C

Oracle Messaging Gateway

> **Note:**
>
> Oracle Messaging Gateway is deprecated in 23ai and will be desupported in a future release.

- Introduction to Oracle Messaging Gateway
- Getting Started with Oracle Messaging Gateway
- Working with Oracle Messaging Gateway
- Oracle Messaging Gateway Message Conversion
- Monitoring Oracle Messaging Gateway
- Oracle Messaging Gateway Views

Introduction to Oracle Messaging Gateway

The Messaging Gateway administration package DBMS_MGWADM provides an interface for creating Messaging Gateway agents, managing agents, creating messaging system links, registering non-Oracle queues, and setting up propagation jobs.

Topics:

- Oracle Messaging Gateway Overview
- Oracle Messaging Gateway Features
- Oracle Messaging Gateway Architecture
- Propagation Processing Overview
- Oracle Database AQ Buffered Messages and Messaging Gateway

Oracle Messaging Gateway Overview

Messaging Gateway enables communication between applications based on non-Oracle messaging systems and Oracle Database Advanced Queuing.

Oracle Database Advanced Queuing provides propagation between two Oracle Database Advanced Queuing queues to enable e-business (HTTP through IDAP). Messaging Gateway extends this to applications based on non-Oracle messaging systems.

Because Messaging Gateway is integrated with Oracle Database Advanced Queuing and Oracle Database, it offers reliable message delivery. Messaging Gateway guarantees that messages are delivered once and only once between Oracle Database Advanced Queuing and non-Oracle messaging systems that support persistence. The PL/SQL interface provides an easy-to-learn administrative API, especially for developers already proficient in using Oracle Database Advanced Queuing.

This release of Messaging Gateway supports the integration of Oracle Database Advanced Queuing with applications based on WebSphere MQ 9.0 and TIB/Rendezvous 8.2.

Oracle Messaging Gateway Features

Messaging Gateway provides the following features:

- Extends Oracle Database Advanced Queuing message propagation

  Messaging Gateway propagates messages between Oracle Database Advanced Queuing and non-Oracle messaging systems. Messages sent by Oracle Database Advanced Queuing applications can be received by non-Oracle messaging system applications. Conversely, messages published by non-Oracle messaging system applications can be consumed by Oracle Database Advanced Queuing applications.

- Support for Java Message Service (JMS) messaging systems

  Messaging Gateway propagates messages between Oracle Java Message Service (Oracle JMS) and WebSphere MQ Java Message Service (WebSphere MQ JMS).

- Native message format support

  Messaging Gateway supports the native message formats of messaging systems. Oracle Database Advanced Queuing messages can have `RAW` or any Oracle object type payload. WebSphere MQ messages can be text or byte messages. TIB/Rendezvous messages can be any TIB/Rendezvous wire format datatype except the nested datatype `MSG` and those with unsigned integers.

- Message conversion

  Messaging Gateway facilitates message conversion between Oracle Database Advanced Queuing messages and non-Oracle messaging system messages. Messages are converted through either automatic routines provided by Messaging Gateway or customized message transformation functions that you provide.

  > **Note:**
  >
  > – Messaging Gateway does not support message propagation between JMS and non-JMS messaging systems.
  >
  > – Oracle Database AQ Sharded Queues are not supported by MGW in 12*c* Release 2 (12.2)

- Integration with Oracle Database

  Messaging Gateway is managed through a PL/SQL interface similar to that of Oracle Database Advanced Queuing. Configuration information is stored in Oracle Database tables. Message propagation is carried out by an external process of the Oracle Database server.

- Guaranteed message delivery

  If the messaging systems at the propagation source and propagation destination both support transactions, then Messaging Gateway guarantees that persistent messages are propagated exactly once. If messages are not persistent or transactions are not supported by the messaging systems at the propagation source or propagation destination, then at-most-once propagation is guaranteed.

- Security support

  Messaging Gateway supports client authentication of Oracle Database and non-Oracle messaging systems.

  Messaging Gateway also allows Secure Socket Layer (SSL) support for IBM WebSphere MQ and WebSphere MQ JMS connections made by the Messaging Gateway agent.

- Multiple agent support

  Messaging Gateway supports multiple agents for a given database. Users can partition propagation jobs based on functionality, organizations, or workload and assign them to different Messaging Gateway agents. This allows Messaging Gateway to scale in an Oracle RAC environment and enables propagation job grouping and isolation.

> **See Also:**
>
> - "Propagation Processing Overview"
> - Oracle Messaging Gateway Message Conversion
> - "Converting Oracle Messaging Gateway Non-JMS Messages"

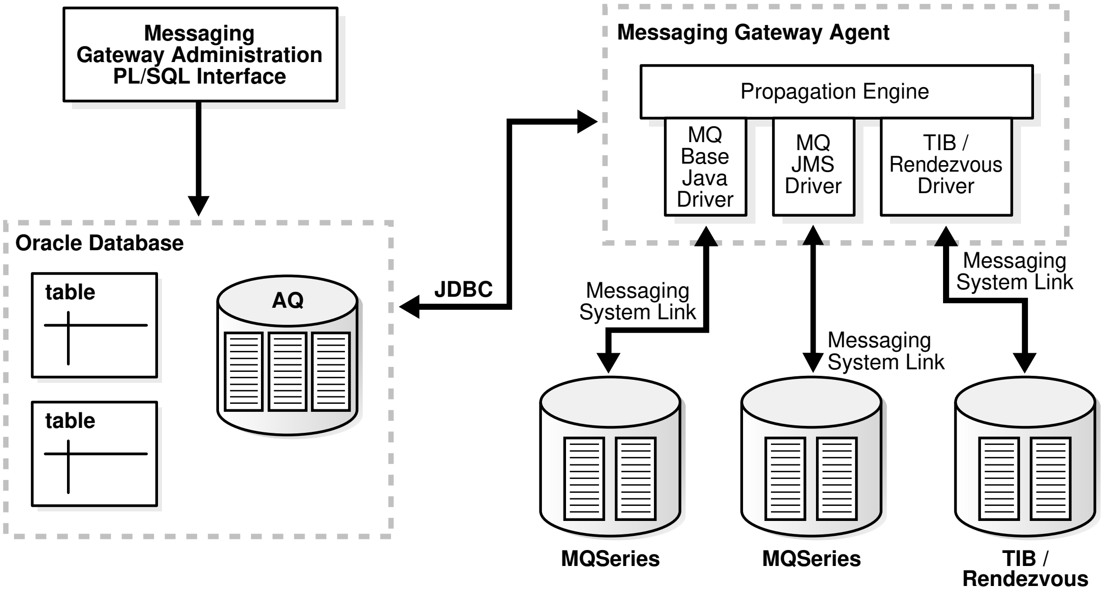
Oracle Messaging Gateway Architecture

Messaging Gateway has two main components:

- Administration Package DBMS_MGWADM
- Messaging Gateway Agent

Figure C-1 shows how these components work together with Oracle Database and non-Oracle messaging systems.

Figure C-1 Messaging Gateway Architecture



Administration Package DBMS_MGWADM

The Messaging Gateway administration package DBMS_MGWADM provides an interface for creating named Messaging Gateway agents, managing agents, creating messaging system links, registering non-Oracle queues, and setting up propagation jobs.

Users call the procedures in the package to make configuration changes regardless of whether the Messaging Gateway agent is running. If the Messaging Gateway agent is running, then the procedures in the package send notifications for configuration changes to the agent. The agent dynamically alters its configuration for most configuration changes, although some changes require that the agent be shut down and restarted before they take effect. All the procedures in the package are serialized to guarantee that the Messaging Gateway agent receives and processes notifications in the same order as they are made.

> **See Also:**
>
> *Oracle Database PL/SQL Packages and Types Reference* for more information on DBMS_MGWADM

Oracle Messaging Gateway Agent

The Messaging Gateway agent runs as an external process of the Oracle Database server and processes propagation jobs. It is started and shut down by calling the STARTUP and SHUTDOWN procedures in DBMS_MGWADM package.

The Messaging Gateway agent contains a multithreaded propagation engine and a set of drivers for messaging systems. The propagation engine fairly schedules propagation jobs and processes propagation jobs concurrently. The polling thread in the agent periodically polls the source queues of enabled propagation jobs and wakes up worker threads to process propagation jobs if messages are available. The drivers for non-Oracle messaging systems run as clients of the messaging systems for all messaging operations.

Oracle Database

As an Oracle Database feature, Messaging Gateway provides a mechanism of message propagation between Oracle Database Advanced Queuing and non-Oracle messaging systems. Oracle Database Advanced Queuing is involved in every propagation job as either propagation source or propagation destination.

Messaging Gateway is managed through the PL/SQL administration package `DBMS_MGWADM`. All configuration information and execution state information of Messaging Gateway are stored in Oracle Database and can be accessed through database views.

The Messaging Gateway agent runs as an external procedure of the Oracle Database server. Therefore, it runs only when its associated database server is running.

Non-Oracle Messaging Systems

The Messaging Gateway agent connects to non-Oracle messaging systems through messaging system links. Messaging system links are communication channels between the Messaging Gateway agent and non-Oracle messaging systems. Users can use the administration package `DBMS_MGWADM` to configure multiple links to the same or different non-Oracle messaging systems.

Queues in non-Oracle messaging systems, such as WebSphere MQ queues, TIB/Rendezvous subjects, and WebSphere MQ JMS destinations (queues and topics) can all serve as propagation sources and destinations for Messaging Gateway. They are referred to as foreign queues. All foreign queues involved in message propagation as source queues, destination queues, or exception queues must be registered through the administration package. The registration of a foreign queue does not create the physical queue in a non-Oracle messaging system, but merely records information about the queue, such as the messaging system link to access it, its native name, and its domain (queue or topic). The physical queue must be created through the administration interface of the non-Oracle messaging system.

> **See Also:**
>
> "Registering a Non-Oracle Queue"

Propagation Processing Overview

Propagation jobs must be defined in order for messages to be propagated from one messaging system to another. A propagation job defines the source queue, destination queue, and various other attributes that affect the processing of the propagation job.

If the propagation source is a queue (point-to-point), then the Messaging Gateway agent moves all messages in the queue to the destination. If the propagation source is a topic (publish/subscribe), then the Messaging Gateway agent creates a subscription on the propagation source topic. The agent moves all messages that are published to the topic after the subscription is created.

A propagation job is processed when it is enabled. Disabling a propagation job stops propagation processing but does not stop message subscription.

When the Messaging Gateway agent processes a propagation job, it dequeues messages from the source queue and enqueues the messages to the destination queue. As each message is propagated, it is converted from its native format in the source messaging system to its native format in the destination messaging system. Messaging Gateway provides automatic message conversions between simple and commonly used message formats. You can customize message conversions by providing your own message transformation functions.

When the Messaging Gateway agent fails to convert a message from the source format to the destination format, the agent moves the message from the source queue to an exception queue, if the exception queue exists, and continues to process the propagation job.

If the Messaging Gateway agent runs into failures when processing a propagation job, it retries up to sixteen times in an exponential backoff scheme (from two seconds up to thirty minutes) before it stops retrying.

To guarantee reliable message delivery, Messaging Gateway requires logging queues in messaging systems that support transactions and persistent messages. The Messaging Gateway agent uses the logging queues to store the processing states of propagation jobs so that it can restore propagation processing from failures.

> **See Also:**
>
> "Configuring Oracle Messaging Gateway Propagation Jobs"

Oracle Database AQ Buffered Messages and Messaging Gateway

Messaging Gateway does not support propagation of buffered messages. In outbound propagation, the Messaging Gateway agent dequeues only persistent messages from AQ queues. In inbound propagation, the Messaging Gateway agent always enqueues persistent messages into AQ queues.

Getting Started with Oracle Messaging Gateway

The following topics describe Oracle Messaging Gateway (MGW) prerequisites and how to load, set up, and unload Messaging Gateway. They also describe how to set up and modify the `mgw.ora` initialization file.

- Oracle Messaging Gateway Prerequisites
- Loading and Setting Up Oracle Messaging Gateway
- Setting Up Non-Oracle Messaging Systems
- Verifying the Oracle Messaging Gateway Setup
- Unloading Oracle Messaging Gateway
- Understanding the mgw.ora Initialization File

Oracle Messaging Gateway Prerequisites

Messaging Gateway uses one Oracle Scheduler job for each Messaging Gateway agent. If the value of the `JOB_QUEUE_PROCESSES` database initialization parameter is zero, then no Oracle

Scheduler jobs will run. If the value is non-zero, it effectively becomes the maximum number of Scheduler jobs and job queue jobs that can run concurrently.

Verify that a non-zero value is set, and that it is large enough to accommodate a Scheduler job for each Messaging Gateway agent to be started.

Loading and Setting Up Oracle Messaging Gateway

Perform the following procedures before running Messaging Gateway:

- Loading Database Objects into the Database
- Modifying listener.ora for the External Procedure
- Modifying tnsnames.ora for the External Procedure
- Setting Up an mgw.ora Initialization File
- Creating an Oracle Messaging Gateway Administrator User
- Creating an Oracle Messaging Gateway Agent User
- Configuring Oracle Messaging Gateway Connection Information
- Configuring Oracle Messaging Gateway in an Oracle RAC Environment

> **Note:**
>
> These setup instructions are specific to 32-bit and 64-bit versions of the Windows and 64-bit versions of the Unix-based operating systems. The tasks apply to both Windows and Unix-based operating systems, except where "Windows Operating System Only" or "Linux Operating System Only" is indicated. For other operating systems, see operating-system specific documentation.

Loading Database Objects into the Database

Using SQL*Plus, run `ORACLE_HOME/mgw/admin/catmgw.sql` as user `SYS` as `SYSDBA`. This script loads the database objects necessary for Messaging Gateway, including roles, tables, views, object types, and PL/SQL packages. It creates public synonyms for Messaging Gateway PL/SQL packages. It creates two roles, `MGW_ADMINISTRATOR_ROLE` and `MGW_AGENT_ROLE`, with certain privileges granted. All objects are owned by `SYS`.

> **Note:**
>
> In a CDB environment, run `ORACLE_HOME/mgw/admin/catmgw.sql` as PDB root.

Modifying listener.ora for the External Procedure

This procedure is for Unix-based operating systems only. Static service information for the listener is not necessary on the Windows operating system.

You must modify `listener.ora` so that the Messaging Gateway PL/SQL packages can call the external procedure.

1. Verify that the default Inter-process Communication (IPC) protocol address for the external procedures is set.

   ```
   LISTENER = (ADDRESS_LIST=
   (ADDRESS=(PROTOCOL=IPC)(KEY=EXTPROC))
   ```

2. Add static service information for the listener in step 1. This involves setting a `SID_DESC` for the listener. Within the `SID_DESC`, the parameters described in Table C-1 are important to Messaging Gateway and must be specified according to your own situation.

Example C-1 Adding Static Service Information for a Listener

```
# Add a SID_DESC
SID_LIST_LISTENER= (SID_LIST=
(SID_DESC =
   (SID_NAME= mgwextproc)
   (ENVS=
     "LD_LIBRARY_PATH=JRE_HOME/lib/amd64:JRE_HOME/lib/amd64/server:ORACLE_HOME/lib")
   (ORACLE_HOME=ORACLE_HOME)
   (PROGRAM = extproc))
```

Table C-1 SID_DESC Parameters

Parameter	Description
`SID_NAME`	The SID that is specified in the net service name in `tnsnames.ora`. In the following example, the `SID_NAME` is `mgwextproc`.

`ENVS`	Set up the `LD_LIBRARY_PATH` environment needed for the external procedure to run. The `LD_LIBRARY_PATH` must contain the following paths:
	`JRE_HOME``/lib/`*`PLATFORM_TYPE`*
	`JRE_HOME``/lib/`*`PLATFORM_TYPE`*`/server`
	`ORACLE_HOME``/lib`
	It should also contain any additional libraries required by third-party messaging systems. See "Setting Up Non-Oracle Messaging Systems".
	The `ENVS` value is `LD_LIBRARY_PATH_64` for Oracle Solaris on SPARC (64-Bit) and Oracle Solaris on x86-64 (64-Bit), `LIBPATH` for IBM AIX on POWER Systems, and `LD_LIBRARY_PATH` for the remaining platforms.

`ORACLE_HOME`	Your Oracle home directory. Using `$ORACLE_HOME` does not work.
`PROGRAM`	The name of the external procedure agent, which is `extproc`

> **Note:**
>
> *`JRE_HOME`* represents the root directory of a JRE installation, just as *`ORACLE_HOME`* represents the root directory of an Oracle installation. Oracle recommends that you use the JRE installed with Oracle Database.

Example C-1 adds `SID_NAME mgwextproc` to a listener.ora file for Linux x86.

Modifying tnsnames.ora for the External Procedure

For the external procedure, configure a net service name `MGW_AGENT` in `tnsnames.ora` whose connect descriptor matches the information configured in `listener.ora`, as shown in Example C-2. The net service name must be `MGW_AGENT` (this value is fixed). The `KEY` value must match the KEY value specified for the IPC protocol in listener.ora. The SID value must match the value specified for SID_NAME of the SID_DESC entry in listener.ora.

> **Note:**
>
> If the names.default_domain parameter for sqlnet.ora has been used to set a default domain, then that domain must be appended to the MGW_AGENT net service name in tnsnames.ora. For example, if sqlnet.ora contains the entry names.default_domain=acme.com, then the net service name in tnsnames.ora must be MGW_AGENT.acme.com.

Example C-2 Configuring MGW_AGENT

```
MGW_AGENT =
(DESCRIPTION=
   (ADDRESS_LIST= (ADDRESS= (PROTOCOL=IPC)(KEY=EXTPROC)))
   (CONNECT_DATA= (SID=mgwextproc)))
```

Setting Up an mgw.ora Initialization File

The Messaging Gateway default initialization file *ORACLE_HOME*/mgw/admin/mgw.ora is a text file. The Messaging Gateway external procedure uses it to get initialization parameters to start the Messaging Gateway agent. Copy *ORACLE_HOME*/mgw/admin/sample_mgw.ora to mgw.ora and modify it according to your situation.

The following procedure sets environment variables and other parameters required for all applications of Messaging Gateway:

1. **Windows Operating System Only:** Set the MGW_PRE_PATH variable. Its value is the path to the jvm.dll library:

   ```
   set MGW_PRE_PATH = JRE_HOME\bin\client
   ```

   This variable is prepended to the path inherited by the Messaging Gateway agent process.

2. Set CLASSPATH to include at least the following:

   - JRE runtime classes:

     JRE_HOME/lib/rt.jar

   - Oracle JDBC classes:

     ORACLE_HOME/jdbc/lib/ojdbc6.jar

   - Oracle internationalization classes:

     ORACLE_HOME/jlib/orai18n.jar

   - SQLJ runtime:

     ORACLE_HOME/sqlj/lib/runtime12.jar

   - Java Message Service (JMS) interface

     ORACLE_HOME/rdbms/jlib/jmscommon.jar

   - Oracle JMS implementation classes

     ORACLE_HOME/rdbms/jlib/aqapi.jar

- Java transaction API

  ORACLE_HOME/jlib/jta.jar
- Any additional classes needed for Messaging Gateway to access non-Oracle messaging systems

> **Note:**
>
> Replace *ORACLE_HOME* with the appropriate, spelled-out value.
> Using $ORACLE_HOME, for example, does not work.
>
> Users of the Windows operating system must set CLASSPATH using the Windows operating system path syntax.

> **See Also:**
>
> "Setting Up Non-Oracle Messaging Systems"

Creating an Oracle Messaging Gateway Administrator User

To perform Messaging Gateway administration work, a database user must be created with MGW_ADMINISTRATOR_ROLE privileges, as shown in Example C-3.

Example C-3 Creating a Messaging Gateway Administrator User

```
CREATE USER admin_user IDENTIFIED BY admin_password;
GRANT CREATE SESSION to admin_user;
GRANT MGW_ADMINISTRATOR_ROLE to admin_user;
```

Creating an Oracle Messaging Gateway Agent User

To establish the Messaging Gateway agent connection back to the database, a database user with MGW_AGENT_ROLE privileges must be created, as shown in Example C-4.

Example C-4 Creating a Messaging Gateway Agent User

```
CREATE USER agent_user IDENTIFIED BY agent_password;
GRANT CREATE SESSION to agent_user;
GRANT MGW_AGENT_ROLE to agent_user;
```

Configuring Oracle Messaging Gateway Connection Information

After the Messaging Gateway agent user is created, the administration user uses DBMS_MGWADM.ALTER_AGENT to configure Messaging Gateway with the user name, password, and database connect string used by the Messaging Gateway agent to connect back to the database, as shown in Example C-5. Use the Messaging Gateway user name and password that you created in "Creating an Oracle Messaging Gateway Agent User". The database connect string parameter can be set to either a net service name in tnsnames.ora (with IPC protocol for better performance) or NULL. If NULL, then the oracle_sid parameter must be set in mgw.ora.

For this release, always specify a not NULL value for the database connect string parameter when calling DBMS_MGWADM.ALTER_AGENT.

Example C-5 Configuring Messaging Gateway Connection Information

```
set echo off
set verify off
connect admin_user

ACCEPT password CHAR PROMPT 'Enter the password for AGENT_USER: ' HIDE

EXEC DBMS_MGWADM.ALTER_AGENT(
      agent_name  => 'default_agent',
      username    => 'agent_user',
      password    => '&password',
      database    => 'agent_database');
```

Configuring Oracle Messaging Gateway in an Oracle RAC Environment

This section contains these topics:

- Configuring Connection Information for the MGW Agent Connections
- Setting the Oracle RAC Instance for the Messaging Gateway Agent

Configuring Connection Information for the MGW Agent Connections

You must make all database connections made by the Messaging Gateway agent to the instance on which the Messaging Gateway agent process is running. This ensures correct failover behavior in an Oracle RAC environment. You can configure connections this way by having the instances use slightly different `tnsnames.ora` files. Each file contains an entry with the same net service name, but the connect data refers to only the instance associated with that `tnsnames.ora` file. The common net service name would then be used for the database parameter when `DBMS_MGWADM.ALTER_AGENT` is used to configure the Messaging Gateway agent database connection information.

For example, in a two-instance Oracle RAC environment with instances `OraDB1` and `OraDB2`, where the net service name `AGENT_DB` is to be used, the `tnsnames.ora` for instance `OraDB1` would look like this:

```
AGENT_DB =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = myhost1.mycorp.com)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = OraDB10.mycorp.com)
      (INSTANCE_NAME = OraDB1)
    )
  )
```

The `tnsnames.ora` for `OraDB2` would look like this:

```
AGENT_DB =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = myhost2.mycorp.com)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = OraDB10.mycorp.com)
      (INSTANCE_NAME = OraDB2)
    )
  )
```

You would then configure Messaging Gateway agent user connection information by running the following command:

```
EXEC DBMS_MGWADM.ALTER_AGENT(
   agent_name    => 'default_agent',
   username      => 'agent_user',
   password      => 'agent_password',
   database      => 'agent_db');
```

Setting the Oracle RAC Instance for the Messaging Gateway Agent

Messaging Gateway provides service affinity for the Messaging Gateway agent external process by leveraging the database service support of the Oracle Scheduler. By default, a Messaging Gateway agent will use the default database service that is mapped to all instances. If you want a Messaging Gateway agent to start on a select group of database instances, you must create a database service for those instances and then assign the database service to the Messaging Gateway agent using the `SERVICE` parameter of the `DBMS_MGWADM.CREATE_AGENT` or `DBMS_MGWADM.ALTER_AGENT` procedures. The `DBMS_MGWADM.STARTUP` procedure submits an Oracle Scheduler job that starts the Messaging Gateway agent external process when the Scheduler job is executed. The Scheduler job will use the database service configured for the Messaging Gateway agent.

The database service specified by the `SERVICE` parameter is only used for the service affinity of the Oracle Scheduler job and thus the service affinity for the Messaging Gateway external process. It is not used for the database connections made by the Messaging Gateway agent user. Those JDBC client connections are based on the values specified for the `DATABASE` and `CONNTYPE` parameters.

See Also:

"Running the Oracle Messaging Gateway Agent on Oracle RAC"

Setting Up Non-Oracle Messaging Systems

This section contains these topics:

- Setting Up for TIB/Rendezvous
- Setting Up for WebSphere MQ Base Java or JMS

Setting Up for TIB/Rendezvous

Running as a TIB/Rendezvous Java client application, the Messaging Gateway agent requires TIB/Rendezvous software to be installed on the computer where the Messaging Gateway agent runs. In this section *`TIBRV_HOME`* refers to the installed TIB/Rendezvous software location.

Modifying listener.ora

On the Linux operating system, `LD_LIBRARY_PATH` in the entry for Messaging Gateway must include *`TIBRV_HOME`*`/lib` for the agent to access TIB/Rendezvous shared library files.

See Also:

"Modifying listener.ora for the External Procedure"

On the Windows operating system, you are not required to modify `listener.ora`. But the system environment variable `PATH` must include *`TIBRV_HOME`*`\bin`.

Modifying mgw.ora

`MGW_PRE_PATH` must include the directory that contains the TIB/Rendezvous license ticket file (`tibrv.tkt`), which usually is located in *`TIBRV_HOME/bin`*.

`CLASSPATH` must include the TIB/Rendezvous jar file *`TIBRV_HOME`*`/lib/tibrvj.jar`. If you use your own customized TIB/Rendezvous advisory message callback, then the location of the callback class must also be included.

You can set the following Java properties to change the default setting:

- `oracle.mgw.tibrv.encoding`
- `oracle.mgw.tibrv.intraProcAdvSubjects`
- `oracle.mgw.tibrv.advMsgCallback`

> **See Also:**
>
> "Understanding the mgw.ora Initialization File"

Example C-6 Setting Java Properties

```
setJavaProp oracle.mgw.tibrv.encoding=ISO8859_1
setJavaProp oracle.mgw.tibrv.intraProcAdvSubjects=_RV.>
setJavaProp oracle.mgw.tibrv.advMsgCallback=MyadvCallback
```

Setting Up for WebSphere MQ Base Java or JMS

The WebSphere MQ client and WebSphere MQ classes for Java and JMS must be installed on the computer where the Messaging Gateway agent runs. In this section *`MQ_HOME`* refers to the location of the installed client. On the Linux operating system, this location is always `/opt/mqm`. On the Windows operating system, the installed location can vary.

Modifying listener.ora

No extra modification of `listener.ora` is necessary for Messaging Gateway to access WebSphere MQ.

Modifying mgw.ora

When using WebSphere MQ Base Java (non-JMS) interface, set `CLASSPATH` to include at least the following (in addition to those in "Setting Up an mgw.ora Initialization File"):

- *`MQ_HOME`*`/java/lib/com.ibm.mq.jar`
- *`MQ_HOME`*`/java/lib/connector.jar`

When using WebSphere MQ JMS interface, set `CLASSPATH` to include at least the following (in addition to those in "Setting Up an mgw.ora Initialization File"):

- *`MQ_HOME`*`/java/lib/com.ibm.mqjms.jar`
- *`MQ_HOME`*`/java/lib/com.ibm.mq.jar`
- *`MQ_HOME`*`/java/lib/connector.jar`

Verifying the Oracle Messaging Gateway Setup

The following procedure verifies the setup and includes a simple startup and shutdown of the Messaging Gateway agent:

1. Start the database listeners.

   Start the listener for the external procedure and other listeners for the regular database connection.

2. Test the database connect string for the Messaging Gateway agent user.

   Run `sqlplus` *`agent_user`*`/`*`agent_password`*`@`*`agent_database`*.

   If it is successful, then the Messaging Gateway agent can connect to the database.

3. **Linux Operating System Only:** Test the net service entry used to call the external procedure.

   Run `sqlplus` *`agent_user`*`/`*`agent_password`*`@MGW_AGENT`.

   This should fail with "ORA-28547: connection to server failed, probable Oracle Net admin error". Any other error indicates that the `tnsnames.ora`, `listener.ora`, or both are not correct.

4. Connect as *`admin_user`* and call `DBMS_MGWADM.STARTUP` to start the Messaging Gateway agent.

5. Using the `MGW_GATEWAY` view, wait for `AGENT_STATUS` to change to `RUNNING` and `AGENT_PING` to change to `REACHABLE`.

6. Connect as *`admin_user`* and call `DBMS_MGWADM.SHUTDOWN` to shut down the Messaging Gateway agent.

7. Using the `MGW_GATEWAY` view, wait for `AGENT_STATUS` to change to `NOT_STARTED`.

Unloading Oracle Messaging Gateway

Use this procedure to unload Messaging Gateway:

1. Shut down Messaging Gateway.

2. Remove any user-created queues whose payload is a Messaging Gateway canonical type (for example, `SYS.MGW_BASIC_MSG_T`).

3. Using SQL*Plus, run *`ORACLE_HOME`*`/mgw/admin/catnomgw.sql` as user `SYS` as `SYSDBA`.

   This drops the database objects used by Messaging Gateway, including roles, tables, views, packages, object types, and synonyms.

4. Remove entries for Messaging Gateway created in `listener.ora` and `tnsnames.ora`.

Understanding the mgw.ora Initialization File

Messaging Gateway reads initialization information from a text file when the Messaging Gateway agent starts. The initialization file contains lines for setting initialization parameters, environment variables, and Java properties. Each entity must be specified on one line. Leading whitespace is trimmed in all cases.

A Messaging Gateway administrator can specify the initialization file to be used for a Messaging Gateway agent via DBMS_MGWADM.CREATE_AGENT and DBMS_MGWADM.ALTER_AGENT. If an initialization file is not specified then a default initialization file will be used.

The default initialization file for the default agent is ORACLE_HOME/mgw/admin/mgw.ora.

The default initialization file for a named agent is ORACLE_HOME/mgw/admin/mgw_AGENTNAME.ora where AGENTNAME is the name in uppercase of the Messaging Gateway agent. For example, if the agent name is my_agent then the name of the agent's default initialization file is ORACLE_HOME/mgw/admin/mgw_MY_AGENT.ora. If the default initialization file for a named agent is not found then ORACLE_HOME/mgw/admin/mgw.ora will be used.

mgw.ora Initialization Parameters

The initialization parameters are typically specified by lines having a "*name*=*value*<NL>" format where *name* represents the parameter name, *value* represents its value and <NL> represents a new line.

log_directory

Usage:

Specifies the directory where the Messaging Gateway log/trace file is created.

Format:

```
log_directory = value
```

Default:

ORACLE_HOME/mgw/log

Example:

```
log_directory = /private/mgwlog
```

log_level

Usage:

Specifies the level of logging detail recorded by the Messaging Gateway agent. The logging level can be dynamically changed by calling DBMS_MGWADM.SET_LOG_LEVEL while the Messaging Gateway agent is running. Oracle recommends that log level 0 (the default value) be used at all times.

Format:

```
log_level = value
```

Values:

0 for basic logging; equivalent to DBMS_MGWADM.BASIC_LOGGING

1 for light tracing; equivalent to DBMS_MGWADM.TRACE_LITE_LOGGING

2 for high tracing; equivalent to DBMS_MGWADM.TRACE_HIGH_LOGGING

3 for debug tracing; equivalent to DBMS_MGWADM.TRACE_DEBUG_LOGGING

`4` for high debug tracing; equivalent to `DBMS_MGWADM.TRACE_DEBUG_HIGH_LOGGING`

Example:

```
log_level = 0
```

mgw.ora Environment Variables

Because the Messaging Gateway process environment is not under the direct control of the user, certain environment variables should be set using the initialization file. The environment variables currently used by the Messaging Gateway agent are `CLASSPATH`, `MGW_PRE_PATH`, and `ORACLE_SID`.

Environment variables such as `CLASSPATH` and `MGW_PRE_PATH` are set so the Messaging Gateway agent can find the required shared objects, Java classes, and so on. Environment variables are specified by lines having a "`set` *`env_var`*`=value<NL>`" or "`setenv` *`env_var`*`=`*`value`*`<NL>`" format where *`env_var`* represents the name of the environment variable to set, *`value`* represents the value of the environment variable, and `<NL>` represents a new line.

CLASSPATH

Usage:

Used by the Java Virtual Machine to find Java classes needed by the Messaging Gateway agent for propagation between Oracle Database Advanced Queuing and non-Oracle messaging systems.

Format:

```
set CLASSPATH=value
```

Example:

```
set CLASSPATH=ORACLE_HOME/jdbc/lib/ojdbc6.jar:JRE_HOME/lib/rt.jar:
ORACLE_HOME/sqlj/lib/runtime12.jar:ORACLE_HOME/jlib/orai18n.jar:ORACLE_HOME/rdbms/
jlib/jmscommon.jar:ORACLE_HOME/rdbms/jlib/aqapi.jar:ORACLE_HOME/jlib/jta.jar:
/opt/mqm/java/lib/com.ibm.mq.jar:/opt/mqm/java/lib/com.ibm.mqjms.jar:/opt/mqm/java
/lib/connector.jar
```

MGW_PRE_PATH

Usage:

Appended to the front of the path inherited by the Messaging Gateway process. For the Windows operating system, this variable must be set to indicate where the library `jvm.dll` is found.

Format:

```
set MGW_PRE_PATH=value
```

Example:

```
set MGW_PRE_PATH=JRE_HOME\bin\client
```

ORACLE_SID

Usage:

Can be used when a service name is not specified when configuring Messaging Gateway.

Format:

```
set ORACLE_SID=value
```

Example:

```
set ORACLE_SID=my_sid
```

mgw.ora Java Properties

You must specify Java system properties for the Messaging Gateway JVM when working with TIB/Rendezvous subjects. You can use the `setJavaProp` parameter of the Messaging Gateway initialization file for this. Java properties are specified by lines having a "`setJavaProp` *`prop_name`*`=`*`value`*`<NL>`" format, where *`prop_name`* represents the name of the Java property to set, *`value`* represents the value of the Java property, and `<NL>` represents a new line character.

oracle.mgw.batch_size

Usage:

This Java property represents the maximum number of messages propagated in one transaction. It serves as a default value if the Messaging Gateway job option, `MsgBatchSize`, is not specified. If altered from the default, then consideration should be given to the expected message size and the Messaging Gateway agent memory (see `max_memory` parameter of `DBMS_MGWADM.ALTER_AGENT`). The minimum value of this Java property is 1, the maximum is 100, and the default is 30.

> **See Also:**
>
> "DBMS_MGWADM" in *Oracle Database PL/SQL Packages and Types Reference*

Syntax:

```
setJavaProp oracle.mgw.batch_size=value
```

Example:

```
setJavaProp oracle.mgw.batch_size=10
```

oracle.mgw.polling_interval

Usage:

This parameter specifies the time (in milliseconds) that must elapse between polls for available messages of a propagation source queue. The default polling interval used by Messaging Gateway is 5000 milliseconds (5 seconds). The minimum value of this Java property is 100 millisecond, and the maximum value is 60 seconds.

Syntax:

```
setJavaProp oracle.mgw.polling_interval=value
```

Example:

```
setJavaProp oracle.mgw.polling_interval=1000
```

oracle.mgw.tibrv.encoding

Usage:

This parameter specifies the character encoding to be used by the TIB/Rendezvous messaging system links. Only one character set for all configured TIB/Rendezvous links is allowed due to TIB/Rendezvous restrictions. The default is ISO 8859-1 or the character set specified by the Java system property `file.encoding`.

Syntax:

```
setJavaProp oracle.mgw.tibrv.encoding=value
```

Example:

```
setJavaProp oracle.mgw.tibrv.encoding=ISO8859_1
```

oracle.mgw.tibrv.intraProcAdvSubjects

Usage

Used for all TIB/Rendezvous messaging system links, this parameter specifies the names of system advisory subjects that present on the intraprocess transport.

Syntax

```
setJavaProp oracle.mgw.tibrv.intraProcAdvSubjects=
advisorySubjectName[:advisorySubjectName]
```

Example:

```
setJavaProp oracle.mgw.tibrv.intraProcAdvSubjects=_RV.>
```

oracle.mgw.tibrv.advMsgCallback

Usage:

Used for all TIB/Rendezvous messaging system links, this parameter specifies the name of the Java class that implements the `TibrvMsgCallback` interface to handle system advisory messages. If it is not specified, then the default system advisory message handler provided by Messaging Gateway is used, which writes system advisory messages into Messaging Gateway log files. If it is specified, then the directory where the class file is stored must be included in the `CLASSPATH` in `mgw.ora`.

Syntax:

```
setJavaProp oracle.mgw.tibrv.advMsgCallback=className
```

Example:

```
setJavaProp oracle.mgw.tibrv.advMsgCallback=MyAdvCallback
```

oracle.net.tns_admin

Usage:

This parameter specifies the directory of the tnsnames.ora file. It must be set if the Messaging Gateway agent is configured to use the JDBC Thin driver and the database specifier of the agent connection information is a TNSNames alias. This does not need to be set if the JDBC OCI driver is used or the database specifier is something other than a TNSNames alias.

Syntax:

```
setJavaProp oracle.net.tns_admin=value
```

Example:

```
setJavaProp oracle.net.tns_admin=/myoraclehome/network/admin
```

mgw.ora Comment Lines

Comment lines are designated with a `#` character as the first character of the line.

Working with Oracle Messaging Gateway

After Oracle Messaging Gateway (MGW) is loaded and set up, it is ready to be configured and run. You can use `DBMS_MGWADM.ALTER_AGENT` to set the user name, password, database specifier, and connection type the Messaging Gateway agent will use for creating database connections.

Topics:

- Configuring the Oracle Messaging Gateway Agent
- Starting and Shutting Down the Oracle Messaging Gateway Agent
- Configuring Messaging System Links
- Configuring Non-Oracle Messaging System Queues
- Configuring Oracle Messaging Gateway Propagation Jobs
- Propagation Jobs_ Subscribers_ and Schedules
- Configuration Properties

> **Note:**
>
> All commands in the examples must be run as a user granted `MGW_ADMINISTRATOR_ROLE`.

> **See Also:**
>
> "DBMS_MGWADM" and "DBMS_MGWMSG" in *Oracle Database PL/SQL Packages and Types Reference*

Configuring the Oracle Messaging Gateway Agent

Messages are propagated between Oracle Database Advanced Queuing and non-Oracle messaging systems by the Messaging Gateway agent. The Messaging Gateway agent runs as an external process of the Oracle Database server.

Messaging Gateway supports multiple agents for a given database. A default agent is automatically created that has the name of DEFAULT_AGENT. Additional named agents can be created to provide propagation job isolation and grouping, and scaling in an Oracle RAC environment. The default agent is usually sufficient for single instance, non-Oracle RAC, environments.

This section contains these topics:

- Creating a Messaging Gateway Agent
- Removing a Messaging Gateway Agent
- Setting Database Connection
- Setting the Resource Limits

Creating a Messaging Gateway Agent

You can use DBMS_MGWADM.CREATE_AGENT to create additional Messaging Gateway agents. The Messaging Gateway default agent, DEFAULT_AGENT, is automatically created when Messaging Gateway is installed and will always exist.

Agents can be configured with an agent user, connection information, database service, and resource limits when the agent is created, or at a later time using DBMS_MGWADM.ALTER_AGENT. A Messaging Gateway agent must be configured with a database user that has been granted the role MGW_AGENT_ROLE before the agent can be started.

Example C-7 creates the agent named myagent and specifies the database connection information for the agent user. Default values are used for all other parameters.

Example C-7 Creating a Messaging Gateway Agent

```
SQL> exec DBMS_MGWADM.CREATE_AGENT(
                         agent_name  => 'myagent',
                         username    => 'mgwagent',
                         password    => 'mgwagent_password',
                         database    => 'mydatabase');
```

Removing a Messaging Gateway Agent

A Messaging Gateway agent can be removed by calling DBMS_MGWADM.REMOVE_AGENT. Before an agent can be removed, all Messaging Gateway links associated with the agent must be removed and the agent shut down. The default agent, DEFAULT_AGENT, cannot be removed. Example C-8 removes the agent named myagent.

Example C-8 Removing a Messaging Gateway Agent

```
SQL> exec DBMS_MGWADM.REMOVE_AGENT(agent_name   => 'myagent');
```

Setting Database Connection

The Messaging Gateway agent runs as a process external to the database. To access Oracle Database Advanced Queuing and the Messaging Gateway packages, the Messaging Gateway agent needs to establish connections to the database. You can use DBMS_MGWADM.ALTER_AGENT to set the user name, password and the database connect string that the Messaging Gateway agent will use for creating database connections. The user must be granted the role MGW_AGENT_ROLE before the Messaging Gateway agent can be started.

Example C-9 shows the Messaging Gateway default agent being configured for user mgwagent with password mgwagent_password using net service name mydatabase.

Example C-9 Setting Database Connection Information

```
SQL> exec DBMS_MGWADM.ALTER_AGENT (
                             agent_name => 'default_agent',
                             username   => 'mgwagent',
                             password   => 'mgwagent_password',
                             database   => 'mydatabase');
```

Setting the Resource Limits

You can use DBMS_MGWADM.ALTER_AGENT to set resource limits for the Messaging Gateway agent. For example, you can set the heap size of the Messaging Gateway agent process and the number of propagation threads used by the agent process. The default values are 64 MB of memory heap and one propagation thread. For named agents, these values can also be specified when the agent is created by DBMS_MGWADM.CREATE_AGENT.

Example C-10 sets the heap size to 96 MB and two propagation threads for the agent myagent.

The memory heap size and the number of propagation threads cannot be altered when the Messaging Gateway agent is running.

Example C-10 Setting the Resource Limits

```
SQL> exec DBMS_MGWADM.ALTER_AGENT(
                        agent_name  => 'myagent',
                        max_memory  => 96,
                        max_threads => 2);
```

Starting and Shutting Down the Oracle Messaging Gateway Agent

This section contains these topics:

- Starting the Oracle Messaging Gateway Agent
- Shutting Down the Oracle Messaging Gateway Agent
- Oracle Messaging Gateway Agent Scheduler Job
- Running the Oracle Messaging Gateway Agent on Oracle RAC

Starting the Oracle Messaging Gateway Agent

After the Messaging Gateway agent is configured, you can start the agent with DBMS_MGWADM.STARTUP. Example C-11 shows how to start the default agent and agent myagent.

You can use the MGW_GATEWAY view to check the status of the Messaging Gateway agent, as described in Monitoring Oracle Messaging Gateway.

Example C-11 Starting the Messaging Gateway Agent

```
SQL> exec DBMS_MGWADM.STARTUP;
SQL> exec DBMS_MGWADM.STARTUP ('myagent');
```

Shutting Down the Oracle Messaging Gateway Agent

You can use DBMS_MGWADM.SHUTDOWN to shut down the Messaging Gateway agent. Example C-12 shows how to shut down the Messaging Gateway default agent and agent myagent.

You can use the MGW_GATEWAY view to check if the Messaging Gateway agent has shut down successfully, as described in Monitoring Oracle Messaging Gateway.

Example C-12 Shutting Down the Messaging Gateway Agent

```
SQL> exec DBMS_MGWADM.SHUTDOWN;
SQL> exec DBMS_MGWADM.SHUTDOWN ('myagent');
```

Oracle Messaging Gateway Agent Scheduler Job

Messaging Gateway uses a Scheduler job to start the Messaging Gateway agent. This job is created when procedure DBMS_MGWADM.STARTUP is called. When the job is run, it calls an external procedure that creates the Messaging Gateway agent in an external process. The job is removed after:

- The agent shuts down because DBMS_MGWADM.SHUTDOWN was called
- The agent terminates because a non-restartable error occurs

Messaging Gateway uses DBMS_SCHEDULER to create a repeatable Scheduler job with a repeat interval of one minute. The job is owned by SYS. A repeatable job enables the Messaging Gateway agent to restart automatically when a given job instance ends because of a database shutdown, database malfunction, or a restartable error. Only one instance of a Messaging Gateway agent job runs at a given time.

Each agent uses a Scheduler job class to specify the service affinity for the agent's Scheduler job. The job class will be configured with the database service specified by DBMS_MGWADM.CREATE_AGENT or DBMS_MGWADM.ALTER_AGENT. A database administrator is responsible for setting up the database service. If no database service is specified, the default database service that maps to every instance is used.

The name of the Scheduler job class used by the Messaging Gateway default agent is SYS.MGW_JOBCLS_DEFAULT_AGENT. The Scheduler job used by the default agent is SYS.MGW_JOB_DEFAULT_AGENT.

The name of the Scheduler job class used by a Messaging Gateway named agent is SYS.MGW_JOBCLS_<agent_name>. The Scheduler job used by a named agent is SYS.MGW_JOB_<agent_name>.

If the agent job encounters an error, then the error is classified as either a restartable error or non-restartable error. A restartable error indicates a problem that might go away if the agent job were to be restarted. A non-restartable error indicates a problem that is likely to persist and be encountered again if the agent job restarts. ORA-01089 (immediate shutdown in progress) and ORA-28576 (lost RPC connection to external procedure) are examples of restartable errors. ORA-06520 (error loading external library) is an example of a non-restartable error.

Messaging Gateway uses a database shutdown trigger. If the Messaging Gateway agent is running on the instance being shut down, then the trigger notifies the agent of the shutdown, and upon receipt of the notification, the agent will terminate the current run. The job scheduler will automatically schedule the job to run again at a future time.

If a Messaging Gateway agent job instance ends because of a database malfunction or a restartable error detected by the agent job, then the job will not be removed and the job scheduler will automatically schedule the job to run again at a future time.

The `MGW_GATEWAY` view shows the agent status, database service, and the database instance on which the Messaging Gateway agent is current running. The Oracle Scheduler views provide information about Scheduler jobs, job classes, and job run details.

See Also:

- "DBMS_SCHEDULER" in *Oracle Database PL/SQL Packages and Types Reference*
- Monitoring Oracle Messaging Gateway

Running the Oracle Messaging Gateway Agent on Oracle RAC

While the Messaging Gateway job startup and shutdown principles are the same for Oracle Real Application Clusters (Oracle RAC) and non-Oracle RAC environments, there are some things to keep in mind for an Oracle RAC environment.

A single process of each configured Messaging Gateway agent can be running, even in an Oracle RAC environment. For example, if the default agent and two named agents have been configured with an agent user, then one instance of all three agents could be running at the same time. The database service associated with each agent determines the service affinity of the agent's Scheduler job, and thus, the database instance on which the agent process can run.

When a database instance is shut down in an Oracle RAC environment, the Messaging Gateway shutdown trigger will notify the agent to shut down only if the Messaging Gateway agent is running on the instance being shut down. The job scheduler will automatically schedule the job to be run again at a future time, either on another instance, or if the job can only run on the instance being shut down, when that instance is restarted.

Oracle recommends that all database connections made by the Messaging Gateway agent be made to the instance on which the Messaging Gateway agent process is running. This ensures correct failover behavior in an Oracle RAC environment.

If a Messaging Gateway agent has been associated with a database service, the agent's Scheduler job will not run unless that service is current enabled on a running instance. When you shut down a database Oracle stops all services to that database and you may need to manually restart the services when you start the database.

See Also:

- "Configuring Oracle Messaging Gateway in an Oracle RAC Environment"
- "`DBMS_MGWADM`" and "`DBMS_SCHEDULER`" in *Oracle Database PL/SQL Packages and Types Reference*

Configuring Messaging System Links

Running as a client of non-Oracle messaging systems, the Messaging Gateway agent communicates with non-Oracle messaging systems through messaging system links. A messaging system link is a set of connections between the Messaging Gateway agent and a non-Oracle messaging system.

To configure a messaging system link of a non-Oracle messaging system, users must provide information for the agent to make connections to the non-Oracle messaging system. Users can specify the maximum number of messaging connections.

An agent name will be associated with each messaging system link. This is done when the link is created and cannot be changed. The agent associated with the link is then responsible for processing all propagation jobs that use a registered queue associated with that link. The Messaging Gateway default agent will be used if an agent name is not specified when the messaging system link is created.

When configuring a messaging system link for a non-Oracle messaging system that supports transactions and persistent messages, the native name of log queues for inbound and outbound propagation must be specified in order to guarantee exactly-once message delivery. The log queues should be used only by the Messaging Gateway agent. No other programs should enqueue or dequeue messages of the log queues. The inbound log queue and outbound log queue can refer to the same physical queue, but better performance can be achieved if they refer to different physical queues.

One and only one Messaging Gateway agent should access a propagation log queue. This insures that a given log queue contains log records for only those propagation jobs processed by that agent and that the agent is free to discard any other log records it might encounter.

When configuring a messaging system link, users can also specify an `options` argument. An `options` argument is a set of {name, value} pairs of type `SYS.MGW_PROPERTY`.

This section contains these topics:

- Creating a WebSphere MQ Base Java Link
- Creating a WebSphere MQ JMS Link
- Creating a WebSphere MQ Link to Use SSL
- Creating a TIB/Rendezvous Link
- Altering a Messaging System Link
- Removing a Messaging System Link
- Views for Messaging System Links

Creating a WebSphere MQ Base Java Link

A WebSphere MQ Base Java link is created by calling `DBMS_MGWADM.CREATE_MSGSYSTEM_LINK` with the following information provided:

- Interface type: `DBMS_MGWADM.MQSERIES_BASE_JAVA_INTERFACE`
- WebSphere MQ connection information:
  - Host name and port number of the WebSphere MQ server
  - Queue manager name
  - Channel name
  - User name and password
- Maximum number of messaging connections allowed
- Log queue names for inbound and outbound propagation
- Optional information such as:
  - Send, receive, and security exits
  - Character sets

Example C-13 configures a WebSphere MQ Base Java link `mqlink`. The link is configured to use the WebSphere MQ queue manager `my.queue.manager` on host `myhost.mydomain` and port 1414, using WebSphere MQ channel `mychannel`.

This example also sets the option to register a WebSphere MQ `SendExit` class. The class `mySendExit` must be in the `CLASSPATH` set in `mgw.ora`. The Messaging Gateway default agent (`DEFAULT_AGENT`) is responsible for the link and all propagation jobs using the link.

See Also:

- "Understanding the mgw.ora Initialization File" for information on setting the CLASSPATH of the Messaging Gateway agent
- "WebSphere MQ System Properties"

Example C-13   Configuring a WebSphere MQ Base Java Link

```
DECLARE
  v_options sys.mgw_properties;
  v_prop sys.mgw_mqseries_properties;
BEGIN
  v_prop := sys.mgw_mqseries_properties.construct();

  v_prop.interface_type := dbms_mgwadm.MQSERIES_BASE_JAVA_INTERFACE;
  v_prop.max_connections := 1;
  v_prop.username := 'mqm';
  v_prop.password := 'mqm';
  v_prop.hostname := 'myhost.mydomain';
  v_prop.port     := 1414;
  v_prop.channel  := 'mychannel';
  v_prop.queue_manager := 'my.queue.manager';
  v_prop.outbound_log_queue := 'mylogq';
```

```
    -- Specify a WebSphere MQ send exit class 'mySendExit' to be associated with
    -- the queue.
    -- Note that this is used as an example of how to use the options parameter,
    -- but is not an option that is usually set.
    v_options := sys.mgw_properties(sys.mgw_property('MQ_SendExit',
                                                     'mySendExit'));
    dbms_mgwadm.create_msgsystem_link(
        linkname => 'mqlink', agent_name=>'default_agent', properties => v_prop,
        options => v_options );
END;
```

Creating a WebSphere MQ JMS Link

A WebSphere MQ JMS link is created by calling `DBMS_MGWADM.CREATE_MSGSYSTEM_LINK` with the following information provided:

- Interface type

  Java Message Service (JMS) distinguishes between queue and topic connections. The Sun Microsystem JMS 1.1 standard supports domain unification that allows both JMS queues and topics to be accessed by a single JMS connection:

  - A WebSphere MQ JMS link created with interface type `DBMS_MGWADM.JMS_CONNECTION` can be used to access both JMS queues and topics. This is the recommended interface for a WebSphere MQ JMS link.
  - A WebSphere MQ JMS link created with interface type `DBMS_MGWADM.JMS_QUEUE_CONNECTION` can be used to access only JMS queues.
  - A WebSphere MQ JMS link created with interface type `DBMS_MGWADM.JMS_TOPIC_CONNECTION` can be used to access only JMS topics.
- WebSphere MQ connection information:
  - Host name and port number of the WebSphere MQ server
  - Queue manager name
  - Channel name
  - User name and password
- Maximum number of messaging connections allowed

  A messaging connection is mapped to a JMS session.
- Log destination (JMS queue or JMS topic) for inbound and outbound propagation

  The log destination type must be valid for the link type. JMS unified links and JMS queue links must use JMS queues for log destinations, and JMS topic links must use topics:

  - For a WebSphere MQ JMS unified or queue link, the log queue name must be the name of a physical WebSphere MQ JMS queue created using WebSphere MQ administration tools.
  - For a WebSphere MQ JMS topic link, the log topic name must be the name of a WebSphere MQ JMS topic. The physical WebSphere MQ queue used by that topic must be created using WebSphere MQ administration tools. By default, the physical queue used is `SYSTEM.JMS.D.SUBSCRIBER.QUEUE`. A link option can be used to specify a different physical queue.
- Optional information such as:
  - Send, receive, and security exits
  - Character sets

  – WebSphere MQ publish/subscribe configuration used for JMS topics

Example C-14 configures a Messaging Gateway link to a WebSphere MQ queue manager using a JMS topic interface. The link is named `mqjmslink` and is configured to use the WebSphere MQ queue manager `my.queue.manager` on host `myhost.mydomain` and port 1414, using WebSphere MQ channel `mychannel`.

This example also uses the `options` parameter to specify a nondefault durable subscriber queue to be used with the log topic. The Messaging Gateway agent `myagent` is responsible for the link and all propagation jobs using the link.

> **See Also:**
>
> - "Registering a WebSphere MQ JMS Queue or Topic" for more information on JMS queues and topics
> - "WebSphere MQ System Properties"

Example C-14   Configuring a WebSphere MQ JMS Link

```
DECLARE
  v_options sys.mgw_properties;
  v_prop sys.mgw_mqseries_properties;
BEGIN
  v_prop := sys.mgw_mqseries_properties.construct();
  v_prop.max_connections := 1;

  v_prop.interface_type := DBMS_MGWADM.JMS_TOPIC_CONNECTION;
  v_prop.username := 'mqm';
  v_prop.password := 'mqm';
  v_prop.hostname := 'myhost.mydomain';
  v_prop.port     := 1414;
  v_prop.channel  := 'mychannel';
  v_prop.queue_manager := 'my.queue.manager';

  v_prop.outbound_log_queue := 'mylogtopic'

  -- Specify a WebSphere MQ durable subscriber queue to be used with the
  -- log topic.
  v_options := sys.mgw_properties(
  sys.mgw_property('MQ_JMSDurSubQueue', 'myDSQueue'));

  DBMS_MGWADM.CREATE_MSGSYSTEM_LINK(
      linkname     =>  'mqjmslink',
      agent_name   =>  'myagent',
      properties   =>   v_prop,
      options      =>   v_options );
END;
```

Creating a WebSphere MQ Link to Use SSL

Messaging Gateway allows SSL support for IBM WebSphere MQ and WebSphere MQ JMS connections. This section describes how to configure Messaging Gateway to use SSL for a WebSphere MQ Base Java link and the same information applies to a WebSphere MQ JMS link. There are no differences in terms of the Messaging Gateway configuration.

The following are needed in order to use SSL for WebSphere MQ connections:

- A WebSphere MQ channel configured to use SSL.
- A truststore and optionally a keystore file that are in a location accessible to the Messaging Gateway agent process. In an Oracle RAC environment, these files must be accessible to all instances on which the Messaging Gateway agent process might run, using the same path specification.
- Use `DBMS_MGWADM.CREATE_MSGSYSTEM_LINK` to create a WebSphere MQ link with the desired SSL related link options. At minimum, the `MQ_SSLCIPHERSUITE` property should be set to specify the SSL ciphersuite used by the channel.
- Use `DBMS_MGWADM.SET_OPTION` to set certain JSSE Java properties for the Messaging Gateway agent assigned to the link.

JSEE related properties:

- `java.net.ssl.keyStore`

  This property is used to specify the location of the keystore. A keystore is a database of key material used for various purposes, including authentication and data integrity.
- `java.net.ssl.keyStorePassword`

  This property is used to specify the password of the keystore. This password is used to check the integrity of the data in the keystore before accessing it.
- `java.net.ssl.trustStore`

  This property is used to specify the location of the truststore. A truststore is a keystore that is used when making decisions about which clients and servers are trusted.
- `java.net.ssl.trustStorePassword`

  This property is used to specify the password of the truststore. This password is used to check the integrity of the data in the truststore before accessing it.

The `java.net.ssl.keyStore` and `java.net.ssl.keyStorePassword` properties are only needed if the WebSphere MQ channel is configured to use SSL client authentication.

Example C-15 configures a WebSphere MQ Base Java link mqssllink to use SSL connections using the `SSL_RSA_WITH_RC4_128_MD5` ciphersuite. It assumes the channel has been configured for SSL client authentication so the Messaging Gateway agent associated with the link, `DEFAULT_AGENT`, is configured with Java properties for both a keystore and a truststore.

This configuration should be done when the Messaging Gateway agent is shut down since the Java properties set by `DBMS_MGWADM.SET_OPTION` are set only when the agent first starts. If the agent is running when the configuration is done it will need to be shutdown and restarted before the SSL connections will be used.

> **Note:**
>
> "WebSphere MQ System Properties"

Example C-15 Configuring a WebSphere MQ Base Java Link for SSL

```
DECLARE
  v_options sys.mgw_properties;
  v_prop sys.mgw_mqseries_properties;
  v_agent varchar2(30) := 'default_agent';
BEGIN
     v_prop := sys.mgw_mqseries_properties.construct();
     v_prop.interface_type := DBMS_MGWADM.MQSERIES_BASE_JAVA_INTERFACE;
     v_prop.max_connections := 1;
     v_prop.username := 'mqm';
     v_prop.password := 'mqm';
     v_prop.hostname := 'myhost.mydomain';
     v_prop.port := 1414;
     v_prop.channel := 'mysslchannel';
     v_prop.queue_manager := 'my.queue.manager';
     v_prop.outbound_log_queue := 'mylogq';

   -- specify the SSL ciphersuite
     v_options := sys.mgw_properties(
        sys.mgw_property('MQ_SSLCIPHERSUITE','SSL_RSA_WITH_RC4_128_MD5') );

     -- create the MQSeries link
    DBMS_MGWADM.CREATE_MSGSYSTEM_LINK(linkname => 'mqssllink',
                                    agent_name => v_agent,
                                    properties => v_prop,
                                    options => v_options);

   -- set Java properties for the agent that specify the JSSE security
   -- properties for the keystore and truststore; the paths will be
   -- saved as cleartext and the passwords encrypted

   DBMS_MGWADM.SET_OPTION(target_type  => DBMS_MGWADM.AGENT_JAVA_PROP,
                       target_name  => v_agent,
                       option_name  => 'javax.net.ssl.keyStore',
                       option_value => '/tmp/mq_ssl/key.jks',
                       encrypted    => false);

   DBMS_MGWADM.SET_OPTION(target_type  => DBMS_MGWADM.AGENT_JAVA_PROP,
                       target_name  => v_agent,
                       option_name  => 'javax.net.ssl.keyStorePassword',
                       option_value => 'welcome',
                       encrypted    => true);

   DBMS_MGWADM.SET_OPTION(target_type  => DBMS_MGWADM.AGENT_JAVA_PROP,
                       target_name  => v_agent,
                       option_name  => 'javax.net.ssl.trustStore',
                       option_value => '/tmp/mq_ssl/trust.jks',
                       encrypted    => false);

   DBMS_MGWADM.SET_OPTION(target_type  => DBMS_MGWADM.AGENT_JAVA_PROP,
                       target_name  => v_agent,
                       option_name  => 'javax.net.ssl.trustStorePassword',
                       option_value => 'welcome',
                       encrypted    => true);
END;
```

Creating a TIB/Rendezvous Link

A TIB/Rendezvous link is created by calling `DBMS_MGWADM.CREATE_MSGSYSTEM_LINK` with three parameters (`service`, `network` and `daemon`) for the agent to create a corresponding transport of `TibrvRvdTransport` type.

A TIB/Rendezvous message system link does not need propagation log queues. Logging information is stored in memory. Therefore, Messaging Gateway can only guarantee at-most-once message delivery.

Example C-16 configures a TIB/Rendezvous link named `rvlink` that connects to the `rvd` daemon on the local computer. An agent name is not specified for the link so the Messaging

Gateway default agent (`DEFAULT_AGENT`) is responsible for the link and all propagation jobs using the link.

> **See Also:**
>
> "TIB/Rendezvous System Properties"

Example C-16   Configuring a TIB/Rendezvous Link

```
DECLARE
  v_options sys.mgw_properties;
  v_prop     sys.mgw_tibrv_properties;
BEGIN
  v_prop := sys.mgw_tibrv_properties.construct();

  DBMS_MGWADM.CREATE_MSGSYSTEM_LINK(linkname => 'rvlink', properties => v_prop);
END;
```

Altering a Messaging System Link

Using `DBMS_MGWADM.ALTER_MSGSYSTEM_LINK`, you can alter some link information after the link is created. You can alter link information with the Messaging Gateway agent running or shut down. Example C-17 alters the link `mqlink` to change the `max_connections` property.

> **See Also:**
>
> "Configuration Properties" for restrictions on changes when the Messaging Gateway agent is running

Example C-17   Altering a WebSphere MQ Link

```
DECLARE
  v_options sys.mgw_properties;
  v_prop sys.mgw_mqseries_properties;
BEGIN
  -- use alter_construct() for initialization
  v_prop := sys.mgw_mqseries_properties.alter_construct();
  v_prop.max_connections := 2;

  DBMS_MGWADM.ALTER_MSGSYSTEM_LINK(
    linkname => 'mqlink', properties => v_prop);
END;
```

Removing a Messaging System Link

You can remove a Messaging Gateway link to a non-Oracle messaging system with `DBMS_MGWADM.REMOVE_MSGSYSTEM_LINK`, but only if all registered queues associated with this link have already been unregistered. The link can be removed with the Messaging Gateway agent running or shut down. Example C-18 removes the link `mqlink`.

Example C-18 Removing a Messaging Gateway Link

```
BEGIN
  dbms_mgwadm.remove_msgsystem_link(linkname =>'mqlink');
END;
```

Views for Messaging System Links

You can use the `MGW_LINKS` view to check links that have been created. It lists the name and link type, as shown in Example C-19.

You can use the `MGW_MQSERIES_LINKS` and `MGW_TIBRV_LINKS` views to check messaging system type-specific configuration information, as shown in Example C-20.

Example C-19 Listing All Messaging Gateway Links

```
SQL> select link_name, link_type from MGW_LINKS;

LINK_NAME     LINK_TYPE
------------------------
MQLINK        MQSERIES
RVLINK        TIBRV
```

Example C-20 Checking Messaging System Link Configuration Information

```
SQL> select link_name, queue_manager, channel, hostname from mgw_mqseries_links;

LINK_NAME   QUEUE_MANAGER      CHANNEL     HOSTNAME
----------------------------------------------------------
MQLINK      my.queue.manager  mychannel  myhost.mydomain

SQL> select link_name, service, network, daemon from mgw_tibrv_links;

LINK_NAME   SERVICE      NETWORK     DAEMON
-----------------------------------------------------
RVLINK
```

Configuring Non-Oracle Messaging System Queues

All non-Oracle messaging system queues involved in propagation as a source queue, destination queue, or exception queue must be registered through the Messaging Gateway administration interface. You do not need to register Oracle Database Advanced Queuing queues involved in propagation.

This section contains these topics:

- Registering a Non-Oracle Queue
- Unregistering a Non-Oracle Queue
- View for Registered Non-Oracle Queues

Registering a Non-Oracle Queue

You can register a non-Oracle queue using `DBMS_MGWADM.REGISTER_FOREIGN_QUEUE`. Registering a non-Oracle queue provides information for the Messaging Gateway agent to access the queue. However, it does not create the physical queue in the non-Oracle messaging system. The physical queue must be created using the non-Oracle messaging system administration interfaces before the Messaging Gateway agent accesses the queue.

The following information is used to register a non-Oracle queue:

- Name of the messaging system link used to access the queue
- Native name of the queue (its name in the non-Oracle messaging system)
- Domain of the queue
  - DBMS_MGWADM.DOMAIN_QUEUE for a point-to-point queue
  - DBMS_MGWADM.DOMAIN_TOPIC for a publish/subscribe queue
- Options specific to the non-Oracle messaging system

  These options are a set of {name, value} pairs, both of which are strings.

> **See Also:**
>
> "Optional Foreign Queue Configuration Properties"

Example C-21 shows how to register the WebSphere MQ Base Java queue my_mq_queue as a Messaging Gateway queue destq.

Example C-21 Registering a WebSphere MQ Base Java Queue

```
BEGIN
  DBMS_MGWADM.REGISTER_FOREIGN_QUEUE(
    name            => 'destq',
    linkname        => 'mqlink',
    provider_queue  => 'my_mq_queue',
    domain          => dbms_mgwadm.DOMAIN_QUEUE);
END;
```

Registering a WebSphere MQ Base Java Queue

The domain must be DBMS_MGWADM.DOMAIN_QUEUE or NULL, because only point-to-point queues are supported for WebSphere MQ.

Registering a WebSphere MQ JMS Queue or Topic

When registering a WebSphere MQ JMS queue, the domain must be DBMS_MGWADM.DOMAIN_QUEUE, and the linkname parameter must refer to a WebSphere MQ JMS unified link or queue link.

When registering a WebSphere MQ JMS topic, the domain must be DBMS_MGWADM.DOMAIN_TOPIC, and the linkname parameter must refer to a WebSphere MQ JMS unified link or topic link. The provider_queue for a WebSphere MQ JMS topic used as a propagation source may include wildcards. See WebSphere MQ documentation for wildcard syntax.

Registering a TIB/Rendezvous Subject

When registering a TIB/Rendezvous subject with Messaging Gateway, the provider_queue parameter specifies a TIB/Rendezvous subject name. The domain of a registered TIB/Rendezvous queue must be DBMS_MGWADM.DOMAIN_TOPIC or NULL.

A registered TIB/Rendezvous queue with provider_queue set to a wildcard subject name can be used as a propagation source queue for inbound propagation. It is not recommended to use queues with wildcard subject names as propagation destination queues or exception queues. As documented in TIB/Rendezvous, sending messages to wildcard subjects can trigger unexpected behavior. However, neither Messaging Gateway nor TIB/Rendezvous prevents you from doing so.

Unregistering a Non-Oracle Queue

A non-Oracle queue can be unregistered with DBMS_MGWADM.UNREGISTER_FOREIGN_QUEUE, but only if there are no propagation jobs referencing it.

Example C-22 unregisters the queue destq of the link mqlink.

Example C-22 Unregistering a Non-Oracle Queue

```
BEGIN
  DBMS_MGWADM.UNREGISTER_FOREIGN_QUEUE(name =>'destq', linkname=>'mqlink');
END;
```

View for Registered Non-Oracle Queues

You can use the MGW_FOREIGN_QUEUES view to check which non-Oracle queues are registered and what link each uses, as shown in Example C-23.

Example C-23 Checking Which Queues Are Registered

```
SELECT name, link_name, provider_queue FROM MGW_FOREIGN_QUEUES;

NAME   LINK_NAME   PROVIDER_QUEUE
------------------------------------
DESTQ  MQLINK       my_mq_queue
```

Configuring Oracle Messaging Gateway Propagation Jobs

Propagating messages between an Oracle Database AQ queue and a non-Oracle messaging system queue requires a propagation job. Each propagation job will have a unique propagation type, source, and destination triplet.

You can create a propagation job to propagate messages between JMS destinations. You can also create a propagation job to propagate messages between non-JMS queues. Messaging Gateway does not support message propagation between a JMS destination and a non-JMS queue.

This section contains these topics:

- Propagation Job Overview
- Creating an Oracle Messaging Gateway Propagation Job
- Enabling and Disabling a Propagation Job
- Resetting a Propagation Job
- Altering a Propagation Job
- Removing a Propagation Job

Propagation Job Overview

A propagation job specifies what messages are propagated and how the messages are propagated.

Messaging Gateway allows bidirectional message propagation. An outbound propagation moves messages from Oracle Database Advanced Queuing to non-Oracle messaging systems. An inbound propagation moves messages from non-Oracle messaging systems to Oracle Database Advanced Queuing.

If the propagation source is a queue (point-to-point), then the Messaging Gateway agent moves all messages from the source queue to the destination queue. If the propagation source is a topic (publish/subscribe), then the Messaging Gateway agent creates a subscriber of the propagation source queue in the messaging system. The agent only moves messages that are published to the source queue after the subscriber is created.

When propagating a message, the Messaging Gateway agent converts the message from the format in the source messaging system to the format in the destination messaging system. Users can customize the message conversion by providing a message transformation. If message conversion fails, then the message will be moved to an exception queue, if one has been provided, so that the agent can continue to propagate messages for the subscriber.

A Messaging Gateway exception queue is different from an Oracle Database Advanced Queuing exception queue. Messaging Gateway moves a message to a Messaging Gateway exception queue when message conversion fails. Oracle Database Advanced Queuing moves a message to an Oracle Database Advanced Queuing exception queue after `MAX_RETRIES` dequeue attempts on the message.

Messages moved to an Oracle Database Advanced Queuing exception queue may result in irrecoverable failures on the associated Messaging Gateway propagation job. To avoid the problem, the `MAX_RETRIES` parameter of any Oracle Database Advanced Queuing queue that is used as the propagation source of a Messaging Gateway propagation job should be set to a value much larger than 16.

If the messaging system of the propagation source queue supports message selection, then a message selection rule can be specified for a propagation subscriber. Only messages that satisfy the message selector will be propagated.

Users can also specify propagation job options to control how messages are propagated, such as options for JMS message delivery mode and TIB/Rendezvous queue policies.

The `MGW_JOBS` view can be used to check the configuration and status of Messaging Gateway propagation jobs.

> **See Also:**
>
> Monitoring Oracle Messaging Gateway

Creating an Oracle Messaging Gateway Propagation Job

Messaging Gateway propagation jobs are created by `DBMS_MGWADM.CREATE_JOB`.

If the propagation source for non-JMS propagation is an Oracle Database AQ queue, then the queue can be either a single consumer queue or multiple consumer queue. If it is a multiple consumer queue, Messaging Gateway creates a corresponding Oracle Database AQ subscriber `MGW_job_name` for the propagation job `job_name` when `DBMS_MGWADM.CREATE_JOB` is called.

If the propagation source is a JMS topic, such as an Oracle Java Message Service (Oracle JMS) topic or a WebSphere MQ JMS topic, then a JMS subscriber `MGW_job_name` is created on the topic in the source messaging system by the Messaging Gateway agent. If the agent is not running, then the subscriber will not be created until the agent is restarted.

If the propagation source is a queue, then only one propagation job can be created using that queue as the propagation source. If the propagation source is a topic, then multiple propagation jobs can be set up using that topic as the propagation source with each propagation job having its own corresponding subscriber on the topic in the messaging system.

Example C-24 creates Messaging Gateway propagation job `job_aq2mq`.

> **Note:**
>
> If a WebSphere MQ JMS topic is involved in a propagation job and the interface type of the link is `DBMS_MGWADM.JMS_TOPIC_CONNECTION`, then a durable subscriber MGL_*subscriber_id* is created on the log topic. The durable subscriber is removed when the Messaging Gateway propagation job is successfully removed.

Example C-24 Creating a Messaging Gateway Propagation Job

```
BEGIN
   DBMS_MGWADM.CREATE_JOB(
     job_name            =>  'job_aq2mq',
     propagation_type    =>   DBMS_MGWADM.OUTBOUND_PROPAGATION,
     source              =>  'mquser.srcq',
     destination         =>  'deqq@mqlink');
END;
```

Enabling and Disabling a Propagation Job

A propagation job can be initially enabled or disabled when it is created by `DBMS_MGWADM.CREATE_JOB`. By default, a job is enabled when it is created. You can use `DBMS_MGWADM.ENABLE_JOB` to enable a propagation job and `DBMS_MGWADM.DISABLE_JOB` to disable a job. No propagation processing will occur when the job is disabled.

Example C-25 enables the propagation for propagation job `job_aq2mq`.

Example C-26 disables the propagation for propagation job `job_aq2mq`.

Example C-25 Enabling a Messaging Gateway Propagation Job

```
BEGIN
  DBMS_MGWADM.ENABLE_JOB(job_name => 'job_aq2mq');
END;
```

Example C-26 Disabling a Messaging Gateway Propagation Job

```
BEGIN
  DBMS_MGWADM.DISABLE_JOB(job_name => 'job_aq2mq');
END;
```

Resetting a Propagation Job

When a problem occurs with a propagation job, the Messaging Gateway agent retries the failed operation up to 16 times in an exponential backoff scheme before the propagation job stops. You can use `DBMS_MGWADM.RESET_JOB` to reset the failure count to zero to allow the agent to retry the failed operation immediately.

Example C-27 resets the failure count for propagation job `job_aq2mq`.

Example C-27 Resetting a Propagation Job

```
BEGIN
  DBMS_MGWADM.RESET_JOB (job_name => 'job_aq2mq');
END;
```

Altering a Propagation Job

After a propagation job is created you can alter the selection rule, transformation, exception queue, job options, and poll interval of the job using DBMS_MGWADM.ALTER_JOB. The job can be altered with the Messaging Gateway running or shut down.

Example C-28 adds an exception queue for a propagation job.

Example C-29 changes the polling interval for a propagation job. The polling interval determines how soon the agent can discover the available messages in the propagation source queue. The default polling interval is 5 seconds or the value set for oracle.mgw.polling_interval in the Messaging Gateway initialization file.

Example C-28 Altering Propagation Job by Adding an Exception Queue

```
BEGIN
    DBMS_MGWADM.ALTER_JOB(
       job_name         =>  'job_aq2mq',        exception_queue  =>
'mgwuser.my_ex_queue');
END;
```

Example C-29 Altering Propagation Job by Changing the Polling Interval

```
BEGIN
    DBMS_MGWADM.ALTER_JOB(
        job_name       =>  'job_aq2mq',        poll_interval  =>  2);
END;
```

Removing a Propagation Job

You can remove a Messaging Gateway propagation job by calling DBMS_MGWADM.REMOVE_JOB.

Before removing the propagation job from the Messaging Gateway configuration, Messaging Gateway does the following cleanup:

- Removes from the messaging system the associated subscriber that may have been created by Messaging Gateway
- Removes propagation log records from log queues for the job being removed

Messaging Gateway may fail to do the cleanup because:

- The Messaging Gateway agent is not running
- Non-Oracle messaging system is not running
- The Messaging Gateway agent cannot interact with the source or destination messaging system

If the Messaging Gateway cleanup fails for any reason, then the propagation job being removed is placed in a DELETE_PENDING state. The Messaging Gateway agent tries to clean up propagation jobs in a DELETE_PENDING state when:

- DBMS_MGWADM.REMOVE_JOB is called and the Messaging Gateway agent is running.
- The Messaging Gateway agent is starting and finds a propagation job in a DELETE_PENDING state.

DBMS_MGWADM.REMOVE_JOB has a force parameter that allows you to force the propagation job to be removed from the Messaging Gateway configuration without placing it in DELETE_PENDING state. This is useful in case of cleanup failures or if you want to remove a propagation job when the Messaging Gateway agent is not running.

Forcing a propagation job to be removed may result in obsolete log records being left in the log queues, and subscriptions in the messaging systems that may cause unnecessary message accumulation. Oracle recommends that the force option not be used for DBMS_MGWADM.REMOVE_JOB if possible.

Example C-30 removes a propagation job in a non-forced manner.

Example C-30   Removing a Propagation Job

```
BEGIN
    DBMS_MGWADM.REMOVE_JOB (job_name  =>  'job_aq2mq');
END;
```

Propagation Jobs, Subscribers, and Schedules

Subprograms are provided as part of the DBMS_MGWADM package that simplify the creation and management of propagation jobs. Those subprograms allow a user to configure a propagation job rather than a disjoint subscriber and schedule as was done in prior releases. Oracle recommends that you use the propagation job procedures but still supports the subscriber and schedule procedures for backward compatibility.

Table C-2 lists the Messaging Gateway propagation job procedures and shows which subscriber and/or schedule procedures it replaces. All procedures are from the DBMS_MGWADM package.

Table C-2   Messaging Gateway Propagation Job Subprograms

Job Procedure	Replaces Subscriber, Schedule Procedure
CREATE_JOB	ADD_SUBSCRIBER, SCHEDULE_PROPAGATION
ALTER_JOB	ALTER_SUBSCRIBER, ALTER_PROPAGATION_SCHEDULE
REMOVE_JOB	REMOVE_SUBSCRIBER, UNSCHEDULE_PROPAGATION
ENABLE_JOB	ENABLE_PROPAGATION_SCHEDULE
DISABLE_JOB	DISABLE_PROPAGATION_SCHEDULE
RESET_JOB	RESET_SUBSCRIBER

This section contains the following topics:

- Propagation Job_ Subscriber_ Schedule Interface Interoperability
- Propagation Job_ Subscriber_ Schedule Views
- Single Consumer Queue as Propagation Source

Propagation Job, Subscriber, Schedule Interface Interoperability

The user can create two types of propagation jobs, a new style job or an old style job. A new style job is created by DBMS_MGWADM.CREATE_JOB. An old style job is created by calling DBMS_MGWADM.ADD_SUBSCRIBER and DBMS_MGWADM.SCHEDULE_PROPAGATION using the same {propagation_type, source, destination} triplet. A subscriber that does not have a matching schedule, or a schedule that does not have a matching subscriber, is not considered to be a propagation job.

For new style job, the job name will serve as both the subscriber ID and the schedule ID. For an old style job, the subscriber ID is used as the job name.

Both the propagation job subprograms and the subscriber/schedule subprograms can be used for old style propagation jobs. Oracle recommends that you use the job subprograms to create and manage propagation jobs. The job subprograms cannot be used for an unmatched subscriber or schedule since those do not constitute a propagation job.

Only the new job subprograms can be used for new style propagation jobs. An error will occur if a user tries to call a subscriber or scheduler procedure on a new style job.

Other than `DBMS_MGWADM.REMOVE_JOB`, calling the job subprograms for an old style job is straightforward and the results are effectively the same as calling the corresponding subscriber/schedule subprograms. There may be certain restrictions in the future but there are none at this time.

The `DBMS_MGWADM.REMOVE_JOB` procedure can be used to remove both new style and old style jobs. A forced and non-forced remove is supported. If the Messaging Gateway agent is not running when a non-forced remove is done, the job will be flagged as delete pending and neither the underlying subscriber nor schedule will be removed at that time. The job (subscriber /schedule pair) will be removed once the agent is restarted and performs its cleanup work or a forced `DBMS_MGWADM.REMOVE_JOB` is performed. In order to insure that the subscriber/schedule pair is removed at the same time, an error will occur if you first call `DBMS_MGWADM.REMOVE_JOB` and subsequently attempt to call `DBMS_MGWADM.REMOVE_SUBSCRIBER` or `DBMS_MGWADM.UNSCHEDULE_PROPAGATION` for an old style job.

Once `DBMS_MGWADM.REMOVE_JOB` as been called for a job and it has been flagged as delete pending, all job procedures, other than `DBMS_MGWADM.REMOVE_JOB`, will fail for both new style and old style jobs. In addition, all subscriber and schedule subprograms will fail if the propagation job happens to be an old style job.`DBMS_MGWADM.REMOVE_SUBSCRIBER` and `DBMS_MGWADM.UNSCHEDULE_PROPAGATION` can be used for an old style job as long as `DBMS_MGWADM.REMOVE_JOB` has not been called for that job. If `DBMS_MGWADM.UNSCHEDULE_PROPAGATION` is called for an old style job, the schedule is immediately removed and it ceases to be a propagation job and `DBMS_MGWADM.REMOVE_SUSCRIBER` must be used to remove the subscriber. If `DBMS_MGWADM.REMOVE_SUBSCRIBER` is called for an old style job, the user can subsequently call `DBMS_MGWADM.REMOVE_JOB` as long as the subscriber exists.

Propagation Job, Subscriber, Schedule Views

The `MGW_JOBS` view shows information for the current propagation jobs, both new style jobs and old style jobs, and includes all the pertinent information shown by the `MGW_SUBSCRIBERS` and `MGW_SCHEDULES` views. The `MGW_SUBSCRIBERS` and `MGW_SCHEDULES` views are still useful for finding an unmatched subscriber or schedule since they don't constitute a propagation job and will not show up in the `MGW_JOBS` view.

Single Consumer Queue as Propagation Source

Messaging Gateway allows an Oracle Database AQ multiple consumer queue or a single consumer queue to be a propagation source for an outbound new style job created by `DBMS_MGWADM.CREATE_JOB`. A multiple consumer queue must be used for the propagation source for an outbound old style job. An error will occur if an administrator attempts to call `DBMS_MGWADM.ADD_SUBSCRIBER` and the source is a single consumer queue.

An Oracle Database AQ dequeue condition is not supported for native (non-JMS) outbound propagation when the propagation source is a single consumer queue.

Configuration Properties

This section summarizes basic and optional properties related to Messaging Gateway links, foreign queues, and propagation jobs.

This section contains these topics:

- WebSphere MQ System Properties
- TIB/Rendezvous System Properties
- Optional Link Configuration Properties
- Optional Foreign Queue Configuration Properties
- Optional Job Configuration Properties

WebSphere MQ System Properties

Table C-3 summarizes the basic configuration properties for a WebSphere MQ messaging link. The table indicates which properties of SYS.MGW_MQSERIES_PROPERTIES are optional (NULL allowed), which can be altered, and if alterable, which values can be dynamically changed.

> **See Also:**
>
> "SYS.MGW_MQSERIES_PROPERTIES Type" in *Oracle Database PL/SQL Packages and Types Reference*

Table C-3 WebSphere MQ Link Properties

Attribute	NULL Allowed?	Alter Value?	Dynamic?
queue_manager	no	no	no
hostname	yes (1)	no	no
port	yes (1)	no	no
channel	yes (1), (6)	yes	no
interface_type	yes (2)	no	no
max_connections	yes (3)	yes	yes
username	yes	yes	yes
password	yes	yes	yes
inbound_log_queue	yes (4)	yes(4)	yes
outbound_log_queue	yes (5)	yes(5)	yes

Notes on Table C-3

- If hostname is NULL, then the port and channel must be NULL. If the hostname is not NULL, then the port must be not NULL. If the hostname is NULL, then a WebSphere MQ bindings connection is used; otherwise a client connection is used.

- If interface_type is NULL, then a default value of DBMS_MGWADM.MQSERIES_BASE_JAVA_INTERFACE is used.
- If max_connections is NULL, then a default value of 1 is used.
- Attribute inbound_log_queue can be NULL if the link is not used for inbound propagation. The log queue can be altered only when no inbound propagation job references the link.
- Attribute outbound_log_queue can be NULL if the link is not used for outbound propagation. The log queue can be altered only when no outbound propagation job references the link.
- The channel attribute must be NULL if a client channel definition table (CCDT) is used. The MQ_ccdtURL link option can be used to specify a CCDT.

Table C-4 summarizes the optional configuration properties supported when a WebSphere MQ Base Java interface is used to access the WebSphere MQ messaging system. Table C-5 summarizes the optional configuration properties supported when a WebSphere MQ JMS interface is used. Each table lists the property name, where that property applies, whether the property can be altered, and if alterable, whether the value can be dynamically changed. Only the properties listed in the tables are supported, and any extra properties are ignored.

Table C-4 Optional Configuration Properties for WebSphere MQ Base Java

Property Name	Used For	Alter Value?	Dynamic?
MQ_ccdtUrl	link	yes	no
MQ_ccsid	link	yes	no
MQ_ReceiveExit	link	yes	no
MQ_SecurityExit	link	yes	no
MQ_SendExit	link	yes	no
MQ_SSLCipherSuite	link	yes	no
MQ_SSLFipsRequired	link	yes	no
MQ_SSLPeerName	link	yes	no
MQ_SSLResetCount	link	yes	no
MQ_openOptions	foreign queue	no	no
MsgBatchSize	job	yes	yes
PreserveMessageID	job	yes	yes

Table C-5 Optional Configuration Properties for WebSphere MQ JMS

Property Name	Used For	Alter Value?	Dynamic?
MQ_BrokerControlQueue	link	yes	no
MQ_BrokerPubQueue	link	yes	no
MQ_BrokerQueueManager	link	yes	no
MQ_BrokerVersion	link	yes	no
MQ_ccdtUrl	link	yes	no
MQ_ccsid	link	yes	no
MQ_JmsDurSubQueue	link	no	no
MQ_PubAckInterval	link	yes	no
MQ_ReceiveExit	link	yes	no

Table C-5 (Cont.) Optional Configuration Properties for WebSphere MQ JMS

Property Name	Used For	Alter Value?	Dynamic?
MQ_ReceiveExitInit	link	yes	no
MQ_SecurityExit	link	yes	no
MQ_SecurityExitInit	link	yes	no
MQ_SendExit	link	yes	no
MQ_SendExitInit	link	yes	no
MQ_SSLCipherSuite	link	yes	no
MQ_SSLCrl	link	yes	no
MQ_SSLFipsRequired	link	yes	no
MQ_SSLPeerName	link	yes	no
MQ_SSLResetCount	link	yes	no
MQ_CharacterSet	foreign queue	no	no
MQ_JmsDurSubQueue	foreign queue	no	no
MQ_JmsTargetClient	foreign queue	no	no
JMS_DeliveryMode	job	yes	yes
JMS_NoLocal	job	no	no
MsgBatchSize	job	yes	yes
PreserveMessageID	job	yes	yes

TIB/Rendezvous System Properties

Table C-6 summarizes the basic configuration properties for a TIB/Rendezvous messaging link. It indicates which properties of `SYS.MGW_TIBRV_PROPERTIES` are optional (`NULL` allowed), which can be altered, and if alterable, which values can be dynamically changed.

> **See Also:**
>
> "SYS.MGW_TIBRV_PROPERTIES Type" in *Oracle Database PL/SQL Packages and Types Reference*

Table C-6 TIB/Rendezvous Link Properties

Attribute	NULL allowed?	Alter value?	Dynamic?
`service`	yes(1)	no	no
`daemon`	yes(1)	no	no
`network`	yes(1)	no	no
`cm_name`	yes(2)	no	no
`cm_ledger`	yes(2)	no	no

Notes on Table C-6:

- System default values will be used if service, daemon, or network are NULL.
- The cm_name and cm_ledger attributes are reserved for future use when TIB/Rendezvous certified messages are supported. At present, a NULL must be specified for these parameters when a TIB/Rendezvous link is configured.

Table C-7 summarizes the optional configuration properties supported when a TIB/Rendezvous messaging system is used. The table lists the property name, where that property applies, whether the property can be altered, and if alterable, whether the value can be dynamically changed. Only the properties listed in the table are supported, and any extra properties will be ignored.

Table C-7 Optional Properties for TIB/Rendezvous

Property Name	Used For	Alter Value?	Dynamic?
AQ_MsgProperties	job	yes	yes
MsgBatchSize	job	yes	yes
PreserveMessageID	job	yes	yes
RV_discardAmount	job	yes	no
RV_limitPolicy	job	yes	no
RV_maxEvents	job	yes	no

Optional Link Configuration Properties

This section describes optional link properties you can specify using the options parameter of DBMS_MGWADM.CREATE_MSGSYSTEM_LINK and DBMS_MGWADM.ALTER_MSGSYSTEM_LINK. Each listing also indicates which messaging system might use that property.

MQ_BrokerControlQueue
This property is used by WebSphere MQ JMS. It specifies the name of the broker control queue and corresponds to WebSphere MQ JMS administration tool property BROKERCONQ. The WebSphere MQ default is SYSTEM.BROKER.CONTROL.QUEUE.

MQ_BrokerPubQueue
This property is used by WebSphere MQ JMS. It specifies the name of the broker publish queue and corresponds to WebSphere MQ JMS administration tool property BROKERPUBQ. The WebSphere MQ default is SYSTEM.BROKER.DEFAULT.STREAM.

MQ_BrokerQueueManager
This property is used by WebSphere MQ JMS. It specifies the name of the broker queue manager and corresponds to WebSphere MQ administration tool property BROKERQMGR. If it is not set, then no default is used.

MQ_BrokerVersion
This property is used by WebSphere MQ JMS. It specifies the broker version number and corresponds to WebSphere MQ JMS administration tool property BROKERVER. The WebSphere MQ default is 0.

MQ_ccdtUrl
This property is used by WebSphere MQ Base Java and WebSphere MQ JMS. It specifies the URL string of a client channel definition table (CCDT) to be used. If not set, a CCDT is not used. If a CCDT is used, then the `SYS.MGW_MQSERIES_PROPERTIES.channel` link property must be `NULL`.

MQ_ccsid
This property is used by WebSphere MQ Base Java and WebSphere MQ JMS. It specifies the character set identifier to be used to translate information in the WebSphere MQ message header. This should be the integer value of the character set (for example, `819`) rather than a descriptive string. If it is not set, then the WebSphere MQ default character set `819` is used.

MQ_JmsDurSubQueue
This property is used by WebSphere MQ JMS. It applies to WebSphere MQ JMS topic links only. The `SYS.MGW_MQSERIES_PROPERITES` attributes, `inbound_log_queue` and `outbound_log_queue`, specify the names of WebSphere MQ JMS topics used for propagation logging. This property specifies the name of the WebSphere MQ queue from which durable subscription messages are retrieved by the log topic subscribers. The WebSphere MQ default queue is `SYSTEM.JMS.D.SUBSCRIBER.QUEUE`.

MQ_PubAckInterval
This property is used by WebSphere MQ JMS. It specifies the interval, in number of messages, between publish requests that require acknowledgment from the broker and corresponds to WebSphere MQ JMS administration tool property `PUBACKINT`. The WebSphere MQ default is `25`.

MQ_ReceiveExit
This property is used by WebSphere MQ Base Java and WebSphere MQ JMS. It specifies the fully qualified Java classname of a class implementing the `MQReceiveExit` interface. This class must be in the `CLASSPATH` of the Messaging Gateway agent. There is no default.

MQ_ReceiveExitInit
This initialization string is used by WebSphere MQ JMS. It is passed by WebSphere MQ JMS to the constructor of the class specified by `MQ_ReceiveExit` and corresponds to WebSphere MQ JMS administration tool property `RECEXITINIT`. There is no default.

MQ_SecurityExit
This property is used by WebSphere MQ Base Java and WebSphere MQ JMS. It specifies the fully qualified Java classname of a class implementing the `MQSecurityExit` interface. This class must be in the `CLASSPATH` of the Messaging Gateway agent. There is no default.

MQ_SecurityExitInit
This initialization string is used by WebSphere MQ JMS. It is passed by WebSphere MQ JMS to the constructor of the class specified by `MQ_SecurityExit` and corresponds to WebSphere MQ JMS administration tool property `SECEXITINIT`. There is no default.

MQ_SendExit
This property is used by WebSphere MQ Base Java and WebSphere MQ JMS. It specifies the fully qualified Java classname of a class implementing the `MQSendExit` interface. This class must be in the `CLASSPATH` of the Messaging Gateway agent. There is no default.

MQ_SendExitInit
This initialization string is used by WebSphere MQ JMS. It is passed by WebSphere MQ JMS to the constructor of the class specified by `MQ_SendExit`. It corresponds to WebSphere MQ JMS administration tool property `SENDEXITINIT`. There is no default.

MQ_SSLCipherSuite
This property is used by WebSphere MQ Base Java and WebSphere MQ JMS. It specifies the CipherSuite to be used; for example, `SSL_RSA_WITH_RC4_128_MD5`. This corresponds to the WebSphere MQ `SSLCIPHERSUITE` administration property.

MQ_SSLCrl
This property is used by WebSphere MQ JMS. It specifies a space-delimited list of LDAP servers that can be used for certificate revocation list (CRL) checking. If not set, no CRL checking is done. This corresponds to the WebSphere MQ `SSLCRL` administration property. This option is not supported for WebSphere MQ Base Java, and instead, a client channel definition table (CCDT) must be used if CRL checking is needed.

MQ_SSLFipsRequired
This property is used by WebSphere MQ Base Java and WebSphere MQ JMS. It indicates whether the CipherSuite being used is supported by the IBM Java JSSE FIPS provider (IBMSJSSEFIPS). The value should be `TRUE` or `FALSE`. The default value is `FALSE`. This corresponds to the WebSphere MQ `SSLFIPSREQUIRED` administration property.

MQ_SSLPeerName
This property is used by WebSphere MQ Base Java and WebSphere MQ JMS. It specifies a distinguished name (DN) pattern that the queue manager certificate must match in order for a connection to be established. If not set, no DN check is performed. This corresponds to the WebSphere MQ `SSLPEERNAME` administration property.

MQ_SSLResetCount
This property is used by WebSphere MQ Base Java and WebSphere MQ JMS. It specifies the total number of bytes sent and received before the secret key is renegotiated. If not set, the key is not renegotiated. This corresponds to the WebSphere MQ `SSLRESETCOUNT` administration property.

Optional Foreign Queue Configuration Properties

This section describes optional foreign queue properties that you can specify using the `options` parameter of `DBMS_MGWADM.REGISTER_FOREIGN_QUEUE`. Each listing also indicates which messaging system might use that property.

MQ_CharacterSet
This property is used by WebSphere MQ JMS. It is used only for outbound propagation to a JMS queue or topic. It specifies the character set to be used to encode text strings sent to the destination. It should be the integer value of the character set (for example, `1208`) rather than a descriptive string. The default value used by Messaging Gateway is `1208` (UTF8).

MQ_JmsDurSubQueue
This property is used by WebSphere MQ JMS. It is a string representing the name of the WebSphere MQ queue from which durable subscription messages are retrieved by subscribers on this topic. It applies only to WebSphere MQ JMS topics. The WebSphere MQ default queue is `SYSTEM.JMS.D.SUBSCRIBER.QUEUE`.

MQ_JmsTargetClient
This property is used by WebSphere MQ JMS. It is used only for outbound propagation to a JMS queue or topic. Supported values are `TRUE` and `FALSE`. `TRUE` indicates that WebSphere MQ should store the message as a JMS message. `FALSE` indicates that WebSphere MQ should store the message in non-JMS format so that non-JMS applications can access it. Default is `TRUE`.

MQ_openOptions
This property is used by WebSphere MQ Base Java. It specifies the value used for the `openOptions` argument of the WebSphere MQ Base Java `MQQueueManager.accessQueue` method. No value is required. But if one is given, then the Messaging Gateway agent adds `MQOO_OUTPUT` to the specified value for an enqueue (`put`) operation. `MQOO_INPUT_SHARED` is added for a dequeue (`get`) operation. The default is `MQOO_OUTPUT` for an enqueue/put operation; `MQOO_INPUT_SHARED` for a dequeue/get operation.

Optional Job Configuration Properties

This section describes optional propagation job properties that you can specify using the options parameter of `DBMS_MGWADM.CREATE_JOB` and `DBMS_MGWADM.ALTER_JOB`.

AQ_MsgProperties
This property is used by TIB/Rendezvous. It specifies how Oracle Database AQ message properties will be used during message propagation. Supported values are `TRUE` and `FALSE`. The default value is `FALSE`.
For an outbound propagation job, if the value is `TRUE` (case insensitive), then the Messaging Gateway agent will add a field for most Oracle Database AQ message properties to the message propagated to the TIB/Rendezvous subject.
For an inbound propagation job, if the value is `TRUE` (case insensitive), then the Messaging Gateway agent will search the source message for a field with a reserved name, and if it exists, use its value to set the corresponding Oracle Database AQ message property. A default value will be used if the field does not exist or does not have an expected datatype.

JMS_DeliveryMode
This property is used by WebSphere MQ JMS and Oracle JMS. You can use this property when the propagation destination is a JMS messaging system. It sets the delivery mode of messages enqueued to the propagation destination queue by a JMS `MessageProducer`. The default is `PRESERVE_MSG`. Supported values and their associated delivery modes are:

- `PERSISTENT` (`DeliveryMode.PERSISTENT`)
- `NON_PERSISTENT` (`DeliveryMode.NON_PERSISTENT`)
- `PRESERVE_MSG` (delivery mode of the source JMS message is used)

JMS_NoLocal
This property is used by WebSphere MQ JMS and Oracle JMS. You can use it when the propagation source is a JMS messaging system. It sets the `noLocal` parameter of a JMS `TopicSubscriber`. `TRUE` indicates that messages that have been published to this topic through the same Messaging Gateway link will not be propagated. The default value `FALSE` indicates that such messages will be propagated from the topic.

MsgBatchSize
This property can be used by any supported messaging system. It specifies the maximum number of messages, if available, to be propagated in one transaction. The default is 30.

PreserveMessageID
This property is used by WebSphere MQ Base Java, WebSphere MQ JMS, TIB/Rendezvous, and Oracle JMS. It specifies whether Messaging Gateway should preserve the original message identifier when the message is propagated to the destination messaging system. The exact details depend on the capabilities of the messaging systems involved. Supported values are `TRUE` and `FALSE`. The default value is `FALSE`.

RV_discardAmount
This property is used by TIB/Rendezvous. It specifies the discard amount of a queue. It is meaningful only for an inbound propagation job. The default is 0.

RV_limitPolicy
This property is used by TIB/Rendezvous. It specifies the limit policy for resolving overflow of a queue limit. It is meaningful only for an inbound propagation job. The default is `DISCARD_NONE`. Supported values and their associated limit policies are: `DISCARD_NONE`, `DISCARD_FIRST`, `DISCARD_LAST` and `DISCARD_NEW`.

- `DISCARD_NONE` (`TibrvQueue.DISCARD_NONE`)
- `DISCARD_FIRST` (`TibrvQueue.DISCARD_FIRST`)
- `DISCARD_LAST` (`TibrvQueue.DISCARD_LAST`)
- `DISCARD_NEW` (`TibrvQueue.DISCARD_NEW`)

RV_maxEvents
This property is used by TIB/Rendezvous. It specifies the maximum event limit of a queue. It is meaningful only for an inbound propagation job. The default is 0.

Oracle Messaging Gateway Message Conversion

The following topics discuss how Oracle Messaging Gateway (MGW) converts message formats from one messaging system to another. A conversion is generally necessary when moving messages between Oracle Database Advanced Queuing and another system, because different messaging systems have different message formats. Java Message Service (JMS) messages are a special case. A JMS message can be propagated only to a JMS destination, making conversion a simple process.

- Converting Oracle Messaging Gateway Non-JMS Messages
- Message Conversion for WebSphere MQ
- Message Conversion for TIB/Rendezvous
- JMS Messages

Converting Oracle Messaging Gateway Non-JMS Messages

MGW converts the native message format of the source messaging system to the native message format of the destination messaging system during propagation. MGW uses canonical types and a model centering on Oracle Database Advanced Queuing for the conversion.

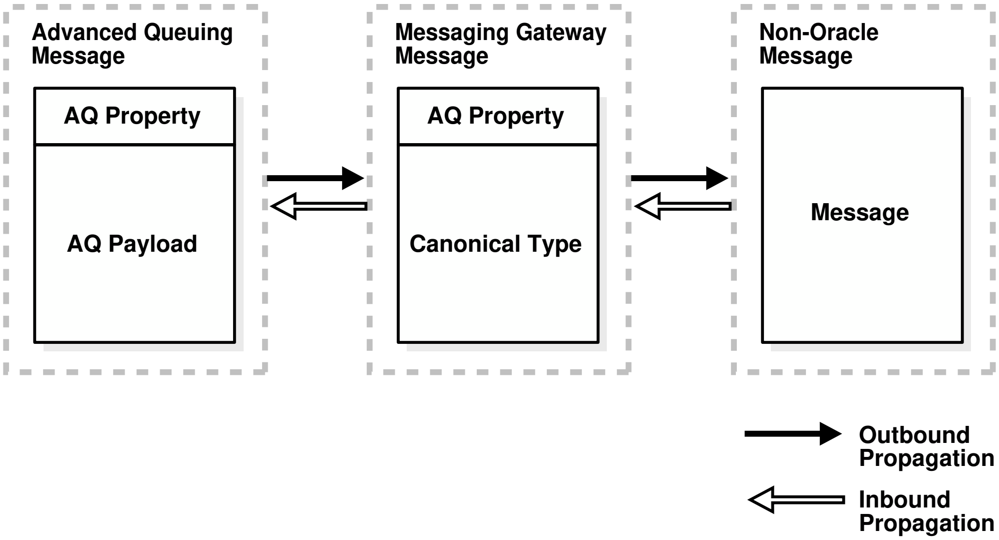
Overview of the Non-JMS Message Conversion Process

When a message is propagated by MGW, the message is converted from the native format of the source queue to the native format of the destination queue.

A native message usually contains a message header and a message body. The header contains the fixed header fields that all messages in that messaging system have, such as message properties in Oracle Database Advanced Queuing and the fixed header in WebSphere MQ. The body contains message contents, such as the Oracle Database Advanced Queuing payload, the WebSphere MQ message body, or the entire TIB/Rendezvous message. MGW converts both message header and message body components.

Figure C-2 shows how non-JMS messages are converted in two stages. A message is first converted from the native format of the source queue to the MGW internal message format, and then it is converted from the internal message format to the native format of the destination queue.

Figure C-2 Non-JMS Message Conversion



The MGW agent uses an internal message format consisting of a header that is similar to the Oracle Database Advanced Queuing message properties and a body that is a representation of an MGW canonical type.

Oracle Messaging Gateway Canonical Types

MGW defines canonical types to support message conversion between Oracle Database Advanced Queuing and non-Oracle messaging systems. A canonical type is a message type representation in the form of a PL/SQL Oracle type in Oracle Database. The canonical types are `RAW`, `SYS.MGW_BASIC_MSG_T`, and `SYS.MGW_TIBRV_MSG_T`.

WebSphere MQ propagation supports the canonical types `SYS.MGW_BASIC_MSG_T` and `RAW`. TIB/Rendezvous propagation supports the canonical types `SYS.MGW_TIBRV_MSG_T` and `RAW`.

> **See Also:**
>
> "DBMS_MGWMSG" in *Oracle Database PL/SQL Packages and Types Reference* for Syntax and attribute information for `SYS.MGW_BASIC_MSG_T` and `SYS.MGW_TIBRV_MSG_T`

Message Header Conversion

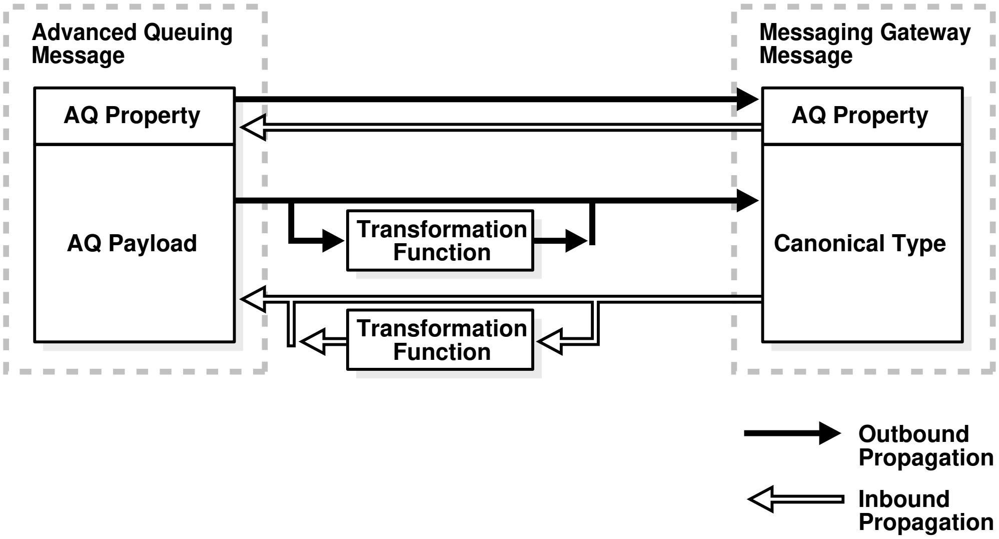
MGW provides default mappings between Oracle Database Advanced Queuing message properties and non-Oracle message header fields that have a counterpart in Oracle Database Advanced Queuing message properties with the same semantics. Where MGW does not provide a mapping, the message header fields are set to a default value, usually the default value defined by the messaging system.

Handling Arbitrary Payload Types Using Message Transformations

When converting to or from Oracle Database Advanced Queuing messages, the MGW agent uses only its canonical types. Arbitrary payload types are supported, however, with the assistance of user-defined Oracle Database Advanced Queuing message transformations to convert between an Oracle Database Advanced Queuing queue payload and an MGW canonical type.

For MGW to propagate messages from an Oracle Database Advanced Queuing queue with an arbitrary ADT payload (outbound propagation), you must provide a mapping to an MGW canonical ADT. The transformation is invoked when the MGW agent dequeues messages from the Oracle Database Advanced Queuing queue. Similarly, for MGW to propagate messages to an Oracle Database Advanced Queuing queue with an arbitrary ADT payload (inbound propagation), you must provide a mapping from an MGW canonical ADT. The transformation is invoked when the MGW agent enqueues messages to the Oracle Database Advanced Queuing queue.

Figure C-3 Oracle Database Advanced Queuing Message Conversion



The transformation is always executed in the context of the MGW agent, which means that the MGW agent user (the user specified using `DBMS_MGWADM.CREATE_AGENT` or `DBMS_MGWADM.ALTER_AGENT`) must have `EXECUTE` privileges on the transformation function and the Oracle Database Advanced Queuing payload type. This can be accomplished by granting the `EXECUTE` privilege to `PUBLIC` or by granting the `EXECUTE` privilege directly to the MGW agent user.

To configure a MGW propagation job with a transformation:

1. Create the transformation function.
2. Grant `EXECUTE` to the MGW agent user or to `PUBLIC` on the function and the object types it references.
3. Call `DBMS_TRANSFORM.CREATE_TRANSFORMATION` to register the transformation.
4. Call `DBMS_MGWADM.CREATE_JOB` to create a MGW propagation job using the transformation, or `DBMS_MGWADM.ALTER_JOB` to alter an existing job.

Example C-31 Transformation Function Signature

```
FUNCTION trans_sampleadt_to_mgw_basic(in_msg IN mgwuser.sampleADT)
RETURN SYS.MGW_BASIC_MSG_T;
```

You can create a transformation using `DBMS_TRANSFORM.CREATE_TRANSFORMATION`, as shown in Example C-32.

Example C-32 Creating a Transformation

```
BEGIN
  DBMS_TRANSFORM.CREATE_TRANSFORMATION(
        schema         => 'mgwuser',
        name           => 'sample_adt_to_mgw_basic',
        from_schema    => 'mgwuser',
        from_type      => 'sampleadt',
        to_schema      => 'sys',
        to_type        => 'MGW_BASIC_MSG_T',
        transformation => 'mgwuser.trans_sampleadt_to_mgw_basic(user_data)');
END;
```

Example C-33 Registering a Transformation

```
BEGIN
  DBMS_MGWADM.CREATE_JOB(
    job_name          =>   'job_aq2mq',
    propagation_type  =>    DBMS_MGWADM.OUTBOUND_PROPAGATION,
    source            =>   'mgwuser.srcq',
    destination       =>   'destq.mqlink',
    transformation    =>   'mgwuser.sample_adt_to_mgw_basic',
    exception_queue   =>   'mgwuser.excq');
END;
```

The value passed in the transformation parameter for these APIs must be the registered transformation name and not the function name. For example, `trans_sampleadt_to_mgw_basic` is a stored procedure representing a transformation function with the signature shown in Example C-31.

Note:

All commands in the examples must be run as a user granted `MGW_ADMINISTRATOR_ROLE`, except for the commands to create transformations.

Once created, this transformation can be registered with MGW when creating a propagation job. Example C-33 creates job `job_aq2mq`, for whom messages are propagated from Oracle Database Advanced Queuing queue `mgwuser.srcq` to non-Oracle messaging system queue `destq@mqlink` using transformation `mgwuser.sample_adt_to_mgw_basic`.

An error that occurs while attempting a user-defined transformation is usually considered a message conversion exception, and the message is moved to the exception queue if it exists.

See Also:

"DBMS_MGWADM", "DBMS_MGWMSG", and "DBMS_TRANSFORM" in *Oracle Database PL/SQL Packages and Types Reference*

Handling Logical Change Records

MGW provides facilities to propagate Logical Change Records (LCRs). Routines are provided to help in creating transformations to handle the propagation of both row LCRs and DDL LCRs stored in queues with payload type ANYDATA. An LCR is propagated as an XML string stored in the appropriate message type.

Note:

For LCR propagation, you must load the XDB package.

Because Oracle Streams uses ANYDATA queues to store LCRs, an ANYDATA queue is the source for outbound propagation. The transformation must first convert the ANYDATA object containing an LCR into an XMLType object using the MGW routine DBMS_MGWMSG.LCR_TO_XML. If the ANYDATA object does not contain an LCR, then this routine raises an error. The XML document string of the LCR is then extracted from the XMLType and placed in the appropriate MGW canonical type (SYS.MGW_BASIC_MSG_T or SYS.MGW_TIBRV_MSG_T).

Example C-34 illustrates a simplified transformation used for LCR outbound propagation. The transformation converts an ANYDATA payload containing an LCR to a SYS.MGW_TIBRV_MSG_T object. The string representing the LCR as an XML document is put in a field named ORACLE_LCR.

For LCR inbound propagation, an MGW canonical type (SYS.MGW_BASIC_MSG_T or SYS.MGW_TIBRV_MSG_T) is the transformation source type. A string in the format of an XML document representing an LCR must be contained in the canonical type. The transformation function must extract the string from the message, create an XMLType object from it, and convert it to an ANYDATA object containing an LCR with the MGW routine DBMS_MGWMSG.XML_TO_LCR. If the original XML document does not represent an LCR, then this routine raises an error.

Example C-35 illustrates a simplified transformation used for LCR inbound propagation. The transformation converts a SYS.MGW_TIBRV_MSG_T object with a field containing an XML string representing an LCR to an ANYDATA object. The string representing the LCR as an XML document is taken from a field named ORACLE_LCR.

See Also:

- "DBMS_MGWMSG" in *Oracle Database PL/SQL Packages and Types Reference*
- *ORACLE_HOME*/mgw/samples/lcr for complete examples of LCR transformations

Example C-34 Outbound LCR Transformation

```
create or replace function any2tibrv(adata in anydata)
return SYS.MGW_TIBRV_MSG_T is
    v_xml   XMLType;
    v_text  varchar2(2000);
    v_tibrv sys.mgw_tibrv_msg_t;
```

```
BEGIN
    v_xml   := dbms_mgwmsg.lcr_to_xml(adata);
    -- assume the lcr is smaller than 2000 characters long.
    v_text  := v_xml.getStringVal();
    v_tibrv := SYS.MGW_TIBRV_MSG_T.CONSTRUCT;
    v_tibrv.add_string('ORACLE_LCR', 0, v_text);
    return v_tibrv;
END any2tibrv;
```

Example C-35 Inbound LCR Transformation

```
create or replace function tibrv2any(tdata in sys.mgw_tibrv_msg_t)
return anydata is
    v_field   sys.mgw_tibrv_field_t;
    v_xml     XMLType;
    v_text    varchar2(2000);
    v_any     anydata;
BEGIN
    v_field := tdata.get_field_by_name('ORACLE_LCR');
    -- type checking
    v_text  := v_field.text_value;
    -- assume it is not null
    v_xml  := XMLType.createXML(v_text);
    v_any  := dbms_mgwmsg.xml_to_lcr(v_xml);
    return v_any;
END tibrv2any;
```

Message Conversion for WebSphere MQ

MGW converts between the MGW canonical types and the WebSphere MQ native message format. WebSphere MQ native messages consist of a fixed message header and a message body. The message body is treated as either a `TEXT` value or `RAW` (bytes) value. The canonical types supported for WebSphere MQ propagation are `SYS.MGW_BASIC_MSG_T` and `RAW`.

Figure C-4 Message Conversion for WebSphere MQ Using MGW_BASIC_MSG_T

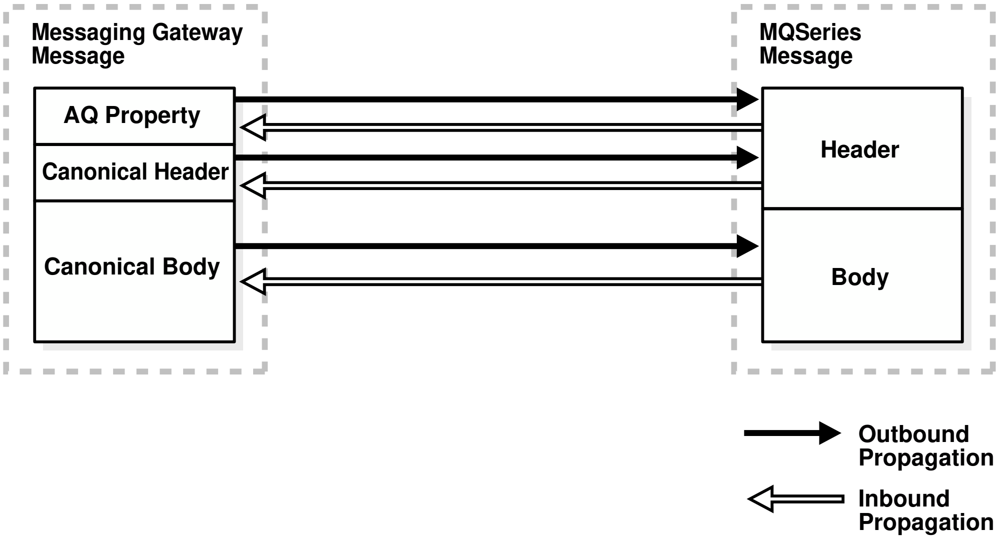


Figure C-4 illustrates the message conversion performed by the MGW WebSphere MQ driver when using the canonical type `SYS.MGW_BASIC_MSG_T`. For outbound propagation, the driver maps the Oracle Database Advanced Queuing message properties and canonical message to a WebSphere MQ message having a fixed header and a message body. For inbound propagation, the driver maps a native message to a set of Oracle Database Advanced Queuing message properties and a canonical message. When the canonical type is RAW, the mappings are the same, except no canonical headers exist.

WebSphere MQ Message Header Mappings

When the MGW canonical type used in an outbound propagation job is RAW, no WebSphere MQ header information is set from the RAW message body. Similarly, for inbound propagation no WebSphere MQ header information is preserved in the RAW message body. MGW canonical type SYS.MGW_BASIC_MSG_T, however, has a header that can be used to specify WebSphere MQ header fields for outbound propagation, and preserve WebSphere MQ header fields for inbound propagation.

This section describes the message properties supported for the WebSphere MQ messaging system when using SYS.MGW_BASIC_MSG_T as the canonical type. Table C-8 defines the MGW {name, value} pairs used to describe the WebSphere MQ header properties. The first column refers to valid string values for the SYS.MGW_NAME_VALUE_T.NAME field in the SYS.MGW_BASIC_MSG_T header. The second column refers to the SYS.MGW_NAME_VALUE_T.TYPE value corresponding to the name. (Refer to "Notes on Table C-8" for explanations of the numbers in parentheses.)

For inbound propagation, the WebSphere MQ driver generates {name,value} pairs based on the source message header and stores them in the header part of the canonical message of the SYS.MGW_BASIC_MSG_T type. For outbound propagation, the WebSphere MQ driver sets the message header and enqueue options from {name,value} pairs for these properties stored in the header part of the SYS.MGW_BASIC_MSG_T canonical message.

Table C-8 MGW Names for WebSphere MQ Header Values

MGW Name	MGW Type	WebSphere MQ Property Name	Used For
MGW_MQ_accountingToken	RAW_VALUE (size 32)	accountingToken	Outbound (1), Inbound
MGW_MQ_applicationIdData	TEXT_VALUE (size 32)	applicationIdData	Outbound (1), Inbound
MGW_MQ_applicationOriginData	TEXT_VALUE (size 4)	applicationOriginData	Outbound (1), Inbound
MGW_MQ_backoutCount	INTEGER_VALUE	backoutCount	Inbound
MGW_MQ_characterSet	INTEGER_VALUE	characterSet	Outbound, Inbound
MGW_MQ_correlationId	RAW_VALUE (size 24)	correlationId	Outbound (1), Inbound
MGW_MQ_encoding	INTEGER_VALUE	encoding	Outbound, Inbound
MGW_MQ_expiry	INTEGER_VALUE	expiry	Outbound, Inbound
MGW_MQ_feedback	INTEGER_VALUE	feedback	Outbound, Inbound

Table C-8 (Cont.) MGW Names for WebSphere MQ Header Values

MGW Name	MGW Type	WebSphere MQ Property Name	Used For
`MGW_MQ_format`	`TEXT_VALUE (size 8)`	`format`	Outbound (1), Inbound
`MGW_MQ_groupId`	`RAW_VALUE (size 24)`	`groupId`	Outbound (1), Inbound
`MGW_MQ_messageFlags`	`INTEGER_VALUE`	`messageFlags`	Outbound, Inbound
`MGW_MQ_messageId`	`RAW_VALUE (size 24)`	`messageId`	Outbound, Inbound
`MGW_MQ_messageSequenceNumber`	`INTEGER_VALUE`	`messageSequenceNumber`	Outbound, Inbound
`MGW_MQ_messageType`	`INTEGER_VALUE`	`messageType`	Outbound, Inbound
`MGW_MQ_offset`	`INTEGER_VALUE`	`offset`	Outbound, Inbound
`MGW_MQ_originalLength`	`INTEGER_VALUE`	`originalLength`	Outbound, Inbound
`MGW_MQ_persistence`	`INTEGER_VALUE`	`persistence`	Inbound
`MGW_MQ_priority`	`INTEGER_VALUE`	`priority`	Outbound, Inbound
`MGW_MQ_putApplicationName`	`TEXT_VALUE (size 28)`	`putApplicationName`	Outbound (1), Inbound
`MGW_MQ_putApplicationType`	`INTEGER_VALUE`	`putApplicationType`	Outbound (1), Inbound
`MGW_MQ_putDateTime`	`DATE_VALUE`	`putDateTime`	Inbound
`MGW_MQ_putMessageOptions`	`INTEGER_VALUE`	`putMessageOptions`	Outbound (1) (2)
`MGW_MQ_replyToQueueManagerName`	`TEXT_VALUE (size 48)`	`replyToQueueManagerName`	Outbound, Inbound
`MGW_MQ_replyToQueueName`	`TEXT_VALUE (size 48)`	`replyToQueueName`	Outbound, Inbound
`MGW_MQ_report`	`INTEGER_VALUE`	`report`	Outbound (1), Inbound
`MGW_MQ_userId`	`TEXT_VALUE (size 12)`	`userId`	Outbound, Inbound

See Also:

"DBMS_MGWMSG" in *Oracle Database PL/SQL Packages and Types Reference*

Notes on Table C-8

1. This use is subject to WebSphere MQ restrictions. For example, if `MGW_MQ_accountingToken` is set for an outgoing message, then WebSphere MQ overrides its value unless `MGW_MQ_putMessageOptions` is set to the WebSphere MQ constant `MQPMD_SET_ALL_CONTEXT`.
2. `MGW_MQ_putMessageOptions` is used as the `putMessageOptions` argument to the WebSphere MQ Base Java `Queue.put()` method. It is not part of the WebSphere MQ header information and is therefore not an actual message property.

   The value for the `openOptions` argument of the WebSphere MQ Base Java `MQQueueManager.accessQueue` method is specified when the WebSphere MQ queue is registered using the `DBMS_MGWADM.REGISTER_FOREIGN_QUEUE` call. Dependencies can exist between the two. For instance, for `MGW_MQ_putMessageOptions` to include `MQPMD_SET_ALL_CONTEXT, the MQ_openMessageOptions` queue option must include `MQOO_SET_CONTEXT`.

   The MGW agent adds the value `MQPMO_SYNCPOINT` to any value that you can specify.

MGW sets default values for two WebSphere MQ message header fields: `messageType` defaults to `MQMT_DATAGRAM` and `putMessageOptions` defaults to `MQPMO_SYNCPOINT`.

MGW provides two default mappings between Oracle Database Advanced Queuing message properties and WebSphere MQ header fields.

One maps the Oracle Database Advanced Queuing message property `expiration`, representing the time-to-live of the message at the time the message becomes available in the queue, to the WebSphere MQ header field `expiry`, representing the time-to-live of the message. For outbound propagation, the value used for `expiry` is determined by subtracting the time the message was available in the queue from the `expiration`, converted to tenths of a second. Oracle Database Advanced Queuing value `NEVER` is mapped to `MQEI_UNLIMITED`. For inbound propagation, the value of `expiration` is simply `expiry` converted to seconds. WebSphere MQ value `MQEI_UNLIMITED` is mapped to `NEVER`.

The other default maps Oracle Database Advanced Queuing message property `priority` with the WebSphere MQ header field `priority`. It is described in Table C-9.

Table C-9 Default Priority Mappings for Propagation

Propagation Type	Message System	Priority Values									
Outbound	Oracle Database Advanced Queuing	0	1	2	3	4	5	6	7	8	9
Outbound	WebSphere MQ	9	8	7	6	5	4	3	2	1	0
Inbound	Oracle Database Advanced Queuing	9	8	7	6	5	4	3	2	1	0
Inbound	WebSphere MQ	0	1	2	3	4	5	6	7	8	9

> **Note:**
>
> For outbound propagation, Oracle Database Advanced Queuing priority values less than 0 are mapped to WebSphere MQ priority 9, and Oracle Database Advanced Queuing priority values greater than 9 are mapped to WebSphere MQ priority 0.

WebSphere MQ Outbound Propagation

If no message transformation is provided for outbound propagation, then the Oracle Database Advanced Queuing source queue payload type must be either `SYS.MGW_BASIC_MSG_T` or `RAW`. If a message transformation is specified, then the target ADT of the transformation must be `SYS.MGW_BASIC_MSG_T`, but the source ADT can be any ADT supported by Oracle Database Advanced Queuing.

If the Oracle Database Advanced Queuing queue payload is RAW, then the resulting WebSphere MQ message has the message body set to the value of the RAW bytes and, by default, the `format` field set to the value "`MGW_Byte`".

If the Oracle Database Advanced Queuing queue payload or transformation target ADT is `SYS.MGW_BASIC_MSG_T`, then the message is mapped to a WebSphere MQ native message as follows:

- The WebSphere MQ fixed header fields are based on the internal Oracle Database Advanced Queuing message properties and the `SYS.MGW_BASIC_MSG_T.header` attribute of the canonical message, as described in "WebSphere MQ Message Header Mappings".
- If the canonical message has a `TEXT` body, then the WebSphere MQ format header field is set to `MQFMT_STRING` unless overridden by the header property `MGW_MQ_format`. The message body is treated as text.
- If the canonical message has a `RAW` body, then the WebSphere MQ format header field is set to "`MGW_Byte`" unless overridden by the header property `MGW_MQ_format`. The message body is treated as raw bytes.
- If the canonical message has both a `TEXT` and `RAW` body, then message conversion fails.
- If the canonical message has neither a `TEXT` nor `RAW` body, then no message body is set, and the WebSphere MQ format header field is `MQFMT_NONE`.
- If the canonical message has a `TEXT` body with both small and large values set (`SYS.MGW_BASIC_MSG_T.TEXT_BODY.small_value` and `SYS.MGW_BASIC_MSG_T.TEXT_BODY.large_value` not empty), then message conversion fails.
- If the canonical message has a `RAW` body with both small and large values set (`SYS.MGW_BASIC_MSG_T.RAW_BODY.small_value` and `SYS.MGW_BASIC_MSG_T.RAW_BODY.large_value` not empty), then message conversion fails.

If the job option `PreserveMessageID` is specified with a value of `TRUE`, then the `correlationId` field of the WebSphere message header will be set to the AQ source message identifier. The `correlationId` value will be a 24-byte value of the form `"AQMSGID:"+`*`AQ_msgid`* where *`AQ_msgid`* represents the 16-byte Database AQ message identifier.

WebSphere MQ Inbound Propagation

If no message transformation is provided for inbound propagation, then the Oracle Database Advanced Queuing destination queue payload type must be either `SYS.MGW_BASIC_MSG_T` or `RAW`. If a message transformation is specified, then the source ADT of the transformation must be `SYS.MGW_BASIC_MSG_T`, but the destination ADT can be any ADT supported by Oracle Database Advanced Queuing.

If the Oracle Database Advanced Queuing queue payload is `RAW` and the incoming WebSphere MQ message has a `format` of `MQFMT_STRING`, then message conversion fails. Otherwise the message body is considered as raw bytes and enqueued directly to the destination queue. If the number of bytes is greater than 32KB, then message conversion fails. The actual limit is 32512 bytes rather than 32767 bytes.

If the Oracle Database Advanced Queuing queue payload or transformation source ADT is `SYS.MGW_BASIC_MSG_T`, then the WebSphere MQ message is mapped to a `SYS.MGW_BASIC_MSG_T` message as follows:

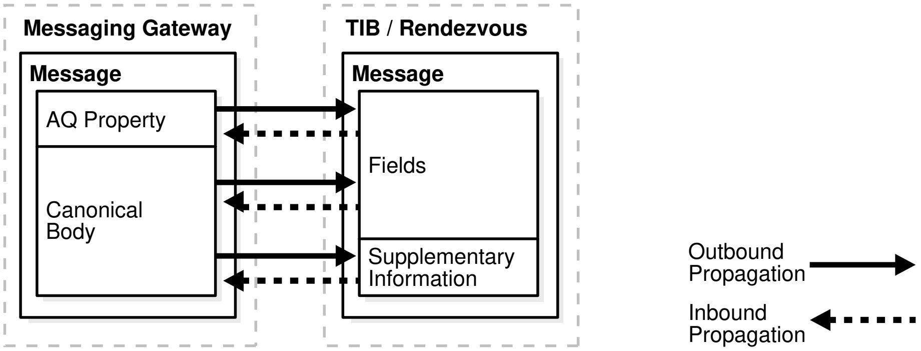
- Specific WebSphere MQ header fields are mapped to Oracle Database Advanced Queuing message properties as previously described.
- The `SYS.MGW_BASIC_MSG_T.header` attribute of the canonical message is set to {name, value} pairs based on the WebSphere MQ header fields, as described in Table C-8. These values preserve the original content of the WebSphere MQ message header.
- If the WebSphere MQ `format` header field is `MQFMT_STRING`, then the WebSphere MQ message body is treated as text, and its value is mapped to `SYS.MGW_BASIC_MSG_T.text_body`. For any other `format` value, the message body is treated as raw bytes, and its value is mapped to `SYS.MGW_BASIC_MSG_T.raw_body`.

> **See Also:**
>
> "WebSphere MQ Message Header Mappings"

Message Conversion for TIB/Rendezvous

MGW regards a TIB/Rendezvous message as a set of fields and supplementary information. Figure C-5 shows how messages are converted between MGW and TIB/Rendezvous.

Figure C-5 Message Conversion for TIB/Rendezvous



When a message conversion failure occurs, messages are moved to an exception queue (if one has been provided), so that MGW can continue propagation of the remaining messages in the source queue. In inbound propagation from TIB/Rendezvous, an exception queue is a registered subject.

All TIB/Rendezvous wire format datatypes for TIB/Rendezvous fields are supported, except for the datatypes with unsigned integers and the nested message type. User-defined custom datatypes are not supported in this release. If a message contains data of the unsupported datatypes, then a message conversion failure occurs when the message is processed. A message conversion failure results in moving the failed message from the source queue to the exception queue, if an exception queue is provided.

Table C-10 shows the datatype mapping used when MGW converts between a native TIB/Rendezvous message and the canonical ADT. For each supported TIB/Rendezvous wire format type, it shows the Oracle type used to store the data and the `DBMS_MGWMSG` constant that represents that type.

Table C-10 TIB/Rendezvous Datatype Mapping

TIB/Rendezvous Wire Format	Oracle Type	ADT Field Type
`Bool`	`NUMBER`	`TIBRVMSG_BOOL`
`F32`	`NUMBER`	`TIBRVMSG_F32`
`F64`	`NUMBER`	`TIBRVMSG_F64`
`I8`	`NUMBER`	`TIBRVMSG_I8`
`I16`	`NUMBER`	`TIBRVMSG_I16`
`I32`	`NUMBER`	`TIBRVMSG_I32`
`I64`	`NUMBER`	`TIBRVMSG_I64`
`U8`	not supported	not supported
`U16`	not supported	not supported
`U32`	not supported	not supported
`U64`	not supported	not supported
`IPADDR32`	`VARCHAR2`	`TIBRVMSG_IPADDR32`
`IPPORT16`	`NUMBER`	`TIBRVMSG_IPPORT16`
`DATETIME`	`DATE`	`TIBRVMSG_DATETIME`
`F32ARRAY`	`SYS.MGW_NUMBER_ARRAY_T`	`TIBRVMSG_F32ARRAY`
`F64ARRAY`	`SYS.MGW_NUMBER_ARRAY_T`	`TIBRVMSG_F64ARRAY`
`I8ARRAY`	`SYS.MGW_NUMBER_ARRAY_T`	`TIBRVMSG_I8ARRAY`
`I16ARRAY`	`SYS.MGW_NUMBER_ARRAY_T`	`TIBRVMSG_I16ARRAY`
`I32ARRAY`	`SYS.MGW_NUMBER_ARRAY_T`	`TIBRVMSG_I32ARRAY`
`I64ARRAY`	`SYS.MGW_NUMBER_ARRAY_T`	`TIBRVMSG_I64ARRAY`
`U8ARRAY`	not supported	not supported
`U16ARRAY`	not supported	not supported
`U32ARRAY`	not supported	not supported
`U64ARRAY`	not supported	not supported
`MSG`	not supported	not supported

Table C-10 (Cont.) TIB/Rendezvous Datatype Mapping

TIB/Rendezvous Wire Format	Oracle Type	ADT Field Type
OPAQUE	RAW or BLOB	TIBRVMSG_OPAQUE
STRING	VARCHAR2 or CLOB	TIBRVMSG_STRING
XML	RAW or BLOB	TIBRVMSG_XML

For propagation between Oracle Database Advanced Queuing and TIB/Rendezvous, MGW provides direct support for the Oracle Database Advanced Queuing payload types RAW and SYS.MGW_TIBRV_MSG_T. To support any other Oracle Database Advanced Queuing payload type, you must supply a transformation.

AQ Message Property Mapping for TIB/Rendezvous

This section describes the mapping between Oracle Database AQ message properties and TIB/Rendezvous fields. This mapping is used to preserve Database AQ message properties during outbound propagation, and set Database AQ message properties during inbound propagation.

Table C-11 describes the Database AQ message properties supported using TIB/Rendezvous fields. The first column indicates the DBMS_AQ.MESSAGE_PROPERTIES_T field for the Database AQ message property. The second and third columns indicate the name and datatype used for the TIB/Rendezvous field. The last column indicates if the message property is supported for inbound and outbound propagation.

Table C-11 TIB/Rendezvous and MGW Names for Oracle Database Advanced Queuing Message Properties

Oracle Database Advanced Queuing Message Property	MGW Name	TIB/Rendezvous Wire Format Datatype	Used For
priority	MGW_AQ_priority	TibrvMsg.I32	Outbound, Inbound
expiration	MGW_AQ_expiration	TibrvMsg.I32	Outbound, Inbound
delay	MGW_AQ_delay	TibrvMsg.I32	Outbound, Inbound
correlation	MGW_AQ_correlation	TibrvMsg.STRING	Outbound, Inbound
exception_queue	MGW_AQ_exception_queue	TibrvMsg.STRING	Outbound, Inbound
enqueue_time	MGW_AQ_enqueue_time	TibrvMsg.DATETIME	Outbound
original_msgid	MGW_AQ_original_msgid	TibrvMsg.OPAQUE	Outbound
msgid (1)	MGW_AQ_messageID	TibrvMsg.OPAQUE	Outbound

Notes on Table C-11:

1. The msgid Database AQ property represents the Database AQ message identifier, rather than a particular field of the DBMS_AQ.MESSAGE_PROPERTIES_T record.

TIB/Rendezvous Outbound Propagation

If no propagation transformation is provided for outbound propagation, then the Oracle Database Advanced Queuing source queue payload type must be either `SYS.MGW_TIBRV_MSG_T` or `RAW`. If a propagation transformation is specified, then the target ADT of the transformation must be `SYS.MGW_TIBRV_MSG_T`, but the source ADT can be any ADT supported by Oracle Database Advanced Queuing.

If the Oracle Database Advanced Queuing queue payload or transformation target ADT is `SYS.MGW_TIBRV_MSG_T`, then:

- Every field in the source message is converted to a TIB/Rendezvous message field of the resulting TIB/Rendezvous message.
- If the `reply_subject` attribute is not `NULL`, then the reply subject supplementary information is set.
- The `send_subject` field is ignored.

If the Oracle Database Advanced Queuing queue payload is `RAW`, then:

- The resulting message contains a field named `MGW_RAW_MSG` with value `TibrvMsg.OPAQUE`. The field ID is set to 0.

If the job option `AQ_MsgProperties` is specified with a value of `TRUE`, then the MGW agent generates fields to preserve the Database AQ message properties in the TIB/Rendezvous message according to Table C-11.

If the `PreserveMessageID` job option is specified with a value of `TRUE`, then the Database AQ message identifier (`msgid`) is preserved in the TIB/Rendezvous message according to Table C-11.

TIB/Rendezvous Inbound Propagation

If no propagation transformation is provided for inbound propagation, then the Oracle Database Advanced Queuing destination queue payload type must be either `RAW` or `SYS.MGW_TIBRV_MSG_T`. If a propagation transformation is specified, then the target ADT of the transformation can be any ADT supported by Oracle Database Advanced Queuing, but the source ADT of the transformation must be `SYS.MGW_TIBRV_MSG_T`.

If the Oracle Database Advanced Queuing queue payload or transformation source ADT is `SYS.MGW_TIBRV_MSG_T`, then:

- Every field in the source TIB/Rendezvous message is converted to a field of the resulting message of the `SYS.MGW_TIBRV_MSG_T` type.
- The MGW agent extracts the send subject name from the source TIB/Rendezvous message and sets the `send_subject` attribute in `SYS.MGW_TIBRV_MSG_T`. The send subject name is usually the same as the subject name of the registered propagation source queue, but it might be different when wildcards are used.
- The MGW agent extracts the reply subject name from the source TIB/Rendezvous message, if it exists, and sets the `reply_subject` attribute in `SYS.MGW_TIBRV_MSG_T`.
- If the source TIB/Rendezvous message contains more than three large text fields (greater than 4000 bytes of text) or more than three large bytes fields (greater than 2000 bytes), then message conversion fails.

If the Oracle Database Advanced Queuing queue payload is `RAW`, then:

- The Oracle Database Advanced Queuing message payload is the field data if the source TIB/Rendezvous message has a field named `MGW_RAW_MSG` of type `TibrvMsg.OPAQUE` or `TibrvMsg.XML`. The field name and ID are ignored. If no such field exists or has an unexpected type, then a message conversion failure occurs.
- A message conversion failure occurs if the `RAW` data size is greater than 32KB. This is due to a restriction on the data size allowed for a bind variable. Also, the actual limit is 32512 rather than 32767.

If the job option `AQ_MsgProperties` is specified with a value of `TRUE`, then the MGW agent searches for fields in the original TIB/Rendezvous messages with reserved field names. Table C-11 shows the field name strings and the corresponding values used in the TIB/Rendezvous message.

If such fields exist, then the MGW agent uses the field value to set the corresponding Oracle Database Advanced Queuing message properties, instead of using the default values. If there is more than one such field with the same name, then only the first one is used. Such fields are removed from the resulting payload only if the Oracle Database Advanced Queuing queue payload is `RAW`. If a field with the reserved name does not have the expected datatype, then it causes a message conversion failure.

> **See Also:**
>
> "DBMS_MGWMSG" in *Oracle Database PL/SQL Packages and Types Reference* for the value datatypes

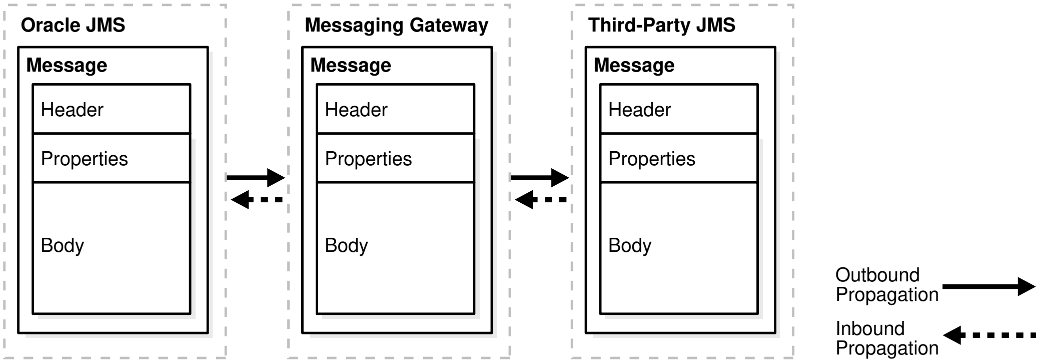
JMS Messages

MGW propagates only JMS messages between Oracle JMS and non-Oracle JMS systems, without changing the message content. Figure C-6 shows JMS message propagation.

MGW supports only the standard JMS message types. It does not support:

- JMS provider extensions, because any such extensions would not be recognized by the destination JMS system. An attempt to propagate any such non-JMS message results in an error.
- User transformations for JMS propagation.
- Propagation of Logical Change Records (LCRs).

Figure C-6 JMS Message Propagation



For the purposes of this discussion, a JMS message is a Java object of a class that implements one of the five JMS message interfaces. Table C-12 shows the JMS message interfaces and the corresponding Oracle JMS ADTs. The table also shows the interface, `javax.jms.Message`, which can be any one of the five specific types, and the corresponding generic Oracle JMS type `SYS.AQ$_JMS_MESSAGE`.

Table C-12 Oracle JMS Message Conversion

JMS Message	ADT
javax.jms.TextMessage	SYS.AQ$_JMS_TEXT_MESSAGE
javax.jms.BytesMessage	SYS.AQ$_JMS_BYTES_MESSAGE
javax.jms.MapMessage	SYS.AQ$_JMS_MAP_MESSAGE
javax.jms.StreamMessage	SYS.AQ$_JMS_STREAM_MESSAGE
javax.jms.ObjectMessage	SYS.AQ$_JMS_OBJECT_MESSAGE
javax.jms.Message	SYS.AQ$_JMS_MESSAGE

When a propagation job is activated, the MGW agent checks the Oracle Database Advanced Queuing payload type for the propagation source or destination. If the type is one of those listed in Table C-12 or `ANYDATA`, then message propagation is attempted. Otherwise an exception is logged and propagation is not attempted.

The MGW agent may add a JMS `String` property named `OracleMGW_OriginalMessageID` to the JMS message sent to the destination queue in order to preserve the original message identifier of the source message. This property is added if the `PreserveMessageID` job option is specified with a value of `TRUE`. It will also be added for any message moved to an exception queue upon a message conversion failure.

JMS Outbound Propagation

When dequeuing a message from an Oracle Database Advanced Queuing queue, Oracle JMS converts instances of the ADTs shown in Table C-12 into JMS messages. In addition it can convert instances of `ANYDATA` into JMS messages, depending on the content.

A queue with payload type `ANYDATA` can hold messages that do not map to a JMS message. MGW fails to dequeue such a message. An error is logged and propagation of messages from that queue does not continue until the message is removed.

JMS Inbound Propagation

Every message successfully dequeued using WebSphere MQ JMS is a JMS message. No message conversion is necessary prior to enqueuing using Oracle JMS. However, if the payload ADT of the propagation destination does not accept the type of the inbound message, then an exception is logged and an attempt is made to place the message in an exception queue. An example of such type mismatches is a JMS `TextMessage` and a queue payload type `SYS.AQ$_JMS_BYTES_MESSAGE`.

Monitoring Oracle Messaging Gateway

The following topics discuss means of monitoring the Oracle Messaging Gateway (MGW) agent, abnormal situations you may experience, several sources of information about Messaging Gateway errors and exceptions, and suggested remedies.

- Oracle Messaging Gateway Log Files
- Monitoring the Oracle Messaging Gateway Agent Status
- Monitoring Oracle Messaging Gateway Propagation
- Oracle Messaging Gateway Agent Error Messages

Oracle Messaging Gateway Log Files

Messaging Gateway agent status, history, and errors are recorded in Messaging Gateway log files. A different log file is created each time the Messaging Gateway agent is started. You should monitor the log file because any errors, configuration information read at startup time, or dynamic configuration information is written to the log.

The format of the log file name for the default agent is:

```
oramgw-hostname-timestamp-processid.log
```

The format of the log file name for a named agent is:

```
oramgw-AGENTNAME-hostname-timestamp-processid.log
```

By default the Messaging Gateway log file is in `ORACLE_HOME/mgw/log`. This location can overridden by the parameter `log_directory` in the Messaging Gateway initialization file used by the agent, usually `mgw.ora`.

This section contains these topics:

- Sample Oracle Messaging Gateway Log File
- Interpreting Exception Messages in an Oracle Messaging Gateway Log File

Sample Oracle Messaging Gateway Log File

The following sample log file shows the Messaging Gateway agent starting. The sample log file shows that a messaging link, a registered foreign queue, a propagation job, and a schedule associated with the job have been added. The log file shows that the propagation job has been activated. The last line indicates that the Messaging Gateway is up and running and ready to propagate messages.

Example C-36 Sample Messaging Gateway Log File

```
>>2007-01-16 15:04:49 MGW C-Bootstrap 0 LOG process-id=11080
Bootstrap program starting
>>2007-01-16 15:04:50 MGW C-Bootstrap 0 LOG process-id=11080
JVM created -- heapsize = 64
>>2007-01-16 15:04:53 MGW Engine 0 200 main
MGW Agent version: 11.1.0.0
>>2007-01-16 15:04:53 MGW AdminMgr 0 LOG main
Connecting to database using connect string = jdbc:oracle:oci:@INST1
>>2007-01-16 15:05:00 MGW Engine 0 200 main
MGW Component version: 11.1.0.3.0
>>2007-01-16 15:05:01 MGW Engine 0 200 main
MGW agent name: DEFAULT_AGENT, MGW job instance id: 273006EC6ED255F1E040578C6D021A8C,
MGW database instance: 1
>>2007-01-16 15:05:09 MGW Engine 0 1 main
Agent is initializing.
>>2007-01-16 15:05:09 MGW Engine 0 23 main
The number of worker threads is set to 1.
>>2007-01-16 15:05:09 MGW Engine 0 22 main
The default polling interval is set to 5000ms.
```

```
>>2007-01-16 15:05:09 MGW MQD 0 LOG main
Creating MQSeries messaging link:
link : MQLINK
link type : Base Java interface
queue manager : my.queue.manager
channel : channel1
host : my.machine
port : 1414
user :
ccdt url :
ssl cipherSuite :
connections : 1
inbound logQ : logq1
outbound logQ : logq2
>>2007-01-16 15:05:09 MGW Engine 0 4 main
Link MQLINK has been added.
>>2007-01-16 15:05:09 MGW Engine 0 7 main
Queue DESTQ@MQLINK has been registered; provider queue: MGWUSER.MYQUEUE.
>>2007-01-16 15:05:09 MGW Engine 0 9 main
Propagation Schedule JOB_AQ2MQ (MGWUSER.MGW_BASIC_SRC --> DESTQ@MQLINK) has been
added.
>>2007-01-16 15:05:09 MGW AQN 0 LOG main
Creating AQ messaging link:
link : oracleMgwAq
link type : native
database : INST1
user : MGWAGENT
connection type : JDBC OCI
connections : 1
inbound logQ : SYS.MGW_RECV_LOG
outbound logQ : SYS.MGW_SEND_LOG
>>2007-01-16 15:05:10 MGW Engine 0 19 main
MGW propagation job JOB_AQ2MQ has been activated.
>>2007-01-16 15:05:10 MGW Engine 0 14 main
MGW propagation job JOB_AQ2MQ (MGWUSER.MGW_BASIC_SRC --> DESTQ@MQLINK) has been added.
>>2007-01-16 15:05:11 MGW Engine 0 2 main
Agent is up and running.
```

Interpreting Exception Messages in an Oracle Messaging Gateway Log File

Exception messages logged to the Messaging Gateway log file may include one or more linked exceptions, identified by `[Linked-exception]` in the log file. These are often the most useful means of determining the cause of a problem. For instance, a linked exception could be a `java.sql.SQLException`, possibly including an Oracle error message, a PL/SQL stack trace, or both.

The following example shows entries from a Messaging Gateway log file when an invalid value (`bad_service_name`) was specified for the `database` parameter of `DBMS_MGWADM.CREATE_AGENT` or `DBMS_MGWADM.ALTER_AGENT`. This resulted in the Messaging Gateway agent being unable to establish database connections.

Example C-37 Sample Exception Message

```
>>2003-07-22 15:27:26  MGW  AdminMgr  0  LOG  main
Connecting to database using connect string = jdbc:oracle:oci8:@BAD_SERVICE_NAME
>>2003-07-22 15:27:29  MGW  Engine  0  EXCEPTION  main
oracle.mgw.admin.MgwAdminException: [241]  Failed to connect to database. SQL
error: 12154, connect string: jdbc:oracle:oci8:@BAD_SERVICE_NAME
[ …Java stack trace here…]
[Linked-exception]
java.sql.SQLException: ORA-12154: TNS:could not resolve the connect identifier
```

```
specified
[ …Java stack trace here…]
>>2003-07-22 15:27:29  MGW  Engine  0  25  main
Agent is shutting down.
```

Monitoring the Oracle Messaging Gateway Agent Status

This section contains these topics:

- MGW_GATEWAY View
- Oracle Messaging Gateway Irrecoverable Error Messages
- Other Oracle Messaging Gateway Error Conditions

MGW_GATEWAY View

The `MGW_GATEWAY` view monitors the progress of the Messaging Gateway agent. Among the fields that can be used to monitor the agent are:

- `AGENT_NAME`
- `AGENT_INSTANCE`
- `AGENT_PING`
- `AGENT_STATUS`
- `LAST_ERROR_MSG`
- `SERVICE`

The `AGENT_STATUS` field shows the status of the agent. This column has the following possible values:

NOT_STARTED
Indicates that the agent is neither running nor scheduled to be run.

START_SCHEDULED
Indicates that the agent job is waiting to be run by the job scheduler.

STARTING
Indicates that the agent is in the process of starting.

INITIALIZING
Indicates that the agent has started and is reading configuration data.

RUNNING
Indicates that the agent is ready to propagate any available messages or process dynamic configuration changes.

SHUTTING_DOWN
Indicates that the agent is in the process of shutting down.

BROKEN
Indicates that, while attempting to start an agent process, Messaging Gateway has detected another agent already running. This situation should never occur under normal usage.

Querying the `AGENT_PING` field pings the Messaging Gateway agent. Its value is either `REACHABLE` or `UNREACHABLE`. An agent with status of `RUNNING` should almost always be `REACHABLE`.

The columns LAST_ERROR_MSG, LAST_ERROR_DATE, and LAST_ERROR_TIME give valuable information if an error in starting or running the Messaging Gateway agent occurs. AGENT_INSTANCE indicates the Oracle Database instance on which the Messaging Gateway instance was started.

> **See Also:**
>
> "DBMS_MGWADM" in *Oracle Database PL/SQL Packages and Types Reference* for more information on the MGW_GATEWAY view

Oracle Messaging Gateway Irrecoverable Error Messages

A status of NOT_STARTED in the AGENT_STATUS field of the MGW_GATEWAY view indicates that the Messaging Gateway agent is not running. If the AGENT_STATUS is NOT_STARTED and the LAST_ERROR_MSG field is not NULL, then the Messaging Gateway agent has encountered an irrecoverable error while starting or running. Check if a Messaging Gateway log file has been generated and whether it indicates any errors. If a log file is not present, then the Messaging Gateway agent process was probably not started.

This section describes the causes and solutions for some error messages that may appear in the LAST_ERROR_MSG field of the MGW_GATEWAY view. Unless indicated otherwise, the Messaging Gateway agent will not attempt to restart itself when one of these errors occurs.

ORA-01089: Immediate shutdown in progress - no operations are permitted

The Messaging Gateway agent has shut down because the SHUTDOWN IMMEDIATE command was used to shut down a running Oracle Database instance on which the agent was running. The agent will restart itself on the next available database instance on which it is set up to run.

ORA-06520: PL/SQL: Error loading external library

The Messaging Gateway agent process was unable to start because the shared library was not loaded. This may be because the Java shared library was not in the library path. Verify that the library path in listener.ora has been set correctly.

ORA-28575: Unable to open RPC connection to external procedure agent

The Messaging Gateway agent was unable to start. It will attempt to start again automatically.

Possible causes include:

- The listener is not running. If you have modified listener.ora, then you must stop and restart the listener before the changes will take effect.
- Values in tnsnames.ora, listener.ora, or both are not correct.

  In particular, tnsnames.ora must have a net service name entry of MGW_AGENT. This entry is not needed for Messaging Gateway on Windows. The SID value specified for CONNECT_DATA of the MGW_AGENT net service name in tnsnames.ora must match the SID_NAME value of the SID_DESC entry in listener.ora. If the MGW_AGENT net service name is set up for an Inter-process Communication (IPC) connection, then the KEY values for ADDRESS in tnsnames.ora and listener.ora must match. If the names.default_domain parameter for sqlnet.ora has been used to set a default domain, then that domain must be appended to the MGW_AGENT net service name in tnsnames.ora.

ORA-28576: Lost RPC connection to external procedure agent

The Messaging Gateway agent process ended prematurely. This may be because the process was stopped by an outside entity or because an internal error caused a malfunction. The agent will attempt to start again automatically. Check the Messaging Gateway log file to determine if further information is available. If the problem persists, then contact Oracle Support Services for assistance.

ORA-32830: Result code -2 returned by Messaging Gateway agent

An error occurred when the Messaging Gateway agent tried to read its initialization file, usually `mgw.ora`. Verify that the file is readable.

ORA-32830: Result code -3 returned by Messaging Gateway agent

An error occurred creating the Messaging Gateway log file. Verify that the log directory can be written to. The default location is *`ORACLE_HOME`*`/mgw/log`.

ORA-32830: Result code -8 returned by Messaging Gateway agent

An error occurred starting the Java Virtual Machine (JVM). Verify that:

- You are using the correct Java version
- Your operating system version and patch level are sufficient for the JDK version
- You are using a reasonable value for the JVM heap size

  The heap size is specified by the `max_memory` parameter of `DBMS_MGWADM.ALTER_AGENT`
- On Windows platforms, verify the `MGW_PRE_PATH` set in `mgw.ora` contains the path to the correct JVM library (`jvm.dll`).

ORA-32830: Result code -12 returned by Messaging Gateway agent

An error occurred writing to the Messaging Gateway log file. Check the free disk space or any other issues that might result in file I/O problems.

ORA-32830: Result code -17 returned by Messaging Gateway agent

The JVM was successfully created but an error occurred trying to call the MGW Java agent program. Verify that the CLASSPATH set in `mgw.ora` is correct.

ORA-32830: Result code -19 returned by Messaging Gateway agent

The Messaging Gateway agent was configured to use a particular initialization file but that file does not exist. The `INITFILE` field of the `MGW_GATEWAY` view shows the full pathname of the file specified by the administrator. Either create that initialization file, or use `DBMS_MGWADM.ALTER_AGENT` to set `INITFILE` to another file or `NULL` to use the default initialization file.

ORA-32830: Result code -100 returned by Messaging Gateway agent

The Messaging Gateway agent JVM encountered a runtime exception or error on startup before it could write to the log file.

ORA-32830: Result code -101 returned by Messaging Gateway agent

An irrecoverable error caused the Messaging Gateway agent to shut down. Check the Messaging Gateway log file for further information. Verify that the values specified in `mgw.ora` are correct. Incorrect values can cause the Messaging Gateway agent to terminate due to unusual error conditions.

ORA-32830: Result code -102 returned by Messaging Gateway agent

The Messaging Gateway agent shut down because the version of file `ORACLE_HOME/mgw/jlib/mgw.jar` does not match the version of the Messaging Gateway PL/SQL packages. Verify that all Messaging Gateway components are from the same release.

ORA-32830: Result code -103 returned by Messaging Gateway agent

The Messaging Gateway agent shut down because the database instance on which it was running was shutting down. The agent should restart automatically, either on another instance if set up to do so, or when the instance that shut down is restarted.

ORA-32830: Result code -104 returned by Messaging Gateway agent

See previous error.

ORA-32830: Result code -105 returned by Messaging Gateway agent

The Messaging Gateway agent detected that it was running when it should not be. This should not happen. If it does, `AGENT_STATUS` will be `BROKEN` and the agent will shut down automatically. If you encounter this error:

- Terminate any Messaging Gateway agent process that may still be running. The process is usually named `extprocmgwextproc`.
- Run `DBMS_MGWADM.CLEANUP_GATEWAY(DBMS_MGWADM.CLEAN_STARTUP_STATE)`.
- Start the Messaging Gateway agent using `DBMS_MGWADM.STARTUP`.

ORA-32830: Result code -106 returned by Messaging Gateway agent

See previous error.

> **See Also:**
>
> "DBMS_MGWADM" in *Oracle Database PL/SQL Packages and Types Reference*

Other Oracle Messaging Gateway Error Conditions

This section discusses possible causes for `AGENT_STATUS` remaining `START_SCHEDULED` in `MGW_GATEWAY` view for an extended period.

Database Service Not Started

Messaging Gateway uses an Oracle Scheduler job to start the Messaging Gateway agent. Oracle Scheduler allows you to specify a database service under which a job should be run (service affinity). Messaging Gateway allows an administrator to configure the Messaging Gateway agent with a database service that will be used to configure the Scheduler job class associated with that agent.

When you shutdown a database Oracle stops all services to that database. You may need to manually restart the services when you start the database. If a Scheduler job is associated with a service then the job will not run until the service is started. If `AGENT_STATUS` for a Messaging

Gateway agent remains START_SCHEDULED for an extended period that might indicate that the database service is disabled or no database instances associated with the service are running. Use the MGW_GATEWAY view, Oracle Scheduler views, and service views to determine how the agent was configured and the current state of the Scheduler job and database service.

> **Note:**
>
> Oracle Messaging Gateway Agent Scheduler Job for information about Oracle Scheduler objects used by Messaging Gateway.

Too Few Job Queue Processes

Messaging Gateway uses Oracle Scheduler to start the Messaging Gateway external process. When AGENT_STATUS is START_SCHEDULED, the Messaging Gateway agent Scheduler job is waiting to be run by the Scheduler. The Messaging Gateway job will not run until there is an available job process. Messaging Gateway holds its Scheduler job process for the lifetime of the Messaging Gateway agent session. If multiple Messaging Gateway agents have been started, each agent uses its own Scheduler job and require its own job process.

If the value of the database initialization parameter JOB_QUEUE_PROCESSES is zero, then no Oracle Scheduler jobs will run. If the value is non-zero, it effectively becomes the maximum number of Scheduler jobs and job queue jobs than can concurrently run.

If Messaging Gateway status remains START_SCHEDULED for an extended period of time, then it may indicate that the database has been started with a value for JOB_QUEUE_PROCESSES that is zero or is too low and that all secondary job processes are busy. Verify that the value is non-zero and that the database instance has been started with enough job queue processes so that one is available for each Messaging Gateway agent.

Scheduler Job Broken or Disabled

The Messaging Gateway agent status will remain START_SCHEDULED if the Oracle Scheduler job associated with a Messaging Gateway agent has become disabled or broken for some reason. To determine if this is the case, use the DBA_SCHEDULER_JOBS view to look at STATE field for the agent's Scheduler job. Normally the Scheduler job state will be SCHEDULED when the Messaging Gateway agent's Scheduler job is waiting to be run, or RUNNING when the Messaging Gateway agent is running. The agent's Scheduler job should not exist if the Messaging Gateway agent status is NOT_STARTED.

Check other Scheduler views, such as DBA_SCHEDULER_JOB_RUN_DETAILS, for additional information about the Messaging Gateway Scheduler jobs. Also check the MGW_GATEWAY view and the Messaging Gateway log file for any error messages that may indicate a problem.

> **Note:**
>
> Oracle Messaging Gateway Agent Scheduler Job for information about Oracle Scheduler objects used by Messaging Gateway

Oracle Real Application Clusters (Oracle RAC) Environment

If Messaging Gateway is being used in an Oracle RAC environment and the agent has been configured with a database service but no database instances are running that have the service enabled, then the Messaging Gateway AGENT_STATUS will remain START_SCHEDULED until the service is started on a running database instance.

Monitoring Oracle Messaging Gateway Propagation

Messaging Gateway propagation can be monitored using the MGW_JOBS view and the Messaging Gateway log file. The view provides information on propagated messages and errors that may have occurred during propagation attempts. The log file can be used to determine the cause of the errors.

Besides showing configuration information, the MGW_JOBS view also has dynamic information that can be used to monitor message propagation. Applicable fields include STATUS, ENABLED, PROPAGATED_MSGS, EXCEPTIONQ_MSGS, FAILURES, LAST_ERROR_MSG, LAST_ERROR_DATE, and LAST_ERROR_TIME.

The STATUS field indicates current status of the job. READY means that the job is ready for propagation (but only if the ENABLED field is TRUE). RETRY means that a propagation failure occurred but that propagation will be retried. FAILED means that the agent has stopped propagation for the job due to an unrecoverable error or the maximum number of consecutive propagation failures has been reached. DELETE_PENDING means job removal is pending due to DBMS_MGWADM.REMOVE_JOB being called but certain cleanup tasks pertaining to the job are still outstanding. SUBSCRIBER_DELETE_PENDING means that DBMS_MGWADM.REMOVE_SUBSCRIBER has been called on an old style propagation job but certain cleanup tasks pertaining to the job are still outstanding.

The ENABLED field indicates whether the propagation job is currently enabled. TRUE indicates the job is enabled while FALSE indicates the job is disabled. No propagation will occur unless the job is enabled.

The PROPAGATED_MSGS field of the MGW_JOBS view indicates how many messages have been successfully propagated. This field is reset to zero when the Messaging Gateway agent is started.

If a Messaging Gateway propagation job has been configured with an exception queue, then the Messaging Gateway agent will move messages to that exception queue the first time the Messaging Gateway agent encounters a propagation failure caused by a message conversion failure. A message conversion failure is indicated by oracle.mgw.common.MessageException in the Messaging Gateway log file. The EXCEPTIONQ_MSGS field indicates how many messages have been moved to the exception queue. This field is reset to zero when the Messaging Gateway agent is started.

If an error occurs during message propagation for a propagation job, a count is incremented in the FAILURES field. This field indicates the number of failures encountered since the last successful propagation of messages. Each time a failure occurs, an error message and the time it occurred will be shown by LAST_ERROR_MSG, LAST_ERROR_DATE, and LAST_ERROR_TIME. When the number of failures reaches sixteen, Messaging Gateway halts propagation attempts for this propagation job. To resume propagation attempts you must call DBMS_MGWADM.RESET_JOB for the propagation job.

If an error occurs, then examine the Messaging Gateway log file for further information.

> **See Also:**
>
> "DBMS_MGWADM" in *Oracle Database PL/SQL Packages and Types Reference*

Oracle Messaging Gateway Agent Error Messages

This section lists some of the most commonly occurring errors that are shown in the `LAST_ERROR_MSG` column of the `MGW_JOBS` view and logged to the Messaging Gateway agent log file. Also shown are some errors that require special action. When you notice that a failure has occurred, look at the linked exceptions in the log file to determine the root cause of the problem.

Two primary types of errors are logged to the Messaging Gateway agent log file:

- `oracle.mgw.common.MessageException`

  This error type is logged when a message conversion failure occurs. The Messaging Gateway agent probably cannot propagate the message causing the failure, and the propagation job will eventually be stopped.
- `oracle.mgw.common.GatewayException`

  This error type is logged when some failure other than message conversion occurs. Depending on the cause, the problem may fix itself or require user action.

[221] Failed to access <*messaging_system*> queue: <*queue*>

An error occurred while trying to access either an Oracle Database Advanced Queuing queue or a non-Oracle queue. Check the linked exception error code and message in the log file.

[241] Failed to connect to database. SQL error: <*error*>, connect string: <*connect_string*>

This is probably caused by incorrect MGW agent connection information specified for `DBMS_MGWADM.CREATE_AGENT` or `DBMS_MGWADM.ALTER_AGENT`. Either the Messaging Gateway agent user or password is incorrect or the database specifier (database parameter) is incorrect. Verify that the connection information is correct for the connection type used by the agent, JDBC OCI or JDBC Thin.

If the `database` parameter is `NULL`, then check the Messaging Gateway log file for the following Oracle linked errors:

```
ORA-01034: ORACLE not available
ORA-27101: shared memory realm does not exist
```

These two errors together indicate that the Messaging Gateway agent is attempting to connect to the database using a local IPC connection, but the `ORACLE_SID` value is not correct.

A local connection is used when the database parameter is set to `NULL`. If a local connection is desired, the correct `ORACLE_SID` value must be set in the Messaging Gateway agent process. This can be done by adding the following line to the MGW initialization file, usually `mgw.ora`:

```
set ORACLE_SID = sid_value
```

`ORACLE_SID` need not be set in the MGW initialization file if the database parameter is not `NULL`.

If setting `ORACLE_SID` in the MGW initialization file does not work, then the database parameter must be set to a value that is not `NULL`.

If the JDBC Thin connection is used, then the database parameter must be not `NULL`. If the JDBC Thin connection is used and the database parameter is a TNSNames alias, make sure that the `oracle.net.tns_names` Java property is set in the MGW initialization file. The property can be set by adding the following line to the MGW initialization file:

```
setJavaProp oracle.net.tns_admin=<directory containing tnsnames.ora>
```

> **Note:**
>
> "oracle.net.tns_admin" for more information

[415] Missing messages from source queue of job <*job_name*>

Possible causes include:

- The agent partially processed persistent messages that were dequeued by someone other than the Messaging Gateway agent.
- The propagation source queue was purged or re-created.
- A message was moved to the Oracle Database Advanced Queuing exception queue.

If this error occurs, then call procedure `CLEANUP_GATEWAY` in the `DBMS_MGWADM` package:

```
DBMS_MGWADM.CLEANUP_GATEWAY (
       action => DBMS_MGWADM.RESET_SUB_MISSING_MESSAGE,
       sarg => <job_name>);
```

The call takes effect only if the propagation job has encountered the missing message problem and the agent is running. The agent treats the missing messages as nonpersistent messages and continues processing the propagation job.

> **See Also:**
>
> "Propagation Job Overview" for more information on Messaging Gateway exception queues

[416] Missing log records in receiving log queue for job <*job_name*>

Possible causes include:

- Log records were dequeued from the log queues by someone other than the Messaging Gateway agent.
- The log queues were purged or re-created.

If this error occurs, then call procedure `CLEANUP_GATEWAY` in the `DBMS_MGWADM` package:

```
DBMS_MGWADM.CLEANUP_GATEWAY (
       action => DBMS_MGWADM.RESET_SUB_MISSING_LOG_REC,
       sarg => <job_name>);
```

The call takes effect only if the propagation job has encountered the missing log records problem and the agent is running.

> **Note:**
>
> Calling procedure `DBMS_MGWADM.CLEANUP_GATEWAY` may result in duplicated messages if the missing messages have already been propagated to the destination queue. Users should check the source and destination queues for any messages that exist in both places. If such messages exist, then they should be removed from either the source or destination queue before calling this procedure.

[417] Missing log records in sending log queue for job <*job_name*>

See previous error.

[421] WARNING: Unable to get connections to recover job <*job_name*>

This message is a warning message indicating that the Messaging Gateway agent failed to get a connection to recover the propagation job, because other propagation jobs are using them all. The agent will keep trying to get a connection until it succeeds.

If this message is repeated many times for a WebSphere MQ link, then increase the maximum number of connections used by the Messaging Gateway link associated with the propagation job.

> **See Also:**
>
> "Altering a Messaging System Link"

[434] Failed to access queue <*queue*>; provider queue <*queue*>

This message indicates that a messaging system native queue cannot be accessed. The queue may have been registered by `DBMS_MGWADM.REGISTER_FOREIGN_QUEUE`, or it may be an Oracle Database Advanced Queuing queue. The linked exceptions should give more information.

Possible causes include:

- The foreign queue was registered incorrectly, or the Messaging Gateway link was configured incorrectly.

  Verify configuration information. If possible, use the same configuration information to run a sample application of the non-Oracle messaging system.
- The non-Oracle messaging system is not accessible.

  Check that the non-Oracle messaging system is running and can be accessed using the information supplied in the Messaging Gateway link.
- The Oracle Database Advanced Queuing queue does not exist. Perhaps the queue was removed after the Messaging Gateway propagation job was created.

  Check that the Oracle Database Advanced Queuing queue still exists.

[436] LOW MEMORY WARNING: total memory = < >, free_mem = < >

The Messaging Gateway agent JVM is running low on memory. Java garbage collection will be invoked, but this may represent a JVM heap size that is too small. Use the `max_memory` parameter of `DBMS_MGWADM.ALTER_AGENT` to increase the JVM heap size. If the Messaging Gateway agent is running, then it must be restarted for this change to take effect.

[703] Failed to retrieve information for transformation <*transformation_id*>

The Messaging Gateway agent could not obtain all the information it needs about the transformation. The transformation parameter of `DBMS_MGWADM.CREATE_JOB` must specify the name of the registered transformation and not the name of the transformation function.

Possible causes include:

- The transformation does not exist. Verify that the transformation has been created. You can see this from the following query performed as user SYS:

  ```
  SELECT TRANSFORMATION_ID, OWNER FROM DBA_TRANSFORMATIONS;
  ```

- The wrong transformation is registered with Messaging Gateway. Verify that the transformation registered is the one intended.
- The Messaging Gateway agent user does not have `EXECUTE` privilege on the object type used for the `from_type` or the `to_type` of the transformation indicated in the exception.

  It is not sufficient to grant `EXECUTE` to `MGW_AGENT_ROLE` and then grant `MGW_AGENT_ROLE` to the agent user. You must grant `EXECUTE` privilege on the object type directly to the agent user or to `PUBLIC`.

  The following example shows such a case for the `from_type`. It also shows the use of linked exceptions for determining the precise cause of the error.

  No EXECUTE Privilege on Object Type

  ```
  Errors occurred during processing of job JOB_AQ2MQ_2
  oracle.mgw.common.GatewayException: [703] Failed to retrieve information for
  transformation mgwuser.SAMPLEADT_TO_MGW_BASIC_MSG
  […Java stack trace here…]
  [Linked-exception]
  java.sql.SQLException: "from_type" is null
  […Java stack trace here…]
  ```

[720] AQ payload type <*type*> not supported; queue: <*queue*>

The payload type of the Oracle Database Advanced Queuing queue used by a Messaging Gateway propagation job is not directly supported by Messaging Gateway. For non-JMS propagation, Messaging Gateway directly supports the payload types `RAW`, `SYS.MGW_BASIC_MSG_T` and `SYS.MGW_TIBRV_MSG_T`.

Possible actions include:

- Configure the Messaging Gateway propagation job to use a transformation that converts the queue payload type to a supported type.
- Remove the Messaging Gateway propagation job and create a new job that uses an Oracle Database Advanced Queuing queue with a supported payload type.

  For Java Message Service (JMS) propagation, the Messaging Gateway propagation job must be removed and a new job created whose Oracle Database Advanced Queuing payload type is supported by Oracle Java Message Service (Oracle JMS). Transformations are not supported for JMS propagation.

[721] Transformation type <*type*> not supported; queue: <*queue_name*>, transform: <*transformation*>

A Messaging Gateway propagation job was configured with a transformation that uses an object type that is not one of the Messaging Gateway canonical types.

For an outbound job, the transformation `from_type` must be the Oracle Database Advanced Queuing payload type, and the `to_type` must be a Messaging Gateway canonical type. For an inbound job, the transformation `from_type` must be a Messaging Gateway canonical type and the `to_type` must be the Oracle Database Advanced Queuing payload type.

[722] Message transformation failed; queue: <*queue_name*>, transform: <*transformation*>

An error occurred while attempting execution of the transformation. ORA-25229 is typically thrown by Oracle Database Advanced Queuing when the transformation function raises a PL/SQL exception or some other Oracle error occurs when attempting to use the transformation.

Possible causes include:

- The Messaging Gateway agent user does not have `EXECUTE` privilege on the transformation function. This is illustrated in the following example.

  It is not sufficient to grant `EXECUTE` to `MGW_AGENT_ROLE` and then grant `MGW_AGENT_ROLE` to the Messaging Gateway agent user. You must grant `EXECUTE` privilege on the transformation function directly to the Messaging Gateway agent user or to `PUBLIC`.

  No EXECUTE Privilege on Transformation Function

  ```
  Errors occurred during processing of job JOB_MQ2AQ_2
  oracle.mgw.common.GatewayException: [722] Message transformation failed queue:
  MGWUSER.DESTQ_SIMPLEADT, transform: MGWUSER.MGW_BASIC_MSG_TO_SIMPLEADT
  […Java stack trace here…]
  [Linked-exception]
  oracle.mgw.common.MessageException: [722] Message transformation failed;
  queue: MGWUSER.DESTQ_SIMPLEADT, transform:
  MGWUSER.MGW_BASIC_MSG_TO_SIMPLEADT
  […Java stack trace here…]
  [Linked-exception]
  java.sql.SQLException: ORA-25229: error on transformation of message msgid:
  9749DB80C85B0BD4E03408002086745E
  ORA-00604: error occurred at recursive SQL level 1
  ORA-00904: invalid column name
  […Java stack trace here…]
  ```

- The transformation function does not exist, even though the registered transformation does. If the transformation function does not exist, it must be re-created.
- The Messaging Gateway agent user does not have `EXECUTE` privilege on the payload object type for the queue indicated in the exception.

  It is not sufficient to grant `EXECUTE` to `MGW_AGENT_ROLE` and then grant `MGW_AGENT_ROLE` to the Messaging Gateway agent user. You must grant `EXECUTE` privilege on the object type directly to the Messaging Gateway agent user or to `PUBLIC`.

- The transformation function raised the error. Verify that the transformation function can handle all messages it receives.

[724] Message conversion not supported; to AQ payload type: <*type*>, from type: <*type*>

A Messaging Gateway propagation job is configured for inbound propagation where the canonical message type generated by the non-Oracle messaging system link is not compatible with the Oracle Database Advanced Queuing queue payload type. For example, propagation from a TIB/Rendezvous messaging system to an Oracle Database Advanced Queuing queue with a `SYS.MGW_BASIC_MSG_T` payload type, or propagation from WebSphere MQ to an Oracle Database Advanced Queuing queue with a `SYS.MGW_TIBRV_MSG_T` payload type.

Possible actions include:

- Configure the Messaging Gateway propagation job with a transformation that maps the canonical message type generated by the non-Oracle messaging link to the Oracle Database Advanced Queuing payload type.
- Remove the Messaging Gateway propagation job and create a new job whose Oracle Database Advanced Queuing queue payload type matches the canonical message type generated by the non-Oracle link.

[725] Text message not supported for RAW payload

A Messaging Gateway propagation job is configured for inbound propagation to an Oracle Database Advanced Queuing destination having a `RAW` payload type. A text message was received from the source (non-Oracle) queue resulting in a message conversion failure.

If support for text data is required, remove the Messaging Gateway propagation job and create a new job to an Oracle Database Advanced Queuing destination whose payload type supports text data.

[726] Message size <*size*> too large for RAW payload; maximum size is <*size*>

A Messaging Gateway propagation job is configured for inbound propagation to an Oracle Database Advanced Queuing destination having a `RAW` payload type. A message conversion failure occurred when a message containing a large `RAW` value was received from the source (non-Oracle) queue.

If large data support is required, remove the Messaging Gateway propagation job and create a new job to an Oracle Database Advanced Queuing destination whose payload type supports large data, usually in the form of an object type with a BLOB attribute.

[728] Message contains too many large (BLOB) fields

The source message contains too many fields that must be stored in `BLOB` types. `SYS.MGW_TIBRV_MSG_T` is limited to three `BLOB` fields. Reduce the number of large fields in the message, perhaps by breaking them into smaller fields or combining them into fewer large fields.

[729] Message contains too many large (CLOB) fields

The source message contains too many fields that contain a large text value that must be stored in a `CLOB`. `SYS.MGW_TIBRV_MSG_T` is limited to three `CLOB` fields. Reduce the number of large fields in the message, perhaps by breaking them into smaller fields or combining them into fewer large fields.

[805] MQSeries Message error while enqueuing to queue: <*queue*>

WebSphere MQ returned an error when an attempt was made to put a message in a WebSphere MQ queue. Check the linked exception error code and message in the log file. Consult WebSphere MQ documentation.

Oracle Messaging Gateway Views

MGW_GATEWAY: Configuration and Status Information

This view lists configuration and status information for Messaging Gateway.

Table C-13 MGW_GATEWAY View Properties

Name	Type	Description
AGENT_DATABASE	VARCHAR2	The database connect string used by the Messaging Gateway agent. NULL indicates that a local connection is used.
AGENT_INSTANCE	NUMBER	The database instance on which the Messaging Gateway agent is currently running. This should be NULL if the agent is not running.
AGENT_JOB	NUMBER	[Deprecated] Job number of the queued job used to start the Messaging Gateway agent process. The job number is set when Messaging Gateway is started and cleared when it shuts down.
AGENT_NAME	VARCHAR2	Name of the Messaging Gateway agent
AGENT_PING	VARCHAR2	Gateway agent ping status. Values: • NULL means no ping attempt was made. • REACHABLE means ping attempt was successful. • UNREACHABLE means ping attempt failed. AGENT_PING attempts to contact the Messaging Gateway agent. There is a short delay (up to 5 seconds) if the ping attempt fails. No ping is attempted if the AGENT_STATUS is NOT_STARTED or START_SCHEDULED.
AGENT_START_TIME	TIMESTAMP	The time when the Messaging Gateway agent job currently running was started. This should be NULL if the agent is not running.

Table C-13 (Cont.) MGW_GATEWAY View Properties

Name	Type	Description
AGENT_STATUS	VARCHAR2	Status of the Messaging Gateway agent. Values: • NOT_STARTED means the Messaging Gateway agent has not been started • START_SCHEDULED means Messaging Gateway agent has been scheduled to start. That is, Messaging Gateway has been started using DBMS_MGWADM.STARTUP, but the queued job used to start the Messaging Gateway agent has not yet run. • STARTING means Messaging Gateway agent is starting. That is, Messaging Gateway has been started using DBMS_MGWADM.STARTUP, the queued job has run, and the Messaging Gateway agent is starting up. • INITIALIZING means the Messaging Gateway agent has started and is initializing • RUNNING means the Messaging Gateway agent is running • SHUTTING_DOWN means the Messaging Gateway agent is shutting down • BROKEN means an unexpected condition has been encountered that prevents the Messaging Gateway agent from starting. DBMS_MGWADM.CLEANUP_GATEWAY must be called before the agent can be started.
AGENT_USER	VARCHAR2	Database user name used by the Messaging Gateway agent to connect to the database
COMMENTS	VARCHAR2	Comments for the agent
CONNTYPE	VARCHAR2	Connection type used by the agent: • JDBC_OCI if the JDBC OCI driver is used • JDBC_THIN if the JDBC Thin driver is used
INITFILE	VARCHAR2	Name of the Messaging Gateway initialization file used by the agent. NULL indicates that the default initialization file is used.
LAST_ERROR_DATE	DATE	Date of last Messaging Gateway agent error. The last error information is cleared when Messaging Gateway is started. It is set if the Messaging Gateway agent fails to start or terminates due to an abnormal condition.
LAST_ERROR_MSG	VARCHAR2	Message for last Messaging Gateway agent error
LAST_ERROR_TIME	VARCHAR2	Time of last Messaging Gateway agent error
MAX_CONNECTIONS	NUMBER	[Deprecated] Maximum number of messaging connections to Oracle Database
MAX_MEMORY	NUMBER	Maximum heap size used by the Messaging Gateway agent (in MB)
MAX_THREADS	NUMBER	Maximum number of messaging threads created by the Messaging Gateway agent
SERVICE	VARCHAR2	Name of the database service that is associated with an Oracle Scheduler job class used by the agent

MGW_AGENT_OPTIONS: Supplemental Options and Properties

This view lists supplemental options and properties for a Messaging Gateway agent.

Table C-14 MGW_AGENT_OPTIONS View

Column	Type	Description
AGENT_NAME	VARCHAR2	Name of the Messaging Gateway agent
ENCRYPTED	VARCHAR2	Indicates whether the value is stored as encrypted: • TRUE if the value is stored encrypted • FALSE if the value is stored as cleartext
NAME	VARCHAR2	Name of the option
TYPE	VARCHAR2	Option type or usage: JAVA_SYSTEM_PROP if the option is used to set a Java System property
VALUE	VARCHAR2	Value for the option. This will be <<ENCRYPTED>> if the value is stored in an encrypted form.

MGW_LINKS: Names and Types of Messaging System Links

This view lists the names and types of messaging system links currently defined.

Table C-15 MGW_LINKS View Properties

Name	Type	Description
AGENT_NAME	VARCHAR2	Name of the Messaging Gateway agent that will process propagation jobs for this link
LINK_COMMENT	VARCHAR2	User comment for the link
LINK_NAME	VARCHAR2	Name of the messaging system link
LINK_TYPE	VARCHAR2	Type of messaging system link. Values • MQSERIES is for WebSphere MQ links. • TIBRV is for TIB/Rendezvous links.

MGW_MQSERIES_LINKS: WebSphere MQ Messaging System Links

This view lists information for the WebSphere MQ messaging system links. The view includes most of the messaging system properties specified when the link is created.

Table C-16 MGW_MQSERIES_LINKS View Properties

Name	Type	Description
AGENT_NAME	VARCHAR2	Name of the Messaging Gateway agent that will process propagation jobs for this link
CHANNEL	VARCHAR2	Connection channel
HOSTNAME	VARCHAR2	Name of the WebSphere MQ host
INBOUND_LOG_QUEUE	VARCHAR2	Inbound propagation log queue

Table C-16 (Cont.) MGW_MQSERIES_LINKS View Properties

Name	Type	Description
INTERFACE_TYPE	VARCHAR2	Messaging interface type. Values: • BASE_JAVA is for WebSphere MQ Base Java interface • JMS_CONNECTION is for WebSphere MQ JMS unified, domain-independent connections • JMS_QUEUE_CONNECTION is for WebSphere MQ JMS queue connections • JMS_TOPIC_CONNECTION is for WebSphere MQ JMS topic connections
LINK_COMMENT	VARCHAR2	User comment for the link
LINK_NAME	VARCHAR2	Name of the messaging system link
MAX_CONNECTIONS	NUMBER	Maximum number of messaging connections
OPTIONS	SYS.MGW_PROPERTIES	Link options
OUTBOUND_LOG_QUEUE	VARCHAR2	Outbound propagation log queue
PORT	NUMBER	Port number
QUEUE_MANAGER	VARCHAR2	Name of the WebSphere MQ queue manager

MGW_TIBRV_LINKS: TIB/Rendezvous Messaging System Links

This view lists information for TIB/Rendezvous messaging system links. The view includes most of the messaging system properties specified when the link was created.

Table C-17 MGW_TIBRV_LINKS View Properties

Property Name	Type	Description
AGENT_NAME	VARCHAR2	Name of the Messaging Gateway agent that will process propagation jobs for this link
CM_LEDGER	VARCHAR2	TIB/Rendezvous CM ledger file name
CM_NAME	VARCHAR2	TIB/Rendezvous CM correspondent name
DAEMON	VARCHAR2	TIB/Rendezvous daemon parameter for RVD transport
LINK_COMMENT	VARCHAR2	User comment for the link
LINK_NAME	VARCHAR2	Name of the messaging system link
NETWORK	VARCHAR2	TIB/Rendezvous network parameter for rvd transport
OPTIONS	SYS.MGW_PROPERTIES	Link options
SERVICE	VARCHAR2	TIB/Rendezvous service parameter for rvd transport

MGW_FOREIGN_QUEUES: Foreign Queues

This view lists information for foreign queues. The view includes most of the queue properties specified when the queue is registered.

Table C-18 MGW_FOREIGN_QUEUES View Properties

Name	Type	Description
DOMAIN	VARCHAR2	Queue domain type. Values: • NULL means the queue domain type is automatically determined by the messaging system • QUEUE is for a queue (point-to-point) model • TOPIC is for a topic (publish-subscribe) model
LINK_NAME	VARCHAR2	Name of the messaging system link
NAME	VARCHAR2	Name of the registered queue
OPTIONS	SYS.MGW_PROPERTIES	Optional queue properties
PROVIDER_QUEUE	VARCHAR2	Message provider (native) queue name
QUEUE_COMMENT	VARCHAR2	User comment for the foreign queue

MGW_JOBS: Messaging Gateway Propagation Jobs

This view lists information for Messaging Gateway propagation jobs. The view includes most of the job properties specified when the propagation job was created, as well as other status and statistical information.

Table C-19 MGW_JOBS View

Column	Type	Description
AGENT_NAME	VARCHAR2	Name of the Messaging Gateway agent that processes this job
COMMENTS	VARCHAR2	Comments for the propagation job
DESTINATION	VARCHAR2	Destination queue to which messages are propagated
ENABLED	VARCHAR2	Indicates whether the job is enabled or not: • TRUE if the job is enabled • FALSE if the job is disabled
EXCEPTION_QUEUE	VARCHAR2	Exception queue used for propagation logging purposes
EXCEPTIONQ_MSGS	NUMBER	Option type or usage: JAVA_SYSTEM_PROP if the option is used to set a Java System property
FAILURES	NUMBER	Number of messages moved to exception queue since the last time the agent was started
JOB_NAME	VARCHAR2	Name of the propagation job
LAST_ERROR_MSG	VARCHAR2	Message for the last propagation error
LAST_ERROR_DATE	DATE	Date of the last propagation error
LAST_ERROR_TIME	VARCHAR2	Time of the last propagation error
LINK_NAME	VARCHAR2	Name of the Messaging Gateway link used by this job
OPTIONS	SYS.MGW_PROPERTIES	Job options
POLL_INTERVAL	INTEGER	Propagation poll interval (in seconds)
PROPAGATED_MSGS	NUMBER	Number of messages propagated since the last time the agent was started

Table C-19 (Cont.) MGW_JOBS View

Column	Type	Description
PROP_STYLE	VARCHAR2	Message propagation style: • NATIVE for native message propagation • JMS for JMS message propagation
PROPAGATION_TYPE	VARCHAR2	Propagation type: • OUTBOUND is for Oracle Database AQ to non-Oracle propagation • INBOUND is for non-Oracle to Oracle Database AQ propagation
RULE	VARCHAR2	Subscription rule used for the propagation source
SOURCE	VARCHAR2	Source queue from which messages are propagated
STATUS	VARCHAR2	Job status: • READY means the job is ready for propagation. The job must be enabled and the Messaging Gateway agent running before messages are actually propagated. • RETRY means the agent encountered errors when attempting to propagate messages for the job and will retry the operation • FAILED means the job has failed and agent has stopped trying to propagate messages. Usually this is due to an unrecoverable error or the propagation failure limit being reached. The job must be reset before the agent will attempt to propagate messages. The job is automatically reset each time the agent is started and can be manually reset by DBMS_MGWADM.RESET_JOB. • DELETE_PENDING means that job removal is pending. DBMS_MGWADM.REMOVE_JOB has been called but certain cleanup tasks for this job are still outstanding. • SUBSCRIBER_DELETE_PENDING means that removal is pending for the subscriber associated with the job. DBMS_MGWADM.REMOVE_SUBSCRIBER has been called but certain cleanup tasks are still outstanding.
TRANSFORMATION	VARCHAR2	Transformation used for message conversion

MGW_SUBSCRIBERS: Information for Subscribers

This view lists configuration and status information for Messaging Gateway subscribers. The view includes most of the subscriber properties specified when the subscriber is added, as well as other status and statistical information.

Table C-20 MGW_SUBSCRIBERS View Properties

Name	Type	Description
DESTINATION	VARCHAR2	Destination queue to which messages are propagated
EXCEPTIONQ_MSGS	NUMBER	Number of messages moved to the propagation exception queue since the last time the agent was started
EXCEPTION_QUEUE	VARCHAR2	Exception queue used for logging purposes
FAILURES	NUMBER	Number of propagation failures
LAST_ERROR_DATE	DATE	Date of last propagation error
LAST_ERROR_MSG	VARCHAR2	Message for last propagation error

Table C-20 (Cont.) MGW_SUBSCRIBERS View Properties

Name	Type	Description
LAST_ERROR_TIME	VARCHAR2	Time of last propagation error
OPTIONS	SYS.MGW_PROPERTIES	Subscriber options
PROP_STYLE	VARCHAR2	Message propagation style. Values: • NATIVE is for native message propagation • JMS is for JMS message propagation
PROPAGATED_MSGS	NUMBER	Number of messages propagated to the destination queue since the last time the agent was started
PROPAGATION_TYPE	VARCHAR2	Propagation type. Values: • OUTBOUND is for Oracle Database AQ to non-Oracle propagation • INBOUND is for non-Oracle to Oracle Database AQ propagation
QUEUE_NAME	VARCHAR2	Subscriber source queue
RULE	VARCHAR2	Subscription rule
STATUS	VARCHAR2	Subscriber status. Values: • ENABLED means the subscriber is enabled • DELETE_PENDING means subscriber removal is pending, usually because DBMS_MGWADM.REMOVE_SUBSCRIBER has been called but certain cleanup tasks pertaining to this subscriber are still outstanding
SUBSCRIBER_ID	VARCHAR2	Propagation subscriber identifier
TRANSFORMATION	VARCHAR2	Transformation used for message conversion

MGW_SCHEDULES: Information About Schedules

This view lists configuration and status information for Messaging Gateway schedules. The view includes most of the schedule properties specified when the schedule is created, as well as other status information.

Table C-21 MGW_SCHEDULES View Properties

Name	Type	Description
DESTINATION	VARCHAR2	Propagation destination
LATENCY	NUMBER	Propagation window latency (in seconds)
NEXT_TIME	VARCHAR2	Reserved for future use
PROPAGATION_TYPE	VARCHAR2	Propagation type. Values: • OUTBOUND is for Oracle Database AQ to non-Oracle propagation • INBOUND is for non-Oracle to Oracle Database AQ propagation
PROPAGATION_WINDOW	NUMBER	Reserved for future use
SCHEDULE_DISABLED	VARCHAR2	Indicates whether the schedule is disabled. Y means the schedule is disabled. N means the schedule is enabled.
SCHEDULE_ID	VARCHAR2	Propagation schedule identifier
SOURCE	VARCHAR2	Propagation source
START_DATE	DATE	Reserved for future use

Table C-21 (Cont.) MGW_SCHEDULES View Properties

Name	Type	Description
`START_TIME`	`VARCHAR2`	Reserved for future use