IBM InfoSphere DataStage and QualityStage
Version 8 Release 7

Connectivity Guide for Microsoft SQL Server and OLE DB Data

SC19-3437-01

IBM InfoSphere DataStage and QualityStage
Version 8 Release 7

Connectivity Guide for Microsoft SQL Server and OLE DB Data

SC19-3437-01

Note

Before using this information and the product that it supports, read the information in “Notices and trademarks” on page 61.

Contents

Chapter 1. Introduction

IBM® InfoSphere® DataStage® provides the ability to load tables in Microsoft SQL Server databases and read data from tables in SQL Server databases. InfoSphere DataStage also lets you retrieve information from any type of information repository, such as a relational source, an ISAM file, a personal database, or a spreadsheet.

InfoSphere DataStage provides several ways to access Microsoft SQL Server databases and as well as other sources of information (using Microsoft OLE DB). The following topics provide an introduction to the stages that accomplish this access. The later topics cover installation instructions and configuration information

This guide describes how to use
- BCPLoad stages
- SQL Server Enterprise stages
- MS SQL Server Load stages
- MS OLEDB stages

in InfoSphere DataStage server jobs.

These connectivity stages are installed automatically when you install InfoSphere DataStage. They appear in the **Database** category on the server job palette.

Although this guide contains several stages specifically designed to access Microsoft SQL Server databases, you can also access them by using the following stages:
- Dynamic Relational Stage (DRS)

  Use this stage to create a job that requires one relational database at design time and another at runtime. For more information about this stage, see the *Connectivity Guide for the Dynamic Relational Stage*.
- Stored Procedure (STP)

  Use this stage to include Microsoft SQL Server stored procedures as part of your InfoSphere DataStage job. For more information about this stage, see the *Connectivity Guide for the Dynamic Relational Stage*.

This guide is intended for:
- Designers who create or modify jobs that interface with SQL Server or OLE DB and
- InfoSphere DataStage administrators who install or upgrade InfoSphere DataStage.

SQL Server BCPLoad Stages

Microsoft SQL Server and Sybase have a utility called BCP (Bulk Copy Program). This command line utility copies SQL Server data to or from an operating system file in a user-specified format. BCP uses the bulk copy API in the SQL Server client libraries.

By using BCP, you can load large volumes of data into a table without recording each insert in a log file. You can run BCP manually from a command line by using command line options (switches). A format (*.fmt*) file is created which is used to load the data into the database.

SQL Server Load Stages

The SQL Server Load is a passive stage that bulk loads data into an SQL Server database table. This stage uses OLE DB SDK, which is part of Microsoft Data Access SDK (MDAC). It uses the native SQL Server OLE DB provider SQLOLEDB to bulk load the data.

Each input link to the stage represents a stream of rows to bulk load into an SQL Server table. The metadata for each input column determines how it is loaded. You can use any number of input links to this stage. Reference links and output links are not allowed.

For more information about Microsoft Data Access, consult your MDAC documentation.

MS OLEDB Stages

MS OLEDB is a passive stage that lets InfoSphere DataStage retrieve information from any type of information repository, such as a relational source, an ISAM file, a personal database, or a spreadsheet. This stage uses OLE DB SDK that is part of Microsoft Data Access SDK (MDAC). For more information about Microsoft Data Access, consult your MDAC documentation.

Note: You can also use this stage to create a local cube file, and load the cube with the data from the underlying database. You can use the CREATE CUBE and INSERT INTO statements to do this.

However, you cannot create a cube on the OLAP server yet since MSOLAP does not currently support this. Additionally, there is no GUI browser for local cube files, but you can modify the MDX sample application that comes with the Microsoft SQL Server for OLAP Services software to browse local cube files.

MS OLEDB provides a user-friendly client GUI. You can install the GUI separately from MS OLEDB on the server. Or, you can use the server MS OLEDB stage without installing the custom GUI on the client. If it is not installed, you can still use the property grid interface.

You can use any number of input, output, or reference output links with this stage:

- Input links specify the data you are writing, which is a stream of rows to be loaded into the data source.
- Output links specify the data you are extracting, which is a stream of rows to be read from the data source. You can specify the data on an input or an output link by using a SQL statement constructed by InfoSphere DataStage or a user-defined query.
- Each reference output link represents rows that are key read from the data source. The link uses the key columns in a WHERE clause of the SELECT statement that is constructed by InfoSphere DataStage or specified by the user.

The MS OLEDB stage can read and write data directly without SQL statements. Or, the stage can use SQL statements generated by the stage or user-defined SQL statements.

Installing the BCPLoad Stage

The BCPLoad stage installs automatically when you install InfoSphere DataStage.

Installing the MS SQL Server Load Stage

For instructions and information supporting the installation, see the *IBM InfoSphere Information Server Planning, Installation, and Configuration Guide*.

Installing the MS OLEDB Stage

For instructions and information supporting the installation, see *Enterprise Installation and Administration Guide*.

Configuring the Environment for BCPLoad

About this task

Before you can use the BCPLoad stage you must:

- Install and configure the SQL Server or Sybase client software. The BCPLoad stage uses the BCP API in the DBLIB/CTLIB and NetLIB client libraries. You must ensure that these components are installed on the InfoSphere DataStage server which is acting as a client to the SQL Server DBMS. See the documentation supplied with your DBMS for more details.
- Use one of the client tools (for example, ISQLW in the case of Microsoft SQL Server or WISQL32 for Sybase) to ensure that the connectivity between the InfoSphere DataStage server and the SQL Server host is operational.
- Create the table in the database on the SQL Server.
- Configure your database to use the fast copy (bulk load) option. By using this option, the data is loaded without each insert being recorded in a log file. If you do not specify this setting, all transactions are logged, slowing down the rate at which data is loaded. The fast copy option can be switched on by a stored procedure. For more information about using stored procedures, see "Using Stored Procedures".

There are some special points to note about SQL Server. If the following error is returned when you use the BCPLoad stage with data in YMD format, and the Date Format has been set:

```
Attempt to convert data stopped by syntax error in source field.
If your table contains date fields in ymd format
make sure the Date Format property is set
```

then clear the **Use International Settings** check box in the DB-Library option page of the SQL Server Client Network Utility.

If your job uses data in the upper 128 characters of the character set and the data is not appearing correctly on the database then clear the **Automatic ANSI to OEM conversion** check box in the DB-Library option page of the SQL Server Client Network Utility.

Reject Row Handling

You cannot use IBM InfoSphere DataStage Reject Row Handling capability if you write multiple rows at one time. If you want to process rejected rows, you must write one row at a time.

Connectivity Stages and the Parallel Canvas

SQL Server Enterprise stage is available only on the Parallel Canvas.

Neither SQL Server BCPLoad stage, MS SQL Server Load stage, nor MS OLEDB stage is available on the Parallel Canvas.

Chapter 2. BCPLoad Stages

This topic describes BCPLoad stages, which are used to bulk load data into a single table in a Microsoft SQL Server 2000 or Sybase (System 11.5 or 12.5) database.

The BCPLoad stage is a passive connectivity stage. The stage is installed automatically when you install InfoSphere DataStage and appears in the **Database** category on the server job palette.

This topic describes the following topics for the BCPLoad stage:
- "BCPLoad Stage Overview"
- "Table Definitions"
- "SQL Data Types"
- "Using BCPLoad Stages"
- "Defining BCPLoad Input Data"

BCPLoad Stage Overview

The BCPLoad stage uses the same API that BCP does, but loads data directly, without the need for a format file. The command line switches are set using stage properties.

By default, the BCPLoad stage is configured to bulk load data into Microsoft SQL Server. You can configure the BCPLoad stage properties to bulk load data into a Sybase SQL Server table using the Sybase DBLIB or CTLIB client libraries.

Note: The client libraries used by the BCPLoad stage are not supplied as part of InfoSphere DataStage. You must obtain these libraries from your DBMS vendor, and ensure they are installed and configured on your system, before attempting to use the BCPLoad stage.

Because this version of the BCPLoad stage supports both Microsoft SQL Server and Sybase, only BCP switches common to both servers have been included as stage properties. The following command line switches are not supported for Microsoft SQL Server:
- -T, trusted connection
- -q, quote identifiers

The following command line switches are not supported for Sybase:
- -I, interface file
- -J, the client character set
- -q, the data character set

For more information about the BCP switches that can be set, see "Editing Stage Properties".

The BCPLoad stage does not support the loading of native data files.

Table Definitions

About this task

You can import a table definition from the table in your SQL Server database by selecting **Import** > **Table Definitions** from the Designer menu. The table definition is imported via an ODBC connection to the Server. You can then load this table definition into the stage by clicking **Load** on the **Columns** tab on the BCPLoad stage Inputs page.

SQL Data Types

The following SQL Server data types are supported by the BCPLoad stage:

- Bit
- Char
- DateTime
- Decimal
- Float
- Integer
- Money
- Numeric
- Real
- SmallDateTime
- SmallInt
- SmallMoney
- TinyInt
- VarChar

When you import metadata from your database table, these data types are mapped to appropriate SQL data types by the ODBC driver. You can view the data types used in the table definition from the repository, or when you edit a stage in your job design.

The following SQL Server data types are not supported by the BCPLoad stage:

- Binary
- VarBinary
- Image
- Text (large text which is a binary type)

Using BCPLoad Stages

The BCPLoad stage is a target stage. It has one input link, which provides a sequence of rows to load into the SQL Server or Sybase database table. The metadata for each input column determines how it is loaded. There are no output links from this stage.

When you edit the BCPLoad stage, the **BCPLoad Stage** dialog box opens. This dialog box has two pages:

- **Stage**. Contains the name of the stage you are editing. This page has up to three tabs:

  - **General**. Contains an optional description of the stage and the stage type (BCPLoad).
  - **Properties**. Contains the stage properties and their current values. You can edit the default settings for the stage properties or specify job parameters. For details, see "Editing Stage Properties".
  - **NLS**. If NLS is enabled and you do not want to use the project default character set map, you can select an alternative character set map from this tab.
- **Inputs**. Specifies the column definitions for the data input link.

Click **OK** to close this dialog box. Changes are saved when you save the job.

Must Do's

About this task

This section specifies the minimum steps needed to get a BCPLoad stage functioning.

To use the BCPLoad stage:

Procedure

1. Start the Designer and open your server job design.
2. Click the **Sybase BCP Load** button on the tool palette.
3. Click in the Diagram window where you want to position the stage.
4. Link an output from a relevant stage in the job design to the input of the BCPLoad stage.
5. Configure the BCPLoad stage:
   a. Edit stage properties on the **Properties** tab or specify job parameters.
   b. Optionally define a character set map on the **NLS** tab if NLS is enabled.
   c. Define the data on the input links.

   These steps are performed in the BCPLoad Stage dialog box.

Editing Stage Properties

The **Properties** tab on the Stage page allows you to view and edit properties for the BCPLoad stage. It contains a grid displaying the following property names and values:

- **SQL-Server Name**. The name of the SQL Server to connect to. This property corresponds to the BCP -S switch. This property is optional and has no default value. If you leave this property blank, the stage assumes the SQL Server resides on the same machine as the InfoSphere DataStage Server.
- **User ID**. The logon name of the SQL user. This property corresponds to the BCP -U option. This property is required and there is no default value.
- **Password**. The password of the SQL user. This property corresponds to the BCP -P option. This property is required and there is no default value.
- **Database Name**. The name of the database to use on the SQL Server. This property is required and there is no default value.
- **Table Name**. The name of the table to load data into. This property is required and there is no default value.

- **Before Load Stored Procedure**. The name of a stored procedure that is executed before the database table is loaded. This property is optional and has no default value. For more information about using a before-load stored procedure, see "Using Stored Procedures".
- **After Load Stored Procedure**. The name of a stored procedure that is executed after the database table is loaded. This property is optional and has no default value. For more information about using an after-load stored procedure, see "Using Stored Procedures".
- **Batch Size**. The number of rows to include in the BCP batch. This property corresponds to the BCP -b option. The default setting for this property is 0, that is, all the data rows are treated in one batch. If an error occurs, all rows are rolled back.
- **Packet Size**. The number of bytes per network packet sent to and from the server. The default value is 4096. You can specify any number from 512 through 65535.
- **Use Source Identity Data**. This property corresponds to the BCP -E switch. Setting this property tells the SQL Server to use the identity values passed to it by the BCPLoad stage to populate the corresponding identity column in the SQL Server table.
- **Date Format**. This property provides a workaround to the problem that Microsoft SQL Server has with dates in *YMD* format. If your target table has a date column and your data has dates in *YMD* format, a conversion is required for the date to load successfully. By setting this property to `ymd`, dates are automatically converted during loading to a format that Microsoft SQL Server accepts.
- **Client Library**. The type of client library to use. The default setting is MSDBLIB (the Microsoft DBLibrary). Other valid settings are SYBDBLIB for the Sybase DBLibrary and SYBCTLIB for the Sybase CTLibrary. There are some restrictions on UNIX servers about which libraries you can use; refer to the *IBM InfoSphere Information Server Planning, Installation, and Configuration Guide* for details.

There are also four buttons on this tab:

- **Insert Job Parameter...**. Allows you to insert a job parameter as the value for a selected property. You can use job parameters for any of the properties on this tab. When you validate or run the job, you are prompted to specify suitable values for the properties.

  When you click this button, a list appears displaying the currently defined job parameters. Choose a parameter from the list or click (**New...**) to define a new one. The Job Properties dialog box appears with the **Parameters** tab displayed. You can also insert a job parameter by using the **F9** key.
- **Set to Default**. Sets the value for the selected property to the default value.
- **All to Default**. Sets the values for all properties to the default values.
- **Property Help**. Displays the help text supplied by the creator of the stage definition.

You can edit the value for any property listed in the grid. Click **OK** to save the settings and close the **BCPLoad** Stage dialog box.

Using Stored Procedures

You can specify the name of a stored procedure to run before or after loading the database. Before-load stored procedures can be used to perform tasks such as dropping indexes and turning on the database bulk copy option. After-load stored procedures can be used to turn off the bulk copy option and recreate any indexes. For a detailed description of how to write a stored procedure, see the SQL Server documentation.

The stored procedure name is entered as the value for the **Before Load Stored Procedure** or **After Load Stored Procedure** stage property. As well as entering the name of a stored procedure, you can also include parameter values. To specify parameters for the stored procedure, use the following format in the **Value** field on the **Properties** tab:

```
procedurename P1, P2, P3, ..., Pn
```

procedurename is the name of the stored procedure.

P1...Pn are parameter values, in the order expected by the stored procedure. Note that string values must be quoted.

If you want to return messages from a stored procedure and write them to the job log file, you can use the output parameters DSSeverity and DSMessage. These parameters return messages to the job log file with an appropriate severity. The type of message written to the job log file depends on the value returned by the DSSeverity parameter:

- Return value of 0. Nothing is written.
- Return value of 1. An informational message is written.
- Return value of 2. A warning message is written.
- Return value of 3. A fatal message is written. The InfoSphere DataStage job aborts and any return values from the stored procedure, other than the InfoSphere DataStage expected output parameters, are ignored.

The following example is of a before-load stored procedure. This stored procedure demonstrates the use of the output parameters DSSeverity and DSMessage:

```
create proc DemoBeforeSP
   @lReplace bit,
   @DSSeverity int output,
   @DSMessage varchar(255) = "" output
as
/* Remove the following three lines if running on Sybase */
declare @sSetDBOption varchar(255)
select @sSetDBOption = 'sp_dboption' + DB_NAME() + ", 'select
Â into/bulkcopy', TRUE"
exec (@sSetDBOption)
if @lReplace = 1
   begin
   truncate table BCPLoadSample
end
if @@ERROR = 0
   begin
      select @DSMessage = "Before SP completed: "
      if @lReplace = 1
         begin
         select @DSMessage = @DSMessage + "replacing existing data"
         end
      else
         begin
         select @DSMessage = @DSMessage + "appending data"
      end
      select @DSSeverity = 1                /* INFO */
   end
else
   begin
```

```
      select @DSMessage = "Before SP failed"
      select @DSSeverity = 2                /* WARNING */
end
GO
```

To use this stored procedure, type `DemoBeforeSP 1,DSSeverity, DSMessage` as the value for the **Before Load Stored Procedure** property when you edit stage properties.

To use existing stored procedures, type the name of the stored procedure and appropriate parameter values as the value for the **Before Load Stored Procedure** or **After Load Stored Procedure** property.

For example, suppose your stored procedure includes this:

```
create proc sp_TrustyDebuggedProcedure
   @strTableName char(30),
   @strSurname char(30),
   @iRowCount int = 0 output
as
...
...
```

If you want to use this procedure as a before-load procedure, you would type `sp_TrustyDebuggedProcedure "Table1","Smith"` in the **Value** field for the **Before Load Stored Procedure** property. "Table1" and "Smith" are passed in as **strTableName** and **strSurname** respectively.

If you want to modify an existing stored procedure to return a severity warning and an error message, the create procedure needs to be modified to include the two output parameters DSSeverity and DSMessage. In the earlier example, the create procedure would become:

```
create proc sp_TrustyDebuggedProcedure
   @strTableName char(30),
   @strSurname char(30),
   @iRowCount int = 0 output,
   @DSSeverity int output,
   @DSMessage varchar(255) = "" output
as
...
.../* Somewhere in the procedure set appropriate
values for DSSeverity and DSMessage*/
```

In this case, you would type the following text in the **Value** field for the **Before Load Stored Procedure**:

```
sp_TrustyDebuggedProcedure "Table1","Smith",0,DSSeverity,DSMessage
```

You can include job parameters to represent the value of a stored procedure parameter. To use job parameters in the earlier example, you would type the following text in the **Value** field for the **Before Load Stored Procedure**:

```
sp_TrustyDebuggedProcedure #Table#,#Name#,0,DSSeverity,DSMessage
```

Table and *Name* are the names of two defined job parameters.

Defining Character Set Maps

About this task

You can define a character set map for the BCPLoad stage by using the **NLS** tab on the Stage page. You can choose a specific character set map from the list, or accept the default setting for the whole project. This tab also has the following options:

- **Show all maps**. Displays all the maps supplied with InfoSphere DataStage. Maps cannot be used unless they have been loaded by using the IBM InfoSphere DataStage and QualityStage® Administrator.
- **Loaded maps only**. Displays the maps that are loaded and ready for use.
- **Use Job Parameter...**. Allows you to specify a character set map as a parameter to the job containing the stage. If the parameter has not yet been defined, you are prompted to define it from the Job Properties dialog box.

Defining BCPLoad Input Data

When you write data to a file in BCP load format, the BCPLoad stage has an input link. The properties of this link and the column definitions of the data are described on the Inputs page in the BCPLoad Stage dialog box. This page has two tabs:

- **General**. Contains an optional description of the link.
- **Columns**. Contains the column definitions for the data you are loading into your database table. These column definitions are usually specified by the metadata defined on the output link of the previous stage in the job design. If the columns are not already defined, you can click **Load** to load columns from a table definition in the repository, or you can type column definitions manually and click **Save...** to save them as a table definition.

Chapter 3. SQL Server Enterprise Stage

Overview of SQL Server Enterprise Stage

The SQL Server enterprise stage extends the capabilities of IBM InfoSphere DataStage to communicate with external data sources. Currently, connectivity to external data sources from the parallel canvas of InfoSphere DataStage is limited to four databases: IBM DB2®, Informix®, Oracle, and Teradata.

SQL Server enterprise stage operates in four modes:
- Read
- Write
- Upsert
- Lookup

Read

Use the SQL Server enterprise stage in **Read** mode to read records from an external data source table and place them in an IBM InfoSphere DataStage output resource such as a dataset or a sequential file.

Write

Use the SQL Server enterprise stage in **Write** mode to configure a connection to an external data source and insert records into a table. The Stage takes records from a single input resource such as a dataset, sequential file, and so on. The **Write** mode determines how the records of a dataset are inserted into the table.

Upsert

Use the SQL Server Enterprise stage in **Upsert** mode to insert/update/delete/insert and then update/update and then insert records into an external data source table. You can match records based on field names.

Lookup

Use the SQL Server enterprise stage in **Lookup** mode to perform a **join** operation between one or more external data source tables and an IBM InfoSphere DataStage input resource such as a dataset or a sequential file. The data output is an InfoSphere DataStage dataset.

Connecting to the SQL server

You can connect to Microsoft SQL Server by using an existing ODBC Data Source connection on Windows.

Procedure

1. Open the ODBC Data Source Administrator on Windows.
2. Create a Data Source entry by using the Microsoft SQL Server client driver.

3. Make sure that you test the Data Source connection by using the Microsoft SQL Server client configuration before using it in any SQL Server Enterprise stage Jobs.

Stage Page

This page is always present. You use this page to specify general information about the SQL Server enterprise stage. The following tabs are present on this page:

- **General:** Use the **General** tab to specify an optional description of the SQL Server enterprise stage.
- **Advanced:** Use the **Advanced** tab to configure properties that determine how the SQL Server enterprise stage behaves. You can usually ignore this tab and accept the default values. This tab is primarily intended for advanced users to finely tune operations.
  - **Execution mode -** This is one of the advanced options available. This option determines the mode in which an IBM InfoSphere DataStage job is executed - parallel or sequential. The value for the **Execution mode** option is set automatically; you cannot change this value. If the SQL Server enterprise stage is operating on only one file and there is one reader, the execution mode is sequential. Otherwise, it is parallel.

Input Page

Use the Input page to provide specific information about how data is being transferred from the SQL Server Enterprise stage to a remote host by using the SQL Server protocol. You can view this page only from the **Write** and **Upsert** modes of the SQL Server Enterprise stage.

Input Page in Write Mode

The Input page in the **Write** mode displays the following tabs and buttons:

- “General tab”
- “Properties tab”
- “Partitioning tab” on page 18
- “Columns tab” on page 18
- “Advanced tab” on page 18
- “Columns button” on page 18

General tab

Use the **General** tab to enter a description of the Input page. This is an optional feature.

Properties tab

Use the **Properties** tab to specify properties that determine the activities of the SQL Server enterprise stage. These properties are displayed in a tree structure and are divided into categories for ease of navigation. All the mandatory properties are included in the tree by default and cannot be removed. The properties that do not have a default value and need you to specify values are displayed in red, along with a question mark. When you set a value for such a property, the text color changes to black and the question mark disappears.

The following table is a quick reference list of the properties and their attributes. A more detailed description of each property follows.

Table 1. Input Properties and Attributes

Category/Property	Values	Default	Required?	Dependent on
Target/Table	Name of the table	N/A	Y	Y
Target/Write Method	Upsert, Write	N/A	Y	N
Target/Write Mode	Append, Create, Replace, Truncate	N/A	Y	N
Connection/ Data source	Data source	N/A	Y	Y
Connection/ User	User name	N/A	Y	Y
Connection/ Password	Password	N/A	Y	Y
Options/Insert Array Size	Integer	Y	N	N
Options/Truncate Column Name	False/True	Y	Y	N
Options/Open Command	Open Command	N	N	N
Options/Close Command	Close Command	N	N	N
Options/Length to Truncate	Integer	N	N	N
Options/Isolation Level	Read Uncommitted, Read Committed, Repeatable Read, Serializable	Y	N	N
Options/Drop Unmatched Field	True/False	Y	N	N

Target

Specify the **Table**, **Write Method** and **Write Mode** values here.

- **Table:** Specify the appropriate value here to connect the SQL Server enterprise stage to a target file located in a remote host.
- **Write Method: Specify** the appropriate value here to write and export data into a single table. You must set this value to **Write** so that the SQL Server enterprise stage operates in **Write** mode.
- **Write Mode:** Specify the appropriate value here to define how the records from the data source are inserted into the destination table. The **Write** mode can have one of the following values. Note that each of the below modes requires specific user privileges.
  - **Append -** This is the default mode. The **Append** mode requires that the destination table exists and the record schema of the dataset is compatible with the schema of the table. In this mode, the **write** operation appends new rows to the existing destination table. The schema of the existing table determines the input interface of the stage.
  - **Create -** In this mode, the **write** operation creates a new destination table. If a table exists with the same name as the one being created, the operation

terminates and an error message is displayed. The schema of the IBM InfoSphere DataStage dataset determines the schema of the new table. The table is created with simple default properties. To create a table with any properties other than the default properties, such as partitioned, indexed, or in a non-default table space, use the `-createstmt` option with your own `-createtable` statement.

  - **Replace -** In this mode, the **write** operation drops the table and creates a new one if the table with the same name exists. If a table with the specified name does not exist, the write operation creates a new table. The schema of the InfoSphere DataStage data set determines the schema of the new table.
  - **Truncate -** This mode requires an destination table. In this mode, the **write** operation retains the attributes of the destination table, but discards existing records and appends new records. The schema of the existing table determines the input interface of the SQL Server enterprise stage.

Connection

Under this category, you specify values for the **Data source**, **Password** and **User** fields.

- **Data source:** This is a mandatory field. Specify the database connection in this field by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **Data source** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.
- **Password:** This is a mandatory field. Specify in this field the password for connecting to the data source by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **Password** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.
- **User:** This is a mandatory field. Specify in this field the user name for connecting to the data source by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **User** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

Using the InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.

Note: If you have inserted all or some of the **Connection** category values from the job parameter list popup, then the job for which you provided these specifications takes the environment variables from the operating system. At runtime, you are prompted to modify the values for those environment variables.

Options

Under this category, you specify values for **Insert Array Size**, **Truncate Column Names**, **Close Command**, **Length to Truncate**, **Drop unmatched field**, **Open Command**, and **Isolation level** properties. Under the **Options** category, the **Truncate Column Names** property appears by default. You can add the other properties mentioned above from the **Available properties to add** list.

- **Insert Array Size:** In this field, you specify the size of the insert host array. You can only enter an integer in this field. The default value is 2000.
- **Truncate Column Names:** You can set any of two values below, depending upon your requirement:
  - **True -** Set this value to indicate that column names will be truncated to the size allowed by the SQL Server driver.
  - **False -** Set this value to disable truncating of column names.
- **Close Command:** Enter the SQL statement to be executed after an insert array is processed. This statement is executed only once on the conductor node.
- **Length to Truncate:** Specify the appropriate value for this option to indicate the length at which you want column names to be truncated.
- **Drop unmatched field:** You can set one of the values below, depending upon your requirement:
  - **True -** Set this value to indicate that unmatched columns in the dataset should be dropped. An unmatched column is a column for which there is no identically named column in the data source table.
  - **False -** This is the default value. This value indicates that unmatched fields of the dataset will not be dropped.
- **Open Command:** Enter the SQL statement to be executed before the insert array is processed. This statement is executed only once on the conductor node.
- **Isolation level:** Select the isolation level for accessing data from five available options:
  - Read
  - Uncommitted
  - Read Committed
  - Repeatable Read
  - Serializable

  The database specified in the data source determines the default isolation level.

Partitioning tab

Use the **Partitioning** tab to specify details about how the incoming data is partitioned or collected before operations are performed on the data. You can also specify that the data should be sorted before the operations are performed. The default mode of partitioning data is the **Auto Mode**. Use the partitioning or collection options to determine the partitioning method that you require. However, the partitioning method that you can set up depends upon the execution mode settings of the current and any preceding stages, as well as the number of nodes that is specified in the **Configuration** file.

If the SQL Server enterprise stage is operating in sequential mode, collection of data takes place before writing the data to the destination file by using the default **Auto** collection method. Use the settings available under the **Partitioning** tab to override this default behavior. The exact behavior of the data partitioning or collection method depends on:
- Whether the stage is set to execute in parallel or sequential mode.
- Whether the preceding stage in the job is set to execute in parallel or sequential mode.

Columns tab

This tab is displayed on all **Input** and **Output** pages.

Use this tab to view and modify column metadata for the Input or Output link that you have selected. Use the **Save** button to save any modifications that you make in the column metadata. Use the **Load** button to load an existing source table. You must select the appropriate table to load and click **OK**. The Select Column dialog is displayed. To ensure appropriate conversion of data types, clear the **Ensure all Char columns use Unicode** check box at the bottom of this dialog.

Advanced tab

Use the **Advanced** tab to specify how input or output data for the SQL Server enterprise stage is buffered. By default, data is buffered so that no deadlocks can arise. A deadlock is a situation in which a number of stages are mutually dependent. An example is a situation in which one or more stages are waiting for input from another stage, and they cannot output data until they have received it.

Columns button

Use the **Columns** button to define a list of column names for the destination table.

Input Page in Upsert Mode

The **upsert** method is one of the options of the **write** method. The Input page in **Upsert** mode has the following tabs and buttons:
- “General tab” on page 19
- “Properties tab” on page 19
- “Partitioning tab” on page 21
- “Columns tab” on page 21
- “Advanced tab” on page 21
- “Columns button” on page 21

General tab

Use the **General** tab to enter an description for the Input page. This is an optional feature.

Properties tab

Use the **Properties** tab to specify properties that determine the activities of the SQL Server enterprise stage. These properties are displayed in a tree structure and are divided into categories for ease of navigation. All the mandatory properties are included in the tree by default and cannot be removed. The properties that do not have a default value and need you to specify values are displayed in red, along with a question mark. When you set a value for such a property, the text color changes to black and the question mark disappears.

Target

Under this category, you specify values at least for the **Insert SQL**, **Update SQL**, **Delete SQL**, **Upsert mode**, and **Write Method** properties. Based on the values that you select for the **Upsert mode** , you see additional properties. Specify values for these additional properties depending on your requirement.

- **Insert SQL**: The SQL INSERT statement to be executed by the Upsert Write method.
- **Update SQL**: The SQL UPDATE statement to be executed by the Upsert Write method.
- **Delete SQL**: The SQL DELETE statement to be executed by the Upsert Write method.
- **Upsert mode:** The mode to be used when two statement options are specified. If only one statement option is specified, Upsert Mode is ignored. Upsert mode has the following properties.

Table 2. Upsert Mode Properties

Property	Description
Delete then Insert	The delete statement is executed first. Then the insert statement is executed.
Insert then Update	The insert statement is executed first. If the insert fails due to a duplicate key violation (that is, record exists), the update statement is executed. This is the default upsert mode.
Update then Insert	The update statement is executed first. If the update fails because the record doesn't exist, the insert statement is executed.

- **Write Method:** Select an appropriate write method to write and export data into a single table. **Upsert** uses an **insert** and an **update** method on one or more tables.

Connection

Under this category, you specify values for **Data source**, **Password** and **User**.

- **Data source:** This is a mandatory field. Specify the database connection in this field by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **Data source** field on the right side of the Properties page.

  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the IBM InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.
- **Password:** This is a mandatory field. Specify in this field the password for connecting to the data source by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **Password** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.
- **User:** This is a mandatory field. Specify in this field the user name for connecting to the data source by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **User** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.

    Note: If you have inserted all or some of the **Connection** category values from the job parameter list popup, then the job for which you provided these specifications takes the environment variables from the operating system. At runtime, you are prompted to modify the values for those environment variables.

Options

Under the **Options** category, you specify values for **Open Command**, **Close Command**, **Output Reject Records**, and **Insert Array Size** properties. Under the **Options** category, the **Output Reject Records** property appears by default. You can add the other properties mentioned above from the **Available properties to add** list.

- **Open Command:** Specify in single quotes a command to be parsed and executed by the SQL Server database on all processing nodes before the SQL Server table is opened. You can specify this value as a job parameter.

- **Close Command:** Specify in single quotes a command to be parsed and executed by the SQL Server database on all processing nodes after the SQL Server Enterprise stage completes processing the SQL Server table. You can specify this value as a job parameter.
- **Output Reject Records:** Select one of the below values:
  - **True:** Select this value to indicate that the rejected records should be sent to the reject link.
  - **False:** This is the default value. Select this value to indicate that rejected records should not be sent to the reject link.
- **Insert Array Size:** Specify the size of the insert host array. This property only accepts an integer. The default value is 2000.

Partitioning tab

Use the **Partitioning** tab to specify details about how the incoming data is partitioned or collected before operations are performed on the data. You can also specify that the data should be sorted before the operations are performed. The default mode of partitioning data is the **Auto Mode**. Use the partitioning or collection options to determine the partitioning method that you require. However, the partitioning method that you can set up depends upon the execution mode settings of the current and any preceding stages, as well as the number of nodes that is specified in the **Configuration** file.

If the SQL Server enterprise stage is operating in sequential mode, collection of data takes place before writing the data to the destination file by using the default **Auto** collection method. Use the settings available under the **Partitioning** tab to override this default behavior. The exact behavior of the data partitioning or collection method depends on:
- Whether the stage is set to execute in parallel or sequential mode.
- Whether the preceding stage in the job is set to execute in parallel or sequential mode.

Columns tab

This tab appears on all Input and Output pages. Click this tab to view column meta data for the input or output link that you have selected.

Advanced tab

Use the **Advanced** tab to specify how input and output data for this stage is buffered. By default, data is buffered so that no deadlocks can arise. A deadlock is a situation in which a number of stages are mutually dependent. An example is a situation in which one or more stages are waiting for input from another stage, and they cannot output data until they have received it.

Columns button

Use the **Columns** button to define a list of column names for the destination table. If the SQL Server Enterprise stage editor for the Input page in **Read** or **Upsert** mode shows both input and output links, then you can choose one of them to function as the reject link.

A reject link contains raw data for columns rejected due to schema mismatch, after the SELECT statement that you specified is executed.

Output Page

Use the Output page to provide details about the link to the SQL Server Enterprise stage from a remote host. By using this link, you can access data from a remote host through SQL Server. The Output page appears both in the **Read** and **Lookup** modes of the SQL Server Enterprise stage. In the **Read** mode, the SQL Server Enterprise stage only has an output link. In the **Lookup** mode, the stage has an output link as well as a reference link that connects the stage to a lookup stage.

Output Page in Read Mode

The Output page in **Read** mode displays the following buttons and tabs:

- “General tab”
- “Properties tab”
- “Columns tab” on page 24
- “Advanced tab” on page 24
- “Columns button” on page 24
- “View Data button” on page 24

General tab

Use the **General** tab to enter a description for the Output page in **Read** mode. This is an optional feature.

Properties tab

The **Properties** tab in the Output pages displays a screen with the same fields as the **Properties** tab in the Input pages. The only difference is that you do not see the **Target** category, and you see the **Source** category.

Source

Under this category, you specify values for **Read Method** and **Table**.

- **Table:** If you have chosen **Table** as the read method, then you must specify the name of the source SQL Server table. Note that the specified table must exist and you must have **SELECT** privileges for this table. If your SQL Server user name does not correspond to the owner of the specified table, you can prefix it with a table owner. You must add a new job parameter to fix the table owner name.

  To fix the table owner name:

  1. Click **Table** and then the arrow on the right side of the dialog.
  2. Click **Insert job parameter** and then **[New...]** from the popup list.
  3. In the Job Properties dialog that appears, enter the required table details in **Default Value** column for the **$user** parameter. Use the below format:

     ```
     table_owner.table_name
     ```

     Before you select a value for this option, you must fulfil the below dependencies:
  4. Use the **WHERE** clause in your **SELECT** statement to specify the rows of the table to be included or excluded from the read operation. If you do not supply a **WHERE** clause, all rows are read.

5. You can specify in your **SELECT** statement the columns that you wish to read. You must specify the columns in this list in the same order as they are defined in the record schema of the input table.

- **Read Method:** Use this property to specify a table or a query for reading the SQL Server database. The default value for **Read Method** is **Table**. If you choose **Table** as the **read** method, then you must specify the data source table for the **Table** option Alternatively, you can setup **Read Method** as an SQL query. In that case, you must specify whether you want the query to be generated automatically or you want to define the query yourself.

Connection

Under this category, you specify the **Data source**, **Password** and **User** values.

- **Data source:** This is a mandatory field. Specify the database connection in this field by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **Data source** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the IBM InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.
- **Password:** This is a mandatory field. Specify in this field the password for connecting to the data source by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **Password** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.
- **User:** This is a mandatory field. Specify in this field the user name for connecting to the data source by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **User** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.

Note: If you have inserted all or some of the **Connection** category values from the job parameter list popup, then the job for which you provided these specifications takes the environment variables from the operating system. At runtime, you are prompted to modify the values for those environment variables.

Options

Under this category, you specify values for **FetchArraySize**, **Isolation Level**, **Close Command** and **Open Commands**. All of these properties are optional. You see these properties in the **Available properties to add** list that appears in the bottom right corner of the Output page. To add any of these subproperties under **Options**, click **Options**, and then the property that you wish to add from the **Available properties to add** list.

- **Fetch Array Size:** Specify the number of rows to retrieve during each **fetch** operation. The default value is 1.
- **Isolation Level:** Enter the isolation level for accessing data. Choose from the four available options:
  - Read Committed
  - Read Uncommitted
  - Repeatable Read
  - Serializable

  The database that you specified for the **Data source** option (see the **Data source** section on the above) determines the default isolation level.
- **Close Command:** Enter an SQL statement to be executed after the insert array is processed. You cannot commit work by using this option. The statements are executed only once on the conductor node.
- **Open Command:** Enter an SQL statement to be executed before the insert array is processed. The statements are executed only once on the conductor node.

Columns tab

Like the Input pages, Output pages have a **Columns** tab by default. Clicking the **Columns** tab displays the column metadata for the link that you have selected.

Advanced tab

Use the **Advanced** tab to specify how input and output data for the SQL Server Enterprise stage is buffered. By default, data is buffered so that no deadlocks can arise. A deadlock is a situation in which a number of stages are mutually dependent, they are waiting for input from another stage, and they cannot output data until they have received it.

Columns button

Use the **Columns** button to define a list of column names for the output table.

View Data button

To view the number of rows in the table that you specified for the **Table** option under **Source**, click the **View Data** button. You can specify the number of rows that you wish to view at a time.

Output Page in Lookup Mode

The Output page in **Lookup** mode displays the following tabs and buttons:
- "General tab"
- "Properties tab"
- "Columns tab" on page 27
- "Advanced tab" on page 27
- "Columns button" on page 27

General tab

Use the **General** tab to enter an description for the Output page in **Lookup** mode. This is an optional feature.

Properties tab

Use the **Properties** tab to set appropriate values for **Source**, **Connection**, and **Options** properties.

Source

Under this category, you specify values for **Table**, **Read Method** and an additional property **Lookup Type**.

Lookup Type: You can choose **Normal** or **Sparse** as the lookup method.
- **Normal:** This is the default lookup method. A normal lookup is an in-memory lookup on an SQL Server database table. In case of a normal lookup, the **lookup** stage can have multiple reference links.
- **Sparse:** A sparse lookup accesses the source database directly. In case of a sparse lookup, the **lookup** stage has one reference link.
- **Table:** If you have chosen **Table** as the read method, then you must specify the name of the source SQL Server table. you can add the additional property Lookup Key. Note that the specified table must exist and you must have **SELECT** privileges for this table. If your SQL Server user name does not correspond to the owner of the specified table, you can prefix it with a table owner. You must add a new job parameter to fix the table owner name.

  To fix the table owner name:
  1. Click **Table** and then the arrow on the right side of the dialog.
  2. Click **Insert job parameter** and then **[New...]** from the popup list.
  3. In the Job Properties dialog that appears, enter the required table details in the **Default Value** column for the **$user** parameter. Use the below format:
     ```
     table_owner.table_name
     ```
     Before you select a value for this option, you must fulfill the below dependencies:
  4. Use the **WHERE** clause in your **SELECT** statement to specify the rows of the table to be included or excluded from the read operation. If you do not supply a **WHERE** clause, all rows are read.
  5. You can specify in your **SELECT** statement the columns that you wish to read. You must specify the columns in this list in the same order as they are defined in the record schema of the input table.

- **Lookup Key:** A lookup key is a column in the destination table that is used to join with a identically named column in a dataset. You can specify multiple lookup keys.
- **Read Method:** Specify **Table** or **User-defined SQL** as the read method.

Connection

Under this category, you specify the **Data source**, **Password** and **User** values.

- **Data source:** This is a mandatory field. Specify the database connection in this field by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **Data source** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the IBM InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.
- **Password:** This is a mandatory field. Specify in this field the password for connecting to the data source by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **Password** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.
- **User:** This is a mandatory field. Specify in this field the user name for connecting to the data source by using any one of the methods below:
  - **Method 1:** Enter the data source name in the **User** field on the right side of the Properties page.
  - **Method 2:** Insert the desired value as a job parameter. Click the pointer button on the extreme right side of the Properties page, and then **Insert Job Parameters**.

    A popup appears with a list of available job parameters from which you can choose. If you wish to create a new parameter for the job, click **[New...]** from the popup list, and create an appropriate environment variable by using the Job Properties dialog that appears.

    Using the InfoSphere DataStage and QualityStage Administrator, you can also create parameters at the project level for all jobs within the project.

    Note: If you have inserted all or some of the **Connection** category values from the job parameter list popup, then the job for which you provided these specifications takes the environment variables from the operating system. At runtime, you are prompted to modify the values for those environment variables.

Options

Under this category, specify the **Fetch Array Size**, **Isolation Level**, **Close Command** and **Open Commands**. All of these properties are optional. You see these properties in the **Available properties to add** list that appears in the bottom right corner of the Output page. To add any of these subproperties under **Options**, click **Options**, and then the property that you wish to add from the **Available properties to add** list. Note that Isolation Level appears on the list only if you select **Lookup Type** as **Normal**.

- **Fetch Array Size:** Specify the number of rows to retrieve during each **fetch** operation. The default value is 1.
- **Isolation Level:** Enter the isolation level for accessing data. Choose from the four available options:
  - Read Committed
  - Read Uncommitted
  - Repeatable Read
  - Serializable

  The database that you specified for the **Data source** option (see the **Data source** section above) determines the default isolation level.
- **Close Command:** Enter an SQL statement to be executed after the insert array is processed. You cannot commit work by using this option. The statements are executed only once on the conductor node.
- **Open Command:** Enter an SQL statement to be executed before the insert array is processed. The statements are executed only once on the conductor node.

Columns tab

The **Input** and Output pages always have a **Columns** tab. Click this tab to view column metadata for the link that you have selected.

Advanced tab

Use the **Advanced** tab to specify how input and output data for the SQL Server Enterprise stage is buffered. By default, data is buffered so that no deadlocks can arise. A deadlock is a situation in which a number of stages are mutually dependent. An example is a situation in which one or more stages are waiting for input from another stage, and they cannot output data until they have received it.

Columns button

Use the **Columns** button to define a list of column names for the output table.

Output Page in Upsert Mode with a Reject Link

If the SQL Server Enterprise stage editor in **Lookup** or **Write** mode shows both input and output links, then you can choose any of those links as the reject link. A reject link contains raw data for columns rejected due to schema mismatch, after your SELECT statement is executed.

The Output page for a reject link has the following tabs and buttons:
- General tab
- Properties tab
- Columns tab
- Advanced tab
- Columns button

General tab

Use the **General** tab to enter a description of the Output Page in **Upsert** mode for a reject link. This is an optional feature.

Properties tab

Use the **Properties** tab to specify the output options in **Upsert** mode for a reject link.

Columns tab

The **Input** and Output pages have a **Columns** tab by default. Clicking this tab displays the column metadata for the selected that you have selected.

Advanced tab

Use the **Advanced** tab to specify how input or output data for the SQL Server Enterprise stage is buffered. By default, data is buffered so that no deadlocks can arise. A deadlock is a situation in which a number of stages are mutually dependent. An example is a situation in which one or more stages are waiting for input from another stage, and they cannot output data until they have received it.

Columns button

Use the **Columns** button to define a list of column names for the output table.

Chapter 4. MS SQL Server Load Stages

This topic describes the following for MS SQL Server Load stages:
- Functionality of MS SQL Server stages
- Stage properties
- Link properties

Functionality of MS SQL Server Load Stages

The MS SQL Server Load has the following functionality and benefits:
- Support for different actions to take before loading data into tables. (Load Action)
- The ability to specify the number of rows to be written to the target table. (Commit Size)
- The ability to specify the isolation level used for transactions. (Transaction Isolation Level)
- The ability to control the type of tracing information to add to the log. (Trace Level)
- Support for supplying the identity column value in the data or by the SQL server. (Use Source Identity Data)
- The ability to specify SQL statements to be run before and after processing job data rows. (Before/After Load Statements)
- Support for NLS (National Language Support).

Meta data import is not supported.

Loading a SQL Server Database

About this task

Using the IBM InfoSphere DataStage and QualityStage Designer:

Procedure

1. Add an MS SQL Server Load stage to your InfoSphere DataStage job.
2. Link the MS SQL Server Load stage to its data source.
3. Specify column definitions by using the **Columns** tab.
4. Add the appropriate stage property values on the **Properties** tab of the Stage page, as documented in "Stage Properties".
5. Add the appropriate property values on the **Properties** tab of the **Inputs** page, as documented in "Link Properties".
6. Compile the job.
7. If the job compiles correctly, you can choose one of the following:
   - Run the job from within InfoSphere DataStage and QualityStage Designer.
   - Run or schedule the job by using the InfoSphere DataStage and QualityStage Director.
8. If the job does not compile correctly, correct the errors and recompile.

Properties

The MS SQL Server Load stage supports stage and link properties that are visible from the InfoSphere DataStage Designer. You need to supply values for these properties in the stage grid-style editor.

The tables in the next two sections include the following column heads:

- **Prompt** is the text that the job designer sees in the stage editor user interface.
- **Type is the data type of the property.**
- **Default** is the value used if the job designer does not supply a value.
- **Description** gives details about the properties.

Stage Properties

Use the properties on the **Properties** tab of the "Stage" page to specify how the stage connects to the SQL Server data source.

../com.ibm.swg.im.iis.ds.sqlsrvload.help.doc/topics/
r_cmsftref_Stage_Properties.dita

Link Properties

Use the properties on the **Properties** tab of the "Input" page to specify how the stage loads data to the SQL Server data source.

MS SQL Server Load stage Input page - Properties tab

Chapter 5. MS OLEDB Stages

MS OLEDB lets you retrieve information from any type of information repository, such as a relational source, an ISAM file, a personal database, or a spreadsheet. This stage uses OLE DB SDK that is part of Microsoft Data Access SDK (MDAC). This topic describes the following for the MS OLEDB stage:

- "Functionality of MS OLEDB Stages"
- "Defining the MS OLEDB Stage"
- "Connecting to an OLE DB Data Source"
- "Creating and Populating Cubes"
- "Defining Character Set Mapping"
- "Defining an Input Link"
- "Defining an Output Link"
- "OLE DB Server Data Type Support"
- "Stored Procedure Support"
- "CREATE CUBE Statement"
- "INSERT INTO Statement"

Functionality of MS OLEDB Stages

The MS OLEDB stage can do the following:

- Create local multidimensional databases called local cube files, and load these cubes with the data from the underlying database.
- Support stream input, stream output, and reference output links.
- Specify the number of rows to retrieve from the data source.
- Specify the number of rows to update at one time.
- Specify the isolation level used for transactions.
- Specify the number of rows to write before committing them.
- Control the type of tracing information to add to the log.
- Specify which generated or user-defined SQL statements to execute for reading or writing data.
- Specify additional clauses to append to the generated SQL statements.
- Specify SQL statements to run before and after processing job data rows.
- Specify which mode to use to retrieve or insert data.
- Browse source or target data by using the GUI.
- Import table and column definitions from the target OLE DB data source and store them in the InfoSphere DataStage Repository.
- Use NLS (National Language Support).

Terminology

The following table lists the Microsoft OLAP Services Bulk Load terms used in this document:

Table 3. Description of Terms

Term	Description
Dimension	A collection of measures and a set of dimensions. Each measure has an aggregate function, and each dimension contains one or more level. Optionally, dimensions can include multiple hierarchies. Each hierarchy contains levels.
Objects	Distinct items in the database such as dimensions, variables, formulas, and so forth. Used to create OLAP applications to access, manipulate, and display data stored in a multidimensional database management system.
OLAP	Online Analytical Processing. This processing uses multidimensional data.
UNC	Universal Naming Convention. A PC format that specifies the location of resources on a network.
Valueset	An object that contains a list of dimension values for a particular dimension.
Variable	An object that stores the actual data. All of the data in a variable represents the same unit of measurement with the same data type. Typically a variable is a multidimensional array, from which you can uniquely select any data value within it by specifying one member from each of its dimensions.

Defining the MS OLEDB Stage

When you use the custom GUI to edit an MS OLEDB stage, the MS OLEDB Stage dialog box appears. This dialog box has the **Stage**, **Input**, and **Output** pages (depending on whether there are inputs to and outputs from the stage):

- **Stage.** This page displays the name of the stage you are editing. The **General** tab defines the MS OLEDB tracing, provider, connection, and login information. The properties on this page define the connection to the OLE DB data source. Additionally, you can specify information to create and populate a cube. For connection details, see "Connecting to an OLE DB Data Source".

  The **NLS** tab defines a character set map to use with the stage. This tab appears only if you have installed NLS for InfoSphere DataStage. For details, see "Defining Character Set Mapping".

  Note: You cannot change the name of the stage from this dialog box.
- **Input.** This page is displayed only if you have an input link to this stage. It specifies the data source to use and the associated column definitions for each

data input link. It also specifies how data is written, the transaction isolation level, array size, and tracing information used to write data to a data source.
- **Output.** This page is displayed only if you have an output or reference link to this stage. It specifies the data sources to use and the associated column definitions for each data output link. It also specifies how to read the data, the transaction isolation level, array size, and tracing information used to read the data.

Retrieving Rows from a Data Source

About this task

The main phases in defining an MS OLEDB stage from the MSOLEDB Stage dialog box are as follows:

Procedure

1. Connect to an OLE DB data source.
2. Optional. Generate a new connection string.
3. Optional. Use the cube wizard to define a cube.
4. Optional. Define a character set map.
5. Define the data on the input links.

Writing to a Data Source

About this task

The main phases in defining an MS OLEDB stage from the MSOLEDB Stage dialog box are as follows:

Procedure

1. Connect to an OLE DB data source. (See “Connecting to an OLE DB Data Source”).
2. Optional. Generate a new connection string. (See Creating and Populating Cubes).
3. Optional. Use the cube wizard to define a cube. (See Creating and Populating Cubes).
4. Optional. Define a character set map. (See Defining Character Set Mapping).
5. Define the data on the output links. (See Defining an Output Link).

Connecting to an OLE DB Data Source

The OLE DB connection parameters are set on the **General** tab of the **Stage** page.

Creating and Populating Cubes

The MS OLEDB stage loads data to and from a database table. It also provides functionality to create local multidimensional databases called local cube files, and loads these cubes with the data from the underlying database.

It uses the Microsoft Pivot Table Service Provider (MSOLAP) to create and populate the cubes. MSOLAP does not distinguish between the creation of a cube and inserting into or populating the cube. This is because an INSERT INTO statement must be used with the CREATE CUBE statement to provide structure for a multidimensional database.

Note: Currently MSOLAP does not support cube creation on the OLAP server. Therefore, a cube is a local cube file.

You can view a cube file as a multidimensional data repository. Each cube file ends in a .cub extension and can contain multiple cubes. Each cube in the file can contain multiple catalogs.

For example, suppose the Sales.Cub is the multidimensional data repository. The CREATE CUBE statement creates the Sales_USA, Sales_India, Sales_UK, and Sales_Rest cubes. The INSERT INTO statement creates the North and South catalogs inside Sales_USA, the All catalog inside Sales_India, and the All catalog inside Sales_Rest. Every INSERT INTO statement must be used with a corresponding CREATE CUBE statement. If the specified cube already exists, the statement is ignored. Otherwise, it is created and a new catalog specified by the INSERT INTO statement is created in that cube.

Editing data link properties to create the cube

About this task

To edit the data link properties to create the cube:

Procedure

1. Click **Cube Wizard** from the **General** tab of the **Stage** page to connect to the provider. You can also enter the connection string information in the **Create and populate cube** field on the same tab. In this case, the cube is created at the end of the table loading process.

   Note: Release 2.0 of the MS OLEDB stage must have only one input link and no output links to create the cube successfully. You should not use multiple input links or input and output links in the same MS OLEDB stage.
2. Select **Microsoft OLE DB Provider for OLAP Services**. The Data Link Properties dialog box appears.
3. Click the **All** tab, then specify the parameters to create and populate a local cube file.
4. Select the parameter, then click **Edit Value...** to enter the appropriate information for the data link initialization properties described in the following table. See "Properties for data link initialization" for specific information about the properties.

Properties for data link initialization

The following table describes the most important properties. Supply information for the Data Source, CREATECUBE, INSERTINTO, SOURCE_DSN, User ID, and Password properties:

Table 4. Properties for Data Link Initialization

Property	Description
ARTIFICIALDATA	The artificial aggregate values calculated instead of calculating real values. These values are calculated by using a simple algorithm when the first character of this string is Y, T, or a numeric digit other than 0.
Data Source	The name of the local cube file you want to create. The default extension for a local cube file is .cub, but any extension can be used.
Initial Catalog	The name of the default initial database catalog. Use this property unless you are creating a local cube. The value is used when a session is established, but you cannot change the value during the session.
DBPROP_INIT_ASYNCH	Optional. Specifies asynchronous initialization. Do not use this property in the connection string. However, you can use it programmatically. For more information, see the OLE DB documentation.
CREATECUBE	The CREATE CUBE statement to create a local cube file. You must use this property if you also use the INSERTINTO and SOURCE_DSN properties. These three properties are always used together. This value is used when a session is established, but you cannot change it during the session.
INSERTINTO	The INSERT INTO statement to populate a local cube file that was created by using the CREATE CUBE statement.
SOURCE_DSN	The ODBC connection string, OLE DB connection string, or DSN for the source relational database, used only when creating a local cube file.
User ID	The ODBC UID for the source database, used only when creating a local cube file.
Password	The ODBC PWD for the source database, used only when creating a local cube file.

Table 4. Properties for Data Link Initialization (continued)

Property	**Description**
Auto Synch Period	Specifies how often queries are made to the source database. The default value on the server is 10,000 milliseconds (10 seconds). By setting this property to a null value or 0, automatic synchronization is turned off, and synchronization does not occur at a constant interval. The frequency of synchronization depends on client activity. Some client queries are resolved solely from the client cache. Therefore, a high value can cause more frequent query results that do not reflect recent updates in the data source. A low value can reduce the likelihood of these events. However, a low value can impede performance. The lowest valid, nonzero value is 250 milliseconds. A value of 250 milliseconds is used for any value from 1 to 249. This value is used when a session is established, but you cannot change it during the session.
Cache Policy	Specifies information about memory.
Client Cache Size	The specified number of kilobytes (KB) of memory in the client cache. If set to 0, the client cache can use unlimited memory. If set to a value from 1 to 99, the client cache can use the specified percentage of total available virtual memory (physical and page file). If this property is set to 100 or more, the client cache can use up to the specified KB of memory. This value is used when a session is established, but you cannot change it during the session.
CompareCaseSensitive StringFlags	The flags used in case-sensitive string comparisons to control string comparisons and sort order.
CompareCaseNotSensitive StringFlags	The flags used in string comparisons that are not case-sensitive to control string comparisons and sort order. This property is used more frequently in NLS versions. The default is the value of the CompareCaseSensitive StringFlags registry on the client if this registry exists.

Table 4. Properties for Data Link Initialization (continued)

Property	**Description**
Default Isolation Mode	If the first character of this string is Y, T, or a numeric digit other than 0, the isolation level is isolated. Otherwise, the isolation level is determined by the cursor type requested by the rowset properties. For more information about isolation levels, see the OLE DB documentation.
Execution Location	The location of the query execution. Use one of the following values: 0 - The default value. This is equivalent to a value of 1. 1 - The automatic selection of query execution location, either client or server. 2 - The query executes on the client. 3 - The query executes on the server. Exceptions include queries that contain session-scoped calculated members, user-defined sets, or user-defined functions.
Extended Properties	
Integrated Security	
Large Level Threshold	Specifies whether dimension levels are sent from the server to the client incrementally or in their entirety. Dimension levels that contain a number of members greater than or equal to the value of this property are sent incrementally. A level that contains fewer members than the value of this property is sent to the client in its entirety. This helps manage client memory usage. The default value is set on the server in the **Large level defined as** box in the **Data Link** Properties dialog box. The minimum value is 10. Settings less than 10 are ignored, and the minimum value is used. In this case, no error is returned.
Locale Identifier	The ID for the locale (LCID), which the client can modify by setting the DBPROP_INIT_LCID property. Pivot Table Service can have only one LCID per Windows process. The LCID must be installed in Control Panel in Windows, or the attempt to set the LCID fails. By default, the DBPROP_INIT_LCID is reported as null.
Location	

Table 4. Properties for Data Link Initialization (continued)

Property	**Description**
Mode	
Persist Security Info	
Read only Session	
SOURCE_DSN_SUFFIX	The suffix used only when creating or connecting to a local cube. This value is not stored in the local cube file. This property is useful for separating data persisted in the local cube file from data used only for the session. (Session data includes user account and password.)
Source OLE DB Provider	A predefined string containing other initialization properties. This standard OLE DB property does not specify usage. Usage is specific to those providers that use it.
USEEXISTINGFILE	Specifies whether to connect to the existing local cube. If the first character of this value is Y, T, or a numeric digit other than 0, and the cube file specified in the Data Source property already exists, the CREATECUBE and INSERTINTO properties are ignored. A connection to the existing local cube is established. If this value is not used, or the first character of this value is not Y, T, or a numeric digit other than 0, and the cube file specified in the Data Source property already exists, the statements in the CREATECUBE and INSERTINTO properties overwrite the existing cube file.
Writeback Timeout	The number of seconds before an update occurs. The attempt to communicate updates is triggered by a commit, which begins a counting of seconds. The counting continues until the commit is successful or the specified number of seconds is reached. If this value is reached, the commit fails and the update does not occur. You can then attempt another commit or a rollback.

For details about the CREATE CUBE and INSERT INTO statements, see CREATE CUBE Statement **and** page INSERT INTO Statement**.** For more information about Pivot Table Service, see the documentation for Microsoft SQL Server for OLAP Services and Microsoft Data Access Component SDK.

Defining Character Set Mapping

You can define a character set map for a stage. Do this from the **NLS** tab that appears on the Stage page. The **NLS** page appears only if you have installed NLS.

Specify information by using the following button and fields:

- **Map name to use with stage.** The default character set map is defined for the project or the job. You can change the map by selecting a map name from the list.
- **Use Job Parameter....** Specifies parameter values for the job. Use the format *#Param#*, where *Param* is the name of the job parameter. The string *#Param#* is replaced by the job parameter when the job is run.
- **Show all maps.** Lists all the maps that are shipped with InfoSphere DataStage.
- **Loaded maps only.** Lists only the maps that are currently loaded.

Defining an Input Link

About this task

When you write data to a data source, the MS OLEDB stage has an input link. Define the properties of this link and the column definitions of the data on the Input page in the **MSOLEDB Stage** dialog box.

About the Input Page

The Input page has an **Input name** field, the **General, Columns,** and **SQL** tabs, and the **Columns... and View Data... buttons. (The View Data... button is disabled in this release.)**

- **Input name.** The name of the input link. Choose the link you want to edit from the **Input name** drop-down list box. This list displays all the input links to the MS OLEDB stage.
- Click the **Columns...** button to display a brief list of the columns designated on the input link. As you enter detailed metadata in the **Columns** tab, you can leave this list displayed.
- Click the **View Data...** button to start the Data Browser. This lets you look at the data associated with the input link. **(The View Data... button is disabled in this release.)**

General Tab of the Input Page

Use the **General** tab of the “Input” page to specify details about how data is written to an MS OLEDB data source.

MS OLEDB stage Input page - General Tab

Columns Tab of the Input Page

This tab contains the column definitions for the data written to the data source. The **Columns** tab behaves the same way as the **Columns** tab in the ODBC stage.

SQL Tab

Use the **SQL** tab to view a generated SQL query or to specify your own query.

This tab displays the stage-generated or user-defined SQL statements used to read data from OLE DB. It contains the **Generated**, **User-defined**, **Before**, and **After** tabs, which are the same as those for the Input page under the **SQL** tab.

- **Generated.** This contains the SQL statements constructed by IBM InfoSphere DataStage as a result of the **Output action** from the **General** tab of the Output page. You cannot edit these statements, but you can use **Copy** to copy them to the Clipboard for use elsewhere.

- **User-defined.** This tab is displayed by default. It contains the SQL statements executed to read data from the data source. The GUI displays the stage-generated SQL statement on this tab as a starting point. However, you can enter any valid, appropriate SQL statement. The box size changes proportionately when you resize the main window to display long SQL statements.
- **Before.** This tab contains the SQL statements executed before the stage processes any job data rows. Use a semicolon (;) to separate multiple BeforeSQL statements. The SQL statement is entered in a resizable edit box. Execution occurs immediately after a successful data source connection. The **Before** and **After** tabs look alike.
- **After.** This tab contains the SQL statements executed after the stage processes any job data rows. Use a semicolon (;) to separate multiple AfterSQL statements. The SQL statement is entered in a resizable edit box. Execution occurs immediately after the last row is processed, before the data source connection is terminated. The **Before** and **After** tabs look alike.

Writing Data to OLE DB

The following sections describe the differences when you use generated or user-defined SQL INSERT, DELETE, or UPDATE statements to write data from InfoSphere DataStage to a data source. You can also execute BeforeSQL or AfterSQL statements before or after the stage processes job data rows.

Using Generated SQL Statements

About this task

By default, the IBM InfoSphere DataStage writes data to a data source by using an SQL INSERT, DELETE, or UPDATE statement that it constructs. The generated SQL statement is automatically constructed by using the InfoSphere DataStage table and column definitions that you specify in the input properties for this stage. The **Generated** tab on the **SQL** tab displays the SQL statement used to write the data.

To use a generated statement:

Procedure

1. Enter a table name in the **Table name** field on the **General** tab of the Input page.
2. Specify how you want the data to be written by choosing an option from the **Input action** list box. See ""General Tab of the Input Page" for a description of the input actions.
3. Optional. Enter a description of the input link in the **Description** field.
4. Click the **Columns** tab on the Input page.
5. Edit the Columns grid to specify the column definitions for the columns you want to write. The SQL statement is automatically constructed by using your chosen input action and the columns you have specified. You can now optionally view this SQL statement.
6. Click the **SQL** tab on the Input page, then the **Generated** tab to view this SQL statement. You cannot edit the statement here, but you can always access this tab to select and copy parts of the generated statement to paste into the user-defined SQL statement.

7. Click **OK** to close this dialog box. Changes are saved when you save your job design.

Using User-Defined SQL Statements

About this task

Instead of writing data by using an SQL statement constructed by the IBM InfoSphere DataStage, you can enter your own SQL INSERT, DELETE, or UPDATE statement or call stored procedures for each MS OLEDB input link. Ensure that the SQL statement contains the table name, the type of input action you want to perform, and the columns you want to write.

To use your own SQL statement:

Procedure

1. Set **SQL generation** to **No** on the **General** tab of the Input page.
2. Specify how you want the data to be written by choosing an option from the **Input action** drop-down list box. See ""General Tab of the Input Page" for a description of the input actions.
3. Click the **SQL** tab, then the **User-defined** tab. By default you see the stage-generated SQL statement. You can edit this statement or enter your own SQL statement to write data to the target data sources. This statement must contain the table name, the type of input action you want to perform, and the columns you want to write.

   When writing data, the INSERT statements must contain a VALUES clause with a parameter marker (?) for each stage input column. UPDATE statements must contain a SET clause with parameter markers for each stage input column. UPDATE and DELETE statements must contain a WHERE clause with parameter markers for the primary key columns.

   The type of SQL statement used depends on the number of parameters and key columns required. The parameter markers must be in the same order as the associated columns listed in the stage properties.

   Table 5. Guidelines for User-Defined SQL Statements

If a statement has...	Use a statement like...
As many parameters as there are key columns	DELETE from TABLE WHERE Key1=? and Key2=?
As many parameters as there are columns	INSERT into TABLE (Col1, Col2) VALUES (?, ?)
As many parameters as there are columns and key columns	UPDATE TABLE SET Col1=?, Col2=?, Key1=?, Key2=? WHERE Key1=? and Key2=?

   The size of this box changes proportionately when the main window is resized in order to conveniently display very long or complex SQL statements.

   Unless you specify a user-defined SQL statement, the stage automatically generates an SQL statement.

   If you specify multiple SQL statements, each is executed as a separate transaction. End SQL statements by using a semicolon (;) as the end-of-batch signal. You cannot combine multiple INSERT, UPDATE, and DELETE statements in one batch. You must execute each in a separate command batch.
4. Click **OK** to close the this dialog box. Changes are saved when you save your job design.

Using BeforeSQL Statements

About this task

You can execute SQL statements before the stage processes any job data rows. To specify SQL statements before processing any data:

Procedure

1. Enter the SQL statements you want to be executed before data is processed in the text entry area on the **Before** tab of the **SQL** tab.

   Execution occurs immediately after a successful data source connection. If you specify multiple SQL statements, they are executed as one or more Transact-SQL command batches by using a semicolon (;) as the end-of-batch signal.
2. Select the **Treat errors as non-fatal** check box to log BeforeSQL execution errors as warnings. Processing continues with the next command batch, if any. Each successful execution is committed as a separate transaction.

   If this check box is cleared, BeforeSQL execution errors are considered fatal to the job and result in a transaction rollback. The transaction is committed only if all BeforeSQL statements successfully execute.

Using AfterSQL Statements

About this task

You can execute SQL statements after the stage processes all job data rows. To specify SQL statements after processing data:

Procedure

1. Enter the SQL statements you want to be executed after the data is processed in the text entry area on the **After** tab of the **SQL** tab.

   Execution occurs immediately before the data source connection is terminated. If you specify multiple SQL statements, they are executed as one or more Transact-SQL command batches by using a semicolon (;) as the end-of-batch signal.
2. Select the **Treat errors as non-fatal** check box to log AfterSQL execution errors as warnings. Processing continues with the next command batch, if any. Each successful execution is committed as a separate transaction.

   If this check box is cleared, AfterSQL execution errors are considered fatal to the job and result in a transaction rollback. The transaction is committed only if all AfterSQL statements successfully execute.

Defining an Output Link

About this task

When you read data from a data source, the MS OLEDB stage has an output link. Define the properties of this link and the column definitions of the data on the Output page in the MSOLEDB Stage dialog box.

About the Output Page

The Output page has an **Output name** field, the **General**, **Columns**, **Selection**, and **SQL** tabs, and the **Columns...** and **View Data...** buttons. **(The View Data... button is disabled in this release.)** The tabs displayed depend on how you specify the SQL statement to output the data.

- **Output name.** The name of the output link. Choose the link you want to edit from the **Output name** drop-down list box. This list displays all the output links from the MS OLEDB stage.
- Click the **Columns...** button to display a brief list of the columns designated on the output link. As you enter detailed metadata in the **Columns** tab, you can leave this list displayed.
- Click the **View Data...** button to start the Data Browser. This lets you look at the data associated with the output link. **(The View Data... button is disabled in this release.)**

General Tab of the Output Page

Use the **General** tab of the “Output” page to specify details about how data is read from an MS OLEDB data source.

../com.ibm.swg.im.iis.ds.msoledb.help.doc/topics/
r_cmsftref_General_Tab_of_the_Output_Page.dita

Columns Tab of the Output Page

This tab contains the column definitions for the data being output on the chosen link. The column definitions are used in the order they appear in the Columns grid. The **Columns** tab behaves the same way as the **Columns** tab in the ODBC stage.

The column definitions for output and reference links contain a key field. Key fields are used to join primary and reference inputs to a Transformer stage. MS OLEDB key reads the data by using a WHERE clause in the SQL SELECT statement.

Selection Tab

This tab is used primarily with generated SQL queries. It contains optional SQL SELECT clauses, such as WHERE, HAVING, or ORDER BY for the conditional extraction of data.

If you want to use the additional SQL SELECT clauses, you must enter them on the **Selection** tab of the Output page. These clauses are appended to the SQL statement that is generated by the stage. If this link is a reference link, only the WHERE clause is enabled.

The **Selection** tab is divided into two areas (panes). You can resize an area by dragging the split bar.

- **WHERE clause.** This text box allows you to insert an SQL WHERE clause to specify criteria that the data must meet before being selected.
- **Other clauses.** This text box allows you to insert a HAVING or an ORDER BY clause.

SQL Tab

Use the **SQL** tab to view a generated SQL query or to specify your own query.

This tab displays the stage-generated or user-defined SQL statements used to read data from OLE DB. It contains the **Generated**, **User-defined**, **Before**, and **After** tabs, which are the same as those for the Input page under the **SQL** tab.

- **Generated.** This contains the SQL statements constructed by IBM InfoSphere DataStage as a result of the **Output action** from the **General** tab of the Output page. You cannot edit these statements, but you can use **Copy** to copy them to the Clipboard for use elsewhere.
- **User-defined.** This tab is displayed by default. It contains the SQL statements executed to read data from the data source. The GUI displays the stage-generated SQL statement on this tab as a starting point. However, you can enter any valid, appropriate SQL statement. The box size changes proportionately when you resize the main window to display long SQL statements.
- **Before.** This tab contains the SQL statements executed before the stage processes any job data rows. Use a semicolon (;) to separate multiple BeforeSQL statements. The SQL statement is entered in a resizable edit box. Execution occurs immediately after a successful data source connection. The **Before** and **After** tabs look alike.
- **After.** This tab contains the SQL statements executed after the stage processes any job data rows. Use a semicolon (;) to separate multiple AfterSQL statements. The SQL statement is entered in a resizable edit box. Execution occurs immediately after the last row is processed, before the data source connection is terminated. The **Before** and **After** tabs look alike.

Reading Data from OLE DB

The following sections describe the differences when you use generated queries or user-defined queries to read data from a data source into the IBM InfoSphere DataStage.

The column definitions for reference links must contain a key field. You use key fields to join primary and reference inputs to a Transformer stage. MS OLEDB key reads the data by using a WHERE clause in the SQL SELECT statement.

Using Generated Queries

By default, the IBM InfoSphere DataStage extracts data from a data source by using an SQL SELECT statement that it constructs. The SQL statement is automatically constructed by using the table and column definitions that you entered on the **Output** page.

When you select **Yes** in **SQL generation**, data is extracted from a data source by using an SQL SELECT statement constructed by the InfoSphere DataStage. SQL SELECT statements have the following syntax:

```
SELECT clause FROM clause
[WHERE clause]
[GROUP BY clause]
[HAVING clause]
[ORDER BY clause];
```

When you specify the data sources to use and the columns to be output from the MS OLEDB stage, the SQL SELECT statement is automatically constructed and can be viewed by clicking the **SQL** tab on the Output page.

For example, if you extract the Name, Address, and Phone columns from a table called Table1, the SQL statement displayed of the **SQL** tab is:

```
SELECT Name, Address, Phone FROM Table1;
```

The SELECT and FROM clauses are the minimum required and are automatically generated by the InfoSphere DataStage. If you want to use the following additional SQL SELECT clauses, you must enter them on the **Selection** tab of the **Output** page:

- **SELECT clause.** Specifies the columns to select from the database.
- **FROM clause.** Specifies the tables containing the selected columns.
- **WHERE clause.** Specifies the criteria that rows must meet to be selected.
- **GROUP BY clause.** Groups rows to summarize results.
- **HAVING clause.** Specifies the criteria that grouped rows must meet to be selected.
- **ORDER BY clause.** Sorts selected rows.

Using User-Defined Queries

About this task

Instead of using the SQL statement constructed by IBM InfoSphere DataStage, you can enter your own SQL statement for each MS OLEDB output link. To enter an SQL statement:

Procedure

1. Set **SQL generation** to **No** on the **General** tab of the **Output** page. The **User-defined** tab on the **SQL** tab is enabled. It looks like the **User-defined** tab for the input link.
2. You can edit the statements or drag the selected columns into your user-defined SQL statement. You must ensure that the table definitions for the output link are correct and represent the columns that are expected. The result set generated from this statement returns at least one row. If more than one result set is produced, only the first set is used.
3. Click **OK** to close this dialog box. Changes are saved when you save your job design.

Results

Restrictions for using the user-defined SQL queries are as follows:

- If you use multiple SQL SELECT statements to read the data, only the last statement returns the result.
- Nested SQL statements are not supported.
- If more than one result set is produced, only the first set is used.
- Rowsets resulting from the execution of BeforeSQL and AfterSQL statements are not processed.
- For reference output links, the SELECT statement should have parameter markers (?) specified in the column definitions and the "Where clause" property. The parameter markers must be in the same order as the associated key columns listed in the stage properties.

OLE DB Server Data Type Support

The following table documents the support for OLE DB character, numeric, and date data types. It summarizes the data types for IBM InfoSphere DataStage SQL type definitions, and their OLE DB SQL server data types:

Table 6. Support for OLE DB Data Types in the OLEDB Stage

InfoSphere DataStage SQL Data Type	OLE DB Data Type
SQL_CHAR	DBTYPE_WSTR (Unicode string)
SQL_VARCHAR	DBTYPE_WSTR
SQL_STRING	DBTYPE_WSTR
SQL_LONGVARCHAR	DBTYPE_WSTR
SQL_BINARY	DBTYPE_BYTES (array of bytes)
SQL_VARBINARY	DBTYPE_BYTES
SQL_LONGVARBINARY	DBTYPE_BYTES
SQL_NUMERIC	DBTYPE_R8 (double)
SQL_DECIMAL	DBTYPE_R8
SQL_FLOAT	DBTYPE_R8
SQL_REAL	DBTYPE_R8
SQL_DOUBLE	DBTYPE_R8
SQL_INTEGER	DBTYPE_I4 (long)
SQL_SMALLINT	DBTYPE_I4
SQL_BIGINT	DBTYPE_I4
SQL_TINYINT	DBTYPE_I4
SQL_BIT	DBTYPE_I4
SQL_DATE	DBTYPE_DBDATE
SQL_TIME	DBTYPE_DBTIME
SQL_TIMESTAMP	DBTYPE_DBTIMESTAMP

Stored Procedure Support

You can call stored procedures from the server MS OLEDB. The following rules apply:

- Specify input parameters as literal values. Passing row values as parameter values is not supported.
- Output parameters are not supported.
- You can call stored procedures as part of the BeforeSQL and AfterSQL statements. Any result sets generated by the procedure are discarded.
- You can also call stored procedures as part of the user-defined SQL statement for all links. The stored procedure must generate a row result set that matches the stage output column definitions. Only one row result set is processed, and any

additional result sets are discarded. The input link parameter count should correspond to the input action. For examples, see ""Using User-Defined SQL Statements".

CREATE CUBE Statement

The CREATE CUBE statement defines the structure of a new local cube. This statement shares much of the syntax for SQL-92 and the CREATE TABLE statement, but has added syntax for cubes. Use the INSERT INTO statement to populate the cube. For further details on the INSERT INTO statement, see ""INSERT INTO Statement" .

The following sections document the syntax in Backus Naur Form (BNF) notation. For more information about cubes, see the SQL Server documentation.

CREATE CUBE Syntax in BNF Notation

The syntax for the CREATE CUBE statement uses BNF notation. BNF is a notation format that uses a series of symbols and production rules that successively break down statements into their components.

```
<create-cube-statement > ::= CREATE CUBE <cube name> <open paren>
DIMENSION <dimension name> [TYPE TIME],
<hierarchy def> [<hierarchy def>...]
[{, DIMENSION <dimension name> [TYPE TIME],
<hierarchy def> [<hierarchy def>...]}...] ,
MEASURE <measure name> <measure function def> [<measure format def>]
[<measure type def>]
[{, MEASURE <measure name> <measure function def> [<measure format def>]
[<measure type def>] }...]
[,COMMAND <expression>]
[,COMMAND <expression>...]
<close paren>
.<dimension name> ::= <legal name>
<hierarchy def> ::= [HIERARCHY <hierarchy name>,] <level def> [,<level def>...]
<level def> ::= LEVEL <level name> [TYPE <level type>] [<level format def>]
[<level options def>]
<level type> ::= ALL | YEAR | QUARTER | MONTH | WEEK | DAY | DAYOFWEEK
|DATE | HOUR | MINUTE | SECOND<level format def> ::= FORMAT_NAME <expression>
[FORMAT_KEY <expression>]
<level options def> ::= OPTIONS (<option_list>)
<option_list> :: = <option> [,<option_list>]
<option> ::= UNIQUE | SORTBYNAME | SORTBYKEY
<measure function def> ::= FUNCTION <function name>
<measure format def> ::= FORMAT <expression>
<function name> ::= SUM | MIN | MAX | COUNT
<measure type def> ::= TYPE <supported OLEDB numeric types>
<supported OLEDB numeric types> :: = DBTYPE_I1 | DBTYPE_I2 | DBTYPE_I4
| DBTYPE_I8 | DBTYPE_UI1 | DBTYPE_UI2 | DBTYPE_UI4 | DBTYPE_UI8 | DBTYPE_R4 |
DBTYPE_R8 | DBTYPE_CY | DBTYPE_DECIMAL | DBTYPE_NUMERIC | DBTYPE_DATE
```

DIMENSION Clause

The name given to a TYPE ALL level applies the specified name to the ALL member rather than the ALL level. The ALL level always has the name All. For example, the clause LEVEL [All Customers] TYPE ALL creates a level named (All) containing a single member named [All Customers]. There is no [All Customers] level.

COMMAND Clause

If the <expression> value has spaces, use brackets to surround the whole expression. Do not use quotation marks because the body of the command can include quotation marks. (OLAP Services supports nested brackets but not nested quotation marks.)

Example

```
CREATE CUBE Sales
(
DIMENSION Time TYPE TIME,
HIERARCHY [Fiscal],
LEVEL [Fiscal Year] TYPE YEAR,
LEVEL [Fiscal Qtr] TYPE QUARTER,
LEVEL [Fiscal Month] TYPE MONTH OPTIONS (SORTBYKEY, UNIQUE),
HIERARCHY [Calendar],
LEVEL [Calendar Year] TYPE YEAR,
LEVEL [Calendar Month] TYPE MONTH,
DIMENSION Products,
LEVEL [All Products] TYPE ALL,
LEVEL Category,
LEVEL [Sub Category],
LEVEL [Product Name],
DIMENSION Geography,
LEVEL [Whole World] TYPE ALL,
LEVEL Region,
LEVEL Country,
LEVEL City,
MEASURE [Sales]
FUNCTION SUM
FORMAT 'Currency',
MEASURE [Units Sold]
FUNCTION SUM
TYPE DBTYPE_UI4
)
```

INSERT INTO Statement

The INSERT INTO statement is similar to the SQL-92 syntax for creating and populating tables. It populates a local cube with dimension members. If the local cube is in multidimensioned (MOLAP) storage mode, the INSERT INTO statement also populates the local cube with data. The INSERT INTO statement is used after a CREATE CUBE statement to create a local cube.

Backus Naur Form (BNF) Notation

The following syntax in BNF notation documents the INSERT INTO statement.

```
<insert-into-statement> ::= INSERT INTO <target-clause>
[<options-clause>] [<bind-clause>] <source-clause>
<target-clause> ::= <cube-name> <open-paren> <target-element-list>
<close-paren>
<target-element-list> ::= <target-element>[, <target-element-list>]
<target-element> ::= [<dim-name>.[<hierarchy-name>.]]<level-name>
| <time-dim-name>| [Measures.]<measure-name>
| SKIPONECOLUMN <level-name> ::= <simple-level-name>
| <simple-level-time>.
NAME|<simple-level-time>.KEY <time-dim-name>
 ::= <dim-name-type-time>
| <dim-name-type-time>.NAME | <dim-name-type-time>.KEY<options-clause> ::=
OPTIONS <options-list>
<options-list> ::= <option>[, <options-list>]
<option> ::= <defer-options> | < analysis-options>
```

```
<defer-options> ::= DEFER_DATA | ATTEMPT_DEFER<analysis-options> ::= PASSTHROUGH
| ATTEMPT_ANALYSIS <bind-clause>
::= BIND (<bind-list>)<bind-list>
::= <simple-column-name>[,<simple-column-name>]
<simple-column-name> ::= <identifier>
<source-clause> ::= SELECT <columns-list>
FROM <tables-list>
[ WHERE <where-clause> ]
| DIRECTLYFROMCACHEDROWSET <hex-number>
<columns-list> ::= <column-expression> [, < columns-list> ]
<column-expression> ::= <column-expression-name>
<column-expression-name> ::= <column-name> [AS <alias-name>]
| <alias name> <column-name>
<column-name> ::= <table-name>.<column-name>
| <column-function> | <ODBC scalar function> | <braced-expression>
<column function> ::= <identifier>(...)
<ODBC scalar function> ::= {FN<column-function>}
<braced-expression> ::= (...)
<tables list> ::= <table-expression> [, <table-list>]
<table-expression> ::= <table-name> [ [AS] <table-alias>]
<table-alias> ::= <identifier>
<table-name> ::= <identifier>
<where clause> ::= <where-condition> [AND <where-clause>]
<where condition> ::= <join-constraint> | <application constraint>
<join-constraint> ::= <column-name> = <column-name>
| <open-paren><column-name> = <column-name><close-paren>
<application-constraint> ::= (...)
| NOT (...)| (...) OR (...)
<identifier> ::= <letter>{<letter>|<digit>|<underline>|<dollar>|<sharp>}...
```

Names of Elements.

These are level and measure names, sometimes qualified with dimension name or the Measures keyword to avoid ambiguity. The Measures keyword is case-sensitive in binary comparisons. If you use binary comparison or are unsure of your comparison method, use Measures as shown with initial uppercase.

Each level and each measure in a cube is derived from a column in the SELECT clause.

The Columns Specified in the Associated SELECT Clause.

These are bound to the elements of the INSERT INTO statement in the order specified and in a one-to-one relationship.

Each level can be derived from two columns, with one used as a name column and the other used as a key column. Both columns must be in the same table. If there are two columns associated with a level, use the .NAME or .KEY suffix in the INSERT INTO statement after the level name.

If a column specified in the SELECT clause does not have a related element in the INSERT INTO statement, use the SKIPONECOLUMN keyword as a placeholder for the unused column. You can use SKIPONECOLUMN more than once.

Dimension of TYPE TIME.

Specify by the name of the dimension. The dimension name correlates the entire dimension with a single column in the source table that contains data with a date or time data type. The TYPE <level type> levels, identified for the time dimension in the CREATE CUBE statement cause the time information to be extracted from the source column specified in the SELECT clause. See "Example 4".

The WHERE Clause.

This clause can have both application and join constraints. The parser parses only join constraints, using the join constraint to find a path from all tables to the fact table and the dimension tables. The application constraint is used only to specify constraints on a fact table and is passed through unmodified.

Expressions Between Parentheses.

These expressions are treated as application constraints. For example, if the expression Sales.Product_ID = Products.Product_ID AND Sales.Customer_ID = Customers. Customer_ID is enclosed in parentheses, it is treated as an application constraint and is not used as a join constraint. Use parentheses only around application constraints in your application, for example, (Product.Price < 100 AND Product.Category = 1).

BIND Clause.

This is used to bind level and measure names specified with column names used to create rowsets.

AS <alias-name> Syntax.

This is not supported for local cubes in relational (ROLAP) storage mode.

Example 1

```
INSERT INTO MyCube (Year, Month.Name, Month.Key, [Product Group],
[Product Name], Country, Sales, Cost)
OPTIONS DEFER_DATA
SELECT MyTable.Year, MyTable.Month, MONTH(MyTable.Month), MyTable.ProdGroup,
MyTable.ProdName, MyTable.Country, MyTable.Sales, MyTable.Cost
FROM MyTable
WHERE MyTable.SalesRep = "Amir" and MyTable.CustomerGroup = "Industry"
```

Example 2

```
INSERT INTO MyCube (Year, Month, [Product Group], [Product Name], Country,
Sales, Cost)
OPTIONS PASSTHROUGH SELECT MyTable.Year, MyTable.Month, MyTable.ProdGroup,
MyTable.ProdName, MyTable.Country, MyTable.Sales, MyTable.Cost
FROM MyTable
WHERE MyTable.SalesRep = "Amir" and MyTable.CustomerGroup = "Industry"
```

Note: The PASSTHROUGH option specifies that the SELECT clause that follows it is to be passed directly to the database engine with no parsing by Pivot Table Service.

Example 3

```
INSERT INTO MyCube (Year, Month, [Product Group], [Product Name],
Country, Sales, Cost)
DIRECTLYFROMCACHEDROWSET 0x00001284
```

Note: The DIRECTLYFROMCACHEDROWSET keyword directs data to be read from the address in memory that is identified immediately after the keyword. You must specify the correct address in memory in your application. At run time, the number is assumed to be the in-process address of an IUnknown pointer to an OLE DB rowset.

Example 4

```
CREATE CUBE MyCube (
DIMENSION TimeDim TYPE TIME,
LEVEL MyYear TYPE YEAR,
LEVEL MyQtr TYPE QUARTER,
LEVEL MyMonth TYPE MONTH,
DIMENSION Products,
LEVEL [Product Group],
LEVEL [Product Name],
DIMENSION Geography,
LEVEL State,
LEVEL City,
MEASURE [Sales]
FUNCTION SUM
FORMAT 'Currency',
MEASURE [Units Sold]
FUNCTION SUM
)
INSERT INTO MyCube (TimeDim, [Product Group],
[Product Name], State, City, Sales, [Units Sold])
OPTIONS DEFER_DATA
SELECT MyTable.TransDate, MyTable.ProdGroup,
MyTable.ProdName, MyTable.State,
MyTable.City, MyTable.Sales, MyTable.UnitsSold
FROM MyTable
WHERE MyTable.SalesRep = "Jacobsen" and
MyTable.CustomerGroup = "Industry"
```

Passthrough OPTION

The PASSTHROUGH option provides advanced query processing. It causes the SELECT clause to be passed directly to the source database without modification by Pivot Table Service. If PASSTHROUGH is not specified, Pivot Table Service parses the query and formulates a set of queries equivalent to the original. These queries are optimized for the source database and index structures. This set of queries is often more efficient than the specified query.

The DEFER_DATA option causes the query to be parsed locally. It is executed only when necessary to retrieve data to satisfy a user request. DEFER_DATA specifies that a local cube is defined in the ROLAP storage mode.

The ATTEMPT_DEFER option causes Pivot Table Service to parse the query and defer data loading if successful. If the query cannot be parsed, it processes the specified query immediately as if PASSTHROUGH were specified.

The ATTEMPT_ANALYSIS option causes Pivot Table Service to parse the query and formulate an optimized set of queries (process in the MOLAP mode). If the query cannot be parsed, it processes the specified query immediately as if PASSTHROUGH were specified.

Passthrough Compatibility

The following table summarizes the options for the INSERT INTO statement for the MOLAP and ROLAP storage modes. PT indicates PASSTHROUGH functionality.

Table 7. Options for INSERT INTO Statements for MOLAP and ROLAP Storage Modes

Option	**Parse**	**Neither PASS-THROUGH Nor ATTEMPT_ ANALYSIS**	**PASS-THROUGH**	**ATTEMPT_ ANALYSIS**
Neither DEFER_DATA nor ATTEMPT_ DEFER	Succeeded Failed	MOLAP Error	MOLAP (PT) N/A	MOLAP MOLAP (PT)
DEFER_DATA	Succeeded Failed	ROLAP Error	Error N/A	ROLAP Error
ATTEMPT_ DEFER	Succeeded Failed	ROLAP MOLAP (PT)	MOLAP (PT)	ROLAP

Contacting IBM

You can contact IBM for customer support, software services, product information, and general information. You also can provide feedback to IBM about products and documentation.

The following table lists resources for customer support, software services, training, and product and solutions information.

Table 8. IBM resources

Resource	Description and location
IBM Support Portal	You can customize support information by choosing the products and the topics that interest you at www.ibm.com/support/entry/portal/Software/Information_Management/InfoSphere_Information_Server
Software services	You can find information about software, IT, and business consulting services, on the solutions site at www.ibm.com/businesssolutions/
My IBM	You can manage links to IBM Web sites and information that meet your specific technical support needs by creating an account on the My IBM site at www.ibm.com/account/
Training and certification	You can learn about technical training and education services designed for individuals, companies, and public organizations to acquire, maintain, and optimize their IT skills at http://www.ibm.com/software/sw-training/
IBM representatives	You can contact an IBM representative to learn about solutions at www.ibm.com/connect/ibm/us/en/

Providing feedback

The following table describes how to provide feedback to IBM about products and product documentation.

Table 9. Providing feedback to IBM

Type of feedback	Action
Product feedback	You can provide general product feedback through the Consumability Survey at www.ibm.com/software/data/info/consumability-survey

Table 9. Providing feedback to IBM (continued)

Type of feedback	**Action**
Documentation feedback	To comment on the information center, click the Feedback link on the top right side of any topic in the information center. You can also send comments about PDF file books, the information center, or any other documentation in the following ways: • Online reader comment form: www.ibm.com/software/data/rcf/ • E-mail: comments@us.ibm.com

Accessing product documentation

Documentation is provided in a variety of locations and formats, including in help that is opened directly from the product client interfaces, in a suite-wide information center, and in PDF file books.

The information center is installed as a common service with IBM InfoSphere Information Server. The information center contains help for most of the product interfaces, as well as complete documentation for all the product modules in the suite. You can open the information center from the installed product or from a Web browser.

Accessing the information center

You can use the following methods to open the installed information center.

- Click the **Help** link in the upper right of the client interface.

  Note: From IBM InfoSphere FastTrack and IBM InfoSphere Information Server Manager, the main Help item opens a local help system. Choose **Help > Open Info Center** to open the full suite information center.
- Press the F1 key. The F1 key typically opens the topic that describes the current context of the client interface.

  Note: The F1 key does not work in Web clients.
- Use a Web browser to access the installed information center even when you are not logged in to the product. Enter the following address in a Web browser: http://host_name:port_number/infocenter/topic/com.ibm.swg.im.iis.productization.iisinfsv.home.doc/ic-homepage.html. The host_name is the name of the services tier computer where the information center is installed, and port_number is the port number for InfoSphere Information Server. The default port number is 9080. For example, on a Microsoft® Windows® Server computer named iisdocs2, the Web address is in the following format: http://iisdocs2:9080/infocenter/topic/com.ibm.swg.im.iis.productization.iisinfsv.nav.doc/dochome/iisinfsrv_home.html.

A subset of the information center is also available on the IBM Web site and periodically refreshed at http://publib.boulder.ibm.com/infocenter/iisinfsv/v8r7/index.jsp.

Obtaining PDF and hardcopy documentation

- A subset of the PDF file books are available through the InfoSphere Information Server software installer and the distribution media. The other PDF file books are available online and can be accessed from this support document: https://www.ibm.com/support/docview.wss?uid=swg27008803&wv=1.
- You can also order IBM publications in hardcopy format online or through your local IBM representative. To order publications online, go to the IBM Publications Center at http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss.

Providing feedback about the documentation

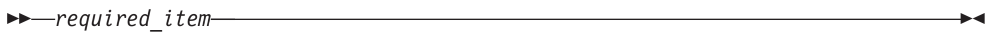
You can send your comments about documentation in the following ways:

- Online reader comment form: www.ibm.com/software/data/rcf/
- E-mail: comments@us.ibm.com

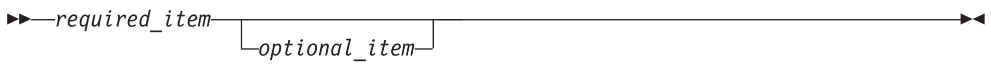
How to read syntax diagrams

The following rules apply to the syntax diagrams that are used in this information:

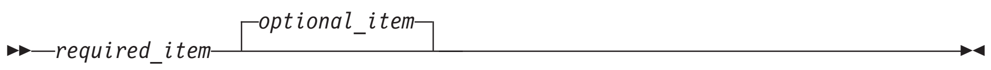
- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following conventions are used:
  - The >>--- symbol indicates the beginning of a syntax diagram.
  - The ---> symbol indicates that the syntax diagram is continued on the next line.
  - The >--- symbol indicates that a syntax diagram is continued from the previous line.
  - The --->< symbol indicates the end of a syntax diagram.
- Required items appear on the horizontal line (the main path).



- Optional items appear below the main path.

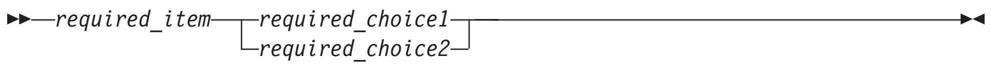


  If an optional item appears above the main path, that item has no effect on the execution of the syntax element and is used only for readability.

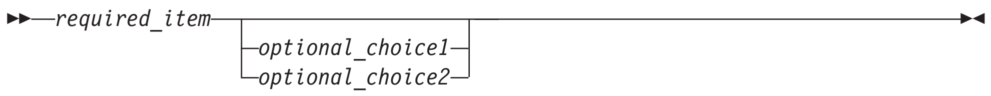


- If you can choose from two or more items, they appear vertically, in a stack.

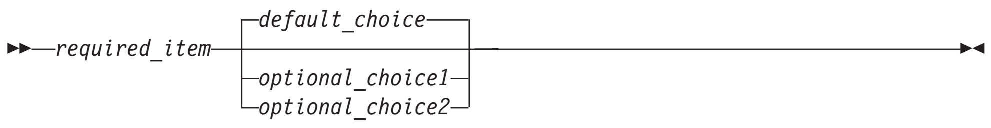
  If you must choose one of the items, one item of the stack appears on the main path.



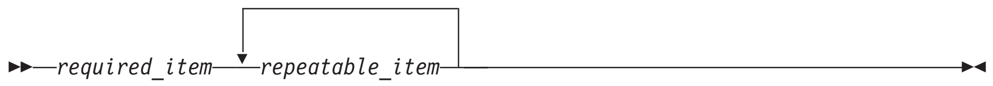
  If choosing one of the items is optional, the entire stack appears below the main path.



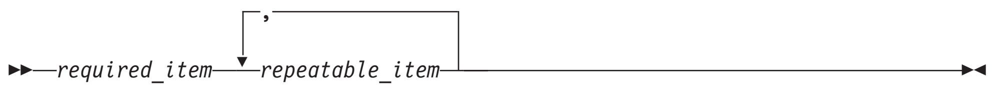
  If one of the items is the default, it appears above the main path, and the remaining choices are shown below.



- An arrow returning to the left, above the main line, indicates an item that can be repeated.

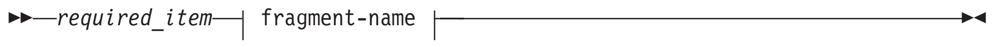


If the repeat arrow contains a comma, you must separate repeated items with a comma.

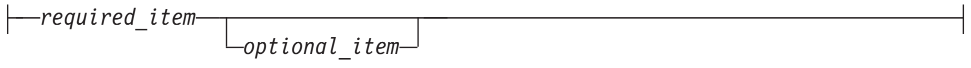


A repeat arrow above a stack indicates that you can repeat the items in the stack.

- Sometimes a diagram must be split into fragments. The syntax fragment is shown separately from the main syntax diagram, but the contents of the fragment should be read as if they are on the main path of the diagram.



Fragment-name:



- Keywords, and their minimum abbreviations if applicable, appear in uppercase. They must be spelled exactly as shown.
- Variables appear in all lowercase italic letters (for example, **column-name**). They represent user-supplied names or values.
- Separate keywords and parameters by at least one space if no intervening punctuation is shown in the diagram.
- Enter punctuation marks, parentheses, arithmetic operators, and other symbols, exactly as shown in the diagram.
- Footnotes are shown by a number in parentheses, for example (1).

Product accessibility

You can get information about the accessibility status of IBM products.

The IBM InfoSphere Information Server product modules and user interfaces are not fully accessible. The installation program installs the following product modules and components:

- IBM InfoSphere Business Glossary
- IBM InfoSphere Business Glossary Anywhere
- IBM InfoSphere DataStage
- IBM InfoSphere FastTrack
- IBM InfoSphere Information Analyzer
- IBM InfoSphere Information Services Director
- IBM InfoSphere Metadata Workbench
- IBM InfoSphere QualityStage

For information about the accessibility status of IBM products, see the IBM product accessibility information at http://www.ibm.com/able/product_accessibility/index.html.

Accessible documentation

Accessible documentation for InfoSphere Information Server products is provided in an information center. The information center presents the documentation in XHTML 1.0 format, which is viewable in most Web browsers. XHTML allows you to set display preferences in your browser. It also allows you to use screen readers and other assistive technologies to access the documentation.

IBM and accessibility

See the IBM Human Ability and Accessibility Center for more information about the commitment that IBM has to accessibility.

Notices and trademarks

This information was developed for products and services offered in the U.S.A.

Notices

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web

Trademarks

Index

Printed in USA

SC19-3437-01