Teiid Designer User Guide

1

8.0.0

1. Introduction	. 1
1.1. What is Teiid Designer?	1
1.2. Metadata Overview	. 2
1.2.1. What is Metadata	2
1.2.2. Business and Technical Metadata	4
1.2.3. Design-Time and Runtime Metadata	. 5
1.2.4. Source and View Metadata	6
1.3. It's all in the Modeling	10
1.3.1. What Are Models?	10
1.3.2. How is a Model Defined?	11
1.3.3. Guiding through the process	12
1.3.4. Model Classes and Types	12
1.3.5. The Virtual Database	13
1.3.6. Model Validation	14
1.3.7. Testing Your Models	14
1.3.8. Model Object Extensions	15
2. Dive Right In!	19
2.1. Guide Example	19
2.1.1. Model a JDBC Source	20
2.2. Cheat Sheet Example	34
2.2.1. Consume a SOAP Web Service	34
3. New Model Wizards	47
3.1. Creating New Relational Source Model	48
3.1.1. Generate File Translator Procedures	49
3.1.2. Generate Web Service Translator Procedures	50
3.1.3. Copy From Existing Model	52
3.2. Creating New Relational View Model	53
3.2.1. Copy From Existing Model	54
3.2.2. Transform From Existing Model	55
3.3. Creating XML Document View Model	55
3.3.1. Copy From Existing Model	55
3.3.2. Build XML Documents From XML Schema	56
3.4. Creating XML Schema Model	60
3.4.1. Copy From Existing Model	61
3.5. Creating Web Service View Model	61
3.5.1. Copy From Existing Model	62
3.5.2. Build From Existing WSDL File(s) or URL	63
3.5.3. Build From Relational Models	63
3.5.4. Build From XML Document View Models	64
4. Importers	67
4.1. Import DDL	67
4.2. Import From JDBC Database	70
4.3. Import From Flat File Source	76
4.4. Import From XML Data File Source	90

4.5. Import From Salesforce	96
4.6. Import Metadata From Text File	101
4.6.1. Import Relational Model (XML Format)	102
4.6.2. Import Relational Tables (CSV Format)	105
4.6.3. Import Relational View Tables (CSV Format)	107
4.6.4. Import Datatypes (CSV Format)	109
4.7. Import WSDL into Relational Models	111
4.7.1. Circular References in WSDL Schemas	117
4.8. Import WSDL Into Web Service	118
4.8.1. Import WSDL From Workspace Location	119
4.8.2. Import WSDL From File System Location	125
4.8.3. Import WSDL From URL	131
4.9. XSD Schema File	137
5. Creating and Editing Model Objects	143
5.1. Creating New Model Objects	143
5.1.1. New Child Action	143
5.1.2. New Sibling Action	146
5.1.3. New Association Action	148
5.2. Model Object Editors	150
5.2.1. Transformation Editor	152
5.2.2. Input Set Editor (XML)	170
5.2.3. Choice Editor (XML)	172
5.2.4. Recursion Editor (XML)	175
5.2.5. Operation Editor	179
5.3. Managing Model Object Extensions	180
5.3.1. Create New MED	181
5.3.2. Edit MED	181
5.3.3. Extending Models With MEDs	185
5.3.4. Setting Extended Property Values	186
6. Metadata-specific Modeling	189
6.1. Relational Source Modeling	189
6.1.1. Source Function	189
6.1.2. Create Relational Table Wizard	192
6.2. Relational View Modeling	198
6.2.1. Create Materialized Views	198
6.2.2. Create Relational View Table Wizard	202
6.3. XML Document Modeling	206
6.3.1. Create XML View Documents from schema	206
6.4. Web Services Modeling	210
6.4.1. Create Web Service Action	210
6.4.2. Web Services War Generation	212
7. Editing Models and Projects	225
7.1. Rename A Model	225
7.2. Move Model	226

7.3. Save Copy of Model	228
7.4. Clone Project	230
8. Managing VDBs	233
8.1. Creating a VDB	233
8.2. Editing a VDB	234
8.3. Test a VDB	234
8.4. Reusing VDBs	234
9. Testing Your Models	241
9.1. Manage Connection Profiles	241
9.1.1. Set Connection Profile for Source Model	241
9.1.2. View Connection Profile for Source Model	241
9.1.3. Remove Connection Profile from Source Model	242
9.2. Previewing Data For a Model	243
9.2.1. Preview Relational Table or View	244
9.2.2. Preview Relational Table With Access Pattern	245
9.2.3. Preview Relational Procedure	245
9.2.4. Preview Web Service Operation	246
9.2.5. Sample SQL Results for Preview Data	247
9.2.6. Execution Plans	248
9.3. Testing With Your VDB	249
9.3.1. Creating Data Sources	249
9.3.2. Execute VDB from Model Explorer	251
9.3.3. Deploy VDB from Model Explorer	252
9.3.4. Executing a Deployed VDB	252
10. Searching	259
10.1. Finding Model Objects	259
10.2. Search Transformation SQL	261
10.3. Search Models Via Metadata Properties	263
A. Supported Data Sources	265
B. User Preferences	267
B.1. Teiid Designer Preferences	267
B.1.1. Diagram Preferences	268
B.1.2. Diagram Printing Preferences	270
B.1.3. Editor Preferences	271
B.1.4. Validation Preferences	274
C. Teiid Designer Ui Reference	279
C.1. Teiid Designer Perspectives	279
C.1.1. Teiid Designer Perspective	279
C.1.2. Opening a Perspective	281
C.1.3. Further information	283
C.2. Teiid Designer Views	283
C.2.1. Model Explorer View	283
C.2.2. Outline View	286
C.2.3. Teiid View	288

C.2.4. Properties View	291
C.2.5. Description View	293
C.2.6. Problems View	295
C.2.7. Search Results View	297
C.2.8. Datatype Hierarchy View	299
C.2.9. Teiid Model Classes View	300
C.2.10. System Catalog View	301
C.2.11. SQL Reserved Words View	302
C.2.12. Model Extension Definition Registry View (MED Registry View)	303
C.2.13. Guides View	304
C.2.14. Status View	306
C.2.15. Cheat Sheets View	307
C.3. Editors	308
C.3.1. Model Editor	310
C.3.2. VDB Editor	329
C.3.3. Model Extension Definition Editor	335
C.4. Teiid Designer Main Menu	337
C.4.1. File Menu	338
C.4.2. Edit Menu	341
C.4.3. Refactor Menu	342
C.4.4. Navigate Menu	343
C.4.5. Search Menu	343
C.4.6. Project Menu	345
C.4.7. Metadata Menu	346
C.4.8. Run Menu	346
C.4.9. Window Menu	347
C.4.10. Help Menu	348
C.5. Teiid Server Management	349

Introduction

The Teiid Designer User's Guide provides detailed descriptions of Teiid Designer features and functionality.

1.1. What is Teiid Designer?

Teiid Designer is an Eclipse-based graphical modeling tool for modeling, analyzing, integrating and testing multiple data sources to produce Relational, XML and Web Service Views that expose your business data.

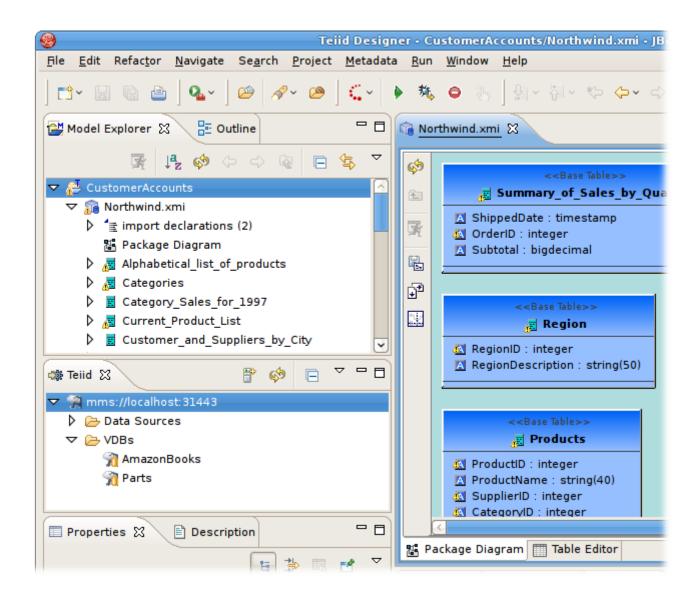


Figure 1.1. Teiid Designer

Why Use Teiid Designer?

Teiid Designer is a visual tool that enables rapid, model-driven definition, integration and testing of data services without programming. With Teiid Designer, not only do you map from data sources to target formats using a visual tool, but you can also:

- resolve semantic differences
- create virtual data structures at a physical or logical level
- use declarative interfaces to integrate, aggregate, and transform the data on its way from source to a target format which is compatible and optimized for consumption by your applications This allows you to abstract the structure of the information you expose to and use in your applications from the underlying physical data structures. With Teild Designer, data services are defined quickly, the resulting artifacts are easy to maintain and reuse, and all the valuable work and related metadata are saved for later reference.

You can use Teiid Designer to integrate multiple sources, and access them using the common data access standards:

- Web Services / SOAP / XML
- JDBC / SQL
- ODBC / SQL

Teiid Designer is an integral part of the Teiid Designer enterprise-class system for providing data services for service-oriented architectures.

1.2. Metadata Overview

1.2.1. What is Metadata

Metadata is data about data. A piece of metadata, called a meta object in the Teiid Designer, contains information about a specific information structure, irrespective of whatever individual data fields that may comprise that structure.

Let's use the example of a very basic database, an address book. Within your address book you certainly have a field or column for the ZIP code (or postal code number). Assuming that the address book services addresses within the United States, you can surmise the following about the column or field for the ZIP code:

- Named ZIPCode
- Numeric

- A string
- Nine characters long
- Located in the StreetAddress table
- Comprised of two parts: The first five digits represent the five ZIP code numbers, the final four represent the ZIP Plus Four digits if available, or 0000 if not
- Formatted only in integer numeric characters. Errors will result if formatted as 631410.00 or 6314q0000

This definition represents metadata about the ZIP code data in the address book database. It abstracts information from the database itself and becomes useful to describe the content of your enterprise information systems and to determine how a column in one enterprise information source relates to another, and how those two columns could be used together for a new purpose

You can think of this metadata in several contexts:

- What information does the metadata contain? (see Section 1.2.2, "Business and Technical Metadata")
- What data does the metadata represent? (see Section 1.2.4, "Source and View Metadata")
- How will my organization use and manage this metadata? (see Section 1.2.3, "Design-Time and Runtime Metadata")

Editing Metadata vs. Editing Data

The Teiid Designer helps you to create and describe an abstract graphic representation of your data structure of your data in the original data sources. It also describes whether those data sources are composed of Relational databases, text files, data streams, legacy database systems, or some other information type.

The Teiid Designer allows you to create, edit, and link these graphically-represented meta objects that are really a description of your data, and not the data itself.

So when this documentation describes the process of creating, deleting, or editing these meta objects, **remember** that you are not, in fact, modifying the underlying data.

Metadata Models

A **metadata model** represents a collection of metadata information that describes a complete structure of data.

In a previous example we described the field ZIPCode as a **metadata object** in an address book database. This **meta object** represents a single distinct bit of metadata information. We alluded to its parent table, StreetAddress. These **meta objects**, and others that would describe the other

tables and columns within the database, would all combine to form a **Source Metadata** model for whichever enterprise information system hosts all the objects.

You can have **Source Models** within your collection of **metadata models** These model physical data storage locations. You can also have **View Models**, which model the business view of the data. Each contains one type of metadata or another. For more information about difference between Source and View metadata, (see *Section 1.2.4, "Source and View Metadata"*).

NOTE: For detailed information about creating models from your metadata, see Section 1.3, "It's all in the Modeling..."

1.2.2. Business and Technical Metadata

Metadata can include different types of information about a piece of data.

- **Technical metadata** describes the information required to access the data, such as where the data resides or the structure of the data in its native environment.
- **Business metadata** details other information about the data, such as keywords related to the meta object or notes about the meta object.

Note that the terms **technical and business metadata**, refer to the content of the metadata, namely what type of information is contained in the metadata. Don't confuse these with the terms "physical" and "view" metadata that indicate what the metadata represents. For more information, (see Section 1.2.4, "Source and View Metadata").

Technical Metadata

Technical metadata represents information that describes how to access the data in its original native data storage. Technical metadata includes things such as datatype, the name of the data in the enterprise information system, and other information that describes the way the native enterprise information system identifies the meta object

Using our example of an address book database, the following represent the technical metadata we know about the ZIP code column:

- Named ZIPCode
- Nine characters long
- A string
- Located in the StreetAddress table
- Uses SQL Query Language

These bits of information describe the data and information required to access and process the data in the enterprise information system.

Business Metadata

Business metadata represents additional information about a piece of data, not necessarily related to its physical storage in the enterprise information system or data access requirements. It can also represent descriptions, business rules, and other additional information about a piece of data.

Continuing with our example of the ZIP Code column in the address book database, the following represents business metadata we may know about the ZIP code:

- The first five characters represent the five ZIP code numbers, the final four represent the ZIP Plus Four digits if available, or 0000 if not
- The application used to populate this field in the database strictly enforces the integrity of the data format

Although the first might seem technical, it does not directly relate to the physical storage of the data. It represents a business rule applied to the contents of the column, not the contents themselves.

The second, of course, represents some business information about the way the column was populated. This information, although useful to associate with our definition of the column, does not reflect the physical storage of the data.

1.2.3. Design-Time and Runtime Metadata

Teiid Designer software distinguishes between design-time metadata and run-time metadata. This distinction becomes important if you use the Teiid Designer Server. Design-time data is laden with details and representations that help the user understand and efficiently organize metadata. Much of that detail is unnecessary to the underlying system that runs the Virtual Database that you will create. Any information that is not absolutely necessary to running the Virtual Database is stripped out of the run-time metadata to ensure maximum system performance.

Design-Time Metadata

Design-time metadata refers to data within your local directory that you have created or have imported. You can model this metadata in the Teiid Designer, adding **Source** and **View** metadata.

Runtime Metadata

Once you have adequately modeled your enterprise information systems, including the necessary technical metadata that describes the physical structure of your sources, you can use the metadata for data access.

To prepare the metadata for use in the Teiid Designer Server, you take a snapshot of a metadata model for the Teiid Designer Server to use when resolving queries from your client applications. This run-time metadata represents a static version of design-time metadata you created or imported. This snapshot is in the form of a **Virtual Database** definition, or **VDB**.

As you create this **runtime metadata**, the Teiid Designer:

- · derives the runtime metadata from a consistent set of metadata models.
- creates a subset of design-time metadata, focusing on the technical metadata that describes the access to underlying enterprise information systems.
- optimizes runtime metadata for data access performance.

You can continue to work with the design-time metadata, but once you have created a runtime metadata model, it remains static.

1.2.4. Source and View Metadata

In addition to the distinction between business and technical metadata, you should know the difference between **Source Metadata and View Metadata**.

Source and View metadata refer to what the metadata represents, not its content.

Source Metadata directly represents metadata for an enterprise information system and captures exactly where and how the data is maintained. Source Metadata sounds similar to technical metadata, but Source Metadata can contain both technical and business metadata. When you model Source Metadata, you are modeling the data that your enterprise information systems contain.

View Metadata, on the other hand, represent tailored views that **transform** the **Source Metadata** into the terminology and domain of different applications. **View Metadata**, too, can contain both technical and business metadata. When you model **View Metadata**, you're modeling the data as your applications (and your enterprise) ultimately use it.

Modeling Your Source Metadata

When you model the **Source Metadata** within your enterprise information systems, you capture some detailed information, including:

- Identification of datatype
- Storage formats
- Constraints
- · Source-specific locations and names

The **Source Metadata** captures this detailed technical metadata to provide a map of the data, the location of the data, and how you access it.

This collection of **Source Metadata** comprises a direct mapping of the information sources within your enterprise. If you use the Teiid Designer Server for information integration, this technical metadata plays an integral part in query resolution.

For example, our ZIPCode column and its parent table StreetAddress map directly to fields within our hypothetical address book database.

To extend our example, we might have a second source of information, a comma-separated text file provided by a marketing research vendor. This text file can supply additional demographic information based upon address or ZIP code. This text file would represent another Enterprise Information System (EIS), and the meta objects in its Source Model would describe each comma-separated value.

Modeling Your View Metadata

When you create **View Metadata**, you are not describing the nature of your physical data storage. Instead, you describe the way your enterprise uses the information in its day-to-day operations.

View Metadata derives its classes and attributes from other metadata. You can derive View Metadata from Source Metadata that describes the ultimate sources for the metadata or even from other View Metadata. However, when you model View Metadata, you create special "views" on your existing enterprise information systems that you can tailor to your business use or application expectations. This View Metadata offers many benefits:

- You can expose only the information relevant to an application. The application uses this **View Metadata** to resolve its queries to the ultimate physical data storage.
- You can add content to existing applications that require different views of the data by adding the **View Metadata** to the existing **View Metadata** that application uses. You save time and effort since you do not have to create new models nor modify your existing applications.
- Your applications do not need to refer to specific physical enterprise information systems, offering flexibility and interchangeability. As you change sources for information, you do not have to change your end applications.
- The **View Metadata** models document the various ways your enterprise uses the information and the different terminology that refers to that information. They do so in a central location.

Our example enterprise information sources, the address book database, and the vendor-supplied comma-delimited text file, reside in two different native storage formats and therefore have two Source Metadata models. However, they can represent one business need: a pool of addresses for a mass mailing.

By creating a **View Metadata** model, we could accurately show that this single View Table, the AddressPool, contains information from the two enterprise information systems. The **View Metadata** model not only shows from where it gets the information, but also the SQL operations it performs to select its information from its source models.

This **View Metadata** can not only reflect and describe how your organization uses that information, but, if your enterprise uses the Teiid Designer Server, your applications can use the **View Metadata** to resolve queries.

To create this **View Metadata**, you create a view and define a **transformation** for that view, a special query that enables you to select information from the source (or even other view) metadata models. For more information, see "*Section 5.2.1, "Transformation Editor"*."

Metadata Transformations

By modeling View Metadata, you can illustrate the business view of your enterprise information sources. View Metadata models not only describe that business view, but also illustrate how the meta objects within the View Metadata models derive their information from other metadata models.

Let's return to the example of our address book database and the vendor's comma-separated list. We want to generate the View Metadata model, Address Pool, from these enterprise information systems.

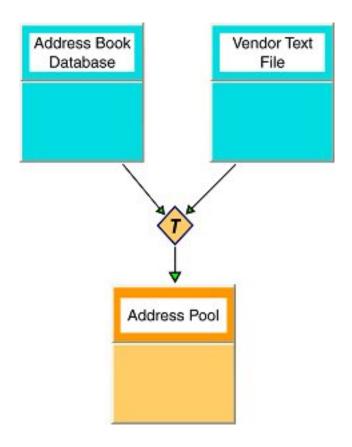


Figure 1.2. Data Flow for View Transformations

The transformation that joins these metadata models to create the virtual Address Pool metadata model contains a SQL query, called a union, that determines what information to draw from the source metadata and what to do with it.

The resulting Address Pool contains not only the address information from our Address Book database, but also that from our vendor-supplied text file.

SQL in Transformations

Transformations contain SQL queries that SELECT the appropriate attributes from the information sources.

For example, from the sources the transformation could select relevant address columns, including first name, last name, street address, city, state, and ZIP code. Although the metadata models could contain other columns and tables, such as phone number, fax number, e-mail address, and Web URL, the transformation acts as a filter and populates the Address Pool metadata model with only the data essential to building our Address Pool.

You can add other SQL logic to the transformation query to transform the data information. For example, the address book database uses a nine-character string that represents the ZIP Plus Four. The transformation could perform any SQL-supported logic upon the ZIPCode column to substring this information into the format we want for the Address Pool View metadata model.

Mapping XML Transformations

When you model View Metadata, you can also create a View XML Document model. This View Document lets you select information from within your other data sources, just like a regular View Metadata model, but you can also map the results to tags within an XML document.

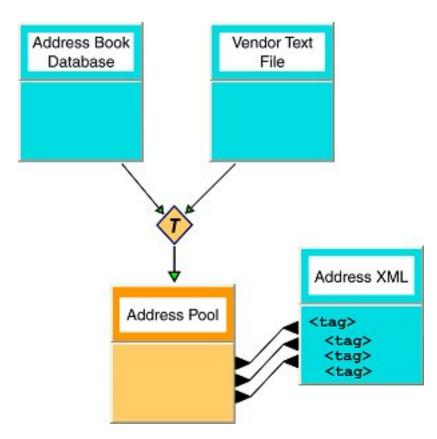


Figure 1.3. Data Flow for XML Transformations

In this example, the Address Pool View Metadata model still selects its information from the Address Book Database and the Vendor Text File, but it also maps the resulting columns into tags in the Address XML document.

1.3. It's all in the Modeling...

1.3.1. What Are Models?

A model is a representation of a set of information constructs. A familiar model is the relational model, which defines tables composed of columns and containing records of data. Another familiar model is the XML model, which defines hierarchical data sets.

In Teiid Designer, models are used to define the entities, and relationships between those entities, required to fully define the integration of information sets so that they may be accessed in a uniform manner, using a single API and access protocol. The file extension used for these models is .xmi (Example: NorthwindOracle.xmi) which adheres to the XMI syntax defined by the OMG.

Below is an example of the partial contents of a model file.

```
<?xml version="1.0" encoding="ASCII"?>
<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xm</pre>
  <mmcore:ModelAnnotation xmi:uuid="mmuuid:b0355f00-413b-1079-90</pre>
    <modelImports xmi:uuid="mmuuid:2d815780-4140-1079-9d18-8acf.</pre>
    <modelImports xmi:uuid="mmuuid:2e663940-4140-1079-9d18-8acf-
  </mmcore:ModelAnnotation>
  <relational:BaseTable xmi:uuid="mmuuid:b20e64c0-413b-1079-9d1;</pre>
    <columns xmi:uuid="mmuuid:bb5ac3c0-413b-1079-9d18-8acf4a712"
      <type href="http://www.w3.org/2001/XMLSchema#long"/>
    </columns>
    <columns xmi:uuid="mmuuid:bc4ee7c0-413b-1079-9d18-8acf4a712"
      <type href="http://www.w3.org/2001/XMLSchema#string"/>
    </columns>
    <columns xmi:uuid="mmuuid:bc4ee7c1-413b-1079-9d18-8acf4a712"
      <type href="http://www.w3.org/2001/XMLSchema#string"/>
    </columns>
    <columns xmi:uuid="mmuuid:bc4ee7c2-413b-1079-9d18-8acf4a712"
      <type href="http://www.metamatrix.com/metamodels/SimpleDay
    </columns>
    <primaryKey xmi:uuid="mmuuid:d481f940-413b-1079-9d18-8acf4a"</pre>
  </relational:BaseTable>
```

Figure 1.4. Sample Model File



Note

Model files should never be modified "by hand". While it is possible to do so, there is the possibility that you may corrupt the file such that it cannot be used within the Teiid Designer system.

The fundamental models in Teiid Designer define the structural and data characteristics of the information contained in data sources. These are referred to as source models (represented by

G

). Teiid Designer uses the information in source models to federate the information in multiple sources, so that from a user's viewpoint these all appear to be in a single source.

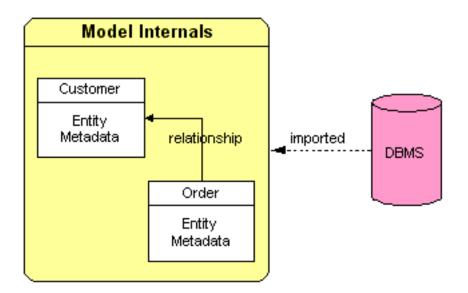


Figure 1.5. Model Internals

In	addition	to	source	models,		Teiid	Designer	provides	the
ability	to	define	а	variety	of	view	models	represented	by

). These can be used to define a layer of abstraction above the physical (or source) layer, so that information can be presented to end users and consuming applications in business terms rather than as it is physically stored. Views are mapped to sources using transformations between models. These business views can be in a variety of forms:

- Relational Tables and Views
- XML
- Web services
- Functions

For full list of supported model types see Chapter 3, New Model Wizards

A third model type, logical, provides the ability to define models from a logical or structural perspective.

1.3.2. How is a Model Defined?

Models are defined using Teiid Designer in various ways:

- Created via importing source data characteristics. (see Chapter 4, Importers)
- Manual creation via Chapter 3, New Model Wizards
- Transforming or copying from one model into another (see *Chapter 3, New Model Wizards* options)
- · Various custom actions

1.3.3. Guiding through the process

To make the process of using Teiid Designer to build models more as easy as posssible, a guides view (*Section C.2.13, "Guides View*') has been introduced. It provides action sets which bring together the actions necessary to develop models for specific use-cases. Action sets are available for the following scenerios:

- Consuming a SOAP Web Service
- Creating a REST WAR archive
- Creating a SOAP WAR archive
- Modelling from a Flat File Source (a text file)
- Modelling from a JDBC Data Source
- Modelling from a Local XML File Source
- Modelling from a Remote XML File Source
- · Connecting to a Teiid Server

1.3.4. Model Classes and Types

Teiid Designer can be used to model a variety of classes of models. Each of these represent a conceptually different classification of models.

- **Relational** Model data that can be represented in table columns and records form. Relational models can represent structures found in relational databases, spreadsheets, text files, or simple Web services.
- XML Model that represents the basic structures of XML documents. These can be "backed" by XML Schemas. XML models represent nested structures, including recursive hierarchies.
- XML Schema W3C standard for formally defining the structure and constraints of XML documents, as well as the datatypes defining permissible values in XML documents.
- Web Services which define Web service interfaces, operations, and operation input and output parameters (in the form of XML Schemas).

• **Function** - The Function metamodel supports the capability to provide user-defined functions, including binary source jars, to use in custom transformation SQL statements.

1.3.5. The Virtual Database

The critical artifact that **Teiid Designer** is intended to manage is the **VDB**, or **Virtual DataBase**. Through the Teiid server, VDB's behave like standard JDBC database schema which can be connected to, queried and updated based on how the VDB is configured. Since VDB's are just databases once they are deployed, they can be used as sources to other view model transformations. This allows creating and deploying re-usable or common VDB's in multiple layers depending on your business needs.

1.3.5.1. VDB Content and Structure

In Designer, the VDB file names use a "**.vdb**" file extension. VDBs are structurally just ZIP archive files containing 3 folders:

- META-INF
 - contains "vdb.xml" definition file
- runtime-inf
 - · contains a binary INDEX file for each model included in your VDB
- <project folder name>
 - contains of the models you will be adding in the VDB Editor (i.e. *.xmi and *.xsd files)

When deployed, the metadata is consumed by Teiid in order to create the necessary runtime metadata for your model definitions.

The vdb.xml file contains:

- VDB name, version, properties
- contained model information (name, translator name, connection info)
- · translator info
- · data role definitions for the referenced models
- import VDB references

The **vdb.xml** file example below highlights the basic model information. Note the VIRTUAL and PHYSICAL <model> elements containing property references to the INDEX files as well as the <source> element info for the PHYSICAL (aka source) model EU_CustomerAccounts.xmi.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<vdb version="1" name="Financials">
```

```
<model visible="true" type="VIRTUAL" name="US_CustomerAccounts" path="/</pre>
Financials/US_CustomerAccounts.xmi">
        <property value="4097408696" name="checksum"/>
        <property value="Relational" name="modelClass"/>
        <property value="false" name="builtIn"/>
        <property value="1592679058.INDEX" name="indexName"/>
        <property value="/Financials/US_CustomerAccounts.xmi"</pre>
name="imports"/>
    </model>
    <model visible="true" type="PHYSICAL" name="EU_CustomerAccounts" path="/
Financials/EU_CustomerAccounts.xmi">
        <property value="525566235" name="checksum"/></property value="525566235" name="checksum"/>
        <property value="Relational" name="modelClass"/>
        <property value="false" name="builtIn"/>
        <property value="1119071590.INDEX" name="indexName"/>
        <source translator-name="postgresql" connection-jndi-
name="EU_CustomerAccounts" name="EU_CustomerAccounts"/>
    </model>
</vdb>
```

Fortunately, Teiid Designer simplifies the management of your VDBs by providing a dedicated VDB Editor which maintains a consistent, valid vdb.xml file for you and assists in synchronizing your workspace models with any related models in your VDB. (See the *Section C.3.2, "VDB Editor"* section)

1.3.6. Model Validation

Models must be in a valid state in order to be used for data access. Validation of a single model means that it must be in a self-consistent and complete state, meaning that there are no "missing pieces" and no references to non-existent entities. Validation of multiple models checks that all inter-model dependencies are present and resolvable.

Models must always be validated when they are deployed in a VDB for data access purposes.

Teiid Designer will automatically validate your models whenever the user Saves (Note: the "Project > Build Automatically" menu option must be checked). When editing models, the editor tabs will display a "*" to indicate that the model has unsaved changes.

1.3.7. Testing Your Models

Designing and working with data is often much easier when you can see the information you're working with. The Teiid Designer's **Preview Data** feature makes this possible and allows you to instantly preview the information described by any object, whether it's a physical table or a virtual view. In other words, you can test the views with actual data by simply selecting the table, view, procedure or XML document. The preview functionality insures that data access behavior in the Teiid Designer will reliably match when the VDB is deployed to the Server. (See: Section C.5, "Teiid Server Management" for details)

Previewing information is a fast and easy way to sample the data. Of course, to run more complicated queries like what your application likely uses, simply execute the VDB in the Teiid Designer and type in any query or SQL statement.

After creating your models, you can test them by using the **Preview Data** action

By selecting a desired table object and executing the action, the results of a simple query will be displayed in the Data Tools SQL Results view. This action is accessible throughout the Teiid Designer in various view toolbars and context menus.

Previewable objects include:

- Relational table or view, including tables involving access patterns.
- Relational procedure.
- Web Service operation.
- XML Document staging table.



Note

If attempting to preview a relational access pattern, a web service operation or a relational procedure with input parameters, a dialog will request values for required parameters.

1.3.8. Model Object Extensions

Teild Designer in conjunction with Teild provides an extensible framework to define custom properties for model objects over-and-above what is defined in the metamodel. These custom property values are added to your VDB and included in your runtime metadata. This additional metadata is available to use in your custom translators for both source query manipulation as well as adjusting your result set data being returned.

In the 7.6 release, Teiid Designer introduces a new **Model Extension Definition (MED)** framework that will replace the current EMF-based Model Extension metamodel in a later 8.0 release.

This new MED framework provides the following improvements:

- Eliminate need for separate EMF metamodel.
- Simpler approach including reduction of extendable metamodels and metamodel objects (Relational, Web Services, XML Document, User Defined Functions) and replacing EMF terminology with basic object types.

- · Allows metamodels to be extended by multiple MEDs
- · MEDs are stored in models so no added dependency needed in VDB

Also see: Section 5.3, "Managing Model Object Extensions" and Section C.3.3, "Model Extension Definition Editor".

1.3.8.1. Model Extension Definition (MED)

The purpose of a MED is to define one or more sets of extension properties. Each set of extension properties pertains to one model object type (or metaclass). Each MED consists of the following:

- Namespace Prefix a unique identifier. Typically only a small number of letters and can be used as an abbreviation for the namespace URI.
- Namespace URI a unique URI.
- Extended Metamodel URI (Model Class) the metamodel URI that is being extended. Each metamodel URI also has model class and that is typically what is shown in the Designer. The model classes supported for extension are: Relational, Web Service, XML Document, and Function.
- Version (currently not being used)
- **Description** an optional description or purpose.
- Extended Model Object Types (Metaclasses) a set of model object types, or metaclasses, that have extension properties defined.
- Properties the extension property definitions grouped by model object type.

A **MED** file is an XML file with an extension of "mxd." A MED schema file (see attached modelExtension.xsd file) is used to validate a MED file. Here is a sample MED file:

The *MED Registry* is where the MEDs used by Designer are stored. MED files can be edited by opening the .mxd file in the *Section C.3.3, "Model Extension Definition Editor*".

1.3.8.2. Model Extension Definition Registry (MED Registry)

A MED registry keeps track of all the MEDs that are registered in a workspace. Only registered MEDs can be used to extend a model. There are 2 different types of MEDs stored in the registry:

- Built-In MED these are registered during Designer installation. These MEDs cannot be updated or unregistered by the user.
- User-Defined MED these are created by the user. These MEDs can be updated, registered, and unregistered by the user.

The MED Registry state is persisted and is restored each time a new session is started.

Dive Right In!

We are going to dive right into a couple examples of common tasks in this section. These examples will give you a quick introduction to the capabilities that are built into Designer to assist you with common design tasks. Specifically, we will introduce the following concepts:

• Guides

The **Guides View** is a good starting point for many common modeling tasks. The view includes categorized Modeling Actions and also links to Cheat Sheets for common tasks. The categorized Modeling Actions simply group together all of the actions that you'll need to accomplish a task. You can launch the actions directly from the Guides view, rather than hunting through the various Designer menus.

Cheat Sheets

The **Cheat Sheets** go beyond even the categorized Action Sets, and walk you step-by-step through some common tasks. At each step, the data entered in the previous step is carried through the process when possible.

After seeing the Guides and Cheat Sheets in action, subsequent chapters will offer detailed explanations of the various concepts and actions.

2.1. Guide Example

In this section, we introduce the **Guides View** by walking through a simple example. For this example, we will follow the **Model JDBC Source** Action Set. The actions appear in the following order:

- 1. Define Teiid Model Project
- 2. Create JDBC connection
- 3. Create source model for JDBC data source
- 4. Preview Data
- 5. Define VDB
- 6. Execute VDB

The action names are self explanatory. We will create a new "Model Project" in the workspace, then define our connection properties to a MySQL database. We will then connect to the database and import the 'metadata', creating a source model in Designer. Next we will 'preview' the database contents. Finally we will define a 'VDB' and then deploy it to a running Teiid Server to execute.

2.1.1. Model a JDBC Source

This section shows how to Model a JDBC Source, using the Guide View action set. We will connect to a MySQL database for this example, but you can use the same process to connect to any supported database.

1. Open Guides View

To open the **Teiid Designer's Guides** view, select the main menu's **Window > Show View > Other...** and select the **Teiid Designer > Guides** view in the dialog.

The Guides view is shown below, with the Model JDBC Source Action Set selected:

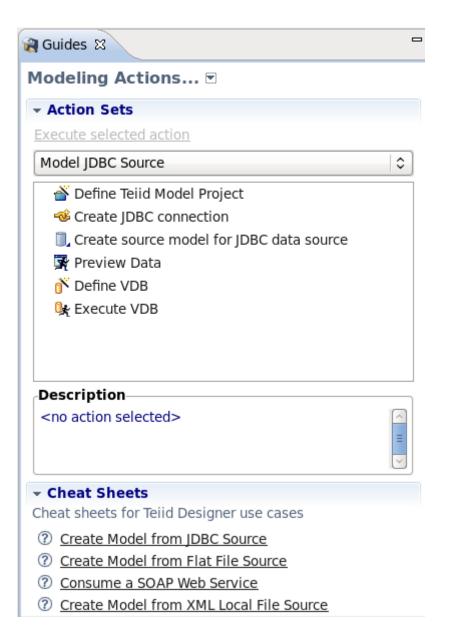


Figure 2.1. Guides View

2. Define Teiid Model Project

The **Define Teiid Model Project** action launches the **New Model Project Wizard**. In the Action Set list, double-click the action (or select it, then click 'Execute selected action'). The wizard is launched as shown below:

New Model Project	
New Model Project	
Create a new model project.	
Project name: MyProject	
✓ Use default location	
Location: //home/mdrillin/Runtime-Workspaces/7_7_x/MyProject	Browse
Working sets	
Add project to working sets	
Working sets:	Select
Sack Next > Cancel	Finish

Figure 2.2. New Project Wizard

Enter a project name, e.g. 'MyProject' for the name. Then click **Next**. The next page of the wizard is shown below:

	New Model Project	×
	ject Options to create folders for your new project	7
Create Fo	lders	_
🗹 Name	sources]
🗹 Name	views	
🗆 Name	schemas	
🗆 Name	web_services	
🗆 Name	functions	
🗆 Name	extensions	
?	< Back Next > Cancel Finish	

Figure 2.3. New Project Folders

Under 'Create Folders', de-select 'schemas' and 'web_services' - we won't need them for this example. Now, click **Finish** to exit the wizard. The project has now been created - your Model Explorer view should like like this:

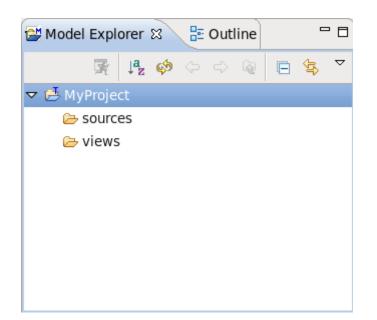


Figure 2.4. Model Explorer

3. Create JDBC connection

The **Create JDBC connection** action will create the 'Connection profile' for your database. The connection profile defines the properties and driver to be used when connecting to the database. In the Action Set list, double-click the action (or select it, then click 'Execute selected action'). The wizard is launched as shown below:

New Connection Profile	OX
Connection Profile	
Create a MySQL connection profile.	
Connection Profile Types:	
type filter text	
😫 Ingres	
🔮 MaxDB	
III ModeShape	=
😫 MySQL	
@ Oracle	
Name:	
TestMySQL	
Description (optional):	
? < Back Next > Cancel	Finish

Figure 2.5. Connection Profile Name and Type

Select the type of database that you are connecting to (e.g. MySQL), and enter a name for the connection profile, e.g. 'TestMySQL'. Click **Next**.

E	New Connection Profile	
Specify a Dri	ver and Connection Details	\diamond
Select a driver connection.	from the drop-down and provide login details for the	-
Drivers: MySC	QL JDBC Driver	Δ
Properties		
G <u>e</u> neral <u>O</u> p	tional	
D <u>a</u> tabase:	database	
URL:	jdbc:mysql://localhost:3306/database	Ξ
U <u>s</u> er name:	root	
Pass <u>w</u> ord:		
🗌 Sa <u>v</u> e pas	sword	~
✓ Connect whe	en the wizard completes <u>T</u> est Conne	ction
Connect eve	ry time the workbench is <u>s</u> tarted	
?	< Back Next > Cancel Finish	

Figure 2.6. Connection Profile properties

Now, select the driver and enter the login properties for your database. Click **Finish** to complete the profile creation.

4. Create source model for JDBC data source

The **Create source model for JDBC data source** action will now utilitze the **Connection profile** that you just created, to import the metadata from the database to create your Teiid Source Model. In the Action Set list, double-click the action (or select it, then click 'Execute selected action'). The wizard is launched as shown below:

E	Import Database via JDBC
Import Data	base via JDBC
Select the JDE model.	C source configuration for the database to be imported into a relational
Connection	Profile
TestMySQL	S New Edit
JDBC Metad	ata Processor
JDBC (defau	lt)
Properties	
Driver:	MySQL JDBC Driver
URL:	jdbc:mysql://localhost:3306/Accounts
User Name:	root
Password:	****
(?)	< Back Next > Cancel Finish
U	< Back Next > Cancel Finish

Figure 2.7. Select Connection Profile

On this page, select the 'TestMySQL' Connection profile that you created in the previous step. Click **Next**.

E	Import Database via JDBC	
Select Database Metadata Select the types of objects in the	e database to import.	
 Foreign Keys Include Incomplete FKs Indexes Unique Only Approximations Allowed Procedures 	Table Types TABLE VIEW LOCAL TEMPORARY	Select All Deselect All
?	< Back Next > Cancel	Finish

Figure 2.8. Select Database Metadata

On this page, select the database metadata that you want to import. When finished, click Next.

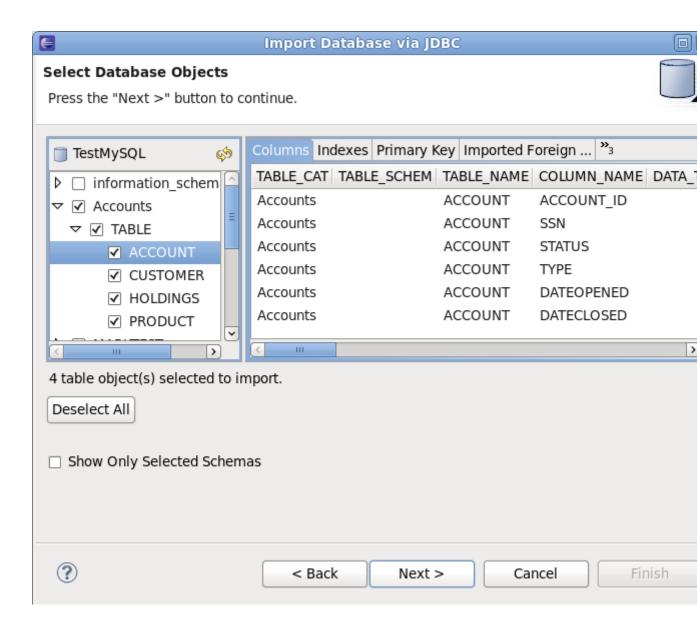


Figure 2.9. Select Database Objects

On this page, select the specific objects from the database that you want to import. When finished, click **Next**.

E	Import Database via JDBC	
Specify Impor	t Options	
Press the "Next	>" button to continue or the "Finish" button to finish.	
Relational Mo	odel Definition	
Model Name:	TestMySQL.xmi	
Into Folder:	MyProject/sources	
🗌 Make targe	t a view model	
🗆 Update (if e	existing model selected)	
	ualified Names (Example: partssupplier.dbo.PARTS) e For All Characters	
Case Op		
	All Upper Case (Example: Suppliers > SUPPLIERS)	
O Make	All Lower Case (Example: SUPPLIERS > suppliers)	
?	< Back Next > Cancel	Finish

Figure 2.10. Import Options

Finally, choose the name for the model to be created (defaults to 'profileName'.xmi). The 'Into Folder' field defines the target location for your new model. Select the 'MyProject/sources' folder. Now, click **Finish**. The source model has now been created - your Model Explorer view should like like this:

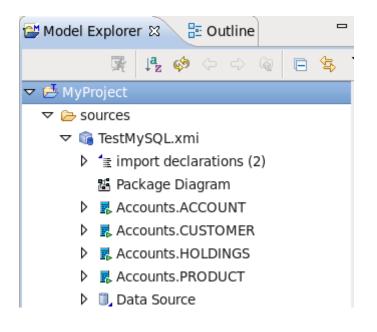


Figure 2.11. Model Explorer

5. Preview Data

All execution capabilities in Designer (Preview Data, VDB execution) require you to connect to a running Teild Server. See *Section C.2.3, "Teild View"* for instructions on establishing a Teild Server connection. Once you are connected to a Teild Server, you can proceed with the following steps.

The **Preview Data** action allows you to preview a sample of data rows from your source. In the Action Set list, double-click the action (or select it, then click 'Execute selected action'). In the dialog, select the source table you want to preview, as shown below:

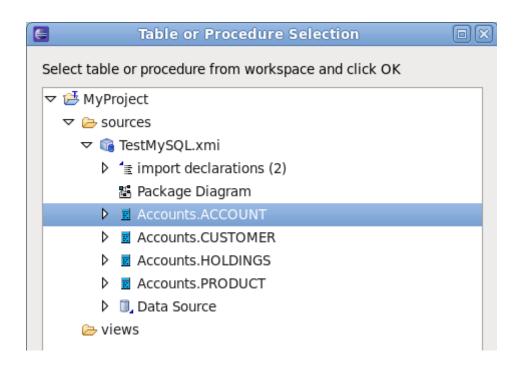


Figure 2.12. Select Preview Table

After selecting the table, click **OK**. Now, the preview results will be displayed:

🖹 Problems 👰 Error Log 🔲 SQL Results 🕱 🔌 Teiid Execution Plan								
Type query expression here Status Result1								
Status Operation Date Co		ACCOUNT_ID	SSN	STATUS				
✓ Succe select * fro Jun 8, 2012 Tr	1	19980002	CST01002	Persona				
	2	19980003	CST01003	Persona				
	3	19980004	CST01004	Persona				
	4	19980005	CST01005	Persona				
	5	19980006	CST01006	Persona				
	6	19990007	CST01007	Persona				
< <u> </u>	Solution	l 17 records shown						

Figure 2.13. Preview Results

6. Define VDB

The **Define VDB** action allows you to create a VDB (Virtual Database) artifact for deployment to a Teiid Server. In the Action Set list, double-click the action (or select it, then click 'Execute selected action'). The following dialog is displayed:

E	New VDB	
New VDB		
Press the "Fi	inish" button to finish.	
In Folder:	MyProject	
VDB Name:	myVDB	
Descriptio	n	
Selected M	Models	
Add Remove	📬 TestMySQL.xmi	
?	Cancel	Finish

Figure 2.14. New VDB

In the dialog, select the target 'In Folder' location where the VDB will be placed. Enter a Name for the VDB, for example 'myVDB'. Finally, select the models that will be included in the VDB. When finished, click **Finish**. The VDB will be created in your Teild Model Project - as shown in the following figure.

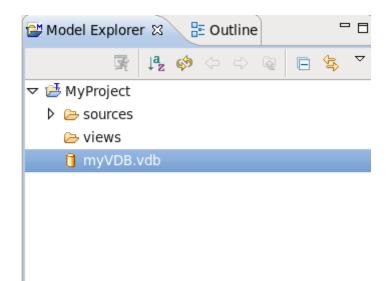


Figure 2.15. Model Explorer

7. Execute VDB

Finally, the **Execute VDB** action allows you to execute your VDB and run sample queries against it. In the Action Set list, double-click the action (or select it, then click 'Execute selected action'). In the dialog, select the VDB you want to execute, then click **OK**. The VDB will be deployed and executed, and the perpective will switch to the 'Database Development' perspective. You can now run queries against the VDB, as show in the following example:

			~			
📬 TestMySQL.xmi 📋 myVDB.vdb	ſ	🗄 *SQL Scrapbook0 🛽	ß			
Connection profile						
Type: Teiid_7.x		≎ <u>N</u> an	me:	myVDB -	localhost -	Teiid Conne
SELECT * FROM Accounts.ACCOU	лит					
<						Ш
🔲 SQL Results 😫 🛛 🐗 Teiid Executio	n Pla	n				
Type query expression here	Stat	us Result1				
Status Operation Date (ACCOUNT_ID		SSN		STATUS
✓ Succe select * fro Jun 8, 2012	1	19980002		CST010	02	Personal
✓ Succe SELECT * FI Jun 8, 2012	2	19980003		CST010	003	Personal
	3	19980004		CST010)04	Personal
	4	19980005		CST010)05	Personal
	5	19980006		CST010)06	Personal

Figure 2.16. Execute VDB Example

2.2. Cheat Sheet Example

In this section, we introduce **Cheat Sheets** by walking through a simple example. For this example, we will follow the **Consume a SOAP Web Service** Cheat Sheet.

2.2.1. Consume a SOAP Web Service

This section shows how to consume a SOAP Web Service, using a Cheat Sheet. We will demonstrate connection to a publicly accessible web service. You can use this process as an example for modeling other web services

1. Open the Cheat Sheet

You can access the **Cheat Sheet** from the Designer Menu. From the Designer main menu, select **Window > Show View > Other...**, then select **Help > Cheat Sheets** in the dialog.

Alternately, you can access the **Cheat Sheet** from the **Guide View**. A sample Guide view is shown below, with the **Consume a SOAP Web Service** Action Set selected:

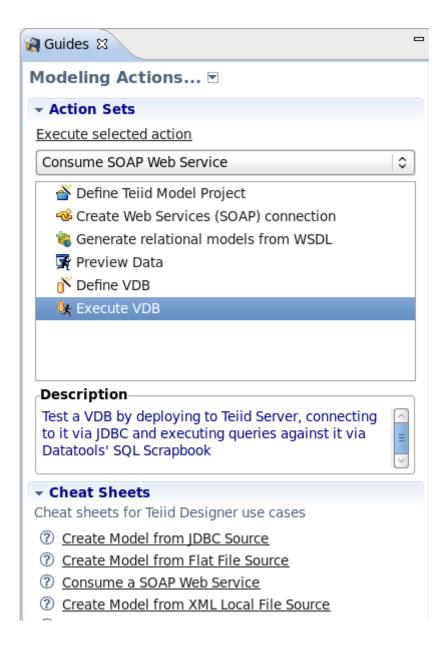


Figure 2.17. Guides View

To open the **Cheat Sheet** from the Guide View, expand the Cheat Sheet section in the lower portion of the Guide View, then select the **Consume a SOAP Web Service** link.

2. Begin the Cheat Sheet

The Consume a SOAP Web Service Cheat Sheet is shown below:

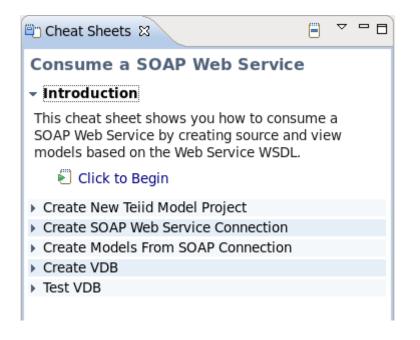


Figure 2.18. Consume SOAP Web Service Cheat Sheet

To start the Cheat Sheet process, expand the **Introduction** section, then select **Click to Begin**. The **Create New Teild Model Project** section opens, as shown.

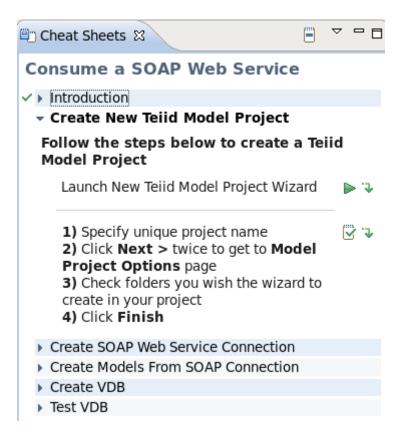


Figure 2.19. Create Model Project

Note that the instructions each section of sheet has basic outlining what to do at each Click step. ▶ 🤉

next to Launch New Teiid Model Project Wizard to launch the 'New Project' wizard.

Follow the wizard to create a new Model Project. For this example, we will use **SOAPProj** for our project name. On the second page of the wizard, select the 'sources' and 'views' folders. Click **Finish**. The new project is created.

In the Cheat Sheet, you can advance to the next step - once the wizard has completed. Click

🕑 🥆

to advance to the next step.

3. Create SOAP Web Service Connection

This section of the Cheat Sheet provides instructions for creating a connection profile for the SOAP Web Service, as shown below:

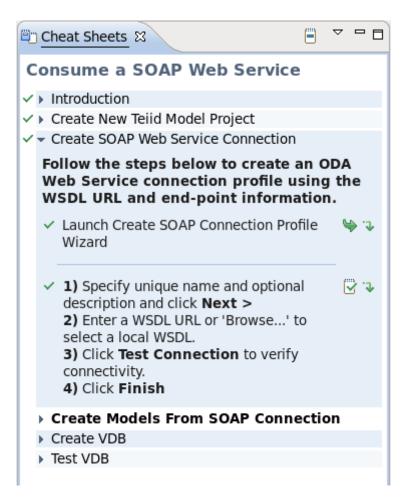


Figure 2.20. Create SOAP Connection Profile

Click

▶ 🤉

next to Launch Create SOAP Connection Profile Wizard to launch the wizard. The first page of the wizard is shown below:

New Connection Profile	
New Connection Profile	\Rightarrow
Create a SOAP Web Service connection profile.	
Connection Profile Types:	
type filter text	
A Web Services Data Source (SOAP)	
Name:	
CountryInfoConn	
Description (optional):	
? < Back Next > Cancel Finish	

Figure 2.21. Create SOAP Connection Profile

The **Web Services Data Source (SOAP)** profile type will be selected. Enter **CountryInfoConn** for the profile name, then click **Next**. The next page of the wizard is shown below:

E	New connection profile
Web Service Connection Click Next or Finish	Properties
Properties	
Profile Name	CountryInfoConn
Profile Description	
Connection URL or File Path	http://www.oorsprong.org/websamples.countryinfo/CountryInfoService
	Workspace
Description	
	nection profile provides the hook to inject the 'Endpoint' property into Set Connection Profile' action. This 'Endpoint' value is required by the '
(see 'Import > Teiid Desig	ner > WSDL File or URL >> Source and View Model (SOAP))
	Ī
?	< Back Next > Cancel

Figure 2.22. SOAP Connection Properties

The connection profile properties are entered on this page. Click on the URL... button, then enter the following URL: http://www.oorsprong.org/websamples.countryinfo/ CountryInfoService.wso?WSDL

Select 'None' for SecurityType, then click **OK** to complete the wizard. In the Cheat Sheet, you can now continue - once the wizard has completed. Click

to advance to the next step.

4. Create Models from SOAP Connection

This section of the Cheat Sheet provides instructions for creating relational models using the previously-created connection profile for the SOAP Web Service, as shown below:

Consume a SOAP Web Service	
✓ ► Introduction	
✓ ► Create New Teiid Model Project	
Create SOAP Web Service Connection	
Create Models From SOAP Connection	
Follow the steps below to create models using the previously defined connection profile.	e
✓ Launch the Consume SOAP Web Service Wizard	₽ 🗣
 Select the connection profile, then press Validate WSDL. Select the Port and Service Mode as desired. Select one or more operations under Select the desired WSDL Operations. Click Next >. Select the location and name for the Source and View Models to be created. Click Next >. For each operation, define the request and response XML documents: Select the Operation Select the Request tab, then select desired element(s) under schema contents and Add to Element Info. Select the Response tab, then select desired element(s) under schema contents and Add to Column Info. Select the Wrapper Procedure tab to view the generated procedure SQL. When all operation(s) have been defined, click Finish. 	3.0
Create VDB	
Test VDB	

Figure 2.23. Create Models from SOAP Connection





next to **Launch the Consume SOAP Web Service Wizard** to launch the wizard. The first page of the wizard is shown below:

		Create Bala	tional Madal-6	rom Web Comice	
				rom Web Service	
		Model from Web Serv	/ice		
ress the "N	ext >"	button to continue.			
Connectio	n Pro	file			
CountryIn					0
WSDL UR	L or L	ocation:			
http://www	v.oorsp	prong.org/websamples.cou	ntryinfo/Country	InfoService.wso?WSDL	
Coloct Dort	Court	ta dafa Can da Cana			
Select Port	\geq	tryInfoServiceSoap		<u></u>	
Binding	SOAP	11	Service Mode	PAYLOAD	
Select the	e desii	red WSDL Operations-			
Select A	IL	Operation			
		🗹 🗅 CapitalCity			
Deselect	All	🗌 🕞 CountriesUsingCu	rrency		
		CountryCurrency			
		🗌 📄 CountrvFlag			
Selection [Details	:			
Operation:			Diadian		
bindir port:	ıg:	CountryInfoServiceSoap CountryInfoService			
servio	e:	CountryInfoService			
<			Ш		
					Connel
?			< Back	Next >	Cancel

Figure 2.24. Consume SOAP Wizard

For **Connection Profile**, select the previously-created **CountryInfoConn** profile. The available WSDL Operations will then be displayed under **Select the desired WSDL Operations**. Select only the first **CapitalCity** Operation for this example. Click **Next** to proceed to the next page, as shown below:

E	Create Relational Model from Web Service
Models D	efinition
All inputs	OK. Click 'Next>' to define custom procedures.
Source	Model Definition
Location	SOAPProj/sources
Name	CountryInfoService.xmi
Status	
	model CountryInfoService.xmi does not exist and will be created and contain the required procedure.
View Mo	del Definition
Location	SOAPProj/views
Name	CountryInfoServiceView.xmi
Status	
View mo procedu	odel CountryInfoServiceView.xmi does not exist and will be created and contain your gene res.
Procedu	re Generation Options
O User-	specified Procedures (recommended)
Define us and resp	ser-specified request and response procedures from your WSDL schema elements. This op onse.
🔿 Defau	Ilt Procedures
Generate	default request and response procedures. A new procedure will be generated for each co
1	
?	< Back Next > Cancel

Figure 2.25. Consume SOAP Wizard

On the **Model Definition** page, the source and view model info section will be pre-filled. We will keep the names and location defaults for the source and view models. Click **Next** to proceed to the next page, as shown below:

E			Create Re	elatio	n <mark>al Model</mark> 1	from W	/eb Servic	e
	re Definitio umns are def		the response	e proce	edure result	set for t	he operatio	n: CapitalCity
Operati	ons							
Capital	City							
🗆 Over	write existin	g proced	ures for this	s opera	tion			
Request	Response	Wrapper	Procedure					
Genera	ted Procedur	e Name	CapitalCity	_reque	est			
BODY	HEADER							
Sche	ma Content	ts					Element	Info
	CapitalCity						Add	
	•••• sequence					_	Delete	e sCountryISOCo
		-	de				Up	
	🗟 strir	ng						
							Down	
Gener	ated SQL S	tateme	nt					
	E VIRTUAL P							
BEGIN								
SELI		NAME Ca	apitalCity, 🔉		MESPACES(D	DEFAULT	http://ww	w.oorsprong.org/
?					< Back		Next >	Cancel
							NEAL >	Cancer

Figure 2.26. Consume SOAP Wizard

On the **Procedure Definition** page, the **CapitalCity** Operation will be selected since it is the only one used for this example. On the **Request** tab, select the **sCountrylSOCode** element - then click the **Add** button. This will add the selected element to the request. Now select the **Response** tab, as shown below:

E Create Rela	tional Mod	el from Web Servic	e		
Procedure Definition Press the "Finish" button to finish.					
Operations					
CapitalCity					
Overwrite existing procedures for this ope	eration				
Request Response Wrapper Procedure					
Generated Procedure Name CapitalCity_res	ponse				
BODY HEADER					
Schema Contents	Column I	nfo			
	Root Path /ns:CapitalCityResponse				
	Add	Name	Ordinality	Data	
	Delete	CapitalCityResult		string	
a string			:	:	
	Down		1		
Generated SQL Statement					
CREATE VIRTUAL PROCEDURE BEGIN SELECT t.* FROM					
	< E	Back Next >	Can	cel	

Figure 2.27. Consume SOAP Wizard

On the **Response** tab, select the **Body** sub-tab. In the **Schema Contents**, select the **CapitalCityResult**, then click the **Add** button. This will add the selected element to the response.

Select the **Wrapper Procedure** tab to see the full Generated Procedure SQL, as shown below.

Create Relational Model from Web Service
Procedure Definition Press the "Finish" button to finish.
Operations
CapitalCity
Overwrite existing procedures for this operation
Request Response Wrapper Procedure
Generated Procedure Name
CapitalCity
Generated SQL Statement
CREATE VIRTUAL PROCEDURE BEGIN SELECT t.* FROM TABLE(EXEC CountryInfoServiceView.CapitalCity_request (CountryInfoServiceView.CapitalCity.sCountryISOCode) AS request, TABLE(EXEC CountryInfoService.invoke('SOAP11', null, REQUEST.xml_out, null)) AS response, TABLE(EXEC CountryInfoServiceView.CapitalCity_response(RESPONSE.result)) AS t; END
Output > Cancel

Figure 2.28. Consume SOAP Wizard

Click Finish to exit the wizard. In the Cheat Sheet, you can now continue. Click

to advance to the next step.

5. Create VDB

This section of the Cheat Sheet provides instructions for creating a VDB using the models that you created in the previous step. The Cheat Sheet section is shown below:

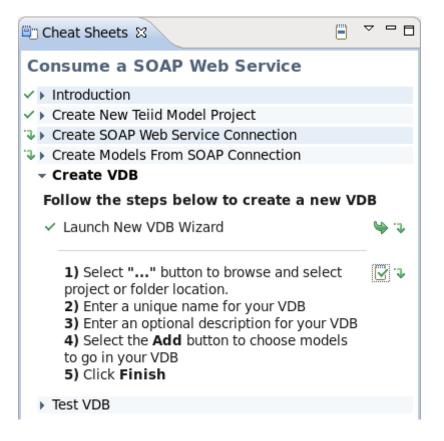


Figure 2.29. Create VDB

Click

```
▶ 🤉
```

next to Launch New VDB Wizard to launch the wizard. Follow the steps to create a VDB in your workspace. When complete, exit the wizard. In the Cheat Sheet, you can now continue. Click

🖓 🥆

to advance to the next step.

6. Test VDB

This	final	section	of	the	Cheat	Sheet	provides	inst	ructions
for	executing	the	VDB	create	ed in	the	previous	step.	Click
<i>,</i>									

```
▶ 🤉
```

next to **Launch Execute VDB Dialog** to launch the wizard. Select the previously-created VDB to execute it.

New Model Wizards

Models are the primary resource used by the Teiid Designer. Creating models can be accomplished by either directly importing existing metadata or by creating them using one of several **New Model** wizard options. This section describes these wizards in detail.

• The Teiid Designer currently supports the following types of models:

Section 3.1, "Creating New Relational Source Model"

G
 Section 3.2, "Creating New Relational View Model"

Section 3.3, "Creating XML Document View Model"

Section 3.4, "Creating XML Schema Model"

• 📬

Section 3.5, "Creating Web Service View Model"

Use one of the following options to launch the New Model Wizard.

New Model Wizard

 Choose 	the	File	>	New	>	Metadata	Model	action
ð								

• Select a project or folder in the Section C.2.1, "Model Explorer View" and choose the same action in the right-click menu.

Select	the	New	button	on	the	main	toolbar
- 23							
and	select		the	Metadata		Model	action
ð							



Note

Model names are required to be unique within Designer. When specifying model names in new model wizards and dialogues error messages will be presented and you will prevented from entering an existing name.

	New Model Wizard	×					
New Model Wizard							
Specify model	name and options then create model file.						
Location:	NorthwindSales	Browse					
		Diowsein					
Model Name:	Northwind						
Model Class:	Relational 🗘						
Model Type:	Source Model						
		,					
	el builder (optional):						
-	File Translator Procedures Web Service Translator Procedures						
	n an existing model of the same model class						
φ							
~							
?	< <u>Back</u> <u>N</u> ext > Cancel	<u>F</u> inish					

Figure 3.1. Import Wizard Selection Dialog

3.1. Creating New Relational Source Model

Create New Relational Source Model

- To create a new empty relational source model:
 - Step 1 Launch the New Model Wizard.

- Step 2 Specify a unique model name.
- Step 3 Select Relational option from Model Class drop-down menu.
- Step 4 Select Source Model from Model Type drop-down menu.
- Step 5 Click Finish.



Note

You can change the target location (i.e. project or folder) by selecting the **Browse...** button and selecting a project or folder within your workspace.

- In addition to creating a new empty relational source model, the following builder options are available:
 - Copy from existing model of the same model class.

3.1.1. Generate File Translator Procedures

This builder option allows construction of a relational model containing one or more of the procedures required for accessing file-based data via a file translator.

- To create a new relational model containing file translator procedures, complete *Create New Relational Source Model* above and continue with these additional steps:
 - Step 5 Select the model builder labeled Generate File Translator Procedures and click Next >. The Generate File Translator Procedures dialog will be displayed.
 - Step 6 Check one ore more of the Available File Translator Procedures, then Click Finish

New Model Wizard	\otimes					
Generate File Translator Procedures						
Generate default relational procedures compatible with Teiid File translator. Select desired procedures and 'Finish'						
Available File Translator Procedures						
getFiles('path/*.ext') return blob						
✓ getTextFiles('path/*.ext') return clob						
✓ saveFile('path', value) return void						
? < Back Next > Cancel	īnish					

Figure 3.2. Generate File Translator Procedures Dialog

3.1.2. Generate Web Service Translator Procedures

This builder option allows construction of a relational model containing one or more of the procedures required for accessing web-service-based XML data via a web s translator.

- To create a new relational model containing web-service-based translator procedures, complete *Create New Relational Source Model* above and continue with these additional steps:
 - Step 5 Select the model builder labeled Generate Web Service Translator Procedures and click Next >. The Generate Web Service Translator Procedures dialog will be displayed.
 - Step 6 Check one ore more of the Available Web Services Translator Procedures, then Click Finish

$\Theta \bigcirc \odot$	New Model Wizard
Generate Web Service Translator P	rocedures
Generate default relational proce desired procedures and 'Finish'	dures compatible with Teiid Web Se
Available Web Service Translato	Procedures
🗹 invoke(binding in String, action	in STRING, request in OBJECT, endpoir
☑ invokeHttp(action in STRING, re	quest in OBJECT, endpoint in STRING, i
?	< Back Next >

Figure 3.3. Generate Web Service Translator Procedures Dialog

3.1.3. Copy From Existing Model

This builder option performs a structural copy of the contents of an existing model to a newly defined model. You can choose a full copy or select individual model components for copy.

- To create a new relational model by copying contents from another relational source model, complete *Create New Relational Source Model* above and continue with these additional steps:
 - Step 5 Select the model builder labeled Copy from existing model of the same model class and click Next >. The Copy Existing Model dialog will be displayed.
 - Step 6 Select an existing relational model from the workspace using the browse button.



1

Note

An existing model will be pre-selected if a relational model in the workspace is selected in the *Section C.2.1, "Model Explorer View"* prior to starting the new model wizard.

• Step 7 - Check the Copy all descriptions option if desired. Click Finish

9		New M	odel Wizard		×
Copy an Existi Select an existing		copied.			
Existing Model: U	JSCustomers	xmi			
 ▶ ■ ACCOUNT ▶ ■ ACCOUNT ▶ ■ ACCOUNT ▶ ■ CUSTOMI ▶ ■ CUSTOMI ▶ ■ Data South 	THOLDINGS ER				
Copy all desc	riptions				
?		< <u>B</u> ack	Next >	Cancel	<u>F</u> inish

Figure 3.4. Copy An Existing Model Dialog

3.2. Creating New Relational View Model

Create New Relational View Model

- To create a new empty relational view model:
 - Step 1 Launch the New Model Wizard.
 - Step 2 Specify a unique model name.
 - Step 3 Select Relational option from Model Class drop-down menu.
 - Step 4 Select View Model from Model Type drop-down menu.
 - Step 5 Click Finish.



Note

You can change the target location (i.e. project or folder) by selecting the **Browse...** button and selecting a project or folder within your workspace.

- In addition to creating a new empty relational view model, the following builder options are available:
 - · Copy from existing model of the same model class.
 - Transform from existing model.

3.2.1. Copy From Existing Model

This builder option performs a structural copy of the contents of an existing model to a newly defined model. You can choose a full copy or select individual model components for copy.

- To create a new relational model by copying contents from another relational view model, complete *Create New Relational View Model* above and continue with these additional steps:
 - Step 5 Select the model builder labeled Copy from existing model of the same model class and click Next >. The Copy Existing Model dialog will be displayed.
 - Step 6 Select an existing relational model from the workspace using the browse button.



• Step 7 - Check the Copy all descriptions option if desired. Click Finish

۲	New Model Wizard	
Copy an Exis	ting Model	
Select an existin	ing model to be copied.	
Existing Model:	USCustomers.xmi	
-Model Contents	s	
	NT	
	NTHOLDINGS	
D 🛛 CUSTO		
👂 🧻 Data So	ource	
🖌 Copy all des	scriptions	
?	< Back Next > Cancel	Finish

Figure 3.5. Copy An Existing Model Dialog

3.2.2. Transform From Existing Model

This option is only applicable for creating a relational view model from a relational source model with the added feature of creating default transformations (SELECT * FROM SourceModel.Table_X) for each source table. The steps are the same as for the Section 3.2.1, "Copy From Existing Model" described above.

There is an additional option in the second page of the wizard which can automatically set the relational table's supports update property to false. If this is unchecked the default value will be true.

3.3. Creating XML Document View Model

Create XML Document View Model

- To create a new empty XML document view model:
 - Step 1 Launch the New Model Wizard.
 - Step 2 Specify a unique model name.
 - Step 3 Select XML option from Model Class drop-down menu.
 - Step 4 Select View Model from Model Type drop-down menu.
 - Step 5 Click Finish.



Note

You can change the target location (i.e. project or folder) by selecting the **Browse...** button and selecting a project or folder within your workspace.

- In addition to creating a new empty XML document view model, the following builder options are available:
 - Copy from existing model of the same model class.
 - Build XML documents from XML schema.

3.3.1. Copy From Existing Model

This builder option performs a structural copy of the contents of an existing model to a newly defined model. You can choose a full copy or select individual model components for copy.

- To create a new relational model by copying contents from another XML document view model, complete Create XML Document View Model above and continue with these additional steps:
 - Step 5 Select the model builder labeled Copy from existing model of the same model class and click Next >. The Copy Existing Model dialog will be displayed.
 - Step 6 Select an existing relational model from the workspace using the browse button.



• Step 7 - Check the Copy all descriptions option if desired. Click Finish

		New M	odel Wizard			X
Copy an Exis Select an existin		copied.				
Existing Model: Model Contents	NT NTHOLDINGS MER	xmi				
✓ Copy all des	criptions					
?	(< <u>B</u> ack	<u>N</u> ext >	Ca	ancel	<u>F</u> inish

Figure 3.6. Copy An Existing Model Dialog

3.3.2. Build XML Documents From XML Schema

This option creates an XML View document model based on a selected XML schema and its dependencies.

- To create a new XML document view model by from XML schema, complete *Create XML Document View Model* above and continue with these additional steps:
 - Step 5 Select the model builder labeled Build XML documents from XML schema and click Next >. The Select XML Schema dialog will be displayed.

• Step 6 - Select an existing schema model from the workspace using the browse button.



i

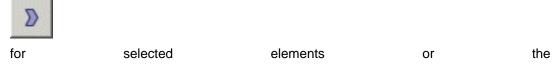
Note

An existing model will be pre-selected if an XSD model in the workspace is selected in the VDB explorer prior to starting the new model wizard. The schema must be found in the workspace so if you need to get one or more into the workspace use the XSD Schemas on file system importer.

1 🥹	New Model Wizard					
Select XML Scl	hema					
Select XML Schem	na file, then select Sch	nema Root Elements. C)r press Finish to	create an empty	/ XML Document.	
VML Cabarra Elas						
	/BooksProject/Books					
		ents to generate Virtua	al Documents			
Schema Root Ele	ements				ocuments	
e bookListing :	on : BooksNS:BookSet BooksNS:BookSetFlat d : BooksNS:BookSetN	:			ography : BooksN	
Reset						
Document Option	IS					
Build full virte	ual documents from th	ne schema				
 Build only first 	st level of each docum	ient <mark>(</mark> for large schema	where many port	tions will not be	used)	
Mapping Options						
🗹 Build Mappin	g Classes					
<						
0			- Deals	Nexts	Canad	
Ø			< <u>B</u> ack	<u>N</u> ext >	Cancel	

Figure 3.7. Select XML Schema Dialog

• Step 7 - Move the available schema root elements you want to become virtual documents in the new model over to the Virtual Documents list by using the arrow button





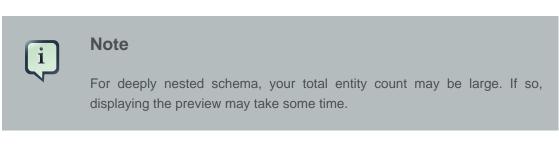
button to move all elements.

- Step 8 Select the appropriate document options and mapping options. Click Finish
- Step 9 Click Finish to create a model of all selected document entities or (optional) click Next > to view Selected Documents Statistics page which shows document entity statistics and gives you an idea the size of the model being created.

🥮 New Model Wizard 🛛 🕅							
Selected Documents S	Selected Documents Statistics						
	This is an overview of the documents to be generated. Select Next to preview and edit the document, or Finish to create the document with default settings						
Documents:	1						
Elements:	7						
Recursive Elements:	0						
Complex Subtype Elements:	0						
Attributes:	0						
Total entity count:	13						
?	< <u>B</u> ack <u>Next</u> > Cancel <u>F</u> inish						

Figure 3.8. Selected Documents Statistics Dialog

 Step 10 - (Optional) Click Finish to create a model of all selected document entities or click Next > to view Preview Generated Documents page that allows you to exclude document specific entities then click Finish.



🛞 New Model Wizard
Preview Generated Documents
Preview and edit the documents generated from the XSDs you selected. Elements marked with a boxed arrow contain recursive data. Elements marked with indicate that the builder stopped at the node.
▽ 🗹 🖹 bibliographyDocument
▽ 🗹 🕼 bibliography : BooksNS:Bibliography
▷ ▼ { } sequence
✓ 〈:〉 BooksNS = http://www.metamatrix.com/XMLSchema/DataSets/Books
✓ (:) xsd = http://www.w3.org/2001/XMLSchema
✓ 〈:〉 BookTypesNS = http://www.metamatrix.com/XMLSchema/DataSets/Books/BookDatatypes

Figure 3.9. Preview Generated Documents Dialog

3.4. Creating XML Schema Model

Create XML Schema Model

- To create a new empty XML schema (.xsd) model:
 - Step 1 Launch the New Model Wizard.
 - Step 2 Specify a unique model name.
 - Step 3 Select XML Schema (XSD) option from Model Class drop-down menu.
 - Step 4 Select Datatype Model from Model Type drop-down menu.
 - Step 5 Click Finish.



Note

You can change the target location (i.e. project or folder) by selecting the **Browse...** button and selecting a project or folder within your workspace.

• In addition to creating a new empty XML schema model, the following builder option is available:

• Copy from existing model of the same model class.

3.4.1. Copy From Existing Model

This builder option performs a structural copy of the contents of an existing model to a newly defined model. You can choose a full copy or select individual model components for copy.

- To create a new relational model by copying contents from another XML schema model, complete *Create XML Schema Model* above and continue with these additional steps:
 - Step 5 Select the model builder labeled Copy from existing model of the same model class and click Next >. The Copy Existing Model dialog will be displayed.
 - Step 6 Select an existing relational model from the workspace using the browse button.



• Step 7 - Check the Copy all descriptions option if desired. Click Finish

🛞 New Model	Wizard
Copy an Existing Model Select an existing model to be copied.	
Existing Model: USCustomers.xmi	
 ▶ ■ ACCOUNT ▶ ■ ACCOUNTHOLDINGS ▶ ■ CUSTOMER ▶ ■ Data Source 	
Copy all descriptions	
? < <u>B</u> ack	Next > Cancel Finish

Figure 3.10. Copy An Existing Model Dialog

3.5. Creating Web Service View Model

Create Web Service View Model

- To create a new empty web service view model:
 - Step 1 Launch the New Model Wizard.
 - Step 2 Specify a unique model name.
 - Step 3 Select Web Service option from Model Class drop-down menu.
 - Step 4 Select View Model from Model Type drop-down menu.
 - Step 5 Click Finish.



Note

You can change the target location (i.e. project or folder) by selecting the **Browse...** button and selecting a project or folder within your workspace.

- In addition to creating a new empty web service view model, the following builder options are available:
 - Copy from existing model of the same model class.
 - Build from existing WSDL file(s) or URL.

3.5.1. Copy From Existing Model

This builder option performs a structural copy of the contents of an existing model to a newly defined model. You can choose a full copy or select individual model components for copy.

- To create a new relational model by copying contents from another web service view model, complete *Create Web Service View Model* above and continue with these additional steps:
 - Step 5 Select the model builder labeled Copy from existing model of the same model class and click Next >. The Copy Existing Model dialog will be displayed.
 - Step 6 Select an existing relational model from the workspace using the browse button.
 - Step 7 Check the Copy all descriptions option if desired. Click Finish

		New Moo	lel Wizard		
Copy an Existin Select an existing	-	pied.			
Existing Model: U	SCustomers.xn	ni			
 ▷ I ACCOUNT ▷ I ACCOUNT ▷ I CUSTOME ▷ I Data Sour 	HOLDINGS R				
✓ Copy all descr	ptions				
?		< <u>B</u> ack	<u>N</u> ext >	Cancel	Finish

Figure 3.11. Copy An Existing Model Dialog

3.5.2. Build From Existing WSDL File(s) or URL

This builder option creates a Web service model based on a user-defined WSDL file and its referenced schemas. In addition, applicable XML schema files and XML View document models (optional) are created.

- To create a new relational model by copying contents from another web service view model, complete *Create Web Service View Model* above and continue with these additional steps:
 - Step 5 Select the model builder labeled Build from existing WSDL file(s) or URL and click Next >.
 - The remaining wizard steps are identical to those found using the Section 4.8, "Import WSDL Into Web Service" action option.

3.5.3. Build From Relational Models

See Section 6.4.1, "Create Web Service Action"

3.5.4. Build From XML Document View Models

Web Service models and their corresponding **Interfaces** and **Operations** can be generated in Teiid Designer from **XML View** model components. Namely, XML View Documents and XML View Document roots.

- To create a new Web service model from XML components::
 - Step 1 Select either a single XML Document or single XML Document root in Section C.2.1, "Model Explorer View".
 - Step 2 Right-click select Modeling > Create Web Service action
 - Step 3 Fill in missing properties in Web Service Generation Wizard shown below.

(@	Web S	ervice Generation V	Vizard	×			
Generate a Web Service.							
Click 'Finish' to generate the Web Service. Click 'Cancel' to abort.							
Specify location for generated model(s) (i.e. Project or Folder)							
BooksProject Browse							
Bold Blue text indicates an existing workspace entity. Bold Red text indicates an error.							
WebService Model:	rvice Model: bookCollection_WS						
Interface Name:	bookCollection						
Operation Name:	getbookCollection						
Operation Definition							
Input Message E	lement:	e bookCollection :	BooksNS:BookSetNe	ested			
Input Message Name:		bookCollectionInput					
Output Message Element:		e bookCollection : E	BooksNS:BookSetNe	sted			
Output Message	Name:	bookCollectionOuput					
?			Cancel	<u>F</u> inish			

Figure 3.12. Generate A Web Service Dialog

• Step 4 - Click Finish to generate model. When model generation is complete, a confirmation dialog should appear. Click OK.



Figure 3.13. Generation Completed Dialog



Note

Users can change the **Web Service Model** and **Interface Name** values (via "..." buttons) to use existing **Web service** model components. This will create a new operation in an existing model.

Importers

The **Import Wizard** provides a means to create a model based on the structure of a data source, to convert existing metadata (i.e. **WSDL** or **XML Schema**) into a source model or to load existing metadata files into the current VDB.

To launch the **Import Wizard**, choose the **File > Import** action or select a project, folder or model in the tree and right-click choose "**Import...**"

🔲 Import 🕅
Select
Choose import source.
<u>S</u> elect an import source:
type filter text
Plug-in Development
Run/Debug
Digital Team
▼ 🦻 Teiid Designer
DDL File >> Source or View Model
Designer Text File >> Source or View Models
File Source (Flat) >> Source and View Model
File Source (XML) >> Source and View Model
JDBC Database >> Source Model
WSDL File or URL >> Source Model
B WSDL File or URL >> Web Service Model
S XML Schemas
(?) < Back Next > Cancel Finish

Figure 4.1. Import Wizard

4.1. Import DDL

Source relational models can be created by importing DDL.

• You can create relational source models from your DDL using the steps below.

• Step 1 - In Model Explorer choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

- Step 2 Select the import option Teild Designer > DDL File >> Source or View Model and click Next>
- Step 3 Select existing DDL from either Choose from file system... or Choose from workspace.... set the Model folder location, enter or select valid model name, set Model type (Source Model or View Model), set desired options and click NEXT> (or Finish if enabled)

E	Import DDL	×
Provide DDL	source	
	al source model "ProductsSQLServer" using D e-Workspaces/test_7_1_0_E/DDLTest/ProductsS	
DDL file:	/home/blafond/Runtime-Workspaces/test	Choose from file system Choose from workspace
Model folder:	(/DDLTest	Choose
Model name:	ProductsSQLServer	Choose
Model type:	Source Model	
NAS100Comp AMEXINTComp PrimaryBusine) GO	VARCHAR(10) NOT NULL, VARCHAR(60), VARCHAR(15), VARCHAR(10), VARCHAR(10),	
.<.		
?	< <u>B</u> ack <u>N</u> ext >	Cancel <u>Finish</u>

Figure 4.2. DDL Import Options

• **Step 4** - If *NEXT*> is pressed, a difference report is presented for viewing or de-selecting individual relational entities. Press *Finish* to complete.

🔚 Import DDL 🕅
Review Model Updates
Review the changes that will be applied to your model to bring it up to date with the source metadata. Uncheck any changes you do not wish to apply.
Only In Old Model (OLD): 0 Changes (A): 0 Only In New Model (NEW): 0
马····································
 ✓ file:/home/blafond/Runtime-Workspaces/test_7_1_0_E/DDLTest/ProductsSQLS ▶ ♥ ■ ProductData ▶ ♥ ■ ProductSymbols
Selection Details
< Back Next > Cancel Finish

Figure 4.3. Review DDL Updates Dialog

4.2. Import From JDBC Database

 You can create relational source models from your JDBC source schema data using the steps below.



• Step 1 - In Model Explorer choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

- Step 2 Select the import option Metadata Modeling > JDBC Database >> Source Model and click Next>
- Step 3 Select existing or previous connection profile from the drop-down selector or press New... button to launch the New Connection Profile dialog (See Eclispe Data Tools documentation) or Edit... to modify/change an existing connection profile prior to selection. Note that the Connection Profile selection list will be populated with only JDBC Database connections.

,	Import Database via JDBC	Σ
Select a JDBC	C source configuration	
Select the JDBC model.	source configuration for the database to be imported into a relational	
Connection	Profile	
	C New	Edit
JDBC Metad	lata Processor	
		0
Properties-		
Driver:		
URL:		
User Name:		
Password:		
(?)	< <u>B</u> ack <u>N</u> ext > Cancel	Finish

Figure 4.4. Select JDBC Source Configuration Dialog

Because JDBC databases are different, special processing of your metadata to be required in order to convert datatypes or to interpret your metadata. The **JDBC Metadata Processor** drop-down selector will be auto-selected based on your selected connection profile. Special processors are available for DB2, Modeshape, ODBC, Oracle, PostgeSQL, SQL Server and Sybase. For all other DB's ajdefault JDBC processor is available.

• Step 4 - After selecting a *Connection Profile*, input password (if not provided). Press Next> (or Finish if enabled)

	Import Database via JDBC
Select a JDB	C source configuration
Press the "Ne>	t >" button to continue or the "Finish" button to finish.
Connectior	Profile
PartsOracle	11 New Edit
JDBC Meta	data Processor
Oracle	0
Properties	
Driver:	Oracle 11 Thin Driver
URL:	jdbc: oracle: thin: @ db0025. www. my db. com: 1521: db25
User Name:	partssupplier
Password:	**
?	< <u>B</u> ack <u>N</u> ext > Cancel <u>F</u> inish

Figure 4.5. Select JDBC Source Configuration Dialog

• Step 5 - On the Select Database Metadata page, select the types of objects in the database to import. Press Next> (or Finish if enabled).

Select Database Metadata Press the "Next >" button to cont	inue or the "Finish" button to finish.	
Foreign Keys	Table Types	
✓ Indexes	SYNONYM]
Unique Only	TABLE VIEW	
Approximations Allowed		Select All
Procedures		Deselect A

Figure 4.6. Select Database Metadata Dialog

 Step 6 - On the Select Database Objects page, view the contents of the schema, or change selections. Select which database schema objects will be used to construct relational objects. Press Next> (or Finish if enabled)

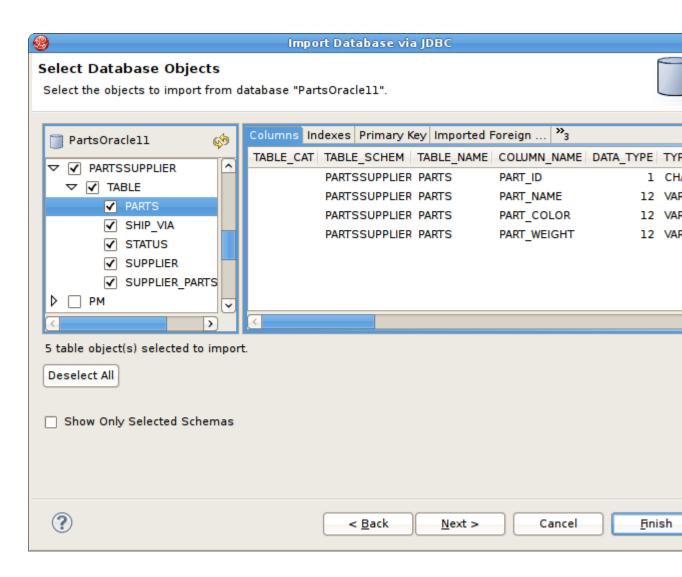


Figure 4.7. Select Database Options Dialog

• Step 7 - On the Specify Import Options page, specify desired Model Name as well as any other options used to customize the naming of your relational objects. Press Finish to complete.

	Import Database via JDBC	×
Specify Impo	ort Options	\square
Specify the nam database "Parts	ne, location, and import options for the new relational model of sOracle11.xmi".	
Relational N	1odel Definition	
Model Name:	PartsOracle11.xmi	
Into Folder:	PartsTest	
🗌 Update (if (existing model selected)	
🔲 Include Cata	alog For Fully Qualified Names	
Model Obje	ct Names (Tables, Procedures, Columns, etc)	
☑ Use Fully Q)ualified Names (Example: partssupplier.dbo.PARTS)	
Change Ca	se For All Characters	
Case 0)ptions	
Make	e All Upper Case (Example: Suppliers > SUPPLIERS)	
O Make	e All Lower Case (Example: SUPPLIERS > suppliers)	
]
?	< <u>B</u> ack <u>N</u> ext > Cancel <u>F</u>	inish

Figure 4.8. Specify Import Options Dialog

During the *Finish* processing, a monitor will be displayed providing feedback on the import progress.

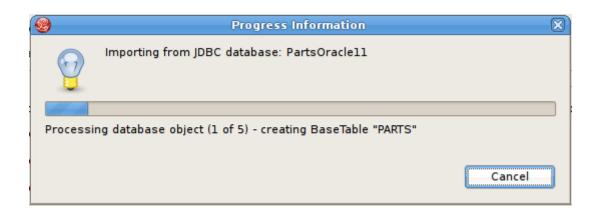


Figure 4.9. JDBC Import Progress Dialog

4.3. Import From Flat File Source

 You can import metadata from your flat file data sources and create the metamodels required to query your data in minutes. Using the steps below you will define your flat file data source, configure your parsing paramaters for the flat file, generate a source model containing the standard Teiid flat file procedure and create view tables containing the SQL defining the column data in your flat file.

Teiid supports Flat Files as data sources. Teiid Designer provides an Import wizard designed to assist in creating the metadata models required to access the data in your flat files. As with Designer's JDBC, Salesforce and WSDL importers, the Flat File importer is based on utilizing a specific Data Tools Connection Profile.

The results of the importer will include a source model containing the getTextFiles() procedures supported by Teiid.

The importer will also create a new view model containing a view table for your selected flat file source file. Within the view table will be generated SQL transformation containing the "getTextFiles()" procedure from your source model as well as the column definitions and parameters required for the Teiid TEXTTABLE() function used to query the data file. You can also choose to update an existing view model instead of creating a new view model.

The TEXTTABLE function, as defined in the Teild documentation, processes character input to produce tabular ouptut. It supports both fixed and delimited file format parsing. The function itself defines what columns it projects. The TEXTTABLE function is implicitly a nested table and may be correlated to preceeding FROM clause entries.

```
TEXTTABLE(expression COLUMNS <COLUMN>, ... [DELIMITER char] [(QUOTE|
ESCAPE) char]
[HEADER [integer]] [SKIP integer]) AS name
```

Teiid Designer will construct the full SQL statement for each view table in the form:

SELECT
A.Name, A.Sport, A.Position, A.Team, A.City, A.StateCode, A.AnnualSalary
FROM
(EXEC PlayerDataSource.getTextFiles('PlayerData.txt')) AS f,
TEXTTABLE(f.file COLUMNS Name string, Sport string, Position string,
Team string, City string, StateCode string, AnnualSalary string
HEADER 2 SKIP 3) AS A

To import from your flat file source follow the steps below.

• Step 1 - In Model Explorer choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

 Step 2 - Select the import option Teild Designer > File Source (Flat) >> Source and View Model and click Next>

Import Import	×
Select	
Choose import source.	r¥g
<u>S</u> elect an import source:	
type filter text	4
Plug-in Development	<u>^</u>
Run/Debug	
👂 🗁 Team	
▼ 🔁 Teiid Designer	
DDL File >> Source or View Model	
Designer Text File >> Source or View Models File Source (Flat) >> Source and View Model	
File Source (XML) >> Source and View Model	
], JDBC Database >> Source Model	
Salesforce >> Source Model	
WSDL File or URL >> Source Model	
B WSDL File or URL >> Web Service Model S XML Schemas	
< Back Next > Cancel	Finish

Figure 4.10. Import from Flat File Source

Step 3 - Select existing or previous connection profile from the drop-down selector or press New... button to launch the New Connection Profile dialog (See Eclispe Data Tools documentation) or Edit... to modify/change an existing connection profile prior to selection. Note that the Flat File Source selection list will be populated with only Flat File connection profiles.

After selecting a **Connection Profile**, the file contents of the folder defined in the connection profile will be displayed in the **Available Data Files** panel. Check the the data file you wish to process. The data from this file, along with your custom import options, will be used to construct a view table containing the required SQL transformation for retrieving your data and returning a result set.

Lastly enter or unique source model name in the **Source Model Definition** section at the bottom of the page or select an existing source model using the browse button. Note the

Model Status section which will indicate the validity of the model name, whether the model exists or not and whether the model already contains the getTextFiles() procedure. In this case, the source model nor the procedure will be generated.

When finished with this page, click Next>.

Data File Source Selection Press the "Next >" button to continue. Data File Source EmployeeData Chailable Data Files Folder location: (nome/blafond/TestDesignerFolder/FlatFileData/employee-data Data File Name EmployeeData txt PlayerData txt PlayerData txt PlayerData txt Selected Data File: PlayerData txt Model Definition Location TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFileS() procedure.		Import From Flat File Source
Data File Source EmployeeData New Edit Available Data Files Folder location: /home/blafond/TestDesignerFolder/FlatFileData/employee-data Data File Name EmployeeData.txt ImployeeData.txt Selected Data File: PlayerData.txt Source Model Definition Location TestXmilmport Name: TextFileProcedures.xmi Model Status ExistTiNG MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.	Data File	e Source Selection
EmployeeData New Edit Available Data Files Folder location: /home/blafond/TestDesignerFolder/FlatFileData/employee-data Data File Name EmployeeData.bt PlayerData.bt PlayerData.bt PlayerData.tt Selected Data File: PlayerData.txt Source Model Definition Location TestXmlimport Name: TestXmlimport Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.	Press th	e "Next >" button to continue.
EmployeeData New Edit Available Data Files Folder location: /home/blafond/TestDesignerFolder/FlatFileData/employee-data Data File Name EmployeeData.bt PlayerData.bt PlayerData.bt PlayerData.tt Selected Data File: PlayerData.txt Source Model Definition Location TestXmlimport Name: TestXmlimport Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Available Data Files Folder location: // home/blafond/TestDesignerFolder/FlatFileData/employee-data Data File Name EmpDataFixedWidth.txt EmployeeData.txt Image: PlayerData txt Selected Data File: PlayerData.txt Source Model Definition Location TestXmlimport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.	Data Fi	le Source
Folder location: /home/blafond/TestDesignerFolder/FlatFileData/employee-data Data File Name	Employe	eeData \$ New Edit
Folder location: /home/blafond/TestDesignerFolder/FlatFileData/employee-data Data File Name	Availab	le Data Files
Data File Name EmpDataFixedWidth.txt EmployeeData.txt ImployeeData.txt Selected Data File: PlayerData.txt Source Model Definition Location TestXmlimport Name: TextFileProcedures.xmi ImployeeData.txt Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
EmplataFixedWidth.txt EmployeeData.txt PlayerData.txt Selected Data File: PlayerData.txt Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
EmployeeData.txt PlayerData.txt Selected Data File: PlayerData.txt Source Model Definition Location TestXmlimport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
PlayerData.txt Selected Data File: PlayerData.txt Source Model Definition Location TestXmlimport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Selected Data File: PlayerData.txt Source Model Definition Location TestXmlimport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Source Model Definition Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.	Selected	Data File: PlayerData.txt
Location TestXmlImport Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		
Name: TextFileProcedures.xmi Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.	Source	Model Definition
Model Status EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.	Location	TestXmlimport
EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.	Name:	TextFileProcedures.xmi
EXISTING MODEL: Source model TextFileProcedures.xmi already contains required getTextFiles() procedure.		-Model Status
getTextFiles() procedure.		
Cancel Enish		getTextFiles() procedure.
(?) < <u>B</u> ack <u>N</u> ext > Cancel		
(?) < <u>B</u> ack <u>N</u> ext > Cancel		
< <u>B</u> ack <u>N</u> ext > Cancel <u>Enish</u>		
	\odot	< <u>B</u> ack <u>N</u> ext > Cancel <u>Finish</u>

Figure 4.11. Data File Source Selection Page

• Step 4 - The next page, titled Flat File Column Format Definition, requires defining the format of your column data in the file. The options are Character delimited and Fixed width.

This page contains a preview of the contents of your file to aid in determining the format. The wizard defaults to displaying the first 20 lines, but you can change that value if you wish.

When finished with this page, click **Next>**.

Import From Flat File Source
Flat File Column Format Definition
Press the "Next >" button to continue.
Selected Data File: EmployeeData.txt
File Preview Options
Number of lines in file : 52 Number of preview lines 20
Select Column Format
Character delimited Fixed width
File Contents Preview
LastName,FirstName,MiddleName,EmpId,Department,AnnualSalary,Title,HomePhone,M
Kisselmeyer,Abbiegale,Tikvica ,9000059,G,64000.00,MGMT WannaBe,670-270-7947,9
Glore,Diodie,Vojvoda ,9000060,G,71000,Associate,480-650-9750,9000073,127 State
Dawson,Pinckney,Ostoja ,9000061,G,71000,Associate,110-400-3600,9000073,135 St
Waldrip,Trixie,Curic ,9000062,G,57000,Newbie,820-210-7045,9000073,136 State S
Kitchen,Zilpha,Buic ,9000063,G,60000,MGMT WannaBe,660-390-3785,9000073,138 St
Wakeman,Gerard,Vlahovic ,9000064,G,78000,Newbie,700-190-5880,9000073,130 State
Rafferty,Dock,Korda ,9000065,G,70000,Newbie,400-190-6192,9000073,128 State St
Kersavage,Zelda,Mjesecevic ,9000066,G,56000,MGMT WannaBe,802-930-1482,9000073
Zummer,Gerda,Milan ,9000067,G,69000,Newbie,920-100-9701,9000073,131 State St.
Davies,Allwyne,Radic ,9000068,G,61000,Associate,470-820-6096,9000073,126 State
Deanford, Abe, Skrabalo , 9000069, G, 67000, Associate, 907-660-8233, 9000073, 124 Sta
Garcia,Orsal,Ucovic ,9000070,ML,79000,CF0,480-490-7710,9000075,150 State St.,
Zook,Orson,Bendevis ,9000071,G,71000,Newbie,350-260-8654,9000073,133 State St
Rainier,Adelaid,Marinovic ,9000072,G,67000,Newbie,316-550-3499,9000073,125 St
Nealon,General,,9000073,G,77000,C00,203-420-3113,9000075,129 State St.,Hartfo
Garahana, Jarrod, Cvjetkovic ,9000074, G,66000, Associate, 620-430-1782,9000073, 13
Neely,Petronella,Goravica ,9000075,G,80000,CE0,230-320-6330,9000076,134 State
(?) < <u>Back</u> <u>Next</u> > Cancel

Figure 4.12. Data File Source Selection Page

Step 5a : Character Delimited Option - The primary purpose of this importer is to help you create a view table containing the transformation required to query the user-defined data file. This page presents a number of options you can use to customize the Generated SQL Statement , shown in the bottom panel, for the character delimited option. Specify header options (Column names in header, header line number and first data line number), Parse selected row, changed character delimiter and edit the TEXTTABLE() function options. See the Teiid User's Guide for details on the TEXTTABLE() function.

If columns names are not defined in a file header or if you wish to modify or create custom columns, you can use the **ADD**, **DELETE**, **UP**, **DOWN** to manage the column info in your SQL.

When finished with this page, click Next>.

🔳 import	: From Flat File Source 🛞
Flat File Delimited Columns Pars Press the "Next >" button to continue.	er Settings
Selected Data File: EmployeeData.txt	
Format Options	File Contents Preview
☑ Column names in header	LastName,FirstName,MiddleName,EmpId,De
Header line # 1	Kisselmeyer,Abbiegale,Tikvica ,9000055
Data line # 2	Glore,Diodie,Vojvoda ,9000060,G,71000,
	Dawson,Pinckney,Ostoja ,9000061,G,7100
Parse Selected Row	Waldrip,Trixie,Curic ,9000062,G,57000,
Edit Delimiter Character	Kitchen,Zilpha,Buic ,9000063,G,60000,M
Edit TEXTTABLE() function options	Wakeman,Gerard,Vlahovic ,9000064,G,780 🗸
	<>
Column Information	
Column Name	Datatype
🔺 LastName	string
A FirstName	string
MiddleName	string
🖾 Empld	string
ADD DELETE	UP DOWN
A.HomePhone, A.Mgrld, A.Street, A.City FROM (EXEC AAAA.getTextFiles('EmployeeDa string, FirstName string, MiddleName st	ame, A.Empld, A.Department, A.AnnualSalary, A.Title, A.State, A.ZipCode ata.txt')) AS f, TEXTTABLE(file COLUMNS LastName tring, Empld string, Department string, AnnualSalary Mgrld string, Street string, City string, State string,
?	Back Next > Cancel Finish

Figure 4.13. Flat File Delimited Columns Options Page

To aid in determining if your parser settings are correct you can select a data row in your **File Contents Preview** section and click the **Parse Selected Row** button. A dialog will be displayed showing the list of columns and the resulting column data. If your column data is not what you expected, you'll need to adjust your settings accordingly.

Parsed Column Data	×
The following list contains the parsed column data from the selected line	73
Parsed Column Data	
lastName : Glore	
FirstName : Diodie	
MiddleName : Vojvoda	
Empld : 9000060	
Department : G	
AnnualSalary : 71000	
Title : Associate	
HomePhone : 480-650-9750	
Mgrld : 9000073	
Street : 127 State St.	
City : Los Angeles	
State : CA	
ZipCode : 10005	
Cancel	ок

Figure 4.14. Parse Column Data Dialog

Step 5b : Fixed Column Width Option - The primary purpose of this importer is to help you create a view table containing the transformation required to query the user-defined data file. This page presents a number of options you can use to customize the Generated SQL Statement, shown in the bottom panel, for the fixed column width option. Specify header options (Column names in header, header line number and first data line number), Parse selected row, changed character delimiter and edit the TEXTTABLE() function options. See the Teiid User's Guide for details on the TEXTTABLE() function.

If columns names are not defined in a file header or if you wish to modify or create custom columns, you can use the **ADD**, **DELETE**, **UP**, **DOWN** to manage the column info in your SQL.

You can also utilize the cursor postion and text length values in the upper left panel to determine what your column widths are in your data file.

When finished with this page, click Next>.

ļ	Im	port From Flat	File So	urce			
lat File Fixed Colu	mns Width Par	ser Settings					
Press the "Next >" but	ton to continue.						
Selected Data File: Em	pDataFixedWidth.txt	:					
Farmet Outlines		Den inc					
Format Options	File Contents						
Data line # 2		tment string integer WID					-
Cursor Position 24	//	incogor mit		50,000	5011119		, , , , ,
Text Length 12	Kisselmeyer	-	Tikvi		900005	9 G	64000
Text Length 12	Glore	Diodie	Vojvo		900006	60 G	71000
	Dawson	Pinckney	Ostoj		900006		71000
	Waldrip	Trixie	Curio		900006		57000
	Kitchen Wakeman	Zilpha	Buic		900006		60000
	Wakeman Refferty	Gerard	Vlaho		900006		78000
					Sector		>
Column Options	Column Inform	nation]
ADD	Column Name			Datatype		Width	
DELETE	🔺 firstName			string		12	
DELETE	🛛 middleName			string		12	
UP	🖪 lastName			string		12	
DOWN							
]
eiid TEXTTABLE()	Function Optio	ns					
🗌 Include HEADER 🔲	Include SKIP	🗌 In	clude Qt	JOTE 🕛		Include	ESCAPE \
Generated SQL Sta	atement						
SELECT A firstName A midd	laNama A column	2					2
A.firstName, A.midd FROM	iewame, A.column_	-3					
(EXEC ssss.getTextF				BLE(file C	OLUMNS f	irstName	e string width
12, middleName string	width 12, column_	3 string width 12	2) AS A				
~					-		
(?)		< Back		Next >	C	ancel	Finish

Figure 4.15. Flat File Fixed Columns Width Options Page

• Step 6 - On the View Model Definition page, select the target folder location where your new view model will be created. You can also select an existing model for your new view tables. Note the Model Status section which will indicate the validity of the model name, whether the model exists or not. Lastly, enter a unique, valid view table name. Press Finish to generate your models and finish the wizard.

	Import From Flat File Source	
View Model Definit	ion	E.
Press the "Finish" butt	ton to finish.	
Selected Data File: Play	yerData.txt	
View Model Defini	ition	
Location	TestXmlimport	
Name:	MyEmployeeViews.xmi	
	Model Status	
	EXISTING MODEL: New view tables will be created in your existing view model	
	MyEmployeeViews.xmi on FINISH.	
New view table name:	PlayerDataTable	
New view table name.		
?	< Back Next > Cancel	<u>F</u> ini
<u> </u>		

Figure 4.16. View Model Definition Page

When your import is finished your source model will be opened in an editor and show a diagram containing the your getTextFiles() procedure.

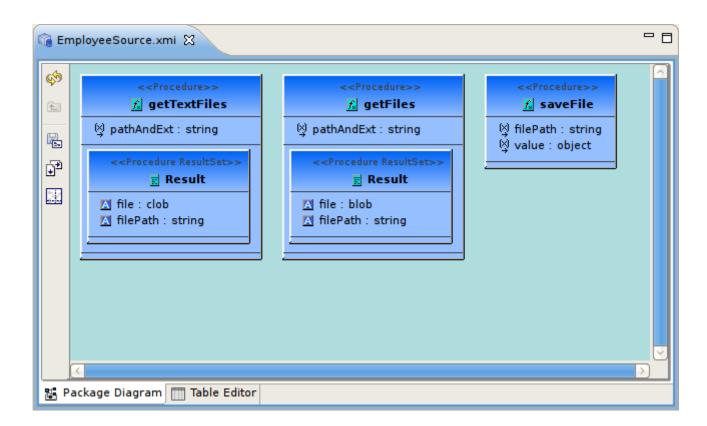


Figure 4.17. Generated Flat File Procedures

In addition, the view model will be opened in an editor and will show the generated view tables containing the completed SQL required to access the data in your flat file using the "getTextFiles" procedure above and the Teiid TEXTTABLE() function. The following figure is an example of a generated view table.

👘 Ei	mploy	eeViews.xmi	x												
69		1	VIEW			SOU	RCI	ES							
		A LastNam A FirstNam A FirstNam A MiddleNa A Empld : A Departm A AnnualS A Title : st	<base table=""/> > ployeeDataView ne : string(10) ne : string(10) ame : string(10) string(10) nent : string(10) alary : string(10) tring(10) none : string(10)	<		< <p ▲ g Emp ♀ pathAndd</p 	rocedu etTex loyeeS Ext : s ure Re Resu ob	ure>> ctFile Source string string ultSe	1						
₽	2	A Street : A City : str A State : s	string(10) ring(10)							_					
₽ €[1	Transf	ormation Diag	ram 🔲 Table Editor												
Tr(or 🤦	§ 📌 🐺	Cursor at (3, 6)		Sup	ports Update	₽₽	VE		極	S	A	A *		a.b.) Lipi
A.I	Mgrid, FROM (E) stNan	A.Street, A EC Employe ne string, Mi	A.FirstName, A.Midd City, A.State, A.Zip eeSource.getTextFile iddleName string, E Marld string, Street	oCod es('Er mplo	e mploj strir	yeeData.txt') ng, Departme) AS	f, TE) tring,	KTTAB Annu	BLE(f.f	ile Co lary s	⊃LUM tring	INS L , Title	astN strir	lam ng,

Figure 4.18. Generated Flat File View Table

4.4. Import From XML Data File Source

 Teiid supports XML Files as data sources. You can import from these data sources and create the metamodels required to query your data in minutes. Using the steps below you will define your flat file data source, configure your parsing paramaters for the xml data file, generate a source model containing the required Teiid procedure and create a view table containing the SQL defining the column data in your xml data file.

As with Designer's JDBC, Salesforce and WSDL importers, the XML File importer is based on utilizing a specific Data Tools Connection Profile.

The results of the importer will include a source model containing the getTextFiles() procedure or invokeHTTP() procedure which are both supported by Teiid.

The importer will also create a new view model containing a view table for your selected flat file source file. Within the view table will be generated SQL transformation containing the "getTextFiles()" procedure from your source model as well as the column definitions and parameters required for the Teiid XMLTABLE() function used to query the data file. You can also choose to update an existing view model instead of creating a new view model.

The XMLTABLE function uses XQuery to produce tabular ouptut. The XMLTABLE function is implicitly a nested table and may be correlated to preceeding FROM clause entries. XMLTABLE is part of the SQL/XML 2006 specification.

```
XMLTABLE([<NSP>,] xquery-expression [<PASSING>] [COLUMNS <COLUMN>, ... )]
AS name
```

COLUMN := name (FOR ORDINALITY | (datatype [DEFAULT expression] [PATH
string]))

Teiid Designer will construct the full SQL statement for each view table in the form:

```
SELECT
A.entryDate AS entryDate, A.internalAudit AS internalAudit
FROM
(EXEC CCC.getTextFiles('sample.xml')) AS f,
XMLTABLE(XMLNAMESPACES('http://www.kaptest.com/schema/1.0/party' AS
pty),
'/pty:students/student' PASSING XMLPARSE(DOCUMENT f.file)
COLUMNS entryDate FOR ORDINALITY, internalAudit string PATH '/
internalAudit') AS A
```

To import from your XML data file source follow the steps below.

• Step 1 - In Model Explorer choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

 Step 2 - Select the import option Teild Designer > File Source (XML) >> Source and View Model and click Next>

🔲 Import	×
Select	
Choose import source.	<u>r≚ŋ</u>
<u>S</u> elect an import source:	
type filter text	4
Plug-in Development	^
Run/Debug	
🕨 🧀 Team	
▼ 🔁 Teiid Designer	
DDL File >> Source or View Model	
File Source (Flat) >> Source and View Model	
File Source (XML) >> Source and View Model	
JDBC Database >> Source Model	
X Salesforce >> Source Model	
B WSDL File or URL >> Source Model	
B WSDL File or URL >> Web Service Model S XML Schemas	
	~
(?) < <u>Back</u> <u>Next</u> > Cancel Cancel	Finish

Figure 4.19. Import from XML File Source

 Step 3 - The next page of the wizard allows selection of the XML Import mode that specifies whether the XML file is local or remote. The description at the top describes what operations this wizard will perform. Select either the XML file on local file system or XML file via remote URL and click Next>

lmport F	From XML File Source				
XML Import File Options					
Select the desired XML Import Mode					
•					
Description					
This wizard automates the construction of the source procedure and view table transformation necessary to que from your XML file source. The view table transformation will contain a getTextFiles() (or invokeHTTP()) procedure utilize the XMLTABLE() function.					
The XML structure of your file will be presented to aid in	n selecting data elements to define your table column data				
Select XML Import Mode					
 XML file on local file system 					
○ XML file via remote URL					
?	< Back Next > Enish Ca				

Figure 4.20. XML Import File Options Page

 Step 4 - Select existing or previous connection profile from the drop-down selector or press New... button to launch the New Connection Profile dialog (See Eclispe Data Tools documentation) or Edit... to modify/change an existing connection profile prior to selection.

After selecting a **Connection Profile**, the XML data file from the connection profile will be displayed in the **Available Data Files** panel. Check the the data file you wish to process. The data from this file, along with your custom import options, will be used to construct a view table containing the required SQL transformation for retrieving your data and returning a result set.

Lastly enter or unique source model name in the **Source Model Definition** section at the bottom of the page or select an existing source model using the browse button. Note the **Model Status** section which will indicate the validity of the model name, whether the model

exists or not and whether the model already contains the getTextFiles() procedure. In this case, the source model nor the procedure will be generated.

When finished with this page, click Next>.

90			Import Fre	om XML File Sourc	:e			
XML Data	File Sou	arce Selection						
Press th	e "Next >	" button to continue.						
Data File	e Source							
Parts XM	1L File						•	New
XML Dat	a File							
Folder lo	cation:	/usr/share/teiid						
Data File	e Name							
🗹 parts	uxml							
Selected	Data File	: parts.xml						
Source		,						
Location	parts							
Name:	PartXML	Source						
	Model	Status						
	NEW M	ODEL: Source model [PartX with required getTextFiles() p						
0				- Dook	Nexts	-		
()				< <u>B</u> ack	<u>N</u> ext >	Enist	1	Ca

Figure 4.21. XML Data File Source Selection Page

• Step 5 - The primary purpose of this importer is to help you create a view table containing the transformation required to query the user-defined data file. This page presents a number of options you can use to customize the **Generated SQL Statement**, shown in the bottom panel. The to panel contains an XML tree view of your file contents and actions/buttons you can use to create column entries displayed in the middle, **Column Information** panel.

To create columns, select a root XML element and right-click select **Set as root path** action. This populates the root path value. Next, select columns in the tree that you wish to include on your query and select **Add selection as new column** button. You can also modify or

create custom columns, by using the **ADD**, **DELETE**, **UP**, **DOWN** to manage the column info in your SQL.

Note that the **Path** property value for a column is the selected element's path relative to the defined root path. If no root path is defined all paths are absolute. Each column entry requires a datatype and an optional default value. See the Teiid User's Guide for details on the XMLTABLE() function.

When finished with this page, click Next>.

9 O	Imp	ort From XML File Source			
XML Data File Import Options					
Press the "Next >" button to continu	ue.				
XML File parts.xml					
XML File Contents	-Column Info				
▼ e partssupplier	Root Path	/partssupplier/parts			
 ✓ e parts 			1	1	1
() id	Add	Column Name	For Ordinality		De
e name	Delete	🖾 id		string	
e color	Move Up	🖾 name		string	
		🖾 color		string	
weight	Move Down	🛛 weight		string	
▶ e parts					
▶ e parts					
e parts					
 e parts 					
▶ e parts		1			
Generated SQL Statement					
A.id AS id, A.name AS name, A.col	or AS color. A w	veight AS weight			
FROM					
(EXEC PartXMLSource.getTextFiles (DOCUMENT f.file) COLUMNS id strin	s('parts.xml')) A	S f, XMLTABLE('/partssupplier/p	arts' PASSING >	(MLPARSE	0
string PATH 'weight/text()') AS A	ig PATH @id, II	ane sung PATH hame/cext(),	COIOF SUILING PAR	In color/text	.(<i>)</i> , v
				I	6
?		< <u>B</u> ack <u>N</u> ext	>	nish	Ca

Figure 4.22. XML File Delimited Columns Options Page

• Step 6 - On the View Model Definition page, select the target folder location where your new view model will be created. You can also select an existing model for your new view tables. Note the Model Status section which will indicate the validity of the model name,

whether the model exists or not. Lastly, enter a unique, valid view table name. Press **Finish** to generate your models and finish the wizard.

9 O	Import From XML File Source
View Model Definitio	n
Press the "Finish" but	ton to finish.
Selected Data File: par	ts.xml
p.	
View Model Definiti	on
Location:	parts
	P
Name:	PartView
	Model Status
	NEW MODEL: New view tables will be created in a new view model [PartView] on FINISH.
New view table name:	PartsViewTable
	,
?	< <u>B</u> ack <u>Next</u> > <u>F</u> inish C

Figure 4.23. View Model Definition Page

4.5. Import From Salesforce

• You can create relational source models from your Salesforce connection using the steps below.



• Step 1 - In Model Explorer choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

- Step 2 Select the import option Teiid Designer > Salesforce >> Source Model and click Next>
- Step 3 Select existing or previous connection profile from the drop-down selector or press New... button to launch the New Connection Profile dialog (See Eclispe Data Tools documentation) or Edit... to modify/change an existing connection profile prior to selection. Note that the Connection Profile selection list will be populated with only Salesforce connection profiles.

ŧ	Create Relational Model from SalesForce Data Model	×
SalesForce (Validate the Sa	Credentials alesForce Connection Profile	(X)
Connection	n Profile	
Billing Sales	force New Ed	it
Properties		
URL:	<default></default>	
User Name:	joeBlow@bogus.ortg	
Password:	********	
Ping		
?	< <u>B</u> ack <u>N</u> ext > Cancel Finish	

Figure 4.24. Select Salesforce Credentials Dialog

 Step 4 - After selecting a Connection Profile, input password (if not provided). Press Next> to display the Salesforce Objects selection page.

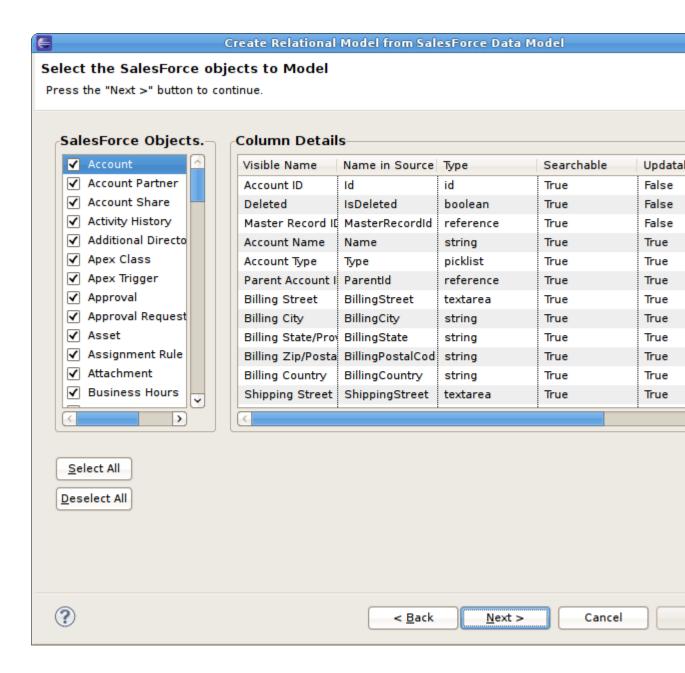


Figure 4.25. Select Salesforce Objects Dialog

 Step 5 - On the Target Model Selection page, specify the target folder location for your generated model, a unique model name and select desired import options. Press Next> (or Finish if enabled).

	Create Relatio	nal Model fro	n SalesForce D	ata Model	
arget Model	Selection				
Press the "Next	>" button to continue	or the "Finish" b	outton to finish.		
Select Target R	elational Model				
Model Name:	BillingSourceModel				Browse
Location:	SalesforceImportTest				Browse
Select Import C	ptions				
Model aud	it fields.				<u>^</u>
Selecting this	option will cause the i	importer to mod	lel the 'Audit Field	s' for each SalesFo	orce object.
(Created By,	Created Date, Last Mo	dified By, Last N	lodified Date, Sys	tem Modification T	imestamp)
🗌 Do not gat	her Cardinalities				
Selecting this	option will stop the in	porter from cal	culating and setti	ng the cardinalities	metadata
For large sale	sforce applications thi	is can become a	a long running op	eration.	
🗌 Gather Co	lumn Distinct Value Co	unt			
Selecting this	option will cause the i	importer to calc	ulate and set the	distinct value coun	t metadata
This will scan	the data in each field	individually and	can be a very lor	ng running operatio	in.
🗌 Set name	to SalesForce label.				
Selecting this	option will cause the i	importer to set	the name metada	atain the genera	ted model 1
By default the	importer uses the int	ernal data nam	e because SalesF	orce labels are oft	en invalid f
🗌 Create a p	rocedure for the GetU	pdated operatio	in.		
Selecting this	option will cause the i	importer to crea	te a procedure fo	or the getUpdated o	operation ir
🗌 Create a p	rocedure for the GetD	eleted operation	n.		
Selecting this	option will cause the i	importer to crea	te a procedure fo	or the getDeleted o	peration in
					>
?		< <u>B</u> ack	<u>N</u> ext >	Cancel	<u>F</u> inish

Figure 4.26. Target Model Selection Dialog

• Step 5a - If you are updating an existing relational model, the next page will be **Review** Model Updates page. Any differences. Press Finish to create your models and tables.

Create Relational Model from SalesForce Data Model 🛛 🕅
Review Model Updates
Review the changes that will be applied to your model to bring it up to date with the source metadata. Uncheck any changes you do not wish to apply.
Only In Old Model (OLD): 0 Changes (스): 0 Only In New Model (NEW): 1 · · · · · · · · · · · · · · · · ·
▼ ✓ file:/TestSF/1314365845645_temp.xmi
✓ NEW salesforce
Selection Details
salesforce - this selection is an Addition

Figure 4.27. Review Model Updates Dialog

When finished, the new or changed relational model's package diagram will be displayed showing your new tables.

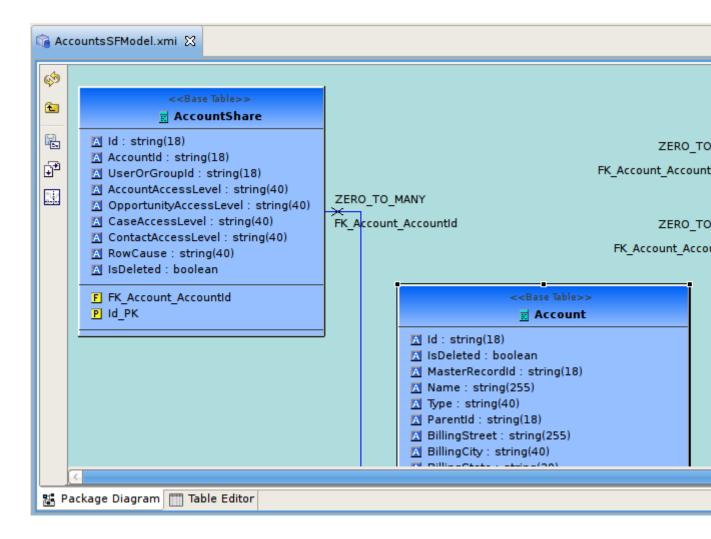


Figure 4.28. New Saleforce Tables Diagram

4.6. Import Metadata From Text File

- The Teiid Designer provides various import options for parsing comma delimited text file metadata into models. This is accomplished via the Import > Teiid Designer > Designer Text File >> Source or View Models option.
 - Step 1 In Designer choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

- Step 2 Select the import option Teild Designer > Designer Text File >> Source or View Models and click Next>
- Step 3 Select an import type via the drop-down menu shown below.

Select Impo	rt Type of text file import desired.	
Select the type		· \
Import Type:	Relational Model (XML Format)	~
Purpose: Imp	Relational Model (XML Format)	
relational sou	Relational Tables (CSV Format)	
	Relational Virtual Tables (CSV Format)	
Sample File Fo	Datatypes (CSV Format)	

Figure 4.29. Import Wizard

- These steps are required for each type are defined below:
 - Section 4.6.1, "Import Relational Model (XML Format)"
 - Section 4.6.2, "Import Relational Tables (CSV Format)"
 - Section 4.6.3, "Import Relational View Tables (CSV Format)"
 - Section 4.6.4, "Import Datatypes (CSV Format)"

4.6.1. Import Relational Model (XML Format)

А

- To create relational tables from imported xml text file metadata:
 - Perform Steps 1 through 3 (above) and select the Relational Model (XML Format) import type, then click Next >

🗧 🛛 Import Metadata From Text File 🛛 🕅
Select Import Type
Select the type of text file import desired.
Import Type: Relational Model (XML Format)
Import Type: Relational Model (XML Format)
Purpose: Imports relational tables, views, procedures and indexes from an XML File into a relational source model.
Sample File Format:
<pre><relational-model> </relational-model></pre>

Figure 4.30. Select Import Type - Relational Model (XML Format)

• Perform **Steps 4** - On the next page, select the XML file on your local file system via the **Browse...** button. Select a target model to which the imported relational objects will be added via the second **Browse...** button. The dialog allows selecting an existing relational model or

creating a new model. Note the contents of your selected XML file will be display in the File Contents viewer. Click **Finish** to create your new

	Import Metadata From Text File	6
Select Source Text	File and Target Relational Model.	(Ph
Select Metadata File to I	mport and Destination Model or model folder	TAT
	-	¥-
Select Source Text File	≥r/Release_7.6/planning/SampleRelationalXML.xml	B <u>r</u> owse
-		
Target Location Produc	tsHomXMLIext.xmi	Browse
File Contents		
xml version="1.0" er</td <td>ncoding="UTF-8"?></td> <td><u></u></td>	ncoding="UTF-8"?>	<u></u>
<relational-model nam<="" td=""><td>e="myRelationalModel" xmlns="http://www.teiid.org"</td><td></td></relational-model>	e="myRelationalModel" xmlns="http://www.teiid.org"	
xsi:schemaLocation="h	ttp://www.teiid.org file:///home/blafond/Documents/Teii	d_Designer/Rele
	oductData" nameInSource="dbo.products.ProductData'	
	="Basic stocks or bond data table second line &	#xa; &#x</td></tr><tr><td>cardinality=</td><td></td><td></td></tr><tr><td></td><td>="true" supportsUpdate="true" system="true"></td><td></td></tr><tr><td></td><td>me="ProductID" description="Unique ID for this product</td><td></td></tr><tr><td></td><td>nSource="dbo.products.ProductData.ProductID" dataty ="11" nullable="NO NULLS"></td><td>pe= varchar</td></tr><tr><td></column></td><td></td><td></td></tr><tr><td></td><td>me="ProductName" nameInSource="dbo.products.Pro</td><td>ductData.Produ</td></tr><tr><td></td><td>pe="varchar" length="60" nullable="NULLABLE"></td><td></td></tr><tr><td></column></td><td>-</td><td></td></tr><tr><td><column na</td><td>me="ProductType" nameInSource="dbo.products.Prod</td><td>uctData.Product</td></tr><tr><td>dataty</td><td>pe="varchar" length="15" nullable="NULLABLE"></td><td></td></tr><tr><td></column></td><td></td><td></td></tr><tr><td><column na</td><td>me="ISSUER" nameInSource="dbo_products_ProductDa</td><td></td></tr><tr><td></td><td></td><td>></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr><tr><td>?</td><td>< Back Next > Cancel</td><td>Finish</td></tr><tr><td>U</td><td></td><td></td></tr></tbody></table>

Figure 4.31. Select Source Text File and Target Relational Model Page

If the target model contains named children (tables, views, procedures) that conflict with the objects being imported, a dialog will be displayed giving you options on how to proceed including: replacing specific existing objects, creating new same-named objects or cancel import entirely.

E Duplicate Objects Exist	8
Options	
 Replace Existing Objects (See list below) 	
 Create New Objects Anyway (results in validation errors) 	
 Cancel Import 	
Duplicate Objects	5
getProductInfo	
✓ ProductData	
ProductIDIndex	
✓ ProductSymbols	
ProductView	
	-1
Select All Deselect All	
Cancel OK	

Figure 4.32. Duplicate Objects Dialog

4.6.2. Import Relational Tables (CSV Format)

- To create relational tables from imported text file metadata:
 - Perform Steps 1 through 3 (above) and select the Relational Tables (CSV Format) import type, then click Next >

B Import Metadata From Text File
Select Import Type
Select the type of text file import desired.
Import Type: Relational Tables
 Purpose: Imports relational schema, catalogs, tables, columns, and indexes from a CSV text file into a relation of the schema, Catalog, and Table data is expected to be of the form: >> TYPE (i.e. SCHEMA, CATALOG, or TABLE), Name, "Description" (Optional), Location (Optional) >> Locations are of the form: ProjectName/FolderName/ModelName/SchemaName. >> If the project, folder, model or schema/catalog containers do not exist, they will be created 2) The Column data is expected to be in the form: >> COLUMN, ColumnName, JDBCType, Length, "Description" 3) Column data rows for each table must appear immediately following the table data row. 4) The Index data is expected to be of the form: >> INDEX, IndexName, Type, Uniqueness, Tablespace, Column 5) Index data rows must appear immediately following the table column data rows. A sample of typical input data is shown below:
CATALOG, Catalog 1, "Catalog 1 Description", Project 1/MyModel_1
SCHEMA, Schema_1, "Schema_1 Description", Project_1/MyModel_1/Catalog_1

Figure 4.33. Select Import Type - Relational Tables (CSV Format)

• **Step 4** - In the next page, you'll need to provide a source text file containing the metadata formatted to the specifications on the previous page.

Import Metadata From Text File	×
Select Source Text File and Target Relational Model. Select CSV File to Import and Destination Model or model folder	
Select Source Text File	B <u>r</u> owse
Target Location CustomerInfo.xmi	Browse
File Contents	
Options	
Use default datatype: string(255)	
(?) < <u>Back</u> <u>Next</u> > Cancel	<u>F</u> inish

Figure 4.34. Select Source Text File and Target Relational Model

- **Step 5** Select an existing relational model as the target location for your new relational components using the **Browse...** button to open the Relational Model Selector Dialog. Select a relational model from your workspace or specify a unique name to create a new model.
- Step 6 Select any additional options and choose Finish.

4.6.3. Import Relational View Tables (CSV Format)

- To create relational virtual tables from imported text file metadata:
 - Perform Steps 1 through 3 (above) and select the Relational Virtual Tables (CSV Format) import type, then click Next >

Import Metadata From Text File	
Select Import Type	
Select the type of text file import desired.	
Import Type: Relational Virtual Tables	
Purpose: Imports virtual relational tables and procedures from a CSV text file into a relational virtual model. Table data is expected to be of the form: >> TableName, "SQL Statement", Description A sample of typical input data is shown below:	
Sample File Format:	
VTable1, "SELECT * FROM PMod1.PTab1, PMod1.PTable2", "Descrip, no embedded quotes"	
VTable2, "SELECT * FROM PMod1.PTab2", "Descrip 2 with quotes ""xxx"""	
VTable3, "SELECT * FROM PMod1.PTab3", "Descrip 3"	

Figure 4.35. Select Import Type - Relational Virtual Tables (CSV Format)

• **Step 4** - In the next page, you'll need to provide a source text file containing the metadata formatted to the specifications on the previous page.

9	Import Metadata From Text File	×
	Text File and Target Virtual Relational Model	
Select Source Te	ext File Brows	e
Target Location	Brows	e
File Contents		
?	< Back Next > Cancel Finis	sh

Figure 4.36. Select Source Text File and Target Virtual Relational Model

- Step 5 Select an existing relational virtual model as the target location for your new model components using the **Browse...** button to open the Virtual Model Selector Dialog. Select a virtual relational model from your workspace or specify a unique name to create a new model.
- Step 6 Select Finish.

4.6.4. Import Datatypes (CSV Format)

- To create datatypes from imported text file metadata:
 - Perform Steps 1 through 3 (above) and select the Datatypes (CSV Format) import type, then click Next >

6	Import Metadata From Text File
	Select Import Type Select the type of text file import desired.
	Import Type: Datatypes
	Purpose: Imports datatypes from a CSV text file into a datatype model. Datatype data is expected to be of the form: >> TYPE or ENUM, Name, "Description", baseTypeName, Length, MinLength, MaxLength, MinBou MaxInclusive, TotalDigits, FractionDigits. Blank or zero values are ignored.
	A sample of typical input data is shown below: Sample File Format:
	TYPETAG, Name, Description, BaseType, Length, MinLength, MaxLength, MinBound, MinInclusive, MaxBound, MaxInclusi TYPE, Atomic1, "Description for Type1", BType, 5, 1, 10, 1, 2, 1, 2, 5, 2 TYPE, Atomic2, "Description for Type2", BType, 5, 1, 10, 1, 2, 1, 2, 5, 2

Figure 4.37. Select Import Type - Datatypes (CSV Format)

• **Step 4** - In the next page, you'll need to provide a source text file containing the metadata formatted to the specifications on the previous page.

٠	Import Met	adata From Te	xt File	×
	Text File and Ta Import and Destina			
Select Source Te	ext File		~ (B <u>r</u> owse
Target Location				Browse
File Contents				
?	< <u>B</u> ack	<u>N</u> ext >	Cancel	Enish

Figure 4.38. Select Source Text File and Datatypes Model

- Step 5 Select an existing datatype model as the target location for your new model components using the **Browse...** button to open the Datatypes Model Selector Dialog. Select a datatypes model from your workspace or specify a unique name to create a new model.
- Step 6 Select any additional options and choose Finish.

4.7. Import WSDL into Relational Models

You can turn your WSDL file (local or URL) into a queryable relational procedures that represent your desired request and response web service structure defined through your WSDL's schema definition. This importer is accessed by launching Eclipse's "Import..." action and selecting the "Teiid Designer > WSDL File or URL >> Source and View Model (SOAP)" option. *Web Services Connection Profile* defined by a WSDL file in your workspace or defined by a URL. Designer will interpret the WSDL, locate any associated or dependent XML schema files, generate a physical model to invoke the service, and generate virtual models containg procedures to build and parse the XML declared as the service messages.

• To create relational models from WSDL use the steps below.

- Step 1 In Model Explorer choose the File > Import action in the toolbar or select a project, folder or model in the tree and choose Import...
- Step 2 Select the import option Teild Designer > WSDL File or URL >> Source and View Model (SOAP) and click Next>
- **Step 3** On the next page select an existing Web Service Connection Profile from the list, or click the **New** Button to create a new profile.

te Relational Model from Web Serv	ice		
SDL Operations Definition			
xt >" button to continue.			2
Profile			
fo			▼ New Edi
or Location:			
.oorsprong.org/websamples.coun	tryinfo/CountryInfoServio	ce.wso?WSDL	Validate WS
CountryInfoServiceSoap			
SOAP11	Service Mode	PAYLOAD	
lesired WSDL Operations	,		
Operation	Code		
All 🖸 📄 CountryName			
CurrencyName			
🔽 📄 FullCountryInfo			
📄 🕞 FullCountryInfoA	llCountries		
LanguagelSOCod	le		
LanguageName			
Details:			
	oop		
		tryinfo}CountryInfoServiceSo	oapBinding.CountryNam
message: CountryNameSoanRe	Pullest		
		Back Next >	Cancel Finis
	SDL Operations Definition xt >" button to continue. Profile fo or Location: All Operation SOAP11 desired WSDL Operations All Operation All Operation CountryInfoServiceSoap SOAP11 desired WSDL Operations All Operation CountryIntPhone CountryIntPhone CountryIntPhone CountryIntPhone CountryIntPhone CountryIntPhone CountryInto CountryInt	xt >" button to continue. Profile fo or Location: coorsprong.org/websamples.countryinfo/CountryInfoService CountryInfoServiceSoap SOAP11 Service Mode desired WSDL Operations All Operation Operation All Operation CountryInfPhoneCode All Operation CountryInfoName CountryInfoName Details: CountryInfoServiceSoapBinding CountryInfoServiceSoap e: CountryInfoServiceSoap e: CountryInfoServiceSoapBinding CountryInfoServiceSoap e: CountryInfoServiceSoapBinding CountryInfoServiceSoapBi	SDL Operations Definition xt >" button to continue. Profile fo or Location: coorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL CountryInfoServiceSoap SOAP11 Service Mode PAYLOAD desired WSDL Operations All Operation Image: CountryInfoName Image: CountryInfoAllCountries Image: CountryInfoServiceSoapBinding countryInfoServiceSoapBinding

Figure 4.39. WSDL Source Selection

• Step 4 - Select individual Web Service Operations to model. The default behavior of this page selects all available operations in the tree. Operations can be de-selected if they are not being modeled. The Selection Details panel displays static information about the operation such as the names of the input and output messages, and faults thrown by the operation.

Click Next >

• Step 5 - The next page entitled Model Definiton requires both a model location (i.e. folder or project) and a valid model name for both source and view models. Use the **Browse...** button to select existing folders or models. Click **Next>** when all the information is defined.

	reate Relational Model from Web Service
odels Defi	inition
ll inputs (OK. Click 'Next>' to define custom procedures.
Source M	odel Definition
Location	StockService
Name	CountryInfoService.xmi
Status-	
Source procedu	model CountryInfoService.xmi does not exist and will be created and contain the required invoke() web service ure.
View Mod	lel Definition
Location	StockService
Name	CountryInfoServiceView.xmi
-Status -	
	adal Country of a San iso View you door not exist and will be created and contain your concreted presedures
	odel CountryInfoServiceView.xmi does not exist and will be created and contain your generated procedures.
View mo	odel CountryInfoServiceView.xmi does not exist and will be created and contain your generated procedures.
View mo	
View mo Procedur © User-	e Generation Options
View mo Procedur	specified Procedures
View mo Procedur () User- Define us () Lega	e Generation Options specified Procedures ser-specified request and response procedures from your WSDL schema elements.
View mo Procedur () User- Define us () Lega	e Generation Options specified Procedures ser-specified request and response procedures from your WSDL schema elements. cy Procedures
View mo Procedur () User- Define us () Lega	e Generation Options specified Procedures ser-specified request and response procedures from your WSDL schema elements. cy Procedures
View mo Procedur () User- Define us () Lega	e Generation Options specified Procedures ser-specified request and response procedures from your WSDL schema elements. cy Procedures

Figure 4.40. WSDL Source Selection

• **Step 6** - This wizard generates both request and response procedures that are used in the queryable wrapped procedure. The next page, Procedure Definition, provides the means to define the details of your request and response structures.

In the Request tab, select and double-click the schema elements you wish to be input parameters for your request. These will be added to the Element Info panel and the resulting generated SQL statement will be updated to reflect the new element. Note the BODY and HEADER tabs which exist on both the Request and Response tabs. If the selected service mode for this procedure is set to MESSAGE, the HEADER tab will be enabled and allow you to define the SOAP header variables utilizing the same schema tree.

Select the Response tab and create the response procedures result set columns in the same way.

Repeat this process for all operations by changing the selection target operation via the **Operations** selector at the top.

Operations	
FullCountryInfo	
Overwrite existing procedures for this operat	tion
Request Response Wrapper Procedure	
Generated Procedure Name FullCountryInfo_re	equest
P00% (UE4055)	
BODY HEADER	
▼ 🖫 FullCountryInfo . type	
▼ ■=== sequence	Add Name
▶ e sCountryISOCode	Delete
	Up
	Down
Generated SQL Statement	
CREATE VIRTUAL PROCEDURE	
BEGIN SELECT	
	.NAMESPACES(DEFAULT 'http://www.oorsprong.org/websamples.countr CountryInfoServiceView.FullCountryInfo_request.sCountryISOCode))
xml_out; END	

Figure 4.41. Procedure Definition Page

• **Step 7** - Click **Finish**. After generation the new models can be found in the specified location in your workspace.

In the **Model Explorer** you can see the importer created the following a single **physical model** containing a single procedure called invoke. This model and procedure correspond to the single port declared in the WSDL.

A single **view model** was also created containing your new procedures named after the operations declared in the WSDL. For each operation a wrapper procedure was created which can be previewed in Designer. Below is an example dependency diagram showing the sources for the wrapper procedure as request, response procedures and the invoke() source procedure.

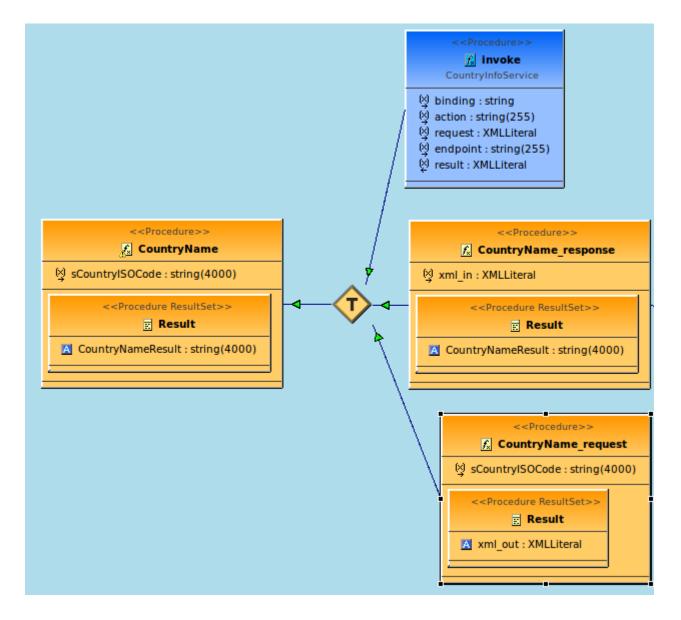


Figure 4.42. Example Web Services Wrapper Procedure

4.7.1. Circular References in WSDL Schemas

It is possible for a WSDL schema to either contain a very deep set of XML type references or indeed for such references to be circular. This is legal in the WSDL schema but can make

processing the schema in Designer difficult. If left unchecked such circular references can result in a JVM **StackOverFlow** exception and exiting of the application.

To mitigate this possibility a depth limit of 750 references has been introduced. Should the depth exceed this limit then a warning is displayed and further processing of that fragment of the schema will end. It may be the case that the reference in question is not circular but just very deep so in such a case it is possible to increase the depth limit by setting the JVM property **WsdlSchemaHandlerRecursiveDepth** to a larger value, eg. -D WsdlSchemaHandlerRecursiveDepth=800. This should only be used with caution as on some systems it is possible the JVM throws a **StackOverFlow** exception before the new depth limit is reached.

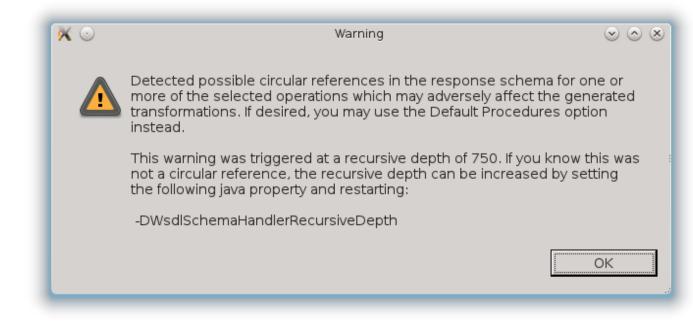


Figure 4.43. Warning message displayed if depth limit has been reached

4.8. Import WSDL Into Web Service

You can create a **Web Service** model by selecting a **WSDL** file in your workspace, importing **WSDL** files from the file system or by defining a URL. The Teiid Designer will interpret the **WSDL**, locate any associated or dependent **XML Schema** files, generate an **XML View** of the schema components and create a **Web Service** model representing the interfaces and operations defined in the **WSDL**.

- There are three options for selecting the WSDL for your Web Service generation
 - Workspace Location
 - File System Location

• URL

Detailed steps for each of these options is described below, as well as a description of how the wizard handles **WSDL** errors.

4.8.1. Import WSDL From Workspace Location

- You can create a Web Service model by selecting a WSDL file from your workspace.
 - Step 1 Choose the File Import choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

- Step 2 Select the import option Teild Designer > WSDL File or URL >> Web Service Model option shown below and click Next>
- Step 3 Input a valid name for your Web Service model and select the Workspace... button. Locate your workspace WSDL file in the selection dialog and click OK>. Click Next> to continue.

Screate Web Service from WSDL File	×
WSDL File Selection Press the "Next >" button to continue.	8
Web Service Model Name OnHandWS	
WSDL Files Ider/Test In JBDS/JBDS_H45/workspace/WSDLImportProject/O	nHand.wsdl
	>
Cancel Can	Ēnish

Figure 4.44. WSDL File Selection Dialog

	WSDL	. File Se	lection	×
Choose a	WSDL file to ad	ld:		
🗢 📑 W	SDLImportProjec	t		
	OnHand.wsdl			
		_		
Ø		C	ancel	ок

Figure 4.45. WSDL File Workspace Selection Dialog

Note

i

- If no WSDL is selected or specified then the importer will only create an empty Web Service model. No XML Schema or XML View models will be generated.
- Any referenced files (**WSDL**s or schemas) must either be embedded in the **WSDL** file or exist on your file system.
- Step 4 The next page is titled Namespace Resolution. This page identifies successful and errant WSDL namespace resolution. The main WSDL document will essentially always be resolved, since the workspace file chooser is used to obtain the path. Problems will occur when the main WSDL file imports other WSDL files that cannot be resolved. If no errors, select Next to proceed, or Finish (if enabled) to complete with default options.

Name	espaces		
400	Namespace	Path	
	http://dladb07/	/home/blafond/Test Designer Folder/example files/wsdls/OnHam	d.wsdl

Figure 4.46. Namespace Resolution Dialog

• Step 5 - The next page WSDL Operations Selection allows customizing the resulting content of your Web Service model by selecting/deselecting various operations and interfaces in the following dialog.

9	Create Relat	tional Model f	rom Web Service		δ
WSDL Operations	Selection				
Press the "Finish" but	on to finish.				S
Select the desir	ed WSDL Operatio	ons			
⊽ 🗹 📄 Share	M_Real_Time_Data_Des _BSM_Real_Time_Data_ are_BSM_Real_Time_Da	Design_Server	RealTimeData_port	Port0	Select All
	OnHand			, and the second s	
Selection Details: OnHand [Operation]					
id: input message: output message: fault names:	{http://dladb07/ _OnHandInput OnHandOutput none	}Share_BSM_R	eal_Time_Data_Des	ign_Server_Re	alTimeData_p
<u></u>					>
?	(< <u>B</u> ack	Next >	Cancel	<u>F</u> inish

Figure 4.47. Namespace Resolution Dialog

 Step 6 - The next page is titled Schema Workspace Location Selection. This page lists all schemas imported by the WSDL (along with any dependent schemas referenced within schemas) as well as schemas embedded in the WSDL and indicates whether or not they are resolvable. All resolved schemas will be created in a separate file and added to the workspace. The editor panel allows you to change the default file name of the new schema file(s).

If no errors, select Next to proceed, or Finish to complete with default option

9		Create Web Service from WSDL File	
Sche	ma Workspace Location 9	Selection	
All wo	orkspace schema locations are v	alid. Click "Next" to continue.	
Sche	ema Location Information		
40	Namespace		Source File
	http://localhost/Share_BSM_Re	al_Time_Data/Design_Server/RealTimeData_port/OnHand	/home/blafond/T
E E	ditor - http://localhost/Share_BS	M_Real_Time_Data/Design_Server/RealTimeData_port/On	Hand
	Folder:	/WSDLImportProject	
	Name (without file extension):	OnHand	
?		< <u>B</u> ack <u>N</u> ext >	Cancel

Figure 4.48. Namespace Resolution Dialog

• Step 7 - The last page titled XML Model Generation allows you to change the name of the XML View model if the Generate virtual XML document model is checked. Input desired name or use the default name provide. Select Finish to complete.

8	Create Web Service from WSDL File 🛛 🕅		
XML Mode	XML Model Generation		
Ready to gen	erate the Web service model		
🗹 Generate	virtual XML document model		
Location:	/WSDLImportProject 2		
XML Model:	OnHandWSResponses		
?	< Back Next > Cancel Finish		

Figure 4.49. Namespace Resolution Dialog

In order to successfully generate Web Services from WSDL, the WSDL must be error free. WSDL validation is performed during *Step 3* above. If errors do exist, a error summary dialog will be displayed (shown below) and you will not be able to *Finish* the wizard until the WSDL problems are fixed or you re-import and select a valid WSDL file.

-	Create Web Service from WSDL File
	Validation Problems ② One or more selected WSDL files have errors. Please correct or remove the file(s)to proceed.
	WSDL Validation Panel
	▲ message wsdl file
	😢 cvc-elt.1: Cannot find the declaration of element 'wsdl:definitions'. http://terraservice.net/terraservice2.asm
	< Back Next > Cancel

Figure 4.50. WSDL Validation Problems Dialog

4.8.2. Import WSDL From File System Location

- You can create a Web Service model by selecting a WSDL file from your local file system.
 - Step 1 Choose the File Import choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

- Step 2 Select the import option Teild Designer > WSDL File or URL >> Web Service Model and click Next>
- Step 3 Input a valid name for your Web Service model and select the File System... button. Locate your file system WSDL file in the selection dialog and click OK>.

Create Web Service from WSDL File			×
WSDL File Selection			
Press the "Next >" button to continue.			
Web Service Model Name OnHandWS Workspace File System			
WSDL Files		×	
/home/blafond/Test Designer Folder/example files/wsdls/OnHand.v	vsdl		
? < <u>Back Next</u> > Cancel)	inish	

Figure 4.51. WSDL File Selection Dialog

Note

i

- If no WSDL is selected or specified then the importer will only create an empty Web Service model. No XML Schema or XML View models will be generated.
- Any referenced files (**WSDL**s or schemas) must either be embedded in the **WSDL** file or exist on your file system.
- Step 4 The next page is titled Namespace Resolution. This page identifies successful and errant WSDL namespace resolution. The main WSDL document will essentially always be resolved, since the workspace file chooser is used to obtain the path. Problems will occur when the main WSDL file imports other WSDL files that cannot be resolved. If no errors, select Next to proceed, or Finish (if enabled) to complete with default options.

lam	espaces		a.
400	Namespace	Path	
	http://dladb07/	/home/blafond/Test Designer Folder/example files/wsdls/OnHand.ws	dl

Figure 4.52. Namespace Resolution Dialog

• Step 5 - The next page WSDL Operations Selection allows customizing the resulting content of your Web Service model by selecting/deselecting various operations and interfaces in the following dialog.

8	Create Relational Model from Web S	ervice	2
WSDL Operations	Selection		
Press the "Finish" but	on to finish.		Z
Select the desir	ed WSDL Operations]
▽ 🖌 📄 Share	M_Real_Time_Data_Design_Server_RealTimeData_ BSM_Real_Time_Data_Design_Server_RealTimeDa are_BSM_Real_Time_Data_Design_Server_RealTime	ata_portPort0	Select All
	OnHand		
< <u> </u>		Þ	
Selection Details:			
OnHand [Operation] id: input message: output message: fault names:	{http://dladb07/}Share_BSM_Real_Time_Da _OnHandInput OnHandOutput none	ıta_Design_Server_Real1	TimeData_p
Æ			>
-			
(?)	< <u>B</u> ack <u>N</u> ext >	Cancel	<u>F</u> inish

Figure 4.53. Namespace Resolution Dialog

• Step 6 - The next page is titled Schema Workspace Location Selection. This page lists all schemas imported by the WSDL (along with any dependent schemas referenced within schemas) as well as schemas embedded in the WSDL and indicates whether or not they are resolvable. All resolved schemas will be created in a separate file and added to the workspace. The editor panel allows you to change the default file name of the new schema file(s).

If no errors, select Next to proceed, or Finish to complete with default option

-			
۲		Create Web Service from WSDL File	
Sche	ma Workspace Location 9	Selection	
	rkspace schema locations are v		
All WO		and, check Next to continue.	
Sche	ma Location Information		
40	Namespace		Source File
	http://localhost/Share_BSM_Re	al_Time_Data/Design_Server/RealTimeData_port/OnHand	/home/blafond/T
	ditar http://lacalhaat/Shara RS	M_Real_Time_Data/Design_Server/RealTimeData_port/On	Hand
	altor - http://localnost/share_b3		nanu
	Folder:	WSDLImportProject	
	Name (without file extension):	OnHand	
0		< <u>B</u> ack <u>N</u> ext >	Cancel

Figure 4.54. Namespace Resolution Dialog

• Step 7 - The last page titled XML Model Generation allows you to change the name of the XML View model if the Generate virtual XML document model is checked. Input desired name or use the default name provide. Select Finish to complete.

8	Create Web Service from WSDL File 🛛 🕅		
XML Mode	XML Model Generation		
Ready to gen	erate the Web service model		
🗹 Generate	virtual XML document model		
Location:	/WSDLImportProject 2		
XML Model:	OnHandWSResponses		
?	< Back Next > Cancel Finish		

Figure 4.55. Namespace Resolution Dialog

In order to successfully generate Web Services from WSDL, the WSDL must be error free. WSDL validation is performed during *Step 3* above. If errors do exist, a error summary dialog will be displayed (shown below) and you will not be able to *Finish* the wizard until the WSDL problems are fixed or you re-import and select a valid WSDL file.

Screate Web Servic	e from WSDL File
Validation Problems One or more selected WSDL files have errors. Please correct	or remove the file(s)to proceed.
WSDL Validation Panel	
▲ message	wsdl file
🔕 cvc-elt.1: Cannot find the declaration of element 'wsdl:def	initions'. http://terraservice.net/terraservice2.asm
?	< <u>B</u> ack <u>N</u> ext > Cancel

Figure 4.56. WSDL Validation Problems Dialog

4.8.3. Import WSDL From URL

- You can create a Web Service model by selecting a WSDL file based on a URL.
 - Step 1 Choose the File Import choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

- Step 2 Select the import option Teiid Designer > WSDL File or URL >> Web Service Model and click Next>
- Step 3 Input a valid name for your Web Service model and select the URL... button.
 - Enter a valid WSDL URL. If the URL cannot be validated then an error will be displayed and the **OK>** button disabled.
 - If the WSDL is protected by basic HTTP authentication then this option should be selected and the appropriate username and password entered.
 - Click **OK>** to continue.

Click Next> to continue.

9	V	VSDL URL	×
Enter WSDL Url:	http://terrase	rvice.net/terraservice2.asmx?wsdl)
			1
	Security Type	HTTPBasic [▼	
	User Name	user	
	Password	••••	
		Cancel OK	
			·

Figure 4.57. WSDL URL Dialog

į	 Note If no WSDL is selected or specified then the importer will only create an empty Web Service model. No XML Schema or XML View models will be generated.
	 generated. Any referenced files (WSDLs or schemas) must either be embedded in the WSDL file or exist on your file system.
Step 4 - ⊤	he next page is titled Namespace Resolution. This page identifies successful

 Step 4 - The next page is titled Namespace Resolution. This page identifies successful and errant WSDL namespace resolution. The main WSDL document will essentially always be resolved, since the workspace file chooser is used to obtain the path. Problems will occur when the main WSDL file imports other WSDL files that cannot be resolved. If no errors, select Next to proceed, or Finish (if enabled) to complete with default options.

Create Web Service from WSDL File	×
WSDL File Selection Press the "Next >" button to continue.	8
Web Service Model Name TerraServiceWS	
Workspace	
WSDL Files	B X B
http://terraservice.net/terraservice2.asmx?wsdl	
? < <u>B</u> ack <u>Next</u> > Cancel	Enish

Figure 4.58. Namespace Resolution Dialog

• Step 5 - The next page WSDL Operations Selection allows customizing the resulting content of your Web Service model by selecting/deselecting various operations and interfaces in the following dialog.

	Create We	eb Service from WSDL File
ame	espace Resolution	
ll nar vorks	mespaces are resolved. Selec pace locations.	ct "Next" to modify the created schemas
Name	espaces	
4	Namespace	Path
		http://terraservice.net/terraservice2.asmx?wsdl

Figure 4.59. Namespace Resolution Dialog

 Step 6 - The next page is titled Schema Workspace Location Selection. This page lists all schemas imported by the WSDL (along with any dependent schemas referenced within schemas) as well as schemas embedded in the WSDL and indicates whether or not they are resolvable. All resolved schemas will be created in a separate file and added to the workspace. The editor panel allows you to change the default file name of the new schema file(s).

If no errors, select Next to proceed, or Finish to complete with default option

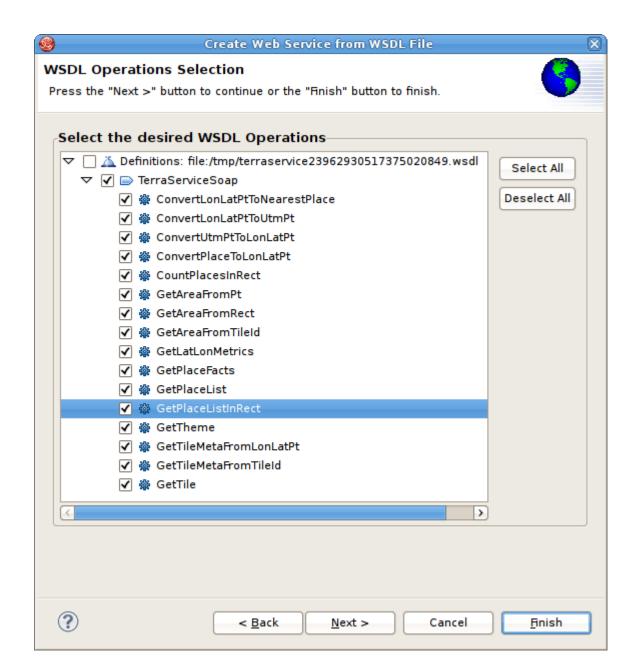


Figure 4.60. Namespace Resolution Dialog

• Step 7 - The last page titled XML Model Generation allows you to change the name of the XML View model if the Generate virtual XML document model is checked. Input desired name or use the default name provide. Select Finish to complete.

۲	B Create Web Service from WSDL File			
Sche	ma Workspace Location	Selection		
All wo	orkspace schema locations are	valid. Click "Next"	to continue.	1
Sche	ema Location Information			
-	Namespace	Source File	Target Path	
	http://terraservice-usa.com/	/tmp/terraservice2	WSDLImportProject/terrase	rvice23962930517375020849
ſ∦ E	ditor - http://terraservice-usa.c	com/		
	Folder:	/WSDLImportPro	ject	
	Name (without file extension)	terraservice23962930517375020849		
0		ſ	e Pack North	Cancel
0		l	< <u>B</u> ack <u>N</u> ext >	Cancel <u>Fin</u>

Figure 4.61. Namespace Resolution Dialog

In order to successfully generate Web Services from WSDL, the WSDL must be error free. WSDL validation is performed during *Step 3* above. If errors do exist, a error summary dialog will be displayed (shown below) and you will not be able to *Finish* the wizard until the WSDL problems are fixed or you re-import and select a valid WSDL file.

Screate We	b Service from WSDL File
Validation Problems One or more selected WSDL files have errors. Please	correct or remove the file(s)to proceed.
WSDL Validation Panel	
▲ message	wsdl file
😣 cvc-elt.1: Cannot find the declaration of element	'wsdl:definitions'. http://terraservice.net/terraservice2.asm
?	< <u>B</u> ack <u>N</u> ext > Cancel

Figure 4.62. WSDL Validation Problems Dialog

4.9. XSD Schema File

- You can import XML Schema file (XSD) files using the steps below.
 - Step 1 In Model Explorer choose the File > Import action

in the toolbar or select a project, folder or model in the tree and choose Import...

- Step 2 Select the import option Metadata Modeling > XSD Schemas and click Next>
- Step 3 Select either Import XSD Schemas from file system or Import XSD Schemas via URL and click Next >
- Step 4a If importing from file system, the Import XSD Files dialog is displayed. Click on the Browse button to find the directory that contains the XSD file(s) you wish to import.
 - To select all of the XSD files in the directory, click the checkbox next to the folder in the left panel.
 - To select individual XSD files, click the checkboxes next to the files you want in the right panel

🧐 Import XML Schema Files 🛛 🕅
File system Import resources from the local file system.
From directory: np/testdata/johndoe/test-data/schemas/books 🗸 🛛 🛛 🗛 🖉
🗖 🗁 books
✓ X Books.xsd
☐ X BooksInput.xsd
Filter Types Select All
Into fo <u>l</u> der: BooksProject Bro <u>w</u> se
Options
Overwrite existing resources without warning
O <u>C</u> reate complete folder structure
Create selected folders only
Advanced >>
✓ Add Dependent Schema Files
? < <u>Back</u> <u>Next</u> > Cancel <u>Finish</u>

Figure 4.63. Select XSD From File System

• **Step 4b** - If importing from URL, select the *Import XML Schemas via URL* option and click *OK* to display the final *Add XML Schema URLs* wizard page.

9	Imp	ort XML Sche	ma Files	×
Add XML S	Schemas UR	Ls		sî)
XML schem	a URLs			C1 ×
Into folder:	BooksProject			Browse
Options-				
Overwri	ite existing res	ources without	warning	
Add Dep	pendent Schem	ia Files		
?	< <u>B</u> ack	<u>N</u> ext >	Cancel	Finish

Figure 4.64. Add XML Schema URLs Dialog

Step 5 - Click the Add XML Schema URL button

Enter a valid schema URL. Click OK. Schema will be validated and resulting entry added to the list of XML Schema URLs.

@	XML Schema Url	×
Enter XML schema URL: http	o://ns.hr-xml.org/2_4/HR-XML-2_4/StandAlone/Resume.xsd)
Optional User Name Password I Verify Hostname (HTTPS)		
	Cancel OK]

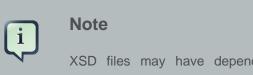
Figure 4.65. Add XSD Schema URLs

The schema URL is now displayed in the XML Schema URLs list.

	Imp	ort XML Scho	ma Files	×
Add XML Sc	hemas Uf	₹Ls		SP
XML schema (JRLs			C1 🗙
http://ns.hr-	xml.org/2_4/	'HR-XML-2_4/St	andAlone/Resum	ie.xsd
Into folder: B Options <u>O</u> verwrite		sources withou	t warning	Browse
✓ Add Depe	ndent Schen	na Files		
?	<u>B</u> ack	Next >	Cancel	<u> </u>

Figure 4.66. Add XSD Schema URLs

• Step 6 - Click Finish.



XSD files may have dependent files. This importer will determine these dependencies and import these as well if Add Dependent Schema Files is checked

Creating and Editing Model Objects

This section summarizes Teiid Designer features for creating and editing existing model objects contained in your models.

5.1. Creating New Model Objects

As discussed in the introduction, Section 1.1, "What is Teild Designer?", Teild Designer provides a framework to model various types of metadata. Each metamodel type has a set of parentchild relationships that establish constraints on what can be created and where. You cannot, for example, create a column attribute in a stored procedure, nor can you create a mapping class column in a Web service operation's output message.

The Teiid Designer provides a common set of actions to create new children of these models as well as children of children.

- You can create new model objects directly in the Section C.2.1, "Model Explorer View" view, Section C.3.1.1, "Diagram Editor" or Section C.3.1.2, "Table Editor" using the following actions:
 - New Child Action
 - New Sibling Action
 - New Association Action

5.1.1. New Child Action

- To create new child model objects in the Section C.2.1, "Model Explorer View":
 - **Step 1** Select the parent object to which you want to add a child. For example, you can add a package to a package or an attribute to a class.
 - Step 2 Right-click on a container object. From the pop-up menu, select New Child. You can now select the child object you would like to add.

▼ 🚰 TwoSourceParts ▼ 🏫 PartsDB2.xmi ▷ ⁴宝 import dec ಔ Package Di				
PARTS PARTS SHIP_VIA STATUS SUPPLIER SUPPLIER	New Child New Sibling New Association Modeling	> > >	A F	Access Pattern Column Foreign Key Primary Key
 Data Sour math PartsSQL.xm 	<₽ Undo ♥> Redo	Ī	U	Unique Constraint

Figure 5.1. New Child Action In Model Explorer

• Step 3 - The new model object displays on the Section C.2.1, "Model Explorer View" and is highlighted for renaming.

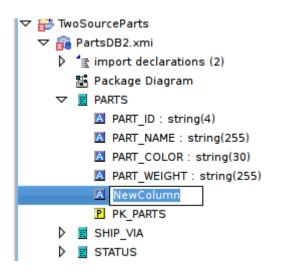


Figure 5.2. New Model Object In Explorer

- To create new child model objects in the Section C.3.1.1, "Diagram Editor":
 - **Step 1** Select the parent object to which you want to add a child. For example, you can add a package to a package or an attribute to a class.
 - Step 2 Right-click on a container object. From the pop-up menu, select New Child. You can now select the child object you would like to add.

< <base table=""/> >	i		
SHIP VIA	New Child	\rightarrow	A Access Pattern
	New Sibling	>	🔺 Column
SHIPPER_ID : bigdecimal SHIPPER NAME : string(3)	New Association	>	F Foreign Key
A SHIFFER_NAME . Suring(.	Modeling	>	P Primary Key
P PK_SHIP_VIA	💛 Undo Create NewColumn		Unique Constraint
••	> Redo		
	~		

Figure 5.3. New Child Action In Diagram

• Step 3 - The new model object displays on the diagram and is highlighted for renaming.

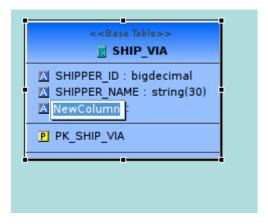


Figure 5.4. New Model Object In Diagram

- To create new child model objects in the Section C.3.1.2, "Table Editor":
 - **Step 1** Select the row for the parent object to which you want to add a child. For example to add a column, click the **Base Table** tab and select base table row.
 - Step 2 Right-click on a table row. From the pop-up menu, select New Child. You can now select the child object you would like to add.

PartsDB2.xn	ni 🕄 🕠	PartsViews.xmi		
🗾 Base Table	s 🖪 Colum	ns F Foreign Keys P	Primary Keys	
Location	Name	Name In Source	System	Cardinality Supports U
	PARTS SHIP_VIA STATUS SUPPLIER SUPPLIER	New Child New Sibling New Association Modeling Windo Delete multiple	objects	 A Access Pattern Column F Foreign Key Primary Key Unique Constraint

Figure 5.5. New Child Action In Table Editor

• Step 3 - The selected tab in the Table Editor changes to the tab for the child object type, the new model object row is displayed and the row's name table cell is highlighted for renaming.

5.1.2. New Sibling Action

- To create new sibling model objects in the Section C.2.1, "Model Explorer View":
 - **Step 1** Select the object to which you want to add a sibling. For example, you can add a column sibling to a column.
 - Step 2 Right-click on that object. From the pop-up menu, select New Sibling. You can now select the sibling object you would like to add.

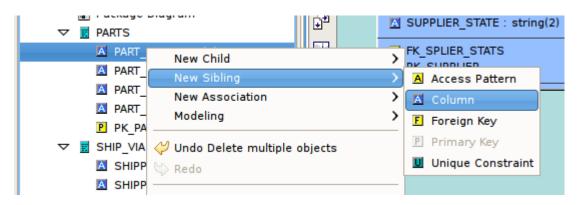


Figure 5.6. New Sibling Action In Model Explorer

- Step 3 The new model object displays on the Section C.2.1, "Model Explorer View" and is highlighted for renaming.
- To create new sibling model objects in the Section C.3.1.1, "Diagram Editor":

- **Step 1** Select the object to which you want to add a sibling. For example, you can add a column sibling to a column.
- Step 2 Right-click on that object. From the pop-up menu, select New Sibling. You can now select the sibling object you would like to add.

< <base table=""/> > SHIP_VIA			
SHIPPER_ID : bigdecin	New Child	>	
	New Sibling	\rightarrow	Access Pattern
P PK_SHIP_VIA	New Association	>	A Column
	Modeling	>	F Foreign Key
	💛 Undo Delete multiple objects		P Primary Key
	> Redo		Unique Constraint

Figure 5.7. New Sibling Action In Diagram

- Step 3 The new model object displays on the diagram and is highlighted for renaming.
- To create new sibling model objects in the Section C.3.1.2, "Table Editor":
 - **Step 1** Select the row for the object to which you want to add a sibling. For example, you can add a column sibling to a column.
 - Step 2 Right-click on a row. From the pop-up menu, select New Sibling. You can now select the sibling object you would like to add.

📬 PartsDB2.xmi 🕄 👘 PartsViews.xmi						
🖪 Base Tab	oles 🖪 Columns F	Foreign Key	/s Primary Keys			
Location	Name	Name In So	ource System	Cardinality S	Supp	orts Up Materialized
	PARTS	PARTS	New Child	:	>	false
	SHIP_VIA	SHIP_VIA	New Sibling		⊃)Î	👿 Base Table
	STATUS	STATUS	New Association		>	🗖 Catalog
	SUPPLIER	SUPPLIER	Modeling		>	I Index
	SUPPLIER_PARTS	SUPPLIER	💛 Undo Delete multij 🔆 Redo	ole objects		 Logical Relationsh Procedure Schema
			of Cut	Ctrl+	۰x	View
			📄 Сору	Ctrl+	-c	🔠 Custom Diagram
			🛗 Table Paste		ľ	

Figure 5.8. New Sibling Action In Table Editor

• Step 3 - The selected tab in the Table Editor changes to the tab for the child object type, the new model object row is displayed and the row's name table cell is highlighted for renaming.

5.1.3. New Association Action

- To create new associations between model objects in the Section C.2.1, "Model Explorer View":
 - Step 1 Select two objects you wish to associate. For example, select columns in different base tables.
 - Step 2 Right-click. From the pop-up menu, select New Association > Foreign Key Relationship.

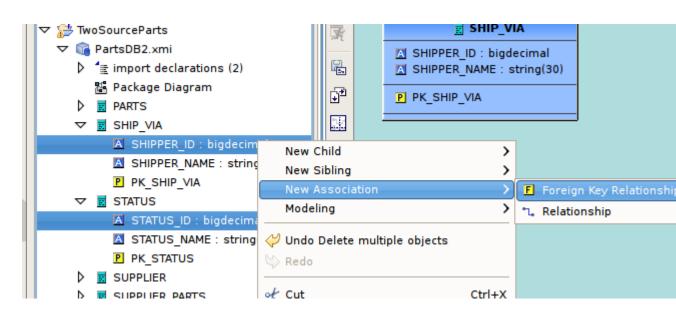


Figure 5.9. New Association Action In Model Explorer

• Step 3 - The new relationship link is displayed in the diagram.

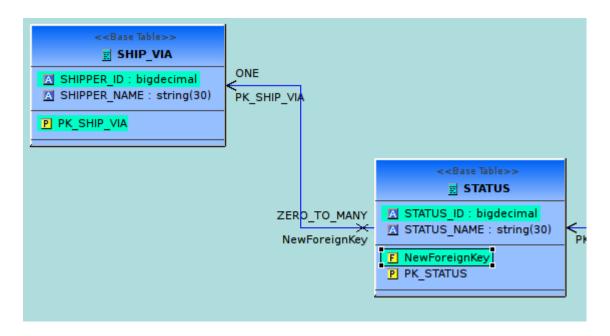


Figure 5.10. New Association In Diagram

- To create new associations between model objects in the Section C.3.1.1, "Diagram Editor":
 - Step 1 Select two objects you wish to associate. For example, select columns in different base tables.
 - Step 2 Right-click. From the pop-up menu, select New Association > Foreign Key Relationship..

• **Step 3** - The new relationship link is displayed in the diagram. The Column, Foreign Key, Primary Key reference properties are properly set on the selected columns, new primary key and new foreign key.

OR

- Step 1 Select a column in table.
- **Step 2** Drag the column to another table and drag over a column and drop onto this column. The target column should highlight in Yellow.
- **Step 3** The new relationship link is displayed in the diagram. The Column, Foreign Key, Primary Key reference properties are properly set on the selected columns, new primary key and new foreign key.
- To create new associations between model objects in the Section C.3.1.2, "Table Editor":
 - Step 1 Select two objects you wish to associate. For example, select columns in different base tables.
 - Step 2 Right-click. From the pop-up menu, select New Association > Foreign Key Relationship..
 - **Step 3** New Foreign Key and Primary Key objects will be added to the contents of their respective tabs in the Table Editor. The Column, Foreign Key, Primary Key reference properties are properly set on the selected columns, new primary key and new foreign key.

5.2. Model Object Editors

The primary actions for editing model objects are:

• • t

Cut - Deletes the selected object(s) and copies it to the clipboard.

• 🗈

Copy - Copies the selected object(s) to the clipboard.

• 🖻

Paste - Pastes the contents of the clipboard to the selected context.

• **Clone** - Duplicates the selected object in the same location with the same name; user is able to rename the new object right in the tree.

×

Delete - Deletes the selected object(s).

• Rename - Allows a user to rename an object.

These actions are presented in Teiid Designer's main **Edit** menu and also in the right-click context menus for model objects selected in the Section C.2.1, "Model Explorer View", Section C.3.1.1, "Diagram Editor" and Section C.3.1.2, "Table Editor".

Modeling Sub-Menu. In addition to the New Child/Sibling/Association menus available for object creation Designer provides a *Modeling* > sub-menu which presents various object-specific actions which can be performed.

If you select a source table, for instance, the modeling menu below would be presented:



Figure 5.11. Modeling Sub-Menu for Source Table

If a view table is selected, the menu would reflect the actions related to virtual operations:

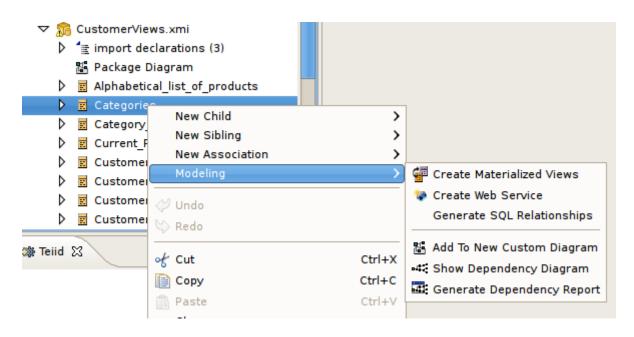


Figure 5.12. Modeling Sub-Menu for Source Table

Teiid Designer also provides specialized object editors to handle complex model objects and their unique properties. These editors include:

- Section 5.2.1, "Transformation Editor"
- Section 5.2.2, "Input Set Editor (XML)"
- Section 5.2.3, "Choice Editor (XML)"
- Section 5.2.4, "Recursion Editor (XML)"
- Section 5.2.5, "Operation Editor"

This section describes these editors in detail.

5.2.1. Transformation Editor

The **Teiid Designer's Transformation Editor** enables you to create the query transformations that describe how to derive your virtual metadata information from physical metadata sources or other virtual metadata and how to update the sources.

The **Transformation Editor** provides a robust set of tools you can use to create these SQL queries. You can use these tools, or you can simply type a SQL query into the Transformation Editor.

To edit a transformation you can:

- Double-click Edit
 - A relational view table or procedure in the Model Explorer or Diagram Editor
 - A transformation node in a transformation diagram or mapping transformation diagram
 - A mapping class in a mapping diagram or mapping transformation diagram
- Right-click Edit action on selected object in the Model Explorer, Diagram Editor or Table Editor
 - A relational view table or procedure
 - A transformation node in a transformation diagram or mapping transformation diagram
 - A mapping class in a mapping diagram or mapping transformation diagram

If a Model Editor is not currently open for the selected object's model, a Model Editor will be opened.

After the corresponding transformation diagram is opened in the Diagram Editor, the Transformation Editor is displayed in the lower section of the Diagram Editor.

Transformation Editor	🕺 🖋 🐺	Cursor at (1, 1)	🗌 Supports U	Ipdate 🖏	VB 😭	
SUPPLIER_CITY, SUP FROM PartsSourceA.S WHERE	PLIER_STATE	SUPPLIER_ID, PAR PartsSourceB.SU	IPPLIER	_		R_NAME, S

Figure 5.13. Editing String Property

If this virtual class supports updates, the tabs on the bottom of the **Transformation Editor** allow you to enter SQL for each type of query this virtual class supports. If this virtual class does not support updates, only the **SELECT** tab is available.

You can enter separate SQL queries on each available tab to accommodate that type of query.

Within the Transformation Editor, you can:

- Disable specific update transformation types on this virtual class.
- Start your transformation with a provided SQL Template.
- Build or edit a criteria clause to use in your transformation.
- Build or edit an expression to use in your transformation.
- Find and replace a string within your transformation.
- Validate the transformation to ensure its content contains no errors.
- Reconcile target attributes to ensure the symbols in your transformation match the attributes in your virtual metadata class.

You can also set preferences that impact the display of your **Transformation Editor**. For more information, see Section B.1.3.3, "Transformation Editor Preferences"

• The Transformation Editor toolbar actions are summarized below.

• 🐝

Prevew Virtual Data - executes a simple preview query for the target table or procedure of the transformation being edited.

• 🚀

Search Transformations - provides a simple way select and edit another transformation based SQL text search criteria.

• 🛒

Edit Transformation - provides a simple way to change which transformation to edit without searching in a diagram or the Model Explorer. Simply click the action and select from a list of views, tables, procedures or operations from the currently edited model.

Cursor at (1, 1)

Cursor Position (line, column) - shows the current line and column position of the insertion cursor. For example, Cursor Position(1,4) indicates that the cursor is presently located at column 4 of line 1.

🗌 🔲 Supports Update

Supports Update - checkbox allows you to enable or disable updates for the current transformation target. If 'Supports Update' is checked, the editor shows four tabs at the bottom for the Select, Update, Insert and Delete transformations. If 'Supports Update' is unchecked, all updates are disabled and only the Select transformation is displayed.

• 🖬 🎝

Reconcile - allows you to resolve any discrepancies between the transformation symbols and the target attributes. Pressing this button will display the "Reconcile Virtual Target Attributes" dialog box in which you can resolve discrepancies. See *Section 5.2.1.1, "Using the Reconciler"* for more information about the Reconciler Dialog.

• V_{PP}

Save/Validate - saves edits to the current transformation and validates the transformation SQL. Any Warning or Error messages will be displayed at the bottom of the editor in the messages area. If the SQL validates without error, the message area is not displayed.

• 🛅

Criteria Builder - allows you to build a criteria clause in your transformation. The button will enable if the cursor position is within a query that allows a criteria. Pressing the button will launch the Criteria Builder dialog. If the Criteria Builder is launched inside an existing criteria, that criteria will be displayed for edit, otherwise the Criteria Builder will be initially empty. See *Section 5.2.1.3, "Using the Criteria Builder"* for further information.

・ 檀

Expression Builder - allows you to build an expression within your transformation. The button will enable if the cursor position is at a location that allows an expression. Pressing the button will launch the Expression Builder dialog. If the Expression Builder is launched inside an existing expression, that expression will be displayed for edit, otherwise the Expression Builder will be initially empty. See *Section 5.2.1.4, "Using the Expression Builder"* for further information.

• 🧕

Expand Select * - allows you to expand a "SELECT *" clause into a SELECT clause which contains all of the SELECT symbols. The button will enable only if the cursor is within a query that contains a SELECT * clause that can be expanded.

۲.

Increase Font Size - increases the font size of all editor text by 1.

• 🗸

Decrease Font Size - decreases the font size of all editor text by 1.



2

Show/Hide Messages - toggles the display of the message area at the bottom of the transformation editor.

• a.b.c • ⊂

Optimize SQL - when toggled 'ON', will use the short names of all SQL symbols that can be optimized. Some symbol names may remain fully qualified in the event of a duplicate name or if the optimizer is unable to optimize it. When the action is toggled 'OFF', all symbol names will be fully-qualified.

· 💪

Import SQL Text - allows you to import a sql statement from a text file on your file system. Pressing this button will display an import dialog in which you can navigate to the file.

• 🔓

Export SQL Text - allows you to export the currently displayed SQL statement into a text file on your file system. Pressing this button will display an export dialog in which you can choose the location for export.

- Close "X" closes the transformation editor.
- The **Transformation Editor** context menu can be displayed by Rt-Clicking within the editor's text area. The context menu is show below:

Transformation	Editor	B
	Cut Copy Paste Select All	
	Undo Redo	
ERROR: SQL	Find Apply Template	fault) DELETE (

Figure 5.14. Transformation Editor context menu

Following is a summary of the context menu actions:

- Cut Copy Paste Typical text editor actions to cut, copy or paste text within the editor.
- Undo Redo Allows you to Undo or Redo the previous action.
- **Find** Displays a Find-Replace Dialog which allows you to search and replace text within the transformation.
- Apply Template... Displays the 'Choose a SQL Template' Dialog, which allows you to choose a starting SQL Template from a list of common SQL patterns. See Section 6.2.2, "Create Relational View Table Wizard" for a description of this dialog.

5.2.1.1. Using the Reconciler

The Transformation Editor's **Reconciler** offers you a quick, graphical means to reconcile the Target View attributes and the Transformation SQL. As you make changes, the overall status will appear at the top of the dialog to assist you in successfully completing your edits.

To launch the Reconciler, click on the **Reconcile Transformation** button

in the Transformation Editor. The Reconciler Dialog is shown below:

9	Reconcile Virtual Target Attributes				
Reconciler Status					
🔞 The SQL contains unmatched symbo	ols - Remove them or create new attribut	es.			
SCOPE: Reconciling Query			Unmatched SQL Syr		
Note: The data types shown are the ro	untime data types.		S PART ID		
			S QUANTITY		
Target Locked	Attribute - SQL Bindings		SHIPPER_ID		
Virtual Target Attributes	SQL Symbol				
SUPPLIER ID : string	SUPPLIER ID : string	< Bind			
SUPPLIER_NAME : string	SUPPLIER_NAME : string	Unbind >			
SUPPLIER_STATUS : bigdecimal	SUPPLIER_STATUS : bigdecimal				
SUPPLIER_CITY : string	SUPPLIER_CITY : string	< New			
SUPPLIER_STATE : string	SUPPLIER_STATE : string	< Null			
PART_ID : string	S PART_ID : string				
QUANTITY : bigdecimal	S QUANTITY : bigdecimal	7 200			
SHIPPER_ID : bigdecimal	SHIPPER_ID : bigdecimal				
Top Up Swap Down Bottom Delete Type Resolver Add Remove					
SELECT PartsDB2. SUPPLIER. SUPPLIER_ID, SHIPPER_ID, PartsSQL. SUPPLIER_PARTS FROM PartsDB2. SUPPLIER, PartsSQL. SU	SUPPLIER_NAME, SUPPLIER_STATUS, SUP .PART_ID, PartsSQL.SUPPLIER_PARTS.QUA IPPLIER_PARTS	PLIER_CITY, SUP NTITY, PartsSQL	PLIER_STATE, PART_ID SUPPLIER_PARTS.SHI		
?			Cancel		

Figure 5.15. Reconciler Dialog

To summarize the different sections of the dialog:

• Target Attributes - SQL Symbol Table: This table shows the target attributes in the left column and the SQL Symbols in the right column. The SQL Symbols are the symbols that are 'projected' from the SQL transformation. A symbol is referred to as being 'bound' to a target attribute when it is displayed next to the attribute.

If a target attribute is 'unbound', its row is highlighted in red. The transformation is not valid until all attributes have a corresponding SQL symbol binding.

Here are a few things you can do in the table section:

- Lock Target Attributes: To 'lock' the target attribute ordering, check the 'Lock Target Attributes' checkbox. This will lock the attributes in place.
- **Re-Order Attributes**: To change the ordering of the target attributes, use the 'Top', 'Up', 'Swap', 'Down', and 'Bottom' controls beneath the table. Select or multi-select the table rows, then click the desired action button.
- **Delete Attributes**: To delete one or more of the target attributes, select the table row(s) you want to delete then click the 'Delete' button.
- **Resolve Types**: If an Attribute-SQL Symbol binding has a datatype conflict, a message will be displayed. To assist in resolving the datatype conflict, a 'Datatype Resolver Dialog' is provided. Click on the table row, then click the 'Type Resolver...' button to display the dialog. See *Section 5.2.1.2, "Using the Datatype Resolver"* for further information.
- Unmatched SQL Symbols list: This list is to the right of the attribute-symbol binding table, and shows the SQL symbols from the transformation SQL that are not 'bound' to a target table attribute.

Here are a few things you can do in the list section:

- Add SQL Symbols: To 'Add' SQL Symbols to the list, click the 'Add' button. You will be presented with a dialog showing all available symbols from your transformation source tables. Click on the symbols you want to add, then click 'OK'.
- Remove or Clear Symbols: To remove one or more of the SQL symbols, select the list items then click the 'Remove' button. To clear the entire SQL symbols list, click the 'Clear' button.
- **Sort Symbols**: By default, the symbols are shown in the order that they appear in the SQL query. To show them alphabetically in the list, click the 'Sort' button.
- **Binding Controls**: The 'Binding Controls' are located between the Attribute-Symbol table and the Unmatched SQL Symbols list. Use these buttons to define the Attribute-Symbol bindings.

Here are a few things you can do with the binding controls:

- **Bind**: This button will 'Bind' a SQL Symbol to a target attribute. Select an Unmatched SQL symbol and select a target attribute, then click 'Bind' to establish the binding.
- **Unbind**: This button will 'Unbind' an Attribute-Symbol binding. Select an already-bound attribute in the table, then click 'Unbind'. The SQL Symbol will be released to the Unmatched Symbols list.
- New: This button will create a new target attribute, using an Unmatched SQL Symbol. Select an Unmatched Symbol from the list, then click 'New'. A new target attribute will be added to the bottom of the Attribute-Symbol table, bound to the selected SQL symbol.
- Null: This button allows you to bind 'null' to a target attribute instead of binding a SQL Symbol to it. Select a row in the Attribute-Symbol table, then click 'Null'. The target attribute will be

bound to 'null'. If it was originally bound to a SQL Symbol, the symbol will be released to the Unmatched Symbol list.

- Function: This button allows you to define an expression instead of just a SQL Symbol for the binding. To define the expression, select a row in the Attribute-Symbol table, then click the 'Function' button. The Expression Builder Dialog will display, allowing you to define any type of expression. See Section 5.2.1.4, "Using the Expression Builder" for further information about the Expression Builder.
- SQL Display: The current transformation SQL is shown at the bottom of the reconciler dialog. As you add / remove SQL symbols and make other changes, you can see the SQL display change to reflect those changes. When you 'OK' the dialog, this SQL will be your new transformation SQL. If desired, the SQL Display can be hidden by un-checking the 'Show SQL Display' checkbox.

Once you are finished defining the bindings and resolving datatypes, click 'OK' to accept the changes. The transformation SQL will change to reflect your edits.

5.2.1.2. Using the Datatype Resolver

This dialog is accessible from the **Reconciler** dialog (See Section 5.2.1.1, "Using the Reconciler") and offers you a quick way to resolve datatype conflicts between a target attribute and its SQL Symbol. The Datatype Resolver Dialog is shown below:

🛞 Resolve Data	type Conflicts 🛞			
Oatatype Status One or more of the Bindings has a type conflict.				
Attribute - SQL Bindings				
Virtual Target Attributes	SQL Symbol			
SHIPPER_ID : biginteger	SHIPPER_ID : bigdecimal			
EDIT Virtual Target Attribute SHIPPER_ID Runtime Type: D biginteger				
Apply Convert All Change Integer : xs:decimal				
SQL Symbol				
Runtime Type: bigdecimal Converted Symbol: convert(SHIPPER_ID, biginteger) AS SHIPPER_ID * Warning * this type conversion may result in loss of precision Apply Convert All				
?	Cancel OK			

Figure 5.16. Datatype Resolver Dialog

To summarize the different sections of the dialog:

- Target Attribute SQL Symbol Table: This table shows all target attribute SQL Symbol bindings from the Reconciler Dialog which have a type conflict. Select on a table row to populate the lower Edit Panel
- **Edit Panel**: The lower panel shows the Target Attribute and SQL Symbol datatype information for the selected binding. You can resolve the conflict in one of the following ways:

- Virtual Target Attribute: Resolve the type conflict by changing the target attribute type to be compatible with the SQL Symbol type. The attribute's current runtime type is shown, along with a potential new datatype and some button controls:
 - **Apply Button**: If the suggested datatype is acceptable, click 'Apply' to allow the attribute type to be changed.
 - **Convert All Button**: If you wish to change all of the attribute types in the table to be compatible with its corresponding SQL Symbol datatype, click the 'Convert All' button.
 - **Change Button**: If the suggested datatype is not acceptable, click 'Change' to choose your own datatype from a datatype dialog.
- **SQL Symbol**: Resolve the type conflict by applying a CONVERT function to the SQL Symbol, so that its type is compatible with the target attribute type. The SQL Symbol's current type is shown, along with a suggested CONVERT function and two button controls:
 - **Apply Button**: If the suggested CONVERT function is acceptable, click 'Apply' to apply the CONVERT function to the SQL Symbol.
 - **Convert All Button**: If you wish to apply a CONVERT function to all of the SQL Symbols in the table so that their datatype is compatible with the corresponding attribute datateyp, click the 'Convert All' button.

Once you are finished resolving datatypes, click 'OK' to accept the changes. You are directed back to the Reconciler Dialog, which will be updated to reflect your edits.

5.2.1.3. Using the Criteria Builder

The Transformation Editor's **Criteria Builder** offers you a quick, graphical means to build criteria clauses in your transformations based on meta objects in your diagram. If you launch the **Criteria Builder** with your cursor within an existing criteria in your transformation SQL, the builder will open in Edit mode. If your cursor is not in an existing criteria location, the builder will open in create mode and allow you to create it from scratch.

This procedure provides an example of building a criteria clause using the **Criteria Builder**. When building your own criteria, you can mix and match the values and constants with whatever logic you need to build powerful and complex criteria.

- To use the Criteria Builder:
 - Step 1 In the Transformation Editor, click the Launch Criteria Builder button.
 - Step 2 The Criteria Builder displays.

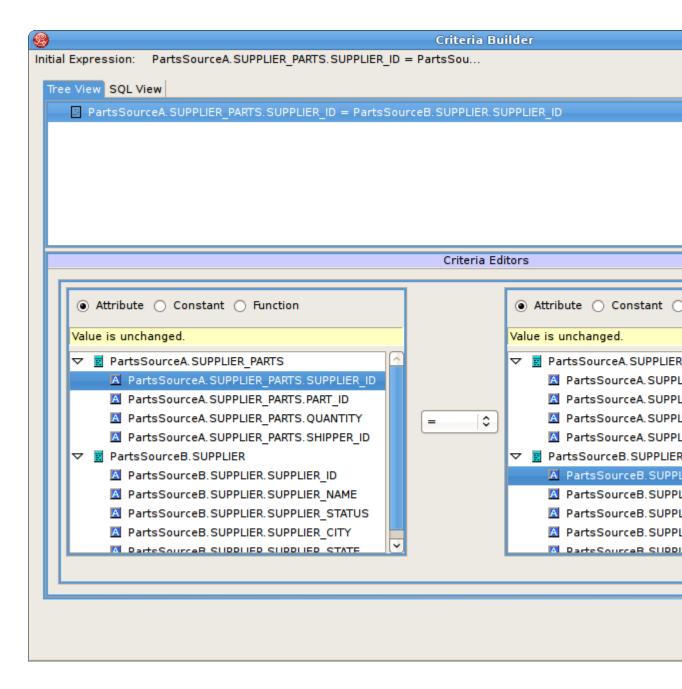


Figure 5.17. Editing String Property

The two tabs at the top, **Tree View** and **SQL View**, show the current contents of the criteria you have built.

The **Criteria Editor** at the bottom allows you to build a criteria clause. To build a criteria clause, you must add information to the left side of the predicate, select a comparison operator, and add a value to the right side.

• Step 3 - The radio buttons on either side of the **Predicate Editor** let you choose what type of content to place in that side of your predicate. Click the radio button of the type of content you want to place in your criteria. You can click:

• Attribute to add an attribute to the predicate. If you click the Attribute radio button, the Predicate Editor looks like this:

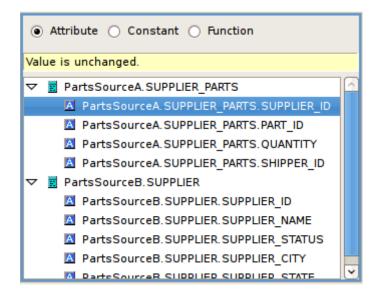


Figure 5.18. Attribute Panel

From the tree, select the attribute you want to add to the expression. You can select an attribute from any of the source classes in the transformation.

• **Constant** to add a hard-wired constant value to the predicate. If you click this radio button, the **Predicate Editor** looks like this:

O Attribute Constant O Function						
'Apply' to replace selected Attribute with Constant below.						
Type: string						
Value						
CompanyID						

Figure 5.19. Constants Panel

Select the datatype for this constant from the Type drop-down list and enter the value in the Value edit box.

• Function to add a function. If you click the Function radio button, the Predicate Editor looks like this:

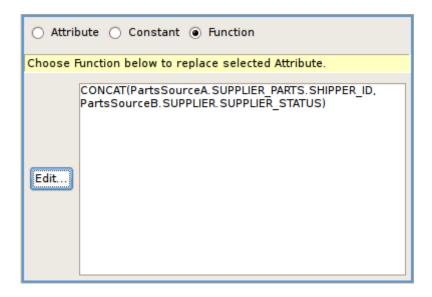


Figure 5.20. Functions

Click the Edit button to use the Expression Builder to construct a function to use in the predicate of your SQL Criterion. For more information about the Expression Builder, see *Section 5.2.1.4, "Using the Expression Builder"*

- Step 4 Set a value left side of the predicate and, when necessary, the right side of the predicate. If the right side of the predicate does not require a value of some sort, the Criteria Builder will not let you enter one.
- Step 5 Click Apply.
- **Step 6** When you have created both a Left Expression and a Right Expression in the Predicate Editor, click Apply to add the criterion to the tree view at the top of the dialog box.

The criteria clause displays in the Criteria tree.

You can create complex criteria by joining other criteria with this one. To join criteria with this one, select the criteria in the Criteria tree and click:

- Delete to remove the selected criterion.
- AND to create a new criterion that must also be true.
- OR to create a new criterion that can be true instead of the selected criterion.

• NOT to establish negative criterion.

If you join a criterion to the one you just completed, you build the expression the same way, using the Expression Editors panel and the Predicate Editor panel. You can create complex, nested criteria by judicious use of the AND and OR buttons.

Once you have created the complete criteria you want, click OK to add it to your transformation.

5.2.1.4. Using the Expression Builder

The **Transformation Editor's Expression Builder** offers you a quick, graphical means to build expressions in your transformations. This **Expression Builder** lets you create:

- Attributes by selecting an attribute.
- Constants by selecting the datatype and value.
- Functions from both the standard Teiid Designer SQL functions and your enterprise's custom user-defined functions. If you select a function before you launch the Expression Builder, you can use the Expression Builder to edit the selected function; otherwise, you can create a new function from scratch.
- To use the Expression Builder:
 - Step 1 In the Transformation Editor, click the location where you want to insert the function.
 - Step 2 Click the Expression Builder

button.

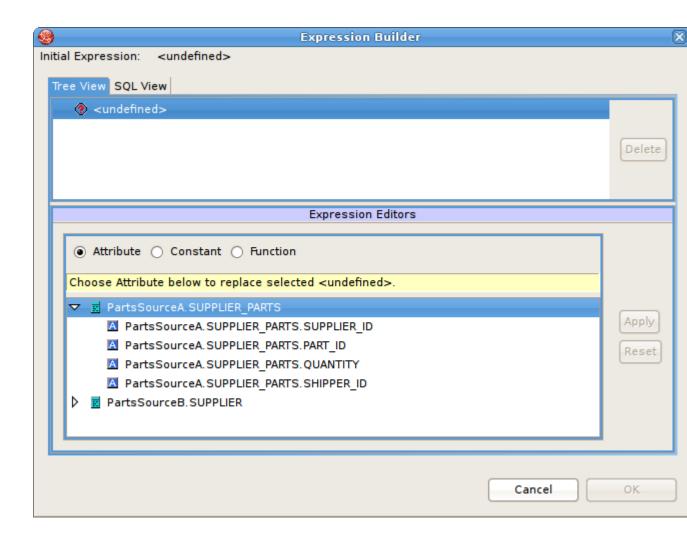


Figure 5.21. Expression Builder

The two tabs at the top, Tree View and SQL View, show the current contents of the expression you have built. To build an expression, you must specify the type of expression you want to build and populate it. In most cases, you will use the Expression Builder to construct a complex expression.

• Step 3 -Click the Function radio button to add a function.

i	Note
$\langle \gamma \rangle$	You can simply add constants and attributes as expressions by themselves using the Attribute or Constant radio buttons; however, the Expression Editor is most useful for functions.

• Step 4 - The Expression Editor displays the Function editor.

9		Expression Builder		×		
Initial Expression	n: <unde< th=""><th>efined></th><th></th><th></th></unde<>	efined>				
Tree View SQL	View					
📀 <undef< th=""><th>ined></th><th></th><th></th><th></th></undef<>	ined>					
				Delete		
		Expression Editors				
		Expression Editors				
 Attribut 	🔿 Attribute 🔿 Constant 💿 Function					
Apply to collected supplefineds with Eurotian below						
Apply to re	'Apply' to replace selected <undefined> with Function below.</undefined>					
Category:	STRING		•	Apply		
Function:	CONCAT(STRING1, STRING2)	0	Reset		
Argument	Name	Value		Reset		
STRING		<undefined></undefined>				
STRING	G2	<undefined></undefined>				
		_				
			Cancel	OK		

Figure 5.22. Function Panel Selected

From the Category drop-down list, choose the type of function you want to add. By default, the Teiid Designer System offers the following categories:

- Conversion for functions that convert one datatype into another.
- Datetime for functions that handle date or time information.
- Miscellaneous for other functions.
- Numeric for mathematic and other numeric functions.
- String for string manipulation functions.



Note

Any additional categories represent those containing user-defined functions your site has created.

- **Step 5** From the **Function** drop-down list, select the function you want. The table beneath the drop-down lists displays the number of arguments required for this function.
- Step 6 Click Apply.
- Step 7 Your function displays in the tree at the top. Sub nodes display for each argument you need to set for this function.

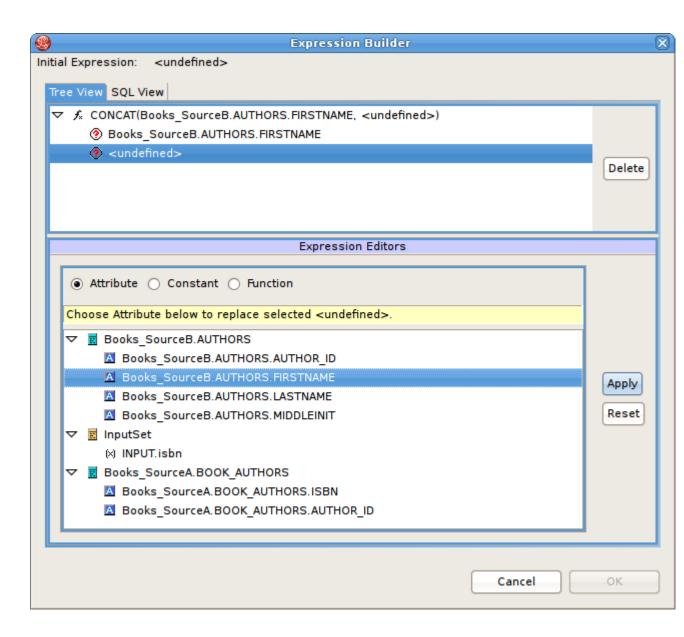


Figure 5.23. New Blank Function Created

You need to set an attribute or constant value for each sub node in the tree to specify the arguments this function needs. You can also nest another function in the tree using the **Function** editor.

		Expression Builder	
al Expression	c <undefined></undefined>	•	
ee View SQL	View		
		B.AUTHORS.FIRSTNAME, Books_SourceB.AUTHORS.LASTNAME)	
	—	HORS.FIRSTNAME	
		HORS.LASTNAME	
0 000	Ca_Sourceb.Aon	HORS. ERS HIRANE	Delete
		Expression Editors	
	STRING	1. STRING2)	Apply
Category: (Function: (STRING CONCAT(STRING	31, STRING2)	Apply Reset
Category: Function: Argument	STRING CONCAT(STRING Name	61, STRING2) Value	
Category: Function: Argument STRING	STRING CONCAT(STRING Name	S1, STRING2) Value Books_SourceB.AUTHORS.FIRSTNAME	
Category: Function: Argument	STRING CONCAT(STRING Name	61, STRING2) Value	
Category: Function: Argument STRING	STRING CONCAT(STRING Name	S1, STRING2) Value Books_SourceB.AUTHORS.FIRSTNAME	
Category: Function: Argument STRING	STRING CONCAT(STRING Name	S1, STRING2) Value Books_SourceB.AUTHORS.FIRSTNAME	
Category: Function: Argument STRING	STRING CONCAT(STRING Name	S1, STRING2) Value Books_SourceB.AUTHORS.FIRSTNAME	

Figure 5.24. Nested Function Example

- **Step 8** Click each sub node in the tree and use the editors at the bottom of the dialog box to apply an attribute, constant, or function value to it.
- Step 9 When you have added values to all nodes, as shown below, click OK. to add this expression to your query or Cancel to close the dialog box without inserting the expression.

If the **OK** button does not enable, you have not added a value to all nodes in the tree. You can also nest functions within your expressions by selecting an argument and selecting a function for that argument. The nested function displays in the tree beneath your root function and its arguments display as well. Using the Expression Builder and nested functions, you can create complex logic within your query transformations.

5.2.2. Input Set Editor (XML)

The **Input Set** represents a special class that contains attributes from a parent mapping class. When you create mapping classes for an **XML Document** model, the Teiid Designer automatically adds an **Input Set** to all XML transformation diagrams for mapping classes beneath the highest node in the Document meta object.

The **Input Set** proves especially useful for information integration using the **Teiid Designer Server**. Through the **Input Set**, you can access a row of data generated by any XML transformation in a mapping class higher in the XML document's hierarchy. You can use **Input Set** attributes, which are individual columns from the rows of data, within the criteria of an XML transformation query of the child mapping class.

You cannot use the **Input Set** attributes within the SELECT portion of the XML transformation query.

To use an Input Set, you must use the Input Set Editor to bind attributes from parent classes.

Once you have created an **Input Set**, you can use the attributes within it as source material for the XML transformation diagram's query.

The **Input Set** only serves to enable data flow between nested mapping classes. If you use the **Teiid Designer Server** for data access, your applications cannot directly query an **Input Set**. **Input Sets** only display in the XML transformation diagram to which they belong. **Input Sets** do not display on the *Section C.2.1, "Model Explorer View"* view and you cannot use them as you would a normal class, such as for source classes in other transformations.

To open the **Input Set Editor**, either double-click the input set in the **Mapping Transformation Diagram** or click the edit button on the **Input Set** in the diagram. (see below)



Figure 5.25. Edit Input Set Button

nput Set Editor			
Input Parameters	Mapping Class Bindings		▼ 🖻 book
(×) isbn : string	🛛 book.isbn : string	< New	🔺 isbn : string
			🔺 type : string
		< Bind	🔺 title : string
		Unbind >	🛛 publisher : string
			🛛 publishDate : string
		Delete	🔺 edition : string

Figure 5.26. Input Set Editor Panel

The **Input Parameters** table contains a list of mapping attributes within the input set and the mapping attributes bound to input set mapping attributes. The tree on the right displays the parent mapping classes and the attributes available from each.

Using the Input Set Editor, you can:

- Add a mapping attribute from a parent mapping class to the **Input Set**. In the tree on the right, select the symbol for which you want to create an attribute and click **New**. The item displays in the **Input Parameters** and **Mapping Class Bindings** table.
- Delete a mapping attribute from the Input Set. Click the row in the Input Parameters and Mapping Class Bindings table that you want to delete and click Delete. The Teiid Designer removes this row from the table and this mapping attribute from your Input Set.
- Bind and Unbind Input Parameters.

Once you have created the mapping attributes within the **Input Set** that you need, you can use the **Input Set Parameters** within a mapping class transformation to produce mapping attributes you can map to your XML document.

5.2.3. Choice Editor (XML)

Within an XML Document model, a choice compositor defines all possible document data structures (sometimes called fragments) that can appear at that location in an XML instance document. When the Teiid Designer Server populates an XML instance document at runtime based upon your virtual XML document, it will choose the first fragment that matches the criteria you specify within the **Choice Editor**.

To view the choice editor, right-click on the choice node in the mapping diagram's XML Document tree view and select **Edit** from the right-click pop up menu.

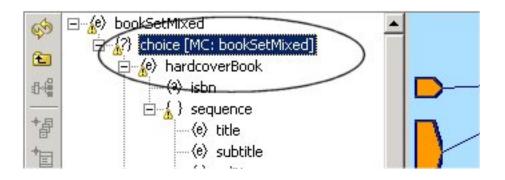


Figure 5.27. Opening The Choice Editor

	Option Name	Include	Criteria
1	😥 hardcoverBook		AudioBooks.bookSetMixedDocument.MappingClass
2	(e) softcoverBook	~	UCASE(AudioBooks.bookSetMixedDocument.Mapp
3	(0) audioBook		AudioBooks.bookSetMixedDocument.MappingClass

Figure 5.28. The Choice Editor

The table on this panel displays fragment options for the choice, each represented by the top node of the document fragment.

The Summary tab, shown below, displays a SQL-like version of the current choice criteria.

Choice Editor: bookSetMixed / choice	Def
 (1) hardcoverBook: AudioBooks.bookSetMixedDocument.MappingClasses.bookSetMixed.TYPE = 'H (2) softcoverBook: UCASE(AudioBooks.bookSetMixedDocument.MappingClasses.bookSetMixed.TY (3) audioBook: AudioBooks.bookSetMixedDocument.MappingClasses.bookSetMixed.TYPE IN ('Default: (3) audioBook 	PE) LIKE 'SOF%'
Edit Summary	

Figure 5.29. Choice Criteria Summary

5.2.3.1. Using the Choice Editor

You should address each choice option by performing one of the following:

• **Specify a criteria** statement for the Teiid Designer Server to apply in order to determine which elements or elements to insert into the result document.

- Exclude or include the option's fragment from the document.
- Set the elements' criteria test order.
- Set a default action that occurs if none of the criteria you set is met.

5.2.3.2. Excluding Fragments

The XML Schema upon which you based the XML Document model determines the nature of the options available to the choice. A schema you share with other, external sources (such as business partners) might include information that you do not want to include within XML files.

For example, Sample Financial Services shares an XML schema with its partners Example Mutual Insurance, Illustrative Brokerage, and FinancialPartners.com. The partners created the schema broadly, to cover all possibilities for information they might need to interchange. As such, the customer information XML document might include a choice compositor based on a list of all products all companies offer.

However, Sample Financial does not offer a credit card; so it could exclude those elements from the XML documents its Teiid Designer Server creates since it will never have credit card information for an XML document.

The table on the **Choice Editor** contains the **Include** column. By default, all elements specified by the schema are included. You can click to remove the checkmark beside any element you do not want to include within your XML documents generated by this virtual XML document metadata model. By removing the checkmark, you are not removing the element from the XML Document model; you are merely telling the Teiid Designer Server that it will never use this element as part of the choice.

You cannot edit criteria for excluded elements. However, if you exclude an option for which you have established a criteria, Teiid Designer will retain the criteria if you want to include the option in the future.

5.2.3.3. Editing Choice Criteria

- To edit the criteria for a choice element:
 - Step 1 In the table on the Choice Editor panel, select the element you want to edit..
 - Step 2 Click Edit Criteria button to launch the Criteria Builder dialog.
 - Step 3 Use the Criteria Builder to create the conditions for which the Teiid Designer Server will test to determine whether to choose this option in the XML instance document.
 - Step 4 Click OK. The criteria you set displays both in the table and in the summary tab.

You must set a criterion for each option in your document unless you have selected to exclude that option or specify that option will be the default option.

5.2.3.4. Setting Choice Element Order

To edit the criteria for a choice element:

The Teiid Designer Server evaluates the choice criteria in the order in which they appear, and when one choice criteria is met, the Teiid Designer Server populates the XML instance document with that option. The Teiid Designer Server might not test all criteria for all options, so their order matters a great deal.

Therefore, the order in which your options appear within the choice criteria often determines what information appears ultimately in your XML instance documents. You can reorder the option list within the choice to set the order in which the Teiid Designer Server tests the criteria.

To set this order, select an element in the table and use the





 \bigtriangledown

button to move it into a new position in the table. The new order displays both in the table and in the Choice Criteria box and reorders the XML document as well.

5.2.3.5. Setting a Default Choice Action

The default action represents the course the Teiid Designer Server should take if none of the criteria you set evaluates to true.

You can set this default using the combo box available in the Choice Editor's toolbar to:

- Any of the options within the table except those you have excluded from the document.
- **THROW** to throw a Teiid Designer Server exception.
- RECORD to record the Teiid Designer Server exception.
- **DISCARD** to place no element within the XML instance document.

Note: You must set a default action for your choice criteria.

5.2.4. Recursion Editor (XML)

Some **XML schemas** define data structures that contain self-referencing elements or datatypes. When generating XML documents, such data structures can produce an endless repetition of nested tags. This self-nesting pattern is known as **recursion**.

When generating virtual documents from **XML Schema**, the Teiid Designer detects recursive data structures in the XML Schema model and halts the recursive nesting pattern after two cycles. These two cycles serve different purposes when mapping the document:

- The **first cycle** can be thought of as an "**entry condition**" for the **recursion**. The mapping class located at this node defines a normal mapping transformation like that of any other in the document model.
- The second cycle defines a mapping transformation that will be performed repeatedly until conditions are met that will halt the document instance being generated by the Teiid Designer Server. This fragment of the document model is called the recursive fragment. The mapping transformation for this fragment is no different from the first, except that you can access the first cycle's mapping class attributes, plus you have the opportunity to specify the conditions that will halt the recursion.

You can recognize a mapping class located at the second, recursive document fragment by the looping arrow button in the top-left-hand corner of the diagram object as shown below.

When you model a virtual document based on an **XML Schema** model containing recursion, you can choose whether to treat the nested fragments as recursive. You should only use recursion when the data access pattern from your data source(s) is also recursive; in other words, when the same query transformation should be executed over and over to generate and map the nested document's data content.

By default, the Teiid Designer does not mark the recursive fragments in document models to execute recursively in the Teiid Designer Server. To take advantage of this behavior, you must open the **Recursion Editor** in the recursive mapping class *Section C.3.1.1.5, "Mapping Transformation Diagram*", mark the transformation query as recursive, and specify the recursion limit properties.

5.2.4.1. The Recursion Editor

The Recursion Editor lets you enable and limit recursion. The Recursion Editor button only displays on mapping classes, which have recursive patterns. For example, if you have an element named Employee which contains a element named Supervisor which itself contains an Employee element nested within it, you might need to limit the number of times the elements are nested within the document.

You can set the following conditions to limit the recursion:

- A fixed number of results to the query.
- A SQL-based criteria limit condition.
- A combination of both.

То	open	the	Recursion	Editor,	click	on	the	Recursion	Editor	button

Q

on the displayed mapping class.

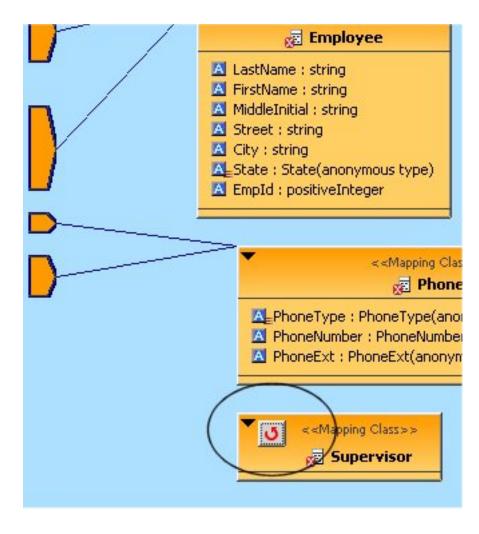


Figure 5.30. Open Recursion Editor Button

Recursion Editor: Supervisor	
Count Limit: 5 📑 Action When Count Limit Exceeded: THROW 💌	
EmployeeDocs.EmployeeDocument.MappingClasses.Supervisor.name = 'Joe Smith'	

Figure 5.31. Recursion Editor

- To edit recursion properties:
 - Step 1 Click the Enable Recursion check box if you want the Teiid Designer Server to perform the query you specify to generate the nested tags within the XML document.
 - Step 2 Click the arrows beside the Count Limit box to limit the number of times to recursively perform the query. If you do not set a Limit Condition in the text area, the recursion finishes when the query reaches this limit. You can only set this limit to a maximum supported by your Teild Designer Server. For more information about this limit, contact your system administrator
 - Step 3 Click the Action When Count Limit Exceeded drop down menu to instruct the Teiid Designer Server what to do if it encounters more results for the query than the count limit before it reaches the limit condition.
 - Step 4 Click the Edit button to launch the SQL Section 5.2.1.3, "Using the Criteria Builder" to build a limiting condition for this recursion.



Note

The Teiid Designer Server will evaluate this condition each time it recursively performs this query. If this criteria clause evaluates false, the Teiid Designer Server performs the query recursively again unless it has reached the **Count Limit**. If the criteria evaluates true, the Teiid Designer Server performs the mapping for the current level and ends its recursive loop.

When you have created the criteria, it displays in the Limit Condition box.

When the Teiid Designer Server dynamically populates your XML documents at runtime, it will use the recursion specifications you entered here.

5.2.5. Operation Editor

Editing of **Web Service** Operation transformations is simplified via the **Operation Editor**. When editing a Web Service model, an additional editor tab labeled "**Operation Editor**" is available. This editor, shown below is comprised of:

- **Operations** section showing a tree view of Interfaces and Operations contained within the Web Service model.
- Input Variables section providing editing of desired Input Variable declarations.
- **Procedure** section providing SQL editing of the procedure.

Operation Editor	📑 🧖 Cursor at (1, 1) 🕼 🖍 🛦							
Operations	Input Variables							
	The variables representing values from the request document that are available to operation's procedure.							
🔅 getbookCollection	Edit variables							
	IN_author : string							
	IN_edition : string							
	IN_isbn : string							
	IN_publishDate : string							
	Procedure The body of the selected operation's procedure (i.e., it is not necessary to include BEGIN/END clauses).							
	CREATE VIRTUAL PROCEDURE BEGIN							
	DECLARE string VARIABLES.IN_isbn = xpathvalue (BOOKCOLLECTION_WS.BOOKCOLLECTION.GETBOOKCOLLECTION.bookCo t, '/*:bookCollection/*:book/@*:isbn'); DECLARE string VARIABLES.IN_title = xpathvalue (BOOKCOLLECTION_WS.BOOKCOLLECTION.GETBOOKCOLLECTION.bookCo t, '/*:bookCollection/*:book/*:title'); DECLARE string VARIABLES.IN_subtitle = xpathvalue							

Figure 5.32. Operation Editor

The **Operations** section contains all interfaces and operations currently defined in the model.

Selecting an operation will display the variables related to the input parameter's content in the **Input Variables** section and the body of its procedure (minus the CREATE VIRTUAL PROCEDURE BEGIN - END keywords and the input variable declarations and assignments) in the **Procedure section**.

When pasting in SQL, do not include the **CREATE VIRTUAL PROCEDURE BEGIN - END** keywords. Input variables will be automatically generated when the Content via Element property is set on an operation's input parameter. Input variables may be edited using the Edit link in the **Input Variables** section, and may only represent XPath values to single attributes and elements within the input contents; other variable declarations and assignments must be typed directly into the **Procedure** section. Clicking the *Edit* link will display the following dialog:

Edit Input Variables	
Request Document Check a request document component to define an input variable for the Web Service procedure. Uncheck a request document component to delete an existing input variable. ▼ ● bookCollection : BooksNS:BookSetNested ▼ ● bookCollection : BooksNS:BookSetNested ▼ ● book : BooksNS:BookNested ● ● bookTypesNS:ISBN ✓ ● title : xsd:string ✓ ● dition : BookTypesNS:BookEdition ● ● edition : BooksNS:Authors ● ● publishingInformation : BooksNS:PublishingInformation	Input Variables Select an input variable to the XPath that will be used select its value from the request document. IN_author : string IN_edition : string IN_isbn : string IN_publishDate : string IN_publisher : string IN_subtitle : string IN_title : string
XPath for IN_title /*:bookCollection/*:book/*:title	ок

Figure 5.33. Edit Input Variables Dialog

5.3. Managing Model Object Extensions

Extending a model adds extra properties to its model objects. One good use of these extension properties is for passing data to a customized Teiid translator. The Designer model extension framework consists of:

- Model Extension Definitions (MEDs) (See Section 1.3.8, "Model Object Extensions")
- MED Registry -keeps track of all the MEDs that are registered in a workspace. Only registered MEDs can be used to extend a model.)See Section C.2.12, "Model Extension Definition Registry View (MED Registry View)")
- MED Editor (See Section C.3.3, "Model Extension Definition Editor")

5.3.1. Create New MED

To create a new MED select the **File > New > Other...** action to display the New wizard dialog. Select the **Teiid Designer > Teiid Model Extension Defn** option which displays the **New Model Extension Definition** dialog. Browse and select existing project or project folder location for MED file and specify unique file name and press **Finish**. (Note if a project is already selected when wizard is launched, the location field will be pre-populated).

New Moo	del Extension Definition 🛛 🛞
New Model Extension Define Press the "Finish" button to finish.	ition
Folder Location: Model Extension Definition Name:	ExtraParts
✓ Open in Extension Editor ? < Back	Next > Cancel Finish

Figure 5.34. MED Editor Overview Tab

5.3.2. Edit MED

To edit an MED file select an existing '.mxd' file in your workspace and right-click select the **Open** action. The MED Editor will be opened to allow editing (See Section C.3.3, "Model Extension Definition Editor").

On the Overview tab, you can specify or change the **Namespace Prefix**, **Namespace URI**, the **Model Class** you wish to extend (Relational, Web Service, XML Document, and Function) and a description. Note that version number is available, but is not currently being used in release 7.6.

After entering the basic MED info, you can now switch to the **Properties** tab and begin creating your extended property definitions for specific model objects supported by selected model class.

Propert	ties 🔹							
Extend	ed Model O	bjects 📑	8 8 8					
Manage ti	ne extended	model of	ojects that	have ext	ension	propert	ies	
BaseTabl	e							
	-							
• Extensi	on Properti	es 🖭	2 X					
	on Properti							
	on Properti xtension pro			tended n	nodel o	bject		
		perties f	or each ex			-	Default	Va
Manage e	xtension pro Runtime Type	perties f	or each ex Modifiable			Indexed	Default	Va
Manage e	xtension pro Runtime Type	perties f	or each ex			-	Default	Va
Manage e	xtension pro Runtime Type	perties f	or each ex Modifiable			Indexed	Default	Va
Manage e	xtension pro Runtime Type	perties f	or each ex Modifiable			Indexed	Default	Va
Manage e	xtension pro Runtime Type	perties f	or each ex Modifiable			Indexed	Default	Va
Manage e	xtension pro Runtime Type	perties f	or each ex Modifiable			Indexed	Default	Va
Manage e	xtension pro Runtime Type	perties f	or each ex Modifiable			Indexed	Default	Va
Manage e	xtension pro Runtime Type	perties f	or each ex Modifiable			Indexed	Default	Va

Figure 5.35. MED Editor Properties Tab

Start by selecting the **Add Extended Model Object** toolbar button to display the **Model Object Name** selection dialog. Select an object and press **OK**.

		Add Model Object Name	×
¢	Model Object Name	Choose the Model object that will have extension properties.	
$\overline{}$	BaseTable		
	Column		
	PrimaryKey		
	ForeignKey		
	AccessPattern		
	UniqueConstraint		
\neg	View		
	Column		
	AccessPattern		
\neg	Procedure		
·	✓ ProcedureResult		
	Column		
	ProcedureParameter		
	Index		
Ċ	?	Cancel OK	

Figure 5.36. Select Model Object Name Dialog

Next, select the model object in the **Extended Model Objects** section and use the actions and propertes table in the lower **Extension Properties** section to add/remove or edit your actual extended properties. Selecting the add or edit extension properties actions displays a dialog containing sections to edit general properties, value definition (required, masked, allowed values) as well as display name and description values which can be internationalized.

000	Edit Property Definition
Property Defin	ition Enter the property's general information, value info
▼ General	
Enter general prop	erty characteristics
Model Object:	BaseTable
Namespace Prefix:	mymodelextension
ID:	copyable
Runtime Type:	boolean
Should only be	modified by advanced users
🗹 Will be used by	Teiid server
 Value Definition 	-
	istics of the property value
A value is requi	
	erty value when shown to user
	be one of the following:
true false	
Use this initial	value
	an only be this value
	an only be this value
b. Display Nama	
Display Name	
Description	
Figure 5.37. Edit Prope	rty Definition Dialog

5.3.3. Extending Models With MEDs

MEDs must be applied to a model in order for their extension properties to be available to that model's model objects. To manage the applied MEDs for a specific model select the model and right-click select the **Modeling > Manage Model Extension Definitions** action. This will display a dialog listing the current applied MEDS and actions and buttons to add or remove MEDs from a model, extract a MED from a model and save a copy of it locally as a '.mxd' file and lastly, update the version of MED in a model if it differs from a version in your MED registry.

	00	Manage Model Extension Definitions
--	----	------------------------------------

Manage Model Extension Definitions

Add or Remove Model Extension Definitions to be associated with this model.

Location:	Test/					
Model:	View					
Current	Model Ex	tensions for the	selecte	d model:		_
Registere	d Different	Namespace Prefix	Version	Description		
						ſ
?					Cancel	
\odot						

Figure 5.38. Manage Model Extension Definitions Dialog

Selecting the Add button displays a list of applicable MEDS based on model class.

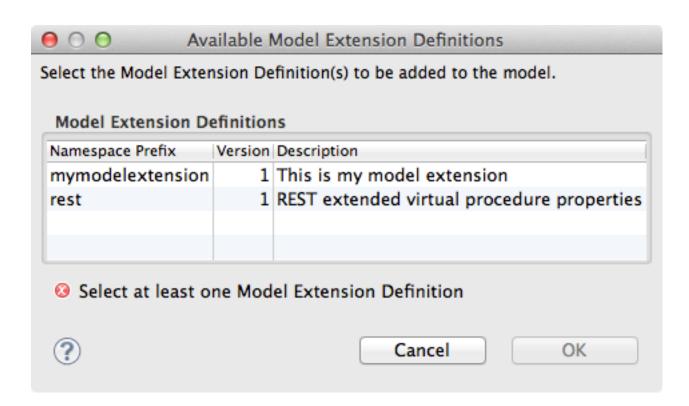


Figure 5.39. Add Model Extension Definitions Dialog



5.3.4. Setting Extended Property Values

Extension properties are user-defined properties available to any extended model object via the Properties View. As shown below, all extension properties are available up under the "Extension" category and are prefixed with a MED's namespace prefix. If there is an initial value for an extension property it will be set to the default value using the property definition found in the MED.

Properties 🛿 📄 Description	ᄩ 羚 💀 🛃 🌄 🗖
Property	Value
▼Extension	
mymodelextension:Copyable	true
▼Info	
Object URI	
▼Misc	
Cardinality	L1.1 O
Logical Relationships	
Materialized	™ false
Materialized Table	EE.
Name	LA NewBaseTable

Figure 5.40. Properties View For Extended Model Object

Metadata-specific Modeling

This chapter discusses various features targeted at defining and managing metamodel-specific objects.

6.1. Relational Source Modeling

6.1.1. Source Function

To improve ability to utilize database functions within View transformations, a new **New Child** > **Source Function** action and wizard was added to assist in building a source procedure that conforms to a function structure, including input and output parameters. Prior to Teiid Designer 7.5, these functions were required to be modeled as User Defined functions.

Running the action will display the **Create New Source Function** dialog. Enter your database function name, define input parameters including datatype and length, specify output parameter info, set options and click **OK**.

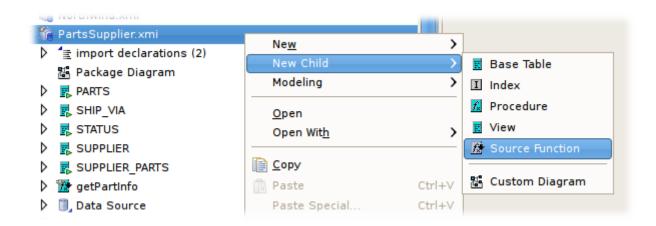


Figure 6.1. New Source Function Action

The resulting source function will be added to your model and will be represented by the

f‡

icon.

E	Create S	ource Function	X
Create	New Source Function		
Press Ok	(to finish		
Name_			
getEmp	loyeeInfo		
Input P	arameters		
Include		Data Type	Length
\checkmark	employeeID	integer	\$ 12
		string	\$ 255
Return	Parameter		
Name		Data Type	Length
employe	eeInfo	clob	\$ 65532
Proper	ties		
Name In			
Dete	rministic		
Descrip	otion		
?		Cancel	ОК

Figure 6.2. Create New Source Function Dialog

When finished, the new source function will be displayed in your model's package diagram.

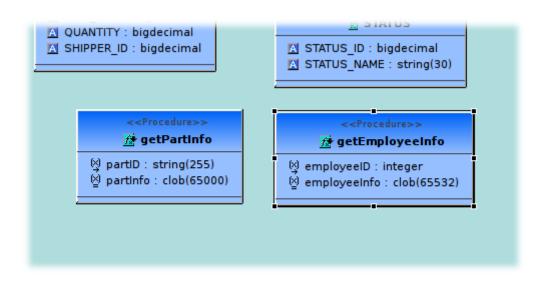


Figure 6.3. New Source Function In Package Diagram

After saving your model, your new source function will be available for use in your transformations. If you open the Expression builder, your source functions will be selectable in the Function dropdown selector for a Category named for the model as shown in the example below.

	Expression Builder		
nitial Expression	n: GETPRODUCTINFO(ProductsOracle.PRODUCT	DATA.INSTR_ID	
Tree View SQ	View		
🗢 🖌 GETPR	DDUCTINFO(ProductsOracle.PRODUCTDATA.INSTR	R ID)	
Pro	ductsOracle.PRODUCTDATA.INSTR_ID	_	
			Delete
			\square
	Expression Editors		
	·		
🔿 Attribut	e 🔿 Constant 🖲 Function		
Value is un	changed		
value is un	inangea.		
Category:	PRODUCTSORACLE	0	
Function:	GETPRODUCTINFO(PRODUCTID)		Apply
	S		Reset
Argument PRODU		INSTR ID	
PRODUC			
		Cancel	ок

Figure 6.4. Source Function In Expression Builder

6.1.2. Create Relational Table Wizard

In Teiid Designer 7.6 a new action, **New Child > Base Table...**, was added to aid in creating tables, complete with all of it's child entities. This action replaces the original **New Child > Base Table**, which only created the empty table.

lest		
Ne <u>w</u>	>	
New Child	\rightarrow	🗾 Base Table
Modeling	>	I Index
Open		🔏 Procedure
Open Wit <u>h</u>	>	🖬 View
<u>[]</u> <u>С</u> ору		🚼 Base Table
💼 Paste	Ctrl+V	뭡. Custom Diagram
Paste Special	Ctrl+V	Custom Diagram
Delete		

Figure 6.5. New Base Table Wizard Action

Running the action will display the **Create Relational Table** wizard. The wizard page contains 5 tabbed panels representing the various properties and components that make up the possible definition of a relational table. Enter your table name, define columns, keys, constraints and other options, then click **OK**.

This wizard is designed to provide feedback as to the completeness of the relational table information as well as the validation state of the table and it's components. Note that although errors may be displayed during editing, the wizard is designed to allow finishing with the construction of an incomplete table containing errors.

The first tab labeled **Properties** contains the input for the simple table properties including name, name in source, cardinality, supports update and is system table properties.

	🗾 Create Relational Table 🛛 🗶				
Create Relational	Create Relational Table				
🚯 No columns defined for table					
and foreign keys defin	ating a complete relational table including columns, unique keys ition.				
Properties Col	umns P Primary Key U Unique Constraint F Foreign Keys				
Model File	SimpleRelational.xmi				
Name	PartsInfo				
Name In Source	PartsInfo				
Cardinality	0				
-					
Supports Update					
Is System Table					
-					
(?)	Cancel OK				

Figure 6.6. Properties Tab

The **Columns** tab allows creation and editing of basic relational columns. This includes adding, deleting or moving columns as well as changing the name, datatype and length properties.

Create Relational Table Click OK to add new object to model This wizard allows creating a complete relational table including columns, unique and foreign keys definition. Primary Key Unique Constraint F Foreign Key Add Delete Move Up Move Down Column Name Data Type Length							
This wizard allows creating a complete relational table including columns, unique and foreign keys definition. Properties Columns Primary Key Unique Constraint F Foreign Key Add Delete Move Up Move Down Column Name Data Type Length							
and foreign keys definition. Properties Columns Primary Key Unique Constraint F Foreign Key Add Delete Move Up Move Down Column Name Data Type Length							
and foreign keys definition. Properties Columns Primary Key Unique Constraint F Foreign Key Add Delete Move Up Move Down Column Name Data Type Length							
and foreign keys definition. Properties Columns Primary Key Unique Constraint F Foreign Key Add Delete Move Up Move Down Column Name Data Type Length							
Properties Columns Primary Key Unique Constraint F Foreign Key Add Delete Move Up Move Down Column Name Data Type Length	keys						
Add Delete Move Up Move Down Column Name Data Type Length							
Column Name Data Type Length	ys						
Column Name Data Type Length							
ID string 10							
Iname string 10							
category string 10							
quantity string 10							
Cancel OK							

Figure 6.7. Columns Tab

The **Primary Key** tab allows editing of the name, name in source and column definitions. Note that un-checking the box will clear the data. The **Unique Constraint** tab contains the identical information.

		Create Relati	onal Table	×	
Create Relational Table Click OK to add new object to model					
This wizard allows creating a complete relational table including columns, unique keys and foreign keys definition.					
Properties 🖸	Columns	P Primary Key	📕 Unique Constraint	F Foreign Keys	
✓ Include					
Name	PK_PartInf	fo			
Name In Source	PK_PartInf	fo			
Change	Column N	ame			
	C ID				
?			Cancel	ОК	

Figure 6.8. Primary Key Tab

The Foreign Keys tab allows creating, editing and deleting multiple foreign keys.

Create Relational Ta	ble 🛛 🗙
Create Relational Table	
Click OK to add new object to model	
This wizard allows creating a complete relational tabl and foreign keys definition.	e including columns, unique keys
🔄 Properties 💽 Columns 🖻 Primary Key 🔟 Uniqu	ue Constraint F Foreign Keys
Add Foreign Key Name	
Edit Delete	
?	Cancel OK

Figure 6.9. Foreign Keys Tab

To create a new **Foreign Key**, select the **Add** button and enter/select the properties, key references in the tables shown below. Note the **Select Primary Key** or **Unique Constraint** table will display any PK/UC existing in the selected relational model. If no tables in that model contain a PK or UC, then the table will be empty.

	Create Foreign Key	×
Create Foreign Key Enter valid name, selec	/ ct a key or constraint, select column references and press OK	
Name	FK_Supplier	
Name In Source	FK_Supplier	
Foreign Key Multiplicity	ONE	•
Unique Key Multiplicity	ONE	0
Select Primary Key	or Unique Constraint	
 P Table_A: PK_A Table_A: UC_A P BBBB: PK_BB BBBBB: UC_BBB BBBBB: UC_BBB P TABLE_X: PK_A 		
Select Column Ref	erences To FK	
 Image name Image category Image quantity 		
?	Cancel	ок

Figure 6.10. Create Foreign Key Dialog

6.2. Relational View Modeling

This section contains descriptions of various features related to creating and managing relational view model objects.

6.2.1. Create Materialized Views

For any relational view table you can enable it's materialized view option by setting the **Materialized** property to TRUE and setting the **Materialized Table** reference, as shown in the figure below. Note that you are required to have already created your relational tables.

Pro	operty	Value	
⊳	Info		
~	Misc		
	Cardinality	L1.1 O	
		碱 true	If true, the table can be materialized (cached). Only applies
		🗾 PARTS (Pat	th=/PartsTest/MaterializedParts.xmi/MaterializedParts)
	Name	LA PARTS	
	Name In Source	LA PARTS	
	Supports Update	碱 false	
System		🔙 false	

Figure 6.11. Materialized Table Properties

Designer includes a feature to assist in quickly creating materialized relational tables based on your existing view tables.

To create materialized views:

• Step 1 - Right-click on one or more view tables in the Section C.2.1, "Model Explorer View" and select the Modeling > Create Materialized Views action.

E Packa	age Diagram		
PARTS	5		
🕨 🖡 SHIP			
👂 🛃 STAT	New Child	,	
	New Sibling	>	
	New Association	>	
D 🛃 SUPF	Modeling	>	🚰 Create Materialized Views
Product			
👕 Salesforce	💛 Undo Delete multiple objects		Create Web Service
👕 SearchProj			踞 Add To New Custom Diagram
SF_Testing	-		
👕 TestRels	of Cut	Ctrl+X	🏠 Add to Metadata Favorites
· -•	Сору	Ctrl+C	
and manual M	Paste	Ctrl+V	

• Step 2 - In the Create Materialized View Model dialog specify or select a target relational model for your generated relational tables.

	Create Materialized View Model	×
Create Materialize	d View Model	
All inputs OK. Press Fir	nish to create materialized view tables.	
Materialization Model	MaterializedParts	
Target Location	MaterializedParts	
Selected Virtual	Tables	
PARTS		
SHIP_VIA		
?	Cancel	Finish
\sim		

• Step 2a - Selecting the browse '...' button displays the **Relational Model Selector dialog** where you select an existing relational model or specify a unique name for a new model.

	Relational Mo	odel Selector Dialog		×
Select the Relationa	l Model or schema locat	ion within a model in v	which the entitie	s will be created.
🗢 🛃 PartsTest				
Northwind.>	mi			
👂 📬 PartsSuppli	er.xmi			
🗍 PartsTestVD)B.vdb			
PartsViewM	odel.xmi			
ProductsOr	acle.xmi			
New Model Name:	MaterializedParts			
Click OK to create the	relational model			
?		Properties	Cancel	ОК

• Step 3 - Click OK to create relational tables corresponding to your selected view tables and automatically set the Materialized property to TRUE and the Materialized Table reference value to your newly generated table.

When finished your view tables will be configured with their new materialized properties and the corresponding relational tables will be shown in their package diagram.

👘 Par	tsViewModel.xmi 🛛 😭 Materializ	edParts.xmi 🛛	
୍ଚ 1	< <base table=""/> >	< <base table=""/> >	<u>^</u>
14 14 14	 PART_ID : string(50) PART_NAME : string(255) PART_COLOR : string(30) PART_WEIGHT : string(255) 	SHIPPER_ID : bigdecimal SHIPPER_NAME : string(30)	
	<u>د</u>		 >)
뭡 Pa	ackage Diagram 🥅 Table Editor		

Figure 6.12. Materialized Table Properties

6.2.2. Create Relational View Table Wizard

In Teiid Designer 7.7 a new action, **New Child > Base Table...**, was added to aid in creating relational view tables, complete with its target columns and SQL transformation. This action replaces the original **New Child > Base Table** action, which only created the empty table.

I Index
Procedure
E View
Base Table
Ctrl+V

Figure 6.13. New View Table Wizard Action

Running the action will display the **Create Relational View Table** wizard. The wizard page contains 3 tabbed panels representing the various properties and components that make up the

possible definition of a relational view table. Enter your table name, define the desired columns and specify the SQL transformation, then click **OK**.

This wizard is designed to provide feedback as to the completeness of the relational view table information, as well as the validation state of the table and it's components. Note that although errors or warnings may be displayed during editing, the wizard is designed to allow finishing even if the table definition is incomplete.

The first tab labeled **Properties** contains the input for the simple table properties including name and supports update.

	Create Relational View Table			
Create Relational View Table				
ANo columns defined	d for table			
This wizard allows cre	ating a complete relational view table including SQL definition			
	ating a complete relational view table including SQL definition			
Properties Colu	mns SQL			
Model File	vModel.xmi			
Name	myTable			
✓ Supports Update				
B Supports Opdate				
L				
?	Cancel OK			

Figure 6.14. Properties Tab

The **Columns** tab allows creation and editing of basic relational columns. This includes adding, deleting or moving columns as well as changing the name, datatype and length properties.

Create Relational View Table	
Create Relational View Table	
ANO columns defined for table	
This wizard allows creating a complete relational view table including SQL definition	n.
Properties Columns SQL	
Add Delete Move Up Move Down	
Column Name Data Type Length	
Cancel OK	

Figure 6.15. Columns Tab

The **SQL** tab allows editing of the SQL Transformation for the relational view. The desired SQL can be entered directly into the text area or a SQL Template may be selected by clicking the **Templates...** button.

Create Relational View Table	
Create Relational View Table	
ANO columns defined for table	
This wizard allows creating a complete relational view table including SQL definition.	
Start with a SQL Template or enter the desired SQL	
Templates	
Transformation SQL	
SELECT [TABLEA.COL1], [TABLEA.COL2], [TABLEB.COL1] FROM [TABLEA], [TABLEB] WHERE [TABLEA.COL1] = [TABLEB.COL1]	
? Cancel OK	

Figure 6.16. SQL Tab

If the **Templates...** button is selected on the SQL tab, the **Choose a SQL Template** dialog will display, as shown below.

E Choose a SQL Tem	nplate 🗆 🛛 🗙
Choose a SQL Template	
Choose a SQL Template from the available option	s. 👘 🏹
Template Options	
SELECT Queries PROCEDURES	
Simple SELECT	
 SELECT with Join Criteria 	
O UNION Query	
O Flat File - Local Source	
O XML File - Local Source	
O XML File - URL Source	
SQL Template	
SELECT * FROM [TABLEA]	
	=
?	Cancel OK

Figure 6.17. SQL Templates Dialog

A number of common SQL templates may be chosen using the dialog. Depending on usage context, the PROCEDURES tab may also be available in addition to the SELECT queries tab. The SQL Template dialog can also be accessed in the Transformation Editor, from the rt-click context menu.

6.3. XML Document Modeling

6.3.1. Create XML View Documents from schema

You can create XML View Documents by selecting an element in the Model Explorer and selecting the **Modeling > Create XML View Documents** action.

V 🔩 BOOKDatatypes.xsc			
▽ 🐻 Books.xsd			
S Books.xsd			
- BookDatatyp	es.xsd		
🕕 MetaMatrix 🛙	ata Set Library Version 1.1. Boo		
e bibliography	: BooksNS'Bibliography		1
e bookListing	New Child	>	
e bookCollecti	New Sibling	>	
e bookSetMixe	New Association	>	
Bibliography	Modeling	\rightarrow	Create Scalar To XML Tr
Bilbiography			Create XML To Scalar Tr
BookSetFlat	Undo Set org.ecii) to FK_Account_Accountid		Create XML View Docum
BookFlat	😂 Redo		
BookSetNes			☆ Add to Metadata Favorit
BookNested		Ctrl+X	
V ⊡ BookNested	Copy	Ctrl+C	

Figure 6.18. Create XML View Documents Action

The action will query you for a target XML Document model. You can either select an existing XML Document model from your workspace, or enter a unique model name and the wizard will create a new model for you.

Select or create XML View model	×
▼ 🛃 BooksXMLTest	<u></u>
BookDatatypes.xsd	
Books.xsd	
Booksinput.xsd	
BooksXML.xmi	
Northwind.xmi	
PartsSupplier.xmi	
PartsTestVDB.vdb	
ProductsOracle.xmi	
New Model Name:	
😣 Enter name and click OK to create a new XML View Model in fold	er "BooksXMLTest"
? Properties	Cancel OK

Figure 6.19. Select or Create XML View Model Dialog

After selecting or creating your new XML Document model, the XML Document builder page will be displayed. This page is explained in greater detail in *Section 3.3.2, "Build XML Documents From XML Schema*" section.

	Build XML D	ocuments From XML Schema
Select XML Sc		
		ements. Or press Finish to create an empty XML Document.
	BooksXMLTest/Books.xsd	
Select one or mo Schema Root Ele	ore Schema Root Elements to gene	rate Virtual Documents Virtual Documents
e bookCollectio e bookListing :	on : BooksNS:BookSetNested BooksNS:BookSetFlat d : BooksNS:BookSetMixed	<pre> bibliography : BooksNS:Bibliography Contents Content</pre>
Reset		Keep sorted alphabetically
Document Option	1S	
	ual documents from the schema	
		e schema where many portions will not be used)
Mapping Options	-	
Build Mappin		
Mapping Class	-	
 Locate at 	XML Schema iteration nodes (max	occurs > 1) - for schema-compliant data access
🔿 Locate at	XML Schema complex types - for s	emantic use only
Mapping Class	s Column Datatypes	
Use XML 1	ypes from the document	
🔘 Use "Strin	ng" mapping type (disables automa	atic output conversion to closest built-in XSD type)
?		< <u>Back</u> <u>N</u> ext > Cancel <u>Finis</u>

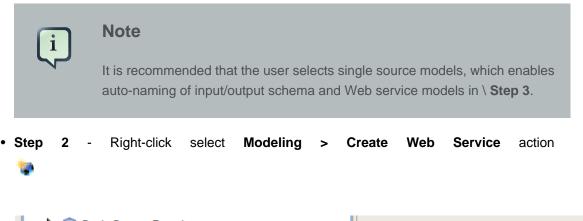
Figure 6.20. Build XML Documents From XML Schema Dialog

6.4. Web Services Modeling

6.4.1. Create Web Service Action

This method is recommended for experienced users for consistent and rapid deployment of **Web** services designed to query relational sources. It provides detailed control of all **Web** service interfaces, operations and required transformations from **XML Views**

- To create a Web service model from relational models or objects:
 - Step 1 Select any combination of relational models, tables and/or procedures in the Section C.2.1, "Model Explorer View" tree.



PartsSour	ceB.xmi		
🔰 🚺 PartsVirtu	al vezi		1
	Ne <u>w</u>	>	
	New Child	>	
	Modeling	\rightarrow	🐞 Create Web Service
	0.0.0.0		Generate SQL Relationships
	<u>O</u> pen		Transformations - Export SQL
	Open Wit <u>h</u>	>	Transformations - Import SQL
🎲 Teiid 🔀	Copy		

Figure 6.21. Create Web Service Action

• Step 3 - In the Create Web Service dialog, specify file names for the generated Input Schema file, Output Schema file and Web service model. Change options as desired. Click Finish when done.

E	Create Web Service 🗙
Create	Web Service
Click Fin	ish to generate.
Locat	ion
Specify	location for generated model(s) (i.e. Project or Folder)
TestW	S Browse
Outpu	ıt Schema Model
Name:	Parts_O_Output
	☑ Generate flat XSD
Input	Schema Model
Ger	nerate
Name:	Parts_O_Input
	Generate flat XSD
Web 9	Service Model
Name:	Parts_O_WS
	Generate default SQL for XML mappings
?	Cancel <u>F</u> inish

Figure 6.22. Create Web Service Dialog

• Step 4 - When model generation is complete, a confirmation dialog should appear. Click OK.

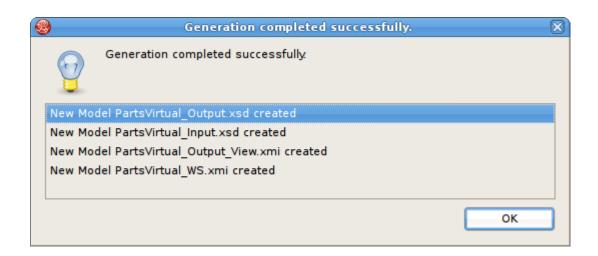


Figure 6.23. Generation Completed Dialog

6.4.2. Web Services War Generation

Teiid Designer allows you to expose your VDBs via a SOAP or REST interface. JBossWS-CXF or RESTEasy wars can be generated based on models within your VDBs. This section describes these wizards in detail.

6.4.2.1. Generating a JBossWS-CXF War

The Teiid Designer provides web service generation capabilities in the form of a JBossWS-CXF war. Once you have added your **Web Service Models** as described in *Section 3.5, "Creating Web Service View Model"* to your **VDB**, deployed the **VDB** to a running Teiid instance and created your VDB's data source, you are ready to expose the web service using the generated war.

- To generate a new JBossWS-CXF war using the VDB:
 - Step 1 Right-click on the VDB containing your web service model(s) and select the Modeling > Generate JBossWS-CXF War action.

MarketDataView_W	Ne <u>w</u>	••••••••••••••••••••••••••••••••••••••	tput.xsd	/Portfolic
PortfolioVDB.vdb	Modeling	>	Deploy	
RestfulBooks	Open		Execute VDB	
 A UndateableBooksWeb 	Open With	>	Generate JBossWS	-CXF War

• Step 2 - Fill in missing properties in Web Service War Generation Wizard shown below.

000	Create Web Service WAR
Create a WAR file to deploy	as a Web Service
Enter the required informa	ation, then click OK to create the WAR file.
WAR Creation Information	
Context Name:	PortfolioVDB
Web Server Host:	localhost
Web Server Port:	8080
VDB JNDI Name:	PortfolioVDB
Security	
When using HTTPBasic	security, a local Teiid connection is required u
 None 	
○ HTTPBasic	
OWS-Security (Usernam	e Token)
HTTPBasic Options	
Realm:	
Role:	
WS-Security Options	
Username:	
Password:	
General Options	
Enable MTOM	
Target namespace:	http://teiid.org
WAR File Save Location:	/Users/macuser/WARFiles
Change	

?

Field Name	Description		
Name	The name of the generated war file.		
Host	The server host name (or IP).		
Port	The server port.		
VDB JNDI Name	The JNDI connection name to the deployed Teild source VDB.		
Security options	• None - no username/password required to connect to the VDB through the generated web service.		
	 HTTP Basic - the specified security realm and role will be used. The default realm value is the realm that comes out-of-the-box with Teiid (teiid-security). The role needs to be defined in the appropriate security mechanism. In the case of Teiid, use the teiid-security-roles.properties file. When using HTTPBasic, a local Teiid connection using the PassthroughAuthentication property is required. See the Teiid user's manual for details on PassthroughAuthentication. 		
	• WS-Security - a password callback class will be generated for you which will validate that the username/password values you specified in the war generator dialog are passed in. This is meant to be a testing mechanism for your WS-Security enabled web service and your own security mechanism should be implemented in this class. All source code is included in the generated war along with the compiled class files.		
Target namespace	This is the target namespace that will be used in the generated WSDL and subsequent generated web service classes.		
MTOM (Message Transmission Optimization Mechanism)	If selected, MTOM will be enabled for the web service endpoint(s). You will also need to update your output schema accordingly		

Field Name	Description
	by adding the xmlns:xmime="http:// www.w3.org/2005/05/xmlmim" schema and adding type="xs:base64Binary" xmime:expectedContentTypes="application/ octet-stream" to the output element you wish to optimize.
War File Save Location	The folder where the generated WAR file should be saved.

• Step 3 - Click OK to generate the web service war. When war generation is complete, a confirmation dialog should appear. Click OK.

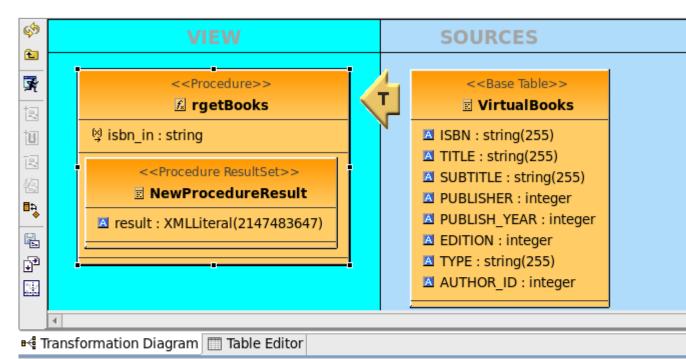


Figure 6.25. Generation Completed Dialog

6.4.2.2. Generating a RESTEasy War

In Teiid Designer, it is also possible to expose your VDBs over REST using a generated RESTEasy war. Also, if your target virtual model has update, insert and delete SQL defined, you can easily provide CRUD capabilities via REST. Accepted inputs into the generated REST operations are URI path parameters and/or XML/JSON. JSON is exposed over a URI that includes "json". For example, "http://{host}:{port}/{war_context}/{model_name}/resource" will accept URI path parameters and/or XML while "http://{host}:{port}/{war_context}/{model_name}/json/resource" will accept URI path parameters and/or JSON.

• Step 1 - In a virtual model, add a procedure(s) that returns an XMLLiteral object. The target of your procedure can be any models in your VDB. Here is an example procedure that selects from a virtual table (VirtualBooks) and returns the results as an XMLLiteral:



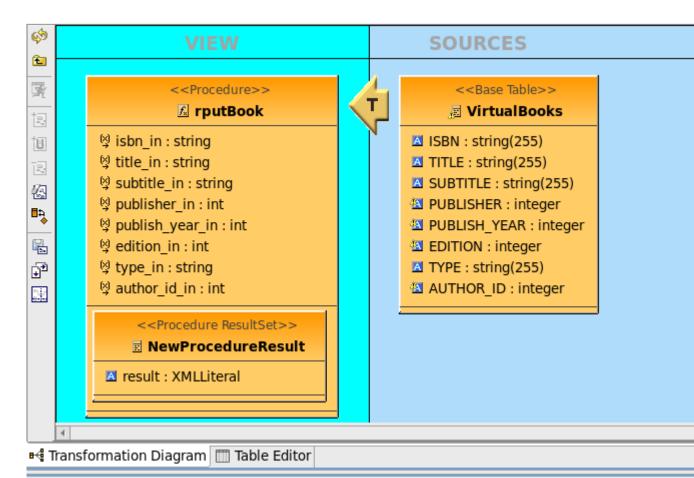
Transformation Editor

CREATE VIRTUAL PROCEDURE BEGIN SELECT XMLELEMENT(NAME books, XMLAGG(XMLELEMENT(NAME book, XMLFOREST(RestfulBooks RestfulBooks.VirtualBooks.SUBTITLE, RestfulBooks.VirtualBooks.PUBLISHER, RestfulBooks.VirtualBooks RestfulBooks.VirtualBooks.TYPE, RestfulBooks.VirtualBooks.AUTHOR_ID)))) AS result FROM RestfulBooks RestfulBooks.VirtualBooks.ISBN; END

7

Notice the syntax used to convert the relation table result of the select from VirtualBooks, to an XMLLiteral. All XML functions are documented in the **Scalar Functions** chapter of the **Teiid Reference Guide**.

Here is an example of an update procedure that will insert a row and return an XMLLiteral object:



Transformation Editor

CREATE VIRTUAL PROCEDURE
BEGIN
DECLARE integer VARIABLES.insert_count = 0;
INSERT INTO RestfulBooks.VirtualBooks (RestfulBooks.VirtualBooks.ISBN, RestfulBooks.VirtualBook
RestfulBooks.VirtualBooks.PUBLISH_YEAR, RestfulBooks.VirtualBooks.EDITION, RestfulBooks.VirtualB
RestfulBooks.rputBook.title_in, RestfulBooks.rputBook.subtitle_in, RestfulBooks.rputBook.publisher_i
RestfulBooks.rputBook.type_in, RestfulBooks.rputBook.author_id_in);
VARIABLES.insert_count = VARIABLES.ROWCOUNT;
IF(VARIABLES.insert_count = 1)
BEGIN
SELECT XMLCOMMENT('Insert was successful!') AS result;
END
ELSE
BEGIN
SELECT XMLCOMMENT('Insert failed!') AS result;
END
IEND

The input format for the REST procedure could be URI parameters, an XML/JSON document, or some combination of both. When using an XML document your root node should be **<input>** and the XML nodes should correspond to order of the procedure's input parameters. For example, here is the input for the above insert procedure:

```
<input>
<ISBN>0-13-014714-1-999999</ISBN>
<TITLE>The XML Handbook</TITLE>
<SUBTITLE>Updated Edition</SUBTITLE>
<PUBLISHER>16</PUBLISHER>
<PUBLISH_YEAR>2000</PUBLISH_YEAR>
<EDITION>2</EDITION>
<TYPE>Hardback</TYPE>
<AUTHOR_ID>49</AUTHOR_ID>
</input>
```

Figure 6.26. Sample XML Input

When using a JSON document, your values should match the order of your procedure input parameters as well. Here is the input for the above insert procedure:

```
{
"ISBN": "1-55615-484-4",
"TITLE": "Code Complete",
"SUBTITLE": "A Practical Handbook of Software Construction",
"PUBLISHER": 5,
"PUBLISH_YEAR": 1993,
"EDITION": 1,
"TYPE": "Hardback",
"AUTHOR_ID": 31
}
```

Figure 6.27. Sample JSON Input

• Step 2 - Now we need to identify our procedure as REST eligible. To do this we add enable REST properties for the procedure(s) via the **Modeling->Enable** context menu option.

New Child	Þ	< <procedure resultset="">></procedure>
New Sibling	•	GetCustomerProcResul
New Association	•	esult : XMLLiteral
Modeling	•	🐄 Create Web Service
Undo Disable REST for Procedures	^Z	Enable REST for Pro
> Redo	^ Y	Preview Data for Ge

This will enable two new properties in the property tab for all procedures defined in the model. The two required properties are defined in the table below:

Table 6.2. Required Extended Properties for RESTful Procedures

Property Name	Description			
Rest Method	The HTTP method that will determine the REST mapping of this procedure. Supported methods are: GET, PUT, POST and DELETE			
URI	The resource path to the procedure. For example, if you use "books/ {isbn}" as your URI value for a procedure, http://{host}:{port}/{war_context}/ {model_name}/books/123 would execute this procedure and pass 123 in as a parameter.			

Here's what the above example would look like in the Property tab:

Properties	🖹 De	escription	ł		đ	$\overline{}$	
Property		Value					
▼Extension							
rest:Rest Method		GET					
rest:URI		📄 books/	{isbn	ı}			

Note that the generated URI will have the model name included as part of the path, so full URL would look like this: http://{host}:{port}/{war_context}/{model_name}/books/123. If you wanted a REST service to return all books, you would write your procedure just as it is above, but remove the input parameter. The URI property would then just be 'books' (or whatever you want) and the URL would be http://{host}:{port}/{war_context}/{model_name}/books.

Once you have added all of your procedures along with the required extended properties, be sure and add the model to your VDB or synchronize if it's already included in the VDB. You will then need to re-deploy the VDB.

Important

If you redeploy your VDB during development, you may receive an "Invalid Session Exception" due to a stale connection obtained for the pool. This can be corrected by flushing the data source or, alternatively, you could add a test query to your VDB connection's -ds.xml file. This will insure you get a valid connection after redeploying your VDB. The syntax for the test query is as follows: <checkvalid-connection-sql>some arbitrary sql</check-valid-connection-sql>"

- Step 3 3. If you have not already done so, you will need to create a data source for your VDB. This can be done in the Teiid View of Designer. Right-click on your deployed VDB and select Create Data Source. The Generate REST WAR dialog will ask you for the JNDI name for your created source so that it can connect to your VDB.
- Step 4 Right-click on the VDB containing your virtual model(s) with REST eligible procedures and select the Modeling > Generate RESTEasy War action. If there are no procedures that are REST eligible, the "Generate RESTEasy War" option will not be enabled.

📑 Bor	Jeon Contradio	BOOKSKESTW
_	Ne <u>w</u>	bksToo.xmi /BooksRESTW
X M	Modeling	🕨 🙀 Deploy
📑 m 🗋	Open	🖳 Execute VDB
🔸 竊 M	Open With	😭 Generate JBossWS-CXF War
) 😘 Re 🕽		📆 Generate RESTEasy War
_		

• Step 5 - Fill in missing properties in the REST War Generation Wizard shown below.

▼ Create REST WAR File			
Create a REST WAR file to d	8		
Enter the required information,	٦		
REST WAR Creation Inform	ation		
Context Name:	BooksREST		
VDB JNDI Name:	{REPLACE_WITH_VDB_JNDI_NAM		
REST WAR File Save Location:			
Change			
Include RESTEasy Jars in lib	Folder of WAR		Restore Default
?		Cancel	ОК

Figure 6.28. Generate a REST WAR War File Dialog

Field Name	Description
Name	The name of the generated war file.
Connection JNDI Name	The JNDI connection name to the deployed Teild source VDB.
War File Save Location	The folder where the generated WAR file should be saved.
Include RESTEasy Jars in lib Folder of WAR	If selected, the RESTEasy jars and there dependent jars will be included in the lib foled of the generated WAR. If not selected, the jars will not be included. This should be de- selected in environments where RESTEasy is installed in the classpath of the server installation to avoid conflicts.

Table 6.3. Field Descriptions

• Step 6 - Click OK to generate the REST war. When war generation is complete, a confirmation dialog should appear. Click OK.

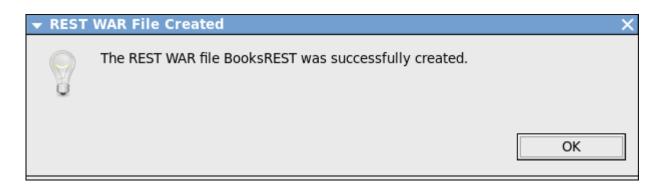


Figure 6.29. Generation Completed Dialog

6.4.2.3. Deploying Your Generated WAR File

Once you have generated your war file, you will need to deploy it to your JBoss AS instance. There are a few ways to accomplish this.

• From JBDS or JBoss Tools

1. Insure target JBossAS is configured and running.

2. Select your WAR file in the Model Explorer view. If you didn't generate your war to that location, you can copy and paste it there.

3. Right-click on the WAR file and select 'Mark as Deployable'. This will cause you WAR file to be automatically deployed the JBoss AS instance you have defined.

マ 🛃 SoapyBooksProje	ct	
🔋 BooksWS.vdb		Spramic Web Project
📄 BooksWS.war	Ne <u>w</u>	🖘 Java EE Web Dreiget
👂 💱 MySqlBooks_C		
🕨 🔹 MySqlBooks_C	Open Wit <u>h</u>	•
👂 🕵 MySqlBooks_V	<u>С</u> ору	
👂 痛 MySqlBooks.xı	Paste	Ctrl+V
👂 🔂 test	<u>D</u> elete	
IS_States	Refactor	•
	🚵 Import	
	🖾 Exp <u>o</u> rt	
	🔊 Re <u>f</u> resh	
	👈 Mark as Deploy	able
	<u>V</u> alidate	

• Using the JBoss AS Administration Console

Using the administration console that comes with JBoss AS, you can deploy WAR files. The administration console is available at http://{host:port}/admin-console. Once logged on, simply use the "Add a New Resource" button of the "Web Application (WAR)" resource folder.

Manual Deployment to JBossAS

It is possible to deploy the generated WAR by manually copying the file to the "deploy" folder of the target JBoss AS. If the server is running, the WAR will deploy automaticall via "hot" deploy. Otherwise, the WAR will deploy at the next start of the server.

6.4.2.4. Testing Your Generated WAR Files

Once you have deployed your war file, you are ready to test it out. There are a few ways to accomplish this.

SOAP WAR Testing

• Determining Your WSDL URL

You can get your WSDL URL at http://{server:port}/jbossws/services. This is where all the deployed web services for the target JBossAS server will be listed. Find your service and click the Endpoint Address link. This will retrieve your web service's WSDL and the WSDL URL address will appear in the browser's address bar.

Endpoint Name	idpoint Name jboss.ws:context=books,endpoint=MySqlBooks_BOOKS				
Endpoint Address	http://127.0.0.1:8080/boo	ks/MySqlBooks_BOOKS			
StartTime		StopTime			
Mon Jun 11 15:16:02 CDT 2012					
RequestCount		ResponseCount	FaultCount		
0		0	0		
MinProcessingTim	e	MaxProcessingTime	AvgProcessingTime		
0		0	0		

Now that you have your WSDL URL, you can use any SOAP testing tool such as the Web Service Tester that comes with JBDS and JBoss Tools or an external tool like soapUI.

• Using the JBoss AS Administration Console

Using the administration console that comes with JBoss AS, you can deploy WAR files. The administration console is available at http://{host:port}/admin-console. Once logged on, s simply use the "Add a New Resource" button of the "Web Application (WAR)" resource folder.

REST WAR Testing

• What is my URI?

When you modeled your REST procedures, you assigned a URI for each HTTP Operation you defined along with the corresponding operation (GET, PUT, POST or DELETE). The full path of each URI is defined as /{war_context}/{model_name}/{resource} for XML input/output and / {war_context}/{model_name}/json/{resource} for JSON input/output.

Using your REST URL, you can use any testing tool with REST support such as the Web Service Tester included with JBDS and JBoss Tools or an external tool like soapUI or cURL.

Editing Models and Projects

Teiid Designer offers three basic model edit actions: **Rename**, **Move** and **Save As...** and one project-related action, **Clone Project**. These actions are described below.

7.1. Rename A Model

- To rename a model in your workspace:
 - Step 1 Select a model in the Section C.2.1, "Model Explorer View".
 - Step 2 Right-click select the Refactor > Rename action.

▽ <table-cell-rows> AllCustomerAccou</table-cell-rows>	ints		
 Morthwind.xmi BooksProject PartsProject WSDLImportProje 	Ne <u>w</u> New Child Modeling	>	
	<u>O</u> pen Open Wit <u>h</u>	>	
	Copy	Chillin V	
	🛅 Paste	Ctrl+V	
	Paste Special	Ctrl+V	
	<u>D</u> elete		
	Refactor	>)	Undo
	占 Import		Redo
	🛃 Exp <u>o</u> rt		Rename
	🔊 Re <u>f</u> resh		Namespace URI Move

Figure 7.1. Refactor Rename Action In Model Explorer

• Step 3 - Specify unique model name in the Rename Model File dialog. Click OK.

	Rename Model File	
Enter new name:	AllCustomers	
?	Cancel OK	

Figure 7.2. Rename Model File Dialog



Note

Renaming a model that is a dependency to another model will automatically change the model imports for those models. If source model CustomerSource is renamed to OldCustomerSource, for instance, the import statement for the view model CustomerAccounts which imports CustomerSource will be changed to reflect the new name.

7.2. Move Model

- To move a model in your workspace:
 - Step 1 Select a model in the Section C.2.1, "Model Explorer View".
 - Step 2 Right-click select the Refactor > Move action.

	S		
 Morthwind.xmi BooksProject PartsProject WSDLImportProject 	Ne <u>w</u> New Child Modeling	> > >	
V wsbLimportProject	<u>O</u> pen Open Wit <u>h</u>	>	
	<u>C</u> opy Paste Paste Special	Ctrl+V Ctrl+V	
	Delete Refactor <u>Import</u>	<u> </u>	Undo Redo
	🖄 Exp <u>o</u> rt 🔊 Re <u>f</u> resh		Rename Namespace URI Move

Figure 7.3. Refactor Move Action In Model Explorer

• Step 3 - Select a new location (i.e. Project or Folder) and click OK.

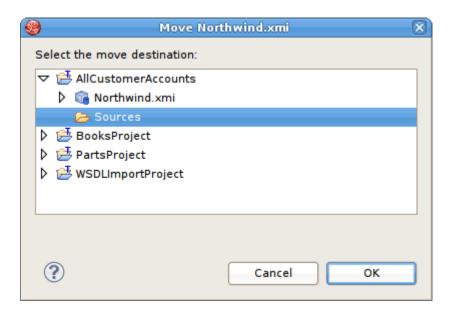


Figure 7.4. Move Model Dialog

7.3. Save Copy of Model

The **Save As...** action performs a similar function as the **Refactor > Rename** action except the renamed model is a structural copy of the original model.

Note

Each model object maintains it's own unique ID, so copying a model will result in a exact structural copy of your original model but with re-generated unique object IDs. Be aware that locating and copying your models via your local file system may result in runtime errors within Designer. Each model is expected to be unique and duplicate models are not permitted.

- To create a duplicate model using Save As...:
 - Step 1 Open the model you wish to copy in a **Model Editor** by double-clicking the model in Section C.2.1, "Model Explorer View" or right-click select **Open** action.
 - Step 2 Select the editor tab for the model you opened.



Figure 7.5. Select Editor Tab

• Step 3 - Select File > Save As... action to open the Save Model As dialog.

9	Save Model As 🛛 🕅					
Se	Select a destination for the copy					
	Northwind.xmi					
	🔁 Sources					
⊳	🖻 🚰 BooksProject					
	PartsProject					
⊳	WSDLImportProject					
_	er the new model name					
Nor	rthwind_West					
Ready to copy						
(Cancel OK					

Figure 7.6. Save Model As Dialog

- Step 4 Enter a unique model name in the new model name text field and click OK.
- Step 5 If dependent models are detected, the Save Model As Import References dialog is presented to give you the opportunity to change any of the dependent models imports to reference the new model or not.

🧐 🛛 Save Model As - Import References 🛛 🛞				
The following model files use or reference the model to be copied.				
Select any that should be modified to use the new copy.				
AllCustomerAccounts/CustomerViews.xmi				
Select All Deselect All				
Cancel OK				

Figure 7.7. Save Model As Dialog

7.4. Clone Project

Because each instance of of a model contains a unique ID and each object in each model contains a unique ID, copying a project is a delicate task. For this reason, the Clone Project action was created to manage the creation of exact structural copies of all models in the source project.

- The following lists specific rules and limitations for this action.
 - This action clones a complete model project containing any number of model (XMI or XSD) files organized in a user-defined directory structure.
 - All object references (UUIDs) within the original project will be replaced with new unique references.
 - Any model dependencies or internal object references are refactored to reflect the dependencies within the cloned project.
 - Any model references to models in projects external to the original project will NOT be replaced.
 - Only XMI and XSD files are cloned. All other file types in your project will NOT be processed nor copied into your newly cloned project including VDBs
 - If one or more editors that require "save" are open, the user will be asked to save them before continuing with the cloning process.
- To clone a model project::

- Step 1 Select an existing model project in the Section C.2.1, "Model Explorer View".
- Step 2 Right-click, then select Model Project > Clone in the context menu. Otherwise you can select the Project > Clone Project action, located in the Teiid Designer's main menu bar.

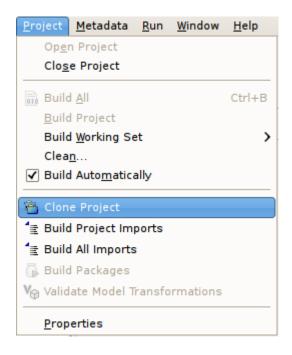


Figure 7.8. Clone Project In Project Menu

• Step 3 - On the Clone Project wizard page, provide a name for your new project.

🥮 Clone Model Project 🛛 🕅					
	Clone Model Project Specify cloned model project name.				
Project name: ValuedCustomerAccounts					
✓ Use <u>d</u> efault	t location				
	ne/blafond/Test Designer Folder/T se file system: default \$	B <u>r</u> owse			
?	Cancel	<u>Finish</u>			

Figure 7.9. Clone Project In Project Menu

- **Step 4** (Optional) If you wish to create your cloned project in a location other than your default workspace location, uncheck the **Use default location** check-box and specify (type in or browse to) a new directory location on your local file system.
- Step 5 Click Finish to generate your new project.

Managing VDBs

As stated in the introduction, the critical artifact for Teiid Designer the VDB, or Virtual DataBase. This section describes the details of how to create, edit, deploy and test your VDBs.

8.1. Creating a VDB

To create an empty VDB launch Eclipse's **New** wizard, open the **Teiid Designer** category folder and select **Teiid VDB**. You can also select one or more models in a model project, right-click and select **New > Teiid VDB** action.>

Launching this wizard will open the New VDB dialog. If you launched with one or more models selected the dialog will contain the pre-selected models for inclusion in the new VDB.

	New VDB	×
New VDB		
Press the "Fi	nish" button to finish.	
In Folder:	Books Project	
VDB Name:	Books	
Description	on	
Join of tw	vo books sources	<u>_</u>
		-
Selected	Models]
Add	🙃 BooksView.xmi	
	Books_SourceA.xmi	
Remove	Books_SourceB.xmi	
\odot	Cancel	Finish

Figure 8.1. New VDB Dialog

Note that a VDB is scoped to be aware of models and files within the same model project as the VDB. You will not be allowed to add models to a VDB that exist in a different project.

8.2. Editing a VDB

To Edit an existing VDB, select the VDB in the explorer and right-click select Open action or simply double-click the VDB. The VDB will be opened in a VDB Editor. (See the Section C.3.2, "VDB Editor" section)

8.3. Test a VDB

For details on how to test your VDB, see Section 9.3, "Testing With Your VDB" section

8.4. Reusing VDBs

Teiid 8.1 introduced the ability to treat your deployed VDB as just another database where the database category is your VDB name and each visible model in your VDB is treated as a schema. This is accomplished via a new <import-vdb> element in the vdb.xml definition. (see *Teiid 8.2 VDB Reuse section* [https://docs.jboss.org/author/display/TEIID/VDB+Reuse]). By allowing VDB's to referenced other VDBs, users can create reusable database components and reduce the amount of modeling required to create complex transformations.

The sample vdb.xml file below highlights the <import-vdb> element and the corresponding import-vdb-reference within the view model's <model> element.

Teiid Designer exposes this capability by allowing users to import metadata from deployed VDBs via the JDBC Import option. Through this import, relational VDB source models are created which structurally represent the Catalog (VDB), Schema (Model) and Tables in Virtual DataBase.

When dealing with the these VDB source models there are some limitations or rules, namely:

- VDB source models are read-only
- VDB source model name is determined by the deployed model name (schema) from the VDB it was imported from
- Model names have to be unique within a model project
 - VDB source models have to be imported/created in a project different than the project used to create and deploy the Reuse VDB
- The JDBC Import Wizard will restrict your options to comply with these rules

To create a VDB source model:

- Step 5 Select the model builder labeled Build XML documents from XML schema and click Next >. The Select XML Schema dialog will be displayed.
- Step 1 Deploy your VDB
- Step 2 Launch the JDBC Import Wizard via the "Import > Teiid Designer > JDBC Database >> Source Model" action
- Step 3 On the first page of the wizard create/select a valid connection profile for your deployed VDB.
 - The wizard will detect that the connection profile is a Teiid VDB connection and a section will be displayed on the wizard page titled **Teiid VDB Source Options**
 - If **Import as VDB source model** is NOT checked, then the wizard will continue importing as a normal JDBC import

-						
	🗶 🗉 Import Data	base via JDBC				
1	Select a JDBC source configuration					
	Press the "Next >" button to continue or the "Finish" button to finish.					
	Connection P	rofile				
Barry_Test_VDB						
	JDBC Metadat					
	JDBC (defau					
	JDBC (delad					
	Properties					
l	Driver:	Teiid 8.x				
1	URL:	jdbc:teiid:BarryParts_VDB@mm://localhost:31000				
	User Name:	user				
	Password:	****				
	Teiid VDB Sou	urce Options				
		VDB source model				
	A JDBC connection to a deployed VDB allows creating a read-only VDB source model that you can use in your view transform					
-		ing these view models will reference an import to the original deployed VDB and will not contain the metadata				
1	source mode					
1	If checked, o	nly one schema in your VDB can be selected for import at a time.				
A						
n						
	?	< Back Next >				

- Step 4 On the 3rd page, titled Select Database Objects, select a single schema to use to create as VDB source model.
 - Note the schema names are the names of the visible models in your deployed VDB.

×	🕱 🗖 Import Database via JDBC					
s	Select Database Objects					
	🛞 Can only select a single schema for VDB source model import.					
L						
	Barry_Test_VDB	\$	<no available="" details=""></no>			
	Barry_lest_vob	<u></u>				
	AllParts					
	PartsSupplierA					
	PartsSupplierB					

- **Step 5** The final page shows the name of the resulting VDB source model and the name is NOT editable.
 - All other options are disabled
 - The target **Into Folder** must NOT contain a model with the same name or the **Finish** button will be disabled

Specify Import Options Press the "Next >" button to continue or the "Finish" button to finish. Relational Model Definition Model Name: AllParts	
Relational Model Definition	
Madal Nama	
Model Name: AllParts	
Into Folder:	
TeiidProject_A	
Make target a view model	
Update (if existing model selected)	
Include Catalog For Fully Qualified Names	
Model Object Names (Tables, Procedures, Columns, etc)	
Use Fully Qualified Names (Example: partssupplier.dbo.PARTS)	
Change Case For All Characters	
Case Options	
Make All Upper Case (Example: Suppliers > SUPPLIERS)	
Make All Lower Case (Example: SUPPLIERS > suppliers)	
4	
Rext Next :	

You can use your VDB source model like any other source model in your project. VDB source model tables can be used in your transformation queries and the view models will contain model imports to your VDB source models. However, when your view model is added to a VDB, any referenced VDB source models do NOT get added to your VDB. Instead, an <import-vdb> element (described above) reference is added in it's place.

If VDB imports exist for a VDB, the **Show Import VDBs** button will be enabled and allow viewing the names of the imported VDBs as shown below.

Iodels Other Files	Modeling Action
Model Path 03	Source Name Translator JNDI Name C
😭 AllPartsViews.xmi /ImportVdbTest 🛛 🔽	Model JDBC Sour
	VDB Names PartsTestVDB
\$	
Synchronize All Show Import VDBs	
ata Roles VDB Properties Translator Overr	
Data Role Description	
	Cancel OK
	£14.14

Testing Your Models

As described briefly in Section 1.3.7. "Testing Your Models", can you test your models in Teiid Designer by using the Preview Data action S.

or test your models via your deployable VDB. These two options will be described in detail in this chapter as well as managing your required connection profiles.

9.1. Manage Connection Profiles

Teiid Designer utilizes the Eclipse Data Tools Platform (DTP) Connection Profile framework for connection management. Connection Profiles provide a mechanism to connect to JDBC and non-JDBC sources to access metadata for constructing metadata source models. Teiid Designer also provides a custom Teiid connection profile template designed as a JDBC source to a deployed VDB.

By selecting various Teiid Designer Import options, any applicable Connection Profiles you have defined in your Database Development perspective will be available to use as your import source. From these import wizards you can also create new connection profiles or edit existing connection profiles without leaving the wizard.

The Section C.2.3, "Teiid View" provides access to running Teiid instances and shows data source and VDB artifacts deployed there. The "Create Data Source" action available on this view utilizes the available and applicable connection profiles.

9.1.1. Set Connection Profile for Source Model

Teiid Designer integrates Data Tools Connection Profiles by persisting pertinent connection information in each source model. This can occur through Importing process or through the **Modeling > Set Connection Profile** action.

9.1.2. View Connection Profile for Source Model

In addition to setting the connection profile on a source model you can also view a source model's connection profile information via the **Modeling > View Connection Info** action which displays the detailed properties of the connection.

PartsSourceB.xmi			
roperties			
name	value		
connectionProfileDescription			
connectionProfileCategory	org.eclipse.datatools.connectivity.db.category		
connectionProfileProviderId	org.eclipse.datatools.enablement.oracle.connectionProfile		
connection-url	jdbc:oracle:thin:@englxdbs11.mm.atl2.redhat.com:1521:ORCL		
connectionProfileName	PartsOracle11		
connectionProfileInstanceID	org.eclipse.datatools.enablement.oracle.connectionProfile		
driver-class	oracle.jdbc.OracleDriver		
translator name	oracle		
user-name	partssupplier		

Figure 9.1. Connection Profile Information Dialog

Note that if a source model has no associated connection profile the following dialog will be displayed.



Figure 9.2. No Connection Info Dialog

9.1.3. Remove Connection Profile from Source Model

As a user, you may not want this connection information (i.e. URL, username, etc...) shared through your VDB. Designer provides a means to remove this connection information via a

Modeling > Remove Connection Info action. When adding a source model without connection information will require the user to supply or select the correct translator type.

9.2. Previewing Data For a Model

Designing and working with data is often much easier when you can see the information you're working with. The Designer's Preview Data feature makes this possible and allows you to instantly preview the information described by any object, whether it's a physical table or a virtual view. In other words, you can test the views with actual data by simply selecting the table, view, procedure or XML document. Previewing information is a fast and easy way to sample the data. Of course, to run more complicated queries like what your application likely uses, simply execute the VDB Via DTP and type in any query or SQL statement.

After creating your models, you can test them by using the **Preview Data** action

By selecting a desired table object and executing the action, the results of a simple query will be displayed in the *Section 9.2.5, "Sample SQL Results for Preview Data"* view. This action is accessible throughout the Teiid Designer in various view toolbars and context menus.

There are two requirements for previewing your data:

1. The selected object must be one of several previewable model object types

2. All source models within the model dependency tree must reference a connection profile. Model objects that can be previewed include: relational tables and views (including tables involving access patterns), relational procedures, Web service operations and XML document staging tables.

Note that any virtual table, view or procedure is previewable as long as all "physical" source models reference sufficient connection info. (See Section C.2.3, "Teiid View" view)

After selecting the Preview Data action, Designer will insure that all source models are associated with connection profiles and that all required passwords are set.

If the model selected for preview is a source model and there is insufficient connection info for that model, the following dialog will be displayed and the action terminated.



Figure 9.3. Preview Not Available

If any of the source models in the corresponding project require a password that can't be retrieved from an existing connection profile, the user will be queried for each missing password

E	Missing Password Required 🛛 🛞				
Preview data requires password for model >> Products.xmi Referenced connection profile >> AllProducts					
Password:	****				
	Cancel OK				

Figure 9.4. Missing Password

Testing Your Transformations

When editing transformation SQL Transformation in the Editor. а SQL special Results data action is provided in the editor tool-bar 5

You can change your transformation SQL, re-validate and preview your the data for your modified SQL.

The following sections provide steps for previewing your data. Note that all steps assume that all source models referenced by your models, either directly or through dependencies, are bound to connector bindings.

9.2.1. Preview Relational Table or View

- To preview a relational table, relational view or staging table:
 - Step 1 Select a relational table or view in the Section C.2.1, "Model Explorer View" or diagram. The table or view can be in a view model as well as a source model. Staging tables are not visible in the Section C.2.1, "Model Explorer View", so you need to open the mapping diagram and select it there.
 - Step 2 Right-click select the Preview Data action

You can also select the same action in the tool-bar of either the Section C.2.1, "Model Explorer View" or diagram.

• Step 3 - Your query results will be displayed in the Section 9.2.5, "Sample SQL Results for Preview Data" view. The view will automatically open or get focus if not visible in your perspective.

9.2.2. Preview Relational Table With Access Pattern

- To preview a relational table or view with access pattern:
 - Step 1 Select a relational table or view in the Section C.2.1, "Model Explorer View" or diagram that contains an access pattern. The table or view can be in a view model as well as a source model.
 - Step 2 Right-click select the Preview Data action

You can also select the same action in the tool-bar of either the Section C.2.1, "Model Explorer View" or diagram.

• Step 3 - A column input dialog is presented. Select each access pattern and enter a value for each required column. Note that if data entered does not match the column datatype (String, integer, etc...), an error message will be displayed in the dialog header. When all required values are entered, click the **OK** button to execute the query.

To preview data from this table,	select an access pattern and provide a
All access pattern column values are set. S	Select OK to continue.
Access Patterns	Required Columns PUBLISHER_ID : long 2002 NAME : string(255) Miffling

Figure 9.5. Access Pattern Column Input Dialog

• Step 4 - Your query results will be displayed in the Section 9.2.5, "Sample SQL Results for Preview Data" view. The view will automatically open or get focus if not visible in your perspective.

9.2.3. Preview Relational Procedure

• To preview a relational procedure:

- **Step 1** Select a relational procedure in the *Section C.2.1, "Model Explorer View*" or diagram. The procedure can be in a view model as well as a source model.
- Step 2 Right-click select the Preview Data action

You can also select the same action in the tool-bar of either the Section C.2.1, "Model Explorer View" or diagram.

• Step 3 - An input parameter input dialog is presented. Enter a valid value for each parameter. Note that if data entered does not match the parameter datatype (String, integer, etc...), an error message will be displayed in the dialog header. When all required values are entered, click the **OK** button to execute the query.

🗠 Preview Data	Ē			Đ
To execute this All parameter valu		******	red.	1
CINPUT Parame	blue			
			ОК	Cancel

Figure 9.6. Procedure Parameter Input Dialog

• Step 4 - Your query results will be displayed in the Section 9.2.5, "Sample SQL Results for Preview Data" view. The view will automatically open or get focus if not visible in your perspective.

9.2.4. Preview Web Service Operation

- To preview a Web service operation:
 - Step 1 : Select a Web service operation in the Section C.2.1, "Model Explorer View" or diagram. The operation can be in a view model as well as a source model.

Step 2 : Right-click select the Preview Data action

You can also select the same action in the tool-bar of either the Section C.2.1, "Model Explorer View" or diagram.

• Step 3 : An input parameter input dialog is presented. Enter a valid value for each parameter. Note that if data entered does not match the parameter datatype (String, integer, etc...), an error message will be displayed in the dialog header. When all required values are entered, click the OK button to execute the query.

🍽 Preview Data		
To execute this proc All parameter values are s	edure, input values are required. et. Select OK to continue.	
-Input Parameters - AUTHOR_ID : xs:long	10022	
	OK	Cancel

Figure 9.7. Procedure Parameter Input Dialog

• Step 4 - Your query results will be displayed in the Section 9.2.5, "Sample SQL Results for Preview Data" view. The view will automatically open or get focus if not visible in your perspective.

9.2.5. Sample SQL Results for Preview Data

Preview Data results are displayed in the Eclipse Datatools SQL Results view as shown below. Note there are a number of display preference and filter options for this view via toolbar buttons and the dropdown menu.

🖹 Problems 🚱 Message Log 🔲 SQL Results 🛿			. 3	¢
Type query expression here		INSTR_ID	NAME	TY
Status A Operation Date Connection Profile	1	PRD01088	Novell Incorporated	St
✓ Succee select * from Aug 28, 2010 Transient.org.teiid.d	2	PRD01089	Amazon.com, Incorporated	St
✓ Succee select * from Aug 28, 2010 Transient.org.teiid.d	3	PRD01090	Juniper Networks, Incorporated	S
•	4	PRD01091	Red Hat, Incorporated	St
	5	PRD01092	Boston Scientific Corporation	St
	6	PRD01093	Inex Pharmaceuticals, Incorpora	St
	7	PRD01094	Pfizer, Inc.	St
	8	PRD01095	Cytovax Biotechnologies Incorpo	St
	9	PRD01096	Commonwealth Biotechnologies	St
	10	PRD01097	British Biotechnology plc	St
	11	PRD01220	Unisys Corporation	St
	12	PRD01099	Honeywell International	St
	13	PRD01100	Hilton Hotels Corporation	S
	14	PRD01101	Hilton Hotels Corporation	С
	15	PRD01102	Mercury Interactive Corporation	S
	16	PRD01103	Fidelity Freedom Income Fund	м
		00001104	Edulation Standard Stand	٠.
 ۲ 	\leq			

Figure 9.8. SQL Results View

9.2.6. Execution Plans

When Preview Data is executed, the **Teiid Execution Plan** is also displayed as shown below. The Execution Plan may also be obtained by rt-clicking on a previewable object, then selecting **Modeling > Show Execution Plan** in the context menu.

🖬 SQL Results 🔊 Teiid Execution Plan 😫
Preview Data for : myTbl
SQL
select * from "vModel"."myTbl"
Sciece nom viloder. myrbi
CExecution Plan
Text Plan Tree
xml version="1.0" encoding="UTF-8" standalone="yes"?
<node name="AccessNode"></node>
<property name="Output Columns"></property>
<value>INTKEY (integer)</value>
<value>STRINGKEY (string)</value>
<value>INTNUM (integer)</value>
<value>STRINGNUM (string)</value>
<value>FLOATNUM (double)</value>
<value>LONGNUM (long)</value>
<value>DOUBLENUM (double)</value>
zvalues DVTENUM (bigdocimal) zvalues

Figure 9.9. Teiid Execution Plan View

9.3. Testing With Your VDB

In Teiid Designer you can execute a VDB to test/query actual data.

The requirements for VDB execution are:

- A deployed VDB backed by valid deployed Data Sources
- An instance of a Teiid Connection Profile configured for the deployed VDB

Teiid Designer simplifies this process via Deploy VDB and Execute VDB actions. Deploy VDB does just that, deploy a selected VDB to a running Teiid instance. Execute VDB performs the VDB deployment, creates a Teiid Connection Profile, opens the Database Development perspective and creates a connection to your VDB.

9.3.1. Creating Data Sources

The mechanism by which VDBs are able to query actual data sources is the Data Source. These are deployed configurations backed by database or source connection jars. Each source model referenced within a VDB requiries a JNDI name representing a deployed Data Source.

When creating VDBs you do not need to have deployed data sources on your Teiid server, but if you wish to test your VDB, the data sources need to be present.

Teiid Designer provides a **Create Data Source** action so you can create compatible data sources for your source model. If you wish to create a data source for a specific model you can select that source model in your workspace and select the **Modeling > Create Data Source** action. This will extract the connection profile data from your source model and create a corresponding data source on your default Teiid server.

You can also create data sources from the Teiid view. Select a Teiid server instance in the Teiid view and right-click select the **Create Data Source** action. This will launch the Create Data Source Dialog shown below.

٠		Cre	ate Data Sour	ce .	×
Create Data	a Soui	ce			
All inputs are	valid. S	elect Finish to d	create data sour	rce.	
Teiid Server:	mm://	localhost: 31443	3		
Name	Books	DB2			
Connection	n Sou	rce			
🗌 Use Mode	el Conn	ection Info			
Model					
✓ Use Conr	nection	Profile Info			
Connection P	Profile	BooksDB2		0	New Edit
Connection	n Proj	oerties —			
Name			Value		
password			****		
user-name			books		
connection-	url		jdbc:db2://db(000255.org.my	dbs.com :50000
driver-class			com.ibm.db2.j	cc.DB2Driver	
?				Cancel	<u>Finish</u>

Figure 9.10. Create Data Source Dialog

You can either select and existing Connection Profile from the drop-down list (Use Connection Profile Info option) or check the Use Model Info option and select an existing source model containing connection info.

After creating your new data source it should now be shown in the **Data Sources** folder of the corresponding Teiid server.

9.3.2. Execute VDB from Model Explorer

- If you have a Teiid instance defined and connected in your Teiid view you can:
 - Step 1 : Right-click a VDB in your Model Explorer select Modeling > Execute VDB action. This action will insure your selected VDB is deployed to Teiid, create a Teiid Connection Profile specific for that VDB, open the Database Development perspective and create a connection to your VDB.

マ 🛃 PartsProject			
🕨 🗁 data			
👂 🗁 readme			
🗍 Parts.v	Ne <u>w</u>	>	
👂 宿 PartsS	Modeling	\rightarrow	n Deploy
👂 📬 PartsS(🗌	_		😭 Execute VDB
👂 浳 PartsVi	<u>O</u> pen		💮 Generate JBossWS-CXF War
👂 📬 Sample	Open Wit <u>h</u>	>	,eg

Figure 9.11. Execute VDB Action

 Step 2: Select your new Teild connection profile and right-click select Open SQL Scrapbook, enter your designer SQL (i.e. SELECT * FROM TableXXXX), select all text and right-click select Execute Selected Text

_	Scrapbook 0 🕱 ction <u>p</u> rofile					
Type:	Teiid_7.1		≎ <u>N</u> ame:	Parts - Teiid Connecti 🗘	Data <u>b</u> ase:	s
CEL				<u> </u>		_
SEL	ECT * FROM P	artsSourceA.Pa	rts			

Figure 9.12. SQL Scrapbook Editor

• Step 3 : Results of query should be displayed in the SQL Results view on the Result1 tab.

Туре	query ex	pression h	ere					
Stat	us	Operation		Date	Co	nnection Profile		
•	Succee	SELECT *	FROM PartsSourceA.F	arts Oct 15, 1	2010 Pa	rts - Teiid Connection		
<								
<	PART_ID		PART_NA	ME		PART_COLOR		PAF
1	PART_ID P300		PART_NA Nut	ME		PART_COLOR Red		PAF 12
	-			ME	_	-		
1	P300		Nut	ME		Red		12
1 2	P300 P301		Nut Bolt	ME		Red Green		12 12

Figure 9.13. SQL Results View

9.3.3. Deploy VDB from Model Explorer

You can also deploy your VDB first by selecting it in the Model Explorer and dragging/dropping it onto a connected Teild instance in the Teild view, or right-click select **Modeling > Deploy** action.

Once deployed, you can select the VDB in the Teiid View and right-click select the **Execute VDB** action there. This will create a Teiid Connection Profile specific for that VDB, open the Database Development perspective and create a connection to your VDB. Continue with Step's 2 and 3 above.

Note that if you do not have a Teild instance defined or your default Teild instance is disconnected, the following dialog will be displayed if the **Modeling > Deploy** action is launched.

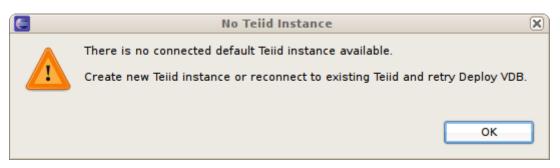


Figure 9.14. No Teiid Instance Defined

9.3.4. Executing a Deployed VDB

To execute a VDB, that's been deployed manually, follow the steps below:

• To execute a VDB, that's been deployed manually, follow the steps below:

- Step 1 : Open the Database Development perspective.
- Step 2 : Select the Database Connections folder and choose the New action to display the New Connection Profile dialog.

Sew Connection Profile	×
Connection Profile	\diamond
Create a Teiid connection profile.	
<u>Connection Profile Types:</u>	
type filter text	
Ingres	
MaxDB	
III ModeShape	
₩ySQL	
🗃 Oracle	
PostgreSQL	
SQL Server	
SQLite	
🔮 Sybase ASA	
💕 Sybase ASE	
🔮 Teiid	
Na <u>m</u> e:	
Products	
Description (optional):	
(?) < <u>Back</u> <u>Next</u> > Cancel <u>Enist</u>	

Figure 9.15. New Connection Profile Dialog

- Step 3 : Enter unique name for your profile, select an existing connection profile type and hit Next.
- Step 4: In the Teiid Profile Wizard page, select the New Driver Definition button to locate and select the Teiid client jar on your file system. Configure your URL using your VDB Name, Host, Port, Username (default = "admin") and Password (default = "teiid").

🧐 Teiid Profile Wizard 🛛 🕅
Specify a Driver and Connection Details
Select a driver from the drop-down and provide login details for the connection.
Drivers: Teiid Server JDBC Driver
Properties
General Optional
V <u>D</u> B Name
Products
Host
localhost
Port
31000
U <u>s</u> ername
admin
P <u>a</u> ssword
SSL Connection
Save password
Connection UNL
✓ <u>C</u> onnect when the wizard completes
Connect every time the workbench is <u>s</u> tarted
< Back Next > Cancel Finish

Figure 9.16. Teiid Connection Profile Dialog

• Step 5 : Select Next to view a summary of your new Teiid Connection Profile.

Property	Value
Name	Products
Description	
Auto connect at startup	false
Auto connect on finish	true
Database	Products
Host	localhost
Port	31000
Username	admin
SSL protocol	false
Save password	true
URL	jdbc:teiid:Products@mm://localhost:31000
	Jabc.tella.Products@mm.//localnost.31000

Figure 9.17. Teiid Connection Profile Summary

- Step 6 : Select Finish.
- Step 7 : Select your new Teiid connection profile and right-click select Open SQL Scrapbook, enter your designer SQL (i.e. SELECT * FROM TableXXXX), select all text and right-click select Execute Selected Text.

*SQL د Connec			X (Parts.	vdb								
<u>T</u> ype:	Teiid	7.1				≎ <u>N</u> a	ame: (Parts - T	eiid Conne	cti (🗘	Data <u>b</u> ase	e: 🔤 C	Sta
SEL	ЕСТ	* FRO	Part	Sourc	eA.Part	ts							

Figure 9.18. SQL Scrapbook Editor

• Step 8 : Results of query should be displayed in the SQL Results view on the Result1 tab.

ype	e query expression here					
Stat	Operation		Date	Connection Profile		
•	Succee SELECT * FROM	PartsSourceA.Parts	Oct 15, 2010	Parts - Teiid Connection		
	PART_ID	PART_NAME		PART_COLOR		P
	PART_ID P300	PART_NAME Nut		PART_COLOR Red		P
	-			_		
2	P300	Nut		Red		
1 2 3 4	P300 P301	Nut Bolt		Red Green	-	1

Figure 9.19. SQL Results View

• Step 9 : The query Execution Plan should also be displayed on the Teiid Execution Plan view tab. The Execution Plan can also be generated without running the query. In the SQL scrapbook, rt-click then select Teiid_7.x > Get Execution Plan.

SQL Results 🐙 Teiid Execution Plan 🖾
Preview Data for : myTbl
SQL select * from "vModel"."myTbl"
Execution Plan
Text Plan Tree
xml version="1.0" encoding="UTF-8" standalone="yes"? <node name="AccessNode"> <property name="Output Columns"> <value>INTKEY (integer)</value> <value>STRINGKEY (string)</value> <value>STRINGKEY (string)</value> <value>INTNUM (integer)</value> <value>STRINGNUM (string)</value> <value>FLOATNUM (double)</value> <value>LONGNUM (long)</value> <value>DOUBLENUM (double)</value></property></node>

Figure 9.20. Teiid Execution Plan View

Searching

Designer provides multiple search actions located via Teiid Designer sub-menu in Eclipses Search menu. **Search** menu.

🔗 Se <u>a</u> rch	Ctrl+H
🐶 <u>F</u> ile	
ኛ <u>R</u> emote	
🛷 Pointcut Matches	
🛷 Beans	
💫 Find Seam References	
Find Seam Declarations	
Text	>
Teiid Designer	>

Figure 10.1. Search Options

- The individual actions in the Teiid Designer sub-menu are described below:
 - 🚀

Transformations... - Launches the Transformation Search dialog. User can search models in the workspace for matching SQL text. Search results appear in the dialog and user can select and view SQL as well as open desired transformations for editing.

8

Metadata... - Launches the Search dialog. User can search for models in the workspace by specifying an Object Type, and/or a Data Type, and/or a property value. Search results appear in the *Section C.2.7, "Search Results View"* view, and double-clicking a result will open that model in the appropriate editor.

• 🔎

Find Model Object - Launches the Find Model Object dialog, which can be used to find an object in the workspace by specifying all or part of its name. Selecting the object will open it in the appropriate editor.

10.1. Finding Model Objects

The Teiid Designer provides a name-based search capability to quickly locate and display model objects.

• To find a model object:

• Step 1 - Open the Find Model Object dialog by either selecting the

action on the main Teiid Designer tool-bar.

			Teiid Des	signer	- PartsPr	oject/PartsSour
e	Se <u>a</u> rch	<u>P</u> roject	<u>M</u> etadata	<u>R</u> un	<u>W</u> indow	<u>H</u> elp
1	🔗 Se <u>a</u> r	ch		Ctrl+H	•	in 'n 🗩
]	将 <u>F</u> ile				• •] 💷 📼 🎤 [
	ኛ <u>R</u> em	ote				
_		tcut Match	ies			~
Du	🛷 Bear	1S				
	💫 Find	Seam Ref	erences			
	😕 Find	Seam De	clarations			
	Text			>		
	Teiid	Designer				ationship
	lond	Designer			1	
					Nr Trai	nsformations
					🔗 Met	adata
					🥭 Find	l Model Object

Figure 10.2. Find Model Object Action In Toolbar

or select the same action via the main menu's Search > Find Model Object

action.

Find Model Object	×
Type Object Name (? = any character, * = any String);	
Part]
Matching Model Objects:	
A PART_COLOR	
A PART_ID	
A PART_NAME	
A PART_WEIGHT	
PARTS	
A parteByColor	5
Locations:	
A /PartsProject/PartsSourceA.xmi/PARTS/PART_COLOR	
/PartsProject/PartsSourceB.xmi/PARTS/PART_COLOR	
/PartsProject/PartsVirtual.xmi/PARTS/PART_COLOR	
/PartsProject/PartsVirtual.xmi/partsByColor/NewProcedureRe	21
	9
Cancel OK)

Figure 10.3. Find Model Object Dialog

- Step 2 Begin typing a word or partial word in the Type Object Name field. Wild-card (*) characters will be honored. As you type, the objects which match the desired name will be displayed in the Matching Model Objects list. If there are more than one objects with the same name, the locations or paths of the objects are displayed in the Locations list.
- Step 3 If more than one object exists with the desired name, select the one of the locations.
- Step 4 Click OK. If editor is not open for the object's model, an editor will open. The desired object should end up displayed in a diagram (if applicable) and selected.

10.2. Search Transformation SQL

The Teiid Designer provides a search capability to string values present in transformation SQL text.

• To search for string values in your transformations SQL:

• Step 1 - Select Search > Transformations... action on the Teiid Designer main menu

which opens the Search Transformations dialog.

۲	Search Tranformations 🛛 🛞
Inpu	ıt
-	Parts
	Case Sensitive
Perfo	orm Search
Sea	rch Status
Sea	rch complete for: [Parts]
Virt	ual Tables, Views, Procedures _Operations
[2] \$	SQL matches for: [Parts]
∇	SupplierInfo <table> : /PartsProject/PartsVirtual.xmi</table>
	VE SELECT
	partsByColor <procedure> : /PartsProject/PartsVirtual.xmi</procedure>
	R SELECT
SEL	ECT SQL String
	ATE VIRTUAL PROCEDURE BEGIN SELECT * FROM PartsSourceB.PARTS WHERE
	sSourceB.PARTS.PART_COLOR = PARTSVIRTUAL.PARTSBYCOLOR.colorin; END
	Edit Close

Figure 10.4. Search Transformations Dialog

- Step 2 Specify a string segment in the Find: field and specify/change your case sensitive preference.
- Step 3 Select Perform Search button. Any transformation object containing SQL text which contains occurances of your string will be displayed in the results section.

You can select individual objects and view the SQL. If a table or view supports updates and there is insert, update or delete SQL present, you can expand the object and select the individual SQL type as shown below.

)	Search Tranformations	(
Input-		
Find: Pa	arts	~
Cas	e Sensitive	
Perform	Search	
Search	n Status	
	complete for: [Parts]	
Virtual	l Tables, Views, Procedures _Operations	
[3] SQL	. matches for: [Parts]	
🕨 🖻 S	SupplierInfo <table> : /PartsProject/PartsVirtual.xmi</table>	A
▶ <u>f</u> , p	partsByColor <procedure> :/PartsProject/PartsVirtual.xmi</procedure>	
🗢 🖪 P	PARTS <table> : /PartsProject/PartsVirtual.xmi</table>	
E	E SELECT	
Q	E INSERT	
E	UPDATE	
E	E DELETE	·
NSERT	r sqL string	
CREATE (PartsS PartsSo	E PROCEDURE BEGIN ROWS_UPDATED = INSERT INTO PartsSourceB.PARTS SourceB.PARTS.PART_ID, PartsSourceB.PARTS.PART_NAME, ourceB.PARTS.PART_COLOR, PartsSourceB.PARTS.PART_WEIGHT) VALUES S.PART_ID, INPUTS.PART_NAME, INPUTS.PART_COLOR, INPUTS.PART_WEIGHT); EN	D
	Edit	Close

Figure 10.5. Insert SQL Example

If you wish to view the selected object and its SQL in a **Model Editor**, you can click the **Edit** button. An editor will be opened if not already open. If an editor is open its tab will be selected. In addition, the **Transformation Editor** will be opened and you can perform **Find/Replace** (Ctrl-F) actions to highlight your original searched text string and edit your SQL if you wish.

10.3. Search Models Via Metadata Properties

The Teiid Designer provides a search capability to find model objects that are characterized by one or more metadata property values.

- To search your models using metadata:
 - Step 1 Select Search > Metadata... action on the main Teild Designer toolbar

which opens the **Search** dialog.

9	Search 🕺
🕻 漀 Remote Sea	rch 💱 File Search 🔗 Metadata Search 🔯 Relationship Search 🕽
Object Type	
Any	
O Object Class:	
Data Type	
Any	
◯ Simple Type:	
🗌 Include su	btypes
O Runtime Type:	bigdecimal 🗘
Properties	
✔ Where:	Name
Contains	O Does NOT Contain C Exact Match
	case insensitive, * = any string, ? = any character):
	ase insensitive, * = any string, ? = any character).
Part	▼
Scope	
<u>W</u> orkspace	○ Selecte <u>d</u> resources
○ Wor <u>k</u> ing set:	Choose
Customize	Cancel <u>S</u> earch

Figure 10.6. Metadata Search Dialog

- Step 2 Specify desired search options for Object Type, Data Type and Properties.
- Step 3 Click Search. The search will be performed and the results will be displayed in the Section C.2.7, "Search Results View". If the view is not yet open, it will be opened automatically.

Appendix A. Supported Data Sources

The matrix indicates for a given data source how a model can be created (Desginer Import Option) and how the data source is integrated (Translator) for data access.

NOTE: The DDL Import option is an available option to build a source model for any data source. Its only indicated below when there's no specific importer created for that specific data source type.

Data Source	Translator Type	Designer Import Option
Apache Derby	JDBC - derby	JDBC Importer
Files	file	File Importer
General JDBC	JDBC - jdbc-simple	JDBC Importer
HDFS (Hadoop)	hive	use DDL importer or perform modeling manually
HSQL	JDBC - hsql	JDBC Importer
H2	JDBC - h2	JDBC Importer
Ingres	JDBC - ingres (Ingres 2006 or later) JDBC - ingres93 (Ingres 9.3 or later)	JDBC Importer
IBM DB2	JDBC - db2	JDBC Importer
Informix	JDBC - informix	JDBC Importer
LDAP/ActiveDirectory	ldap	use DDL importer or perform modeling manually
LoopBack	JDBC - loopback	use DDL importer or perform modeling manually
MetaMatrix	JDBC - metamatrix	JDBC Importer
ModeShape/JCR	JDBC - modeshape	JDBC Importer
Mondrian	olap	use DDL importer or perform modeling manually
MS Access	JDBC - access	JDBC Importer
MS Excel	JDBC - excel-odbc	JDBC Importer
MS SQL Server	JDBC - sqlserver	JDBC Importer
MySQL	JDBC - mysql5 (mysql)	JDBC Importer

Table A.1. Teiid Designer Supported Data Sources

Data Source	Translator Type	Designer Import Option
Netezza	JDBC – netezza	JDBC Importer
Oracle	JDBC - oracle	JDBC Importer
PostgreSQL	JDBC - postgresql	JDBC Importer
Salesforce.com	salesforce	SalesForce Importer
SAP Gateway	WS	File Source (XML) Importer
SAP R/3	WS	
SAP Services Registry	WS	WSDL Importer
Sybase ASE	JDBC - sybase	JDBC Importer
Teradata	JDBC - teradata	JDBC Importer
Teiid	JDBC - teiid	JDBC Importer
Web Services (SOAP/WSDL)	WS	WSDL or URL Importer
Web Services (Rest/OData)	WS	File Source (XML) Importer

Appendix B. User Preferences

The Teiid Designer provides options or preferences which enable customization of various modeling and UI behaviors. Preferences can be accessed via the Edit > Preferences action on the Main toolbar.

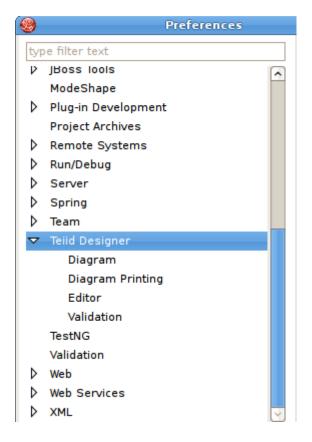


Figure B.1. Preferences Dialog

B.1. Teiid Designer Preferences

General Teiid Designer preferences include.

- Always open editor without prompting To change/edit a model, it must be opened for editing. Checking this box will automatically open the model in an editor if the user attempts to perform a change in a model. If unchecked, the user will be informed that an editor will be opened before the operation is completed.
- Open Designer perspective when model is opened If a model is opened via importing projects, the New > Teiid Metadata Model menu and the Teiid Designer perspective is not open, you may want to automatically open the perspective and beging working on your model. This preference has 3 settings. Always open, which means always open the perspective without prompting; never open, which means do not open the Teiid Designer perspective, or prompt, which will always ask you if you wish to open the Teiid Designer perspective.

- Check and update imports during save occasionally editing a model may add or remove objects in one model that reference objects in another model. Model Imports keep track of these dependencies within each model. A validation error or warning may appear during a build. Checking this box will automatically check and update imports during the save process. This will result in any unneeded imports being removed from the model or any required imports added to the model. If unchecked, no updating of imports will be performed.
- Enable Preview If the Designer Runtime feature is installed and a Teiid Instance is defined, Teiid Designer will automatically keep the preview artifacts (VDBs) in sync with the workspace models. Unchecking this preference will disable preview feature and not create preview artifacts.
- Enable Preview Teiid Cleanup If operating Designer with Enable Preview = TRUE, then this
 preference will result in automatic clean-up of your preview artifacts from your Teiid servers.
 Any preview VDBs or preview data sources will be undeployed from your servers as part of
 Eclipse's shut-down process.

Teiid Designer	ר י≎ י
Specify values for general Designer preferences.	
Preview Data/VDB Execution	
✓ Enable auto-creating of a source model's data source on Teiid server (VD	B Execution)
✓ Enable Preview	
✓ Enable Preview Teiid Cleanup	
Teiid Designer	
Always open editor without prompting	
Open Designer perspective when a model is opened	
◯ Always open ◯ Never open ④ Prompt	
Check and update imports during save	
✓ Import Logical ER Models as View Relational Models	

Figure B.2. General Teiid Designer Preferences Panel

B.1.1. Diagram Preferences

Several diagram preferences are available to customize your diagrams.

• Notations - Standard diagram notation for Teild Designer is based on UML notation. Future releases may include alternate notations.

- **Routers** The relationship link type for Package and Custom diagrams (Foreign Key Primary Key relationships) can be customized. Available options include Orthogonal (default), Direct or Manual (user defined breakpoints).
- Font Settings Select font type, style and size.
- **Background Color Settings** Select a unique background color for each diagram type to help differentiate between types.
- **Model Size** Displaying very large diagram may take a considerably long time. This preference allows users to set an upper limit on the number of objects to display in a diagram. If this limit is exceeded, a warning is displayed to the user and the diagram is not constructed.
- Relationship Options UML-type relationships can be customized in a couple of ways. Role Names and Multiplicity labels can be shown or hidden using the check-boxes labeled Show Role Names and Show Multiplicity.

Diagram	⇔ → →
General settings for Diagrams	
Notations	
Default Notation: UML	•
Routers	
Default Router Style: Orthogonal	•
Font Settings	
Font Type: Sans - Regular - 8	ew / Change
Background Color Settings	
Dependency Diagram: Custom Diagram:	
Mapping Diagram: Relationship Diagram:	
Transformation Diagram: Custom Relationship Diagram:	
Package Diagram:	
Model Size	
Warn If Model Size is Larger Than 20000	
Relationship Options	
☑ Show Role Names	
Show Multiplicity	
Restore <u>D</u> efaults	Apply

Figure B.3. Diagram Preferences Panel

B.1.2. Diagram Printing Preferences

Diagram print options are stored as preferences. These can be accessed through this preference page, by right-click on diagram **Page Setup action**



or via the **Modify Diagram Printing Preferences** action

located on the vertical diagram toolbar .

Diagram Printing	<	≻ ⇒ →
Diagram Print		
Orientation		
🔿 Portrait 💿 Landscape		
Scaling		
○ Fit to: One page		
○ Fit to: One page high, as many as necessary wide		
○ Fit to: One page wide, as many as necessary high		
Adjust to:		
100 % normal size		
Margins		
Тор: 0.0		
Left: 0.0	Righ	t 0.0 🗸
Bottom: 0.0		
Page Order		
 Over, then down 		
○ Down, then over		
	Restore <u>D</u> efaults	Apply

Figure B.4. Diagram Preferences Panel

B.1.3. Editor Preferences

B.1.3.1. XML Document Preferences

XML Document Mapping Preferences provide ways to customize *Section C.3.1.1.4, "Mapping Diagram*" and *Section 5.2.4, "Recursion Editor (XML)*" behavior.

Editor $\Leftrightarrow \checkmark \Leftrightarrow \checkmark \checkmark$	•
XML Document Mapping Settings Transformation Table VDB	
XML Mapping Diagram Settings	
Auto-expand tree when model has fewer than 21 mapping classes.	
Auto-expand XML document tree to level:	
Hide Mapping Class Columns by Default	
XML Document Mapping	
Upper XML Recursion Limit: 10	
Remove Duplicate Attributes on Merge	

Figure B.5. XML Document Preferences Panel

B.1.3.2. Table Editor Preferences

Section C.3.1.2, "Table Editor" Preferences provide a way to customize the order and the information content for each model object type.

ML Document Mapping Settings	Transformation Table	VDB	
Preferences for a given table will or it has been viewed. After res viewed again for the default prefe	toring defaults, the tab	le editor mus	
Table	Columns		
Base Tables	Column	Visible	<u>_</u>
Columns	Location	Yes	
Foreign Keys	Name	Yes	
Function Parameters	Name In Source	Yes	
Inputs	Native Type	Yes	
Interfaces	Length	Yes	
Operations	Length Fixed	Yes	
Outputs	Numeric Precision	Yes	Up
Primary Keys	Numeric Scale	Yes	Down
Procedures	Nullable	Yes	
Procedure Parameters	Auto Incremented	Yes	
Procedure Results	Default Value	Yes	
Return Parameters	Minimum Value	Yes	
Sample Messages	Maximum Value	Yes	
Scalar Functions	Format	Yes	
Schemas	Character Set Name	Yes	
Views	Collation Name	Yes	

Figure B.6. Table Editor Preferences Panel

B.1.3.3. Transformation Editor Preferences

Section 5.2.1, "Transformation Editor" Preferences provide a way to customize SQL formatting, diagram layout, and default view entity properties.

Editor	⇔ • ⇔~ ▼
XML Document Mapping Settings Transformation Table VDB	
SQL Clauses ✓ Start Clauses on New Line ✓ Indent Clause Content Auto-expand default SELECT * FROM on construction	
 Diagram Layout View and Sources Tree Layout 	
Miscellaneous Default String Length: 10	

Figure B.7. Transformation Editor Preferences Panel

B.1.3.4. VDB Editor Preferences

Section C.3.2, "VDB Editor" Preferences provide a way to customize VDB editor behavior.

Editor	⇔• ⇔• ▼
XML Document Mapping Settings Transformation Table VDB Image: Synchronize without showing warning dialog Synchronize without showing warning dialog Synchronize without showing warning dialog	

Figure B.8. VDB Editor Preferences Panel

B.1.4. Validation Preferences

Validation Preferences provide a way to customize the severity of some of the rules checked during model validation.

The Validation preference pages, shown below, include the validation preferences for **Core**, **Relational**, **XML and XSD** (**XML Schema**) models.

Validation	⇔
Specify the Severity Level for validation-related occurrences.	
Core Relational XSD XML	
Core Validation Control Settings	
Multiple EObjects with same uuid.	Ignore 😂
String functions (SUBSTRING, LOCATE, and INSERT) 0 to 1 base check:	Warning 😂

Figure B.9. Core Model Validation Preferences Panel

Validation	⇔ • ⇔ •
Specify the Severity Level for validation-related occurrences.	
Core Relational XSD XML	
Relational Validation Control Settings	
Missing length property on columns with string or character types:	Warning 🗘
Missing precision property on columns with numeric types:	Warning 🗘
Columns of builtin type integer:	Warning 😂
Name in source conflicts with sibling entities:	Ignore 😂
Indexes with columns from more than one table:	Warning 😂
Empty transformations:	Error \$

Figure B.10. Relational Model Validation Preferences Panel

Validation	⇔
Specify the Severity Level for validation-related occurrences. Core Relational XSD XML	
XML Validation Control Settings	
XML Document Elements/Attributes not referencing an XML Schema component:	Warning 😂
Excluded Element from XML Document required by XML Schema:	Warning 😂
XML document entity violates max occurs specified by its schema component:	Warning \$
Unmapped required XML element or attribute:	Error 😂
Excluded Elements/Attributes from XML Document are mapped:	Warning \$
Mapped XML Element/Attribute has fixed or default value:	Warning \$
Mapped XML Elements/Attributes with zero minimum occurrences:	Ignore 🗘
Mapped XML Elements/Attributes with one maximum occurrence:	Warning 😂
XML Root Element mapped to Mapping Class:	Warning \$
Mapped XML Elements/Attributes Nillable:	Ignore 😂
Incompatible Datatypes for Column-to-Element/Attribute Mappings:	Warning \$
Restore <u>D</u> efaults	Apply

Figure B.11. XML Document Model Validation Preferences Panel

Validation	⇔ • ⇔∘ ▼
Specify the Severity Level for validation-related occurrences. Core Relational XML	
XSD Validation Control Settings	
XML Schema Document validation problems	Ignore 🗘

Figure B.12. XSD Schema Model Validation Preferences Panel



Note

Increasing the severity level to error will prevent you from testing your VDB or deploying a web service if violations of that preference are found during validation.

Appendix C. Teiid Designer Ui Reference

C.1. Teiid Designer Perspectives

Teiid Designer utilizes the *Eclipse* [http://www.eclipse.org] Workbench environment which controls visual layout via perspectives. A **Perspective** defines the initial set and layout of views and editors. Within the application window, each perspective shares the same set of editors. Each perspective provides a set of functionality aimed at accomplishing a specific set of tasks.

Perspectives also control what appears in certain menus and toolbars. They define visible action sets, which you can change to customize a perspective. You can save a perspective that you build in this manner, making your own custom perspective that you can open again later.

C.1.1. Teiid Designer Perspective

The Teiid Designer perspective provides access to fundamental model editing and management capabilities. This perspective includes 6 main UI components (or groups of components) as shown below. They include:

- Section C.2.1, "Model Explorer View" Teiid 'tree' view of Model Objects.
- Section C.2.3, "Teiid View" Teiid Server instance view. Provides view of contents for connected instances of installed Teiid runtime.
- Section C.3.1, "Model Editor" Custom editors targeted for ".xmi" metadata model files.
- Section C.2.4, "Properties View" Standard property values for selected workbench objects.
- Section C.2.13, "Guides View" Guides and Status Views which enhance usability.
- **Miscellaneous Views** Includes the Problems view, Message Log view and the SQL Results view (opened if Preview Data action is performed)

			Teii	d De	esigne	er - MyP	roject/1	testM	odel.	xmi -	Eclip	se SD	К			
Run Win	dow	Help														
⇒~] ₽	100	% ~	P	A	▲ 🖁	800										E 🕥
	🔓 te	estMode	el.xm	ាi ន												
₿ 🕏 🎽	ø															
	£															
	Ð															
	1															
		<								ш						>
	E I	Package	e Dia	gran	ו 🎹 Ta	able Edito	or									
V	🖹 P	roblem	s 🛙	Q	J Error	Log										
	0 ite	ms														
	Des	criptior	۱										Res	ource	Path	
	2								111							

Figure C.1. Teiid Designer Perspective Layout

C.1.2. Opening a Perspective

There are two ways to open a perspective:

- Using the Open Perspective button
 on the shortcut bar.
- Choosing a perspective from the **Window > Open Perspective** menu.
- To open a perspective by using the shortcut bar button:
 - Step 1 Click on the Open Perspective button
 - Step 2 A menu appears showing the same choices as shown on the Window > Open Perspective menu. Choose Other from the menu.



Figure C.2. Perspectives Menu

• Step 3 - In the Select Perspective dialog choose Teiid Designer and click OK.

	Open Perspective	×
	🐉 Java	
	🔊 Java Browsing	^
	Pava EE	
	Java Type Hierarchy	
	ava specific and the second se	
	JBoss AS	
	BPM jPDL 3	
	a [©]]MX	
	↓ JPA	
	↓ Plug-in Development	
	Remote System Explorer	
	Seam (default)	
	F ⁰ Team Synchronizing	
	😭 Teiid Designer	
	Web	
	🛞 Web Development	
	X XML	-
L		Ū
	Cancel OK	

Figure C.3. Select Perspective Dialog

The Teiid Designer perspective is now displayed.

There are few additional features of perspectives to take note of.

• The title of the window will indicate which perspective is in use.



Figure C.4. Workbench Window Title Bar

- The shortcut bar may contain multiple perspectives. The perspective button which is pressed in, indicates that it is the current perspective.
- To display the full name of the perspectives, right click the perspective bar and select **Show Text** and conversely select **Hide Text** to only show icons.
- To quickly switch between open perspectives, select the desired perspective button. Notice that the set of views is different for each of the perspectives.

🖹 🕞 Teiid Desig 🔓 Database D	
------------------------------	--

Figure C.5. Workbench Window Title Bar

C.1.3. Further information

For more details on perspectives, views and other Eclipse workbench details, see formal *Eclipse Documentation* [http://help.eclipse.org/ganymede/index.jsp].

C.2. Teiid Designer Views

Views are dockable windows which present data from your models or your modeling session in various forms. Some views support particular *Section C.3.1, "Model Editor"* and their content is dependent on workspace selection. This section summarizes most of the views used and available in Teiid Designer. The full list is presented in the main menu's *Window > Show View > Other...* dialog under the **Teiid Designer** category.

🔲 Show View	\otimes
type filter text	
👂 🗁 Team	^
▽ 🗁 Teiid Designer	
Datatypes	
Description	
🏫 Metadata Favorites	
Model Explorer	
Relationship Navigator	
ở System Catalog	
I Tags	
t Teiid	
🔥 Teiid Model Classes	~
Cancel	

Figure C.6. Eclipse Show View Dialog

C.2.1. Model Explorer View

Teiid Designer allows you manage multiple projects containing multiple models and any corresponding or dependent resources. The *Model Explorer* provides a simple file-structured view of these resources.

The Model Explorer (shown below) is comprised of a toolbar and a tree view.

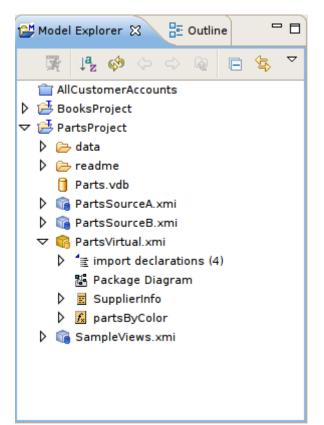


Figure C.7. Model Explorer View

The toolbar consists of nine common actions:

• 록

Preview Data - Executes a simple preview query (SELECT * FROM).

• [a,

Sort Model Contents - Sorts the contents of the models based on object type and alphabetizing.

ଂ 🏟

Refresh Markers - Refreshes error and warning markers for objects in tree.

• 🔶

Back - Displays the last "Go Into" location. (See Eclipse Help)

• 🔶

Forward - Displays the next "Go Into" location. (See Eclipse Help)

ଂ 🧔

Up - Navigates up one folder/container location. (See Eclipse Help)

Collapse All - Collapses all projects.

```
• 🚖
```

Link with Editor - When object is selected in an open editor, this option auto-selects and reveals object in Model Explorer.



Additional Actions

The additional actions are shown in the following figure:

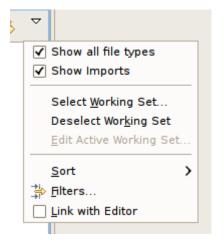


Figure C.8. Additional Actions

If **Show Model Imports** is checked, the imports will be displayed directly under a model resource as shown below.

- 🗢 💼 PartsVirtual.xmi
 - ▼ ¹
 [†]
 [†]
 - -- /PartsProject/PartsSourceA.xmi
 - -- /PartsProject/PartsSourceB.xmi
 - --- http://www.w3.org/2001/XMLSchema
 - -- /PartsProject/SampleViews.xmi
 - 🔠 Package Diagram
 - 👂 🧮 SupplierInfo

Figure C.9. Show Model Imports Action

C.2.1.1. Selection-Based Action Menus

Selecting specific objects in the **Model Explorer** provides a context from which the Teiid Designer presents a customized menu of available actions.

Selecting a **view model**, for instance, results in a number of high-level options to manage edit model content, perform various operations and provides quick access to other important actions available in Teiid Designer. These may include specialized actions based on model type.

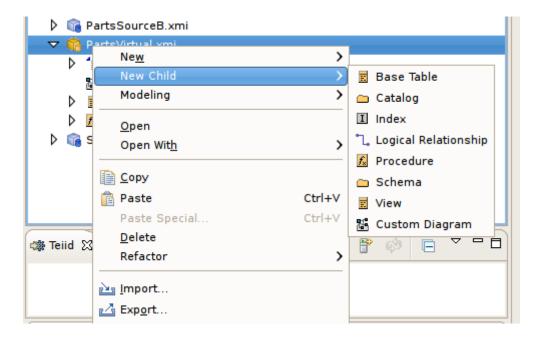


Figure C.10. Sample Context Menu

C.2.2. Outline View

The Outline View is a utility view which provides both at tree view dedicated to a specific model (open in an editor) and a scaled thumbnail diagram representative of the diagram open in the corresponding Diagram Editor.

You can show the Outline View by clicking on its tab. If there is no open editors, the view indicates that Outline is not available. If a Model Editor is open, then the root of the displayed tree will be the model for the editor that is currently in focus in Teiid Designer (tab on top).

C.2.2.1. Outline Tree View

This tree view provides the same basic editing and navigation behavior as the Model Explorer. One additional capability is the drag and drop feature which provides re-ordering and re-parenting of objects in a model.

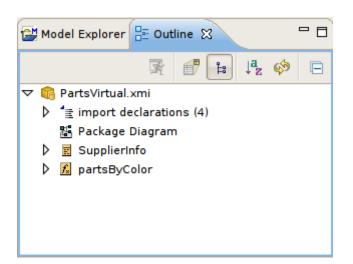


Figure C.11. Outline View

C.2.2.2. Outline Thumbnail View

The Outline View also offers you a way to view а thumbnail sketch of your diagram regardless of its size. To view this diagram thumbnail from the Outline panel, click the Diagram Overview button í,

at the top of the view. The diagram overview displays in the Outline View.

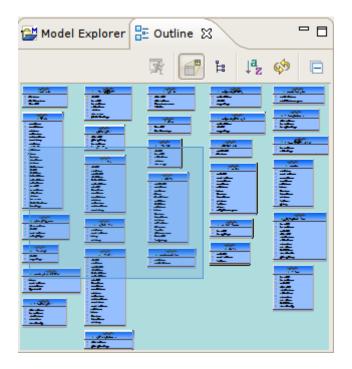
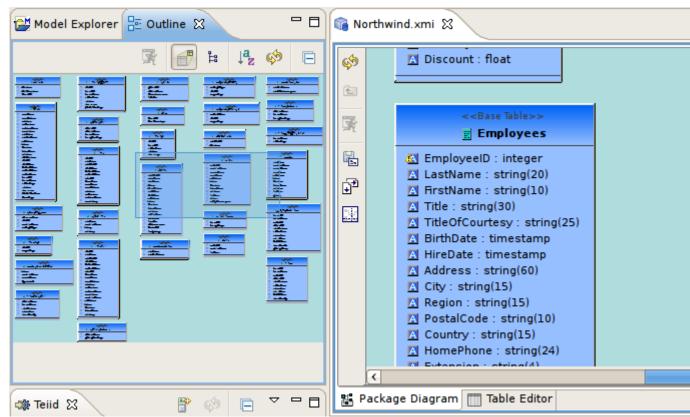


Figure C.12. Outline View

The view contains a thumbnail of your entire diagram. The shaded portion represents the portion visible in the Diagram Editor view.

To move to a specific portion of your diagram, click the shaded area and drag to the position you want displayed in the Diagram Editor view.



C.2.3. Teiid View

The Teiid View provides a means to display and manage Teiid server instances and their contents within Designer.

To show the Teiid View click "Window > Show View > Other..." to display the Show View dialog. Choose "Teiid Designer > Teiid" view and hit OK.

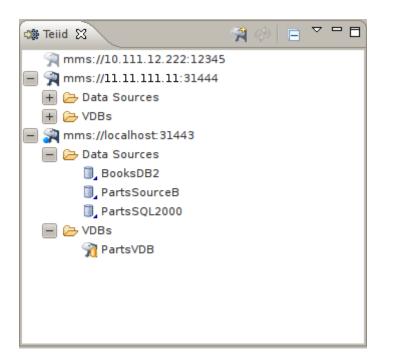


Figure C.13. Teiid View

То create new Teiid instance, either right-click select а New Teiid Instance action click action the same button, or Å

, in the toolbar.

You'll get the New Teiid Instance dialog shown below.

reate a new Teiid S	erver Connection	
Define the Teiid Serve	er connection properties required to perform both JDBC and Admin tasks.	
Display Name Teiid	8.2 Final on AS7	
JBoss Server		
JBoss Server	N	lew
Teiid JDBC Connect Used to auto-connect determined at exec	ect to your VDB as part of the Execute VDB and Preview Data actions. The value of the <vdbname> wi</vdbname>	ill be
Used to auto-conne	ect to your VDB as part of the Execute VDB and Preview Data actions. The value of the <vdbname> wi</vdbname>	ill be
Used to auto-conne determined at exec	ect to your VDB as part of the Execute VDB and Preview Data actions. The value of the <vdbname> wi cution time.</vdbname>	ill be
Used to auto-conne determined at exec User name Password	ect to your VDB as part of the Execute VDB and Preview Data actions. The value of the <vdbname> wi cution time.</vdbname>	ill be
Used to auto-conne determined at exec User name Password	ect to your VDB as part of the Execute VDB and Preview Data actions. The value of the <vdbname> wi cution time. user ****</vdbname>	ill be
Used to auto-conne determined at exec User name Password	ect to your VDB as part of the Execute VDB and Preview Data actions. The value of the <vdbname> wi cution time. user ****</vdbname>	ill be

Figure C.14. New Teiid Instance Dialog

The dialog contains two 2 sections in addition to entering the displayed name of your Teiid Server connection. The first panel, **JBoss Server**, requires you to select the JBoss Server definition where your Teiid Server has been installed.

The second panel, **Teiid JDBC Connection Info**, provides for entering the connection information for that same Teiid instance. This information is required for Designer to make JDBC connections during execution of the Preview Data feature.

Enter valid User name and Password information, edit any options and click Finish.

Actions available in this view include:

New Teiid Instance - Create a new instance of a running Teiid server

• 🚀

Teiid Server Properties - View and edit properties of an existing Teiid instance

ଂ 🏟

- Reconnect and refresh contents of the selected Teiid instance

×

Delete - Disconnect and delete the selected Teiid instance

` <mark>0</mark>-{

Execute VDB - Creates a JDBC Teild connection profile and opens the Data Tools Database Development perspective

• 🗙

Undeploy VDB - Removes the selected VDB from the Teiid instance

່ À

Create Data Source - Launches the New Data Source wizard

• 🗙

Delete Data Source - Removes the selected Data Source from the Teiid instance

Note that once your server is started, **WTP**'s **Server** view will now display the data source, translator and VDB content for your running Teiid server as shown below.

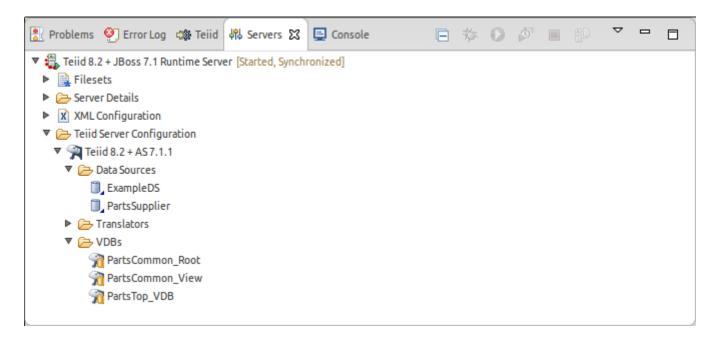


Figure C.15. Teiid Contents in Server View

C.2.4. Properties View

The **Properties View** provides editing capabilities for the currently selected object in Teild Designer. The selection provided by whichever view or editor is currently in focus will determine the its contents.

To edit a property, click a cell in the Value column. As in the **Table Editor**, each cell provides a UI editor specific to the property type.

	Properties 🕱 📄 Des	scription 🔚 🛱 🖾 🖻 🗖 🗉	3
Pro	perty	Value	4
$\overline{}$	Info		
	Object URI		
	Searchability	E SEARCHABLE	
	Selectable	🙀 true	
	Updateable	🐼 true	
$\overline{}$	Misc		-
	Distinct Value Count	un -1	
	Name	LA SUPPLIER_NAME	
	Name In Source	LA SUPPLIER_NAME	
	Null Value Count	ui -1	
$\overline{}$	Туре		
	Datatype	string (Path=http://www.w3.org/2001/XMLSci	
	Native Type	LA varchar	
	Nullable	E NULLABLE	
∇	Tine Detail	l l	~]

Figure C.16. Properties View

If the model for the object being edited is not open in an editor, a dialog may appear confirming the attempt to modify the model and asking the user to confirm or cancel. This dialog can be prevented by checking the preference Always open editor without prompting. You can re-set/uncheck this property via the Teiid Designer's main preference page.

۲	Open Model Editor 🛛 🛞					
?	You must open this model in an editor tab before you can modify the property value. Do you want to open the editor?					
🗌 Alway	vs open editor without prompting					
	<u>N</u> o Cancel <u>Y</u> es					

Figure C.17. Open Model Editor Dialog

Properties can also be edited via a right-click menu presented below.

	Value		
nt	41 -1		<
	LA SUPPLIER_NA		1.1
	K SUPPLIER NA	Cut	:ka
	u.i -1	Copy	
		📳 <u>P</u> aste	lei
	string (Path=	🗃 <u>D</u> elete	
	🛯 🗛 varchar		_
	E NULLABLE	📄 Select <u>A</u> ll	ge
		Input <u>M</u> ethods	> ior
	🔩 false	Insert Unicode Control Character	5 Int
	🙀 true		Jnh

Figure C.18. Open Model Editor Dialog

The **Properties** toolbar contains the following actions:

• 🗄

Show Categories - toggles between categorized properties and flat alphabetical properties list.

• ∔

Show Advanced Properties - shows/hide advanced properties (if available).

• 🛃

Restore Default Value - for a selected property, this action will reset the current to a default value (if available).

C.2.5. Description View

The **Description View** provides a means to display and edit (add, change or remove) a description for any model or model object. To show the Description View click **Window > Show View > Other...** to display the *Figure C.6, "Eclipse Show View Dialog*" dialog. Choose **Teiid Designer > Description** view and hit **OK**.

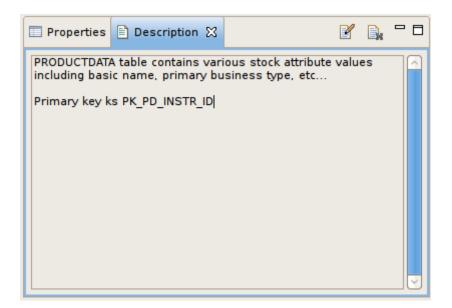


Figure C.19. Description View

You can click the edit description action in the toolbar or right-click select "Edit" in the context menu to bring up the Edit Description dialog. edit actions.

Properties	Description	×	[4		
			ous stock attribute v siness type, etc	alue	es	
Primary key k	s PK_PD_INSTR_	ID				
			Select <u>A</u> ll			
			🔗 Find			
			Edit			
			🔒 Clear			. 🗸
BaseTa	ble: Products/PR		Input Methods CTUATA	6	>	

Figure C.20. Description View Context Menu

Edit Description for PRODUCTDATA	(
Description	
PRODUCTDATA table contains various stock attribute values including basic name, primary business type, etc	^
Primary key ks PK_PD_INSTR_ID	
	~
Cancel OK	

Figure C.21. Edit Description Dialog

C.2.6. Problems View

The **Problems View** displays validation errors, warnings, or information associated with a resource contained in open projects within your workspace.

ems 🖾				
warning, 0 others				
ion	Resource	Path	Location	Тур
rors (1 item)				
SQL statement is empty.	Books.xmi	/BooksProject	bookCollection	Pro
rnings (1 item)				
The type referenced by column book_edition must be marked as an enterprise datatype	Books.xmi	/BooksProject	bookCollection/Ł	Pro

Figure C.22. Problems View

By default, the **Problems View** is included in the Teiid Designer perspective. If the **Problems View** is not showing in the current perspective click **Window > Show View > Other > Teiid Designer > Problems**.

There are 5 columns in the view's table which include:

1. Description - A description of the problem preceded by a severity icon (i.e., error



2. **Resource** - The name of the resource.

3. Path - The project name.

4. Location - The object within the resource that has a validation error.

5. type - Type of validation item.

C.2.6.1. Toolbar Menu

Click	the	upside-down	triangle

icon to open the View Menu icon to see various options including sorting, filtering, displayed columns and much more.

C.2.6.2. Context Menu

Additional actions are available by selecting a problem and right-click to open a context menu.

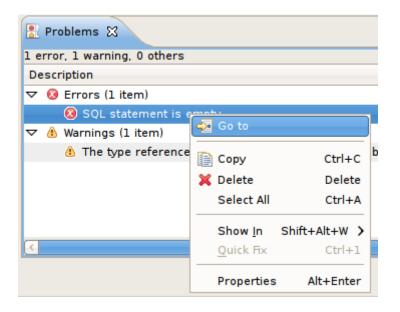


Figure C.23. Problems View Context Menu

- Go To will open the appropriate editor and select the affected/referenced object.
- Show In Navigator Opens the Basic > Navigator view (if not open) and expands file system tree and reveals applicable resource.
- Copy Copies the problem information to the system clipboard.
- **Paste** Pastes the problem information located in the system clipboard (if applicable) into the curor location for a specified text editor.
- Delete Deletes the selected problem rows (if applicable).
- Select All selects all problems in the table.
- Quick Fix (Not yet implemented in Teiid Designer).
- Properties displays a dialog containing additional information.

C.2.7. Search Results View

Below is an example set of search results. The view contains rows representing matches for your search parameters. You can double-click a entry and the object will be opened and selected in an editor and/or the Model Explorer if applicable.

🖹 Problems 🔗 Search 🕱	Û	Û	ж	*	Ŧ		>>		! ?~	Ľ
feature=Name, pattern=*Part* - 26 matches										
🕨 📬 Found 8 matches - /PartsProject/PartsSourceA.xmi										
Found 1 match - /PartsProject/SampleViews.xmi										
🗢 💼 Found 7 matches - /PartsProject/PartsVirtual.xmi										
PART_NAME : PART_NAME - http://www.metamatrix.c	om/metai	mode	els/Rel	ationa	al#//C	olum	n			
PART_COLOR : PART_COLOR - http://www.metamatri	ix.com/me	etamo	odels/	Relati	onal#	//Colu	umn			
PART_ID : PART_ID - http://www.metamatrix.com/me	tamodels,	/Rela	tional	#//Co	lumn					
A PART ID : PART ID - http://www.metamatrix.com/me	tamodels,	/Rela	tional	#//Co	lumn					
PART WEIGHT : PART WEIGHT - http://www.metamat	trix.com/m	netan	nodels	s/Rela	tional	#//Co	lumn			
▲ PartsVirtual : PartsVirtual - http://www.metamatrix.	.com/meta	amod	lels/Co	ore#//	/Mode	IAnno	tation			
🔏 partsByColor : partsByColor - http://www.metamat	rix.com/m	etam	nodels	/Relat	tional	#//Pro	ocedui	re		
Found 10 matches - /PartsProject/PartsSourceB.xmi										

Figure C.24. Search Results View

The toolbar actions for the Search Results view are:

• 🕂

Show Next Match - Navigates down one row in the view.

• 🔶

Show Previous Match - Navigates up one row in the view.

• 🕱

Remove Selected Matches - Removes selected results from the view.

• 🎉

Remove All Matches - Clears the view.

• 🔏

Search - Launches the MoTeiid Designerearch Dialog.

• 🐶

Previous Search Results - Select previous search results from history.

You can also perform some of these actions via the right-click menu:

🖹 Problems 🔗 Sear	ch 🕄	Û
	n=*Part* - 26 matches	Ai
	s - /PartsProject/PartsSource	
A PART_ID : http://www.inter- A PART_NAME :	Show In	Shift+Alt+W 💙 at
	🕂 Next Match	Ctrl+.
A PART_ID : htt PARTS : http		Ctrl+, h
1 ■ PartsSource	🕂 Expand All	: o
PART_WEIGH SUPPLIER PA	Сору	Ctrl+C ^e s
Found 1 match	X Remove Match	
Found 7 matche	Remove Selected Matche	s Delete
👂 📬 Found 10 matcl	💥 Remove All Matches	
	💖 Search Again	F5
	Edit	
<u> </u>		

Figure C.25. Search Results Context Menu

C.2.8. Datatype Hierarchy View

To open **Teiid Designer's Datatype Hierarchy** view, select the main menu's **Window > Show View > Other...** and select the **Teiid Designer > Datatypes** view in the dialog.

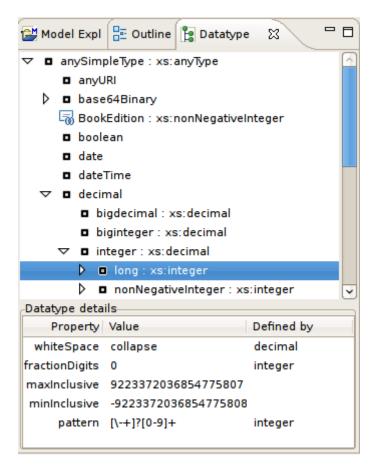


Figure C.26. Datatype Hierarchy View

C.2.9. Teiid Model Classes View

The Model Classes View provides a hiearchical EMF-centric view of the various metamodel classes available within Teiid Designer. This view is primarily for informational purposes, but can be used as a reference if creating relationships or searching your workspace for specific metamodel constructs.

2	Мо	del Explorer 🗄 Outline 航 Teiid Model Classes 🕱 🛛 🧧	E
⊳	*	http://www.eclipse.org/emf/2002/Ecore	6
⊳	÷	http://www.eclipse.org/emf/2002/Mapping	
⊳	*	http://www.eclipse.org/uml2/3.0.0/UML	
⊳	*	http://www.eclipse.org/xsd/2002/XSD	
⊳	÷	http://www.metamatrix.com/metamodels/Compare	
⊳	灓	http://www.metamatrix.com/metamodels/Core	
⊳	쒏	http://www.metamatrix.com/metamodels/Dependency	
⊳	\$\$	http://www.metamatrix.com/metamodels/Diagram	
⊳	\$\$	http://www.metamatrix.com/metamodels/Extension	
⊳	*	http://www.metamatrix.com/metamodels/JDBC	
⊳	*	http://www.metamatrix.com/metamodels/MetaMatrixFunction	
$\overline{}$	쒏	http://www.metamatrix.com/metamodels/Relational	
	Þ	AccessPattern -> RelationalEntity [com.metamatrix.metamodels.r	
	Þ	BaseTable -> Table [com.metamatrix.metamodels.relational.Base	
	Þ	Catalog -> RelationalEntity [com.metamatrix.metamodels.relation]	
	Þ	Column -> RelationalEntity [com.metamatrix.metamodels.relation	
	Þ	ColumnSet -> RelationalEntity [com.metamatrix.metamodels.rela	
	Þ	DirectionKind	L
	Þ	ForeignKey -> Relationship [com.metamatrix.metamodels.relation	
	Þ	Index -> RelationalEntity [com.metamatrix.metamodels.relational	
	Þ	LogicalRelationship -> Relationship [com.metamatrix.metamodels	
	Þ	LogicalRelationshipEnd -> RelationalEntity [com.metamatrix.meta]	
	Þ	MultiplicityKind	
	Þ	🖀 NullableType	•
<		>	

Figure C.27. Datatype Hierarchy View

C.2.10. System Catalog View

To open **Teiid Designer's System Catalog** view, select the main menu's **Window > Show View > Other...** and select the **Teiid Designer > System Catalog** view in the dialog..

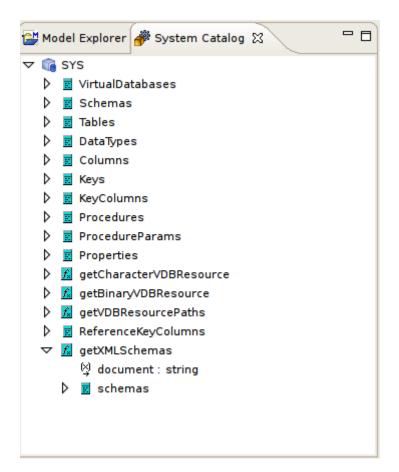


Figure C.28. System Catalog View

C.2.11. SQL Reserved Words View

To open **Teiid Designer's SQL Reserved Words** view, select the main menu's **Window > Show View > Other...** and select the **Teiid Designer > SQL Reserved Words** view in the dialog.

🗖 SQL Reserved Words 😫 🖳 Problems	- 8
Filter Text (? = any character, * = any String):	
Se	
Matching Reserved Words:	
SEARCH	
SECOND	
SELECT	
SENSITIVE	
SESSION_USER	
SET	

Figure C.29. System Catalog View

You can also display the view by selecting the the main menu's **Metadata > Show SQL Reserved Words** action as shown below.

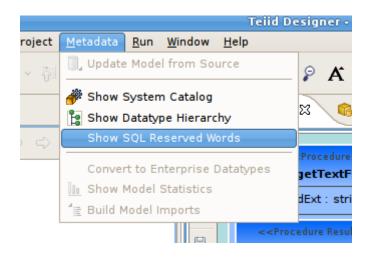


Figure C.30. SQL Reservered Words Action

C.2.12. Model Extension Definition Registry View (MED Registry View)

To open **Teiid Designer's MED Registry** view, select the main menu's **Window > Show View > Other...** and select the **Teiid Designer > Model Extension Registry** view in the dialog.

The **Model Extension Registry** view shows the currently registered MEDs. Registered MEDs can be applied to models in the workspace (see *Section 5.3, "Managing Model Object Extensions*"). The Model Extension Registry view looks like this:

Built-	In Namespace Prefix	Namespace URI	Mode
✓	ext-custom	org.teiid.designer.extension.deprecated	Relat
	mymodelextension	mymodelextension	Relat
\checkmark	rest	org.teiid.designer.extension.rest	Relat
✓	salesforce	org.teiid.designer.extension.salesforce	Relat
✓	sourcefunction	org.teiid.designer.extension.sourcefunction	Relat

Figure C.31. System Catalog View

You can also open the view by selecting the MED Editor toolbar action in the right corner of shared sub-editor header section.

For each registered MED, the namespace prefix, namespace URI, extended model class, version, and description is shown. In addition, a flag indicating if the MED is built-in is shown. The Model Extension Registry view has toolbar actions that register a workspace MED file, unregister a userdefined MED, and copy a registered MED to the workspace. All these actions are also available via a context menu.

A MED registry keeps track of all the MEDs that are registered in a workspace. Only registered MEDs can be used to extend a model. There are 2 different types of MEDs stored in the registry:

- **Built-In MED** these are registered during Designer installation. These MEDs cannot be updated or unregistered by the user.
- User-Defined MED these are created by the user. These MEDs can be updated, registered, and unregistered by the user.

Note: When a workspace MED is registered it can be deleted from the workspace if desired. The registry keeps its own copy. And a registered MED can always be copied back to the workspace by using the appropriate toolbar or context menu action.

C.2.13. Guides View

To open **Teiid Designer's Guides** view, select the main menu's **Window > Show View > Other...** and select the **Teiid Designer > Guides** view in the dialog.

The **Guides** view provides assistance for many common modeling tasks. The view includes categorized Modeling Actions and also links to 'Cheat Sheets' for common processes. 'Cheat Sheets' are an eclipse concept for which Teiid Designer has provided contributions (see *Section C.2.15, "Cheat Sheets View"*). The Guides view is shown below:

😭 Guides 🛱 🖓 Status 🗖	
Modeling Actions 🖲	Â
- Action Sets	
Execute selected action	
Model JDBC Source	
💣 Create Teiid Model Project	
🐝 Create JDBC connection	
Create source model for JDBC data source	
👺 Preview Data	
💦 Create VDB	
🚱 Execute VDB	
Description	
<no action="" selected=""></no>	Ξ
✓ Cheat Sheets Cheat sheets for Teiid Designer use cases	
⑦ Create Model from JDBC Source	
⑦ Create Model from Flat File Source	
⑦ Consume a SOAP Web Service	
⑦ Create Model from XML Local File Source	
⑦ Create Model from XML Remote Source	
⑦ Create multi-source VDB	
⑦ Create and test a VDB	

Figure C.32. Guides View

The upper 'Action Sets' section provides categorized sets of actions. Select the desired category in the dropdown, then the related actions for the selected category are displayed in the list below it. Execute an action by clicking the 'Execute selected action' link or double-clicking on the action.

The lower 'Cheat Sheets' section provides a list of available 'Cheat Sheet' links, which will launch the appropriate Cheat Sheet to guide you step-by-step through the selected process.

C.2.14. Status View

To open **Teiid Designer's Status** view, select the main menu's **Window > Show View > Other...** and select the **Teiid Designer > Status** view in the dialog.

The **Status** view provides a quick overview status of the selected project. A sample Status view for a project is shown below:

🍓 Guides	🎆 Status ස	
Project	Status	Enable Cnange Project
Project:	TestProj	
	Source Connections	
	Sources	
<u>4</u>	XML Schema	
×	Views	
	VDBs	
×	Model validation (OFF)	
	Test	

Figure C.33. Status View

The status view is broken down into common project areas:

- Source Connections all Source Connections are fully defined.
- Sources Source Models exist.
- XML Schema XML Schemas exist.
- Views View Models exist.
- VDBs VDBs exist and are deployable.
- Model Validation (Status) all Models pass validation.
- Test all defined VDBs pass validation.

The status of each area is denoted by an icon: A green check indicates OK, a red 'x' indicates errors and a 'warning' icon indicates potential problems. The project can be changed by selecting the 'Change Project' button.

C.2.15. Cheat Sheets View

To open **Cheat Sheets** view, select the main menu's **Window > Show View > Other...** and select the **Help > Cheat Sheets** view in the dialog.

The **Cheat Sheets** view is a standard Eclipse Help concept. Cheat Sheets provide step-bystep assistance for common process workflows. Teiid Designer has contributed to the Eclipse help framework to provide assistance for many common modeling tasks. The Guides View (see *Section C.2.13, "Guides View*") provides links to these Cheat Sheets, as previously described. A sample Cheat Sheet is shown below:



Figure C.34. Cheat Sheet Sample

C.3. Editors

Editors are the UI components designed to assist editing your models and to maintain the state for a given model or resource in your workspace. When editing a model, the model will be opened in a **Model Editor**. Editing a property value, for instance, will require an open editor prior to actually changing the property.

Any number of editors can be open at once, but only one can be active at a time. The main menu bar and toolbar for Teiid Designer may contain operations that are applicable to the active editor (and removed when editor becomes inactive).

Tabs in the editor area indicate the names of models that are currently open for editing. An asterisk (*) indicates that an editor has unsaved changes.

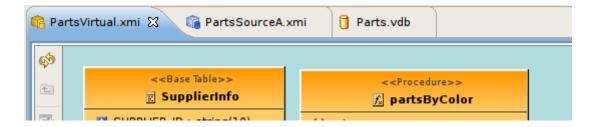


Figure C.35. Editor Tabs

By default, editors are stacked in the editors area, but you can choose to tile them vertically, and or horizontally in order to view multiple models simultaneously.

🏻 PartsSourceA.xmi 🛛					😭 Pa	rtsVirtual.xmi	щ
	: string(255) R : string(30)	SHIPPER_ID	ME : string(Table>> IER_PARTS) : string(10 ring(50) short			A SUPPL A PART A QUAN A SHIPP A SUPPL A SUPPL A SUPPL	<base table=""/> SupplierIr IBR_ID : string ID : string(50) TITY : short ER_ID : short IER_NAME : s' IER_STATUS : IER_CITY : str IER_STATE : s
Package Diagram	Table Editor				E P	∢ ackage Diagrar	m Table E
Models							
Model	Path	Synchronized?	Visible?	Source Na	ame	Translator	JNDI Name
PartsSourceA.xmi	/PartsProject	√	Z	PartsSou	rceA	db2	PartsSourc
PartsSourceB.xmi	/PartsProject /PartsProject	N N	V	PartsSou	rceB	sqlserver	PartsSourc
Other Files						VDB Data Ro	oles
File Path Sy	nchronized? D	escription			dd	Data Role	Descriptio
	Syr	nchronize All					

Figure C.36. Viewing Multiple Editors

The Teiid Designer provides main editor views for XMI models and VDBs.

The Model Editor contains sub-editors which provide different views of the data or parts of data within an XMI model. These sub-editors, specific to model types are listed below.

- Diagram Editor All models except XML Schema models.
- Table Editor All models.
- Simple Datatypes Editor XML Schema models only.
- Semantics Editor XML Schema models only.
- Source Editor XML Schema models only.

The *VDB Editor* is a single page editor containing panels for editing description, model contents and data roles.

In addition to general Editors for models, there are detailed editors designed for editing specific model object types. These "object" editors include:

- **Transformation Editor** Manages Transformation SQL for Relational View Base Tables, Procedures and XML Web Service Operations.
- **Choice Editor** Manages properties and criteria for XML choice elements in XML Document View models.
- Input Editor Manages Input Set parameters used between Mapping Classes in XML Document View models.
- Recursion Editor Manages recursion properties for recursive XML Elements in XML Document View models.
- Operation Editor Manages SQL and Input Variables for Web Service Operations.

C.3.1. Model Editor

The Model Editor is comprised of sub-editors which provide multiple views of your data. The Diagram Editor provides a graphical while the Table Editor provides spreadsheet-like editing capabilities. This section describes these various sub-editors.

C.3.1.1. Diagram Editor

The Diagram Editor provides a graphical view of the a set of model components and their relationships.

Several types of diagrams are available depending on model type. They include:

Package Diagram

Custom Diagram

• в-€

Transformation Diagram

• Rapping Diagram

• ⊷∉

Mapping Transformation Diagram

You can customize various diagram visual properties via Diagram Preferences.

Each diagram provides actions via the Main toolbar, diagram toolbar and selection-based context menus. These actions will be discussed below in detail for each diagram type.

When a **Diagram Editor** is in focus, a set of common diagram actions is added to the application's main toolbar.



Figure C.37. Main Toolbar Diagram Actions

- The actions include:
 - ₽

Zoom In



Zoom to Level

Zoom Out

• A Increase Font Size

"∧

Decrease Font Size

• Perform Diagram Layout

C.3.1.1.1. Package Diagram

The Package Diagram provides a graphical view of the contents of a model container, be it the model itself, a relational catalog or schema.

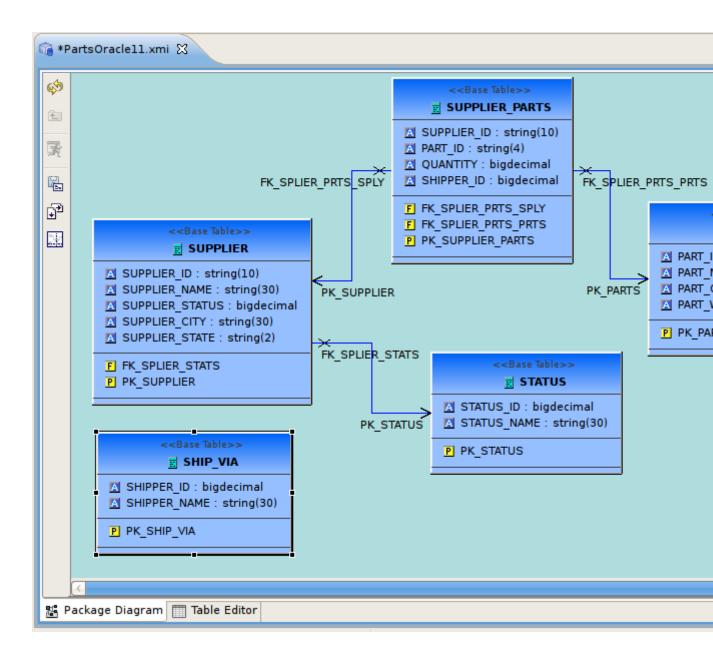


Figure C.38. Package Diagram Example

[•] Package Diagram toolbar actions include:

```
• 🧒
```

Refresh Diagram - Re-draws diagram.

• 🔁

Show Parent Diagram - Navigates to diagram for parent object (if available).

• 록

Preview Data - Executes a simple preview query (SELECT * FROM).

• 🖥

Save Diagram as Image - Save the diagram image to file in JPG or BMP format.

• 🗗

Modify Diagram Printing Preferences - Modify page layout information for printing diagrams. Includes margins, orientation, etc...

Show/Hide Page Grid - Show current page boundaries as grid in diagram.

Context menus provide a flexible means to edit model data, especially from Package Diagrams. Each Package Diagram represents the contents of some container (i.e. Model, Category, Schema, etc...), so New Child, New Sibling and New Association actions are almost always available in addition to standard Edit actions (Delete, Cut, Copy, Paste, Rename, Clone).

A sample context menu for a relational base table is shown below.

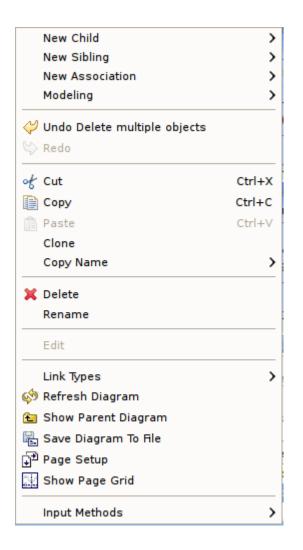


Figure C.39. Package Diagram Context Menu

C.3.1.1.2. Custom Diagram

The **Custom Diagram** represents a view of user-defined model objects. Unlike **Package Diagrams**, **Custom Diagrams** can contain objects that are not only unrelated, but can be from different containers and even models.

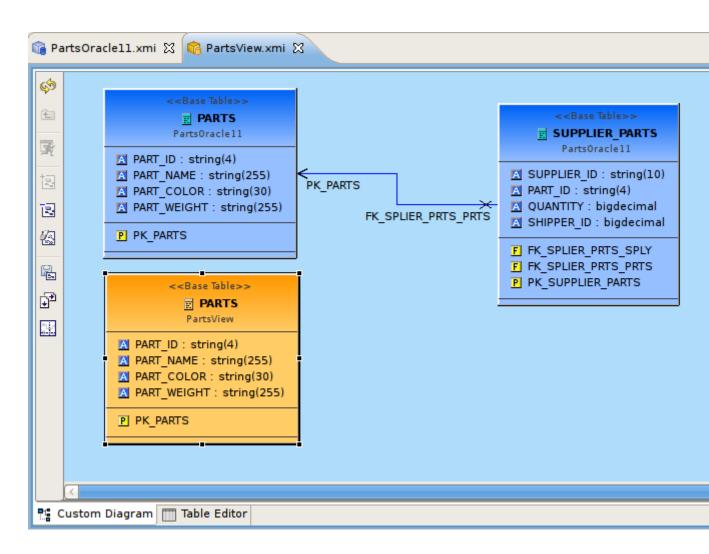


Figure C.40. Package Diagram Example

- Custom Diagram toolbar actions include:
 - ଂ 🏟

Refresh Diagram - Re-draws diagram.

• 😢

Show Parent Diagram - Navigates to diagram for parent object (if available).

• 록

Preview Data - Executes a simple preview query (SELECT * FROM).

• 🕄

Add To Diagram - Add objects selected in Model Explorer to diagram.

• 🖪

Remove From Diagram - Removed objects selected in diagram from diagram.

• 🛵

Clear Diagram - Remove all objects from diagram.

• 🖳

Save Diagram as Image - Save the diagram image to file in JPG or BMP format.

• 🗗

Modify Diagram Printing Preferences - Modify page layout information for printing diagrams. Includes margins, orientation, etc...



Show/Hide Page Grid - Show current page boundaries as grid in diagram.

Since **Custom Diagrams** do not represent represents the contents of container objects (i.e. Model, Category, Schema, etc...) its **context menus** are limited to adding/removing objects from diagram and basic diagram-related display options.

C.3.1.1.3. Transformation Diagram

The **Transformation Diagram** represents a view of the relationships defined by the source inputs described in a view table's SQL transformation.

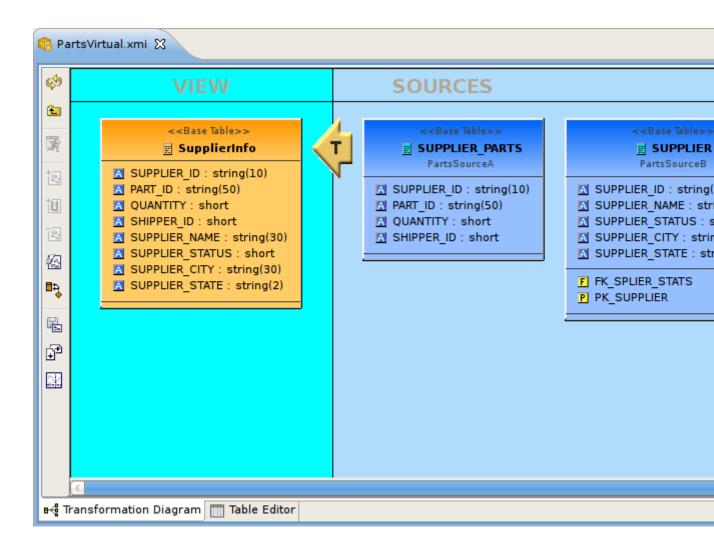


Figure C.41. Transformation Diagram Example

- Transformation Diagram toolbar actions include:
 - ଂ 🏟

Refresh Diagram - Re-draws diagram.

• 🔁

Show Parent Diagram - Navigates to diagram for parent object (if available).

• 록

Preview Data - Executes a simple preview query (SELECT * FROM).

• 🕄

Add Transformation Sources - Add selected sources to transformation.

· 🔃

Add Union Transformation Sources - Add selected sources as union sources.

• 🖪

Remove Transformation Sources - Removed sources selected in diagram from transformation.

• 🐼

Clear Transformation - Remove all sources from transformation.

• 🖬

Open Transformation Reconciler dialog

•

Save Diagram as Image - Save the diagram image to file in JPG or BMP format.

• 🗗

Modify Diagram Printing Preferences - Modify page layout information for printing diagrams. Includes margins, orientation, etc...

-	
•	10.0
	1 C 1 C 1 C 1
	de la cinci



Show/Hide Page Grid - Show current page boundaries as grid in diagram.

Context menus for the

C.3.1.1.4. Mapping Diagram

The **Mapping Diagram** represents a view of the mapping between virtual mapping class columns and XML document elements. This mapping defines how source data is transformed from row-based results into XML formatted text.

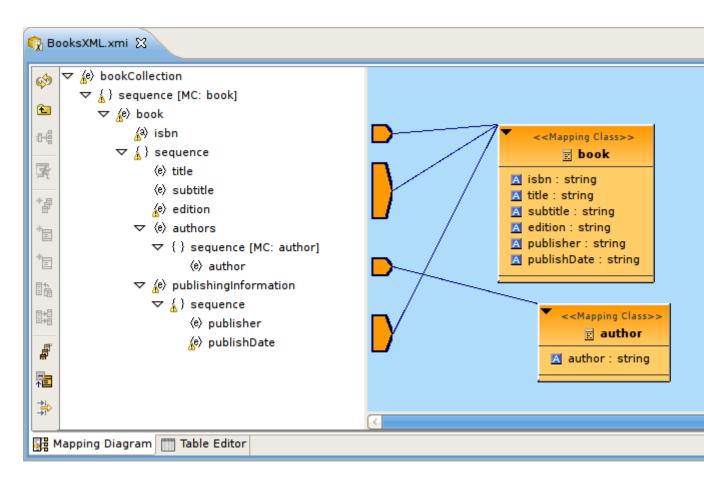


Figure C.42. Mapping Diagram Example

- Mapping Diagram toolbar actions include:
 - ଂ 🏟

Refresh Diagram - Re-draws diagram.

° 🔁

Show Parent Diagram - Navigates to diagram for parent object (if available).

• ₽+<mark>8</mark>

Show Mapping Transformation Diagram - Show detailed mapping transformation diagram for selected mapping class.

• 醒

Preview Data - Executes a simple preview query (SELECT * FROM).

• *₽

Generate Mapping Classes - Generate mapping classes for the selected XML document root element.

•*=

New Mapping Class - Insert new mapping class referenced to the selected XML document element or attribute..

• * 🗉

New Staging Table - Insert new staging table referenced to the selected XML document element or attribute.

• 🛯 🖥

Merge Mapping Classes - Merge selected mapping classes.

• 🔡

Split Mapping Class - Split selected mapping class.

• 🗗

Display All Mapping Classes

•

Show Mapping Class Columns

‡‡⊳ ∔T

Filter Displayed Mapping Classes with Selection

Context menus for **Mapping Diagrams** provide Edit capability to the mapping class in addition to mapping class manipulation actions (i.e. Merge Mapping Classes, Split Mapping Class, etc..)

C.3.1.1.5. Mapping Transformation Diagram

The **Mapping Transformation Diagram** is identical to a Transformation Diagram except for displaying an Input Set and possibly Staging Tables as sources for the Mapping Class's transformation.

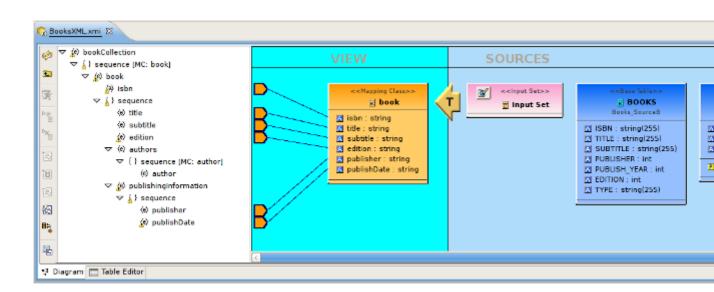


Figure C.43. Mapping Transformation Diagram Example

- Mapping Transformation Diagram toolbar actions include:
 - ଂ 🧒

Refresh Diagram - Re-draws diagram.

• 🖭

Show Parent Diagram - Navigates to diagram for parent object (if available).

• 록

Preview Data - Executes a simple preview query (SELECT * FROM).

• •*

New Mapping Link - Create a mapping link between selected mapping extent (i.e. XML element or attribute) and mapping class column.

• 🛰

Remove Mapping Link - Delete mapping link between selected mapping extent (i.e. XML element or attribute) and mapping class column.

• 🖪

Add Transformation Sources - Add selected sources to transformation.

· 🔃

Add Union Transformation Sources - Add selected sources as union sources.

• 🖪

Remove Transformation Sources - Removed sources selected in diagram from transformation.

• 🕼

Clear Transformation - Remove all sources from transformation.

• 🗖 🛓

Open Transformation Reconciler dialog

• 🖫

Save Diagram as Image - Save the diagram image to file in JPG or BMP format.

· 🗗

Modify Diagram Printing Preferences - Modify page layout information for printing diagrams. Includes margins, orientation, etc...

Context menus for **Mapping Transformation Diagrams** identical capabilities to the Transformation Diagram with the addition of managing and editing Input Sets.

C.3.1.2. Table Editor

The **Table Editor** provides a table-based object type structured view of the contents of a model. The figure below shows a relational model viewed in the **Table Editor**. Common object types are displayed in individual folders/tables. All base tables, for instance, are shown in one table independent of their parentage.

Base lable	es 🖪 Columns	S					
ocation	Name	Name In Source	System	Cardinality	Supports Up	Materialized	Logical Rel
	PARTS	PARTS	false	16	true	false	
	SHIP_VIA	SHIP_VIA	false	3	true	false	
	STATUS	STATUS	false	3	true	false	
	SUPPLIER	SUPPLIER	false	16	true	false	
	SUPPLIER_PA	SUPPLIER_PARTS	false	227	true	false	
	SUPPLIER_PA	SUPPLIER_PARTS	false	227	true	false	

Figure C.44. Table Editor Example

You can customize Table Editor properties via Table Editor Preferences.

These are the primary features of the Table Editor:

- Edit existing properties.
- Add, remove or edit objects, via the main Edit menu and context menu (Cut, Copy, Paste, Clone, Delete, Rename, Insert Rows).
- Paste information from your clipboard into the table.
- Print your tables.

When a Table Editor is in focus, the Insert Table Rows action

is added to the application's main toolbar.

A few Table Editor actions are contributed to the right-click menu for selected table rows. These actions, described and shown below include:

Paste - Paste common spreadsheet data (like Microsoft Excel) to set object properties.

• Table Editor Preferences - Change table editor preferences, including customizing visible properties.

Т

^{• 🛱}

• Insert Rows - Create multiple new sibling objects.

ଂ 🏟

Table - Refreshes the contents of the current Table Editor to insure it is in sync with the model.

R

Nar	me		Name In Sou	urce	Sy	ste
PAF SH ST/ SU SU		New Child New Sibling New Association Modeling	B1070		> > > >	se se se
		Undo Redo				
	Đ	Cut Copy Table Paste		Ctrl+ Ctrl+		
9 Diagr		Clone Copy Name			>	ol
:e parti ndled e		Delete Rename		F	2	es
ndled e ndled e		Open Edit		Ctrl+	0	.u .u
dler co Model b Model F Model F Model F Model F ndled e	19 ())	Add to Metadata A Open in Relations <u>R</u> un As <u>D</u> ebug As <u>P</u> rofile As <u>V</u> alidate Mark as Deployab T <u>e</u> am Comp <u>a</u> re With	hip Navigator		>>> >>>	.u :s :s :s :s .u .u
L.4 KB 2.5 KB).8 KB L.1 KB		Rep <u>l</u> ace With Guvnor Table Editor Prefe Insert Rows Refresh Table G image		07 Se	> >	DI D1 D1

Figure C.45. Table Editor Example

C.3.1.2.1. Editing Properties

You can edit properties for an object by double-clicking a table cell.

For String properties, the table cell will become an in-place text editor field.

Base	Tables 🛛 Columns 📕	Foreign
ation	Name	Na
	CATEGORIES	C4
	CUSTOMERCUSTOMERDE	MO CL
	CUSTOMERDEMOGRAPHIC	S CL
	CUSTOMERS	CL
	EMPLOYEES	EN

Figure C.46. Editing String Property

If a property is of a boolean (true or false) type or has multiple, selectable values, a combo box will be displayed to change the value.

Support	s U
true	
true	-
true	
Frite	

Figure C.47. Editing Boolean Value

. Searchability	Current
ALL_EXCEP	false
ALL_EXCEP	false
ALL_EXCEP	false
ARCHABLE -	false
SEARCHABLE	
ALL_EXCEPT_LI	KE
LIKE_ONLY	
UNSEARCHABLE	
SEARCHABLE	false
SEARCHABLE	false

Figure C.48. Editing Multi-Value Property

For multi-valued properties where the available values are dynamic (i.e. can change based on available models or data), a picker-button ("....") will be displayed.

Datatype		1
int : xs:lo	ng	
short : xs	:int	
long : xs:	integer	
string (Pat	h=http://www.w3.org/200	
long : xs:	integer	1
string		
string		
long : xs:	integer	
chring	(h)	

Figure C.49. Editing Multi-Value With Picker

An example of of this type is the relational column datatype property. Editing via the table cell and clicking the "..." button for datatype will display the following dialog.

Datatype	(
int : xs:long	
short : xs:int	
long : xs:integer	
string (Path=http://www.w3.org/200	
long : xs:integer	
string	
string	
long : xs:integer	
string	

Figure C.50. Editing Datatype Values

C.3.1.2.2. Inserting Table Rows

The Insert Rows action provides an additional way to create objects in a model. Insert Rows action performs the same function as Insert Sibling action, but allows you to create multiple children at the same time. All new rows will correspond to an object of the same type as the selected object and be located under the same parent as the selected object.

To Insert Rows in a table:

Step 1: Select a table row to insert rows after.

Step 2: Right-click select "Insert Rows" action or select the Insert Rows action on the main toolbar. The following dialog will be displayed.

۹	Insert Table Row	s 🛞
Insert Rows (M 20 rows will be in:	l aximum of 1000) serted	
Number of Rows:	20	
?		Cancel OK

Figure C.51. Editing String Property

Step 3: Edit the Number of Rows value in the dialog, or use the up/down buttons to change the value.

Step 4: Select OK in dialog.

The desired number of rows (new model objects) will be added after the original selected table row.

C.3.1.3. Simple Datatypes Editor

The Simple Datatype Editor provides a form-based properties view of XML Schema data.

🕞 BookDatatypes.xsd 🕱			
New Delete	Simple Datatype Editor		
Delete	Simple Datatype Editor		
type filter text Clear	Target Namespace: http://www.metamatrix.com/XMLSche (change)	ma/DataSe	ets/Books/BookData
Datatypes in BookDatatypes.xsd:	- Identification		
BookType	Name: ISBN (change)		
CommaSeparatedList	Description ISBN, using very simple formatting rules.		
ISBN			
PublicationDate			
PublicationYear	▼ Inheritance		
	Base datatype: <u>string</u> (http://www.w3.org/2001/XMLSche	ema) <u>(cha</u>	nge)
	Create subtype of this datatype		Restrictions: 🗌 A
	Open in Datation Historichi davi		
	Open in Datatype Hierarchy view		
	 Enterprise Information in this section is required in order to use this 	s type in ct	her models
		s type in ot	and models.
	Enterprise Datatype		
	Runtime type: string 🗌 Fixed 🗹 Use	Default In	herited from <u>string</u>
Hide Built-in Datatypes	✓ Length		
Built-in Datatypes:		Add Des	cription
anySimpleType			
anyURI	Minimum Length 🛛 🗘 🗆 Fixed 🗹 Use Default		
base64Binary bigdecimal	Maximum Length 🛛 🗘 🗌 Fixed 🗹 Use Default	Add Des	cription
biginteger			
blob	✓ Format		
boolean	Whitespace Collapse 🗘	🗌 Fixed	🗹 Use Default
byte	Enumerations Value Description	Add	🗸 Use Default
char		Edit	
clob			
date dateTime		Remove	
datelime decimal	Patterns Value Description	Add	🗹 Use Default
double	[0-9.\X]{10.13}	Edit	
duration		Remove	
ENTITY			
float			
Simple Datatypes III Ta	ble Editor 🚯 Semantics 🧮 Source		

Figure C.52. Editing String Property

C.3.1.4. Semantic Editor

The Semantic Editor is a tree based editor for XML Schema elements and attributes.

C.3.1.5. Source Editor

The Source Editor is a simple text editor which is aware of XML Schema formatting rules.

C.3.1.6. Model Object Editors

The Model Object Editors represent specialized sub-editors which are available for specific model object types.

• For details, select a specific editor listed below:

 Section 5.2.1, "Transformation Editor" Section 5.2.2, "Input Set Editor (XML)" Section 5.2.3, "Choice Editor (XML)" Section 5.2.4, "Recursion Editor (XML)" Section 5.2.5, "Operation Editor"

C.3.2. VDB Editor

A **VDB**, or **virtual database** is a container for components used to integrate data from multiple data sources, so that they can be accessed in a federated manner through a single, uniform API. A **VDB** contains models, which define the structural characteristics of data sources, views, and Web services. The **VDB Editor**, provides the means to manage the contents of the **VDB** as well as its deployable (validation) state.

The VDB Editor, shown below, contains a upper and lower panels. The upper panel contains the Models tab and an Other Files tab. The lower panel contains tabs for managing Data Roles, the VDB Description and Translator Overrides.

Model	Path	02	2	Source Name	Translator	JNDI Name	Descri
Products.xmi		☑		Products	oracle	Products	
😚 Updates.xmi	/UpdateVGTest	✓	✓				
Synchronize All	escription Translate	or Overri	des				
Synchronize All	escription Translate	or Overri	des				
Synchronize All Data Roles VDB De	Description	or Overri	des				
Synchronize All Data Roles VDB De Data Role	Description	or Overri	des				
Synchronize All Data Roles VDB De Data Role	Description	or Overri	des				

Figure C.53. VDB Editor

You can manage your VDB contents by using the *Add* or *Remove* models via the buttons at the right.

Set individual model visibility via the *Visibility* checkbox for each model. This provides low level data access security by removing specific models and their metadata contents from schema exposed in GUI tools.

In order for a VDB to be fully queryable the "Source Name", "Translator" and "JNDI Names" must have valid values and represent deployed artifacts on your Teiid server.

If you have Designer runtime plugins installed, and have a Teiid server running, you can select a source model in the VDB Editor and right-click select "Change Translator" or "Change JNDI Data Source" which will allow you to select any applicable artifacts on your server.

Models Other Files					
Model	Path	02	Q.	Source Name	Translato
🕞 DataDictionary.xsd	/FinancialsDemo/EnterpriseDataLayer	V	V		
💏 EU_Customers.xmi	/FinancialsDemo/DataSources			EU_Customers	mysql5
🕞 EU_Customers_DDC.xmi	/FinancialsDemo/EnterpriseDataLayer	- 🗹	√		loopback
🕞 EU_Customers_VBL.xmi	/FinancialsDemo/VirtualBaseLayer	- 🗹	√_		metama
📬 Products.xmi	/FinancialsDemo/DataSources	- 🗹		Products	modesh
🕞 Products_DDC.xmi	/FinancialsDemo/EnterpriseDataLayer	- 🗹	✓		mysgl
😚 Products_VBL.xmi	/FinancialsDemo/VirtualBaseLayer				mysql mysql5
🙃 QuataSan <i>ica</i> ymi	/EnancialeDome/EnterpriceDatal aver				cipevin

Figure C.54. Change Translator or Data Source Actions

If you have a default Teiid server instance defined and connected the translator and JNDI table cells will contain drop-down lists of available translator and JNDI names available on that server.

C.3.2.1. Editing Data Roles

Teiid Designer provides a means to create, edit and manage data roles specific to a VDB. Once deployed within a Teiid server with the security option turned on (by default) any query run against this VDB via a Teiid JDBC connection will adhere to the data access permissions defined by the VDB's data roles.

The VDB Editor contains a VDB Data Roles section consisting of a List of current data roles and *New...*, Edit... and Remove action buttons.

Data Role	Description
NoAccountHoldingsAccess	Client cannot access customer account holdings (financial product) info
VirtualOnly	Client users can only access virtual layer. Access requests to physical sources

Figure C.55. VDB Data Roles Panel

Clicking *New...* or *Edit...* will launch the *New VDB Data Role* editor dialog. Speicify a unique data role name, add a optional description and modify the individual model element CRUD values by check or unchecking entries in the models section.

E	New VDB	Data Rol	e				×
New VDB Data Role							_ N.
Select Finish to save data role							- M
							R
Name Inventory							
Description							
							~
Allow usage of temporary t	ables						
✓ Apply this role to All Users							
Mapped Role Names							
							<u>A</u> dd
							Remove
							Edit
							<u>E</u> dit
Models							
Model		Create	Read	Update	Delete	Execute	Alter
🗢 🎆 ProductsSQL.xmi			</td <td>√</td> <td></td> <td>-</td> <td>-</td>	√		-	-
ProductData ProductSymbols			⊻	⊻		V	⊻
🕨 🧮 ProductSymbols			\checkmark	\checkmark			
System Tables Access							
✓ Allow this role to access \$	SYSADMIN model						
?					Cancel		<u>F</u> inish
				_			

Figure C.56. VDB Data Roles Tab

C.3.2.2. Editing Translator Overrides

Teiid Designer provides a means to create, edit and manage translator override properties specific to a VDB via the Tranlator Overrides tab. A translator override is a set of non-default properties targeted for a specific source model's data source. So each translator override requires a target translator name like "oracle", db2, mysql, etc. and a set of non-default key-value property sets.

The *VDB Editor* contains a *Tranlator Overrides* section consisting of a List of current tranlator overrides on the left, a properties editor panel on the right and *Add* (+) and *Remove* (-) action buttons on the lower part of the panel.

*UpdateVG_VDB.vd	db 🖾						
Models Other Files							
Model	Path	02	P	Source Name	Translator	JNDI Name	Descriptio
💼 Products.xmi	/UpdateVGTest	1	-1	Products	oracle	Products	
👘 Updates.xmi	/UpdateVGTest	√	⊿				
Synchronize All	scription Translator	r Override					
	scription mansiator	overnu	es				
Overridden Trans oracle-1	lators	Descripti	on:				
		Property	,			Value	
				Type Name		oracle	
		Databas					
		Databas	se Versi	on			
		Executio	on Facto	y Class name		org.teiid.tra	anslator.jdbc
		Is Immu	ıtable			🏶 true	
		Max nur	mber of	dependent values	across all IN	50	
		Max nur	mber of	IN predicate entri	es	1000	
		Max Pre	pared Ir	isert Batch Size		2048	
		name				translator-o	racle
		property	y				
		Requrie	s Criteri	a		false	
		Support	-			ANY	
		Support	s Full O	uter Joins		🏶 false	
+ -		+ -	0_				

Figure C.57. VDB Translator Overrides Tab

To override a specific translator type, select the add translator action (+). If a default Teiid server instance is connected and available the Add Translator Override dialog (below) is presented, the user selects an existing tranlator type and clicks OK. Note that the override is only applicable to sources within the VDB, so be sure and select a translator type that corresponds to one of the VDB's source models. The properties panel on the right side of the panel will contain editiable cells for each property type based on the data-type of the property. (i.e. boolean, integer, string, etc.).

loopback	^
metamatrix	
modeshape	
mysql	
mysql5	
olap	
oracle	
postgresql	
salesforce	
sqlserver	
sybase	

Figure C.58. Add Translator Override Dialog

If no default Teiid server instance is available, the "Add New Translator Override" dialog is presented. Enter a unique name for the tranlator override (i.e. "oracle_override"), a valid translator type name (i.e. "oracle") and click OK. The properties panel on the right side of the panel will allow adding, editing and removing key-value string-based property sets. When editing these properties all values will be treated as type string.

	Add New Translator Override	×				
?	A Teiid server is not available so translator types are not available. A translator name and translator type are both required when creating a translator override.					
	Press "OK" when finished.					
Name:	<u>[</u>					
Type:						
	Cancel OK					

Figure C.59. Add New Translator Override Dialog

C.3.3. Model Extension Definition Editor

The **MED Editor** is a multi-tabbed editor and is used to create and edit user-defined MEDs (*.mxd files) in the workspace. The MED Editor has 3 sub-editors (**Overview, Properties, and Source**) which share a common header section. Here are the MED sub-editor tabs:

• **Overview Sub-Editor** - this editor is where the general MED information is managed. This information includes the namespace prefix, namespace URI, extended model class, and the description. The Overview sub-editor looks like this:

🦻 mymxd.mxd 😫	
🗳 Overview 🔹	1
Namespace Prefix:	mymodelextension
Namespace URI:	mymodelextension
Model Class:	Relational
Version:	1
Description:	This is my model extension
Overview Properties	Source

Figure C.60. Overview Tab

Properties Sub-Editor - this editor is where the MED extension properties are managed. Each
extension property must be associated with a model object type. The Properties sub-editor is
divided into 2 sections (Extended Model Objects and Extension Properties) and looks like
this:

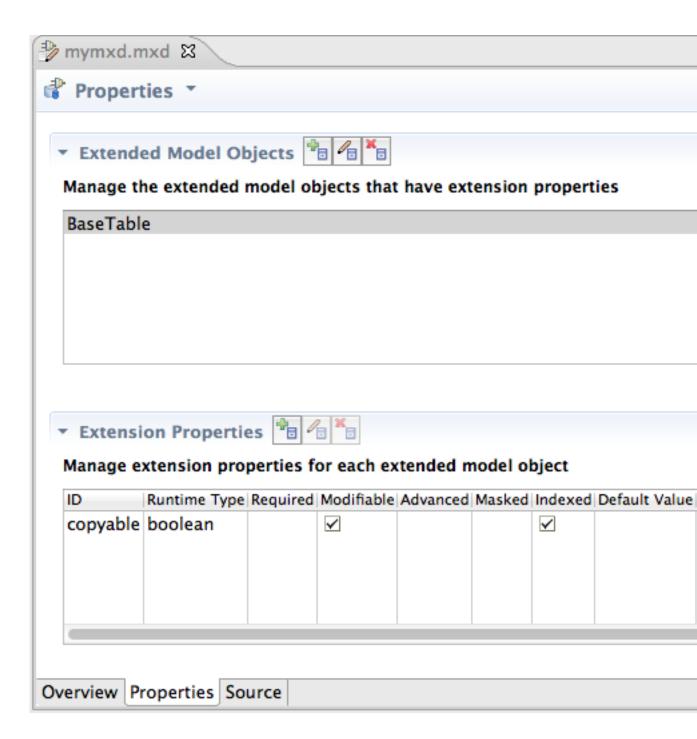


Figure C.61. Properties Tab

• **Source** - this tab is a read-only XML source viewer if you wish to view the details of your MED. This source viewer is NOT editable.

The GUI components on the Overview and Properties sub-editors will be decorated with an error icon when the data in that GUI component has a validation error. Hovering over an error decoration displays a tooltip with the specific error message. Those error message relate to the error messages shown in the common header section. Here is an example of the error decoration:

Namespace URI:

Figure C.62. Text Field With Error

The MED sub-editors share a header section. The header is composed of the following:

- **Status Image** an image indicating the most severe validation message (error, warning, or info). If there are no validation messages the model extension image is shown.
- Title the title of the sub-editor being shown.
- **Menu** a drop-down menu containing actions for (1) adding to and updating the MED in the registry, and (2) for showing the Model Extension Registry View.
- Validation Message this area will display an OK message or an error summary message.
 When a summary message is shown, the tooltip for that message will enumerate all the messages.
- Toolbar contains the same actions as the drop-down menu.

Below is an example of the shared header section which includes an error message tooltip.

🙆 Overview 🔻	1 error detected	گ
C .	Namespace URI: The "namespace	URI" cannot be empty.

Figure C.63. Shared Header Example

C.4. Teiid Designer Main Menu

There are 8 categories of actions on Teiid Designer's main menu bar.

- These categories include:
 - Section C.4.1, "File Menu" Resource management actions.
 - Section C.4.2, "Edit Menu" Standard edit actions including undo/redo.
 - Section C.4.3, "Refactor Menu" Resource actions (i.e. Rename, Move, etc...).
 - Section C.4.5, "Search Menu" Find data within your workspace.
 - Section C.4.6, "Project Menu" Model level actions.
 - Section C.4.7, "Metadata Menu" Custom metadata-related actions.
 - Section C.4.9, "Window Menu" Change perspectives or add/remove views to your perspective.

• Section C.4.10, "Help Menu" - Access available Teiid Designer help documents, Teiid Designer SQL Support Guide and Eclipse Overview information.

 Teiid Designer - PartsProject/PartsSourceA.xmi - JBoss Developer Studio

 File
 Edit
 Refactor
 Navigate
 Search
 Project
 Metadata
 Run
 Window
 Help

Figure C.64. Application Main Menu

C.4.1. File Menu

The File menu provides actions to manage your workspace resources.

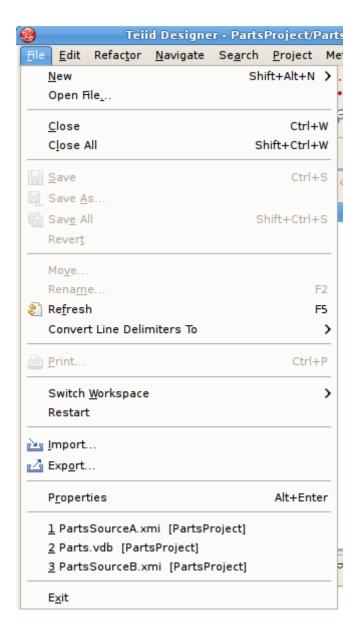


Figure C.65. File Menu

The **New** > sub-menu provides specific actions to create various generic workspace resources as well as Teiid Designer models and VDBs.

<u>File</u> <u>E</u> dit Refac <u>t</u> or	<u>N</u> avigate Se <u>a</u> rch	<u>P</u> roject M	etadata	<u>R</u> un	<u>W</u> indow	<u>H</u> elp
<u>N</u> ew	Sh	ift+Alt+N ゝ	🔺 Tei	id Mode	el Project	
Open File			📬 P <u>r</u> o	oject		
<u>C</u> lose		Ctrl+W	🔐 Fol	der		
C <u>l</u> ose All	S	hift+Ctrl+W			data Mode	el
Save		Ctrl+S	💦 Tei	id VDB		
Save <u>A</u> s			E Ex	ample		
📓 Sav <u>e</u> All	S	hift+Ctrl+S				
Revert			📑 <u>O</u> t	ner		Ctrl+N

Figure C.66. File Menu

- The File menu contains the following actions:
 - · 者

New > Model Project - Create user a new model project.

• 😭

New > Folder - Create new folder within an existing project or folder.

• 💣

New > Model - Create a new model of a specified model type and class using the *Chapter 3, New Model Wizards*.

• 💦

New > Virtual Database Definition - Create a new VDB, or Virtual Database Definition.

- Open File Enables you to open a file for editing including files that do not reside in the Workspace.
- Close (Ctrl+W) Closes the active editor. You are prompted to save changes before the file closes.
- Close All (Shift+Ctrl+W) Closes all open editors. You are prompted to save changes before the files close.
- 📳

Save (Ctrl+S) - Saves the contents of the active editor.

• 🔡

Save As - Enables you to save the contents of the active editor under another file name or location.

•

Save All (Shift+Ctrl+S) - Saves the contents of all open editors.

- Move... Launches a Refactor > Move resource dialog...
- **Rename... (F2)** Launches a Refactor > Rename resource dialog if resource selected, else in-line rename is preformed.
- Refresh Refreshes the resource with the contents in the file system.
- **Convert Line Delimiters To** Alters the line delimiters for the selected files. Changes are immediate and persist until you change the delimiter again you do not need to save the file.

• 🖻

Print (Ctrl+P) - Prints the contents of the active editor. In the Teiid Designer, this action prints the diagram in the selected editor. Allows control over orientation (portrait or landscape), scaling, margins and page order. User can also specify a subset of the pages to print (i.e., 2 through 8).

- Switch Workspace Opens the Workspace Launcher, from which you can switch to a different workspace. This restarts the Workbench.
- Restart Exits and restarts the Workbench.
- ès

Import - Launches the **Import Wizard** which provides several ways to construct or import models..

് പ്

Export - Launches the Export Wizard which provides options for exporting model data.

- **Properties (Alt+Enter)** Opens the **Properties** dialog for the currently selected resource. These will include path to the resource on the file system, date of last modification and its writable or executable state.
- Most Recent Files List Contains a list of the most recently accessed files in the Workbench. You can open any of these files from the File menu by simply selecting the file name.
- Exit Closes and exits the Workbench.

C.4.2. Edit Menu

The **Edit** menu provides actions to manage the content, structure and properties of your model and project resources. The figure below represents the Edit menu presented when a metadata model is selected.

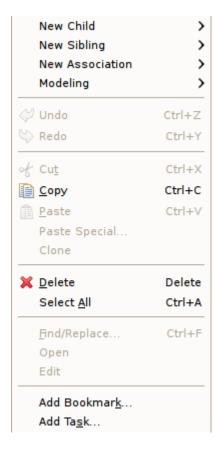


Figure C.67. Edit Menu

- The Edit menu contains the following actions:
 - New > Child This menu is created dynamically to support the creation of whatever types of child objects can be created under the selected object.
 - New > Sibling This menu is created dynamically to support the creation of whatever types of sibling objects can be created under the same parent as the selected object
 - New > Association This menu is created dynamically to support the creation of whatever types of associations can be created with the selected object.
 - **Modeling** > This menu is created dynamically. Various modeling operations are presented based on selected model object type.
 - 🤣

Undo - Reverses the effect of the most recent command.

• 铃

Redo - Reapplies the most recently undone command.

• • t

Cut - Deletes the selected object(s) and copies it to the clipboard.

• 🗈

Copy - Copies the selected object(s) to the clipboard.

• 💼

Paste - Pastes the contents of the clipboard to the selected context.

- Paste Special... Provides additional paste capabilities for complex clipboard objects.
- **Clone** Duplicates the selected object in the same location with the same name. User is able to rename the new object right in the tree.

• 🗙

Delete - Deletes the selected object(s).

- Select All Select All objects in current view.
- Rename Allows a user to rename an object in the tree.
- Find/Replace Launches dialog that can be used to search in the current text view, such as a Transformation Editor.
- **Open** Opens the selected object in the appropriate editor.
- Edit Opens the selected object in the appropriate specialized editor, such as the Choice Editor or Recursion Editor..
- Add Bookmark... This command adds a bookmark in the active file on the line where the cursor is currently displayed.
- Add Task... This command adds a task in the active file on the line where the cursor is currently displayed.

C.4.3. Refactor Menu

The Refactor menu provides Teiid Designer specific actions for file-level changes to the models.

Undo	
Redo	
Rename	
Move	

Figure C.68. Refactor Menu

- The Refactor menu contains the following actions:
 - Undo Undo the last refactor command.
 - Redo Redo the last undone refactor command.
 - Move Move a model from one container (folder or project) to another.
 - Rename Rename a model.

C.4.4. Navigate Menu

Teiid Designer currently does not contribute actions to the Navigate menu. See Eclipse documention for details.

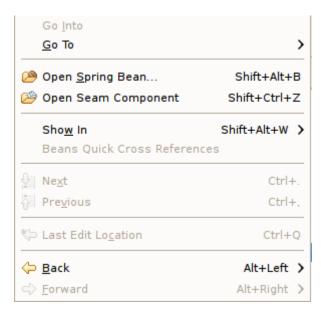


Figure C.69. Navigate Menu

C.4.5. Search Menu

The Search menu presents several specific search options.

🔗 Se <u>a</u> rch	Ctrl+H
📝 <u>F</u> ile	
💖 <u>R</u> emote	
🛷 Pointcut Matches	
🛷 Beans	
💫 Find Seam References	
🖄 Find Seam Declarations	
Text	>
Teiid Designer	>

Figure C.70. Search Menu

Teiid Designer contributes a sub-menu (i.e. Teiid Designer >) to the main search menu, as shown above.

Text	>
Teiid Designer	🗾 📌 Transformations
, (0)	🔗 Metadata
	ind Model Object 👰

Figure C.71. Search Menu

- The individual actions in the Teiid Designer sub-menu are described below:
 - 🚀

Transformations... - Launches the Transformation Search dialog. User can search models in the workspace for matching SQL text. Search results appear in the dialog and user can select and view SQL as well as open desired transformations for editing.

• 🔗

Metadata... - Launches the Search dialog. User can search for models in the workspace by specifying an Object Type, and/or a Data Type, and/or a property value. Search results appear in the *Section C.2.7, "Search Results View"* view, and double-clicking a result will open that model in the appropriate editor.

' 🥟

Find Model Object - Launches the Find Model Object dialog, which can be used to find an object in the workspace by specifying all or part of its name. Selecting the object will open it in the appropriate editor.

C.4.6. Project Menu

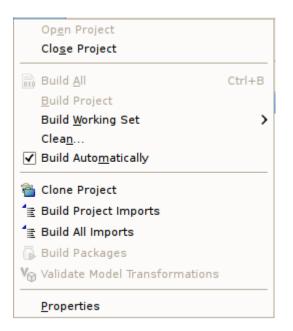


Figure C.72. Project Menu

- The individual actions in the Project menu are described below:
 - Open Project Launches the Open Project dialog.
 - Close Project Closes the currently selected project(s).

Build All - Validates the contents of the entire workspace. Any errors or warnings will appear in the **Problems View**.

- **Build Project** Validates the contents of the selected project(s). Any errors or warnings will appear in the **Problems View**.
- Build Working Set Validates the contents of the selected working set. Any errors or warnings will appear in the Problems View.
- Clean.. Launches the Clean dialog.
- Build Automatically Sets the Build Automatically flag on or off. When on, a check-mark appears to the left of this menu item. When this is turned on, validation of changes is done automatically each time a **Save** is done.
- Clone Project Launches the Clone Project dialog.
- **Build Project Imports** Reconciles all model import dependencies for models contained within the selected project.

[•]

- **Build All Imports** Reconciles all model import dependencies for models contained within the workspace.
- Build Packages TBD
- Validate Model Transformations Revalidates all transformations for the selected view model.
- Properties Displays the operating system's file properties dialog for the selected file.

C.4.7. Metadata Menu

The Metadata menu provides Teiid Designer-specific actions.

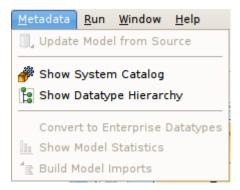


Figure C.73. Metadata Menu

- The Metadata menu contains the following actions:
 - Update Model from Source If the selected model is a relational source model that was originally created via JDBC Import, then the model will be updated based on changes in the database schema.
 - Show System Catalog Opens the Section C.2.10, "System Catalog View".
 - Show Datatype Hierarchy Opens the Section C.2.10, "System Catalog View".
 - Re-resolve References Analyzes references within models to other model components.
 - **Convert to Enterprise Datatypes** Adds an additional property to simple datatypes within your selected schema model to label them as enterprise datatypes.
 - Show Model Statistics Opens the Model Statistics dialog for the selected model.
 - Build Model Imports Reconciles all model import dependencies for the selected model.

C.4.8. Run Menu

Teiid Designer currently does not contribute actions to the Run menu. See Eclipse documention for details.

💁 External Tools 🔹 🔉

Figure C.74. Window Menu

C.4.9. Window Menu

The **Window** menu shown below contains no Teiid Designer-specific actions. See Eclipse Workbench documentation for details.

<u>N</u> ew Window	
New <u>E</u> ditor	
Open Perspective	>
Show View	>
Customi <u>z</u> e Perspectiv	/e
Save Perspective <u>A</u> s.	
<u>R</u> eset Perspective	
<u>C</u> lose Perspective	
Close All Perspective	s
Navi <u>g</u> ation	>
<u>P</u> references	

Figure C.75. Window Menu

The **Preferences...** action launches the Preferences dialog, which can be used to set preferences and default values for many features of Teiid Designer.

Note that these menu items may vary depending on your set of installed Eclipse features and plugins.

If you wish to customize a perspective to include one or more Teiid Designer views, select the **Show View > Other...** action and expand the Teiid Designer category to show the available views.

🔲 Show View 🔅	3
type filter text	
🕨 🗁 Team	
▽ 🗁 Teiid Designer	
Datatypes	
Description	
🏫 Metadata Favorites	
Model Explorer	
Relationship Navigator	
ở System Catalog	
Jags 🖉	
taiid	
🔥 Teiid Model Classes 🗸 🗸	
Cancel OK	

Figure C.76. Show View Dialog

C.4.10. Help Menu

The **Help Menu** shown below contains no Teiid Designer-specific actions. See Eclipse Workbench documentation for details.

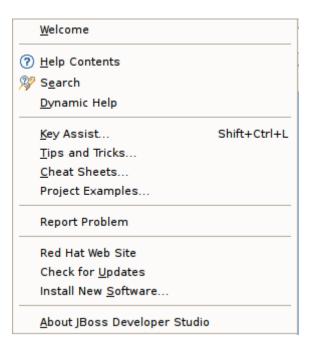


Figure C.77. Help Menu

- The individual actions are described below:
 - **Welcome** Shifts to the Welcome perspective, which contains links to documentation, examples and 'how-to' starting points.

• Help

Contents

- Launches the Help Window. All of Designer's online documentation is accessible from there as well.

Search

- Launches the Help Search view, which can be used to search for phrases in the documentation.

- Dynamic Help Opens the docked dynamic help view.
- Key Assist (Ctrl-Shift-L) ... Launches a dialog describing existing key assist bindings.
- Tips and Tricks... Launches a dialog to select one of any contributed "Tips and Tricks" help pages.
- Cheat Sheets... Launches a dialog to select one of any contributed Eclipse cheat sheets.
- **Project Examples...** A JBoss contributed action which provides quick access to import various project examples into your workspace.
- Report Problem A JBoss contributed action which provides simple problem reporting.
- Check for Updates... provides access to retrieve updates to installed Eclipse software.
- Install New Software... provides access to install new software into your workbench.
- About JBoss Developer Studio Launches the About dialog.

C.5. Teiid Server Management

The preview data and test VDB features in **Teiid Designer** require access to a live Teiid server running on an application server. For release 8.0, **Teiid Designer** requires a **JBoss AS 7.1.1** or higher with a instance of **Teiid 8.2** or higher installed.

To create your Teiid instance:

• Select the New... action in the Teild view or click the New Teild Instance button,

, in the toolbar.

• This will launch the New Teiid Instance dialog shown below.

efine the Teiid !	erver connec	tion properties	required to perfo	m both JDBC and A	Admin tasks.		
Display Name	iid 8.2 Final	on AS7					
JBoss Server							
JBoss Server							New
Used to auto-c determined at	nnect to you execution tin	r VDB as part of	the Execute VDB	and Preview Data a	ctions. The value o	f the <vdbn< th=""><th>ame> will be</th></vdbn<>	ame> will be
Used to auto-c	nnect to you	r VDB as part of	the Execute VDB	and Preview Data a	ctions. The value o	f the <vdbn< td=""><td>ame> will be</td></vdbn<>	ame> will be
Used to auto-c determined at	nnect to you execution tin	r VDB as part of	the Execute VDB	and Preview Data a	ctions. The value o	f the <vdbn< td=""><td>ame> will be</td></vdbn<>	ame> will be
determined at User name Password	user	r VDB as part of ne.	the Execute VDB		ctions. The value o	f the <vdbna< td=""><td>ame> will be</td></vdbna<>	ame> will be
Used to auto-c determined at User name Password	user	r VDB as part of ne.			ctions. The value o	f the <vdbn< td=""><td>ame> will be</td></vdbn<>	ame> will be

Figure C.78. New Teiid Instance Dialog

- Enter a unique display name for your Teiid instance.
- Select the New... button in the JBoss Server section to launch the New Server wizard. Select the JBoss AS 7.1.1 server type under the JBoss Community category and click Next>.

Ī	😣 💷 New Server	
l	Define a New Server	
	Choose the type of server to create	
n		
0		Download additional server adapters
	Select the server type:	
ł	type filter text	
I	JBoss AS 5.1	
1	JBoss AS 6.x	
ł	JBoss AS 7.0	
1	JBoss AS 7.1	
ł	JBoss Enterprise Middleware	
ł	JBoss Application Server 7.1	
te	Server's host name: localhost Server name: JBoss 7.1 Runtime Serv	ver
	Back Next >	Cancel Finish

Figure C.79. New Server Dialog

• On the **JBoss Runtime** page, click the top **Browse...** button to select the installation folder of your **JBoss AS 7.1** server.

-	A O H F				
I	🛞 📋 New Server				
L	JBoss Runtime				
1	JBoss Application Server 7.1				
4	by Red Hat				
n	A JBoss Server runtime references a JBoss installation directory.				
as well as by a "server" which will be able to start and stop instances of.					
1	Name				
1	JBoss 7.1 Runtime				
1	Home Directory Download and install runtime				
I	fond/Testing/Servers/AS_7.1.1_Teiid_8.2_Beta2_10_19_2012 Browse				
1	JRE				
1	Default JRE for JavaSE-1.6				
I					
te	Configuration file: standalone-teiid.xml Browse				
÷					
1					
	Cancel Finish				

Figure C.80. JBoss Runtime Definition

• Then click the bottom **Browse...** button to select the **standalone-teiid.xml** configuration file located under the **standalone/configuration/** folder on your file system. Then click **Finish** to return the the **New Teiid Instance** dialog.

Name 👻	Size	Modified
standalone_xml_history		11:14
application-roles.properties	634 bytes	03/10/2012
application-users.properties	812 bytes	03/10/2012
logging.properties	2.0 KB	03/10/2012
mgmt-users.properties	836 bytes	03/10/2012
💿 standalone.xml	14.6 KB	11:11
💿 standalone-full.xml	20.3 KB	03/10/2012
💿 standalone-full-ha.xml	26.4 KB	03/10/2012
i standalone-ha.xml	19.9 KB	03/10/2012
👩 standalone-teiid.xml	23.1 KB	11:14
teiid-security-roles.properties	106 bytes	10/19/2012
teiid-security-users.properties	95 bytes	10/19/2012

Figure C.81. Teiid Configuration File Selection

• Click **Finish** and your new Teild server configuration will be opened in a **Teild Editor** for viewing. In this editor you can test both Teild admin and JDBC connections.

🐗 Teiid Server	x	JBoss 7.1 Runtime Server				
Teiid Serve	Teild Server					
Overview	Overview					
Display Name	lay Name Teiid 8.2					
Host localhost						
JBoss Server	<u>JB</u>	oss 7.1 Runtime Server				
Administratio	Administration Connection					
Administratio	Administration is performed via the iboss management configuration					
<u>Test Administ</u>	Test Administration Connection					
JDBC Connection						
Username			user			
Password			••••			
Password						
Port			31000			
Test JDBC Cor	nec	tion				

Figure C.82. Teiid Editor