Informatica PowerCenter Express
(Version 9.6.1)

User Guide
This product includes software licensed under the Academic Free License (http://www.opensource.org/licenses/afl-3.0.php), the Common Development and Distribution License (http://www.opensource.org/licenses/cddl1.php) the Common Public License (http://www.opensource.org/licenses/cpl1.0.php), the Sun Binary Code License Agreement Supplemental License Terms, the BSD License (http://www.opensource.org/licenses/bsd-license.php), the new BSD License (http://opensource.org/licenses/BSD-3-Clause), the MIT License (http://www.opensource.org/licenses/mit-license.php), the Artistic License (http://www.opensource.org/licenses/artistic-license-1.0) and the Initial Developer’s Public License Version 1.0 (http://www.firebirdsql.org/en/initial-developer-s-public-license-version-1-0/).

This product includes software copyright © 2003-2006 Joe Walnes, 2006-2007 XStream Committers. All rights reserved. Permissions and limitations regarding this software are subject to terms available at http://xstream.codehaus.org/license.html. This product includes software developed by the Indiana University Extreme! Lab. For further information please visit http://www.extreme.indiana.edu/

This product includes software Copyright (c) 2013 Frank Balluffi and Markus Moeller. All rights reserved. Permissions and limitations regarding this software are subject to terms of the MIT license.

This Software is protected by U.S. Patent Numbers 5,794,246; 6,014,670; 6,016,501; 6,029,178; 6,032,158; 6,035,307; 6,044,374; 6,092,086; 6,208,990; 6,339,775; 6,640,226; 6,789,096; 6,823,373; 6,850,947; 6,895,471; 7,117,215; 7,162,643; 7,243,110; 7,254,590; 7,281,001; 7,421,458; 7,496,588; 7,523,121; 7,584,422; 7,675,516; 7,720,842; 7,721,270; 7,774,791; 8,065,296; 8,150,803; 8,166,048; 8,166,071; 8,200,622; 8,224,973; 8,271,477; 8,327,419; 8,386,435; 8,392,400; 8,453,159; 8,458,230; and RE44,478, International Patents and other Patents Pending.

DISCLAIMER: Informatica Corporation provides this documentation "as is" without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of noninfringement, merchantability, or use for a particular purpose. Informatica Corporation does not warrant that this software or documentation is error free. The information provided in this software or documentation may include technical inaccuracies or typographical errors. The information in this software and documentation is subject to change at any time without notice.

NOTICES
This Informatica product (the "Software") includes certain drivers (the "DataDirect Drivers") from DataDirect Technologies, an operating company of Progress Software Corporation ("DataDirect") which are subject to the following terms and conditions:

1. THE DATADIRECT DRIVERS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.

2. IN NO EVENT WILL DATADIRECT OR ITS THIRD PARTY SUPPLIERS BE LIABLE TO THE END-USER CUSTOMER FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL OR OTHER DAMAGES ARISING OUT OF THE USE OF THE ODBC DRIVERS, WHETHER OR NOT INFORMED OF THE POSSIBILITIES OF DAMAGES IN ADVANCE. THESE LIMITATIONS APPLY TO ALL CAUSES OF ACTION, INCLUDING, WITHOUT LIMITATION, BREACH OF CONTRACT, BREACH OF WARRANTY, NEGLIGENCE, STRICT LIABILITY, MISREPRESENTATION AND OTHER TORTS.

Part Number: PCX-DUG-96100-0001
Table of Contents

Preface ................................................................. viii
  Informatica Resources. ......................................... viii
  Informatica My Support Portal. ............................ viii
  Informatica Documentation. .................................... viii
  Informatica Web Site. .......................................... viii
  Informatica Support YouTube Channel. ...................... viii
  Informatica Marketplace. .................................. ix
  Informatica Velocity. ........................................ ix

Chapter 1: Introduction to PowerCenter Express................. 1
  PowerCenter Express Overview. ............................... 1
  PowerCenter Express Example. ............................... 1
  PowerCenter Express Architecture. ......................... 2
  Data Integration Process. ..................................... 3

Chapter 2: Informatica Developer........................................ 5
  Informatica Developer Overview. ............................ 5
  Start Informatica Developer. ................................ 5
    Starting the Developer Tool on a Local Machine. ........ 6
    Starting the Developer Tool on a Remote Machine. ...... 6
  Informatica Developer User Interface. ..................... 7
    Informatica Developer Welcome Page. .................. 8
    Cheat Sheets. ............................................. 8
    Informatica Preferences. ................................ 9
    Informatica Marketplace. ................................ 9
  Setting Up Informatica Developer. .......................... 9
    Adding the Domain and Repository. ..................... 10
    Selecting the Default Data Integration Service. ....... 13
  The Model Repository. ......................................... 14
    Objects in Informatica Developer. ....................... 14
    Object Properties. ....................................... 15
    Connecting to a Model Repository. ....................... 16
    Model Repository Service Refresh. ....................... 16
  Projects. ..................................................... 16
    Creating a Project. ....................................... 17
    Filter Projects. .......................................... 17
  Project Permissions. ......................................... 18
    Permissions for External Objects. ....................... 18
    Permissions for Dependent Object Instances. .......... 18
    Parent Object Access. ................................... 19
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assigning Permissions.</td>
<td>20</td>
</tr>
<tr>
<td>Folders.</td>
<td>20</td>
</tr>
<tr>
<td>Creating a Folder.</td>
<td>21</td>
</tr>
<tr>
<td>Copy.</td>
<td>21</td>
</tr>
<tr>
<td>Copying an Object.</td>
<td>21</td>
</tr>
<tr>
<td>Saving a Copy of an Object.</td>
<td>22</td>
</tr>
<tr>
<td>Tags.</td>
<td>22</td>
</tr>
<tr>
<td>Creating a Tag.</td>
<td>22</td>
</tr>
<tr>
<td>Assigning a Tag.</td>
<td>22</td>
</tr>
<tr>
<td>Viewing Tags.</td>
<td>23</td>
</tr>
<tr>
<td>Chapter 3: Searches in Informatica Developer</td>
<td>24</td>
</tr>
<tr>
<td>Searches in Informatica Developer Overview</td>
<td>24</td>
</tr>
<tr>
<td>Model Repository Search.</td>
<td>24</td>
</tr>
<tr>
<td>Searching for Objects and Properties</td>
<td>25</td>
</tr>
<tr>
<td>Editor Search.</td>
<td>25</td>
</tr>
<tr>
<td>Chapter 4: Connections</td>
<td>27</td>
</tr>
<tr>
<td>Connections Overview.</td>
<td>27</td>
</tr>
<tr>
<td>Connection Explorer View.</td>
<td>28</td>
</tr>
<tr>
<td>Connection Management.</td>
<td>29</td>
</tr>
<tr>
<td>Creating Connections.</td>
<td>29</td>
</tr>
<tr>
<td>Showing Connections.</td>
<td>30</td>
</tr>
<tr>
<td>Editing a Connection.</td>
<td>30</td>
</tr>
<tr>
<td>Copying a Connection.</td>
<td>30</td>
</tr>
<tr>
<td>Deleting a Connection.</td>
<td>31</td>
</tr>
<tr>
<td>Refreshing the Connections List.</td>
<td>31</td>
</tr>
<tr>
<td>Third-Party JDBC Drivers.</td>
<td>31</td>
</tr>
<tr>
<td>Chapter 5: Connection Properties</td>
<td>33</td>
</tr>
<tr>
<td>DataSift Connection Properties.</td>
<td>33</td>
</tr>
<tr>
<td>Facebook Connection Properties.</td>
<td>34</td>
</tr>
<tr>
<td>Greenplum Connection Properties.</td>
<td>34</td>
</tr>
<tr>
<td>IBM DB2 Connection Properties.</td>
<td>36</td>
</tr>
<tr>
<td>JDBC Connection Properties.</td>
<td>38</td>
</tr>
<tr>
<td>LinkedIn Connection Properties.</td>
<td>41</td>
</tr>
<tr>
<td>MS SQL Server Connection Properties.</td>
<td>41</td>
</tr>
<tr>
<td>ODBC Connection Properties.</td>
<td>44</td>
</tr>
<tr>
<td>Oracle Connection Properties.</td>
<td>45</td>
</tr>
<tr>
<td>Twitter Connection Properties.</td>
<td>48</td>
</tr>
<tr>
<td>Twitter Streaming Connection Properties.</td>
<td>49</td>
</tr>
<tr>
<td>Web Content-Kapow Katalyst Connection Properties.</td>
<td>50</td>
</tr>
<tr>
<td>Web Services Connection Properties.</td>
<td>51</td>
</tr>
</tbody>
</table>
# Chapter 6: Physical Data Objects

Physical Data Objects Overview .................................................. 53
Relational Data Objects .............................................................. 54
   Key Relationships .............................................................. 55
   Creating a Read Transformation from Relational Data Objects .......... 56
   Importing a Relational Data Object ....................................... 57
Customized Data Objects ........................................................... 57
   Key Relationships .............................................................. 59
   Customized Data Object Write Properties ................................ 59
   Creating a Customized Data Object ....................................... 60
   Adding Relational Resources to a Customized Data Object ............ 61
   Adding Relational Data Objects to a Customized Data Object ........ 61
   Creating Keys in a Customized Data Object ............................ 62
   Creating Relationships within a Customized Data Object .............. 62
Custom Queries ........................................................................... 62
   Custom Query Optimization .................................................. 63
   Default Query ........................................................................ 63
   Hints .................................................................................... 64
   Select Distinct ...................................................................... 66
   Filters ................................................................................... 66
   Sorted Ports .......................................................................... 67
   User-Defined Joins ............................................................... 68
   Outer Join Support .................................................................. 69
   Informatica Join Syntax ........................................................ 70
   Pre- and Post-Mapping SQL Commands ..................................... 74
   Creating a Custom Query ....................................................... 74
Flat File Data Objects .................................................................... 75
   Flat File Data Object Overview Properties ................................ 76
   Flat File Data Object Read Properties ...................................... 77
   Flat File Data Object Write Properties ...................................... 81
   Flat File Data Object Advanced Properties ............................... 85
   Creating a Flat File Data Object .............................................. 86
   Importing a Fixed-Width Flat File Data Object ......................... 87
   Importing a Delimited Flat File Data Object ............................... 88
WSDL Data Object ........................................................................ 89
   WSDL Data Object Overview View ......................................... 90
   WSDL Data Object Advanced View ......................................... 91
   Importing a WSDL Data Object .............................................. 91
   WSDL Synchronization .......................................................... 92
   Certificate Management ......................................................... 92
Synchronization ............................................................................ 93
Synchronizing a Flat File Data Object. .................................................. 94
Synchronizing a Relational Data Object. .................................................. 94
Reject Files. ......................................................................................... 94
Location of Reject Files. ....................................................................... 94
Content of Reject Files. ........................................................................ 95
Troubleshooting Physical Data Objects. .................................................. 96

Chapter 7: Profiles. .................................................................................. 98
Profiles Overview. ................................................................................. 98
Profile Views. ......................................................................................... 99
Column Profiling Process. ...................................................................... 100
Column Profile Options. ........................................................................ 100
  Rules. .................................................................................................. 101
  Filtering Options. ............................................................................... 102
  Sampling Properties. .......................................................................... 102
Creating a Profile. .................................................................................. 103
Column Profile Results. ......................................................................... 103
  Column Value Properties. .................................................................. 104
  Column Pattern Properties. ............................................................... 104
  Column Statistics Properties. ........................................................... 105
  Column Datatype Properties. ............................................................ 105
Curation in Informatica Developer. .......................................................... 106
  Approving Datatypes in Informatica Developer. .................................. 106
  Rejecting Datatypes in Informatica Developer. .................................... 106
Profile Results Export. ........................................................................... 107
Exporting Profile Results from Informatica Developer. ......................... 107
Mapplet and Mapping Profiles. ............................................................... 108
  Generating a Mapping from a Profile. ............................................... 108
  Running a Profile on a Mapplet or Mapping Object. ......................... 109
  Comparing Profiles for Mapping or Mapplet Objects. ...................... 109

Chapter 8: Logical View of Data. ............................................................. 110
Logical View of Data Overview. .............................................................. 110
  Logical Data Object Model Example. ................................................ 111
Developing a Logical View of Data. ....................................................... 111
Logical Data Object Models. ................................................................. 112
  Creating a Logical Data Object Model. ............................................. 112
  Importing a Logical Data Object Model from a Modeling Tool. .......... 113
Logical Data Object Model Properties. ................................................. 114
  CA ERwin Data Modeler Import Properties. ................................... 114
  IBM Cognos Business Intelligence Reporting - Framework Manager Import Properties. . 115
  SAP BusinessObjects Designer Import Properties. .......................... 116
  Sybase PowerDesigner CDM Import Properties. .............................. 117
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sybase PowerDesigner OOM 9.x to 15.x Import Properties</td>
<td>118</td>
</tr>
<tr>
<td>Sybase PowerDesigner PDM Import Properties</td>
<td>119</td>
</tr>
<tr>
<td>XSD Import Properties</td>
<td>120</td>
</tr>
<tr>
<td>Logical Data Objects</td>
<td>120</td>
</tr>
<tr>
<td>Logical Data Object Properties</td>
<td>121</td>
</tr>
<tr>
<td>Attribute Relationships</td>
<td>121</td>
</tr>
<tr>
<td>Creating a Logical Data Object</td>
<td>122</td>
</tr>
<tr>
<td>Logical Data Object Mappings</td>
<td>124</td>
</tr>
<tr>
<td>Logical Data Object Read Mappings</td>
<td>124</td>
</tr>
<tr>
<td>Logical Data Object Write Mappings</td>
<td>124</td>
</tr>
<tr>
<td>Creating a Logical Data Object Mapping</td>
<td>125</td>
</tr>
<tr>
<td>Chapter 9: Transformations</td>
<td>126</td>
</tr>
<tr>
<td>Transformations Overview</td>
<td>126</td>
</tr>
<tr>
<td>Active Transformations</td>
<td>126</td>
</tr>
<tr>
<td>Passive Transformations</td>
<td>127</td>
</tr>
<tr>
<td>Unconnected Transformations</td>
<td>127</td>
</tr>
<tr>
<td>Transformation Descriptions</td>
<td>127</td>
</tr>
<tr>
<td>Developing a Transformation</td>
<td>128</td>
</tr>
<tr>
<td>Reusable Transformations</td>
<td>129</td>
</tr>
<tr>
<td>Reusable Transformation Instances and Inherited Changes</td>
<td>129</td>
</tr>
<tr>
<td>Editing a Reusable Transformation</td>
<td>129</td>
</tr>
<tr>
<td>Expressions in Transformations</td>
<td>129</td>
</tr>
<tr>
<td>The Expression Editor</td>
<td>131</td>
</tr>
<tr>
<td>Port Names in an Expression</td>
<td>131</td>
</tr>
<tr>
<td>Adding an Expression to a Port</td>
<td>131</td>
</tr>
<tr>
<td>Comments in an Expression</td>
<td>132</td>
</tr>
<tr>
<td>Expression Validation</td>
<td>132</td>
</tr>
<tr>
<td>Creating a Transformation</td>
<td>132</td>
</tr>
<tr>
<td>Chapter 10: Viewing Data</td>
<td>133</td>
</tr>
<tr>
<td>Viewing Data Overview</td>
<td>133</td>
</tr>
<tr>
<td>Configurations</td>
<td>133</td>
</tr>
<tr>
<td>Configuration Properties</td>
<td>134</td>
</tr>
<tr>
<td>Data Viewer Configurations</td>
<td>137</td>
</tr>
<tr>
<td>Mapping Configurations</td>
<td>138</td>
</tr>
<tr>
<td>Updating the Default Configuration Properties</td>
<td>140</td>
</tr>
<tr>
<td>Troubleshooting Configurations</td>
<td>140</td>
</tr>
<tr>
<td>Exporting Data</td>
<td>141</td>
</tr>
<tr>
<td>Object Dependencies</td>
<td>141</td>
</tr>
<tr>
<td>View Object Dependencies</td>
<td>141</td>
</tr>
<tr>
<td>Viewing Object Dependencies</td>
<td>142</td>
</tr>
<tr>
<td>Filtering Object Dependencies</td>
<td>142</td>
</tr>
</tbody>
</table>
Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logs.</td>
<td>143</td>
</tr>
<tr>
<td>Log File Format.</td>
<td>144</td>
</tr>
<tr>
<td>Validation Preferences.</td>
<td>144</td>
</tr>
<tr>
<td>Grouping Error Messages.</td>
<td>144</td>
</tr>
<tr>
<td>Limiting Error Messages.</td>
<td>145</td>
</tr>
<tr>
<td>Monitoring Jobs from the Developer Tool.</td>
<td>145</td>
</tr>
<tr>
<td>Chapter 11: Deployment.</td>
<td>146</td>
</tr>
<tr>
<td>Deployment Overview.</td>
<td>146</td>
</tr>
<tr>
<td>Deployment Methods.</td>
<td>147</td>
</tr>
<tr>
<td>Mapping Deployment Properties.</td>
<td>147</td>
</tr>
<tr>
<td>Creating an Application.</td>
<td>149</td>
</tr>
<tr>
<td>Deploying an Object to a Data Integration Service.</td>
<td>149</td>
</tr>
<tr>
<td>Deploying an Object to a File.</td>
<td>150</td>
</tr>
<tr>
<td>Updating an Application.</td>
<td>150</td>
</tr>
<tr>
<td>Importing Application Archives.</td>
<td>150</td>
</tr>
<tr>
<td>Application Redeployment.</td>
<td>151</td>
</tr>
<tr>
<td>Redeploying an Application.</td>
<td>151</td>
</tr>
<tr>
<td>Chapter 12: Object Import and Export.</td>
<td>152</td>
</tr>
<tr>
<td>Object Import and Export Overview.</td>
<td>152</td>
</tr>
<tr>
<td>Import and Export Objects.</td>
<td>153</td>
</tr>
<tr>
<td>Object Export.</td>
<td>154</td>
</tr>
<tr>
<td>Exporting Objects.</td>
<td>154</td>
</tr>
<tr>
<td>Object Import.</td>
<td>154</td>
</tr>
<tr>
<td>Importing Projects.</td>
<td>155</td>
</tr>
<tr>
<td>Importing Objects.</td>
<td>156</td>
</tr>
<tr>
<td>Appendix A: Datatype Reference.</td>
<td>158</td>
</tr>
<tr>
<td>Datatype Reference Overview.</td>
<td>158</td>
</tr>
<tr>
<td>Transformation Datatypes.</td>
<td>159</td>
</tr>
<tr>
<td>Integer Datatypes.</td>
<td>159</td>
</tr>
<tr>
<td>Binary Datatype.</td>
<td>161</td>
</tr>
<tr>
<td>Date/Time Datatype.</td>
<td>161</td>
</tr>
<tr>
<td>Decimal and Double Datatypes.</td>
<td>162</td>
</tr>
<tr>
<td>String Datatypes.</td>
<td>163</td>
</tr>
<tr>
<td>Flat File and Transformation Datatypes.</td>
<td>163</td>
</tr>
<tr>
<td>DB2 for LUW and Transformation Datatypes.</td>
<td>164</td>
</tr>
<tr>
<td>Unsupported DB2 for LUW Datatypes.</td>
<td>165</td>
</tr>
<tr>
<td>JDBC and Transformation Datatypes.</td>
<td>165</td>
</tr>
<tr>
<td>Microsoft SQL Server and Transformation Data Types.</td>
<td>167</td>
</tr>
<tr>
<td>Uniqueidentifier Data Type.</td>
<td>168</td>
</tr>
<tr>
<td>Unsupported Microsoft SQL Server Data Types.</td>
<td>168</td>
</tr>
</tbody>
</table>
ODBC and Transformation Datatypes. .................................................. 169
Oracle and Transformation Data Types. ........................................ 170
   Number(P,S) Data Type. .......................................................... 171
   Char, Varchar, Clob Data Types ............................................. 171
   Unsupported Oracle Data Types. ............................................. 172
XML and Transformation Datatypes. ............................................. 172
Converting Data. ................................................................. 174
   Port-to-Port Data Conversion. ............................................... 174

Appendix B: Keyboard Shortcuts...................................................... 176
   Keyboard Shortcuts for Objects. ............................................. 176
   Keyboard Shortcuts for Ports. ............................................... 177
   Keyboard Shortcuts for the Transformation Palette. .................... 178
   Keyboard Shortcuts for the Workbench. ................................... 178

Index. ......................................................................................... 180
Preface

The Informatica PowerCenter Express User Guide is written for data integration developers. This guide assumes that you have an understanding of flat file and relational database concepts, the database engines in your environment, and data integration concepts.

Informatica Resources

Informatica My Support Portal


The site contains product information, user group information, newsletters, access to the Informatica How-To Library, the Informatica Knowledge Base, Informatica Product Documentation, and access to the Informatica user community.

Informatica Documentation

The Informatica Documentation team takes every effort to create accurate, usable documentation. If you have questions, comments, or ideas about this documentation, contact the Informatica Documentation team through email at infa_documentation@informatica.com. We will use your feedback to improve our documentation. Let us know if we can contact you regarding your comments.

The Documentation team updates documentation as needed. To get the latest documentation for your product, navigate to Product Documentation from http://mysupport.informatica.com.

Informatica Web Site

You can access the Informatica corporate web site at http://www.informatica.com. The site contains information about Informatica, its background, upcoming events, and sales offices. You will also find product and partner information. The services area of the site includes important information about technical support, training and education, and implementation services.

Informatica Support YouTube Channel

You can access the Informatica Support YouTube channel at http://www.youtube.com/user/INFASupport. The Informatica Support YouTube channel includes videos about solutions that guide you through performing specific tasks. If you have questions, comments, or ideas about the Informatica Support YouTube channel,
contact the Support YouTube team through email at supportvideos@informatica.com or send a tweet to @INFASupport.

Informatica Marketplace

The Informatica Marketplace is a forum where developers and partners can share solutions that augment, extend, or enhance data integration implementations. By leveraging any of the hundreds of solutions available on the Marketplace, you can improve your productivity and speed up time to implementation on your projects. You can access Informatica Marketplace at http://www.informaticamarketplace.com.

Informatica Velocity

You can access Informatica Velocity at http://mysupport.informatica.com. Developed from the real-world experience of hundreds of data management projects, Informatica Velocity represents the collective knowledge of our consultants who have worked with organizations from around the world to plan, develop, deploy, and maintain successful data management solutions. If you have questions, comments, or ideas about Informatica Velocity, contact Informatica Professional Services at ips@informatica.com.
CHAPTER 1

Introduction to PowerCenter Express

This chapter includes the following topics:

- PowerCenter Express Overview, 1
- PowerCenter Express Architecture, 2
- Data Integration Process, 3

PowerCenter Express Overview

Use PowerCenter Express to design and implement data integration solutions.

You can use PowerCenter Express to extract data from multiple sources, transform the data according to business logic that you build in the client application, and load the transformed data to targets. You can also run a profile to analyze the structure and content of your data, and to determine the quality of your data.

You can access data in relational databases, flat files, web services, and social media web sites.

PowerCenter Express includes the Informatica domain, application services that process data, repositories to store metadata, Informatica Administrator (the Administrator tool), and Informatica Developer (the Developer tool).

The Administrator tool, the repositories, and application services run on a single machine. The Developer tool can run on one or more machines.

PowerCenter Express Example

Organizations can use PowerCenter Express to perform data integration tasks. For example, use PowerCenter Express to consolidate data.

An organization wants to consolidate data from three flat files that contain customer records from different geographic areas. After analyzing the content of the data, the organization wants to load the data to a relational database table.

To complete this task, the organization uses the Developer tool to run a profile on the data, consolidate the data, and write the consolidated data to a relational database. Then, the organization uses the Administrator tool to monitor the progress of the profile job and the progress of the workflow that consolidates the data.
PowerCenter Express Architecture

The PowerCenter Express application clients, application services, and repositories are components that run within the Informatica domain. The Informatica domain is the fundamental administrative unit in Informatica.

This topic is shared among the PowerCenter Express Getting Started Guide, Data Services User Guide, and the Data Services Getting Started Guide. Also, this topic contains a conref from the Informatica Domain Components topic.

The following figure shows the PowerCenter Express components that run within the Informatica domain:

**Condition:** InfaProduct=PCX

---

**Application Clients**

A group of clients that you use to access underlying Informatica functionality. Application clients send requests to the Service Manager or application services. The Service Manager runs the application services and performs domain functions including authentication, authorization, and logging.

The Informatica domain includes the following application clients for PowerCenter Express:

**Informatica Developer**

Informatica Developer (the Developer tool) is an application client that developers can use to design and implement data integration and data federation solutions.

**Informatica Administrator**

Informatica Administrator (the Administrator tool) is an application client that consolidates the administrative tasks for domain objects such as services, connections, and licenses. Administrators manage the domain and the security of the domain through the Administrator tool.

**Application Services**

A group of services that represent server-based functionality. Administrators configure the application services that are required by the application clients.

The Informatica domain includes the following application services for PowerCenter Express:
Data Integration Service

The Data Integration Service is an application service that runs data integration jobs for the Developer tool and external clients. Data integration jobs include previewing data and running profiles, mappings, and workflows.

Model Repository Service

The Model Repository Service is an application service that manages the Model repository.

Repositories

Repositories are a group of relational databases that store metadata about objects and processes required to handle user requests from application clients.

The Informatica domain includes the following databases and directory for PowerCenter Express:

Model repository

The Model repository is a relational database that stores the metadata for projects. The Model repository also stores run-time and configuration information for applications that are deployed to a Data Integration Service.

Domain configuration repository

The domain configuration repository is a set of domain metadata tables stored in a relational database. Each time an administrator makes a change to the domain, the Service Manager writes the change to the domain configuration repository.

Profile warehouse

The profile warehouse is a relational database that the Data Integration Services uses to store profile results.

Data Integration Process

Use the PowerCenter Express client applications to complete your data integration solutions. Use the Developer tool to create connections, import metadata, run profiles, create mappings, and run mappings as part of workflows. Then, use the Administrator tool to monitor workflow progress.

Step 1. Create connections

Create connections to access data from relational databases, third-party web services, or social media web sites. Create a connection to create data objects, preview data, run profiles, and run mappings. The Developer tool uses the connection when you import a data object. The Data Integration Service uses the connection when you preview data, run profiles, and run mappings.

Step 2. Import metadata to create data objects

Import metadata to create data objects for sources and targets that you want to use in a mapping. When you develop a mapping, you can use data objects to define the input and output of the mapping.

Step 3. Run a profile

Run a profile to analyze the structure and content of your data, and to determine the quality of your data. When you run a profile, the Data Integration Service applies the profiling rules and runs the profile.

Step 4. Develop mappings

Develop mappings to implement data integration tasks. A mapping is a set of inputs and outputs that represent the data flow between sources and targets. Link the sources and targets with transformation
objects that define the rules for data transformation. The Data Integration Service uses the instructions configured in the mapping to read, transform, and write data. You can add a mapping as a task in a workflow.

**Step 5. Create and run workflows**

Create a workflow to define a sequence of events, tasks, and decisions based on a business process. Then, deploy the workflow to the Data Integration Service and run the workflow. The Data Integration Service uses the instructions configured in the workflow to run the objects.

**Step 6. Monitor workflows**

Monitor the workflow instance run on the Monitoring tab of the Administrator tool. The Monitoring tab shows the status of running workflow and workflow object instances. You can abort or cancel a running workflow instance in the Monitoring tool. You can also use the Monitoring tool to view logs for workflow instances and to view workflow reports.
CHAPTER 2

Informatica Developer

This chapter includes the following topics:

- Informatica Developer Overview, 5
- Start Informatica Developer, 5
- Informatica Developer User Interface, 7
- Setting Up Informatica Developer, 9
- The Model Repository, 14
- Projects, 16
- Project Permissions, 18
- Folders, 20
- Copy, 21
- Tags, 22

Informatica Developer Overview

The Developer tool is an application that you use to design and implement data integration solutions.

You can use the Developer tool to import metadata, create connections, and create logical data objects. You can also use the Developer tool to create and run profiles, mappings, workflows.

Start Informatica Developer

If the Developer tool is installed on a local machine, use the Windows Start menu to start the tool. If the Developer tool is installed on a remote machine, use the command line to start the tool.
Starting the Developer Tool on a Local Machine

Use the Windows Start menu to start the Developer tool installed on a local machine.

1. From the Windows Start menu, click All Programs > Informatica PowerCenter Express > Launch Informatica Developer.
   The first time you run the Developer tool, the Welcome page displays multiple icons. The Welcome page does not appear when you run the Developer tool again.

2. Click Workbench.
   If you installed Informatica services and Informatica client separately, the first time you start the Developer tool, you must set up the tool by adding a domain, adding a Model repository, and selecting a default Data Integration Service.
   If you installed Informatica services and Informatica client at the same time on the same machine, the Object Explorer view displays the Model repository with the default name ModelRepository.

Starting the Developer Tool on a Remote Machine

Use the command line to start the Developer tool installed on a remote machine.

When the Developer tool is installed on a remote machine, you might not have write access to the installation directory. You must specify a workspace directory on your local machine where the Developer tool can write temporary files. An administrator can configure the default local workspace directory for all users. You can override the default directory when you start the Developer tool.

If the configured local workspace directory does not exist, the Developer tool creates the directory when it writes temporary files.

1. Open a command prompt.

2. Enter the command to start the Developer tool. You can use the default local workspace directory or override the default directory.
   - To use the default local workspace directory, enter the following command:
     \<remote installation directory>\developer.exe
     For example:
     \MyRemoteMachine\Informatica\PCExpress\client\DeveloperClient\developer.exe
   - To override the default local workspace directory, enter the following command:
     \<remote installation directory>\developer.exe -data <local workspace directory>
     For example:
     \MyRemoteMachine\Informatica\PCExpress\client\DeveloperClient\developer.exe -data C:\temp\MyWorkspace

   Folder names in the local workspace directory cannot contain the number sign (#) character. If folder names in the local workspace directory contain spaces, enclose the full directory in double quotes.

   The first time you run the Developer tool, the Welcome page displays multiple icons. The Welcome page does not appear when you run the Developer tool again.

3. Click Workbench.
   If you installed Informatica services and Informatica client separately, the first time you start the Developer tool, you must set up the tool by adding a domain, adding a Model repository, and selecting a default Data Integration Service.
   If you installed Informatica services and Informatica client at the same time on the same machine, the Object Explorer view displays the Model repository with the default name ModelRepository.
The Developer tool is an application that you use to design and implement data integration solutions. The Developer tool workbench includes an editor and views.

You edit objects, such as mappings, in the editor. The Developer tool displays views, such as the Properties view, based on which object is selected in the editor and your selection of which views you want to display.

The following figure shows the Developer tool workbench:

1. Object Explorer view
2. Outline view
3. Properties view
4. Data Viewer view
5. Tags view
6. Object Dependencies view
7. Alerts view
8. Connection Explorer view
9. Editor

The Developer tool displays the following views by default:

**Outline view**
Displays objects that are dependent on an object selected in the Object Explorer view.

**Object Explorer view**
Displays projects, folders, and the objects within the projects and folders.

**Connection Explorer view**
Displays connections to relational databases.

**Properties view**
Displays the properties for an object that is selected in the editor.
Data Viewer view
Displays source data, profile results, and previews the output of a transformation.

Object Dependencies view
Displays object dependencies when you view, modify, or delete an object.

Tags view
Displays tags that define an object in the Model repository based on business usage.

Alerts view
Displays connection status alerts.

You can hide views and move views to another location in the Developer tool workbench. Click Window > Show View to select the views you want to display.

The Developer tool workbench also displays the following views:

Cheat Sheets view
Displays the cheat sheet that you open. To open a cheat sheet, click Help > Cheat Sheets and select a cheat sheet.

Help view
Displays context-sensitive online help.

Progress view
Displays the progress of operations in the Developer tool, such as a mapping run.

Search view
Displays the search results. You can also launch the search options dialog box.

Validation Log view
Displays object validation errors.

Informatica Developer Welcome Page

The first time you open the Developer tool, the Welcome page appears. Use the Welcome page to learn more about the Developer tool, set up the Developer tool, and to start working in the Developer tool.

The Welcome page displays the following options:

Overview
Click the Overview button to get an overview of PowerCenter Express and a cheat sheet that includes the first steps to begin using PowerCenter Express.

Tutorials
Click the Tutorials button to see cheat sheets for data integration tasks.

Web Resources
Click the Web Resources button for a links to Informatica resources that you can access on the web. Web resources include product documentation, how-to articles, and video tutorials.

Click Help > Welcome to access the welcome page after you close it.

Cheat Sheets

The Developer tool includes cheat sheets as part of the online help. A cheat sheet is a step-by-step guide that helps you complete one or more tasks in the Developer tool.
When you follow a cheat sheet, you complete the tasks and see the results. For example, you can complete a cheat sheet to import and preview a physical data object.

To access cheat sheets, click Help > Cheat Sheets.

Informatica Preferences

The Preferences dialog box contains settings for the Developer tool and for the Eclipse platform.

Use the Informatica preferences to manage settings in the Developer tool. For example, use Informatica preferences to manage configurations, connections, transformation settings, tags, or available Data Integration Services.

The Developer tool is built on the Eclipse platform. The Preferences dialog box also includes preferences to manage settings for the Eclipse platform. Informatica supports only the Informatica preferences.

To access Informatica preferences, click Window > Preferences. In the Preferences dialog box, select Informatica.

Informatica Marketplace

The Informatica Marketplace provides prebuilt solutions to augment, extend, or enhance your data integration implementation.

To access Informatica Marketplace, click Marketplace on the toolbar. The Marketplace view appears in the Developer tool.

You must register as a user before you can log in to the Marketplace for the first time.

After you log in, you can view links to prebuilt solutions in the editor. You can search for a solution in the Marketplace search box and view the search results to find the solution. A solution might contain mappings, mapping objects, profiles, or workflows that you can import into the Model repository for use in the Developer tool.

To import a Marketplace solution, click the Import button next to a Marketplace solution and follow the steps to import the solution into the Model repository. You must be connected to the Model repository to import a solution. You must select a folder during the import process to copy the related source files and documentation for the solution.

After you import the solution into the Model repository, you can run the mapping or you can edit it before you run it.

You can also post a solution to help other users in the Marketplace community.

Setting Up Informatica Developer

To set up the Developer tool, you must establish a connection to the Model repository and select the default Data Integration Service. To establish a connection to the Model repository, you must add the domain and Model repository in the Developer tool. The setup process is different based on what you installed.

If you installed the Informatica services and Informatica client separately, complete the following tasks to set up the Developer tool:

1. Add the domain and Model repository.
2. Select the default Data Integration Service.
If you installed the Informatica services and client at the same time on the same machine, you do not have to set up the Developer tool. The installation process adds the domain, Model repository, and default Data Integration Service in the Developer tool.

After you set up the Developer tool, you can create projects and folders in the Model repository to store your work.

Adding the Domain and Repository

If you installed the Informatica services and Informatica client separately, you must add the domain and Model repository the first time you set up the Developer tool.

1. From the Developer tool menu, click **File > Connect to Repository**.

   The **Connect to Repository** dialog box appears.

   ![Connect to Repository dialog box](image)

2. Click **Configure Domains**.
The *Preferences* dialog box appears.

3. Click **Add**.

   The *New Domain* dialog box appears.

4. Enter the domain name, host name, and port number for the domain.

   The following table lists the default values for the domain:

<table>
<thead>
<tr>
<th>Property</th>
<th>Default Value</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain name</td>
<td>Domain_&lt;machine name&gt;</td>
<td>Domain_SalesServer</td>
</tr>
<tr>
<td>Host name</td>
<td>&lt;machine name&gt;</td>
<td>SalesServer</td>
</tr>
<tr>
<td>Port number</td>
<td><strong>Conditioned this phrase: InfaProduct=PCX 7006</strong></td>
<td><strong>Conditioned this phrase: InfaProduct=PCX 7006</strong></td>
</tr>
</tbody>
</table>

5. Click **Test Connection** to test the connection.

   The Developer tool displays a message that states whether the connection is successful.

6. Click **OK**.

7. Click **Finish**.
The domain appears in the **Available Domains** panel of the **Preferences** dialog box.

8. Click **OK**. The **Connect to Repository** dialog box appears.

9. Click **Browse** to select the Model Repository Service associated with the Model repository. The **Choose Service** dialog box appears.

10. Expand the domain and select the Model Repository Service. The following figure shows the selected Model Repository Service:
11. Click **OK** and then click **Next**.
   The **Login** window appears.

12. Enter the user name and password provided to you.

13. Click **Finish**.
   The Developer tool connects to the Model repository. The Object Explorer view of the Developer tool shows a Model Repository Service named **MRS**. The other views are empty.

**Conditioned this phrase: InfaProduct=PCX**

**Conditioned this figure: InfaProduct=PCX**

### Selecting the Default Data Integration Service

If you installed the Informatica services and Informatica client separately, you must select the default Data Integration Service that the Developer tool uses to preview data and run mappings.

1. From the Developer tool menu, click **Window > Preferences**.
   The **Preferences** dialog box appears.

2. Select **Informatica > Data Integration Services**.

3. Expand the domain.

4. Select the Data Integration Service and click **Set as Default**.
The Model Repository

The Model repository is a relational database that stores the metadata for projects and folders.

If you installed the Informatica client only, you need to add a Model repository when you set up the Developer tool. Each time you open the Developer tool, you connect to the Model repository to access projects and folders.

Objects in Informatica Developer

You can create, manage, or view certain objects in a project or folder in the Developer tool.

You can create the following Model repository objects in the Developer tool:

**Application**

A deployable object that can contain data objects, mappings, and workflows. You can create, edit, and delete applications.

**Folder**

A container for objects in the Model repository. Use folders to organize objects in a project and create folders to group objects based on business needs. You can create, edit, and delete folders.
Logical data object
An object in a logical data object model that describes a logical entity in an enterprise. It has attributes and keys, and it describes relationships between attributes. You can create, edit, and delete logical data objects in a logical data object model.

Logical data object mapping
A mapping that links a logical data object to one or more physical data objects. It can include transformation logic. You can create, edit, and delete logical data object mappings for a logical data object.

Logical data object model
A data model that contains logical data objects and defines relationships between them. You can create, edit, and delete logical data object models.

Mapping
A set of inputs and outputs linked by transformation objects that define the rules for data transformation. You can create, edit, and delete mappings.

Mapplet
A reusable object that contains a set of transformations that you can use in multiple mappings or validate as a rule. You can create, edit, and delete mapplets.

Physical data object
A physical representation of data that is used to read from, look up, or write to resources. You can create, edit, and delete physical data objects.

Profile
An object that contains rules to discover patterns in source data. Run a profile to evaluate the data structure and verify that data columns contain the type of information that you expect. You can create, edit, and delete profiles.

Rule
Business logic that defines conditions applied to source data when you run a profile. It is a midstream mapplet that you use in a profile. You can create, edit, and delete rules.

Transformation
A repository object in a mapping that generates, modifies, or passes data. Each transformation performs a different function. A transformation can be reusable or non-reusable. You can create, edit, and delete transformations.

Workflow
A graphical representation of a set of events, tasks, and decisions that define a business process. You can create, edit, and delete workflows.

Object Properties
You can view the properties of a project, folder, or any other object in the Model repository.

The General tab in the Properties view shows the object properties. Object properties include the name, description, and location of the object in the repository. Object properties also include the user who created and last updated the object and the time the event occurred.

To access the object properties, select the object in the Object Explorer view and click File > Properties.
Connecting to a Model Repository

Each time you open the Developer tool, you connect to a Model repository to access projects and folders. When you connect to a Model repository, you enter connection information to access the domain that includes the Model Repository Service that manages the Model repository.

1. In the **Object Explorer** view, right-click a Model repository and click **Connect**.
   The **Connect to Repository** dialog box appears.
2. Enter the domain user name and password.
3. Click **OK**.
   The Developer tool connects to the Model repository. The Developer tool displays the projects in the repository.

Model Repository Service Refresh

In the Developer tool, you can refresh the Model Repository Service to see new and updated objects in the Model repository.

Refresh the Model Repository Service after other users add, delete, or update objects in the Developer tool or the Analyst tool and save the changes to the Model repository. When you refresh the Model Repository Service, the editors that you have open in the Developer tool do not close.

To refresh the Model Repository Service, right-click the Model Repository Service in the **Object Explorer** view and select **Refresh**.

Projects

A project is the top-level container that you use to store folders and objects in the Developer tool.

Use projects to organize and manage the objects that you want to use for data integration solutions.

You manage and view projects in the **Object Explorer** view. When you create a project, the Developer tool stores the project in the Model repository.

The following table describes the tasks that you can perform on a project:

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage projects</td>
<td>Manage project contents. You can create, duplicate, rename, and delete a project. You can view project contents.</td>
</tr>
<tr>
<td>Filter projects</td>
<td>Filter the list of projects that appear in the <strong>Object Explorer</strong> view.</td>
</tr>
<tr>
<td>Manage folders</td>
<td>Organize project contents in folders. You can create, duplicate, rename, and move folders within projects.</td>
</tr>
<tr>
<td>Manage objects</td>
<td>View object contents, duplicate, rename, move, and delete objects in a project or in a folder within a project.</td>
</tr>
</tbody>
</table>
### Task Description

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search projects</td>
<td>Search for folders or objects in projects. You can view search results and select an object from the results to view its contents.</td>
</tr>
<tr>
<td>Assign permissions</td>
<td>Depending on your license, you can select the users and groups that can view and edit objects in the project. Specify which users and groups can assign permissions to other users and groups.</td>
</tr>
</tbody>
</table>

### Creating a Project

Create a project to store objects and folders.

1. Select a Model Repository Service in the **Object Explorer** view.
2. Click **File > New > Project**.
   The **New Project** dialog box appears.
3. Enter a name for the project.
   If you installed PowerCenter Express Personal Edition, skip to step 6.
4. Click **Next**.
   The **Project Permissions** page of the **New Project** dialog box appears.
5. Optionally, select a user or group and assign permissions.
6. Click **Finish**.
   The project appears under the Model Repository Service in the **Object Explorer** view.

### Filter Projects

You can filter the list of projects that appear in the **Object Explorer** view. You might want to filter projects if you have access to a large number of projects but need to manage only some of them.

The Developer tool retains the list of projects that you filter the next time that you connect to the repository.

You can filter projects at the following times:

**Before you connect to the repository**

When you filter projects before you connect to the repository, you can decrease the amount of time that the Developer tool takes to connect to the repository.

Select **File > Connect to Repository**. After you select the repository and enter your user name and password, click **Next**. The **Open Project** dialog box displays all projects to which you have access. Select the projects that you want to open in the repository and then click **Finish**.

**After you connect to the repository**

If you are connected to the repository, click **File > Close Projects** to filter projects out of the **Object Explorer** view. The **Close Project** dialog box displays all projects that are currently open in the **Object Explorer** view. Select the projects that you want to filter out and then click **Finish**.

To open projects that you filtered, click **File > Open Projects**.
Project Permissions

Assign project permissions to users or groups. Project permissions determine whether a user or group can view objects, edit objects, or assign permissions to others.

Depending on the type of PowerCenter Express license, you can assign the following permissions:

**Read**
- The user or group can open, preview, export, validate, and deploy all objects in the project. The user or group can also view project details.

**Write**
- The user or group has read permission on all objects in the project. Additionally, the user or group can edit all objects in the project, edit project details, delete all objects in the project, and delete the project.

**Grant**
- The user or group has read permission on all objects in the project. Additionally, the user or group can assign permissions to other users or groups.

Users assigned the Administrator role for a Model Repository Service inherit all permissions on all projects in the Model Repository Service. Users assigned to a group inherit the group permissions.

Permissions for External Objects

Permissions apply to objects within a project. The Developer tool does not extend permissions to dependent objects when the dependent objects exist in other projects.

Dependent objects are objects that are used by other objects. For example, you create a mapplet that contains a nonreusable Expression transformation. The mapplet is the parent object. The Expression transformation is a dependent object of the mapplet.

The Developer tool creates instances of objects when you use reusable objects within a parent object. For example, you create a mapping with a reusable Lookup transformation. The mapping is the parent object. It contains an instance of the Lookup transformation.

An object can contain instances of dependent objects that exist in other projects. To view dependent object instances from other projects, you must have read permission on the other projects. To edit dependent object instances from other projects, you must have write permission on the parent object project and read permission on the other projects.

Permissions for Dependent Object Instances

You might need to access an object that contains dependent object instances from another project. If you do not have read permission on the other project, the Developer tool gives you different options based on how you access the parent object.

When you try to access a parent object that contains dependent object instances that you cannot view, the Developer tool displays a warning message. If you continue the operation, the Developer tool produces results that vary by operation type.
The following table lists the results of the operations that you can perform on the parent object:

<table>
<thead>
<tr>
<th>Operation</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open the parent object.</td>
<td>The Developer tool prompts you to determine how to open the parent object:</td>
</tr>
<tr>
<td>- Open a Copy. The Developer tool creates a copy of the parent object. The copy does not contain the dependent object instances that you cannot view.</td>
<td></td>
</tr>
<tr>
<td>- Open. The Developer tool opens the object, but it removes the dependent object instances that you cannot view. If you save the parent object, the Developer tool removes the dependent object instances from the parent object. The Developer tool does not remove the dependent objects from the repository.</td>
<td></td>
</tr>
<tr>
<td>- Cancel. The Developer tool does not open the parent object.</td>
<td></td>
</tr>
<tr>
<td>Export the parent object to an XML file for use in the Developer tool.</td>
<td>The Developer tool creates the export file without the dependent object instances.</td>
</tr>
<tr>
<td>Export the parent object to PowerCenter.</td>
<td>You cannot export the parent object.</td>
</tr>
<tr>
<td>Validate the parent object.</td>
<td>The Developer tool validates the parent object as if the dependent objects were not part of the parent object.</td>
</tr>
<tr>
<td>Deploy the parent object.</td>
<td>You cannot deploy the parent object.</td>
</tr>
<tr>
<td>Copy and paste the parent object.</td>
<td>The Developer tool creates the new object without the dependent object instances.</td>
</tr>
</tbody>
</table>

**Security Details**

When you access an object that contains dependent object instances that you cannot view, the Developer tool displays a warning message. The warning message allows you to view details about the dependent objects.

To view details about the dependent objects, click the **Details** button in the warning message. If you have the Show Security Details Model Repository Service privilege, the Developer tool lists the projects that contain the objects that you cannot view. If you do not have the Show Security Details privilege, the Developer tool indicates that you do not have sufficient privileges to view the project names.

**Parent Object Access**

If you create parent objects that use dependent object instances from other projects, users might not be able to edit the parent objects. If you want users to be able to edit the parent object and preserve the parent object functionality, you can create instances of the dependent objects in a mapplet.

For example, you create a mapping that contains a reusable Lookup transformation from another project. You want the users of your project to be able to edit the mapping, but not the Lookup transformation.

If you place the Lookup transformation in the mapping, users that do not have read permission on the other project get a warning message when they open the mapping. They can open a copy of the mapping or open the mapping, but the Developer tool removes the Lookup transformation instance.

To allow users to edit the mapping, perform the following tasks:

1. Create a mapplet in your project. Add an Input transformation, the reusable Lookup transformation, and an Output transformation to the mapplet.
2. Edit the mapping, and replace the Lookup transformation with the mapplet.
3. Save the mapping.

When users of your project open the mapping, they see the mapplet instead of the Lookup transformation. The users can edit any part of the mapping except the mapplet.

If users export the mapping, the Developer tool does not include the Lookup transformation in the export file.

Assigning Permissions

You can add users and groups to a project and assign permissions for the users and groups. Assign permissions to determine the tasks that users can complete on objects in the project.

**Note:** If you have PowerCenter Express Personal Edition, you cannot assign permissions.

1. Select a project in the Object Explorer view.
2. Click File > Properties.
   The Properties window appears.
3. Select Permissions.
4. Click Add to add a user and assign permissions for the user.
   The Domain Users and Groups dialog box appears.
5. To filter the list of users and groups, enter a name or string.
   Optionally, use the wildcard characters in the filter.
6. To filter by security domain, click the Filter by Security Domains button.
7. Select Native to show users and groups in the native security domain. Or, select All to show all users and groups.
8. Select a user or group, and click OK.
   The user or group appears in the Project Permissions page of the New Project dialog box.
9. Select read, write, or grant permission for the user or group.
10. Click OK.

Folders

Use folders to organize objects in a project. Create folders to group objects based on business needs. For example, you can create a folder to group objects for a particular task in a project. You can create a folder in a project or in another folder.

Folders appear within projects in the Object Explorer view. A folder can contain other folders, data objects, and object types.

You can perform the following tasks on a folder:
- Create a folder.
- View a folder.
- Rename a folder.
- Duplicate a folder.
- Move a folder.
- Delete a folder.
Creating a Folder

Create a folder to store related objects in a project. You must create the folder in a project or another folder.

1. **This topic is shared with the Developer User Guide and H2L article 0467.**
   In the Object Explorer view, select the project or folder where you want to create a folder.
2. Click **File > New > Folder**.
   The New Folder dialog box appears.
3. Enter a name for the folder.
4. Click **Finish**.
   The folder appears under the project or parent folder.

Copy

You can copy objects within a project or to a different project. You can also copy objects to folders in the same project or to folders in a different project.

You can copy the following objects to another project or folder or save copies of the objects with different names:

*The option to copy as a link is not in the PCX build.*

- Application
- Logical data object model
- Mapping
- Mapplet
- Physical data object
- Profile
- Reusable transformation
- Rule
- Workflow

Use the following guidelines when you copy objects:

- You can copy segments of mappings, mapplets, and rules.
- You can copy a folder to another project.
- You can paste an object multiple times after you copy it.
- If the project or folder contains an object with the same name, you can rename or replace the object.

Copying an Object

Copy an object to make it available in another project or folder.

1. Select an object in a project or folder.
2. Click **Edit > Copy**.
3. Select the project or folder that you want to copy the object to.
4. Click **Edit > Paste**.

**Saving a Copy of an Object**

Save a copy of an object to save the object with a different name.

1. Open an object in the editor.
2. Click **File > Save a Copy As**.
3. Enter a name for the copy of the object.
4. Click **Browse** to select the project or folder that you want to copy the object to.
5. Click **Finish**.

**Tags**

A tag is metadata that defines an object in the Model repository based on business usage. Create tags to group objects according to their business usage.

After you create a tag, you can associate the tag with one or more objects. You can remove the association between a tag and an object. You can use a tag to search for objects associated with the tag in the Model repository. The Developer tool displays a glossary of all tags.

For example, you create a tag named XYZCorp_CustomerOrders and assign it to tables that contain information for the customer orders from the XYZ Corporation. Users can search by the XYZCorp_CustomerOrders tag to identify the tables associated with the tag.

**Creating a Tag**

Create a tag to add metadata that defines an object based on business usage.

1. Use one of the following methods to create a tag:
   - Click **Window > Preferences**. In the **Preferences** dialog box, select **Informatica > Tags**. Select a Model Repository Service and click **Add**.
   - Open an object in the editor. In the **Tags** view, click **Edit**. In the **Assign Tags for Object** dialog box, click **New**.
2. Enter a name for the tag.
3. Optionally, enter a description.
4. Click **OK**.

**Assigning a Tag**

Assign a tag to an object to associate the object with the metadata definition.

1. Open an object in the editor.
2. In the **Tags** view, click **Edit**.

The **Assign Tags for Object** dialog box appears. The **Available Tags** area displays all tags defined in the repository. You can search for a tag by name or description. The **Assign Tags** area displays the opened object and any tags assigned to the object.
3. In the **Available Tags** area, select a tag.
4. In the **Assign Tags** area, select the object.
5. Click **Assign**.
6. To remove a tag from an object, select the tag in the **Available Tags** area and the object in the **Assign Tags** area, and then click **Remove**.

### Viewing Tags

You can view all tags assigned to an object, or you can view all tags defined in the Model repository.

1. To view tags assigned to an object, open the object in the editor.
2. Select the **Tags** view.
   
   The **Tags** view displays all tags assigned to the object.
3. To view all the tags defined in the Model repository, click **Window > Preferences**.
   
   The **Preferences** dialog box appears.
4. Select **Informatica > Tags**.
   
   The **Tags** area displays all tags defined in the Model repository. You can search for a tag by name or description.
Searches in Informatica Developer Overview

You can perform searches in Informatica Developer to search for objects.

You can search the Model repository to find the latest versions of saved objects and object properties. You can also find objects, ports, groups, expressions, and attributes in an editor.

Model Repository Search

You can search for objects and object properties in the Model repository.

You can create a search query and then filter the search results. You can view search results and select an object from the results to view its contents. Search results appear on the Search view. The search cannot display results if more than 2048 objects are found. If search fails because the results contain more than 2048 objects, change the search options so that fewer objects match the search criteria.

The following table lists the search options that you can use to search for objects:

<table>
<thead>
<tr>
<th>Search Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Containing text</td>
<td>Object or property that you want to search for. Enter an exact string or use a wildcard. Not case sensitive.</td>
</tr>
<tr>
<td>Name patterns</td>
<td>One or more objects that contain the name pattern. Enter an exact string or use a wildcard. Not case sensitive.</td>
</tr>
</tbody>
</table>
Search Option | Description
--- | ---
Tag patterns | One or more objects that use a tag. Enter an exact string or use a wildcard. Not case sensitive.
Search for | One or more object types to search for.
Scope | Search the workspace or an object that you selected.

The Model Repository Service uses a search engine to index the metadata in the Model repository. To correctly index the metadata, the search engine uses a search analyzer appropriate for the language of the metadata that you are indexing. The Developer tool uses the search engine to perform searches on objects contained in projects in the Model repository. You must save an object before you can search on it.

**Searching for Objects and Properties**

Search for objects and properties in the Model repository.

1. Click **Search > Object Search**.
   The **Search** dialog box appears.
2. Enter the object or property you want to search for. Optionally, include wildcard characters.
3. If you want to search for a property in an object, optionally enter one or more name patterns or tags separated by a comma.
4. Optionally, choose the object types that you want to search for.
5. Choose to search the workspace or the object that you selected.
6. Click **Search**.
   The search results appear in the **Search** view.
7. In the **Search** view, double-click an object to open it in the editor.

**Editor Search**

Use the editor to find objects, ports, groups, expressions, and attributes that are open in the editor. You can find objects in any mapping, mapplet, logical data object model, or workflow editor. The Developer tool highlights the objects within the open editor.

When you find objects, the Developer tool finds objects that are open in the editor. The objects do not need to be in the Model repository.

To display the find fields below the editor, select **Edit > Find/Replace**. To find an object, specify a search string and the types of objects to find. The types of objects that you can find varies by editor. If you do not specify any type of object, the Developer tool finds the search string in transformations.

When you search for ports, columns, or attributes, you can also select the datatype. For example, you can find integer or bigint ports with names that contain the string "_ID."
The following table lists the types of objects that you can find in each editor:

<table>
<thead>
<tr>
<th>Editor</th>
<th>Object types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapping</td>
<td>Mapping objects, expressions, groups, and ports</td>
</tr>
<tr>
<td>Mapplet</td>
<td>Mapplet objects, expressions, groups, and ports</td>
</tr>
<tr>
<td>Logical data object model</td>
<td>Logical data objects and attributes</td>
</tr>
<tr>
<td>Physical data object read or write mapping</td>
<td>Mapping objects and columns</td>
</tr>
<tr>
<td>Workflow</td>
<td>Workflow objects</td>
</tr>
</tbody>
</table>

When the Developer tool finds the search string, it displays the object locations. It also highlights the object in which the search string occurs. If the search string occurs in an iconized transformation in the mapping editor, the Developer tool highlights the iconized transformation.

You can select the following options to navigate the results of a find:

- Next Match. Finds the next occurrence of the search string.
- Previous Match. Finds the previous occurrence of the search string.
- Highlight All. Highlights all occurrences of the search string.
- Expand Iconized Transformations. Expands all iconized transformations in which the search string occurs.
Connections Overview

A connection is a repository object that defines a connection in the domain configuration repository.

Create a connection to import data objects, preview data, profile data, and run mappings. The Developer tool uses the connection when you import a data object. The Data Integration Service uses the connection when you preview data, run mappings, or consume web services.

The Developer tool stores connections in the domain configuration repository. Any connection that you create in the Developer tool is available in the Administrator tool.

Create and manage connections using the Create Connection and Show Connections buttons.

The following figure shows the Create Connection and the Show Connections buttons in the Developer tool:

1. Create Connection
2. Show Connections
3. Create Connection - Connection Explorer view

After you create a connection, you can perform the following actions on the connection:
Edit the connection.

You can change the connection name and the description. You can also edit connection details such as the user name, password, and connection strings.

The Data Integration Service identifies connections by the connection ID. Therefore, you can change the connection name. When you rename a connection, the Developer tool updates the objects that use the connection.

Parameter files identify a connection by name, not by connection ID. Therefore, when you rename a connection, you must also update all parameter files that use the connection parameter.

Copy the connection.

Copy a connection to create a connection that is similar to another connection. For example, you might create two Oracle connections that differ only in user name and password.

Delete the connection.

When you delete a connection, objects that use the connection are no longer valid. If you accidentally delete a connection, you can re-create it by creating another connection with the same connection ID as the deleted connection.

Refresh the connections list.

You can refresh the connections list to see the latest list of connections for the domain. Refresh the connections list after a user adds, deletes, or renames a connection in the Administrator tool.

Connection Explorer View

Use the **Connection Explorer** view to view relational or nonrelational database connections and to create relational or nonrelational data objects.

The following figure shows the **Connection Explorer** view in the Developer tool:

![Connection Explorer View](image)

You can complete the following tasks in the **Connection Explorer** view:

- Create a database connection using the **Create Connection** button.
- Add a connection to the view. Click the **Select Connection** button to choose one or more connections to add to the **Connection Explorer** view.
- Connect to a relational or nonrelational database. Right-click the database and click **Connect**.
- Disconnect from a relational or nonrelational database. Right-click the database and click **Disconnect**.
- Create a relational data object. After you connect to a relational database, expand the database to view tables. Right-click a table, and click **Add to Project** to open the **New Relational Data Object** dialog box.
• Create a nonrelational data object. After you connect to a nonrelational database, expand the database to view data maps. Right-click a data map, and click Add to Project to open the New Non-relational Data Object dialog box.

• Refresh a connection. Right-click a connection and click Refresh.

• Show only the default schema. Right-click a connection and click Show Default Schema Only. Default is enabled.

• Delete a connection from the Connection Explorer view. The connection remains in the Model repository. Right-click a connection and click Delete.

Connection Management

Create and manage connections in the Preferences dialog box or the Connection Explorer view.

Creating Connections

Create a database, social media, or web services connection. Create the connection before you import physical data objects, preview data, profile data, or run mappings. You can also add a database connection to the Connection Explorer view after you create it.

1. Click Create Connection button in the Developer tool bar.
   The New <Connection Type> Connection dialog box appears.

2. Select the type of connection that you want to create.
   • To select a database connection, select Databases.
   • To select a social media connection, select Social Media.
   • To select a web services connection, select Web Services Connections.

3. Enter the following information:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. It cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! $ % ^ &amp; * ( ) _ + = { [ ] }</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description for the connection.</td>
</tr>
<tr>
<td>Location</td>
<td>Domain in which the connection exists.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of connection.</td>
</tr>
</tbody>
</table>

4. Click Next.
5. Configure the connection properties.
6. Click **Test Connection** to verify that you entered the connection properties correctly and that you can connect.

7. Click **Finish**.

8. When you create a database connection, the **Add Connection** dialog box appears. You can choose to add the connection to the **Connection Explorer** view.

   The new connection appears in the **Connection Explorer** view.

### Showing Connections

You can see and manage available connections in the Informatica domain.

1. Click **Show Connections** in the Developer toolbar.
   
   The **Preferences** dialog box appears.

2. Select the type of connection.
   
   - To select a non-web services connection, select **Informatica > Connections**.
   
   - To select a web services connection, select **Informatica > Web Services > Connections**.

3. Expand the domain in the **Available Connections** list to see available connections.

   You can also add, edit, remove, copy, and refresh the connection list.

### Editing a Connection

You can edit the connection name, description, and connection properties.

1. Click **Window > Preferences**.

2. Select the type of connection that you want to edit.

   - To select a non-web services connection, select **Informatica > Connections**.
   
   - To select a web services connection, select **Informatica > Web Services > Connections**.

3. Expand the domain in the **Available Connections** list.

4. Select the connection in **Available Connections**, and click **Edit**.

   The **Edit Connection** dialog box appears.

5. Optionally, edit the connection name and description.

   **Note**: If you change a connection name, you must redeploy all applications that use the connection. You must also update all parameter files that use the connection parameter.

6. Click **Next**.

7. Optionally, edit the connection properties.

8. Click **Test Connection** to verify that you entered the connection properties correctly and that you can connect to the database.

9. Click **OK** to close the **Edit Connection** dialog box.

10. Click **OK** to close the **Preferences** dialog box.

### Copying a Connection

You can copy a connection within a domain or into another domain.

1. Click **Window > Preferences**.
2. Select the type of connection that you want to copy.
   • To select a non-web services connection, select Informatica > Connections.
   • To select a web services connection, select Informatica > Web Services > Connections.
3. Expand the domain in the Available Connections list.
4. Select the connection in Available Connections, and click Copy.
   The Copy Connection dialog box appears.
5. Enter the connection name and ID, and select the domain.
   The name and ID must be unique in the domain.
6. Click OK to close the Copy Connection dialog box.
7. Click OK to close the Preferences dialog box.

Deleting a Connection

When you delete a connection through the Preferences dialog box, the Developer tool removes the connection from the Model repository.

1. Click Window > Preferences.
2. Select the type of connection that you want to delete.
   • To select a non-web services connection, select Informatica > Connections.
   • To select a web services connection, select Informatica > Web Services > Connections.
3. Expand the domain in the Available Connections list.
4. Select the connection in Available Connections, and click Remove.
5. Click OK to close the Preferences dialog box.

Refreshing the Connections List

Refresh the connections list to see the latest list of connections in the domain.

1. Click Window > Preferences.
2. Select the type of connection that you want to refresh.
   • To select a non-web services connection, select Informatica > Connections.
   • To select a web services connection, select Informatica > Web Services > Connections.
3. Select the domain in the Available Connections list.
4. Click Refresh.
5. Expand the domain in the Available Connections list to see the latest list of connections.
6. Click OK to close the Preferences dialog box.

Third-Party JDBC Drivers

If you want to connect to sources and targets through JDBC, install and configure a JDBC Type 4 driver from a third-party vendor.

To import metadata in the Developer tool, copy the third-party JDBC driver jar file to the following location:
To run data previews, profiles, or mappings, copy the JDBC driver jar file to the following location:
<InformaticaInstallationDir>\externaljdbcjars

Update the CLASSPATH environment variable to include the fully qualified path to the JDBC driver.
This chapter contains connection properties for each of the connections you can create and manage through Informatica clients.

DataSift Connection Properties

Use a DataSift connection to extract data from the DataSift streams. A DataSift connection is a social media connection. You can create and manage a DataSift connection in the Administrator tool or the Developer tool.

**Note:** The order of the connection properties might vary depending on the tool where you view them.

The following table describes DataSift connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: `~ ! $ % ^ &amp; * ( ) - + = [ ]</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>Location</td>
<td>The domain where you want to create the connection.</td>
</tr>
<tr>
<td>Type</td>
<td>The connection type. Select DataSift.</td>
</tr>
<tr>
<td>Username</td>
<td>User name for the DataSift account.</td>
</tr>
<tr>
<td>API Key</td>
<td>API key. The Developer API key is displayed in the Dashboard or Settings page in the DataSift account.</td>
</tr>
</tbody>
</table>
Facebook Connection Properties

Use a Facebook connection to access data from the Facebook web site. A Facebook connection is a social media connection. You can create and manage a Facebook connection in the Administrator tool or the Developer tool.

Note: The order of the connection properties might vary depending on the tool where you view them.

The following table describes Facebook connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: `~ ! $ % ^ * ( ) = + - [ ]</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>Location</td>
<td>The domain where you want to create the connection.</td>
</tr>
<tr>
<td>Type</td>
<td>The connection type. Select Facebook.</td>
</tr>
<tr>
<td>Do you have OAuth details?</td>
<td>Indicates whether you want to configure OAuth. Select one of the following values:</td>
</tr>
<tr>
<td></td>
<td>- Yes. Indicates that you have the access token.</td>
</tr>
<tr>
<td></td>
<td>- No. Launches the OAuth Utility.</td>
</tr>
<tr>
<td>Consumer Key</td>
<td>The App ID that you get when you create the application in Facebook. Facebook uses the key to identify the application.</td>
</tr>
<tr>
<td>Consumer Secret</td>
<td>The App Secret that you get when you create the application in Facebook. Facebook uses the secret to establish ownership of the consumer key.</td>
</tr>
<tr>
<td>Access Token</td>
<td>Access token that the OAuth Utility returns. Facebook uses this token instead of the user credentials to access the protected resources.</td>
</tr>
<tr>
<td>Access Secret</td>
<td>Access secret is not required for Facebook connection.</td>
</tr>
<tr>
<td>Scope</td>
<td>Permissions for the application. Enter the permissions you used to configure OAuth.</td>
</tr>
</tbody>
</table>

Greenplum Connection Properties

Use a Greenplum connection to connect to the Greenplum database. The Greenplum connection is a relational type connection. You can create and manage a Greenplum connection in the Administrator tool or the Developer tool.

Note: The order of the connection properties might vary depending on the tool where you view them.
When you create a Greenplum connection, you enter information for metadata and data access.

The following table describes Greenplum connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the Greenplum relational connection.</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>Location</td>
<td>Domain where you want to create the connection.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of connection.</td>
</tr>
</tbody>
</table>

The user name, password, driver name, and connection string are required to import the metadata. The following table describes the properties for metadata access:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>User name with permissions to access the Greenplum database.</td>
</tr>
<tr>
<td>Password</td>
<td>Password to connect to the Greenplum database.</td>
</tr>
<tr>
<td>Driver Name</td>
<td>The name of the Greenplum JDBC driver. For example: <code>com.pivotal.jdbc.GreenplumDriver</code> For more information about the driver, see the Greenplum documentation.</td>
</tr>
<tr>
<td>Connection String</td>
<td>Use the following connection URL: <code>jdbc:pivotal:greenplum://&lt;hostname&gt;:&lt;port&gt;;DatabaseName=&lt;database_name&gt;</code> For more information about the connection URL, see the Greenplum documentation.</td>
</tr>
</tbody>
</table>

PowerExchange for Greenplum uses the host name, port number, and database name to create a control file to provide load specifications to the Greenplum gpload bulk loading utility.
The following table describes the properties for data access:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Name</td>
<td>Host name or IP address of the Greenplum server.</td>
</tr>
<tr>
<td>Port Number</td>
<td>Greenplum server port number. If you enter 0, the gpload utility reads from the environment variable $PGPORT. Default is 5432.</td>
</tr>
<tr>
<td>Database Name</td>
<td>Name of the database.</td>
</tr>
</tbody>
</table>

IBM DB2 Connection Properties

Use an IBM DB2 connection to access IBM DB2. An IBM DB2 connection is a relational database connection. You can create and manage an IBM DB2 connection in the Administrator tool, the Developer tool, or the Analyst tool.

**Note:** The order of the connection properties might vary depending on the tool where you view them.

*This topic is used in: Informatica Administrator Guide, Informatica Developer Tool Guide*

The following table describes DB2 connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Type</td>
<td>The database type.</td>
</tr>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! $ % ^ * ( ) + _ = { }</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>User Name</td>
<td>The database user name.</td>
</tr>
<tr>
<td>Password</td>
<td>The password for the database user name.</td>
</tr>
<tr>
<td>Pass-through security enabled</td>
<td>Enables pass-through security for the connection. When you enable pass-through security for a connection, the domain uses the client user name and password to log into the corresponding database, instead of the credentials defined in the connection object.</td>
</tr>
<tr>
<td>Connection String for data access</td>
<td>The DB2 connection URL used to access metadata from the database. dbname Where dbname is the alias configured in the DB2 client.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Metadata Access Properties: Connection String</strong></td>
<td>Use the following connection URL: jdbc:informatica:db2://&lt;host name&gt;:&lt;port&gt;;DatabaseName=&lt;database name&gt;</td>
</tr>
</tbody>
</table>
| **AdvancedJDBCSecurityOptions** | Database parameters for metadata access to a secure database. Informatica treats the value of the AdvancedJDBCSecurityOptions field as sensitive data and stores the parameter string encrypted. To connect to a secure database, include the following parameters:  
- EncryptionMethod. Required. Indicates whether data is encrypted when transmitted over the network. This parameter must be set to SSL.  
- ValidateServerCertificate. Optional. Indicating whether Informatica validates the certificate that is sent by the database server.  
  If this parameter is set to True, Informatica validates the certificate that is sent by the database server. If you specify the HostNameInCertificate parameter, Informatica also validates the host name in the certificate.  
  If this parameter is set to false, Informatica does not validate the certificate that is sent by the database server. Informatica ignores any truststore information that you specify.  
- HostNameInCertificate. Optional. Host name of the machine that hosts the secure database. If you specify a host name, Informatica validates the host name included in the connection string against the host name in the SSL certificate.  
- TrustStore. Required. Path and file name of the truststore file that contains the SSL certificate for the database.  
- TrustStorePassword. Required. Password for the truststore file for the secure database.  
  **Note:** Informatica appends the secure JDBC parameters to the connection string. If you include the secure JDBC parameters directly to the connection string, do not enter any parameters in the AdvancedJDBCSecurityOptions field. |

<p>| Data Access Properties: Connection String | The connection string used to access data from the database. For IBM DB2 this is &lt;database name&gt; |
| Code Page | The code page used to read from a source database or to write to a target database or file. |
| Environment SQL | SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the connection environment SQL each time it connects to the database. |
| Transaction SQL | SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the transaction environment SQL at the beginning of each transaction. |
| Retry Period | This property is reserved for future use. |
| Tablespace | The tablespace name of the database. |
| SQL Identifier Character | The type of character used to identify special characters and reserved SQL keywords, such as WHERE. The Data Integration Service places the selected character around special characters and reserved SQL keywords. The Data Integration Service also uses this character for the Support Mixed-case Identifiers property. Select the character based on the database in the connection. |</p>
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support Mixed-case Identifiers</td>
<td>When enabled, the Data Integration Service places identifier characters around table, view, schema, synonym, and column names when generating and executing SQL against these objects in the connection. Use if the objects have mixed-case or lowercase names. By default, this option is not selected.</td>
</tr>
</tbody>
</table>
| ODBC Provider                  | ODBC. The type of database to which ODBC connects. For pushdown optimization, specify the database type to enable the Data Integration Service to generate native database SQL. The options are:  
- Other  
- Sybase  
- Microsoft_SQL_Server  
  Default is Other.                                                                                                                              |

**JDBC Connection Properties**

You can use a JDBC connection to access tables in a database. You can create and manage a JDBC connection in the Administrator tool, the Developer tool, or the Analyst tool.

**Note:** The order of the connection properties might vary depending on the tool where you view them.

This topic is used in: Informatica Administrator Guide, Informatica Developer Tool Guide, Analyst Tool Guide

The following table describes JDBC connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Type</td>
<td>The database type.</td>
</tr>
</tbody>
</table>
| Name                      | Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters:  
~ ` ! $ % ^ & ( ) _ + - = [ ] { } | \ ; : " ' < , > . ? /                                                                                                                                  |
| ID                        | String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name. |
| Description               | The description of the connection. The description cannot exceed 765 characters.                                                                                                                                 |
| User Name                 | The database user name.                                                                                                                                                                                      |
| Password                  | The password for the database user name.                                                                                                                                                                    |
| JDBC Driver Class Name    | Name of the JDBC driver class.                                                                                                                                                                               |
| Connection String         | Connection string to connect to the database. Use the following connection string:  
jdbc:<subprotocol>:<subname>                                                                                                                                 |
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environment SQL</td>
<td>Optional. Enter SQL commands to set the database environment when you connect to the database. The Data Integration Service executes the connection environment SQL each time it connects to the database.</td>
</tr>
<tr>
<td>Transaction SQL</td>
<td>Optional. Enter SQL commands to set the database environment when you connect to the database. The Data Integration Service executes the transaction environment SQL at the beginning of each transaction.</td>
</tr>
<tr>
<td>SQL Identifier Character</td>
<td>The type of character used to identify special characters and reserved SQL keywords, such as WHERE. The Data Integration Service places the selected character around special characters and reserved SQL keywords. The Data Integration Service also uses this character for the Support Mixed-case Identifiers property.</td>
</tr>
<tr>
<td>Support Mixed-case Identifiers</td>
<td>When enabled, the Data Integration Service places identifier characters around table, view, schema, synonym, and column names when generating and executing SQL against these objects in the connection. Use if the objects have mixed-case or lowercase names. By default, this option is not selected.</td>
</tr>
<tr>
<td>Pass-through security enabled</td>
<td>Enables pass-through security for the connection. When you enable pass-through security for a connection, the domain uses the client user name and password to log into the corresponding database, instead of the credentials defined in the connection object.</td>
</tr>
</tbody>
</table>
| Metadata Access Properties: Connection String | The JDBC connection URL used to access metadata from the database.  
  - IBM DB2:  
    jdbc:informatica:db2://<host name>;<port>;DatabaseName=<database name>  
  - Oracle:  
    jdbc:informatica:oracle://<host_name>;<port>;SID=<database name>  
  - Microsoft SQL Server:  
    jdbc:informatica:sqlserver://<host name>;<port>;DatabaseName=<database name>  
  Not applicable for ODBC. |
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| AdvancedJDBCSecurityOptions  | Database parameters for metadata access to a secure database. Informatica treats the value of the AdvancedJDBCSecurityOptions field as sensitive data and stores the parameter string encrypted. To connect to a secure database, include the following parameters:  
- EncryptionMethod. Required. Indicates whether data is encrypted when transmitted over the network. This parameter must be set to SSL.  
- ValidateServerCertificate. Optional. Indicates whether Informatica validates the certificate that is sent by the database server. If this parameter is set to True, Informatica validates the certificate that is sent by the database server. If you specify the HostNameInCertificate parameter, Informatica also validates the host name in the certificate. If this parameter is set to false, Informatica does not validate the certificate that is sent by the database server. Informatica ignores any truststore information that you specify.  
- HostNameInCertificate. Optional. Host name of the machine that hosts the secure database. If you specify a host name, Informatica validates the host name included in the connection string against the host name in the SSL certificate.  
- TrustStore. Required. Path and file name of the truststore file that contains the SSL certificate for the database.  
- TrustStorePassword. Required. Password for the truststore file for the secure database. Not applicable for ODBC.  
**Note:** Informatica appends the secure JDBC parameters to the connection string. If you include the secure JDBC parameters directly to the connection string, do not enter any parameters in the AdvancedJDBCSecurityOptions field. |
| Code Page                    | The code page used to read from a source database or to write to a target database or file.                                                                                                                     |
| Environment SQL              | SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the connection environment SQL each time it connects to the database.                                   |
| Transaction SQL              | SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the transaction environment SQL at the beginning of each transaction.                                 |
| Retry Period                 | This property is reserved for future use.                                                                                                                                                                   |
| SQL Identifier Character     | The type of character used to identify special characters and reserved SQL keywords, such as WHERE. The Data Integration Service places the selected character around special characters and reserved SQL keywords. The Data Integration Service also uses this character for the Support Mixed-case Identifiers property. Select the character based on the database in the connection. |
| Support Mixed-case Identifiers| When enabled, the Data Integration Service places identifier characters around table, view, schema, synonym, and column names when generating and executing SQL against these objects in the connection. Use if the objects have mixed-case or lowercase names. By default, this option is not selected. |
LinkedIn Connection Properties

Use a LinkedIn connection to extract data from the LinkedIn web site. A LinkedIn connection is a social media type connection. You can create and manage a LinkedIn connection in the Administrator tool or the Developer tool.

**Note:** The order of the connection properties might vary depending on the tool where you view them.

This topic is used in: Informatica Administrator Guide, Informatica Developer Tool Guide, Analyst Tool Guide

The following table describes LinkedIn connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: <code>~</code> <code>!</code> <code>@</code> <code>#</code> <code>^</code> <code>&amp;</code> <code>*</code> <code>(</code> <code>)</code> <code>-</code> <code>+</code> <code>=</code> <code>{</code> <code>}</code> `</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>Location</td>
<td>The domain where you want to create the connection.</td>
</tr>
<tr>
<td>Type</td>
<td>The connection type. Select LinkedIn.</td>
</tr>
</tbody>
</table>
| Do you have OAuth details? | Indicates whether you want to configure OAuth. Select one of the following values:  
  - Yes. Indicates that you have the access token and secret.  
  - No. Launches the OAuth Utility. |
| Consumer Key    | The API key that you get when you create the application in LinkedIn. LinkedIn uses the key to identify the application.                     |
| Consumer Secret | The Secret key that you get when you create the application in LinkedIn. LinkedIn uses the secret to establish ownership of the consumer key. |
| Access Token    | Access token that the OAuth Utility returns. The LinkedIn application uses this token instead of the user credentials to access the protected resources. |
| Access Secret   | Access secret that the OAuth Utility returns. The secret establishes ownership of a token.                                                   |
| Scope           | Optional. Permissions for the application. Enter the permissions that you used to configure OAuth.                                          |

MS SQL Server Connection Properties

Use a Microsoft SQL Server connection to access Microsoft SQL Server. A Microsoft SQL Server connection is a connection to a Microsoft SQL Server relational database. You can create and manage a Microsoft SQL Server connection in the Administrator tool or the Developer tool.

**Note:** The order of the connection properties might vary depending on the tool where you view them.
The following table describes MS SQL Server connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Type</td>
<td>The database type.</td>
</tr>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! $ % ^ &amp; * ( ) - + = [ ]</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>Use trusted connection</td>
<td>Enables the application service to use Windows authentication to access the database. The user name that starts the application service must be a valid Windows user with access to the database. By default, this option is cleared.</td>
</tr>
<tr>
<td>User Name</td>
<td>The database user name.</td>
</tr>
<tr>
<td>Password</td>
<td>The password for the database user name.</td>
</tr>
<tr>
<td>Pass-through security enabled</td>
<td>Enables pass-through security for the connection. When you enable pass-through security for a connection, the domain uses the client user name and password to log into the corresponding database, instead of the credentials defined in the connection object.</td>
</tr>
<tr>
<td>Metadata Access Properties: Connection String</td>
<td>Use the following connection URL: jdbc:informatica:sqlserver://&lt;host name&gt;:&lt;port&gt;;DatabaseName=&lt;database name&gt;</td>
</tr>
</tbody>
</table>
## AdvancedJDBCSecurityOptions

Database parameters for metadata access to a secure database. Informatica treats the value of the `AdvancedJDBCSecurityOptions` field as sensitive data and stores the parameter string encrypted.

To connect to a secure database, include the following parameters:

- **EncryptionMethod.** Required. Indicates whether data is encrypted when transmitted over the network. This parameter must be set to SSL.
- **ValidateServerCertificate.** Optional. Indicates whether Informatica validates the certificate that is sent by the database server.
  
  If this parameter is set to True, Informatica validates the certificate that is sent by the database server. If you specify the `HostNameInCertificate` parameter, Informatica also validates the host name in the certificate.
  
  If this parameter is set to false, Informatica does not validate the certificate that is sent by the database server. Informatica ignores any truststore information that you specify.
- **HostNameInCertificate.** Optional. Host name of the machine that hosts the secure database. If you specify a host name, Informatica validates the host name included in the connection string against the host name in the SSL certificate.
- **TrustStore.** Required. Path and file name of the truststore file that contains the SSL certificate for the database.
- **TrustStorePassword.** Required. Password for the truststore file for the secure database.

Not applicable for ODBC.

**Note:** Informatica appends the secure JDBC parameters to the connection string. If you include the secure JDBC parameters directly to the connection string, do not enter any parameters in the `AdvancedJDBCSecurityOptions` field.

### Data Access Properties: Connection String

Use the following connection string:

```
<server name>@<database name>
```

If the database does not use the default port, use the following connection strings:

```
<server name>:<port>@<dbname>
<servername>/<instancetype>:<port>@<dbname>
```

### Code Page

The code page used to read from a source database or to write to a target database or file.

### Domain Name

The name of the domain.

### Packet Size

The packet size used to transmit data. Used to optimize the native drivers for Microsoft SQL Server.

### Owner Name

The name of the owner of the schema.

### Schema Name

The name of the schema in the database. You must specify the schema name for the Profiling Warehouse and staging database if the schema name is different from the database user name. You must specify the schema name for the data object cache database if the schema name is different from the database user name and you manage the cache with an external tool.

### Environment SQL

SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the connection environment SQL each time it connects to the database.
Property | Description
--- | ---
Transaction SQL | SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the transaction environment SQL at the beginning of each transaction.
Retry Period | This property is reserved for future use.
SQL Identifier Character | The type of character used to identify special characters and reserved SQL keywords, such as WHERE. The Data Integration Service places the selected character around special characters and reserved SQL keywords. The Data Integration Service also uses this character for the Support Mixed-case Identifiers property. Select the character based on the database in the connection.
Support Mixed-case Identifiers | When enabled, the Data Integration Service places identifier characters around table, view, schema, synonym, and column names when generating and executing SQL against these objects in the connection. Use if the objects have mixed-case or lowercase names. By default, this option is not selected.
ODBC Provider | ODBC. The type of database to which ODBC connects. For pushdown optimization, specify the database type to enable the Data Integration Service to generate native database SQL. The options are:
- Other
- Sybase
- Microsoft_SQL_Server
Default is Other.

**ODBC Connection Properties**

Use an ODBC connection to access ODBC data. An ODBC connection is a relational database connection. You can create and manage an ODBC connection in the Administrator tool, the Developer tool, or the Analyst tool.

**Note:** The order of the connection properties might vary depending on the tool where you view them.

This topic is used in: Informatica Administrator Guide, Informatica Developer Tool Guide, Analyst Tool Guide

The following table describes ODBC connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Type</td>
<td>The database type.</td>
</tr>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: <code>~ </code> ! $ % ^ &amp; * ( ) - + = \ [ ] \ : ; &quot; ' , &gt; . ? /`</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>User Name</td>
<td>The database user name.</td>
</tr>
<tr>
<td>Password</td>
<td>The password for the database user name.</td>
</tr>
<tr>
<td>Pass-through security enabled</td>
<td>Enables pass-through security for the connection. When you enable pass-through security for a connection, the domain uses the client user name and password to log into the corresponding database, instead of the credentials defined in the connection object.</td>
</tr>
<tr>
<td>Data Access Properties: Connection String</td>
<td>The ODBC connection URL used to access metadata from the database. &lt;data source name&gt;</td>
</tr>
<tr>
<td>Code Page</td>
<td>The code page used to read from a source database or to write to a target database or file.</td>
</tr>
<tr>
<td>Environment SQL</td>
<td>SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the connection environment SQL each time it connects to the database.</td>
</tr>
<tr>
<td>Transaction SQL</td>
<td>SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the transaction environment SQL at the beginning of each transaction.</td>
</tr>
<tr>
<td>Retry Period</td>
<td>This property is reserved for future use.</td>
</tr>
<tr>
<td>SQL Identifier Character</td>
<td>The type of character used to identify special characters and reserved SQL keywords, such as WHERE. The Data Integration Service places the selected character around special characters and reserved SQL keywords. The Data Integration Service also uses this character for the Support Mixed-case Identifiers property. Select the character based on the database in the connection.</td>
</tr>
<tr>
<td>Support Mixed-case Identifiers</td>
<td>When enabled, the Data Integration Service places identifier characters around table, view, schema, synonym, and column names when generating and executing SQL against these objects in the connection. Use if the objects have mixed-case or lowercase names. By default, this option is not selected.</td>
</tr>
<tr>
<td>ODBC Provider</td>
<td>The type of database to which ODBC connects. For pushdown optimization, specify the database type to enable the Data Integration Service to generate native database SQL. The options are: Other, Sybase, Microsoft_SQL_Server. Default is Other.</td>
</tr>
</tbody>
</table>

**Oracle Connection Properties**

Use an Oracle connection to connect to an Oracle database. The Oracle connection is a relational connection type. You can create and manage an Oracle connection in the Administrator tool, the Developer tool, or the Analyst tool.
**Note:** The order of the connection properties might vary depending on the tool where you view them.

*This topic is used in: Informatica Administrator Guide, Informatica Developer Tool Guide, Analyst Tool Guide*

The following table describes Oracle connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Database Type</strong></td>
<td>The database type.</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! % ^ * ( ) - + = { }</td>
</tr>
<tr>
<td><strong>ID</strong></td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td><strong>User Name</strong></td>
<td>The database user name.</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>The password for the database user name.</td>
</tr>
<tr>
<td><strong>Pass-through security enabled</strong></td>
<td>Enables pass-through security for the connection. When you enable pass-through security for a connection, the domain uses the client user name and password to log into the corresponding database, instead of the credentials defined in the connection object.</td>
</tr>
<tr>
<td><strong>Metadata Access Properties: Connection String</strong></td>
<td>Use the following connection URL: jdbc:informatica:oracle://&lt;host_name&gt;:&lt;port&gt;;SID=&lt;database name&gt;</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AdvancedJDBCSecurityOptions</td>
<td>Database parameters for metadata access to a secure database. Informatica treats the value of the AdvancedJDBCSecurityOptions field as sensitive data and stores the parameter string encrypted.</td>
</tr>
<tr>
<td></td>
<td>To connect to a secure database, include the following parameters:</td>
</tr>
<tr>
<td></td>
<td>- EncryptionMethod. Required. Indicates whether data is encrypted when transmitted over the network. This parameter must be set to SSL.</td>
</tr>
<tr>
<td></td>
<td>- ValidateServerCertificate. Optional. Indicates whether Informatica validates the certificate that is sent by the database server. If this parameter is set to True, Informatica validates the certificate that is sent by the database server. If you specify the HostNameInCertificate parameter, Informatica also validates the host name in the certificate. If this parameter is set to false, Informatica does not validate the certificate that is sent by the database server. Informatica ignores any truststore information that you specify.</td>
</tr>
<tr>
<td></td>
<td>- HostNameInCertificate. Optional. Host name of the machine that hosts the secure database. If you specify a host name, Informatica validates the host name included in the connection string against the host name in the SSL certificate.</td>
</tr>
<tr>
<td></td>
<td>- TrustStore. Required. Path and file name of the truststore file that contains the SSL certificate for the database.</td>
</tr>
<tr>
<td></td>
<td>- TrustStorePassword. Required. Password for the truststore file for the secure database.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Informatica appends the secure JDBC parameters to the connection string. If you include the secure JDBC parameters directly to the connection string, do not enter any parameters in the AdvancedJDBCSecurityOptions field.</td>
</tr>
</tbody>
</table>

**Data Access Properties:**

<table>
<thead>
<tr>
<th>Connection String</th>
<th>Use the following connection string: &lt;database name&gt;.world</th>
</tr>
</thead>
</table>

**Code Page**

The code page used to read from a source database or to write to a target database or file.

**Environment SQL**

SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the connection environment SQL each time it connects to the database.

**Transaction SQL**

SQL commands to set the database environment when you connect to the database. The Data Integration Service runs the transaction environment SQL at the beginning of each transaction.

**Retry Period**

This property is reserved for future use.

**Enable Parallel Mode**

Enables parallel processing when loading data into a table in bulk mode. By default, this option is cleared.
Twitter Connection Properties

Use a Twitter connection to extract data from the Twitter web site. The Twitter connection is a connection to social media. You can create and manage a Twitter connection in the Administrator tool or the Developer tool.

**Note:** The order of the connection properties might vary depending on the tool where you view them.

**This topic is used in:** Informatica Administrator Guide, Informatica Developer Tool Guide, INFA PowerExchange for Twitter User Guide

The following table describes Twitter connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: `~ ! $ % ^ &amp; * ( ) - + = [ ]</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>Location</td>
<td>The domain where you want to create the connection.</td>
</tr>
<tr>
<td>Type</td>
<td>The connection type. Select Twitter.</td>
</tr>
</tbody>
</table>
| Do you have OAuth details?      | Indicates whether you want to configure OAuth. Select one of the following values:  
  - Yes. Indicates that you have the access token and secret.  
  - No. Launches the OAuth Utility.                               |
| Consumer Key                    | The consumer key that you get when you create the application in Twitter. Twitter uses the key to identify the application. |
| Consumer Secret                 | The consumer secret that you get when you create the Twitter application. Twitter uses the secret to establish ownership of the consumer key. |
**Property Description**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Token</td>
<td>Access token that the OAuth Utility returns. Twitter uses this token instead of the user credentials to access the protected resources.</td>
</tr>
<tr>
<td>Access Secret</td>
<td>Access secret that the OAuth Utility returns. The secret establishes ownership of a token.</td>
</tr>
</tbody>
</table>

**Twitter Streaming Connection Properties**

Use a Twitter Streaming connection to access near-real time data from the Twitter web site. The Twitter Streaming connection is a connection to the social media company’s streaming API. You can create and manage a Twitter Streaming connection in the Administrator tool or the Developer tool.

**Note:** The order of the connection properties might vary depending on the tool where you view them.

This topic is used in: Informatica Administrator Guide, Informatica Developer Tool Guide

The following table describes the general properties for a Twitter Streaming connection:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: <code>~</code> ! $ % ^ &amp; * ( ) - + = { [ ] }</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>Location</td>
<td>The domain where you want to create the connection.</td>
</tr>
<tr>
<td>Type</td>
<td>The connection type. Select Twitter Streaming.</td>
</tr>
</tbody>
</table>

The following table describes the properties for hose type and OAuth authentication:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| Hose Type | Streaming API methods. You can specify one of the following methods:  
- Filter. The Twitter `statuses/filter` method returns public statuses that match the search criteria.  
- Sample. The Twitter `statuses/sample` method returns a random sample of all public statuses. |
| Consumer Key | The consumer key that you get when you create the application in Twitter. Twitter uses the key to identify the application. |
## Web Content-Kapow Katalyst Connection Properties

Use a Web Content-Kapow Katalyst connection to access robots in Kapow Katalyst. This is a social media type connection. You can create and manage a Web Content-Kapow Katalyst connection in the Administrator tool or the Developer tool.

**Note:** The order of the connection properties might vary depending on the tool where you view them.


The following table describes Web Content-Kapow Katalyst connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! $ % ^ &amp; * ( ) - _ = { }</td>
</tr>
<tr>
<td>ID</td>
<td>String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the connection. The description cannot exceed 765 characters.</td>
</tr>
<tr>
<td>Location</td>
<td>The Informatica domain where you want to create the connection.</td>
</tr>
<tr>
<td>Type</td>
<td>The connection type. Select Web Content-Kapow Katalyst.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Management Console URL</td>
<td>URL of the Management Console where the robot is uploaded. The URL must start with http or https. For example, <a href="http://localhost:50080">http://localhost:50080</a>.</td>
</tr>
<tr>
<td>RQL Service Port</td>
<td>The port number where the socket service listens for the RQL service. Enter a value from 1 through 65535. Default is 50000.</td>
</tr>
<tr>
<td>Username</td>
<td>User name required to access the Local Management Console.</td>
</tr>
<tr>
<td>Password</td>
<td>Password to access the Local Management Console.</td>
</tr>
</tbody>
</table>

**Web Services Connection Properties**

Use a web services connection to connect a Web Service Consumer transformation to a web service.

The following table describes the web services connection properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username</td>
<td>User name to connect to the web service. Enter a user name if you enable HTTP authentication or WS-Security. If the Web Service Consumer transformation includes WS-Security ports, the transformation receives a dynamic user name through an input port. The Data Integration Service overrides the user name defined in the connection.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the user name. Enter a password if you enable HTTP authentication or WS-Security. If the Web Service Consumer transformation includes WS-Security ports, the transformation receives a dynamic password through an input port. The Data Integration Service overrides the password defined in the connection.</td>
</tr>
<tr>
<td>End Point URL</td>
<td>URL for the web service that you want to access. The Data Integration Service overrides the URL defined in the WSDL file. If the Web Service Consumer transformation includes an endpoint URL port, the transformation dynamically receives the URL through an input port. The Data Integration Service overrides the URL defined in the connection.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds that the Data Integration Service waits for a response from the web service provider before it closes the connection.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| HTTP Authentication Type                     | Type of user authentication over HTTP. Select one of the following values:  
- None. No authentication.  
- Automatic. The Data Integration Service chooses the authentication type of the web service provider.  
- Basic. Requires you to provide a user name and password for the domain of the web service provider. The Data Integration Service sends the user name and the password to the web service provider for authentication.  
- Digest. Requires you to provide a user name and password for the domain of the web service provider. The Data Integration Service generates an encrypted message digest from the user name and password and sends it to the web service provider. The provider generates a temporary value for the user name and password and stores it in the Active Directory on the Domain Controller. It compares the value with the message digest. If they match, the web service provider authenticates you.  
- NTLM. Requires you to provide a domain name, server name, or default user name and password. The web service provider authenticates you based on the domain you are connected to. It gets the user name and password from the Windows Domain Controller and compares it with the user name and password that you provide. If they match, the web service provider authenticates you. NTLM authentication does not store encrypted passwords in the Active Directory on the Domain Controller. |
| WS Security Type                             | Type of WS-Security that you want to use. Select one of the following values:  
- None. The Data Integration Service does not add a web service security header to the generated SOAP request.  
- PasswordText. The Data Integration Service adds a web service security header to the generated SOAP request. The password is stored in the clear text format.  
- PasswordDigest. The Data Integration Service adds a web service security header to the generated SOAP request. The password is stored in a digest form which provides effective protection against replay attacks over the network. The Data Integration Service combines the password with a nonce and a time stamp. The Data Integration Service applies a SHA hash on the password, encodes it in base64 encoding, and uses the encoded password in the SOAP header. |
| Trust Certificates File                      | File containing the bundle of trusted certificates that the Data Integration Service uses when authenticating the SSL certificate of the web service. Enter the file name and full directory path.  
**Default** is `<Informatica installation directory>/services/shared/bin/ca-bundle.crt`.                        |
| Client Certificate File Name                | Client certificate that a web service uses when authenticating a client. Specify the client certificate file if the web service needs to authenticate the Data Integration Service. |
| Client Certificate Password                 | Password for the client certificate. Specify the client certificate password if the web service needs to authenticate the Data Integration Service. |
| Client Certificate Type                     | Format of the client certificate file. Select one of the following values:  
- PEM. Files with the .pem extension.  
- DER. Files with the .cer or .der extension.  
Specify the client certificate type if the web service needs to authenticate the Data Integration Service. |
| Private Key File Name                        | Private key file for the client certificate. Specify the private key file if the web service needs to authenticate the Data Integration Service. |
| Private Key Password                         | Password for the private key of the client certificate. Specify the private key password if the web service needs to authenticate the Data Integration Service. |
| Private Key Type                             | Type of the private key. PEM is the supported type. |
CHAPTER 6

Physical Data Objects

This chapter includes the following topics:

- Physical Data Objects Overview, 53
- Relational Data Objects, 54
- Customized Data Objects, 57
- Custom Queries, 62
- Flat File Data Objects, 75
- WSDL Data Object, 89
- Synchronization, 93
- Reject Files, 94
- Troubleshooting Physical Data Objects, 96

Physical Data Objects Overview

A physical data object is the physical representation of data that is used to read from, look up, or write to resources.

A physical data object can be one of the following types:

Relational data object

A physical data object that uses a relational table, view, or synonym as a source. For example, you can create a relational data object from an Oracle view.

Depending on the object type, you can add a relational data object to a mapping or mapplet as a source, a target, or a Lookup transformation.

Customized data object

A physical data object that uses one or multiple related relational resources or relational data objects as sources. Relational resources include tables, views, and synonyms. For example, you can create a customized data object from two Microsoft SQL Server tables that have a primary key-foreign key relationship.

Create a customized data object if you want to perform operations such as joining data, filtering rows, sorting ports, or running custom queries in a reusable data object.

Flat file data object

A physical data object that uses a flat file as a source. You can create a flat file data object from a delimited or fixed-width flat file.
**WSDL data object**

A physical data object that uses a WSDL file as a source.

If the data object source changes, you can synchronize the physical data object. When you synchronize a physical data object, the Developer tool reimports the object metadata.

You can create any physical data object in a project or folder. Physical data objects in projects and folders are reusable objects. You can use them in any type of mapping, mapplet, or profile, but you cannot change the data object within the mapping, mapplet, or profile. To update the physical data object, you must edit the object within the project or folder.

You can include a physical data object in a mapping, mapplet, or profile. You can add a physical data object to a mapping or mapplet as a read, write, or lookup transformation. You can add a physical data object to a logical data object mapping to map logical data objects.

---

**Relational Data Objects**

Import a relational data object to include in a mapping, mapplet, or profile. A relational data object is a physical data object that uses a relational table, view, or synonym as a source.

The following figure shows a sample relational data object that is open in the editor:

![Relational Data Object Editor](image)

You can create primary key-foreign key relationships between relational data objects. You can create key relationships between relational data objects whether or not the relationships exist in the source database.

You can include relational data objects in mappings and mapplets. You can add a relational data object to a mapping or mapplet as a read, write, or lookup transformation. You can add multiple relational data objects to a mapping or mapplet as sources. When you add multiple relational data objects at the same time, the Developer tool prompts you to add the objects in either of the following ways:

- As related data objects. The Developer tool creates one read transformation. The read transformation has the same capabilities as a customized data object.
As independent data objects. The Developer tool creates one read transformation for each relational data object. The read transformations have the same capabilities as relational data objects.

You can import the following types of relational data objects:

- IBM DB2
- JDBC
- Microsoft SQL Server
- ODBC
- Oracle

**Key Relationships**

You can create key relationships between relational data objects. Key relationships allow you to join relational data objects when you use them as sources in a customized data object or as read transformations in a mapping or mapplet.

When you import relational data objects, the Developer tool retains the primary key information defined in the database. When you import related relational data objects at the same time, the Developer tool also retains foreign keys and key relationships. However, if you import related relational data objects separately, you must re-create the key relationships after you import the objects.

To create key relationships between relational data objects, first create a primary key in the referenced object. Then create the relationship in the relational data object that contains the foreign key.

The key relationships that you create exist in the relational data object metadata. You do not need to alter the source relational resources.

**Creating Keys in a Relational Data Object**

Create key columns to identify each row in a relational data object. You can create one primary key in each relational data object.

1. Open the relational data object.
2. Select the **Keys** view.
   
   The following figure shows the **Keys** view for a sample relational data object that is open in the editor:

3. Click **Add**.
   
   The **New Key** dialog box appears.
4. Enter a key name.
5. If the key is a primary key, select **Primary Key**.
6. Select the key columns.
7. Click **OK**.
8. Save the relational data object.

### Creating Relationships between Relational Data Objects

You can create key relationships between relational data objects. You cannot create key relationships between a relational data object and a customized data object.

The relational data object that you reference must have a primary key.

1. Open the relational data object where you want to create a foreign key.
2. Select the **Relationships** view.
3. Click **Add**.
   
   The **New Relationship** dialog box appears.
4. Enter a name for the foreign key.
5. Select a primary key from the referenced relational data object.
6. Click **OK**.
7. In the **Relationships** properties, select the foreign key columns.
8. Save the relational data object.

### Creating a Read Transformation from Relational Data Objects

You can add a relational data object to a mapping or mapplet as a read transformation. When you add multiple relational data objects at the same time, you can add them as related or independent objects.

1. Open the mapping or mapplet in which you want to create a read transformation.
2. In the **Object Explorer** view, select one or more relational data objects.
3. Drag the relational data objects into the mapping editor.
   
   The **Add to Mapping** dialog box appears.
4. Select the **Read** option.
5. If you add multiple data objects, select one of the following options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>As related data objects</td>
<td>The Developer tool creates one read transformation. The read transformation has the same capabilities as a customized data object.</td>
</tr>
<tr>
<td>As independent data objects</td>
<td>The Developer tool creates one read transformation for each relational data object. Each read transformation has the same capabilities as a relational data object.</td>
</tr>
</tbody>
</table>

6. If the relational data objects use different connections, select the default connection.
7. Click **OK**.
   
   The Developer tool creates one or multiple read transformations in the mapping or mapplet.
Importing a Relational Data Object

Import a relational data object to add to a mapping, mapplet, or profile.

Before you import a relational data object, you must configure a connection to the database.

1. Select a project or folder in the Object Explorer view.
2. Click File > New > Data Object.
   The New dialog box appears.
3. Select Relational Data Object and click Next.
   The New Relational Data Object dialog box appears.
4. Click Browse next to the Connection option and select a connection to the database.
5. Click Create data object from existing resource.
6. Click Browse next to the Resource option.
   The Select a Resource dialog box appears.
7. Select the table, view, or synonym that you want to import.
8. To filter data objects, enter a name in the Filter section and click Search.
   Enclose the name in double quotes (") to search for case-sensitive object names.
9. Enter a name for the physical data object.
10. Click Browse next to the Location option and select the project where you want to import the relational data object.
11. Click Finish.
   The data object appears under Physical Data Objects in the project or folder in the Object Explorer view.

Customized Data Objects

Customized data objects are reusable physical data objects with one or more relational resources. Create a customized data object if you want to perform operations such as joining data, filtering rows, sorting ports, or running custom queries when the Data Integration Service reads source data. You can reuse a customized data object in a mapping, mapplet, or profile.

You can create customized data objects in projects and folders. You cannot change the customized data object from within a mapping, mapplet, or profile. If you change a customized data object in a project or folder, the Developer tool updates the object in all mappings, mapplets, and profiles that use the object.
The following figure shows a sample customized data object that is open in the editor:

Create a customized data object to perform the following tasks:

- Create a custom query to replace the default query that the Data Integration Service runs to read the source data. The default query is a SELECT statement that references each column that the Data Integration Service reads from the source.
- Define parameters for the data object. You can define and assign parameters in a customized data object to represent connections. When you run a mapping that uses the customized data object, you can define different values for the connection parameters at runtime.
- Join source data that originates from the same source database. You can join multiple tables with primary key-foreign key relationships whether or not the relationships exist in the database.
- Retain key relationships when you synchronize the object with the sources. If you create a customized data object that contains multiple tables, and you define key relationships that do not exist in the database, you can retain the relationships when you synchronize the data object.
- Select distinct values from the source. If you choose Select Distinct, the Data Integration Service adds a SELECT DISTINCT statement to the default SQL query.
- Filter rows when the Data Integration Service reads source data. If you include a filter condition, the Data Integration Service adds a WHERE clause to the default query.
- Specify sorted ports. If you specify a number for sorted ports, the Data Integration Service adds an ORDER BY clause to the default SQL query.
- Specify an outer join instead of the default inner join. If you include a user-defined join, the Data Integration Service replaces the join information specified by the metadata in the SQL query.
- Add pre- and post-mapping SQL commands. The Data Integration Service runs pre-mapping SQL commands against the source database before it reads the source. It runs post-mapping SQL commands against the source database after it writes to the target.

You can create customized data objects from the following types of connections and objects:

- IBM DB2 connections
You can also add sources to a customized data object through a custom SQL query.

**Key Relationships**

You can create key relationships between sources in a customized data object when the sources are relational resources. Key relationships allow you to join the sources within the customized data object.

**Note:** If a customized data object uses relational data objects as sources, you cannot create key relationships within the customized data object. You must create key relationships between the relational data objects instead.

When you import relational resources into a customized data object, the Developer tool retains the primary key information defined in the database. When you import related relational resources into a customized data object at the same time, the Developer tool also retains key relationship information. However, if you import related relational resources separately, you must re-create the key relationships after you import the objects into the customized data object.

When key relationships exist between sources in a customized data object, the Data Integration Service joins the sources based on the related keys in each source. The default join is an inner equijoin that uses the following syntax in the WHERE clause:

```
Source1.column_name = Source2.column_name
```

You can override the default join by entering a user-defined join or by creating a custom query.

To create key relationships in a customized data object, first create a primary key in the referenced source transformation. Then create the relationship in the source transformation that contains the foreign key.

The key relationships that you create exist in the customized data object metadata. You do not need to alter the source relational resources.

**Customized Data Object Write Properties**

The Data Integration Service uses write properties when it writes data to relational resources. To edit write properties, select the Input transformation in the **Write** view, and then select the **Advanced** properties.
The following table describes the write properties that you configure for customized data objects:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load type</td>
<td>Type of target loading. Select Normal or Bulk.</td>
</tr>
<tr>
<td></td>
<td>If you select Normal, the Data Integration Service loads targets normally. You can choose Bulk when you load to DB2, Sybase, Oracle, or Microsoft SQL Server. If you specify Bulk for other database types, the Data Integration Service reverts to a normal load. Bulk loading can increase mapping performance, but it limits the ability to recover because no database logging occurs.</td>
</tr>
<tr>
<td></td>
<td>Choose Normal mode if the mapping contains an Update Strategy transformation. If you choose Normal and the Microsoft SQL Server target name includes spaces, configure the following environment SQL in the connection object:</td>
</tr>
<tr>
<td></td>
<td>SET QUOTED_IDENTIFIER ON</td>
</tr>
<tr>
<td>Update override</td>
<td>Overrides the default UPDATE statement for the target.</td>
</tr>
<tr>
<td>Delete</td>
<td>Deletes all rows flagged for delete.</td>
</tr>
<tr>
<td></td>
<td>Default is enabled.</td>
</tr>
<tr>
<td>Insert</td>
<td>Inserts all rows flagged for insert.</td>
</tr>
<tr>
<td></td>
<td>Default is enabled.</td>
</tr>
<tr>
<td>Truncate target table</td>
<td>Truncates the target before it loads data.</td>
</tr>
<tr>
<td></td>
<td>Default is disabled.</td>
</tr>
<tr>
<td>Update strategy</td>
<td>Update strategy for existing rows. You can select one of the following strategies:</td>
</tr>
<tr>
<td></td>
<td>- Update as update. The Data Integration Service updates all rows flagged for update.</td>
</tr>
<tr>
<td></td>
<td>- Update as insert. The Data Integration Service inserts all rows flagged for update. You must also select the Insert target option.</td>
</tr>
<tr>
<td></td>
<td>- Update else insert. The Data Integration Service updates rows flagged for update if they exist in the target and then inserts any remaining rows marked for insert. You must also select the Insert target option.</td>
</tr>
<tr>
<td>PreSQL</td>
<td>SQL command the Data Integration Service runs against the target database before it reads the source. The Developer tool does not validate the SQL.</td>
</tr>
<tr>
<td>PostSQL</td>
<td>SQL command the Data Integration Service runs against the target database after it writes to the target. The Developer tool does not validate the SQL.</td>
</tr>
</tbody>
</table>

Creating a Customized Data Object

Create a customized data object to add to a mapping, mapplet, or profile. After you create a customized data object, add sources to it.

1. Select a project or folder in the Object Explorer view.
2. Click File > New > Data Object.
   The New dialog box appears.
3. Select Relational Data Object and click Next.
   The New Relational Data Object dialog box appears.
4. Click Browse next to the Connection option and select a connection to the database.
5. Click Create customized data object.
6. Enter a name for the customized data object.

7. Click **Browse** next to the Location option and select the project where you want to create the customized data object.

8. Click **Finish**.

   The customized data object appears under Physical Data Objects in the project or folder in the **Object Explorer** view.

Add sources to the customized data object. You can add relational resources or relational data objects as sources. You can also use a custom SQL query to add sources.

**Adding Relational Resources to a Customized Data Object**

After you create a customized data object, add sources to it. You can use relational resources as sources.

Before you add relational resources to a customized data object, you must configure a connection to the database.

1. In the **Connection Explorer** view, select one or more relational resources in the same relational connection.

2. Right-click in the **Connection Explorer** view and select **Add to project**.

   The **Add to Project** dialog box appears.

3. Select **Add as related resource(s) to existing customized data object** and click **OK**.

   The **Add to Data Object** dialog box appears.

4. Select the customized data object and click **OK**.

5. If you add multiple resources to the customized data object, the Developer tool prompts you to select the resource to write to. Select the resource and click **OK**.

   If you use the customized data object in a mapping as a write transformation, the Developer tool writes data to this resource.

   The Developer tool adds the resources to the customized data object.

**Adding Relational Data Objects to a Customized Data Object**

After you create a customized data object, add sources to it. You can use relational data objects as sources.

1. Open the customized data object.

2. Select the **Read** view.

3. In the **Object Explorer** view, select one or more relational data objects in the same relational connection.

4. Drag the objects from the **Object Explorer** view to the customized data object **Read** view.

5. If you add multiple relational data objects to the customized data object, the Developer tool prompts you to select the object to write to. Select the object and click **OK**.

   If you use the customized data object in a mapping as a write transformation, the Developer tool writes data to this relational data object.

   The Developer tool adds the relational data objects to the customized data object.
Creating Keys in a Customized Data Object

Create key columns to identify each row in a source transformation. You can create one primary key in each source transformation.

1. Open the customized data object.
2. Select the Read view.
3. Select the source transformation where you want to create a key.
   The source must be a relational resource, not a relational data object. If the source is a relational data object, you must create keys in the relational data object.
4. Select the Keys properties.
5. Click Add.
   The New Key dialog box appears.
6. Enter a key name.
7. If the key is a primary key, select Primary Key.
8. Select the key columns.
9. Click OK.
10. Save the customized data object.

Creating Relationships within a Customized Data Object

You can create key relationships between sources in a customized data object.

The source transformation that you reference must have a primary key.

1. Open the customized data object.
2. Select the Read view.
3. Select the source transformation where you want to create a foreign key.
   The source must be a relational resource, not a relational data object. If the source is a relational data object, you must create relationships in the relational data object.
4. Select the Relationships properties.
5. Click Add.
   The New Relationship dialog box appears.
6. Enter a name for the foreign key.
7. Select a primary key from the referenced source transformation.
8. Click OK.
9. In the Relationships properties, select the foreign key columns.
10. Save the customized data object.

Custom Queries

A custom SQL query is a SELECT statement that overrides the default SQL query in a customized data object or relational data object.
A custom query overrides the default SQL query that the Data Integration Service uses to read data from a relational source. The custom query also overrides the simple query settings that you define when you enter a source filter, use sorted ports, enter a user-defined join, or select distinct ports.

You can create a custom query to perform SQL operations that are valid in the database language but not available in the transformation language. The Data Integration Service can push a custom query to a relational data object. When you define the query in a relational data object, you must define it for an instance of the relational data object that is in a mapping, mapplet, or profile. When you define a custom query in a customized data object, you can reuse the object in multiple mappings or profiles.

Use the following guidelines when you create a custom query in a customized data object or relational data object:

- In the SELECT statement, list the column names in the order in which they appear in the source transformation.
- Enclose all database reserved words in quotes.
- Add an escape character before a dollar sign ($). If the $ has preceding forward slash (/) characters, add an escape character (\) to both the forward slash and dollar sign characters, for example, enter $ as \$ and \$ as \\$.

If you use a customized data object to perform a self-join, you must enter a custom SQL query that includes the self-join. You can use a customized data object with a custom query as a read transformation in a mapping. The source database executes the query before it passes data to the Data Integration Service. You can create a custom query to add sources to an empty customized data object. You can also use a custom query to override the default SQL query.

**Custom Query Optimization**

The Data Integration Service can push a custom query to run in a relational data object for increased performance. Choose to push a custom query if the query forms a valid subquery for the database.

When you use a custom query to read data in a relational data object, the Data Integration Service can optimize the query by pushing the query to run in the database. The Data Integration Service can push the custom query if the query forms a valid subquery for the database. If the SQL syntax for the custom query is not valid in a subquery for the database, the resulting query fails to run.

If you push a custom query to a relational database other than IBM DB2, you must specify an alias for each expression in the select list that is not a column reference. The aliases allow the Data Integration Service to refer to the expressions in the select list.

See the database documentation for information about valid SQL syntax for aliases and subqueries.

**Default Query**

The Data Integration Service generates a default SQL query that it uses to read data from relational sources. You can override the default query in a customized data object or an instance of a relational data object.

You can override the default query through the simple or advanced query. Use the simple query to select distinct values, enter a source filter, sort ports, or enter a user-defined join. Use the advanced query to create a custom SQL query for reading data from the sources. The custom query overrides the default and simple queries.

If any table name or column name contains a database reserved word, you can create and maintain a reserved words file, reswords.txt. Create the reswords.txt file on any machine the Data Integration Service can access.
When the Data Integration Service runs a mapping, it searches for the reswords.txt file. If the file exists, the Data Integration Service places quotes around matching reserved words when it executes SQL against the database. If you override the default query, you must enclose any database reserved words in quotes.

When the Data Integration Service generates the default query, it delimits table and field names containing the following characters with double quotes:

/ + - = \ ! % ^ & * ( ) [ ] { } ' ; , < > \ | <space>

Creating a Reserved Words File

Create a reserved words file if any table name or column name in the customized data object contains a database reserved word.

You must have administrator privileges to configure the Data Integration Service to use the reserved words file.

1. Create a file called "reswords.txt."
2. Create a section for each database by entering the database name within square brackets, for example, [Oracle].
3. Add the reserved words to the file below the database name.

For example:

```plaintext
[Oracle]
OPT
START
where
number

[SQL Server]
CURRENT
where
number
```

Entries are not case sensitive.

4. Save the reswords.txt file.
5. In Informatica Administrator, select the Data Integration Service.
6. Edit the custom properties.
7. Add the following custom property:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved Words File</td>
<td>&lt;path&gt;reswords.txt</td>
</tr>
</tbody>
</table>

8. Restart the Data Integration Service.

Hints

You can add hints to the source SQL query to pass instructions to a database optimizer. The optimizer uses the hints to choose a query run plan to access the source.

This topic is shared by the Source Optimization chapter of Mercury Performance Tuning and the Physical Data Objects chapter of Developer User Guide.

The Hints field appears in the Query view of a relational data object instance or a customized data object. The source database must be Oracle, Sybase, IBM DB2, or Microsoft SQL Server. The Hints field does not appear for other database types.
When the Data Integration Service generates the source query, it adds the SQL hints to the query exactly as you enter them in the Developer tool. The Data Integration Service does not parse the hints. When you run the mapping that contains the source, the mapping log shows the query with the hints in the query.

The Data Integration Service inserts the SQL hints in a position in the query depending on the database type. Refer to your database documentation for information about the syntax for hints.

**Oracle**

The Data Integration Service add hints directly after the SELECT/UPDATE/INSERT/DELETE keyword.

```
SELECT /+ <hints> */ FROM ...
```

The '+' indicates the start of hints.

The hints are contained in a comment (/* ... */ or --... until end of line)

**Sybase**

The Data Integration Service adds hints after the query. Configure a plan name in the hint.

```
SELECT ... PLAN <plan>
```

```
select avg(price) from titles plan "(scalar_agg (i_scan type_price_ix titles )"
```

**IBM DB2**

You can enter the optimize-for clause as a hint. The Data Integration Service adds the clause at the end of the query.

```
SELECT ... OPTIMIZE FOR <n> ROWS
```

The optimize-for clause tells the database optimizer how many rows the query might process. The clause does not limit the number of rows. If the database processes more than <n> rows, then performance might decrease.

**Microsoft SQL Server**

The Data Integration Service adds hints at the end of the query as part of an OPTION clause.

```
SELECT ... OPTION ( <query_hints> )
```

**Hints Rules and Guidelines**

This topic is shared by the Developer User Guide Physical Data Objects and the Performance Tuning Guide Source Optimization chapter.

Use the following rules and guidelines when you configure hints for SQL queries:

- If you enable pushdown optimization or if you use a semi-join in a relational data object, then the original source query changes. The Data Integration Service does not apply hints to the modified query.
- You can combine hints with join and filter overrides, but if you configure a SQL override, the SQL override takes precedence and the Data Integration Service does not apply the other overrides.
- The Query view shows a simple view or an advanced view. If you enter a hint with a filter, sort, or join override on the simple view, and you the Developer tool shows the full query override on the advanced view.

**Creating Hints**

Create hints to send instructions to the database optimizer to determine a query plan.
This topic is shared by the Developer User Guide Physical Data Objects and the Performance Tuning Guide Source Optimization chapter.

1. Open the customized data object or the relational data object instance.
2. Select the Read view.
3. Select the Output transformation.
4. Select the Query properties.
5. Select the simple query.
6. Click Edit next to the Hints field.
   The Hints dialog box appears.
7. Enter the hint in the SQL Query field.
   The Developer tool does not validate the hint.
8. Click OK.
9. Save the data object.

Select Distinct

You can select unique values from sources in a customized data object or a relational data object instance with the select distinct option. When you enable select distinct, the Data Integration Service adds a SELECT DISTINCT statement to the default SQL query.

Use the select distinct option to filter source data. For example, you might use the select distinct option to extract unique customer IDs from a table that lists total sales. When you use the relational data object in a mapping, the Data Integration Service filters data earlier in the data flow, which can increase performance.

Using Select Distinct

Select unique values from a relational source with the Select Distinct property.

1. Open the customized data object or relational data object instance.
2. Select the Read view.
3. Select the Output transformation.
4. Select the Query properties.
5. Select the simple query.
6. Enable the Select Distinct option.
7. Save the customized data object.

Filters

You can enter a filter value in a custom query. The filter becomes the WHERE clause in the query SELECT statement. Use a filter to reduce the number of rows that the Data Integration Service reads from the source table.

Entering a Source Filter

Enter a source filter to reduce the number of rows the Data Integration Service reads from the relational source.
1. Open the customized data object or the relational data object instance.
2. Select the Read view.
3. Select the Output transformation.
4. Select the Query properties.
5. Select the simple query.
6. Click Edit next to the Filter field.
   The SQL Query dialog box appears.
7. Enter the filter condition in the SQL Query field.
   You can select column names from the Columns list.
8. Click OK.
9. Click Validate to validate the filter condition.
10. Save the data object.

Sorted Ports

You can sort rows in the default query for a customized data object or a relational data object instance. Select the ports to sort by. The Data Integration Service adds the ports to the ORDER BY clause in the default query.

You might sort the source rows to increase performance when you include the following transformations in a mapping:

- Aggregator. When you configure an Aggregator transformation for sorted input, you can send sorted data by using sorted ports. The group by ports in the Aggregator transformation must match the order of the sorted ports in the customized data object.
- Joiner. When you configure a Joiner transformation for sorted input, you can send sorted data by using sorted ports. Configure the order of the sorted ports the same in each customized data object.

Note: You can also use the Sorter transformation to sort relational and flat file data before Aggregator and Joiner transformations.

Sorting Column Data

Use sorted ports to sort column data in a customized data object or relational data object instance. When you use the data object as a read transformation in a mapping or mapplet, you can pass the sorted data to transformations downstream from the read transformation.

1. Open the customized data object or relational data object instance.
2. Select the Read view.
3. Select the Output transformation.
4. Select the Query properties.
5. Select the simple query.
6. Click Edit next to the Sort field.
   The Sort dialog box appears.
7. To specify a column as a sorted port, click the New button.
8. Select the column and sort type, either ascending or descending.
9. Repeat steps 7 and 8 to select other columns to sort.
   The Developer tool sorts the columns in the order in which they appear in the Sort dialog box.
10. Click OK.
    In the Query properties, the Developer tool displays the sort columns in the Sort field.
11. Click Validate to validate the sort syntax.
12. Save the data object.

User-Defined Joins

You can configure a user-defined join in a customized data object or relational data object instance. A user-defined join defines the condition to join data from multiple sources in the same data object.

When you add a user-defined join to a customized data object or a relational data object instance, you can use the data object as a read transformation in a mapping. The source database performs the join before it passes data to the Data Integration Service. Mapping performance increases when the source tables are indexed.

Create a user-defined join to join data from related sources. The user-defined join overrides the default inner join that the Data Integration creates based on the related keys in each source. When you enter a user-defined join, enter the contents of the WHERE clause that specifies the join condition. If the user-defined join performs an outer join, the Data Integration Service might insert the join syntax in the WHERE clause or the FROM clause, based on the database syntax.

You might need to enter a user-defined join in the following circumstances:

- Columns do not have a primary key-foreign key relationship.
- The datatypes of columns used for the join do not match.
- You want to specify a different type of join, such as an outer join.
Use the following guidelines when you enter a user-defined join in a customized data object or relational data object instance:

- Do not include the WHERE keyword in the user-defined join.
- Enclose all database reserved words in quotes.
- If you use Informatica join syntax, and Enable quotes in SQL is enabled for the connection, you must enter quotes around the table names and the column names if you enter them manually. If you select tables and columns when you enter the user-defined join, the Developer tool places quotes around the table names and the column names.

User-defined joins join data from related resources in a database. To join heterogeneous sources, use a Joiner transformation in a mapping that reads data from the sources. To perform a self-join, you must enter a custom SQL query that includes the self-join.

**Entering a User-Defined Join**

Configure a user-defined join in a customized data object or relational data object to define the join condition for the data object sources.

1. Open the customized data object or relational data object instance.
2. Select the Read view.
3. Select the Output transformation.
4. Select the Query properties.
5. Select the simple query.
6. Click Edit next to the Join field.
   The SQL Query dialog box appears.
7. Enter the user-defined join in the SQL Query field.
   You can select column names from the Columns list.
8. Click OK.
9. Click Validate to validate the user-defined join.
10. Save the data object.

**Outer Join Support**

You can use a customized data object to perform an outer join of two sources in the same database. When the Data Integration Service performs an outer join, it returns all rows from one source resource and rows from the second source resource that match the join condition.

Use an outer join when you want to join two resources and return all rows from one of the resources. For example, you might perform an outer join when you want to join a table of registered customers with a monthly purchases table to determine registered customer activity. You can join the registered customer table with the monthly purchases table and return all rows in the registered customer table, including customers who did not make purchases in the last month. If you perform a normal join, the Data Integration Service returns only registered customers who made purchases during the month, and only purchases made by registered customers.

With an outer join, you can generate the same results as a master outer or detail outer join in the Joiner transformation. However, when you use an outer join, you reduce the number of rows in the data flow which can increase performance.
You can enter two kinds of outer joins:

- Left. The Data Integration Service returns all rows for the resource to the left of the join syntax and the rows from both resources that meet the join condition.
- Right. The Data Integration Service returns all rows for the resource to the right of the join syntax and the rows from both resources that meet the join condition.

**Note:** Use outer joins in nested query statements when you override the default query.

You can enter an outer join in a user-defined join or in a custom SQL query.

### Informatica Join Syntax

When you enter join syntax, use the Informatica or database-specific join syntax. When you use the Informatica join syntax, the Data Integration Service translates the syntax and passes it to the source database during a mapping run.

**Note:** Always use database-specific syntax for join conditions.

When you use Informatica join syntax, enclose the entire join statement in braces ({{Informatica syntax}}).

When you use database syntax, enter syntax supported by the source database without braces.

When you use Informatica join syntax, use table names to prefix column names. For example, if you have a column named FIRST_NAME in the REG_CUSTOMER table, enter "REG_CUSTOMER.FIRST_NAME" in the join syntax. Also, when you use an alias for a table name, use the alias within the Informatica join syntax to ensure the Data Integration Service recognizes the alias.

You can combine left outer or right outer joins with normal joins in a single data object. You cannot combine left and right outer joins. Use multiple normal joins and multiple left outer joins. Some databases limit you to using one right outer join.

When you combine joins, enter the normal joins first.

### Normal Join Syntax

You can create a normal join using the join condition in a customized data object or relational data object instance.

When you create an outer join, you must override the default join. As a result, you must include the normal join in the join override. When you include a normal join in the join override, list the normal join before outer joins. You can enter multiple normal joins in the join override.

To create a normal join, use the following syntax:

```
{ source1 INNER JOIN source2 on join_condition }
```

The following table displays the syntax for normal joins in a join override:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>source1</code></td>
<td>Source resource name. The Data Integration Service returns rows from this resource that match the join condition.</td>
</tr>
<tr>
<td><code>source2</code></td>
<td>Source resource name. The Data Integration Service returns rows from this resource that match the join condition.</td>
</tr>
<tr>
<td><code>join_condition</code></td>
<td>Condition for the join. Use syntax supported by the source database. You can combine multiple join conditions with the AND operator.</td>
</tr>
</tbody>
</table>
For example, you have a REG_CUSTOMER table with data for registered customers:

<table>
<thead>
<tr>
<th>CUST_ID</th>
<th>FIRST_NAME</th>
<th>LAST_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>Marvin</td>
<td>Chi</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
</tr>
<tr>
<td>00003</td>
<td>John</td>
<td>Bowden</td>
</tr>
<tr>
<td>00004</td>
<td>J.</td>
<td>Marks</td>
</tr>
</tbody>
</table>

The PURCHASES table, refreshed monthly, contains the following data:

<table>
<thead>
<tr>
<th>TRANSACTION_NO</th>
<th>CUST_ID</th>
<th>DATE</th>
<th>AMOUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>06-2000-0001</td>
<td>00002</td>
<td>6/3/2000</td>
<td>55.79</td>
</tr>
<tr>
<td>06-2000-0002</td>
<td>00002</td>
<td>6/10/2000</td>
<td>104.45</td>
</tr>
<tr>
<td>06-2000-0003</td>
<td>00001</td>
<td>6/10/2000</td>
<td>255.56</td>
</tr>
<tr>
<td>06-2000-0004</td>
<td>00004</td>
<td>6/15/2000</td>
<td>534.95</td>
</tr>
<tr>
<td>06-2000-0005</td>
<td>00002</td>
<td>6/21/2000</td>
<td>98.65</td>
</tr>
<tr>
<td>06-2000-0007</td>
<td>NULL</td>
<td>6/24/2000</td>
<td>325.45</td>
</tr>
</tbody>
</table>

To return rows displaying customer names for each transaction in the month of June, use the following syntax:

```sql
{ REG_CUSTOMER INNER JOIN PURCHASES on REG_CUSTOMER.CUST_ID = PURCHASES.CUST_ID }
```

The Data Integration Service returns the following data:

<table>
<thead>
<tr>
<th>CUST_ID</th>
<th>DATE</th>
<th>AMOUNT</th>
<th>FIRST_NAME</th>
<th>LAST_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>00002</td>
<td>6/3/2000</td>
<td>55.79</td>
<td>Dinah</td>
<td>Jones</td>
</tr>
<tr>
<td>00002</td>
<td>6/10/2000</td>
<td>104.45</td>
<td>Dinah</td>
<td>Jones</td>
</tr>
<tr>
<td>00001</td>
<td>6/10/2000</td>
<td>255.56</td>
<td>Marvin</td>
<td>Chi</td>
</tr>
<tr>
<td>00004</td>
<td>6/15/2000</td>
<td>534.95</td>
<td>J.</td>
<td>Marks</td>
</tr>
<tr>
<td>00002</td>
<td>6/21/2000</td>
<td>98.65</td>
<td>Dinah</td>
<td>Jones</td>
</tr>
</tbody>
</table>

The Data Integration Service returns rows with matching customer IDs. It does not include customers who made no purchases in June. It also does not include purchases made by non-registered customers.

**Left Outer Join Syntax**

You can create a left outer join with a join override. You can enter multiple left outer joins in a single join override. When using left outer joins with other joins, list all left outer joins together, after any normal joins in the statement.
The Data Integration Service returns the following data:

To create a left outer join, use the following syntax:

```
( source1 LEFT OUTER JOIN source2 on join_condition )
```

The following tables displays syntax for left outer joins in a join override:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source1</td>
<td>Source resource name. With a left outer join, the Data Integration Service returns all rows in this resource.</td>
</tr>
<tr>
<td>source2</td>
<td>Source resource name. The Data Integration Service returns rows from this resource that match the join condition.</td>
</tr>
<tr>
<td>join_condition</td>
<td>Condition for the join. Use syntax supported by the source database. You can combine multiple join conditions with the AND operator.</td>
</tr>
</tbody>
</table>

For example, using the same REG_CUSTOMER and PURCHASES tables described in “Normal Join Syntax” on page 70, you can determine how many customers bought something in June with the following join override:

```
( REG_CUSTOMER LEFT OUTER JOIN PURCHASES on REG_CUSTOMER.CUST_ID = PURCHASES.CUST_ID )
```

The Data Integration Service returns the following data:

<table>
<thead>
<tr>
<th>CUST_ID</th>
<th>FIRST_NAME</th>
<th>LAST_NAME</th>
<th>DATE</th>
<th>AMOUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>Marvin</td>
<td>Chi</td>
<td>6/10/2000</td>
<td>255.56</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
<td>6/3/2000</td>
<td>55.79</td>
</tr>
<tr>
<td>00003</td>
<td>John</td>
<td>Bowden</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>00004</td>
<td>J.</td>
<td>Marks</td>
<td>6/15/2000</td>
<td>534.95</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
<td>6/10/2000</td>
<td>104.45</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
<td>6/21/2000</td>
<td>98.65</td>
</tr>
</tbody>
</table>

The Data Integration Service returns all registered customers in the REG_CUSTOMERS table, using null values for the customer who made no purchases in June. It does not include purchases made by non-registered customers.

Use multiple join conditions to determine how many registered customers spent more than $100.00 in a single purchase in June:

```
( REG_CUSTOMER LEFT OUTER JOIN PURCHASES on (REG_CUSTOMER.CUST_ID = PURCHASES.CUST_ID AND PURCHASES.AMOUNT > 100.00) )
```

The Data Integration Service returns the following data:

<table>
<thead>
<tr>
<th>CUST_ID</th>
<th>FIRST_NAME</th>
<th>LAST_NAME</th>
<th>DATE</th>
<th>AMOUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>Marvin</td>
<td>Chi</td>
<td>6/10/2000</td>
<td>255.56</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
<td>6/10/2000</td>
<td>104.45</td>
</tr>
<tr>
<td>00003</td>
<td>John</td>
<td>Bowden</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>00004</td>
<td>J.</td>
<td>Marks</td>
<td>6/15/2000</td>
<td>534.95</td>
</tr>
</tbody>
</table>
You might use multiple left outer joins if you want to incorporate information about returns during the same time period. For example, the RETURNS table contains the following data:

<table>
<thead>
<tr>
<th>CUST_ID</th>
<th>CUST_ID</th>
<th>RETURN</th>
</tr>
</thead>
<tbody>
<tr>
<td>00002</td>
<td>6/10/2000</td>
<td>55.79</td>
</tr>
<tr>
<td>00002</td>
<td>6/21/2000</td>
<td>104.45</td>
</tr>
</tbody>
</table>

To determine how many customers made purchases and returns for the month of June, use two left outer joins:

```sql
{ REG_CUSTOMER LEFT OUTER JOIN PURCHASES on REG_CUSTOMER.CUST_ID = PURCHASES.CUST_ID
  LEFT OUTER JOIN RETURNS on REG_CUSTOMER.CUST_ID = PURCHASES.CUST_ID }
```

The Data Integration Service returns the following data:

<table>
<thead>
<tr>
<th>CUST_ID</th>
<th>FIRST_NAME</th>
<th>LAST_NAME</th>
<th>DATE</th>
<th>AMOUNT</th>
<th>RET_DATE</th>
<th>RETURN</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>Marvin</td>
<td>Chi</td>
<td>6/10/2000</td>
<td>255.56</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
<td>6/3/2000</td>
<td>55.79</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>00003</td>
<td>John</td>
<td>Bowden</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>00004</td>
<td>J.</td>
<td>Marks</td>
<td>6/15/2000</td>
<td>534.95</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
<td>6/10/2000</td>
<td>104.45</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
<td>6/21/2000</td>
<td>98.65</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
<td>NULL</td>
<td>NULL</td>
<td>6/10/2000</td>
<td>55.79</td>
</tr>
<tr>
<td>00002</td>
<td>Dinah</td>
<td>Jones</td>
<td>NULL</td>
<td>NULL</td>
<td>6/21/2000</td>
<td>104.45</td>
</tr>
</tbody>
</table>

The Data Integration Service uses NULLs for missing values.

**Right Outer Join Syntax**

You can create a right outer join with a join override. The right outer join returns the same results as a left outer join if you reverse the order of the resources in the join syntax. Use only one right outer join in a join override. If you want to create more than one right outer join, try reversing the order of the source resources and changing the join types to left outer joins.

When you use a right outer join with other joins, enter the right outer join at the end of the join override.

To create a right outer join, use the following syntax:

```sql
{ source1 RIGHT OUTER JOIN source2 on join_condition }
```
The following table displays syntax for a right outer join in a join override:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source1</td>
<td>Source resource name. The Data Integration Service returns rows from this resource that match the join condition.</td>
</tr>
<tr>
<td>source2</td>
<td>Source resource name. With a right outer join, the Data Integration Service returns all rows in this resource.</td>
</tr>
<tr>
<td>join_condition</td>
<td>Condition for the join. Use syntax supported by the source database. You can combine multiple join conditions with the AND operator.</td>
</tr>
</tbody>
</table>

**Pre- and Post-Mapping SQL Commands**

You can create SQL commands in a customized data object or relational data object instance. The Data Integration Service runs the SQL commands against the source relational resource.

When you run the mapping, the Data Integration Service runs pre-mapping SQL commands against the source database before it reads the source. It runs post-mapping SQL commands against the source database after it writes to the target.

Use the following guidelines when you configure pre- and post-mapping SQL commands:

- Use any command that is valid for the database type. The Data Integration Service does not allow nested comments, even though the database might allow them.
- Use a semicolon (;) to separate multiple statements. The Data Integration Service issues a commit after each statement.
- The Data Integration Service ignores semicolons within /* ... */.
- If you need to use a semicolon outside comments, you can escape it with a backslash (\). When you escape the semicolon, the Data Integration Service ignores the backslash, and it does not use the semicolon as a statement separator.
- The Developer tool does not validate the SQL in a pre- and post-mapping SQL commands.

**Adding Pre- and Post-Mapping SQL Commands**

You can add pre- and post-mapping SQL commands to a customized data object or relational data object instance. The Data Integration Service runs the SQL commands when you use the data object in a mapping.

1. Open the customized data object.
2. Select the Read view.
3. Select the Output transformation
4. Select the Advanced properties.
5. Enter a pre-mapping SQL command in the PreSQL field.
6. Enter a post-mapping SQL command in the PostSQL field.
7. Save the customized data object.

**Creating a Custom Query**

Create a custom query to issue a special SELECT statement for reading data from the sources. The custom query overrides the default query that the Data Integration Service issues to read source data.
1. Open the customized data object or the relational data object instance.
2. Select the Read view.
3. Select the Output transformation.
4. Select the Query properties.
5. Select the advanced query.
6. Select Use custom query.
   The Data Integration Service displays the query it issues to read source data.
7. Change the query or replace it with a custom query.
8. If you want to push the custom query to the relational data source, select Push custom query to database.
   The Data Integration Service does not the push custom query to the database by default. Choose to push the custom query if the query forms a valid subquery for the database.
9. Save the data object.

**Flat File Data Objects**

Create or import a flat file data object to include in a mapping, mapplet, or profile. You can use flat file data objects as sources, targets, and lookups in mappings and mapplets. You can create profiles on flat file data objects.

A flat file physical data object can be delimited or fixed-width. You can import fixed-width and delimited flat files that do not contain binary data.

After you import a flat file data object, you might need to create parameters or configure file properties. Create parameters through the Parameters view. Edit file properties through the Overview, Read, Write, and Advanced views.

The Overview view allows you to edit the flat file data object name and description. It also allows you to update column properties for the flat file data object.

The Read view controls the properties that the Data Integration Service uses when it reads data from the flat file. The Read view contains the following transformations:

- Source transformation. Defines the flat file that provides the source data. Select the source transformation to edit properties such as the name and description, column properties, and source file format properties.
- Output transformation. Represents the rows that the Data Integration Service reads when it runs a mapping. Select the Output transformation to edit the file run-time properties such as the source file name and directory.

The Write view controls the properties that the Data Integration Service uses when it writes data to the flat file. The Write view contains the following transformations:

- Input transformation. Represents the rows that the Data Integration Service writes when it runs a mapping. Select the Input transformation to edit the file run-time properties such as the target file name and directory.
- Target transformation. Defines the flat file that accepts the target data. Select the target transformation to edit the name and description and the target file format properties.

The Advanced view controls format properties that the Data Integration Service uses when it reads data from and writes data to the flat file.
When you create mappings that use file sources or file targets, you can view flat file properties in the Properties view. You cannot edit file properties within a mapping, except for the reject file name, reject file directory, and tracing level.

Flat File Data Object Overview Properties

The Data Integration Service uses overview properties when it reads data from or writes data to a flat file. Overview properties include general properties, which apply to the flat file data object. They also include column properties, which apply to the columns in the flat file data object. The Developer tool displays overview properties for flat files in the Overview view.

The following table describes the general properties that you configure for flat files:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the flat file data object.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the flat file data object.</td>
</tr>
</tbody>
</table>

The following table describes the column properties that you configure for flat files:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the column.</td>
</tr>
<tr>
<td>Native type</td>
<td>Native datatype of the column.</td>
</tr>
<tr>
<td>Bytes to process (fixed-width flat files)</td>
<td>Number of bytes that the Data Integration Service reads or writes for the column.</td>
</tr>
<tr>
<td>Precision</td>
<td>Maximum number of significant digits for numeric datatypes, or maximum number of characters for string datatypes. For numeric datatypes, precision includes scale.</td>
</tr>
<tr>
<td>Scale</td>
<td>Maximum number of digits after the decimal point for numeric values.</td>
</tr>
<tr>
<td>Format</td>
<td>Column format for numeric and datetime datatypes.</td>
</tr>
<tr>
<td></td>
<td>For numeric datatypes, the format defines the thousand separator and decimal separator. Default is no thousand separator and a period (.) for the decimal separator.</td>
</tr>
<tr>
<td></td>
<td>For datetime datatypes, the format defines the display format for year, month, day, and time. It also defines the field width. Default is &quot;A 19 YYYY-MM-DD HH24:MI:SS.&quot;</td>
</tr>
<tr>
<td>Visibility</td>
<td>Determines whether the Data Integration Service can read data from or write data to the column. For example, when the visibility is Read, the Data Integration Service can read data from the column. It cannot write data to the column. For flat file data objects, this property is read-only. The visibility is always Read and Write.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the column.</td>
</tr>
</tbody>
</table>
Flat File Data Object Read Properties

The Data Integration Service uses read properties when it reads data from a flat file. Select the source transformation to edit general, column, and format properties. Select the Output transformation to edit runtime properties.

The following figure shows the read properties for the selected SALES_TRANSACTIONS source transformation:

1. Source transformation
2. Output transformation
3. Read view
4. Properties view
General Properties

The Developer tool displays general properties for flat file sources in the source transformation in the Read view.

The following table describes the general properties that you configure for flat file sources:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the flat file. This property is read-only. You can edit the name in the Overview view. When you use the flat file as a source in a mapping, you can edit the name within the mapping.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the flat file.</td>
</tr>
</tbody>
</table>

Column Properties

The Developer tool displays column properties for flat file sources in the source transformation in the Read view.

The following table describes the column properties that you configure for flat file sources:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the column.</td>
</tr>
<tr>
<td>Native type</td>
<td>Native datatype of the column.</td>
</tr>
<tr>
<td>Bytes to process (fixed-width flat files)</td>
<td>Number of bytes that the Data Integration Service reads for the column.</td>
</tr>
<tr>
<td>Precision</td>
<td>Maximum number of significant digits for numeric datatypes, or maximum number of characters for string datatypes. For numeric datatypes, precision includes scale.</td>
</tr>
<tr>
<td>Scale</td>
<td>Maximum number of digits after the decimal point for numeric values.</td>
</tr>
<tr>
<td>Format</td>
<td>Column format for numeric and datetime datatypes.</td>
</tr>
<tr>
<td></td>
<td>For numeric datatypes, the format defines the thousand separator and decimal separator. Default is no thousand separator and a period (.) for the decimal separator.</td>
</tr>
<tr>
<td></td>
<td>For datetime datatypes, the format defines the display format for year, month, day, and time. It also defines the field width. Default is &quot;A 19 YYYY-MM-DD HH24:MI:SS.&quot;</td>
</tr>
<tr>
<td>Shift key (fixed-width flat files)</td>
<td>Allows the user to define a shift-in or shift-out statefulness for the column in the fixed-width flat file.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the column.</td>
</tr>
</tbody>
</table>
Format Properties

The Developer tool displays format properties for flat file sources in the source transformation in the **Read** view.

The following table describes the format properties that you configure for delimited flat file sources:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start import at line</td>
<td>Row at which the Data Integration Service starts importing data. Use this option to skip header rows. Default is 1.</td>
</tr>
<tr>
<td>Row delimiter</td>
<td>Octal code for the character that separates rows of data. Default is line feed, \012 LF (\n).</td>
</tr>
<tr>
<td>Escape character</td>
<td>Character used to escape a delimiter character in an unquoted string if the delimiter is the next character after the escape character. If you specify an escape character, the Data Integration Service reads the delimiter character as a regular character embedded in the string. <strong>Note:</strong> You can improve mapping performance slightly if the source file does not contain quotes or escape characters.</td>
</tr>
<tr>
<td>Retain escape character in data</td>
<td>Includes the escape character in the output string. Default is disabled.</td>
</tr>
<tr>
<td>Treat consecutive delimiters as one</td>
<td>Causes the Data Integration Service to treat one or more consecutive column delimiters as one. Otherwise, the Data Integration Service reads two consecutive delimiters as a null value. Default is disabled.</td>
</tr>
</tbody>
</table>

The following table describes the format properties that you configure for fixed-width flat file sources:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start import at line</td>
<td>Row at which the Data Integration Service starts importing data. Use this option to skip header rows. Default is 1.</td>
</tr>
<tr>
<td>Number of bytes to skip between records</td>
<td>Number of bytes between the last column of one row and the first column of the next. The Data Integration Service skips the entered number of bytes at the end of each row to avoid reading carriage return characters or line feed characters. Enter 1 for UNIX files and 2 for DOS files. Default is 2.</td>
</tr>
<tr>
<td>Line sequential</td>
<td>Causes the Data Integration Service to read a line feed character or carriage return character in the last column as the end of the column. Select this option if the file uses line feeds or carriage returns to shorten the last column of each row. Default is disabled.</td>
</tr>
</tbody>
</table>
### Property Description

**Strip trailing blanks**
Strips trailing blanks from string values. Default is disabled.

**User defined shift state**
Allows you to select the shift state for source columns in the Columns properties. Select this option when the source file contains both multibyte and single-byte data, but does not contain shift-in and shift-out keys. If a multibyte file source does not contain shift keys, you must select a shift key for each column in the flat file data object. Select the shift key for each column to enable the Data Integration Service to read each character correctly. Default is disabled.

### Run-time Properties

The Developer tool displays run-time properties for flat file sources in the Output transformation in the **Read** view.

The following table describes the run-time properties that you configure for flat file sources:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input type</strong></td>
<td>Type of source input. You can choose the following types of source input:</td>
</tr>
<tr>
<td></td>
<td>- File. For flat file sources.</td>
</tr>
<tr>
<td></td>
<td>- Command. For source data or a file list generated by a shell command.</td>
</tr>
<tr>
<td><strong>Source type</strong></td>
<td>Indicates source type of files with the same file properties. You can choose one of the following source types:</td>
</tr>
<tr>
<td></td>
<td>- Direct. A source file that contains the source data.</td>
</tr>
<tr>
<td></td>
<td>- Indirect. A source file that contains a list of files. The Data Integration Service reads the file list and reads the files in sequential order.</td>
</tr>
<tr>
<td></td>
<td>- Directory. Source files that are in a directory. You must specify the directory location in the source file directory property. The Data Integration Service reads the files in ascending alphabetic order. The Data Integration Service does not read files in the subdirectories.</td>
</tr>
<tr>
<td><strong>Source file name</strong></td>
<td>File name of the flat file source.</td>
</tr>
<tr>
<td><strong>Source file directory</strong></td>
<td>Directory where the flat file sources exist. The machine that hosts Informatica services must be able to access this directory. Default is the SourceDir system parameter.</td>
</tr>
<tr>
<td><strong>Concurrent Read Partitioning</strong></td>
<td>Order in which multiple partitions read input rows from a source file. If the Data Integration Service does not create partitions for the mapping, it ignores this value. Select one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- Optimize throughput. The Data Integration Service does not preserve input row order.</td>
</tr>
<tr>
<td></td>
<td>- Keep relative order. The Data Integration Service preserves the input row order for the rows read by each partition.</td>
</tr>
<tr>
<td></td>
<td>- Keep absolute order. The Data Integration Service preserves the input row order for all rows read by all partitions.</td>
</tr>
<tr>
<td></td>
<td>PowerCenter Express does not support this option. If you select a value, the Data Integration Service ignores it.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Command</td>
<td>Command used to generate the source file data. Use a command to generate or transform flat file data and send the standard output of the command to the flat file reader when the mapping runs. The flat file reader reads the standard output as the flat file source data. Generating source data with a command eliminates the need to stage a flat file source. Use a command or script to send source data directly to the Data Integration Service instead of using a pre-mapping command to generate a flat file source. You can also use a command to generate a file list. For example, to use a directory listing as a file list, use the following command: <code>cd MySourceFiles; ls sales-records-Sep-*-2005.dat</code></td>
</tr>
<tr>
<td>Truncate string null</td>
<td>Strips the first null character and all characters after the first null character from string values. Enable this option for delimited flat files that contain null characters in strings. If you do not enable this option, the Data Integration Service generates a row error for any row that contains null characters in a string. Default is disabled.</td>
</tr>
<tr>
<td>Line sequential buffer length</td>
<td>Number of bytes that the Data Integration Service reads for each line. This property, together with the total row size, determines whether the Data Integration Service drops a row. If the row exceeds the larger of the line sequential buffer length or the total row size, the Data Integration Service drops the row and writes it to the mapping log file. To determine the total row size, add the column precision and the delimiters, and then multiply the total by the maximum bytes for each character. Default is 1024.</td>
</tr>
</tbody>
</table>

Configuring Flat File Read Properties

Configure read properties to control how the Data Integration Service reads data from a flat file.

1. Open the flat file data object.
2. Select the **Read** view.
3. To edit general, column, or format properties, select the source transformation. To edit run-time properties, select the **Output** transformation.
4. In the **Properties** view, select the properties you want to edit.
   - For example, click **Columns** properties or **Runtime** properties.
5. Edit the properties.
6. Save the flat file data object.

Flat File Data Object Write Properties

The Data Integration Service uses write properties when it writes data to a flat file. Select the **Input** transformation to edit run-time properties. Select the **Target** transformation to edit general and column properties.
The following figure shows the write properties for the selected Input transformation:

1. Input transformation
2. Target transformation
3. Write view
4. Properties view
### Run-time Properties

The Developer tool displays run-time properties for flat file targets in the Input transformation in the **Write** view.

The following table describes the run-time properties that you configure for flat file targets:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Append if exists</td>
<td>Appends the output data to the target files and reject files.</td>
</tr>
<tr>
<td></td>
<td>If you do not select this option, the Data Integration Service truncates the target file and</td>
</tr>
<tr>
<td></td>
<td>reject file before writing data to them. If the files do not exist, the Data Integration</td>
</tr>
<tr>
<td></td>
<td>Service creates them. Default is disabled.</td>
</tr>
<tr>
<td>Create directory if not</td>
<td>Creates the target directory if it does not exist.</td>
</tr>
<tr>
<td>exists</td>
<td>Default is disabled.</td>
</tr>
<tr>
<td>Header options</td>
<td>Creates a header row in the file target. You can choose the following options:</td>
</tr>
<tr>
<td></td>
<td>- No header. Does not create a header row in the flat file target.</td>
</tr>
<tr>
<td></td>
<td>- Output field names. Creates a header row in the file target with the output port names.</td>
</tr>
<tr>
<td></td>
<td>- Use header command output. Uses the command in the Header Command field to generate a header</td>
</tr>
<tr>
<td></td>
<td>row. For example, you can use a command to add the date to a header row for the file target.</td>
</tr>
<tr>
<td></td>
<td>Default is no header.</td>
</tr>
<tr>
<td>Header command</td>
<td>Command used to generate the header row in the file target.</td>
</tr>
<tr>
<td>Footer command</td>
<td>Command used to generate the footer row in the file target.</td>
</tr>
<tr>
<td>Output type</td>
<td>Type of target for the mapping. Select File to write the target data to a flat file. Select</td>
</tr>
<tr>
<td></td>
<td>Command to output data to a command.</td>
</tr>
<tr>
<td>Output file directory</td>
<td>Output directory for the flat file target. The machine that hosts Informatica services must</td>
</tr>
<tr>
<td></td>
<td>be able to access this directory. Default is the TargetDir system parameter.</td>
</tr>
<tr>
<td>Output file name</td>
<td>File name of the flat file target.</td>
</tr>
<tr>
<td>Merge type</td>
<td>Type of merge that the Data Integration Service performs on the data for partitioned targets.</td>
</tr>
<tr>
<td></td>
<td>If the Data Integration Service does not create partitions for the target, it ignores this value.</td>
</tr>
<tr>
<td></td>
<td>Select one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- No merge. The Data Integration Service concurrently writes the target output to a separate</td>
</tr>
<tr>
<td></td>
<td>file for each partition.</td>
</tr>
<tr>
<td></td>
<td>- Sequential. The Data Integration Service creates an output file for each partition and then</td>
</tr>
<tr>
<td></td>
<td>merges them into a single merge file at the end of the mapping.</td>
</tr>
<tr>
<td></td>
<td>- File list. The Data Integration Service creates a target file for each partition and creates</td>
</tr>
<tr>
<td></td>
<td>a file list that contains the paths of the individual files.</td>
</tr>
<tr>
<td></td>
<td>- Concurrent. The Data Integration Service concurrently writes the data for all target partitions</td>
</tr>
<tr>
<td></td>
<td>to the merge file. Because the Data Integration Service writes to the merge file concurrently for</td>
</tr>
<tr>
<td></td>
<td>all partitions, the sort order of the data in the merge file might not be sequential.</td>
</tr>
<tr>
<td></td>
<td>PowerCenter Express does not support this option. If you select a merge type, the Data Integration</td>
</tr>
<tr>
<td></td>
<td>Service ignores it.</td>
</tr>
</tbody>
</table>

Flat File Data Objects 83
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merge file directory</td>
<td>Directory for the merge file for all target partitions. The machine that hosts Informatica services must be able to access this directory. PowerCenter Express does not support this option. If you enter a value, the Data Integration Service ignores it. Default is the TargetDir system parameter.</td>
</tr>
<tr>
<td>Merge file name</td>
<td>Name of the merge file for all target partitions. Default is the output file name. PowerCenter Express does not support this option. If you enter a value, the Data Integration Service ignores it.</td>
</tr>
<tr>
<td>Command</td>
<td>Command used to process the target data. On UNIX, use any valid UNIX command or shell script. On Windows, use any valid DOS command or batch file. The flat file writer sends the data to the command instead of a flat file target. You can improve mapping performance by pushing transformation tasks to the command instead of the Data Integration Service. You can also use a command to sort or to compress target data. For example, use the following command to generate a compressed file from the target data: <code>compress -c - &gt; MyTargetFiles/MyCompressedFile.Z</code></td>
</tr>
<tr>
<td>Merge command</td>
<td>Merge command used to process merge data for all target partitions. The Data Integration Service must use a concurrent merge type for a command to process merge data. The command might not maintain the order of the target data. PowerCenter Express does not support this option. If you enter a value, the Data Integration Service ignores it.</td>
</tr>
<tr>
<td>Reject file directory</td>
<td>Directory where the reject file exists. Default is the RejectDir system parameter. <strong>Note:</strong> This field appears when you edit a flat file target in a mapping.</td>
</tr>
<tr>
<td>Reject file name</td>
<td>File name of the reject file. Default is <code>&lt;output_file_name&gt;.bad</code>. <strong>Note:</strong> This field appears when you edit a flat file target in a mapping.</td>
</tr>
</tbody>
</table>

**General Properties**

The Developer tool displays general properties for flat file targets in the target transformation in the **Write** view.

The following table describes the general properties that you configure for flat file targets:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the flat file. This property is read-only. You can edit the name in the <strong>Overview</strong> view. When you use the flat file as a target in a mapping, you can edit the name within the mapping.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the flat file.</td>
</tr>
</tbody>
</table>
Column Properties

The Developer tool displays column properties for flat file targets in the target transformation in the Write view.

The following table describes the column properties that you configure for flat file targets:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the column.</td>
</tr>
<tr>
<td>Native type</td>
<td>Native datatype of the column.</td>
</tr>
<tr>
<td>Bytes to process (fixed-width files)</td>
<td>Number of bytes that the Data Integration Service writes for the column.</td>
</tr>
<tr>
<td>Precision</td>
<td>Maximum number of significant digits for numeric datatypes, or maximum number of characters for string datatypes. For numeric datatypes, precision includes scale.</td>
</tr>
<tr>
<td>Scale</td>
<td>Maximum number of digits after the decimal point for numeric values.</td>
</tr>
<tr>
<td>Format</td>
<td>Column format for numeric and datetime datatypes.</td>
</tr>
<tr>
<td></td>
<td>For numeric datatypes, the format defines the thousand separators and decimal separators. Default is no thousand separator and a period (.) for the decimal separator.</td>
</tr>
<tr>
<td></td>
<td>For datetime datatypes, the format defines the display format for year, month, day, and time. It also defines the field width. Default is &quot;A 19 YYYY-MM-DD HH24:MI:SS.&quot;</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the column.</td>
</tr>
</tbody>
</table>

Configuring Flat File Write Properties

Configure write properties to control how the Data Integration Service writes data to a flat file.

1. Open the flat file data object.
2. Select the Write view.
3. To edit run-time properties, select the Input transformation. To edit general or column properties, select the target transformation.
4. In the Properties view, select the properties you want to edit.
   For example, click Runtime properties or Columns properties.
5. Edit the properties.
6. Save the flat file data object.

Flat File Data Object Advanced Properties

The Data Integration Service uses advanced properties when it reads data from or writes data to a flat file. The Developer tool displays advanced properties for flat files in the Advanced view.
The following table describes the advanced properties that you configure for flat files:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code page</td>
<td>Code page of the flat file data object. For source files, use a source code page that is a subset of the target code page. For lookup files, use a code page that is a superset of the source code page and a subset of the target code page. For target files, use a code page that is a superset of the source code page. Default is &quot;MS Windows Latin 1 (ANSI), superset of Latin 1.&quot;</td>
</tr>
<tr>
<td>Format</td>
<td>Format for the flat file, either delimited or fixed-width.</td>
</tr>
<tr>
<td>Delimiters (delimited flat files)</td>
<td>Character used to separate columns of data.</td>
</tr>
<tr>
<td>Null character type (fixed-width flat files)</td>
<td>Null character type, either text or binary.</td>
</tr>
<tr>
<td>Null character (fixed-width flat files)</td>
<td>Character used to represent a null value. The null character can be any valid character in the file code page or any binary value from 0 to 255.</td>
</tr>
<tr>
<td>Repeat null character (fixed-width flat files)</td>
<td>For source files, causes the Data Integration Service to read repeat null characters in a single field as one null value. For target files, causes the Data Integration Service to write as many null characters as possible into the target field. If you do not enable this option, the Data Integration Service enters one null character at the beginning of the field to represent a null value. Default is disabled.</td>
</tr>
<tr>
<td>Datetime format</td>
<td>Defines the display format and the field width for datetime values. Default is &quot;A 19 YYYY-MM-DD HH24:MI:SS.&quot;</td>
</tr>
<tr>
<td>Thousand separator</td>
<td>Thousand separator for numeric values. Default is None.</td>
</tr>
<tr>
<td>Decimal separator</td>
<td>Decimal separator for numeric values. Default is a period (.).</td>
</tr>
<tr>
<td>Tracing level</td>
<td>Controls the amount of detail in the mapping log file. <strong>Note:</strong> This field appears when you edit a flat file source or target in a mapping.</td>
</tr>
</tbody>
</table>

### Creating a Flat File Data Object

Create a flat file data object to define the data object columns and rows.

1. Select a project or folder in the **Object Explorer** view.
2. Click **File > New > Data Object**.
3. Select **Physical Data Objects > Flat File Data Object** and click **Next**.
   - The **New Flat File Data Object** dialog box appears.
4. Select Create as Empty.
5. Enter a name for the data object.
6. Optionally, click **Browse** to select a project or folder for the data object.
7. Click **Next**.
8. Select a code page that matches the code page of the data in the file.
9. Select **Delimited** or **Fixed-width**.
10. If you selected **Fixed-width**, click **Finish**. If you selected **Delimited**, click **Next**.
11. Configure the following properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delimiters</td>
<td>Character used to separate columns of data. If you enter a delimiter that is the same as the escape character or the quote character, you might receive unexpected results.</td>
</tr>
<tr>
<td>Text Qualifier</td>
<td>Quote character that defines the boundaries of text strings. If you select a quote character, the Developer tool ignores delimiters within a pair of quotes.</td>
</tr>
</tbody>
</table>

12. Click **Finish**.

The data object appears under Data Object in the project or folder in the **Object Explorer** view.

### Importing a Fixed-Width Flat File Data Object

Import a fixed-width flat file data object when you have a fixed-width flat file that defines the metadata you want to include in a mapping, mapplet, or profile.

1. Click **File > New > Data Object**.
   The **New** dialog box appears.
2. Select **Physical Data Objects > Flat File Data Object** and click **Next**.
   The **New Flat File Data Object** dialog box appears.
3. Enter a name for the data object.
4. Click **Browse** and navigate to the directory that contains the file.
5. Click **Open**.
   The wizard names the data object the same name as the file you selected.
6. Optionally, edit the data object name.
7. Click **Next**.
8. Select a code page that matches the code page of the data in the file.
9. Select **Fixed-Width**.
10. Optionally, edit the maximum number of rows to preview.
11. Click **Next**.
12. Configure the following properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import Field Names From First Line</td>
<td>If selected, the Developer tool uses data in the first row for column names. Select this option if column names appear in the first row.</td>
</tr>
<tr>
<td>Start Import At Row</td>
<td>Row number at which the Data Integration Service starts reading when it imports the file. For example, if you specify to start at the second row, the Developer tool skips the first row before reading.</td>
</tr>
</tbody>
</table>

13. Click **Edit Breaks** to edit column breaks. Or, follow the directions in the wizard to manipulate the column breaks in the file preview window.

You can move column breaks by dragging them. Or, double-click a column break to delete it.

14. Click **Next** to preview the physical data object.

15. Click **Finish**.

The data object appears under Data Object in the project or folder in the **Object Explorer** view.

### Importing a Delimited Flat File Data Object

Import a delimited flat file data object when you have a delimited flat file that defines the metadata you want to include in a mapping, mapplet, or profile.

1. Select a project or folder in the **Object Explorer** view.
2. Click **File > New > Data Object**.
   The **New** dialog box appears.
3. Select **Physical Data Objects > Flat File Data Object** and click **Next**.
   The **New Flat File Data Object** dialog box appears.
4. Enter a name for the data object.
5. Click **Browse** and navigate to the directory that contains the file.
6. Click **Open**.
   The wizard names the data object the same name as the file you selected.
7. Optionally, edit the data object name.
8. Click **Next**.
9. Select a code page that matches the code page of the data in the file.
10. Select **Delimited**.
11. Optionally, edit the maximum number of rows to preview.
12. Click **Next**.
13. Configure the following properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delimiters</td>
<td>Character used to separate columns of data. If you enter a delimiter that is the same as the escape character or the quote character, you might receive unexpected results.</td>
</tr>
<tr>
<td>Text Qualifier</td>
<td>Quote character that defines the boundaries of text strings. If you select a quote character, the Developer tool ignores delimiters within pairs of quotes.</td>
</tr>
<tr>
<td>Import Field Names From First Line</td>
<td>If selected, the Developer tool uses data in the first row for column names. Select this option if column names appear in the first row. The Developer tool prefixes &quot;FIELD_ &quot; to field names that are not valid.</td>
</tr>
<tr>
<td>Row Delimiter</td>
<td>Specify a line break character. Select from the list or enter a character. Preface an octal code with a backslash (). To use a single character, enter the character. The Data Integration Service uses only the first character when the entry is not preceded by a backslash. The character must be a single-byte character, and no other character in the code page can contain that byte. Default is line-feed, '\012 LF (\n).</td>
</tr>
<tr>
<td>Escape Character</td>
<td>Character immediately preceding a column delimiter character embedded in an unquoted string, or immediately preceding the quote character in a quoted string. When you specify an escape character, the Data Integration Service reads the delimiter character as a regular character.</td>
</tr>
<tr>
<td>Start Import At Row</td>
<td>Row number at which the Data Integration Service starts reading when it imports the file. For example, if you specify to start at the second row, the Developer tool skips the first row before reading.</td>
</tr>
<tr>
<td>Treat Consecutive Delimiters as One</td>
<td>If selected, the Data Integration Service reads one or more consecutive column delimiters as one. Otherwise, the Data Integration Service reads two consecutive delimiters as a null value.</td>
</tr>
<tr>
<td>Remove Escape Character From Data</td>
<td>Removes the escape character in the output string.</td>
</tr>
</tbody>
</table>

14. Click **Next** to preview the data object.
15. Click **Finish**.

The data object appears under Data Object in the project or folder in the **Object Explorer** view.

**WSDL Data Object**

A WSDL data object is a physical data object that uses a WSDL file as a source. You can use a WSDL data object to create a Web Service Consumer transformation. Import a WSDL file to create a WSDL data object. After you import a WSDL data object, you can edit general and advanced properties in the **Overview** and **Advanced** views. The **WSDL** view displays the WSDL file content.
The following figure shows a sample WSDL data object:

Consider the following guidelines when you import a WSDL:

- The WSDL file must be WSDL 1.1 compliant.
- The WSDL file must be valid.
- Operations that you want to include in a Web Service Consumer transformation must use Document/Literal encoding. The WSDL import fails if all operations in the WSDL file use an encoding type other than Document/Literal.
- The Developer tool must be able to access any schema that the WSDL file references.
- If a WSDL file contains a schema or has an external schema, the Developer tool creates an embedded schema within the WSDL data object.
- If a WSDL file imports another WSDL file, the Developer tool combines both WSDLs to create the WSDL data object.
- If a WSDL file defines multiple operations, the Developer tool includes all operations in the WSDL data object.

**WSDL Data Object Overview View**

The WSDL data object **Overview** view displays general information about the WSDL and operations in the WSDL.

The following table describes the general properties that you configure for a WSDL data object:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the WSDL data object.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the WSDL data object.</td>
</tr>
</tbody>
</table>
The following table describes the columns for operations defined in the WSDL data object:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
<td>The location where the WSDL defines the message format and protocol for the operation.</td>
</tr>
<tr>
<td>Input</td>
<td>The WSDL message name associated with the operation input.</td>
</tr>
<tr>
<td>Output</td>
<td>The WSDL message name associated with the operation output.</td>
</tr>
<tr>
<td>Fault</td>
<td>The WSDL message name associated with the operation fault.</td>
</tr>
</tbody>
</table>

**WSDL Data Object Advanced View**

The WSDL data object **Advanced** view displays advanced properties for a WSDL data object.

The following table describes the advanced properties for a WSDL data object:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>Default web service connection for a Web Service Consumer transformation.</td>
</tr>
<tr>
<td>File Location</td>
<td>Location where the WSDL file exists.</td>
</tr>
</tbody>
</table>

**Importing a WSDL Data Object**

To create a Web Service Consumer transformation, import a WSDL data object. You can import a WSDL data object from a WSDL file or a URI that points to the WSDL location. You can import a WSDL data object from a WSDL file that contains either a SOAP 1.1 or SOAP 1.2 binding operation or both.

1. Click **File > New > Data Object**.
2. Select **WSDL data object** and click **Next**.
   The **New WSDL Data Object** dialog box appears.
3. Click **Browse** next to the **WSDL** option and enter the location of the WSDL. Then, click **OK**.
   When you enter the location of the WSDL, you can browse to the WSDL file or you can enter the URI to the WSDL.
   **Note**: If the URI contains non-English characters, the import might fail. Copy the URI to the address bar in any browser. Copy the location back from the browser. The Developer tool accepts the encoded URI from the browser.
4. Enter a name for the WSDL.
5. Click **Browse** next to the **Location** option to select the project or folder location where you want to import the WSDL data object.
6. Click **Next** to view the operations in the WSDL.
7. Click **Finish**.

   The data object appears under **Physical Data Object** in the project or folder in the **Object Explorer** view.

**WSDL Synchronization**

You can synchronize a WSDL data object when the WSDL files change. When you synchronize a WSDL data object, the Developer tool reimports the object metadata from the WSDL files.

You use a WSDL data object to create a Web Service Consumer transformation. When you update a WSDL data object, the Developer tool updates the objects that reference the WSDL and marks them as changed when you open them. When the Developer tool compares the new WSDL with the old WSDL, it identifies WSDL components through the name attributes.

If no name attribute changes, the Developer tool updates the objects that reference the WSDL components. For example, you edit a WSDL file and change the type for simple element "CustID" from xs:string to xs:integer. When you synchronize the WSDL data object, the Developer tool updates the element type in all Web Service Consumer transformations that reference the CustID element.

If a name attribute changes, the Developer tool marks the objects that reference the WSDL component as changed when you open them.

The Developer tool validates the WSDL files before it updates the WSDL data object. If the WSDL files contain errors, the Developer tool does not import the files.

**Synchronizing a WSDL Data Object**

Synchronize a WSDL data object when the WSDL files change.

1. Right-click the WSDL data object in the **Object Explorer** view, and select **Synchronize**.

   The **Synchronize WSDL Data Object** dialog box appears.

2. Click **Browse** next to the **WSDL** field, and enter the location of the WSDL. Then, click **OK**.

   When you enter the location of the WSDL, you can browse to the WSDL file or you can enter the URI to the WSDL.

   **Note**: If the URI contains non-English characters, the import might fail. Copy the URI to the address bar in any browser. Copy the location back from the browser. The Developer tool accepts the encoded URI from the browser.

3. Verify the WSDL name and location.

4. Click **Next** to view the operations in the WSDL.

5. Click **Finish**.

   The Developer tool updates the objects that reference the WSDL and marks them as changed when you open them.

**Certificate Management**

The Developer tool must use a certificate to import WSDL data objects and schema objects from a URL that requires client authentication.

By default, the Developer tool imports objects from URLs that require client authentication when the server that hosts the URL uses a trusted certificate. When the server that hosts the URL uses an untrusted certificate, add the untrusted certificate to the Developer tool. If you do not add the untrusted certificate to the
Developer tool, the Developer tool cannot import the object. Request the certificate file and password from the server administrator for the URL that you want import objects from.

The certificates that you add to the Developer tool apply to imports that you perform on the Developer tool machine. The Developer tool does not store certificates in the Model repository.

Informatica Developer Certificate Properties

Add certificates to the Developer tool when you want to import objects from a URL that requires client authentication with an untrusted certificate.

The following table describes the certificate properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Name</td>
<td>Name of the server that hosts the URL.</td>
</tr>
<tr>
<td>Port Number</td>
<td>Port number of the URL.</td>
</tr>
<tr>
<td>Certificate File Path</td>
<td>Location of the client certificate file.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the client certificate file.</td>
</tr>
</tbody>
</table>

Adding Certificates to Informatica Developer

When you add a certificate, you configure the certificate properties that the Developer tool uses when you import objects from a URL that requires client authentication with an untrusted certificate.

1. Click Windows > Preferences.
2. Select Informatica > Web Services > Certificates.
3. Click Add.
4. Configure the certificate properties.
5. Click OK.

Synchronization

You can synchronize physical data objects when their sources change. When you synchronize a physical data object, the Developer tool reimports the object metadata from the source you select.

You can synchronize all physical data objects. When you synchronize relational data objects or customized data objects, you can retain or overwrite the key relationships you define in the Developer tool.

You can configure a customized data object to be synchronized when its sources change. For example, a customized data object uses a relational data object as a source, and you add a column to the relational data object. The Developer tool adds the column to the customized data object. To synchronize a customized data object when its sources change, select the Synchronize input and output option in the Overview properties of the customized data object.

To synchronize any physical data object, right-click the object in the Object Explorer view, and select Synchronize.
Synchronizing a Flat File Data Object

You can synchronize the changes to an external flat file data source with its data object in Informatica Developer. Use the Synchronize Flat File wizard to synchronize the data objects.

1. In the Object Explorer view, select a flat file data object.
2. Right-click and select Synchronize.
   The Synchronize Flat File Data Object wizard appears.
3. Verify the flat file path in the Select existing flat file field.
4. Click Next.
5. Optionally, select the code page, format, delimited format properties, and column properties.
6. Click Finish, and then click OK.

Synchronizing a Relational Data Object

You can synchronize external data source changes of a relational data source with its data object in Informatica Developer. External data source changes include adding, changing, and removing columns, and changes to rules.

1. In the Object Explorer view, select a relational data object.
2. Right-click and select Synchronize.
   A message prompts you to confirm the action.
3. To complete the synchronization process, click OK. Click Cancel to cancel the process.
   If you click OK, a synchronization process status message appears.
4. When you see a Synchronization complete message, click OK.
   The message displays a summary of the metadata changes made to the data object.

Reject Files

During a mapping run, the Data Integration Service creates a reject file for each target instance in the mapping. If the writer or the target rejects data, the Data Integration Service writes the rejected row to the reject file. The reject file and mapping log contain information that helps you determine the cause of the rejection.

If the reject file does not contain any rejected rows, the Data Integration Service deletes the reject file at the end of the mapping run.

Each time you run a mapping, the Data Integration Service appends rejected data to the reject file. Depending on the source of the problem, you can correct the mapping and target database to prevent rejects in subsequent mappings.

Location of Reject Files

The Data Integration Service creates reject files for each target instance in the mapping. It creates reject files in the target reject file directory.

Configure the target reject file directory in the run-time properties for a flat file or relational target in a mapping. By default, the Data Integration Service creates reject files in the directory defined by the RejectDir
system parameter. The Data Integration Service names reject files after the target instance name. The default name for reject files is `<file_name>.bad`

**Content of Reject Files**

After you find a reject file, you can read it using a text editor that supports the reject file code page.

Reject files contain rows of data rejected by the writer or the target database. The Data Integration Service writes the entire row in the reject file. However, the problem usually centers on one column within the row. To help you determine which column caused the row to be rejected, the reject file contains indicators that give you more information about each column.

Reject files contain the following indicators:

- **Row indicator**
  - The first column in each row of the reject file is the row indicator. The row indicator defines whether the row was marked for insert, update, delete, or reject.

- **Column indicator**
  - Column indicators appear after every column of data. The column indicator defines whether the column contains valid, overflow, null, or truncated data.

**Row Indicators**

The first column in the reject file is the row indicator. The row indicator is a flag that defines the update strategy for the data row.

This text is conref'd from Shared > Reject Files > Reject Files topic

The following table describes the row indicators in a reject file:

<table>
<thead>
<tr>
<th>Row Indicator</th>
<th>Meaning</th>
<th>Rejected By</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Insert</td>
<td>Writer or target</td>
</tr>
<tr>
<td>1</td>
<td>Update</td>
<td>Writer or target</td>
</tr>
<tr>
<td>2</td>
<td>Delete</td>
<td>Writer or target</td>
</tr>
<tr>
<td>3</td>
<td>Reject. Marked for reject by an update strategy expression.</td>
<td>Writer</td>
</tr>
<tr>
<td>4</td>
<td>Rolled-back insert</td>
<td>Writer</td>
</tr>
<tr>
<td>5</td>
<td>Rolled-back update</td>
<td>Writer</td>
</tr>
<tr>
<td>6</td>
<td>Rolled-back delete</td>
<td>Writer</td>
</tr>
<tr>
<td>7</td>
<td>Committed insert</td>
<td>Writer</td>
</tr>
<tr>
<td>8</td>
<td>Committed update</td>
<td>Writer</td>
</tr>
<tr>
<td>9</td>
<td>Committed delete</td>
<td>Writer</td>
</tr>
</tbody>
</table>
The following sample reject file shows the row indicator of "0" for each row that indicates an insert update strategy for the row:

0,D,1921,D,Nelson,D,William,D,415-541-5145,D
0,D,1922,D,Page,D,Ian,D,415-541-5145,D
0,D,1923,D,Osborne,D,Lyle,D,415-541-5145,D
0,D,1928,D,De Souza,D,Leo,D,415-541-5145,D
0,D,2001123456789,O,S. MacDonald,D,Ira,D,415-541-514566,T

Column Indicators

A column indicator appears after every column of data. A column indicator defines whether the data is valid, overflow, null, or truncated.

<table>
<thead>
<tr>
<th>Column Indicator</th>
<th>Type of data</th>
<th>Writer Treats As</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>Valid data.</td>
<td>Good data. Writer passes it to the target database. The target accepts it unless a database error occurs, such as finding a duplicate key.</td>
</tr>
<tr>
<td>N</td>
<td>Null. The column contains a null value.</td>
<td>Good data. Writer passes it to the target, which rejects it if the target database does not accept null values.</td>
</tr>
<tr>
<td>T</td>
<td>Truncated. String data exceeded a specified precision for the column, so the value was truncated.</td>
<td>Bad data, if you configured the mapping target to reject overflow or truncated data.</td>
</tr>
</tbody>
</table>

Null columns appear in the reject file with commas marking their column. The following example shows a null column surrounded by good data:

0,D,5,,N,5,D

The column indicator "D" also appears after each row indicator. The following example shows the column indicator "D" after the row indicator "0":

0,D,2001123456789,O,S. MacDonald,D,Ira,D,415-541-514566,T

Either the writer or target database can reject a row. Consult the log to determine the cause for rejection.

Troubleshooting Physical Data Objects

I am trying to preview a relational data object or a customized data object source transformation and the preview fails.

Verify that the resource owner name is correct.

When you import a relational resource, the Developer tool imports the owner name when the user name and schema from which the table is imported do not match. If the user name and schema from which the table is imported match, but the database default schema has a different name, preview fails because the Data
Integration Service executes the preview query against the database default schema, where the table does not exist.

Update the relational data object or the source transformation and enter the correct resource owner name. The owner name appears in the relational data object or the source transformation Advanced properties.

I am trying to preview a flat file data object and the preview fails. I get an error saying that the system cannot find the path specified.

Verify that the machine that hosts Informatica services can access the source file directory.

For example, you create a flat file data object by importing the following file on your local machine, MyClient:

```
C:\MySourceFiles\MyFile.csv
```

In the Read view, select the Runtime properties in the Output transformation. The source file directory is "C:\MySourceFiles."

When you preview the file, the Data Integration Service tries to locate the file in the "C:\MySourceFiles" directory on the machine that hosts Informatica services. If the directory does not exist on the machine that hosts Informatica services, the Data Integration Service returns an error when you preview the file.

To work around this issue, use the network path as the source file directory. For example, change the source file directory from "C:\MySourceFiles" to "\MyClient\MySourceFiles." Share the "MySourceFiles" directory so that the machine that hosts Informatica services can access it.
Profiles Overview

A column profile determines the characteristics of columns in a data source, such as value frequency, percentages, and patterns.

Column profiling discovers the following facts about data:

- The number of unique and null values in each column, expressed as a number and a percentage.
- The patterns of data in each column and the frequencies with which these values occur.
- Statistics about the column values, such as the maximum and minimum lengths of values and the first and last values in each column.

Use column profile options to select the columns on which you want to run a profile, set data sampling options, and set drill-down options when you create a profile.

A rule is business logic that defines conditions applied to source data when you run a profile. You can add a rule to the profile to validate data.
Profile Views

You can view and add information about a profile in Informatica Developer using the **Overview**, **Definition**, **Comments**, and **Results** views.

The following figure shows the profile views in the editor:

![Profile Views](Profile_Customer_Central.png)

When you open a profile from the **Object Explorer** view, the editor on the right pane shows the profile information under the following views:

**Overview**

- View and provide general information about the profile, such as name, description, and location.

**Definition**

- View and set up the profile definition.
  - This information includes the list of filters and rules you assign to the profile.

**Results**

- Shows the results of profile run. You can export the results after you run a profile.

**Comments**

- View and add comments to the profile.
Column Profiling Process

As part of the column profiling process, you can choose to either include all the source columns for profiling or select specific columns. You can also accept the default profile options or configure the profile results, sampling, and drill-down options.

The following steps describe the column profiling process:

1. Select the data object you want to profile.
2. Determine whether you want to create a profile with default options or change the default profile options.
3. Choose where you want to save the profile.
4. Select the columns you want to profile.
5. Select the profile results option.
6. Choose the sampling options.
7. Choose the drill-down options.
8. Define a filter to determine the rows that the profile reads at run time.
9. Run the profile.

Column Profile Options

When you create a profile with the **Column Profiling** option, you can use the profile wizard to define filter and sampling options. These options determine how the profile reads rows from the data set.

The following figure shows a sample filter definition in a profile:

![Profile Filter definition](Profile_Filter.png)

Folder: Samples\ProfilingSamples; Object: Profile_Customer_Central_Filter; Version: PowerCenter Express 9.5.1; Image file name: Profile_Filter.png; Instructions: Double-click the Profile_Customer_Central_Filter profile in the Samples\ProfilingSamples folder. Click the Definition tab in the editor and then click Filters; Rajesh-03/28/13

After you complete the steps in the profile wizard, you can add a rule to the profile. The rule can have the business logic to perform data transformation operations on the data before column profiling.
The following figure shows a rule titled Rule_FullName that merges the LastName and FirstName columns into the Fullname column:

Folder: Samples\ProfilingSamples; Object: Profile_Customer_Central_Filter; Version: PowerCenter Express 9.5.1; Image file name: Profile_Rule.png; Instructions: Double-click the Profile_Customers_Rules profile in the Samples\ProfilingSamples folder. Click the Rules tab in the editor, Select Rule_FullName, and then click Edit.; Rajesh-03/28/13

Rules

Create and apply rules within profiles. A rule is business logic that defines conditions applied to data when you run a profile. Use rules to further validate the data in a profile and to measure data quality progress.

You can add a rule after you create a profile. Add rules to a profile by selecting a reusable rule. To create a rule, validate a mapplet as a rule.

Creating a Rule in Informatica Developer

You need to validate a mapplet as a rule to create a rule in the Developer tool.

Create a mapplet in the Developer tool.
1. Right-click the mapplet editor.
2. Select Validate As > Rule.

Applying a Rule

You can add a rule to a saved column profile.

1. Browse the Object Explorer view and find the profile you need.
2. Right-click the profile and select Open.
   The profile opens in the editor.
3. Click the Definition tab, and select Rules.
4. Click **Add**.
   The **Apply Rule** dialog box opens.

5. Click **Browse** to find the rule you want to apply.
   Select a rule from a repository project, and click **OK**.

6. Click the **Value** column under **Input Values** to select an input port for the rule.

7. Optionally, click the **Value** column under **Output Values** to edit the name of the rule output port.
   The rule appears in the **Definition** tab.

**Guidelines for Rules**

You can create reusable rules from mapplets in the Developer tool. You can use these rules in profiles to validate source data.

A rule must meet the following requirements:

- It must contain an Input and Output transformation. You cannot use data sources in a rule.
- It can contain Expression transformations, Lookup transformations, and passive transformations. It cannot contain any other type of transformation. For example, a rule cannot contain a Sorter transformation as it is an active transformation.
- It does not specify cardinality between input groups.

**Filtering Options**

You can add filters to determine the rows that a column profile uses when performing profiling operations. The profile does not process rows that do not meet the filter criteria.

1. Create or open a column profile.
2. Select the **Filter** view.
3. Click **Add**.
4. Select a filter type and click **Next**.
5. Enter a name for the filter. Optionally, enter a text description of the filter.
6. Select **Set as Active** to apply the filter to the profile. Click **Next**.
7. Define the filter criteria.
8. Click **Finish**.

**Sampling Properties**

Configure the sampling properties to determine the number of rows that the profile reads during a profiling operation.

The following table describes the sampling properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Rows</td>
<td>Reads all rows from the source. Default is enabled.</td>
</tr>
<tr>
<td>First</td>
<td>Reads from the first row up to the row you specify.</td>
</tr>
</tbody>
</table>
Creating a Profile

You can create a profile for one or more columns in a data object and store the profile object in the Model repository.

1. In the **Object Explorer** view, select the data object you want to profile.
2. Click **File > New > Profile** to open the profile wizard.
3. Select **Profile** and click **Next**.
4. Enter a name for the profile and verify the project location. If required, browse to a new location.
5. Optionally, enter a text description of the profile.
6. Verify that the name of the data object you selected appears in the **Data Objects** section.
7. Click **Next**.
8. Configure the column profile options such as column selection, filter, sampling, and drilldown options.
9. Click **Finish**.

Column Profile Results

Column profile analysis provides information about data quality by highlighting value frequencies, patterns and statistics of data.

Column profiling analysis generates the following profile results:

- Percentage and count statistics for unique and null values
- Inferred datatypes
- The datatype that the data source declares for the data
- The maximum and minimum values
- The date and time of the most recent profile run
- Percentage and count statistics for each unique data element in a column
- Percentage and count statistics for each unique character pattern in a column
The following figure shows the column profile results:

Folder: Samples\ProfilingSamples; Object: Profile_Customer_Central; Version: PowerCenter Express 9.5.1; Image file name: ColumnProfileResults.png; Instructions: Right-click the Profile_Customer_Central profile in the Samples\ProfilingSamples folder and select Run Profile. Open the profile and click the Results tab in the editor. Select CUSTOMER_NO under the Column header; Rajesh-03/28/13

**Column Value Properties**

Column value properties show the values in the profiled columns and the frequency with which each value appears in each column. The frequencies are shown as a number, a percentage, and a bar chart.

To view column value properties, select Values from the Show list. Double-click a column value to drill-down to the rows that contain the value.

The following table describes the properties for column values:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values</td>
<td>List of all values for the column in the profile.</td>
</tr>
<tr>
<td>Frequency</td>
<td>Number of times a value appears in a column.</td>
</tr>
<tr>
<td>Percent</td>
<td>Number of times a value appears in a column, expressed as a percentage of all values in the column.</td>
</tr>
<tr>
<td>Chart</td>
<td>Bar chart for the percentage.</td>
</tr>
</tbody>
</table>

**Column Pattern Properties**

Column pattern properties show the patterns of data in the profiled columns and the frequency with which the patterns appear in each column. The patterns are shown as a number, a percentage, and a bar chart.

To view pattern information, select Patterns from the Show list. Double-click a pattern to drill-down to the rows that contain the pattern.

The following table describes the properties for column value patterns:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patterns</td>
<td>Pattern for the selected column.</td>
</tr>
<tr>
<td>Frequency</td>
<td>Number of times a pattern appears in a column.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Percent</td>
<td>Number of times a pattern appears in a column, expressed as a percentage of all values in the column.</td>
</tr>
<tr>
<td>Chart</td>
<td>Bar chart for the percentage.</td>
</tr>
</tbody>
</table>

**Column Statistics Properties**

Column statistics include properties, such as maximum and minimum lengths of values and first and last values.

To view statistical information, select Statistics from the Show list.

The following table describes the column statistics properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Length</td>
<td>Length of the longest value in the column.</td>
</tr>
<tr>
<td>Minimum Length</td>
<td>Length of the shortest value in the column.</td>
</tr>
<tr>
<td>Bottom</td>
<td>Last five values in the column.</td>
</tr>
<tr>
<td>Top</td>
<td>First five values in the column.</td>
</tr>
<tr>
<td>Sum</td>
<td>Sum of all values in the column with a numeric datatype.</td>
</tr>
</tbody>
</table>

**Note:** The profile also displays average and standard deviation statistics for columns of type Integer.

**Column Datatype Properties**

Column datatypes include all the inferred datatypes for each column in the profile results.

To view datatype information, select Datatypes from the Show list. Double-click a datatype to drill-down to the rows that contain the datatype.

The following table describes the properties for the column datatypes:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datatype</td>
<td>List of all the inferred datatypes for the column in the profile.</td>
</tr>
<tr>
<td>Frequency</td>
<td>Number of times a datatype appears for a column, expressed as a number.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>% Conformance</td>
<td>Percentage that a datatype appears for a column.</td>
</tr>
<tr>
<td>Status</td>
<td>Indicates the status of the datatype. The statuses are Inferred, Approved, or Rejected.</td>
</tr>
<tr>
<td>Inferred</td>
<td>Indicates the datatype of the column that the Developer tool inferred.</td>
</tr>
<tr>
<td>Approved</td>
<td>Indicates an approved datatype for the column. When you approve a datatype, you commit the datatype to the Model repository.</td>
</tr>
<tr>
<td>Rejected</td>
<td>Indicates a rejected datatype for the column.</td>
</tr>
</tbody>
</table>

Curation in Informatica Developer

Curation is the process of validating and managing discovered metadata of a data source so that the metadata is fit for use and reporting. When you curate metadata in the Developer tool, you can approve, reject, and reset the inferred datatypes or data domains in profile results.

You can approve one datatype or data domain for a column. You can hide the rejected datatypes or data domains for a column. After you approve or reject an inferred datatype or data domain, you can reset the datatype or data domain to restore the inferred status.

Approving Datatypes in Informatica Developer

The profile results include the inferred datatypes, frequency, percentage of conformance, and the inference status for each column in the data source. You can choose and approve a single datatype for each column in the Analyst tool.

1. In the **Object Explorer** view, select and open a profile.
2. Verify that you are in the **Results** tab.
3. In the **Column Profiling** view, select a column to view the value frequencies, patterns, datatypes, and statistics in the right panel.
4. Under the **Details** panel, select **Datatypes** from the **Show** list.
   - The inferred datatypes for the column appear.
5. Right-click the column that you want to approve and click **Approve**.
   - The status of the datatype changes to **Approved**.
6. To restore the inferred status of the datatype, right-click the datatype and click **Reset**.

Rejecting Datatypes in Informatica Developer

When you view the profile results, the Developer tool displays inferred datatypes by default. You can reject inferred or approved datatypes. You can choose to show or hide the rejected datatypes.
1. In the **Object Explorer** view, select a profile.
2. Double-click the profile to open it.
   The profile opens in a tab.
3. In the **Column Profiling** view, select a row.
4. To reject inferred column datatypes, select the **Datatypes** view in the right panel. Select the inferred datatype that you want to reject, right-click the row, and then select **Reject**.
   The Developer tool greys out the rejected datatype in the list of datatypes.
5. To hide the rejected datatypes, right-click the row and select **Hide Rejected**.
6. To view the rejected datatypes, right-click one of the rows, and then select **Show Rejected**.

### Profile Results Export

You can export column profile results to a CSV file or Microsoft Excel file based on whether you choose a part of the profile results or the complete results summary. When you export the complete profile results summary, the Developer tool saves the information to multiple worksheets in a Microsoft Excel file.

The following table describes the information that appears on each worksheet in the export file:

<table>
<thead>
<tr>
<th>Tab</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column Profile</td>
<td>Column profile summary information including column names, rule names, number of unique values, number of null values, inferred datatypes, and date and time of the last profile run.</td>
</tr>
<tr>
<td>Values</td>
<td>Values for the columns and rules and the frequency in which the values appear for each column.</td>
</tr>
<tr>
<td>Patterns</td>
<td>Value patterns for the columns and rules you ran the profile on and the frequency in which the patterns appear.</td>
</tr>
<tr>
<td>Data Types</td>
<td>All the datatypes for the columns, frequency of each datatype, percentage value, and status of the datatype, such as Inferred, Approved, or Rejected.</td>
</tr>
<tr>
<td>Statistics</td>
<td>Statistics about each column and rule. Examples are average, length, top values, bottom values, and standard deviation.</td>
</tr>
<tr>
<td>Properties</td>
<td>Properties information including profile name, type, sampling policy, and row count.</td>
</tr>
</tbody>
</table>

### Exporting Profile Results from Informatica Developer

You can export column profile results to a .csv file or Microsoft Excel file. When you export the profile results to a Microsoft Excel file, the Developer tool saves the information to an .xlsx file.
1. In the **Object Explorer** view, open a profile.
2. Optionally, run the profile to update the profile results.
3. Select the **Results** view.
4. Select a column.
5. Under **Details**, select **Values**, **Patterns**, or **Datatypes** and click the **Export** icon.
   
   The **Export data to a file** dialog box opens.
6. Accept or change the default file name.
7. Select the type of data to export. You can select **Values for the selected column**, **Patterns for the selected column**, **Datatypes for the selected column**, or **All (Summary, Values, Patterns, Datatypes, Statistics, Properties)**.
8. Click **Browse** to select a location and save the file locally in your computer.
9. If you do not want to export field names as the first row, clear the **Export field names as first row** check box.
10. Click **OK**.

**Mapplet and Mapping Profiles**

You can define a column profile for an object in a mapplet or mapping. Run a profile on a mapplet or a mapping object when you want to verify the design of the mapping or mapplet without saving the profile results. You can also generate a mapping from a profile.

**Generating a Mapping from a Profile**

You can create a mapping object from a profile. Use the mapping object you create to develop a valid mapping. The mapping you create has a data source based on the profiled object and can contain transformations based on profile rule logic. After you create the mapping, add objects to complete it.

1. In the **Object Explorer** view, find the profile on which to create the mapping.
2. Right-click the profile name and select **Generate Mapping**.

   The **Generate Mapping** dialog box displays.
3. Enter a mapping name. Optionally, enter a description for the mapping.
4. Confirm the folder location for the mapping.
   
   By default, the Developer tool creates the mapping in the **Mappings** folder in the same project as the profile. Click **Browse** to select a different location for the mapping.
5. Confirm the profile definition that the Developer tool uses to create the mapping. To use another profile, click **Select Profile**.
6. Click **Finish**.

   The mapping appears in the **Object Explorer**.

Add objects to the mapping to complete it.
Running a Profile on a Mapplet or Mapping Object

When you run a profile on a mapplet or mapping object, the profile runs on all data columns and enables drill-down operations on the data. You can run a profile on a mapplet or mapping object with multiple output ports.

The profile traces the source data through the mapping to the output ports of the object you selected. The profile analyzes the data that would appear on those ports if you ran the mapping.

1. Open a mapplet or mapping.
2. Verify that the mapplet or mapping is valid.
3. Right-click a data object or transformation and select **Profile Now**.
   
   If the transformation has multiple output groups, the **Select Output Group** dialog box appears. If the transformation has a single output group, the profile results appear on the **Results** tab of the profile.
4. If the transformation has multiple output groups, select the output groups as necessary.
5. Click **OK**.
   
   The profile results appear in the **Results** tab of the profile.

Comparing Profiles for Mapping or Mapplet Objects

You can create a profile that analyzes two objects in a mapplet or mapping and compares the results of the column profiles for those objects.

Like profiles of single mapping or mapplet objects, profile comparisons run on all data columns.

1. Open a mapplet or mapping.
2. Verify that the mapplet or mapping is valid.
3. Press the **CTRL** key and click two objects in the editor.
4. Right-click one of the objects and select **Compare Profiles**.
5. Optionally, configure the profile comparison to match columns from one object to the other object.
6. Optionally, match columns by clicking a column in one object and dragging it onto a column in the other object.
7. Optionally, choose whether the profile analyzes all columns or matched columns only.
8. Click **OK**.
Logical View of Data Overview

A logical view of data is a representation of data that resides in an enterprise. A logical view of data includes a logical data model, logical data objects, and logical data object mappings.

With a logical view of data, you can achieve the following goals:

- Use common data models across an enterprise so that you do not have to redefine data to meet different business needs. It also means if there is a change in data attributes, you can apply this change one time and use one mapping to make this change to all databases that use this data.

- Find relevant sources of data and present the data in a single view. Data resides in various places in an enterprise, such as relational databases and flat files. You can access all data sources and present the data in one view.

- Expose logical data as relational tables to promote reuse.
Logical Data Object Model Example

Create a logical data object model to describe the representation of logical entities in an enterprise. For example, create a logical data object model to present account data from disparate sources in a single view.

American Bank acquires California Bank. After the acquisition, American Bank has the following goals:

- Present data from both banks in a business intelligence report, such as a report on the top 10 customers.
- Consolidate data from both banks into a central data warehouse.

Traditionally, American Bank would consolidate the data into a central data warehouse in a development environment, verify the data, and move the data warehouse to a production environment. This process might take several months or longer. The bank could then run business intelligence reports on the data warehouse in the production environment.

A developer at American Bank can use the Developer tool to create a model of customer, account, branch, and other data in the enterprise. The developer can link the relational sources of American Bank and California bank to a single view of the customer. The developer can then make the data available for business intelligence reports before creating a central data warehouse.

Developing a Logical View of Data

Develop a logical view of data to represent how an enterprise accesses data and uses data.

Before you develop a logical view of data, you can define the physical data objects that you want to use in a logical data object mapping. You can also profile the physical data sources to analyze data quality.

1. Create or import a logical data model.
2. Optionally, add logical data objects to the logical data object model and define relationships between objects.
3. Create a logical data object mapping to read data from a logical data object or write data to a logical data object. A logical data object mapping can contain transformation logic to transform the data.
4. View the output of the logical data object mapping.

Logical Data Object Models

A logical data object model describes the structure and use of data in an enterprise. The model contains logical data objects and defines relationships between them.

Define a logical data object model to create a unified model of data in an enterprise. The data in an enterprise might reside in multiple disparate source systems such as relational databases and flat files. A logical data object model represents the data from the perspective of the business regardless of the source systems. Create a logical data object model to study data, describe data attributes, and define the relationships among attributes.

For example, customer account data from American Bank resides in an Oracle database, and customer account data from California Banks resides in an IBM DB2 database. You want to create a unified model of customer accounts that defines the relationship between customers and accounts. Create a logical data object model to define the relationship.

You can import a logical data object model from a modeling tool. You can also import a logical data object model from an XSD file that you created in a modeling tool. Or, you can manually create a logical data object model in the Developer tool.

You add a logical data object model to a project or folder and store it in the Model repository.

Creating a Logical Data Object Model

Create a logical data object model to define the structure and use of data in an enterprise. When you create a logical data object model, you can add logical data objects. You associate a physical data object with each logical data object. The Developer tool creates a logical data object read mapping for each logical data object in the model.

1. Select a project or folder in the Object Explorer view.
2. Click File > New > Logical Data Object Model.
   The New dialog box appears.
3. Select Logical Data Object Model and click Next.
   The New Logical Data Object Model dialog box appears.
4. Enter a name for the logical data object model.
5. To create logical data objects, click Next. To create an empty logical data object model, click Finish.
   If you click Next, the Developer tool prompts you to add logical data objects to the model.
6. To create a logical data object, click the New button.
   The Developer tool adds a logical data object to the list.
7. Enter a name in the Name column.
8. Optionally, click the Open button in the Data Object column to associate a physical data object with the logical data object.
   The Select a Data Object dialog box appears.
9. Select a physical data object and click OK.
10. Repeat steps 6 through 9 to add logical data objects.
11. Click **Finish**.
   
The logical data object model opens in the editor.

**Importing a Logical Data Object Model from a Modeling Tool**

You can import a logical data object model from a modeling tool or an XSD file. Import a logical data object model to use an existing model of the structure and data in an enterprise.

1. Select the project or folder to which you want to import the logical data object model.
2. Click **File > New > Logical Data Object Model**.
   
The **New Logical Data Object Model** dialog box appears.
3. Select **Logical Data Object Model from Data Model**.
4. Click **Next**.
5. In the Model Type field, select the modeling tool from which you want to import the logical data object model.
6. Enter a name for the logical data object model.
7. Click **Browse** to select the location of the logical data object model.
8. Click **Next**.
9. Browse to the file that you want to import, select the file, and click **Open**.
10. Configure the import properties.
11. Click **Next**.
12. Add logical data objects to the logical data object model.
13. Click **Finish**.
   
The logical data objects appear in the editor.
Logical Data Object Model Properties

When you import a logical data object model from a modeling tool, provide the properties associated with the tool.

CA ERwin Data Modeler Import Properties

Configure the import properties when you import a logical data object model from CA ERwin Data Modeler. The following table describes the properties to configure when you import a model from CA ERwin Data Modeler:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import UDPs</td>
<td>Specifies how to import user-defined properties.</td>
</tr>
<tr>
<td></td>
<td>Select one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- As metadata. Import an explicit value as the property value object. Explicit values are not exported.</td>
</tr>
<tr>
<td></td>
<td>- As metadata, migrate default values. Import explicit and implicit values as property value objects.</td>
</tr>
<tr>
<td></td>
<td>- In description, migrate default values. Append the property name and value, even if implicit, to the object description property.</td>
</tr>
<tr>
<td></td>
<td>- Both, migrate default values. Import the UDP value, even if implicit, both as metadata and in the object's description.</td>
</tr>
<tr>
<td></td>
<td>Default is As metadata.</td>
</tr>
<tr>
<td>Import relationship name</td>
<td>Specifies how to import the relationship names from ERwin.</td>
</tr>
<tr>
<td></td>
<td>Select one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- From relationship name</td>
</tr>
<tr>
<td></td>
<td>- From relationship description</td>
</tr>
<tr>
<td></td>
<td>Default is From relationship name.</td>
</tr>
<tr>
<td>Import IDs</td>
<td>Specifies whether to set the unique ID of the object as the NativeId property.</td>
</tr>
<tr>
<td>Import subject areas</td>
<td>Specifies how to import the subject areas from ERwin.</td>
</tr>
<tr>
<td></td>
<td>Select one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- As diagrams</td>
</tr>
<tr>
<td></td>
<td>- As packages and diagrams</td>
</tr>
<tr>
<td></td>
<td>- As packages and diagrams, assuming one subject area for each entity</td>
</tr>
<tr>
<td></td>
<td>- Do not import subject areas</td>
</tr>
<tr>
<td></td>
<td>Default is As diagrams.</td>
</tr>
<tr>
<td>Import column order form</td>
<td>Specifies how to import the position of columns in tables.</td>
</tr>
<tr>
<td></td>
<td>Select one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- Column order. Order of the columns displayed in the ERwin physical view.</td>
</tr>
<tr>
<td></td>
<td>- Physical order. Order of the columns in the database, as generated in the SQL DDL.</td>
</tr>
<tr>
<td></td>
<td>Default is Physical order.</td>
</tr>
<tr>
<td>Import owner schemas</td>
<td>Specifies whether to import owner schemas.</td>
</tr>
</tbody>
</table>
IBM Cognos Business Intelligence Reporting - Framework Manager Import Properties

Configure the import properties when you import a logical data object model from IBM Cognos Business Intelligence Reporting - Framework Manager.

The following table describes the properties to configure when you import a model from IBM Cognos Business Intelligence Reporting - Framework Manager:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Folder Representation</td>
<td>Specifies how to represent folders from the Framework Manager. Select one of the following options: Ignore, Ignore folders, Flat, Represent folders as diagrams but do not preserve hierarchy, Hierarchial, Represent folders as diagrams and preserve hierarchy. Default is Ignore.</td>
</tr>
<tr>
<td>Package Representation</td>
<td>Specifies how to represent packages from Cognos Framework Manager. Select one of the following options: Ignore, Ignore subject areas, Subject Areas, Represent packages as subject areas, Model, Represent the package as the model. Default is Ignore.</td>
</tr>
<tr>
<td>Reverse engineer relationships</td>
<td>Specifies whether the Developer tool computes the relationship between two dbQueries as referential integrity constraints.</td>
</tr>
<tr>
<td>Tables design level</td>
<td>Specifies how to control the design level of the imported tables: Select one of the following options: Logical and physical, The tables appear in both the logical view and in the physical view of the model, Physical, The tables appear only in the physical view of the model. Default is Physical.</td>
</tr>
<tr>
<td>Ignore usage property</td>
<td>Specify whether the usage property of a queryItem should be used.</td>
</tr>
</tbody>
</table>
SAP BusinessObjects Designer Import Properties

Configure the import properties when you import a logical data object model from SAP BusinessObjects Designer.

The following table describes the properties to configure when you import a model from SAP BusinessObjects Designer:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
<td>Name of the BusinessObjects repository. For BusinessObjects versions 11.x and 12.x (XI), enter the name of the Central Management Server. For BusinessObjects version 5.x and 6.x, enter name of the repository defined by the Supervisor application.</td>
</tr>
<tr>
<td>Authentication mode</td>
<td>Login authentication mode. This parameter is applicable to SAP BusinessObjects Designer 11.0 and later. Select one of the following authentication modes: - Enterprise. Business Objects Enterprise login - LDAP. LDAP server authentication - Windows AD. Windows Active Directory server authentication - Windows NT. Windows NT domain server authentication - Standalone. Standalone authentication Default is Enterprise.</td>
</tr>
<tr>
<td>User name</td>
<td>User name in the BusinessObjects server. For version 11.x and 12.x (XI), you need to be a member of BusinessObjects groups.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for the BusinessObjects server.</td>
</tr>
<tr>
<td>Silent execution</td>
<td>Specifies whether to execute in interactive or silent mode. Default is Silent.</td>
</tr>
<tr>
<td>Close after execution</td>
<td>Specify whether to close BusinessObjects after the Developer Tool completes the model import.</td>
</tr>
<tr>
<td>Table design level</td>
<td>Specifies the design level of the imported tables. Select one of the following options: - Logical and physical. The tables appear both in the logical view and in the physical view of the model. - Physical. The tables appear both in the physical view of the model. Default is Physical.</td>
</tr>
<tr>
<td>Transform Joins to Foreign Keys</td>
<td>Transforms simple SQL joins in the model into foreign key relationships. Select the parameter if you want to export the model to a tool that only supports structural relational metadata, such as a database design tool.</td>
</tr>
</tbody>
</table>
### Class representation

Specifies how to import the tree structure of classes and sub-classes. The Developer Tool imports each class as a dimension as defined by the CWM OLAP standard. The Developer Tool also imports classes and sub-classes as a tree of packages as defined by the CWM and UML standards.

Select one of the following options:
- As a flat structure. The Developer tool does not create package.
- As a simplified tree structure. The Developer tool creates package for each class with sub-classes.
- As a full tree structure. The Developer tool creates a package for each class.

Default is As a flat structure.

### Include List of Values

Controls how the Developer tool imports the list of values associated with objects.

### Dimensional properties transformation

Specifies how to transfer the dimension name, description, and role to the underlying table and the attribute name, description, and datatype to the underlying column.

Select one of the following options:
- Disabled. No property transfer occurs.
- Enabled. Property transfer occurs where there are direct matches between the dimensional objects and the relational objects. The Developer tool migrates the dimension names to the relational names.
- Enabled (preserve names). Property transfer occurs where there are direct matches between the dimensional objects and the relational objects. The Developer tool preserves the relational names.

Default is Disabled.

## Sybase PowerDesigner CDM Import Properties

Configure the import properties when you import a logical data object model from Sybase PowerDesigner CDM.

The following table describes the properties to configure when you import a model from Sybase PowerDesigner CDM:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import UDPs</td>
<td>Specifies how to import user-defined properties.</td>
</tr>
<tr>
<td></td>
<td>Select one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- As metadata. Import an explicit value as the property value object.</td>
</tr>
<tr>
<td></td>
<td>Explicit values are not exported.</td>
</tr>
<tr>
<td></td>
<td>- As metadata, migrate default values. Import explicit and implicit values</td>
</tr>
<tr>
<td></td>
<td>as property value objects.</td>
</tr>
<tr>
<td></td>
<td>- In description, migrate default values. Append the property name and value,</td>
</tr>
<tr>
<td></td>
<td>even if implicit, to the object description property.</td>
</tr>
<tr>
<td></td>
<td>- Both, migrate default values. Import the UDP value, even if implicit, both as</td>
</tr>
<tr>
<td></td>
<td>metadata and in the object's description.</td>
</tr>
<tr>
<td></td>
<td>Default is As metadata.</td>
</tr>
<tr>
<td>Import Association Classes</td>
<td>Specifies whether the Developer tool should import association classes.</td>
</tr>
</tbody>
</table>
Sybase PowerDesigner OOM 9.x to 15.x Import Properties

Configure the import properties when you import a logical data object model from Sybase PowerDesigner OOM 9.x to 15.x.

When you import a logical data object model from Sybase PowerDesigner OOM, the Developer tool imports the classes and attributes but leaves out other entities. To import a logical data object model, export the model from Sybase PowerDesigner in the UML 1.3 - XMI 1.0 XML format.

The following table describes the properties to configure when you import a model from Sybase PowerDesigner OOM:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import IDs</td>
<td>Specifies whether to set the unique ID of the object as the NativeId property.</td>
</tr>
<tr>
<td>Append volume info</td>
<td>Import and append the number of occurrences information to the description property.</td>
</tr>
<tr>
<td>Remove text formatting</td>
<td>Specifies whether to remove or keep rich text formatting. Select this option if the model was generated by PowerDesigner 7.0 or 7.5 Clear this option if the model was generated by PowerDesigner 8.0 or greater.</td>
</tr>
</tbody>
</table>

**Property**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Tool</td>
<td>Specifies which tool generated the model you want to import. Select one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- Auto Detect. The Developer tool detects which tool generated the file.</td>
</tr>
<tr>
<td></td>
<td>- OMG XMI. The file conforms to the OMG XMI 1.0 standard DTDs.</td>
</tr>
<tr>
<td></td>
<td>- Argo/UML 0.7. The file was generated by Argo/UML 0.7.0 or earlier.</td>
</tr>
<tr>
<td></td>
<td>- Argo/UML 0.8. The file was generated by Argo/UML 0.7.1 or later.</td>
</tr>
<tr>
<td></td>
<td>- XMI Toolkit. The file was generated by IBM XMI Toolkit.</td>
</tr>
<tr>
<td></td>
<td>- XMI Interchange. The file was generated by Unisys Rose XMI Interchange.</td>
</tr>
<tr>
<td></td>
<td>- Rose UML. The file was generated by Unisys Rose UML.</td>
</tr>
<tr>
<td></td>
<td>- Visio UML. The file was generated by Microsoft Visio Professional 2002 and Visio for Enterprise Architects using UML to XMI Export.</td>
</tr>
<tr>
<td></td>
<td>- PowerDesigner UML. The file was generated by Sybase PowerDesigner using XMI Export.</td>
</tr>
<tr>
<td></td>
<td>- Component Modeler. The file was generated by CA AllFusion Component Modeler using XMI Export.</td>
</tr>
<tr>
<td></td>
<td>- Netbeans XMI Writer. The file was generated by one of applications using Netbeans XMI Writer such as Poseidon.</td>
</tr>
<tr>
<td></td>
<td>- Embarcadero Describe. The file was generated by Embarcadero Describe. Default is Auto Detect.</td>
</tr>
<tr>
<td>Auto Correct</td>
<td>Fix and import an incomplete or incorrect model in the XML file.</td>
</tr>
<tr>
<td>Model Filter</td>
<td>Model to import if the XML file contains more than one model. Use a comma to separate multiple models.</td>
</tr>
<tr>
<td>Top Package</td>
<td>The top-level package in the model.</td>
</tr>
<tr>
<td>Import UUIDs</td>
<td>Import UUIDs as NativeId.</td>
</tr>
</tbody>
</table>
Sybase PowerDesigner PDM Import Properties

Configure the import properties when you import a logical data object model from Sybase PowerDesigner PDM.

The following table describes the properties to configure when you import a model from Sybase PowerDesigner PDM:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import UDPs</td>
<td>Specifies how to import user-defined properties. Select one of the following options:</td>
</tr>
<tr>
<td></td>
<td>- As metadata. Import an explicit value as the property value object. Explicit values are not exported.</td>
</tr>
<tr>
<td></td>
<td>- As metadata, migrate default values. Import explicit and implicit values as property value objects.</td>
</tr>
<tr>
<td></td>
<td>- In description, migrate default values. Append the property name and value, even if implicit, to the object description property.</td>
</tr>
<tr>
<td></td>
<td>- Both, migrate default values. Import the UDP value, even if implicit, both as metadata and in the object's description.</td>
</tr>
<tr>
<td></td>
<td>Default is As metadata.</td>
</tr>
<tr>
<td>Import IDs</td>
<td>Specifies whether to set the unique id of the object as the NativeId property.</td>
</tr>
<tr>
<td>Append volumetric information</td>
<td>Import and append the number of occurrences information to the description property.</td>
</tr>
<tr>
<td>to the description field</td>
<td></td>
</tr>
<tr>
<td>Remove text formatting</td>
<td>Specifies whether to remove or keep rich text formatting. Select this option if the model was generated by PowerDesigner 7.0 or 7.5</td>
</tr>
<tr>
<td></td>
<td>Clear this option if the model was generated by PowerDesigner 8.0 or greater.</td>
</tr>
</tbody>
</table>
XSD Import Properties

You can import logical data object models from an XSD file exported by a modeling tool.

The following table describes the properties to configure when you import a model from an XSD file:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elements content name</td>
<td>Attribute to hold the textual content like #PCDATA in the XSD file. Default is As metadata.</td>
</tr>
</tbody>
</table>
| Collapse Level                | Specifies when to collapse a class. The value you select determines whether the Developer tool imports all or some of the elements and attributes in the XSD file. Select one of the following options:  
  - None. Every XSD element becomes a class and every XSD attribute becomes an attribute.  
  - Empty. Only empty classes collapse into the parent classes.  
  - Single Attribute. Only XSD elements with a single attribute and no children collapse into the parent class.  
  - No Children. Any XSD element that has no child element collapse into the parent class.  
  - All. All collapsible XSD elements collapse into the parent class. Default is All. |
| Collapse Star                 | Specifies whether the Developer tool should collapse XML elements with an incoming xlink into the parent class. |
| Class Type                    | Specifies whether the Developer tool should create a class type an element collapses into the parent element. |
| Any                           | Specifies whether to create a class or entity for the 'xs:any' pseudo-element. |
| Generate IDs                  | Specifies whether to generate additional attributes to create primary and foreign keys. By default, the Developer tool does not generate additional attributes. |
| Import substitutionGroup as   | Specifies how to represent inheritance. Select one of the following options:  
  - Generalization. Represents inheritance as generalization.  
  - Roll down. Duplicate inherited attributes in the subclass. Default is Roll down. |
| Include Path                  | Path to the directory that contains the included schema files, if any. |
| UDP namespace                 | Namespace that contains attributes to be imported as user-defined properties. |

Logical Data Objects

A logical data object is an object in a logical data object model that describes a logical entity in an enterprise. It has attributes, keys, and it describes relationships between attributes.
You include logical data objects that relate to each other in a data object model. For example, the logical data objects Customer and Account appear in a logical data object model for a national bank. The logical data object model describes the relationship between customers and accounts.

In the model, the logical data object Account includes the attribute Account_Number. Account_Number is a primary key, because it uniquely identifies an account. Account has a relationship with the logical data object Customer, because the Customer data object needs to reference the account for each customer.

You can drag a physical data object into the logical data object model editor to create a logical data object. Or, you can create a logical data object and define the attributes and keys.

Logical Data Object Properties

A logical data object contains properties that define the data object and its relationship to other logical data objects in a logical data object model.

The logical data object properties are on the tabs in the Properties view. The following figure shows the tabs of a logical data object:

The following table describes the tabs of a logical data object:

<table>
<thead>
<tr>
<th>Tab Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>Name and description of the logical data object.</td>
</tr>
<tr>
<td>Attributes</td>
<td>Comprise the structure of data in a logical data object.</td>
</tr>
<tr>
<td>Keys</td>
<td>One or more attributes in a logical data object can be primary keys or unique keys.</td>
</tr>
<tr>
<td>Relationships</td>
<td>Associations between logical data objects.</td>
</tr>
<tr>
<td>Access</td>
<td>Type of access for a logical data object and each attribute of the data object.</td>
</tr>
<tr>
<td>Mappings</td>
<td>Logical data object mappings associated with a logical data object.</td>
</tr>
</tbody>
</table>

Attribute Relationships

A relationship is an association between primary or foreign key attributes of one or more logical data objects.

You can define the following types of relationship between attributes:
Identifying

A relationship between two attributes where an attribute is identified through its association with another attribute.

For example, the relationship between the Branch_ID attribute of the logical data object Branch and the Branch_Location attribute of the logical data object Customer is identifying. This is because a branch ID is unique to a branch location.

Non-Identifying

A relationship between two attributes that identifies an attribute independently of the other attribute.

For example, the relationship between the Account_Type attribute of the Account logical data object and the Account_Number attribute of the Customer logical data object is non-identifying. This is because you can identify an account type without having to associate it with an account number.

When you define relationships, the logical data object model indicates an identifying relationship as a solid line between attributes. It indicates a non-identifying relationship as a dotted line between attributes.

Creating a Logical Data Object

You can create a logical data object in a logical data object model to define a logical entity in an enterprise.

1. Click File > New > Logical Data Object.
2. Enter a logical data object name.
3. Select the logical data object model for the logical data object and click Finish.
   
   The logical data object appears in the logical data object model editor.
4. Select the logical data object and click the Properties view.
5. On the General tab, optionally edit the logical data object name and description.
6. On the Attributes tab, create attributes and specify their datatype and precision.
7. On the **Keys** tab, optionally specify primary and unique keys for the data object.

8. On the **Relationships** tab, optionally create relationships between logical data objects.

9. On the **Access** tab, optionally edit the type of access for the logical data object and each attribute in the data object.

   Default is read only.
10. On the **Mappings** tab, optionally create a logical data object mapping.

![Screen capture of the Mappings tab](image)

### Logical Data Object Mappings

A logical data object mapping is a mapping that links a logical data object to one or more physical data objects. It can include transformation logic.

A logical data object mapping can be of the following types:

- Read
- Write

You can associate each logical data object with one logical data object read mapping or one logical data object write mapping.

### Logical Data Object Read Mappings

A logical data object read mapping contains one or more physical data objects as input and one logical data object as output. The mapping can contain transformation logic to transform the data.

It provides a way to access data without accessing the underlying data source. It also provides a way to have a single view of data coming from more than one source.

For example, American Bank has a logical data object model for customer accounts. The logical data object model contains a Customers logical data object.

American Bank wants to view customer data from two relational databases in the Customers logical data object. You can use a logical data object read mapping to perform this task and view the output in the **Data Viewer** view.

### Logical Data Object Write Mappings

A logical data object write mapping contains a logical data object as input. It provides a way to write to targets from a logical data object.

The mapping can contain transformation logic to transform the data. The mapping runs without accessing the underlying data target. It provides a single view of the transformed data without writing to the target.
Creating a Logical Data Object Mapping

You can create a logical data object mapping to link data from a physical data object to a logical data object and transform the data.

1. In the Data Object Explorer view, select the logical data object model that you want to add the mapping to.
2. Click File > New > Other.
3. Select Informatica > Data Objects > Data Object Mapping and click Next.
4. Select the logical data object you want to include in the mapping.
5. Select the mapping type.
6. Optionally, edit the mapping name.
7. Click Finish.
   The editor displays the logical data object as the mapping input or output, based on whether the mapping is a read or write mapping.
8. Drag one or more physical data objects to the mapping as read or write objects, based on whether the mapping is a read or write mapping.
9. Optionally, add transformations to the mapping.
10. Link ports in the mapping.
11. Right-click the mapping editor and click Validate to validate the mapping.
    Validation errors appear on the Validation Log view.
12. Fix validation errors and validate the mapping again.
13. Optionally, click the Data Viewer view and run the mapping.
    Results appear in the Output section.
Transformations Overview

A transformation is an object that generates, modifies, or passes data.

Informatica Developer provides a set of transformations that perform specific functions. For example, an Aggregator transformation performs calculations on groups of data.

Transformations in a mapping represent the operations that the Data Integration Service performs on the data. Data passes through transformation ports that you link in a mapping or mapplet.

Transformations can be active or passive. Transformations can be connected to the data flow, or they can be unconnected.

For more information, see the PowerCenter Express Transformation Guide.

Active Transformations

An active transformation changes the number of rows that pass through a transformation. Or, it changes the row type.

For example, the Filter transformation is active, because it removes rows that do not meet the filter condition. The Update Strategy transformation is active, because it flags rows for insert, delete, update, or reject.

You cannot connect multiple active transformations or an active and a passive transformation to the same downstream transformation or transformation input group, because the Data Integration Service might not be able to concatenate the rows passed by active transformations.

For example, one branch in a mapping contains an Update Strategy transformation that flags a row for delete. Another branch contains an Update Strategy transformation that flags a row for insert. If you connect these transformations to a single transformation input group, the Data Integration Service cannot combine the delete and insert operations for the row.
Passive Transformations

A passive transformation does not change the number of rows that pass through the transformation, and it maintains the row type.

You can connect multiple transformations to the same downstream transformation or transformation input group if all transformations in the upstream branches are passive. The transformation that originates the branch can be active or passive.

Unconnected Transformations

Transformations can be connected to the data flow, or they can be unconnected. An unconnected transformation is not connected to other transformations in the mapping. An unconnected transformation is called within another transformation, and returns a value to that transformation.

Transformation Descriptions

The Developer tool contains transformations that perform different data integration functions.

The following table describes each transformation:

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregator</td>
<td>Active/ Connected</td>
<td>Performs aggregate calculations.</td>
</tr>
<tr>
<td>Expression</td>
<td>Passive/ Connected</td>
<td>Calculates a value.</td>
</tr>
<tr>
<td>Filter</td>
<td>Active/ Connected</td>
<td>Filters data.</td>
</tr>
<tr>
<td>Java</td>
<td>Active or Passive/ Connected</td>
<td>Executes user logic coded in Java. The byte code for the user logic is stored in the repository.</td>
</tr>
<tr>
<td>Joiner</td>
<td>Active/ Connected</td>
<td>Joins data from different databases or flat file systems.</td>
</tr>
<tr>
<td>Lookup</td>
<td>Active or Passive/ Connected or Unconnected</td>
<td>Look up and return data from a flat file, logical data object, reference table, relational table, view, or synonym.</td>
</tr>
<tr>
<td>Normalizer</td>
<td>Active/ Connected</td>
<td>Processes source rows that contain multiple-occurring data and returns a target row for each instance of the multiple-occurring data.</td>
</tr>
<tr>
<td>Output</td>
<td>Passive/ Connected</td>
<td>Defines mapplet output rows.</td>
</tr>
<tr>
<td>Rank</td>
<td>Active/ Connected</td>
<td>Limits records to a top or bottom range.</td>
</tr>
<tr>
<td>Transformation</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Router</td>
<td>Active/Connected</td>
<td>Routes data into multiple transformations based on group conditions.</td>
</tr>
<tr>
<td>Sequence Generator</td>
<td>Passive/Connected</td>
<td>Generates a numeric sequence of values.</td>
</tr>
<tr>
<td>Sorter</td>
<td>Active/Connected</td>
<td>Sorts data based on a sort key.</td>
</tr>
<tr>
<td>SQL</td>
<td>Active or Passive/Connected</td>
<td>Executes SQL queries against a database.</td>
</tr>
<tr>
<td>Union</td>
<td>Active/Connected</td>
<td>Merges data from different databases or flat file systems.</td>
</tr>
<tr>
<td>Update Strategy</td>
<td>Active/Connected</td>
<td>Determines whether to insert, delete, update, or reject rows.</td>
</tr>
<tr>
<td>Web Service Consumer</td>
<td>Active/Connected</td>
<td>Connects to a web service as a web service client to access or transform data.</td>
</tr>
</tbody>
</table>

Developing a Transformation

When you build a mapping, you add transformations and configure them to handle data according to a business purpose.

Complete the following tasks to develop a transformation and incorporate it into a mapping:

1. Add a nonreusable transformation to a mapping or mapplet. Or, create a reusable transformation that you can add to multiple mappings or mapplets.
2. Configure the transformation. Each type of transformation has a unique set of options that you can configure.
3. If the transformation is reusable, add it to the mapping or mapplet.
4. Link the transformation to other objects in the mapping or mapplet.

   You drag ports from upstream objects to the transformation input ports. You drag output ports from the transformation to ports on downstream objects. Some transformations use predefined ports that you can select.

   **Note:** If you create a reusable transformation, you add the input and output ports you need before you link the transformation to other objects. You cannot add ports to the transformation instance on the mapplet or mapping canvas. To update the ports on a reusable transformation, open the transformation object from the repository project and add the ports.
Reusable Transformations

Reusable transformations are transformations that you can use in multiple mappings or mapplets.

For example, you might create an Expression transformation that calculates value-added tax for sales in Canada to analyze the cost of doing business in that country. Rather than perform the same work every time, you can create a reusable transformation. When you need to incorporate this transformation into a mapping, you add an instance of it to the mapping. If you change the definition of the transformation, all instances of it inherit the changes.

The Developer tool stores each reusable transformation as metadata separate from any mapping or mapplet that uses the transformation. It stores reusable transformations in a project or folder.

When you add instances of a reusable transformation to mappings, changes you make to the transformation might invalidate the mapping or generate unexpected data.

Reusable Transformation Instances and Inherited Changes

When you add a reusable transformation to a mapping or mapplet, you add an instance of the transformation. The definition of the transformation still exists outside the mapping or mapplet, while an instance of the transformation appears within the mapping or mapplet.

When you change the transformation, instances of the transformation reflect these changes. Instead of updating the same transformation in every mapping that uses it, you can update the reusable transformation one time, and all instances of the transformation inherit the change. Instances inherit changes to ports, expressions, properties, and the name of the transformation.

Editing a Reusable Transformation

When you edit a reusable transformation, all instances of that transformation inherit the changes. Some changes might invalidate the mappings that use the reusable transformation.

You can open the transformation in the editor to edit a reusable transformation. You cannot edit an instance of the transformation in a mapping. However, you can edit the transformation runtime properties.

If you make any of the following changes to a reusable transformation, mappings that use instances of it might not be valid:

- When you delete one or more ports in a transformation, you disconnect the instance from part or all of the data flow through the mapping.
- When you change a port datatype, you make it impossible to map data from that port to another port that uses an incompatible datatype.
- When you change a port name, expressions that refer to the port are no longer valid.
- When you enter an expression that is not valid in the reusable transformation, mappings that use the transformation are no longer valid. The Data Integration Service cannot run mappings that are not valid.

Expressions in Transformations

You can enter expressions in the Expression Editor in some transformations. Expressions modify data or test whether data matches conditions.
Create expressions that use transformation language functions. Transformation language functions are SQL-like functions that transform data.

Enter an expression in a port that uses the value of data from an input or input/output port. For example, you have a transformation with an input port IN_SALARY that contains the salaries of all the employees. You might use the values from the IN_SALARY column later in the mapping. You might also use the transformation to calculate the total and average salaries. The Developer tool requires you to create a separate output port for each calculated value.

The following table lists the transformations in which you can enter expressions:

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Expression</th>
<th>Return Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregator</td>
<td>Performs an aggregate calculation based on all data passed through the transformation. Alternatively, you can specify a filter for records in the aggregate calculation to exclude certain kinds of records. For example, you can find the total number and average salary of all employees in a branch office using this transformation.</td>
<td>Result of an aggregate calculation for a port.</td>
</tr>
<tr>
<td>Expression</td>
<td>Performs a calculation based on values within a single row. For example, based on the price and quantity of a particular item, you can calculate the total purchase price for that line item in an order.</td>
<td>Result of a row-level calculation for a port.</td>
</tr>
<tr>
<td>Filter</td>
<td>Specifies a condition used to filter rows passed through this transformation. For example, if you want to write customer data to the BAD_DEBT table for customers with outstanding balances, you could use the Filter transformation to filter customer data.</td>
<td>TRUE or FALSE, based on whether a row meets the specified condition. The Data Integration Service passes rows that return TRUE through this transformation. The transformation applies this value to each row that passes through it.</td>
</tr>
<tr>
<td>Joiner</td>
<td>Specifies an advanced condition used to join unsorted source data. For example, you can concatenate first name and last name master ports and then match them with the full name detail port.</td>
<td>TRUE or FALSE, based on whether the row meets the specified condition. Depending on the type of join selected, the Data Integration Service either adds the row to the result set or discards the row.</td>
</tr>
<tr>
<td>Rank</td>
<td>Sets the conditions for rows included in a rank. For example, you can rank the top 10 salespeople who are employed with the organization.</td>
<td>Result of a condition or calculation for a port.</td>
</tr>
<tr>
<td>Transformation</td>
<td>Expression</td>
<td>Return Value</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Router</td>
<td>Routes data into multiple transformations based on a group expression. For example, use this transformation to compare the salaries of employees at three different pay levels. You can do this by creating three groups in the Router transformation. For example, create one group expression for each salary range.</td>
<td>TRUE or FALSE, based on whether a row meets the specified group expression. The Data Integration Service passes rows that return TRUE through each user-defined group in this transformation. Rows that return FALSE pass through the default group.</td>
</tr>
<tr>
<td>Update Strategy</td>
<td>Flags a row for update, insert, delete, or reject. You use this transformation when you want to control updates to a target, based on some condition you apply. For example, you might use the Update Strategy transformation to flag all customer rows for update when the mailing address has changed. Or, you might flag all employee rows for reject for people who no longer work for the organization.</td>
<td>Numeric code for update, insert, delete, or reject. The transformation applies this value to each row passed through it.</td>
</tr>
</tbody>
</table>

### The Expression Editor

Use the **Expression Editor** to build SQL-like statements.

You can enter an expression manually or use the point-and-click method. Select functions, ports, variables, and operators from the point-and-click interface to minimize errors when you build expressions. The maximum number of characters you can include in an expression is 32,767.

### Port Names in an Expression

You can enter transformation port names in an expression.

For connected transformations, if you use port names in an expression, the Developer tool updates that expression when you change port names in the transformation. For example, you write an expression that determines the difference between two dates, Date_Promised and Date_Delivered. If you change the Date_Promised port name to Due_Date, the Developer tool changes the Date_Promised port name to Due_Date in the expression.

**Note**: You can propagate the name Due_Date to other non-reusable transformations that depend on this port in the mapping.

### Adding an Expression to a Port

You can add an expression to an output port.

1. In the transformation, select the port and open the **Expression Editor**.
2. Enter the expression.
   - Use the Functions and Ports tabs and the operator keys.
3. Optionally, add comments to the expression.
   - Use comment indicators -- or //.
4. Click the Validate button to validate the expression.
5. Click **OK**.
6. If the expression is not valid, fix the validation errors and validate the expression again.

7. When the expression is valid, click OK to close the Expression Editor.

Comments in an Expression

You can add comments to an expression to describe the expression or to specify a valid URL to access business documentation about the expression.

To add comments within the expression, use -- or // comment indicators.

Expression Validation

You need to validate an expression to run a mapping or preview mapplet output.

Use the Validate button in the Expression Editor to validate an expression. If you do not validate an expression, the Developer tool validates it when you close the Expression Editor. If the expression is invalid, the Developer tool displays a warning. You can save the invalid expression or modify it.

Creating a Transformation

You can create a reusable transformation to reuse in multiple mappings or mapplets. Or, you can create a non-reusable transformation to use one time in a mapping or mapplet.

To create a reusable transformation, click File > New > Transformation and complete the wizard.

To create a non-reusable transformation in a mapping or mapplet, select a transformation from the Transformation palette and drag the transformation to the editor.

Certain transformations require you to choose a mode or perform additional configuration when you create the transformation. For example, the Parser transformation requires that you choose either token parsing mode or pattern parsing mode when you create the transformation.

After you create a transformation, it appears in the editor. Some transformations contain predefined ports and groups. Other transformations are empty.
Viewing Data Overview

You can run a mapping, view profile results, view source data, view dependencies on an object, or preview data for a transformation.

Run a mapping to move output from sources to targets and transform data. You can run a mapping from the command line or from the Run dialog box. View profile results in the editor.

You view source data or preview data for a transformation in the Data Viewer view.

Note: The maximum number of rows you can preview in the Data Viewer is 100,000.

Before you can view data, you must select the default Data Integration Service. You can create configurations to control settings that the Developer tool applies when you view data.

When you view data in the Data Viewer view, you can export the data to a file. You can also access logs that show log events.

You can also view object dependencies when you view, modify, or delete Model repository objects. You can view object dependencies in the Object Dependencies view.

Configurations

A configuration is a group of settings that the Developer tool applies when you run a mapping or preview data.
A configuration controls settings such as the default Data Integration Service, number of rows to read from a source, default date/time format, and optimizer level. The configurations that you create apply to your installation of the Developer tool.

You can create the following configurations:

- Data viewer configurations. Control the settings the Developer tool applies when you preview output in the Data Viewer view.
- Mapping configurations. Control the settings the Developer tool applies when you run mappings through the Run Configurations dialog box or from the command line.

### Configuration Properties

The Developer tool applies configuration properties when you preview output or you run mappings. Set configuration properties for the Data Viewer view or mappings in the Run dialog box.

### Data Integration Service Properties

The Developer tool displays the Data Integration Service tab for data viewer and mapping configurations. The following table displays the properties that you configure for the Data Integration Service:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use default Data Integration Service</td>
<td>Uses the default Data Integration Service to run the mapping. Default is enabled.</td>
</tr>
<tr>
<td>Data Integration Service</td>
<td>Specifies the Data Integration Service that runs the mapping if you do not use the default Data Integration Service.</td>
</tr>
</tbody>
</table>

### Source Properties

The Developer tool displays the Source tab for data viewer and mapping configurations. The following table displays the properties that you configure for sources:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read all rows</td>
<td>Reads all rows from the source. Default is enabled.</td>
</tr>
<tr>
<td>Read up to how many rows</td>
<td>Specifies the maximum number of rows to read from the source if you do not read all rows. <strong>Note:</strong> If you enable the this option for a mapping that writes to a customized data object, the Data Integration Service does not truncate the target table before it writes to the target. Default is 1000.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Read all characters</td>
<td>Reads all characters in a column.</td>
</tr>
<tr>
<td></td>
<td>Default is disabled.</td>
</tr>
<tr>
<td>Read up to how many characters</td>
<td>Specifies the maximum number of characters to read in each column if you do not read all characters. The Data Integration Service ignores this property for SAP sources.</td>
</tr>
<tr>
<td></td>
<td>Default is 4000.</td>
</tr>
</tbody>
</table>

**Results Properties**

The Developer tool displays the **Results** tab for data viewer configurations.

The following table displays the properties that you configure for results in the **Data Viewer** view:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show all rows</td>
<td>Displays all rows in the <strong>Data Viewer</strong> view.</td>
</tr>
<tr>
<td></td>
<td>Default is disabled.</td>
</tr>
<tr>
<td>Show up to how many rows</td>
<td>Specifies the maximum number of rows to display if you do not display all rows.</td>
</tr>
<tr>
<td></td>
<td>Default is 1000.</td>
</tr>
<tr>
<td>Show all characters</td>
<td>Displays all characters in a column.</td>
</tr>
<tr>
<td></td>
<td>Default is disabled.</td>
</tr>
<tr>
<td>Show up to how many characters</td>
<td>Specifies the maximum number of characters to display in each column if you do not display all characters.</td>
</tr>
<tr>
<td></td>
<td>Default is 4000.</td>
</tr>
</tbody>
</table>
Advanced Properties

The Developer tool displays the Advanced tab for data viewer and mapping configurations.

The following table displays the advanced properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default date time format</td>
<td>Date/time format the Data Integration Services uses when the mapping converts strings to dates. Default is MM/DD/YYYY HH24:MI:SS.</td>
</tr>
</tbody>
</table>
| Override tracing level        | Overrides the tracing level for each transformation in the mapping. The tracing level determines the amount of information that the Data Integration Service sends to the mapping log files. Choose one of the following tracing levels:  
- None. The Data Integration Service uses the tracing levels set in the mapping.  
- Terse. The Data Integration Service logs initialization information, error messages, and notification of rejected data.  
- Normal. The Data Integration Service logs initialization and status information, errors encountered, and skipped rows due to transformation row errors. Summarizes mapping results, but not at the level of individual rows.  
- Verbose initialization. In addition to normal tracing, the Data Integration Service logs additional initialization details, names of index and data files used, and detailed transformation statistics.  
- Verbose data. In addition to verbose initialization tracing, the Data Integration Service logs each row that passes into the mapping. Also notes where the Data Integration Service truncates string data to fit the precision of a column and provides detailed transformation statistics. Default is None. |
| Sort order                    | Order in which the Data Integration Service sorts character data in the mapping. Default is Binary. |
| Optimizer level               | Controls the optimization methods that the Data Integration Service applies to a mapping as follows:  
- None. The Data Integration Service does not optimize the mapping.  
- Minimal. The Data Integration Service applies the early projection optimization method to the mapping.  
- Normal. The Data Integration Service applies the early projection, early selection, pushdown, and predicate optimization methods to the mapping.  
- Full. The Data Integration Service applies the early projection, early selection, pushdown, predicate, cost-based, and semi-join optimization methods to the mapping. Default is Normal. |
| High precision                | Runs the mapping with high precision. High precision data values have greater accuracy. Enable high precision if the mapping produces large numeric values, for example, values with precision of more than 15 digits, and you require accurate values. Enabling high precision prevents precision loss in large numeric values. Default is enabled. |
| Send log to client            | Allows you to view log files in the Developer tool. If you disable this option, you must view log files through the Administrator tool. Default is enabled. |
Data Viewer Configurations

Data viewer configurations control the settings that the Developer tool applies when you preview output in the Data Viewer view.

You can select a data viewer configuration when you preview output for the following objects:

- Custom data objects
- Logical data objects
- Logical data object read mappings
- Physical data objects
- Sources and transformations within mappings

Creating a Data Viewer Configuration

Create a data viewer configuration to control the settings the Developer tool applies when you preview output in the Data Viewer view.

1. Click Run > Open Run Dialog.
   The Run Configurations dialog box appears.

2. Click Data Viewer Configuration.

3. Click the New button.
The right panel of the Run Configurations dialog box displays the data viewer configuration properties.

4. Enter a name for the data viewer configuration.
5. Configure the data viewer configuration properties.
6. Click Apply.
7. Click Close.

The Developer tool creates the data viewer configuration.

Mapping Configurations

Mapping configurations control the mapping deployment properties that the Developer tool uses when you run a mapping through the Run Configurations dialog box or from the command line.

To apply a mapping configuration to a mapping that you run through the Developer tool, you must run the mapping through the Run Configurations dialog box. If you run the mapping through the Run menu or mapping editor, the Developer tool runs the mapping with the default mapping deployment properties.

To apply mapping deployment properties to a mapping that you run from the command line, select the mapping configuration when you add the mapping to an application. The mapping configuration that you select applies to all mappings in the application.

You can change the mapping deployment properties when you edit the application. An administrator can also change the mapping deployment properties through the Administrator tool. You must redeploy the application for the changes to take effect.

Creating a Mapping Configuration

Create a mapping configuration to control the mapping deployment properties that the Developer tool uses when you run mappings through the Run dialog box or from the command line.
1. Click **Run > Open Run Dialog**.
   The **Run Configurations** dialog box appears.

2. Click **Mapping Configuration**.

3. Click the **New** button ( ).
   The right panel of the **Run Configurations** dialog box displays the mapping configuration properties.

4. Enter a name for the mapping configuration.
5. Configure the mapping configuration properties.
6. Click **Apply**.
7. Click **Close**.

The Developer tool creates the mapping configuration.

### Updating the Default Configuration Properties

You can update the default data viewer and mapping configuration properties.

1. Click **Window > Preferences**.
   
   The **Preferences** dialog box appears.
2. Click **Informatica > Run Configurations**.
3. Select the **Data Viewer** or **Mapping** configuration.

   The following figure shows the default data viewer configuration properties:

   ![Default Data Viewer Configuration Properties](image)

4. Configure the default data viewer or mapping configuration properties.
5. Click **OK**.

   The Developer tool updates the default configuration properties.

### Troubleshooting Configurations

I created two configurations with the same name but with different cases. When I close and reopen the Developer tool, one configuration is missing.

Data viewer and mapping configuration names are not case sensitive. If you create multiple configurations with the same name but different cases, the Developer tool deletes one of the configurations when you exit. The Developer tool does not consider the configuration names unique.
I tried to create a configuration with a long name, but the Developer tool displays an error message that says it cannot not write the file.

The Developer tool stores data viewer and mapping configurations in files on the machine that runs the Developer tool. If you create a configuration with a long name, for example, more than 100 characters, the Developer tool might not be able to save the file to the hard drive.

To work around this issue, shorten the configuration name.

**Exporting Data**

You can export the data that appears in the Data Viewer view to a tab-delimited flat file, such as a TXT or CSV file. Export data when you want to create a local copy of the data.

1. In the Data Viewer view, right-click the results and select Export Data.
2. Enter a file name and extension.
3. Select the location where you want to save the file.
4. Click OK.

**Object Dependencies**

Before you change or delete an object, you can view object dependencies for all objects in the Model repository that the object has an impact on. You can view object dependencies for all objects that are dependent on the object and objects that this object depends on.

For example, you want to delete a data object that is deployed in multiple applications. However, you do not know if deleting this data object has an impact on the deployed applications. You can view object dependencies to determine if this data object has an impact on the deployed applications. After you delete the data object, you can redeploy the applications.

You can see object dependencies in the Object Dependencies view. If a dependent object is in a project that you do not have permission to read, the Developer tool does not display the object in the Object Dependencies view. The Developer tool displays a message that states the Developer tool cannot access the object.

**View Object Dependencies**

You can view object dependencies for an object in the Object Dependencies view. You can view dependencies for objects that you select from the Object Explorer view or for connections that you select from the Connection Explorer view.

You can perform the following tasks to view object dependencies:

**View circular dependencies.**

A circular dependency occurs when two objects depend on each other. For example object A depends on object B, which depends on object A. When the Developer tool encounters the second instance of the object in the object dependency tree, it does not display further instances of the object. Instead, the Developer tool adds a circle icon against the object to indicate that a circular dependency occurred.
View downstream or upstream dependencies.

View downstream dependencies to view the objects that depend on the selected object. The Developer tool displays downstream dependencies by default.

View upstream dependencies to view objects that the selected object is dependent on.

Filter object dependencies.

Filter object dependencies to narrow the list of dependent objects. You can choose to filter by types of objects or by projects. For example, you might want to see workflows that a particular object has an impact on. You can filter by object type and select workflows.

View object dependencies history.

View history for the last 10 object dependencies.

Viewing Object Dependencies

When you view dependencies, you can apply filters, clear history, or choose more information about particular dependencies.

1. In the Object Explorer view or Connection Explorer view, right-click an object or connection and click Find Dependencies.
   The Object Dependencies view displays a list of object dependencies for the selected object or connection in an object dependency tree.

2. Select an object or connection in the object dependency tree to view the object or connection properties.

3. Optionally, to filter dependencies, click the Filters icon and choose to filter by types of objects or projects.

4. You can view the following types of dependencies:
   - Circular. The Developer tool adds a circle icon against the object to indicate that a circular dependency occurred.
   - Upstream. Click the Upstream Dependencies icon to view the upstream dependencies.
   - Downstream. Click the Downstream Dependencies icon to view the downstream dependencies.

5. Optionally, to view dependency history, click the History icon to view the history of object dependencies. Optionally, click Clear History to clear the history.

Filtering Object Dependencies

You can filter the object dependency tree by Types and Projects. In the filtered results, some objects may not be direct dependents of the parent object. These objects appear with a plus (+) icon that hides the full path to the parent object in the object dependency tree. You can choose to expand the path to insert all object levels till the parent object.

1. In the Object Dependencies view, click the Filters icon.
   The Filters dialog box opens.

2. Accept the default selections on the Types tab to filter by object type.

3. Optionally, click the Projects tab and choose to filter by all projects in the Model repository or by each project.
   - To filter by all projects in the Model repository, select the option to include all projects.
   - To filter by each project, select the option to include opened projects.
4. Choose to select an object type or project, select all object types or projects, or clear the selected object types and projects.
   - To select an object type or project, select each object type or project.
   - To select all object types or projects, click Select All.
   - To remove all selected object types and projects, click Select None.

5. Click OK.
   The filtered object dependency tree appears in the Object Dependencies view.

6. Optionally, if a plus (+) icon appears against an object name, right-click the object and click Expand Path to insert all object levels till the parent object.

Logs

The Data Integration Service generates log events when you run a mapping, run a profile, or preview data. Log events include information about the tasks performed by the Data Integration Service, errors, and load summary and transformation statistics.

You can view the logs generated from the Developer tool and save them to a local directory.

You can view log events from the Show Log button in the Data Viewer view.

The following figure shows the Show Log button in the Data Viewer view:

When you run a mapping from Run > Run Mapping, you can view the log events from the Progress view. To open the log events in the Developer tool, click the link for the mapping run and select Go to Log.
The following figure shows the link for a mapping run in the **Progress** view:

When you run a profile, you can view the log events from the Monitoring tool.

To save the log to a file, click **File > Save a Copy As** and choose a directory. By default the log files are stored in the following directory: `c:\[TEMP]\AppData\Local\Temp`.

**Log File Format**

The information in the log file depends on the sequence of events during the run. The amount of information that is sent to the logs depends on the tracing level.

The Data Integration Service updates log files with the following information when you run a mapping, run a profile, or preview data:

- **Logical DTM messages**
  
  Contain information about preparing to compile, to optimize, and to translate the mapping. The log events and the amount of information depends on the configuration properties set.

- **Data Transformation Manager (DTM) messages**
  
  Contain information about establishing a connection to the source, reading the data, transforming the data, and loading the data to the target.

- **Load summary and transformation statistics messages**
  
  Contain information about the number of rows read from the source, number of rows output to the target, number of rows rejected, and the time to execute.

**Validation Preferences**

You can limit the number of error messages that appear in the **Validation Log** view. You can also group error messages by object or object type in the **Validation Log** view.

**Grouping Error Messages**

Group error messages in the **Validation Log** view to organize messages by object or object type. Otherwise, messages appear alphabetically.

To group error messages in the **Validation Log** view, select **Menu > Group By** and then select **Object** or **Object Type**.
To remove error message groups, select **Menu > Group By > None**. Error messages appear ungrouped, listed alphabetically in the **Validation Log** view.

**Limiting Error Messages**

You can limit the number of error messages that appear in the **Validation Log** view. The limit determines how many messages appear in a group or the total number of messages that appear in the **Validation Log** view. Error messages are listed alphabetically and get deleted from bottom to top when a limit is applied.

1. Click **Window > Preferences**.
   The **Preferences** dialog box appears.
2. Select **Informatica > Validation**.
3. Optionally, set the error limit and configure the number of items that appear.
   Default is 100.
4. To restore the default values, click **Restore Defaults**.
5. Click **Apply**.
6. Click **OK**.

**Monitoring Jobs from the Developer Tool**

You can access the Monitoring tool from the Developer tool to monitor the status of applications and jobs, such as a profile jobs. As an administrator, you can also monitor applications and jobs in the Administrator tool.

Monitor applications and jobs to view properties, run-time statistics, and run-time reports about the integration objects. For example, you can see the general properties and the status of a profiling job. You can also see who initiated the job and how long it took the job to complete.

To monitor applications and jobs from the Developer tool, click the Menu button in the **Progress** view and select **Monitor Jobs**. Select the Data Integration Service that runs the applications and jobs and click **OK**. The Monitoring tool opens.
CHAPTER 11

Deployment

This chapter includes the following topics:

- Deployment Overview, 146
- Deployment Methods, 147
- Mapping Deployment Properties, 147
- Creating an Application, 149
- Deploying an Object to a Data Integration Service, 149
- Deploying an Object to a File, 150
- Updating an Application, 150
- Importing Application Archives, 150
- Application Redeployment, 151

Deployment Overview

Deploy objects to make them accessible outside of the Developer tool. You can deploy physical data objects, logical data objects, mappings, mapplets, workflows, and applications.

Deploy objects so that you can run mappings or workflows at the command line. When you deploy an object, you isolate the object from changes in data structures. If you make changes to an object in the Developer tool after you deploy it, you must redeploy the application that contains the object for the changes to take effect.

You can deploy objects to a Data Integration Service or a network file system. When you deploy an application to a Data Integration Service, end users can connect to the application. Depending on the types of objects in the application, end users can then run queries against the objects, or run mappings or workflows. The end users must have the appropriate permissions in the Administrator tool to perform these tasks.

When you deploy an object to a network file system, the Developer tool creates an application archive file. Deploy an object to a network file system if you want to check the application into a version control system. You can also deploy an object to a file if your organization requires that administrators deploy objects to a Data Integration Service. An administrator can deploy application archive files to a Data Integration Service through the Administrator tool. You can also import objects from an application archive into projects or folders in the Model repository.
Deployment Methods

Deploy objects or deploy an application that contains one or more objects. The object deployment method differs based on the type of object that you deploy.

Deploy an object

Deploy an object to make the object available to end users. If you redeploy an object to a Data Integration Service, you cannot update the application. The Developer tool creates an application with a different name.

When you deploy the following objects, the Developer tool prompts you to create an application and the Developer tool adds the object to the application:

- Mappings
- Workflows

Deploy an application that contains objects

Create an application to deploy multiple objects at the same time. When you create an application, you select the objects to include in the application. If you redeploy an application to a Data Integration Service you can update or replace the application.

To deploy an application that contains a large number of objects, run infacmd oie DeployApplication. When you deploy an application with a large number of objects from the Developer tool, the deployment might fail with an out of memory error. For more information about out of memory errors, see the Informatica Knowledgebase article 159120.

Mapping Deployment Properties

When you update an application that contains a mapping, you can set the deployment properties that the Data Integration Service uses when end users run the mapping.

Set mapping deployment properties on the Advanced view of the application.
The following table describes the mapping deployment properties that you can set:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default date time format</td>
<td>Date/time format that the Data Integration Service uses when the mapping converts strings to dates. Default is MM/DD/YYYY HH24:MI:SS.</td>
</tr>
<tr>
<td>Override tracing level</td>
<td>Overrides the tracing level for each transformation in the mapping. The tracing level determines the amount of information that the Data Integration Service sends to the mapping log files.</td>
</tr>
<tr>
<td>Sort order</td>
<td>Order in which the Data Integration Service sorts character data in the mapping. Default is Binary.</td>
</tr>
<tr>
<td>Optimizer level</td>
<td>Controls the optimization methods that the Data Integration Service applies to a mapping as follows:</td>
</tr>
<tr>
<td>High precision</td>
<td>Runs the mapping with high precision. High precision data values have greater accuracy. Enable high precision if the mapping produces large numeric values, for example, values with precision of more than 15 digits, and you require accurate values. Enabling high precision prevents precision loss in large numeric values. Default is enabled.</td>
</tr>
</tbody>
</table>
Creating an Application

Create an application when you want to deploy multiple objects at the same time or if you want to be able to update or replace the application when it resides on the Data Integration Service. When you create an application, you select the objects to include in the application.

1. Select a project or folder in the Object Explorer view.
2. Click File > New > Application.
   The New Application dialog box appears.
3. Enter a name for the application.
4. Click Browse to select the application location.
   You must create the application in a project or a folder.
5. Click Next.
   The Developer tool prompts you for the objects to include in the application.
6. Click Add.
   The Add Objects dialog box appears.
7. Select one or more objects and click OK.
   The Developer tool lists the objects you select in the New Application dialog box.
8. If the application contains mappings, choose whether to override the default mapping configuration when you deploy the application. If you select this option, choose a mapping configuration.
   The Developer tool sets the mapping deployment properties for the application to the same values as the settings in the mapping configuration.
9. Click Finish.
   The Developer tool adds the application to the project or folder.

After you create an application, you must deploy the application so end users can run mappings or workflows.

Deploying an Object to a Data Integration Service

Deploy an object to a Data Integration Service so end users can run mappings or workflows from the command line.

1. Right-click an object in the Object Explorer view and select Deploy.
   The Deploy dialog box appears.
2. Select Deploy to Service.
3. Click Next.
4. Enter an application name.
5. Click Finish.
   The Developer tool deploys the application to the Data Integration Service.
Deploying an Object to a File

Deploy an object to an application archive file if you want to check the application into version control or if you want to deploy objects to the Data Integration Service.

1. Right-click an object in the Object Explorer view and select **Deploy**. The **Deploy** dialog box appears.
2. Select **Deploy to File System**.
3. Click **Browse** to select the directory. The **Choose a Directory** dialog box appears.
4. Select the directory and click **OK**. Then, click **Next**.
5. Enter an application name.
6. Click **Finish**. The Developer tool deploys the application to an application archive file.

Before end users can access the application, you must deploy the application to a Data Integration Service. Or, an administrator must deploy the application to a Data Integration Service through the Administrator tool.

Updating an Application

Update an application when you want to add objects to an application, remove objects from an application, or update mapping deployment properties.

1. Open the application you want to update.
2. To add or remove objects, click the **Overview** view.
3. To add objects to the application, click **Add**. The Developer tool prompts you to choose the objects to add to the application.
4. To remove an object from the application, select the object, and click **Remove**.
5. To update mapping deployment properties, click the **Advanced** view and change the properties.
6. Save the application.
Redeploy the application if you want end users to be able to access the updated application.

Importing Application Archives

You can import objects from an application archive file. You import the application and dependent objects into the repository.

1. Click **File > Import**. The **Import** wizard appears.
2. Select **Informatica > Application Archive**.
3. Click **Next**.
4. Click **Browse** to select the application archive file. The Developer tool lists the application archive file contents.

5. Select the repository into which you want to import the application.

6. Click **Finish**. The Developer tool imports the application into the repository. If the Developer tool finds duplicate objects, it renames the imported objects.

**Application Redeployment**

When you change an application or change an object in the application and you want end users to access the latest version of the application, you must deploy the application again.

When you change an application or its contents and you deploy the application to the same Data Integration Service, the Developer tool gives you the following choices:

- **Update.** The Data Integration Service replaces the objects and preserves the object properties in the Administrator tool.
- **Replace.** The Data Integration Service replaces the objects and resets the object properties in the Administrator tool to the default values.

To update or replace an application that is running, you must first stop the application. When you stop an application, the Data Integration Service aborts all running objects in the application. If you do not want to abort running objects, you can rename the application or deploy the application to a different service.

When you change an application and deploy it to a network file system, the Developer tool allows you to replace the application archive file or cancel the deployment. If you replace the application archive file, the Developer tool replaces the objects in the application and resets the object properties.

**Redeploying an Application**

Redeploy an application to a Data Integration Service when you want to update or replace the application.

1. Right-click an application in the **Object Explorer** view and click **Deploy**. The **Deploy** dialog box appears.

2. Select **Deploy to Service**.

3. Click **Browse** to select the domain. The **Choose Domain** dialog box appears.

4. Select a domain and click **OK**. The Developer tool lists the Data Integration Service associated with the domain in the **Available Services** section of the **Deploy Application** dialog box.

5. Select the Data Integration Service to which you want to deploy the application.

6. If the Data Integration Service already contains the deployed application, select to update or replace the application in the **Action** column.

7. If the deployed application is running, select **Force the Existing Application to Stop**.

8. Click **Finish**.
Object Import and Export Overview

You can export multiple objects from a project to one XML file. When you import objects, you can choose individual objects in the XML file or all the objects in the XML file.

You can export objects to an XML file and then import objects from the XML file. When you export objects, the Developer tool creates an XML file that contains the metadata of the exported objects. Use the XML file to import the objects into a project or folder. You can also import and export objects through infacmd command.

Export and import objects to accomplish the following tasks:

Deploy metadata into production

After you test a mapping in a development repository, you can export it to an XML file and then import it from the XML file into a production repository.

Archive metadata

You can export objects to an XML file that you no longer need before you remove them from the repository.

Share metadata

You can share metadata with a third party. For example, you can send a mapping to someone else for testing or analysis.

You can use infacmd to generate a readable XML file from an export file. You can also edit the object names in the readable XML and update the export XML before you import the objects into a repository.
You can import and export projects and objects in a project. You can also import and export application archive files in a repository.

When you export an object, the Developer tool also exports the dependent objects. A dependent object is an object that is used by another object. For example, a physical data object used as a mapping input is a dependent object of that mapping. When you import an object, the Developer tool imports all the dependent objects.

When you export or import objects in a project or folder, the Model Repository Service preserves the object hierarchy.

The following table lists objects and dependent objects that you can export:

<table>
<thead>
<tr>
<th>Object</th>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>- Mappings, or workflows and their dependent objects</td>
</tr>
<tr>
<td>Project</td>
<td>- Projects contain other objects, but they do not have dependent objects</td>
</tr>
<tr>
<td>Folder</td>
<td>- Folders contain other objects, but they do not have dependent objects</td>
</tr>
<tr>
<td>Physical data object (except for customized data object)</td>
<td>- Physical data objects do not have dependent objects</td>
</tr>
<tr>
<td>Customized data object</td>
<td>- Physical data objects</td>
</tr>
<tr>
<td>Logical data object model</td>
<td>- Logical data objects</td>
</tr>
<tr>
<td></td>
<td>- Physical data objects</td>
</tr>
<tr>
<td></td>
<td>- Reusable transformations and their dependent objects</td>
</tr>
<tr>
<td></td>
<td>- Mapplets and their dependent objects</td>
</tr>
<tr>
<td>Transformation</td>
<td>- Physical data objects</td>
</tr>
<tr>
<td>Mapplet</td>
<td>- Logical data objects</td>
</tr>
<tr>
<td></td>
<td>- Physical data objects</td>
</tr>
<tr>
<td></td>
<td>- Reusable transformations and their dependent objects</td>
</tr>
<tr>
<td></td>
<td>- Mapplets and their dependent objects</td>
</tr>
<tr>
<td>Mapping</td>
<td>- Logical data objects</td>
</tr>
<tr>
<td></td>
<td>- Physical data objects</td>
</tr>
<tr>
<td></td>
<td>- Reusable transformations and their dependent objects</td>
</tr>
<tr>
<td></td>
<td>- Mapplets and their dependent objects</td>
</tr>
<tr>
<td>Profile</td>
<td>- Logical data objects</td>
</tr>
<tr>
<td></td>
<td>- Physical data objects</td>
</tr>
<tr>
<td>Workflow</td>
<td>- Mappings and their dependent objects</td>
</tr>
</tbody>
</table>
Object Export

When you export an object, the Developer tool creates an XML file that contains the metadata of the objects.

You can choose the objects to export. You must also choose to export all dependent objects. The Developer tool exports the objects and the dependent objects. The Developer tool exports the last saved version of the object. The Developer tool includes Cyclic Redundancy Checking Value (CRCVALUE) codes in the elements in the XML file. If you modify attributes in an element that contains a CRCVALUE code, you cannot import the object.

You can also export objects with the infacmd oie ExportObjects command.

Exporting Objects

To use Model repository objects in another repository, you can export the objects as an XML metadata file.

1. Click File > Export.
   The Export wizard opens.
   Click Next.
3. Click Browse. Select the repository project that contains the objects to export.
   Click Next.
4. Select one or more objects to export. If you highlighted a repository object before you started the export process, the wizard selects the object for you.
5. Enter a file name and a location for the XML metadata file. The Developer tool exports all the objects that you select to a single file.
   Click Next.
6. The wizard displays any dependent object that the metadata objects use.
   Click Next to accept the dependent objects.
7. Click Finish to export the selected objects.
   The Developer tool exports the object metadata to an XML file and exports any dependent reference data files to a ZIP file.

Object Import

You can import a project or objects within a project from an export file. You can import the objects and any dependent objects into a project or folder.

When you import objects, you can import a project or individual objects. Import a project when you want to reuse all objects in the project. Import individual objects when you want to reuse objects across projects.

When you import an object, the Developer tool lists all the dependent objects. You must add each dependent object to the target before you can import the object.

When you import objects, an object in the export file might have the same name as an object in the target project or folder. You can choose how you want to resolve naming conflicts.

You can also import objects with the infacmd oie ImportObjects command.
Importing Projects

You can import a project from an XML file into the target repository. You can also import the contents of the project into a project in the target repository.

1. Click File > Import.
2. Select Informatica > Import Object Metadata File (Basic).
3. Click Next.
4. Click Browse and select the export file that you want to import.
5. Click Next.
6. Select the project or select "<project name> Project Content" in the Source pane.
   • If you select the project in the Source pane, select the Model Repository Service in the Target pane where you want to import the project.
   • If you select the project content in the Source pane, select the project to which you want to import the project contents in the Target pane.
7. Click Add to Target to add the project to the target.
   Tip: You can also drag the project from the Source pane into the repository in the Target pane. Or, you can drag the project content in the Source pane into a project in the Target pane.
8. Click Resolution to specify how to handle duplicate objects.
   You can rename the imported object, replace the existing object with the imported object, or reuse the existing object. The Developer tool renames all the duplicate objects by default.
9. Click Next.
The Developer tool summarizes the objects to be imported. Click **Link Source and Target Objects** to link the objects in the Source and Target display panes when you select one of the objects. For example, if you select this option and then select an object in the Source pane, the Developer tool selects the same object in the Target pane.

10. Map the connections from the import file to the target domain connections in the Additional Import Settings pane. You can also select whether to overwrite existing tags on the objects.

11. Click **Finish**.

   If you chose to rename the duplicate project, the Model Repository Service appends a number to the object name. You can rename the project after you import it.

### Importing Objects

You can import objects from an XML file or application archive file. You import the objects and any dependent objects into a project.

1. Click **File > Import**.
2. Select **Informatica > Import Object Metadata File (Advanced)**.
3. Click **Next**.
4. Click **Browse** to select the export file that you want to import.
5. Click **Next**.
6. Select the object in the Source pane that you want to import.
7. Select the project in the Target pane to which you want to import the object.
8. Click **Add to Target** to add the object to the target.
If you click **Auto Match to Target**, the Developer tool tries to match the descendants of the current source selection individually by name, type, and parent hierarchy in the target selection and adds the objects that match.

If you want to import all the objects under a folder or a project, select the target folder or project and click **Add Content to Target**.

**Tip:** You can also drag the object from the Source pane into the required project in the Target pane. Press the control key while you drag to maintain the object hierarchy in source and target.

9. Click to specify how to handle duplicate objects.

You can rename the imported object, replace the existing object with the imported object, or reuse the existing object. The Developer tool renames all the duplicate objects by default.

10. Click **Next**.

The Developer tool lists any dependent objects in the import file.

11. Add dependent objects to a target folder or project.

12. Click **Next**.

The Developer tool summarizes the objects to be imported. Click **Link Source and Target Objects** to link the objects in the Source and Target display panes when you select one of the objects. For example, if you select this option and then select an object in the Source pane, the Developer tool selects the same object in the Target pane.

13. Map the connections from the import file to the target domain connections in the Additional Import Settings pane. You can also select whether to overwrite existing tags on the objects.

14. Click **Finish**.

If you choose to rename the duplicate project, the **Import** wizard names the imported project as "<Original Name>_<number of the copy>." You can rename the project after you import it.
Datatype Reference Overview

When you create a mapping, you create a set of instructions for the Data Integration Service to read data from a source, transform it, and write it to a target. The Data Integration Service transforms data based on dataflow in the mapping, starting at the first transformation in the mapping, and the datatype assigned to each port in a mapping.

The Developer tool displays two types of datatypes:

- Native datatypes. Specific to the relational table or flat file used as a physical data object. Native datatypes appear in the physical data object column properties.

- Transformation datatypes. Set of datatypes that appear in the transformations. They are internal datatypes based on ANSI SQL-92 generic datatypes, which the Data Integration Service uses to move data across platforms. The transformation datatypes appear in all transformations in a mapping.

When the Data Integration Service reads source data, it converts the native datatypes to the comparable transformation datatypes before transforming the data. When the Data Integration Service writes to a target, it converts the transformation datatypes to the comparable native datatypes.

When you specify a multibyte character set, the datatypes allocate additional space in the database to store characters of up to three bytes.
Transformation Datatypes

The following table describes the transformation datatypes:

<table>
<thead>
<tr>
<th>Datatype</th>
<th>Size in Bytes</th>
<th>Description</th>
</tr>
</thead>
</table>
| Bigint   | 8 bytes       | -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807  
  Precision of 19, scale of 0  
  Integer value. |
| Binary   | Precision     | 1 to 104,857,600 bytes  
  You cannot use binary data for flat file sources. |
| Date/Time| 16 bytes      | Jan 1, 0001 A.D. to Dec 31, 9999 A.D.  
  Precision of 29, scale of 9  
  (precision to the nanosecond)  
  Combined date/time value. |
| Decimal  | 8 bytes (if high precision is off or precision is greater than 28)  
  16 bytes (if precision <= 18 and high precision is on)  
  20 bytes (if precision >18 and <= 28)  
  Precision 1 to 28 digits, scale 0 to 28  
  Decimal value with declared precision and scale.  
  Scale must be less than or equal to precision. |
| Double   | 8 bytes       | Precision of 15 digits  
  Double-precision floating-point numeric value. |
| Integer  | 4 bytes       | -2,147,483,648 to 2,147,483,647  
  Precision of 10, scale of 0  
  Integer value. |
| String   | Unicode mode: (precision + 1) * 2  
  ASCII mode: precision + 1  
  1 to 104,857,600 characters  
  Fixed-length or varying-length string. |
| Text     | Unicode mode: (precision + 1) * 2  
  ASCII mode: precision + 1  
  1 to 104,857,600 characters  
  Fixed-length or varying-length string. |

Integer Datatypes

You can pass integer data from sources to targets and perform transformations on integer data. The transformation language supports Bigint and Integer datatypes.

The transformation integer datatypes represent exact values.

Integer Values in Calculations

When you use integer values in calculations, the Integration Service sometimes converts integer values to floating-point numbers before it performs the calculation. For example, to evaluate MOD( 12.00, 5 ), the
Integration Service converts the integer value "5" to a floating-point number before it performs the division operation. The Integration Service converts integer values to double or decimal values depending on whether you enable high precision.

The Integration Service converts integer values in the following arithmetic operations:

<table>
<thead>
<tr>
<th>Arithmetic Operation</th>
<th>High Precision Disabled</th>
<th>High Precision Enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functions and calculations that cannot introduce decimal points. For example, integer addition, subtraction, and multiplication, and functions such as CUME, MOVINGSUM, and SUM.</td>
<td>No conversion (1)</td>
<td>Decimal</td>
</tr>
<tr>
<td>Non-scientific functions and calculations that can introduce decimal points. For example, integer division, and functions such as AVG, MEDIAN, and PERCENTILE.</td>
<td>Double</td>
<td>Decimal</td>
</tr>
<tr>
<td>All scientific functions and the EXP, LN, LOG, POWER, and SQRT functions.</td>
<td>Double</td>
<td>Double</td>
</tr>
</tbody>
</table>

1. If the calculation produces a result that is out of range, the Integration Service writes a row error.

The transformation Double datatype supports precision of up to 15 digits, while the Bigint datatype supports precision of up to 19 digits. Therefore, precision loss can occur in calculations that produce Bigint values with precision of more than 15 digits.

For example, an expression transformation contains the following calculation:

```powershell
POWER(BIGINTVAL, EXPVAL)
```

Before it performs the calculation, the Integration Service converts the inputs to the POWER function to double values. If the BIGINTVAL port contains the Bigint value 9223372036854775807, the Integration Service converts this value to 9.22337203685478e+18, losing the last 4 digits of precision. If the EXPVAL port contains the value 1.0 and the result port is a Bigint, this calculation produces a row error since the result, 9223372036854780000, exceeds the maximum bigint value.

When you use an Integer datatype in a calculation that can produce decimal values and you enable high precision, the Integration Service converts the integer values to decimal values. The transformation Decimal datatype supports precision of up to 28 digits. Therefore, precision loss does not occur in a calculation unless the result produces a value with precision greater than 28 digits. In this case, the Integration Service stores the result as a double.

**Integer Constants in Expressions**

The Integration Service interprets constants in an expression as floating-point values, even if the calculation produces an integer result. For example, in the expression INTVALUE + 1000, the Integration Service converts the integer value "1000" to a double value if high precision is not enabled. It converts the value 1000 to a decimal value if high precision is enabled. To process the value 1000 as an integer value, create a variable port with an Integer datatype to hold the constant and modify the expression to add the two ports.
**NaN Values**

NaN (Not a Number) is a value that is usually returned as the result of an operation on invalid input operands, especially in floating-point calculations. For example, when an operation attempts to divide zero by zero, it returns a NaN result.

Operating systems and programming languages may represent NaN differently. For example the following list shows valid string representations of NaN:

```
nan
NaN
NaN
NAN
NaNQ
NaNQ
qNaN
sNaN
1.#QNAN
1.#QNAN
```

The Integration Service converts QNAN values to 1.#QNAN on Win64EMT platforms. 1.#QNAN is a valid representation of NaN.

**Write Integer Values to Flat Files**

*Topic is used in both PC Designer Guide and Infa Developer User Guide.*

When writing integer values to a fixed-width flat file, the file writer does not verify that the data is within range. For example, the file writer writes the result 3,000,000,000 to a target Integer column if the field width of the target column is at least 13. The file writer does not reject the row because the result is outside the valid range for Integer values.

**Binary Datatype**

*Topic is used in both PC Designer Guide and Infa Developer User Guide.*

If a mapping includes binary data, set the precision for the transformation binary datatype so that the Integration Service can allocate enough memory to move the data from source to target.

You cannot use binary datatypes for flat file sources.

**Date/Time Datatype**

*Topic is used in both PC Designer Guide and Infa Developer User Guide.*

The Date/Time datatype handles years from 1 A.D. to 9999 A.D. in the Gregorian calendar system. Years beyond 9999 A.D. cause an error.

The Date/Time datatype supports dates with precision to the nanosecond. The datatype has a precision of 29 and a scale of 9. Some native datatypes have a smaller precision. When you import a source that contains datetime values, the import process imports the correct precision from the source column. For example, the Microsoft SQL Server Datetime datatype has a precision of 23 and a scale of 3. When you import a Microsoft SQL Server source that contains Datetime values, the Datetime columns in the mapping source have a precision of 23 and a scale of 3.

The Integration Service reads datetime values from the source to the precision specified in the mapping source. When the Integration Service transforms the datetime values, it supports precision up to 29 digits. For example, if you import a datetime value with precision to the millisecond, you can use the ADD_TO_DATE function in an Expression transformation to add nanoseconds to the date.
If you write a Date/Time value to a target column that supports a smaller precision, the Integration Service truncates the value to the precision of the target column. If you write a Date/Time value to a target column that supports a larger precision, the Integration Service inserts zeroes in the unsupported portion of the datetime value.

### Decimal and Double Datatypes

**Topic is used in both PC Designer Guide and Infa Developer User Guide.**

You can pass decimal and double data from sources to targets and perform transformations on decimal and double data. The transformation language supports the following datatypes:

- **Decimal.** Precision 1 to 28 digits, scale 0 to 28. You cannot use decimal values with scale greater than precision or a negative precision. Transformations display any range you assign to a Decimal datatype, but the Integration Service supports precision only up to 28.

- **Double.** Precision of 15.

### Decimal and Double Values in Calculations

**Topic is used in both PC Designer Guide and Infa Developer User Guide.**

The transformation Decimal datatype supports precision of up to 28 digits and the Double datatype supports precision of up to 15 digits. Precision loss can occur with either datatype in a calculation when the result produces a value with a precision greater than the maximum.

If you disable high precision, the Integration Service converts decimal values to doubles. Precision loss occurs if the decimal value has a precision greater than 15 digits. For example, you have a mapping with Decimal (20,0) that passes the number 40012030304957666903. If you disable high precision, the Integration Service converts the decimal value to double and passes $4.00120303049577 \times 10^{19}$.

To ensure precision of up to 28 digits, use the Decimal datatype and enable high precision. When you enable high precision, the Integration Service processes decimal values as Decimal. Precision loss does not occur in a calculation unless the result produces a value with precision greater than 28 digits. In this case, the Integration Service stores the result as a double. Do not use the Double datatype for data that you use in an equality condition, such as a lookup or join condition.

The following table lists how the Integration Service handles decimal values based on high precision configuration:

<table>
<thead>
<tr>
<th>Port Datatype</th>
<th>Precision</th>
<th>High Precision Off</th>
<th>High Precision On</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decimal</td>
<td>0-28</td>
<td>Double</td>
<td>Decimal</td>
</tr>
<tr>
<td>Decimal</td>
<td>Over 28</td>
<td>Double</td>
<td>Double</td>
</tr>
</tbody>
</table>

When you enable high precision, the Integration Service converts numeric constants in any expression function to Decimal. If you do not enable high precision, the Integration Service converts numeric constants to Double.

To ensure the maximum precision for numeric values greater than 28 digits, truncate or round any large numbers before performing any calculations or transformations with the transformation functions.

### Rounding Methods for Double Values

Due to differences in system run-time libraries and the computer system where the database processes double datatype calculations, the results may not be as expected. The double datatype conforms to the IEEE...
794 standard. Changes to database client library, different versions of a database or changes to a system run-time library affect the binary representation of mathematically equivalent values. Also, many system run-time libraries implement the round-to-even or the symmetric arithmetic method. The round-to-even method states that if a number falls midway between the next higher or lower number it round to the nearest value with an even least significant bit. For example, with the round-to-even method, 0.125 is rounded to 0.12. The symmetric arithmetic method rounds the number to next higher digit when the last digit is 5 or greater. For example, with the symmetric arithmetic method 0.125 is rounded to 0.13 and 0.124 is rounded to 0.12.

To provide calculation results that are less susceptible to platform differences, the Integration Service stores the 15 significant digits of double datatype values. For example, if a calculation on Windows returns the number 1234567890.1234567890, and the same calculation on UNIX returns 1234567890.1234569999, the Integration Service converts this number to 1234567890.1234600000.

String Datatypes

The transformation datatypes include the following string datatypes:

- String
- Text

Although the String and Text datatypes support the same precision up to 104,857,600 characters, the Integration Service uses String to move string data from source to target and Text to move text data from source to target. Because some databases store text data differently than string data, the Integration Service needs to distinguish between the two types of character data. In general, the smaller string datatypes, such as Char and Varchar, display as String in transformations, while the larger text datatypes, such as Text, Long, and Long Varchar, display as Text.

Use String and Text interchangeably within transformations. However, in Lookup transformations, the target datatypes must match. The database drivers need to match the string datatypes with the transformation datatypes, so that the data passes accurately. For example, Varchar in a lookup table must match String in the Lookup transformation.

Flat File and Transformation Datatypes

Flat file datatypes map to transformation datatypes that the Data Integration Service uses to move data across platforms.

The following table compares flat file datatypes to transformation datatypes:

<table>
<thead>
<tr>
<th>Flat File</th>
<th>Transformation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bigint</td>
<td>Bigint</td>
<td>Precision of 19 digits, scale of 0</td>
</tr>
<tr>
<td>Datetime</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>Double</td>
<td>Double</td>
<td>Precision of 15 digits</td>
</tr>
<tr>
<td>Int</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
</tbody>
</table>
### Flat File Datatypes

<table>
<thead>
<tr>
<th>Flat File</th>
<th>Transformation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nstring</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Number</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>String</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
</tbody>
</table>

When the Data Integration Service reads non-numeric data in a numeric column from a flat file, it drops the row and writes a message in the log. Also, when the Data Integration Service reads non-datetime data in a datetime column from a flat file, it drops the row and writes a message in the log.

### DB2 for LUW and Transformation Datatypes

DB2 for LUW datatypes map to transformation datatypes that the Data Integration Service uses to move data across platforms.

The following table compares DB2 for LUW datatypes and transformation datatypes:

<table>
<thead>
<tr>
<th>Datatype</th>
<th>Range</th>
<th>Transformation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blob</td>
<td>1 to 2,147,483,647 bytes</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Char</td>
<td>1 to 254 characters</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Char for bit data</td>
<td>1 to 254 bytes</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Clob</td>
<td>1 to 2,447,483,647 bytes</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Date</td>
<td>0001 to 9999 A.D. Precision 19; scale 0 (precision to the day)</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>Decimal</td>
<td>Precision 1 to 31, scale 0 to 31</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>Float</td>
<td>Precision 1 to 15</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>Smallint</td>
<td>-32,768 to 32,767</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>Datatype</td>
<td>Range</td>
<td>Transformation</td>
<td>Range</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------------------</td>
<td>----------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Time</td>
<td>24-hour time period</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td></td>
<td>Precision 19, scale 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(precision to the second)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timestamp</td>
<td>26 bytes</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td></td>
<td>Precision 26, scale 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(precision to the microsecond)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Varchar</td>
<td>Up to 4,000 characters</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Varchar for bit data</td>
<td>Up to 4,000 bytes</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
</tbody>
</table>

**Unsupported DB2 for LUW Datatypes**

The Developer tool does not support certain DB2 for LUW datatypes.

The Developer tool does not support the following DB2 for LUW datatypes:

- Dbclob
- Graphic
- Long Varchar
- Long Vargraphic
- Numeric
- Vargraphic

**JDBC and Transformation Datatypes**

When the Data Integration Service reads data from a JDBC source, it converts the native datatypes into the corresponding JDBC datatypes and then to the transformation datatypes. It uses the transformation datatypes to move data across platforms.

The following table compares the JDBC datatypes to the transformation datatypes:

<table>
<thead>
<tr>
<th>JDBC Datatype</th>
<th>Transformation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Precision 19, scale 0</td>
</tr>
<tr>
<td>Binary*</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Bit</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Precision 10, scale 0</td>
</tr>
<tr>
<td>Blob*</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>JDBC Datatype</td>
<td>Transformation</td>
<td>Range</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>-------</td>
</tr>
<tr>
<td>Boolean</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647 (\text{Precision 10, scale 0})</td>
</tr>
<tr>
<td>Char*</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Clob*</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Date</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (\text{precision to the nanosecond})</td>
</tr>
<tr>
<td>Decimal</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>Double</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Float</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Integer</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647 (\text{Precision 10, scale 0})</td>
</tr>
<tr>
<td>Long VarBinary*</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Long Varchar*</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Numeric</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>Real</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Smallint</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647 (\text{Precision 10, scale 0})</td>
</tr>
<tr>
<td>Time</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (\text{precision to the nanosecond})</td>
</tr>
<tr>
<td>Timestamp</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (\text{precision to the nanosecond})</td>
</tr>
<tr>
<td>Tinyint</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647 (\text{Precision 10, scale 0})</td>
</tr>
<tr>
<td>Varchar*</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Varbinary*</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
</tbody>
</table>

*If the size of data in a port is greater than 100 MB, the Developer tool sets the port precision to 4000 by default. To process data with a larger size, increase the port precision.*
Microsoft SQL Server and Transformation Data Types

Microsoft SQL Server data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares Microsoft SQL Server data types and transformation data types:

<table>
<thead>
<tr>
<th>Microsoft SQL Server</th>
<th>Range</th>
<th>Transformation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>1 to 8,000 bytes</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Bit</td>
<td>1 bit</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Char</td>
<td>1 to 8,000 characters</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Datetime</td>
<td>Jan 1, 1753 A.D. to Dec 31, 9999 A.D. (Precision 23, scale 3 (precision to 3.33 milliseconds))</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>Decimal</td>
<td>Precision 1 to 38, scale 0 to 38</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>Float</td>
<td>-1.79E+308 to 1.79E+308</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Image</td>
<td>1 to 2,147,483,647 bytes</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Int</td>
<td>-2,147,483,648 to 2,147,483,647</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647 (Precision 10, scale 0)</td>
</tr>
<tr>
<td>Money</td>
<td>-922,337,203,685,477.5807 to 922,337,203,685,477.5807</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>Numeric</td>
<td>Precision 1 to 38, scale 0 to 38</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>Real</td>
<td>-3.40E+38 to 3.40E+38</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Smalldatetime</td>
<td>Jan 1, 1900, to June 6, 2079 (Precision 19, scale 0 (precision to the minute))</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>Smallint</td>
<td>-32,768 to 32,768</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647 (Precision 10, scale 0)</td>
</tr>
<tr>
<td>Smallmoney</td>
<td>-214,748,3648 to 214,748,3647</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>Sysname</td>
<td>1 to 128 characters</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Text</td>
<td>1 to 2,147,483,647 characters</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Timestamp</td>
<td>8 bytes</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
</tbody>
</table>
Uniqueidentifier Data Type

Uniqueidentifier is a Microsoft SQL Server data type that is used to store Globally Unique Identifiers (GUIDs). It can store 16 bytes of data.

The Developer tool treats the Uniqueidentifier data type as String. To move or change Uniqueidentifier data, connect the Uniqueidentifier column to a String column. To successfully move or change Uniqueidentifier data, ensure that the data is in the following format:

```
xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

where `x` is a hexadecimal digit in the range 0-9 or a-f.

For example, `6F9619FF-8B86-D011-B42D-00C04FC964FF` is a valid Uniqueidentifier value.

The Developer tool can store 16 bytes of Uniqueidentifier data in 36 characters. However, since Uniqueidentifier data can be represented within two curly brackets, the Developer tool assigns two additional characters to the precision to accommodate the curly brackets. When you connect a Uniqueidentifier column to a String column, set the precision of the String column to 38 to successfully move or change Uniqueidentifier data.

Unsupported Microsoft SQL Server Data Types

The Developer tool does not support certain Microsoft SQL Server data types.

The Developer tool does not support the following Microsoft SQL Server data types:

- BigInt
- Datetime2
- Nchar
- Ntext
- Numeric Identity
- Nvarchar
- Sql_variant
ODBC and Transformation Datatypes

ODBC datatypes map to transformation datatypes that the Data Integration Service uses to move data across platforms.

The following table compares ODBC datatypes, such as Microsoft Access or Excel, to transformation datatypes:

<table>
<thead>
<tr>
<th>Datatype</th>
<th>Transformation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Precision 19, scale 0</td>
</tr>
<tr>
<td>Binary</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Bit</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Char</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Date</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>Decimal</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>Double</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Float</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Integer</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Precision 10, scale 0</td>
</tr>
<tr>
<td>Long Varbinary</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Nchar</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Nvarchar</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Ntext</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Numeric</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>Real</td>
<td>Double</td>
<td>Precision 15</td>
</tr>
<tr>
<td>Smallint</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Precision 10, scale 0</td>
</tr>
<tr>
<td>Text</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Time</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>Timestamp</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>Datatype</td>
<td>Transformation</td>
<td>Range</td>
</tr>
<tr>
<td>------------</td>
<td>----------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Tinyint</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Precision 10, scale 0</td>
</tr>
<tr>
<td>Varbinary</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Varchar</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
</tbody>
</table>

**Oracle and Transformation Data Types**

Oracle data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares Oracle data types and transformation data types:

<table>
<thead>
<tr>
<th>Oracle</th>
<th>Range</th>
<th>Transformation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blob</td>
<td>Up to 4 GB</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Char(L)</td>
<td>1 to 2,000 bytes</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Clob</td>
<td>Up to 4 GB</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Date</td>
<td>Jan. 1, 4712 B.C. to Dec. 31, 4712 A.D.</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>Float</td>
<td>Precision of 1 to 15, scale 0</td>
<td>Double</td>
<td>Precision of 15, scale 0</td>
</tr>
<tr>
<td>Long</td>
<td>Up to 2 GB</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If you include Long data in a mapping, the Integration Service converts it to the transformation String data type, and truncates it to 104,857,600 characters.</td>
</tr>
<tr>
<td>Long Raw</td>
<td>Up to 2 GB</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Nchar</td>
<td>1 to 2,000 bytes</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Oracle</td>
<td>Range</td>
<td>Transformation</td>
<td>Range</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------</td>
<td>----------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>Nclob</td>
<td>Up to 4 GB</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Number</td>
<td>Precision of 1 to 38</td>
<td>Double</td>
<td>Precision of 15</td>
</tr>
<tr>
<td>Number(P,S)</td>
<td>Precision of 1 to 38, scale of 0 to 38</td>
<td>Decimal</td>
<td>Precision of 1 to 28, scale of 0 to 28</td>
</tr>
<tr>
<td>Nvarchar2</td>
<td>1 to 4,000 bytes</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Raw</td>
<td>1 to 2,000 bytes</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>Timestamp</td>
<td>Jan. 1, 4712 B.C. to Dec. 31, 9999 A.D. Precision 19 to 29, scale 0 to 9 (precision to the nanosecond)</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>Varchar</td>
<td>1 to 4,000 bytes</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Varchar2</td>
<td>1 to 4,000 bytes</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>XMLType</td>
<td>Up to 4 GB</td>
<td>Text</td>
<td>1 to 104,857,600 characters</td>
</tr>
</tbody>
</table>

**Number(P,S) Data Type**

The Developer tool supports Oracle Number(P,S) values with negative scale. However, it does not support Number(P,S) values with scale greater than precision 28 or a negative precision.

If you import a table with an Oracle Number with a negative scale, the Developer tool displays it as a Decimal data type. However, the Data Integration Service converts it to a double.

**Char, Varchar, Clob Data Types**

When the Data Integration Service uses the Unicode data movement mode, it reads the precision of Char, Varchar, and Clob columns based on the length semantics that you set for columns in the Oracle database.

If you use the byte semantics to determine column length, the Data Integration Service reads the precision as the number of bytes. If you use the char semantics, the Data Integration Service reads the precision as the number of characters.
Unsupported Oracle Data Types

The Developer tool does not support certain Oracle data types.

The Developer tool does not support the following Oracle data types:

- Bfile
- Interval Day to Second
- Interval Year to Month
- Mslabel
- Raw Mslabel
- Rowid
- Timestamp with Local Time Zone
- Timestamp with Time Zone

XML and Transformation Datatypes

XML datatypes map to transformation datatypes that the Data Integration Service uses to move data across platforms.

The Data Integration Service supports all XML datatypes specified in the W3C May 2, 2001 Recommendation. However, the Data Integration Service may not support the entire XML value range. For more information about XML datatypes, see the W3C specifications for XML datatypes at the following location: [http://www.w3.org/TR/xmlschema-2](http://www.w3.org/TR/xmlschema-2).

The following table compares XML datatypes to transformation datatypes:

<table>
<thead>
<tr>
<th>Datatype</th>
<th>Transformation</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>anyURI</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>base64Binary</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>boolean</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>byte</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>date</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>dateTime</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>decimal</td>
<td>Decimal</td>
<td>Precision 1 to 28, scale 0 to 28</td>
</tr>
<tr>
<td>double</td>
<td>Double</td>
<td>Precision of 15 digits</td>
</tr>
<tr>
<td>duration</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>ENTITIES</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>ENTITY</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Datatype</td>
<td>Transformation</td>
<td>Range</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>float</td>
<td>Double</td>
<td>Precision of 15 digits</td>
</tr>
<tr>
<td>gDay</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>gMonth</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>gMonthDay</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>gYear</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>gYearMonth</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>hexBinary</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>ID</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>IDREF</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>IDREFS</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>int</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>integer</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>language</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>long</td>
<td>Bigint</td>
<td>-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807</td>
</tr>
<tr>
<td>Name</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>NCName</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>negativeInteger</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>NMTOKEN</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>NMTOKENS</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>nonNegativeInteger</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>nonPositiveInteger</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>normalizedString</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>NOTATION</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>positiveInteger</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>QName</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>short</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>string</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>Datatype</td>
<td>Transformation</td>
<td>Range</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td>time</td>
<td>Date/Time</td>
<td>Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)</td>
</tr>
<tr>
<td>token</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>unsignedByte</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>unsignedInt</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>unsignedShort</td>
<td>Integer</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
</tbody>
</table>

### Converting Data

You can convert data from one datatype to another.

To convert data from one datatype to another, use one of the following methods:

- Pass data between ports with different datatypes (port-to-port conversion).
- Use transformation functions to convert data.
- Use transformation arithmetic operators to convert data.

### Port-to-Port Data Conversion

The Data Integration Service converts data based on the datatype of the port. Each time data passes through a port, the Data Integration Service looks at the datatype assigned to the port and converts the data if necessary.

When you pass data between ports of the same numeric datatype and the data is transferred between transformations, the Data Integration Service does not convert the data to the scale and precision of the port that the data is passed to. For example, you transfer data between two transformations in a mapping. If you pass data from a Decimal port with a precision of 5 to a Decimal port with a precision of 4, the Data Integration Service stores the value internally and does not truncate the data.

When you pass data between ports with different datatypes, the Data Integration Service uses the conversion functions in the transformation language to convert the data. For example, you link a String port to an Integer port. When the Data Integration Service runs the mapping, it uses the TO_INTEGER function to convert the String to an Integer.

When the Data Integration Service performs port-to-port conversions, the data that you pass must be valid for the conversion datatype. Any values that cannot be converted result in a transformation row error. For example, you link a String port that contains the value "9,000,000,000,000,000,000,777" to a Bigint port. The Data Integration Service cannot convert the String to a Bigint value and returns an error.

The Data Integration Service performs port-to-port conversions between transformations and between the last transformation in a dataflow and a target.
The following table describes the port-to-port conversions that the Data Integration Service performs:

<table>
<thead>
<tr>
<th>Datatype</th>
<th>Bigint</th>
<th>Integer</th>
<th>Decimal</th>
<th>Double</th>
<th>String, Text</th>
<th>Date/Time</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bigint</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Integer</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Decimal</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Double</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>String, Text</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Date/Time</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Binary</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Keyboard Shortcuts

This appendix includes the following topics:

- **Keyboard Shortcuts for Objects, 176**
- **Keyboard Shortcuts for Ports, 177**
- **Keyboard Shortcuts for the Transformation Palette, 178**
- **Keyboard Shortcuts for the Workbench, 178**

Keyboard Shortcuts for Objects

You can use keyboard shortcuts to work with objects that are in the editor.

When you select an object in the editor, you can change the appearance of the object and move the object. When an object is selected, a green border appears around the object. A dotted line also appears around a selected object indicating that the object is in focus. You can focus on an object while allowing other objects to be selected. Move the focus from a selected object to another object when you want to select multiple objects in the editor.

**Note**: The navigation order of objects in the editor is from top to bottom and left to right.

You can perform the following tasks with keyboard shortcuts:

**Select an object.**

When the editor is selected, press `Tab`. The object is also in focus. Press `Tab` again to select the next object.

**Select the previous object.**

Press `Shift+Tab`.

**Find a particular object.**

Press `Ctrl+O`.

**Focus on the next object.**

When an object is in focus, press `Ctrl+Tab` to move the focus to the next object. The previous object remains selected. Press `Ctrl+Tab` again to focus on the next object.

**Select multiple objects individually.**

When an object is in focus, press `Ctrl+Tab` to move the focus to the next object that you want to select. Then press `Ctrl+Spacebar` to select the focused object. Repeat these steps to select multiple objects individually.
Iconize selected objects.
Press Ctrl+I.

Restore selected objects.
Press Ctrl+R.

Resize selected objects.
Press and hold Ctrl+Shift, and then use the Up, Down, Left, and Right arrow keys.

Move selected objects.
Use the arrow keys to move one or more objects one pixel at a time in any direction.

Go from an object to the editor.
When one or multiple objects are selected or in focus, press Esc.

Keyboard Shortcuts for Ports

You can use keyboard shortcuts to work with ports.

When you select a port, you can edit the port and link the port to other ports. When a port is selected, the entire port appears green. A dotted line also appears around one port value of the selected port indicating that the port is in focus. You can focus on a port while allowing other ports to be selected. Move the focus from one port to another port when you want to select multiple ports individually.

You can perform the following tasks with keyboard shortcuts:

Select a port.
When an object is selected and in focus, press Ctrl+G. Use the Up and Down arrow keys to select a different port.

Select multiple ports.
Press Shift+Up or Shift+Down.

Select multiple ports individually.
Press and hold the Ctrl key, and then use the Up and Down arrow keys to focus on the port that you want to select. Then press Ctrl+Spacebar to select the port. Repeat these steps to select multiple ports individually.

Focus on and edit the next port value.
Press Tab.

Focus on and edit the previous port value.
Press Shift+Tab.

Go from a port to the object.
Press Esc.

Link ports in two objects.
Select the ports that you want to link in the first object, and then press Ctrl+L. The selected ports become highlighted. Navigate to the second object and select the ports that you want to link. Press Ctrl +L to link the ports.
End linking mode.

Press **Ctrl+Alt+L** to clear the ports that you selected in the first object.

Select a link.

Press **Ctrl+G** from the selected port to select the outgoing link from that port.

Select the next link.

Press **Tab**.

Select the previous link.

Press **Shift+Tab**.

---

**Keyboard Shortcuts for the Transformation Palette**

You can use keyboard shortcuts to navigate the Transformation palette.

You can perform the following tasks with keyboard shortcuts:

**Go from the editor to the Transformation palette.**

When an editor is selected, press **Ctrl+Shift+P**.

**Navigate the Transformation palette.**

Press **Tab**, or use the **Up** and **Down** arrow keys.

**Go from the Transformation palette to the editor.**

Press **Esc**.

---

**Keyboard Shortcuts for the Workbench**

You can use keyboard shortcuts to navigate editors and views in the workbench.

When you select an editor, you can navigate within the editor. A green border appears around an editor when it is selected.

You can focus on editors and views. A blue border appears around an editor or view when it is in focus. A dotted line appears around the tab of a view that is in focus. When a view is in focus, you can select the view or you can focus on another view.

You can perform the following tasks with keyboard shortcuts:

**Focus on a view.**

When an editor is selected, press **Shift+Tab**. Then use the **Left** and **Right** arrow keys to focus on another view within the same editor. Alternatively, press **Shift+Tab** until another view in the workbench is in focus.

**Select the first area in a view.**

When a view is in focus in an editor, press **Ctrl+Tab**. Press **Ctrl+Tab** again to select the next area.

**Select the previous area in a view.**

Press **Ctrl+Shift+Tab**.
Minimize an editor.

When a view is in focus, press **Shift+Tab** to select the **Minimize** control, and then press the spacebar.

Maximize an editor.

When a view is in focus, press **Shift+Tab** to select the **Minimize** control. Use the **Right** arrow key to select the **Maximize** control, and then press the spacebar.

Select an area of the workbench.

When an editor is selected, press **Ctrl+Tab** to select another area of the workbench, such as a view. Press **Ctrl+Tab** again to select the next area.

Select the previous area of the workbench.

Press **Ctrl+Shift+Tab**.
INDEX

A
active transformations
description 126
applications
creating 149
mapping deployment properties 147
redeploying 151
updating 150, 151
attributes
relationships 121

B
bigint
constants in expressions 160
high precision handling 159
using in calculations 159
writing to flat files 161
binary datatypes
overview 161

certificates
adding untrusted certificates 93
certificate properties 93
managing certificates 92
untrusted certificates 92
cheat sheets
description 9
column profile
options 100
overview 98
process 100
column profile results
column profile 103
configurations
troubleshooting 140
connections
Connection Explorer view 28
creating 29, 30
deleting 27
editing 27
overview 27
renaming 27
web services properties 51
copy
description 21
objects 21
curation
Informatica Developer 106

custom queries
Informatica join syntax 70
left outer join syntax 72
normal join syntax 70
outer join support 69
right outer join syntax 73
custom SQL queries
creating 75
customized data objects 63
customized data objects
adding pre- and post-mapping SQL commands 74
adding relational data objects 61
adding relational resources 61
advanced query 63
creating 60
creating a custom query 75
creating key relationships 62
creating keys 62
custom SQL queries 63
default query 63
description 57
entering source filters 67
entering user-defined joins 69
key relationships 59
pre- and post-mapping SQL commands 74
reserved words file 63
select distinct 66
simple query 63
sorted ports 67
troubleshooting 96
user-defined joins 68
using select distinct 66
using sorted ports 67
write properties 60

D
Data Integration Service
selecting 13
data object profiles
creating a single profile 103
data types
Microsoft SQL Server 167
Oracle 170
data viewer
configuration properties 134
configurations 134, 137
creating configurations 137
troubleshooting configurations 140
database hints
entering in Developer tool 66
DataSift connections
properties 33
datatypes
Bigint 159
datatypes (continued)
  binary 161
  Date/Time 161
decimal 162
double 162
  flat file 163
  IBM DB2 164
  implicit conversion 174
  Integer 159
  JDBC 165
  ODBC 169
  overview 158
  port-to-port data conversion 174
  string 153
  transformation 159
  XML 172
Date/Time datatypes
  overview 161
decimal
  high precision handling 159, 162
decimal datatypes
  overview 162
default SQL query
  viewing 75
deployment
  mapping properties 147
  overview 146
  redeploying an application 151
  replacing applications 151
  to a Data Integration Service 149
  to file 150
  updating applications 161
Developer tool
  workspace directory 6
double
  high precision handling 162
double datatypes
  overview 162
E
error messages
  grouping 144
  limiting 145
export
  dependent objects 153
  objects 154
  overview 152
  XML file 154
Expression Editor
  description 131
  validating expressions 132
expressions
  adding comments 132
  adding to a port 131
  entering 131
  in transformations 130
  validating 132
F
Facebook connections
  properties 34
  filtering object dependencies
  object dependencies 142
flat file data objects
  advanced properties 86
  column properties 76
  configuring read properties 81
  configuring write properties 85
  creating 86
  delimited, importing 88
  description 75
  fixed-width, importing 87
  general properties 76
  read properties 77, 82
  reject files 94
  flat file targets
    reject files 94
  folders
    creating 21
    description 20
G
Greenplum connections
  properties 34
H
high precision
  Bigint datatype 159
  Decimal datatype 159
  hints
    Query view 66
IBM DB2 connections
  properties 36
  identifying relationships
    description 121
  import
    application archives 150
    dependent objects 153
    objects 156
    overview 152
    XML file 154
  Informatica Developer
    overview 5
    profile views 99
    searches 24
    setting up 9
    starting 8
  Informatica Marketplace
    description 9
  integers
    constants in expressions 160
    using in calculations 159
    writing to flat files 161
J
JDBC connections
  properties 38
  join syntax
    customized data objects 70
    Informatica syntax 70
    left outer join syntax 72
join syntax (continued)
    normal join syntax 70
    right outer join syntax 73

K
key relationships
    creating between relational data objects 56
    creating in customized data objects 62
    customized data objects 59
    relational data objects 56
keyboard shortcuts
    objects 176
    ports 177
    Transformation palette 178
    workbench 178

L
LinkedIn connections
    properties 41
local workspace directory
    configuring 6
logical data object mappings
    creating 125
    read mappings 124
    types 124
    write mappings 124
logical data object models
    creating 112
    description 112
    example 111
    importing 113
logical data objects
    attribute relationships 121
    creating 122
    description 121
    example 111
    properties 121
logical view of data
    developing 111
    overview 110
logs
    description 143

M
mapping object
    running a profile 109
mappings
    configuration properties 134
    configurations 134, 138
    creating configurations 139
    deployment properties 147
    reject files 94
    troubleshooting configurations 140
Mapplet and Mapping Profiles
    Overview 108
Model repository
    connecting 16
    description 14
    objects 14
    search 24
    searching for objects and properties 25
Model Repository Service
    refreshing 16
    monitoring
        description 145
MS SQL Server connections
    properties 41
N
NaN
    described 161
non-identifying relationships
    description 121
O
object dependencies
    view object dependencies 141
Object Dependencies view
    viewing data 141
objects
    copying 21
    ODBC connections
        properties 44
    Oracle connections
        properties 46
    outer join support
    customized data objects 69
    overview
        transformations 126
partitioning
    reject files 94
passive transformations
    description 127
performance tuning
    creating data viewer configurations 137
    creating mapping configurations 139
    data viewer configurations 137
    mapping configurations 138
    physical data objects
        customized data objects 57
        description 53
        flat file data objects 75
        relational data objects 54
        synchronization 93
        troubleshooting 96
    pre- and post-mapping SQL commands
        adding to relational data objects 74
        customized data objects 74
    primary keys
        creating in customized data objects 62
        creating in relational data objects 55
    profile results
        approving datatypes in Informatica Developer 106
        column datatypes 105
        exporting in Informatica Developer 108
        rejecting datatypes in the Developer tool 107
        profile results export
        Excel 107
    project permissions
        allowing parent object access 19
        assigning 20
project permissions (continued)
  dependent object instances 18
  external object permissions 18
  grant permission 18
  read permission 18
  showing security details 19
  write permission 18
projects
  assigning permissions 20
  creating 17
  description 16
  filtering 17
  permissions 18
  sharing 16

QNAN
  converting to 1.#QNAN 161
Query view
  configuring hints 66

read transformations
  creating from relational data objects 56
  reject file
    column indicators 96
    row indicators 95
  reject files
    locating 94
    partitioning 94
    reading 95
    targets 94
    viewing 94
relational connections
  adding to customized data objects 61
relational data objects
  adding to customized data objects 61
  creating key relationships 56
  creating keys 55
  creating read transformations 56
  description 54
  importing 57
  key relationships 55
  reject files 94
  troubleshooting 96
relational targets
  reject files 94
reserved words file
  creating 64
reusable transformations
  description 129
  editing 129
row indicators
  reject file 95
rules
  applying in PowerCenter Express 101
  creating in Informatica Developer 101
  overview 101
  prerequisites 102

S
search
  overview 24
  editor 25
Model repository 24
select distinct
  customized data objects 66
  using in customized data objects 66
self-joins
  custom SQL queries 63
sorted ports
  customized data objects 67
  using in customized data objects 67
source filters
  entering 67
SQL hints
  entering in Developer tool 66
string datatypes
  overview 163
synchronization
  customized data objects 93
physical data objects 93

T
transformation datatypes
  list of 159
transformations
  active 126
  connected 127
  creating 132
  developing 128
  editing reusable 129
  expression validation 132
  expressions 130
  passive 127
  reusable 129
  unconnected 127
Twitter connections
  properties 48
Twitter Streaming connections
  properties 49

U
user-defined joins
  customized data objects 68
  entering 69
Informatica syntax 70
left outer join syntax 72
normal join syntax 70
outer join support 69
right outer join syntax 73

V
validation
  configuring preferences 144
  grouping error messages 144
  limiting error messages 145
view object dependencies
  filtering object dependencies 142
  object dependencies 141
W

Web content-Kapow Katalyst connections
  properties 50
web service
  configuration properties 134

Welcome page
  description 8
workspace directory
  configuring 6
WSDL data objects
  advanced view 91
importing 89
overview view 90
schema view 89
synchronization 92