



Informatica®

10.5.2

Developer Tool Guide

Informatica Developer Tool Guide
10.5.2
April 2022

© Copyright Informatica LLC 2009, 2023

This software and documentation are provided only under a separate license agreement containing restrictions on use and disclosure. No part of this document may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without prior consent of Informatica LLC.

Informatica, the Informatica logo, PowerCenter, and PowerExchange are trademarks or registered trademarks of Informatica LLC in the United States and many jurisdictions throughout the world. A current list of Informatica trademarks is available on the web at <https://www.informatica.com/trademarks.html>. Other company and product names may be trade names or trademarks of their respective owners.

Subject to your opt-out rights, the software will automatically transmit to Informatica in the USA information about the computing and network environment in which the Software is deployed and the data usage and system statistics of the deployment. This transmission is deemed part of the Services under the Informatica privacy policy and Informatica will use and otherwise process this information in accordance with the Informatica privacy policy available at <https://www.informatica.com/in/privacy-policy.html>. You may disable usage collection in Administrator tool.

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation is subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License.

Portions of this software and/or documentation are subject to copyright held by third parties. Required third party notices are included with the product.

See patents at <https://www.informatica.com/legal/patents.html>.

The information in this documentation is subject to change without notice. If you find any problems in this documentation, report them to us at infa_documentation@informatica.com.

Informatica products are warranted according to the terms and conditions of the agreements under which they are provided. INFORMATICA PROVIDES THE INFORMATION IN THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

Publication Date: 2023-07-24

Table of Contents

Preface	13
Informatica Resources.	13
Informatica Network.	13
Informatica Knowledge Base.	13
Informatica Documentation.	13
Informatica Product Availability Matrices.	14
Informatica Velocity.	14
Informatica Marketplace.	14
Informatica Global Customer Support.	14
Chapter 1: Informatica Developer	15
Informatica Developer Overview.	15
Informatica Data Quality and Profiling.	15
Informatica Data Services.	17
Start Informatica Developer.	18
Starting the Developer Tool on a Local Machine.	18
Starting the Developer Tool on a Remote Machine.	18
Informatica Developer Interface.	19
Informatica Developer Views.	19
Informatica Developer Welcome Page.	21
Cheat Sheets.	21
Informatica Developer Online Help.	21
Informatica Preferences.	22
Informatica Marketplace.	22
Setting Up Informatica Developer.	22
Step 1. Add a Domain.	23
Step 2. Add a Model Repository.	23
Step 3. Select a Default Data Integration Service.	23
Step 4. Select a Default Metadata Access Service.	24
Domains.	24
Projects.	24
Creating a Project.	25
Filter Projects.	25
Project Permissions.	26
Permissions for External Objects.	26
Permissions for Dependent Object Instances.	27
Parent Object Access.	27
Assigning Permissions.	28
Folders.	28
Creating a Folder.	29

Copy Object Operations.	29
Copying an Object.	30
Saving a Copy of an Object.	30
Tags.	30
Creating a Tag.	31
Assigning a Tag.	31
Viewing Tags.	31
Chapter 2: The Model Repository.	32
Model Repository Overview.	32
Objects in Informatica Developer.	33
Object Properties.	35
Repository Object Locks.	36
Lock Management.	36
Rules and Guidelines for Lock Management.	36
Team-based Development with Versioned Objects.	37
Versioned Object Management.	38
Historical Versions of Objects.	40
Checked Out Objects View.	40
Version History View.	41
Troubleshooting Team-based Development.	41
Connecting to a Model Repository.	41
Model Repository Service Refresh.	42
Chapter 3: Searches in Informatica Developer.	43
Searches in Informatica Developer Overview.	43
Domain Search.	43
Searching for Objects and Properties.	44
Business Glossary Search.	45
Business Glossary Desktop Lookup.	45
Looking Up a Business Term.	45
Customizing Hotkeys to Look Up a Business Term.	46
Editor Search.	46
Chapter 4: Connections.	48
Connections Overview.	48
Connection Types	50
Connection Explorer View.	50
Connection Management.	51
Creating a Connection	51
Editing a Connection.	52
Copying a Connection.	52
Deleting a Connection.	53

Refreshing the Connections List.	53
Connection Switching.	53
Before You Switch a Connection.	54
Switching a Connection.	55
After You Switch a Connection.	55
Third-Party JDBC Drivers.	57
Environment SQL.	57
Connection Environment SQL.	58
Transaction Environment SQL.	58
Guidelines for Configuring Environment SQL.	58
Chapter 5: Physical Data Objects.	59
Physical Data Objects Overview.	59
Physical Data Object Types.	60
Relational Data Objects.	60
Importing a Relational Data Object.	62
Key Relationships.	62
Customized Data Objects.	64
Key Relationships.	65
Customized Data Object Write Properties.	66
Creating a Customized Data Object.	67
Adding Relational Resources to a Customized Data Object.	67
Adding Relational Data Objects to a Customized Data Object.	68
Creating Keys in a Customized Data Object.	68
Creating Relationships within a Customized Data Object.	69
Create or Replace Target Tables.	69
Rules and Guidelines to Create or Replace Target Tables.	70
Generating and Executing DDL at Design-Time.	70
Generating and Executing DDL at Run Time	72
DDL Generation Errors.	72
Custom Queries.	72
Custom Query Optimization.	73
Default Query.	73
Hints.	74
Select Distinct.	76
Filters.	76
Sorted Ports.	77
User-Defined Joins.	78
Outer Join Support.	79
Informatica Join Syntax.	79
PreSQL and PostSQL Commands.	83
Creating a Custom Query.	84
Nonrelational Data Objects.	85

Importing Nonrelational Data Objects.	85
Creating a Read, Write, or Lookup Transformation from Nonrelational Data Operations.	86
WSDL Data Object.	86
WSDL Data Object Overview View.	87
WSDL Data Object Advanced View.	87
Importing a WSDL Data Object.	87
WSDL Synchronization.	88
Certificate Management.	89
Synchronization.	90
Synchronizing a Flat File Data Object in Informatica Developer.	90
Synchronizing a Relational Data Object in Informatica Developer.	92
Troubleshooting Physical Data Objects.	93
Chapter 6: Flat File Data Objects.	94
Flat File Data Objects Overview.	94
Generate the Source File Name.	95
Flat File Data Object Overview Properties.	95
Flat File Data Object Advanced Properties.	96
Format Properties.	97
Column Format: Delimited Properties.	98
Column Format: Fixed-width Properties.	99
Run-time: Read Properties.	100
Run-time: Write Properties.	102
Control File.	103
Update Columns at Run Time.	104
Generate Run-time Column Names Automatically.	104
Generate Run-time Column Names From Data File Header.	105
Generate Column Metadata from Control Files.	105
Control File Formats.	106
Parameterization of Run-time Properties.	106
Run-time Processing of Control Files.	107
Rules and Guidelines for Control Files.	107
Copying from Excel to a Flat File Data Object.	108
Editing Flat File Data Objects in Excel.	108
Copying Metadata to a Flat File Data Object.	108
Example, Editing Data Object in Excel.	108
Create a Flat File Data Object.	109
Creating an Empty Flat File Data Object.	109
Creating a Flat File Data Object from an Existing Flat File.	110
Creating a Flat File Data Object from a Control File.	112
Chapter 7: Logical View of Data.	113
Logical View of Data Overview.	113

Logical Data Object Model Example.	114
Developing a Logical View of Data.	114
Logical Data Object Models.	115
Creating a Logical Data Object Model.	115
Importing a Logical Data Object Model from a Modeling Tool.	116
Logical Data Object Model Properties.	116
CA ERwin Data Modeler Import Properties.	117
IBM Cognos Business Intelligence Reporting - Framework Manager Import Properties.	118
SAP BusinessObjects Designer Import Properties.	119
SAP PowerDesigner CDM Import Properties.	120
SAP PowerDesigner PDM Import Properties.	121
XSD Import Properties.	122
Logical Data Objects.	123
Logical Data Object Properties.	123
Attribute Relationships.	124
Creating a Logical Data Object.	124
Logical Data Object Mappings.	127
Logical Data Object Read Mappings.	128
Logical Data Object Write Mappings.	128
Creating a Logical Data Object Mapping.	128
Chapter 8: Viewing Data.	130
Viewing Data Overview.	130
Configurations.	131
Configuration Properties.	131
Data Viewer Configurations.	135
Mapping Configurations.	137
Web Service Configurations.	138
Updating the Default Configuration Properties.	138
Troubleshooting Configurations.	139
Exporting Data.	139
Object Dependencies.	140
View Object Dependencies.	140
Viewing Object Dependencies.	140
Filtering Object Dependencies.	141
Logs.	141
Log File Format.	142
Validation Preferences.	142
Grouping Error Messages.	142
Limiting Error Messages.	143
Monitoring Jobs from the Developer Tool.	143

Chapter 9: Application Deployment.....	144
Application Deployment Overview.	144
Application Creation.	145
Full Application.	145
Incremental Application.	146
Application Properties.	146
Deployment to an Application.	148
Deployment to an Archive File.	150
Deployment with Resource Parameters.	150
Viewing Run-time Objects.	152
Application Redeployment.	152
Application State Information.	153
How to Create, Deploy, and Update an Application.	154
Creating an Application.	155
Deploying an Application.	157
Deploying an Object.	158
Importing Application Archives.	159
Connecting to a Run-time Application.	159
Editing an Application.	160
Redeploying an Application to a Data Integration Service.	160
Chapter 10: Application Patch Deployment.....	162
Application Patch Deployment Overview.	162
Application Object Instances.	163
Application Object Dependencies.	163
Incremental Deployment Wizard.	166
Select Applications.	167
Source Object Selection.	168
Patch Deployment Strategy.	170
Choose the Application Patch Type.	170
Inheriting Only Direct Dependencies.	172
Inheriting Direct and Indirect Dependencies.	173
Inheriting Direct, Indirect, and Remote Dependencies.	174
Determine Application Object Impact.	176
Preview Object Impact.	177
Identify Affected Objects.	177
Identify the Scope of Object Impact.	180
Specify a Patch Deployment Strategy.	180
Patch Deployment Strategy Properties.	181
Deploying an Application Patch.	182
Rules and Guidelines for Application Patch Deployment.	184

Chapter 11: Application Patch Deployment Example.....	185
Application Patch Deployment Example Overview.	185
Patch Type: Direct.	187
Patch Type: Direct and Indirect.	189
Patch Type: Direct, Indirect, and Remote.	191
Conclusion.	193
Chapter 12: Continuous Integration and Continuous Delivery (CI/CD).....	194
CI/CD Overview.	194
REST API and infacmd Quick Reference.	195
Using the REST API or infacmd.	197
Using the REST API to Deploy and Redeploy an Application.	197
Continuous Integration.	198
Deploy Objects.	198
Assign Objects.	198
Resolve Object Conflicts.	199
Continuous Delivery.	199
Sample Automated Integration Process.	200
Step 1. Prepare Mappings for Deployment.	200
Step 2. Deploy Mappings to a Data Integration Service.	201
Step 3. Run Sanity Tests.	201
Step 4. Prepare Mappings for Editing.	202
Sample Automated Delivery Process.	202
Chapter 13: Object Import and Export.....	204
Object Import and Export Overview.	204
Import and Export Objects.	205
Object Export.	206
Exporting Objects.	206
Object Import.	207
Importing Projects.	207
Importing Objects.	209
Importing Objects from a Previous Informatica Release.	210
Appendix A: Data Type Reference.....	211
Data Type Reference Overview.	211
Transformation Data Types.	212
Integer Data Types.	213
Binary Data Type.	215
Date/Time Data Type.	215
Decimal and Double Data Types.	217
String Data Types.	219

Complex Data Types.	220
Complex File and Transformation Data Types.	223
Avro Data Types and Transformation Data Types.	224
JSON Data Types and Transformation Data Types.	226
ORC Data Types and Transformation Data Types.	226
Parquet Data Types and Transformation Data Types.	227
Flat File and Transformation Data Types.	229
DB2 for LUW and Transformation Data Types.	230
DB2 for i5/OS, DB2 for z/OS, and Transformation Datatypes.	232
Unsupported DB2 for i5/OS and DB2 for z/OS Datatypes.	233
JDBC and Transformation Datatypes.	233
Microsoft SQL Server and Transformation Data Types.	234
Uniqueidentifier Data Type.	236
Nonrelational and Transformation Datatypes.	237
ODBC and Transformation Data Types.	239
Oracle and Transformation Data Types.	240
Number(P,S) Data Type.	242
Char, Varchar, Clob Data Types	242
Unsupported Oracle Data Types.	243
SAP HANA and Transformation Datatypes.	243
XML and Transformation Datatypes.	244
Converting Data.	246
Port-to-Port Data Conversion.	247
Appendix B: Keyboard Shortcuts.	249
Keyboard Shortcuts for Advanced Mapping Options.	249
Keyboard Shortcuts for Objects.	250
Keyboard Shortcuts for Ports.	251
Keyboard Shortcuts for the Transformation Palette.	252
Keyboard Shortcuts for the Workbench.	252
Appendix C: Connection Properties.	254
Connection Properties Overview.	255
Adabas Connection Properties.	256
Amazon Redshift Connection Properties.	258
Amazon S3 Connection Properties.	260
Blockchain Connection Properties.	262
Cassandra Connection Properties.	264
Confluent Kafka Connection.	265
General Properties.	266
Confluent Kafka Broker Properties.	266
SSL Properties	267
Creating a Confluent Kafka Connection Using infacmd.	267

Databricks Connection Properties.	267
Greenplum Connection Properties.	269
Google Analytics Connection Properties.	271
Google BigQuery Connection Properties.	271
Connection Modes.	273
Google Cloud Spanner Connection Properties.	273
Google Cloud Storage Connection Properties.	274
Google PubSub Connection Properties.	275
Hadoop Connection Properties.	276
Hadoop Cluster Properties.	277
Common Properties.	278
Reject Directory Properties.	280
Blaze Configuration.	280
Spark Configuration.	281
HBase Connection Properties.	282
HDFS Connection Properties.	283
HBase Connection Properties for MapR-DB.	285
Hive Connection Properties.	285
HTTP Connection Properties.	289
IBM DB2 Connection Properties.	291
IBM DB2 for i5/OS Connection Properties.	294
IBM DB2 for z/OS Connection Properties.	297
IMS Connection Properties.	300
JDBC Connection Properties.	302
JDBC V2 Connection Properties.	305
JD Edwards EnterpriseOne Connection Properties.	307
Kafka Connection Properties.	308
General Properties.	309
Kafka Broker Properties.	309
SSL Properties.	310
Creating a Kafka Connection Using infacmd.	311
Kudu Connection Properties	311
LDAP Connection Properties.	312
Microsoft Azure Blob Storage Connection Properties.	313
Microsoft Azure Cosmos DB SQL API Connection Properties.	314
Microsoft Azure Data Lake Storage Gen1 Connection Properties.	315
Microsoft Azure Data Lake Storage Gen2 Connection Properties.	316
Microsoft Azure SQL Data Warehouse Connection Properties.	317
MS SQL Server Connection Properties.	319
Netezza Connection Properties.	323
OData Connection Properties.	324
ODBC Connection Properties.	325

Oracle Connection Properties.	326
Salesforce Connection Properties.	329
Salesforce Marketing Cloud Connection Properties.	330
SAP Connection Properties.	331
Sequential Connection Properties.	334
Snowflake Connection Properties.	336
Teradata Parallel Transporter Connection Properties.	337
Tableau Connection Properties.	339
Tableau V3 Connection Properties.	340
Twitter Streaming Connection Properties.	341
VSAM Connection Properties.	342
Web Services Connection Properties.	345
Identifier Properties in Database Connections.	346
Regular Identifiers.	347
Delimited Identifiers.	347
Identifier Properties.	347
Index.	350

Preface

Use the *Informatica® Developer Tool Guide* to learn about the user interface in the Developer tool. Learn how to connect to a Model repository, create connections and data objects, and deploy applications.

Informatica Resources

Informatica provides you with a range of product resources through the Informatica Network and other online portals. Use the resources to get the most from your Informatica products and solutions and to learn from other Informatica users and subject matter experts.

Informatica Network

The Informatica Network is the gateway to many resources, including the Informatica Knowledge Base and Informatica Global Customer Support. To enter the Informatica Network, visit <https://network.informatica.com>.

As an Informatica Network member, you have the following options:

- Search the Knowledge Base for product resources.
- View product availability information.
- Create and review your support cases.
- Find your local Informatica User Group Network and collaborate with your peers.

Informatica Knowledge Base

Use the Informatica Knowledge Base to find product resources such as how-to articles, best practices, video tutorials, and answers to frequently asked questions.

To search the Knowledge Base, visit <https://search.informatica.com>. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team at KB_Feedback@informatica.com.

Informatica Documentation

Use the Informatica Documentation Portal to explore an extensive library of documentation for current and recent product releases. To explore the Documentation Portal, visit <https://docs.informatica.com>.

If you have questions, comments, or ideas about the product documentation, contact the Informatica Documentation team at infa_documentation@informatica.com.

Informatica Product Availability Matrices

Product Availability Matrices (PAMs) indicate the versions of the operating systems, databases, and types of data sources and targets that a product release supports. You can browse the Informatica PAMs at <https://network.informatica.com/community/informatica-network/product-availability-matrices>.

Informatica Velocity

Informatica Velocity is a collection of tips and best practices developed by Informatica Professional Services and based on real-world experiences from hundreds of data management projects. Informatica Velocity represents the collective knowledge of Informatica consultants who work with organizations around the world to plan, develop, deploy, and maintain successful data management solutions.

You can find Informatica Velocity resources at <http://velocity.informatica.com>. If you have questions, comments, or ideas about Informatica Velocity, contact Informatica Professional Services at ips@informatica.com.

Informatica Marketplace

The Informatica Marketplace is a forum where you can find solutions that extend and enhance your Informatica implementations. Leverage any of the hundreds of solutions from Informatica developers and partners on the Marketplace to improve your productivity and speed up time to implementation on your projects. You can find the Informatica Marketplace at <https://marketplace.informatica.com>.

Informatica Global Customer Support

You can contact a Global Support Center by telephone or through the Informatica Network.

To find your local Informatica Global Customer Support telephone number, visit the Informatica website at the following link:

<https://www.informatica.com/services-and-training/customer-success-services/contact-us.html>.

To find online support resources on the Informatica Network, visit <https://network.informatica.com> and select the eSupport option.

CHAPTER 1

Informatica Developer

This chapter includes the following topics:

- [Informatica Developer Overview, 15](#)
- [Start Informatica Developer, 18](#)
- [Informatica Developer Interface, 19](#)
- [Setting Up Informatica Developer, 22](#)
- [Domains, 24](#)
- [Projects, 24](#)
- [Project Permissions, 26](#)
- [Folders, 28](#)
- [Copy Object Operations, 29](#)
- [Tags, 30](#)

Informatica Developer Overview

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

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

Informatica Data Quality and Profiling

Use the data quality capabilities in the Developer tool to analyze the content and structure of your data and enhance the data in ways that meet your business needs.

Use the Developer tool to design and run processes to complete the following tasks:

- Profile data. Profiling reveals the content and structure of data. Profiling is a key step in any data project as it can identify strengths and weaknesses in data and help you define a project plan.
- Create scorecards to review data quality. A scorecard is a graphical representation of the quality measurements in a profile.
- Standardize data values. Standardize data to remove errors and inconsistencies that you find when you run a profile. You can standardize variations in punctuation, formatting, and spelling. For example, you can ensure that the city, state, and ZIP code values are consistent.

- Parse data. Parsing reads a field composed of multiple values and creates a field for each value according to the type of information it contains. Parsing can also add information to records. For example, you can define a parsing operation to add units of measurement to product data.
- Validate postal addresses. Address validation evaluates and enhances the accuracy and deliverability of postal address data. Address validation corrects errors in addresses and completes partial addresses by comparing address records against address reference data from national postal carriers. Address validation can also add postal information that speeds mail delivery and reduces mail costs.
- Find duplicate records. Duplicate analysis calculates the degrees of similarity between records by comparing data from one or more fields in each record. You select the fields to be analyzed, and you select the comparison strategies to apply to the data. The Developer tool enables two types of duplicate analysis: field matching, which identifies similar or duplicate records, and identity matching, which identifies similar or duplicate identities in record data.
- Manage exceptions. An exception is a record that contains data quality issues that you correct by hand. You can run a mapping to capture any exception record that remains in a data set after you run other data quality processes. You review and edit exception records in the Analyst tool.
- Create reference data tables. Informatica provides reference data that can enhance several types of data quality process, including standardization and parsing. You can create reference tables using data from profile results.
- Create and run data quality rules. Informatica provides rules that you can run or edit to meet your project objectives. You can create maplets and validate them as rules in the Developer tool.
- Collaborate with Informatica users. The Model repository stores reference data and rules, and this repository is available to users of the Developer tool and Analyst tool. Users can collaborate on projects, and different users can take ownership of objects at different stages of a project.
- Use the Developer tool or infacmd to import mappings from PowerCenter or export mappings to PowerCenter to reuse the metadata.

To import data from PowerCenter into the Model Repository, complete the following tasks:

1. Export PowerCenter objects to a file using the PowerCenter Client or with the following command:
`pmrep ExportObject`
2. Convert the export file to a Model repository file using the following command:
`infacmd ipc importFromPC`
3. Import the objects using the Developer tool or with the following command:
`infacmd tools importObjects`

To export data from the Model repository into the PowerCenter repository, complete the following tasks:

1. Export Model Repository objects to a file using the Developer tool or with the following command:
`infacmd tools ExportObjects`
Or, you can directly run `infacmd ipc ExportToPC` to export.
2. Convert the export file to a PowerCenter file using the following command:
`infacmd ipc ExporttoPC`
3. Import the objects using the PowerCenter or with the following command:
`pmrep importObjects`

Informatica Data Services

Data services are a collection of reusable operations that you can run to access and transform data.

Use the data services capabilities in the Developer tool to complete the following tasks:

- Define logical views of data. A logical view of data describes the structure and use of data in an enterprise. You can create a logical data object model that shows the types of data your enterprise uses and how that data is structured.
- Map logical models to data sources or targets. Create a mapping that links objects in a logical model to data sources or targets. You can link data from multiple, disparate sources to create a single view of the data. You can also load data that conforms to a model to multiple, disparate targets.
- Create virtual views of data. You can deploy a virtual federated database to a Data Integration Service. End users can run SQL queries against the virtual data without affecting the actual source data.
- Provide access to data integration functionality through a web service interface. You can deploy a web service to a Data Integration Service. End users send requests to the web service and receive responses through SOAP messages.
- Use the Developer tool or `infacmd` to import mappings from PowerCenter or export mappings to PowerCenter to reuse the metadata.

To import data from PowerCenter into the Model Repository, complete the following tasks:

1. Export PowerCenter objects to a file using the PowerCenter Client or with the following command:
`pmrep ExportObject`
2. Convert the export file to a Model repository file using the following command:
`infacmd ipc importFromPC`
3. Import the objects using the Developer tool or with the following command:
`infacmd tools importObjects`

To export data from the Model repository into the PowerCenter repository, complete the following tasks:

1. Export Model Repository objects to a file using the Developer tool or with the following command:
`infacmd tools ExportObjects`
Or, you can directly run `infacmd ipc ExportToPC` to export.
 2. Convert the export file to a PowerCenter file using the following command:
`infacmd ipc ExporttoPC`
 3. Import the objects using the PowerCenter or with the following command:
`pmrep importObjects`
- Create and deploy mappings that domain users can run from the command line.
 - Profile data. If you use the Profiling option, profile data to reveal the content and structure of data. Profiling is a key step in any data project, as it can identify strengths and weaknesses in data and help you define a project plan.

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 [Version] > Client > Developer Client > 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**.

The first time you start the Developer tool, you must add a domain and a Model repository. If the domain includes more than one Data Integration Service, you must also select a default service.

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\9.5.1\clients\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\9.5.1\clients\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**.

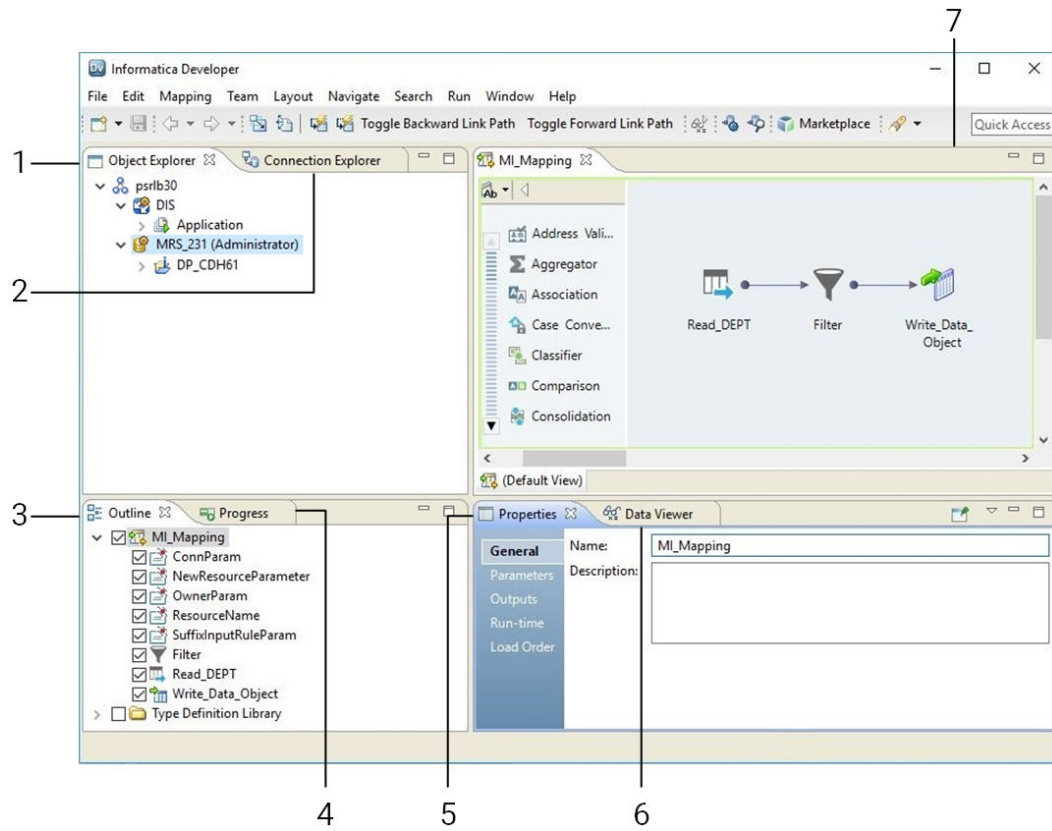
The first time you start the Developer tool, you must add a domain and a Model repository. If the domain contains more than one Data Integration Service, you must also select a default Data Integration Service.

Informatica Developer Interface

The Developer tool lets you design and implement data quality and data services solutions.

You can work on multiple tasks in the Developer tool at the same time. You can also work in multiple folders and projects at the same time. To work in the Developer tool, you access the Developer tool workbench.

The following figure shows the Developer tool workbench:



1. Object Explorer view
2. Connection Explorer view
3. Outline view
4. Progress view
5. Properties view
6. Data Viewer view
7. Editor

The recommended screen resolution for the Developer tool is 1920 * 1080 on Windows.

Informatica Developer Views

The Developer tool workbench includes an editor and views. You edit objects, such as mappings, in the editor. The Developer tool displays views based on which object is selected in the editor.

You can select additional views, hide views, and move views to another location in the Developer tool workbench.

To select the views you want to display, click **Window > Show View**.

Default Views

The Developer tool displays the following views by default:

Connection Explorer view

Displays connections to relational databases.

Data Viewer view

Displays source data, profile results, and previews the output of a transformation.

Object Explorer view

Displays the domain and the design-time and run-time objects in the domain. Design-time objects are stored in projects and folders in the Model repository. Run-time objects are stored as part of a run-time application on a Data Integration Service.

Outline view

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

Progress view

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

Properties view

Displays the properties for an object that is selected in the editor.

Additional Views

The Developer tool workbench also displays the following views:

Alerts view

Displays connection status alerts.

Audit Viewer view

Displays audit results for rules and conditions after a mapping runs with an audit.

Cheat Sheets view

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

Checked Out Objects view

Displays all objects that you have checked out.

Data Processor Events view

Displays information about initialization, execution, and summary events that occur when you run a Data Processor transformation in the Developer tool.

Data Processor Hex Source view

Displays an input document in hexadecimal format.

Help view

Displays context-sensitive online help.

Object Dependencies view

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

Search view

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

Tags view

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

Notifications view

Displays options to notify users or groups when all work in the Human task is complete.

Validation Log view

Displays object validation errors.

Version History view

Displays the version history of selected objects. You can read check-in comments and view user information about object check-ins.

Informatica Developer Welcome Page

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

The Welcome page displays the following options:

- Overview. Click the Overview button to get an overview of data quality and data services solutions.
- First Steps. Click the First Steps button to learn more about setting up the Developer tool and accessing Informatica Data Quality and Informatica Data Services lessons.
- Tutorials. Click the Tutorials button to see cheat sheets for the Developer tool and for data quality and data services solutions.
- Web Resources. Click the Web Resources button for a link to the Informatica Knowledge Base. You can access the Informatica How-To Library. The Informatica How-To Library contains articles about Informatica Data Quality, Informatica Data Services, and other Informatica products.
- Workbench. Click the Workbench button to start working in the Developer tool.

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 Developer Online Help

The Informatica Developer online help system contains information that can help you get the most from the Developer tool. Use the Contents, Search, and Index options to learn about the Developer tool features and capabilities. Add bookmarks to pages that you find useful. Use the Related Topics option to find pages that contain similar information.

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

Set up Informatica Developer to access Model repository objects. Select a Data Integration Service to preview data and run profiles, mappings, and workflows.

To set up the Developer tool, complete the following tasks:

1. Add a domain.
2. Add a Model repository.
3. If the domain includes more than one Data Integration Service, select a default service.
4. If the domain includes more than one Metadata Access Service, select a default service.

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

Step 1. Add a Domain

Add a domain in the Developer tool to access services that run on the domain.

Before you add a domain, verify that you have a domain name, host name, and port number to connect to a domain. You can get this information from an administrator.

1. Click **Window > Preferences**.
The **Preferences** dialog box appears.
2. Select **Informatica > Domains**.
3. Click **Add**.
The **New Domain** dialog box appears.
4. Enter the domain name, host name, and port number.
5. Click **Finish**.
6. Click **OK**.

Step 2. Add a Model Repository

Add a Model repository to access projects and folders.

Before you add a Model repository, verify the following prerequisites:

- An administrator has configured a Model Repository Service in the Administrator tool.
- You have a user name and password to access the Model Repository Service. You can get this information from an administrator.

1. Click **File > Connect to Repository**.
The **Connect to Repository** dialog box appears.
2. Click **Browse** to select a Model Repository Service.
3. Click **OK**.
4. Click **Next**.
5. Enter your user name and password.
6. Click **Next**.
The **Open Project** dialog box appears.
7. To filter the list of projects that appear in the **Object Explorer** view, clear the projects that you do not want to open.
8. Click **Finish**.
The Model Repository Service appears in the **Object Explorer** view and displays the projects that you chose to open. The **Object Explorer** view also shows the domain that contains the service.

Step 3. Select a Default Data Integration Service

The Data Integration Service performs data integration tasks in the Developer tool. If the domain includes more than one Data Integration Service, select a default service. You can override the default Data Integration Service when you run a mapping or preview data.

Note: If the domain contains only one Data Integration Service, this step is optional.

Add a domain before you select a Data Integration Service.

1. Click **Window > Preferences**.
The **Preferences** dialog box appears.
2. Select **Informatica > Data Integration Services**.
3. Expand the domain.
4. Select a Data Integration Service.
5. Click **Set as Default**.
6. Click **OK**.

Step 4. Select a Default Metadata Access Service

The Metadata Access Service allows the Developer tool to access Hadoop connection information to import and preview metadata . If the domain includes more than one Metadata Access Service, select a default service. You can override the default Metadata Access Service when you import a data object.

Note: If the domain contains only one Metadata Access Service, this step is optional.

Add a domain before you select a Metadata Access Service.

1. Click **Window > Preferences**.
The **Preferences** dialog box appears.
2. Select **Informatica > Metadata Access Services**.
3. Expand the domain.
4. Select a Metadata Access Service.
5. Click **Set as Default**.
6. Click **OK**.

Domains

The Informatica domain is a collection of nodes and services that define the Informatica environment.

You add a domain in the Developer tool. You can also edit the domain information or remove a domain. You manage domain information in the Developer tool preferences.

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 services and data quality 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.

Each project that you create also appears in the Analyst tool.

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

Task	Description
Manage projects	Manage project contents. You can create, duplicate, rename, and delete a project. You can view project contents.
Filter projects	Filter the list of projects that appear in the Object Explorer view.
Manage folders	Organize project contents in folders. You can create, duplicate, rename, and move folders within projects.
Manage objects	View object contents, duplicate, rename, move, and delete objects in a project or in a folder within a project.
Search projects	Search for folders or objects in projects. You can view search results and select an object from the results to view its contents.
Assign permissions	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.

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.
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.

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 maplet that contains a non-reusable Expression transformation. The maplet is the parent object. The Expression transformation is a dependent object of the maplet.

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:

Operation	Result
Open the parent object.	The Developer tool prompts you to determine how to open the parent object: <ul style="list-style-type: none">- 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.- 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.- Cancel. The Developer tool does not open the parent object.
Export the parent object to an XML file for use in the Developer tool.	The Developer tool creates the export file without the dependent object instances.
Export the parent object to PowerCenter.	You cannot export the parent object.
Validate the parent object.	The Developer tool validates the parent object as if the dependent objects were not part of the parent object.
Deploy the parent object.	You cannot deploy the parent object.
Copy and paste the parent object.	The Developer tool creates the new object without the dependent object instances.

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

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

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.

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. 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 Object Operations

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 save a copy of an object with a different name. You can also copy an object as a link to view the object in the Analyst tool or to provide a link to the object in another medium, such as an email message:

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

- Application
- Data service
- Logical data object model
- Mapping
- Maplet
- Physical data object
- Profile
- Reference table
- Reusable transformation
- Rule
- Scorecard
- Virtual stored procedure
- Workflow

Use the following guidelines when you copy objects:

- You can copy segments of mappings, maplets, rules, and virtual stored procedures.

- You can copy a folder to another project.
- You can copy a logical data object as a link.
- 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.

When you save a copy of a reference data object that specifies reference data values in a location outside the Model repository, the Model repository marks the object as not valid. To create a copy, select the object in the **Object Explorer** view, click **Edit > Copy**, and then paste the object to the location you want.

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 domain 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 also remove the association between a tag and an object. You can use the tag to search the domain for objects associated with the tag. 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.

When you deploy a mapping that is associated with a tag, the tag is propagated to the run-time version of the mapping on the Data Integration Service. If you update the deployed mapping using an application patch, the name of the patch is associated as a tag with the run-time version of the mapping.

Note: Tags that are associated with an object in the Developer tool appear as tags for the same objects in the Analyst tool.

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.

CHAPTER 2

The Model Repository

This chapter includes the following topics:

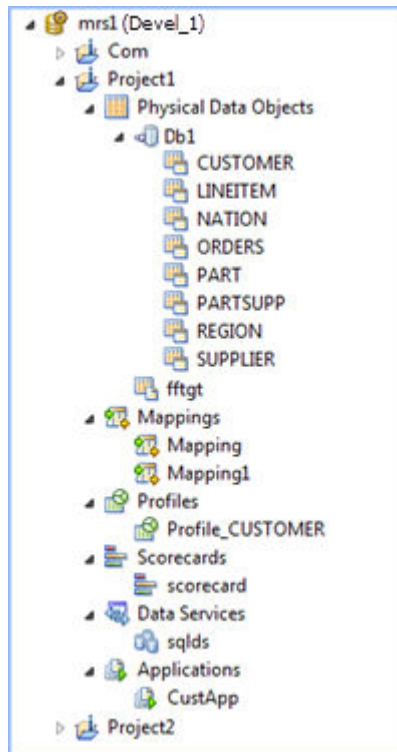
- [Model Repository Overview, 32](#)
- [Objects in Informatica Developer, 33](#)
- [Repository Object Locks, 36](#)
- [Team-based Development with Versioned Objects, 37](#)
- [Connecting to a Model Repository, 41](#)
- [Model Repository Service Refresh, 42](#)

Model Repository Overview

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

Connect to the Model repository to create and edit physical data objects, mapping, profiles, and other objects. Include objects in an application, and then deploy the application to make the objects available for access by end users and third-party tools.

The following image shows an open Model repository named mrs1 in the **Object Explorer** view:



The Model Repository Service manages the Model repository. All client applications and application services that access the Model repository connect through the Model Repository Service. Client applications include the Developer tool and the Analyst tool. Informatica services that access the Model repository include the Model Repository Service, the Analyst Service, and the Data Integration Service.

When you set up the Developer tool, you must add a Model repository. Each time you open the Developer tool, you connect to the Model repository to access projects and folders.

When you edit an object, the Model repository locks the object for your exclusive editing. You can also integrate the Model repository with a third-party version control system. With version control system integration, you can check objects out and in, undo the checkout of objects, and view and retrieve historical versions of objects.

Objects in Informatica Developer

You can create, manage, or view certain objects in the **Object Explorer** view 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, SQL data services, web services, and workflows. You can create, edit, and delete applications.

Data service

A collection of reusable operations that you can run to access and transform data. A data service provides a unified model of data that you can access through a web service or run an SQL query against. You can create, edit, and delete data services.

Data object operation

Repository object that contains properties required to perform certain run-time operations on sources or targets. Required for some PowerExchange adapter data sources.

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.

Operation mapping

A mapping that performs the web service operation for the web service client. An operation mapping can contain an Input transformation, an Output transformation, and multiple Fault transformations. You can create, edit, and delete operation mappings in a web service.

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.

Reference table

Contains the standard versions of a set of data values and any alternative version of the values that you may want to find. You can view and delete reference tables.

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.

Rule Specification

A reusable object that contains the logic of one or more business rules. An Analyst tool user builds a rule specification and saves the rule specification to the Model repository. You can select the rule specification in the Object Explorer view and drag the rule specification into a mapping.

A rule specification is a read-only object in the Developer tool. To view or edit the rule specification logic, right-click the rule specification and select the option to open the Analyst tool.

Note: An Analyst tool user can also generate a mapplet from a rule specification. The rule specification and the corresponding mapplet are independent objects in the Model repository. You can edit the mapplet in the Developer tool.

Scorecard

A graphical representation of valid values for a source column or output of a rule in profile results. You can create, edit, and delete scorecards.

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.

Type definition library

A repository object that stores complex data type definitions for a mapping that runs on the Spark engine. Complex data type definitions represent the schema of struct data. You can view the type definition library and the complex data type definitions for a mapping or a mapplet in the Outline view and in the mapping editor.

Virtual schema

A schema in a virtual database that defines the database structure. You can create, edit, and delete virtual schemas in an SQL data service.

Virtual stored procedure

A set of procedural or data flow instructions in an SQL data service. You can create, edit, and delete virtual stored procedures in a virtual schema.

Virtual table

A table in a virtual database. You can create, edit, and delete virtual tables in a virtual schema.

Virtual table mapping

A mapping that contains a virtual table as a target. You can create, edit, and delete virtual table mappings for a virtual table.

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**.

Repository Object Locks

When you begin to edit an object, the Model repository locks the object so that other users cannot save changes to it. When you save the object, you retain the lock. When you close the object, the repository releases the lock.

Object locking is part of continuous integration in a CI/CD pipeline. For more information about CI/CD, see [“CI/CD Overview” on page 194](#).

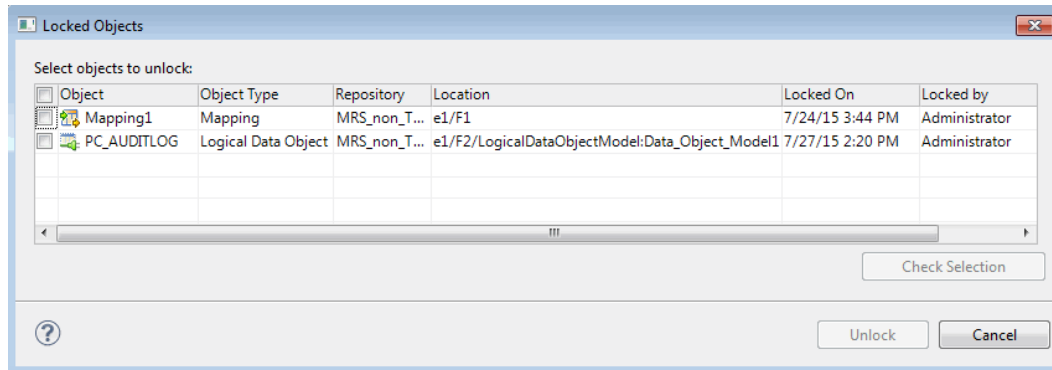
Lock Management

The Model repository retains object locks if the Developer tool stops unexpectedly. When you connect to the Model repository again, you can view the objects that you have locked. You can continue to edit the objects, or you can unlock the objects.

You might want to unlock objects if the developer who locked the object is unavailable and another developer is assigned to develop the object. You view and unlock locked objects through the **Locked Objects** dialog box.

To view the **Locked Objects** dialog box, click **File > Locked Objects**.

The following image shows the **Locked Objects** dialog box:



You can unlock one or more objects in the **Locked Objects** dialog box. Select objects to unlock by performing one of the following actions:

- Use the check box to select one or more objects.
- Select one or more rows and click **Check Selection** to enable the check box for each object that you selected.

Then you can click **Unlock** to unlock all the selected objects.

Tip: If your work is interrupted, you can view the objects that you have locked to identify the objects you were working on.

Rules and Guidelines for Lock Management

Consider the following rules and guidelines when you manage object locks:

- The Model repository does not lock the object when you open it. The Model repository locks the object only after you begin to edit it. For example, the Model repository locks a mapping when you insert a cursor in an editable field or connect mapping objects.

- You can use more than one client tool to develop an object. For example, you can edit an object on one machine and then open the object on another machine and continue to edit it. When you return to the first machine, you must close the object editor and reopen it to regain the lock. The same principle applies when a user with administrative privileges unlocks an object that you had open.
- If the Model Repository Service restarts while you have an object open for editing, you lose the lock on the object. Until you regain a lock on the object, another user can open and edit it. To regain a lock on the object, save changes to the object and close it, then reopen the object for editing.
- When you delete a folder that contains objects, and you are not allowed to delete any one of the objects, then you cannot delete the folder. For example, if you cannot delete an object because you do not own the lock on the object, the object and the folder remain.
- More than one developer can open and edit the contents of an SQL data service object at the same time. For example, UserA can open and begin to edit an SQL data service, and then UserB can open and begin to edit the same object. If UserB saves and closes the object before UserA, the Model repository does not tell UserA of the potential conflict until UserA saves the object. In this case, UserA can save changes by saving the SQL data service with another name.
- An administrator can revoke your write permission on an object you have locked, or reassign the lock to another user. In this case, you cannot edit or save the object. You can save the object with another name.

Team-based Development with Versioned Objects

Team-based development is the integration of the Model repository with a third-party version control system. The version control system saves multiple versions of objects and assigns each version a version number.

You manage object versions through the Developer tool. You can perform actions such as checking objects out and in, viewing and retrieving historical versions of objects, and undoing a checkout.

The Model repository protects objects from being overwritten by other members of the development team. If you open an object that another user has checked out, you receive a notification that identifies the user who checked it out. You can open a checked out object in read-only mode, or you can save it with a different name.

When the connection to the version control system is active, the Model repository has the latest version of each object.

The Model repository maintains the state of checked-out objects if it loses the connection to the version control system. While the connection with the version control system is down, you can continue to open, edit, save, and close objects. The Model repository tracks and maintains object states.

When the connection is restored, you can resume version control system-related actions, such as checking in or undoing the checkout of objects. If you opened and edited an object during the time when the connection was down, the Model repository checks out the object to you.

Note: SQL data service objects are not included in version control.

Object versioning is part of continuous integration in a CI/CD pipeline. For more information about CI/CD, see [“CI/CD Overview” on page 194](#).

Versioned Object Management

When the Model repository is integrated with a version control system, you can manage versions of objects. For example, you can check out and check in objects, undo checkouts, and view objects that you have checked out.

You can perform the following actions:

Check out an object.

When you check out an object, it maintains a checked-out state until you check it in or undo the checkout. You can view objects that you have checked out in the **Checked Out Objects** view. To check out an object, you can begin to edit it, or you can right-click the object in the Object Explorer and select Check Out.

Undo a checkout of an object.

When you undo a checkout, you check in the object without changes and without incrementing the version number or version history. All changes that you made to the object after you checked it out are lost. To undo a checkout, you can use the **Checked Out Objects** view or the object right-click menu. For example, you might want to undo a checkout to delete changes to an object.

Note: If a user moved an object while it was checked out and you undo the checkout, the object remains in its current location, and its version history restarts. Undoing the checkout does not restore it to its pre-checkout location.

Check in an object.

When you check in an object, the version control system updates the version history and increments the version number. You can add check-in comments up to a 4 KB limit. To check in an object, you can use the **Checked Out Objects** view or the object right-click menu.

Note: The Perforce and SVN version control systems limit file path lengths due to a Windows restriction that restricts path lengths to 260 bytes. If you attempt to check in a file with a path that exceeds the limitation, Perforce returns an error. A 260 byte path length results in varying path lengths depending on the character set of your system.

Delete an object.

A versioned object must be checked out before you can delete it. If it is not checked out when you perform the delete action, the Model repository checks the object out to you and marks it for delete. To complete the delete action, you must check in the object.

When you delete a versioned object, the version control system deletes all versions.

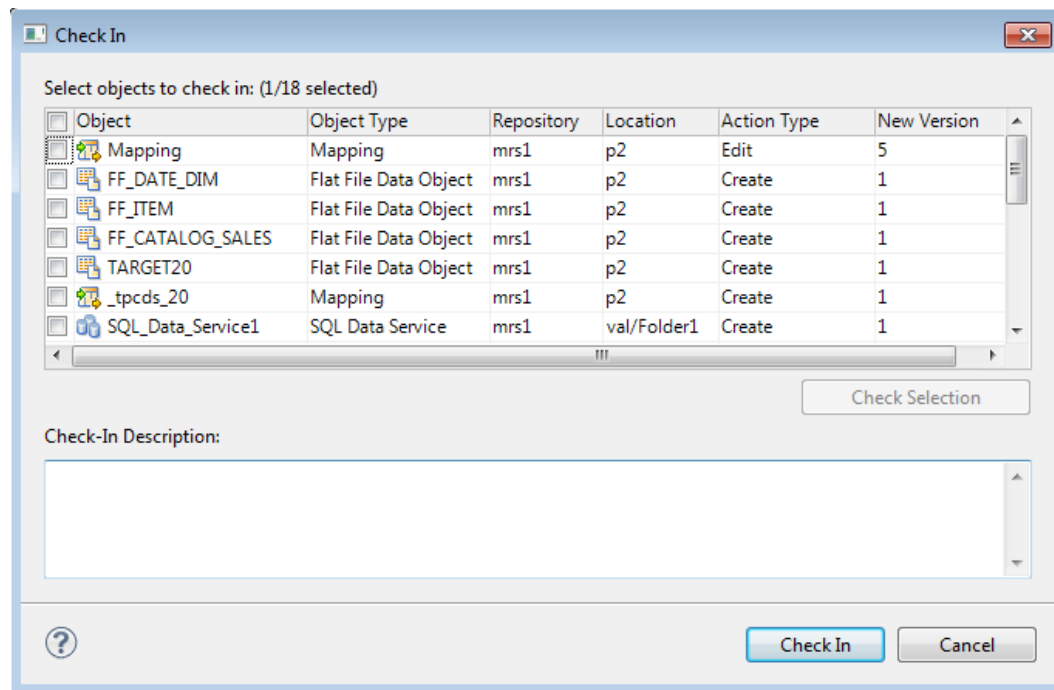
To delete an object, you can use the **Checked Out Objects** view or the object right-click menu.

Check In Dialog Box

The **Check In** dialog box displays all objects you have checked out.

Use the **Check In** dialog box to check in objects. You can also use it to delete objects that you marked for delete.

The following image shows the **Check In** dialog box:



The **Action Type** column shows the action that the Model Repository Service performs on each item when you check it in. The Model Repository Service performs one of the following action types:

Action Type	Description
Create	The object is new and has never been checked in. When you select it and click Check In , the Model Repository Service checks it in to the version control system and assigns it a version number of 1.
Delete	The object is marked for delete. When you select it and click Check In , the Model Repository Service deletes the object.
Edit	The object is marked for edit. When you select it and click Check In , the Model Repository Service replaces the version in the version control system with the new version.
Move	After you checked out the object, you moved it from one Model repository project or folder to another. When you select it and click Check In , the Model Repository Service checks it in to the version control system in the new location.

The **Location** column shows the current location of the object. If you moved the object, the **Action Type** is Move.

Tip: You can see previous locations of a moved object in the **Version History** view.

The **New Version** column shows the version number of the object after you complete the checkin.

You can check in one or more objects in the **Check In** dialog box. Select objects to check in by performing one of the following actions:

- Use the check box to select one or more objects.
- Select one or more rows and click **Check Selection** to enable the check box for each object that you selected.

Optionally, you can add a description in the **Check-In Description** text pane. Then you can click **Check In** to check in all the selected objects.

Deleting a Versioned Object

When you want to delete a versioned object, you mark it for deletion, and then check it in.

1. Right-click an object in the **Object Explorer** or the **Checked Out Objects** view, and select **Delete**.
The Action attribute changes from **Edit** to **Delete**.
2. Select the object and select **Check In**.
The Model repository deletes the object.

Historical Versions of Objects

The version control system saves a copy of an object every time you check it in.

The version control system saves a version of the object and assigns a version number to it. The highest number designates the latest version.

Use the **Version History** view to view the version history of objects, open historical versions in read-only view, and restore historical versions of objects. You might want to restore a historical version of an object to restore previous functionality.

When you restore a historical version of an object, that version becomes the latest version. The Model repository replaces the current version of the object with the historical version and checks the object out to you. If the object is open in an editor, the Developer tool refreshes the object to the restored version.

Restoring a Historical Version of an Object

You can view or get a historical version of an object from the version control system.

1. Right-click an object in the **Object Explorer** and select **View Version History**.
The **Revision History** view opens.
2. To view a historical version, right-click one of the versions listed in the **Revision History** view and select **View Version**.
The Developer tool opens a read-only version of the object. The Developer tool lists the version number in the title bar of the editor window.
3. To restore a historical version, right-click one of the versions listed in the **Revision History** view and select **Restore Version**.
The object opens in the editor. You can edit the object, or check it in without editing it.
4. Check in the object.
The restored version of the object becomes the latest version in the Model repository and the version control system.

Checked Out Objects View

The **Checked Out Objects** view lists all the objects that you have checked out.

You can perform the following actions in the **Checked Out Objects** view:

- Undo a checkout of an object
- Check in an object

To perform one of these actions, right-click the object and select the action.

To delete an object that is marked for delete in the **Action Type** column, right-click the object and select **Check In**.

Version History View

The **Version History** view displays the version history of selected objects. You can read check-in comments and view user information about object check-ins.

The view displays the following information:

- **Version.** Integer showing the order in which versions were saved. The version control system assigns the highest number to the most recent checked-in version.
- **Name.** Name of the object.
- **Location.** Path to the object in the Model repository.
- **Action type.** The action that the user took when checking in the object. You can add, edit, or delete and object when you check it in.
- **User.** User ID of the team member who checked in the object.
- **Version date.** Date and time that a user checked in the object.
- **Version description.** Check-in comments for the selected version.

To view version history, right-click an object in the **Object Explorer** and select **View Version History**.

Troubleshooting Team-based Development

Consider the following troubleshooting tips when you use features related to team-based development:

[The Perforce version control system fails to check in some objects, with an error about excessively long object path names.](#)

Due to Windows OS limitations on the number of characters in a file path, Model repository objects with long path and file names fail when you try to check them in. The Perforce error message reads "Submit aborted" and says the file path exceeds the internal length limit.

To work around this problem, limit the length of project, folder, and object names in the Model repository. Shorter names in all instances help limit the total number of characters in the object path name.

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. Select a namespace.

4. Click **OK**.

The Developer tool connects to the Model repository. The Developer tool displays the projects in the repository.

Model Repository Service Refresh

You can refresh the Model Repository Service to see new and updated objects in the Model repository.

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

CHAPTER 3

Searches in Informatica Developer

This chapter includes the following topics:

- [Searches in Informatica Developer Overview, 43](#)
- [Domain Search, 43](#)
- [Business Glossary Search, 45](#)
- [Editor Search, 46](#)

Searches in Informatica Developer Overview

You can perform searches in Informatica Developer to search for objects and look up business terms.

You can search the domains that appear in the **Object Explorer** view to find the latest versions of objects and object properties.

You can access the Business Glossary Desktop from the Developer tool to look up a Developer tool object name as a business term. You can also find objects, ports, groups, expressions, and attributes in an editor.

Domain Search

You can search for objects and object properties in a domain.

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 it finds more than 2048 objects. 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:

Search Option	Description
Containing text	Object or property that you want to search for. Enter an exact string or use a wildcard. Not case sensitive.
Names	One or more objects that contain the name. Enter an exact string or use a wildcard. Not case sensitive.
Tags	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.

A search engine indexes the metadata in the domain. 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 design-time and run-time objects.

You can search in different languages. To search in a different language, an administrator must change the search analyzer and configure the following Model repositories to use the search analyzer:

- Model repository that stores the design-time objects.
- Model repository that is associated with the Data Integration Service where the run-time objects are deployed.

Note: To search for an object or an object property, you must have connected to the object in the **Object Explorer** view. To expand the search to additional objects, connect to the Model repository or to the run-time application that contains the objects.

Guidelines for Searches

Refer to the following guidelines when you search a domain:

- You must save a design-time object before you can search on it.
- A search returns ports and dynamic ports but does not return generated ports.
- When you search for a mapping, mapplets with a name that is similar to the search pattern appear in the search results.
- When you refresh a Model Repository Service or a run-time application, the search results in Search view no longer appear. You must search the domain again to get results based on the refreshed Model Repository Service or the refreshed run-time application.

Searching for Objects and Properties

Search for objects and properties in a domain.

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 names 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**.
If the object is part of a run-time application, the object is read-only.

Business Glossary Search

Look up the meaning of a Developer tool object name as a business term in the Business Glossary Desktop to understand its business requirement and current implementation.

A business glossary is a set of terms that use business language to define concepts for business users. A business term provides the business definition and usage of a concept. The Business Glossary Desktop is a client that connects to the Metadata Manager Service, which hosts the business glossary. Use the Business Glossary Desktop to look up business terms in a business glossary.

If Business Glossary Desktop is installed on your machine, you can select an object in the Developer tool and use hotkeys or the Search menu to look up the name of the object in the business glossary. You can look up names of objects in Developer tool views, such as the **Object Explorer** view, or names of columns, profiles, and transformation ports in the editor.

For example, a developer wants to find a business term in a business glossary that corresponds to the Sales_Audit data object in the Developer tool. The developer wants to view the business term details to understand the business requirements and the current implementation of the Sales_Audit object in the Developer tool. This can help the developer understand what the data object means and what changes may need to be implemented on the object.

Business Glossary Desktop Lookup

The Business Glossary Desktop can look up object names in the business glossary and return business terms that match the object name.

The Business Glossary Desktop splits object names into two if the names are separated by a hyphen, underscore, or upper case.

For example, if a developer looks up a data object named Sales_Audit, the Business Glossary Desktop displays Sales_Audit in the search box but splits the name into Sales and Audit and looks up both business terms.

Looking Up a Business Term

Look up a Developer tool object name in the Business Glossary Desktop as a business term to understand its business requirement and current implementation.

You must have the Business Glossary Desktop installed on your machine.

1. Select an object.
2. Choose to use hotkeys or the Search menu to open the Business Glossary Desktop.
 - To use hotkeys, use the following hotkey combination:

CTRL+Shift+F

- To use the Search menu, click **Search > Business Glossary**.

The **Business Glossary Desktop** appears and displays business terms that match the object name.

Customizing Hotkeys to Look Up a Business Term

Customize hotkeys to change the combination of keys that open the Business Glossary Desktop.

1. From the Developer tool menu, click **Window > Preferences > General > Keys**.
2. To find or search **Search Business Glossary** in the list of commands, select one of the following choices:
 - To search for the keys, enter Search Business Glossary in the search box.
 - To scroll for the keys, scroll to find the **Search Business Glossary** command under the **Command** column.
3. Click the **Search Business Glossary Command**.
4. Click **Unbind Command**.
5. In the **Binding** field, enter a key combination.
6. Click **Apply** and then click **OK**.

Editor Search

You can find objects, ports, groups, expressions, and attributes that are open in the editor. The Developer tool highlights the objects within the open 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, enter a search string and the types of objects to find. The object types 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 data type. 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:

Editor	Object types
Mapping	Mapping objects, expressions, groups, ports, and complex data type definitions
Mapplet	Mapplet objects, expressions, groups, ports, and complex data type definitions
Logical data object model	Logical data objects and attributes
Physical data object read or write mapping	Mapping objects and columns
SQL data service	Virtual tables and attributes
Virtual stored procedure	Transformations, expressions, groups, and ports
Virtual table mapping	Virtual table mapping objects, expressions, groups, and ports

Editor	Object types
Web service operation mapping	Web service operation mapping objects, expressions, groups, and ports
Workflow	Workflow objects
<i>Note: The Model repository does not store the names of generated ports, and you cannot search for them in the editor.</i>	

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.

CHAPTER 4

Connections

This chapter includes the following topics:

- [Connections Overview, 48](#)
- [Connection Explorer View, 50](#)
- [Connection Management, 51](#)
- [Connection Switching, 53](#)
- [Third-Party JDBC Drivers, 57](#)
- [Environment SQL, 57](#)

Connections Overview

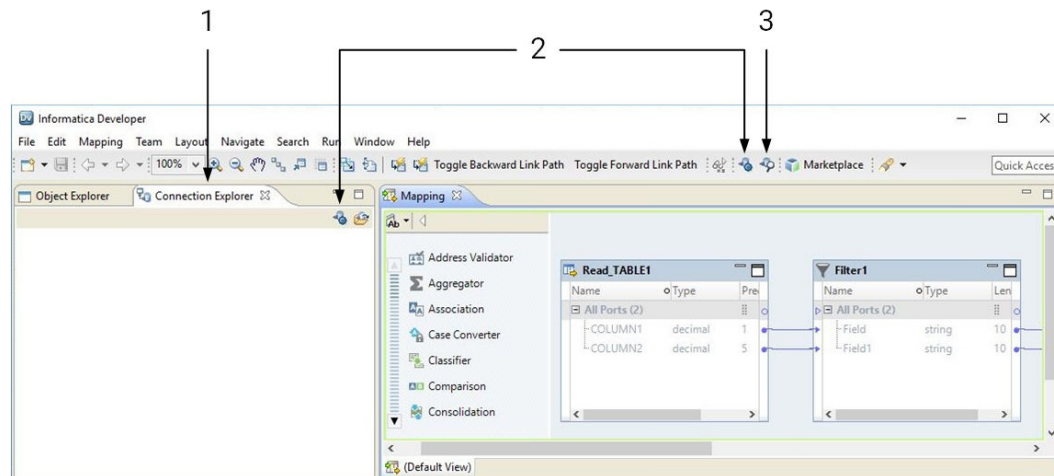
A connection is a repository object that defines a connection in the domain configuration repository.

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 a connection when you preview data, run mappings, or consume web services.

Note: The Developer tool does not use connections to import flat file data objects or preview, read, or write flat file data.

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

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



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

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.

Deployed applications and parameter files identify a connection by name, not by connection ID. Therefore, when you rename a connection, you must redeploy all applications that use the 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 or the Analyst tool.

Connection Types

Connections enable you to read from and write data to data sources.

Create a connection to access non-native environments, Hadoop and Databricks. If you access HBase, HDFS, or Hive sources or targets in the Hadoop environment, you must also create those connections. You can create the connections using the Developer tool, Administrator tool, and infacmd.

You can create the following kinds of connections:

Hadoop connection

Create a Hadoop connection to run mappings in the Hadoop environment.

HBase connection

Create an HBase connection to access HBase. The HBase connection is a NoSQL connection.

HDFS connection

Create an HDFS connection to read data from or write data to the HDFS file system on a Hadoop cluster.

Hive connection

Create a Hive connection to access Hive as a source or target. You can access Hive as a source if the mapping is enabled for the native or Hadoop environment. You can access Hive as a target if the mapping runs on the Blaze engine.

JDBC connection

Create a JDBC connection and configure Sqoop properties in the connection to import and export relational data through Sqoop.

Databricks connection

Create a Databricks connection to run mappings in the Databricks environment.

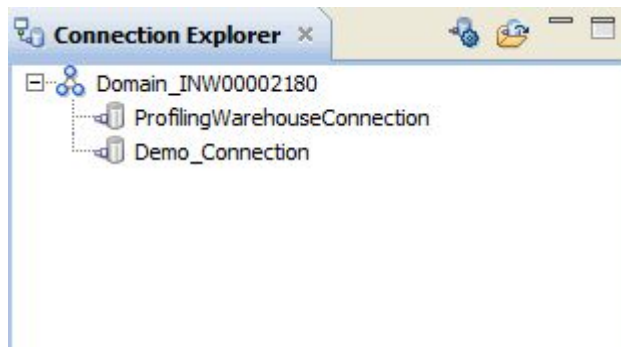
Note: For information about creating connections to other sources or targets such as social media web sites or Teradata, see the respective PowerExchange adapter user guide for information.

For information about supported versions of any data source, see the [Informatica Product Availability Matrix](#).

Connection Explorer View

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

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



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

- 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 database. Right-click the database and click **Connect**.
- Disconnect from a relational 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.
- 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 a Connection

Create a connection to access a compute cluster, database, enterprise application, file system, nonrelational database, NoSQL database, social media application, or web service. Create the connection before you import physical data objects, preview data, profile data, or run mappings.

1. Click **Window > Preferences**.
2. Select the type of connection that you want to create:
 - 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 a connection type in the **Available Connections** list, and click **Add**.
The **New <Connection Type> Connection** dialog box appears.
5. Enter the following information:

Property	Description
Name	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: ~ ` ! \$ % ^ & * () - + = { [] \ : ; " ' < , > . ? /
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	Optional description for the connection.

Property	Description
Location	Domain in which the connection exists.
Type	Specific connection type, such as Oracle, Twitter, or Web Services.

6. Click **Next**.
7. Configure the connection properties.
8. Click **Test Connection** to verify that you entered the connection properties correctly and that you can connect to the database, application, file system, or URI.

Note: **Test Connection** is not supported with Databricks.

9. Click **Finish**.

After you create a connection, you can add it to the **Connection Explorer** view.

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.

Note: Before you copy a Hadoop, HDFS, HBase, Hive, or Databricks connection to another domain, create a cluster configuration in the domain with a name that matches the cluster configuration associated with the connection. For example, if the connection is associated with a cluster configuration named XYZ, create a cluster configuration named XYZ in the target 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.

Connection Switching

You can switch the connection of a relational data object or customized data object to use a different relational database connection. You can also simultaneously switch the connections for multiple data

objects. Switching the connection saves time and effort because you do not have to update each mapping to use the new connection.

After you switch the connection, the Developer tool updates the connection details for the data object in all Read, Write, and Lookup transformations that are based on the data object. The Developer tool also updates the database type for the data object based on the database that the new connection points to.

You can switch a connection to one of the following connection types:

- IBM DB2
- Microsoft SQL Server
- ODBC
- Oracle

When the Developer tool switches a connection, it does not validate the metadata compatibility. Therefore, before you switch the connection, you must ensure that the database that the new connection points to contains a table with the same columns and metadata as the data object for which you are switching the connection. Otherwise, data loss or inconsistencies can occur.

Example

You have created an Oracle relational data object in the Developer tool and added it as a Write transformation in multiple mappings.

You migrate the Oracle database to an IBM DB2 database. You want to update the existing mappings in the Developer tool to write data to the IBM DB2 database.

Instead of replacing the data object that the Write transformation is based on in each mapping, you can switch the connection of the Oracle data object to an IBM DB2 connection. The new connection points to the IBM DB2 database to which you want to write data. When you run mappings that contain the Oracle relational data object, the Data Integration Service uses the new IBM DB2 connection to run the mappings.

Before You Switch a Connection

Before you switch a connection, verify that the following requirements are met:

- You have write permissions on the data object for which you want to switch the connection.
- The database that the new connection points to contains a table with the same columns and metadata as the data object for which you want to switch the connection. If the precision and scale of columns in the table are lower than that of the data object, data loss or inconsistencies can occur.
- The data object for which you want to switch the connection does not contain unsaved changes. If it contains unsaved changes, the Developer tool does not switch the connection.
- The customized data object for which you want to switch the connection does not contain relational data objects of different database types. If it does, the Developer tool does not display an error when you switch the connection. However, when you run a mapping with the customized data object, the mapping fails.
- The table names that the original connection and new connection point to match exactly, including the case. If the connection supports quoted identifiers and mixed-case table names, the Developer tool treats the table names as case sensitive.
- The data object for which you want to switch the connection is checked out if the object is stored in a versioned repository.

Switching a Connection

You can switch the connection for a relational data object or customized data object to use a different relational database connection and simultaneously update the existing mappings to use the new connection.

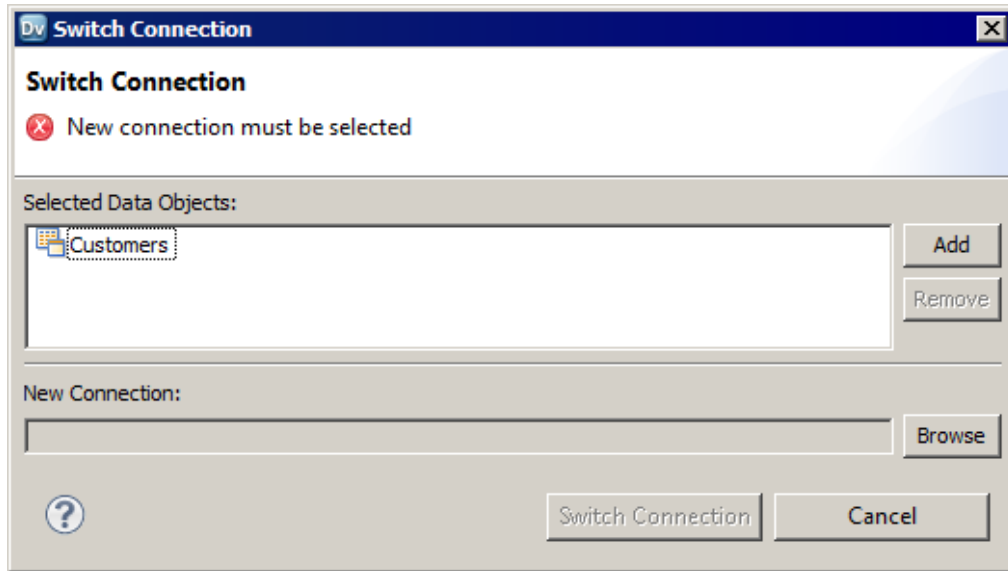
1. In the **Object Explorer** view, right-click the data object for which you want to switch the connection.

You can select multiple data objects under different folders of a project or across projects.

You can also select a connection and simultaneously switch the connection for all relational data objects and customized data objects that use the connection.

2. Click **Switch Connection**.

The **Switch Connection** dialog box appears and displays the selected data objects.



3. Click **Add** to update the list of data objects for which you want to switch the connection.

To remove an object, select the object and click **Remove**.

4. Click **Browse** next to the **New Connection** field.

The Developer tool displays the connections that you can use for the data object.

5. Select the new connection that you want to use for the data object and click **OK**.

6. Click **Switch Connection**.

A message appears prompting you to ensure that the database that the new connection points to contains a table with the same columns and metadata as the data object for which you are switching the connection.

7. Click **OK** to switch the connection for the data object.

A message appears stating that the connection was switched successfully. The Developer tool updates the connection details and the database type for the data object in all Developer tool objects associated with the data object. When you run mappings that contain the data object, the Data Integration Service uses the new connection.

After You Switch a Connection

After you switch a connection, verify the data object properties and manually edit the properties, if required.

Perform the following tasks after you switch a connection:

- Verify the data type mapping.
- Verify the table owner name.
- Verify Lookup transformations.
- Reconfigure hints.
- Synchronize data objects.

Verify the Data Type Mapping

When you switch a connection, the Developer tool identifies the best data type match between the databases that the original connection and the new connection point to. It sets the data types for the data object and the transformations in the mapping that are based on the data object accordingly. Before you run a mapping, verify the data types and manually update them, if required.

For example, you switch a connection from Oracle to Microsoft SQL Server. For data types with a fixed precision, by default, the Developer tool sets the precision based on the precision of the Microsoft SQL Server database. However, for data types with a variable precision, the Developer tool sets the precision and scale based on the Oracle database. For data types such as Timestamp with Time Zone, the Developer tool sets the data type as Varchar (0,0) or an equivalent data type that the Microsoft SQL Server database supports for Varchar.

Verify the Table Owner Name

When you switch a connection, the Developer tool retains the table owner name of the data object. You can manually edit the table owner name in the data object properties, if required.

If you set the table owner name to blank and switch the connection, you can successfully preview the data only if the table exists in the default schema or public schema of the database that the new connection points to. Otherwise, the data preview fails. You must manually update the table owner name in the run-time properties of the data object to successfully preview the data.

Verify Lookup Transformations

After you switch a connection, rebuild the lookup cache and verify the lookup conditions for Lookup transformations based on the data object, if required.

Perform the following tasks:

Rebuild the lookup cache

If you configure the Data Integration Service to persist the lookup cache and switch the connection for the associated data object, you must update the Lookup transformation to rebuild the lookup cache based on the new connection. Otherwise, when you run the mapping, an error occurs stating that the cache file was created with a different database connection.

Verify the lookup conditions

When you switch a connection, the Developer tool identifies the best data type match between the databases that the original connection and the new connection point to, and sets the data types accordingly. After you switch a connection, the lookup conditions might not be valid because of the data type change. You must verify the lookup conditions and manually update them.

For example, you create a mapping that contains an IBM DB2 source table, lookup table, and target table. You configure the lookup condition on an Integer column of the source table and lookup table. If you switch the connection of the lookup table from IBM DB2 to Oracle, the transformation data type of the

Integer column in the lookup table changes to Decimal. The lookup condition is not valid because you cannot compare an Integer column with a Decimal column.

Reconfigure Hints

When you switch the connection of a customized data object, the Developer tool does not retain the hints that you had configured for the customized data object. You must manually reconfigure the hints.

Synchronize Data Objects

After you switch the connection, the Developer tool retains only the active reference key constraints.

When you switch the connection for multiple data objects simultaneously, you must synchronize the data objects to ensure that the key relationships are accurate.

If the data objects contain multiple tables with cyclic reference key constraints between them and you switch the connection for a subset of tables to a different database type, the Developer tool switches the connection without displaying any error. However, when you view the key relationships for the data objects, the Developer tool displays the key relationships with reference to the original database. To update the key relationships and point them to the new database, you must synchronize the data objects.

Third-Party JDBC Drivers

If you want to connect to sources, targets, and lookups 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:

```
<Informatica installation directory>\clients\externaljdbcjars
```

To run data previews, profiles, or mappings, copy the JDBC driver jar file to the following location:

```
<Informatica installation directory>\externaljdbcjars
```

Environment SQL

The Integration Service runs environment SQL in auto-commit mode and closes the transaction after it issues the SQL. Use SQL commands that do not depend on a transaction being open during the entire read or write process. For example, if a source database is set to read only mode and you create an environment SQL statement in the source connection to set the transaction to read only, the Integration Service issues a commit after it runs the SQL and cannot read the source in read only mode.

You can configure connection environment SQL or transaction environment SQL.

Use environment SQL for source, target, lookup, and stored procedure connections. If the SQL syntax is not valid, the Integration Service does not connect to the database, and the session fails.

Note: When a connection object has “environment SQL,” the connection uses “connection environment SQL.”

Connection Environment SQL

This custom SQL string sets up the environment for subsequent transactions. The Integration Service runs the connection environment SQL each time it connects to the database. If you configure connection environment SQL in a target connection, and you configure three partitions for the pipeline, the Integration Service runs the SQL three times, once for each connection to the target database. Use SQL commands that do not depend on a transaction being open during the entire read or write process.

For example, use the following SQL statement to set the quoted identifier parameter for the duration of the connection:

```
SET QUOTED_IDENTIFIER ON
```

Use the SQL statement in the following situations:

- You want to set up the connection environment so that double quotation marks are object identifiers.
- You configure the target load type to Normal and the Microsoft SQL Server target name includes spaces.

Transaction Environment SQL

This custom SQL string also sets up the environment, but the Integration Service runs the transaction environment SQL at the beginning of each transaction.

Use SQL commands that depend on a transaction being open during the entire read or write process. For example, you might use the following statement as transaction environment SQL to modify how the session handles characters:

```
ALTER SESSION SET NLS_LENGTH_SEMANTICS=CHAR
```

This command must be run before each transaction. The command is not appropriate for connection environment SQL because setting the parameter once for each connection is not sufficient.

Guidelines for Configuring Environment SQL

Consider the following guidelines when creating the SQL statements:

- You can enter any SQL command that is valid in the database associated with the connection object. The Integration Service does not allow nested comments, even though the database might.
- When you enter SQL in the SQL Editor, you type the SQL statements.
- Use a semicolon (;) to separate multiple statements.
- The Integration Service ignores semicolons within /*...*/.
- If you need to use a semicolon outside of comments, you can escape it with a backslash (\).
- You can use parameters and variables in the environment SQL. Use any parameter or variable type that you can define in the parameter file. You can enter a parameter or variable within the SQL statement, or you can use a parameter or variable as the environment SQL. For example, you can use a session parameter, \$ParamMyEnvSQL, as the connection or transaction environment SQL, and set \$ParamMyEnvSQL to the SQL statement in a parameter file.
- You can configure the table owner name using sqlid in the connection environment SQL for a DB2 connection. However, the table owner name in the target instance overrides the SET sqlid statement in environment SQL. To use the table owner name specified in the SET sqlid statement, do not enter a name in the target name prefix.

CHAPTER 5

Physical Data Objects

This chapter includes the following topics:

- [Physical Data Objects Overview, 59](#)
- [Physical Data Object Types, 60](#)
- [Relational Data Objects, 60](#)
- [Customized Data Objects, 64](#)
- [Create or Replace Target Tables, 69](#)
- [Custom Queries, 72](#)
- [Nonrelational Data Objects, 85](#)
- [WSDL Data Object, 86](#)
- [Synchronization, 90](#)
- [Troubleshooting Physical Data Objects, 93](#)

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 data sources. 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, maplet, or profile, but you cannot change the data object within the mapping, maplet, 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, maplet, or profile. You can add a physical data object to a mapping or maplet 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.

You can also include a physical data object in a virtual table mapping when you define an SQL data service. You can include a physical data object in an operation mapping when you define a web service.

Physical Data Object Types

You can create different types of physical data objects based on the type of data source that you want to read data from or write data to.

Physical data objects include 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.

Nonrelational data object

A physical data object that uses a nonrelational database resource as a source. For example, you can create a nonrelational data object from a VSAM source.

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.

Informatica PowerExchange® adapters also provide access to many data sources that you can use to create physical data objects, such as SAP, Salesforce, and Netezza.

Relational Data Objects

A relational data object is a physical data object that has a relational table, view, or synonym as a source. You can include a relational data object in a mapping, mapplet, or profile.

The relational data object describes a resource in a database. When you add a relational data object to the repository, you import it from a source database. You can change the relational data object definition after you import it to the repository. You can add and remove ports, define primary keys, and configure relationships between multiple relational data objects in the repository. You can change or parameterize the default connection, the database owner, and the resource name. You can also select or clear all nullable values of the columns using Enable All Nullable or Disable All Nullable icons.

When you add the relational data object to a mapping, mapplet, or profile, you indicate whether you want to create a read or a write object. You can configure different run-time properties based on whether the object is a source, a target, or a lookup table.

The following image shows a sample relational data object in the editor:

Overview

General

Name: ReportDefinition_RelationalDataObject

Description:

ReportDefinition Re...

Name	Native T
DEALERSHIP_ID	number
REGION	varchar2
CUSTOMER_NO	number
FIRSTNAME	varchar2

Columns

Search

	Name	Native Type	Precision	Scale	Primary Key	Nullable	Description
1	DEALERSHIP_ID	number	15	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
2	REGION	varchar2	10	0	<input type="checkbox"/>	<input type="checkbox"/>	
3	CUSTOMER_NO	number(p,s)	5	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
4	FIRSTNAME	varchar2	20	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
5	LASTNAME	varchar2	20	0	<input type="checkbox"/>	<input type="checkbox"/>	

Overview | Keys | Relationships | Advanced

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

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 with multiple relational resources. 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:

- DB2 for i5/OS
- DB2 for z/OS
- HAWQ
- IBM DB2
- JDBC
- Microsoft SQL Server
- Netezza
- ODBC
- Oracle
- SAP HANA

Importing a Relational Data Object

Import a relational data object to add to a mapping, maplet, 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**.

You can use a question mark (?) as a single-character wild card, or an asterisk (*) as a multiple-character wild card in a search query.

Note: The search displays a maximum of 20K results.

9. If an physical data object with the same name as the resource you are importing already exists, the **Import Conflict Resolution** dialog box appears. Choose how to resolve the import conflict. You can create a physical data object with a different name, reuse the existing object, or replace the existing object.

Select **Apply to all conflicts** to apply the same solution to all import conflicts.

10. Enter a name for the physical data object.

11. Click **Browse** next to the **Location** option and select the project where you want to import the relational data object.

12. Click **Finish**.

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

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 maplet.

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.

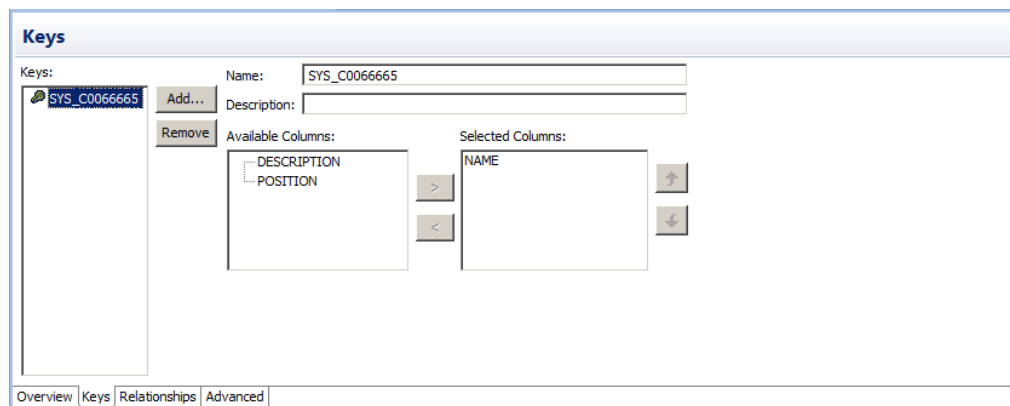
The key relationships that you create exist in the relational data object metadata. You do not need to alter the source relational resources.

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.

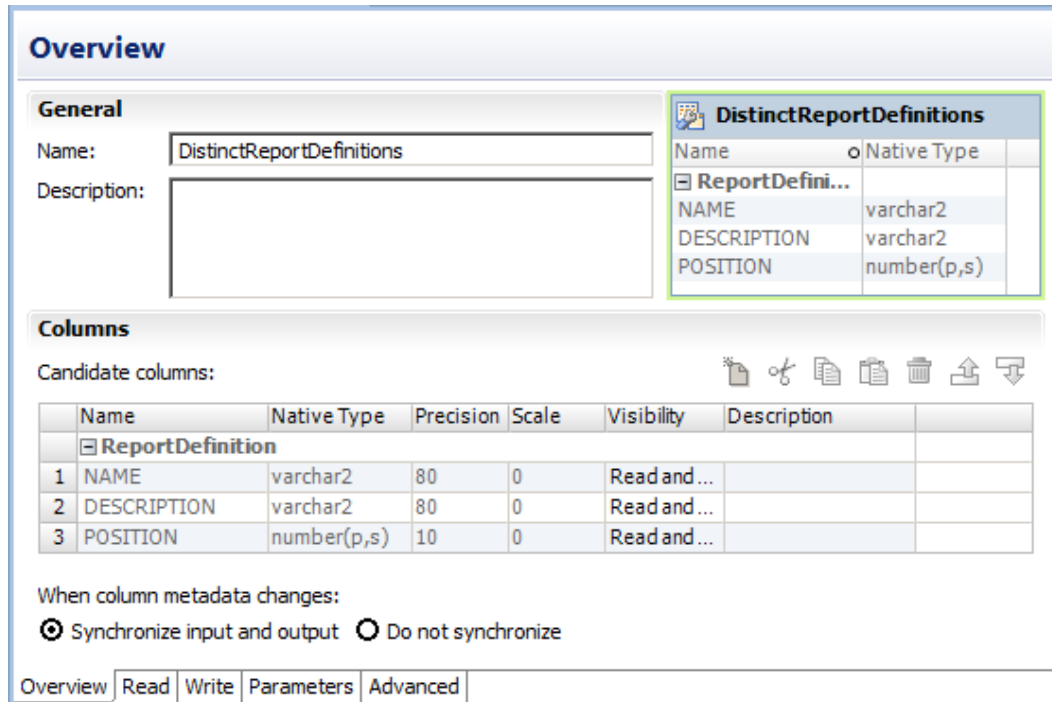
1. Open the relational data object where you want to create a foreign key.
2. Select the **Relationships** view.
3. Click **Add**.
The **New Relationship** dialog box appears.
4. Enter a name for the foreign key.
5. Select a primary key from the referenced relational data object.
6. Click **OK**.
7.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

Customized Data Objects

Customized data objects are 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. You can define parameters for the connection name, table owner, and table name. 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:

- DB2 i5/OS connections
- DB2 z/OS connections
- IBM DB2 connections
- JDBC connections
- Microsoft SQL Server connections
- ODBC connections
- Oracle connections
- Relational data objects

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:

Property	Description
Truncate Hive Target Partition	Overwrites the partition in the Hive target in which the data is being inserted. To enable this option, you must also select the option to truncate target tables. You can truncate external partitioned tables and non-partitioned tables. Default is disabled.
Load type	Type of target loading. Select Normal or Bulk. 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. When you write to an Oracle target with bulk loading, you can optimize performance by disabling constraints in the Oracle database. 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: <code>SET QUOTED_IDENTIFIER ON</code>
Update override	Overrides the default UPDATE statement for the target.
Delete	Deletes all rows flagged for delete. Default is enabled.
Insert	Inserts all rows flagged for insert. Default is enabled.
Target Schema Strategy	Type of target schema strategy for the relational or Hive target table. You can select one of the following target schema strategies: <ul style="list-style-type: none"> - RETAIN - Retain existing target schema. The Data Integration Service retains the existing target schema. - CREATE - Create or replace table at run time. The Data Integration Service drops the target table at run time and replaces it with a table based on a target table that you identify. - Assign Parameter. You can assign a parameter to represent the value for the target schema strategy and then change the parameter at run time.
DDL query for create or replace	The DDL query based on which the Data Integration Service creates or replaces the target table. This option is applicable when you select the CREATE - Create or replace table at run time target schema strategy.
Truncate target table	Truncates the target before it loads data. Default is disabled.

Property	Description
Update strategy	Update strategy for existing rows. You can select one of the following strategies: <ul style="list-style-type: none"> - Update as update. The Data Integration Service updates all rows flagged for update. - Update as insert. The Data Integration Service inserts all rows flagged for update. You must also select the Insert target option. - 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.
PreSQL	SQL command the Data Integration Service runs against the target database before it reads the source. The Developer tool does not validate the SQL.
PostSQL	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.

Creating a Customized Data Object

Create a customized data object to add to a mapping, maplet, 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.

Create or Replace Target Tables

In the Developer tool, you can generate a DDL script for one or more relational data objects in the Model repository, and run the DDL script to create or replace tables in the target database. If a target already exists in that database, you can drop the target and re-create it.

You can create or replace the target table at design-time or at run time. To create or replace tables at design-time, you must create and execute the DDL script before you run a mapping. You can configure a target schema strategy in the Write transformation in a mapping to create or replace a relational target at run time.

The Developer tool generates a database-specific version of the DDL script for all supported connection types. If you select a JDBC or ODBC target, the Developer tool generates the ANSI SQL-92 generic data type format. You can generate DDL scripts for the following database types:

- IBM DB2
- Greenplum
- Hive
- JDBC
- Microsoft SQL Server
- Netezza
- ODBC
- Oracle
- Teradata

Note: You cannot generate a DDL script if you use an OLE DB SQL Server connection.

Rules and Guidelines to Create or Replace Target Tables

Consider the following rules and guidelines when you generate and execute a DDL script:

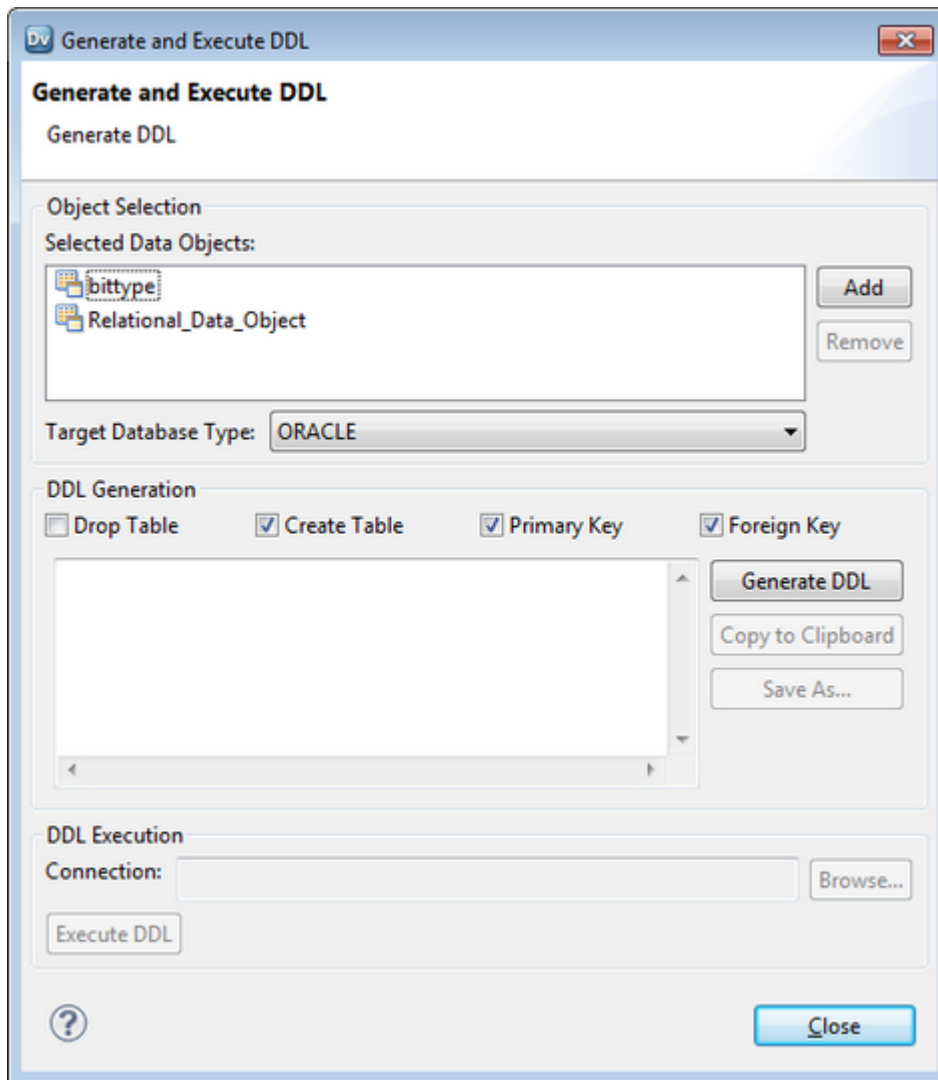
- Avoid selecting multiple objects with the same source table name. When you select multiple objects with the same source table name, the DDL code fails. If you select the Drop Table and the Create Table options for three objects with the same source table name, the DDL code succeeds for the first Drop Table and Create Table commands, and fails for the subsequent commands.
- Before you run a mapping, verify the data types and manually update them, if required. The char and byte semantics in Oracle are ignored when you generate the DDL script. When you create a table that contains char and varchar2 columns in an Oracle database, you can define the storage for char and byte semantics. When you import the metadata for the Oracle table into the Developer tool, the char and byte semantics are ignored. If you generate the DDL script for the Oracle table, the Developer tool defines the data type as Char.
- If you generate DDL for ODBC connections, the Developer tool creates the DDL script in ANSI SQL-92 generic data type format. The ANSI SQL-92 format might not run on all databases as the target database might not support the data type or data length.
- If you create a DDL script to migrate database tables from Greenplum to Netezza, you can incorporate only 16000 characters in the NVARCHAR column because the NVARCHAR data type supports only up to 16000 characters in a Netezza database.
- When you generate the DDL script, the Developer tool identifies the best data type match between the databases that the original connection and the new connection point to. The precision and scale for data types vary between databases. In an Oracle database, the default precision and scale for the Timestamp data type is (29, 9). When you generate the DDL script from Oracle to Microsoft SQL Server, the precision and scale for the Timestamp data type reduce to (26, 6). When you generate the DDL script from Oracle to DB2, the precision and scale for the Timestamp data type reduce to (27, 7).

Generating and Executing DDL at Design-Time

Before you generate and run the DDL script, ensure that the user has the appropriate read and write permissions to access the target database.

1. In the **Object Explorer** view of the Developer tool, select the relational data object for which you want to create a table in the target database. If you want to create multiple tables, hold down the **Shift** and **Ctrl** keys to select multiple data objects.
2. Right-click a selected data object and select **Generate and Execute DDL**.

The Generate and Execute DDL dialog box appears.



3. In the Object Selection area, select the **Target Database Type** for which you want to generate DDL.
4. In the DDL Generation area, you can select the following options:
 - **Drop Table**. Drop a table in the database before you create it. Select this option to replace a table.
 - **Create Table**. Creates a table on the target database.
 - **Primary Key**. Creates primary keys based on the selected data objects.
 - **Foreign Key**. Creates foreign keys based on the selected data objects.
 - **Copy to Clipboard**. Copies the DDL script to the clipboard.
 - **Save As**. Saves the DDL script to a file.
5. In the DDL Generation area, click **Generate DDL**.
The DDL generated for the selected data objects appears in the DDL Generation area.
Warning: If you edit the DDL script generated in the DDL Generation area, you might encounter errors when you run the DDL script.
6. In the DDL Execution area, click **Browse** and choose a target database connection.

The **Choose Connection** dialog box lists the target database connections based on the target database type that you choose in the Object Selection area. For example, if you choose Oracle as the target database type, the Developer tool displays Oracle connections.

7. Click **Execute DDL**.
8. Click **Close**.

Generating and Executing DDL at Run Time

When you select a target schema strategy in a Write transformation in a mapping to create or replace a relational or Hive target at run time, you can define a DDL query.

The Data Integration Service creates or replaces relational and Hive target tables based on the DDL query that you define.

You can define a DDL query to customize the table or you can specify additional parameters such as partitions, based on which the Data Integration Service must create or replace the target table. The table contains the columns that you define in the DDL query. You can also enter placeholders in the DDL query. The Data Integration Service substitutes the placeholders with the actual values at run time.

For more information, see the *Informatica Developer Mapping Guide*.

DDL Generation Errors

When you generate and run a DDL script, you might encounter an error. The error can occur due to one of the following reasons:

- If the target database does not support the data type that you selected.
- If you select a physical data object that contains a cyclic dependency.
- If you select a physical data object that has no columns.

Custom Queries

A custom SQL query is a SELECT statement that overrides the default SQL query in a customized 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. 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:

- 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 source 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:

```
[Oracle]
OPTION
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:

Name	Value
Reserved Words File	<path>\reswords.txt

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.

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

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.

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.

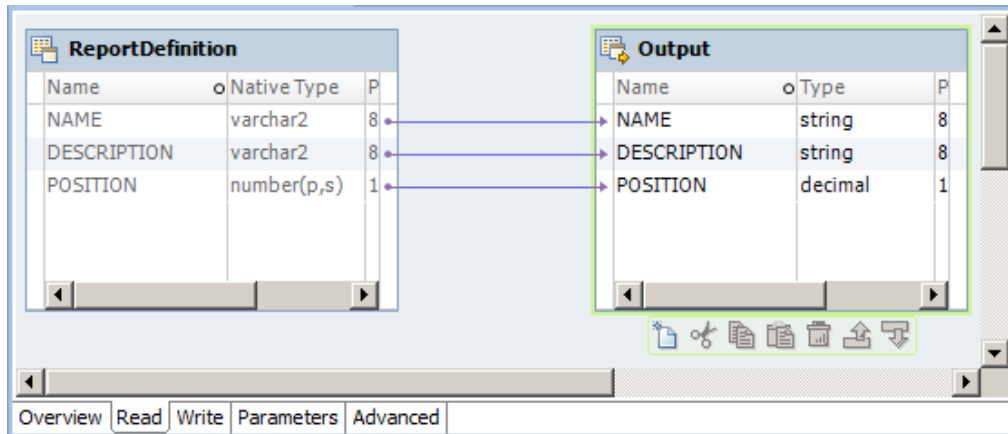
Note: You can also use the Sorter transformation to sort relational and flat file data before Aggregator and Joiner transformations.

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.

The following figure shows the **Read** view of a customized data object that is open in the editor:



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:

Syntax	Description
<i>source1</i>	Source resource name. The Data Integration Service returns rows from this resource that match the join condition.
<i>source2</i>	Source resource name. The Data Integration Service returns rows from this resource that match the join condition.
<i>join_condition</i>	Condition for the join. Use syntax supported by the source database. You can combine multiple join conditions with the AND operator.

For example, you have a REG_CUSTOMER table with data for registered customers:

CUST_ID	FIRST_NAME	LAST_NAME
00001	Marvin	Chi
00002	Dinah	Jones
00003	John	Bowden
00004	J.	Marks

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

TRANSACTION_NO	CUST_ID	DATE	AMOUNT
06-2000-0001	00002	6/3/2000	55.79
06-2000-0002	00002	6/10/2000	104.45
06-2000-0003	00001	6/10/2000	255.56
06-2000-0004	00004	6/15/2000	534.95
06-2000-0005	00002	6/21/2000	98.65
06-2000-0006	NULL	6/23/2000	155.65
06-2000-0007	NULL	6/24/2000	325.45

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

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

The Data Integration Service returns the following data:

CUST_ID	DATE	AMOUNT	FIRST_NAME	LAST_NAME
00002	6/3/2000	55.79	Dinah	Jones
00002	6/10/2000	104.45	Dinah	Jones
00001	6/10/2000	255.56	Marvin	Chi
00004	6/15/2000	534.95	J.	Marks
00002	6/21/2000	98.65	Dinah	Jones

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.

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:

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

For example, using the same REG_CUSTOMER and PURCHASES tables described in [“Normal Join Syntax” on page 80](#), 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:

CUST_ID	FIRST_NAME	LAST_NAME	DATE	AMOUNT
00001	Marvin	Chi	6/10/2000	255.56
00002	Dinah	Jones	6/3/2000	55.79
00003	John	Bowden	NULL	NULL

CUST_ID	FIRST_NAME	LAST_NAME	DATE	AMOUNT
00004	J.	Marks	6/15/2000	534.95
00002	Dinah	Jones	6/10/2000	104.45
00002	Dinah	Jones	6/21/2000	98.65

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:

CUST_ID	FIRST_NAME	LAST_NAME	DATE	AMOUNT
00001	Marvin	Chi	6/10/2000	255.56
00002	Dinah	Jones	6/10/2000	104.45
00003	John	Bowden	NULL	NULL
00004	J.	Marks	6/15/2000	534.95

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:

CUST_ID	CUST_ID	RETURN
00002	6/10/2000	55.79
00002	6/21/2000	104.45

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

```
{ 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:

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

CUST_ID	FIRST_NAME	LAST_NAME	DATE	AMOUNT	RET_DATE	RETURN
00002	Dinah	Jones	NULL	NULL	6/10/2000	55.79
00002	Dinah	Jones	NULL	NULL	6/21/2000	104.45

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:

```
{ source1 RIGHT OUTER JOIN source2 on join_condition }
```

The following table displays syntax for a right outer join in a join override:

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

PreSQL and PostSQL Commands

You can create SQL commands in a customized data object or relational data object instance. You can run the SQL commands to execute SQL statements such as insert, update, and delete. The Data Integration Service runs the SQL commands against the source relational resource.

When you run the mapping, the Data Integration Service runs PreSQL commands against the source database before it reads the source. You might want to use a PreSQL command if you have to update the source before you use the source in the mapping. For example, you can configure a PreSQL command to delete records in the source and update the source to load the latest records in the source database. When you run the mapping, the source that the mapping uses contains the latest records.

The Data Integration Service can also run PostSQL commands against the source database after it writes to the target. For example, you might want to use a PostSQL command to delete the records in the source after the records are loaded to the mapping target. You can configure a PostSQL command to delete the records after the mapping writes to the target.

Use the following guidelines when you configure PreSQL and PostSQL 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 PreSQL and PostSQL 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 push the 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.

Nonrelational Data Objects

Import a nonrelational data object to use in a mapping, mapplet, or profile. A nonrelational data object is a physical data object that uses a nonrelational data source.

You can import nonrelational data objects for the following connection types:

- Adabas
- IMS
- Sequential
- VSAM

When you import a nonrelational data object, the Developer tool reads the metadata for the object from its PowerExchange data map. A data map associates nonrelational records with relational tables so that the product can use the SQL language to access the data. To create a data map, use the PowerExchange Navigator.

After you import the object, you can include its nonrelational operations as read, write, or lookup transformations in mappings and mapplets. Each nonrelational operation corresponds to a relational table that the data map defines. To view the mapping of fields in one or more nonrelational records to columns in the relational table, double-click the nonrelational operation in the **Object Explorer** view.

For more information about data maps, see the *PowerExchange Navigator Guide*.

Notes:

- Before you work with nonrelational data objects that were created with Informatica 9.0.1, you must upgrade them. To upgrade nonrelational data objects, issue the `infacmd pwx UpgradeModels` command.
- When you run a mapping that includes an Update Strategy transformation and writes data to a nonrelational target, the mapping might fail with an error message. The message indicates that the target table has no keys, even though the data map for the target has keys defined.
- For nonrelational data objects with the Sequential connection type, you can parameterize the file name and default connection.

Importing Nonrelational Data Objects

Import one or more nonrelational data objects to use in a mapping, mapplet, or profile.

Before you import a nonrelational data object, you need to configure a connection to the database or data set. You also need to create a data map for the object.

1. Select a project or folder in the **Object Explorer** view.
2. Click **File > New > Data Object**.
3. Select **Non-relational Data Object** and click **Next**.
The **New Non-relational Data Object** dialog box appears.
4. Optionally, enter a name for the physical data object.
5. Click **Browse** next to the Connection option and select a connection.
6. Click **Browse** next to the Data Map option.
The **Select a Data Map** dialog box appears.
7. Optionally, enter filter criteria.
8. In the **Resources** area, expand the connection and schemas as needed and select the desired data maps.

9. Click **OK** to close the **Select a Data Map** dialog box, and then click **Finish**.

The nonrelational data objects and corresponding nonrelational operations appear under the connection name under **Physical Data Objects** in the project or folder in the **Object Explorer** view.

Note: You can also import nonrelational data objects by using the **Connection Explorer** view.

Creating a Read, Write, or Lookup Transformation from Nonrelational Data Operations

You can add a nonrelational data operation to a mapping or maplet as a read, write, or lookup transformation.

1. Open the mapping or maplet in which you want to create a read, write, or lookup transformation.
2. In the **Object Explorer** view, select one or more nonrelational data operations.
3. Drag the nonrelational data operations into the mapping editor.
The **Add to Mapping** dialog box appears.
4. Select the **Read, Write, or Lookup** option.
As independent data object(s) is automatically selected.
5. Click **OK**.
The Developer tool creates a read, write, or lookup transformation for each nonrelational data operation.

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 or 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.

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 or 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. When you create a web service from a WSDL data object, you can choose to include one or more operations.

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:

Property	Description
Name	Name of the WSDL data object.
Description	Description of the WSDL data object.

The following table describes the columns for operations defined in the WSDL data object:

Property	Description
Operation	The location where the WSDL defines the message format and protocol for the operation.
Input	The WSDL message name associated with the operation input.
Output	The WSDL message name associated with the operation output.
Fault	The WSDL message name associated with the operation fault.

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:

Property	Description
Connection	Default web service connection for a Web Service Consumer transformation.
File Location	Location where the WSDL file exists.

Importing a WSDL Data Object

To create a web service from a WSDL or 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 can use a WSDL data object to create a web service or 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 services and 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. For example, you edit a WSDL and change the name of an element from "Resp" to "RespMsg." You then synchronize the WSDL. When you open a web service that references the element, the Developer tool marks the web service name in the editor with an asterisk to indicate that the web service contains changes. The Developer tool updates the element name in the web service, but it cannot determine how the new element maps to a port. If the Resp element was mapped to a port in the Input transformation or the Output transformation, you must map the RespMsg element to the appropriate port.

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:

Property	Description
Host Name	Name of the server that hosts the URL.
Port Number	Port number of the URL.
Certificate File Path	Location of the client certificate file.
Password	Password for the client certificate file.

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 **Window > 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.

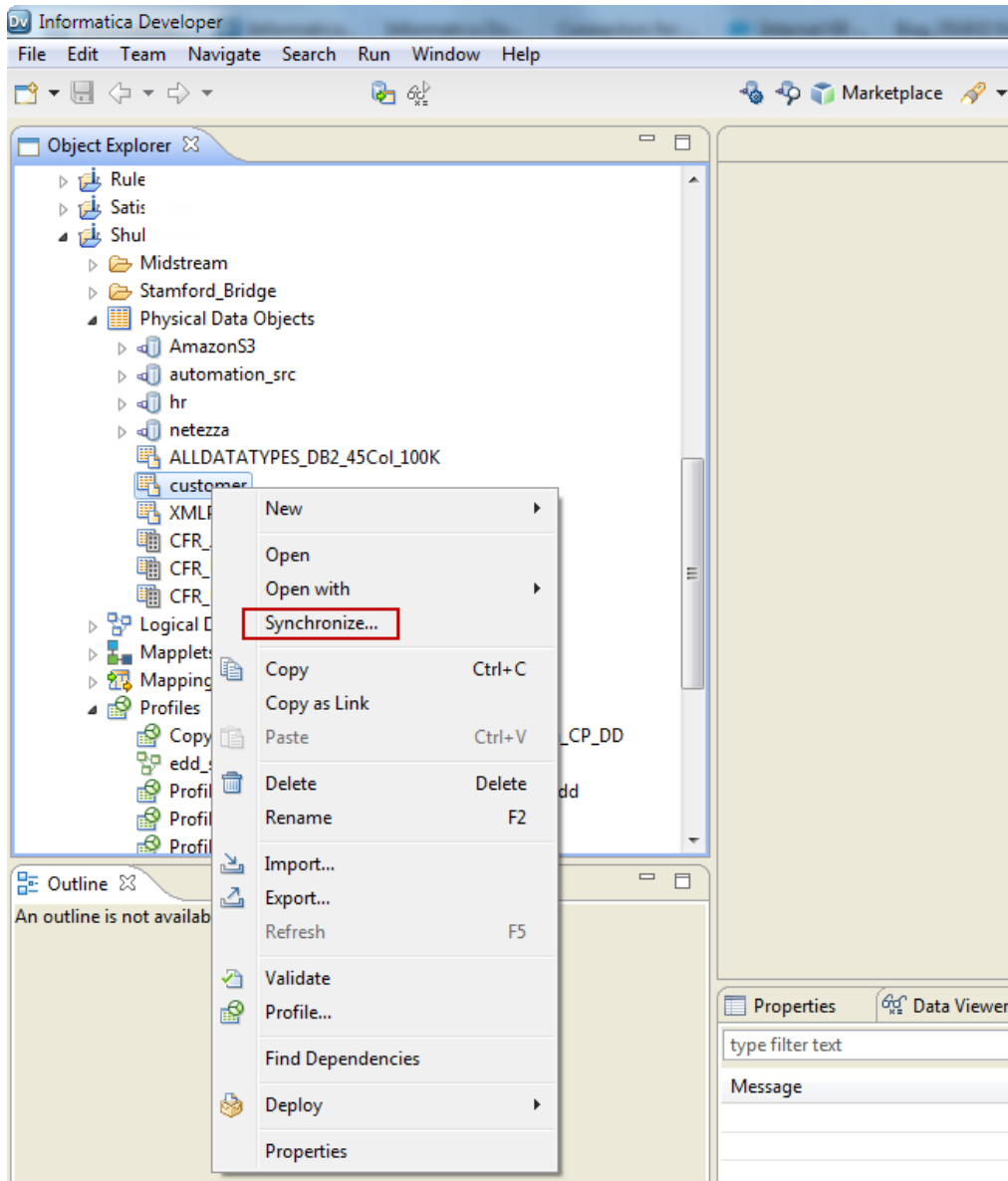
To synchronize any physical data object, right-click the object in the **Object Explorer** view, and select **Synchronize**.

Synchronizing a Flat File Data Object in Informatica Developer

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**.

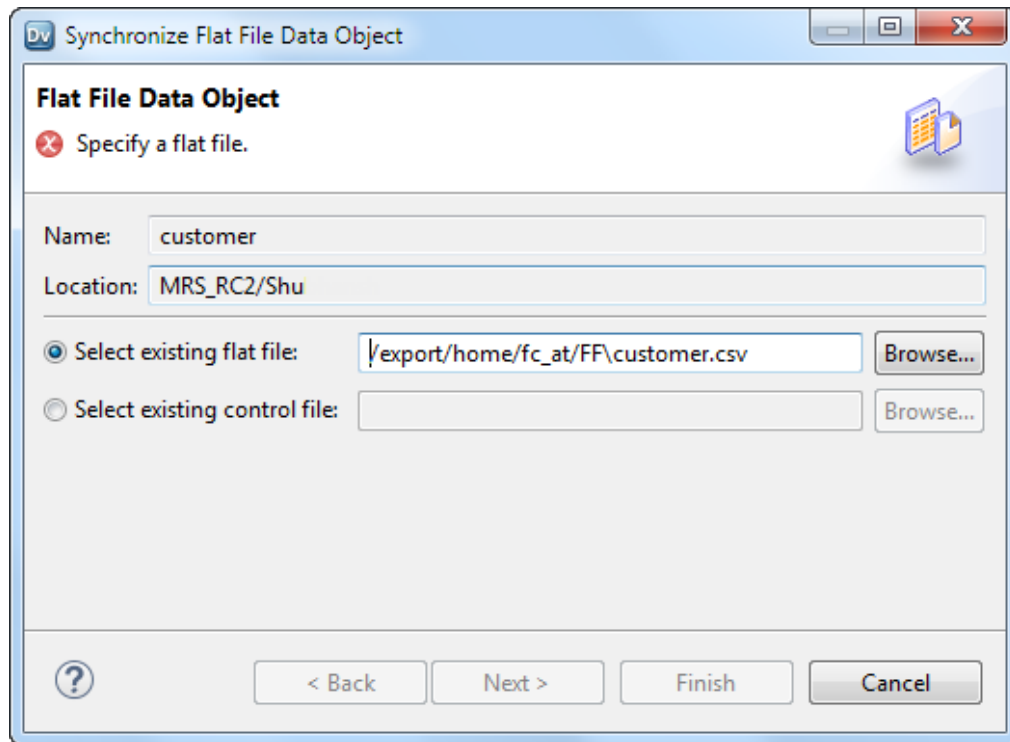
The following image shows the Synchronize option for a data object:



The **Synchronize Flat File Data Object** wizard appears.

3. In the **Synchronize Flat File Data Object** wizard, verify the flat file path in the **Select existing flat file** field.

The following image shows the Synchronize Flat File Data Object wizard:



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 in Informatica Developer

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**.
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.

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.

Mapping validation fails with Databricks physical objects.

If you used an escape character when you imported a physical data object from Databricks, mapping validation fails. Avoid escaping characters in object names.

CHAPTER 6

Flat File Data Objects

This chapter includes the following topics:

- [Flat File Data Objects Overview, 94](#)
- [Generate the Source File Name, 95](#)
- [Flat File Data Object Overview Properties, 95](#)
- [Flat File Data Object Advanced Properties, 96](#)
- [Control File, 103](#)
- [Update Columns at Run Time, 104](#)
- [Generate Column Metadata from Control Files, 105](#)
- [Copying from Excel to a Flat File Data Object, 108](#)
- [Create a Flat File Data Object, 109](#)

Flat File Data Objects Overview

Create a flat file physical data object to include in a mapping, mapplet, or profile. You can add flat file data objects to mappings and mapplets as Read, Write, and Lookup transformations. You can create and run a profile on flat file data objects.

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

You can configure a flat file data object to handle the changes in the flat file data source at run time. You can also generate column names for the flat file data object by using the information from the flat file or from the control file. The control file contains information on the column name, precision, scale, and the number of bytes to process.

After you create a flat file data object, use the following views to configure flat file properties:

Overview view

Configure the flat file data object name and description and update column properties.

Parameters view

Create parameters for the flat file data object.

Advanced view

Configure the format and run-time properties that the Data Integration Service uses when it reads data from and writes data to the flat file.

When you add flat file data objects to mappings as Read or Write transformations, you can view the format and run-time properties for the flat file data object in the **Properties** view. You cannot edit the flat file properties within a mapping.

Generate the Source File Name

You can add a file name column to the flat file data object. The file name column helps you identify the source file that contains a particular record of data. This is useful when the data comes from multiple sources.

You can configure the file name column in the Overview view of a flat file data object. The file name column is an optional column in the flat file data object. You can configure the mapping to write the source file name to each source row with the File Name Column port in the flat file data object. The file name column contains the fully qualified path and the file name.

The File Name Column port appears as the last column of the source data object. You can add only one File Name Column port in the source data object. When the port has the name as FileName and if you try to create a File Name Column port, the File Name Column port is named as FileName1.

For example, a mapping uses a source file that contains a list of files. The file names in the file list refers to the department names across the organization. For instance, the SYSA_Finance.txt file contains data from the Finance department. In the mapping, you can use string functions to extract the department name from the output of the file name column. You can use the extracted department name to process data differently for each department.

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:

Property	Description
Name	Name of the flat file data object.
Description	Description of the flat file data object.

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

Property	Description
Name	Name of the column or the file name column port.
Native type	Native datatype of the column.

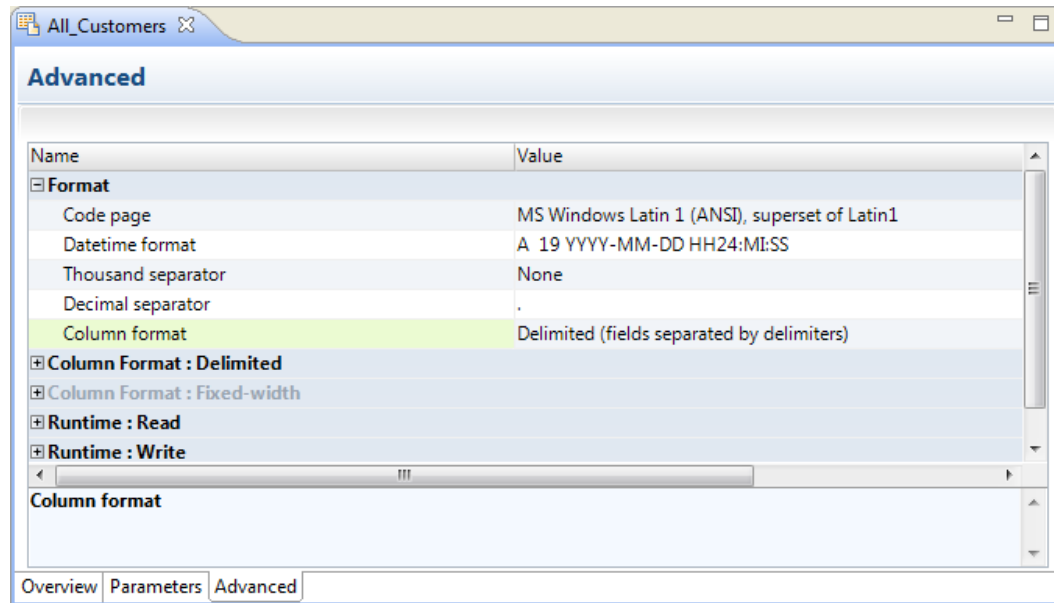
Property	Description
Bytes to process (fixed-width flat files)	Number of bytes that the Data Integration Service reads or writes for the column.
Precision	Maximum number of significant digits for numeric datatypes, or maximum number of characters for string datatypes. For numeric datatypes, precision includes scale.
Scale	Maximum number of digits after the decimal point for numeric values.
Format	Column format for numeric and datetime datatypes. For numeric datatypes, the format defines the thousand separator and decimal separator. Default is no thousand separator and a period (.) for the decimal separator. For datetime datatypes, the format defines the display format for year, month, day, and time. It also defines the field width. Default is "A 19 YYYY-MM-DD HH24:MI:SS."
Visibility	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.
Description	Description of the column.

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. Advanced properties include format and run-time properties. The Developer tool displays advanced properties for flat files in the **Advanced** view.

The **Advanced** view contains property sections that you can collapse and expand. The column format sections that display depend on whether you configure a delimited or fixed-width column format.

The following image shows the property sections in the **Advanced** view:



Format Properties

The Developer tool displays format properties for flat file data objects in the **Format** section in the **Advanced** view.

The following table describes the format properties that you configure for flat file data objects:

Property	Description
Code page	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 "MS Windows Latin 1 (ANSI), superset of Latin 1."
Datetime format	Defines the display format and the field width for datetime values. Default is "A 19 YYYY-MM-DD HH24:MI:SS."
Thousand separator	Thousand separator for numeric values. Default is None.
Decimal separator	Decimal separator for numeric values. Default is a period (.).
Column Format	Format for the flat file, either delimited or fixed-width.

Column Format: Delimited Properties

When the flat file is delimited, the Developer tool displays delimited properties in the **Column Format: Delimited** section in the **Advanced** view.

The following table describes the delimited properties that you configure for flat file data objects:

Property	Description
Delimiters	Character used to separate columns of data. Click the Delimiters field to select a character or to assign a parameter to the property. Delimiters must be printable characters and must be different from the text qualifier and the escape character if selected. Default is Comma.
Text qualifier	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. Default is No Quotes.
Start import at line	Row at which the Data Integration Service starts importing data. Use this option to skip header rows. Default is 1.
Row delimiter	Octal code for the character that separates rows of data. Default is line feed, \012 LF (\n). Note: The row delimiter applies to reading source data. When the Data Integration Service writes to a target file, it always uses the default delimiter, \n.
Escape character	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. Note: You can improve mapping performance slightly if the source file does not contain quotes or escape characters.
Retain escape character in data	Includes the escape character in the output string. Default is disabled.
Treat consecutive delimiters as one	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.

Column Format: Fixed-width Properties

When the flat file is fixed-width, the Developer tool displays fixed-width properties in the **Column Format: Fixed-width** section in the **Advanced** view.

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

Property	Description
Null character type	Null character type, either text or binary.
Null character value	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.
Repeat null character	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.
Start import at line	Row at which the Data Integration Service starts importing data. Use this option to skip header rows. Default is 1.
Number of bytes to skip between records	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.
Line sequential	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.
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: Read Properties

The Developer tool displays run-time properties for flat file sources in the **Run-time: Read section** in the **Advanced** view. The Data Integration Service uses this information when it reads data from a flat file.

The following table describes the read properties that you configure for flat file data objects:

Property	Description
Input type	Type of source input. You can choose the following types of source input: <ul style="list-style-type: none"> - File. For flat file sources. - Command. For source data or a file list generated by a shell command.
Source type	Indicates source type of files with the same file properties. You can choose one of the following source types: <ul style="list-style-type: none"> - Direct. A source file that contains the source data. - 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. - 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.
Source file name	File name of the flat file source.
Source file directory	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.
Concurrent Read Partitioning	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: <ul style="list-style-type: none"> - Optimize throughput. The Data Integration Service does not preserve input row order. - Keep relative order. The Data Integration Service preserves the input row order for the rows read by each partition. - Keep absolute order. The Data Integration Service preserves the input row order for all rows read by all partitions.
Connection Type	The type of connection. Select from the following options: <ul style="list-style-type: none"> - None. The source file does not require a connection. - Hadoop File System. The source file resides in HDFS. Default is None.
Command	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>

Property	Description
Truncate string null	<p>Strips the first null character and all characters after the first null character from string values.</p> <p>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.</p> <p>Default is disabled.</p>
Line sequential buffer length	<p>Number of bytes that the Data Integration Service reads for each line.</p> <p>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.</p> <p>Default is 1024.</p>
Generate Run-time Column Names	<p>Determines how to generate the column metadata at run time.</p> <p>Select one of the following options:</p> <ul style="list-style-type: none"> - Automatically (Field1, Field 2...). The Data Integration Service includes column metadata based on the position of the column in the data. The column metadata for the flat file is constructed at run time from the row that the user specifies in the start from import line attribute in the format properties of the flat file data object. - From data file header (first line). The first row of the flat file contains a header row that the Data Integration Service uses to determine the column metadata. - From control file. The Data Integration Service constructs column metadata based on the data in a control file, such as column name, precision, data type, scale, and number of bytes to process.
Control file name	Name of the control file. Required if you generate run-time column names from control file.
Control file directory	Directory where the control file exist. Required if you generate run-time column names from control file.
Default Field Type	Data type of the additional ports generated at run time.
Default Precision	Precision of the additional ports generated at run time.
Default Scale	Scale of the additional ports generated at run time.
Constraints	<p>Conditional expression that the values on a data row must satisfy. Use the Expression editor to enter an expression that evaluates to TRUE. When the Data Integration Service reads constraints, it drops the rows that do not evaluate to TRUE.</p> <p>For example, a source flat file has an AGE column. You can set a constraint with AGE < 70 on the flat file data object. The Data Integration Service reads rows from the source flat file with the constraint AGE < 70. If the Data Integration Service reads rows with AGE >= 70, it drops those rows.</p>

Run-time: Write Properties

The Developer tool displays run-time properties for flat file targets in the **Run-time: Write section** in the **Advanced** view. The Data Integration Service uses this information when it writes data to a flat file.

The following table describes the write properties that you configure for flat file data objects:

Property	Description
Append if exists	<p>Appends the output data to the target files and reject files.</p> <p>If you do not select this option, the Data Integration Service truncates the target file and reject file before writing data to them. If the files do not exist, the Data Integration Service creates them.</p> <p>Default is disabled.</p>
Create directory if not exists	<p>Creates the target directory if it does not exist.</p> <p>Default is disabled.</p>
Header options	<p>Creates a header row in the file target. You can choose the following options:</p> <ul style="list-style-type: none"> - No header. Does not create a header row in the flat file target. - Output field names. Creates a header row in the file target with the output port names . - Use header command output. Uses the command in the Header Command field to generate a header row. For example, you can use a command to add the date to a header row for the file target. <p>Default is no header.</p>
Header command	Command used to generate the header row in the file target.
Footer command	Command used to generate the footer row in the file target.
Output type	Type of target for the mapping. Select File to write the target data to a flat file. Select Command to output data to a command.
Command	<p>Command used to process the target data.</p> <p>On UNIX, use any valid UNIX command or shell script. For example, use the following command to generate a compressed file from the target data on UNIX:</p> <pre>compress -c - > MyTargetFiles/MyCompressedFile.Z</pre> <p>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. For example, use cmd as the target command on Windows to avoid staging data in the file system and to avoid any security breaches.</p> <p>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.</p>
Merge command	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.
Output file directory	<p>Output directory for the flat file target. The machine that hosts Informatica services must be able to access this directory.</p> <p>Enter multiple directories separated by semicolons to increase performance when multiple partitions write to the flat file target.</p> <p>Default is the TargetDir system parameter.</p>

Property	Description
Output file name	File name of the flat file target. If multiple partitions write to the flat file target and you choose not to merge target data, each partition writes to a separate output file named <output_file_name><partition_number>.out.
Merge type	Type of merge that the Data Integration Service performs on the data for partitioned targets. If the Data Integration Service does not create partitions for the target, it ignores this value. Select one of the following options: <ul style="list-style-type: none"> - No merge. The Data Integration Service concurrently writes the target output to a separate file for each partition. - Sequential. The Data Integration Service creates an output file for each partition and then merges them into a single merge file at the end of the mapping. - File list. The Data Integration Service creates a target file for each partition and creates a file list that contains the paths of the individual files. - Concurrent. The Data Integration Service concurrently writes the data for all target partitions to the merge file. Because the Data Integration Service writes to the merge file concurrently for all partitions, the sort order of the data in the merge file might not be sequential.
Merge file directory	Directory for the merge file for all target partitions. The machine that hosts Informatica services must be able to access this directory. Default is the TargetDir system parameter.
Merge file name	Name of the merge file for all target partitions. Default is the output file name.
Connection type	The type of connection. Select from the following options: <ul style="list-style-type: none"> - None. The target file does not require a connection. The target file location is specified by the output file directory. - Hadoop File System. The target file is in HDFS. Default is None.

Control File

The Data Integration Service can update the column metadata for the flat file based on a control file.

A control file is a simple text file with a field name, data type, precision, and scale. Each line in the control file contains one column of data.

To generate column names from a control file at run-time, select **From control file** for the **Generate Run-time Column Names** property in the Advanced properties of the flat file data object. You must also configure the Read transformation to get the column metadata at run time.

You can specify any of the following data types in the control file:

- Bigint
- Date/Time
- Decimal
- Double
- Integer

- String
- Text
- TimestampwithTZ

You can use a control file to generate run-time column names for a Read transformation based on a flat file data object or to create a flat file data object. The Data Integration Service constructs the column metadata for the flat file by using the information present in the control file. The column name, precision, data type, and scale attributes are comma delimited. The newline character delimits information across columns.

Update Columns at Run Time

You can configure a flat file data object to accommodate changes to source metadata at run time. The Data Integration Service can read data from a flat file where the number or order of the columns is different from that of the columns in the flat file physical data object.

For example, you have a mapping whose source is provided by another department. The department that provides the source cannot guarantee the order of columns in the source file. Sometimes, the department might change the columns in the file or contain additional columns. You can configure the flat file data object to accept changes in the source metadata at run time.

You can generate the run-time column names automatically from any row in the source, from the flat file header, or from the control file.

You can use input rules to project the run-time columns from the Read transformation based on the flat file data object to the next transformation in the mapping.

You can configure the Read transformation data object property to select **At runtime, get data object columns from the data source** on the Data Object tab. After you configure the property, the options in the flat file data object determine how the Data Integration Service processes the data.

Generate Run-time Column Names Automatically

The Data Integration Service can update the column metadata for the flat file automatically based on the column position.

The Data Integration Service dynamically updates the column metadata for the flat file based on the row that you specify to start the import in the format properties of the flat file. The default data type, precision, and scale is used for the column metadata for the run-time columns without changing the column position.

To generate column names automatically at run time, select **Automatically (Field1, Field 2...)** for the **Generate Run-time Column Names** property in the Advanced properties of the flat file data object. You must also configure the Read transformation to get the column metadata at run time.

For example, you want to configure a flat file data object to accept changes at run time. The data in the flat file determines the column metadata based on the column position. You have defined a flat file data object with the columns `Dept`, `Name`, `Place`. You want the Data Integration Service to dynamically construct the column metadata for the flat file at run time.

The input file contains the following information:

```
HR,Bob,Chicago,US,87675
Finance,Mary,California,US,65437,t567,4200
```

In the Format properties, you configured the flat file data object to start the import of data from line 1. Since there are five columns in the first row, the Data Integration Service processes the flat file data object with five

columns. The Data Integration Service creates additional run-time columns in the flat file data object with unique names based on position, Field4 and Field5. The Data Integration Service does not process the additional columns in the second row.

When you add a flat file data object to a mapping, configure the Read transformation to get column metadata at run time. At run time, the Data Integration Service constructs the flat file data object and process the rows in the flat file with the following values:

Dept	Name	Place	Field4	Field5
HR	Bob	Chicago	US	87675
Finance	Mary	California	US	65437

Generate Run-time Column Names From Data File Header

The Data Integration Service can use a header row to determine the column metadata. The first row of the flat file is the header.

The default data type, precision, and scale is used for the column metadata for the run-time columns and you can change the column position at run time.

To generate column names from a data file header at run-time, select **From data file header (first line)** for the **Generate Run-time Column Names** property in the Advanced properties of the flat file data object. You must also configure the Read transformation to get the column metadata at run time.

For example, you want to configure a flat file data object to accept changes at run time. The data in the flat file determines the column metadata based on the data file header.

You have defined a flat file data object with the columns *Name*, *Age*, *Dept*. You can configure the flat file data object advanced property to generate column names at run time from a data file header.

The data flat file contains the following information:

Dept	Name	Place	Country	Phone_No.
HR	Bob	Chicago	US	87675
Finance	Mary	California	US	65437

When you add a flat file data object to a mapping, configure the Read transformation to get column metadata at run time. During run time, the Data Integration Service generates the flat file data object with the following columns: *Dept*, *Name*, *Place*, *Country* and *Phone_No*. Since *Age* column is not present in the flat file header row, the flat file data object constructed at run time does not have the *Age* column.

The Data Integration Service propagates the run-time columns, such as *Country* and *Phone Number* to the downstream transformation based on the mapping rules.

Generate Column Metadata from Control Files

Control files are simple text files based on which you can create flat file data objects. You can also use control files to update columns at run time for a Read transformation based on a flat file data object.

When the Data Integration Service generates columns at run time, it uses the decimal and thousands separators specified in the flat file data object properties. You must specify only the name of the column in the control file if the column exists in the flat file data object.

When you specify the source type as indirect in the run-time properties in the Advanced view of the flat file source, you can generate column names at run time with the control file.

You can create a control file for a delimited or for a fixed-width flat file.

You can parameterize the control file name and the control file directory in the run-time properties of the flat file data object.

The Data Integration Service processes a control file based on the column metadata that you specify in the control file and the input data.

When you generate column names from a control file at run time and the source data object contains the file name column, the Data Integration Service logs a validation error with the duplicate file names.

Control File Formats

Delimited and fixed-width flat files have different control file formats. You can use control files to generate column metadata at run time or to create a flat file data object.

Use one of the following formats to create a control file:

Delimited control file

Each line in a delimited control file has the following format:

```
[column name],<data type>,<precision>,<scale>
```

The following example shows a delimited flat file control file:

```
Dept,String,10  
Name  
Place,String,20  
Country  
Phone number,string,30
```

Fixed-width control file

Each line in a fixed-width control file has the following format:

```
[column name],<data type>,<precision>,<scale>,<number of bytes to process>
```

The following example shows a delimited flat file control file:

```
Dept,String,10  
Name  
Place,String,20  
Country  
Phone number,string,30
```

For a fixed-width source, the control file contains an additional column for the number of bytes to process. If you do not specify the number of bytes to process, the Data Integration Service uses the value specified for the precision as the number of bytes to process.

Parameterization of Run-time Properties

You can parameterize or specify values for the default scale, precision, and data type of the additional columns from the control file in the run-time properties of the flat file data object. You can parameterize the control file name and the control file directory in the run-time properties of the flat file data object. Configure the parameters on the Advanced tab of the physical data object properties. When you create the transformation from the physical data object, you can use mapping parameters to override the parameter default values.

Run-time Processing of Control Files

When you develop a mapping, you define data object read properties that determine how data is read from a flat file. The Data Integration Service can process the columns of the flat file data object that is based on a control file.

When the Data Integration Service constructs the column metadata based on the control file, it applies the following criteria for processing data:

The Data Integration Service applies default values for the column properties that you do not specify for the flat file data object

The Data Integration Service applies the default run-time properties of the flat file data object when you do not specify the run-time column properties in the control file. When you have additional columns in the control file without a data type, precision, or scale, the Data Integration Service uses the default data type, precision, and scale.

The Data Integration Service processes data as NULL for columns that do not appear in the control file but are present in the flat file data object

When the column in the flat file data object does not exist in the control file, the Data Integration Service processes the data as NULL during data preview. During run time, the Data Integration Service cannot process the column as that column does not exist in the control file.

The Data Integration Service fails mapping when the source metadata mismatches with a control file

The Data Integration Service fails to process the data when the source metadata based on the control file does not match the input data. For example, the Data Integration Service fails the mapping when the source contains a data type in the control file that is not compatible with the data type in the source.

Rules and Guidelines for Control Files

Consider the following guidelines when you use a control file:

- The control file must be accessible to the machine where the Data Integration Service runs.
- After you import a control file as a flat file data object, you cannot change the data type, precision, and scale of the columns in the data object. You can change the column position and column metadata for the additional columns from the control file.
- When you generate column names at run time based on the control file, the lookup cache reflects the metadata in the control file. If you update the same control file with additional columns after you run the mapping once, you must select the **Recache from Lookup source** property from the Runtime properties of the Lookup transformation.
- You cannot specify date format columns in the control file. The Data Integration Service uses the flat file date format from the flat file data object advanced properties in the Advanced view. If the flat file source contains multiple date formats, the Data Integration Service will use only the one date format specified in the Advanced view for all the dates.
- You cannot specify a binary data type in the control file. If the control file contains a TimestampwithTZ data type, ensure that the data type has the precision set to 36 and the scale set to 9 in the control file.

Copying from Excel to a Flat File Data Object

You can configure flat file properties in Excel and copy them into a flat file data object in the Developer tool. Flat file properties include the column name, native type, precision, and scale. You might want to do this when you need to develop or edit a flat file data object with many columns.

Note: You must confirm that the values in each cell are valid before copying the values to a flat file data object. For example, a string type cannot have a scale value other than "0." Precision values cannot be words and type values cannot be numbers. If your information is incorrect, you will receive an error message.

Editing Flat File Data Objects in Excel

To edit a large portion of a flat file data object, you do not need to change every value in the Developer tool. Instead, you can copy the flat file columns to Excel, change all the values simultaneously using Auto Fill, and then **Paste (Replace)** the flat file back into the Developer tool.

1. To copy the metadata from the Developer tool, right-click within the flat file data object and click **Select All**.
2. Copy the metadata to an Excel spreadsheet.
3. Make changes within the Excel spreadsheet.
4. Copy the metadata from Excel.
5. To update the flat file data object with the changes, right-click within the flat file columns and click **Paste (Replace)**.

Copying Metadata to a Flat File Data Object

You can create metadata in Excel and then copy it to a flat file data object in the Developer tool.

1. Create a mapping that includes a flat file data object.
2. Define metadata for a flat file data object in Excel.
3. Copy the metadata from Excel.
4. To move the metadata to the flat file data object, right-click within columns and click **Paste (Replace)**. The following image shows a sample Excel table and the resulting flat file data object after you copy the metadata to the Developer tool:

The diagram illustrates the process of copying metadata from an Excel spreadsheet to a flat file data object. On the left, an Excel spreadsheet shows a table with columns A, B, C, and D. The rows contain metadata for various fields: Name, Native type, Precision, and Scale. The fields listed are busn_addr_zip_code, cust_day_phone, busn_addr_cntry_code, pmry_email_addr, busn_srvc_email, and cust_acct_type_code. An arrow points from this Excel table to a 'Columns' table on the right, which contains the same metadata copied into the Developer tool.

	A	B	C	D
1	Name	Native type	Precision	Scale
2	busn_addr_zip_code	string	60	0
3	cust_day_phone	string	40	0
4	busn_addr_cntry_code	string	3	0
5	pmry_email_addr	string	127	0
6	busn_srvc_email	string	127	0
7	cust_acct_type_code	number	1	0

Columns				
	Name	Native Type	Precision	Scale
1	busn_addr_zip_code	string	60	0
2	cust_day_phone	string	40	0
3	busn_addr_cntry_code	string	3	0
4	pmry_email_addr	string	127	0
5	busn_srvc_email	string	127	0
6	cust_acct_type_code	number	1	0

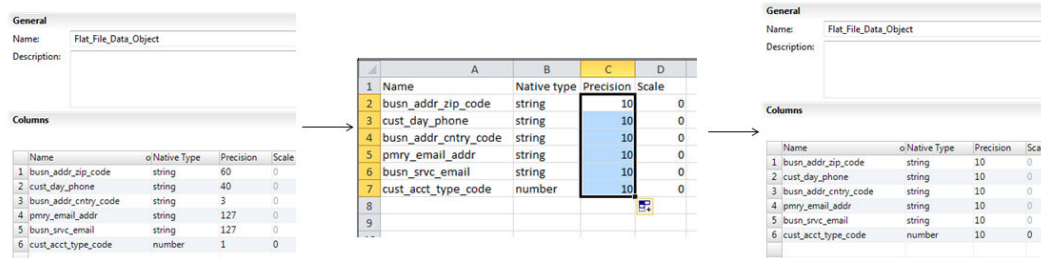
You can still make changes to the flat file data object after you copy the metadata to the Developer tool.

Example, Editing Data Object in Excel

You are developing a flat file data object and you need to change all the precision values to 10. Instead of changing each field individually, you make global changes through Excel.

You copy the metadata from the Developer tool into Excel, make changes to it, and copy it back to the flat file columns in the Developer tool. By using Excel, you avoid having to change each field individually.

The following image shows the process of moving a flat file to Excel, using Auto Fill to change certain values, and then copying the metadata back to the flat file data object in the Developer tool:



Create a Flat File Data Object

You can create a flat file data object as an empty data object, from an existing flat file, or from a control file. You can create a delimited or fixed-width flat file data object using any method.

When you create a flat file data object, you can create the data object in the following ways:

As an empty data object

Create an empty flat file data object when you want to define the data object columns and rows in the Developer tool.

From an existing flat file

Create a flat file data object from an existing flat file when you have a flat file that defines the metadata you want to include in the data object.

From a control file

Create a flat file data object from a control file when you want to define the data object columns and rows based on a control file.

Creating an Empty Flat File Data Object

Create an empty flat file data object when you want to define the data object columns and rows in the Developer tool.

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 that you want to process.

9. Select **Delimited** or **Fixed-width**.
10. If you selected **Fixed-width**, click **Finish**. If you selected **Delimited**, click **Next**.
11. Configure the following delimited format properties:

Property	Description
Delimiters	Character used to separate columns of data. If you enter a delimiter that is the same as the escape character or the text qualifier, you might receive unexpected results.
Text Qualifier	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.

12. Click **Finish**.
The empty data object opens in the editor. Define the columns for the data object in the **Overview** view.

Creating a Flat File Data Object from an Existing Flat File

Create a flat file data object from an existing flat file when you have a flat file that defines the metadata you want to include in the data object.

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 from an existing flat file**.
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. Optionally, click **Browse** to select a project or folder for the data object.
9. Click **Next**.
10. Select a code page that matches the code page of the data that you want to process.
11. Select **Delimited** or **Fixed-width**.
12. Optionally, edit the maximum number of rows to preview.
13. Click **Next**.
14. Configure the format properties, based on whether the flat file is delimited or fixed-width.

- For a delimited flat file, configure the following properties:

Property	Description
Delimiters	Character used to separate columns of data. If you enter a delimiter that is the same as the escape character or the text qualifier, you might receive unexpected results.
Text Qualifier	Quote character that defines the boundaries of text strings. If you select a quote character, the Developer tool ignores delimiters within pairs of quotes.
Import Column Names From First Line	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 "FIELD_" to field names that are not valid.
Row Delimiter	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).
Escape Character	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.
Start Import at Line	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.
Treat Consecutive Delimiters as One	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.
Retain Escape Character in Data	Includes the escape character in the output string.

- For a fixed-width flat file, configure the following properties:

Property	Description
Import Column Names From First Line	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.
Start Import at Line	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.
Column breaks	Configures the column breaks in the fixed-width file. 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.

15. Click **Next** to preview the flat file data object.

16. Click **Finish**.

The data object opens in the editor.

Creating a Flat File Data Object from a Control File

Create a flat file data object from a control file when you want to define the data object columns and rows based on a control file.

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 from a control file**.
5. Click **Browse** and navigate to the directory that contains the control file.
6. Click **Open**.
The wizard names the data object the same name as the control file you selected.
7. Optionally, edit the data object name.
8. Optionally, click **Browse** to select a project or folder for the data object.
9. Click **Next**.
10. Select a code page that matches the code page of the data that you want to process.
11. Select **Delimited** or **Fixed-width**.
12. If you selected **Fixed-width**, click **Finish**. If you selected **Delimited**, click **Next**.
13. Configure the following delimited format properties:

Property	Description
Delimiters	Character used to separate columns of data. If you enter a delimiter that is the same as the escape character or the text qualifier, you might receive unexpected results.
Text Qualifier	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.

14. Click **Finish**.
The data object opens in the editor.

CHAPTER 7

Logical View of Data

This chapter includes the following topics:

- [Logical View of Data Overview, 113](#)
- [Developing a Logical View of Data, 114](#)
- [Logical Data Object Models, 115](#)
- [Logical Data Object Model Properties, 116](#)
- [Logical Data Objects, 123](#)
- [Logical Data Object Mappings, 127](#)

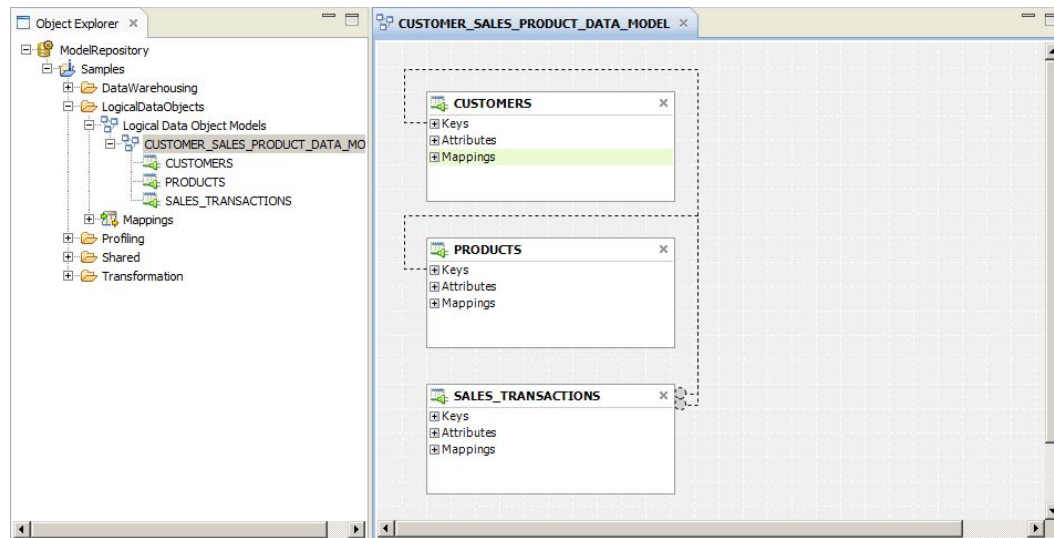
Logical View of Data Overview

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.

The following figure shows a sample of related logical data objects:



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.

After you develop a logical view of data, you can add it to a data service to make virtual data available for end users.

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. The transformations can include data quality transformations to validate and cleanse 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.

To allow end users to run SQL queries against a logical data object, include it in an SQL data service. Make the logical data object the source for a virtual table. To allow end users to access a logical data object over the Web, include it in a web service. Make the logical data object the source for an operation.

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:

Property	Description
Import UDPs	<p>Specifies how to import user-defined properties.</p> <p>Select one of the following options:</p> <ul style="list-style-type: none"> - As metadata. Import an explicit value as the property value object. Explicit values are not exported. - As metadata, migrate default values. Import explicit and implicit values as property value objects. - In description, migrate default values. Append the property name and value, even if implicit, to the object description property. - Both, migrate default values. Import the UDP value, even if implicit, both as metadata and in the object's description. <p>Default is As metadata.</p>
Import relationship name	<p>Specifies how to import the relationship names from ERwin.</p> <p>Select one of the following options:</p> <ul style="list-style-type: none"> - From relationship name - From relationship description <p>Default is From relationship name.</p>
Import IDs	<p>Specifies whether to set the unique ID of the object as the NativeId property.</p>
Import subject areas	<p>Specifies how to import the subject areas from ERwin.</p> <p>Select one of the following options:</p> <ul style="list-style-type: none"> - As diagrams - As packages and diagrams - As packages and diagrams, assuming one subject area for each entity - Do not import subject areas <p>Default is As diagrams.</p>
Import column order form	<p>Specifies how to import the position of columns in tables.</p> <p>Select one of the following options:</p> <ul style="list-style-type: none"> - Column order. Order of the columns displayed in the ERwin physical view. - Physical order. Order of the columns in the database, as generated in the SQL DDL. <p>Default is Physical order.</p>
Import owner schemas	<p>Specifies whether to import owner schemas.</p>

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:

Property	Description
Folder Representation	Specifies how to represent folders from the Framework Manager. Select one of the following options: <ul style="list-style-type: none">- Ignore. Ignore folders.- Flat. Represent folders as diagrams but do not preserve hierarchy.- Hierarchical. Represent folders as diagrams and preserve hierarchy. Default is Ignore.
Package Representation	Specifies how to represent packages from Cognos Framework Manager. Select one of the following options: <ul style="list-style-type: none">- Ignore. Ignore subject areas.- Subject Areas. Represent packages as subject areas.- Model. Represent the package as the model. Default is Ignore.
Reverse engineer relationships	Specifies whether the Developer tool computes the relationship between two dbQueries as referential integrity constraints.
Tables design level	Specifies how to control the design level of the imported tables: Select one of the following options: <ul style="list-style-type: none">- 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.
Ignore usage property	Specify whether the usage property of a queryItem should be used.

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:

Property	Description
System	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
Authentication mode	Login authentication mode. This parameter is applicable to SAP BusinessObjects Designer 11.0 and later. Select one of the following authentication modes: <ul style="list-style-type: none"> - 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.
User name	User name in the BusinessObjects server. For version 11.x and 12.x (XI), you need to be a member of BusinessObjects groups.
Password	Password for the BusinessObjects server.
Silent execution	Specifies whether to execute in interactive or silent mode. Default is Silent.
Close after execution	Specify whether to close BusinessObjects after the Developer Tool completes the model import.
Table design level	Specifies the design level of the imported tables. Select one of the following options: <ul style="list-style-type: none"> - 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.
Transform Joins to Foreign Keys	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.

Property	Description
Class representation	<p>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.</p> <p>Select one of the following options:</p> <ul style="list-style-type: none"> - 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. <p>Default is As a flat structure.</p>
Include List of Values	<p>Controls how the Developer tool imports the list of values associated with objects.</p>
Dimensional properties transformation	<p>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.</p> <p>Select one of the following options:</p> <ul style="list-style-type: none"> - 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. <p>Default is Disabled.</p>

SAP PowerDesigner CDM Import Properties

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

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

Property	Description
Import UDPs	<p>Specifies how to import user-defined properties.</p> <p>Select one of the following options:</p> <ul style="list-style-type: none"> - As metadata. Import an explicit value as the property value object. Explicit values are not exported. - As metadata, migrate default values. Import explicit and implicit values as property value objects. - In description, migrate default values. Append the property name and value, even if implicit, to the object description property. - Both, migrate default values. Import the UDP value, even if implicit, both as metadata and in the object's description. <p>Default is As metadata.</p>
Import Association Classes	<p>Specifies whether the Developer tool should import association classes.</p>
Import IDs	<p>Specifies whether to set the unique ID of the object as the NativeId property.</p>

Property	Description
Append volumetric information to the description field	Import and append the number of occurrences information to the description property.
Remove text formatting	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.

SAP PowerDesigner PDM Import Properties

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

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

Property	Description
Import UDPs	Specifies how to import user-defined properties. Select one of the following options: <ul style="list-style-type: none"> - As metadata. Import an explicit value as the property value object. Explicit values are not exported. - As metadata, migrate default values. Import explicit and implicit values as property value objects. - In description, migrate default values. Append the property name and value, even if implicit, to the object description property. - Both, migrate default values. Import the UDP value, even if implicit, both as metadata and in the object's description. Default is As metadata.
Import IDs	Specifies whether to set the unique id of the object as the NativeId property.
Append volumetric information to the description field	Import and append the number of occurrences information to the description property.
Remove text formatting	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.

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:

Property	Description
Elements content name	Attribute to hold the textual content like #PCDATA in the XSD file. Default is As metadata.
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: <ul style="list-style-type: none">- 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: <ul style="list-style-type: none">- 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 in the editor and on editor tabs. The following image shows the logical data object editor:

Department Overview

General

Name: Department

Description:

Read Mapping: [Department Read Mapping](#) Add... Delete

Write Mapping: Add... Delete

Department	
Name	Type
DEPTNO	decimal
DNAME	string
LOC	string
FIRSTNAME	string
LASTNAME	string

Attributes

Search

	Name	Type	Primary Key	Precision	Scale	Nullable	Description
1	DEPTNO	decimal	<input type="checkbox"/>	2	0	<input checked="" type="checkbox"/>	
2	DNAME	string	<input type="checkbox"/>	10	0	<input type="checkbox"/>	
3	LOC	string	<input type="checkbox"/>	8	0	<input checked="" type="checkbox"/>	
4	FIRSTNAME	string	<input type="checkbox"/>	20	0	<input type="checkbox"/>	
5	LASTNAME	string	<input type="checkbox"/>	20	0	<input checked="" type="checkbox"/>	

Overview | Keys | Relationships | Access | Read Mapping | Advanced

The following table describes information that appears in the logical data object editor:

Tab Name	Description
Overview	The General area includes the object name, description, and read and write mapping if applicable, of the logical data object. The Attributes area displays the structure of data in a logical data object.
Keys	One or more attributes in a logical data object can be primary keys or unique keys.
Relationships	Associations between logical data objects.
Access	Type of access for a logical data object and each attribute of the data object.
Read Mapping	Logical data object read mapping associated with a logical data object. If the logical data object contains no read mapping, the Read Mapping tab is not visible.
Write Mapping	Logical data object write mapping associated with a logical data object. If the logical data object contains no write mapping, the Write Mapping tab is not visible.

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.

- On the **Overview** tab in the General area, you can optionally edit the logical data object name and description.
- On the **Overview** tab in the Attributes area, you can create attributes and specify their datatype and precision.

You can use the options in the toolbar to update the attribute values. The available options are New, Cut, Copy, Paste, Delete, Move Up, Move Down, Enable All Nullable, and Disable All Nullable. You can select or clear all nullable values of the attributes using Enable All Nullable or Disable All Nullable icons.

Department

Overview

General

Name:

Description:

Read Mapping:

Write Mapping:

Attributes

Search

	Name	Type	Primary Key	Precision	Scale	Nullable	Description
1	DEPTNO	decimal	<input type="checkbox"/>	2	0	<input checked="" type="checkbox"/>	
2	DNAME	string	<input type="checkbox"/>	10	0	<input type="checkbox"/>	
3	LOC	string	<input type="checkbox"/>	8	0	<input checked="" type="checkbox"/>	
4	FIRSTNAME	string	<input type="checkbox"/>	20	0	<input type="checkbox"/>	
5	LASTNAME	string	<input type="checkbox"/>	20	0	<input checked="" type="checkbox"/>	

Overview | Keys | Relationships | Access | Read Mapping | Advanced

- On the **Keys** tab, optionally specify primary and unique keys for the data object.

Keys

Keys: Key

Add... Remove

Name:

Description:

Available Attributes:

- Status
- LastName
- FirstName

Selected Attributes:

- CUSTID

Overview | Keys | Relationships | Access | Read Mapping | Advanced

- On the **Relationships** tab, optionally create relationships between logical data objects.

Relationships

Relationships: Relationship

Add... Remove

Name:

Description:

Relationship Type:

- Identifying
- Non-Identifying

Referenced Key:

Key: Browse...

Attributes:

- CUSTID

Logical Data Object

Available Attributes:

- CUSTID
- Status
- LastName
- FirstName

Selected Attributes:

- CUSTID1

Overview | Keys | Relationships | Access | Read Mapping | Advanced

- 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.

Access

Write Access:

Attributes:

	Name	Type	Write Access
1	CUSTID	decimal	Read Only
2	Status	string	Read Only
3	LastName	string	Read Only
4	FirstName	string	Read Only
5	CUSTID1	decimal	Read Only

Overview | Keys | Relationships | Access | Read Mapping | Advanced

- On the **Read Mapping** tab, optionally create a logical data object read mapping.

Read Mapping

Overview | Keys | Relationships | Access | Read Mapping | Advanced

- On the **Write Mapping** tab, optionally create a logical data object write mapping.
- Save the logical data object.

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 **Object Explorer** view, double-click the logical data object that you want to add the mapping to physical data objects.

The logical data object editor opens.

2. On the **Overview** tab in the General area, click **Add** to add a read mapping or a write mapping.
3. Enter a name for the mapping, and then 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.

4. Link data from a physical data object to the logical data object.

- a. Click the **Read Mapping** or **Write Mapping** tab to edit the mapping.

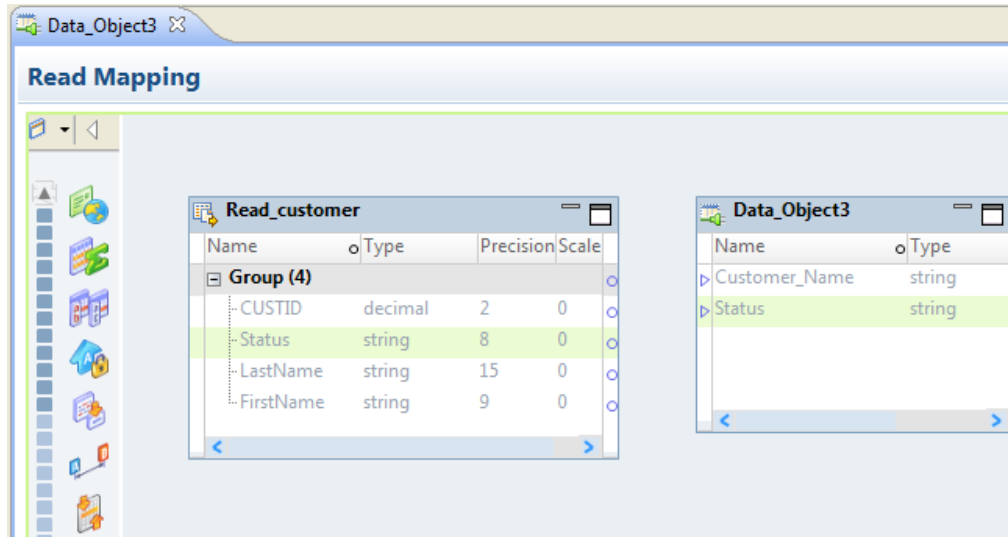
The mapping editor palette opens.

- b. In the **Object Explorer** view, browse to the physical data object you want to link, and drag the physical data object into the mapping editor palette.

The **Add to Mapping** dialog box opens.

- c. Choose to create a read mapping, write mapping, or lookup mapping.
- d. If you chose to create a read mapping, designate the object access of the read mapping as a related data object or as an independent data object.
- e. Click **OK**.

The following image shows the result after a user drags the **customer** flat file data object to the mapping editor palette and designates it a read mapping.



5. Optionally, add a reusable object to the mapping.
 You can add a logical data object or other Model repository object.
 - a. Right-click in the mapping editor and select **Add Reusable Object**.
 - b. Select one of the objects from the Model repository, and click **OK**.
 - c. Choose to designate the reusable object a Read mapping, Write mapping, or Lookup mapping.
 - d. If you chose to designate the reusable object a Read mapping, designate the object access of the Read mapping as a related data object or as an independent data object.
 - e. Click **OK**.
6. Optionally, add additional objects and transformations to the mapping, and create links between mapping objects, then click the **Data Viewer** view and run the mapping.
 Results appear in the **Output** section.

CHAPTER 8

Viewing Data

This chapter includes the following topics:

- [Viewing Data Overview, 130](#)
- [Configurations, 131](#)
- [Exporting Data, 139](#)
- [Object Dependencies, 140](#)
- [Logs, 141](#)
- [Validation Preferences, 142](#)
- [Monitoring Jobs from the Developer Tool, 143](#)

Viewing Data Overview

You can run a mapping, view profile results, view source data, preview data for a transformation, run an SQL query, preview web service messages, or view dependencies on an object.

Use the following views to manipulate and view data and mappings:

Mapping Editor view

Use the mapping editor to run mappings and view profile results. 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.

Data Viewer view

Use the **Data Viewer** view to view source data, preview data for a transformation, run an SQL query, or preview web service messages.

Before you can view data, you must select a default Data Integration Service if the domain includes more than one service. You can also add other Data Integration Services to use when you view data. You can create configurations to control settings that the Developer tool applies when you view data.

You can preview data within a mapping that runs on either the Data Integration Service or the Spark engine. You can choose sources and transformations as preview points in a mapping.

The Data Integration Service determines whether to run the job in the native environment or on the Spark engine based on the preview point. The preview point is the object in a mapping that you choose to view data for. For more information about data preview on the Spark engine see the *Data Engineering Integration User Guide*.

Note: The maximum number of rows you can preview in the Data Viewer is 100,000.

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.

Object Dependencies view

Use the **Object Dependencies** view to view object dependencies when you view, modify, or delete Model repository objects.

Configurations

A configuration is a group of settings that the Developer tool applies when you run a mapping, preview data, run an SQL query, or preview web service messages.

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.
- Web service configurations. Controls the settings that the Developer tool applies when you preview the output of a web service in the **Data Viewer** view.

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, mapping, and web service configurations.

The following table displays the properties that you configure for the Data Integration Service:

Property	Description
Use the default Data Integration Service	Uses the default Data Integration Service to run the mapping. Default is enabled.
Data Integration Service	Specifies the Data Integration Service that runs the mapping if you do not use the default Data Integration Service.
Available operating system profiles	Specifies the operating system profile to run the mapping when the Data Integration Service is enabled to use operating system profiles. The Developer tool displays this property only if the administrator assigned at least one operating system profile to the user. The Data Integration Service runs the mapping with the default operating system profile assigned to the user. You can change the operating system profile from the list of available operating system profiles.

Source Properties

The Developer tool displays the **Source** tab for data viewer, mapping, and web service configurations.

The following table displays the properties that you configure for sources:

Property	Description
Read all rows	Reads all rows from the source. Default is enabled.
Read up to how many rows	Specifies the maximum number of rows to read from the source if you do not read all rows. Note: 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.
Read all characters	Reads all characters in a column. Default is disabled.
Read up to how many characters	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. Default is 4000.

Results Properties

The Developer tool displays the **Results** tab for data viewer and web service configurations.

The following table displays the properties that you configure for results in the **Data Viewer** view:

Property	Description
Show all rows	Displays all rows in the Data Viewer view. Default is disabled.
Show up to how many rows	Specifies the maximum number of rows to display if you do not display all rows. Default is 1000.
Show all characters	Displays all characters in a column. Default is disabled.
Show up to how many characters	Specifies the maximum number of characters to display in each column if you do not display all characters. Default is 4000.

Messages Properties

The Developer tool displays the **Messages** tab for web service configurations.

The following table displays the properties that you configure for messages:

Property	Description
Read up to how many characters for request message	Specifies the maximum number of characters to process in the input message.
Show up to how many characters for response message	Specifies the maximum number of characters to display in the output message.

Advanced Properties

The Developer tool displays the **Advanced** tab for data viewer, mapping, and web service configurations.

The following table displays the advanced properties:

Property	Description
Default date time format	Date/time format the Data Integration Services uses when the mapping converts strings to dates. Default is MM/DD/YYYY HH24:MI:SS.
Override tracing level	<p>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.</p> <p>Choose one of the following tracing levels:</p> <ul style="list-style-type: none"> - 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. <p>Default is None.</p>
Sort order	Order in which the Data Integration Service sorts character data in the mapping. Default is Binary.
Optimizer level	<p>Controls the optimization methods that the Data Integration Service applies to a mapping as follows:</p> <p>Auto</p> <p>The Data Integration Service applies optimizations based on the execution mode and mapping contents.</p> <p>None</p> <p>The Data Integration Service does not apply any optimization.</p> <p>Minimal</p> <p>The Data Integration Service applies the early projection optimization method.</p> <p>Normal</p> <p>The Data Integration Service applies the early projection, early selection, branch pruning, push-into, global predicate optimization, and predicate optimization methods.</p> <p>Full</p> <p>The Data Integration Service applies the cost-based, early projection, early selection, branch pruning, predicate, push-into, semi-join, and dataship-join optimization methods.</p> <p>Default is Auto.</p>

Property	Description
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:

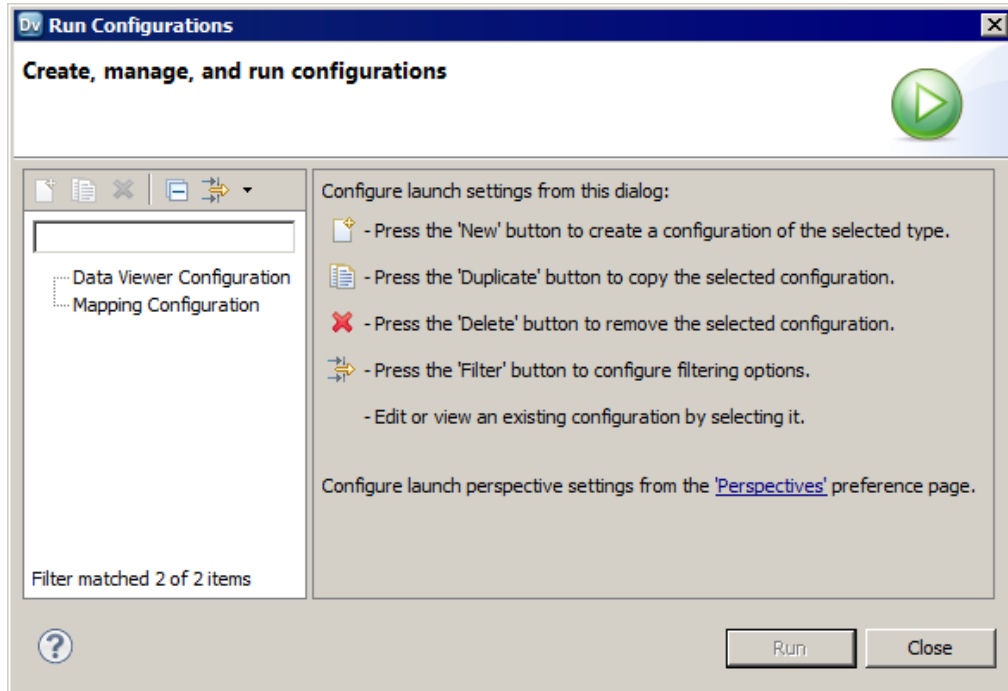
- Custom data objects
- Logical data objects
- Logical data object read mappings
- Physical data objects
- Sources and transformations within mappings
- Virtual stored procedures
- Virtual tables
- Virtual table mappings

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**.

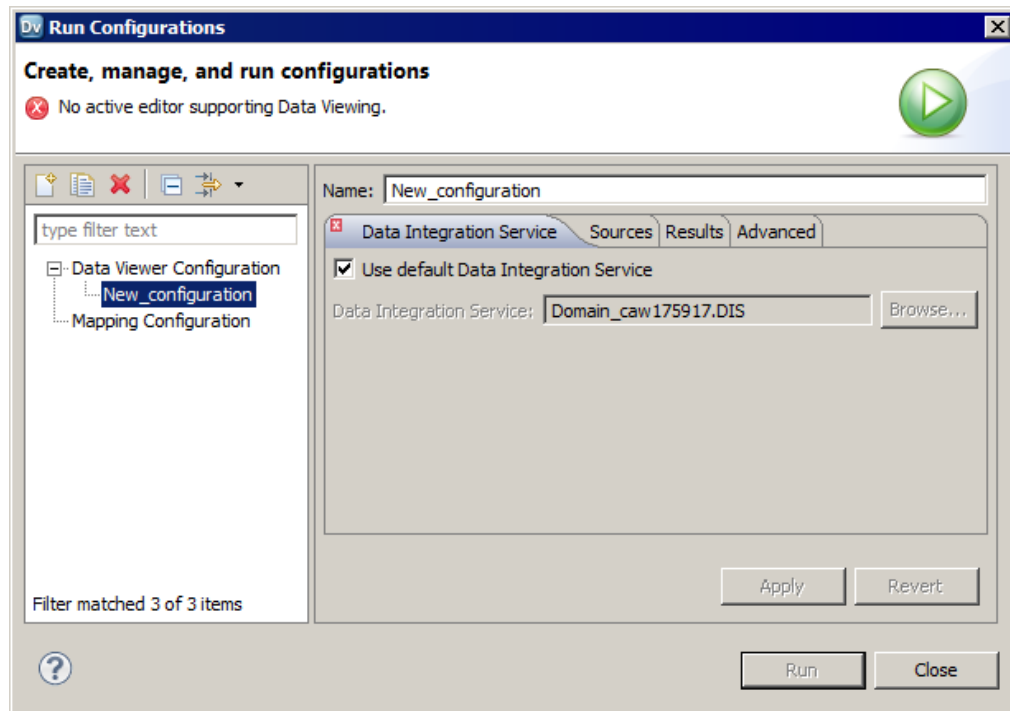
The **Run Configurations** dialog box appears.



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, run the mapping through the **Run Configurations** dialog box or run the mapping using advanced options. If you run the mapping through the **Run** menu or the 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.

For more information about using parameter sets with infacmd, see the *Developer Mapping Guide*.

Creating a Mapping Configuration

Create a mapping configuration to control the mapping deployment properties that the Developer tool uses when you run mappings.

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.

Web Service Configurations

Web Service configurations control the settings that the Developer tool applies when you preview the output of a web service in the **Data Viewer** view.

Create a web service configuration to control the setting that you want to use for specific web services. You can select a web service configuration when you preview the output of an operation mapping or transformations in an operation mapping.

Note: To create a web service configuration that applies to all web services that you preview, use the **Preferences** dialog box to update the default web service configuration.

Creating a Web Service Configuration

Create a web service configuration to control the settings the Developer tool applies when you preview the output of a web service in the **Data Viewer** view.

1. Click **Run > Open Run Dialog**.
The **Run** dialog box appears.
2. Click **Web Service Configuration**.
3. Click **New**.
4. Enter a name for the web service configuration.
5. Configure the web service configuration properties.
6. Click **Apply**.
7. Click **Close**.

Updating the Default Configuration Properties

You can update the default data viewer, mapping, and web service configuration properties.

1. Click **Window > Preferences**.
The **Preferences** dialog box appears.

2. Click **Informatica > Run Configurations**.
3. Select the **Data Viewer, Mapping**, or **Web Service** configuration.
4. Configure the default data viewer, mapping, or web service 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.

[My preview data failed for a customized data object.](#)

The Developer tool cannot display data in the Data Viewer view if the customized data object performs a data type conversion and pushdown optimization is not set to source.

To work around this issue, configure pushdown optimization to push processing to the source.

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. You can export the data of a hierarchical data type to a JSON 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, preview data, or run an SQL query. 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.

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**.

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, preview data, or run an SQL query:

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. 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 objects.

When you monitor a job, you can view summary statistics or execution statistics for the job. The **Summary Statistics** view displays a graphical overview of the status of jobs in the domain.

The **Execution Statistics** view displays general properties and status information for the jobs. For example, you can see who initiated the job and how long it took the job to complete. If you monitor a mapping job, you can also view throughput and resource usage statistics for the job run.

To monitor applications and jobs from the Developer tool, open the **Progress** view and click **View Menu > Monitor Jobs**. Select the Data Integration Service that runs the applications and jobs and click **OK**. The Monitoring tool opens.

CHAPTER 9

Application Deployment

This chapter includes the following topics:

- [Application Deployment Overview, 144](#)
- [Application Creation, 145](#)
- [Application Properties, 146](#)
- [Deployment to an Application, 148](#)
- [Deployment to an Archive File, 150](#)
- [Deployment with Resource Parameters, 150](#)
- [Viewing Run-time Objects, 152](#)
- [Application Redeployment, 152](#)
- [Application State Information, 153](#)
- [How to Create, Deploy, and Update an Application, 154](#)

Application Deployment Overview

An application is a deployable object that can contain physical data objects, logical data objects, data services, mappings, mapplets, transformations, web services, workflows.

To make application objects accessible outside of the Developer tool, deploy an application that contains the objects. You can create a new application, or create the application as you deploy an object. When you deploy an application, you isolate the application objects from changes in data structures.

You can deploy application objects in the following ways:

Deploy to a Data Integration Service.

When you deploy an application to a Data Integration Service, the application runs and end users can connect to the application.

Depending on the types of objects in the application, end users with appropriate permissions can query the objects, access web services, or run mappings or workflows from the command line. You can also deploy objects to allow users to query the objects through a third-party client tool.

Deploy to a network file system.

When you deploy an object to a network file system, the Developer tool creates an application archive file. Deploy an object to an archive file if you want to archive the object in a version control system.

If your organization requires that administrators deploy objects to a Data Integration Service, an administrator can deploy the application archive file to a Data Integration Service using the Administrator tool.

You can also import objects from an application archive file into projects or folders in the Model repository.

After you deploy application objects or an application archive file to a Data Integration Service, you can use the Developer tool to connect to the run-time application and view read-only copies of the run-time objects.

Application Creation

You create an application and deploy it to run the mappings and other objects that the application contains.

You can create an application when you deploy an object to a Data Integration Service, or you can create a new application. When you create a new application, you select objects to include in the application.

You can add objects such as mappings, workflows, data objects, and parameter sets. You cannot add a logical data object directly to an application. However, you can create applications for web services and SQL data services that contain logical data objects.

When you create an application, you can create a full application or an incremental application. A full application is an application that you must redeploy each time that you update the application. An incremental application is an application that you can update by deploying an application patch. A patch can update application objects while other application objects continue running, but the patch might require more time to execute compared to redeploying a full application.

After you create an application, the Developer tool validates the application. The validity of an application depends on the validity of the objects it contains. For example, the configuration of links between objects in a mapping must be valid for the application that contains the mapping to be valid. When an application is not valid, errors appear in the Validation Log view or in an error dialog box.

Full Application

A full application is an application that you can deploy to a Data Integration Service. Each time that you edit the application or the application objects, you must redeploy the application to apply the changes.

Note: During application redeployment, all of the objects that the application contains are inaccessible.

Example

You create one mapping that runs a search function, and another mapping that puts selected search results in a shopping cart. You can create an application to contain the two mappings, and then deploy the application to a Data Integration Service. After testing the output of the application objects, you make changes to the objects. To update the application, you redeploy the application. You also deploy the application to an application archive file, and an administrator checks the archive file into a version control system.

Incremental Application

An incremental application is an application that you can deploy to the Data Integration Service and update by deploying an application patch.

The patch includes the objects that you want to update or add to the deployed incremental application. When you deploy the patch, the Data Integration Service updates the deployed application based on the patch contents.

You can use a patch to increase application up-time. For example, you might have an application that contains a streaming mapping. You can update other objects in the application while the streaming mapping continues processing data.

For more information about application patches, see [Chapter 10, “Application Patch Deployment” on page 162](#).

Note: An incremental application cannot contain web services, SQL data services, or REST web services.

Example

You create one hundred mappings and an incremental application that contains the mappings. Then, you deploy the application to a Data Integration Service. After testing the output of the application objects, you make changes to a few of the objects.

To update the objects in the run-time application, you create an application patch that contains the objects that you changed. You deploy the application patch to the Data Integration Service and the service applies the patch to the incremental application to update the objects without stopping other application objects that other users might be running. You also deploy the application and the application patch to archive files, and an administrator checks the archive files into a version control system.

Application Properties

After you create an application, you can edit properties in the **Application** editor.

General Application Properties

The following table describes general application properties that you can set on the **Overview** tab:

Property	Description
Name	Name of the application.
Description	Optional. Description of the application.
Application Type	Type of application. Full or incremental.

Mapping Deployment Properties

The following table describes the mapping deployment properties that you can set on the **Advanced** tab when the application contains a mapping:

Property	Description
Default date time format	Date/time format that the Data Integration Service uses when the mapping converts strings to dates. Default is MM/DD/YYYY HH24:MI:SS.
Override tracing level	<p>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.</p> <p>Choose one of the following tracing levels:</p> <ul style="list-style-type: none">- None. The Data Integration Service does not override the tracing level that you set for each transformation.- Terse. The Data Integration Service logs information about initialization, error messages, and notification of rejected data.- Normal. The Data Integration Service logs information about initialization and status, errors encountered, and skipped rows due to transformation row errors. The Data Integration Service 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. The log notes where the Data Integration Service truncates string data to fit the precision of a column. The log contains detailed transformation statistics. The Data Integration Service writes row data for all rows in a block when it processes a transformation. <p>Note: To view Spark engine information in the workflow log, select Verbose Initialization or Verbose Data in the Developer tool.</p> <p>Default is None.</p>
Sort order	Order in which the Data Integration Service sorts character data in the mapping. Default is Binary.

Property	Description
Optimizer level	<p>Controls the optimization methods that the Data Integration Service applies to a mapping as follows:</p> <p>Auto</p> <p>The Data Integration Service applies optimizations based on the execution mode and mapping contents.</p> <p>None</p> <p>The Data Integration Service does not apply any optimization.</p> <p>Minimal</p> <p>The Data Integration Service applies the early projection optimization method.</p> <p>Normal</p> <p>The Data Integration Service applies the early projection, early selection, branch pruning, push-into, global predicate optimization, and predicate optimization methods.</p> <p>Full</p> <p>The Data Integration Service applies the cost-based, early projection, early selection, branch pruning, predicate, push-into, semi-join, and dataship-join optimization methods.</p> <p>Default is Auto.</p>
High precision	<p>Runs the mapping with high precision.</p> <p>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.</p> <p>Default is enabled.</p>

Deployment to an Application

You can deploy an object as an application or as a data service that is part of an application.

You can deploy some objects as a web service or as an SQL data service. First, you create the application and add the objects. Then, when you deploy the application, the Developer tool prompts you to create a service based on the object. The Developer tool adds the service to the application.

You can also deploy objects to allow users to query the objects through a third-party client tool.

You can deploy an application to a Data Integration Service to allow users to access the mappings, workflows, and other objects that the application contains.

When you deploy an application, the Data Integration Service runs application objects.

The Developer tool prompts you to create an application when you deploy mappings or workflows.

You can add parameter sets to applications that you deploy. A parameter set is an object in the Model repository that contains parameters and parameter values to use with mappings and workflows. Deploy a parameter set with the workflow or mapping to use the parameter values when the Data Integration Service

runs the workflow or mapping. You can add multiple parameter sets to an application and then use different parameter sets for mapping or workflow runs.

When you deploy an application of the same name to the same Data Integration Service, the Data Integration Service overwrites the deployed application and all of the objects in the application.

Deploy Objects as a Web Service

You can deploy the following objects as a web service:

- Mapplets
- Transformations except the Web Service Consumer transformation
- Flat file data objects
- Relational data objects
- Logical data objects

When you deploy an object as a web service, the Developer tool prompts you to create a web service based on the object, and then prompts you to create an application to contain the web service.

When you deploy an object as a web service, you enter the following information:

Property	Description
Name	Name of the web service.
Location	Model repository project folder where you want to put the application.
Namespace	URL where you want users to access the web service. The Data Integration Service overrides the URL defined in the WSDL file.
Prefix	The prefix of the target namespace.

Deploy Objects as an SQL Data Service

You can deploy the following data objects as an SQL data service:

- Physical data objects
- Logical data objects

When you deploy a data object as an SQL data service, the Developer tool prompts you to create an SQL data service based on the object, and then prompts you to create an application to contain the service.

When you deploy an object as an SQL data service, you enter the following information:

Property	Description
Name	Name of the web service.
Location	Model repository project folder where you want to put the application.

Deployment to an Archive File

An archive file stores application objects and metadata in an XML format.

You can create the following types of archive files:

Application archive file

An application archive file is an XML file that stores all of the information about application objects and metadata for an application. The file has an .iar extension.

Application patch archive file

An application patch archive file is an XML file that stores information about an application patch. The file stores information about the application objects in the patch, the patch description, the name of the corresponding incremental application, and whether the patch retains or discards state information. The file has a .piar extension.

You might want to create an archive file for any of the following reasons:

- Deploy the application or application patch. If your organization restricts the ability to deploy applications or application patches to a Data Integration Service to administrators, an administrator can deploy an archive file to the Data Integration Service.

To deploy an application, the administrator can use the Administrator tool, or `infacmd deployApplication`. To deploy an application patch, the administrator can use `infacmd tools patchApplication`.

- Import the application or application patch to a Model repository. An administrator can import an archive file to a Model repository using `infacmd tools importObjects`.

If you import an archive file that contains objects that are already in the Model repository, the Developer tool creates a new project folder.

Note: The release version of the Model repository where you import the application must be compatible with the version from which you exported the application to an archive file.

- Archive the file in another system. For example, if the Model repository is not integrated with a version control system, an administrator can check the archive file into a version control system for project auditing or object recovery.

Deployment with Resource Parameters

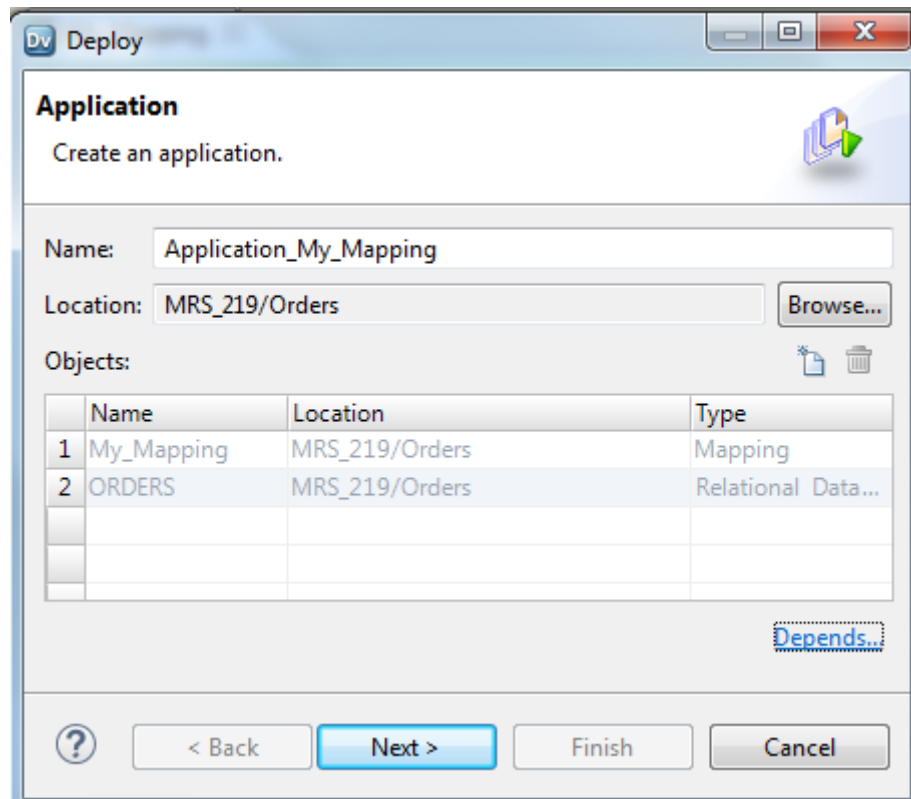
When you deploy a mapping that has resource parameters in the source, the lookup, or the target transformation, you can view a list of the data objects that the application requires to run the mapping.

You can check the data objects in the resource parameter values when you create an application or when you deploy a mapping. To view the data objects, click the **Depends** link in the **Create Application** dialog box or in the **Deploy** dialog box. When you view the data objects, you can choose to exclude a data object from the application. You change the value of the resource parameter to a different data object at run time.

To exclude a data object from the application, clear the Include/Exclude checkbox.

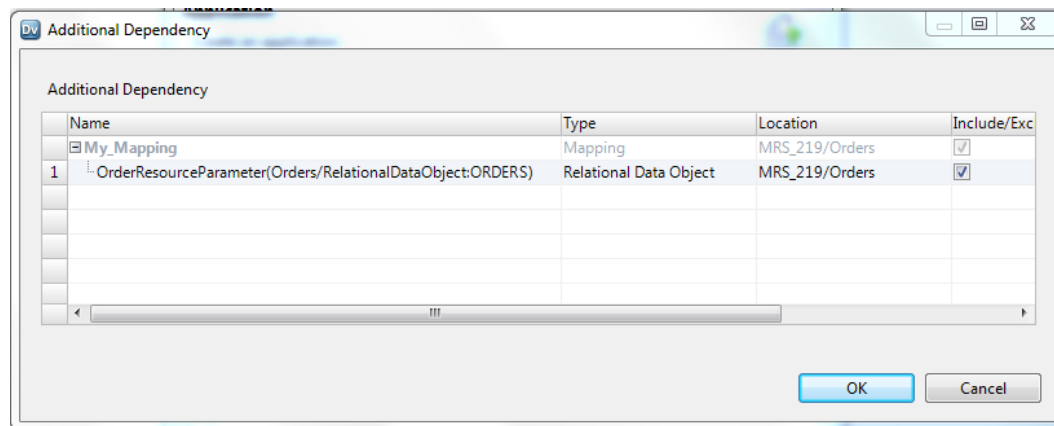
A mapping can include multiple mapplets and a mapplet can include other mapplets. The dependencies list does not show a resource parameter value if another parameter overrides it in the parameter hierarchy.

The following image shows the **Depends** link in the **Deploy** dialog box:



When you click the **Depends** link, the **Additional Dependency** dialog box appears. The Developer tool shows a list of the data objects that the application requires to run the mapping.

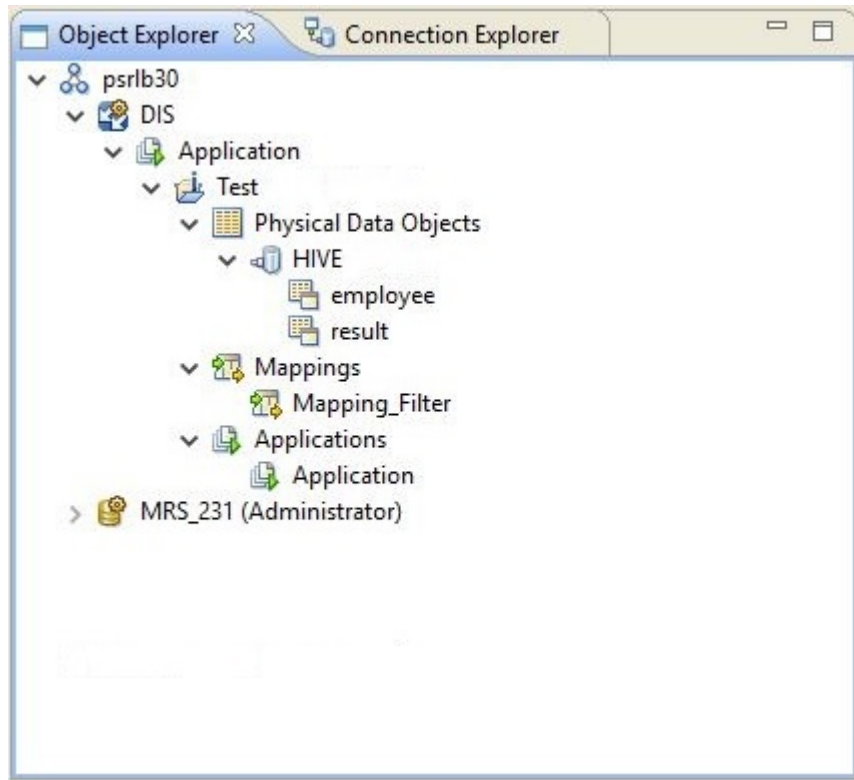
The following image shows the **Additional Dependency** dialog box:



Viewing Run-time Objects

Connect to a run-time application to view read-only copies of run-time objects in a deployed application.

The following image shows a run-time application in the **Object Explorer** view:



You can open the run-time objects in the **Editor** to verify that the application or application patch was deployed correctly. After you redeploy an application or deploy an application patch, you can refresh the run-time application to view the updated objects.

You can also reference the run-time objects while you are editing the corresponding design-time objects based on the results of testing or new release requirements. Because a run-time object is isolated from changes in data structures in the corresponding design-time object, you can continue editing the design-time object without altering the run-time object.

Application Redeployment

After you change the contents of an application, you might want to redeploy the application to apply the changes.

You might redeploy the application for the following reasons:

- You edit, add, or delete application objects in a full application and you want to apply the changes to the deployed full application.
- You delete application objects in an incremental application and you want to apply the changes to the deployed incremental application.

- An application patch for an incremental application contains a large number of objects and it is inefficient for the user to deploy the patch. Deploying a patch might be inefficient if the patch contains a large number of objects due to patch inheritance or there is a large number of affected objects in the run-time application.

When you deploy an edited application, you choose to retain or discard application state information. State information refers to mapping properties, mapping outputs, and the properties of run-time objects.

If you want to redeploy an application, you must choose to stop 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 update an application and export it to a network file system, you can 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.

Application State Information

When you update a deployed application by redeploying the application or by deploying an application patch, you can choose to retain the state information for the deployed application or to discard it.

State information refers to mapping properties and the properties of run-time objects such as mapping outputs or the Sequence Generator transformation. When you retain state information, you retain these settings and properties in the deployed application. When you discard state information, you discard the state of these settings and properties in the deployed application.

Example: Retaining or discarding configurable properties

An application includes a mapping with configurable run-time properties. You set the High Precision property to True. After you deploy the application, you edit the mapping and change the High Precision property to False. When you redeploy the application and check **Retain state information**, the Data Integration Service retains state information in the deployed application and does not recognize changes to the run-time properties. You must uncheck **Retain state information** for the change to the property to take effect.

Example: Retaining or discarding sequences

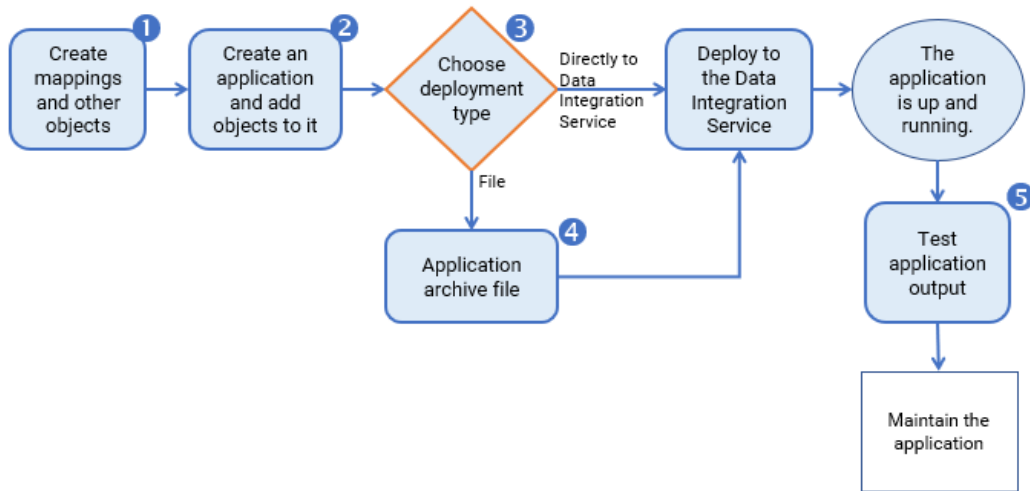
A mapping includes a Sequence Generator transformation that generates unique keys for rows in a target table. After you deploy the application that contains the mapping, the mapping runs and the Sequence Generator transformation generates keys for rows 1-2000. The next time the mapping runs, the Sequence Generator transformation would generate keys beginning with 2001, but you decide you want the sequence to start at 10000. Edit the Sequence Generator transformation to specify the new start value. When you redeploy the application and check **Retain state information**, the Data Integration Service retains state information in the deployed application and does not recognize changes to the setting. When the mapping runs again, the sequence resumes at 2001. You must uncheck **Retain state information** for the change to the setting to take effect.

Note: Your choice to retain or discard application state has no effect on the Reset property of a Sequence Generator transformation.

How to Create, Deploy, and Update an Application

Create and edit mappings and other objects and deploy them in an application to make them accessible to end users.

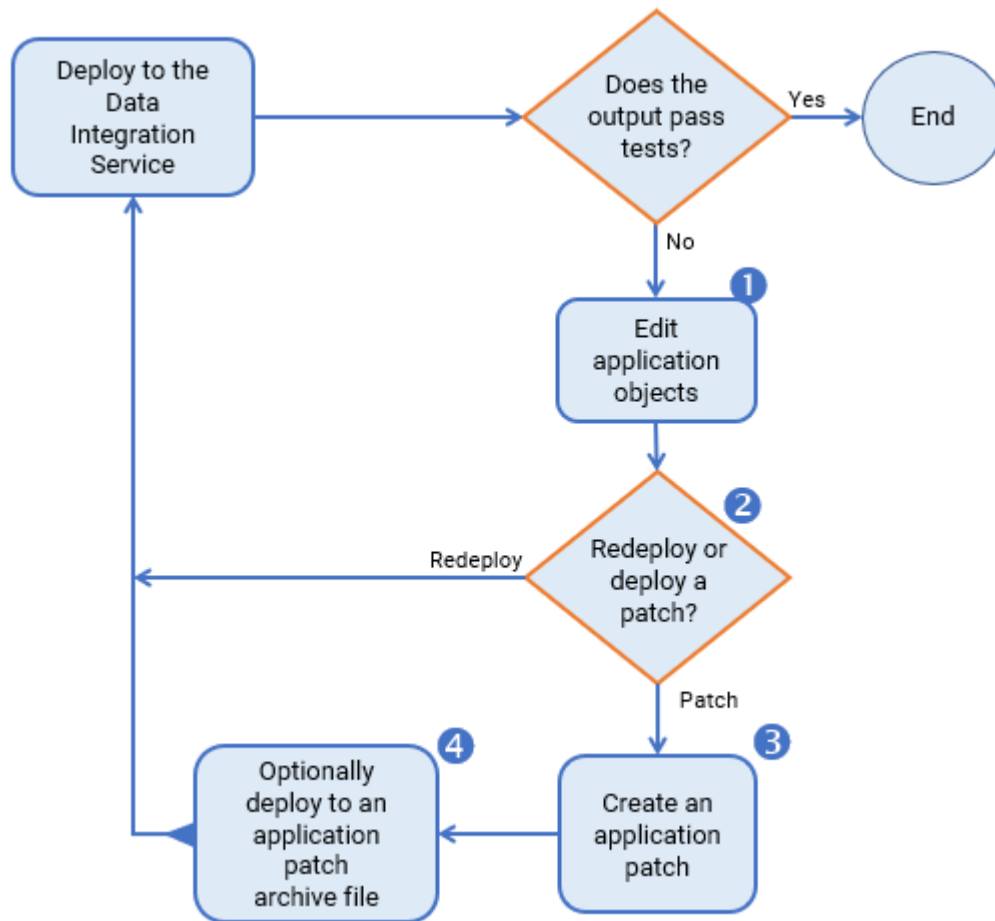
The following image shows the process of developing and deploying an application:



1. Create mappings, workflows, transformations, and other objects to access and transform data.
2. Create an application and add objects to the application.
3. Choose to deploy the application directly to a Data Integration Service or to an application archive file:
 - Deploy the application directly to a Data Integration Service to allow the Data Integration Service to run objects.
 - Deploy the application to an application archive file to allow an administrator to archive the file.
4. Deploy the archive file to a Data Integration Service.
5. Test the application to ensure that it meets organizational requirements.

After you test the application, you can maintain the application by editing application objects and updating the deployed application. To help you edit the application objects, you can connect to the run-time application to review how the run-time objects are designed.

The following image shows how an application is maintained:



1. If the output does not meet requirements, edit the objects in the application.
2. Choose to update the application by redeploying the application or by deploying an application patch. You can deploy an application patch only if the application is an incremental application.
3. If you choose to deploy a patch, create the patch using the Incremental Deployment Wizard.
4. Deploy the patch to the Data Integration Service, optionally deploying the application patch to an archive file.

For information on how to create and deploy an application patch, see the "Application Patch Deployment" chapter.

Note: An administrator can also deploy an application and perform other administrative tasks from the command line or from the Administrator tool. For information about these tasks, see the *Informatica Application Service Guide*.

Creating an Application

When you create an application, you select the objects to include in the application. Create an application when you want to deploy one or more objects so end users can access the data through third-party tools.

1. Select a project or folder in the **Object Explorer** view.
2. Click **File > New > Application**.

The **New Application** dialog box appears.

dv New Application

Application
Create an application.

Name:

Location: [Browse...](#)

Application Type: Full Incremental

Objects:

	Name	Location	Type
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			

[Depends...](#)

[Finish](#) [Cancel](#)

3. Enter a name for the application.

The application name is part of the session log file name when you deploy the application. Depending on the application, the session log file name might include the application name, feature name, connection ID, and the date and time.

The session log file name is included with the log directory path in the log file path. On Windows operating systems, the log file path has a limit of 259 characters. Consider the length limitation when you name the application.

4. Click **Browse** to select the application location.
You must create the application in a project or a folder.
5. Select the application type. You can select Full or Incremental.
After you create the application, you cannot edit the application type.
6. Click the Add button to add objects to the application.
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. Click **Finish** to create the application.

After you create an application, you can deploy it to a Data Integration Service or to an application archive file.

Deploying an Application

After you create an application, deploy the application to a Data Integration Service to run application objects, or deploy it to an application archive file.

1. Right-click an application in the **Object Explorer** view and select **Deploy**.

The **Deploy** dialog box appears.

Deploy

Deployment Method
Choose how to deploy the application.

Application: MyApplication (Incremental)

Deploy to Service

Domain:

Available Services:

<input checked="" type="checkbox"/>	Service	Target Name	Action
<input checked="" type="checkbox"/>	DIS	MyApplication (Running)	Add
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			

Export as application archive file

Location:

2. Choose one of the following options to deploy the application:

- To deploy the application to a Data Integration Service, select **Deploy to Service**.
 - Optionally, click **Browse** to select a domain.
 - Select the Data Integration Services to which you want to deploy the application.
- To deploy the application to an application archive file, select **Export as application archive file**. Then click the **Browse** button to select the directory where you want to save the file.

If you deploy the application to a Data Integration Service, the Data Integration Service runs the application and the objects that it contains. You can test the output of application objects and validate the output against requirements. If necessary, edit the application objects and update the deployed application.

If you deploy the application to an application archive file, you can store the file, make it available for import to another Model repository, or deploy the application from the file to a Data Integration Service at a later time.

Deploying an Object

Deploy an object to a Data Integration Service as an application, or deploy the object to an application archive file to export the object as an application.

1. Choose to deploy an executable object or a flat file data object.
 - To deploy a mapping, workflow, or other executable object, right-click the object and select **Deploy**.
 - To deploy a flat file data object, right-click the object and choose to deploy the object as a web service, SQL data service, or REST web service.

The Developer tool prompts you to create an application.

2. Enter an application name.
3. To choose a location for the application, accept the default location, or click **Browse** and select another location.
4. Select the application type. You can select Full or Incremental.

If you deploy a flat file data object as a web service, SQL data service, or REST web service, the application cannot be an incremental application.

After you create the application, you cannot edit the application type.

5. To add objects to the application, click the Add button and select objects.
6. Click **Next**.
7. Choose one of the following options to deploy the application:
 - To deploy the application to a Data Integration Service, select **Deploy to Service**.
 - If the Developer tool contains the connection information for multiple domains, click **Browse** to select a domain.
 - Select the Data Integration Services to which you want to deploy the application.
 - To deploy the application to an application archive file, select **Export as application archive file**. Then click the **Browse** button to select the directory where you want to save the file.
8. To deploy the object to a web service, complete the following steps:
 - a. Configure properties for the web service.
 - b. To add operations to the web service, click **Next**.

By default, the Developer tool creates an operation for each object that you deploy as a web service.
 - c. Select each operation, operation input, and operation output to display and configure the properties.
 - d. Click **Finish**.
9. To deploy the object to an SQL data service, complete the following steps:
 - a. Enter a name for the SQL data service.
 - b. Accept the default location, or click **Browse** to select a Model repository and a project folder location for the SQL data service.
 - c. Click **Next**.

The **Add Virtual Tables to SQL Data Service** dialog box appears.
 - d. Click the Add button.
 - e. Enter a name for the virtual table.
 - f. Click the Open button in the **Data Object** column.

The **Select a Data Object** dialog box appears.

- g. Select a physical data object and click **OK**.
 - h. Enter the virtual schema name in the **Virtual Schema** column.
 - i. Select **Read** in the Data Access column to link the virtual table with the data object. Select **None** if you do not want to link the virtual table with the data object.
10. To deploy an object to a REST web service, complete the following steps:
 - a. Configure properties for the REST web service.
 - b. Select the columns to include in the definition.
 - c. Select the HTTP method.
11. Click **Finish**.

The Developer tool deploys the application to the Data Integration Service. The Data Integration Service runs the application and the objects that it contains.

You can test the output of application objects and validate the output against requirements. If necessary, edit the application objects and update the deployed application.

Importing Application Archives

Import an application or an application patch that was deployed to an archive file to make the file contents available in the Model repository.

If you import a file that was archived from a different Model repository or was retrieved from a version control system, you can import the application and the application objects that were used to create the archive file.

1. Click **File > Import**.
The **Import** wizard appears.
2. Select **Informatica > Application Archive**.
3. Click **Next**.
4. Click **Browse** to select the archive file. To import an application archive file, use the .iar extension. To import an application patch archive file, use the .piar extension.
The Developer tool lists the archive file contents.
5. Select the Model repository into which you want to import the application.
6. Click **Finish**.

The Developer tool imports the application into the repository.

Connecting to a Run-time Application

Connect to a run-time application so that you can use the **Editor** to view the run-time objects in the application.

1. Click **File > Connect to Run-time Application**.
2. Select a domain.
3. Select a Data Integration Service.
4. Select a deployed application on the Data Integration Service.
5. Click **OK**.

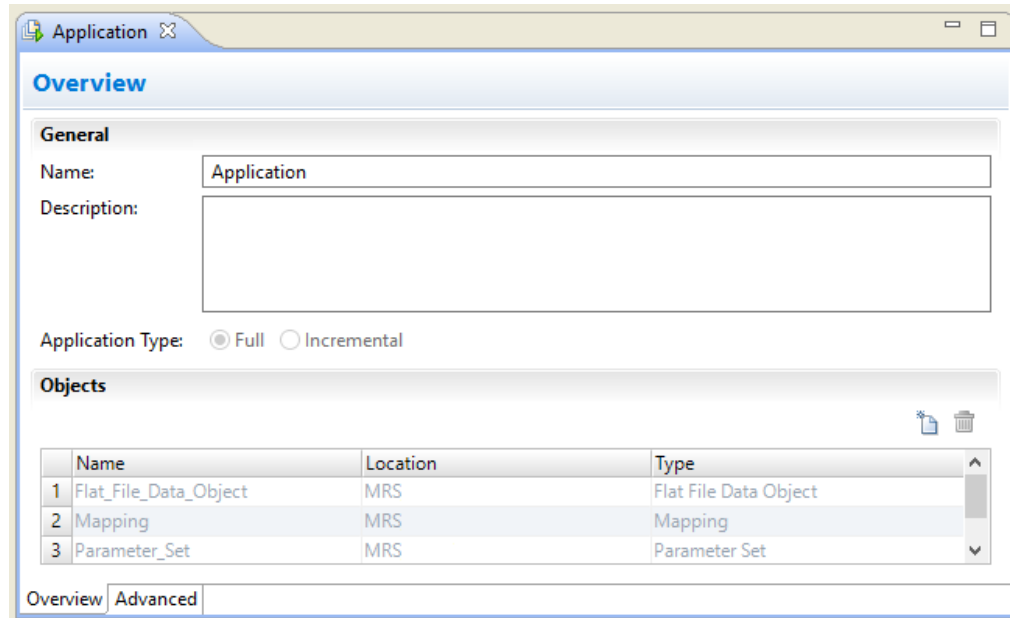
The run-time application appears in the **Object Explorer** view. You can expand the application to view the run-time objects in the application.

Editing an Application

Edit application objects, or edit the application to add and delete objects, and edit application properties.

1. To edit an application object, open the object from the **Object Explorer** view. Make the required changes.
2. To edit the application, right-click the application in the **Object Explorer** and select **Open**.

The **Application Editor** appears.



3. Click the Add button to add objects to the application.
The **Add Objects** dialog box appears.
4. Select one or more objects and click **OK**.
5. Select one or more objects to delete and click the Delete button.
6. Save the application.

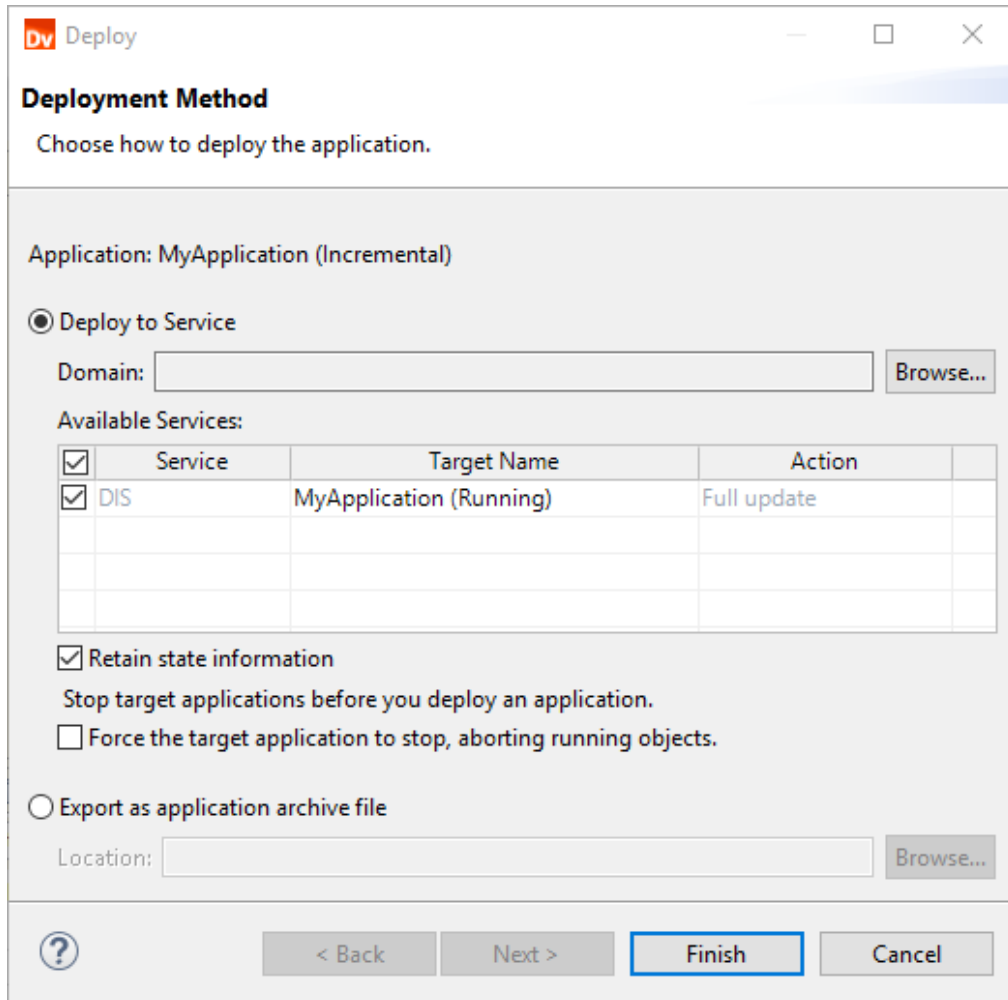
After you edit the application, update the application to propagate the changes to the deployed application.

Redeploying an Application to a Data Integration Service

Redeploy an application to a Data Integration Service to replace the run-time application on the Data Integration Service with the design-time 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. If no default Data Integration Service is set, click **Browse** to select the domain.
The **Choose Domain** dialog box appears.
4. Select a domain and click **OK**, and then select a Data Integration Service.

The Target Name column displays the application with the same name by default. For example, the following image shows the deployment of the application MyApplication:



5. Click the Target Name column to choose a different target application on the Data Integration Service.
6. To retain the state of run-time objects that are part of the application, select **Retain state information**.
7. Optionally, select **Force target application to stop, aborting running objects**.

You can redeploy an application without stopping it only if the state of the application is not Running. If the administrator has not stopped the application, verify with the administrator whether you can choose to force the application to stop.

8. Click **Finish**.

After you redeploy the application, you can validate it against organizational or project requirements.

CHAPTER 10

Application Patch Deployment

This chapter includes the following topics:

- [Application Patch Deployment Overview, 162](#)
- [Application Object Instances, 163](#)
- [Application Object Dependencies, 163](#)
- [Incremental Deployment Wizard, 166](#)
- [Choose the Application Patch Type, 170](#)
- [Determine Application Object Impact, 176](#)
- [Specify a Patch Deployment Strategy, 180](#)
- [Deploying an Application Patch, 182](#)
- [Rules and Guidelines for Application Patch Deployment, 184](#)

Application Patch Deployment Overview

An application patch is a deployable object that you can use to update the objects in a deployed incremental application.

To design an application patch, you must consider the design-time and run-time instances of the application objects and the dependencies between the application objects in order to decide the type of application patch that you want to create. Some application objects might be related to the other objects that you want to update, and other application objects might be affected by the objects that you want to update.

You can use a patch type to choose how to inherit the related objects and determine how much impact you want the patch to have on the deployed application. To manage patch inheritance and object impact, you can use the tools in the Incremental Deployment Wizard to understand the contents of a patch and how the Data Integration Service will apply the contents to the deployed application.

Note: You cannot use an application patch to perform the following tasks:

- Update application and mapping deployment properties, such as the application description, the default date time format, or the sort order.
- Add web services, SQL data services, or REST web services to the deployed application.
- Delete application objects from the deployed application.

Application Object Instances

The application objects in an application have design-time instances and run-time instances.

A design-time instance of an application object refers to the application object that developers create and edit in the Developer tool. When you deploy an application that contains the object to the Data Integration Service, you generate a run-time instance of the application object. The run-time instance of the object is isolated from changes in data structures in the design-time instance, so you can continue editing the design-time object in the Developer tool without altering the run-time object.

The Incremental Deployment Wizard uses both the design-time and run-time instances of application objects to decide how to apply patch contents to a deployed application. For example, the wizard decides how the patch inherits other objects based on the set of dependencies of the design-time and run-time instances of a selected object, and it fetches run-time instances of application objects to determine how they are impacted.

Application Object Dependencies

An application object that you select to include in an application patch might be related to other application objects based on the selected object's set of dependencies.

Objects within an application can contain the following dependencies:

Direct

A direct dependency is an application object that the selected object uses and any direct dependencies of the direct dependency.

The following table lists the direct dependencies for each object type:

Object	Possible Direct Dependencies
Workflow	<ul style="list-style-type: none">- Mapping- Mapplet- Reusable transformation- Logical data object- Physical data object
Mapping	<ul style="list-style-type: none">- Mapplet- Reusable transformation- Logical data object- Physical data object
Mapplet	<ul style="list-style-type: none">- Reusable transformation- Logical data object- Physical data object
Reusable Transformation	<ul style="list-style-type: none">- Logical data object- Physical data object
Logical Data Object	<ul style="list-style-type: none">- Physical data object

Object	Possible Direct Dependencies
Physical Data Object	- A physical data object cannot have a direct dependency.
Parameter Set	- A parameter set cannot have a direct dependency or be a direct dependency.

Indirect

An indirect dependency is an application object that uses the selected object, or an application object that uses a direct dependency of the selected object. Indirect dependencies include any direct dependencies of the indirect dependency.

Remote

A remote dependency is an application object that is an indirect dependency of an indirect dependency for the selected object. Remote dependencies include any direct dependencies of the remote dependency.

An application patch inherits other application objects based on a selected object's dependencies. For example, you can create an application patch that inherits a selected object's direct and indirect dependencies.

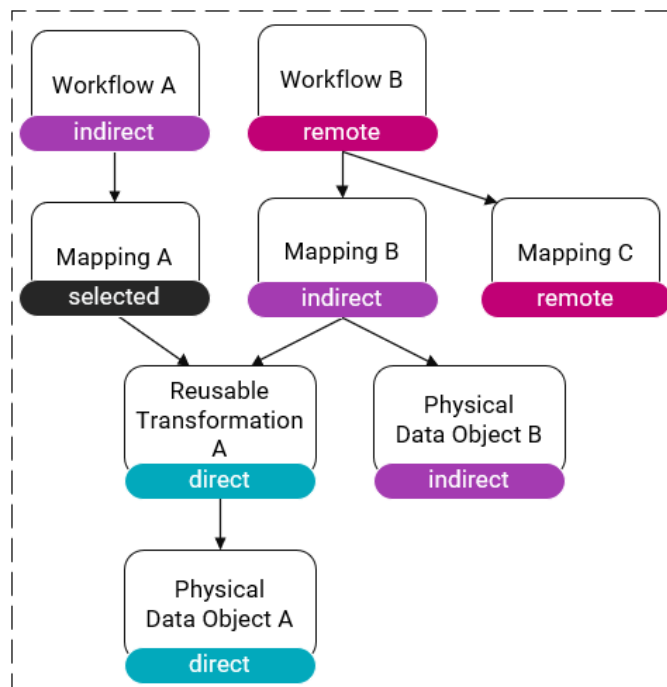
You cannot select mapplets, reusable transformations, and logical data objects in an application patch, but the patch can inherit these objects.

For more information about identifying indirect and remote dependencies, see the following Informatica How-To Library article:

[Identifying Indirect and Remote Dependencies for an Application Patch in the Developer Tool](#)

Example. Dependency Sets

The following image shows the set of dependencies for an application object `Mapping A`:



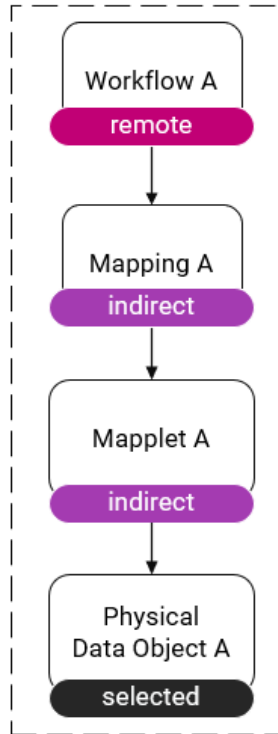
The following table describes the set of dependencies for the mapping `Mapping A`:

Dependency Type	Object	Description
Direct	Reusable Transformation A	The reusable transformation is a direct dependency because the mapping <code>Mapping A</code> uses the reusable transformation.
Direct	Physical Data Object A	The physical data object is a direct dependency because the mapping <code>Mapping A</code> ultimately uses the data object through the reusable transformation <code>Reusable Transformation A</code> .
Indirect	Workflow A	The workflow is an indirect dependency because the workflow uses the mapping <code>Mapping A</code> .
Indirect	Mapping B	The mapping <code>Mapping B</code> is an indirect dependency because the mapping <code>Mapping B</code> uses a direct dependency of the mapping <code>Mapping A</code> : the reusable transformation <code>Reusable Transformation A</code> .
Indirect	Physical Data Object B	The physical data object is an indirect dependency because it is a direct dependency for the mapping <code>Mapping B</code> which is an indirect dependency for the mapping <code>Mapping A</code> .
Remote	Workflow B	The workflow is a remote dependency because it is an indirect dependency for the mapping <code>Mapping B</code> which is an indirect dependency for the mapping <code>Mapping A</code> .
Remote	Mapping C	The mapping <code>Mapping C</code> is a remote dependency because it is a direct dependency for the workflow <code>Workflow B</code> which is a remote dependency for the mapping <code>Mapping A</code> .

Relationship Priority

Relationships between physical data objects, mappings, and workflows take priority over relationships between other types of objects.

For example, the following image shows a set of dependencies for a data object `Physical Data Object A`:



The mapping `Mapping A` is an indirect dependency because the relationship between the mapping `Mapping A` and the data object `Physical Data Object A` takes priority over the following relationships:

- The relationship between the data object `Physical Data Object A` and the mapplet `Mapplet A`.
- The relationship between the mapplet `Mapplet A` and the mapping `Mapping A`.

Since the mapping `Mapping A` is an indirect dependency, its direct dependency `Mapplet A` is an indirect dependency for the data object `Physical Data Object A`.

Incremental Deployment Wizard

The Incremental Deployment Wizard is the wizard that you use to create an application patch.

To run the wizard, right-click an incremental application and select **Update Application Objects**.

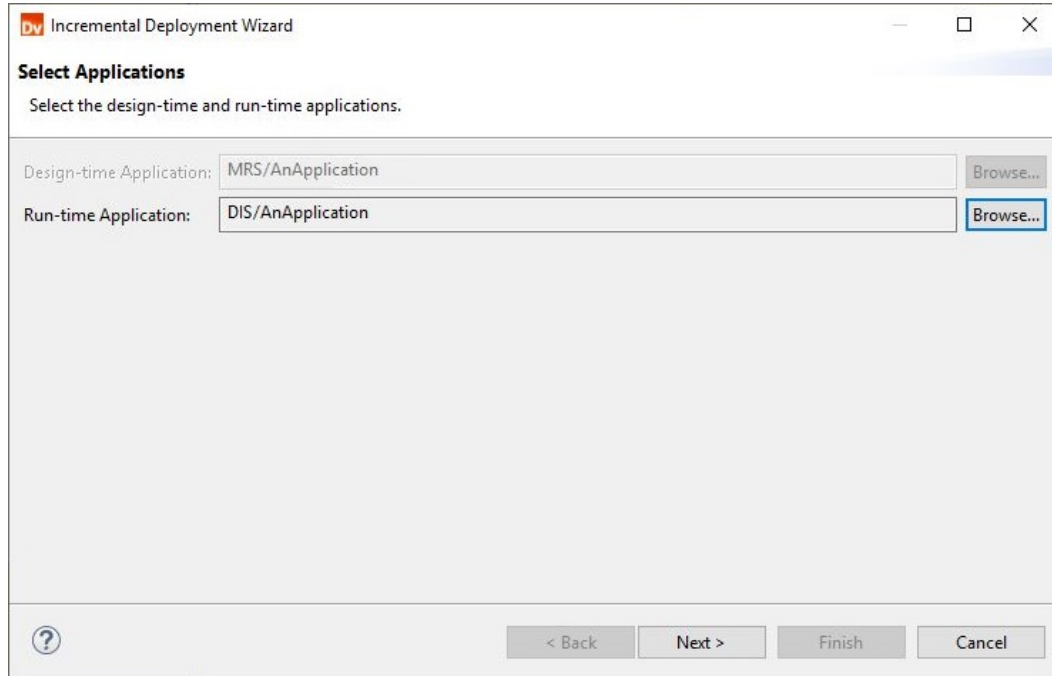
You use the following pages in the wizard to create an application patch:

1. On the **Select Applications** page, you select the run-time application that you want to update.
2. On the **Source Object Selection** page, you choose the objects that you want to include in the patch. You can also preview the type and amount of impact that the patch will have on the run-time application.
3. On the **Patch Deployment Strategy** page, you define additional details about the patch and decide how to deploy the patch.

Select Applications

On the **Select Applications** page, select a run-time application that you want to update.

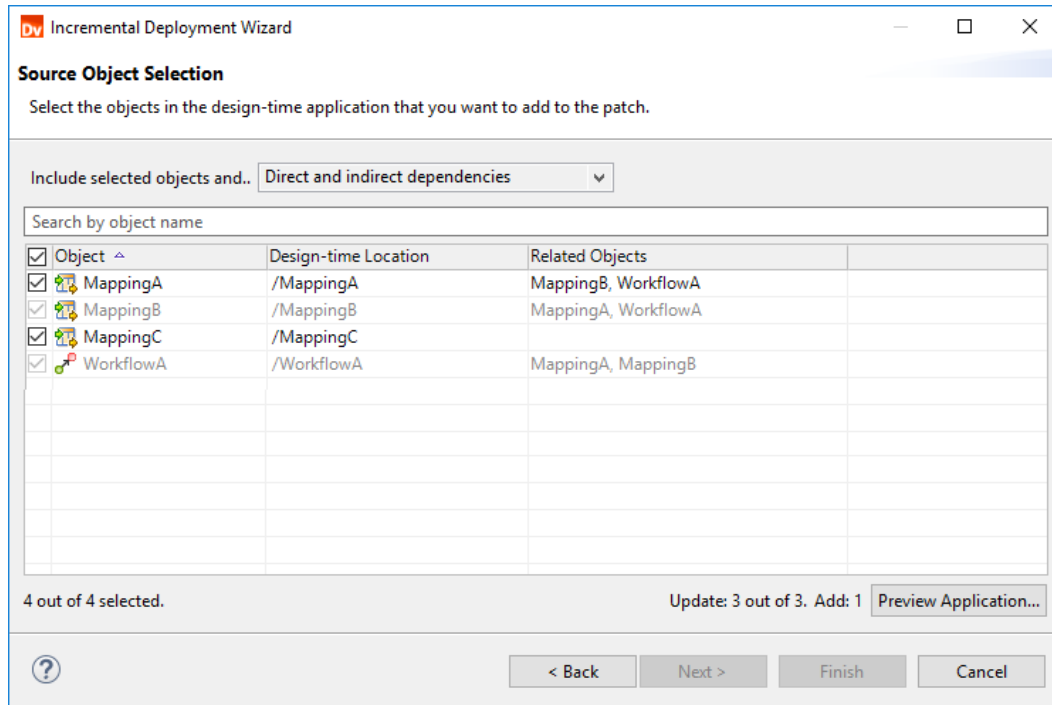
The following image shows the **Select Applications** page:



Source Object Selection

On the **Source Object Selection** page, choose the type of patch that you want to create and select the objects that you want to include in the patch.

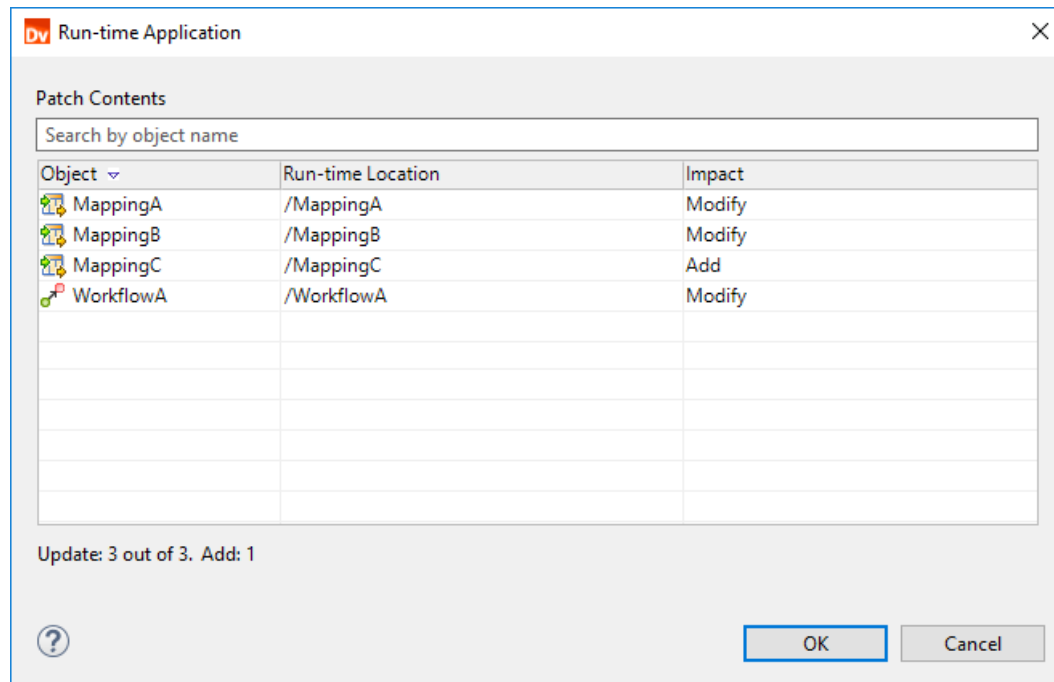
The following image shows the **Source Object Selection** page:



For each object, you can use the Design-time Location column to view the path to the object in the design-time application. The wizard also lists the objects' dependencies in the Related Objects column based on the patch type. For example, for a patch type that inherits direct and indirect dependencies, the wizard lists each object's direct and indirect dependencies.

In the bottom-right corner of the page, you can preview a summary of the impact on the run-time objects. The Update statistic shows the total number of objects that will be modified or affected. The Add statistic shows the total number of objects that will be added. To view the impact on a particular run-time object, you can preview the application.

The following image shows the application preview:



You can match the list of run-time objects in the preview to the design-time objects by matching the object names and matching the run-time location to the design-time location. For each object, the impact lists whether the object will be modified, affected, added, or unchanged.

Patch Deployment Strategy

On the **Patch Deployment Strategy** page, describe the patch, define how you want to deploy it, and choose whether the patch retains state information.

The following image shows the **Patch Deployment Strategy** page:

The screenshot shows a window titled "Incremental Deployment Wizard" with a sub-header "Patch Deployment Strategy". Below the sub-header is the instruction "Select the deployment strategy for the patch." The form contains the following fields and controls:

- Patch Name:** A text box containing "Patch2".
- Patch Description:** A text box containing "This is another patch."
- Retain State Information:** A checked checkbox.
- Deploy:** A group box containing two radio buttons: "Deploy to Data Integration Service" (selected) and "Deploy to File" (unselected). A "Browse..." button is located to the right of the "Deploy to File" radio button.
- Patch History:** A table with two columns: "Name" and "Description". The table contains one row with the following data:

	Name	Description
1	Patch1	24-Sep-2019 11:12:03 This is a patch.

At the bottom of the window, there are four buttons: a help icon (?), "< Back", "Next >", and "Finish" (highlighted with a blue border). A "Cancel" button is also present.

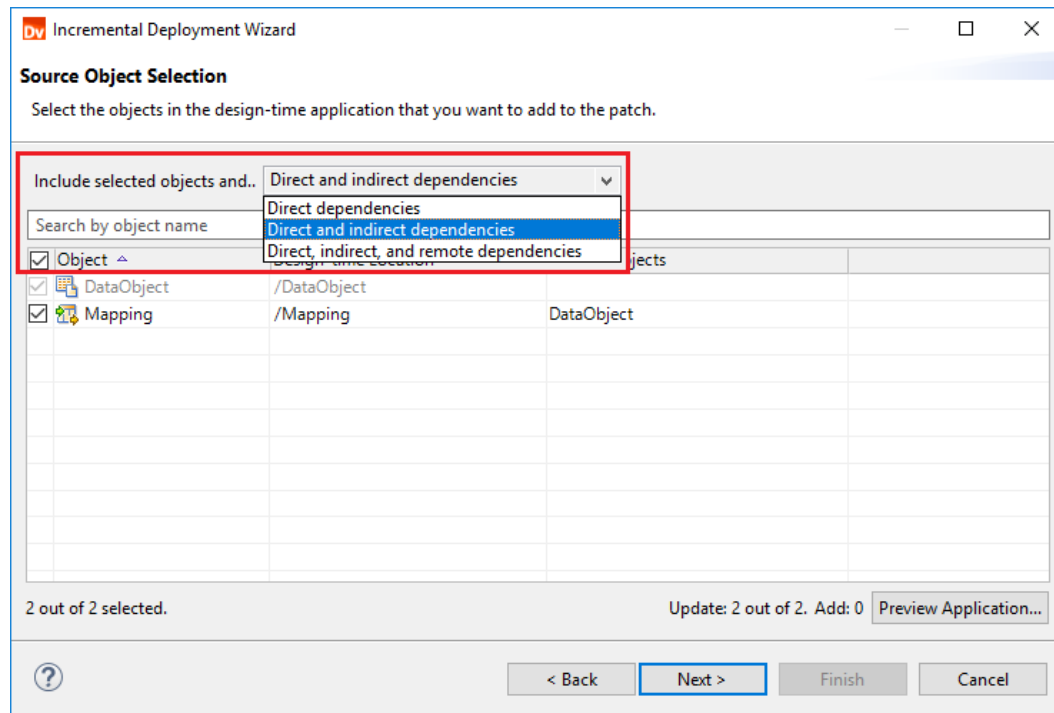
When you choose a deployment strategy, you can also use the patch history to review the patches that have been deployed to update the run-time application.

Choose the Application Patch Type

When you use the Incremental Deployment Wizard, you can choose the type of application patch that you want to create. The application patch type determines the level of dependencies that the patch inherits.

A patch can inherit direct, indirect, and remote dependencies. With each level of dependencies, the patch inherits more objects. The inherited objects are included in the patch, and the overall impact that the patch has on the run-time application increases. You can choose the dependencies to inherit depending on the application design and the application objects that you want to update.

The following image shows the list of patch types in the wizard:

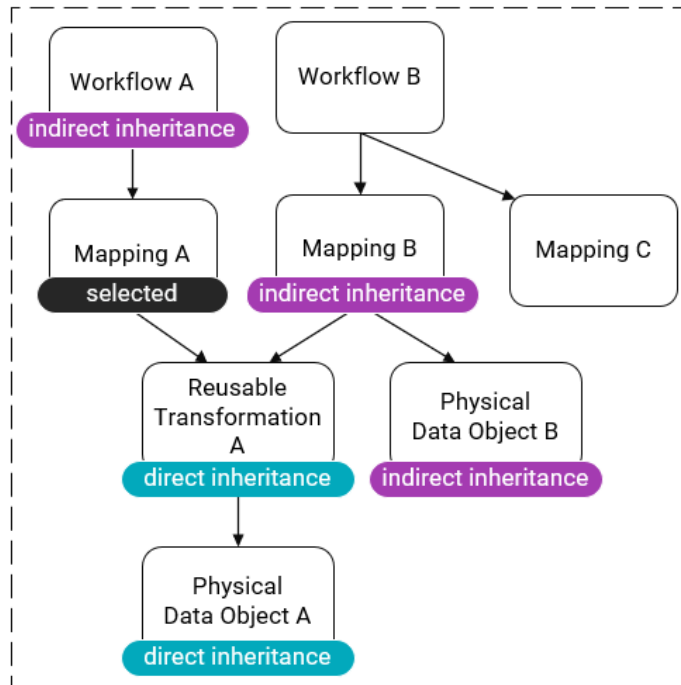


When you deploy the patch, the Data Integration Service replaces the run-time instances of the selected objects and the inherited dependencies in the run-time application, and it might affect other run-time objects. The Data Integration Service updates the affected objects to use the replaced run-time objects, but the affected objects are not replaced with their design-time instances.

If the application contains highly-dependent objects, the dependencies that an application patch inherits can be inversely related to the number of affected objects in the run-time application. As an application patch inherits fewer dependencies, the run-time application might contain more affected objects.

Example. Dependency Inheritance

The following image shows the direct and indirect dependencies that a patch inherits where the selected object is a mapping `Mapping A`:



The patch inherits the following direct and indirect dependencies:

- **Direct dependencies.** The patch inherits the reusable transformation `Reusable Transformation A` and the data object `Physical Data Object A` because the objects are direct dependencies for the selected mapping `Mapping A`.
- **Indirect dependencies.** The patch inherits the mapping `Mapping B` and the data object `Physical Data Object B` because the objects are indirect dependencies for the selected mapping `Mapping A`.

Note that the patch does not inherit the workflow `Workflow B` and the mapping `Mapping C` because the objects are remote dependencies for the selected mapping `Mapping A`.

Inheriting Only Direct Dependencies

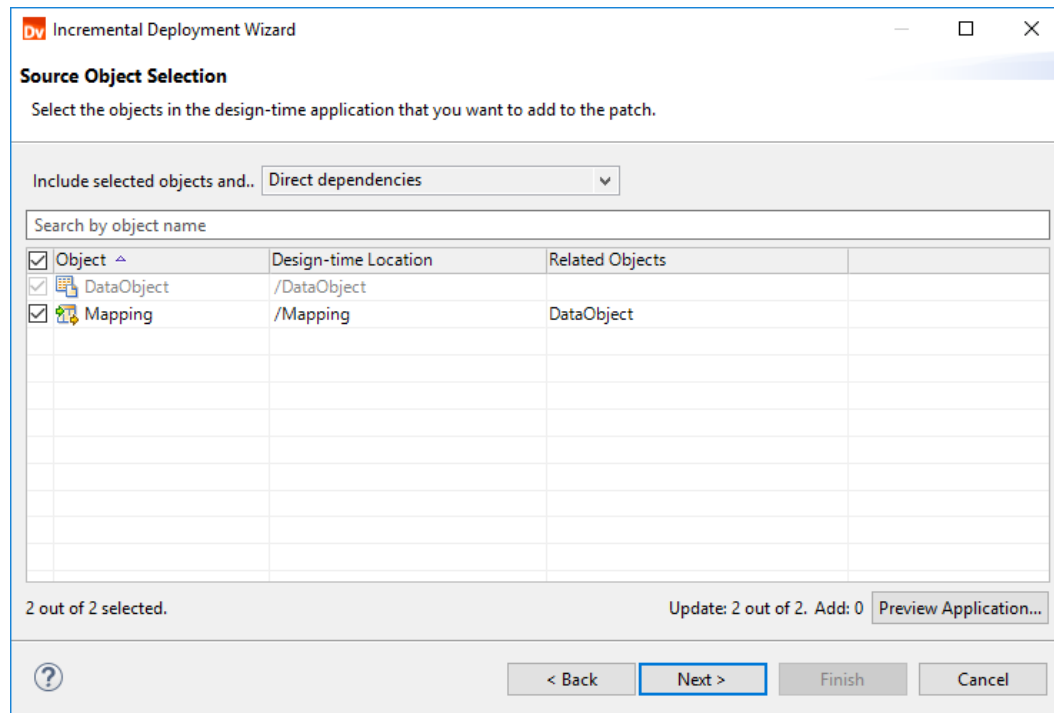
Create a patch that inherits only direct dependencies when you do not have complete information about the design-time and run-time applications in order to minimize the impact that you have on the objects in the run-time application.

Example

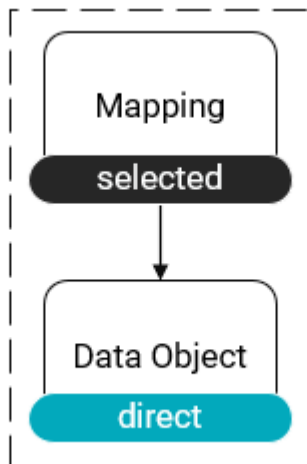
You share an application with thirty other users. The application object that you need to update is a single streaming mapping that reads data from a Kafka data object. After testing the mapping in the run-time application, you edit the mapping in the Developer tool and deploy an application patch to update the mapping in the run-time application.

When you create the application patch, you select the mapping and choose to inherit only direct dependencies. When you select the mapping, the patch inherits the data object that the mapping uses.

The following image shows how the inheritance appears in the Incremental Deployment Wizard:



The following image indicates the selected object and the inherited direct dependency:



Inheriting Direct and Indirect Dependencies

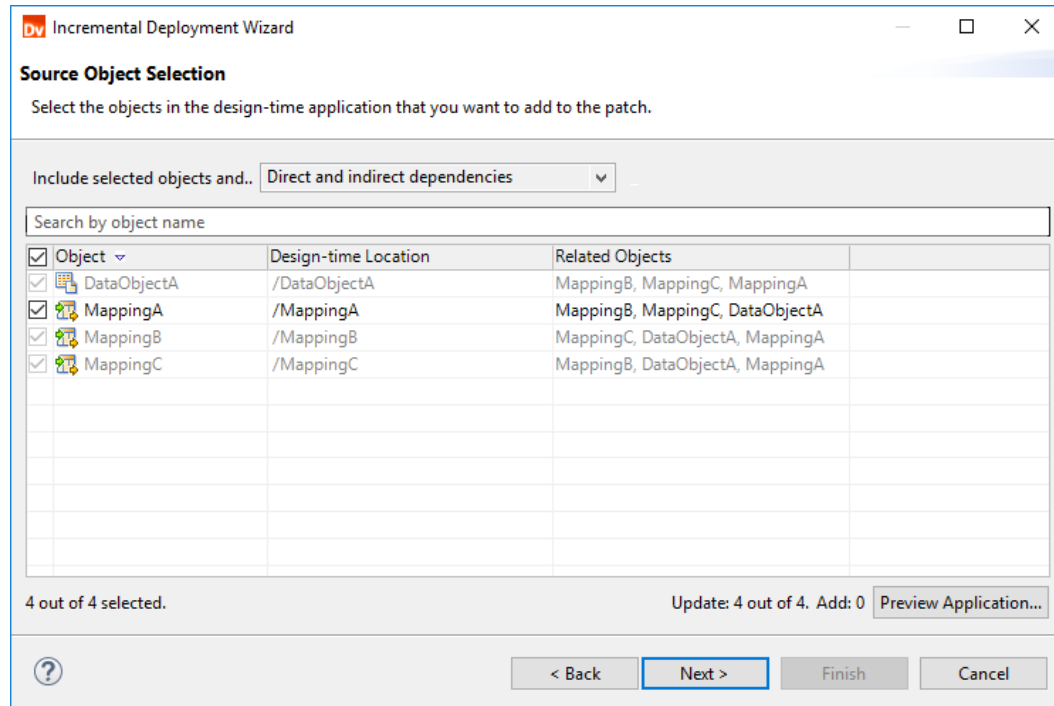
Choose to inherit direct and indirect dependencies when you edit an application object that is used by many other application objects in order to propagate the changes to the parent objects.

Example

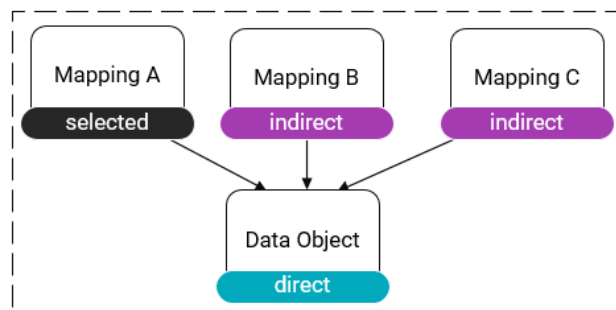
You have an application that contains multiple mappings that use the same relational data object. Each mapping ultimately transforms the data in the data object for a different purpose. To schedule mapping runs, you deploy the application to the Data Integration Service.

Later, you use the Developer tool to synchronize the relational data object with the relational database. To propagate the synchronized data to the mappings in the run-time application, you create a patch that inherits direct and indirect dependencies:

When you create the patch, you select one of the mappings that uses the data object and the patch inherits every other mapping that uses the data object. The following image shows how the inheritance appears in the Incremental Deployment Wizard:



The following image indicates the selected object and the inherited indirect dependencies:



Inheriting Direct, Indirect, and Remote Dependencies

Inherit direct, indirect, and remote dependencies when you have complete information about the design-time and run-time applications.

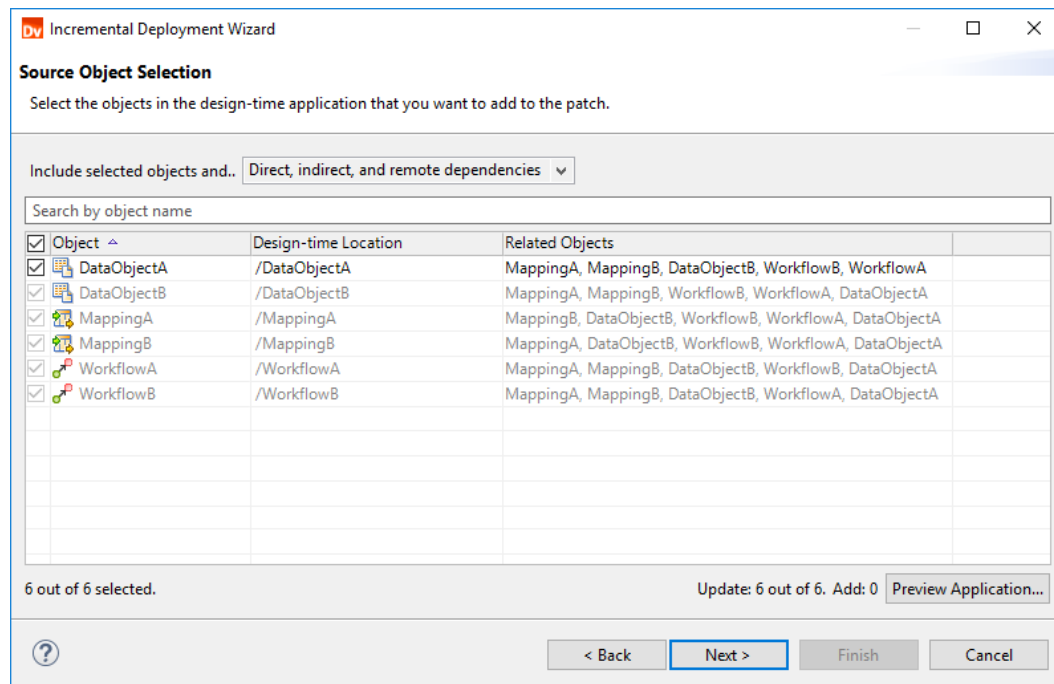
You might have complete information about an application when you are the sole developer or the functional administrator for an application. When you make changes to the design-time application, you might expect all of the changes to be propagated to the run-time application to ensure that the run-time application mirrors the design-time application.

Example

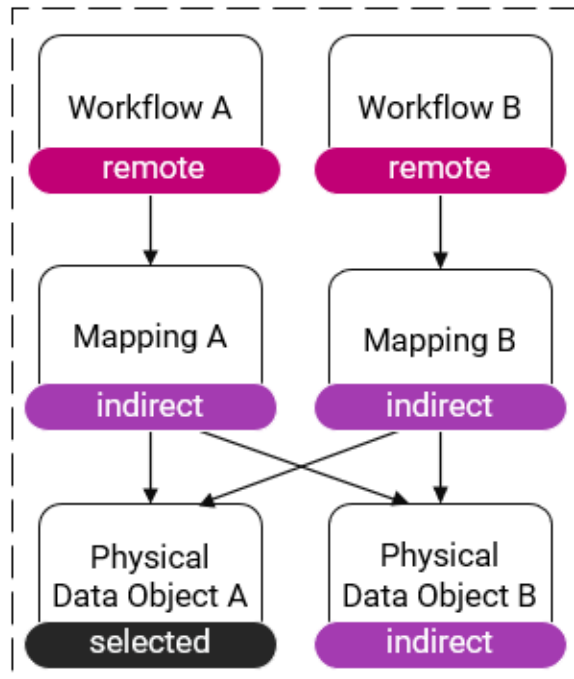
You manage an application that contains workflows that run mappings which share different data objects. After you deploy the application and test the outputs, you edit one of the data objects.

To propagate the changes in the data object to every other application object that uses the data object, you create a patch that inherits direct, indirect, and remote dependencies. When you select the data object, the patch inherits any workflows and mappings that use the data object.

The following image shows how the inheritance appears in the Incremental Deployment wizard:



The following image indicates the selected object and the inherited direct, indirect, and remote dependencies:



Note: To propagate the changes in the data object `Physical Data Object A` to the mappings and workflows that use the data object in the run-time application, you can also use a patch that inherits only direct dependencies. If you select the data object, the wizard identifies the mappings and workflows to be affected objects and the Data Integration Service updates the mappings and workflows during patch deployment.

A patch that inherits direct, indirect, and remote dependencies, however, provides more transparency about how the objects will be updated. It provides a higher guarantee that updated objects in the run-time application will transform data in the same way as the objects in the design-time application.

Determine Application Object Impact

When you create and deploy an application patch, the impact that the patch has on the run-time application extends to the objects that you modify as well as any affected objects. If multiple users share the application, the patch might impact objects that other users maintain.

To understand how much impact a patch will have on the run-time application, you can perform the following tasks:

- Preview the object impact. The object impact is a summary that describes how each application object will be impacted.
- Identify the affected objects. Affected objects might be objects that belong to other users. You can identify affected objects to understand how other users' objects might change.
- Review the scope of the patch. The scope is a numerical summary of how much many objects a patch will impact.

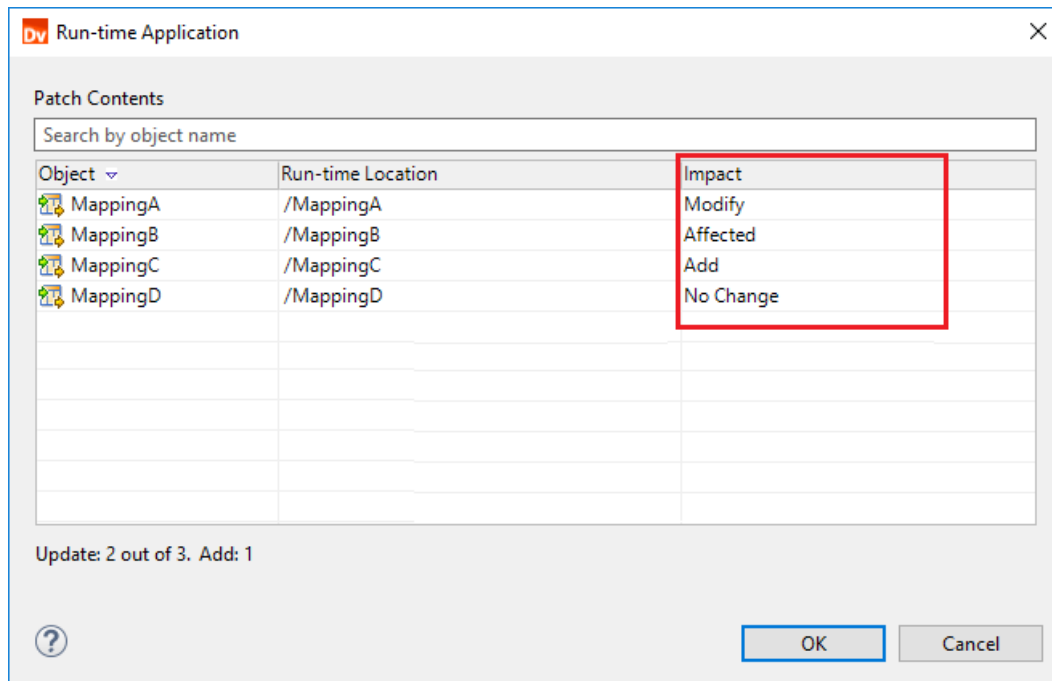
Preview Object Impact

The Incremental Deployment Wizard analyzes the impact to the run-time application while you add objects to the application patch and the patch inherits the objects' dependencies. You can preview the impact on each application object to understand how the run-time instance of an object will change after you deploy the patch.

The preview lists the following types of impact for each application object:

- **Modify.** A run-time instance of the object exists in the run-time application, and the Data Integration Service will replace the run-time instance with the design-time instance.
- **Affected.** The wizard will fetch the object from the run-time application because its run-time dependencies will be modified. The Data Integration Service updates the affected object in the run-time application to use the modified dependencies, but the service does not replace the it.
- **Add.** A run-time instance of the object does not exist in the run-time application, so the Data Integration Service will create a run-time instance of the object in the run-time application.
- **No Change.** The object will not be modified, affected, or added to the run-time application.

The following image shows where you can preview the object impact in the Developer tool:



Identify Affected Objects

An affected object is an object that the Incremental Deployment Wizard fetches from the run-time application in order to maintain the deployed application. Users who own the affected objects will not be able to run the objects while the Data Integration Service applies a patch.

The wizard fetches affected objects while you design the application patch. You can view the objects that are affected when you preview object impact.

The wizard fetches affected objects for the following reasons:

Change propagation

When you select objects in an application patch and the patch inherits other application objects, the wizard must decide how the changes in the objects will be propagated to their parent objects in the run-time application. The wizard identifies the parent objects to be affected due to the changes.

When you deploy the patch, the Data Integration Service does not replace the run-time instances of affected objects, but it updates the affected objects to consume any changes that are made to their run-time direct dependencies.

Note: If a workflow is an affected object, the workflow is updated the next time that it runs. For example, if you modify a mapping and a workflow uses the mapping in a Mapping task, the workflow uses the modified mapping in subsequent executions.

Validity

The wizard must ensure that application objects in the run-time application will be valid after the Data Integration Service applies a patch. After the wizard decides how changes will be propagated to affected objects, the wizard validates the affected objects. An affected object might not be valid if it is not compatible with a modified object.

Any application patch type can result in affected objects. For example, when you create a patch that inherits direct, indirect, and remote dependencies, a remote dependency might have a direct dependency that the patch inherits. If the direct dependency is used by a different object in the run-time application, the parent object becomes an affected object.

After the Data Integration Service updates an affected object, the data in the affected object might not be consistent with the application design. Consider a run-time application that contains a mapping which uses a mapplet. If the modified mapplet uses a different transformation, the affected mapping might be valid but it might also transform the data in an unpredictable way.

If possible, try to select the objects that you want to update, or design an application patch to inherit the objects to minimize the number of affected objects and guarantee that the objects in the run-time application transform data in the same way as the objects in the design-time application.

Example. Change Propagation and Validity

You deploy a patch that modifies a data object. The wizard propagates the modifications in the data object to any other run-time objects that use the data object. As a result, an affected run-time mapping that uses the data object will use the modified data object after the Data Integration Service applies the patch.

If a mapping uses the modified data object and the data object contains fewer rows than before the patch was applied, the mapping might no longer be valid. The data object might not be compatible with the transformations in the mapping due to a fewer number of links between upstream and downstream transformations. If the affected mapping is not valid, you cannot deploy the patch.

Deleted Objects

When you delete an application object from the design-time application and deploy a patch, the Data Integration Service does not delete the object from the run-time application. In some cases, the deleted object might be an affected object.

If the wizard identifies a deleted object as an affected object, the wizard will attempt to propagate changes in the deleted object's child objects to the deleted object. The wizard will also attempt to validate the deleted object. If the deleted object is not valid, you cannot deploy the patch.

If users are not running deleted objects, redeploy the application to remove the objects from the run-time application.

Example. Identifying Affected Objects

The following example shows how you can identify the affected objects that the Incremental Deployment Wizard fetches from the run-time application.

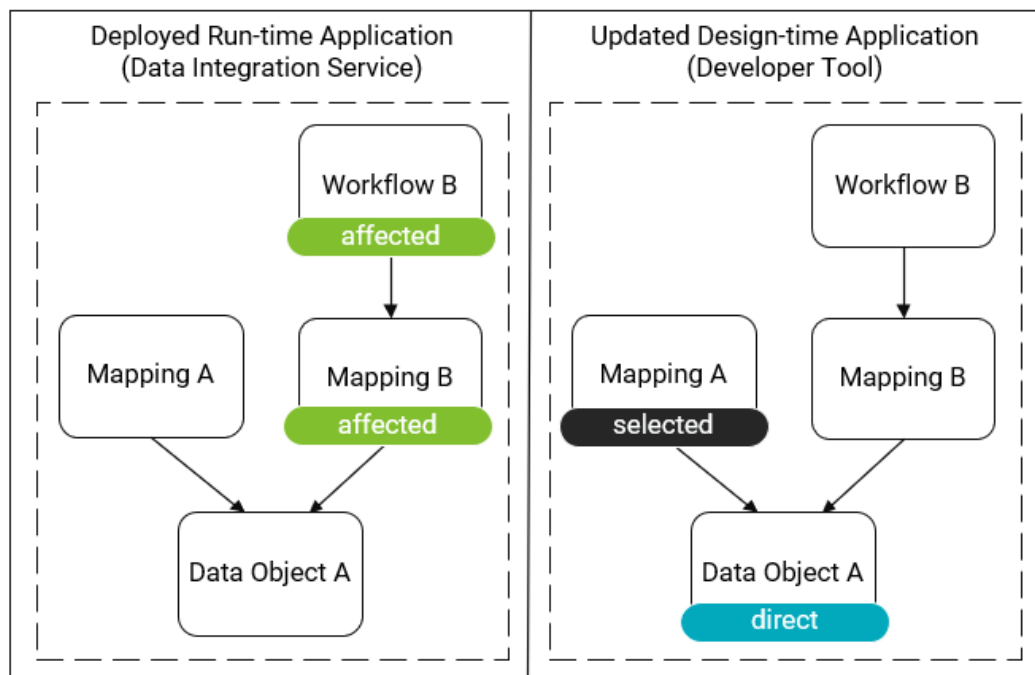
You might have an application that contains two mappings `Mapping A` and `Mapping B` that share a physical data object `Data Object A`, and a workflow `Workflow B` that uses the mapping `Mapping B`. After you deploy the application, you edit the mapping `Mapping A` and decide to update the mapping in the run-time application.

You create a patch that inherits only direct dependencies and select the mapping `Mapping A`. Because you select the mapping, the patch inherits the data object that the mapping uses, `Data Object A`.

Since the wizard expects the Data Integration Service to replace the run-time instances of the mapping and the data object, the wizard locates any parent objects that use the mapping or the data object in the run-time application. The mapping `Mapping B` uses the data object, so the wizard marks the mapping as an affected object.

Because the mapping `Mapping B` will be updated in the run-time application and the workflow `Workflow B` uses the mapping `Mapping B`, the workflow `Workflow B` is also an affected object.

The following image shows the design-time and run-time conditions, the selected mapping, the dependencies that the patch inherits, and the affected objects:



Note: The run-time instances of the objects `Mapping B` and `Workflow B` will not be replaced with their design-time instances. If you edited the mapping `Mapping B` in the design-time application, the run-time instance of `Mapping B` will be different from the design-time instance.

Identify the Scope of Object Impact

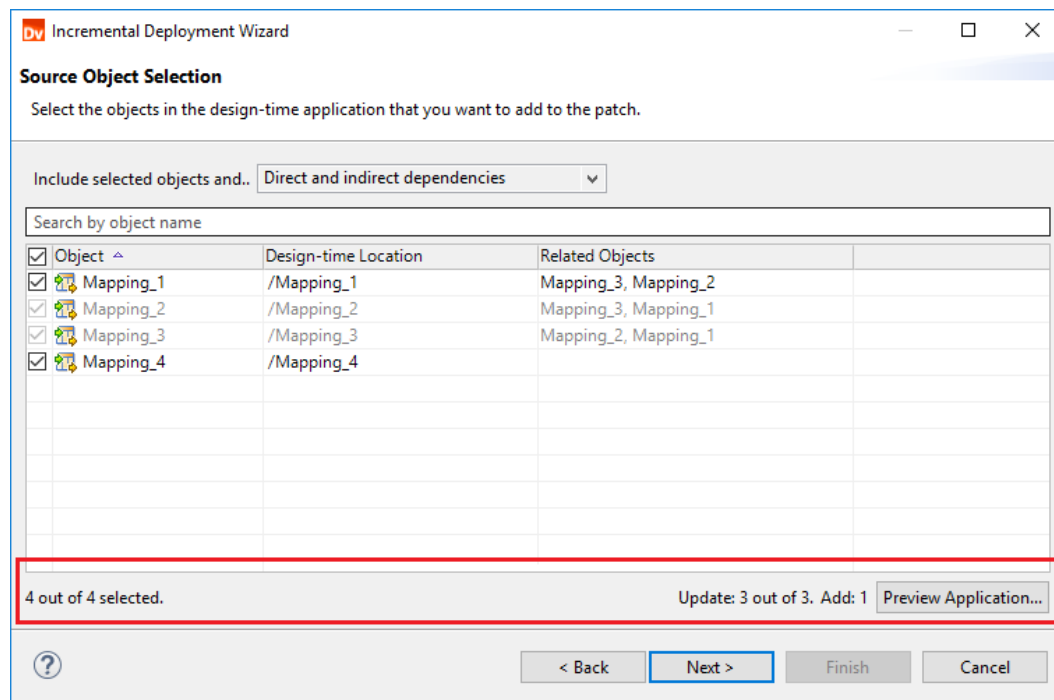
Identify the scope of the impact that an application patch will have on the deployed application to understand how many objects will experience down-time during patch deployment. When an application has multiple users, a larger scope lowers the number of objects that are available to those users.

To identify the scope, use values from the impact summary based on the following object types:

- Selected objects. The sum of the objects that you select to appear in the patch and the objects that the patch inherits. The maximum value is the number of application objects in the design-time application.
- Updated objects. The sum of the objects that will be modified or affected. The maximum value is the number of application objects in the run-time application.
- Added objects. The number of objects that will be added to the run-time application.

Note: The number of selected objects might be different from the number of updated objects if the design-time application contains a different set of dependencies from the run-time application.

The following image shows where the Incremental Deployment wizard displays the summary:



Specify a Patch Deployment Strategy

Specify a patch deployment strategy to determine how you want to make the contents of an application patch available to end users depending on organizational policies.

You can deploy an application patch in the following ways:

- Deploy to a Data Integration Service. When you deploy the patch to a Data Integration Service, the service automatically updates the corresponding incremental application based on the patch contents. Objects that are not impacted will continue running, and impacted objects will be updated with the patch contents.

- Deploy to an archive file. When you deploy the patch to an archive file, you can save the patch as a file on your local machine, store the file in a version control system, and deploy the patch to a Data Integration Service at a later time.

Deploying a patch does not change the state of the application. For example, a stopped application does not begin running.

If you cancel the deployment, the Data Integration Service might partially apply the patch contents to the deployed application. Canceling deployment does not roll back the incremental application.

Patch Deployment Strategy Properties

On the **Patch Deployment Strategy** page, you can configure general properties and deployment options. You can also review the patch history of the run-time application.

The following table describes the general properties that you can configure:

General Property	Description
Patch Name	Name of the patch.
Patch Description	Optional. Description of the patch. The patch description cannot exceed 270 characters.
Retain State Information	Optional. Indicates whether state information is retained or discarded. State information refers to mapping properties and the properties of run-time objects such as mapping outputs or the Sequence Generator transformation. By default, state information is retained. Note: Your choice to retain or discard application state has no effect on the Reset property of a Sequence Generator transformation. For more information about state information, see "Application State Information" on page 153 .

The following table describes the patch deployment options that you choose from:

Deployment Option	Description
Deploy to Data Integration Service	Select this option to deploy the patch to a Data Integration Service.
Deploy to File	Select this option to deploy the patch to an application patch archive file.

The following table describes the properties that you can review for each patch in the patch history:

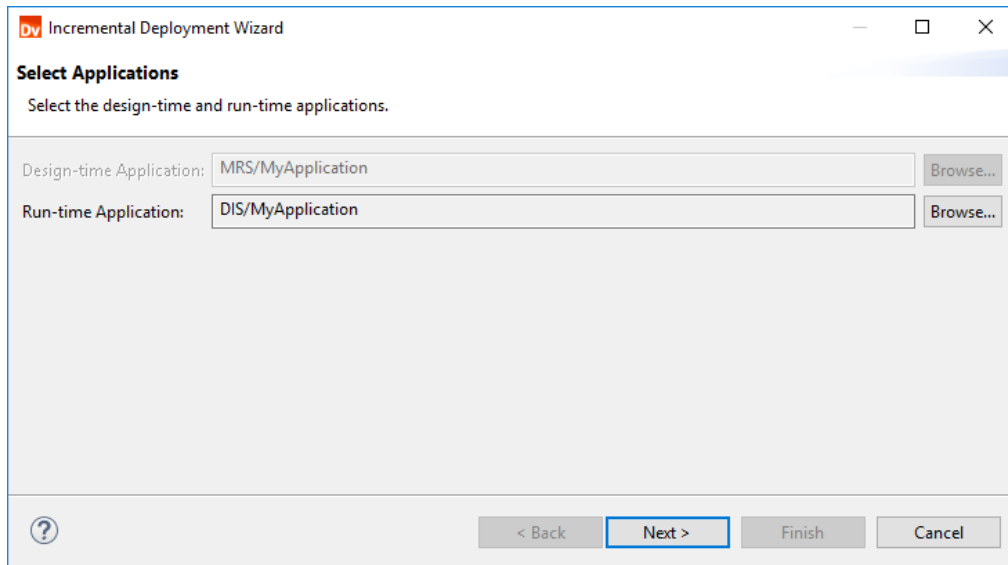
Patch Property	Description
Name	Name of the deployed patch.
Description	Description of the deployed patch. The time that the patch was created is appended to the beginning of the patch description.

Note: By default, the patch history lists patches by the time that they were created.

Deploying an Application Patch

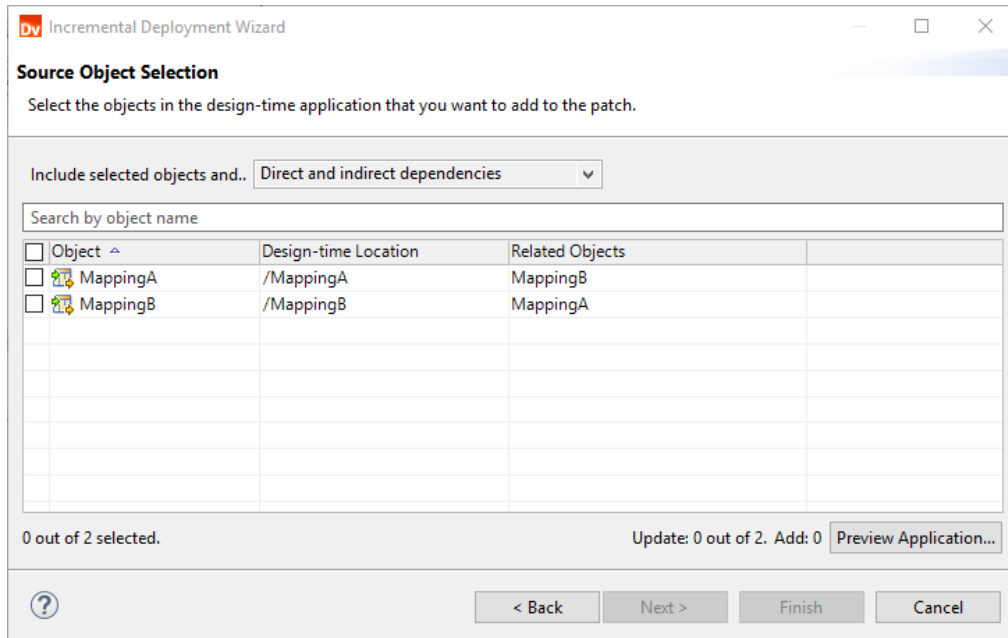
If you edit an object in an incremental application, you can update the object in the deployed application by creating and deploying an application patch. Deploy the application patch to a Data Integration Service or to an application patch archive file.

1. In the **Object Explorer** view, right-click an incremental application and select **Update Application Objects**. The **Select Applications** page of the Incremental Deployment Wizard appears.



2. On the **Select Applications** page, click **Browse** and select the run-time application that you want to update on the Data Integration Service.
3. Click **Next**.

The wizard fetches the run-time application. Then, the **Source Object Selection** page appears.



4. On the **Source Object Selection** page, select the objects in the design-time application that you want the Data Integration Service to replace in the run-time application.
5. Select one of the following application patch types:
 - Direct dependencies. The patch inherits the direct dependencies of the selected objects.
 - Direct and indirect dependencies. The patch inherits the direct and indirect dependencies of the selected objects.
 - Direct, indirect, and remote dependencies. The patch inherits the direct, indirect, and remote dependencies of the selected objects.

Default is Direct and indirect dependencies.

The dependencies appear in the **Related Objects** column. For example, for an application patch that inherits direct and indirect dependencies, the column lists the selected objects' direct and indirect dependencies.

6. To review the impact on each object in the run-time application, click **Preview Application**.
7. Click **Next**.

The **Patch Deployment Strategy** page appears.

Incremental Deployment Wizard

Patch Deployment Strategy
Select the deployment strategy for the patch.

Patch Name:

Patch Description:

Retain State Information:

Deploy

Deploy to Data Integration Service

Deploy to File

Patch History

	Name	Description
1	Patch1	24-Sep-2019 11:12:03 This is a patch.

< Back Next > **Finish** Cancel

8. Enter the name of the patch.
9. Optionally, enter a description of the patch.
10. Optionally, choose to retain state information.
If you choose to not retain state information, state information will be discarded.
11. Select one of the following deployment options:
 - Deploy to Data Integration Service. Deploys the application patch to the Data Integration Service. The Data Integration Service applies the application patch to the run-time application.
 - Deploy to File. Deploys the application patch to an archive file. If you deploy the patch to an archive file, select a directory on your local machine where you want to save the file.
12. Review the patch history.

The patch history lists the application patches that have been deployed to update the incremental application.

Rules and Guidelines for Application Patch Deployment

Refer to the following rules and guidelines when you update an incremental application by deploying an application patch:

- If you select to update application objects from the Object Explorer view, the Incremental Deployment Wizard begins constructing the application dependencies. You can view the progress in the Progress view, but you cannot cancel the operation.

- When you create a patch, the Incremental Deployment wizard does not list any objects that are not objects in the application.

For example, an application can contain a mapping and exclude the data objects that the mapping uses to transform data. In this case, the data objects are not application objects. When you create a patch, you will not be able to see the data objects, but the wizard will process the data objects when it identifies the dependencies between application objects.

- If you change the name or the project of an application object in the design-time application and update the object using an application patch, the object is duplicated in the run-time application under different paths.
- If the application description of the run-time application is empty the first time that you deploy a patch, the Incremental Deployment Wizard updates the description using the application's patch history. When you deploy additional patches, the wizard appends the patch names to the description. The wizard continues appending patch names until you edit the application description using any client.

CHAPTER 11

Application Patch Deployment Example

This chapter includes the following topics:

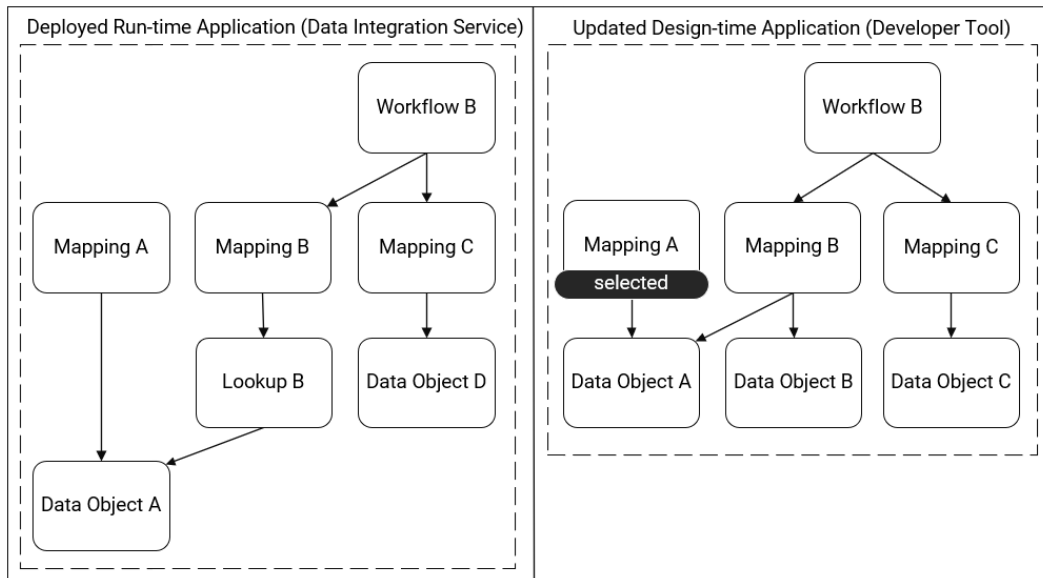
- [Application Patch Deployment Example Overview, 185](#)
- [Patch Type: Direct, 187](#)
- [Patch Type: Direct and Indirect, 189](#)
- [Patch Type: Direct, Indirect, and Remote, 191](#)
- [Conclusion, 193](#)

Application Patch Deployment Example Overview

You are a DevOps engineer who observes changes in design-time objects and merges the changes to the corresponding run-time objects in a deployed application on the Data Integration Service.

Recently, a developer edited a mapping `Mapping A` in an incremental application. You want to create a patch to update the mapping in the run-time application.

The following image shows the run-time application on the Data Integration Service and the design-time application in the Developer tool where the mapping Mapping A is the selected object:



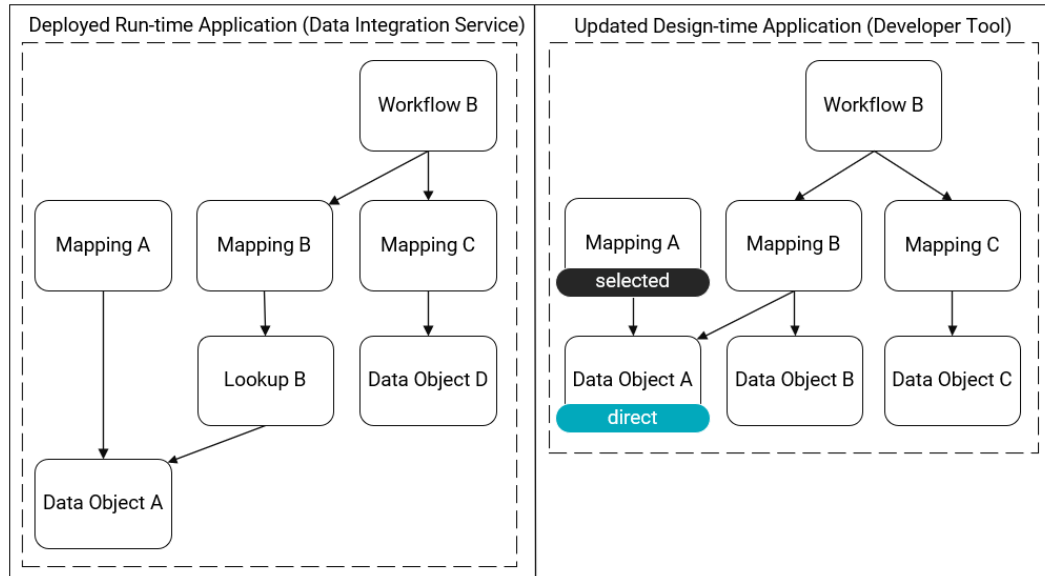
To create an application patch, you need to choose the dependencies that you want the patch to inherit. You identify the inherited objects, assess the object impact, and evaluate the scope of the changes to the deployed application.

Patch Type: Direct

This section describes an application patch that inherits only direct dependencies.

Identify the Inherited Objects

The following image shows the objects that the patch includes when you inherit only direct dependencies:

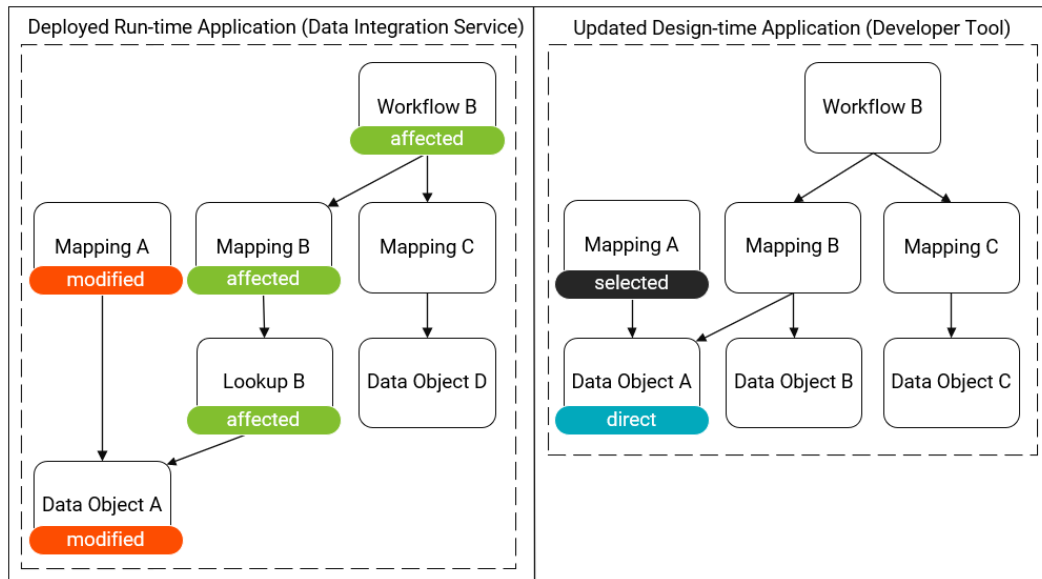


The following table lists the dependencies that the patch inherits when the mapping `Mapping A` is the selected object:

Type of Dependency	Dependencies
Direct	- Data Object A

Assess the Object Impact

The following image shows the impact to the run-time objects after the patch is deployed:



Evaluate the Scope of the Changes

The following table shows the number of objects that are impacted during patch deployment:

Scope Item	Number of Objects
Selected	2
Updated	Modified: 2 Affected: 3* Total: 5*
Added	0

* The reusable transformation *Lookup B* behaves as an affected object at run time, but the Incremental Deployment wizard does not list reusable transformations as affected objects. In the Incremental Deployment wizard, the number of affected objects is 2.

Findings

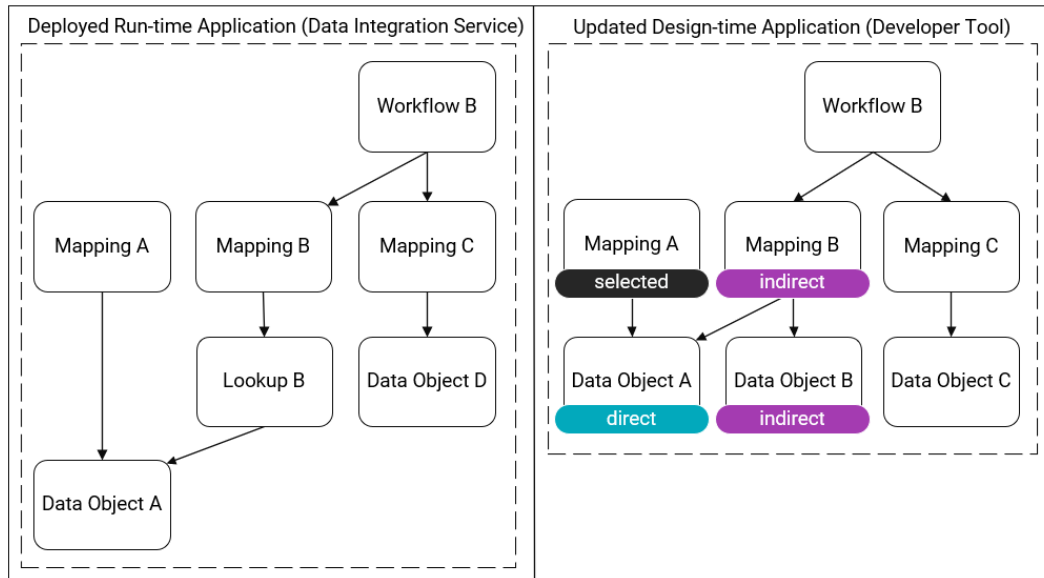
A patch that inherits only direct dependencies has a low impact on the run-time application, but the run-time application contains affected objects. Affected objects can be unpredictable compared to modified objects, so there is less transparency about how the application will transform data after you apply the patch.

Patch Type: Direct and Indirect

This section describes an application patch that inherits direct and indirect dependencies.

Identify the Inherited Objects

The following image shows the objects that the patch includes when you inherit direct and indirect dependencies:

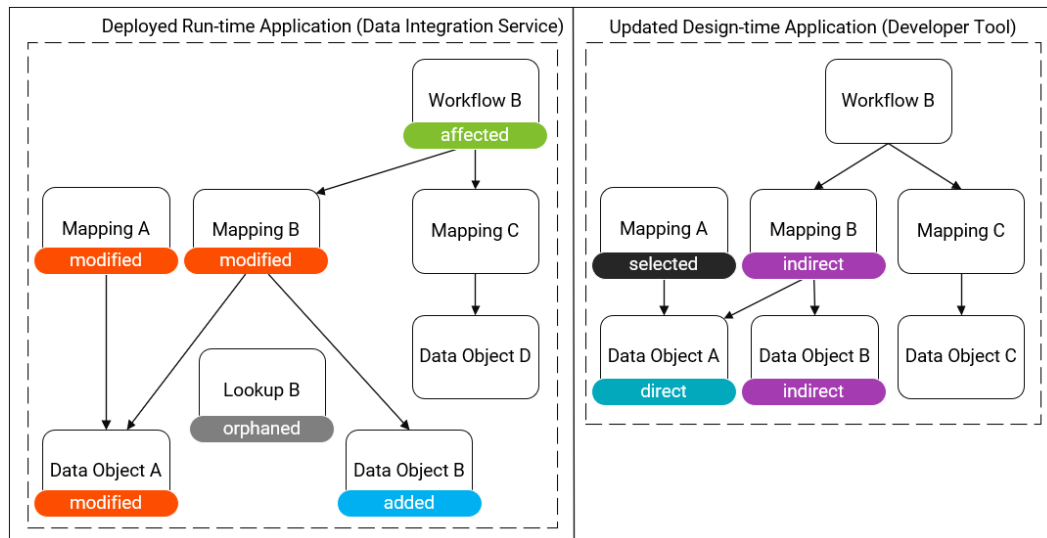


The following table lists the dependencies that the patch inherits when the mapping `Mapping A` is the selected object:

Type of Dependency	Dependencies
Direct	- Data Object A
Indirect	- Mapping B - Data Object B

Assess the Object Impact

The following image shows the impact to the run-time objects after the patch is deployed:



Because the patch inherits the mapping `Mapping B` as an indirect dependency, the mapping will use its design-time dependencies and discard its previous run-time dependencies. It no longer uses the reusable transformation `Lookup B` in the run-time application. Instead, it uses the data objects `Data Object A` and `Data Object B`. The transformation `Lookup B` becomes an orphan in the run-time application.

Evaluate the Scope of the Changes

The following table shows the number of objects that are impacted during patch deployment:

Scope Item	Number of Objects
Selected	4
Updated	Modified: 3 Affected: 1 Total: 4
Added	1

Findings

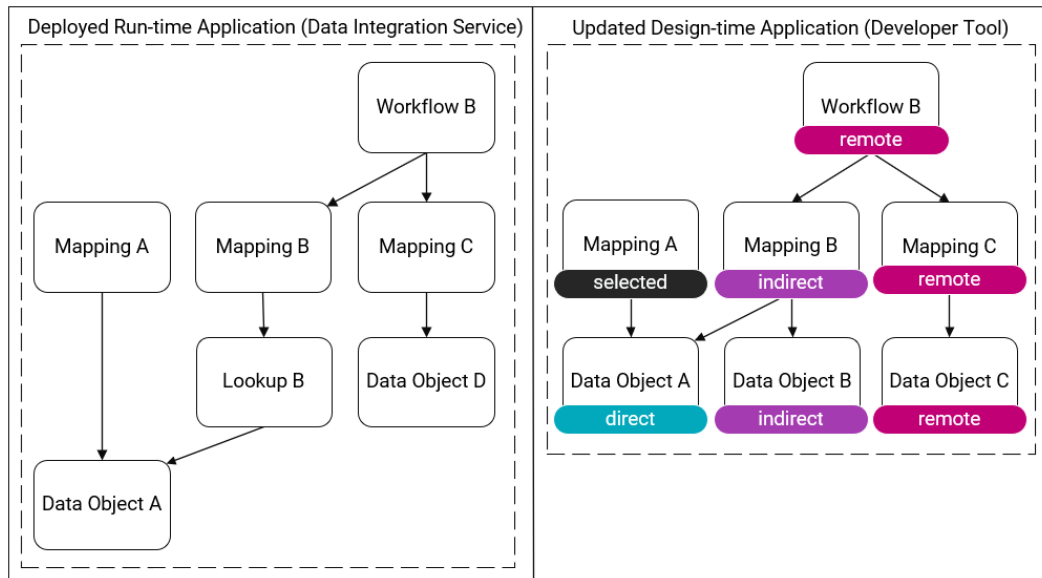
Inheriting direct and indirect dependencies has a similar impact to inheriting only direct dependencies, but fewer objects are affected. The mapping `Mapping B` is modified instead of affected, so it is updated in a more predictable way.

Patch Type: Direct, Indirect, and Remote

This section describes an application patch that inherits direct, indirect, and remote dependencies.

Identify the Inherited Objects

The following image shows the objects that the patch includes when you inherit direct, indirect, and remote dependencies:

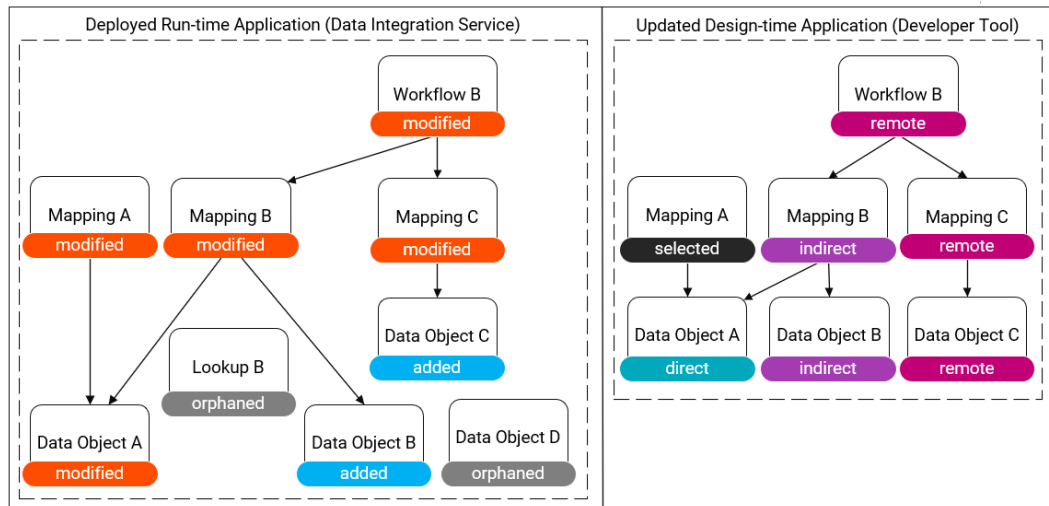


The following table lists the dependencies that the patch inherits when the mapping `Mapping A` is the selected object:

Type of Dependency	Dependencies
Direct	- Data Object A
Indirect	- Mapping B - Data Object B
Remote	- Workflow B - Mapping C - Data Object C

Assess the Object Impact

The following image shows the impact to the run-time objects after the patch is deployed:



Because the patch inherits mappings `Mapping B` and `Mapping C`, the mappings use their design-time dependencies in the run-time application. The reusable transformation `Lookup B` and the data object `Data Object B` become orphans in the run-time application.

Evaluate the Scope of the Changes

The following table shows the number of objects that are impacted during patch deployment:

Scope Item	Number of Objects
Selected	7
Updated	Modified: 5 Affected: 0 Total: 5
Added	2

Findings

If you create a patch that inherits direct, indirect, and remote dependencies, the changes to the run-time application are transparent. The updated and added objects in the run-time application match the selected objects in the design-time application and you can expect the run-time application to transform data in the same way as the objects in the design-time application.

The trade-off is that all objects experience down-time, and you might be better off by redeploying the application. When you redeploy the application, you update the same objects with the benefit of removing the orphaned objects from the run-time application: the reusable transformation `Lookup B` and the data object `Data Object D`.

Conclusion

In this example, creating a patch that inherits direct and indirect dependencies has the least impact on the run-time application, so it maximizes object up-time. It allows you to take full advantage of the abilities of an incremental application.

You might choose to create a patch that inherits direct and indirect dependencies if the application is deployed in a development or QA environment. For example, you might want to test that the selected mapping `Mapping A` runs successfully using sample data before you begin processing enterprise data.

If the application is deployed in a user acceptance testing (UAT) or production environment and you have the capacity to stop the application, you might choose to redeploy the application instead. Redeploying the application creates the most transparent changes in the run-time application to ensure that the data is transformed according to your requirements.

CHAPTER 12

Continuous Integration and Continuous Delivery (CI/CD)

This chapter includes the following topics:

- [CI/CD Overview, 194](#)
- [REST API and infacmd Quick Reference, 195](#)
- [Continuous Integration, 198](#)
- [Continuous Delivery, 199](#)
- [Sample Automated Integration Process, 200](#)
- [Sample Automated Delivery Process, 202](#)

CI/CD Overview

CI/CD, or continuous integration and continuous delivery, is a practice that automates the integration and delivery operations in a CI/CD pipeline. You can automate each integration and delivery operation using the Data Integration Service REST API or the infacmd command line programs.

A CI/CD pipeline includes the integration operations that developers use to design objects and the delivery operations that deliver the objects to the production environment. You can use the REST API or infacmd to automate integration and delivery in the following ways:

- Deploy and test every change that a developer makes to an object. Developers receive instant feedback about whether objects pass or fail testing and the types of changes that objects require.
- Deliver objects that pass testing to the production environment. Based on organizational requirements, you can deliver objects to additional requirements, such as QA and UAT, before delivering the objects to the production environment.

Continuous integration and continuous delivery are fully logged and visible to the entire team so that team members can allocate time away from manual tasks.

REST API and infacmd Quick Reference

Refer to the common integration and delivery operations that you can use to set up continuous integration and continuous delivery.

For more information about the REST API requests, see one of the following locations:

- REST API documentation that you access through the Data Integration Service process properties or the REST Operations Hub Service process properties.
- *Informatica Application Service Guide*.

For more information about the infacmd commands, see the *Informatica Command Reference*.

Get Information about Objects

The following table lists the operations that you can automate to get information about objects:

Operation	REST API	infacmd
Query design-time objects	Get design-time objects	dis queryDesignTimeObjects
Query run-time objects	Get run-time objects	dis queryRunTimeObjects
List applications	Get run-time objects	dis listApplications
List application objects	Get run-time objects	dis listApplicationObjects
List parameter sets	Get run-time objects	dis listParameterSets
List application patches	Not applicable	dis listPatchNames
Compare mappings	Compare two queried mappings	dis compareMapping

Tag Objects

The following table lists the operations that you can automate to tag objects:

Operation	REST API	infacmd
Tag an object	Assign tags to design-time objects	dis tag
Replace object tags	Replace repository object tags	dis replaceAllTag
Untag an object	Remove tags from design-time objects	dis untag

Detect and Resolve Conflicts

The following table lists the operations that you can automate to detect and resolve conflicts:

Operation	REST API	infacmd
Unlock an object	Not applicable	mrs unlockObject
List locked objects	Not applicable	mrs listLockedObjects

Perform Version Control Operations

The following table lists the operations that you can automate to perform version control operations:

Operation	REST API	infacmd
Check in an object	Perform version control operations	mrs checkInObject
Check out an object	Perform version control operations	Not applicable
Undo a check-out	Perform version control operations	mrs undoCheckout
Reassign a checked-out object	Perform version control operations	mrs reassignCheckedOutObject
List checked-out objects	Get design-time objects	mrs listCheckedOutObjects

Note: Version control operations are available only if the Model repository is integrated with a version control system.

Deploy Objects

The following table lists the operations that you can automate to deploy objects:

Operation	REST API	infacmd
Deploy objects to an application archive file	Not applicable*	tools deployApplication
Deploy objects to an application patch archive file	Deploy design-time objects to an archive file	dis deployObjectsToFile
Deploy an application archive file to a Data Integration Service	Deploy an application	dis deployApplication
Deploy an application patch archive file to a Data Integration Service	Update an application	tools patchApplication
Redeploy an application archive file to a Data Integration Service	Update an application	dis updateApplication

* You cannot use the REST API to deploy objects to an application archive file. Use an application patch archive file to deploy an application to a Data Integration Service for the first time or to redeploy an application.

Manage Deployed Applications

The following table lists the operations that you can automate to manage deployed applications:

Operation	REST API	infacmd
Start an application	Start or stop an application	dis startApplication
Stop an application	Start or stop an application	dis stopApplication
Backup an application	Not applicable	dis backupApplication

Operation	REST API	infacmd
Restore an application	Not applicable	dis restoreApplication
Rename an application	Not applicable	dis renameApplication
Undeploy an application	Undeploy an application	dis undeployApplication
Run a deployed mapping	Run a deployed mapping	ms runMapping

Using the REST API or infacmd

To decide whether to use the Data Integration Service REST API or the infacmd command line programs to automate integration and delivery operations, assess the requirements of your organization.

The REST API and infacmd have the following differences:

- You can achieve more use cases with infacmd compared to REST API. For example, you can use an infacmd command to backup applications.
- To use infacmd, you have to install Informatica services or the infacmd utilities. In contrast, the REST API does not have any setup requirements and you can run the REST API in environments that do not have Informatica services installed on the client machine.

For information about the Data Integration Service REST API, see the *Informatica Application Service Guide*.

For information about the infacmd command line programs, see the *Informatica Command Reference*.

Using the REST API to Deploy and Redeploy an Application

The Data Integration Service REST API uses application patch archive files to deploy objects. Instead of using an application archive file, you use an application patch archive file to deploy objects to a Data Integration Service for the first time and to redeploy an application.

Use an application patch archive file to perform the following tasks:

Deploy an application for the first time

To deploy objects as an application to a Data Integration Service for the first time, complete the following tasks:

1. Use the REST API `Deploy design-time objects to an archive file to create an application patch archive file`.
2. Use the REST API `Deploy an application to deploy the patch to a Data Integration Service`.

The Data Integration Service uses the objects in the patch to create an incremental application.

Redeploy an application

To redeploy an application, use the REST API `Update an application` and configure the following components:

1. Specify the application patch archive file.
2. Set the request payload parameter `"redeployApplication":true`.

The Data Integration Service replaces the objects in the deployed incremental application with the objects in the patch.

Continuous Integration

Continuous integration ensures that every change that a developer makes to an object is deployed and tested in the development environment before the object is approved for delivery to additional environments.

To set up continuous integration, you can automate the following tasks:

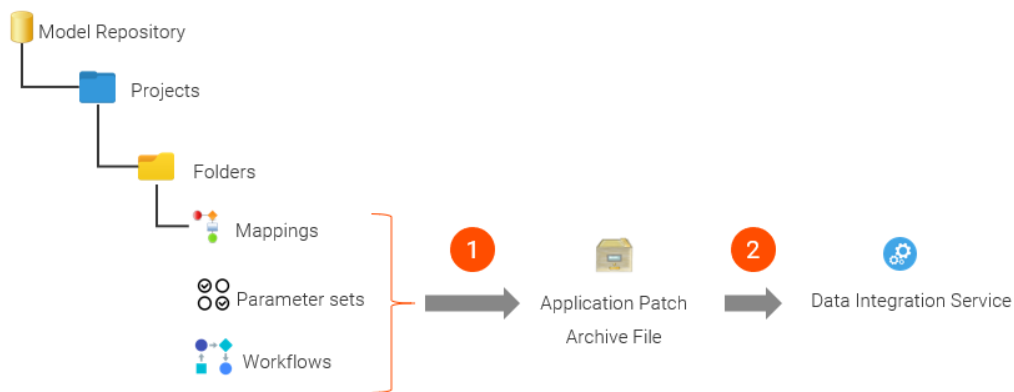
- Deploy objects to run sanity tests.
- Run version control operations to assign objects to developers.
- Detect and resolve conflicts due to object locks.

Deploy Objects

Set up continuous integration to ensure that every change that a developer makes to an object is deployed and tested.

To isolate objects for testing, you deploy the objects to an application archive file and then to a Data Integration Service. When the object is deployed to a Data Integration Service, developers can continue to access and edit the object in the Developer tool without interfering with test results.

The following image shows how objects are deployed:



1. The automated process deploys objects to an application patch archive file.
2. The automated process deploys the archive file to a Data Integration Service.

After an object is deployed, you can run sanity tests to verify that mappings run successfully. To perform further testing, you can use a third-party tool to verify that the mapping output is correct. Based on test results, you can approve the object for delivery to additional environments or assign the object to a developer for editing.

Assign Objects

In a team-based development environment, you can set up continuous integration to assign objects to developers.

For example, if a mapping fails a sanity test, you can assign the mapping for editing to the developer that last modified the mapping. To assign the mapping, you can use the REST API `Perform version control operations` or the command `infacmd mrs reassignCheckedOutObject`.

To assign an object to a developer or to reassign an object from one developer to another developer, you must be able to perform version control operations. Ensure that the Model repository is integrated with a version control system, such as GIT, SVN, or Perforce.

For information about how to integrate a Model repository with a version control system, see the *Informatica Application Service Guide*.

Resolve Object Conflicts

If an object requires changes, you can set up continuous integration to use intent-based locks to detect and resolve conflicts between objects and facilitate developer collaboration.

An intent-based lock prevents other developers from saving changes to the object. Occasionally, locks create conflicts between developers that require access to the same object.

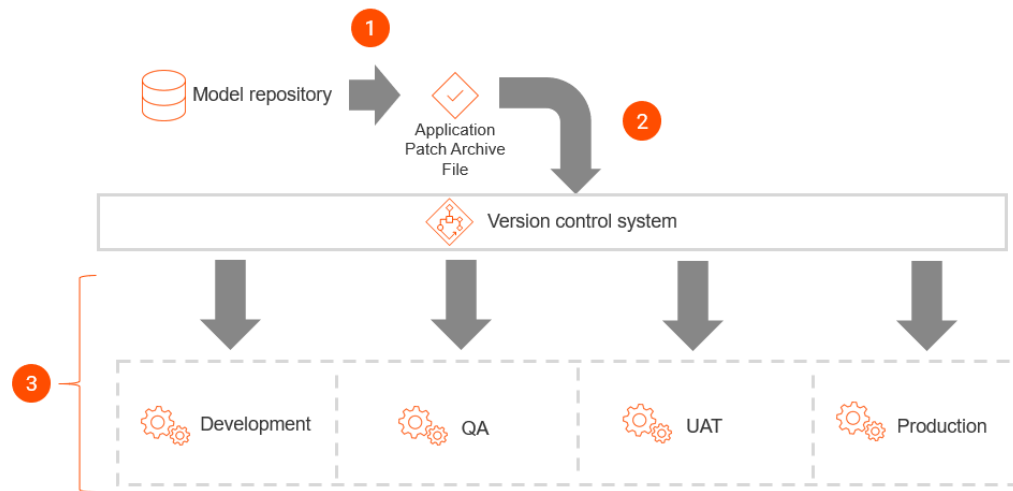
For example, a developer locks an object before going on vacation. The lock creates a conflict for other developers that require access to the object. To resolve such conflicts, you can use a third-party tool to read employee work schedules, detect potential conflicts, and pass logic to `infacmd` commands that list the locked objects and unlock the objects if necessary.

Continuous Delivery

As objects pass testing in the development environment, the objects are continuously delivered to the production environment. Based on organizational requirements, you can set up continuous delivery to additional environments, such as QA and UAT, before delivering the objects to the production environment.

To simplify continuous delivery, you can use archive files to create consistency between the objects that were tested and approved in the development environment and the objects that are delivered to additional environments. To create consistency, set up continuous delivery to deploy objects to one archive file, store the archive file in a version control system, and deploy the archive file to each environment.

The following image shows how one application patch archive file can be deployed to development, QA, UAT, and production environments:



1. Objects are deployed to an application patch archive file.
2. The archive file is stored in a version control system.
3. The same archive file is deployed to development, QA, UAT, and production environments.

Sample Automated Integration Process

This section shows how you can set up continuous integration to deploy and test every change that a developer makes to mappings in the development environment.

Set up continuous integration to perform the following tasks:

1. Prepare mappings for deployment.
2. Deploy mappings to a Data Integration Service.
3. Run sanity tests.
4. Prepare mappings for editing.

Step 1. Prepare Mappings for Deployment

Before you deploy mappings for testing, identify the mappings that have been changed and mark the mappings for deployment.

Set up continuous integration to perform the following integration operations:

1. Identify mappings that have been changed.

To identify mappings that have been changed, continuously compare each design-time mapping to the corresponding run-time mapping that is deployed to the Data Integration Service.

To compare mappings, pass a query to the REST API `Compare two queried mappings` or to the command `infacmd dis compareMapping`.

2. Assign a tag to mark the mappings for deployment.

If the comparison report identifies differences between a design-time mapping and the corresponding run-time mapping, assign a tag to the design-time mapping that marks the mapping for deployment.

For example, assign the tag `DeploymentInProgress` to the mapping. You can pass the tag to the REST API `Assign tags to design-time objects` or to the command `infacmd dis tag`.

Step 2. Deploy Mappings to a Data Integration Service

After mappings are prepared for deployment, deploy the changed mappings to an application patch archive file and then to a Data Integration Service.

Set up continuous integration to perform the following integration operations:

1. Run a query to identify mappings that are marked for deployment.

To identify the mappings that require deployment, query the design-time objects that have the tag `DeploymentInProgress`. Pass the query to the REST API `Get design-time objects` or to the command `infacmd dis queryDesignTimeObjects`.

2. Deploy the mappings to an application patch archive file.

If the query returns at least one mapping, deploy the mappings to an application patch archive file.

Pass the query to the REST API `Deploy design-time objects to an archive file` or to the command `infacmd dis deployObjectsToFile`. The REST API request or the `infacmd` command saves the archive file on the Data Integration Service machine.

3. Deploy the archive file to a Data Integration Service.

Deploy the archive file to a Data Integration Service in the development environment. Pass the location of the archive file to the REST API `Update an application` or to the command `infacmd tools patchApplication`.

Step 3. Run Sanity Tests

When the mappings are updated on the Data Integration Service, run sanity tests to verify that mappings run successfully.

Set up continuous integration to perform the following integration operations:

1. Verify that deployed mappings run successfully.

To verify that deployed mappings run successfully, perform the following tasks:

1. To run deployed mappings, use the REST API `Run a deployed mapping` or the command `infacmd ms runMapping`.
2. Run sanity tests based on organizational requirements. For example, if you automate integration tasks using a script in a third-party tool, you can write test cases to read the data in the mapping target and verify that the mapping output is correct.

2. Assign tags to mark mappings that require additional changes.

To mark mappings that require additional changes, pass the tag `RequiresChanges` to the REST API `Assign tags to design-time objects` or to the command `infacmd dis tag`.

You can also remove the tag `DeploymentInProgress` from the mappings that require changes. Pass the tag `DeploymentInProgress` to the REST API `Remove tags from design-time objects` or to the command `infacmd dis untag`.

3. Assign tags to approve mappings that ran successfully.

To approve mappings that ran successfully, pass the tag `ReadyForDelivery` to the REST API `Replace repository object tags` or to the command `infacmd dis replaceAllTag`. The REST API request or the `infacmd` command replaces all tags that are assigned to the mapping with the tag `ReadyForDelivery`.

Step 4. Prepare Mappings for Editing

If updated mappings fail testing, prepare the mappings for editing.

Set up continuous integration to perform the following integration operations:

1. Assign the mappings to the appropriate developer for editing.

To assign mappings to the appropriate developer, perform the following tasks:

1. Query the design-time mappings that have the tag `RequiresChanges` using the REST API `Get design-time objects` or the command `infacmd dis queryDesignTimeObjects`.
2. Based on the third-party tool that you use, you can parse the query result to get the user name of the developer that last modified each mapping.
3. To assign the mapping to the developer for editing, use the REST API `Perform version control operations` or the command `infacmd mrs reassignCheckedOutObject`.

Note: Version control operations are available only if the Model repository is integrated with a version control system.

2. Notify the developer about the test results.

Optionally, you can use a third-party tool to notify developers about the mappings that require changes.

For example, you can query the design-time mappings that have the tag `RequiresChanges` using the REST API `Get design-time objects` or the command `infacmd dis queryDesignTimeObjects`.

Then, you can run a script in a third-party tool to perform the following tasks:

1. Read the query results to parse the response and get the user name of the developer that last modified the mapping.
2. Look up the e-mail address that is associated with the user name in the employee database.
3. Generate an e-mail message to convey information about the mappings that failed testing and the reason that the mappings failed.

Sample Automated Delivery Process

This section shows how you can set up continuous delivery to deploy mappings to the production environment as the mappings are approved in the development environment.

Set up continuous delivery to perform the following delivery operations:

1. Identify mappings that are approved for delivery.

To identify mappings that are approved for delivery, perform the following tasks:

1. Continuously check whether any mappings in the development environment have the tag `ReadyForDelivery`.
2. Query the design-time mappings for the tag `ReadyForDelivery`.

3. Pass the query to the REST API `Get design-time objects` or to the command `infacmd dis queryDesignTimeObjects`.

2. Deploy the mappings to an application patch archive file.

If the query returns at least one mapping, deploy the mappings to an application patch archive file.

Pass the query to the REST API `Deploy design-time objects to an archive file` or to the command `infacmd dis deployObjectsToFile`. The REST API request or the `infacmd` command save the archive file on the Data Integration Service machine.

3. Deploy the archive file to a Data Integration Service.

Deploy the archive file to a Data Integration Service in the production environment. Pass the location of the archive file to the REST API `Update an application` or to the command `infacmd tools patchApplication`.

CHAPTER 13

Object Import and Export

This chapter includes the following topics:

- [Object Import and Export Overview, 204](#)
- [Import and Export Objects, 205](#)
- [Object Export, 206](#)
- [Object Import, 207](#)

Object Import and Export Overview

You can export multiple objects from a project to an `..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.

Copy metadata between repositories

You can copy objects between repositories that you cannot connect to from the same client. Export the object and transfer the `.xml` file to the target machine. Then import the object from the `.xml` file into the target repository. You can export and import objects between repositories with the same version. If the objects contain tags, the Developer tool automatically imports into the repository.

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` file, and then update the export `.xml` file before you import the objects into a repository.

You can upgrade objects exported to an .xml file from a previous Informatica release to the current metadata format, and then import the updated objects into the current release. Use the `mrs UpgradeExportedObjects` command to upgrade the exported objects to the current metadata format before you import the objects.

Import and Export Objects

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:

Object	Dependency
Application	- SQL data services, mappings, or workflows and their dependent objects
Project	- Projects contain other objects, but they do not have dependent objects
Folder	- Folders contain other objects, but they do not have dependent objects
Reference table	- Reference tables do not have dependent objects
Content set	- Content sets do not have dependent objects
Physical data object (except for customized data object)	- Physical data objects do not have dependent objects
Customized data object	- Physical data objects
Logical data object model	- Logical data objects - Physical data objects - Reusable transformations and their dependent objects - Mapplets and their dependent objects
Transformation	- Physical data objects - Reference tables - Content sets
Mapplet	- Logical data objects - Physical data objects - Reusable transformations and their dependent objects - Mapplets and their dependent objects

Object	Dependency
Mapping	<ul style="list-style-type: none"> - Logical data objects - Physical data objects - Reusable transformations and their dependent objects - Mapplets and their dependent objects
SQL data service	<ul style="list-style-type: none"> - Logical data objects - Physical data objects - Reusable transformations and their dependent objects - Mapplets and their dependent objects
Profile	<ul style="list-style-type: none"> - Logical data objects - Physical data objects
Scorecard	<ul style="list-style-type: none"> - Profiles and their dependent objects
Web service	<ul style="list-style-type: none"> - Operation mappings
Workflow	<ul style="list-style-type: none"> - Mappings and their dependent objects

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. If you want to modify the attributes use the `infacmd xrf` command.

You can also export objects with the `infacmd` tools `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.
2. Select **Informatica > Export Object Metadata File**.
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. If the objects that you select include reference data objects, select the **Export content** option and verify the export settings:

- Verify the name and location for the reference data files that you export. The Developer tool Service exports the reference data files to a single ZIP file. By default, the wizard exports the ZIP file and the XML metadata file to the same directory.
- Verify the code page that the reference data uses. The default code page is UTF-8. If you export reference table data, accept the default code page.
- Verify the probabilistic model data to export. By default, the wizard exports all model data. If the objects that you select do not include a probabilistic model, the export process ignores the option.

8. 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.

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 tools ImportObjects` command.

You cannot import objects from an export file that you created in a previous version.

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.

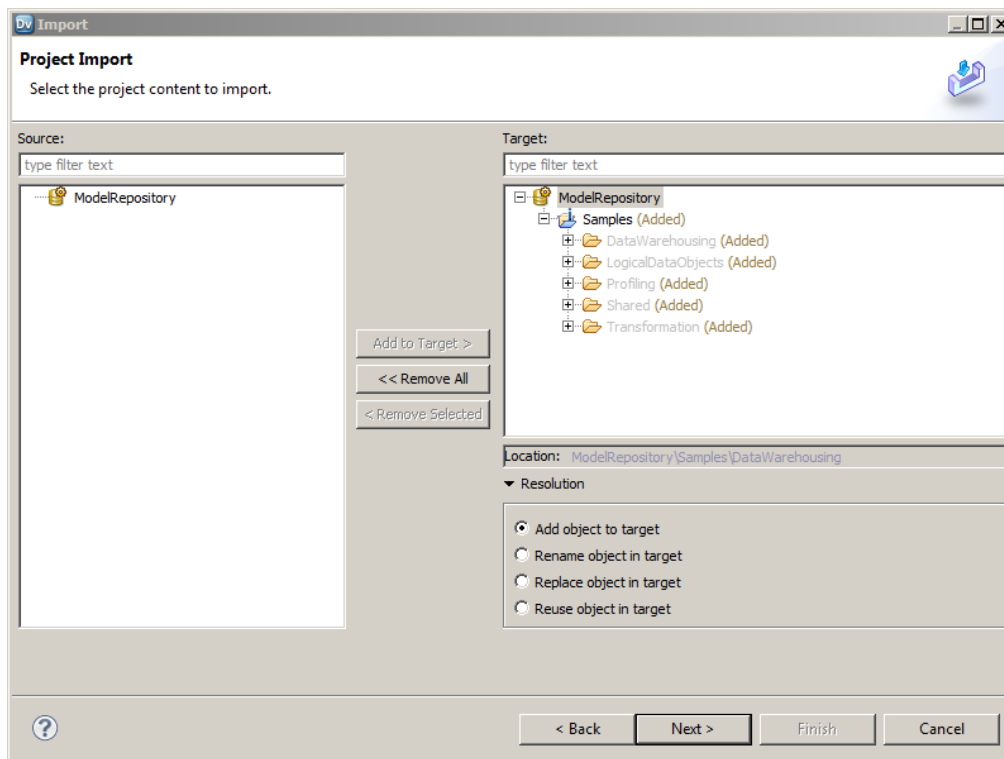
1. Click **File > Import**.
2. Select **Informatica > Import Object Metadata File (Basic)**.
3. Click **Next**.
4. Click **Browse** and select the export file that you want to import.
5. Click **Next**.
6. Select the project or select "<project name> Project Content" in the Source pane.
 - If you select 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.



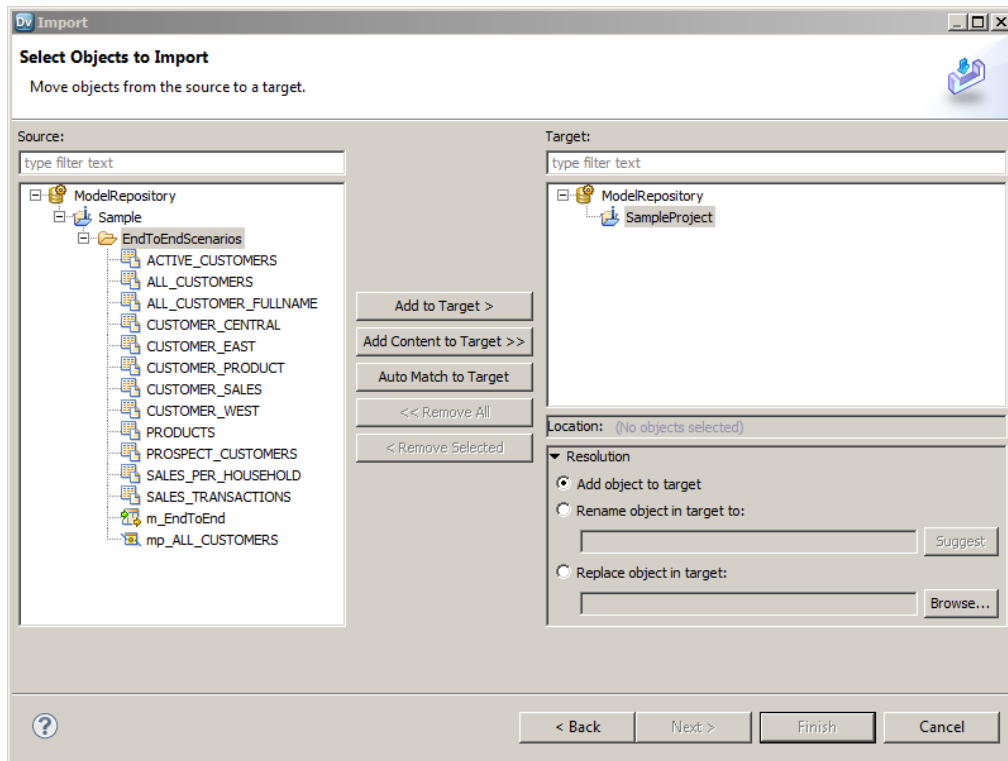
9. Click **Next**.
The Developer tool lists any reference table data that you are importing. Specify the additional reference table settings.
10. Click **Next**.
The Developer tool summarizes the objects to be imported. Click **Link Source and Target Objects** to link the objects in the Source and Target 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.
11. 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.
12. 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 lists any reference table data that you are importing. Specify the additional reference table settings.

13. 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.

14. 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.

15. 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.

Importing Objects from a Previous Informatica Release

You can upgrade objects exported from a previous Informatica release to the current metadata format, and then import the upgraded objects into the current Informatica release. You must upgrade the exported objects to the current metadata format before you import the objects into the current release.

You might import objects from a previous Informatica release into the current Informatica release in an enterprise with multiple Informatica releases installed. For example, you might export objects from a previous Informatica release development domain, and then import the objects into a current Informatica release test domain.

You export objects from an Informatica 10.1 or 10.1.1 Model repository to an .xml file that contains the object metadata. You use the `mrs UpgradeExportedObjects` command to upgrade the exported objects to the current metadata format. You must specify the path and file name of the .xml file that contains the exported objects when you run the command.

Do not modify the .xml file before you run the `mrs UpgradeExportedObjects` command. If you modify the file before you run the command, the objects in the file might not upgrade successfully.

The `mrs UpgradeExportedObjects` command upgrades the objects, and then generates an .xml file containing the upgraded objects. You must specify the file name and path for the .xml file. You can import the .xml file into a current Informatica release repository.

The `mrs UpgradeExportedObjects` command requires access to an Informatica 10.2 Model Repository Service. You must supply the service name of an Informatica 10.2 Model Repository Service running within the domain when you run the command.

For more information about the `mrs UpgradeExportedObjects` command, see the *Informatica Command Reference*.

APPENDIX A

Data Type Reference

This appendix includes the following topics:

- [Data Type Reference Overview, 211](#)
- [Transformation Data Types, 212](#)
- [Complex File and Transformation Data Types, 223](#)
- [Flat File and Transformation Data Types, 229](#)
- [DB2 for LUW and Transformation Data Types, 230](#)
- [DB2 for i5/OS, DB2 for z/OS, and Transformation Datatypes, 232](#)
- [JDBC and Transformation Datatypes, 233](#)
- [Microsoft SQL Server and Transformation Data Types, 234](#)
- [Nonrelational and Transformation Datatypes, 237](#)
- [ODBC and Transformation Data Types, 239](#)
- [Oracle and Transformation Data Types, 240](#)
- [SAP HANA and Transformation Datatypes, 243](#)
- [XML and Transformation Datatypes, 244](#)
- [Converting Data, 246](#)

Data Type 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 data type assigned to each port in a mapping.

The Developer tool displays two types of data types:

Native data types

Native data types are specific to the relational table or flat file used as a physical data object. Native data types appear in the physical data object column properties.

Transformation data types

Transformation data types are set of data types that appear in the transformations. They are internal data types based on ANSI SQL-92 generic data types, which the Data Integration Service uses to move data across platforms. Transformation data types appear in all transformations in a mapping.

Transformation data types include the following data types:

- Primitive data type. Represents a single data value in a single column position.
- Complex data type. Represents multiple data values in a single column position. Use complex data types in mappings that run on the Spark engine to process hierarchical data in complex files.

When the Data Integration Service reads source data, it converts the native data types to the comparable transformation data types before transforming the data. When the Data Integration Service writes to a target, it converts the transformation data types to the comparable native data types.

When you specify a multibyte character set, the data types allocate additional space in the database to store characters of up to three bytes.

Transformation Data Types

The following table describes the transformation data types:

Data Type	Size in Bytes	Description
Array	Unlimited number of characters.	Complex data type. You can use arrays with complex sources and targets.
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 38) 16 bytes (if precision <= 18 and high precision is on) 20 bytes (if precision > 18 and <= 28) 24 bytes (if precision > 28 and <= 38)	Decimal value with declared precision and scale. Scale must be less than or equal to precision. For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
Double	8 bytes	Double-precision floating-point numeric value. You can edit the precision and scale. The scale must be less than or equal to the precision.

Data Type	Size in Bytes	Description
Integer	4 bytes	-2,147,483,648 to 2,147,483,647 Precision of 10, scale of 0 Integer value.
Map	Unlimited number of characters.	Complex data type. You can use maps with complex sources and targets.
String	Unicode mode: (precision + 1) * 2 ASCII mode: precision + 1	1 to 104,857,600 characters Fixed-length or varying-length string.
Struct	Unlimited number of characters.	Complex data type You can use structs with complex sources and targets.
Text	Unicode mode: (precision + 1) * 2 ASCII mode: precision + 1	1 to 104,857,600 characters Fixed-length or varying-length string.
timestampWith TZ	40 bytes	Aug. 1, 1947 A.D to Dec. 31, 2040 A.D. -12:00 to +14:00 Precision of 36 and scale of 9. (precision to the nanosecond) Timestamp with Time Zone data type does not support the following time zone regions: - AFRICA_CAIRO - AFRICA_MONROVIA - EGYPT - AMERICA_MONTREAL

Integer Data Types

You can pass integer data from sources to targets and perform transformations on integer data. The transformation language supports Bigint and Integer data types.

The transformation integer data types represent exact values.

Integer Values in Calculations

When you use integer values in calculations, the Data Integration Service sometimes converts integer values to floating-point numbers before it performs the calculation. For example, to evaluate `MOD(12.00, 5)`, the Data Integration Service converts the integer value "5" to a floating-point number before it performs the division operation. The Data Integration Service converts integer values to double or decimal values depending on whether you enable high precision.

The Data Integration Service converts integer values in the following arithmetic operations:

Arithmetic Operation	High Precision Disabled	High Precision Enabled
Functions and calculations that cannot introduce decimal points. For example, integer addition, subtraction, and multiplication, and functions such as CUME, MOVINGSUM, and SUM.	No conversion ¹	Decimal
Non-scientific functions and calculations that can introduce decimal points. For example, integer division, and functions such as AVG, MEDIAN, and PERCENTILE.	Double	Decimal
All scientific functions and the EXP, LN, LOG, POWER, and SQRT functions.	Double	Double

¹ If the calculation produces a result that is out of range, the Integration Service writes a row error.

The transformation Double data type supports precision of up to 15 digits, while the Bigint data type 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:

```
POWER( BIGINTVAL, EXPVAL )
```

Before it performs the calculation, the Data Integration Service converts the inputs to the POWER function to double values. If the BIGINTVAL port contains the Bigint value 9223372036854775807, the Data 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 data type in a calculation that can produce decimal values and you enable high precision, the Data Integration Service converts the integer values to decimal values.

For transformations that support the Decimal data type with precision up to 28 digits, precision loss does not occur in a calculation unless the result produces a value with precision greater than 28 digits in high precision mode. In this case, the Data Integration Service stores the result as a double. If the port precision is less than or equal to 28 digits and the result produces a value greater than 28 digits in high precision mode, the Data Integration Service rejects the row.

For transformations that support the Decimal data type with precision up to 38 digits, precision loss does not occur in a calculation unless the result produces a value with precision greater than 38 digits in high precision mode. In this case, the Data Integration Service stores the result as a double. If the port precision is less than or equal to 38 digits and the result produces a value greater than 38 digits in high precision mode, the Data Integration Service rejects the row.

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 data type 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
NaNS
qNaN
sNaN
1.#SNAN
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

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 Data Type

If a mapping includes binary data, set the precision for the transformation binary data type so that the Integration Service can allocate enough memory to move the data from source to target.

You cannot use binary data types for flat file sources.

Date/Time Data Type

The Date/Time data type 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 data type supports dates with precision to the nanosecond. The data type has a precision of 29 and a scale of 9. Some native data types 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 data type 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.

Timestamp with Time Zone

Timestamp with Time Zone is a variant of the Timestamp data type that includes a time zone offset, TIME_ZONE_HOUR: TIME_ZONE_MINUTE, with or without daylight savings or time zone region name. The time zone offset is the difference, in hours and minutes, between the local time zone and UTC (Coordinated Universal Time).

For example, '16-JUN-08 07.01.25.376000 PM -06:00'. In the example, -06:00 is the time zone offset.

Another example is, '05-JUN-2008 07:01:25.376000 PM America/Los_Angeles'. In the example, America/Los_Angeles is the time zone region name.

When you import the Timestamp with Time Zone data type into the Developer tool, the associated transformation data type is timestampWithTZ.

timestampWithTZ has a precision of 36 and a scale of 9. Timestamp with Time Zone displacement value range is from -12:00 < UTC < +14:00.

The Data Integration Service can process Timestamp with Time Zone data type in Oracle and flat file data objects. Timestamp with Time Zone data type is applicable only for the data within the range of Aug 1947 to Dec 2040.

Timestamp with Time Zone data type does not support the following time zone regions in the Developer tool:

- AFRICA_CAIRO
- AFRICA_MONROVIA
- EGYPT
- AMERICA_MONTREAL

Timestamp with Time Zone data type uses the IANA standard for the Time Zone Database version 2015b. To avoid data corruption when using the Timestamp with Time Zone data type, ensure that the Oracle DST patch 24 is present on the Oracle server and client. When you enable data object caching, ensure that the Oracle DST patch 24 is present on the database server used for data object caching.

If the patch is not present, download the Oracle DST patch 24 from the Oracle website. To verify that the Oracle server uses the Oracle DST patch 24, run the following command in the command prompt at the Oracle server:

```
SELECT VERSION FROM v$timezone_file;
```

You can verify that the version appears as 24. You can also verify the version when you run the following command at the Oracle server:

```
SELECT TZ_VERSION FROM registry$database;
```

You can verify that the TZ_VERSION appears as 24.

Timestamp with Local Time Zone

Timestamp with Local Time Zone is a variant of the Timestamp data type where the timestamp data is normalized to the database time zone. The time zone displacement is not part of the column data. When the Data Integration Service reads the data, Oracle returns the data in the time zone of the Data Integration Service.

When you import the Timestamp with Local Time Zone data type into the Developer tool, the associated transformation data type is date/time.

For example, '04-APR-10 10.27.451 AM'

Timestamp (9) with Local Time Zone has a precision of 29 and a scale of 9. It is mapped to the date/time (29,9) transformation data type.

To set the default session time zone when the Data Integration Service reads or writes the Timestamp with Local Time Zone data, specify the ORA_SDTZ environment variable. You can set the ORA_SDTZ environment variable to any of the following values:

- Operating system local time zone ('OS_TZ')
- Database time zone ('DB_TZ')
- Absolute offset from UTC (for example, '-05:00')
- Time zone region name (for example, 'America/Los_Angeles')

Supported Time Zones for Oracle

In the following install location, you can see that the time zones file contains the list of all supported time zones for Oracle except the four time zones of AFRICA_CAIRO, AFRICA_MONROVIA, EGYPT, and AMERICA_MONTREAL:

```
<Informatica installation directory>/services/shared/timezones/timezones.txt
```

If you want to add time zones later based on Oracle support for additional time zones, you can place a new file in the same install location that includes the new Time Zones.

Decimal and Double Data Types

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 data types:

Decimal

For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. You cannot use decimal values with a scale greater than the precision or with a negative precision. Transformations display any range that you assign to the Decimal data type, but the Data Integration Service supports precision only up to 38 digits or 28 digits depending on the transformation.

When you enable high precision and the port precision is greater than 38 digits or 28 digits, depending on the transformation, the Data Integration Service stores the result as a double.

Double

Double-precision floating-point numeric value.

You can edit the precision and scale. The scale must be less than or equal to the precision.

Decimal and Double Values in Calculations

The following table lists how the Data Integration Service handles decimal values based on high precision configuration:

Port Data Type	Precision	High Precision Off	High Precision On
Decimal	0 to 15	Decimal	Decimal
Decimal	15 to 38 for transformations that support the Decimal data type with precision up to 38 digits. 15 to 28 for transformations that support the Decimal data type with precision up to 28 digits.	Double	Decimal
Decimal	Over 38 for transformations that support the Decimal data type with precision up to 38 digits. Over 28 for transformations that support the Decimal data type with precision up to 28 digits.	Double	Double

When you enable high precision, the Data Integration Service converts numeric constants in any expression function to Decimal. If you do not enable high precision, the Data Integration Service converts numeric constants to Double.

You can ensure the maximum precision for numeric values greater than 28 or 38 digits depending on the transformation. Before you perform any calculations or transformations with the transformation functions, truncate or round any large numbers.

For transformations that support Decimal data type of precision up to 38 digits, use the Decimal data type and enable high precision to ensure precision of up to 38 digits.

Do not use the Double data type for data that you use in an equality condition, such as a lookup or join condition.

Precision Loss for Decimal and Double Values

Precision loss can occur with decimal and double data types in a calculation when the result produces a value with a precision greater than the maximum allowed digits.

If the precision is greater than the maximum allowed digits, the Data Integration Service stores the result as a double.

Precision loss can also occur during intermediate calculations if there is a change in the precision or scale of intermediate results. The precision and scale of intermediate results can affect the precision or scale of the final result.

Precision Loss when High Precision is Disabled

If you disable high precision, the Data Integration Service converts decimal values to double. 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 Data Integration Service converts the decimal value to double and passes $4.00120303049577 \times 10^{19}$.

Additional precision loss can occur if you use expressions in transformations that have intermediate calculations. If you disable high precision, the Data Integration Service converts the intermediate result to double and uses the intermediate result with lower precision to perform the following calculation in the expression. As a result, the Data Integration Service might produce a different output for expressions based on the order of the arguments. For example, when you disable high precision, the output for $\tan(90 * \pi / 180)$ is different from $\tan(90 / 180 * \pi)$. The Data Integration Service performs the intermediate calculations in these expressions as follows:

```
tan(90*$pi/180) = tan(1.5707963267950003) = -9.649380295141232E12
tan(90/180*$pi) = tan(1.570796326795) = -9.670099380792184E12
```

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 Data Types

The transformation data types include the following string data types:

- String
- Text

Although the String and Text data types 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 data types, such as Char and Varchar, display as String in transformations, while the larger text data types, such as Text, Long, and Long Varchar, display as Text.

Use String and Text interchangeably within transformations. However, in Lookup transformations, the target data types must match. The database drivers need to match the string data types with the transformation data types, so that the data passes accurately. For example, Varchar in a lookup table must match String in the Lookup transformation.

Complex Data Types

Use complex data types to represent multiple data values in a single row or column position in a transformation. You use complex data types to enable mappings that run on the Spark engine to directly read, process, and write hierarchical data.

You assign complex data types to complex ports in a mapping to process hierarchical data. You call ports that process hierarchical data complex ports. You specify the complex data type to use to process data from a complex port in the port type configuration in the Developer tool.

You can use the following complex data types in transformations in mappings that run on the Spark engine:

array

An array is an ordered collection of elements. The elements can be primitive data types such as integers or strings, or complex data types such as arrays, structs, or maps. All elements in the array must be of the same data type.

map

A map contains an unordered collection of key-value pairs. The value part can be a primitive data type, or a complex data type. The value can only be of a single data type.

struct

A struct is a collection of elements of different data types. A struct data type is conceptually similar to a table row. The data type contains a fixed number of named fields, each with a predefined data type.

Complex data types can contain primitive data types or complex data types. There are no restrictions on the number of characters contained within a complex data type.

You can use a nested data type in a complex port. A nested data type is a complex data type that contains other complex data types. For example, you can create an array of structs, or a struct containing an array of other structs. A nested data type can contain up to 10 levels of nesting.

Array Data Type

An array data type represents an ordered collection of elements. To pass, generate, or process array data, assign array data type to ports.

An array is a zero-based indexed list. An array index indicates the position of the array element. For example, the array index 0 indicates the first element in an array. The transformation language includes operators to access array elements and functions to generate and process array data.

An array can be one-dimensional or multidimensional. A one-dimensional array is a linear array. A multidimensional array is an array of arrays. Array transformation data types can have up to five dimensions.

Format

```
array <data_type> []
```

The following table describes the arguments for this data type:

Argument	Description
array	Name of the array column or port.
data_type	Data type of the elements in an array. The elements can be primitive data types or complex data types. All elements in the array must be of the same data type.
[]	Dimension of the array represented as subscript. A single subscript [] represents a one-dimensional array. Two subscripts [][] represent a two-dimensional array. Elements in each dimension are of the same data type.

The elements of an array do not have names. The number of elements in an array can be different for each row.

Array Examples

One-dimensional array

The following array column represents a one-dimensional array of string elements that contains customer phone numbers:

```
custphone string[]
```

The following example shows data values for the custphone column:

```
custphone
```

```
[205-128-6478,722-515-2889]
```

```
[107-081-0961,718-051-8116]
```

```
[107-031-0961,NULL]
```

Two-dimensional array

The following array column represents a two-dimensional array of string elements that contains customer work and personal email addresses.

```
email_work_pers string[][]
```

The following example shows data values for the email_work_pers column:

```
email_work_pers
```

```
[john_baer@xyz.com,jbaer@xyz.com][john.baer@fgh.com,jbaer@ijk.com]
```

```
[bobbi_apperley@xyz.com,bapperl@xyz.com][apperlbob@fgh.com,bobbi@ijk.com]
```

```
[linda_bender@xyz.com,lbender@xyz.com][l.bender@fgh.com,NULL]
```

Map Data Type

A map data type represents an unordered collection of key-value pair elements. A map element is a key and value pair that maps one thing to another. To pass, generate, or process map data, assign map data type to ports.

The key must be of a primitive data type. The value can be of a primitive or complex data type. A map data type with values of a complex data type is a nested map. A nested map can contain up to three levels of nesting of map data type.

Note: The Spark engine allows duplicate keys in the map data.

The transformation language includes subscript operator to access map elements. It also includes functions to generate and process map data.

Format

```
map <primitive_type -> data_type>
```

The following table describes the arguments for this data type:

Argument	Description
map	Name of the map column or port.
primitive_type	Data type of the key in a map element. The key must be of a primitive data type.
data_type	Data type of the value in a map element. The value can be of a primitive or complex data type.

Map Example

The following map column represents map data with an integer key and a string value to map customer ids with customer names:

```
custid_name <integer -> string>
```

The following example shows data values for the custid_name column:

```
custid_name
```

```
<26745 -> 'John Baer'>
```

```
<56743 -> 'Bobbi Apperley'>
```

```
<32879 -> 'Linda Bender'>
```

Struct Data Type

A struct data type represents a collection of elements of different data types. A struct data type has an associated schema that defines the structure of the data. To pass, generate, or process struct data, assign the struct data type to ports.

The schema for the struct data type determines the element names and the number of elements in the struct data. The schema also determines the order of elements in the struct data. Informatica uses complex data type definitions to represent the schema of struct data.

The transformation language includes operators to access struct elements. It also includes functions to generate, process, and parse struct data and to modify the schema of the data.

Format

```
struct {element_name1:value1 [, element_name2:value2, ...]}
```

Schema for the struct is of the following format:

```
schema {element_name1:data_type1 [, element_name2:data_type2, ...]}
```

The following table describes the arguments for this data type:

Argument	Description
struct	Name of the struct column or port.
schema	A definition of the structure of data. Schema is a name-type pair that determines the name and data type of the struct elements.
element_name	Name of the struct element.
value	Value of the struct element.
data_type	Data type of the element value. The element values can be of a primitive or complex data type. Each element in the struct can have a different data type.

Struct Example

The following schema is for struct data to store customer addresses:

```
address  
{st_number:integer,st_name:string,city:string,state:string,zip:string}
```

The following example shows struct data values for the cust_address column:

cust_address

```
{st_number:154,st_name:Addison Ave,city:Redwood City,state:CA,zip:94065}  
  
{st_number:204,st_name:Ellis St,city:Mountain View,state:CA,zip:94043}  
  
{st_number:357,st_name:First St,city:Sunnyvale,state:CA,zip:94085}
```

Complex File and Transformation Data Types

You can use complex data types in mappings to process hierarchical data in complex files.

You can use complex data types in the following complex files in mappings that run on the Spark and Databricks Spark engines:

- Avro
- JavaScript Object Notation (JSON)
- Optimized Row Columnar (ORC)

- Parquet

Avro Data Types and Transformation Data Types

Avro data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares the Avro data types that the Data Integration Service supports and the corresponding transformation data types:

Avro Data Type	Transformation Data Type	Range
Array	Array	Unlimited number of characters.
Boolean	Integer	TRUE (1) or FALSE (0).
Bytes	Binary	Precision 4000.
Date	Date/Time	January 1, 0001 to December 31, 9999.
Decimal	Decimal	Decimal value with declared precision and scale. Scale must be less than or equal to precision. For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
Double	Double	Precision 15.
Fixed	Binary	1 to 104,857,600 bytes.
Float	Double	Precision 15.
Int	Integer	-2,147,483,648 to 2,147,483,647 Precision 10 and scale 0.
Long	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807. Precision 19 and scale 0.
Map	Map	Unlimited number of characters.
Record	Struct	Unlimited number of characters.
String	String	1 to 104,857,600 characters.
Time	Date/Time	Time of the day. Precision to microsecond.

Avro Data Type	Transformation Data Type	Range
Timestamp	Date/Time	January 1, 0001 00:00:00 to December 31, 9999 23:59:59.997. Precision to microsecond.
Union	Corresponding data type in a union of ["primitive_type complex_type", "null"] or ["null", "primitive_type complex_type"].	Dependent on primitive or complex data type.

Avro Union Data Type

A union indicates that a field might have more than one data type. For example, a union might indicate that a field can be a string or a null. A union is represented as a JSON array containing the data types.

The Developer tool only interprets a union of ["primitive_type|complex_type", "null"] or ["null", "primitive_type|complex_type"]. The Avro data type converts to the corresponding transformation data type.

Avro Timestamp Data Type Support

The following table lists the Timestamp data type support for Avro file formats:

Timestamp Data type	Native	Spark
Timestamp_micros	Yes	Yes
Timestamp_millis	Yes	No
Time_millis	Yes	No
Time_micros	Yes	No

Unsupported Avro Data Types

The Developer tool does not support the following Avro data types:

- Enum
- Null
- Timestamp_tz

JSON Data Types and Transformation Data Types

JSON data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares the JSON data types that the Data Integration Service supports and the corresponding transformation data types:

JSON	Transformation	Range
Array	Array	Unlimited number of characters.
Double	Double	Precision of 15 digits.
Integer	Integer	-2,147,483,648 to 2,147,483,647. Precision of 10, scale of 0.
Object	Struct	Unlimited number of characters.
String	String	1 to 104,857,600 characters.

Unsupported JSON Data Types

The Developer tool does not support the following JSON data types:

- Date
- Decimal
- Timestamp
- Enum
- Union

ORC Data Types and Transformation Data Types

ORC file data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table lists the ORC file data types that the Data Integration Service supports and the corresponding transformation data types:

ORC File Data Type	Transformation Data Type	Range and Description
BigInt	BigInt	-9223372036854775808 to 9,223,372,036,854,775,807.
Boolean	Integer	TRUE (1) or FALSE (0).
Char	String	1 to 104,857,600 characters.
Date	Date/Time	January 1, 0001 to December 31, 9999.
Double	Double	Precision of 15 digits.

ORC File Data Type	Transformation Data Type	Range and Description
Float	Double	Precision of 15 digits.
Integer	Integer	-2,147,483,648 to 2,147,483,647.
SmallInt	Integer	-32,768 to 32,767.
String	String	1 to 104,857,600 characters.
Timestamp	Date/Time	January 1, 0001 00:00:00 to December 31, 9999 23:59:59.997. Precision to microsecond.
TinyInt	Integer	-128 to 127.
Varchar	String	1 to 104,857,600 characters.

When you run a mapping on the Spark or Databricks Spark engine to write an ORC file to a target, the Data Integration Service writes the data of the Char and Varchar data types as String.

Note: You can use ORC data types to read and write complex file objects in mappings that run on the Spark engine only.

Unsupported ORC Data Types

The Developer tool does not support the following ORC data types:

- Map
- List
- Struct
- Union

Parquet Data Types and Transformation Data Types

Parquet data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares the Parquet data types that the Data Integration Service supports and the corresponding transformation data types:

Parquet	Transformation	Range
Binary	Binary	1 to 104,857,600 bytes
Binary (UTF8)	String	1 to 104,857,600 characters
Boolean	Integer	-2,147,483,648 to 2,147,483,647 Precision of 10, scale of 0
Date	Date/Time	January 1, 0001 to December 31, 9999.

Parquet	Transformation	Range
Decimal	Decimal	<p>Decimal value with declared precision and scale. Scale must be less than or equal to precision.</p> <p>For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38.</p> <p>For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28.</p> <p>If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.</p>
Double	Double	Precision of 15 digits.
Float	Double	Precision of 15 digits.
Int32	Integer	-2,147,483,648 to 2,147,483,647 Precision of 10, scale of 0
Int64	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision of 19, scale of 0
Map	Map	Unlimited number of characters.
Struct	Struct	Unlimited number of characters.
Time	Date/Time	Time of the day. Precision to microsecond.
Timestamp	Date/Time	January 1, 0001 00:00:00 to December 31, 9999 23:59:59.997. Precision to microsecond.
group (LIST)	Array	Unlimited number of characters.

The Parquet schema that you specify to read or write a Parquet file must be in smaller case. Parquet does not support case-sensitive schema.

Parquet Timestamp Data Type Support

The following table lists the Timestamp data type support for Parquet file formats:

Timestamp Data type	Native	Spark
Timestamp_micros	Yes	No
Timestamp_millis	Yes	No
Time_millis	Yes	No

Timestamp Data type	Native	Spark
Time_micros	Yes	No
int96	Yes	Yes

Unsupported Parquet Data Types

The Developer tool does not support the following Parquet data types:

- Timestamp_nanos
- Time_nanos
- Timestamp_tz

Flat File and Transformation Data Types

Flat file data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares flat file data types to transformation data types:

Flat File Data type	Transformation Data type	Range
Bigint	Bigint	Precision of 19 digits, scale of 0
Datetime	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Double	Double	Precision of 15 digits
Int	Integer	-2,147,483,648 to 2,147,483,647
Nstring	String	1 to 104,857,600 characters
Number	Decimal	<p>For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38.</p> <p>For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28.</p> <p>If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.</p> <p>If the precision is greater than 15, the Data Integration Service converts decimal values to double in low-precision mode.</p>

Flat File Data type	Transformation Data type	Range
String	String	1 to 104,857,600 characters
TimestampWithTZ	TimestampWithTZ	Aug. 1, 1947 A.D to Dec. 31, 2040 A.D. -12:00 to +14:00 Precision of 36 and scale of 9. (precision to the nanosecond) Timestamp with Time Zone data type does not support the following time zone regions: - AFRICA_CAIRO - AFRICA_MONROVIA - EGYPT - AMERICA_MONTREAL Note: TimestampWithTZ is applicable only to the native environment.

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 Data Types

DB2 for LUW data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares DB2 for LUW data types and transformation data types:

Data Type	Range	Transformation	Range
Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
Blob	1 to 2,147,483,647 bytes	Binary	1 to 104,857,600 bytes
Char	1 to 254 characters	String	1 to 104,857,600 characters
Char for bit data	1 to 254 bytes	Binary	1 to 104,857,600 bytes
Clob	1 to 2,447,483,647 bytes	Text	1 to 104,857,600 characters
Date	0001 to 9999 A.D. Precision 19; scale 0 (precision to the day)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Dbclob	1 to 1073741823 bytes	Dbclob	1 to 104,857,600 bytes
Decimal	Precision 1 to 31, scale 0 to 31	Decimal	Precision 1 to 31, scale 0 to 31

Data Type	Range	Transformation	Range
Float	Precision 1 to 15	Double	Precision 15
Graphic	1 to 127 bytes	Graphic	1 to 127 bytes
Integer	-2,147,483,648 to 2,147,483,647	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Long Varchar	32700 bytes	Long Varchar	32700 bytes
Long Vargraphic	163350 bytes	Long Vargraphic	163350 bytes
Numeric	Precision 1 to 31, scale 0 to 31	Decimal	Precision 1 to 28, scale 0 to 28
Smallint	-32,768 to 32,767	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Time ¹	24-hour time period Precision 19, scale 0 (precision to the second)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Timestamp	26 bytes Precision 26, scale 6 (precision to the microsecond)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Varchar	Up to 4,000 characters	String	1 to 104,857,600 characters
Varchar for bit data	Up to 4,000 bytes	Binary	1 to 104,857,600 bytes
Vargraphic	1 to 16336 bytes	Vargraphic	1 to 16336 bytes
1. When the Data Integration Service converts the DB2 Time data type to the transformation Date/Time data type, the date value is added to the time value.			

DB2 for i5/OS, DB2 for z/OS, and Transformation Datatypes

DB2 for i5/OS and DB2 for z/OS datatypes map to transformation datatypes in the same way that IBM DB2 datatypes map to transformation datatypes. The Data Integration Service uses transformation datatypes to move data across platforms.

The following table compares DB2 for i5/OS and DB2 for z/OS datatypes with transformation datatypes:

Datatype	Range	Transformation	Range
Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
Char	1 to 254 characters	String	1 to 104,857,600 characters
Char for bit data	1 to 254 bytes	Binary	1 to 104,857,600 bytes
Date	0001 to 9999 A.D. Precision 19; scale 0 (precision to the day)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Decimal	Precision 1 to 31, scale 0 to 31	Decimal	Precision 1 to 31, scale 0 to 31.
Float	Precision 1 to 15	Double	Precision 15
Integer	-2,147,483,648 to 2,147,483,647	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Smallint	-32,768 to 32,767	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Time ¹	24-hour time period Precision 19, scale 0 (precision to the second)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Timestamp ²	26 bytes Precision 26, scale 6 (precision to the microsecond)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Varchar	Up to 4,000 characters	String	1 to 104,857,600 characters
Varchar for bit data	Up to 4,000 bytes	Binary	1 to 104,857,600 bytes
<p>1. When the Data Integration Service converts the DB2 Time data type to the transformation Date/Time data type, the date value is added to the time value.</p>			
<p>2. DB2 for z/OS Version 10 extended-precision timestamps map to transformation datatypes as follows:</p> <ul style="list-style-type: none"> - If scale=6, then precision=26 and transformation datatype=date/time - If scale=0, then precision=19 and transformation datatype=string - If scale=1-5 or 7-12, then precision=20+scale and transformation datatype=string 			

Unsupported DB2 for i5/OS and DB2 for z/OS Datatypes

The Developer tool does not support certain DB2 for i5/OS and DB2 for z/OS datatypes.

The Developer tool does not support DB2 for i5/OS and DB2 for z/OS large object (LOB) datatypes. LOB columns appear as unsupported in the relational table object, with a native type of varchar and a precision and scale of 0. The columns are not projected to customized data objects or outputs in a mapping.

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:

JDBC Datatype	Transformation	Range
Bigint	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
Binary*	Binary	1 to 104,857,600 bytes
Bit	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Blob*	Binary	1 to 104,857,600 bytes
Boolean	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Char*	String	1 to 104,857,600 characters
Clob*	Text	1 to 104,857,600 characters
Date	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Decimal	Decimal	Precision 1 to 28, scale 0 to 28
Double	Double	Precision 15
Float	Double	Precision 15
Integer	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Long VarBinary*	Binary	1 to 104,857,600 bytes
Long Varchar*	Text	1 to 104,857,600 characters
Numeric	Decimal	Precision 1 to 28, scale 0 to 28

JDBC Datatype	Transformation	Range
Real	Double	Precision 15
Smallint	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Time	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Timestamp	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Tinyint	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Varchar*	String	1 to 104,857,600 characters
Varbinary*	Binary	1 to 104,857,600 bytes
*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:

Microsoft SQL Server	Range	Transformation	Range
Binary	1 to 8,000 bytes	Binary	1 to 104,857,600 bytes
bigint	- 9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
Bit	1 bit	String	1 to 104,857,600 characters
Char	1 to 8,000 characters	String	1 to 104,857,600 characters
Date	Jan 1, 0001 A.D. to Dec 31, 9999 A.D.	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D.

Microsoft SQL Server	Range	Transformation	Range
Datetime	Jan 1, 1753 A.D. to Dec 31, 9999 A.D. Precision 23, scale 3 (precision to 3.33 milliseconds)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Datetime2	Jan 1, 0001 A.D. 00:00:00 to Dec 31, 9999 A.D. 23:59:59.9999999	Timestamp	Precision 22 to 27
Decimal	Precision 1 to 38, scale 0 to 38	Decimal	For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
Float	-1.79E+308 to 1.79E+308	Double	Precision 15
Image	1 to 2,147,483,647 bytes	Binary	1 to 104,857,600 bytes
Int	-2,147,483,648 to 2,147,483,647	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Money	-922,337,203,685,477.5807 to 922,337,203,685,477.5807	Decimal	Precision 1 to 28, scale 0 to 28
nchar	1 to 4000 characters	String	1 to 104,857,600 characters
ntext	1 to 1,073,741,823 bytes	Text	1 to 104,857,600 characters
Numeric	Precision 1 to 38, scale 0 to 38	Decimal	For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
nvarchar	1 to 4000 characters	String	1 to 104,857,600 characters
Real	-3.40E+38 to 3.40E+38	Double	Precision 15

Microsoft SQL Server	Range	Transformation	Range
Smalldatetime	Jan 1, 1900, to June 6, 2079 Precision 19, scale 0 (precision to the minute)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Smallint	-32,768 to 32,768	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Smallmoney	-214,748.3648 to 214,748.3647	Decimal	Precision 1 to 28, scale 0 to 28
Sysname	1 to 128 characters	String	1 to 104,857,600 characters
Text	1 to 2,147,483,647 characters	Text	1 to 104,857,600 characters
Timestamp	8 bytes	Binary	1 to 104,857,600 bytes
Tinyint	0 to 255	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Uniqueidentifier	Precision 38, scale 0	String	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.
Varbinary	1 to 8,000 bytes	Binary	1 to 104,857,600 bytes
Varchar	1 to 8,000 characters	String	1 to 104,857,600 characters

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.

Nonrelational and Transformation Datatypes

Nonrelational datatypes map to transformation datatypes that the Data Integration Service uses to move data across platforms.

Nonrelational datatypes apply to the following types of connections:

- Adabas
- IMS
- Sequential
- VSAM

The following table compares nonrelational datatypes and transformation datatypes:

Nonrelational	Precision	Transformation	Range
BIN	10	Binary	1 to 104,857,600 bytes
CHAR	10	String	1 to 104,857,600 characters Fixed-length or varying-length string.
DATE	10	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. Combined date/time value, with precision to the nanosecond.
DOUBLE	18	Double	Precision of 15 digits Double-precision floating-point numeric value.
FLOAT	7	Double	Precision of 15 digits Double-precision floating-point numeric value.
NUM8	3	Integer	Precision of 10 and scale of 0 Integer value.
NUM8U	3	Integer	Precision of 10 and scale of 0 Integer value.
NUM16	5	Integer	Precision of 10 and scale of 0 Integer value.
NUM16U	5	Integer	Precision of 10 and scale of 0 Integer value.
NUM32	10	Integer	Precision of 10 and scale of 0 Integer value.
NUM32U	10	Double	Precision of 15 digits Double-precision floating-point numeric value.
NUM64	19	Decimal	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. If you pass a value with negative scale or declared precision greater than 28, the Data Integration Service converts it to a double.

Nonrelational	Precision	Transformation	Range
NUM64U	19	Decimal	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. If you pass a value with negative scale or declared precision greater than 28, the Data Integration Service converts it to a double.
NUMCHAR	100	String	1 to 104,857,600 characters Fixed-length or varying-length string.
PACKED	15	Decimal	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. If you pass a value with negative scale or declared precision greater than 28, the Data Integration Service converts it to a double.
TIME	5	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. Combined date/time value, with precision to the nanosecond.
TIMESTAMP	5	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. Combined date/time value, with precision to the nanosecond.
UNPACKED	15	Decimal	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. If you pass a value with negative scale or declared precision greater than 28, the Data Integration Service converts it to a double.
UZONED	15	Decimal	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. If you pass a value with negative scale or declared precision greater than 28, the Data Integration Service converts it to a double.
VARBIN	10	Binary	1 to 104,857,600 bytes
VARCHAR	10	String	1 to 104,857,600 characters Fixed-length or varying-length string.
ZONED	15	Decimal	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. If you pass a value with negative scale or declared precision greater than 28, the Data Integration Service converts it to a double.

ODBC and Transformation Data Types

ODBC data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares ODBC data types, such as Microsoft Access or Excel, to transformation data types:

Data Type	Transformation	Range
Bigint	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
Binary	Binary	1 to 104,857,600 bytes
Bit	String	1 to 104,857,600 characters
Char	String	1 to 104,857,600 characters
Date	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Decimal	Decimal	For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
Double	Double	Precision 15
Float	Double	Precision 15
Integer	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Long Varbinary	Binary	1 to 104,857,600 bytes
Nchar	String	1 to 104,857,600 characters
Nvarchar	String	1 to 104,857,600 characters
Ntext	Text	1 to 104,857,600 characters
Numeric	Decimal	For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
Real	Double	Precision 15
Smallint	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Text	Text	1 to 104,857,600 characters

Data Type	Transformation	Range
Time	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Timestamp	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Tinyint	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
Vbinary	Binary	1 to 104,857,600 bytes
Varchar	String	1 to 104,857,600 characters

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:

Oracle	Range	Transformation	Range
Blob	Up to 4 GB	Binary	1 to 104,857,600 bytes
Char(L)	1 to 2,000 bytes	String	1 to 104,857,600 characters
Clob	Up to 4 GB	Text	1 to 104,857,600 characters
Date	Jan. 1, 4712 B.C. to Dec. 31, 4712 A.D. Precision 19, scale 0	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Float	Precision of 1 to 15, scale 0	Double	Precision of 15, scale 0
Long	Up to 2 GB	Text	1 to 104,857,600 characters 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.

Oracle	Range	Transformation	Range
Long Raw	Up to 2 GB	Binary	1 to 104,857,600 bytes
Nchar	1 to 2,000 bytes	String	1 to 104,857,600 characters
Nclob	Up to 4 GB	Text	1 to 104,857,600 characters
Number	Precision of 1 to 38	Double	Precision of 15
Number(P,S)	Precision of 1 to 38, scale of 0 to 38	Decimal	For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
Nvarchar2	1 to 4,000 bytes	String	1 to 104,857,600 characters
Raw	1 to 2,000 bytes	Binary	1 to 104,857,600 bytes
Timestamp	Jan. 1, 4712 B.C. to Dec. 31, 9999 A.D. Precision 19 to 29, scale 0 to 9 (precision to the nanosecond)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)

Oracle	Range	Transformation	Range
Timestamp with Time Zone	Jan. 1, 4712 B.C. to Dec. 31, 9999 A.D. -12:00 to +14:00 Precision of 26 to 36, scale 0 to 9. (precision to the nanosecond)	timestampWithTZ	Aug. 1, 1947 A.D to Dec. 31, 2040 A.D. -12:00 to +14:00 Precision of 36 and scale of 9. (precision to the nanosecond) Timestamp with Time Zone data type does not support the following time zone regions: - AFRICA_CAIRO - AFRICA_MONROVIA - EGYPT - AMERICA_MONTREAL
Timestamp with Local Time Zone	Jan. 1, 4712 B.C. to Dec. 31, 9999 A.D. Precision 19 to 29, scale 0 to 9 (precision to the nanosecond)	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Varchar	1 to 4,000 bytes	String	1 to 104,857,600 characters
Varchar2	1 to 4,000 bytes	String	1 to 104,857,600 characters
XMLType	Up to 4 GB	Text	1 to 104,857,600 characters

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 38 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

SAP HANA and Transformation Datatypes

SAP HANA datatypes map to transformation datatypes that the Data Integration Service uses to move data across platforms.

The following table compares SAP HANA datatypes and transformation datatypes:

SAP HANA Datatype	Range	Transformation Datatype	Range
Alphanum	Precision 1 to 127	Nstring	1 to 104,857,600 characters
Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
Binary	Used to store bytes of binary data	Binary	1 to 104,857,600 bytes
Blob	Up to 2 GB	Binary	1 to 104,857,600 bytes
Clob	Up to 2 GB	Text	1 to 104,857,600 characters
Date	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. Precision 10, scale 0	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Decimal (precision, scale) or Dec (p, s)	Precision 1 to 34	Decimal	For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.

SAP HANA Datatype	Range	Transformation Datatype	Range
Double	Specifies a single-precision 64-bit floating-point number	Double	Precision 15
Float	Precision 1 to 53	Double	Precision 15
Integer	-2,147,483,648 to 2,147,483,647	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
NClob	Up to 2 GB	Ntext	1 to 104,857,600 characters
Nvarchar	Precision 1 to 5000	Nstring	1 to 104,857,600 characters
Real	Specifies a single-precision 32-bit floating-point number	Real	Precision 7, scale 0
Seconddate	0001-01-01 00:00:01 to 9999-12-31 24:00:00	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Shorttext	Specifies a variable-length character string, which supports text search and string search features	Nstring	1 to 104,857,600 characters
Smalldecimal	Precision 1 to 16	Decimal	Precision 1 to 28, scale 0 to 28
Smallint	-32,768 to 32,767	Small Integer	Precision 5, scale 0
Text	Specifies a variable-length character string, which supports text search features	Text	1 to 104,857,600 characters
Time	24-hour time period	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Timestamp	0001-01-01 00:00:00.0000000 to 9999-12-31 23:59:59.9999999	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
Tinyint	0 to 255	Small Integer	Precision 5, scale 0
Varchar	Precision 1 to 5000	String	1 to 104,857,600 characters
Varbinary	1 to 5000 bytes	Binary	1 to 104,857,600 bytes

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>.

The following table compares XML datatypes to transformation datatypes:

Datatype	Transformation	Range
anyURI	String	1 to 104,857,600 characters
base64Binary	Binary	1 to 104,857,600 bytes
boolean	String	1 to 104,857,600 characters
byte	Integer	-2,147,483,648 to 2,147,483,647
date	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
dateTime	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
decimal	Decimal	Precision 1 to 28, scale 0 to 28
double	Double	Precision of 15 digits
duration	String	1 to 104,857,600 characters
ENTITIES	String	1 to 104,857,600 characters
ENTITY	String	1 to 104,857,600 characters
float	Double	Precision of 15 digits
gDay	String	1 to 104,857,600 characters
gMonth	String	1 to 104,857,600 characters
gMonthDay	String	1 to 104,857,600 characters
gYear	String	1 to 104,857,600 characters
gYearMonth	String	1 to 104,857,600 characters
hexBinary	Binary	1 to 104,857,600 bytes
ID	String	1 to 104,857,600 characters
IDREF	String	1 to 104,857,600 characters
IDREFS	String	1 to 104,857,600 characters
int	Integer	-2,147,483,648 to 2,147,483,647
integer	Integer	-2,147,483,648 to 2,147,483,647
language	String	1 to 104,857,600 characters
long	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

Datatype	Transformation	Range
Name	String	1 to 104,857,600 characters
NCName	String	1 to 104,857,600 characters
negativeInteger	Integer	-2,147,483,648 to 2,147,483,647
NMTOKEN	String	1 to 104,857,600 characters
NMTOKENS	String	1 to 104,857,600 characters
nonNegativeInteger	Integer	-2,147,483,648 to 2,147,483,647
nonPositiveInteger	Integer	-2,147,483,648 to 2,147,483,647
normalizedString	String	1 to 104,857,600 characters
NOTATION	String	1 to 104,857,600 characters
positiveInteger	Integer	-2,147,483,648 to 2,147,483,647
QName	String	1 to 104,857,600 characters
short	Integer	-2,147,483,648 to 2,147,483,647
string	String	1 to 104,857,600 characters
time	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
token	String	1 to 104,857,600 characters
unsignedByte	Integer	-2,147,483,648 to 2,147,483,647
unsignedInt	Integer	-2,147,483,648 to 2,147,483,647
unsignedLong	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
unsignedShort	Integer	-2,147,483,648 to 2,147,483,647

Converting Data

You can convert data from one data type to another.

To convert data from one data type to another, use one the following methods:

- Pass data between ports (port-to-port data 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 data type of the port. Each time data passes through a port, the Data Integration Service identifies the data type assigned to the port and converts the data based on the supported data type if required.

Pass Data Between Ports of the Same Data Type

You can create a mapping to pass data between ports of the same data type. When you pass data between ports of the same data type, the Data Integration Service passes data without changing the data or data type.

When you pass data between ports of the same numeric data type and the data is transferred between transformations, the Data Integration Service does not always convert the data to the precision and scale of the port that the data is passed to.

You can transfer Decimal data between ports of different precision, scale, and precision modes in the following ways:

Pass Decimal data in low-precision mode

If you pass data to a Decimal port in low-precision mode, the Data Integration Service converts all Decimal ports to Double with a precision of 15 and a scale of 0. For example, you pass a value from Decimal (14, 3) to Decimal (9, 5) in low-precision mode. The Data Integration Service stores the value internally and does not truncate the data in low-precision mode.

Pass Decimal data with reduced scale in high-precision mode

If you pass data to a Decimal port in high-precision mode and with reduced scale between the Decimal ports, scale truncation occurs. For example, you pass a value from Decimal (18, 5) to Decimal (18, 2). When you pass 18.01234, the Data Integration Service truncates the scale of the data and the output data is 18.01.

Pass Decimal data with reduced precision in high-precision mode

You can pass data to a Decimal port in high-precision mode with reduced precision. For example, you pass a value from Decimal (19, 5) to a Decimal (17, 2) in high-precision mode. When the output field contains a value that exceeds 17 digits, the Data Integration Service rejects the row.

Pass Data Between Ports of Different Data Types

When you pass data between ports of different data types, the Data Integration Service uses the conversion functions in the transformation language to convert the data from one data type to another.

For example, you connect a String port to an Integer port. When the Data Integration Service runs the mapping, it uses the TO_INTEGER function to convert the input data from a string to an integer data type.

When the Data Integration Service performs port-to-port conversions, the data that you pass must be valid for the conversion data type. Any value that the Data Integration Service cannot convert results in a transformation row error. For example, you connect a String port that contains the value "9,000,000,000,000,000.777" 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 pipeline and a target.

The following table describes the port-to-port conversions that the Data Integration Service performs:

Data Type	Bigint	Integer	Decimal	Double	String, Text	Date/Time	Binary	Timestamp with Time Zone
Bigint	No	Yes	Yes	Yes	Yes	No	No	No
Integer	Yes	No	Yes	Yes	Yes	No	No	No
Decimal	Yes	Yes	No	Yes	Yes	No	No	No
Double	Yes	Yes	Yes	No	Yes	No	No	No
String, Text	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Date/Time	No	No	No	No	Yes	Yes	No	No
Binary	No	No	No	No	No	No	Yes	No
Timestamp with Time Zone	No	No	No	No	Yes	No	No	Yes

APPENDIX B

Keyboard Shortcuts

This appendix includes the following topics:

- [Keyboard Shortcuts for Advanced Mapping Options, 249](#)
- [Keyboard Shortcuts for Objects, 250](#)
- [Keyboard Shortcuts for Ports, 251](#)
- [Keyboard Shortcuts for the Transformation Palette, 252](#)
- [Keyboard Shortcuts for the Workbench, 252](#)

Keyboard Shortcuts for Advanced Mapping Options

You can use keyboard shortcuts to navigate the Run Mapping Using Advanced Options dialog box.

You can perform the following tasks with keyboard shortcuts:

Select a mapping configuration.

Press **Shift+a**.

Specify a custom mapping configuration.

Press **Shift+c**.

Use the default Data Integration Service.

Press **Shift+u**.

Data Integration Service.

Press **Shift+i**. To browse for more options, press **Shift+b**.

Override tracing level.

Press **Shift+t**.

Optimizer level.

Press **Shift+z**.

Apply the default values in the mapping.

Press **Shift+d**.

Apply a parameter set.

Press **Shift+s**. To browse for more options, press **Shift+o**.

Apply a parameter file.

Press **Shift+f**. To browse for more options, press **Shift+o**.

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 the **Tab** button three times to select the next widget.

Select the previous area in a view.

Press **Ctrl+Shift+Tab**.

In the mapping Properties view, select and read a description.

Press **Ctrl+A**.

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 the **Ctrl+Tab** again to select the next area.

Select the previous area of the workbench.

Press **Ctrl+Shift+Tab**.

Jump to the error message in a dialog box.

Press **Ctrl+M**.

APPENDIX C

Connection Properties

This appendix includes the following topics:

- [Connection Properties Overview, 255](#)
- [Adabas Connection Properties, 256](#)
- [Amazon Redshift Connection Properties, 258](#)
- [Amazon S3 Connection Properties, 260](#)
- [Blockchain Connection Properties, 262](#)
- [Cassandra Connection Properties, 264](#)
- [Confluent Kafka Connection, 265](#)
- [Databricks Connection Properties, 267](#)
- [Greenplum Connection Properties, 269](#)
- [Google Analytics Connection Properties, 271](#)
- [Google BigQuery Connection Properties, 271](#)
- [Google Cloud Spanner Connection Properties, 273](#)
- [Google Cloud Storage Connection Properties, 274](#)
- [Google PubSub Connection Properties, 275](#)
- [Hadoop Connection Properties, 276](#)
- [HBase Connection Properties, 282](#)
- [HDFS Connection Properties, 283](#)
- [HBase Connection Properties for MapR-DB, 285](#)
- [Hive Connection Properties, 285](#)
- [HTTP Connection Properties, 289](#)
- [IBM DB2 Connection Properties, 291](#)
- [IBM DB2 for i5/OS Connection Properties, 294](#)
- [IBM DB2 for z/OS Connection Properties, 297](#)
- [IMS Connection Properties, 300](#)
- [JDBC Connection Properties, 302](#)
- [JDBC V2 Connection Properties, 305](#)
- [JD Edwards EnterpriseOne Connection Properties, 307](#)
- [Kafka Connection Properties, 308](#)
- [Kudu Connection Properties, 311](#)
- [LDAP Connection Properties, 312](#)

- [Microsoft Azure Blob Storage Connection Properties, 313](#)
- [Microsoft Azure Cosmos DB SQL API Connection Properties, 314](#)
- [Microsoft Azure Data Lake Storage Gen1 Connection Properties, 315](#)
- [Microsoft Azure Data Lake Storage Gen2 Connection Properties, 316](#)
- [Microsoft Azure SQL Data Warehouse Connection Properties, 317](#)
- [MS SQL Server Connection Properties, 319](#)
- [Netezza Connection Properties, 323](#)
- [OData Connection Properties, 324](#)
- [ODBC Connection Properties, 325](#)
- [Oracle Connection Properties, 326](#)
- [Salesforce Connection Properties, 329](#)
- [Salesforce Marketing Cloud Connection Properties, 330](#)
- [SAP Connection Properties, 331](#)
- [Sequential Connection Properties, 334](#)
- [Snowflake Connection Properties, 336](#)
- [Teradata Parallel Transporter Connection Properties, 337](#)
- [Tableau Connection Properties, 339](#)
- [Tableau V3 Connection Properties, 340](#)
- [Twitter Streaming Connection Properties, 341](#)
- [VSAM Connection Properties, 342](#)
- [Web Services Connection Properties, 345](#)
- [Identifier Properties in Database Connections, 346](#)

Connection Properties Overview

Connection properties enable the Informatica client to connect to data sources.

This chapter contains connection properties for each of the connections you can create and manage through Informatica clients.

Adabas Connection Properties

Use an Adabas connection to access an Adabas database. The Adabas connection is a mainframe database type connection. You create an Adabas connection in the Developer tool. You can manage an Adabas connection in the Administrator tool or the Developer tool.

The following table describes Adabas connection properties:

Option	Description
Location	Node name for the location of the PowerExchange Listener that connects to Adabas. The node name is defined in the first parameter of the NODE statement in the PowerExchange dbmover.cfg configuration file.
User Name	Database user name. For a database on a supported Linux or UNIX system, if you have enabled PowerExchange LDAP user authentication, the user name is the enterprise user name. For more information, see the <i>PowerExchange Reference Manual</i> .
Password	<p>Password for the database user name or a valid PowerExchange passphrase.</p> <p>A PowerExchange passphrase can be from 9 to 128 characters in length and can contain the following characters:</p> <ul style="list-style-type: none"> - Uppercase and lowercase letters - The numbers 0 to 9 - Spaces - The following special characters: ' - ; # \ , . / ! % & * () _ + { } : @ < > ? <p>Note: The first character is an apostrophe.</p> <p>Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.</p> <p>To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the <i>PowerExchange Reference Manual</i>.</p> <p>The allowable characters in the IBM IRRPHREX exit do not affect the allowable characters in PowerExchange passphrases.</p> <p>Note: A valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.</p>
Code Page	Required. Name of the code page to use for reading from or writing to the data source. Usually, this value is an ISO code page name, such as ISO-8859-6.
Pass-through security enabled	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.
Encryption Type	<p>The type of encryption that the Data Integration Service uses. Select one of the following options:</p> <ul style="list-style-type: none"> - None - AES <p>Default is None.</p> <p>Note: Informatica recommends that you use Secure Sockets Layer (SSL) authentication instead of configuring the Encryption Type and Level connection properties. SSL authentication provides stricter security and is used by several Informatica products. For more information about implementing SSL authentication in a PowerExchange network, see the <i>PowerExchange Reference Manual</i>.</p>

Option	Description
[Encryption] Level	<p>If you select AES for Encryption Type, select one of the following options to indicate the encryption level that the Data Integration Service uses:</p> <ul style="list-style-type: none"> - 1. Use a 128-bit encryption key. - 2. Use a 192-bit encryption key. - 3. Use a 256-bit encryption key. <p>If you do not select AES for Encryption Type, this option is ignored.</p> <p>Default is 1.</p>
Pacing size	<p>Optional. Amount of data that the source system can pass to the PowerExchange Listener. Set the pacing size if an external application, database, or the Data Integration Service node is a bottleneck. User lower values for faster performance.</p> <p>The minimum value and default value is 0. A value of 0 provides the best performance.</p>
Interpret as rows	<p>Optional. Select this option to express the pacing size as a number of rows. Clear this option to express the pacing size in kilobytes. By default, this option is not selected and the pacing size is in kilobytes.</p>
Compression	<p>Optional. Select this option to enable source data compression. By compressing data, you can decrease the amount of data that Informatica applications send over the network. By default, this option is not selected and compression is disabled.</p>
Offload processing	<p>Optional. Controls whether to offload some bulk data processing from the source machine to the Data Integration Service machine. Select one of the following options:</p> <ul style="list-style-type: none"> - AUTO. The Data Integration Service determines whether to use offload processing. - Yes. Use offload processing. - No. Do not use offload processing. <p>Default is AUTO.</p>
Worker threads	<p>Optional. Number of threads that the Data Integration Service uses to process bulk data when offload processing is enabled. For optimal performance, this value should not exceed the number of available processors on the Data Integration Service machine. Valid values are 1 through 64. Default is 0, which disables multithreading.</p>
Array size	<p>Optional. The number of records in the storage array for the worker threads. This option is applicable when you set the Worker Threads option to a value greater than 0. Valid values are 1 to 5000. Default is 25.</p>
Write mode	<p>Optional. Mode in which Data Integration Service sends data to the PowerExchange Listener. Select one of the following write modes:</p> <ul style="list-style-type: none"> - CONFIRMWRITEON. Sends data to the PowerExchange Listener and waits for a response before sending more data. Select this option when error recovery is a priority. However, this option might degrade performance. - CONFIRMWRITEOFF. Sends data to the PowerExchange Listener without waiting for a response. Use this option if you can reload the target table when an error occurs. - ASYNCHRONOUSWITHFAULTTOLERANCE. Sends data to the PowerExchange Listener without waiting for a response. This option also enables error detection. This option combines the speed of CONFIRMWRITEOFF and the data integrity of CONFIRMWRITEON. <p>Default is CONFIRMWRITEON.</p>

Amazon Redshift Connection Properties

When you set up an Amazon Redshift connection, you must configure the connection properties.

The following table describes the Amazon Redshift connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Amazon Redshift in the Database.

The **Details** tab contains the connection attributes of the Amazon Redshift connection. The following table describes the connection attributes:

Property	Description
Username	User name of the Amazon Redshift account.
Password	Password for the Amazon Redshift account.
Access Key ID	Amazon S3 bucket access key ID. Note: Required if you do not use AWS Identity and Access Management (IAM) authentication.
Secret Access Key	Amazon S3 bucket secret access key ID. Note: Required if you do not use AWS Identity and Access Management (IAM) authentication.
Master Symmetric Key	Optional. Provide a 256-bit AES encryption key in the Base64 format when you enable client-side encryption. You can generate a key using a third-party tool. If you specify a value, ensure that you specify the encryption type as client side encryption in the advanced target properties.
JDBC URL	Amazon Redshift connection URL.

Property	Description
Cluster Region	<p>Optional. The AWS cluster region in which the bucket you want to access resides.</p> <p>Select a cluster region if you choose to provide a custom JDBC URL that does not contain a cluster region name in the JDBC URL connection property.</p> <p>If you specify a cluster region in both Cluster Region and JDBC URL connection properties, the Data Integration Service ignores the cluster region that you specify in the JDBC URL connection property.</p> <p>To use the cluster region name that you specify in the JDBC URL connection property, select None as the cluster region in this property.</p> <p>Select one of the following cluster regions:</p> <p>Select one of the following regions:</p> <ul style="list-style-type: none"> - Asia Pacific (Mumbai) - Asia Pacific (Seoul) - Asia Pacific (Singapore) - Asia Pacific (Sydney) - Asia Pacific (Tokyo) - AWS GovCloud (US) - Canada (Central) - China (Beijing) - China (Ningxia) - EU (Ireland) - EU (Frankfurt) - EU (London) - EU (Paris) - South America (Sao Paulo) - US East (Ohio) - US East (N. Virginia) - US West (N. California) - US West (Oregon) <p>Default is None.</p> <p>You can only read data from or write data to the cluster regions supported by AWS SDK used by PowerExchange for Amazon Redshift.</p>
Customer Master Key ID	<p>Optional. Specify the customer master key ID generated by AWS Key Management Service (AWS KMS) or the Amazon Resource Name (ARN) of your custom key for cross-account access. You must generate the customer master key corresponding to the region where Amazon S3 bucket resides. You can specify any of the following values:</p> <p>Customer generated customer master key</p> <p>Enables client-side or server-side encryption.</p> <p>Default customer master key</p> <p>Enables client-side or server-side encryption. Only the administrator user of the account can use the default customer master key ID to enable client-side encryption.</p>

Amazon S3 Connection Properties

When you set up an Amazon S3 connection, you must configure the connection properties.

The following table describes the Amazon S3 connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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	Optional. The description of the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection.
Type	The Amazon S3 connection type.
Access Key	Access key to access the Amazon S3 bucket. Provide the access key value based on the following authentication methods: <ul style="list-style-type: none">- Basic authentication: provide the actual access key value.- IAM authentication: do not provide the access key value.- Temporary security credentials via assume role: provide access key of an IAM user with no permissions to access Amazon S3 bucket.
Secret Key	Secret access key to access the Amazon S3 bucket. The secret key is associated with the access key and uniquely identifies the account. Provide the access key value based on the following authentication methods: <ul style="list-style-type: none">- Basic authentication: provide the actual access secret value.- IAM authentication: do not provide the access secret value.- Temporary security credentials via assume role: provide access secret of an IAM user with no permissions to access Amazon S3 bucket.
IAM Role ARN	The ARN of the IAM role assumed by the user to use the dynamically generated temporary security credentials. Enter the value of this property if you want to use the temporary security credentials to access the AWS resources. If you want to use the temporary security credentials with IAM authentication, do not provide the Access Key and Secret Key connection properties. If you want to use the temporary security credentials without IAM authentication, you must enter the value of the Access Key and Secret Key connection properties. For more information about how to obtain the ARN of the IAM role, see the AWS documentation.
Folder Path	The complete path to Amazon S3 objects. The path must include the bucket name and any folder name. Do not use a slash at the end of the folder path. For example, <bucket name>/<my folder name>.
Master Symmetric Key	Optional. Provide a 256-bit AES encryption key in the Base64 format when you enable client-side encryption. You can generate a master symmetric key using a third-party tool.

Property	Description
S3 Account Type	<p>The type of the Amazon S3 account.</p> <p>Select Amazon S3 Storage or S3 Compatible Storage.</p> <p>Select the Amazon S3 storage option to use the Amazon S3 services. Select the S3 compatible storage option to specify the endpoint for a third-party storage provider such as Scalify RING.</p> <p>By default, Amazon S3 storage is selected.</p>
REST Endpoint	<p>The S3 storage endpoint.</p> <p>Specify the S3 storage endpoint in HTTP/HTTPS format when you select the S3 compatible storage option. For example, <code>http://s3.isv.scalify.com</code>.</p>
Region Name	<p>Select the AWS region in which the bucket you want to access resides.</p> <p>Select one of the following regions:</p> <ul style="list-style-type: none"> - Asia Pacific (Mumbai) - Asia Pacific (Seoul) - Asia Pacific (Singapore) - Asia Pacific (Sydney) - Asia Pacific (Tokyo) - AWS GovCloud (US) - Canada (Central) - China (Beijing) - China (Hong Kong) - China (Ningxia) - EU (Ireland) - EU (Frankfurt) - EU (London) - EU (Paris) - South America (Sao Paulo) - US East (Ohio) - US East (N. Virginia) - US West (N. California) - US West (Oregon) <p>Default is US East (N. Virginia).</p> <p>Not applicable for S3 compatible storage.</p>
Customer Master Key ID	<p>Optional. Specify the customer master key ID or alias name generated by AWS Key Management Service (AWS KMS) or the Amazon Resource Name (ARN) of your custom key for cross-account access. You must generate the customer master key for the same region where Amazon S3 bucket reside.</p> <p>You can specify any of the following values:</p> <p>Customer generated customer master key</p> <p>Enables client-side or server-side encryption.</p> <p>Default customer master key</p> <p>Enables client-side or server-side encryption. Only the administrator user of the account can use the default customer master key ID to enable client-side encryption.</p>
Federated SSO IdP	<p>SAML 2.0-enabled identity provider for the federated user single sign-on to use with the AWS account.</p> <p>PowerExchange for Amazon S3 supports only the ADFS 3.0 identity provider.</p> <p>Select <code>None</code> if you do not want to use federated user single sign-on.</p>

Federated user single sign-on connection properties

Configure the following properties when you select `ADFS 3.0` in **Federated SSO IdP**:

Property	Description
Federated User Name	User name of the federated user to access the AWS account through the identity provider.
Federated User Password	Password for the federated user to access the AWS account through the identity provider.
IdP SSO URL	Single sign-on URL of the identity provider for AWS.
SAML Identity Provider ARN	ARN of the SAML identity provider that the AWS administrator created to register the identity provider as a trusted provider.
Role ARN	ARN of the IAM role assumed by the federated user.

Blockchain Connection Properties

When you set up a blockchain connection, you must configure the connection properties.

The following table describes the general connection properties for a blockchain connection:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! \$ % ^ & * () - + = { [] \ : ; " ' < , > . ? /
ID	The 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. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Swagger File Path	The absolute path of the swagger file path that contains the REST API to communicate with the blockchain. The swagger file must be a JSON file that is stored on the Data Integration Service machine. If the swagger file is in a different file format, such as YAML, convert the file to JSON format.
Base URL	Required. The base URL that is used to access assets on the blockchain.
Auth Type*	Authentication method that the run-time engine uses to connect to the REST server. You can use none, basic, digest, or OAuth.
Auth User ID*	User name to authenticate to the REST server.
Auth Password*	Password for the user name to authenticate to the REST server.

Property	Description
OAuth Consumer Key*	Required for the OAuth authentication type. Client key that is associated with the REST server.
OAuth Consumer Secret*	Required for the OAuth authentication type. Client password to connect to the REST server.
OAuth Token*	Required for the OAuth authentication type. Access token to connect to the REST server.
OAuth Token Secret*	Required for the OAuth authentication type. Password associated with the OAuth token.
Proxy Type*	Type of proxy. You can use no proxy, platform proxy, or custom.
Proxy Details*	Proxy configuration using the format <host>:<port>.
TrustStore File Path*	The absolute path of the truststore file that contains the SSL certificate.
TrustStore Password*	Password for the truststore file.
KeyStore File Path*	The absolute path of the keystore file that contains the keys and certificates required to establish a two-way secure connection with the REST server.
KeyStore Password*	Password for the keystore file.
Advanced Properties	<p>List of advanced properties to access an asset on the blockchain. Specify the advanced properties using name-value pairs that are separated by a semicolon.</p> <p>You can use the following advanced properties:</p> <ul style="list-style-type: none"> - X-API-KEY. Required if you authenticate to the REST server using an API key. <p>The advanced properties that you configure in the connection override the values for the corresponding advanced properties in the blockchain data object. For example, if the connection and the data object both specify a base URL, the value in the connection overrides the value in the data object.</p> <p>Note: The advanced properties have the precedence level, Operation level > Object level > Connection level. The properties configured at the operation level will override the properties configured at the object or connection level.</p>
Cookies	<p>Required based on how the REST API is implemented. List of cookie properties to specify the cookie information that is passed to the REST server. Specify the properties using name-value pairs that are separated by a semicolon.</p> <p>The cookie properties that you configure in the connection override the values for the corresponding cookie properties in the blockchain data object.</p>
<p>* The property is ignored. To use the functionality, configure the property as an advanced property and provide a name-value pair based on the property name in the swagger file.</p> <p>For example, configure the following name-value pair to use basic authorization:</p> <pre>Authorization=Basic <credentials></pre> <p>Note: You cannot use Test Connection to validate a Blockchain connection.</p>	

Cassandra Connection Properties

When you set up a Cassandra connection, you must configure the connection properties.

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

The following table describes the Cassandra connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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. The ID 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	Optional. The description of the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection. Not applicable for Data Engineering Streaming.
Type	The connection type. Select Cassandra .
Host Name	Host name or IP address of the Cassandra server.
Port	Cassandra server port number. Default is 9042.
User Name	User name to access the Cassandra server.
Password	Password corresponding to the user name to access the Cassandra server.
Default Keyspace	Name of the Cassandra keyspace to use by default.
SQL Identifier Character	Type of character that the database uses to enclose delimited identifiers in SQL or CQL queries. The available characters depend on the database type. Select None if the database uses regular identifiers. When the Data Integration Service generates SQL or CQL queries, the service does not place delimited characters around any identifiers. Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL or CQL queries, the service encloses delimited identifiers within this character.
SSL Mode	Select disabled . Not applicable for PowerExchange for Cassandra JDBC. SSL mode indicates the encryption type to use for the connection. You can choose a mode from the following SSL modes: <ul style="list-style-type: none">- Disabled- One way- Two way

Property	Description
SSL Truststore Path	Not applicable for PowerExchange for Cassandra JDBC or when disabled SSL mode is selected. Absolute path and file name of the SSL truststore file that contains certificates of the trusted SSL server.
SSL Truststore Password	Not applicable for PowerExchange for Cassandra JDBC or when disabled SSL mode is selected. Password for the SSL truststore.
SSL Keystore Path	Not applicable for PowerExchange for Cassandra JDBC or when disabled SSL mode is selected. Absolute path and file name of the SSL keystore file that contains private keys and certificates for the SSL server.
SSL Keystore Password	Not applicable for PowerExchange for Cassandra JDBC or when disabled SSL mode is selected. Password for the SSL keystore.
Additional Connection Properties	<p>Enter one or more JDBC connection parameters in the following format: <param1>=<value>;<param2>=<value>;<param3>=<value></p> <p>PowerExchange for Cassandra JDBC supports the following JDBC connection parameters:</p> <ul style="list-style-type: none"> - BinaryColumnLength - DecimalColumnScale - EnableCaseSensitive - EnableNullInsert - EnablePaging - RowsPerPage - StringColumnLength - VTableSeparator

Confluent Kafka Connection

The Confluent Kafka connection is a Messaging connection. Use the Confluent Kafka connection to access a Kafka broker or a Confluent Kafka broker as a source or a target. You can create and manage a Confluent Kafka connection in the Developer tool or through infacmd.

When you configure a Confluent Kafka connection, you configure the following properties:

- List of Kafka brokers or Confluent Kafka brokers that the connection reads from or writes to.
- Number of seconds the Integration Service attempts to reconnect to the database if the connection fails.
- Version of the Confluent Kafka messaging broker.

General Properties

The following table describes the general connection properties for the Confluent Kafka connection:

Property	Description
Name	Name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! \$ % ^ & * () - + = { [] \ : ; " ' < , > . ? /
ID	The 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	Description of the connection. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Location	Domain where you want to create the connection. Select the domain name.
Type	Connection type. Select <code>Messaging/ConfluentKafka</code> .

Confluent Kafka Broker Properties

The following table describes the Kafka broker properties for the Confluent Kafka connection:

Property	Description
Kafka Broker List	Comma-separated list of Confluent Kafka brokers that maintain the configuration of the Confluent Kafka messaging broker. To specify a Confluent Kafka broker, use the following format: <code><IP address>:<port></code>
Retry Timeout	Number of seconds after which the Data Integration Service attempts to reconnect to the Confluent Kafka broker to read or write data. If the source or target is not available for the time you specify, the mapping execution stops to avoid any data loss.
Kafka Broker Version	Version of the Confluent Kafka messaging broker.
Additional Connection Properties	Optional. Comma-separated list of connection properties to connect to the Kafka broker.
Schema Registry URL	Location and port of the schema registry provider on which to connect.

Additional Connection Properties

You can use the following syntax for specifying the additional connection properties:

```
request.timeout.ms=<value>,session.timeout.ms=<value>,
fetch.max.wait.ms=<value>,heartbeat.interval.ms=<value>,
security.protocol=SASL_PLAINTEXT,sasl.kerberos.
service.name=<kerberos name>,sasl.mechanism=GSSAPI,
sasl.jaas.config=com.sun.security.auth.module.
Krb5Login Modulerequired useKeyTab=true
doNotPrompt=true storeKey=true client=true
keyTab="<Keytab Location>" principal="<principal>";
```

SSL Properties

The following table describes the SSL properties for the Confluent Kafka connection:

Property	Description
SSL Mode	Optional. SSL mode indicates the encryption type to use for the connection. You can choose one of the following SSL modes: <ul style="list-style-type: none">- Disabled- One way- Two way The default value is <code>Disabled</code> .
SSL TrustStore File Path	Required when <code>One way</code> SSL mode is selected. Absolute path and file name of the SSL truststore file that contains certificates of the trusted SSL server.
SSL TrustStore Password	Required when <code>One way</code> SSL mode is selected. Password for the SSL truststore.
SSL KeyStore File Path	Required when <code>Two way</code> SSL mode is selected. Absolute path and file name of the SSL keystore file that contains private keys and certificates for the SSL server.
SSL KeyStore Password	Required when <code>Two way</code> SSL mode is selected. Password for the SSL keystore.
Additional Security Properties	Optional. Comma-separated list of connection properties to connect to the Confluent Kafka broker in a secured way.

Creating a Confluent Kafka Connection Using `infacmd`

You can use the `infacmd` command line program to create a Confluent Kafka connection.

To create a Confluent Kafka connection on UNIX, run the following command:

```
sh infacmd.sh createConnection -dn <domain name> -un <domain user> -pd <domain password>
-cn <connection name> -cid <connection id> -ct ConfluentKafka -o
"kfkBrkList='<host1:port1>,<host2:port2>,<host3:port3>' kafkabrokerverversion='<version>'
schemaregistryurl='<schema registry URL>'"
```

For more information about the `CreateConnection` command, see the *Informatica Command Reference*.

Databricks Connection Properties

Use the Databricks connection to run mappings on a Databricks cluster.

A Databricks connection is a cluster type connection. You can create and manage a Databricks connection in the Administrator tool or the Developer tool. You can use `infacmd` to create a Databricks connection.

Configure properties in the Databricks connection to enable communication between the Data Integration Service and the Databricks cluster.

The following table describes the general connection properties for the Databricks connection:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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	Optional. The description of the connection. The description cannot exceed 4,000 characters.
Connection Type	Choose Databricks.
Cluster Configuration	Name of the cluster configuration associated with the Databricks environment. Required if you do not configure the cloud provisioning configuration.
Cloud Provisioning Configuration	Name of the cloud provisioning configuration associated with a Databricks cloud platform. Required if you do not configure the cluster configuration.
Staging Directory	The directory where the Databricks Spark engine stages run-time files. If you specify a directory that does not exist, the Data Integration Service creates it at run time. If you do not provide a directory path, the run-time staging files are written to <code><cluster staging directory>/DATABRICKS</code> .
Advanced Properties	List of advanced properties that are unique to the Databricks environment. You can configure run-time properties for the Databricks environment in the Data Integration Service and in the Databricks connection. You can override a property configured at a high level by setting the value at a lower level. For example, if you configure a property in the Data Integration Service custom properties, you can override it in the Databricks connection. The Data Integration Service processes property overrides based on the following priorities: 1. Databricks connection advanced properties 2. Data Integration Service custom properties Note: Informatica does not recommend changing these property values before you consult with third-party documentation, Informatica documentation, or Informatica Global Customer Support. If you change a value without knowledge of the property, you might experience performance degradation or other unexpected results.

Advanced Properties

Configure the following properties in the **Advanced Properties** of the Databricks configuration section:

infaspark.json.parser.mode

Specifies the parser how to handle corrupt JSON records. You can set the value to one of the following modes:

- **DROPMALFORMED.** The parser ignores all corrupted records. Default mode.
- **PERMISSIVE.** The parser accepts non-standard fields as nulls in corrupted records.
- **FAILFAST.** The parser generates an exception when it encounters a corrupted record and the Spark application goes down.

infaspark.json.parser.multiLine

Specifies whether the parser can read a multiline record in a JSON file. You can set the value to true or false. Default is false. Applies only to non-native distributions that use Spark version 2.2.x and above.

infaspark.flatfile.writer.nullValue

When the Databricks Spark engine writes to a target, it converts null values to empty strings (" "). For example, 12, AB,"",23p09udj.

The Databricks Spark engine can write the empty strings to string columns, but when it tries to write an empty string to a non-string column, the mapping fails with a type mismatch.

To allow the Databricks Spark engine to convert the empty strings back to null values and write to the target, configure the property in the Databricks Spark connection.

Set to: TRUE

infaspark.pythontx.exec

Required to run a Python transformation on the Databricks Spark engine. Set to the location of the Python executable binary on the worker nodes in the Databricks cluster.

When you provision the cluster at run time, set this property in the Databricks cloud provisioning configuration. Otherwise, set on the Databricks connection.

For example, set to:

```
infaspark.pythontx.exec=/databricks/python3/bin/python3
```

infaspark.pythontx.executorEnv.PYTHONHOME

Required to run a Python transformation on the Databricks Spark engine. Set to the location of the Python installation directory on the worker nodes in the Databricks cluster.

When you provision the cluster at run time, set this property in the Databricks cloud provisioning configuration. Otherwise, set on the Databricks connection.

For example, set to:

```
infaspark.pythontx.executorEnv.PYTHONHOME=/databricks/python3
```

Greenplum Connection Properties

Use a Greenplum connection to connect to a 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:

Property	Description
Name	Name of the Greenplum relational connection.
ID	String that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or fewer and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.
Description	Description of the connection. The description cannot exceed 765 characters.
Location	Domain on which you want to create the connection.
Type	Type of connection.

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

Property	Description
User Name	User name with permissions to access the Greenplum database.
Password	Password to connect to the Greenplum database.
Driver Name	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.
Connection String	Use the following connection URL: <code>jdbc:pivotal:greenplum://<hostname>:<port>;DatabaseName=<database_name></code> For more information about the connection URL, see the Greenplum documentation.

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. It uses the Enable SSL option and the certificate path to establish secure communication to the Greenplum server over SSL.

The following table describes the connection properties for data access:

Property	Description
Host Name	Host name or IP address of the Greenplum server.
Port Number	Greenplum server port number. If you enter 0, the gpload utility reads from the environment variable \$PGPORT. Default is 5432.
Database Name	Name of the database.
Enable SSL	Select this option to establish secure communication between the gpload utility and the Greenplum server over SSL.
Certificate Path	Path where the SSL certificates for the Greenplum server are stored. For information about the files that need to be present in the certificates path, see the gpload documentation.

Google Analytics Connection Properties

When you set up a Google Analytics connection, you must configure the connection properties.

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

The following table describes the Google Analytics connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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. The ID 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	Optional. The description of the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Google Analytics .
Service Account ID	Specifies the client_email value present in the JSON file that you download after you create a service account.
Service Account Key	Specifies the private_key value present in the JSON file that you download after you create a service account.
APIVersion	API that PowerExchange for Google Analytics uses to read from Google Analytics reports. Select Core Reporting API v3 . Note: PowerExchange for Google Analytics does not support Analytics Reporting API v4.

Google BigQuery Connection Properties

When you set up a Google BigQuery connection, you must configure the connection properties.

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

The following table describes the Google BigQuery connection properties:

Property	Description
Service Account ID	Specifies the client_email value present in the JSON file that you download after you create a service account in Google BigQuery.
Service Account Key	Specifies the private_key value present in the JSON file that you download after you create a service account in Google BigQuery.

Property	Description
Connection mode	<p>The mode that you want to use to read data from or write data to Google BigQuery.</p> <p>Select one of the following connection modes:</p> <ul style="list-style-type: none"> - Simple. Flattens each field within the Record data type field as a separate field in the mapping. - Hybrid. Displays all the top-level fields in the Google BigQuery table including Record data type fields. PowerExchange for Google BigQuery displays the top-level Record data type field as a single field of the String data type in the mapping. - Complex. Displays all the columns in the Google BigQuery table as a single field of the String data type in the mapping. <p>Default is Simple.</p>
Schema Definition File Path	<p>Specifies a directory on the client machine where the must create a JSON file with the sample schema of the Google BigQuery table. The JSON file name is the same as the Google BigQuery table name.</p> <p>Alternatively, you can specify a storage path in Google Cloud Storage where the must create a JSON file with the sample schema of the Google BigQuery table. You can download the JSON file from the specified storage path in Google Cloud Storage to a local machine.</p>
Project ID	<p>Specifies the project_id value present in the JSON file that you download after you create a service account in Google BigQuery.</p> <p>If you have created multiple projects with the same service account, enter the ID of the project that contains the dataset that you want to connect to.</p>
Storage Path	<p>This property applies when you read or write large volumes of data.</p> <p>Path in Google Cloud Storage where the creates a local stage file to store the data temporarily. You can either enter the bucket name or the bucket name and folder name.</p> <p>For example, enter <code>gs://<bucket_name></code> or <code>gs://<bucket_name>/<folder_name></code></p>
Dataset ID	Not applicable for PowerExchange for Google BigQuery.
Use Legacy SQL For Custom Query	Not applicable for PowerExchange for Google BigQuery.
Dataset Name for Custom Query	Not applicable for PowerExchange for Google BigQuery.
Region ID	<p>The region name where the Google BigQuery dataset resides.</p> <p>For example, if you want to connect to a Google BigQuery dataset that resides in Las Vegas region, specify us-west4 as the Region ID.</p> <p>Note: In the Storage Path connection property, ensure that you specify a bucket name or the bucket name and folder name that resides in the same region as the dataset in Google BigQuery.</p> <p>For more information about the regions supported by Google BigQuery, see the following Google BigQuery documentation:https://cloud.google.com/bigquery/docs/locations</p>

Property	Description
Optional Properties	<p>Specifies whether you can configure certain source and target functionalities through custom properties.</p> <p>You can select one of the following options:</p> <ul style="list-style-type: none"> - None. Select if you do not want to configure any custom properties. - Required. If you want to specify custom properties to configure the source and target functionalities. <p>Default is None.</p>
Provide Optional Properties	<p>Comma-separated key-value pairs of custom properties to enable certain source and target functionalities.</p> <p>Appears only when you select Required in the Optional Properties.</p> <p>For more information about the list of custom properties that you can specify, see the Informatica Knowledge Base article: https://kb.informatica.com/faq/7/Pages/26/632722.aspx</p>

Connection Modes

You can configure a Google BigQuery connection to use one of the following connection modes:

Simple mode

If you use simple mode, PowerExchange for Google BigQuery flattens each field within the Record data type field as a separate field in the Google BigQuery data object.

Hybrid mode

If you use hybrid mode, PowerExchange for Google BigQuery displays all the top-level fields in the Google BigQuery table including Record data type fields. PowerExchange for Google BigQuery displays the top-level Record data type field as a single field of the String data type in the Google BigQuery data object.

Complex mode

If you use complex mode, PowerExchange for Google BigQuery displays all the columns in the Google BigQuery table as a single field of the String data type in the Google BigQuery data object.

Google Cloud Spanner Connection Properties

When you set up a Google Cloud Spanner connection, you must configure the connection properties.

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

The following table describes the Google Cloud Spanner connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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. The ID 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	Optional. The description of the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Google Cloud Spanner.
Project ID	Specifies the project_id value present in the JSON file that you download after you create a service account. If you have created multiple projects with the same service account, enter the ID of the project that contains the bucket that you want to connect to.
Service Account ID	Specifies the client_email value present in the JSON file that you download after you create a service account.
Service Account Key	Specifies the private_key value present in the JSON file that you download after you create a service account.
Instance ID	Name of the instance that you created in Google Cloud Spanner.

Google Cloud Storage Connection Properties

When you set up a Google Cloud Storage connection, you must configure the connection properties.

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

The following table describes the Google Cloud Storage connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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. The ID 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.

Property	Description
Description	Optional. The description of the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Google Cloud Storage .
Project ID	Specifies the <code>project_id</code> value present in the JSON file that you download after you create a service account. If you have created multiple projects with the same service account, enter the ID of the project that contains the bucket that you want to connect to.
Service Account ID	Specifies the <code>client_email</code> value present in the JSON file that you download after you create a service account.
Service Account Key	Specifies the <code>private_key</code> value present in the JSON file that you download after you create a service account.

Google PubSub Connection Properties

When you create a Google PubSub connection, you must configure the connection properties.

The following table describes the Google PubSub connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! \$ % ^ & * () - + = { [] \ : ; " ' < , > . ? /
ID	The 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. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Connection Type	The connection type. Select Pub Sub connection type.
Client Email	The <code>client_email</code> value available in the JSON file that you download after you create a service account.
Client Id	The <code>client_id</code> value available in the JSON file that you download after you create a service account.
Private Key Id	The <code>private_key_id</code> value available in the JSON file that you download after you create a service account.

Property	Description
Private Key	The <code>private_key</code> value available in the JSON file that you download after you create a service account.
Project Id	The <code>project_id</code> value available in the JSON file that you download after you create a service account.

Hadoop Connection Properties

Use the Hadoop connection to configure mappings to run on a Hadoop cluster. A Hadoop connection is a cluster type connection. You can create and manage a Hadoop connection in the Administrator tool or the Developer tool. You can use `infacmd` to create a Hadoop connection. Hadoop connection properties are case sensitive unless otherwise noted.

You can configure run-time properties for the Hadoop environment in the Data Integration Service, the Hadoop connection, and in the mapping. You can override a property configured at a high level by setting the value at a lower level. For example, if you configure a property in the Data Integration Service custom properties, you can override it in the Hadoop connection or in the mapping. The Data Integration Service processes property overrides based on the following priorities:

1. Mapping custom properties set using `infacmd ms runMapping` with the `-cp` option
2. Mapping run-time properties for the Hadoop environment
3. Hadoop connection advanced properties for run-time engines
4. Hadoop connection advanced general properties, environment variables, and classpaths
5. Data Integration Service custom properties

Note: When a mapping uses Hive Server 2 to run a job or parts of a job, you cannot override properties that are configured on the cluster level in preSQL or post-SQL queries or SQL override statements. Workaround: Instead of attempting to use the cluster configuration on the domain to override cluster properties, pass the override settings to the JDBC URL. For example: `beeline -u "jdbc:hive2://<domain host>:<port_number>/tpch_text_100" --hiveconf hive.execution.engine=tez`

Hadoop Cluster Properties

Configure properties in the Hadoop connection to enable communication between the Data Integration Service and the Hadoop cluster.

The following table describes the general connection properties for the Hadoop connection:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Cluster Configuration	The name of the cluster configuration associated with the Hadoop environment. Required if you do not configure the Cloud Provisioning Configuration.
Cloud Provisioning Configuration	Name of the cloud provisioning configuration associated with a cloud platform such as Amazon AWS or Microsoft Azure. Required if you do not configure the Cluster Configuration.
Cluster Environment Variables*	<p>Environment variables that the Hadoop cluster uses.</p> <p>If you use a Cloudera CDH 6.x cluster or a Cloudera CDP cluster, configure the locale setting as cluster environment variables. In Cloudera Manager, you must also add the environment variables to the following YARN property:</p> <pre>yarn.nodemanager.env-whitelist</pre> <p>For example, the variable ORACLE_HOME represents the directory where the Oracle database client software is installed.</p> <p>You can configure run-time properties for the Hadoop environment in the Data Integration Service, the Hadoop connection, and in the mapping. You can override a property configured at a high level by setting the value at a lower level. For example, if you configure a property in the Data Integration Service custom properties, you can override it in the Hadoop connection or in the mapping. The Data Integration Service processes property overrides based on the following priorities:</p> <ol style="list-style-type: none"> 1. Mapping custom properties set using <code>infacmd ms runMapping</code> with the <code>-cp</code> option 2. Mapping run-time properties for the Hadoop environment 3. Hadoop connection advanced properties for run-time engines 4. Hadoop connection advanced general properties, environment variables, and classpaths 5. Data Integration Service custom properties <p>Note: When a mapping uses Hive Server 2 to run a job or parts of a job, you cannot override properties that are configured on the cluster level in preSQL or post-SQL queries or SQL override statements. Workaround: Instead of attempting to use the cluster configuration on the domain to override cluster properties, pass the override settings to the JDBC URL. For example: <code>beeline -u "jdbc:hive2://<domain host>:<port_number>/tpch_text_100" --hiveconf hive.execution.engine=tez</code></p>
Cluster Library Path*	The path for shared libraries on the cluster. The <code>\$DEFAULT_CLUSTER_LIBRARY_PATH</code> variable contains a list of default directories.

Property	Description
Cluster Classpath*	<p>The classpath to access the Hadoop jar files and the required libraries.</p> <p>The \$DEFAULT_CLUSTER_CLASSPATH variable contains a list of paths to the default jar files and libraries.</p> <p>You can configure run-time properties for the Hadoop environment in the Data Integration Service, the Hadoop connection, and in the mapping. You can override a property configured at a high level by setting the value at a lower level. For example, if you configure a property in the Data Integration Service custom properties, you can override it in the Hadoop connection or in the mapping. The Data Integration Service processes property overrides based on the following priorities:</p> <ol style="list-style-type: none"> 1. Mapping custom properties set using infacmd ms runMapping with the <code>-cp</code> option 2. Mapping run-time properties for the Hadoop environment 3. Hadoop connection advanced properties for run-time engines 4. Hadoop connection advanced general properties, environment variables, and classpaths 5. Data Integration Service custom properties <p>Note: When a mapping uses Hive Server 2 to run a job or parts of a job, you cannot override properties that are configured on the cluster level in preSQL or post-SQL queries or SQL override statements. Workaround: Instead of attempting to use the cluster configuration on the domain to override cluster properties, pass the override settings to the JDBC URL. For example: <code>beeline -u "jdbc:hive2://<domain host>:<port_number>/tpch_text_100" --hiveconf hive.execution.engine=tez</code></p>
Cluster Executable Path*	<p>The path for executable files on the cluster.</p> <p>The \$DEFAULT_CLUSTER_EXEC_PATH variable contains a list of paths to the default executable files.</p>
<p>* Informatica does not recommend changing these property values before you consult with third-party documentation, Informatica documentation, or Informatica Global Customer Support. If you change a value without knowledge of the property, you might experience performance degradation or other unexpected results.</p>	

Common Properties

The following table describes the common connection properties that you configure for the Hadoop connection:

Property	Description
Impersonation User Name	<p>Required if the Hadoop cluster uses Kerberos authentication. Hadoop impersonation user. The user name that the Data Integration Service impersonates to run mappings in the Hadoop environment.</p> <p>Data Engineering Integration supports operating system profiles on all Hadoop distributions. In the Hadoop run-time environment, the Data Integration Service pushes the processing to the Hadoop cluster and the run-time engines run mappings with the operating system profile specified Hadoop impersonation properties.</p>
Temporary Table Compression Codec	<p>Hadoop compression library for a compression codec class name.</p> <p>Note: The Spark engine does not support compression settings for temporary tables. When you run mappings on the Spark engine, the Spark engine stores temporary tables in an uncompressed file format.</p>
Codec Class Name	<p>Codec class name that enables data compression and improves performance on temporary staging tables.</p>

Property	Description
Hive Staging Database Name	<p>Namespace for Hive staging tables. Use the name <code>default</code> for tables that do not have a specified database name.</p> <p>If you do not configure a namespace, the Data Integration Service uses the Hive database name in the Hive target connection to create staging tables.</p> <p>When you run a mapping in the native environment to write data to Hive, you must configure the Hive staging database name in the Hive connection. The Data Integration Service ignores the value you configure in the Hadoop connection.</p>
Environment SQL	<p>SQL commands to set the Hadoop environment. The Data Integration Service executes the environment SQL at the beginning of each Hive script generated by a HiveServer2 job.</p> <p>The following rules and guidelines apply to the usage of environment SQL:</p> <ul style="list-style-type: none"> - You can use environment SQL to define Hadoop or Hive parameters that you want to use in the PreSQL commands or in custom queries. - If you use multiple values for the Environment SQL property, ensure that there is no space between the values.
Engine Type	<p>The Data Integration Service uses HiveServer2 to process portions of some jobs by running HiveServer2 tasks on the Spark engine. When you import the cluster configuration through the Administrator tool, you can choose to create connections. The engine type property is populated by default based on the distribution.</p> <p>When you manually create a connection, you must configure the engine type.</p> <p>You can specify the engine type based on the following Hadoop distributions:</p> <ul style="list-style-type: none"> - Amazon EMR. Tez - Azure HDI. Tez - Cloudera CDH. MRv2 - Cloudera CDP. Tez - Dataproc. MRv2 - Hortonworks HDP. Tez - MapR. MRv2
Advanced Properties	<p>List of advanced properties that are unique to the Hadoop environment. The properties are common to the Blaze and Spark engines. The advanced properties include a list of default properties.</p> <p>You can configure run-time properties for the Hadoop environment in the Data Integration Service, the Hadoop connection, and in the mapping. You can override a property configured at a high level by setting the value at a lower level. For example, if you configure a property in the Data Integration Service custom properties, you can override it in the Hadoop connection or in the mapping. The Data Integration Service processes property overrides based on the following priorities:</p> <ol style="list-style-type: none"> 1. Mapping custom properties set using <code>infacmd ms runMapping</code> with the <code>-cp</code> option 2. Mapping run-time properties for the Hadoop environment 3. Hadoop connection advanced properties for run-time engines 4. Hadoop connection advanced general properties, environment variables, and classpaths 5. Data Integration Service custom properties <p>Note: When a mapping uses Hive Server 2 to run a job or parts of a job, you cannot override properties that are configured on the cluster level in preSQL or post-SQL queries or SQL override statements. Workaround: Instead of attempting to use the cluster configuration on the domain to override cluster properties, pass the override settings to the JDBC URL. For example: <code>beeline -u "jdbc:hive2://<domain host>:<port_number>/tpch_text_100" --hiveconf hive.execution.engine=tez</code></p> <p>Note: Informatica does not recommend changing these property values before you consult with third-party documentation, Informatica documentation, or Informatica Global Customer Support. If you change a value without knowledge of the property, you might experience performance degradation or other unexpected results.</p>

Reject Directory Properties

The following table describes the connection properties that you configure to the Hadoop Reject Directory.

Property	Description
Write Reject Files to Hadoop	If you use the Blaze engine to run mappings, select the check box to specify a location to move reject files. If checked, the Data Integration Service moves the reject files to the HDFS location listed in the property, Reject File Directory. By default, the Data Integration Service stores the reject files based on the RejectDir system parameter.
Reject File Directory	The directory for Hadoop mapping files on HDFS when you run mappings.

Blaze Configuration

The following table describes the connection properties that you configure for the Blaze engine:

Property	Description
Blaze Staging Directory	The HDFS file path of the directory that the Blaze engine uses to store temporary files. Verify that the directory exists. The YARN user, Blaze engine user, and mapping impersonation user must have write permission on this directory. Default is <code>/blaze/workdir</code> . If you clear this property, the staging files are written to the Hadoop staging directory <code>/tmp/blaze_<user name></code> .
Blaze User Name	The owner of the Blaze service and Blaze service logs. When the Hadoop cluster uses Kerberos authentication, the default user is the Data Integration Service SPN user. When the Hadoop cluster does not use Kerberos authentication and the Blaze user is not configured, the default user is the Data Integration Service user.
Minimum Port	The minimum value for the port number range for the Blaze engine. Default is 12300.
Maximum Port	The maximum value for the port number range for the Blaze engine. Default is 12600.
YARN Queue Name	The YARN scheduler queue name used by the Blaze engine that specifies available resources on a cluster. Note: If YARN preemption is enabled on the cluster, verify with the Hadoop administrator that preemption is disabled on the queue associated with the Blaze engine.
Blaze Job Monitor Address	The host name and port number for the Blaze Job Monitor. Use the following format: <code><hostname>:<port></code> Where <ul style="list-style-type: none">- <code><hostname></code> is the host name or IP address of the Blaze Job Monitor server.- <code><port></code> is the port on which the Blaze Job Monitor listens for remote procedure calls (RPC). For example, enter: <code>myhostname:9080</code>

Property	Description
Blaze YARN Node Label	<p>Node label that determines the node on the Hadoop cluster where the Blaze engine runs. If you do not specify a node label, the Blaze engine runs on the nodes in the default partition.</p> <p>If the Hadoop cluster supports logical operators for node labels, you can specify a list of node labels. To list the node labels, use the operators && (AND), (OR), and ! (NOT).</p> <p>Note: You cannot use node labels on a Cloudera CDH cluster.</p>
Advanced Properties	<p>List of advanced properties that are unique to the Blaze engine. The advanced properties include a list of default properties.</p> <p>You can configure run-time properties for the Hadoop environment in the Data Integration Service, the Hadoop connection, and in the mapping. You can override a property configured at a high level by setting the value at a lower level. For example, if you configure a property in the Data Integration Service custom properties, you can override it in the Hadoop connection or in the mapping. The Data Integration Service processes property overrides based on the following priorities:</p> <ol style="list-style-type: none"> 1. Mapping custom properties set using <code>infacmd ms runMapping</code> with the <code>-cp</code> option 2. Mapping run-time properties for the Hadoop environment 3. Hadoop connection advanced properties for run-time engines 4. Hadoop connection advanced general properties, environment variables, and classpaths 5. Data Integration Service custom properties <p>Note: When a mapping uses Hive Server 2 to run a job or parts of a job, you cannot override properties that are configured on the cluster level in preSQL or post-SQL queries or SQL override statements. Workaround: Instead of attempting to use the cluster configuration on the domain to override cluster properties, pass the override settings to the JDBC URL. For example: <code>beeline -u "jdbc:hive2://<domain host>:<port_number>/tpch_text_100" --hiveconf hive.execution.engine=tez</code></p> <p>Note: Informatica does not recommend changing these property values before you consult with third-party documentation, Informatica documentation, or Informatica Global Customer Support. If you change a value without knowledge of the property, you might experience performance degradation or other unexpected results.</p>

Spark Configuration

The following table describes the connection properties that you configure for the Spark engine:

Property	Description
Spark Staging Directory	<p>The HDFS file path of the directory that the Spark engine uses to store temporary files for running jobs. The YARN user, Data Integration Service user, and mapping impersonation user must have write permission on this directory.</p> <p>If you do not specify a file path, by default, the temporary files are written to the Hadoop staging directory <code>/tmp/SPARK_<user name></code>.</p> <p>When you run Sqoop jobs on the Spark engine, the Data Integration Service creates a Sqoop staging directory within the Spark staging directory to store temporary files: <code><Spark staging directory>/sqoop_staging</code></p>
Spark Event Log Directory	Optional. The HDFS file path of the directory that the Spark engine uses to log events.

Property	Description
YARN Queue Name	The YARN scheduler queue name used by the Spark engine that specifies available resources on a cluster. The name is case sensitive.
Advanced Properties	<p>List of advanced properties that are unique to the Spark engine. The advanced properties include a list of default properties.</p> <p>You can configure run-time properties for the Hadoop environment in the Data Integration Service, the Hadoop connection, and in the mapping. You can override a property configured at a high level by setting the value at a lower level. For example, if you configure a property in the Data Integration Service custom properties, you can override it in the Hadoop connection or in the mapping. The Data Integration Service processes property overrides based on the following priorities:</p> <ol style="list-style-type: none"> 1. Mapping custom properties set using <code>infacmd ms runMapping</code> with the <code>-cp</code> option 2. Mapping run-time properties for the Hadoop environment 3. Hadoop connection advanced properties for run-time engines 4. Hadoop connection advanced general properties, environment variables, and classpaths 5. Data Integration Service custom properties <p>Note: When a mapping uses Hive Server 2 to run a job or parts of a job, you cannot override properties that are configured on the cluster level in preSQL or post-SQL queries or SQL override statements. Workaround: Instead of attempting to use the cluster configuration on the domain to override cluster properties, pass the override settings to the JDBC URL. For example: <code>beeline -u "jdbc:hive2://<domain host>:<port_number>/tpch_text_100" --hiveconf hive.execution.engine=tez</code></p> <p>Note: Informatica does not recommend changing these property values before you consult with third-party documentation, Informatica documentation, or Informatica Global Customer Support. If you change a value without knowledge of the property, you might experience performance degradation or other unexpected results.</p>

HBase Connection Properties

Use an HBase connection to access HBase. The HBase connection is a NoSQL connection. You can create and manage an HBase connection in the Administrator tool or the Developer tool. HBase connection properties are case sensitive unless otherwise noted.

The following table describes HBase connection properties:

Property	Description
Name	<p>The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters:</p> <pre>~ ` ! \$ % ^ & * () - + = { [] \ : ; " ' < , > . ? /</pre>
ID	<p>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.</p>
Description	<p>The description of the connection. The description cannot exceed 4,000 characters.</p>
Location	<p>The domain where you want to create the connection.</p>

Property	Description
Type	The connection type. Select HBase.
Database Type	Type of database that you want to connect to. Select HBase to create a connection for an HBase table.

HDFS Connection Properties

Use a Hadoop File System (HDFS) connection to access data in the Hadoop cluster. The HDFS connection is a file system type connection. You can create and manage an HDFS connection in the Administrator tool, Analyst tool, or the Developer tool. HDFS connection properties are case sensitive unless otherwise noted.

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

The following table describes HDFS connection properties:

Property	Description
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.
Location	The domain where you want to create the connection. Not valid for the Analyst tool.
Type	The connection type. Default is Hadoop File System.
User Name	User name to access HDFS.
NameNode URI	The URI to access the storage system. You can find the value for <code>fs.defaultFS</code> in the <code>core-site.xml</code> configuration set of the cluster configuration. If you create connections when you import the cluster configuration, the NameNode URI property is populated by default, and it is updated each time you refresh the cluster configuration. If you use a Cloudera CDP Public Cloud compute cluster and the HDFS is on a Cloudera Data Lake cluster, set the property <code>spark.yarn.access.hadoopFileSystems</code> in the Spark properties of the Hadoop Connection to the same value as set here.

Accessing Multiple Storage Types

Use the NameNode URI property in the connection parameters to connect to various storage types. The following table lists the storage type and the NameNode URI format for the storage type:

Storage	NameNode URI Format
HDFS	<code>hdfs://<namenode>:<port></code> where: <ul style="list-style-type: none">- <namenode> is the host name or IP address of the NameNode.- <port> is the port that the NameNode listens for remote procedure calls (RPC). <code>hdfs://<nameservice></code> in case of NameNode high availability.
MapR-FS	<code>maprfs:///</code>
WASB in HDInsight	<code>wasb://<container_name>@<account_name>.blob.core.windows.net/<path></code> where: <ul style="list-style-type: none">- <container_name> identifies a specific Azure Storage Blob container. Note: <container_name> is optional. <ul style="list-style-type: none">- <account_name> identifies the Azure Storage Blob object. Example: <code>wasb://infabdmoffering1storage.blob.core.windows.net/infabdmoffering1cluster/mr-history</code>
ADLS in HDInsight	<code>adl://home</code>

When you create a cluster configuration from an Azure HDInsight cluster, the cluster configuration uses either ADLS or WASB as the primary storage. You cannot create a cluster configuration with ADLS or WASB as the secondary storage. You can edit the NameNode URI property in the HDFS connection to connect to a local HDFS location.

HBase Connection Properties for MapR-DB

Use an HBase connection to connect to a MapR-DB table. The HBase connection is a NoSQL connection. You can create and manage an HBase connection in the Administrator tool or the Developer tool. HBase connection properties are case sensitive unless otherwise noted.

The following table describes the HBase connection properties for MapR-DB:

Property	Description
Name	Name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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	Description of the connection. The description cannot exceed 4,000 characters.
Location	Domain where you want to create the connection.
Type	Connection type. Select HBase .
Database Type	Type of database that you want to connect to. Select MapR-DB to create a connection for a MapR-DB table.
Cluster Configuration	The name of the cluster configuration associated with the Hadoop environment.
MapR-DB Database Path	Database path that contains the MapR-DB table that you want to connect to. Enter a valid MapR cluster path. When you create an HBase data object for MapR-DB, you can browse only tables that exist in the MapR-DB path that you specify in the Database Path field. You cannot access tables that are available in sub-directories in the specified path. For example, if you specify the path as <code>/user/customers/</code> , you can access the tables in the <code>customers</code> directory. However, if the <code>customers</code> directory contains a sub-directory named <code>regions</code> , you cannot access the tables in the following directory: <code>/user/customers/regions</code>

Hive Connection Properties

Use the Hive connection to access Hive data. A Hive connection is a database type connection. You can create and manage a Hive connection in the Administrator tool, Analyst tool, or the Developer tool. Hive connection properties are case sensitive unless otherwise noted.

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

The following table describes Hive connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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 4000 characters.
Location	The domain where you want to create the connection. Not valid for the Analyst tool.
Type	The connection type. Select Hive.
LDAP username	LDAP user name of the user that the Data Integration Service impersonates to run mappings on a Hadoop cluster. The user name depends on the JDBC connection string that you specify in the Metadata Connection String or Data Access Connection String for the native environment. If the Hadoop cluster uses Kerberos authentication, the principal name for the JDBC connection string and the user name must be the same. Otherwise, the user name depends on the behavior of the JDBC driver. With Hive JDBC driver, you can specify a user name in many ways and the user name can become a part of the JDBC URL. If the Hadoop cluster does not use Kerberos authentication, the user name depends on the behavior of the JDBC driver. If you do not specify a user name, the Hadoop cluster authenticates jobs based on the following criteria: <ul style="list-style-type: none"> - The Hadoop cluster does not use Kerberos authentication. It authenticates jobs based on the operating system profile user name of the machine that runs the Data Integration Service. - The Hadoop cluster uses Kerberos authentication. It authenticates jobs based on the SPN of the Data Integration Service. LDAP username will be ignored.
Password	Password for the LDAP username.

Property	Description
Environment SQL	<p>SQL commands to set the Hadoop environment. In native environment type, the Data Integration Service executes the environment SQL each time it creates a connection to a Hive metastore. If you use the Hive connection to run profiles on a Hadoop cluster, the Data Integration Service executes the environment SQL at the beginning of each Hive session.</p> <p>The following rules and guidelines apply to the usage of environment SQL in both connection modes:</p> <ul style="list-style-type: none"> - Use the environment SQL to specify Hive queries. - Use the environment SQL to set the classpath for Hive user-defined functions and then use environment SQL or PreSQL to specify the Hive user-defined functions. You cannot use PreSQL in the data object properties to specify the classpath. If you use Hive user-defined functions, you must copy the .jar files to the following directory: <pre><Informatica installation directory>/services/shared/hadoop/ <Hadoop distribution name>/extras/hive-auxjars</pre> - You can use environment SQL to define Hadoop or Hive parameters that you want to use in the PreSQL commands or in custom queries. - If you use multiple values for the Environment SQL property, ensure that there is no space between the values.
SQL Identifier Character	<p>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.</p>

Properties to Access Hive as Source or Target

The following table describes the connection properties that you configure to access Hive as a source or target:

Property	Description
JDBC Driver Class Name	Name of the Hive JDBC driver class. If you leave this option blank, the Developer tool uses the default Apache Hive JDBC driver shipped with the distribution. If the default Apache Hive JDBC driver does not fit your requirements, you can override the Apache Hive JDBC driver with a third-party Hive JDBC driver by specifying the driver class name.
Metadata Connection String	<p>The JDBC connection URI used to access the metadata from the Hadoop server.</p> <p>You can use PowerExchange for Hive to communicate with a HiveServer service or HiveServer2 service. To connect to HiveServer, specify the connection string in the following format:</p> <pre>jdbc:hive2://<hostname>:<port>/<db></pre> <p>Where</p> <ul style="list-style-type: none"> - <hostname> is name or IP address of the machine on which HiveServer2 runs. - <port> is the port number on which HiveServer2 listens. - <db> is the database name to which you want to connect. If you do not provide the database name, the Data Integration Service uses the default database details. <p>To connect to HiveServer2, use the connection string format that Apache Hive implements for that specific Hadoop Distribution. For more information about Apache Hive connection string formats, see the Apache Hive documentation.</p> <p>For user impersonation, you must add <code>hive.server2.proxy.user=<xyz></code> to the JDBC connection URI. If you do not configure user impersonation, the current user's credentials are used connect to the HiveServer2.</p> <p>If the Hadoop cluster uses SSL or TLS authentication, you must add <code>ssl=true</code> to the JDBC connection URI. For example: <code>jdbc:hive2://<hostname>:<port>/<db>;ssl=true</code></p> <p>If you use self-signed certificate for SSL or TLS authentication, ensure that the certificate file is available on the client machine and the Data Integration Service machine. For more information, see the <i>Data Engineering Integration Guide</i>.</p>
Bypass Hive JDBC Server	<p>JDBC driver mode. Select the check box to use the embedded JDBC driver mode.</p> <p>To use the JDBC embedded mode, perform the following tasks:</p> <ul style="list-style-type: none"> - Verify that Hive client and Informatica services are installed on the same machine. - Configure the Hive connection properties to run mappings on a Hadoop cluster. <p>If you choose the non-embedded mode, you must configure the Data Access Connection String. Informatica recommends that you use the JDBC embedded mode.</p>
Fine Grained Authorization	<p>When you select the option to observe fine grained authorization in a Hive source, the mapping observes the following:</p> <ul style="list-style-type: none"> - Row and column level restrictions. Applies to Hadoop clusters where Sentry or Ranger security modes are enabled. - Data masking rules. Applies to masking rules set on columns containing sensitive data by Dynamic Data Masking. <p>If you do not select the option, the Blaze and Spark engines ignore the restrictions and masking rules, and results include restricted or sensitive data.</p>

Property	Description
Data Access Connection String	<p>The connection string to access data from the Hadoop data store. To connect to HiveServer, specify the non-embedded JDBC mode connection string in the following format:</p> <pre>jdbc:hive2://<hostname>:<port>/<db></pre> <p>Where</p> <ul style="list-style-type: none"> - <hostname> is name or IP address of the machine on which HiveServer2 runs. - <port> is the port number on which HiveServer2 listens. - <db> is the database to which you want to connect. If you do not provide the database name, the Data Integration Service uses the default database details. <p>To connect to HiveServer2, use the connection string format that Apache Hive implements for the specific Hadoop Distribution. For more information about Apache Hive connection string formats, see the Apache Hive documentation.</p> <p>For user impersonation, you must add <code>hive.server2.proxy.user=<xyz></code> to the JDBC connection URI. If you do not configure user impersonation, the current user's credentials are used connect to the HiveServer2.</p> <p>If the Hadoop cluster uses SSL or TLS authentication, you must add <code>ssl=true</code> to the JDBC connection URI. For example: <code>jdbc:hive2://<hostname>:<port>/<db>;ssl=true</code></p> <p>If you use self-signed certificate for SSL or TLS authentication, ensure that the certificate file is available on the client machine and the Data Integration Service machine. For more information, see the <i>Data Engineering Integration Guide</i>.</p>
Hive Staging Directory on HDFS	<p>HDFS directory for Hive staging tables. You must grant execute permission to the Hadoop impersonation user and the mapping impersonation users.</p> <p>This option is applicable and required when you write data to a Hive target in the native environment.</p>
Hive Staging Database Name	<p>Namespace for Hive staging tables.</p> <p>The Hive Staging Database Name is automatically updated from the Data Access Connection String. If you want to override the default name, you need to configure the Hive Staging Database Name in the Hive connection.</p> <p>This option is applicable when you run a mapping in the native environment to write data to a Hive target.</p> <p>If you run the mapping on the Blaze or Spark engine, you do not need to configure the Hive staging database name in the Hive connection. The Data Integration Service uses the value that you configure in the Hadoop connection.</p>

HTTP Connection Properties

Use an HTTP connection to connect a REST Web Service Consumer transformation to a web service. The HTTP connection is a web type connection. You create an HTTP connection in the Developer tool. You can manage an HTTP 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 HTTP connection properties:

Property	Description
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.
Username	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.
Password	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.
End Point URL	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.
Timeout	Number of seconds that the Data Integration Service waits for a response from the web service provider before it closes the connection. Specify a timeout value between 1 and 10,000 seconds.
HTTP Authentication Type	Type of user authentication over HTTP. Select one of the following values: <ul style="list-style-type: none"> - 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.
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.

Property	Description
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.

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.

The following table describes DB2 connection properties:

Property	Description
Database Type	The database type.
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.

Property	Description
Pass-through security enabled	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.
Connection String for data access	<p>The DB2 connection URL used to access metadata from the database.</p> <p>dbname</p> <p>Where dbname is the alias configured in the DB2 client.</p>
Metadata Access Properties: Connection String	<p>Use the following metadata connection string URL:</p> <pre>jdbc:informatica:db2://<host name>:<port>;DatabaseName=<database name></pre> <p>When you import a table, by default, all tables are displayed under the default schema name. To view tables under a specific schema instead of the default schema, you can specify the schema name from which you want to import the table. Include the ischename parameter in the URL to specify the schema name. For example, use the following syntax to import a table from a specific schema:</p> <pre>jdbc:informatica:db2://<host name>:<port>;DatabaseName=<database name>;ischename=<schema_name></pre> <p>To search for a table in multiple schemas and import it, you can specify multiple schema names in the ischename parameter. The schema name is case sensitive. You cannot use special characters when you specify multiple schema names. Use the pipe () character to separate multiple schema names. For example, use the following syntax to search for a table in three schemas and import it:</p> <pre>jdbc:informatica:db2://<host name>:<port>;DatabaseName=<database name>;ischename=<schema_name1> <schema_name2> <schema_name3></pre> <p>When you specify multiple schema names, you must clear the Show Default Schema Only option to view the tables under the specified schema names.</p>

Property	Description
AdvancedJDBCSecurityOptions	<p>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.</p> <p>To connect to a secure database, include the following parameters:</p> <ul style="list-style-type: none"> - 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. <p>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.</p> <p>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.</p> <ul style="list-style-type: none"> - 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. - cryptoProtocolVersion. Optional. If you enable TLS for the IBM DB2 instance, set the cryptoProtocolVersion parameter as follows: cryptoProtocolVersion=TLSv<version number>. For example, cryptoProtocolVersion=TLSv1.2 <p>Note: The version number must be the same as the TLS version you configured for the server.</p> <ul style="list-style-type: none"> - TrustStore. Required. Path and file name of the truststore file. <p>Note: If you configure SSL or TLS and specify only the file name, you must copy the truststore file to the Informatica installation directory. To test the connection and import metadata, copy the truststore file to the following directory:</p> <pre><Informatica client installation directory>/clients/DeveloperClient</pre> <p>To run the mapping, copy the truststore file to the following directory:</p> <pre><Informatica server installation directory>/tomcat/bin</pre> <ul style="list-style-type: none"> - TrustStorePassword. Required. Password for the truststore file for the secure database. <p>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	<p>The connection string used to access data from the database. For IBM DB2 this is <database name></p>
Code Page	<p>The code page used to read from a source database or to write to a target database or file.</p>
Environment SQL	<p>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.</p>
Transaction SQL	<p>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.</p>
Retry Period	<p>This property is reserved for future use.</p>

Property	Description
Tablespace	The tablespace name of the database.
SQL Identifier Character	Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type. Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers. Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character.
Support Mixed-case Identifiers	Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the SQL Identifier Character property. When the SQL Identifier Character property is set to none, the Support Mixed-case Identifiers property is disabled.
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.

IBM DB2 for i5/OS Connection Properties

Use an IBM DB2 for i5/OS connection to access tables in IBM DB2 for i5/OS. An IBM DB2 for i5/OS connection is a relational database connection. You can create and manage an IBM DB2 for i5/OS 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 DB2 for i5/OS connection properties:

Property	Description
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 255 characters.
Connection Type	The connection type (DB2I).

Property	Description
Username	A database user name.
Password	<p>A password for the specified user name or a valid PowerExchange passphrase.</p> <p>A PowerExchange passphrase can be from 9 to 31 characters in length and can contain the following characters:</p> <ul style="list-style-type: none"> - Uppercase and lowercase letters - The numbers 0 to 9 - Spaces - The following special characters: ' - ; # \ , . / ! % & * () _ + { } : @ < > ? <p>Note: The first character is an apostrophe.</p> <p>Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.</p> <p>To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the <i>PowerExchange Reference Manual</i>.</p>
Pass-through security enabled	Enables pass-through security for the connection.
Database name	The database instance name.
Location	Node name for the location of the PowerExchange Listener that connects to DB2. The node name is defined in the first parameter of the NODE statement in the PowerExchange dbmover.cfg configuration file.
Environment SQL	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.
Database file overrides	<p>Specifies the i5/OS database file override in the following format:</p> <pre>from_file/to_library/to_file/to_member</pre> <p>Where:</p> <ul style="list-style-type: none"> - <i>from_file</i> is the file to be overridden. - <i>to_library</i> is the new library to use. - <i>to_file</i> is the file in the new library to use. - <i>to_member</i> is optional and is the member in the new library and file to use. *FIRST is used if nothing is specified. <p>You can specify up to eight unique file overrides on a single connection. A single override applies to a single source or target. When you specify more than one file override, enclose the string of file overrides in double quotes (") and include a space between each file override.</p> <p>Note: If you specify both Library List and Database File Overrides and a table exists in both, the Database File Overrides value takes precedence.</p>
Library list	<p>List of libraries that PowerExchange searches to qualify the table name for Select, Insert, Delete, or Update statements. PowerExchange searches the list if the table name is unqualified.</p> <p>Separate libraries with commas.</p> <p>Note: If you specify both Library List and Database File Overrides and a table exists in both, the Database File Overrides value takes precedence.</p>

Property	Description
Code Page	The code page used to read from a source database or write to a target database or file.
SQL Identifier character to use	Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type. Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers. Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character.
Support mixed case identifiers	Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the SQL Identifier Character property. When the SQL Identifier Character property is set to none, the Support Mixed-case Identifiers property is disabled.
Isolation level	Commit scope of the transaction. Select one of the following options: - None - CS . Cursor stability. - RR . Repeatable Read. - CHG . Change. - ALL Default is CS.
Encryption type	Optional. The type of encryption that the Data Integration Service uses. Select one of the following options: - None - AES Default is None. Note: Informatica recommends that you use Secure Sockets Layer (SSL) authentication instead of configuring the Encryption type and Encryption level connection properties. SSL authentication provides stricter security and is used by several Informatica products. For more information about implementing SSL authentication in a PowerExchange network, see the <i>PowerExchange Reference Manual</i> .
Encryption level	If you select AES for Encryption Type , select one of the following options to indicate the encryption level that the Data Integration Service uses: - 1 . Use a 128-bit encryption key. - 2 . Use a 192-bit encryption key. - 3 . Use a 256-bit encryption key. If you do not select AES for Encryption Type , this option is ignored. Default is 1.
Pacing size	Optional. Amount of data that the source system can pass to the PowerExchange Listener. Set the pacing size if an external application, database, or the Data Integration Service node is a bottleneck. User lower values for faster performance. The minimum value and default value is 0. A value of 0 provides the best performance.
Interpret as rows	Optional. Select this option to express the pacing size as a number of rows. Clear this option to express the pacing size in kilobytes. By default, this option is not selected and the pacing size is in kilobytes.
Compression	Optional. Select this option to enable source data compression. By compressing data, you can decrease the amount of data that Informatica applications send over the network. By default, this option is not selected and compression is disabled.

Property	Description
Array size	Optional. The number of records in the storage array for the worker threads. This option is applicable when you set the Worker Threads option to a value greater than 0. Valid values are 25 to 5000. Default is 25.
Write mode	Optional. Mode in which the Data Integration Service sends data to the PowerExchange Listener. Select one of the following write modes: <ul style="list-style-type: none"> - CONFIRMWRITEON. Sends data to the PowerExchange Listener and waits for a response before sending more data. Select this option when error recovery is a priority. However, this option might degrade performance. - CONFIRMWRITEOFF. Sends data to the PowerExchange Listener without waiting for a response. Use this option if you can reload the target table when an error occurs. - ASYNCHRONOUSWITHFAULTTOLERANCE. Sends data to the PowerExchange Listener without waiting for a response. This option also enables error detection. This option combines the speed of CONFIRMWRITEOFF and the data integrity of CONFIRMWRITEON. Default is CONFIRMWRITEON.
Reject file	Overrides the default prefix of PWXR for the reject file. PowerExchange creates the reject file on the target machine when the write mode is ASYNCHRONOUSWITHFAULTTOLERANCE. Enter PWXDISABLE to prevent the creation of the reject files.

IBM DB2 for z/OS Connection Properties

Use an IBM DB2 for z/OS connection to access tables in IBM DB2 for z/OS. An IBM DB2 for z/OS connection is a relational database connection. You can create and manage an IBM DB2 for z/OS 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 DB2 for z/OS connection properties:

Property	Description
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	Description of the connection. The description cannot exceed 255 characters.
Connection Type	Connection type (DB2Z).
Username	Database user name.

Property	Description
Password	<p>Password for the specified user name or a valid PowerExchange passphrase.</p> <p>A PowerExchange passphrase can be from 9 to 128 characters in length and can contain the following characters:</p> <ul style="list-style-type: none"> - Uppercase and lowercase letters - The numbers 0 to 9 - Spaces - The following special characters: ' - ; # \ , . / ! % & * () _ + { } : @ < > ? <p>Note: The first character is an apostrophe.</p> <p>Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.</p> <p>To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the <i>PowerExchange Reference Manual</i>.</p> <p>The allowable characters in the IBM IRRPHREX exit do not affect the allowable characters in PowerExchange passphrases.</p> <p>Note: A valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.</p>
Pass-through security enabled	Enables pass-through security for the connection.
DB2 Subsystem ID	Name of the DB2 subsystem.
Location	Node name for the location of the PowerExchange Listener that connects to DB2. The node name is defined in the first parameter of the NODE statement in the PowerExchange dbmover.cfg configuration file.
Environment SQL	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.
Correlation ID	Value to be concatenated to prefix PWX to form the DB2 correlation ID for DB2 requests.
Code Page	Code page used to read from a source database or write to a target database or file.
SQL identifier character to use	<p>Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type.</p> <p>Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers.</p> <p>Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character.</p>
Support mixed case identifiers	<p>Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the SQL Identifier Character property.</p> <p>When the SQL Identifier Character property is set to none, the Support Mixed-case Identifiers property is disabled.</p>

Property	Description
Encryption type	<p>Optional. The type of encryption that the Data Integration Service uses. Select one of the following options:</p> <ul style="list-style-type: none"> - None - AES <p>Default is None.</p> <p>Note: Informatica recommends that you use Secure Sockets Layer (SSL) authentication instead of configuring the Encryption Type and Level connection properties. SSL authentication provides stricter security and is used by several Informatica products.</p> <p>For more information about implementing SSL authentication in a PowerExchange network, see the <i>PowerExchange Reference Manual</i>.</p>
Encryption level	<p>If you select AES for Encryption Type, select one of the following options to indicate the encryption level that the Data Integration Service uses:</p> <ul style="list-style-type: none"> - 1. Use a 128-bit encryption key. - 2. Use a 192-bit encryption key. - 3. Use a 256-bit encryption key. <p>If you do not select AES for Encryption Type, this option is ignored.</p> <p>Default is 1.</p>
Pacing size	<p>Optional. Amount of data that the source system can pass to the PowerExchange Listener. Set the pacing size if an external application, database, or the Data Integration Service node is a bottleneck. User lower values for faster performance.</p> <p>The minimum value and default value is 0. A value of 0 provides the best performance.</p>
Interpret as rows	<p>Optional. Select this option to express the pacing size as a number of rows. Clear this option to express the pacing size in kilobytes. By default, this option is not selected and the pacing size is in kilobytes.</p>
Compression	<p>Optional. Select this option to enable source data compression. By compressing data, you can decrease the amount of data that Informatica applications send over the network. By default, this option is not selected and compression is disabled.</p>
Offload processing	<p>Optional. Controls whether to offload some bulk data processing from the source machine to the Data Integration Service machine. Select one of the following options:</p> <ul style="list-style-type: none"> - AUTO. The Data Integration Service determines whether to use offload processing. - Yes. Use offload processing. - No. Do not use offload processing. <p>Default is No.</p>
Worker threads	<p>Optional. Number of threads that the Data Integration Service uses to process bulk data when offload processing is enabled. For optimal performance, this value should not exceed the number of available processors on the Data Integration Service machine. Valid values are 1 through 64. Default is 0, which disables multithreading.</p>
Array size	<p>Optional. The number of records in the storage array for the worker threads. This option is applicable when you set the Worker Threads option to a value greater than 0. Valid values are 1 to 5000. Default is 25.</p>

Property	Description
Write mode	<p>Mode in which the Data Integration Service sends data to the PowerExchange Listener. Configure one of the following write modes:</p> <ul style="list-style-type: none"> - CONFIRMWRITEON. Sends data to the PowerExchange Listener and waits for a response before sending more data. Select if error recovery is a priority. This option might decrease performance. - CONFIRMWRITEOFF. Sends data to the PowerExchange Listener without waiting for a response. Use this option when you can reload the target table if an error occurs. - ASYNCHRONOUSWITHFAULTTOLERANCE. Sends data to the PowerExchange Listener without waiting for a response. This option also provides the ability to detect errors. This provides the speed of Confirm Write Off with the data integrity of Confirm Write On. Default is CONFIRMWRITEON.
Reject file	<p>Overrides the default prefix of PWXR for the reject file. PowerExchange creates the reject file on the target machine when the write mode is ASYNCHRONOUSWITHFAULTTOLERANCE. Enter PWXDISABLE to prevent the creation of reject files.</p>

IMS Connection Properties

Use an IMS connection to access an IMS database. The IMS connection is a non-relational mainframe database type connection. The Data Integration Service connects to IMS through PowerExchange. You create an IMS connection in the Developer tool. You can manage an IMS connection in the Administrator tool or the Developer tool.

The following table describes IMS connection properties:

Option	Description
Location	Node name for the location of the PowerExchange Listener that connects to IMS. The node name is defined in the first parameter of the NODE statement in the PowerExchange dbmover.cfg configuration file.
User name	Database user name.

Option	Description
Password	<p>Password for the specified database user name or a valid PowerExchange passphrase. A PowerExchange passphrase can be from 9 to 128 characters in length and can contain the following characters:</p> <ul style="list-style-type: none"> - Uppercase and lowercase letters - The numbers 0 to 9 - Spaces - The following special characters: ' - ; # \ , . / ! % & * () _ + { } : @ < > ? <p>Note: The first character is an apostrophe.</p> <p>Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.</p> <p>The allowable characters in the IBM IRRPHREX exit do not affect the allowable characters in PowerExchange passphrases.</p> <p>Note: A valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.</p> <p>To use passphrases for IMS connections, ensure that the following requirements are met:</p> <ul style="list-style-type: none"> - The PowerExchange Listener must run with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the <i>PowerExchange Reference Manual</i>. - You must configure ODBA access to IMS as described in the <i>PowerExchange Navigator User Guide</i>. - You must use IMS data maps that specify IMS ODBA as the access method. Do not use data maps that specify the DL/1 BATCH access method because this access method requires the use of netport jobs, which do not support passphrases. - The IMS database must be online in the IMS control region to use ODBA access to IMS.
Code page	<p>Required. Name of the code page to use for reading from or writing to the data source. Usually, this value is an ISO code page name, such as ISO-8859-6.</p>
Pass-through security enabled	<p>Enables pass-through security for the connection.</p>
Encryption type	<p>The type of encryption that the Data Integration Service uses. Select one of the following options:</p> <ul style="list-style-type: none"> - None - AES <p>Default is None.</p> <p>Note: Informatica recommends that you use Secure Sockets Layer (SSL) authentication instead of configuring the Encryption Type and Level connection properties. SSL authentication provides stricter security and is used by several Informatica products. For more information about implementing SSL authentication in a PowerExchange network, see the <i>PowerExchange Reference Manual</i>.</p>
[Encryption] Level	<p>If you select AES for Encryption Type, select one of the following options to indicate the encryption level that the Data Integration Service uses:</p> <ul style="list-style-type: none"> - 1. Use a 128-bit encryption key. - 2. Use a 192-bit encryption key. - 3. Use a 256-bit encryption key. <p>If you do not select AES for Encryption Type, this option is ignored.</p> <p>Default is 1.</p>

Option	Description
Pacing size	Optional. Amount of data that the source system can pass to the PowerExchange Listener. Set the pacing size if an external application, database, or the Data Integration Service node is a bottleneck. User lower values for faster performance. The minimum value and default value is 0. A value of 0 provides the best performance.
Interpret as rows	Optional. Select this option to express the pacing size as a number of rows. Clear this option to express the pacing size in kilobytes. By default, this option is not selected and the pacing size is in kilobytes.
Compression	Optional. Select this option to enable source data compression. By compressing data, you can decrease the amount of data that Informatica applications send over the network. By default, this option is not selected and compression is disabled.
Offload processing	Optional. Controls whether to offload some bulk data processing from the source machine to the Data Integration Service machine. Select one of the following options: <ul style="list-style-type: none"> - AUTO. The Data Integration Service determines whether to use offload processing. - Yes. Use offload processing. - No. Do not use offload processing. Default is AUTO.
Worker threads	Optional. Number of threads that the Data Integration Service uses to process bulk data when offload processing is enabled. For optimal performance, this value should not exceed the number of available processors on the Data Integration Service machine. Valid values are 1 through 64. Default is 0, which disables multithreading.
Array size	Optional. The number of records in the storage array for the worker threads. This option is applicable when you set the Worker Threads option to a value greater than 0. Valid values are 1 to 5000. Default is 25.
Write mode	Optional. Mode in which Data Integration Service sends data to the PowerExchange Listener. Select one of the following write modes: <ul style="list-style-type: none"> - CONFIRMWRITEON. Sends data to the PowerExchange Listener and waits for a response before sending more data. Select this option when error recovery is a priority. However, this option might degrade performance. - CONFIRMWRITEOFF. Sends data to the PowerExchange Listener without waiting for a response. Use this option if you can reload the target table when an error occurs. - ASYNCHRONOUSWITHFAULTTOLERANCE. Sends data to the PowerExchange Listener without waiting for a response. This option also enables error detection. This option combines the speed of CONFIRMWRITEOFF and the data integrity of CONFIRMWRITEON. Default is CONFIRMWRITEON.

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.

The following table describes JDBC connection properties:

Property	Description
Database Type	The database type.
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. The following list provides the driver class name that you can enter for the applicable database type: - DataDirect JDBC driver class name for Oracle: <code>com.informatica.jdbc.oracle.OracleDriver</code> - DataDirect JDBC driver class name for IBM DB2: <code>com.informatica.jdbc.db2.DB2Driver</code> - DataDirect JDBC driver class name for Microsoft SQL Server: <code>com.informatica.jdbc.sqlserver.SQLServerDriver</code> - DataDirect JDBC driver class name for Sybase ASE: <code>com.informatica.jdbc.sybase.SybaseDriver</code> - DataDirect JDBC driver class name for Informix: <code>com.informatica.jdbc.informix.InformixDriver</code> - DataDirect JDBC driver class name for MySQL: <code>com.informatica.jdbc.mysql.MySQLDriver</code> - JDBC driver for Databricks Delta Lake: the name of the driver that you downloaded from Databricks. For information about the driver, see the topic on configuring storage access in the "Before You Begin Databricks Integration" chapter of the <i>Data Engineering Integration Guide</i> . For more information about which driver class to use with specific databases, see the vendor documentation.
Connection String	Connection string to connect to the database. Use the following connection string: <code>jdbc:<subprotocol>:<subname></code> For more information about the connection string to use with specific drivers, see the vendor documentation.
Environment SQL	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. Note: If you enable Sqoop, Sqoop ignores this property.
Transaction SQL	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. Note: If you enable Sqoop, Sqoop ignores this property.

Property	Description
SQL Identifier Character	<p>Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type.</p> <p>Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers.</p> <p>Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character.</p> <p>Note: If you enable Sqoop, Sqoop ignores this property.</p>
Support Mixed-case Identifiers	<p>Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the SQL Identifier Character property.</p> <p>When the SQL Identifier Character property is set to none, the Support Mixed-case Identifiers property is disabled.</p> <p>Note: If you enable Sqoop, Sqoop honors this property when you generate and execute a DDL script to create or replace a target at run time. In all other scenarios, Sqoop ignores this property.</p>
Use Sqoop Connector	<p>Enables Sqoop connectivity for the data object that uses the JDBC connection. The Data Integration Service runs the mapping in the Hadoop run-time environment through Sqoop.</p> <p>You can configure Sqoop connectivity for relational data objects, customized data objects, and logical data objects that are based on a JDBC-compliant database.</p> <p>Select Sqoop v1.x to enable Sqoop connectivity.</p> <p>Default is None.</p>
Sqoop Arguments	<p>Enter the arguments that Sqoop must use to connect to the database. Separate multiple arguments with a space.</p> <p>To run the mapping on the Blaze engine with the Teradata Connector for Hadoop (TDCH) specialized connectors for Sqoop, you must define the TDCH connection factory class in the Sqoop arguments. The connection factory class varies based on the TDCH Sqoop Connector that you want to use.</p> <ul style="list-style-type: none"> - To use Cloudera Connector Powered by Teradata, configure the following Sqoop argument: <ul style="list-style-type: none"> - <code>Dsqoop.connection.factories=com.cloudera.connector.teradata.TeradataManagerFactory</code> - To use Hortonworks Connector for Teradata (powered by the Teradata Connector for Hadoop), configure the following Sqoop argument: <ul style="list-style-type: none"> - <code>Dsqoop.connection.factories=org.apache.sqoop.teradata.TeradataManagerFactory</code> <p>To run the mapping on the Spark engine, you do not need to define the TDCH connection factory class in the Sqoop arguments. The Data Integration Service invokes the Cloudera Connector Powered by Teradata and Hortonworks Connector for Teradata (powered by the Teradata Connector for Hadoop) by default.</p> <p>Note: To run the mapping with a generic JDBC connector instead of the specialized Cloudera or Hortonworks connector, you must define the <code>--driver</code> and <code>--connection-manager</code> Sqoop arguments in the JDBC connection. If you define the <code>--driver</code> and <code>--connection-manager</code> arguments in the Read or Write transformation of the mapping, Sqoop ignores the arguments.</p> <p>If you do not enter Sqoop arguments, the Data Integration Service constructs the Sqoop command based on the JDBC connection properties.</p>

JDBC V2 Connection Properties

When you set up a JDBC V2 connection, you must configure the connection properties.

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

The following table describes the JDBC V2 connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select JDBC V2.

The **Details** tab contains the connection attributes of the JDBC V2 connection. The following table describes the connection attributes:

Property	Description
Username	The database user name. User name with permissions to access the database that supports the Type 4 JDBC driver.
Password	The password for the database user name.
Schema Name	Optional. The schema name to connect in the database. If you do not specify the schema name, all the schemas available in the database are listed.
JDBC Driver Class Name	Name of the JDBC driver class. The following list provides the driver class name that you can enter for the applicable database type: <ul style="list-style-type: none">- JDBC driver class name for Azure SQL Database: <code>com.microsoft.sqlserver.jdbc.SQLServerDriver</code>- JDBC driver class name for Aurora PostgreSQL: <code>org.postgresql.Driver</code>- JDBC driver class name for SAP HANA Database: <code>com.sap.db.jdbc.Driver</code> For more information about which driver class to use with specific databases, see the third-party vendor documentation.

Property	Description
Connection String	<p>Connection string to connect to the database.</p> <p>Use the following connection string:</p> <pre>jdbc:<subprotocol>:<subname></pre> <p>The following list provides sample connection strings that you can enter for the applicable database type:</p> <ul style="list-style-type: none"> - Connection string for Azure SQL Database JDBC driver: <pre>jdbc:sqlserver://<host>:<port>;database=<database_name></pre> - Connection string for Aurora PostgreSQL JDBC driver: <pre>jdbc:postgresql://<host>:<port>[/<database_name>]</pre> - Connection string for SAP HANA Database driver: <pre>jdbc:sap://<host>:<port>/?databaseName=<Database_Name></pre> <p>For more information about the connection string to use with specific drivers, see the third-party vendor documentation.</p>
Sub Type	<p>The database type to which you want to connect.</p> <p>You can select from the following database types to connect:</p> <ul style="list-style-type: none"> - Azure SQL Database. Connects to Azure SQL database. - PostgreSQL. Connects to Aurora PostgreSQL database. - SAP HANA Database. Connects to SAP HANA database. - Others. Connects to any database that supports the Type 4 JDBC driver.
Support Mixed-case Identifiers	<p>Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the SQL Identifier Character property.</p> <p>For example, Aurora PostgreSQL database supports mixed-cased characters. You must enable this property to connect to the Aurora PostgreSQL database.</p> <p>When the SQL Identifier Character property is set to none, the Support Mixed-case Identifiers property is disabled.</p>
SQL Identifier Character	<p>Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type.</p> <p>Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers.</p> <p>Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character.</p> <p>Note: Select SQL Identifier Character as None when you specify the SAP HANA Database subtype.</p>

JD Edwards EnterpriseOne Connection Properties

Use a JD Edwards EnterpriseOne connection to connect to a JD Edwards EnterpriseOne object.

The following table describes the JD Edwards EnterpriseOne connection properties:

Property	Description
Name	The 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: ~ ` ! \$ % ^ & * () - + = { [] } \ : ; " ' < , > . ? /
ID	The 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.
Location	The Informatica domain where you want to create the connection.
Type	The connection type. Select JD Edwards EnterpriseOne.
Host Name	JD Edwards EnterpriseOne server host name.
Enterprise Port	JD Edwards EnterpriseOne server port number. Default is 6016.
User Name	The JD Edwards EnterpriseOne database user name.
Password	The password for the JD Edwards EnterpriseOne database user.
Environment	Name of the JD Edwards EnterpriseOne environment you want to connect to.
Role	Role of the JD Edwards EnterpriseOne user. Default is *ALL.
User Name	The JD Edwards EnterpriseOne database user name.
Password	Password for the database user.

Property	Description
Driver Class Name	<p>The following list provides the driver class name that you can enter for the applicable database type:</p> <ul style="list-style-type: none"> - DataDirect JDBC driver class name for Oracle: <code>com.informatica.jdbc.oracle.OracleDriver</code> - DataDirect JDBC driver class name for IBM DB2: <code>com.informatica.jdbc.db2.DB2Driver</code> - DataDirect JDBC driver class name for Microsoft SQL Server: <code>com.informatica.jdbc.sqlserver.SQLServerDriver</code> <p>For more information about which driver class to use with specific databases, see the vendor documentation.</p>
Connection String	<p>The connection string to connect to the database. Use the following connection string:</p> <p>The JDBC connection string uses the following syntax:</p> <ul style="list-style-type: none"> - For Oracle: <code>jdbc:informatica:oracle://<host name>:<port>,ServiceName=<db service name></code> - For DB2: <code>jdbc:informatica:db2://<host name>:<port>;databaseName=<db name></code> - For Microsoft SQL: <code>jdbc:informatica:sqlserver://<host name>:<port>;databaseName=<db name></code>

Kafka Connection Properties

The Kafka connection is a Messaging connection. Use the Kafka connection to access an Apache Kafka broker as a source or a target. You can create and manage a Kafka connection in the Developer tool or through infacmd.

When you configure a Kafka connection, you configure the following properties:

- The list of Kafka brokers that the connection reads from or writes to.
- The number of seconds the Integration Service attempts to reconnect to the database if the connection fails.
- Version of the Kafka messaging broker. Configure the Kafka messaging broker version to Apache 0.10.1.1 & above.

General Properties

The following table describes the general connection properties for the Kafka connection:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! \$ % ^ & * () - + = { [] } \ : ; " ' < , > . ? /
ID	The 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. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection. Select the domain name.
Type	The connection type. Select Messaging/Kafka.

Kafka Broker Properties

The following table describes the Kafka broker properties for the Kafka connection:

Property	Description
Kafka Broker List	Comma-separated list of Kafka brokers which maintain the configuration of the Kafka messaging broker. To specify a Kafka broker, use the following format: <IP Address>:<port>
Retry Timeout	Number of seconds after which the Integration Service attempts to reconnect to the Kafka broker to read or write data. If the source or target is not available for the time you specify, the mapping execution stops to avoid any data loss.

Property	Description
Kafka Broker Version	Configure the Kafka messaging broker version to Apache 0.10.1.1 & above.
Additional Connection Properties	<p>Optional. Comma-separated list of connection properties to connect to the Kafka broker.</p> <p>For example, you can use the following syntax:</p> <pre>request.timeout.ms=<value>,session.timeout.ms=<value>, fetch.max.wait.ms=<value>,heartbeat.interval.ms=<value>, security.protocol=SASL_PLAINTEXT,sasl.kerberos.service.name=<kerberos_name>,sasl.mechanism=GSSAPI, sasl.jaas.config=com.sun.security.auth.module.Krb5LoginModule;required useKeyTab=true doNotPrompt=true storeKey=true client=true keyTab="<Keytab Location>" principal="<principal>";</pre> <p>To reduce the time taken to connect to the Kafka broker, ensure that you set the following properties:</p> <ul style="list-style-type: none"> - request.timeout.ms - session.timeout.ms - fetch.max.wait.ms - heartbeat.interval.ms <p>To connect to the Kafka broker in a secured way, ensure that you set one of the following values for the <code>security.protocol</code> property:</p> <ul style="list-style-type: none"> - SASL_SSL - SSL <p>The default value of <code>security.protocol</code> property is SASL_PLAINTEXT.</p> <p>Technical Preview: The Additional Connection Properties is available for technical preview. Technical preview functionality is supported but is unwarranted and is not production-ready. Informatica recommends that you use in non-production environments only.</p> <p>For more information about the connection properties, see https://kafka.apache.org/documentation/.</p>

SSL Properties

The following table describes the SSL properties for the Kafka connection:

Property	Description
SSL Mode	<p>Required. SSL mode indicates the encryption type to use for the connection.</p> <p>You can choose a mode from the following SSL modes:</p> <ul style="list-style-type: none"> - Disabled - One way - Two way
SSL TrustStore File Path	<p>Required when One way SSL mode is selected.</p> <p>Absolute path and file name of the SSL truststore file that contains certificates of the trusted SSL server.</p>
SSL TrustStore Password	<p>Required when One way SSL mode is selected.</p> <p>Password for the SSL truststore.</p>

Property	Description
SSL KeyStore File Path	Required when Two way SSL mode is selected. Absolute path and file name of the SSL keystore file that contains private keys and certificates for the SSL server.
SSL KeyStore Password	Required when Two way SSL mode is selected. Password for the SSL keystore.

Creating a Kafka Connection Using infacmd

You can use the infacmd command line program to create a Kafka connection.

To create a Kafka connection on UNIX, run the following command:

```
sh infacmd.sh createConnection -dn <domain name> -un <domain user> -pd <domain password> -cn
<connection name> -cid <connection id> -ct Kafka -o
kfkBrkList=<host1:port1>,<host2:port2>,<host3:port3> kafkabrokerverversion=<version>
additionalConnectionProperties=<additional properties>
```

For more information about the CreateConnection command, see the *Informatica Command Reference*.

Kudu Connection Properties

Use a Kudu connection to access Kudu.

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

You can create and manage a Kudu connection in the Administrator tool or the Developer tool. The following table describes the Kudu connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Kudu.

The following table describes the properties for metadata access:

Property	Description
Kudu Master URLs	The URLs of the Kudu master tables.
Kudu Library Version	The version number of the Kudu library.
Cluster Configuration	The Hadoop cluster that you use for the connection.

LDAP Connection Properties

Use an LDAP connection to connect to an LDAP object.

The following table describes the LDAP connection properties:

Property	Description
Name	The 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: ~ ` ! \$ % ^ & * () - + = { [] \ : ; " ' < , > . ? /
ID	The 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.
Location	The Informatica domain where you want to create the connection.
Type	The connection type. Select LDAP.
Host Name	LDAP directory server host name. Default is localhost.
Port	LDAP directory server port number. Default is 389.
Anonymous Connection	Establishes an anonymous connection with the LDAP directory server. Select anonymous connection to access a directory server as an anonymous user without authentication. Note: You cannot establish an anonymous connection with Active Directory.
User Name	The LDAP user name to connect to the LDAP directory server.
Password	The password to connect to the LDAP directory server.
Secure Connection	Establishes a secure connection with the LDAP directory server through the TLS protocol.

Property	Description
TrustStore File Name	The file name of the truststore that contains the TLS certificate to establish a secure connection with the LDAP directory server. Default is <code>infa_truststore.jks</code> . Required if you select Secure Connection. Contact the LDAP Administrator for the truststore file name and password.
TrustStore Password	The password for the truststore file that contains the SSL certificate.
KeyStore File Name	The file name of the keystore that contains the keys and certificates required to establish a secure communication with the LDAP directory server. Required if you select Secure Connection. Contact the LDAP Administrator for the keystore file name and password.
KeyStore Password	The password for the keystore file required for secure communication.

Microsoft Azure Blob Storage Connection Properties

Use a Microsoft Azure SQL Blob Storage connection to access a Microsoft Azure Blob Storage.

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

You can create and manage a Microsoft Azure Blob Storage connection in the Administrator tool or the Developer tool. The following table describes the Microsoft Azure Blob Storage connection properties:

Property	Description
Name	Name of the Microsoft Azure Blob Storage connection.
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	Description of the connection.
Location	The domain where you want to create the connection.
Type	Type of connection. Select Azure Blob Storage.

The **Connection Details** tab contains the connection attributes of the Microsoft Azure Blob Storage connection. The following table describes the connection attributes:

Property	Description
Account Name	Name of the Microsoft Azure Storage account.
Authorization Type	Authorization type. You can select any of the following authorization mechanisms: <ul style="list-style-type: none"> - Shared Key Authorization - Shared Access Signatures
Account Key	Microsoft Azure Storage access key. Applies when you select shared key authorization.
SAS Token	SAS URI with SAS token that you generate on Microsoft Azure portal for your account. Applies when you select shared access signature authorization type. <p>Note: You must provide a valid SAS URI with a valid SAS token.</p>
Container Name	The root container or sub-folders with the absolute path. <p>Note: To import complex files, specify only the root container.</p>
Endpoint Suffix	Type of Microsoft Azure end-points. You can select any of the following end-points: <ul style="list-style-type: none"> - <code>core.windows.net</code>: Default - <code>core.usgovcloudapi.net</code>: To select the US government Microsoft Azure end-points - <code>core.chinacloudapi.cn</code>: Not applicable

Microsoft Azure Cosmos DB SQL API Connection Properties

Use a Microsoft Azure Cosmos DB connection to connect to the Cosmos DB database. When you create a Microsoft Azure Cosmos DB connection, you enter information for metadata and data access.

The following table describes the Microsoft Azure Cosmos DB connection properties:

Property	Description
Name	Name of the Cosmos DB connection.
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	Description of the connection. The description cannot exceed 765 characters.
Location	The project or folder in the Model repository where you want to store the Cosmos DB connection.
Type	Select Microsoft Azure Cosmos DB SQL API.
Cosmos DB URI	The URI of Microsoft Azure Cosmos DB account.

Property	Description
Key	The primary and secondary key to which provides you complete administrative access to the resources within Microsoft Azure Cosmos DB account.
Database	Name of the database that contains the collections from which you want to read or write JSON documents.

Note: You can find the Cosmos DB URI and Key values in the **Keys** settings on Azure portal. Contact your Azure administrator for more details.

Microsoft Azure Data Lake Storage Gen1 Connection Properties

Use a Microsoft Azure Data Lake Storage Gen1 connection to access a Microsoft Azure Data Lake Storage Gen1.

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

You can create and manage a Microsoft Azure Data Lake Storage Gen1 connection in the Administrator tool or the Developer tool. The following table describes the Microsoft Azure Data Lake Storage Gen1 connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Microsoft Azure Data Lake Storage Gen1.

The following table describes the properties for metadata access:

Property	Description
ADLS Account Name	The name of the Microsoft Azure Data Lake Storage Gen1.
ClientID	The ID of your application to complete the OAuth Authentication in the Active Directory.
Client Secret	The client secret key to complete the OAuth Authentication in the Active Directory.

Property	Description
Directory	The Microsoft Azure Data Lake Storage Gen1 directory that you use to read data or write data. The default is root directory.
AuthEndpoint	The OAuth 2.0 token endpoint from where access code is generated based on based on the Client ID and Client secret is completed.

For more information about creating a client ID, client secret, and auth end point, contact the Azure administrator or see Microsoft Azure Data Lake Storage Gen1 documentation.

Microsoft Azure Data Lake Storage Gen2 Connection Properties

Use a Microsoft Azure Data Lake Storage Gen2 connection to access a Microsoft Azure Data Lake Storage Gen2.

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

You can create and manage a Microsoft Azure Data Lake Storage Gen2 connection in the Administrator tool or the Developer tool. The following table describes the Microsoft Azure Data Lake Storage Gen2 connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Microsoft Azure Data Lake Storage Gen2.

The following table describes the properties for metadata access:

Property	Description
Account Name	The Microsoft Azure Data Lake Storage Gen2 account name or the service name.
Authentication Type	Authentication type to access the Microsoft Azure Data Lake Storage Gen2 account. Select one of the following options: <ul style="list-style-type: none"> - Service Principal Authentication. Select to use the client ID, client secret, and tenant ID to connect to Microsoft Azure Data Lake Storage Gen2. - Shared Key Authentication. Select to use the account key to connect to Microsoft Azure Data Lake Storage Gen2. <p>Note: You cannot use shared key authentication in a streaming mapping.</p>
Client ID	Applicable for Service Principal Authentication. The ID of your application to complete the OAuth Authentication in the Azure Active Directory (AD).
Client Secret	Applicable for Service Principal Authentication. The client secret key to complete the OAuth Authentication in the Azure Active Directory.
Tenant ID	Applicable for Service Principal Authentication. The Directory ID of the Azure Active Directory.
Account Key	Applicable for Shared Key Authentication. The account key for the Microsoft Azure Data Lake Storage Gen2 account.
File System Name	The name of an existing file system in the Microsoft Azure Data Lake Storage Gen2.
Directory Path	The path of an existing directory without the file system name. There is no default directory. You can select one of the following syntax: <ul style="list-style-type: none"> - / for root directory. - /dir1 - dir1/dir2
Adls Gen2 End-point	Type of Microsoft Azure endpoints. You can select any of the following endpoints: <ul style="list-style-type: none"> - <code>core.windows.net</code>: Default - <code>core.usgovcloudapi.net</code>: To select the Azure Government endpoints

For more information about creating a client ID, client secret, tenant ID, and file system name, contact the Azure administrator or see Microsoft Azure Data Lake Storage Gen2 documentation.

Microsoft Azure SQL Data Warehouse Connection Properties

Use a Microsoft Azure SQL Data Warehouse connection to access a Microsoft Azure SQL Data Warehouse.

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

You can create and manage a Microsoft Azure SQL Data Warehouse connection in the Administrator tool or the Developer tool. The following table describes the Microsoft Azure SQL Data Warehouse connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Azure SQL Data Warehouse.

The following table describes the properties for metadata access:

Property	Description
Azure DW JDBC URL	Microsoft Azure Data Warehouse JDBC connection string. For example, you can enter the following connection string: <code>jdbc:sqlserver:// <Server>.database.windows.net:1433;database=<Database></code> The Administrator can download the URL from Microsoft Azure portal.
Azure DW JDBC Username	User name to connect to the Microsoft Azure SQL Data Warehouse account. You must have permission to read, write, and truncate data in Microsoft Azure SQL Data Warehouse.
Azure DW JDBC Password	Password to connect to the Microsoft Azure SQL Data Warehouse account.
Azure DW Schema Name	Name of the schema in Microsoft Azure SQL Data Warehouse.
Azure Storage Type	Type of Azure storage to stage the files. You can select any of the following storage type: - Azure Blob. Default. To use Microsoft Azure Blob Storage to stage the files. - ADLS Gen2. To use Microsoft Azure Data Lake Storage Gen2 as storage to stage the files.
Azure Blob Account Name	Name of the Microsoft Azure Storage account to stage the files.
Azure Blob Account Key	The key that authenticates the access to the Blob storage account.
ADLS Gen2 Storage Account Name	Name of the Microsoft Azure Data Lake Storage Gen2 account to stage the files.
ADLS Gen2 Account Key	Microsoft Azure Data Lake Storage Gen2 access key to stage the files.

Property	Description
Blob End-point	Type of Microsoft Azure endpoints. Select one of the following options: <ul style="list-style-type: none"> - <code>core.windows.net</code>: Default - <code>core.usgovcloudapi.net</code>: Select to access the US government Microsoft Azure Data Warehouse endpoints. - <code>core.chinacloudapi.cn</code>: Select to access a Microsoft Azure Data Warehouse endpoint in the China region. <p>You can configure the US government Microsoft Azure end-points when a mapping runs in the native environment and on the Spark engine.</p>
VNet Rule	Enable to connect to a Microsoft Azure SQL Data Warehouse endpoint residing in a virtual network (VNet).

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:

Property	Description
Database Type	The database type.
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: <code>~ ` ! \$ % ^ & * () - + = { [] } \ : ; " ' < , > . ? /</code>
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.
Use trusted connection	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. Note: Windows and NTLM authentication is not certified for a Microsoft SQL Server 2017 version hosted on Linux.
User Name	The database user name. Required if Microsoft SQL Server uses NTLMv1 or NTLMv2 authentication.
Password	The password for the database user name. Required if Microsoft SQL Server uses NTLMv1 or NTLMv2 authentication.

Property	Description
Pass-through security enabled	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.
Metadata Access Properties: Connection String	<p>Connection string used to access metadata from the database.</p> <p>Use the following connection string:</p> <pre>jdbc:informatica:sqlserver://<host name>:<port>;DatabaseName=<database name></pre> <p>To test the connection with NTLM authentication, include the following parameters in the connection string:</p> <ul style="list-style-type: none"> - AuthenticationMethod. The NTLM authentication version to use. <p>Note: UNIX supports NTLMv1 and NTLMv2 but not NTLM.</p> <ul style="list-style-type: none"> - Domain. The domain that the SQL server belongs to. <p>The following example shows the connection string for a SQL server that uses NTLMv2 authentication in a NT domain named Informatica.com:</p> <pre>jdbc:informatica:sqlserver://host01:1433;DatabaseName=SQL1;AuthenticationMethod=ntlm2java;Domain=Informatica.com</pre> <p>If you connect with NTLM authentication, you can enable the Use trusted connection option in the MS SQL Server connection properties. If you connect with NTLMv1 or NTLMv2 authentication, you must provide the user name and password in the connection properties.</p>

Property	Description
AdvancedJDBCSecurityOptions	<p>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.</p> <p>To connect to a secure database, include the following parameters:</p> <ul style="list-style-type: none"> - 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. <p>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.</p> <p>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.</p> <ul style="list-style-type: none"> - 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. - cryptoProtocolVersion. Optional. If you enable TLS for the Microsoft SQL Server instance, set the cryptoProtocolVersion parameter as follows: cryptoProtocolVersion=TLsv<version number>. For example, cryptoProtocolVersion=TLsv1.2 <p>Note: The version number must be the same as the TLS version you configured for the server.</p> <ul style="list-style-type: none"> - TrustStore. Required. Path and file name of the truststore file. <p>Note: If you configure SSL or TLS and specify only the file name, you must copy the truststore file to the Informatica installation directory. To test the connection and import metadata, copy the truststore file to the following directory:</p> <pre data-bbox="646 1163 1281 1213"><Informatica client installation directory>/clients/DeveloperClient</pre> <p>To run the mapping, copy the truststore file to the following directory:</p> <pre data-bbox="646 1266 1305 1287"><Informatica server installation directory>/tomcat/bin</pre> <ul style="list-style-type: none"> - TrustStorePassword. Required. Password for the truststore file for the secure database. <p>Not applicable for ODBC.</p> <p>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: Provider Type	<p>The connection provider that you want to use to connect to the Microsoft SQL Server database.</p> <p>You can select the following provider types:</p> <ul style="list-style-type: none"> - ODBC - Oldeb(Deprecated) <p>Default is ODBC.</p> <p>Note: Although the Microsoft SQL Server connection user interface shows the OLEDB provider type as deprecated, Informatica supports the OLEDB provider type. For more information about the OLEDB provider type support statement, see the following Knowledge Base article KB 522895.</p>

Property	Description
Use DSN	Enables the Data Integration Service to use the Data Source Name for the connection. If you select the Use DSN option, the Data Integration Service retrieves the database and server names from the DSN. If you do not select the Use DSN option, you must provide the database and server names.
Connection String	Use the following connection string if you do not enable DSN mode: <server name>@<database name> If you enable DSN mode, use the following connection strings: <DSN Name>
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. Note: When you generate a table DDL through a dynamic mapping or through the Generate and Execute DDL option, the DDL metadata does not include schema name and owner name properties.
Schema Name	The name of the schema in the database. You must specify the schema name for the Profiling Warehouse 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 if you configure user-managed cache tables. Note: When you generate a table DDL through a dynamic mapping or through the Generate and Execute DDL option, the DDL metadata does not include schema name and owner name properties.
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	Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type. Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers. Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character.

Property	Description
Support Mixed-case Identifiers	<p>Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the SQL Identifier Character property.</p> <p>When the SQL Identifier Character property is set to none, the Support Mixed-case Identifiers property is disabled.</p>
ODBC Provider	<p>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:</p> <ul style="list-style-type: none"> - Other - Sybase - Microsoft_SQL_Server <p>Default is Other.</p>

Netezza Connection Properties

Use a Netezza connection to access a Netezza database. The Netezza connection is a database connection. You can create and manage a Netezza 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 the Netezza connection properties:

Property	Description
Name	<p>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:</p> <p>~ ` ! \$ % ^ & * () - + = { [] \ : ; " ' < , > . ? /</p>
ID	<p>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.</p>
Description	<p>Description of the connection. The description cannot exceed 765 characters.</p>
Location	<p>Domain where you want to create the connection.</p>
Type	<p>Connection type. Select Netezza.</p>
User name	<p>User name with the appropriate permissions to access the Netezza database.</p>
Password	<p>Password for the database user name.</p>
JDBC Url	<p>JDBC URL that the Developer tool must use when it connects to the Netezza database. Use the following format:</p> <p><code>jdbc:netezza://<hostname>:<port>/<database name></code></p>

Property	Description
Connection String	Name of the ODBC data source that you want to use to connect to the Netezza database.
Timeout	Number of seconds that the Developer tool waits for a response from the Netezza database before it closes the connection.

OData Connection Properties

Use an OData connection to access an OData URL. The OData connection is a Web connection. You can create and manage an OData 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 the OData connection properties:

Property	Description
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	Description of the connection. The description cannot exceed 4,000 characters.
Location	Domain where you want to create the connection.
Type	Connection type. Select OData .
User name	Optional. User name with the appropriate permissions to read data from the OData resource.
Password	Optional. Password for the OData URL user name.
URL	OData service root URL that exposes the data that you want to read.
Security Type	Optional. Security protocol that the Developer tool must use to establish a secure connection with the OData server. Select one of the following values: - None - SSL - TLS Default is None.
TrustStore File Name	Required if you select a security type. Name of the truststore file that contains the public certificate for the OData server. Default is <code>infa_truststore.jks</code> .

Property	Description
Password	Required if you select a security type. Password for the truststore file that contains the public certificate for the OData server.
KeyStore File Name	Required if you select a security type. Name of the keystore file that contains the private key for the OData server. Default is <code>infa_truststore.jks</code> .
Password	Required if you select a security type. Password for the keystore file that contains the private key for the OData server.

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.

The following table describes ODBC connection properties:

Property	Description
Database Type	The database type.
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.
Pass-through security enabled	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.
Data Access Properties: Connection String	The ODBC connection URL used to access metadata from the database. <data source name>
Code Page	The code page used to read from a source database or to write to a target database or file.

Property	Description
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	Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type. Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers. Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character.
Support Mixed-case Identifiers	Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the SQL Identifier Character property. When the SQL Identifier Character property is set to none, the Support Mixed-case Identifiers property is disabled.
ODBC Provider	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: <ul style="list-style-type: none"> - Other - Sybase - Microsoft_SQL_Server - Snowflake Default is Other.

Note: Use an ODBC connection to connect to Microsoft SQL Server when the Data Integration Service runs on UNIX or Linux. Use a native connection to Microsoft SQL Server when the Data Integration Service runs on Windows.

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.

The following table describes Oracle connection properties:

Property	Description
Database Type	The database type.
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.
Pass-through security enabled	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.
Metadata Access Properties: Connection String	<p>Connection string used to access metadata from the database.</p> <p>Use the following connection string:</p> <pre>jdbc:informatica:oracle://<host_name>:<port>;SID=<database name></pre> <p>Use the following connection string to connect to Oracle database through Oracle Connection Manager:</p> <pre>jdbc:informatica:oracle:TNSNamesFile=<fully qualified path to the tnsnames.ora file>;TNSServerName=<TNS server name>;</pre>

Property	Description
AdvancedJDBCSecurityOptions	<p>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.</p> <p>To connect to a secure database, include the following parameters:</p> <ul style="list-style-type: none"> - 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. <p>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.</p> <p>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.</p> <ul style="list-style-type: none"> - 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. - cryptoProtocolVersion. Optional. If you enable TLS for the Oracle instance, set the cryptoProtocolVersion parameter as follows: cryptoProtocolVersion=TLSv<version number>. For example, cryptoProtocolVersion=TLSv1.2 <p>Note: The version number must be the same as the TLS version you configured for the server.</p> <ul style="list-style-type: none"> - TrustStore. Required. Path and file name of the truststore file. <p>Note: If you configure SSL or TLS and specify only the file name, you must copy the truststore file to the Informatica installation directory. To test the connection and import metadata, copy the truststore file to the following directory:</p> <pre data-bbox="646 1163 1284 1213"><Informatica client installation directory>/clients/DeveloperClient</pre> <p>To run the mapping, copy the truststore file to the following directory:</p> <pre data-bbox="646 1266 1308 1289"><Informatica server installation directory>/tomcat/bin</pre> <ul style="list-style-type: none"> - TrustStorePassword. Required. Password for the truststore file for the secure database. - KeyStore. Required. Path and file name of the keystore file. - KeyStorePassword. Required. Password for the keystore file for the secure database. <p>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	<p>Use the following connection string:</p> <pre data-bbox="621 1608 881 1631"><database name>.world</pre>
Code Page	<p>The code page used to read from a source database or to write to a target database or file.</p>
Environment SQL	<p>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.</p>

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.
Enable Parallel Mode	Enables parallel processing when loading data into a table in bulk mode. By default, this option is cleared.
SQL Identifier Character	Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type. Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers. Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character.
Support Mixed-case Identifiers	Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the SQL Identifier Character property. When the SQL Identifier Character property is set to none, the Support Mixed-case Identifiers property is disabled.

Salesforce Connection Properties

Use a Salesforce connection to connect to a Salesforce object. The Salesforce connection is an application connection type. You can create and manage a Salesforce 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 Salesforce connection properties:

Property	Description
Name	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: ~ ` ! \$ % ^ & * () - + = { [] \ : ; " ' < , > . ? /
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.
Location	The Informatica domain where you want to create the connection.

Property	Description
Type	The connection type. You can select the Standard connection type or OAuth connection type.
User Name	Applicable for Standard connection type. Salesforce user name.
User Password	Applicable for Standard connection type. Password for the Salesforce user name. To access Salesforce outside the trusted network of your organization, you must append a security token to your password to log in to the API or a desktop client. To receive or reset your security token, log in to Salesforce and click Setup > My Personal Information > Reset My Security Token . Password is case sensitive.
Service URL	URL of the Salesforce service you want to access. For example, <code>https://login.salesforce.com/services/Soap/u/54.0</code> In a test or development environment, you might want to access the Salesforce Sandbox testing environment. For more information about the Salesforce Sandbox, see the Salesforce documentation.
Refresh Token	Applicable for OAuth connection type. Refresh Token of Salesforce.
Consumer Key	Applicable for OAuth connection type. The Consumer Key obtained from Salesforce, required to generate the Refresh Token. For more information about how to generate the Consumer Key, see the Salesforce documentation.
Consumer Secret	Applicable for OAuth connection type. The Consumer Secret obtained from Salesforce, required to generate the Refresh Token. For more information about how to generate the Consumer Secret, see the Salesforce documentation.

Salesforce Marketing Cloud Connection Properties

Use a Salesforce Marketing Cloud connection to connect to a Salesforce Marketing Cloud object. You can create and manage a Salesforce Marketing Cloud 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 the Salesforce Marketing Cloud connection properties:

Connection property	Description
Name	Name of the Salesforce Marketing Cloud connection.
ID	The Data Integration Service uses the ID to identify the connection.
Description	Optional. The description of the connection.
Location	Informatica domain where you want to create the connection.

Connection property	Description
Type	Connection type. Select Salesforce Marketing Cloud.
Salesforce Marketing Cloud Url	<p>The URL that the Data Integration Service uses to connect to the Salesforce Marketing Cloud WSDL.</p> <p>The following URL is an example for OAuth 1.0 URL: <code>https://webservice.s7.exacttarget.com/etframework.wsdl</code></p> <p>The following URL is an example for OAuth 2.0 URL: <code>https://<SUBDOMAIN>.soap.marketingcloudapis.com/etframework.wsdl</code></p> <p>Informatica recommends that you upgrade to OAuth 2.0 before Salesforce Marketing Cloud drops support for OAuth 1.0.</p>
Username	User name of the Salesforce Marketing Cloud account.
Password	Password for the Salesforce Marketing Cloud account.
ClientId	The client ID of Salesforce Marketing Cloud required to generate a valid access token.
ClientSecret	The client secret of Salesforce Marketing Cloud required to generate a valid access token.
Enable Logging	When you enable logging you can see the session log for the tasks.
UTC Offset	The Secure Agent uses the UTC offset connection property to read data from and write data to Salesforce Marketing Cloud in UTC offset time zone.
Batch Size	<p>Number of rows that the Secure Agent writes in a batch to the target.</p> <p>When you insert or update data and specify the contact key, the data associated with the specified contact ID is inserted or updated in a batch to Salesforce Marketing Cloud. When you upsert data to Salesforce Marketing Cloud, do not specify the contact key.</p>
Enable Multiple BU	Select this option if there are multiple business units in your Salesforce Marketing Cloud account. You can use the Salesforce Marketing Cloud connection to access data across all business units.

SAP Connection Properties

Use an SAP connection to access an SAP table or an SAP BW object. The SAP connection is an enterprise application connection. You can create and manage an SAP 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 the SAP connection properties:

Property	Description
Username	Required. User name for the SAP source system that you want to access.
Password	Required. Password for the user name.
Connection type	<p>Required. Type of connection that you want to create.</p> <p>Select one of the following values:</p> <ul style="list-style-type: none"> - Application. Create an application connection when you want to connect to a specific SAP application server. - Load balancing. Create a load balancing connection when you want to use SAP load balancing. <p>Default is Application.</p> <p>Based on the connection type you select, the corresponding connection property fields become available in the Connection Details dialog box. The Developer tool greys out the connection property fields that are not applicable for a particular connection type.</p>
Host name	<p>Required when you create an SAP application connection.</p> <p>Host name or IP address of the SAP server that you want to connect to.</p>
System number	<p>Required when you create an SAP application connection.</p> <p>SAP system number.</p>
Message host name	<p>Required when you create an SAP load balancing connection.</p> <p>Host name of the SAP message server.</p>
R3 name/SysID	<p>Required when you create an SAP load balancing connection.</p> <p>Name of the SAP system.</p>
Group	<p>Required when you create an SAP load balancing connection.</p> <p>Group name of the SAP application server.</p>
Client	Required. SAP client number.
Language	<p>Optional. Language that you want to use for mappings and workflows.</p> <p>Must be compatible with the Developer tool code page.</p> <p>If you leave this option blank, the Developer tool uses the default language of the SAP system.</p>
Trace	<p>Optional. Use this option to track the JCo calls that the SAP system makes. SAP stores the information about the JCo calls in a trace file.</p> <p>Specify one of the following values:</p> <ul style="list-style-type: none"> - 0. Off - 1. Full <p>Default is 0.</p> <p>You can access the trace files from the following directories:</p> <ul style="list-style-type: none"> - <Informatica installation directory>/tomcat/bin directory on the machine where you installed the Informatica services - <Informatica installation directory>/clients/DeveloperClient directory on the machine where you installed the Developer tool
Additional parameters	<p>Optional. Enter any other connection parameter that you want to use.</p> <p>Use the following format:</p> <p><parameter name>=<value></p>

Property	Description
Staging directory	Path in the SAP system where the stage file will be created.
Source directory	Path that contains the source file. The path must be accessible to the Data Integration Service.
Use FTP	Enables FTP access to SAP.
FTP user	Required when you use FTP. User name to connect to the FTP server.
FTP password	Required when you use FTP. Password for the FTP user.
FTP host	Required when you use FTP. Host name or IP address of the FTP server. Optionally, you can specify a port number from 1 through 65535, inclusive. Default for FTP is 21. Use one of the following syntax to specify the host name: - hostname:port_number - IP address:port_number When you specify a port number, enable that port number for FTP on the host machine. If you enable SFTP, specify a host name or port number for an SFTP server. Default for SFTP is 22.
Retry period	Number of seconds that the Data Integration Service attempts to reconnect to the FTP host if the connection fails. If the Data Integration Service cannot reconnect to the FTP host in the retry period, the mapping or workflow fails. Default is 0. A value of 0 indicates an infinite retry period.
Use SFTP	Enables SFTP access to SAP.
Public key file name	Required when you enable SFTP and the SFTP server uses public key authentication. Public key file path and file name.
Private key file name	Required when you enable SFTP and the SFTP server uses public key authentication. Private key file path and file name.
Private key file name password	Required when you enable SFTP, and the SFTP server uses public key authentication and the private key is encrypted. Password to decrypt the private key file.
Port Range	HTTP port range that the Data Integration Service must use to read data from the SAP server in streaming mode. Enter the minimum and maximum port numbers with a hyphen as the separator. The minimum and maximum port number can range between 10000 and 65535. You can also specify the port range according to your organization. Default is 10000-65535.

Property	Description
Use HTTPS	Select this option to enable HTTPS streaming when you read data from SAP tables. By default, the Use HTTPS check box is not selected. For more information about configuring HTTPS for table reader mappings in streaming mode, see the article "HTTPS Configuration for Table Reader Mappings in Streaming Mode for PowerExchange for SAP NetWeaver" on the Informatica Documentation Portal.
Key store file path	Required when you use HTTPS. Path to the keystore file that contains the private or public key pairs and the associated certificates.
Key store password	Required when you use HTTPS. Password for the keystore file.
Private key password	Required when you use HTTPS. Password to decrypt the private key file.

Sequential Connection Properties

Use a Sequential connection to access sequential data sources. You create a Sequential connection in the Developer tool. You can manage a Sequential connection in the Administrator tool or the Developer tool.

A sequential data source is a data source that PowerExchange can access by using a data map defined with an access method of SEQ. The Data Integration Service connects to the data source through PowerExchange.

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

The following table describes Sequential connection properties:

Option	Description
Location	Node name for the location of the PowerExchange Listener that connects to the sequential data set. The node name is defined in the first parameter of the NODE statement in the PowerExchange dbmover.cfg configuration file.
User name	A user name that has the authority to access the sequential data set.

Option	Description
Password	<p>Password for the specified user name or a valid PowerExchange passphrase.</p> <p>A PowerExchange passphrase can be from 9 to 128 characters in length and can contain the following characters:</p> <ul style="list-style-type: none"> - Uppercase and lowercase letters - The numbers 0 to 9 - Spaces - The following special characters: ' - ; # \ , . / ! % & * () _ + { } : @ < > ? <p>Note: The first character is an apostrophe.</p> <p>Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.</p> <p>To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the <i>PowerExchange Reference Manual</i>.</p> <p>The allowable characters in the IBM IRRPHREX exit do not affect the allowable characters in PowerExchange passphrases.</p> <p>Note: A valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.</p>
Code page	<p>Required. Name of the code page to use for reading from or writing to the sequential data set. Usually, this value is an ISO code page name, such as ISO-8859-6.</p>
Pass-through security enabled	<p>Enables pass-through security for the connection.</p>
Encryption type	<p>Optional. The type of encryption that the Data Integration Service uses. Select one of the following options:</p> <ul style="list-style-type: none"> - None - AES <p>Default is None.</p> <p>Note: Informatica recommends that you use Secure Sockets Layer (SSL) authentication instead of configuring the Encryption Type and Level connection properties. SSL authentication provides stricter security and is used by several Informatica products. For more information about implementing SSL authentication in a PowerExchange network, see the <i>PowerExchange Reference Manual</i>.</p>
[Encryption] Level	<p>If you select AES for Encryption Type, select one of the following options to indicate the encryption level that the Data Integration Service uses:</p> <ul style="list-style-type: none"> - 1. Use a 128-bit encryption key. - 2. Use a 192-bit encryption key. - 3. Use a 256-bit encryption key. <p>If you do not select AES for Encryption Type, this option is ignored.</p> <p>Default is 1.</p>
Pacing size	<p>Optional. Amount of data that the source system can pass to the PowerExchange Listener. Set the pacing size if an external application, database, or the Data Integration Service node is a bottleneck. User lower values for faster performance.</p> <p>Minimum value and default value is 0. A value of 0 provides the best performance.</p>
Interpret as rows	<p>Optional Select this option to express the pacing size as a number of rows. Clear this option to express the pacing size in kilobytes. By default, this option is not selected and the pacing size is in kilobytes.</p>

Option	Description
Compression	Optional. Select this option to enable source data compression. By compressing data, you can decrease the amount of data that Informatica applications send over the network. By default, this option is not selected and compression is disabled.
Offload processing	Optional. Controls whether to offload some bulk data processing from the source machine to the Data Integration Service machine. Select one of the following options: <ul style="list-style-type: none"> - AUTO. The Data Integration Service determines whether to use offload processing. - Yes. Use offload processing. - No. Do not use offload processing. Default is AUTO.
Worker threads	Optional. Number of threads that the Data Integration Service uses to process bulk data when offload processing is enabled. For optimal performance, this value should not exceed the number of available processors on the Data Integration Service machine. Valid values are 1 through 64. Default is 0, which disables multithreading.
Array size	Optional The number of records in the storage array for the worker threads. This option is applicable when you set the Worker Threads option to a value greater than 0. Valid values are 25 to 5000. Default is 25.
Write mode	Optional. Mode in which the Data Integration Service sends data to the PowerExchange Listener. Select one of the following write modes: <ul style="list-style-type: none"> - CONFIRMWRITEON. Sends data to the PowerExchange Listener and waits for a response before sending more data. Select this option when error recovery is a priority. However, this option might degrade performance. - CONFIRMWRITEOFF. Sends data to the PowerExchange Listener without waiting for a response. Use this option if you can reload the target table when an error occurs. - ASYNCHRONOUSWITHFAULTTOLERANCE. Sends data to the PowerExchange Listener without waiting for a response. This option also enables error detection. This option combines the speed of CONFIRMWRITEOFF and the data integrity of CONFIRMWRITEON. Default is CONFIRMWRITEON.

Snowflake Connection Properties

When you set up a Snowflake connection, you must configure the connection properties.

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

The following table describes the Snowflake connection properties:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. 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. The ID 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.

Property	Description
Description	Optional. The description of the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Snowflake.
Username	The user name to connect to the Snowflake account.
Password	The password to connect to the Snowflake account.
Account	The name of the Snowflake account.
Warehouse	The Snowflake warehouse name.
Role	The Snowflake role assigned to the user.
Additional JDBC URL Parameters	<p>Enter one or more JDBC connection parameters in the following format:</p> <pre><param1>=<value>&<param2>=<value>&<param3>=<value>...</pre> <p>For example:</p> <pre>user=jon&warehouse=mywh&db=mydb&schema=public</pre> <p>To access Snowflake through Okta SSO authentication, enter the web-based IdP implementing SAML 2.0 protocol in the following format:</p> <pre>authenticator=https://<Your_Okta_Account_Name>.okta.com</pre> <p>Note: Microsoft ADFS is not supported.</p> <p>For more information about configuring Okta authentication, see the following website: https://docs.snowflake.net/manuals/user-guide/admin-security-fed-auth-configure-snowflake.html#configuring-snowflake-to-use-federated-authentication</p>

Teradata Parallel Transporter Connection Properties

Use a Teradata PT connection to access Teradata tables. The Teradata PT connection is a database type connection. You can create and manage a Teradata PT 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 Teradata PT connection properties:

Property	Description
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.

Property	Description
Description	Description of the connection. The description cannot exceed 765 characters.
Location	Domain where you want to create the connection.
Type	Connection type. Select Teradata PT.
User Name	Teradata database user name with the appropriate read and write permissions to access the database.
Password	Password for the Teradata database user name.
Driver Name	Name of the Teradata JDBC driver.
Connection String	Connection string used to access metadata from the database. Use the following connection string: <code>jdbc:teradata://<hostname>/database=<database name>,tmode=ANSI,charset=UTF8</code>

The following table describes the properties for data access:

Property	Description
TDPID	Name or IP address of the Teradata database machine.
Database Name	Teradata database name. If you do not enter a database name, Teradata PT API uses the default login database name.
Data Code Page	Code page associated with the database. When you run a mapping that writes data to a Teradata target, the code page of the Teradata PT connection must be the same as the code page of the Teradata target. Default is UTF-8.
Tenacity	Number of hours that Teradata PT API continues trying to log on when the maximum number of operations run on the Teradata database. Must be a positive, non-zero integer. Default is 4.
Max Sessions	Maximum number of sessions that Teradata PT API establishes with the Teradata database. Must be a positive, non-zero integer. Default is 4.
Min Sessions	Minimum number of Teradata PT API sessions required for the Teradata PT API job to continue. Must be a positive integer between 1 and the Max Sessions value. Default is 1.
Sleep	Number of minutes that Teradata PT API pauses before it retries to log on when the maximum number of operations run on the Teradata database. Must be a positive, non-zero integer. Default is 6.
Use Metadata JDBC URL for TDCH	Indicates that the Teradata Connector for Hadoop (TDCH) must use the JDBC URL that you specified in the connection string under the metadata access properties. Default is selected. Clear this option to enter a different JDBC URL that TDCH must use when it runs the mapping.

Property	Description
TDCH JDBC Url	<p>Enter the JDBC URL that TDCH must use when it runs a Teradata mapping. Use the following format:</p> <pre>jdbc:teradata://<hostname>/database=<database name>,tmode=ANSI,charset=UTF8</pre> <p>This field is available only when you clear the Use Metadata JDBC URL for TDCH option.</p>
Data Encryption	<p>Enables full security encryption of SQL requests, responses, and data on Windows. Default is disabled.</p>
Additional Sqoop Arguments	<p>This property is applicable if you use a Hortonworks or Cloudera cluster, and run a Teradata mapping on the Blaze or Spark engine through Sqoop.</p> <p>Enter the arguments that Sqoop must use to process the data. For example, enter <code>--method split.by.amp</code>. Separate multiple arguments with a space.</p> <p>See the Hortonworks for Teradata Connector and Cloudera Connector Powered by Teradata documentation for a list of arguments that you can specify.</p> <p>Note: If you use Hortonworks for Teradata Connector, the <code>--split-by</code> argument is required if you add two or more source tables in the read operation. If you use Cloudera Connector Powered by Teradata, the <code>--split-by</code> argument is required in the source connection if the source table does not have a primary key defined.</p>
Authentication Type	<p>Method to authenticate the user.</p> <p>Select one of the following authentication types:</p> <ul style="list-style-type: none"> - Native. Authenticates your user name and password against the Teradata database specified in the connection. - LDAP. Authenticates user credentials against the external LDAP directory service. <p>Default is Native.</p>

Tableau Connection Properties

Use a Tableau connection to connect to Tableau. When you create a Tableau connection, you enter information to access Tableau.

The following table describes the Tableau connection properties:

Property	Description
Name	Name of the Tableau connection.
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	Description of the connection. The description cannot exceed 765 characters.
Location	The Informatica domain where you want to create the connection.
Type	Type of connection. Select Tableau.

The following table describes the properties to connect to Tableau:

Connection Property	Description
Tableau Product	The name of the Tableau product to which you want to connect. You can choose one of the following Tableau products to publish the TDE or TWBX file: <ul style="list-style-type: none"> - Tableau Desktop. Creates a TDE file in the Data Integration Service machine. You can then manually import the TDE file to Tableau Desktop. - Tableau Server. Publishes the generated TDE or TWBX file to Tableau Server. - Tableau Online. Publishes the generated TDE or TWBX file to Tableau Online.
Connection URL	URL of Tableau Server or Tableau Online to which you want to publish the TDE or TWBX file. The URL has the following format: <code>http://<Host name of Tableau Server or Tableau Online>:<port></code>
User Name	User name of the Tableau Server or Tableau Online account.
Password	Password for the Tableau Server or Tableau Online account.
Content URL	The name of the site on Tableau Server or Tableau Online where you want to publish the TDE or TWBX file. Contact the Tableau administrator to provide the site name.

Tableau V3 Connection Properties

When you set up a Tableau V3 connection, you must configure the connection properties.

The following table describes the Tableau V3 connection properties:

Property	Description
Name	Name of the Tableau V3 connection.
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	Description of the connection. The description cannot exceed 765 characters.
Location	The Informatica domain where you want to create the connection.
Type	Type of connection. Select Tableau V3.

The following table describes the properties to connect to Tableau:

Connection Property	Description
Tableau Product	<p>The name of the Tableau product to which you want to connect.</p> <p>You can choose one of the following Tableau products to publish the <code>.hyper</code> or TWBX file:</p> <p>Tableau Desktop</p> <p>Creates a <code>.hyper</code> file in the Data Integration Service machine. You can then manually import the <code>.hyper</code> file to Tableau Desktop.</p> <p>Tableau Server</p> <p>Publishes the generated <code>.hyper</code> or TWBX file to Tableau Server.</p> <p>Tableau Online</p> <p>Publishes the generated <code>.hyper</code> or TWBX file to Tableau Online.</p>
Connection URL	<p>The URL of Tableau Server or Tableau Online to which you want to publish the <code>.hyper</code> or TWBX file.</p> <p>Enter the URL in the following format: <code>http://<Host name of Tableau Server or Tableau Online>:<port></code></p>
User Name	The user name of the Tableau Server or Tableau Online account.
Password	The password for the Tableau Server or Tableau Online account.
Site ID	<p>The ID of the site on Tableau Server or Tableau Online where you want to publish the or TWBX file.</p> <p>Note: Contact the Tableau administrator to provide the site ID.</p>
Schema File Path	<p>The path to a sample <code>.hyper</code> file from where the Data Integration Service imports the Tableau metadata.</p> <p>Enter one of the following options for the schema file path:</p> <ul style="list-style-type: none"> - Absolute path to the <code>.hyper</code> file. - Directory path for the <code>.hyper</code> files. - Empty directory path. <p>The path you specify for the schema file becomes the default path for the target <code>.hyper</code> file. If you do not specify a file path, the Data Integration Service uses the following default file path for the target <code>.hyper</code> file:</p> <pre><Data Integration Service installation directory>/apps/ Data_Integration_Server/<latest version>/bin/rtdm</pre>

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.

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

Property	Description
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.
Location	The domain where you want to create the connection.
Type	The connection type. Select Twitter Streaming.

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

Property	Description
Hose Type	Streaming API methods. You can specify one of the following methods: <ul style="list-style-type: none"> - Filter. The Twitter <code>statuses/filter</code> method returns public statuses that match the search criteria. - Sample. The Twitter <code>statuses/sample</code> 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.
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.
Do you have OAuth details?	Indicates whether you want to configure OAuth. Select one of the following values: <ul style="list-style-type: none"> - Yes. Indicates that you have the access token and secret. - No. Launches the OAuth Utility.
Access Token	Access token that the OAuth Utility returns. Twitter uses the 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.

VSAM Connection Properties

Use a VSAM connection to access VSAM data tables. The VSAM connection is a flat file connection type. You create a VSAM connection in the Developer tool. You can manage a VSAM 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 VSAM connection properties:

Option	Description
Location	Node name for the location of the PowerExchange Listener that connects to the VSAM data set. The node name is defined in the first parameter of the NODE statement in the PowerExchange dbmover.cfg configuration file.
User name	A user name that has the authority to connect to the VSAM data set.
Password	<p>A password for the specified user or a valid PowerExchange passphrase.</p> <p>A PowerExchange passphrase can be from 9 to 128 characters in length and can contain the following characters:</p> <ul style="list-style-type: none"> - Uppercase and lowercase letters - The numbers 0 to 9 - Spaces - The following special characters: ' - ; # \ , . / ! % & * () _ + { } : @ < > ? <p>Note: The first character is an apostrophe.</p> <p>Passphrases cannot include single quotation marks ('), double quotation marks ("), or currency symbols.</p> <p>To use passphrases, ensure that the PowerExchange Listener runs with a security setting of SECURITY=(1,N) or higher in the DBMOVER member. For more information, see "SECURITY Statement" in the <i>PowerExchange Reference Manual</i>.</p> <p>The allowable characters in the IBM IRRPHREX exit do not affect the allowable characters in PowerExchange passphrases.</p> <p>Note: A valid RACF passphrase can be up to 100 characters in length. PowerExchange truncates passphrases longer than 100 characters when passing them to RACF for validation.</p>
Code page	Required. Name of the code page to use for reading from or writing to the VSAM data set. Usually, this value is an ISO code page name, such as ISO-8859-6.
Pass-through security enabled	Enables pass-through security for the connection.
Encryption type	<p>Optional. The type of encryption that the Data Integration Service uses. Select one of the following options:</p> <ul style="list-style-type: none"> - None - AES <p>Default is None.</p> <p>Note: Informatica recommends that you use Secure Sockets Layer (SSL) authentication instead of configuring the Encryption Type and Level connection properties. SSL authentication provides stricter security and is used by several Informatica products. For more information about implementing SSL authentication in a PowerExchange network, see the <i>PowerExchange Reference Manual</i>.</p>
[Encryption] Level	<p>If you select AES for Encryption Type, select one of the following options to indicate the encryption level that the Data Integration Service uses:</p> <ul style="list-style-type: none"> - 1. Use a 128-bit encryption key. - 2. Use a 192-bit encryption key. - 3. Use a 256-bit encryption key. <p>If you do not select AES for Encryption Type, this option is ignored.</p> <p>Default is 1.</p>

Option	Description
Pacing size	Optional. Amount of data that the source system can pass to the PowerExchange Listener. Set the pacing size if an external application, database, or the Data Integration Service node is a bottleneck. User lower values for faster performance. Minimum value and default value is 0. A value of 0 provides the best performance.
Interpret as rows	Optional. Select this option to express the pacing size as a number of rows. Clear this option to express the pacing size in kilobytes. By default, this option is not selected and the pacing size is in kilobytes.
Compression	Optional. Select this option to enable source data compression. By compressing data, you can decrease the amount of data that Informatica applications send over the network. By default, this option is not selected and compression is disabled.
Offload processing	Optional. Controls whether to offload some bulk data processing from the source machine to the Data Integration Service machine. Select one of the following options: <ul style="list-style-type: none"> - AUTO. The Data Integration Service determines whether to use offload processing. - Yes. Use offload processing. - No. Do not use offload processing. Default is AUTO.
Worker threads	Optional. Number of threads that the Data Integration Service uses to process bulk data when offload processing is enabled. For optimal performance, this value should not exceed the number of available processors on the Data Integration Service machine. Valid values are 1 through 64. Default is 0, which disables multithreading.
Array size	Optional. The number of records in the storage array for the worker threads. This option is applicable when you set the Worker Threads option to a value greater than 0. Valid values are 25 to 5000. Default is 25.
Write mode	Optional. Mode in which Data Integration Service sends data to the PowerExchange Listener. Select one of the following write modes: <ul style="list-style-type: none"> - CONFIRMWRITEON. Sends data to the PowerExchange Listener and waits for a response before sending more data. Select this option when error recovery is a priority. However, this option might degrade performance. - CONFIRMWRITEOFF. Sends data to the PowerExchange Listener without waiting for a response. Use this option if you can reload the target table when an error occurs. - ASYNCHRONOUSWITHFAULTTOLERANCE. Sends data to the PowerExchange Listener without waiting for a response. This option also enables error detection. This option combines the speed of CONFIRMWRITEOFF and the data integrity of CONFIRMWRITEON. Default is CONFIRMWRITEON.

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:

Property	Description
Username	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.
Password	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.
End Point URL	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.
Timeout	Number of seconds that the Data Integration Service waits for a response from the web service provider before it closes the connection. Specify a timeout value between 1 and 10,000 seconds.
HTTP Authentication Type	Type of user authentication over HTTP. Select one of the following values: <ul style="list-style-type: none">- 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.

Property	Description
WS Security Type	Type of WS-Security that you want to use. Select one of the following values: <ul style="list-style-type: none"> - 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: <ul style="list-style-type: none"> - 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.

Identifier Properties in Database Connections

When you create most relational database connections, you must configure database identifier properties. The identifier properties determine whether the Data Integration Service encloses identifiers within delimited characters when the service generates SQL queries to access the database.

A database identifier is a database object name. Tables, views, columns, indexes, triggers, procedures, constraints, and rules can have identifiers. You use the identifier to reference the object in SQL queries. A database can have regular identifiers or delimited identifiers that must be enclosed within delimited characters.

Regular Identifiers

Regular identifiers comply with the format rules for identifiers. Regular identifiers do not require delimited characters when they are used in SQL queries.

For example, the following SQL statement uses the regular identifiers *MYTABLE* and *MYCOLUMN*:

```
SELECT * FROM MYTABLE
WHERE MYCOLUMN = 10
```

Delimited Identifiers

Delimited identifiers must be enclosed within delimited characters because they do not comply with the format rules for identifiers.

Databases can use the following types of delimited identifiers:

Identifiers that use reserved keywords

If an identifier uses a reserved keyword, you must enclose the identifier within delimited characters in an SQL query. For example, the following SQL statement accesses a table named *ORDER*:

```
SELECT * FROM "ORDER"
WHERE MYCOLUMN = 10
```

Identifiers that use special characters

If an identifier uses special characters, you must enclose the identifier within delimited characters in an SQL query. For example, the following SQL statement accesses a table named *MYTABLE\$@*:

```
SELECT * FROM "MYTABLE$@"
WHERE MYCOLUMN = 10
```

Case-sensitive identifiers

By default, identifiers in IBM DB2, Microsoft SQL Server, and Oracle databases are not case sensitive. Database object names are stored in uppercase, but SQL queries can use any case to refer to them. For example, the following SQL statements access the table named *MYTABLE*:

```
SELECT * FROM mytable
SELECT * FROM MyTable
SELECT * FROM MYTABLE
```

To use case-sensitive identifiers, you must enclose the identifier within delimited characters in an SQL query. For example, the following SQL statement accesses a table named *MyTable*:

```
SELECT * FROM "MyTable"
WHERE MYCOLUMN = 10
```

Identifier Properties

When you create most database connections, you must configure database identifier properties. The identifier properties that you configure depend on whether the database uses regular identifiers, uses keywords or special characters in identifiers, or uses case-sensitive identifiers.

Configure the following identifier properties in a database connection:

SQL Identifier Character

Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type.

Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers.

Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character.

Support Mixed-case Identifiers

Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the **SQL Identifier Character** property.

In the Informatica client tools, you must refer to the identifiers with the correct case. For example, when you create the database connection, you must enter the database user name with the correct case.

When the **SQL Identifier Character** property is set to none, the **Support Mixed-case Identifiers** property is disabled.

Example: Database Uses Regular Identifiers

In this example, the database uses regular identifiers. No identifiers contain a reserved keyword or a special character. The database uses identifiers that are not case sensitive.

In the database connection, set the **SQL Identifier Character** property to (None). When **SQL Identifier Character** is set to none, the **Support Mixed-case Identifiers** property is disabled.

When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers.

Example: Database Uses Keywords or Special Characters in Identifiers

In this example, the database uses keywords or special characters in some identifiers. The database uses identifiers that are not case sensitive.

In the database connection, configure the identifier properties as follows:

1. Set the **SQL Identifier Character** property to the character that the database uses for delimited identifiers.

This example sets the property to `"` (*quotes*).

2. Clear the **Support Mixed-case Identifiers** property.

When the Data Integration Service generates SQL queries, the service places the selected character around identifiers that use a reserved keyword or that use a special character. For example, the Data Integration Service generates the following query:

```
SELECT * FROM "MYTABLES@" /* identifier with special characters enclosed within
delimited
character */
WHERE MYCOLUMN = 10 /* regular identifier not enclosed within delimited character */
```

Example: Database Uses Case-Sensitive Identifiers

In this example, the database uses case-sensitive identifiers. The database might use keywords or special characters in some identifiers, or it might not.

In the database connection, configure the identifier properties as follows:

1. Set the **SQL Identifier Character** property to the character that the database uses for delimited identifiers.

This example sets the property to `"` (*quotes*).

2. Select the **Support Mixed-case Identifiers** property.

When the Data Integration Service generates SQL queries, the service places the selected character around all identifiers. For example, the Data Integration Service generates the following query:

```
SELECT * FROM "MyTable"      /* case-sensitive identifier enclosed within delimited
character */
WHERE "MYCOLUMN" = 10      /* regular identifier enclosed within delimited character */
```

INDEX

A

- Adabas connections
 - properties [256](#)
- Amazon Redshift connection
 - properties [258](#)
- Amazon S3 connection
 - properties [260](#)
- application archive file [150](#)
- application patch
 - add [177](#)
 - affected [177](#)
 - affected object [177](#)
 - affected objects
 - find [179](#)
 - create [182](#)
 - delete [178](#)
 - dependency
 - direct [163](#)
 - indirect [163](#)
 - remote [163](#)
 - deploy [180, 182](#)
 - deployment strategy [180](#)
 - design-time [163](#)
 - direct and indirect dependencies [173](#)
 - direct dependencies [172](#)
 - direct, indirect, and remote dependencies [174](#)
 - first-degree related objects [170](#)
 - guidelines [184](#)
 - impact [176, 180](#)
 - inherit [163, 170](#)
 - modify [177](#)
 - numbers [180](#)
 - object impact [176, 177, 180](#)
 - related objects [170](#)
 - rules [184](#)
 - run-time [163](#)
 - scope [180](#)
 - selected objects [170](#)
 - type [170](#)
- applications
 - connecting to a run-time application [159](#)
 - creating [155](#)
 - creating an application [145](#)
 - creation [145](#)
 - deploying to a Data Integration Service [148](#)
 - deploying to a file [150](#)
 - exporting to a file [150](#)
 - full [145](#)
 - how to create, deploy, and update [154](#)
 - incremental [146, 162](#)
 - mapping deployment properties [146](#)
 - overview [144](#)
 - properties [146](#)
 - redeploying [160](#)
 - replacing [152](#)

- applications (*continued*)
 - updating [152](#)
 - validating an application [145](#)
 - viewing run-time objects [152](#)
- array
 - complex data type [220](#)
 - dimension [220](#)
 - example [220](#)
 - format [220](#)
 - index [220](#)
 - multi-dimensional [220](#)
 - one-dimensional [220](#)
- attributes
 - relationships [124](#)
- Avro data types
 - transformation data types [224](#)

B

- bigint
 - constants in expressions [214](#)
 - high precision handling [213](#)
 - using in calculations [213](#)
 - writing to flat files [215](#)
- binary datatypes
 - overview [215](#)
- Blaze engine
 - connection properties [276](#)
- blockchain
 - connection properties [262](#)
- business terms
 - customizing hotkeys [46](#)
 - looking up [45](#)

C

- Cassandra connections
 - properties [264](#)
- certificates
 - adding untrusted certificates [89](#)
 - certificate properties [89](#)
 - managing certificates [89](#)
 - untrusted certificates [89](#)
- cheat sheets
 - description [21](#)
- check in/check out objects [40](#)
- checking objects out and in [40](#)
- CI/CD
 - assign objects [198, 202](#)
 - conflict detection [199](#)
 - conflict resolution [199](#)
 - continuous delivery [199](#)
 - continuous delivery example [202](#)
 - continuous integration [198](#)

- CI/CD (*continued*)
 - continuous integration example [200](#)
 - deploy application [197](#)
 - deploy objects [198](#), [200–202](#)
 - developer collaboration [199](#)
 - infacmd [195](#), [197](#)
 - object deployment [198](#)
 - overview [194](#)
 - redeploy application [197](#)
 - REST API [195](#), [197](#)
 - team-based development [198](#)
 - test objects [201](#)
- complex data types
 - array [220](#)
 - map [222](#)
 - overview [220](#)
 - struct [222](#)
- complex file formats
 - overview [223](#)
- configurations
 - troubleshooting [139](#)
- Confluent Kafka connection
 - Confluent Kafka broker properties [266](#)
 - create using infacmd [267](#)
 - general properties [266](#)
- Connection
 - details [313](#)
 - properties [313](#)
- connection environment SQL
 - configuring [57](#), [58](#)
- connection properties
 - Databricks [267](#)
 - blockchain [262](#)
- connection switching
 - configuring [55](#)
 - data types mapping [56](#)
 - description [54](#)
 - lookup behavior [56](#)
 - postrequisites [55](#)
 - prerequisites [54](#)
- connections
 - properties [276](#)
 - Connection Explorer view [50](#)
 - creating [51](#)
 - database identifier properties [346](#)
 - deleting [48](#)
 - editing [48](#)
 - Google PubSub [275](#)
 - overview [48](#)
 - renaming [48](#)
 - Salesforce Marketing Cloud [330](#)
 - switching [48](#)
 - web services properties [345](#)
- copy
 - description [29](#)
 - objects [30](#)
- Cosmos DB connection
 - creating [314](#)
- creating
 - Cosmos DB connection [314](#)
- custom queries
 - Informatica join syntax [79](#)
 - left outer join syntax [81](#)
 - normal join syntax [80](#)
 - outer join support [79](#)
 - right outer join syntax [83](#)
- custom SQL queries
 - creating [84](#)

- custom SQL queries (*continued*)
 - customized data objects [72](#)
- customized data objects
 - adding pre- and post-mapping SQL commands [84](#)
 - adding relational data objects [68](#)
 - adding relational resources [67](#)
 - advanced query [73](#)
 - creating [67](#)
 - creating a custom query [84](#)
 - creating key relationships [69](#)
 - creating keys [68](#)
 - custom SQL queries [72](#)
 - default query [73](#)
 - description [64](#)
 - entering source filters [76](#)
 - entering user-defined joins [78](#)
 - key relationships [65](#)
 - pre- and post-mapping SQL commands [83](#)
 - reserved words file [73](#)
 - select distinct [76](#)
 - simple query [73](#)
 - sorted ports [77](#)
 - troubleshooting [93](#)
 - user-defined joins [78](#)
 - using select distinct [76](#)
 - using sorted ports [77](#)
 - write properties [66](#)

D

- Data Integration Service
 - selecting [23](#)
- data type reference
 - flat files [229](#)
- data types
 - complex files [223](#)
 - decimal [217](#)
 - double [217](#)
 - IBM DB2 [230](#)
 - Microsoft SQL Server [234](#)
 - ODBC [239](#)
 - Oracle [240](#)
 - overview [211](#)
 - transformation [212](#)
- data viewer
 - configuration properties [131](#)
 - configurations [131](#), [135](#)
 - creating configurations [136](#)
 - troubleshooting configurations [139](#)
- database connections
 - identifier properties [346](#)
- database hints
 - entering in Developer tool [75](#)
- databases
 - environment SQL [57](#)
- Databricks connection properties [267](#)
- datatypes
 - DB2 for i5/OS [232](#)
 - Bigint [213](#)
 - binary [215](#)
 - Date/Time [215](#)
 - DB2 for z/OS [232](#)
 - implicit conversion [247](#)
 - Integer [213](#)
 - JDBC [233](#)
 - nonrelational [237](#)
 - port-to-port data conversion [247](#)

- datatypes (*continued*)
 - SAP HANA [243](#)
 - string [219](#)
 - XML [244](#)
- Date/Time datatypes
 - overview [215](#)
- DDL Generation
 - DDL Generation Errors [72](#)
 - Generate and Execute DDL [69](#)
- decimal
 - high precision handling [213](#), [217](#)
 - precision loss [218](#)
- Decimal
 - data type conversion [247](#)
- decimal data types
 - overview [217](#)
- default SQL query
 - viewing [84](#)
- delete [178](#)
- delimited identifiers
 - database connections [347](#)
- delimited properties
 - flat file data objects [98](#)
- deployment
 - mapping properties [146](#)
 - overview [144](#)
 - replacing applications [152](#)
 - to a Data Integration Service [157](#), [158](#)
 - to an application archive file [157](#), [158](#)
 - updating applications [152](#)
- Developer tool
 - workspace directory [18](#)
- domain
 - search [43](#)
 - searching for objects and properties [44](#)
- domains
 - adding [23](#)
 - description [24](#)
- double
 - high precision handling [217](#)
- double data types
 - overview [217](#)

E

- editing
 - application objects [160](#)
 - applications [160](#)
- environment SQL
 - configuring [57](#)
 - guidelines for entering [58](#)
- error messages
 - grouping [142](#)
 - limiting [143](#)
- example
 - array [220](#)
 - map [222](#)
 - struct [222](#)
- Excel
 - configuring flat files [108](#)
 - copying to a flat file [108](#)
 - editing a flat file data object [108](#)
- export
 - dependent objects [205](#)
 - objects [206](#)
 - overview [204](#)
 - to an application archive file [157](#), [158](#)

- export (*continued*)
 - XML file [206](#)

F

- filtering object dependencies
 - object dependencies [141](#)
- fixed-width properties
 - flat file data objects [99](#)
- flat file data objects
 - configuring metadata in Excel [108](#)
 - copying metadata [108](#)
 - advanced properties [96](#)
 - column properties [95](#)
 - creating as empty [109](#)
 - creating from control file [112](#)
 - creating from existing flat file [110](#)
 - delimited properties [98](#)
 - description [94](#)
 - editing in Excel [108](#)
 - fixed-width properties [99](#)
 - format properties [97](#)
 - general properties [95](#)
 - run-time read properties [100](#)
 - run-time write properties [102](#)
- folders
 - creating [29](#)
 - description [28](#)
- format properties
 - flat file data objects [97](#)

G

- Google Analytics connections
 - properties [271](#)
- Google BigQuery connection
 - properties [271](#)
- Google BigQuery connections
 - connection modes [273](#)
- Google Cloud Spanner connection
 - properties [273](#)
- Google Cloud Storage connections
 - properties [274](#)
- Google PubSub
 - connection properties [275](#)
- Greenplum connections
 - properties [269](#)

H

- HBase connections
 - MapR-DB properties [285](#)
 - properties [282](#)
- HDFS connections
 - properties [283](#)
- high precision
 - Bigint data type [213](#)
 - Decimal data type [213](#)
- hints
 - Query view [75](#)
- Hive connections
 - properties [285](#)
- Hive pushdown
 - connection properties [276](#)

how to
create, deploy, and update an application [154](#)
HTTP connections
properties [289](#)

I

IBM DB2 connections
properties [291](#)
IBM DB2 for i5/OS connections
properties [294](#)
IBM DB2 for z/OS connections
properties [297](#)
identifiers
delimited [347](#)
regular [347](#)
identifying relationships
description [124](#)
import
application archives [159](#)
dependent objects [205](#)
objects [209](#)
overview [204](#)
XML file [206](#)
IMS connections
properties [300](#)
incremental application
guidelines [184](#)
patch [162](#)
rules [184](#)
update [182](#)
incremental deployment
Incremental Deployment wizard [166](#)
patch deployment strategy [170](#)
patch deployment strategy properties [181](#)
select applications [167](#)
source object selection [168](#)
Informatica Data Services
overview [17](#)
Informatica Developer
overview [15](#)
searches [43](#)
setting up [22](#)
starting [18](#)
Informatica Marketplace
description [22](#)
integers
constants in expressions [214](#)
using in calculations [213](#)
writing to flat files [215](#)
intermediate calculation
in decimal [218](#)

J

JD Edwards EnterpriseOne connection
properties [307](#)
JDBC connections
properties [302](#)
JDBC V2 connection
properties [305](#)
join syntax
customized data objects [79](#)
Informatica syntax [79](#)
left outer join syntax [81](#)
normal join syntax [80](#)

join syntax (*continued*)
right outer join syntax [83](#)
JSON data types
transformation data types [226](#)

K

Kafka connection
create using infacmd [311](#)
general properties [309](#)
Kafka broker properties [309](#)
key relationships
creating between relational data objects [63](#)
creating in customized data objects [69](#)
customized data objects [65](#)
relational data objects [62](#)
Kudu connection
properties [311](#)

L

LDAP connection
properties [312](#)
local workspace directory
configuring [18](#)
logical data object mappings
creating [128](#)
read mappings [128](#)
types [127](#)
write mappings [128](#)
logical data object models
creating [115](#)
description [115](#)
example [114](#)
importing [116](#)
logical data objects
attribute relationships [124](#)
creating [124](#)
description [123](#)
example [114](#)
properties [123](#)
logical view of data
developing [114](#)
overview [113](#)
logs
description [141](#)
look up
Business Glossary Desktop [45](#)

M

map
complex data type [222](#)
example [222](#)
format [222](#)
key-value pair [222](#)
mappings
configuration properties [131](#)
configurations [131](#), [137](#)
creating configurations [138](#)
deployment properties [146](#)
troubleshooting configurations [139](#)
Messaging connection
Confluent Kafka connection [265](#)
Kafka connection [308](#)

- Metadata Access Service
 - selecting [24](#)
- Microsoft Azure Data Lake Storage Gen1 connection
 - properties [315](#)
- Microsoft Azure Data Lake Storage Gen2 connection
 - properties [316](#)
- Microsoft Azure SQL Data Warehouse connection
 - properties [317](#)
- Model repository
 - adding [23](#)
 - checking objects out and in [40](#)
 - connecting [41](#)
 - description [32](#)
 - non-versioned [37](#)
 - objects [33](#)
 - searching for objects and properties [44](#)
 - team-based development [37](#)
 - versioned [37](#)
 - versioned objects [38](#)
- Model Repository Service
 - refreshing [42](#)
- monitoring
 - description [143](#)
- MS SQL Server connections
 - properties [319](#)

N

- NaN
 - described [215](#)
- Netezza connections
 - properties [323](#)
- non-identifying relationships
 - description [124](#)
- nonrelational data objects
 - description [85](#)
 - importing [85](#)
- nonrelational data operations
 - creating read, write, and lookup transformations [86](#)

O

- object dependencies
 - view object dependencies [140](#)
- Object Dependencies view
 - viewing data [140](#)
- objects
 - copying [30](#)
 - locked objects [36](#)
 - version history [41](#)
- OData connections
 - properties [324](#)
- ODBC connections
 - properties [325](#)
- Oracle connections
 - properties [326](#)
- ORC file data types
 - transformation data types [226](#)
- outer join support
 - customized data objects [79](#)

P

- Parquet data types
 - transformation data types [227](#)

- performance tuning
 - creating data viewer configurations [136](#)
 - creating mapping configurations [138](#)
 - data viewer configurations [135](#)
 - mapping configurations [137](#)
 - web service configurations [138](#)
- physical data objects
 - customized data objects [64](#)
 - description [59](#)
 - flat file data objects [94](#)
 - nonrelational data objects [85](#)
 - relational data objects [60](#)
 - synchronization [90](#)
 - troubleshooting [93](#)
- pre- and post-mapping SQL commands
 - adding to relational data objects [84](#)
 - customized data objects [83](#)
- precision loss
 - in low precision [218](#)
- primary keys
 - creating in customized data objects [68](#)
 - creating in relational data objects [63](#)
- project permissions
 - allowing parent object access [27](#)
 - assigning [28](#)
 - dependent object instances [27](#)
 - external object permissions [26](#)
 - grant permission [26](#)
 - read permission [26](#)
 - showing security details [27](#)
 - write permission [26](#)
- projects
 - assigning permissions [28](#)
 - creating [25](#)
 - description [24](#)
 - filtering [25](#)
 - permissions [26](#)
 - sharing [24](#)

Q

- QNaN
 - converting to 1.#QNaN [215](#)
- Query view
 - configuring hints [75](#)

R

- regular identifiers
 - database connections [347](#)
- relational connections
 - adding to customized data objects [67](#)
- relational data objects
 - adding to customized data objects [68](#)
 - creating key relationships [63](#)
 - creating keys [63](#)
 - description [60](#)
 - importing [62](#)
 - key relationships [62](#)
 - troubleshooting [93](#)
 - wild card search [62](#)
- reserved words file
 - creating [73](#)
 - customized data objects [73](#)
- resource parameters
 - deploying in a mapping [150](#)

- run-time application
 - searching for objects and properties [44](#)
- run-time read properties
 - flat file data objects [100](#)
- run-time write properties
 - flat file data objects [102](#)

S

- Salesforce Marketing Cloud
 - connection properties [330](#)
- SAP connections
 - properties [331](#)
- search
 - overview [43](#)
 - business glossary [45](#)
 - domain [43](#)
 - editor [46](#)
 - Model repository [43](#)
 - run-time application [43](#)
- select distinct
 - customized data objects [76](#)
 - using in customized data objects [76](#)
- self-joins
 - custom SQL queries [72](#)
- Sequential connections
 - properties [334](#)
- Snowflake connection
 - properties [336](#)
- sorted ports
 - customized data objects [77](#)
 - using in customized data objects [77](#)
- source filters
 - entering [76](#)
- Spark deploy mode
 - Hadoop connection properties [276](#)
- Spark engine
 - connection properties [276](#)
- Spark Event Log directory
 - Hadoop connection properties [276](#)
- Spark execution parameters
 - Hadoop connection properties [276](#)
- Spark HDFS staging directory
 - Hadoop connection properties [276](#)
- SQL
 - configuring environment SQL [57](#)
 - guidelines for entering environment SQL [58](#)
- SQL hints
 - entering in Developer tool [75](#)
- string datatypes
 - overview [219](#)
- struct
 - complex data type [222](#)
 - example [222](#)
 - format [222](#)
 - name-type pair [222](#)
 - schema [222](#)
- synchronization
 - customized data objects [90](#)
 - physical data objects [90](#)

T

- Tableau V3 connection
 - properties [340](#)

- team-based development
 - troubleshooting [41](#)
- Teradata Parallel Transporter connections
 - properties [337](#)
- transaction environment SQL
 - configuring [57](#), [58](#)
- transformation data types
 - list of [212](#)
- troubleshooting
 - versioning [41](#)
- Twitter Streaming connections
 - properties [341](#)

U

- update application
 - objects [182](#)
- user-defined joins
 - customized data objects [78](#)
 - entering [78](#)
 - Informatica syntax [79](#)
 - left outer join syntax [81](#)
 - normal join syntax [80](#)
 - outer join support [79](#)
 - right outer join syntax [83](#)

V

- validation
 - configuring preferences [142](#)
 - grouping error messages [142](#)
 - limiting error messages [143](#)
- versioning
 - troubleshooting [41](#)
- view object dependencies
 - filtering object dependencies [141](#)
 - object dependencies [140](#)
- views
 - Connection Explorer view [50](#)
 - description [19](#)
- VSAM connections
 - properties [342](#)

W

- web connections
 - properties [289](#)
- web service
 - configuration properties [131](#)
 - configurations [138](#)
 - creating configurations [138](#)
- Welcome page
 - description [21](#)
- workbench
 - description [19](#)
- workspace directory
 - configuring [18](#)
- WSDL data objects
 - advanced view [87](#)
 - creating [87](#)
 - importing [86](#)
 - overview view [87](#)
 - schema view [86](#)
 - synchronization [88](#)