# Table of Contents

- **Preface** .............................................. vii

  - Informatica Resources ................................ vii
  - Informatica Customer Portal ............................ vii
  - Informatica Documentation ............................... vii
  - Informatica Web Site ................................ vii
  - Informatica Knowledge Base ............................. vii
  - Informatica Global Customer Support .................. viii

- **Chapter 1: Understanding PowerExchange for WebSphere MQ** .............. 1
  - Overview ............................................. 1
  - WebSphere MQ Architecture ............................. 1
  - Queue Manager ...................................... 2
  - Message Queue ..................................... 2
  - WebSphere MQ Messages ................................ 2
  - WebSphere MQ Sources and Targets .................... 3
  - WebSphere MQ Sources ................................ 3
  - WebSphere MQ Targets ................................ 4
  - WebSphere MQ Source Qualifier Transformations ........ 4
  - MQ Source Qualifier Transformation .................... 4
  - Associated Source Qualifier Transformation .......... 4
  - Understanding Secure Sockets Layer ................. 4
  - Using Code Pages .................................... 5
  - Processing Unicode Data ............................... 5

- **Chapter 2: Configuring PowerExchange for WebSphere MQ** .................. 7
  - Overview ............................................. 7
  - Minimum System Requirements ......................... 7
  - Prerequisites ....................................... 7
  - Configuring PowerExchange for WebSphere MQ .......... 7
  - Step 1. Configure Channels for Queue Managers ........ 8
  - Step 2. Set Environment Variables ..................... 8
  - Setting Environment Variables on the Integration Service Machine 8
  - Setting System Environment Variables on AIX .......... 9
  - Step 3. Configure WebSphere MQ Messages for Unicode 10

- **Chapter 3: Working with WebSphere MQ Sources** ............................ 11
  - Overview ............................................. 11
  - MQSeries Source Definition ............................ 11
  - Associated Source Definition .......................... 11
  - Working with an MQSeries Source Definition .......... 12
  - Creating and Editing an MQSeries Source Definition 12
  - Message Header Fields ............................... 12
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>MQ Source Qualifier Transformation</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>Overview</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>Working with an MQ Source Qualifier</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>Working with an Associated Source Qualifier</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>Filtering Messages from the Source Queue</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>Controlling End of File</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>Controlling Data Extraction in Real Time</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Controlling Forced End of Queue</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Controlling Incremental Extraction</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>Controlling Queue Clean-up</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>Entering a Filter Condition in the MQ Source Qualifier</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>Filter Syntax</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>Syntax Rules</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>Filtering Messages Using Message Header Ports</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>Entering a Filter Condition</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>Configuring an MQ Source Qualifier</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>Creating and Configuring an Associated Source Qualifier</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>Working with WebSphere MQ Targets</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>Overview</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>Dynamic MQSeries Target Definitions</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>Static WebSphere MQ Target Definitions</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>Working with a Dynamic MQSeries Target Definition</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>Maintaining Transactional Consistency</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>Writing XML Message Data</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>Working with a Static WebSphere MQ Target Definition</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>Writing Message Header Data WebSphere MQ</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>Multiple Static WebSphere MQ Target Definitions</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>Creating Static WebSphere MQ Target Definitions</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>Creating a Dynamic MQSeries Target Definition</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>Editing an MQSeries Target Definition</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>Troubleshooting</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>Creating and Configuring WebSphere MQ Workflows</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>Overview</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>Working with Lookups</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>Configuring High Availability</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>Working with WebSphere MQ Sessions</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>Entering a Filter Condition in the Session Properties</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>Extracting Data in Real Time</td>
<td>34</td>
</tr>
</tbody>
</table>
Table of Contents

Configuring Source-Based Commits ..................................................... 35
Message Recovery ................................................................. 35
Configuring Destructive Read ...................................................... 35
Configuring Resilience .............................................................. 35
Configuring Transactional Consistency for WebSphere MQ Targets .......... 36
Pipeline Partitioning ................................................................. 36
Configuring a Session with a WebSphere MQ Mapping ......................... 37
Scheduling Workflows .............................................................. 38
Continuous Workflows and Real-time Sessions ............................... 39
Running Workflows ................................................................. 39
Optimizing WebSphere MQ ......................................................... 39
Troubleshooting ................................................................. 40

Appendix A: WebSphere MQ Datatype Reference ................................. 41
WebSphere MQ and Transformation Datatypes .................................. 41
MQHEX Datatype Conversion ..................................................... 41

Appendix B: Code Pages ............................................................... 43
Code Pages for Integration Service Processes in WebSphere MQ Sessions .... 43
Code Pages for WebSphere MQ Sources and Targets ....................... 44

Appendix C: Glossary of Terms ..................................................... 47

Index ................................................................. 49
Preface

The Informatica PowerExchange for WebSphere MQ User Guide provides information to build WebSphere MQ mappings, read data from message queues, and write data to message queues. It is written for the database administrators and developers who are responsible for reading data from message queues and writing data to data warehouses or message queues.

This book assumes you have knowledge of relational database concepts and the database engines, PowerCenter, and the WebSphere MQ system. You should also be familiar with the interface requirements for other supporting applications.

Informatica Resources

Informatica Customer Portal

As an Informatica customer, you can access the Informatica Customer Portal site at http://my.informatica.com. The site contains product information, user group information, newsletters, access to the Informatica customer support case management system (ATLAS), the Informatica Knowledge Base, Informatica Documentation Center, and access to the Informatica user community.

Informatica Documentation

The Informatica Documentation team takes every effort to create accurate, usable documentation. If you have questions, comments, or ideas about this documentation, contact the Informatica Documentation team through email at infa_documentation@informatica.com. We will use your feedback to improve our documentation. Let us know if we can contact you regarding your comments.

Informatica Web Site

You can access the Informatica corporate web site at http://www.informatica.com. The site contains information about Informatica, its background, upcoming events, and sales offices. You will also find product and partner information. The services area of the site includes important information about technical support, training and education, and implementation services.

Informatica Knowledge Base

As an Informatica customer, you can access the Informatica Knowledge Base at http://my.informatica.com. Use the Knowledge Base to search for documented solutions to known technical issues about Informatica products. You can also find answers to frequently asked questions, technical white papers, and technical tips.
Informatica Global Customer Support

There are many ways to access Informatica Global Customer Support. You can contact a Customer Support Center through telephone, email, or the WebSupport Service.

Use the following email addresses to contact Informatica Global Customer Support:
- support@informatica.com for technical inquiries
- support_admin@informatica.com for general customer service requests

WebSupport requires a user name and password. You can request a user name and password at http://my.informatica.com.

Use the following telephone numbers to contact Informatica Global Customer Support:

<table>
<thead>
<tr>
<th>North America / South America</th>
<th>Europe / Middle East / Africa</th>
<th>Asia / Australia</th>
</tr>
</thead>
<tbody>
<tr>
<td>Informatica Corporation</td>
<td>Informatica Software Ltd.</td>
<td>Informatica Business Solutions Pvt. Ltd.</td>
</tr>
<tr>
<td>Headquarters</td>
<td>6 Waltham Park</td>
<td>Diamond District</td>
</tr>
<tr>
<td>100 Cardinal Way</td>
<td>Waltham Road, White Waltham</td>
<td>Tower B, 3rd Floor</td>
</tr>
<tr>
<td>Redwood City, California</td>
<td>Maidenhead, Berkshire</td>
<td>150 Airport Road</td>
</tr>
<tr>
<td>94063</td>
<td>SL6 3TN</td>
<td>Bangalore 560 008</td>
</tr>
<tr>
<td>United States</td>
<td>United Kingdom</td>
<td>India</td>
</tr>
</tbody>
</table>

Toll Free
- +1 877 463 2435
- +55 11 3523 7761, Brazil
- +52 55 1168 9763, Mexico
- +1 650 385 5800, United States

Standard Rate
- +32 15 281 702, Belgium
- +33 1 41 38 92 26, France
- +49 1805 702 702, Germany
- +31 306 022 797, Netherlands
- +44 1628 511 445, United Kingdom
- +61 1800 151 830, Australia
- 001 800 4632 4357, Singapore
- +91 80 4112 5738, India
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:

<table>
<thead>
<tr>
<th>WebSphere MQ Message Header</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StrucId</td>
<td>String(4)</td>
<td>Structure identifier.</td>
</tr>
<tr>
<td>Version</td>
<td>Number</td>
<td>Structure version number.</td>
</tr>
<tr>
<td>Report</td>
<td>Number</td>
<td>Options for report messages.</td>
</tr>
<tr>
<td>MsgType</td>
<td>Number</td>
<td>Message type.</td>
</tr>
<tr>
<td>Expiry</td>
<td>Number</td>
<td>Message lifetime.</td>
</tr>
<tr>
<td>Feedback</td>
<td>Number</td>
<td>Feedback or reason code.</td>
</tr>
<tr>
<td>Encoding</td>
<td>Number</td>
<td>Data encoding.</td>
</tr>
<tr>
<td>CodedCharSetId</td>
<td>Number</td>
<td>Coded character set identifier.</td>
</tr>
<tr>
<td>Format</td>
<td>String(8)</td>
<td>Format name.</td>
</tr>
<tr>
<td>Priority</td>
<td>Number</td>
<td>Message priority.</td>
</tr>
</tbody>
</table>
WebSphere MQ Sources and Targets

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.

<table>
<thead>
<tr>
<th>WebSphere MQ Message Header</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence</td>
<td>Number</td>
<td>Message persistence.</td>
</tr>
<tr>
<td>MsgId</td>
<td>Binary(24)</td>
<td>Message identifier.</td>
</tr>
<tr>
<td>CorrId</td>
<td>Binary(24)</td>
<td>Correlation identifier.</td>
</tr>
<tr>
<td>BackoutCount</td>
<td>Number</td>
<td>Backout counter.</td>
</tr>
<tr>
<td>ReplytoQ</td>
<td>String(48)</td>
<td>Name of reply queue.</td>
</tr>
<tr>
<td>ReplytoQMgr</td>
<td>String(48)</td>
<td>Name of reply queue manager.</td>
</tr>
<tr>
<td>UserIdentifier</td>
<td>String(12)</td>
<td>User identifier.</td>
</tr>
<tr>
<td>AccountingToken</td>
<td>Binary(32)</td>
<td>Accounting token.</td>
</tr>
<tr>
<td>ApplIdentityData</td>
<td>String(32)</td>
<td>Application data relating to identity.</td>
</tr>
<tr>
<td>PutAppType</td>
<td>Number</td>
<td>Type of application that put the message on queue.</td>
</tr>
<tr>
<td>PutAppName</td>
<td>String(28)</td>
<td>Name of application that put the message on queue.</td>
</tr>
<tr>
<td>PutDate</td>
<td>String(8)</td>
<td>Date when message was put on queue.</td>
</tr>
<tr>
<td>PutTime</td>
<td>String(8)</td>
<td>Time when message was put on queue.</td>
</tr>
<tr>
<td>ApplOrigData</td>
<td>String(4)</td>
<td>Application data relating to origin.</td>
</tr>
<tr>
<td>GroupId</td>
<td>Binary(24)</td>
<td>Group identifier.</td>
</tr>
<tr>
<td>MsgSeqNumber</td>
<td>Number</td>
<td>Sequence number of logical messages within group.</td>
</tr>
<tr>
<td>Offset</td>
<td>Number</td>
<td>Offset of data in physical message from start of logical message.</td>
</tr>
<tr>
<td>MsgFlags</td>
<td>Number</td>
<td>Message flags.</td>
</tr>
<tr>
<td>OriginalLength</td>
<td>Number</td>
<td>Length of original message.</td>
</tr>
</tbody>
</table>

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 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 Integration Service can read message data in the following formats:
WebSphere MQ Targets

The 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 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 8
Using Code Pages

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

When the Integration Service writes data to dynamic WebSphere MQ targets that contain a different code page than the Integration Service, the 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 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 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 Integration Service uses the default code page of the WebSphere MQ system.

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

- The 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 Integration Service cannot use the CCSID of the queue manager.
Overview

This chapter provides information about configuring PowerExchange for WebSphere MQ.

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 and configure WebSphere MQ client and server. Install a WebSphere MQ client or the WebSphere MQ server on the machine running the Integration Service.

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

PowerCenter uses client connections to connect to WebSphere MQ queue managers. Create a client-connection channel if the WebSphere MQ system does not already have one.

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.

To create a client-connection channel:

1. From the command prompt on the machine running the 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:

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

   A message appears stating the channel was successfully created.

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

Step 2. Set Environment Variables

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

Setting Environment Variables on the Integration Service Machine

Set system environment variables before starting the Integration Service process. You can set the following system environment variables:

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MQCHLTAB</td>
<td>Name of the file that contains the client channel definition table.</td>
</tr>
<tr>
<td>MQCHLLIB</td>
<td>Directory path to the file that contains the client channel definition table.</td>
</tr>
</tbody>
</table>
Step 2. Set Environment Variables

Use the following guidelines:

- Set MQCHLTAB and MQCHLLIB for the 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 manager 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 Integration Service process.
2. Set the following values for the environment variables on the machine running the Integration Service process:

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MQCHLTAB</td>
<td>MQCHLTAB = AMQCLCHL.TAB</td>
</tr>
<tr>
<td>MQCHLLIB</td>
<td>MQCHLLIB = &lt;WebSphereMQ installation directory&gt;/Qmgrs/&lt;queue manager name&gt;/@ipcc</td>
</tr>
<tr>
<td>MQSSLKEYR</td>
<td>MQSSLKEYR = &lt;WebSphereMQ installation directory&gt;/Qmgrs/&lt;queue manager name&gt;/ssl/&lt;key repository&gt;</td>
</tr>
<tr>
<td>MQSERVER</td>
<td>MQSERVER = &lt;server-connection channel name&gt;/TCP/&lt;host name&gt;({port number})</td>
</tr>
<tr>
<td></td>
<td>For example: MQ_cseversonpc/TCP/cseversonpc(1414)</td>
</tr>
</tbody>
</table>

3. Start the Integration Service process.

Setting System Environment Variables on AIX

When you configure the 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 Integration Service.

To set system environment variables on AIX:

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

   In a C shell environment, type:
   ```bash
   setenv LDR_CNTRL MAXDATA=0x60000000
   ```

   In a Bourne shell environment, type:
   ```bash
   LDR_CNTRL = "MAXDATA=0x60000000"; export LDR_CNTRL
   ```

2. Stop the WebSphere MQ queue manager.
3. Set the IPCBaseAddress address parameter in the mqs.ini file to reallocate system memory for WebSphere MQ.

4. Enter 11 for the IPCBaseAddress parameter in the mqs.ini file to reallocate system memory for WebSphere MQ. For example, enter:
   ```
   IPCBaseAddress=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 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 Integration Service uses the WebSphere MQ client library, set the CCSID in the MQCCSID environment variable.
C H A P T E R  3

Working with WebSphere MQ Sources

This chapter includes the following topics:
♦ Overview, 11
♦ Working with an MQSeries Source Definition, 12
♦ Working with an Associated Source Definition, 14

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 Integration Service to only read messages from a particular application and messages of a particular format.

RELATED TOPICS:
“Message Header” on page 2

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 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 Integration Service treats the message data differently depending on the message data format. When the message data is binary, the 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 an XML Parser transformation in a mapping, the Integration Service reads data WebSphere MQ, transforms the message data, and then writes the output to a target.
Figure 3-1 displays a WebSphere MQ message that contains flat file message data:

![Figure 3-1. WebSphere MQ Message and Source Definitions](image)

The message header contains fields such as MessageId, CorrelId, and ReplyToQ, while the message data contains fields like Last_Name, First_Name, Address, City, and State. The MQSeries source definition includes metadata for these fields.

Figure 3-1 includes an MQSeries source definition and an associated source definition. The associated source definition contains the metadata for the message data field. Since the message data is flat file, the associated source definition is a flat file source definition.

### 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.

Figure 3-2 shows a mapping with an XML Parser transformation:

![Figure 3-2. Mapping with an XML Parser Transformation](image)

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

![Diagram](image)
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.

Figure 3-3 shows a mapping with a Message_ID port added as a pass-through port in the XML Parser transformation:

![Figure 3-3. XML Parser Transformation with Pass-Through Port](image)

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.

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.
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 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 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 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 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:

<table>
<thead>
<tr>
<th>Message Data Format</th>
<th>Associated Source Qualifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flat file</td>
<td>Source Qualifier transformation</td>
</tr>
<tr>
<td>COBOL</td>
<td>Normalizer transformation</td>
</tr>
</tbody>
</table>

When you add an associated source qualifier to a mapping, associate it with an MQ Source Qualifier. The association allows the 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 Integration Service to correlate the messages in the queue with their message data.

When you run a session, the 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 Integration Service reads from the queue.
- **Idle(n).** Controls the duration of time the Integration Service remains idle before it stops reading from the queue.
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 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 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 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 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 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 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 Integration Service waits 30 seconds after reading from the queue. If no new messages arrive on the queue within 30 seconds, the 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 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 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:

\[
\text{StartTime} \left(09/02/2000 \ 09:00:00\right) \ \&\& \ \text{EndTime} \left(09/02/2000 \ 17:30:00\right)
\]

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

\[
\text{StartTime} \left("\$$\$$SessStartTime\$$\"\right)
\]

### 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 Integration Service flushes data from the source. For more information about real-time processing, see the PowerCenter Workflow Administration Guide.

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

\[
\text{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 \text{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 34

### Controlling Forced End of Queue

Use \text{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 \text{ForcedEOQ}\(n\), the 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 \text{ForcedEOQ}\(n\):

\[
\text{ForcedEOQ}\(n\)
\]

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

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

For example, you want to run a real-time session with a 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 Integration Service cannot use \text{FlushLatency}\(n\) to commit messages to a target based on a specified maximum latency period.

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

**RELATED TOPICS:**

・“Extracting Data in Real Time” on page 34
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:

\[
\text{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 or for mappings from PowerCenter Connect for IBM MQSeries 7.0 or earlier.

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 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 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 Integration Service truncates the message while reading it from the WebSphere MQ queue, the session fails. The 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 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 35

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, or for mappings from PowerCenter Connect for IBM MQSeries 7.0 or earlier.

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 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 Integration Service reads from the message queue. When you enter a filter in the MQ Source Qualifier, the 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.

### Filter Syntax

The filter condition is an expression that returns TRUE or FALSE. For example, if you want the 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:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Textual Representation</th>
<th>Alternate Textual Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>EQUAL TO</td>
<td>=</td>
<td>==</td>
</tr>
<tr>
<td>NOT EQUAL TO</td>
<td>!=</td>
<td>&lt;&gt;</td>
</tr>
<tr>
<td>LESS THAN</td>
<td>&lt;</td>
<td></td>
</tr>
<tr>
<td>LESS THAN OR EQUAL TO</td>
<td>&lt;=</td>
<td></td>
</tr>
<tr>
<td>GREATER THAN</td>
<td>&gt;</td>
<td></td>
</tr>
<tr>
<td>GREATER THAN OR EQUAL TO</td>
<td>&gt;=</td>
<td></td>
</tr>
<tr>
<td>NOT</td>
<td>not</td>
<td>!</td>
</tr>
<tr>
<td>AND</td>
<td>and</td>
<td>&amp;&amp;</td>
</tr>
<tr>
<td>OR</td>
<td>or</td>
<td></td>
</tr>
<tr>
<td>LEFT PARENTHESES</td>
<td>(</td>
<td></td>
</tr>
<tr>
<td>RIGHT PARENTHESES</td>
<td>)</td>
<td></td>
</tr>
</tbody>
</table>
The following table shows the precedence of the logical operators:

<table>
<thead>
<tr>
<th>Precedence</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highest</td>
<td>Equals (==), Not Equal to (!=), Less than (&lt;), Less than or equal to (&lt;=), Greater than (&gt;), Greater than or equal to (&gt;=)</td>
</tr>
<tr>
<td></td>
<td>NOT</td>
</tr>
<tr>
<td></td>
<td>AND</td>
</tr>
<tr>
<td>Lowest</td>
<td>OR</td>
</tr>
</tbody>
</table>

**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:
   \[
   \text{<Port>} \ [\text{=}, \text{>=}, \text{<=}, \text{<}, \text{>}, \text{<>}] \text{'value'}
   \]

2. Use the following syntax in the filter condition if you create a filter with a function:
   \[
   \text{<Function>} \ (\text{'value'})
   \]

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:
   \[
   \text{PutDate} = "20000901" \ & \ \text{PutDate} = "20001001"
   \]
   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:
   \[
   \text{PutDate} = "20000901" \ & \ \text{PutTime} = "173000"
   \]

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:
   \[
   \text{ReplyToQ} = "PRODUCTION"
   \]

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**

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 any of the following properties for the MQ Source Qualifier:

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

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.
Overview

The Integration Service writes target data to a WebSphere MQ message queue. It can write data to one message queue or multiple message queues. The 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 Integration Service can maintain transactional consistency when writing binary and XML data to WebSphere MQ targets. The 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 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 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 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 Integration Service writes data to WebSphere MQ:

<table>
<thead>
<tr>
<th>Message Header Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserIdentifier</td>
<td>Defined by the environment. If the WebSphere MQ server cannot determine this value, the value for the field is null.</td>
</tr>
<tr>
<td>AccountingToken</td>
<td>Defined by the environment. If the WebSphere MQ server cannot determine this value, the value for the field is MQACT_NONE.</td>
</tr>
<tr>
<td>ApplIdentityData</td>
<td>Value for ApplIdentityData is null.</td>
</tr>
<tr>
<td>PutAppType</td>
<td>Defined by the environment.</td>
</tr>
<tr>
<td>PutAppName</td>
<td>Defined by the environment. If the WebSphere MQ server cannot determine this value, the value for the field is null.</td>
</tr>
<tr>
<td>PutDate</td>
<td>Date when the message arrives in the queue.</td>
</tr>
<tr>
<td>PutTime</td>
<td>Time when the message arrives in the queue.</td>
</tr>
<tr>
<td>ApplOriginData</td>
<td>Value for ApplOriginData is null.</td>
</tr>
</tbody>
</table>

To write data to the GroupId field in the target, project a value to the MsgFlags field in the target. Otherwise, the 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 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 Integration Service can maintain transactional consistency when writing data to message queues. When the 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 Integration Service commits when it reaches a commit point. If a session aborts or fails during a transaction, the 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 Integration Service writes the messages to the message queues. This is because the 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 36

### 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 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 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 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 Integration Service commits data to the target based on the number of records in an active source. When you set a commit interval, the 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 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 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 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:

![Mapping Diagram]

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 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 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 Integration Service reads message header data from the source and stores it in a cache file. During a second pass, the 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 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.

### Editing an MQSeries Target Definition

After creating the MQSeries target definition, you can edit the target definition properties.

**To edit an MQSeries target definition:**

1. In the Target Designer, double-click the title bar of the target definition.
2. On the Table tab, optionally edit the following settings:

<table>
<thead>
<tr>
<th>Target Settings</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Description of the message target.</td>
</tr>
<tr>
<td>Keywords</td>
<td>Add keywords to the target definition when you want to perform searches in the Repository Manager.</td>
</tr>
</tbody>
</table>
3. Click the Columns tab.
The Columns tab lists all the message header ports and the message data port with their datatype, precision, and scale.

4. Optionally, edit the datatype and precision of the columns in the target definition.

5. Click OK.

Troubleshooting

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.
Overview

When you configure a WebSphere MQ session, you define properties that determine how the 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 High Availability

You can configure PowerExchange for WebSphere MQ for high availability. The Integration Service can make multiple attempts to connect to a WebSphere MQ queue to read WebSphere MQ messages. If the 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.

Working with WebSphere MQ Sessions

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

- **Filter conditions.** Set filter conditions. For more information, see “Entering a Filter Condition in the Session Properties” on page 34.
- **Real-time data extraction.** Configure the session to extract data in real time. For more information, see “Extracting Data in Real Time” on page 34.

- **Source-based commits.** Configure source-based commit for a real-time session. For more information, see “Configuring Source-Based Commits” on page 35.

- **Message recovery.** Enable message recovery. For more information, see “Message Recovery” on page 35.

- **Destructive read.** Configure the session to remove messages from the source queue after the Integration Service puts the messages in the recovery cache folder. For more information, see “Configuring Destructive Read” on page 35.

- **Resilience.** If you want the 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. For more information see “Configuring Resilience” on page 35.

- **Transactional consistency.** Configure the session to maintain transactional consistency. For more information, see “Configuring Transactional Consistency for WebSphere MQ Targets” on page 36.

- **Pipeline partitioning.** Set partitions in a pipeline. For more information, see “Pipeline Partitioning” on page 36.

### 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 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 Integration Service flushes data from the source. For more information about real-time processing, see the *PowerCenter Workflow Administration Guide*.

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.
  -or-

- 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. For more information about flush latency, see the *PowerCenter Workflow Administration Guide*.

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

**RELATED TOPICS:**

- “Controlling Data Extraction in Real Time” on page 20
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 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 20

Configuring Source-Based Commits

The 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 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 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 Integration Service sends messages to the target after receiving 1,000 messages from the source and after each five second flush latency interval.

The 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 Integration Service can recover unprocessed messages from a failed session. When you enable message recovery for a real-time session, the 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 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. Enter the recovery queue name in the following custom property in the session configuration object, session properties, or Integration Service properties:

MQRecoveryQueueName=<recovery queue name>

Warning: 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 Integration Service puts the messages in the recovery cache folder.

RELATED TOPICS:
♦ “Controlling Queue Clean-up” on page 21

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 Integration Service tries to connect to the queue for a specified period of time before the session fails.

You can configure the period of time you want the 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.

When you run a session configured for resilience, and the Integration Service loses the connection to the WebSphere MQ queue, the following filter conditions continue to increment while the 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 Integration Service is idle for 10 seconds, and it loses the connection to the WebSphere MQ queue. While the Integration Service attempts to reconnect to the queue, the idle value continues to increase. After 15 seconds, the Integration Service reconnects to the queue. The 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 Integration Service can maintain transactional consistency for WebSphere MQ sessions with dynamic targets. With transactional consistency, the Integration Service commits messages to dynamic targets in transaction groups. If the session aborts or fails during a transaction, the 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 28

### Pipeline Partitioning

You can increase the number of partitions in a pipeline to improve session performance. When you increase the number of partitions, the 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:
- **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 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 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 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 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:
   ♦ For associated source definitions, such as VSAM or flat file, set the connection type to None.
   ♦ For MQ Source Qualifiers, the connection type is set to Queue.
7. In the Value pane, select a connection value for the MQ Source Qualifier.
If you want the 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 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.


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 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.

---

**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 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 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 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 \texttt{Idle(n)} function in a filter condition with the \texttt{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 \texttt{Idle(3513600)}.

**Running Workflows**

Before you run a WebSphere MQ workflow, make sure the WebSphere MQ listener is running. This enables the 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:

\begin{verbatim}
runmqlsr -m <queue_manager_name> -t TCP -p <port_number> &
\end{verbatim}

The following table describes the parameters for the \texttt{runmqlsr} command:

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

**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:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXUMSGS</td>
<td>Queue manager parameter that specifies the maximum uncommitted messages.</td>
</tr>
<tr>
<td></td>
<td>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 Integration Service reads and writes from the message queue within a synchronization point exceeds this value.</td>
</tr>
<tr>
<td>LogFilePages</td>
<td>Log parameter that specifies the size of each primary and secondary log file.</td>
</tr>
<tr>
<td>LogBufferPages</td>
<td>Log parameter that specifies the amount of memory that is allocated to buffer records for log writes.</td>
</tr>
</tbody>
</table>
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

I ran a WebSphere MQ session for which resilience failed.

If a WebSphere MQ session fails, and the 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

WebSphere MQ Datatype Reference

This appendix includes the following topic:
♦ WebSphere MQ and Transformation Datatypes, 41

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 Integration Service reads source data, it converts the native datatypes to the comparable transformation datatypes before transforming the data. When the Integration Service writes data to a target, it converts the transformation datatypes to the comparable native datatypes. For more information about datatypes, see the PowerCenter Designer Guide.

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

<table>
<thead>
<tr>
<th>WebSphere MQ Datatype</th>
<th>Range</th>
<th>Transformation Datatype</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>MQBYTE</td>
<td>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.</td>
<td>Binary</td>
<td>1 to 104,857,600 bytes</td>
</tr>
<tr>
<td>MQCHAR</td>
<td>1 to 104,857,600 characters Fixed-length or varying-length string.</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>MQHEX</td>
<td>1 to 104,857,600 characters Fixed-length or varying-length string.</td>
<td>String</td>
<td>1 to 104,857,600 characters</td>
</tr>
<tr>
<td>MQLONG</td>
<td>Precision of 10 and scale of 0. Integer value.</td>
<td>Integer</td>
<td>Precision 10, scale 0</td>
</tr>
</tbody>
</table>

MQHEX Datatype Conversion

If a value for a field with the datatype MQHEX is shorter than 48 bytes, the Integration Service adds 0s to the data. If the value for the field is longer than 48 bytes, Integration Service truncates the data. If a value for a field
with the datatype MQHEX is invalid, the 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 Integration Service drops both of these rows.
This appendix includes the following topics:

- Code Pages for Integration Service Processes in WebSphere MQ Sessions, 43
- Code Pages for WebSphere MQ Sources and Targets, 44

## Code Pages for 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 Integration Service processes you can use in WebSphere MQ sessions:

<table>
<thead>
<tr>
<th>IBM Codeset Name</th>
<th>CCSID</th>
<th>ICU Name</th>
<th>ICU Description</th>
<th>ICU ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISO-8859-3</td>
<td>913</td>
<td>ISO-8859-3</td>
<td>ISO 8859-3 Southeast European</td>
<td>6</td>
</tr>
<tr>
<td>ISO8859-5</td>
<td>915</td>
<td>ISO-8859-5</td>
<td>ISO 8859-5 Cyrillic</td>
<td>8</td>
</tr>
<tr>
<td>ISO8859-6</td>
<td>1089</td>
<td>ISO-8859-6</td>
<td>ISO 8859-6 Arabic</td>
<td>9</td>
</tr>
<tr>
<td>ISO8859-7</td>
<td>813</td>
<td>ISO-8859-7</td>
<td>ISO 8859-7 Greek</td>
<td>10</td>
</tr>
<tr>
<td>ISO8859-7@euro</td>
<td>4909</td>
<td>ISO-8859-7</td>
<td>ISO 8859-7 Greek</td>
<td>10</td>
</tr>
<tr>
<td>ISO8859-8</td>
<td>916</td>
<td>ISO-8859-8</td>
<td>ISO 8859-8 Hebrew</td>
<td>11</td>
</tr>
<tr>
<td>ISO8859-9</td>
<td>920</td>
<td>ISO-8859-9</td>
<td>ISO 8859-9 Latin 5 (Turkish)</td>
<td>12</td>
</tr>
<tr>
<td>JISx6937</td>
<td>1350</td>
<td>MS932</td>
<td>MS Windows Japanese, Shift-JIS</td>
<td>2024</td>
</tr>
<tr>
<td>UTF-8</td>
<td>1208</td>
<td>UTF-8</td>
<td>UTF-8 encoding of Unicode</td>
<td>106</td>
</tr>
<tr>
<td>IBM-037</td>
<td>37</td>
<td>IBM037</td>
<td>IBM EBCDIC US English</td>
<td>2028</td>
</tr>
<tr>
<td>IBM-273</td>
<td>273</td>
<td>IBM273</td>
<td>IBM EBCDIC German</td>
<td>2030</td>
</tr>
<tr>
<td>IBM-280</td>
<td>280</td>
<td>IBM280</td>
<td>IBM EBCDIC Italian</td>
<td>2035</td>
</tr>
<tr>
<td>IBM-285</td>
<td>285</td>
<td>IBM285</td>
<td>IBM EBCDIC UK English</td>
<td>2038</td>
</tr>
<tr>
<td>IBM-297</td>
<td>297</td>
<td>IBM297</td>
<td>IBM EBCDIC French</td>
<td>2040</td>
</tr>
<tr>
<td>IBM-500</td>
<td>500</td>
<td>IBM500</td>
<td>IBM EBCDIC International Latin-1</td>
<td>2044</td>
</tr>
<tr>
<td>IBM Codeset Name</td>
<td>CCSID</td>
<td>ICU Name</td>
<td>ICU Description</td>
<td>ICU ID</td>
</tr>
<tr>
<td>------------------</td>
<td>-------</td>
<td>----------</td>
<td>-----------------</td>
<td>-------</td>
</tr>
<tr>
<td>IBM-874</td>
<td>874</td>
<td>MS874</td>
<td>MS-DOS Thai, superset of TIS 620</td>
<td>874</td>
</tr>
<tr>
<td>IBM-930</td>
<td>930</td>
<td>IBM930</td>
<td>IBM EBCDIC Japanese</td>
<td>930</td>
</tr>
<tr>
<td>IBM-935</td>
<td>935</td>
<td>IBM935</td>
<td>IBM EBCDIC Simplified Chinese</td>
<td>935</td>
</tr>
<tr>
<td>IBM-936</td>
<td>936</td>
<td>IBM936</td>
<td>MS Windows Simplified Chinese, superset of GB 2312-80, EUC encoding</td>
<td>936</td>
</tr>
<tr>
<td>IBM-937</td>
<td>937</td>
<td>IBM937</td>
<td>IBM EBCDIC Traditional Chinese</td>
<td>937</td>
</tr>
<tr>
<td>IBM-939</td>
<td>939</td>
<td>IBM939</td>
<td>IBM EBCDIC Japanese CP939</td>
<td>939</td>
</tr>
<tr>
<td>IBM-949</td>
<td>949</td>
<td>MS949</td>
<td>MS Windows Korean, superset of KS C 5601-1992</td>
<td>949</td>
</tr>
<tr>
<td>IBM-950</td>
<td>950</td>
<td>MS950</td>
<td>MS Windows Traditional Chinese, superset of Big 5</td>
<td>950</td>
</tr>
<tr>
<td>IBM-932</td>
<td>932</td>
<td>MS932</td>
<td>MS Windows Japanese, Shift-JIS</td>
<td>2024</td>
</tr>
</tbody>
</table>

## 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:

<table>
<thead>
<tr>
<th>IBM Codeset Name</th>
<th>CCSID</th>
<th>ICU Name</th>
<th>ICU Description</th>
<th>ICU ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM-277</td>
<td>277</td>
<td>IBM277</td>
<td>EBCDIC Denmark, Norway</td>
<td>10115</td>
</tr>
<tr>
<td>IBM-278</td>
<td>278</td>
<td>IBM278</td>
<td>EBCDIC Finland, Sweden</td>
<td>10116</td>
</tr>
<tr>
<td>IBM-284</td>
<td>284</td>
<td>IBM284</td>
<td>EBCDIC Spain, Latin America</td>
<td>10117</td>
</tr>
<tr>
<td>IBM-290</td>
<td>290</td>
<td>IBM2e90</td>
<td>EBCDIC Japanese Katakana SBCS</td>
<td>10118</td>
</tr>
<tr>
<td>IBM-367</td>
<td>367</td>
<td>IBM367</td>
<td>IBM367</td>
<td>10012</td>
</tr>
<tr>
<td>IBM-420</td>
<td>420</td>
<td>IBM420</td>
<td>EBCDIC Arabic</td>
<td>10119</td>
</tr>
<tr>
<td>IBM-424</td>
<td>424</td>
<td>IBM424</td>
<td>EBCDIC Hebrew (updated with new sheqel, control characters)</td>
<td>10120</td>
</tr>
<tr>
<td>IBM-437</td>
<td>437</td>
<td>IBM437</td>
<td>PC United States</td>
<td>10035</td>
</tr>
<tr>
<td>IBM-803</td>
<td>803</td>
<td>IBM-803</td>
<td>EBCDIC Hebrew</td>
<td>10121</td>
</tr>
<tr>
<td>IBM-833</td>
<td>833</td>
<td>IBM833</td>
<td>IBM EBCDIC Korean CP833</td>
<td>833</td>
</tr>
<tr>
<td>IBM-834</td>
<td>834</td>
<td>IBM834</td>
<td>IBM EBCDIC Korean CP834</td>
<td>834</td>
</tr>
<tr>
<td>IBM-838</td>
<td>838</td>
<td>IBM-838</td>
<td>EBCDIC Thai</td>
<td>10122</td>
</tr>
<tr>
<td>IBM-850</td>
<td>850</td>
<td>cp850</td>
<td>PC Latin1</td>
<td>10036</td>
</tr>
<tr>
<td>IBM-852</td>
<td>852</td>
<td>IBM852</td>
<td>PC Latin2 (without euro update)</td>
<td>10038</td>
</tr>
<tr>
<td>IBM-855</td>
<td>855</td>
<td>IBM855</td>
<td>PC Cyrillic (without euro update)</td>
<td>10039</td>
</tr>
<tr>
<td>IBM-856</td>
<td>856</td>
<td>cp856</td>
<td>PC Hebrew (old)</td>
<td>10040</td>
</tr>
<tr>
<td>IBM-857</td>
<td>857</td>
<td>cp857</td>
<td>PC Latin5 (without euro update)</td>
<td>10041</td>
</tr>
<tr>
<td>IBM-858</td>
<td>858</td>
<td>cp858</td>
<td>PC Latin1 (with euro update)</td>
<td>10042</td>
</tr>
<tr>
<td>IBM-860</td>
<td>860</td>
<td>cp860</td>
<td>PC Portugal</td>
<td>10043</td>
</tr>
<tr>
<td>IBM Code Set Name</td>
<td>CCSID</td>
<td>ICU Name</td>
<td>ICU Description</td>
<td>ICU ID</td>
</tr>
<tr>
<td>------------------</td>
<td>-------</td>
<td>----------</td>
<td>----------------</td>
<td>-------</td>
</tr>
<tr>
<td>IBM-861</td>
<td>861</td>
<td>cp861</td>
<td>PC Iceland</td>
<td>10044</td>
</tr>
<tr>
<td>IBM-860</td>
<td>862</td>
<td>cp862</td>
<td>PC Hebrew (without euro update)</td>
<td>10045</td>
</tr>
<tr>
<td>IBM-863</td>
<td>863</td>
<td>cp863</td>
<td>PC Canadian French</td>
<td>10046</td>
</tr>
<tr>
<td>IBM-864</td>
<td>864</td>
<td>cp864</td>
<td>PC Arabic (without euro update)</td>
<td>10047</td>
</tr>
<tr>
<td>IBM-865</td>
<td>865</td>
<td>cp865</td>
<td>PC Nordic</td>
<td>10048</td>
</tr>
<tr>
<td>IBM-866</td>
<td>866</td>
<td>cp866</td>
<td>PC Russian (without euro update)</td>
<td>10049</td>
</tr>
<tr>
<td>IBM-867</td>
<td>867</td>
<td>IBM-867</td>
<td>PC Hebrew (with euro update)</td>
<td>10050</td>
</tr>
<tr>
<td>IBM-868</td>
<td>868</td>
<td>cp868</td>
<td>PC Urdu</td>
<td>10051</td>
</tr>
<tr>
<td>IBM-869</td>
<td>869</td>
<td>cp869</td>
<td>PC Greek (without euro update)</td>
<td>10052</td>
</tr>
<tr>
<td>IBM-870</td>
<td>870</td>
<td>IBM870</td>
<td>EBCDIC Latin2</td>
<td>10123</td>
</tr>
<tr>
<td>IBM-871</td>
<td>871</td>
<td>IBM871</td>
<td>EBCDIC Iceland</td>
<td>10124</td>
</tr>
<tr>
<td>IBM-875</td>
<td>875</td>
<td>IBM-875</td>
<td>EBCDIC Greek</td>
<td>10125</td>
</tr>
<tr>
<td>koi8-r</td>
<td>878</td>
<td>KOI8-R</td>
<td>iRussian Internet</td>
<td>10053</td>
</tr>
<tr>
<td>IBM-897</td>
<td>897</td>
<td>JIS_X0201</td>
<td>ISO-2022 encoding for Japanese (JIS_X0201)</td>
<td>10093</td>
</tr>
<tr>
<td>IBM-918</td>
<td>918</td>
<td>IBM918</td>
<td>EBCDIC Urdu</td>
<td>10126</td>
</tr>
<tr>
<td>IBM-921</td>
<td>921</td>
<td>ISO-8859-13</td>
<td>ISO 8859-13 PC Baltic (without euro update)</td>
<td>10014</td>
</tr>
<tr>
<td>IBM-922</td>
<td>922</td>
<td>cp922</td>
<td>IPC Estonian (without euro update)</td>
<td>10056</td>
</tr>
<tr>
<td>IBM-933</td>
<td>933</td>
<td>IBM933</td>
<td>IBM EBCDIC Korean CP933</td>
<td>933</td>
</tr>
<tr>
<td>IBM-942</td>
<td>942</td>
<td>IBM-942</td>
<td>PC Japanese SJIS-78 syntax (IBM-942)</td>
<td>10015</td>
</tr>
<tr>
<td>IBM-943</td>
<td>943</td>
<td>IBM-943</td>
<td>PC Japanese SJIS-90 (IBM-943)</td>
<td>10016</td>
</tr>
<tr>
<td>eucKR</td>
<td>970</td>
<td>EUC-KR</td>
<td>EUC Korean</td>
<td>10029</td>
</tr>
<tr>
<td>eucCN</td>
<td>1383</td>
<td>GB2312</td>
<td>Chinese EUC</td>
<td>10024</td>
</tr>
<tr>
<td>IBM-4899</td>
<td>4899</td>
<td>IBM-4899</td>
<td>EBCDIC Hebrew (with euro)</td>
<td>10159</td>
</tr>
<tr>
<td>IBM-4971</td>
<td>4971</td>
<td>IBM-4971</td>
<td>EBCDIC Greek (with euro update)</td>
<td>10160</td>
</tr>
<tr>
<td>eucJP</td>
<td>5050</td>
<td>IBM-5050</td>
<td>Japanese EUC (Packed Format)</td>
<td>10018</td>
</tr>
<tr>
<td>IBM-5123</td>
<td>5123</td>
<td>IBM-5123</td>
<td>EBCDIC Japanese Latin (with euro update)</td>
<td>10164</td>
</tr>
<tr>
<td>IBM-5346</td>
<td>5346</td>
<td>MS1250</td>
<td>MS Windows Latin 2 (Central Europe)</td>
<td>2250</td>
</tr>
<tr>
<td>IBM-5347</td>
<td>5347</td>
<td>MS1251</td>
<td>MS Windows Cyrillic (Slavic)</td>
<td>2251</td>
</tr>
<tr>
<td>IBM-5348</td>
<td>5348</td>
<td>MS1252</td>
<td>MS Windows Latin1 (ANSI), superset of Latin1</td>
<td>2252</td>
</tr>
<tr>
<td>IBM-5349</td>
<td>5349</td>
<td>MS1253</td>
<td>MS Windows Greek</td>
<td>2253</td>
</tr>
<tr>
<td>IBM-5350</td>
<td>5350</td>
<td>MS1254</td>
<td>MS Windows Latin 5 (Turkish), superset of ISO 8859-9</td>
<td>2254</td>
</tr>
<tr>
<td>IBM-5351</td>
<td>5351</td>
<td>IBM-5351</td>
<td>MS Windows Hebrew (older version)</td>
<td>10061</td>
</tr>
<tr>
<td>IBM-5352</td>
<td>5352</td>
<td>IBM-5352</td>
<td>MS Windows Arabic (older version)</td>
<td>10063</td>
</tr>
<tr>
<td>IBM-5353</td>
<td>5353</td>
<td>IBM-5353</td>
<td>MS Windows Baltic (older version)</td>
<td>10065</td>
</tr>
<tr>
<td>IBM-5354</td>
<td>5354</td>
<td>MS1258</td>
<td>MS Windows Vietnamese</td>
<td>2258</td>
</tr>
<tr>
<td>GB18030</td>
<td>5488</td>
<td>gb18030</td>
<td>GB 18030 MBCS code page</td>
<td>1392</td>
</tr>
<tr>
<td>IBM Codeset Name</td>
<td>CCSID</td>
<td>ICU Name</td>
<td>ICU Description</td>
<td>ICU ID</td>
</tr>
<tr>
<td>------------------</td>
<td>-------</td>
<td>----------</td>
<td>-----------------</td>
<td>--------</td>
</tr>
<tr>
<td>IBM-8482</td>
<td>8482</td>
<td>IBM-8482</td>
<td>EBCDIC Japanese Katakana SBCS (with euro update)</td>
<td>10165</td>
</tr>
<tr>
<td>IBM-9027</td>
<td>9027</td>
<td>IBM-1371</td>
<td>EBCDIC Taiwan Extended (SBCS IBM-1159 combined with DBCS IBM-9027)</td>
<td>10154</td>
</tr>
</tbody>
</table>
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 Integration Service to connect to multiple queue managers.

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

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.

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 Integration Service reads messages from the queue for a specified period of time.

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

A
associated source definition
  description 14
  importing 14
  types 14
associated source qualifier
  configuring 24
  description 4, 17
  types 18
  WebSphere MQ source message data format 18
  working with 18

C
channel
  WebSphere MQ, definition 47
channel table file
  WebSphere MQ, definition 47
client-connection channel
  WebSphere MQ queue manager, configuring 8
  WebSphere MQ, definition 47
code pages
  supported code pages for WebSphere MQ sessions 43
  supported code pages for WebSphere MQ sources and targets 44
  using in WebSphere MQ sessions 5
  WebSphere MQ sources 44
  WebSphere MQ targets 44
coded character set identifier (CCSID)
  WebSphere MQ, configuring 10
Concat
  description for WebSphere MQ 18
  continuous workflows
    WebSphere MQ, configuring 38

D
datatypes
  MQHEX datatype conversion 41
  PowerExchange for WebSphere MQ 41
destructive read
  for WebSphere MQ source queue clean-up 21
  resilience, configuring WebSphere MQ sessions 36
  WebSphere MQ session, configuring 35
dynamic MQSeries target definitions
  creating 31
  description 27
  editing 31
  message data field datatype 28
  working with 28

E
EndTime
  description for WebSphere MQ 19
environment variables
  WebSphere MQ, setting 8

F
filter conditions
  creating in MQ Source Qualifier 23
  description for WebSphere MQ 22
  in highly available WebSphere MQ sessions 36
  MQ Source Qualifier syntax 22
  session properties, configuring 34
  WebSphere MQ message header ports 23
flat files
  associated MQ source qualifier 25
  associated source definition 11
  static WebSphere MQ target 30
FlushLatency
  description for WebSphere MQ 19
  MQ Source Qualifier, configuring 20
ForcedEOQ
  description for WebSphere MQ 19
functions
  Concat for WebSphere MQ 18
  EndTime for WebSphere MQ 19
  FlushLatency for WebSphere MQ 19
  ForcedEOQ for WebSphere MQ 19
  Idle for WebSphere MQ 18
  in MQ Source Qualifier filter condition 18
  MsgCount for WebSphere MQ 18
  queue reading mode, defining for WebSphere MQ 19
  RemoveMsg(TRUE) 21
  StartTime for WebSphere MQ 19

G
glossary
  WebSphere MQ terms 47
grid
  WebSphere MQ MessageCount mode restriction 19
H

high availability
description for WebSphere MQ 33
WebSphere MQ message count mode restriction 19
WebSphere MQ session, configuring 35

I

idle mode
description for WebSphere MQ 18
MQ Source Qualifier, configuring 19
WebSphere MQ, definition 47
incremental data extraction
description for WebSphere MQ 21
Integration Service process
supported code pages for WebSphere MQ 43

L

Lookup transformation
in WebSphere MQ mappings 33
lookups
WebSphere MQ sources 33

M

mapping parameters
WebSphere MQ, setting 34
mapping variables
MQ Source Qualifier filter, setting 34
message count
See MsgCount
message count mode
WebSphere MQ, definition 47
message data
description for WebSphere MQ 3
message header
description for WebSphere MQ 2
WebSphere MQ message data, writing 30
WebSphere MQ, list of fields 2
message queue
description WebSphere MQ 2
WebSphere MQ data source 11
WebSphere MQ, definition 47
message recovery
description for WebSphere MQ 35
specifying partitions and a recovery cache folder for WebSphere MQ 37
WebSphere MQ session, configuring 35
WebSphere MQ session, configuring resilience 36
WebSphere MQ sessions, configuring 35
messages
WebSphere MQ message data field 3
WebSphere MQ, filtering 18
minimum system requirements
PowerExchange for WebSphere MQ 7
MQ Source Qualifier
configuring 24
description 4, 17
filtering data 24

for multiple message queues 18
message data size, setting 24
working with 17
MQHEX
WebSphere MQ datatype conversion 41
MQSeries source definitions
description 11
editing 12
joining 18
message data field 12
message header fields 12
working with 12
MQSeries target definitions
See dynamic MQSeries target definitions
See static WebSphere MQ target definitions
MsgCount
configuring 19
description for WebSphere MQ 18
MsgId
MQSeries target definitions 28

N

Normalizer transformation
associated source qualifier 25

P

partitioning
MQ sessions 36
partitions
specifying partitions and a recovery cache folder for WebSphere MQ 37
pass-through ports
WebSphere MQ message IDs in XML Generators 29
performance
WebSphere MQ sessions 39
PowerExchange for WebSphere MQ
configuring 7
minimum system requirements 7
prerequisites
PowerExchange for WebSphere MQ 7

Q

queue clean-up
destructive read 21
RemoveMsg(TRUE) 21
WebSphere MQ, definition 47
queue manager
configuring channels 8
description for WebSphere MQ 2
WebSphere MQ, definition 47
queue reading modes
description for WebSphere MQ 19
idle mode 19
message count mode 19
queue reading source
time slice mode 19
real-time sessions
   description for WebSphere MQ 34
   WebSphere MQ continuous workflows, configuring 38
   WebSphere MQ FlushLatency, configuring 34
   WebSphere MQ forced end of queue, configuring 35
   WebSphere MQ sessions, performance 39
recovery
   See message recovery
Recovery Cache Folder (property)
   WebSphere MQ, configuring 24
Recovery Strategy (property)
   WebSphere MQ session, configuring 36
RemoveMsg(TRUE)
   description for WebSphere MQ 19, 21
   for queue clean-up 21
resilience
   WebSphere MQ session, configuring 35
scheduling workflows
   WebSphere MQ 38
server-connection channel
   description 8
   WebSphere MQ, definition 47
sessions
   partitioning for MQ 36
   $$$SessStartTime variable
   WebSphere MQ, using for StartTime 20
source definitions
   associated 11
   MQSeries 11
SSL
   overview for WebSphere MQ 4
   WebSphere MQ, configuring 9
StartTime
   description for WebSphere MQ 19
   WebSphere MQ, using $$$SessStartTime 20
static WebSphere MQ target definitions
   creating 31
   description 27
   importing 31
   message header data, writing 30
   multiple targets 31
   working with 30
target connection groups
   transactional consistency for WebSphere MQ targets 36
   WebSphere MQ transactional consistency 28
time slice mode
   description for WebSphere MQ 19
   WebSphere MQ, configuring 19
   WebSphere MQ, definition 48
transactional consistency
   dynamic MQ targets 28
   WebSphere MQ targets, configuring 36
   WebSphere MQ, definition 48
troubleshooting
   WebSphere MQ mappings, importing 32
   WebSphere MQ, resilience 40
Unicode
   WebSphere MQ queue manager, configuring 10
VSAM
   associated source definition 11
WebSphere MQ
   architecture 1
   channels for queue managers, configuring 8
   description 1
   installing 7
   WebSphere MQ listener
   starting before a workflow run 39
   WebSphere MQ message IDs
   in XML Generator transformations 29
   in XML Parser transformation 13
   WebSphere MQ server
   assigning message header values 28
   WebSphere MQ sessions
   message recovery, configuring 35
   performance, increasing 39
   tuning parameters 39
   WebSphere MQ sources
   code pages 44
   description 11
   WebSphere MQ targets
   code pages 44
   description 27
XML
   static WebSphere MQ targets 30
   WebSphere MQ message data, reading 13
   WebSphere MQ message data, writing 29
   XML Generator transformation for WebSphere MQ target 29
   XML Parser transformation for WebSphere MQ source 13
   XML Generator transformation
   pass-through ports for WebSphere MQ message ID data 29
NOTICES

This Informatica product (the "Software") includes certain drivers (the "DataDirect Drivers") from DataDirect Technologies, an operating company of Progress Software Corporation ("DataDirect") which are subject to the following terms and conditions:

1. THE DATADIRECT DRIVERS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.

2. IN NO EVENT WILL DATADIRECT OR ITS THIRD PARTY SUPPLIERS BE LIABLE TO THE END-USER CUSTOMER FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL OR OTHER DAMAGES ARISING OUT OF THE USE OF THE ODBC DRIVERS, WHETHER OR NOT INFORMED OF THE POSSIBILITIES OF DAMAGES IN ADVANCE. THESE LIMITATIONS APPLY TO ALL CAUSES OF ACTION, INCLUDING, WITHOUT LIMITATION, BREACH OF CONTRACT, BREACH OF WARRANTY, NEGLIGENCE, STRICT LIABILITY, MISREPRESENTATION AND OTHER TORTS.