Informatica® PowerExchange
10.4.0

Bulk Data Movement Guide

Informatica PowerExchange Bulk Data Movement Guide
10.4.0
December 2019

Publication Date: 2020-05-05

Table of Contents

Preface

Use the *Informatica® PowerExchange® Bulk Data Movement Guide* to learn how to configure PowerExchange bulk data movement operations for each data source type. PowerExchange can read data at a specific point-in-time from PowerExchange relational and nonrelational sources and make the data available to PowerCenter batch sessions to extract and load to targets.

Informatica Resources

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

Informatica Network

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

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

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

Informatica Knowledge Base

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

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

Informatica Documentation

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

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

Informatica Product Availability Matrices

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

Informatica Velocity

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

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

Informatica Marketplace

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

Informatica Global Customer Support

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

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

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

CHAPTER 1

Introduction to PowerExchange Bulk Data Movement

This chapter includes the following topics:

PowerExchange Bulk Data Movement Overview

PowerExchange can efficiently move large amounts of data between different operating systems in a single operation. You can use this bulk data movement capability to materialize or completely refresh a data target, resynchronize a data source and data target, or create a data warehouse, data mart, or operational data store.

After you materialize a target, you can use PowerExchange Change Data Capture (CDC), if this option is available at your site, to update the target on a continuous basis. By moving only the change data, you can refresh data on the target faster.

PowerExchange can move bulk data between sources and targets of different types on the same operating system or on different operating systems. PowerExchange works in conjunction with PowerCenter® to extract data from and write data to relational databases, nonrelational databases, flat files, and sequential data sets. The types of data sources and targets that you can use for bulk data movement depend on your PowerExchange license.

On Linux, UNIX, and Windows systems, PowerExchange can move bulk data for the following types of data sources and targets:

- DB2 for Linux, UNIX, and Windows tables
- Oracle tables
- Microsoft SQL Server tables
- Flat files

On i5/OS systems, PowerExchange can move bulk data for DB2 for i5/OS tables and flat files.

On z/OS systems, PowerExchange can use the following types of databases and files as either bulk data sources or targets, except where noted:

- Adabas files
- CA Datacom databases, as data sources only
- DB2 for z/OS tables
- CA IDMS databases, as data sources only
- IMS databases
- VSAM data sets
- Sequential data sets

For DB2 for z/OS, PowerExchange can use the DB2 LOAD utility to load data to DB2 tables. For IMS, PowerExchange can use an IMS unload data set as the source.

If the source data resides on tape, PowerExchange can read the data from the tape data sets, including data sets that have a block size greater than 32,760 bytes.

Restriction: Where not otherwise noted, the maximum length of a source record for which PowerExchange can move bulk data is 144 KB.

Bulk Data Movement Components

PowerExchange uses some or all of the following components to move bulk data:

PowerExchange Client for PowerCenter (PWXPC)

A PowerCenter component that integrates PowerCenter with PowerExchange so that PowerCenter can access PowerExchange-controlled data and write it to various targets. Alternatively, you can use the PowerExchange ODBC drivers. However, Informatica recommends using PWXPC instead of the ODBC drivers. PWXPC provides additional functionality and better performance.

For information about PWXPC and the ODBC drivers, see *PowerExchange Interfaces for PowerCenter*.

PowerExchange Navigator

The graphical user interface from which you define and manage data maps for nonrelational data sources and targets and for DB2 tables that require user-defined fields and expressions. PowerExchange uses these definitions to determine the data sources and targets to process. Also, from the PowerExchange Navigator, you can create optional personal metadata profiles and perform database row tests. By performing a row test on a data map, you can view the source data to verify that PowerExchange can access it. By performing a row test on a personal metadata profile, you can view metadata for a data source.

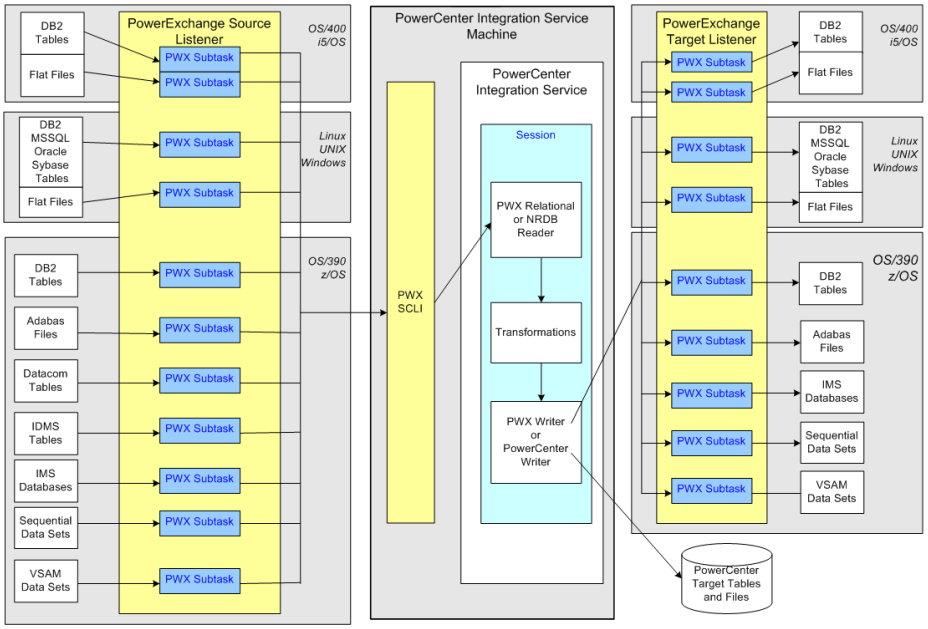
PowerExchange Listener

A PowerExchange component that manages data maps for nonrelational files and DB2 tables and maintains them in the DATAMAPS file. The PowerExchange Listener also handles bulk data extraction requests from PowerCenter.

If a data source or target is on a system that is remote from the one on which you are using PowerExchange, you must run an additional PowerExchange Listener on that remote system to communicate with PowerExchange.

Example Bulk Data Movement Configuration

The following figure shows an example bulk data movement configuration of PowerExchange with PowerCenter, which includes sources and targets on z/OS, i5/OS, and distributed systems:



In this example, PowerExchange uses PWXPC to integrate with PowerCenter. PowerExchange extracts bulk data from and loads bulk data to the following types of databases and files:

- Relational databases and flat files on Linux, UNIX, and Windows
- DB2 tables and flat files on i5/OS
- Relational databases, nonrelational databases, and sequential data sets on z/OS

PowerExchange uses its access methods to read bulk data from and write bulk data to relational and nonrelational databases and sequential data sets on z/OS and to DB2 tables and flat files on i5/OS. On Linux, UNIX, and Windows systems, the PowerCenter Reader and Writer extracts and loads data for relational data sources and targets and flat files.

Note: PowerExchange also can read data from and write data to DB2, Microsoft SQL Server, and Oracle tables on Linux, UNIX, or Windows.

Because the target systems are remote from the source systems, a PowerExchange Listener runs on all of these systems.

Bulk Data Movement Security

A PowerExchange user must have the proper level of authority to connect to the PowerExchange Listener and to access the PowerExchange resources and the data sources and targets.

On z/OS and i5/OS systems, you can include the SECURITY statement in the DBMOVER configuration member to require a user ID and password from an attached client for access to PowerExchange resources and data sources and targets.

On supported Linux, UNIX, and Windows systems, you can include the SECURITY statement along with related statements in the DBMOVER configuration member to require LDAP authentication to verify the credentials of a client that is trying to connect to the PowerExchange Listener or PowerExchange Logger.

On z/OS, the following additional levels of security are available, depending the type of data source or target:

- For DB2 for z/OS data, you can include the MVSDB2AF statement in the DBMOVER configuration member to control whether PowerExchange uses the call attachment facility (CAF) or Recovery Resource Services attachment facility (RRSAF) to connect to DB2. To use DB2 authentication methods to control user access to DB2 tables, select RRSAF. To use the PowerExchange Listener user ID to verify user access, select CAF.
- For Datacom data, you can define RACF resource profiles to prevent unauthorized READ access to Datacom tables.
- For Adabas data, you can use any of the following security methods:
  - Adabas password security at the file level
  - Adabas Security by Value method
  - RACF resource profiles to restrict WRITE access to Adabas databases and files

For more information about security, see the *PowerExchange Reference Manual*.

Bulk Data Movement Task Flow

After you install PowerExchange, perform tasks to implement a bulk data movement that uses PowerExchange in conjunction with PowerCenter.

The following table lists the high-level tasks that you perform and provides references to documentation for performing the tasks:

Step	Task	References
1	Configure the PowerExchange Listener.	Chapter 2, "PowerExchange Listener" on page 16
2	Start the PowerExchange Listener.	Chapter 2, "PowerExchange Listener" on page 16
3	Create the target database structure, if necessary.	Your DBMS documentation
4	From the PowerExchange Navigator, create data maps for any nonrelational sources and for any DB2 sources that require user-defined fields and expressions.	*PowerExchange Navigator User Guide*
5	(Optional, if using a data map) From the PowerExchange Navigator, perform a database row test of each data source to verify that PowerExchange can access the data and to test any data maps that you are using.	*PowerExchange Navigator User Guide*

Step	Task	References
6	From the PowerCenter Designer, create a mapping.	- *PowerExchange Interfaces for PowerCenter* - *PowerCenter Designer Guide*
7	From the PowerCenter Workflow Manager, define a workflow and session, configure a connection, and run the workflow to perform the bulk data movement	- *PowerExchange Interfaces for PowerCenter* - *PowerCenter Workflow Basics Guide*

Note: The task flow can vary significantly by DBMS type and bulk data movement strategy. For example, for DB2 for z/OS, you can use a DB2 image copy as a data source or use the DB2 LOAD utility to load data to the target. For IMS and IDMS, you can use a netport job.

CHAPTER 2

PowerExchange Listener

This chapter includes the following topics:

PowerExchange Listener Overview

The PowerExchange Listener provides the following services:

- Stores and manages data maps for data sources.
- Makes bulk data available to PowerCenter for data extraction and loads.
- Makes bulk data available to the PowerExchange Navigator for database row tests.

A PowerExchange Listener interacts with the following PowerExchange components:

- PowerExchange Navigator
- Other PowerExchange Listeners

PowerExchange Listener Configuration Tasks for Bulk Data Movement

Complete some or all of the following PowerExchange Listener configuration tasks, as needed for your environment:

- Configure the PowerExchange Listener by defining statements in the DBMOVER configuration file. You can accept default settings or you can customize configuration statements for your environment.
- Configure the TCP/IP port number if the default port number of 2480 is not available or if you use multiple PowerExchange Listeners and need to define additional ports for them.
- Configure alternative logging for message and trace information. PowerExchange provides a single log file by default. Because logging data to a single file is inefficient and can result in the log file becoming full, use the PowerExchange alternative logging strategies that are described in the *PowerExchange Reference Manual*. To implement alternative logging, you must specify the TRACING parameter in the DBMOVER configuration file.

  On z/OS systems, the default DBMOVER configuration file that is shipped with PowerExchange includes the TRACING statement. Also, the JCL that is provided for PowerExchange jobs, such as netport jobs, contains the following statement for allocating log data sets:

  ```
  DTLLOG01 DD SYSOUT=*
  ```

  Often, this default logging setup is sufficient. However, multiple other logging strategies are available.
- On Linux, UNIX, or Windows, place an exact or modified copy of the dbmover.cfg and license.key files in a location other than the default location. This practice preserves the original files if you upgrade to a new release. To define the alternative files to the PowerExchange Listener, you can enter the full path and file name of each file in the CONFIG and LICENSE statements on the dtllst command when you start the PowerExchange Listener. Alternatively, you can specify the alternative files in the PWX_CONFIG and PWX_LICENSE environment variables.
- On z/OS, configure the PowerExchange Listener JCL.
- On z/OS, if you use CA TCPAcess instead of IBM z/OS TCP/IP, set up TCP/IP sockets to use TCPAccess.

Customizing the PowerExchange DBMOVER File

You probably will need to customize the PowerExchange DBMOVER configuration file. The configuration file includes the following parameters that affect a PowerExchange Listener:

- APPBUFSIZE
- APPBUFSIZEDYN
- CMDNODE
- CREDENTIALS_CASE
- DATAMAP_SERVER
- DM_SUBTASK
- DMX_DIR
- LISTENER
- LOGPATH
- MAXTASKS

- NODE
- SECURITY
- SUBTIMEOUT
- SVCNODE
- TCPIVER
- TRACING

After editing the DBMOVER configuration file, restart the PowerExchange Listener for your changes to take effect.

For descriptions of all DBMOVER statements, see the *PowerExchange Reference Manual*.

APPBUFSIZE Statement

The APPBUFSIZE statement specifies the maximum buffer size, in bytes, for reading or writing data.

When the buffer size is reached, PowerExchange sends the buffer data across the network to the system that requests the data and then starts using another buffer.

If dynamic application buffer sizing is enabled, the APPBUFSIZE statement specifies the initial maximum buffer size. Dynamic application buffer sizing is enabled by default. You can explicitly enable it by specifying Y for the APPBUFSIZEDYN statement in the DBMOVER configuration file.

Operating Systems: All

Related Statements: APPBUFSIZEDYN

Required: No

Syntax:

```
APPBUFSIZE={buffer_size|256000}
```

Value: For the *buffer_size* variable, enter a value that is greater than the maximum size of a single row. Valid values are from 34816 through 8388608. Default is 256000.

Usage Notes:

- For bulk data movement sessions that use reader or writer partitions, you can increase the APPBUFSIZE value to help improve session performance.

APPBUFSIZEDYN Statement

The APPBUFSIZEDYN statement specifies whether to enable dynamic application buffer sizing.

The DBMOVER APPBUFSIZE statement defines the initial size of the application buffer for all connections made during a PowerExchange Listener run. If APPBUFSIZEDYN=Y, PowerExchange resizes the application buffers for individual connection as needed.

The APPBUFSIZEDYN statement applies to PowerExchange connections to data sources with either fixed-length or variable-length records. A variable-length record is a record with at least one variable-length field. A variable-length field has a datatype of VARCHAR or VARBIN.

For each connection to a data source with variable-length records, PowerExchange resizes the application buffer when it encounters a record that is too large to fit into the buffer. PowerExchange increases the size of the application buffer to a value of ten times the size of the record that has overflowed, up to the maximum application buffer size of 8 MB. The new size remains in effect for the duration of the Listener run or until the application buffer is resized again. PowerExchange never decreases the application buffer size for a connection after the Listener run has started.

For each connection to a data source with fixed-length records, PowerExchange determines the record length when the connection is opened and resizes the application buffer once, up to the maximum application buffer size of 8 MB, as needed.

Operating Systems: All

Data Sources: All

Related Statements: APPBUFSIZE

Required: No

Syntax:

```
APPBUFSIZEDYN={N|Y}
```

Valid Values:

- **N.** PowerExchange does not perform dynamic application buffer sizing.
- **Y.** PowerExchange performs dynamic application buffer sizing.

Default is Y.

CMDNODE Statement

The CMDNODE statement specifies connection information for a PowerExchange process that is the target of pwxcmd commands.

Include the CMDNODE statement in the dbmover.cfg file on the Linux, UNIX, or Windows system from which you issue pwxcmd commands.

Operating Systems: Linux, UNIX, and Windows

Related Statements: SVCNODE

Required: No

Syntax:

```
CMDNODE=(service_name
        ,{CONDENSE|ECCR|LISTENER}
        ,host_name
        ,connect_port
)
```

Parameters:

service_name

Required. User-defined name of the command-handling service for the PowerExchange process to which you want to send pwxcmd commands. Use this service name when issuing pwxcmd commands to this PowerExchange process. For connection to a PowerExchange log-based ECCR or Datacom table-based ECCR on z/OS, this value is the ECCR name. Ensure that this value matches the ECCR name specified in the corresponding SVCNODE statement.

{CONDENSE|ECCR|LISTENER}

Required. The PowerExchange service type.

Options are:

- **CONDENSE**. PowerExchange Condense or PowerExchange Logger for Linux, UNIX, and Windows.
- **ECCR**. A PowerExchange Adabas, IDMS, or IMS log-based ECCR or Datacom table-based ECCR on z/OS.

- **LISTENER**. A PowerExchange Listener.

No default value is provided.

host_name

Required. The host name or IP address of the target system to which you want to send commands.

connect_port

Required. The port number on which the command-handling service for the PowerExchange process listens for pwxcmd commands. This port number must match the port number in the corresponding SVCNODE statement.

CREDENTIALS_CASE Statement

The CREDENTIALS_CASE statement controls the case that PowerExchange uses for operating system user IDs and passwords.

Operating Systems: All

Required: No

Syntax:

```
CREDENTIALS_CASE={A|D|S]
```

Valid Values:

- **A.** On z/OS or i5/OS, PowerExchange processes and passes user IDs and passwords to the operating system for authentication, as follows:
  1. PowerExchange converts the user ID to uppercase.
  2. PowerExchange checks whether the operating system is configured to handle mixed-case passwords.
     - If so, PowerExchange passes the user ID in uppercase and the password in the case that you supplied it to the operating system for authentication.
     - If not, PowerExchange converts the password to uppercase and passes the user ID and password to the operating system for authentication.

  On Linux, UNIX, or Windows, PowerExchange passes the user ID and password in the case that you supplied them to the operating system for authentication.
- **D.** On i5/OS or z/OS, PowerExchange converts user IDs and passwords to uppercase and then passes them to the operating system for authentication.

  On Linux, UNIX, or Windows, PowerExchange passes the user ID and password in the case that you supplied them to the operating system for authentication.
- **S.** On i5/OS or z/OS, PowerExchange converts the user ID to uppercase and leaves the password in the case that you supplied it. Then, PowerExchange passes the user ID and password to the operating system for authentication.

  On Linux, UNIX, or Windows, PowerExchange passes the user ID and password in the case that you supplied them to the operating system for authentication.

Default is D.

DATAMAP_SERVER Statement

The DATAMAP_SERVER statement specifies the node name of the PowerExchange Listener that acts as the server for access requests to the file that stores data maps.

Use this statement to reduce overhead in an environment where multiple PowerExchange Listeners are running and make frequent open and close requests on the data map file.

Operating Systems: z/OS

Related Statements: DM_SUBTASK and NODE

Required: No

Syntax:

```
DATAMAP_SERVER=node_name
```

Value: For the *node_name* variable, enter the node name from a NODE statement in the DBMOVER member that points to the PowerExchange Listener that accesses the DATAMAPS data set.

Generally, you designate one PowerExchange Listener as the data map server. To do so, define the DATAMAP_SERVER statement in the DBMOVER members of the PowerExchange Listeners that contact the data map server.

Usage Notes:

- If you have two or more PowerExchange Listeners that share the same DATAMAPS data set, you can configure PowerExchange to use the PowerExchange Listener that starts first as the data map server. If you have three or more PowerExchange Listeners that share the same DATAMAPS data set, you must designate one of them as the data map server.

  To use a PowerExchange Listener as the data map server, configure the following statements in the DBMOVER member for each PowerExchange Listener:

  - In the DATAMAP_SERVER statement, specify the node name of the other PowerExchange Listener.
  - Specify Y for the DM_SUBTASK statement.

  The first PowerExchange Listener that starts becomes the data map server and the other PowerExchange Listeners access data maps through it. PowerExchange writes the following messages to the PowerExchange message log file:

  - The PWX-02804 message to the PowerExchange message log file of the PowerExchange Listener that acts as the data map server.
  - The PWX-02800 and PWX-02805 messages to the PowerExchange message log files of the PowerExchange Listeners that do not act as the data map server. The PWX-02805 message indicates node name, IP address, and port number of the PowerExchange Listener that is the data map server.
- If you also specify DM_SUBTASK=Y and a PowerExchange Listener holds an exclusive lock on the DATAMAPS data set, enter DATAMAP_SERVER=*node_name* to enable other tasks, such the IMS synchronous ECCR or a netport job, to access the data set. Otherwise, the tasks fail.

  Alternatively, use data maps caching. Informatica recommends this approach to improve performance and to avoid access problems that can occur if the PowerExchange Listener data maps server stops. To implement data maps caching, specify DM_SUBTASK=R and DMXCACHE_MAX_MEMORY_MB=20. With these settings, the PowerExchange Listener opens the data set in read-only mode and allows other tasks to access the data set.

DM_SUBTASK Statement

The DM_SUBTASK statement controls how the PowerExchange Listener accesses the file that stores data maps.

Use this statement to reduce overhead in an environment where multiple PowerExchange Listeners are running and make frequent open and close requests on the data map file.

Operating Systems: z/OS

Related Statements: DATAMAP_SERVER, DM_RESOURCE, DMXCACHE_DELETEECSA, DMXCACHE_MAX_MEMORY_MB, DMXCACHE_MULTIPLEJOBS, RACF_CLASS, and SECURITY

Required: No

Syntax:

```
DM_SUBTASK={N|R|Y}
```

Valid Values:

- **N**. PowerExchange opens and closes the DATAMAPS data set in PowerExchange Listener subtasks, as required.
- **R**. To improve the read performance for data maps, the PowerExchange Listener starts a subtask that opens the DATAMAPS data set in read mode. No change in processing or performance occurs for insertion and deletion of data maps.

  The R option is faster than the N option but slower than the Y option.
- **Y**. A single PowerExchange subtask owns and opens the DATAMAPS data set. All other subtasks use the data map subtask to access data maps. The DATAMAPS data set remains open until the PowerExchange Listener is shut down.

  Enter Y to decrease resource utilization and improve the performance of the PowerExchange Listener. PowerExchange opens that DATAMAPS data set one time during the life of a PowerExchange Listener address space instead of opening and closing the data set for each request.

  If you enter Y and also define the DATAMAP_SERVER statement, PowerExchange designates the PowerExchange Listener as a data map server.

Default is N.

Usage Notes:

- If you specify DM_SUBTASK=Y and also specify 2 for the first parameter of the SECURITY statement, PowerExchange checks this resource profile to determine whether to permit access to the DATAMAPS data set. In this case, you must code the resource profile in your security system and grant access to all users that read or write data maps on z/OS.

  By default, PowerExchange checks the DTL.DATAMAP.DATASET resource profiles in the FACILITY class. The profile name and class might be different if you specify other values on the DM_RESOURCE and RACF_CLASS statements.
- If you specify DM_SUBTASK=Y and a PowerExchange Listener holds an exclusive lock on the DATAMAPS data set, also enter DATAMAP_SERVER=*node_name* to enable other tasks, such the IMS synchronous ECCR or a netport job, to access the data set. Otherwise, the tasks fail.

  Alternatively, use data maps caching. Informatica recommends this approach to improve performance and to avoid access problems that can occur if the PowerExchange Listener data maps server stops. To implement data maps caching, specify DM_SUBTASK=R and DMXCACHE_MAX_MEMORY_MB=20. With these settings, the PowerExchange Listener opens the data set in read-only mode and allows other tasks to access the data set.

DMX_DIR Statement

The DMX_DIR statement specifies the location that PowerExchange uses to read and store data maps.

Operating Systems: All

Required: No

Syntax:

```
DMX_DIR=location
```

Value: For the *location* variable, enter one of the following values, based on the operating system:

- **i5/OS**. The library name for the data maps. Maximum length is 10 characters.

  Default is STDATAMAPS.
- **Linux, UNIX, and Windows**. The fully qualified path that contains the data maps. Maximum length is 512 characters.

  On Linux and UNIX, default is ./datamaps.

  On Windows, default is .\datamaps.
- **z/OS**. The DD statement name from the JCL that points to the DATAMAPS data set. Maximum length is eight characters.

  Default is DATAMAP.

LISTENER Statement

The LISTENER statement defines the TCP/IP port on which a named PowerExchange Listener process listens for work requests.

You can define up to 10 LISTENER statements in a DBMOVER configuration file.

For netport jobs on z/OS, define a LISTENER statement with a unique port and define a NETPORT statement that references that port.

Optionally, you can specify SSL authentication and additional parameters that control TCP/IP buffer sizes and wait times.

Operating Systems: All

Related Statements: NETPORT for netport jobs and SSL for SSL authentication

Required: No

Syntax:

```
LISTENER=({listener_node|node1}
          ,TCPIP
          ,{port|2480}
         [,{send_bufsize|65536}]
         [,{receive_bufsize|65536}]
         [,{send_size|4096}]
         [,{receive_size|4096}]
         [,receive_timeout]
         [,ip_address]
         [,SSL]
)
```

Parameters:

{listener_node|node1}

Required. Node name of the TCP/IP port on which the PowerExchange Listener process listens.

Use this node name to select a LISTENER statement when you start the PowerExchange Listener process, as follows:

- On Linux, UNIX, and Windows, specify the listener node name on the dtllst command.
- On i5/OS, specify the listener node name in the SBMJOB command that runs the DTLLST program. For more information, see the *PowerExchange Command Reference*.
- On z/OS, specify the listener node name in the PARM field of the EXEC card in the JCL.

Default is node1.

TCPIP

Required. Communications protocol. TCPIP is the only valid option.

{*port*|2480}

Required. TCP/IP port used to listen for work requests. Valid values are from 1 through 65535. Default is 2480.

{*send_bufsize*|65536}

Optional. Size, in bytes, of the data portion of the TCP/IP send buffer. Valid values are from 1024 through 1048576. Default is 65536.

{*receive_bufsize*|65536}

Optional. Size, in bytes, of the data portion of the TCP/IP receive buffer. Valid values are from 1024 through 1048576. Default is 65536.

{*send_size*|4096}

Optional. Maximum size, in bytes, of the block of data that PowerExchange sends to TCP/IP at one time. If the data exceeds this size, PowerExchange splits data into multiple blocks until all of the data is sent. Valid values are from 512 through 1048576. Default is 4096.

Tip: Enter a value that is less than or equal to the TCP/IP send buffer size.

{*receive_size*|4096}

Optional. Maximum size, in bytes, of the block of data that PowerExchange processes from TCP/IP in one operation. If the data exceeds this size, PowerExchange splits data into multiple blocks until all of the data is received. Valid values are from 512 through 1048576. Default is 4096.

Tip: Enter a value that is greater than or equal to the TCP/IP receive buffer size.

receive_timeout

Optional. Number of seconds that PowerExchange uses as the receive timeout value when a long wait is required. PowerExchange uses this value for this request only. Valid values are from 1 through 14400.

ip_address

Optional. IP address that PowerExchange uses on the bind operation for the socket.

If you do not specify an IP address, PowerExchange uses INADDR_ANY on the bind operation, which causes TCP/IP to bind to all network interfaces on the host. Use this parameter if you have multiple network interfaces and want to restrict the port on which the PowerExchange Listener listens to a specific interface.

SSL

Optional. Specifies that PowerExchange uses SSL authentication. Specify this parameter on a Linux, UNIX, or Windows Listener machine that serves as the SSL server during SSL communciations.

Usage Notes:

- If you create a PowerExchange Listener Service, use the node name that you define in the LISTENER statement as follows:
  - If you create the Listener Service through Informatica Administrator, the node name value that you specify in the **Start Parameters** property must match the node name that you define in the LISTENER statement.
  - If you create the Listener Service through the infacmd pwx CreateListenerService command, the node name value that you specify for the -StartParameters option on the command must match the node name that you define in the LISTENER statement.

  When you create the Listener Service, the Service Manager associates it with the PowerExchange Listener process on the node. For more information about configuring and creating a Listener Service, see the *Informatica Application Service Guide*.
- If you issue pwxcmd commands or infacmd pwx commands to the PowerExchange Listener process, the node name that you define in the LISTENER statement must match the node name that you define in the SVCNODE statement in the DBMOVER configuration file. In this case, enter a string of up to 12 characters in length for the *listener_node* parameter of the LISTENER statement. This is the maximum length of the service name that you can specify in the SVCNODE statement.

LOGPATH Statement

The LOGPATH statement specifies a unique path and directory for PowerExchange message log files on a Linux, UNIX, or Windows system.

Define this statement to create message log files in a directory that is separate from your current working directory so that you can find the message log files more easily.

Operating Systems: Linux, UNIX, and Windows

Required: No

Syntax:

```
LOGPATH=directory
```

Value: For the *directory* variable, enter the full path to the directory where you want PowerExchange to write message log files. Default is the current working directory.

Usage Notes: If you also specify a value in the DETAIL_LOGPATH environment variable, the environment variable overrides the LOGPATH statement.

MAXTASKS Statement

The MAXTASKS statement defines the maximum number of concurrent tasks that can run under the PowerExchange Listener.

Operating Systems: All

Required: No

Syntax:

```
MAXTASKS={maximum_tasks|5}
```

Value: For the *maximum_tasks* variable, enter a number from 1 through 255. Default is 5.

Usage Notes:

- If resource usage by PowerExchange Listener tasks exceeds the resources available for the PowerExchange Listener, the Listener abends. In this case, you can try decreasing the MAXTASKS value to run fewer concurrent tasks. Although tasks can have different resource requirements, this statement can help limit the amount of resources that are used for PowerExchange Listener tasks and prevent these types of abends.
- The MAXTASKS parameter is not intended to be an operating system performance and tuning parameter. To balance the workload of multiple concurrent tasks, use the workload management facility of the operating system, for example, Workload Management (WLM) on z/OS.
- If you use PowerExchange tuning features that result in multiple threads and PowerExchange Listener subtasks, such as pipeline partitioning or multithreaded processing, you might need to increase the MAXTASKS value if PowerExchange processing slows or hangs. You also might need to increase this value if you add PowerCenter workflows. After the number of concurrent tasks reaches the MAXTASKS limit, the PowerExchange Listener rejects requests for additional tasks with message PWX- 00609:

  ```
  PWX-00650 10.3.0.111:3720 : Listener 10.33.40.42 -> 10.3.0.1 on port 1234 socket 51
  PWX-00609 Listener has temporarily stopped accepting connections.
  ```

  When the number of the concurrent tasks drops below the MAXTASKS limit once again, the PowerExchange Listener begins accepting requests for additional tasks.
- If you use connection pooling, ensure that the MAXTASKS value is large enough to accommodate the size of the connection pool.
- The maximum number of concurrent tasks that a PowerExchange Listener can support might be substantially less than 255 because of operating system resource constraints, such as virtual storage limitations on z/OS.

NODE Statement

The NODE statement defines the TCP/IP host name and port that PowerExchange uses to contact a PowerExchange Listener process.

You can specify up to 128 NODE statements in a DBMOVER configuration file.

Optionally, specify SSL authentication and additional parameters that control TCP/IP buffer sizes and wait times. And optionally, specify the *service_name* parameter to identify a PowerExchange Listener Service.

Operating Systems: All

Related Statements: NETPORT for netport jobs and SSL for SSL authentication

Required: No

Syntax:

```
NODE=({node_name|node1}
      ,TCPIP
      ,host_name
     ,{port|2480}
    [,{send_bufsize|65536}]
    [,{receive_bufsize|65536}]
    [,{send_size|4096}]
    [,{receive_size|4096}]
    [,receive_timeout]
    [,{SSL|ZOSSSL}]
    [,service_name]
    )
```

Parameters:

***node_name*|node1**

Required. Unique, user-defined name for this NODE statement. The name does not need to match the name of the PowerExchange Listener process. To contact the PowerExchange Listener process to which the statement points, enter this name in user interfaces that prompt for the location of the PowerExchange Listener, including the following interfaces:

- The **Location** attribute in a connection definition in the Informatica Developer or PowerCenter Workflow Manager
- The **Location** attribute in a source or target definition in the PowerCenter Designer
- The **Location** field in the PowerExchange Navigator dialog boxes

Default is node1.

TCPIP

Required. Communications protocol. TCPIP is the only valid option.

host_name

Required. TCP/IP host name or IP address for the PowerExchange Listener process that listens on the port specified in the *port* parameter. If the *service_name* parameter is specified, *host_name* is ignored.

{*port*|2480}

Required. TCP/IP port on which the PowerExchange Listener process that runs on the system specified in *host_name* listens. Valid values are from 1 through 65535. Default is 2480.

{*send_bufsize*|65536}

Optional. Size, in bytes, of the data portion of the TCP/IP send buffer. Valid values are from 1024 through 1048576. Default is 65536.

{*receive_bufsize*|65536}

Optional. Size, in bytes, of the data portion of the TCP/IP receive buffer. Valid values are from 1024 through 1048576. Default is 65536.

{*send_size*|4096}

Optional. Maximum size, in bytes, of the block of data that PowerExchange sends to TCP/IP at one time. If the data exceeds this size, PowerExchange splits data into multiple blocks until it sends all the data. Valid values are from 512 through 1048576. Default is 4096.

Tip: Enter a value that is less than or equal to the TCP/IP send buffer size.

{*receive_size*|4096}

Optional. Maximum size, in bytes, of the block of data that PowerExchange processes from TCP/IP in one operation. If the data exceeds this size, PowerExchange splits data into multiple blocks until it receives all the data. Valid values are from 512 through 1048576. Default is 4096.

Tip: Enter a value that is greater than or equal to the TCP/IP receive buffer size.

receive_timeout

Optional. Number of seconds that PowerExchange uses as the receive timeout value when a long wait is required. PowerExchange uses this value for this request only. Valid values are from 1 through 14400.

{SSL|ZOSSSL}

Optional. Specifies that PowerExchange uses SSL authentication.

Specify the ZOSSSL option only on a Linux, UNIX, or Windows SSL client that is communicating with a z/OS system. Otherwise, specify the SSL option on the Linux, UNIX, or Windows SSL client.

service_name

Optional. To configure an Informatica client tool or integration service to locate a PowerExchange Listener Service in the Informatica domain, specify the name of the Listener Service in the *service_name* parameter.

A client tool is the Developer tool or PowerCenter Client. An integration service is the PowerCenter Integration Service or Data Integration Service.

If you include this parameter, the Informatica client tool or integration service ignores the *host_name* parameter on the NODE statement and uses the *service_name* and *port* parameters to locate the Listener Service in the Informatica domain.

For more information about the PowerExchange Listener Service, see the *Informatica Application Service Guide*.

SECURITY Statement

The SECURITY statement controls PowerExchange user authentication and access to resources and commands.

Use the SECURITY statement in the DBMOVER configuration file to configure the following types of security:

- User authentication to access PowerExchange
- Access to files and data sets by PowerExchange jobs and tasks on z/OS and i5/OS
- User authorization to issue infacmd pwx commands to a PowerExchange application service in the Informatica domain
- User authorization to issue pwxcmd commands to a PowerExchange process
- User authorization to issue PowerExchange Listener LISTTASK and STOPTASK commands from the PowerExchange Navigator

Operating Systems: All

Related Statements: DM_RESOURCE, MVSDB2AF, and RACF_CLASS

Required: No

Syntax:

```
SECURITY=({0|1|2}
          ,{N|Y}
          [,LDAP]
          [,{ORACLE_LDAP|OPEN_LDAP}]
)
```

Parameters: The first positional parameter has the following valid values:
{0|1|2}

Controls whether PowerExchange requires users to enter a valid operating system user ID and a password or passphrase. Also controls whether PowerExchange checks user-entered credentials to control access to file and database resources and the issuance of certain PowerExchange commands.

Enter one of the following options:

- **0.** PowerExchange does not require users to specify a valid operating system user ID and password and ignores any credentials that users supply.

  On z/OS and i5/OS, PowerExchange uses the user ID under which the PowerExchange Listener or PowerExchange Condense task runs to control access to file resources. PowerExchange passes this user ID to the database system.

  On Linux, UNIX, and Windows, PowerExchange uses the user ID under which the PowerExchange Listener task runs to control access to file resources. RDBMS security controls PowerExchange access to database resources based on the user ID that users specify on the PWX connection or in the PowerExchange Logger CAPTURE_NODE_UID parameter.

  On all operating systems, PowerExchange does not check user authorization to issue commands. Any user can issue a command.

- **1.** On z/OS and i5/OS, PowerExchange requires users to specify a valid operating system user ID and a password or valid PowerExchange passphrase. PowerExchange checks these credentials when a PowerExchange task starts. Thereafter, PowerExchange controls access to file resources in the same manner as for option 0. For file access, PowerExchange uses the user ID under which the PowerExchange Listener or PowerExchange Condense task runs and passes this user ID to the database system.

  On Linux, UNIX, and Windows, unless you specify LDAP for the third parameter of the SECURITY statement on supported systems, PowerExchange does not require users to specify a valid operating system user ID and password to access file or database resources and does not check for these credentials. As for option 0, PowerExchange uses the user ID under which the PowerExchange Listener task runs to control access to file resources. RDBMS security controls PowerExchange access to database resources based on the user ID that users specify on the PWX connection or in the PowerExchange Logger CAPTURE_NODE_UID parameter.

  On all operating systems, PowerExchange does not check user authorization to issue commands. Any user can issue a command.

- **2.** Provides the most specific level of security.

  - On z/OS, Informatica recommends that you use option 2. PowerExchange controls access based on 1) an MVS user ID and a password or valid PowerExchange passphrase and 2) the access control features of your z/OS security product, such as RACF or ACF2.

    To read change data from the change stream, the ECCR must use a valid z/OS user ID and password or passphrase. The PowerExchange Listener checks these credentials when the ECCR task or job starts. To access the database to read data, PowerExchange passes the z/OS user ID and password or passphrase to the database system for database-specific security checking. In conjunction with the z/OS security product and MVS System Authorization Facility (SAF), PowerExchange checks the z/OS user ID and password or passphrase against the CAPX.REG.* resource profiles to control access to capture registrations.

    To extract change data, run PowerCenter CDC sessions with a PWXPC connection that specifies a valid z/OS user ID and password or passphrase. For the session to access extraction maps, these user credentials must have READ access to the PowerExchange data set that is defined in the DTLCAMAP DD statement of the PowerExchange Listener JCL.

    Note: A connection to DB2 for z/OS through the Call Attachment Facility (CAF) runs under the user ID of the PowerExchange Listener regardless of the security settings. DB2 uses the user ID that is specified on the connection only if the connection type is Recoverable Resource Manager Service Attachment Facility (RRSAF) or if offload processing is enabled.

PowerExchange also uses resource profiles to control who can run the following types of commands:

- pwxcmd commands for a PowerExchange Listener or PowerExchange Condense process that are issued form a Linux, UNIX, or Windows system
- PowerExchange Listener LISTTASK and STOPTASK commands that are issued from the PowerExchange Navigator or the DTLUTSK utility

- On i5/OS, PowerExchange requires users to specify a valid operating system user ID and password or passphrase. PowerExchange checks these credentials when a PowerExchange task starts. PowerExchange Listener subtask processes run under the supplied user ID and password or passphrase. PowerExchange uses this user ID and password or passphrase to control access to PowerExchange files. PowerExchange also passes this user ID and password or passphrase to the database system for data access.

  PowerExchange uses security objects to control who can run the following types of commands:

  - pwxcmd commands for a PowerExchange Listener or PowerExchange Condense process that are issued form a Linux, UNIX, or Windows system
  - PowerExchange Listener LISTTASK and STOPTASK commands that are issued from the SNDLSTCMD interface, the PowerExchange Navigator, or the DTLUTSK utility

- On Linux, UNIX, and Windows, unless you specify LDAP for the third parameter of the SECURITY statement on supported systems, PowerExchange does not require users to specify an operating system ID and password to access PowerExchange files or a database. PowerExchange uses the user ID and password under which the PowerExchange Listener runs or that PowerExchange Logger for Linux, UNIX, and Windows uses to control access to PowerExchange files. RDBMS security controls access to the database.

  However, you must specify a valid operating system user ID and password to run the following types of commands:

  - An infacmd pwx command to a PowerExchange application service in the Informatica domain
  - A pwxcmd command to a PowerExchange process

  PowerExchange checks these user credentials against the USER and AUTHGROUP COMMANDS statements in the sign-on file to determine if a user is authorized to issue an infacmd pwx or pwxcmd command. In this case, the second positional parameter in the SECURITY statement is ignored.

Default is 0.

The second positional parameter has the following valid values:

{N|Y}

Controls use of PowerExchange selective sign-on file to authorize users to connect to the PowerExchange Listener.

Enter one of the following options:

- **N.** PowerExchange does not use the selective sign-on file.
- **Y.** PowerExchange uses the USER statement with the ALLOW and IP subparameters in the selective sign-on file to restrict users who can connect to the PowerExchange Listener.

  Note: If you specify Y and also set the first parameter in the SECURITY statement to 1, PowerExchange uses the TASKCNTRL parameter in the USER statements in the sign-on file to control access to PowerExchange Listener LISTTASK and STOPTASK commands that are issued from the PowerExchange Navigator.

Default is N.

The optional third positional parameter has the following valid value:

LDAP

If you specify LDAP for the third positional parameter and specify 1 or 2 as the first positional parameter, PowerExchange uses LDAP authentication on supported Linux, UNIX, and Windows systems.

If you do not include the third parameter, PowerExchange does not use LDAP authentication.

The fourth positional parameter has the following valid values:

{ORACLE_LDAP|OPEN_LDAP}

If you specify LDAP for the third positional parameter, specifies which set of LDAP client libraries to load.

Enter one of the following options:

- **ORACLE_LDAP.** PowerExchange loads the Oracle LDAP client libraries.

  Select this option only if you have an Oracle LDAP installation. PowerExchange does not provide the Oracle LDAP client libraries.
- **OPEN_LDAP.** PowerExchange loads the OpenLDAP client libraries.

Default is ORACLE_LDAP.

Usage Notes:

- In the z/OS Installation Assistant, if you click **Advanced Parms** on the **General Parameters** page, you can define the SECURITY_LEVEL and SECURITY_PWX parameters. The SECURITY_LEVEL parameter corresponds to the first parameter in the SECURITY statement. The SECURITY_PWX parameter corresponds to the second parameter in the SECURITY statement.
- On z/OS, when you set the first parameter of the SECURITY statement to 1 or 2, you must APF-authorize the STEPLIB for the PowerExchange Listener and netport jobs. Otherwise, PowerExchange cannot complete user authentication or control resource access, and instead operates as if you set this parameter to 0.
- If you offload column-level processing for a z/OS data source to the Linux, UNIX, or Windows system where the PowerCenter Integration Service runs, PowerCenter CDC sessions use the **Map Location User** and **Map Location Password** values that you specify on the connection to control access to all resources. The connection must be a PWX NRDB CDC application connection or PWX DB2zOS CDC application connection for which offload processing is enabled.
- If you log data from z/OS data sources to remote PowerExchange Logger for Linux, UNIX, and Windows log files, set the SECURITY option to 2 in the DBMOVER configuration member on z/OS. Ensure that the user ID and password in the PowerExchange Logger for Linux, UNIX, and windows configuration file, pwxccl, is a valid z/OS user ID and password that can pass z/OS security checking. To read captured data from the PowerExchange Logger for z/OS log files, these user credentials must have READ access to CAPX.REG.* resources profiles in the FACILITY class, which are managed by your z/OS security product. Also, for CDC sessions to extract data from the log files, the PWXPC connection must specify the z/OS user ID and password in the **Map Location User** and **Map Location Password** connection attributes. These user credential needs READ access to the CAPX.CND.* resource profiles.

SUBMITTIMEOUT Statement

The SUBMITTIMEOUT statement specifies the time, in seconds, that a PowerExchange Listener waits to receive notification from a spawned batch job that it has started.

Operating Systems: z/OS

Related Statements: LOADJOBFILE and NETPORT

Required: No

Syntax:

```
SUBMITTIMEOUT={timeout_seconds|60}
```

Value: For the *timeout_seconds* variable, enter a number from 1 through 86400. Default is 60.

Usage Notes:

- By default, a PowerExchange Listener waits for 60 seconds for the spawned jobs to start. If this time expires, PowerExchange times out the job, stops the task in the PowerExchange Listener, and writes the PWX-00426 message to the PowerExchange message log.
- This statement applies to all batch jobs that a PowerExchange Listener spawns, which includes the following:
  - Netport jobs
  - DB2 LOAD utility jobs
  - CA IDMS/DB metadata jobs
  - Jobs submitted using the PROG=SUBMIT option of the DTLREXE utility

SVCNODE Statement

The SVCNODE statement specifies the TCP/IP port on which a PowerExchange process listens for commands. Process types include the PowerExchange Listener, PowerExchange Condense, PowerExchange log-based or table-based ECCRs, and the PowerExchange Logger for Linux, Unix and Windows.

You can issue commands to a PowerExchange process through the following programs:

- Use the infacmd pwx program to issue a command to a process on Linux, UNIX, or Windows.
- Use the pwxcmd program to issue a command to a process on i5/OS, Linux, UNIX, Windows, or z/OS.

Operating Systems: All

Related Statements: CMDNODE, LISTENER, and NODE

Required: No

Syntax:

```
SVCNODE=(service_name
        ,listen_port
)
```

Parameters:

service_name

Required. PowerExchange service name of up to 12 characters. This name must match one of the following values:

- For a PowerExchange Condense process, use the service name that you specified in the CONDENSENAME statement in the CAPTPARM configuration member or file.
- For a PowerExchange Adabas, IDMS, or IMS log-based ECCR process or a Datacom table-based ECCR process on z/OS, use the ECCR name that you specified in the CMDNODE statement in the DBMOVER configuration file.
- For a PowerExchange Logger for Linux, UNIX, and Windows process, use the service name that you specified in the CONDENSENAME statement in the pwxccl.cfg file.

- For a PowerExchange Listener process, use the name that you specified for the *listener_node* parameter in the LISTENER statement in the DBMOVER configuration file.

  To issue infacmd pwx commands to connect to the Listener through the Listener application service, this name must match one of the following values:

  - If you created the application service through Informatica Administrator, use the service name that you specified in the **Start Parameters** property.
  - If you created the application service through the infacmd pwx CreateListenerService command, use the name that you specified for the *service_name* parameter of the -StartParameters option on the command.

To issue infacmd pwx commands to connect to the PowerExchange Logger for Linux, UNIX, and Windows through the Logger application service, ensure that the pwxccl.cfg file that the Logger Service uses includes the corrrect value for CONDENSENAME. When you create the Logger Service, you can optionally specify a configuration file different from the default pwxccl.cfg file in the installation directory, as follows:

- If you created the application service through Informatica Administrator, specify the *cs* value in the **Start Parameters** property.
- If you created the application service through the infacmd pwx CreateLoggerService command, specify the the *cs* value in the -StartParameters option on the command.

listen_port

Required. Unique port number on which the PowerExchange Listener, PowerExchange log-based or table-based ECCR, PowerExchange Logger for Linux, UNIX, and Windows, or PowerExchange Condense listens for commands.

To issue infacmd pwx commands to connect to the Listener or Logger through the Listener or Logger application service, the port number that you specify in the SVCNODE statement must match one of the following values:

- If you created the application service through Informatica Administrator, the value that you specified for the **SVCNODE Port Number** property.
- If you created the application service through the infacmd pwx CreateListenerService or CreateLoggerService command, the value that you specified for the -SvcPort option on the command.

TCPIPVER Statement

The TCPIPVER statement specifies alternative TCP/IP socket code that PowerExchange uses instead of the standard IBM TCP/IP UNIX socket code.

By default, PowerExchange uses IBM TCP/IP UNIX sockets. Define this statement if you use CA TCPAccess.

Operating Systems: z/OS

Required: No

Syntax:

```
TCPIPVER={2|3}
```

Valid Values:

- **2.** PowerExchange uses CA TCPAccess socket code on z/OS instead of IBM TCP/IP UNIX socket code.
- **3.** PowerExchange uses native MVS socket code. Specify this value at the direction of Informatica Global Customer Support.

TRACING Statement

The TRACING statement enables PowerExchange alternative logging and specifies attributes for the alternative log files. PowerExchange uses the alternative log files instead of the default PowerExchange message log file to store messages.

Operating Systems: All

Related Statements: LOGPATH

Required: No

Syntax:

```
TRACING=(PFX=prefix
        [,APPEND={N|Y}]
        [,BUFFERS={number_of_buffers|100}]
        [,FILENUM={number_of_files|5}]
        [,FLUSH={flush_interval|99}]
        [,RECLEN={record_length|80}]
        [,SIZE={log_size|100}]
        [,VIEW={N|Y}]
)
```

Parameters:

PFX=*prefix*

Required. Specifies the prefix for the alternative log file names.

PowerExchange uses the following system-based rules to create the alternative log file names:

i5/OS

PowerExchange uses the PFX value to create the member names of the log files in the PowerExchange data library. The generated log file names vary based on whether the PowerExchange Listener, PowerExchange Condense, or other PowerExchange jobs create the files.

- The PowerExchange Listener uses the following file naming convention:

  datalib/P*listener_port*(*prefixnnn*)

- PowerExchange Condense and other PowerExchange jobs use the following file naming convention:

  datalib/JOB*job_number*(*prefixnnn*)

These naming conventions include the following variables:

- *datalib* is the PowerExchange data library name specified during PowerExchange installation.
- *listener_port* is the PowerExchange Listener port number.
- *job_number* is the i5/OS job number for the tracing subtask, DTLTRTSK, that runs under PowerExchange Condense or other PowerExchange jobs.
- *prefixnnn* is the PFX parameter value with an appended sequential number from 001 through 999.

For example, a PowerExchange Listener that has a listener port number of 2480, a PFX value of PWXLOG, and a FILENUM value of 3 creates the following log files:

```
datalib/P02480(PWXLOG001)
datalib/P02480(PWXLOG002)
datalib/P02480(PWXLOG003)
```

Maximum length for the PFX value is seven characters.

Linux, UNIX, and Windows

PowerExchange uses the PFX value as the subdirectory name in which to place the log files. PowerExchange uses the LOGPATH statement in the dbmover.cfg file to determine the directory in which to place this log subdirectory.

The generated log file names vary based on whether the PowerExchange Listener, PowerExchange Condense, or other PowerExchange tasks create the files.

- The PowerExchange Listener uses the following file-naming convention:

  Linux and UNIX:

  logpath/*prefix*/DTLLST1.p*listener_port*.n*nnnn*.log

  Windows Listener Service:

  *logpath**prefix*\DTLLSTNT.p*listener_port*.n*nnnn*.log

  Windows Listener:

  *logpath**prefix*\DTLLST1.p*listener_port*.n*nnnn*.log

- The PowerExchange Logger for Linux, UNIX, and Windows uses the following file-naming convention:

  Linux and UNIX:

  logpath/*prefix*/PWXCCL.t*yyyymmddhhmmss*.p*pid*.n*nnnn*.log

  Windows:

  *logpath**prefix*\PWXCCL.t*yyyymmddhhmmss*.p*pid*.n*nnnn*.log

- For other tasks, PowerExchange uses the following file-naming convention:

  Linux and UNIX:

  logpath/*prefix*/*module*.t*yyyymmddhhmmss*.p*pid*.n*nnnn*.log

  Windows:

  *logpath**prefix**module*.t*yyyymmddhhmmss*.p*pid*.n*nnnn*.log

The variables represent the following values:

- *logpath* is the value of the LOGPATH statement in the dbmover.cfg file.
- *prefix* is the PFX parameter value.
- *module* is the name of the PowerExchange module that is running, such as DTLURDMO for that utility or DTLODBCDRVR for PowerCenter operations.
- *listener_port* is the PowerExchange Listener port number.
- *yyyymmddhhmmss* is the time stamp when the file was created.
- *pid* is the process ID of the PowerExchange task.
- *nnn* is a sequential number from 001 through 999.

For example, a PowerExchange Listener that runs on UNIX with a port number of 2480, a PFX value of PWXLOG, and a FILENUM value of 3 creates the following log files:

```
logpath/PWXLOG/DTLLST1.p02480.n001.log
logpath/PWXLOG/DTLLST1.p02480.n002.log
logpath/PWXLOG/DTLLST1.p02480.n003.log
```

Maximum length for the PFX value is 210 characters.

z/OS

PowerExchange uses the PFX value as the high-level qualifier or qualifiers (HLQ) for dynamically allocated alternative log data sets. These data sets are sequential data sets. Alternatively, you can

specify DTLLOG*nn* DD statements in the JCL for a PowerExchange task to allocate the alternative log data sets. By default, PowerExchange uses dynamically allocated alternative log data sets.

If you use a DTLLOG*nn* DD statement that allocates an extended sequential data set, PowerExchange writes only one message on each track. If the DD statement allocates a normal sequential data set, PowerExchange writes one message to each data block.

Note: Do not use DFSMS compression for alternative log data sets.

For dynamically allocated log data sets, the generated data set names vary based on whether the PowerExchange Listener or other PowerExchange batch jobs or started tasks create the files.

- The PowerExchange Listener uses the following file naming convention:

  prefix.*sysid*.P*listener_port*.N*nnn*

- All other PowerExchange batch jobs and started tasks use the following file naming convention:

  prefix.*job_name*.*job_number*.*sysid*.N*nnn*

The variables represent the following values:

- *prefix* is the high-level qualifier or qualifiers that you specify in the PFX parameter. Maximum length of the entire prefix is 16 characters.
- *sysid* is the system ID of the z/OS system on which the batch job or started task runs.
- *listener_port* is the PowerExchange Listener port number.
- *job_name* is the job name of the batch job or started task.
- *job_number* is the JES job number, which begins with JOB for batch jobs and STC for started tasks.
- *nnn* is a generated sequential number from 001 through 999.

For example, a PowerExchange Listener that runs on the MVS1 system with a port number of 2480, a PFX value of PWXLOG, and a FILENUM value of 3 creates the following log files:

```
PWXLOG.MVS1.P02480.N001
PWXLOG.MVS1.P02480.N002
PWXLOG.MVS1.P02480.N003
```

APPEND={N|Y}

Optional. Controls how PowerExchange uses the message log files when the PowerExchange component that issues messages is restarted.

Options are:

- **N**. PowerExchange opens a new log file or overwrites the oldest log file as the log file.

  For example, if you set FILENUM=3 to use three dynamically allocated log files, when the PowerExchange Listener starts, it tries to open log file 1, 2, and 3, in that order. Then, PowerExchange completes one of the following actions:

  - If one or more of the dynamically allocated log files do not exist, PowerExchange uses the first nonexistent log file as the initial log file. For example, if log files 1 and 2 exist but log file 3 does not exist, PowerExchange uses log file 3 as the initial log file. If no log files exist, PowerExchange uses log file 1 as the initial log file.
  - If all three log files exist, PowerExchange uses the oldest log file as the initial log file, completely overwriting it.

  Note: If you use a GDG on z/OS for alternative logging, PowerExchange creates a new generation when the PowerExchange Listener starts.

- **Y**. PowerExchange opens the most recent log file, if one exists, and appends log messages to the end of it. If no log files exist, PowerExchange opens a new log file.

  For example, if you set FILENUM=3 to use three log files, when the PowerExchange Listener starts, it tries to open log file 1, 2, and 3, in that order. Then, PowerExchange completes one of the following actions:

  - If one or more log files exist, PowerExchange opens the most recent log file and appends log messages to the end of it.

    If you use a GDG for alternative logging on z/OS and specify GDG(0) in the DTLLOG*nn* DD statement of the PowerExchange Listener JCL, PowerExchange appends messages to the end of the current generation. If you do not use GDG(0), PowerExchange ignores this parameter.

  - If no log files exist, PowerExchange opens a new log file 1 and uses it as the log file.

Default is Y.

BUFFERS={*number_of_buffers*|100}

Optional. Specifies the number of buffers that PowerExchange allocates to receive message and trace information from PowerExchange subtasks. If the buffer space is full, the PowerExchange subtasks that generate message and trace information wait until buffer space is available. PowerExchange programs use this buffer space internally.

Valid values are 5 through 9999. Default is 100.

Specify this parameter only at the direction of Informatica Global Customer Support.

FILENUM={*number_of_files*|5}

Optional. Specifies the number of alternative log files that PowerExchange creates and uses when the log files are dynamically allocated. When a log file becomes full, PowerExchange switches to the oldest alternative log file and overwrites it.

Valid values are from 1 through 99. Default is 5.

Note: On z/OS, the FILENUM parameter is ignored if you use a GDG for alternative logging or if you specify DTLLOG*nn* DD statements in the JCL for a PowerExchange component that issues DTLLOG messages. For a GDG, the parameter is ignored regardless of whether you specify GDG(0) or GDG(+1) in the DTLLOG*nn* DD statement in the JCL.

FLUSH={*flush_interval*|99}

Optional. Specifies the number of log records that PowerExchange collects before it flushes them to the log file on disk. PowerExchange must periodically flush log records to enable PowerExchange to recover from out-of-space conditions. Low flush values result in more I/O activity to the log file.

Valid values are 1 through 99. Default is 99.

RECLEN={*record_length*|80}

Optional. Specifies the record length that PowerExchange uses to write log records to the log file. PowerExchange writes the log record on multiple lines if the length of the message exceeds the record length.

Valid values are 80 through 255. Default is 80.

Note: If you do not specify the RECLEN parameter and if you enter a value greater than 80 in the LOG_LINE_LIMIT statement, PowerExchange uses the LOG_LINE_LIMIT value as the RECLEN value.

SIZE={*log_size*|100}

Optional. Specifies the approximate amount of log data, in megabytes, that PowerExchange writes to an alternative log file. After PowerExchange reaches this value, it closes the current log file and opens the next log file to continue writing log records.

Valid values are 1 through 2048. Default is 100.

Note: On z/OS, if a manually allocated data set is larger than the SIZE value, PowerExchange limits the amount of log data that it writes to the data set to the SIZE value. If the data set is smaller than the SIZE value, the data set size limits the amount of log data that PowerExchange can write to it. When an out-of-space condition occurs, PowerExchange switches to the next manually allocated message data set.

VIEW={N|Y}

Optional. Controls whether PowerExchange periodically closes and reopens the current log file when the FLUSH interval expires. You can specify this parameter on all operating systems, but it is most useful on z/OS. On z/OS, you cannot view the alternative message log records until the log data set is closed. On operating systems other than z/OS, you can view the log records after PowerExchange flushes the log records to disk based on the FLUSH interval. The current log file does not need to be closed for you to view the log records.

Options are:

- **N**. PowerExchange does not periodically close and reopen the current log file.
- **Y**. PowerExchange periodically closes and reopens the current log file.

  Tip: On z/OS, Informatica recommends that you specify VIEW=Y to periodically close and reopen the alternative log data set so that you can view the log records.

  If you specify VIEW=Y on z/OS, the following considerations apply:

  - If you use a GDG for alternative logging on z/OS, you must specify GDG(0) and DISP=SHR in a single DTLLOG*nn* DD statement in the PowerExchange Listener JCL. Also, allocate and create at least one generation of the GDG data set before starting the PowerExchange Listener.
  - If you use third-party products that manipulate data set allocations, these products might interfere with VIEW=Y processing. For example, the products might change the SYSDSN ENQ to EXCLUSIVE mode, which prevents you from viewing the data set.
  - The performance of the PowerExchange job that writes data to the alternative log data set might be degraded because of frequent data set open and close requests. Use the default value of 99 for the FLUSH parameter to minimize performance degradation.

Default is N.

Usage Notes:

- Use alternative logging to improve logging performance and to customize the amount of data logged for long-running jobs, such as a PowerExchange Logger for Linux, UNIX, and Windows process that runs in continuous mode.
- When dynamic alternative logging is enabled, PowerExchange creates a set of alternative log files for each PowerExchange process in a separate directory.

  You can specify the location, the number of log files, and the log file size in megabytes. When a log file reaches the specified size, PowerExchange switches to the next log file and begins overwriting any data in that file.
- If you define the TRACING statement, also define the LOGPATH statement to specify a directory for the alternative log files on a Linux, UNIX, or Windows system.

- PowerExchange dynamically allocates the alternative log data sets unless you define DTLLOG*nn* DD statements in the JCL for a PowerExchange job or started task.
- On z/OS, Informatica recommends that you specify SYSOUT=* in a DTLLOG01 DD statement that you use in the JCL for all PowerExchange jobs and started tasks that issue messages, for example:

  ```
  //DTLLOG01 DD SYSOUT=*
  ```

  This strategy simplifies configuration because you define only one DTLLOG01 DD statement with a single SYSOUT option. Also, this strategy makes finding message output for a particular execution of a job or task easier because PowerExchange writes all of the message output to a single SYSOUT data set, which is available with the other job output.
- If you use a GDG on z/OS for the alternative logging, specify GDG(0) in the DTLLOG*nn* DD statement of the PowerExchange Listener JCL. For example:

  ```
  DTLLOG01 DD   DSN=USER1.V901.TRCGDG(0),DISP=SHR
  ```

  By using GDG(0), you can use APPEND=Y to resume logging messages to the current generation. Also, you can use VIEW=Y to view log records in the GDG while the PowerExchange Listener task is active. If you specify APPEND=N, PowerExchange creates a new generation when the PowerExchange Listener starts.

  If you use GDG(+1) instead, PowerExchange ignores the APPEND and VIEW parameters on the TRACING statement and creates a new generation whenever the PowerExchange Listener starts.

  Also, when using a GDG, allocate and create at least one generation of the GDG before starting the PowerExchange Listener.
- On z/OS, if you use a GDG for alternative logging or specify a DTLLOG01 DD statement in the JCL for a PowerExchange job or started task, the FILENUM parameter is ignored.

GDG Example:

To append messages to the current generation of a GDG on z/OS, GDG(0), and be able to view the messages periodically, complete the following steps:

1. Allocate and create at least one generation data set in the GDG by running a batch job that contains JCL statements such as:

   ```
   //DJEGDG@  JOB  (ACCOUNT),'GDG',NOTIFY=&SYSUID
   //JSTEP01 EXEC PGM=IDCAMS
   //SYSPRINT DD SYSOUT=*
   //SYSIN DD *
   DEFINE GDG-
   (NAME(USER1.V901.TRCGDG)-
         LIMIT(4)-
         NOEMPTY-
         SCRATCH)
   //STEP2   EXEC PGM=IEFBR14
   //DLLOG01  DD DSN=USER1.V901.TRCGDG(+1),DISP=(NEW,CATLG),
                    DCB=(BLKSIZE=32718,LRECL=132,RECFM=VB),
                    SPACE=(CYL,(1,1))
   ```

2. In the DBMOVER configuration member, define a TRACING statement that includes VIEW=Y and APPEND=Y. For example:

   ```
   TRACING=(PFX=PWX,APPEND=Y,VIEW=Y)
   ```

   This statement configures PowerExchange to append messages to the end of the current generation of the GDG and to periodically close and reopen the generation data set to make the messages available for viewing.
3. Configure the DTLLOG*nn* DD statement in the PowerExchange Listener JCL to reference the current generation data set of the GDG, for example:

   ```
   DTLLOG01  DD DSN=USER1.V901.TRCGDG(0),DISP=SHR
   ```

   Use DISP=SHR in this statement.

4. Start the PowerExchange Listener.

   PowerExchange begins appending messages at the end of the current generation data set.

   If the current generation does not exist, a JCL error occurs and the PowerExchange Listener does not start.

Configuring the TCP/IP Port Number

During PowerExchange installation, the default TCP/IP port number for the PowerExchange Listener is set to 2480 in the DBMOVER configuration file:

```
LISTENER=(node_name,TCPIP,2480)
```

TCP/IP port 2480 is registered with the Internet Assigned Numbers Authority (IANA) exclusively for PowerExchange use. Ask your network administrator to verify that this port is available for PowerExchange at your site.

If another product or a previous PowerExchange installation is using port 2480, you must get another valid TCP/IP port number from your network administrator. Enter that port number in the LISTENER statement of the DBMOVER configuration file.

If you need multiple ports for multiple PowerExchange Listeners, ask your network administrator to provide additional valid port numbers for PowerExchange use. Also, refer to the IANA Web site to determine if the ports have been registered for use by other products.

Tip: Use the IANA information as a general guide. A port that is not registered with IANA still might be used by a product.

After you identify the port numbers, add a LISTENER or NODE statement for each one in the DBMOVER configuration file. The NODE statement applies only to a PowerExchange Listener on a remote platform from which PowerExchange reads data. In each statement, include the IP address or host name of the source platform, for example:

```
NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
```

You can use the HOSTENT utility to determine the IP address or host name of the source. You can also use this utility to verify that the IP address matches the one in the PowerExchange license key file.

Configuring Multiple Log Files

By default, PowerExchange writes log messages and trace information to a single log file. Although you can easily search for error messages in a single file, the log file can become full, especially if you have long-running jobs such as the PowerExchange Listener. To resolve an out-of-space condition, you usually need to stop PowerExchange.

To avoid PowerExchange outages that result from an out-of-space log file and to make logging more efficient, you can use PowerExchange alternative logging. To implement alternative logging, you must include the TRACING statement in the DBMOVER configuration file. This statement enables PowerExchange to write message and trace information to a series of reusable log files that have a generated numeric suffix. When a reusable log file becomes full, PowerExchange begins writing information to the next log file in the sequence based on the numeric suffix.

On z/OS systems, PowerExchange includes the TRACING statement in the default DBMOVER configuration member. Also, the default JCL for PowerExchange jobs contains the following statement for allocating log data sets:

```
DTLLOG01 DD SYSOUT=*
```

If this default logging setup is not sufficient, you can use any of the multiple logging strategies that are available for z/OS.

To configure multiple log files:

- In the DBMOVER configuration file, enter the TRACING statement. Use the following syntax:

  ```
  TRACING=(PFX=prefix,[APPEND=Y|N,][BUFFERS=nnn,][FILENUM=nnn,][FLUSH=nn,][RECLEN=nnn,]
  [SIZE=nnnn,][VIEW=Y|N])
  ```

  This syntax includes all available parameters. The parameters that you use depend on logging and tracing requirements and the system.

Specifying an Alternative Configuration File or License Key File

When you start the PowerExchange Listener on an i5/OS, Linux, UNIX, or Windows system, you can include the CONFIG and LICENSE statements in the start command. These statements point to alternative configuration and license key files that you want to use for the PowerExchange Listener execution instead of the default DBMOVER configuration and license.key files. Usually, you create the alternative files by copying the default files to a different location or under a different file name and then customizing the copies. This practice prevents your customized configuration and license key files from being overwritten when you upgrade or reinstall PowerExchange.

On a Linux, UNIX, or Windows system, you can also specify alternative configuration and license key files in the PWX_CONFIG and PWX_LICENSE environment variables. If you do so, the alternative files identified in the CONFIG and LICENSE statements override the alternative files identified in the environment variables for the current PowerExchange Listener execution.

Configuring the PowerExchange Listener JCL on z/OS

PowerExchange provides PowerExchange Listener JCL in the following members of the RUNLIB library:

- PWXLSTNR runs PowerExchange as a started task.
- STARTLST runs PowerExchange as a batch job.

Note: When you ran the XIZZZ998 job during installation, the PWXLSTNR member was copied to the PROCLIB library.

Review the PowerExchange Listener JCL before starting the PowerExchange Listener the first time.

To configure the PowerExchange Listener JCL on z/OS:

1. Verify that the STEPLIB DD concatenation in the PowerExchange Listener JCL includes the load libraries for your source DBMSs.

   The STEPLIB concatenation in the STARTLST member includes the load libraries for all data sources in a proper order.
2. Verify that the PARM statement specifies the node name that is defined in the LISTENER statement of the DBMOVER configuration file.
3. In the REGION parameter of the EXEC statement, enter a region size. The region size indicates the amount of storage to allocate for running the PowerExchange Listener.

   The sample JCL specifies a region size of 384 MB in the EXEC statement.

   Informatica recommends a region size of 0M, which indicates that z/OS gives the job all the storage available below and above 16 megabytes. For important considerations about specifying a region size of 0M, see the *z/OS MVS JCL Reference*.

   If you omit the REGION parameter, the system default applies.
4. Verify that the TIME parameter in the EXEC statement is set to NOLIMIT. The NOLIMIT setting enables the PowerExchange Listener job to run without a time limit and prevents timeout-related abends that might occur with abend code S322.
5. If the SECURITY statement in the DBMOVER configuration file is set to 1 or 2, APF-authorize the PowerExchange LOADLIB library and any other load libraries in the STEPLIB DD concatenation of the PowerExchange Listener JCL.

Configuring PowerExchange TCP/IP Sockets for CA TCPAcess

PowerExchange uses the IBM TCP/IP software by default. However, on a z/OS system, you can optionally configure PowerExchange to use CA TCPAcess software. PowerExchange supports TCPAccess version 5.3 or later.

Note: If you upgrade TCPAcess from version 5.3 to 6.0, you do not need to change PowerExchange configuration.

To configure PowerExchange TCP/IP Sockets for CA TCPAccess:

1. Edit the PowerExchange Listener JCL as follows:
   - Add the TCPAccess LOAD and LINK libraries to the STEPLIB DD concatenation.
   - Add the following DD statement:

     ```
     //SYSTCPD  DD  DISP=SHR,hlq.TCPIP.DATA
     ```
2. Add the TCPAcess LOAD and LINK libraries to one of the following members of the PowerExchange RUNLIB library:
   - STARTLST if you run the PowerExchange Listener as a batch job
   - PWXLSTNR if you run the PowerExchange Listener as a started task
3. Add the TCPAccess LOAD and LINK libraries to the STEPLIB DD concatenation of any PowerExchange jobs that communicate with netport jobs.
4. Add the following statement to the DBMOVER configuration member in the RUNLIB library:

   ```
   TCPIPVER=2
   ```

Environment Variable Incompatibilities Between PowerExchange and PowerCenter

When PowerCenter® and PowerExchange are installed on the same Linux, UNIX, or Windows machine, in certain cases, they have conflicting requirements for the PATH and LD_LIBRARY_PATH environment variables. To run correctly in these cases, PowerExchange and PowerCenter must run in separate environments.

This requirement applies when the PowerCenter Integration Service or PowerCenter Repository Service runs on the same machine as one of the following PowerExchange components:

- PowerExchange Listener
- PowerExchange Logger for Linux, UNIX, and Windows
- PowerExchange Navigator
- Any PowerExchange utility except the createdatamaps utility

The following table describes the restrictions that apply to the PATH and LD_LIBRARY_PATH variables in the PowerExchange and PowerCenter environments:

Environment	PATH	LD_LIBRARY_PATH
PowerExchange	$INFA_HOME must not precede $PWX_HOME. Otherwise, you cannot start the PowerExchange Listener or Logger from the command line.	LD_LIBRARY_PATH must not contain an entry for PowerCenter. This requirement ensures that PowerExchange utilities pick up their libraries from $PWX_HOME only.
PowerCenter	The $PWX_HOME entry must not precede the $INFA_HOME entry.	The $LD_LIBRARY_PATH variable definition must include both $INFA_HOME and $PWX_HOME, and $INFA_HOME must precede $PWX_HOME. For example: `$INFA_HOME/server/bin:$PWX_HOME:$LD_LIBRARY_PATH`

To set the correct environment for PowerExchange or PowerCenter instances on the same machine, use one of the following strategies:

- Always start PowerExchange and PowerCenter using separate user accounts, and set the environment variables appropriately for each account.
- Run the pwxsettask.sh or pwxsettask.bat script each time you start a PowerExchange component.

Starting the PowerExchange Listener

Use the start command that is valid for the platform on which the PowerExchange Listener runs.

Starting the PowerExchange Listener on i5/OS

On i5/OS, use the SBMJOB command to invoke the PowerExchange Listener DTLLST program.

Before starting the PowerExchange Listener, verify that the following prerequisites are met:

- The QMLTTHDACN system value is set to 1 or 2. For more information about the QMLTTHDACN system value, see the IBM i Information Center for i5/OS.
- The JOBD description includes the ALWMLTTHD(*YES) parameter to allow multiple threads.

Use the following command syntax:

```
SBMJOB CMD(CALL PGM(dtllib/DTLLST) PARM('node_name' '[CONFIG=library/
file(myconfig_member)]' '[LICENSE=library/file(mylicense_key_member)]') JOB(job_name)
JOBD(datalib/DTLLIST) PRTDEV(*JOBD) OUTQ(*JOBD) CURLIB(*CRTDFT) INLLIBL(*JOBD)
```

This syntax contains the following variables:

- *dtllib* is the name of the PowerExchange software library that was entered at installation.
- *node_name* is the PowerExchange Listener node name that was specified in the LISTENER statement of the *datalib*/CFG(DBMOVER) configuration member.
- *job_name* is the name of the PowerExchange Listener job or started task.
- *datalib* is the user-specified name for the PowerExchange data library that was entered at installation.

You can enter the SBMJOB command at the command line.

Alternatively, you can execute the SBMJOB command by using an automated scheduler, a CL program, or a REXX procedure. For example, include the SBMJOB command in a REXX member named STARTLST and then use the following statement to start the PowerExchange Listener:

```
STRREXPRC SRCMBR(STARTLST) SRCFILE(datalib/REXX)
```

Note: You cannot use the pwxcmd program to start the PowerExchange Listener.

Starting the PowerExchange Listener on Linux and UNIX

Run the PowerExchange Listener dtllst program from the command prompt:

```
dtllst node_name [config=directory/config_file] [license=directory/license_key_file]
```

Include the optional config and license parameters if you want to specify configuration and license key files that override the original dbmover.cfg and license.key files.

You can add an ampersand (&) at the end to run dtllst in background mode. Also, you can add the nohup prefix to run dtllist permanently.

Alternatively, use the startlst script that was shipped with PowerExchange. The startlst script deletes the detail.log file and starts the PowerExchange Listener.

Caution: If you run PowerExchange and PowerCenter on the same machine, using the same user account, you must create separate environments for PowerExchange and PowerCenter. To create the appropriate PowerExchange environment and start the PowerExchange Listener, run the pwxsettask.sh script.

Use the following syntax:

```
pwxsettask.sh dtllst node_name ["config=directory/config_file"] ["license=directory/
license_key_file"]
```

The quotation marks are optional.

For more information, see "Environment Variable Incompatibilities Between PowerExchange and PowerCenter" on page 43.

Starting the PowerExchange Listener on z/OS

Issue the standard MVS START (S) command if you running the PowerExchange Listener as a started task:

```
S task_name
```

Alternatively, submit the JCL in the STARTLST member of the RUNLIB library if you are running the PowerExchange Listener as a batch job.

Starting the PowerExchange Listener on Windows

To start the PowerExchange Listener on a Windows system, use one of the following methods:

- Run the PowerExchange Listener as a Windows service by completing one of the following actions:
  - From the Windows Start menu, click **Start** > **Programs** > **Informatica PowerExchange** > **Start PowerExchange Listener**.
  - Use the dtllstsi program to enter the start command from a Windows command prompt:

    ```
    dtllstsi start "service_name"
    ```
- Enter dtllst.

  The syntax is the same as that for Linux and UNIX except that the & and nohup operands are not supported. You must have a product license that allows you to manually run dtllst.

If you run the PowerExchange Listener as an application service in the Informatica domain, enable the PowerExchange Listener Service from the Informatica Administrator tool to start it. For more information, see the *Informatica Application Service Guide*.

Caution: If you run PowerExchange and PowerCenter on the same machine, using the same user account, you must create separate environments for PowerExchange and PowerCenter. To create the appropriate PowerExchange environment and start the PowerExchange Listener, run the pwxsettask.bat script.

Use the following syntax:

```
pwxsettask dtllst node_name ["config=directory/config_file"] ["license=directory/
license_key_file"]
```

The quotation marks are required.

For more information, see "Environment Variable Incompatibilities Between PowerExchange and PowerCenter" on page 43.

Managing the PowerExchange Listener

You can use PowerExchange Listener commands to control PowerExchange Listener processing. This section identifies the commands that you are likely to need.

Stopping the PowerExchange Listener

To stop the PowerExchange Listener, use the CLOSE or CLOSE FORCE command on the system where the PowerExchange Listener runs.

You can issue the CLOSE and CLOSE FORCE commands from the command line. Alternatively, you can issue pwxcmd close or closeforce commands from a Linux, UNIX, or Windows system to a PowerExchange Listener running on any system.

The CLOSE or pwxcmd close command stops the PowerExchange Listener after the following subtasks complete:

- Bulk data movement subtasks
- CDC subtasks, which stop at the next commit of a unit of work (UOW)
- PowerExchange Listener subtasks

The CLOSE FORCE or pwxcmd closeforce command forces the cancellation of all user subtasks and then stops the PowerExchange Listener. This option is useful if you have long-running jobs on the PowerExchange Listener. When you specify the FORCE parameter, PowerExchange performs the following processing:

1. Checks if any PowerExchange Listener subtasks are active.
2. If active subtasks exist, polls the number of active subtasks every second.
3. During this wait period, ends any subtasks that are waiting for TCP/IP network input.
4. Cancels any active subtasks that still remain after 30 seconds elapse.
5. Stops the PowerExchange Listener.

Note: Do not use the MVS CANCEL command. If you cancel a PowerExchange Listener, its port is not available for use until after TCP/IP completes clean up. If you submit another PowerExchange Listener job before TCP/IP completes this clean up, PowerExchange issues an error message. If you stop a PowerExchange Listener on z/OS while an active PowerExchange Navigator session is using it, PowerExchange issues the same error messages as for the MVS CANCEL command. In this case, exit the PowerExchange Navigator and restart the PowerExchange Listener on z/OS.

The syntax of the commands varies by operating system and method of issuing them.

Testing a Remote PowerExchange Listener

To test the connectivity and status of a remote PowerExchange Listener, run the DTLREXE utility with the **ping** program. Configure the NODE statement for the remote PowerExchange Listener in the DBMOVER configuration file.

The syntax for running the utility varies by operating system.

On Linux, UNIX, and z/OS, use the following general syntax:

```
dtlrexe loc=node_name prog=ping [uid=user_id] [pwd=password]
```

On i5/OS, use the following syntax:

```
CALL PGM(DTLREXE) PARM('prog=ping loc=node_name [uid=user_id] [pwd=password]')
```

On z/OS or i5/OS, if the PowerExchange Listener is running with a setting of 1 or 2 in the first parameter of the SECURITY statement, you must include the user ID and password.

On UNIX, if a password includes a dollar ($) sign, DTLREXE receives only the portion of the password up to dollar ($) sign. To resolve this problem, enclose the password in single-quotation marks, which act as escape characters:

```
'pwd=yourpwd'
```

If DTLREXE successfully contacts the PowerExchange Listener, PowerExchange issues the following messages:

```
PWX-00750 DTLREXE Input LOC=node_name, PROG=PING, PARMS=none, UID=user_id.
PWX-00650 Listener 127.127.127.127 -> 127.127.127.127 on port 1234 socket 4
PWX-00591 Tasks now active = 1.
PWX-00753 DTLREXEL Input received LOC=node_name, PROG=PING, PARMS=none.
PWX-00754 DTLREXEL User Return Codes 1=0, 2=0.
PWX-00755 DTLREXE Command OK!
```

Controlling the PowerExchange Listener

You can use PowerExchange Listener commands to list all active tasks and stop specific tasks.

A list of the active PowerExchange Listener tasks includes the following information:

- TCP/IP address
- Port number
- Application name
- Access type
- Status

The following table identifies the commands that list all active PowerExchange Listener tasks and the types of systems from which the commands can be issued:

Command	Systems
DISPLAY ACTIVE	- i5/OS - z/OS - Linux, UNIX, and Windows
LISTTASK	- z/OS - From the PowerExchange Navigator on Windows to a PowerExchange Listener on any system
pwxcmd listtask {-service|-sv} *service*	From a Linux, UNIX, or Window system to a PowerExchange Listener on any system

The following table identifies the commands that stop a PowerExchange Listener task and the types of systems from which the commands can be issued:

Command	Systems
STOPTASK	- All - From the PowerExchange Navigator on Windows to a PowerExchange Listener on any system
pwxcmd stoptask {-service|-sv} *service*	From a Linux, UNIX, or Window system to a PowerExchange Listener on any system

For more information, see the *PowerExchange Command Reference*.

CHAPTER 3

Adabas Bulk Data Movement

This chapter includes the following topics:

Introduction to Adabas Bulk Data Movement

PowerExchange, in conjunction with PowerCenter, can move bulk data from or to an Adabas database.

Because Adabas is a nonrelational database, you must create a data map. PowerExchange uses the data map to access the Adabas data and metadata to create a relational row-type view of the records. PowerExchange requires a relational view to use SQL-type statements to read or write bulk data.

If you run the Adabas source database on a system that is remote from the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service systems, you must run a separate PowerExchange Listener on the remote system. For example, if you run Adabas on z/OS, you must install and configure a PowerExchange Listener on the z/OS system. Also verify that PowerExchange and PowerCenter can communicate with the remote PowerExchange Listener.

Considerations for Adabas Bulk Data Movement

Review the following considerations before implementing bulk data movement for an Adabas source or target.

- PowerExchange imports Long Alpha (LA) fields with a default length of 1,024 bytes. You can override this default length from the PowerExchange Navigator by editing the data map. Open the **Record** window of an Adabas file and then open the **Field Properties** dialog box for the LA field. In the **Length** field, you can enter an override value of up to 16,381.
- If the Adabas field definition table (FDT) for an Adabas source or target is password-protected, the Adabas FDT password is required to connect to the Adabas source or target during a PowerCenter batch session. To enter the Adabas FDT password, edit the session in Task Developer. On the **Mapping** tab of the **Edit Tasks** dialog box, under **Sources** or **Targets**, click the Adabas source or target definition. In the right pane under **Properties**, enter the FDT password in the **ADABAS Password** attribute.

- If you use Adabas 8.2.2 or later on z/OS, PowerExchange supports Adabas files that contain spanned records as data sources. A spanned record is a logical record that is composed of a single physical primary record and up to four second secondary records. Each record is stored in a separate data storage block.
- If an Adabas descriptor field has the null suppression (NU) attribute, a PowerCenter workflow that accesses the field might return unexpected results. A descriptor field with the NU attribute is excluded from indexes. If a workflow attempts to perform a write based on the key field value, a SQLCODE=256 failure code or other error condition might result.

Configuring Adabas Bulk Data Movement

To configure PowerExchange for bulk data movement operations, the PowerExchange administrator completes the following tasks:

- Set up and test connectivity to the PowerExchange Listener on the Adabas z/OS system.
- Configure Adabas bulk data movement and optimize Adabas read operations by defining Adabas-specific statements in the DBMOVER configuration member.
- Set up PowerExchange Listener access to the Adabas LOAD library.
- Verify that the PowerExchange Listener on the z/OS system uses the *nonreusable* versions of the following Adabas load modules: ADARUN, ADAIOR, ADAIOS, ADAMLF, ADAPRF, and ADALNK.
- Override the default SVC parameter value for the ADARUN control statement, if necessary.
- Configure PowerExchange to read an Adabas database that is encrypted with a cipher code, if applicable.

Setting Up and Testing Connectivity to a Remote Adabas Source or Target

To access an Adabas data source or target on a remote z/OS system, PowerExchange must be able to communicate with the PowerExchange Listener on the remote system.

To set up and test connectivity to a remote Adabas source or target:

1. On the systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the **ping** command to test network access to the remote system with the Adabas source or target.
2. Add the NODE statement to the dbmover.cfg file on each PowerExchange and PowerCenter system to define the remote PowerExchange Listener and its port:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   Tip: To determine the port number, consult with the person in your organization who installed the PowerExchange Listener on the remote system.
3. Run the DTLREXE ping utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Adding Adabas-Specific Statements to the DBMOVER Configuration Member

To configure Adabas bulk data movement and to optimize Adabas read operations, you can include the following Adabas-specific statements in the DBMOVER configuration file on your UNIX, Windows, or z/OS system:

- ADA_L3_ALLOW
- ADABAS_DEFAULT_DBID
- ADABAS_MU_SEARCH
- ADABAS_PREFIX
- ADABASCODEPAGE
- ADAOPT
- ADAPREFETCH
- ADASTATS
- START_UP_USER_EXIT

After editing the DBMOVER configuration file, you must restart the PowerExchange Listener for your changes to take effect.

For descriptions of all DBMOVER statements, see the *PowerExchange Reference Manual*.

ADA_L3_ALLOW Statement

The ADA_L3_ALLOW statement is an Adabas optimization statement that controls whether PowerExchange uses Adabas L3 commands to read records from a file in logical sequence by descriptor value.

Operating Systems: z/OS

Data Sources: Adabas

Related Statements: ADAOPT and ADASTATS

Required: No

Syntax:

```
ADA_L3_ALLOW={N|Y}
```

Valid Values:

- **N.** PowerExchange attempts to determine if the Adabas version that is running supports beginning and ending ranges. If PowerExchange determines that the Adabas version supports ranges, PowerExchange uses L3 commands. Otherwise, PowerExchange uses L2 commands.
- **Y.** If you specify Adabas descriptors on the WHERE clause of an SQL SELECT statement, PowerExchange uses L3 commands to read records in logical order from the Adabas file. If the SQL statement does not contain a WHERE clause, PowerExchange uses L2 commands to read records from the file in the physical order in which Adabas stored them.

  PowerExchange does not verify that the Adabas version in use supports beginning and ending descriptor key ranges with L3 commands.

  Specify Y if PowerExchange cannot correctly determine the running Adabas version.

Default is N.

Usage Notes:

- To use L3 commands, you must also specify Y for the ADAOPT statement.
- Unlike the ADAOPT statement, the ADA_L3_ALLOW statement does not cause PowerExchange to verify that Adabas version 7 or later is running. Use the ADA_L3_ALLOW statement when user modifications to Adabas prevent PowerExchange from verifying which Adabas version is installed.
- By default, PowerExchange selects **Ranges Only** in the **Optimization Level** list in data maps. If you select **OFF** in the **Optimization Level** list, PowerExchange ignores the ADA_L3_ALLOW statement and does not use L3 commands when processing data for that data map.
- If you specify Y for the ADASTATS statement, PowerExchange writes message PWX-02196 in the PowerExchange message log file. This message indicates whether you can use L3 commands and the status of optimization.

ADABAS_DEFAULT_DBID Statement

The ADABAS_DEFAULT_DBID statement specifies the DBID value that PowerExchange uses when a data map specifies 0 in the **Database ID** property and the bulk data movement session does not specify an override value.

Operating Systems: z/OS

Data Sources: Adabas

Required: No

Syntax:

```
ADABAS_DEFAULT_DBID={dbid|0}
```

Value: For the *dbid* variable, enter a number from 0 through 65535. Default is 0.

Usage Notes:

- You can include leading zeros in the DBID value. For example, to specify 100 as the default DBID, you can define any of the following statements:
  - ADABAS_DEFAULT_DBID=100
  - ADABAS_DEFAULT_DBID=0100
  - ADABAS_DEFAULT_DBID=00100
- To ease migration of data maps from one PowerExchange environment to another, enter 0 in the **Database ID** property of the data map. Then, define an ADABAS_DEFAULT_DBID statement in each environment to provide the DBID value. Alternatively, to override the DBID value specified in a data map, set a value in the **Database Id Override** attribute of the bulk data movement session in PowerCenter or, if you use ODBC, in the ODBC parameters.

ADABAS_MU_SEARCH Statement

The ADABAS_MU_SEARCH statement specifies whether Adabas multi-value (MU) fields are included in search functions. An MU field is a single field that occurs a variable number of times in a record. The MU field name is prefixed with a $ symbol and is allowed in searches. The MU field must be an Adabas descriptor field. It appears as type MU in an Adabas record description.

Operating Systems: z/OS

Data Sources: Adabas

Required: No

Syntax:

```
ADABAS_MU_SEARCH={Y|N}
```

Valid Values:

- **Y.** PowerExchange uses multi-value Adabas descriptor fields in searches. If you specify a multi-value field in a WHERE clause of an SQL SELECT statement, PowerExchange scans the field for values specified in the search.
- **N.** PowerExchange will not include Adabas multi-value fields in searches.

Default is N.

Usage Notes:

- In an existing data map definition, the field name for a multi-value descriptor (DE) field can be used by adding the $ prefix to the multi-value field name and eliminating the subscripting index.
- Limit search criteria that includes multi-value fields to an EQUAL comparator or a BETWEEN phrase. Complex searches using an OR comparator or non-descriptor search values are not supported. The following SELECT statements show simple search criteria for multi-value fields:

```
SELECT * from table_name where $MU_field = 'JONES'
SELECT * from table_name where $MU_field >=  10 and $MU_field <= 20
SELECT * from table_name where $MU_field is between 10 and 20
SELECT AA_field, AB_field, $MU_field where $MU_field = 10
```

- The multi-value field can also be used in a SQL SELECT statement if the field was used in the search criteria. The SQL SELECT statement presents the contents of the multi-value array data that meets the search criteria.
- If you set the ADABAS_MU_SEARCH statement to Y, make sure that the ADAOPT statement is also set to Y to use L3 commands to read records from file in logical sequence by descriptor value. If the search cannot be optimized, and there are OR criteria in the SQL, the search will fail.

ADABAS_PREFIX Statement

The ADABAS_PREFIX statement specifies the prefix that PowerExchange uses to construct a user ID to access Adabas files.

Operating Systems: z/OS

Data Sources: Adabas

Related Statements: ADAUSER, for netport jobs

Required: No

Syntax:

```
ADABAS_PREFIX={prefix|DTL0}
```

Value: For the *prefix* variable, enter a one- to four-character alphanumeric prefix. To construct the user ID, PowerExchange appends an additional four characters to generate a unique value for each Adabas file that the PowerExchange Listener opens. Default is DTL0.

Usage Notes:

- To access the same Adabas file simultaneously with different user IDs, specify a unique user ID. If each user ID that PowerExchange constructs to access an Adabas file is not unique, one or more PowerExchange Listeners might be unable to access the Adabas file. In this case, the read operation fails with Adabas Response Code 48 Sub Code 8 and PowerExchange message PWX-00416.

- To ensure a unique Adabas user ID when multiple PowerExchange Listeners access the same Adabas file, use the ADABAS_PREFIX statement to specify a different prefix for each PowerExchange Listener. If the user ID is not unique, the first PowerExchange Listener that tries to access the Adabas file succeeds, and the second PowerExchange Listener fails. Also, the PowerCenter session fails.
- If you use netport jobs to access Adabas files, define the ADAUSER statement to ensure a unique user ID for each netport job.

ADABASCODEPAGE Statement

The ADABASCODEPAGE statement specifies the single-byte and multibyte code pages to use for an Adabas database.

Enter up to 20 ADABASCODEPAGE statements in the DBMOVER configuration file.

Operating Systems: z/OS

Data Sources: Adabas

Related Statements: CODEPAGE

Required: No

Syntax:

```
ADABASCODEPAGE=(dbid
               [,single_cp]
               [,multi_cp])
```

Parameters:

dbid

Required. The Adabas database identifier (DBID) to which the code page or code pages apply.

Specify 0 to indicate the Adabas default database.

single_cp

Optional. The name of a single-byte code page.

multi_cp

Optional. The name of a multibyte code page. If the database contains WIDECHAR fields, enter a multibyte code page.

Usage Notes:

- Usually, you specify this statement for databases that have WIDECHAR fields that require a code page other than the default code page or the code page that is set in the CODEPAGE statement.

  When you define a data map, you can override the code pages that this statement specifies. In the data map, specify a code page for a specific source file, a wide-character code page for all WIDECHAR fields, or specific code pages for each field. A field-level code page overrides a data map code page, and a data map code page overrides any code page that this statement or the CODEPAGE statement specify.

ADAOPT Statement

The ADAOPT statement is an Adabas optimization statement that controls whether PowerExchange uses Adabas L3 commands to read records from a file in logical sequence by descriptor value.

Operating Systems: z/OS

Data Sources: Adabas

Related Statements: ADA_L3_ALLOW, and ADASTATS

Required: No

Syntax:

```
ADAOPT={N|Y}
```

Valid Values:

- **N.** PowerExchange disables optimization and uses L2 commands to read records from files.
- **Y.** If you specify Adabas descriptors on the WHERE clause of a SQL SELECT statement, PowerExchange uses L3 commands to read records in logical order from the Adabas file. If the SQL statement does not contain a WHERE clause, PowerExchange uses L2 commands to read records from the file in the physical order in which they were stored by Adabas.

  PowerExchange verifies that the Adabas version that is being used supports beginning and ending descriptor key ranges with L3 commands.

  If you have installed user modifications to Adabas, PowerExchange might be unable to correctly determine the installed version of Adabas. In this case, you must also specify Y for the ADA_L3_ALLOW statement to use L3 commands.

Default is Y.

Usage Notes:

- By default, PowerExchange selects **Ranges Only** in the **Optimization Level** list in data maps. If you select **OFF** in the **Optimization Level** list, PowerExchange ignores this statement and does not use L3 commands when processing data for that data map.
- If you specify Y for the ADASTATS statement, PowerExchange writes message PWX-02196 in the PowerExchange message log file. This message indicates whether you can enter L3 commands and the status of optimization.

ADAPREFETCH Statement

The ADAPREFETCH statement controls whether PowerExchange uses the Adabas prefetch feature to improve performance when reading records.

Operating Systems: z/OS

Data Sources: Adabas

Required: No

Syntax:

```
ADAPREFETCH={N|Y}
```

Valid Values:

- **N.** PowerExchange does not use the Adabas prefetch feature when reading records.
- **Y.** PowerExchange uses the Adabas prefetch feature when reading records to improve read performance.

Default is N.

Caution: On UNIX and Windows, Adabas does not allow prefetch processing with the ACBX interface. If you are using ACBX on one of these systems, and ADAPREFETCH=Y, PowerExchange reverts to ACB calls, which cannot read records larger than 32 KB. If you want to use ACBX on UNIX or Windows, do not specify the ADAPREFETCH=Y statement.

ADASTATS Statement

The ADASTATS statement controls whether PowerExchange writes statistical information about Adabas operations to the PowerExchange message log file.

Operating Systems: z/OS

Data Sources: Adabas

Related Statements: ADA_L3_ALLOW, and ADAOPT

Required: No

Syntax:

```
ADASTATS={N|Y}
```

Valid Values:

- **N.** PowerExchange does not write Adabas statistics messages to the PowerExchange message log file.
- **Y.** PowerExchange writes Adabas statistics messages to the PowerExchange message log file.

  Before reading records, PowerExchange writes messages that contain the following Adabas information:

  - Whether PowerExchange uses prefetch
  - Whether PowerExchange uses L2 or L3 commands to read data

    If PowerExchange uses L3 commands, PowerExchange writes additional messages that contain the key and key values used with L3 commands.
  - The format buffer that PowerExchange uses

  After reading records, PowerExchange issues a message that displays the number of L2 and L3 commands used to read the data.

Default is N.

START_UP_USER_EXIT Statement

The START_UP_USER_EXIT statement specifies the name and programming language of a user-defined exit program that PowerExchange calls each time the PowerExchange Listener starts or shuts down.

Use this statement to enable the PowerExchange Listener to decrypt an Adabas database that is encrypted with an Adabas cipher code. The user exit program that you specify must provide a result set that includes the cipher code and some additional information.

Operating Systems: z/OS

Required: No

Syntax:

```
START_UP_USER_EXIT=(PROGRAM_NAME=program_name,LANGUAGE=language)
```

Parameters:

PROGRAM_NAME=*program_name*

Required. Name of the user exit program.

LANGUAGE=*language*

Required. Programming language in which the user exit program is written. Options are:

- **A**. Assembler language.
- **C**. C language.

Usage Notes:

- You can specify up to ten statements, each pointing to a different exit program. When the PowerExchange Listener starts, the user exit programs are executed in the order in which the associated START_UP_USER_EXIT statements occur in the DBMOVER configuration file. When the Listener shuts down, the user exit programs are executed in reverse order.

Related Topics:

- "Decrypting Adabas Sources that Are Encrypted with a Cipher Code" on page 57

Configuring Access to the Adabas LOAD Library (z/OS)

On a z/OS system, the PowerExchange Listener must be able to access an Adabas LOAD library to move bulk data.

When you ran the z/OS Installation Assistant, you entered an Adabas LOAD library. The z/OS Installation Assistant customized the PowerExchange Listener STARTLST and PWXLSTNR members in the RUNLIB library with the Adabas LOAD library that you entered.

Note: The PowerExchange Listener requires the nonreusable versions of the following Adabas load modules: ADARUN, ADAIOR, ADAIOS, ADAMLF, ADAPRF, and ADALNK.

1. Verify that the STEPLIB DD concatenation in the PowerExchange Listener JCL includes the Adabas LOAD library.

   This JCL is in the STARTLST or PWXLSTNR member of the RUNLIB library.
2. If you APF-authorized the PowerExchange Listener LOADLIB library, also APF-authorize the Adabas LOAD library.

Overriding the Default SVC for the ADARUN Control Statement (z/OS)

On z/OS, the PowerExchange Listener uses the Adabas default SVC number of 249 by default. If you want the PowerExchange Listener to use another SVC number, you must define the override SVC parameter for the ADARUN control statement in a separate member.

Note: The ADARUN control statement defines and starts the Adabas operating environment, starts Adabas utilities, and loads the ADAIOR module, which performs all database I/O.

1. Create a member to hold the override parameter value for the ADARUN control statement.
2. In the new member, define the override SVC parameter that the PowerExchange Listener will use. Use the following syntax:

   ```
   ADARUN SVC=nnn
   ```

   The *nnn* variable is the override SVC number. Valid values are 200-255. Adabas default is 249.
3. Add a DDCARD DD statement n the PowerExchange Listener JCL that points to the override member that you created. For example:

   ```
   //DDCARD DD DSN=library.file(member),DISP=SHR
   ```

For more information about the ADARUN control statement and its parameters, see the Adabas documentation.

Decrypting Adabas Sources that Are Encrypted with a Cipher Code

PowerExchange provides bulk data movement support for Adabas databases that are encrypted with a cipher code.

To enable support for Adabas cipher codes, perform the following actions:

- Write a user exit program in Assembler or C that returns a result set with the information that PowerExchange requires to perform the decryption.
- Include the START_UP_USER_EXIT statement in the DBMOVER configuration file on the PowerExchange Listener machine. This statement identifies the user exit program and programming language.

PowerExchange calls the user exit program each time the PowerExchange Listener starts or shuts down. At Listener startup, the exit program provides information for accessing one or more Adabas databases that are protected by an Adabas cipher code. At Listener shutdown, the exit program cleans up resources that the exit program allocated or used.

PowerExchange stores the cipher code in encrypted format in memory while the PowerExchange Listener is running so that the cipher code is not visible in memory dumps.

The user exit program returns one of the following return codes:

- 0 = The program completed successfully.
- 4 = A failure occurred, but the PowerExchange Listener continues running. The Listener ignores the result set.
- Other value = A failure occurred, and the Listener task terminates.

Informatica recommends that you set the first parameter of the SECURITY statement in the DBMOVER configuration file to 2. This setting ensures that the user ID assigned to the PowerExchange Listener can run with a RACF authorization and authentication that are different from those of the individual data access tasks. By defining security setup in this way, PowerExchange switches the Listener subtask to the RACF user ID that is provided on the PowerExchange Listener request, regardless of whether the request is from the PowerExchange Navigator, a PowerCenter workflow, or an Informatica client tool. All data access requests that are directed to PowerExchange are performed with the RACF authorization of the user account that is making the request.

Sample user exit programs, such as LSUUXADC, are provided in the SRCLIB library on z/OS.

Notes:

- If a netport job has an Adabas source that is encrypted with a cipher code, the user ID under which the netport job runs must have READ permission on resource DTL.LISTENER.AMVALUES in the class that is specified in the RACF_CLASS statement in the DBMOVER configuration file. The netport job communicates with the PowerExchange Listener through AMTSK (listamvalues) to obtain the Adabas cipher.
- If a netport job has an Adabas source that is encrypted with a cipher code, set OUSP to Y in the DBMOVER configuration file.
- If an Adabas source that is encrypted with a cipher code contains spanned records, you must apply SAG fixes AN826117 and AU826076. Otherwise, PowerExchange might encounter problems when it tries to decrypt the spanned records.

Related Topics:

- "START_UP_USER_EXIT Statement" on page 55

User Exit Program Result Set

The user exit program returns a result set that consists of statements separated by semicolons.

For example, the sample user exit program in C that is provided in the SRCLIB library returns the following result set:

```
char results_set_without_terminators[] =
{
  "ADA,DBID=8242,FILENUM=104,ActionFlag=1,ActionValue=12345678;"
  "ADA,DBID=8242,FILENUM=105,ActionFlag=1,ActionValue=12345678;"
  "ADA,DBID=8262,FILENUM=105,ActionFlag=1,ActionValue=12345678;"
  "ADA,DBID=0,FILENUM=0,ActionFlag=1,ActionValue=87654321;"
};
```

Each statement in the result set contains the following comma-separated fields:

- ADA. Required. Identifies the source type to which the user exit program applies. ADA is the only supported value.
- DBID=*nnn*. Optional. The database ID of a database that is protected by an Adabas cipher code. If DBID is not specified or is equal to 0, the cipher code applies to all DBIDs.
- FILENUM=*nnn*. Optional. The file number of an Adabas file that is protected by an Adabas cipher code. If FILENUM is not specified or is equal to 0, the cipher code applies to all FILENUM numbers.
- ActionFlag=*n*. Required. The type of action that the user exit performs. A value of 1 indicates the decryption of data by using an Adabas cipher code.
- ActionValue=*cipher_code*. Required. An Adabas cipher code of up to eight numeric digits.

For example, the following statement identifies a cipher code to be used only for the Adabas database with database ID 83:

```
ADA,DBID=83,ActionFlag=1,ActionValue=12345678
```

The following statement omits the DBID value to indicate that the cipher code is to be used with all Adabas databases on the z/OS system:

```
ADA,ActionFlag=1,ActionValue=12345678
```

Note: In both examples, because no FILENUM value is provided, the cipher code applies to all ciphered Adabas files in the database.

Moving Adabas Bulk Data

Use the following procedure to move Adabas bulk data.

Before you begin, gather the following information:

- TCP/IP address or host name of the host that contains the Adabas database
- Adabas database ID
- Adabas file numbers for the files to be accessed
- Your z/OS user ID and password, if required by the security settings in the SECURITY statement in the DBMOVER configuration file

To move Adabas bulk data:

1. In the PowerExchange Navigator, create a data map.

   Click **ADABAS** in the **Access Method** list. To import metadata about data source records and fields, also select the **Import Record Definitions** option, the **Import Key Fields/FDT** option, or both.

   Tip: If you do not select the **Import Record Definitions** or **Import Key Fields/FDT** option, you can import the metadata later. Open the data map in the Resource Explorer and click **File** > **Import Copybook**.

2. Open the data map in the Resource Explorer, and click the nonrelational “record” or relational “table” view to verify that the metadata was successfully retrieved.

3. If the data map is for a data source, send it to the PowerExchange Listener on the remote Adabas system. In the Resource Explorer, select the data map and click **File** > **Send to Remote Node**. This step enables PowerExchange extraction routines to access the data map at runtime.

4. In the **Location** field, enter the PowerExchange Listener node name that is specified in the NODE statement of the dbmover.cfg file on the local PowerExchange system. Complete the other fields, as needed.

   Note: If you do not complete this step, you are prompted to send the data map to the remote node when you run a database row test.

5. Run a database row test on the table view of the metadata to test that actual data can be returned from the Adabas source database.

6. In the PowerCenter Designer, import the PowerExchange data map and create a mapping.

7. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX NRDB Batch application connection. Then start the workflow to perform the bulk data movement.

CHAPTER 4

Datacom Bulk Data Movement

This chapter includes the following topics:

Introduction to Datacom Bulk Data Movement

PowerExchange can read bulk data from a CA Datacom source on an MVS system. However, PowerExchange cannot write bulk data to a Datacom target.

PowerExchange treats Datacom as a nonrelational DBMS. Consequently, you must create a data map for a Datacom data source from the PowerExchange Navigator. PowerExchange uses the data map to access Datacom source data and metadata to create a relational view of the source records for processing.

Because a Datacom database is on an MVS system, it is remote from the Windows system or systems on which the PowerExchange Navigator, PowerCenter Client, and Power Integration Service run. Therefore, you must run an additional PowerExchange Listener on the remote MVS system and verify that PowerExchange and PowerCenter can communicate with it.

Configuring Datacom Bulk Data Movement

To configure PowerExchange for bulk data movement operations, the PowerExchange administrator must set up and test connectivity to the PowerExchange Listener on the remote MVS system.

Setting Up and Testing Connectivity to a Remote Datacom Source

To access a Datacom data source on a remote z/OS system, PowerExchange and PowerCenter must be able to communicate with the PowerExchange Listener on the remote system.

To set up and test connectivity to a remote Datacom source:

1. On the system or systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the **ping** command to test network access to the remote system with the Datacom database.

2. Add the NODE statement to the dbmover.cfg file on each PowerExchange or PowerCenter system to identify the remote PowerExchange Listener and its port:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   Tip: To determine the port number, consult with the person in your organization who installed the PowerExchange Listener on the remote system.

3. Run the DTLREXE ping utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Moving Datacom Bulk Data

Use the following procedure to perform a Datacom bulk data movement. This procedure assumes that you are using PWXPC to integrate PowerExchange with PowerCenter.

Before you begin, gather the following information:

- TCP/IP address of the host that contains the Datacom source database
- Datacom database name and ID
- Your MVS user ID and password, if required by the PowerExchange SECURITY statement setting

To move Datacom bulk data:

1. In the PowerExchange Navigator, create a data map.

   Click **DATACOM** in the **Access Method** list. To import metadata about records and fields, also select the **Import Record Definitions** option.

   Tip: If you do not select the **Import Record Definitions** option, you can import the metadata later. Open the data map in the Resource Explorer and click **File** > **Import Copybook**.

2. Open the data map in the Resource Explorer, and click the nonrelational "record" or relational "table" view to verify that the metadata was successfully retrieved.

3. Send the data map to the PowerExchange Listener on the remote Datacom system. In the Resource Explorer, select the data map and click **File** > **Send to Remote Node**. This step enables PowerExchange extraction routines to access the data map at runtime.

   In the **Location** field, enter the PowerExchange Listener node name that is specified in the NODE statement of the dbmover.cfg file on the local PowerExchange system. Complete the other fields, as needed.

   Note: If you do not complete this step, you are prompted to send the data map to the remote node when you run a database row test.

4. Run a database row test on the table view of the metadata to test that data can be returned from the Datacom source database.

5. In the PowerCenter Designer, import the PowerExchange data map. Click **Sources > Import from PowerExchange**. Then enter the following required attributes:

   - In the **Location** field, enter the PowerExchange Listener node name that you specified in the NODE statement of the dbmover.cfg file.

- In the **User name** and **Password** fields, enter your z/OS user ID and password, if required by the SECURITY statement setting.
- In the **Source Type** list, select **DATACOM**.

  Complete the optional attributes as needed. Then connect to PowerExchange and import the data map.

6. In the PowerCenter Designer, create a mapping.
7. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX NRDB Batch application connection. Then start the workflow to perform the bulk data movement operation.

CHAPTER 5

DB2 for i5/OS Bulk Data Movement

This chapter includes the following topics:

Introduction to DB2 for i5/OS Bulk Data Movement

PowerExchange, in conjunction with PowerCenter, can move bulk data from or to a DB2 for i5/OS source or target database.

You can use either the PowerExchange DB2 or SEQ database access method. These access methods generally correspond to the DB2 for i5/OS SQL or system file access methods for processing database tables and data. The DB2 access method processes the source or target as relational DB2 objects, whereas the SEQ method processes the source or target as libraries, files, and members. The SEQ method enables you to specify all or selected members for bulk data movement processing.

Depending on the access method that you select, you might not need to create a data map. You can use the following strategies to define the bulk data movement:

- To use the DB2 access method, you can define the bulk data movement entirely from PowerCenter. A data map is not required. From the PowerCenter Designer, import the metadata that PowerExchange reads from the DB2 catalog as a “DB2” source type. Then create a mapping. In the PowerCenter Workflow Manager, create and run a workflow to perform the bulk data movement.

  Alternatively, in the PowerExchange Navigator, you can create a data map that specifies the “DB2” access method. Then import that data map into the PowerCenter Designer as a nonrelational source or target definition. In a data map, you can define user-defined fields and build expressions to populate them.

- To use the SEQ access method, you must create a data map from the PowerExchange Navigator that specifies the “SEQ” access method. In the PowerCenter Designer, import the data map as a nonrelational source or target and create a mapping. From the PowerCenter Workflow Manager, create and run a workflow to perform the bulk data movement. With the SEQ access method, you can process all or some of the members in the database file and use file list processing.

Because the DB2 database runs on an i5/OS system, it is remote from the system on which the PowerExchange Navigator and PowerCenter Integration Service runs. Therefore, you must run an additional PowerExchange Listener on the remote i5/OS system and verify that PowerExchange can communicate with it.

Considerations for DB2 for i5/OS Bulk Data Movement

The following considerations apply to DB2 for i5/OS bulk data movement operations:

- The maximum length of a DB2 for i5/OS record, excluding LOB columns, that you can include in a PowerExchange bulk data movement operation is 32,766 bytes. If the record contains variable length or nullable columns, DB2 adds additional bytes to the record. With these additional bytes, the combined length of all columns still cannot exceed 32,766 bytes. For more information, see the DB2 for i5/OS documentation.
- The maximum length of a DB2 for i5/OS record, including LOB columns, that you can include in a PowerExchange bulk data movement operation is 8 MB. The following restrictions apply to records that contains LOB columns:

  - You cannot access the data by using an NRDB SEQ data map or a DB2 data map.

  - You cannot perform an insert into a target table that contains LOB columns.

  Tip: You can use the SUBSTRING command to reduce the amount of data that PowerExchange reads from a LOB column. Use a statement such as the following one:

  ```
  select SUBSTRING(LOBVALUE,1,900) from myschema.mytable
  ```
- For DB2 for i5/OS bulk data movement operations that use the DB2 access method, PowerExchange uses a DB2 multiple-row FETCH statement to retrieve multiple rows of data at a time from a source table. This feature can help improve performance by reducing the number of database accesses for reading source data. By default, 25 rows are retrieved. In PowerCenter, you can configure the number of rows retrieved by setting the **Array Size** attribute on a PWX DB2i5OS relational connection used by PWXPC.

  PowerExchange dynamically lowers the array size when all the following conditions are true:

  - The database type is DB2.

  - The table contains LOB columns.

  - The **Array Size** value is greater than 1.

  - Row size * **Array Size** is greater than 16000000 bytes.

  If these conditions are met, PowerExchange reduces the array size and logs message PWX-00186 on both the client and PowerExchange Listener machines.
- To propagate the Relative Record Number (RRN) values of records in a DB2 for i5/OS source file, create a data map that has an access method of SEQ and add a user-defined field that is populated by the PowerExchange GetDatabaseKey() expression function. This function populates the user-defined field with the RRN of the source record.

- Bulk data movement supports Row and Column Access Control (RCAC) rules that database administrators can create to control the visibility of sensitive DB2 data. These rules were introduced in DB2 for i5/OS 7.2.

Datatypes Supported for DB2 for i5/OS Bulk Data Movement

PowerExchange supports most DB2 for i5/OS datatypes for bulk data movement.

The following table identifies the DB2 for i5/OS datatypes that PowerExchange supports and does not support for bulk data movement:

DB2 Datatype	Supported for Bulk Data Movement?
BIGINT	Yes
BINARY	Yes
BLOB	Yes (sources only)
CHAR	Yes
CLOB	Yes (sources only)
DATALINK	No
DATE	Yes
DBCLOB	Yes (sources only)
DECFLOAT	No
DECIMAL	Yes
DISTINCT (user-defined)	No
DOUBLE	Yes
FLOAT	Yes
GRAPHIC	Yes
INTEGER	Yes
LONG VARCHAR	Yes
LONG VARGHAPHIC	Yes
NUMERIC	Yes
REAL	Yes

DB2 Datatype	Supported for Bulk Data Movement?
ROWID	No
SMALLINT	Yes
TIME	Yes
TIMESTAMP	Yes
VARBINARY	Yes
VARCHAR	Yes
VARGRAPHIC	Yes

Configuring DB2 for i5/OS Bulk Data Movement

To configure PowerExchange for bulk data movement operations, the PowerExchange administrator completes the following tasks:

- Set up and test connectivity to the PowerExchange Listener on the remote DB2 for i5/OS system.
- Configure DB2 for i5/OS bulk data movement by defining DB2-specific statements in the DBMOVER configuration member.

Setting Up and Testing Connectivity to a DB2 for i5/OS Source or Target

To access DB2 data on a remote i5/OS system, PowerExchange must be able to communicate with the PowerExchange Listener on the remote system. Use the following procedure to set up and test connectivity to the remote i5/OS system.

To set up and test connectivity to a DB2 for i5/OS source or target:

1. On the systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the **ping** command to test network access to the remote system with the DB2 source or target.
2. Add the NODE statement to the dbmover.cfg file on each PowerExchange or PowerCenter system to identify the remote PowerExchange Listener and its port:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   Tip: To determine the port number, consult with the person in your organization who installed the PowerExchange Listener on the remote system.
3. Run the DTLREXE ping utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Adding DB2-Specific Statements to the DBMOVER Configuration Member

To configure DB2 for i5/OS bulk data movement, you can include the following DB2-specific statements in the DBMOVER configuration member on the i5/OS source or target system:

- DB2_BIN_AS_CHAR
- DB2_BIN_CODEPAGE
- DB2_ERROR_FILE
- RDBMSINSRTDFLT

After editing the DBMOVER configuration member, you must restart the PowerExchange Listener for your changes to take effect.

For descriptions of all DBMOVER statements, see the *PowerExchange Reference Manual*.

DB2_BIN_AS_CHAR Statement

The DB2_BIN_AS_CHAR statement controls whether PowerExchange considers data in CHAR and VARCHAR columns that you define with the FOR BIT DATA clause as character data.

To override character data that is incorrectly assigned with CCSID 65535, use this statement in conjunction with the DB2_BIN_CODEPAGE statement.

Operating Systems: i5/OS

Data Sources: DB2 for i5/OS

Related Statements: DB2_BIN_CODEPAGE

Required: No

Syntax:

```
DB2_BIN_AS_CHAR={N|Y}
```

Valid Values:

- **N.** Data in CHAR and VARCHAR columns defined with the FOR BIT DATA clause is binary data.

  PowerExchange does not perform code page translation for binary data.
- **Y.** Data in CHAR and VARCHAR columns defined with the FOR BIT DATA clause is character data.

Default is N.

DB2_BIN_CODEPAGE Statement

The DB2_BIN_CODEPAGE statement defines the single-byte and multibyte CCSID values that PowerExchange uses to process character data in columns that you define with the FOR BIT DATA clause, if you specify Y for the DB2_BIN_AS_CHAR statement.

To override character data incorrectly assigned with CCSID 65535, use this statement in conjunction with the DB2_BIN_AS_CHAR statement.

Operating Systems: i5/OS

Data Sources: DB2 for i5/OS

Related Statements: DB2_BIN_AS_CHAR

Required: No

Syntax:

```
DB2_BIN_CODEPAGE=(sbcs_ccsid
                 ,dbcs_ccsid
)
```

Parameters:

sbcs_ccsid

Required. CCSID value for single-byte data. Valid values are from 1 through 65534.

dbcs_ccsid

Required. CCSID value for multibyte data. Valid values are from 1 through 65534.

Example: The following statement defines CCSID values for Japanese data:

```
DB2_BIN_CODEPAGE=(8482,1390)
```

DB2_ERRORFILE Statement

The DB2_ERRORFILE statement specifies the name of the user-customized SQL error file that PowerExchange uses for DB2 bulk data movement operations.

Operating Systems: All

Data Sources: DB2

Required: No

Syntax:

```
DB2_ERRORFILE=file_name
```

Value: For the *file_name* variable, enter the name of the file or member that contains the SQL error codes that you want PowerExchange to treat as either recoverable or fatal.

The content of *file_name* varies based on the operating system on which DB2 runs, as follows:

- **i5/OS**. Specify the library and file, and optionally member.
- **z/OS**. Specify the sequential or PDS and member.
- **Linux, UNIX, and Windows**. Specify the path and file name.

On i5/OS or z/OS, if you include a file or member name, enclose the file or member name in double quotation marks (“).

Usages Notes: PowerExchange supplies sample error files.

The following table lists the sample error files for each DB2 data source:

Data Source	Sample Error File
DB2 for z/OS	DB2ERR member in RUNLIB
DB2 for i5/OS	DB2ERR member in *datalib*/CFG
DB2 for Linux, UNIX, and Windows	db2err.act in the PowerExchange installation directory

RDBMSINSRTDFLT Statement

The RDBMSINSRTDFLT statement controls whether PowerExchange uses default values for columns that you define with the WITH DEFAULT clause in an RDBMS.

Operating Systems: All

Data Sources: DB2, Microsoft SQL Server, MySQL, and Oracle targets

Required: No

Syntax:

```
RDBMSINSRTDFLT={N|Y}
```

Valid Values:

- **N.** PowerExchange uses PowerExchange defaults when writing data to columns that you define with the WITH DEFAULT clause.
- **Y.** PowerExchange uses RDBMS defaults when writing data to columns that you define with the WITH DEFAULT clause.

  You must define the columns with a clause that enables the RDBMS to supply a default. Otherwise, an SQL error is generated.

Default is N.

Moving DB2 for i5/OS Bulk Data

To move DB2 for i5/OS bulk data, use one of the following access methods:

- DB2 access method
- SEQ access method

Moving DB2 for i5/OS Bulk Data - DB2 Access Method

Use the following procedure to move DB2 for i5/OS bulk data. This procedure assumes that you use the DB2 access method and PWXPC without a data map.

Before you begin, gather the following information:

- DB2 database name and schema name
- DB2 table names in one of the following formats: *schema.table* if you are using the SQL table format or *library.file* if you are using the sequential file format
- i5/OS user ID and password, if required by the security settings in the SECURITY statement in the DBMOVER configuration file

Also, if you want to preview the DB2 data from the PowerExchange Navigator first, you can create a personal metadata profile and perform a database row test on it. The PowerExchange Navigator displays metadata for each DB2 column. The metadata shows column attributes such as datatype, date format, and CCSID. DB2 for i5/OS can store dates and timestamps in multiple date formats.

To move DB2 for i5/OS bulk data by using the DB2 access method:

1. In the PowerCenter Designer, click **Source** > **Import from PowerExchange** for a DB2 data source or **Target** > **Import from PowerExchange** for a DB2 data target.

2. In the Import from PowerExchange dialog box, select **DB2i5OS** as the source type. Also, enter the location, the DB2 database name, and the user ID and password that are required to access the database, if any. Complete the other fields, as needed.

   Note: In the **Location** field, enter the PowerExchange Listener node name that you specified in the NODE statement in the DBMOVER configuration file on the PowerExchange system.
3. In the PowerCenter Designer, create a mapping that includes the DB2 for i5/OS source or target.
4. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX DB2i5OS Batch relational database connection. Then start the workflow to perform the bulk data movement.

Moving DB2 for i5/OS Bulk Data - SEQ Access Method

Use the following procedure to move DB2 for i5/OS bulk data. This procedure assumes that you use the SEQ access method and PWXPC.

To move DB2 for i5/OS bulk data by using the SEQ access method:

1. In the PowerExchange Navigator, create a data map.

   Select **SEQ** as the access method. To import metadata about records and fields, also select the **Import Record Definitions** option.

   Tip: If you do not select the **Import Record Definitions** option, you can import the metadata later. To do so, select the data map in the Resource Explorer and click **File > Import Copybook**.
2. In the **File Name** field, specify the i5/OS file using one of the following formats:
   - *library*/*file*
   - *library*/*file*(*member*)
3. Open the data map in the Resource Explorer, and click the **Record** or **Table** view to verify that the metadata was successfully retrieved.
4. To send the data map to the PowerExchange Listener on the remote i5/OS system, select the data map in the Resource Explorer and click **File > Send to Remote Node**.

   In the **Location** field, enter the value that is specified in the NODE statement of the dbmover.cfg file on the local PowerExchange system. Complete the other fields, as needed.

   Note: If you do not complete this step, you are prompted to send the data map to the remote node when you run a database row test.
5. Run a database row test on the table view of the metadata to test that actual data can be returned from the Datacom source database.
6. From the PowerCenter Designer, import the PowerExchange data map and create a mapping.

   Tip: When you import the data map, in the **Location** field, enter the PowerExchange Listener node name that is specified in the NODE statement of the dbmover.cfg on local system.
7. From the PowerCenter Workflow Manager, define a workflow and a session and configure a PWX NRDB Batch application connection. Then start the workflow to perform the bulk data movement operation.

Generating SQL Statements to Re-create a Source or Target for Troubleshooting

PowerExchange provides a command that generates SQL statements for re-creating SQL source or target objects that are used in a PowerExchange environment. The command is intended to produce SQL statements that Informatica Global Customer Support can use for troubleshooting bulk data movement or CDC problems.

To generate the SQL statements for an i5/OS SQL object, enter the RTVSQLSTMT command from the i5/OS system where PowerExchange is installed. The PowerExchange *dtllib* library must be in the current library list for the i5/OS job. The RTVSQLSTMT command prompts you for a series of parameters that control what SQL statements are generated. The RTVSQLSTMT command validates your entries to reduce the risk of errors.

Important: At any point, you can display Help information in the i5/OS Console to see field descriptions, command examples, and the authorities that are required to run the command.

The command can generate DDL statements for re-creating many types of objects, including aliases, constraints, indexes, procedures, schemas (libraries or collections), tables, triggers, and views. The command can also generate GRANT statements that provide the authority required to use the objects. Options are available to control the types of SQL statements to generate. For example, you can control whether to generate DROP statements that precede the CREATE statements for the selected object types. Also, the command can generate SQL statements even for some objects that were not created with SQL. For example, if you used data description specifications (DDS) to define physical and logical database files, the command can generate equivalent SQL CREATE statements to re-create those files.

The following sample command shows all of the required and optional parameters:

```
RTVSQLSTMT SQLOBJNAM(SQL_OBJECT_NAME) SQLOBJLIB(OBJECT_LIB) SQLTYPE(*TYPE)
SRCFILE(SOURCE_LIB/QSQLSRC) SRCMBR(SQLOBJNAM) RPLSRCMBR(*NO) CRTDROP(*NO) CRTCOMMENT(*NO)
CRTHEADER(*NO) CRTTRIGGER(*NO) CRTCONSTR(*NO) CRTRENAME(*NO) CRTGRANT(*NO) GENCCSID(*NO)
CRTORRPL(*NO) CRTOBFSTM(*NO)ACTROWCOL(*NO) CRTMSKPRM(*NO) CRTQUALSTM(*NO) CRTADDINDX(*NO)
CRTVWINDX(*NO)
```

Note: The parameter names are highlighted in boldface. For descriptions of these parameters and all of their options, see the Help.

After you run the command, the command writes the SQL statements to the source file member that you specified. If a member that has the same source member name already exists, you can configure the RPLSRCMBR parameter to indicate whether to replace it.

Send the source file member that contains the SQL statements to Informatica Global Customer Support. Support personnel can then run the SQL statements on another machine to re-create the environment in which your error occurred.

For information about the validation messages that PowerExchange generates when you use the interface, see messages DTL5001 through DTL5008 in the *PowerExchange Message Reference Volume 3*.

Refreshing the PowerExchange Environment After an i5/OS Upgrade

If you upgrade the i5/OS operating system on the DB2 system where PowerExchange is installed or on a remote DB2 system from which PowerExchange accesses data, you must run a few commands to refresh the

PowerExchange environment for any changes to the system metadata. Run the commands after you complete the i5/OS upgrade and before you re-submit the PowerExchange Listener job or PowerExchange Condense job and resume bulk data movement or CDC processing.

You must run the commands under one of the following types of user profiles:

- The IBM-supplied user profile of QSECOFR
- A user profile defined with the USRCLS of *SECOFR and SPCAUT of *USRCLS
- A user profile defined with at least the following SPCAUT values: *SECADM, *ALLOBJ and *JOBCTL

1. Issue the following ADDLIBLE command:

   ```
   ADDLIBLE LIB(dtllib) POSITION(*FIRST)
   ```

   The *dtllib* variable is the PowerExchange software library.

2. To refresh the PowerExchange environment, issue one of the following CRTDTLENVF commands with the same parameters that you used for the CRTPWXENV command during the last full PowerExchange installation:

   - If the DB2 for i5/OS source database is on the same i5/OS server as the PowerExchange installation, use:

     ```
     CRTDTLENVF DTLLIB(dtllib) DATALIB(datalib) RMTRDBDIRE(*LOCAL) OSLEVEL(*LOCAL)
     ```

   - If the DB2 for i5/OS source database is on a server other than the one where PowerExchange is installed, use:

     ```
     CRTDTLENVF DTLLIB(dtllib) DATALIB(datalib) RMTRDBDIRE(database_name)
     RMTSYSNAME(host_name) RMTOSLEVEL(os_level)
     ```

   For descriptions of the parameters, see Chapter 4 in the *PowerExchange Installation and Upgrade Guide*.

3. If you changed the ownership of objects such the *dtllib*, *datalib*, *condlib*, and *cpxlib* libraries during the last full PowerExchange installation, run the commands for changing object ownership again.

   For more information, see Chapter 4 in the *PowerExchange Installation and Upgrade Guide*.

CHAPTER 6

DB2 for Linux, UNIX, and Windows Bulk Data Movement

This chapter includes the following topics:

Introduction to DB2 Bulk Data Movement

You can optionally use PowerExchange, in conjunction with PowerCenter, to move bulk data from or to a DB2 for Linux, UNIX, and Windows database.

Because DB2 is a relational database, you do not need to create a data map from the PowerExchange Navigator. You can define the bulk data movement entirely from the PowerCenter Designer and Workflow Manager. In the PowerCenter Designer, you can import the DB2 source or target metadata that PowerExchange reads from the DB2 catalog to create the source or target definitions.

If the DB2 database is on a system that is remote from the system or systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, you must run an additional PowerExchange Listener on the remote system and verify that PowerExchange can communicate with it.

Alternatively, you can use the following bulk data movement strategies:

- Perform the bulk data movement entirely from PowerCenter.
- Create DB2 data maps in the PowerExchange Navigator and then import them into PowerCenter in the same manner as nonrelational data maps. You might want to use this strategy if you want to create a user-defined field and build expressions to populate it.

Datatypes Supported for DB2 for Linux, UNIX, and Windows Bulk Data Movement

PowerExchange supports most DB2 for Linux, UNIX, and Windows datatypes for bulk data movement. Exceptions are columns with LOB, DECFLOAT, or XML datatypes.

The following table lists the DB2 datatypes that can appear in the DB2 catalog and indicates whether PowerExchange supports them for bulk data movement:

DB2 Datatype	Supported for Bulk Data Movement?
BIGINT	Yes
BLOB	No
CHAR	Yes
CLOB	No
DATE	Yes
DBCLOB	No
DECFLOAT	No
DECIMAL	Yes
DOUBLE	Yes
GRAPHIC	No
INTEGER	Yes
LONG VARCHAR	Yes
LONG VARGRAPHIC	No
REAL	Yes
REF	No
SMALLINT	Yes
TIME	Yes
TIMESTAMP	Yes
UDTs[1]	No
VARCHAR	Yes
VARGRAPHIC	No

DB2 Datatype	Supported for Bulk Data Movement?
XML	No
1. User-defined datatypes, such as DISTINCT and STRUCT.	

Configuring DB2 Bulk Data Movement

To configure PowerExchange for bulk data movement operations, the PowerExchange administrator completes the following tasks:

- If the DB2 data source or target is on a remote system, set up and test connectivity to the PowerExchange Listener on the remote system.
- In the dbmover.cfg file, include the DB2_ERRORFILE statement if you created an error action file to customize PowerExchange fault tolerance behavior.
- To specify the code page, you must define the following environment variable, `DBCODEPAGE=1208`.

Setting Up and Testing Connectivity to a Remote DB2 Source or Target

If the DB2 data source or target is on a remote system, PowerExchange and PowerCenter must be able to communicate with the PowerExchange Listener on that remote system.

To set up and test connectivity to a remote DB2 source or target:

1. On the systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the **ping** command to test network access to the remote system with the DB2 source or target.
2. Add the NODE statement to the dbmover.cfg file on each PowerExchange and PowerCenter system to identify the remote PowerExchange Listener and its port:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   Tip: To determine the port number, consult with the person in your organization who installed the PowerExchange Listener on the remote system.
3. Run the DTLREXE ping utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Configuring DB2 Statements in the DBMOVER Configuration File

To configure DB2 for Linux, UNIX, and Windows bulk data movement, you can include the following DB2-specific statements in the dbmover.cfg file on the Linux, UNIX, or Windows source or target system:

- DB2_ERRORFILE
- RDBMSINSRTDFLT

After editing the DBMOVER configuration file, restart the PowerExchange Listener for your changes to take effect.

For descriptions of all DBMOVER statements, see the *PowerExchange Reference Manual*.

DB2_ERRORFILE Statement

The DB2_ERRORFILE statement specifies the name of the user-customized SQL error file that PowerExchange uses for DB2 bulk data movement operations.

Operating Systems: All

Data Sources: DB2

Required: No

Syntax:

```
DB2_ERRORFILE=file_name
```

Value: For the *file_name* variable, enter the name of the file or member that contains the SQL error codes that you want PowerExchange to treat as either recoverable or fatal.

The content of *file_name* varies based on the operating system on which DB2 runs, as follows:

- **i5/OS**. Specify the library and file, and optionally member.
- **z/OS**. Specify the sequential or PDS and member.
- **Linux, UNIX, and Windows**. Specify the path and file name.

On i5/OS or z/OS, if you include a file or member name, enclose the file or member name in double quotation marks (“).

Usages Notes: PowerExchange supplies sample error files.

The following table lists the sample error files for each DB2 data source:

Data Source	Sample Error File
DB2 for z/OS	DB2ERR member in RUNLIB
DB2 for i5/OS	DB2ERR member in *datalib*/CFG
DB2 for Linux, UNIX, and Windows	db2err.act in the PowerExchange installation directory

RDBMSINSRTDFLT Statement

The RDBMSINSRTDFLT statement controls whether PowerExchange uses default values for columns that you define with the WITH DEFAULT clause in an RDBMS.

Operating Systems: All

Data Sources: DB2, Microsoft SQL Server, MySQL, and Oracle targets

Required: No

Syntax:

```
RDBMSINSRTDFLT={N|Y}
```

Valid Values:

- **N.** PowerExchange uses PowerExchange defaults when writing data to columns that you define with the WITH DEFAULT clause.
- **Y.** PowerExchange uses RDBMS defaults when writing data to columns that you define with the WITH DEFAULT clause.

  You must define the columns with a clause that enables the RDBMS to supply a default. Otherwise, an SQL error is generated.

Default is N.

Moving DB2 Bulk Data

Use the following procedure to move bulk data from or to a DB2 for Linux, UNIX, and Windows database. This procedure assumes that you are using PWXPC to integrate PowerExchange with PowerCenter.

Before you begin, get the following information:

- DB2 database name
- Table names for the DB2 source or target tables
- Database user ID and password

  Tip: From the PowerExchange Navigator, you can preview DB2 source data before moving it. Create a personal metadata profile and perform a database row test on it. The PowerExchange Navigator displays metadata, such as datatype, date format, and CCSID, for each DB2 column.

To move DB2 bulk data:

1. In the PowerCenter Designer, click **Source** > **Import from PowerExchange** for a DB2 data source or **Target** > **Import from PowerExchange** for a DB2 data target.
2. In the **Import from PowerExchange** dialog box, enter the following required attributes:
   - In the **Location** field, enter the PowerExchange Listener node name that you specified in the LISTENER statement of the local dbmover.cfg file if the DB2 database is on the local system where the PowerCenter runs. If the DB2 database is on another system, enter the node name that is specified in the NODE statement of the local dbmover.cfg.
   - In the **User name** and **Password** fields, enter the user ID and password for accessing the DB2 source or target tables.
   - In the **Source Type list, select DB2LUW.**
   - In the **Database Name** field, enter the DB2 database name.

     Complete the optional attributes as needed.
3. In the PowerCenter Designer, create a mapping that includes the DB2 source or target.
4. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX DB2LUW relational database connection. Then start the workflow to perform the bulk data movement.

CHAPTER 7

DB2 for z/OS Bulk Data Movement

This chapter includes the following topics:

Introduction to DB2 for z/OS Bulk Data Movement

PowerExchange, in conjunction with PowerCenter, can move bulk data from or to DB2 for z/OS tables.

You can optionally create a DB2 image copy or unload file for a data source and then use that image copy or unload file for the bulk data movement operation. To create an image copy that PowerCenter can use, you must use the DB2 online COPY utility or a utility that produces a COPY-compatible image copy. Create a full image copy of either a DB2 source table space or table space partition.

Also, you can use the DB2 LOAD utility to load bulk data to a DB2 for z/OS target. PowerExchange provides JCL and control card templates for running the DB2 LOAD utility. PowerExchange also generates a data file that contains the data for the LOAD utility to load to the DB2 target. If appropriate, you can run a bulk data LOAD operation as a task of a netport job.

If you use PowerExchange with any of these utilities, you must perform some additional configuration tasks for the bulk data movement.

For relational sources and targets such as DB2 for z/OS tables, you do not need to create a data map. PowerCenter can import the metadata that PowerExchange reads from the DB2 catalog to create source or target definitions. PowerExchange uses these definitions to access the source or target tables when PowerCenter runs the bulk data movement.

However, PowerCenter cannot treat DB2 sources such as DB2 unload files as relational sources. To have PowerCenter treat DB2 data sources as nonrelational sources, you must create a data map with the DB2 or DB2UNLD access method in PowerExchange Navigator. Then import that data map into PowerCenter to create nonrelational-type source definitions.

Datatypes Supported for DB2 for z/OS Bulk Data Movement

PowerExchange supports most DB2 for z/OS datatypes for bulk data movement.

The following table identifies the DB2 for z/OS datatypes that PowerExchange supports and does not support for bulk data movement:

DB2 Datatype	Supported for Bulk Data Movement?
BIGINT	Yes
BINARY	Yes
BLOB	Yes (sources only)
CHAR	Yes
CLOB	Yes (sources only)
DATE	Yes
DBCLOB	Yes (sources only)
DECFLOAT	No
DECIMAL	Yes
DISTINCT (user-defined)	No
DOUBLE	Yes
FLOAT	Yes
GRAPHIC	Yes
LONG VARCHAR	Yes
LONG VARGHAPHIC	Yes
INTEGER	Yes
REAL	Yes
ROWID	No
SMALLINT	Yes
TIME	Yes
TIMESTAMP	Yes, including extended-precision TIMESTAMP columns, which support fractional seconds up to 12 digits
TIMESTAMP WITH TIME ZONE	No

DB2 Datatype	Supported for Bulk Data Movement?
VARBINARY	Yes
VARCHAR	Yes
VARGRAPHIC	Yes
XML	No

DB2 for z/OS TIMESTAMP Datatype

DB2 10 for z/OS introduced support for extended-precision TIMESTAMP columns, which can have subsecond values of up to 12 digits. How PowerCenter maps extended-precision TIMESTAMP columns to transformation datatypes depends on the scale.

The following table summarizes how the extended-precision TIMESTAMP datatype maps to transformation datatypes:

Scale	Precision	Transformation Datatype
6	26	date/time
0	19	string
1 to 5 or 7 to 12	20+scale	string

When you write DB2 TIMESTAMP data to a DB2 target, the source and target must have the same scale. Otherwise, unpredictable results can occur.

If the DB2 source TIMESTAMP field has a scale greater than 9 and the corresponding DB2 target TIMESTAMP field has a scale of 6, the value is truncated before it is written to the target.

When you write extended-precision TIMESTAMP data to a nonrelational target, define the following session attributes in the **Edit Tasks** dialog box to ensure that all TIMESTAMP fields have the same format:

- Set the **Date Time Format String** to **YYYY-MM-DD HH24:MI:SS**.
- Clear **Pre 85 Timestamp Compatibility**.

DB2 for z/OS LOB Datatypes

You can include a DB2 table that contains columns with LOB data as a source in a bulk data movement session. LOB datatypes include BLOB, CLOB, and DBCLOB.

The maximum length of a DB2 for z/OS row with LOB columns that you can define in a data map or include in a PowerExchange bulk data movement operation is 8 MB.

The following restrictions apply to tables that include LOB columns:

- You cannot use an unload file as a data source.
- You can use an image copy as a data source only if the LOB columns are inline.
- You cannot perform an insert into a target table that contains LOB columns.

Tip: When reading data from DB2, you can use the SUBSTRING command to reduce the amount of data that PowerExchange reads from a LOB column.

To select data from a BLOB or CLOB column, use a statement such as the following one:

```
select SUBSTRING(LOBVALUE,1,900,OCTETS) from myschema.mytable
```

To select data from a DBCLOB column, use a statement such as the following one:

```
select SUBSTRING(DBCLOBVALUE,1,900,CODEUNITS16) from myschema.mytable
```

Note that the PowerExchange nonrelational SQL that is used when processing DB2 image copy files or CDC data does not support the SUBSTRING() function.

Configuring DB2 for z/OS Bulk Data Movement

To configure PowerExchange bulk data movement operations, the PowerExchange administrator completes the following configuration tasks:

1. Set up and test connectivity to the PowerExchange Listener on the remote DB2 for z/OS system.
2. Configure DB2 for z/OS bulk data movement by defining DB2-specific statements in the DBMOVER configuration member on the z/OS source or target system.
3. Grant users the required authorities to PowerExchange user applications to access the DB2 data that they need.
4. Optional. If you want PowerExchange to support the DB2 multi-row fetch and insert feature, verify that the requirements for this feature are met.
5. Optional. To use the DB2 LOAD utility to load bulk data to a DB2 target, configure PowerExchange for LOAD utility use.

Related Topics:

- "Adding DB2-Specific Statements to the DBMOVER Configuration Member" on page 82
- "Required Authorities for Access to DB2 Resources" on page 90
- "DB2 Multiple-Row FETCH and INSERT Statements" on page 92
- " PowerExchange Templates for the DB2 LOAD Utility" on page 98
- "Setting Up and Testing Connectivity to a Remote DB2 for z/OS Source or Target" on page 81

Setting Up and Testing Connectivity to a Remote DB2 for z/OS Source or Target

To access DB2 bulk data on a remote z/OS system, PowerExchange must be able to communicate with the PowerExchange Listener on the remote system.

To set up and test connectivity to a remote DB2 for z/OS source or target:

1. On the systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the **ping** command to test network access to the remote MVS system with the DB2 source or target.
2. Add the NODE statement to the dbmover.cfg file on each PowerExchange or PowerCenter system to identify the remote PowerExchange Listener and its port:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   Tip: To determine the port number, consult with the person in your organization who installed the PowerExchange Listener on the remote system.

3. Run the DTLREXE ping utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Adding DB2-Specific Statements to the DBMOVER Configuration Member

To configure DB2 for z/OS bulk data movement, you can include the following DB2-specific statements in the DBMOVER configuration member on the z/OS source or target system:

- DB2_ERRORFILE
- DB2CODEPAGE
- DB2DEF_ENCODING
- DB2ID
- DB2PLAN
- MVSDB2AF
- RDBMSINSRTDFLT
- SESSID

After editing the DBMOVER configuration member, you must restart the PowerExchange Listener for your changes to take effect.

For descriptions of all DBMOVER statements, see the *PowerExchange Reference Manual*.

DB2_ERRORFILE Statement

The DB2_ERRORFILE statement specifies the name of the user-customized SQL error file that PowerExchange uses for DB2 bulk data movement operations.

Operating Systems: All

Data Sources: DB2

Required: No

Syntax:

```
DB2_ERRORFILE=file_name
```

Value: For the *file_name* variable, enter the name of the file or member that contains the SQL error codes that you want PowerExchange to treat as either recoverable or fatal.

The content of *file_name* varies based on the operating system on which DB2 runs, as follows:

- **i5/OS**. Specify the library and file, and optionally member.
- **z/OS**. Specify the sequential or PDS and member.
- **Linux, UNIX, and Windows**. Specify the path and file name.

On i5/OS or z/OS, if you include a file or member name, enclose the file or member name in double quotation marks (").

Usages Notes: PowerExchange supplies sample error files.

The following table lists the sample error files for each DB2 data source:

Data Source	Sample Error File
DB2 for z/OS	DB2ERR member in RUNLIB
DB2 for i5/OS	DB2ERR member in *datalib*/CFG
DB2 for Linux, UNIX, and Windows	db2err.act in the PowerExchange installation directory

DB2CODEPAGE Statement

The DB2CODEPAGE statement defines the single-byte, graphic, and mixed CCSIDs that PowerExchange uses to process CHAR and VARCHAR column data in a DB2 for z/OS subsystem that is involved in bulk data movement.

Operating Systems: z/OS

Data Sources: DB2 for z/OS

Required: No

Syntax:

```
DB2CODEPAGE=(db2_subsystem
            [,DB2TRANS={P|N|R}]
            [,MIXED={N|Y}]
            [,EBCDIC_CCSID=({sbcs_ccsid|037}
                           ,{graphic_ccsid|037}
                           ,{mixed_ccsid|037})]
            [,ASCII_CCSID=({sbcs_ccsid|850}
                          ,{graphic_ccsid|65534}
                          ,{mixed_ccsid|65534})]
            [,UNICODE_CCSID=({sbcs_ccsid|367}
                            ,{graphic_ccsid|1200}
                            ,{mixed_ccsid|1208})]
            [,PLAN_CCSID=({sbcs_ccsid|037}
                         ,{graphic_ccsid|037}
                         ,{mixed_ccsid|037})]
            [,REMAPn=(current_data_ccsid),(remapped_data_ccsid)

)
```

Parameters:

db2_subsystem

Required. The DB2 subsystem identifier (SSID) of a source or target subsystem.

DB2TRANS={P|N|R}

Indicates whether DB2 translates the encoding of data that it passes to or receives from PowerExchange. Options are:

- **P**. DB2 translates the code pages in which column data is stored into the code pages defined in the DB2 plan that was bound for PowerExchange. You must also specify the EBCDIC_CCSID parameter and optionally the PLAN_CCSID parameter. If you specify both, the PLAN_CCSID parameter takes precedence. If you have ASCII and Unicode data, you can also specify the ASCII_CCSID and UNICODE_CCSID parameters to map to the EBCDIC code pages.

  Note: To use any of the *_CCSID parameters, you must set DB2TRANS to P.

- **N**. DB2 does not translate the code pages of the column data to equivalent EBCDIC code pages. PowerExchange uses the native code page in which the data is stored. You do not need to define the EBCDIC_CCSID, ASCII_CCSID, UNICODE_CCSID, or PLAN_CCSID parameters.
- **R**. DB2 translates certain user-specified data code pages to other code pages, as defined in one or more REMAP*n* parameters. In each REMAP*n* parameter, the first positional parameter identifies a data code page to remap, and the second positional parameter identifies the code page to use. Use a code page other than the code page in which the PowerExchange DB2 plan is bound.

Default is P.

MIXED={N|Y}

Indicates whether DB2 columns contain ASCII and EBCDIC character strings with mixed data and graphic data. Mixed data consists of both single-byte and double-byte characters, and graphic data consists of double-byte characters.

Options are:

- **N.** Columns with ASCII and EBCDIC character strings contain only single-byte (SBCS) data. Mixed and graphic data does not occur.
- **Y.** Columns with ASCII and EBCDIC character strings contain mixed data and graphic data.

Default is N.

EBCDIC_CCSID=({*sbcs_ccsid*|037},{*graphic_ccsid*|037},{*mixed_ccsid*|037})

Required if DB2TRANS=P (the default). The CCSIDs to use for EBCDIC single-byte, graphic double-byte, and mixed (single-byte and double-byte) data.

Valid values are from 1 through 65534 for each positional subparameter. A value of 65534 indicates no CCSID.

The default values of 037,037,037 are used if you do not specify the DB2CODEPAGE statement or if you accept the values from the z/OS Installation Assistant without modification. If you changed these values during installation, the Installation Assistant customizes the DB2CODEPAGE statement in the DBMOVER file with the values you entered.

If an EBCDIC code page does not have an ICU conversion table, or if the EBCDIC code page is different from the one in which the DB2 plan for PowerExchange is bound, you can use the PLAN_CCSID parameter to map to another EBCDIC code page that does have an ICU conversion table or that reflects the DB2 plan.

Note: DB2 delivers data to PowerExchange in an EBCDIC code page even if the data is physically stored in an ASCII or Unicode code page. The DB2 catalog tables are stored in UTF-8, which corresponds to CCSID 1208. This CCSID is remapped from the UNICODE_CCSID *mixed_ccsid* code page value to the EBCDIC_CCSID *mixed_ccsid* code page value.

On single-byte systems, either repeat the SBCS code page three times or use 65534 with the MIXED=N parameter. For example:

```
EBCDIC_CCSID=(1047,1047,1047)
```

or

```
DB2CODEPAGE=(D91G,EBCDIC_CCSID=(1047,65534,65534),MIXED=N)
```

PowerExchange then uses the single-byte EBCDIC code page.

ASCII_CCSID=({*sbcs_ccsid*|850},{*graphic_ccsid*|65534},{*mixed_ccsid*|65534})

Optional. The CCSIDs to use for ASCII single-byte, graphic, and mixed data. Specify this parameter only if your subsystem contains ASCII encoded data that is involved in bulk data movement. PowerExchange maps these ASCII code pages to equivalent EBCDIC code pages that are defined in the EBCDIC_CCSID parameter. Also include the EBCDIC_CCSID parameter.

Valid values are from 1 through 65534 for each subparameter. A value of 65534 indicates no CCSID.

The z/OS Installation Assistant inserts the values 850,65534,65534 unless you change them during installation.

UNICODE_CCSID=({*sbcs_ccsid*|367},{*graphic_ccsid*|1200},{*mixed_ccsid*|1208})

Optional. The CCSIDs for Unicode single-byte, graphic, and mixed data. PowerExchange maps these Unicode code pages to equivalent EBCDIC code pages that are defined in the EBCDIC_CCSID parameter. Also include the EBCDIC_CCSID parameter.

Valid values are from 1 through 65534 for each subparameter. A value of 65534 indicates no CCSID.

The default values of 367,1208,1200 are used if you do not specify the DB2CODEPAGE statement or if you accepted the z/OS Installation Assistant values without modification. Usually, these default values are suitable.

Note: In DB2 Version 8.1 and later, the DB2 catalog tables are stored in UTF-8, which corresponds to CCSID 1208.

PLAN_CCSID=({*sbcs_ccsid*|037},{*graphic_ccsid*|037},{*mixed_ccsid*|037})

Optional. The CCSIDs to use for EBCDIC single-byte, graphic, and mixed data instead of those in the EBCDIC_CCSID parameter. Use this parameter when you need to redirect the EBCDIC code pages to other EBCDIC code pages.

For example, use this parameter in the following situations:

- The EBCDIC_CCSID code pages do not have an ICU conversion table that PowerExchange can use for ICU-based code page conversion.
- The EBCDIC_CCSID code pages match the default code pages that were defined for the DB2 subsystem but differ from the EBCDIC code pages for a particular source or target table.

Default is 037,037,037.

REMAP*n*=(*current_data_ccsid*,*remapped_data_ccsid*)

Optional. If you specified DB2TRANS=R, you can use this parameter to have DB2 remap the code page in which the data is stored to another code page that you specify. For example, if you have ASCII data that does not map to the code page in which the DB2 plan is bound and that does not have an ICU convertor, use this parameter to remap the ASCII code page to a supported EBCDIC code page.

Alternatively, if you specified DB2TRANS=N, DB2 does not translate or remap the data. However, PowerExchange can use the REMAP*n* statement to substitute the correct code page for the incorrect one. For example, DB2 might report a data code page that does not match the code page defined in the DB2 catalog, possibly because the data was loaded incorrectly. In this case, you can specify the correct code page in the REMAP*n* parameter.

You can specify up to six REMAP*n* parameters in a DB2CODEPAGE statement, each for a different DB2 table. Increment the *n* number at the end of the parameter names so that each name is unique.

Usage Notes:

- If a PowerExchange Listener connects to a DB2 subsystem for which there is no DB2CODEPAGE statement, PowerExchange uses the code page of the PowerExchange Listener.

- During PowerExchange installation, a bind job binds the DB2 plan for PowerExchange as EBCDIC, without specifying the CCSIDs. As a result, PowerExchange uses the default application programming CCSIDs that were defined for the DB2 subsystem when it was created. PowerExchange retrieves these CCSIDs from the DB2 catalog tables and uses them along with the DB2CODEPAGE parameters to determine the code page to use.
- The values that you define for the DB2CODEPAGE statement must match the values that were specified for the DB2 subsystem on the application programming defaults panel, DSNTIPF. If you did not specify a value for the graphic and mixed CCSIDs in DB2, specify 65534 for the *graphic_ccsid* and *mixed_ccsid* parameters in the DB2CODEPAGE statement.

  If the CCSIDs that you specified for the DB2 subsystem differ from the DB2CODEPAGE default CCSIDs, you must edit the DB2CODEPAGE statement to match the DB2 CCSIDs.

  The following table shows the DB2 configuration options on the DSNTIPF panel and the DSNHDECP load module that correspond to the DB2CODEPAGE parameters:

DB2CODEPAGE Parameter	DSNTIPF Field	DSNHDECP Parameter
EBCDIC_CCSID	EBCDIC CCSID	SCCSID (single-byte), MCCSID (mixed), GCCSID (graphic)
ASCII_CCSID	ASCII CCSID	ASCCSID (single-byte), AMCCSID (mixed) AGCCSID (graphic)
UNICODE_CCSID	UNICODE CCSID	USCCSID (single-byte), UMCCSID (mixed), UGCCSID (graphic)
MIXED	MIXED DATA	MIXED

  For more information about CCSID values and their meanings, see the IBM *DB2 for z/OS Installation Guide* for the DB2 version.
- If you click **Advanced Parms** on the **DB2 Parameters** page of the z/OS Installation Assistant, you can enter CCSID values for the following parameters:

  •DB2CODEPAGE_ASCII_DBCS_CCSID

  •DB2CODEPAGE_ASCII_MIXED_CCSID

  •DB2CODEPAGE_ASCII_SBCS_CCSID

  •DB2CODEPAGE_EBCDIC_DBCS_CCSID

  •DB2CODEPAGE_EBCDIC_MIXED_CCSID

  •DB2CODEPAGE_EBCDIC_SBCS_CCSID

  •DB2CODEPAGE_MIXED

  •DB2CODEPAGE_UNICODE_DBCS_CCSID

  •DB2CODEPAGE_UNICODE_MIXED_CCSID

  •DB2CODEPAGE_UNICODE_SBCS_CCSID

  These installation parameters populate the EBCDIC_CCSID, ASCII_CCSID, and UNICODE_CCSID parameters in the DB2CODEPAGE statement of the DBMOVER file. You can edit them in the DBMOVER file if necessary.
- PowerExchange automatically generates a minimal DB2CODEPAGE specification that includes the UNICODE_CCSID and EBCDIC_CCSID parameters if no DB2CODEPAGE parameters are defined. The UNICODE_CCSID is included because DB2 catalog tables use Unicode encoding.
- The DB2CODEPAGE statement applies to bulk data movement operations only. For DB2 CDC, PowerExchange always uses the native code page of the DB2 data.

- The DB2CODEPAGE statement does not affect the CODEPAGE statement. If you use the DB2DEF_ENCODING statement, in certain cases, you might need to edit the DB2CODEPAGE statement to set the *mixed_ccsid* value to 037 in the ASCII_CCSID, EBCDIC_CCSID, or UNICODE_CCSID parameter that corresponds to the DB2DEF_ENCODING option.
- If PowerExchange tries to read columns that contain VARGRAPHIC or LOB data in a table on a single-byte subsystem, and the default DB2CODEPAGE setting of DB2TRANS=P is in effect, the you might receive the following SQL error:

  ```
  -332 CHARACTER CONVERSION BETWEEN CCSID from_ccsid TO to_ccsid REQUESTED BY
  reason_code IS NOT SUPPORTED
  ```

  If this situation occurs, specify DB2TRANS=N in the DB2CODEPAGE statement.

DB2DEF_ENCODING Statement

The DB2DEF_ENCODING statement defines the default encoding scheme that PowerExchange assigns to any DB2 columns that do not have an encoding scheme when you create a DB2 data map.

Operating Systems: z/OS

Data Sources: DB2 for z/OS

Related Statements: DB2CODEPAGE

Required: No

Syntax:

```
DB2DEF_ENCODING={A|E|U}
```

Valid Values:

- **A.** ASCII encoding.
- **E.** EBCDIC encoding.
- **U.** Unicode encoding.

Default is E.

Usage Notes:

- An encoding scheme might not be specified in the SYSIBM.SYSDATABASE table as a result of multiple DB2 migrations.
- In general, specify the encoding that is defined for DB2 in the **DEF ENCODING SCHEME** field on the application programming defaults panel, **DSNTIPF**. However, if DSNTIPF specifies 65534 for the mixed code page, edit the DB2CODEPAGE statement to set the *mixed_ccsid* value of the comparable ASCII, EBCDIC, or Unicode parameter to 037. The 037 value is required for PowerExchange to read the DB2 catalog tables.

  For example, if you use the default value of E for the DB2DEF_ENCODING statement and the DSNTIPF panel specifies 65534 for the EBCDIC mixed code page, specify 037 as the last positional subparameter of the EBCDIC_CCSID parameter in the DB2CODEPAGE statement:

  ```
  EBCDIC_CCSID=(sbcs_ccsid,graphic_ccsid,037)
  ```

DB2ID Statement

The DB2ID statement defines the DB2 subsystem, plan, and access method load module that PowerExchange uses to process data from a DB2 for z/OS source for bulk data movement.

Operating Systems: z/OS

Data Sources: DB2 for z/OS

Related Statements: DB2PLAN

Required: No

Syntax:

```
DB2ID=(db2_subsystem
      [,plan]
      [,{module_name|DTLAMV8F}]
)
```

Parameters:

db2_subsystem

The DB2 subsystem identifier. The value is populated at installation. If you are creating a DB2ID statement for another DB2 subsystem, you must enter this value.

plan

The DB2 plan name for PowerExchange bulk data movement operations. Default is the value from the DB2PLAN statement.

{module_name|DTLAMV8F}

The PowerExchange DB2 access method load module. Options are:

- **DEFAULT**. The default access method load module, which is DTLAMV8F.
- **DTLAMV8F**. The module that PowerExchange uses to process multiple rows of data at a time by using DB2 multiple-row FETCH and INSERT SQL statements.

Default is DTLAMV8F.

Usage Notes:

- If you can use the defaults for all parameters in this statement, including the subsystem ID from installation, you do not need to specify the DB2ID statement. Otherwise, you must define a DB2ID statement.
- You can specify up to 25 DB2ID statements in a DBMOVER member.

DB2PLAN Statement

The DB2PLAN statement defines the DB2 plan that PowerExchange uses for bulk data movement processing.

Operating Systems: z/OS

Data Sources: DB2 for z/OS

Related Statements: DB2ID

Required: No

Syntax:

```
DB2PLAN={plan|DTLPLvrm}
```

Value: For the *plan* variable, enter the DB2 plan name for PowerExchange bulk data movement operations. Default is DTLPL*vrm*, where *vrm* is the numeric value for the PowerExchange version, release, and modification level. For example, for PowerExchange 8.6.1, the *vrm* value is 861. For PowerExchange 9.5.0, the *vrm* is 950.

Usage Notes:

- PowerExchange uses the plan name from the DB2PLAN statement to access DB2 subsystems that you do not define with a DB2ID statement, or for which you do not specify a plan name in a DB2ID statement. If you use unique plan names for PowerExchange in different DB2 subsystems, define the subsystems and their unique plan names in the DB2ID statement.
- During the installation process, PowerExchange customizes the DBMOVER member and the XIDDB210 bind job with the plan name you specify in the z/OS Installation Assistant.

MVSDB2AF Statement

The MVSDB2AF statement specifies which DB2 for z/OS attachment facility PowerExchange uses for DB2 bulk data movement operations.

Operating Systems: z/OS

Data Sources: DB2 for z/OS

Required: No

Syntax:

```
MVSDB2AF={CAF|RRSAF}
```

Valid Values:

- **CAF.** PowerExchange uses the DB2 call attachment facility (CAF) to connect to DB2.

  When you use CAF, PowerExchange and DB2 use the user ID of the PowerExchange Listener or netport job for authentication of DB2 resources.
- **RRSAF.** PowerExchange uses the DB2 Resource Recovery Services attachment facility (RRSAF) to connect to DB2. To control user access to DB2 tables, specify RRSAF.

  RRSAF enables PowerExchange to sign on to DB2 and use the user ID of the user that requested the bulk data movement operation for authorization of DB2 resources.

  To use RRSAF, you must configure and run RRS on the z/OS system.

Default is CAF.

Usage Notes: If you specify RRSAF, PowerExchange uses the following values for correlation-id and accounting-token when signing on to RRSAF:

correlation-id

PowerExchange uses one of the following values:

- If provided, PowerExchange uses the value specified for the Correlation ID connection property in the PowerCenter Workflow Manager or informatica Developer tool.
- If no value is specified in the client for the Correlation ID connection property, PowerExchange uses the value specified in the SESSID statement in the DBMOVER configuration file on the PowerExchange Listener machine.
- If no value is specified in the client Correlation ID connection property or in the DBMOVER configuration file on the PowerExchange Listener machine, PowerExchange uses the default value of DETAIL.

accounting-token

PowerExchange concatenates the 8-byte job name, 8-byte user ID, and 6-byte task ID:

```
jobname||user_id||task_id
```

RDBMSINSRTDFLT Statement

The RDBMSINSRTDFLT statement controls whether PowerExchange uses default values for columns that you define with the WITH DEFAULT clause in an RDBMS.

Operating Systems: All

Data Sources: DB2, Microsoft SQL Server, MySQL, and Oracle targets

Required: No

Syntax:

```
RDBMSINSRTDFLT={N|Y}
```

Valid Values:

- **N.** PowerExchange uses PowerExchange defaults when writing data to columns that you define with the WITH DEFAULT clause.
- **Y.** PowerExchange uses RDBMS defaults when writing data to columns that you define with the WITH DEFAULT clause.

  You must define the columns with a clause that enables the RDBMS to supply a default. Otherwise, an SQL error is generated.

Default is N.

SESSID Statement

The SESSID statement specifies the default value to use as the DB2 Correlation ID for DB2 requests.

Operating Systems: z/OS

Required: No

Syntax:

```
SESSID=correlation_id
```

Value: For *correlation_id*, enter a string of up to eight characters.

Usage Notes:

- To specify a default value for the DB2 Correlation ID for DB2 requests, include the SESSID statement in the DBMOVER configuration file on the PowerExchange Listener machine. To override this value for a specific DB2 for z/OS connection, specify the Correlation ID connection property in the PowerCenter Workflow Manager or the Informatica Developer tool.
- If no value is specified in either the client Correlation ID connection property or the SESSID statement in the DBMOVER configuration file on the PowerExchange Listener machine, PowerExchange uses the default value of DETAIL.
- The SESSID statement applies only if PowerExchange uses RRSAF to connect to DB2. To use RRSAF connections, include the MVSDB2AF statement in the DBMOVER configuration file.

Required Authorities for Access to DB2 Resources

PowerExchange requires the following authority levels for access to DB2 for z/OS resources:

- EXECUTE authority on the DB2 plan
- SELECT authority on DB2 source tables
- INSERT authority on DB2 target tables

- LOAD authority on any DB2 target tables to be loaded by the DB2 LOAD utility
- SELECT authority on the following DB2 catalog tables:
  - SYSIBM.SYSCHECKS
  - SYSIBM.SYSCHECKDEP
  - SYSIBM.SYSCOLUMNS
  - SYSIBM.SYSDATABASE
  - SYSIBM.SYSDATATYPES
  - SYSIBM.SYSDUMMY1
  - SYSIBM.SYSFOREIGNKEYS
  - SYSIBM.SYSINDEXES
  - SYSIBM.SYSKEYS
  - SYSIBM.SYSPACKAGE
  - SYSIBM.SYSPACKSTMT
  - SYSIBM.SYSPARMS
  - SYSIBM.SYSRELS
  - SYSIBM.SYSROUTINES
  - SYSIBM.SYSSEQUENCES
  - SYSIBM.SYSSEQUENCESDEP
  - SYSIBM.SYSSYNONYMS
  - SYSIBM.SYSTABLES
  - SYSIBM.SYSTABLESPACE
  - SYSIBM.SYSTRIGGERS

  Note: PowerExchange requires SELECT authority on these tables to import metadata into the Designer or Developer tool when you create a mapping for a bulk data movement session.

Also, the user ID that DB2 uses to control PowerExchange user access to DB2 data in specific tables depends on the SECURITY and MVSDB2AF statement settings in the DBMOVER configuration member of the RUNLIB library.

The following table describes the user ID that is used under alternative statement settings:

MVSDB2AF Setting	SECURITY Setting	User ID for DB2 Access
CAF	0, 1, or 2	User ID under which the PowerExchange Listener is running
RRSAF	0 or 1	User ID under which the PowerExchange Listener is running
RRSAF	2	MVS user ID under which the user application is running

Granting Authorities for Access to DB2 Resources

Grant the appropriate authorities to PowerExchange to execute the PowerExchange DB2 plan and to access the DB2 catalog and source and target tables.

To grant authorities for access to DB2 resources:

- Issue the following SQL GRANT statements:

```
GRANT EXECUTE on PLAN db2_plan_name to user_id              [for the DB2 plan]

GRANT SELECT on SYSIBM.SYSTABLES to user_id                 [for the DB2 catalog]
GRANT SELECT on SYSIBM.SYSCOLUMNS to user_id

GRANT SELECT on creator.table to user_id                    [for each DB2 source
table]
GRANT INSERT on creator.table to user_id                    [for each DB2 target
table]
```

  Where:

  - *db2_plan_name* is the DB2 plan name that is specified in the DB2PLAN statement in the DBMOVER configuration member.
  - *user_id* is the z/OS user ID or PowerExchange Listener user ID that is associated with the user request to access DB2 source or target data by means of the PowerExchange Listener.
  - *creator*.*table_name* identifies a specific DB2 source or target table that is involved in bulk data movement.

Related Topics:

- "Required Authorities for Access to DB2 Resources" on page 90

DB2 Multiple-Row FETCH and INSERT Statements

When you use multiple-row FETCH and INSERT statements, DB2 fetches or inserts multiple rows of data at one time. As a result, PowerExchange accesses the database fewer times, which improves bulk data movement performance.

For all of the DB2 versions that PowerExchange supports, PowerExchange can use multiple-row FETCH and INSERT statements.

The PowerExchange default access method module, DTLAMV8F, uses multiple-row FETCH and INSERT statements.

To use multiple-row statements for bulk data movement operations, ensure that the following configuration requirements are met:

- In the DBMOVER configuration member, set the third positional parameter in the DB2ID statement to DTLAMV8F or DEFAULT to use multiple-row FETCH and INSERT statements. For more information about the DB2ID statement, see "DB2ID Statement" on page 87.
- Set the region size in the PWXLSTNR or STARTLST member for the PowerExchange Listener started task or job to a size that is large enough to accommodate the increased storage requirements of multiple-row FETCH and INSERT processing.
- Set the **Array Size** connection attribute to the number of rows that you want to fetch or insert at a time. Default is 25.

PowerExchange verifies that the version of the DB2 subsystem in the DB2ID statement supports multiple-row SQL statements.

PowerExchange dynamically lowers the array size when all the following conditions are true:

- The database type is DB2.
- The table contains LOB columns.
- The **Array Size** value is greater than 1.
- Row size ***Array Size** is greater than 16000000 bytes.

If these conditions are met, PowerExchange reduces the array size and logs message PWX-00186 on both the client and PowerExchange Listener machines.

For more information about DB2 multiple-row FETCH and INSERT statements, see the IBM DB2 for z/OS documentation.

DB2 for z/OS Bulk Data Movement with Relational Source or Target Definitions

To perform a bulk data movement with DB2 for z/OS source data, use a relational source database connection, except under the conditions listed in "DB2 for z/OS Bulk Data Movement with Nonrelational Source Definitions" on page 96.

When you use a DB2 image copy as a data source, special considerations and procedures apply. For more information, see "DB2 for z/OS Bulk Data Movement with an Image Copy Source" on page 94.

To perform a bulk data movement with a DB2 for z/OS target, use a relational target database connection.

Before you define the bulk data movement, gather the following information:

- DB2 subsystem ID
- DB2 table names in the format *creator.table_name*
- z/OS user ID and password if required by the security settings in the SECURITY statement in the DBMOVER configuration member

If you want to preview the DB2 data from the PowerExchange Navigator first, you can create a personal metadata profile and run a database row test on it. The PowerExchange Navigator displays metadata for each DB2 column in a table. The metadata shows column attributes such as datatype, date format, and CCSID.

Moving DB2 for z/OS Bulk Data with Relational Source or Target Definitions

Use the following procedure to have PowerCenter import DB2 for z/OS source or target definitions as relational data and perform the bulk data movement:

1. In the PowerCenter Designer, click **Source > Import from PowerExchange** for a DB2 data source or **Target** > **Import from PowerExchange** for a DB2 data target.
2. In the Import from PowerExchange dialog box, select **DB2zOS** as the **Source Type** field (also displayed for a target). Also, enter the location, the DB2 subsystem ID, and any user ID or password that is required for database access. Complete the other fields, as needed.

   Note: In the **Location** field, enter the PowerExchange Listener node name that you specified in the NODE statement in the dbmover.cfg file on the local PowerCenter system.
3. In the PowerCenter Designer, create a mapping that includes the DB2 for z/OS source or target.
4. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX DB2zOS relational database connection. Then start the workflow to move the bulk data.

DB2 for z/OS Bulk Data Movement with an Image Copy Source

You can use a full image copy of a DB2 table space or table space partition as a data source for a PowerExchange bulk data movement session. To create a full image copy, use the DB2 COPY utility.

For example, assume that you want to materialize a target based on an existing DB2 source. You can take a full image copy of the DB2 source. Then use that image copy as input to a PowerCenter workflow for materializing the target. After target materialization is complete, you can use PowerExchange change data capture (CDC) to keep the source and target synchronized.

When you use a full image copy as a data source, you do not need to create a data map. Instead, you can use a relational source definition that you import for a DB2 table.

Also, you can use a FlashCopy image copy or sequential image copy as a data source. If you use a FlashCopy image copy, ensure that it is for a table space that contains a single table and is not a deprecated type of table space.

When you define PowerCenter session properties, set the properties that control which image copy to read. You can configure the PowerExchange Reader for DB2 Image Copy to read any of the following image copies:

- The latest SHRLEVEL REFERENCE FULL image copy
- An image copy that you specify
- An image copy that was created on a different DB2 subsystem

Notes:

- If you create an image copy of a compressed table space, the compressed rows can be unloaded from the image copy data set only if the dictionary for decompression has been retrieved and assembled in the correct order.
- For PowerExchange to successfully process an image copy data set, the information in the DB2 catalog that describes the table and the table space containing the table must reflect the exact state of the objects at the time when the image copy was created. DB2 allows some ALTER statements to be issued and then later implemented (materialized) by running the REORG utility. Thus, a mismatch can occur between the information in the DB2 catalog and the content of the most recent image copy of a table space. Before you use an image copy as a source for bulk data movement, you must verify that no ALTER statements have been issued against the table or table space since the image copy was created.

Creating a DB2 for z/OS Image Copy

To create a full image copy as input for a bulk data movement session, define the JCL and control cards. Then submit the DB2 COPY utility job manually or from a job scheduler.

The JCL for creating a full image copy executes the DB2 DSNUTILB program. When the DB2 DSNUTILB program reads the control cards, it invokes the COPY utility to create the full image copy.

Include the following parameters in each COPY TABLESPACE statement:

- FULL YES
- SHRLEVEL REFERENCE

If you plan to use PowerExchange CDC to write changes to the target after materialization, include a QUIESCE TABLESPACE statement before the COPY TABLESPACE statements. The QUIESCE TABLESPACE statement causes PowerExchange to set event markers in the PowerExchange Logger log files. You need the event markers to create restart tokens for CDC. A restart token determines the start point in the change stream for a PowerExchange extraction.

For more information about creating restart tokens, see *PowerExchange Interfaces for PowerCenter*. For more information about creating image copies, see the appropriate IBM documentation.

Compressed Image Copies as Data Sources

When you use a compressed image copy as a data source for PowerExchange bulk data movement, the following requirements and considerations apply:

- Image copy decompression requires hardware support for the ESA/390 hardware instruction CMPSC. Verify that your hardware supports this instruction. If you enabled PowerExchange zIIP exploitation, decompression uses zIIP processing if available.

  If you use zIIP processing, PowerExchange can dispatch an array of compressed rows to the zIIP processor for expansion. The **Array Size** connection attribute controls the number of rows that are dispatched.

  Informatica recommends that you use the default array size of 25 unless you are able to test and determine whether the extra memory that is allocated to a larger array size has been beneficial and has not degraded server performance. If you are able to make these determinations, Informatica recommends an array size of 500 to 1000 if you use a compressed image copy as a data source for PowerExchange bulk data movement and you have enabled zIIP processing on z/OS.
- Use of a compressed image copy in a bulk data movement session increases memory usage. Verify that the REGION size that is specified in the DTLLST step in the JCL is large enough to run the PowerExchange Listener job.
- PowerExchange does not support compressed image copies that are created by a COPY TABLESPACE statement that includes the SYSTEMPAGES NO option. If you use an image copy of this type as a data source, PowerExchange issues the following message:

  ```
  PWX-09278 Compressed record encountered before dictionary complete.
  ```

  In certain cases, an S0C7 abend might occur.

Materializing a Target from a DB2 Image Copy

Use the following procedure to create a full image copy of a DB2 source table space and use it to materialize a target:

1. Create the JCL and control cards for creating the full image copy.
2. Submit the job for the DB2 COPY utility manually or from a job scheduler or a batch job. When the DB2 DSNUTILB program reads the control cards, it invokes the COPY utility to create the full image copy.
3. Create a target database structure, such as a table.
4. In the PowerCenter Designer, create a mapping. Include the DB2 table for which you created the image copy as the source and the database object from step 3 as the target.
5. In the PowerCenter Workflow Manager, create a connection, workflow, and session.

   Create a new application connection, and select **PWX NRDB Batch** as the connection type. If you want to enable offload processing, select **Filter After** for the **Offload Processing** attribute.

   When you configure the session, select **PowerExchange Reader for DB2 Image Copy** as the reader, and select the application connection that you just created from the list of connections.

   To specify the image copy, perform one of the following actions:

   - To read the latest image copy, do not enter a data set name in the **Image Copy Data Set** attribute for the reader.

- To read a full image copy other than the latest one, enter the data set name of that image copy in the **Image Copy Data Set** attribute for the reader.
- To read a full image copy file that was created on a different DB2 subsystem, enter the data set name of that image copy in the **Image Copy Data Set** attribute for the reader, and select **Disable consistency checking**.

Warning: If you select the **Disable consistency checking** property and use an image copy that was created on a different DB2 subsystem or with a previous DB2 version, the following additional restrictions apply:

- The table definition in the catalog of the DB2 subsystem being used to read the image copy must correctly reflect the table definition when the image copy was created.
- The image copy must contain data for only one table.
- The image copy must be created on a subsystem at the same DB2 version as the subsystem used to read the image copy.

6. Start the workflow to move the image copy data to the target table.

   If you plan to use PowerExchange CDC to update the target, you must set up restart tokens based on the event markers from the QUIESCE TABLESPACE step. For information about setting up restart tokens, see *PowerExchange Interfaces for PowerCenter*.

DB2 for z/OS Bulk Data Movement with Nonrelational Source Definitions

You can define a DB2 for z/OS data source as a nonrelational source. Use this approach with the following DB2 for z/OS data sources:

- DB2 unload files
- DB2 tables that contain columns that you want to split into multiple fields
- DB2 tables for which you want to define fields that use one or more PowerExchange functions in an expression to process source data
- DB2 tables that contain columns for which you want to change the data type, such as a column that is defined as CHARACTER but contains binary data

If you want PowerCenter to treat a DB2 for z/OS data source as a nonrelational source, create a data map in the PowerExchange Navigator and then import the data map into PowerCenter.

DB2 Unload Files as Bulk Data Sources

You can use a DB2 unload file as a data source for a PowerExchange bulk data movement. First create the unload file and then use it to add a data map that can be imported into PowerCenter. The data map uses the DB2UNLD access method.

To create a DB2 unload file, use any of the following IBM or BMC Software utilities:

- DB2 for z/OS online REORG TABLESPACE utility with UNLOAD EXTERNAL
- DB2 for z/OS online UNLOAD utility
- DB2 for z/OS sample DSNTIAUL unload program
- BMC UNLOAD PLUS for DB2

If you use the BMC UNLOAD PLUS for DB2 utility, the following utility options can result in errors:

- If you use the AUTOTAG YES option, the utility adds a 4-byte field at the beginning of each output record for each SELECT statement. The DB2UNLD data map that you create based on the unload file does not include this 4-byte field, which causes subsequent fields to be mapped incorrectly. If you then import the data map into PowerCenter for a PWX Batch session and try to run the session, the session fails. To avoid this problem either do not use the AUTOTAG YES option or manually add a 4-byte field at the beginning of each data map.
- If you use the FIXEDVARCHAR NO option, the PowerExchange Navigator issues mapping errors when it tries to access the unload file to generate a data map. To avoid this problem, generate the unload file with the FIXEDVARCHAR YES option.

If you use the DB2 Version 11 REORG TABLESPACE utility with UNLOAD EXTERNAL, the NOPAD option is used by default. This option causes variable-length columns in the unloaded records to have the actual data length, without padding. When the PowerExchange Navigator tries to access the unload file to generate a data map, record mapping errors occur. To avoid this problem, generate the unload file with the NOPAD NO option.

For information about adding a data map based on an unload file, see the *PowerExchange Navigator User Guide*.

Moving DB2 for z/OS Bulk Data with Nonrelational Source Definitions

Use this procedure to move DB2 for z/OS bulk data with nonrelational source definitions. The procedure assumes that you are using PWXPC to integrate PowerExchange with PowerCenter.

Before you begin, gather the following information:

- DB2 subsystem ID
- DB2 table names in the format *creator.table_name*
- Name of the unload file that you are using as a data source, if applicable.
- MVS user ID and password if required by the security settings in the SECURITY statement in the DBMOVER configuration member

1. From the PowerExchange Navigator, create a data map. In the **Access Method** list, select **DB2UNLD** if the data source is a DB2 unload file or **DB2** if the data source is a DB2 table that requires a data map. To import metadata about data source records and fields, also select the **Import Record Definitions** option.

   Tip: If you do not select the **Import Record Definitions**, you can import the metadata later. Open the data map in the Resource Explorer and click **File > Import Copybook**.

   For more information about creating a data map, see the *PowerExchange Navigator User Guide*.
2. Open the data map in the Resource Explorer, and click the “record” or “table” view to verify that the metadata was successfully retrieved.
3. Send the data map to the PowerExchange Listener on the remote DB2 for z/OS system. In the Resource Explorer, select the data map and click **File > Send to Remote Node**. This step enables PowerExchange extraction routines to access the data map at runtime.

   Note: If you do not complete this step, you are prompted to send the data map to the remote node when you run a database row test.
4. Run a database row test on the table view of the metadata to verify that data can be returned from the DB2 source database.
5. In the PowerCenter Designer, import the PowerExchange data map. For the source type, select **DB2UNLD** for an unload data map or **DB2MAP** for a DB2 table. In the **Location** field, enter the PowerExchange

Listener node name that you specified in the NODE statement of the local dbmover.cfg file. Complete the other fields, as needed.

6. Create a mapping.
7. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX NRDB Batch application connection. One of the following readers is automatically selected, depending on the data source:
   - PowerExchange Reader for DB2 Datamaps
   - PowerExchange Reader for DB2 Unload Datasets

   Then start the workflow to perform the bulk data movement.

Using the DB2 LOAD Utility to Load Bulk Data

You can use the DB2 LOAD utility to efficiently load large amounts of data to DB2 tables that PowerExchange or PowerCenter has read from data sources. PowerExchange includes JCL and control card templates for running the DB2 LOAD utility. PowerCenter uses PowerExchange Client for PowerCenter (PWXPC) connections to communicate with PowerExchange to implement the bulk data movement.

Depending on how you configure PWXPC connection attributes, PowerExchange can perform the following functions for the DB2 LOAD utility:

- Produce customized JCL and load cards for running the LOAD utility, generate a data file that contains the data to be loaded, and automatically submit the bulk load job to load data to the target.
- Create only a data file for the DB2 LOAD utility to load. In this case, you must run the DB2 LOAD utility outside of PowerExchange. You can submit the job manually, or use a job scheduler or the PowerExchange DTLREXE utility.

Note: Alternatively, you can use relational connections with the PowerExchange ODBC drivers. However, Informatica recommends that you use PWXPC connections. In general, PWXPC provides additional functionality, reliability, and performance.

Usually, a bulk data load operation runs either as a task in the PowerExchange Listener or as a separate job that the PowerExchange Listener submits. As an alternative, you can run the bulk data load as a netport job. This strategy is useful if you expect PowerExchange to generate very large load data sets for the LOAD utility and you want to store these data sets on tape. You can configure a netport job to wait for tape mounts.

PowerExchange Templates for the DB2 LOAD Utility

PowerExchange provides JCL and control card templates for bulk data load jobs that use the DB2 LOAD utility.

The following table describes these templates:

RUNLIB Member	Description
DB2LDCTL	Control card template for nonpartitioned tables.
DB2LDCTP	Control card template for partitioned tables.

RUNLIB Member	Description
DB2LDJCL	JCL template for nonpartitioned tables.
DB2LDJCP	JCL template for partitioned tables.

Use the DB2LDCTP and DB2LDJCP members to load bulk data to DB2 tables in partitioned table spaces. Use DB2LDCTL and DB2LDJCL to load bulk data to DB2 tables in nonpartitioned table spaces.

You can edit these templates to use other load utilities, or you can create alternative templates.

Note: For PowerExchange to use these templates to generate batch bulk load jobs, you must specify the names of the members that contain the templates in the PWXPC connection attributes for PWX DB2zOS relational database connections.

Related Topics:

- "PWX DB2zOS Connection Attributes for DB2 LOAD Jobs" on page 102

Summary of Configuration Steps

To configure PowerExchange to run DB2 bulk load operations, complete the following steps:

1. Customize the DBMOVER configuration member. This step is required.
2. Customize the PowerExchange JCL and control card templates for the DB2 LOAD utility. This step is required.
3. Customize the PowerExchange Listener JCL. This step is required.
4. Optionally, customize the netport JCL for the DB2 LOAD utility.
5. Set the PWXPC connection attributes or ODBC parameters for DB2 LOAD jobs. This step is required.

Related Topics:

- "Customizing the DBMOVER Configuration Member for the DB2 LOAD Utility" on page 100
- "Customizing the PowerExchange Templates for Bulk Data Loads" on page 100
- "Customizing the PowerExchange Listener JCL" on page 101

Customizing the DBMOVER Configuration Member for the DB2 LOAD Utility

In the DBMOVER configuration member, set statements to identify the libraries that contain the PowerExchange JCL and control card templates for the DB2 LOAD utility.

The following table describes these statements:

Statement	Description	Default
LOADJOBFILE	Name of PDS library that contains the JCL template for bulk data movement jobs that use the DB2 LOAD utility.	Default is the PowerExchange RUNLIB library. **Note:** The default template member name is DB2LDJCL.
LOADCTLFILE	Name of PDS library that contains the control card template for bulk data movement jobs that use the DB2 LOAD utility.	Default is the PowerExchange RUNLIB library. **Note:** The default template member name is DB2LDCTL.

Review the following considerations for other DBMOVER configuration member statements:

- **PC_AUTH=Y.** If you specify PC_AUTH=Y in the DBMOVER member of the PowerExchange Listener, you cannot run bulk load operations under the control of that PowerExchange Listener. Instead, use a netport job that specifies a DBMOVER member without this statement to run bulk load operations.
- **MVSDB2AF=RRSAF.** If you specify MVSDB2AF=RRSAF, specify **JOB** for **Mode Type** in the PWX DB2zOS relational connection parameters to run the bulk load operation as a separate batch job. To run bulk load operations as tasks in the PowerExchange Listener address space, you must specify MVSDB2AF=CAF.
- **SUBMITTIMEOUT.** If the PowerExchange Listener submits batch jobs for bulk load operations, verify that the value of the SUBMITTIMEOUT statement provides sufficient time for the batch job to start execution.
- **MAXTASKS.** If you use netport jobs for bulk load operations, verify that the value of the MAXTASKS statement is sufficient. Each bulk load operation run with a netport job uses three tasks from the MAXTASKS value.

Customizing the PowerExchange Templates for Bulk Data Loads

You can use the JCL and control card templates without modification if you accept the default JCL statements and control cards for the DB2 LOAD utility. Alternatively, you can edit these templates to change the control cards or to use other load utilities.

If the SECURITY statement is set to 0 or 1 in the DBMOVER configuration member, edit the following specifications on the JOB card in the DB2LDJCL or DB2LDJCP template member:

- Remove or change the value of the NOTIFY parameter. Otherwise, the load job may fail.
- Remove the USER and PASSWORD parameters. Otherwise, the load job will fail.

Also, to identify the name of the reject data set, which PowerExchange allocates dynamically, edit the JCL that defines the data set. The skeleton JCL that is provided in DB2LDJCL defines the data set as follows:

```
//SYSDISC  DD  UNIT=SYSDA,SPACE=(CYL,(100,100),RLSE)
```

For example, you can replace the JCL above with the following JCL:

```
//SYSDISC DD DISP=SHR,DSN=%REJECT
```

This example defines a reject data set with the name &HLQ.*taskid*.REJECT.

Notes:

- If the DB2 bulk files are uploaded with "NOSUBMIT", %REJECT is resolved to $HLQ.REJECT, not $HLQ.*taskid*.REJECT.
- To include the output from a failed DB2 LOAD job and the name of the reject data set in the PowerCenter session log, specify RETLOGINFOMSG=Y in the PWX Override connection attribute. The name of the reject data set appears in the PWX-00409 message.

  For more information, see "PWX DB2zOS Connection Attributes for DB2 LOAD Jobs" on page 102.

Customizing the PowerExchange Listener JCL

If you run bulk load operations as a task in the PowerExchange Listener address space, make the following changes to the PowerExchange Listener JCL:

- Comment-out or omit the SYSPRINT DD statement from the PowerExchange Listener JCL, unless you use netport jobs for all bulk load operations. PowerExchange dynamically allocates the SYSPRINT DD statement for the source data that it reads to a data file for the DB2 LOAD utility.
- The DB2 LOAD utility requires the SYSERR DD statement to allocate the work data set for error processing. For example:

```
//SYSERR   DD  UNIT=SYSDA,SPACE=(CYL,(100,100),RLSE)
```

- For tables with indexes, the DB2 LOAD utility requires the following DD statements in order to sort key data:
  - SORTOUT, which is the default DD name for the sort output data set
  - SYSUT1, which is the default DD name for the sort input data set

  Code these two DD statements to allocate temporary data sets large enough to hold the key data for any tables subject to bulk load operations. For example:

```
//SORTOUT  DD  UNIT=SYSDA,SPACE=(CYL,(100,100),RLSE)
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(100,100),RLSE)
```

- For tables with referential constraints, the DB2 LOAD utility requires the SYSMAP DD statement to allocate the work data set for identifying errors. For example:

```
//SYSMAP   DD  UNIT=SYSDA,SPACE=(CYL,(100,100),RLSE)
```

If you customize the DB2 LOAD utility control card template to add additional options, you may require additional DD statements. For more information about DD statement requirements for the DB2 LOAD utility, see the IBM DB2 documentation.

Configuring DB2 Bulk Data Load Sessions in PowerCenter

To control bulk data loads that use the DB2 LOAD utility, you must set the relevant PWX DB2zOS relational database connection attributes for bulk data movement sessions in PowerCenter.

If you use the PowerExchange ODBC drivers, set the relevant ODBC parameters in the odbc.ini file.

PWX DB2zOS Connection Attributes for DB2 LOAD Jobs

The following table describes each connection attribute that you can set to control bulk data load operations that use the DB2 LOAD utility:

PWXPC Connection Attribute	Description	Valid Values
Bulk Load	Select to cause PowerExchange to load data to DB2 for z/OS targets using the DB2 LOAD utility.	Default is No.
CTL Template	Member name of the control file template for the DB2 LOAD utility in the RUNLIB library on the PowerExchange target system.	Default is DB2LDCTL.
JCL Template	Member name of the JCL template for the DB2 LOAD utility in the RUNLIB library on the PowerExchange target system.	Default is DB2LDJCL.
Load Options	Option that indicates whether to use the LOAD REPLACE control card for DB2 LOAD utility bulk data load operations.	Select one of the following values: - INSERT. Select this option to add the new data being loaded to the data currently in the tablespace or tablespace partition. Corresponds to the LOAD RESUME YES control card. - REPLACE. Select this option to replace the data currently in the tablespace or tablespace partition with the new data being loaded. Corresponds to the LOAD REPLACE control card. Default is INSERT.

PWXPC Connection Attribute	Description	Valid Values

Mode Time	Wait period for the execution of the DB2 LOAD utility job.	Enter one of the following values:
		- WAIT. Waits for the job to end before returning control to PowerCenter. Specify this option only with a Mode Type of JOB or TASK.
		- NO WAIT. Returns to PowerCenter without waiting for the job to end. Specify this option only with a Mode Type of JOB or NOSUBMIT.
		- TIMED. Waits the number of seconds specified in the TIME connection attribute before returning control to PowerCenter. Specify this option only if you also specified a Mode Type of JOB.
		- DATAONLY. Creates the data file only. Does not create the files and JCL to run the DB2 LOAD utility. Usually, this option is used with a Mode Type of NOSUBMIT.
		Default is WAIT.
		Notes:
		- If you enter WAIT, PowerExchange disables network operation timeouts for the operation, regardless of the value specified by the TCPIP_OP_TIMEOUT parameter of the PWX Override connection attribute. If you enter TIMED, PowerExchange adds 300 seconds to the value of the TIME connection attribute.
		- If you enter WAIT and the submitted job fails when you run the workflow, the PowerExchange Listener on the z/OS system continues to run. To stop the Listener, perform one of the following actions:
		  - Enter the following command from the z/OS operator console:
		  `F task_name,STOPTASK TASKID=task_id`
		  For more information, see the "STOPTASK Command" topic in the *PowerExchange Command Reference*.
		  - Enter the STOPTASK command in the **Database Row Test** dialog box of the PowerExchange Navigator. Select **TASK_CNTL** in the **DB Type** list, and select **Stop Task** in the **Fetch** box. The **SQL Statement** box displays `stoptask taskid=`. Enter a task ID.
		  For more information, see the topic, "Issuing PowerExchange Listener Commands in a Database Row Test," in the *PowerExchange Navigator User Guide*.

Mode Type	Mode in which PowerExchange runs the DB2 LOAD utility to load data into DB2 tables.	Enter one of the following values: - TASK. Runs the LOAD utility as a subtask under the PowerExchange Listener. - JOB. Submits a separate job to run the DB2 LOAD utility. - NOSUBMIT. Unless Mode Time is DATAONLY, creates the files and JCL to run the DB2 LOAD utility. Does not submit the load job. You must submit the job manually. Default is TASK.

PWXPC Connection Attribute	Description	Valid Values
Time	The wait time in seconds when you select JOB for the Mode Type and TIMED for Mode Time.	Valid values are 1 to 99998. Default is 0.
Filename	Data set name or high-level qualifier that PowerExchange uses to create temporary files for bulk data load operations of the DB2 LOAD utility. For more information, see "Data Set Names for DB2 Bulk Load Operations" on page 104	If you need PowerExchange to create a data file for a single partition of a partitioned table, use the following syntax: *filename*/part*xxxx* The *xxxx* variable is the partition number.
Delete Temporary Files	How PowerExchange should handle the temporary files it creates with the DB2 LOAD utility to load data into a DB2 table.	Select one of the following values: - NO does not delete the temporary files. - BEFORE deletes the temporary files before running the utility. - AFTER SUCCESS ONLY deletes the temporary files after running the utility if it ends with a return code 0. - AFTER deletes the temporary files after running the utility. Default is NO.
PWX Override	PowerExchange connection overrides, separated by semicolons.	To write informational messages, including the output from a failed LOAD job and the name of the DB2 LOAD reject data set, to the PowerCenter session log, specify the following override: RETLOGINFOMSG=Y The name of the reject file appears in the PWX-00409 message. By default, PWXPC writes PowerExchange error and warning messages but not informational messages to the session log.

Note: For DB2 bulk data load operations, specify either **Confirm Write On** or **Confirm Write Off** for the **Write Mode** connection attribute.

For more information about PWX DB2zOS relational connections in PowerCenter, see *PowerExchange Interfaces for PowerCenter*.

Data Set Names for DB2 Bulk Load Operations

PowerExchange creates temporary files for bulk data load operations of the DB2 LOAD utility. The data sets that are generated and their names depend on the **Mode Time** and **Mode Type** connection attributes that you select.

The descriptions of data set names that follow include the following variables:

- *filename*: the high-level qualifier or data set name that you specify for the **Filename** connection attribute.
- *nnnn*: PID of the task.
- *xxxx*: partition number that you specify for the **Filename** connection attribute, using the following syntax: *filename*/part*xxxx*.

The following table describes the files that PowerExchange creates if you select **JOB** for the **Mode Type** connection attribute:

File Name	Description
filename.DTL*nnnnn* or *filename*.PRN*xxxx*	File that contains the data for the DB2 LOAD utility to load to a nonpartitioned table or File that contains the data for the DB2 LOAD utility to load to a partitioned table
filename.DTL*nnnnn*.CTL	File that contains the LOAD commands
filename.DTL*nnnnn*.SQL	File that contains the SQL template for CREATE TABLE statements
filename.DTL*nnnnn*.SYSPRINT	File that contains the output from bulk load jobs

The following table describes the files that PowerExchange creates if you select **NOSUBMIT** for the **Mode Type** connection attribute:

File Name	Description
filename or *filename*.PRN*xxxx*	File that contains the data for the DB2 LOAD utility to load to a nonpartitioned table or File that contains the data for the DB2 LOAD utility to load to a partitioned table
filename.CTL	File that contains the LOAD commands
filename.SQL	File that contains the SQL template for CREATE TABLE statements
filename.SYSPRINT	File that contains the output from bulk load jobs

The following table describes the files that PowerExchange creates if you select **DATAONLY** for the **Mode Time** connection attribute:

File Name	Description
filename or *filename*.PRN*xxxx*	File that contains the data for the DB2 LOAD utility to load to a nonpartitioned table or File that contains the data for the DB2 LOAD utility to load to a partitioned table

Utility ID for DB2 Bulk Load Operations

To generate the utility ID for bulk load operations, the JCL template includes the following parameter:

```
PWX%N5
```

Prior to submitting a job to z/OS, PowerExchange substitutes an incrementing five-digit number for %N5. This method ensures that the utility ID uniquely identifies the job.

Substitution Variables in the Loader JCL Templates

The JCL and control card templates DB2LDCTL, DB2LDCTP, DB2LDJCL, and DB2LDJCP include substitution variables. Prior to submitting a job to z/OS, PowerExchange resolves the defined substitution variables with the appropriate values.

The following table describes these variables:

Variable	Description
%ENDJOB	16-character input string for DTLNTS. Enables the PowerExchange Listener to determine if the job has ended.
%JOBCLASS	Substituted for the job class specified in the loader parameters.
%JOBFILE	Substituted for the name of the file that contains the JCL skeleton used to build the job.
%N*n*	Substitutes an incrementing number of 1 to 7 significant digits.
%PARTDDN	Generates the DDNAME for the input data for a partition. Has the form PAR*nnnn*, where *nnnn* represents the partition number. The skeleton JCL can contain multiple instances of this variable. Unused instances are set to PRTnnnn.
%PN*n*	Substitutes a partition number of 1 to 4 significant digits only if the load is for a partitioned table. Do not use with a load JCL skeleton for a non-partitioned table.
%PWD	Password. Will appear in the plaintext in the job.
%SSID	DB2 subsystem ID of the load target.
%STRTJOB	16-character input string for DTLNTS. Enables the PowerExchange Listener to determine if the job has started.
%SYSIN	Substitutes the name of the file containing the loader control statements.
%SYSPRINT	Substitutes the name of the file to which the loader messages are written.
%SYSREC	Substitutes the DSNAME for the input data. For unused instances in a partitioned load, this value is set to NULLFILE.
%USER	User ID.

DB2 LOAD Utility Options

Unless you select **DATAONLY** for the **Mode Time** connection attribute, PowerExchange uses the DB2LDCTL or DB2LDCTP control card template to generate a load control file for the DB2 LOAD utility. If you select **DATAONLY** for the **Mode Time** connection attribute, you must provide the load control file.

PowerExchange generates the following load options for nonpartitioned tables:

LOAD DATA RESUME NO REPLACE INTO TABLE *table_name*

or

LOAD DATA RESUME YES INTO TABLE *table_name*

PowerExchange generates the following load options for partitioned tables:

LOAD DATA INTO TABLE *table_name* PART *n* INDDN PAR0000*n* REPLACE

or

LOAD DATA INTO TABLE *table_name* PART *n* INDDN PAR0000*n* RESUME YES

Bulk Loads to a Single Partition and Multiple Partitions

You can use the DB2 LOAD utility with PowerExchange to load data to a table partition. As with a nonpartitioned table, PowerExchange can produce customized JCL and load cards for running the LOAD utility, generate a data file that contains the data to be loaded, and automatically submit the bulk load job to load data to the target.

However, PowerExchange cannot generate and submit a single bulk load job to load data to multiple table partitions. To write to multiple partitions in a single load job, you can define a separate target in PowerCenter for each partition and define connection attributes to create only a data file for the DB2 LOAD utility to load for each target. You can provide the loader control file and JCL for a single DB2 LOAD job that loads the data into each partition. In the PowerCenter workflow, after the data is written to a data file for each partition, you can include a separate PowerCenter command that users DTLREXE to submit the DB2 LOAD job. Alternatively, you can submit the job outside of PowerCenter by using a job scheduler or some other facility.

CHAPTER 8

IDMS Bulk Data Movement

This chapter includes the following topics:

Introduction to IDMS Bulk Data Movement

PowerExchange can read bulk data from a CA IDMS source on a z/OS system. However, PowerExchange cannot write bulk data to an IDMS target.

PowerExchange treats IDMS as a nonrelational DBMS. Consequently, you must create a data map for an IDMS data source from the PowerExchange Navigator. PowerExchange uses the data map to access IDMS source data and metadata to create a relational view of the source records for processing.

Because the IDMS database runs on a z/OS system, it is remote from the system or systems on which the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run. Therefore, you must run an additional PowerExchange Listener on the remote z/OS system and verify that PowerExchange and PowerCenter can communicate with it.

Also, PowerExchange might require that the IDMS load libraries or the PowerExchange copies of these load libraries be APF-authorized, depending on how you set some DBMOVER configuration statements.

Configuring PowerExchange for IDMS

To configure PowerExchange for bulk data movement operations, the PowerExchange administrator completes the following tasks:

- Set up and test connectivity to the PowerExchange Listener on the remote system.
- Configure IDMS bulk data movement by defining IDMS-specific statements in the DBMOVER configuration member on the IDMS source or target system.
- Assess security requirements for accessing IDMS data and configure PowerExchange accordingly.
- If the first parameter in the SECURITY statement is set to 2 in the DBMOVER configuration member, set up netport jobs for IDMS bulk data movement.

PowerExchange for IDMS Security

Security requirements primarily depend on how you set the first parameter of the SECURITY statement in the DBMOVER configuration member on the z/OS system.

If you set the first parameter of the SECURITY statement to 1 or 2, the PowerExchange Listener must run APF-authorized. However, IDMS load libraries are usually *not* APF-authorized. To handle this situation, use either of the following methods:

- Run the XIDIDM10 job during PowerExchange installation, and APF-authorize the copies of the IDMS load libraries that the XIDIDM10 job creates. Then verify that these libraries are specified in the PowerExchange Listener STEPLIB DD statement.
- Set the PC_AUTH statement to Y in the DBMOVER configuration member on the z/OS system. This setting causes PowerExchange to use an z/OS Program Call (PC) services routine to get the APF-authorization that the PowerExchange Listener requires.

  Note: You must define the DBMS load libraries in the DTLLOAD DD statement instead of in the STEPLIB DD statement. If you use netport jobs to access data, define the load libraries in the DTLLOAD DD statement of the netport JCL.

  Use this method if you do not want to maintain and APF-authorize copies of the IDMS load libraries.

If you set the first parameter of the SECURITY statement to 2, you must also set up a netport job for each IDMS bulk data request for security checking to be properly performed.

If you set the first parameter of the SECURITY statement to 0, the PowerExchange Listener does not need to run APF-authorized. In the STEPLIB DD statement of the PowerExchange Listener JCL, you can specify the IDMS load libraries rather than the PowerExchange copies of these libraries.

Note: Some DBMOVER configuration statements other than SECURITY, for example, STATS, might require some functions to be APF-authorized.

Related Topics:

- "APF-Authorizing Copies of IDMS Load Libraries" on page 112
- "Using the z/OS Program Call Services Routine for APF-authorized Access" on page 112
- "Setting Up a Netport Job for IDMS Bulk Data Movement" on page 113

Setting Up and Testing Connectivity to a Remote IDMS Source

To access an IDMS data source on a remote z/OS system, PowerExchange and PowerCenter must be able to communicate with the PowerExchange Listener on the remote system.

To set up and test connectivity to a remote IDMS source:

1. On the systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the **ping** command to test network access to the remote system.
2. Add the NODE statement to the dbmover.cfg file on each PowerExchange or PowerCenter system to identify the remote PowerExchange Listener and its port:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   Tip: To determine the port number, consult with the person in your organization who installed the PowerExchange Listener on the remote system.
3. Run the DTLREXE ping utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Adding IDMS-Specific Statements to the DBMOVER Configuration Member

To configure IDMS bulk data movement, you can include the following IDMS-specific statements in the DBMOVER configuration member on the IDMS source or target system:

- LOADJOBFILE
- PC_AUTH
- TEMPHLQ

After editing the DBMOVER configuration member, you must restart the PowerExchange Listener for your changes to take effect.

Note: When you set the UNIT and VOLSER statements, enter values that enable the temporary data sets for IDMS metadata to be allocated by the IDMSMJCL job.

For descriptions of all DBMOVER statements, see the *PowerExchange Reference Manual*.

LOADJOBFILE Statement

The LOADJOBFILE statement specifies the PDS data set that contains the JCL template member for DB2 for z/OS LOAD utility and CA IDMS/DB metadata retrieval batch jobs.

Operating Systems: z/OS

Data Sources: CA IDMS/DB and DB2 for z/OS

Related Statements: SUBMITTIMEOUT

Required: No

Syntax:

```
LOADJOBFILE={pds_name|A}
```

Value: For the *pds_name* variable, enter the PDS data set that contains the JCL template member for DB2 for z/OS LOAD utility and CA IDMS/DB metadata retrieval batch jobs. For DB2 LOAD utility operations, PowerExchange reads this data set on the system where you perform the bulk load. Default is A.

Usage Notes:

- When you install PowerExchange, the z/OS Installation Assistant includes your RUNLIB data set name in the LOADJOBFILE statement in the DBMOVER member.
- PowerExchange provides the following JCL template members in RUNLIB:
  - **DB2LDJCL**. Sample JCL for DB2 LOAD utility jobs for nonpartitioned tables.
  - **DB2LDJCP**. Sample JCL for DB2 LOAD utility jobs for partitioned tables.
  - **IDMSMJCL**. Sample JCL for CA IDMS/DB metadata retrieval.
  - **IDMSMJCX**. Sample JCL for CA IDMS/DB metadata retrieval that creates a temporary load library for the subschema load module.
- By default, a PowerExchange Listener waits for 60 seconds for the spawned jobs to start. You can increase this timeout period by defining the SUBMITTIMEOUT statement. If the batch job does not start in the timeout period, PowerExchange times out the job, stops the task in the PowerExchange Listener, and writes the PWX-00426 message to the PowerExchange message log.

PC_AUTH Statement

The PC_AUTH statement controls whether the PowerExchange Listener uses its MVS Program Call (PC) services routine to acquire the authorization to access DBMS load libraries.

Operating Systems: z/OS

Data Sources: All

Required: No

Syntax:

```
PC_AUTH={N|Y}
```

Valid Values:

- **N.** The PowerExchange Listener runs APF-authorized, and you must include APF-authorized copies of the DBMS load libraries in the STEPLIB DD statement of the PowerExchange Listener.
- **Y.** The PowerExchange Listener runs APF-authorized, and the PowerExchange Listener uses the PowerExchange-provided Program Call (PC) services routine to acquire the necessary authorization to access the DBMS load libraries.

  Note: You must define the DBMS load libraries in the DTLLOAD DD statement instead of in the STEPLIB DD statement. If you use netport jobs to access data, define the load libraries in the DTLLOAD DD statement of the netport JCL.

  Specify Y if you do not want to maintain any APF-authorized PowerExchange copies of the DBMS load libraries.

Default is N.

Usage Note:

- Use the PC_AUTH DBMOVER statement and the DTLLOAD DD statement to access load libraries that are not required to be APF-authorized. For example, use these statements to access CA IDMS/DB load libraries or to access a user-defined program that is invoked by the CallProg function in a data map to process source data in a record.

TEMPHLQ Statement

The TEMPHLQ statement overrides the high-level qualifier that PowerExchange uses by default when creating a temporary file for CA IDMS/DB metadata.

If you do not want PowerExchange to create data sets with the PowerExchange Listener user ID, define this statement. PowerExchange ignores the TEMPHLQ statement when you specify 2 for the first parameter of the SECURITY statement.

Operating Systems: z/OS

Data Sources: CA IDMS/DB

Related Statements: SECURITY

Required: No

Syntax:

```
TEMPHLQ=hlq
```

Value: For the *hlq* variable, enter the high-level qualifier (HLQ) for temporary data sets that PowerExchange creates for CA IDMS/DB metadata. A valid value is a 1- to 17-character string. By default, PowerExchange uses the PowerExchange Listener user ID as the HLQ for the metadata temporary data sets.

To use the TEMPHLQ statement, you must also specify 0 or 1 in the first parameter in the SECURITY statement.

Example: If you define the following SECURITY and TEMPHLQ statements:

```
SECURITY=(0,N)
TEMPHLQ=B.C
```

PowerExchange creates the following data set during the IDMS copybook import process to hold the CA IDMS/DB metadata:

```
B.C.METADATA.DTL000001
```

APF-Authorizing Copies of IDMS Load Libraries

To meet the PowerExchange APF-authorization requirement, you can APF-authorize the copies of the system IDMS load libraries that are optionally created during PowerExchange installation.

Warning: If you apply maintenance or schema changes to the system IDMS load libraries, you must also apply these changes to the copies of the load libraries or re-create the copies by rerunning the XIDIDM10 job. Otherwise, the bulk data movement might not preserve data integrity.

If you do not want to maintain and APF-authorize copies of the IDMS load libraries, you can use the MVS Program Call services routine that PowerExchange provides to meet the APF authorization requirement.

To APF-authorize copies of IDMS load libraries:

1. During PowerExchange installation, add the following statement to the SETUPIDM JCL before you run the job:

   ```
   //STEP1 EXEC SUBMJOB,MEMBER=XIDIDM10
   ```

   The XIDIDM10 job creates copies of the IDMS load libraries under the default names of *hlq*.IDMS.LOADLIB and *hlq*.IDMS.DBA.LOADLIB.
2. APF-authorize the IDMS load library copies that were created by the XIDIDM10 job at PowerExchange installation.
3. Verify that the STEPLIB DD in the PowerExchange Listener JCL includes the *hlq*.IDMS.LOADLIB and the *hlq*.IDMS.DBA.LOADLIB libraries.

   If you run the PowerExchange Listener as a batch job, the JCL is in the STARTLST member of the RUNLIB library. If you run the PowerExchange Listener as a started task, the JCL is in the PWXLSTNR member of the PROCLIB library.
4. Verify that all other libraries in the STEPLIB DD statement of the PowerExchange Listener JCL are APF-authorized. Lack of APF-authorization for some of these libraries negates PowerExchange authorization to access IDMS data through the PowerExchange Listener and will cause the PowerExchange Listener to not start.

Related Topics:

- "Using the z/OS Program Call Services Routine for APF-authorized Access" on page 112

Using the z/OS Program Call Services Routine for APF-authorized Access

To meet the PowerExchange authorization requirement for running the PowerExchange Listener, you can use the z/OS Program Call (PC) services routine that PowerExchange provides. The z/OS PC services routine works with the system IDMS load libraries instead of the PowerExchange copies of these libraries. Use the z/OS PC services routine if you do *not* want to maintain and APF-authorize PowerExchange copies of the

IDMS load libraries. You still must APF-authorize all libraries in the STEPLIB DD statement of the PowerExchange Listener JCL. Also, you must specify the system IDMS load libraries in the DTLLOAD DD statement of the PowerExchange Listener JCL.

To use the z/OS PC services routine for APF-authorized access:

1. Add a DTLLOAD DD statement to the PowerExchange Listener JCL and specify the system IDMS load libraries in this DD.

   If you run the PowerExchange Listener as a batch job, the JCL is in the STARTLST member of the RUNLIB library. If you run the PowerExchange Listener as a started task, the JCL is in the PWXLSTNR member of the PROCLIB library.
2. Add the following statement to the DBMOVER configuration member of the RUNLIB library: `PC_AUTH=Y`. This statement directs PowerExchange to use the z/OS PC services routine.
3. If applicable, remove the copies of the IDMS load libraries that were optionally created at PowerExchange installation, *hlq*.IDMS.LOADLIB and *hlq*.IDMS.DBA.LOADLIB, from the PowerExchange Listener STEPLIB DD statement. The z/OS PC services routine uses the system IDMS load libraries instead.
4. Verify that every library in the STEPLIB DD statement of the PowerExchange Listener JCL is APF-authorized.

Related Topics:

- "APF-Authorizing Copies of IDMS Load Libraries" on page 112

Setting Up a Netport Job for IDMS Bulk Data Movement

If the SECURITY statement in the DBMOVER configuration member is set to 2, you must use a netport job to run an IDMS bulk data movement job. A SECURITY setting of 2 requires your z/OS security management system, for example, RACF, to verify the logon user ID of each user application that attempts to access IDMS data.

To set up a netport job for access to an IDMS data source, use the sample netport job in the TAPEJCL member of the RUNLIB library as a template.

To set up a netport job for IDMS bulk data movement:

1. Copy the TAPEJCL member to create a basis for the IDMS netport job.
2. If you are not using the z/OS PC services routine, verify that the PowerExchange copies of the IDMS load libraries, *hlq*.IDMS.LOADLIB and *hlq*.IDMS.DBA.LOADLIB, are specified in the STEPLIB DD statement of the netport JCL.
3. Add the following DD statement to the IDMS netport job:

   ```
   //SYSIDMS    DD DSN=&HLQ..RUNLIB(DTLDMCL),
   //              DISP=(SHR)
   ```
4. Verify that the relevant SYSCTL statement is included in the netport JCL if you are running an IDMS Central Version, or verify that the following DD statements are included in the netport job if you are running IDMS in Local mode:

   ```
   //IDMSDCT    INCLUDE MEMBER=IDMSDICT
   //IDMSFIL    INCLUDE MEMBER=IDMSFILE
   ```

   Note: The IDMSDICT and IDMSFILE members require the relevant dictionary definitions and database file definitions.
5. In the DBMOVER configuration member, edit the LISTENER and NETPORT statements that are supplied for TAPEJCL. In each statement, enter a valid port number for the netport job. In the NETPORT statement, assign the netport job member to the specified port.

If you want to run a database row test from the PowerExchange Navigator to read IDMS data, also add a NODE statement to the dbmover.cfg on the system where the PowerExchange Navigator runs, for example:

```
NODE=(idmsnet,TCPIP,ipaddress_or_hostname,port_number)
```

Edit other statements, as needed.

6. If you are using the z/OS PC services routine, specify `PC_AUTH=Y` in the DBMOVER configuration member that the netport job uses. Also add a DTLLOAD DD statement to the netport job that points to the system IDMS load libraries, not to the PowerExchange copies of these libraries.
7. Restart the PowerExchange Listener so that PowerExchange detects the netport job.

Moving IDMS Bulk Data

Use the following procedure to move a IDMS bulk data either by using the PowerExchange Listener or a netport job. This procedure assumes that you are using PWXPC to integrate PowerExchange with PowerCenter.

Before you begin, gather the following information:

- TCP/IP address of the host that contains the IDMS source database
- Port number of the PowerExchange Listener and port number of the netport job
- IDMS metadata, including the subschema name and database name
- z/OS user ID and password, if required by the PowerExchange SECURITY statement setting

To move IDMS bulk data:

1. From the PowerExchange Navigator, create a data map. Select **IDMS** as the access method. To import metadata about records and fields, also select the **Import Record Definitions** option.

   Tip: If you do not select the **Import Record Definitions** option, you can import the metadata later. Open the data map in the Resource Explorer and click **File** > **Import Copybook**.
2. Open the data map in the Resource Explorer, and click the nonrelational “record” or relational “table” view to verify that the metadata was successfully retrieved.
3. Send the data map to the PowerExchange Listener on the remote IDMS system. In the Resource Explorer, select the data map and click **File > Send to Remote Node**. This step enables PowerExchange extraction routines to access the data map at runtime.

   In the **Location** field, enter the PowerExchange Listener node name that is specified in the NODE statement of the dbmover.cfg file on the local PowerExchange system. Complete the other fields, as needed.

   Note: If you do not complete this step, you are prompted to send the data map to the remote node when you run a database row test.
4. Run a database row test on the table view of the metadata to test that data can be returned from the IDMS source database. Use the node name that is specified for the netport job if the SECURITY statement is set to 2. Otherwise, use the node name that is specified for the PowerExchange Listener.
5. In the PowerCenter Designer, import the PowerExchange data map. Click **Sources > Import from PowerExchange**. Then enter the following required attributes:
   - In the **Location** field, enter the PowerExchange Listener node name that you specified in the NODE statement of the dbmover.cfg file.

- In the **User name** and **Password** fields, enter your z/OS user ID and password, if required by the SECURITY statement setting.
- In the **Source Type** list, select **IDMS**.

  Complete the optional attributes as needed. Then connect to PowerExchange and import the data map.

6. In the PowerCenter Designer, create a mapping.
7. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX NRDB Batch application connection. Then start the workflow to perform the bulk data movement.

CHAPTER 9

IMS Bulk Data Movement

This chapter includes the following topics:

Introduction to IMS Bulk Data Movement

PowerExchange, in conjunction with PowerCenter, can move bulk data from or to an IMS database.

You must run a PowerExchange Listener on the z/OS system where the IMS database is located. Verify that this PowerExchange Listener can communicate with the PowerExchange system where the PowerExchange Navigator and PowerCenter Integration Service run.

Because IMS is a nonrelational database, you must create a data map by using the PowerExchange Navigator. PowerExchange uses data maps for the following purposes:

- Access IMS data and metadata.
- Create a relational row-type view of the records.

  PowerExchange requires a relational view to use SQL-type statements to read or write bulk data.

If an IMS database descriptor (DBD) is available, you can import it from the DBDLIB library. Alternatively, PowerExchange can get the DBD metadata for source objects from the IMS catalog. When you create a data map, import the DBD metadata to define the following items:

- Segments
- Segment hierarchy
- Key fields
- Search fields

You can import DBD metadata for all or selected segments. For each IMS segment for which you import DBD metadata, you can also import a COBOL or PL/I copybook. The copybook provides detailed segment layout information, which augments and overlays the segment information from the DBD.

You can optionally create an IMS unload data set that contains the source data and then use that data set for the bulk data movement operation. To create an unload data set that PowerExchange and PowerCenter can use, use either the IBM unload utility for your IMS database type or the Unload function of the BMC Database Management for IMS solutions.

Methods of Accessing IMS Data

To access IMS data, use one of the following methods:

- Run a netport job as a DL/I Batch or BMP job to access an IMS database. You must create a DL/1 Batch data map.
- Use the PowerExchange Listener to access either an IMS database or IMS unload data set.
  - To access an IMS database, you must create an IMS ODBA data map. PowerExchange uses the Open Database Access (ODBA) interface to access the IMS data.
  - To access an IMS unload data set, create either an IMS ODBA or DL/1 Batch data map.

    You do not need to create a netport job if you use the PowerExchange Listener.

Group Source Processing in PowerExchange

Group source processing in PowerExchange reads data stored in the same physical source in a single pass, which enhances throughput and reduces resource consumption by eliminating multiple passes of the source data.

For an IMS unload data source, you can use Designer to import a data map with multiple record types to create a PowerCenter source definition. If you want the source definition to represent only a single record type, import a single table from the data map. If you want the source definition to include all record types, import the data map as a multi-record data map.

When you run a session that contains a mapping with source definitions for each table in a multi-record data map, PowerExchange reads the data set or file once for each source definition. When you run a session that contains a mapping with a single source definition for all records in a multi-record data map, PowerExchange uses group source processing to read all of the records in the data set or file in a single pass.

For more information, see *PowerExchange Interfaces for PowerCenter*.

IMS Catalog Use

PowerExchange requires access to IMS database definitions (DBDs) in both source format and DBDLIB (DBGEN) format to get metadata for source objects. You can configure PowerExchange to access this information directly from the IMS catalog instead from the DBDLIB library.

When properly configured, PowerExchange can transparently get DBD information for IMS source objects from the IMS catalog by using the following IMS tools:

- **IMS catalog API**. This API consists of the DFS3CATQ assembly program in the IMSxx.SDFSRESL.RESLIB library and the DFS3CATQ macro in the IMSxxx.SDFSMAC library. The API gets DBD information in DBGEN format directly from the IMS catalog for IMS unload processing. You must provide the high-level qualifier of the bootstrap data set if the IMS control region is not running or if you are using an IMS version earlier than IMS 15.
- **IMS Catalog Library Builder Utility, DFS3LU00**. The utility gets DBD information in source format from the IMS catalog and writes this DBD information to a PDSE that you pre-allocated. When you create data maps in the PowerExchange Navigator, PowerExchange retrieves the source metadata from the PDSE. After the metadata is imported to the Navigator machine for creating data maps, PowerExchange deletes the information from the PDSE so that no PDSE maintenance is required.

Use of the IMS catalog is optional in IMS and PowerExchange. However, certain IMS functionality, such as database versioning and the management of run-time application control blocks requires the IMS catalog. For more information see the IBM IMS documentation.

For PowerExchange to use the IMS catalog, you must complete several configuration tasks. For more information, see "Configuration Tasks for Using the IMS Catalog" on page 118.

Configuring IMS Bulk Data Movement

The tasks that the PowerExchange administrator completes for IMS bulk data movement operations partially depend on the database access method.

General Configuration Considerations

- To reduce the number of locking conflicts, verify that the IMS program communications block (PCB) for each source IMS database is defined with the keyword PROCOPT=GO*x*. This setting provides read-only access to the IMS database.
- If you plan to use IMS ODBA or a netport BMP job to access an IMS database, verify that the program specification block (PSB) is defined in the IMS SYSGEN.
- PowerExchange does not support PSBs that were generated with the LANG=PLI keyword.
- If you use an IMS HDAM, DEDB, or PHDAM database as a data source or lookup, PowerExchange cannot fully optimize database access. PowerExchange assumes that data is stored in random order, even if you store the data in sequential order.

  PowerExchange determines an access path that ensures that all candidate records are selected. For HDAM, DEDB, and PHDAM databases, PowerExchange accesses records sequentially and disregards any non-matching records. In the SELECT statements that PowerExchange builds for accessing data, you should include a WHERE clause with KEY=*value*.

Configuration Tasks for Using the IMS Catalog

You can configure PowerExchange to get DBD metadata for source objects from the IMS catalog.

To use the IMS catalog, perform the following configuration tasks:

1. Allocate a partitioned data set extended (PDSE) to hold output from the IMS Catalog Library Builder Utility, DFS3LU00. For more information, see "Allocate a PDSE for the IMS Catalog Builder Utility Output" on page 119.
2. Customize the PowerExchange Listener JCL or PROC to specify the DD statements required for IMS catalog use. For more information, see "Customize the PowerExchange Listener JCL for IMS Catalog Use" on page 119.
3. In the DBMOVER configuration member on the PowerExchange Listener machine, define the following required statements:
   - IMSID. You must specify an *ims_ssid* value that matches the *ims_ssid* in the IMSBSDS statement.
   - IMSBSDS. Make sure the *ims_ssid* in this statement matches the *ims_ssid* in the IMSID statement.
   - LU00FILE. Enter the name of the PDSE that you created to store DBD output from the IMS Catalog Library Builder Utility, DFS3LU00.

   For more information, see "DBMOVER Configuration File" in the *PowerExchange Reference Manual*.
4. When you add data maps for IMS source objects in the PowerExchange Navigator, be sure to configure the following fields:
   - On the **Name** page, select **Import Record Definitions**.
   - On the **DL/1 Batch Access Method** page, enter an IMS SSID value that matches the *ims_ssid* value that you specified in the IMSBSDS statement in the DBMOVER configuration file.
   - On the **Import Copybook - Source Details** page, select **Remote** under **Source** and select **DBD** in the **Type** field.

- On the **Import Copybook - Remote DBD Details** page, in the **File Name** field, enter the name of a PDS along with the DBD member name that you want to look for in the IMS catalog.

Note: If the DBD source information in the IMS catalog defines a field with an EXTERNALNAME value but no NAME value, for example, `FIELD EXTERNALNAME=external_name`, the PowerExchange Navigator uses the EXTERNALNAME value as the field name in the data map record. This field name will not appear in the **Search Fields** list for the record and cannot be used in the Segment Search Arguments (SSAs) that PowerExchange creates.

Allocate a PDSE for the IMS Catalog Builder Utility Output

If you use the IMS Catalog Builder Utility, DFS3LU00, to create database descriptors (DBDs) from the IMS catalog for the IMS system, allocate an extended partitioned data set (PDSE) for the utility output. PowerExchange reads DBD information from the PDSE for source objects when you create data maps.

The data set must have the following DCB characteristics:

- Record format (RECFM) of F for fixed-length
- Record length (LRECL) of 80
- Block size (BLKSIZE) of 80

Also, the data set name must match the PDSE member name specified in the LU00FILE statement in the DBMOVER configuration file.

For example, you could allocate a data set with the following characteristics:

```
Data Set Name  . . . . : USER1.IMS.DFS3LU

General Data                          Current Allocation
 Management class . . : DEFAULT        Allocated blocks  . : 15,360
 Storage class  . . . : STANDARD       Allocated extents . : 1
  Volume serial . . . : Z6SMLF         Maximum dir. blocks : NOLIMIT
  Device type . . . . : 3390
 Data class . . . . . : DEFAULT
  Organization  . . . : PO            Current Utilization
  Record format . . . : F              Used pages  . . . . : 11
  Record length . . . : 80             % Utilized  . . . . : 3
  Block size  . . . . : 80             Number of members . : 0
  1st extent blocks . : 15360
  Secondary blocks  . : 13
  Data set name type  : LIBRARY       Dates
  Data set version  . : 1              Creation date . . . : 2019/06/20
                                       Referenced date . . : 2019/08/30
                                       Expiration date . . : ***None***
```

Note: After data maps are successfully created, PowerExchange automatically deletes DBD information from the PDSE. No PDSE maintenance is required.

Customize the PowerExchange Listener JCL for IMS Catalog Use

You must edit the PowerExchange Listener JCL or PROC to add the DD statements that define the data sets used to get DBD information for IMS source objects from the IMS catalog. PowerExchange transparently

uses the IMS catalog API, DFS3CATQ, to get DBD information in DBDGEN format and uses the IMS Catalog Library Builder Utility, DFS3LU00, to get DBD information in source format.

Add the following DD statements to the Listener JCL or PROC:

DD Name	Description
DD DBDSOR	Specifies the PDSE to which the IMS Catalog Library Builder Utility, DFS3LU00, writes DBD information in source format for IMS source objects. You must pre-allocate this PDSE. Make sure that the PDSE name specified in this DD statement matches the PDSE name in the LU00FILE statement in the DBMOVER configuration file. When you create data maps, the source object metadata stored in this PDSE is sent to the local machine where you run the PowerExchange Navigator or infacmd pwx createdatamaps utility. **Note:** After the data maps are successfully created, PowerExchange automatically deletes DBD information from the PDSE. No PDSE maintenance is required.
DD DSN =IMS*xxx*.SDFSRESL	Defines the IMS*xxx*.SDFSRESL library that contains the IMS catalog API DFS3CATQ program and macro.
LUSYPRT DD SYSOUT=*	Defines the data set that contains statistics from IMS Catalog Library Builder utility, DFS3LU00, activity.
SYSPRINT DD SYSOUT=*	Defines the data set that contains control cards that the PowerExchange Navigator passed to the IMS Catalog Library Builder utility, DFS3LU00.

Sample JCL:

```
//STEPLIB  DD  DSN=CEE.SCEERUN,DISP=SHR
//         DD  DSN=&HLQ..LOADLIB,DISP=SHR
//         DD DSN=DSNB10.DSNB.SDSNEXIT,DISP=SHR
//         DD DSN=DSNB10.SDSNLOAD,DISP=SHR
//*        DD DSN=DSN910.SDSNEXIT,DISP=SHR
//*        DD DSN=DSN910.SDSNLOAD,DISP=SHR
//*        DD DSN=ICU.PMICU.LOAD,DISP=SHR
//* IF USING IMS THEN CUSTOMIZE BELOW
//        DD DSN=IMS1510.SDFSRESL,
//           DISP=SHR
//LUSYSPRT DD  SYSOUT=*
//SYSABEND DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//CEEDUMP  DD  SYSOUT=*
//DBDSOR   DD  DISP=OLD,DSN=GRAHAM.IMS.DFS3LU,
//             DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
```

Configuring PowerExchange for Bulk Data Movement with a DL/1 Batch Data Map

Use the following procedure to set up PowerExchange for bulk data movement operations that use a DL/1 Batch data map and a netport job. A netport job can run either as an IMS DL/I Batch or BMP job.

To configure PowerExchange for bulk data movement with a DL/1 batch data map:

1. Create one or more netport jobs for accessing IMS metadata. Use the sample netport job in the IMSJCL member of the RUNLIB library as a template. Customize this member so that it runs in your environment as either a DL/I Batch or BMP job.

2. In the DBMOVER configuration member on the MVS system where the IMS database is located, uncomment and customize the LISTENER and NETPORT statements for the IMSJCL member. Use the following syntax:

```
LISTENER=(node_name,TCPIP,port_number)
NETPORT=(node_name,port_number,,,"hlq.RUNLIB(ims_netport_member)",psb_id)
```

Where:

- *node-name* is a name that you define for the PowerExchange Listener that runs on the IMS source or target system. Specify the same node name in both the LISTENER and NETPORT statements. Also, define a NODE statement that contains the same node name in the dbmover.cfg file on the Windows system or systems on which you run the PowerExchange Navigator and PowerCenter Integration Service.
- *port_number* is a valid port number at your site. To get a valid port number, consult with your network administrator. Specify the same port number in both LISTENER and NETPORT statements.
- *psb_id* is a valid IMS PSB identifier for the IMS source or target.
- *hlq*.RUNLIB.*ims_netport_member* is the fully-qualified data set name and member name for the member where the IMS netport job is stored. You created this member based on the sample IMSJCL member.

Restriction: You must run the specified PowerExchange Listener and netport job on the same MVS image. Otherwise, the netport job fails.

If you plan to move bulk data for multiple PSBs, enter a pair of LISTENER and NETPORT statements for each PSB.

3. Restart the PowerExchange Listener after you finish making changes to the DBMOVER configuration member.

Configuring PowerExchange for Bulk Data Movement with an IMS ODBA Data Map

Use the following procedure to set up PowerExchange for bulk data movement operations that use an IMS ODBA data map.

1. Verify that the IMS RESLIB data set that contains the IMS load modules is included either in the STEPLIB DD concatenation of the PowerExchange Listener JCL or in the LNKLST library. The PowerExchange Listener JCL is in the PWXLSTNR or STARTLST member of the RUNLIB library.
2. In the DBMOVER configuration member on the system where an IMS source or target resides, add the following ODBASUPP statement to enable PowerExchange to use ODBA:

```
ODBASUPP=YES
```

For more information about this statement, see the *PowerExchange Reference Manual*.

3. Verify that the IMS PSB has named PCBs. For ODBA access, a PCB name is required.

Setting Up and Testing Connectivity to an IMS Source or Target

To access an IMS data source or target on a remote z/OS system, PowerExchange must be able to communicate with the PowerExchange Listener on the remote system. Use the following procedure to configure and test connectivity to the remote PowerExchange Listener.

To set up and test connectivity to an IMS source or target:

1. On the systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the **ping** command to test network access to the remote z/OS system.

2. Add the NODE statement to the dbmover.cfg file on each PowerExchange and PowerCenter system to identify the remote node where the IMS source or target is located. Use the following syntax:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   If you plan to use a DL/1 Batch data map, this statement points to the z/OS system where the batch netport job for the bulk data movement will run.

   If you plan to use an IMS ODBA data map, this statement points to the z/OS system where the remote PowerExchange Listener runs.

   Tip: To determine the port number, consult with the person in your organization who installed the PowerExchange Listener on the remote system.

3. Run the DTLREXE utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Implementing an IMS Bulk Data Movement

To prepare for moving IMS bulk data, gather the prerequisite information and review the implementation considerations. Then perform the procedure for moving bulk data.

Prerequisite Information

Gather the following information before you implement a bulk data movement operation:

- IMS subsystem identifier (SSID)
- For a DL/I batch data map: IMS PCB number
- For an IMS ODBA data map: Name of the IMS PSB, and name of the PCB in the PSB
- Fully-qualified data set name and member name of the IMS DBD source library
- Fully-qualified data set names and member names of the copybooks for the IMS segments
- IMS segments to be accessed
- z/OS user ID and password, if required by the PowerExchange SECURITY statement setting
- For DL/I batch netport jobs: Each database data set name that is specified in the z/OS DD and DSN statements of the netport JCL or in the IMS dynamic allocation libraries in the STEPLIB DD

Implementation Considerations

Review the following considerations before implementing a bulk data movement.

General Considerations

For IMS bulk data movement, be aware of the following considerations.

- If a data map does not represent a complete IMS segment, as defined in the DBD, a PowerExchange INSERT operation on a segment that was defined without the FILLER keyword might cause non-blank data to be written to the end of the segment. You can avoid this problem by adding FILLER to the COBOL copybook for the segment before importing the segment metadata into PowerExchange.
- In the PowerExchange Navigator, the term *record* refers to an IMS segment.
- The PCB value that you use for the IMS database depends on which access method you select, as follows:
  - If you select the DL/1 batch access method in the data map, use the PCB number that indicates the PCB offset in the PSB. If you generated the PSB with CMPAT=YES, add 1 to this PCB number.
  - If you select the IMS ODBA access method in the data map, use the PCB name that you specified either in the PCBNAME parameter or columns 1 through 8 of the PCB statement when generating the PSB.
- Import metadata from the IMS DBD source to define the segments, segment hierarchy, key fields, and search fields for an IMS database. Also import a COBOL or PL/I copybook for each segment to overlay key fields and search fields from the DBD and to define all of the other fields. PowerExchange updates the data map with the copybook information while maintaining the IMS hierarchical structure from the DBD.
- If you are moving bulk data from an IMS source that has one or more segments without a key field or with a non-unique key, and you need a unique identifier for each IMS segment, you can define an expression in a user-defined field in the data map to return an RBA for each of these segments. To add the segment RBA to segment data in an IMS data map, Informatica recommends that you use the GetIMSRBAByLevel function rather than the GetDatabaseKey function. The GetIMSRBAByLevel function enables you to get the RBA of an unkeyed or non-unique keyed parent segment. You can then use this RBA to build a unique key in the target.

  If you plan to perform change data capture for the IMS source after materialization is complete, PowerExchange can use the RBA for IMS synchronous CDC but cannot use the RBA for IMS log-based CDC. For IMS log-based CDC, PowerExchange cannot use the GetDataBaseKey or GETIMSRBAByLevel functions in expressions in the data map.
- When you run a database row test on an IMS data map, the node name that you specify in the **Location** field depends on the access method that you select in the data map and the type of IMS source.
- When you run a database row test on a table view of a data map, the PowerExchange Navigator generates SELECT statements that are appropriate for the database type.

  For example:
  - If you select NRDB2 or IMSUNLD in the **DB Type** list in the database row test, the generated SQL appears in the **SQL Statement** box with the underscore (_) character, as follows:

    ```
    SELECT * FROM IMSUNLD.DBLOG5OF_DB#AASEG
    ```
  - If you select NRDB in the **DB Type** list, the generated SQL appears in the **SQL Statement** box with the period (.) character, as follows:

    ```
    SELECT * FROM IMSUNLD.DBLOG5OF.DB#AASEG
    ```

Related Topics:

- "Settings That Affect Access to IMS Sources" on page 124

Lookup Transformations

In PowerCenter, you can add a Lookup transformation to a mapping to look up data in an IMS database. You should use a separate netport job for each Lookup transformation for PCBs with a "GOx" PROCOPT.

Tip: When creating Lookup transformations, specify the key fields of the IMS segments so that PowerExchange can create a fully qualified Segment Search Argument (SSA) to improve the efficiency of IMS database searching.

For more information about using a lookup transformation, see *PowerExchange Interfaces for PowerCenter*.

IMS Target Considerations

- To reduce the risk of locking segments, you can reduce the commit frequency from PowerCenter.
- If an IMS target contains fields to which blanks or no data will be written, define those fields as optional fields in the data map. Otherwise, PowerExchange will issue errors during the bulk data movement.
- You cannot write data to IMS SDEP segments or to a segment that is below an unkeyed segment.
- If you want to write data to a segment that is mapped by an OCCURS clause, change the copybook to identify each field in the OCCURS clause as a separate field and then remove the OCCURS clause. PowerExchange cannot generate a separate row for each occurrence defined by an OCCURS clause.
- When you import the IMS data map into PowerCenter, verify that any CCK field that you want to use as a key field is identified as a key field.
- Informatica recommends that you use a separate netport job to write IMS data. This practice enables you to access a PSB with write intent and to modify the JCL, as needed, for updated IMS data, such as the IEFRDER log.

Settings That Affect Access to IMS Sources

When you perform certain tasks from the PowerExchange Navigator, such as creating a data map or performing a database row test, you enter several settings that affect how PowerExchange accesses an IMS source. These settings are interrelated and depend on the IMS source type. The following settings are of particular interest:

- **Access method** option that you select in the IMS data map
- Node name that you specify in **Location** field when sending the data map to the remote MVS system or performing a database row test
- **DB Type** option that you select for a database row test

The following table summarizes how these settings are related to one another and the IMS source type:

IMS Source Type	Access Method in Data Map	DB Type for Row Test	Location
IMS database	DL/1 Batch	NRDB or NRDB2	Node name for the netport port on the MVS system
IMS database	IMS ODBA	NRDB	Node name for the PowerExchange Listener port on the MVS system
IMS unload data set	DL/1 Batch	IMSUNLD	Node name for the PowerExchange Listener port on the MVS system
IMS unload data set	IMS ODBA	IMSUNLD	Node name for the PowerExchange Listener port on the MVS system

Lookup Transformations

You can optionally include a Lookup transformation in a mapping for a bulk data movement to add related values from a lookup source or to determine if some of the records already exist on the target.

In a Lookup transformation, you must indicate the key fields that you want PowerExchange to use in the Segment Search Argument (SSA) for searching the IMS database. Concatenated keys (CCK) fields achieve the best performance and have the least impact on the IMS database.

To minimize contention on the IMS database during lookup processing, also create the IMS PCBs with the PROCOPT=GO*x* parameter. This parameter indicates what IMS should do if it encounters an invalid pointer while reading a segment that is being updated.

For more information about Lookup transformations, see *PowerExchange Interfaces for PowerCenter* and *PowerCenter Transformation Guide*.

Materializing a Target from an IMS Database

Use the following general procedure to move bulk data from an IMS database to a relational or nonrelational target. This procedure assumes that you are using PWXPC to integrate PowerExchange with PowerCenter.

Before you begin, complete these tasks:

- If you plan to create a DL/1 Batch data map, create a netport job and complete the other configuration tasks required to configure PowerExchange for bulk data movement with a DL/1 batch data map.
- If you plan to create an IMS ODBA data map, complete the configuration tasks required to configure PowerExchange for bulk data movement with an IMS ODBA data map.

To materialize a target from an IMS database:

1. Create the target database structure.
2. From the PowerExchange Navigator, create a data map for the IMS source.
3. Open the data map in the Resource Explorer, and click the nonrelational “record” or relational “table” view to verify that the metadata was successfully retrieved.

   Also, you can view the IMS segment hierarchy that was imported from the DBD source by right-clicking the data map and clicking **Display IMS Hierarchy**.
4. Send the data map to the PowerExchange Listener on the remote z/OS system. Select the data map in the Resource Explorer and click **File > Send to Remote Node**.
5. Run a database row test on the table view of the IMS source or target to verify that actual data can be returned from the IMS database.
6. In the PowerCenter Designer, import the PowerExchange data map and create a mapping.

   When you import the data map, in the **Location** field, enter the PowerExchange Listener node name that you specified in the NODE statement of the local dbmover.cfg file. Also, specify **IMS** in the **Source Type** field. This field is also displayed for targets.

   Tip: You can create a Lookup transformation to get related values or to determine if source records already exist on the target. Add the Lookup transformation to the mapping.
7. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX NRDB Batch application connection. Then start the workflow to perform the bulk data movement.

If you plan to use PowerExchange CDC to update the target, you must set up restart tokens.

Related Topics:

- "Configuring PowerExchange for Bulk Data Movement with an IMS ODBA Data Map" on page 121

Bulk Data Movement and IMS Unload Files

You can create a PowerCenter bulk data movement session that uses an IMS unload data set as a data source. You can also create a PowerCenter bulk data movement session that uses an IMS unload file as the source and a second IMS unload file as the target.

Using IMS Unload Data Sets as Data Sources

You can use an IMS unload data set as a data source for a PowerExchange bulk data movement operation. To create the unload data set, use either the IBM unload utility for your IMS database type or the Unload function of the BMC Database Management for IMS solutions.

For example, to materialize a target based on an IMS source, create an IMS unload data set that contains the source data of interest. Then use that data set as input to a PowerCenter workflow that materializes the target. After target materialization is complete, use PowerExchange change data capture (CDC) to synchronize the source and the target.

For unload data sets, PowerExchange can perform group source processing. That is, PowerExchange can process multiple IMS segments by reading the entire unloaded database in a single pass.

The following table shows the unload record formats and key definition methods that you can use for different types of IMS databases:

IMS Database Type	Format of Unload Records	Get CCK?	Get Key Value by Using GetDatabaseKey Function?	Get Key Value by Using GetIMSRBAByLevel Function?
Fast Path [1]	IBM	Yes	Not available	Not available
Fast Path [1]	BMC LONG	Yes	Yes	Yes
HALDB	IBM	Yes	Yes	Yes
HALDB	BMC LONG	Yes	Yes	Yes
HDAM	IBM	Yes	Yes	Yes
HDAM	BMC LONG	Yes	Yes	Yes
HDAM	BMC SHORT	Yes	Not available	Not available
HDAM	BMC XSHORT	Yes	Not available	Not available
HIDAM	IBM	Yes	Yes	Yes
HIDAM	BMC LONG	Yes	Yes	Yes
HIDAM	BMC SHORT	Yes	Not available	Not available

IMS Database Type	Format of Unload Records	Get CCK?	Get Key Value by Using GetDatabaseKey Function?	Get Key Value by Using GetIMSRBAByLevel Function?
HIDAM	BMC XSHORT	Yes	Not available	Not available
HISAM	IBM	Yes	Not available	Not available
HISAM	BMC LONG	Yes	Not available	Not available
HISAM	BMC SHORT	Yes	Not available	Not available
HISAM	BMC XSHORT	Yes	Not available	Not available
1. PowerExchange does not support IMS Fast Path SDEP segments.				

Note: For BMC Software unload data sets, the record formats of LONG, SHORT, and XSHORT are specified by the DFSURGU1 keyword in the PLUSIN DD statement of the BMC Unload JCL.

The table shows that PowerExchange can get a concatenated key (CCK) for all unload record types. For unkeyed segments, PowerExchange can process any GetDatabaseKey and GetIMSRBAByLevel functions that you define in a user-defined field in an IMS data map to return a value that can be used as key. The GetDatabaseKey and GetIMSRBAByLevel functions return an 8-byte RBA value for a specific segment.

To add the segment RBA to segment data in an IMS data map, Informatica recommends that you use the GetIMSRBAByLevel function rather than the GetDatabaseKey function. The GetIMSRBAByLevel function enables you to get the RBA of an unkeyed or non-unique keyed parent segment.

Use the GetDatabaseKey and GetIMSRBAByLevel functions in IMS data maps used for bulk data movement operations or IMS synchronous CDC.

You cannot use the GetDatabaseKey or GetIMSRBAByLevel function in the following types of data maps:

- IMS data maps used for IMS log-based CDC
- IMS data maps used to access IMS unload files that contain the following types of records:
  - Unload records of any format for HISAM data sets
  - Unload records that have the format BMC SHORT or BMC XSHORT for HDAM and HIDAM data sets
  - Unload records that have the standard IBM format for IMS Fast Path data sets

Related Topics:

- "Materializing a Target from an IMS Unload Data Set" on page 128

DBMOVER Configuration Statement for IMS Unload Data Sets

If you use an IMS unload data set, you must specify the IMSID parameter in the DBMOVER configuration member.

For bulk data movement, this parameter defines the DBDLIB and the RECON data sets:

```
IMSID=(ims_ssid,DBD_lib,RECON=(recon,recon,...))
```

Where:

- *ims_ssid* is the IMS subsystem identifier. Specify a 1- to 8-character alphanumeric string.

  In the IMS data map, enter an IMS SSID value that matches this value to enable PowerExchange to find the DBDLIB data set.
- *DBD_lib* is the name of the IMS DBDLIB data set that contains the DBD load modules.
- *recon* is an IMS RECON data set. Specification of RECON data sets is required for IMS log-based CDC only. Use commas to separate RECON data sets.

Note: For IMS unload data sets, only the IMS subsystem identifier and IMS DBDLIB data set options are required.

Materializing a Target from an IMS Unload Data Set

Use the following general procedure to create an unload data set of an IMS source database and use it to materialize a relational or nonrelational target. This procedure assumes that you are using PWXPC.

For an IMS unload data source, you can use Designer to import a data map with multiple record types to create a PowerCenter source definition.

To materialize a target from an IMS unload data set:

1. Create an IMS unload data set that contains the source data that you want to use for materializing the target. For more information, see the IBM or BMC documentation for the unload utility that you use.
2. Create the target database structure.
3. In the PowerExchange Navigator, create a data map for the IMS source from which you unloaded data.
4. In the PowerCenter Designer, import the data map and create a mapping. In the mapping, specify the IMS segments from which you unloaded data as the source and the database structure that you created in step 2 as the target.

   If the data map contains multiple record types and you want PowerExchange to perform group source processing, select **Multi-Record Datamaps** in the **Import from PowerExchange** dialog box. Selecting this option causes PowerExchange to import the data map as a multi-record data map.
5. In the PowerCenter Workflow Manager, create a workflow and session and configure a PWX NRDB Batch application connection.

   In the application connection, specify a **Location** value that points to the PowerExchange Listener on the z/OS system that contains the IMS data. In the session properties, specify the IMS unload data set name in the **IMS Unload File Name** attribute.

   Then start the workflow to move the IMS unload data to the target database.

   If you plan to use PowerExchange CDC to update the target, you must set up restart tokens.

Multiple-Record Writes to IMS Unload Data Sets

During bulk data movement sessions, PowerExchange can use a multiple-record data map to read data from an IMS unload data set on z/OS and write the multiple records in a single pass to an IMS unload data set target. This process is called a *multiple-record write*.

When PowerExchange performs a multiple-record write, it preserves source sequencing information. To enable multiple-record writes with sequencing, select the **Multi-Record Datamaps** and **Use Sequence Fields** options in the **Import from PowerExchange** dialog box when you create the source and target definitions.

To perform multiple-record writes with sequencing, PowerExchange uses group source processing to read source data in a single pass and uses group target processing to write data to the target in a single pass.

PowerExchange generates sequence fields to pass metadata about the source record relationships to the target. After you enable multiple-record writes with sequencing for a PowerCenter workflow, the workflow can read the multiple-record source, use the generated sequence fields to preserve the sequencing information, and write data to the target in the same record sequence as in the source.

To determine the relationships among IMS record types, PowerExchange uses the segment information that you imported into the multiple-record data map from the DBD. PowerExchange uses the segment information to generate primary and foreign keys and sequencing metadata in the source and target definitions.

When you run a PowerCenter session, PowerExchange uses the generated key values to reconstruct and write the data to the target in the correct sequence. PowerExchange maintains the data in a sequencing and queuing cache on the Integration Service node. When PowerExchange writes data to the target, it deletes the generated key fields and sends the data across the network in the correct sequence to the target file.

Rules and Guidelines for Multiple-Record Writes to IMS Unload Data Sets

Review the rules and guidelines that pertain to multiple-record writes to IMS unload data set targets before you configure a workflow that uses this feature.

For more information, see the *PowerExchange Interfaces for PowerCenter*.

Source and Target Files

The following rules and guidelines apply to sources and targets:

- If the target is an IMS unload file, the source must also be an IMS unload file.
- The source IMS unload file can be in any format that PowerExchange supports as a data source. For more information, see "Using IMS Unload Data Sets as Data Sources" on page 126.
- Only standard IBM unload formats are supported for targets. BMC or partitioned unload files are not supported.
- The unload file is written in Fast Path database format if the data map specifies a Fast Path DEDB database. Otherwise, the file is written in the standard IBM unload format.
- If you run the IBM unload utility to unload data from multiple partitions, the partitions are interleaved in the output unload file. Before you use the unload file as a data source in a workflow that performs a multiple-record write, you must sort the unload file to sequence the partitions. Alternatively, you can add the following control card to the unload job:

  ```
  TASKCTL=(1)
  ```

- If you unload segments for multiple areas of a Fast Path DEDB to a single unload file, you must sort the unload file before you can use it as a source in a workflow that performs a multiple-record write.

Data Maps, Source Definitions, and Target Definitions

The following rules and guidelines apply to data maps, source definitions, and target definitions:

- You must create new source and target definitions for multiple-record writes. Existing definitions without sequencing information do not work for multiple-record writes. You can continue to use existing definitions for workflows that do not use multiple-record writes.
- The source and target definitions for IMS unload files must be based on the same data map.
- The data map must define the SSID and DBD name. These fields might be blank on existing DL/1 or ODBA data maps. If you try to import an IMS data map with a blank SSID or DBD field into a source definition for which you enable multiple-record writes, PowerExchange issues messages PWX-03611 or PWX-03612.
- Verify that all tables that form a complete hierarchy are included in the data map. The import process does not verify hierarchical completeness.

To ensure that all of the required tables in the data map are included in the source definition, verify that they are defined in the data map as simple tables. A *simple table* is based on a single record.

If the data map includes complex tables, PowerExchange does not include them in the source or target definition. A *complex table* is based on more than one record.

- You cannot select which tables in the data map to import. PowerCenter imports all of the simple tables in the data map.
- PowerExchange excludes any user-defined, RBA-related, or CCK-related fields that are in the imported IMS data map from the source definition.
- The data map must have exactly one table for each record.

Mappings and Transformations

The following rules and guidelines apply to PowerCenter mappings and transformations:

- Mapping transformations might drop rows. In this case, ensure that the mapping does not create any orphan rows. When a multiple-record write workflow runs, it drops orphan rows instead of writing them to the target.
- The mapping must not alter the position or the name of the generated sequence fields or drop them.
- Transformations that insert records into a mapping are not supported.
- Transformations that might alter row IDs, including the Sorter, Joiner, and Rank transformations, are not supported.

Connections

The following rules and guidelines apply to connections and connection properties:

- You can use multiple-record writes with PWXPC NRDB Batch connections only. PowerExchange ODBC connections are not supported.
- Select **Filter After** for the **Offload Processing** source and target connection attributes. PowerExchange and PWXPC perform offload processing on the Integration Service machine before writing the output records to z/OS.

  If you select a different value, PowerExchange changes the value to **Filter After**.

  However, if RBAs are being read using the GetDatabaseKey or GetIMSRBAByLevel function from an IMS unload file in the BMC LONG format, you must select **No** for the **Offload Processing** source connection attributes. Otherwise, the session will fail with message PWX-03803.
- Multithreaded processing is supported for IMS source connections if you set the **Offload Processing** connection attribute to **Filter After** and set the **Worker Threads** connection attribute to a non-zero value.
- Multithreaded processing is not supported for IMS target connections. If you set the **Worker Threads** connection attribute, the setting is ignored.
- Select **Off** for the **Confirm Write** source and target connection attributes. If you select a different value, PowerExchange changes the value to **Off**.
- In certain cases, you might need to change the value of the **CSQ_MEMSIZE** parameter of the **PWX Override** connection attribute. This parameter defines the maximum amount of memory that the cache can consume for multiple-record writes.

Session Properties

The following rules and guidelines apply to PowerCenter session properties:

- Pipeline partitioning is not supported for the reader or writer if you defined sources or targets with sequencing enabled.
- You must specify the **IMS Unload File Name** property for IMS unload sources and targets.

- For sources, the **Flush After N Blocks** property specifies the maximum number of block flushes that can occur across all groups without a given block being flushed. Define this property to ensure that all blocks are flushed at regular intervals.

Performing a Multiple-Record Write to an IMS Unload Data Set

To transfer data in the correct physical sequence from a multiple-record IMS unload data set source to a multiple-record IMS unload data set target, configure a session that performs multiple-record writes.

1. Create a multiple-record data map for the IMS data source, if you have not already created one.
2. In the PowerCenter Designer, click **Sources > Import from PowerExchange**.
3. In the **Import from PowerExchange** dialog box, enter the required information.

   The following table describes the required information:

Attribute	Description
Location	Name of the node on which the source file resides. This value must match the name of a NODE statement in the PowerExchange dbmover.cfg file.
User Name	User name that has the database authority to connect to the target.
Password	Password that is associated with the user name.
Multi-Record Datamaps	Whether to list multiple-record data maps for selection. Select this check box.
Use Sequence Fields	Whether to generate sequence fields for multiple-record write operations. Select this check box.
Source Type	Data source type. Select **IMS.**

   For more information about this dialog box, see *PowerExchange Interfaces for PowerCenter*.
4. Click **Connect.**

   The available multiple-record data maps appear in the **Selected Datamaps** box.
5. Select the data map that you want to import.
6. Click **OK.**

   The source definition appears in the workspace.
7. Click **Targets > Import from PowerExchange**.
8. In the **Import from PowerExchange** dialog box, enter the required information.

The following table describes the required information:

Attribute	Description
Location	Name of the node on which the source file resides. This value must match the name of a NODE statement in the PowerExchange dbmover.cfg file.
User Name	User name that has the database authority to connect to the source.
Password	Password that is associated with the user name.
Multi-Record Datamaps	Whether to list multiple-record data maps for selection. Select this check box.
Use Sequence Fields	Whether to generate sequence fields for multiple-record write operations. Select this check box.
Source Type	Data target type. Select **IMS.**

9. Click **Connect.**

   The available multiple-record data maps appear in the **Selected Datamaps** box.
10. Select the same data map that you selected for the source definition.
11. Click **OK.**

    The target definition appears in the workspace.
12. Create a mapping that includes the source definition, Source Qualifier transformation, and target definition.

    Between the Source Qualifier transformation and the target definition, you can include transformations that meet the requirements in "Rules and Guidelines for Multiple-Record Writes to IMS Unload Data Sets" on page 129.
13. Define a connection for the IMS database, if one does not already exist.
14. Define PowerCenter session properties:
    a. In the Task Developer, double-click the session to edit it.
    b. On the **Mapping** tab, click the **Sources** view.
    c. Under **Properties**, set the **IMS Unload File Name** property to the name of the IMS source unload file.
    d. Define additional source properties as needed.
    e. On the **Mapping** tab, click the **Targets** view.
    f. Under **Properties**, set the **IMS Unload File Name** property to the name of the IMS unload file that you want the session to allocate.
    g. Define additional target properties as needed.

       Note: If you want PowerExchange to create the empty target data set before the PowerCenter session runs, include the CREATEFILE command in the **Pre SQL** session property for the target. For more information, see the discussion of empty files in the appendix, "PowerExchange Interfaces for PowerCenter Tips," in the *PowerExchange Interfaces for PowerCenter* guide.
    h. Click **OK**.

CHAPTER 10

Microsoft SQL Server Bulk Data Movement

This chapter includes the following topics:

Introduction to Microsoft SQL Server Bulk Data Movement

You can optionally use PowerExchange, in conjunction with PowerCenter, to move bulk data from or to a Microsoft SQL Server database on a Windows system. PowerExchange establishes the connection to the SQL Server database for reading or writing data.

You can use PowerCenter without PowerExchange to move SQL Server bulk data. However, using PowerExchange for SQL Server bulk data movement provides the following advantages:

- PowerExchange provides a connection to SQL Server when a native connection is unavailable on the platform where the PowerCenter Integration Service is running.
- An ODBC connection is unnecessary.
- PowerCenter can take advantage of the unique features of PowerExchange connections.

You can use the SQL Server bulk load utility with PowerExchange to efficiently load large amounts of data to SQL Server tables.

Because SQL Server is a relational database, you do not need to create a data map from the PowerExchange Navigator. You can define the bulk data movement entirely from the PowerCenter Designer and Workflow Manager. In the PowerCenter Designer, you can import SQL Server source or target metadata from PowerExchange to create the source or target definitions.

To access a SQL Server source or target, run a PowerExchange Listener on a Windows system and verify that PowerExchange can communicate with it. Also, if you run the PowerExchange Listener on a Windows system that is remote from the SQL Server database, install the Microsoft SQL Server 2012 Native Client on the PowerExchange Listener system. For more information, see "Setting Up and Testing Connectivity to a SQL Service Source or Target" on page 136.

Datatypes Supported for SQL Server Bulk Data Movement

The following table identifies the SQL Server datatypes that PowerExchange supports and does not support for bulk data movement:

Datatype	Supported for Bulk Movement?	Comments
bigint	Yes	-
binary	Yes	-
bit	Yes	-
char	Yes	-
date	Yes	In PowerCenter, when you import source metadata from PowerExchange to create a source definition, PowerExchange converts date columns to varchar(10) columns. This conversion is for consistency with PowerCenter datatype handling.
datetime	Yes	-
datetime2	Yes	-
datetimeoffset	Yes	-
decimal	Yes	-
float	Yes	-
geography	No	-
geometry	No	-
hierarchyid	No	-
image	No	Use varbinary(MAX) instead.
int	Yes	-
money	Yes	-
nchar	Yes	-
ntext	No	Use nvarchar(MAX) instead.
numeric	Yes	-
nvarchar	Yes	-
real	Yes	-

Datatype	Supported for Bulk Movement?	Comments
smalldatetime	Yes	-
smallint	Yes	-
smallmoney	Yes	-
sql_variant	No	PowerExchange does not capture change data for sql_variant columns but does capture change data for other columns in the same table.
text	No	Use varchar(MAX) instead.
time	Yes	-
timestamp	Yes	-
tinyint	Yes	-
uniqueidentifier	Yes	PowerCenter imports the uniqueidentifier datatype as a varchar datatype of 38 characters.
user-defined datatypes (UDTs)	Yes	PowerExchange treats a UDT in the same way as the datatype on which the UDT is based.
varbinary	Yes	-
varchar	Yes	-
xml	Yes	PowerExchange treats this datatype as varchar(MAX).

Configuring Microsoft SQL Server Bulk Data Movement

To configure PowerExchange for bulk data movement operations, complete the following tasks:

- Verify that PowerExchange has read access to Microsoft SQL Server source tables. If the SQL Server database is on the system where PowerExchange runs, PowerExchange uses the Windows authorization for connections to SQL Server. If the SQL Server database is on a remote Windows system, PowerExchange uses the user ID under which the remote PowerExchange Listener runs.
- Configure Microsoft SQL Server bulk data movement by defining the MSS_ERRORFILE statement in the DBMOVER configuration file.
- Set up and test connectivity to SQL Server sources and targets.

Adding a Microsoft SQL Server Statement to the DBMOVER Configuration File

To specify the name of the user-customized SQL error file that PowerExchange uses for Microsoft SQL Server bulk data movement operations, include the MSS_ERRORFILE statement in the DBMOVER configuration file.

For more information about the user-customized SQL error file, see "Creating Error Action Files for Customized Error Handling" on page 174.

MSS_ERRORFILE Statement

The MSS_ERRORFILE statement specifies the name of the user-customized SQL error file that PowerExchange uses for Microsoft SQL Server bulk data movement operations.

Operating Systems: Windows

Data Sources: Microsoft SQL Server

Required: No

Syntax:

```
MSS_ERRORFILE=file_name
```

Value: For the *file_name* variable, enter the complete path and file name that contains the SQL error codes that you want PowerExchange to treat as either recoverable or fatal. PowerExchange provides a sample error action file called mssqlerr.act in the PowerExchange installation directory.

Setting Up and Testing Connectivity to a SQL Service Source or Target

To access a SQL Server source or target, run a PowerExchange Listener on a Windows system and verify that PowerExchange can communicate with it. Also, if you run the PowerExchange Listener on a Windows system that is remote from the SQL Server database, install the Microsoft SQL Server 2012 Native Client on the PowerExchange Listener system.

1. If you run the PowerExchange Listener on a Windows system that is remote from the SQL Server source system with the distribution database, install the Microsoft SQL Server 2012 Native Client on the PowerExchange Listener system.
2. On the system or systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, perform the following tasks:
   - Use the **ping** command to test network access to the PowerExchange Listener system, if it is remote.
   - Add the NODE statement to the dbmover.cfg file to identify the PowerExchange Listener and its port:

     ```
     NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
     ```

     Tip: To determine the port number, consult with the person in your organization who installed the PowerExchange Listener on the remote system.
   - Run the DTLREXE ping utility to test connectivity to the PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Moving Microsoft SQL Server Bulk Data

Use the following procedure to move bulk data from or to an Microsoft SQL Server database on Windows. This procedure assumes that you are using PWXPC to integrate PowerExchange with PowerCenter.

Before you begin, get the following information:

- Name of the SQL Server server
- Name of the SQL Server database
- Names of the SQL Server source or target tables
- Database user ID and password

  Tip: From the PowerExchange Navigator, you can preview source table metadata and data before performing the bulk data movement operation. To view the data in each table column, perform a database row test on the personal metadata profile for the table.

To move Microsoft SQL Server bulk data:

1. In the PowerCenter Designer, click **Source** > **Import from PowerExchange** for a SQL Server data source or click **Target** > **Import from PowerExchange** for a SQL Server data target.
2. In the Import from PowerExchange dialog box, enter the following required attributes:
   - In the **Location** field, enter the PowerExchange Listener node name that you specified in the LISTENER statement of the local dbmover.cfg file if the SQL Server database is on the local system where PowerCenter runs. If the SQL Server database is on another system, enter the node name that is specified in the NODE statement of the local dbmover.cfg.
   - In the **User name** and **Password** fields, enter the user ID and password for accessing the SQL Server source or target tables.
   - In the **Source Type** list, select **MSSQL**.
   - In the **Server Name** field, enter the SQL Server instance name.
   - In the **Database Name** field, enter the SQL Server database name.

     Complete the optional attributes as needed.
3. In the PowerCenter Designer, create a mapping.
4. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX MSSQLServer relational database connection. Then start the workflow to perform the bulk data movement.

Using the SQL Server Bulk Load Utility to Load Bulk Data

You can use the SQL Server bulk load utility to efficiently load large amounts of data to SQL Server tables that PowerExchange has read from data sources. PowerCenter uses PowerExchange Client for PowerCenter

(PWXPC) connections to communicate with PowerExchange to implement the bulk data movement. PowerExchange invokes the SQL Server bcp utility to perform the bulk load.

Summary of Configuration Steps

To configure PowerExchange to run Microsoft SQL Server bulk load operations, complete the standard procedures for moving SQL Server bulk data:

- "Configuring Microsoft SQL Server Bulk Data Movement" on page 135
- "Moving Microsoft SQL Server Bulk Data" on page 137

In addition, perform these specific tasks:

- Adjust the value of APPBUFSIZE in the DBMOVER configuration file as needed.
- Set the PWXPC connection attributes for SQL Server bulk loads.
- Set the Target Load Type session property.
- Review the PowerCenter session log to confirm that the SQL Server bulk loader is being used.

PWX MSSQLServer Connection Attributes for Microsoft SQL Server Bulk Loads

The following table describes each connection attribute that you can set to control bulk data load operations that use the Microsoft SQL Server bcp utility:

Connection Attribute	Description
Array Size	Specifies the size of the storage array, in number of records, for SQL Server bulk loads. Valid values are from 1 through 5000. Default is 25.
Reject File	Overrides the default prefix of PWXR for the reject file. PowerExchange creates the reject file on the target machine when the Write Mode is Asynchronous with Fault Tolerance. **Note:** Specifying PWXDISABLE will prevent creation of the reject files.
Write Mode	Select the write mode. Default is Confirm Write On.

For more information about PWX MSSQLServer relational connections in PowerCenter, see *PowerExchange Interfaces for PowerCenter*.

Target Load Type Session Property

In the Properties settings on the Mapping tab of the session properties, you can choose **Normal** or **Bulk** for the **Target Load Type** session property. To use the SQL Server bcp utility to load bulk data, select **Bulk**.

Note: Choose **Normal** if the mapping contains an Update Strategy transformation.

PowerCenter Session Log for SQL Server Bulk Loader Sessions

To confirm that the SQL Server bulk loader is being used, review the PowerCenter session log. The following messages show that the bulk loader is being used:

```
2012-03-27 09:37:44 : INFO : (13240 | WRITER_1_*_1) : (IS | i_opal) : n_opal :
WRT_8146 : Writer: Target is database [@Async], user [], bulk mode [ON]
2012-03-27 09:37:44 : INFO : (13240 | WRITER_1_*_1) : (IS | i_opal) : n_opal :
WRT_8106 : Warning! Bulk Mode session - recovery is not guaranteed.
2012-03-27 09:37:44 : INFO : (13240 | WRITER_1_*_1) : (IS | i_opal) : n_opal :
CMN_1021 : Database driver event...
Connection Information for the PowerExchange Listener: Location[beefy64], Database
Type[MSSQL], User Name[stqa], Compression[N], Encryption[N], Pacesize[0], Interpret As
Rows[Y], Confirm Network Write[Y]
Database Type Specific parameters:  Server Name[MHV2K8QA6L03], Database Name[Async],
Array Size[50], Bulk load[Yes]
```

Tuning Considerations for Bulk Loads

For optimum performance, you might need to increase the following values:

- Array Size connection attribute. For best performance, specify a value of 500 or greater.
- APPBUFSIZE statement in the DBMOVER configugration file.

CHAPTER 11

Oracle Bulk Data Movement

This chapter includes the following topics:

Introduction to Oracle Bulk Data Movement

You can optionally use PowerExchange, in conjunction with PowerCenter, to move bulk data from or to an Oracle database on Linux, UNIX, or Windows.

Because Oracle is a relational database, you do not need to create a data map from the PowerExchange Navigator. You can define the bulk data movement entirely from the PowerCenter Designer and Workflow Manager. In the PowerCenter Designer, you can import Oracle source or target metadata from PowerExchange to create the source or target definitions.

If the Oracle database is on a system that is remote from the system or systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, you can use either Oracle SQL*Net, also called Net8, or a PowerExchange Listener to communicate with the remote Oracle database.

Note: It is possible to use PowerCenter only to move Oracle bulk data.

PowerExchange bulk data movement does not support Oracle 12*c* Multitenant environments.

Datatypes Supported for Oracle Bulk Data Movement

PowerExchange supports most Oracle datatypes for bulk data movement.

The following table identifies the Oracle datatypes that PowerExchange supports and does not support for bulk data movement:

Datatype	Supported for Bulk Data Movement?	Comments
BFILE	No	-
BINARY_DOUBLE	Yes	Supported if you have Oracle Client 10*g* or later
BINARY_FLOAT	Yes	Supported if you have Oracle Client 10*g* or later
CHAR	Yes	-
DATE	Yes	-
FLOAT	Yes	-
LOB	No	-
LONG	No	-
LONG RAW	No	-
NCHAR	Yes	-
NUMBER	Yes	PowerExchange handles NUMBER columns as follows: - Numbers with a scale of 0 and a precision value less than 10 are treated as INTEGER. - Numbers with a defined precision and scale are treated as NUMCHAR. - Numbers with an undefined precision and scale are treated as DOUBLE.
NVARCHAR2	Yes	-
RAW	Yes	-
ROWID	Yes	-
TIMESTAMP	Yes	-
TIMESTAMP WITH TIME ZONE	No	-
TIMESTAMP WITH LOCAL TIME ZONE	No	-
UROWID	No	-
VARCHAR2	Yes	-
XML types	No	-

Configuring Oracle Bulk Data Movement

To configure PowerExchange for bulk data movement operations, the PowerExchange administrator completes the following tasks:

- If the Oracle data source or target is on a remote system and you want to use a PowerExchange Listener to communicate with it, set up and test connectivity to the PowerExchange Listener on the remote system. Alternatively, you can use SQL*Net.
- Configure Oracle bulk data movement by defining Oracle-specific statements in the dbmover.cfg configuration file. If you created an error action file to customize PowerExchange fault tolerance behavior, include the ORA_ERRORFILE statement.

Setting Up and Testing Connectivity to a Remote Oracle Source or Target

If the Oracle data source or target is on a remote system, PowerExchange and PowerCenter must be able to communicate with the PowerExchange Listener on that remote system unless you are using SQL*Net.

To set up and test connectivity to a remote Oracle source or target:

1. On the system or systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the **ping** command to test network access to the remote system.
2. Add the NODE statement to the dbmover.cfg file on each PowerExchange and PowerCenter system to identify the remote PowerExchange Listener and its port:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   Tip: To determine the port number, consult with the person in your organization who installed the PowerExchange Listener on the remote system.
3. Run the DTLREXE ping utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Adding Oracle-Specific Statements to the DBMOVER Configuration File

To configure Oracle bulk data movement, you can include the following Oracle-specific statements in the dbmover.cfg configuration file on Linux, UNIX, or Windows:

- ORA_ERRORFILE
- ORACLECODEPAGE

After editing the DBMOVER configuration member, you must restart the PowerExchange Listener for your changes to take effect.

For descriptions of all DBMOVER statements, see the *PowerExchange Reference Manual*.

ORA_ERRORFILE Statement

The ORA_ERRORFILE statement specifies the name of the user-customized SQL error file that PowerExchange uses for Oracle bulk data movement operations.

Operating Systems: Linux, UNIX, and Windows

Data Sources: Oracle

Required: No

Syntax:

```
ORA_ERRORFILE=file_name
```

Value: For the *file_name* variable, enter the path and file name that contains the SQL error codes that you want PowerExchange to treat as either recoverable or fatal. PowerExchange provides a sample error action file called ora8err.act in the PowerExchange installation directory.

ORACLECODEPAGE Statement

If the NLS_LANG environment variable specifies a character set other than UTF8 or AL32UTF8, you must define the ORACLECODEPAGE statement. This statement determines the code pages that PowerExchange and PowerCenter use for a specific Oracle database during bulk data movement operations.

The code page or pages that you specify in this statement must match the character set that is identified in the NLS_LANG environment variable. The NLS_LANG environment variable determines how the Oracle client libraries deliver data to PowerExchange.

Operating Systems: Linux, UNIX, and Windows

Data Sources: Oracle

Related Statements: CODEPAGE

Required: If NLS_LANG specifies a character set other than UTF8 or AL32UTF8

Syntax:

```
ORACLECODEPAGE=(tnsname_host
               ,pwx_codepage
               ,pc_codepage
)
```

Parameters:

tnsname_host

Required. An entry in the Oracle tnsnames.ora configuration file for an Oracle database. The entry defines the database address that PowerExchange uses to connect to the database.

pwx_codepage

Required. A code page number or alias name that PowerExchange uses to identify a code page. To determine valid PowerExchange code page and alias values, use the ICUCHECK utility to generate report 5, “PowerExchange Code Page Names and Aliases.”

Note: PowerExchange supports some common Oracle character set names as aliases to code pages.

pc_codepage

Optional. A name that controls the processing of the SQL statements that PowerExchange passes to Oracle on behalf of PowerCenter bulk data movement sessions. PowerExchange supplies a default that is usually suitable.

Do not specify this parameter except in special circumstances when the default does not work. For example, if you specify a user-defined ICU code page for the *pwx_codepage* parameter, you need to specify this parameter.

Usage Notes:

- You can specify up to 20 ORACLECODEPAGE statements, each for a separate database, in a dbmover.cfg configuration file.
- If PowerExchange uses the same NLS_LANG environment variable to access multiple Oracle databases, you do not need to specify a separate ORACLECODEPAGE statement for each database. Instead, specify a single ORACLECODEPAGE statement and leave the *tnsname_host* parameter blank. The specified code page then applies to all databases that have an entry in the tnsnames.ora file. The following example shows a statement without a *tnsname_host* parameter:

  ```
  ORACLECODEPAGE=(,MS1252)
  ```

- If you enter an incorrect PowerCenter code page value, the ODLNumResultCols routine usually reports Oracle return code 911.

Example: If the NLS_LANG environment variable specifies Korean_Korea.KO16MSWIN949, define the following ORACLECODEPAGE statement:

```
ORACLECODEPAGE=(KO102DTL,MS949)
```

Moving Oracle Bulk Data

Use the following procedure to move bulk data from or to an Oracle database on Linux, UNIX, or Windows. This procedure assumes that you are using PWXPC to integrate PowerExchange with PowerCenter.

Before you begin, get the following information:

- Oracle SID
- Database Service Name if you plan to use SQL*Net for communication with Oracle
- Table names for the Oracle source or target tables
- Database user ID and password

  Tip: From the PowerExchange Navigator, you can preview Oracle source data before moving it. Create a personal metadata profile and perform a database row test on it. The PowerExchange Navigator displays metadata, such as datatype, date format, and CCSID, for each Oracle column.

To move Oracle bulk data:

1. In the PowerCenter Designer, click **Source** > **Import from PowerExchange** for an Oracle data source or click **Target** > **Import from PowerExchange** for an Oracle data target.
2. In the Import from PowerExchange dialog box, enter the following required attributes:
   - In the **Location** field, enter the PowerExchange Listener node name that you specified in the LISTENER statement of the local dbmover.cfg file if the Oracle database is on the local system where the PowerCenter Integration Service runs. If the Oracle database is on another system, enter the node name that is specified in the NODE statement of the local dbmover.cfg.
   - In the **User name** and **Password** fields, enter the user ID and password for accessing the Oracle source or target tables.
   - In the **Source Type** list, select **ORACLE**.
   - In the **TNS Name** field, enter the SID for the Oracle instance.

     Complete the optional attributes as needed.
3. In the PowerCenter Designer, create a mapping that includes the Oracle source or target.

4. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX Oracle relational database connection. Then start the workflow to perform the bulk data movement.

CHAPTER 12

Sequential File Bulk Data Movement

This chapter includes the following topics:

Introduction to Bulk Data Movement for Sequential Files

PowerExchange, in conjunction with PowerCenter, can move bulk data from or to sequential data sets on z/OS or files on i5/OS, Linux, UNIX, or Windows. These types of sources and targets use the SEQ access method.

Hereafter, the term *sequential file* is used to refer to sequential data sets on z/OS and files on i5/OS, Linux, UNIX, or Windows systems.

You must create a data map that uses the SEQ access method in the PowerExchange Navigator for a sequential file source or target. PowerExchange uses the data map to access data and metadata and to create a relational type view of the records. PowerExchange requires a relational view to use SQL-type statements to read bulk data.

Also, run a PowerExchange Listener on the system where the source is located. Verify that this PowerExchange Listener can communicate with the system or systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run.

When reading a sequential file, the system determines the record length according to what has been specified in the NRDB data map properties and the operating system file APIs, then maps it into columns ready for processing by a relational system.

When writing a sequential file, the system appends the column data into a record using the data map field sizes and then writes it according to what has been specified in the NRDB data map properties.

Setting Up and Testing Connectivity to a Remote Source or Target

To access a data source or target on a remote system, PowerExchange must be able to communicate with the PowerExchange Listener on the remote system.

1. On the systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the ping command to test network access to the remote source or target system.
2. Add the NODE statement to the dbmover.cfg file on each PowerExchange and PowerCenter system to identify the remote node where the source or target is located. Use the following syntax:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   Tip: To determine the port number, consult with the person who installed the PowerExchange Listener on the remote system.
3. Run the DTLREXE ping utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Moving Sequential File Bulk Data

Use the following procedure to perform a bulk data movement operation for a sequential file data source. This procedure assumes that you use PWXPC to integrate PowerExchange with PowerCenter.

1. In the PowerExchange Navigator, create a data map for the source sequential file.
2. Import a local or remote COBOL or PL/I copybook, if available, to define the layout of all or selected sequential records.
3. Open the data map in the Resource Explorer, and click the nonrelational record view or the relational table view to verify that the metadata was successfully retrieved.
4. To send the data map to the PowerExchange Listener on the remote system, select the data map in the Resource Explorer and click **File > Send to Remote Node**.

   In the **Location** field, enter the node name of the remote node that is specified in a NODE statement in the dbmover.cfg file on the local PowerExchange system. Complete the other fields, as needed.

   Note: If you do not complete this step, you are prompted to send the data map to the remote node when you run a database row test.
5. Run a database row test on the table view of the metadata to verify that actual data can be returned from the sequential file source.

   Tip: If you need to change the data map to get the data you want, re-run the database row test on the revised data map.
6. In the PowerCenter Designer, import the PowerExchange data map and create a mapping.

   When you import the data map, in the **Location** field, enter the node name of the remote node that is specified in a NODE statement in the dbmover.cfg file on the local PowerExchange system. Also, in the **Source Type** field, select **SEQ**.

7. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX NRDB Batch application connection. Then start the workflow to perform bulk data movement.

   Note: If you are writing to a sequential target file using a PowerExchange connection and you want PowerExchange to create the empty target file before the PowerCenter session runs, include the CREATEFILE command in the **Pre SQL** session property for the target. For more information, see the discussion of empty files in the appendix, "PowerExchange Interfaces for PowerCenter Tips," in the *PowerExchange Interfaces for PowerCenter* guide.

Data Map Properties of Sequential Files

To move bulk data to or from a sequential file, you must create a data map for the file in the PowerExchange Navigator.

When you define a data map and select **SEQ** for the **Access Method**, you define a number of attributes for the sequential file. For descriptions of all of the attributes for the SEQ access method, see "Appendix D: Data Map Properties" in the *PowerExchange Navigator User Guide*.

Sequential File Concepts

Review the discussion of sequential file concepts to better understand how PowerExchange processes sequential files.

Record Format

On the **SEQ Access Method** tab of the **Data Map Properties** dialog box, you select one of the following record formats:

- Fixed
- Variable
- Default

Select **Fixed** or **Variable** only when Linux, UNIX or Windows files contain binary data and the record boundaries are not determined by record delimiter characters. Record delimiter characters are line feed (LF) on Linux and UNIX and carriage return / line feed (CR/LF) on Windows.

If all records have the same length, select **Fixed** for the record format and specify the size of each record.

If the records have variable lengths that are determined by integer record prefixes within the file, select **Variable** for the record format and specify the type of record prefix. For example, you can use a type of VBP2 with z/OS records that are downloaded by using FTP with the BINARY and the QUOTE SITE RDW options.

In all other situations, Informatica recommends that you select the **Default** record format so that the file APIs for the operating system determine the redord length. Specify the **Default** record format for the following kinds of files:

- i5/OS files
- z/OS files
- Linux, UNIX and Windows files where the record boundaries are found through the delimiter characters LF or CR/LF
- Linux and UNIX named pipes

Variable-Length Stream Data

In rare situations where the record boundaries of variable-length records cannot be established from the operating system APIs, you can select **Variable** for the record format and **VS** (variable-length stream) as the variable-length data file type.

Variable-length stream data processing works as follows:

1. PowerExchange reads a block of data that is large enough to hold the largest record.
2. PowerExchange tries to map the data onto each record type in the sequence that the record types are defined in the data map.
3. When a match is found, that record type is used and the pointer within the buffer is moved the required number of bytes.

Informatica recommend that you use variable-length stream data processing only as a last resort because of the following restrictions:

- Variable-length stream data processing does not scale well if many record types are used.
- You must define all record types before attempting to read the file. As a result, defining the data map is time-consuming.
- User error is possible if the records are not sequenced correctly. Record ID conditions are based on the sequence of records in the data map, and PowerExchange does not validate sequencing to prevent user errors.
- Considerable CPU time is consumed, because most record types get fully mapped before PowerExchange determines whether to select this record.
- Although writing is allowed using a variable-length stream data map, such maps are intended solely for reading. When writing, PowerExchange uses the default record format.
- In a mapping that contains user sequence fields, you might be able to write successfully using a variable-length stream data map on z/OS and i5/OS operating systems, which do not add record terminators. But writing might result in unwanted record terminator characters on Linux, UNIX, and Windows.

Determining Record Boundaries on Linux, UNIX, and Windows

On Linux and UNIX, text data records are delimited by the single record terminator character LF (X'0A'). On Windows, text data records are delimited by the CR/LF (X'0D0A') two-character sequence. Typically, text files do not contain integers that might conflict with the terminator X'0A' characters. Instead, they usually contain data in display format that you can view by using a text editor such as notepad or vi.

For text files, select the **Default** record format. Also select this format when field boundaries are identified by using delimiters (for example, commas in CSV files). When text files are transferred by using FTP, typically the ASCII option is used, which is the default.

If a file contains binary data such as integer, packed decimal, or float, then select **Fixed** or **Variable** as the record format so that the system can determine record boundaries without the boundaries being affected by the data within each record. When the file is transferred by using FTP, the BINARY option must be used.

Determining Record Boundaries on i5/OS and z/OS

On i5/OS and z/OS, the operating system file APIs return the record lengths. The same file APIs are used for all files. Consequently, on these machine types it is not necessary to distinguish between text and binary data.

Maximum Record Size

The maximum record size that PowerExchange can process is 147,456 bytes (144 KB). However, the operating system on certain machines enforces a lower limit.

i5/OS Files

i5/OS native system files have a fixed record length. Although you can select a record format of **Fixed** and specify the record size, it is simpler to select a record format of **Default**.

i5/OS has limited support for variable file types. Only files in the i5/OS integrated file system (IFS) can use UNIX-like variable-length records.

i5/OS files are typically partitioned. File names are expressed in terms of *library*/*filename*(*member*). You can omit the member component when the file has only a single member. Otherwise, the member component must either specify the exact member name or be set to an operating system keyword such as *ALL, *FIRST, or *LAST.

The maximum record size that i5/OS can process is 32,768 bytes (32 KB).

Linux and UNIX Files

PowerExchange can process text and binary data on Linux and UNIX. PowerExchange can also read text data from a named pipe.

Linux and UNIX Text Data

The record boundaries on text data are delimited by the record terminator character LF (X'0A'). Select a record format of **Default** for text data files. The file must be mapped only with display fields (such as CHAR, VARCHAR, NUMCHAR, and UZONED) that do not conflict with the record terminator value.

Unlike many Linux and UNIX programs, PowerExchange does not truncate text data records if the data contains embedded binary zero characters. PowerExchange accomplishes this by:

- Reading the next record using the fgets() C library function, which returns a record delimited by a record terminator.
- Determining the record start and end positions using the ftell() C library function.

Linux and UNIX Binary Data

If the file contains binary data such as integer, packed decimal, or float fields, select the **Fixed** or **Variable** record format so that the system can determine record boundaries without the boundaries being affected by the data within each record.

If every record in the file has the same fixed length, select **Fixed** and specify the size of each record.

If a file contains variable-length records where a record prefix contains an integer that contains the length of the remaining data in the record, select the **Variable** record format. Linux, UNIX and Windows applications seldom write binary variable-length records, but sometimes it is useful to download z/OS files to Linux, UNIX, or Windows and process them there. A commonly used variable format is VBP2, which you can use to map a file downloaded by FTP with the BINARY and QUOTE SITE RDW options. For more information, see "Downloading z/OS Files for Reading on Linux, UNIX, or Windows" on page 152.

It is best to view the file in hexadecimal mode to determine if the file has record length prefixes and to choose the appropriate variable format type. This procedure might require a certain amount of experimentation with database row tests.

Named Pipes on Linux and UNIX

On Linux and UNIX, PowerExchange can read text files from named pipes in the same manner as normal disk files. However, if the final record contains a binary zero, the record is truncated.

When processing named pipes, PowerExchange cannot use the ftell() API. Instead, PowerExchange uses one of the following values as the record length:

- For the last record, the number of bytes preceding the binary zero terminator
- For records other than the last record, the number of bytes preceding the LF terminator

If you must read named pipes on Linux or UNIX, you can work around the restrictions by using one of the following methods:

- Have the program writing the data write an extra trailer record prior to closing the pipe.
- Write a user access method program to read the pipe. For more information, see the appendix, "User Access Method Programs," in the *PowerExchange Navigator User Guide*.

The maximum record size that PowerExchange can process is 147,456 bytes (144 KB). However, certain variable types that use NUM16 integers enforce a 32-KB limit, such as VB12, VBX2, and VBP2.

Windows Files

The CR/LF (X'0D0A') characters delimit records in Windows text files. However, it is also possible on Windows to read files created on Linux or UNIX systems that contain the single LF (X'0A') delimiter.

PowerExchange cannot read named pipes on Windows as if they were files by using an SEQ data map. To read named pipes on Windows, you must write a user access method program. For more information, see the appendix, "User Access Method Programs," in the *PowerExchange Navigator User Guide*.

In other respects, Windows file processing is similar to Linux and UNIX file processing and shares the same distinction between text and binary data files.

The maximum record size that PowerExchange can process is 147,456 bytes (144 KB). However, certain variable types that use NUM16 integers enforce a 32-KB limit, such as VB12, VBX2, and VBP2.

z/OS Files

PowerExchange supports fixed-length and variable-length files on z/OS.

z/OS Record Formats

PowerExchange supports most record formats on z/OS.

PowerExchange can read or write the following types of z/OS files:

- Fixed length files with the following record formats: F (Fixed-length, unblocked), FB (Fixed-length, blocked), and FS (Fixed-length, unblocked, standard)
- Variable length files with the following record formats: V (Variable-length, unblocked), VB (Variable-length, blocked), and VS (Variable-length, unblocked, spanned)

PowerExchange does not support the following record formats:

- VBS (Variable-length, blocked, spanned)
- U (Undefined)

Informatica recommend that you select the **Default** record format for data maps that read or write z/OS files whenever possible.

The maximum record size that PowerExchange can process is 147,456 bytes (144 KB), which z/OS supports for RECFM values of FS, FBS, and VS.

Other z/OS File Attributes

PowerExchange can read or write members of partitioned files.

PowerExchange can read files with UNIX-like names on the USS file system but cannot write to them.

Dynamic Allocation of z/OS Files

Typically, you specify the file name when you define the data map for a sequential file. In this way, PowerExchange can process file names without the need to pre-define them in the JCL for the PowerExchange Listener or netport job. PowerExchange achieves this goal by dynamically allocating DD names to files using the z/OS SVC99() API.

Instead of using a specified file name, PowerExchange can also read a file using a name such as `DD:`*`JCLNAME`*, where *JCLNAME* defines the DDNAME of a DD statement in the JCL for thePowerExchange Listener or netport job.

Creation of New Files on z/OS

You can create a new z/OS data set in a PowerCenter workflow. Define the z/OS file attributes as session properties for the SEQ target.

Select **NEW** for the **Disp** session attribute. Also, at a minimum, you must define the **RECFM**, **LRECL**, **UNIT**, and **Space** allocation attributes. Optionally, you can define class attributes, including **STORCLAS** and **MGMTCLAS**.

You can define default allocation parameters as statements in the DBMOVER configuration file on the z/OS Listener machine. For example, you can define RECFM, LRECL, UNIT, and SPACE in this manner. However, Informatica recommend that you explicitly define all attributes in the workflow and not use PowerExchange Listener defaults.

Overwriting Existing Files on z/OS

In a PowerCenter workflow with a sequential file target, PowerExchange can overwrite an existing file on disk or tape. PowerExchange reuses the existing RECFM, LRECL and BLKSIZE attributes, regardless of the values that you specify in the session attributes for the connection.

Technically, PowerExchange calls the fopen() API, which uses the existing file attributes.

A PowerCenter workflow can also overwrite an existing tape file. However, in certain cases the process might be affected by constraints from the virtual tape system. PowerExchange calls the fldata() API to retrieve the attributes of the existing tape file. PowerExchange uses these attributes when it reopens the file.

z/OS Tape Files

PowerExchange can read or write tape files using data maps of either type SEQ or TAPE. However, a data map of type TAPE is required if the VOLSER needs to be specified in the data map.

PowerExchange can read and write to tape files with large block sizes.

Downloading z/OS Files for Reading on Linux, UNIX, or Windows

To move bulk data from a sequential file on z/OS to a relational table on Linux, UNIX, and Windows, Informatica recommends that you create a workflow that reads the sequential file directly from z/OS and

offloads the CPU-intensive mapping of fields to relational columns by specifying the **Filter After** connection attribute.

This processing style avoids the disadvantages of downloading files to Linux, UNIX, or Windows. These disadvantages include scheduling the downloads; the cost of the disk space on Linux, UNIX, or Windows; security issues during the network transfer; and FTP and mapping issues.

However, in certain situations it might be useful to download z/OS files to Linux, UNIX or Windows machines. You might want to download z/OS files in the following scenarios:

- You already have file transfers as part of the scheduled production jobs.
- The source z/OS system is not accessible from the Integration Service machine. For example, the data comes from an external site.
- In certain mapping and testing situations, you might need to download the z/OS file to Linux, UNIX, or Windows.

The following download types are available:

- FTP in ASCII mode
- FTP in BINARY mode for fixed-length records
- FTP in BINARY mode for variable-length records

FTP in ASCII Mode

You can use this download type if the following conditions apply:

- The z/OS file contains display fields only. Display fields include CHAR, VARCHAR, NUMCHAR, and UZONED fields.
- Character fields are converted correctly from EBCDIC to ASCII. Typically, character fields are converted correctly when character data is all single byte and the default FTP code pages are suitable.

In most cases, you can specify the **Default** record format to open a file that was downloaded in ASCII mode. The system determines the record lengths using the record delimiter character or characters.

FTP in BINARY Mode for Fixed-Length Records

When you perform the download, include the following FTP commands and keywords:

- FTP
- BINARY keyword to switch from ASCII to binary mode
- GET or PUT keyword to transfer the file

When you define the data map, set the following properties:

- Set the record format to **Fixed** and specify the size of each record.
- Set **Encoding** to EBCDIC so that numeric fields are processed correctly. This is particularly required on low-ended Linux or Windows machines.
- Set **Code Page** to the required z/OS code page, such as IBM-037.

FTP in BINARY Mode for Variable-Length Records

When you perform the download, include the following FTP commands and keywords:

- FTP.
- BINARY keyword to switch from ASCII to binary mode.
- QUOTE SITE RDW so that the z/OS FTP system sends the record length prefix prior to the data in each record. This method works only with variable length files such as RECFM=VB.
- GET or PUT keyword to transfer the file.

When you define the data map, set the following properties:

- In the data map properties, set the record format to **Variable** and specify the variable type as **VBP2**.
- Set **Encoding** to EBCDIC so that numeric fields are processed correctly. In particular, this setting is required on little-endian Linux or Windows machines, and it affects the record length prefix numbers.
- Set **Code Page** to the required z/OS code page, such as IBM-037.

User Access Method Examples

You can develop user access method libraries to handle unusual file situations. Code for a generic sample user access method is shipped with the product. You must compile the program to build the user access method library. Typically, you write user access methods in the C language, as it works on all operating systems. The data map with access method type USER enables the specification of filename, integer, and string parameters to control processing.

Example: File Not Allowed to Persist Unencrypted on Disk

You receive an encrypted file from an external source. The security policy prevents the file from being persisted on disk in decrypted, readable format.

A solution that calls a user access method program includes the following elements:

- A data map with access method type USER specifies the file name, a decryption command such as "gpg -d %FILENAME%", and the file type of text or binary.
- The pipe is created using API popen(), which executes the gpd command to decrypt the file and write the output to stdout.
- The decrypted records in the pipe are read by using the APIs fgets() for a text file or fread() for a binary file.
- The pipe is closed at the end of the run.

Example: Reading z/OS Files Downloaded by IBM Sterling Connect:Direct

Although a z/OS file downloaded by IBM Sterling Connect:Direct contains both a leading record length and trailing record terminator bytes, PowerExchange can have difficulties reading such a file for the following reasons:

- The default record format truncates reads if X'0A' occurs within binary data, such as in an integer or multibyte character field.
- The leading record length is too small because it does not count the record terminator bytes. The first record can be read with variable type VBP2, but the record boundary is not correct for the second and subsequent records.

A solution includes the following elements:

- A data map with access method type USER specifies parameters for the file name, size of the length integer, and number of bytes to strip from the end of each record.
- The user access method library reads the file, gets the length from the first two bytes, removes the specified number of bytes from the end of the record, and writes the record with a length that matches the actual size of the record according to the variable type VBP2.

- The data map can read the data in a single pass if only one record is being used.

  Multiple-record mappings require two passes. The first pass uses the data map with access method type USER to correct the record lengths, and the second pass uses a data map with access method type SEQ to read the file using variable type VBP2.

Group Source Processing of Multiple-Record Sequential Files

PowerExchange uses group source processing to read bulk data from sequential files with multiple record types.

PowerExchange group source processing reads data that is stored in the same physical source in a single pass. By eliminating multiple passes of the source data, group source processing enhances bulk data movement performance and reduces resource consumption on the source system.

To use group source processing, complete the following tasks:

1. Create a multiple-record data map for the sequential file data source.

   Use the SEQ access method.
2. Import the data map into PowerCenter.

   For PowerExchange to use group source processing, you must select the **Multi-Record Datamaps** option in the **Import from PowerExchange** dialog box.

Multiple-Record Writes to Sequential File or VSAM Targets

During bulk data movement sessions, PowerExchange can use a multiple-record data map to read data from a sequential file source and write the multiple record types to nonrelational targets. This process is called a *multiple-record write*.

When PowerExchange performs a multiple-record write, it preserves source sequencing information. To enable multiple-record writes with sequencing, select the **Multi-Record Datamaps** and **Use Sequence Fields** options in the **Import from PowerExchange** dialog box for both the source and target definitions.

You can perform a multiple-record write to the following target types:

- VSAM ESDS, KSDS, or RRDS data sets on z/OS
- Sequential data sets on z/OS
- Files on i5/OS
- Files on Linux, UNIX, and Windows

To perform multiple-record writes with sequencing, PowerExchange uses group source processing to read source data in a single pass and uses group target processing to write data to the target in a single pass.

PowerExchange generates sequence fields to pass metadata about the source record relationships to the target. After you enable multiple-record writes with sequencing for a PowerCenter workflow, the workflow

can read the multiple-record source, use the generated sequence fields to preserve the sequencing information, and write data to the target in the same record sequence as in the source.

To determine the relationships among records for sequential file and VSAM sources and generate sequence fields, PowerExchange uses the record properties that you define in the data map. These record properties include the parent record name, the name of the base record that the current record redefines, and whether the record is a header or trailer record. PowerExchange uses these record properties to generate primary and foreign keys and sequencing metadata in the source and target definitions.

When you run a PowerCenter session, PowerExchange uses the generated key values to reconstruct and write the data to the target in the correct sequence. PowerExchange maintains the data in a sequencing and queuing cache on the Integration Service node. When PowerExchange writes data to the target, it deletes the generated key fields and sends the data across the network in the correct sequence to the target file.

Rules and Guidelines for Multiple-Record Writes to Sequential File or VSAM Targets

Review the rules and guidelines that pertain to multiple-record writes to sequential file or VSAM targets before you configure a workflow that uses this feature.

For more information, see the *PowerExchange Interfaces for PowerCenter*.

Source and Target Files

The following rules and guidelines apply to sources and targets:

- PowerExchange supports multiple-record writes to VSAM ESDS, KSDS, and RRDS targets. PowerExchange does not support multiple-record writes to VSAM VRRDS targets.
- PowerExchange does not support changing the values inside header or trailer records.
- A parent must precede its children in the source data.

Data Maps, Source Definitions, and Target Definitions

The following rules and guidelines apply to data maps, source definitions, and target definitions:

- In PowerCenter, you must create new source and target definitions for multiple-record writes. Existing definitions without sequencing information do not work for multiple-record writes. You can continue to use existing definitions for workflows that do not use multiple-record writes.
- A data map for a sequential source or target uses the SEQ access method. A data map for a VSAM target uses one of the following access methods: ESDS, KSDS, or RRDS.
- The source data map must specify unique record ID conditions so that each record in the input file is distributed to one and only one table. If the same record is distributed to more than one table, processing fails. This situation can occur if record ID conditions overlap or if you do not define record ID conditions in the data map.
- Verify that all tables that form a complete hierarchy are included in the data map. The import process does not verify hierarchical completeness.

  To ensure that all of the required tables in the data map are included in the source definition, define them in the data map as simple tables. A *simple table* is based on a single record.

  If the data map includes complex tables, PowerExchange does not include them in the source or target definition. A *complex table* is based on more than one record.
- You cannot select which tables in the data map to import. PowerCenter imports all of the simple tables in the data map.
- The data map must have exactly one table for each record.

Mappings and Transformations

The following rules and guidelines apply to PowerCenter mappings and transformations:

- Mapping transformations might drop rows. In this case, ensure that the mapping does not create any orphan rows. When a multiple-record write workflow runs, it drops orphan rows instead of writing them to the target.
- The mapping must not alter the position or the name of the generated sequence fields or drop them.
- Transformations that insert records into a mapping are not supported.
- Transformations that might alter row IDs, including the Sorter, Joiner, and Rank transformations, are not supported.

Connections

The following rules and guidelines apply to connections and connection properties:

- You can use multiple-record writes with PWXPC NRDB Batch connections only. PowerExchange ODBC connections are not supported.
- Select **Filter After** for the **Offload Processing** source and target connection attributes. If you select a different value, PowerExchange changes the value to **Filter After**.

  PowerExchange and PWXPC perform offload processing on the Integration Service machine before writing the output records to z/OS.
- Multithreaded processing is not supported for sequential or VSAM source or target connections.

  If you set the **Worker Threads** connection attribute to a nonzero value, the setting is ignored.
- Select **Off** for the **Confirm Write** source and target connection attributes. If you select a different value, PowerExchange changes the value to **Off**.
- In certain cases, you might need to change the value of the **CSQ_MEMSIZE** parameter of the **PWX Override** connection attribute. This parameter defines the maximum amount of memory that the cache can consume for multiple-record writes.

Session Properties

The following rules and guidelines apply to PowerCenter session properties:

- Pipeline partitioning is not supported on the reader or writer for sources or targets that were defined with sequencing enabled.
- You can specify the **File Name Override** property for sequential or VSAM sources and targets.
- For sources, the **Flush After N Blocks** property specifies the maximum number of block flushes that can occur across all groups without a given block being flushed. Define this property to ensure that all blocks are flushed at regular intervals.
- For SEQ targets on z/OS, PowerExchange can allocate the target data sets based on z/OS data set allocation properties that you optionally define in the session properties.

Performing a Multiple-Record Write to a Sequential File or VSAM Target

To transfer data in the correct physical sequence from a multiple-record sequential file source to a multiple-record sequential file or VSAM target, configure a session that performs multiple-record writes.

1. Create a multiple-record data map for the sequential file data source, if you have not already created one.

Enter the following information:

- To identify each record type, assign a record ID value in the **Record ID Values** box for a field.
- Assign a simple table to each record. Do not create a complex table.
- Assign the **Parent Record Name**, **Redefines Record Name**, or **Header/Trailer Record** optional property to each record, as appropriate. These properties are used to generate sequencing information in the source and target definitions.

For more information, see the *PowerExchange Navigator User Guide*.

2. If the source and target have different **Source Type** values, create a multiple-record data map for the target.

   If the source and target have the same data source type, you can use the same data map for both the source and target.
3. In the PowerCenter Designer, click **Sources > Import from PowerExchange**.
4. In the **Import from PowerExchange** dialog box, enter the required information.

   The following table describes the required information:

Attribute	Description
Location	Name of the node on which the source file resides. This value must match the name of a NODE statement in the PowerExchange dbmover.cfg file.
User Name	User name that has the database authority to connect to the source.
Password	Password that is associated with the user name.
Multi-Record Datamaps	Whether to list multiple-record data maps for selection. Select this check box.
Use Sequence Fields	Whether to generate sequence fields for multiple-record write operations. Select this check box.
Source Type	Data source type. The source type of the source and target must be the same. Select one of the following values: - **SEQ** for sequential files - **ESDS**, **KSDS**, or **RRDS** for VSAM data sets

   For more information about this dialog box, see *PowerExchange Interfaces for PowerCenter*.
5. Click **Connect.**

   The available multiple-record data maps appear in the **Selected Datamaps** box.
6. Select the data map that you want to import.
7. Click **OK.**

   The source definition appears in the workspace.
8. Click **Targets > Import from PowerExchange**.
9. In the **Import from PowerExchange** dialog box, enter the required information.

The following table describes the required information:

Attribute	Description
Location	Name of the node on which the target file resides. This value must match the name of a NODE statement in the PowerExchange dbmover.cfg file.
User Name	User name that has the database authority to connect to the target.
Password	Password that is associated with the user name.
Multi-Record Datamaps	Whether to list multiple-record data maps for selection. Select this check box.
Use Sequence Fields	Whether to generate sequence fields for multiple-record write operation. Select this check box.
Source Type	Data target type. The source type of the source and target must be the same. Select one of the following values: - **SEQ** for sequential files - **ESDS**, **KSDS**, or **RRDS** for VSAM data sets

Note: You cannot create a sequential file target definition for multiple-record writes by dragging and dropping the source definition.

10. Click **Connect.**

    The available multiple-record data maps appear in the **Selected Datamaps** box.

11. If the source and the target have the same **Source Type** values, select the same data map for the target definition that you selected for the source definition. Otherwise, select the data map for the target.

12. Click **OK.**

    The target definition appears in the workspace.

13. Create a mapping that includes the source definition, Source Qualifier transformation, and target definition.

    Between the Source Qualifier transformation and the target definition, you can include transformations that meet the requirements in "Rules and Guidelines for Multiple-Record Writes to Sequential File or VSAM Targets" on page 156.

14. Define PWX NRDB Batch connections for the source and target, as needed.

15. Define PowerCenter session properties:

    a. In the Task Developer, double-click the session to edit it.

    b. On the **Mapping** tab, click the **Sources** view.

    c. Under **Properties**, set the **File Name Override** property, if required, to the name of the source file.

    d. Define additional source properties as needed.

    e. On the **Mapping** tab, click the **Targets** view.

    f. Under **Properties**, set the **File Name Override** property, if required, to the name of the file that you want the session to allocate.

    g. Define additional target properties as needed.

Note: If you want PowerExchange to create the empty target file before the PowerCenter session runs, include the CREATEFILE command in the **Pre SQL** session property for the target. For more information, see the discussion of empty files in the appendix, "PowerExchange Interfaces for PowerCenter Tips," in the *PowerExchange Interfaces for PowerCenter* guide.

h. Click **OK**.

CHAPTER 13

VSAM Bulk Data Movement

This chapter includes the following topics:

Introduction to VSAM Bulk Data Movement

PowerExchange, in conjunction with PowerCenter, can move bulk data from or to the following types of VSAM data sets:

Entry-sequenced data set (ESDS)

A data set that stores records in the order in which they were created, with the latest records at the end. ESDSs have no index. ESDS records can be accessed directly by using a relative byte address (RBA) or sequentially based on entry sequence. PowerExchange accesses ESDS data sets sequentially.

Key-sequenced data set (KSDS)

A data set that contains records that are loaded in key sequence and controlled by an index. KSDSs can be accessed directly by using a key or RBA or sequentially based on key sequence.

By default, PowerExchange accesses KSDS records sequentially if no WHERE clause is specified.

PowerExchange accesses records by using a key under the following conditions:

- A WHERE clause is specified.
- The primary field in the KSDS key is used in one of more WHERE clause conditions.

For example, PowerExchange uses keyed access to perform a PowerCenter Lookup transformation that uses a KSDS lookup source when the lookup condition includes question marks that are substituted at run time. PowerExchange always uses a key to access a KSDS file if possible when processing a WHERE clause.

PowerExchange can access KSDS files through an alternate index that you specify by using a file name override. PowerExchange determines whether to use keyed or sequential access at run time after it opens the file and determines the start position and length of the key.

When you define the data map, you can optimize the data map for sequential access by enabling CI access and setting the number of data buffers to a high value. You can optimize the data map for keyed access by setting the number of index buffers to a high value and the Number of data buffers to a low value, such as 2.

Relative record data set (RRDS)

A data set that contains fixed-length records. RRDSs are accessed sequentially or directly by using a relative record number (RRN). PowerExchange accesses RRDS data sets sequentially based on RRN.

Variable-length relative record data set (VRRDS)

A data set that contains variable-length records. VRRDSs can be accessed sequentially or directly by using a relative record number (RRN). PowerExchange accesses VRRDS records sequentially based on RRN.

PowerExchange can process VSAM data sets that are in basic or extended format. PowerExchange can also processes data sets that contain compressed data.

Because VSAM is not a relational database, you must create a data map in the PowerExchange Navigator. PowerExchange uses the data map to access VSAM data and metadata and to create a relational row-type view of the records. PowerExchange requires a relational view to use SQL-type statements to read or write bulk data.

When you create VSAM data maps, you must select the access method that corresponds to the VSAM data set type. The following VSAM access methods are available:

- ESDS
- KSDS
- RRDS, for both fixed-length and variable-length relative record data sets

If a copybook is available for the VSAM data set, you can import it for the data map to get a detailed layout of each record. If you do not import a copybook, you will need to manually define the fields in the VSAM records.

Also, you must run a PowerExchange Listener on the MVS system where the VSAM source or target is located. Verify that this PowerExchange Listener can communicate with the system or systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run.

Configuring VSAM Bulk Data Movement

To configure PowerExchange for bulk data movement operations, the PowerExchange administrator sets up and tests connectivity to the PowerExchange Listener on the remote z/OS system and optionally includes the VSAM and SHOW_THREAD_PERF statements in the DBMOVER configuration member.

Restriction: To write data to VSAM ESDS, RRDS, or VRRDS data sets, PowerExchange can perform only Insert operations. To write data to KSDS data sets, PowerExchange can perform Insert, Update, and Delete operations.

Setting Up and Testing Connectivity to a Remote Source or Target

To access a data source or target on a remote system, PowerExchange must be able to communicate with the PowerExchange Listener on the remote system.

1. On the systems where the PowerExchange Navigator, PowerCenter Client, and PowerCenter Integration Service run, use the ping command to test network access to the remote source or target system.
2. Add the NODE statement to the dbmover.cfg file on each PowerExchange and PowerCenter system to identify the remote node where the source or target is located. Use the following syntax:

   ```
   NODE=(node_name,TCPIP,ipaddress_or_hostname,port_number)
   ```

   Tip: To determine the port number, consult with the person who installed the PowerExchange Listener on the remote system.
3. Run the DTLREXE ping utility on the local PowerExchange system to test connectivity to the remote PowerExchange Listener that you defined in the NODE statement.

Related Topics:

- "Managing the PowerExchange Listener" on page 45

Adding a VSAM-Specific Statement to the DBMOVER Configuration Member

To configure VSAM bulk data movement, you can include the optional VSAM statement in the DBMOVER configuration member on the VSAM source or target system.

Tip: The usual practice is to set these values in data maps for individual bulk data movements rather than to use this global parameter.

After editing the DBMOVER configuration member, you must restart the PowerExchange Listener for your changes to take effect.

For descriptions of all DBMOVER statements, see the *PowerExchange Reference Manual*.

VSAM Statement

The VSAM statement specifies the number of buffers that PowerExchange uses for data and index control intervals when processing VSAM data sets.

For more information about optimizing performance and system-managed buffering, see the DFSMS documentation.

Operating Systems: z/OS

Data Sources: VSAM data sets

Required: No

Syntax:

```
VSAM=([BUFNI=index_buffers]
     [,BUFND=data_buffers]
)
```

Parameters:

BUFNI=index_buffers

Optional. Number of I/O buffers that PowerExchange uses for VSAM index control intervals. A buffer is the size of a control interval in the index.

For the *index_buffers* variable, enter one of the following values:

- 0, to use the operating system default
- A number from 1 through 255

BUFND=*data_buffers*

Optional. Number of I/O buffers that PowerExchange uses for VSAM data control intervals. A buffer is the size of a control interval in the data component.

For the *data_buffers* variable, enter one of the following values:

- 0, to use the operating system default
- A number from 2 through 255

Note: You cannot specify 1 for BUFND.

Usage Notes:

- Additional index buffers improve performance by providing for the residency of some or all of the high-level index. Additional buffers minimize the number of high-level index records retrieved from DASD for key-direct processing.

  Tip: The usual practice is to set the number of buffers that PowerExchange uses for data and index control intervals in data maps for individual bulk data movement operations rather than to use the global VSAM statement to set these values.
- Additional data buffers benefit direct inserts or updates during control area splits and spanned record processing.
- The maximum number of buffers allowed is 255, which represents 254 data buffers and one insert buffer.

Moving VSAM Bulk Data

Use the following procedure to perform a VSAM bulk data movement operation. This procedure assumes that you are using PWXPC to integrate PowerExchange with PowerCenter.

Before you begin, get the following information:

- Fully-qualified VSAM data set name
- Fully-qualified data set name and member name of any COBOL or PL/I copybook that you want to use to define the layout of source or target records
- z/OS user ID and password, if required by the PowerExchange SECURITY statement setting

To move VSAM bulk data:

1. In the PowerExchange Navigator, create a data map. Include the following information:
   - In the **Access Method** list, select **ESDS**, **KSDS**, or **RRDS**, depending on type of VSAM data set that you are using as the source or target.
   - Select the **Import Record Definitions** option to be prompted to import metadata for VSAM records from a COBOL or PL/I copybook. If you do not select this option, you can import a copybook later by clicking **File > Import Copybook**.
   - Select the **Prefix record with XRBA value** option for an ESDS data set or the **Prefix record with RRN value** option for an RRDS data set if you want PowerExchange to add an 8-byte XRBA or RRN value as

a prefix in each record. If you select either option, make sure that the record layout in the data map includes sufficient space to accommodate the XRBA or RRN length.

- Select the **CI ACCESS** option if you want the PowerExchange to access the contents of the entire control interval rather than individual data records when reading VSAM data set. If you want to extract all data from the control interval, this option might improve sequential read performance. However, if you specify a WHERE clause, this option does not improve performance.
- For KSDS data sets, set the **Offload Processing** option to control if PowerExchange offloads processing for converting and merging data records from the PowerExchange Listener on the z/OS system to the system where extraction processing occurs. The extraction system is usually the system where the PowerCenter Client and PowerCenter Integration Service run.
- In the **Number of Data Buffers** field, enter the number of I/O buffers to use for the data control intervals that PowerExchange requests VSAM to allocate. Increase the number of buffers to improve sequential access performance. In the **Number of Index Buffers** field, for KSDS, RRDS, and VRDS data sets only, you can also specify the number of I/O buffers to use for index control intervals. These buffer values override the VSAM statement, if specified, in the DBMOVER configuration member on the z/OS system.

  Complete the other fields as needed. Click **Finish** when you are done.

  Restriction: PowerExchange does not support the CI ACCESS option for compressed data or spanned records.

  If you did not select the **Import Record Definitions** option, go to step "Moving VSAM Bulk Data" on page 164.

  If you selected the **Import Record Definitions** option, The **VSAM Copybook - Source Details** page is displayed.

2. Import a local or remote COBOL or PL/I copybook, if available, to define the layout of all or selected VSAM records.

   Depending on how you set the configuration options for the copybook import, the PowerExchange Navigator can prompt you to indicate for each record whether to import the record definition and to specify the record or field definition to use when alternative ones are defined in the copybook.

3. Open the data map in the Resource Explorer, and click the nonrelational record view or the relational table view to verify that the metadata was successfully retrieved.
4. To send the data map to the PowerExchange Listener on the remote z/OS system, select the data map in the Resource Explorer and click **File > Send to Remote Node**.

   In the **Location** field, enter the PowerExchange Listener node name that is specified in the NODE statement of the dbmover.cfg file on the local PowerExchange system. Complete the other fields, as needed.

   Note: If you do not complete this step, you are prompted to send the data map to the remote node when you run a database row test.

5. Run a database row test on the table view of the metadata to verify that actual data can be returned from the VSAM source database.

   Tip: If you need to change the data map to get the data you want, re-run the database row test on the revised data map.

6. In the PowerCenter Designer, import the PowerExchange data map and create a mapping.

   When you import the data map, in the **Location** field, enter the PowerExchange Listener node name that you specified in the NODE statement of the dbmover.cfg file. Also, specify **VSAM** in the **Source Type** field (also displayed for targets).

7. In the PowerCenter Workflow Manager, define a workflow and session and configure a PWX NRDB Batch application connection. Then start the workflow to perform the bulk data movement.

Group Source Processing of Multiple-Record VSAM Data Sets

PowerExchange uses group source processing to read bulk data from ESDS, KSDS, or RRDS data sets with multiple record types.

PowerExchange group source processing reads data that is stored in the same physical source in a single pass. By eliminating multiple passes of the source data, group source processing enhances bulk data movement performance and reduces resource consumption on the source system.

To use group source processing, complete the following tasks:

1. Create a multiple-record data map for the VSAM data source.

   Use the ESDS, KSDS, or RRDS access method.

2. Import the data map into PowerCenter.

   For PowerExchange to use group source processing, you must select the **Multi-Record Datamaps** option in the **Import from PowerExchange** dialog box.

Multiple-Record Writes to Sequential File or VSAM Targets

During bulk data movement sessions, PowerExchange can use a multiple-record data map to read data from a VSAM data set on z/OS and write the multiple record types in a single pass to nonrelational targets. This process is called a *multiple-record write*.

When PowerExchange performs a multiple-record write, it preserves source sequencing information. To enable multiple-record writes with sequencing, select the **Multi-Record Datamaps** and **Use Sequence Fields** options in the **Import from PowerExchange** dialog box for both the source and target definitions.

You can perform a multiple-record write from a VSAM ESDS, KSDS, or RRDS source to the following target types:

- VSAM ESDS, KSDS, or RRDS data sets on z/OS
- Sequential data sets on z/OS
- Files on i5/OS
- Files on Linux, UNIX, and Windows

To perform multiple-record writes with sequencing, PowerExchange uses group source processing to read source data in a single pass and uses group target processing to write data to the target in a single pass.

PowerExchange generates sequence fields to pass metadata about the source record relationships to the target. After you enable multiple-record writes with sequencing for a PowerCenter workflow, the workflow can read the multiple-record source, use the generated sequence fields to preserve the sequencing information, and write data to the target in the same record sequence as in the source.

To determine the relationships among records for sequential file and VSAM sources and generate sequence fields, PowerExchange uses the record properties that you define in the data map. These record properties include the parent record name, the name of the base record that the current record redefines, and whether the record is a header or trailer record. PowerExchange uses these record properties to generate primary and foreign keys and sequencing metadata in the source and target definitions.

When you run a PowerCenter session, PowerExchange uses the generated key values to reconstruct and write the data to the target in the correct sequence. PowerExchange maintains the data in a sequencing and queuing cache on the Integration Service node. When PowerExchange writes data to the target, it deletes the generated key fields and sends the data across the network in the correct sequence to the target file.

Rules and Guidelines for Multiple-Record Writes to VSAM or Sequential File Targets

Review the rules and guidelines that pertain to multiple-record writes to sequential file or VSAM targets before you configure a workflow that uses this feature.

For more information, see *PowerExchange Interfaces for PowerCenter*.

Source and Target Files

The following rules and guidelines apply to sources and targets:

- PowerExchange supports multiple-record writes to VSAM ESDS, KSDS, and RRDS targets. PowerExchange does not support multiple-record writes to VSAM VRRDS targets.
- PowerExchange does not support changing the values inside header or trailer records.
- A parent must precede its children in the source data.

Data Maps, Source Definitions, and Target Definitions

The following rules and guidelines apply to data maps, source definitions, and target definitions:

- In PowerCenter, you must create new source and target definitions for multiple-record writes. Existing definitions without sequencing information do not work for multiple-record writes. You can continue to use existing definitions for workflows that do not use multiple-record writes.
- A data map for a VSAM source or target uses one of the following access methods: ESDS, KSDS, or RRDS. A data map for a sequential file source or target uses the SEQ access method.
- The source data map must specify unique record ID conditions so that each record in the input file is distributed to one and only one table. If the same record is distributed to more than one table, processing fails. This situation can occur if record ID conditions overlap or if you do not define record ID conditions in the data map.
- Verify that all tables that form a complete hierarchy are included in the data map. The import process does not verify hierarchical completeness.

  To ensure that all of the required tables in the data map are included in the source definition, define them in the data map as simple tables. A *simple table* is based on a single record.

  If the data map includes complex tables, PowerExchange does not include them in the source or target definition. A *complex table* is based on more than one record.
- You cannot select which tables in the data map to import. PowerCenter imports all of the simple tables in the data map.
- The data map must have exactly one table for each record.

Mappings and Transformations

The following rules and guidelines apply to PowerCenter mappings and transformations:

- Mapping transformations might drop rows. In this case, ensure that the mapping does not create any orphan rows. When a multiple-record write workflow runs, it drops orphan rows instead of writing them to the target.
- The mapping must not alter the position or the name of the generated sequence fields or drop them.

- Transformations that insert records into a mapping are not supported.
- Transformations that might alter row IDs, including the Sorter, Joiner, and Rank transformations, are not supported.

Connections

The following rules and guidelines apply to connections and connection properties:

- You can use multiple-record writes with PWXPC NRDB Batch connections only. PowerExchange ODBC connections are not supported.
- Select **Filter After** for the **Offload Processing** source and target connection attributes. If you select a different value, PowerExchange changes the value to **Filter After**.

  PowerExchange and PWXPC perform offload processing on the Integration Service machine before writing the output records to z/OS.
- Multithreaded processing is not supported for sequential or VSAM source or target connections.

  If you set the **Worker Threads** connection attribute to a nonzero value, the setting is ignored.
- Select **Off** for the **Confirm Write** source and target connection attributes. If you select a different value, PowerExchange changes the value to **Off**.
- In certain cases, you might need to change the value of the **CSQ_MEMSIZE** parameter of the **PWX Override** connection attribute. This parameter defines the maximum amount of memory that the cache can consume for multiple-record writes.

Session Properties

The following rules and guidelines apply to PowerCenter session properties:

- Pipeline partitioning is not supported for the reader or writer for sources or targets that were defined with sequencing enabled.
- You must specify the **File Name Override** property for sequential or VSAM sources and targets.
- For sources, the **Flush After N Blocks** property specifies the maximum number of block flushes that can occur across all groups without a given block being flushed. Define this property to ensure that all blocks are flushed at regular intervals.

Performing a Multiple-Record Write to a Sequential or VSAM Target

To transfer data in the correct physical sequence from a multiple-record VSAM source to a multiple-record sequential or VSAM target, configure a session that performs multiple-record writes.

1. Create a multiple-record data map for the VSAM data source, if you have not already created one.

   Enter the following information:

   - To identify each record type, assign a record ID value in the **Record ID Values** box for a field.
   - Assign a simple table to each record. Do not create a complex table.
   - Assign the **Parent Record Name**, **Redefines Record Name**, or **Header/Trailer Record** optional property to each record, as appropriate. These properties are used to generate sequencing information in the source and target definitions.

   For more information, see the *PowerExchange Navigator User Guide*.
2. If the source and target have different **Source Type** values, create a multiple-record data map for the target.

If the source and target have the same data source type, you can use the same data map for both the source and target.

3. In the PowerCenter Designer, click **Sources > Import from PowerExchange**.
4. In the **Import from PowerExchange** dialog box, enter the required information.

   The following table describes the required information:

Attribute	Description
Location	Name of the node on which the source file resides. This value must match the name of a NODE statement in the PowerExchange dbmover.cfg file.
User Name	User name that has the database authority to connect to the source.
Password	Password that is associated with the user name.
Multi-Record Datamaps	Whether to list multiple-record data maps for selection. Select this check box.
Use Sequence Fields	Whether to generate sequence fields for multiple-record write operations. Select this check box.
Source Type	Data source type. For a VSAM data source, select one of the following source types: **ESDS**, **KSDS**, or **RRDS**.

   For more information about this dialog box, see *PowerExchange Interfaces for PowerCenter*.

5. Click **Connect.**

   The available multiple-record data maps appear in the **Selected Datamaps** box.
6. Select the data map that you want to import.
7. Click **OK.**

   The source definition appears in the workspace.
8. Click **Targets > Import from PowerExchange**.
9. In the **Import from PowerExchange** dialog box, enter the required information.

   The following table describes the required information:

Attribute	Description
Location	Name of the node on which the source file resides. This value must match the name of a NODE statement in the PowerExchange dbmover.cfg file.
User Name	User name that has the database authority to connect to the source.
Password	Password that is associated with the user name.
Multi-Record Datamaps	Whether to list multiple-record data maps for selection. Select this check box.

Attribute	Description
Use Sequence Fields	Whether to generate sequence fields for multiple-record write operation. Select this check box.
Source Type	Data target type. Select one of the following source types: - **SEQ** for sequential files - **ESDS**, **KSDS**, or **RRDS** for VSAM data sets

Note: You cannot create a sequential file target definition for multiple-record writes by dragging and dropping the source definition.

10. Click **Connect.**

    The available multiple-record data maps appear in the **Selected Datamaps** box.

11. If the source and the target have the same **Source Type** values, select the same data map for the target definition that you selected for the source definition. Otherwise, select the data map for the target.

12. Click **OK.**

    The target definition appears in the workspace.

13. Create a mapping that includes the source definition, Source Qualifier transformation, and target definition.

    Between the Source Qualifier transformation and the target definition, you can include transformations that meet the requirements in "Rules and Guidelines for Multiple-Record Writes to VSAM or Sequential File Targets" on page 167.

14. Define PWX NRDB Batch connections for the source and target, as needed.

15. Define PowerCenter session properties:

    a. In the Task Developer, double-click the session to edit it.

    b. On the **Mapping** tab, click the **Sources** view.

    c. Under **Properties**, set the **File Name Override** property to the name of the source file.

    d. Define additional source properties as needed.

    e. On the **Mapping** tab, click the **Targets** view.

    f. Under **Properties**, set the **File Name Override** property to the name of the file that you want the session to allocate.

    g. Define additional target properties as needed.

    h. Click **OK**.

CHAPTER 14

Writing Data with Fault Tolerance

This chapter includes the following topics:

Modes of Writing Data to PowerExchange Targets

PowerExchange can write data to databases in the following modes:

- **Synchronous**. PowerExchange sends a block of data to the PowerExchange Listener and waits for a response before sending another block. This mode preserves data integrity and recoverability but degrades performance.
- **Asynchronous**. PowerExchange sends a block of data to the PowerExchange Listener and does not wait for a response before sending another block of data. This mode provides optimal performance but no recoverability.
- **Asynchronous with Fault Tolerance**. PowerExchange sends blocks of data to the PowerExchange Listener asynchronously. The PowerExchange Listener writes errors to log files based on how you configured items such as the error handling, update strategy, error threshold, and reject files. Use this mode to optimize performance and to log errors based on the fault tolerance requirements of your site.

You can specify the write mode either in a PowerExchange Client for PowerCenter (PWXPC) database connection attribute or from the Windows ODBC Data Source Administrator.

The following table shows how to set the PWXPC **Write Mode** connection attribute or ODBC Write Mode setting for each write mode:

Write Mode	PWXPC Write Mode Setting	ODBC Write Mode Setting
Synchronous	Confirm Write On	`CONFIRMWRITE = Y` On the PowerExchange **Data Source** tab of the Windows ODBC Data Source Administrator, select **Confirm Write On** in the **Write Mode** list.
Asynchronous	Confirm Write Off	`CONFIRMWRITE = N` On the PowerExchange **Data Source** tab of the Windows ODBC Data Source Administrator, select **Confirm Write Off** in the **Write Mode** list.
Asynchronous with Fault Tolerance	Asynchronous with Fault Tolerance **Note:** Available only for PWX DB2zOS, PWX DB2i5OS, PWX DB2LUW, PWX MSSQL Server, and PWX Oracle relational connections.	`CONFIRMWRITE = T` On the PowerExchange **Data Source** tab of the Windows ODBC Data Source Administrator, select **Asynchronous With Fault Tolerance** in the **Write Mode** list.

Fault Tolerance Overview

Fault tolerance enables you to control how PowerExchange handles errors that are generated during database write operations. To configure asynchronous write with fault tolerance, you must perform the following tasks:

- Set the write mode to **Asynchronous with Fault Tolerance** to enable fault tolerance.
- Customize how PowerExchange handles errors. You can set the error handling behavior and the error threshold at which write processing stops.
- Specify the location and naming strategy for reject files. A reject file contains the rows of data that the writer did not write to targets.

You can use the Asynchronous Write with Fault Tolerance mode with the following databases:

- DB2 for Linux, UNIX, and Windows
- DB2 for i5/OS
- DB2 for z/OS
- Microsoft SQL Server
- Oracle on Linux, UNIX, or Windows

Related Topics:

- "Error Handling with Fault Tolerance" on page 173
- "Reject Files with Fault Tolerance" on page 175

Error Handling with Fault Tolerance

You can customize the way that PowerExchange handles write errors by setting the error handling behavior and the error threshold that controls when processing stops.

Configuring Error Handling

To define error handling for UPDATE and DELETE operations on nonexistent rows, include the ERRROWNOTFOUND statement in the DBMOVER configuration file on the target system:

ERRROWNOTFOUND Statement

The ERRROWNOTFOUND statement specifies whether PowerExchange generates or does not generate errors for UPDATE or DELETE operations on nonexistent rows.

Include the ERRROWNOTFOUND statement in the DBMOVER configuration file on the target system.

Data Sources: All

Required: No

Syntax:

```
ERRROWNOTFOUND={N|Y}
```

Valid Values:

- **N.** PowerExchange does not generate errors.
- **Y.** PowerExchange generates an error, increments the error count, and writes the record in error to the reject file.

Default is N.

Usage Notes: This statement is valid only with Asynchronous with Fault Tolerance write mode. To use this mode, set the **Write Mode** value to **Asynchronous with Fault Tolerance** in the PWXPC Connection attribute.

Configuring the Error Threshold

The error threshold specifies the maximum number of non-fatal errors that can occur. When this threshold is reached, PowerExchange stops write processing for the session.

The following table describes how to specify the error threshold from PWXPC and ODBC interfaces:

Interface	Configuration Method
PowerExchange Client for PowerCenter (PWXPC)	Change the value of the **stop on errors** attribute in the session properties. For more information, see *PowerExchange Interfaces for PowerCenter.*
Windows ODBC Data Source Administrator - PowerExchange Data Source tab	On the PowerExchange Data Source tab of the Windows ODBC Data Source Administrator dialog box, enter the value in the **Stop On Errors** field.
UNIX ODBC.ini file	Add the following parameter to the ODBC.ini file: `STOPONERRORS=number`

Creating Error Action Files for Customized Error Handling

You can customize fault tolerance behavior if you require a specific error handling strategy. However, this task is optional because PowerExchange provides a default list of recoverable and fatal error codes for each supported database.

Use the following procedure to customize PowerExchange behavior for certain database error codes.

To create error action files for customized error handling:

1. Create an error action file that defines all of the database error codes for which you want to customize PowerExchange handling of the error. You can use one of the sample error action files that PowerExchange provides as a template. To specify a fault tolerance option for an error code, use the following syntax:

   ```
   error_code,option
   ```

   Where:

   - *error_code* is a database-specific code, such as a DB2 SQLSTATE code or an Oracle ORA-*nnnnn* error code.
   - *option* is either **R** for recoverable or **F** for fatal.

     For example:

     ```
     00017,R
     00018,F
     ```

2. Specify the full path and file name of your customized error action file in the DBMOVER configuration file on the target system. Use the statement for your database, for example:

   ```
   DB2_ERRORFILE=path_filename
   ORA_ERRORFILE=path_filename
   MSS_ERRORFILE=path_filename
   ```

After the error action file is defined, PowerExchange uses the customized error code specifications instead of its default error handling. PowerExchange processes any error codes that you do not specify in the error action file as recoverable errors.

PowerExchange Example Error Action Files

The following table lists the sample error action files that PowerExchange supplies:

Database and Platform	Sample Error File
DB2 for z/OS	DB2ERR member in RUNLIB
DB2 for i5/OS	DB2ERR member in *datalib*/CFG
DB2 for Linux, UNIX, and Windows	db2err.act in the PowerExchange installation directory
Microsoft SQL Server	mssqlerr.act in the PowerExchange installation directory
Oracle	ora8err.act in the PowerExchange installation directory

Reject Files with Fault Tolerance

PowerExchange creates reject files on the target platform. The reject files contain the rows of data that the writer did not write to targets.

The writer might reject a row that violates a database constraint or that contains a field with truncated or overflow data if the target database is configured to reject such data.

You can read a reject file by using a text editor that supports the reject file code page.

In a reject file, the row and column indicators provide information that is noteworthy. For example, an attempt to insert a duplicate key generates the following reject file entry:

```
23,D,FAILNAME,D,1,D
```

To identify the reason why rows were rejected, review the column indicators and consult the session log.

Note: PowerCenter reject files are also created during a session that uses PowerExchange asynchronous write. However, the PowerCenter reject files are empty. The rejected data is written to the PowerExchange reject files.

Structure of Reject Files

PowerExchange reject files differ in structure from PowerCenter reject files. PowerExchange creates a separate reject file for each Insert (I), Update (U), Delete (D), and Mutated Update (M) operation. PowerCenter produces a single reject file for all operations.

Column Indicators in the Reject File

In PowerExchange reject files, a column indicator appears after every column of data to define the preceding data.

The following table describes the column indicators that PowerExchange uses:

Column Indicator	Type of Data	Writer Action
D	Valid data	Writer passes the data to the target database. The target accepts the data unless a database error, such as a duplicate key, occurs.
N	Null value	Writer passes the null value to the target. The null value is rejected if the target database does not accept null values.

Null values in reject files are indicated by “N,” for example:

```
1,D,,N,35,D
```

Binary values in reject files are indicated by the text string “BINARY DATA,” for example:

```
0,D,1,D,PAUL       ,D,BINARY DATA,D
```

Reject File Delimiter

The default delimiter in a reject file is a comma (,). If you want to override this default delimiter, you can specify another delimiter in the REJECT_FILE_DELIMITER statement of the DBMOVER configuration file. You might want to specify an override delimiter if the data that is written to the database contains commas.

To use a semicolon (;) as the delimiter, enter the semicolon enclosed in double-quotation marks:

```
REJECT_FILE_DELIMITER=";"
```

PowerExchange escapes a delimiter character that is part of the data by adding a backslash (\) before it. For example, if you use the default comma (,) delimiter, a comma in the data appears as follows:

```
\,
```

Alternatively, if you use the semi-colon (;) as the column delimiter, a semicolon in the data appears as follows:

```
\;
```

For more information about the REJECT_FILE_DELIMITER statement, see the *PowerExchange Reference Manual*.

Example of Reject File

The following example shows a reject file:

```
/* Insert reject file
/* Table: USER1.EMPLOYEE
/* Columns: EMPNO,ENAME,JOB,MGR,HIREDATE,SAL,COMM,DEPTNO
/* Column indicator key: D=Valid, N=Null
93,D,SURNAME,D,ANALYST,D,7902,D,20060705163431000000,D,9001.00,D,0.00,D,10,D
8,D,SURNAME,D,ANALYST,D,7902,D,20060705163431000000,D,9001.00,D,0.00,D,10,D
58,D,SURNAME,D,MANAGER,D,7902,D,20060705163431000000,D,9001.00,D,0.00,D,10,D
108,D,SURNAME,D,ANALYST,D,7902,D,20060705163431000000,D,1002.00,D,0.00,D,10,D
90,D,SURNAME,D,SALESMAN,D,7902,D,20060705163430000000,D,7001.00,D,0.00,D,10,D
92,D,SURNAME,D,ANALYST,D,7902,D,20060705163431000000,D,9001.00,D,0.00,D,10,D
82,D,SURNAME,D,MANAGER,D,7902,D,20060705163431000000,D,10001.00,D,0.00,D,10,D
132,D,SURNAME,D,SALESMAN,D,7902,D,20060705163432000000,D,2001.00,D,0.00,D,10,D
87,D,SURNAME,D,PRESIDENT,D,7902,D,20060705163431000000,D,4001.00,D,0.00,D,10,D
82,D,SURNAME,D,ANALYST,D,7902,D,20060705163432000000,D,10002.00,D,0.00,D,10,D
52,D,SURNAME,D,CLERK,D,7902,D,20060705163431000000,D,5001.00,D,0.00,D,10,D
```

Reject File Naming Conventions

How you specify the location and naming of reject files depends on your platform and whether you are using PWXPC or ODBC connections.

The following table summarizes configuration methods:

Interface	Configuration Method
PWXPC relational connection	Change the **Reject File** attribute in the Connection Object Definition dialog box to the required path and file name.
Windows ODBC Data Source Administrator - PowerExchange Data Source tab	On the PowerExchange Data Source tab, enter the path and file name in the **Reject File** field.
UNIX ODBC.ini file	REJECTFILE=*path_filename*

PowerExchange creates the reject files according to platform-specific rules.

Rules for Reject File Naming on Linux, UNIX, and Windows

On Linux, UNIX, and Windows, PowerExchange uses the following reject file naming rules:

- If you do not specify a reject file path and file name, PowerExchange creates a reject file in the PowerExchange installation directory that contains the DETAIL.LOG file. The reject file name has the following default format:

  pwxr.*schema_name*.*table_name*.*strategy*.*timestamp*.bad

- To create the reject file in another directory, enter a reject file path that ends with a slash (/) or backslash (\),whichever is valid for your platform. For example, you could specify "C:\PWX\Logs\" on Windows. PowerExchange generates the file name based on the default format.
- To add a prefix to the default file name and create the file in a specific directory, enter a reject file path that ends with a slash (/) or backslash (\) followed by a prefix, for example, "C:\PWX\Logs*prefix*." PowerExchange creates the file in the specified directory and generates a file name that has the following format:

  prefix.*schema_name*.*table_name*.*strategy*.*timestamp*.bad

Note: On Linux and UNIX, use a slash (/) in a reject file path. On Windows, use a backslash (\).

Rules for Reject File Naming on z/OS

On MVS, PowerExchange uses the following reject file naming rules:

- If you do not specify a reject file value, PowerExchange writes the reject data to the following data set:

  userid.*jname*.JOB*nnnnn*.PWXR.*strategy*.T*nnnnnnn*

  Where:

  - *userid* is the user ID under which the job runs.
  - *jname* is the job name.
  - JOB*nnnnn* is the job number.
  - *strategy* is I (Insert), U (Update), D (Delete), or M (Mutated Update).
  - T*nnnnnnn* is a unique name of up to eight characters in length.

    PowerExchange uses JOB*nnnnn* if the PowerExchange Listener or netport job runs as a batch job or uses STC*nnnnn* if the PowerExchange Listener or netport job runs as a started task.

- Optionally, you can enter a data set prefix. PowerExchange generates a unique data set name that has the following format:

  prefix.J*nnnn*.PWXR.*strategy*.T*nnnnnnn*

  The *prefix* qualifier can be up to 19 characters in length.

Rules for Reject File Naming on i5/OS

On i5/OS, PowerExchange uses the following reject file naming rules:

- If you do not specify a reject file value, all reject file members that are created for a single session are located under one FILENAME. PowerExchange uses the following defaults:

  LIBRARYNAME=DATALIB

This value is from the CRTPWXENV installation step.

- FILENAME=PWXR*nnnnnn*

  Where *nnnnnn* is the six-digit i5/OS job number.

  For example:

  FILENAME=PWXR713895

- MEMBERNAME=*strategy_i5OSjobnumber_uniquepwxref*

  The following table describes the variables in MEMBERNAME:

Variable	Description
strategy	I (Insert), U (Update), D (Delete), or M (Mutated Update)
i5OSjobnumber	A six-digit job number (*nnnnnn*)
uniquepwxref	A unique three-digit number (*nnn*)

  For example:

```
MEMBERNAME=I713895001
```

  Optionally, you can enter a LIBRARYNAME and FILENAME separated by a slash (/).

- If you specify a LIBRARYNAME only, PowerExchange creates the FILENAME based on the default FILENAME format.

  PowerExchange writes the reject file members for each session to a separate i5/OS file.

- If you specify a FILENAME that ends with a slash (/) or a FILENAME that is more than four characters in length but does not end with a slash (/), PowerExchange uses that value as specified.

  PowerExchange writes all of the reject file members for multiple sessions to a single i5/OS file.

  For example, assume that you enter the following reject file value:

```
MYLIBNAME/MYFILENAME/
```

  PowerExchange creates reject file members that have names in the following format:

```
MYLIBNAME/MYFILENAME/member_name
```

  For the first session, PowerExchange writes the following reject file members to an i5/OS file:

```
LIBRARY/MYFILENAME/I713895001
LIBRARY/MYFILENAME/U713895002
LIBRARY/MYFILENAME/D713895003
```

  For a subsequent session, PowerExchange adds the following reject file members to the same file:

```
LIBRARY/MYFILENAME/I713895001
LIBRARY/MYFILENAME/U713895002
LIBRARY/MYFILENAME/D713895003
LIBRARY/MYFILENAME/I839406001
LIBRARY/MYFILENAME/U839406002
LIBRARY/MYFILENAME/D839406003
```

- If you enter a FILENAME value that is one to four characters in length but does not end with a slash (/), PowerExchange appends a six-digit job number to the file name.

  For example, assume that you specify the following FILENAME value:

```
MYLIBNAME/PREF
```

  PowerExchange will create reject file members that have names in the following format:

```
MYLIBNAME/PREFnnnnnn/member_name
```

  PowerExchange writes the reject file members for each session to a separate i5/OS file.

  For example, for the first session, PowerExchange writes the following members to a single i5/OS file:

```
LIBRARY/PREF713895/I713895001
LIBRARY/PREF713895/U713895002
LIBRARY/PREF713895/D713895003
```

For a subsequent session, PowerExchange writes the following members to a new i5/OS file:

```
LIBRARY/PREF839406/I839406001
LIBRARY/PREF839406/U839406002
LIBRARY/PREF839406/D839406003
```

Disabling Reject File Creation

You can disable the creation of reject files by setting the REJECTFILE statement to PWXDISABLE.

The following table describes how to set this statement from each available interface:

Interface	Configuration Method
PWXPC relational connection	Change the **Reject File** attribute in the **Connection Object Definition** dialog box to PWXDISABLE.
Windows ODBC Data Source Administrator - PowerExchange Data Source tab	On the PowerExchange Data Source tab, enter PWXDISABLE in the **Reject File** field.
UNIX ODBC.ini file	REJECTFILE=PWXDISABLE

CHAPTER 15

Monitoring and Tuning Options

This chapter includes the following topics:

Monitoring Bulk Data Movement Sessions

PowerExchange issues messages that you can use to monitor the progress of bulk data movement sessions. PWXPC can also display progress and statistical information about bulk data movement sessions in the PowerCenter Workflow Monitor.

Monitoring Bulk Data Movement Sessions in PowerExchange

In PowerExchange, you can use the following methods to monitor data extraction by bulk data movement sessions:

- **Read progress messages.** You can request that PowerExchange write messages that indicate the number of data records read by a bulk data movement session.
- **Bulk data movement statistics messages.** When bulk data movement sessions end, PowerExchange writes messages that include statistical information about the quantity of data records processed.
- **Multithreaded processing statistics messages.** You can request that PowerExchange write statistical information about bulk data movement sessions that use multithreaded processing.
- **LISTTASK command output.** You can use the LISTTASK command to display active bulk data movement sessions.
- **DISPLAYSTATS command output.** You can use the DISPLAYSTATS command to publish monitoring statistics for a PowerExchange Listener that runs on Linux, zLinux, UNIX, Windows, or z/OS.

PowerExchange writes read progress messages, bulk data movement statistics messages, and multithreaded processing statistics messages to the PowerExchange log file. You can also configure PWXPC to write these and other PowerExchange informational messages to the PowerCenter session log.

By default, PWXPC writes error and warning messages but not informational messages to the session log. To configure PWXPC to write PowerExchange informational messages to the session log, define one of the following connection attributes or options:

- Select the **Retrieve PWX Log Entries** connection attribute. (PWXPC application connection types only)
- Add the RETLOGINFOMSG=Y option in the **PWX Override** connection attribute. (All PWXPC connection types)

The **Retrieve PWX Log Entries** connection attribute and the RETLOGINFOMSG=Y option of the **PWX Override** connection attribute are equivalent.

Read Progress Messages

You can configure PowerExchange to issue PWX-04587 messages to report read progress.

To direct PowerExchange to write read progress messages, include the following parameters in the DBMOVER configuration file:

- **PRGIND.** Specify Y to have PowerExchange write PWX-04587 messages that indicate the number of records read for a bulk data movement session. Default is N.
- **PRGINT.** Specify the number of records that PowerExchange reads before writing the PWX-04587 messages to the PowerExchange log file. Default is 250 records.

The PWX-04587 messages have the following format:

```
PWX-04587 int_server/workflow_name/session_name: Records read=num_records
```

Where:

- *int_server* is the name of the PowerCenter Integration Service.
- *workflow_name* is the name of the workflow that contains the bulk data movement session.
- *session_name* is the name of the bulk data movement session.
- *num_records* is the cumulative number of records read since the bulk data movement session started.

For example, to direct PowerExchange to write read progress messages after 100 records, the DBMOVER configuration file contains the following statements:

```
PRGIND=Y
PRGINT=100
```

When a bulk data movement session that has a session name of s_DB2_SQL_stats runs, PowerExchange writes the following messages to the PowerExchange log file:

```
PWX-04587 intserv/wf_blk_mon_stats/s_DB2_SQL_stats: Records read=100
PWX-04587 intserv/wf_blk_mon_stats/s_DB2_SQL_stats: Records read=200
PWX-04587 intserv/wf_blk_mon_stats/s_DB2_SQL_stats: Records read=300
```

PowerExchange continues to write PWX-04587 messages for this bulk data movement session until the session ends. In the PowerExchange log file, each of these messages has a date and time stamp. You can use this information to determine the rate at which PowerExchange processes bulk data on the source system.

Notes:

- For nonrelational bulk data movement sources, include the PRGIND and PRGINT statements in the DBMOVER configuration file on the PowerExchange Listener machine. For relational bulk data movement sources, include the PRGIND and PRGINT statements in the DBMOVER configuration file on the PowerCenter Integration Service machine.

- For DB2 relational connections, PowerExchange writes a PWX-04587 message after it reads the following number of records:

  The smallest multiple of the **Array Size** connection attribute that is greater than or equal to the value of PRGINT
- Because the PRGIND and PRGINT statements apply to every workflow that runs on the Integration Service machine and connects to the PowerExchange Listener, Informatica recommends that you use these statements judiciously.

Bulk Data Movement Statistics Messages

When a bulk data movement session ends, PowerExchange writes the following messages that contain statistical information about the session:

- **PWX-00408.** PowerExchange writes this message on the source system for each source in the bulk data movement session. This message includes the number of records and bytes read from the data source.
- **PWX-00409.** PowerExchange writes this message on the target system for the each target in the bulk data movement session. This message includes the number of records and bytes read for that bulk data movement session.

Multithreaded Processing Statistics

If you use bulk data offload processing, you can also use multithreaded processing to attempt to increase throughput on the PowerCenter Integration Service machine where the offloaded processing runs.

To issue these messages, you must specify the SHOW_THREAD_PERF statement in the DBMOVER configuration file on the PowerCenter Integration Service machine:

```
SHOW_THREAD_PERF=number_of_records
```

This statement specifies the number of records PowerExchange must process before writing statistics messages about multithreaded extraction processing to the PowerExchange message log file. For more information about this statement, see the *PowerExchange Reference Manual*.

Also, you must specify 1 or greater for the **Worker Threads** attribute on the application connection to implement multithreaded processing so that statistics can be generated.

PowerExchange writes some or all of the following messages during each statistics interval:

- **PWX-31255.** Cycle time, which is the total time that PowerExchange on the Integration Service machine spent processing the bulk data before passing it to PWXPC. This message includes the total percentage of time and average, minimum, and maximum times in microseconds.
- **PWX-31256.** I/O time, which is the time that PowerExchange on the Integration Service machine spent reading bulk data from the PowerExchange Listener on the source system. When reading local flat files, the I/O time is the time that PowerExchange spent reading the file. This message includes the I/O percentage of the total time and average, minimum, and maximum times in microseconds.
- **PWX-31257.** Parsing time, which is the time that PowerExchange on the Integration Service machine spent in column-level processing for the bulk data records on all threads. This message includes the parsing percentage of the total time and average, minimum, and maximum times in microseconds.
- **PWX-31258.** External time, which is the time that PowerExchange on the Integration Service machine spent combining the bulk data records from all threads back into a single stream to pass to PWXPC and for PWXPC to flush the data to PowerCenter. This message includes the external percentage of the total time and average, minimum, and maximum times in microseconds.
- **PWX-31259.** Delay time, which is the time that the PowerExchange on the Integration Service machine waited to receive new bulk data records to process from the PowerExchange Listener on the source

system. This message includes the delay percentage of the total time and average, minimum, and maximum times in microseconds.

- **PWX-31261**. PowerExchange joiner thread I/O time. If you use partitioning, this message is included and shows the amount of time spent by the joiner performing I/O during a multi-threaded session. When not performing I/O, the joiner is waiting for the partitions to complete processing.
- **PWX-31262**. Thread I/O delay. If you use partitioning, this message is included and shows the average time that each partition spends waiting for the reader to provide data to parse.

For example, if you specify SHOW_THREAD_PERF=10000 in the DBMOVER configuration file, PowerExchange writes the following sample messages after 10,000 bulk data records have been read:

```
PWX-31254 PowerExchange threading stats for last 10000 rows. Cycle (array) size is 25
rows. 0 out of array occured.
PWX-31255 Cycle  time: 100% (avg:      5709 min:     4741 max:      7996 usecs)
PWX-31256 IO     time:   4% (avg:       235 min:       51 max:      1021 usecs)
PWX-31257 Parse  time:  79% (avg:      4551 min:     4102 max:      5495 usecs)
PWX-31258 Extern time:  20% (avg:      1145 min:      618 max:      3287 usecs)
PWX-31259 Delay  time:   0% (avg:         7 min:        4 max:       165 usecs)
PWX-31254 PowerExchange threading stats for last 100000 rows. Cycle (array) size is 25
rows. 0 out of array occured.
PWX-31255 Cycle  time:  99% (avg:      5706 min:     4735 max:      7790 usecs)
PWX-31256 IO     time:   4% (avg:       234 min:       51 max:       950 usecs)
PWX-31257 Parse  time:  79% (avg:      4549 min:     4108 max:      5425 usecs)
PWX-31258 Extern time:  20% (avg:      1144 min:      616 max:      3242 usecs)
PWX-31259 Delay  time:   0% (avg:         7 min:        4 max:       115 usecs)
```

If the parsing and external processing times are greater than the I/O time, you might be able to improve throughput by increasing the number of threads for the bulk data movement session.

PowerExchange Listener LISTTASK Command

You can use the PowerExchange Listener LISTTASK command to display the bulk data movement sessions that are active in a PowerExchange Listener.

Alternatively, issue the pwxcmd listtask command from a Linux, UNIX, or Windows system to a PowerExchange Listener running on the local system or a remote system.

The command output includes the **PwrCntrSess** field. This field provides the PowerCenter session name in the following format:

*integration_server_*name/*workflow_name*/*session_name*

For example, if two active bulk data movement sessions are active, the command displays the following output:

```
PWX-00711 Active tasks:
PWX-00712 TaskId=1, Partner=10.10.10.01, Port=2480 ,
PwrCntrSess=intserv1/workflow1/bulk_sess1,
Application=appl_name1, Status=Active, AM=DB2, Mode=Append, Process=,  SessId=
PWX-00712 TaskId=2, Partner=10.10.10.02, Port=2480 ,
PwrCntrSess=intserv2/workflow2/bulk_sess2,
Application=appl_name2, Status=Active, AM=DB2, Mode=Append, Process=,  SessId=
PWX-00713 2 active tasks
PWX-00709 0 Dormant TCBs
```

PowerExchange Listener DISPLAYSTATS Command

You can use the PowerExchange Listener DISPLAYSTATS or pwxcmd displalystats command to publish monitoring statistics for a PowerExchange Listener that runs on i5/OS, Linux, zLinux, UNIX, Windows, or z/OS.

Before you run the command, configure the following statements in the DBMOVER configuration file:

- Specify the MONITOR parameter in the STATS statement in the DBMOVER configuration file to enable PowerExchange to collect these statistics. You can include the *interval* subparameter to publish the statistics at a regular interval as well as on demand.
- For the proper display of monitoring output on z/OS, set the LOG_LINE_LIMIT statement to 132. Otherwise, the lines might wrap awkwardly, making the output hard to read.

You can issue the command in any of the followings ways:

- From the command line on the Linux, UNIX, Windows, or zLinux system where the PowerExchange Listener runs.
- With the MVS MODIFY (F) command on the z/OS system where the PowerExchange Listener runs.
- With the pwxcmd program from a remote Linux, UNIX, and Windows system to a Listener on any supported operating system.

  Note: You must use this method to publish monitoring statistics for a PowerExchange Listener on i5/OS on demand.

The command syntax depends on the operating system type and whether you use pwxcmd. For more information, see the *PowerExchange Command Reference*.

Depending on which command parameter you use, you can publish one of the following types of reports:

- **Listener**. Reports PowerExchange Listener summary statistics on memory usage, CPU processing time, and activity on behalf of client requests. These statistics include counts of client tasks, connections, number of messages sent and received, bytes of data sent and received, and netport jobs (z/OS only). These statistics include both bulk data movement and CDC tasks.

  Note: If you run a PowerExchange Listener Service in the Informatica Domain, you can use the infacmd pwx displayStatsListener command to publish these statistics. For more information, see the *Informatica Command Reference*.
- **Accessmethods**. Reports statistics on PowerExchange Listener message and data transfer activity by client task and access method. For each active task and access method combination, these statistics include the number of rows read and written, bytes of data read and written, the source or target file name or data map file name, and the CPU processing time. For CDC requests that use the CAPX or CAPXRT access method, the report also includes counts of the SQL inserts, updates, and deletes that the task processed.
- **Clients**. Reports information about the active client tasks that are running under the PowerExchange Listener. For each task, the statistics show some or all of the following information: the status, access method, read or write mode, process name and session ID if available, CPU processing time, and start date and time. The statistics also include the client port number and IP address. If the client is PowerCenter, the statistics include the PowerCenter session ID and the application name for CDC.

By default, the Listener report is published.

The reports for a PowerExchange Listener on z/OS are similar to those for a PowerExchange Listener on i5/OS, Linux, zLinux, UNIX, or Windows.

The following example Listener report is for a PowerExchange Listener on z/OS:

```
PWX-00723 Command <displaystats Listener> succeeded
PWX-37101  Listener <PWXLST  > ASID=375 (x'177') UserID=AUSRID
PWX-37102   Memory
PWX-37103    Region below 16-MB line:  In Use        108 KB  Limit Value       9192 KB  Free       9084 KB
PWX-37104    Region above 16-MB line:  In Use      53912 KB  Limit Value    1675264 KB  Free    1621352 KB
PWX-37117   CPU Time
```

```
PWX-37118    TCB Time   = 0 SRB Time     = 0  zIIP-NTime = 0
PWX-37119    Listener   = 0 hrs, 0 mins, 1 secs, 275762 mcrs
PWX-37106   Cumulative Requests
PWX-37107    Total Tasks=        11 Active Tasks =         3 HWM Tasks =         3 Maxtasks  =        50
PWX-37108    Connections=        11 Accepted     =        11 Active    =         0
PWX-37109    Msgs Sent  =         0 Msgs Received=        22
PWX-37110    Data Sent  =         0 Data Received=      7304
PWX-37111    NetportJobs=         0
```

The **Memory**, **TCB Time**, **SRB Time**, and **NetportJobs** values are specific to the PowerExchange Listener on z/OS. For a PowerExchange Listener on i5/OS, Linux, UNIX, or Windows, the report displays the total memory usage.

You can use this report determine if the number of client tasks is reaching the limit that is set in the MAXTASKS statement of the DBMOVER configuration file. Compare the **HWM Tasks** value to the **Maxtasks** value. If the HWM Tasks value reaches the MAXTASKS limit, PowerExchange Listener processing might be delayed, which can cause reduced throughput and connection timeouts.

The following example accessmethods report is for a PowerExchange Listener on z/OS, but the same fields are displayed for a PowerExchange Listener on i5/OS, Linux, UNIX, Windows, or zLinux:

```
PWX-00723 Command <displaystats AccessMethods> succeeded
PWX-37201   Active Access Methods
PWX-37202    Task ID    = 42412 AM           =     CAPXRT
PWX-37203                    Rows read    =       1029  Rows written  =          0
PWX-37204                    Bytes read   =     116277  Bytes written =          0
PWX-37205                    File         = d2ivd0.d002root_ROOT
PWX-37206                    Table        = <Capture Extract Realtime>
PWX-37208                    Inserts      =        564  Updates       =          0  Deletes     =        465
PWX-37121                    CPU time     = 0 hrs, 0 mins, 0 secs, 299809 mcrs
PWX-37202    Task ID    = 42414 AM           =       NRDB
PWX-37203                    Rows read    =         10  Rows written  =          0
PWX-37204                    Bytes read   =        570  Bytes written =          0
PWX-37205                    File         = ABC.VSAM.MASTER_REC
PWX-37206                    Table        = <Non-relational source>
PWX-37202    Task ID    = 42414 AM           =       KSDS
PWX-37203                    Rows read    =         10  Rows written  =          0
PWX-37204                    Bytes read   =        800  Bytes written =          0
PWX-37205                    File         = XYQ.TEST.V1.KSDS
PWX-37206                    Table        = XYQ.TEST.V1.KSDS
PWX-37121                    CPU time     = 0 hrs, 0 mins, 0 secs, 76151 mcrs
```

For the CAPXRT and CAPX access methods, the report includes the number of SQL inserts, updates, and deletes that the task processed for a CDC request.

A client task can have multiple access methods, for example, one for reading source data and one for mapping nonrelational source data to a relational format. In the example output, task 42414 uses the NRDB access method with the data map file specified in the **File** field to map nonrelational data to a relational format. The same task uses the KSDS access method to retrieve data from the KSDS data set specified in the **File** field.

The following example clients report is for a PowerExchange Listener on Windows, but the same fields are displayed for a PowerExchange Listener on i5/OS, Linux, zLinux, UNIX, or z/OS:

```
PWX-00723 Command <displaystats Clients> succeeded
PWX-37112   Active Tasks
PWX-37113    Task ID    = 41942 Status       = Active
PWX-37114                    Port         = 2480  Partner = 127.0.0.1
PWX-37115                    PwrCntrSess  = N/A
PWX-37207                    Application  = N/A
PWX-37116                    AM           = NRDB  Mode    = Read  Process = DTLLST3 SessionId =
PWX-37121                    CPU time     = 0 hrs, 0 mins, 0 secs, 62400 mcrs
PWX-37122                    Start time   = 2014-05-01 14:21:37
PWX-37113    Task ID    = 41943 Status       = Active
PWX-37114                    Port         = 2480  Partner = 127.0.0.1
PWX-37115                    PwrCntrSess  = N/A
PWX-37207                    Application  = N/A
PWX-37116                    AM           = NRDB  Mode    = Read  Process = DTLLST3 SessionId =
PWX-37121                    CPU time     = 0 hrs, 0 mins, 0 secs, 124800 mcrs
PWX-37122                    Start time   = 2014-05-01 14:22:01
```

The **Partner** field displays the IP address of the client that issued the request that caused the PowerExchange Listener to create the task. This value begins with ::ffff for an IPv6 address.

For more information about the fields in each of these reports, see the *PowerExchange Command Reference*.

Tuning Bulk Data Movement Sessions Overview

To tune bulk data movement sessions, you can use PowerExchange DBMOVER configuration statements and PWXPC connection attributes. You can also use connection pooling, data maps caching, offload processing, and multithreaded processing. And on z/OS, you can assign the PowerExchange Listener started task to the appropriate Workload Manager service class.

Use any of the following tuning methods:

- Configuration statements and connection attributes. Define certain DBMOVER statements and PWX Batch connection attributes.
- Connection pooling. Network connection information is cached and reused, which decreases the connection time for subsequent connections. For example, opening a connection to a PowerExchange Listener on z/OS might take two elapsed seconds. With connection pooling, subsequent connections to the PowerExchange Listener take a fraction of a second.
- Data maps caching. PowerExchange retrieves data maps defined for z/OS nonrelational data sources from job-level memory rather than by accessing the data maps file. By eliminating accesses to the data maps file, data maps caching improves performance.
- Offload processing. For DB2 for z/OS tables and image copies, IMS unload data sets, VSAM data sets, and sequential files, use offload processing to distribute PowerExchange bulk data column-level processing and filtering to the PowerCenter Integration Service machine that runs the bulk data movement session. By distributing processing to another machine, you reduce PowerExchange bulk data movement overhead on the source system.
- Multithreaded processing. If you use bulk data offload processing for DB2 for z/OS tables, VSAM data sets, or sequential files, you can also use multithreaded processing to attempt to increase throughput. Multithreaded processing uses multiple threads on the PowerCenter Integration Service machine to perform offloaded PowerExchange processing.
- Pipeline partitioning. Use multiple partitions in a pipeline for a bulk data movement session to improve session performance. You can use reader partitioning, writer partitioning, or both. Partitioning enables the session to process data in the partitions concurrently, which is especially beneficial for resource-intensive, column-level processing that is offloaded to the PowerCenter Integration Service machine.
- Workload Manager (WLM) service classes. Assign an appropriate z/OS WLM service class to the PowerExchange Listener to ensure that processes that access your z/OS data through PowerExchange meet the desired performance goals.

Using PowerExchange DBMOVER Statements to Tune Bulk Data Movement Sessions

To tune PowerExchange bulk data movement sessions, you can include the following statements in the DBMOVER configuration file:

APPBUFSIZE=*bytes*

Defines the maximum size, in bytes, of the buffer that PowerExchange uses to read or write data. This data buffer can exist on a source or target system.

PowerExchange writes bulk data to its application data buffer on the source system until the buffer is full. PowerExchange then sends the data to a sending TCP/IP buffer on the source system. TCP/IP transports the bulk data to a receiving TCP/IP buffer on the target system. PowerExchange on the target

system reads the bulk data from the TCP/IP buffer into its application data buffer. PWXPC then reads the bulk data and passes it to PowerCenter. PowerCenter processes the data and applies it to the targets.

Enter an APPBUFSIZE value that is greater than the maximum size of any single data row to be sent.

Valid values are from 34816 through 8388608. Default is 256000.

If the target system is remote, enter the same APPBUFSIZE value in the DBMOVER configuration files on the source and target systems.

If the APPBUFSIZE value is not optimal, PowerExchange writes the PWX-01295 message in the PowerExchange log file on the source system. This message includes a recommended value.

If dynamic application buffer sizing is enabled, the APPBUFSIZE statement defines the initial size of the application data buffer for all connections made during a PowerExchange Listener run. PowerExchange resizes the application data buffer dynamically for individual connections as needed. Dynamic application buffer sizing is enabled by default. You can explicitly enable it by specifying Y for the APPBUFSIZEDYN statement in the DBMOVER configuration file.

APPBUFSIZEDYN={N|Y}

Specifies whether to enable dynamic application buffer sizing.

The DBMOVER APPBUFSIZE statement defines the initial size of the application buffer for all connections made during a PowerExchange Listener run. If APPBUFSIZEDYN=Y, PowerExchange resizes the application buffers for individual connection as needed.

The APPBUFSIZEDYN statement applies to PowerExchange connections to data sources with either fixed-length or variable-length records. A variable-length record is a record with at least one variable-length field. A variable-length field has a datatype of VARCHAR or VARBIN.

For each connection to a data source with variable-length records, PowerExchange resizes the application buffer when it encounters a record that is too large to fit into the buffer. PowerExchange increases the size of the application buffer to a value of ten times the size of the record that has overflowed, up to the maximum application buffer size of 8 MB. The new size remains in effect for the duration of the Listener run or until the application buffer is resized again. PowerExchange never decreases the application buffer size for a connection after the Listener run has started.

For each connection to a data source with fixed-length records, PowerExchange determines the record length when the connection is opened and resizes the application buffer once, up to the maximum application buffer size of 8 MB, as needed.

COMPRESS={Y|N}

Defines whether PowerExchange uses its proprietary compression algorithm to compress data before PowerExchange sends data to TCP/IP for transmission to the remote system.

Default is Y.

PowerExchange uses the COMPRESS setting in the DBMOVER configuration file on the remote system that contacts the PowerExchange Listener. On PWX NRDB Batch application connections and PWX relational connections, you can override the compression setting in the DBMOVER configuration file. If you enable compression, the CPU consumption of the PowerExchange Listener on the source system might increase.

To avoid unnecessary CPU consumption, set COMPRESS to N in the PowerExchange DBMOVER configuration file on the PowerCenter Integration Service machine.

LISTENER=(*node_name*,TCPIP,*port*,*send_bufsize*,*receive_bufsize*,*send_size*,*receive_size*, ...)

A TCP/IP port on which a named PowerExchange Listener process listens for work requests.

The *send_bufsize* and *receive_bufsize* positional parameters define the data portion of the TCP/IP send and receive buffer sizes that PowerExchange uses. If you do not specify these values, PowerExchange uses the operating system defaults.

To increase throughput, try increasing the *send_bufsize* and *receive_bufsize* values in the LISTENER statement in the DBMOVER configuration file on the source system. For help in determining the best values to use, contact your network administrator.

NODE=(*node_name*,TCPIP,*host_name*,*port*,*send_bufsize*,*receive_bufsize*,*send_size*,*receive_size*, ...)

A TCPIP host name and port that PowerExchange uses to contact a PowerExchange Listener process.

The *send_bufsize* and *receive_bufsize* positional parameters define the data portion of the send and receive buffer sizes that PowerExchange uses. If you do not specify these values, PowerExchange uses the operating system defaults.

To increase throughput, try increasing the *send_bufsize* and *receive_bufsize* values in the NODE statement in the DBMOVER configuration file on the target system. For help in determining the best values to use, contact your network administrator.

TRACE=(*trace_id*,*trace_level*,99)

Activates PowerExchange diagnostic traces that Informatica Global Customer Support uses to solve problems with PowerExchange code.

TRACE statements can severely impact PowerExchange performance. Use these statements only at the direction of Informatica Global Customer Support.

After Informatica Global Customer Support diagnoses the problem, remove or comment out all TRACE statements in the DBMOVER configuration files on all systems.

Using Connection Attributes to Tune Bulk Data Movement Sessions

In PowerCenter, you can customize options on PWX NRDB Batch application connections and PWX bulk relational connections to tune bulk data movement sessions.

The following table describes the connection attributes that you can use to tune bulk data movement sessions:

Connection Attribute	Description	Tuning Suggestion
Compression	Select this option to compress source data during the PowerCenter session. Default is disabled.	Do not use compression.
Encryption Type	The type of data encryption that PowerExchange uses. Default is None.	Do not use encryption.
Offload Processing	Select this option to request bulk data offload processing. Default is No.	For more information, see "Bulk Data Offload and Multithreaded Processing" on page 193.

Connection Attribute	Description	Tuning Suggestion
Worker Threads	If you enable offload processing for data types that support worker threads, set this option to use multiple threads for offloaded processing on the Integration Service machine. Enter the number of threads that you want PowerExchange to use. Valid values are from 1 through 64. Default is 0, which disables multithreading.	For more information, see "Bulk Data Offload and Multithreaded Processing" on page 193.
Array Size	For VSAM data sets or sequential files, if the **Worker Threads** value is greater than zero, **Array Size** specifies the size of the storage array, in number of records, for the threads. For DB2 for z/OS tables, **Array Size** specifies the DB2 fetch array size. For DB2 for z/OS image copies, **Array Size** determines the maximum number of memory buffers that might need to be allocated. If zIIP processing is enabled, **Array Size** also specifies how many compressed rows to dispatch to the zIIP processor. Valid values are 25 through 5000. Default is 25.	Informatica recommends that you use the default value of 25 unless you are able to test and determine whether the extra memory that is allocated to a larger array size has been beneficial and has not degraded server performance. If you are able to make these determinations, Informatica recommends an array size of 500 to 1000 in the following cases: - You enable offload processing and select a nonzero value for **Worker Threads**. - You are using the SQL Server bulk load utility to load data to SQL Server tables. - You use a compressed image copy as a data source for PowerExchange bulk data movement, and you have enabled zIIP processing on z/OS. Informatica does not recommend increasing the array size in the following cases: - You are performing NRDB processing and not using offload processing. - You are moving LOB columns or rows that are very large because they contain many columns.

Related Topics:

- "Bulk Data Offload and Multithreaded Processing" on page 193

Connection Pooling

Connection pooling is a method of reusing cached connection information and eliminating overhead in opening network connections and creating new tasks.

When connection pooling is enabled, PowerExchange maintains a pool of PowerExchange Listener connections. It does not keep files or databases open for the pooled connections.

The following clients support connection pooling:

- Data Integration Service. Define connection pooling properties for the Data Integration Service by using the Informatica Administrator. Before you enable connection pooling, verify that the value of the MASTASKS statement in the DBMOVER file on the PowerExchange Listener system is large enough to accommodate the maximum number of connections in the pool for the Listener task. For more information, see the *Informatica Administrator Guide*.
- PowerExchange Navigator. Configure connection pooling for the PowerExchange Navigator by defining connection pooling properties in the **Preferences** dialog box. For more information, see the *PowerExchange Navigator User Guide*.
- DTLUCBRG utility. DTLUCBRG automatically enables connection pooling with a connection pool size of six connections and a retention period of 30 seconds. You cannot change these values.
- DTLURDMO utility. DTLURDMO automatically enables connection pooling with a connection pool size of two connections and a retention period of 30 seconds. You cannot change these values.
- PowerExchange Agent on z/OS. The PowerExchange Agent uses connection pooling automatically in certain cases. You cannot configure connection pooling for the PowerExchange Agent.
- PowerCenter Integration Service. The PowerCenter Integration Service uses connection pooling automatically in certain cases. You cannot configure connection pooling for the PowerCenter Integration Service.
- Informatica Developer tool. The Developer tool uses connection pooling automatically in certain cases. You cannot configure connection pooling for the Developer tool.

Data Maps Caching

You can configure PowerExchange to cache data maps defined for nonrelational z/OS data sources on the system where the data maps are used by a single PowerExchange Listener job or by multiple PowerExchange Listener and netport jobs.

Note: You cannot use data maps caching with IMS netport jobs.

When you enable data maps caching, PowerExchange retrieves data maps from job-level memory rather than by accessing the data maps file. By eliminating accesses to the data maps file, data maps caching improves performance.

When you enable data maps caching, you can decrease initialization time by 0.1 second in addition to the performance improvements that you achieve through connection pooling. Data maps caching can be of use when many short transactions are completed for each second. However, if you run long-running bulk data movement sessions and initialization time is a small part of the overall processing time, data maps caching might not give you any performance benefits.

To enable and configure data maps caching, define DBMOVER statements on the system where the PowerExchange Listener or netport jobs run. You can configure PowerExchange to run data maps caching in either single-job mode or multiple-jobs mode.

Enabling Data Maps Caching

To enable data maps caching, set the DMXCACHE_MAX_MEMORY_MB statement to a value greater than 0 in the DBMOVER configuration file.

This value configures the maximum size for the data maps cache in megabytes.

For information about the DMXCACHE_MAX_MEMORY_MB statement, see the *PowerExchange Reference Manual*.

Note: Verify that the REGION size specified in the DTLLST step in the JCL is large enough to run the PowerExchange Listener job. For example, if you define DMXCACHE_MAX_MEMORY_MB=20, you might have to increase the REGION size by 20 MBs.

Configuring Data Maps Caching to Run in Single-Job or Multiple-Jobs Mode

You can configure PowerExchange to run data maps caching in either single-job mode or multiple-jobs mode.

If you use one PowerExchange Listener job and no netport jobs, configure PowerExchange to run data maps caching in single-job mode.

If you use multiple PowerExchange Listener or netport jobs to access the same data maps file, configure PowerExchange to run data maps caching in multiple-jobs mode.

Single-Job Mode

In single-job mode, PowerExchange maintains copies of previously read data maps in job-level memory. PowerExchange does not use ESCA memory to maintain information about data maps.

On subsequent reads of data maps, PowerExchange retrieves the data maps from job-level memory, which eliminates the overhead of enqueues, opens, points, reads, and closes of the data maps KSDS file.

When a data map changes or is deleted, PowerExchange deletes the copy of that data map in the cache. For changed data maps, PowerExchange does not add the data map to the cache again until the data map is actually used.

Single-job mode is more efficient than multiple-jobs mode because in single-job mode, the PowerExchange Listener job is the only job that updates the data maps file and the data maps cache does not become stale. Consequently, PowerExchange never needs to completely clear the cache like it does in multiple-jobs mode.

To configure PowerExchange to run data maps caching in single-job mode, define the following DBMOVER configuration statement:

```
DMXCACHE_MULTIPLEJOBS=N
```

For information about the DMXCACHE_MULTIPLEJOBS statement, see the *PowerExchange Reference Manual*.

Multiple-Jobs Mode

In multiple-jobs mode, PowerExchange maintains copies of previously read data maps in job-level memory. Additionally, when you start a PowerExchange Listener, PowerExchange dynamically allocates 4096 bytes of ECSA memory in which PowerExchange maintains the name of the data maps KSDS file and the time stamp of its last update. PowerExchange uses this information to determine whether the data maps cache is stale and needs to be cleared.

On subsequent reads of data maps, PowerExchange retrieves the data maps from job-level memory, which eliminates the overhead of enqueues, opens, points, reads, and closes of the data maps KSDS file.

When a data map changes or is deleted, PowerExchange completes the following processing:

- In ECSA memory, PowerExchange changes the time stamp of the last update to the data maps KSDS file.
- When another task tries to access any data map, PowerExchange determines if the cache is stale by comparing the time stamp in ECSA memory against the time stamp of the data maps cache. If the cache is stale, PowerExchange clears it and PowerExchange reads the required data map from disk and adds it to the cache.

To configure PowerExchange to run data maps caching in multiple-jobs mode, define the following DBMOVER configuration statement:

```
DMXCACHE_MULTIPLEJOBS=Y
```

Additionally, if you want PowerExchange to free the ECSA memory after the PowerExchange Listener closes, define the following DBMOVER configuration statement:

```
DMXCACHE_DELETEECSA=Y
```

However, because shared memory allocation is a complex task and 4096 bytes of memory is a small amount, you can accept the default value, which is N, to permanently retain the ECSA memory.

For information about these DBMOVER statements, see the *PowerExchange Reference Manual*.

Configuring Data Maps Caching - Example

In this example, you enable data maps caching and configure a maximum size of 20 MBs for the data maps cache, which is large enough to hold 20 large IDMS data maps or thousands of small data maps.

Verify that the REGION size specified in the DTLLST step in the JCL is large enough to run the PowerExchange Listener job. For example, if you define DMXCACHE_MAX_MEMORY_MB=20, you might have to increase the REGION size by 20 MBs.

1. To enable data maps caching and to configure a data maps cache size of 20 MBs, define the following statement in the DBMOVER configuration file:

   ```
   DMXCACHE_MAX_MEMORY_MB=20
   ```

2. Based on your environment, configure PowerExchange to run data maps caching in single-job or multiple-jobs mode.

   The following table describes the environment and configuration steps for each mode:

Environment	Mode	Configuration Steps
You use one PowerExchange Listener job and no netport jobs.	Single-job mode	Define the following DBMOVER configuration statement: `DMXCACHE_MULTIPLEJOBS=N` In this situation, the default cache size is sufficient. Default is N. For more information about single-job mode, see "Single-Job Mode" on page 191.

You use multiple PowerExchange Listener or netport jobs to access the same data maps file.	Multiple-jobs mode	Define the following DBMOVER configuration statement:
		`DMXCACHE_MULTIPLEJOBS=Y`
		Additionally, if you want PowerExchange to free the ECSA memory when the PowerExchange Listener closes, define the following DBMOVER configuration statement:
		`DMXCACHE_DELETEECSA=Y`
		However, because shared memory allocation is a complex task and 4096 bytes of memory is a small amount, you can accept the default value, which is N, to permanently retain the ECSA memory.
		For more information about multiple-jobs mode, see "Multiple-Jobs Mode" on page 191.

3. Optionally, if you run data maps caching in multiple-jobs mode, you can run the PWXUDMX utility to view the ECSA memory.

For more information about the PWXUDMX utility, see the *PowerExchange Utilities Guide*.

4. To determine if data maps caching improved performance, close the PowerExchange Listener and review messages in the PowerExchange log.

Bulk Data Offload and Multithreaded Processing

When you use bulk data offload processing, PowerExchange moves the column-level processing to the PowerCenter Integration Service machine that runs the bulk data movement session. For most data sources, PowerExchange can optionally move the filtering of source data to the PowerCenter Integration Service machine.

You can use bulk data offload processing for the following data sources:

- DB2 for z/OS tables
- DB2 for z/OS image copies
- Sequential and flat files
- VSAM data sets
- IMS unload files

You can use bulk data offload processing for the following data targets:

- z/OS sequential files
- VSAM data sets
- IMS unload files

For most data sources, you can use multithreaded processing in bulk data movement sessions that use offload processing. By default, PowerExchange uses a single thread to process bulk data on the PowerCenter Integration Service machine. When you configure multithreaded processing, PowerExchange uses multiple threads to process the bulk data records.

Rules and Guidelines for Bulk Data Offload Processing

The following restrictions apply to bulk data movement operations that use offload processing:

- You cannot perform bulk data offload processing for data maps that use any of the following options:
  - User access methods
  - User-defined fields that invoke programs by using the CALLPROG function
  - Record-level exits
- For source data maps that have an **Encoding** setting other than **Default** in the Access Method dialog box, specify **Filter After** for the **Offload Processing** option on the PWX NRDB connection. For these data maps, do not specify **Filter Before** for the **Offload Processing** option.
- If you select **Filter Before** for an IMS unload file DB2 for z/OS image copy data source, PowerExchange changes the selection to **Filter After**.
- You cannot include multiple data targets in a mapping that point to the same VSAM file.
- Only inserts to data targets are supported. Deletes and updates are not supported.
- The data map for a data target must not use record-level exits.

Rules and Guidelines for Multithreaded Processing

In specific situations, multithreaded processing can help improve performance of bulk data movement sessions.

Before you configure multithreaded processing, review the following rules and guidelines:

- Use multithreaded processing when the PWX reader thread of a bulk data movement session uses 100% of a CPU on a multi-CPU server on the PowerCenter Integration Service machine. In this situation, multithreaded processing can help improve session throughput by spreading bulk data processing across multiple threads. However, if the single CPU is not fully consumed, additional threads do not improve throughput.
- If the network processing between the source system and PowerCenter Integration Service machine is slow, specify 1 for the **Worker Threads** connection attribute to help improve throughput. When you use 1 or more worker threads, PowerExchange overlaps network processing with bulk data processing on the PowerCenter Integration Service machine.
- For optimal performance, the value for the **Worker Threads** attribute cannot exceed the number of installed or available processors on the PowerCenter Integration Service machine.
- You cannot use multithreaded processing for sequential-file and flat-file data maps that specify **VS** for the **Variable** property in the SEQ Access Method dialog box.
- To use multithreaded processing on Linux, UNIX, or Windows for z/OS sequential files that have variable length records, ensure that the file transfer mechanism does not remove the record descriptors. Each record must be prefixed with the record length.
- You cannot use multiple worker threads to write data to targets. Instead, consider using writer partitioning if the data contains only Inserts.
- If you use reader or writer partitioning for a bulk data movement session, the **Worker Threads** setting is ignored.
- Not all PWX NRDB Batch connection types support worker threads. If the **Worker Threads** connection attribute for one of these connections is set to a nonzero value, processing continues without threads.
- Message PWX-31505 reports connection performance settings, including the number or worker threads. If worker threads are not used, message PWX-31505 reports a zero value for worker threads.

Enabling Offload and Multithreaded Processing for Bulk Data Movement Sessions

To use bulk data offload processing and multithreaded processing, you must configure some connection attributes for the bulk data movement session.

For sequential and flat files, VSAM data sets, IMS unload data sets, and DB2 image copies, configure the attributes on the PWX NRDB Batch application connection for the bulk data movement session.

The following table describes the attributes to configure in the PWX NRDB Batch application connection:

Connection Attribute	Description
Offload Processing	Specifies whether to use offload processing to move PowerExchange bulk data processing from the source or target system to the PowerCenter Integration Service. For data sources, select one of the following values: - **No**. Do not use offload processing. - **Auto**. PowerExchange determines whether to use offload processing. - **Filter Before**. Offload column-level processing to the Integration Service machine, but continue to filter data on the source system. - **Filter After.** Offload column-level processing and data filtering to the Integration Service machine. For data targets, select one of the following values: - **No**. Do not use offload processing. - **Auto**. Do not use offload processing. - **Filter Before**. Use offload processing. Filtering does not apply to offload processing for data targets. - **Filter After.** Use offload processing. Filtering does not apply to offload processing for data targets. Default is No.
Worker Threads	When offload processing is enabled, specifies the number of threads that PowerExchange uses on the Integration Service machine to process bulk data. For optimal performance, assign a value that is equal to or less than the number of installed or available processors on the PowerCenter Integration Service machine. Valid values are 1 through 64. Default is 0, which disables multithreading.
Array Size	For VSAM data sets or sequential files, if the **Worker Threads** value is greater than zero, the size of the storage array, in number of records, for the threads. For DB2 for z/OS image copies, determines the maximum number of memory buffers that might need to be allocated. If zIIP processing is enabled, **Array Size** also specifies how many compressed rows to dispatch to the zIIP processor. Valid values are 25 through 5000. Default is 25. Informatica recommends that you use the default value unless you are able to test and quantify whether the extra memory that is allocated to a larger array size has been beneficial and has not degraded server performance. If you are able to make these determinations, Informatica recommends an array size of 500 to 1000 when you enable offload multithreaded processing. **Warning:** If you specify a large value, have large records, or run many sessions that use multithreaded processing, you might experience memory shortages on the Integration Service machine.

For DB2 for z/OS tables, configure attributes on the PWX DB2zOS relational connection for the bulk data movement session.

The following table describes the attributes to configure on the PWX DB2zOS relational connection:

Connection Attribute	Description
Offload Processing	Specifies whether to use bulk data offload processing to move column-level processing for DB2 data from the source system to the PowerCenter Integration Service machine. Select one of the following values: - **No**. Do not use offload processing. - **Yes**. Use offload processing. - **Auto**. PowerExchange determines whether to use offload processing. Default is No.
Worker Threads	When offload processing is enabled, specifies the number of threads that PowerExchange uses on the Integration Service machine to process bulk data. For optimal performance, this value should not exceed the number of installed or available processors on the Integration Service machine. Valid values are 1 through 64. Default is 0, which disables multithreading.
Array Size	Specifies the DB2 fetch array size. For more information about DB2 fetch array sizes, see the IBM DB2 documentation. Valid values are 1 through 100000. Default is 25. Informatica recommends that you use the default value unless you are able to test and quantify whether the extra memory that is allocated to a larger array size has been beneficial and has not degraded server performance. If you are able to make these determinations, Informatica recommends an array size of 500 to 1000 when you enable offload and multithreaded processing. **Warning:** If you specify a large value, have large records, or run many sessions that use multithreaded processing, you might experience memory shortages on the Integration Service machine.

Pipeline Partitions in Bulk Data Movement Sessions

You can use pipeline partitioning at reader and writer partition points in bulk data movement sessions to improve performance. With partitioning enabled, the sessions can process data across multiple partitions concurrently.

You can configure reader partitions, writer partitions, or both. For reader partitioning, alternative partitioning schemes are available, which cover some or all data sources. For writer partitioning, you must use pass-through partitioning with VSAM or sequential file targets.

Reader and writer partitioning schemes follow standard PowerCenter pipeline partitioning behavior. If you enable either reader or writer partitioning, PowerCenter uses partitions throughout the entire pipeline. If you change the number of partitions at one stage in the pipeline, PowerCenter updates the partition count at all other stages to match. For more information about pipeline partitioning, see the *PowerCenter Advanced Workflow Guide*.

Reader Partitioning

You can define pass-through or key-range partitions at the source qualifier, or reader, partition point to improve the performance of bulk data movement sessions. With reader partitioning, a session can use one or more partitions to read and process data from the source.

The degree of concurrent processing depends on the data source and partitioning scheme that you use.

- For offloaded DB2 for z/OS unload data sets, VSAM data sets, and sequential files, PowerExchange opens a single connection to the data source and uses multiple partitions to read and process the source data.
- For other data sources, PowerExchange opens multiple connections to the data source or reads the source data into a single partition. If PowerExchange reads data into a single partition, you can redistribute the data at a repartition point to use partitioning in subsequent pipeline stages.

The session might process data out of sequence because partitions process data at varying rates.

The following table summarizes the reader partitioning schemes that you can use for different types of bulk data sources:

Reader Partitioning Scheme	Supported Data Sources	Description
Key range	All relational bulk data sources	Rows from the data source are partitioned based on key range values. This partitioning scheme is recommended for relational data sources.
Pass-through partitioning without SQL overrides	Offloaded DB2 unload data sets, VSAM data sets, and sequential data sets	PowerExchange reads the source data once and automatically distributes the rows among the partitions. PowerExchange ignores the **Worker Threads** connection attribute when you use pass-through partitioning without SQL overrides. This partitioning scheme is recommended for the specified z/OS data sources.
Pass-through partitioning without SQL overrides	All other nonrelational bulk data sources	Data is read into the first partition only. You can use round-robin partitioning at a subsequent partition point to redistribute the data.
Pass-through partitioning with SQL overrides	All	Data for each row is read into a partition based on the SQL override. The partitions run independently of each other and treat each query as an independent PowerExchange request.

Setting Up Pass-Through Partitioning Without SQL Overrides for Data Sources

Complete the following steps to set up pass-through partitioning without SQL overrides if you use the following data sources with offload processing: DB2 for z/OS unload data sets, VSAM data sets, and sequential files.

PowerExchange reads the source data once and automatically distributes the rows across the partitions.

1. In the Workflow Manager, open a PWX application connection by completing the following steps:
   a. Click **Connections** > **Application**.

      The **Application Connection Browser** dialog box appears.
   b. In the **Select Type** field, select **PWX NRDB Batch**.
   c. Select a connection object the **Objects** list and click **Edit**. Or click **New** to create a connection.
2. In the **Connection Object Definition** dialog box, select one of the following options from the **Offload Processing** list:
   - **Filter Before**. PowerExchange offloads column-level processing to the PowerCenter Integration Service machine but continues to filter data on the source system.
   - **Filter After**. PowerExchange offloads column-level processing and data filtering to the PowerCenter Integration Service machine.
3. Click **OK**.
4. In the Task Developer, double-click the session to open the **Edit Tasks** dialog box.
5. On the **Properties** tab, select one of the following options for the **PWX Partition Strategy** attribute:
   - **Single Connection**. PowerExchange creates a single connection to the data source. Any overrides specified for the first partition are used for all partitions. If you specify any overrides for other partitions that are different from the overrides for the first partition, the session fails with an error message.
   - **Overrides Driven**. If the specified overrides are the same for all partitions, PowerExchange creates a single connection to the data source. If the overrides are not identical for all partitions, PowerExchange creates multiple connections.
6. On the **Mapping** tab, click the **Partitions** view.
7. Select the Source Qualifier transformation, and click **Edit Partition Point**.

   The **Edit Partition Point** dialog box appears.
8. Click **Add** for each partition that you want to add.
9. Verify that the **Partition Type** is **Pass Through**.
10. Click **OK**.
11. On the **Mapping** tab, click the **Transformations** view.

    The **Properties** area displays the **SQL Query** attribute for each partition.
12. Verify that the **SQL Query** attribute is empty or contains an identical query for each partition.
13. If the **SQL Query** attribute is not empty and does not contain identical queries, perform the following steps for each partition:
    a. Click the Browse button in the **SQL Query** attribute.
    b. Clear the query in the **SQL Editor** dialog box, or enter the same query for each partition.
    c. Click **OK**.
14. Click **OK**.

Setting Up Pass-Through Partitioning with SQL Overrides for Nonrelational Data Sources

Complete the following steps to set up pass-through partitioning with SQL overrides for nonrelational data sources for which you do not use offload processing.

The partitions run independently of each other and treat each query as an independent PowerExchange request.

1. In the Task Developer, double-click the session to open the session properties.
2. On the **Mapping** tab, click the **Partitions** view.
3. Select the Source Qualifier transformation, and click **Edit Partition Point**.

   The **Edit Partition Point** dialog box appears.
4. Click **Add** for each partition that you want to add.
5. Verify that the **Partition Type** is **Pass Through**.
6. Click **OK**.
7. On the **Mapping** tab, click the **Transformations** view.

   The **Properties** area displays the **SQL Query** attribute for each partition.
8. For each partition, click the Browse button in the **SQL Query** field. Then, in the **SQL Editor** dialog box, enter the query and click **OK**.

   Tip: If you entered a query in Designer when you configured the Source Qualifier transformation, the query appears in the **SQL Query** field for each partition. To override this query, enter another query in the **SQL Editor** dialog box.

Reading Data into the First Reader Partition Only

Complete the following steps to read data only into the first reader partition. This partitioning scheme applies to nonrelational data sources for which you do not use offload processing. You can redistribute rows from the reader partition across subsequent partitions to increase the speed of session processing.

You must use pass-through partitioning without SQL overrides. To redistribute rows from the first reader partition in subsequent partitions, add a round-robin partition point immediately after the reader partition.

1. In the Task Developer, double-click the session to open the **Edit Tasks** dialog box.
2. On the **Mapping** tab, click the **Partitions** view.
3. Select the Source Qualifier transformation, and click **Edit Partition Point**.

   The **Edit Partition Point** dialog box appears.
4. Click **Add** for each partition that you want to add.
5. Verify that the **Partition Type** is **Pass Through**.
6. Click **OK**.
7. On the **Mapping** tab, click the **Transformations** view.

   The **Properties** area displays the **SQL Query** attribute for each partition.
8. Verify that the **SQL Query** attribute is empty or contains an identical query for each partition.
9. If the **SQL Query** attribute is not empty and does not contain an identical query for each partition, perform the following steps for each partition:
   a. In the **SQL Query** field, click the Browse button.
   b. Clear the query in the **SQL Editor** dialog box, or enter the same query for each partition.

c. Click **OK**.

10. On the **Mapping** tab, click the **Partitions** view.
11. To redistribute rows from the first reader partition across subsequent partitions, complete the following substeps:
    a. Click the **Add Partition Point** icon to add a partition point after the Source Qualifier transformation.
    b. Click **Edit Partition Point**.
    c. Select **Round Robin** in the **Partition Type** list, and click **OK**.
12. Click **OK**.

Setting Up Key-Range Partitioning for Relational Data Sources

Complete the following steps to set up key-range partitioning for relational data sources.

1. In the Task Developer, double-click the session to open the session properties.
2. On the **Mapping** tab, click the **Partitions** view.
3. Select the Source Qualifier transformation, and click **Edit Partition Point**.

   The **Edit Partition Point** dialog box appears.
4. Click **Add** for each partition that you want to add.
5. In the **Partition Type** list, select **Key Range**.
6. In the **Edit Partition Key** dialog box, select one or more ports for the key, and click **OK**.
7. For each partition, enter values in the **Start Range** and **End Range** boxes.
8. Click **OK**.

Writer Partitioning

You can define pass-through partitions at the target instance, or writer, partition point to improve the performance of bulk data movement sessions that have VSAM or sequential file targets. With writer partitioning, a session can use multiple partitions to process SQL inserts and write them to targets concurrently.

Use writer partitioning to improve the performance of the bulk data movement session. Writer partitioning is especially beneficial when a session includes heavy transformation processing.

Optionally, you can also use offload processing to move field-level processing from the target system to the PowerCenter Integration Service machine where the session runs. Field-level processing, such as translating data to the target format, can be resource intensive and increase charge-back costs on a target z/OS system. When this processing is offloaded to the PowerCenter Integration Service machine, the processing runs concurrently across multiple writer partitions. To optimize the efficiency of writing data to the target, use both writer partitioning and target-side offload processing.

Writer partitioning uses pass-through partitions and processes SQL insert operations only. If the partitions receive SQL updates or deletes, the session fails with error message PWXPC_12183. If your data contains many updates or deletes, do not use writer partitioning.

Each writer partition uses a single thread. If you enable multithreading, the session continues to run but ignores the **Worker Threads** setting.

You can specify some session properties at the partition level. Based on the **PWX Partition Strategy** setting and whether you specify the same or different session properties across the partitions, PowerExchange uses a single target connection or multiple connections, one for each partition.

If you use a single connection, a joiner thread merges the data from the partitions before sending the data across the network to the target. The joiner does not maintain the order in which it receives data from the partitions. Consequently, the PowerExchange Listener on the target gets data in an order different from the order in which the partitions processed the data.

You must use a PWX NRDB Batch application connection to the target. Use the **Write Mode** connection attribute to control whether the joiner sends data synchronously or asynchronously to the PowerExchange Listener.

If you want to write data to multiple data sets or files, specify the **File Name Override** session property for the writer partitions. Set this property to point to the appropriate target file for each partition. Contention might occur if multiple partitions write to the same target file at the same time.

Rules and Guidelines for Writer Partitions

The following rules and guidelines apply to writer partitioning:

- Writer partitions process SQL inserts only. They do not process SQL updates or deletes.
- You can use writer partitioning for sequential file and VSAM targets, including VSAM ESDS, KSDS, and RRDS data sets.
- Writer partitions must use pass-through partitioning.
- You must use PWX NRDB Batch application connections to the target.
- If resources on the target system are constrained, consider using offload processing in combination with writer partitioning to improve session performance.
- The performance of partitioned sessions depends on the PowerCenter Integration Service machine, network speed, and target system. If a session has high CPU usage, use a PowerCenter Integration Service machine with multiple CPUs. Also ensure that the target system and network speed are fast enough to keep up with partitioned session processing. Otherwise, bottlenecks might occur.
- If you want to use a single target connection and the same session overrides for all partitions, set the **PWX Partition Strategy** session property to **Single Connection**. With this setting, you can specify the session overrides for the first partition only, and the PowerCenter Integration Service uses these overrides for all other writer partitions.
- If you specify a SQL statement in the **Pre SQL** property for a bulk data movement session that uses a PWX NRDB Batch connection and writer partitioning, the session executes the SQL statement in each partition. The session fails if you specify a SQL statement such as CREATEFILE that can run only once for the session. The session executes the statement in the first partition and then fails when trying to execute it again in another partition. Either enter a SQL statement that can run once in each partition, or do not specify the **Pre SQL** attribute for writer partitions.

Session Properties for Writer Partitions

For bulk data movement sessions with writer partitions, you can specify certain session properties at the session level and at the partition level. These properties are optional.

At the session level, you can enter the following session properties for VSAM data sets:

- Insert Only
- Truncate Table Option

For individual partitions, you can enter the following session properties, including overrides, for either VSAM or sequential file targets:

- File Name Override

- Insert SQL Override
- Post SQL
- Pre SQL

Note: For the **Pre SQL** session property, enter SQL statements that can be executed in each partition. If you enter a statement such as CREATEFILE that can be executed only once for the session, the session fails.

For more information about these properties, see *PowerExchange Interfaces for PowerCenter*.

Based on the **PWX Partition Strategy** session property and whether you specify the same or different partition-level session properties across the partitions, the joiner determines whether to use a single target connection or multiple target connections.

- If you select **Single Connection** for the **PWX Partition Strategy** property, the session uses a single connection. If you enter session properties for the first partition only, the PowerCenter Integration Service applies the properties to all other partitions. If you specify different session properties across the partitions, the session fails.
- If you select **Overrides Driven** and enter the same session properties for all partitions, the session uses a single connection to the target. If you enter different session properties for the partitions, the session uses multiple connections, one for each partition.

Configuring Pass-Through Writer Partitioning

Complete the following steps to configure pass-through writer partitioning for bulk data movement sessions that have VSAM or sequential file targets. In these steps, you customize connection attributes, add partitions, and configure session properties for writer partitions.

1. In the Workflow Manager, create or open a PWX NRDB Batch application connection as follows:
   a. Click **Connections** > **Application**.

      The **Application Connection Browser** dialog box appears.

   b. In the **Select Type** list, select **PWX NRDB Batch**.

   c. Select a connection object in the **Objects** list and click **Edit**. Or click **New** to create a connection.
2. In the **Connection Object Definition** dialog box, configure connection attributes.

   The following table describes attributes that affect writer partitioning:

Attribute	Description
Offload Processing	Controls whether offload processing is used. Enter one of the following options: - **No**. Do not use offload processing. - **Auto**. Allow PowerExchange to determine whether to use offload processing. - **Filter After**. Offload the record-level processing to the PowerCenter Integration Service machine but continue to filter data on the target system. - **Filter Before**. Offload the record-level processing and filtering to the PowerCenter Integration Service machine. Default is **No**. For partitioned sessions, select **Filter Before** to optimize performance.
Worker Threads	Controls whether multithreaded processing is used. Enter 0 to disable multithreaded processing. You cannot use multiple worker threads with reader or writer partitions.

Attribute	Description
Array Size	Storage array size, in number of records, that is shared across the partitions. Enter a valid value from 1 through 5000. You might need to adjust the array size to tune the partitioned bulk data movement session. Default is 25.
Write Mode	Controls whether the joiner sends data from the writer partitions synchronously or asynchronously to the target. Select one of the following options: - **Confirm Write On**. The joiner sends data synchronously across the network to the PowerExchange Listener on the target. The joiner waits for confirmation from the PowerExchange Listener before sending more data. If the PowerExchange Listener encounters errors, it returns error messages to the partitions. This option provides better recovery from errors but might slow data transfer. - **Confirm Write Off**. The joiner sends data asynchronously across the network to the target until encountering an end-of-file (EOF) or an error due to an old block of data. When an error occurs, the session might fail with an error message that reports incorrect record information such as the record number, record data, or number of records processed. This option provides faster data transfer but makes recovery from errors more difficult. Select this option only if you can load the target file with data again if an error occurs. Default is **Confirm Write On**.
PWX Override	Enter the following override value to use an alternative error-handling method for errors returned from the PowerExchange Listener: `WRT_ERROR_HANDLING=Y` The alternative method might be more efficient when the **Write Mode** connection attribute is set to **Confirm Write On** and input data for the writer partitions contains many errors. Do not specify this override unless these conditions exist. You can also specify the WRT_ERROR_HANDLING statement in the dbmover.cfg file on the PowerCenter Integration Service machine.

3. Click **OK**. Then click **Close**.
4. In the Task Developer, double-click the session.

   The **Edit Tasks** dialog box appears.
5. On the **Mapping** tab, click the **Partitions** tab to display the partition view.
6. Select the VSAM or sequential file target, and click **Edit Partition Point**.

   The **Edit Partition Point** dialog box appears.
7. To add a partition, click **Add** and select **Pass Through** as the partition type. You can also add a description.
8. After you add all of the partitions, click **OK**.
9. On the **Mapping** tab, click the **Transformation** tab to display the transformation view.

   Verify that the target is selected.
10. In the **Connections Value** field, select a PWX NRDB Batch application connection.
11. Under **Properties**, select one of the following options in the **PWX Partition Strategy** list:
    - **Single Connection**. PowerExchange creates a single connection to the data target. Any overrides specified for the first partition are used for all partitions. If you specify any overrides for other partitions that are different from the overrides for the first partition, the session fails with an error message.

- **Overrides Driven**. If the specified overrides are the same for all partitions, PowerExchange creates a single connection to the data target. If the overrides are not identical for all partitions, PowerExchange creates multiple connections.

12. Set optional override properties at the session level or partition level, based on the target type and property.

    If you selected **Single Connection** for the **PWX Partition Strategy** property, you can enter overrides for the first partition only. The PowerCenter Integration Service then uses them for all partitions in the session.

    The following table describes the session-level and partition-level session properties that pertain to writer partitions:

Property	Description	Target Type	Level
File Name Override	Specifies a fully qualified data set name that overrides the data set name in the PowerExchange data map.	Sequential files and VSAM	Partition level
Insert Only	Processes SQL updates and deletes as inserts. You must select this option when the target does not have keys.	VSAM	Session level
Insert SQL Override	Specifies insert SQL that overrides the default insert SQL sent to PowerExchange.	Sequential files and VSAM	Partition level
Pre SQL	Specifies SQL statements that run before the session with the target database connection.	Sequential files and VSAM	Partition level
Post SQL	Specifies SQL statements that run after the session with the target database connection.	Sequential files and VSAM	Partition level
Truncate Table Option	Truncates the target contents before loading bulk data. **Note:** VSAM data sets must be defined with the REUSE option for truncation to function correctly.	VSAM	Session level

    For more information about all of these properties, see the *PowerExchange Interfaces for PowerCenter Guide*.

13. Click **OK**.

Assessing the Performance of Partitioned Bulk Data Movement Sessions

Use the SHOW_THREAD_PERF statement in the dbmover.cfg file to print messages that contain statistics for assessing the performance of partitioned bulk data movement sessions at user-defined intervals. The messages pertain to sessions with reader or writer partitions.

Enter the SHOW_THREAD_PERF statement in the dbmover.cfg configuration file on the PowerCenter Integration Service machine. The statement defines the number of records that PowerExchange must process before writing statistics messages PWX-31261 and PWX-31262 to the PowerExchange message log. If you specify the **Retrieve PWX Log Entries** attribute on the connection, PWXPC writes these messages in the PowerCenter session log too.

The messages provide the following information:

- PWX-31261. Average, minimum, and maximum amounts of time, in microseconds, that the joiner thread spent performing I/O over a specific number of records.
- PWX-31262. Average, minimum, and maximum I/O delay times, in microseconds, for a partition over a specific number of records. The delay time is the time that the partition waits to receive data. This message is issued for each partition in a session.

Tuning Partitioned Sessions

If partitioned bulk data movement sessions do not run as efficiently as expected, you can adjust tuning parameters to improve performance.

Use the following statements in the DBMOVER configuration file to tune partitioned sessions:

APPBUFSIZE statement

Configure the APPBUFSIZE statement in the DBMOVER configuration file on the PowerCenter Integration Service machine. This statement specifies the maximum buffer size, in bytes, that PowerExchange uses to read or write data. When the buffer reaches this maximum size, PowerExchange sends the buffer data across the network to the requesting system before starting to use another buffer. To help improve session performance, increase the buffer size.

If dynamic application buffer sizing is enabled, the APPBUFSIZE statement defines the initial size of the application data buffer for all connections made during a PowerExchange Listener run. PowerExchange resizes the application data buffer dynamically for individual connections as needed.

APPBUFSIZEDYN statement

Enable dynamic application buffer sizing on the PowerCenter Integration Service machine. For each connections to a data source, if dynamic application buffer sizing is enabled, PowerExchange resizes the application buffer if the data source contains records that are too large to fit into the buffer. Dynamic application buffer sizing is enabled by default. You can explicitly enable it by specifying APPBUFSIZEDYN=Y in the DBMOVER configuration file.

MAXTASKS statement

If processing slows or hangs for bulk data movement sessions that use reader or writer partitioning, increase the MAXTASKS value to help improve performance.

Use the following PowerCenter parameters to tune partitioned sessions:

Array Size connection attribute

Enter this attribute on the PWX Batch NRDB application connection in PowerCenter. This attribute specifies the storage array size, in number of records, that is shared across the partitions in the partitioned bulk data movement session. This attribute also pertains to worker threads for

multithreading. To help improve session performance, increase the array size. Valid values are from 1 through 5000.

Note: If you set the **Write Mode** connection attribute to **Confirm Write On** and the array size is greater than the APPBUFSIZE, PowerExchange sends the buffer across the network each time the number of records in the storage array is:

```
records=array size/partition_count
```

Default buffer block size attribute

Set this session configuration attribute to accommodate the partitioned bulk data movement session. This attribute specifies the default buffer block size that the PowerCenter Integration Service uses to move data and index caches from sources to targets. A low block size can reduce the performance gains from increasing the APPBUFSIZE or **Array Size** value. For example, if the block size accommodates only 10 records, the array size for each partition is limited to 10 records. If you set the **Write Mode** connection attribute to **Confirm Write On**, the APPBUFSIZE is also limited to 10 records.

When setting the block size, use the following criteria:

- If you use **Confirm Write On**, enter a block size that can accommodate the number of records calculated as *array_size/partition_count*. Also, set the APPBUFSIZE to hold the same number of records.
- If you use **Confirm Write Off**, enter a block size that can accommodate the number of records calculated as *array_size/partition_count*.

Also, if you do not use offload processing and resources on the source or target system are constrained, enable offload processing. This feature moves field-level processing of bulk data from the source or target system to the PowerCenter Integration Service machine.

PowerExchange zIIP Exploitation

The IBM System z Integrated Information Processor (zIIP) is designed to help free up general computing capacity and lower the overall total cost of computing for certain data and transaction processing workloads on z/OS. The zIIP can execute any suitable workload provided that software is designed to run the workload on the zIIP.

If you have one or more zIIPs installed, you can configure the PowerExchange Listener on z/OS so that some of its work is offloaded to a zIIP. If multiple PowerExchange Listeners are running, you can configure each of them to offload work to a zIIP.

To be eligible to run on a zIIP, work must meet the following requirements:

- Run in a WorkLoad Manager enclave that is classified as being able to offload to a zIIP, also called a System Utility Processor (SUP)
- Run in an enclave System Request Block (SRB)

Programs that run in an SRB must meet the following requirements:

- Run in Supervisor state, key zero.
- Do not issue SVCs except for SVC 13 (ABEND).
- For subroutines, do not call another subroutine.

DBMOVER Statements for PowerExchange zIIP Exploitation

The following DBMOVER statements control zIIP configuration:

SUP_SSNAME=*subsystem_name*

Defines the subsystem name that identifies the PowerExchange Listener started task to the IBM Workload Manager for offloading work to a zIIP. If your system includes multiple Listeners, you can define a different name for each Listener. Enter a name of up to eight characters.

Default is PWXLSTNR.

SUP_SSTYPE=*subsystem_type*

Defines the name that the IBM Workload Manager uses as the subsystem type for the enclave SRB under which work is dispatched on the zIIP. Enter a name of up to four characters.

Default is PWX.

USESUP={N|Y}

Controls whether PowerExchange offloads zIIP-enabled PowerExchange Listener functions to a zIIP. Specify USESUP=Y to enable offloading to a zIIP.

WORKCLASS

Defines the transaction name for Workload Manager classification. Enter a name of up to eight characters.

Default is PWXWORK.

z/OS System Log Messages for PowerExchange zIIP Exploitation

PowerExchange issues messages to the z/OS system log to report the status of PowerExchange zIIP operations.

The message IDs have the following form:

PWX*mmm*34*xxs*

The string *mmm* represents the calling routine and might be helpful to Informatica Global Customer support if you receive an error message.

xx are the last two digits of the nessage number.

The code *s* is either I for an informational message or E for an error message.

Use these messages to determine whether zIIP configuration was successful, as follows:

- Informational messages indicate successful configuration. The absence of these messages might indicate that prerequisites for zIIP offloading were not satisfied. For more information, see "Configuring PowerExchange to Offload Work to a zIIP" on page 208.
- Error messages indicate an error condition that, in most cases, requires you to call Informatica Global Customer Support.
- Messages PWX*mmm*3412E and PWX*mmm*3414E indicate possible error conditions but might not require you to contact Informatica Global Customer Support if rc = 4.

For more information, See *PowerExchange Message Reference Volume 1*.

Configuring PowerExchange to Offload Work to a zIIP

Before you configure PowerExchange to offload work to a zIIP, verify that the following prerequisites are satisfied:

- The system callable services library SYS1.CSSLIB is available through the LNKLST contatenation or the LPALIB data set.
- The projected usage function (PROJECTCPU) in the IEAOPT*xx* member in the system PARMLIB is enabled. If you enable zIIP usage on a system without a zIIP, and PROJECTCPU is set to FALSE, the system does not project CPU usage as if a zIIP were present, and PowerExchange reports RC = 4 from IWM4EOCT. PowerExchange continues to run zIIP-enabled functions in SRB mode.
- All the libraries in the PowerExchange Listener STEPLIB concatenation are APF-authorised.
- The DBMOVER configuration member does not include any TRACE statements.

1. Include the USESUP=Y statement in the DBMOVER configuration file on z/OS, and optionally include the following statements:
   - SUP_SSNAME
   - SUP_SSTYPE
   - WORKCLASS
2. Add PWX to the IBM Workload Manager for z/OS (WLM):
   a. From the main menu of the WLM ISPF application, add **PWX** as a subsystem type, or specify whatever value you specified for the SUB_SSTYPE statement in the DBMOVER configuration member.
   b. For each PowerExchange Listener, add a work qualfier with a type of SI (system instance) to the list. The name must match the value in the DBMOVER SUP_SSNAME statement (default PWXLSTNR).
   c. Optionally, change the default transaction name by using the TN qualifier type. This value must match the value in the DBMOVER WORKCLASS statement (default PWXWORK).
   d. Check the job log to verify that zIIP enablement is successful.

      If zIIP enablement is successful, the z/OS system log includes informational messages such as the following ones:

```
PWXDSP3400I Checking processors...
PWXDSP3401I Cpu 00 Serial FF04EEC52098 Type CP   Rel. Speed 1
PWXDSP3401I Cpu 01 Serial FF04EEC52098 Type CP   Rel. Speed 1
PWXDSP3401I Cpu 06 Serial FF04EEC52098 Type zIIP Rel. Speed 1
PWXDSP3403I 1 Processor available for zIIP offload
PWXWCO3405I Connect to WLM Sub = PWX  Subn = GRAPWX   token = 140C2118
PWXWCF3409I Classify work to WLM Service class = 00238000
PWXWCE3411I WLM Create Enclave function = PWXFUNC  enclave token = 0000003C00000033
PWXWSE3415I WLM Set Rules tok = PWXR id = IWMOCT   ver = 00 cnt = 01 Dur = 1000000 Pct = 100
DTL-00607 Listener NODE1 VRM 9.5.0 Build DEV_BUILD started.
```

      If the job log does not include messages that indicate that zIIP was successfully enabled, verify that the prerequisites for zIIP enablement are satisfied. If not all libraries in the PowerExchange Listener STEPLIB concatenation are APF-authorized, or if the DBMOVER configuration member includes a TRACE statement, zIIP exploitation is disabled.

Assigning the PowerExchange Listener to a WLM Service Class

The Workload Manager (WLM) is a z/OS component that dynamically manages the prioritization of request for z/OS shared resources such as storage, CPU, and I/O devices across one or more z/OS images based on

service classes that you define. By assigning the PowerExchange Listener started task to the appropriate WLM service class, you can ensure that processes that access your z/OS data through PowerExchange meet the desired performance goals.

You define service classes by using IBM z/OS WLM tools, such as the WLM ISPF application. Each service class includes a goal and importance level, which prioritize work based on your business requirements. Because of the nature of the PowerExchange Listener task, Informatica recommends that you assign the PowerExchange Listener to a service class that has a velocity execution goal.

PowerExchange Listener Subtasks

The PowerExchange Listener provides read and write access to data that is stored in databases and sequential and VSAM files on z/OS. This access can be to read or write large volumes of data or to retrieve a single row. Each access request is handled by a separate subtask that the PowerExchange Listener attaches.

The same Listener can also be used for CDC data access requests. For information about these requests, see the "Tuning CDC Sessions" chapter in the *PowerExchange CDC Guide for z/OS*.

PowerExchange Listener Resource Usage

A PowerExchange Listener that participates in bulk data movement has the following resource usage characteristics:

- CPU usage is moderate. It depends on the volume and complexity of data that the Listener is reading or writing.

  Tip: Use offload processing to reduce the number of CPU cycles that are required to process bulk data on z/OS.
- I/O activity is moderate. It depends on the volume of data that the Listener is reading or writing.
- Virtual memory usage varies, depending on the number of concurrently active Listener subtasks. A separate subtask handles each request to read or write data.

The required performance profile for the PowerExchange Listener depends on the relative importance of the processes that access data by using the PowerExchange Listener compared to other processes running on z/OS. Assign the PowerExchange Listener to a service class that is similar to that of other batch jobs that access z/OS data.

INDEX

E

F

G

I

K

L

M

N

O

P

R

S

T

U

V

W

Z