TIBCO ActiveMatrix® Adapter for Database (TIBCO Business Studio™) User's Guide

Software Release 1.3
April 2016

Two-Second Advantage®

TIBCO®

Important Information

Contents

TIBCO Documentation and Support Services

Documentation for this and other TIBCO products is available on the TIBCO Documentation site. This site is updated more frequently than any documentation that might be included with the product. To ensure that you are accessing the latest available help topics, please visit:

https://docs.tibco.com

Product-Specific Documentation

Documentation for TIBCO products is not bundled with the software. Instead, it is available on the TIBCO Documentation site at https://docs.tibco.com/products/tibco-activematrix-adapter-for-database-tibco-business-studio. To directly access documentation for this product, double-click the following file:

TIBCO_HOME/release_notes/TIB_adadbbs_*version*_docinfo.html

where *TIBCO_HOME* is the top-level directory in which TIBCO products are installed. On Windows, the default *TIBCO_HOME* is `C:\tibco`. On UNIX systems, the default *TIBCO_HOME* is `/opt/tibco`.

The following documents for this product can be found on the TIBCO Documentation site:

- *TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) Installation*
- *TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) User's Guide*
- *TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) Example*
- *TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) Release Notes*

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, contact TIBCO Support:

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:

  http://www.tibco.com/services/support

- If you already have a valid maintenance or support contract, visit this site:

  https://support.tibco.com

  Entry to this site requires a user name and password. If you do not have a user name, you can request one.

How to Join TIBCOmmunity

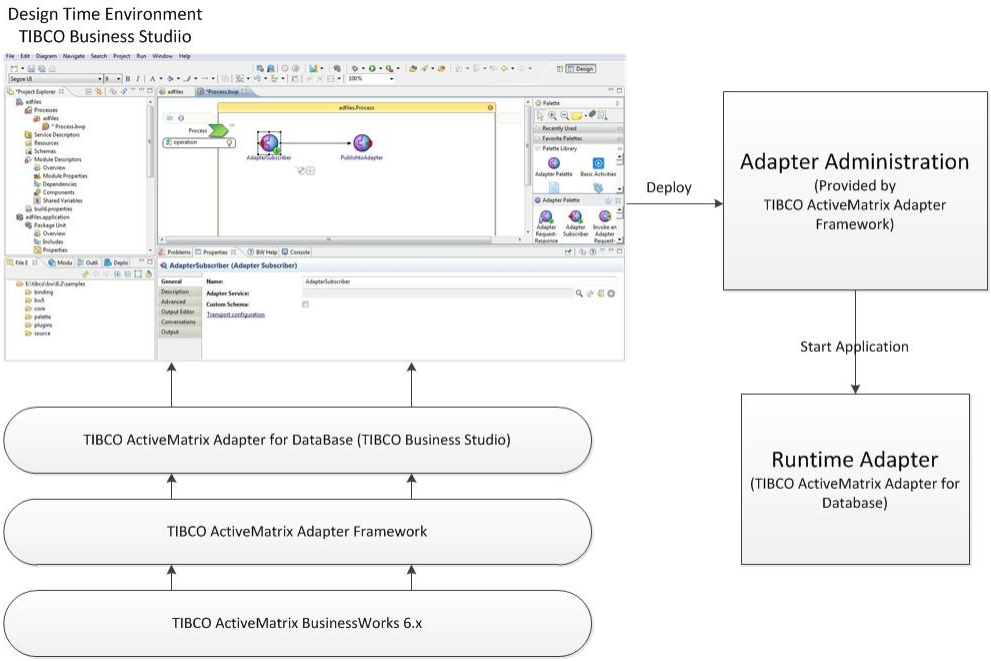
TIBCOmmunity is an online destination for TIBCO customers, partners, and resident experts. It is a place to share and access the collective experience of the TIBCO community. TIBCOmmunity offers forums, blogs, and access to a variety of resources. To register, go to the following web address:

https://www.tibcommunity.com

Product Overview

TIBCO ActiveMatrix® Adapter for Database (TIBCO Business Studio™) provides a design-time environment to configure adapter configurations, Publication Service, Subscription Service, Request-Response Service, and schemas in TIBCO Business Studio.

The following figure shows how TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) works with other TIBCO products:



TIBCO ActiveMatrix BusinessWorks™

TIBCO ActiveMatrix BusinessWorks supports your integration project throughout the project life cycle. It includes a common graphical user interface for configuration, process design, and deployment.

TIBCO Business Studio

TIBCO Business Studio is the Eclipse graphical user interface (GUI) used by TIBCO ActiveMatrix BusinessWorks. TIBCO Business Studio can integrate TIBCO ActiveMatrix® Adapter Framework and TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) to design adapter configurations and business processes, verfiy adapter and processes and generate deployable artifacts in the form of archive files.

TIBCO ActiveMatrix Adapter Framework

TIBCO ActiveMatrix Adapter Framework provides a visual, model-driven development environment for configuring supported TIBCO adapters in TIBCO Business Studio, and integrating them with TIBCO ActiveMatrix BusinessWorks 6 applications and processes. It also provides administration capabilities for deploying the TIBCO adapters developed in TIBCO Business Studio to runtime environments.

TIBCO ActiveMatrix Adapter Framework works as a bridge between TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) and TIBCO ActiveMatrix BusinessWorks 6.

The following installation components are available for TIBCO ActiveMatrix Adapter Framework:

- Design: this component includes all the Eclipse Graphical User Interface (GUI) for TIBCO ActiveMatrix Adapter Framework. You can configure your own adapter configurations and business processes in TIBCO Business Studio with the Eclipse GUI.
- Runtime: this component includes Adapter Administration, which is used to deploy and run Enterprise Archive (EAR) files of adapter projects by using a command line.

You can install this component independently without installing TIBCO ActiveMatrix BusinessWorks 6.x.

TIBCO ActiveMatrix® Adapter for Database

TIBCO ActiveMatrix Adapter for Database is a bidirectional gateway between databases and applications configured for the TIBCO environment. The adapter supports publication, subscription, and request-response interactions.

As shown in the preceding figure, you can configure the adapter through TIBCO ActiveMatrix Adapter Framework in TIBCO Business Studio and TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio). You can also use TIBCO ActiveMatrix BusinessWorks to develop a business process.

After TIBCO ActiveMatrix Adapter Framework is installed, the Adapter palette is available for TIBCO adapters. The Adapter palette contains activities for communicating with configured TIBCO Adapter services. You can add activities and design business processes in TIBCO Business Studio. The configurations of all the features are similar to TIBCO ActiveMatrix Adapter for Database. When you run adapter configurations, TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) calls for the runtime properties in TIBCO ActiveMatrix Adapter for Database.

Getting Started

This tutorial is designed for beginners who want to use TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio).

This tutorial emulates the configurations of the Demo1Oracle project in the *TIB_ADADB_HOME*\demo\bw6\Demo1 directory and uses the Oracle SQL script in this directory to create database tables.

You can perform this exercise on any supported operating system. The databases you can use with this example include Oracle, Microsoft SQL, and DB2. This tutorial describes how to run the example with an Oracle database on a Windows platform.

All the operations are processed in TIBCO Business Studio. See TIBCO Business Studio Overview to get familiar with TIBCO Business Studio.

Before you use the Oracle SQL script, you have to unzip the `Demo1.zip` file to a local directory.

A basic procedure of using the adapter includes the following tasks:

1. Creating database tables
2. Creating a project
3. Creating a database connection
4. Creating adapter configurations
5. Adding the database connection
6. Adding adapter services
7. Specifying a working directory
8. Starting the adapter configurations
9. Changing the table values and receiving a notification
10. Removing the tables from a database

After doing this exercise, you will know how to create tables in your database, create a project, adapter configurations, and adapter services, and insert data into the source table and receive a notification.

Tables and Data Flow

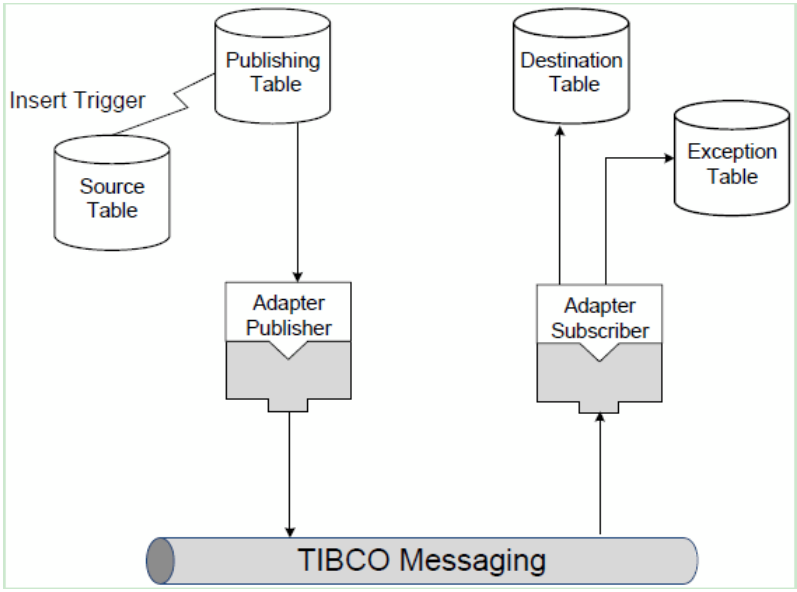
In this tutorial, you can use the source table, publishing table, destination table, and exceptions table to exchange data.

For details about the descriptions of these tables, see Table Reference.

When you insert, update, delete, or upsert data in the source table, the following actions occur:

- If you set `Publish by Value` as the storage mode, the insert, update, delete, or upsert operation fires a trigger and the publisher adapter copies all the changed rows to the publishing table. If you set `Publish by Reference` as the storage mode, the publisher adapter copies only the rows changed in the primary key column and the user-defined key column to the publishing table.
- The publisher adapter polls the publishing table to check if any new row was inserted. If any, the publisher adapter fetches the newly inserted rows by using JDBC, packages them into a message, and then publishes the message.
- The subscriber adapter listens for a message. Upon receipt of the message, the subscriber adapter inserts or deletes a row in the destination table by using JDBC.

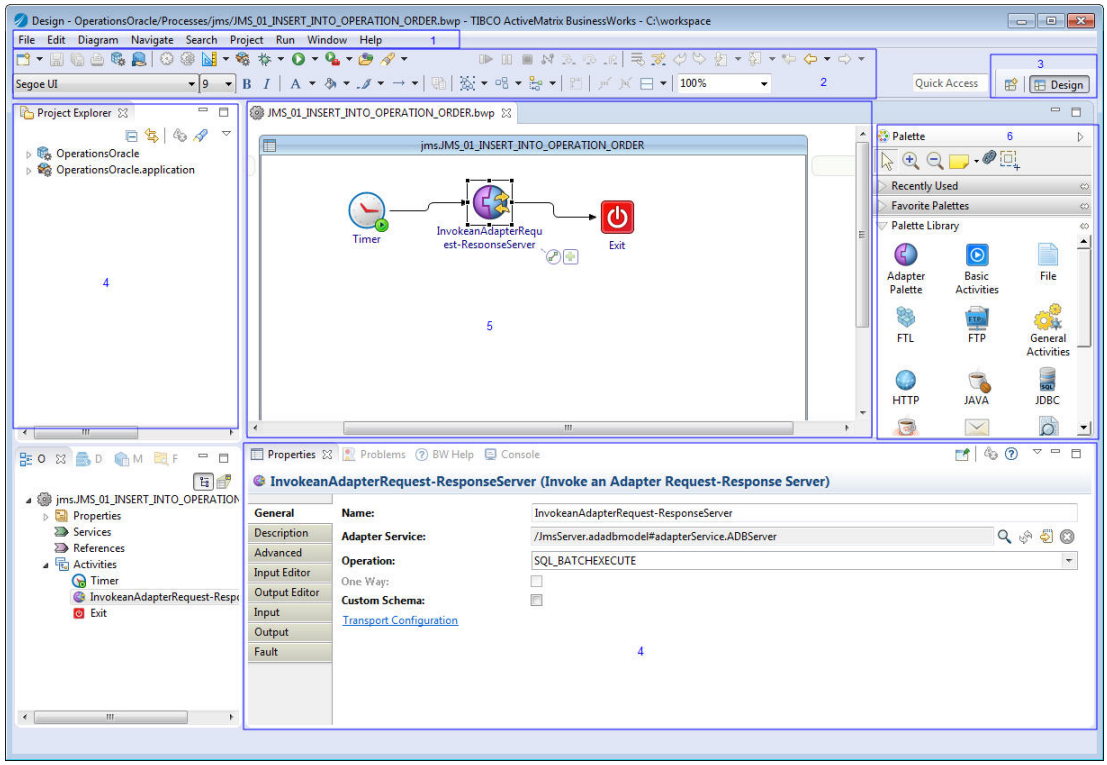
The following figure illustrates the data flow:



TIBCO Business Studio Overview

TIBCO Business Studio is an Eclipse-based integration development environment that is used to design, develop, and test ActiveMatrix BusinessWorks applications.

TIBCO Business Studio provides a workbench in which you can create, manage, and navigate resources in your workspace. A *workspace* is the central location on your machine where all data files are stored.



The workbench consists of:

1. **Menu**: Contains menu items such as File, Edit, Diagram, Navigate, Search, Project, Run, Window, and Help.

2. **Toolbar**: Contains buttons for frequently used commands such as New  , Save  , Enable/ Disable Business Studio Capabilities  , Create a new BusinessWorks Application Module  , Create a new BusinessWorks Shared Module , Debug , Run  , and so on.
3. **Perspective**: Contains an initial set and layout of views that are required to perform a certain task. TIBCO Business Studio launches the Modeling perspective by default. You can change the perspective from the menu **Window** > **Open Perspective** > *Perspective_Name*.
4. **View**: Displays resources. For example, the Project Explorer view displays the ActiveMatrix BusinessWorks applications, modules, and other resources in your workspace, and the Properties view displays the properties for the selected resource. You can open a view from the menu **Window** > **Show View** > *View_Name*.
5. **Editor**: Provides a canvas to configure, edit, or browse a resource. Double-click a resource in a view to open the appropriate editor for the selected resource. For example, double-click an ActiveMatrix BusinessWorks process (`MortgageAppConsumer.bwp`) in the Project Explorer view to open the process in the editor.
6. **Palette**: Contains a set of widgets and a palette library. A *palette* groups activities that perform similar tasks and provide quick access to activities when configuring a process.

Creating Database Tables

Before you create a project in TIBCO Business Studio, you must create tables in a database.

Procedure

1. Open a command window and change the directory to the `Demo1` subdirectory.
   For example: `> cd TIB_ADADB_HOME\demo\bw6\Demo1`
2. Execute the `demo1_database_vendor.sql` script in the subdirectory to create tables for your database. Use your environment-specific user ID, password, and database service.
   For example: `> sqlplus userid/password@dbService @demo1_ora.sql .`
   The script creates the items and displays the status. For example:

```
TIB_ADADB_HOME\demo\bw6\Demo1>sqlplus karlh/karlh@ORCL
@demo1_ora.sql

SQL*Plus: Release 11.2.0.1.0 Production on Mon Mar 18 17:23:39 2013

Copyright (c) 1982, 2010, Oracle. All rights reserved.

Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application
Testing options

Table created.
Table created.
Table created.
Index created.
Index created.
Index created.
Sequence created.
Trigger created.
Table created.
Table created.
Table created.
Table created.
Index created.
Index created.
Sequence created.
Trigger created.
```

```
Table created.
Commit complete.
SQL>
```

Creating a Project

The first task of using the adapter is to create a project. After creating a project, you can create adapter configurations and add services.

A project is named BusinessWorks Application Module in TIBCO Business Studio. You must create a project before creating adapter configurations.

Procedure

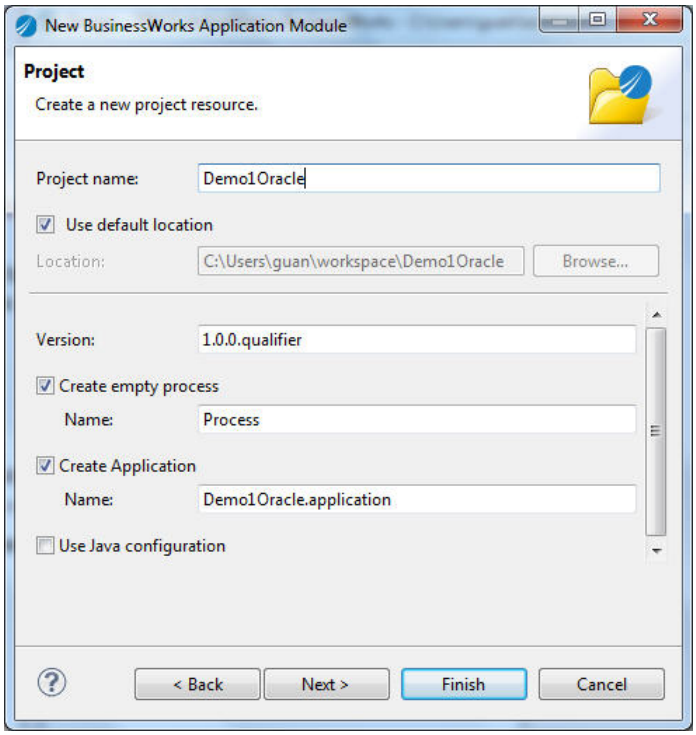
1. Open TIBCO Business Studio in one of the following ways:
   - Microsoft Windows: click **Start** > **All Programs** > **TIBCO** > **TIBCO Business Studio** *version_number* > **Studio for Designers**.
   - Linux: run the TIBCO Business Studio executable located in the *TIBCO_HOME*/studio/*version_number*/eclipse directory.
2. On the Workspace Launcher page, use the default workspace, or click **Browser** to create a new workspace, and then click **OK**.
3. From the main menu, click **File** > **New** > **BusinessWorks Resources** to open the BusinessWorks Resource Wizard window.
4. In the BusinessWorks Resource Wizard window, click the **BusinessWorks Application Module** resource, and then click **Next**.

   TIBCO ActiveMatrix BusinessWorks provides several ways to launch BusinessWorks Resource Wizard. For detailed information, see the TIBCO ActiveMatrix BusinessWorks documentation.
5. In the Project window, enter a project name in the **Project name** field. To use the default location, keep the **Use default location**, **Create empty process**, and **Create Application** check boxes selected.

   Do not use spaces or special characters in a project name.



6. Click **Finish**.

Result

The project with the given name Demo1Oracle is created. Meanwhile, an application with the same name is created for this project.

The project contains the **Processes**, **Service Descriptors**, **Resources**, **Schemas**, and **Module Descriptors** resources, which are created automatically in the project.

Creating a Database Connection

After creating a project, you have to create a database connection to connect an adapter configuration to your database.

Procedure

1. Use one of the following ways to open the Adapter for Database Connection wizard:
   - From the main menu, click **File** > **New** > **Other**.
   - In the Project Explorer view, right-click the project folder and from the pop-up menu, click **New** > **Other**.
2. In the "Select a wizard" window, click **TIBCO Adapters** > **Adapter for Database** > **Adapter for Database Connection**. Click **Next**.
3. To create an Oracle connection, select **Oracle** from the **Vendor** list and specify the values in the **Connection Resource** field, **JDBC URL** field, **User Name** field, **Password** field, and **Default Schema** field (optional).

   For details about connection fields, see New Connection Information.
4. Click **Test Connection** to verify your configuration and click **OK**; then click **Finish**.

Creating Adapter Configurations

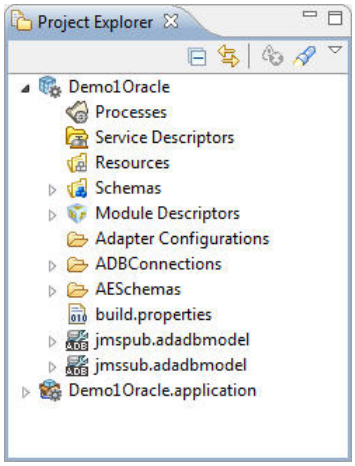
After creating a project, you can create adapter configurations to use the database connection to connect to your database. An adapter configuration contains all information required by the runtime adapter.

Procedure

1. From the main menu, click **File** > **New** > **Other**.
2. In the "Select a wizard" window, click **TIBCO Adapters** > **Adapter for Database** > **Adapter for Database Configuration**. Click **Next**.
   The Adapter for Database Configuration window opens.

   You can use different ways to open the Adapter for Database Configuration window. For details, see Creating an Adapter Configuration.
3. In the Adapter for Database Configuration window, specify a name for the adapter configuration.
4. Click **Finish**.
   The adapter configuration is created with the `.adadbmodel` suffix added to the name.
5. Repeat Step 1 to Step 4 to create another adapter configuration.



Two adapter configurations are created: **jmspub.adadbmodel** and **jmssub.adadbmodel**.

Adding the Database Connection

After creating the adapter configurations, you can add the database connection for the adapter configurations.

Procedure

1. In the Project Explorer view, double-click **jmspub.adadbmodel**.
2. In the Database Configuration panel of the **Configuration** tab, click  next to the **Connection Reference** field.
   The Adapter for Database Connection Configuration window opens.
3. Select the database connection that you created and click **OK**.
4. Click **Save** on the toolbar of TIBCO Business Studio.
5. Repeat Step 1 to Step 4 to add the database connection for the jmssub adapter configuration.

You can click **Link Connection To Schema Browser** to add the existing database connection to the Schema Browser view. Then you can fetch tables or stored procedures in the Schema Browser view if you want, avoiding the need of creating another database connection in the Schema Browser view.

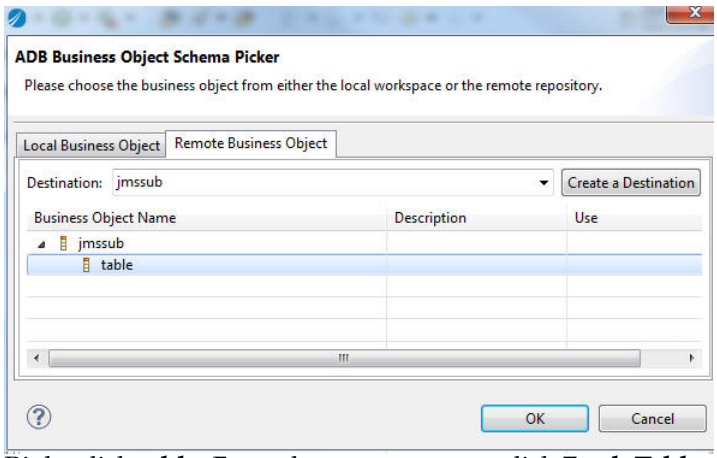
Adding Adapter Services

After configuring adapter configurations, you can add adapter services to communicate with source or target applications.
You can add an adapter service in several ways. This tutorial describes how to add an adapter service in the **Adapter Services** tab. For more information on how to create adapter services of different types, see:

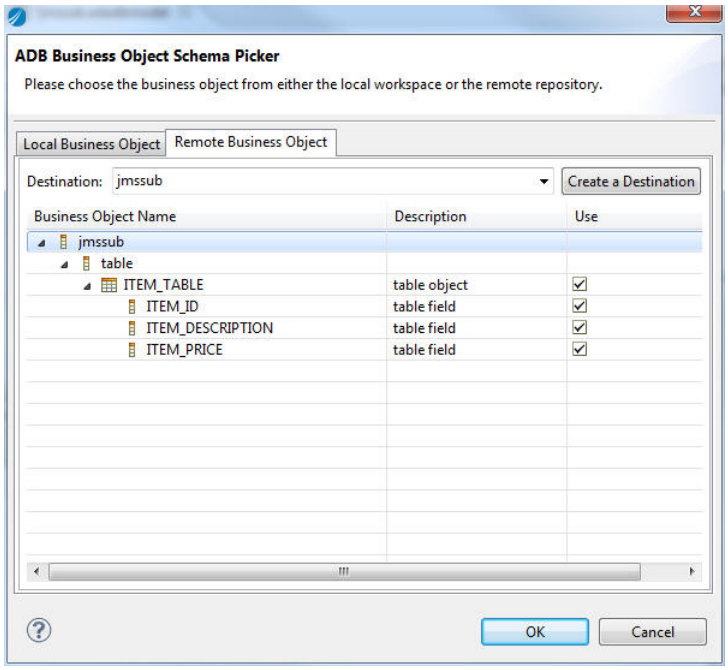
- Creating Publication Service
- Creating Subscription Service
- Creating Request-Response Service

Procedure

1. In the jmspub adapter configuration editor, click the **Adapter Services** tab.
2. In the All Adapter Services panel, click **Add**.
   The Create Service Wizard wizard opens.
3. In the Adapter Service General Configuration window, select **Publication** from the **Service Type** list and click **Next**.
   The Schema Type window opens.
4. In the Schema Type window, select a table for Publication Service:
   a) Click  next to the **Schema** field to open the ADB Business Object Schema Picker window.
   b) Click the **Remote Business Object** tab, and expand the node with the name of the created adapter configuration.

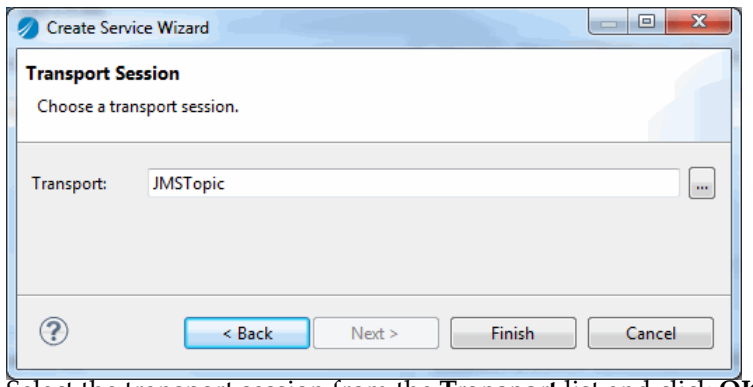


   c) Right-click **table**. From the pop-up menu, click **Fetch Table**.
   d) Enter a table name pattern or click **OK** in the Table name Pattern window to open the Table Download Dialog window.
   e) In the Table Download Dialog window, select the table you want to fetch and click **OK**.



5. Optional: Click **Next** to select a transport session for the service if you do not want to use the default setting:

   a) Click ... next to the **Transport** field to open the Transport Selection window.



   b) Select the transport session from the **Transport** list and click **OK**.
   c) Click **Finish**.

6. Repeat Step 1 to Step 5 to add Subscription Service in the jmssub adapter configuration.
7. Click **Save** on the toolbar of TIBCO Business Studio.

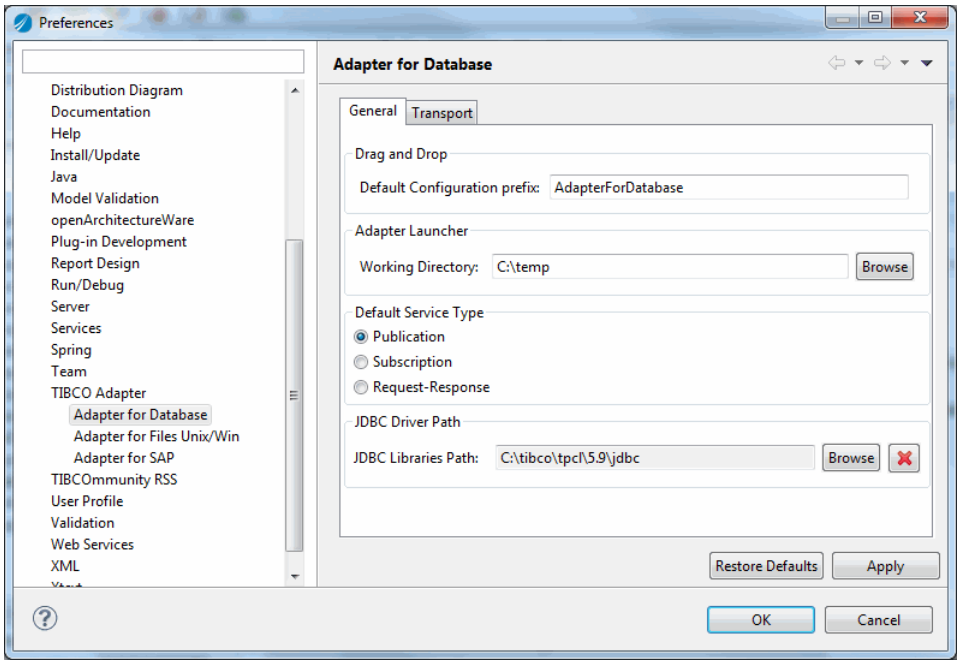
Specifying a Working Directory

Before you start the adapter configurations, you have to specify a working directory to save the temporary files for the runtime adapter.

Alternatively, you can specify a working directory by using the Adapter Launcher tool. For details, see Using Adapter Launcher to Start an Adapter Configuration.

Procedure

1. From the main menu of TIBCO Business Studio, click **Window** > **Preferences**.
2. In the Preferences window, click **TIBCO Adapter** > **Adapter for Database** to open the preferences for the adapter.



3. In the **General** tab, provide the directory where you want to save the temporary files for the runtime adapter in the **Working Directory** field.
4. Click **OK**.

Starting the Adapter Configurations

After configuring adapter services and specifying a working directory, you can start the adapter configurations containing these services so that the adapter exchanges data with the database.

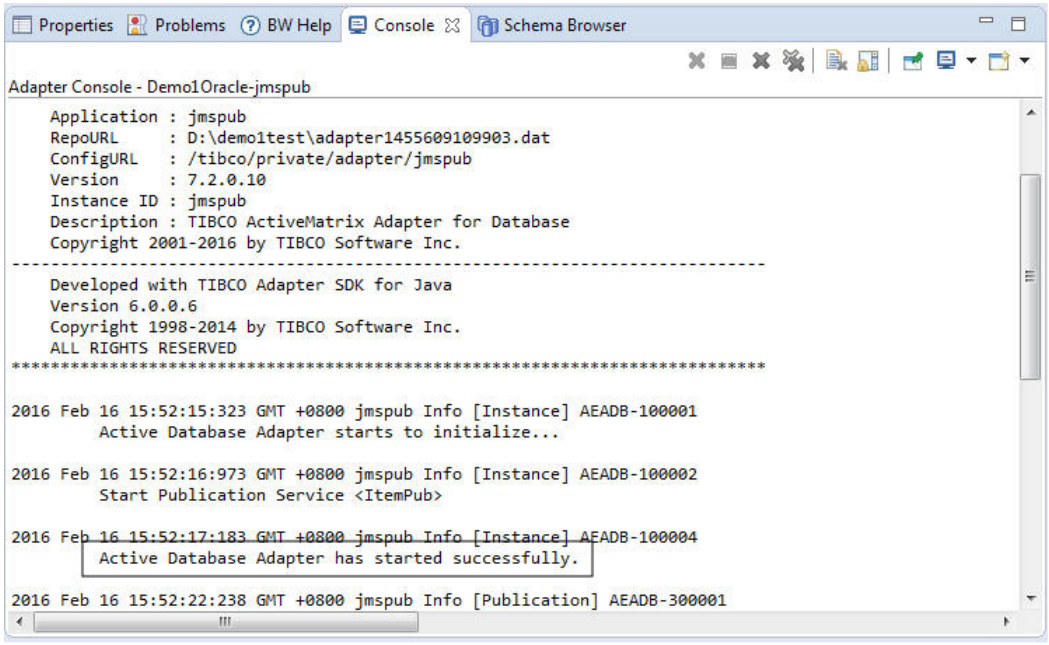
In this tutorial, the adapter services use the JMS transport type. In this case, you have to start the TIBCO EMS server before you start the adapter configurations. If you use the TIBCO Rendezvous transport type, you do not have to start the TIBCO EMS server.

Prerequisites

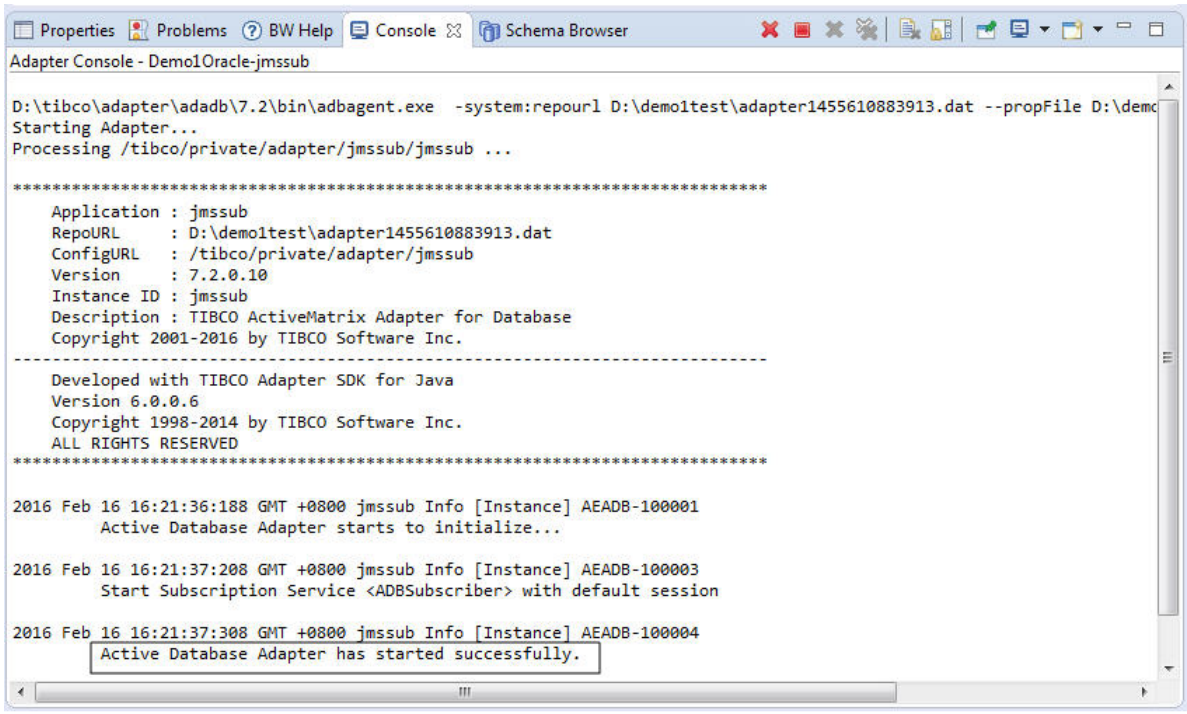
Ensure that Publication Service and Subscription Service share the same destination. For details about destinations, see the description of the **Destination** field in Endpoint Reference.

Procedure

1. Start the TIBCO EMS server:
   a) From the Windows **Start** menu, click **Run** and enter `services.msc`.
   b) In the Services window, start **TIBCO EMS Server**.
2. In the Project Explorer view, right-click the jmspub adapter configuration, and from the pop-up menu, click **Run As** > **Adapter Launcher**.
   A message similar to the following example is displayed in the Console view when the adapter is started successfully.



3. In the Project Explorer view, right-click the jmssub adapter configuration, and from the pop-up menu, click **Run As** > **Adapter Launcher**.
   A message similar to the following is displayed in the Console view when the adapter is started successfully.



Changing the Table Values and Receiving a Notification

After starting the adapter, you can change the values in the source table and test the adapter services.

Procedure

1. Insert three values into the source table (ITEM_TABLE), and then commit the changes.

```
SQL> insert into ITEM_TABLE values(1,'Mickey Mouse',12.12);
SQL> commit;
```

   You can insert additional data rows if you want. If you do so, ensure that the first column is a primary key and must contain a value that is unique within the table.

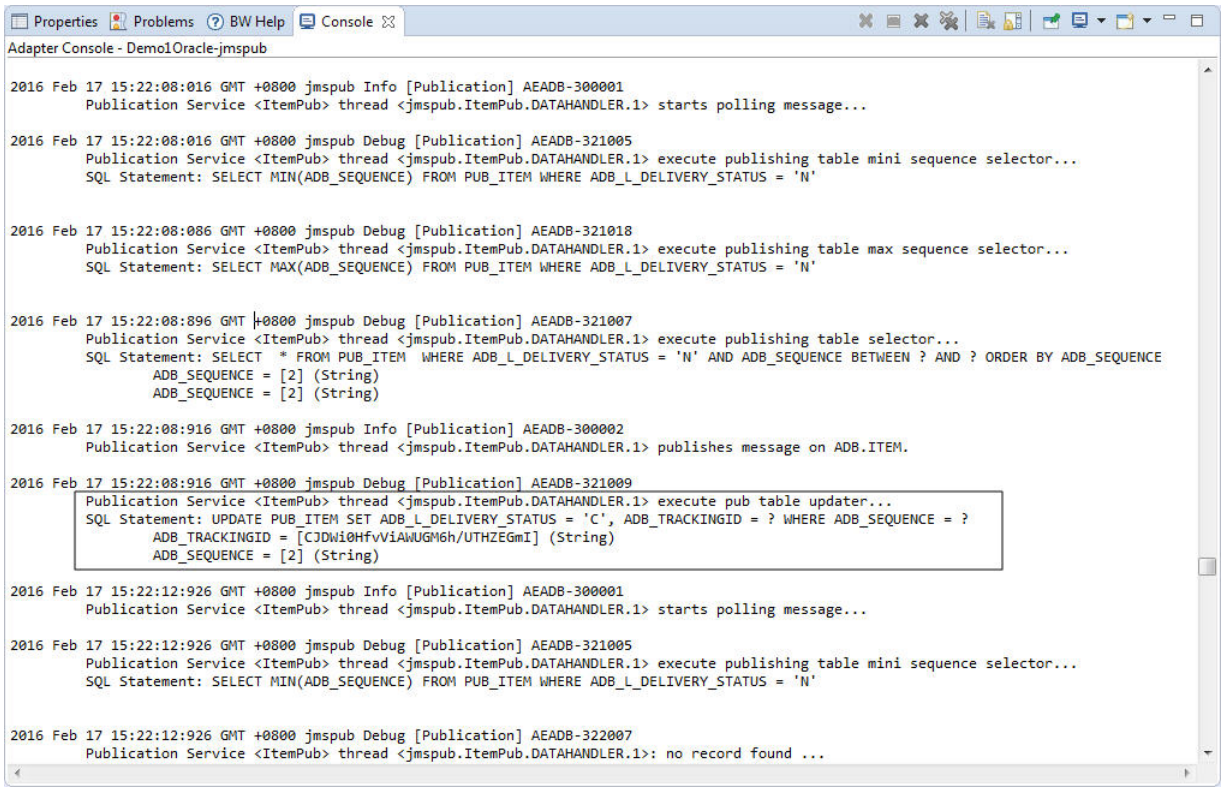
2. To verify whether the rows in the source table are inserted into the destination table, execute the following query SQL statement:

```
SQL> select * from SUB_ITEM;
```

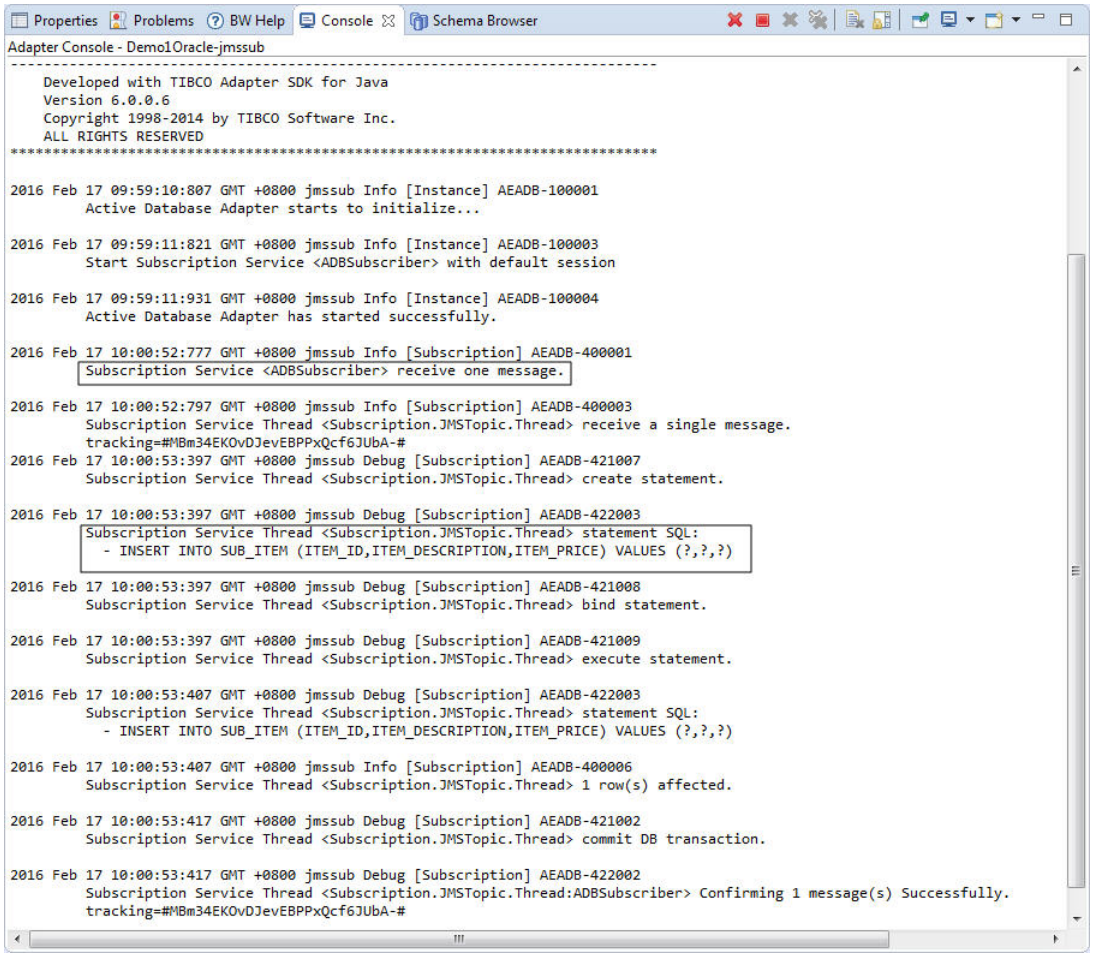
   The following example result confirms that the data is inserted:

```
ITEM_ID
----------
ITEM_DESCRIPTION
--------------------------------------------------
ITEM_PRICE
-----------
1
Mickey Mouse
12.12
```

3. Verify the status messages logged for Publication Service.
   Messages similar to the following example are displayed in the Console view when Publication Service confirms the data.



4. Verify the status messages logged for Subscription Service.
   Messages similar to the following example are displayed in the Console view when Subscription Service receives new data.



5. To stop the adapter, click the stop button on the top left of the Console view.
6. Stop the TIBCO EMS server:
   a) From the Windows Start menu, click **Run** and enter `services.msc`.
   b) In the Services window, stop **TIBCO EMS Server**.

Removing the Tables from a Database

After running the adapter services and obtaining your test results, you can execute the cleanup script to remove all the example tables that were created by the `demo1_databasevendor.sql` script.

Procedure

1. On a command line, change directory to the demo1 directory.
   For example: `> cd TIB_ADADB_HOME\demo\bw6\demo1`
2. Execute the appropriate demo_cleanup.sql script in the subdirectory.
   `> sqlplus userid/password@dbService @demo1_cleanup_databasevendor.sql`

Working with an Adapter Configuration

An adapter configuration in a project contains all information required by the runtime adapter. Each adapter configuration must have its corresponding resources configured.

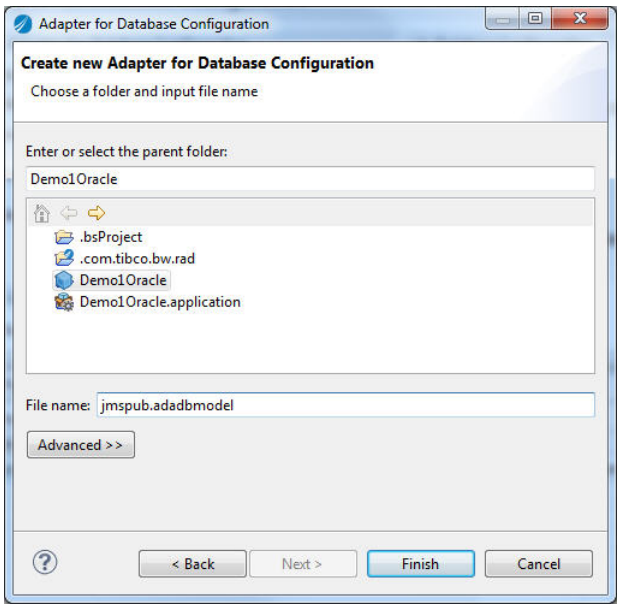
Working with an adapter configuration includes creating an adapter configuration and configuring the adapter configuration.

Creating an Adapter Configuration

Before you configure an adapter configuration, you must create an adapter configuration in an existing project. You can use the Adapter for Database Configuration wizard to do so.

Procedure

1. Use one of the following ways to open the Adapter for Database Configuration wizard:
   - From the main menu, click **File** > **New** > **Other**.
   - In the Project Explorer view, right-click the project folder and from the pop-up menu, click **New** > **Other**.
2. In the "Select a wizard" window, click **TIBCO Adapters** > **Adapter for Database** > **Adapter for Database Configuration**. Click **Next**.
3. In the Adapter for Database Configuration window, specify a name for the adapter configuration.



4. Click **Finish**.
   The adapter configuration is created with the `.adadbmodel` suffix added to the name.

The `Adapter Configurations` folder is created along with the creation of the first adapter configuration. This folder is used to create configurations under it.

Configuring an Adapter Configuration

Each adapter configuration must have its corresponding resources configured. The configurations are accessed whenever an adapter application is started. You can use the resources provided by the tabs in the Adapter Configuration editor to configure an adapter configuration.

The configurations of an adapter configuration include:

- In the **Configuration** tab, you can define the general information, such as an adapter configuration name and description, and database connection. For details, see Configuration Tab.
- In the **Adapter Services** tab, you can add and configure an adapter service. For details, see Adapter Services Tab.
- In the **Transports** tab, you can create a transport session and an endpoint to encapsulate information necessary for data communication. For details, see Transports Tab.
- In the **Logging** tab, you can define logging options. For details, see Logging Tab.
- In the **Monitoring** tab, you can specify monitoring options. For details, see Monitoring Tab.
- In the **Advisories** tab, you can configure TIBCO Rendezvous advisory messages and TIBCO Adapter SDK advisory messages. For details, see Advisories Tab.
- In the **Advanced** tab, you can configure advanced options, such as debug level and reconnection configuration. For details, see Advanced Tab.

Adapter Configuration Reference

The Adapter Configuration editor is a form-based editor split into several tabs. Each tab contains one type of resource and has one or more configuration sections that are presented in panels. You can use these tabs to configure an adapter configuration.

The Project Explorer view shows a virtual outline of an adapter configuration. The outline depicts the virtual containment hierarchy of the adapter configuration. Each node in the hierarchy is an Eclipse resource that is backed by a physical file. In most cases, a resource node in the outline maps to a tab in the Adapter Configuration editor.

You can first set the preferred configuration settings by using the Preferences window. Then configure each resource by using the tabs in the editor. For details about preferences configuration, see Configuring Adapter Preferences. The Adapter Configuration editor includes the following tabs:

- Configuration Tab
- Adapter Services Tab
- Transports Tab
- Logging Tab
- Monitoring Tab
- Advisories Tab
- Advanced Tab

Many of the configuration options support module properties. For details on how to define and use module properties, see Configuring Module Properties.

Configuration Tab

You can use the **Configuration** tab to specify the general information and crucial options for an adapter configuration. The information is grouped in panels, including Adapter for Database Configuration, Database Configuration, and Getting Started.

Adapter for Database Configuration Panel

In the **Adapter for Database Configuration** panel, you can specify the general information, including the name and description, for an adapter configuration.

The following table lists the general information of an adapter configuration.

Name	Description
Adapter Name	Type of the adapter configuration. The default value is `ADBAdapter`.
Instance Id	Name of the adapter configuration. For more information, see Guidelines for Choosing an Instance ID.
Description	(Optional) Description of the adapter configuration.

Name	Description
Message Filter	(Optional) Filter that performs manipulations on incoming and outgoing data before sending it on the network or handing it to the target application. You can write filters by using TIBCO Adapter SDK. See *TIBCO Adapter SDK Programmer's Guide* for information about writing a message filter.

Database Configuration Panel

In the Database Configuration panel, you can set a database connection.

The following table lists the fields and check boxes for the configuration.

Name	Description
Vendor	You can select a database vendor that the adapter connects to. This selection populates the **JDBC Driver** and **JDBC URL** fields in the New Connection Information window with the appropriate data. In the **Vendor** list, vendor names enclosed in parenthesis, such as **(Informix)** and **(INGRES)**, are not supported.  If you select **Teradata** from the **Vendor** list, you have to first execute the `grant select on dbc.UDTInfo to username;` SQL statement to grant the adapter select permission on DBC.UDTInfo.
DB2 AS400 Library	This field is displayed if **DB2 AS400** is selected from the **Vendor** list. A DB2 AS400 library name is required. The default library name is the same as the AlternateID specified in the **JDBC URL** field in the New Connection Information window. Other database libraries are also supported.  Note the following conditions: • You must configure the **DB2 AS400 Library** field. If not, the publishing table cannot be created. • You must also set the trigger option in the **DB2 AS400 Options** tab of Publication Service.
Convert Number Datatype to String	This check box is displayed if **Oracle** is selected from the **Vendor** list. If this option is selected, the adapter uses the String data type to represent the Number data type at run time. By default, this check box is selected.
Write to Database on Save	You can use this check box if you want to write configuration information to the database when you save a project. By default, this check box is selected.

Name	Description
Connection Reference	Specifies a database connection resource to be used by the adapter configuration. To specify a database connection reference, click the browse button next to the **Connection Reference** field and select a reference from the displayed connection configuration list. If the list does not include the database connection reference you want to use, click **New** to open the New Connection Information window and add one. For details about how to add a connection, see Adding a Database Connection. For details about the fields in the New Connection Information window, see New Connection Information.
Link Connection To Schema Browser	Click this link, and a new destination is created in the Schema Browser view. The name and database connection information of the new destination is the same as the specified connection reference. The generated connection name in the Schema Browser view is the same as the configuration file name.

Getting Started Panel

The Getting Started panel lists the **Configure Adapter Services** link. Click this link, and you are directed to the **Adapter Services** tab. For details, see Adapter Service Reference.

Guidelines for Choosing an Instance ID

When you specify the file name for an adapter configuration, you must follow certain rules. Otherwise, the instance ID of the adapter configuration might not be valid.

The following guidelines are provided for you to choose an Instance ID:

- An instance ID must use alphanumeric characters. An instance ID can use underscore (_) characters, but cannot contain any spaces. The entire instance name must be less than 80 characters.
- An instance ID cannot use module properties.
- An instance ID must be unique in a project.

New Connection Information

When you click the browse button next to the **Connection Reference** field, the Adapter for Database Connection Configuration window opens. You can click **New** to open the New Connection Information window. To add a database connection resource, you have to specify values for the fields in the New Connection Information window.

The following table lists and describes the fields for creating a database connection resource.

Name	Description
Connection Resource	Name of the database connection resource.
JDBC Driver	Name and URL of the JDBC driver used for the design-time configuration. For the list of supported JDBC drivers and their URLs, see

Name	Description
JDBC URL	Using Connection Settings Templates. For detailed parameter descriptions, see your JDBC driver documentation.
User Name	User name that the adapter uses to connect to the database.
Password	Password that the adapter uses to connect to the database.
Default Schema	Schema name used for the current user. Usually the schema name is the same as the database name. But for some specific databases, the schema name and the database name are different. For example, when using a Microsoft SQL Server database, the default schema name is DBO. When using a PostgreSQL database, the default schema name is Public. When using a MySQL database, the **Default Schema** field must be empty.  When you use a MySQL database, if you enter a name for the default schema, a publishing table cannot be created for Publication Service. For more information about the default schema, see Default Schema.
Test Connection	Click this button to verify whether the connection fields are specified correctly. A message is displayed to indicate whether the connection succeeds or not.

Using Connection Settings Templates

TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) provides connection settings templates with JDBC driver information for each supported database vendor. The templates populate the **JDBC Driver** and **JDBC URL** fields with the default settings as shown in the following table, and you can replace the variables with values appropriate for your configuration.

Database	JDBC Driver	JDBC URL
Oracle	tibcosoftwareinc.jdbc.oracle.OracleDriver	jdbc:tibcosoftwareinc:oracle://*server_name*:1521;databaseName=*database_name*  The default port number is `1521`. When you configure the adapter to connect to an Oracle database, if you want to insert data with the TIMESTAMP with Time Zone data type, configuration of the **JDBC URL** field affects the data type of the data to be inserted: • If you set the `FetchTSWTZasTimestamp=true` property in the **JDBC URL** field, the inserted data with a certain format is parsed into the JDBC TIMESTAMP data type. You can use the `adb.timestampPattern` property to set a format for the data to be inserted. • If you set the `FetchTSWTZasTimestamp=false` property or do not add this property in the **JDBC URL** field, the inserted data is processed in the JDBC VARCHAR data type. Therefore, you must set the data in a format complying with the Oracle database rules before inserting. For supported data formats, see Oracle websites.
Microsoft SQL Server	tibcosoftwareinc.jdbc.sqlserver.SQLServerDriver	jdbc:tibcosoftwareinc:sqlserver://*server_name*:1433;databaseName=*database_name*  The default port number is 1433.
Sybase	tibcosoftwareinc.jdbc.sybase.SybaseDriver	jdbc:tibcosoftwareinc:sybase://*server_name*:5000;DatabaseName=*database_name*
DB2 OS390	tibcosoftwareinc.jdbc.db2.DB2Driver	jdbc:tibcosoftwareinc:db2://*server_name*:*port*; locationName=*location_name*;packageName=*packageName*
DB2 AS400	tibcosoftwareinc.jdbc.db2.DB2Driver	jdbc:tibcosoftwareinc:db2://*server_name*:*port*;locationName=*location_name*;AlternateID=*library*  The publishing table cannot be created if the `AlternateID=library` property is not in the JDBC URL.
DB2 UDB	tibcosoftwareinc.jdbc.db2.DB2Driver	jdbc:tibcosoftwareinc:db2://*server_name*:50000;databaseName=*database_name*;packageName=DEF00

Database	JDBC Driver	JDBC URL
MySQL	tibcosoftwareinc.jdbc.mysql.MySQLDriver	jdbc:tibcosoftwareinc:mysql://*server_name*:3306/DatabaseName=*database_name*  Regardless of MySQL or MySQL Community Server, if you create different services of Publication Service and configure these services to load the same source table, a `java.sql.SQLException` error is thrown when you save the project. This is a limitation of the MySQL database.
Teradata	com.teradata.jdbc.TeraDriver	jdbc:teradata://*server_name*/database=*database_name*
PostgreSQL	org.postgresql.Driver	jdbc:postgresql://*server_name*:*port*/*database_name*
Microsoft Azure	tibcosoftwareinc.jdbc.sqlserver.SQLServerDriver	jdbc:tibcosoftwareinc:sqlserver://*server_name*:1433;databaseName=*database_name*
Redshift	org.postgresql.Driver	jdbc:postgresql://*server_name*:*port*/*database_name*
Amazon RDS (Oracle)	tibcosoftwareinc.jdbc.oracle.OracleDriver	jdbc:tibcosoftwareinc:oracle://*amazon_server_name*.rds.amazonaws.com:1521;databaseName=*database_name*
Amazon RDS (Microsoft SQL Server)	tibcosoftwareinc.jdbc.sqlserver.SQLServerDriver	jdbc:tibcosoftwareinc:sqlserver://*amazon_server_name*.rds.amazonaws.com:1433;databaseName=*database_name*
Amazon RDS (MYSQL)	com.mysql.jdbc.Driver	jdbc:mysql://*amazon_server_name*.rds.amazonaws.com:3306/*database_name*

Default Schema

The **Default Schema** field in the New Connection Information window indicates the schema name used for the current user.

You must set a value in the **Default Schema** field for some databases if the database user name and the default schema specified in the database are different:

- If you set the value of the **Default Schema** field to the same as the default schema specified in an Oracle database or leave the **Default Schema** field empty for an Oracle database, the table you select in the adapter service **Schema** tab is loaded from the default schema and the publishing table is also generated in the default schema.
- If you set the value of the **Default Schema** field to the same as the default schema specified in an Oracle database, and select a table from a different schema (other schema) than the default schema by clicking **Add From Other Schema** in the Table Download Dialog window, the table you select in the **Schema** tab is loaded from the other schema but the publishing table is generated in the default schema. If you want the publishing table to be generated in the other schema, you must prefix the publishing table name with the other schema name in the **Publication Options** tab.
- If you set the value of the **Default Schema** field to a different schema (other schema) name than the default schema specified in an Oracle database, and select a table from the other schema by clicking

Add From Other Schema in the Table Download Dialog window, the table you select in the **Schema** tab is loaded from the other schema and the publishing table is also generated in the other schema.

- If you set the value of the **Default Schema** field to `DBO` for a Microsoft SQL Server database or `Public` for a PostgreSQL database, the DBO schema or Public schema is used as the default schema. The table you select in the adapter service **Schema** tab is loaded from the default schema, and the publishing table is also generated in the default schema.
- If you set the default schema to a non-DBO schema (other schema) for a Microsoft SQL Server database or a non-Public schema (other schema) for a PostgreSQL database, and select a table from the other schema by clicking **Add From Other Schema** in the Table Download Dialog window, the table you select in the **Schema** tab is loaded from the other schema and the publishing table is also generated in the other schema.

Adapter Services Tab

You can use the **Adapter Services** tab to add or remove an adapter service. You can also configure an adapter service by using the tabs in the Adapter Services panel.

For details about these tabs, see Adapter Service Reference.

Transports Tab

The adapter user interface automatically creates a corresponding session and an endpoint to encapsulate transport information necessary for data communication.

In the **Transports** tab, you can use the All Adapter Transports panel to add or remove a session and its associated endpoint. You can also use the configuration panel to change the configurations of a session or endpoint.

Both sessions and endpoints are concepts in TIBCO Adapter SDK, which is the fundamental class library used in the adapter implementation.

Endpoints send or receive data. They represent services that the adapter provides. Each endpoint is associated with a session that is used to communicate with the source or target application. A session encapsulates the transport information of an adapter service.

For more information, see *TIBCO Adapter SDK Programmer's Guide*.

Supported Endpoints

Publisher, Subscriber, Client, and Server are the endpoints that are available in an adapter. An adapter service encapsulates both the corresponding endpoint and session. The services that TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) supports include:

- Publication Service: a publisher endpoint and associated session
- Subscription Service: a subscriber endpoint and associated session
- Request-Response Service: a server endpoint and associated session

A client endpoint is supported for Request-Response Invocation Service. However, this type of service is not supported in TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio).

Supported Sessions

Both TIBCO Rendezvous and JMS transport sessions are supported.

Sessions for the TIBCO Rendezvous Transport Type

If you select the TIBCO Rendezvous transport type, you have Quality of Service sessions. They are Reliable (RV), Certified (RVCM), and Distributed Queue (RVCMQ).

Reliable (RV)

Reliable Message Delivery ensures that each multicast or broadcast message is received as long as the physical network and packet receivers are working. It also ensures that the loss of a message is detected.

Reliable Message Delivery can compensate for brief network failures, because it can retransmit a message on request if the first attempt fails. This option is appropriate when message delivery is expected but some loss can be tolerated.

Certified (RVCM)

Certified Message Delivery ensures that every certified message reaches its intended receiver in an order. A message can be sent across network boundaries. If a network fails, delivery attempts continue until delivery succeeds or until the time limit of the message expires. This is often called guaranteed delivery.

Distributed Queue (RVCMQ)

This type of session is supported only for Subscription Service and Request-Response Service. Distributed Queue delivers a message to one of many service listeners (workers). Distributed Queue contains features of both Certified Messaging and Fault Tolerance.

For more information about the TIBCO Rendezvous transport type and Quality of Service sessions, see *TIBCO Rendezvous Concepts*.

Sessions for the JMS Transport Type

If you select the JMS transport type, you have connection factory sessions. They are Topic and Queue.

Topic

A message published to a topic is broadcast to one or more subscribers. All messages published to the topic are received by all services that have subscribed to the topic. This messaging model is known as publish-subscribe.

Queue

A message sent to a queue is consumed by one and only one receiver. Each message has only one receiver, although multiple receivers might be connected to the queue. The first receiver to access the queue obtains the message. The other receivers do not. This messaging model is known as point-to-point.

For more information about the JMS transport type and connection factories, see *TIBCO Enterprise Message Service User's Guide*.

Session Reference

After you add a session, the configuration options for the session are displayed in the Configuration panel. The transport session type you select determines the specific options you have to configure.

Rendezvous Sessions

You can use the Rendezvous Configuration panel to specify the general information. For options specific to this transport type, you can use the Rendezvous Options panel.

The following table lists the options in the Rendezvous Configuration panel:

Name	Description
Name	Name of the TIBCO Rendezvous transport session.
Description	(Optional) Description of the transport session.
Daemon	Module property that specifies the TIBCO Rendezvous daemon for this session.
Network	Module property that specifies the network for this session. By default, the property is an empty string, which is interpreted as the primary network. Using this attribute only makes sense on computers with more than one network interface.
Service	Module property that specifies the service for this transport. By default, the property is defined to be the default TIBCO Rendezvous service (7500).
Connection Type	The type of the connection: • Reliable • Certified • Distributed Queue

If the connection type is Reliable, no additional configuration is required.

If the connection type is Certified, you have to specify additional options in the Rendezvous Options panel. The following table lists the options specific to the Certified connection type:

Name	Description
CM Name	Used to identify the delivery tracking session. It must be unique across the entire network.
Ledger File	Module property that points to a ledger file. If the value of the variable value is a valid file name, the transport uses a file-based ledger.
Sync Ledger File	This check box controls the behavior of TIBCO Rendezvous when the ledger file is updated: • Selected: operations that update the ledger file do not return until the changes are written to the storage medium. By default, this check box is selected. • Cleared: the operating system writes changes to the storage medium asynchronously.
Relay Agent	Relay agent for this transport.

Name	Description
Require Old Message	This check box indicates whether a persistent correspondent requires delivery of messages sent to a previous transport with the same name for which delivery was not confirmed. Its value affects the behavior of other delivery-tracking senders: • Selected: if the name attribute is non-NULL, this transport requires certified senders to retain unacknowledged messages sent to this persistent correspondent. By default, this check box is selected. • Cleared: messages are not retained.
Message Timeout (ms)	Maximum time (in milliseconds) that this call can block while waiting for a reply. The default value is `0`.

If the connection type is Distributed Queue, you have to specify additional options in the Rendezvous Options panel. The following table lists the options specific to the Distributed Queue connection type:

Name	Description
CMQ name	Sequence of the module property that specifies the name of the queue.
Worker Weight	Relative worker weights assist the scheduler in assigning tasks. When the scheduler receives a task, it assigns the task to the available listener with the greatest listener weight. The default value is `1`.
Worker Tasks	Worker tasks for this session. The default value is `1`.
Worker Complete Time (ms)	If the complete time is non-zero, the scheduler waits for a worker member to complete an assigned task. If the complete time elapses before the scheduler receives completion from the worker member, the scheduler reassigns the task to another worker member. The default value is `0`.
Scheduler Weight	Represents the ability of this session to fulfill the role of the scheduler, relative to other members of the same queue. The queue members use relative scheduler weight values to elect one member as the scheduler. Members with higher scheduler weight take precedence. The value is in the range `1 - 65545`. The default value is `1`.
Scheduler Heartbeat (ms)	The active scheduler sends heartbeat messages at this interval (in milliseconds). The same scheduler heartbeat interval must be specified for all members in the queue. The default value is `1000` ms.
Scheduler Activation (ms)	When the heartbeat signal from the scheduler has been silent for this interval (in milliseconds), the queue member with the greatest scheduler weight takes its place as the new scheduler. The same scheduler activation interval must be specified for all members in the queue. The default value is `3000` ms.

JMS Sessions

You can use the JMS Configuration panel to specify the general configuration information for a JMS session. For options specific to this transport type, you can use the More Options panel.

The following table lists the options in the JMS Configuration panel:

Name	Description
Name	Name of the JMS transport.
Description	(Optional) Description of the transport.
Connection Type	Lists the available connection types: • **Direct**: this is the default setting. The connection is direct. • **JNDI**: a JNDI server is used.
Provider URL	(Direct connection type only) URL of the server.
Connection Factory	(Direct connection type only) Two connection factory types are available: • **TopicConnectionFactory**: a message published to a topic is broadcast to one or more subscribers. All messages published to the topic are received by all services that have subscribed to the topic. This messaging model is known as publish-subscribe. • **QueueConnectionFactory**: a message sent to a queue is consumed by one and only one receiver. Each message has only one receiver, though multiple receivers might connect to the queue. The first receiver to access the queue obtains the message. The other receivers do not. This messaging model is known as point-to-point.
JNDI Reference	(JNDI connection type only) JNDI server information.

In addition to the options in the JMS Configuration panel, you have to specify the options in the More Options panel. The following table lists the options in the More Options panel:

Name	Description
Client ID	ID of the client.
User Identity	Detailed information of the user.

Endpoint Reference

After you add an endpoint, the configuration options for the endpoint are displayed in the configuration panel. The session type of the endpoint you select determines the endpoint specific options you have to configure.

Publisher Endpoint Reference

A publisher endpoint sends data to TIBCO Rendezvous or TIBCO Enterprise Message Service. Only Reliable sessions and Certified sessions can be associated with publisher endpoints. The session for a publisher endpoint determines the endpoint options you have to specify.

The following table lists the options for a publisher endpoint associated with a TIBCO Rendezvous session:

Name	Description
Name	Name of the publisher endpoint.
Description	(Optional) Description of the publisher endpoint.
Endpoint Type	Type of the publisher endpoint: • For a Reliable publisher endpoint, the default value is `Rv Publisher`. • For a Certified publisher endpoint, the default value is `RvCm Publisher`.
Wire Format	Format in which messages are sent. The following values are available: • ActiveEnterprise Message (Default) • XML Message For details, see Guideline for Configuring the Wire Format.
Subject	Subject with which the publisher endpoint sends messages.
Reply Subject	Reply subject for the publisher endpoint.
Message Timeout (ms)	(Publisher endpoints for TIBCO Rendezvous Certified Sessions only) Time after which a message is discarded from the ledger file. The default value is `0` ms, indicating that the timeout is infinite.
Pre-registered Listeners	(Publisher endpoints for TIBCO Rendezvous Certified Sessions only) Comma-separated list of listeners preregistered for the publisher endpoint. Refer to each listener using the `CmName` of the session.

The following table lists the options for a publisher endpoint associated with a JMS session:

Name	Description
Name	Name of the publisher endpoint.
Description	(Optional) Description of the publisher endpoint.
Endpoint Type	Type of the publisher endpoint. The only available value is `Jms Publisher`.
Delivery Mode	Mode in which messages are delivered. The following values are available: • Non-Persistent • Persistent (Default) For details, see Guideline for Configuring the Delivery Mode (JMS Only).

Name	Description
Destination	Destination with which the publisher endpoint sends messages.
Reply destination	Reply destination for the publisher endpoint.
Message Priority	Priority with which messages are sent. The value is in the range `0 - 9`. The default value is `4`.
isCompressed	Enables JMS compression. You can select this check box if you want to compress the body of a message before sending the message to the EMS server. Setting compression ensures that messages take less memory space in storage. For more information about JMS compression, see Compressing JMS Messages.
Message Timeout (ms)	Time after which a message is discarded from the ledger file. The default value is `0` ms, indicating that the timeout is infinite.

Subscriber Endpoint Reference

A subscriber endpoint specifies the data consumers in target applications.

The following table lists the options for a subscriber endpoint associated with a TIBCO Rendezvous session:

Name	Description
Name	Name of the subscriber endpoint.
Description	(Optional) Description of the subscriber endpoint.
Endpoint Type	Type of the subscriber endpoint. The following values are available: • For a Reliable subscriber endpoint, the default value is `Rv Subscriber`. • For a Certified Subscriber endpoint, the default value is `RvCm Subscriber`. • For a Distributed Queue Subscriber endpoint, the default value is `RvCmq Subscriber`.
Startup State	State when the subscriber endpoint starts. The following values are available: • None • Active (Default) • Inactive

Name	Description
Wire Format	Format in which messages are sent. The following values are available: • ActiveEnterprise Message • XML Message For details, see Guideline for Configuring the Wire Format.
Subject	Subject with which the subscriber endpoint receives messages.
Listen Timeout(ms)	If no message is received after this amount of time, the adapter performs any actions specified in the program for that case. The default value is `0` ms.

The following table lists the options for a subscriber endpoint associated with a JMS session:

Name	Description
Name	Name of the subscriber endpoint.
Description	(Optional) Description of the subscriber endpoint.
Endpoint Type	Type of the subscriber endpoint. The only available value is `Jms Subscriber`.
Auto Confirm	You can select this check box to enable TIBCO Adapter SDK to automatically confirm events for the subscriber endpoint. By default, this check box is selected.
Destination	Destination with which the subscriber endpoint receives messages.
Message Selector	A message selector is a string that lets a client program specify a set of messages, based on the values of message headers and properties. A selector matches a message if, after substituting header and property values from the message into the selector string, the string evaluates to true. Consumers can request that the server deliver only those messages that match a selector.
Delivery Mode	(Subscriber endpoints for JMS Topic Sessions only) Mode in which messages are delivered. The following values are available: • Durable (Default) • Non-Durable For details, see Guideline for Configuring the Delivery Mode (JMS Only).
Durable Name	(Subscriber endpoints for JMS Topic Sessions only) Name of the durable subscriber endpoint.

Server Endpoint Reference

Server endpoints are used by an adapter service that communicates with a remote or local client (request-response server).

The following table lists the options for a server endpoint associated with a TIBCO Rendezvous session:

Name	Description
Name	Name of the server endpoint.
Description	(Optional) Description of the server endpoint.
Endpoint Type	Type of the server endpoint. The following values are available: • For a Reliable subscriber, the default value is `Rv RPC Server`. • For a Certified subscriber, the default value is `RvCm RPC Server`. • For a Distributed Queue subscriber, the default value is `RvCmq RPC Server`.
Startup State	State when the endpoint starts. The following values are available: • None • Active (Default) • Inactive
Subject	Subject with which the server endpoint communicates with clients. The default value is `Subscriber`.

The following table lists the options for a server endpoint associated with a JMS session:

Name	Description
Name	Name of the server endpoint.
Description	(Optional) Description of the server endpoint.
Endpoint Type	Type of the server endpoint. The only available value is `Jms RPC Server`.
Destination	Destination with which the server endpoint receives messages.
Message Selector	A message selector is a string that lets a client program specify a set of messages, based on the values of message headers and properties. A selector matches a message if, after substituting header and property values from the message into the selector string, the string evaluates to `true`. Consumers can request that the server deliver only those messages that match a selector.
Delivery Mode	(Server endpoints for JMS Topic Sessions only) Mode in which messages are delivered. The following values are available: • Durable (Default) • Non-Durable For details, see Guideline for Configuring the Delivery Mode (JMS Only).

Name	Description
Durable Name	(Server endpoints for JMS Topic Sessions only) Name of the durable server endpoint. The default value is the server name.

Logging Tab

The adapter uses log sinks to generate logs. The adapter defines traces with different roles and sends them to log sinks with corresponding roles.

You can use standard I/O to configure logging. You can also fine-tune when and where different types of information are sent by defining sinks and mapping each sink to one or more roles.

You can use the **Logging** tab to configure the logging options. The information is grouped in panels:

- If you want to use the console window for logging, you have to select the **Log To Standard I/O** check box in the Log Level panel in this tab. You can send the information to multiple locations and choose to log one or more message types. For more information, see Log Level Panel.
- If you want to use custom roles, you can use the All Log Sinks panel to add or remove log sinks and their roles. Then you can configure the logging options for selected sinks in the Configuration panel. See Supported Log Sinks and Roles and Log Sink Configuration Reference.

For trace messages that the adapter can log to a log sink, see Trace Messages.

Log Level Panel

You can use the Log Level panel to configure logging through standard I/O.

The following table lists the options in the Log Level panel.

Name	Description
Log to Standard I/O	You can select this check box to enable the sending of logging information to the console window when the adapter is started. By default, this check box is selected. Logging information is displayed when this check box is selected.
Log Info Messages	You can select this check box to enable the sending of all messages of the INFO type to the specified location(s). By default, this check box is selected. For more information, see "Supported Log Roles" in Supported Log Sinks and Roles.
Log Debug Messages	You can use this check box to enable the sending of all messages of the DEBUG type to the specified location(s). By default, this check box is not selected. For more information, see "Supported Log Roles" in Supported Log Sinks and Roles.
Log Warning Messages	You can use this check box to enable the sending of all messages of the WARNING type to the specified location(s). By default, this check box is selected. For more information, see "Supported Log Roles" in Supported Log Sinks and Roles.

Name	Description
Log Error Messages	You can use this check box to enable the sending of all messages of the ERROR type to the specified location(s). By default, this check box is selected. For more information, see "Supported Log Roles" in Supported Log Sinks and Roles.

Supported Log Sinks and Roles

You can use the All Log Sinks panel to add or remove a log sink and its roles.

The log sinks and roles that you can add or remove are listed as follows:

Supported Log Sinks

The adapter supports the following log sinks at run time:

- **fileSink**: a file sink sends messages to a file.
- **hawkSink**: a Hawk sink sends messages to TIBCO Hawk.
- **networkSink**: a network sink sends messages over the network.
- **stdioSink**: a standard I/O sink sends messages to standard input and output.

Supported Log Roles

You can add to a log sink one or more log roles that decide the log levels of the sink.

The adapter supports the following log roles:

- Debug
- Error
- Information
- Warning

Logging messages are generated depending on the roles you add to the log sink.

The use of levels affects the performance of the adapter. For best results select the required levels only. By default, the Info, Warning, and Error levels are selected; the Debug level is reserved. Do not select the Debug level unless requested by the TIBCO Product Support Group. This option writes a great deal of debugging information to the log file and significantly lowers the speed of the adapter.

Log Sink Configuration Reference

When you select a log sink, the configuration options for the sink are displayed. The log sink you select determines the options you have to configure.

File Sink Configuration Reference

A file sink sends messages to a file. If you select a file sink, the log sink specific options are displayed as follows:

Name	Description
Name	Name of the sink.
Description	(Optional) Description of the sink.

Name	Description
File Name	File to which logging information is written.
File Limit (bytes)	Maximum size of the file, in bytes. The default value is `30000` bytes. The maximum value is `2147483647` bytes.
File Count	Number of rollover files. The default value is `3`.
Append Mode	You can use this check box to control whether to add traces to the existing file at startup: • Selected: Traces are added to the existing file at startup. By default, this check box is selected. • Cleared: The existing file is overwritten at startup if a duplicate file name exists.

Hawk Sink Configuration Reference

A Hawk sink uses the HAWK session, which is created and used by the adapter for monitoring purposes, to send logging information to TIBCO Hawk. If you select a Hawk sink, the log sink specific options are displayed as follows:

Name	Description
Name	Name of the sink.
Description	(Optional) Description of the sink.
MicroAgent Name	Name of the microagent for traces from this Hawk sink.

Network Sink Configuration Reference

A Network sink sends logging information on both TIBCO Rendezvous and TIBCO JMS. If you select a network sink, the log sink specific options are displayed as follows:

Name	Description
Name	Name of the sink.
Description	(Optional) Description of the sink.
Subject	Subject of TIBCO Rendezvous messages to be sent.
Session Reference	Click **Browse** and select one of the sessions you have defined.

Standard I/O Sink Configuration Reference

A standard I/O sink sends messages to standard input and output. If you select a standard I/O sink, the log sink specific options are displayed as follows:

Name	Description
Name	Name of the sink.
Description	(Optional) Description of the sink.
Output Stream	Output information that is logged. The following values are available: • **stdout**: general output information. • **stderr**: error information.

Monitoring Tab

TIBCO Hawk monitors the runtime adapter. You can use the **Monitoring** tab to configure the monitoring options.

You can use microagents to supplement the monitoring information provided by the standard logging capability. Examples of supplemental information that you can obtain with microagents include the repository URL and the command line arguments used to start the adapter.

The following table lists the monitoring options:

Name	Description
Enable Standard MicroAgent	(Optional) You can select this check box to turn on the standard TIBCO Hawk microagent. By default, this check box is selected.
Standard MicroAgent Name	Name for the standard microagent that is registered with the TIBCO Hawk system. In most cases, keep the default value. You do not have to specify the *InstanceId* variable, because it is automatically set at run time by the runtime adapter.
Standard MicroAgent Timeout (ms)	Specifies the amount of time a Hawk agent has to wait for Hawk Microagent method invocations to be completed before timing them out. The default value is `10000` ms. Normally you do not have to change this value. However, on machines under extreme stress where method invocations are timing out, you can use this option to increase the timeout value.
Enable Class MicroAgent	(Optional) You can select this check box to turn on the instance-specific or class-specific standard TIBCO Hawk microagent.
Class MicroAgent Name	(Optional) Name for the class microagent that is registered with the TIBCO Hawk system. In most cases, keep the default value. You do not have to specify the *InstanceId* variable, because it is automatically set at run time by the runtime adapter.

Name	Description
Class MicroAgent Timeout (ms)	(Optional) Timeout value for the class microagent in milliseconds. The default value is `10000` ms. Normally you do not have to change this value. However, on machines under extreme stress where method invocations are timing out, you can use this option to increase the timeout value.
Default Microagent Session	(Optional) TIBCO Rendezvous session to be used by the TIBCO Hawk microagents by default. This field is disabled and you cannot change it. The session name and the corresponding session are automatically generated. However, you can modify the session parameters if required by using the **Transports** tab and modifying the session parameters.

Advisories Tab

The adapter contains TIBCO Rendezvous advisory messages and TIBCO Adapter SDK advisory messages. You can configure the two types of advisory messages in the **Advisories** tab.

The **Advisories** tab consists of the All Advisories panel and the Configuration panel. The All Advisories panel lists all Rendezvous and SDK advisories, and you can add or remove advisories.

The Configuration panel is used to configure the selected advisory in the All Advisories panel. The Configuration panel contains the following fields:

Name	Description
Name	Name of the advisory message.
Description	Description of the advisory message.

Subject	TIBCO Rendezvous advisory messages have the following structure:
	`_RV.<class>.<source>.<category>.<role>.<condition>.<name>`
	For details, see *TIBCO Rendezvous Concepts*.
	TIBCO Adapter SDK advisory messages have the following structure:
	`_SDK.<class>.<category>.<name>` or `_SDK.<class>.<category>.<subject suffix>`
	For details, see *TIBCO Adapter SDK Programmer's Guide*.

Advanced Tab

You can use the **Advanced** tab to configure advanced options for an adapter configuration, such as termination subject or topic, debug level, alerter name, and so on.

The following table lists the advanced options:

Name	Description
General Information Panel	

Name	Description
Termination Subject or Topic	If a message is sent on the termination subject (if TIBCO Rendezvous is the transport) or topic (if JMS is the transport), the adapter will stop. The default value of the termination subject or topic is as follows: `%%Domain%%.%%Deployment%%.adb.%%InstanceId%%.exit` See *TIBCO Rendezvous Concepts* for information about specifying subject names. See *TIBCO Enterprise Message Service User's Guide* for information about publishing on a topic.
JMS Destination Prefix	You have to specify this prefix if you use the JMS transport type and have selected the **Use Separate Session** check box for Subscription Service. The destination prefix is the subject used by single-threaded Subscription Service. By default, the adapter uses a dynamic destination that is generated using the `Domain and Deployment` module properties, and the adapter configuration name. If you use this default dynamic destination, ensure that the values for `Domain and Deployment` are not empty. You can override the default dynamic destination by specifying the static destination in this field. The static destination must be defined with write permissions on the TIBCO EMS server before it can be used by the runtime adapter.
Debug Level	This field is valid only if the **Log Debug Messages** check box is selected, as described in Log Level Panel. Select how much debugging output to be provided in the console window or log file location specified in the **Log File** field. The options are: • Log no debug information • Log SQL commands executed against the database • Log binding data for each SQL Command (default) • Log all debug information
Generate Verbose Output	This field is valid only if the **Log Info Messages** check box is selected, as described in Log Level Panel. Select this check box to include verbose output (all available information) in the console window or log file location specified in the **Log File** field.
Generate Payload on Error	Select this check box to log the Subscription Service reply payload information under Error role when an error occurs in Subscription Service.

Name	Description
Script File	You can use the script file to change the location where the SQL script file is written.
Alerter Name	Name of the alerter service registered in the database.  When you use Oracle or Microsoft SQL Server databases, the value of the **Alerter Name** field cannot be null. If the value is null, the adapter configuration uses the existing adapter configuration name as the alerter name. If the adapter configuration name contains more than 30 characters, an error occurs in TIBCO Business Studio. This is because the alerter name length must be equal to or less than 30 characters.
Reconnection Configuration Panel	
Maximum Number of Reconnect Attempts	Total number of reconnection attempts to be made after a service is suspended. When this number is reached, the runtime adapter or adapter service is terminated. See the following two examples: • The value of `3` indicates that reconnection attempts continue 3 times. • The value of `-1` indicates that reconnection attempts continue indefinitely. To prevent data loss, it is good practice to configure preregistered listeners for a TIBCO Rendezvous Certified transport session. When a database connection is lost, reconnection attempts vary according to different adapter services: • Publication Service obtains an exception immediately and attempts to reconnect to the database. • Subscription Service and Request-Response Service do not obtain an exception until the services receive a message. After that, they attempt to reconnect to the database.  If you set a value for the `adb.RetryTotal <total reconnection attempts>` property in the `adbagent.tra` file, the value you set in TIBCO Business Studio is replaced.
Number of Reconnect Attempts Before Suspending Impacted Service(s)	Number of reconnection attempts to be made before a service is suspended. This field has a fixed value of `1`, and the value cannot be changed.

Name	Description
Interval between Reconnect Attempts (milliseconds)	Time interval (in milliseconds) at which reconnection attempts are made.  If you set a value for the `adb.SleepBetweenRetries <milliseconds of sleep between two reconnection attempts>` property in the `adbagent.tra` file, the value you set in TIBCO Business Studio is replaced.
Adapter Services General Configuration Panel	
Use Exception Table	Select this check box to use an exceptions table. The exceptions table is defined when you create Subscription Service.
Number of Request-Response Service Default Session Threads	Number of request-response threads to be used. Valid values are in the range `1 - n`. Each thread has a separate connection to the database. You can specify multiple threads to load balance incoming RPC requests.

Creating and Configuring Publication Service

When running as a publisher, the adapter extracts data in the changed rows from database tables and publishes them on appropriate subject names. You can create Publication Service and then configure the service by using the adapter service tabs.

For details, see the following configurations:

1. Creating Publication Service
2. Fetching a Table After Creating Publication Service
3. Specifying a Publishing Table
4. Specifying a Polling Method
5. Selecting a Storage Mode
6. Enabling Loop Detection

Creating Publication Service

Before you configure Publication Service, you have to create a service of Publication Service.

You can use the Create Service Wizard window to create an adapter service.

Prerequisites

An adapter configuration is created in a project.

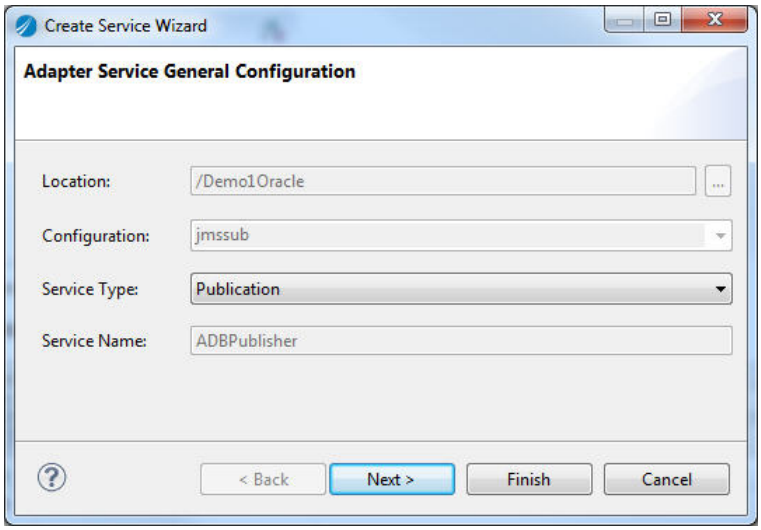
Procedure

1. Use one of the following ways to open the Create Service Wizard window:
   - In the Project Explorer view, right-click the **Adapter Services** folder of the adapter configuration, and from the pop-up menu, click **New Service**.
   - In the **All Adapter Services** panel of the **Adapter Services** tab, either click **Add**, or right-click the service list pane and from the pop-up menu, click **Add Service**.
   - In the Schema Browser view: fetch a table or stored procedure and then drag the fetched object into the All Adapter Services panel. For details on how to fetch an object in the Schema Browser view, see Fetching Tables and Stored Procedures.

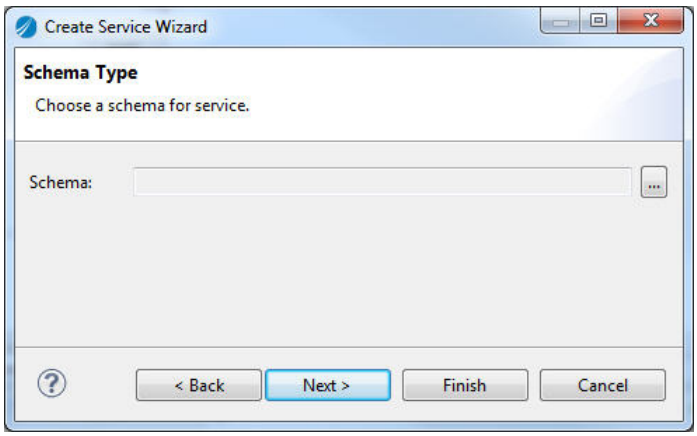
   

   The default service type and transport in the Create Service Wizard window are determined by the adapter preferences. To modify or maintain the adapter preferences, click **Window** > **Preferences** to open the Preferences window.
2. In the Adapter Service General Configuration window, select **Publication** from the **Service Type** list and click **Next**.

   The adapter **Location**, **Configuration**, and **Service Name** fields are filled automatically.



3. In the Schema Type window, select a table and click **Next**.



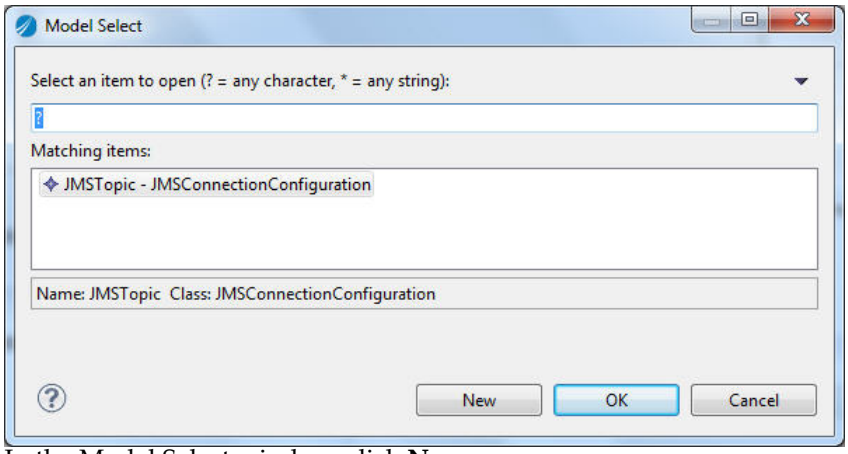
For the detailed procedure, see Fetching a Table When Creating Publication Service. If you open the Create Service Wizard wizard by using the Schema Browser view, the table fetched in the Schema Browser view is automatically populated in the **Schema** field, so you can skip this step.

You can also skip this step and add tables after you create Publication Service. For details on how to add tables for Publication Service, see Fetching a Table After Creating Publication Service.

4. Optional: In the Transport Session window, create a transport session for the service.

   The Transport Session window displays the default transport for a service. You can create a different transport session.

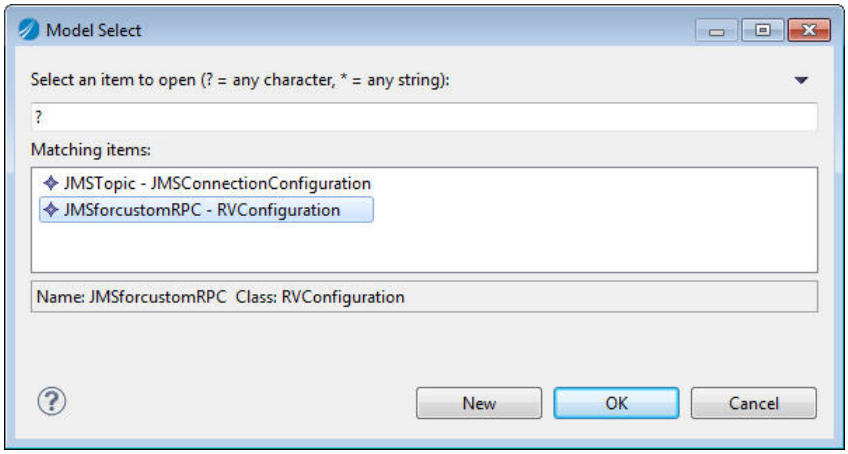
   a) Click the browse button ... to open the Model Select window.



b) In the Model Select window, click **New**.

c) In the New Transport window, specify a name for the transport you want to use, select a transport type and a specific option. Click **Finish**.

d) Back in the Model Select window, select the transport you just created and click **OK**.



5. Click **Finish** to exit the Create Service Wizard window.

What to do next

After you create Publication Service, the service-specific tabs are displayed in the configuration panel. You can use these tabs to configure Publication Service.

Fetching a Table for Publication Service

To use the adapter to interact with source and target applications, you must fetch tables for Publication Service.

You can use either of the following ways to fetch a table:

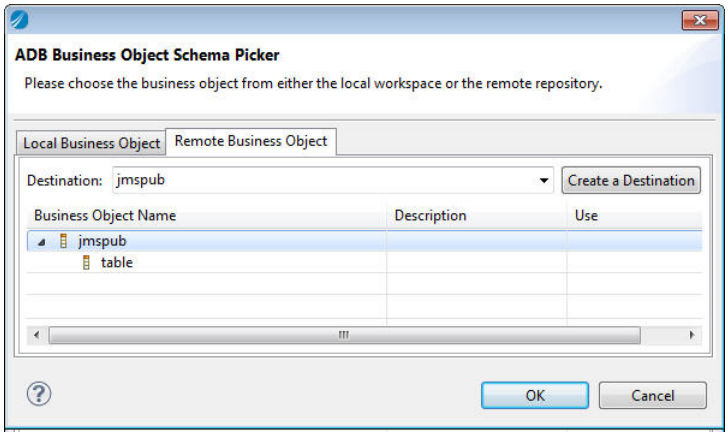
- Using the Create Service Wizard wizard. For details, see Fetching a Table When Creating Publication Service.
- Using the **Schema** tab. For details, see Fetching a Table After Creating Publication Service.

Fetching a Table When Creating Publication Service

You can use the Create Service Wizard wizard to fetch a table when you create Publication Service. In the Create Service Wizard wizard, after you select **Publication** from the **Service Type** list and click **Next**, the Schema Type window opens.

Procedure

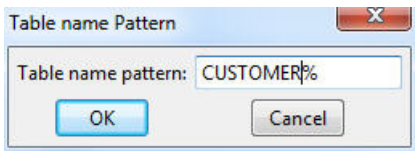
1. In the Schema Type window, click the browse button ... to open the ADB Business Object Schema Picker window.
2. Click the **Remote Business Object** tab and expand the destination with the adapter configuration name.



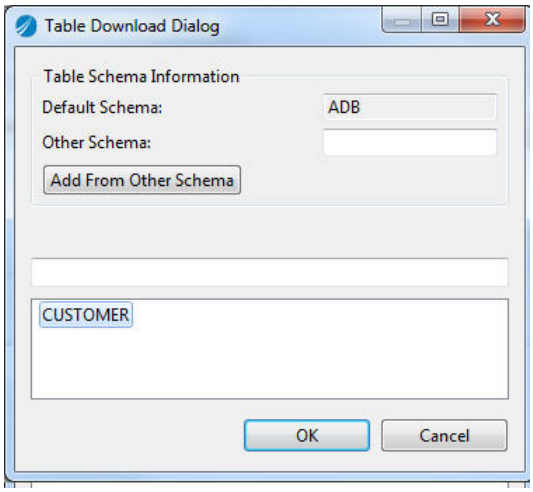
Use the **Remote Business Object** tab when a schema object that you want to fetch does not exist in the local repository, and use the **Local Business Object** tab when a schema object that you want to fetch exists in the local repository.

3. Right-click **table** and click **Fetch Table** to select a table.
4. In the "Table name Pattern" window, enter a search criteria and click **OK**.

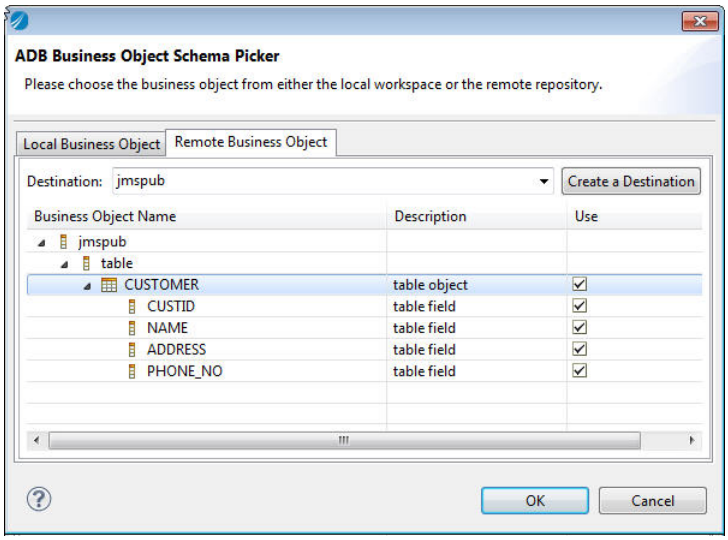
   For example, if you want to fetch a table that starts with "CUSTOMER", you can specify `CUSTOMER%` as the search criteria.



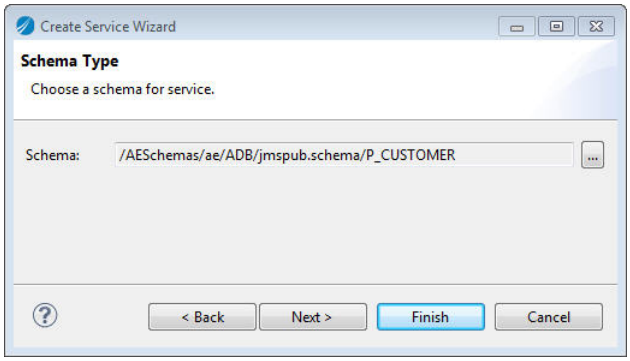
5. In the Table Download Dialog window, select the table you want to fetch and click **OK**.



6. Back in the ADB Business Object Schema Picker window, click the fetched table and click **OK**.



The fetched table is displayed in the **Schema** field.



Fetching a Table After Creating Publication Service

If you skip the Schema Type window when you create Publication Service, you can fetch a table in the **Schema** tab after creating the service.

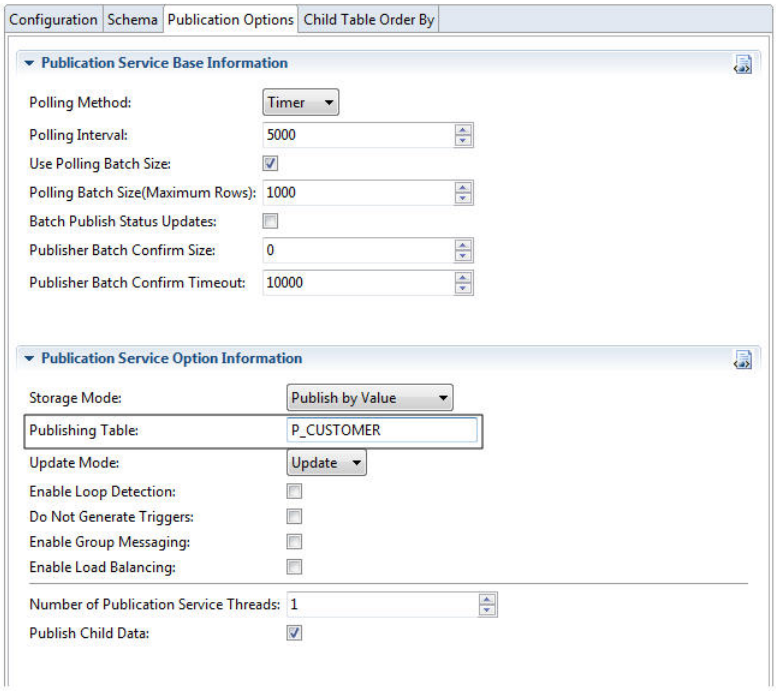
After you create Publication Service, the service-specific tabs are displayed in the configuration panel.

Procedure

1. In Publication Service, click the **Schema** tab and click **Add Table**  .
2. In the "Table name Pattern" window, specify a search criterion, such as `CUSTOMER%` and click **OK**.
3. In the "Select table" window, select **CUSTOMER** from the table list and click **Finish**.

   If you click **Next** in the "Select table" window, you can edit the selected table by selecting or clearing the **Use** check box next to a table column. If the **Use** check box next to a table column is cleared, the table column is not used. By default, all columns of the table are selected.
4. Click the **Publication Options** tab and verify that `P_CUSTOMER` is set in the **Publishing Table** field.

   If this value is not set, type this value in the field.



What to do next

If you want to use parent-child table relationships, you can add child tables in Publication Service. For details, see Adding Child Tables in Publication Service.

Specifying a Publishing Table

After you create Publication Service and select a table for the service, when you save the adapter service configuration, the **Publishing Table** field in the **Publication Options** tab is automatically filled with the name of the selected table, with the prefix "P_".

If you select the **Write to Database on Save** check box in the adapter **Configuration** tab, after you create Publication Service with a source table selected and then save the adapter service configuration, a publishing table with the name filled in the **Publishing Table** field is automatically created in the database. If you want to change the name of the publishing table, you can change the value in the **Publishing Table** field in the **Adapter Services** tab.

If you do not select the **Write to Database on Save** check box, after you create Publication Service with a source table selected and then save the adapter service configuration, you have to create a publishing table in the database based on the name in the **Publishing Table** field.

A publishing table contains additional columns that are used by the adapter to detect new rows. For details about the additional columns, see Publishing Table.

Specifying a Polling Method

Publication Service uses periodic polling or an alerter to monitor changes to a database table. You can select a polling method from the **Polling Method** list in the **Publication Options** tab.

Using a Timer

To use a timer, select **Timer** from the **Polling Method** list. Then specify a polling interval in milliseconds in the **Polling Interval** field. This sets how often Publication Service checks the publishing table for new rows.

For details about each field, see Publication Options Tab.

Using an Alerter

To use an alerter, select **Alerter** from the **Polling Method** list, and you have to set up and start the alerter on your database as well.

For more information about an alerter, see Working with an Alerter.

Selecting a Storage Mode

To copy columns from a source table to a publishing table, you can use the **Storage Mode** field in the **Publication Options** tab to select a storage mode. Publication Service provides two storage modes, Publish by Value and Publish by Reference. The selection depends on your specific requirements.

Publish by Value

With the Publish by Value storage mode, all specified columns in a source table are copied to the publishing table. Publish by Value makes for a fast copy, but does not support some data types, such as Oracle LONG and LONG RAW.

If you use Publish by Value, the following restrictions apply to publishing tables:

- Publishing tables cannot contain columns with the LONG data type. If a source table contains a column with the LONG data type, this column cannot be copied to the publishing table. This is because the trigger generated by the palette cannot copy the LONG column value through the `:new` construct.

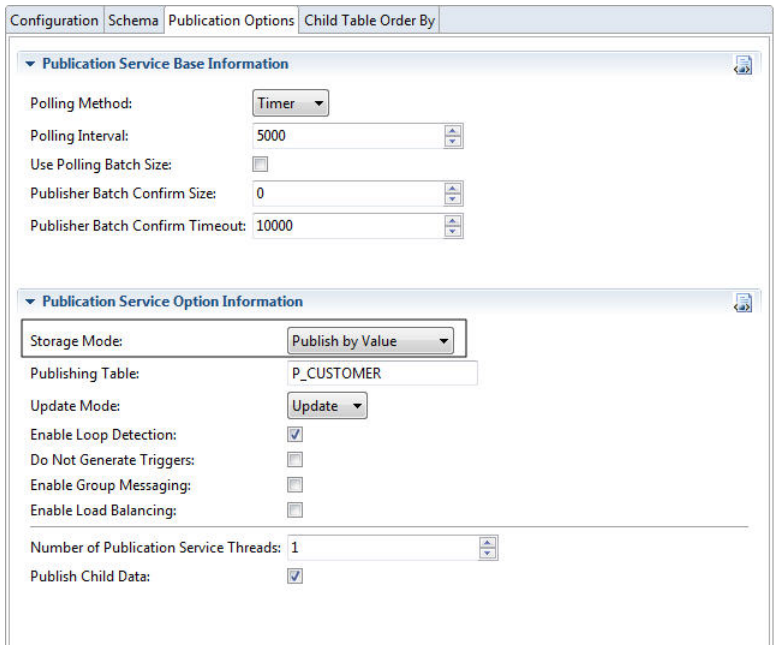
  This is an Oracle restriction documented in the *Oracle SQL Reference* manual. The problem is not detected by Oracle during trigger creation. However, when the trigger fires and attempts to copy the LONG column value to the publishing table, the database connection will hang for some time and then eventually terminate.
- When you define parent-child relationships between tables, the publishing table that is created for a parent table cannot contain a column with the LONG data type. However, a child table can contain a column with the LONG data type. This is because data in child table rows is not copied using the `:new` construct.
- The LONG RAW data is not supported in the publishing table.

These restrictions do not apply to publishing tables when you use the Publish by Reference storage mode, and LONG and LONG RAW are non-key types.

An example of Publish by Value is as follows:

A publishing table named P_CUSTOMER is created for the source table, CUSTOMER. When the CUSTOMER table is updated, the new data is copied to the P_CUSTOMER table. The adapter polls the P_CUSTOMER table and publishes the new data.

The following figure shows the Publish by Value storage mode.



If you use the Publish by Value storage mode, loop detection can be enabled. If Subscription Service exists and uses the same subject and the CUSTOMER table as the destination table, any changes to the CUSTOMER table are not published repeatedly.

Publish by Reference

With the Publish by Reference storage mode, only key column values are copied to the publishing table. By using this feature, Publication Service can publish data directly from the source table without copying the data from the source table to the publishing table first. A trigger, a stored procedure, and a publishing table are created, but the publishing table contains the necessary adapter fields and only the key fields of the source table.

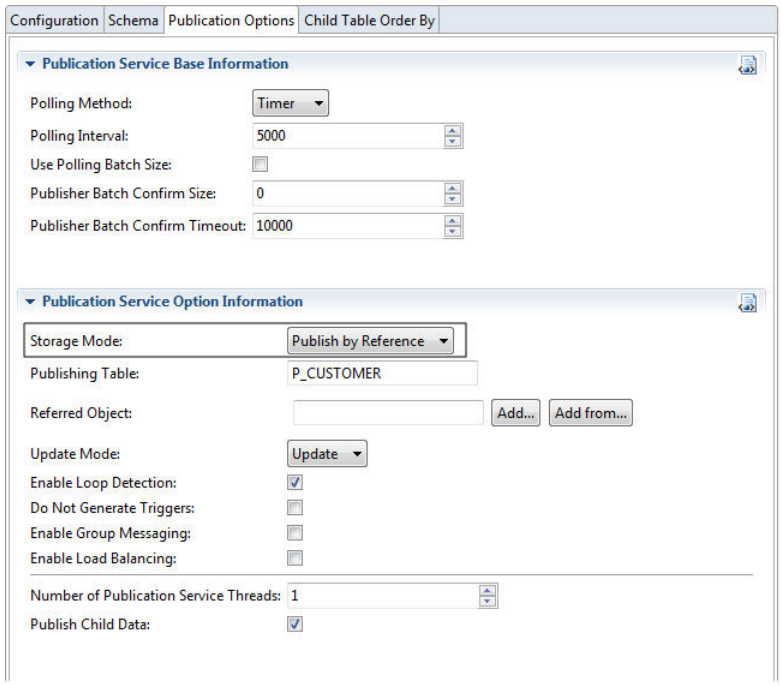
The advantage of Publish by Reference is that the data to be published is stored just once. Also, data types, such as Oracle LONG and LONG RAW are supported for Publish by Reference.

A key column or substitute key column is required when you use Publish by Reference, because the publishing table contains only key values. If no column is specified, the publication is not added.

An example of Publish by Reference is as follows:

A publisher endpoint is configured to publish from the P_CUSTOMER table with the **CUST_ID** key field. The publishing table is created with the necessary fields and the **CUST_ID** field. When a row in the P_CUSTOMER table is modified, the trigger fires, populates fields, and copies the **CUST_ID** field value to the publishing table. When the adapter polls the publishing table, it detects the new row and selects it from the P_CUSTOMER table by using the **CUST_ID** field value found in the publishing table. Then the message is published.

The following figure shows the Publish by Reference storage mode.



You can load a source table from the default schema or add a table from a different schema. By default, the publishing table is created in the default schema. However, if you specify a prefix in the publishing table name, the publishing table is created in the schema with the specified prefix.

Publish by Reference Object

To use a view or a different database object as the source table, you can configure the adapter to publish data by a reference object, where key columns are stored in the publishing table and data to be published is selected from the referred object.

Similar to the Publish by Reference feature, Publish by Reference Object copies only key values from the source table to the publishing table.

The difference is that, if Publish by Reference Object is used, when a row changes in the source table and the associated trigger fires, the adapter fetches data from the referred object instead of the source table. The name of the referred object is stored in the **ADB_REF_OBJECT** column in the publishing table. It is good practice to use Publish by Reference Object when a view provides the most efficient access to the source data, for example, when many levels of nesting exist between a parent table and a child table.

When you use the Publish by Reference Object feature, note the following conditions:

- No matter if a single table or parent-child tables are used as a source table, even if you do not specify a referred object, the adapter can run properly without additional configurations for Publication Service.
- A referred object can be a table or a view in the default schema. However, if you add a prefix in the referred object name, the table or view in the schema with the specified prefix is used for reference.
- If you specify a table or view as a referred object, the referred object must contain columns with the same names and data types as the primary key columns in the source table. The primary key values in the source table must be the same as the primary key values or the subset of primary key values in the referred object.

For details on how to use Publish by Reference Object, see Using Publish by Reference Object.

Enabling Loop Detection

If a table is used as both a source table and a destination table for the same subject, a loop occurs and the same changes are published repeatedly. To prevent this issue, you can enable the loop detection feature by selecting the **Enable Loop Detection** check box in the **Publication Options** tab.

If the loop detection feature is enabled, an additional column, ADB_SOURCE, is added to the source table. When the adapter receives a message, it inserts or updates a row into the source table and enters a T in the ADB_SOURCE column to denote that this row was inserted or updated according to the received message, rather than user intervention.

Triggers created by the adapter are defined not to copy the rows with T in the ADB_SOURCE column to the publishing table, which means that the rows will not be published. If you must update a received row (that is, has a T in the ADB_SOURCE column) and want the updated row published, you have to change the ADB_SOURCE column to NULL, and then the trigger retrieves the row and sends it.

Loop detection is disabled for DB2 on z/OS because DB2 on z/OS does not support the drop column feature.

Master-Master Replication

You can use the loop detection feature to implement a simple master-master replication scheme. When master-master replication is implemented, multiple sites act as peers to copy and maintain groups of replicated objects.

When loop detection is enabled, the adapter can be configured as both a Publisher endpoint and a Subscriber endpoint to the same table on the same subject. When the Subscriber endpoint receives a message, it compares the adapter ID of the message with its own adapter ID. If the values of the adapter IDs are the same, and the source and destination tables are the same table in the database, the Subscriber endpoint discards the message. Other Subscriber endpoints listening on the subject receive the message only once.

The adapter does not resolve any replication conflicts, for example, two applications updating the same row of their corresponding source table both publish the change. Regarding advanced replication scenarios, use the replication tools provided by your database vendor.

Creating and Configuring Subscription Service

When running as a subscriber, the adapter listens on a subject, receives messages, and updates the relevant tables in its associated database. The data is then available to other applications that have access to the database.

For details, see the following configurations:

1. Creating Subscription Service
2. Fetching a Table After Creating Subscription Service
3. Specifying an Exceptions Table
4. Using an Exceptions Table as a Source Table
5. Specifying an Opaque Exceptions Table
6. Specifying a Precommit Stored Procedure Call
7. Configuring a Subscriber Reply Sender

Creating Subscription Service

Before you configure Subscription Service, you have to create a service of Subscription Service.

You can use the Create Service Wizard window to create an adapter service.

Prerequisites

An adapter configuration is created in a project.

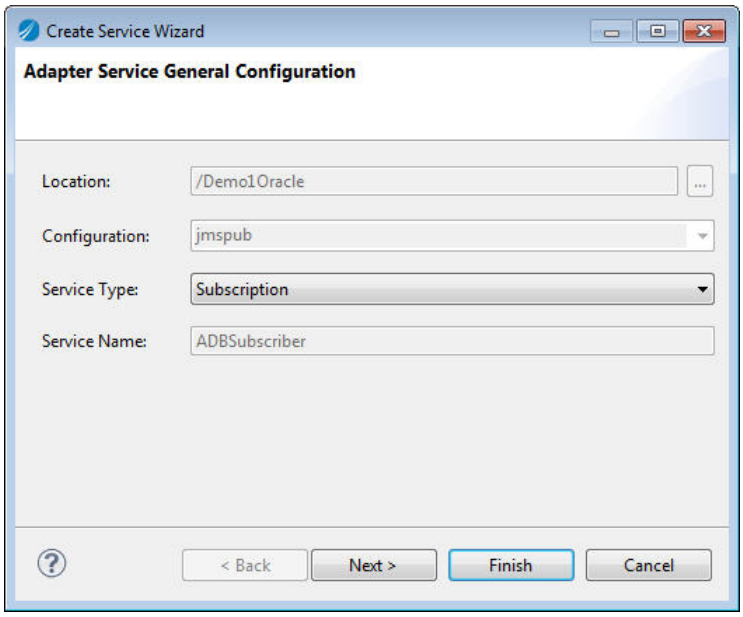
Procedure

1. Use one of the following ways to open the Create Service Wizard window:
   - In the Project Explorer view, right-click the **Adapter Services** folder of the adapter configuration, and from the pop-up menu, click **New Service**.
   - In the **All Adapter Services** panel of the **Adapter Services** tab, either click **Add**, or right-click the service list pane and from the pop-up menu, click **Add Service**.
   - In the Schema Browser view: fetch a table or stored procedure and then drag the fetched object into the All Adapter Services panel. For details on how to fetch an object in the Schema Browser view, see Fetching Tables and Stored Procedures.

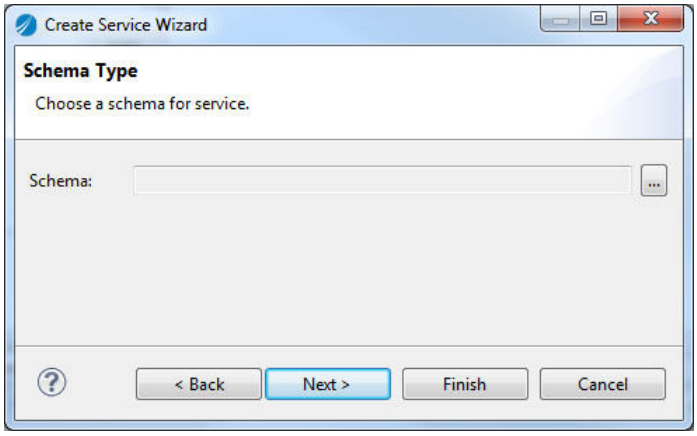
   

   The default service type and transport in the Create Service Wizard window are determined by the adapter preferences. To modify or maintain the adapter preferences, click **Window** > **Preferences** to open the Preferences window.
2. In the Adapter Service General Configuration window, select **Subscription** from the **Service Type** list and click **Next**.

   The adapter **Location**, **Configuration**, and **Service Name** fields are filled automatically.



3. In the Schema Type window, select a table and click **Next**.



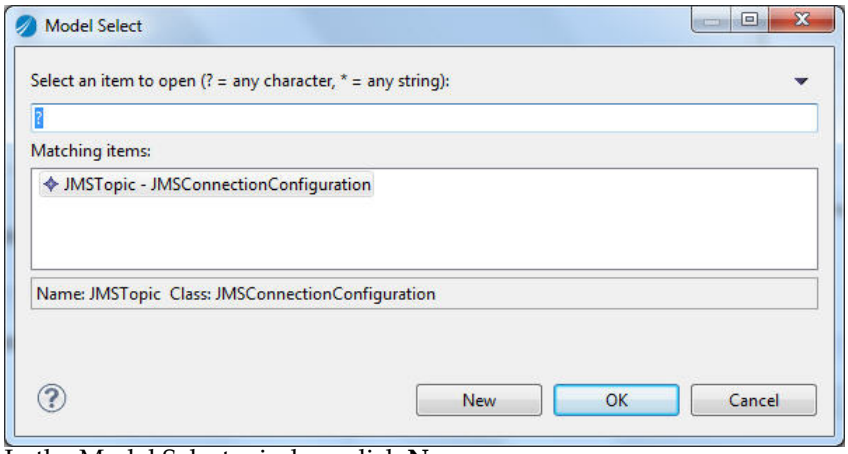
For the detailed procedure, see Fetching a Table When Creating Subscription Service. If you open the Create Service Wizard wizard by using the Schema Browser view, the table fetched in the Schema Browser view is automatically populated in the **Schema** field, so you can skip this step.

You can also skip this step and add tables after you create Subscription Service. For details, see Fetching a Table After Creating Subscription Service.

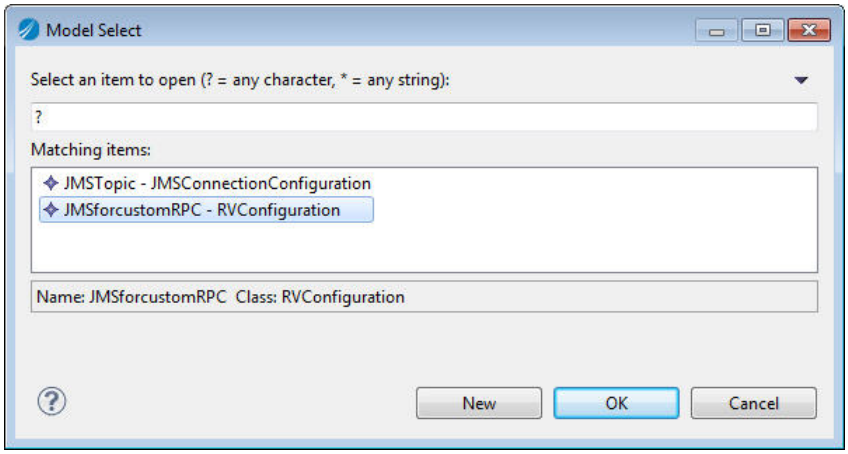
4. Optional: In the Transport Session window, create a transport session for the service.

   The Transport Session window displays the default transport for a service. You can create a different transport session.

   a) Click the browse button ... to open the Model Select window.



b) In the Model Select window, click **New**.
c) In the New Transport window, specify a name for the transport you want to use, select a transport type and a specific option. Click **Finish**.
d) Back in the Model Select window, select the transport you just created and click **OK**.



5. Click **Finish** to exit the Create Service Wizard window.

What to do next

After you create Subscription Service, the service-specific tabs are displayed in the configuration panel. You can use these tabs to configure Subscription Service.

Fetching a Table for Subscription Service

To use the adapter to interact with source and target applications, you must fetch tables for Subscription Service.

You can use the following ways to fetch a table:

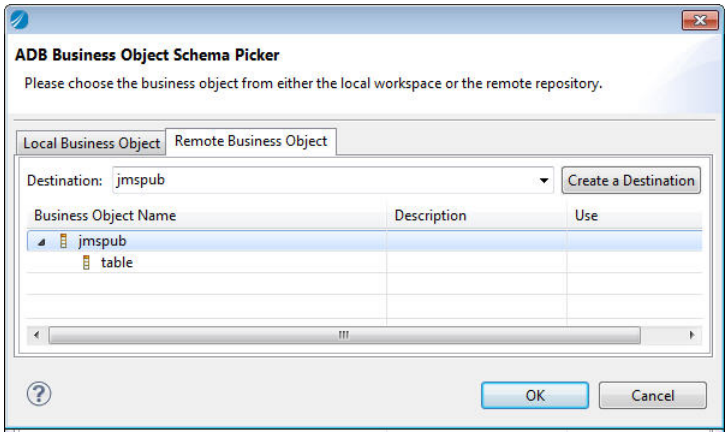
- Using the Create Service Wizard wizard. For details, see Fetching a Table When Creating Subscription Service.
- Using the **Schema** tab. For details, see Fetching a Table After Creating Subscription Service.

Fetching a Table When Creating Subscription Service

You can use the Create Service Wizard wizard to fetch a table when you create Subscription Service. In the Create Service Wizard wizard, after you select **Subscription** from the **Service Type** list and click **Next**, the Schema Type window opens.

Procedure

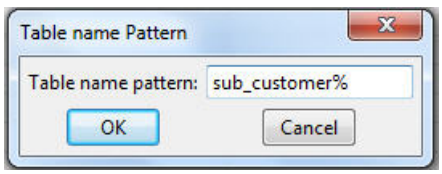
1. In the Schema Type window, click the browse button ... to open the ADB Business Object Schema Picker window.
2. Click the **Remote Business Object** tab and expand the destination with the adapter configuration name.



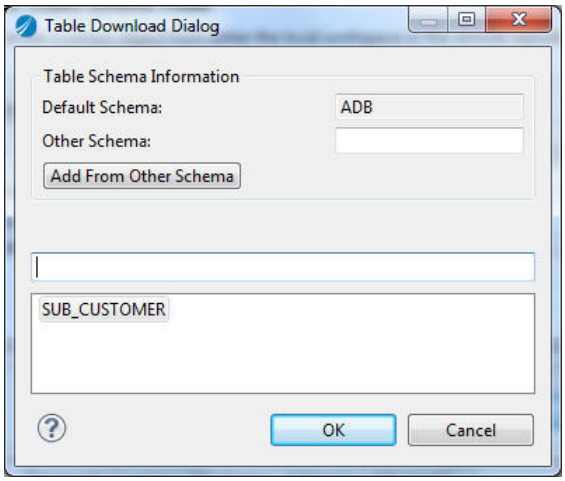
Use the **Remote Business Object** tab when a schema object that you want to fetch does not exist in the local repository, and use the **Local Business Object** tab when a schema object that you want to fetch exists in the local repository.

3. Right-click **table** and click **Fetch Table** to select a table.
4. In the "Table name Pattern" window, enter a search criteria and click **OK**.

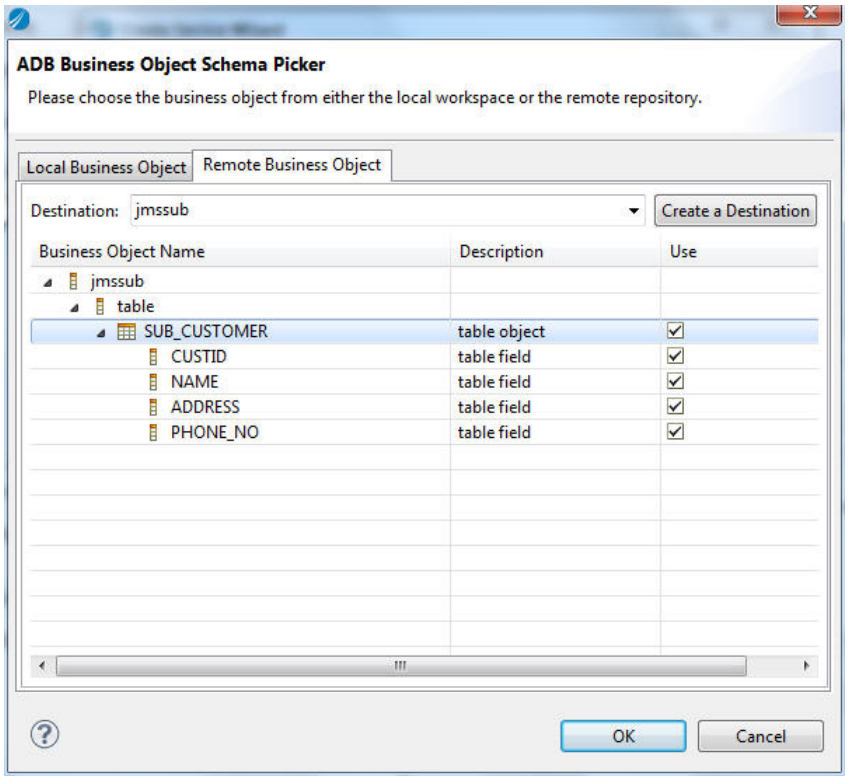
   For example, if you want to fetch a table that starts with "SUB_CUSTOMER", you can specify `SUB_CUSTOMER%` as the search criteria.



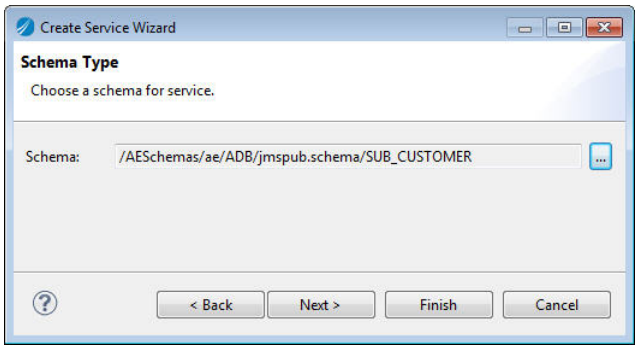
5. In the Table Download Dialog window, select the table you want to fetch and click **OK**.



6. Back in the ADB Business Object Schema Picker window, click the fetched table and click **OK**.



The fetched table is displayed in the **Schema** field.



Fetching a Table After Creating Subscription Service

If you skip the Schema Type window when you create Subscription Service, you can fetch a table in the **Schema** tab after creating the service.

After you create Subscription Service, the service-specific tabs are displayed in the configuration panel.

Procedure

1. In Subscription Service, click the **Schema** tab and click **Add Table** .
2. In the "Table name Pattern" window, specify a search criterion, such as `SUB_CUSTOMER%` and click **OK**.
3. In the "Select table" window, select **SUB_CUSTOMER** from the table list and click **Finish**.

   If you click **Next** in the "Select table" window, you can edit the selected table by selecting or clearing the **Use** check box next to a table column. If the **Use** check box next to a table column is cleared, the table column is not used. By default, all columns of the table are selected.

What to do next

If you want to use parent-child table relationships, you can add child tables in Subscription Service. For details, see Adding Child Tables in Subscription Service.

Specifying an Exceptions Table

After fetching a table in Subscription Service, you can specify an exceptions table for Subscription Service. If an error occurs when the subscriber endpoint inserts, updates, or deletes data in the destination table, the subscriber endpoint writes the data to the exceptions table to record the error.

Procedure

1. In the configuration panel of Subscription Service, click the **Subscription Options** tab.
2. Specify a name, such as `EXCP_CUSTOMER` in the **Exceptions Table** field. Click **Save** on the toolbar.
   When you specify an exceptions table, the adapter transforms the table structure into an AE schema. To open the AE schema, click the **Class Reference** link in the Schema panel in the adapter service **Configuration** tab.

   For details about the columns in an exceptions table, see Exceptions Table.
3. Click the **Configuration** tab of the adapter configuration and keep the **Write to Database on Save** check box selected. Click **Save** on the toolbar.
   This setting ensures that such exceptions table is created in the database.

Configuring an Exceptions Table as a Source Table

If you want to publish data from an exceptions table and also want to use this exceptions table as the source table, you cannot use the ADB_ERROR_TEXT or ADB_OPCODE columns.
To avoid using the ADB_ERROR_TEXT or ADB_OPCODE columns in the exceptions table, you can rename these columns.

Procedure

1. Create a database view that mirrors the exceptions table, but rename the ADB_ERROR_TEXT and ADB_OPCODE columns so that they do not begin with "ADB_."
2. Use the Publish by Reference Object feature and select your view as the referred object. For details about Publish by Reference Object, see Specifying a Reference Object.

Specifying an Opaque Exceptions Table

You can specify an Opaque Exceptions table for Subscription Service. If a subscriber endpoint fails to generate records in the destination table or fails to insert a message into an exceptions table, the subscriber endpoint logs the error into the opaque exceptions table.

Procedure

1. In the **Subscription Options** tab of Subscription Service, select the **Use Opaque Exceptions Table** check box.
2. Enter a table name in the **Opaque Exceptions Table** field.
   When you specify an opaque exceptions table, the adapter transforms the table structure into an AE schema. Click the **Class Reference** link in the **Schema** panel in the **Configuration** tab to open the AE schema.

   For details about the columns in an opaque exceptions table, see Opaque Exceptions Table.

Specifying a Precommit Stored Procedure Call

You can configure Subscription Service to call a stored procedure after a database inserts, updates, or deletes data and before committing the data. You can use this stored procedure as a hook to accomplish further processing inside the database and have results returned to the adapter.

To specify a precommit stored procedure call, specify the precommit stored procedure name in the **Pre-Commit Stored Procedure** field of the **Subscription Options** tab.

The adapter calls the precommit stored procedure with the following syntax:

```
{call <pre-commit stored procedure name>(?, ?, ?)}.
```

This stored procedure has to be defined with a specific interface as described in the following table:

Name	Type	Description
RETURN_CODE	Integer	`RETURN_CODE = 0`, the adapter assumes the procedure was successful and writes a success message to the SDK INFO trace role when the verbose mode is used. `RETURN_CODE <> 0`, the adapter assumes the procedure was not successful and writes SP_TEXT to the SDK ERROR trace role (whether verbose mode is `on` or `off`). If the message has a reply subject, this output value will be returned to the message sender.
SP_TEXT	Varchar	If the message has a reply subject, this output string will be returned to the message sender.
DO_ROLL BACK	Integer	DO_ROLLBACK = 0, the adapter commits the transaction and confirms the original message. DO_ROLLBACK <> 0, the adapter rolls back the transaction and does not confirm the message.  Not confirming the message changes the RVCM behavior. All subsequent RVCM messages will not be confirmed. It can only be done when this RVCM behavior is really desired.

Configuring a Subscriber Reply Sender

A subscriber endpoint can be configured to send the subscription status as a reply to the message sender if the message contains a reply subject. The ADB_SUBSCRIBER_STATUS AE schema is used for this reply message.

The following table lists and describes the columns in the ADB_SUBSCRIBER_STATUS AE schema:

Column	Type	Description
RETURN_CODE	i4	Different values of the RETURN_CODE column are returned on the following conditions: • `0`: when data is inserted into the destination table successfully. • `-1`: when the data is inserted into the exceptions table or opaque exceptions table. • `5`: when the data fails to be inserted in both destination table and exceptions table (or opaque exceptions table). For example, if the data fails to be inserted into the destination table and you do not configure the exceptions table or opaque exceptions table, the value of the RETURN_CODE column is `5`; if the data fails to be inserted into the exceptions table and you do not configure the opaque exceptions table, the value of the RETURN_CODE column is also `5`. If you specify a subscriber precommit stored procedure, this column contains the value of the **`RETURN_CODE`** output parameter.  The RETURN_CODE column does not support bulk insert operations.
ADB_TEXT	string	Contains the status or error message of the subscriber operation.  The ADB_TEXT column does not support bulk insert operations.
NUMBER_OF_UPDATED_ROW	i4	Indicates the number of records that have been updated when you execute a SQL update statement. This column helps you track the current behavior of Subscription Service.  The following situations affect the value of this column: • When Subscription Service receives a group message, the value of the NUMBER_OF_UPDATED_ROW column is the sum of the messages updated in the database. • When Subscription Service receives a parent-child message, the value of the NUMBER_OF_UPDATED_ROW column is the number of rows of the parent table that are updated in the database. • If a record is inserted into the exceptions table or opaque exceptions table, the value of the NUMBER_OF_UPDATED_ROW column is `0`.  The NUMBER_OF_UPDATED_ROW column does not support bulk insert operations.

Column	Type	Description
CALLOUT_TEXT	string	Contains an error message from the database driver in case of an error operation. If you specify a subscriber precommit stored procedure, this field contains the value of the **SP_TEXT** output parameter.  The CALLOUT_TEXT column does not support bulk insert operations.

Creating and Configuring Request-Response Service

Request-Response Service often acts as a Request Reply server or Remote Procedural Call (RPC) server. When running Request-Response Service, the adapter receives requests from other applications, parses them, and translates them into queries to the database to retrieve data. The output data is wrapped in an MInstance and sent back to the caller.
For details, see the following configurations:

1. Request-Response Implementation
2. Creating Request-Response Service
3. Fetching a Schema Object for Custom RPC Service
4. Selecting a Service Mode
5. Using Quotes in Microsoft SQL Server
6. Using the REF Data Type with Oracle Databases

Request-Response Implementation

With Request-Response Service, client applications can send SQL statements, stored procedures, functions, or packages on a specified subject to an adapter configuration. The adapter processes the request and returns the results in a response to the client.

Request-Response Service works in the following process:

1. A client application sends a request on a subject to the adapter. The request might be a query to the database or any DDL or DML command to be performed on the database. Multiple statements, stored procedures, or both can be sent in a single message.
2. Upon receipt of the message, the adapter executes the statements, stored procedures, or both. Requests within a message are processed as a single transaction.
3. The adapter sends back a response to the Inbox, subject, or response subject that was set up by the client application. The response consists of a result code or one or more result sets, based on the request. A response can also be an error code or error description, if the request was not successful.

Multiple adapter configurations can be configured so that any one of the adapter configurations receives and processes the request. In case of multiple adapter configurations, you can use TIBCO Rendezvous Distributed Queue or JMS Queue to balance the load across a number of adapter configurations in a queue. The task is assigned to the least loaded member of the queue. For details, see Load Balancing.

For details about requests and responses, see Requests and Responses.

Requests

A request contains one or more SQL statements, stored procedures, functions, or packages to be executed as a transaction.

The text of the SQL statement follows the conventions for JDBC SQL syntax. All SQL statements supported by the DBMS and placeholders (represented by a question mark, '?') can be used in a SQL statement, conforming to the JDBC rules. For performance reasons, it is good practice to use a SQL statement. The '?' convention is only used to bind binary data or call stored procedures.

The adapter listens on the subject on which a client application sends requests. The application uses TIBCO Rendezvous or TIBCO Adapter SDK to send a self-describing message containing a request on the agreed subject to the adapter configuration.

A request supports the following conditions:

- DDL and DML SQL statements.
- The column size of the returned result set cannot exceed 128 characters.
- Multiple statements or procedures in a single transaction that send a nested TIBCO Rendezvous message.
- `rv_Send()`, `rv_SendWithReply()`, and `rv_Rpc()` calls.
- TIBCO Rendezvous Java APIs.

If the `rv_SendWithReply()` or `rv_Rpc()` call is used, the adapter sends a response to the Inbox subject that was set by the client application. If the `rv_SendWithReply()` or `rv_Rpc()` call is not used, the adapter sends a response on the response subject that was set at configuration time.

TIBCO ActiveEnterprise or XML Message Request Format

In TIBCO ActiveEnterprise or XML Message Request format, the input class is SQL_REQUEST. Also see "SQL_STATEMENT Class" in Standard RPC Operation.

The SQL_REQUEST class is described as follows.

```
<object name="SQL_REQUEST" lastModified="1036435805361" id="503">
<assoc name="attribute">
<string name="name" value="STATEMENTS"/>
<ref name="attributeType"
value="/tibco/public/sequence/ae/class/ae/ADB/adbmetadata/sequence
[SQL_STATEMENT]"/>
<string name="isKey" value="false"/>
<string name="isReadable" value="true"/>
<string name="isWriteable" value="true"/>
</assoc>
<assoc name="attribute">
<string name="name" value="CLOSURE"/>
<ref name="attributeType" value="/tibco/public/scalar/ae/any"/>
<string name="isKey" value="false"/>
<string name="isReadable" value="true"/>
<string name="isWriteable" value="true"/>
</assoc>
<string name="family" value="ae"/>
<string name="objectType" value="class"/>
</object>
```

Responses

A response from the adapter to a client application has a result code and one or more result sets. Each result set contains nested self-describing messages, each of which encodes a result row, such as that returned from a query. A response can also return an error code and error description if the request was not successful.

TIBCO ActiveEnterprise or XML Message Response Format

In TIBCO ActiveEnterprise Message or XML format, the default output class is SQL_BATCHRETURN.

For details about the SQL_BATCHRETURN class, see "SQL_ BATCHRETURN Class" in Standard RPC Operation.

Creating Request-Response Service

Before you configure Request-Response Service, you have to create a service of Request-Response Service.

You can use the Create Service Wizard window to create an adapter service.

Prerequisites

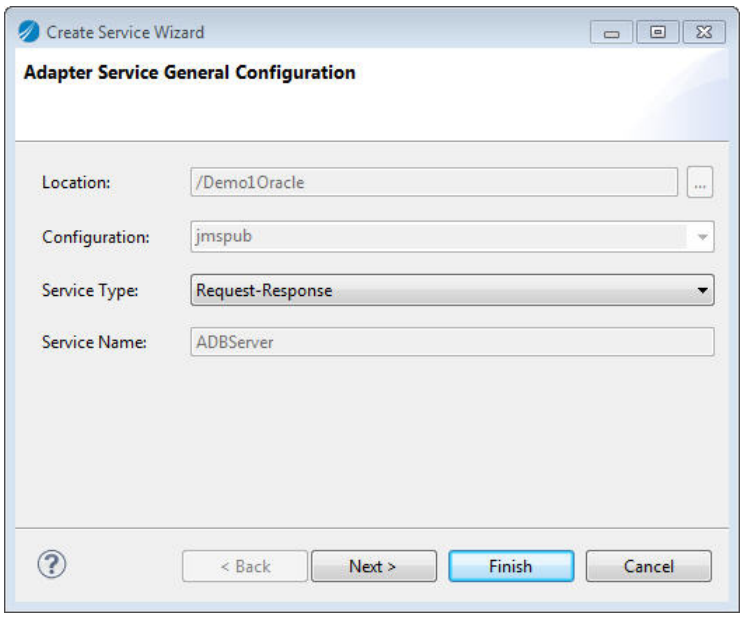
An adapter configuration is created in a project.

Procedure

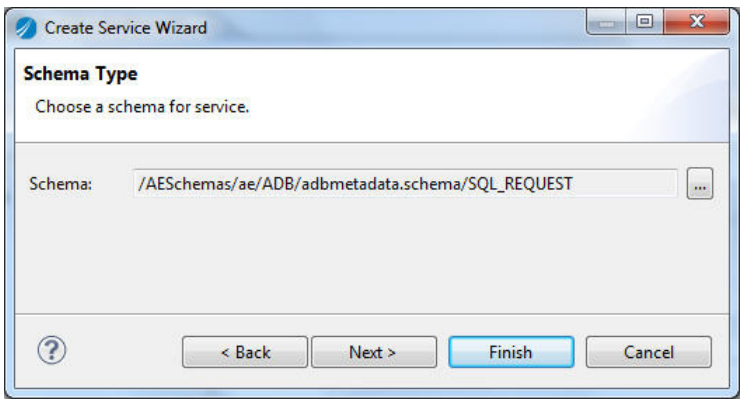
1. Use one of the following ways to open the Create Service Wizard window:
   - In the Project Explorer view, right-click the **Adapter Services** folder of the adapter configuration, and from the pop-up menu, click **New Service**.
   - In the **All Adapter Services** panel of the **Adapter Services** tab, either click **Add**, or right-click the service list pane and from the pop-up menu, click **Add Service**.
   - In the Schema Browser view: fetch a table or stored procedure and then drag the fetched object into the All Adapter Services panel. For details on how to fetch an object in the Schema Browser view, see Fetching Tables and Stored Procedures.

   The default service type and transport in the Create Service Wizard window are determined by the adapter preferences. To modify or maintain the adapter preferences, click **Window** > **Preferences** to open the Preferences window.
2. In the Adapter Service General Configuration window, select **Request-Response** from the **Service Type** list and click **Next**.
   The adapter **Location**, **Configuration**, and **Service Name** fields are filled automatically.



3. In the Schema Type window, select a stored procedure, function, or package and click **Next**.
   By default, the **Schema** field is automatically filled with the SQL_REQUEST class.

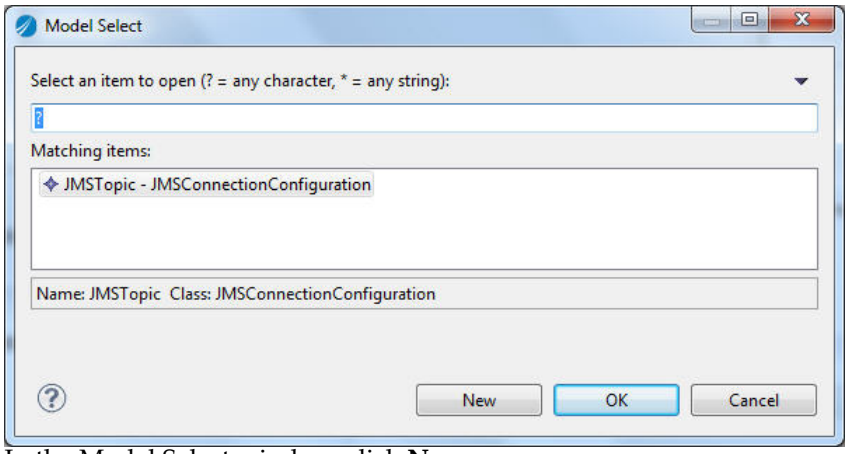


   For the detailed procedure, see Fetching a Schema Object for Custom RPC Service.

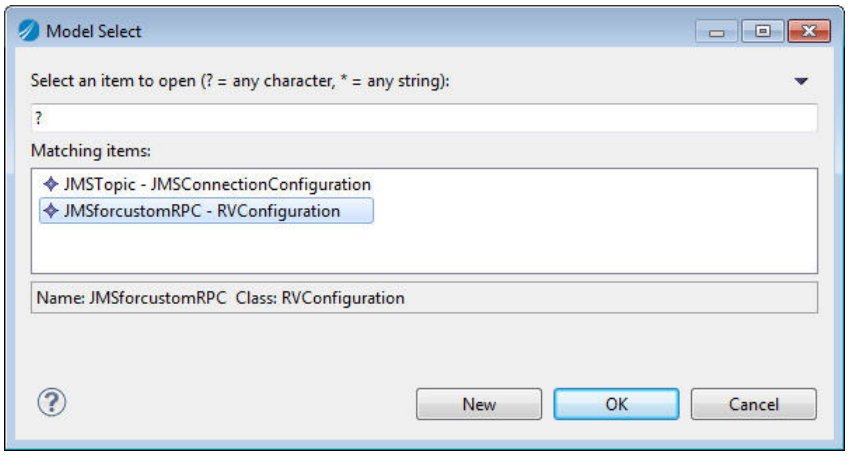
4. Optional: In the Transport Session window, create a transport session for the service.

   The Transport Session window displays the default transport for a service. You can create a different transport session.

   a) Click the browse button ... to open the Model Select window.



   b) In the Model Select window, click **New**.
   c) In the New Transport window, specify a name for the transport you want to use, select a transport type and a specific option. Click **Finish**.
   d) Back in the Model Select window, select the transport you just created and click **OK**.



5. Click **Finish** to exit the Create Service Wizard window.

What to do next

After you create Request-Response Service, the service-specific tabs are displayed in the configuration panel. You can use these tabs to configure Request-Response Service.

Fetching a Schema Object for Custom RPC Service

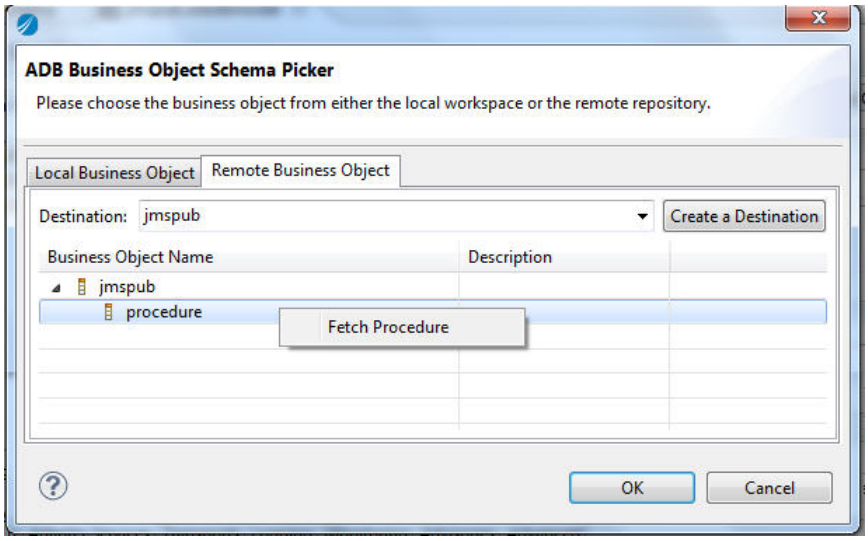
You must fetch a stored procedure, function, or package when you create custom RPC Service. However, you do not have to fetch a schema object for Request-Reply RPC Service and standard RPC Service.

In the Create Service Wizard wizard, after you select **Request-Response** from the **Service Type** list and click **Next**, the Schema Type window opens.

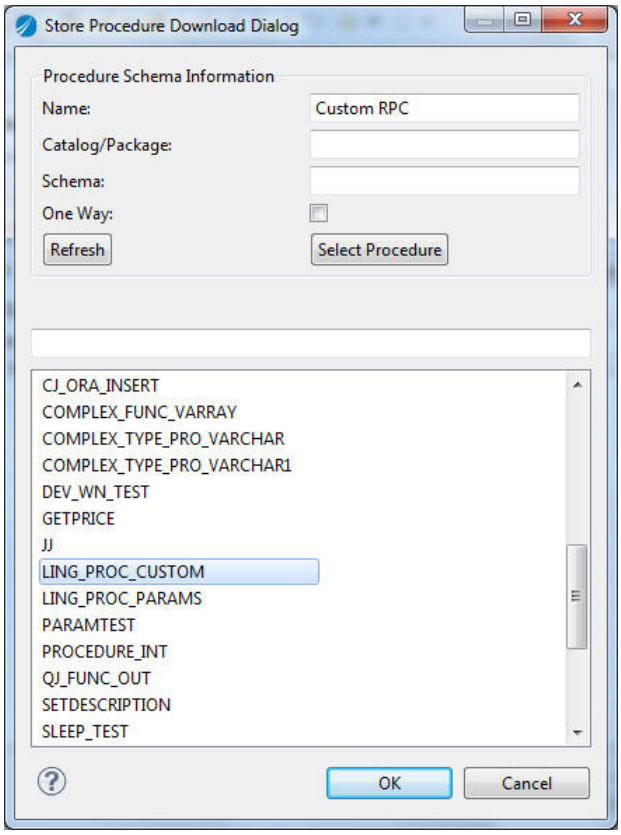
Procedure

1. In the Schema Type window, click the browse button ... to open the ADB Business Object Schema Picker window.
2. Click the **Remote Business Object** tab and expand the destination with the adapter configuration name.

   Use the **Remote Business Object** tab when a schema object that you want to fetch does not exist in the local repository, and use the **Local Business Object** tab when a schema object that you want to fetch exists in the local repository.
3. Right-click **procedure** and click **Fetch Procedure**.



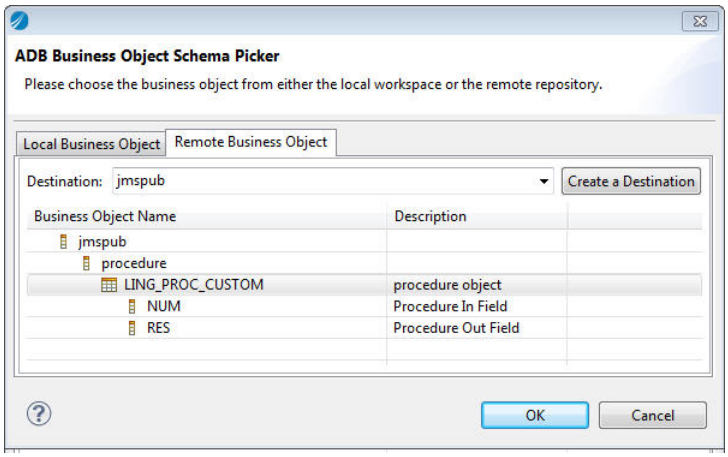
4. In the Store Procedure Download Dialog window, specify a name for the call operation and select a stored procedure, function, or package from the schema object list. Then click **OK**.



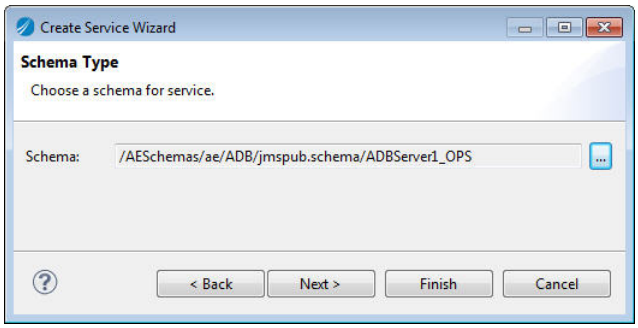
You can also specify the options in the Procedure Schema Information panel. For details, see Procedure Schema Information Reference.

TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) retrieves the signature of each stored procedure or function from the database. If you change the stored procedure or database connection while editing your project, you must return to the Store Procedure Download Dialog window and click **Refresh** to retrieve the changes from the database.

5. Back in the ADB Business Object Schema Picker window, click the fetched object and click **OK**.



The fetched object is displayed in the **Schema** field.



Procedure Schema Information Reference

The Procedure Schema Information panel in the Store Procedure Download Dialog window contains the following options:

Name	Description
Name	Name of the call operation.
Catalog/Package	(Optional) Only applicable to a database that has more than one catalog or package. The catalog or package in which the procedure is stored. This name is used to resolve naming conflicts when more than one catalog or package in the database has the selected procedure with the same name. See your database documentation for more information about catalogs and packages.
Schema	(Optional) The schema is the collection of database objects. Schema objects are the logical structure that directly refer to the database data. In TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio), this is a mandatory field for most databases. Generally, the name of the default schema has to be the same as the user name.
One Way	(Optional) Select this check box to execute the procedure by using a one-way operation. Clear this check box to execute the procedure by using a two-way operation. If you select the one-way operation to execute the procedure, the service will send a message without expecting a response.

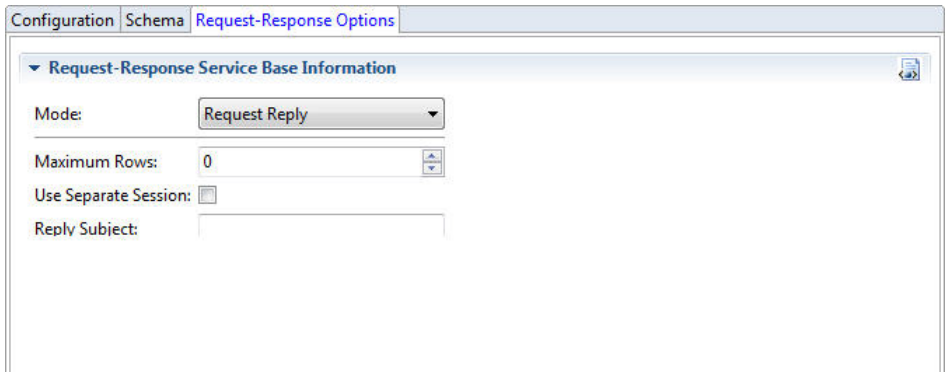
Selecting a Service Mode

You can configure Request-Response Service to use different modes to interact with other applications. Request-Response Service supports three service modes: Request-Reply mode, standard RPC mode, and custom RPC mode. You have to select a service mode depending on your specific requirements.

Configuring Request-Reply Mode

To configure Request-Reply mode, select **Request Reply** from the **Mode** list in the **Request-Response Options** tab.

The following figure shows the configuration of Request Reply mode:



Request Response Service in Request Reply mode can be regarded as Subscription Service with SQL statement-based schema. In Request-Reply mode, a subscriber endpoint is created. This subscriber endpoint listens to requests and publishes the response to the reply subject.

TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) supports the following message formats for Request-Reply mode:

- XML Message (TIBCO Rendezvous transport type or JMS transport type)
- ActiveEnterprise Message (TIBCO Rendezvous transport type)

Configuring RPC Mode

To configure RPC mode, select **RPC** from the **Mode** list in the **Request-Response Options** tab. RPC mode involves standard RPC operation and custom RPC operation. For details, see Using RPC Mode.

Using RPC Mode

You can configure an adapter configuration to act as a remote procedure call (RPC) server on behalf of a client. TIBCO ActiveMatrix Adapter for Database uses the TIBCO Adapter SDK Operations API for an RPC server. Selecting RPC mode creates an object in the repository to describe the RPC server that the adapter configuration starts.
The structure of the TIBCO Repository class objects that provide MOperation support is presented. Based on these class descriptions, you can create an RPC client to send requests to the adapter in the expected structure. For convenience, class structure is shown in XML format.

One-Way and Two-Way Invocation

You can use one-way or two-way messaging in an RPC operation:

One-way Messaging

The BusinessWorks client sends a request containing a message to the adapter server, and no reply is expected from the adapter server. The client does not have to wait for the reply and is free to continue processing the request. The request is not always sent immediately to the server but waits in a queue until the server processes it.

Two-way messaging

In an RPC operation, two-way messaging is used by default. The BusinessWorks client waits until it receives an answer from the adapter server before continuing processing. If the BusinessWorks client does not receive a reply from the adapter server within a certain period of time, a timeout occurs. The client cannot send a second request until the first request is processed or until a timeout occurs.

Schema Types

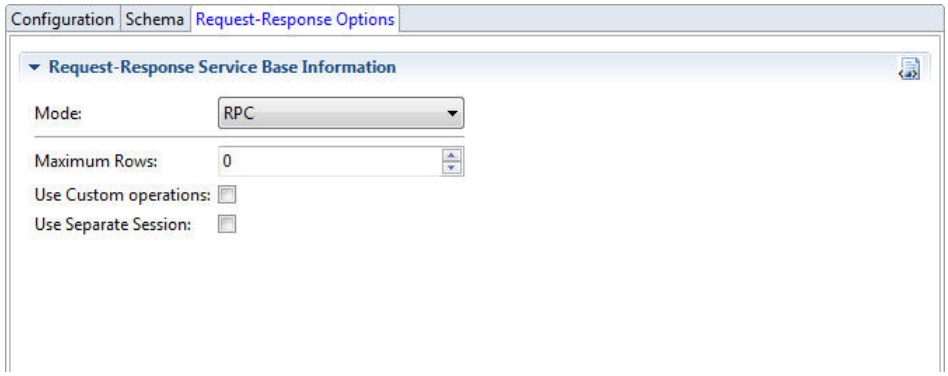
Two schema types can be used for a server object created in the repository. They are standard RPC operation and custom RPC operation.

For details, see Standard RPC Operation and Custom RPC Operation.

Standard RPC Operation

A standard RPC operation is the standard request and reply object schema predefined by TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio). This schema can be used to describe the input and output of any database request so that Request-Response Service can process database operations on any table or execute any stored procedure.

The following figure shows the configuration of the standard RPC mode:



The repository contains descriptions of the classes and operations provided for MOperation support. The SQL_OPS class describes the operations that the adapter configuration, acting as an RPC server, can handle.

The structure of the server object is:

```
<servers>
      <rvCmRpcServer
        name = "agentNamereqreprvcmRPCServer"
        session = "agentNamepubreqreprvcmRvCmSession"
        subject = "ADB.SDK.OPERATION"
        classRef = "SQL_OPS"
      />
</servers>
```

Two types of operations are supported, `SQL_EXECUTE` and `SQL_BATCHEXECUTE`. The `SQL_EXECUTE` operation takes a single SQL statement and processes it. The `SQL_BATCHEXECUTE` operation takes a sequence of SQL statements and processes them. The other classes, such as `SQL_STATEMENT`, `SQL_BIND`, and `SQL_RETURN`, describe metadata for the input and output parameters to the operations.

- Use the `SQL_BATCHEXECUTE` operation to send one or more SQL statements in a request.
- The `SQL_BATCHRETRUN` operation is the return class.
- If an error occurs while a statement is processed, the adapter returns the error immediately without processing the remaining statements.
- On Oracle databases only, all statements are executed within the same transaction.

On a Sybase database, do not use the `SQL_BATCHEXECUTE` operation to process the `ddl` statements. If you have to process the `ddl` statements, you must set the `ddl in tran` parameter to `true` in the database.

SQL_OPS Class

```
<class
  name = "SQL_OPS">
  <operation
  name = "SQL_EXECUTE"
  returnClass = "SQL_RETURN">
  <parameter name = "STATEMENT" classRef = "SQL_STATEMENT"
   direction = "in"> </parameter>
```

```
 </operation>
 <operation
 name = "SQL_BATCHEXECUTE"
 returnClass = "SQL_BATCHRETURN">
 <parameter name = "STATEMENTS" classRef =
  "sequence[SQL_STATEMENT]"
   direction = "in" />
 </operation>
</class>
```

SQL_RESULTSET Class

```
<class name = "SQL_RESULTSET">
  <attribute name = "HEADER" class = "sequence[string]">
  </attribute>
  <attribute name = "ROWVALUES" class = "sequence[SQL_ROW]">
  </attribute>
  <attribute name = "OUTBINDS" class = "sequence[SQL_BIND]">
  </attribute>
</class>
```

SQL_ROW Class

```
<class name = "SQL_ROW">
  <attribute name = "ROW" class = "sequence[any]">
  </attribute>
 </class>
```

SQL_RETURN Class

```
<object name="SQL_RETURN" lastModified="1046487158293" id="202">
  <assoc name="attribute">
    <string name="name" value="STATUS"/>
    <ref name="attributeType" value="/tibco/public/scalar/ae/string"/>
    <string name="isKey" value="false"/>
    <string name="isReadable" value="true"/>
    <string name="isWriteable" value="true"/>
  </assoc>
  <assoc name="attribute">
    <string name="name" value="SQL"/>
    <ref name="attributeType" value="/tibco/public/scalar/ae/string"/>
    <string name="isKey" value="false"/>
    <string name="isReadable" value="true"/>
    <string name="isWriteable" value="true"/>
  </assoc>
  <assoc name="attribute">
    <string name="name" value="ERROR_DESC"/>
    <ref name="attributeType" value="/tibco/public/scalar/ae/string"/>
    <string name="isKey" value="false"/>
    <string name="isReadable" value="true"/>
    <string name="isWriteable" value="true"/>
  </assoc>
  <assoc name="attribute">
    <string name="name" value="CLOSURE"/>
    <ref name="attributeType" value="/tibco/public/scalar/ae/any"/>
    <string name="isKey" value="false"/>
    <string name="isReadable" value="true"/>
    <string name="isWriteable" value="true"/>
  </assoc>
  <assoc name="attribute">
    <string name="name" value="RETURN VALUE"/>
    <ref name="attributeType"
value="/tibco/public/class/ae/ADB/adbmetadata/SQL_RETURNVALUE"/>
    <string name="isKey" value="false"/>
    <string name="isReadable" value="true"/>
    <string name="isWriteable" value="true"/>
  </assoc>
    <string name="family" value="ae"/>
    <string name="objectType" value="class"/>
</object>
```

SQL_RETURNVALUE Class

```
<object name="SQL_RETURNVALUE" lastModified="1046487158293" id="204">
  <assoc name="attribute">
    <string name="name" value="OUTBINDS"/>
    <ref name="attributeType"
value="/tibco/public/sequence/ae/class/ae/ADB/adbmetadata/sequence[SQL_BIND]"/>
    <string name="isKey" value="false"/>
    <string name="isReadable" value="true"/>
    <string name="isWriteable" value="true"/>
</assoc>
<assoc name="attribute">
    <string name="name" value="RESULTSETS"/>
    <ref name="attributeType"
value="/tibco/public/sequence/ae/class/ae/ADB/adbmetadata/sequence[SQL_RESULTSET]"/
>
    <string name="isKey" value="false"/>
    <string name="isReadable" value="true"/>
    <string name="isWriteable" value="true"/>
</assoc>
    <string name="family" value="ae"/>
    <string name="objectType" value="class"/>
</object>
```

SQL_ BATCHRETURN Class

```
<class name = "SQL_BATCHRETURN">
  <attribute name = "STATUS" class = "string">
  </attribute>
  <attribute name = "RESULTSETS" class = "sequence[SQL_RESULTSET]">
  </attribute>
  <attribute name = "SQL" class = "string">
  </attribute>
  <attribute name = "ERROR_DESC" class = "string">
  </attribute>
  <attribute name = "CLOSURE" class = "any"
  </attribute>
</class>
```

SQL_BIND Class

```
<class name = "SQL_BIND">
  <attribute name = "POSITION" class = "i4">
  </attribute>
  <attribute name = "TYPE" class = "string">
  </attribute>
  <attribute name = "DATA" class = "any">
  </attribute>
  <attribute name = "NAME" class = "string">
  </attribute>
</class>
```

SQL_STATEMENT Class

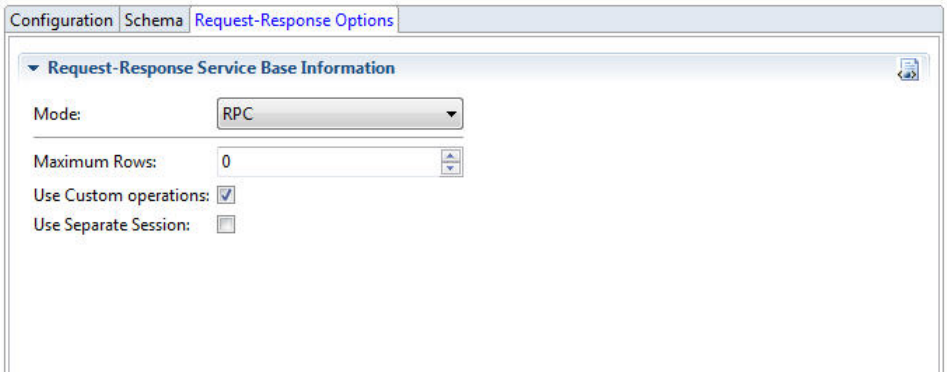
```
<class name = "SQL_STATEMENT">
  <attribute name = "SQL_STRING" class = "string">
  </attribute>
  <attribute name = "BINDS" class = "sequence[SQL_BIND]">
  </attribute>
  <attribute name = "CLOSURE" class = "any"
  </attribute>
  <attribute name = "MAXROWS" class = "i4">
  </attribute>
</class>
```

Custom RPC Operation

When you select the **Use Custom Operations** check box in the **Request-Response Options** tab, you can create a custom schema for your operations. You can define the call operations to be processed by the

RPC server in advance. TIBCO Business Studio automatically generates the input and output schema for the call operation.

The following figure shows the configuration of the custom RPC mode:



The following table describes the request schema.

Input Item	Data Type or Class Name	Description
INBINDS	INPUT_BINDS	Input parameters of the stored procedure.
OPTIONS	INPUT_OPTIONS	The input options described in the following table.
CLOSURE	Any	Closure argument. The reply returns this closure argument untouched.

The following table describes the input options.

Option	Data Type	Description
MAXROWS	Integer	The maximum number of rows to retrieve.
SQL	String	The SQL string uses to execute the stored procedure. This string is automatically generated by the palette.
CACHE	Boolean	True if users want the agent to cache the statement for performance optimization.
PACKAGE	String	Read only. Uses the call operation form to modify the package of the stored procedure.
SCHEMA	String	Read only. Uses the call operation form to modify the package of the stored procedure.

The following table describes the reply schema.

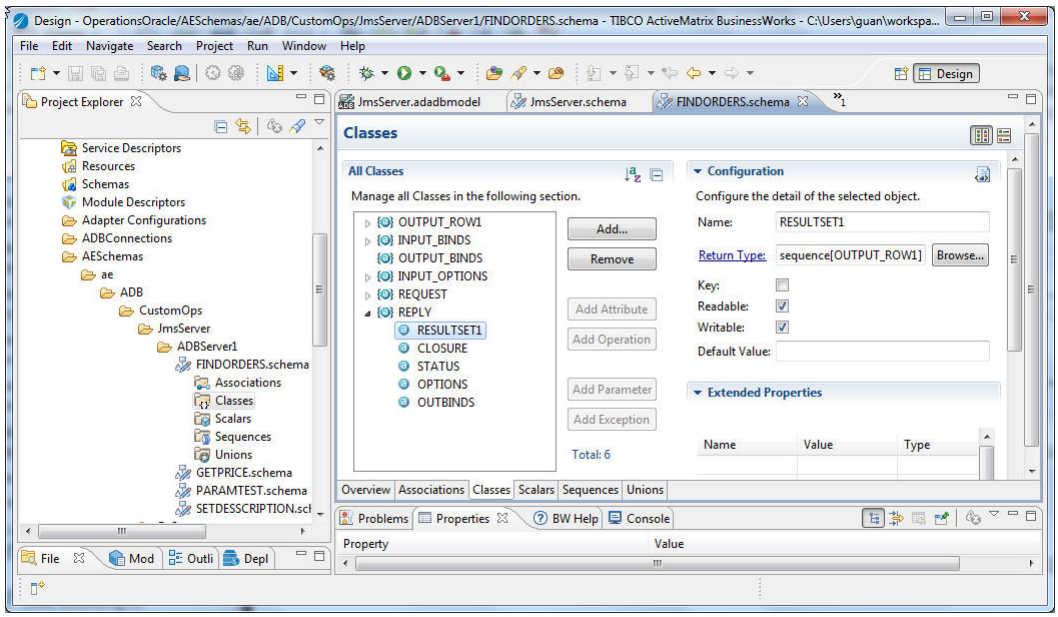
Output Item	Data Type or Class Name	Description
OUTBINDS	OUTPUT_BINDS	Output parameters of the stored procedure.

Output Item	Data Type or Class Name	Description
RESULTSET[1..n]/ RESULTSETS	OUTPUT_ROWS/ SQL_RESULTSET	The result set(s) returned by the stored procedure, see "Result Set Support" for more description.
STATUS	String	`SUCCESS` if the stored procedure is executed successfully. `FAILURE` if there is an error. Error details are stored in the OPTIONS class.
OPTIONS	CUSTOM_OP_OUTPUT_ OPTION	Contains the error description and the SQL statement if an error occurred.
CLOSURE	Any	Closure argument obtained from the request.

Result Set Support

TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) supports multiple resultsets returned for a stored procedure. The class generated for the resultset output depends on the schema information provided by the JDBC driver.

For drivers that return valid information of a resultset schema, TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) creates an output class, named `OUTPUT_ROW[1..n]`, for each result set.



If a driver does not return valid information for the resultset schema, TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) uses the generic class SQL_RESULTSETS as the output schema, as described in Standard RPC Operation.

Using Quotes in Microsoft SQL Server

When you construct a request in a client application to send to the adapter with a Microsoft SQL Server database, you must note the use of double quotation marks in quotes.

For example, the following procedure is part of a request from an application. Double quotation marks are used, which is incorrect. An error will be returned.

```
select @qry = "Update " + @tablename + " set ORDER_DESCRIPTION = 'UPDATE TEST'" + ",
ORDER_PRICE = 10109.25"
```

The following procedure is the same as the previous one, but uses single quotes. It will be correctly processed.

```
select @qry = 'Update ' + @tablename + ' set ORDER_DESCRIPTION = ''UPDATE TEST'',
ORDER_PRICE = 10109.25'
```

See *Delimited Identifiers* in your Microsoft SQL Server documentation for details.

The SQL statement must contain only ASCII characters. If your SQL statements contain non-ASCII characters, convert them into a stored procedure and invoke the procedure by using custom RPC operations.

Using the REF Data Type with Oracle Databases

The adapter supports the use of the REF data type (cursor) as an OUT parameter only in an Oracle stored procedure. The adapter returns a result in the same way as it does a result set from a SELECT query. The exact usage depends on the driver used.
When you use DataDirect Connect® (Merant) drivers, the procedure is called by leaving the REF cursor parameter out of the procedure's call list.

For example, the `ORDER_DEMO.FindOrders` procedure with an input price of $1.00 can be called through the request program with the following request:

```
{stmt={sql="{call ORDER_DEMO.FindOrders(?)}" bind={position=1
type="IN" data=1}}}
```

The REF cursor data will be returned in the results section of the reply.

Considering the following PL/SQL procedure:

```
-- Create a REF cursor procedure
CREATE OR REPLACE PACKAGE ORDER_DEMO AS
   TYPE OrderCurTyp IS REF CURSOR RETURN REPLYTEST%ROWTYPE;
   PROCEDURE FindOrders (price IN NUMBER, order_cv IN OUT
OrderCurTyp);
END ORDER_DEMO;

CREATE OR REPLACE PACKAGE BODY ORDER_DEMO AS
   PROCEDURE FindOrders (price IN NUMBER, order_cv IN OUT
OrderCurTyp) IS
   BEGIN
      OPEN order_cv FOR SELECT * FROM REPLYTEST where ID > price;
   END FindOrders;
END ORDER_DEMO;
```

Adapter Service Reference

After you create an adapter service, the adapter service configuration resources are displayed in the right panel of the **Adapter Services** tab. These resources are divided into different tabs for you to customize your adapter service.

The displayed tabs vary according to the types of services.

Publication Service Reference

To configure Publication Service, you can use the following tabs:

- Configuration Tab
- Schema Tab
- Publication Options Tab
- DB2/390 Options Tab
- DB2/AS400 Options Tab

Subscription Service Reference

To configure Subscription Service, you can use the following tabs:

- Configuration Tab
- Schema Tab
- Subscription Options Tab
- Child Table Mappings Tab
- Child Exception Table Mappings Tab
- DB2/390 Options Tab

Request-Response Service Reference

To configure Request-Response Service, you can use the following tabs:

- Configuration Tab
- Request-Response Service Schema Tab
- Request-Response Options Tab

Configuration Tab

After creating an adapter service, you can use the adapter service **Configuration** tab to specify general information. The **Configuration** tabs for Publication Service, Subscription Service, and Request-Response Service are the same.

The general information of an adapter service includes the following options:

- Service name and description
- Transport options
- Schema

Each panel provides specific configuration resources.

Service Name and Description

In the Configuration panel, you can specify a service name and description.

The following table lists the fields and descriptions in the Configuration panel.

Name	Description
Name	Name of an adapter service.
Description	(Optional) Description of the adapter service.

Transport Options

In the Transport panel, you can specify transport options. The transport options include the transport type, destination or subject, wire format, delivery mode (JMS transport only), session reference, and endpoint reference used by an adapter service.

When you add a service, the adapter user interface automatically creates a corresponding session and an endpoint, depending on the transport protocol and delivery mode being used. For details about sessions and endpoints, see TIBCO Adapter SDK documentation.

For details about the transport options, see Transport Options.

Class Reference

In the Schema panel, you can specify the class reference. The class reference indicates an Active Enterprise (AE) schema used by an adapter service. When you add a service with a schema object, the adapter user interface transforms the schema information into the AE schema. Different from a database schema, an AE schema is recognized by TIBCO Adapter SDK.

For details about the class reference, see Class Reference.

Transport Options

You can use the Transport panel to change and configure transport options.

After you create an adapter service, a corresponding endpoint and a session are displayed in the **Transport** panel. In general, you do not have to change these default settings. If you want to change the session, click the browse button next to the **Session Reference** field to open the Model Select window to either select an existing session or create a new session. For details on how to create a session, see Step 4 in Creating Publication Service.

For supported sessions and endpoints and their configuration options, see Transports Tab.

You can click the **Session Reference** or **Endpoint Reference** link to open a specific session or endpoint in the **Transports** tab.

Transport Type: JMS

If you select JMS as the transport type, JMS and its options are displayed in the Transport panel.

The following table lists the options for the JMS transport session.

Name	Description
Destination	Name on which the adapter service publishes a message to a topic or sends messages to a queue. You can specify a destination name different from the default value in this field.

Name	Description
Wire Format	Format in which messages are sent. Only the `XML Message` option is available. For details, see Guideline for Configuring the Wire Format.
Delivery Mode	(JMS Connection only) Delivery mode in which messages are sent. Regarding Publication Service, the `Non-Persistent` and `Persistent` options are available. Regarding Subscription Service and Request-Response Service, the `Non-Durable` and `Durable` options are available. For details, see Guideline for Configuring the Delivery Mode (JMS Only).

Transport Type: Rendezvous

If you select Rendezvous as the transport type, Rendezvous and its options are displayed in the Transport panel.

The following table lists the options for the Rendezvous transport session.

Name	Description
Subject	Subject name to be used by default when the adapter service publishes a message. You can specify a subject name different from the default value in this field.
Wire Format	Format in which messages are sent. The following options are available: • ActiveEnterprise Message • XML Message For details, see Guideline for Configuring the Wire Format.

Guideline for Configuring the Wire Format

The **Wire Format** option indicates the wire format in which data is sent. Services must use the same wire format to exchange data.

The wire formats for Publication Service and Subscription Service must be the same; otherwise, an error occurs.

Two types of wire formats are available:

- ActiveEnterprise Message (TIBCO Rendezvous only): ActiveEnterprise Message is an externally-described XML message format supported by TIBCO Adapter SDK. The ActiveEnterprise standard wire format provides class information and packing rules for the TIBCO Adapter SDK set of data types. TIBCO ActiveEnterprise components use this format to perform extra validation on sent or received messages. Control information for validation is sent in a message. For details, see *TIBCO Adapter SDK Programmer's Guide*.

- XML Message: XML Message conforms to specifically constructed and fully compliant XML Schema (XSD) based on the existing definition of the Active enterprise schema.

Guideline for Configuring the Delivery Mode (JMS Only)

Delivery mode indicates the mode in which messages are delivered. It is only supported for JMS connections.

For a JMS transport session, delivery modes vary according to different service types:

- Regarding Publication Service, the following delivery modes are supported:
  - **Persistent**: in general, a message marked persistent is available to a JMS client even if the EMS server goes down. Persistent messages are held in secondary storage in the server and have guaranteed delivery when they are sent to a topic that has durable subscribers.
  - **Non-persistent**: a message marked non-persistent is not available to a JMS client if the EMS server goes down. These messages are never written to persistent storage.
- For Subscription Service and Request-Response Service, the `Delivery Mode` option is available only when the `Connection Factory Type` is `Topic`. For details about connection factory types, see "JMS Sessions" in Session Reference.

  The following delivery modes are supported:

  - **Durable**: The JMS server stores messages sent to the service even when the service is down. When the service recovers, the service receives the stored messages.

    If a topic has no durable subscribers, no subscribers require messages to be resent in the event of a server failure and therefore messages do not have to be saved. Performance is improved because disk I/O is not required.
  - **Non-durable**: The service is not registered with the JMS server. The JMS server does not hold messages sent to a non-durable service when the service is down.

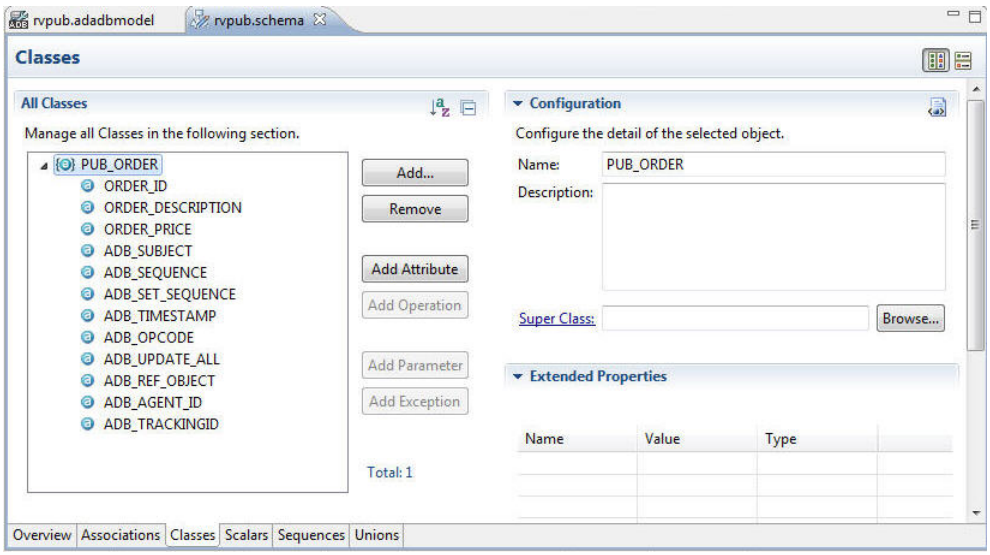
For more information about delivery modes, see *TIBCO Enterprise Message Service User's Guide*.

Class Reference

The Schema panel displays the associated AE schema for an adapter service.

To open the specific AESchemas file, click the **Class Reference** link. The file is opened in the editor with the **Classes** tab displayed.

The following figure shows the **Classes** tab.



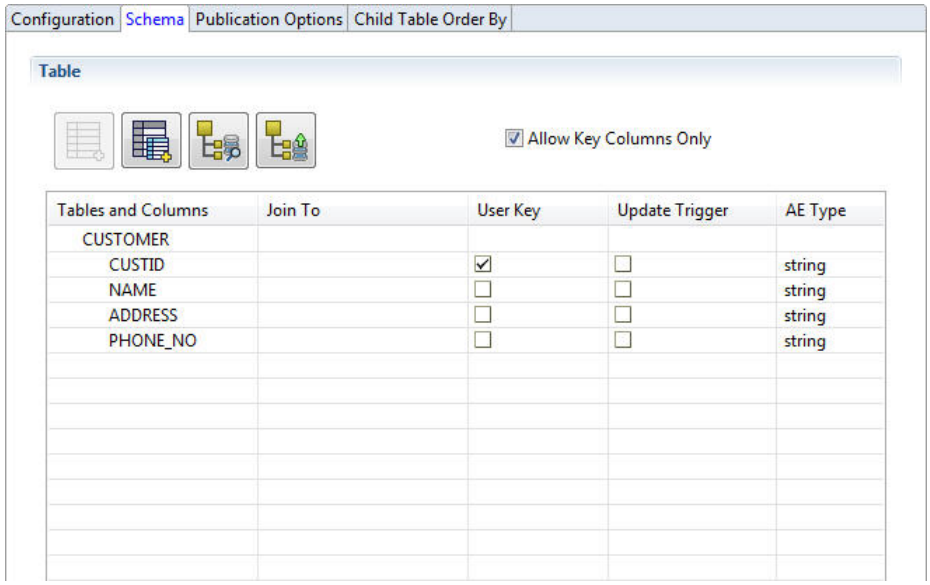
Different from a database schema, an AE schema is recognized by TIBCO Adapter SDK. For more information, see "Difference Between Database Schema and TIBCO ActiveEnterprise Schema" in Working with Schemas.

Schema Tab

In Publication Service, you can use the **Schema** tab to select tables, add child tables, refresh table information, and load tables.

Added tables are displayed in the **Table** panel.

The following figure shows the **Schema** tab of Publication Service:



The **Schema** tab includes the following tool icons:

- **Add Table** : click this icon to add a root table. In Publication Service, this table is the source table.
- **Add Child table** : click this icon to add a secondary table to the source table.
- **Re-find schema from database** : click this icon to refresh stored table information in TIBCO Business Studio by retrieving new information from the database.
- **Load table schema from database** : click this icon to load a database table schema and convert it into an AE schema, click this icon.
- **Delete Table**: to delete a table, right-click the table, and from the pop-up menu, click **Delete Table**. If the table has child tables, its child tables are removed as well.

The following table lists the fields in the Table view of an AE schema:

Name	Description
Allow Key Columns Only	Use this check box to determine whether child columns are joined to a column that is a non-key column. By default, this check box is selected. The status is as follows: • Selected: child columns are joined only to a key column. • Cleared: child columns can be joined to any columns, including a non-key column.
Tables and Columns	Loaded tables and columns.
Join To	Name of a parent table column to which a child table column is joined for parent-child relationships.
User Key	Click this check box to define a column as a user key.
Update Trigger	Click this check box to fire a trigger when an UPDATE statement changes a value in a column.  Only the following databases are supported in the Update Trigger option: • Oracle • Microsoft SQL Server • DB2 UDB, AS400, or OS390 • Sybase
AE Type	Primitive type mapped to the TIBCO ActiveEnterprise (AE) type.

Publication Options Tab

In Publication Service, you can use the **Publication Options** tab to select a polling method, select a storage mode, specify a publishing table, and so on.

The following table lists and describes the fields, which are grouped in two panels:

Field	Description
Publication Service Base Information Panel	
Polling Method	Publication Service uses the following two methods to monitor changes to a database table: • **Timer** (default setting): used to retrieve records from the database in a specified time. This specified time can be set in the **Polling Interval** field. • **Alerter** (Only available when using Oracle and Microsoft SQL Server databases): used to asynchronously notify running Publication Service of database changes. Use the alerter method only when database changes are infrequent.

Field	Description
Polling Interval	Specifies how often the adapter checks the publishing table for new rows, in milliseconds. The default value is `5000` ms, or once every five seconds.  If you select **Alerter** from the **Polling Method** list, the **Polling Interval** field is not displayed.
Use Polling Batch Size	Select this check box to determine whether you want to poll rows in a batch or not. By default, the **Use Polling Batch Size** check box is selected.
Polling Batch Size(Maximum Rows)	This field is displayed when the **Use Polling Batch Size** check box is selected. This field specifies the maximum number of messages that are retrieved at each polling interval. The adapter returns to the event loop when the adapter finishes sending messages. Using this option helps process events in an efficient manner. For example, when polling a large number of rows, the adapter works best if a fixed number of rows is specified in this field. If the **Use Polling Batch Size** check box is selected, the default value of the **Polling Batch Size(Maximum Rows)** field is `1000`. In the previous release, the **Use Polling Batch Size** check box is not selected by default. The default value of the **Polling Batch Size (Maximum Rows)** field is `0`, which indicates that all new rows are fetched. If you use the default value, when the adapter receives a large number of messages from a database within a short period, in which the receiving speed exceeds the processing speed of the adapter, an OutOfMemory error might occur. To resolve this issue, the default value is changed to `1000` in this release. Even within a short time, the adapter fetches only 1000 rows, which contributes to a reasonable performance of the adapter.  When you use the **Polling Batch Size(Maximum Rows)** option, note the following conditions: • When you use the default polling interval, a value specified in the **Polling Batch Size(Maximum Rows)** field might affect the performance of the adapter. For example, if you set the value to `5`, the adapter fetches 5 rows within 5 seconds. At this rate, the adapter can fetch 60 rows within 1 minute. If you set the value to `100`, the adapter fetches 100 rows within 5 seconds. At this rate, the adapter can fetch 1200 rows within 1 minute. In general, the adapter can handle hundreds of or even thousands of rows within one second mainly depending on your machine performance. If the value of the **Polling Batch Size(Maximum Rows)** field is set excessively small, the performance of the adapter might deteriorate. • When you use the **Polling Batch Size(Maximum Rows)** and **Enable Group Messaging** options together in Publication Service, it is good practice to set the value of the **Polling Batch Size(Maximum Rows)** field to the value greater than or equal to the value of the **Group Size** field to improve the performance. If the value of the **Polling Batch Size(Maximum Rows)** field is set to be smaller than the value of the **Group Size** field, the group cannot be fully filled.

Field	Description
Batch Publish Status Updates	Available only if the **Use Polling Batch Size** check box is selected. Select this check box to optimize the publishing performance by batching message status updates to the publishing table. If an adapter configuration stops before a batch update is performed, the status column is not updated. As a result, duplicate messages might be published when the adapter configuration is restarted.  Do not use this option when messages are published using a parameterized subject name.
Publisher Batch Confirm Size	Specifies the number of messages that require their status to be updated in a single batch. The default value is `0`. This field is used only for the TIBCO Rendezvous Certified transport type. Do not use this option when messages are published using a parameterized subject name. Entering a value in this field optimizes performance. However, if an adapter configuration stops before a batch update is performed, the status column is not updated. As a result, messages that were successfully published might still have a status of P (pending) in the publishing table when the adapter is restarted. In this case, the ledger file contains the correct status information. Smaller values in this field decrease this risk.
Publisher Batch Confirm Timeout	Specifies the number of milliseconds to wait before updating the status column. After this interval, an update is performed even if the batch size value is not reached. The default value is `10,000` (10 seconds). A value of `0` means that no timeout interval is used. This field is used only for the TIBCO Rendezvous Certified transport type. Do not use this option when messages are published using a parameterized subject name. If an adapter configuration stops before a batch update is performed, the status column is not updated. As a result, messages that were successfully published might still have a status of P (pending) in the publishing table when the adapter is restarted. In this case, the ledger file contains the correct status information. Smaller timeout values decrease this risk.
Publication Service Option Information Panel	
Storage Mode	For each service, you must specify a storage mode: • **Publish by Value**: copies all specified columns in the source table to the publishing table. • **Publish by Reference**: copies only key column values to the publishing table. If no key column is defined in the database, a substitute non-key column must be defined to publish by reference. For more information, see Selecting a Storage Mode.

Field	Description
Publishing Table	Name of the database table that is used to store a copy of data to be published. The table name can be qualified with the `schema.table_name` format. The publishing table cannot contain any user-created columns in which the column name starts with ADB_. These characters are reserved for use by the adapter. For better results use the publishing table name prefixed by P_. For example, if your source table is MY_ORDER, its publishing table is named P_MY_ORDER. A publishing table name must be less than 64 characters.
Referred Object	Type the name of a view or a different database object in this field to select source data. This field is displayed when you select **Publish by Reference** from the **Storage Mode** list. If you want to select source data from a list of tables in the current user schema, click **Add**. If you want to select source data from a different schema, click **Add from**.
Update Mode	Select either of the following two methods to update tables: • **Update**: updates a row in the destination table only when the row exists. • **Upsert**: updates a row in the destination table if the row exists. If no such row exists, it performs an insert.
Enable Loop Detection	Select this check box to enable loop detection and prevent an infinite loop from occurring when the publishing table and destination table are the same table. Loop detection is disabled for DB2 on z/OS.
Do Not Generate Triggers	Select this check box to prevent the automatic generation of triggers. If this check box is selected, you have to generate triggers manually. It is good practice not to select this check box, but using this option helps manually manage the insertion of data into the publishing table.
Enable Group Messaging	Select this check box to enable group messaging.
Group Size	Available only if the **Enable Group Messaging** check box is selected. Specifies the number of rows to be published in a single message. You can use a module property to set a value for this field.
Enable Load Balancing	Select this check box to enable load balancing so that multiple Publisher endpoints can poll and publish changes of the same source (publishing) table. Set a batch polling size before selecting this option.  When you enable load balancing for Publication Service, the added table, automatically created publishing table, mutex name, and message subject name must be the same.
Mutex Name	Available only if the **Enable Load Balancing** check box is selected. Specifies the name for the mutex table.

Field	Description
Number of Publication Service Threads	These threads are allocated on demand. For example, if the **Number of Publication Service Threads** field is set to 1, a publication thread and a database connection are created automatically. With this number set, each Publication Service service holds a specified number of threads that connect to the database separately and process requests in parallel. For related information, see "Multithreading in Publication Service" in Multithreading.
Publish Child Data	This check box is selected by default. When this check box is selected, the parent row and all related child rows are published when a parent row is updated. Upon receipt of such a message, Subscription Service updates the parent row and then updates all the child rows with the data in a received message. Any change to the child tables without a change in the parent table will not be processed. The adapter monitors only the parent table for publishing. The adapter updates the child rows by deleting all the related child rows, and then inserts child rows again based on the data in the received message. For information on how to add related tables for Publication Service, see Adding Child Tables in Publication Service. For information on how to add related tables for Subscription Service, see Adding Child Tables in Subscription Service.
Polling Commit for DB2	Select the **Polling Commit for DB2** check box to enable Publication Service to perform a commit operation after selecting a query for DB2. This check box is displayed when you select a DB2 database as a vendor, including DB2 OS390, DB2 AS400, and DB2 UDB.

DB2/390 Options Tab

The **DB2/390 Options** tab is displayed if you select **DB2 OS390** from the **Vendor** list in the **Configuration** tab of an adapter configuration.

When the DB2 load utility loads rows to the source table, it does not activate the INSERT triggers of the table. The loaded data is not published.

The following table lists and describes the fields in the **DB2/390 Options** tab of Publication Service:

Field	Description
Database Name	Name of the database in which you want to put your publishing table.
Table Space Name	Name of the table space in which the publishing table is located.
Storage Group (optional)	Designator of the storage group that holds the publishing table indexes.
Buffer Pool (optional)	Name of the buffer pool to be used for indexes.
Index Suffix	Suffix that the adapter appends to each of the indexes (IDX1_, IDX2_, and IDX3_). The maximum value is 13 characters.

Field	Description
Trigger Suffix	Suffix that the adapter appends to each of the triggers (T1, T2, and T3). The maximum value is 5 characters.

DB2/AS400 Options Tab

The **DB2/AS400 Options** tab is displayed if you select **DB2 AS400** from the **Vendor** list in the **Configuration** tab of an adapter configuration.

You can specify a trigger in the **Trigger Option** field to copy new data from the source table to the publishing table. The following trigger options are available:

- **SQL**: the trigger program is defined entirely by using SQL. The SQL trigger can be an insert, update, or a delete trigger. When the adapter copies data from the source table to the publishing table, the prompt is not returned until all data is written. Currently only the SQL trigger can be used for a DB2 AS400 database.
- **Synchronous** (not applicable): the trigger is written in RPG. When the adapter copies data from the source table to the publishing table, the prompt is not returned until all data is written.
- **Asynchronous** (not applicable): when the adapter copies data from the source table, the data is inserted into a data queue and then inserted into the publishing table asynchronously. The prompt is not blocked, so the adapter continues working when the data is inserted into the publishing table.

Schema Tab

In Subscription Service, you can use the **Schema** tab to select tables, add child tables, and refresh table information. You must set subscriber table options before configuring other subscriber options.

An incoming message does not have to contain data for all the columns defined in the subscriber table. An adapter can be configured to expect only a subset of the columns. The adapter checks the repository for attributes defined in the class object definition of the subscriber table. When a message arrives, the adapter iterates through the attributes in the class object definition and looks for those same attributes in the incoming message.

Subscriber table names can be qualified with a schema name, such as SCOTT.EMP. To access tables in other schemas, the database user defined in the adapter **Configuration** tab must have the proper set of permissions granted. For more information, see *TIBCO ActiveMatrix Adapter for Database Concepts*.

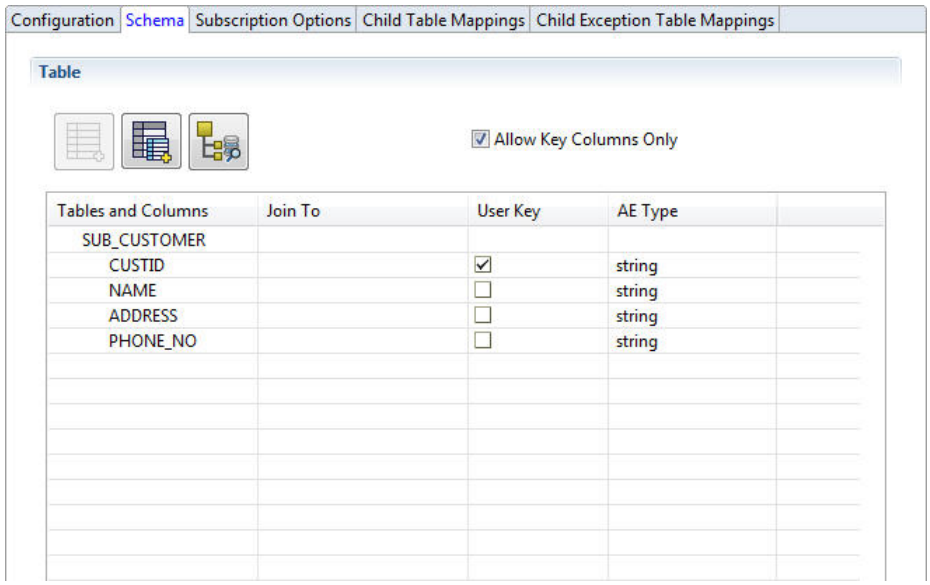
When you configure the destination table, you only subscribe to columns that can be updated. If you subscribe to a column that cannot be updated and a message arrives with no data for that column, a null value will be written to that column.

For example:

- If a source table is configured to send data for columns, c1, c2, and c3, and the destination table is configured to receive data for columns c1, c2, c3, c4, and c5, columns c4 and c5 will be ignored and take on whatever default values they are supposed to have, regardless of the TIBCO ActiveEnterprise message or XML message wire format.
- If a source table is configured to send data for columns, c1, c2, and c3, and the destination table is configured to receive data for columns c1, c2, and c3 but not configured to receive data for columns c4 and c5, columns c4 and c5 will retain the defaults applicable to both tables.

When the publishing table is configured to use parent-child relationships, the subscriber adapter must use the same repository as the publisher adapter.

The following table shows the **Schema** tab of Subscription Service:



The **Schema** tab includes the following tool icons:

- **Add Table** : click this icon to add a root table. In Subscription Service, this table is the destination table, which is used to receive data.
- **Add Child table** : click this icon to add a secondary table to the destination table.
- **Re-find schema from database** : click this icon to refresh stored table information in TIBCO Business Studio by retrieving new information from the database.
- **Delete Table**: to delete a table, right-click the table, and from the pop-up menu, click **Delete Table**. If the table has child tables, its child tables are removed as well.

The following table lists the fields in the Table view of an AE schema:

Name	Description
Allow Key Columns Only	Use this check box to determine whether child columns are joined to a column that is a non-key column. By default, this check box is selected. The status is as follows: • Selected: child columns are joined only to a key column. • Cleared: child columns can be joined to any columns, including a non-key column.
Tables and Columns	Loaded tables and columns.
Join To	Name of a parent table column to which a child table column is joined for parent-child relationships.
User Key	Click this check box to define a column as a user key.
AE Type	Primitive type mapped to the TIBCO ActiveEnterprise (AE) type.

Subscription Options Tab

In Subscription Service, you can use the **Subscription Options** tab to specify an exceptions table and an opaque exceptions table and perform other configurations. The adapter uses the exceptions table to store error messages from database operation failures.

The following table lists and describes the fields, which are grouped in two panels.

Name	Description
Subscription Service Base Information Panel	
Use Separate Session	The **Use Separate Session** check box is used with Subscription Service multithreading. When you select this check box, the following options are displayed: • **useSerial** check box • **Number of Subscription Service Threads** field • **Batch Commit Size** field • **Batch Commit Timeout(milliseconds)** field After you set a value in the **Number of Subscription Service Threads** field, the multiple threads can share the same session and dispatcher. By default, the **Use Separate Session** check box is not selected.  If you want to use multiple threads by configuring a separate session, the transport session name must end with one or more digits from `0` to `9`.
useSerial	This check box is displayed when the **Use Separate Session** check box is selected. This option is designed to ensure that all messages are processed in order.  Note the following conditions about this option: • The primary key in the destination table cannot be null. • If the destination table has no primary key, you have to select the **User Key** check box instead of the primary key. When processing the application requests, messages are handled in order. • If the destination table has no primary key, and the **User Key** check box is cleared, it is good practice to clear the **useSerial** check box.

Name	Description
Number of Subscription Service Threads	This field is displayed when the **Use Separate Session** check box is selected. Threads are allocated on demand. The valid values range from `1` through `n`. The number of threads you set in Subscription Service indicates the number of subscription threads that will connect to the database. For more information, see Multithreading and Session in Subscription Service.
Subscription Service Option Information Panel	
Exceptions Table	Name of the exceptions table to which data is written if the adapter cannot write the data to the destination table. This table holds messages that caused an exception. If a table does not exist, Subscription Service creates one such table. For details, see Exceptions Table. The exceptions table cannot contain any user-created columns in which the column name starts with ADB_. These characters are reserved for use by the adapter.  For a parent-child relationship, this is the parent exceptions table. If you do not set the parent exceptions table, the child exceptions table is invalid.
Use Opaque Exceptions Table	Select this check box to use an opaque exceptions table. The table records each message entirely into a column, along with the error message. A message is logged into the exceptions table if Subscription Service fails to generate records in the destination table or the adapter fails to insert a message into an exceptions table. For details, see Opaque Exceptions Table. For DB2/390 databases, you must create a LOB table space before using the opaque exceptions table, which uses the LOB table space.
Opaque Exceptions Table	Name of the opaque exceptions table. For details, see Opaque Exceptions Table.
Pre Commit Stored Procedure	Name of a stored procedure Subscription Service calls after the database insert, update, or delete operation and before the commit.

Name	Description
Reply Sender Quality of Service	If Subscription Service must send a reply to the sender, this value identifies the quality of service or delivery mode to be used to send the reply. The available options are: • Reliable (TIBCO Rendezvous transport type only) • Certified (TIBCO Rendezvous transport type only) • Persistent (JMS transport type only) • Non-Persistent (JMS transport type only)
Bulk Insert Size	When configured for batch processing, the adapter stores incoming subscription requests in a batch. All incoming messages to be inserted are stored until this size is reached. Then, a bulk insert operation is performed on the destination table by using a single processing. This number must be less than or equal to the value in **Batch Commit Size**. The default value is `1`. The batch is emptied when the batch size is reached.  When using a parent-child relationship, you must note the following conditions: • Bulk Insert is valid if you enable group messaging and only send the data to the parent table. • Bulk Insert is invalid if you send the data to the parent table and child table. • Increasing Bulk Insert Size does not significantly improve the Subscription Service performance. • If you want to use JDBC driver options to improve the Subscription Service performance, for better results set the `batchPerformanceWorkaround=true` property instead of the `EnableBulkLoad` and `BulkLoadBatchSize` options.

Name	Description
Batch Commit Size	The number of messages to be batched before a commit operation is invoked. The default value is `0`. This field is displayed when the **Use Separate Session** check box is selected. Note the following conditions regarding the **Batch Commit Size** field: • If you use the default value of `0` in the **Batch Commit Size** field, the adapter takes the value `1` as the batch commit size when running Subscription Service. • The **Batch Commit** option is not supported with the RVCMQ Quality of Service type. The RVCMQ scheduler requires a message to be confirmed before dispatching the next message, which prevents the adapter from operating in batch mode. For more information about batch commit, see Configuring Batch Commit for Subscription Service.
Batch Commit Timeout(milliseconds)	Specifies the longest interval between two batch commits. When the specified timeout expires, Subscription Service commits all the received messages even if the number does not reach the **Batch Commit Size** value. The default value is `10,000` milliseconds. This field is displayed when the **Use Separate Session** check box is selected. The batch commit feature does not commit all received messages if the adapter configuration terminates before the batch commit value or time-out value is met. If RVCMQ is enabled, when the batch commit size is greater than 1, batch commit times out after each operation (insert, update, or delete).  When you use the default session for Subscription Service, if you set the **Batch Commit Size** and **Batch Commit Timeout(milliseconds)** fields in TIBCO Business Studio and then upgrade the adapter from an earlier version to the current version, the configurations of these fields in TIBCO Business Studio are lost and you have to reset the corresponding properties in the `adbagent.tra` file. For more information about batch commit, see Configuring Batch Commit for Subscription Service.
Rendezvous Maximum Queue Size	Maximum number of messages that can be put in the TIBCO Rendezvous event queue. The default value is `0`, which indicates that no limit is placed on the event queue size. Use this option to prevent the memory of a subscriber endpoint from overflowing if the publisher endpoint is too fast.  If the value of this field is set to be smaller than the actual number of messages in the event queue, only the allowable number of messages can be sent and the subsequent messages are lost.

At startup, the Subscription Service dispatchers wait for all components to start before dispatching subscriber messages.

Child Table Mappings Tab

In Subscription Service, you can use the **Child Table Mappings** tab to map a child table of Publication Service with a child table of Subscription Service if the tables do not have the same name.

These options are available only when child tables are specified in the **Schema** tab of Subscription Service.

You can specify values for the following fields in the **Child Table Mappings** tab.

Name	Description
Subscriber Child Table Name	Displays the child tables that can be mapped. Entries are automatically created when you add child tables for Subscription Service.
Publisher Child Table Name	Click this column and type the name of a publisher child table that corresponds to the subscriber child table name. Prefix the table name with a database user if your database requires it.

For information on how to map child tables, see Setting Up the Mapping Relationship Between Child Tables.

Child Exception Table Mappings Tab

In Subscription Service, you can use the **Child Exception Table Mappings** tab to configure each child table to use an exceptions table.

To make the configurations of the **Child Exception Table Mappings** tab effective, you have to not only set the fields in the **Child Exception Table Mappings** tab, but also set the **Exceptions Table** field in the **Subscription Options** tab.

The following table lists the columns in the **Child Exception Table Mappings** tab.

Name	Description
Subscriber Child Table Name	Child tables that can be mapped. Entries are automatically created when you add child tables for Subscription Service.
Child Exception Table Name	Click this column and type the name of a child exception table.  The child exception table is invalid if you do not set the parent exception table.

DB2/390 Options Tab

The **DB2/390 Options** tab is displayed if you select **DB2 OS390** from the **Vendor** list in the **Configuration** tab of an adapter configuration.

The following table lists and describes the fields in the **DB2/390 Options** tab of Subscription Service:

Field	Description
Database Name	Name of the database in which you want to put the destination table.

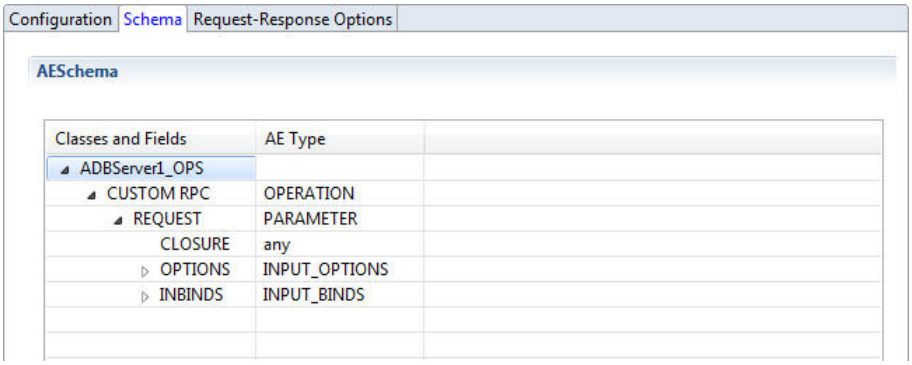
Field	Description
Table Space Name	Name of the table space in which the destination table is located.
LOB Table Space Name	Name of the Large Objects (LOB) table space in which the auxiliary table of the opaque exceptions table is located.

Schema Tab

In Request-Response Service, you can use the **Schema** tab to view stored procedures, functions, packages, or other information. The information displayed in this tab varies depending on different types of Request-Response Service.

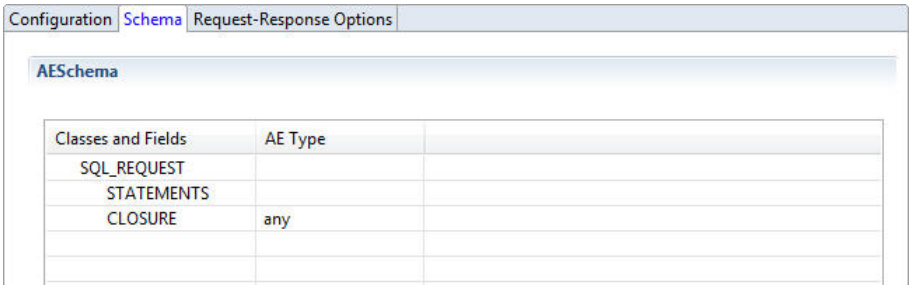
Custom RPC Service

In custom RPC Service, the **Schema** tab displays the stored procedure, function, and package information. The following figure shows the **Schema** tab of custom RPC Service:



Request-Reply RPC Service

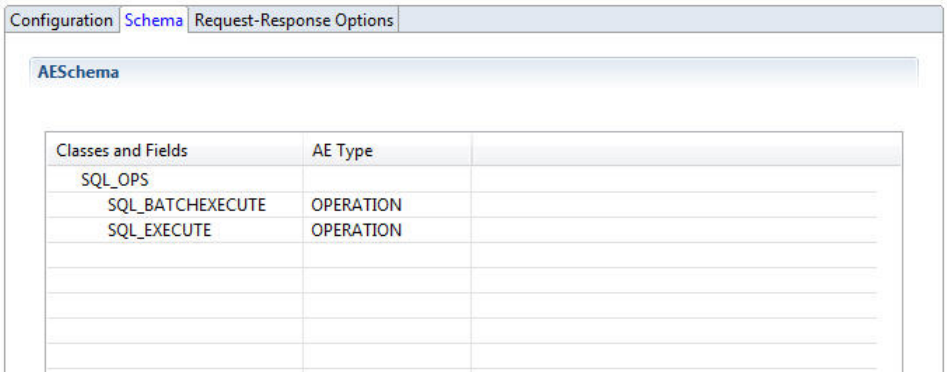
In Request-Reply RPC Service, the **Schema** tab displays the SQL_REQUEST class information. The following figure shows the **Schema** tab of Request-Reply RPC Service:



For details about the SQL_REQUEST class, see "TIBCO ActiveEnterprise or XML Message Request Format" in Requests.

Standard RPC Service

In standard RPC Service, the **Schema** tab displays the SQL_OPS class information. The following figure shows the **Schema** tab of standard RPC Service:



For details about the SQL_OPS class, see "SQL_OPS Class" in Standard RPC Operation.

The following table lists the columns in the AESchema table:

Field	Description
Classes and Fields	Loaded classes and fields.
AE Type	Primitive type of the AE schema class or field.

Request-Response Options Tab

In Request-Response Service, you can use the **Request-Response Options** tab to set different modes of RPC Service.

The following table lists and describes the fields in the **Request-Response Options** tab:

Field	Description
Request-Response Service Base Information Panel	
Mode	Specifies an endpoint to be a Request Reply server or Remote Procedural Call (RPC) server. The value can be: • Request Reply: A subscriber endpoint is created. This endpoint listens to a request and publishes a reply to the reply subject. • RPC: A server endpoint is created. This endpoint receives a request from and sends back a reply to the client.

Field	Description
Maximum Rows	Specifies the maximum number of rows to be fetched in the service level. This can be used to limit the memory usage of the adapter. The unfetched rows are ignored by the adapter.  You can also set this field in the operation level when using custom RPC Service, and the **Maximum Rows** value set in the operation level takes higher precedence over the value set in the service level. For details on how to perform the configuration in the operation level, see Setting Maximum Rows in the Operation Level. In the log file, only the maximum number of rows in the service level is printed.
Use Custom operations	This check box is displayed if you select **RPC** from the **Mode** list. Select this check box and fetch the stored procedure by following the steps in Fetching Stored Procedures.
Use Separate Session	Select this check box to create a new session, and move the service endpoint to this session. This feature is used with Request-Response Service multithreading. When you select this check box, the **Number of Request-Response Service Threads** field is displayed. After you set the value in this field, the multiple threads can share the same session and dispatcher. See Multithreading and Session in Request-Response Service for details.  If you want to use multiple threads by configuring a separate session, the transport session name must end with one or more digits from `0` to `9`.
Number of Request-Response Service Threads	Indicates the number of threads the adapter service uses to connect to the database. The valid values are from `1` through `n`. Each thread has a separate connection to the database. By specifying multiple threads, you can balance the incoming RPC request load. See Multithreading and Session in Request-Response Service for details.
Reply Subject	(Request-Reply mode only) Specifies a subject name that the adapter uses to respond to a request. If no response, a subject is specified in the request message. The subject name in the **Reply Subject** field cannot be the same as the name in the **Destination** field.

Working with Parent-Child Table Relationships

Data models typically contain tables that share column data through a relationship. You can configure a publishing table to include related data from another table for publication. Data from the related table is not copied to the publishing table, but is fetched by reference.
When rows are inserted into the publishing table, a message that includes the data from the source table and related (child) table is published. Accordingly, a table with the same columns as the child table associated with the publishing table must be specified for Subscription Service.

Adding child tables for Publication Service and Subscription Service requires separate procedures. In Publication Service, you can add child tables to the source table; in Subscription Service, you can add child tables to the destination table.

To work with parent-child relationships, complete the following tasks:

1. Adding Child Tables in Publication Service
2. Adding Child Tables in Subscription Service
3. Mapping Child Tables in Publication Service and Subscription Service

Restrictions

The database schema must be the same for all tables, but the table names can be different. If the child table associated with the publishing table and the child table associated with the destination table have different names, you must set a mapping between the child tables.

The following restrictions apply to parent and child tables:

- The child table in the source database and child table in the destination database must have the same columns.
- When parent-child relationships are defined, a subscriber adapter must use the same repository as the publisher adapter.
- When working with parent-child table relationships, it is good practice to set the **ADB_OPCODE** values in the parent table and child table to the same value. Or only set the value of the **ADB_OPCODE** column in the parent table and leave the child table empty.

Adding Child Tables in Publication Service

To work with parent-child relationships, you have to first add child tables in Publication Service.

When you add a child table, TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) creates a class object in the repository for the child table and an association object for the parent-child relationship.

Prerequisites

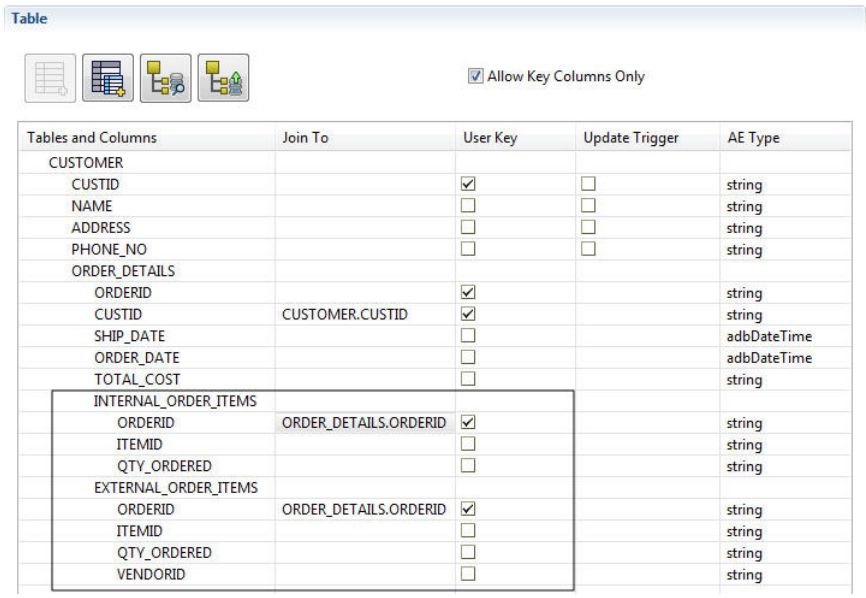
A source table named CUSTOMER is added in Publication Service. For details on how to add a table in the **Schema** tab for Publication Service, see Fetching a Table After Creating Publication Service.

Procedure

1. Click the **Schema** tab. Select the CUSTOMER table and click **Add Child table**  .
2. In the "Table name Pattern" window, specify another search criterion, such as `ORDER_DETAILS%`.
3. In the "Select table" window, select **ORDER_DETAILS** from the table list and click **Finish**.
A child table ORDER_DETAILS is added to the source table CUSTOMER.

4. Optional: If you want to add a table from another schema, specify the schema name in the **Other Schema** field, click **Add From Other Schema**, and select a table you want.
   To reference an external schema, the default schema must have the proper access privileges. For information on how to grant access privileges to an external schema, see Referencing an External Schema.
5. Optional: If you click **Next** in Step 3, you can edit the selected table by selecting or clearing the **Use** check box next to a table column.
   If the **Use** check box next to a table column is cleared, the table column is not used. By default, all columns of the table are selected.
6. Expand the ORDER_DETAILS table and add a join between the CUSTOMER table and ORDER_DETAILS table.
   a) In the ORDER_DETAILS CUSTID entry, select the **User Key** check box and then click the **Join To** list.
   b) Select **CUSTOMER.CUST_ID** from the **Join To** list and click **Save** on the toolbar.
   The source table and the child table are joined together.

7. To add child tables to the ORDER_DETAILS table, repeat Step 3 to Step 6.
   Two child tables, INTERNAL_ORDER_ITEMS and EXTERNAL_ORDER_ITEMS, are added and joined to the ORDER_DETAILS table.



Adding Child Tables in Subscription Service

After you add child tables in Publication Service, you can use the same way to add tables in Subscription Service. After that, you can map the tables in Subscription Service with the tables in Publication Service.

When you add a child table, TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) creates a class object in the repository for the child table and an association object for the parent-child relationship.

Prerequisites

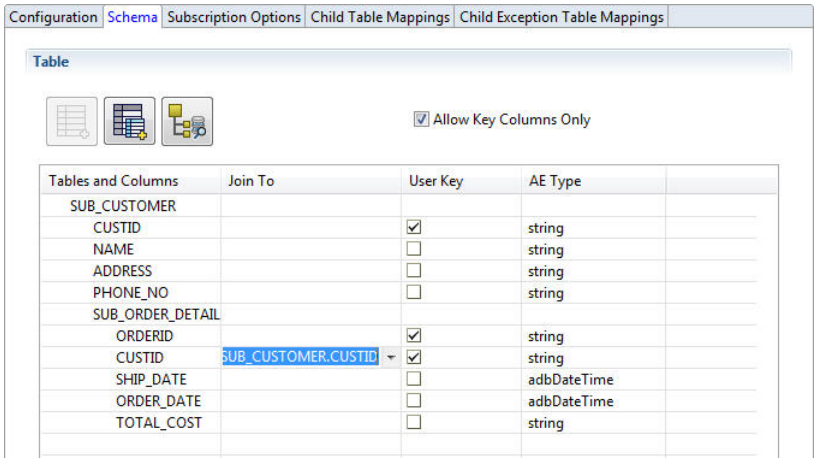
A destination table named SUB_CUSTOMER is added in Subscription Service. For details on how to add a table in the **Schema** tab for Subscription Service, see Fetching a Table After Creating Subscription Service.

Procedure

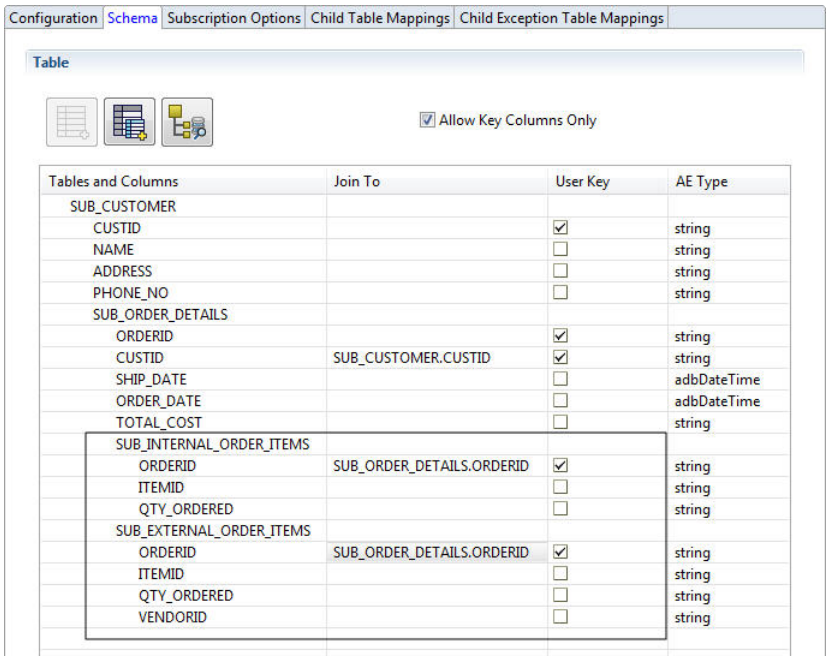
1. In the **Schema** tab, select the SUB_CUSTOMER table and click **Add Child table**  .
2. In the "Table name Pattern" window, specify another search criterion, such as `SUB_ORDER_DETAILS%`.
3. In the "Select table" window, select **SUB_ORDER_DETAILS** from the table list and click **Finish**.
   A child table SUB_ORDER_DETAILS is added to the destination table SUB_CUSTOMER.
4. Optional: If you click **Next** in Step 3, you can edit the selected table by selecting or clearing the **Use** check box next to a table column.

   If the **Use** check box next to a table column is cleared, the table column is not used. By default, all columns of the table are selected.
5. Expand the SUB_ORDER_DETAILS table and add a join between the SUB_CUSTOMER table and SUB_ORDER_DETAILS table.
   a) In the SUB_ORDER_DETAILS CUSTID entry, select the **User Key** check box and then click the **Join To** list.
   b) Select **SUB_CUSTOMER.CUST_ID** from the **Join To** list and click **Save** on the toolbar.

   The destination table and the child table are joined together.



6. To add child tables to the SUB_ORDER_DETAILS table, repeat Step 1 to Step 5.
   Two child tables, SUB_INTERNAL_ORDER_ITEMS and SUB_EXTERNAL_ORDER_ITEMS, are added and jointed to the SUB_ORDER_DETAILS table.



Mapping Child Tables in Publication Service and Subscription Service

Child tables in Publication Service and Subscription Service must be mapped unless the child tables have the same names. After mapping, data can be transferred between Publication Service and Subscription Service.

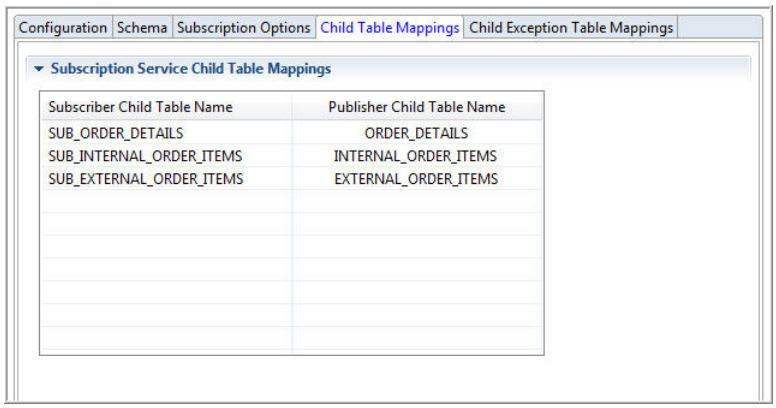
Procedure

1. In Subscription Service, click the **Child Table Mappings** tab.

   In the **Child Table Mappings** tab, the **Subscriber Child Table Name** column displays the child tables that can be mapped. Entries are automatically created when you add child tables for Subscription Service.

2. Click the **Publisher Child Table Name** column and enter the following values:

   - For the SUB_ORDER_DETAILS table, type `ORDER_DETAILS`.
   - For the SUB_INTERNAL ORDER_ITEMS table, type `INTERNAL_ORDER_ITEMS`.
   - For the SUB_EXTERNAL_ORDER_ITEMS table, type `EXTERNAL_ORDER_ITEMS`.



3. Click **Save** on the toolbar.

Working with Schema Browser

TIBCO adapters use schemas to manage data that is retrieved from and exported to a source or target application. You can use the Schema Browser view provided by TIBCO ActiveMatrix Adapter Framework to add and manage schemas.

During the configuration phase, you can use the Schema Browser view to fetch a schema from the source or target application and apply the schema to an adapter service. The adapter adds that schema definition as a schema class. You can configure a schema to describe the structure of messages processed by the adapter.

To work with the Schema Browser view, complete the following tasks:

1. Adding a destination
2. Fetching tables or stored procedures
3. Replicating schema information

Difference Between Database Schema and TIBCO ActiveEnterprise Schema

When you create an adapter service with a table or stored procedure selected, the adapter user interface creates a folder in the `AESchemas/ae/ADB` folder for the adapter configuration and transforms the created folder into an AE schema.

Different from database schema objects, an AE schema is recognized by TIBCO Adapter SDK, which is the fundamental class library used in the adapter implementation.

An AE schema consists of five parts: associations, classes, scalars, sequences, and unions. The transformation is done automatically. For most cases, do not modify the AE schema manually.

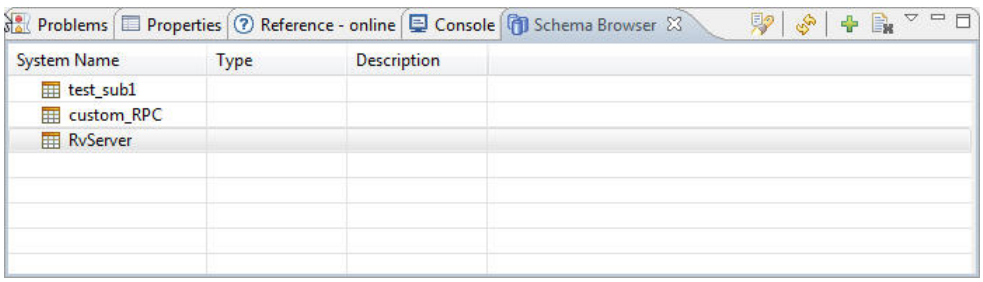
Adding a Destination

You can add destinations for multiple backend data sources, for example, SAP and databases. Each destination represents the root of a repository, and when expanded, the destination displays a hierarchical structure representing the organization of the backend system.
If a destination is added for a database, when expanded, the nodes for tables, stored procedures, or functions are displayed.

You can add a database destination to the Schema Browser view by launching the Application Explorer Wizard window.

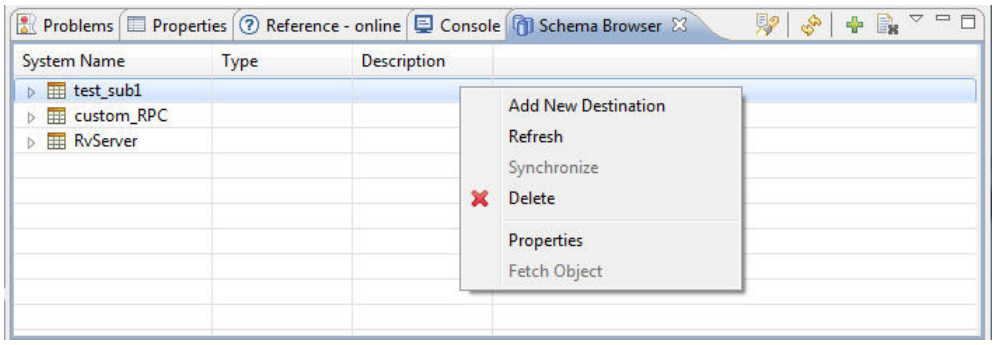
Procedure

1. Open the Schema Browser view in one of the following ways:
   - From the main menu, click **Window** > **Show View** > **Other** to open the Show View window. Then click **TIBCO Adapters** > **Schema Browser** and click **OK**.
   - Click the **Browsing Schema** link in the **Adapter Services** tab.



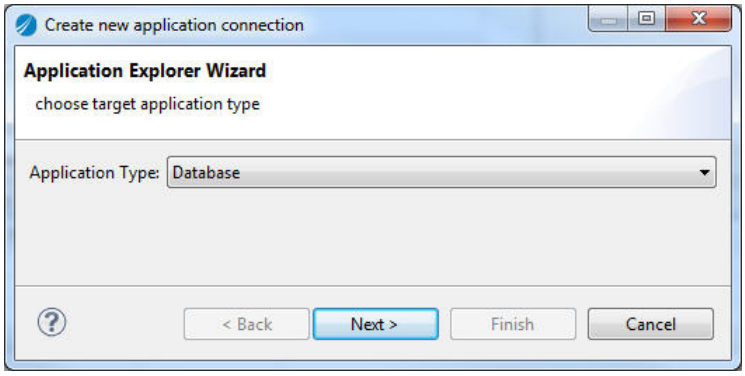
2. Launch the Application Explorer Wizard window in one of the following ways:

   - Right-click **Schema Browser**, and click **Add New Destination** from the pop-up menu.



   - Click the Add button from the action bar of the Schema Browser view.

3. In the Application Explorer Wizard window, select **Database** from the **Application Type** list. Click **Next**.



4. In the Basic Information window, specify a display name and provide a description if necessary. Click **Next**.
5. In the New Connection Information window, specify database connection details. Click **Test Connection** and ensure that the information is correct.

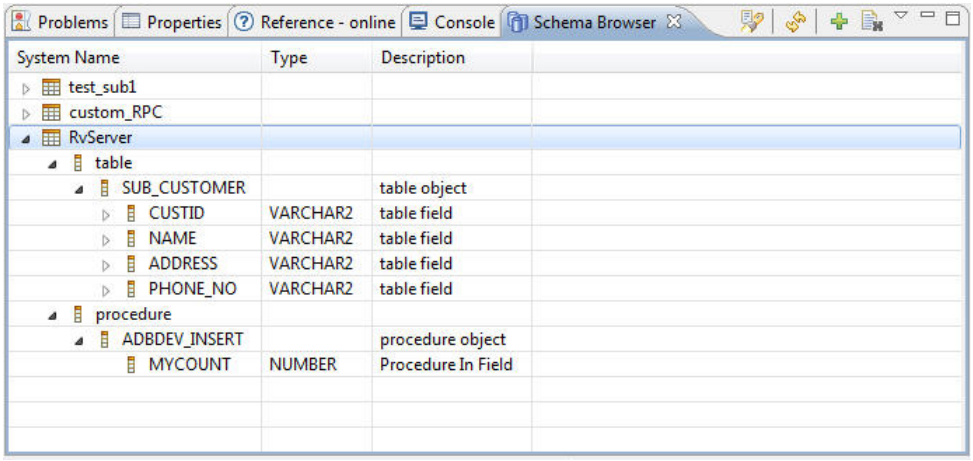
   For details about a database connection, see New Connection Information.

   To ensure that the adapter works correctly, the database connection created in the Schema Browser view has to be the same as the one you configured in the adapter configuration.

6. Click **Finish**.
   A destination with the name you specified is added in the Schema Browser view and you can expand it to show top-level objects.

Fetching Tables and Stored Procedures

You can use the Schema Browser view to fetch tables, stored procedures, functions, and packages from a database destination.
The following figure shows the Schema Browser view with fetched tables and stored procedures.



The fetch action following a query only downloads the descriptors for various fetched objects instead of the entire object content. The descriptor contains information, such as the name of a fetched object, associated connection, columns or parameters of the fetched object, data type of the columns or parameters, and so on.

An actual object is downloaded in situations, such as service creation and schema import.

For details on how to fetch tables and stored procedures, see Fetching Tables and Fetching Stored Procedures.

Fetching Tables

Each service of Publication Service or Subscription Service must be associated with a table that describes data the service receives from or sends to the database.

You can use the following ways to fetch tables and child tables:

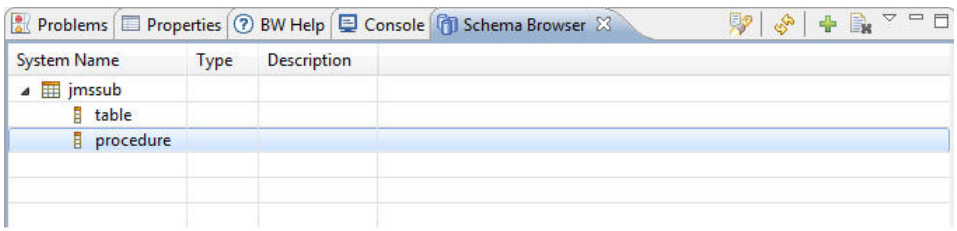
- Use the Schema Browser view. This way is used in the following procedure.
- Use the **Schema** tab of an adapter service. For details, see Fetching a Table After Creating Publication Service and Fetching a Table After Creating Subscription Service.
- Use the **Remote Business Object** tab in the Create Service Wizard window. For details, see Fetching a Table When Creating Publication Service and Fetching a Table When Creating Subscription Service.

Prerequisites

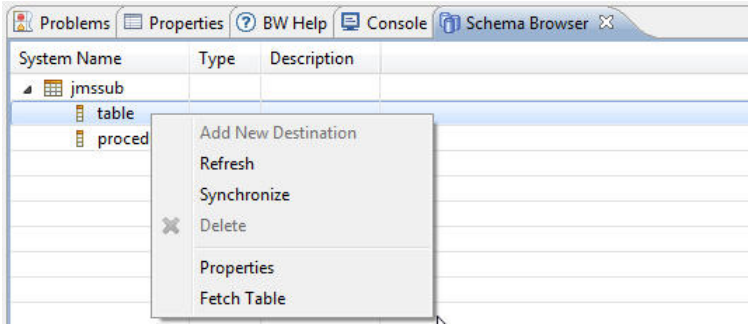
Ensure that a destination is created in the Schema Browser view by clicking the **Link Connection To Schema Browser** link after you create an adapter connection.

Procedure

1. In the Schema Browser view, expand the database destination from which you want to fetch a table.

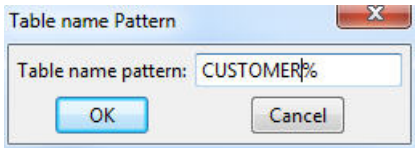


2. Right-click the table node and from the pop-up menu, click **Fetch Table**.



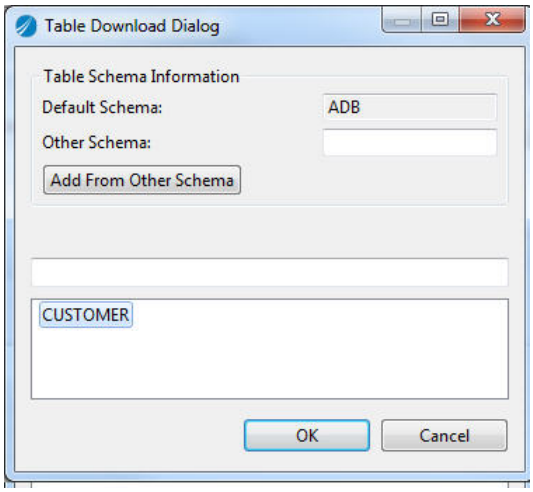
3. In the Table name Pattern window, enter a search criteria and click **OK**.

   For example, if you want to fetch a table that starts with "CUSTOMER", you can specify `CUSTOMER%` as the search criteria.



You can choose to fetch a set of tables at one time from a database. However, you can fetch only one stored procedure or function at each time, because a unique name must be specified for each stored procedure or function.

4. In the Table Download Dialog window, select the table you want to fetch and click **OK**.



   If you want to reference an external schema, you can specify the schema name in the **Other Schema** field and click **Add From Other Schema** to load its tables. However, you must have the proper access privileges to the referenced schema. For more information, see Referencing an External Schema.

What to do next

If you do not create Publication Service or Subscription Service, after you fetch a table in the Schema Browser view, you can drag the fetched table into the All Adapter Services panel to create an adapter service.

If you have created Publication Service or Subscription Service without fetching a table, after you fetch a table in the Schema Browser view, you can add this table to the adapter service by dragging the fetched table into the **Schema** tab.

Fetching Child Tables

In the Schema Browser view, you can add child tables to a table under a specific destination.

Procedure

1. In the Schema Browser view, expand the database destination for which you want to fetch a table.
2. Expand the table node and select a parent table.
3. Right-click the parent table node and from the pop-up menu, click **Fetch Child Table**.
4. In the Table name Pattern window, enter a search criteria and click **OK**.

   For example, if you want to fetch a table that starts with "ORDER_DETAILS", you can specify `ORDER_DETAILS%` as the search criteria.

   You can choose to fetch a set of tables at one time from a database. However, you can fetch only one stored procedure or function at each time, because a unique name must be specified for each stored procedure or function.
5. In the Table Download Dialog window, select the table you want to fetch and click **OK**.

   If you want to reference an external schema, you can specify the schema name in the **Other Schema** field and click **Add From Other Schema** to load its tables. However, the default schema must have the proper access privileges to the referenced schema. For more information, see Referencing an External Schema.
6. To fetch more child tables, repeat Step 1 to Step 5.

What to do next

If you do not create Publication Service or Subscription Service, after you fetch a table in the Schema Browser view, you can drag the fetched table into the All Adapter Services panel to create an adapter service.

If you have created Publication Service or Subscription Service without fetching a table, after you fetch a table in the Schema Browser view, you can add this table to the adapter service by dragging the fetched table into the **Schema** tab.

Fetching Stored Procedures

Custom RPC Service must be associated with a stored procedure or a SQL statement that describes a client request that the adapter has to execute.

You can use the following ways to fetch a stored procedure:

- Use the Schema Browser view. This way is used in the following procedure.
- Use the **Remote Business Object** tab in the Create Service Wizard window. For details, see Fetching a Schema Object for Custom RPC Service.

Procedure

1. In the Schema Browser view, expand the database destination.
2. Right-click the procedure node and from the pop-up menu, click **Fetch Procedure**.
3. In the Store Procedure Download Dialog window, select the stored procedure or function you want to fetch and click **OK**.

   You can also specify the following options in the Procedure Schema Information panel. For details about these options, see Procedure Schema Information Reference.

What to do next

If you do not create custom RPC Service, after you fetch a stored procedure in the Schema Browser view, you can drag the fetched procedure into the All Adapter Services panel to create such an adapter service.

Referencing an External Schema

You can reference a different schema than the default schema to fetch a table. To reference an external schema, the default schema must have the proper access privileges. The privileges are set on a command line.
The ways for granting access privileges vary according to different databases. In the following syntax, *database_username* refers to the default schema in the `create_user.sql` statement.

Oracle Database

Log in as `system` and grant the `create any trigger` and `drop any trigger` permissions to the user. For example,

```
grant create any trigger to adb_schema
grant drop any trigger to adb_schema
```

Sybase Database

Use the following command before creating catalog tables for the external schema:

```
sp_role "grant", sa_role, adb_schema
```

In addition, Oracle and Sybase users must have permission to access a source table in an external schema. If table relationships are used, the SELECT permission is required for both parent and child tables. The SELECT, INSERT, UPDATE, and DELETE permissions are required for accessing a destination table in an external schema.

SQL Server Database

Log in as `sa` and then use the following command before creating catalog tables for the external schema:

```
use master
EXEC sp_addsrvrolemember 'adb_schema', 'sysadmin'
```

DB2 on IBM i/AS400

You can avoid table access problems by changing the ActiveDatabase user authority to `*ALLOBJ`.

Replicating Schema Information

You can replicate schema information from a workspace to another workspace.

Procedure

1. Open TIBCO Business Studio and create a project.
   For details, see Creating a Project.
2. In the project, create an adapter configuration.
   For details, see Creating an Adapter Configuration.
3. Open the adapter configuration in the adapter configuration editor, and create a database connection.
   For details, see Creating a Database Connection.
4. In the workspace from which you want to replicate the schema information, copy the `.appexplorerrep` file from the `source_workspace_directory/.metadata/.plugins/com.tibco.adapter.tool. app.explorer` directory.

5. Replace the `.appexplorerrep` file in the current workspace with the copied file.
6. In the current workspace, right-click the Schema Browser view. From the pop-up menu, click **Refresh**.
   The schema information is refreshed with the information from the source workspace.

Table Reference

When TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) publishes data to or retrieves data from a source or target application, the adapter has to deal with several tables.

The tables are listed as follows:

- Publishing Table
- Source Table
- Exceptions Table
- Child Exceptions Table
- Opaque Exceptions Table

For details about supported data types, see Supported Data Types.

Publishing Table

Publishing tables mirror tables that you have identified for monitoring. They contain additional columns, primarily a sequence number and delivery status, which are required by the adapter to detect new rows. You can create a publishing table for each table that you want to activate in TIBCO Business Studio.

The trigger generated at design time automatically populates the values for these additional columns. It is good practice not to modify these values.

In addition to a copy of columns in the source table, the publishing table has the following additional columns:

Column Name	Type	Description
ADB_SUBJECT	Varchar2(255)	Specifies the subject used to publish the current row. The length is 255 characters. You can set a message subject in this field, which takes precedence over the default service subject. The adapter publishes the current row with the new message subject. In group messaging, the group messages are sent to the subject that is set for the last row of the group.  Subscription Service discards a message in which the ADB_SUBJECT column value is the same as the ADB_SUBJECT column value in an existing record in the destination table of Subscription Service.
ADB_SEQUENCE	Integer	Stores the continuously increasing sequence number that represents new rows in the publishing table. If a column with this name exists, a number is generated automatically. By default, the schema type is String. The adapter treats this number as a string. This number can be larger than an integer if the database supports it.
ADB_SET_SEQUENCE	Integer	Currently not used.

Column Name	Type	Description
ADB_TIMESTAMP	Date	Time of row insertion in the publishing table that is used to calculate expiration of rows. The timestamp is generated automatically.
ADB_TRACKINGID	Varchar2(40)	Tracking ID of the message. This is automatically added to the publishing table and the publishing schema. If you do not want to monitor the tracking ID, you can manually remove this field from the project schema and the publishing table.
ADB_OPCODE	Integer	Operation code used by an adapter configuration: • `1` indicates INSERT. • `2` indicates UPDATE. • `3` indicates DELETE. • `4` indicates UPSERT. If a row exits, an UPDATE operation is performed; otherwise an INSERT operation is performed. • `10` indicates BYPASS. The ADB_OPCODE_BYPASS column is used to bypass the current table operation. For details, see Incremental Parent-Child Operation. If an incoming TIBCO Rendezvous message does not have an operation code, an INSERT operation is performed.  When you use a parent-child table relationship, it is good practice to set the same value for the ADB_OPCODE columns in the parent and child tables. Or only set the value of ADB_OPCODE column in the parent table and leave the child table empty.
ADB_UPDATE_ALL	INTEGER	Currently not used.
ADB_REF_OBJECT	VARCHAR 2(64)	If Publish by Reference Object is used, this column contains the name of the reference object that provides source data.
ADB_L_DELIVERY_ STATUS	CHAR	Delivery status of a TIBCO Rendezvous message: • `P` indicates pending acknowledgment. • `N` indicates that a new message has arrived, but has not yet been published. • `S` indicates that one thread has got this record, and will not be taken again when other records are polled. • `C` indicates complete. • `F` indicates failure.
ADB_L_CMSEQUE NCE	NUMBER (38, 0)	Certified messaging sequence number associated with this message.

A publishing table cannot contain any user-created columns where the column name starts with ADB_. These characters are reserved for adapter use.

When you use a publishing table, note the following conditions:

- When the adapter publishes a parent-child record or uses Publish by Reference mode, if the key value is empty (null) in the publishing table, the adapter considers this as an error operation. Meanwhile, the adapter updates the failure status in the publishing table by setting the `adb_l_delivery_status` status to `F`. As a result the data is not published.
- In an adapter configuration, if two or more Publication Service services with the Rendezvous Certified (RVCM) transport enabled are set to the same message subject, the `ADB_L_DELIVERY_Status` status in the publishing table cannot be changed from `P` to `C` after data transmission.

Source Table

You can insert, update, and delete data in a source table. If loop detection is enabled, the ADB_SOURCE column is added to the source table.

The following table lists and describes the additional column of the source table if loop detection is enabled:

Column Name	Type	Description
ADB_SOURCE	CHAR	Used for loop detection. Indicates whether the row is inserted or updated as a result of a TIBCO Rendezvous message, rather than user intervention. The valid values are `T` or `NULL`: • `T` indicates that the row is not to be published. • `NULL` indicates the row can be published.  If Subscription Service uses the same source table, you must reload the updated source table for Subscription Service after enabling loop detection. You must also ensure that the ADB_SOURCE column is displayed in the loaded table.

The source table cannot contain any user-created columns where the column name starts with ADB_. These characters are reserved for adapter use.

Exceptions Table

If a database restriction or failure occurs, an exceptions table can be configured to receive a message. If insertion into an exception table also fails, an error message is displayed and the adapter configuration is terminated.

You can build a TIBCO Hawk rulebase that detects when the adapter configuration is down and automatically restarts it when the database is up. See *TIBCO Hawk Administrator's Guide* for details on how to create a rulebase.

In addition to the columns in a destination table, the following columns are added to the exceptions table:

Column Name	Type	Description
ADB_OPCODE	Number(38)	Operation code used by the adapter. The values can be: • 1 indicates INSERT. • 2 indicates UPDATE. • 3 indicates DELETE. • 4 indicates UPDATE if a row exists, otherwise INSERT. If an incoming TIBCO Rendezvous message does not have an operation code, an INSERT operation is performed.  When Subscription Service works with a parent-child table relationship, set the same value for the ADB_OPCODE column in the parent and child tables. Or only set the value for the ADB_OPCODE column in the parent table and leave the ADB_OPCODE column in the child table empty.
ADB_UPDATE_ALL	Number(38)	Currently not used.
ADB_TRACKINGID	Varchar2(40)	Tracking ID of a message. This column is the primary key. Each exceptions table that is mapped to a child table is connected to the parent exceptions table by this column.
ADB_ERROR_TEXT	Varchar2(4000)	Text of an error from the database server or another source that caused the exception.
ADB_ERROR_TIME	Date	Timestamp of an inserted record. For Oracle databases, the timestamp includes the time zone information.
ADB_JOIN_ID	Varchar(46)	Joined column used to link a parent record with its child record. The ADB_JOIN_ID column is generated from the ADB_TRACKINGID column and concatenated with the record number in the group. The exceptions table of a child table is connected to the exceptions table of a parent table by the ADB_JOIN_ID column.

An exceptions table cannot contain any user-created columns where the column name starts with ADB_. These characters are reserved for the adapter use.

 When you use Teradata and PostgreSQL databases to work with the exceptions table, if the operation fails, all uncommitted data will be rolled back.

Child Exceptions Table

A child exceptions table contains not only columns in a child table, but also additional columns.

The following table lists the additional columns in a child exceptions table:

Column Name	Type	Description
ADB_ERROR_TEXT	Varchar	Text of an error from the database server or another source that caused the exception.
ADB_TRACKINGID	Varchar	Tracking ID of a message.
ADB_JOIN_ID	Varchar	Joined column used to link a parent record with its child record. The ADB_JOIN_ID column is generated from the ADB_TRACKINGID column and concatenated with the record number in the group. The exceptions table of a child table is connected to the exceptions table of a parent table by the ADB_JOIN_ID column.

Opaque Exceptions Table

An opaque exceptions table records a whole message into a column along with the error message.

Subscription Service uses two logical layers when processing a message. The first layer decodes data from the message and the second layer provides the database transaction. If an exception occurs in the first layer, the adapter logs the message to the opaque exceptions table.

In the second layer, if any DML command fails at any level, the adapter rolls back this transaction and starts another transaction, inserting into exceptions tables. If the insert into exceptions table transaction fails, the adapter then logs the message to the opaque exceptions table.

An opaque exceptions table has the following columns:

Column Name	Type	Description
ADB_TRACKINGID	Varchar	Tracking ID of a message.
ADB_ERROR_TEXT	Varchar	Text of an error from the database server, Adapter SDK or other source that caused the exception.
ADB_ERROR_TIME	Date	Timestamp of the inserted record. For Oracle databases, the timestamp includes the time zone information.
ADB_MSG	Blob	Raw bytes of the message.  The default column size is 1 M. Subscription Service will stop when the inserted data size is larger than the objective table column size and the opaque exceptions table column size.
ADB_SUBTAB	Varchar	Destination table name.
ADB_SUBJECT	Varchar	Subscription Service destination or subject.

Column Name	Type	Description
ADB_TRANSPORT	Integer	Subscription Service transport type: • 0 indicates unknown. • 1 indicates Rendezvous. • 2 indicates JMS.

You can configure several Subscription Service services in the ADB_SUBTAB column by using only one opaque exceptions table in the same database schema.

Incremental Parent-Child Operation

To support incremental parent-child operations, each child row has an opcode, that is, an extra ADB_OPCODE column that is added to the child schema. The ADB_OPCODE_BYPASS column is used to bypass the current table operation.

The adapter determines if the operation is an incremental parent-child operation by checking the first level child opcode. If the first level child opcode is not set, the adapter treats it as a complete operation. For the subsequent child level, if the child opcode is not set, the child opcode inherits the parent opcode.

Mixed parent-child operations are also supported. You can send a message to insert new child rows, update other child rows, and delete other child rows for an existing parent-child object.

When Publication Service works with parent and child tables, the value of the ADB_OPCODE column in the parent table and that in the child table must be the same. Or you can just set the value of the ADB_OPCODE column in the parent table and leave the ADB_OPCODE column in the child table empty.

The following example shows a mixed parent-child operation:

```
adb.key
{
  RVMSG_INT        2   ^type^          1
  RVMSG_INT        2   ^pfmt^          10
  RVMSG_INT        2   ^ver^           30
  RVMSG_INT        2   ^encoding^      1
  RVMSG_RVMSG    110   ^prefixList^
  {
    RVMSG_STRING    49   1
"/tibco/public/sequence/ae/class/ae/ADB/abc"
    RVMSG_STRING    37   default "/tibco/public/class/ae/ADB/abc"
  }
  RVMSG_RVMSG     77  ^tracking^
  {
    RVMSG_STRING    30  ^id^    "Gi2--4--DGMSk--s-064zzw8L-lZW"
    RVMSG_STRING    22  ^1^     "adb.key"
  }
  RVMSG_RVMSG 1200 ^data^
  {
    RVMSG_STRING    8  ^class^                        "S_KEYP1"
    RVMSG_INT       4  ADB_OPCODE                     10
    RVMSG_RVMSG   480  ADB_SEQUENCE_S_KEYP2
    {
      RVMSG_STRING  18  ^class^    "sequence[S_KEYP2]"
      RVMSG_INT      4  ^idx^      1
      RVMSG_RVMSG  210  ^1^
      {
        RVMSG_STRING    8  ^class^                  "S_KEYP2"
        RVMSG_INT       4  ADB_OPCODE               10

        RVMSG_RVMSG   119  ADB_SEQUENCE_S_KEYP3
          {
             RVMSG_STRING 18   ^class^     "sequence[S_KEYP3]"
```

```
            RVMSG_INT      4    ^idx^        1
            RVMSG_RVMSG   58    ^1^
            {
              RVMSG_STRING  8    ^class^               "S_KEYP3"
              RVMSG_INT     4    ADB_OPCODE            4
              RVMSG_REAL    8    A                     1
              RVMSG_STRING  2    B                     "a"
              RVMSG_REAL    8    C                     4
            }
            RVMSG_RVMSG    58    ^2^
            {
              RVMSG_STRING  8    ^class^               "S_KEYP3"
              RVMSG_INT        4  ADB_OPCODE           3
              RVMSG_REAL    8    A                     1
              RVMSG_STRING  2    B                     "a"
              RVMSG_REAL    8    C                     4
            }
          }
    }
    RVMSG_RVMSG       210     ^2^
    {
      RVMSG_STRING          8  ^class^                "S_KEYP2"
      RVMSG_INT             4  ADB_OPCODE             1
      RVMSG_REAL            8  A                      2
      RVMSG_STRING          2  B                      "a"
      RVMSG_REAL            8  C                      4
      RVMSG_RVMSG         119  ADB_SEQUENCE_S_KEYP3
      {
        RVMSG_STRING       18  ^class^    "sequence[S_KEYP3]"
        RVMSG_INT           4  ^idx^      1
        RVMSG_RVMSG        58  ^1^
        {
          RVMSG_STRING      8  ^class^               "S_KEYP3"
          RVMSG_REAL        8  A                     2
          RVMSG_STRING      2  B                     "a"
          RVMSG_REAL        8  C                     4
        }
      }
    }
  }
 }
}
```

When the adapter receives this message, it performs the following database operations in sequence:

1. Bypass the parent table operation.
2. Bypass the first row operation on the `S_KEYP2` child table.
3. `UPDATE S_KEYP3 set B = 'a', C = 4 where A = 1;`
4. `DELETE FROM S_KEYP3 where A = 1;`
5. `INSERT INTO S_KEYP2 (A, B, C) values (2, 'a', 4);`
6. `INSERT INTO S_KEYP3 (A, B, C) values (2, 'a', 4);`

Supported Data Types

For TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio), DataDirect database drivers support different data types depending on the database conditions.

If you select a database that contains data of the BIT type, the BIT data cannot be sent using TIBCO BusinessWorks. In Java, the smallest data type, other than Boolean, are bytes. Generally, a byte requires 8 bits of space to be stored. So the bit column in the database cannot be less than 8 bits of space.

DB2

The following DB2 data types are supported by DataDirect database drivers:

DB2 Data Type	JDBC Data Type
BIGINT  Supported for DB2 V9.1, V10 for z/OS.	BIGINT
BLOB  Supported only for DB2 V8.1 and later versions for Linux/UNIX/Windows, DB2 for z/OS, and DB2 V5R2 for iSeries.	BLOB
CHAR	CHAR
CLOB	CLOB
DATE	DATE or TIMESTAMP  For DB2 V9.7 for Linux/UNIX/Windows with the Oracle compatibility feature enabled, the Date type maps to the JDBC TIMESTAMP type.
DECIMAL	DECIMAL
DOUBLE	DOUBLE
FLOAT	FLOAT
GRAPHIC	CHAR or NCHAR  When JDBCBehavior=0, the data type depends on the JVM the application uses. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and later versions, the second value applies.
INTEGER	INTEGER
LONG VARCHAR	LONG VARCHAR
LONG VARGRAPHIC	LONGVARCHAR or LONGNVARCHAR  When JDBCBehavior=0, the data type depends on the JVM the application uses. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and later versions, the second value applies.
NUMERIC	NUMERIC

DB2 Data Type	JDBC Data Type
REAL	REAL
SMALLINT	SMALLINT
TIME	TIME
TIMESTAMP	TIMESTAMP
VARCHAR	VARCHAR

Limitations of Using DB2 Data Types

When you select the DB2 OS390 database from the **Vendor** list in the **Configuration** tab, note the following conditions:

- When you create a table that includes these data types, BLOB, CLOB, and VARCHAR, you can create a publishing table by selecting the Publish by Reference storage mode.
- When you create a table with the CLOB or BLOB as the data type and the data maximum value exceeds 32K limit, Subscription Service will not accept the data.

- When you create a procedure, datatype LONGVARCHAR and LONGVARGRAPHIC are not supported in DB2 for zOS.
- When you create an opaque exceptions table, you have to create a new LOB table space first, and then create the corresponding auxiliary table and the index.
- When you insert a table with the DECIMAL or NUMERIC data type into a project, if you request the numbers with high precision, you have to choose string instead of r8 in the **AE Type** field in the **Schema** tab.

Microsoft SQL Server

The following Microsoft SQL Server data types are supported by DataDirect database drivers:

Microsoft SQL Server Data Type	JDBC Data Type
BIGINT  Supported only for Microsoft SQL Server 2000 and later versions.	BIGINT
BINARY	BINARY
BIT	BIT
CHAR	CHAR
DATETIME	TIMESTAMP
DECIMAL	DECIMAL
FLOAT	FLOAT

Microsoft SQL Server Data Type	JDBC Data Type
LONGBLOB	LONGVARBINARY
LONGTEXT	LONGVARCHAR
MEDIUMBLOB	LONGVARBINARY
MEDIUMINT	INTEGER
MEDIUMINT UNSIGNED	INTEGER
MEDIUMTEXT	LONGVARCHAR
IMAGE	LONGVARBINARY
INT	INTEGER
MONEY	DECIMAL
NCHAR	CHAR or NCHAR  When JDBCBehavior=0, this data type depends on the JVM used by the application. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and later versions, the second value applies.
NTEXT	LONGVARCHAR or LONGNVARCHAR   When JDBCBehavior=0, this data type depends on the JVM used by the application. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and later versions, the second value applies.
NUMERIC	NUMERIC
NVARCHAR	VARCHAR or NVARCHAR  When JDBCBehavior=0, this data type depends on the JVM used by the application. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and later versions, the second value applies.
REAL	REAL
SMALLDATETIME	TIMESTAMP
SMALLINT	SMALLINT

Microsoft SQL Server Data Type	JDBC Data Type
SMALLMONEY	DECIMAL
TEXT	LONGVARCHAR
TIMESTAMP  When Microsoft SQL Server 2012 is used, the adapter does not support the TIMESTAMP data type.	TIMESTAMP
TINYINT	TINYINT
TINYTEXT	LONGVARCHAR
UNIQUEIDENTIFIER	CHAR
VARBINARY	VARBINARY
VARCHAR	VARCHAR

Limitations of Using Microsoft SQL Server Data Types

When you use Microsoft SQL Server data types, note the following conditions:

- When you insert a table with the TEXT, NTEXT, or IMAGE data type into a new project, the adapter fails to create a trigger, and the project cannot be saved.
- When you insert a table with the MONEY, SMALLMONEY, DECIMAL, or NUMERICA data type into a project, if you request the numbers with high precision, you must choose `string` instead of `r8` in the **AE Type** field in the **Schema** tab.
- When you create a table that contains the TIMESTAMP data type, the publishing tables cannot be created.

MySQL

The following MySQL data types are supported by DataDirect Database dirvers:

MySQL Data Type	JDBC Data Type
BIGINT	BIGINT
BIGINT UNSIGNED	BIGINT
BINARY	BINARY
BIT	BINARY
BLOB	LONGVARBINARY
CHAR(n)	CHAR
DATE	DATE

MySQL Data Type	JDBC Data Type
DATETIME	TIMESTAMP
DECIMAL	DECIMAL
DOUBLE	DOUBLE
FLOAT	REAL
INTEGER	INTEGER
LONGBLOB	LONGVARBINARY
LONGTEXT	LONGVARCHAR
MEDIUMBLOB	LONGVARBINARY
MEDIUMINT	INTEGER
MEDIUMINT UNSIGNED	INTEGER
MEDIUMTEXT	LONGVARCHAR
SMALLINT	SMALLINT
SMALLINT UNSIGNED	SMALLINT
TEXT	LONGVARCHAR
TIME	TIME
TIMESTAMP	TIMESTAMP
TINYBLOB	LONGVARBINARY
TINYINT	TINYINT
TINYINT UNSIGNED	TINYINT
TINYTEXT	LONGVARCHAR
VARBINARY	VARBINARY
VARCHAR(n)	VARCHAR
YEAR	SMALLINT

Limitations of Using MySQL Database Types

When you use the MySQL database, note the following conditions:

- When you run Subscription Service with the MySQL database connected, Subscription Service does not support the YEAR data type. To resolve this issue, add the `yearIsDateType=false` property in the JDBC URL.
- When you run Request-Response Service with the MySQL database connected and set DOUBLE as the data type, you have to change the DOUBLE data type to the STRING data type to avoid a boundary value error. You can set the connection option in the **JDBC Properties** tab in TIBCO Business Studio.

Oracle

The following Oracle data types are supported by DataDirect database drivers:

Oracle Data Type	JDBC Data Type
BLOB	BLOB
CHAR	CHAR
CLOB	CLOB
DATE	TIMESTAMP
FLOAT(n)	DOUBLE
LONG	LONGVARCHAR
LONG RAW	LONGVARBINARY
NCHAR	CHAR or NCHAR  When JDBCBehavior=0, this data type depends on the JVM used by the application. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and later versions, the second value applies.
NCLOB	CLOB or NCLOB  When JDBCBehavior=0, this data type depends on the JVM used by the application. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and later versions, the second value applies.
NUMBER	DECIMAL
NUMBER(p, s)	DECIMAL

Oracle Data Type	JDBC Data Type
NVARCHAR2	VARCHAR or NVARCHAR  When JDBCBehavior=0, this data type depends on the JVM used by the application. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and later versions, the second value applies.
RAW	VARBINARY
TIMESTAMP  Supported only for Oracle 9i and later versions.	TIMESTAMP
TIMESTAMP WITH LOCAL TIMEZONE  Supported only for Oracle 9i and later versions.	TIMESTAMP
TIMESTAMP WITH TIMEZONE  Supported only for Oracle 9i and later versions.  When FetchTSWTZasTimestamp=false (default), this data type is mapped to the JDBC VARCHAR data type; when FetchTSWTZasTimestamp=true, it is mapped to the JDBC TIMESTAMP data type.	VARCHAR or TIMESTAMP
VARCHAR2  Supported only for Oracle 9i and later versions.	VARCHAR
XMLType	CLOB or SQLXM  When JDBCBehavior=0, this data type depends on the JVM used by the application. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and later versions, the second value applies.

Complex Data Types

Request-Response Service in the custom RPC mode supports Oracle complex data types, which are defined by the standard data types. Currently the following three Oracle complex data types are supported:

Object Type

You can use this data type in the same ways that you use standard data types such as NUMBER or VARCHAR2. For example, you can specify an object type as the data type of a column in a relational table, and you can declare variables of an object type. The value is a variable or an instance of that type. An object instance is also called an object.

Nested Table Type

This is one of the collection data types. A nested table can have any number of elements and is not in order. If you want to run efficient queries on a collection, handle arbitrary numbers of elements, or perform mass insert, update, or delete operations, then use a nested table.

Varray Type

This is one of the collection data types. A varray is an ordered collection of elements. If you want to store only a fixed number of items, or loop through the elements in order, or if you want to retrieve and manipulate the entire collection as a value, then use a varray.

Note the following conditions about Oracle complex data types:

- Before using Oracle complex data types, you have to add the `CatalogOptions= 8` property to the **JDBC URL** field in the Adapter for Database Connection Configuration window. For example,

  jdbc:tibcosoftwareinc:oracle://*server_name*:1521;databaseName=*database_name*;CatalogOptions=8
- The complex data types of Oracle Database 12c multitenant container database (CDB) and pluggable databases (PDBs) are not supported. However, the complex data types of Oracle Database 12c NON CDB are supported.
- Request-Response Service in the custom RPC mode supports the invocation of stored procedures that contain the three Oracle complex data types.
- Oracle functions, procedures, and packages all support these complex data types.

Nested Complex Data Types

Complex data types can also nest complex data types. The following examples show nested complex data types:

Example 1

```
create TYPE num_varray AS VARRAY(4) OF NUMBER(12,2);
create TYPE num_object AS OBJECT(
a num_varray,
b number(8,0)
);
```

Example 2

```
create TYPE float_varray AS VARRAY(4) OF float;
create TYPE float_table AS TABLE OF float_varray;
Example 3:
create TYPE char_object as OBJECT(
a char,
b varchar2(5)
);
CREATE TYPE char_table AS TABLE OF char_object;
```

PostgreSQL

The following PostgreSQL data types are supported by DataDirect database drivers:

PostgreSQL Data Type	JDBC Data Type
BIGINT	BIGINT
BOOL	BIT
BYTEA	BINARY
CHAR	CHAR
DATE	DATE
DECIMAL	DECIMAL
DOUBLE	DOUBLE
INT2	TINYINT, SMALLINT
INT4	INTEGER
INT8	BIGINT
NUMERIC	DECIMAL, NUMERIC
REAL	REAL
TIME	TIME
TIMESTAMP	TIMESTAMP
VARCHAR	VARCHAR, LONGVARCHAR

Sybase and Sybase Adaptive Server Anywhere

The following Sybase data types are supported by DataDirect database drivers:

Sybase Data Type	JDBC Data Type
BIGINT  Supported only for Sybase 15.0 and later versions.	BIGINT
BINARY	BINARY
BIT	BIT
CHAR	CAHR

Sybase Data Type	JDBC Data Type
DATETIME	TIMESTAMP
DECIMAL	DECIMAL
FLOAT	FLOAT
IMAGE	LONGVARBINARY
INT	INTEGER
MONEY	DECIMAL
NUMERIC	NUMERIC
REAL	REAL
SMALLDATETIME	TIMESTAMP
SMALLINT	SMALLINT
SMALLMONEY	DECIMAL
TEXT	LONGVARCHAR
TINYINT	TINYINT
VARBINARY	VARBINARY
VARCHAR	VARCHAR

Teradata

The following Teradata data types are supported by DataDirect database drivers:

Teradata Data Type	JDBC Data Type
BYTE	BINARY
BYTEINT	TINYINT
CHAR	CHAR
DATE	DATE
DECIMAL	DECIMAL
FLOAT	FLOAT
INTEGER	INTEGER
NUMERIC	NUMERIC

Teradata Data Type	JDBC Data Type
REAL	REAL
SMALLINT	SMALLINT
TIME	TIME
TIME WITH TIME ZONE	TIME
TIMESTAMP	TIMESTAMP
TIMESTAMP WITH TIME ZONE	TIMESTAMP
VARBYTE	VARBINARY
VARCHAR	VARCHAR

Working with BusinessWorks Processes

Processes in TIBCO ActiveMatrix BusinessWorks capture and manage the flow of business information in an enterprise between different data sources and destinations. The Adapter palette installed with TIBCO ActiveMatrix Adapter Framework provides activities for wiring adapter services into processes.

For reference information of each adapter activity, see *TIBCO ActiveMatrix Adapter Framework Reference*.

Workflow with Activities and Processes

TIBCO Business Studio Workbench provides a design environment to develop and test a process:

1. Creating a Business Process
2. Testing a Specified BusinessWorks Process
3. Testing Multiple BusinessWorks Processes in Multiple Applications

For details about process development, see *TIBCO ActiveMatrix BusinessWorks Application Development*.

Creating a Business Process

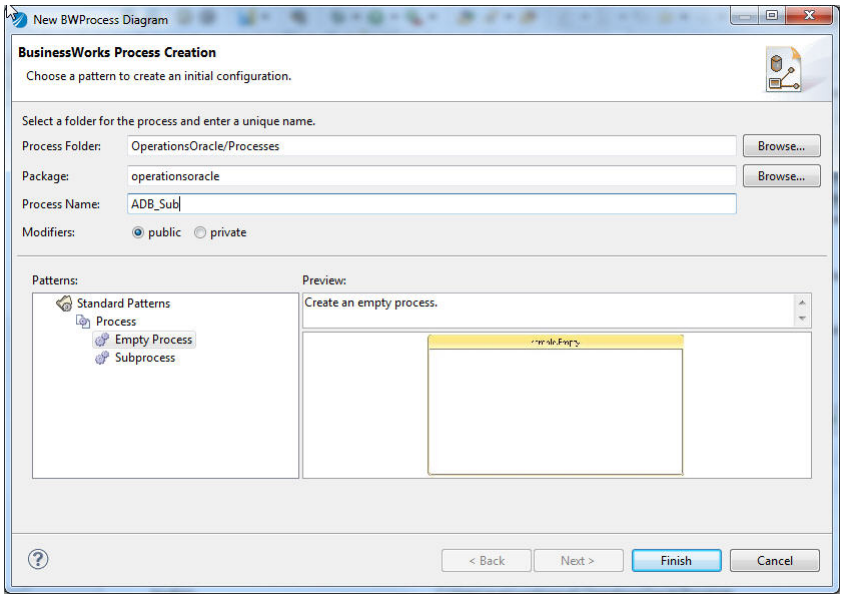
With the TIBCO Business Studio workbench, you can create business processes. The following example is provided to show how to create a business process in TIBCO Business Studio.
In this example, a Publish to Adapter activity is added to a business process to communicate with Subscription Service.

Prerequisites

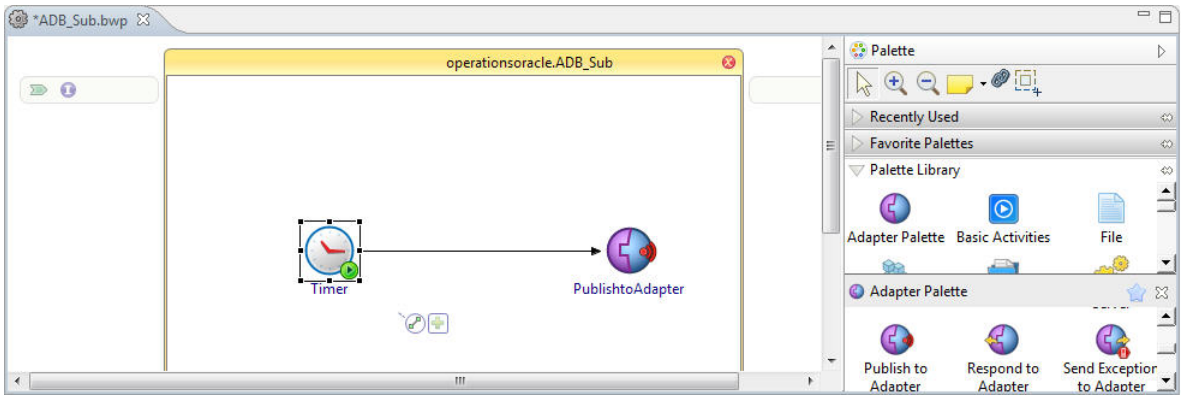
A project is already created in TIBCO Business Studio, and Subscription Service is created in the project.

Procedure

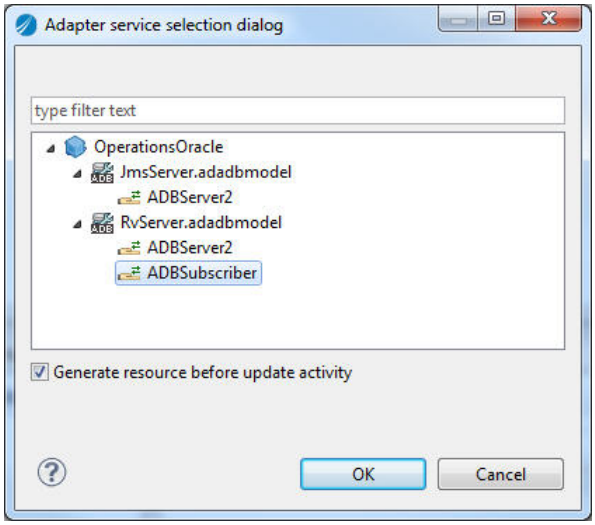
1. In the Project Explorer view, expand your project and right-click **Processes**. From the pop-up menu, click **New** > **BusinessWorks Process**.
2. In the New BWProcess Diagram window, specify a name in the **Process Name** field and click **Finish**.

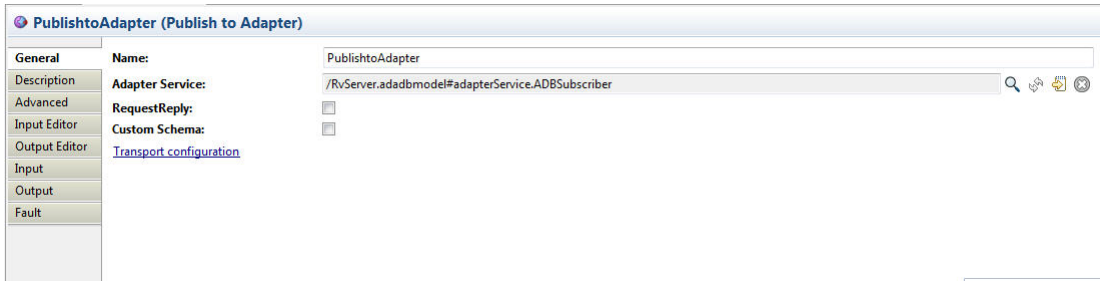


3. In the Project Explorer view, double-click the created process.
4. In the process editor, add the following activities:
   a) Right-click the blank space, and from the pop-up menu click **Add Activity** > **General Activities** > **Timer**.
   b) In the Adapter palette, add a Publish to Adapter activity to the process editor.
   c) Click the icon next to the Timer activity and drag the icon to the Publish to Adapter activity to create a transition between them.

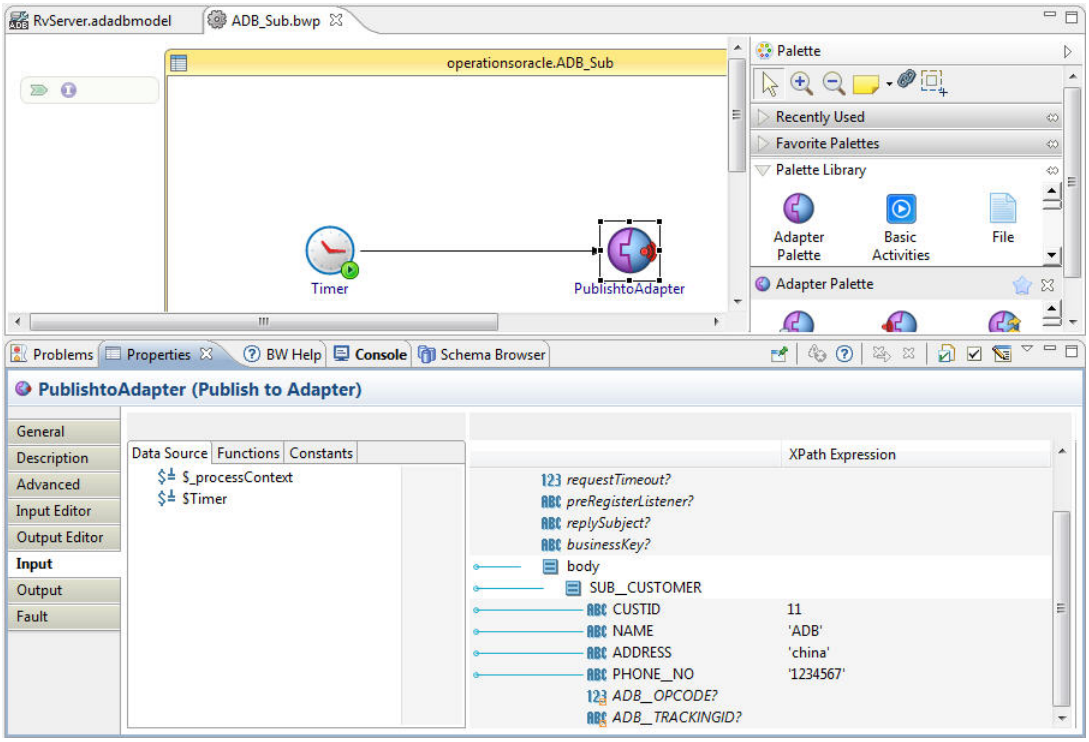


5. Double-click the Publish to Adapter activity and click the **General** tab in the Properties view.
6. Click the browse button next to the **Adapter Service** field to select Subscription Service.





7. In the Properties view, click the **Input** tab. Enter values in the items under the destination table.



Testing a Specified BusinessWorks Process

Similar to testing an adapter configuration, in TIBCO Business Studio, you can test a BusinessWorks process in a few ways. The procedure varies depending on how many projects are created and opened in the workspace.

If only one project is created in the workspace, to test the process, perform the following steps.

If more than one project is created and opened in the workspace, see Testing Multiple BusinessWorks Processes in Multiple Applications for specific procedure.

Procedure

1. From the Project Explorer view, expand *project_name* > **Module Descriptors**, and double-click **Components**.
2. In the Components editor, select the processes you do not test, and click the  button on the right. The processes that are not to be tested are removed from the project components.
3. Save the project.
4. In the Project Explorer view, right-click the `.bwp` file. From the pop-up menu, click **Run As** > **Launch BusinessWorks**.

   Only the processes in the project components are tested.

Testing Multiple BusinessWorks Processes in Multiple Applications

If more than one projects are created and opened in the workspace, the testing method is different from that of testing a single BusinessWorks process. In this case, you can use the Run Configurations window to test multiple BusinessWorks processes.

If you want to test certain processes in the application, you have to first remove the other processes from the project components before you use the Run Configurations window. To do this, repeat step 1 to step 3 in Testing a Specified BusinessWorks Process.

Procedure

1. Open the Run Configurations window in one of the following ways and configure the test details.
   - From the main menu, click **Run** > **Run Configurations**.
   - From the toolbar, click the down arrow of the  button and click **Run Configurations**.
2. In the Run Configurations window, expand **BusinessWorks Application** > **BWApplication**.
3. Click the **Applications** tab and select the applications you want to test. Click **Run**.

Activities for Communicating with Adapter Services

The Adapter palette contains activities for communicating with configured adapter services.

For more information on when and how to use these activities, see *TIBCO ActiveMatrix Adapter Framework Reference*.

The following table lists the adapter services and associated activities in the Adapter palette.

Adapter Service	Associated Activities	Description
Publication Service	Adapter Subscriber	(Process Starter) Starts a process based on the receipt of a message from Publication Service of the specified adapter. Process Starters are activities used to start a process when an event occurs. When a process starter is placed into a process definition, it replaces the default Start activity, and becomes the first activity in the process. Timer, File Poller, and any of the process starter activities are activities used to start a process when an event occurs.
	Wait for Adapter Message	Waits for the receipt of a message from Publication Service of the specified adapter.
Subscription Service	Publish to Adapter	Publishes a message received by Subscription Service of the specified adapter.
Request-Response Service	Invoke an Adapter Request-Response Server	(RPC Mode) Communicates as a client with Request-Response Service of the specified adapter.

Adapter Service	Associated Activities	Description
	Publish to Adapter	(Request-Reply Mode) Publishes a message received by Subscription Service of the specified adapter.

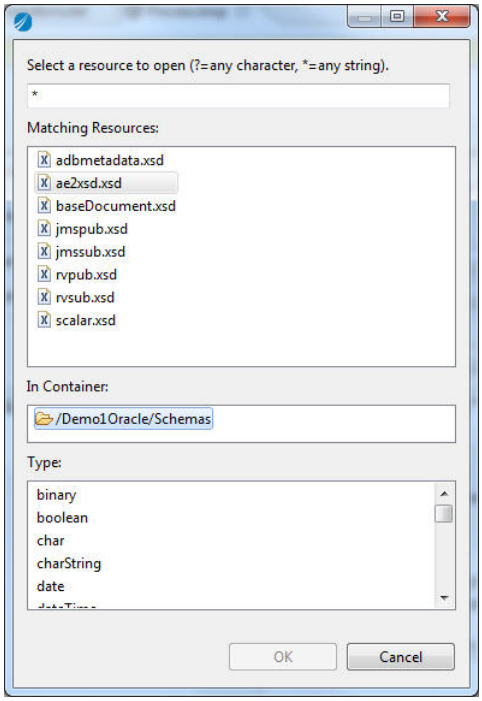
Handling the anyType AE Data Type

When you use activities that are associated with Request-Response Service, the anyType data type is used as the default value in the **DATA** field. If you use this data type, an error message indicating a wrong message format is displayed when you run the BusinessWorks process.

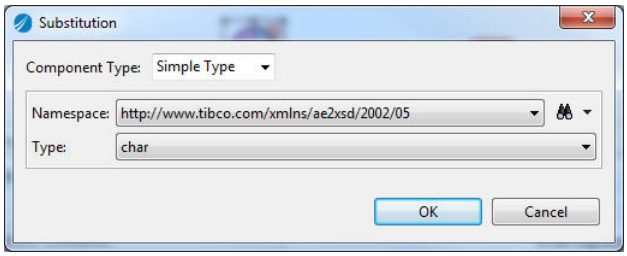
To resolve this issue, you have to substitute this data type with Simple Type that matches the database metadata and select **ae2xsd.xsd** from the resource list.

Procedure

1. In the **Input** tab of the activity, right-click the variable that you want to replace with a new type.
2. From the pop-up menu, click **Substitution**.
3. In the Substitution window, specify the following fields:
   - **Component Type**: select **Simple Type** from the list.
   - **Namespace**: click the browse button to open the window for selecting the data type. Select **ae2xsd.xsd** from the **Matching Resource** list.



   - Select the AE schema in the current project, and select the data type you want to use from the **Type** list.


Substitution
Component Type:
Simple Type
Namespace:
http://www.tibco.com/xmlns/ae2xsd/2002/05
Type:
char
OK
Cancel

4. Click **OK**.

Migrating an Adapter Project Created in TIBCO Designer

You can migrate an adapter project from TIBCO Designer with TIBCO ActiveMatrix BusinessWorks 5.x to TIBCO Business Studio with TIBCO ActiveMatrix BusinessWorks 6.x by using the migration tool in TIBCO Business Studio.

The migration tool in TIBCO Business Studio supports migration from TIBCO ActiveMatrix Adapter for Database 7.2 or higher to TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio).

For related information, see the TIBCO ActiveMatrix BusinessWorks documentation.

When you migrate a project in TIBCO Designer, note the following conditions:

- You can migrate a project from TIBCO Designer with TIBCO ActiveMatrix BusinessWorks 5.x to TIBCO Business Studio with TIBCO ActiveMatrix BusinessWorks 6.x, but not in reverse.
- TIBCO Business Studio does not support the importing of `.dat` files from TIBCO Designer. To migrate a project stored in a `.dat` file, you have to import the file in TIBCO Designer and save it as a multi-file project.
- The name of a project migrated to TIBCO Business Studio cannot contain any of the following characters:

  (space) ! $ % & + . / @ \ ~
- If you want to migrate a TIBCO ActiveMatrix Adapter for Database 6.x project to TIBCO Business Studio, you have to first migrate the project to TIBCO ActiveMatrix Adapter for Database 7.2 in TIBCO Designer.
- You do not have to migrate an existing BusinessWorks 6 project to TIBCO Business Studio. You can import the BusinessWorks 6 project to TIBCO Business Studio. For details on how to import a BusinessWorks 6 project, see *TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) Examples*.

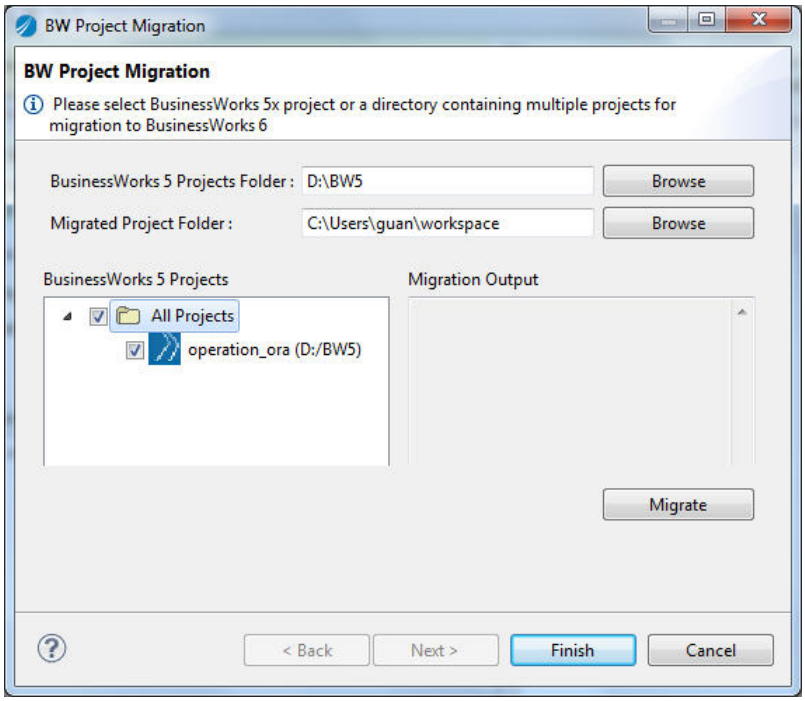
Prerequisites

The project to be migrated is already validated in TIBCO Designer.

TIBCO Designer includes reference-checking and other validation facilities that you can use to ensure that a project is internally consistent. For details, see "Validating Projects" in *TIBCO Designer User's Guide*.

Procedure

1. Open TIBCO Business Studio.
2. From the main menu, click **Project** > **Migrate BW Projects**.
3. In the BusinessWorks Migration Tool window, specify the following fields:
   - **BusinessWorks 5 Projects Folder**: the source location of the projects to be migrated.
   - **Migrated Project Folder**: the target location of the migrated projects.



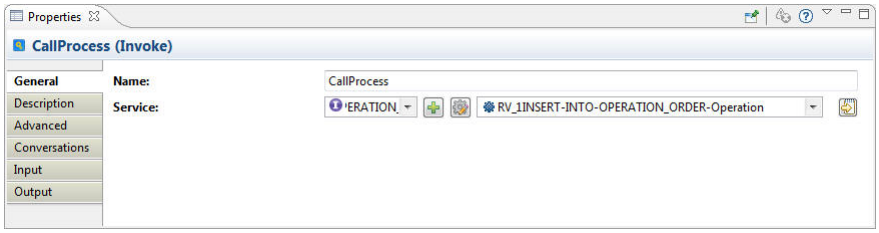
4. After you specify the BusinessWorks 5 project folder, in the displayed project tree, select the adapter projects you want to migrate. Click **Migrate**.
5. When the migration is complete, click **Close**.
   The migrated projects are displayed in the Project Explorer view.

Calling a Process That Uses the Start Activity

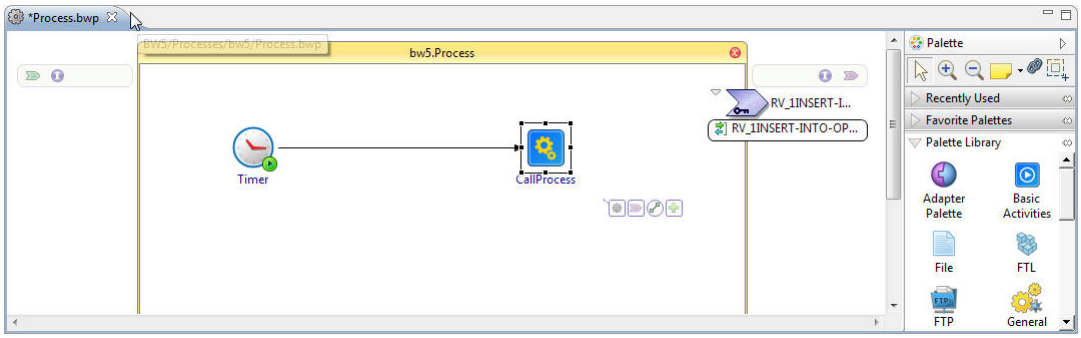
If the migrated project has a process that uses the Start activity, you have to manually create a new BusinessWorks process to call the migrated process.

Procedure

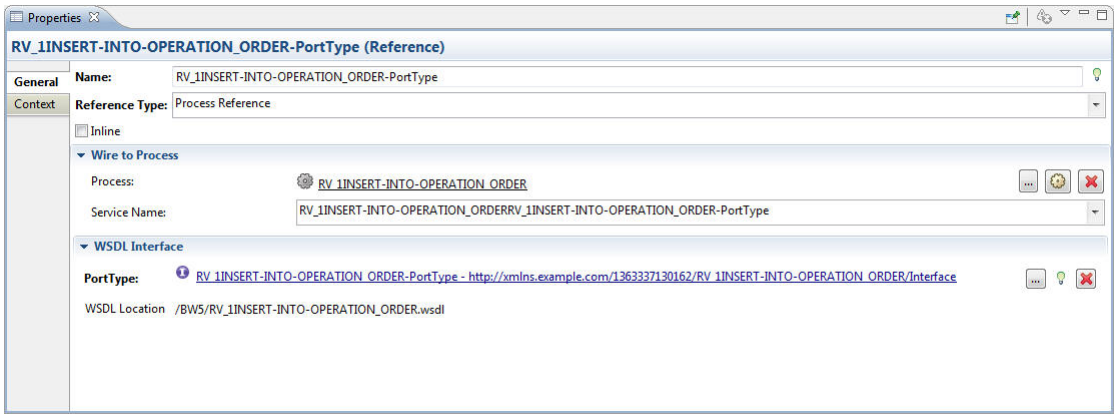
1. Create a new process in TIBCO Business Studio.
2. In the new process, add a Timer, File Poller, or any activity of the process starter type.
3. In the new process, add a CAllProcess activity.
4. Add a transition between the process starter and the CallProcess activity.
5. Select the CallProcess activity and configure the activity:
   a) In the Properties view, click the **General** tab.
   b) Click the  button next to the **Service** field.
   c) In the Select a Service window, select the migrated process. Click **OK**.



After you select the process to call, the process reference is displayed in the created process.



6. Select the reference added to the new process and configure the reference:
   a) In the Properties view, click the **General** tab.
   b) Click **Select Process** next to the **Process** field under **Wire to Process**.
   c) In the Select a BusinessWorks Process window, select the migrated process. Click **OK**.
   d) In the **Service Name** field, select the service from the list.



7. Save the project.

Working with an Alerter

An alerter is used to asynchronously notify Publication Service of database changes. When an alerter is used, the adapter does not poll its publishing table for new rows at every interval. You can use an alerter when database changes are infrequent.

Procedures can be executed on the SQL command line or through any supported Application Programming Interfaces (APIs) (the procedures cannot be invoked successfully within a trigger). The procedures commit the inserts into the database table and notify the adapter of the commit.

Because an alerter runs with a database instance, not user schema, an adapter configuration cannot poll the publishing table name for just one user schema. For example, if the same publishing table exists in two user schemas and two adapter configurations are monitoring the same publishing table, each adapter configuration with a different user, both adapter configurations check the same publishing table whenever the alerter checks for changes in the publishing table.

Currently the alerter method is supported in the following databases:

- Oracle: an alerter for an Oracle database is part of the adapter and available on all operating systems supported by the adapter.
- Microsoft SQL Server 2005 or higher

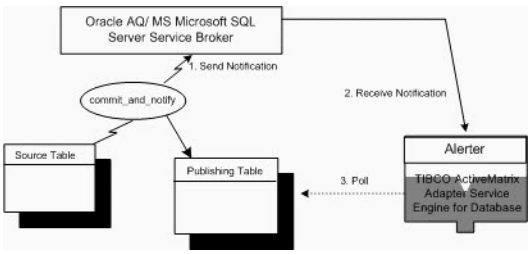
The alerter method is supported for both the TIBCO Rendezvous and JMS transport types.

Alerter Process

To exchange information between sessions, an alerter on an Oracle database uses the Oracle AQ package while an alerter on a Microsoft SQL Server 2005 or higher database uses the Service Broker component. In this way, two or more sessions in the same database instance can synchronously communicate with each other.

Microsoft SQL Server 2005 or higher is shortened as Microsoft SQL Server hereafter.

The following figure shows the alerter process on an Oracle or Microsoft SQL Server database.



As shown in the figure, when a source table is updated with data and the `commit_and_notify` stored procedure is executed, a trigger copies the data to the publishing table and notifies Oracle AQ or Microsoft SQL Server Service Broker that the publishing table has changed.

The alerter receives the notification from Oracle AQ or Microsoft SQL Server Service Broker and sends a message to the adapter to inform that a publishing table has changed. The adapter then polls all its configured publishing tables for the new data and sends it on a subject to the TIBCO transport.

If multiple publishing tables exist under the same database account, you can use the `commit_and_notify_table` stored procedure to specify that only a particular table is checked by the adapter. This prevents the adapter from needlessly checking all its publishing tables for updates when only one table has been updated with new data. This notification can be sent to one or more adapter configurations.

Alerter Object Types and Stored Procedures

If you use an alerter as the polling method, object types and stored procedures are available for committing messages. The following table lists the available object types and stored procedures.

Name	Type	Description
adb_alerter_qtbl (Oracle only)	AQ queue table	Queue table for all alerters.
adb_alerter_q (Oracle only)	AQ queue	Queue for all alerters.
adb_alerter_typ	Service Broker Message Type	Object type for all messages.
adb_alerter_contract (Microsoft SQL Server 2005 only)	Service Broker Contract	The contract type used. The contract specifies the message types that can be used.
*instance_Id*_q (Microsoft SQL Server 2005 only)	Service Broker Queue	Created by the `config_alerter` stored procedure.
*instance_Id*_send_service (Microsoft SQL Server 2005 only)	Service Broker Service	Created by the `config_alerter` stored procedure.
*instance_Id*_rcv_service (Microsoft SQL Server 2005 only)	Service Broker Service	Created by the `config_alerter` stored procedure.
adb_alerter_typ	Object	Object type for the payload message.
commit_and_notify	Procedure	Sends a change notification to all adapters.
commit_and_notify_agent *alerter_name*	Procedure	Sends a change notification to the named alerter.
commit_and_notify_table *table_name*	Procedure	Sends a change notification for a named table.
stop_alerter	Procedure	Stops all running alerters.
stop_alerter_agent *alerter_name*	Procedure	Stops the alerter for the named alerter.
config_alerter	Procedure	Alerter call to register itself as an AQ subscriber.

Name	Type	Description
cleanup_alerter	Procedure	Alerter call to unregister itself as an AQ subscriber.
listen_pipe	Procedure	Alerter call to block on alerts.

Starting an Alerter

To use the alerter polling method, you must install the Oracle AQ package or Microsoft SQL Server package on your specific database. See your Oracle documentation for information on how to install these components.
When configuring Publication Service, you must enable the alerter function by selecting **Alerter** from the **Polling Method** list in the **Configuration** tab. When the adapter starts, the alerter also starts.

When you use the alerter method, note the following:

- If you select the alerter method, the **Polling Interval** field is not displayed.
- If an adapter configuration that uses the alerter method is not shut down cleanly, you must call the `cleanup_alerter` stored procedure before restarting the adapter configuration. This stored procedure is normally called by the adapter configuration when it is shut down cleanly.

For details on how to start an alerter on an Oracle database, see Example for Starting an Oracle Alerter.

For details on how to start an alerter on a Microsoft SQL Server database, see Example for Starting a Microsoft SQL Server Alerter.

Example for Starting an Oracle Alerter

The following example shows how to execute the stored procedure on an Oracle database. In this example, a service of Publication Service with the TIBCO Rendezvous transport is configured with the alerter method selected. It is assumed that the `rvpub.tra` configuration has its data source name, user account name (demo) and password defined in the repository.

Prerequisites

The `create_user.sql` and `alerter.sql` statements are executed on the database server to initialize the database environment. For details, see "Post-Installation Tasks" in *TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) Installation*.

Procedure

1. Start the adapter configuration by typing the following command:

   ```
   adbagent --propFile rvpub.tra
   ```

2. Insert a message and then execute the `commit_and_notify_table` stored procedure with the publishing table monitored by the adapter.

   For example, if the adapter is monitoring the PUB_ORDER table, execute the stored procedure as follows:

   ```
   sqlplus demo/demo
   insert into ORDER_TABLE values(111,'Oak Table',499.95);
   SQL> call commit_and_notify_table ('PUB_ORDER');
   ```

   The procedure puts a message on Oracle AQ. The publisher endpoint then reads its publishing table and sends a message containing the changed data on its configured subject.

Result

This example shows how to notify an adapter configuration to poll a single table, that is, PUB_ORDER, for changes. You can use the `commit_and_notify` stored procedure to poll all publishing tables for changes.

Example for Starting a Microsoft SQL Server Alerter

The following example shows how to execute the stored procedure on a Microsoft SQL Server database. A service of Publication Service with the TIBCO Rendezvous transport type is configured with the alerter method selected. It is assumed that the `rvpub.tra` configuration has its data source name, user account name (demo) and password defined in the repository.

Prerequisites

Ensure that the following two conditions are met:

- The `create_user70.sql` and `alerter_ms.sql` statements are executed on the database server to initialize the database environment.
- The service broker function is enabled by running the `Alter database` *`dbname`* `set enable_broker with rollback immediate` statement on the database server.

  To check whether the service broker function is enabled, execute the following statement: `SELECT is_broker_enabled FROM sys.databases WHERE database_id = DB_ID()`

  If the returned value is `1`, the service broker function is enabled.

Procedure

1. Start the adapter configuration by typing the following command:

   ```
   adbagent --propFile rvpub.tra
   ```

2. Insert a message and then execute the `commit_and_notify_table` stored procedure with the publishing table monitored by the adapter.

   For example, if the adapter is monitoring the PUB_ORDER table, execute the stored procedure as follows:

   ```
   isql -Udemo -Pdemo
   SQL> insert into ORDER_TABLE values(111,'Oak Table',499.95);
   SQL> execute commit_and_notify_table 'PUB_ORDER';
   ```

   The procedure puts a message in the Service Broker queue, which is retrieved by the adapter. The publisher endpoint then reads its publishing table and sends a message containing the changed data on its configured subject.

Result

This example shows how to notify an adapter configuration to poll a single table, that is, PUB_ORDER, for changes. You can use the `commit_and_notify` stored procedure to poll all publishing tables for changes.

Multithreading

Using multiple threads, an application can simultaneously process multiple and independent events. TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) supports multithreading to efficiently handle requests and increase the response speed of the system.

Multithreading is supported for Publication Service, Subscription Service, and Request-Response Service.

When a group message arrives, no matter how many records exist in the group, the records are processed by the same thread as a whole.

For details about the configurations for each service, see:

- Configuring Multithreading for Publication Service
- Configuring Multithreading for Subscription Service
- Configuring Multithreading for Request-Response Service

Configuring Multithreading for Publication Service

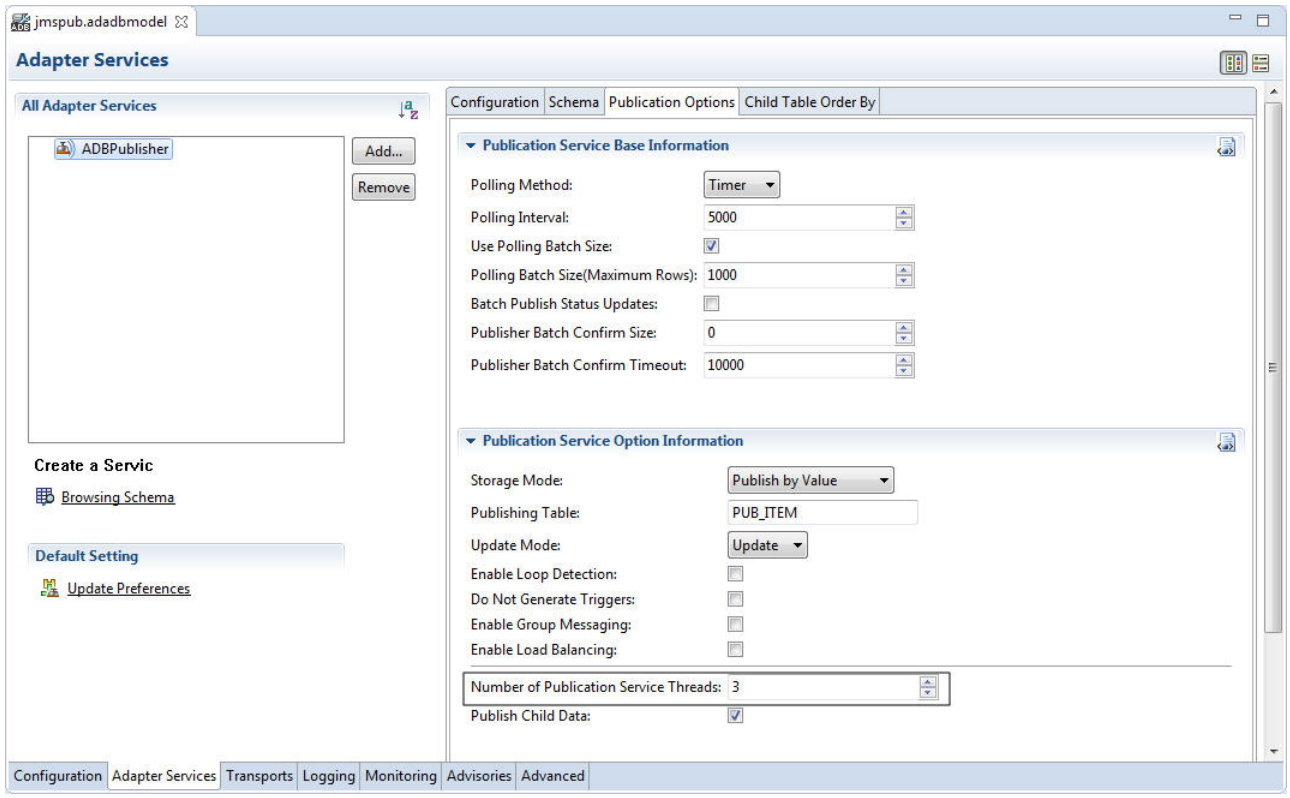
Each service of Publication Service can be configured with multiple threads that connect to the database separately and then process requests in parallel.

TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) supports multithreading for Publication Service. Each Publication Service can hold a session with multiple threads that connect to the database separately, and process requests in parallel.

Load balancing and multithreading cannot be configured together in Publication Service.

You can set the number of threads for Publication Service by configuring the **Number of Publication Service Threads** field in the **Publication Options** tab.

The following figure shows an example of this configuration:



Configuring Multithreading for Subscription Service

To improve the performance in high load scenarios, the adapter uses multiple threads to spread a subscription message load, which prevents blocking the processing on a specific subscription table.

When you configure multithreading in Subscription Service, note the following conditions:

- TIBCO Hawk statistics display all Subscription Service thread information and operation counts.
- Debug messages provide Subscription Service thread names to distinguish which message is from which Subscription Service thread.

If a Subscription Service thread encounters fatal errors that the adapter cannot handle, the adapter terminates. Fatal errors include but are not limited to the following ones:

- Fail to roll back a transaction.
- Fail to insert received data into the destination table, but the exception table is not configured.
- Fail to insert error data into the exception table, but the opaque exception table is not configured.
- Fail to insert error data into the opaque exception table.

The use of multithreading is directly related to the session to be used for Subscription Service. For details, see Multithreading and Session in Subscription Service.

Although multithreading helps improve the performance of Subscription Service, the improvement is affected by what table structures are selected for Subscription Service, a single table or a parent-child table. For details, see Multithreading and Tables for Subscription Service.

Multithreading and Session in Subscription Service

Configuring multithreading for Subscription Service depends on the session used for this type of service.

Default Session

If you do not want to use multithreading, Subscription Service services with the same transport type use the default session to receive messages and update the relevant tables in the database.

To use the default session, you have to keep the **Use Separate Session** check box cleared in the **Subscription Options** tab of the service.

Separate Session

If you want to use multithreading, you must configure Subscription Service to use separate sessions. To do this, finish the following configurations:

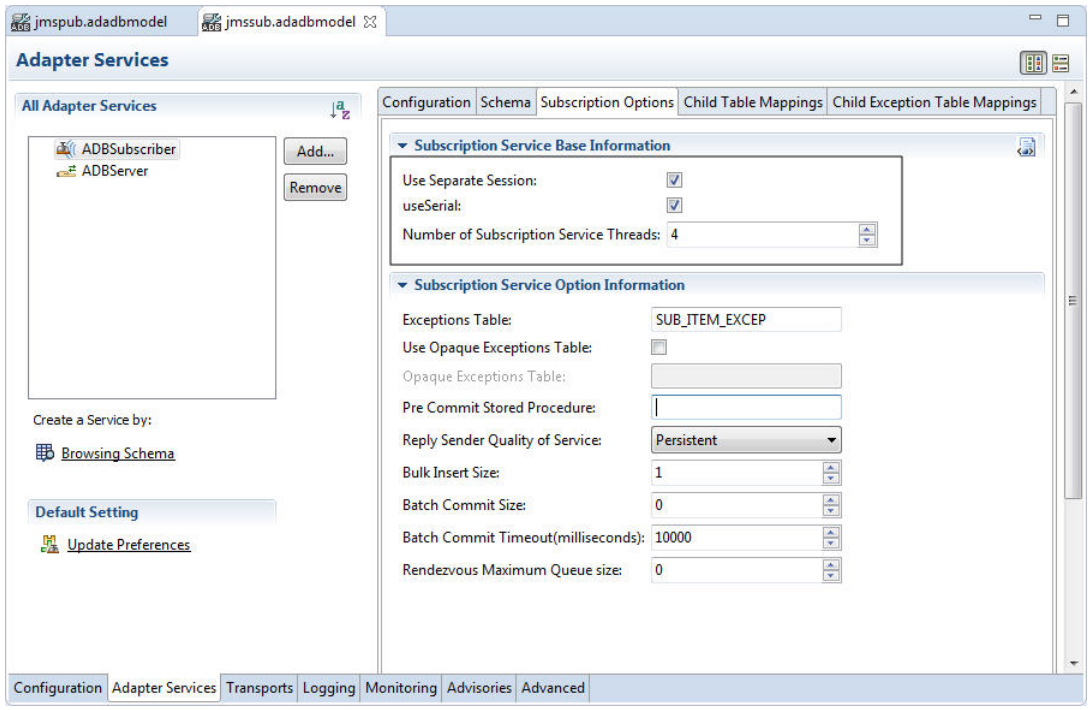
1. Select the **Use Separate Session** check box in the **Subscription Options** tab of the service. The **useSerial** check box and the **Number of Subscription Service Threads** field are displayed.
2. (Optional) Select the **useSerial** check box if you want messages to be handled in order.

To ensure the proper functioning of the **useSerial** option, note the following conditions:

- The destination table must have a primary key.
- If the destination table has no primary key column, you have to manually specify one or more columns by selecting the **User Key** check box. When processing the application requests, Subscription Service handles messages in order.
- If the destination table has no primary key column and the **User Key** check box is cleared, it is good practice to clear the **useSerial** check box.

3. Specify the number of Subscription Service threads.

The following table shows an example of configuration:



When you configure a separate session to use multiple threads, the transport session name must end with one or more digits in the range 0 to 9.

Multithreading and Tables for Subscription Service

Multithreading helps improve the performance of Subscription Service, but the improvement is affected by what table structures are selected for the service. You have two table structure options: a single table and a parent-child table.

The table structures affect the performance of Subscription Service with multithreading in the following ways:

- If a single table structure is used, because only one table is concerned, multithreading helps improve the performance.
- If a parent-child table structure is used, the tables form a tree. That is, a table is either the parent of some of the rest tables or a child of another table, or both. In this case, the performance improvement might be minimal. This is because a thread accesses tables in the exclusive mode for data integrity. As a result, a table currently being processed by a thread is inaccessible to other threads. This table lock must be released by another thread before the table can be used, negating any multithreading advantages on the adapter side.

  Other factors also affect the performance of Subscription Service with multiple tables, such as the number of selected tables, the number of table columns, and the level of the hierarchy of a table tree.

Configuring Multithreading for Request-Response Service

You can improve the performance of Request-Response Service by specifying the number of threads to process application requests. Each Request-Response Service thread is dedicated to listen on an agreed request subject.

You can set multithreading at the adapter configuration level or the adapter service level:

- To set multithreading at the adapter configuration level, configure the **Number of Request-Response Service Default Session Threads** field in the **Advanced** tab of an adapter configuration.
- To set multithreading at the adapter service level, configure the **Request-Response Options** tab of Request-Response Service:

  1. Select the **Use Separate Session** check box.
  2. Specify a value in the **Number of Request-Response Service Threads** field.

If you set multithreading at both the adapter configuration level and the service level, Request-Response Service takes the thread number at the adapter service level during the process.

The configuration of multithreading is directly related to the session to be used for Request-Response Service. For details, see Multithreading and Session in Request-Response Service.

Multithreading and Session in Request-Response Service

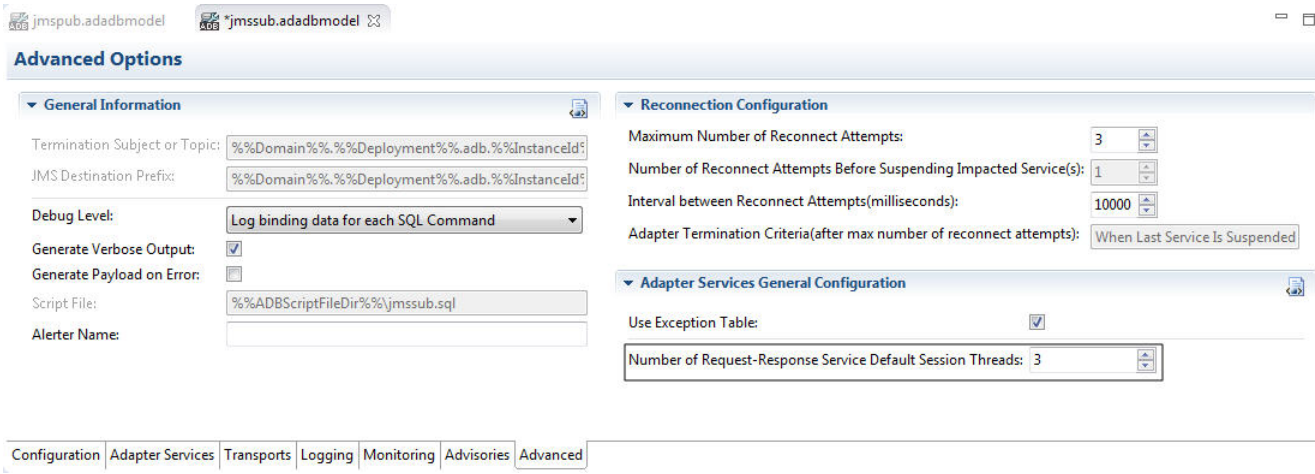
Since for Request-Response Service, multithreading can be configured at both the adapter configuration level and the service level, the configurations at different levels decide the sessions to be used for Request-Response Service.

Specifically, the use of the default session or a separate session depends on the following conditions:

Default Session

If multithreading is configured at the adapter configuration level, Request-Response Service services with the same transport type share the default session to process requests and return the results in response to the client no matter how many services exist in the adapter configuration.

The following figure shows an example of the default session configuration:

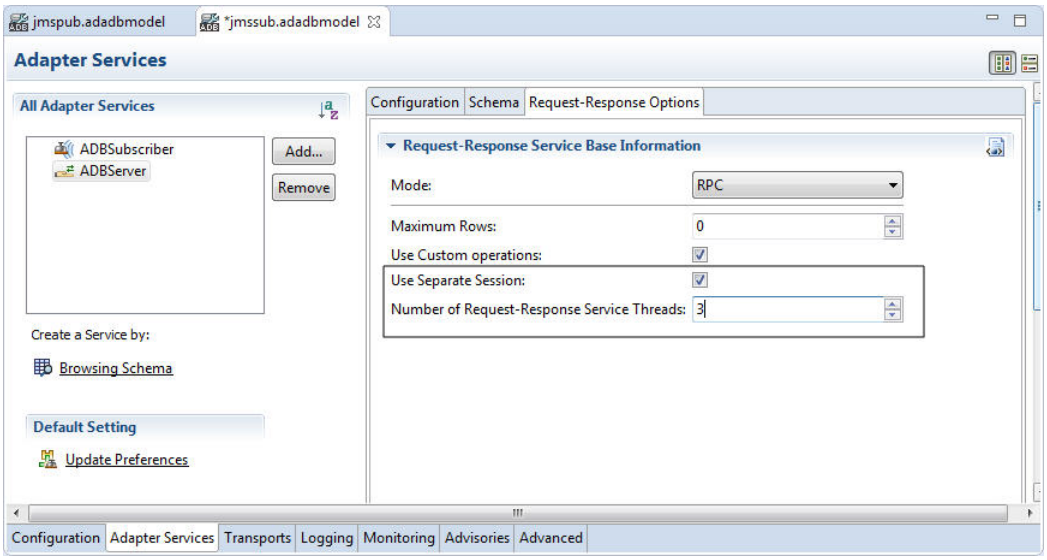


Separate Session

If multithreading is configured at the service level, each service of Request-Response Service holds a separate session with multiple threads that process requests and returns the results in a response to the client separately.

If you want to configure Request-Response Service to use separate sessions for multithreading, the transport session name must end with one or more digits in the range 0 to 9.

The following figure shows an example of the separate session configuration:



Advanced Operations

In addition to the common configurations of the adapter, such as creating a project, creating and configuring adapter configurations, and adding and configuring adapter services, you can perform advanced configurations for the adapter.

The advanced configurations include:

Starting an Adapter Configuration with Adapter Launcher

If you did not specify a working directory for an adapter configuration, you can use the Adapter Launcher tool to perform the configuration before you start the adapter configuration.

Procedure

1. Open the Adapter Launcher tool in one of the following ways:
   - From the main menu, click **Run** > **Run Configurations**.
   - On the toolbar, click the down arrow of the  button and click **Run Configurations**.
2. In the Run Configurations window, right-click **Adapter Launcher** and click **New**.

3. In the displayed configuration panel on the right, specify the options as described in the following table. Then click **Apply**.

Name	Description
Name	Name of an adapter launcher for each adapter configuration.  After you select an adapter configuration to run, the name of the selected adapter configuration is automatically filled in the **Name** field.
Adapter Configuration	Click **Browse** to select the adapter configuration you want to test. All adapter configurations in this workspace are displayed.
Adapter Executable	Select an adapter executable from the list. The adapter executable to run your packaged adapter is shown from a list of choices (each matching a particular installation).
Working Directory	Click **Browse** to provide the directory. The Adapter Launcher tool creates the necessary runtime and support files required by the adapter in this directory. For best results do not edit the files in the working directory. Ensure that the disk where the working directory is located contains enough space to save multiple copies of your project.

4. Click **Run**.
When the adapter configuration is running, the display changes to the Console view. You can now observe the adapter as it is running. Click the stop button  when you want to stop the adapter.

 If you specified the **Working Directory** field of Adapter Launcher in adapter preferences, you can directly run the adapter configuration by clicking **Run As** > **Adapter Launcher**.

Configuring Load Balancing

To improve the performance of an adapter service in heavy load scenarios, TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) supports the configuration of load balancing to distribute workload across services. Load balancing is supported for Publication Service, Subscription Service, and Request-Response Service.

For details, see:

- Configuring Load Balancing in Publication Service
- Configuring Load Balancing in Subscription Service and Request-Response Service

Configuring Load Balancing in Publication Service

Publication Service services with the same mutex name specified are in the same load balancing group. You can configure load balancing at design time for the publisher endpoints.

To configure load balancing for Publication Service, complete the following tasks:

1. Creating an Adapter for Database Connection
2. Creating Publication Adapter Configurations
3. Creating Services of Publication Service

4. Configuring Load Balancing for Publication Service

When working with load balancing in Publication Service, note the following conditions:

- When creating two or more adapter configurations, you must load the same source table to all Publication Service services in each adapter configuration.
- Publishing tables must be the same for all Publication Service services in each adapter configuration. If you select the **Write to Database on Save** check box in one adapter configuration, to ensure that all publishing tables are the same, you must keep this check box cleared in the other adapter configurations.
- Specify the same mutex name in the **Publication Options** tab for all Publication Service services in each adapter configuration.
- Specify the same message subject or destination in the **Configuration** tab for all Publication Service services in each adapter configuration.

Creating an Adapter for Database Connection

This example uses an Oracle database connection.

For detailed procedure, see Creating a Database Connection.

Creating Publication Adapter Configurations

After creating an Adapter for Database connection, you can create publication adapter configurations.

Create two publication adapter configurations, for example, test_pub1.adadbmodel and test_pub2.adadbmodel and configure them to connect to the Oracle database. You must ensure that the test_pub1.adadbmodel adapter configuration is enabled with the **Write to Database on Save** option and the test_pub2.adadbmodel is disabled from the **Write to Database on Save** option.

For detailed procedure, see Creating Adapter Configurations and Adding the Database Connection.

Creating Services of Publication Service

After creating adapter configurations for load balancing, you have to add Publication Service to each adapter configuration.

Create Publication Service in the test_pub1.adadbmodel adapter configuration and fetch a table for the service from the **Remote Business Object** tab. Create another Publication Service in the test_pub2.adadbmodel adapter configuration and fetch a table for the service from the **Local Business Object** tab.

For the detailed procedure, see Creating Publication Service and Fetching a Table When Creating Publication Service.

Configuring Load Balancing for Publication Service

At run time, the Publication Service services of the same load balancing group create a mutex table that was configured in the load balancing group, poll the same publishing table, and distribute loads. The Publication Service services synchronize by locking the mutex table to obtain and update the sequence information.

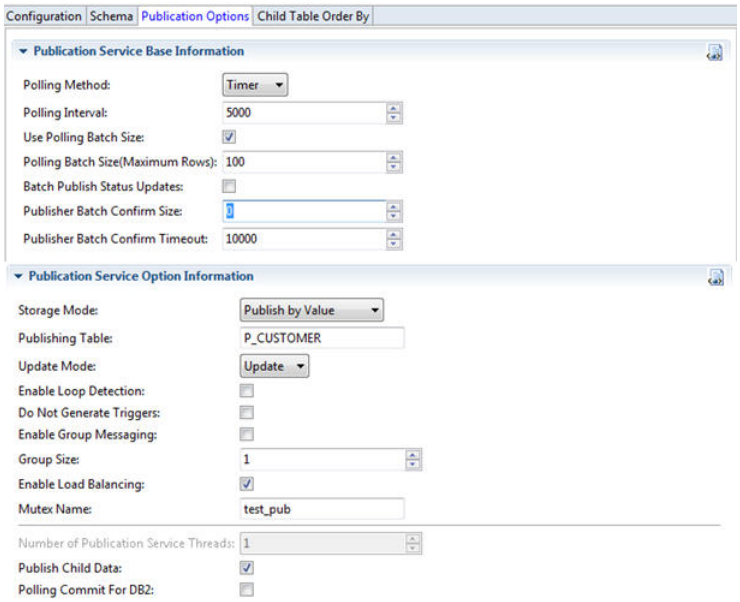
For Sybase, load balancing for Publication Service requires the `ddl in tran` option to be set to `true`. For example, `sp_dboption` *`database_name`*`, "ddl in tran", true`

Procedure

1. Click **test_pub1.adadbmodel** in the Project Explorer view and click the **Adapter Services** tab.
2. Select the created Publication Service from the All Adapter Services panel.
3. In the **Configuration** tab, specify the name in the **Subject** or **Destination** field. For example, **test_loadbalancing_pub**.
4. In the **Publication Options** tab, select the **Use Polling Batch Size** check box, and set a value in the **Polling Batch Size(Maximum Rows)** field.
5. Select the **Enable Load Balancing** check box, and the **Mutex Name** field is displayed.
6. Select one of the following ways to create the mutex table:

   - Enter a name in the **Mutex Name** field, and Publication Service will create a mutex table. For example, **test_pub**.
   - Define a mutex table in your database and then enter the mutex table name in the **Mutex Name** field. For example, you can use the following SQL statement to create a mutex table in the load balancing mode: `CREATE TABLE test_pub (COL1 INT)`

   The following figure shows the configuration in the **Publication Options** tab:

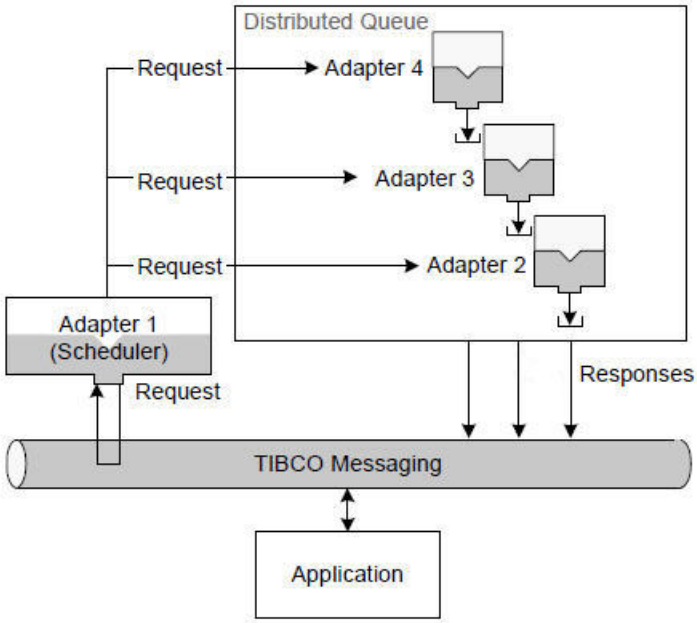


7. Save the project.
8. Repeat Step 1 to Step 7 to configure load balancing in Publication Service test_pub2.

Configuring Load Balancing in Subscription Service and Request-Response Service

TIBCO Rendezvous Distributed Queue or TIBCO JMS Queue can be used to implement load balancing across adapter configurations for Subscription Service and Request-Response Service.

The following figure shows an example on how to use TIBCO Rendezvous Distributed Queue for load balancing:



This example contains three adapter configurations, each connected to a database server (not shown) that contains replicated data. The adapter configurations have been set up to use distributed queue with one adapter configuration acting as the scheduler. Only one of the three adapter configurations receives an incoming request from an application. The scheduler assigns certified requests to the least loaded member of the queue.

The load of a queue member is determined by the number of pending processes that are waiting to be processed by the database Request-Response thread of the member. After the request is processed, the processing configuration sends a response back to the requesting application.

For details, see Example for Configuring Load Balancing in Subscription Service and Example for Configuring Load Balancing in Custom RPC Service.

Request-Response Service in custom RPC mode is hereafter shortened as custom RPC Service.

When working with load balancing in Subscription Service or Request-Response Service, you must note the following conditions:

- In case of two or more adapter configurations, if you want to create the same exceptions table or opaque exceptions table, select the **Write to Database on Save** check box in one adapter configuration and clear this check box in the other adapter configurations. If you do not want to create the same exceptions table or opaque exceptions table, you can select the **Write to Database on Save** check box in all adapter configurations.
- When you create two or more Subscription Service services, you must load the same source table to all the services you created.
- When you specify the message subject or destination name in the **Configuration** tab for all Subscription Service services you created, you must ensure that the message subject or destination names are all the same.
- When you specify the message subject or destination name in the **Configuration** tab for all Request-Response Service services you created, you must ensure that the message subject or destination names are all the same.
- To avoid confusion, do not use multithreading and keep the **Use Separate Session** check box cleared in the **Subscription Options** tab.

Example for Configuring Load Balancing in Subscription Service

An example is provided to show how to configure load balancing in TIBCO Business Studio for subscriber endpoints.

The example includes the following tasks:

1. Creating an Adapter for Database Connection
2. Creating Subscription Adapter Configurations
3. Creating Services of Subscription Service
4. Configuring Load Balancing for Subscription Service

Creating an Adapter for Database Connection (Subscription Service)

You can use the existing database connection Oracle 6694 for the subscription adapter configuration that you will create.
You can also create a new database connection by following the steps in Creating an Adapter for Database Connection.

Creating Subscription Adapter Configurations

After creating an Adapter for Database connection, you can create subscription adapter configurations.

Create two subscription adapter configurations, for example, test_sub1.adadbmodel and test_sub2.adadbmodel and configure them to connect to the Oracle database. You must ensure that the test_sub1.adadbmodel adapter configuration is enabled with the **Write to Database on Save** option and the test_sub2.adadbmodel is disabled from the **Write to Database on Save** option.

For the detailed procedure, see Creating Adapter Configurations and Adding the Database Connection.

Creating Services of Subscription Service

After creating adapter configurations for load balancing, you have to add Subscription Service to each adapter configuration.

Create Subscription Service in the test_sub1.adadbmodel adapter configuration and fetch a table for the service from the **Remote Business Object** tab. Create another Subscription Service in the test_sub2.adadbmodel adapter configuration and fetch a table for the service from the **Local Business Object** tab.

For the detailed procedure, see Creating Subscription Service and Fetching a Table When Creating Subscription Service.

When you create Subscription Service, you must use the Rendezvous Distributed Queue or JMS Queue transport session.

Configuring Load Balancing for Subscription Service

If you use the TIBCO Rendezvous Distributed Queue transport, to configure load balancing in Subscription Service, you have to specify a scheduler weight and a worker weight.

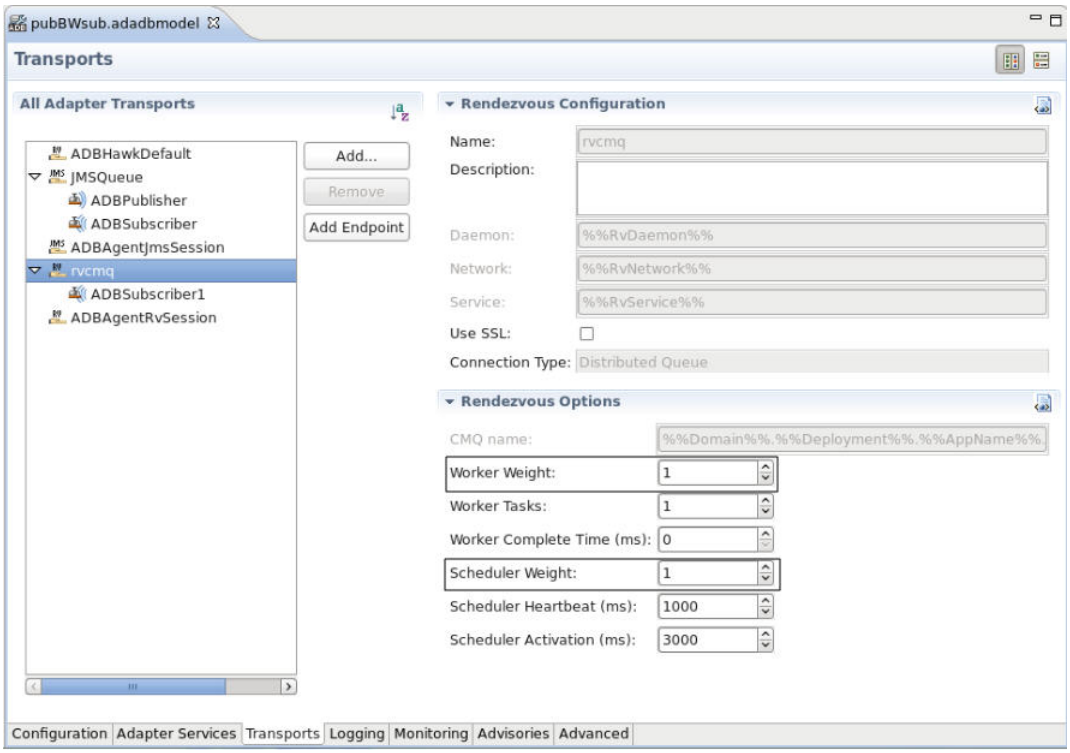
The **Worker Weight** field and the **Scheduler Weight** field are only available for the TIBCO Rendezvous Distributed Queue transport. If you use the JMS Queue transport, you do not have to specify these two options.

The scheduler weight presents how you plan to allocate the jobs. Its default value is 1. The worker weight represents the exact number of processing jobs. No matter what value you set for the scheduler

weight during run time, the adapter processes the jobs according to the value you set for the worker weight. You have to specify the same destination name in the **Configuration** tab for all Subscription Service services.

For details on how to specify a scheduler weight and a worker weight, see "Configuring an Adapter Instance to use TIBCO Rendezvous Distributed Queue or TIBCO JMS Queue for Load Balancing" in *TIBCO ActiveMatrix Adapter for Database Configuration and Deployment*.

The following figure shows how to specify a scheduler weight and a worker weight in TIBCO Business Studio.



Example for Configuring Load Balancing in Custom RPC Service

To configure load balancing in custom RPC Service, you can add just one service to an adapter configuration and then start the adapter service twice. If you create two adapter configurations and add a custom RPC Service service to each adapter configuration, only one adapter configuration handles messages successfully.
This is because the adapter configuration names contained in the operation `aeschema class` paths are different. When you develop a business process, you can only relate one adapter configuration to the business process. When you use the business process to send requests, only the related adapter configuration handles messages. The other adapter configuration fails to handle messages.

Prerequisites

An adapter configuration named custom_RPC is created.

Procedure

1. In the Project Explorer view, double-click **custom_RPC.adadbmodel**.
2. Click the **Adapter Services** tab in the adapter configuration editor.
3. Click **Add** in the All Adapter Services panel.

4. In the Adapter Service General Configuration window, select **Request-Response** from the **Service Type** list and click **Next**.
5. In the Schema Type window, select a stored procedure, function, or package for the service.
   By default, the SQL_REQUEST class is displayed in the **Schema** field.
6. In the Transport Session window, select the TIBCO Rendezvous Distributed Queue or JMS Queue transport session.
   In this example, the JMS Queue transport session is used.
7. Click **Finish**.
8. Start the custom_RPC adapter configuration twice.

What to do next

After you complete the load balancing configuration, you can develop a business process to verify your configuration. In the business process, create two requests and send data to the adapter configuration custom_RPC. If the adapter configuration can handle the requests respectively, the load balancing configuration takes effect.

Batch Processing

TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) supports batch processing for Publication Service and Subscription Service.

For details, see Batch Processing in Publication Service and Batch Processing in Subscription Service.

Batch Processing in Publication Service

Publication Service publishes inserted data from the publishing table and updates the message status to the publishing table. You can configure the service to publish data and update the message status in batches of a specified size. If you are using certified message delivery, you can also configure the service to send confirmation in batches.

You can perform all the configurations by using the **Publication Options** tab.

For better results, do not configure multiple adapter configurations based on the same table for publication. Such a configuration might cause multiple publications of the same message or cause a database deadlock.

Publishing Data and Updating Message Status in a Batch

To publish data and update the message status in a batch, you can select the **Use Polling Batch Size** check box and configure the related options.

After this check box is selected, the following fields are displayed, which you have to configure:

- **Polling Batch Size(Maximum Rows)**
- **Batch Publish Status Updates**

For details about these fields, see Publication Options Tab.

Sending Confirmation in a Batch

If you use TIBCO Rendezvous certified message delivery, to send confirmation in batches, you have to configure the **Publisher Batch Confirm Size** and **Publisher Batch Confirm Timeout** options.

For details about these fields, see Publication Options Tab.

Batch Processing in Subscription Service

The adapter supports the processing of subscription requests in batches at two operation levels: one is insert of the incoming messages; the other is commit of the insert, update, and delete operations.

To use this feature, you have to configure the following fields. You can perform all the configurations by using the **Subscription Options** tab.

- **Bulk Insert Size**
- **Batch Commit Size**
- **Batch Commit Timeout(milliseconds)**

You must select the **Use Separate Session** check box in the **Subscription Options** tab before configuring the **Batch Commit Size** and **Batch Commit Timeout(milliseconds)** fields.

For details about these fields, see Subscription Options Tab.

You can also configure batch commit for Subscription Service without using separate sessions. For details about the configuration, see Configuring Batch Commit for Subscription Service.

Configuring Batch Commit for Subscription Service

In TIBCO ActiveMatrix Adapter for Database 7.2, the `adb.<default_session_name>.subBatchCommitTimeout` and `adb.<default_session_name>.subBatchCommitSize` properties were added to configure batch commit for Subscription Service with the default session.

You can set the following properties in the `adbagent.tra` file:

- `adb.<default_session_name>.subBatchCommitTimeout`
- `adb.<default_session_name>.subBatchCommitSize`

After the properties are set, all services of Subscription Service, which are configured with the same transport type, will share one batch commit timer and one batch commit size.

If you want to use batch commit without using separate sessions, you can set either of the following properties in the `adbagent.tra` file:

- `adb.subBatchCommitTimeout` and `adb.subBatchCommitSize` (with precedence)
- `adb.<default session name>.subBatchCommitTimeout` and `adb.<default session name>.subBatchCommitSize`

For details about these properties, see Adapter Properties.

Handling Passwords

You can use the obfuscate utility installed with TIBCO Runtime Agent to encrypt confidential information, such as passwords in property files, with an encryption key. The obfuscate utility rewrites a Java property file by encrypting property values that start with a #!! or #! prefix. The utility is located in the *TIB_TRA_HOME*`/bin` directory.

If you want to run the adapter locally, specify the runtime password value to be a module property. Before starting the adapter, include the runtime password as a client variable in the properties file and obfuscate it by using the obfuscate utility.

The following example shows how to encrypt a password.

For more information about how to use the obfuscate utility, see "Obfuscate Utility" in *TIBCO Runtime Agent Installation*.

Procedure

1. Specify the password to be `myPassword` as a module property without a value.
2. Include the following entry in the TRA properties file of the adapter at run time:
   `tibco.clientVar.myPassword=#!!`*`passwordValue`*
3. On the command line, enter the following commands:
   `TIB_TRA_HOME``/bin/obfuscate `*`TRAFileName`*

Subject and Destination Names

An adapter configuration uses a subject name or destination name to send and receive messages. When you enable the TIBCO Rendezvous transport type, the adapter configuration uses a subject name to transmit messages; when you enable the JMS transport type, the adapter configuration uses a destination name to transmit messages.

Both subject and destination names are structured strings of alphanumeric characters that are divided into elements by the dot delimiter (.). For example, `Tibco.Tsi` and `News.Sports.Baseball` are valid subject and destination names.

TIBCO Business Studio provides the default subject name and destination name. You can change these default values to specify a custom subject name or destination name. A default subject or destination name, depending on the configured transport type, is created each time an adapter service is created.

The following are limitations of subject names and destination names:

- Length of subject names and destination names:
  - Subject names are limited to a total length of 255 characters (including the dot delimiters). Even though TIBCO Rendezvous has a 255-character subject length limit, applications do not have to use the full length. If you want to run an adapter configuration with the Microsoft SQL server used, you must design your subject space to conform to the length limitation. A subject name contains at most 100 elements. The maximum element length is 127 characters.
  - Destination names are limited to a total length of 249 characters. Even though JMS has a 249-character destination length limit, some of that length is reserved for internal use. You must design your destination name to conform to the length limitation. A destination name contains at most 64 elements. The maximum element length is 127 characters. Dot delimiters are not included in an element length.
- Wildcard characters:

  The asterisk (*) and greater than (>) are wildcard characters that can be used when you specify subject names or destination names.

  For example, if messages are published on the `tsi.sales.sports` and `tsi.sales.clothing` subjects and your subscriber adapter listens on the `tsi.sales.*` subject, the adapter configuration receives messages sent on both subjects. This feature also applies to destination names.
- Reserved underscore (_):

  Subject names or destination names beginning with underscores (_) are reserved. Application programs cannot send subjects or destinations with an underscore as the first character of the first element, however, `_INBOX` is an exception for subject names, and `_INBOX` is an exception for destination names.

  Do not use tabs, spaces, or any unprintable characters. An underscore (_) can be used elsewhere in subject names or destination names. Subject names are case sensitive.
- Empty strings: Do not use empty strings ("") in subject names or destination names.
- Dot character: A dot character cannot be incorporated into an element by using an escape sequence.

For a full explanation of subject names, see *TIBCO Rendezvous Concepts*; for a full explanation of destination names, see *TIBCO Enterprise Message Service User's Guide*.

Parameterized Subject and Destination Names

A subject name or destination name can be created from one or more columns in a publishing table. When you enable the TIBCO Rendezvous transport type, a subject name is created in the publishing table; when you enable the JMS transport type, a destination name is created in the publishing table. A subject or destination created in this way is known as a parameterized subject or destination.

For example, you can use TIBCO Business Studio to define a publication subject as `MYSUBJECT.$COLUMN1.$COLUMN2`, where `COLUMN1` and `COLUMN2` are names of columns in the publishing table. For example: `TIBCO.ORDER.$ORDER_ID.$ORDER_DESCRIPTION`.

On publication, a subject or destination is created from the contents of those columns so that receiving applications can filter publications based on the values of certain fields. Parameterized subject or destination names are supported with certified or reliable delivery.

The following are limitations of parameterized subject names and destination names:

- Length of parameterized subject and destination names:
  - A TIBCO Rendezvous parameterized subject name can be at most 255 characters in length, with each component no more than 127 characters.
  - A JMS parameterized destination name can be at most 249 characters in length, with each component no more than 127 characters.

  In the previous example, if the `ORDER_DESCRIPTION` column contains a description value that is longer than 127 characters, the parameter name `$ORDER_DESCRIPTION` is used as the subject or destination in place of the description value.
- Wildcard characters: Do not use the asterisk (*) or greater than (>) characters, or empty strings as attribute values (VARCHAR, CHAR) in a column that is part of a parameterized subject or destination. Doing so makes the subject or destination invalid.

  For example, if the `TIBCO.ORDER.$ORDER_ID.$ORDER_DESCRIPTION` parameterized subject or destination name is defined and the following values are inserted into the ORDER_TABLE table:

  ```
  insert into ORDER_TABLE values(111,'*',88.81);
  ```

  and the following values are inserted into the `ORDER_DESCRIPTION` table:

  ```
  insert into ORDER_DESCRIPTION TABLE values(112,'>',99.91);
  ```

  the publisher returns an error.
- Data type restrictions:
  - Do not use columns that contain the Date types as part of a parameterized subject or destination. Date values are likely to contain characters, such as dashes and spaces that cause problems when the data values are used as part of a subject or destination.
  - Do not use columns that contain the Binary types as part of a parameterized subject or destination.
  - Do not insert Float-type data as an attribute value into a column that is part of a parameterized subject or destination. Doing so makes the subject or destination invalid.
- Column name: Each element cannot represent more than one column. For example, `MYSUBJECT.$COLUMN1$COLUMN2` is an invalid parameterized subject name.
- Group messaging: The group messaging feature does not support parameterized subjects or destinations. An error message will be displayed, and the adapter will automatically disable the group messaging feature for the publisher. For example, if the TIBCO Rendezvous transport type is used, the following error message will be displayed:

```
Cannot use parameterized subject with group Publishing. Disable group publishing.
```

- Preregistration: Preregistration in the adapter does not work when a parameterized subject or destination is used.

Preregistering a Certified Subscriber

You can add one or more certified subscribers or listeners for the publisher adapter that sends messages on subjects using the TIBCO Rendezvous Certified (RVCM) quality of service. The name of the subscriber is added to the preregistered list of the publisher adapter. The subscriber can be a TIBCO ActiveMatrix Adapter for Database subscriber, or any other TIBCO Rendezvous subscriber.
A subscriber adapter on the preregistered list is certified to receive all messages sent on the specified subject by using the RVCM quality of service. If a subscriber is not preregistered, it might miss one or more of the initial messages sent by the publisher adapter.

When you preregister a subscriber, note the following conditions:

- Wildcard TIBCO Rendezvous subject names are not supported for preregistered listeners.
- Preregistration does not work when parametrized subjects are used.
- When you use Publication Service or a business process to send messages to Subscription Service, it is best practice to configure the preregistered listeners for the RVCM quality of service to prevent data loss.

You can preregister a certified subscriber in TIBCO Business Studio. For specific steps, see Preregistering in TIBCO Business Studio.

Apart from preregistering a certified subscriber in TIBCO Business Studio, you can add a Subscriber adapter to a preregistered list by setting the preregistered listeners in the `adbagent.tra` file. For specific steps, see Setting Preregistered Listeners in TRA Properties File.

Preregistering in TIBCO Business Studio

You can preregister a certified subscriber in TIBCO Business Studio.

Prerequisites

Ensure that the following conditions are met:

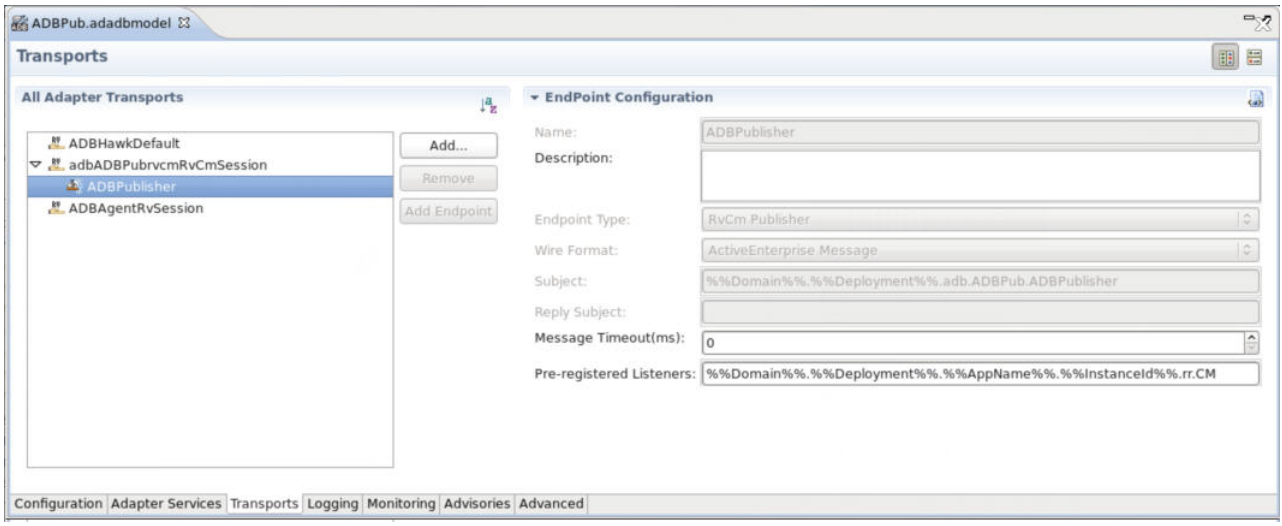
1. An adapter configuration has been created in TIBCO Business Studio.
2. A Publication Service service has been created with the Rendezvous Certified (RVCM) quality of service enabled.

Procedure

1. In the adapter configuration editor for the publisher adapter, click the **Transports** tab.
2. Select **adb***adapter_instance_name***rvcmRvCmSession** from the All Adapter Transports panel.
3. Click its publisher endpoint.
   The EndPoint Configuration panel for this endpoint is displayed.
4. Verify the name of the certified subscriber in the **Pre-registered Listeners** field: `%%Domain%%.%%Deployment%%.%%AppName%%.%%InstanceId%%.rr.CM`.

Note the following conditions about the name of a certified subscriber:

- This name must be the subscriber CM name.
- The *AppName* and *InstanceId* variables are the same as the current adapter name.



5. Save the adapter configuration.

Setting Preregistered Listeners in TRA Properties File

Apart from preregistering a certified subscriber in TIBCO Business Studio, you can add a subscriber adapter to a preregistered list by setting the preregistered listeners in the `adbagent.tra` file.

Prerequisites

Ensure that the following conditions are met:

1. An adapter configuration has been created in TIBCO Business Studio.
2. A Publication Service service has been added to the adapter configuration with Rendezvous Certified (RVCM) quality of service enabled.
3. A Subscription Service service has been added to the adapter configuration with RVCM quality of service enabled.

Procedure

1. Set the message subject of the Publication Service service to a parameterized subject, and set the message subject of the Subscription Service service to a default subject.

   For example, the message subject of the Publication Service service is `adb.$B.kk`, and the message subject of the Subscription Service service is `adb.sub.kk`.
2. Set the CM name in the RVCM session to `sub.CM.tra`.
3. Set the `adb.`*`pub service name.preRegisteredListeners subjectName1:listenerName1`*`, `*`subjectName2:listenerName2`* property to `adb.ADBPublisher.preRegisteredListeners adb.sub.kk:sub.CM.tra` in the `adbagent.tra` file.

Result

The preceding configuration does not take effect, if the database column for creating the parameterized subject contains either of the following characters:

- A period (.)
- Spaces, including fixed-length database column type

Changing the Location of the Ledger File

The ledger file provides temporary storage for published messages that are sent by a publisher adapter using the TIBCO Rendezvous certified messaging quality of service. Each message is held in the ledger file until an acknowledgment that the message has been consumed is received by the subscriber.
If multiple subscribers exist, the ledger file grows to a substantial size. For best results store the ledger file in a subdirectory under the default directory, which is *TIB_ADADB_HOME*, rather than the default directory itself.

You can change the location of the ledger file in one of the following methods:

- By performing configurations in TIBCO Business Studio: For specific steps, see Configuring in TIBCO Business Studio.
- By modifying the TRA properties file: For specific steps, see Modifying the TRA Properties File.

Configuring in TIBCO Business Studio

You can change the directory to store the ledger file in TIBCO Business Studio.

Prerequisites

An adapter configuration has been created in TIBCO Business Studio.

Procedure

1. Select the **adb***adapterInstanceName***rvcmRvCmSession** transport session in the **Transports** tab of the adapter configuration.
2. Specify the **Ledger File** field in the Rendezvous Options panel of the **Transports** tab.

Modifying the TRA Properties File

You can change the directory to store the ledger file by modifying the TRA properties file.

Procedure

1. Navigate to the *TIB_ADADB_HOME*\bin directory, and then open the `adbagent.tra` file with the text editor.
2. Remove the pound symbol (#) at the beginning of the ledger directory command line.
   For example: `tibco.clientVar.DirLedger c:/tibco/adapter/adadb/7.2/ledger`
3. Save and close the `adbagent.tra` file.
   When you restart the adapter configuration, the adapter configuration writes the ledger file to the new location.

Sending DateTime Data to an Adapter Configuration

When TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) publishes messages including datetime data, the wire format is handled implicitly since because Publication Service and Subscription Service use the same format.
In many scenarios, however, a separate product acts as a publisher, that is, you must explicitly specify the wire format. For example, an adapter configuration can subscribe to messages published by another TIBCO product instead of the adapter, such as TIBCO ActiveMatrix BusinessWorks, after some transformation has been performed.

Messages containing datetime data sent to the adapter must have that data in a string format. By formatting dates as strings, the adapter handles date values outside the range supported by

RVMSG_DATETIME, January 1, 1970 to January 1, 2034. The adapter TRA properties also include a few date and time patterns that you can specify to use.

adbDateTime Object

To view the structure of the adbDateTime object, go to the Project Explorer view, and expand **AESchemas** > **ae** > **ADB** > **adbmetadata.schema** > **Classes** > **adbDateTime** > **dateTime**.

The input is a string value in one of the following formats:

- `yyyy-mm-dd`
- `yyyy-mm-dd hh:mm:ss.xxx`

  where `xxx` represents the millisecond value.

Do not change any adbDateTime object information in your screen. Use this information to determine how to format the third-party subscriber date value.

Timezone values are not supported.

The adapter returns an error when the input data contains timezone information. You can use the TIBCO BusinessWorks XPath function to parse the time zone information before the information is passed to the adapter.

Specifying Date and Time Patterns

You can use properties in the `adbagent.tra` file to specify the date, time, and timestamp format patterns. The TRA properties include:

- `adb.datePattern`: the date format pattern for `java.text.SimpleDateFormat`. The default format is `yyyy-MM-dd`.
- `adb.timePattern`: the time format pattern for `java.text.SimpleDateFormat`. The default format is `HH:mm:ss`.
- `adb.timestampPattern`: the standard timestamp format pattern for `java.text.SimpleDateFormat`. The default format is `yyyy-MM-dd HH:mm:ss.S`.

Using the User Callout Java Library

The user callout Java library is used to transform a message that the adapter publishes into a structure that you want to publish. The user callout Java library can be customized to apply limited transformations to outgoing and incoming messages.
If a message is modified before it is sent by a publisher adapter, all subscribers will receive the modified message. If a message is modified before it is received by a subscriber adapter, only that subscriber adapter will receive the modified message.

For example, a publisher adapter can modify a message by adding a field, or publish an empty message if a certain criteria is met. A subscriber adapter can modify a message so that the message can be written to the database by using a filter, or have the subject name inserted into the table.

Before a message is sent or received, the adapter checks the callout library. If the callout library has not been modified, the message is passed back to the adapter unchanged. If the user callout Java library has been modified, the message is passed back to the adapter with the modification. After receiving a modified message from the callout library, an adapter configuration sends or consumes the message like any other message.

Building the User Callout Java Library

The user callout Java library is built with TIBCO Adapter SDK and TIBCO Rendezvous software. For compiler requirements and instructions, see the *TIBCO Adapter for SDK Programmer's Guide*.

To build the user callout Java library, complete the following tasks:

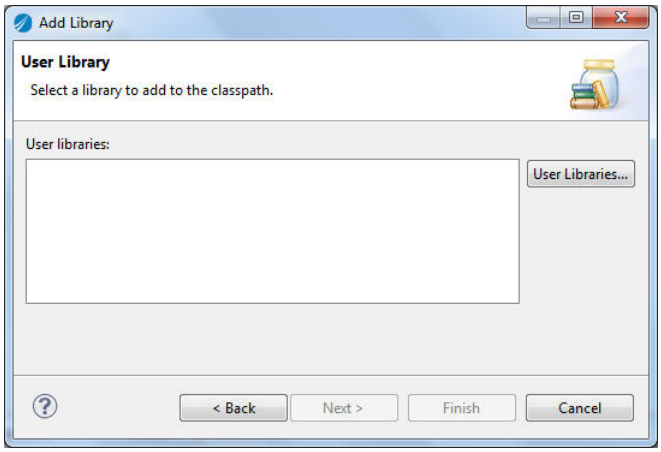
1. Adding a User Library in a Project
2. Adding a Java Package and Class
3. Using the alterMsgPub() and alterMsgSub() Functions
4. Using the adbPreCommit Function
5. Exporting the JAR File
6. Replacing the Original JAR File

Adding a User Library in a Project

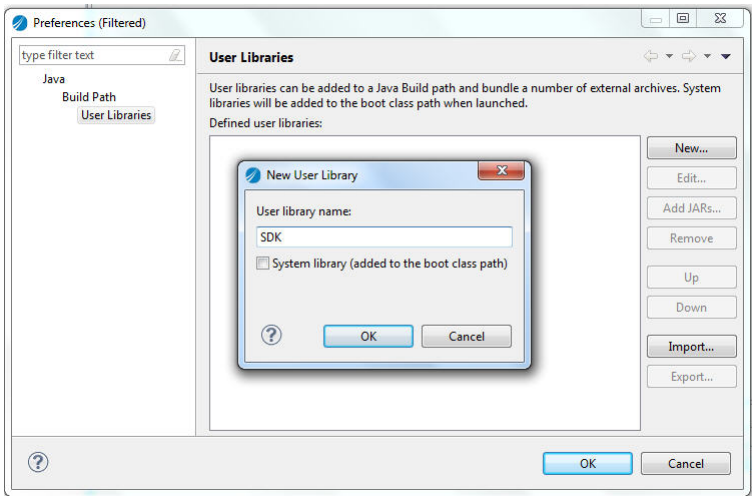
To build the callout library, you have to first add a user library.

Procedure

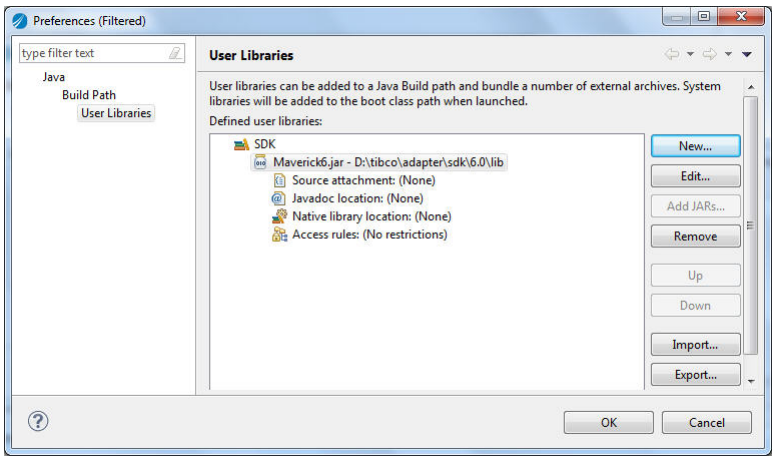
1. In TIBCO Business Studio, create a Java project in your workspace.
2. Right-click the new project, and then select **Build Path** > **Add libraries** from the pop-up menu.
   The Add Library window opens.
3. Select **User Library** from the library type list, and then click **Next**.
   The User Library window opens.



4. Click **User Libraries**.
   The Preferences (Filtered) window opens.
5. Click **New** on the right.
   The New User Library window opens.
6. Enter the library name in the **User library name** field, and then click **OK**.



7. Click **Add JARs**, and open the *TIBCO_HOME*`\adapter\sdk\`*version_number*`\lib\Maverick6.jar` file. Then click **OK** to exit the Preferences (Filtered) dialog.



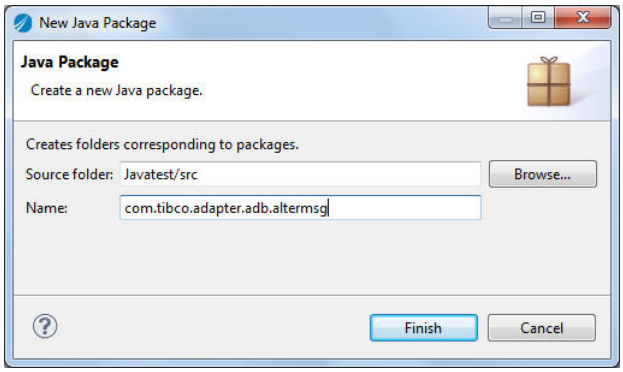
8. Click **Finish** in the User Library window.
   The user library is created.

Adding a Java Package and Class

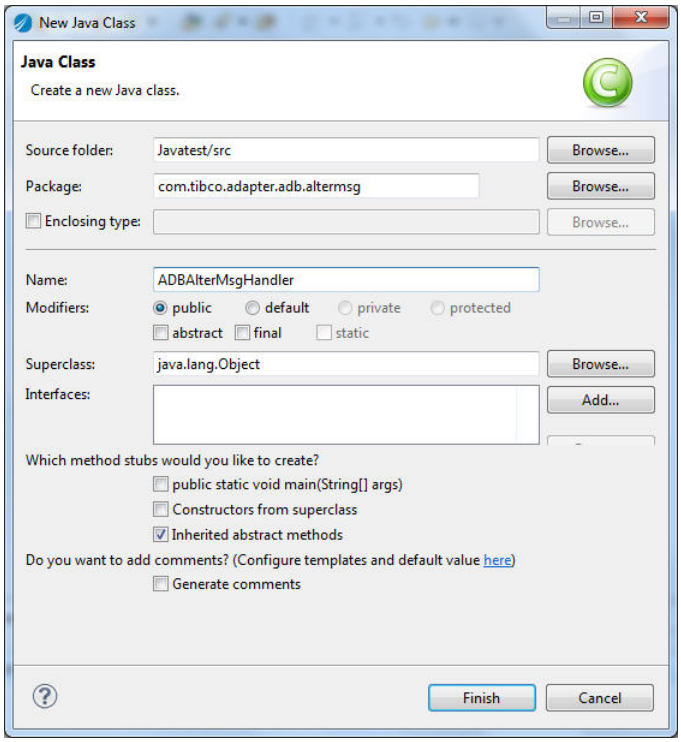
After you add a user library in a project, you have to add a Java package and class.

Procedure

1. Right-click the `src` folder, and then select **New** > **Package** from the pop-up menu.
   The New Java Package window opens.
2. Enter the `com.tibco.adapter.adb.altermsg` package name in the **Name** field, and then click **Finish**.



3. Right-click **com.tibco.adapter.adb.altermsg**, and then select **New** > **Class** from the pop-up menu. The New Java Class window opens.



4. Enter the `ADBAlterMsgHandler` class name in the **Name** field, and then click **Finish**.

Using the adbPreCommit Function

The `adbPreCommit` function is used to perform a custom operation, such as invoking a stored procedure or sending a TIBCO Rendezvous message, just before a transaction is committed. This function is called by the subscriber after all insert, update, and delete operations have been performed on both parent and child tables.

The function signature is as follows:

```
public static int adbPreCommit(Connection connect handle,
                    MInstance data,
                    MApp pMapp, int messageFormat)
```

If the return value is not zero, then the message will be rolled back.

The following table contains parameter descriptions and a column for each format.

Parameter	TIBCO Rendezvous Format	TIBCO ActiveEnterprise Format	Description
connectHandle	X	X	The database connection handle.
pMapp	X	X	TIBCO Adapter SDK MApp structure for this MInstance.
messageFormat	X	X	Format of a message. The valid values are: • `M_AERV_MESSAGE_FORMAT` • `M_XMLJMS_MESSAGE_FORMAT` • `M_XMLRV_MESSAGE_FORMAT` For details about these values, see *TIBCO Adapter SDK Programmer's Guide*.

If an adapter is configured to use the `adb.subBatchCommitTimeout` or `adb.subBatchCommitSize` properties, the custom operation is performed once for each received message. However, the entire transaction is not committed until the size or timeout values are reached. Adapter configurations that use the bulk-insert-size option does not support the `adbPreCommit` function.

For more information about these properties, see "TIBCO ActiveMatrix Adapter for Database Properties" in Adapter Properties.

Using the alterMsgPub() and alterMsgSub() Functions

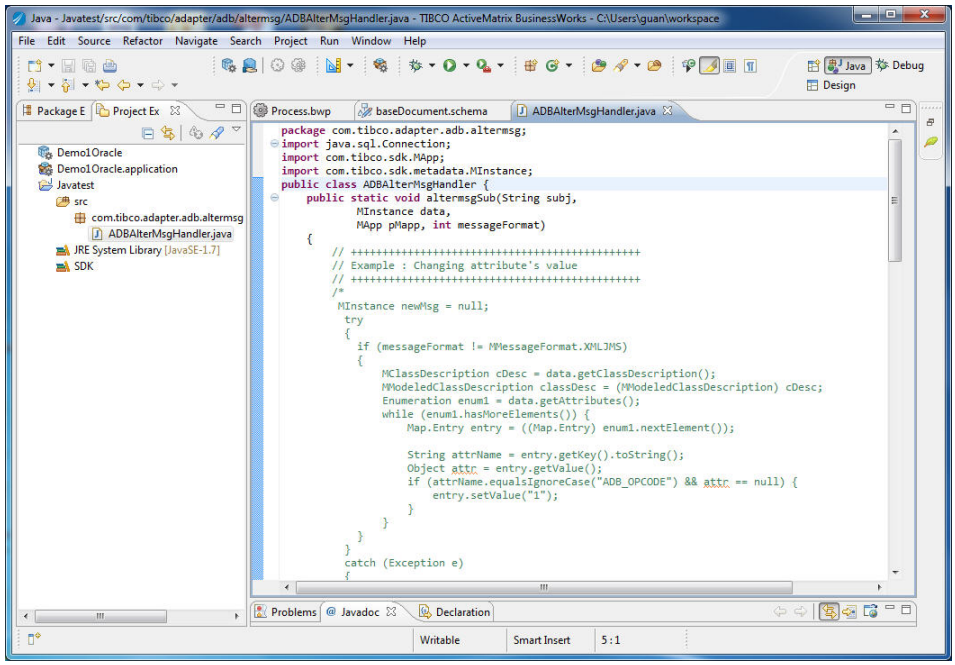
Two functions are available for creating callouts: `alterMsgPub()` is used to create a callout for a publisher, and `alterMsgSub()` is used to create a callout for a subscriber. You can use the `alterMsgPub()` and `alterMsgSub()` functions to alter messages in the TIBCO ActiveEnterprise Message and XML wire formats.
The `alterMsgPub()` and `alterMsgSub()` functions are the entry points into the callout library. These two function names and their signatures must not be changed.

The function signatures are:

```
public static void altermsgPub(String subj,
                   MInstance data,
                   MApp pMapp, int messageFormat)

public static void altermsgSub(String subj,
                   MInstance data,
                   MApp pMapp, int messageFormat)
```

The following figure shows the use of `altermsgPub` and `altermsgSub` functions.



Both functions share the same set of parameters. However, several parameters are specific to a particular wire format.

The following table contains parameter descriptions and a column for each format. If the M column contains an X, use the parameter for TIBCO ActiveEnterprise Message or XML wire format. Otherwise, leave the value as NULL.

Parameter	M	Description
subj	X	Subject on which the message is sent or received.
data		MInstance of the data. Alter directly if necessary.
pMapp	X	TIBCO Adapter SDK MApp structure for this MInstance.
messageFormat	X	Format of a message. The valid values are: • M_AERV_MESSAGE_FORMAT • M_XMLJMS_MESSAGE_FORMAT • M_XMLRV_MESSAGE_FORMAT For details about these values, see TIBCO Adapter For SDK documentation.

Exporting the JAR File

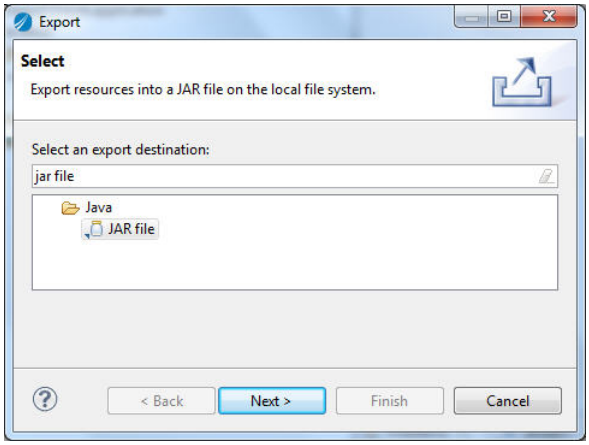
After you create a Java project and add a callout library and Java package and class, you can export the project by exporting the JAR file.

Procedure

1. Right-click the project you want to export, and then click **Export** from the pop-up menu.

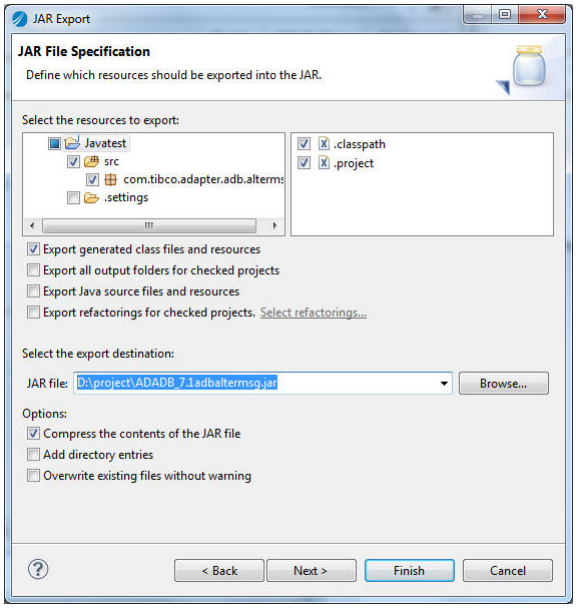
The Export window opens.

2. Type `jar file` in the **Select an export destination** field, and then click **Next**.



The JAR Export window opens.

3. Select the resources to export, and then type the export destination where you want to save the JAR file; or you can select a file repository by clicking **Browse**. After that, click **Finish**.



Replacing the Original JAR File

After exporting the JAR file, you have to use it to replace the original JAR file.

Procedure

1. Open the `adapter lib` directory.
   For example: *`TIB_ADADB_HOME`*`\lib`
2. Back up the original `adbaltermsg.jar` file to another directory.
3. Copy and replace the original JAR file with the exported JAR file in the `adapter lib` directory.

What to do next

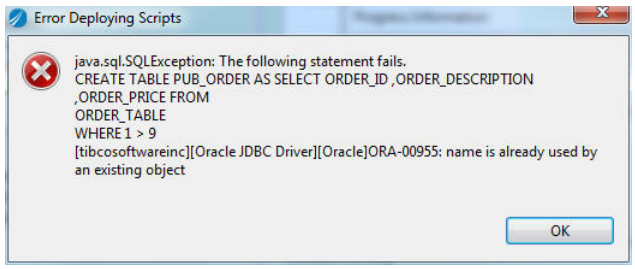
If you no longer use the user callout library, you have to restore the original JAR file back to the `adapter lib` directory.

Using Database Deployment and Cleanup Scripts

Changing an existing adapter configuration typically generates legitimate changes in a connected database. During this change process, the adapter creates a SQL script for changing the database objects and an associated cleanup script and stores them in the *TIB_ADADB_HOME*\sql directory. If the legitimate database changes result in error messages, you have to execute these scripts.
If one of the following conditions is met, an error message is displayed:

- The scripts created for generating legitimate database changes are not executed successfully.
- The scripts created for generating legitimate database changes are executed successfully, but an error occurs when these changes are being saved to the database.

The following figure shows an error message.



Running the Deployment Scripts

The generated SQL script for changes to database objects is stored in the *instance_name*.sql file in the *TIB_ADADB_HOME*\sql directory. You can modify this script to deploy changes to different database environments. For example, if the user schemas are different between the production and testing environments, you can change the schema names of the database objects in the script and deploy the changes accordingly.

Running the Cleanup Scripts

If an error occurs when you execute a deployment script, you can execute the associated cleanup script to delete the operation.

Procedure

1. In TIBCO Business Studio, close the project containing the adapter configuration that you changed.
2. Fix the errors that caused the database changes to fail.
3. If necessary, clean the old database configuration by executing the scripts created by the adapter. The scripts are in the *TIB_ADADB_HOME*\sql directory and are named *instanceId*.sql or *instanceId*.undo.sql, where *instanceId* is the instance ID of the adapter configuration you changed.
4. Reopen the project in TIBCO Business Studio, select the adapter configuration you were attempting to change when the errors occurred, and then save the project.

OS Authentication

This release of the adapter provides SQL Server OS Authentication on Windows. Microsoft SQL Server uses integrated login security to establish connections using this data source, regardless of the current

login security mode at the server. Any login ID or password supplied is ignored. The Microsoft SQL Server system administrator must have associated your Windows network ID with a Microsoft SQL Server login ID.

Runtime Schema

In the case where the schema names of the table objects are different between the development and production environments, you can use TRA properties to specify schema configurations.

For all database environments other than IBM UDB iSeries (AS400), you can use the following TRA properties:

- `adb.originalSchema`: specifies the design-time database object schema.
- `adb.runtime.schema`: specifies the runtime table objects schema. Note the following conditions:
  - If the `adb.originalSchema` property is specified, the adapter compares the `adb.originalSchema` property with the prefix of the table objects. If they are the same, the adapter replaces the design-time schema with this runtime schema. Otherwise, no action is taken.
  - If the `adb.orignalSchema` property is not specified, the adapter appends this runtime schema to all the table objects that do not have any schema specified at design time.
  - If the name of the `adb.runtime.schema` property contains a capital letter, the name must be with the double quote. For example: `adb.runtime.schema="Test"`
- `adb.runtime.publisherSchema`: specifies the runtime publishing table schema. The setting of this property overrides the setting of the `adb.runtime.schema` property. You can use the `adb.runtime.publisherSchema` property if you have a different schema for the publishing table than the other table objects, such as the source table.

These configurations do not apply to the referred object in the Publish by Reference mode. You have to specify the runtime schema for the referred object during design time, or in the `ADB_REF_OBJECT` column entry in the publishing table.

For the IBM iSeries (AS400) database, you can use the following TRA properties:

- `adb.as400.defaultLibrary`: specifies the default iSeries library that the adapter will access.
- `adb.as400.library`: specifies the name of the runtime iSeries library that the adapter will access.

Runtime Table Schema Configuration

You can specify the runtime schema module properties in the administrator console during the deployment of the adapter. In this way, the adapter can use these schemas at run time, which are different from the design-time schema configurations. By specifying the runtime schema module properties, you can easily move from one environment to another, most likely from the test environment to the product environment.

For the referred object in Publish by Reference mode, ensure that the schema and referred object name are set correctly in the publishing table, because the referred object name is fetched from the publishing table, not a property from the design-time project file or the TRA properties file.

Specifying Query Timeout

The adapter provides the query timeout function on all databases. When a database server is busy and the polling operation fails to be completed, the adapter returns an error instead of hanging.
For Publication Service, the adapter returns a null value in a polling operation and waits for the next polling operation. For Subscription Service, the adapter returns the connection dead error, and then reconnects to the database.

Compressing JMS Messages

A client can use TIBCO Enterprise Message Service (EMS) to compress the body of a message before sending the message to the server. JMS compression is specified for individual messages. It is especially useful when messages are to be stored on the EMS server, including persistent queue messages and topics with durable subscribers.

Enabling compression ensures that messages take up less memory space in storage and are handled faster by the EMS server. When JMS messages are compressed and stored, they are handled in the compressed form by the EMS server.

Compression normally takes time, and therefore the time to send or publish and receive compressed messages is generally longer than the time to send the same messages that are not compressed.

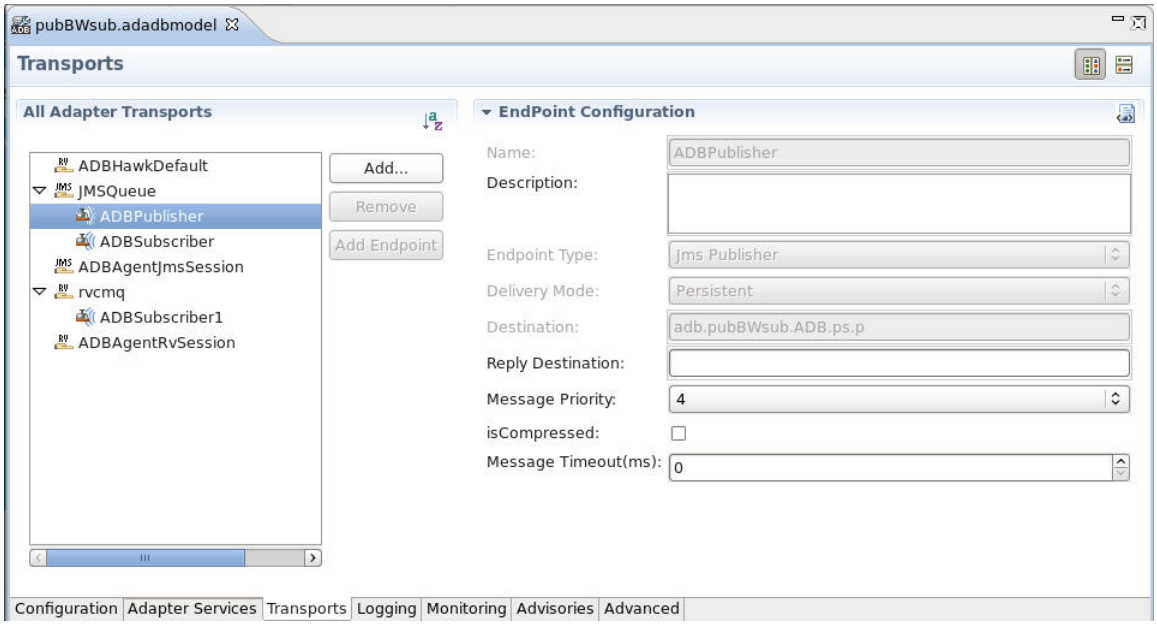
When you configure JMS compression, note the following conditions:

- JMS compression applies to adapter services with the JMS transport type.
- Currently, JMS compression is not supported for Subscription Service and Remote Procedural Call (RPC) Service in the Request-Reply mode.

Publication Service

The methods of enabling JMS compression vary depending on the adapter service types. TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) provides three ways to set JMS compression for Publication Service.

- In TIBCO Business Studio, select the **isCompressed** check box in the **Configuration** tab of the JMS Publication endpoint.



- In the `adbagent.tra` file, add either of the following properties:
  - `tibco.sdk.session.jmsCompressed` *`session_name`* Specifies a list of JMS session names. All the JMS endpoints under the specified session list send compressed JMS messages to the EMS server.
  - `tibco.sdk.endpoint.jmsCompressed` *`endpoint_name`* Specifies a list of JMS endpoints. All the specified JMS endpoints send compressed JMS messages to the EMS server.

Separate multiple session names or endpoint names with vertical bars (|).

- In the `adbagent.tra` file, set the `adb.jmscompress=<on/off>` property or the `adb.<publisher_service_name>.jmsProperties JMS_TIBCO_COMPRESS <on/off>` property to `on`. The default value is `off`, indicating that messages are not compressed.

When you enable JMS compression, the `adb.<publisher_service_name>.jmsProperties JMS_TIBCO_COMPRESS=<on/off>` property is only used for Publication Service.

When JMS compression is enabled by using any of the preceding methods, published messages will be compressed. For example, when a Publication endpoint is set to be compressed in TIBCO Business Studio, the messages are compressed even if no related TRA property is set or a related TRA property is set to `off`.

Remote Procedural Call (RPC) Service

To enable JMS compression for both standard RPC Service and custom RPC Service, add either of the following properties in the `adbagent.tra` file:

- `tibco.sdk.session.jmsCompressed` *`session_name`*
- `tibco.sdk.endpoint.jmsCompressed` *`endpoint_name`*

The RPC server replies to JMS messages depending on whether the request messages are compressed. If the request messages are compressed, the RPC server replies to the compressed JMS messages. Otherwise, the RPC server replies to messages according to the configurations of the TRA properties.

Configuring RVCMQ Backlog Size

The Rendezvous Distributed Queue (RVCMQ) scheduler receives inbound messages and assigns them to the worker. The scheduler stores tasks in a message queue.
You can limit the maximum size of a message queue with either of or both of the following adapter properties:

- `adb.taskBackLogLimitInBytes`: This property specifies the maximum size of the scheduler task queue by the number of bytes. This value must be an integer. The default value is unspecified.
- `adb.taskBackLogLimitInMessages`: This property specifies the maximum size of the scheduler task queue by the number of messages. This value must be an integer. The default value is unspecified.

When the number of task messages in a queue exceeds either of the preceding limits, TIBCO Rendezvous deletes subsequent inbound task messages.

If neither of the preceding properties is specified, the size of the scheduler task queue will not be limited.

The RVCMQ backlog size configuration only applies to Subscription Service and Request-Response Service, which have the RVCMQ or RVDQ transport type. This configuration does not apply to Publication Service, because Publication Service does not have this transport type.

Implementing Refresh Connection

After extensive use of an adapter, the performance of the adapter might deteriorate. If you restart the adapter, its performance recovers. To resolve this issue, TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) supports the new refresh connection function. The connection between the adapter and database can be refreshed after a specified time elapses.
To specify a time for a refresh connection, add two parameters in the adbagent.tra file:

- `adb.connDBRefreshInterval` *`minute`*: After this configured time elapses, the adapter triggers a refresh connection timer, which is an adapter configuration-level parameter. The default value is *300* minutes.
- `adb.connDBRefresh` *`on/off`*: This is an adapter configuration-level parameter, which indicates that every service under an adapter configuration can refresh a connection after a configured time elapses. Its default value is `off`. An adapter configuration has only one timer while a service does not have a timer. When the `adb.connDBRefresh` property is set to `on`, a timer is created.

Error Handling by Subscription Service

When Subscription Service inserts data into a database and an error occurs, the subscriber endpoint rolls back all previous operations on the database to the original status. Subscription Service handles error data in two ways: normal error handling and bulk insert error handling.

Normal Error Handling

Principles for normal error handling are as follows:

- If you do not set a bulk insert size, the subscriber adapter inserts error data into either the exceptions table or opaque exceptions table.
- If you set both a bulk insert size and a batch commit size, the subscriber adapter inserts data in the following ways:
  - If all the data in a bulk is processed successfully, the subscriber adapter inserts all the data in the bulk into the destination table.
  - If not all the data in a bulk is processed successfully, the subscriber adapter inserts all the data, including the error data in the bulk into either the exceptions table or opaque exceptions table.

Bulk Insert Error Handling

Bulk insert error handling is a feature that supports bulk insert operations. This feature improves data accuracy and processing efficiency as well as preventing data failure in the bulk insert operation.

The bulk insert error handling feature restrictions are listed as follows:

- Bulk insert error handling applies to Subscription Service only.
- Bulk insert error handling can be implemented only when the `java.sql.BatchUpdateException` exception occurs.
- Bulk insert error handling supports the Oracle database only.
- Bulk insert error handling does not support Parent-Child message or Group message.

Without the bulk insert error handling feature, the agent inserts all rows (both correct and incorrect) into the exceptions table when Subscription Service performs the bulk insert operation.

For example, the agent attempts to perform a bulk insert of 500 rows to the destination table. If the first 300 rows are inserted successfully but the last 200 rows are not, the agent inserts the successful 300 rows into the destination table and the unsuccessful 200 rows into the exceptions table.

For information about the prerequisites of using bulk insert error handling, see Prerequisites of Using Bulk Insert Error Handling.

For information on how to implement bulk insert error handling, see Working with Bulk Insert Error Handling.

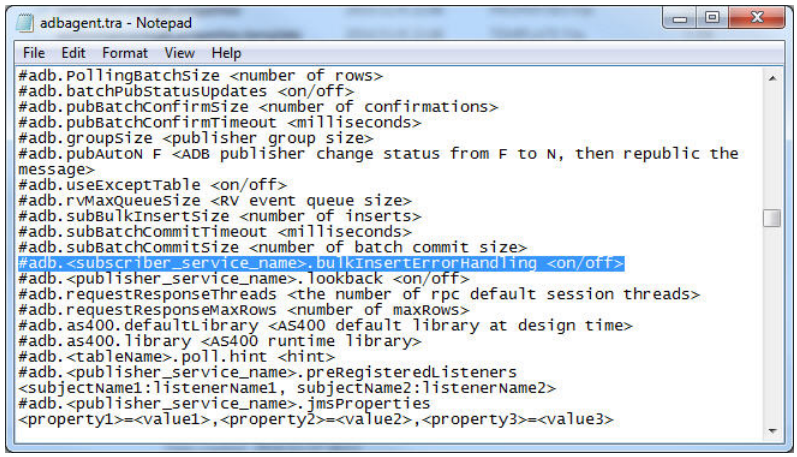
Prerequisites of Using Bulk Insert Error Handling

Before using the bulk insert error handling feature, you have to add a necessary parameter and then configure Subscription Service.

Adding a Parameter

To enable bulk insert error handling, add the `adb.<sub-service-name>.bulkInsertErrorHandling <on/off>` parameter to the `adbagent.tra` file in the *TIB_ADADB_HOME*`\bin` directory.

This parameter indicates whether the bulk insert error handling feature in Subscription Service is enabled or not. Its default value is off. You can refer to the `adbagent.tra` file for the parameter format.



To enable the bulk insert error handling feature, change the <*sub-service-name*> variable to the name of Subscription Service, and set the value to `on`. For example,

```
adb.ADBSubscriber.bulkInsertErrorHandling on
```

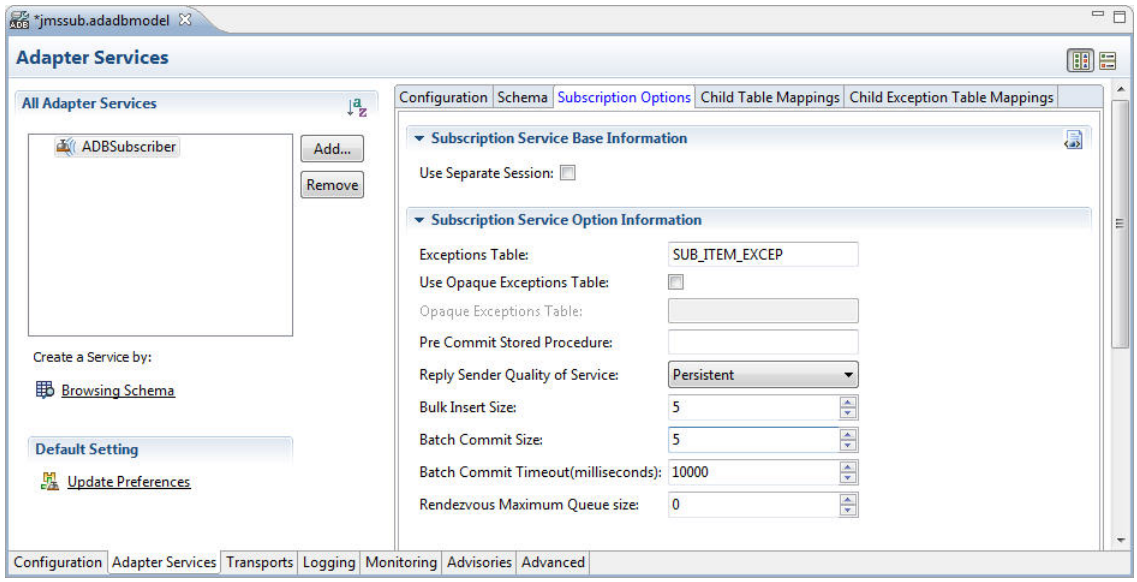
After the `adb.<sub-service-name>.bulkInsertErrorHandling` property is set to `on`, all correct data is inserted into the destination table, error data is inserted into the exceptions table or opaque exceptions table, and all the data is confirmed. However, if the `adb.useExceptTable` property is set to `off` in the `adbagent.tra` file and no opaque exceptions table is specified, error data cannot be inserted into the exceptions table or opaque exceptions table. As a result, the adapter does not confirm the error data. Instead, the adapter inserts all the correct data into the destination table and just confirms the correct data.

Configuring Subscription Service

After creating an adapter configuration and Subscription Service, you have to configure Subscription Service with bulk insert error handling.

If you want to use separate sessions, perform the following steps:

1. In the **Subscription Options** tab, select the **Use Separate Session** check box.
2. Set values in the **Bulk Insert Size**, **Batch Commit Size**, and **Batch Commit Timeout(milliseconds)** fields. Then click **Save**.



You can also configure batch commit for Subscription Service by setting the `adb.subBatchCommitTimeout` and `adb.subBatchCommitSize` properties in the `adbagent.tra` file. For details about these two properties, see Adapter Properties.

If you do not want to use the default session, perform the following steps:

1. In the **Subscription Options** tab, enter a value in the **Bulk Insert Size** field.

   Ensure that the **Use Separate Session** check box is not selected.

2. Set either of the following properties in the `adbagent.tra` file to configure batch commit:

   - `adb.subBatchCommitTimeout` and `adb.subBatchCommitSize` (with precedence)
   - `adb.<default session name>.subBatchCommitTimeout` and `adb.<default session name>.subBatchCommitSize`

   For details on how to configure batch commit without using separate sessions, see Configuring Batch Commit for Subscription Service.

Working with Bulk Insert Error Handling

To familiarize you with the work flow of bulk insert error handling, an example is provided. In this example, the adapter attempts to execute a bulk insert that contains the following five records. Here, `correct` means a record without errors and `wrong` means a record with errors.

- M1 (correct)
- M2 (wrong)
- M3 (correct)
- M4 (correct)
- M5 (wrong)

Based on the prerequisites in Prerequisites of Using Bulk Insert Error Handling, the following process shows how bulk insert error handling handles these five records:

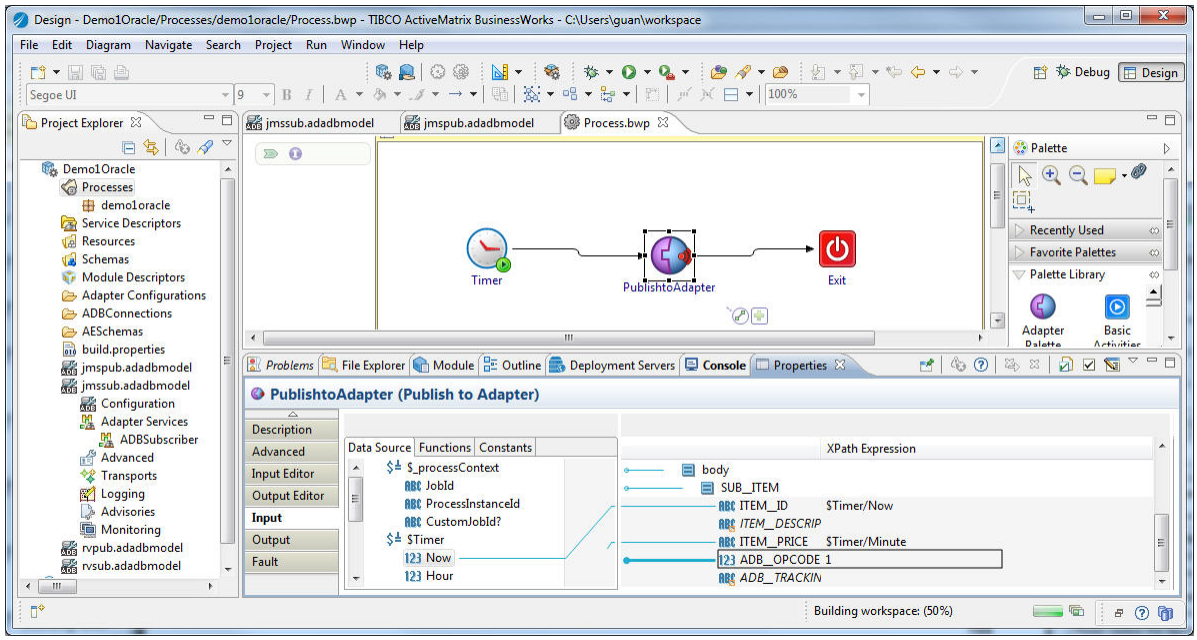
1. When the bulk commit size arrives, the adapter starts to commit this bulk insert into the database.
2. When the adapter hits the error message (M2), bulk insert error handling inserts this error message into the exceptions table.

3. Then the adapter handles the remaining records, and when it hits the error record (M5), bulk insert error handling inserts this error record into the exceptions table.
4. Check the database after processing the bulk insert operation, you can see that all error records (M2 and M5) are in the exceptions table, and the correct records (M1, M3, and M4) are in the destination table.

If an adapter configuration contains multiple services of Subscription Service, each service handles an exception logic based on its own configurations. Therefore, the exception logic of each service might be different.

Using Opcodes to Set Operations for Subscription Service

In TIBCO Business Studio, when you use a Publish to Adapter activity to publish messages to Subscription Service, you can set the **ADB_OPCODE** field to specify insert, update, delete, and upsert operations for Subscription Service.
In the Publish to Adapter **Input** tab, click **aePubInputType** > **body** > *Table_Name* to find the **ADB_OPCODE** field, as shown in the following figure.



To specify an operation for Subscription Service, set the field to any of the following values:

- 1: indicates insert.
- 2: indicates update.
- 3: indicates delete.
- 4: indicates upsert. (If the primary key exists in the table, 4 indicates update; if the primary key does not exist, 4 indicates insert.)

Configuring Publish by Reference Object

When source data is stored in a view or a different database object, you can publish the data by using a referred object. Publish by Reference Object is an extension of the Publish by Reference feature.

For a description of the Publish by Reference mode, see "Publish by Reference" in Selecting a Storage Mode.

In both cases, only key values from the source table are stored in the publishing table. However, publishing by reference object is used when a row changes in the source table and the associated trigger

fires, the adapter fetches data from the referred object, rather than the source table. The name of the referred object is stored in the ADB_REF_OBJECT column in the publishing table. For a description of this column, see Publishing Table.

For best results use Publish by Reference Object when a view provides the most efficient access to source data, for example, when many levels of nesting exist between a parent table and a child table.

To use Publish by Reference Object, complete the following tasks:

1. Specifying a Referred Object
2. Changing Repository Objects for Parent-Child Table Relationships

Publishing a Message with a Referred Object

In the following example, the publisher adapter is configured to publish source data from the CUSTOMER table and its child table, EXTERNAL_ORDER_DETAILS. The relevant key columns are CUST_ID and ORDER_ID.

The publishing table is created with necessary adapter columns, as well as the CUST_ID and ORDER_ID columns. When a row in the CUSTOMER table is modified, the trigger fires, populating columns and copying the CUST_ID and ORDER_ID values, as well as the name of the referred object, ORDER_VIEW, to the publishing table. When the adapter polls the publishing table, it detects the new row. The adapter then selects the order and customer data from the referred object, ORDER_VIEW, using the CUST_ID and ORDER_ID values along with the view name found in the publishing table. Then the message is published.

Table to Record Sequence Numbers (DB2 on iSeries)

The palette attempts to create a table (if it does not exist) to record sequence numbers.

A table is created by using the following statement:

```
CREATE TABLE library.ADB_SEQTAB (
   PUB_TABLE VARCHAR(64)  NOT NULL,
      ADB_SEQ NUMERIC(20),
      constraint ADB.ADB_SEQTAB_KEY primary key (PUB_TABLE))
```

The table must be journalled.

You can create this table manually. If journalling is automatically turned on, the palette creates the table automatically.

Specifying a Referred Object

To configure Publish by Reference Object, you must first specify a referred object for the source table and designate a column of the referred object as the key.

Procedure

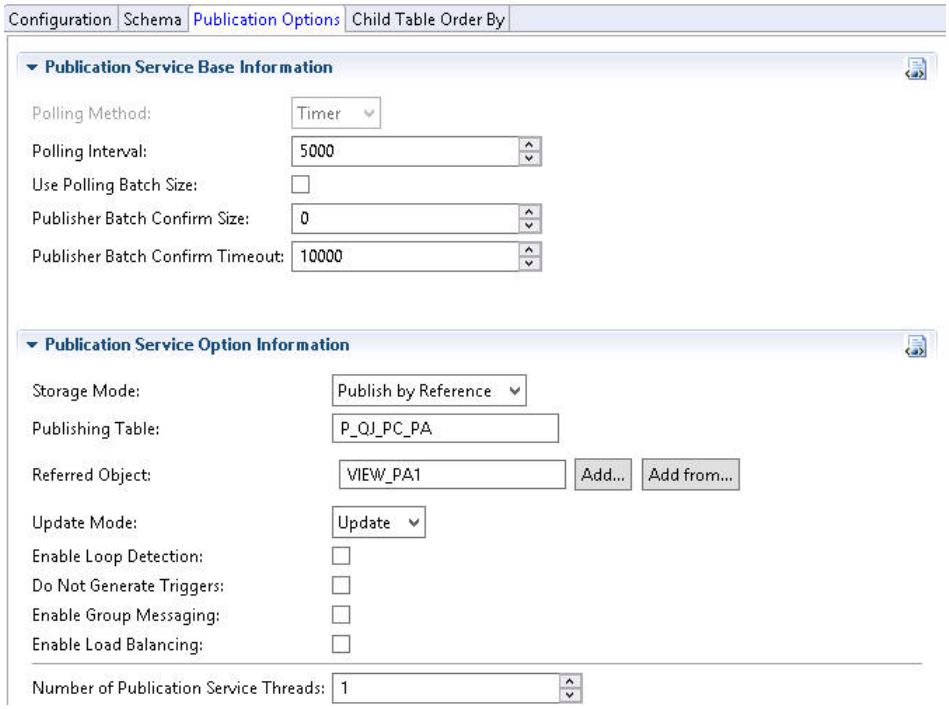
1. Add a source table with a parent-child relationship and create a view for Publication Service.
2. Click the **Publication Options** tab and specify the following options:
   a) Select **Publish by Reference** from the **Storage Mode** list.
   b) Type the table name to be used for storage in the **Publishing Table** field. A common practice is to use the source table name prefixed by P_.
   c) Type the name of the view or a different database object to select source data from the **Referred Object** field.

      To select source data from a list of tables in the current user schema, click **Add**. To select an object from a different schema, click **Add From**. For example, select VIEW_PA1.

If you select a table from a different schema, ensure that the primary key in the selected table and the primary key in the source table are the same.

d) Select a method for updating tables from the **Update Mode** list.

For details about the modes, see the description about the **Update Mode** field in Publication Options Tab.

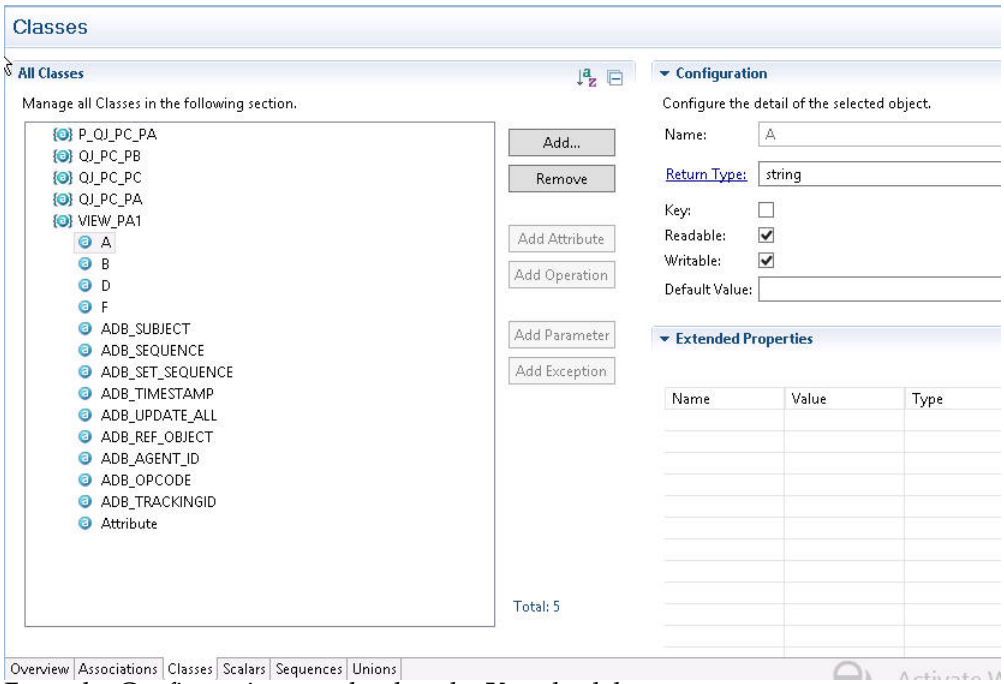


3. Save the adapter configuration.
4. Designate a column of the referred object as the key.

You must explicitly designate a key column or substitute key column for the referred object, because a referred object has no external designation of the key column.

a) Click the **Configuration** tab of Publication Service.
b) In the tab, click the **Class Reference** link in the Schema panel to open the AE schema used by the adapter service.
c) In the **Classes** tab, expand the referred object VIEW_PA1 in the All Classes panel and select the column you want to designate as the key.

The configuration fields of the selected column are displayed in the Configuration panel.



d) From the Configuration panel, select the **Key** check box.

5. Save the AE schema.

Changing Repository Objects for Parent-Child Table Relationships

If your Publication Service includes parent-child table relationships, you must add a sequence and association to the metadata stored in the repository.

For details, see:

- Adding a Sequence
- Adding an Association

Adding a Sequence

If you use parent-child table relationships for Publication Service, you have to add a sequence to the metadata stored in the AE schema.

Prerequisites

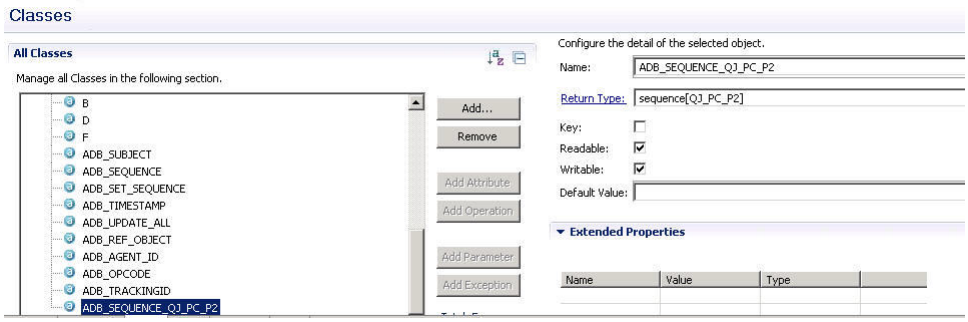
A referred object is specified in Publication Service.

Procedure

1. In the **Configuration** tab of the service, click the **Class Reference** link to open the AE Schema file in the editor.
2. Click the **Classes** tab, and select the referred object from the All Classes panel.
3. Click **Add Attribute**. In the Configuration panel, specify the following fields for the attribute:
   - **Name**: Type **ADB_SEQUENCE_*child_table_name*** in the **Name** field.
   - **Return Type**: Click **Browse**. In the Schema Selection window, select the schema of the adapter configuration. In the displayed list, select the sequence for the child table. Click **OK**.
   - **Readable**: select this check box.

- **Writable**: select this check box.

4. Save the AE schema.
   The sequence is added to the metadata for the reference object.



Adding an Association

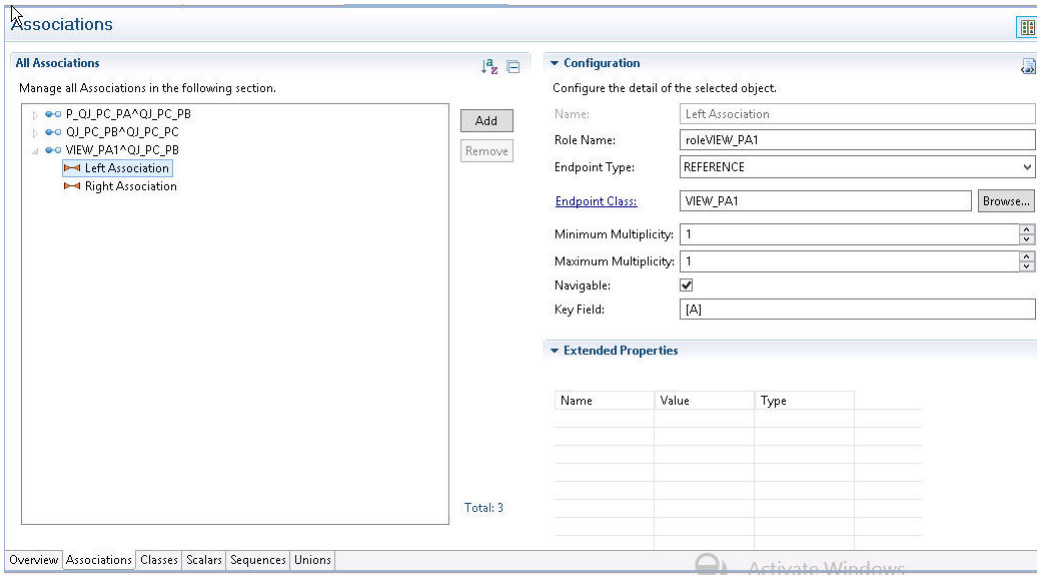
If you use parent-child table relationships for Publication Service, you have to add an association to the metadata stored in the AE schema.

Prerequisites

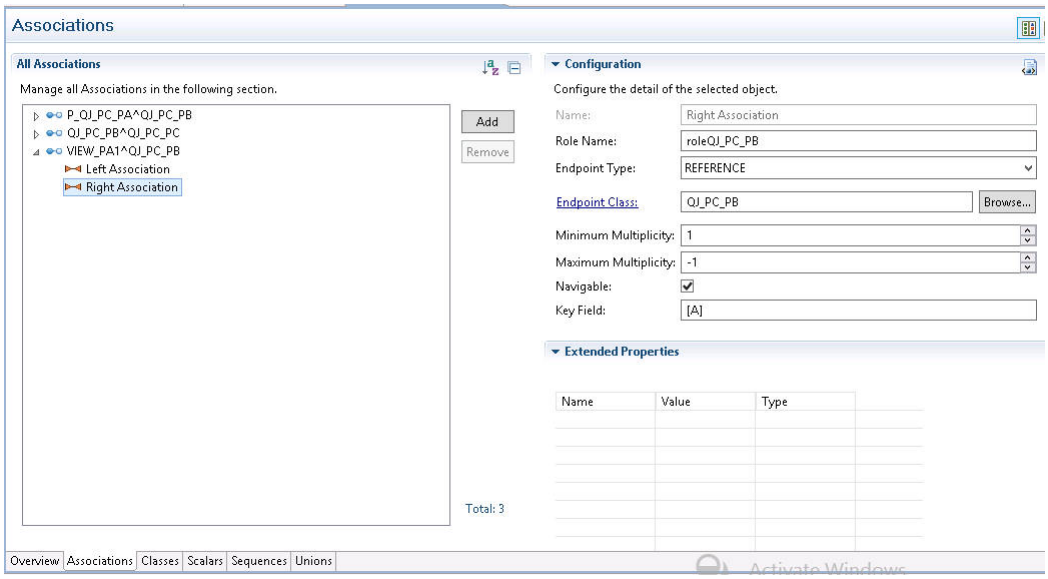
A referred object is specified in Publication Service.

Procedure

1. In the AE Schema editor, click the **Associations** tab.
2. Select a parent-child association. Several associations are displayed in the All Associations panel, at least one for the parent table with a child table.
3. In the Configuration panel, replace the name of the publishing table with the name of the referred object in the **Name** field.
4. Expand the parent-child association in the All Associations panel and select **Left Association**.
5. In the Configuration panel, specify the following fields and keep the rest fields unchanged:
   - **Name**: Replace the name of the publishing table with the name of the referred object.
   - **Endpoint Class**: Click **Browse**. In the Schema Selection window, select the schema of the adapter configuration. In the displayed list, select the referred object, VIEW_PA1. Click **OK**.



6. In the All Associations panel, select **Right Association**.
7. In the Configuration panel, repeat Step 5 to specify the right association.



8. Save the AE schema.

Setting Maximum Rows in the Operation Level

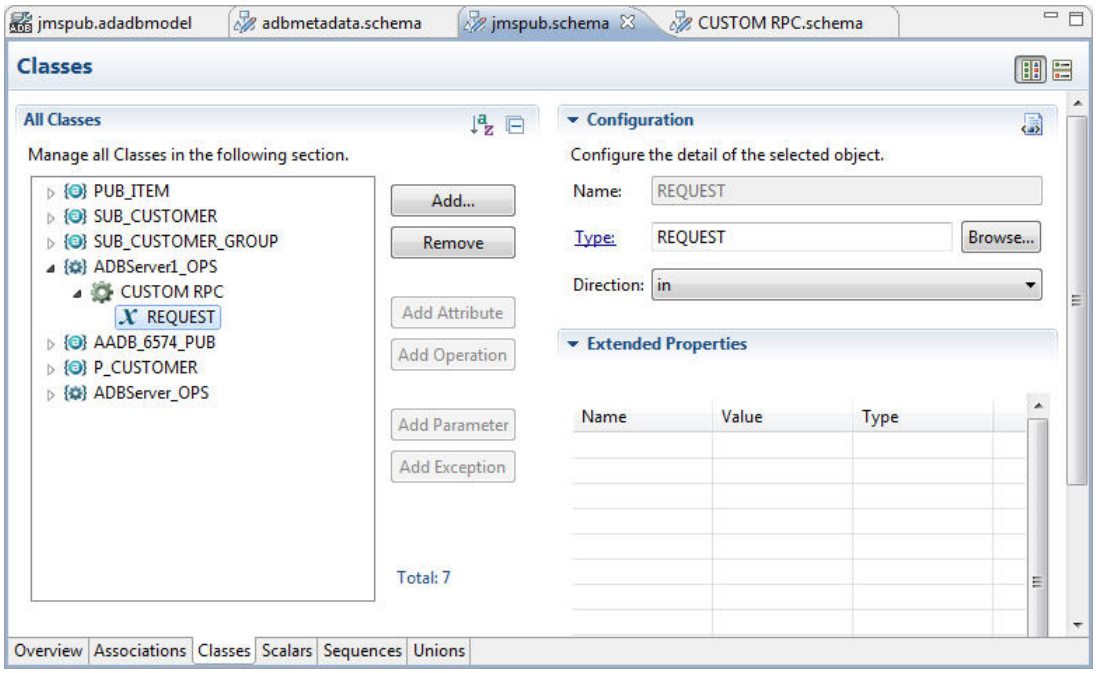
For custom RPC Service, you can specify the maximum number of rows to be fetched in the operation level. At run time, the adapter fetches the number of rows according to the value set in the operation level instead of the service level.

Prerequisites

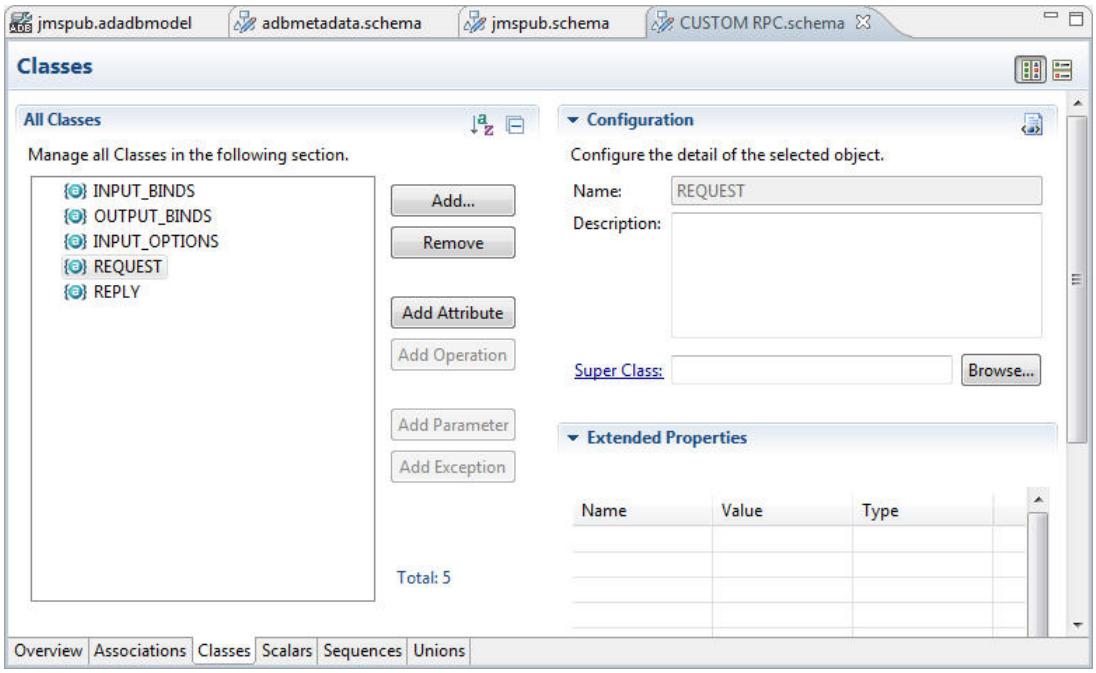
A stored procedure, function, or package is selected for custom RPC Service.

Procedure

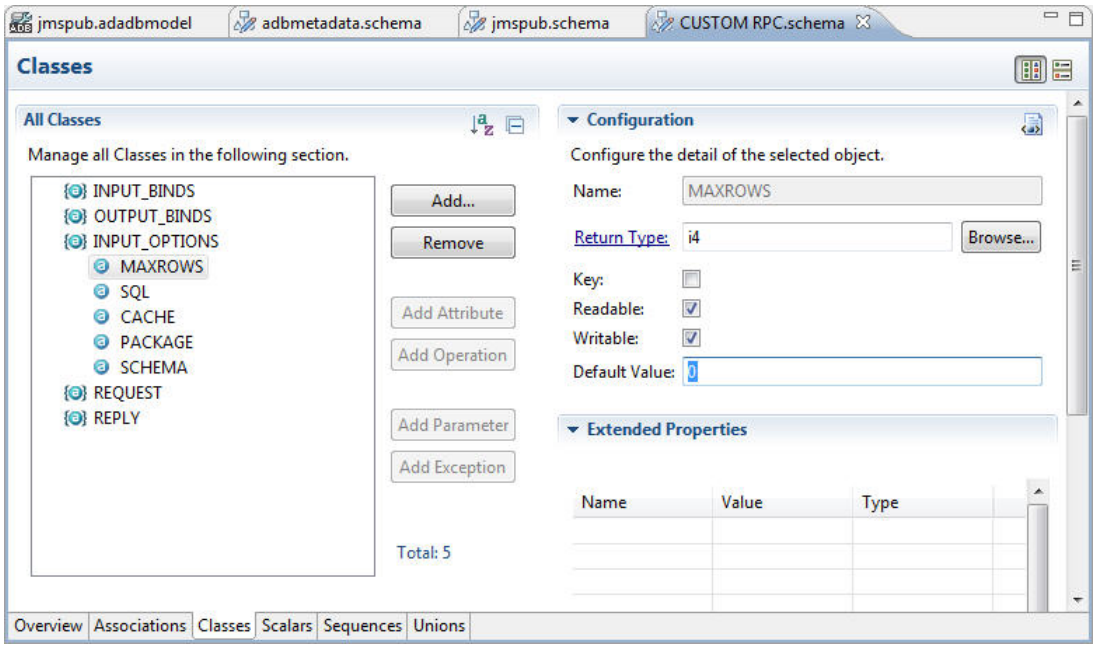
1. In the **Configuration** tab of the adapter service, click the **Class Reference** link.
2. In the opened AE schema, click the **Classes** tab.
3. In the All Classes panel, click **procedure**_*aeclass_name* > *user_specified_procedure_name* > **REQUEST**.
   The Configuration panel of the selected object is displayed.



4. In the Configuration panel, click the **Type** link.
The **Classes** tab of the selected procedure is displayed.



5. In the All Classes panel, click **INPUT_OPTIONS** > **MAXROWS**.
6. In the Configuration panel, specify a value in the **Default Value** field to set the maximum number of rows to be fetched.



Using Hints (Publication Service Only)

Hints help improve the performance of your queries. When Publication Service performs a poll operation to fetch data from a table, using hints greatly enhances the query.

Hints are only supported in Oracle and SQL Server databases.

To use hints, you can add the following line to the adapter TRA properties file:

`adb.`*`table_name`*`.poll.hint` *`hint_value`*

where, *table_name* is the name of your table and *hint_value* is the hint.

Examples

The ways of using hints vary depending on different databases you use.

The following examples show the ways of using hints based on different databases:

- To force an index scan when polling in an Oracle database, set the property as follows:

  ```
  adb.p1.poll.hint /*+INDEX(P1,P1_INDX)*/
  ```

  where `p1` is the publisher table, and `P1_INDX` is the index created on the publishing table.

  The adapter processes this select query on the publishing table:

  ```
  SELECT /*+INDEX(P1,P1_INDX)*/ * FROM P1 WHERE ID = ?
  ```

- To use the NOLOCK hint when polling in an SQL Server database, set the property as follows:

  ```
  adb.p1.poll.hint WITH(NOLOCK)
  ```

  where `P1` is the publisher table.

  The adapter processes this select query on the publishing table, but will not issue shared locks or honor exclusive locks:

  ```
  SELECT * FROM P1 WITH(NOLOCK) WHERE ADB_L_DELIVERY_STATUS ='N'
  ```

- To force an index scan when the adapter fetches records from a child table, set the property as follows:

  ```
  adb.C1.poll.hint /*+INDEX(C1,C1_INDEX)*/
  ```

where C1 is the child table, and C1_INDEX is the index created on the child table.

The adapter processes this select query on the child table:

```
SELECT /*+INDEX(C1,C1_INDEX)*/ * FROM C1 WHERE ID = ?
```

Configuring the Internal Message Queue

When TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) interacts with a TIBCO environment, the adapter first retrieves messages from a transport queue and caches them in the internal message queue for processing. You can configure the internal message queue to cater to the performance of the adapter.

Internal Message Queue Size

When the adapter retrieves messages from a transport queue and puts them in the internal queue, if the transport queue contains an excessively large number of messages and the size of the internal queue is not set or set to an excessively large value, the `OutOfMemoryError` error occurs.

To prevent the `OutOfMemoryError` error, you can set the `adb.<sessionName/serviceName>.ADBQueueSize` property in the `adbagent.tra` file to restrict the internal queue size.

However, you must set the internal queue size to a reasonable value; if the internal message queue size is set to an excessively small value, the adapter only handles the specified size number of messages at a time and has to wait for the timeout period of 500 ms (default timeout) to handle subsequent messages, which prolongs the message handling process. Setting of the `ADBQueueSize` property depends on the size of a single message and the value of the `tibco.env.HEAP_SIZE` property, especially when an excessively large number of records are sent to Subscription Service and the single message size is in the Mbit-level.

For details about the `adb.<sessionName/serviceName>.ADBQueueSize` property, see Adapter Properties.

Internal Message Queue Timeout

To compensate for the improper setting of the `ADBQueueSize` property, you can set the `adb.<sessionName/serviceName>.ADBQueueTimeout` property to control the timeout for processing messages. For example, if the internal queue size is set to an excessively small value, it is good practice to set the value of the `adb.<sessionName/serviceName>.ADBQueueTimeout` property to less than `500 ms` (the default value) in the `adbagent.tra` file. This configuration helps improve the performance of the adapter.

Additionally, to ensure better performance of the adapter, it is good practice to set the value of the `ADBQueueTimeout` property based on the message size. For byte-level messages, set the value to a smaller value such as 20 ms; for Mbit-level messages, set the value to a larger value such as 200 ms.

For details about the `adb.<sessionName/serviceName>.ADBQueueTimeout` property, see Adapter Properties.

Frequently Asked Questions

When you configure and deploy the adapter project, possibly you encounter some problems. Those frequently asked questions are listed and their answers are provided to facilitate your configuration and deployment. The frequently asked questions involve both general questions and Request-Response Service questions.

General Questions and Answers

General questions are the questions that you possibly have when configuring and deploying the adapter, such as how to find the version number of an adapter configuration and how does the exceptions table work, are listed, and their answers are provided.

The general questions and their answers are listed as follows:

Q1: How to find the version number of an adapter configuration?

A1: A banner is displayed when an adapter configuration starts. The banner lists component versions for the adapter and for the TIBCO Adapter SDK software. You can use this information to diagnose compatibility issues or to report any problem details to Customer Support.

Q2: Why is a database trigger error not logged in the exception table?

A2: When you use an adapter configuration as a publisher, if an error occurs in the database trigger that is used to copy data from the source table to the publishing table, the database trigger error will not be logged in the exception table for the subscriber adapter.

Q3: How does the adapter react if a database connection is lost and the database is then restarted, and does it automatically try to reconnect?

A3: If TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) detects that it has lost its database connection, it tries to connect to the database in three times by default. If the connection is not restored after three times, the adapter exits. You can configure the adapter for automatic reconnection times and intervals (see Reconnection Configuration). Alternatively, TIBCO Hawk rules can be written to detect a disconnection and restart the adapter whenever a disconnection occurs.

Q4: Does an incoming message have to contain all the columns that are defined in the destination table?

A4: The incoming message does not have to contain all the columns defined in the destination table. You can configure the adapter to expect only a subset of the columns, defined in the repository. The adapter is driven from the subscribing class description and will iterate through the attributes in the class definition for the subscribing table and specifically look for those attributes in the incoming messages. It inserts NULLs for the attributes that the adapter is expecting but does not find in the message. If the subscribing table contains more columns than the subscribing class lists (set when adding a subscription), those extra columns will get whatever default values were specified during the table creation.

Q5: Can an existing table be used as the publishing table?

A5: No. TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) requires additional columns in the publishing table. Even when every field in a table is published, a separate publishing table is required.

Q6: Does a published TIBCO Rendezvous message have to contain every field in the publishing table?

A6: Yes. You can control which fields are copied to the publishing table by configuring the adapter and by changing the publication trigger to publish a subset of rows. You can also append additional fields to a message or drop a message based on some criteria using the user callout library. For more information on the user callout library, see Using the User Callout Java Library.

Q7: If multiple updates occur between polling intervals, are updates published in multiple TIBCO Rendezvous messages or in a single large message?

A7: If you are using `publish by value`, a TIBCO Rendezvous message is created for each individual update. If you are using `publish by reference`, that operation will get the last update.

Q8: Can earlier entries be deleted in the publishing table?

A8: Yes. When a row is published, the value of the **ADB_L_DELIVERY_STATUS** field in the publishing table changes to either C (complete) or F (failed). You can write a trigger in your publishing table that deletes the row when the delivery status changes to C or F.

You can also publish data directly from the source table by configuring the adapter configuration to publish by reference. A publishing table is created, but the publishing table contains only required fields and key fields of the source table.

Q9: How does the exception table work?

A9: Before starting an adapter configuration, you must set the `adb.useExceptTable` property in the adapter's properties file to `on` and specify an exception table when configuring the adapter Subscription Service. If an error occurs when you insert data into the destination table, the error will be inserted into the exception table. The transaction will be committed and a confirmation will be sent back for the message (RVCM delivery). If the insertion into the exception table also fails, an error message will be displayed and the adapter configuration will terminate.

Q10: Can an adapter configuration be used to replicate binary types, such as BLOB?

A10: Only limited support is available for binary large object (BLOB) data types. Oracle LONG and LONG RAW types are supported in top-level tables when the adapter is configured to publish by reference. Oracle BLOB and CLOB data types are supported.

Q11: Can an adapter configuration write to tables that belong to a database account different from that used by the adapter?

A11: Yes. A source table or destination table can belong to a different database user than the default account created in the `create_user.sql` script. For more information on referencing external schemas, see Referencing an External Schema.

Q12: Can a publisher adapter and a subscriber adapter use different projects?

A13: Yes, unless the publisher adapter is configured to use parent-child relationships.

Q13: Can TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) guarantee exactly once delivery of messages over RVCM?

A13: Exactly once delivery of messages over RVCM is not currently guaranteed. The same quality of service that RVCM provides is supported, which is at least once. To ensure exactly once delivery requires combining the messaging operations and the database operations in a single atomic transaction, which is not supported in RVCM.

Q14: What guarantees does the adbagent make with regards to the order of database operations? For example, is it guaranteed that for a given table, modifications are made in the same order that they were made to the source database? What guarantees are made for operations across different tables?

A14: TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) guarantees that for database operations that are published on the same subject, the order of the operations is preserved. Usually, this applies to database operations made to one table. It does not usually apply to database operations across different tables.

Q15: When you use RVCM for delivery, at what point does the subscribing adapter acknowledge an incoming message?

A15: A subscriber adapter confirms the message only after the database operation is committed. If an error occurs and no exception table is used, the database operation is rolled back and no confirmation is sent. If an error occurs and an exception table is used, the insert to the exception table is committed and the message is then confirmed.

Q16: By maintaining the publishing tables, all changes to the source table can be captured. If, however, a failure occurs between the point of publishing a message using RVCM and updating the publishing table, will the adapter republish the message that has already been sent?

A16: Yes, the message will be republished and the subscriber will have to deal with the duplicate message.

Q17: For the subscriber adapter, if a failure occurs after doing a database update and before sending an acknowledgment back to the publisher adapter, can the restarted adapter configuration be prevented from redoing the update operation?

A17: No. This will cause a duplicate insert.

Q18: For a certified subscriber adapter, what happens if an insert exception occurs and no exception table is specified?

A18: If the `tibco.clientVar.DirTrace` property is specified in the adapter's properties file when the adapter is started, exception handling information is written to the log file and the adapter configuration continues to run. Since the insert could not be performed, the ADB_L_DELIVERY_STATUS publishing table column has a value of P for the message.

Q19: How to preregister non-TIBCO ActiveMatrix Adapter for Database subscribers, such as custom adapters, to ensure that no messages are lost?

A19: Specify the CM name of the listener's RVCM session.

Q20: The session names are automatically created. Can these names be changed without confusing TIBCO ActiveMatrix Adapter for Database, so that we can use a standard naming convention throughout the entire integration process?

A20: No, the session names are fixed and used by TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) internally. They cannot be altered.

Q21: What's the proper way to permanently remove an adbagent subscriber when you use CM?

A21: Two ways are available: One way is to completely remove the publisher's ledger and to change all 'P' records back to 'N'. Then restart the publisher. The other is using the TIBCO Hawk method, unRegisterListener(), which unregisters a CM subscription. This is the proper way to remove the adbagent subscriber as a CM listener.

Q22: Is it possible to run an adapter configuration using a remote TIBCO Rendezvous daemon?

A22: Yes. Change the default settings for network, service, and daemon parameters for the adapter using TIBCO Business Studio.

Q23: Is it possible to run two adapter configurations on the same machine?

A23: It's possible to run multiple configurations of the adapter on the same machine if each adapter configuration has a unique name. If both configurations use TIBCO Rendezvous certified messaging, each must use a different RVCM session.

Q24: Can an adapter configuration collate information from several database tables to send as a single TIBCO Rendezvous message, or can an adapter only publish data from a single table, in the format defined by that table?

A24: You can publish related tables in either of the following ways:

- Set the `adb.publishChildData` property to `on` in the adapter's properties file. When there are insertions into parent table, the adapter will publish parent rows and the corresponding child rows using TIBCO ActiveEnterprise or XML format.
- Combine several tables into one table using a trigger, and then publish from the combined table.

When an adapter configuration publishes or subscribes to a message, you can change the message by using the callout library. See Using the User Callout Java Library for details.

Q25: How to fix an adapter that hangs problem when my Sybase transaction log becomes full?

A25: When a Sybase transaction log becomes full, if the `abort trans on log full` database setting is set to `false`, your application will hang instead of printing a transaction log full error.

To resolve this issue, type the following command:

```
sp_dboption dbname, "abort tran on log full", true
```

Q26: On a Red Hat platform, if I use the Rendezvous transport, a BusinessWorks process fails to be run with the "no tibrvnativesd in java.library.path" error. How to resolve this issue?

A26: Before you run a BusinessWorks process, you can add the following environment variables:

- `export` *`RV_HOME`*`={`*`RV_HOME`*`}`
- `export LD_LIBRARY_PATH=$`*`RV_HOME`*`/lib$LD_LIBRARY_PATH`

Request-Response Service Questions and Answers

You might encounter questions specifically when you are working on Request-Response Service. The frequently asked questions are listed, and their answers are provided.

The Request-Response Service questions and their answers are listed as follows:

Q1: When I use Request-Response Service, can an INSERT statement only with field values (without field names) be sent to improve the performance of the application?

A1: Yes this is supported. Your application can also send INSERT statements without the binds.

Q2: Can an application send UPDATE statements to a subscriber adapter only with fields that are being updated?

A2: Yes, this is supported.

Q3: Does an adapter configuration send responses back as a large message containing all rows or as chunks of messages?

A3: The adapter sends results back to an application as one large message.

Q4: What is returned to the application if a failure occurs when the adapter performs an insert or update?

A4: If an error occurs while the adapter is processing a request, an error code and description is returned to the application. In the case of success, a result set and row count is returned to the application.

Using Module Properties

When you configure a project, you can use module properties. Module properties provide an easy way to set default values for use throughout your project. You can use the variable substitution mechanism to replace the input values in a project with the values of predefined module properties.

A number of module properties are predefined in TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio). For details, see Predefined Module Properties.

Module properties used in the adapter are called Global Variables when you use TIBCO Designer with TIBCO ActiveMatrix BusinessWorks 5.x.

To use the module properties in any of the resource fields, enter the property name surrounded by `%%` on both sides. For example, you must enter `%%UserName%%` in the **User Name** field to use the `UserName` property.

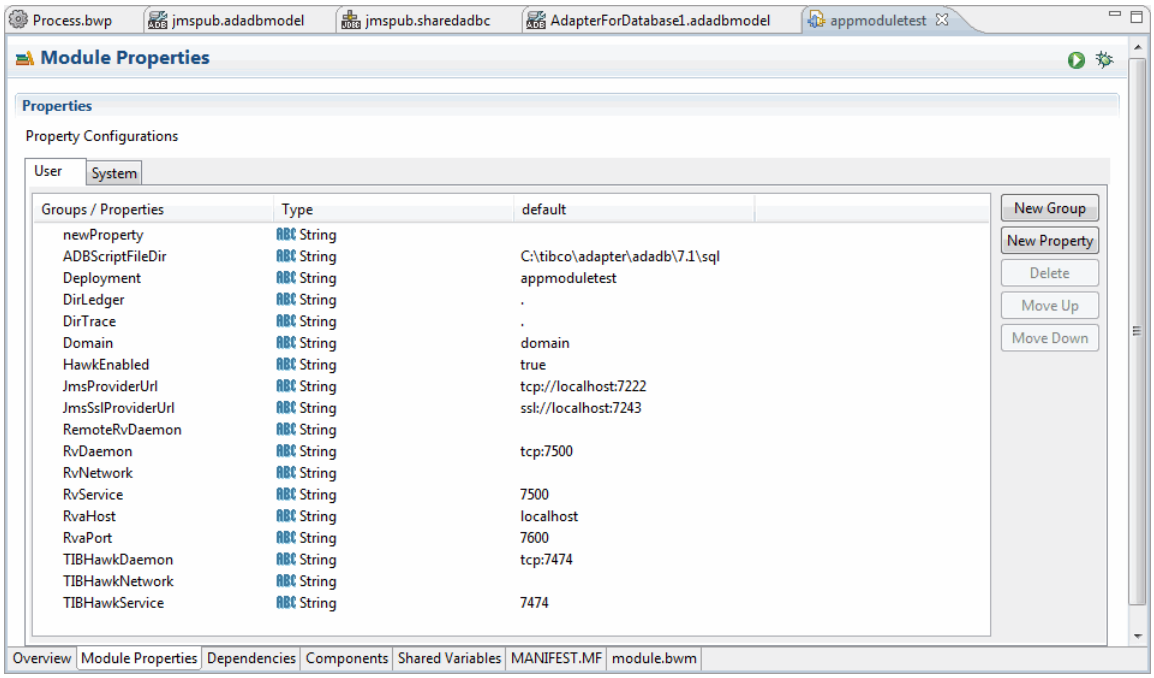
Configuring Module Properties

You can add, specify, and group module properties by using the Module Properties editor in TIBCO Business Studio.

You can also specify variable values in a TRA properties file. A variable value set in the properties file overrides the same variable set as a module property in TIBCO Business Studio.

Procedure

1. In the Project Explorer view, click **Module Descriptors** and double-click **Module Properties** from the project.



2. Click **New Property** to create a new module property.
3. Optional: Click **New Group** to create a group for a set of module properties.

Defining a Module Property with a Combined Value

You can define a module property with a combined value of other module properties and some literal value. But this type of definition is only supported in an adapter configuration, not in a BusinessWorks process.

For example, you can define the following two module properties in TIBCO Business Studio:

- Name: *port* Value: `8080`
- Name: *URL* Value: `http://localhost:port`

Based on these definitions, the literal value of *URL* is `http://localhost:8080`. But this definition is not supported in a BusinessWorks process.

Predefined Module Properties

TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio) contains predefined module properties. Some module properties are automatically used within the system when an adapter configuration is defined.

Variable	Description
ADBScriptFileDir	Directory to store the `.sql` scripts for cleaning up the database. Do not include the `.sql` file name in the property value.
Deployment	Defaults to the adapter configuration name. This value can be any string value. This global variable is used by the system to partially define the subject name defined for a service.
DirLedger	Specifies the path name of the TIBCO Rendezvous certified messaging ledger file. The default value is the root installation directory.
DirTrace	Specifies the path name for a log file used by the adapter. The default value is the root installation directory.
Domain	The default value for file-based local projects is domain. The value for server-based projects is the domain to which the project was saved.
HawkEnabled	Indicates whether TIBCO Hawk is used to monitor the adapter. Two values are available: • True: a Hawk microagent is defined for the adapter. • False: the microagent is not to be used. Default is The default value is `True` in TIBCO Business Studio.
JmsProviderUrl	Tells applications where the JMS daemon is located. Setting this value mostly makes sense in early stages of a project, when only one JMS daemon is used.
JmsSslProviderUrl	Tells applications where the JMS SSL daemon is located.

Variable	Description
RemoteRvDaemon	TIBCO Rendezvous routing daemon (rvrd) to be used. See *TIBCO Administrator Server Configuration Guide* for details about setting up a domain using rvrd.
RvDaemon	TIBCO Rendezvous daemon. Sessions use this daemon to establish communications. The default value is 7500.
RvNetwork	TIBCO Rendezvous network. Set this variable only on computers with more than one network interfaces. If you specify this variable, the TIBCO Rendezvous daemon uses that network for all outbound messages. In most cases, you can leave the default.
RvService	TIBCO Rendezvous service. The TIBCO Rendezvous daemon divides the network into logical partitions. Each transport communicates on a single service. A transport can communicate only on the same service with other transports. Unless you are using a non-default TIBCO Rendezvous configuration, you have to keep the default value (7500).
RvaHost	Computer on which the TIBCO Rendezvous agent runs. This variable is relevant only when you are using the TIBCO Rendezvous Agent (rva) instead of the TIBCO Rendezvous daemon, and if you have configured a non-default setup. See *TIBCO Rendezvous Administration* for details about specifying the rva parameters.
RvaPort	TCP port on which the TIBCO Rendezvous agent (rva) listens for client connection requests. See *TIBCO Rendezvous Administration* for details about specifying the rva parameters. The default value is 7501.
TIBHawkDaemon	TIBCO Rendezvous daemon used in the TIBCO Hawk session. Specifies which Hawk daemon handles communication for the session. A local daemon is specified by the communications type (always tcp) and a socket number. The default configuration uses the local daemon with the TCP socket number 7474. Specify a remote daemon by inserting its host name or IP address between the tcp entry and the port number of the daemon parameter, such as `tcp:remote_computer:7800`. See *TIBCO Hawk Installation and Configuration* manual for details about this parameter.
TIBHawkNetwork	TIBCO Rendezvous network used by the TIBCO Hawk session. Specifies which network to use for outbound session communications when a computer is connected to more than one network, and also specifies the multicast groups to use for communication. See *TIBCO Hawk Installation and Configuration* manual for details about this parameter.

Variable	Description
TIBHawkService	TIBCO Rendezvous service used by the TIBCO Hawk session. The **`Service`** parameter specifies which User Datagram Protocol (UDP) service group the TIBCO Rendezvous daemon must use for session communications. The default service port is 7474. See *TIBCO Hawk Installation and Configuration* manual for details about this parameter.

Adapter Properties File

The runtime adapter parses one or more properties files at startup. The default properties file for TIBCO ActiveMatrix Adapter for Database is the `adbagent.tra` file, which is located in the *`TIB_ADADB_HOME`*`\bin` subdirectory. The runtime adapter provides a template TRA file for each properties file when the actual properties file is corrupted or deleted by mistake.

Each line in a properties file is a single property. Each property consists of a key and a value:

- The key starts with the first non-whitespace character and ends at the first occurrence of these characters. The format is as follows:

  (space) : =

- The value starts at the first character after any of the preceding three characters.

The following example shows some adapter properties and their formats:

```
tibco.configurl=/tibco/private/adapter/test/config/config1
tibco.repourl=tibcr://TEST_PROJECT
tibco.username=admin
tibco.password=samplePassword
tibco.clientVar.service=7600
tibco.clientVar.daemon=tcp:7600
```

Properties defined in the properties file of a deployed service override the same properties defined in the project. The properties file for a deployed service is located in the following directory:

`TIBCO_TRA_DOMAIN_HOME``/`*`domainName`*`/application/`*`applicationDeploymentName`*

The following restrictions apply to properties:

- Do not use the exclamation point (!) as a comment line indicator. Instead, use the number sign (#).
- The line continuation character is ignored, and you cannot define a property with multi-line values.
- A key cannot contain any of the termination characters. Although you can use termination characters by escaping the value with a preceding backslash (\) in Java, TIBCO ActiveMatrix Adapter for Database does not support this syntax.
- All paths inside a properties file, including Microsoft Windows directory names, must use forward slashes (/).

Adapter Properties

The runtime adapter properties are in two categories: required properties and TIBCO ActiveMatrix Adapter for Database properties.

Required Properties

To use a properties file with a runtime adapter, you must use correct properties for the configuration of the adapter.

The following table lists the properties that the runtime adapter requires. Most required properties are predefined by TIBCO Adapter SDK. For more information, see "Properties Files" in *TIBCO Adapter SDK Programmer's Guide*.

Property	Description
`tibco.repourl` *`repourl`*	Absolute pathname of the local repository where the adapter configuration is defined. For a remote project, the `repourl` value must use the form `tibco.repourl tibcr@name` where *name* is the repository name. For example: `tibco.repourl tibcr@ADBRepoDefault` For Unix systems, the path separator includes a single forward slash (/). For example: `/local/tibco/repo/repo.dat`
`tibco.configurl` *`relative_path`* or `tibco.configurl` *`absolute_path`*	Location of the adapter service inside the project file. If a relative path is specified, the adapter service is assumed to be under the default area in the project file (/tibco/private/adapter/). For example, the following value connects to an adapter service named adbpub in the `/tibco/private/adapter/` directory: tibco.configurl adbpub If an absolute path is specified, the adapter configuration is looked up in the repository as defined by the argument. For example: `tibco:configurl /tibco/private/adapter/adbpub`
`tibco.instanceid` *`instance name`*	Name of an adapter configuration. The length of a name cannot be larger than 80 characters.
`tibco.clientVar.adb.password`	Password used to connect to the targeted database. This can be obfuscated by using the instructions in Password Handling.

`application.args`	Properties (.tra) file to pass to TIBCO ActiveMatrix Adapter for Database. For example:
	`application.args adbagent -system:propFile`
	`D:/tibco/adapter/adadb/7.2/bin/adbsub.tra`

`application.start.dir`	Path name of the adapter to start. For example: `application.start.dir D:/tibco/adapter/adadb/7.2/bin/`

TIBCO ActiveMatrix Adapter for Database Properties

In addition to properties that are listed in the adapter properties file, some TRA properties are also predefined in TIBCO ActiveMatrix Adapter for Database. Properties that start with `ntservice` are available only on Microsoft Windows platforms.

The following table lists the predefined properties in the alphabetical order:

Property	Description
adb.*<publisher_service_name>*.jmsProperties *<property1>=<value1>*, *<property2>=<value2>*, *<property3>=<value3>*	Set JMS message properties when you use the JMS transport type for Publication Service. For more information, see TIBCO Enterprise Message Service documentation. During the run time, this property has higher precedence than the adb.jmscompress property.
adb.<publisher_service_name>.lookback *<on/off>*	When you set the value to on, the adapter can poll remaining records continuously regardless of the Polling Interval value you set.
adb.*<publiser_service_name>*.preregistered listeners *<subjectName1:listenerName1*, *subjectName2:listenerName2>*	Preregisters RVCM names for the specified subjects. For example, adb.ADBPublisher.preRegisteredListeners=adb.sub1:rvcm1, adb.sub2:rvcm2 preregisters RVCM name rvcm1 to subject adb.sub1, and RVCM name rvcm2 to subject adb.sub2. For more information, see Preregistering a Certified Subscriber.
adb.*<sessionName/serviceName>*.ADBQueueSize *<queue size>*	Size of an internal message queue in the service level. The default value is 1000. In the previous release, the default value of this property is -1, which indicates that the adapter does not limit the queue size. If you use the default value, when the adapter receives a large number of messages from a TIBCO messaging bus within a short period, in which the receiving speed exceeds the processing speed of the adapter, the OutOfMemory error might occur. To resolve this issue, the default value is changed to 1000 in this release. Even within a short time, the adapter fetches only 1000 rows, which contributes to a reasonable performance of the adapter. Selection of the *sessionName* or *serviceName* variable depends on whether you use a separate session. Specify the *sessionName* variable if you do not use a separate session; specify the *serviceName* variable if you use a separate session. For more information on how to configure the internal queue, see Configuring the Internal Message Queue.
adb.*<sessionName/serviceName>*.ADBQueueTimeout *<milliseconds>*	Interval that the adapter waits to retrieve messages from the transport queue when the internal message queue is saturated. The default value is 500 ms. Selection of the *sessionName* or *serviceName* variable depends on whether you use a separate session. Specify the *sessionName* variable if you do not use a separate session; specify the *serviceName* variable if you use a separate session. For more information on how to configure the internal queue, see Configuring the Internal Message Queue.

Property	Description
`adb.<tableName>.poll.hint <hint>`	This feature helps improve the performance of your queries. This feature is only supported by Oracle and SQLServer databases. For more information, see Using Hints (Publication Service Only).
`adb.addCustomHawkMethodsToClassMAgent <on/off>`	To add custom methods to the standard micro agent, set the value of this property to `on`. To disable the add operation, set the value of this property to *off*.
`adb.as400.defaultLibrary <AS400 default libarary at design time>`	This property identifies the default iSeries library to be accessed and only used for library name verification. This property applies only to Publication Service and Subscription Service.
`adb.as400.library <AS400 runtime library>`	It applies only to Publication Service and Subscription Service. The rules for which a library name is used are as follow: • If the library name specified at design time is equal to the `adb.as400.defaultlibrary` name, the adapter will use the `adb.as400.library` name specified in the TRA file. • If the `adb.as400.library` name specified in the TRA file is equal to the `adb.as400.defaultlibrary` name, the adapter will use the library name specified at design time. • If the library name is not defined at design time, the publishing table cannot be created.
`adb.batchPubStatusUpdates <on/off>`	The default value is set to `off`. When the `adb.PollingBatchSize` property is used, the `adb.batchPubStatusUpdates` property optimizes the publication performance by batching message status updates to the publishing table. Do not use this property when messages are published by using a parameterized subject name. If an adapter configuration stops before a batch update is performed, the status column is not updated. As a result, duplicate messages are published when the adapter configuration is restarted.
`adb.customScaleForNumberType <table>.<column>=<scale>, <table>.<column>=<scale>`	Applies only to Subscription Service. This property sets the default scale of the Oracle Number (empty) data type. It is used to control the entered number with high precision.
`adb.datePattern <date pattern for java.text.SimpleDateFormat, such as yyyy-MM-dd>`	The standard date format pattern for `java.text.SimpleDateFormat`. The default format is `yyyy-MM-dd`.

Property	Description
`adb.debug <level>`	The debug printing level. If this property is not specified, the default value of 2 is used. Possible values are: • 0 - Log no debug information. • 1 - Log SQL commands executed against the database. • 2 - Log binding data for each SQL command. • 3 - Log all debug information.
`adb.<default_session_name>.subBatchCommitSize <number of batch commit size>`	Applies only to Subscription Service. This property specifies the number of messages to be batched before invoking a commit operation. The default value is `1`. You can set this property if you want to enable batch commit with the default session in TIBCO Business Studio. If you set values for both the `adb.subBatchCommitSize` property and the `adb.<default session name>.subBatchCommitSize` property, the value of the `adb.subBatchCommitSize` property is used. If messages greater than 32K are published, batching is automatically turned off. For details on how to configure batch commit for Subscription Service, see Configuring Batch Commit for Subscription Service.
`adb.<default_session_name>.subBatchCommitTimeout <milliseconds>`	Applies only to Subscription Service. This property specifies the longest interval between two batch commits.When the specified timeout expires, the subscriber adapter commits all the received messages even if the number does not reach the specified batch commit size. You can set this property if you want to enable batch commit without using separate sessions in TIBCO Business Studio. The default value is `10,000 milliseconds`. If you set values for both the `adb.subBatchCommitTimeout` property and the `adb.<default session name>.subBatchCommitTimeout` property, the value of the `adb.subBatchCommitTimeout` property is used. For details on how to configure batch commit for Subscription Service, see Configuring Batch Commit for Subscription Service.
`adb.disableTerminationSubject <on/off>`	When this property is set to `on`, the adapter does not terminate on receiving a termination subject message.
`adb.groupSize <publisher group size>`	Specifies the number of rows to be published in a single message. Setting of this property overrides the Group Size setting in the Publication Service configuration.
`adb.jmsBytesMsg <on/off>`	Determines whether you use the JMS transport type to send byte messages. The default value is `off`.

Property	Description
`adb.jmscompress <on/off>`	One of the JMS properties. This property applies only to Publication Service with the JMS transport type. The value can be `on` or `off`. The default value is `off`, which indicates that JMS messages are not compressed. For more information, see Compressing JMS Messages.
`adb.log4jPropFile <The path of the properties file>`	The path in which the properties file is stored.
`adb.maxLongLen -1 <maximum size of handle data. default value is -1, not check data length>`	Buffer size. Used for long data types, such as BLOB.
`adb.noDupDetection <on/off>`	Disables detection of duplicate configurations. The value can be `on` or `off`. The default value is `off`, which indicates that duplicate configurations are detected.
`adb.password`	Password used by the adapter to access databases. If it is not specified, the runtime adapter will call the password at the design time. If it is set in the TRA file, the runtime adapter will call the password from the TRA file.
`adb.payloadOnError on|off`	This property specifies whether only the error information or all the logging information is printed in the log file. The default value is `off`, and it applies only to Subscription Service. When configuring this property, you must note the following: • When creating the BusinessWorks process, you must set a value in the **Reply Subject** field in the **Input** tab. • If the value is `on` or the **Generate Payload On Error** check box in the **General** tab is selected when configuring an adapter configuration, and the **Log Info Messages** check box in the **Logging** tab is cleared, the error log information that belongs to Debug role is displayed in Error role.
`adb.perf <the number of messages, it will print the performance report after process the number of message>`	Number of messages. It prints the performance report after processing the number of messages.
`adb.perfMon <on/off>`	The value can be `on` or `off`. The default value is `off`.

Property	Description
`adb.PollingBatchSize <number of rows>`	Applies only to publication adapter configurations. Limits the amount of messages to be fetched. The value indicates the number of parent rows to be fetched for a polling interval. The default value is `1000`. In the previous release, the default value of this property is `0`, which indicates that all new rows are fetched. If you use the default value, when the adapter receives a large number of messages from a database within a short period, in which the receiving speed exceeds the processing speed of the adapter, the OutOfMemory error might occur. To resolve this issue, the default value is changed to `1000` in this release. Even within a short time, the adapter fetches only 1000 rows, which contributes to a reasonable performance of the adapter.
`adb.PollingInterval <milliseconds>`	Specific polling period. Applies only to Publication Service. If not specified, the default value of `10000` milliseconds is used.
`adb.PollingCommitForDB2 <on/off>`	The value can be `on` or `off`. The default value is `off`.
`adb.pubBatchConfirmSize <number of confirmations>`	Applies only to Publication Service that uses certified message delivery. Optimizes performance by batching message status advisories to the publishing table. The value indicates the number of advisory messages to include in a single batch. Do not use this option when messages are published using a parameterized subject name.  If an adapter configuration stops before a batch update is performed, the status column is not updated. As a result, messages that were successfully published still have a status of P (pending) in the publishing table when the adapter configuration is restarted. In this case, the ledger file contains the correct status information. A smaller value for this property decreases this risk.

Property	Description
`adb.pubBatchConfirmTimeout <milliseconds>`	Applies only to Publication Service that uses certified message delivery. This property specifies the number of milliseconds to wait before updating the status column. After this interval, an update is performed even if the batch size value is not reached. The default value is `10000` milliseconds (10 seconds). The value of 0 milliseconds means that no timeout interval is used. Do not use this option when messages are published using a parameterized subject name.  If an adapter configuration stops before a batch update is performed, the status column is not updated. As a result, messages that were successfully published still have a status of P (pending) in the publishing table when the adapter configuration is restarted. In this case, the ledger file contains the correct status information. A smaller value for this property decreases this risk.
`adb.publishChildData <on/off>`	Applies only to parent-child relationships in Publication Service. Enables publishing child table rows. The default value is `on`.
`adb.requestResponseMaxRows <number of maxRows>`	This property specifies the maximum number of rows to be fetched. This can be used to limit the memory usage of the adapter. The unfetched rows will be ignored by the adapter.
`adb.requestResponseThreads <the number of rpc default session threads>`	The number of threads used by Request-Response Service. The default value is `1`.
`adb.RetryTotal <total reconnection attempts>`	Total number of reconnection attempts.
`adb.runtime.publisherSchema <runtime publisher table schema name>`	The runtime publisher table schema name. Setting of this property overrides setting of the `adb.runtime.schema` property. This can be used if you have a different schema for the publishing table than the other table objects, such as the source table. For more information, see Runtime Table Schema Configuration.

Property	Description
`adb.runtime.schema <runtime database object schema name>`	The runtime database object schema name. If the `adb.originalSchema` property is specified, the adapter will compare the `adb.originalSchema` property with the prefix of the table objects. If they are the same, the adapter will replace the design-time schema with this runtime schema. Otherwise, no action is taken. If the `adb.orignalSchema` property is not specified, the adapter will append this runtime schema to all table objects that do not have any schema specified at design time. For more information, see Runtime Table Schema Configuration.
`adb.rvAdvisoryNoLog <on/off>`	This property specifies whether RV advisory messages are to be logged in the adapter log files. The default value is `off`.
`adb.rvMaxQueueSize <RV event queue size>`	Applies only to Subscription Service. Specifies the maximum number of messages that can be put in a TIBCO Rendezvous event queue. The default value is `0`, which means no limit is placed on the event queue size.  If you set the value to be smaller than the message count, the message might be lost.
`adb.setClientInfo <Client info setting on Oracel>`	When the value of this property is set to `on`, the adapter will call SET_CLIENT_INFO to set the database session client information. The default value is `on`.
`adb.setEmptyStringNullForRvMsg <on/off>`	Specifies whether the RVMSG fields of an empty string are treated as NULL or "" (empty strings). The value can be `on` or `off`. The default setting is `off`. If the property value is set to `on`, empty strings are treated as NULL. If the property value is set to `off`, empty strings are treated as empty strings if the database supports it.
`adb.setEmptyStringToDefault <on/off>`	The value can be `on` or `off`. The default setting is `off`. This property only works when the following three criteria are met: • Oracle database • Insert operations • `JDBC Property CatalogOptions=1` When you set a default value to a field in creating a table, if the property value is set to `on`, the empty strings are changed to the default value.

Property	Description
`adb.SleepBetweenRetries <milliseconds of sleep between two reconnection attempts>`	Milliseconds of sleep between two reconnecting attempts.
`adb.stmtCache <Number of statements to be cached>`	Number of cache statements for a generic RPC request/ reply service. The number of statements that the adapter caches will be executed directly for repeated requests. If the cache is full, the adapter will remove the oldest message from the cache and add the new statement. The default value is `1`.
`adb.subBatchCommitSize <number of commit size>`	Applies only to Subscription Service. Specifies the number of messages to be batched before you invoke a commit operation. The default value is `0`. For more information, see Batch Processing in Subscription Service.  If messages greater than 32K are published, batching is automatically turned off.
`adb.subBatchCommitTimeout <milliseconds>`	Applies only to Subscription Service. The amount of time that can expire after a batch commit operation is invoked. If not specified, the default value is `10,000` milliseconds. For more information, see Batch Processing in Subscription Service.
`adb.subBulkInsertSize <number of inserts>`	Applies only to Subscription Service. All incoming messages to be inserted are stored until this size is reached. Then a bulk insert operation is performed on the destination table. This value must be less than or equal to the value specified for the `adb.subBatchCommitSize` property, if used. The default value is `1`. If an update statement is published while messages are being batched, the bulk insert is performed regardless of whether the size value has been reached. After records have been inserted, the update operation is performed. For more information, see Batch Processing in Subscription Service.  Do not use this option if LONG, LONG RAW, image, or varbinary records are published.

Property	Description
adb.supportOracleAppsForRPC *<on/off>*	If you set the value of the adb.supportOracleAppsForRPC property to on in the adbagent.tra file, the project containing Request-Response Service can work properly after being migrated from TIBCO Adapter for Oracle Applications to TIBCO ActiveMatrix Adapter for Database 6.x and then migrated from version 6.x to the current version. If you set the value of this property to off, only the projects created in TIBCO ActiveMatrix Adapter for Database 6.x are supported in the current version after migration. The default value of the adb.supportOracleAppsForRPC property is off.
adb.taskBackLogLimitInBytes *<The maximum size (in bytes) of the scheduler task queue>*	The scheduler stores tasks in a queue. This property limits the maximum size of that scheduler task queue by the number of bytes. The properties can control the memory usage on the adapter side. This value must be an integer. The default value is unspecified, which indicates that the size of the scheduler task queue is not limited.
adb.taskBackLogLimitInMessages *<The maximum size (in messages) of the scheduler task queue>*	The scheduler stores tasks in a queue. This property limits the maximum size of that scheduler task queue by the number of messages. The properties can control the memory usage on the adapter side. This value must be an integer. The default value is unspecified, which indicates that the size of the scheduler task queue is not limited.
adb.terminateOnPubFail *<on/off>*	Specifies that if publication fails during the message transmission, the agent will terminate after the status has been updated to 'F'. The default value is off.
adb.timePattern <time pattern for java.text.SimpleDateFormat, such as *HH:mm:ss*>	The standard time format pattern for java.text.SimpleDateFormat. The default format is HH:mm:ss.
adb.timestampPattern <timestamp pattern for java.text.SimpleDateFormat, such as *yyyy-MM-dd HH:mm:ss.S*>	The standard timestamp format pattern for java.text.SimpleDateFormat. The default format is yyyy-MM-dd HH:mm:ss.S.
adb.useBetweenClause *<on/off>*	This property disables the use of the BETWEEN clause in the select query of the publisher. The default value is on.
adb.useExceptTable *<on/off>*	Enables the use of the exception table. The exception table is defined when Subscription Service is created. The default value is set to on.

Property	Description
`adb.url`	Database URL used by the adapter to access database. For example, jdbc:tibcosoftwareinc:db2://IP:50000;databaseName=adb
`adb.user`	Database account name used by the adapter to access databases. If it is not specified, the runtime adapter will call the user name at the design time. If it is set in the TRA file, the runtime adapter will call the user name from the TRA file.
`adb.verbose <on/off>`	Verbose mode. Print all available information to the console window or log file location. By default, the verbose mode is on.
`ntservice.account`	Username that you use to run the Windows Service. You can use this property to initially set the account for the service, but once the service is installed, use the Services control to change the user account of services.

`ntservice.binary.path.absolute`	Absolute path in which the executable is run when a service is started. For example,
	`ntservice.binary.path.absolute`
	`D:/tibco/adapter/adadb/7.2/bin/adbagent.exe`

`ntservice.dependencies`	Number of dependencies.
`ntservice.displayname`	Name to be displayed in the Services control for this Windows Service. This property is useful if you want to have multiple Windows Services for the same executable. That is, you want to have two adapters running on the same machine. By specifying different service names and display names for the adapters, you can accomplish this.
`ntservice.interactive true|false`	The value can be `true` or `false`. Specifies whether the Windows Service is interactive. Set the value to `false` if you are not using a system account.
`ntservice.name`	Name for this Windows Service. This property is useful if you want to have multiple Windows services for the same executable. That is, you want to have two adapters running on the same machine. By specifying different service names and display names for the adapters, you can accomplish this. For example, `ntservice.name` *`adapter_instance_name`*

Property	Description
`ntservice.password`	Password for the user name in the `ntservice.account` property. You can use this property to initially set the password for the user account, but once the service is installed, use the Services control to change the password.
`ntservice.starttype manual|automatic`	Start type for this Windows Service. The value can be `manual` or `automatic`. For example, `ntservice.starttype automatic` You can use this property to initially set the start type for the service, but once the service is installed, use the Windows Services control to change the start type of services.

`tibco.clientVar`	Runtime values for global variables defined inside the repository. This value takes precedence over any global value set in the repository. Substitution takes place at run time.
	You append the global variable to the `tibco.clientVar` property, then give its value. For example, a global variable named DirLedger is specified as follows:
	`tibco.clientVar.DirLedger`
	`C:/tibco/adapter/adadb/7.2/myledger`
	Do not include the % character of substitution variables. For example, to set `%%RvDaemon%%="tcp:7500"`, use `tibco.clientVar.RvDaemon "tcp:7500"`.

`tibco.username` `tibco.password`	User name and password used by the repository server to access the project. The password can be obfuscated by using the instructions in Handling Passwords.
`-version`	Displays a banner with version information, and then exits. This option is for troubleshooting purposes only.

Trace Messages

Trace messages provide information about adapter activities. The messages are logged to the console where the runtime adapter is started and to a log file. Trace messages can also be redirected to the TIBCO Hawk Display application, or sent to other applications using the TIBCO Rendezvous transport type.

Each trace message includes the fields as shown in the following order:

1. Timestamp
2. Adapter Identifier
3. Role
4. Category
5. Status Code
6. Tracking Identifier

The preceding fields are described in Trace Message Fields. The following example shows the field structure of a trace message:

```
2015 Apr 07 16:53:06:487 GMT +0800 AdapterForDatabase Debug [Publication] AEADB-321009
        Publication Service <ORDER_TABLEADBPublisher> thread <AdapterForDatabase.ORDER_TABLEADBPublisher.DATAHANDLER.1> execute pub table updater...
        SQL Statement: UPDATE PUB_ORDER SET ADB_L_DELIVERY_STATUS = 'C', ADB_TRACKINGID = ? WHERE ADB_SEQUENCE = ?
               ADB_TRACKINGID = [DnkjGiywYlJ11kgIpnSKf0RE4po] (String)
               ADB_SEQUENCE = [12] (String)
```

The structure of this trace message is as follows:

1. Timestamp: 2015 Apr 07 16:53:06:487 GMT +0800
2. Adapter Identifier: AdapterForDatabase
3. Role: Debug
4. Category: [Publication]
5. Status Code: AEADB-321009 Publication Service <ORDER_TABLEADBPublisher> thread ....
6. Tracking Identifier: ADB_TRACKINGID = [DnkjGiywYlJ11kgIpnSKf0RE4po]

Trace Message Examples

The following trace messages are written during a session where TIBCO ActiveMatrix Adapter for Database receives a message from a business process, and then processes the message.

Example 1: Adapter Started

The following message indicates that TIBCO ActiveMatrix Adapter for Database has started. The timestamp indicates when the adapter started, and the role indicates that the trace message is informational, which means the activity is normal for the adapter. The category is identified, and the corresponding status code is displayed. The status code indicates that the adapter has started successfully.

```
2015 Mar 26 14:30:36:068 GMT +0800 jmsServer Info [Instance] AEADB-100004
 Active Database Adapter has started successfully.
```

Example 2: Message Received

The next set of trace messages indicates the adapter has received a message that is sent on the JMS destination. The `#sUUUhFMQJAcV9UKXuKQcePoEb5k#` tracking identifier that is included in the trace

message uniquely identifies the message. The adapter from which the message originated provided the identifier.

```
2015 Mar 26 14:34:35:634 GMT +0800 jmsServer Info [RPC] AEADB-500001
 RPC Service Thread <RPC.JMSTopic.Thread.1> receive data.

tracking=#sUUUhFMQJAcV9UKXuKQcePoEb5k#BW.operation_ora.application.operation_ora.JMS
_7
Execute-procedure-getPrice_output.Invoke-an-Adapter-Request-Response-Service#
```

Example 3: Object Moved

The final trace message indicates the message has been moved to the output directory, which completes the interaction of the adapter with the message. Because the trace message is the termination point, the tracking identifier is not displayed.

```
2003 Feb 22 20:15:42:812 GMT -8
FileAdapter.FileAdapterConfiguration Info [Adapter] AEFA-000070
File customers.txt is moved to the Output Directory
F:\ca\integration\001\data_sets\files\solutions\output
```

Trace Message Fields

Each trace message includes the following fields: **Timestamp**, **Adapter Identifier**, **Role**, **Category**, **Status Code**, **Tracking Identifier**, and **Application Information**.

Field	Description
Timestamp	Timestamp of occurrence. For example, `2003 Feb 22 20:14:51:718 GMT -8`.
Adapter Identifier	Name of the adapter that wrote the trace message. This is a combination of the adapter acronym and adapter configuration name. For example, the application identifier, `ADB.publisher1` identifies a TIBCO ActiveMatrix Adapter for Database service named publisher1.
Role	A role can be: **Info** Indicates a normal adapter operation. No action is necessary. A tracing message tagged with the **Info** field indicates that a significant processing step was reached and has been logged for tracking or auditing purposes. Only info messages preceding a tracking identifier are considered significant steps. **Warn** Indicates that an abnormal condition is found. Processing continues, but special attention from an administrator is required. **Error** Indicates that an unrecoverable error occurs. Depending on the error severity, the adapter might continue with the next operation or stop altogether. **Debug** Indicates a developer-defined tracing message. In normal operating conditions and debug messages, this field is not displayed. When configuring the adapter, you can define what roles must be logged. For example, to improve performance, you can decide not to log Info roles.

Field	Description
Category	The following lists all categories: **Adapter** The adapter is processing an event. **Application** The adapter is interacting with the database. **Configuration** The adapter is reading configuration information. **Database** The adapter is interacting with a database. **Metadata** The adapter is retrieving metadata from the database. **Palette** The adapter is interacting with the palette. **Publisher Service** Publication Service is reporting this trace message. **Request-Response Client Service** Request-Response Invocation Service is reporting this trace message. **Request-Response Server** Request-Response Service is reporting this trace message. **Shutdown** The adapter is being shut down. **Startup** The adapter is starting. **Subscription Service** Subscription Service is reporting this trace message. **System** This category is not linked to a specific event process. The trace message might be related to a Windows service related messages, memory allocation, file system error, and so on. **TibRvComm** The adapter is communicating with TIBCO Rendezvous. **XML** The adapter is parsing XML documents.
Status Code	Unique code for the message and description. Status codes are identified by a unique number and description. If a trace message includes an error or the Warn role, the status code documentation includes a resolution. See Trace Messages Reference for details.

Field	Description
Tracking Identifier	A unique identifier that is "stamped" on each message by the originating adapter. The tracking identifier remains in effect from the beginning of a message to its completion as the message is exchanged by TIBCO applications. If the adapter is the termination point of the message, the tracking identifier is not displayed in the trace message. You cannot modify the tracking identifier format or configure what information is displayed.
Application Information	Application-specific information added to the tracking info to trace the message back to its source. Set initially by the originating adapter and carried forward. It is augmented by each intermediate component.

Trace Messages Reference

In environments where multiple applications are used simultaneously, the possible status of messages increases as well. The trace messages are listed in numerical order with their fields and resolutions.

Status Code	Role	Category	Resolution
AEADB-100001	`Adapter starts to initialize.`		
	infoRole	Adapter	Normal operation; no action is necessary.
AEADB-100002	`Start %1 <%2>.`		
	infoRole	Adapter	Normal operation; no action is necessary.
AEADB-100003	`Start %1 <%2> with %3 session.`		
	infoRole	Adapter	Normal operation; no action is necessary.
AEADB-100004	`Adapter has started successfully.`		
	infoRole	Adapter	Normal operation; no action is necessary.
AEADB-100005	`Shutdown the adapter, current status is %1.`		
	infoRole	Adapter	Check the database connection configuration or other configuration.
AEADB-100006	`Start to reconnect.`		
	infoRole	Adapter	Check whether the database is connected or not.
AEADB-100007	`Reconnect successfully on attempt %2.`		
	infoRole	Adapter	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-100 008	`The <%1> is an Oracle Streams service. ADB will ignore it during starting up.\n\t`		
	infoRole	Adapter	Normal operation; no action is necessary.
AEADB-110 001	`There is a same instance startup in the ip: <%1>.`		
	warnRole	Adapter	Check if the same instance name is started by another agent.
AEADB-110 002	`Received duplication detection request from ip: <%1>, host:<%2>.`		
	warnRole	Adapter	Normal operation; no action is necessary.
AEADB-110 003	`No service type matched with the service <%1>.`		
	warnRole	Adapter	Check the transport type or the service type.
AEADB-110 004	`Adapter thread <%1> the field <%2> size <%3> is larger than adb.maxLongLen <%4>.`		
	warnRole	Adapter	If you are not sure whether you want to stop the adapter, you have to check the data length. To stop the adapter, set the value of the `adb.maxLongLen` property smaller than the data length; otherwise, set the value to `-1`.
AEADB-110 005	`ADB adapter refresh connection timer is started.`		
	warnRole	Adapter	Normal operation; no action is necessary.
AEADB-110 006	`ADB adapter refresh connection time out.`		
	warnRole	Adapter	Normal operation; no action is necessary.
AEADB-123 001	`TIBCO Hawk agent is found and agent implementation has been registered.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123 002	`Adapter connection test succeeded in startup.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123 003	`Adapter configuration is parsed.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123 004	`Adapter starts duplicate instance detection with the session <%1>.`		
	debugRole	Adapter	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-123005	`Create thread <%1> for %2 <%3>.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123006	`Adapter termination event listener is created with session.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123007	`Monitor advisory %1 with session <%2>.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123008	`Adapter reconnection thread is started.`		
	debugRole	Adapter	Check whether the database connected or not.
AEADB-123009	`Create dispatcher for session %1.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123010	`Publication Service <%1> preRegisteredListeners <%2> registered successfully.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123011	`Subscription Service Thread <%1> CustomScaleForNumberType set successfully for Table Columns: %2.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123012	`%1 <%2> Catalog:`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-123013	`Thread %1 counts down the stop latch.`		
	debugRole	Adapter	Normal operation; no action is necessary.
AEADB-130001	`Adapter throw exception %1 when shut down adapter.`		
	errorRole	Adapter	Check the error message and the configuration.
AEADB-130002	`Adapter failed to create termination event listener with session <%1>.`		
	errorRole	Adapter	Check the configuration.
AEADB-130003	`Adapter failed to reconnect.`		
	errorRole	Adapter	Check whether the database connected or not.

Status Code	Role	Category	Resolution
AEADB-130004	`Adapter duplication detection error: %1.`		
	errorRole	Adapter	Check if the message is null.
AEADB-130005	`Adapter connection test failed in startup.`		
	errorRole	Adapter	Check the configuration in the **Design-time Connection** tab.
AEADB-200001	`Adapter receives one message from termination subject <%1>.`		
	infoRole	Adapter	Check if the message is MdataEvent or MexceptionEvent.
AEADB-200002	`Adapter will shutdown.`		
	infoRole	Adapter	Check the logs and the configuration.
AEADB-200003	`Adapter termination subject is disabled, so keep running.`		
	infoRole	Adapter	Normal operation; no action is necessary.
AEADB-300001	`Publication Service <%1> thread <%2> starts polling message.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300002	`Publication Service <%1> thread <%2> publishes message on %3.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300003	`Publication Service <%1> thread <%2> publishes group message on %3.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300004	`Start alerter for Publication Services.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300005	`Adapter handles alerter message.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300006	`Publication Service <%1> thread <%2> add one row into the group message.`		

Status Code	Role	Category	Resolution
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300007	`Publication Service <%1> thread <%2> begin to refresh connection.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300008	`Publication Service <%1> thread <%2> finish refreshing connection.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300009	`Thread <%1> get space <%2>.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300010	`Publication Service <%1> thread <%2> insert data into <%3>.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300011	`Publication Service <%1> thread <%2> start to connect to metaspace <%3>.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300012	`Thread <%1> create space <%2>.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300013	`Publication Service <%1> thread <%2> upsert data into <%3>.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300014	`Publication Service <%1> thread <%2> delete data from <%3>.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300016	`Publication Service <%1> thread <%2> update data in <%3>.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300017	`Publication Service <%1> thread <%2> receives one message.`		

Status Code	Role	Category	Resolution
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300018	`Publication Service <%1> thread <%2> receives one message.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300019	`Publication Service <%1> thread <%2> acknowledges one message.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300020	`Publication Service <%1> thread <%2> illegal Group Size %3, Group Messaging disabled, Group Size set to 0.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-300021	`Thread <%1> the space <%2> %3.`		
	infoRole	Publication Service	Normal operation; no action is necessary.
AEADB-321001	`Publication Service <%1> execute rvcm Advisory Updater...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321002	`Publication Service <%1> RVCM Batch confirm %2 records.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321003	`Publication Service <%1> Activate the Polling Timer.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321004	`Publication Service <%1> Deactivate the Polling Timer.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321005	`Publication Service <%1> thread <%2> execute publishing table mini sequence selector...`		
	debugRole	Publication Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-321006	`Publication Service <%1> thread <%2> execute (ADB_L_DELIVERY_STATUS or ADB_L_DELIVERY) N -> S marker...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321007	`Publication Service <%1> thread <%2> execute publishing table selector...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321008	`Publication Service <%1> thread <%2> execute reference object selector...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321009	`Publication Service <%1> thread <%2> execute pub table updater...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321010	`Publication Service <%1> thread <%2> polling commit for DB2.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321011	`Publication Service <%1> thread <%2> batch update %3 records.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321012	`Publication Service <%1> thread <%2> polls records without polling timer fired...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321013	`Publication Service <%1> thread <%2> execute fault status updater...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321014	`Publication Service <%1> thread <%2> execute 'S' to 'N' updater...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321015	`Execute listen_alert statement for alerter.`		

Status Code	Role	Category	Resolution
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321016	`Publication Service <%1> thread <%2> parses message.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321017	`Publication Service <%1> thread <%2> polling commit for MySql using native driver.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-321018	`Publication Service <%1> thread <%2> execute publishing table max sequence selector.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322001	`Publication Service %1 thread %2 Fetch child data from %3.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322002	`Publication Service %1 thread %2 execute child selector...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322003	`Publication Service <%1> thread <%2> selects record data from source table.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322004	`Publication Service <%1> thread <%2> selects record data from reference object table.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322005	`Publication Service <%1> thread <%2> Start to fetch data from child table.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322006	`Publication Service <%1> thread <%2> Finish fetching data from child table.`		

Status Code	Role	Category	Resolution
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322 007	`Publication Service <%1> thread <%2>: no record found...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322 008	`Publication Service <%1> thread <%2> auto update status to 'N' statement is %3.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322 009	`Publication Service <%1> thread <%2> auto update status effect data count is <%3>.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322 010	`Publication Service <%1> thread <%2> the data does not exist in <%3> <%4>.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-322 011	`Publication Service <%1> thread <%2> the old data in <%3> has changed. <%4> .`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323 001	`Publication Service %1 thread %2 Publishing Message Info:`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323 002	`Publication Service <%1> begin to initialization.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323 003	`Publication Service <%1> create data handler <%2>.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323 004	`Publication Service <%1> data handler <%2> starts...`		
	debugRole	Publication Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-323005	`Publication Service <%1> create MTimer for rvcm batch confirm.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323006	`Publication Service <%1> create MTimer for polling.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323007	`Publication Service <%1> end to initialization.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323008	`Create alerter for Publication Services.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323009	`Initialize alerter for Publication Services.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323010	`Configure alerter of %1 for Publication Services.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323011	`Build listen_alert statement for alerter.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323012	`Publication Service <%1> Activate the RVCM batch confirm timer.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323013	`Publication Service <%1> Deactivate the RVCM batch confirm timer.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323014	`Publication Service <%1> Reset RVCM batch confirm timer.`		
	debugRole	Publication Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-323015	`Publication Service <%1> thread <%2> builds ADB_L_DELIVERY_STATUS or ADB_L_DELIVERY (DB2) N->S marker.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323016	`Publication Service <%1> thread <%2> builds publishing table selector whose ADB_L_DELIVERY_STATUS or ADB_L_DELIVERY (DB2) is S.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323017	`Publication Service <%1> thread <%2> builds publishing table mini sequence selector.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323018	`Publication Service <%1> thread <%2> binds (ADB_L_DELIVERY_STATUS or ADB_L_DELIVERY) N -> S marker...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323019	`Publication Service <%1> thread <%2> builds publishing table selector.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323020	`Publication Service <%1> thread <%2> binds publishing table selector, polling batch size is %3.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323021	`Publication Service <%1> thread <%2> constructs the instance based on the publishing table.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323022	`Publication Service <%1> thread <%2> constructs the instance based on the source table.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323023	`Publication Service <%1> thread <%2> builds reference object selector.`		

Status Code	Role	Category	Resolution
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323024	`Publication Service <%1> thread <%2> bind reference object selector.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323025	`Publication Service <%1> thread <%2> construct publisher endpoint.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323026	`Publication Service <%1> thread <%2>: the group size is %3, the current size is %4.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323027	`Publication Service <%1> thread <%2> builds pub table updater.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323028	`Publication Service <%1> thread <%2> binds pub table updater.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323029	`Publication Service <%1> thread <%2> acquire multithreading semaphore...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323030	`Publication Service <%1> thread <%2> release multithreading semaphore after marking records...`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323031	`Publication Service <%1> builds rvcm advisory updater.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323032	`Publication Service <%1> binds rvcm advisory updater.`		
	debugRole	Publication Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-323033	`Alerter configuration SQL: %1.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323034	`Alerter clean up SQL: %1.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323035	`Publication Service <%1> thread <%2> build fault status updater.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323036	`Publication Service <%1> thread <%2> bind fault status updater.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323037	`Publication Service <%1> thread <%2> build 'S' to 'N' updater.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323038	`Publication Service <%1> thread <%2> bind 'S' to 'N' updater.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323039	`Publication Service <%1> thread <%2> acquire table %3 lock in load balance mode.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323040	`Publication Service <%1> thread <%2> release table %3 lock in load balance mode.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323041	`Publication Service <%1> thread <%2> build child selector.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323042	`Publication Service <%1> thread <%2> bind child selector.`		
	debugRole	Publication Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-323 043	`Publication Service <%1> thread <%2> data: %3`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323 044	`Thread <%1> space fields: %2`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323 045	`Thread <%1> space attributes: %2`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-323 046	`Publication Service <%1> thread <%2> builds publishing table max sequence selector.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-324 001	`Publication Service %1 thread %2 throw exception %3 when initialize the DBLockMechanism.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-324 002	`Publication Thread <%1> when acquire the table lock timeout expired,throws exception %2.when get table lock again,recover to load balance mode.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-324 003	`The mutex table %1 is already existed.`		
	debugRole	Publication Service	Normal operation; no action is necessary.
AEADB-330 001	`Publication Service <%1> throw exception <%2> when create rvcm batch confirm timer.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330 002	`Publication Service <%1> throw exception <%2> when activate rvcm batch confirm timer.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.

Status Code	Role	Category	Resolution
AEADB-330003	`Publication Service <%1> thread <%2> throw exception %3 when get table semaphore in multithreads mode.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330004	`Publication Service <%1> thread <%2> throw exception %3 when stop the adapter.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330005	`Publication Service <%1> throw exception <%2> when create polling timer.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330006	`Publication Service <%1> thread <%2> throw exception %3 when build pub load balancing marker.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330007	`Publication Service <%1> thread <%2> throw exception %3 when build pub load balancing pub table selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330008	`Publication Service <%1> thread <%2> throw exception %3 when build publishing table mini sequence selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330009	`Publication Service <%1> thread <%2> throw exception %3 when execute the mini sequence selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330010	`Publication Service <%1> thread <%2> throw exception %3 when bind pub load balancing marker.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330011	`Publication Service <%1> thread <%2> throw exception %3 when execute pub load balancing marker.`		

Status Code	Role	Category	Resolution
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330012	`Publication Service <%1> thread <%2> throw exception %3 when build publishing table selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330013	`Publication Service <%1> thread <%2> throw exception %3 when bind publishing table selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330014	`Publication Service <%1> thread <%2> throw exception %3 when execute publishing table selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330015	`Publication Service <%1> thread <%2> throw exception %3 when get dbTable whose name is publishingTableName.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330016	`Publication Service <%1> thread <%2> throw exception %3 when construct MInstance.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330017	`Publication Service <%1> thread <%2> throw exception %3 when build reference object selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330018	`Publication Service <%1> thread <%2> throw exception %3 when bind reference object selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330019	`Publication Service <%1> thread <%2> throw exception %3 when construct minstance based on publishing table.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.

Status Code	Role	Category	Resolution
AEADB-330020	`Publication Service <%1> thread <%2> throw exception %3 when construct minstance based on source table.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330021	`Publication Service <%1> thread <%2> throw exception %3 when construct minstance based on reference object table.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330022	`Publication Service <%1> thread <%2> throw exception %3 when fetch child data.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330023	`Publication Service <%1> thread <%2> throw exception %3 when construct publisher endpoint.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330024	`Publication Service <%1> thread <%2> throw exception %3 when publish group message.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330025	`Publication Service <%1> thread <%2> throw exception %3 when build pub table updater.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330026	`Publication Service <%1> thread <%2> throw exception %3 when bind pub table updater.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330027	`Publication Service <%1> thread <%2> throw exception %3 when update entry status.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330028	`Publication Service <%1> throw exception %2 when build rvcm advisory updater.`		

Status Code	Role	Category	Resolution
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330029	`Publication Service <%1> throw exception %2 when bind rvcm advisory updater.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330030	`Publication Service <%1> throw exception %2 when deactivate the RVCM batch confirm timer.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330031	`Publication Service <%1> throw exception %2 when reset rvcm batch confirm timer.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330032	`Main thread throw exception %1 when configure alerter of %2. SQLSTRING : %3.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330033	`Main thread throw exception %1 when build listen_alert statement for alerter. SQL String : %2.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330034	`Alerter thread throw exception %1 when execute listen_alert statement.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330035	`Alerter thread throw exception %1 when clean up alerter. SQL String : %2.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330036	`Publication Service <%1> thread <%2> throw exception %3 when manipulate fault status updater.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.

Status Code	Role	Category	Resolution
AEADB-330037	`Publication Service <%1> thread <%2> throw exception %3 when publish single message.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330038	`Publication Service <%1> thread <%2> throw exception %3 when manipulate S to N status updater.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330039	`Publication Service <%1> thread <%2> failed to construct minstance based on source table because no data fetched from %3.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330040	`Publication Service <%1> thread <%2> failed to construct minstance based on reference object because no data fetched from %3.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330041	`Publication Service <%1> thread <%2> will cause adapter terminate because the Exception: %3.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330042	`Publication Service <%1> thread <%2> throw exception %3 when build child selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330043	`Publication Service <%1> thread <%2> throw exception %3 when bind child selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330045	`Publication Service %1 thread %2 throw exception %3 when parse the message.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330046	`Publication Thread <%1> throws exception %2 when refresh connection.`		

Status Code	Role	Category	Resolution
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330047	`Publication Service <%1> thread <%2> throw exception <%3> when connect to metaspace <%4>.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330048	`Thread <%1> throw exception <%2> when create space <%3>.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330049	`Thread <%1> throw exception <%2> when get space <%3>.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330050	`Publication Service <%1> thread <%2> throw exception <%3> when insert data into <%4>.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330051	`Publication Service <%1> thread <%2> throw exception <%3> when upsert data into <%4>.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330052	`Publication Service <%1> thread <%2> throw exception <%3> when delete data from <%4>.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330053	`Publication Service <%1> thread <%2> throw exception <%3> when update data for <%4>.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330054	`Publication Service <%1> thread <%2> throw exception <%3> when format the <%4>.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330055	`Publication Service <%1> thread <%2> failed to connect to metaspace <%3>.`		

Status Code	Role	Category	Resolution
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330056	`Publication Service %1 thread %2 throw exception %3 when create queue connection to the database.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330057	`Publication Service %1 thread %2 throw exception %3 when create queue session to the database.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330058	`Publication Service <%1> thread <%2> failed to initialize spaces on the metaspace <%3>.`		
	errorRole	Publication Service	Normal operation; no action is necessary.
AEADB-330059	`Publication Service %1 thread %2 throw exception %3 when execute the max sequence selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-330060	`Publication Service %1 thread %2 throw exception %3 when build publishing table max sequence selector.`		
	errorRole	Publication Service	Check the Publication Service configuration and data in the publishing table.
AEADB-400001	`Subscription Service <%1> receive one message.`		
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-400002	`Subscription Service Thread <%1> receive a group message which contain %2 record(s).`		
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-400003	`Subscription Service Thread <%1> receive a single message.`		
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-400004	`Subscription Service Thread <%1> start to handle the %2 item for the group message.`		

Status Code	Role	Category	Resolution
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-400 005	`Subscription Service Thread <%1> batch commit timeout.`		
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-400 006	`Subscription Service Thread <%1> %2 row(s) affected.`		
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-400 007	`Subscription Service Thread <%1> send reply \n%2\n to destination %3.`		
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-400 008	`Subscription Service Thread <%1> begin to refresh connection.`		
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-400 009	`Subscription Service Thread <%1> finish refreshing connection.`		
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-400 010	`Subscription Service shared default session <%1> batch commit Information:%2.`		
	infoRole	Subscription Service	Normal operation; no action is necessary.
AEADB-410 001	`Subscription Service Thread <%1> received its own message. Discarding.`		
	warnRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421 001	`Subscription Service Thread <%1> start DB transaction.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421 002	`Subscription Service Thread <%1> commit DB transaction.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-421003	`Subscription Service Thread <%1> roll back DB transaction.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421004	`Subscription Service Thread <%1> bypass one message.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421005	`Subscription Service Thread <%1> the statement in cache cannot be reused.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421006	`Subscription Service Thread <%1> call pre-commit procedure successfully.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421007	`Subscription Service Thread <%1> create statement.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421008	`Subscription Service Thread <%1> bind statement.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421009	`Subscription Service Thread <%1> execute statement.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421010	`Subscription Service Thread <%1> flush Bulk-Insert.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421011	`Subscription Service Thread <%1> no data change in the database.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-421012	`Subscription Service Thread <%1> will rollback since %2.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB_422 001	`Subscription Service Thread <%1> Confirms %2 error message Success.`		
	debugRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB_422 002	`Subscription Service Thread <%1> Confirming %2 message Successfully.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB_422 003	`Subscription Service Thread <%1> statement SQL: %2.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB_422 004	`Subscription Service Thread <%1> will do incremental operation for child records.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB_422 005	`Subscription Service Thread <%1> will do completion operation for child records.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB_422 006	`Subscription Service Thread <%1> insert the data %2 into opaque exception table %3.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB_422 007	`Subscription Service Thread <%1> insert the data %2 into exception table %3.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-422 008	`Subscription Service Thread <%1> start handle normal exception.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-422 009	`Subscription Service Thread <%1> start handle bulk insert error exception.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-422 010	`Subscription Service Thread <%1> finish handle normal exception.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-422 011	`Subscription Service Thread <%1> finish handle bulk insert error exception.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-422 012	`Subscription Service Thread <%1> didn't confirm error message "%2"(TrackingId %3), since this error message can't be inserted into exception/opaque table.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 001	`Subscription Service Thread <%1> result of pre-commit stored procedure %2 is %3.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 002	`Subscription Service Thread <%1> statement bounds: %2.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 003	`Subscription Service Thread <%1> query related child records %2.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 004	`Subscription Service Thread <%1> delete related child records %2.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 005	`Subscription Service Thread <%1> cache statement %2 for table %3.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 006	`Subscription Service Thread <%1> Message body: %2.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 007	`Subscription Service Thread <%1> deactivate batch commit timer since %2.`		

Status Code	Role	Category	Resolution
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 008	`Subscription Service Thread <%1> activate batch commit timer.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 009	`Subscription Service <%1>, Message queue ID calculation time: %2.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 010	`Subscription Service <%1>, Dispatching the message to Queue ID: %2.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 011	`Subscription Service <%1> thread <%2> set empty string for column <%3>.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 012	`Subscription Service <%1> Thread <%2> prepare roll back the exception data.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 013	`Subscription Service <%1> Thread <%2> handle <%3> of <%4> need execute data.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 014	`Subscription Service <%1> Thread <%2> handle <%3> of <%4> error data.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 015	`Subscription Service <%1> Thread <%2> handle <%3> of <%4> continue operation data.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 016	`Subscription Service Thread <%1> continue handle bulk insert data.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-423017	`Subscription Service <%1> Thread <%2> cache data size <%3>, start handle exception\n %4.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423018	`Subscription Service <%1> Thread <%2> exception cache data size is <%3>, executed successful data size is <%4>.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423019	`Subscription Service <%1> Thread <%2> handle exception data <%3> of <%4>.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423020	`Subscription Service <%1> Thread <%2> handle group exception data <%3> item of <%4>.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423021	`Subscription Service <%1> Thread <%2> execute update count result %3.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423022	`Subscription Service Thread <%1> handle Batch data.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423023	`Subscription Service Thread <%1> can't execute the statement because of the statement is null.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423024	`Subscription Service Thread <%1> the single table opcode %2 is invalid, so discard the single message.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423025	`Subscription Service Thread <%1> the parent table opcode %2 is invalid, so discard the message.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-423 026	`Subscription Service Thread <%1> the child table opcode %2 is invalid, so discard the child table message.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 027	`Subscription Service Thread <%1> Service <%2> handle <%3> of <%4> need execute data.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 028	`Subscription Service <%1> Thread <%2> cache data size <%3>.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-423 029	`Subscription Service shared default session <%1> batch commit Information:%2.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-424 001	`Subscription Service Thread <%1> execute database statement successful count is <%2>.`		
	errorRole	Subscription Service	Normal operation; no action is necessary.
AEADB-424 002	`Subscription Service Parse the <%1> date time string [%2].`		
	errorRole	Subscription Service	Normal operation; no action is necessary.
AEADB-424 003	`Subscription Service Thread <%1> database operation in the batch: batch commit timeout ignored.`		
	debugRole	Subscription Service	Normal operation; no action is necessary.
AEADB-430 001	`Subscription Service <%1> received unexpected data %2.`		
	errorRole	Subscription Service	Check the message data.
AEADB-430 002	`Subscription Service <%1> throw exception %2 when parse data %3.`		
	errorRole	Subscription Service	Check the message data.

Status Code	Role	Category	Resolution
AEADB-430003	`Subscription Service <%1> throw exception %2 when calculate queueID for data %3.`		
	errorRole	Subscription Service	Check the message data.
AEADB-430004	`Subscription Service Thread <%1> throw exception %2 when check if the %3 is a kind of group message.`		
	errorRole	Subscription Service	Check the message data.
AEADB-430005	`Subscription Service Thread <%1> throw exception %2 when create statement for Data %3.`		
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430006	`Subscription Service Thread <%1> throw exception %2 when terminate adapter.`		
	errorRole	Subscription Service	Check the logs and the configuration.
AEADB-430007	`Subscription Service Thread <%1> throw exception %2 when bind parameter to statement %3.`		
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430008	`Subscription Service Thread <%1> throw exception %2 when execute statement %3.`		
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430009	`Subscription Service Thread <%1> throw exception %2 when create batch commit timer %3.`		
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430010	`Subscription Service Thread <%1> throw exception %2 when handle timeout event.`		

Status Code	Role	Category	Resolution
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430011	`Subscription Service Thread <%1> call pre-commit procedure unsuccessfully.`		
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430012	`Subscription Service Thread <%1> throw exception %2 when serialize the message %3.`		
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430013	`Subscription Service Thread <%1> throw exception %2 when activate batch commit timer %2.`		
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430014	`Subscription Service Thread <%1> throw exception %2 when deactivate batch commit timer %2.`		
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430015	`Subscription Service Thread <%1> handle [%2] : %3.`		
	errorRole	Subscription Service	Check the message data or the table to determine whether you need this message to throw the exception.
AEADB-430016	`Subscription Service Thread <%1> the exception table is not found.`		
	errorRole	Subscription Service	Check if you set the exception table.
AEADB-430017	`Subscription Service Thread <%1> the opaque exception table is not found.`		
	errorRole	Subscription Service	Check if you set the opaque exception table.
AEADB-430018	`Subscription Service Thread <%1> the child exception table %2 is not found.`		

Status Code	Role	Category	Resolution
	errorRole	Subscription Service	Check if you set the child exception table.
AEADB-430019	Subscription Service Thread <%1> throw exception %2 when begin transaction.		
	errorRole	Subscription Service	Check the database.
AEADB-430020	Subscription Service Thread <%1> send reply %2 to destination %3 Error data: %4.		
	errorRole	Subscription Service	Check the message data.
AEADB-430021	Subscription Service Thread <%1> throw exception %2 when roll back transaction.		
	errorRole	Subscription Service	Check the database.
AEADB-430022	Subscription Service Thread <%1> throw exception %2 when set custom scale.		
	errorRole	Subscription Service	Check the custom scale.
AEADB-430023	Subscription Service Thread <%1> throw exception %2 when refresh connection.		
	errorRole	Subscription Service	Normal operation; no action is necessary.
AEADB-430024	Subscription Service Thread <%1> can't bing the statement because of the statement is null.		
	errorRole	Subscription Service	Normal operation; no action is necessary.
AEADB-430025	`Subscription Service Thread <%1> the pre-commit procedure output parameter SP_TEXT is %2.`		
	errorRole	Subscription Service	Normal operation; no action is necessary.
AEADB-500001	`RPC Service Thread <%1> receive data.`		
	infoRole	RPC	Normal operation; no action is necessary.
AEADB-500002	`RPC Service Thread <%1> reply message to %2.`		
	infoRole	RPC	Normal operation; no action is necessary.
AEADB-500003	`RPC Service Thread <%1> Database operated successful.`		

Status Code	Role	Category	Resolution
	infoRole	RPC	Normal operation; no action is necessary.
AEADB-500004	`RPC Service Thread <%1> receive one message.`		
	infoRole	RPC	Normal operation; no action is necessary.
AEADB-500005	`RPC Service Thread <%1> Reply Message body is: %2.`		
	infoRole	RPC	Normal operation; no action is necessary.
AEADB-500006	`RPC Service Thread <%1> begin to refresh connection.`		
	infoRole	RPC	Normal operation; no action is necessary.
AEADB-500007	`RPC Service Thread <%1> finish refreshing connection.`		
	infoRole	RPC	Normal operation; no action is necessary.
AEADB-521001	`RPC Service Thread <%1> begin transaction.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-521002	`RPC Service Thread <%1> commit transaction.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-521003	`RPC Service Thread <%1> roll back transaction.`		
	debugRole	RPC	Check the message data and the operation logs.
AEADB-521004	`RPC Service Thread <%1> execute statement.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-521005	`RPC Service Thread <%1> create statement.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-522001	`RPC Service Thread <%1> statement SQL: %2.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-522002	`RPC Service Thread <%1> Confirming message Success.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-523001	`Request-Response (Custom mode) Service <%1> initialize Operation <%2>.`		
	debugRole	RPC	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-523 002	`RPC Service Thread <%1> statement binding data: %2.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-523 003	`<%1> Operation: %2.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-523 004	`RPC Service Thread <%1> do not found reply subject in the dateEvent.`		
	debugRole	RPC	Check the configuration.
AEADB-523 005	`RPC Service Thread <%1> Statement Cached: %2.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-523 006	`RPC Service Thread <%1> Receive Message body is: %2.`		
	debugRole	RPC	Normal operation; no action is necessary.
AEADB-530 001	`RPC Service Thread <%1> throw exception: %2 when create DB Statement for Data: %3.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 002	`RPC Service Thread <%1> throw exception: %2 when begin transaction.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 003	`RPC Service Thread <%1> throw exception: %2 when execute DB Statement %3.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 004	`RPC Service Thread <%1> throw exception: %2 when commit transaction.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 005	`RPC Service Thread <%1> throw exception: %2 when commit transaction.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 006	`RPC Service Thread <%1> throw exception: %2 when reply message to %3.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 007	`RPC Service Thread <%1> Database operated unsuccessful.`		
	errorRole	RPC	Check the message data and the operation logs.

Status Code	Role	Category	Resolution
AEADB-530 008	`RPC Service Thread <%1> throw exception: %2 when get DB procedure %3.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 009	`RPC Service Thread <%1> throw exception: %2 create operation %3.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 010	`RPC Service Thread <%1> throw exception: %2 when parse request %3.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 011	`RPC Service <%1> received unexpected data %2.`		
	errorRole	RPC	Check the message data and the operation logs.
AEADB-530 012	`RPC Service <%1> throw exception \n%2\n when refresh connection.`		
	errorRole	RPC	Normal operation; no action is necessary.
AEADB-530 013	`RPC Service <%1> throw exception \n%2\n when use attribute name %3 to get the value from the message.`		
	errorRole	RPC	Normal operation; no action is necessary.
AEADB-600 001	`Hawk Method %1 is invoked.`		
	infoRole	Hawk	Normal operation; no action is necessary.
AEADB-822 001	`%1 reconnect attempt %2.`		
	debugRole	Database	Check the database connection.
AEADB-823 001	`%1 reconnect failed on attempt %2.`		
	debugRole	Database	Check the database connection.
AEADB-823 002	`Reconnection thread will sleep %1ms.`		
	debugRole	Database	Check the database connection.
AEADB-823 003	`Oracle client info set successfully in the connection of <%1>...`		
	debugRole	Database	Normal operation; no action is necessary.
AEADB-921 001	`Advisory Handler receives RVCM Advisory on %1.`		
	debugRole	Advisory	Normal operation; no action is necessary.

Status Code	Role	Category	Resolution
AEADB-923 001	`Advisory Handler looks for the matched Publication Service by subject %1.`		
	debugRole	Advisory	Normal operation; no action is necessary.
AEADB-923 002	`Advisory Handler finds the matched Publication Service %1.`		
	debugRole	Advisory	Normal operation; no action is necessary.
AEADB-923 003	`Advisory Handler does not find the matched Publication Service.`		
	debugRole	Advisory	Normal operation; no action is necessary.
AEADB-923 004	`Received RV SYSTEM Advisory on %1 %2.`		
	debugRole	Advisory	Normal operation; no action is necessary.
AEADB-930 001	`Received SDK error Advisory on %1 %2.`		
	errorRole	Advisory	SDK error. Check the logs and the configuration.
AEADB-022 001	`Performance Status: %1.`		
	debugRole	Performance	Normal operation; no action is necessary.
AEADB-024 001	`Exception: %1.`		
	debugRole	Exception	Check the logs and the configuration.
AEADB-030 001	`Got SDK error in method <%1>: Error code: %2 Error Message: %3.`		
	errorRole	Exception	SDK error. Check the logs and the configuration.
AEADB-030 002	`Refresh performance logging failed, please enable the performance logging when start the adapter.`		
	errorRole	Performance	Check the logs and the configuration.

Adapter Microagents and Methods

TIBCO Hawk provides microagents and methods to supplement the monitoring information provided by the standard logging levels capability. These microagents and methods help monitor and debug the adapter.

Adapter Microagents

Each adapter has three microagents, with different capabilities and names. The same microagent follows different naming conventions, depending on how an adapter is started: from TIBCO Business Studio, or from TIBCO Administrator.

The following three microagents are predefined:

- Standard Microagent: predefined in TIBCO Adapter SDK. You can use the microagent to perform queries on all running adapters, regardless of their class or application.
  - Naming in TIBCO Business Studio: COM.TIBCO.ADAPTER.adadb.%%Deployment%%.%%InstanceId%%
  - Naming in TIBCO Administrator: COM.TIBCO.ADAPTER.adadb.domainName.%%Deployment%%.%%InstanceId%%
- Class Microagent: predefined in TIBCO Adapter SDK. You can use the microagent to perform queries on one class of the adapter.
  - Naming in TIBCO Business Studio: COM.TIBCO.adadb.%%Deployment%%.%%InstanceId%%
  - Naming in TIBCO Administrator: COM.TIBCO.ADAPTER.adadb.%%Deployment%%.%%InstanceId%%

  When an adapter configuration is started from TIBCO Business Studio, two class microagents with different number suffixes, like 0 and 1, are displayed in the main Hawk Display window. However, the class microagent with the smaller number suffix is invalid.
- Custom Microagent: predefined in TIBCO ActiveMatrix Adapter for Database. You can use the microagent to perform the adapter specific queries.
  - Naming in TIBCO Business Studio: COM.TIBCO.adadb.custom.%%Deployment%%.%%InstanceId%%
  - Naming in TIBCO Administrator: COM.TIBCO.adb.custom.%%Deployment%%.%%InstanceId%%

Available TIBCO Hawk Methods

The following lists the methods available in the custom microagent and the method in the SDK microagent of TIBCO ActiveMatrix Adapter for Database (TIBCO Business Studio).

Although the microagents, methods, and arguments dialog in TIBCO Hawk Display lists more methods than are documented here, only the following methods are supported for the adapter.

For the methods available in the standard and class microagents, see "Appendix C TIBCO Adapter SDK Hawk Microagents and Methods" in *TIBCO Adapter SDK Programmer's Guide*.

The following table lists the custom microagent methods:

Custom Method	Description
getEventQueueSize()	Retrieves the size of the TIBCO Rendezvous event queue for the specified adapter configuration.
setDebugLevel()	Sets the debug level for the current adapter configuration.
toggleVerboseFlag()	Changes the value of the `verbose` flag from `on` to `off`, or the reverse.
showConfiguration()	Shows the configuration defined for the current agent.
terminateADBagent()	Stops the current adapter agent.
getPollingInterval()	Returns the current polling interval setting.
setPollingInterval()	Sets the polling interval for Publication Service.
getPollingBatchSize()	Gets the polling batch size for Publication Service.
setPollingBatchSize()	Sets the polling batch size for Publication Service.

If you set the `adb.perfMon` property to `on`, the methods listed in the following table are displayed and available to use. If you set the `adb.perfMon` property to `off`, the methods are not displayed or useable.

The following table lists the custom microagent methods with the `adb.perfMon` property:

Custom Method	Description
getActivityStatistics()	Returns the total number of objects processed for all the schemas, based on the request type. Also, returns the number of success and error objects.
getActivityStatisticsByService	Returns the total number of objects processed for each of the schemas associated with the specified service.
getActivityStatisticsByOperation()	Returns the total number of objects processed for all the schemas by each service that is associated with a specified operation.
getThreadStatistics()	Returns the operation counts of the current threads.
getQueueStatistics()	Returns the current count of elements in any internal queue used by the adapter.
getConnectionStatistics()	Returns the state and statistics for all the current connections used by the adapter.
resetActivityStatistics()	Resets all the counts for the activity statistics.
resetThreadStatistics()	Resets all the counts for the thread statistics.
resetConnectionStatistics()	Resets all the counts for the queue statistics.

The following table lists the SDK microagent method:

Method	Description
getStatus()	Retrieves basic status information about the adapter. The getStatus method is provided by TIBCO Adapter SDK. Note the following information returned by this method: • Instance ID: instance ID for this adapter configuration • Adapter Name: name of the application • Messages Received: number of TIBCO Rendezvous messages received • Messages Sent: number of TIBCO Rendezvous messages published For more information about the getStatus method, see *TIBCO Adapter SDK Programmer's Guide*.  Information retrieved by the getStatus method does not indicate the records processed by the adapter services. The retrieved information contains not only records of the adapter services, but also records sent and received by the adapter itself for certain functions.

getEventQueueSize()

(Custom) Retrieves the size of the TIBCO Rendezvous event queue for the specified adapter configuration.

For more information about events and event queues, see *TIBCO Rendezvous Concepts*.

The following table lists the parameter:

Parameter	Type	Description
Session	string	Type of listener for this adapter configuration. Possible values are `Subscriber` or `Request/Reply`. The default value is `Subscriber`.

The following table lists the returns:

Return	Type	Description
QueueCount	integer	The number of events currently in the event queue.
QueueLimit	integer	The maximum number of events this adapter configuratrion can have in the event queue. This value is set using the `adb.rvMaxQueueSize` option in the adapter's properties file. The default value is `0`, which means the event queue has unlimited size.  If setting the value of `adb.rvMaxQueueSize` less than the message count, then the message might be lost.

setDebugLevel()

(Custom) Sets the debug level for the current adapter configuration.

The following table lists the parameter:

Parameter	Type	Description
DebugLevel	integer	Sets the debug level to 0 (off), 1, 2, or 3. 0— No debug information displayed. 1—SQL commands executed against the database shown. 2—Binding data for each SQL command shown. 3—All debug information displayed.

Returns OK if successful or an error if not successful.

toggleVerboseFlag()

(Custom) Changes the value of the verbose flag from on to off, or the reverse.

This method returns OK if successful or an error if not successful.

showConfiguration()

(Custom) Shows the configuration defined for the current agent.

The following table lists the parameters:

Parameter	Type	Description
VerboseInfo	string	The current setting for verbose mode, on or off.
DebugLevelInfo	string	The debug level setting: • 0—No debug information displayed. • 1—SQL commands executed against the database shown. • 2—Binding data for each SQL command shown.
PollIntervalInfo	string	The poll interval in milliseconds.

terminateADBagent()

(Custom) Stops the current adapter agent.

Invoking this method displays a warning message with the text `MicroAgent returned error. AMI Method invocation timed out`. The message can be ignored.

This method returns OK if successful or an error if not successful.

getPollingInterval()

(Custom) Returns the current polling interval setting.

The following table lists the return:

Return	Type	Description
PollingInterval	integer	Polling interval in milliseconds.

setPollingInterval()

(Custom) Sets the polling interval for Publication Service.

The following table lists the parameters:

Parameter	Type	Description
PollingInterval	integer	Polling interval in milliseconds.
ServiceName	string	(Optional) Name of service where the polling interval is set.

getPollingBatchSize()

(Custom) Gets the polling batch size for Publication Service.

The following table lists the return:

Return	Type	Description
BatchSize	integer	The batch size for Publication Service.

setPollingBatchSize()

(Custom) Sets the polling batch size for Publication Service.

The following table lists the parameter:

Parameter	Type	Description
BatchSize	integer	The new batch size for Publication Service.

getActivityStatistics()

(Custom with adb.perfMon) Returns the total number of objects processed for all the schemas, based on the request type. Also, returns the number of success and error objects.

The following table lists the parameter:

Parameter	Type	Description
Get Subtotal By	integer	Statistics categorized by service or by operation.

The following table lists the returns:

Returns	Type	Description
Name	string	Name of the service or operation. The operation can be one of the following: • Fetch • Poll • Insert • Update • Delete
Total	integer	Total number of objects processed for this schema for Publication Service. Total number of objects received for this schema for Subscription Service.
Success	integer	The number of objects that are successfully identified for this schema which will be published or written to a file.
Failure	integer	The number of objects that are identified for this schema but are not published because the header of the schema failed validation for the Publication Service, or is written to a file because the schema is not associated with the subscriber for a Subscription Service.

getActivityStatisticsByService

(Custom with adb.perfMon) Returns the total number of objects processed for each of the schemas associated with the specified service. Also, returns the number of success and error objects.

The following table lists the parameter:

Parameter	Type	Description
Service Name	string	Name of the service.

The following table lists the returns:

Returns	Type	Description
Operation	string	Type of operation that the service performs.
Schema Name	string	Name of the schema that is associated with the service.
Total	integer	Number of objects processed for this schema for Publication Service. Number of objects received for this schema for Subscription Service.
Success	integer	The number of objects that are successfully identified for this schema which will be published or written to a file.

Returns	Type	Description
Failure	integer	The number of objects that are identified for this schema but are not published because the header of the schema failed validation for Publication Service, or is written to a file because the schema is not associated with the subscriber for Subscription Service.

getActivityStatisticsByOperation()

(Custom with adb.perfMon) Returns the total number of objects processed for all the schemas by each service that is associated with a specified operation. Also, returns the number of success and error objects.

The following table lists the parameter:

Parameter	Type	Description
Operation	string	Name of operation. Pick from one of the following form the drop-down list: • Fetch • Poll • Insert • Update • Delete

The following table lists the returns:

Returns	Type	Description
Service Name	string	Name of the service that is associated with the specified operation.
Total	integer	Total number of objects processed for this schema for Publication Service. Total number of objects received for this schema for Subscription Service.
Success	integer	The number of objects that are successfully identified for this schema which will be published or written to a file.
Failure	integer	The number of objects that are identified for this schema but are not published because the header of the schema failed validation for Publication Service, or is written to a file because the schema is not associated with the subscriber for Subscription Service.

getThreadStatistics()

(Custom with adb.perfMon) Returns the operation counts of the current threads.

The following table lists the returns:

Returns	Type	Description
ThreadID	string	A unique identification of a particular thread.
ThreadType	string	A type or key that will match this thread to a queue or connection.
TaskType	string	Short description of the tasks this thread processes.
TaskCount	integer	Number of tasks processed by this thread.

getQueueStatistics()

(Custom with adb.perfMon) Returns the current count of elements in any internal queue used by the adapter. This includes the TIBCO Rendezvous event queues automatically spawned by TIBCO Rendezvous for each adapter.

The following table lists the returns:

Returns	Type	Description
QueueID	string	A unique identification of a particular queue.
QueueType	string	A type or key that will match this queue to a thread or connection.
QueueCount	integer	Current number of elements in the queue.
MaxQueueSize	integer	Maximum number of elements in the queue.

getConnectionStatistics()

(Custom with adb.perfMon) Returns the state and statistics for all the current connections used by the adapter.

The following table lists the returns:

Returns	Type	Description
Connection ID	string	A unique identification of a particular connection.
Connection Type	string	A type or key that matches this connection to a thread or queue.
State	string	Current state: `CONNECTED` or `DISCONNECTED`
NumRetries	integer	Total number of times this connection had to be reestablished.
TotalNumOperations	integer	Total number of operations processed by this connection since the adapter started.
CurrentNumOperations	integer	Total number of operations processed by this connection since the last reconnection.

resetActivityStatistics()

(Custom with adb.perfMon) Resets all the counts for the activity statistics.

resetThreadStatistics()

(Custom with adb.perfMon) Resets all the counts for the thread statistics.

resetConnectionStatistics()

(Custom with adb.perfMon) Resets all the counts for the connection statistics.