Informatica

Informatica® PowerExchange for WebSphere MQ

10.4.0

User Guide for PowerCenter

Informatica PowerExchange for WebSphere MQ User Guide for PowerCenter
10.4.0
December 2019

Publication Date: 2021-12-03

Table of Contents

Preface

Use the *Informatica® PowerExchange® for WebSphere MQ User Guide* to learn how to read data from and write data to WebSphere MQ by using PowerCenter Client. Learn to create a WebSphere MQ connection, develop mappings, and run sessions in an Informatica domain.

Informatica Resources

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

Informatica Network

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

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

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

Informatica Knowledge Base

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

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

Informatica Documentation

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

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

Informatica Product Availability Matrices

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

Informatica Velocity

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

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

Informatica Marketplace

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

Informatica Global Customer Support

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

To find your local Informatica Global Customer Support telephone number, visit the Informatica website at the following link:
https://www.informatica.com/services-and-training/customer-success-services/contact-us.html.

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

CHAPTER 1

Understanding PowerExchange for WebSphere MQ

This chapter includes the following topics:

Understanding PowerExchange for WebSphere MQ Overview

PowerExchange for WebSphere MQ integration allows you to read data from WebSphere MQ message queues and write data into WebSphere MQ message queues.

Message queues are used in environments where applications communicate by sending each other data in messages rather than calling each other directly. An application can request data from another application by putting a request message on a message queue. The receiving application retrieves the message from the queue, processes the request, and generates a reply message on a message queue. A messaging and queueing architecture allows programs to run independently of each other, at different speeds and times, and in different locations without having a logical connection between them.

Using PowerExchange for WebSphere MQ integration, you can read data from messages in a queue, transform the data according to your business rules, and then write the data to a target data warehouse or message queue.

This chapter provides an overview of the integration between PowerCenter and WebSphere MQ.

WebSphere MQ Architecture

WebSphere MQ is a messaging and queueing application that enables programs to communicate with one another across heterogeneous platforms and network protocols using a consistent application programming interface.

PowerExchange for WebSphere MQ interacts with the following components of the WebSphere MQ architecture:

- Queue manager
- Message queue
- WebSphere MQ messages

Queue Manager

An application connects to a queue manager to send or receive messages through a message queue. For example, PowerExchange for WebSphere MQ uses a queue manager to read source messages from a queue and write target messages to a queue. All message queues in a WebSphere MQ system belong to a queue manager. The queue manager provides functions to administer queues, create new queues, or control the operation of the queue manager.

Message Queue

A message queue is a destination to which messages can be sent. One program sends a message to another by putting the message on a message queue. The message remains on the queue until the receiving application retrieves it. For example, PowerExchange for WebSphere MQ reads messages from a queue and writes messages to a queue. The queue can either be a volatile buffer area in the memory of a computer or a data set on a permanent storage device set aside by the queue manager to hold messages.

WebSphere MQ Messages

A WebSphere MQ message is a collection of data that one program sends to another program. It has the following components:

- Message header
- Message data

Message Header

The message header component contains data about the message on the queue. Message header data includes a message identification number, message format information, and other message descriptor data. In PowerCenter, MQSeries source definitions and dynamic MQSeries target definitions contain WebSphere MQ message header fields.

The following table lists all the fields in the message header component of a WebSphere MQ message, including descriptions and datatypes:

WebSphere MQ Message Header	Datatype	Description
StrucId	String(4)	Structure identifier.
Version	Number	Structure version number.
Report	Number	Options for report messages.
MsgType	Number	Message type.
Expiry	Number	Message lifetime.
Feedback	Number	Feedback or reason code.
Encoding	Number	Data encoding.
CodedCharSetId	Number	Coded character set identifier.
Format	String(8)	Format name.
Priority	Number	Message priority.
Persistence	Number	Message persistence.
MsgId	Binary(24)	Message identifier.
CorrelId	Binary(24)	Correlation identifier.
BackoutCount	Number	Backout counter.
ReplytoQ	String(48)	Name of reply queue.
ReplytoQMgr	String(48)	Name of reply queue manager.
UserIdentifier	String(12)	User identifier.
AccountingToken	Binary(32)	Accounting token.
ApplIdentityData	String(32)	Application data relating to identity.
PutApplType	Number	Type of application that put the message on queue.
PutApplName	String(28)	Name of application that put the message on queue.
PutDate	String(8)	Date when message was put on queue.
PutTime	String(8)	Time when message was put on queue.
ApplOrigData	String(4)	Application data relating to origin.
GroupId	Binary(24)	Group identifier.
MsgSeqNumber	Number	Sequence number of logical messages within group.

WebSphere MQ Message Header	Datatype	Description
Offset	Number	Offset of data in physical message from start of logical message.
MsgFlags	Number	Message flags.
OriginalLength	Number	Length of original message.

Message Data

Message data is the application data or the contents of the message body. For example, if an application sends a list of customer names to another application, the customer names comprise the message data component of the message. The content and format of the message data is defined by the application that uses the message queue. WebSphere MQ does not enforce any structure on the content and format of the message data. In PowerCenter, MQSeries source definitions and dynamic MQSeries target definitions contain a message data field.

WebSphere MQ Sources and Targets

With PowerExchange for WebSphere MQ, you can read data from WebSphere MQ message queues and write data to WebSphere MQ message queues.

WebSphere MQ Sources

When you read data from a WebSphere MQ source, you read data from the message header and the message data components of a WebSphere MQ message. The PowerCenter Integration Service treats the message header component as message header fields. The message header fields contain information about the message on the queue. Using the message header data, you can filter messages you want to read. For example, you can read messages based on the message format, or you can filter messages that originated from a particular application.

PowerCenter treats the message data component of a message as a field containing message data. The PowerCenter Integration Service can read message data in the following formats:

- Flat file (fixed-width or delimited)
- XML
- COBOL
- Binary

WebSphere MQ Targets

The PowerCenter Integration Service can write data to the message header and the message data fields of a WebSphere MQ message. It can write message data to a WebSphere MQ message in the following formats:

- Flat file (fixed-width or delimited)
- XML
- COBOL

- Binary

WebSphere MQ Source Qualifier Transformations

An MQSeries source definition may require two types of source qualifiers depending on the format of the message data you are reading:

- MQ Source Qualifier transformation
- Associated source qualifier transformation

MQ Source Qualifier Transformation

The MQ Source Qualifier allows you to read data from a WebSphere MQ source. It contains a predefined set of ports representing all the message header fields and the message data field in a WebSphere MQ message. With an MQ Source Qualifier, you can read binary and XML message data from a WebSphere MQ message. If the source message contains flat file or COBOL message data, you also need to use an associated source qualifier.

Associated Source Qualifier Transformation

The associated source qualifier transformation is specific to the message data format. The PowerCenter Integration Service requires an associated source qualifier to read flat file or COBOL message data.

Understanding Secure Sockets Layer

To manage the security of messages transmitted on the Web, PowerExchange for WebSphere MQ uses the Secure Sockets Layer (SSL) protocol. SSL helps you achieve authentication, integrity, and privacy on the Web. It allows for secure communication between client and server by allowing mutual authentication, use of digital signatures on messages for integrity, and encryption for privacy.

Configure the communication channels between the WebSphere MQ server and PowerExchange for WebSphere MQ to use SSL.

Related Topics:

- "Step 1. Configure Channels for Queue Managers" on page 16

Using Code Pages

When the PowerCenter Integration Service reads data from WebSphere MQ sources that contain a different code page than the PowerCenter Integration Service, the PowerCenter Integration Service can convert the data to the PowerCenter Integration Service code page if the data is in string format.

When the PowerCenter Integration Service writes data to dynamic WebSphere MQ targets that contain a different code page than the PowerCenter Integration Service, the PowerCenter Integration Service can

convert the data to the code page of the WebSphere MQ server when you set the data conversion option to “yes” for the remote channel definition in WebSphere MQ.

Processing Unicode Data

If the PowerCenter Integration Service runs the WebSphere MQ server library, it uses the coded character set identifier (CCSID) of the message header to process Unicode message body data. If it cannot determine the CCSID of the message header, it uses the CCSID of the queue manager to process Unicode message body data.

If the PowerCenter Integration Service uses the WebSphere MQ client library, it uses the CCSID of the message header to process Unicode message body data. If it cannot determine the CCSID of the message header, it uses the code page in the CCSID of the MQCCSID environment variable to process Unicode data. If there is no value for the MQCCSID environment variable, the PowerCenter Integration Service uses the default code page of the WebSphere MQ system.

The PowerCenter Integration Service uses the code page in the queue connection in the following cases:

- The PowerCenter Integration Service cannot retrieve the CCSID of the queue manager or MQCCSID environment variable when it uses the WebSphere MQ client library. You might not have authorization on WebSphere MQ to retrieve this information.
- The CCSID value of a WebSphere MQ message is invalid, or WebSphere MQ does not support the value, and the PowerCenter Integration Service cannot use the CCSID of the queue manager.

CHAPTER 2

Configuring PowerExchange for WebSphere MQ

This chapter includes the following topics:

Configuring PowerExchange for WebSphere MQ Overview

PowerExchange for WebSphere MQ installs with the PowerCenter Services. To use PowerExchange for WebSphere MQ, install and configure IBM WebSphere MQ client or WebSphere MQ server on the machine that you have installed the PowerCenter Integration Service. The system administrator must perform the installation and configuration of the IBM WebSphere MQ client and the WebSphere MQ server.

After you install and configure PowerExchange for WebSphere MQ, you must set the environment variables on the machine running the PowerCenter Integration Service.

Minimum System Requirements

Ensure the minimum system requirements are met. For more information about minimum system requirements, see the *PowerCenter Configuration Guide*.

Prerequisites

Before you upgrade or install PowerExchange for WebSphere MQ, install the WebSphere MQ client or WebSphere MQ server on the machine that runs the PowerCenter Integration Service.

Note: You cannot install WebSphere MQ 9.1 client on Suse11 SP4 and AIX 7.1 TL4 operating system versions.

Configuring PowerExchange for WebSphere MQ

To configure PowerExchange for WebSphere MQ, complete the following steps:

1. Configure client-connection channels for a queue manager.
2. Set system environment variables.
3. Configure WebSphere MQ for Unicode (optional).

Step 1. Configure Channels for Queue Managers

If the WebSphere MQ system does not have a client-connection channel, configure a channel for each queue manager you want to connect to. WebSphere MQ does not allow you to connect to multiple queue managers in a session.

WebSphere MQ creates server-connection channels to connect to WebSphere MQ. When you configure PowerExchange for WebSphere MQ, you do not need to configure a server-connection channel unless the WebSphere MQ administrator has changed the default settings for server-connection channels in the queue manager. For more information about configuring a server-connection channel, see the WebSphere MQ documentation.

Note: For more information about Websphere MQ client connectivity to Websphere MQ server or channel, contact IBM technical support.

To create a client-connection channel:

1. From the command prompt on the machine running the PowerCenter Integration Service process, enter the following command to start WebSphere MQ commands:

   ```
   runmqsc <queue manager name>
   ```

2. Enter the following command to create the client-connection channel:

   ```
   DEFINE CHANNEL (<channel_name>) CHLTYPE (<channel_type>)
   CONNAME (<WebSphere MQ_server_IP_address>) QMNAME (<queue_manager_name>)
   ```

   The following table describes the command parameters for defining the client-connection channel:

Option	Required/ Optional	Argument	Description
DEFINE CHANNEL	Required	channel_name	Name for the channel.
CHLTYPE	Required	channel_type	Enter CLNTCONN.
CONNAME	Required	WebSphere MQ_server_IP_address	IP address of the machine hosting the WebSphere MQ server.
QMNAME	Required	queue_manager_name	Queue manager name.

   A message appears stating the channel was successfully created.

3. Repeat step 2 for each queue manager you want the PowerCenter Integration Service to connect to.

Step 2. Set Environment Variables

The PowerCenter Integration Service uses WebSphere MQ environment variables to connect to source and target message queues. Set the environment variables on the machine running the PowerCenter Integration Service process before you can run sessions with WebSphere MQ sources or targets. Also set environment variables when you configure the PowerCenter Integration Service to run on AIX and you install the WebSphere MQ server driver.

Setting Environment Variables on the PowerCenter Integration Service Machine

Set system environment variables before starting the PowerCenter Integration Service process.

The following table describes the system environment variables that you can set:

Environment Variable	Description
MQCHLTAB	Name of the file that contains the client channel definition table.
MQCHLLIB	Directory path to the file that contains the client channel definition table.
MQSERVER	Location of the WebSphere MQ server and the communication method you want to use. When you specify MQSERVER, the PowerCenter Integration Service can read and write WebSphere MQ messages up to 4 MB.
MQSSLKEYR	Location of the key repository that contains the digital certificate for secure sockets layer (SSL) configuration. See the IBM documentation for other SSL environment variables you might want to configure.

Use the following guidelines:

- Set MQCHLTAB and MQCHLLIB for the PowerCenter Integration Service to be able to connect to multiple queue managers in a session.
- To configure WebSphere MQ for SSL, set MQSSLKEYR, MQCHLTAB, and MQCHLLIB. Do not set MQSERVER.
- To connect to one queue manger in a session without secure connections, set MQSERVER. When you set MQSERVER, WebSphere MQ ignores MQCHLTAB, MQCHLLIB, and MQSSLKEYR.

WebSphere MQ also provides other system environment variables. For more information, see the IBM WebSphere MQ documentation.

To set system environment variables:

1. Stop the PowerCenter Integration Service process.

2. Set the following values for the environment variables on the machine running the PowerCenter Integration Service process:

Environment Variable	Value
MQCHLTAB	MQCHLTAB = AMQCLCHL.TAB
MQCHLLIB	MQCHLLIB = <WebSphereMQ installation directory>/Qmgrs/<queue manager name>/@ipcc
MQSSLKEYR	MQSSLKEYR = <WebSphereMQ installation directory>/Qmgrs/<queue manager name>/ssl/ <key repository>
MQSERVER	MQSERVER = <server-connection channel name>/TCP/<host name>(<port nunber>) For example: MQ_cseversonpc/TCP/cseversonpc(1414)

3. Start the PowerCenter Integration Service process.

Setting System Environment Variables on AIX

When you configure the PowerCenter Integration Service to run on AIX and you install the WebSphere MQ server driver, set the memory address space layout to allow the IPC segment to allocate memory for the PowerCenter Integration Service.

To set system environment variables on AIX:

1. Set the LDR_CNTRL system environment variable to MAXDATA=0x60000000 for the PowerCenter Integration Service.

   In a C shell environment, type:

   ```
   setenv LDR_CNTRL MAXDATA=0x60000000
   ```

   In a Bourne shell environment, type:

   ```
   LDR_CNTRL = "MAXDATA=0x60000000"; export LDR_CNTRL
   ```

2. Stop the WebSphere MQ queue manager.
3. Set the IPCCBaseAddress address parameter in the mqs.ini file to reallocate system memory for WebSphere MQ.
4. Enter 11 for the IPCCBaseAddress parameter in the mqs.ini file to reallocate system memory for WebSphere MQ. For example, enter:

   ```
   IPCCBaseAddress=11
   ```

Step 3. Configure WebSphere MQ Messages for Unicode

To configure messages for Unicode in WebSphere MQ, complete the following tasks:

1. In the Workflow Manager, set a code page in the MQ queue connection object.
2. If the PowerCenter Integration Service runs the WebSphere MQ server library, set the coded character set identifier (CCSID) in the queue manager. For example, if you set the CCSID to 1208, messages are UTF-8 encoded.
3. If the PowerCenter Integration Service uses the WebSphere MQ client library, set the CCSID in the MQCCSID environment variable.

CHAPTER 3

Working with WebSphere MQ Sources

This chapter includes the following topics:

Working with WebSphere MQ Sources Overview

When you read data from WebSphere MQ, you read data from messages in a WebSphere MQ message queue. An MQSeries source definition contains message header fields and a message data field. The message header fields contain information about the message. The message data field contains the actual message. The message data can be one or more rows of binary, flat file, COBOL, or XML data. The message data format determines the types of source definitions you need to use.

Use the following types of source definitions in a mapping:

- MQSeries source definition
- Associated source definition

MQSeries Source Definition

When you read data from WebSphere MQ, use an MQSeries source definition. The MQSeries source definition defines all message header fields and the message data field in a WebSphere MQ message. If the source contains binary or XML message data, use the MQSeries source definition to read and transform the message data.

Associated Source Definition

An associated source definition is specific to the message data format and defines the message data field in the WebSphere MQ message. An associated source definition can be a flat file or VSAM source definition. If the WebSphere MQ message contains flat file or COBOL message data, you need to use an associated source definition with the MQSeries source definition to read and transform the message data. You join the MQSeries source definition with the associated source in the MQ Source Qualifier.

Working with an MQSeries Source Definition

The MQSeries source definition represents the metadata for the WebSphere MQ source. Since all WebSphere MQ messages contain the same message header and message data fields, the Designer provides an MQSeries source definition with predefined column names to use in all WebSphere MQ mappings.

Creating and Editing an MQSeries Source Definition

You manually create an MQSeries source definition. When you create an MQSeries source definition, the Designer displays a table with all the message header fields and the message data field from a WebSphere MQ message.

You can edit the source definition to change the datatype and precision of the columns. If the mapping that contains the source definition also contains an associated source definition, the datatype for the column MsgId must be MQBYTE.

Message Header Fields

There are 29 message header fields in a WebSphere MQ message. Message header fields include MsgId, Format, PutTime, PutDate, and other message descriptor fields.

The MsgId field is a primary key field in MQSeries source definitions. It uniquely identifies one message from another within a message queue. When you use an associated source definition to define the message data, the Designer uses the MsgId port from the header fields to associate the WebSphere MQ source with the message data. This association occurs in the MQ Source Qualifier.

When you pass the message header fields through an MQ Source Qualifier, you can also use the message header fields to filter messages that PowerCenter reads from the message queue. For example, a message queue may contain messages from several applications or messages in several formats. You can create a filter condition in the MQ Source Qualifier that allows the PowerCenter Integration Service to only read messages from a particular application and messages of a particular format.

Related Topics:

- "Message Header" on page 10

Message Data Field

The message data field in the MQSeries source definition defines the message data in the WebSphere MQ message. It contains one or more rows of message data that one program sends to another program through a message queue.

The message data field in has an MQ-specific datatype. When the PowerCenter Integration Service reads data from the message data field, it treats the message data as binary unless you define the message data with an associated source definition or use an XML Parser transformation.

The PowerCenter Integration Service treats the message data differently depending on the message data format. When the message data is binary, the PowerCenter Integration Service does not transform the message data. If the message data is flat file or COBOL, you define the message data in the repository with an associated source definition. If the message data is XML, use an XML Parser transformation. When you use an associated source definition or a XML Parser transformation in a mapping, the PowerCenter Integration Service reads data WebSphere MQ, transforms the message data, and then writes the output to a target.

Reading XML Message Data from WebSphere MQ

When a WebSphere MQ message includes XML message data, use the XML Parser transformation to read the data and parse the data into one or more targets. The XML Parser transformation speeds processing by parsing data in the pipeline through the transformation rather than in the target at the end of file.

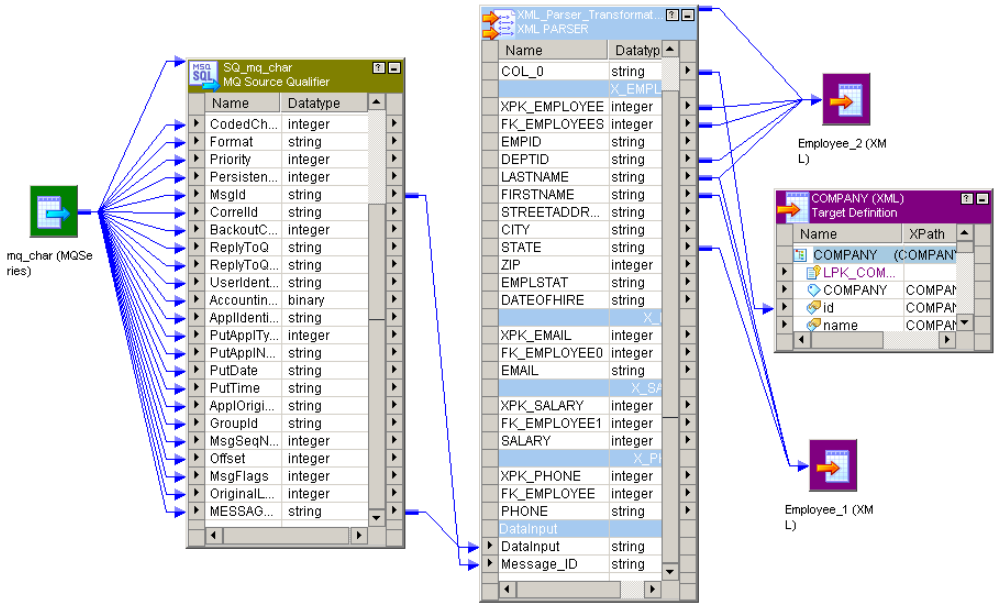
When you create an XML Parser transformation, use the XML Wizard and XML Editor to define the XML groups.

Creating a Pass-Through Port

If you want to pass the message ID of the WebSphere MQ message from the source to the target, you can create pass-through ports in the XML Parser transformation. Use message ID data to correlate input and output messages for requests and replies.

MQ message ID data entered in the new input port passes through to the corresponding reference output port to the target. The reference output port passes the data directly to the target table.

The following figure shows a mapping with a Message_ID port added as a pass-through port in the XML Parser transformation:



This example shows the Message_ID port in the MQ Source Qualifier linked to the Message_ID pass-through port in the XML Parser transformation. The XML Parser transformation passes the data through the reference output port COL0 and to the ID port in the target table.

Streaming Messages to an XML Parser Transformation

You can configure sessions to stream the XML between the WebSphere MQ sources and the XML Parser transformation. When the PowerCenter Integration Service streams XML data, it splits XML data from the MQ Source Qualifier into multiple segments. You can configure a smaller input port in the XML Parser transformation and reduce the amount of memory that the XML Parser transformation requires to process large XML files. The buffer pool size is from 50 to 100 percent smaller, depending on message size.

When you enable XML streaming, the XML Parser transformation receives the XML data in memory. The PowerCenter Integration Service compares the XML file size to the transformation input port precision. If the

XML file is larger than the port precision, the PowerCenter Integration Service divides the data into rows that are less than or equal to the port precision. The PowerCenter Integration Service sets each row type as streaming except for the last row. The last row has a row type insert.

If you enable XML streaming for the WebSphere MQ message, you must also enable it in the XML Parser transformation.

To enable XML streaming for WebSphere MQ:

1. Open the session.
2. In session properties, click the Mapping Tab.
3. Click the WebSphere MQ Source Qualifier in the left panel.
4. Enable Streaming in the WebSphere MQ Source Qualifier session properties.
5. Click the XML Parser transformation in the left panel.
6. Enable XML Input Streaming in the XML Parser transformation session properties.

Working with an Associated Source Definition

The associated source definition in a WebSphere MQ mapping represents the metadata for the message data. You can import the following types of associated source definitions to define the message data in a WebSphere MQ mapping:

- Flat file source definition
- VSAM source definition

Importing an Associated Source Definition

You create the associated source definition in the Designer by importing a source definition that matches the message data format and structure in the WebSphere MQ source. For example, if the message data is flat file, import a flat file source definition. The flat file structure must define the message data. Based on the flat file format and structure, the Designer imports a source definition for the message data.

If the message data in the source is COBOL, import a VSAM source definition from a COBOL file that matches the message data in the source.

Note: The PowerCenter Client does not validate the message data structure in the MQSeries source definition. If the source definition does not match the actual source data, the session fails.

To import an associated source definition:

1. In the Source Analyzer, click Sources.
2. Depending on the format of the WebSphere MQ message data, select one of the following options:
   - Import from File
   - Import from COBOL File
3. The wizard prompts you for the necessary file properties of the associated source definition.

CHAPTER 4

MQ Source Qualifier Transformation

This chapter includes the following topics:

MQ Source Qualifier Transformation Overview

Depending on the type of data you want to read from WebSphere MQ, use one or both of the following types of source qualifier transformation:

- **MQ Source Qualifier transformation.** The MQ Source Qualifier reads all the message header fields and the message data field in a WebSphere MQ message. You can enter filter conditions in the MQ Source Qualifier to determine which messages to read from the source. You can also associate the MQ Source Qualifier with an associated source qualifier.
- **Associated source qualifier.** The associated source qualifier is specific to the format of the WebSphere MQ message data. It determines how the PowerCenter Integration Service reads the message data. You must use a Source Qualifier or Normalizer transformation if the message data is flat file or COBOL.

Working with an MQ Source Qualifier

The Designer connects each column of the MQSeries source definition to a corresponding port in the MQ Source Qualifier. The connection from a source column to an input port in the MQ Source Qualifier is predefined. The PowerCenter Integration Service reads data from all of the message header fields in a WebSphere MQ message. You cannot delete any of the connections.

To read data from multiple WebSphere MQ message queues, you need an MQ Source Qualifier for each MQSeries source definition. You cannot join two sources with an MQ Source Qualifier in a mapping.

Working with an Associated Source Qualifier

Use an associated source qualifier when the WebSphere MQ message data is flat file or COBOL. Without the associated source qualifier, the MQ Source Qualifier treats the message data as binary. The PowerCenter Integration Service can read binary data from the message data field up to 100 MB. It does not transform the binary data.

The associated source qualifier is specific to the format of the message data. It determines how the PowerCenter Integration Service reads the message data from WebSphere MQ.

The following table lists the two types of associated source qualifiers you can use in a WebSphere MQ mapping and the corresponding message data format:

Message Data Format	Associated Source Qualifier
Flat file	Source Qualifier transformation
COBOL	Normalizer transformation

When you add an associated source qualifier to a mapping, associate it with an MQ Source Qualifier. The association allows the PowerCenter Integration Service to read the message data from WebSphere MQ based on the transformation settings in the associated source qualifier. For example, if the source data is flat file, use a flat file Source Qualifier to read the message data.

When you associate an MQ Source Qualifier with a Source Qualifier for a flat file source definition, the Designer creates a MsgId port in the Source Qualifier with the same datatype and precision as the MsgId port in the MQ Source Qualifier. It links the MsgId ports in the MQ Source Qualifier and the Source Qualifier for the flat file source definition. The association through the MsgId preserves the relationship between the header data and the message data in the WebSphere MQ source. It allows the PowerCenter Integration Service to correlate the messages in the queue with their message data.

When you run a session, the PowerCenter Integration Service reads the messages from WebSphere MQ based on the settings of the MQ Source Qualifier. For each message that it reads, it also reads the message data according to the configurations of the Source Qualifier for the flat file source definition. Similarly, if the message data in the source is COBOL, use a Normalizer transformation to read the message data.

Filtering Messages from the Source Queue

You can use the following methods to filter messages from the source queue with an MQ Source Qualifier:

- **Functions in filter conditions.** Use to control the end of file of the WebSphere MQ source, real-time data extraction, end of source queue, and incremental extraction of messages from queue.
- **Destructive Read option.** Evaluates filter conditions in the message header and removes messages from the source queue.

You can use the following built-in functions in the MQ Source Qualifier:

- **Concat(*str1, str2*).** Concatenates two strings into a single string. Use Concat with a mapping variable to control incremental extraction of messages.
- **MsgCount(*n*).** Controls the number of messages that the PowerCenter Integration Service reads from the queue.

- **Idle(*n*).** Controls the duration of time the PowerCenter Integration Service remains idle before it stops reading from the queue.
- **StartTime(*time*).** Defines the time in Greenwich Mean Time for the PowerCenter Integration Service to start reading messages from the queue.
- **EndTime(*time*).** Defines the time in Greenwich Mean Time for the PowerCenter Integration Service to stop reading messages from the queue.
- **FlushLatency(*n*).** Defines a specified period of time for the PowerCenter Integration Service to commit messages read from the source queue.
- **ForcedEOQ(*n*).** Defines a specified period of time for the PowerCenter Integration Service to read messages from the source queue.
- **RemoveMsg(TRUE).** Removes messages from the source queue.

Set the Destructive Read option and filter conditions on the Properties tab of the Source Qualifier.

You can also select Destructive Read and enter filter conditions in the session properties. Destructive Read and filter conditions you set at the session level override settings in the Source Qualifier.

Controlling End of File

You can configure the MQ Source Qualifier to control the end of file of the WebSphere MQ source. The end of file of an WebSphere MQ source determines when the PowerCenter Integration Service stops reading from the source queue. Enter functions in a filter condition to control the end of file of the WebSphere MQ source.

You can define the following queue reading modes for the PowerCenter Integration Service by controlling the end of file of the source:

- Message count mode
- Idle mode
- Time slice mode

Message Count Mode

Use MsgCount(*n*) to specify the end of file in an WebSphere MQ source. When you use MsgCount(*n*), the PowerCenter Integration Service reads a specified number of messages from the queue and stops reading. Enter the following expression when you use MsgCount(*n*):

```
MsgCount(n)
```

where *n* is the number of messages the PowerCenter Integration Service reads from the message queue. For example, to read 100 messages from the source queue, enter the following filter condition:

```
MsgCount(100)
```

If you enter MsgCount(*n*), and you configure the session to use pipeline partitioning, the session can run on a single node only. The PowerCenter Integration Service that runs the session cannot run on a grid or on primary and backup nodes.

Idle Mode

Use Idle(*n*) to indicate how long the PowerCenter Integration Service waits for the queue to be idle before it stops reading from the queue. Enter the following expression when you use Idle(*n*):

```
Idle(n)
```

where *n* is an integer representing seconds. For example, if the value of *n* is 30, the PowerCenter Integration Service waits 30 seconds after reading from the queue. If no new messages arrive on the queue within 30 seconds, the PowerCenter Integration Service stops reading from the queue.

Time Slice Mode

Use StartTime(*time*) with EndTime(*time*) to define a specified range of time for the PowerCenter Integration Service to read messages from the queue. The value you enter for EndTime(*time*) determines the end of file of the WebSphere MQ source. If StartTime(*time*) is not used in the filter condition, the PowerCenter Integration Service reads all messages up to the specified EndTime(*time*). Time values in the function must be relative to Greenwich Mean Time. The format must be "MM/DD/YYYY H24:MM:SS." For example, to read messages from 9:00 AM to 5:30 PM on September 02, 2000, enter the following filter condition:

```
StartTime("09/02/2000 09:00:00") && EndTime("09/02/2000 17:30:00")
```

If you use the $$$SessStartTime variable instead of a specific time, you must enclose the variable in quotes. For example, enter:

```
StartTime("$$$SessStartTime")
```

Controlling Data Extraction in Real Time

You can configure flush latency to process data in real time. A real-time session reads, processes, and writes data to targets continuously. Flush latency determines how often the PowerCenter Integration Service flushes data from the source.

In the MQ Source Qualifier, use FlushLatency(*n*) to commit messages to the target in real time. Enter the following expression when you use FlushLatency(*n*):

```
FlushLatency(n)
```

where *n* is an integer representing seconds. You can configure the session to commit messages to the target in milliseconds rather than seconds. Specify the Millisecond Flush Latency option in the session properties.

Removing Messages from the Source Queue

You can remove messages from the source queue in real time. Specify FlushLatency(*n*) and the Destructive Read option in the MQ Source Qualifier or the session properties.

Related Topics:

- "Extracting Data in Real Time" on page 42

Controlling Forced End of Queue

Use ForcedEOQ(*n*) to define the end of the WebSphere MQ source after a specified period of time. WebSphere MQ can contain an infinite number of messages entering the queue. When you use ForcedEOQ(*n*), the PowerCenter Integration Service stops reading from the source queue after the period you specify or if the queue is empty, whichever comes first.

Use the following expression in the filter condition to set ForcedEOQ(*n*):

```
ForcedEOQ(n)
```

where *n* is an integer representing seconds. For example, if the value of *n* is 60, the PowerCenter Integration Service reads messages from the queue for 60 seconds before stopping.

Use ForcedEOQ(*n*) to run a session in real time when the mapping configuration or session properties do not support FlushLatency(*n*).

For example, you want to run a real-time session with an WebSphere MQ mapping that contains an Aggregator transformation with the All Input option selected for the Transformation Scope property. When a mapping contains an Aggregator transformation with these conditions, the PowerCenter Integration Service cannot use FlushLatency(*n*) to commit messages to a target based on a specified maximum latency period.

To run the session in real time, use ForcedEOQ(*n*) and configure the workflow to run continuously. When you use ForcedEOQ(*n*), the PowerCenter Integration Service stops reading messages from the source at the end of the ForcedEOQ(*n*) period. The PowerCenter Integration Service can then write the data to the target.

Related Topics:

- "Extracting Data in Real Time" on page 42

Controlling Incremental Extraction

In the MQ Source Qualifier, use functions in a filter condition to control incremental extraction of messages from WebSphere MQ. For example, you want to read new messages from the queue and filter out messages that were read in the last session. In the filter expression, use Concat to combine the PutDate and PutTime values from the message header fields into a single string. Then compare the concatenated value to a mapping variable that defines the maximum PutDate | PutTime combination.

For example, to read messages from a queue that have a PutDate | PutTime value greater than the maximum PutDate | PutTime value defined by a mapping variable $$MAX_DATETIME, enter the following filter expression:

```
Concat (PutDate, PutTime) > "$$MAX_DATETIME"
```

Controlling Queue Clean-up

You can control queue clean-up to remove messages from the WebSphere MQ source queue by selecting one of the following:

- **Destructive Read.** Select the Destructive Read option to remove messages from the source queue and evaluate filter conditions in the message header ports.
- **Remove Msg(TRUE).** Enter the Remove Msg(TRUE) function in a filter condition to remove messages from the source queue when using an associated source qualifier.

If you select Destructive Read, you must set RemoveMsg(TRUE) to RemoveMsg(FALSE) in the filter condition or remove the function from the filter condition. If you enable both Destructive Read and RemoveMsg(TRUE), the session fails during initialization.

Controlling Queue Clean-up with Destructive Read

To control queue clean-up to remove messages from the WebSphere MQ source queue, select the Destructive Read option in the MQ Source Qualifier. DestructiveRead lets you evaluate filter conditions in the message header ports and remove messages from the source queue. If a message does not pass the filter condition, the PowerCenter Integration Service does not remove the message from the source queue.

If you do not select Destructive Read, messages remain in the source queue. The PowerCenter Integration Service reads them again each time you run the session.

If you select Destructive Read, the mapping must contain an MQ Source Qualifier. If you link the MESSAGE_DATA port of the MQ Source Qualifier to a downstream transformation, and the PowerCenter Integration Service truncates the message while reading it from the WebSphere MQ queue, the session fails. The PowerCenter Integration Service does not remove the message from the queue. It can read the message in the next session.

If you select Destructive Read, and the mapping contains an associated source qualifier, the PowerCenter Integration Service ignores the Destructive Read option. To remove messages from the source queue for mappings that contain an associated source qualifier, use RemoveMsg(TRUE).

Related Topics:

- "Configuring Destructive Read" on page 43

Controlling Queue Clean-up with the RemoveMsg(TRUE) Function

You can use RemoveMsg(TRUE) to remove messages from the source queue. Use RemoveMsg(TRUE) when a mapping contains an associated source qualifier.

RemoveMsg(TRUE) removes messages from the queue after committing all data to the target. For example, enter the following filter condition:

```
RemoveMsg(TRUE)
```

You can combine RemoveMsg(TRUE) with other functions. For example, you want to read 100 messages and remove them from the queue after the PowerCenter Integration Service writes the messages to the target. Enter the following function in the filter condition:

```
MsgCount(100) && RemoveMsg(TRUE)
```

You can enter a filter condition in the Filter Editor in the Properties tab of the Source Qualifier or the Mapping tab of the session properties.

Entering a Filter Condition in the MQ Source Qualifier

In the MQ Source Qualifier, you can enter a filter condition to reduce the number of messages the PowerCenter Integration Service reads from the message queue. When you enter a filter in the MQ Source Qualifier, the PowerCenter Integration Service only reads messages that fulfill the filter condition. Use the PowerCenter transformation language to enter a single filter condition or a series of conditions.

MQ Source Qualifier Filter Syntax

The filter condition is an expression that returns TRUE or FALSE. For example, if you want the PowerCenter Integration Service to read messages from the queue that are of the lowest priority, enter the following condition:

```
Priority == 0
```

where 0 refers to the lowest priority level.

When you create the filter condition, you do not need to specify TRUE or FALSE as values in the expression, since TRUE or FALSE are implicit return values from any condition you set. If the filter condition evaluates to NULL, the row is assumed to be FALSE.

You can include multiple components in the condition by using logical operators. For example, if you want to filter messages with the lowest priority and messages in "MQSTR" format, enter the following condition:

```
Priority == 0 AND Format == "MQSTR"
```

The following table lists the logical operators you can use to create filter conditions:

Operator	Textual Representation	Alternate Textual Representation
EQUAL TO	=	==
NOT EQUAL TO	!=	<>

Operator	Textual Representation	Alternate Textual Representation
LESS THAN	<	-
LESS THAN OR EQUAL TO	<=	-
GREATER THAN	>	-
GREATER THAN OR EQUAL TO	>=	-
NOT	not	!
AND	and	&&
OR	or	||
LEFT PARENTHESIS	(	-
RIGHT PARENTHESIS	)	-

The following table shows the precedence of the logical operators:

Precedence	Operators
Highest	Equals (==), Not Equal to (!=), Less than (<), Less than or equal to (<=), Greater than (>), Greater than or equal to (>=)
-	NOT
-	AND
Lowest	OR

MQ Source Qualifier Syntax Rules

Use the following rules to create filter conditions in the MQ Source Qualifier:

1. Use the following syntax in the filter condition if you create a filter based on message header ports:

   ```
   <Port> [=, >=, <=, <, >, < >]'value'
   ```

2. Use the following syntax in the filter condition if you create a filter with a function:

   ```
   <Function> ('value')
   ```

3. Use the correct parameters for the 'value' in the filter expression. For example, to filter messages by PutDate, you must use the correct date format used by the PutDate field:

   ```
   PutDate >= "20000901" && PutDate <= "20001001"
   ```

   This example filters messages that were put on the queue from September 1, 2000, to October 1, 2000. The format for the PutDate field is YYYYMMDD.

4. Use time values relative to Greenwich Mean Time and in the format H24MMSS. For example, to read all messages that were put on the queue at 5:30 PM on September 1, 2000, enter the following condition:

   ```
   PutDate = "20000901" && PutTime = "173000"
   ```

5. If the datatype of the value is String, use double quotes around the value. For example, to filter all messages where the ReplyToQ field is “PRODUCTION,” use the following syntax:

```
ReplyToQ = "PRODUCTION"
```

6. Separate multiple components of a filter condition using AND and OR logical operators.

Filtering Messages Using Message Header Ports

You can use all the message header ports in the MQ Source Qualifier to create a source filter condition. To evaluate the filter conditions when removing messages from the source, select the Destructive Read option in the Source Qualifier.

Entering a Filter Condition in the MQ Source Qualifier

Use the following procedure to enter a filter condition in the MQ Source Qualifier.

To enter a filter condition in the MQ Source Qualifier:

1. In the Mapping Designer, double-click the title bar of the MQ Source Qualifier.

   The Edit Transformation dialog box appears.
2. Click the Properties tab.
3. Click the right corner of the Filter option to open the Filter Editor.

   The Filter Editor dialog box appears.
4. On the Ports tab, double-click the Message Header port you want to use in the filter condition.

   You can also select a built-in function from the Variables tab. Click Variables > Built-in and select the function you want to use.
5. Use transformation language syntax to create the filter condition.
6. Click Validate to make sure that the filter condition has no errors.
7. Click OK twice.

Configuring an MQ Source Qualifier

You can configure the following properties for an MQ Source Qualifier:

- **Message data size.** Set the size of the message data in the MESSAGE_DATA field of the MQ Source Qualifier if the message data is binary.
- **Data filter.** Enter a filter to read particular messages from the source queue.
- **Queue clean-up.** Evaluate filter conditions in the message header fields and remove messages from the source queue.

To configure an MQ Source Qualifier:

1. In the Mapping Designer, open a WebSphere MQ mapping.
2. Double-click the title bar of the MQ Source Qualifier.

   The Edit Transformations dialog box appears.
3. Click the Ports tab.
4. Enter a precision for the message data port up to 100 MB.

Default is 64 KB.

The message header ports are predefined. You cannot change their configuration.

5. Click the Properties tab.
6. Enter properties for the MQ Source Qualifier.

   The following table describes the properties you can enter:

Property	Description
Filter	Filter condition the PowerCenter Integration Service uses when reading records.
Tracing Level	Amount of detail included in the session log when you run a session containing this transformation.
Destructive Read	Evaluates filter conditions specified in the message header ports and removes messages from the source queue at synchronization points.
Recovery Cache Folder	Cache folder where the PowerCenter Integration Service stores message data when reading from the WebSphere MQ source.

Creating and Configuring an Associated Source Qualifier

You create an associated source qualifier in a mapping depending on the WebSphere MQ message data format. You can create the following types of associated source qualifier transformations for WebSphere MQ message data:

- Source Qualifier transformation for flat file message data
- Normalizer transformation for COBOL message data

Note: Before you create an associated source qualifier, import the source definition that describes the message data you want to read from the WebSphere MQ message queue.

By default, the Designer creates an associated source qualifier when you add an associated source definition to a mapping. If you configure the Designer to create a source definition without a Source Qualifier, you can create the associated source qualifier manually.

To configure an associated source qualifier:

1. Double-click the MQ Source Qualifier.
2. Click the Associated Source Qualifier tab and select Use Associated Source Qualifier.
3. From the list of associated source qualifiers, select the associated source qualifier you want to use. Click OK.

   The Designer creates a MsgId port in the associated source qualifier, and links the MsgId ports on the MQ Source Qualifier and the associated source qualifier.

CHAPTER 5

Working with WebSphere MQ Targets

This chapter includes the following topics:

Working with WebSphere MQ Targets Overview

The PowerCenter Integration Service writes target data to a WebSphere MQ message queue. It can write data to one message queue or multiple message queues. The PowerCenter Integration Service writes data to the message header fields and the message data field in the message. Depending on the type of data that you want to write to the message queue, choose a dynamic or static target definition.

Dynamic MQSeries Target Definitions

Use a dynamic MQSeries target definition in a mapping when you want to write data dynamically to the message header fields in the WebSphere MQ target. When you use a dynamic MQSeries target definition in a mapping, you can write binary or XML data to the message data field in the message. If the message data is flat file, use a static WebSphere MQ target definition.

The PowerCenter Integration Service can maintain transactional consistency when writing binary and XML data to WebSphere MQ targets. The PowerCenter Integration Service commits messages to WebSphere MQ targets in groups for each transaction you define based on the commit interval. If the session fails or aborts during the transaction, the PowerCenter Integration Service does not commit any data to the targets for that transaction.

Static WebSphere MQ Target Definitions

Use a static WebSphere MQ target definition in a mapping when you only want to write message data to the WebSphere MQ target. When you use a static WebSphere MQ target definition, you can pass data in flat file format to the WebSphere MQ message data field. The message header fields in the target remain static. No data passes to the message header fields from the pipeline.

When you use a static WebSphere MQ target, use a flat file target definition for the message data and configure the session to write messages to a message queue. The PowerCenter Integration Service does not maintain transactional consistency.

Working with a Dynamic MQSeries Target Definition

Use a dynamic MQSeries target definition when you write data to WebSphere MQ message header fields from the pipeline. The MQSeries target definition contains all the message header fields and the message data field of a WebSphere MQ message.

The MsgId field in the dynamic MQSeries target definition is a primary key field. Since MsgId is a unique identifier for a particular message, values you project to the MsgId field must be unique. If you project a MsgId value to a queue where that MsgId already exists, WebSphere MQ can ignore the value you pass to the target or use a generated value for the message in its place.

Certain message headers in a WebSphere MQ message require a set of values that the WebSphere MQ server assigns. When you write data to WebSphere MQ, the WebSphere MQ server assigns values to these fields.

The following table describes the message header fields to which WebSphere MQ assigns values when the PowerCenter Integration Service writes data to WebSphere MQ:

Message Header Field	Value
UserIdentifier	Defined by the environment. If the WebSphere MQ server cannot determine this value, the value for the field is null.
AccountingToken	Defined by the environment. If the WebSphere MQ server cannot determine this value, the value for the field is MQACT_NONE.
ApplIdentityData	Value for ApplIdentityData is null.
PutApplType	Defined by the environment.
PutAppName	Defined by the environment. If the WebSphere MQ server cannot determine this value, the value for the field is null.
PutDate	Date when the message arrives in the queue.
PutTime	Time when the message arrives in the queue.
ApplOriginData	Value for ApplOriginData is null.

To write data to the GroupId field in the target, project a value to the MsgFlags field in the target. Otherwise, the PowerCenter Integration Service writes a null value to the GroupId field. For example, you can set a flag to determine the group ID by the last message in the group.

The datatype for the message data field in the MQSeries target definition is Binary. When you use a dynamic MQSeries target definition in a mapping, the PowerCenter Integration Service can only write binary and XML data to the message data field in the target message.

Maintaining Transactional Consistency

When you use dynamic MQSeries target definitions in a mapping, the PowerCenter Integration Service can maintain transactional consistency when writing data to message queues. When the PowerCenter Integration Service maintains transactional consistency, it commits messages to the target after it writes all messages in a transaction group to the target. A transaction group consists of all messages that the PowerCenter Integration Service commits when it reaches a commit point. If a session aborts or fails during a transaction, the PowerCenter Integration Service rolls back all messages in the transaction group from the target.

When you enable recovery and a session aborts or fails, you can restart the session in recovery mode. In a recovery session, the PowerCenter Integration Service writes the messages to the message queues. This is because the PowerCenter Integration Service could not successfully commit all messages in the group to the targets. It can then process the messages to the targets in the next session.

To ensure transactional consistency, all dynamic MQSeries target definitions in the same pipeline must belong to the same target connection group. MQSeries target definitions that are in the same target connection group receive data from the same transactional source. Additionally, two or more dynamic MQSeries targets are in the same target connection group if they have the same value for the Queue Manager property.

Related Topics:

- "Configuring Transactional Consistency for WebSphere MQ Targets" on page 44

Writing XML Message Data

You can use the XML Generator transformation to write XML data from one or more sources and generate a single XML document for a WebSphere MQ target. The XML Generator transformation speeds processing by generating XML documents with the transformation in the pipeline.

When you create an XML Generator transformation, use the XML Wizard and XML Editor to define the XML groups.

Using On Commit

When you write XML message data to WebSphere MQ, you can use the On Commit property in the XML Generator tab to specify how the PowerCenter Integration Service writes the XML data.

You can select the following values for the On Commit property:

- **Ignore Commit.** Select to write one complete XML document to one WebSphere MQ message. When you select Ignore Commit, the PowerCenter Integration Service creates one XML document and writes all of the content for a document to a WebSphere MQ message at the end of the session.
- **Create New Document.** Select to generate multiple, complete XML documents and write each one to a separate WebSphere MQ message. When you select Create New Document, the PowerCenter Integration Service generates multiple XML documents based on the commit interval that you set.

When you select Create New Document, select Source as the commit type and set a commit interval in the session properties. The PowerCenter Integration Service commits data to the target based on the number of records in an active source. When you set a commit interval, the PowerCenter Integration Service commits data to the target based on the number of records you specify.

The commit interval also specifies the number of second-level nodes in the XML document that you write to a WebSphere MQ message. The number of second-level nodes corresponds to the number of records in the source data. The commit interval you set determines how many second-level nodes the PowerCenter Integration Service writes to a WebSphere MQ message.

For example, if the source data contains 12 records, and you set the commit interval to 4, the PowerCenter Integration Service creates three separate XML documents. Each document contains four second-level nodes to three WebSphere MQ messages. If you select Create New Document, each of the three XML documents are complete XML documents.

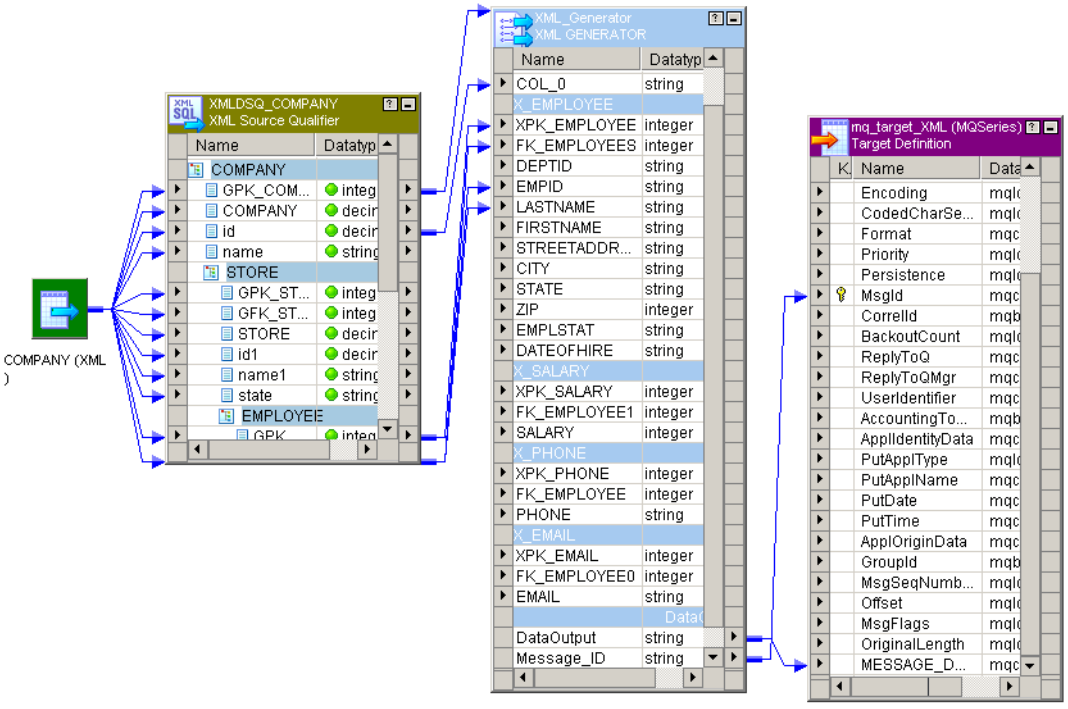
When you configure a session to create a single XML document on commit to WebSphere MQ, the PowerCenter Integration Service generates an extra XML document at the end of the session. It writes the XML document to a WebSphere MQ message.

Creating Pass-Through Ports

Because the MQ message ID is usually not part of the XML data, you can create pass-through ports to pass the MQ message ID data from the source to the target. You can use message ID data to correlate input and output messages for requests and replies.

MQ Message ID data passes from the Source Qualifier to the reference input port and then passes through the corresponding new output port to the target.

The following figure shows a mapping with a Message_ID port added as a pass-through port in the XML Generator transformation:



This example shows the ID data in the Source Qualifier linked to the COL0 reference input port in the XML Generator transformation. The XML Generator transformation passes the data through the Message_ID pass-through port to the MsgId port in the target.

Working with a Static WebSphere MQ Target Definition

Create a static WebSphere MQ target definition when you write flat file data to a WebSphere MQ message data field. The PowerCenter Integration Service writes data to the message data field in the message from the pipeline. It does not write any data to the message header fields in the message. Because the data is flat file, create a flat file target definition as the static WebSphere MQ target definition.

If you configure WebSphere MQ sessions for MQ targets with file writer connection and Target Based Commit, the number of messages committed to MQ targets and number of rows per message might vary.

If you want to write COBOL data to a message queue, use a VSAM target definition as the static WebSphere MQ target definition.

The message header fields in the MQ Source Qualifier are not projected to any port in the target definition. Since you are not writing data to the message header fields, the message header fields in the WebSphere MQ target remain static. The target definition only receives data from the Normalizer Source Qualifier.

Writing Message Header Data WebSphere MQ

When you want to write message header data to WebSphere MQ, you can create a mapping with the following transformations:

- An MQSeries source definition and an MQ Source Qualifier
- An associated flat file source definition and a Source Qualifier
- Two flat file target definitions

The PowerCenter Integration Service reads message header data from the source and stores it in a cache file. During a second pass, the PowerCenter Integration Service matches the flat file data with the message header data in the cache file based on the MsgId.

Note: If the source queue is large, the cache file will contain large amounts of data. This might slow the performance of the session.

Multiple Static WebSphere MQ Target Definitions

With PowerExchange for WebSphere MQ, you can write flat file message data to multiple WebSphere MQ message queues from a single mapping. When you write message data to multiple message queues, you must use a static WebSphere MQ target definition for each target queue to which the PowerCenter Integration Service writes data.

Creating Static WebSphere MQ Target Definitions

The target definition you use for a static WebSphere MQ target is specific to the format of the message data that you want to write to WebSphere MQ. You can define the target for the flat file message data in the following ways:

- Create the target definition by dragging the source definition of the message data into the Target Designer.
- Import the target definition.

Creating a Dynamic MQSeries Target Definition

Since all messages in the WebSphere MQ message queue have the same structure, you can manually create a MQSeries target definition. Or, you can create a target definition based on an MQSeries source definition in the Designer. Similar to the MQSeries source definition, you do not connect to a message queue to import the metadata for the target message queue.

To create an MQSeries target definition:

1. In the Target Designer, click Target > Create.
2. On the Create Target dialog box, enter a name for the target definition, and select MQSeries as the Database type.
3. Click Done.

Troubleshooting WebSphere MQ Targets

The Designer crashes when I import a WebSphere MQ mapping from an XML object, and the system does not have enough memory.

In the Designer, clear the Save MX Data option and import a WebSphere MQ mapping from an XML object. Before you import the mapping, clear the Save MX Data option in the Designer.

CHAPTER 6

Creating and Configuring WebSphere MQ Workflows

This chapter includes the following topics:

Creating and Configuring WebSphere MQ Workflows Overview

When you configure a WebSphere MQ session, you define properties that determine how the PowerCenter Integration Service reads messages from the source queue.

Working with Lookups

You can use Lookup transformations in an WebSphere MQ mapping, but you cannot perform lookups on a WebSphere MQ source.

Configuring a Session with a WebSphere MQ Mapping

When you configure a session with WebSphere MQ sources or targets, set the following session properties:

- Set the connection type and value for WebSphere MQ sources and targets.
- Define session properties for associated source qualifiers for WebSphere MQ sources.
- Define session properties for static WebSphere MQ targets.

You can also set the following session properties:

- Commit type and interval

- Filter conditions
- Message recovery
- Millisecond flush latency
- Pipeline partitioning

To configure session properties for a WebSphere MQ session:

1. In the Task Developer, double-click a WebSphere MQ session to open the session properties.
2. From the General Options on the Properties tab, optionally edit the commit type.

   Select Source as the Commit Type in the following cases:

   - To specify how the PowerCenter Integration Service writes XML data to WebSphere MQ targets.
   - To run the session in real time using the FlushLatency(n) function in a filter condition.
   - To configure transactional consistency for MQ Series targets.
3. Optionally, edit the commit interval.

   To specify how the PowerCenter Integration Service writes XML data to WebSphere MQ targets, and you set Source as the Commit Type, you must set a Commit Interval value.
4. Select a recovery strategy.

   To enable message recovery or if you are configuring the session for resilience, select Resume from Last Checkpoint.

   If you enable message recovery, you can configure a value for the recovery cache folder from the Properties settings of the Mapping tab (Sources node). Or, use the default cache folder $PMCacheDir\.
5. Click the Mapping tab.
6. From the Connections settings on the Mapping tab, select a connection type for each source definition.

   Select **None** to connect to the associated source definitions, such as VSAM or flat file.

   Select **Queue** to connect to the MQ Source Qualifiers.
7. In the Value pane, select a connection value for the MQ Source Qualifier.

   If you want the PowerCenter Integration Service to try to reconnect to the WebSphere MQ queue if the connection fails, select a Message Queue connection object that has a positive value for the Retry Connection Period property.
8. From the Properties settings on the Mapping tab (Sources node), verify the following settings for associated sources:

   - **Source File Directory**. Use the default value.
   - **Source Filename**. Use the default value.
   - **Source Filetype**. Set to Direct.
9. Click the Targets node.
10. From the Connections settings on the Targets node, select a connection type for each MQSeries target definition:

    - For static MQSeries target definitions, set the connection type to Queue.
    - For dynamic MQSeries target definitions, the connection type is set to Queue.

      To specify how the PowerCenter Integration Service writes XML data to WebSphere MQ targets, select Queue as the connection type for XML target definitions in the mapping. If you want to configure the session for transactional consistency, verify that all WebSphere MQ targets in a single pipeline belong to the same target connection group.
11. In the Value pane, select a connection value for each target definition.

If you configure the session for transactional consistency, verify that all dynamic WebSphere MQ targets in a single pipeline belong to the same target connection group.

12. From the Properties settings, optionally select Destructive Read to remove messages from a queue when you enable message recovery and specify the recovery cache folder.

    If you select Destructive Read, the mapping cannot contain an associated source qualifier. If the mapping contains an associated source qualifier and you select Destructive Read, the mapping becomes invalid. To remove messages from a queue for a mapping that contains an associated source qualifier, enter the RemoveMsg(TRUE) function in a filter condition.
13. Optionally, select Millisecond Flush Latency to process flush latency in milliseconds.
14. Optionally, click the right corner of the Filter option to enter a filter condition.
15. Enter the filter conditions. Click OK.

    When you enter a value for the MsgCount(*n*), Idle(*n*), FlushLatency(*n*), or ForcedEOQ(*n*) functions, enter 0 or a positive value. Otherwise, the PowerCenter Integration Service ignores the filter condition.
16. Enter a recovery cache folder if you selected Destructive Read.
17. From the Properties settings on the Mapping tab (Targets node), verify the following settings:
    - **Output File Directory**. Use the default value.
    - **Output Filename**. Use the default value.
18. Click OK.

Working with WebSphere MQ Sessions

You can configure the following session properties for WebSphere MQ sources and targets:

- **Filter conditions.** Set filter conditions.
- **Real-time data extraction.** Configure the session to extract data in real time.
- **Source-based commits.** Configure source-based commit for a real-time session.
- **Message recovery.** Enable message recovery.
- **Destructive read.** Configure the session to remove messages from the source queue after the PowerCenter Integration Service puts the messages in the recovery cache folder.
- **Resilience.** If you want the PowerCenter Integration Service to try to reconnect to the WebSphere MQ queue if the connection fails, select a Message Queue connection object with retry connection value.
- **Transactional consistency.** Configure the session to maintain transactional consistency.
- **Pipeline partitioning.** Set partitions in a pipeline.

Entering a Filter Condition in the Session Properties

When you configure a WebSphere MQ session, you can enter filter conditions for the session in the session properties. When you enter a filter condition, the PowerCenter Integration Service only reads messages from the WebSphere MQ source that fulfill the filter condition. The filter conditions you create in the session properties override the filter conditions in the MQ Source Qualifier.

You can enter the following filter conditions in the session properties:

- Message header ports

- Functions
- Mapping parameters
- Mapping variables

Extracting Data in Real Time

You can configure flush latency to process data in real time. A real-time session reads, processes, and writes data to targets continuously. Flush latency determines how often the PowerCenter Integration Service flushes data from the source.

To read data in real time, use one of the following session configurations:

- Configure the session with the FlushLatency(*n*) function in a filter condition and source-based commit as the commit type.
- Configure the session with the ForcedEOQ(*n*) function in a filter condition.

Configuring the Session with the FlushLatency(n) Function

You can configure a session with flush latency to read data in real time. Enter the FlushLatency(*n*) function in a filter condition in the session properties or MQ Source Qualifier.

If you select the Destructive Read option or enter the RemoveMsg(TRUE) function in a filter condition for a real-time session, the PowerCenter Integration Service removes messages from the source queue when it reaches Flush Latency interval.

Related Topics:

- "Controlling Data Extraction in Real Time" on page 27

Configuring the Session with ForcedEOQ(n)

You can configure a session with ForcedEOQ(*n*) when you want to run a session in real time. Use ForcedEOQ(*n*) to run a session in real time when the mapping configuration or session properties do not support FlushLatency(*n*). When you use ForcedEOQ(*n*), the PowerCenter Integration Service stops reading messages from the source queue at the end of the specified interval.

To configure ForcedEOQ(*n*), enter the filter condition in the session properties. To run a session in real time with ForcedEOQ(*n*), configure the workflow that contains the session to run continuously.

Related Topics:

- "Controlling Forced End of Queue" on page 27

Configuring Source-Based Commits

The PowerCenter Integration Service commits data based on the number of source rows. The commit point is the commit interval you configure in the session properties. The PowerCenter Integration Service commits data to the target based on the number of rows from some active sources in a target load order group. These rows are referred to as source rows.

The PowerCenter Integration Service commits messages based on the commit interval and the flush latency interval. For example, you use five seconds as the flush latency interval and you set the source-based commit interval to 1,000 messages. The PowerCenter Integration Service sends messages to the target after receiving 1,000 messages from the source and after each five second flush latency interval.

The PowerCenter Integration Service writes a message to the session log stating that source-based commit is disabled for the session in the following situations:

- You run a session with an MQ source and an XML associated source definition with more than one group in the mapping.
- You use a source-based commit and configure the filter condition to use the FlushLatency(n) function.

Message Recovery

When you configure message recovery for a real-time session, the PowerCenter Integration Service can recover unprocessed messages from a failed session. When you enable message recovery for a real-time session, the PowerCenter Integration Service stores source messages or message IDs in a recovery file, recovery table, or recovery queue. If the session fails, run the session in recovery mode to recover the messages that the PowerCenter Integration Service did not process.

To configure message recovery for a session that writes to a queue target, create a recovery queue. Create the recovery queue under the same queue manager as the message queue so the commit scope is the same. Configure the Recovery Queue Name in the WebSphere MQ connection properties.

The session fails if the recovery queue name does not match a recovery queue name in the WebSphere MQ system.

Configuring Destructive Read

Configure destructive read to remove messages from the source queue after the PowerCenter Integration Service puts the messages in the recovery cache folder.

Related Topics:

- "Controlling Queue Clean-up" on page 28

Configuring Resilience

You can configure a WebSphere MQ reader session to make multiple attempts to connect or reconnect to WebSphere MQ. If a connection attempt fails because of a network failure or if the WebSphere MQ queue manager is not running, the PowerCenter Integration Service tries to connect to the queue for a specified period of time before the session fails.

Note: You cannot configure connection resiliency for WebSphere MQ targets.

You can configure the period of time you want the PowerCenter Integration Service to attempt to connect to WebSphere MQ in a Message Queue connection. Set the Connection Retry Period property in the Message Queue connection.

When you configure resilience, also configure the following session properties:

- **Recovery Strategy.** Set the Recovery Strategy to Resume from Last Checkpoint.
- **Destructive Read.** Select the Destructive Read option in the session properties.

Note: When you set the Connection Retry Period greater than zero, the PowerCenter Integration Service is resilient to the initial connection to the WebSphere MQ queue manager. The initial connection is resilient even if you do not configure the session for recovery or destructive reads.

When you run a session configured for resilience, and the PowerCenter Integration Service loses the connection to the WebSphere MQ queue, the following filter conditions continue to increment while the PowerCenter Integration Service attempts to reconnect to the queue:

- ForcedEOQ(*n*)
- FlushLatency(*n*)
- Idle(*n*)
- MsgCount(*n*)

For example, you configure a session with the Idle(30) filter condition. The PowerCenter Integration Service is idle for 10 seconds, and it loses the connection to the WebSphere MQ queue. While the PowerCenter Integration Service attempts to reconnect to the queue, the idle value continues to increase. After 15 seconds, the PowerCenter Integration Service reconnects to the queue. The PowerCenter Integration Service has been idle for 25 seconds. If it remains idle for five more seconds, the session ends.

Configuring Transactional Consistency for WebSphere MQ Targets

The PowerCenter Integration Service can maintain transactional consistency for WebSphere MQ sessions with dynamic targets. With transactional consistency, the PowerCenter Integration Service commits messages to dynamic targets in transaction groups. If the session aborts or fails during a transaction, the PowerCenter Integration Service rolls back all messages in the group from the targets.

Use the following guidelines when you configure transactional consistency:

- Select an application connection with the same connection properties for each dynamic MQSeries target in the pipeline. This ensures that all targets in a single pipeline belong to the same target connection group.
- Configure source-based commits for the session.
- Set a commit interval to define the commit point for a transactional group.

Related Topics:

- "Maintaining Transactional Consistency" on page 35

Pipeline Partitioning

You can increase the number of partitions in a pipeline to improve session performance. When you increase the number of partitions, the PowerCenter Integration Service can create multiple connections to sources and process partitions of sources and target data concurrently.

Use the following rules and guidelines when you configure partitioning for WebSphere MQ sessions:

- **Performance improvement.** Performance does not scale beyond 3 partitions.
- **Associated source qualifier.** You can specify multiple partitions if there is no associated source qualifier in the pipeline.
- **MQ Source Qualifier.** You can only specify the pass-through partition type when you create partition points.
- **WebSphere MQ targets.** You can specify pass-through, key range, hash keys, and round-robin partition types when you create partition points. You cannot merge output files from sessions with multiple partitions if you use a WebSphere MQ message queue as the target connection type.
- **Real-time sessions.** The PowerCenter Integration Service may not read messages in the same order as they exist in the source. If you require the delivery of messages in sequence, do not create multiple partitions.

Specifying Partitions and a Recovery Cache Folder

When you specify partitions for a WebSphere MQ session, and you configure the Recovery Cache Folder attribute in the session properties, enter a cache folder on a different device for each reader partition in the pipeline.

Configuring High Availability

You can configure PowerExchange for WebSphere MQ for high availability. The PowerCenter Integration Service can make multiple attempts to connect to a WebSphere MQ queue to read WebSphere MQ messages. If the PowerCenter Integration Service cannot connect to the WebSphere MQ queue, it tries to connect for a specified period of time. You need to configure resilience in the session properties. You also need to configure the queue connection retry period.

Scheduling Workflows

Before you run a session in a workflow, configure and schedule the workflow. You can schedule a workflow to run continuously, run at a given time or interval, or you can manually start a workflow. The PowerCenter Integration Service runs scheduled workflows through the duration of the schedule, unless the workflow fails.

To run a continuous workflow, select Run continuously when you edit the scheduler for the workflow. A continuous workflow starts as soon as the PowerCenter Integration Service initializes. When the workflow stops, it restarts immediately.

Continuous Workflows and Real-time Sessions

If you schedule a real-time session to run as a continuous workflow, the PowerCenter Integration Service starts the next run of the workflow as soon as it finishes the first. If you want to run a real-time session continuously, use the Idle(*n*) function in a filter condition with the FlushLatency(*n*) function. The session stops when it reaches the idle time period and restarts immediately. For example, to run a session for one year, set the filter condition to Idle(3513600).

Running Workflows

Before you run a WebSphere MQ workflow, make sure the WebSphere MQ listener is running. This enables the PowerCenter Integration Service to connect to a queue manager to read and write WebSphere MQ messages.

To start the WebSphere MQ listener, enter the following command from the command line:

```
runmqlsr -m <queue_manager_name> -t TCP -p <port_number> &
```

The following table describes the parameters for the runmqlsr command:

Option	Argument	Description
-m	queue manager name	Required. Name of the queue manager you want to connect to.
-t	protocol type	Required. Enter TCP for the protocol type.
-p	port number	Optional. Port number of the queue manager. Default is 1414. If the queue manager uses 1414, you can omit this option.
&	-	Optional. Enter an ampersand character (&) to run the process in the background.

Optimizing WebSphere MQ

To optimize WebSphere MQ performance and improve throughput, configure the following tasks:

- **Tune the WebSphere MQ queue manager and queue manager logging parameters.**

  The following table describes the WebSphere MQ tuning parameters:

Parameter	Description
MAXUMSGS	Queue manager parameter that specifies the maximum uncommitted messages. Limits the number of messages that applications can put on or retrieve from queues within a synchronization point. **Note:** The session fails if the number of messages the PowerCenter Integration Service reads and writes from the message queue within a synchronization point exceeds this value.
LogFilePages	Log parameter that specifies the size of each primary and secondary log file.
LogBufferPages	Log parameter that specifies the amount of memory that is allocated to buffer records for log writes.

- **Use WebSphere MQ server libraries.** If the WebSphere MQ server and the PowerCenter Server are on the same machine, back up the libpmmqdrvc.a file. Rename the file to libpmmqdrvc.bak. Then, rename the libpmmqdrvs.a file to libpmmqdrvc.a. You can find the files in the following directory:

  ```
  <PowerCenter installation directory>/server/bin
  ```

Troubleshooting WebSphere MQ Workflows

I ran a WebSphere MQ session for which resilience failed.

If a WebSphere MQ session fails, and the PowerCenter Integration Service does not commit messages to the target or rollback messages from the target, the WebSphere MQ session cannot use resilience. Increase the commit interval and the flush latency interval.

APPENDIX A

Datatype Reference

This appendix includes the following topic:

WebSphere MQ and Transformation Datatypes

PowerCenter uses the following datatypes in WebSphere MQ mappings:

- **WebSphere MQ datatypes.** WebSphere MQ datatypes appear in MQSeries source and target definitions in a mapping.
- **Transformation datatypes.** Transformation datatypes are generic datatypes that PowerCenter uses during the transformation process. They appear in all transformations in a mapping.

When the PowerCenter Integration Service reads source data, it converts the native datatypes to the comparable transformation datatypes before transforming the data. When the PowerCenter Integration Service writes data to a target, it converts the transformation datatypes to the comparable native datatypes.

The following table lists the WebSphere MQ datatypes that PowerCenter supports and the corresponding transformation datatypes:

WebSphere MQ Datatype	Range	Transformation Datatype	Range
MQBYTE	1 to 104,857,600 bytes You can pass binary data from a source to a target, but you cannot perform transformations on binary data. PowerCenter does not support binary data for COBOL or flat file sources.	Binary	1 to 104,857,600 bytes
MQCHAR	1 to 104,857,600 characters Fixed-length or varying-length string.	String	1 to 104,857,600 characters
MQHEX	1 to 104,857,600 characters Fixed-length or varying-length string.	String	1 to 104,857,600 characters
MQLONG	Precision of 10 and scale of 0. Integer value.	Integer	Precision 10, scale 0

MQHEX Datatype Conversion

If a value for a field with the datatype MQHEX is shorter than 48 bytes, the PowerCenter Integration Service adds 0s to the data. If the value for the field is longer than 48 bytes, PowerCenter Integration Service truncates the data. If a value for a field with the datatype MQHEX is invalid, the PowerCenter Integration Service drops the row. If a field with the MQHEX datatype contains a decimal value, the data may be inconsistent.

The following examples show invalid values for a field with the MQHEX datatype:

```
"123"
"123W"
```

In the first example, the field contains an odd number of characters. There must be two characters per byte that represent the hexadecimal value of this byte. In the second example, the field contains a W that cannot be converted to a numeric value. As a result, the PowerCenter Integration Service drops both of these rows.

APPENDIX B

Code Pages for PowerExchange for WebSphere MQ

This appendix includes the following topics:

Code Pages for PowerCenter Integration Service Processes in WebSphere MQ Sessions

The following table lists the IBM codeset name, character coded set identifier (CCSID), and the ICU name, description, and ID for supported code pages for the PowerCenter Integration Service processes you can use in WebSphere MQ sessions:

IBM Codeset Name	CCSID	ICU Name	ICU Description	ICU ID
ISO8859-2	912	ISO-8859-2	ISO 8859-2 Eastern European	5
ISO-8859-3	913	ISO-8859-3	ISO 8859-3 Southeast European	6
ISO8859-5	915	ISO-8859-5	ISO 8859-5 Cyrillic	8
ISO8859-6	1089	ISO-8859-6	ISO 8859-6 Arabic	9
ISO8859-7	813	ISO-8859-7	ISO 8859-7 Greek	10
ISO8859-7@euro	4909	ISO-8859-7	ISO 8859-7 Greek	10
ISO8859-8	916	ISO-8859-8	ISO 8859-8 Hebrew	11
ISO8859-9	920	ISO-8859-9	ISO 8859-9 Latin 5 (Turkish)	12
JISeucJP	1350	MS932	MS Windows Japanese, Shift-JIS	2024
UTF-8	1208	UTF-8	UTF-8 encoding of Unicode	106
IBM-037	37	IBM037	IBM EBCDIC US English	2028

IBM Codeset Name	CCSID	ICU Name	ICU Description	ICU ID
IBM-273	273	IBM273	IBM EBCDIC German	2030
IBM-280	280	IBM280	IBM EBCDIC Italian	2035
IBM-285	285	IBM285	IBM EBCDIC UK English	2038
IBM-297	297	IBM297	IBM EBCDIC French	2040
IBM-500	500	IBM500	IBM EBCDIC International Latin-1	2044
IBM-874	874	MS874	MS-DOS Thai, superset of TIS 620	874
IBM-930	930	IBM930	IBM EBCDIC Japanese	930
IBM-935	935	IBM935	IBM EBCDIC Simplified Chinese	935
IBM-936	936	IBM936	MS Windows Simplified Chinese, superset of GB 2312-80, EUC encoding	936
IBM-937	937	IBM937	IBM EBCDIC Traditional Chinese	937
IBM-939	939	IBM939	IBM EBCDIC Japanese CP939	939
IBM-949	949	MS949	MS Windows Korean, superset of KS C 5601-1992	949
IBM-950	950	MS950	MS Windows Traditional Chinese, superset of Big 5	950
IBM-932	932	MS932	MS Windows Japanese, Shift-JIS	2024

Code Pages for WebSphere MQ Sources and Targets

The following table lists the IBM codeset name, character coded set identifier (CCSID), and the ICU name, description, and ID for supported code pages for WebSphere MQ sources and targets:

IBM Codeset Name	CCSID	ICU Name	ICU Description	ICU ID
IBM-277	277	IBM277	EBCDIC Denmark, Norway	10115
IBM-278	278	IBM278	EBCDIC Finland, Sweden	10116
IBM-284	284	IBM284	EBCDIC Spain, Latin America	10117
IBM-290	290	IBM2e90	EBCDIC Japanese Katakana SBCS	10118
IBM-367	367	IBM367	IBM367	10012
IBM-420	420	IBM420	EBCDIC Arabic	10119

IBM Codeset Name	CCSID	ICU Name	ICU Description	ICU ID
IBM-424	424	IBM424	EBCDIC Hebrew (updated with new sheqel, control characters)	10120
IBM-437	437	IBM437	PC United States	10035
IBM-803	803	IBM-803	EBCDIC Hebrew	10121
IBM-833	833	IBM833	IBM EBCDIC Korean CP833	833
IBM-834	834	IBM834	IBM EBCDIC Korean CP834	834
IBM-838	838	IBM-838	EBCDIC Thai	10122
IBM-850	850	cp850	PC Latin1	10036
IBM-852	852	IBM852	PC Latin2 (without euro update)	10038
IBM-855	855	IBM855	PC Cyrillic (without euro update)	10039
IBM-856	856	cp856	PC Hebrew (old)	10040
IBM-857	857	cp857	PC Latin5 (without euro update)	10041
IBM-858	858	cp858	PC Latin1 (with euro update)	10042
IBM-860	860	cp860	PC Portugal	10043
IBM-861	861	cp861	PC Iceland	10044
IBM-860	862	cp862	PC Hebrew (without euro update)	10045
IBM-863	863	cp863	PC Canadian French	10046
IBM-864	864	cp864	PC Arabic (without euro update)	10047
IBM-865	865	cp865	PC Nordic	10048
IBM-866	866	cp866	PC Russian (without euro update)	10049
IBM-867	867	IBM-867	PC Hebrew (with euro update)	10050
IBM-868	868	cp868	PC Urdu	10051
IBM-869	869	cp869	PC Greek (without euro update)	10052
IBM-870	870	IBM870	EBCDIC Latin2	10123
IBM-871	871	IBM871	EBCDIC Iceland	10124
IBM-875	875	IBM-875	EBCDIC Greek	10125
koi8-r	878	KOI8-R	IRussian Internet	10053
IBM-897	897	JIS_X0201	ISO-2022 encoding for Japanese (JIS_X0201)	10093

IBM Codeset Name	CCSID	ICU Name	ICU Description	ICU ID
IBM-918	918	IBM918	EBCDIC Urdu	10126
IBM-921	921	ISO-8859-13	ISO 8859-13 PC Baltic (without euro update)	10014
IBM-922	922	cp922	IPC Estonian (without euro update)	10056
IBM-933	933	IBM933	IBM EBCDIC Korean CP933	933
IBM-942	942	IBM-942	PC Japanese SJIS-78 syntax (IBM-942)	10015
IBM-943	943	IBM-943	PC Japanese SJIS-90 (IBM-943)	10016
eucKR	970	EUC-KR	EUC Korean	10029
eucCN	1383	GB2312	Chinese EUC	10024
IBM-4899	4899	IBM-4899	EBCDIC Hebrew (with euro)	10159
IBM-4971	4971	IBM-4971	EBCDIC Greek (with euro update)	10160
eucJP	5050	IBM-5050	Japanese EUC (Packed Format)	10018
IBM-5123	5123	IBM-5123	EBCDIC Japanese Latin (with euro update)	10164
IBM-5346	5346	MS1250	MS Windows Latin 2 (Central Europe)	2250
IBM-5347	5347	MS1251	MS Windows Cyrillic (Slavic)	2251
IBM-5348	5348	MS1252	MS Windows Latin1 (ANSI), superset of Latin1	2252
IBM-5349	5349	MS1253	MS Windows Greek	2253
IBM-5350	5350	MS1254	MS Windows Latin 5 (Turkish), superset of ISO 8859-9	2254
IBM-5351	5351	IBM-5351	MS Windows Hebrew (older version)	10061
IBM-5352	5352	IBM-5352	MS Windows Arabic (older version)	10063
IBM-5353	5353	IBM-5353	MS Windows Baltic (older version)	10065
IBM-5354	5354	MS1258	MS Windows Vietnamese	2258
GB18030	5488	gb18030	GB 18030 MBCS code page	1392
IBM-8482	8482	IBM-8482	EBCDIC Japanese Katakana SBCS (with euro update)	10165
IBM-9027	9027	IBM-1371	EBCDIC Taiwan Extended (SBCS IBM-1159 combined with DBCS IBM-9027)	10154

APPENDIX C

Glossary of Terms

channel

A one-way communication link between two queue managers. It can carry messages destined for any number of queues hosted by the remote queue manager or any number of target queue managers.

channel table file

A file that contains the details of all the client-connection channels defined at the WebSphere MQ Server.

client-connection channel

A connection channel used by the PowerCenter Integration Service to connect to multiple queue managers.

idle mode

A queue reading mode where the PowerCenter Integration Service waits for the queue to be idle before it stops reading from the queue.

message count mode

A queue reading mode where the PowerCenter Integration Service reads a specified number of messages from the queue and stops reading.

message data

The application data or the contents of the message body. The content and format of the message data is defined by the application that uses the message queue. The message data can be one or more rows of data in binary, flat file, COBOL, or XML format.

message header

A component of a message that contains data about the message on the queue. Message header data includes a message identification number, message format information, and other message descriptor data.

message queue

A storage space in memory or on disk that holds incoming transmissions until the computer can process them. Message queues provide an asynchronous communications protocol, meaning that the sender and receiver of the message do not need to connect to the message queue at the same time. Messages placed onto the queue are stored until the recipient deletes them.

messages

A collection of data that one program sends to another program.

queue clean-up

The process of removing messages from the source queue.

queue manager

A service that maintains queues and ensures that the messages in the queues reach their destination. When a successful connection is made, the queue manager issues a connection handle that is used to identify the connection in subsequent function calls.

server-connection channel

A channel that enables the WebSphere MQ server to receive client connections for the queue manager.

time slice mode

A queue reading mode where the PowerCenter Integration Service reads messages from the queue for a specified period of time.

transactional consistency

With transactional consistency, the PowerCenter Integration Service commits messages to dynamic MQSeries targets in transaction groups. If the session aborts or fails during a transaction, the PowerCenter Integration Service rolls back all messages in the group from the targets.

INDEX

T

U

V

W

X