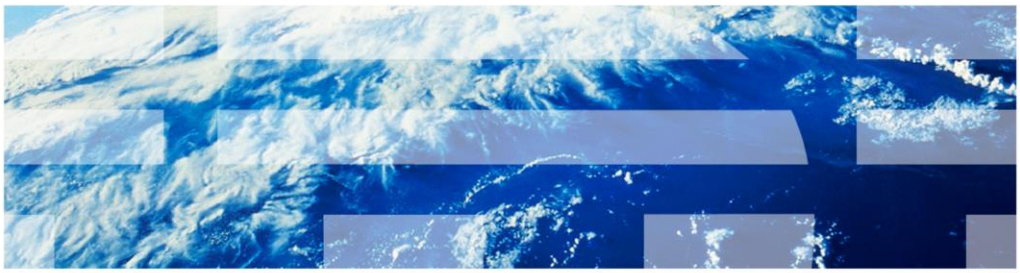
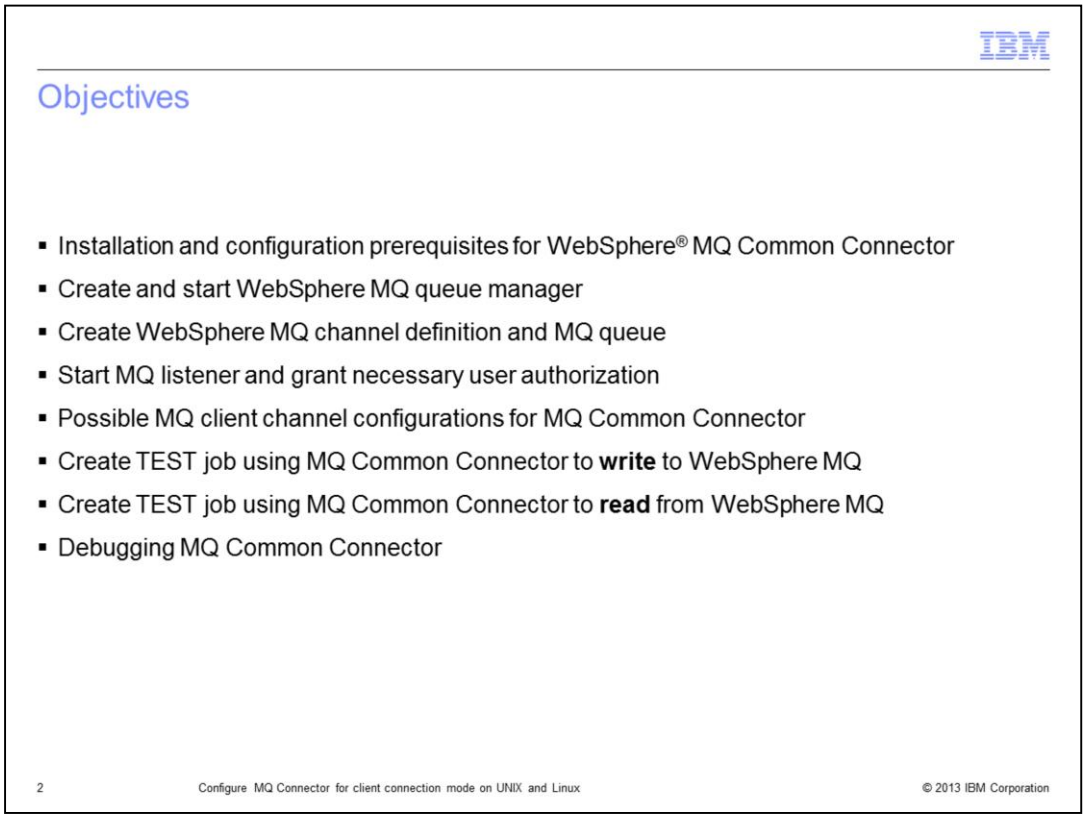
IBM

InfoSphere Information Server DataStage V8.1 - 9.1

Configure MQ Connector for client connection mode on UNIX and Linux



This presentation will discuss how to configure and test the MQ Common Connector for DataStage® version 8.1 to version 9.1 on UNIX® and Linux® using the client connection mode.



The objectives of this presentation are to demonstrate how to setup a simple DataStage job using the MQ Connector stage to write and read messages from a WebSphere MQ queue.

This presentation provides steps to run some simple commands from a command prompt to initially create and start a WebSphere MQ Manager.

Next, the presentation shows you how to create an MQ channel definition and MQ queue. It also shows you how to start the MQ listener that is used by the DataStage MQ Common Connector to run using the client connection mode. The presentation also discusses the necessary authorizations and how to grant them to the user.

Next, it discusses the three possible options to configure the MQ Common Connector channel definitions when using the client connection mode.

Once these steps have been completed, the presentation demonstrates how to create a simple DataStage test job using the DataStage MQ Connector stage that is used to write some test messages to this previously created queue. Also, a simple DataStage job will demonstrate on how to retrieve these same messages using the DataStage MQ Connector stage.

Finally, it discusses how to enable some additional debugging on the MQ Common Connector to help diagnose issues that may arise while configuring and using the connector.

Installation and configuration prerequisites

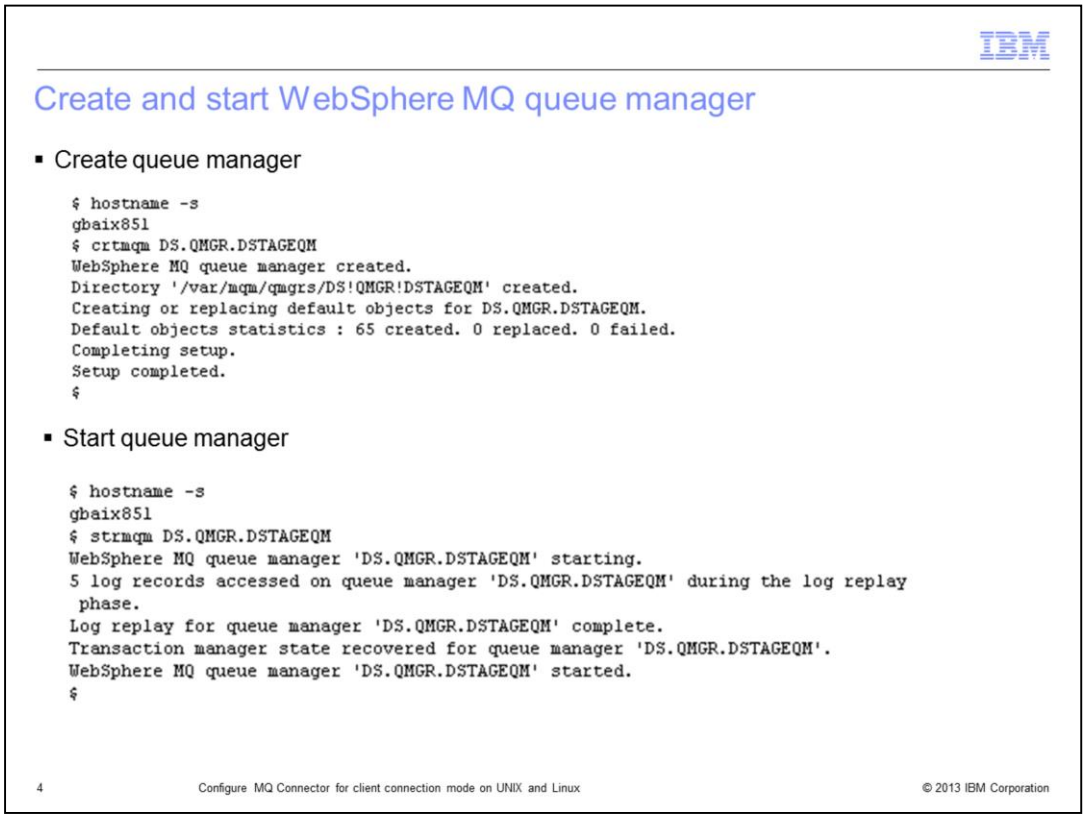
- Client connection mode
  - WebSphere MQ client
    - Must be installed on same node as connector
  - WebSphere MQ server
    - Must be installed on node in same network connector is installed
    - May be installed on same node as connector
  - Must be network connection between client node and server node
- Server connection mode
  - WebSphere MQ server
    - Must be installed on same node as connector
- Connection mode selected in stage properties

The installation requirements must be met depending upon whether you are wanting to use the MQ Common Connector in server connection mode or in client connection mode.

This mode is determined by your selection in the Mode property within the MQ Connector stage properties.

Depending on which of the two options you use, will determine if you need to install the WebSphere MQ clients or WebSphere MQ server on the same node as the connector.

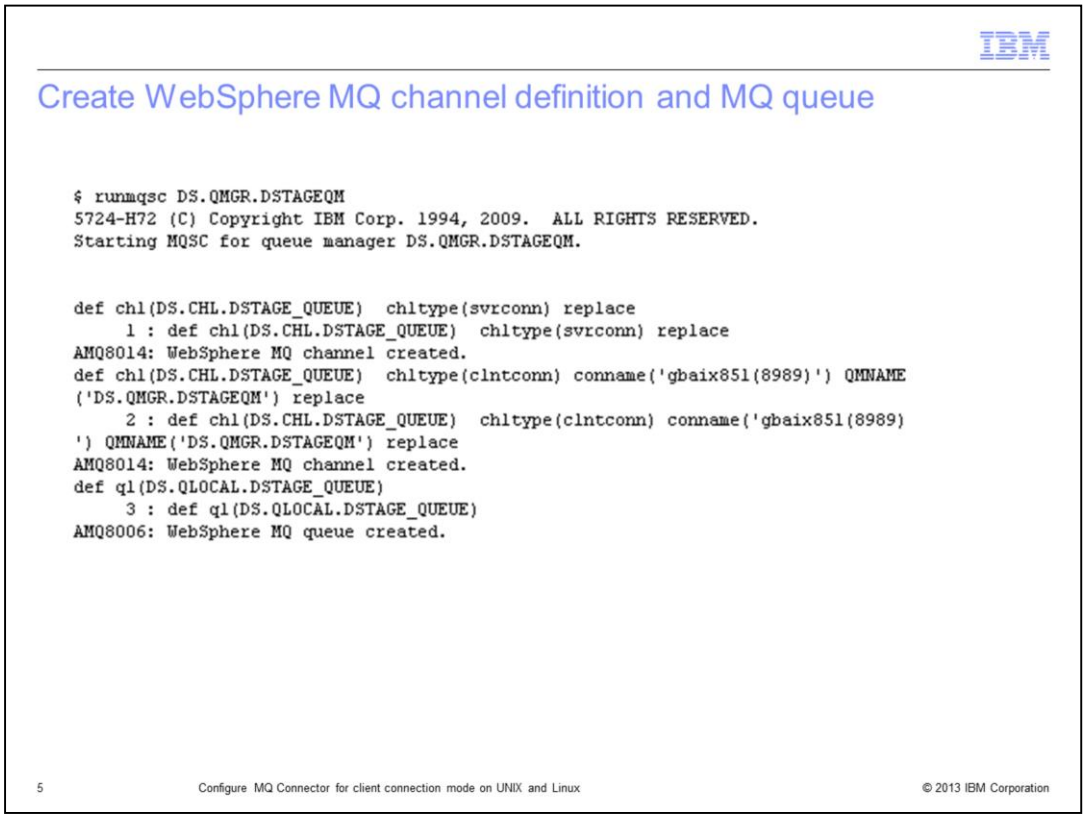
This presentation will discuss the Client connection mode as this is the most commonly used configuration.



This slide displays example commands that can be run as the WebSphere mqm user to create a default queue manager named **DS.QMGR.DSTAGEQM** that is created on an MQ server. In this presentation, the queue manager and queue is created on an MQ server named gbaix851.

Create the queue manager by running the crtmqm command displayed on this slide. A message should be displayed to show that the queue manager has been created successfully.

Once the queue manager has been created, start the queue manager by running the **strmqm** command displayed on this slide. A message should be displayed to show that the queue manager has been started successfully.

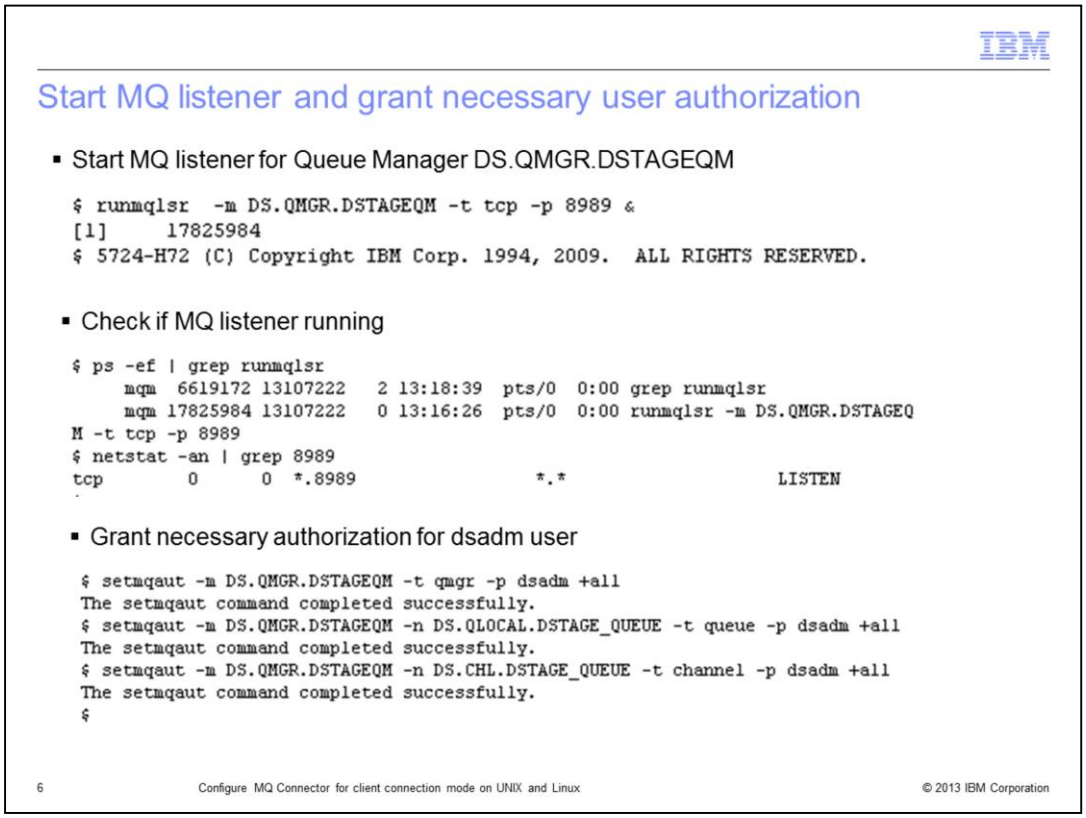


The next step is to create an MQ queue channel definition named DS.CHL.DSTAGE_QUEUE. This needs to be done as the WebSphere mqm user.

This example creates a channel definition using the client connection type that will connect to an MQ listener listening on port 8989 on the server named gbaix851.

These channel definition details are used later in the presentation for the MQ Common Connector connection properties using the client connection mode.

This step also created an MQ queue named DS.QLOCAL.DSTAGE_QUEUE.

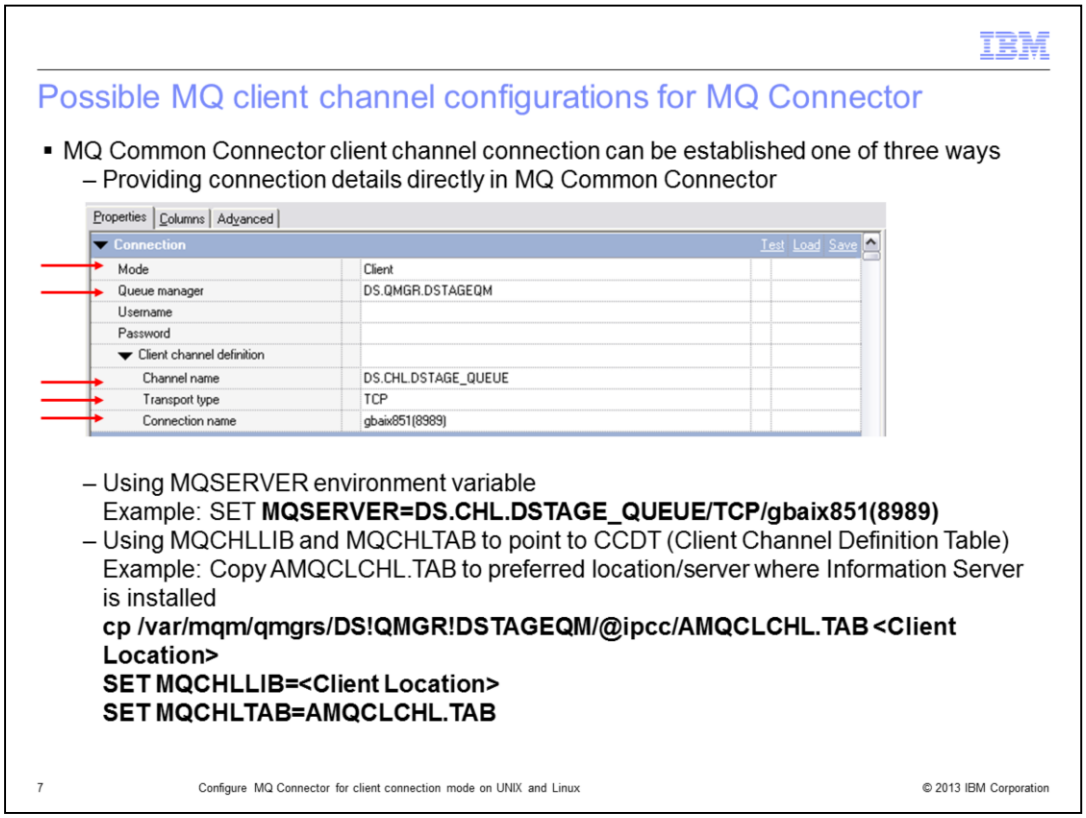


Next, as the mqm user, start the MQ listener for the Queue Manager DS.QMGR.DSTAGEQM that is listening on the port 8989 using TCP transport type on the MQ server gbaix851.

Verify that the MQ listener is up and running on the MQ server by using the ps and neststat commands as displayed on this slide.

To enable the connector to access the specified queue manager and queue objects, certain authorizations must be granted to the user ID under whose credentials the connector runs. In the example displayed on this slide, the necessary authorization is being granted to the dsadm user by running the **setmqaut** command. This is the user ID that is used to run the connector process.

If further assistance is required in setting up the configuration of the MQ Manager, channel definition, queue or setting the necessary authorizations, consult an MQ administrator.



This slide explains the three possible options that are available with the MQ Common Connector client channel connection configuration.

First, ensure the MQ clients are installed on the DataStage conductor node as previously explained in the earlier slides.

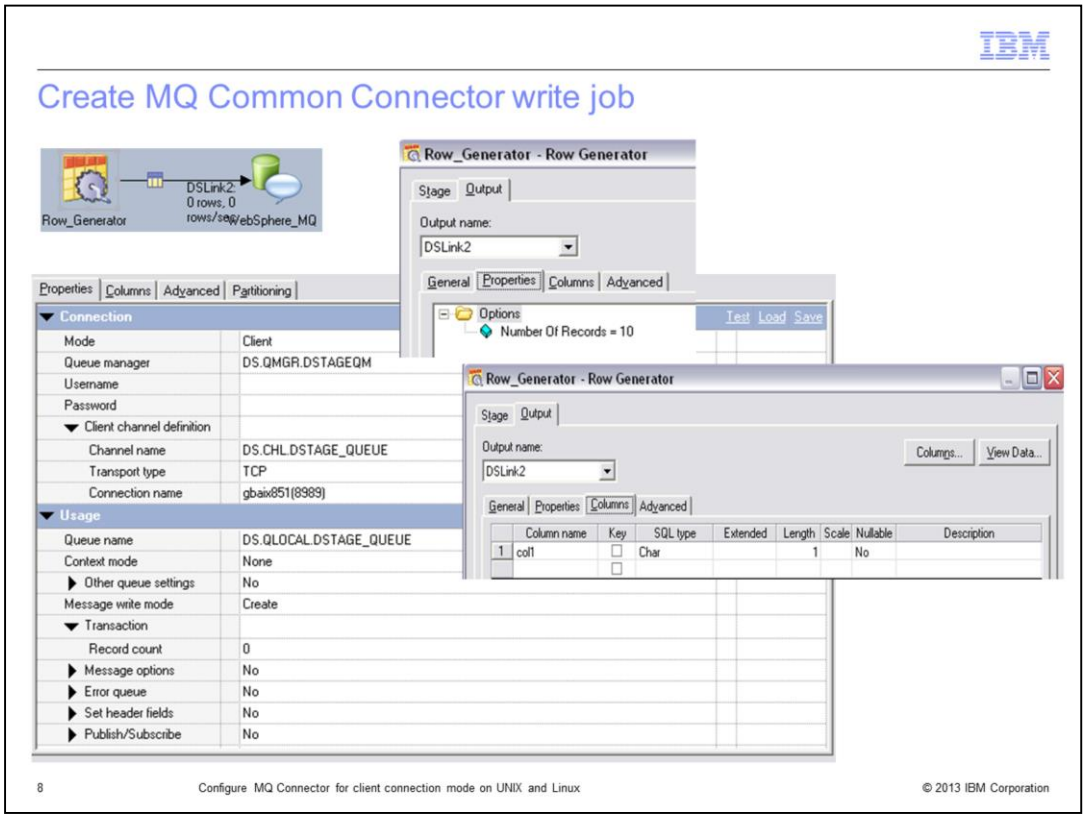
Next, ensure that the network connectivity between the DataStage server and the remote MQ server has been setup correctly. In this example, the remote MQ server is gbaix851.

Again, there are three possible options that are available with the MQ Common Connector client channel connection configuration.

The first option is to enter the client channel definition details directly into the MQ Common Connector properties as in the example screen displayed on this slide.

The second option is to use the environment variable MQSERVER and set this to the correct channel definitions as per the example on this slide. You can set this environment variable at the DataStage engine level in the dsenv file, at the project level, or at the job properties level. If using the environment variable MQSERVER**,** the client channel definition in the MQ Connector can be left blank.

The third option is to copy the AMQCLCHL.TAB file to the Information Server DataStage Engine tier and then set the environment variables MQCHLLIB and MQCHLTAB to point to the Client Channel Definition Table. These two environment variables can be set at the DataStage engine level, project level or job properties level. If using this option, the Client channel definition field in the MQ Connector can be left blank.



Now that the MQ manager, channel definition and queue are setup, a simple test job should be created to test the connector.

Create a simple DataStage job consisting of a Row Generator that writes 10 messages to the previously created queue manager DS.QMGR.DSTAGEQM using the channel definition DS.CHL.DSTAGE_QUEUE and queue DS.QLOCAL.DSTAGE_QUEUE.

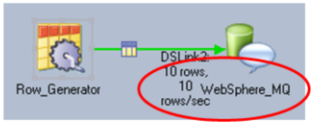
This job and all subsequent jobs in this presentation, uses the client connection mode as this is most commonly used. The server connection mode can also be used by selecting the server mode from the connection details and specify the queue manager DS.QMGR.DSTAGEQM and queue name DS.QLOCAL.DSTAGE_QUEUE. In this case, the client channel definitions are not required.

It is also correct to use any of the other mentioned connection options that were discussed earlier on slide 7.

Ensure that the job can be compiled and no errors are reported.

Run MQ Common Connector write job

- Run TEST write job
- Check queue for rows written by job




```
$ runmqsc DS.QMGR.DSTAGEQM
5724-H72 (C) Copyright IBM Corp. 1994, 2009.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager DS.QMGR.DSTAGEQM.


Display queue(DS.QLOCAL.DSTAGE_QUEUE)
     1 : Display queue(DS.QLOCAL.DSTAGE_QUEUE)
AMQ8409: Display Queue details.
   QUEUE(DS.QLOCAL.DSTAGE_QUEUE)           TYPE(QLOCAL)
   ACCTQ(QMGR)                             ALTDATE(2012-07-25)
   ALTTIME(13.09.31)                       BOQNAME( )
   BOTHRESH(0)                             CLUSNL( )
   CLUSTER( )                              CLWLPRTY(0)
   CLWLRANK(0)                             CLWLUSEQ(QMGR)
   CRDATE(2012-07-25)                      CRTIME(13.09.31)
   CURDEPTH(10)                            DEFBIND(OPEN)
```

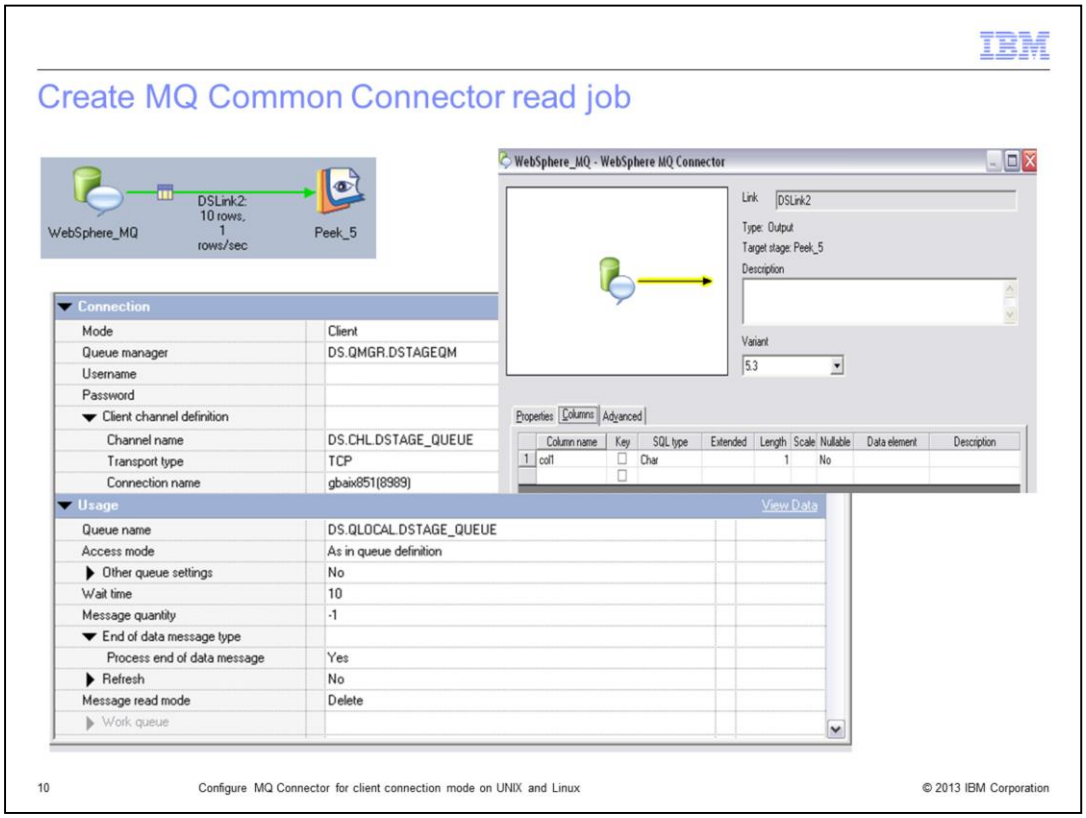
- **CURDEPTH(n)** - Contains number of messages in queue

9 Configure MQ Connector for client connection mode on UNIX and Linux

The next step is to run the test write job. Once the job has been run, display the MQ queue and make sure the 10 rows were written to the queue by the DataStage job.

Verify that these messages have been written to the MQ queue by running the runmqsc command as the mqm user to display the queue information.

The entry for the CURDEPTH will contain the number of messages currently in the queue. The example displayed on this slide shows the 10 messages have been written to the queue as expected.



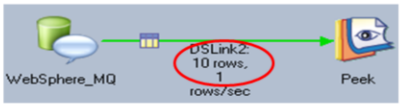
Once the job writing to the queue is working properly, create a simple DataStage read job consisting of an MQ Connector input stage that will read the 10 messages that were previously written to the MQ queue using the earlier job.

This test job is using the client connection mode and the same details that were used in the earlier job to write to the same queue. Just as with the write job, the server mode may be selected from the connection details and the queue manager DS.QMGR.DSTAGEQM and queue name DS.QLOCAL.DSTAGE_QUEUE may be specified. In this case, there is no need for the client channel definitions.

Ensure that the job can be compiled and no errors are reported.

Run MQ Common Connector read job

- Run the TEST read job



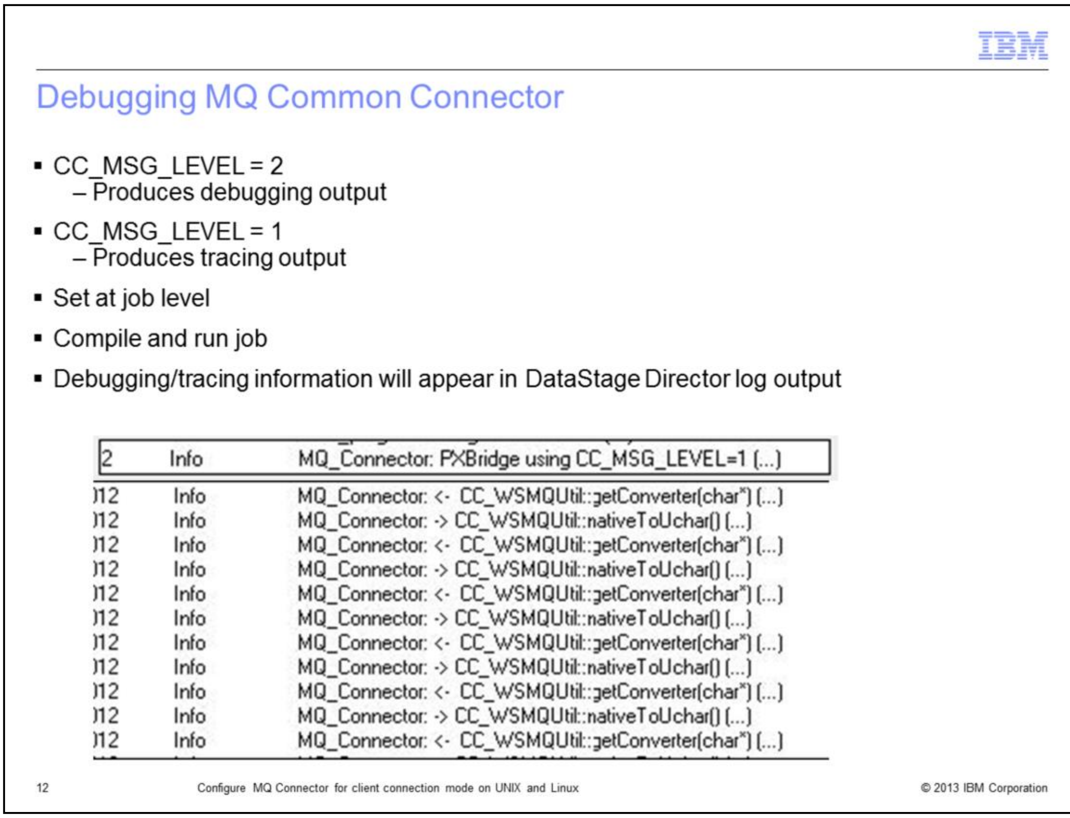

```
$ runmqsc DS.QMGR.DSTAGEQM
5724-H72 (C) Copyright IBM Corp. 1994, 2009.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager DS.QMGR.DSTAGEQM.


display queue(DS.QLOCAL.DSTAGE_QUEUE)
     1 : display queue(DS.QLOCAL.DSTAGE_QUEUE)
AMQ8409: Display Queue details.
   QUEUE(DS.QLOCAL.DSTAGE_QUEUE)           TYPE(QLOCAL)
   ACCTQ(QMGR)                             ALTDATE(2012-07-25)
   ALTTIME(13.09.31)                       BOQNAME( )
   BOTHRESH(0)                             CLUSNL( )
   CLUSTER( )                              CLWLPRTY(0)
   CLWLRANK(0)                             CLWLUSEQ(QMGR)
   CRDATE(2012-07-25)                      CRTIME(13.09.31)
   CURDEPTH(0)                             DEFBIND(OPEN)
```

- Message Read Mode set to Delete in MQ Connector stage properties

The last step is to run the read job. Once the job has been run, the DataStage designer display should show 10 rows processed.

Verify that these messages have been read correctly from the MQ queue by running the runmqsc command as the mqm user to display the queue information.

The entry for the CURDEPTH will contain the number of messages currently in the queue. The example displayed on this slide shows there are zero messages in the queue as CURDEPTH has a value of zero. This is expected when the MQ Connector stage is configured with **Message Read Mode** set to **Delete**. This further confirms that the messages have been retrieved from the queue.



In some cases, there may be a need to enable additional debugging to help diagnose problems. The additional debugging may also be provided to IBM Support should there be a need to raise a support call.

To turn on debugging information for the DataStage MQ Connector, set the environment variable CC_MSG_LEVEL to either 1 or 2. Setting CC_MSG_LEVEL to 2 produces debugging output. Setting CC_MSG_LEVEL to 1 produces tracing output.

The CC_MSG_LEVEL environment variable should always be set in the DataStage Designer at the job level so that no other jobs will run with the debugging or tracing turned on. Save, compile and run the job.

When setting these environment variables, the debugging or tracing information will appear in the DataStage Director log output.

When running with the CC_MSG_LEVEL set, the additional information that is logged in the DataStage Director logs may assist IBM Support to diagnose the issue as it contains all the function call entries and exit points. The example displayed on this slide, shows some of the additional informational messages that appear in the logs and confirm that the CC_MSG_LEVEL has been set correctly.

Trademarks, disclaimer, and copyright information