validate the SQL or the MDX at any time. You can create queries, lists of values, parameters (also called prompts), and navigation path objects. The business layer is the universe under construction, and when the business layer is complete, it is compiled with the connections or connection shortcuts and data foundation, published, and deployed as a universe. Business layers are stored in the local project as .blx files.</td>
</tr>
</tbody>
</table>
A query is a set of objects that define a request to the database for data. A query can be defined and saved in the business layer as a metadata object to be used to test objects in the business layer.

A parameter is a variable in the universe that requires a value at query time. Parameters are often defined to prompt the user to supply a value, and in this case are referred to as prompts.

A list of values is a collection of data values that can be associated with an object in the universe, allowing the user to choose values for a prompt.

Parameters and lists of values can be defined in the data foundation. They are inherited by all business layers based on that data foundation.

Parameters and lists of values can also be defined in the business layer.

A universe is a compiled file that includes all resources used in the definition of the metadata objects built in the design of the business layer.

The universe is used by SAP BusinessObjects data analysis and reporting applications, where the business layer objects are visible for analysis and reporting.

Universes are stored either locally or in a repository as .unx files.

A security profile is a group of security settings that controls the data and metadata that are displayed to users and modifies the parameters defined in the data foundation and/or business layer. Security profiles are defined on published universes and stored in the repository.

**Related Topics**

- [How to create a universe with the information design tool](#)
- [About local projects](#)
- [About connections](#)
- [About data foundations](#)
- [About business layers](#)
- [About queries in a business layer](#)
- [About parameters](#)
- [About lists of values](#)
- [About universe security](#)
2.3 Starting the information design tool

The information design tool is installed with the SAP BusinessObjects Business Intelligence platform Client Tools. For more information on installing the BI platform Client Tools, see the SAP BusinessObjects Business Intelligence Suite 4.0 Master Guide, or the SAP Crystal Server 2011 Getting Started Guide.

Once the client tools are installed on your machine, for example in a Windows installation of the BI platform, you can start the information design tool with the command: Start > All programs > SAP BusinessObjects BI platform 4 > SAP BusinessObjects BI platform Client Tools > Information Design Tool.

No authentication is required to use the information design tool in offline mode (not connected to a repository). You can begin creating and editing local resources.

Note:
When you double-click a resource file in the local file system (for example a .blx, .dfx, or .cnx file), the information design tool opens without opening the specific resource editor. You must open the editor from the Local Projects View.

Related Topics
• About resources in the information design tool
• How to create a universe with the information design tool
• About securing resources in the information design tool

2.4 About the information design tool interface

The information design tool interface is composed of views and editors that let you navigate and work on different resources. For more information about each view, see the Related Topics.

<table>
<thead>
<tr>
<th>View</th>
<th>Description</th>
<th>How to open the view</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local Projects View</td>
<td>Use this view to create and navigate local projects, and to open and validate resources.</td>
<td>Select Window &gt; Local Projects.</td>
</tr>
<tr>
<td>Repository Resources View</td>
<td>Use this view to navigate repository resources and create secured connections.</td>
<td>Select Window &gt; Repository Resources.</td>
</tr>
<tr>
<td>View</td>
<td>Description</td>
<td>How to open the view</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data Foundation Editor</td>
<td>Use this editor to define and maintain the data foundation structure and its connections.</td>
<td>Double-click a data foundation in the Local Projects View.</td>
</tr>
<tr>
<td>Business Layer Editor</td>
<td>Use this editor to define and maintain the business layer and its data source.</td>
<td>Double-click a business layer in the Local Projects View.</td>
</tr>
<tr>
<td>Connection Editor</td>
<td>Use this editor to edit connection and connection shortcut parameters.</td>
<td>Double-click a connection or connection shortcut in the Local Projects View. To open the editor for a secured connection, double-click the connection in the Repository Resources View.</td>
</tr>
<tr>
<td>Project Synchronization View</td>
<td>Use this view to manage shared project resources in the repository and synchronize local resources with the repository.</td>
<td>Select <strong>Window &gt; Project Synchronization</strong>.</td>
</tr>
<tr>
<td>Check Integrity Problems</td>
<td>Use this view to review the results of last integrity check.</td>
<td>Select <strong>Window &gt; Check Integrity Problems</strong>.</td>
</tr>
<tr>
<td>Query Panel</td>
<td>Use this view to run queries on business layers and published universes.</td>
<td>To run a query on a business layer, in the Business Layer Editor Query pane, edit an existing query, or create a query. To run a query on a published universe, in the Repository Resources View, right-click a universe and select <strong>Run Query</strong>. You can also run a query on a published universe in the Security Editor.</td>
</tr>
<tr>
<td>Security Editor</td>
<td>Use this editor to define security profiles and assign profiles to users.</td>
<td>Select <strong>Window &gt; Security Editor</strong>.</td>
</tr>
</tbody>
</table>

**Related Topics**
- [Resetting the user interface display](#)
- [About local projects](#)
- [About managing repository resources](#)
- [About the Data Foundation Editor](#)
- [About the Business Layer Editor](#)
- [About editing connections and connection shortcuts](#)
• About project synchronization
• Running check integrity
• Using the Query Panel
• About the Security Editor

2.4.1 Resetting the user interface display

The information design tool user interface can be customized by dragging and dropping editor tabs and views, minimizing views, and hiding and splitting panels within the views.

To reset the user interface to the default configuration, select **Window > Reset to Default Display**.

2.5 Setting preferences in the information design tool

To set preferences, from the information design tool main menu, select **Window > Preferences**. The following table gives a brief description of the types of preferences you can set. For more information, see the related topics.

<table>
<thead>
<tr>
<th>Preference type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
<td>These settings and customizations require a good understanding of the Eclipse development environment. To learn more about Eclipse, search for the Eclipse Foundation Web site.</td>
</tr>
</tbody>
</table>
| **Help**        | Lets you select how you would like the help topics to display when you click the help icon.  
**>Content:** You can use these settings to include custom help files. These settings require an understanding of the Eclipse help system. To learn more about Eclipse, search for the Eclipse Foundation Web site. |
## Preference type

<table>
<thead>
<tr>
<th>Preference type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Information Design Tool</strong></td>
<td>Lets you set check integrity rules to run automatically when saving resources. You can also set the severity level of the rules. For more information, see the related topic.</td>
</tr>
<tr>
<td>Check Integrity</td>
<td>Lets you set display options for connections in the data foundation editor.</td>
</tr>
</tbody>
</table>
| Data Foundation Editor | > **Appearance**: Set display options for columns, tables, and joins in the Data Foundation Editor.  
| | > **Detections**: Set whether or not to automatically detect tables, joins, and cardinalities when inserting tables into the data foundation.  
| | > **Performance**: Set options that impact how graphics are displayed in the Data Foundation Editor.  
| | > **Printing**: Set print options when saving the data foundation as a report. |
| Languages | Lets you change the language of the user interface and the Preferred Viewing Locale. |
| Online Tutorials | Lets you update the link to the online tutorials. |
| Secured Connections | Lets you set whether to use the server or the local middleware driver for secured relational connections. |
| Show Values | Lets you set how to display table and column values for the Show Values commands. |

### Related Topics
- Setting preferences for check integrity
- Setting preferences for the Data Foundation Editor
- Setting languages used by the information design tool
- Setting a link for online tutorials
- Setting middleware for secured relational connections
2.5.1 Setting preferences for check integrity

In the check integrity preference page, you can select rules to be run automatically whenever you save a resource. You can also change the severity of the messages returned by each rule.

1. From the information design tool main menu, select Window > Preferences.
2. In the "Preferences" dialog box, expand the Information Design Tool node and select Check Integrity.
3. To select rules to be run automatically when you save a resource:
   a. Select the Enable background check integrity on save option.
   b. Select the rules to include in the background check.
      The Cost column indicates the relative processing time required to execute the rule.
4. To change the severity of the messages returned by a rule, click in the Severity column for the rule. Select a severity from the list.
5. To restore the default values for check integrity preferences, click Restore Defaults.
6. To save the changes and continue editing preferences, click Apply.
7. To save the changes and close the "Preferences" dialog box, click OK.
   The background check takes effect immediately.

Related Topics
• Running check integrity

2.5.2 Setting preferences for the Data Foundation Editor

The Data Foundation Editor preference pages allow you to change detection options, how objects are displayed in the Data Foundation Editor, and print options for the data foundation.

1. From the information design tool main menu, select Window > Preferences.
2. In the "Preferences" dialog box, expand the Information Design Tool node and select Data Foundation Editor.
3. The "Data Foundation Editor" page allows you to select options for how connections are displayed in the data foundation.
4. To change other display options, expand the Data Foundation Editor node and select one of the following pages:
5. To restore the default values for preferences on the current page, click **Restore Defaults**.

6. To save the changes and continue editing preferences, click **Apply**.

7. To save the changes and close the "Preferences" dialog box, click **OK**.

   The new preferences take effect immediately.

**Related Topics**
- About the Data Foundation Editor
- Saving resources as reports

### 2.5.3 Setting languages used by the information design tool

1. From the information design tool main menu, select **Window > Preferences**.
2. In the "Preferences" dialog box, expand the **Information Design Tool** node and select **Languages**.
3. To change the language of the user interface, select the language from the "Product Languages" list.
4. To change the **Preferred Viewing Locale**, select the language from the list.
   
   For information about the Preferred Viewing Locale and how it impacts the language display, see the related topic about multilingual universes.
5. To save the changes and continue editing preferences, click **Apply**.
6. To save the changes and close the "Preferences" dialog box, click **OK**.
7. Exit and restart the information design tool for the language change to take effect.

**Related Topics**
- Multilingual universes
2.5.4 Setting a link for online tutorials

You can access online tutorials about the information design tool from the Help menu. The "Online Tutorials" preferences page allows you to update the URL address to the tutorials.

1. From the information design tool main menu, select Window > Preferences.
2. In the "Preferences" dialog box, expand the Information Design Tool node and select Online Tutorials.
3. Enter the new URL address in Online Tutorial Address.
4. To save the change and continue editing preferences, click Apply.
5. To save the change and close the "Preferences" dialog box, click OK.

The new address takes effect immediately.

Related Topics

• How to get help using the information design tool

2.5.5 Setting middleware for secured relational connections

The secure connections middleware preference applies only if the "Download connection locally" right is granted in the Central Management Console for the connection.

When running queries on secured relational connections in the information design tool, you can choose to run queries on the server using the server middleware driver, or locally using the local middleware driver.

1. From the information design tool main menu, select Window > Preferences.
2. In the "Preferences" dialog box, expand the Information Design Tool node and select Secured Connections.
3. Select the middleware to use:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server middleware</td>
<td>To use the middleware driver on the repository server.</td>
</tr>
<tr>
<td>Local middleware</td>
<td>To use the middleware driver on the local machine.</td>
</tr>
</tbody>
</table>

4. To restore the default value, click Restore Defaults.
5. To save the changes and continue editing preferences, click Apply.
6. To save the changes and close the "Preferences" dialog box, click OK.
2.5.6 Setting preferences for showing values

For the show values commands in the data foundation and business layer editors, you can select how to display values.
1. From the information design tool main menu, select **Window > Preferences**.
2. In the "Preferences" dialog box, expand the **Information Design Tool** node and select **Show Values**.
3. Select how you want the values to display.
4. To save the change and continue editing preferences, click **Apply**.
5. To save the change and close the "Preferences" dialog box, click **OK**.

The new display option takes effect immediately.

Related Topics
- Showing and profiling values in a data source

2.6 How to get help using the information design tool

The **Help** menu in the information design tool lets you link to different types of user assistance for the application.
Cheat Sheets shows how to complete a complex task, for example, creating a relational universe.

The Cheat Sheets command displays a list of available cheat sheets. Double-click a cheat sheet to start. It opens in the information design tool Help View.

For some steps, you can click **Click to perform** and the application will start the appropriate wizard to help you complete that step.

To get more help on a step, click the help icon 🔄.

### Online Tutorials

The Online Tutorials command links to the list of official product tutorials for the information design tool on the SAP Community Network.

If the URL address of the online tutorials changes, you can enter the new address in the information design tool preferences.

### Help Contents

The Help Contents command opens the Information Design Tool User Guide in a help window. To display help topics, you can navigate the Table of Contents, search the text, or look up topics in the index.

**Note:**


### Search

The Search command opens the Help View on the search function. To search the contents of the Information Design Tool User Guide, enter text in the **Search expression** field.

**Related Topics**

- Setting a link for online tutorials
Creating Universes

3.1 How to create a universe with the information design tool

Before you begin:

- Make sure the middleware drivers are configured for the data sources to which you want to connect.
  For more information about middleware configuration, see the Data Access Guide.

For information about supported data sources, see the Supported Platforms documentation available at the SAP BusinessObjects Support site at http://service.sap.com/bosap-support.

- Make sure you have the appropriate rights defined in the Central Management Console (CMC). See the related topic about CMC rights for information design tool users.

- Decide if the data foundation type should be single-source or multisource-enabled. The type and number of connections available, as well as the SQL syntax that is used to define SQL structures depends on the data foundation type. For more information, see the related topic about data foundation types.

Connections for multisource-enabled data foundations must be secured, relational connections, and are managed by the data federation service. Information about tuning the data federation service can be found in the Data Federation Administration Tool Guide.

- The procedure to create a universe varies slightly for different types of data sources. More information is available about the following types of universes:
  - Universes on SAP NetWeaver BW
  - Universes on SAP ERP
  - Universes on Microsoft Analysis Services (MSAS)
  - Universes on Essbase
  - Universes on SAS
  - Multilingual universes

  Note:
  For OLAP universes, you do not need to create a data foundation. The business layer is built directly from the objects you select in the source cube.

Links to more information on each step in the procedure can be found in the Related Topics.

1. Create a local project. In the Local Projects View select File > New > Project.
   The resources you use to build the universe are created and stored in the project.

2. Define the connections. Connections can be local or secured:
• Create a local connection if you want to publish the universe on the local file system. Later, you can publish the business layer to a repository.
• Create a secured connection if you want to create a multisource-enabled universe, or if you want to publish the universe in a repository without first publishing locally. For secured connections, you must create connection shortcuts in the local project to reference the secured connections in the repository.

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To create a local relational connection</td>
<td>In the Local Projects View, right-click the project folder and select <strong>New &gt; Relational Connection</strong>.</td>
</tr>
<tr>
<td>To create a local OLAP connection</td>
<td>In the Local Projects View, right-click the project folder and select <strong>New &gt; OLAP Connection</strong>.</td>
</tr>
<tr>
<td>To create a secured relational connection</td>
<td>In the Repository Resources View, start a repository session. Right-click the Connections folder or sub-folder and select <strong>Insert Relational Connection</strong>. To create a connection shortcut, select the connection in the Connections folder of the repository and select <strong>Create Relational Connection Shortcut</strong>.</td>
</tr>
<tr>
<td>To create a secured OLAP connection</td>
<td>In the Repository Resources View, start a repository session. Right-click the Connections folder or sub-folder and select <strong>Insert OLAP Connection</strong>. To create a connection shortcut, select the connection in the Connections folder of the repository and select <strong>Create OLAP Connection Shortcut</strong>.</td>
</tr>
</tbody>
</table>

3. Create the data foundation (relational data sources only). In the Local Projects View, right-click the project folder and select **New > Data Foundation**.
   • For a single-source universe, select a single connection identifying the database source.
   • To build a data foundation with multiple relational connections, create a multisource-enabled data foundation.

   The data foundation opens in the editor. To build the structure of the data foundation, see the related topic.

4. Create the business layer. In the Local Projects View, right-click the project folder and select **New > Business Layer**.
   • For relational business layers, select the data foundation to be the basis for the business layer. You can choose to automatically generate objects in the business layer for all data foundation structures, or select columns to be mapped as objects.

   **Note:**
   To build a universe on multiple data sources (relational data sources only), the business layer must be based on a multisource-enabled data foundation.
• For OLAP business layers, select the connection to the OLAP cube. Objects are created automatically for all structures in the cube.

The business layer opens in the editor. To build the business layer, see the related topic.

5. In the business layer, you can create and run queries to validate and test the universe.

6. Publish the business layer:
   • Business layers based on local connections must be published to a folder on the local file system. Later, you can publish the resulting local universe to a repository.
   • Business layers based on one or more secured connections must be published to the repository on the same Central Management System where the secured connections are stored.

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To publish the universe locally</td>
<td>In the Local Projects View, right-click the business layer and select Publish &gt; To a Local Folder.</td>
</tr>
<tr>
<td>To publish the universe to a repository</td>
<td>In the Local Projects View, right-click the business layer and select Publish &gt; To a Repository.</td>
</tr>
</tbody>
</table>

7. Follow the steps below if you created a local universe and now want to publish the universe to a repository. Business layers based on local connections can be changed to reference secured connections and published to a repository.
   a. In the Local Projects View, right-click the local connection and select **Publish Connection to a Repository**. When asked, create a connection shortcut.
   b. For relational connections, edit the data foundation and change the connection to use the connection shortcut.
   c. For OLAP connections, edit the business layer and change the connection to use the connection shortcut.
   d. In the Local Projects View, right-click the business layer and select **Publish > To a Repository**.

8. Define universe security. To open the Security Editor, on the information design tool main menu, select **Window > Security Editor**. Open a session on the repository where the universe is published. Use the Security Editor to define security profiles on the published universe. You also use the Security Editor to assign profiles to users and groups.

**Related Topics**
- About the information design tool interface
- About resources in the information design tool
- CMC rights for information design tool users
- About data foundation types
- Creating a local project
- Creating a relational connection
- Creating an OLAP connection
- Creating a connection shortcut
3.2 Universes on SAP NetWeaver BW

To build a universe on SAP NetWeaver BW, you must create a multisource-enabled data foundation.

**Note:**
For information on the authorizations needed to allow users of query and reporting applications to access multisource-enabled universes on SAP NetWeaver BW, see SAP Note #1465871.

Relational connections to SAP NetWeaver BW are managed by the data federation service. For information about optimizing queries, see the *Data Federation Administration Tool Guide*.

**Mapping of InfoProvider objects**
When you add an SAP NetWeaver BW connection to a data foundation, by default tables and joins are automatically inserted. When you create a business layer on the data foundation, by default objects are automatically inserted into the business layer.

To turn off automatic insertion, unselect the *Detect tables* option in the advanced properties of the connection when you are adding the connection into the data foundation. To turn off automatic insertion of business layer objects, unselect the *Automatically create classes and objects* option when selecting the data foundation in the New Business Layer wizard.

The following table describes how the objects in an InfoProvider map to the objects that are automatically inserted into the data foundation and business layer in the information design tool. The naming convention for the automatically generated data foundation tables are as follows:

- **I-Table:** A table with a name prefixed by I (InfoCube) that maps to the InfoProvider fact table.
- **D-Table:** A table with a name prefixed by D (Dimension) that maps to an InfoProvider master data table.
- **T-Table:** A table with a name prefixed by T (Text) that maps to an InfoProvider text table.
<table>
<thead>
<tr>
<th>InfoProvider object</th>
<th>Data foundation</th>
<th>Business layer</th>
</tr>
</thead>
</table>
| Fact table          | • Inserts an I-Table mapped to the fact table.  
                      • Inserts D-Tables mapped to each master data table.  
                      • Inserts T-Tables mapped to each text table. |
|                     |                 | ✉️ Inserts a folder in the business layer for each InfoProvider dimension (except for the Data Package and Unit dimensions).  
                      Business layer objects for the characteristics in the Unit dimension are inserted into the folder for the related key figure. The Data Package dimension is not mapped. |
<p>| Dimension 🕵️‍♂️     |                 |                |</p>
<table>
<thead>
<tr>
<th><strong>InfoProvider object</strong></th>
<th><strong>Data foundation</strong></th>
<th><strong>Business layer</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Characteristic</td>
<td>Inserts a column in the data foundation fact table that maps to the master data value.</td>
<td># Inserts a dimension object in the dimension folder associated with the I-Table column.</td>
</tr>
<tr>
<td></td>
<td>Creates an alias of the associated T-Table and links it to the column in the I-table. The T-Table contains columns for the description of the master data in short, medium, or long format. The table also has a column CAPTION which contains the longest description available for the characteristic.</td>
<td>Note: A characteristic of data type DATS or TIMS has no associated text table because date and time characteristics do not have descriptions.</td>
</tr>
</tbody>
</table>
| Display attribute*      | If a characteristic contains at least one display attribute, inserts an alias of the D-Table. Inserts a column in this table mapped to the display attribute. The D-Table contains a column for each display attribute of the characteristic. | ![Inserts folder under the dimension for the parent characteristic.](image)
|                         | Inserts an alias of the T-Table for each display attribute. | ![Inserts attribute objects under the dimension object for each column in the T-Table.](image) |
| Navigation attribute*   | Inserts a column in the I-Table and a column in the D-table mapped to the navigation attribute. T-Tables are linked directly to I-Table for both the parent characteristic and its navigation attributes. | ![Inserts dimension objects in this folder for each column in the D-Table.](image) |

*Display attribute* and *Navigation attribute* indicate attributes that may be specific to certain datasets or requirements.
<table>
<thead>
<tr>
<th><strong>InfoProvider object</strong></th>
<th><strong>Data foundation</strong></th>
<th><strong>Business layer</strong></th>
</tr>
</thead>
</table>

- Inserts a dimension object in the dimension folder associated with the I-Table column.

The dimension object for the navigation attribute is at the same level as the dimension for the parent characteristic but not necessarily adjacent.

**Tip:**

Filters on the object for the navigation attribute are more efficient than filters on the display attribute object. When filtering a navigation attribute, the fact table is filtered directly.
### Key figure

Inserts a column in the I-Table with the technical name of the key figure.

- For a key figure without unit or currency, inserts a measure in the Measures folder.
- For a key figure with unit or currency, inserts a sub-folder in the Measures folder.
- Inserts a measure in the sub-folder for the key figure.
- Inserts a dimension in the sub-folder for each unit or currency characteristic.

<table>
<thead>
<tr>
<th>InfoProvider object</th>
<th>Data foundation</th>
<th>Business layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key figure</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Inserts a column in the I-Table with the technical name of the key figure.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Time-dependent data | If the InfoProvider contains time-dependent data, creates input columns in the appropriate data foundation tables to handle time-dependent data. |
|                     | Creates a parameter in the data foundation called key date for each input column. By default, at query time, the key date parameter is not prompted. It is automatically assigned the current date. You can change this behavior by editing the key date parameter. For more information on input columns and editing parameters, see the related topics. | The business layer inherits the key date parameter. |

* If the flag “Attribute only” is checked for a display or navigation attribute in the InfoProvider, it is not exposed in the data foundation.

**Refreshing a universe**

The **Insert Candidate Objects** command helps you refresh the universe when objects are added to the underlying InfoProvider. Insert updated objects from the SAP NetWeaver BW connection into the existing data foundation. Refresh the business layer based on the data foundation changes using the **Insert Candidate Objects** command. For more information on candidate objects, see the related topic.

**Related Topics**

- How to create a universe with the information design tool
- About business layer objects
- About input columns
- Inserting and editing a parameter
3.3 Universes on SAP ERP

To build a universe on SAP ERP, you must create a data foundation.

When you create a relational connection to the SAP ERP data source, the InfoSets, SAP Queries, and ABAP functions in the data source are exposed as tables in the connection. For more information about how ERP data sources are mapped in the connection, see the Data Access Guide.

The data foundation can be single-source to support local connections. However, single-source data foundations do not support calculated columns or joins between tables inserted from an ERP connection. For support of calculated columns and joins, create a multisource-enabled data foundation on a secured connection.

When you insert a table into the data foundation, the table type of InfoSet, SAP Query, or ABAP Function is saved as a table property in the data foundation.

When you insert an ABAP Function table, one data foundation table is created to map the main function. The table contains input columns for the input parameters of the function. These parameters can be mandatory or optional. To assign a value to mandatory parameters, you need to edit the input columns. To do this, see the related topic.

Restriction:
Measures containing aggregate functions cannot be used as filters in the Query Panel. This limitation is due to the fact that the resulting SQL expression contains the HAVING clause, which is not supported by the SAP ERP connection. If you add a measure containing an aggregation function as a filter, an error occurs when you refresh the query.

Related Topics
- How to create a universe with the information design tool
- Editing input columns
- About parameters and lists of values in the data foundation
- About multisource-enabled data foundations

3.4 Universes on Microsoft Analysis Services (MSAS)

When you create a business layer on an MSAS data source, the business layer objects are generated automatically. The table below gives details about how certain objects in the MSAS cube are mapped in the business layer.
When you create a business layer, in the “Select OLAP Connection” page of the New Business Layer wizard, the base cube in the MSAS data source is listed first in the list of connection cubes. Additional cubes and perspectives in the data source are mapped as cubes and are listed in alphabetical order.

The cube you select in the list of connection cubes becomes the basis for the objects in the business layer.

<table>
<thead>
<tr>
<th>MSAS object</th>
<th>Business layer mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perspective</td>
<td>When you create a business layer, the base cube in the MSAS data source is listed first in the list of connection cubes. Additional cubes and perspectives are listed in alphabetical order. The cube you select in the list of connection cubes becomes the basis for the objects in the business layer.</td>
</tr>
<tr>
<td>Dimension</td>
<td>Analysis dimensions are created in the business layer for each dimension in the cube.</td>
</tr>
<tr>
<td>Display folder</td>
<td>Folders are created in the analysis dimension to contain the hierarchies in the display folder.</td>
</tr>
<tr>
<td>Hierarchy</td>
<td>For value-based (parent-child) hierarchies, a value-based hierarchy is created in the analysis dimension. The attributes are created in the Attributes folder in the hierarchy.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Unbalanced hierarchies are supported.</td>
</tr>
<tr>
<td></td>
<td>For level-based hierarchies, a business layer dimension is created in the analysis dimension. A hierarchy is created in the business layer dimension with the levels and their properties (as level attributes) in the Levels folder.</td>
</tr>
<tr>
<td>Attribute hierarchy</td>
<td>Attribute hierarchies in the cube are created as level-based hierarchies in the analysis dimension.</td>
</tr>
<tr>
<td>Named set</td>
<td>Named sets are created in the related analysis dimension, in the folder Named sets.</td>
</tr>
<tr>
<td>Measure group</td>
<td>Folders are created to contain the measures in measure groups and subgroups.</td>
</tr>
<tr>
<td>Measure</td>
<td>Measures and calculated measures are created as measures in the appropriate measure group folder. A measure attribute is created for the formatted value.</td>
</tr>
</tbody>
</table>
KPIs are not exposed in connection metadata, but you can make the KPI values available to queries by creating measures in the business layer with the MDX functions `KPIValue` and `KPIGoal`.

For example, if the cube contains a KPI with the name Operating Profit, you can create measures in the business layer with the following MDX expressions. In this example, the measures are created in the measure group folder\subfolder Performance\Profit. The name of the KPI value in the cube is surrounded by double quotes in the MDX function.

<table>
<thead>
<tr>
<th>Measure name</th>
<th>MDX Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Profit</td>
<td><code>KPIValue(&quot;Operating Profit&quot;)</code></td>
</tr>
<tr>
<td>Profit Target</td>
<td><code>KPIGoal(&quot;Operating Profit&quot;)</code></td>
</tr>
<tr>
<td>Profit Variance</td>
<td><code>((@select(Performance\Profit\Actual Profit) - @select(Performance\Profit\Profit Target)) / abs(@select(Performance\Profit\Profit Target))</code></td>
</tr>
<tr>
<td>Profit Pct Achieved</td>
<td><code>IIF(ISEMPTY(@select(Performance\Profit\ProfitTarget)),null,@select(Performance\Profit\Profit Variance) +1)</code></td>
</tr>
</tbody>
</table>

You can use the MSAS-specific function `VisualTotals` in the MDX expressions for business layer objects. For an example of implementing visual totals, see the Product Analysis query in the Sample Universe on a Microsoft OLAP cube on the SAP Community Network: [http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/b0b69a4d-2652-2e10-7d8a-ed1a62699f30](http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/b0b69a4d-2652-2e10-7d8a-ed1a62699f30).

The business layer can be refreshed to reflect changes in the underlying cube using the **Refresh Structure** command in the Business Layer Editor **Actions** menu.

**Related Topics**

- How to create a universe with the information design tool
- OLAP data source properties
- About business layer objects
- Inserting and editing measures
- Refreshing an OLAP business layer
3.5 Universes on Essbase

When you create a business layer on an Essbase data source, the business layer objects are generated automatically. The table below gives details about how certain objects in the Essbase cube are mapped in the business layer.

<table>
<thead>
<tr>
<th>Essbase object</th>
<th>Business layer mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension</td>
<td>Analysis dimensions are created in the business layer for each dimension in the cube.</td>
</tr>
<tr>
<td>Accounts dimension</td>
<td>In the New Business Layer wizard, you select which dimension to use to create the measures in the business layer. By default this is the dimension flagged as the Accounts dimension. Measures are created in the business layer for each object in the dimension. The organization of the measures in the Essbase outline is maintained in the business layer. <strong>Tip:</strong> For certain applications you may want to specify a dimension other than the Accounts-type dimension for the measures. In this case, the measures are created as an analysis dimension in the business layer and the analysis capabilities of hierarchies, such as member selection, are available.</td>
</tr>
<tr>
<td>Hierarchy</td>
<td>For each hierarchy in the cube, a hierarchy is created in the analysis dimension. All hierarchies are generated as value-based. <strong>Note:</strong> When opening the business layer in the Query Panel, hierarchy levels are determined spontaneously and can be selected in the query. Also, you can insert levels into the business layer.</td>
</tr>
<tr>
<td>User Defined Attribute (UDA)</td>
<td>UDAs are created as named sets defined on the associated hierarchy and appear in the analysis dimension.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Attributes are created in an Attributes folder in the hierarchy.</td>
</tr>
<tr>
<td>Attribute hierarchy</td>
<td>If the attributes are designed as a hierarchy in the cube, an attribute hierarchy is also created in the analysis dimension.</td>
</tr>
</tbody>
</table>
### Essbase object | Business layer mapping
---|---
Dynamic Time Series (DTS) | A DTS is not generated automatically in the business layer, but you can use MDX functions, such as HTD, QTD (history-to-date, quarter-to-date) in the object definitions.
Substitution variable | Substitution variables are not exposed in the business layer, but you can invoke a substitution variable in an MDX expression. The substitution variable name must be prefixed with the character &.
For example, if the cube contains a variable named CurrentMonth, you can use the variable in the definition of a named set:
```plaintext
WITH SET [Current Month] AS '([Time].[&CurrentMonth])'
```
Example of the substitution variable in the definition of a calculated member:
```plaintext
WITH MEMBER [Measures].[Current Month Quantity] AS '([Measures].[Quantity Sold], [Time].[&CurrentMonth])'
```

The business layer can be refreshed to reflect changes in the underlying cube using the **Refresh Structure** command in the Business Layer Editor **Actions** menu.

**Restriction:**
When you insert an MDX object into an Essbase business layer (for example, a named set, calculated member, or measure), be sure that the object name is not the same as data in the cube. For example, if Region is the name of a hierarchical level in the cube, you cannot name a new MDX object Region. If the new object is given the same name as data in the cube, the object is not usable in a query.

**Related Topics**
- How to create a universe with the information design tool
- About business layer objects
- Refreshing an OLAP business layer

### 3.6 Universes on SAS

To build a universe on SAS, you must create a multisource-enabled data foundation on a secured connection.

Connections to SAS are managed by the data federation service. For information about optimizing queries to SAS data sources, see the *Data Federation Administration Tool Guide*. 

3.7 Multilingual universes

The information design tool supports the creation of multilingual universes. This feature enables a multilingual solution using a single universe metadata model:

- The designer creates the universe in the source language in the information design tool.
- Translators translate the metadata in the data foundation and business layer using the translation management tool. For more information about translating metadata, see the related topic.
- Report designers can then build reports once from the same universe that can be displayed in several languages based on user preferences.

Three language parameters impact how labels, metadata, and data are displayed in the information design tool:

- The Product Language determines the language of the user interface of the information design tool. This parameter is set in the information design tool preferences.
- The Preferred Viewing Locale is the user's preferred language for viewing report and query objects in an application. This parameter is set in the information design tool preferences.
- Connection Language: For data sources that support a language parameter, the language parameter is entered when you create or edit a connection. This determines the language of the data.

Metadata source language in the information design tool

The metadata of the data foundation (table and column names) is created in the language of the metadata in the data source. The metadata that you insert into the data foundation can be entered in any language.

For SAP NetWeaver BW connections, the data foundation can be generated automatically in the language specified in the connection language parameter.

The metadata of a relational business layer is created in the language of the data foundation metadata. For OLAP business layers, the metadata is created in the language of the connection language parameter. The metadata that you insert into the business layer can be entered in any language.

Once the business layer is generated, the metadata (when viewed in the Business Layer Editor) remains in that language, even if you change the connection language parameter.
When designing the universe, with the @Variable function, you can use the PREFERRED_VIEWING_LOCALE and the DOMINANT_PREFERRED_VIEWING_LOCALE variables to customize the universe in order to filter multilingual data and retrieve only data in the user's Preferred Viewing Locale at query time.

**Multilingual display in the information design tool**

The Preferred Viewing Locale (defined in the information design tool preferences) determines the language of the metadata and data in the Query Panel, as long as two conditions are met:

- Translations are available in the language (metadata)
- The connection supports the language parameter (data)

When displaying values from the data source in the Connection Editor, the metadata and data displays in the language of the data source according to the current value of the connection language parameter.

**Related Topics**

- Translating universe metadata
- How to create a universe with the information design tool
- Setting languages used by the information design tool

### 3.7.1 Translating universe metadata

Before you begin, the universe you want to translate must be created in the source language, and published to a repository or local folder.

This procedure presents the recommended way to translate the data foundation and business layer metadata for universes created in the information design tool. For details on the procedures in the translation management tool, see the *Translation Management Tool User Guide*.

1. In the information design tool, create a local project if you do not already have one.
   
   When creating the project, note the file path to the directory where the project files are saved in the file system. The default root directory for all projects is workspace.

2. Retrieve the universe into the local project.
   
   The information design tool saves the .dfx and .blx files in the local project. These files correspond to the data foundation and business layer definitions. These are the files that are used as the source for the translations.

   **Note:**
   
   For OLAP universes, only a .blx file is saved.

3. In the translation management tool, translate the data foundation metadata (for relational universes):
   
   a. Import the .dfx file from project folder in the local file system.
   
   b. Translate the metadata.
   
   c. Export the translated content to the local file system.
For details on these workflows, see the *Translation Management Tool User Guide*.

4. Follow the same procedure as in previous step to translate the .blx file.

5. In the information design tool, to see the translations:
   a. In the application language preferences, select translated language as the Preferred Viewing Language. Exit and restart the information design tool for the language change to take effect.
   b. Open the business layer by double-clicking it in the Local Projects View. You can see the translated metadata in the Query Panel. To open the Query Panel, select the **Queries** pane and click **Insert Query**.

6. Re-publish the business layer so that translations are available to universe users.
   For universes published to a repository, you can open the Query Panel on the published universe by right-clicking the universe the Repository Resources View and selecting **Run Query**.

**Related Topics**

- [Creating a local project](#)
- [Retrieving a published universe from a repository](#)
- [Retrieving a published universe from the local file system](#)
- [Showing and profiling values in a data source](#)
- [Inserting and editing a query in the business layer](#)
- [Setting languages used by the information design tool](#)
- [Publishing a universe](#)
- [Running a query on a universe published in a repository](#)
Converting .unv universes

4.1 About .unv and .unx universes

You can use the **Convert .unv Universe** command in the information design tool to convert a universe created with other SAP BusinessObjects universe design tools, and universes created in previous versions. You can then work on the converted universe in a local project as you would a universe created with the information design tool.

**What is a .unv universe?**

A .unv universe refers to a universe created with any SAP Business Objects XI 3 design tool, for example Universe Designer.

The following SAP Business Objects BI 4 design tools create .unv universes:

- The universe design tool (new name for Universe Designer)
- The universe design tool desktop edition (new name for Universe Designer Personal)

The universe is stored with a file name of `<universe name>.unv` in a local folder, or in a repository.

**What is a .unx universe?**

When you publish a universe using the information design tool, the universe is stored with a file name of `<universe name>.unx`. This is called a .unx universe. The **Convert .unv Universe** command converts a .unv universe to the .unx universe format.

**Related Topics**

- [About converting .unv universes](#)

4.2 About converting .unv universes

Before working with .unv universes in the information design tool, you must convert them.

**What .unv universes can be converted?**

You can convert the following types of .unv universes:
• Relational universes created using the SAP BusinessObjects BI 4 tools universe design tool or universe design tool desktop edition.
• Relational universes created using SAP BusinessObjects Enterprise XI 3 design tools.

**Note:**
Before you can convert universes created in version XI 3 that are saved in a repository, you must upgrade the universes using the upgrade management tool. For more information, see the *SAP BusinessObjects Business Intelligence platform Upgrade Guide*.

You cannot convert the following types of .unv universes:
• OLAP universes
• Stored procedure universes
• Universes based on a Data Federator data source
• Javabean universes

**Note:**
It is not possible to convert Business Views created with Business View Manager XI 3 to a format compatible with version BI 4 reporting tools.

**How to convert .unv universes**

How you convert .unv files depends on the software version of the tool that was used to create the universe, and whether the universe is stored locally or in a repository. The following table describes the steps to follow for different conversion scenarios. For details of the conversion procedures, see the Related Topics.

<table>
<thead>
<tr>
<th>Universe to convert</th>
<th>Workflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>A .unv universe that has been saved in a repository using XI 3 design tools.</td>
<td>First upgrade the universe in the repository to the latest version using the upgrade management tool. In the information design tool, follow the procedure for converting a .unv universe in a repository. The conversion creates an equivalent .unx universe in the repository with the associated universe and connection rights.</td>
</tr>
<tr>
<td>A .unv universe that has been saved in a repository using the universe design tool release BI 4.0 or later.</td>
<td>In the information design tool, follow the procedure for converting a .unv universe in a repository. The conversion creates an equivalent .unx universe in the repository with the associated universe and connection rights.</td>
</tr>
</tbody>
</table>
Converting .unv universes

**Universe to convert**

Any locally-stored .unv universe that has been created using version XI 3 or later design tools.

**Note:**

A locally-stored universe refers to a non-secured universe that was saved for all users.

<table>
<thead>
<tr>
<th>Workflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>In the information design tool, follow the procedure for converting a locally-stored .unv universe.</td>
</tr>
<tr>
<td>The conversion creates the equivalent universe resources (data foundation, business layer, and local connection) in a local project.</td>
</tr>
<tr>
<td>Publish the business layer to create the .unx universe.</td>
</tr>
</tbody>
</table>

**After converting .unv universes**

When converting a .unv universe, the .unv universe is preserved. Documents in SAP BusinessObjects query and reporting tools based on the universe are still linked to the .unv universe. This gives you the opportunity to check and test the converted universe before changing the documents that depend on it.

Some features of .unv universes are implemented differently in the .unx universe. Once you have converted a universe, you can edit the universe resources in a local project in the information design tool to check and correct inconsistencies; and to take advantage of new universe features. For a description of the supported features and how they are implemented in .unx universes, see the related topic.

After converting the universe, it is recommended to refresh the structure of the data foundation and run a check integrity on the universe. For tips on resolving check integrity errors on converted universes, see the related topic.

**Related Topics**

- Converting a .unv universe in a repository
- Converting a locally-stored .unv universe
- Features supported when converting .unv universes
- Tips for resolving check integrity errors after converting .unv universes
- About .unv and .unx universes

### 4.3 Features supported when converting .unv universes

When converting a .unv universe with the information design tool, the conversion process creates equivalent features in the converted universe. The table below indicates which .unv universe features are supported and how they are implemented in the .unx universe. For certain features, tips are described for obtaining best conversion results.
### Feature in original .unv universe

The objects in the universe schema are created in the data foundation:
- Tables
- Alias tables
- Derived tables (including nested derived tables)
- Joins (including shortcut joins)
- Self-joins (converted to column filters)

**Tip:**
@Prompts in self-join expressions may require manual intervention after conversion. See the related topic on resolving check integrity errors.
- Contexts

**Tip:**
Contexts are converted with all joins explicitly included or excluded. In the data foundation editor, you can take advantage of the simplified contexts feature. You can manually restrict the context definition to the ambiguous parts of the schema using neutral joins. For more information about contexts, see the related topic.

### Universe schema

### Universe outline

The objects in the universe outline are created in the business layer with all their properties:
- Classes and sub classes (converted to folders)
- Dimensions. For time hierarchies, a dimension is created for each active level in the hierarchy.
- measures (including aggregation function)
- details (converted to attributes)
- conditions (converted to filters, including properties for mandatory filters)

### Multilingual universe

All translated strings, language settings, and locale settings are converted.
### Converting .unv universes

<table>
<thead>
<tr>
<th>Feature in original .unv universe</th>
<th>Feature in converted .unx universe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linked universe (core and derived)</td>
<td>A core universe is converted like any .unv universe. The conversion of a core universe does not trigger the conversion of the derived universes that depend on it. A derived universe contains a link to a core universe. When you convert a derived universe, the conversion automatically includes all core universes that the derived universe is linked to. You do not need to convert the core universes ahead of time as a separate step. The data foundation of the converted universe contains all the tables and joins from all core universes and any tables, joins, or contexts that were defined in the derived universe. The business layer contains all classes, objects, and conditions from all core universes including the classes, objects and conditions that were defined in the derived universe.</td>
</tr>
<tr>
<td>Universe Controls Parameters: Query Limits</td>
<td>Query limits are converted and can be edited in the business layer.</td>
</tr>
<tr>
<td>Universe SQL Parameters: SQL restrictions</td>
<td>Query, multiple SQL statements, and Cartesian product controls are converted. You can edit the Allow Cartesian products and Multiple SQL statements for each context restrictions in the data foundation. Edit all other SQL restrictions in the business layer.</td>
</tr>
<tr>
<td>Strategies</td>
<td>Custom strategies are not supported in .unx universes.</td>
</tr>
<tr>
<td>Universe Parameters: SQL generation parameters</td>
<td>Customizations to SQL generation parameter settings in the PRM file or in the universe parameters are not converted. You can add customized values to the converted PRM file, and customize universe parameter settings in the converted universe using the information design tool. <strong>Tip:</strong> Check and reset custom settings to SQL generation parameters in the data foundation properties and business layer properties. For information on setting SQL parameters, see the related topic.</td>
</tr>
<tr>
<td>Feature in original .unv universe</td>
<td>Feature in converted .unx universe</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td><strong>@Functions</strong></td>
<td>The following @functions are converted:</td>
</tr>
<tr>
<td></td>
<td>• @Aggregate_Aware</td>
</tr>
<tr>
<td></td>
<td>• @Prompt</td>
</tr>
<tr>
<td></td>
<td>• @DerivedTable</td>
</tr>
<tr>
<td></td>
<td>• @Select</td>
</tr>
<tr>
<td></td>
<td>• @Variable</td>
</tr>
<tr>
<td></td>
<td>• @Where</td>
</tr>
<tr>
<td></td>
<td>The syntax for all functions is supported.</td>
</tr>
<tr>
<td></td>
<td><strong>Tip:</strong> The @Prompt function has a new alternative syntax to take advantage of named parameters. For more information, see the related topic.</td>
</tr>
<tr>
<td><strong>Prompts</strong></td>
<td>@Prompts in dimension and measure expressions in the business layer, and in SQL expressions in the data foundation are supported.</td>
</tr>
<tr>
<td></td>
<td>When converting, you have the option to automatically create a named parameter for the prompt in the business layer.</td>
</tr>
<tr>
<td></td>
<td>@Prompt expressions in the data foundation are not converted. @Prompts in self-join expressions may require manual intervention after conversion. See the related topic on resolving check integrity errors.</td>
</tr>
<tr>
<td></td>
<td><strong>Tip:</strong> Parameters and lists of values in the information design tool can be defined independently from the objects that they reference. You can therefore reference a named parameter or list of values in more than one business layer object.</td>
</tr>
<tr>
<td><strong>Lists of values</strong></td>
<td>Named lists of values are created in the business layer for dimension and measure objects that specify a list of values.</td>
</tr>
<tr>
<td>Feature in original .unv universe</td>
<td>Feature in converted .unx universe</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------</td>
</tr>
</tbody>
</table>
| **Access Restrictions**          | When converting a universe in a repository, universe Access Restrictions are converted into security profiles that can be edited using the Security Editor:  
  - Access Restrictions (except object restrictions) are converted to settings in a Data Security Profile.  
  - Object Access Restrictions are converted to Create Query and Display Data settings in a Business Security Profile.  
  **Tip:**  
  Business Security Profiles allow you to secure the metadata separately from the data. For example, you can allow a user to create a query although the user is not allowed to see the corresponding data. For more information about universe security in the information design tool, see the related topic. |
| **Security assignments and priority** | When converting a universe in a repository, user and group assignments are converted.  
  **Tip:**  
  In the Security Editor, you can take advantage of the feature allowing you to assign more than one security profile to a user or group.  
  Group priority for access restrictions is converted.  
  **Note:**  
  For .unv universes, if a user belongs to different groups, the priority assigned to the groups determines what access restriction the user inherits if he has no access restriction assigned. In the converted universe, priority is assigned to Data Security Profiles instead of groups. If the priority for the profile assigned to the group is higher than the priority of the profile assigned to the user, the group profile is used. |
| **Connections**                  | When converting a universe in a repository, the same secured relational connection is used by both the .unv and .unx universes. If you retrieve the converted universe into a local project, a connection shortcut is created that references the secured connection in the repository.  
  **Tip:**  
  Relational connections can be created and shared by the universe design tool and the information design tool. The connections are published in the same Connections folder in the repository.  
  When converting a locally-stored (non-secured) universe, personal and shared connections are converted to local connections. |
4.4 Tips for resolving check integrity errors after converting .unv universes

After converting a .unv universe, it is recommended to run a check integrity in the information design tool on the converted universe. Certain errors in the check integrity results can be resolved by following the best practices described below.

Errors on column data types

Refresh the structure in the data foundation just after the conversion. This avoids data-type errors in the integrity check.

Errors on self-joins with @Prompts

If a join expression in the .unv universe contains an @Prompt with a list of values referring to an object, the converted join needs to be reworked in the data foundation. The steps to follow for two possible solutions are described. The descriptions use the following example:

The .unv universe contains a self-join on the dimProductStrings table with a prompt called Language. The self-join expression is:

```
dimProductStrings.LanguageID= @Prompt('Language','N','Language\Language Id',mono,constrained)
```

After converting the universe, the data foundation contains a column filter on the dimProductStrings table. The join expression for the filter contains the @Prompt.

The first solution consists of creating a prompted parameter and list of values in the data foundation:

1. Open the data foundation by double-clicking the data foundation name in the Local Projects View.
2. Click the Parameters and Lists of Values tab.
3. In the data foundation Lists of Values pane, define a list of values based on custom SQL for Language. For example, select 'LANGUAGES'.'LANGUAGEID', 'LANGUAGES'.'LANGUAGE CODE' FROM 'LANGUAGES'.
4. In the data foundation Parameters pane, define a parameter for Language. Select the option Prompt to users and associate the Language list of values to it.
5. In the data foundation, edit the column filter in the dimProductStrings table. Change the join expression to reference the new prompted parameter, for example: dimProductStrings.LanguageID= @Prompt(Language).
6. Save and close the data foundation.
The second solution consists of using a mandatory filter in the business layer:

1. Open the data foundation in the editor and delete the column filter on the dimProductStrings table that contains the @Prompt.
2. Save and close the data foundation.
3. Open the business layer in the editor.
4. In the business layer Lists of values pane, define a list of values based on custom SQL for Language. For example, `SELECT "LANGUAGES"."LANGUAGEID", "LANGUAGES"."LANGUAGECODE" FROM "LANGUAGES".
5. In the business layer Parameters pane, define a parameter for Language. Keep the default option to Prompt to users and associate the Language list of values to it.
6. In the business layer, in the folder associated with Product, create a filter with an expression that refers to the Language prompted parameter, for example: `dimProductStrings.LanguageID= @Prompt(Language)`.
7. In the Properties tab of the filter definition, select the Use filter as mandatory in the query option. Select the Filter scope of Apply on Folder.
8. Save and close the business layer.

Related Topics
- About refreshing a data foundation
- About the Data Foundation Editor
- About parameters and lists of values in the data foundation
- Inserting a column filter
- About the Business Layer Editor
- About parameters
- About lists of values
- Inserting and editing filters

4.5 Converting a .unv universe in a repository

The .unv universe to be converted must be stored in a repository compatible with the information design tool. If the .unv universe was created with a design tool version earlier than SAP BusinessObjects BI 4.0, you must first upgrade the universe using the upgrade management tool. For more information about upgrading universes, see the SAP BusinessObjects Upgrade Guide.

If you want to retrieve the converted .unx universe into a local project in order to work on it, you must first have a local project folder in the Local Projects View.

1. In the information design tool, select File > Convert .unv Universe.
2. In the "Convert a .unv Universe" dialog box, click the Select .unv universe from a repository icon.
3. Open a session on the repository where the .unv universe is saved, select the universe, and click OK.
4. Click the Browse button next to the **Destination Repository Folder** field, and then select a folder in the repository where you want to save the converted .unx universe.

5. If you want to retrieve the converted .unx universe into a local project in order to work on it, click the Browse button next to the **Destination Local Project Folder** field, select a project folder, and click **OK**.

6. If you want the conversion to create named parameters for prompts, select the **Automatically convert @Prompt expressions into universe named parameters** option. For more information about named parameters, see the related topic.

7. If you are retrieving the converted universe into a local project and want remove the local security requirement so that any user can open the universe resources without entering repository authentication, select the **Save for all users** option.

8. Click **OK** to start the conversion.

After conversion, it is recommended to refresh the structure of the data foundation, then run a check integrity on the universe to detect problems in the conversion. For tips on resolving check integrity errors, see the related topic.

**Related Topics**

- Features supported when converting .unv universes
- Opening a session
- Creating a local project
- About parameters
- About refreshing a data foundation
- Running check integrity
- Tips for resolving check integrity errors after converting .unv universes

### 4.6 Converting a locally-stored .unv universe

You need to define a local project folder in the Local Projects View in which to save the resources of the converted universe.

1. In the information design tool, select **File > Convert .unv Universe**.
2. In the "Convert a .unv Universe" dialog box, click the **Select .unv universe from the local file system** icon, and select the universe you want to convert.
3. Click the Browse button next to the **Destination Local Project Folder** field, select a project folder, and click **OK**.
4. If you want the conversion to create named parameters for prompts, select the **Automatically convert @Prompt expressions into universe named parameters** option. For more information about named parameters, see the related topic.
5. Click **OK** to start the conversion.
The conversion creates the equivalent universe resources (data foundation, business layer, and local connection) in the specified local project folder.

At this point, it is recommended to refresh the structure of the data foundation.

You can now publish the business layer to create the .unx universe file. This creates a local universe. To publish the universe to the repository, continue with the next step.

6. Publish the local connection to a repository.
7. Edit the data foundation and change the connection to use the secured connection published in the last step.
8. Publish the business layer to the repository.

The Publishing Wizard allows you to run a check integrity on the universe (recommended). For tips on resolving check integrity errors, see the related topic.

**Related Topics**

- Features supported when converting .unv universes
- Creating a local project
- About parameters
- About refreshing a data foundation
- Publishing a local connection to the repository
- Changing a connection in a data foundation
- Publishing a universe
- Running check integrity
- Tips for resolving check integrity errors after converting .unv universes
Converting unv universes
Retrieving published universes

5.1 Retrieving a published universe from the local file system

To retrieve a published universe, you must have a project in the Local Projects View where the business layer and referenced resources are to be saved.

1. In the Local Projects View, right-click the project folder and select Retrieve Universe > from Local Folder.
2. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help button.

When the wizard finishes, the business layer and its dependent resources (connections, connection shortcuts, and data foundation) are created in the local project and are ready to be edited.

Related Topics
  • Creating a local project

5.2 Retrieving a published universe from a repository

To retrieve a published universe, you must have a project in the Local Projects View where the business layer and referenced resources are to be saved.

1. You can retrieve a universe from a repository in two ways:
Retrieve published universes

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>From the Local Projects View</td>
<td>Right-click the project folder in the Local Projects View and select <strong>Retrieve Universe &gt; from a Repository</strong>.</td>
</tr>
<tr>
<td>From the Repository Resources View</td>
<td>Right-click the universe in the Repository Resources View and select <strong>Retrieve Universe</strong>.</td>
</tr>
</tbody>
</table>

**Note:**
By default, the resources are retrieved into the local project and are secured locally by requiring you to enter your repository system authentication when opening a retrieved data foundation or business layer.

To remove the local security requirement, select the **Save for all users** option when selecting the universe in the repository.

2. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help button.

When the wizard finishes, the business layer and its dependent resources (connections, connection shortcuts, and data foundation) are created in the local project and are ready to be edited.

**Related Topics**

- Opening a session
- Selecting a repository folder
- Creating a local project
6.1 About local projects

The first step in creating resources in the information design tool is to create a local project in the Local Projects View. You create and edit all resources (except secured connections and security profiles) in a local project.

The resources and folders in a local project are stored as physical files and folders in the local file system. The Local Projects View lets you to navigate local projects and open resources in the information design tool.

Once you have created a local project, there are several ways you can populate it with resources:

- Create resources using the wizards available on the New menu.
- Convert a .unv universe that was created with the universe design tool, or migrated from an earlier version.
- Retrieve a published universe.

You edit the resources using the information design tool editors by double-clicking the resource name in the local project.

You can also perform the following tasks on resources from the Local Projects View:

- Create a shared project so that you can share resources with other designers.
- Check integrity of data foundations and business layers.
- Publish a business layer as a universe to the local file system or a repository.
- Publish a connection to a repository.
- Show dependent resources.
- Save a resource as a report.

You can delete a project from the Local Projects View. The project files remain in the local file system until you explicitly delete them. Open the project to make it available again in the Local Projects View.

Related Topics

- Creating a local project
- Deleting a local project
- Opening a local project
- About converting .unv universes
- Retrieving a published universe from a repository
- About shared projects
6.1.1 Creating a local project

1. From the information design tool main menu, select **File > New > Project**.
2. Give the project a unique name.
3. In **Project Location**, the file path to the default root directory for all projects (workspace) displays. To select a different local folder to contain the project, click the browse button.
4. Click **Finish**.

The project is created in the local file system and displayed in the Local Projects View.

**Related Topics**
- About local projects

6.1.1.1 Resource names

Resource names identify the connections, data foundations, and business layers in the local project. You give a name to the resource when you create it. The name must be unique within the local project.

**Note:**
If you use the same name for resources in different projects, when publishing the resources to the same repository you may have naming conflicts because the names are not unique.

You can optionally enter a description of the resource.

**Related Topics**
- About resources in the information design tool

6.1.2 Opening a local project
1. From the information design tool main menu, select **File > Open Project**.
2. Select the **Select root directory** option and click **Browse**.
   The "Browse for Folder" dialog box opens with the default root directory for all projects (workspace) already selected.
3. Click **OK** to select the default directory, or browse to the folder containing the project you want to open.
   All projects not already open in the Local Projects View are listed in **Projects** and are selected by default.
4. Clear the checkbox for any projects you do not want to open and click **Finish**.

### 6.1.3 Deleting a local project

1. Right-click the project in the Local Projects View and select **Delete**.
2. If you want to permanently delete the project from the Local Projects View and the local file system, select the **Delete project contents on disk** option in the "Confirm Project Delete" dialog box.
   **Note:**
   If you select this option, the deletion is permanent and cannot be undone.
3. Click **Yes** to confirm the deletion.
   If you did not delete the project contents permanently, you can open the project to make it available again in the Local Projects View.

### Related Topics
- **Opening a local project**

### 6.1.4 Recovering local projects after workspace corruption

Sometimes when the information design tool ends unexpectedly, the local workspace becomes corrupted and you are unable to restart the information design tool. Use this procedure to recreate the workspace and recover your local projects.

1. On the local file system, navigate to the folder `%USERPROFILE%\.businessobjects\bimodeler_14\`. For example:
   ```
   C:\Documents and Settings\Administrator\.businessobjects\bimodeleter_14\
   ```
2. Rename the workspace folder to `workspace.bak`.
3. Start the information design tool.
   A new workspace folder is created automatically.
4. Select **File > Open Project** from the main menu.
5. In the "Import Existing Projects" dialog box, select the root directory and browse to the path of the `workspace.bak` folder created in step 2.
6. Select the projects you want to restore.
7. Select the **Copy projects into workspace** option and click **Finish**.

### 6.2 About shared projects

A shared project is a project in a repository whose resources are available to other designers. You create a shared project in a repository from an existing local project in the Local Projects View.

To start working on shared resources, you use the following tasks in the Project Synchronization View:

- Synchronize the project to copy resources between the local and shared projects.
- Lock and unlock resources in the shared project to inform other designers when you are working on them.
- Synchronize a shared project created by another designer. This creates a local project associated with the shared project so that you can start working on the shared resources.

**Related Topics**

- Creating a shared project from a local project
- Working in a shared project
- Synchronizing a project
- Locking a resource
- Unlocking a resource
- Merging changes to shared resources

### 6.2.1 Creating a shared project from a local project

The shared project will automatically be given the same name as the local project. A project with this name cannot already exist in the repository. If you need to rename an existing shared project, use the rename command in the Project Synchronization View. To do this, see the related topic.

1. In the Local Projects View, right-click the project you want to share and select **New Shared Project**.
2. In the "Open Session" dialog box, select the repository system session you want to open and enter your system authentication.
   
   The Project Synchronization View opens showing a shared project with the same name as the local project. At this point, the shared project is empty.
3. In the Project Synchronization View, synchronize the resources you want to save in the shared project.

**Related Topics**
- Renaming a shared project
- Opening a session
- Synchronizing a project

### 6.2.2 Working in a shared project

Use this procedure to work on resources in an existing shared project.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the **Shared Project** list.
3. In the selected shared project, lock the resources you want to work on.
   
   Locks are available as a communication tool between designers. When other designers open the Project Synchronization View, your lock informs them that you are making changes. It also prevents other designers from updating these resources in the shared project while you have them locked. However, any designer can unlock the resource if necessary.
4. Synchronize the project to update the resources in the local project with the latest changes saved on the server.
   
   If you do not already have a local version of the project, one is created in the Local Projects View.
   
   You may want to review the changes made on the server before updating them in the local project. For more information, see the related topic on merging changes in shared resources.
5. Once you have made your changes, in the Project Synchronization View, synchronize the project to save your changes on the server.
6. Unlock the resources.

**Related Topics**
- Opening the Project Synchronization View
- Locking a resource
- Synchronizing a project
- Merging changes to shared resources
- Unlocking a resource
6.2.3 Renaming a shared project

Use this procedure to rename a shared project that exists in the repository.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the Shared Project list.
3. Click the Rename Shared Project icon
4. Enter a new name that is unique in the repository.
5. To create a local project with the new name in the Local Projects View, synchronize the project.
   Local projects with the original name are no longer associated with the shared project with the new name. The resources in those local projects can no longer be synchronized with the newly named shared project.

Related Topics
- Opening the Project Synchronization View
- Synchronizing a project

6.2.4 Deleting a shared project

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the Shared Project list.
3. Click the Delete Shared Project icon

   Note:
   Deleting a shared project cannot be undone.

The shared project is deleted from the repository. Resources in the local projects associated with the deleted shared project are not affected, however synchronization statuses of the local project are lost.

6.3 About project synchronization
Synchronizing a project starts with comparing the resources in a project in the Local Projects View with an associated shared project on the repository server. Synchronization detects added resources, deleted resources, and differences between the resources. Based on the differences detected, you can update the local and shared resources.

Use the Project Synchronization View to synchronize a project. The view displays synchronization information in two panes:

- The **Shared Project** pane lists the resources in the shared project on the server. A lock icon appears next to the resource if it is locked. Other information about the resources on the server is given: the user who last modified the resource and on what date, the user who locked the resource and on what date.
- The **Synchronization Status** pane lists the status of each resource. The status is determined by comparing the resources in the local and shared projects.

The different synchronization statuses and what they mean are listed in the table.

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Added Locally</td>
<td>The resource was added in the local project, but not in the shared project.</td>
</tr>
<tr>
<td>Changed Locally</td>
<td>The resource was changed in the local project, but not in the shared project since the last synchronization.</td>
</tr>
<tr>
<td>Deleted Locally</td>
<td>The resource was deleted in the local project but still exists in the shared project.</td>
</tr>
<tr>
<td>Added on Server</td>
<td>The resource is not in the local project but exists in the shared project.</td>
</tr>
<tr>
<td>Changed on Server</td>
<td>The resource was changed in the shared project, but not in the local project since the last synchronization.</td>
</tr>
<tr>
<td>Deleted on Server</td>
<td>The resource exists in the local project, but was deleted in the shared project.</td>
</tr>
</tbody>
</table>
| Conflicting       | Any of the following situations creates a conflicting status:  
|                   | - The resource was changed in both the local and shared projects with different changes since the last synchronization.  
|                   | - A resource with the same name was added both in the local and shared projects since the last synchronization.  
|                   | - The resource was changed in the local project, but deleted from the shared project.  
|                   | - The resource was changed in the shared project, but deleted from the local project. |
| Synchronized      | The resources are identical. |
Three commands allow you to synchronize resources. When you select the resources to be synchronized, you can select individual resources or folders. The following table summarizes the possible synchronization actions.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Command</th>
<th>Synchronization Action</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>Get Changes from Server</td>
<td>For the selected resources:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the status is <strong>Added on Server</strong>, the resource is added to the local project.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the status is <strong>Changed on Server</strong>, the resource is updated in the local project.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the status is <strong>Deleted on Server</strong>, the resource is deleted from the local project.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the status is <strong>Conflicting</strong>, the resource on the server (whether it is changed, added, or deleted) is copied to the local project, regardless of the change made in the local project.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For all other statuses, no action is taken.</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>Save Changes on Server</td>
<td>For the selected resources:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the status is <strong>Added Locally</strong>, the resource is added to the shared project on the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the status is <strong>Changed Locally</strong>, the resource is updated in the shared project on the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the status is <strong>Deleted Locally</strong>, the resource is deleted from the shared project on the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the status is <strong>Conflicting</strong>, the resource in the local project (whether it is changed, added, or deleted) is copied to the shared project, regardless of the change made in the local project.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> If a resource is locked by another user, an error message displays and the changes and deletions are not made on the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For all other statuses, no action is taken.</td>
</tr>
</tbody>
</table>
For the selected resources, the local project is updated with the shared project on the server, regardless of the status.

**Note:**

*Revert Changes* updates the local project in the same way as *Get Changes from Server* with the exception that if a resource has been created in the local project and has not yet been saved on the server, *Revert Changes* deletes the new local resource, whereas *Get Changes from Server* preserves the new local resource.

**Related Topics**

- Synchronizing a project
- Locking a resource
- Unlocking a resource
- Merging changes to shared resources

### 6.3.1 Opening the Project Synchronization View

1. To open the Project Synchronization View, from the main menu, select **Window > Project Synchronization**.
2. Click the Change Session icon in the Project Synchronization View to open a session on the repository system where the shared projects are saved.

After entering your authentication information, you can manage shared projects, and select a project to synchronize in the Shared Project list.

**Related Topics**
- Opening a session
- About project synchronization

### 6.3.2 Synchronizing a project

To synchronize a project, the project must be shared.

Synchronize a project when you want to do the following tasks:
- Update local resources with changes stored in the shared project.
- Save in the shared project the changes you made to local resources.
- Revert local resources to the copy stored in the shared project.
- Create a local copy of a shared project.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the Shared Project list.
3. To see the latest synchronization status of the resources in the project, in the pane labeled Synchronization status (Local Project Compared to Shared Project), expand the project and click the refresh icon.
   
   For more information about synchronization status and possible actions, see the related topic about project synchronization.
4. Synchronize the project:
   - To update the local project with resources that have changed in the shared project, select the resources in the list and click the Get Changes from Server icon.
     
     If no project exists in the Local Projects View with the name of the shared project, a local project is created.

   **Note:**
   You may want to review the changes made on the server before updating them in the local project. For more information, see the related topic on merging changes in shared resources.
   - To update the shared project with changes made locally, select the resources in the list and click the Save Changes on Server icon.
Note:
Resources that are locked by another user cannot be updated on the server. However, if needed, any user can unlock the resource.

If you are updating the server with resources that you have locked, synchronizing updates the resources, but does not unlock them. You must explicitly unlock the resources on the server.

- To revert resources in the local project to the copy stored on the server, select the resources and click the Revert Changes icon.

Related Topics
- About project synchronization
- Opening the Project Synchronization View
- Locking a resource
- Unlocking a resource
- Merging changes to shared resources

6.3.3 Locking a resource

To lock a resource, the resource must be in a shared project.

Lock a resource when you want to inform other designers that you are working on the resource when they open the Project Synchronization View.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the Shared Project list.
3. In the Shared Project pane, expand the project.
4. Right-click the resource and select Lock.

Note:
The lock action does not update the resource contents in either the local or shared project. To save any changes, synchronize the resource.

Related Topics
- Opening the Project Synchronization View
- About project synchronization
6.3.4 Unlocking a resource

Unlock a resource once you have updated your changes on the server and you want to inform other designers that you have finished. Once you have unlocked the resource, other designers can lock it and/or update the server version with changes.

**Note:**
If necessary, you can unlock a resource locked by another user.

1. Open the Project Synchronization View with a session on the repository system where the shared project is saved.
2. Select the shared project from the Shared Project list.
3. In the Shared Project pane, expand the project.
4. Right-click the resource and select Unlock.

**Note:**
The unlock action does not update the resource on the server with any changes made in the local project. To save any changes, synchronize the resource.

**Related Topics**
- Opening the Project Synchronization View
- About project synchronization

6.3.5 Merging changes to shared resources

When synchronizing a shared resource, before getting changes from the server, you may want to review the changes and decide which changes to apply to the local resource. This procedure presents a way to manually merge changes between differing resources.

For example, you are working on a resource called NewDatafoundation in a local project. This project is shared in a repository. When you synchronize the project, NewDatafoundation has a synchronization status of Changed on Server, or Conflicting.

To review and manually merge the changes:

1. In the local project, right-click NewDatafoundation and select Copy.
2. Right-click again (in the local project) and select Paste.
   A copy of NewDatafoundation is saved in the local project.
3. In the Project Synchronization View, select NewDatafoundation and synchronize by selecting Get Changes from Server.
4. Open both NewDatafoundation and Copy of NewDatafoundation in the Data Foundation Editor by
double-clicking each resource name in the local project.
   Each copy will open in a separate tab of the editor.
5. Compare the changes from the server in NewDatafoundation to your local changes in Copy of
   NewDatafoundation.
6. In the editor tab with NewDatafoundation open, delete any server changes you do not want to keep,
   and add any changes that you made locally that you want to keep.
7. Save the changes to NewDatafoundation in the editor.
8. In the Project Synchronization View, refresh the synchronization. Update the server by selecting
   NewDatafoundation and then selecting **Save Changes on Server**.

   Eventually, once you have verified the merged changes, you can delete Copy of NewDatafoundation
   from the local project.

---

### 6.4 Saving resources as reports

You can save any resource in a local project as a report in a local file.

1. Right-click the resource name in the Local Projects View, and select **Save As**.
2. In the **Report Location** box, enter a file path, file name, and file type for the report. The file type
   can be .pdf, .html, or .txt.
   To browse the local file system to find a file path, click the browse button.
3. For larger resources (data foundations and business layers), you can select which metadata elements
   to include in the report in the **Metadata Elements** box.
4. Click **Generate** to create the report.
Working with repository resources

7.1 About managing repository resources

Repository resources are the universes and connections that have been secured in a repository on a Central Management System (CMS). The Repository Resources View lets you navigate and interact with the folders and resources in repositories.

The Connections folder contains the secured connections created using the information design tool and universe design tool.

Note:

The Universe folder contains universes published with the information design tool (.unx universes), as well as universes created and exported with the universe design tool or migrated from earlier versions (.unv universes).

To navigate a repository, open a session on the CMS where the repository is stored. For more information about sessions, see the related topic.

The following sections summarize the tasks you can do from the Repository Resources View.

Folder management
With appropriate rights, you can insert, rename, and delete sub-folders in the Connections and Universes folders.

Secured Connection management
- Edit an existing connection.
- Insert a new secured relational or OLAP connection in the repository.
- Create a connection shortcut in a local project from an existing secured connection.
- Delete a secured connection from the repository.

Universe management
You can do the following tasks on .unx universes (published using the information design tool):
- Run a check integrity.
- Run a query. This command opens the Query Panel. The security settings defined in the security profiles for the universe are applied according to the username in the session.
• Retrieve a universe. This command saves the business layer and its referenced resources in a local project so that you can edit them.
• Rename a universe. This command renames only the universe, and not the underlying business layer.
• Delete a universe from the repository.

You can do the following tasks on .unv universes (created using the universe design tool or migrated from earlier versions):
• Convert a universe. You can save the converted resources in a local project, or publish the converted .unx universe in the repository.
• Delete a universe from the repository.

Related Topics
• About session management
• About editing connections and connection shortcuts
• Creating a relational connection
• Creating an OLAP connection
• About connection shortcuts
• Running check integrity
• Running a query on a universe published in a repository
• Retrieving a published universe from a repository
• About converting .unv universes

7.2 About session management

A session contains the Central Management Server (CMS) system name and authentication information needed to access resources stored in a repository. You need at least one session defined to connect to a repository. You can define additional sessions that connect to the same repository as a different user.

Workflows in the information design tool that require access to secured resources prompt you with the “Open Session” dialog box. If you have not already defined a session for the repository you want to access, you can select New Session from the Sessions list. You can also define a session with the Insert Session command in the Repository Resources View.

Once the session is defined, it is retained in the Repository Resources View, and also on the Sessions list. The next time you open the session, you will only need to enter the password.

Once a session is opened it stays open until you exit the information design tool. To explicitly close a session, you must do so from the Repository Resources View.

Several sessions can be open at the same time, as long as the sessions are on different CMS systems. If you need to open a session with a different username and password on a CMS that has another session open, you must first close the open session.
If you no longer need a session and want to delete it from the list, use the **Delete Session** command in the Repository Resources View.

**Related Topics**
- CMC rights for information design tool users
- Opening a session
- Closing a session

### 7.2.1 Opening a session

Different workflows require you to open a session. If you are prompted to open a session, a list of predefined sessions is available. The **Sessions** list is organized in the following order:

- Open sessions in alphabetical order
- Closed sessions in alphabetical order
- **New Session**

To open a session on a repository already defined in the information design tool:

1. Do one of the following:
   - In the Repository Resources View, right-click the repository name and select **Open Session**.
   - Select the session in the **Sessions** list.

2. The authentication information for the CMS is filled in for you. If the session is not already open, enter your **Password**.

   **Note:**
   - If you try to open a session on a repository which has another session already open, an error message displays. To change sessions on a repository, you must first close the open session in the Repository Resources View.

3. Depending on the workflow, click **OK**, **Next**, or **Connect**.

To open a session on a repository not already defined in the information design tool:

1. Do one of the following:
   - In the Repository Resources View, from the **Insert** menu, select **Insert Session**.
   - Select **New Session** from the **Sessions** list.

2. In the **System** box, enter the name of the Central Management System where the repository is located.

   **Note:**
   - To insert a session for a repository that is hosted on a machine in a different domain than the client hosting the application, you need to provide the host information in a hosts file on the client. Update the hosts file is in the following location:
3. Enter your **User Name** and **Password**.

**Note:**
For **Authentication** type **Windows AD**, specify the complete domain name in the **User Name**. For example, enter myuser@domain.com instead of myuser@domain.

4. In the **Authentication** list, select the authentication method to use.
5. Depending on the workflow, click **OK**, **Next**, or **Connect**.

The session remains open until you explicitly close it in the Repository Resources View, or exit the information design tool.

**Related Topics**
* Closing a session

### 7.2.2 Closing a session

All open sessions close when you exit the information design tool. To explicitly close a session:
1. In the Repository Resources View, select the session you want to close.
2. Right-click and select **Close Session**.
Working with connections

8.1 About connections

A connection is a named set of parameters that define how one or more SAP BusinessObjects applications can access relational or OLAP data sources. The connection can be a local file, or a remote object in a repository that is referenced by a local shortcut in the information design tool.

You use connections for the following purposes:

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relational data source for a data foundation</td>
<td>You associate one or more relational connections to a data foundation, and build a business layer on the data foundation. When you publish the business layer as a universe, the connections and data foundation are integrated in the universe and provide the data for queries run against the universe.</td>
</tr>
<tr>
<td>OLAP data source for a business layer</td>
<td>For an OLAP data source, you associate a business layer directly to a connection. The business layer is published as a universe, but the connection provides direct access to the cube.</td>
</tr>
<tr>
<td>Access to an SAP NetWeaver BW BEx Query</td>
<td>You define SAP NetWeaver BW connections that use the SAP BICS Client middleware driver to provide access to a BEx Query. SAP BusinessObjects query and reporting applications connect directly to the BEx Query. You cannot use these connections as a source for business layers or universes. For information on how to build a universe on SAP NetWeaver BW, see the related topic.</td>
</tr>
</tbody>
</table>

Connections can be either local or secured.

Related Topics

- About local connections
- About secured connections
8.1.1 About local connections

You create local connections in the information design tool local project. Local connections are saved as independent objects on the local file system as .cnx files.

Local connections are used for the following purposes:

- To access relational data sources when authoring a data foundation and relational business layer.
  
  **Note:**
  To create a multisource-enabled data foundation, you must reference secured connections.

- To access an OLAP cube when authoring an OLAP business layer.
- To run queries on a target database to test modifications in the business layer, or to build lists of values.

Local connections have limited or no security as they can be used by any user with access to the machine running the information design tool.

To secure a local connection, you publish the connection to a repository.

**Related Topics**

- Publishing a local connection to the repository
- About secured connections
- Creating a relational connection
- Creating an OLAP connection

8.1.2 About secured connections

A secured connection is a connection that has been created in, or published to, a repository. It is stored in a dedicated Connections folder in the repository. You can create subfolders in the Connections folder to organize the storage of connections in the repository.

When a connection is published, a connection object containing the same parameters as the local connection is created in the Connections folder or subfolder in the repository.
You can also create secured connections directly in the repository using the **Insert Relational Connection** and **Insert OLAP Connection** commands from the Repository Resources View.

Secured connections cannot be copied to the local file system, but are made available in the Local Projects View as connection shortcuts. The shortcut can be used in the same way as a local connection. However, the connection properties can only be modified by connecting to the repository system.

Secured connections and connection shortcuts are used for the following purposes:

- To retrieve data for universes published to a repository.
- To retrieve data for SAP BusinessObjects reporting products directly accessing database middleware.
- As a data source when creating a data foundation or OLAP business layer.

A secured connection is subject to the following general security constraints in the repository:

- Users must be authenticated.
- User rights can be defined at the user level to grant or deny access to connections or connection properties.
- Connections can only be shared and used by authenticated users.

**Downloading relational connections locally**

To maintain confidentiality, some sensitive secured connection parameters, for example username and password, remain stored in the repository.

In order to edit the connection in the information design tool, the "Download connection locally" right must be granted in the Central Management Console (in addition to the "Create, modify, or delete connections" application right and the "Edit objects" connection right).

If the "Download connection locally" right is granted, you can choose to run queries on the server using the server middleware driver, or locally using the local middleware driver. To use the local middleware, select the local middleware option in the information design tool preferences. If this right is denied, the information design tool uses the server middleware.

**Related Topics**

- About connection shortcuts
- CMC rights for information design tool users
- Setting middleware for secured relational connections

### 8.1.3 About connection shortcuts

A connection shortcut is an object that references a secured connection in a repository. The shortcut is saved as a .cns file on the local file system. The shortcut contains the repository address and port number, the type of connection (OLAP or relational), and an ID that identifies the connection on the server.

You use a connection shortcut when authoring, or modifying any data foundation or business layer that uses a connection stored in the repository.
You can create a connection shortcut in two ways:

- Publish a local connection to the repository.
- Create a shortcut from an existing secured connection in the Repository Resources View.

**Related Topics**
- Publishing a local connection to the repository
- Creating a connection shortcut

### 8.2 Creating a relational connection

You use the New Relational Connection wizard to create local and secured connections to a relational data source.

Before you create a local connection, you must have a project available in the Local Projects View. For more information on creating local projects, see the related topic.

**Note:**
You must create relational connections to SAP NetWeaver BW and SAS sources directly in the repository as secured connections.

1. Do one of the following:
   - To create a local connection, select the project folder in the Local Projects View. Select **File > New > Relational Connection**.
   - To create a secured connection, in the Repository Resources View, open a session on the repository where you want to create the secured connection. Right-click the Connections folder or subfolder in the repository, and select **Insert Relational Connection**.

2. Follow the steps in the New Relational Connection wizard to enter the following information:
   - Name for the connection
   - Middleware for the target database
   - Login parameters to connect to the relational data source
   - Configuration and custom parameters to optimize the connection

   If you need help on a particular step, click the help icon in the wizard dialog box.

**Related Topics**
- Creating a local project
- About managing repository resources
- Name a connection
- About connections
8.2.1 Name a connection

This section describes the Resource name page of the New Relational Connection wizard.

You name a connection and can enter a description of the data source. The name and description are available as properties of the connection and can be edited at any time.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource name</td>
<td>Connection name. This field is mandatory.</td>
</tr>
<tr>
<td>Description</td>
<td>Information describing data source. This information can be useful when the connection is used for multiple data foundations. This is optional information.</td>
</tr>
</tbody>
</table>

When you have entered name information, click Next to continue the wizard.

Related Topics
• Select a middleware driver

8.2.2 Select a middleware driver

You select a connection driver to connect to the correct middleware version for the target database. The connection driver is the SAP BusinessObjects driver that maps information in the middleware to the user interface of the SAP BusinessObjects application.

Expand the database and middleware node for the target database and select the connection driver. Click Next to continue the wizard.

Note:
• SAP NetWeaver BW and SAS connections are only listed if you are creating the connection directly in the repository.
• If you are using the information design tool from a Crystal Server 2011 installation, SAP middleware drivers are not available.

Related Topics
• Set the connection parameters
8.2.3 Set the connection parameters

The connection parameters vary depending on the type of data source for which you are defining the connection. Select from the related topics the link to more information about the connection parameters.

**Related Topics**
- Login parameters for relational connections
- Login parameters for SAP NetWeaver BW and ERP connections
- Login parameters for SAS connections
- Login parameters for Oracle EBS connections
- Configuration parameters for relational connections
- Custom parameters for relational connections

8.2.3.1 Login parameters for relational connections

The following login parameters apply to most relational connections.

For a description of login parameters for SAP NetWeaver BW, SAP ERP, SAS, and Oracle EBS connections, see the related topics.
**Authentication Mode**

The method used to authenticate the user's login credentials when accessing the data source:

- **Use specified user name and password**: Uses the **User Name** and **Password** parameters defined for the connection.

- **Use BusinessObjects credential mapping**: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

- **Use Single Sign On**: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Authentication Mode** | The method used to authenticate the user's login credentials when accessing the data source:  

- **Use specified user name and password**: Uses the **User Name** and **Password** parameters defined for the connection.  

- **Use BusinessObjects credential mapping**: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

- **Use Single Sign On**: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*. |
| **User Name**   | The user name to access the data source if **Authentication Mode** is **Use specified user name and password**.                                    |
| **Password**    | The password to access the data source if **Authentication Mode** is **Use specified user name and password**.                                   |
| **Server (<host>:<port>)** | The name and port of the server hosting the data source.  

For connections to the SAP HANA appliance, the HANA instance number is represented by the second and third digits of the port number. For example, if the port number is 30215, the HANA instance number is 02.  

To take advantage of the SAP HANA failover option, specify a list of servers separated by a semi-colon, for example: (host1:30015;host2:30015;host3:30015). The JDBC driver will choose one of these hosts for connecting. If a host is not available, the driver chooses the next host from the list.  

For connections to Oracle, you can enter a list of servers separated by commas: (<host1>:<port1>, <host2>:<port2>). |
| **Server**      | The name of the server hosting the data source.                                                                                             |
The database name.

Data Source Name
For ODBC connections, the name of the data source that you have defined using your operating system's data source manager.

Alias
For DB2 connections, this is the alias of the database that you have created in the DB2 configuration assistant.

Net Service
For Oracle connections using JDBC middleware, the Oracle Net Service name.

Service
For Oracle connections, the alias that contains the server IP and Net Service information.

JDBC_URL
JDBC_CLASS
For generic JDBC connections, the JDBC URL and class used to connect to the database.

Informix Server
For Informix connections, the name of the Informix server that you have defined.

OLE DB Provider Name
For generic OLE DB providers, the provider name.

Related Topics
- Login parameters for SAP NetWeaver BW and ERP connections
- Login parameters for SAS connections
- Login parameters for Oracle EBS connections

8.2.3.2 Login parameters for SAP NetWeaver BW and ERP connections

The following parameters apply to connections to SAP NetWeaver BW (relational and BICS Client connections), and SAP ERP.

To set ABAP function and InfoSet parameters for SAP ERP connections, after entering the login parameters, click Next.
### Parameter | Description
---|---
**Authentication Mode** | The method used to authenticate the user's login credentials when accessing the data source:
- **Use specified user name and password**: Uses the **User Name** and **Password** parameters defined for the connection.
- **Use BusinessObjects credential mapping**: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.
- **Use Single Sign On**: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source.
  
  For more information on single sign on, see the *SAP BusinessObjects Business Intelligence platform Administrator Guide*.

**Client Number** | The number used to identify the client on the SAP NetWeaver BW system.

**User Name** | The user name to access the data source if **Authentication Mode** is **Use specified user name and password**.

**Password** | The password to access the data source if **Authentication Mode** is **Use specified user name and password**.

**Language** | The 2-character ISO language code for the language to be used for the connection to the data source. For example, EN for English.

**Note:**
In some cases, select the language from the list.

**Save Language** | Specifies which language to use for the connection:
- If you select the **Save Language** option, the value from the **Language** parameter is used.
- If you clear **Save Language**, the value from the user's session (Preferred Viewing Locale) is used.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System ID</strong></td>
<td>The 3-character SAP system ID.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Required for both application and message server types.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> For a successful connection to the message server, you need to add the message server system ID to the following file on the machine hosting the application:</td>
</tr>
<tr>
<td></td>
<td>C:\WINDOWS\system32\drivers\etc\services</td>
</tr>
<tr>
<td></td>
<td>At the end of the existing file, add the line:</td>
</tr>
<tr>
<td></td>
<td>sapmsXXX &lt;tab&gt; 3601/tcp</td>
</tr>
<tr>
<td></td>
<td>where sapms means SAP message server, xxx is the is the System ID of the server that is used, and 3601/tcp is the default TCP port used for communication.</td>
</tr>
<tr>
<td><strong>Server type</strong></td>
<td>· Select <strong>Application Server</strong> to connect directly to the SAP server without using load balancing.</td>
</tr>
<tr>
<td></td>
<td>· Select <strong>Message Server</strong> to benefit from SAP load balancing capabilities.</td>
</tr>
<tr>
<td><strong>Server Name for Application Server</strong></td>
<td>The name of the SAP application server.</td>
</tr>
<tr>
<td><strong>System Number for Application Server</strong></td>
<td>The system number of the SAP application server. This is a two-digit integer between 00 and 99.</td>
</tr>
<tr>
<td><strong>Server Name for Message Server</strong></td>
<td>The name or IP address of the SAP message server used for load balancing.</td>
</tr>
<tr>
<td><strong>Group Name for Message Server</strong></td>
<td>Name of the Logon group; a set of dedicated application servers used to log on.</td>
</tr>
<tr>
<td><strong>Infoprovieder</strong></td>
<td>For SAP NetWeaver BW relational connections, the name of the InfoCube or MultiProvider to be used as the fact table in the center of the snowflake schema in the data foundation.</td>
</tr>
</tbody>
</table>
8.2.3.2.1 InfoProvider fact table selection

When you create an SAP NetWeaver BW connection, the “Select InfoProvider Fact Table” dialog box lets you choose a fact table that will become the center of the snowflake schema in your data foundation.

The filter button lets you filter by types of InfoProvider.

8.2.3.2.2 ABAP function and InfoSet parameters for ERP connections

The following parameters apply to SAP ERP connections. For more information about SAP ERP connections, see the *Data Access Guide*.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Function Name Wildcard</strong></td>
<td>Wildcards are filters that reduce the number of tables exposed in the connection. The wildcard character is <em>, and it matches 0 to any number of characters. The wildcard character can be used with keywords. For example: <em>keyword_one</em>keyword2</em> The above wildcard will expose only tables containing keyword_one followed by keyword_two.</td>
</tr>
<tr>
<td><strong>Map Table Parameters into Input Columns</strong></td>
<td>If selected, the table parameters are considered as both input and output parameters of the ABAP function. If unselected, the table parameters are considered as output only.</td>
</tr>
</tbody>
</table>
If selected, any selection field in the SAP Query is mapped into a table column and considered as an optional input column:

- A query can contain only the EQUAL filter in this column
- If the column is only in the projection, NULL is returned

If unselected, the selection fields are ignored. No filtering is possible on these fields.

### 8.2.3.3 Login parameters for SAS connections

The following parameters apply to connections to SAS data sources.

To include access to multiple data sets that are not pre-defined to the SAS/SHARE server, after entering the login parameters, click **Next**.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication Mode</td>
<td>The method used to authenticate the user's login credentials when accessing the data source:</td>
</tr>
<tr>
<td></td>
<td>- <strong>Use specified user name and password</strong>: Uses the <strong>User Name</strong> and <strong>Password</strong> parameters defined for the connection.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Use BusinessObjects credential mapping</strong>: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the <em>SAP BusinessObjects Business Intelligence platform Administrator's Guide</em>.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Use Single Sign On</strong>: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the <em>SAP BusinessObjects Business Intelligence platform Administrator Guide</em>.</td>
</tr>
</tbody>
</table>
### Parameter Table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>The user name to access the data source if Authentication Mode is Use specified user name and password.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to access the data source if Authentication Mode is Use specified user name and password.</td>
</tr>
<tr>
<td>Host Name</td>
<td>The host name of the server where the SAS/SHARE is running.</td>
</tr>
<tr>
<td>Port</td>
<td>The port to which to connect.</td>
</tr>
<tr>
<td>Catalog</td>
<td>The name used to identify the connection to the query server. <strong>Note:</strong> A default catalog name is registered automatically with the query server the first time the connection is added to any multisource-enabled data foundation.</td>
</tr>
</tbody>
</table>

### 8.2.3.3.1 Setting SAS data sets

When you create an SAS connection, the "Set SAS Data Sets" dialog box lets you configure the data foundation to access multiple data sets that are not pre-defined to the SAS/SHARE server. These are data sets that are not included in the current SAS configuration.

1. Select the **Use data sets that are non pre-defined to the SAS/SHARE server** option.
2. Click **Add** and, in the **Location** field, enter the path to the data set in the format required for the operating system that you are using.
3. In the **Library Name** field, enter a name to use to refer to the data set.
4. Click **Add** to add other data sets if needed.
5. Click **Finish**.

### 8.2.3.4 Login parameters for Oracle EBS connections

The following login parameters apply to Oracle EBS connections.
The user name to access the Oracle database server.

**Parameter** | **Description**
--- | ---
UserName | The user name to access the Oracle database server.
Password | The password to access the Oracle database server.
Service | The Oracle service name.

**Authentication Mode**

The method used to authenticate the user's login credentials when accessing the EBS application:

- **Use specified user name and password**: Uses the Oracle EBS User and Oracle EBS Password parameters defined for the connection.
- **Use Single Sign On**: Uses the credentials associated with the user's account defined on the Central Management Server (CMS) when users log into the SAP BusinessObjects BI platform using Oracle EBS user name and password. For more information, see the SAP BusinessObjects Business Intelligence platform Administrator's Guide.

Oracle EBS User | The user name to access the application if Authentication Mode is Use specified user name and password.
Oracle EBS Password | The password to access the application if Authentication Mode is Use specified user name and password.
Language | The application language.
Application | The application name.
Security Group | The Oracle security group.

### 8.2.3.5 Configuration parameters for relational connections

The **Configuration Parameters** dialog box contains parameters that you can set to override default configuration options.

The following configuration parameters apply to most relational connections.
<table>
<thead>
<tr>
<th><strong>Parameter</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
</table>
### Parameter | Description
--- | ---
Add Jar(s) | **Restriction:** Specific to connections to Apache Hadoop HIVE data sources. The paths to external JAR files to be added to the Java classpath. This parameter corresponds to the `add JAR` HIVE command line. You can define the paths to more than one JAR file, separated by semicolons. For example:

/usr/lib/hive/hive-contrib-1.jar;/usr/lib/hive/hive-contrib-2.jar

Only UNIX style paths are valid.

JDBC Driver Properties (key=value,key=value) | Values for JDBC driver properties. You can define the value of more than one property, separated by commas. For example, the following value for **JDBC Driver Properties** sets the oracle.jdbc.defaultNChar and defaultNChar driver properties:

oracle.jdbc.defaultNChar=true,defaultNChar=true

**Note:** If a property is defined in the `<driver>.sbo` file, the value defined in this parameter is used. For more information about SBO files, see the *Data Access Guide*.

Owner Name | For DB2 connections, this parameter adds the name of the owner of the table as a prefix on the table name, to match the DB2 convention for naming tables.

Table Suffix | For DB2 connections, this parameter adds a suffix on the table name, to match the DB2 convention for naming tables.

#### 8.2.3.6 Custom parameters for relational connections

The Custom Parameters dialog box lets you override the value of certain parameters. You can also add parameters and their values.
### ConnectInit

The value is added to the SQL and is run once when a user connects to the database.

### Hint

For Oracle connections, the value is used by the Oracle query optimizer to choose an execution plan. Consult your Oracle documentation for full information on the Hints that can be used, and how they can be used to optimize queries.

#### 8.2.3.7 Login and schema parameters for CSV file connections

The following parameters apply to connections to Comma-Separated Value (CSV) files. See the related topic for more information about file format and regional settings for CSV file connections.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location Type</td>
<td>The Location Type and Protocol parameters guide you through setting the necessary parameters for your data source.</td>
</tr>
<tr>
<td>Protocol</td>
<td>If Location Type is set to Local, the protocol and login credential parameters are grayed.</td>
</tr>
<tr>
<td></td>
<td>If Location Type is set to Remote, select a Protocol to make the appropriate login parameters available.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Schema files can be local even if Location Type is set to Remote. If the schema file is remote, the Protocol applies to both the data source and schema files.</td>
</tr>
<tr>
<td><strong>Parameter</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| **Datasource**     | **Filepath or Pattern**  
The path to a single CSV file or the path to a folder containing multiple CSV files. Files can be local or remote (HTTP, FTP, and SMB). If remote, the datasource is the location URL. Paths with MS Windows or UNIX styles are valid. Wildcards can be used.  
For example:  
- `C:\report.csv` for a single file  
- `C:\CSVFiles\` or `C:\CSVFiles\*.csv` for multiple files  
- `/home/user/csvfiles/report.csv` for a single file located on a UNIX machine  
- Remote locations:  
  - `http://host:port/path/file`  
  - `ftp://host:port/path/file`  
  - `smb://server:port/path/file` |
| **Schema Detection** | The method used to detect the schema of the CSV file. Possible values are:  
  - **auto**  
    The data access driver finds the schema automatically. **FileType** must be set to **delimited**.  
  - **no detection**  
    The data access driver skips comment lines, analyzes the first line, and determines the number of columns, but not the column types. **FileType** must be set to **delimited**.  
  - **ddl**  
    The data access driver uses a Data Definition Language (DDL) file to detect the schema.  
  - **sqlddl**  
    The data access driver uses a DDL file corresponding to the Standard SQL to detect the schema. |
| **Schema File**     | The path to one DDL or SQLDDL schema file. If you want to define schemas for several tables, use an SQLDDL file.  
Required if **Schema detection** is set to **ddl** or **sqlddl**.  
If the schema file is remote, the protocol of the file must match the **Protocol** parameter setting. |
### Parameter | Description
--- | ---
#### Probe Rows
- **auto**
  - The driver parses the file until a type is detected for each column. If there is no NULL value in the first row, the parsing terminates after the first row.
  - **Caution:** This method may lead to type conversion conflicts in the case where only the first rows are parsed and other types are used in the next rows.
- **int**
  - Parses a specific number of rows. This setting can be used as a trade-off between scalability for big CSV files and a low data quality of CSV files.
  - **Caution:** This method may lead to conflicts if other types are used in the rows that are not parsed.
- **all**
  - Parses the whole file. This method allows you to find the longest string value, which corresponds to the column size for non-numeric values.
  - **Note:** The recommended method is **all** to allow for correct detection of column information. Since this is the slowest detection method, use DDL files if CSV files are large.

#### Number of Probe Rows
- The number of CSV file lines that are parsed to check the column types.

  Required if **Probe Rows** is set to **int**.

#### Credentials
- The **User Name** and **Password** to access the CSV files in a remote connection.
- If **Protocol** is set to **SMB (Windows Share)**, enter the **SMB Domain** for the connection.
If **Protocol** is set to **HTTP**, enter the proxy parameters for the connection.

**Proxy Address**: The path to the HTTP or FTP proxy server (**host**:**port**).

For example: **myproxy.com**:8080

**Proxy User Name**: The user name used to access the proxy server.

**Proxy Password**: The password used to access the proxy server.

---

**Related Topics**

- [File format and regional settings for CSV file connections](#)

---

### 8.2.3.8 File format and regional settings for CSV file connections

The following parameters apply to connections to Comma-Separated Value (CSV) files.

**File Settings**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>File Charset</strong></td>
<td>The character set used in the CSV files.  &lt;br&gt;<strong>Note</strong>:  &lt;br&gt;All files must have the same character set.</td>
</tr>
<tr>
<td><strong>File Type</strong></td>
<td>The file type can be one of the following:  &lt;br&gt;• <strong>delimited</strong>  &lt;br&gt;The CSV file entries are delimited by a separator.  &lt;br&gt;• <strong>fixed</strong>  &lt;br&gt;The CSV file entries have fixed width.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Lenient Mode</td>
<td>If selected, invalid rows are skipped automatically (not enough or too many columns).</td>
</tr>
<tr>
<td>Column Names in First Line</td>
<td>This Boolean value tells if the first row of the CSV file contains column names.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If the file does not contain any column names and if schema detection is enabled, then</td>
</tr>
<tr>
<td></td>
<td>the data access driver names columns as col1, col2, ..., coln.</td>
</tr>
<tr>
<td>Union to One Table</td>
<td>This Boolean value indicates if tables are automatically concatenated when a pattern is given as</td>
</tr>
<tr>
<td></td>
<td>data source.</td>
</tr>
<tr>
<td></td>
<td><strong>For example,</strong> if the data source pattern is report_<em>.</em>.csv, then the driver concatenates all</td>
</tr>
<tr>
<td></td>
<td>the tables from CSV files that match the pattern.</td>
</tr>
<tr>
<td></td>
<td><strong>Caution:</strong> CSV files must have the same structure.</td>
</tr>
<tr>
<td>Number of Comment Lines in</td>
<td>The number of rows that contain comments at the beginning of the CSV file. Maximum is 1000.</td>
</tr>
<tr>
<td>the Beginning</td>
<td></td>
</tr>
<tr>
<td>Separator</td>
<td>A character that is used to separate CSV file entries. It must be different from text qualifier</td>
</tr>
<tr>
<td></td>
<td>and escape character.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If the tab key is used to separate entries, the word <strong>TAB</strong> can be set as separator.</td>
</tr>
<tr>
<td>Text Qualifier</td>
<td>A character that surrounds a file entry, for example quote (') or double quote (&quot;).</td>
</tr>
<tr>
<td></td>
<td><strong>If you do not want to use a text qualifier, use a character that is not used in the CSV file to</strong></td>
</tr>
<tr>
<td></td>
<td><strong>prevent the data access driver from using the default value.</strong></td>
</tr>
</tbody>
</table>
### Regional Settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decimal Separator</td>
<td>Default value is period (.). For example: 100.20.</td>
</tr>
<tr>
<td>Thousands Grouping Character</td>
<td>Default value is comma (,). For example: 1,000.20.</td>
</tr>
<tr>
<td>Date Format</td>
<td>Date and time formats used in the CSV files. They must match the format in the CSV files to enable the driver to recognize the date and time formats and to parse them.</td>
</tr>
<tr>
<td>Timestamp Format</td>
<td>Default values are:</td>
</tr>
<tr>
<td></td>
<td>* yyyy-MM-dd for dates</td>
</tr>
<tr>
<td></td>
<td>* yyyy-MM-dd HH:mm:ss for timestamps</td>
</tr>
<tr>
<td>Time Format</td>
<td>* HH:mm:ss for times</td>
</tr>
</tbody>
</table>

### 8.3 Creating an OLAP connection

You use the New OLAP Connection wizard to create local and secured connections to an OLAP data source.

Before you create a local connection in the information design tool, you must have a project available in the Local Projects View. For more information on creating local projects, see the related topic.

**Note:**
The OLAP connections you create in the information design tool are not supported by the universe design tool. Also, OLAP connections created in the universe design tool are not available for building universes in the information design tool.

1. Do one of the following:
To create a local connection, select the project folder in the Local Projects View. Select **File > New > OLAP Connection.**

To create a secured connection, in the Repository Resources View, open a session on the repository where you want to create the secured connection. Right-click the Connections folder or subfolder in the repository, and select **Insert OLAP Connection.**

2. Follow the steps in the New OLAP Connection wizard to enter the following information:
   - Name for the connection
   - Middleware driver for the target database
   - Authentication parameters to connect to the OLAP data source
   - OLAP cube to connect to

   If you need help on a particular step, click the help icon in the wizard dialog box.

**Related Topics**
- Creating a local project
- Select an OLAP middleware driver
- About connections

### 8.3.1 Select an OLAP middleware driver

This section describes the OLAP driver selection page of the New OLAP Connection wizard.

You select an OLAP driver to connect to the OLAP server. The OLAP driver maps information from the OLAP server middleware to the user interface of the SAP BusinessObjects application.

Depending on your target OLAP server, expand the middleware node, and select the target driver.

**Note:**
- If you are using the information design tool from a Crystal Server 2011 installation, SAP middleware drivers are not available.

### 8.3.2 Set login parameters for OLAP data sources

The connection parameters vary depending on the type of data source for which you are defining the connection. Select from the related topics the link to more information about the connection parameters.

**Related Topics**
- Login parameters for OLAP connections
- Login parameters for SAP NetWeaver BW and ERP connections
8.3.2.1 Login parameters for OLAP connections

The following parameters apply to most OLAP connections.

For a description of login parameters for SAP NetWeaver BW (BICS Client), see the related topic.

<table>
<thead>
<tr>
<th>Login parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authentication mode</strong></td>
<td>The method used to authenticate the user's login credentials when accessing the data source:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Use specified user name and password</strong>: Uses the <strong>User Name</strong> and <strong>Password</strong> parameters defined for the connection.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Use BusinessObjects credential mapping</strong>: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the <em>SAP BusinessObjects Business Intelligence platform Administrator's Guide</em>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Use Single Sign On</strong>: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the <em>SAP BusinessObjects Business Intelligence platform Administrator Guide</em>.</td>
</tr>
<tr>
<td><strong>Server</strong></td>
<td>For MSAS connections, the URL path, for example: http://&lt;server_name&gt;/olap_2005/msmdpump.dll</td>
</tr>
<tr>
<td></td>
<td>For Essbase connections, the server name for the data source.</td>
</tr>
<tr>
<td><strong>User Name</strong></td>
<td>The user name to use to access the OLAP server if the <strong>Authentication mode</strong> is <strong>Use specified user name and password</strong>.</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>The password to use to access the OLAP server if the <strong>Authentication mode</strong> is <strong>Use specified user name and password</strong>.</td>
</tr>
<tr>
<td><strong>Language</strong></td>
<td>The language that will be used for the connection.</td>
</tr>
</tbody>
</table>
8.3.2.2 Login parameters for SAP NetWeaver BW and ERP connections

The following parameters apply to connections to SAP NetWeaver BW (relational and BICS Client connections), and SAP ERP.

To set ABAP function and InfoSet parameters for SAP ERP connections, after entering the login parameters, click Next.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication Mode</td>
<td>The method used to authenticate the user's login credentials when accessing the data source:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Use specified user name and password</strong>: Uses the User Name and Password parameters defined for the connection.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Use BusinessObjects credential mapping</strong>: Uses the database credentials associated with the user's account defined on the Central Management Server (CMS) to connect to the data source. The database credentials are set in the User's Properties in the Central Management Console. For more information, see the SAP BusinessObjects Business Intelligence platform Administrator's Guide.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Use Single Sign On</strong>: This authentication mode is used to support end-to-end single sign on defined in the Central Management Server (CMS). If you use an external authentication source (for example LDAP), the CMS and the data source must be configured to use this external authentication source. For more information on single sign on, see the SAP BusinessObjects Business Intelligence platform Administrator Guide.</td>
</tr>
<tr>
<td>Client Number</td>
<td>The number used to identify the client on the SAP NetWeaver BW system.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name to access the data source if Authentication Mode is <strong>Use specified user name and password</strong>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>The password to access the data source if <strong>Authentication Mode</strong> is <strong>Use specified user name and password</strong>.</td>
</tr>
<tr>
<td><strong>Language</strong></td>
<td>The 2-character ISO language code for the language to be used for the connection to the data source. For example, EN for English. <strong>Note:</strong> In some cases, select the language from the list.</td>
</tr>
<tr>
<td><strong>Save Language</strong></td>
<td>Specifies which language to use for the connection: • If you select the <strong>Save Language</strong> option, the value from the <strong>Language</strong> parameter is used. • If you clear <strong>Save Language</strong>, the value from the user's session (Preferred Viewing Locale) is used.</td>
</tr>
<tr>
<td><strong>System ID</strong></td>
<td>The 3-character SAP system ID. <strong>Note:</strong> Required for both application and message server types. <strong>Note:</strong> For a successful connection to the message server, you need to add the message server system ID to the following file on the machine hosting the application: C:\WINDOWS\system32\drivers\etc\services At the end of the existing file, add the line: sapmsXXX &lt;tab&gt; 3601/tcp where sapms means SAP message server, xxx is the is the System ID of the server that is used, and 3601/tcp is the default TCP port used for communication.</td>
</tr>
<tr>
<td><strong>Server type</strong></td>
<td>• Select <strong>Application Server</strong> to connect directly to the SAP server without using load balancing. • Select <strong>Message Server</strong> to benefit from SAP load balancing capabilities.</td>
</tr>
<tr>
<td><strong>Server Name for Application Server</strong></td>
<td>The name of the SAP application server.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>System Number for Application Server</strong></td>
<td>The system number of the SAP application server. This is a two-digit integer between 00 and 99.</td>
</tr>
<tr>
<td><strong>Server Name for Message Server</strong></td>
<td>The name or IP address of the SAP message server used for load balancing.</td>
</tr>
<tr>
<td><strong>Group Name for Message Server</strong></td>
<td>Name of the Logon group; a set of dedicated application servers used to log on.</td>
</tr>
<tr>
<td>Infopresenter</td>
<td>For SAP NetWeaver BW relational connections, the name of the InfoCube or MultiProvider to be used as the fact table in the center of the snowflake schema in the data foundation.</td>
</tr>
<tr>
<td>Catalog</td>
<td>For SAP NetWeaver BW relational connections, the name used to identify the connection to the query server. Note: A default catalog name is registered automatically with the query server the first time the connection is added to any multisource-enabled data foundation.</td>
</tr>
</tbody>
</table>

### 8.3.3 Select an OLAP cube

The following options apply to associating a cube with the OLAP connection.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Do not specify a cube in the connection</strong></td>
<td>Select this option to create the connection without specifying a cube. In this case, each time you access the connection, either when building a business layer or in a query and reporting tool, you will be prompted to select a cube.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Specify a cube in the connection | Select this option to always associate a cube with the connection. The cube selection page lists the cubes available for the target database. You can enter a search string in the search text box. Select the cube in the list.  
**Note:** For connections using the SAP BICS Client driver, select the Specify a cube in the connection option. In the Information Area folders, select an InfoCube. Then select a BEx Query for the connection. |
To edit a local connection, or a connection shortcut, double-click the connection or shortcut name in the Local Projects View.

To edit a secured connection, in the Repository Resources View, open a session on the repository where the connection is published. In the Connections folder or subfolder, double-click the connection name.

In the Connection Editor you can do the following tasks depending on the type of connection:

**Relational connections**

In the "General Information" pane:
- To edit the connection parameters, click **Edit**.
- To change the middleware driver, click **Change Driver**.
- To test the availability of the database server, click **Test Connection**.

The "Connection Definition" pane shows the information stored about the connection.

The "Show Values" pane lets you browse the values in the tables referenced by the connection. For more information, see the related topic.

**OLAP connections**

In the "General Information" pane:
- To edit the connection login parameters and cube selection, click **Edit**.
- To test the availability of the cube server, click **Test Connection**.

The "Connection Definition" pane shows the information stored about the connection.

The "Show Values" pane lets you browse the objects in the cube and their properties.

The "Query" pane lets you drag and drop cube objects to create an MDX query and run the query on the cube.

**Connection shortcuts**

In the "General Information" pane:
- To change the secured connection that the shortcut references, click **Change Connection**. Open a session on the repository where the connection you want to change to is published.
- To test the availability of the database or cube server, click **Test Connection**.

**Related Topics**
- **Showing and profiling values in a data source**
Working with data foundations

9.1 About data foundations

A data foundation contains a schema of relevant tables and joins from one or more relational databases that are used as a basis for one or more business layers.

You reference relational connections in the data foundation. You insert tables and joins from the databases referenced in the connections.

Using the Data Foundation Editor, you can enhance the data foundation by adding derived tables, alias tables, calculated columns, additional joins, contexts, prompts, and lists of values. The availability of some features depends on the data foundation type. See the related topic for more information about data foundation types.

You can build any number of business layers on the same data foundation. In this case the data foundation becomes the basis for multiple universes.

Related Topics
- About data foundation types
- How to build a data foundation

9.1.1 About data foundation types

Single-source and multisource-enabled are two types of data foundations that allow you to take advantage of different data foundation features.

Related Topics
- About single-source data foundations
- About multisource-enabled data foundations
9.1.2 About single-source data foundations

Single-source data foundations support a single connection. The connection can be local or secured, which means you can publish universes based on the data foundation either locally or to a repository.

Single-source data foundations support database-specific SQL syntax for derived tables, calculated columns, and join expressions. Database-specific SQL syntax allows functions or operators that are offered by a specific database and not by standard SQL-92 (for example, Oracle analytical functions). You must select single-source if you want to publish to a local folder the universes that are based on this data foundation.

Single-source data foundations are recommended for the following situations:

- You want to work exclusively with database-specific SQL syntax.
- You want to publish the universe locally and work outside of a repository.

Related Topics
- About multisource-enabled data foundations
- How to build a data foundation
- Changing a connection in a data foundation

9.1.3 About multisource-enabled data foundations

Multisource-enabled data foundations support one or more connections. You can add connections when you create the data foundation and anytime later. Multisource-enabled data foundations only support secured connections, and universes based on this type of data foundation can only be published to a repository.

Multisource-enabled data foundations support most relational connections supported in single-source data foundations. In addition, multisource-enabled data foundations support the following relational connections that are not supported in single-source data foundations:

- SAP NetWeaver BW connections
- SAS connections

Connections for multisource-enabled data foundations are managed by the data federation service. For information about tuning the data federation service, see the Data Federation Administration Tool Guide.

SQL-92 standard syntax is the default for calculated columns, derived tables, and join expressions. In addition, the SAP BusinessObjects SQL functions are available. You can use database-specific SQL syntax in a multisource-enabled data foundation by defining a database-specific derived table or
calculated column. Database-specific SQL syntax allows functions or operators that are offered by a
specific database and not by standard SQL-92 (for example, Oracle analytical functions).

Multisource-enabled data foundations are required in the following situations:

- You want to insert tables and joins from more than one relational data source.
- You want to use SAP NetWeaver BW or SAS connections.
- You want to use SQL-92 standard syntax and SAP BusinessObjects SQL functions.

For more information on these situations, see the related topics.

**Related Topics**

- Data foundations with multiple connections
- SQL expressions in multisource-enabled data foundations
- About single-source data foundations
- How to build a data foundation
- Universes on SAP NetWeaver BW

### 9.1.3.1 Data foundations with multiple connections

To be able to add multiple connections to a data foundation, you must select the multisource-enabled
type when you create the data foundation.

You can select multiple connections when you create the data foundation. You can also add connections
to an existing multisource-enabled data foundation. Connections must be secured, and therefore
available in a repository. The connections are represented by a connection shortcut in the local project.

The connections in a multisource-enabled data foundation have the following additional properties:

- A short name used to identify the connection in the data foundation and to modify the table name
  in SQL expressions. You specify the short name when adding the connection. This name must be
  unique within the data foundation and is limited to forty characters. If you change the short name
  for the connection, the SQL expressions are automatically updated with the new name.
- A color for the connection. This color is used in the table header in data foundation views. You select
  the color when adding the connection. You can change the color for a connection at any time.
- A catalog used to identify the connection to the query server. A default catalog name is registered
  automatically with the query server the first time the connection is added to any multisource-enabled
  data foundation.
- For SAP NetWeaver BW connections, properties related to the automatic insertion of tables and
  joins. For more information on these properties, see the related topic.

In a multisource-enabled data foundation, the table name as it appears in SQL expressions has the
format: `<@catalog(short name)."database_qualifier.database_owner"."table_name">`

A multi-source join can be created between tables from different connections. You can use the **Detect Joins** command to detect joins between tables referenced in different connections, or explicitly define them with the **Insert Join** command.
9.1.3.2 SQL expressions in multisource-enabled data foundations

SQL expressions that define joins, calculated columns, and derived tables in a multisource-enabled data foundation use SQL-92 AINSI standard syntax.

In SQL-92 expressions, you can include SAP BusinessObjects SQL functions and @functions. Which @functions you can include depends on the type of expression. For more information, see the related topic.

In order to use functions or operators that are offered by the database and not by SQL-92 (for example, Oracle analytical functions), you define database-specific calculated columns and derived tables. An option in the SQL Expression Editor allows you to use database-specific SQL.

Database-specific calculated columns and derived tables support the SQL syntax of the associated connection. The following rules apply to database-specific SQL expressions:

- You can reference only standard tables and database-specific derived tables in a single connection.
- You cannot reference tables in SAS or SAP NetWeaver BW connections.
- You can include @functions with certain restrictions. For more information, see the related topic.

Related Topics
- About @Functions

9.2 About the Data Foundation Editor

This topic describes how to navigate the Data Foundation Editor. For steps to help you build the structure of your data foundation, see How to build a data foundation.

The Data Foundation Editor is divided into a data foundation view pane, a properties pane, and browsing panes.

The data foundation view is a graphical representation of the tables and joins. The Master view contains all tables and joins and cannot be deleted. You can define custom views that contain subsets of the tables. Access the views by the tabs at the bottom of the view pane. For more information about custom views, see the related topic.
The properties pane displays the properties of the data foundation object that is currently selected (the entire data foundation, a table, a column, or a join). To edit properties that apply to the entire data foundation, see the related topic.

In the data foundation view, you can work on tables and joins using commands in the Insert and Detect menus, or by clicking objects directly in the view.

The browsing panes allow you to work with different elements of the data foundation. Access the panes by clicking the corresponding tab:
- Connections
- Data Foundation (displays a tree view of the tables and joins)
- Aliases and Contexts
- Parameters and Lists of Values

For more information about what you can do in each of the browsing panes, see the related topic.

Navigating the data foundation view

To access a menu of commands that are available on tables, right-click the table header in the data foundation view. To select multiple tables, click the table headers while holding down the CTRL key.

To access commands that are available on columns, right-click the column name in the table in the data foundation view.

Several commands in the table right-click menu are available to help you locate related tables in the data foundation:
- Select Related Tables automatically selects all tables linked by joins to the selected table.
- Highlight Related Tables grays any table that is not linked to the selected table by a join.
- Highlight Aliases grays all tables except the selected original table and its alias tables.
- Highlight Original Table grays all tables except the selected alias table and the original table it is based on.
- Center on Selection lets you temporarily change the zoom on the data foundation display so that all the tables in a selection are visible in the display window.

You can also use the search panel to perform advanced searches on the data foundation. To open the search panel, click .

Related Topics
- Inserting a custom data foundation view
- About connections in the data foundation
- About contexts
- About parameters and lists of values in the data foundation
- About data foundation properties
- Searching for tables and columns in the data foundation
- Centering the view on a selection
9.3 How to build a data foundation

Before you begin:

- You need a local project in which to create the data foundation.
- In the local project, you need the relational connection or connection shortcuts to secured relational connections. Multisource-enabled data foundations require connection shortcuts.

For links to more detailed information on each step, see the Related Topics.

1. To start the New Data Foundation wizard, right-click the project folder in the Local Projects View and select New > Data Foundation.

   Follow the steps in the wizard to select the data foundation type and the connections.

   The data foundation is created in a .dfx file in the local project. It opens automatically in the Data Foundation Editor.

2. In the Data Foundation Editor, insert tables into the data foundation:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To insert tables from the connection</td>
<td>In the Connection pane, open and browse the tables in the connection. You have tools to search for and filter the tables in the Connection pane. Drag and drop the tables you want into the data foundation Master view.</td>
</tr>
<tr>
<td>To insert tables using a wizard</td>
<td>In the data foundation view tool bar, select Insert &gt; Insert Tables. You have the option to detect and insert keys, joins, cardinalities, and row counts automatically. Note: To detect joins between tables referenced by different connections, you need to use the Detect Joins command.</td>
</tr>
</tbody>
</table>

3. Insert joins:
### Option | Command
--- | ---
To insert joins manually | In the data foundation view, click the column name in the first table and drag it to the column in the second table. A join path appears between the two tables.
You can also insert joins by opening the “Edit Join” dialog box. In the data foundation view toolbar, select **Insert > Insert Join**.

To detect joins | In the data foundation view toolbar, select **Detect > Detect Joins**.

To edit a join, double-click the join path. For more information about editing and detecting joins, see the related topics.

4. Check the cardinality of the joins in the data foundation. In the data foundation view toolbar, select **Detect > Detect Cardinalities**.

   In the "Detect Cardinalities" dialog box, you can set or detect cardinality for any or all joins.

5. You can enhance the function of the data foundation in several ways, for example:
   - Insert calculated columns
   - Insert derived tables
   - Insert alias tables
   - Insert parameters with optional prompts
   - Insert lists of values to be associated with a prompt
   - Set SQL options and SQL generation parameters in the data foundation properties

6. Verify the join paths and resolve any loops. Use the commands in the **Aliases and Contexts** pane to detect aliases and contexts automatically.

7. Run an integrity check to validate the tables, columns, and joins in the data foundation. Right-click the data foundation name in the **Data Foundation** pane, and select **Check Integrity**.

8. Save the data foundation by clicking the **Save** icon in the main toolbar.

Some commands to help you maintain the data foundation are listed below.

   - If you change tables or columns, use **Show Local Dependencies** to find the business layers and objects that might be impacted by the changes.
   - Refresh the structure to update the data foundation with changes that have occurred to the databases referenced in the connections.
   - You can change a connection, or for multisource-enabled data foundations, add or remove a connection.
   - You can enhance the ease of maintenance by creating custom views, grouping tables using families, arranging the table display, and inserting comments.

**Related Topics**
- About data foundation types
- Creating a local project
- Creating a relational connection
- Creating a connection shortcut
9.4 About connections in the data foundation

Connections in the data foundation are listed in the "Connection" pane of the Data Foundation Editor. Some connections allow for multiple databases (called qualifiers), with different owners:

- Some data sources provide both qualifiers and owners (for example, MS SQL Server)
- Some data sources only provide qualifiers (for example, MySQL and text files)
- Some data sources only provide owners (for example, Oracle, SAP HANA, DB2, and Teradata)

For single-source data foundations, the "Connection" pane display is as follows:

- Qualifiers, if available, are listed under the connection
  - Owners, if available, are listed under each qualifier
  - Tables are listed under each owner
    - Columns are listed under each table

For multisource-enabled data foundations, the "Connections" pane display is as follows:
* Qualifier/Owners, known as the schema, are listed under the connection (or owners only, if the data source does not provide qualifiers).
  * Tables are listed under each schema
    * Columns are listed under each table

The list of tables is sorted alphabetically.

By default, tables are listed for all qualifiers and owners. To list only tables for the currently used qualifier/owner, click the "Show Qualifiers and Owners" icon to deselect it.

When the connection contains more than 1000 qualifier/owners or tables, the qualifier/owners or tables are grouped in packages of 1000. The packages are listed with the first several letters of the first and last qualifier/owner or table names in the package surrounded by brackets, for example:

* [AAAA....] - [MMMM]
* [NNNN...] - [ZZZZZ]

For tables that are already inserted into the data foundation, the table icon displays a green check mark: . You can insert a table into the data foundation by double-clicking the table name in the "Connections" pane. You can also search for tables in a connection to obtain a filtered list of tables. For more information, see the related topic.

Some connections have different table types (for example, in an SAP HANA connection, you can have several table types, including Analytic View and Calculation View). The Filter by Table Type icon lets you select table types to filter the list of tables displayed in the connection. For more information on filtering by table type, see the related topic.

**Operations on connections**

You can do the following tasks on connections from the "Connection" pane:

* **Change** lets you change the connection and its associated properties. For more information about this task, see the related topic.
* **Open** opens the connection or connection shortcut properties in the Connection Editor.
* **Test** lets you test if the database referenced by the connection is available.

In addition, for multisource-enabled data foundations, you can do the following tasks from the "Connections" pane:

* **Add Connections** lets you add connections to the data foundation. For more information about this task, see the related topic.
* **Remove** lets you remove a connection from the data foundation. The connection itself remains in the repository and registered with the catalog name.

**Related Topics**

* Searching for tables in the Connection pane
* About tables in the data foundation
* Filtering the tables in the connection by table type
* Changing a connection in a data foundation
9.4.1 Adding connections to a data foundation

To add connections to the data foundation, the following conditions are required:

- The data foundation type must be multisource-enabled.
- The connections to add must be relational, secured connections.
- For each connection to add, you must create a connection shortcut in the local project where the data foundation is stored.

1. Double-click the data foundation name in the local project to open the Data Foundation Editor.
2. In the Data Foundation Editor, click the **Connections** tab.
3. In the "Connections" pane, click the **Add Connections** icon 🔄.
   The "Add Connections" dialog box lists the available connections, including the connections currently defined in the data foundation.
4. Select the connection shortcut name of each connection you want to add, and click **Next**.
5. In the "Connections Properties" dialog box, you can define additional connection properties. A dialog box opens for every connection added.
   For more information about multisource-enabled connection properties, see the related topic.
6. When you have finished defining the properties for additional connections, click **Finish**.
7. Save the data foundation by clicking the **Save** icon in the main tool bar.

Related Topics
- Creating a connection shortcut
- Data foundations with multiple connections

9.4.2 Changing a connection in a data foundation

The connection you change to must be a relational connection. For multisource-enabled data foundations, the connection must also be secured.

Before you can change to a connection, you must create the local connection or connection shortcut in the local project where the data foundation is stored.

1. Double-click the data foundation name in the local project to open the Data Foundation Editor.
2. In the Data Foundation Editor, click the **Connections** tab.
3. In the "Connections" pane, right-click the connection and select **Change**.

4. Select a new connection. How to do this depends on the data foundation type:

   - If the data foundation is single-source, the "Change Connection" dialog box lists the available connections, including the currently defined connection. Select the connection you want to change to, and click **OK**.

     If you have delimitation overrides set, you will be prompted for the delimitations you want to use in the new connection. For more information, see the related topic.

   - If the data foundation is multisource-enabled, the "Change Connection" dialog box displays the connection properties for the currently defined connection. Click the browse button in the **Connection** text box. A dialog box lists the available connections. Select the connection you want to change to, and click **Finish**.

5. Save the data foundation by clicking the **Save** icon in the main tool bar.

**Related Topics**

- **Selecting delimitation overrides to keep**

### 9.4.3 Selecting delimitation overrides to keep

When you change the connection in a data foundation, the delimitation requirements for the new connection are automatically detected. Any delimitation overrides made for the previous connection may or may not apply. The "Select Delimitation Overrides to Keep" dialog box displays so that you can indicate which overrides to keep.

Only tables with delimitation overrides for the table name, qualifier, or owner are listed. The override value is shown in the column under **Name**, **Qualifier**, or **Owner**.

1. For each override:

   - To keep the delimitation value set for the previous connection, leave the check box selected.
   - To remove the delimitation override and use the newly detected delimitation value, unselect the check box.

   **Note:**
   A tool tip for each override gives you the following information:

   - The table name, qualifier, or owner in the previous connection with the previous delimitation.
   - The new table name, qualifier, or owner in the new connection with the previous delimitation.
   - The new table name, qualifier, or owner in the new connection with the newly detected delimitation.

2. When you are finished selecting overrides to keep, click **Finish** to complete the connection change.

For all other tables in the data foundation (without previous delimitation overrides) the automatically detected delimitation for the new connection applies.
9.4.4 Searching for tables in the Connection pane

You can create a filtered list of tables in the Connection pane using a search string. You can insert tables into the data foundation from the filtered list.

1. In the Connection pane of the Data Foundation Editor, click the Show/Hide Table Search icon, ![Show/Hide Table Search icon](image)

   The Connection pane splits. The top pane continues to display all tables in all connections. The lower pane (Search pane) displays only the tables matching the search string in the selected connection.

   **Note:**
   If the list of tables in the Connection pane is filtered by table type, the filter applies in both the Connection pane and the Search pane. For more information about filtering by table type, see the related topic.

2. In the Connection pane, select the part of the connection catalog that you want to search. You can select:
   - The entire connection
   - A qualifier (if available)
   - An owner (if available)

   **Note:**
   For data foundations with multiple connections you can search only one connection at a time. You must select a connection. You can select the entire connection or a schema.

3. In the Search pane, enter the string you want to search for.

   **Note:**
   The following limitations:
   - The search is case-sensitive when searching connections in multisource-enabled data foundations.
   - The search will find parts of the table name that match the string, except for connections using OLE DB middleware drivers. In this case, the string will match only the entire table name.
   - No wildcard characters are allowed.

4. Click the search icon ![Search icon](image) in the Search pane.

   The tables with names matching the search string are listed in the Search pane. You can double-click a table in the Search pane to insert it into the data foundation.

5. To start a new search, change the search string, or select a different part of the catalog in the Connections pane to search, and click the search icon in the Search pane again.
Related Topics

• About connections in the data foundation
• Filtering the tables in the connection by table type

9.4.5 Filtering the tables in the connection by table type

In the Connection pane of the Data Foundation Editor, you can filter the list of tables in the connection by table type.

1. Select the connection you want to filter, and open the qualifier and/or owner until tables are listed. The application discovers table types in the connection as tables of each type are listed in the "Connection" pane. Once the first table type is discovered, the Filter by Table Type icon is available.

2. Click the down arrow next to the Filter by Table Type icon.
   All table types discovered so far are listed. When a table type is discovered, it is automatically selected to be displayed in the connection.

3. To stop displaying a table type, unselect it in the list.
   The tables are filtered in both the Connection pane and the table search results pane. For more information on table search, see the related topic.

4. To redisplay a table type, open the Filter by Table Type list and select the table type in the list.

Note:
The table filter only affects the display. When you close the Data Foundation Editor, or when you change the connection, the filter selection is reset.

Related Topics

• About connections in the data foundation
• Searching for tables in the Connection pane

9.5 About tables in the data foundation

A standard table is a graphical representation in the data foundation of a physical database table. You create standard tables when you insert database tables into the data foundation. The table and column names are inherited from the data source.

Note:
The information design tool does not support any table or column names that are the same as an SQL reserved word. Rename these objects in the data source before inserting them into the data foundation.
Some data sources have tables of different types. The table type is inherited from the data source and stored as a table property in the data foundation.

Once you have inserted a standard table, you can modify it in the following ways:

- Edit table properties (name and description)
- Set the case of the table name
- Change the data type of columns
- Set columns as primary and foreign keys

Tables in the data foundation can also be derived or alias tables. For more information, see the related topics.

**About qualifiers and owners**

Some connections allow for multiple databases (called qualifiers), with different owners. Standard tables and their columns inherit the current qualifier and owner from the database. The syntax for a standard table name is as follows:

- A single-source standard table name, if the table is inserted from the current qualifier and owner, has the syntax: "table_name".
- A single-source standard table name, if the table is inserted from a different qualifier or owner, has the syntax: "database_qualifier"."database_owner"."table_name".
- A multisource-enabled standard table name has the syntax: @catalog('short_name')."database_qualifier.database_owner"."table_name".

**Note:**

Qualifier and owner are not relevant for some connections, in which case only the table name is used.

**About delimitation**

When you insert a table into the data foundation, the database requirements for delimited names is determined and the information is stored with the data foundation table properties. If the table or column names need to be delimited, the names are surrounded by double quotes in the display of the table in the data foundation.

The table, column, qualifier, and owner names that need to be delimited are surrounded by double quotes when used in an SQL expression.

For single-source tables, you can override the default delimitation requirements for tables, qualifiers, and owners. For tables, use the **Delimit** command. For qualifiers and owners, use the **Change Qualifier/Owner** command. It is not possible to change whether or not columns are delimited.

When you change a connection in the data foundation, if you have delimitation overrides set, you will be prompted for the delimitations you want to use in the new connection.

**Related Topics**

- Inserting tables into the data foundation
- Editing table properties
- Setting case of table names
- Changing column data types
- About table keys
- About derived tables
About alias tables
Changing qualifiers and owners
Changing table delimitation
Selecting delimitation overrides to keep

9.5.1 Inserting tables into the data foundation

Before you begin, verify that table and column names in the data source are not the same as an SQL reserved word. If so, rename these objects in the data source before inserting them in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Select Insert Tables from the Insert menu in the data foundation view.
   The "Insert Tables" dialog box lists the connections defined in the data foundation.

3. Expand the connection to see the database tables referenced in the connection.
   By default, tables are listed for all qualifiers and owners. To list only tables for the currently used qualifier/owners, click the Show Qualifiers and Owners icon.

4. Select a table name to insert it and all its columns into the data foundation.
   Tables that are already inserted into the data foundation have an icon with a green check mark. If you insert an existing table, an alias table is inserted and you are prompted to enter a name for the alias table.
   To show the values in a table, right-click the table name and select Show Table Values. To show the values in one column, expand the table, right-click the column name and select Show Column Values.

5. Select the objects you want to detect and insert automatically into the data foundation when inserting the selected tables:
## Option | Description
---|---
**Detect keys** | Sets the key columns in the data foundation tables as they are in the database tables.

**Detect row counts** | Saves the number of rows in each table in the data foundation.

**Detect joins** | Inserts the joins between the tables being inserted.
For multisource-enabled data foundations, only joins between tables referenced by the same connection are detected. To detect joins between tables referenced by different connections, use the Detect Joins command after inserting the tables.

**Detect cardinalities** | Saves the cardinalities of the joins as they are in the database joins.

The recommended detect options are selected by default. You can change the defaults in the application preferences. See the related topic on setting preferences for the Data Foundation Editor.

6. Click **Finish** to insert the selected tables.
7. Save the data foundation by clicking the **Save** icon in the main tool bar.

**Tip:**
You can also insert tables into the data foundation by dragging them from the **Connections** pane and dropping them into the data foundation view.

### Related Topics
- About tables in the data foundation
- About alias tables
- Detecting joins
- Setting preferences for the Data Foundation Editor

## 9.5.2 Editing table properties

For alias and standard tables you can edit the table name and description.

For standard tables, you can also remove columns from the table display, edit the column data types, and set or unset primary and foreign keys. The column changes you make are also made to any related alias tables.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view and select **Edit**.
   If you are editing a derived table, the "Edit Derived Table" dialog box appears. For information on editing derived tables, see the related topic.
3. To change the table name, enter a new **Name**.

   **Note:**
   When you change the name of a standard table, you break the link with the database table. See the related link for information about renaming tables using aliases.

4. To remove columns from the table display, unselect the columns that you want to hide and click **OK**. This only affects the display in the data foundation view. The columns remain visible when showing table values or inserting the table into the business layer.

5. To change the data type of a column, select a data type in the list in the **Data Type** column. The next time you refresh the data foundation structure, the original data type of the column in the database is proposed.

6. To set or unset keys, select **None**, **Primary**, or **Foreign** from the list in the **Keys** column. The next time you use the **Detect Keys** command, the keys defined in the database tables override the keys set manually for a table. You can set an application preference so that if no keys are detected, the keys that you set manually in a data foundation table are kept.

7. Optionally, enter or edit the table **Description**.

8. Click **OK** to save the changes.

9. Save the data foundation by clicking the **Save** icon in the main tool bar.

**Related Topics**
- About derived tables
- About alias tables
- About table keys
- Setting preferences for the Data Foundation Editor
- About tables in the data foundation
- About refreshing a data foundation

### 9.5.3 Setting case of table names

Some databases require that table names be all upper case or all lower case. Use the **Set case to** command to change the case of table names.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Right-click the table header in the data foundation view and select **Set case to**. Then select **Upper case** or **Lower case**.

   To select multiple tables, click the table headers while holding down the **CTRL** key.

3. Save the data foundation by clicking the **Save** icon in the main tool bar.
9.5.4 Changing column data types

You can change column data types for standard tables in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view and select **Edit**.
3. In the list of columns, select a data type in the list in the **DataType** column.

**Note:**
The next time you refresh the data foundation structure, the original data type of the column in the database is proposed.

4. Click **OK** to save the changes.
5. Save the data foundation by clicking the **Save** icon in the main tool bar.

9.5.5 Changing qualifiers and owners

You can change the qualifiers and owners for standard tables in the data foundation. For single-source tables, you can also change whether to delimit the qualifier and owner names.

1. Open the data foundation in the editor by selecting it in the Local Projects View.
2. Right-click the table header in the data foundation view and select **Change Qualifier/Owner**.
   To select multiple tables, click the table headers while holding down the **CTRL** key.
3. In the "Change Qualifier/Owner" dialog box, click the browse button in the **Qualifier** field and select a new qualifier.
   If the qualifier name is delimited by default, the **Delimit** option is selected. To override the default delimitation, select or unselect **Delimit**.
4. To change the owner, click the browse button in the **Owner** field and select a new owner.
   If the owner name is delimited by default, the **Delimit** option is selected. To override the default delimitation, select or unselect **Delimit**.
5. When you are finished changing the qualifier and owner information, click **OK**.
6. Save the data foundation by clicking the **Save** icon in the main tool bar.

9.5.6 Changing table delimitation
You can override the default delimitation of table names for single-source standard tables in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Right-click the table header in the data foundation view and select Delimit.
   To select multiple tables, click the table headers while holding down the CTRL key.

3. Select Yes to delimit table names. Select No to stop delimiting table names.

   **Note:**
   The Delimit command effects the delimitation of table names only. Column names keep the default delimitation information.

4. Save the data foundation by clicking the Save icon in the main tool bar.

### 9.6 About table keys

Tables in the data foundation can have two types of keys:

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary</td>
<td>Single or combination of columns in a table whose values identify each row in the table. The primary key guarantees row uniqueness in a table. Each table has only one primary key.</td>
</tr>
<tr>
<td>Foreign</td>
<td>Column or combination of columns whose values are required to match a primary or another unique key in another table. Foreign keys implement constraints, for example, not allowing a sale to be added in the &lt;Sales&gt; table for a customer that does not exist in the &lt;Customer&gt; table. Each table can have multiple foreign keys.</td>
</tr>
</tbody>
</table>

Keys are indicated with icons next to column in the data foundation view.

You can set keys in data foundation tables by detecting the keys in the database table, or by setting the keys manually.

**Related Topics**
- Setting table keys
9.6.1 Setting table keys

You can set keys in data foundation tables by detecting the keys in the database tables, or by setting
the keys manually.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local
Projects View.
2. To set keys according to the keys detected in the database, in the data foundation view, select
Detect Keys from the Detect menu.

   You can set an application preference to automatically detect keys when inserting a table into the
data foundation. See the related topic on setting preferences for the Data Foundation Editor.

3. To set or unset keys manually, right-click the column in the table and select Set as Key and select
Primary, Foreign, or None.
4. Save the data foundation by clicking the Save icon in the main tool bar.

   Note: You cannot set keys for alias tables. Alias tables inherit keys from the original table.

   The next time you use the Detect Keys command, the keys defined in the database tables override
the keys set manually for a table. You can set an application preference so that if no keys are detected,
the keys that you set manually in a data foundation table are kept.

Related Topics
- About table keys
- Setting preferences for the Data Foundation Editor

9.7 About table row counts

Detect row counts

The number of rows in database tables can be detected and stored in the data foundation. Row counts
are used to detect cardinalities in the absence of table keys.

When you detect rows counts, the number of rows for the selected tables is counted and stored.

Note: Column filters are not applied when detecting row counts.
You can also set estimated row counts for tables. This can be useful if you are working with a reduced sample of data, but want queries to be optimized for the size of the production data. The row count that you set is replaced by the detected row count when you do a detect row count for that table.

The **Detect Row Count** command on the Detect menu lists the current row counts for all tables in the data foundation. From this list, you can set row counts and detect row counts for a selection of tables.

To detect row count for one table, right-click the table header in the data foundation view and select **Detect > Row Count**. The row count for the selected table is updated. To select multiple tables, click the table headers while holding down the **CTRL** key.

**Count rows**

Use the **Count Rows** command on multiple tables linked by joins to see the number of rows returned by the resulting query. Column filters are applied.

To count rows returned in a query, select tables in the data foundation view in one of the following ways:

- Right-click a table and select **Select Related Tables**.
- Click the table headers while holding down the **CTRL** key.

Then right-click a table in the selection and select **Count Rows**.

---

**9.8 About joins**

A join is a condition that links tables in the data foundation. A join restricts the data returned when the two tables are queried.

Tables that are joined usually have a parent-child relationship. If tables are not joined, then a query run on the two tables can return a result set that contains all possible row combinations. Such a result set is known as a Cartesian product and is rarely useful.

Joins are defined by linking a column in one table to a column in a second table. You can insert joins into the data foundation, or detect joins automatically.

The following sections describe the types of joins you can create.

**Equi-joins**

An equi-join is the join type created by default between two tables. An equi-join links tables based on the equality between the values in the column of one table and the values in the column of a second table. In a normalized database, the columns used in an equi-join are often the primary key from one table and the foreign key in the other.

**Self-restricting joins**

A self-restricting join is when the two tables are the same. Self-restricting joins are used to define column filters. For more information on column filters, see the related topic.
**Theta joins**
When there is no obvious direct column to column relationship between two tables, you can use a theta join. A theta join links tables based on a relationship other than equality between two columns. It is used to link a value to a range of values. For example, an order date in one table is joined to a date between start date and end date in a second table.

**Outer joins**
An outer join can be used to link tables when one table contains rows that do not have a match in the common column of the other table. Unlike an equi-join, an outer join returns all rows, regardless of whether or not there is a matching value in the joined table.

A left outer join returns all rows in the first (or left-hand side) table, even if they have no match in the second table.

A right outer join returns all rows in the second (or right-hand side) table, even if they have no match in the first table.

A full outer join returns all rows from both tables, with null values when there is no match.

**Shortcut joins**
A shortcut join is a join that provides an alternative path between two tables. Shortcut joins improve the performance of a query by not taking into account intermediate tables, and so shortening a normally longer join path.

Shortcut joins are not taken into account to define contexts, but only to decrease the number of joins whenever possible.

**Related Topics**
- Inserting and editing a join
- Detecting joins
- Inserting a column filter
- About contexts

### 9.8.1 Inserting and editing a join

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Do one of the following:
To edit an existing join, right-click the join line in the data foundation view and select **Edit Join**.

To insert and edit a join, select the **Insert Join** command from the **Insert** menu in the data foundation view.

### Option 3. To define the first side of the join, select the table from the list in **Table 1**, and then select the column name.

### Option 4. To define the second side of the join, select the table from the list in **Table 2**, and then select the column name.

### Option 5. Select the join operator:

- **Between Table 1 and Table 2**, a list of join operators lets you select how to compare the values of the columns in the join.

- The default operator creates an equi-join (\(=\)). The other operators are for joins that are not based on equality between column values (\(>\), \(>=\), \(<\), \(<=\), \(!=\)).

- To create a theta join using the BETWEEN operator, select the \(=\) operator. While holding down the **CTRL** key, select a second column in **Table 2**.

- For more information about the types of joins possible, see the related topic about joins.

### Option 6. To create a shortcut join, select the **Shortcut join** option.

- A shortcut join is a join that provides an alternative path between two tables. Shortcut joins improve the performance of a query by not taking into account intermediate tables, and so shortening a normally longer join path.

### Option 7. To create an outer join, select the **Outer join** options.

- An outer join allows rows to be returned even when there is no matching row in the joined table. Select the options as follows:

  - To create a left outer join, select the **Outer join** option below **Table 1**. This join will return all rows in **Table 1**, even if they have no match in **Table 2**.

  - To create a right outer join, select the **Outer join** option below **Table 2**. This join will return all rows in **Table 2**, even if they have no match in **Table 1**.

  - To create a full outer join, select the **Outer join** option below both tables. This join returns all rows from both tables, with null values when there is no match.

### Option 8. Select the cardinality for the join from the **Cardinality** list. You can also click the **Detect** button to automatically detect the cardinality defined for the join in the database.

- For more information about cardinality, see the related topic.

### Option 9. Optionally, edit and validate the join expression.

- Based on the columns and operators you select, an SQL expression is automatically generated to define the join. You can type a custom expression for the join. To get help editing the join expression, click the **SQL Assistant** icon.
Note:
If you change the table or column names when editing the join expression, the changes are not immediately reflected in the Table 1 and Table 2 lists. The changes are reflected in the lists when you save and re-edit the join.

10. Click **OK** to save the join.
11. Save the data foundation by clicking the **Save** icon in the main tool bar.

**Related Topics**
- About joins
- About cardinality
- Inserting a column filter

### 9.8.2 Detecting joins

Join detection looks at the data foundation tables and proposes appropriate joins. The following methods are used:

- Join detection based on column name. This method looks for identical column names in different tables. It also checks that the data type of the two columns is the same. If more than one column matches between two tables, joins are proposed for each column.

  **Note:**
  Joins between a table and its alias are not proposed.

- Join detection based on database keys. This method looks for relationships defined in the database between primary keys and foreign keys.

- For data foundations with an SAP NetWeaver BW connection, join detection is based on the joins in the database schema referenced in the connection.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Select **Detect Joins** from the **Detect** menu in the data foundation view.

3. Select the join detection method.

   For a multisource-enabled data foundation, select a method for each connection. This method is used to detect joins between tables referenced by the connection. You can also detect joins between tables from different connections. In this case, the method used is by column name.

4. From the detected joins proposed in the dialog box, select the joins to be inserted into the data foundation.

   To automatically detect cardinalities for the selected joins, select the **Detect cardinalities** option.
Note:
You can set an application preference to detect and insert joins automatically every time a table is inserted into the data foundation. See the related topic on setting preferences for the Data Foundation Editor.

Related Topics
- About joins
- About cardinality
- Setting preferences for the Data Foundation Editor

### 9.8.3 Inserting a column filter

A column filter, also called a self-restricting join, lets you restrict the values returned whenever the table is used in a query.

The following rules apply to column filters:
- Only one filter per column is allowed.
- You can insert a filter on a calculated column.
- The expression can contain sub-queries.
- The following @Functions are allowed in the expression: @Prompt and @Variable.
- If you insert a filter into a standard table, and then create an alias from the table, the filter is not inserted into the alias table.
- If you insert a filter into an alias tables, the filter is not automatically inserted into the original standard table.
- When you merge tables that include filters, the filters are not included in the resulting derived table.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the column name in the table in the data foundation view and select Insert Filter.
   A self-join on the column is proposed in the "Edit Join" dialog box. For example, if you insert a filter on the `<age>` column in the `<Customer>` table, the following self-join is proposed:

   "Customer"."age"="Customer"."age"

3. Edit the second part of the self join (the expression to the right of the equal sign) to filter the column values. To get help editing the join expression, click the SQL Assistant icon 📘.

Related Topics
- About joins
9.9 About cardinality

Cardinality further describes how tables are joined by stating how many rows in one table match rows in another table. Cardinalities are needed when detecting aliases and contexts to resolve loops in the data foundation.

Cardinality of a table is expressed as a pair of numbers: the number of rows in one table that match the number of rows in the joined table. The number of rows that match can be none (0), one (1), or many (n) for each table.

For example, the tables <Customer> and <Reservations> are linked by a join.

• For each customer, there can be one or more reservations, so the cardinality of the <Customer> table is one-to-many, or 1,n.
• For each reservation, there can be one and only one customer, so the cardinality of the <Reservations> table is one-to-one, or 1,1.

Cardinality of the join is also expressed as a pair of numbers: the maximum number of rows in the second table that match one row in the first table, and the maximum number of rows in the first table that match one row in the second table.

In the example, the cardinality of the <Customer>-<Reservations> join is n,1, because the maximum rows that can match a row in <Customer> is n, and the maximum rows that can match a row in <Reservations> is 1.

Cardinalities can be detected for joins automatically and stored in the data foundation. The detection method first detects primary and foreign keys. Cardinalities are set according to the key status of the column in the two tables as follows:

<table>
<thead>
<tr>
<th>First table column</th>
<th>Second table column</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary key</td>
<td>Foreign key</td>
<td>1, n</td>
</tr>
<tr>
<td>Foreign key</td>
<td>Primary key</td>
<td>n,1</td>
</tr>
</tbody>
</table>

If no keys are detected, the cardinality is set using table row counts.

Related Topics
- Detecting and setting cardinalities
- About joins
- About table keys
- About table row counts
### 9.9.1 Detecting and setting cardinalities

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Select **Detect Cardinalities** in the **Detect** menu.
   The "Detect Cardinalities" dialog box lists the current cardinalities for all joins in the data foundation.
3. Select the joins for which you want to detect cardinalities and click **Detect Cardinalities**.
4. To set the cardinality of a join manually, select the cardinality from the list in the **Cardinality** column for the join.
5. Click **Finish** to save the changes.

**Note:**
You can set an application preference to detect and insert cardinality automatically every time a join is inserted into the data foundation. See the related topic on setting preferences for the Data Foundation Editor.

**Related Topics**
- About cardinality
- Setting preferences for the Data Foundation Editor

### 9.10 Inserting a calculated column

A calculated column is a new column in a data foundation table that is the result of a calculation based on one or more columns of the same table.

**Note:**
Inserting a calculated time column based on a column with a time-related data type is a special case of a calculated column. For the procedure to insert a time column, see the related topic.

The following rules apply to calculated columns:
- You can insert calculated columns into standard tables only.
- You can include only columns from the same table in the SELECT statement.
- Sub-queries are not allowed.
1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view and select **Insert Calculated Column**.
3. If the data foundation is multisource-enabled and you want to use database-specific SQL to define the calculated column, select the **Database-specific** option.
For more information about SQL expressions in multisource-enabled data foundations, see the related topic.

4. Build the SQL SELECT statement that defines the column by dragging and dropping columns and functions into the SELECT pane.
   For more information about using the SQL expression editor, see the related topic.

5. Click Validate to check the validity of the SQL expression.

6. Click OK.
   The column is inserted into the table and appears in the data foundation view with a special icon. A tool tip displays the SQL expression of the calculated column when you pass the cursor over the column name.

7. To check the results of the calculated column, right-click the column and select Show Column Values.

8. Save the data foundation by clicking the Save icon in the main tool bar.
   To edit the definition, right-click the column name in the table in the data foundation view and select Edit Calculated Column.

Related Topics
• Inserting a time column
• About the SQL/MDX Expression Editor
• SQL expressions in multisource-enabled data foundations

9.11 Inserting a time column

A time column is a calculated column that contains a date part (for example, month, quarter, or year) based on a column with a time-related data type.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Right-click a column with a time-related data type and select Insert Time Column.
   Columns with time-related data types have a special icon that resembles a calendar.

3. Select a date part from the list.
   A calculated column is inserted into the table and appears in the data foundation view with a special icon. A tool tip displays the SQL expression of the calculated column when you pass the cursor over the column name.

4. To check the results of the calculated column, right-click the column and select Show Column Values.

5. Save the data foundation by clicking the Save icon in the main tool bar.
   To edit the definition, right-click the column name in the table in the data foundation view and select Edit Calculated Column.
9.12 About derived tables

A derived table is a virtual table in the data foundation that combines other tables using calculations and functions. You can create objects in the business layer on a derived table in the same way that you do for a standard table. Use derived tables in the following situations:

• To create a table with columns from other tables. The column definitions can include complex calculations and functions.
• To create a single table that combines two or more tables (called merging tables).
• To create a table that contains a selection of columns from different tables.

Related Topics
• Inserting a derived table based on a data foundation table
• Merging tables
• Inserting and editing a derived table

9.12.1 Inserting a derived table based on a data foundation table

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header of the table to be the basis for the derived table, and select Insert > Derived Table.
3. Enter a name for the derived table that is unique in the data foundation, and click OK.

A derived table with the new name and all the columns from the original table is inserted into the data foundation.

Edit the derived table to make the desired modifications.

Related Topics
• Inserting and editing a derived table
• About derived tables
### 9.12.2 Merging tables

Merging tables inserts a derived table into the data foundation that consists of the combined columns from two or more tables linked by joins.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. In the data foundation view select the tables you want to merge:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To select a table and all</td>
<td>Right-click the table and select **Select</td>
</tr>
<tr>
<td>tables related to it by</td>
<td>Related Tables**.</td>
</tr>
<tr>
<td>joins</td>
<td></td>
</tr>
<tr>
<td>To select tables manually</td>
<td>Click the table headers while holding down</td>
</tr>
<tr>
<td></td>
<td>the <strong>CTRL</strong> key.</td>
</tr>
</tbody>
</table>

3. Right-click the selection of tables and select **Merge**.
4. Enter a name for the table that is unique within the data foundation and click **OK**. The merged table is inserted as a derived table. The new table is joined to any tables that the original tables were joined to.
5. Select if you want to delete the original tables.
   The original tables become obsolete and you have the choice of deleting them. If you choose to keep the original tables, the joins linking those tables are deleted, however the tables stay in the data foundation.

**Note:**
In a multisource-enabled data foundation, a derived table that is the result of a merge creates expressions using SQL-92 standard syntax. To use database-specific SQL, you must edit the derived table and explicitly select database-specific syntax.

To edit the merged table, right-click the table header and select **Edit**.

**Related Topics**
- [Inserting and editing a derived table](#)
- [About derived tables](#)
- [SQL expressions in multisource-enabled data foundations](#)

### 9.12.3 Inserting and editing a derived table
1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing derived table</td>
<td>Right-click the table header in the data foundation view and select Edit.</td>
</tr>
<tr>
<td>To insert and edit a derived table</td>
<td>Select the Insert Derived Table command from the Insert menu in the data foundation view.</td>
</tr>
</tbody>
</table>

3. Give the derived table a unique name within the data foundation.

4. In a multisource-enabled data foundation, if you want to include database-specific functions in the definition of the derived table, select the Database-specific option.

   For more information about SQL syntax in multisource-enabled data foundations, see the related topic.

5. Enter or edit the SQL expression for the derived table in Expression.

   **Note:**

   To build an expression for the first time, you can use the SQL Builder. The SQL Builder works like the query panel. You drag and drop tables and columns to be included in the derived table. The SQL expression is generated automatically.

   For more information on using the SQL Expression Editor, see the related topic.

6. Click Validate to check the validity of the SQL expression.

7. Click OK.

8. Link the derived table to other tables in the data foundation by inserting the appropriate joins.

9. To check the results of the derived table, right-click the table and select Show Table Values.

10. Save the data foundation by clicking the Save icon in the main tool bar.

**Related Topics**

- SQL expressions in multisource-enabled data foundations
- About the SQL/MDX Expression Editor
- Inserting and editing a join

### 9.13 About alias tables

An alias table is a reference to a standard or derived table in the data foundation. It is an identical duplicate of the original table (except for column filters), but has a different name. The data in the table
is exactly the same as the original table, but the different name "tricks" the SQL of a query to accept that you are using two different tables.

You use alias tables to break loops in the join paths in the data foundation. The Detect Aliases command analyzes the join paths and proposes alias tables to break any loops detected in the data foundation. For more information about resolving loops, see the related topic.

You also use alias tables to rename a table. The link between the data foundation and database is based on the table name. If you create an alias to give the table a new name, the link to the database table is preserved, but the alias table name is used in the data foundation.

To find alias tables already inserted into the data foundation, you can do a data foundation search. There are also commands to highlight the alias tables of an original table, and highlight the original for an alias table. For more information, see the related topics.

Related Topics
• Detecting alias tables
• Inserting alias tables
• Resolving loops
• Searching for tables and columns in the data foundation
• Highlighting aliases
• Highlighting the original table of an alias

9.13.1 Inserting alias tables

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. In the data foundation view, select the table to be the basis of the alias.
   You can create aliases for more than one table at a time. Click the table headers while holding down the CTRL key.
3. Right-click the selection and select Insert > Alias Table.
4. In the "Insert Alias Tables" dialog box, unselect any aliases you do not want to insert.
5. Edit the names for the alias tables in the Alias Name column, and click OK.

The selected alias tables are inserted into the data foundation. The original table name is listed in parentheses in the table header.

To edit the name and description of an alias table, click the table header in the data foundation view and select Edit.

Related Topics
• About alias tables
9.13.2 Detecting alias tables

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Select Detect Aliases from the Detect menu in the data foundation view. You can also detect aliases from the Aliases and Contexts pane in the Data Foundation Editor.

   Click the Detect Aliases icon.

   The command analyzes the join paths and proposes alias tables to break any loops detected in the data foundation.

3. If alias tables are proposed, select which aliases you want to insert automatically.

   The selected alias tables are inserted into the data foundation. The original table name is listed in parentheses in the table header.

   To edit the name and description of an alias table, click the table header in the data foundation view and select Edit.

Related Topics

• About alias tables

9.13.3 Highlighting aliases

Use this command to highlight the alias tables associated with a standard or derived table in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Right-click the table header of the original table and select Highlight Aliases.

   The original and any associated alias tables are highlighted. All other tables are grayed.

   **Note:**

   Some alias tables may be off the visible area of the data foundation view. You can quickly check for hidden highlighted tables by clicking the Fit to Window icon at the bottom of the data foundation view. To undo the Fit to Window, click the Reset Zoom icon.

3. To return to the normal data foundation view display, click anywhere in the view.
9.13.4 Highlighting the original table of an alias

Use this command to highlight the original table in the data foundation of an alias table.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header of the alias table and select **Highlight Original Table**.
   The original and alias tables are highlighted. All other tables are grayed.

   **Note:**
   The original table may be off the visible area of the data foundation view. You can quickly check for a hidden table by clicking the **Fit to Window** icon at the bottom of the data foundation view. To undo the **Fit to Window**, click the **Reset Zoom** icon.

3. To return to the normal data foundation view display, click anywhere in the view.

9.14 About contexts

A context is a collection of joins which provide a valid query path. The most common use of contexts is to resolve loops in the data foundation when the loop cannot be resolved by creating an alias table. Another use of contexts is when multiple fact tables share a dimension table. In this case, a context is created for each fact table.

In the information design tool, a context resolves a loop by identifying a set of joins that define one specific join path through the tables in the loop. The user is prompted for the context to use at query time. The context ensures that joins are not included from different paths within the same SQL query.

A context is defined by setting states for the joins involved in the ambiguity. In a context, a join has one of three states:

- Included joins: In a part of the schema that is ambiguous, the context solves the loop by defining a path with the included joins.
- Excluded joins: In a part of the schema that is ambiguous, the excluded joins define the path that context will never take.
- Neutral joins are in a part of the schema that is not ambiguous, and are always included in the query path of the context. Any join that is not explicitly included or excluded is neutral.

When a new join or table is inserted into the data foundation, it is neutral by default. Contexts will not need to be updated unless the new table or join is explicitly involved.

You can insert contexts into the data foundation manually, or by detecting contexts. The detect command analyzes the join paths and proposes contexts to resolve any loops that cannot be resolved by alias tables.
9.14.1 Detecting contexts

Before detecting contexts, you need to set cardinalities and detect aliases. See the related topic about resolving loops for the prerequisite tasks.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. From the Aliases and Contexts pane, select the Detect Contexts icon.
   The command analyzes the join paths and proposes contexts to resolve any loops that cannot be resolved by alias tables.

   **Note:**
   You may get a message that the loop can be resolved using aliases. See the related topic on resolving loops.

3. In the “Detect Contexts” dialog box, select the contexts you want to insert.
   To see the context highlighted in the data foundation view, click the proposed context name. A join that is included in the context is shown by an included icon. A join that is excluded is shown by the excluded icon.

4. Click OK to insert the selected contexts into the data foundation.
   The new contexts are listed in the Aliases and Contexts pane in the Contexts folder.

5. Save the data foundation by clicking the Save icon in the main tool bar.

9.14.2 Inserting and editing contexts
1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

2. Select the **Aliases and Contexts** pane.

3. Do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing context</td>
<td>Select the context in the <strong>Contexts</strong> folder.</td>
</tr>
<tr>
<td>To insert and edit an context</td>
<td>Select the <strong>Insert Context</strong> icon.</td>
</tr>
</tbody>
</table>

The properties of the context are displayed in the **Context Properties** pane:

- Context name
- All joins in the data foundation
- The state of the join in this context: whether the join is included, excluded, or neutral

4. Edit the context name in **Name**.

5. To include or exclude a join, or to set the join to neutral, click the join expression in **Join Expression** list. The state toggles each time you click.

   You can also toggle the state by clicking the join line in the data foundation view.

   A join that is included in the context is shown by an included icon. A join that is excluded is shown by the excluded icon.

6. Save the data foundation by clicking the **Save** icon in the main tool bar.

**Related Topics**

- **About contexts**

### 9.15 Resolving loops

Loops occur when multiple paths join tables. The rows that are returned from the query are the intersection of the results for each path, so fewer rows are returned than expected.

An alias table breaks a loop by using the same table twice in the query, once for each path. That way, the rows returned in the query are the union of the results for each path.

A cycle is a loop that occurs when the tables joined by the loop all have a cardinality of (1,n). In this case, the **Detect Aliases** command cannot determine which table to create an alias for.

When loops cannot be resolved with an alias table, contexts are used. You use contexts to resolve the ambiguity by explicitly directing the query as to which join path to use.
1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Detect and insert all the joins in the data foundation.
3. Detect or set the cardinalities for the joins.
4. Make sure there are no cycles in the data foundation. You can do this by examining the cardinalities of any loops, or by refreshing the loop resolution status (see the following steps in this procedure).
5. Make sure none of the joins have a cardinality of (n,n). Aliases and contexts cannot be detected. Set the cardinality manually for (n,n) joins.
6. From the Aliases and Contexts pane, detect aliases.
7. Detect contexts. It is recommended to insert all proposed contexts.
8. In the Loops box, click the Visualize Loops icon to check if all the loops have been resolved. Possible loops are listed in Loops. To check if the loops are resolved, click the Refresh Loop Resolution Status icon.

A message appears suggesting what to do about unresolved loops.

A loop is resolved when you see a green check mark next to the loop name.

Related Topics
- Detecting joins
- Detecting and setting cardinalities
- Detecting alias tables
- Detecting contexts

9.16 About input columns

An input column is a parameter in the data source that expects a value. The parameter is represented in a table column in the data foundation. For each input column, you can provide a static value, or a parameter defined in the data foundation. The parameter can prompt the end user for a value and be associated with a list of values. For some input columns, providing a value is optional.

Following are examples of input columns in the data foundation:
- SAP NetWeaver BW key date variables. An input column is inserted into each table in the data foundation that handles time-dependent data. To resolve these input columns at query time, a parameter is inserted into the data foundation called key date. By default, at query time, the key date parameter is not prompted. It is automatically assigned the current date. You can edit the prompt parameters in the data foundation.
- SAP ERP ABAP function input parameters. One data foundation table is created to map the main function. It contains input columns for the input parameters of the function. These parameters can be mandatory or optional. For mandatory parameters, you enter either a static value or a data foundation parameter for the associated input column.
9.16.1 Editing input columns

To assign a parameter to an input column, you must first define the parameter in the data foundation. For SAP NetWeaver BW key date variables, a data foundation parameter is automatically inserted. For more information on parameters, see the related topic.

1. You can list input columns to edit in two ways:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To list input columns for a table</td>
<td>Right-click the table header in the data foundation view and select <strong>Edit Input Column</strong>.</td>
</tr>
<tr>
<td>To list input columns for all tables</td>
<td>Right-click anywhere in the data foundation view and select <strong>Edit Input Columns</strong>.</td>
</tr>
</tbody>
</table>

If no input columns exist in the table or data foundation, the **Edit Input Column** command is not available.

2. To assign a value to an input column, in the "Edit Input Columns" dialog box, select the column in the list.
   - To assign a static value, select the **Static Values** radio button and enter the values in the text box.
   - To assign a parameter, select the **Parameter** radio button. Click the icon ![Parameter Icon] to select from a list of parameters defined in the data foundation.

**Note:**
The **Values** column indicates mandatory input columns as [mandatory], and optional input columns as [optional].

3. Save the data foundation by clicking the **Save** icon in the main tool bar.

**Related Topics**
- About parameters

---

9.17 About parameters and lists of values in the data foundation
A parameter is a variable in the universe that requires a value at query time. Parameters are often defined to prompt the user to supply a value, and in this case are referred to as prompts.

A list of values is a collection of data values that can be associated with an object in the universe, allowing the user to choose values for a prompt.

You can insert parameters and lists of values into the data foundation. They are inherited by any business layer built on the data foundation, but cannot be modified in the business layer.

To insert a parameter or list of values, go to the Parameters and Lists of Values tab in the Data Foundation Editor. From there, the procedure is the same as inserting parameters and lists of values into a business layer. See the related topics.

**Related Topics**
- About parameters
- About lists of values

### 9.18 About data foundation properties

The following properties apply to the entire data foundation:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Describes the data foundation. The description can be entered when you create the data foundation in the New Data Foundation wizard, and edited at any time in the data foundation properties.</td>
</tr>
<tr>
<td>Allow Cartesian products</td>
<td>When selected, if the SQL expression that defines an object in the data foundation could result in a Cartesian product, the SQL is allowed. <strong>Note:</strong> A Cartesian product is a result set which contains all the possible combinations of each row in each table included in a query. A Cartesian product is almost always an incorrect result.</td>
</tr>
<tr>
<td>Multiple SQL statements for each context</td>
<td>When selected, allows the user to select the query path when the query involves contexts. This option should be selected if the data foundation contains contexts.</td>
</tr>
<tr>
<td>SQL Parameters</td>
<td>Specifies custom values for SQL generation parameters that override the default values.</td>
</tr>
</tbody>
</table>
### Related Topics

- Editing SQL options in the data foundation
- Setting SQL generation parameters in the data foundation
- Displaying a data foundation summary
- About contexts

#### 9.18.1 Editing SQL options in the data foundation

1. Open the data foundation in the editor by double-clicking the data foundation in the Local Projects View.
2. Make sure the top level of the data foundation is selected in the tree view in the Data Foundation pane.
3. Click the Properties tab in the properties pane.
4. Select or unselect options as required. For a description of the options, see the related topic.
5. Save the data foundation by clicking the Save icon in the main tool bar.

#### Related Topics

- About data foundation properties

#### 9.18.2 Setting SQL generation parameters in the data foundation

The custom values for SQL generation parameters in the data foundation override the default values.

1. Open the data foundation in the editor by double-clicking the data foundation in the Local Projects View.
2. Make sure the top level of the data foundation is selected in the tree view in the Data Foundation pane.
3. Make sure the Properties tab is selected in the properties pane.
4. Click the Parameters button.
5. In the "Query Script Parameters" dialog box, edit the parameters:
   The currently defined SQL generation parameters are listed. Non-default parameters and parameters with non-default values are in bold type.

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To change the value of an existing parameter</td>
<td>Click the <strong>Value</strong> column and select or enter the new value.</td>
</tr>
<tr>
<td>To add a predefined parameter</td>
<td>Click the arrow in the list box next to the <strong>Add</strong> button to display the list of predefined parameters. Select the parameter from the list and click <strong>Add</strong>.</td>
</tr>
<tr>
<td>To add a custom parameter</td>
<td>Make sure no predefined parameter is listed in the box next to the <strong>Add</strong> button, then click <strong>Add</strong>. A parameter with a default name is added to the table. To edit the parameter name, click the <strong>Name</strong> column. Click the <strong>Value</strong> column to enter a value.</td>
</tr>
</tbody>
</table>

To see a description of all the predefined SQL generation parameters and their values, click the help button.

6. To return to the default list of parameters and the default values, click **Default Values**. This removes any added parameters from the list and sets all values to the default.

**Related Topics**
- About SQL Generation Parameters

### 9.18.3 Displaying a data foundation summary

Use this command to display the number of each type of object defined in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation in the Local Projects View.
2. Make sure the top level of the data foundation is selected in the tree view in the **Data Foundation** pane.
3. Make sure the **Properties** tab is selected in the properties pane.
4. Click the **Summary** button.

The data foundation summary displays in a new dialog box.

### 9.19 Showing and profiling table and column values
9.19.1 Showing table values

You can show values for one or more tables in a data foundation. If filters are defined on any of the columns, the filters are applied when showing values. To see values in the database for a table (no data foundation filters are applied), show the values on a table in the Connections pane.

The show values command by default opens a tab in the editor to display the values. You can set a preference to have the values open in a dedicated view or a dialog box. For more information, see the related topic.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To show values for one or more tables with filters applied</td>
<td>Right-click the table header in the data foundation view. To select multiple tables, click while holding down the CTRL key.</td>
</tr>
<tr>
<td>To show values for one table without filters applied</td>
<td>From the Connections pane in the Data Foundation Editor, expand the connection and right-click the table name.</td>
</tr>
</tbody>
</table>

3. Select Show Table Values.

The show values window appears. To see what you can do in this window, see the related topic on showing and profiling values in a data source.

Related Topics

- Showing and profiling values in a data source
- Setting preferences for showing values

9.19.2 Showing column values

You can show values for one or more columns in a data foundation table. If a filter is defined on the column, the filter is applied when showing values. To see values in the database for a column (no data foundation filters are applied), show the values on a column in the Connections pane.
The show values command by default opens a tab in the editor to display the values. You can set a preference to have the values open in a dedicated view or a dialog box. For more information, see the related topic.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.

   You can also show column values from the data foundation view in the Business Layer Editor. In this case, open the business layer.

2. Do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To show values for one or more columns with filters applied</td>
<td>Right-click the column name in the data foundation view. To select multiple columns, click while holding down the CTRL key.</td>
</tr>
<tr>
<td>To show values for one column without filters applied</td>
<td>From the Connections pane in the Data Foundation Editor, expand the connection and right-click the column name.</td>
</tr>
</tbody>
</table>

3. Select **Show Column Values**.

   The show values window appears. To see what you can do in this window, see the related topic on showing and profiling values in a data source.

**Related Topics**

- Showing and profiling values in a data source
- Setting preferences for showing values

### 9.19.3 Profiling column values

You can profile the values for a column in a data foundation table. Profiling shows graphically (in a pie or bar chart) the number of occurrences of each value of a column. If the column has a filter defined, the filter is applied.

1. You can profile column values from the data foundation view in either the data foundation or business layer editor. Open the editor by double-clicking the resource in the Local Projects View.

2. In the data foundation view, right-click the column name in the table display and select **Profile Column Values**

   The Profile Column Values window appears. To see what you can do in this window, see the related topic.

**Related Topics**

- Showing and profiling values in a data source
9.20 Showing local dependencies in the data foundation

Use the **Show Local Dependencies** command when you plan to change tables and columns in the data foundation. The command will find the business layers and their objects that depend on the table or column.

1. Right-click the table header or column name in the data foundation view and select **Show Local Dependencies**.
   - You can select several tables and/or columns by holding down the **CTRL** key.
   - The business layers that depend on the selected tables and columns are listed.

2. Select the business layer for which you want to see the dependent objects.
   - A dialog box lists the data foundation tables and columns, and the business layer objects that depend on them.

3. To edit a business object, double-click the object name in the **Business layers and objects** box. The Business Layer opens with the focus on the selected object.

**Related Topics**
- About resource dependencies

9.21 About refreshing a data foundation

Refreshing the structure lets you update the data foundation with changes in the underlying database tables.

To start the Refresh Structure Wizard, in the Data Foundation Editor, select **Refresh Structure** from the **Detect** menu.

The wizard detects the following changes and lists them each in their own dialog box. In each case, you select which of the proposed changes to make in the data foundation.

- Tables in the data foundation that were deleted in the database. The wizard proposes to delete these tables and any related joins from the data foundation.
- Columns in data foundation tables that were deleted in the database tables. The wizard proposes to update each corresponding table in the data foundation to delete these columns and the joins that use these columns.
- Columns added in the database. The wizard proposes to update each corresponding table in the data foundation to add these columns.
- Column data types changed in the database. The wizard proposes to update the data type of each column in the data foundation that is different from the database column type.
The wizard lists your selected changes in a summary dialog box and asks for confirmation before continuing with the refresh.

### 9.22 Inserting a custom data foundation view

A custom data foundation view is a subset of the data foundation Master view. Use views when you are editing a data foundation that contains many tables, and you are interested in working with a subset of tables. You can define multiple custom views for the data foundation.

Table operations are allowed from all views. Any changes to a table, for example assigning the table to a family, are propagated to all views in the data foundation.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Do one of the following:
   
<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To insert an empty view</td>
<td>Select <strong>Insert View</strong> from the Insert menu.</td>
</tr>
<tr>
<td>To insert a view based on a selection of tables</td>
<td>Select one or several tables (click the table headers while holding down the <strong>CTRL</strong> key).</td>
</tr>
<tr>
<td></td>
<td>Right-click the selection and select <strong>Insert &gt; View from Selection</strong>.</td>
</tr>
<tr>
<td>To insert a view based on search results</td>
<td>Use the Search Panel to search for the tables to include in the view. For more information on how to search, see the related topic.</td>
</tr>
<tr>
<td></td>
<td>Right-click the selection in the search results view and select <strong>Insert &gt; View from Selection</strong>.</td>
</tr>
</tbody>
</table>

3. Enter a name for the view and click **OK**.
   A new tab appears at the bottom of the view pane and the new view is displayed.
4. To add a table to a view:
   a. Click the **Master** view tab, or any other view that contains the table you want to add.
   b. Select the table or tables you want to add.
   c. Right-click the table header and select **Add to View**.
   d. Select the view from the list (only views that do not already contain the table are listed).
5. Save the data foundation by clicking the **Save** icon in the main tool bar.
9.23 Searching for tables and columns in the data foundation

1. To open the search panel, in the data foundation view, click the **Show/Hide Search Panel** icon.
2. By default, the search looks for tables. To look for columns, click the **icon** in the filter text box.
3. You can limit your search in several ways:
   - Enter text to search for in the filter text box.
   - Select connections, table types, column types, families, and contexts in the respective lists. The tables that match the search criteria are highlighted in the data foundation view.
4. To modify the view to display only the matching tables, click the **Search Options** icon at the top of the search panel, and select **Auto arrange search results**.
   **Tip:** You can also use the **Center on Selection** command to change the zoom on the data foundation display so that all the tables in a selection are visible in the display window.
5. Click **Reset** to clear the search criteria and start a new search.

**Note:**
Certain operations on tables are not possible when the Search Panel is active, for example inserting alias and derived tables, detecting joins, or checking integrity. The data foundation commands that are not available when using the Search Panel are grayed. To use these commands, close the Search Panel by clicking the **Show/Hide Search Panel** icon.

**Related Topics**
- Centering the view on a selection

9.24 Inserting a comment into the data foundation view

A comment is a note that you can place anywhere in a data foundation view.

1. Open the data foundation in the editor by double-clicking the data foundation in the Local Projects View.
2. Select **Insert Comment** from the **Insert** menu.
3. In the "Edit Comment" dialog box, define the display parameters of the note, and enter the comment text. The comment is inserted into top left corner of current view.

4. Drag the comment to the location in the view where you want it to appear.

5. Save the data foundation by clicking the **Save** icon in the main tool bar.

### 9.25 Centering the view on a selection

The **Center on Selection** command lets you temporarily change the zoom on the data foundation view display so that all the tables in a selection are visible in the display window.

1. In the data foundation editor, make a selection of tables. For example, using the Search Panel, select all tables in a particular family.

2. Right-click the table header of one of the selected tables and select **Center on Selection**. The data foundation display zooms so that all the selected tables appear in the display window. To reset the display, close the Search Panel if it is open, or click the **Reset Zoom** icon in the lower toolbar of the data foundation view.

**Tip:**
You can also center the display on a table or join by selecting the table or join name in the tree view in the **Data Foundation** panel to the left of the display.

**Related Topics**
- Searching for tables and columns in the data foundation

### 9.26 Changing the display of objects in the data foundation

#### 9.26.1 Auto-arranging tables in the data foundation view

Once you have inserted tables and joins into the data foundation view, you can automatically arrange the tables according to the flow of joins from one to many.

To change the display of individual tables, see the related links.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Click the **Auto Arrange Tables** icon in the data foundation view. The tables are arranged in the view.

3. Save the data foundation by clicking the **Save** icon in the main tool bar.

**Related Topics**
- Changing table display
- Removing table columns from the display
- Grouping tables using families

### 9.26.2 Changing table display

For each table in the data foundation, you can select how much table information to display in the view.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view and select **Display**.
3. Select a display mode:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collapsed</td>
<td>Displays only the table header.</td>
</tr>
<tr>
<td>Joins only</td>
<td>Displays the table header and any columns that are part of a join.</td>
</tr>
<tr>
<td>Expanded</td>
<td>Displays the table header and all columns.</td>
</tr>
</tbody>
</table>

To remove specific columns from the table display, see the related topic.

You can also toggle through the different display modes by clicking the arrow icon on the right side of the table header.

4. Save the data foundation by clicking the **Save** icon in the main tool bar.

**Related Topics**
- Removing table columns from the display

### 9.26.3 Removing table columns from the display

You can hide columns in a data foundation table so that they do not display in the view.
1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Right-click the table header in the data foundation view and select **Edit**.
3. Unselect the columns that you want to hide and click **OK**.
   The columns disappear from the table display.

   **Note:**
   This affects the display only in the data foundation view. The columns remain visible when showing table values or inserting the table into the business layer.

4. Save the data foundation by clicking the **Save** icon in the main tool bar.

### 9.26.4 Grouping tables using families

A family is a set of display parameters that can be used to visually group tables of the same type. For example, you might define different families for fact and dimension tables.

The display parameters include background color, text color and font.

1. Open the data foundation in the editor by double-clicking the data foundation name in the Local Projects View.
2. Click the **Edit Families** icon in the data foundation view.
3. In the "Edit Families" dialog box, create a family for each table group:
   a. Click **Add** to define a new family.
   b. Enter a name for the family in **Name**.
   c. Edit the table color, background, and font for the family.
   d. Click **Apply** to save the family definition.

   You can export and import family definitions. Export creates a file in a local folder that can be shared between different users of the information design tool.

4. When you have added all families, click **OK**.
5. Assign tables to families. For each family:
   a. Select the tables to be assigned to one family. Click the table headers while holding down the **CTRL** key.
   b. In the **Families** list in the data foundation view tool bar, select the family.

   **Note:**
   When you assign a table to a family, the table acquires the display attributes of the family in the current view and all data foundation views where the table is present.

6. To remove a table from a family, select the table, and in the **Families** list, select **No family**.
7. Save the data foundation by clicking the **Save** icon in the main tool bar.
Working with business layers

10.1 About business layers

A business layer is a collection of metadata objects that map to SQL or MDX definitions in a database, for example, columns, views, database functions, or pre-aggregated calculations. The metadata objects include dimensions, hierarchies, measures, attributes, and predefined conditions. Each object corresponds to a unit of business information that can be manipulated in a query to return data. Business layers can be created directly on an OLAP cube, or on a data foundation that is built on a relational database.

When a business layer is complete, it is published to a repository or a local folder as a universe. A universe is a published .unx file that includes a business layer and its connection to an OLAP cube, or a business layer and its corresponding data foundation. The universe is available in the repository to SAP BusinessObjects data analysis and report creation applications.

The principle role of the business layer is to define and organize metadata before it is published as a universe. An alternative way to understand the business layer is to think of it as a metadata workbench that a designer uses to assemble and modify a metadata set before publishing as a universe for data analysis and report creation applications.

Related Topics
- About business layer objects
- About business layer properties
- How to build a relational business layer
- How to build an OLAP business layer

10.2 About business layer objects

The Business Layer objects pane contains the metadata objects that make up the business layer.

Every object in the business layer has a name that can be edited. When naming objects, use business vocabulary familiar to the users who will be using the universe for querying, analysis, and reporting.

Objects can have three states:
- **Active**: Object is visible in the Query Panel. This is the default state.
- **Hidden**: Object is valid but not available in the Query Panel (used by other objects as a hidden object).
- ** Deprecated**: Object is hidden and not valid. One possible use for this state is when the target database field no longer exists, but you want to keep the object for possible future use.

Each object in the business layer has properties that are applied in the published universe. You can define properties when you insert an object, and modify object properties at any time. See the related topics about inserting and editing objects.

Depending on the type of data source, you can create and edit the following types of objects in the business layer:

<table>
<thead>
<tr>
<th>Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="folder.png" alt="Folder" /> <strong>Folder</strong></td>
<td>A folder is a container that holds a group of related objects. You create folders to group objects that have a common purpose in the business layer. The folder has no role in a query. It is only used to organize objects.</td>
</tr>
<tr>
<td><img src="dimension.png" alt="Dimension" /> <strong>Dimension</strong></td>
<td>A dimension is an object that maps to one or more table columns or a function in a database and represents an axis of analysis in a query. For example, Product, Geography, Time, and Employee are common dimensions. Each dimension classifies an aspect of an activity in a business environment. In a business layer, dimensions represent contextual information (the axes of analysis).</td>
</tr>
<tr>
<td><img src="measure.png" alt="Measure" /> <strong>Measure</strong></td>
<td>Measures are objects that represent calculations and aggregate functions that map to statistical and analytic data in the database. In a business layer, measures represent the factual information (data). Numeric data is usually, but not always, the source of a measure. Aggregating the information must make sense for the object to be a measure. For example, summing up sales revenue makes sense, so Sales Revenue is a measure. Summing product list prices is not necessarily useful, so List Price is a dimension, or perhaps an attribute of the Product Dimension. You can create measures from non-numeric objects by counting things. This can result in measures like Number of Orders.</td>
</tr>
<tr>
<td><img src="attribute.png" alt="Attribute" /> <strong>Attribute</strong></td>
<td>An attribute is an object attached to a parent object that provides additional descriptive information about the parent. Attributes can be defined for dimensions, hierarchies, and levels.</td>
</tr>
</tbody>
</table>
### Object | Description
--- | ---
Filter | A filter is a condition object that limits the data returned in a query. Filters can be inserted into the Query Filters pane in the Query Panel to be applied to the query. Native filters are defined by an SQL WHERE clause on data foundation tables. Native filters apply to business layers based on data foundations. Business filters are defined by creating and combining conditions on dimensions and measures in the business layer.
Analysis dimension (OLAP only) | An analysis dimension allows you to logically group dimensions and hierarchies that share the same axis of analysis. Analysis dimensions are often used for hierarchical analysis. You define a default hierarchy for the analysis dimension. This is the hierarchy used when the entire analysis dimension is included as a result object in a query. Default hierarchies have the following icon: 📊
Hierarchy (OLAP only) | A hierarchy is the representation in the business layer of the hierarchy in the OLAP cube. If the hierarchy in the cube is level-based, level objects in the business layer represent the levels. If the hierarchy in the cube is value-based (parent-child), the levels are not represented in the business layer. The levels are visible when previewing members and in the Member Selector. Value-based hierarchies generated automatically in the business layer have the following icon: 📊
Level (OLAP only) | A hierarchy level in a level-based hierarchy.
Named set (OLAP only) | A named set is a collection of members of a hierarchy in the business layer. A native named set is defined using an MDX expression. For some connections, native named sets are created automatically to represent named sets in the cube. A business named set is defined by selecting members.
A calculated member is a member of a hierarchy that is calculated using an explicitly defined MDX expression that can include data from the OLAP cube, mathematical operators, numbers, and functions.

Calculated members are available in the "Member Selector" when creating queries.

### Related Topics
- Inserting a folder
- Inserting and editing dimensions
- Inserting and editing measures
- Inserting and editing attributes
- Inserting and editing filters
- Inserting and editing analysis dimensions
- Inserting and editing hierarchies
- Inserting and editing hierarchy levels
- Inserting and editing named sets
- Inserting and editing calculated members

### 10.3 How to build a relational business layer

Before you begin:
- You need a project in the Local Projects View.
- You need a data foundation saved in the same local project.

For links to more detailed information on each step, see the Related Topics.

1. Right-click the project folder in the Local Projects View and select **New > Business Layer**.
2. Follow the steps in the New Business Layer wizard. Select **Relational data foundation** as the data source.
   
   The business layer is created in a .blx file in the local project folder. It opens automatically in the Business Layer Editor.
3. Build the business layer:
   
   If you created the business layer with the **Automatically create classes and objects** option selected (default), all objects are created in the business layer as dimensions. You need to specify the measures explicitly using the **Turn into Measure with Aggregation Function** command.

   Otherwise, insert the business layer objects:
a. Insert folders and subfolders to organize the business layer.

**Note:**
When you drag a table into the business layer from the data foundation, a folder is automatically inserted.

b. Drag and drop tables and columns into the desired folders and rename the objects if needed.

c. Specify the measures using the **Turn into Measure with Aggregation Function** command.

4. You can enhance the function of the business layer in several ways, for example:

   - Insert attributes to provide descriptive information for dimensions
   - Insert additional measures
   - Insert pre-defined filters (mandatory or optional) that can limit data returned in queries
   - Insert parameters with optional prompts
   - Insert custom lists of values to be associated with a prompt
   - Insert navigation paths to define drill paths
   - Create business layer views to restrict the objects seen in the Query Panel
   - Set SQL options and SQL generation parameters in the business layer properties
   - Set up aggregate awareness to improve query performance

5. Run an integrity check to validate the dependencies, object expressions, and parameters and lists of values. Right-click the business layer name in the **Business Layer** pane, and select **Check Integrity**.

6. Save the business layer by clicking the **Save** icon in the main tool bar.

Some commands to help you maintain the business layer are listed below.

   - If you change object definitions, use **Show Local Dependencies** to find the other business layer objects and data foundation objects that might be impacted by the changes.
   - If the related data foundation is modified, you must refresh the business layer manually. The related topic describes commands to help you do this.
   - Use **Change Data Foundation** to change the source data foundation for the business layer.
   - For multisource-enabled business layers, use **Compute Statistics** to improve query performance.

**Related Topics**

- Creating a local project
- How to build a data foundation
- About the Business Layer Editor
- About business layer objects
- Turning a dimension into a measure
- Inserting a folder
- Inserting and editing dimensions
- Inserting dimensions directly from the data foundation
- Inserting and editing measures
- Inserting and editing filters
- Inserting and editing a parameter
- Inserting or editing a list of values
- Associating a list of values with a business object
10.3.1 Specifying the type of data source for a business layer

This section describes the "Select the type of data source for the business layer" page of the New Business Layer wizard.

You choose to create a business layer from either a relational or an OLAP data source.

<table>
<thead>
<tr>
<th>Data Source Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relational</td>
<td>The business layer is based on a data foundation. You can select any data foundation in the current project folder.</td>
</tr>
<tr>
<td>OLAP</td>
<td>The business layer is based on an OLAP cube. You can select any OLAP connection or connection shortcut in the current project folder.</td>
</tr>
</tbody>
</table>

1. Click one of the data source types in the list.
2. Click Next.
   A business layer naming page appears for the new business layer.

Related Topics
- Naming a business layer

10.3.2 Naming a business layer

This section describes the "Resource Name" page of the New Business Layer wizard.

You enter a name and description of the business layer. This is the name of the universe that is published from the business layer.
10.3.3 Selecting a data foundation for a business layer

This section describes the "Select Data Foundation" page of the New Business Layer wizard. Select a data foundation as the data source for the new business layer. You can choose to do one of the following:

- Automatically create the business layer objects from the tables and columns in the data foundation.
- Create an empty business layer. You must manually add the objects from the data foundation after creation.

1. Click the browse button at the end of the Data Foundation text field. A list of available data foundations appears.
2. Select a data foundation in the list and click **OK**. The data foundation name appears in the name field. The **Automatically create classes and objects** option is selected by default.
3. Do one of the following:
   - If you want to automatically populate the business layer with objects and classes, click **Finish**.
   - If you do not want to automatically populate the business layer, unselect the option and click **Finish**.

The new business layer opens in an editing tab. You can now insert and edit the business layer objects.

The **Automatically create classes and objects** option creates all objects in the business layer as dimensions. Specify the measures explicitly using the **Turn into Measure with Aggregation Function** command. For more information, see the related topic.

Related Topics

- How to build a relational business layer
- Turning a dimension into a measure
- About business layer objects

10.4 How to build an OLAP business layer

Before you begin:
• You need a project in the Local Projects View.
• You need an OLAP connection or connection shortcut saved in the same local project.

For links to more detailed information on each step, see the Related Topics.

1. Right-click the project folder in the Local Projects View and select New > Business Layer.
2. Follow the steps in the New Business Layer wizard. Select OLAP connection as the data source.
   The business layer is created in a .blx file in the local project folder. It opens automatically in the Business Layer Editor.

3. Objects in the business layer are inserted automatically based on the cube. You can enhance the function of the business layer in several ways, for example:
   • Insert analytical dimensions, hierarchies, and attributes
   • Insert named sets
   • Insert calculated members
   • Insert measures
   • Insert pre-defined filters (mandatory or optional) to limit data returned in queries
   • Insert parameters with optional prompts
   • Insert custom lists of values to be associated with a prompt
   • Create business layer views to restrict the objects seen in the Query Panel

4. Run an integrity check to validate the dependencies, object expressions, and parameters and lists of values. Right-click the business layer name in the Business Layer pane, and select Check Integrity.

5. Save the business layer by clicking the Save icon in the main tool bar.

Some commands to help you maintain the business layer are listed below.

• If you change object definitions, use Show Local Dependencies to find the other business layer objects that might be impacted by the changes.
• If the underlying data source is modified, use Refresh Structure to refresh the business layer.
• Use Change OLAP Connection to change the connection for the business layer and to edit the OLAP data source properties.

Related Topics
• Creating a local project
• Creating an OLAP connection
• About business layer objects
• Inserting and editing analysis dimensions
• Inserting and editing hierarchies
• Inserting and editing hierarchy levels
• Inserting and editing attributes
• Inserting and editing named sets
• Inserting and editing calculated members
• Inserting and editing measures
• Inserting and editing filters
• Inserting and editing a parameter
• Inserting or editing a list of values
10.4.1 Selecting an OLAP connection and cube for a business layer

This section describes the "Select OLAP Connection" page of the New Business Layer wizard.

Select an OLAP connection and the OLAP cube as the data source for the new business layer.

Note:
You cannot build a business layer on an SAP BICS Client connection although these connections appear in the list. Use SAP BICS Client connections in SAP BusinessObjects query and reporting applications to connect directly to the BEx query. No business layer or universe is required to access BEx queries. For information on building a universe on an SAP NetWeaver BW connection, see the related topic.

<table>
<thead>
<tr>
<th>OLAP connection options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLAP connection</td>
<td>Click the browse button at the end of the text field to select an OLAP connection or connection shortcut defined in the project.</td>
</tr>
<tr>
<td>Detect measure projection function</td>
<td>If not selected the database delegated function is applied.</td>
</tr>
<tr>
<td>Create attribute from unique name</td>
<td>An attribute is created for the unique name for each dimension.</td>
</tr>
<tr>
<td>Search</td>
<td>Enter a search string for a cube and click the search icon.</td>
</tr>
<tr>
<td>List of connection cubes</td>
<td>List of available cubes to the connection. If there are multiple cubes, you browse to and select the target cube.</td>
</tr>
</tbody>
</table>

Related Topics

• Selecting objects from an OLAP cube for a business layer
• About projection functions
• Universes on SAP NetWeaver BW
10.4.2 Selecting an Essbase Accounts dimension

This section describes the "Select Accounts Dimension" page of the New Business Layer wizard. For connections to Essbase data sources, the New Business Layer wizard creates measures in the business layer from the objects in the specified Accounts dimension in the data source. Select one dimension in the list you want to use as the Accounts dimension and click Next.

10.4.3 Selecting objects from an OLAP cube for a business layer

This section describes the "Select Objects" page of the New Business Layer wizard. Expand the object nodes under the selected cube and select the objects to be included in the new business layer. Click Finish when you have completed the selection. The new business layer appears in the Business Layer pane.

10.5 About the Business Layer Editor

You use the Business Layer Editor to create and edit business layer objects and properties. This topic describes how to navigate the editor. For steps to help you build the structure of your business layer, see How to build a relational business layer or How to build an OLAP business layer.

The Business Layer Editor is divided into browsing panes on the left, an editing pane on the upper right, and a data source pane on the lower right. The browsing panes allow you to work with different elements of the business layer. Access the panes by clicking the corresponding tab:

- Business Layer
- Queries
- Parameters and Lists of Values
- Navigation Paths

For more information about what you can do in each of the browsing panes, see the related topic.

The Business Layer is the default browsing pane. It displays a tree view of the objects in the business layer. The following options are available for displaying and navigating the business layer tree view:

- Filter by business layer view
• Search for an object
• Change display options: Show or hide objects, show unique names

The editing pane lets you edit the properties of the object or element selected in the browsing pane.

The data source pane displays the data foundation or the OLAP connection information:
• The data foundation master view containing all tables and joins displays by default. Tabs for other data foundation views, if defined, appear at the bottom of the data source pane. To change views, click the tab.
• The OLAP metadata in the connection displays in the left side of the data source pane. Select a metadata object to display its properties in the right side of the pane.

Related Topics
• About business layer properties
• About business layer objects
• About queries in a business layer
• About parameters
• About lists of values
• About navigation paths for objects
• About business layer views
• Filtering by business layer view
• Searching for business layer objects
• Changing display options of the business layer tree view

10.5.1 Changing display options of the business layer tree view

When editing a business layer, the Business Layer browsing pane displays a tree view of the objects in the business layer. Use this procedure to change the display mode of the business layer objects.

1. Click the Display options icon at the top of the Business Layer browsing pane.
2. For business layers based on an OLAP connection, select one of three options:
   • Display caption to display the object names.
   • Display unique name to display the unique object name from the cube.
   • Display caption and unique name
3. To show only active objects in the business layer tree view, select Hide non-active objects.

The display options remain in effect until you close the editor.

Related Topics
• Changing the state of an object: Active, Hidden, or Deprecated
10.6 About business layer properties

The following properties and options are defined for the entire business layer. The restrictions are applied in the published universe.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Identifies the business layer and also the universe when the business layer is published.</td>
</tr>
<tr>
<td>Description</td>
<td>Describes the universe purpose and content. This description is available to display in the query and reporting tools that use the published universe.</td>
</tr>
<tr>
<td>Limit result size to</td>
<td>Specifies the number of rows that are returned in a query. This limits the number of rows returned, but does not restrict the RDBMS from processing all rows in the query. It only limits the number once the RDBMS has started to send rows.</td>
</tr>
<tr>
<td>Limit execution time to</td>
<td>Specifies the number of minutes to limit the time passed for query execution, but does not stop the process on the database.</td>
</tr>
<tr>
<td>Warn if estimate exceeds</td>
<td>When selected, displays a message when an estimation of the execution time exceeds the specified number of minutes.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Query Options</strong> (apply to business layers based on data foundations)</td>
<td></td>
</tr>
<tr>
<td>Allow use of sub-queries</td>
<td>When selected, allows subqueries in a query.</td>
</tr>
<tr>
<td>Allow use of union, intersect, and minus operators</td>
<td>When selected, lets you combine queries using the data set operators union, intersect, and minus, to obtain one set of results.</td>
</tr>
<tr>
<td>Allow complex operands in Query Panel</td>
<td>When selected, displays complex operands in the list of operands available when defining a filter in the Query Panel.</td>
</tr>
<tr>
<td>Multiple SQL statements for each measure</td>
<td>When selected, one SQL query is generated for each measure or group of measures belonging to a different fact table, or for measures that have a WHERE clause (filtered measure). If the measure objects are based on columns in the same table, then the separate SQL queries are not generated, even if this option is selected.</td>
</tr>
<tr>
<td>Data source</td>
<td>Specifies the data source of the business layer: either a data foundation or an OLAP connection.</td>
</tr>
<tr>
<td>SQL Parameters (apply to business layers based on data foundations)</td>
<td>Specifies custom values for SQL generation parameters that override the default values or any customized value in the data foundation properties.</td>
</tr>
<tr>
<td>Comments</td>
<td>Contains comments about the business layer.</td>
</tr>
<tr>
<td>Summary</td>
<td>Displays a summary of the number of each type of object defined in the business layer. For business layers based on a data foundation, the type and number of data foundation objects are also displayed.</td>
</tr>
</tbody>
</table>

**Related Topics**

- [Editing the business layer name, description, and comments](#)
• Editing query limits and options in the business layer
• Changing the data source of a business layer
• Setting SQL generation parameters in the business layer
• Displaying a business layer summary

10.6.1 OLAP data source properties

The following properties apply to the OLAP data source for the business layer:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLAP Connection</td>
<td>The connection or connection shortcut that provides the access to the OLAP data source. To change the connection, click the browse icon at the end of the field to open a list of available connections.</td>
</tr>
<tr>
<td>Cube</td>
<td>The cube selected for the current connection. You can select a different cube only if a cube was not specified when the connection was defined. To change the cube, click the browse icon at the end of the field to open a list of available cubes.</td>
</tr>
</tbody>
</table>
| END_MDX value      | The value of the END_MDX parameter. The END_MDX parameter is equivalent to the END_SQL parameter available for universes based on data foundations. The value of the END_MDX is added to the end of every MDX statement. For example, you can use the END_MDX parameter to track the database server activity by tracing who is running queries. The solution consists of adding a comment at the end of every MDX query with information about the user and the universe. For example: 

//User: @Variable('BOUSER') Universe: @Variable('UNVNAME')
10.6.2 Editing the business layer name, description, and comments

1. Open the business layer in the editor by double-clicking the business layer in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the Business Layer pane.
3. Change the business layer properties in the editing pane:
   - To change the business layer name, edit Name.
   - To enter or edit the business layer description, click the Properties tab.
   - To enter or edit comments for the business layer, click the Comments tab.
4. Save the business layer by clicking the Save icon in the main tool bar.

Related Topics
• About business layer properties

10.6.3 Editing query limits and options in the business layer

1. Open the business layer in the editor by double-clicking the business layer in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the Business Layer pane.
3. Click the Query Options tab in the editing pane.
4. Select or unselect options and edit limit values as required. For a description of the options, see the related topic.
5. Save the business layer by clicking the Save icon in the main tool bar.

Related Topics
• About business layer properties

10.6.4 Changing the data source of a business layer

To change the data source for a business layer, the new data source (data foundation, OLAP connection or connection shortcut) must be saved in the same local project folder as the business layer.
1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the Business Layer pane.
3. Make sure the Properties tab is selected in the editing pane.
4. Do one of the following depending on the type of data source for the business layer:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>For data foundation sources</td>
<td>Click <strong>Change Data Foundation</strong>. Select the new data foundation from the list and click <strong>OK</strong>.</td>
</tr>
<tr>
<td>For OLAP sources</td>
<td>Click <strong>Change OLAP Connection</strong>. In the &quot;Edit OLAP Data Source Properties&quot; dialog box, click the browse icon at the end of the <strong>OLAP Connection</strong> text box. Select the new OLAP connection or connection shortcut and click <strong>OK</strong>.</td>
</tr>
</tbody>
</table>

**Note:**
For more information on advanced OLAP properties, see the related topic.

5. Save the business layer by clicking the **Save** icon in the main tool bar.

**Related Topics**
- OLAP data source properties

**10.6.5 Setting SQL generation parameters in the business layer**

The custom values for SQL generation parameters in the business layer override the default values or any customized value set in the data foundation properties.
1. Open the business layer in the editor by double-clicking the business layer in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the Business Layer pane.
3. Make sure the Properties tab is selected in the editing pane.
4. Click the Parameters button.
5. In the "Query Script Parameters" dialog box, edit the parameters:
   - The currently defined SQL generation parameters are listed. Non-default parameters and parameters with non-default values are in bold type.
### Option | Action
--- | ---
**Change the value of an existing parameter.** | Click the Value column and select or enter the new value.

**Add a predefined parameter.** | To display the list of predefined parameters, click the arrow in the list box next to the Add button. Select the parameter from the list and click Add.

**Add a custom parameter.** | Make sure no predefined parameter is listed in the box next to the Add button, then click Add. A parameter with a default name is added to the table. To edit the parameter name, click the Name column. Click the Value column to enter a value.

To see a description of all the predefined SQL generation parameters and their values, click the help button.

6. To return to the default list of parameters and the default values, click Default Values. This removes any added parameters from the list and sets all values to the default.

7. Save the business layer by clicking the Save icon in the main tool bar.

**Related Topics**

- About SQL Generation Parameters

### 10.6.6 Displaying a business layer summary

Use this command to display the number of each type of object defined in the business layer. For business layers based on a data foundation, the type and number of data foundation objects are also displayed.

1. Open the business layer in the editor by double-clicking the business layer in the Local Projects View.
2. Make sure the top level of the business layer is selected in the tree view in the Business Layer pane.
3. Make sure the Properties tab is selected in the editing pane.
4. Click the Summary button.

The business layer summary displays in a new dialog box.

### 10.7 Working with business layer objects
10.7.1 Inserting a folder

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the Business Layer pane, select the folder or analysis dimension in which you want to insert the folder.
   To insert a folder at the top level, select the top node (business layer name) in the tree.
3. Click the Insert object icon at the top of the Business Layer pane and select Folder.
4. Edit the folder properties in the "Folder Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the folder as it appears in the Query Panel.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the folder.</td>
</tr>
<tr>
<td>State</td>
<td>The state of the folder, whether it is Active, Hidden, or Deprecated.</td>
</tr>
<tr>
<td>Note:</td>
<td>If the state is set to Hidden or Deprecated, the state of the objects in the folder are unchanged, however they do not appear in the Query Panel.</td>
</tr>
<tr>
<td>Content</td>
<td>A list of objects in the folder that lets you define properties that describe what the object is used for in the query (for result, for filter, for sort). You can change the order of the objects in the folder using the up and down arrow keys to the right of the list.</td>
</tr>
<tr>
<td>Custom Properties</td>
<td>Optional custom properties and their values.</td>
</tr>
</tbody>
</table>

5. Save the business layer by clicking the Save icon in the main tool bar.

Related Topics

- About business layer objects
- Changing the state of an object: Active, Hidden, or Deprecated
10.7.2 Inserting and editing dimensions

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.

2. In the Business Layer pane, do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing dimension</td>
<td>Select the dimension.</td>
</tr>
<tr>
<td>To insert a dimension</td>
<td>Select the folder or analysis dimension in which you want to insert the dimension. To insert a dimension at the top level, select the top node (business layer name) in the tree. Click the Insert object icon at the top of the Business Layer pane and select Dimension.</td>
</tr>
</tbody>
</table>

**Note:**
Because of an MDX limitation, dimensions that are inserted or copied into an OLAP business layer cannot be used in conditions or sorts. The options Object can be used in Condition and Object can be used in Sort are unavailable.

3. Edit the dimension properties in the "Dimension Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the object.</td>
</tr>
<tr>
<td>State</td>
<td>The state of the object, whether it is Active, Hidden, or Deprecated.</td>
</tr>
<tr>
<td>Data Type</td>
<td>The data type of the dimension.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SQL Definition or MDX Definition</td>
<td>The SQL or MDX query expression that defines the object.</td>
</tr>
<tr>
<td>Extra Tables (relational only)</td>
<td>Extra tables that can be included in the query (by a join in the SQL expression) when returning values for the business layer object.</td>
</tr>
<tr>
<td>OLAP Hierarchy (OLAP only)</td>
<td>The hierarchy that is associated with the dimension.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> A hierarchy must be specified if you want to insert dimension attributes.</td>
</tr>
<tr>
<td>Keys tab (relational only)</td>
<td>The database columns used as primary and foreign keys. Keys allow queries to take advantage of indexes on key columns. Defining keys speeds up data retrieval by optimizing the SQL that is generated for the query. For example in a star schema database if you build a query that filters on a value in a dimension table, the filter can be applied directly on the fact table by using the dimension table foreign key. This avoids inefficient joins to dimension tables.</td>
</tr>
<tr>
<td>Advanced tab</td>
<td>Properties that include settings for:</td>
</tr>
<tr>
<td></td>
<td>• Access Levels</td>
</tr>
<tr>
<td></td>
<td>• Where in query expressions the object can be used.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Because of an MDX limitation, dimensions that are inserted or copied into an OLAP business layer cannot be used in conditions or sorts. The options <strong>Object can be used in Condition</strong> and <strong>Object can be used in Sort</strong> are unavailable.</td>
</tr>
<tr>
<td></td>
<td>• List of values</td>
</tr>
<tr>
<td></td>
<td>• Display options</td>
</tr>
<tr>
<td>Source Information</td>
<td>Descriptive fields that apply to objects used by Data Integrator.</td>
</tr>
<tr>
<td>Custom Properties</td>
<td>Optional custom properties and their values.</td>
</tr>
</tbody>
</table>
4. To see the SQL query script for the dimension definition, click **Show Script**.
5. Save the business layer by clicking the **Save** icon in the main tool bar.

**Related Topics**
- About business layer objects
- Inserting dimensions directly from the data foundation
- Changing the state of an object: Active, Hidden, or Deprecated
- Defining the SQL expression for an object
- Defining the MDX expression for an object
- Associating extra tables
- Defining keys for a dimension
- Setting object access levels
- Setting where objects can be used
- Associating a list of values with a business object
- Defining display formats for objects
- About source information for business layer objects
- Inserting and editing custom properties

**10.7.3 Inserting dimensions directly from the data foundation**

For business layers based on a data foundation, you can drag and drop objects from the data foundation into the business layer.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
   The data foundation on which the business layer is based displays in the data source pane on the lower right side of the editing tab.

2. Select the objects in the data foundation view that you want to insert:
   - To select a table, click the table header.
   - To select multiple tables, click the table headers while holding down the **CTRL** key.
   - To select a column, click the column name in the table.
   - To select multiple columns, click the column names while holding down the **CTRL** key.

3. Drag the selection to the **Business Layer** pane and drop the selection into the desired folder in the business layer. To insert the dimensions at the top level, drop the selection into the top node (business layer name) of the tree.
   When you drag and drop a table, a folder is automatically inserted into the business layer. The folder contains a dimension for every column.
   The SQL expression for each dimension is automatically defined.

4. Save the business layer by clicking the **Save** icon in the main tool bar.
If necessary, turn any of the inserted dimensions into measures using the Turn into Measure with Aggregation Function command. For more information, see the related topic.

Related Topics
• Turning a dimension into a measure
• Inserting and editing dimensions

10.7.4 Defining keys for a dimension

Defining keys is only available for dimensions built on a data foundation.
1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Select the dimension in the Business Layer pane.
3. In the "Dimension Properties" pane, click the Keys tab.
4. Click Add key.
   You can define one primary and multiple foreign keys for a dimension. The first key added is the primary key.
   Note:
   To detect existing key columns in the database, click Detect.
5. Select the key in the table and click the SELECT column to add the SQL SELECT statement.
   To build the statement in the SQL Editor, select the [ ] at the end of the SELECT column.
6. Click the WHERE column to add the SQL WHERE statement.
   To build the statement in the SQL Editor, select the [ ] at the end of the WHERE column.
7. Click the Active column to enable or disable the key.
8. Save the business layer by clicking the Save icon in the main tool bar.

Related Topics
• Inserting and editing dimensions
• About the SQL/MDX Expression Editor

10.7.5 Inserting and editing measures

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the **Business Layer** pane, do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing measure</td>
<td>Select the measure.</td>
</tr>
<tr>
<td>To insert a measure</td>
<td>Select the folder or analysis dimension in which you want to insert the measure. To insert a measure at the top level, select the top node (business layer name) in the tree. Click the <strong>Insert object</strong> icon at the top of the <strong>Business Layer</strong> pane and select <strong>Measure</strong>.</td>
</tr>
</tbody>
</table>

3. Edit the measure properties in the "Measure Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the object.</td>
</tr>
<tr>
<td>State</td>
<td>The state of the object, whether it is <strong>Active</strong>, <strong>Hidden</strong>, or <strong>Deprecated</strong>.</td>
</tr>
<tr>
<td>Data Type</td>
<td>The data type of the object.</td>
</tr>
<tr>
<td><strong>Projection Function</strong></td>
<td>Defines how further aggregation is done if needed for the report. Projection aggregation is different from, and is done after the aggregation defined for the measure in the SQL or MDX definition. For more information, see the related topic.</td>
</tr>
<tr>
<td><strong>SQL Definition</strong> or <strong>MDX Definition</strong></td>
<td>The SQL or MDX query expression that defines the object.</td>
</tr>
<tr>
<td><strong>Extra Tables</strong> (relational only)</td>
<td>Extra tables that can be included in the query (by a join in the SQL expression) when returning values for the business layer object.</td>
</tr>
</tbody>
</table>
You can enter values for the following MDX calculation and format properties to be included in the MDX query:

- Solve order
- Format string
- Scope Isolation
- Language

Properties that include settings for:

- Access Levels
- Where in query expressions the object can be used
- List of values
- Display options

Descriptive fields that apply to objects used by Data Integrator.

Optional custom properties and their values.

4. Save the business layer by clicking the **Save** icon in the main tool bar.

**Related Topics**

- About business layer objects
- Changing the state of an object: Active, Hidden, or Deprecated
- About projection functions
- Defining the SQL expression for an object
- Defining the MDX expression for an object
- Associating extra tables
- Setting object access levels
- Setting where objects can be used
- Associating a list of values with a business object
- Defining display formats for objects
- About source information for business layer objects
- Inserting and editing custom properties

**10.7.5.1 About projection functions**
The projection function defines how to re-aggregate a measure locally for a report. Projection functions apply only to SAP BusinessObjects Web Intelligence reports.

Measures can be aggregated at two different times during the query process:

- First, when the query retrieves the data from the data source, the measure is aggregated according to the SQL or MDX definition of the measure.
- Once the data is retrieved, it is possible to change the level of aggregation in the report. For example, the query retrieves Sales aggregated for Country and City. Then in the Web Intelligence report, Sales is reported only by Country. The projection function defines how to do the local aggregation that is needed to project the data onto the report.

If the projection function is **Sum**, the measure will be summed locally in the report. If the projection function is **Delegated**, the projection function requests that the aggregation be done in the database instead of aggregating locally.

**Related Topics**
- Inserting and editing measures

### 10.7.6 Turning a dimension into a measure

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
   
   The business layer objects appear in the **Business Layer** pane and the properties in the editing pane on the right.
2. Select the dimension in the **Business Layer** pane.
   
   To select several dimensions, click the dimensions while holding down the **CTRL** key. The command applies to all dimensions selected.
3. Right-click the selection and select **Turn into Measure with Aggregation Function**.
4. Select the aggregation function for the measure, or **None**.
   
   The SELECT statement in the SQL definition is updated to aggregate the values using the selected function. You can change the aggregation function at any time by editing the SELECT statement for the measure.
5. Save the business layer by clicking the **Save** icon in the main tool bar.

**Related Topics**
- Inserting and editing measures
10.7.7 Inserting and editing attributes

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.

2. In the **Business Layer** pane, do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing attribute</td>
<td>Select the attribute.</td>
</tr>
<tr>
<td>To insert an attribute</td>
<td>Select the dimension, hierarchy, or level in which you want to insert the attribute. Click the <strong>Insert object</strong> icon at the top of the <strong>Business Layer</strong> pane and select <strong>Attribute</strong>.</td>
</tr>
</tbody>
</table>

3. Edit the attribute properties in the "Attribute Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the object.</td>
</tr>
<tr>
<td>State</td>
<td>The state of the object, whether it is <strong>Active</strong>, <strong>Hidden</strong>, or <strong>Deprecated</strong>.</td>
</tr>
<tr>
<td>Data Type</td>
<td>The data type of the object.</td>
</tr>
<tr>
<td>Dimension</td>
<td>The parent dimension for the attribute.</td>
</tr>
<tr>
<td>(relational only)</td>
<td></td>
</tr>
<tr>
<td><strong>SQL Definition</strong> or <strong>MDX Definition</strong></td>
<td>The SQL or MDX query expression that defines the object.</td>
</tr>
</tbody>
</table>
### Property | Description
--- | ---
**Extra Tables**  
(relational only) | Extra tables that can be included in the query (by a join in the SQL expression) when returning values for the business layer object.

**Advanced tab** | Properties that include settings for:  
• Access Levels  
• Where in query expressions the object can be used  
• List of values  
• Display options

**Source Information** | Descriptive fields that apply to objects used by Data Integrator.

**Custom Properties** | Optional custom properties and their values.

4. Save the business layer by clicking the **Save** icon in the main tool bar.

**Related Topics**
- About business layer objects
- Changing the state of an object: Active, Hidden, or Deprecated
- Defining the SQL expression for an object
- Defining the MDX expression for an object
- Associating extra tables
- Setting object access levels
- Setting where objects can be used
- Associating a list of values with a business object
- Defining display formats for objects
- About source information for business layer objects
- Inserting and editing custom properties

---

**10.7.8 Inserting and editing filters**

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the **Business Layer** pane, do one of the following:
<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing filter</td>
<td>Select the filter.</td>
</tr>
<tr>
<td>To insert a filter</td>
<td>Select the folder or analysis dimension in which you want to insert the filter. To insert a filter at the top level, select the top node (business layer name) in the tree. Click the Insert object icon at the top of the Business Layer pane and select Filter.</td>
</tr>
</tbody>
</table>

3. Edit the filter properties in the "Filter Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the object.</td>
</tr>
<tr>
<td>State</td>
<td>The state of the object, whether it is <strong>Active</strong>, <strong>Hidden</strong>, or <strong>Deprecated</strong>.</td>
</tr>
<tr>
<td>Filter Type</td>
<td>Either <strong>Native</strong> (relational business layers only), or <strong>Business</strong>.</td>
</tr>
<tr>
<td>SQL Definition</td>
<td>For native filters, the SQL WHERE expression that defines the object.</td>
</tr>
<tr>
<td>Extra Tables (relational only)</td>
<td>Extra tables that can be included in the query (by a join in the SQL expression) when returning values for the business layer object.</td>
</tr>
<tr>
<td>Filter Definition</td>
<td>For business filters, click <strong>Edit Filter</strong> to define the filter based on objects in the business layer. See the related topic about building a business filter.</td>
</tr>
</tbody>
</table>
When the **Use filter as mandatory in query** option is selected, the filter is applied to every query using any object in either the universe or the folder, depending on the selected scope (**Apply on universe** or **Apply on folder**).

When the **Apply on list of values** option is selected, the filter is applied on list-of-value queries.

When **Use filter as mandatory in query** is unselected, the filter is applied only when explicitly added to the query.

### Custom Properties
Optional custom properties and their values.

---

4. Save the business layer by clicking the **Save** icon in the main tool bar.

**Related Topics**
- [About business layer objects](#)
- [Changing the state of an object: Active, Hidden, or Deprecated](#)
- [Defining the SQL expression for an object](#)
- [Associating extra tables](#)
- [How to build a business filter](#)
- [Inserting and editing custom properties](#)

### 10.7.9 Inserting and editing analysis dimensions

Analysis dimensions can be inserted into OLAP business layers only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the **Business Layer** pane, do one of the following:
3. Edit the analysis dimension properties in the "Analysis Dimension Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the object.</td>
</tr>
<tr>
<td>State</td>
<td>The state of the object, whether it is <strong>Active</strong>, <strong>Hidden</strong>, or <strong>Deprecated</strong>.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If the state is set to <strong>Hidden</strong> or <strong>Deprecated</strong>, the state of the objects in the analysis dimension are unchanged, however they do not appear in the Query Panel.</td>
</tr>
<tr>
<td>Type</td>
<td>This property is not currently used.</td>
</tr>
<tr>
<td>Default hierarchy</td>
<td>The hierarchy that is taken as the default when the entire analysis dimension is added as a result object in the Query Panel.</td>
</tr>
<tr>
<td>Key attribute</td>
<td>This property is not currently used.</td>
</tr>
<tr>
<td>Custom Properties</td>
<td>Optional custom properties and their values.</td>
</tr>
</tbody>
</table>

4. Save the business layer by clicking the **Save** icon in the main tool bar.
Hierarchies can be inserted into OLAP business layers only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the Business Layer pane, do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing hierarchy</td>
<td>Select the hierarchy.</td>
</tr>
<tr>
<td>To insert a hierarchy</td>
<td>Select the folder or analysis dimension in which you want to insert the hierarchy. To insert a hierarchy at the top level, select the top node (business layer name) in the tree. Click the Insert object icon at the top of the Business Layer pane and select Hierarchy.</td>
</tr>
</tbody>
</table>

3. Edit the hierarchy properties in the "Hierarchy Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the object.</td>
</tr>
<tr>
<td>State</td>
<td>The state of the object, whether it is Active, Hidden, or Deprecated.</td>
</tr>
<tr>
<td>MDX Definition</td>
<td>The MDX query expression that defines the object.</td>
</tr>
</tbody>
</table>
### Property Description

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Advanced tab</strong></td>
<td>Properties that include settings for:</td>
</tr>
<tr>
<td></td>
<td>• Access Levels</td>
</tr>
<tr>
<td></td>
<td>• Where in query expressions the object can be used</td>
</tr>
<tr>
<td></td>
<td>• List of values</td>
</tr>
<tr>
<td></td>
<td>• Display options</td>
</tr>
<tr>
<td><strong>Source Information</strong></td>
<td>Descriptive fields that apply to objects used by Data Integrator.</td>
</tr>
<tr>
<td><strong>Custom Properties</strong></td>
<td>Optional custom properties and their values.</td>
</tr>
</tbody>
</table>

4. Save the business layer by clicking the **Save** icon in the main tool bar.

The following objects can be added to the hierarchy:

- Levels
- Attributes
- Named sets
- Calculated members

**Related Topics**

- About business layer objects
- Changing the state of an object: Active, Hidden, or Deprecated
- Defining the MDX expression for an object
- Setting object access levels
- Setting where objects can be used
- Associating a list of values with a business object
- Defining display formats for objects
- About source information for business layer objects
- Inserting and editing custom properties
- Inserting and editing hierarchy levels
- Inserting and editing attributes
- Inserting and editing named sets
- Inserting and editing calculated members

### 10.7.11 Inserting and editing hierarchy levels
Levels can be inserted into hierarchies in OLAP business layers only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.

2. In the Business Layer pane, do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing level</td>
<td>Select the level.</td>
</tr>
<tr>
<td>To insert a level</td>
<td>Select the hierarchy in which you want to insert the level.</td>
</tr>
<tr>
<td></td>
<td>Click the <strong>Insert object</strong> icon at the top of the Business Layer pane and select Level.</td>
</tr>
</tbody>
</table>

3. Edit the level properties in the "Level Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the object.</td>
</tr>
<tr>
<td>State</td>
<td>The state of the object, when it is <strong>Active</strong>, <strong>Hidden</strong>, or <strong>Deprecated</strong>.</td>
</tr>
<tr>
<td>Business Type</td>
<td>This property is not currently used.</td>
</tr>
<tr>
<td>MDX Definition</td>
<td>The MDX query expression that defines the object.</td>
</tr>
<tr>
<td>Advanced tab</td>
<td>Properties that include settings for:</td>
</tr>
<tr>
<td></td>
<td>• Access Levels</td>
</tr>
<tr>
<td></td>
<td>• Where in query expressions the object can be used</td>
</tr>
<tr>
<td></td>
<td>• List of values</td>
</tr>
<tr>
<td></td>
<td>• Display options</td>
</tr>
<tr>
<td>Source Information</td>
<td>Descriptive fields that apply to objects used by Data Integrator.</td>
</tr>
<tr>
<td>Custom Properties</td>
<td>Optional custom properties and their values.</td>
</tr>
</tbody>
</table>
Related Topics

- About business layer objects
- Changing the state of an object: Active, Hidden, or Deprecated
- Defining the MDX expression for an object
- Setting object access levels
- Setting where objects can be used
- Associating a list of values with a business object
- Defining display formats for objects
- About source information for business layer objects
- Inserting and editing custom properties

10.7.12 Inserting and editing named sets

Named sets can be inserted into OLAP business layers only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the Business Layer pane, do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing named set</td>
<td>Select the named set.</td>
</tr>
<tr>
<td>To insert a named set</td>
<td>Select the folder, analysis dimension, or hierarchy in which you want to insert the named set. To insert a named set at the top level, select the top node (business layer name) in the tree. Click the Insert object icon at the top of the Business Layer pane and select Named Set.</td>
</tr>
</tbody>
</table>

3. Edit the named set properties in the "Named Set Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the object.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>State</td>
<td>The state of the object, whether it is <strong>Active</strong>, <strong>Hidden</strong>, or <strong>Deprecated</strong>.</td>
</tr>
<tr>
<td>Hierarchy</td>
<td>The hierarchy for the named set.</td>
</tr>
</tbody>
</table>
| Named Set Type   | The named set type:  
                   - **Native** named sets are defined with an MDX expression.  
                   - **Business** named sets are defined by selecting members using the "Member Selector". |
| MDX Definition   | For native named sets, The MDX query expression that defines the set.  
                   To select members:  
                   a. Select a hierarchy from the **Hierarchy** list.  
                   b. Click **Edit Members**.  
                   c. In the "Member Selector", select or unselect members from the given hierarchy to be included or excluded in the named set.  
                   For more information on using the "Member Selector", see the related topic. |
| Definition tab   | For business named sets, the list of members.  
                   To select members:  
                   a. Select a hierarchy from the **Hierarchy** list.  
                   b. Click **Edit Members**.  
                   c. In the "Member Selector", select or unselect members from the given hierarchy to be included or excluded in the named set.  
                   For more information on using the "Member Selector", see the related topic. |

4. Save the business layer by clicking the **Save** icon in the main tool bar.

**Related Topics**
- **About business layer objects**
- **Changing the state of an object: Active, Hidden, or Deprecated**
- **Defining the MDX expression for an object**
- **About the Member Selector**

**10.7.13 Inserting and editing calculated members**
Calculated members can be inserted into OLAP hierarchies only.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.

2. In the Business Layer pane, do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing calculated member</td>
<td>Select the calculated member.</td>
</tr>
<tr>
<td>To insert a calculated member</td>
<td>Select the hierarchy in which you want to insert the calculated member. Click the <strong>Insert object</strong> icon at the top of the Business Layer pane and select <strong>Calculated Member</strong>.</td>
</tr>
</tbody>
</table>

3. Edit the calculated member properties in the "Calculated Member Properties" pane. See the related topics for more information about specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td>The name (also called caption in OLAP business layers) of the object as it appears in the Query Panel.</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>An optional description of the object.</td>
</tr>
<tr>
<td><strong>State</strong></td>
<td>The state of the object, whether it is <strong>Active</strong>, <strong>Hidden</strong>, or ** Deprecated**.</td>
</tr>
<tr>
<td><strong>Hierarchy</strong></td>
<td>The hierarchy for the calculated member.</td>
</tr>
<tr>
<td><strong>Parent member</strong></td>
<td>The level in the hierarchy under which the calculated member is to appear. If not specified, the member appears at the root level.</td>
</tr>
<tr>
<td><strong>Expression</strong></td>
<td>The MDX expression that defines the calculated member. For more information, see the related topic.</td>
</tr>
</tbody>
</table>
You can enter values for the following MDX calculation and format properties to be included in the MDX query:

- Solve order
- Format string
- Scope Isolation
- Language

4. Save the business layer by clicking the Save icon in the main tool bar.

Related Topics
- About business layer objects
- Changing the state of an object: Active, Hidden, or Deprecated
- Defining the MDX expression for an object

10.7.14 Defining the SQL expression for an object

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the Business Layer pane, select the object.
3. In the object properties pane, select the SQL Definition tab.
4. Enter the SELECT statement directly, or click the SQL Assistant button to use the SQL editor to build the statement.

Most measures require an SQL aggregation function to be defined in the SELECT expression, for example: sum(efashion."Shop_facts"."Amount_sold").

At query run time, the aggregation defined for the measure in the SQL is done before the projection aggregation. You define the projection function separately. For more information, see the related topic.

5. Enter the WHERE statement directly, or click the SQL Assistant button to use the SQL editor to build the statement.
6. Save the business layer by clicking the Save icon in the main tool bar.

Related Topics
- About the SQL/MDX Expression Editor
- About projection functions
10.7.15 Defining the MDX expression for an object

You can edit the MDX expression for objects inserted into the business layer. If you want to edit the definition of a native object (an object generated automatically from the cube when the business layer was created), copy the native object and edit the copy.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the Business Layer pane, select the object.
3. In the object properties pane, select the MDX Definition tab.
4. Enter the Expression directly, or click the MDX Assistant button to use the MDX editor to build the statement.

**Note:**
When inserting a dimension or level, the best practice is to include .members in the MDX expression. For example, following is the expression if you insert the dimension Category in the Category hierarchy in the Product analysis dimension:

```
[Product].[Category].[Category].members
```

For some data sources, it is necessary to add .members in order to preview members.

5. Save the business layer by clicking the Save icon in the main tool bar.

Related Topics
- About the SQL/MDX Expression Editor

10.7.16 Associating extra tables

The "Associate SQL Tables" dialog box lets you associate extra tables with an object in the business layer.

Extra tables are included in the query (by a join in the SQL expression) when returning values for the business layer object. For example, if the object is based on City in table City, when you associate Region and Country as extra tables, the City values from Region and Country are included when values are returned for the business object.

Tables that can be associated in the SQL are proposed as extra tables in the list.

1. To include values from an associated table, check the box next to the table name.
2. To stop including values from a table, uncheck the box next to the table name.
10.7.17 Changing the state of an object: Active, Hidden, or Deprecated

Objects in the business layer can have three states:

- **Active**: Object is visible in the Query Panel. This is the default state.
- **Hidden**: Object is valid but not available in the Query Panel (used by other objects as a hidden object).
- **Deprecated**: Object is hidden and not valid. One possible use for this state is when the target database field no longer exists, but you want to keep the object for possible future use.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
   The business layer objects appear in the **Business Layer** pane and the properties in the editing pane on the right.
2. Select the object in the **Business Layer** pane.
   To select several objects, click while holding down the **CTRL** key.
3. Right-click the selection and select **Change State**.
4. Select the new state.
   The new state is applied to all the objects in the selection. If the state is set to **Hidden** or **Deprecated** for a folder or analysis dimension, the state of the objects in the folder are unchanged, however they do not appear in the Query Panel.
5. Save the business layer by clicking the **Save** icon in the main tool bar.

10.7.18 Setting object access levels

The security access level of an object restricts use of the object to users with the appropriate object-access level granted. You can assign the following access levels to an object:

- **Public**
- **Private**
• Controlled
• Restricted
• Confidential

If you assign Public then all users can see and use the object. If you assign Restricted, then only users that are granted the object-access level of Restricted or higher can see and use the object in the Query Panel.

Universe object-access levels are granted to users and groups in the Central Management Console. For more information, see the SAP BusinessObjects Business Intelligence platform Administrator's Guide.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the Business Layer pane, select the object for which you want to set the access level. To select multiple objects, click while holding down the CTRL key.
3. Right-click the selection and select Change Access Level and select the new access level from the list.
   You can also set the access level for an object on the Advanced tab of the object properties.
4. Save the business layer by clicking the Save icon in the main tool bar.

10.7.19 Setting where objects can be used

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the Business Layer pane, select the object.
3. In the object properties pane, click the Advanced tab.
4. Select or deselect where the object can be used:
<table>
<thead>
<tr>
<th><strong>Option</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Results</strong></td>
<td>When selected, the object can be used in a query.</td>
</tr>
<tr>
<td><strong>Conditions</strong></td>
<td>When selected, the object can be used to set in a condition.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>Because of an MDX limitation, this option is not available for dimensions that are inserted or copied in an OLAP business layer.</td>
</tr>
<tr>
<td><strong>Sort</strong></td>
<td>When selected, returned values can be sorted.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>Because of an MDX limitation, this option is not available for dimensions that are inserted or copied in an OLAP business layer.</td>
</tr>
</tbody>
</table>

5. Save the business layer by clicking the **Save** icon in the main tool bar.

### 10.7.20 Setting options for the default list of values

Dimensions, measures, attributes, and hierarchies are associated with a default list of values. You can set options for the default list of values, or associate a custom list of values with the object.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the **Business Layer** pane, select the object.
3. In the object properties pane, click the **Advanced** tab.
4. Select or deselect list of values options:

<table>
<thead>
<tr>
<th><strong>Option</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Allow users to edit list of values</strong></td>
<td>If selected, the list of values can be edited and personalized by users other than the designer.</td>
</tr>
<tr>
<td><strong>Automatic refresh before use</strong></td>
<td>If selected, the list of values is automatically refreshed each time the list is called. This can have an effect on performance each time the list of values is refreshed. You should disable this option if the list of values returns a large number of values.</td>
</tr>
</tbody>
</table>
If selected, the user running a query using this list of values is required to enter search criteria before getting filtered values for the list of values. Only the values that match the search criteria are returned in the list of values. Characters used to define the matching criteria are:
- * - Matches any number of characters, even zero characters.
- ? - Matches exactly one character.
- \ - Escapes the next character allowing you to search for a wildcard character.

If selected, the user running a query using this list of values can search for a value in the database. This option is useful when the user performs a search on partial list of values results.

5. To associate a custom list of values, see the related topic. The options defined in the custom list of values apply.
6. Save the business layer by clicking the Save icon in the main tool bar.

**Related Topics**
- Associating a list of values with a business object

### 10.7.21 Defining display formats for objects

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the Business Layer pane, select the object.
3. In the object properties pane, click the Advanced tab.
4. To define a display format for business layer objects, Click Edit display format to open the Format Editor.
   Using the Format Editor, you can select a pre-defined format or define a custom format. For more information, see the related topic.
5. Save the business layer by clicking the Save icon in the main tool bar.

**Note:**
The Database Format override for dates is not currently taken into account in query generation.

**Related Topics**
- About the Format Editor
10.7.22 About source information for business layer objects

The **Source Information** tab in the business layer object properties contains information about universes generated from Data Integrator. Technical descriptions and formulas used to calculate target tables are displayed.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical Information</td>
<td>Information about a column, for example the original database name of the concerned column for the object.</td>
</tr>
<tr>
<td>Mapping</td>
<td>Initial formula information describing how a column has been specified (used in Data Integrator), for example revenue = column calculated from several sources.</td>
</tr>
<tr>
<td>Lineage</td>
<td>Source columns for the formula used to calculate the column in the database.</td>
</tr>
</tbody>
</table>

10.7.23 Inserting and editing custom properties

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the **Business Layer** pane, select the object for which you want to insert or edit custom properties.
3. In the object properties pane, select the **Custom Properties** tab.
4. To add a custom property, click **Add**.
5. Edit the property object name and value by clicking the column in the list.
6. To delete a property, select it in the list and click **Delete**.
7. Save the business layer by clicking the **Save** icon in the main tool bar.

10.7.24 Showing associated objects

For relational business layers, you can show the objects in the business layer that reference selected data foundation tables and columns.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the data foundation view pane, select the tables or columns for which you want to see the associated business layer objects. To select a table, click the table header. To select a column, click the column name. To select multiple objects, click while holding down the CTRL key.

3. Right-click the selection and select **Show Associated Objects**.

All objects that reference the selected data foundation objects are highlighted in the business layer.

**Related Topics**
- About resource dependencies

10.7.25 Showing business layer object values

You can show the values in the underlying data source for a business layer object.

Note that for relational business layers:
- If the object references columns in the data foundation for which a filter is defined, the filters are applied.
- You can also show table and column values from the data foundation view in the Business Layer Editor.

The show values command by default opens a tab in the editor to display the values. You can set a preference to have the values open in a dedicated view or a dialog box. For more information, see the related topic.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Right-click an object in the Business Layer pane and select **Show Values**.

The show values window appears. To see what you can do in this window, see the related topic on showing and profiling values in a data source.

**Related Topics**
- Showing and profiling values in a data source
- Showing table values
- Showing column values
- Setting preferences for showing values

10.7.26 Searching for business layer objects

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
The **Business Layer** browsing pane displays a tree view of the objects in the business layer.

2. Click the **Show/Hide Search Panel** icon at the top of the **Business Layer** browsing pane. The **Search Objects** panel opens below the tree view of the business layer and displays all objects.

3. To filter by object type, in the **Search Objects** panel, click the filter icon. From the list of object types, select the types to be included in the search. The **Search Objects** panel displays only objects of the types selected.

4. To perform a text search on object name, select the **Show/Hidesearchbar** icon and type the text for the search. The **Search Objects** panel displays only objects with names containing the entered text.

5. Click an object name in the **Search Objects** panel to open the object properties in the editing pane.

6. When you have finished searching, click the **Show/Hide Search Panel** icon again to hide the **Search Objects** panel.

### 10.8 About business layer views

You can modify the display of business layer objects by using business layer views to restrict the number of objects displayed in the **Business Layer** pane. Use business layer views to group objects that share a business relationship.

Business layer views can be selected in the Query Panel. You can use business layer views to define security to grant or deny the use of business layer objects to certain users or groups. For more information on defining security using business layer views, see the related topic on Business Security Profile Create Query settings.

You can also filter the **Business Layer** pane in the editor by business layer view.

**Related Topics**

- Creating and editing a business layer view
- Filtering by business layer view
- Business Security Profile Create Query settings

### 10.8.1 Creating and editing a business layer view

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Click the **Manage Business Layer Views** icon at the top of the **Business Layer** pane. The "Edit Business Layer View" dialog box opens.

3. Do one of the following:
   - To add a view, click **New**.
   - To edit an existing view, select the view in the list.

   **Note:**
   You cannot edit the **Master** view.

4. Edit the view name in the **Name** text box.
5. In the **Objects in view** box, select or clear the check boxes next to objects in the business layer to include or exclude them from the view.
   To work with only the objects already included in the view, select **Show selected objects only**.
6. Enter or edit a description for the view in the **Description** text box.
7. Click **OK** to save the changes.

**Related Topics**
* About business layer views

### 10.8.2 Filtering by business layer view

By default all the folders and objects in the business layer are displayed in the **Business Layer** pane of the editor. You can filter what you see in the Business Layer pane using a business layer view.

You must have at least one business layer view defined.

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Select the business layer view in the list at the top of the Business Layer pane.
   To return to the display of all objects in the business layer, select **Master** from the list.

**Related Topics**
* Creating and editing a business layer view
* About business layer views

### 10.9 About parameters
A parameter is a variable in the business layer or data foundation that requires a value at run time. A parameter can have two input types:

- User input as a response to a prompt. The prompt is a question or directive that requires a user to set one or more values to restrict a result set.
- Pre-defined input that specifies a fixed value for the parameter at run time.

Parameters are defined as individual components in a business layer or data foundation, and are available to all objects in the business layer. You use parameter objects in the SQL or MDX definition of an object to prompt a user response or to implement a fixed value response to a query.

**Note:**
Parameters inserted in the data foundation are inherited by any business layer based on the data foundation. These parameters cannot be edited in the business layer. You must edit them in the data foundation.

The following properties are available for parameters:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prompt to users</td>
<td>If selected, the user is prompted to enter a value at run time. If cleared, a pre-defined value is entered at runtime for the parameter.</td>
</tr>
<tr>
<td>Prompt Text</td>
<td>The text for the prompt question or directive if Prompt to users is selected.</td>
</tr>
<tr>
<td>Set values</td>
<td>Available when the Prompt to users option is unselected. Lets you enter one or more values to be used for the parameter at the run time.</td>
</tr>
<tr>
<td>Data Type</td>
<td>The data type required for the answer to the prompt.</td>
</tr>
<tr>
<td>Allow multiple values</td>
<td>If selected, lets the user select multiple values from the list of values.</td>
</tr>
<tr>
<td>Keep last values</td>
<td>If selected, the last value chosen by the user is kept when the prompt is re-run.</td>
</tr>
<tr>
<td>Index aware prompt</td>
<td>If selected, the key column is included in the prompt to restrict the values in a list. The key column is not visible to the user.</td>
</tr>
<tr>
<td>Associated list of values</td>
<td>A list of values to provide values for the prompt.</td>
</tr>
<tr>
<td>Select only from list</td>
<td>If selected, the user is forced to select a member in the list.</td>
</tr>
<tr>
<td>Set default value</td>
<td>Lets you select values to be used as default.</td>
</tr>
</tbody>
</table>

**Related Topics**

- [Inserting and editing a parameter](#)
• Associating a list of values with a prompt defined in the business layer

10.9.1 Inserting and editing a parameter

The parameter editor can be started from the business layer or data foundation editor tabs.

**Note:**
Parameters inserted in the data foundation are inherited by any business layer based on the data foundation. These parameters cannot be edited in the business layer. You must edit them in the data foundation.

1. Click the **Parameters and Lists of Values** tab in the browsing pane of the editor.
2. Do one of the following:
   - To insert a parameter, click the **Insert Parameter** icon at the top of the **Parameters** pane.
   - To edit a parameter, click the parameter name in the list.
     The properties for the parameter appear in the editor to the right of the **Parameters** pane.
3. Edit properties as required. Parameter properties are described in the related topics.

**Related Topics**
- **About parameters**
- **Associating a list of values with a prompt defined in the business layer**

10.10 About lists of values

A list of values is a list that contains the data values associated with an object. A list of values allows a user to choose values as a response to a prompt when an associated object is included in a query. The list of values allows a data set to be restricted to the selected values.

A list of values is an independent component in the business layer or data foundation and is available to all business objects in the business layer. A list of values can be associated with an object at any time.

**Note:**
Lists of values inserted in the data foundation are inherited by any business layer based on the data foundation. These lists of values cannot be edited in the business layer. You must edit them in the data foundation.

You can define the following types of lists of values:
<table>
<thead>
<tr>
<th>Type of List of Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of values based on business layer objects (available only in the business layer)</td>
<td>The list of values is based on either a query or a custom hierarchy that includes objects in the business layer. The list is based on the values returned by the query or the hierarchy values.</td>
</tr>
<tr>
<td>Static list of values</td>
<td>The list of values is based on a list of specified values entered manually or imported from a file.</td>
</tr>
<tr>
<td>List of values based on custom SQL</td>
<td>The list of values is based on the values returned by a specified SQL expression.</td>
</tr>
</tbody>
</table>

**Related Topics**

- Inserting or editing a list of values

### 10.10.1 Inserting or editing a list of values

The list of values editor can be started from the business layer or data foundation editor tabs.

**Note:**

Lists of values inserted in the data foundation are inherited by any business layer based on the data foundation. These lists of values cannot be edited in the business layer. You must edit them in the data foundation.

1. Click the **Parameters and Lists of Values** tab in the browsing pane of the editor.
2. Do one of the following:
   - To insert a list of values, click the Insert List of Values icon at the top of the **Lists of Values** pane and select the type of LOV. The types are described in the related topic about lists of values.
   - To edit a list of values, click the list of values name in the list.

   The properties for the list of values appear in the editor to the right of the **Lists of Values** pane.
3. Edit properties and query options as required. The properties vary depending on the type of list of values:
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
</table>
| List of values based on business layer objects (available only in the business layer) | To base the list of values on a query:  
  a. In the **Definition** tab, select **List of values based on the query panel**.  
  b. Click **Edit Query**.  
  c. In the Query Panel, select objects and define query filters to define the query that returns the list of values required.  
  d. Click **OK**.  
  To base the list of values on a custom hierarchy:  
  a. In the **Definition** tab, select **List of values based on a custom hierarchy**.  
  b. Click **Add Dimension**.  
  c. Select dimensions from the list to create the hierarchy required for the list of values. The order of dimensions in the list represents the levels in the hierarchy. Use the up and down arrow keys to modify the order.  
  d. Click **OK**.  
  To see the values on the defined list, click **Preview**. |
| Static list of values                                                 | To add values manually:  
  a. In the **Definition** tab, click **Add Column** to add columns to the table. Enter the values for the columns in the table.  
  b. To add rows, click the **Add Row** icon on the right side of the table.  
  To populate the list from a file:  
  a. In the **Definition** tab, click **Import**.  
  b. Select a .txt, .csv, .prn, or .asc file to import as values for the static list.  
  c. Set the **Data Separator**, **Text Delimiter**, and **Date Format** options according to the format of the data in the file.  
  d. Click **OK**.  
  You can edit column properties in the **Properties** tab. For more information on column properties, see the related topic. |
| List of values based on custom SQL                                     | To add manually:  
  a. In the **Definition** tab, click **Add Column** to add columns to the table. Enter the values for the columns in the table.  
  b. To add rows, click the **Add Row** icon on the right side of the table.  
  To populate the list from a file:  
  a. In the **Definition** tab, click **Import**.  
  b. Select a .txt, .csv, .prn, or .asc file to import as values for the static list.  
  c. Set the **Data Separator**, **Text Delimiter**, and **Date Format** options according to the format of the data in the file.  
  d. Click **OK**.  
  You can edit column properties in the **Properties** tab. For more information on column properties, see the related topic. |
In the **Definition** tab, click **Edit SQL**.

b. In the SQL Editor, build an SQL expression to return the required values, and click **OK**.

To see the values on the defined list, click **Preview**.

You can edit column properties in the **Properties** tab. For more information on column properties, see the related topic.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Allow users to edit list of values</strong></td>
<td>If selected, the list of values can be edited and personalized by users other than the designer.</td>
</tr>
<tr>
<td><strong>Automatic refresh before use</strong></td>
<td>If selected, the list of values is automatically refreshed each time the list is called. This can have an effect on performance each time the list of values is refreshed. You should disable this option if the list of values returns a large number of values.</td>
</tr>
</tbody>
</table>
| **Force users to filter values before use** | If selected, the user running a query using this list of values is required to enter search criteria before getting filtered values for the list of values. Only the values that match the search criteria are returned in the list of values. Characters used to define the matching criteria are:  
  - * - Matches any number of characters, even zero characters.  
  - ? - Matches exactly one character.  
  - \ - Escapes the next character allowing you to search for a wildcard character. |
| **Allow users to search values in the database** | If selected, the user running a query using this list of values can search for a value in the database. This option is useful when the user performs a search on partial list of values results. |
| **Query Execution timeout**      | If selected, limits the time in seconds that the list of values query runs.                                                                 |
| **Max number of rows**          | If selected, you can enter the maximum number of rows to be returned by the list of values query.                                          |

4. In the **Options** tab, set the query options for the list of values:

5. Save the business layer or data foundation.

**Related Topics**

- About lists of values
- List of values column properties
10.10.2 List of values column properties

The Properties tab in list of values properties lets you edit the column properties on lists of values. You can edit the following properties by clicking the property column in the table of properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column Name</td>
<td>Lets you to edit the column name.</td>
</tr>
<tr>
<td>Key Column</td>
<td>Lets you to select a column to be the index-aware key.</td>
</tr>
<tr>
<td>Data Type</td>
<td>Lets you select the data type for the column.</td>
</tr>
<tr>
<td>Hidden</td>
<td>When selected, the column will not be displayed to the user. For example, you can hide a column that is only used as a key for another column.</td>
</tr>
</tbody>
</table>

Related Topics

- About lists of values

10.10.3 Associating a list of values with a business object

Associate a list of values with a business object to restrict possible input values when the object is used as a filter in the Query Panel.

By default, the default list of values is associated with an object.

You can associate a custom list of values with the object. The list of values must be available in the business layer (it is on the list in the Parameters and Lists of Values tab of the Business Layer Editor).

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. In the Business Layer pane, select the object.
3. In the object properties pane, click the Advanced tab.
4. Click the business layer object in the Business Layer pane.
5. Select the Associate list of values option.
6. To associate a custom list of values, click the browse icon, select the list of values from the list, and click OK.
   The options defined for the custom list of values override the options for the default list of values.
7. Save the business layer by clicking the Save icon in the main tool bar.
10.10.4 Associating a list of values with a prompt defined in the business layer

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Click the Parameters and Lists of Values tab under the Business Layer pane.
3. Click a parameter in the list in the Parameters pane, or click the Insert Parameter icon to define a new parameter.
   The properties for the parameter appear in the editor to the right of the Parameters pane.
4. Select the Prompt to users option.
5. Click the browse button at the end of the Associated list of values field.
6. Select the radio button for type of list of values.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Object LOV</td>
<td>Select values for the list of values from an object in the business layer.</td>
</tr>
<tr>
<td>Universe List of Values</td>
<td>Select a pre-defined customized list of values. These are the lists of values listed in the Lists of Values pane.</td>
</tr>
</tbody>
</table>

7. Select either the business layer object or a pre-defined list of values and click OK.
8. If you want to restrict the values available in the list to default values, select Set Default Values, and click the browse button at the end of the field.
   A selection box appears that lists the available values for the selected object or list. Select values on the left to populate the Selected values list and click OK.

You can now include the prompt and list of values in the SQL or MDX definition of an object in the business layer using the @Prompt function with the name of the parameter defined in this procedure: @Prompt(<parameter_name>).

Related Topics
- Inserting or editing a list of values
- About parameters
- About lists of values
- About @Prompt
# 10.11 About navigation paths for objects

A Navigation path is an object that defines the drill path used in SAP BusinessObjects reporting tools. A drill path is a list of drillable business objects that allow a report analyst to drill down on a dimension.

A navigation path object can be one of two types:

<table>
<thead>
<tr>
<th>Navigation path type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>The path is defined by the hierarchical organization of the business objects in the business layer. If the business layer contains analysis dimensions, the navigation paths include the dimensions under each analysis dimension. Otherwise, the navigation paths are the dimensions under each folder. You can view the default navigation path in the Navigation Paths tab of the business layer editor. The default path cannot be edited.</td>
</tr>
<tr>
<td>Custom</td>
<td>You define the path based on the available dimensions.</td>
</tr>
</tbody>
</table>

## Related Topics
- [Inserting a navigation path object into a business layer](#)

### 10.11.1 Inserting a navigation path object into a business layer

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Click the Navigation Paths tab under the Business Layer pane.
3. Select Custom at the top of the Navigation Paths pane.
4. Click the Insert Navigation Path icon.
5. Enter a Name and optionally a Description for the path.
   - The name and description are available to display in the query and reporting tools that use the published universe.
6. Click Add to select dimensions for the path. Use the up and down arrow keys to change the order of dimensions in the list.
7. Save the business layer.
10.12 About queries in a business layer

A query object is a query that is saved and associated with the business layer. You use the Query Panel to create queries. Queries are cataloged in the Query pane of the editor.

**Note:** Queries can be used within the information design tool to test the business layer and to preview queries. Query objects are not available to reporting and analysis products using the published universe.

10.12.1 Inserting and editing a query in the business layer

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. Click the Queries tab under the Business Layer pane.
3. In the Queries pane, do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing query</td>
<td>Select the query. Edit the Name and Description in the Query Properties pane.</td>
</tr>
<tr>
<td></td>
<td>To edit the query results and filters in the Query Panel, click Edit Query.</td>
</tr>
<tr>
<td>To insert a query</td>
<td>Click the Insert Query icon.</td>
</tr>
</tbody>
</table>

4. In the Query Panel, build or edit the query and click OK.
   The new query is available in the Queries pane.

**Related Topics**

- How to build a query
10.13 About aggregate awareness

Aggregate awareness is the ability of a relational universe to take advantage of database tables that contain pre-aggregated data (aggregate tables). Setting up aggregate awareness accelerates queries by processing fewer facts and aggregating fewer rows.

If an aggregate aware object is included in a query, at run time the query generator retrieves the data from the table with the highest aggregation level that matches the level of detail in the query.

For example, in a data foundation there is a fact table for sales with detail on the transaction level, and an aggregate table with sales summed by day. If a query asks for sales details, then the transaction table is used. If a query asks for sales per day, then the aggregate table is used. Which table is used is transparent to the user.

Setting up aggregate awareness in the universe has several steps. See the related topic for more information.

**Related Topics**
- Setting up aggregate awareness

10.13.1 Setting up aggregate awareness

This topic outlines the steps to set up aggregate awareness in a relational universe. For links to more detailed information on each step, see the Related Topics.

1. The first step is done at the database level. The database administrator must define and load the aggregate tables into the database.

   The reliability and usefulness of aggregate awareness in a universe depends on the accuracy of the aggregate tables. They must be refreshed at the same time as all the fact tables.

2. Insert the aggregate tables into the data foundation.

3. Define aggregate aware objects. These are objects in the business layer for which you want queries to use the aggregate tables when possible instead of performing aggregation using non-aggregate tables.

   In the SQL expression for the object, define the SELECT statement to use the `@Aggregate_Aware` function:

   ```sql
   @Aggregate_Aware(sum(aggr_table_1), ..., sum(aggr_table_n))
   ```

   In the `@Aggregate_Aware` function, `aggr_table_1` is the aggregate table with the highest level of aggregation, and `aggr_table_n` the aggregate table with the lowest level (the detailed fact table).
4. Specify the incompatible objects for each aggregate table in the universe. In the business layer, use the Set Aggregate Navigation command.
5. Resolve any loops in the data foundation using contexts if needed.

**Related Topics**
- About aggregate awareness
- Inserting tables into the data foundation
- Defining the SQL expression for an object
- About @Aggregate_Aware
- Setting aggregate navigation
- Resolving loops

### 10.13.2 Setting aggregate navigation

1. Open the business layer in the editor by double-clicking the business layer name in the Local Projects View.
2. From the information design tool main menu, select **Actions > Set Aggregate Navigation**.
   In the "Aggregate Navigation" dialog box, you specify which tables contain objects that are not compatible with aggregate tables containing objects optimized for aggregate awareness:
   - · If the object is at the same level of aggregation or higher, it is compatible with the aggregate table.
   - · If the object is at a lower level of aggregation, it is incompatible.
   - · If the object is unrelated to the aggregate table, it is incompatible.

**Note:**
A measure summed by year is at a higher level of aggregation than a measure summed by quarter.

3. Click an aggregate table in the left pane.
4. In the right pane, select all incompatible objects.
5. Repeat the above steps for each aggregate table in the data foundation.

**Note:**
The dialog box also features a Detect Incompatibility button that can guide you in the process of specifying incompatible objects. When you click a table and then click this button, objects considered as incompatible are automatically selected. Consider the incompatible objects proposed by Detect Incompatibility as suggestions, not final choices.

6. When all incompatible objects for all the tables are specified Click **OK**.
7. Save the business layer by clicking the Save icon in the main tool bar.

**Related Topics**
- About aggregate awareness
10.14 About refreshing business layers

**Refreshing an OLAP business layer**
For business layers based on an OLAP cube, the "Refresh Business Layer" wizard detects changes in the OLAP cube and applies the changes to the business layer.

On the "Select Options" page, you can select which types of changes the wizard should detect in the cube.

Based on the detections, the wizard lists possible update actions in the "Select Actions" page. You can select which update actions you want to apply to the business layer.

Before applying the changes, the wizard displays a summary of update actions on the "Refresh Summary" page. You can save the summary to a file. You can go back and modify your selection before finishing the wizard.

A summary list is shown with the changes that are proposed in the business layer based on the changes in the cube structure. You can clear and select proposed changes before applying the update.

**Note:**
A refresh can be undone using the undo action. Undo will recover the business layer to its state before the refresh. To undo, from the information design tool main menu, select **Edit > Undo**.

**Refreshing an SAP NetWeaver BW multisource-enabled business layer**
When you add tables from the SAP NetWeaver BW connection to an existing data foundation, you can refresh a business layer based on the data foundation using the Insert Candidate Objects command.

**Refreshing relational business layers**
To update the business layer with changes to the underlying data foundation, you must manually delete and insert objects. To insert objects for new tables, in the Business Layer Editor, you can drag and drop tables from the data foundation view to the business layer browsing pane.

To identify objects based on tables that have been deleted from the data foundation, follow these steps:

1. In the Local Projects View, right-click the business layer and select **Refresh**.
2. Open the business layer in the editor. If an object in the business layer is based on a table that was deleted from or changed in the data foundation, in the **SQL Definition** tab of the object properties, the **Extra Tables** field label is in red and the field contains the message **[Unresolved table]**.

**Related Topics**
- Refreshing an OLAP business layer
- Inserting candidate objects
- Inserting dimensions directly from the data foundation
10.14.1 Refreshing an OLAP business layer

Use the "Refresh Business Layer" wizard to update a business layer based on changes in the OLAP cube since the business layer was created, or since the last refresh.

1. Open the business layer by clicking the business layer name in the Local Project View.
2. From the information design tool main menu, select Actions > Refresh Structure.
3. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help icon.

Related Topics
• About refreshing business layers

10.14.2 Inserting candidate objects

For business layers that are based on multisource-enabled data foundations and SAP NetWeaver BW connections, the Insert Candidate Objects command detects tables that have been added to the data foundation since the business layer was created, or since the business layer was last updated with the Insert Candidate Objects command.

For each new table found, the command proposes candidate business layer objects to be inserted into the business layer.

1. Open the business layer by clicking the business layer name in the Local Projects View.
2. Right-click the business layer name in the Business Layer pane and select Insert Candidate Objects. A list of business layer objects displays. The objects are grouped in folders based on the source tables. Candidate objects are highlighted and pre-selected.
3. Select the objects in the list to insert into the business layer.

Note:
You can select objects that exist in the business layer. In this case, the definitions from the data foundation will overwrite the existing business object definitions.

4. To insert the selected objects into the business layer, click Finish, and save the business layer.

Note:
The insertion can be undone using the Edit > Undo command on the main menu.

Related Topics
• Universes on SAP NetWeaver BW
10.15 About computing statistics for optimized query execution

For queries on multisource-enabled universes, you can obtain the best performance if accurate table and columns statistics are available for the data federation service. The cost-based optimizer of the data federation service uses these statistics to determine the optimal join method and order.

The **Compute statistics** command optimizes query execution because it allows you to compute and store statistics in the repository for the universe.

You should compute statistics periodically for tables that might change in volume or for which column values are changing frequently.

The following statistics are generated for the optimization process:

- The table row count
- The number of distinct values for the columns

You set the following options:

- Select all tables and columns that were computed before a certain date
- Select all tables and columns that were never computed
- Select every table and column
- Unselect every table and column

**Related Topics**

- Computing statistics for a multisource-enabled universe

10.15.1 Computing statistics for a multisource-enabled universe

You can compute statistics only for universes based on a multisource-enabled data foundation.

1. Do one of the following:
### Working with business layers

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
</table>
| To compute statistics from the published universe | In the Repository Resources View, open a session on the repository where the universe is published. 
Right-click the universe and select **Compute Statistics**. |
| To compute statistics from the business layer | Open the business layer in the editor by clicking the business layer name in the Local Projects View. 
Right-click the business layer name in the **Business Layer** pane and select **Compute Statistics**. |

2. In the "Compute Statistics" dialog box, select the tables and columns for which to compute statistics. When you select a table, all the columns in the table are selected.

3. Click **Compute**.
   The statistics are calculated and stored in the repository. For large databases, this process can take several minutes or longer. While the computation is in progress, you can close the window and perform other tasks in the information design tool.

**Related Topics**
- [About computing statistics for optimized query execution](#)
Using the Query Panel

Use the Query Panel to build, test, and preview the results of queries on a business layer or published universe.

In the information design tool, you can start the Query Panel in the following ways:

• To insert a query into the business layer.
• To open an existing query in the business layer.
• To run a query on a universe published in a repository.

Related Topics
• Inserting and editing a query in the business layer
• Running a query on a universe published in a repository
• How to build a query

11.1 How to build a query

This procedure assumes that you have opened the Query Panel on a business layer or published universe. See the related topic on Using the Query Panel.

You can use this procedure to run queries on published universes, but to save the query you must start the Query Panel from the Business Layer Editor Queries pane.

For links to more detailed information about each step, see the Related Topics.

1. To select the objects you want to include in the query, drag objects from the business layer on the right into the Result Objects pane.
2. For hierarchy result objects, select members to include or exclude in the results. To open the Member Selector, click the arrow to the right of the hierarchy object name:
3. To filter the results of the query, drag objects from the business layer into the Filter Objects pane.
   If a mandatory filter is defined on an object, the filter is triggered when you add the object to the Result Objects pane. The mandatory filter is visible in the query script, but not in the Filter Objects pane.
   Non-mandatory pre-defined filters are listed in the business layer. You can drag these pre-defined filters into the Filter Objects pane to limit the results. The filter is visible in the query script.
You can also build business filters, including filters that use prompts. For detailed information, see the related topics.

4. For relational universes, you can build combined queries. To open the **Combined Queries** pane, click the icon.

5. To set query properties, click the icon.

6. To see or edit the query script, click **View Script**.

7. To preview the query results, click the refresh button in the **Data Preview** pane.

   You can profile the values in the result columns. In the **Data Preview** pane, click the **Advanced Preview** icon.

   To change the layout of hierarchical data, click the **Result set display options** icon and select an option from the list:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flat layout</td>
<td>Displays repeated values for a level in every row.</td>
</tr>
<tr>
<td>Hierarchical layout</td>
<td>Displays repeated values once for a level.</td>
</tr>
</tbody>
</table>

8. To save the query, click **OK**.

   The **OK** button is only available when running the Query Panel from the Business Layer Editor. The query is saved in the business layer and can be executed or edited from the **Queries** pane.

**Related Topics**
- Using the Query Panel
- About the Member Selector
- How to build a business filter
- Filtering data using prompts
- Setting query properties
- Viewing and editing the query script
- Profiling column values in the query panel

### 11.2 About the Member Selector

The Member Selector lets you visualize and select members in a hierarchy. Use the Member Selector to:

- Select the members you want to appear in the query result set.
- Define members that will be excluded from queries.
• Define prompts to allow the selection of members to appear in the query each time you run the query.
• Select the members for a named set.
• Select the members when defining a Business Security Profile filter.

You open the Member Selector from hierarchy objects that you include in queries in the Query Panel. The Member Selector opens automatically when you edit named sets or filters for a Business Security Profile on a hierarchical business layer.

Related Topics
• Selecting hierarchy members
• About selecting hierarchy members

11.2.1 About selecting hierarchy members

In the Member Selector, you can select members in several ways:
• Select members explicitly in the hierarchy. For example, explicitly select the [California] and [Los Angeles] members of the [Geography] hierarchy.
• Select members implicitly using hierarchy relationships. For example, to select US states, you can select the child members of the [US] member.
• Select members included in a named set, for example Top Cities by Revenue, to include the cities that generate the most revenue.
• Select all members in a hierarchy level.
• Select all members up to a certain level in the hierarchy.
• Select calculated members.

The Member Selector contains three tabs:

<table>
<thead>
<tr>
<th>Tab</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Members</td>
<td>Displays the members arranged hierarchically. Use this tab to select members explicitly, by hierarchical relationships, and by specifying all members up to a given level.</td>
</tr>
<tr>
<td>Metadata</td>
<td>Shows the hierarchy levels (if the hierarchy supports named levels), named sets, and calculated members.</td>
</tr>
<tr>
<td>Prompts</td>
<td>Lets you define and modify prompts.</td>
</tr>
</tbody>
</table>

For information on how to select, display, search for, and sort hierarchy members, see the related topics.
11.2.2 Opening the Member Selector in the Query Panel

1. In the Query Panel, add the hierarchy object to the **Result Objects** pane.
2. To open the Member Selector, click the arrow to the right of the hierarchy object name:
3. You can now select members in the hierarchy for inclusion or exclusion in a query. For descriptions of different ways to select members, see the related topic.

**Related Topics**
- About selecting hierarchy members

11.2.3 Selecting hierarchy members

1. In the Member Selector, click the **Members** tab to display the hierarchy members.
2. Select members in the hierarchy display.
3. To select all members in the hierarchy, click the **Select** icon, and select **Select All**.
4. To select all members up to a specified level in the hierarchy, click the **Select** icon. You can identify the level in two ways:
5. When you complete your selection, click **OK**.

The selected members appear below the hierarchy object in the **Result Objects** pane of the Query Panel. When you run the query, only those members are included in the query result.

**Related Topics**
- Opening the Member Selector in the Query Panel

### 11.2.4 Selecting members by hierarchy relationship

1. In the Member Selector, click the **Members** tab to display the hierarchy members.
2. In the hierarchy, right-click the member for which you want to define the hierarchy relationship.
3. Select the relationship function from the menu:

   **Note:**
   - **Children/Descendants** and **Parents/Ancestors** are mutually exclusive pairs. You cannot select both the children and the descendants of a member, and you cannot select both the parents and the ascendants of a member.

<table>
<thead>
<tr>
<th>Relationship Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Self</strong></td>
<td>Includes only the selected member. This is the default setting.</td>
</tr>
<tr>
<td><strong>Children</strong></td>
<td>Includes members one level below the selected member that have the selected member as their parent. The selected member is not included.</td>
</tr>
<tr>
<td><strong>Descendants</strong></td>
<td>Includes all members at all levels below the selected member. The selected member is not included.</td>
</tr>
<tr>
<td><strong>Descendants until Named Level...</strong></td>
<td>Includes the members at levels below the selected member until the named level you select. This option is only available if the hierarchy has named levels.</td>
</tr>
</tbody>
</table>
### Relationship Function

<table>
<thead>
<tr>
<th>Relationship Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Descendants until...</td>
<td>Includes the members at levels below the selected member until the number of levels you select.</td>
</tr>
<tr>
<td>Parent</td>
<td>Includes the member that is one level above the selected member. The selected member is not included.</td>
</tr>
<tr>
<td>Ancestors</td>
<td>Includes all members at all levels above the selected member. The selected member is not included.</td>
</tr>
<tr>
<td>Siblings</td>
<td>Includes members at the same level that have the same parent as the selected member. The selected member is not included.</td>
</tr>
<tr>
<td>Exclude</td>
<td>Excludes members according to the relationship function (Self/Children/Descendants/Parent/Ancestors/Siblings).</td>
</tr>
</tbody>
</table>

**Related Topics**
- Opening the Member Selector in the Query Panel

---

### 11.2.5 Selecting hierarchy members by level

To select members by level, the hierarchy must have named levels.

1. In the Member Selector, click the **Metadata** tab to display the hierarchy levels.

   **Note:**
   
   If the **Levels** folder does not display in the **Metadata** tab, the hierarchy is not level-based and you cannot select members by level.

2. Select levels in the **Levels** folder.
3. Click **OK**.

**Related Topics**
- Opening the Member Selector in the Query Panel
11.2.6 Selecting named sets

To select members by named set, the hierarchy must have at least one named set defined. Named sets are defined in the business layer of the universe.

1. In the Member Selector, click the Metadata tab to display the named sets.
   
   **Note:**
   
   If the Named Sets folder does not display in the Metadata tab, the hierarchy has no named sets defined.

2. Select named sets in the Named Sets folder.
3. Click OK.

Related Topics
• Opening the Member Selector in the Query Panel
• Inserting and editing named sets

11.2.7 Selecting calculated members

To select calculated members, the hierarchy must have at least one calculated member defined. Calculated members are defined in the business layer of the universe.

1. In the Member Selector, click the Metadata tab to display the calculated members.
   
   **Note:**
   
   If the Calculated Members folder does not display in the Metadata tab, the hierarchy has no calculated members defined.

2. Select calculated members in the Calculated Members folder.
3. Click OK.

Related Topics
• Opening the Member Selector in the Query Panel
• Inserting and editing calculated members

11.2.8 Searching for hierarchy members
Use the Search function in the Member Selector to select hierarchy members from a list of search results.

1. To open the "Member Search" dialog box, in the Member Selector Members tab, click the Search icon.
2. Enter text to search for in the Search pattern box.
   You can use wildcards in the search:

<table>
<thead>
<tr>
<th>Wildcard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Matches any string of characters</td>
</tr>
<tr>
<td>?</td>
<td>Matches any one character</td>
</tr>
</tbody>
</table>

3. To search for text in the keys, select the Search Keys radio button.
4. Click Search.
5. To select members from the search results, select the members in the Search results table.
6. Click OK.

11.2.9 Excluding hierarchy members

1. In the Member Selector, select the members that you want to exclude.
   You can select members explicitly, by hierarchy relationship, by level, by named set, and calculated members.
   The selected members are listed in the Summary pane of the Member Selector.
2. In the Summary pane, select the Exclude option next to the members or member sets you want to exclude.
3. Click OK.

Below the hierarchy object in the Result Objects pane of the Query Panel, the excluded members appear with a line drawn through the names to indicate that they are excluded from the query.

Related Topics
- Opening the Member Selector in the Query Panel
- Selecting hierarchy members
- Selecting members by hierarchy relationship
- Selecting hierarchy members by level
- Selecting named sets
- Selecting calculated members
- Searching for hierarchy members
11.2.10 Defining a prompt to select members

You can define a prompt to defer member selection to the time the query is run.

**Note:**
When selecting members in response to a prompt, you can only select members explicitly. You cannot select members by hierarchy relationship.

1. In the Member Selector, click the **Prompt** tab.
2. Select **Enable Parameter** to defer member selection to when the query is run.
   
   You cannot access the other tabs in the Member Selector when the **Enable Parameter** option is selected.
3. Enter text for the prompt in the **Prompt Text** box.
4. If you want the prompt to select the previously-selected values by default when it displays, select **Keep last values selected**.
5. To define default values for the prompt, select **Set default values** and click **Edit**. In the "Select Parameter Values" dialog box, select default values for the prompt and click **OK**.
6. Click **OK**.

**Related Topics**
- Opening the Member Selector in the Query Panel

11.2.11 Showing selected members in the Member Selector

In the Member Selector **Members** tab, you can click the **Expand tree to show selections** icon to show the selected members in the hierarchy display.

The display automatically expands to show the following members:

- Explicitly selected members.
- Members used to select related members. The related members implicitly selected are not necessarily shown. For example, if the member called France was used to select its children, the tree view expands to show France. If the node France contains no explicitly selected members, the node is not expanded the show the implicitly selected children.

**Tip:**
The **Expand tree to show selections** command does not collapse nodes that are already expanded. To reduce the length of the display, close all open nodes in the hierarchy display before clicking the icon.
11.2.12 Sorting hierarchy members

By default in the Member Selector, hierarchy members are displayed sorted in the order they are stored in the database. To help find members in the hierarchy, you can sort the display in ascending or descending alphabetical order.

In the Member Selector Members tab, click the Sort order icon and select the desired sort order. The members are sorted locally in the Member Selector. Display of members in the query is not affected.

11.2.13 Setting display options

By default, the Member Selector displays hierarchy member captions. You can set the display options to display unique names, or both captions and unique names.

In the Member Selector Members tab, click the Member display options icon and select the desired display option.

11.2.14 Showing estimated child count

The Member Selector estimates the number of children for each member. By default, the estimates are hidden. You can show the estimated child count in the hierarchy display.

In the Member Selector Members tab, click the Show/Hide estimated child count icon to toggle the display of child counts.
11.3 Filtering data in the Query Panel

11.3.1 How to build a business filter

This procedure assumes you are building a business filter either in the Query Panel or the Edit Business Filter dialog box.

Business filters are filters based on objects in the business layer. They limit the data returned in the query.

Query filters have the following structure: filtered object, operator, operand.

For example, in the filter [Country] InList (US;France), the [Country] dimension is the filtered object, InList is the operator, and the list of values (US;France) is the operand. The filter removes all values of [Country] other than US and France from the query result.

The following table describes the components of a filter:

<table>
<thead>
<tr>
<th>Filter Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filtered object</td>
<td>The filtered object is the business layer object whose values are filtered. Dimensions, attributes, measures, hierarchies, and hierarchy levels can be used as filtered objects.</td>
</tr>
<tr>
<td>Operator</td>
<td>The operator is used to compare the filtered object with the operand. For example, the Equal To operator retains only those values of the filtered object that correspond exactly to the value of the operand.</td>
</tr>
<tr>
<td>Operand</td>
<td>The operand supplies the value or values used to filter the filtered object.</td>
</tr>
</tbody>
</table>

1. Drag an object from the business layer to the Filter Objects pane. This is the filtered object.
2. In the Filter Objects pane, select an operator from the list.
3. In the Filter Objects pane, select an operand from the list.

Depending on the type of business layer and purpose of the filter, the following types of operands are available:
<table>
<thead>
<tr>
<th><strong>Operand type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant</td>
<td>Use the constant operand to enter values directly into the filter. For example, you can use a constant to enter France into the filter [Country] Equal To France. You can also enter the @Variable function to retrieve the value of a system variable or User Attribute. For example, to filter on the current user login name, enter the constant operand as @Variable('BOUSER'). For more information about @Variable, see the related topic.</td>
</tr>
<tr>
<td>LOV</td>
<td>Use the list of values (LOV) operand to select values from the list associated with the filtered object. For example, if the filtered object is [City], you can use the list of values to select one or more of the cities associated with the object.</td>
</tr>
<tr>
<td>Object</td>
<td>Use the object operand to specify an object in the business layer. Drag and drop a business layer object to the operand position when defining the filter. <strong>Note:</strong> You cannot select an object as an operand on some OLAP data sources or if the filtered object is a hierarchy.</td>
</tr>
<tr>
<td>Prompt</td>
<td>Use the prompt operand when you want to be prompted for a value when the query is refreshed. See the related topic about filtering using prompts. <strong>Note:</strong> Prompt operands are not available if the business filter is defined for a Business Security Profile.</td>
</tr>
</tbody>
</table>

4. If you want to filter data on more than one criteria, add an additional filter by dragging another object to the **Filter Objects** pane.  
By default, the filters are combined with the AND operator. To use the OR operator, double-click the box with the **And** operator. **Note:** The OR operator is not supported for OLAP data sources.

5. If you want to nest query filters, drag another business object and drop it onto an existing query filter in the **Filter Objects** pane.  
Nesting query filters allows you to create more complex filter conditions than is possible when you combine filters at the same level. When you nest filters, you set the order in which they are evaluated. Nesting filters only makes sense once you have defined two filters at the same level.

**Related Topics**
- Using the Query Panel
- Inserting and editing filters
11.3.2 Filtering data using prompts

A prompt is a special type of query filter. It is a dynamic filter that displays a question every time you refresh the data in a query. You answer prompts by either typing or selecting the values you want to view before you refresh the data. The query then returns only the values you specified.

Prompts allow multiple users viewing a single document to specify a different sub-set of the database information and display it in the same report tables and charts. Prompts also reduce the time it takes for the data to be retrieved from the database.

When you define a prompt query filter, you can either build a new prompt, or use an existing prompt defined as a parameter in the business layer.

If you define more than one prompt in a query, you can change the order in which prompts are presented. Change prompt order in the query properties.

Related Topics
- Building a new prompt to filter data
- Using an existing prompt to filter data
- Setting query properties
- About parameters

11.3.2.1 Building a new prompt to filter data

This procedure assumes you are building a business filter either in the Query Panel or the Edit Business Filter dialog box.

1. Drag the object you want to filter with a prompt and drop it onto the Query Filters pane.
   The query filter appears in outline in the Query Filters pane. The outline shows the filtered object, the operator and the type of filter applied to the object. (By default the filter is a constant.)

2. Select the filter operator from the list.
   **Note:**
   The list of available operators depends on the type of filtered object.

3. Click the arrow at the right of the outline query filter and select Prompt from the menu to filter the object using a prompt.
The **Edit Prompt** dialog box appears and the **New Parameter** option is selected by default.

4. Edit the prompt question in the **Prompt Text** box.

5. Select **Prompt with List of Values** to allow the user to select from a list of values when answering the prompt.

   The option is only available if the filtered object has an associated list of values in the universe.

6. Select **Select only from list** to restrict the user choice to values from the list of values.

   You can select this option only if the **Prompt with List of Values** option is selected.

7. Select **Keep last values** if you want the prompt to propose the last value that the user selected on the previous refresh. The first time the query is run, the default value (if set) is proposed.

8. Select **Optional prompt** to make the prompt optional. If the user does not supply a value for an optional prompt, the prompt is ignored.

9. Select **Set default values** if you want the prompt to propose values by default when it displays.

   a. To enter or select the default values, click **Edit**.

   b. If the filter object has an associated list of values, select the default values from the list.

   c. If the filter object has no associated list of values, enter default values.

   d. Click **OK** to save the default values.

10. Click **OK** to save the new prompt definition.

**Related Topics**

- How to build a business filter
- About lists of values

### 11.3.2.2 Using an existing prompt to filter data

This procedure assumes you are building a business filter either in the Query Panel or the Edit Business Filter dialog box.

1. Drag the object on which you want to apply a prompt and drop it onto the **Query Filters** pane.

   The query filter appears in outline in the **Query Filters** pane.

2. Select the filter operator from the list.

   **Note:**
   The list of available operators depends on the type of filtered object.

3. Click the arrow at the right of the Query Filter and select **Prompt** from the menu.

4. In the **Edit Prompt** dialog box, select the **Use Universe Parameter** option.

5. Select an existing parameter.

   The list displays only those universe prompts that are compatible with the object you are filtering. For example, the filtered object and the universe prompt must have the same data type.
6. Click **OK** to save the prompt definition.

**Related Topics**
- How to build a business filter
- About parameters

### 11.4 Setting query properties

1. In the Query Panel, click the **Query properties** toolbar button.
2. Edit the query property settings as required.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve duplicate rows</td>
<td>When this option is selected, the query returns all related rows, even if there are duplicate rows. If you do not want duplicate rows in the result set, unselect this option.</td>
</tr>
<tr>
<td>Retrieve empty rows</td>
<td>An empty row occurs typically in multidimensional queries when the data for the intersection of two or more dimensions does not exist. When this option is selected, the result set includes rows that can contain empty cells. When this option is unselected, the result set contains only rows with non-empty cells.</td>
</tr>
<tr>
<td>Max retrieval time</td>
<td>Defines the maximum time (in seconds) that a query can run before the query is stopped. By default, this value is the same as the Limit Execution Time parameter in the universe parameters. When you set this value to 0, this option is disabled. When the Limit Execution Time parameter is lower than this setting, the Limit Execution Time value is used for limiting the query execution time.</td>
</tr>
<tr>
<td>Max rows retrieved</td>
<td>Defines the maximum number of rows of data that are displayed when the query runs. The query retrieves all the possible rows, but only displays the first n rows, where n is the maximum number of rows set for this parameter. The administrator can override this setting in the user security profile settings.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Sample result set</td>
<td>This parameter (when supported by the database) samples n database rows, where n is the value set for the sample result set. This method is faster than using the Max rows retrieved parameter.</td>
</tr>
<tr>
<td>Reset contexts on refresh</td>
<td>This is only available on relational universes. When this option is selected, when a user refreshes a query that contains contexts, the user must choose the context(s). The user can clear the previously selected contexts by clicking Clear Contexts. When this option is not selected, the query is refreshed using the original contexts. If the contexts have been edited since the last run for the query, the user must choose the contexts again since the query is considered as a new query.</td>
</tr>
<tr>
<td>Prompt order</td>
<td>When there are several prompts in a query, use this feature to set the order in which prompts are executed in a query. Click a prompt and use the up or down arrow to change the position of the prompt.</td>
</tr>
</tbody>
</table>

3. Click **OK** to close the "Query Properties" and save the changes.

### 11.5 Viewing and editing the query script

You can view the query script of a the query you build in the Query Panel. For relational universes, you can also edit the query script.

1. In the Query Panel, click **View script**.
   - The query script displays in the "Query Script Viewer".
2. For OLAP universes, your only option is to click **OK** to close the "Query Script Viewer".
3. For relational universes, to edit the query script, select the **Use custom query script** option.
   a. In the "Query script" pane, edit the query.
   b. Click **Validate** to check the script syntax.
   c. Click **Undo** to undo the last edit to the script.
   d. Click **OK** to save and use the edited query script.
      - The edited query script is used until you unselect the **Use custom query script** option, or close the Query Panel.
4. To use the query script generated by the Query Panel, select the **Use the query script generated by your query** option.
5. Click **OK** to save the changes.
11.6 Profiling column values in the query panel

You can profile the values for a column in the query results. Profiling shows graphically (in a pie or bar chart) the number of occurrences of each value of a column. If the column has a filter defined, the filter is applied.

1. Open a query in the Query Panel and refresh the results.

2. In the Data Preview pane, click the Advanced Preview icon 📊.

   The Profile Column Values window appears. To see what you can do in this window, see the related topic.

Related Topics

• Showing and profiling values in a data source
• Using the Query Panel
12.1 Running check integrity

Use the **Check Integrity** function to verify aspects of the design of your universe or its elements, for example the data foundation, business layer, parameters, and lists of values. You select pre-defined rules that check the validity of the SQL and MDX expressions, as well as for adherence to design restrictions. Running a check integrity helps to avoid problems when running queries and reports on the published universe.

You can set a background integrity check that performs the integrity check automatically whenever you save a resource. For more information, see the related topic about setting preferences for check integrity.

You can run a check integrity at any time for different objects and resources in the information design tool:

- Resources (data foundations, business layers, connections and shortcuts) in the Local Projects View
- Elements in the data foundation and business layer (tables, contexts, business layer objects, queries, parameters, lists of values) in the editor
- Published universes in the Repository Resources View
- Published universes in the Security Editor (to check the validity of security profiles).

1. Right-click the resource or object you wish to run a check integrity for and select **Check Integrity**.
2. In the left-hand pane of the "Check Integrity" dialog box, select the rules you want to apply.
3. Click **Check Integrity**.

The results of the check integrity are listed in the right-hand pane of the "Check Integrity" dialog box. The results of a rule check can have one of three severities:

<table>
<thead>
<tr>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>The check has detected something that will not work. You must resolve the problem.</td>
</tr>
<tr>
<td>Warning</td>
<td>Warning about a missing object (for example, a missing key or a missing link).</td>
</tr>
<tr>
<td>Information</td>
<td>The check was OK. A green check mark is displayed next to the rule.</td>
</tr>
</tbody>
</table>

**Note:**
You can change the severity of the results of a rule in the information design tool preferences.
4. To save the results in a text file, click Export.
5. When you are finished reviewing the results, click OK.

After closing the "Check Integrity" dialog box, the results of the check integrity can be reviewed in the Check Integrity Problems View until you run the next check integrity. For more information, see the related topic.

**Related Topics**
- Setting preferences for check integrity
- Reviewing check integrity problems

### 12.2 Reviewing check integrity problems

1. From the information design tool main menu, select **Window > Check Integrity Problems**.
   The Check Integrity Problems View opens with a list of the results of the latest check integrity.

2. To correct a problem, double-click the result in the list.
   The editor opens for the object concerned in the result. For example, if the result concerns a problem with table Customer, the Data Foundation Editor opens with the table Customer highlighted.

The list of results remains in the Check Integrity Problems View until you close the view, or run another check integrity.

**Related Topics**
- Running check integrity
Showing and profiling values in a data source

You can show and profile values in the underlying data source for data foundation tables and columns, and business layer objects. This topic describes what you can do when showing or profiling values.

**Note:**
Unless you are showing values from the connection, any column filters defined in the data foundation are applied when retrieving values.

**What you can do when showing values**
- Limit the number of rows returned from the data source: Enter a number in **Max Rows**.
- See the query script: Click **Show expression**.
- Re-order the columns in the display: Drag and drop column headers to the new location in the table.
- Sort rows by column: Click the column header to sort the rows by the column value in ascending or descending order.
- Filter rows by column: Click **Add Filter** and construct a filter for one or multiple columns using the filter value selector.
- Filter results to rows that contain a character or group of characters in any of the columns: Enter the characters to filter on in the **Enter your filter** text box. You can use the * character as a wildcard, for example:
  - If you enter B in the filter text box, only rows with a column value that contains the character B display.
  - If you enter B*, only rows with a column that contains a value starting with B display.
  - If you enter *B, only rows with a column that contains a value ending with B display.
- See distinct values for a selected column: Click the **Distinct Values** tab and select a column.
- Export the results to a local file (.csv or .xml format): Click **Save as File**.

**What you can do when profiling column values**
- Limit the number of rows returned from the data source: Enter a number in **Max Rows**.
- See the query script: Click **Show expression**.
- Change the chart type: Select the **Pie Chart** or **Bar Chart** option and click **Refresh Chart**.
- Change the number of values to profile: Enter a number in **Distinct values to display** and click **Refresh Chart**.
- See distinct column values in a list: Click the **Values** tab.

**Related Topics**
- Showing table values
- Showing column values
- Profiling column values
• **Showing business layer object values**
14.1 About resource dependencies

For any local resource, you can view its relationship to other local resources, and any dependent universes published in a repository.

Making changes to a resource, such as deleting it from a local project, moving it to another local project, renaming or updating it, may impact other resources that depend on it. You are warned of the impact before you delete or move a resource.

To help you understand the impact of changes and plan your work, commands exist that show the dependencies between resources and their objects.

Dependencies between local resources

The Show Local Dependencies command shows the dependencies between resources in a local project.

When you select a resource, two tabs display the dependent and referenced resources: The Dependent Resources tab lists resources in the same local project that depend on the selected resource. The Referenced Resources tab lists the resources in the same local project that are referenced by the selected resource. See example 1.

The paths to referenced resources are relative, not absolute. That means that if you rename a resource, the referenced resources are assumed to be in the same folder. If the referenced resources are in a different sub-folder, the reference is broken. See example 2.

You can also show local dependencies for any object in the business layer. In this case, for relational business layers, the referenced resources include the data foundation tables and columns that the object is based on.

Example: 1: Show local dependencies

The local project Demo contains the following resources:

- Demo_Local_Connection.cnx
- Demo_Data_Foundation.dfx
- Demo_for_Accounting.blx
- Demo_for_Sales.blx
You want to list all the resources that would be impacted if you change the Demo_Data_Foundation. In the Local Projects View, you select the Show Local Dependencies command on Demo_Data_Foundation.dfx. The following dependencies are displayed:

<table>
<thead>
<tr>
<th>Dependent Resources</th>
<th>Referenced Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>\Demo\Demo_for_Accounting.blx</td>
<td>\Demo\Demo_Data_Foundation.dfx</td>
</tr>
<tr>
<td>\Demo\Demo_for_Sales.blx</td>
<td></td>
</tr>
</tbody>
</table>

The two business layers contain references to the data foundation and might contain invalid references if you delete or change Demo_Data_Foundation, so they are listed in the Dependent Resources tab.

The data foundation references the connection. Any changes to Demo_Local_Connection could impact Demo_Data_Foundation and its dependent resources.

Now you want to show the dependencies for Demo_Local_Connection:

<table>
<thead>
<tr>
<th>Dependent Resources</th>
<th>Referenced Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>\Demo\Demo_Data_Foundation.dfx</td>
<td></td>
</tr>
<tr>
<td>\Demo\Demo_for_Accounting.blx</td>
<td></td>
</tr>
<tr>
<td>\Demo\Demo_for_Sales.blx</td>
<td></td>
</tr>
</tbody>
</table>

Notice that Demo_Data_Foundation and its two dependent business layers are listed as dependent resources. Since a connection is the first resource created when building a universe, no resources reference the connection.

Example: 2: Renaming resources

The local project OLAP_Demo contains a folder with the following resources:

Folder_One

• OLAP_Local_Connection.cnx
• OLAP_Business_Layer.blx

Rename OLAP_Business_Layer.blx to OLAP_New_Business_Layer.blx and show local dependencies. In the Local Projects View, you select the Show Local Dependencies command on OLAP_New_Business_Layer.blx. The following dependencies are displayed:
Even though the business layer was renamed, the OLAP_Local_Connection is listed as a referenced resource because it is in the same folder.

Now, create Folder_Two in the OLAP_Demo project and copy OLAP_New_Business_Layer.blx to Folder_Two. In Folder_Two, rename OLAP_New_Business_Layer.blx to OLAP_New2_Business_Layer.blx and show local dependencies.

The reference to OLAP_Local_Connection is broken because when renaming to OLAP_New2_Business_Layer.blx, the information design tool assumes referenced resources are in the same folder.

### Dependencies between data foundation and business layer objects

When editing a data foundation, you can show local dependencies for any table or column. A list of dependent business layers displays. You can then display a list of the objects per business layer that depend on the selected data foundation object.

When editing a relational business layer, in the data foundation view, you can select tables and columns and show associated objects. This will highlight in the business layer all objects which reference the selected data foundation objects.

### Dependencies between local resources and repository resources

The Show Repository Dependencies command lists the universes published in a particular repository that are referenced by the selected local resource.

### Related Topics

- Showing local dependencies
- Showing local dependencies in the data foundation
- Showing associated objects
- Showing repository dependencies

### 14.2 Showing local dependencies

To show resources in the local project that depend on a selected resource:
1. In the Local Projects View, select the resource for which you want to show the dependencies.
2. Right-click and select **Show Local Dependencies**.

   The **Dependent Resources** tab lists the resources in the same local project that contain references to, or depend on, the selected resource.

   The **Referenced Resources** tab lists the resources in the same local project that are referenced by the selected resource.

**Related Topics**
- About resource dependencies

### 14.3 Showing repository dependencies

To show universes in a repository that depend on a selected resource:

1. In the Local Projects View, select the resource for which you want to show the dependent resources published in a repository.
2. Right-click and select **Show Repository Dependencies**.
3. Select a session for the repository system where the resources are published, and log in.

   The published universes in the repository that reference the selected resource are listed.

**Related Topics**
- Opening a session
- About resource dependencies
15.1 About publishing resources

Publication is the last step in the universe creation process. Using the Publish Universe Wizard, you publish a business layer to either your local file system or a repository.

When you publish a business layer, the wizard exports the business layer and the resources it references (local connection, connection shortcuts, and data foundation), and creates a universe which is then available to users of query, reporting and analysis tools.

Publishing locally
Only business layers built on local connections can be published locally. This can be a business layer based on a local OLAP connection, or a business layer based on a single-source data foundation with a local connection.

The published universe is saved in the local file system folder that you specify.

Publishing to a repository
To secure a universe, you must publish it first to a repository on a Central Management Server (CMS). The universe inherits the object-level security and user security rights defined for the CMS. The data and metadata in the universe are secured by defining security profiles in the information design tool Security Editor.

When you create a connection in a local project, it is an unsecured local connection and must be published before you can publish a business layer that references the connection. To secure a connection, publish it to a repository on a CMS. The Publish Connection Wizard creates the secured connection and provides a connection shortcut for the local project.

To browse and manage resources once they are published to a repository, use the Repository Resources View.

Editing published resources
You cannot edit a published universe directly in the information design tool. To work on it, you must retrieve it using the Retrieve Universe Wizard. The wizard retrieves the universe from the local folder or the repository, separates it into the business layer and the resources it references (local connection, connection shortcuts, data foundation), and creates these resources in a local project where they can be edited.

Connections can only be published to a repository. To edit a published connection, you must edit it from the Repository Resources View.
15.2 Publishing a universe

To publish a universe to a repository, the business layer must reference one or more secured connection shortcuts. All shortcuts must reference connections defined in the repository where the universe is to be published.

Note:
If the business layer references a local connection and you want to publish to a repository, first publish the connection and change the connection reference in the data foundation (relational), or in the business layer (OLAP) to use the connection shortcut.

To publish a universe locally, the business layer must reference only a local connection that is not secured in any repository.

Recommended actions before publishing a universe:

- Save the business layer and all the resources it references.
- If the business layer references resources that are shared, synchronize the project to ensure that all changes are taken into account in the published universe.
- Check integrity of the business layer and, if applicable, the data foundation. The Publish Universe Wizard gives you the option to run a check integrity before publishing.

1. Start the Publish Universe Wizard:
   - To publish to a repository, select the business layer in the Local Projects view, right-click the business layer and select Publish > To a Repository.
   - To publish to a local folder, select the business layer in the Local Projects view, right-click the business layer and select Publish > To a Local Folder.

2. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help icon in the lower left corner.

   In the Local Projects View, select the business layer and select Publish > To a Local Folder.

The universe is created as a .unx file in the local folder or the repository.

Related Topics
- Publishing a local connection to the repository
- Running check integrity
- Opening a session
15.2.1 Selecting a repository folder

When publishing or retrieving resources on a repository, the wizard displays the folders in the repository in the left-hand pane. The table in the right-hand pane lists the resources in the folder.

When publishing a resource to a repository, navigate to the repository folder in the navigation tree in the left-hand pane. You can insert a folder.

When retrieving a published universe, navigate to the repository folder in the left-hand pane, and select the universe in the universe list in the right-hand pane.

Note:
By default, the resources are retrieved into the local project and are secured locally by requiring you to enter the CMS authentication when opening a retrieved data foundation or business layer.

To remove the local security requirement, select the **Save for all users** option.

15.2.2 Selecting a local folder

When publishing or retrieving a resource in a local folder, the wizard prompts you for a local folder.

1. Enter the path to a folder accessible from your local machine.
2. To browse the file system and select a folder, click **Browse**.

15.3 Publishing a local connection to the repository

1. To start the Publish Connection Wizard, select the connection in the Local Projects View, right-click the connection and select **Publish Connection to a Repository**.
2. Follow the instructions on the wizard pages. For more information about what to do on a particular page, click the help icon in the lower left corner.

The connection is published in the repository. The local connection is deleted from the Local Projects View. You have the choice to create connection shortcut in the local project. To publish a business layer based on this connection, edit the business layer or data foundation to refer to the new shortcut.
Related Topics

• Opening a session
• About connection shortcuts
• Changing a connection in a data foundation
• Changing the data source of a business layer
• Synchronizing a project
Managing security

16.1 About universe security

Universe security starts when the universe is published to a repository on a Central Management Server (CMS). Published universes are stored in the Universes folder and secured connections are stored in the Connections folder.

You secure universes based on the users and groups that are defined in the system repository using the Central Management Console (CMC).

As a first level of security, using the CMC, you grant the right to access specific folders, resources, universes, and connections in the repository to specific users and groups. How to define these rights is described in the SAP BusinessObjects Business Intelligence platform Administrator's Guide.

You define another level of security using the information design tool Security Editor. You can restrict the data that is returned in a query using query limits and controls, filters, and row restrictions. You can also grant or deny access to objects and views in the business layer. To create this level of security, you define security profiles for the universe and assign these profiles to users and groups. The basics of how security profiles work are presented in this topic.

Security profiles

A security profile is a group of security settings that apply to a universe published in the repository. The settings control the data that is displayed and modify the parameters defined in the data foundation and/or business layer. Once the profile is assigned to a user or a group, the settings in the profile determine what objects, data, and connections the user sees when connecting to the universe. There are two types of profiles:

- Data Security Profiles have security settings defined on objects in the data foundation and on data connections.
- Business Security Profiles have security settings defined on objects in the business layer.

Multiple profiles can be defined for each universe. The profiles are saved in the repository.

How profiles work

A user of query and reporting tools who has been granted access to a universe using the CMC, and who has no security profiles assigned or inherited, can see all the objects in the universe and all the data returned by those objects.

When you assign a profile to the user, the security settings defined in the profile are applied whenever the user runs a query on the universe.
In the information design tool, security profiles are applied when you run a query from the Repository Resources View or the Security Editor. They are applied according to the username you used to open the CMS session. When you run a query from the Business Layer Editor, security profile settings do not apply.

**How multiple profiles are handled**

You can assign more than one profile to a user or group. A user might be assigned a profile and also inherit profiles from groups. When more than one profile is assigned to a user, the profiles are aggregated to produce a single group of settings called the net profile.

Aggregation follows priority and restriction levels that you can modify in the Security Editor. You can also see which profiles a user or group inherits, and preview net profiles for a user or group.

**Profile maintenance**

Profiles are stored independently from the universe itself: changes in the data foundation or business layer of the universe do not affect the profiles when the universe is republished. Similarly, changes in a profile are independent of assignments, so you do not have to reassign a profile when it is modified. It remains assigned, including any changes.

If you republish a universe, run a check integrity on the universe to flag any discrepancies between the universe and its security profiles.

Profiles created for a universe are deleted when the universe is deleted.

**Related Topics**

- Data Security Profile settings
- Business Security Profile settings
- Security profile aggregation
- Displaying profiles assigned to a user and previewing net profiles
- Running a query on a universe published in a repository
- About the Security Editor

**16.2 About securing resources in the information design tool**

No authentication is required to start the information design tool.

A user can create and edit unsecured resources (data foundations, business layers, connections) in the Local Projects view. The resources are saved in a local project.

Resources are secured when a user shares a local project and its resources, or publishes universes or connections to a repository. Shared projects and published resources are stored securely in a repository on the Central Management Server (CMS).

Application rights are granted in the Central Management Console (CMC). How to define these rights is described in the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*. 
A user with appropriate rights can retrieve a published universe from the repository for editing. Resources can also be retrieved from a shared project during project synchronization. In both cases, the resources are retrieved into the local project and are secured locally by requiring the user to enter the CMS authentication when opening a retrieved data foundation or business layer.

When you start the information design tool, it reopens any resources that were open when you last closed the tool. If secured resources are open, you need to enter your CMS authentication to start the tool.

**Note:**
To remove the local security requirement, you must have the "Save for all Users " right granted in the CMC. When a resource is saved for all users, any user can open the resource without entering CMS authentication.

Secured connections cannot be retrieved from the repository and stored locally in the information design tool. Instead, a shortcut to the connection in the repository is stored in the local project. Secured connections must be edited directly in the repository from the Repository Resources View. To be able to access data from a secured connection (for example, show table values or run a query), the user must enter the CMS authentication for the repository where the connection is published. The system uses the authentication to determine what rights the user has for that connection.

**Related Topics**
- About local projects
- About connection shortcuts

### 16.3 CMC rights for information design tool users

The application, universe, and connection rights necessary to do tasks in the information design tool are summarized in this topic.

Rights are granted in the Central Management Console (CMC). How to define these rights is described in the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

To open a session on the repository system and do all secured tasks in the information design tool:

- You must have a username and password configured by the system administrator in the CMC for the CMS where the repository is stored.
- You must have the right "Connect the CMS with Information design tool and view this object in the CMC" granted in the CMC.

Additional rights needed are listed in the table by task.
<table>
<thead>
<tr>
<th>Task</th>
<th>Rights required</th>
</tr>
</thead>
</table>
| Publish a connection to a repository      | • "Create, modify, or delete connections" application right  
• "View objects" right on the connections folder  
• "Add objects to the folder" right on the connections folder |
| Edit a secured connection from the Repository Resources View | • "Create, modify, or delete connections" application right  
• "Add objects to the folder" on the connection folder (to create)  
• "Edit objects" connection right  
• "Download connection locally" connection right (relational connections only) |
| Use a the local middleware driver for a secured connection | • "Download connection locally" connection right (relational connections only) |
| Publish a universe to a repository        | • "Publish universes" application right  
• "View objects" right on the universes folder  
• "Add objects to the folder" right on the universe folder  
• "Edit objects" universe right (to republish) |
| Retrieve a published universe from a repository | • "Retrieve universes" application right  
• "View objects" right on the universe folder  
• "View Objects" universe right  
• "Retrieve universe" universe right |
| Edit secured local resources               | • No rights are required, but the user must provide the CMS authentication of the user who saved the resources. |
| Unsecure local resources                  | • "Save for all users" application right  
• "Retrieve universes" application right  
• "View objects" right on the universes folder  
• "View Objects" universe right  
• "Retrieve universe" universe right  
• "Save for all users" universe right |
| Open the Security Editor                  | • "Administer security profiles" application right |
| Define security profiles                  | • "View objects" universe right  
• "Edit security profiles" universe right |
<table>
<thead>
<tr>
<th>Task</th>
<th>Rights required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign security profiles to users and groups</td>
<td>• &quot;View objects&quot; universe right&lt;br&gt;• &quot;Assign security profiles&quot; universe right</td>
</tr>
<tr>
<td>Run a query on a published universe</td>
<td>• &quot;View objects&quot; universe right&lt;br&gt;• &quot;Create and edit queries based on this universe&quot; universe right&lt;br&gt;• &quot;Data Access&quot; universe right&lt;br&gt;• &quot;View objects&quot; right on underlying connections&lt;br&gt;• &quot;Data Access&quot; right on underlying connections</td>
</tr>
<tr>
<td>Share project resources:</td>
<td>• &quot;Share projects&quot; application right</td>
</tr>
<tr>
<td>• Share a local project</td>
<td></td>
</tr>
<tr>
<td>• Open the Project Synchronization View</td>
<td></td>
</tr>
<tr>
<td>• Synchronize project resources</td>
<td></td>
</tr>
<tr>
<td>• Lock and Unlock resources</td>
<td></td>
</tr>
<tr>
<td>• Rename or delete a shared project</td>
<td></td>
</tr>
<tr>
<td>Convert a .unv universe stored in the repository</td>
<td>• &quot;View objects&quot; right on the universe folder&lt;br&gt;• &quot;Add objects to the folder&quot; right on the universe folder&lt;br&gt;• &quot;View objects&quot; universe right</td>
</tr>
<tr>
<td>Compute statistics for a multisource universe</td>
<td>• &quot;Compute statistics&quot; application right&lt;br&gt;• &quot;View objects&quot; universe right</td>
</tr>
<tr>
<td>Delete a universe from the repository</td>
<td>• &quot;View objects&quot; universe right&lt;br&gt;• &quot;Delete objects&quot; universe right</td>
</tr>
<tr>
<td>Delete a connection from the repository</td>
<td>• &quot;Create, modify, or delete connections&quot; application right&lt;br&gt;• &quot;View objects&quot; connection right&lt;br&gt;• &quot;Delete objects&quot; connection right</td>
</tr>
</tbody>
</table>

**Related Topics**
- [About session management](#)
Use the Security Editor to create and edit security profiles, and assign profiles to users and groups. This topic describes how to navigate the Security Editor. For steps to help you build universe security, see [How to secure a universe using security profiles](#).

The session name displays on the tab of the Security Editor. If the session name is prefixed by an asterisk, it means you have made changes to the security profiles or assignments in the Security Editor that have not yet been saved in the repository.

The Security Editor can be viewed in two ways: by universe, or by users/groups. Select the tab on the left-hand side of the Security Editor to display the view you want to work with.

- The **Universes / Profiles** tab lets you to do tasks by first selecting a universe in the repository.
- The **Users / Groups** tab lets you to do tasks by first selecting a user or group. The three icons in the **Users / Groups** panel lets you to display users and groups in three ways:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>Shows only users.</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>Shows all groups and users they contain. A group is shown even if it has no groups or users assigned. Groups are displayed as a flat list. This is the default display.</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>Shows all groups and the groups and users they contain. Groups are thus displayed with their different parent groups.</td>
</tr>
</tbody>
</table>

Application rights granted to you in the Central Management Console control what tasks you can perform in the Security Editor. For more information, see the Rights Appendix in the *SAP BusinessObjects Business Intelligence platform Administrator’s Guide*.

**Related Topics**
- [About universe security](#)
- [How to secure a universe using security profiles](#)
- [Changing Data Security Profile priority](#)
- [Changing security profile aggregation options](#)
- [Assigning Security Profiles to users](#)
- [Displaying profiles assigned to a user and previewing net profiles](#)

**16.5 How to secure a universe using security profiles**
Before you begin:

- The universe for which you want to create security must be published to a repository. You can create security for .unx universes only.
- Make sure you have the necessary rights defined in the Central Management Console (CMC). See the related topic about CMC rights.

For links to more detailed information on each step, see the Related Topics.

1. Open the Security Editor with a session in the repository where the universe is published.
2. Select the universe in the Universes / Profiles pane to define security profiles.
3. For relational universes, insert a Data Security Profile to define the following types of security:
   - Define replacement connections to override the connections defined in the universe.
   - Define overrides for the query options and query limits defined in the universe.
   - Restrict data returned to specific rows using a WHERE clause.
   - Define replacement tables.

   **Note:**
   You can create more than one Data Security Profile for a universe.

4. Insert a Business Security Profile to define the following types of security:
   - Restrict objects that appear in the Query Panel to create queries.
   - Restrict objects for which data is returned.
   - Filter data returned in queries.

   **Note:**
   You can create more than one Business Security Profile for a universe.

5. Save the changes to the security settings in the repository by clicking the save icon in the main tool bar.
6. Select the Users / Groups pane to assign the profiles to users and groups.
7. If more than one profile is assigned to a user (either directly or by inheritance), preview the net results of the aggregated profiles.
8. If modifications to the way profiles are aggregated is needed, in the Universes / Profiles pane, change Data Security Profile priority and profile aggregation options.
9. Save the changes to the security settings in the repository by clicking the save icon in the main tool bar.
10. Test the security profiles for a particular user:
    a. Open the Security Editor using the login information for the user who is being assigned the security profiles.
    b. In the Universes / Profiles pane, right-click the universe and select Run Query.

       The Query Panel opens. The security profiles assigned to the user are applied.

If you republish a universe, run a check integrity on the universe to flag any discrepancies between the universe and its security profiles. In the Universes / Profiles pane, right-click the universe and select Check Integrity.

**Related Topics**

- [CMC rights for information design tool users](#)
16.6 Opening the Security Editor

1. On the information design tool toolbar, click the **Security Editor** icon.
2. In the "Open Session" dialog box, select the session you want to open.
3. If you are not already logged into the selected session, enter the required information.

The Security Editor opens in a new tab.

**Note:**
You can open more than one session of the Security Editor at one time. The sessions must be on different repositories.

**Related Topics**
- Opening a session
- About the Security Editor

16.7 Inserting and editing a Data Security Profile

1. In the Security Editor **Universes / Profiles** pane, select the universe.
2. Do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing profile</td>
<td>Double-click the profile name.</td>
</tr>
<tr>
<td>To insert a profile</td>
<td>Right-click the universe name and select <strong>Insert Data Security Profile</strong>.</td>
</tr>
</tbody>
</table>

3. Define security settings in each of the tabs by clicking the tab you want.
For more information on the Data Security Profile settings, see the related topics.

**Note:**

Clicking the **Reset** button returns the settings on all tabs to the default values as they are defined in the data foundation and business layer.

4. When you have defined all of the settings, click **OK**.

5. To save the changes to the security settings in the repository, click the save icon in the main tool bar.

**Related Topics**

- [Opening the Security Editor](#)
- [Data Security Profile Connections settings](#)
- [Data Security Profile Controls settings](#)
- [Data Security Profile SQL settings](#)
- [Data Security Profile Rows settings](#)
- [Data Security Profile Tables setting](#)

### 16.7.1 Data Security Profile settings

A Data Security Profile is a group of settings that define security on a published universe using objects in the data foundation and the data connections.

All Data Security Profile settings apply to relational universes only.

**Table 16-3: Security settings in Data Security Profiles**

<table>
<thead>
<tr>
<th>Security Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connections</td>
<td>Defines replacement connections</td>
</tr>
<tr>
<td>Controls</td>
<td>Defines replacement query timeout and size limits</td>
</tr>
<tr>
<td>SQL</td>
<td>Defines replacement query options</td>
</tr>
<tr>
<td>Rows</td>
<td>Defines an SQL WHERE clause to restrict rows returned in the query</td>
</tr>
<tr>
<td>Tables</td>
<td>Defines replacement tables</td>
</tr>
</tbody>
</table>

Each type of Data Security Profile setting is described in a related topic.

**Related Topics**

- [Data Security Profile Connections settings](#)
- [Data Security Profile Controls settings](#)
16.7.2 Data Security Profile Connections settings

Connections settings can be defined for relational universes only.

Use the Data Security Profile Connections setting to define replacement connections that can override the connections defined in the universe. Once a user is assigned or inherits a profile that contains a replacement connection, when the user runs a query on the universe, the replacement connection is used instead of the connection defined in the universe.

Only secured connections can be defined as replacement connections. Relational connections fall into one of three types, listed below. The replacement connection must be of the same type as the original connection.

- SAP NetWeaver BW relational databases
- SAS relational databases
- Other relational databases

You can select a connection in the Connections folder and sub-folders for which you have the "View objects" right granted for the repository where you are defining the security profiles.

For multisource universes that rely on multiple connections, you can define a replacement for each connection.

Related Topics
- Security profile aggregation

16.7.3 Data Security Profile Controls settings

Controls settings can be defined for relational universes only.

Use the Data Security Profile Controls settings to define replacement query limits to override default limits when retrieving data from the database. Default query limits are set by the universe designer in the business layer. Once a user is assigned or inherits a profile with replacement Controls settings, when the user runs a query, the replacement limits are used instead of the limits defined in the business layer properties.
In the editor for Data Security Profiles, the limits selected and the limit values defined in the business layer are displayed. When you select or unselect a limit, or enter a new value for a limit, the label appearance changes to bold. This shows that the limit is an override and not the default limit defined in the universe.

<table>
<thead>
<tr>
<th>Query Limit</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limit size of result set to</td>
<td>• True and a numerical size between 0 and 2147483647 rows</td>
</tr>
<tr>
<td></td>
<td>• False</td>
</tr>
<tr>
<td>Limit execution time to</td>
<td>• True and a numerical size between 0 and 2147483647 minutes</td>
</tr>
<tr>
<td></td>
<td>• False</td>
</tr>
<tr>
<td>Warn if cost estimate exceeds</td>
<td>• True and a numerical size between 0 and 10000 minutes</td>
</tr>
<tr>
<td></td>
<td>• False</td>
</tr>
</tbody>
</table>

For more information on query limits, see the related topic about business layer properties.

**Related Topics**
- [About business layer properties](#)
- [Security profile aggregation](#)

### 16.7.4 Data Security Profile SQL settings

SQL settings can be defined for relational universes only.

Use the Data Security Profile SQL settings to define replacement query options. The universe designer defines default query options in the business layer and data foundation properties. Once a user is assigned or inherits a profile with SQL settings, when the user uses the query panel, the replacement options are used instead of the query options defined in the universe.

In the editor for Data Security Profiles, the SQL settings selected in the business layer and data foundation are displayed. When you select or unselect an option, the label appearance changes to bold. This shows that the option is an override and not the default defined for the universe.
### Query Option

<table>
<thead>
<tr>
<th>Query Option</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow use of subqueries</td>
<td>• True</td>
</tr>
<tr>
<td></td>
<td>• False</td>
</tr>
<tr>
<td>Allow use of union, intersect and minus operators</td>
<td>• True</td>
</tr>
<tr>
<td></td>
<td>• False</td>
</tr>
<tr>
<td>Allow complex operands in Query Panel</td>
<td>• True</td>
</tr>
<tr>
<td></td>
<td>• False</td>
</tr>
<tr>
<td>Multiple SQL statements for each context</td>
<td>• True</td>
</tr>
<tr>
<td></td>
<td>• False</td>
</tr>
<tr>
<td>Multiple SQL statements for each measure</td>
<td>• True</td>
</tr>
<tr>
<td></td>
<td>• False</td>
</tr>
<tr>
<td>Allow Cartesian products</td>
<td>• True</td>
</tr>
<tr>
<td></td>
<td>• False</td>
</tr>
</tbody>
</table>

For more information on query options, see the related topics about business layer and data foundation properties.

**Related Topics**

- About business layer properties
- About data foundation properties
- Security profile aggregation

### 16.7.5 Data Security Profile Rows settings

Rows settings can be defined for relational universes only.

Use Data Security Profile Rows settings to restrict the rows returned in a query. You restrict the rows by defining an SQL WHERE clause for a specified table. Once a user is assigned or inherits a profile with a Rows setting, when the user runs a query on the universe, the defined WHERE clause is added to the SQL generated if the table is referenced in the query.

**Note:**
A user who has the right to edit the generated SQL in the reporting tool can change the WHERE clause generated by the Rows setting. Remember to manage the user's rights in the reporting tool to prevent the user from modifying the SQL.
You can define the WHERE clause for any standard table in the data foundation. The SQL for the WHERE clause can include:

- @Functions such as @Variable and @Prompt
- For multisource enabled universes, references to other tables in any connection defined for the universe
- For multisource-enabled universes, SAP BusinessObjects SQL functions

The SQL for the WHERE clause cannot include:

- Calculated columns
- Derived tables

**Related Topics**

- Security profile aggregation

### 16.7.6 Data Security Profile Tables setting

Tables settings can be defined for relational universes only.

Use the Data Security Profile Tables setting to define replacement tables. Once a user is assigned or inherits a profile with a Tables setting, when the user runs a query that references the original table, the replacement table is used instead.

**Note:**

A user who has the right to edit the generated SQL in the reporting tool can change the replacement table name. Remember to manage the user's rights in the reporting tool to prevent the user from modifying the SQL.

You can replace a standard table in the data foundation with a database table in any connection defined for the universe, or another standard table in the data foundation.

**Note:**

If you want to specify an owner and qualifier for the replacement table, you must enter these in the fields provided, and not as part of the table name. For more information about data foundation table names, see the related topic.

**Related Topics**

- About tables in the data foundation
- Security profile aggregation

### 16.8 Changing Data Security Profile priority
1. In the Security Editor **Universes / Profiles** pane, select the universe.
2. Right-click the universe name and select **Change Data Security Profile Priority**.

   **Note:**
   The command is only available if the universe has more than one Data Security Profile defined.

3. In the dialog box listing the Data Security Profiles, use the arrow buttons to move profiles up and down in the list. The first profile in the list has highest priority.
4. When you have finished prioritizing, click **OK**.
5. To save the changes in the repository, click the save icon in the main tool bar.

**Related Topics**
- Security profile aggregation
- Opening the Security Editor

### 16.9 Inserting and editing a Business Security Profile

1. In the Security Editor **Universes / Profiles** pane, select the universe.
2. Do one of the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>To edit an existing profile</td>
<td>Double-click the profile name.</td>
</tr>
<tr>
<td>To insert a profile</td>
<td>Right-click the universe name and select <strong>Insert Business Security Profile</strong>.</td>
</tr>
</tbody>
</table>

3. Define security settings in each of the tabs by clicking the tab you want.

   For more information on the Business Security Profile settings, see the related topics.

   **Note:**
   Clicking the **Reset** button returns the settings on all tabs to the default values as they are defined in the data foundation and business layer.

4. When you have defined all of the settings, click **OK**.
5. To save the changes to the security settings in the repository, click the save icon in the main tool bar.

**Related Topics**
- Business Security Profile Create Query settings
- Business Security Profile Display Data settings
- Business Security Profile Filters settings
16.9.1 Business Security Profile settings

A Business Security Profile is a group of settings that define security on a published universe using objects in the business layer.

Table 16-4: Security settings for Business Security Profiles

<table>
<thead>
<tr>
<th>Security Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Query</td>
<td>Defines the universe views and business layer objects available to the user in the query panel. <strong>Note:</strong> Create Query settings secure metadata only.</td>
</tr>
<tr>
<td>Display Data</td>
<td>Grants or denies access to the data retrieved by objects in the business layer when the user runs a query.</td>
</tr>
<tr>
<td>Filters</td>
<td>Defines filters using objects in the business layer.</td>
</tr>
</tbody>
</table>

Each type of Business Security Profile setting is described in a related topic.

In the business layer, designers can set the status of objects to **Active**, **Hidden**, or **Deprecated**. When defining profile settings, you have access to all active objects in the business layer. Objects that are hidden or deprecated in the business layer never appear in the query panel or on reports.

**Related Topics**
- Business Security Profile Create Query settings
- Business Security Profile Display Data settings
- Business Security Profile Filters settings
- Security profile aggregation
- Inserting and editing a Business Security Profile

16.9.2 Business Security Profile Create Query settings

Use the Business Security Profile Create Query setting to grant or deny the use of business layer objects in the query panel.
By default, a user with access to the universe granted in the repository can see all universe objects in
the query panel. Once the user is assigned or inherits a profile with a Create Query setting, only the
views and objects granted by the setting are displayed and can be selected for a query.

If an object is not granted and not denied explicitly, it is denied by default. Unlike objects that are explicitly
denied, objects that are denied by default could be granted by inheritance after aggregating Business
Security Profiles to determine the net profile for a user. For more information about aggregating profiles,
see the related topic.

There are two ways to grant or deny objects:

• By business layer view: Grants or denies all objects in a view. The **All business layer views** option
  allows you to grant or deny all views defined for the universe.

• By object: You can grant or deny the objects listed below. The **All objects** option allows you to grant
  or deny all objects in the business layer.
  • Dimensions
  • Attributes
  • Measures
  • Calculated members
  • Filters
  • Prompts
  • Named sets
  • Folders: Grants ordenies all objects in the folder.
  • Analysis dimensions: Grants or denies all objects in the dimension.
  • Hierarchies: Grants or denies all objects in the hierarchy.

  **Note:**
  It is not possible to grant or deny a hierarchy level.

**Tip:**
If most views are allowed, it is easiest to grant all views, and then deny the ones that are not allowed.
Using the **All business layer views** and **All objects** options has the advantage that any new view or
object defined in the business layer is automatically included in the Create Query setting when the
universe is published.

If the **All business layer views** or **All objects** option is used, the settings are aggregated to determine
the net setting for this profile, for example:

• If **All business layer views** are denied and one view is granted, this profile denies all views except
  the one granted.

• If **All business layer views** are granted and one view is denied, this profile grants all views except
  the one denied.

• If **All objects** are denied and one object is granted, any parent folders in the path to access the
  object are granted, but only to access the object. The other objects in the parent folders are denied.

• If **All objects** are granted and one object is denied, the parent folders in the path to access the
  object are denied, but only to prevent access to this object. The other objects in the parent folders
  are granted.

The objects in a granted view are granted in that view only. If the same object is contained in another
view, it is not automatically granted.
Whether or not a user sees a particular object in the query panel is determined after aggregating the Create Query settings in all profiles assigned to the user, and taking into consideration object access level. For more information about aggregating profiles, see the related topic.

**Related Topics**
- Security profile aggregation

### 16.9.3 Business Security Profile Display Data settings

Use the Business Security Profile Display Data settings to grant or deny access to the data retrieved by objects in the business layer.

By default, a user with access to the universe granted in the repository can see the data retrieved by all universe objects. Once the user is assigned or inherits a profile with a Display Data setting, only the data corresponding to the objects granted by the setting is displayed.

If an object is not granted and not denied explicitly, it is denied by default. Unlike objects that are explicitly denied, objects that are denied by default could be granted by inheritance after aggregating Business Security Profiles to determine the net profile for a user. For more information about aggregating profiles, see the related topic.

The following objects can be granted or denied. The **All objects** option allows you to grant or deny all objects in the business layer.

- Dimensions
- Attributes
- Measures
- Calculated Members
- Named Sets
- Folders: Grants or denies all the objects in the folder.
- Hierarchies

Using the **All objects** option has the advantage that any new object defined in the business layer is automatically included in the display data setting when the universe is published.

If the **All objects** option is used, the settings are aggregated to determine the net setting for this profile, for example:

- If **All objects** are denied and one object is granted, any parent folders in the path to access the object are granted, but only to access the object. The other objects in the parent folders are denied.
- If **All objects** are granted and one object is denied, the parent folders in the path to access the object are denied, but only to prevent access to this object. The other objects in the parent folders are granted.

A user who is denied an object by a Display Data setting might refresh a report containing the denied object. You can specify what the refresh should do in this case by setting the SQL generation parameter AUTO_UPDATE_QUERY in the business layer.
• If this parameter is set to No, then refreshing the report generates an error message.
• If this parameter is set to Yes, then the denied objects are removed from the query and from any filters defined in the business layer. Data for other granted objects is retrieved and displayed to the user in a partial report.

Whether or not a user sees data for a particular object is determined after aggregating the Display Data settings in all profiles assigned to the user, and taking into consideration object access level. For more information about aggregating profiles, see the related topic.

Related Topics
• Security profile aggregation

16.9.4 Business Security Profile Filters settings

Use the Business Security Profile Filters setting to define a filter using objects in the business layer, or named member sets. You create and edit filters explicitly for the Business Security Profile using the Security Editor. Filters in the Business Security Profile are not accessible in the business layer. If the Business Security Profile is deleted, the filter or named set is also deleted.

Once the user is assigned or inherits a profile with a Filters setting, the filter is added to the query script (and therefore combined with any filters defined in the business layer) to restrict the data displayed.

Relational universes
For relational universes, you define filters on dimensions and measures in the business layer. You can define compound filters that are linked by the AND and OR operators. You can also define multiple filters to apply to the query.

When a user runs a query, the filters are always applied to the query and to the returned data. This is different from the Data Security Profile Rows setting which only applies if a defined table is referenced in the query.

OLAP universes
For OLAP universes, you define a named set of members. You can include or exclude members for any dimension in the business layer. The excluded members are removed from the query when data is retrieved from the cube.

Note:
The filter does not impact the aggregation of values in the report. Only the display of members is filtered.

You can include or exclude members from multiple dimensions. You can also define multiple named sets to apply to the query.

Related Topics
• How to build a business filter
More than one Data Security Profile or Business Security Profile defined for a universe may be assigned to the same user. Multiple profiles can be directly assigned to a user or group, and be inherited from parent groups. When this happens, the security settings in the different profiles are aggregated to result in one effective Data Security Profile, and one effective Business Security Profile, called net profiles. The settings in the net profiles are applied when the user creates a query or views a report.

Two methods are used for aggregating security settings: priority and restriction level.

Priority is used to aggregate Data Security Profile settings. You can prioritize the Data Security Profiles in the Security Editor.

Some Data Security Profile settings and all Business Security Profile settings are aggregated based on restriction level: very restrictive, moderately restrictive, or less restrictive.

The restriction level defines which operators to use to aggregate profiles. Different aggregation operators are used depending on if the profile is inherited or merged:

- If the user or group is assigned Profile A and belongs to a group that is assigned Profile B, Profile A and Profile B are inherited.
- If the user or group belongs to a group assigned Profile A and another group assigned Profile B, Profile A and Profile B are merged.
- If the user or group is assigned both Profile A and Profile B, Profile A and Profile B are merged.

You can change these restriction levels in the Security Editor to affect how the profiles are aggregated.

- The less restrictive level is appropriate when security is designed with roles, each role granting new rights to the user.
- The most restrictive level is appropriate when each profile is used to restrict what the user can see.
- The moderately restrictive level uses the most restrictive level for inherited profiles and the less restrictive level for merged profiles.

The operations used to aggregate profile settings (for example AND, OR) vary for the different settings. For detailed information about aggregation for each type of setting, see the related topic.

The Data Security Profile Rows setting and Business Security Profile Filters setting both generate a WHERE clause to filter the query. The Rows setting is applied first. The WHERE clause in the Filters setting is then applied to the results of the first query. In effect, the two WHERE clauses are aggregated with the AND operator.

Related Topics

- Aggregation of Connections settings
- Aggregation of Controls settings
16.10.1 Aggregation of Connections settings

If more than one Data Security Profile for a universe is assigned to or inherited by the same user, the connection defined in the Data Security Profile with the highest priority is used.

If the universe has multiple connections, aggregation of the Connections setting is done for each connection independently.

Related Topics
- Changing Data Security Profile priority
- Data Security Profile Connections settings

16.10.2 Aggregation of Controls settings

If more than one Data Security Profile for a universe is assigned to or inherited by the same user, the following rules are used to aggregate the Controls settings. The rules are applied to each query limit to determine the value to be used when the user runs a query or report.
<table>
<thead>
<tr>
<th>Restriction level</th>
<th>Aggregation Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Restrictive</td>
<td>The limit is active only if it is selected in all merged and inherited profiles. The value used is the minimum value for the limit among all the merged and inherited profiles.</td>
</tr>
<tr>
<td>Moderately Restrictive</td>
<td>The limit is active only if it is selected in all inherited profiles and selected in at least one merged profile. First, the minimum value is determined for the limit comparing the inherited profiles. This value is compared to the values among the merged profiles. The value used is the maximum among these values.</td>
</tr>
<tr>
<td>Less Restrictive</td>
<td>The limit is active if it is selected in any merged or inherited profile. The value used is the maximum value for the limit among all the merged and inherited profiles.</td>
</tr>
<tr>
<td>Priority (default)</td>
<td>The activation and value of the limit in the Data Security Profile with the highest priority is used.</td>
</tr>
</tbody>
</table>

**Note:**
For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

**Related Topics**
- Security profile aggregation
- Changing security profile aggregation options
- Changing Data Security Profile priority
- Data Security Profile Controls settings

### 16.10.3 Aggregation of SQL settings

If more than one Data Security Profile for a universe is assigned to or inherited by the same user, the following rules are used to aggregate the SQL settings. The rules are applied to each query option to determine the value to be used when the user creates a query.
### Restriction level

<table>
<thead>
<tr>
<th>Restriction level</th>
<th>Aggregation Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Restrictive</td>
<td>The option is active only if it is selected in all merged and inherited profiles.</td>
</tr>
<tr>
<td>Moderately Restrictive</td>
<td>The option is active if it is selected in all inherited profiles and selected in at least one assigned profile.</td>
</tr>
<tr>
<td>Less Restrictive</td>
<td>The option is active if it is selected in any merged or inherited profile.</td>
</tr>
<tr>
<td>Priority (default)</td>
<td>The activation and value of the option in the Data Security Profile with the highest priority is used.</td>
</tr>
</tbody>
</table>

**Note:**
For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

**Related Topics**
- Security profile aggregation
- Changing security profile aggregation options
- Changing Data Security Profile priority
- Data Security Profile SQL settings

### 16.10.4 Aggregation of Rows settings

If more than one Data Security Profile for a universe is assigned to or inherited by the same user, the following rules are used to aggregate the Rows settings and determine the WHERE clause to be used when the user runs a query or report.

First, the WHERE clauses for each table are aggregated according to the restriction level:
### 16.10.5 Aggregation of Tables settings

If more than one Data Security Profile for a universe is assigned to or inherited by the same user, the replacement table defined in the Data Security Profile with the highest priority is used. If settings are defined for multiple tables, the aggregation is done for each table independently.

**Related Topics**
- [Changing Data Security Profile priority](#)
- [Data Security Profile Tables setting](#)

### 16.10.6 Aggregation of Create Query settings
If more than one Business Security Profile for a universe is assigned to or inherited by the same user, the Create Query settings are aggregated. Object access levels, if they are defined, are applied to determine whether or not a user sees a particular object in the query panel.

First, the list of views that the user can select in the query panel is determined by aggregating the profiles according to the restriction level:

<table>
<thead>
<tr>
<th>Restriction level</th>
<th>Aggregation Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Restrictive (default)</td>
<td>The user can select the view in the query panel only if it is granted in all inher-</td>
</tr>
<tr>
<td></td>
<td>ited and merged profiles.</td>
</tr>
<tr>
<td>Moderately Restrictive</td>
<td>The user can select the view in the query panel only if it is granted in all inher-</td>
</tr>
<tr>
<td></td>
<td>ited profiles and granted in at least one merged profile.</td>
</tr>
<tr>
<td>Less Restrictive</td>
<td>The user can select the view in the query panel if it is granted in any inherited</td>
</tr>
<tr>
<td></td>
<td>or merged profile.</td>
</tr>
</tbody>
</table>

Once a view is selected in the query panel, an object appears if it is included in the view, and if it is not explicitly denied after aggregating the profiles according to restriction level:

<table>
<thead>
<tr>
<th>Restriction level</th>
<th>Aggregation Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Restrictive (default)</td>
<td>The object is denied if it is explicitly denied in any inherited or merged profile.</td>
</tr>
<tr>
<td>Moderately Restrictive</td>
<td>The object is denied if it is explicitly denied in any inherited profile and denied in all merged profiles.</td>
</tr>
<tr>
<td>Less Restrictive</td>
<td>The object is denied only if it is explicitly denied in all inherited and merged profiles.</td>
</tr>
</tbody>
</table>

After aggregation, the denied objects are not displayed even if they belong to a granted view. If a folder is denied, then all sub-folders and objects in the folder are denied.

Finally, the access level granted to the user in the Central Management Console determines which of the objects granted by the net Business Security Profile are available in the query panel. The user sees only the objects with an access level lower than or equal to his authorized access level. You assign access levels to objects in the business layer editor.

**Note:**
For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

For more information about object access levels, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide.*
16.10.7 Aggregation of Display Data settings

If more than one Business Security Profile for a universe is assigned to or inherited by the same user, the Display Data settings are aggregated. Object access levels, if they are defined, are applied to determine whether or not a user sees the data for an object in the business layer.

First, the list of objects that the user can see data for is determined by aggregating the profiles according to restriction level.

<table>
<thead>
<tr>
<th>Restriction level</th>
<th>Aggregation Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Restrictive (default)</td>
<td>The data appears only if it is granted in all inherited and merged profiles.</td>
</tr>
<tr>
<td>Moderately Restrictive</td>
<td>The data appears only if the object is granted in all inherited profiles and granted in at least one merged profile.</td>
</tr>
<tr>
<td>Less Restrictive</td>
<td>The data appears if the object is granted in any inherited or merged profile.</td>
</tr>
</tbody>
</table>

If a folder is denied, then the data for all objects in the folder and its subfolders is denied.

Finally, the access level granted to the user in the Central Management Console determines which of the objects granted by the net Business Security Profile the user sees data for. The user sees data only for the objects with an access level lower than or equal to his authorized access level. You assign access levels to objects in the business layer editor.

**Note:**
For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

For more information about object access levels, see the *SAP BusinessObjects Business Intelligence platform Administrator's Guide*.

Related Topics

- Security profile aggregation
- Changing security profile aggregation options
- Business Security Profile Display Data settings
### 16.10.8 Aggregation of Filters settings

If more than one Business Security Profile for a universe is assigned to or inherited by the same user, the following rules are used to aggregate the filters settings and determine the filter to be added to the query script when the user runs a query or report.

For relational universes, the filters are aggregated according to the restriction level. The resulting filter is added to the WHERE clause applied to the query.

<table>
<thead>
<tr>
<th>Restriction level</th>
<th>Aggregation Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Restrictive (default)</td>
<td>The filters in all profiles are combined using the AND operator.</td>
</tr>
<tr>
<td>Moderately Restrictive</td>
<td>Inherited filters are aggregated using the AND operator. Merged filters are aggregated using the OR operator.</td>
</tr>
<tr>
<td>Less Restrictive</td>
<td>The filters in all profiles are combined using the OR operator.</td>
</tr>
</tbody>
</table>

For OLAP universes, the named sets are aggregated according to the restriction level.

<table>
<thead>
<tr>
<th>Restriction level</th>
<th>Aggregation Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Restrictive (default)</td>
<td>The user sees a member only if it is included in every named set defined in all profiles.</td>
</tr>
<tr>
<td>Moderately Restrictive</td>
<td>The user sees a member if it is included in every named set defined in the inherited profiles, and included in at least one named set defined in the merged profiles.</td>
</tr>
<tr>
<td>Less Restrictive</td>
<td>The user sees a member if it is included in any named set defined in any profile.</td>
</tr>
</tbody>
</table>

**Note:**
For a definition of inherited and merged profiles, see the related topic on security profile aggregation.

**Related Topics**
- [Security profile aggregation](#)
- [Changing security profile aggregation options](#)
- [Business Security Profile Filters settings](#)
16.11 Changing security profile aggregation options

1. In the Security Editor Universes / Profiles pane, select the universe.
   The current aggregation options for the universe display on the lower, right-hand side of the editor.
2. For each security setting, select a new aggregation option from the list.
   The options apply only to the universe currently selected.
3. To save the changes in the repository, click the save icon in the main tool bar.

Related Topics
• Security profile aggregation
• Opening the Security Editor

16.12 Assigning Security Profiles to users

1. In the Security Editor Universes / Profiles pane, select the universe.
   Any currently assigned users or groups appear in the Assigned Users list.
2. To assign, select the user or group in the list of users on the right-hand side of the editor, and click the arrow that points to the Assigned Users list.
3. To unassign, select the user or group in the Assigned Users list and click the arrow that points to the list of all users.
   
   Caution:
   The double-arrow icon unassigns all users and groups whether they are selected or not.
4. To save the changes in the repository, click the save icon in the main tool bar.

Related Topics
• Opening the Security Editor

16.13 Displaying profiles assigned to a user and previewing net profiles

1. In the Security Editor, click the Users / Groups pane on the left-hand side of the editor.
2. In the Users / Groups pane, select the user or group.
3. In the **Universes / Profiles** pane in the upper right-hand side of the editor, select the universe.

   **Tip:**
   You can change the display to list only universes that have profiles assigned to the selected user or group by selecting the **Display only universes assigned to the selected user/group** option.

   Once you have selected a user and a universe, the assigned profiles appear in the profiles list on the lower right-hand side of the editor.

4. To preview the net Data Security Profile or net Business Security Profile, click **Preview Net Profile** below the corresponding profile list.

   The Data Security Profile or Business Security Profile editor opens in read-only mode. The settings on each tab represent the settings that will be used after aggregation of all the profiles assigned to the user is taken into account.

**Related Topics**
- Security profile aggregation
- Opening the Security Editor
When you run a query on a universe published in a repository, the Query Panel applies the settings defined in the security profiles for the universe according to the user name defined in the session.

1. In the Repository Resources View, select the universe. Select .unx universes only.
2. Right-click the universe name and select Run Query.

The Query Panel opens with a list of views and objects granted for your user name.

Related Topics
• How to build a query
Running a query on a universe published in a repository
18.1 About the SQL/MDX Expression Editor

The SQL/MDX Expression Editor helps you write valid SQL/MDX expressions.

You can type SQL/MDX directly into the "Expression" box, or drag and drop table names, column names, business objects, functions, and parameters from the resource panes available in the editor. These panes are described in the following table. To display a resource pane, click the icon in the tool bar of the "Expression" pane. Different icons are available depending on the type of expression you are editing.
<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Tables" /></td>
<td><strong>Tables</strong>&lt;br&gt;The list of tables and columns in the data foundation. To see a list of values for a column, click the <img src="image" alt="icon" /> icon next to column name.</td>
</tr>
<tr>
<td><img src="image" alt="Database Tables" /></td>
<td><strong>Database Tables</strong>&lt;br&gt;For relational connections, the list of database tables in the connections. Used for defining expressions for derived tables and lists of values. To see a list of values for a column, click the <img src="image" alt="icon" /> icon next to column name.</td>
</tr>
<tr>
<td><img src="image" alt="OLAP Metadata" /></td>
<td><strong>OLAP Metadata</strong>&lt;br&gt;For OLAP connections, the list of objects in the source cube.&lt;br&gt;To change the display options, click <img src="image" alt="icon" />. You can display names, keys, or both.&lt;br&gt;To see a list of members for a level, click the <img src="image" alt="icon" /> icon next to the level name.&lt;br&gt;Use the <img src="image" alt="icon" /> icon to search for a string in the object names.&lt;br&gt;&lt;strong&gt;Note:&lt;/strong&gt;&lt;br&gt; A list of values for hierarchy attribute objects is not available in the OLAP Metadata pane. To see values for attributes, use the list in the Business Layer pane.</td>
</tr>
<tr>
<td><img src="image" alt="Functions" /></td>
<td><strong>Functions</strong>&lt;br&gt;The list of functions that can be used in the expression. The functions are grouped by type:&lt;br&gt;• <strong>Operators</strong>: Common database operators, for example *, SUM, IS NOT NULL.&lt;br&gt;• <strong>Database Functions</strong>: The SQL functions that are valid for the databases in the connections.&lt;br&gt;• <strong>System Variables</strong>: The system variables for which you can retrieve the values assigned using the @Variable function. For more information, see the related topic about @Variable.&lt;br&gt;&lt;strong&gt;Note:&lt;/strong&gt;&lt;br&gt;You can also reference User Attributes defined in the Central Management Server using @Variable.&lt;br&gt;• <strong>@Functions</strong>: The @Functions that are valid for this expression. For more information, see the related topic about @Functions.</td>
</tr>
<tr>
<td><img src="image" alt="Business Layer" /></td>
<td><strong>Business Layer</strong></td>
</tr>
</tbody>
</table>
The list of objects in the business layer. To see a list of members for a level, click the icon next to the level name. Use the icon to search for a string in the object names.

Use the toggle button to change how the object-related text is inserted into the expression:
- When the button is unselected (default): The @Select function for the object is inserted, for example: @Select(Account.Account Number)
- When the button is selected: The SQL or MDX expression for the object is inserted, for example: [Account].[Account Number]

The list of parameters defined in the data foundation and business layer.

Click the Validate icon in the tool bar of the "Expression" pane to check if the expression you have defined is valid SQL/MDX.

Related Topics
- About @Variable
- About @Functions
- About tables in the data foundation

18.2 About @Functions

@Functions are special functions that provide more flexible methods for specifying the query script for an object. Select the related topic to see more about an @Function.

Related Topics
- About @Prompt
- About @Variable
- About @DerivedTable
- About @Aggregate_Aware
- About @Select
- About @Where
18.2.1 About @Prompt

Use the @Prompt function to insert a prompt into a query. Prompts can be used to restrict the data when a user creates a report. You use the @Prompt function in the SQL SELECT statement or WHERE clause, or in the MDX expression for an object. It forces a user to enter one or more values (or select from a list of values) for a restriction when that object is used in a query. When the user runs the query, a prompt box appears asking for a value to be entered or selected.

@Prompts are useful when you want to force a restriction in the query script, but do not want to preset the value of the condition.

The @Prompt function is allowed in the following expressions:

- Joins
- Calculated columns (except in database-specific SQL in multisource-enabled data foundations)
- Derived tables
- Business objects in the business layer

You can insert a @Prompt definition in the following ways:

- Define a named parameter for the prompt and reference the parameter in the @Prompt function, for example:

  ```sql
  @Prompt(<parameter_name>)
  ```

  For more information, see the related topic about parameters.

- Type the definition in the SQL or MDX expression of the object. For more information on the syntax and parameters of the @Prompt function, see the related topic.

Related Topics
- About parameters
- @Prompt syntax

18.2.1.1 @Prompt syntax

The syntax for the @prompt function is as follows:

```sql
@Prompt('message','type',['lov'],Mono|Multi[:Any|Leaf],free|constrained|primary_key|constrained_key,persistent|not_persistent,['default values'])
```
The function parameters and the possible values are described in the following table. The parameters are separated by commas. You must specify at least the first two parameters. If you want to specify additional parameters, you must include the intervening commas for the optional parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| 'message' | Text of the prompt message. This parameter is mandatory.  
The text appears in the prompt box when the user runs the query.  
The text must be enclosed in single quotes, for example, 'Choose a Region'.  
For proper function of the prompt, the prompt text should be unique within the universe. |
| 'type'    | The data type of the prompt. This parameter is mandatory.  
The user's response is interpreted with the data type you specify. The list of values (lov parameter), and default values also have this data type. It can be one of the following:  
  - 'A' for alphanumeric string  
  - 'K' for keyword. This type is also an alphanumeric string, however the responses to the prompt will be not be surrounded by quotes in the query script at runtime.  
  - 'N' for number  
  - 'D' for date  
  - 'DT' for datetime  
The specified type must be enclosed in single quotes.  
The type parameter can be a pair of data types to indicate a name and key. The syntax is 'value_type:key_type', for example: 'A:N' where the first type is the data type of the name that the user sees in the list of values, and the second type is the data type of the primary key that is used by the query. Both the value_type and the key_type can be any of the available data types.  

**Note:**  
To use this option, you must make sure the object and the prompt are index aware:  
  - Define a primary key for the object in the business layer.  
  - Specify primary key for the fifth parameter in the @prompt function.  
In this case, if the lov or default_values parameters are used, they must contain a list of pairs of values.
The list of values the user can choose from when prompted. This parameter is optional.

You can specify a list of values in three ways:

- The default list of values associated with an object in the business layer (a dimension, measure, attribute, hierarchy, or hierarchy level). Enter the full path and object name in the business layer between single quotes, for example: 'Myconnection\dimproduct\productname'. In this example, 'productname' is the business layer object name.

  The object must be index aware: a primary key is defined for the object in the business layer.

- A named list of values defined in the business layer or data foundation. Enter the name of the list of values between single quotes: 'G7_Countries'.

  If the list of values is hierarchical with named levels, you can specify the level to use for the prompt: 'Country_Region_City_LOV':'Region'. In this example, 'Country_Region_City_LOV' is the name of the list of values and 'Region' is the target level.

  If the list of values is a multiple column list of values with named columns, you can specify the column to use for the prompt: 'Country_Region_City_LOV':'Region'. In this example, 'Country_Region_City_LOV' is the name of the list of values and 'Region' is the target column.

- A hard-coded list of values or value pairs (‘name’:’key’). The values in a pair are separated by a colon. Each value is enclosed in single quotes. The pairs of values are separated by a comma. The whole list is enclosed in curly brackets:

  The syntax for a single value: {‘value’}
  The syntax for several single values: {‘value1’,’value2’,... ,’valuen’}
  The syntax for a pair of values: {‘value’:’key’}
  The syntax for pairs of values: {‘value1’:’key1’,’value2’:’key2’,...,’valuen’:’keyn’}. For example: {‘France’:’FR’, ‘Germany’:’DE’,’Spain’:’ES’, ‘United Kingdom’:’UK’}

**Note:**
If the list of values is index aware (a primary key has been defined for the object in the business layer, or you use {name, key} pairs for the list of values, specify primary key for the fifth parameter in the @prompt function.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mono</td>
<td>The selection mode. If not specified, Mono is default.</td>
</tr>
<tr>
<td>• Use Mono if the user can select only one value from the list of values.</td>
<td></td>
</tr>
<tr>
<td>• Use Multi if the user can select multiple values from the list of values.</td>
<td></td>
</tr>
<tr>
<td>Multi</td>
<td>You can optionally specify the hierarchical selection mode for hierarchical lists of values. If not specified, Leaf is default:</td>
</tr>
<tr>
<td>[:Any</td>
<td>Leaf]</td>
</tr>
<tr>
<td>• Use Leaf if the user can only select only the leaf members/values from the hierarchical list of values.</td>
<td></td>
</tr>
<tr>
<td>free</td>
<td>The entry constraint type. If not specified, free is default.</td>
</tr>
<tr>
<td>• Use free if the user can enter a value, or select one from the list of values.</td>
<td></td>
</tr>
<tr>
<td>• Use constrained if the user must select a value from the list of values.</td>
<td></td>
</tr>
<tr>
<td>constrained</td>
<td>• Use primary_key when you are using an index aware object or {name, key} pairs. The associated key value for the object is used in the query rather than the entered or displayed name value.</td>
</tr>
<tr>
<td>primary_key</td>
<td>persistent</td>
</tr>
<tr>
<td>• Use persistent if, when refreshing a document, the last values used in the prompt are displayed by default, even when default values are defined.</td>
<td></td>
</tr>
<tr>
<td>• Use not_persistent if, when refreshing a document, no values used are displayed in the prompt by default.</td>
<td></td>
</tr>
<tr>
<td>not_persistent</td>
<td>default values</td>
</tr>
<tr>
<td>Enter default values in the following ways:</td>
<td></td>
</tr>
<tr>
<td>• For a single value: {'France'}</td>
<td></td>
</tr>
<tr>
<td>• For a pair of values: {'France':'FR'}</td>
<td></td>
</tr>
<tr>
<td>• For two pairs of values: {'France':'FR','Germany':DE'}</td>
<td></td>
</tr>
<tr>
<td>• For hierarchical values, use \ to separate the hierarchy level values: {'Europe':2\France\Marseille\CSP Systems','Europe':2\Germany\Berlin'}</td>
<td></td>
</tr>
<tr>
<td>When refreshing a a document, these values are displayed by default, but if the persistent option is set, then the last values used in the prompt are used instead of the default values.</td>
<td></td>
</tr>
<tr>
<td>If you specify the primary_key parameter in the prompt definition, you must provide the key values.</td>
<td></td>
</tr>
</tbody>
</table>
18.2.2 About @Variable

Use the @Variable function in an SQL or MDX expression (usually in the WHERE clause) to retrieve the value assigned to a system variable or user attribute. The syntax of the @Variable function is @Variable('<Variable_name>'). The variable_name must be in single quotes. Possible variables are described in the following table:

<table>
<thead>
<tr>
<th>Variable Name and Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables containing information about the user's authorization:</td>
<td>For example, to restrict data retrieved in a query to the current user, use the BOUSER variable in the WHERE clause: WHERE Employees.Employee_Name = @Variable('BOUSER')</td>
</tr>
<tr>
<td>• BOUSER - User name entered by the user to log into the SAP BusinessObjects BI platform.</td>
<td></td>
</tr>
<tr>
<td>• DBUSER - User name used for authorization when connecting to the data source. This username can be defined in the Central Management Console as part of the user's secondary credentials.</td>
<td></td>
</tr>
<tr>
<td>Variables containing information about the current report or query:</td>
<td>For example, these variables can be referenced in the BEGIN_SQL parameter that will be executed before the SELECT statement. This can be used for audit purposes concerning the use of the database (For example: To determine which report query or which universe is used most frequently).</td>
</tr>
<tr>
<td>• DOCNAME - Name of the document.</td>
<td></td>
</tr>
<tr>
<td>• DPNAME - Name of the Data Provider.</td>
<td></td>
</tr>
<tr>
<td>• DPTYPE - Type of Data Provider.</td>
<td></td>
</tr>
<tr>
<td>• UNVNAME - Name of the universe.</td>
<td></td>
</tr>
<tr>
<td>• UNVID - The ID of the universe.</td>
<td></td>
</tr>
</tbody>
</table>
### Variable Name and Description

<table>
<thead>
<tr>
<th>Variables containing information about the user's current language settings:</th>
</tr>
</thead>
<tbody>
<tr>
<td>* PREFERRED_VIEWING_LOCALE - The user's preferred locale for viewing report and query objects in an application.</td>
</tr>
<tr>
<td>* DOMINANT_PREFERRED_VIEWING_LOCALE - A pre-defined fallback locale that is used when no fallback locale is defined for the resource.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>The following query retrieves the product names in the language determined by the user's Preferred Viewing Locale. The database must contain a column indentifying the locale of the data. To see a list of locales, their abbreviations and dominant locales, refer to the <em>Translation Management Tool User Guide</em>.</td>
</tr>
</tbody>
</table>

```sql
SELECT Product_Name
FROM PRODUCT
WHERE PRODUCT.LOCALE = @Variable('PREFERRED_VIEWING_LOCALE')
```

| User Attributes defined in the User Attribute Management area of the Central Management Console (CMC). |

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>To reference a User Attribute, specify the internal name for the attribute as it is defined in the CMC. @Variable returns the value of the attribute for the current user. For example, the User Attribute MYCOUNTRY contains the value of the country of each user in the CMC. Specify the attribute's internal name surrounded by single quotes:</td>
</tr>
</tbody>
</table>

```sql
@Variable('SI_MYCOUNTRY')
```

The attribute's internal name is defined when the attribute is created in the CMC.

**Note:**

If the Variable_name specified in the @Variable function is unknown to the system, the user is prompted for a value. In this case, the @Variable function behaves in the same way as a single-value @Prompt function with the following settings:

```sql
@Prompt('Variable_name','A',mono,free)
```

The @Variable function is allowed in the following expressions:

- Joins
- Calculated columns

**Note:**

In database-specific SQL (multisource-enabled data foundations), all referenced tables or columns must be from the same connection.

- Derived tables
- Objects in the business layer
18.2.3 About @DerivedTable

Use the @DerivedTable function in the definition of nested derived tables. A nested derived table (also known as a ‘derived table on a derived table’) is a table that is derived from at least one existing derived table.

The syntax of the @DerivedTable is @DerivedTable(<Derived_table_name>). Derived_table_name is the name of the derived table you want to reference. The @DerivedTable function is only used in the definition of derived tables in the data foundation.

**Note:**
In database-specific SQL (multisource-enabled data foundations), all tables referenced must be from the same connection.

18.2.4 About @Aggregate_Aware

Use the @Aggregate_Aware function in the SQL definition of business layer objects to direct the object to first query all the aggregate tables listed as parameters in the @Aggregate_Aware function.

The syntax is @Aggregate_Aware(sum(agg_table_1), ... sum(agg_table_n)) where agg_table_1 is the aggregate with the highest level of aggregation, and agg_table_n the aggregate with the lowest level.

For more information about aggregate awareness in the universe, see the related topic.

**Related Topics**
* About aggregate awareness

18.2.5 About @Select

Use the @Select function in the definition of an object in the business layer to re-use the SELECT statement of another object. The syntax of the @Select function is @select(<full_path\object_name>). The argument full_path\object_name specifies the path of another object in the business layer.

For example, you define a business layer object Promotional_Service_Line as @select(Resort\Service_Line). The SELECT statement defined for Service_Line is used for the definition of Promotional_Service_Line.
Using the @Select function lets you maintain only one instance of the SQL or MDX expression, and ensures the consistency of related object definitions in the business layer. However, @Select creates an object dependency. If you delete the source object, you must manually update the object that uses the @Select function.

### 18.2.6 About @Where

Use the @Where function in the SQL definition of an object in the business layer to re-use the WHERE clause of another object. The syntax of the @Where function is `@select(<full_path\object_name>)`. The argument `full_path\object_name` specifies the path of another object in the business layer.

For example, you define the WHERE clause of business layer object Resort_Service_Line as `@Where(dimResort\Resort)`. The WHERE statement defined for the object Resort is used for the definition of Resort_Service_Line.

Using the @Where function lets you maintain only one instance of the SQL WHERE clause, and ensures the consistency of related object definitions in the business layer. However, @Where creates an object dependency. If you delete the source object, you must manually update the object that uses the @Where function.

### 18.3 About SQL Generation Parameters

SQL generation parameters affect the generation of the query script. The parameters all have default values. Default values can be overridden in the data foundation properties. Some parameters (concerning lists of values) can also be overridden in the business layer properties. At query time, the query server will use the values it finds in the following order:

1. The value in the business layer, if it is set.
2. The value in the data foundation, if it is set.
3. The default value.

The following reference describes the parameters that affect the generation of the query script. The parameters are listed alphabetical order in two groups:

- SQL parameters that you set in the user interface of the information design tool. These are SQL parameters that are common to most data access drivers. Each parameter is valid for the universe in which it is set.
- SQL parameters that you set in the data access parameter (PRM) files. These are connection-specific parameters that are listed in the PRM file for the target data access driver.

**Related Topics**

- SQL generation parameters reference
- SQL generation parameters set in the PRM
18.3.1 SQL generation parameters reference

The following reference describes the SQL generation parameters that can be overridden in the data foundation properties and business layer properties.

18.3.1.1 ANSI92

ANSI92 = Yes|No

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes/No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
</tr>
<tr>
<td>Description</td>
<td>Specifies whether the SQL generated complies to the ANSI92 standard. Yes: Enables the SQL generation compliant to ANSI92 standard. No: SQL generation behaves according to the PRM parameter OUTER_JOIN_GENERATION.</td>
</tr>
</tbody>
</table>

18.3.1.2 AUTO_UPDATE_QUERY

AUTO_UPDATE_QUERY = Yes|No

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes/No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
</tr>
</tbody>
</table>
| Description | Determines what happens when an object in a query is not available to a user profile.  
Yes: Query is updated and the object is removed from the query.  
No: Object is kept in the query. |

### 18.3.1.3 BEGIN_SQL

BEGIN_SQL = <String>

<table>
<thead>
<tr>
<th>Values</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Empty string</td>
</tr>
</tbody>
</table>

**Description**

BEGIN_SQL is used to prefix SQL statements for accounting, prioritization, and workload management. The parameter applies to any SQL generation, including document generation and list of values queries. BEGIN_SQL is supported in Web Intelligence, LiveOffice, Crystal Reports for Enterprise, and QaaWS. It is ignored by Desktop Intelligence.

Example for Teradata:

```sql
BEGIN_SQL=SET QUERY_BAND='string' for transaction;
```

This parameter requires a string that contains one or more name-value pairs, separated by a semicolon, all inside single quotes. All SQL statements are prefixed with the parameter that follows `BEGIN_SQL`. The name-value pairs entered in this parameter are written in the GetQueryBandPairs system table.

Example of three name-value pairs:

```sql
BEGIN_SQL=SET QUERY_BAND='UserID=Jones;JobID=980;AppID=TRM' for transaction;
```

You can also use the `@Variable` function as the value in the name-value pair, the returned value is enclosed in single quotes:

```sql
BEGIN_SQL=SET QUERY_BAND='USER='@Variable('BOUSER');Document='@Variable('DPNAME')';' for transaction;
```
### 18.3.1.4 BLOB_COMPARISON

**BLOB_COMPARISON = Yes|No**

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes/No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
</tr>
<tr>
<td>Can be edited?</td>
<td>No</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>Species if a query can be generated with a <code>DISTINCT</code> statement when a BLOB file is used in the <code>SELECT</code> statement. It is related to the setting <code>No Duplicate Row</code> in the query properties. Yes: The <code>DISTINCT</code> statement can be used within the query. No: The <code>DISTINCT</code> statement cannot be used within the query even if the query setting <code>No Duplicate Row</code> is on.</td>
<td></td>
</tr>
</tbody>
</table>

### 18.3.1.5 BOUNDARY_WEIGHT_TABLE

**BOUNDARY_WEIGHT_TABLE = Integer 32bits [0-9]**

<table>
<thead>
<tr>
<th>Values</th>
<th>Integer 32bits [0-9, or a negative integer]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>-1</td>
</tr>
</tbody>
</table>
Allows you to optimize the **FROM** clause when tables have many rows. If the table size (number of rows) is greater than the entered value, the table is declared as a subquery:

```
FROM (SELECT col1, col2, ...., coln, , ...., FROM Table_Name WHERE simple condition).
```

A simple condition is defined as not having a subquery. -1, 0, or any negative number means that this optimization is not used.

---

### Limitations

Optimization is not implemented when:
- The operator **OR** is in the query condition
- Only one table is involved in the SQL
- The query contains an outer join
- No condition is defined on the table that is being optimized
- The table being optimized is a derived table.

---

**18.3.1.6 CUMULATIVE_OBJECT_WHERE**

**CUMULATIVE_OBJECT_WHERE** = **Yes|No**

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td></td>
<td>No</td>
</tr>
</tbody>
</table>
This parameter applies to filtered objects only. Specifies how to combine the objects `WHERE` clause with the query condition on those objects.

**Yes**: Specifies that `WHERE` clauses are combined with the main query condition with the **AND** operator.

**No**: Specifies that the object's `WHERE` clause is combined with the condition for this object.

**Example:**

If the condition is find all French clients different from John or American cities different from New York, the SQL is:

**Yes**:

```sql
(customer.first_name <> 'John')
OR (city.city <> 'New York'
AND customer_country.country = 'France'
AND city_country.country = 'USA')
```

**No**:

```sql
(customer.first_name <> 'John' AND
customer_country.country = 'France'
)
OR (city.city <> 'New York' AND
city_country.country = 'USA'
)
```

---

### 18.3.1.7 DISABLE_ARRAY_FETCH_SIZE_OPTIMIZATION

**DISABLE_ARRAY_FETCH_SIZE_OPTIMIZATION = Yes|No**

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes/No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
</tr>
</tbody>
</table>

**Description**

- **An optimization algorithm can be used to optimize the size of the returned arrays instead of using the default setting.**

- **No**: All queries run on the universe will benefit from the optimization.

- **Yes**: Queries use the default value set.
18.3.1.8 DISTINCT_VALUES

DISTINCT_VALUES = GROUPBY|DISTINCT

<table>
<thead>
<tr>
<th>Values</th>
<th>GROUPBY</th>
<th>DISTINCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>DISTINCT</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

Specifies whether SQL is generated with a DISTINCT or GROUP BY clause for objects in the business layer, and in lists of values. In the Query Panel, query generation takes the value of DISTINCT_VALUES into account only when the option Retrieve duplicate rows is unselected in the query properties.

DISTINCT: The SQL is generated with a DISTINCT clause, for example:

```sql
SELECT DISTINCT cust_name FROM Customer
```

GROUPBY: The SQL is generated with a GROUP BY clause, for example:

```sql
SELECT cust_name FROM Customer GROUP BY Customer.cust_name
```

18.3.1.9 END_SQL

END_SQL = String

<table>
<thead>
<tr>
<th>Values</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>&lt;empty string&gt;</td>
</tr>
</tbody>
</table>

**Description**

The statement specified in this parameter is added at the end of each SQL statement.
For IBM DB2 databases, you can use the following:

\[ \text{END\_SQL=FOR SELECT ONLY} \]

The server will read blocks of data much faster.

Another example:

\[ \text{END\_SQL=’write ‘ UNVID To Usage\_Audit\_Querieded\_universe} \]

Would write universe id to an audit table, this can be used to record other data such as user and tables queried.

### 18.3.1.10 EVAL\_WITHOUT\_PARENTHESIS

**EVAL\_WITHOUT\_PARENTHESIS = Yes|No**

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

By default, the function \[@Select(Class\object)\] is replaced by the SELECT statement for the object <Class\object> enclosed within brackets.

For example, when combining two @Select statements, @Select(object1) * @Select(object2).

If the SQL(object1) = A-B and SQL(object2) = C,

then the operation is (A-B) * (C).

You avoid the default adding of brackets by setting EVAL\_WITHOUT\_PARENTHESIS = Yes. The operation is then A - B * C.

Yes: Brackets are removed from the SELECT statement for a function @Select(Class\object)

No: Brackets are added around the Select statement for the function @Select(Class\object).
### 18.3.1.11 FILTER_IN_FROM

**FILTER_IN_FROM = Yes|No**

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes/No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
</tr>
</tbody>
</table>

**Description**

Determines if the generated SQL includes query filters in the FROM clause whenever possible.

**Note:**

This setting is only applicable if the SQL generation parameter **ANSI92** is set to **Yes**.

This parameter is useful when querying tables that have outer joins defined. For example, an outer join on tables Customer and Reservations returns all customers, even those without reservations. A query filter in the WHERE clause might filter out customers without reservations. If the **FILTER_IN_FROM** parameter is set to **Yes**, the generated SQL includes query filters in the FROM clause whenever possible in order to preserve the null values returned by the outer join.

**Yes:** When SQL is generated, query filters are put in the FROM clause whenever possible.

**No:** When SQL is generated, query filters are put in the WHERE clause.

### 18.3.1.12 FORCE_SORTED_LOV

**FORCE_SORTED_LOV = Yes|No**

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

| Description | Retrieves a list of values that is sorted. Yes: Specifies that the list of values is sorted. No: Specifies that the list of values is not sorted. |

### 18.3.1.13 GROUPBY_PRIMARY_KEY

\[ \text{GROUPBY\_PRIMARY\_KEY} = \text{YES} \mid \text{NO} \]

<table>
<thead>
<tr>
<th>Values</th>
<th>YES \mid NO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>YES</td>
</tr>
<tr>
<td>Description</td>
<td>Allows you to deactivate the use of the primary key in the \texttt{GROUP BY} clause. By default, if data for an index aware object is retrieved, the SQL is optimized by using the primary key in the \texttt{GROUP BY} clause. YES: Favors the use of the primary key over the column name in the \texttt{GROUP BY} clause. NO: Does not use the primary key in the \texttt{GROUP BY} clause.</td>
</tr>
</tbody>
</table>

### 18.3.1.14 INNERJOIN_IN_WHERE

\[ \text{INNERJOIN\_IN\_WHERE} = \text{Yes} \mid \text{No} \]

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes \mid No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No. You must manually add the parameter to activate it.</td>
</tr>
</tbody>
</table>
### Description

Allows you to force the system to generate SQL syntax with all the inner joins in the `WHERE` clause when `ANSI92` is set to yes. This is only possible if a query contains only inner joins (Does not contain FULL OUTER, RIGHT OUTER, or LEFT OUTER joins).

**Yes:** If `ANSI92` is set to yes, the system generates ANSI92 join syntax in the `FROM` clause except when the query contains only inner joins. In this case, the inner joins go into the `WHERE` clause.

**No:** If `ANSI92` is set to yes, the system generates ANSI 92 join syntax in the `FROM` clause.

### 18.3.1.15 JOIN_BY_SQL

**JOIN_BY_SQL = Yes|No**

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

### Description

Specifies how multiple SQL statements are handled. Multiple statements can be combined (provided that the database permits this).

**Yes:** Specifies that multiple SQL statements are combined.

**No:** Specifies that multiple SQL statements are not combined. This is the default value.

### 18.3.1.16 MAX_INLIST_VALUES

**MAX_INLIST_VALUES = [0-99]**

<table>
<thead>
<tr>
<th>Values</th>
<th>Integer: min-1, max depends on DB</th>
</tr>
</thead>
</table>

---

301 2012-03-29
**18.3.1.17 REPLACE_COMMA_BY_CONCAT**

**REPLACE_COMMA_BY_CONCAT** = Yes|No

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

In previous versions of the universe design tool, a comma could be used to separate multiple fields in an object Select statement. The comma was treated as a concatenation operator. For universes that already use the comma in this way you can set REPLACE_COMMA_BY_CONCAT to No to keep this behavior. In the current version of the universe design tool, this parameter is set to Yes by default, so that any expressions using a comma in this way are automatically changed to use concatenation syntax.

Yes: Comma is replaced by the concatenation expression when multi field object is found.

No: Keep the comma as it is.
18.3.1.18 SELFJOINS_IN_WHERE

SELFJOINS_IN_WHERE = Yes|No

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

Description

Self-joins are usually included in the FROM clause. This allows you to force the system to generate SQL syntax with all the conditions of a self-join in the WHERE clause. The ANSI92 parameter must be set to Yes for this parameter to be taken into account.

You must manually add the parameter to the list to activate it.

Yes: The conditions of a self-join go in the WHERE clause of the SQL query.

No: The syntax for self-joins is generated according to the ANSI 92 convention, and conditions for a self-join go in the ON clause of the table join definition in the FROM clause of the SQL query.

18.3.1.19 SHORTCUT_BEHAVIOR

SHORTCUT_BEHAVIOR = ShortestPath|Global|Successive

<table>
<thead>
<tr>
<th>Values</th>
<th>ShortestPath</th>
<th>Global</th>
<th>Successive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>ShortestPath</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SQL and MDX reference
Specifies how shortcut joins are applied.

ShortestPath: applies shortcuts so as to obtain the smallest number of tables in the query.

Successive: applies shortcuts one after the other. If a shortcut removes a table involved in a potential successive shortcut, the successive shortcut is not applied.

Global: applies all shortcuts. If the resulting query creates a Cartesian product, no shortcut joins are applied.

Note:
This parameter was formerly listed as GLOBAL_SHORTCUTS in the PRM files. The value Global corresponds to Yes, and Successive corresponds to No.

### 18.3.1.20 SMART_AGGREGATE

SMART_AGGREGATE = Yes|No

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
</table>

| Default | No  |
Determines how aggregate tables are used for smart measures that are based on an aggregate tables. This ensures that a universe object based on a ratio is correctly aggregated. By default the system takes the advantage of the pre-calculated values from the aggregated tables, if these table are not consistent during time (different time periods), you use this parameter to ensure the most detailed aggregate tables are used.

This parameter is not visible in the universe parameter list (by default not activated). The universe designer must manually insert it in the parameter list before activating it (value Yes).

Yes: Any additional grouping set query should be based on the aggregate table of the initial query for the smart measure based on aggregate table.

No: The system takes the most appropriate aggregate table.

### 18.3.1.21 THROUGH_AGGREGATE_AWARE

<table>
<thead>
<tr>
<th>THROUGH_AGGREGATE_AWARE = Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Values</strong></td>
<td>Yes/No</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>For universes converted from .unv, the default value is No.</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Determines if aggregate awareness is taken into account when testing the compatibility of query objects. This parameter allows you to possibly improve the results of queries on converted .unv universes that fail for the .unv universe. Yes: Aggregate awareness is taken into account when testing the compatibility of objects in the query. In some cases this allows the query to succeed in situations where there are incompatible objects (split queries) with aggregate aware objects. No: Compatibility testing of objects uses the behavior for .unv universes.</td>
</tr>
</tbody>
</table>
### 18.3.1.22 THOROUGH_PARSE

**THOROUGH_PARSE = Yes|No**

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Specifies the methodology used for default Parsing in the Query pane and individual object parsing.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes: <strong>PREPARE</strong>, <strong>DESCRIBE</strong>, and <strong>EXECUTE</strong> statements are used to parse SQL for objects.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prepare+DescribeCol+Execute</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No: <strong>PREPARE</strong> and <strong>DESCRIBE</strong> statements are used to parse SQL for objects.</td>
<td></td>
</tr>
</tbody>
</table>

### 18.3.1.23 TRUST_CARDINALITIES

**TRUST_CARDINALITIES = Yes|No**

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Allows you to optimize the SQL in case of inflated results.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes: For queries that include a measure, all conditions that inflate the measure and do not appear in the Result Objects, are transformed to sub queries to ensure that tables that may return false results for the measure are not included in the query.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No: No optimization is implemented.</td>
<td></td>
</tr>
</tbody>
</table>
18.3.1.24 UNICODE_STRINGS

UNICODE_STRINGS = Yes|No

<table>
<thead>
<tr>
<th>Values</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

Specifies whether the current universe can manipulate Unicode strings or not. Only applies to Microsoft SQL Server and Oracle 9. If the database character set in the SBO file is set as Unicode, then it is necessary to modify the SQL generation to handle specific Unicode column types like NCHAR and NVARCHAR.

**Yes:** Conditions based on strings are formatted in the SQL according to the value for a parameter UNICODE_PATTERN in the PRM file, for example for MS SQL Server (sqlsrv.prm) : UNICODE_PATTERN=N$

The condition Customer_name='Arai ' becomes

Customer_name=N'Arai'.

**Note:** When you create a prompt with @Prompt syntax based on Unicode value, the data type should be 'U' not 'C'.

**No:** All conditions based on strings are formatted in the standard SQL. For example the condition Customer_name='Arai ' remains Customer_name='Arai'.

18.3.2 SQL generation parameters set in the PRM

The following reference describes the SQL generation parameters that you set in the data access parameter (PRM) file for the target data access driver.
### 18.3.2.1 CASE_SENSITIVE

<Parameter Name="CASE_SENSITIVE">NO</Parameter>

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies if the database is case-sensitive. This parameter is used with Oracle.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values</td>
<td>YES: the database is case-sensitive. NO: the database is not case-sensitive.</td>
</tr>
<tr>
<td>Default</td>
<td>NO</td>
</tr>
</tbody>
</table>

### 18.3.2.2 COMMA

<Parameter Name="COMMA">||' '|</Parameter>

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies what database concatenation operator should be used to replace a comma for objects that have the following syntax: Tab.Col1, Tab.Col2. This parameter is used with all data access drivers.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values</td>
<td></td>
</tr>
<tr>
<td></td>
<td>+ ' '+</td>
</tr>
<tr>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>Tab.Col1</td>
</tr>
</tbody>
</table>
18.3.2.3 CONCAT

<Parameter Name="CONCAT">||</Parameter>

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the concatenation operator. The parameter is used with all data access drivers.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values</td>
<td>double pipe (</td>
</tr>
<tr>
<td>Default</td>
<td></td>
</tr>
</tbody>
</table>

18.3.2.4 DELIMIT_IDENTIFIERS

<Parameter Name="DELIMIT_IDENTIFIERS">YES</Parameter>

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies if database identifiers can be quoted. Identifiers are quoted using the delimiter specified in the IDENTIFIER_DELIMITER parameter.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values</td>
<td>YES: identifiers can be quoted.</td>
</tr>
<tr>
<td></td>
<td>NO: identifiers cannot be quoted.</td>
</tr>
<tr>
<td>Default</td>
<td>YES</td>
</tr>
<tr>
<td>Result</td>
<td>Table name=&quot;my_table&quot;</td>
</tr>
</tbody>
</table>

18.3.2.5 DELIMIT_LOWERCASE

<Parameter Name="DELIMIT_LOWERCASE"></Parameter>
### 18.3.2.6 EXTERN_SORT_EXCLUDE_DISTINCT

<Parameter Name="EXTERN_SORT_EXCLUDE_DISTINCT">YES</Parameter>

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies if the application generates a SELECT DISTINCT when a query contains an ORDER BY clause.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values</td>
<td>YES: a SELECT DISTINCT is not generated when the query contains an ORDER BY clause.</td>
</tr>
<tr>
<td></td>
<td>NO: a DISTINCT is generated when the query contains an ORDER BY clause.</td>
</tr>
<tr>
<td>Default</td>
<td>YES</td>
</tr>
</tbody>
</table>

### 18.3.2.7 GROUPBY_WITH_ALIAS

<Parameter Name="GROUPBY_WITH_ALIAS">YES</Parameter>

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies if the database can create a GROUP BY clause that contains aliases in the SELECT statement.</th>
</tr>
</thead>
</table>
### 18.3.2.8 IDENTIFIER_DELIMITER

| Values | YES: it allows you to create a GROUP BY clause with aliases in the SELECT statement.  
| Default | YES 

| NO: it does not let you create a GROUP BY clause with aliases in the SELECT statement. |

**Description**

Specifies the following features:

- Table or column names that contain spaces or special characters are enclosed within quotes if `BACK_QUOTE_SUPPORTED` parameter is activated.
- Tables or column names regardless of their characters are enclosed within quotes if `DELIMIT_IDENTIFIERS` parameter is activated.

To use this parameter, either `BACK_QUOTE_SUPPORTED` or `DELIMIT_IDENTIFIERS` must be set to `YES`. This is the default value of both parameters.

| Values | " (double quote): table or column names that contain spaces or special characters are enclosed in double quotes.  
| Default | "  

' (single quote): table or column names that contain spaces or special characters are enclosed in single quotes. This value can only be used with Microsoft Access.

| Result | Table name="My Table"
### 18.3.2.9 OUTERJOINS_GENERATION

<Parameter Name="OUTERJOINS_GENERATION">ANSI92</Parameter>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
</table>
| Specifies the SQL syntax for outer joins. The value **ANSI 92** generates an outer join in the FROM clause. Other values generate the outer join in the WHERE clause. When you modify this setting, you should check join properties to verify that the outer join expression is valid, and that the cardinalities are correct. ANSI92 does not support any manual customization in the join syntax. **Note:** The PRM file OUTERJOINS_GENERATION parameter relates to the universe ANSI92 setting in the following way:  
  - If the PRM file OUTERJOINS_GENERATION parameter is set to **ANSI_92** and the universe ANSI92 parameter is set to **NO**, the PRM parameter overrides the universe setting and outer joins conform to ANSI92 behavior.  
  - If the PRM file OUTERJOINS_GENERATION parameter is set to **USUAL**, then the universe ANSI92 setting takes precedence, and outer joins conform to ANSI92 depending on whether the universe ANSI92 setting is **YES** or **NO**. |
| Values |
| The primary values for OUTERJOINS_GENERATION are:  
  - **ANSI_92**: the default outer join behavior conforms to the ANSI92 standard regardless of the ANSI92 parameter value in the universe.  
  - **NO**: outer joins are not supported.  
  - **USUAL**: the database-specific outer join behavior is used. This behavior is overridden if the ANSI92 parameter is set to **YES**. Other settings are available depending on the database. See the defaults below. |
Examples of OUTERJOINS_GENERATION parameter settings

**Setting = USUAL:**

```
FROM T1, T2
WHERE T1.col1(+) = T2.col2
```

**Setting = DB2:**

```
FROM T2 LEFT OUTER JOIN T1
ON T1.col1 = T2.col2
```

**Setting = ODBC:**

```
FROM {oj T1 LEFT OUTER JOIN T2 ON T1.col1=T2.col2}
WHERE (T2.col3 = T3.col1)
```

**Setting = INFORMIX**

```
FROM T2
OUTER T1
WHERE T1.col1=T2.col2
```

**Setting = FULL-ODBC**

```
FROM {oj T1 RIGHT OUTER JOIN T2 ON T2.col2=T1.col1
T2 INNER JOIN 3 on T2.col3 = T3.col1}
```

**Setting = ANSI_92:**

```
SELECT DISTINCT
  t1.col1,
  t2.col2
FROM
  (t1 RIGHT OUTER JOIN t2 ON t1.col1=t2.col2) 
```
<Parameter Name="OVER_CLAUSE">YES</Parameter>

| Description | Allows SAP BusinessObjects applications to include RISQL functions when generating SQL. The supported RISQL functions for the database are listed in the ANALYTIC_FUNCTIONS parameter. |
| Values | YES: applications can include RISQL functions when generating SQL. NO: applications cannot include RISQL functions when generating SQL. |
| Default | YES |

18.3.2.11 OWNER

<Parameter Name="OWNER">YES</Parameter>

| Description | Specifies if the database supports the owner name as prefix for tables. |
| Values | YES: the database supports prefixing tables with the owner name. NO: the database does not support prefixing tables with the owner name. |
| Default | YES |

18.3.2.12 QUALIFIER

<Parameter Name="QUALIFIER">NO</Parameter>

| Description | Specifies if the database supports the qualifier name as prefix for tables. |
YES: the database supports prefixing tables with the qualifier name.

NO: the database does not support prefixing tables with the qualifier name.

**Values**

| Default  | RDBMS-dependent. |

---

### 18.3.2.13 UNICODE_PATTERN

<Parameter Name="UNICODE_PATTERN">UNISTR($)</Parameter>

<table>
<thead>
<tr>
<th>Description</th>
<th>Only applies when the universe SQL generation parameter UNICODE_STRINGS is set to YES. All conditions based on strings are then formatted with this string value. This is used with MS SQL Server and Oracle only.</th>
</tr>
</thead>
</table>
| **Values**  | N$: for MS SQL Server  
UNISTR($): for Oracle |

### 18.3.2.14 USER_INPUT_DATE_FORMAT

<Parameter Name="USER_INPUT_DATE_FORMAT">'dd-MM-yyyy HH:mm:ss'</Parameter>

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the default date and hour formats generated in the WHERE clause of a SQL statement.</th>
</tr>
</thead>
</table>
| Values                           | \{\d 'yyyy-mm-dd'}: default date format with ODBC.  
|                                | 'DD-MM-YYYY HH:MM:SS': default date and hour formats with Oracle.  
|                                | 'MM/DD/YYYY': default date format with IBM Informix.  
|                                | 'yyyy-mm-dd HH:mm:ss': default date and hour formats with MS SQL Server and for most IBM DB2 servers.  
|                                | 'mm/dd/yyyy hh:mm:ss am/pm': default date and hour formats with Sybase.  
|                                | 'yyyy-mm-dd': default date format with a Sybase gateway.  
| Note:                          | If you need to use time or timestamp variables with ODBC, you must replace the default date format value with \{\t 'hh:mm:ss'} or \{\t\s 'yyyy-mm-dd hh:mm:ss'} in the odbc.sbo file.  
| Default                        | See values above.  

18.3.2.15 USER_INPUT_NUMERIC_SEPARATOR

<Parameter Name="USER_INPUT_NUMERIC_SEPARATOR">.</Parameter>

| Description | Specifies the default decimal separator that is used in the generated SQL script.  
|-------------|------------------------------------------------------  
| Values      | 
| Default     | 
|             | '.' (period)  
|             | '.'  
|
19.1 About the Format Editor

The Format Editor lets you define the format used to display date-time and numeric values. You can select default formats or define custom formats. For metadata objects in your document, you can define a format for each locale. The formats are saved in the document as part of its localization.

The application uses the current system date and time and the numeric value 987,654,321 to display formats. In the application preferences, you can define your own number and date-time values to use when displaying formats.

**Default Formats**

Default formats are available based on Common Locale Data Repository (CLDR) recommendations, managed by the UNICODE consortium. Microsoft standards, whenever they differ from the CLDR recommendation, are followed. There are 15 default formats for date and time values, and four default formats for numbers.

The Format Editor lists the default formats available for each category of data.

**Custom Formats**

If the default formats available do not meet your needs, you can create formats using the Custom Format Editor.

You can delete a custom format even if it is used in the document. The next time you start the Format Editor for an object using the deleted format, a custom format is automatically re-created.

A custom format consists of text and tokens. A token is a pre-formatted part of a number or date. For example, **Day: 1-31** is a token that displays the day part of the date as a number between 1 and 31. For more information about tokens, see the related topics.

The Custom Format Editor lists the possible token categories. You define a format by typing text in the **Format Definition** box, and dragging tokens from the **Tokens** box. Tokens appear in the format definition with a rectangle border and a gray background. As you define the format, a preview shows the resulting appearance.

You can define a display color for each custom format. When no color is defined, the client application (for example, Interactive Analysis) manages the color to display.

In the **Format for undefined values** box, you can define text and a color to display if no value is returned at reporting time. By default, in the case of an undefined value, no text is displayed.
When defining a numeric format, you can enter a different format to be displayed when the value is negative or equal to zero. If you do not enter a format, the format defined for positive values is used.

Related Topics
• Date and time format tokens
• Number format tokens

19.2 Date and time format tokens

Example: Date and time format display

This example shows how the date, Wednesday March 5th 2008, is displayed using different formats defined in the Custom Format Editor.
### List of date and time tokens

<table>
<thead>
<tr>
<th>Category</th>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day</td>
<td>Day 01-31</td>
<td>Day in the month with two digits from 01 to 31.</td>
</tr>
<tr>
<td>Day</td>
<td>Day 1-31</td>
<td>Day in the month with one or two digits from 1 to 31.</td>
</tr>
<tr>
<td>Day</td>
<td>Day name</td>
<td>Day name according to the locale, for example, Monday.</td>
</tr>
<tr>
<td>Day</td>
<td>Short day name</td>
<td>Short day name with capitalization according to the locale, for example, Mon.</td>
</tr>
<tr>
<td>Day</td>
<td>Day of year 001-366</td>
<td>Day in the year with three digits from 001 to 366.</td>
</tr>
<tr>
<td>Day</td>
<td>Day of year 01-366</td>
<td>Day in the year with two or three digits from 01 to 366.</td>
</tr>
<tr>
<td>Day</td>
<td>Day of year 1-366</td>
<td>Day in the year with one, two, or three digits from 1 to 366.</td>
</tr>
<tr>
<td>Day</td>
<td>Day of week in month</td>
<td>Day of the week in the month according to the locale, for example, 3 for the 3rd Monday of June.</td>
</tr>
<tr>
<td>Day</td>
<td>Upper-case day name</td>
<td>Day name in upper case, for example, MONDAY.</td>
</tr>
<tr>
<td>Day</td>
<td>Lower-case day name</td>
<td>Day name in lower case, for example, monday.</td>
</tr>
<tr>
<td>Day</td>
<td>Capitalized day name</td>
<td>Capitalized day name, for example, Monday.</td>
</tr>
<tr>
<td>Day</td>
<td>Upper-case short day name</td>
<td>Short day name in upper case, for example, MON.</td>
</tr>
<tr>
<td>Day</td>
<td>Lower-case short day name</td>
<td>Short day name in lower case, for example, mon.</td>
</tr>
<tr>
<td>Category</td>
<td>Token</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Month</td>
<td>Capitalized short day name</td>
<td>Capitalized short day name, for example, Mon.</td>
</tr>
<tr>
<td></td>
<td>Month 01-12</td>
<td>Month in the year with two digits from 01 to 12.</td>
</tr>
<tr>
<td></td>
<td>Month 1-12</td>
<td>Month in the year with one or two digits from 1 to 12.</td>
</tr>
<tr>
<td></td>
<td>Month name</td>
<td>Month name with capitalization according to the locale, for example, June.</td>
</tr>
<tr>
<td></td>
<td>Short month name</td>
<td>Short month name with capitalization according to the locale, for example, Jun.</td>
</tr>
<tr>
<td></td>
<td>Upper-case month name</td>
<td>Month name in upper case, for example, JUNE.</td>
</tr>
<tr>
<td></td>
<td>Lower-case month name</td>
<td>Month name in lower case, for example, june.</td>
</tr>
<tr>
<td></td>
<td>Capitalized month name</td>
<td>Capitalized month name, for example, June.</td>
</tr>
<tr>
<td></td>
<td>Upper-case short month name</td>
<td>Short month name in upper case, for example, JUN.</td>
</tr>
<tr>
<td></td>
<td>Lower-case short month name</td>
<td>Short month name in lower case, for example, jun.</td>
</tr>
<tr>
<td></td>
<td>Capitalized short month name</td>
<td>Capitalized short month name, for example, Jun.</td>
</tr>
<tr>
<td>Year and Era</td>
<td>Year 00-99</td>
<td>Year with two digits from 00 to 99.</td>
</tr>
<tr>
<td></td>
<td>Year 0000-9999</td>
<td>Year with four digits from 0000 to 9999.</td>
</tr>
<tr>
<td></td>
<td>Japanese Imperial period and year</td>
<td>Japanese Imperial period and year number, for example, 平成 20.</td>
</tr>
<tr>
<td></td>
<td>Japanese Imperial period (English) and year</td>
<td>Japanese Imperial period (English abbreviated) and year number, for example, H20.</td>
</tr>
<tr>
<td></td>
<td>Japanese Imperial year number 01-99</td>
<td>Japanese Imperial year number with two digits.</td>
</tr>
<tr>
<td></td>
<td>Japanese Imperial year number 1-99</td>
<td>Japanese Imperial year number with one or two digits.</td>
</tr>
<tr>
<td></td>
<td>Japanese Imperial period</td>
<td>Japanese Imperial period.</td>
</tr>
<tr>
<td></td>
<td>Japanese Imperial year</td>
<td>Deprecated. Returns the same result as Japanese Imperial year number 0-99 token.</td>
</tr>
<tr>
<td></td>
<td>Era</td>
<td>Era abbreviation, for example, AD or BC.</td>
</tr>
<tr>
<td>Category</td>
<td>Token</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Week</td>
<td>Week of month</td>
<td>Week in the month with one digit from 1 to 6.</td>
</tr>
<tr>
<td></td>
<td>Week of year 01-53</td>
<td>Week in the year (ISO week) with two digits from 01 to 53.</td>
</tr>
<tr>
<td></td>
<td>Week of year 1-53</td>
<td>Week in the year (ISO week) with one or two digits from 1 to 53.</td>
</tr>
<tr>
<td></td>
<td>Year of week of year 0000</td>
<td>ISO year number (consistent with ISO week) with four digits from 0000 to 9999.</td>
</tr>
<tr>
<td></td>
<td>Year of week of year 00</td>
<td>ISO year number (consistent with ISO week) with two digits from 00 to 99.</td>
</tr>
<tr>
<td>Quarter and Semester</td>
<td>Quarter number 1-4</td>
<td>Quarter number with one digit from 1 to 4.</td>
</tr>
<tr>
<td></td>
<td>Short quarter name</td>
<td>Quarter short name from Q1 to Q4.</td>
</tr>
<tr>
<td></td>
<td>Quarter name</td>
<td>Quarter name from 1st quarter to 4th quarter.</td>
</tr>
<tr>
<td></td>
<td>Semester 1-2</td>
<td>Semester number from 1 to 2.</td>
</tr>
<tr>
<td>Hour</td>
<td>Hour 00-23</td>
<td>Hour in 24-hour format with two digits from 00 to 23.</td>
</tr>
<tr>
<td></td>
<td>Hour 0-23</td>
<td>Hour in 24-hour format with one or two digits from 0 to 23.</td>
</tr>
<tr>
<td></td>
<td>Hour 01-12</td>
<td>Hour in 12-hour format with two digits from 01 to 12.</td>
</tr>
<tr>
<td></td>
<td>Hour 1-12</td>
<td>Hour in 12-hour format with one or two digits from 1 to 12.</td>
</tr>
<tr>
<td></td>
<td>Hour 01-24</td>
<td>Hour in 24-hour format with two digits from 01 to 24.</td>
</tr>
<tr>
<td></td>
<td>Hour 1-24</td>
<td>Hour in 24-hour format with one or two digits from 1 to 24.</td>
</tr>
<tr>
<td></td>
<td>Hour 00-11</td>
<td>Hour in 12-hour format with two digits from 00 to 11.</td>
</tr>
<tr>
<td></td>
<td>Hour 0-11</td>
<td>Hour in 12-hour format with one or two digits from 0 to 11.</td>
</tr>
<tr>
<td>Minute</td>
<td>Minutes 00-59</td>
<td>Minutes with two digits from 00 to 59.</td>
</tr>
<tr>
<td></td>
<td>Minutes 0-59</td>
<td>Minutes with one or two digits from 0 to 59.</td>
</tr>
<tr>
<td>Category</td>
<td>Token</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Seconds and sub-second</td>
<td>Seconds 00-59</td>
<td>Seconds with two digits from 00 to 59.</td>
</tr>
<tr>
<td></td>
<td>Seconds 0-59</td>
<td>Seconds with one or two digits from 0 to 59.</td>
</tr>
<tr>
<td></td>
<td>Milliseconds 000-999</td>
<td>Milliseconds with three digits from 000 to 999.</td>
</tr>
<tr>
<td></td>
<td>Hundredths of a second 000-999</td>
<td>Hundredths of a second with two digits from 00 to 99.</td>
</tr>
<tr>
<td></td>
<td>Tenths of a second 0-9</td>
<td>Tenths of a second with one digit from 1 to 9.</td>
</tr>
<tr>
<td>Time Zone</td>
<td>Time zone</td>
<td>The offset from Coordinated Universal Time, for example, GMT+00:00.</td>
</tr>
<tr>
<td>AM/PM</td>
<td>AM/PM</td>
<td>Morning/afternoon abbreviation, capitalized according to locale, for example, AM or PM. Recommended.</td>
</tr>
<tr>
<td></td>
<td>Upper-case AM/PM</td>
<td>Morning/afternoon abbreviation in upper-case, for example, AM or PM.</td>
</tr>
<tr>
<td></td>
<td>Lower-case am/pm</td>
<td>Morning/afternoon abbreviation in lower-case, for example, am or pm.</td>
</tr>
<tr>
<td></td>
<td>Capitalized Am/Pm</td>
<td>Capitalized morning/afternoon abbreviation, for example, Am or Pm.</td>
</tr>
<tr>
<td>Separator</td>
<td>Date separator</td>
<td>Deprecated. This token was used as a date separator in Desktop Intelligence and is not recommended. Type the character you wish to use as a date separator directly into the format description, or use a default format.</td>
</tr>
<tr>
<td></td>
<td>Time separator</td>
<td>Deprecated. This token was used as a time separator in Desktop Intelligence and is not recommended. Type the character you wish to use as a time separator directly into the format description, or use a default format.</td>
</tr>
</tbody>
</table>

### 19.3 Number format tokens

**Number format definitions**

A number format definition is made of sections:

- the sign (optional)
• the integer value before the decimal separator
• a grouping separator, to be added in the integer value
• the decimal separator (optional)
• the decimal value after the decimal separator (optional)
• the exponential symbol followed by the exponential value (optional)

Two tokens are used to define the number of significant digits to display in the integer, decimal, and exponential values. Each token in the format definition represents a digit to display:

• The mandatory digit token, 0, displays the digit if it is significant, otherwise displays a zero.
• The optional digit token, #, only displays the digit if it is significant.

When determining the significant digits, the integer value and exponential value are evaluated from right to left, and the decimal value is evaluated from left to right. The last 0 or # token is mapped to the remaining digits, if any.

Example: **Number format display**

This example shows how the value -1,234 is displayed using different formats defined in the Format Editor.

<table>
<thead>
<tr>
<th>Format defined with tokens:</th>
<th>Preview display:</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Sign] [#]</td>
<td>-1234</td>
</tr>
<tr>
<td>[Neg. start] [0] [0] [0] [0] [0] [0] [Neg. end]</td>
<td>(001234)</td>
</tr>
<tr>
<td>[Sign always] [#] [Dec.Sep.] [0] [0]</td>
<td>-1234.00</td>
</tr>
<tr>
<td>[Sign] [#] [Decimal separator] [0] [0] [E+] [0] [0] [0] [0]</td>
<td>-1.23E+003</td>
</tr>
<tr>
<td>Revenue: [Sign always] [#] [Decimal separator ] [0] [0]</td>
<td>Revenue: -1234.00 €</td>
</tr>
<tr>
<td>[Boolean]</td>
<td>true</td>
</tr>
</tbody>
</table>

**List of number format tokens**

<table>
<thead>
<tr>
<th>Category</th>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signs</td>
<td>Sign</td>
<td>Negative sign if the value is negative. Nothing if the value is positive or zero.</td>
</tr>
<tr>
<td></td>
<td>Sign always</td>
<td>Negative sign if the value is negative. Positive sign if the value is positive or zero.</td>
</tr>
<tr>
<td></td>
<td>Negative start</td>
<td>Open parenthesis if the value is negative. Nothing if the value is positive or zero.</td>
</tr>
<tr>
<td></td>
<td>Negative end</td>
<td>Closing parenthesis if the value is negative. Nothing if the value is positive or zero.</td>
</tr>
<tr>
<td>Category</td>
<td>Token</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Digits</td>
<td>#</td>
<td>Optional digit. Displays digit only if significant.</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Mandatory digit. Displays digit if significant, otherwise displays zero.</td>
</tr>
<tr>
<td>Separators</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Decimal separator</td>
<td>The symbol used to separate the integer and decimal parts of the number. The symbol used is determined by the locale. The decimal separator can be used only once in an expression.</td>
</tr>
<tr>
<td></td>
<td>Grouping</td>
<td>By default, digits are grouped using the rule and the separator defined by the locale. The grouping symbol can be used only once in an expression. It must appear before the decimal separator.</td>
</tr>
<tr>
<td>Exponents</td>
<td>E+</td>
<td>Exponent sign in upper case, always signed. Can be used only once in an expression.</td>
</tr>
<tr>
<td></td>
<td>E-</td>
<td>Exponent sign in upper case, signed only if the value is negative. Can be used only once in an expression.</td>
</tr>
<tr>
<td></td>
<td>e+</td>
<td>Exponent sign in lower case, always signed. Can be used only once in an expression.</td>
</tr>
<tr>
<td></td>
<td>e-</td>
<td>Exponent sign in lower case, signed only if the value is negative. Can be used only once in an expression.</td>
</tr>
<tr>
<td>Percent</td>
<td>Percent</td>
<td>The value multiplied by 100.</td>
</tr>
<tr>
<td></td>
<td>Percent %</td>
<td>The value multiplied by 100 followed by the percent sign (%). Can be used only once in an expression.</td>
</tr>
<tr>
<td>Boolean</td>
<td>Boolean</td>
<td>Localized value of true if the numerical value is not zero; localized value of false if the numerical value is zero.</td>
</tr>
<tr>
<td></td>
<td>True</td>
<td>Always displays the localized value of true.</td>
</tr>
<tr>
<td></td>
<td>False</td>
<td>Always displays the localized value of false.</td>
</tr>
</tbody>
</table>
# More Information

<table>
<thead>
<tr>
<th>Information Resource</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP product information</td>
<td><a href="http://www.sap.com">http://www.sap.com</a></td>
</tr>
<tr>
<td>Access the most up-to-date English documentation covering all SAP BusinessObjects products at the SAP Help Portal:</td>
<td></td>
</tr>
<tr>
<td>• <a href="http://help.sap.com/bobi">http://help.sap.com/bobi</a> (Business Intelligence)</td>
<td></td>
</tr>
<tr>
<td>Certain guides linked to from the SAP Help Portal are stored on the SAP Service Marketplace. Customers with a maintenance agreement have an authorized user ID to access this site. To obtain an ID, contact your customer support representative.</td>
<td></td>
</tr>
<tr>
<td>To find a comprehensive list of product documentation in all supported languages, visit: <a href="http://help.sap.com/boall">http://help.sap.com/boall</a></td>
<td></td>
</tr>
<tr>
<td>The SAP Support Portal contains information about Customer Support programs and services. It also has links to a wide range of technical information and downloads. Customers with a maintenance agreement have an authorized user ID to access this site. To obtain an ID, contact your customer support representative.</td>
<td></td>
</tr>
<tr>
<td>These articles were formerly known as technical papers.</td>
<td></td>
</tr>
<tr>
<td><strong>Information Resource</strong></td>
<td><strong>Location</strong></td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Notes</td>
<td><a href="https://service.sap.com/notes">https://service.sap.com/notes</a></td>
</tr>
<tr>
<td></td>
<td>These notes were formerly known as Knowledge Base articles.</td>
</tr>
<tr>
<td></td>
<td>From traditional classroom learning to targeted e-learning seminars, we can offer a training package to suit your learning needs and preferred learning style.</td>
</tr>
<tr>
<td></td>
<td>Consultants can accompany you from the initial analysis stage to the delivery of your deployment project. Expertise is available in topics such as relational and multidimensional databases, connectivity, database design tools, and customized embedding technology.</td>
</tr>
</tbody>
</table>
Index

A
ANSI92
universe parameter 292
AUTO_UPDATE_QUERY
universe parameter 292

B
BLOB_COMPARISON
universe parameter 294
BOUNDARY_WEIGHT_TABLE
universe parameter 294

C
CASE_SENSITIVE 308
COMMA 308
CONCAT 309
CUMULATIVE_OBJECT_WHERE
universe parameter 295

D
date-time format 318
DELIMIT_IDENTIFIERS 309
DELIMIT_LOWERCASE 309
domain names 77

E
END_SQL
universe parameter 297
EVAL_WITHOUT_PARENTHESIS
universe parameter 298
extension PRM 311
CASE_SENSITIVE 308
COMMA 308
CONCAT 309
DELIMIT_IDENTIFIERS 309
DELIMIT_LOWERCASE 309
EXTERN_SORT_EXCLUDE_DISTINCT 310
GROUPBY_WITH_ALIAS 310
OVER_CLAUSE 314
OWNER 314
QUALIFIER 314
UNICODE_PATTERN 315
USER_INPUT_DATE_FORMAT 315
USER_INPUT_NUMERIC_SEPARATOR 316

F
FORCE_SORTED_LOV
universe parameter 299
Format Editor
description 317
formats
date-time tokens 318
number tokens 322

G
GROUPBY_WITH_ALIAS 310

I
IDENTIFIER_DELIMITER 311
INNERJOIN_IN_WHERE
universe parameter 300

J
JOIN_BY_SQL
universe parameter 301

M
MAX_INLIST_VALUES
universe parameter 301

N
number format 322

O
OVER_CLAUSE 314
OWNER 314

P
PRM files
CASE_SENSITIVE 308
COMMA 308
CONCAT 309
DELIMIT_IDENTIFIERS 309
DELIMIT_LOWERCASE 309
EXTERN_SORT_EXCLUDE_DISTINCT 310
GROUPBY_WITH_ALIAS 310
IDENTIFIER_DELIMITER 311
OVER_CLAUSE 314
OWNER 314
QUALIFIER 314
UNICODE_PATTERN 315
USER_INPUT_DATE_FORMAT 315
USER_INPUT_NUMERIC_SEPARATOR 316

Q
QUALIFIER 314

R
REPLACE_COMMA_BY_CONCAT
universe parameter 302

S
SELFJOINS_IN_WHERE
universe parameter 303
SHORTCUT_BEHAVIOR 303
Show dependencies 114
SMART_AGGREGATE
universe parameter 304

T
THOROUGH_PARSE
universe parameter 306
tokens 318, 322
TRUST_CARDINALITIES
universe parameter 306

U
UNICODE_PATTERN 315
UNICODE_STRINGS
universe parameter 307
USER_INPUT_DATE_FORMAT 315