



QuartzDesk Web Application Installation and Upgrade Guide for IBM WebSphere AS 7.0, 8.0 and 8.5

QuartzDesk Version: 3.x

January 21, 2019



Table of Contents

1.	PURPOSE	4
2.	DEFINITIONS	5
3.	REQUIREMENTS.....	6
3.1	SOFTWARE REQUIREMENTS	6
3.1.1	<i>Browser</i>	<i>6</i>
3.1.2	<i>Operating System</i>	<i>6</i>
3.1.3	<i>Java</i>	<i>6</i>
3.1.4	<i>Application Server.....</i>	<i>6</i>
3.1.5	<i>Database</i>	<i>6</i>
3.1.6	<i>Database JDBC Driver</i>	<i>6</i>
3.1.7	<i>QuartzDesk Web Application Archive.....</i>	<i>7</i>
3.2	HARDWARE REQUIREMENTS.....	7
4.	INSTALLATION.....	8
4.1	DATABASE.....	8
4.2	JDBC DRIVER	8
4.3	JDBC PROVIDER	9
4.3.1	<i>DB2</i>	<i>10</i>
4.3.2	<i>H2.....</i>	<i>11</i>
4.3.3	<i>Microsoft SQL Server.....</i>	<i>12</i>
4.3.4	<i>MySQL.....</i>	<i>13</i>
4.3.5	<i>Oracle.....</i>	<i>14</i>
4.3.6	<i>PostgreSQL.....</i>	<i>15</i>
4.4	DATA SOURCE J2C AUTHENTICATION DATA	15
4.5	DATA SOURCE	16
4.5.1	<i>DB2</i>	<i>17</i>
4.5.2	<i>H2.....</i>	<i>20</i>
4.5.3	<i>Microsoft SQL Server.....</i>	<i>23</i>
4.5.4	<i>MySQL.....</i>	<i>26</i>
4.5.5	<i>Oracle.....</i>	<i>29</i>
4.5.6	<i>PostgreSQL.....</i>	<i>32</i>
4.6	APPLICATION WORK DIRECTORY	35
4.7	APPLICATION CONFIGURATION.....	36
4.8	DEPLOY APPLICATION.....	37
4.9	START APPLICATION	41
5.	UPGRADING	43
5.1	STOP EXISTING APPLICATION	43
5.2	BACKUP.....	43
5.3	REMOVE EXISTING APPLICATION	43
5.4	DEPLOY NEW APPLICATION.....	44
5.5	START NEW APPLICATION	44
6.	QUARTZDESK 2.X TO 3.X MIGRATION NOTES	45
6.1	MINIMUM REQUIRED JAVA VERSION.....	45
6.2	RENAME CONFIGURATION FILE	45
6.3	RENAME LOG FILES	45
6.4	ACCESS TO MONITORING URLS (REST API).....	46
6.5	ACCESS TO JAX-WS ENDPOINTS	47
7.	CLUSTER DEPLOYMENT NOTES.....	48
7.1	HTTP SESSION REPLICATION AND AFFINITY	48
7.2	SHARED WORK DIRECTORY	48

7.3	LOGGING CONFIGURATION	48
7.3.1	<i>Using Shared Log Files</i>	49
7.3.2	<i>Using Separate Log Files</i>	50
7.4	INTERNAL QUARTZ SCHEDULER	51



1. Purpose

This document describes the installation and upgrade process for the QuartzDesk web application 3.x on IBM WebSphere Application Server 7.0, 8.0 and 8.5.

If you experience any problems installing or upgrading the QuartzDesk web application, please let us know at support@quartzdesk.com.



2. Definitions

The following table lists all acronyms and shortcuts used throughout this document.

Acronym / Shortcut	Definition
AS	Application Server.
EAR	Enterprise Application Archive. A file with .ear extension.
JAR	Java Application Archive. A file with .jar extension.
JVM	Java Virtual Machine.
WAC	WebSphere Administrative Console.
WAR	Web Application Archive. A file with .war extension.
WAS	WebSphere Application Server.

The following table lists all locations and properties used throughout this document.

Location / Property	Example	Description
DB_HOST	localhost	QuartzDesk web application database server host.
DB_PORT	5432	QuartzDesk web application database server port.
DB_NAME	quartzdesk	QuartzDesk web application database name.
DB_SCHEMA	quartzdesk	QuartzDesk web application database schema.
DB_USER	quartzdesk	QuartzDesk web application database user.
DB_PASSWORD	quartzdesk	QuartzDesk web application database user password.
WAS_INSTALL_ROOT	/usr/local/was7	WebSphere Application Server installation directory.
WAS_SERVER_NAME	server1	WebSphere Application Server name.
WAS_SERVER_PROFILE	/usr/local/was7/profiles/server1	WebSphere Application Server profile directory.
WAS_HTTP_HOST	localhost	WebSphere HTTP listener host.
WAS_HTTP_PORT	9080	WebSphere HTTP listener port.
WORK_DIR	/var/quartzdesk-web.work	QuartzDesk web application work directory.

3. Requirements

3.1 Software Requirements

3.1.1 Browser

QuartzDesk web application GUI requires a modern JavaScript-enabled browser. Please make sure JavaScript is enabled and not blocked by third party anti-virus/anti-malware software.

The QuartzDesk web application has been tested with the following browser versions. These are also the minimum browsers versions required.

Browser	Minimum Version
Chrome	17
FireFox	10
Internet Explorer	8
Opera	12
Safari	6

3.1.2 Operating System

Windows XP, Windows Vista, Windows 7, Windows 8, Windows 10.

Linux (any distribution) with kernel 2.6.x and above.

Solaris 11.x and above.

3.1.3 Java

IBM Java (JDK) 7, 8, or 8 bundled with the IBM Websphere Application Server.

3.1.4 Application Server

IBM WebSphere Application Server 7.0.

IBM WebSphere Application Server 8.0.

IBM WebSphere Application Server 8.5.

3.1.5 Database

Database	Minimum Version
DB2	10.1
H2	1.3.174
Microsoft SQL Server	2008 R2 SP1
MySQL	5.6.4
Oracle	10.2 (10g R2)
PostgreSQL	9.1

3.1.6 Database JDBC Driver

Database	JDBC Driver
DB2	IBM DB2 JDBC 4.0 driver available at http://www-01.ibm.com/support/docview.wss?uid=swg21363866 .

H2	Database engine including the JDBC driver is available at http://www.h2database.com .
Microsoft SQL Server	Microsoft JDBC driver 4.0 for SQL Server available at http://msdn.microsoft.com/en-us/sqlserver/aa937724.aspx . We strongly advise against using the alternative JTDS JDBC driver because it does not support the datetime2 data type at this time. As a result, all datetime values written by the QuartzDesk web application would end up rounded up, or down. For datetime data type rounding details, please refer to http://msdn.microsoft.com/en-us/library/ms187819.aspx .
MySQL	Connector/J JDBC driver available at http://dev.mysql.com/downloads/connector/j/ .
Oracle	Oracle JDBC driver available at http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html . For a comprehensive overview of JDBC driver versions vs. supported database versions, please refer to http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-faq-090281.html#01_02 .
PostgreSQL	JDBC4 PostgreSQL driver available at http://jdbc.postgresql.org/ .

3.1.7 QuartzDesk Web Application Archive

To install QuartzDesk web application, you need to obtain the quartzdesk-web-x.y.z.war file. The latest version can be downloaded at www.quartzdesk.com (click Downloads → Latest Release → View files → quartzdesk-web-x.y.z.war).

3.2 Hardware Requirements

QuartzDesk web application runs on any physical or virtualized hardware that supports the above software requirements.



4. Installation

This chapter describes the standard QuartzDesk installation. If you are only evaluating QuartzDesk, you may be interested in the **one-step installation mode** to dramatically reduce the number of required installation steps. For details, please refer to our [FAQs](#) (search for "one-step installation").

Unless noted otherwise, all objects created / registered in WAC described in this document are created / registered in the application server scope.

4.1 Database

Create a new database user named `quartzdesk` (`DB_USER`) with an arbitrary password (`DB_PASSWORD`).

Create a new QuartzDesk web application database named `quartzdesk1` (`DB_NAME`) owned by `DB_USER`.

In the `quartzdesk` database create a new schema named `quartzdesk` (`DB_SCHEMA`). The schema must be owned by `DB_USER`. Make the created `DB_SCHEMA` the default schema of `DB_USER` and/or add the schema to the `DB_USER`'s schema search path.

Please contact your DBA, or refer to the database engine documentation for instructions on how to complete the above database-specific tasks.



Please note that you do not have to create any database objects (tables, keys, indices etc.) in the `quartzdesk` database / schema. These objects will be automatically created by the QuartzDesk web application during its first start.

4.2 JDBC Driver

Download and install the JDBC driver for the created database. For a list of supported JDBC drivers please refer to chapter 3.1.6.

In WAC (Environment → WebSphere variables) create a new variable pointing to the directory with the JDBC driver JAR file(s) for the QuartzDesk web application database. Make sure the JDBC driver JAR files are readable by the user the WAS process is started under.

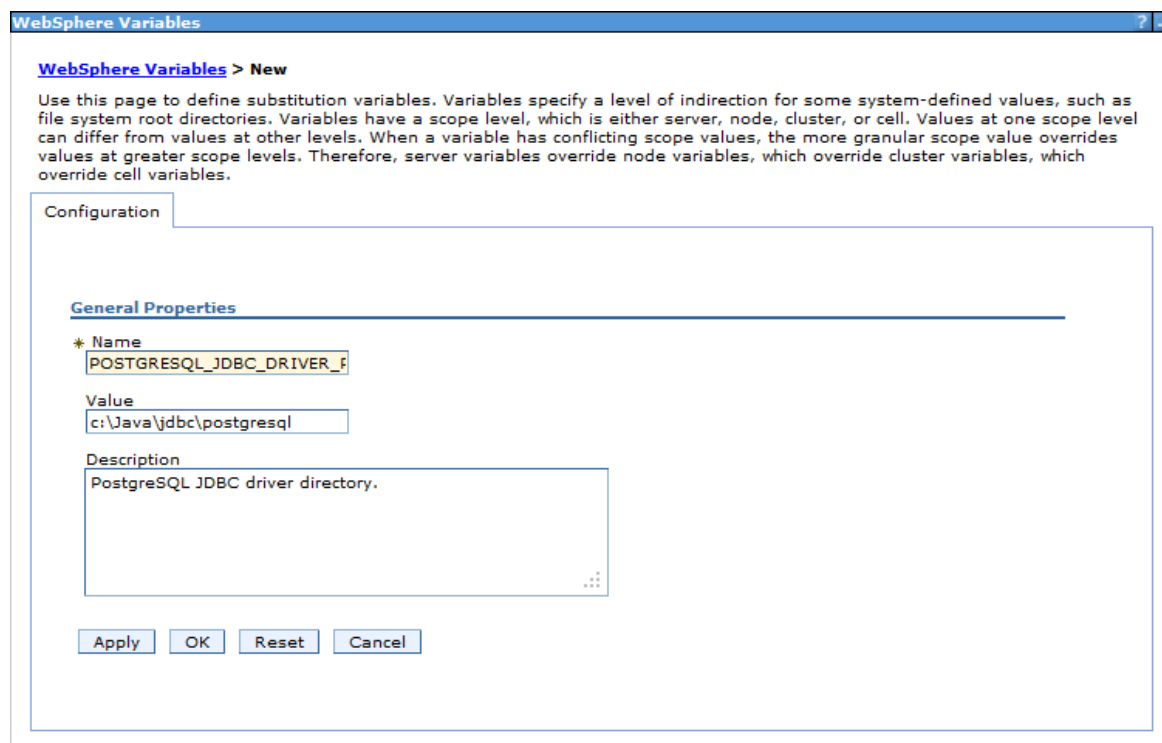
We suggest that you use the following variable names. Please note that some of these variables may already be defined. If the variable is already defined and points to a directory with a JDBC driver that is supported by QuartzDesk web application, then leave the variable as is. Otherwise, add, or adjust the variable value accordingly.

Database	WebSphere Variable Name	Default Value In WAC
DB2	<code>DB2_JCC_DRIVER_PATH</code>	Defined with empty value.
H2	<code>H2_JDBC_DRIVER_PATH</code>	Not defined.
Microsoft SQL Server	<code>MICROSOFT_JDBC_DRIVER_PATH</code>	Defined with empty value.

¹ DB2 restricts the database name length to the maximum of 8 characters. Please adjust the database name accordingly (e.g. `qdesk`).

MySQL	MYSQL_JDBC_DRIVER_PATH	Not defined.
Oracle	ORACLE_JDBC_DRIVER_PATH	Defined with empty value.
PostgreSQL	POSTGRESQL_JDBC_DRIVER_PATH	Not defined.

The following is an example of creating a new variable named POSTGRESQL_JDBC_DRIVER_PATH pointing to the directory that contains the PostgreSQL JDBC driver JAR file.



The screenshot shows the 'WebSphere Variables' configuration page. The title bar reads 'WebSphere Variables'. Below the title bar, there is a breadcrumb 'WebSphere Variables > New'. A paragraph of text explains that variables specify a level of indirection for system-defined values and that values at one scope level can differ from values at other levels. Below this text is a 'Configuration' section with a 'General Properties' sub-section. The 'Name' field contains 'POSTGRESQL_JDBC_DRIVER_PATH'. The 'Value' field contains 'c:\Java\jdbc\postgresql'. The 'Description' field contains 'PostgreSQL JDBC driver directory.'. At the bottom of the configuration area are four buttons: 'Apply', 'OK', 'Reset', and 'Cancel'.

4.3 JDBC Provider

In WAC (Resources → JDBC → JDBC providers) register a JDBC provider for the QuartzDesk web application database data-source.



4.3.1 DB2

Create a new JDBC Provider

Create a new JDBC Provider

→ Step 1: Create new JDBC provider

Step 2: Enter database class path information

Step 3: Summary

Create new JDBC provider

Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.

Scope

* Database type

* Provider type

* Implementation type

* Name

Description

Next Cancel

Create a new JDBC Provider

Create a new JDBC Provider

Step 1: Create new JDBC provider

→ Step 2: Enter database class path information

Step 3: Summary

Enter database class path information

Set the environment variables that represent the JDBC driver class files, which WebSphere(R) Application Server uses to define your JDBC provider. This wizard page displays the file names; you supply only the directory locations of the files. Use complete directory paths when you type the JDBC driver file locations. For example: C:\SQLLIB\java on Windows(R) or /home/db2inst1/sqllib/java on Linux(TM).

If a value is specified for you, you may click Next to accept the value.

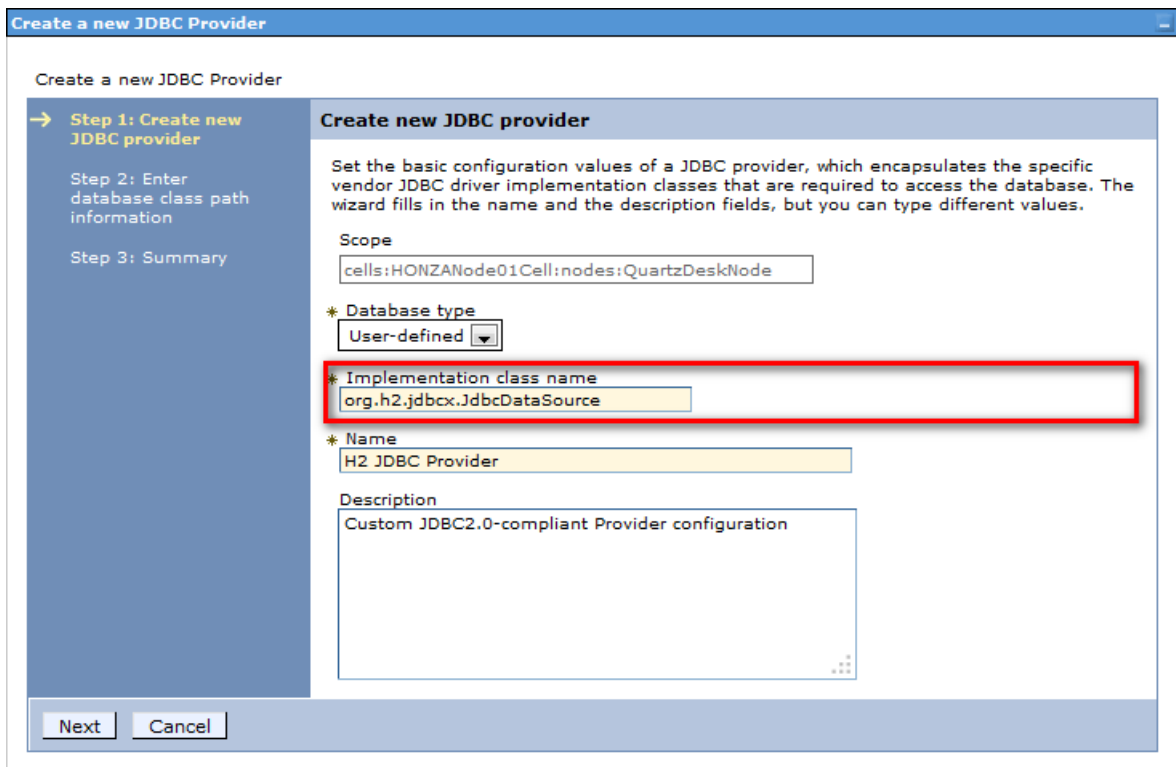
Class path:

Directory location for "db2jcc4.jar, db2jcc_license_cisuz.jar" which is saved as WebSphere variable \${DB2_JCC_DRIVER_PATH}

Native library path
 Directory location which is saved as WebSphere variable \${DB2_JCC_DRIVER_NATIVEPATH}

Previous Next Cancel

4.3.2 H2



The screenshot shows the 'Create a new JDBC Provider' wizard. The left sidebar indicates the current step is 'Step 1: Create new JDBC provider'. The main area is titled 'Create new JDBC provider' and contains the following fields:

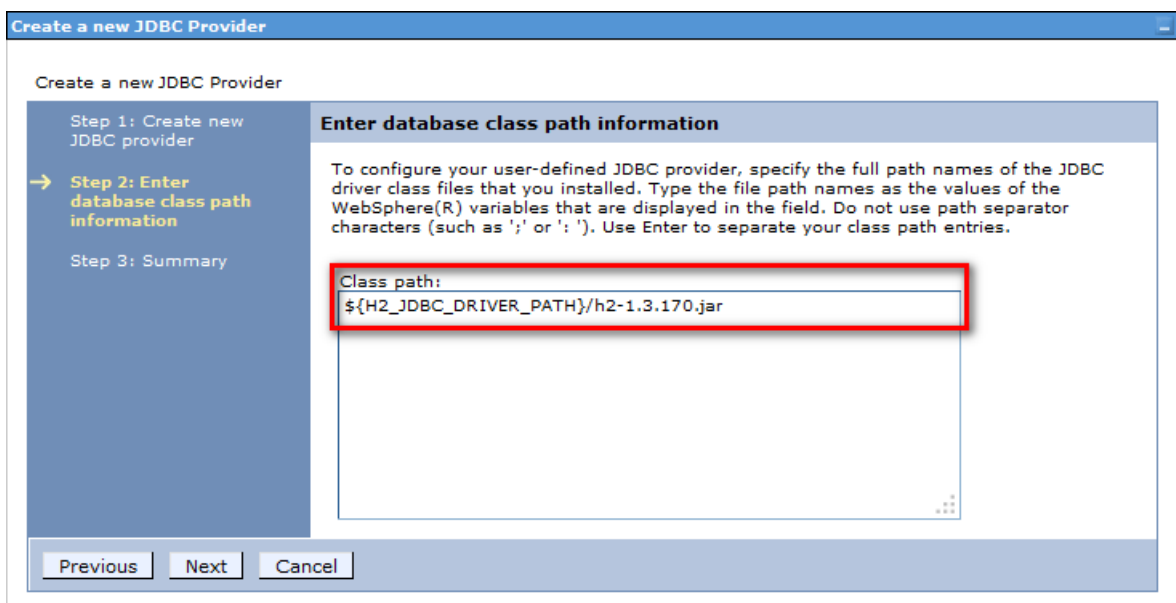
- Scope: cells:HONZANode01Cell:nodes:QuartzDeskNode
- * Database type: User-defined
- * Implementation class name: org.h2.jdbcx.JdbcDataSource (highlighted with a red box)
- * Name: H2 JDBC Provider
- Description: Custom JDBC2.0-compliant Provider configuration

Buttons for 'Next' and 'Cancel' are visible at the bottom.

Database type: User-defined

Implementation class name: org.h2.jdbcx.JdbcDataSource

Name: H2 JDBC Provider



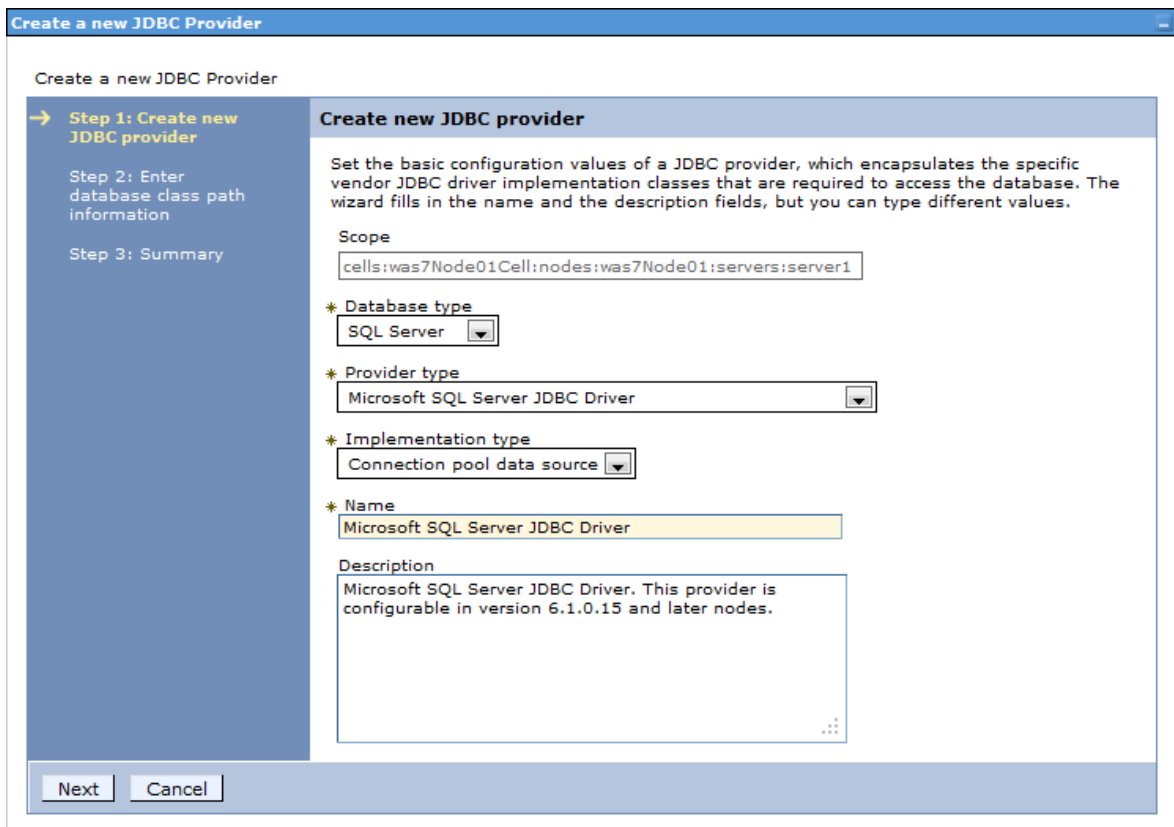
The screenshot shows the 'Create a new JDBC Provider' wizard at Step 2: Enter database class path information. The left sidebar indicates the current step is 'Step 2: Enter database class path information'. The main area is titled 'Enter database class path information' and contains the following field:

- Class path: \${H2_JDBC_DRIVER_PATH}/h2-1.3.170.jar (highlighted with a red box)

Buttons for 'Previous', 'Next', and 'Cancel' are visible at the bottom.

Class path: H2 JDBC driver classpath

4.3.3 Microsoft SQL Server



Create a new JDBC Provider

Step 1: Create new JDBC provider

Step 2: Enter database class path information

Step 3: Summary

Create new JDBC provider

Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.

Scope
cells:was7Node01Cell:nodes:was7Node01:servers:server1

* Database type
SQL Server

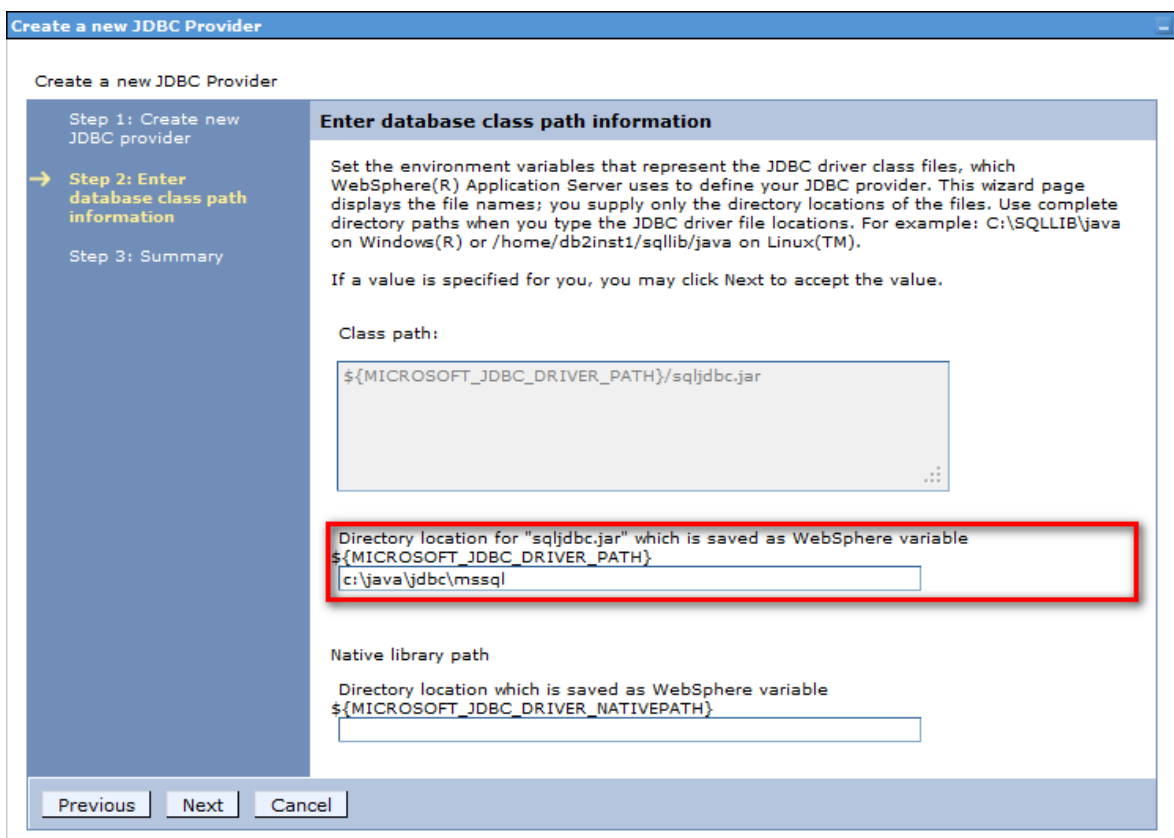
* Provider type
Microsoft SQL Server JDBC Driver

* Implementation type
Connection pool data source

* Name
Microsoft SQL Server JDBC Driver

Description
Microsoft SQL Server JDBC Driver. This provider is configurable in version 6.1.0.15 and later nodes.

Next Cancel



Create a new JDBC Provider

Step 1: Create new JDBC provider

Step 2: Enter database class path information

Step 3: Summary

Enter database class path information

Set the environment variables that represent the JDBC driver class files, which WebSphere(R) Application Server uses to define your JDBC provider. This wizard page displays the file names; you supply only the directory locations of the files. Use complete directory paths when you type the JDBC driver file locations. For example: C:\SQLLIB\java on Windows(R) or /home/db2inst1/sqllib/java on Linux(TM).

If a value is specified for you, you may click Next to accept the value.

Class path:
\${MICROSOFT_JDBC_DRIVER_PATH}/sqljdbc.jar

Directory location for "sqljdbc.jar" which is saved as WebSphere variable
\${MICROSOFT_JDBC_DRIVER_PATH}
c:\java\jdbc\mssql

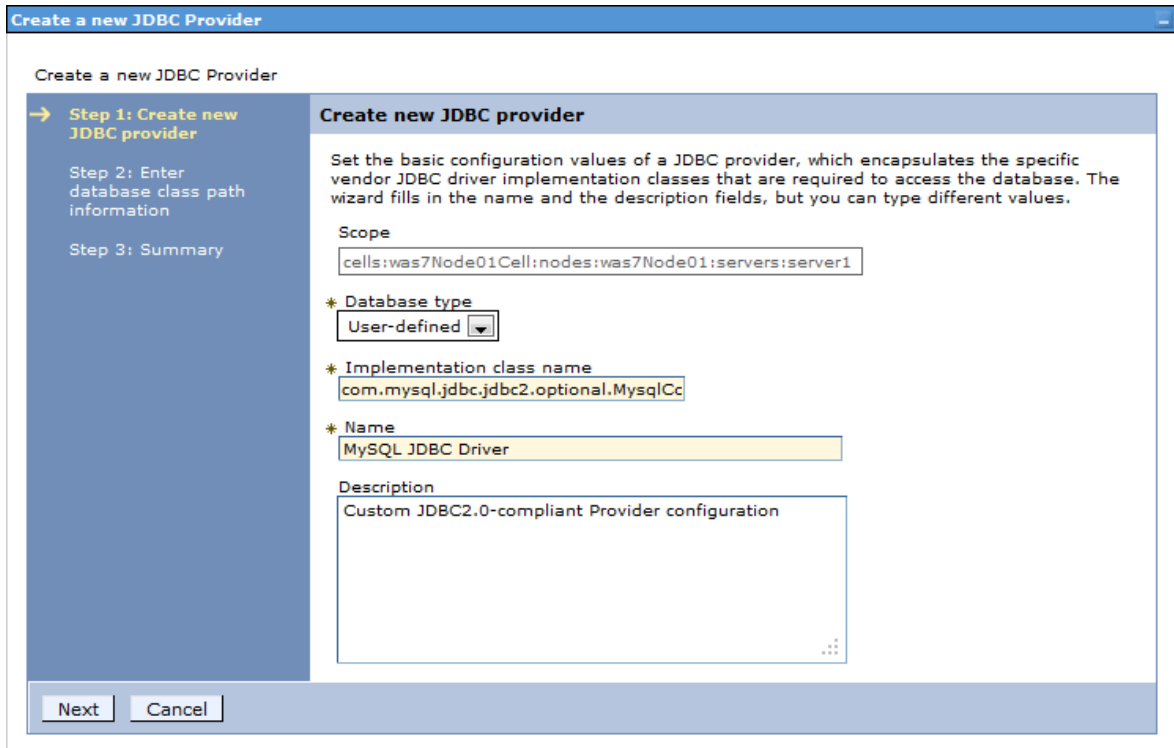
Native library path
Directory location which is saved as WebSphere variable
\${MICROSOFT_JDBC_DRIVER_NATIVEPATH}

Previous Next Cancel



The name of the Microsoft SQL Server driver JAR file is sqljdbc4.jar and not sqljdbc.jar as suggested by WAC (see the default “Class path” value in the figure above). To fix the name of the JAR file you will need to edit the JDBC provider class path after it has been saved.

4.3.4 MySQL



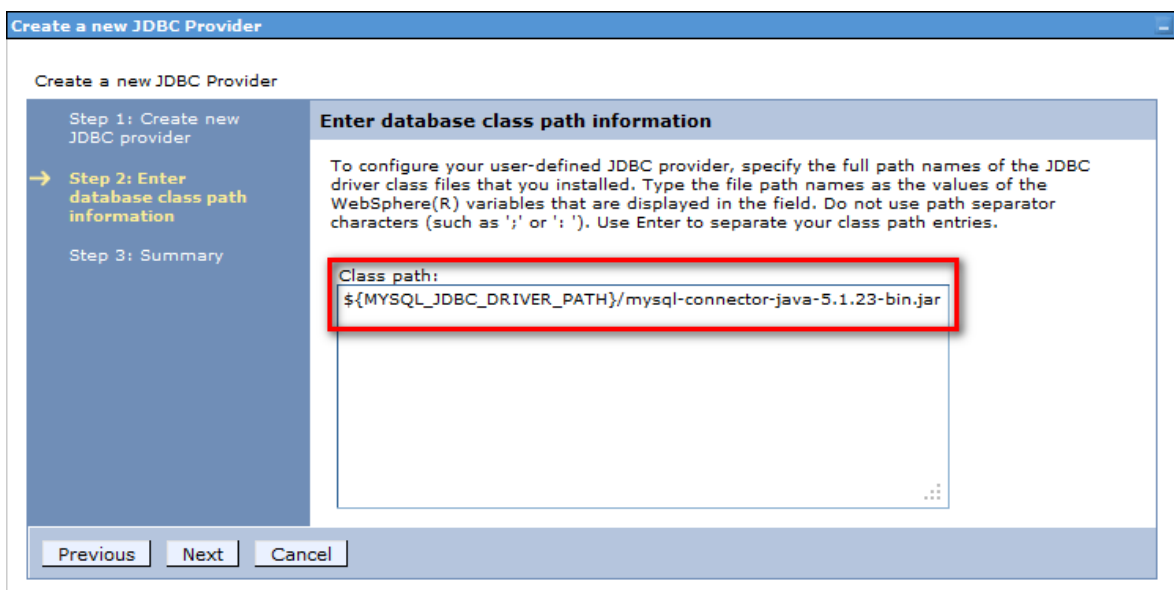
The screenshot shows the 'Create a new JDBC Provider' wizard. The left sidebar indicates the current step is 'Step 1: Create new JDBC provider'. The main area is titled 'Create new JDBC provider' and contains the following fields:

- Scope:** cells:was7Node01Cell:nodes:was7Node01:servers:server1
- * Database type:** User-defined
- * Implementation class name:** com.mysql.jdbc.jdbc2.optional.MysqlCc
- * Name:** MySQL JDBC Driver
- Description:** Custom JDBC2.0-compliant Provider configuration

Buttons for 'Next' and 'Cancel' are visible at the bottom.

Implementation class name:

`com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoolDataSource`



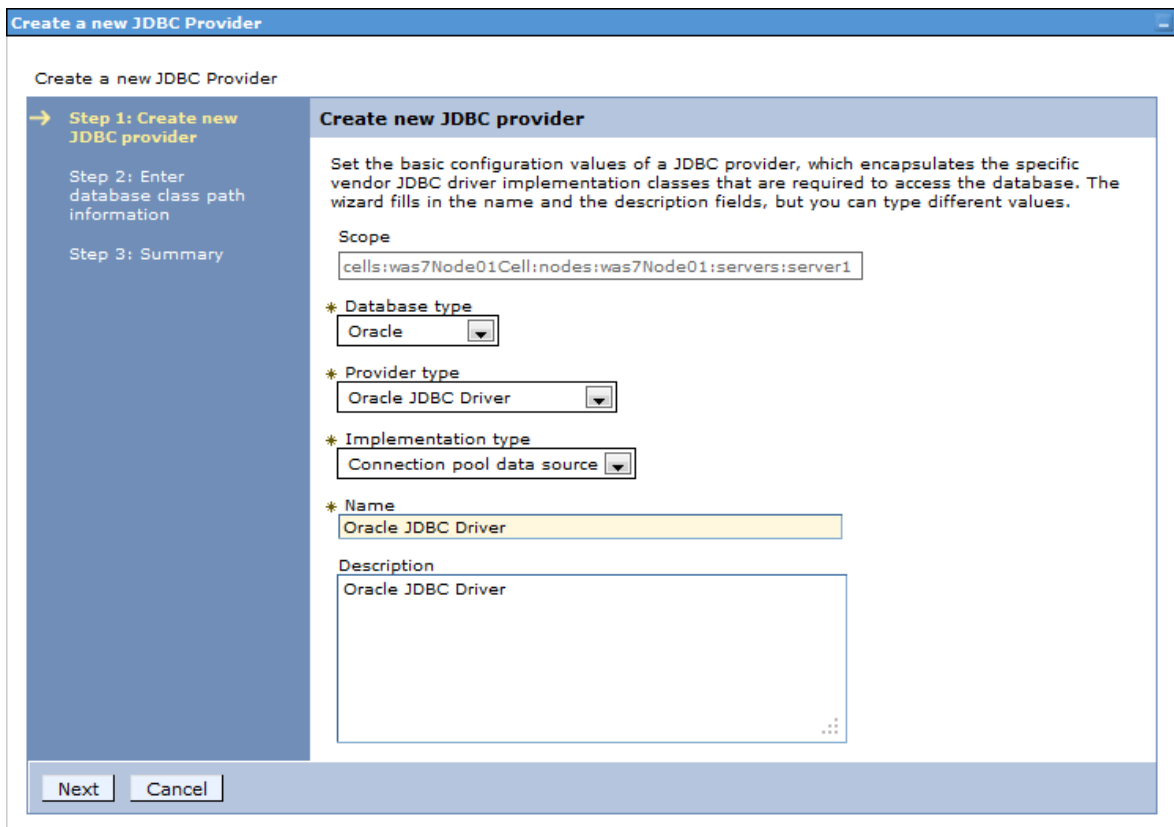
The screenshot shows the 'Create a new JDBC Provider' wizard at 'Step 2: Enter database class path information'. The main area is titled 'Enter database class path information' and contains the following text and field:

To configure your user-defined JDBC provider, specify the full path names of the JDBC driver class files that you installed. Type the file path names as the values of the WebSphere(R) variables that are displayed in the field. Do not use path separator characters (such as ';' or ':'). Use Enter to separate your class path entries.

Class path:
\${MYSQL_JDBC_DRIVER_PATH}/mysql-connector-java-5.1.23-bin.jar

Buttons for 'Previous', 'Next', and 'Cancel' are visible at the bottom.

4.3.5 Oracle



Create a new JDBC Provider

Create a new JDBC Provider

→ **Step 1: Create new JDBC provider**

Step 2: Enter database class path information

Step 3: Summary

Create new JDBC provider

Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.

Scope

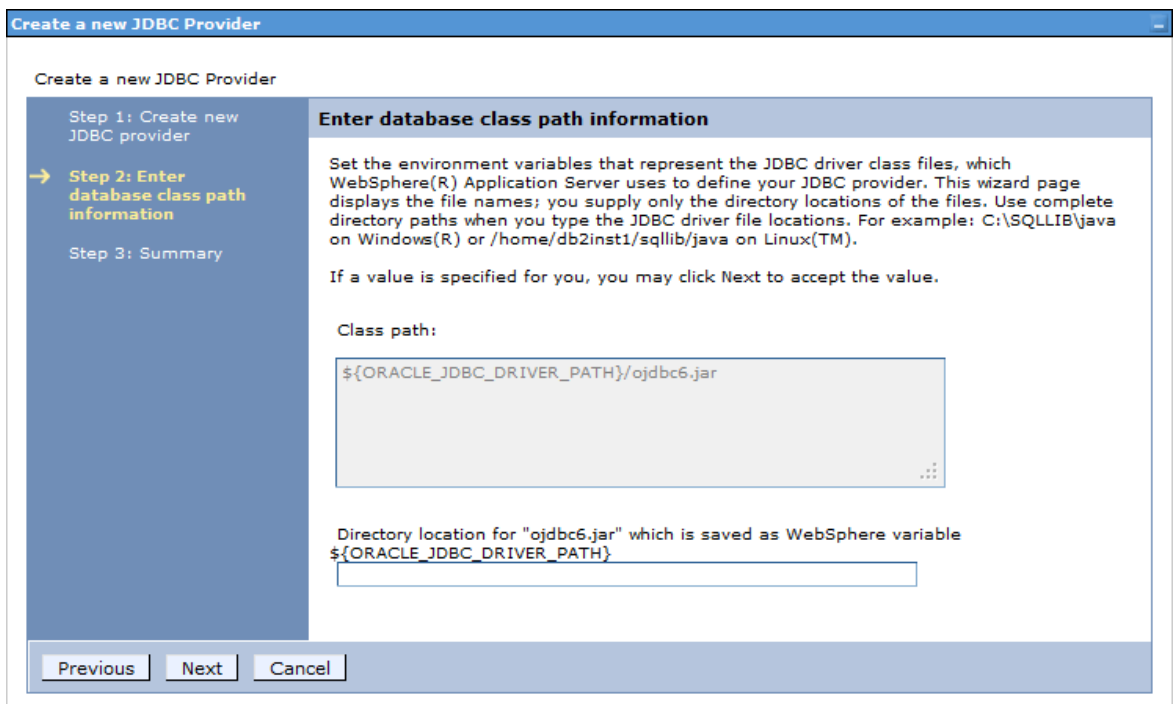
* Database type

* Provider type

* Implementation type

* Name

Description



Create a new JDBC Provider

Create a new JDBC Provider

Step 1: Create new JDBC provider

→ **Step 2: Enter database class path information**

Step 3: Summary

Enter database class path information

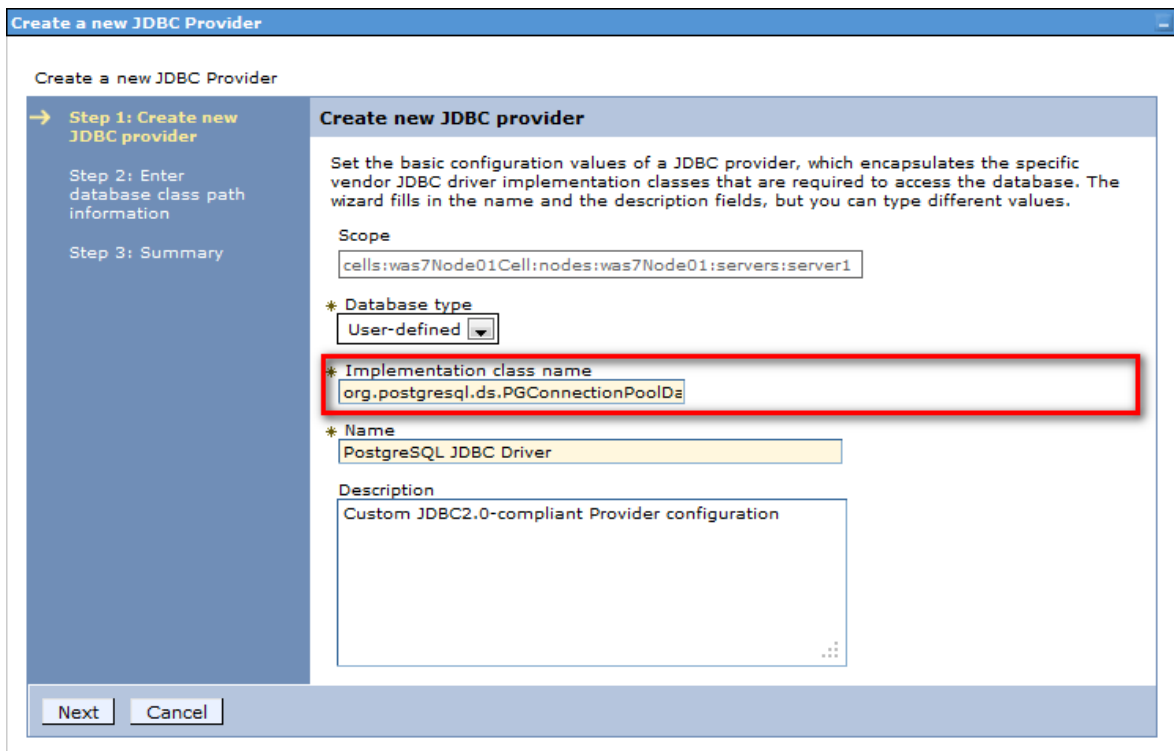
Set the environment variables that represent the JDBC driver class files, which WebSphere(R) Application Server uses to define your JDBC provider. This wizard page displays the file names; you supply only the directory locations of the files. Use complete directory paths when you type the JDBC driver file locations. For example: C:\SQLLIB\java on Windows(R) or /home/db2inst1/sqllib/java on Linux(TM).

If a value is specified for you, you may click Next to accept the value.

Class path:

Directory location for "ojdbc6.jar" which is saved as WebSphere variable
`${ORACLE_JDBC_DRIVER_PATH}`

4.3.6 PostgreSQL

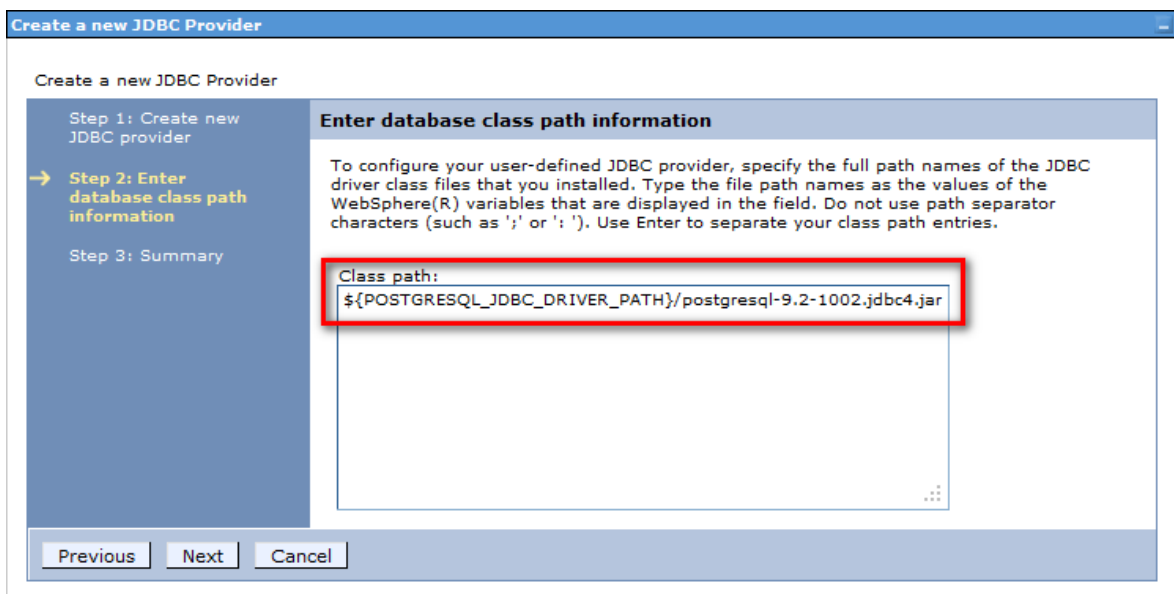


The screenshot shows the 'Create a new JDBC Provider' wizard in Step 1. The left sidebar lists three steps: Step 1: Create new JDBC provider (selected), Step 2: Enter database class path information, and Step 3: Summary. The main area is titled 'Create new JDBC provider' and contains the following fields:

- Scope:** cells:was7Node01Cell:nodes:was7Node01:servers:server1
- * Database type:** User-defined
- * Implementation class name:** org.postgresql.ds.PGConnectionPoolData (highlighted with a red box)
- * Name:** PostgreSQL JDBC Driver
- Description:** Custom JDBC2.0-compliant Provider configuration

Buttons for 'Next' and 'Cancel' are visible at the bottom.

Implementation class name: org.postgresql.ds.PGConnectionPoolDataSource



The screenshot shows the 'Create a new JDBC Provider' wizard in Step 2. The left sidebar lists three steps: Step 1: Create new JDBC provider, Step 2: Enter database class path information (selected), and Step 3: Summary. The main area is titled 'Enter database class path information' and contains the following field:

- Class path:** \${POSTGRESQL_JDBC_DRIVER_PATH}/postgresql-9.2-1002.jdbc4.jar (highlighted with a red box)

Buttons for 'Previous', 'Next', and 'Cancel' are visible at the bottom.

4.4 Data Source J2C Authentication Data

In WAC (Security → Global Security → Java Authentication and Authorization Service → J2C Authentication Data) create a new authentication entry for the QuartzDesk web application database data source.



Global security > JAAS - J2C authentication data > New

Specifies a list of user identities and passwords for Java(TM) 2 connector security to use.

General Properties

- * Alias: AuthQuartzDeskDS
- * User ID: quartzdesk
- * Password:
- Description: Authentication for QuartzDesk

Apply OK Reset Cancel

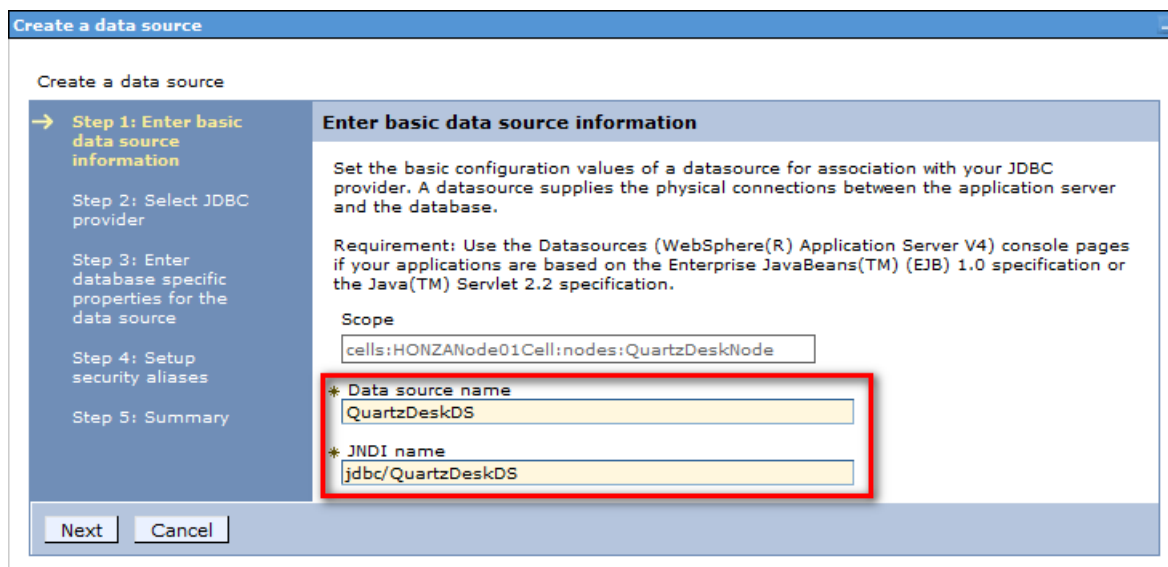
Alias: AuthQuartzDeskDS
User ID: DB_USER
Password: DB_PASSWORD

4.5 Data Source

In WAC (Resources → JDBC → Data sources) create a new data source for the QuartzDesk web application database.

In Step 1, provide the data source name and JNDI name.

Data source name: QuartzDeskDS
JNDI name: jdbc/QuartzDeskDS



Create a data source

→ Step 1: Enter basic data source information

Step 2: Select JDBC provider

Step 3: Enter database specific properties for the data source

Step 4: Setup security aliases

Step 5: Summary

Enter basic data source information

Set the basic configuration values of a datasource for association with your JDBC provider. A datasource supplies the physical connections between the application server and the database.

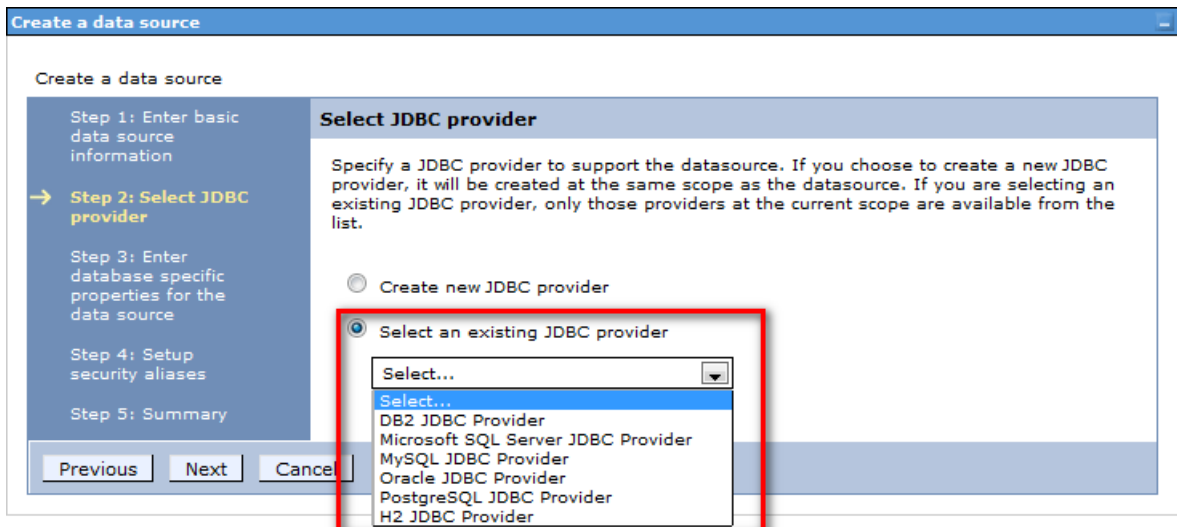
Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification.

Scope: cells:HONZANode01Cell:nodes:QuartzDeskNode

- * Data source name: QuartzDeskDS
- * JNDI name: jdbc/QuartzDeskDS

Next Cancel

In Step 2, select the JDBC provider created in 4.3.



The following steps depend on the selected JDBC provider.

4.5.1 DB2

In Step 3, provide these values:

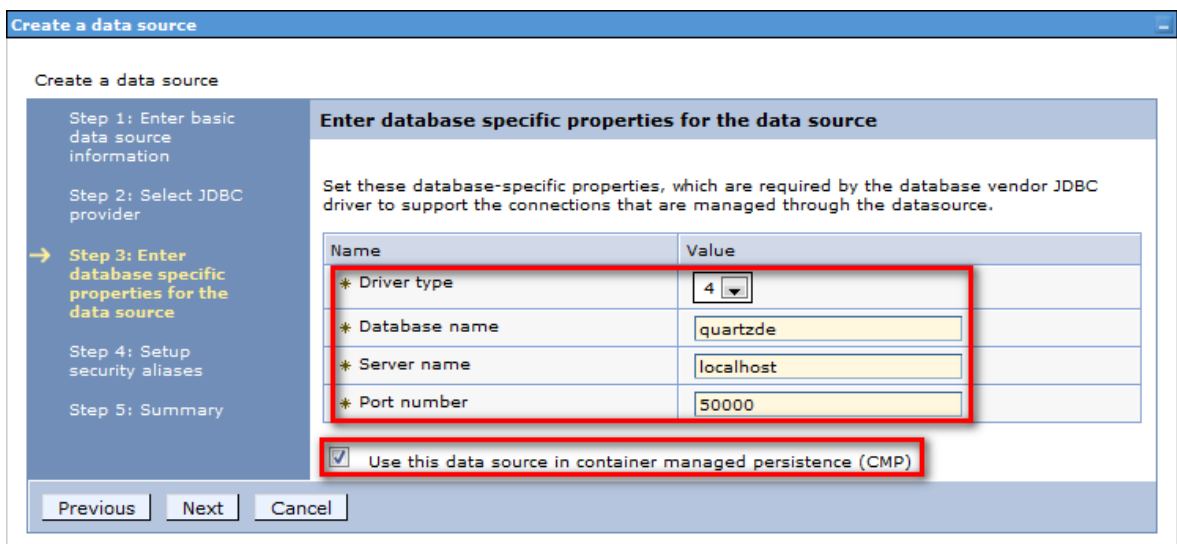
Driver type: 4

Database name: DB_NAME

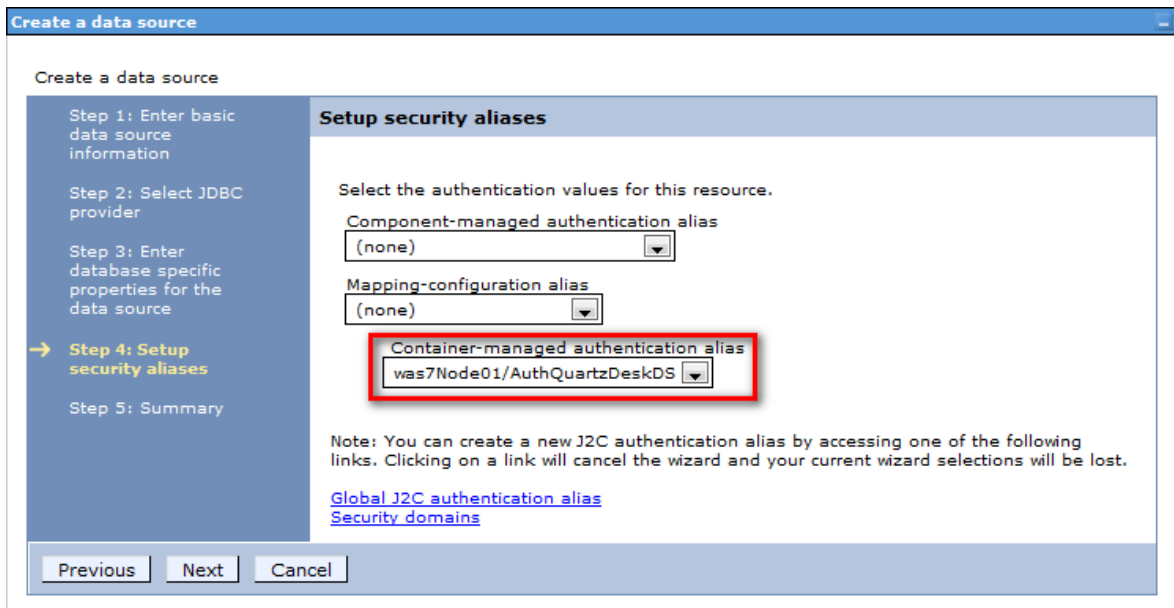
Server name: DB_HOST

Port number: DB_PORT

Leave the “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Create a data source

Step 1: Enter basic data source information

Step 2: Select JDBC provider

Step 3: Enter database specific properties for the data source

→ Step 4: Setup security aliases

Step 5: Summary

Setup security aliases

Select the authentication values for this resource.

Component-managed authentication alias
(none)

Mapping-configuration alias
(none)

Container-managed authentication alias
was7Node01/AuthQuartzDeskDS

Note: You can create a new J2C authentication alias by accessing one of the following links. Clicking on a link will cancel the wizard and your current wizard selections will be lost.

[Global J2C authentication alias](#)
[Security domains](#)

Previous Next Cancel

Click Next and then Finish. Save changes.

Edit the created data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:

Statement cache size: 100

Validate new connectons: checked

Number of retries: 100

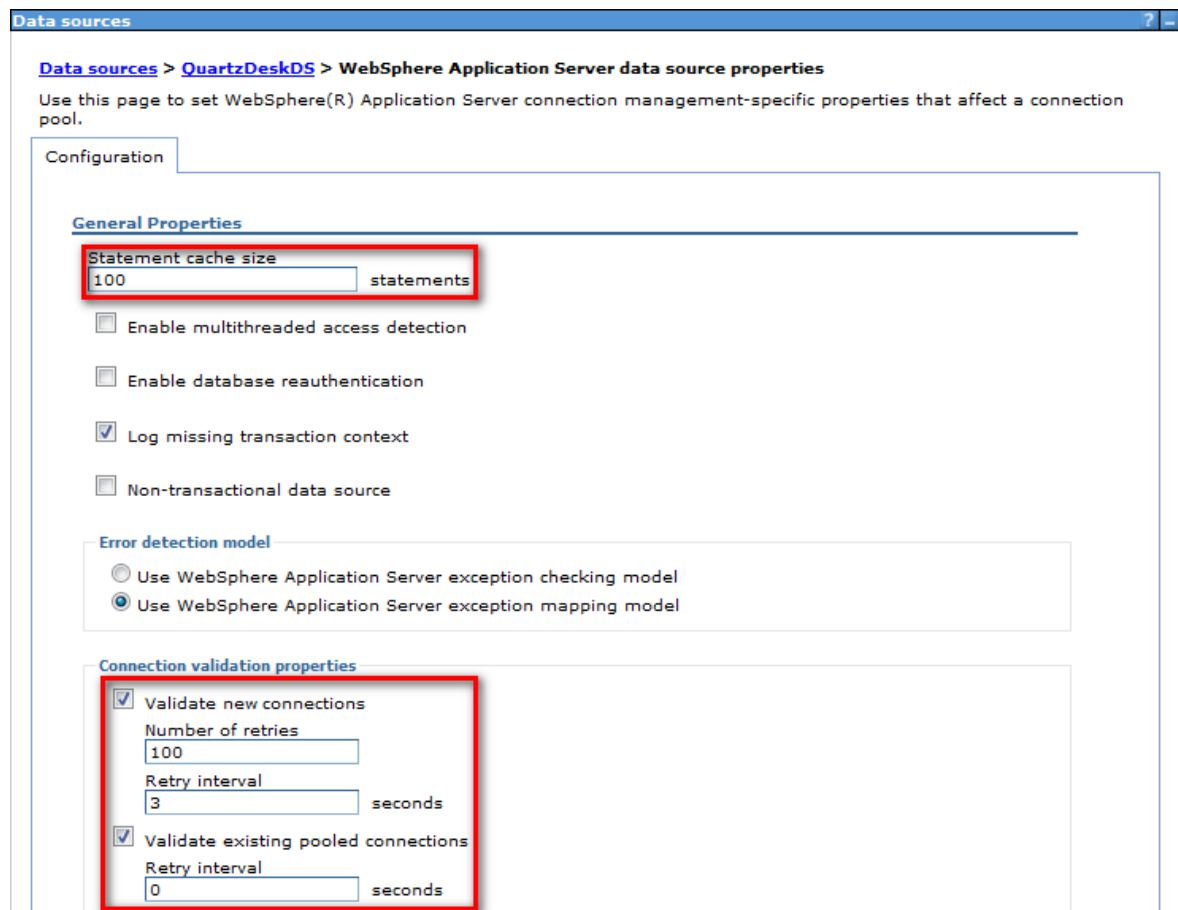
Retry Interval: 3

Validate existing pooled connections: checked

Retry interval: 0

Select "Validation by SQL query" with the following query:

```
select 1 from sysibm.sysdummy1
```



Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

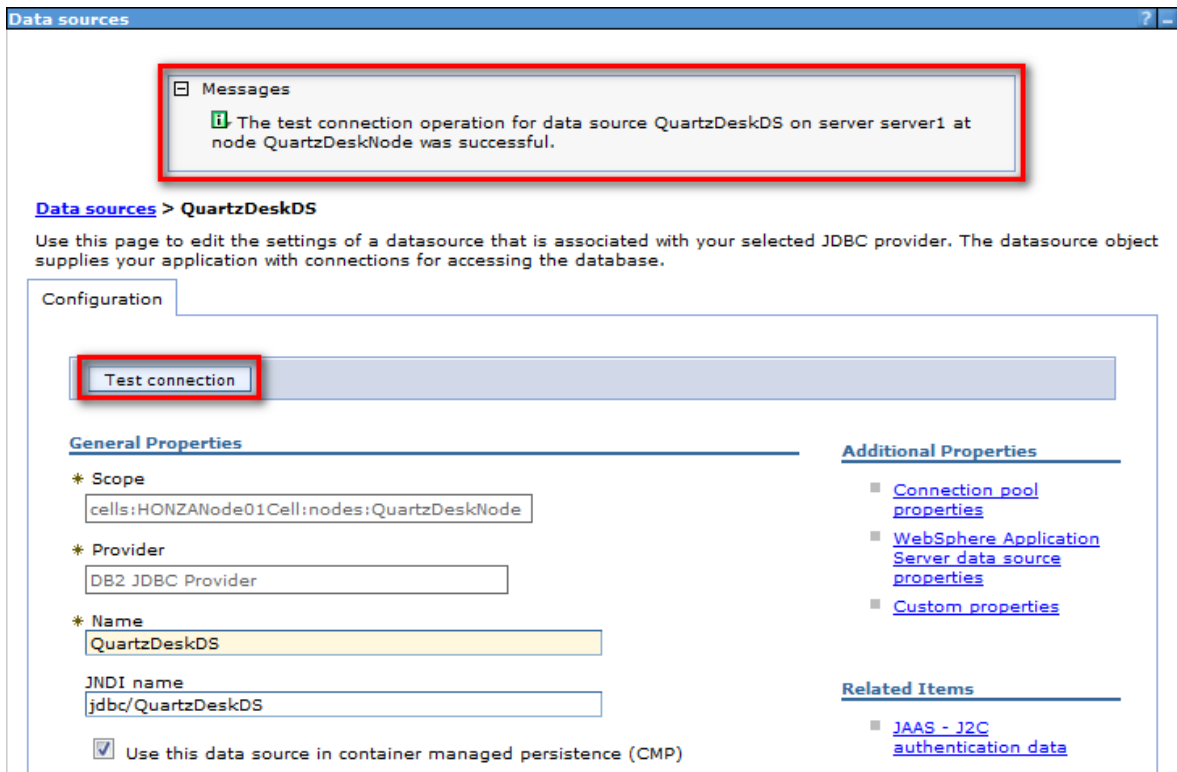
Name: clientApplicationInformation

Value: QuartzDesk

Apply and Save changes.

Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.





Check there are no errors displayed.

4.5.2 H2



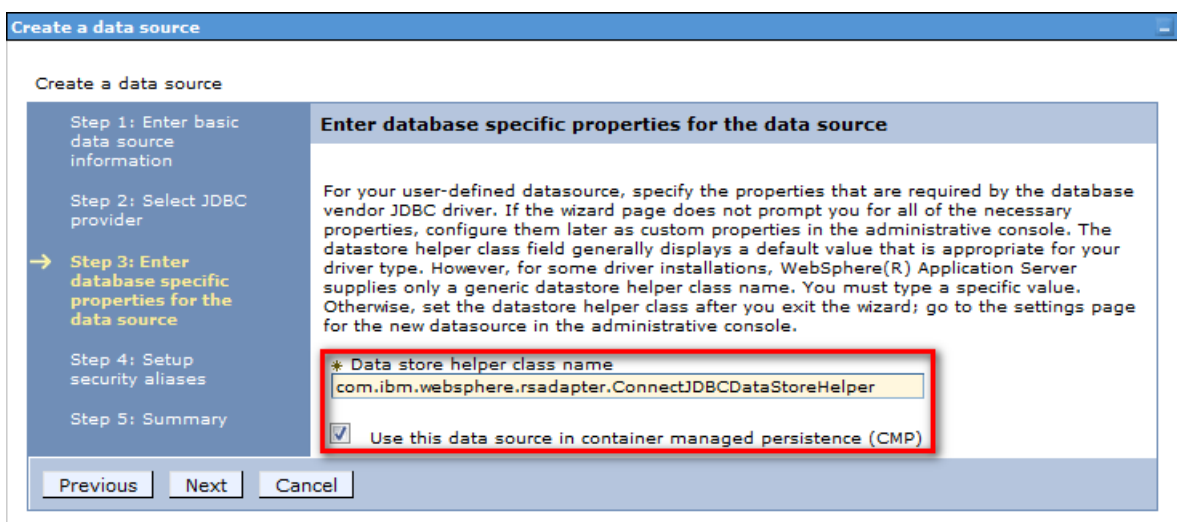
We recommend using H2 for evaluation and/or experimental purposes only. We strongly discourage using H2 in production environments.

In Step 3, change the default data store helper class.

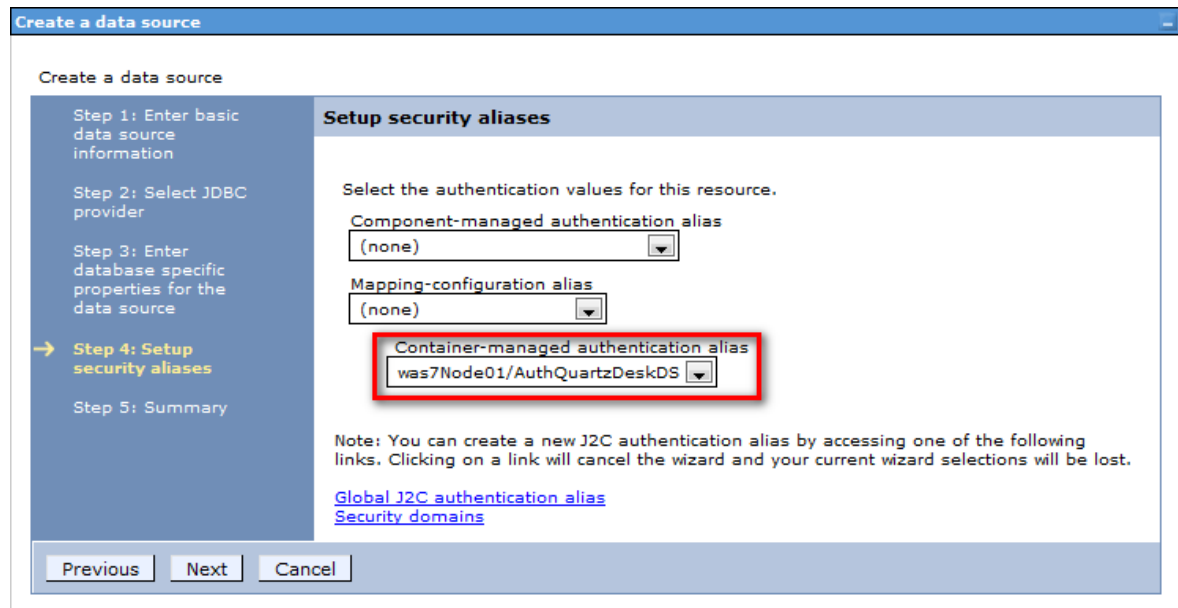
Data store helper class name:

`com.ibm.websphere.rsadapter.ConnectJDBCDataStoreHelper`

Leave the “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Click Next and then Finish. Save changes.

Edit the data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:

Statement cache size: 100

Validate new connectons: checked

Number of retries: 100

Retry Interval: 3

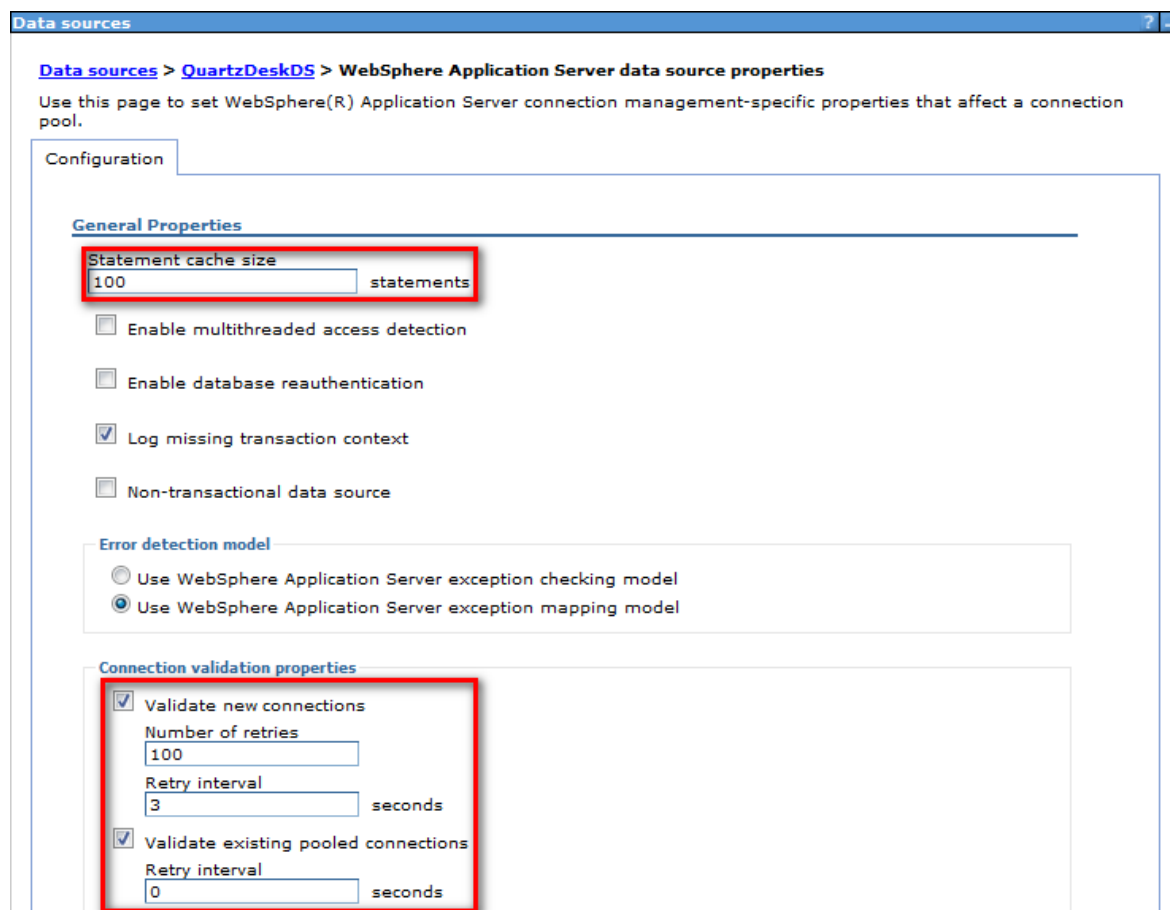
Validate existing pooled connections: checked

Retry interval: 0

Select "Validation by SQL query" with the following query:

```
select 1
```





Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

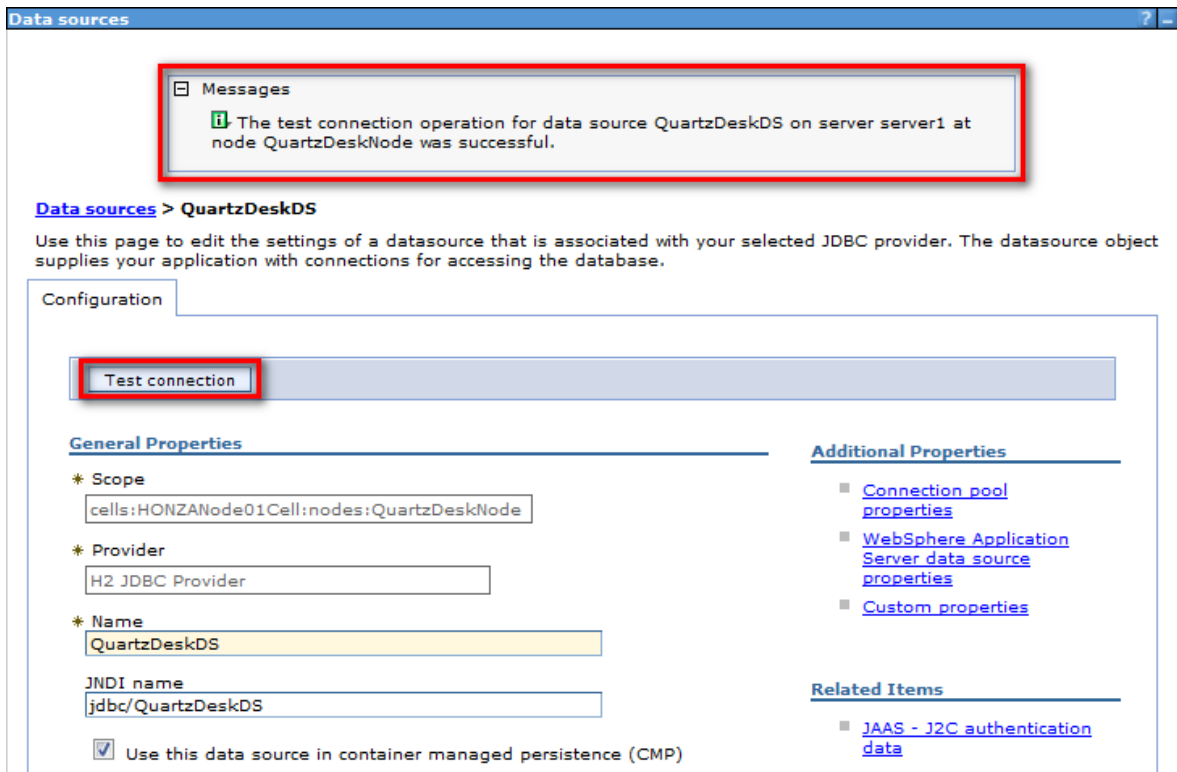
Name: URL

Value: jdbc:h2:file:<H2_DB_FILE_PATH>

Please note that H2 can be configured to run in various operating modes by adjusting the database URL value. For details, please refer to the H2 documentation at http://www.h2database.com/html/features.html#database_url.

Apply and Save changes.

Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



Check there are no errors displayed.

4.5.3 Microsoft SQL Server

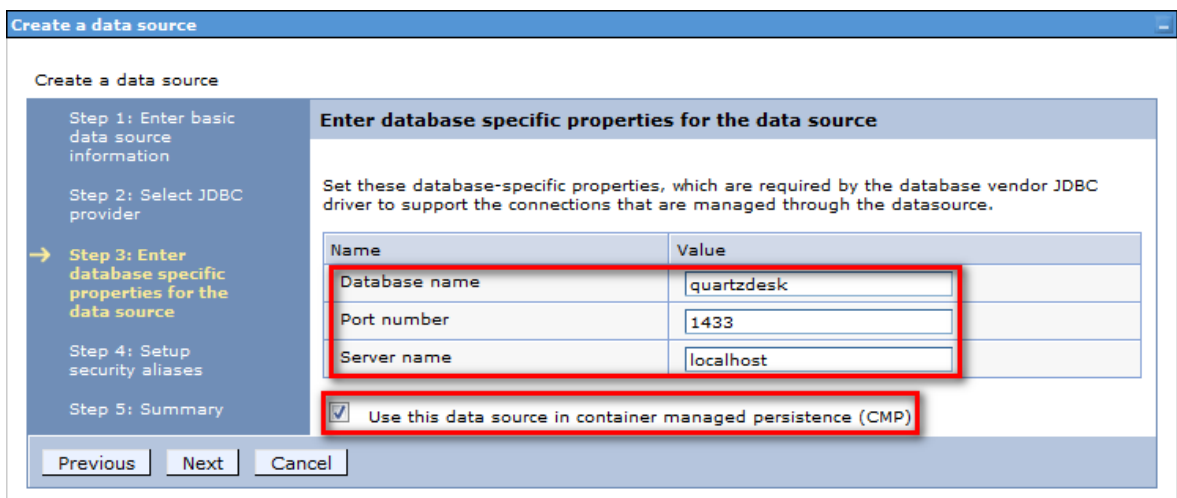
In Step 3, provide these values:

Database name: DB_NAME

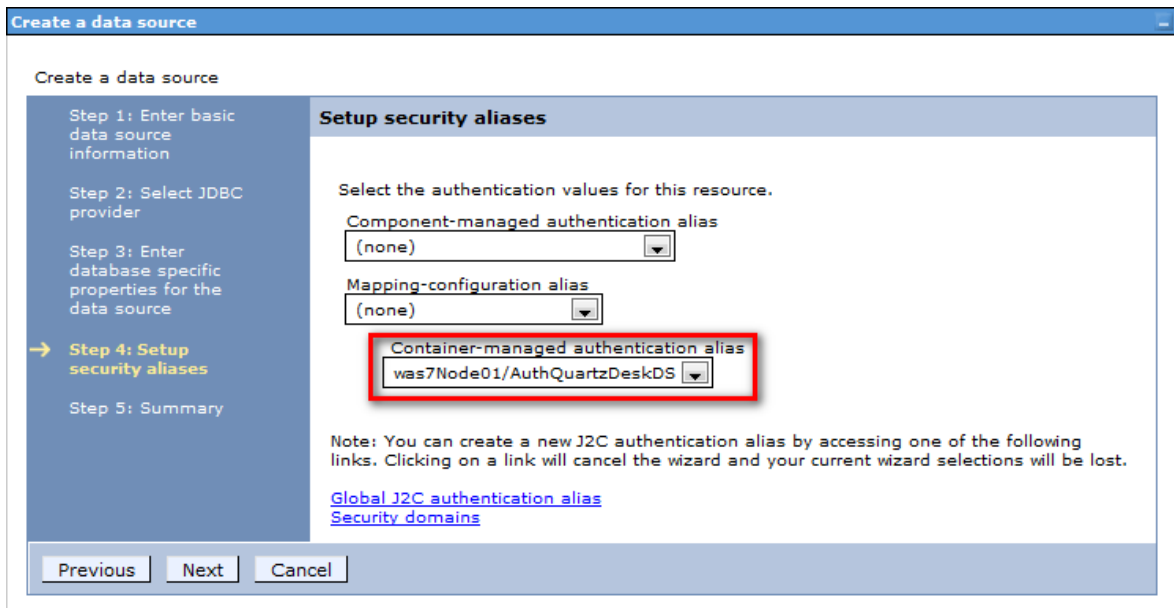
Server name: DB_HOST

Port number: DB_PORT

Leave the “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Click Next and then Finish. Save changes.

Edit the created data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:

Statement cache size: 100

Validate new connectons: checked

Number of retries: 100

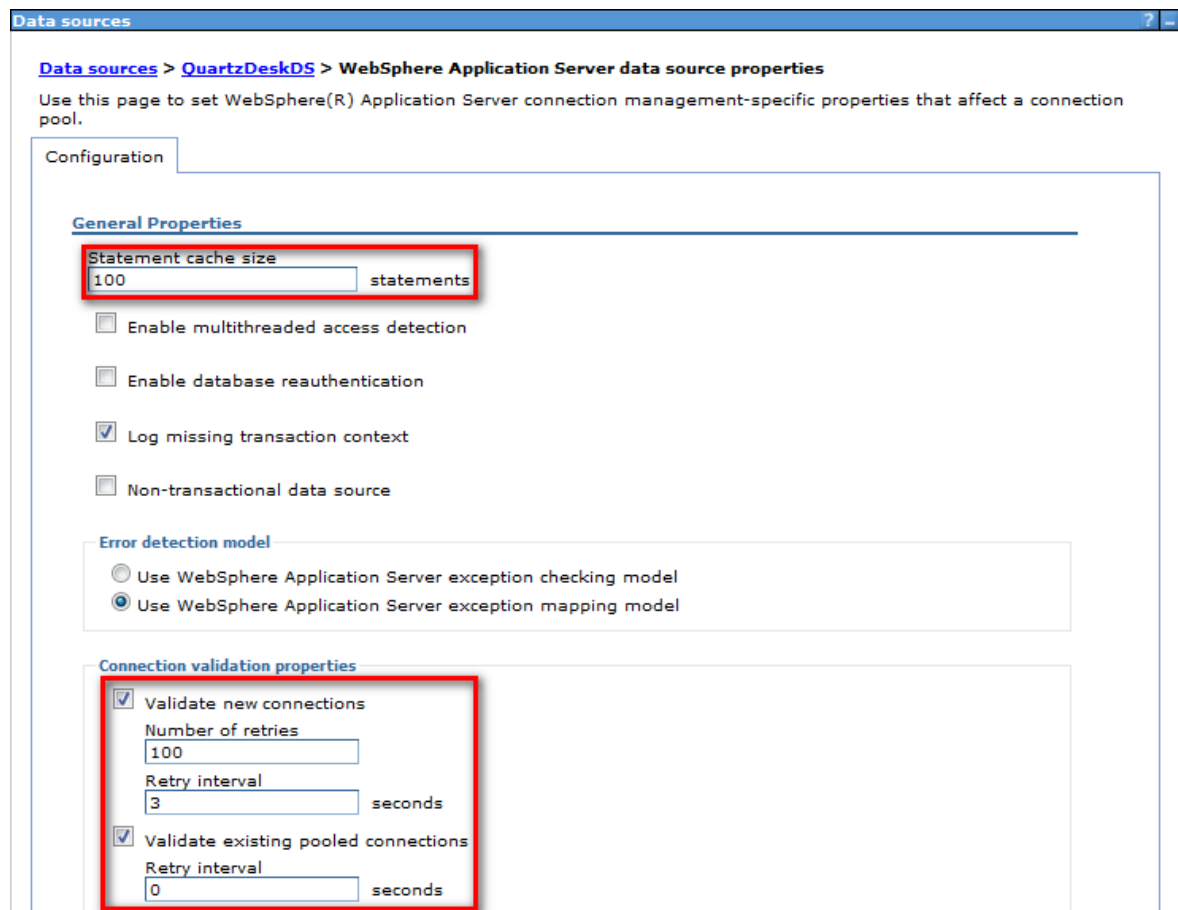
Retry Interval: 3

Validate existing pooled connections: checked

Retry interval: 0

Select "Validation by SQL query" with the following query:

```
select 1
```

Data sources > **QuartzDeskDS** > **WebSphere Application Server data source properties**

Use this page to set WebSphere(R) Application Server connection management-specific properties that affect a connection pool.

Configuration

General Properties

Statement cache size
100 statements

Enable multithreaded access detection

Enable database reauthentication

Log missing transaction context

Non-transactional data source

Error detection model

Use WebSphere Application Server exception checking model

Use WebSphere Application Server exception mapping model

Connection validation properties

Validate new connections

Number of retries
100

Retry interval
3 seconds

Validate existing pooled connections

Retry interval
0 seconds

Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

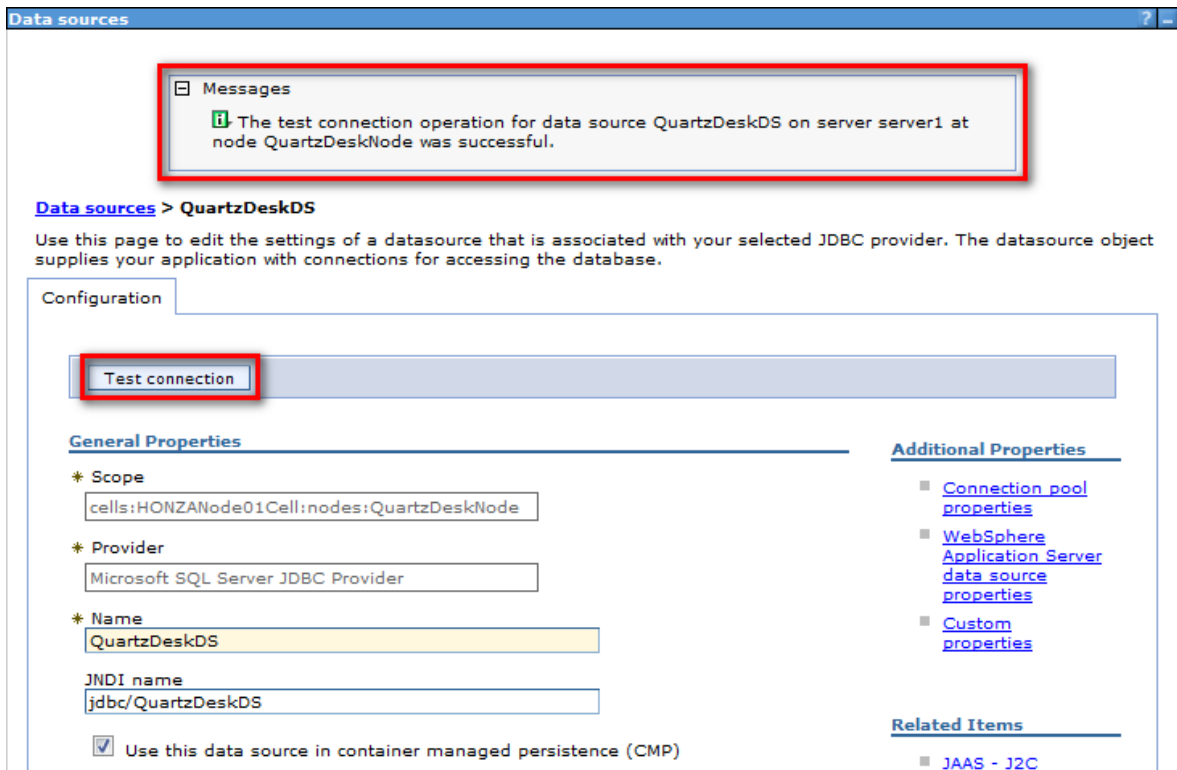
Set the following properties:

Name: applicationName
Value: QuartzDesk

Depending on your Microsoft SQL Server configuration, you may need to set the value of the instanceName property. Read the property description for details.

Apply and Save changes.

Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.

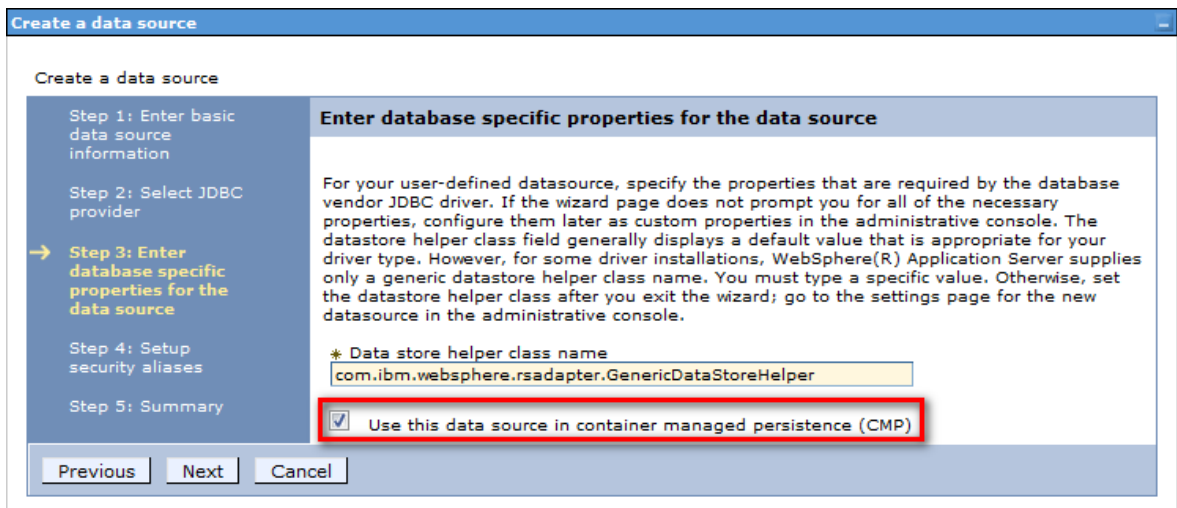


The screenshot shows the 'Data sources' page in the administrative console. At the top, a message box indicates: 'The test connection operation for data source QuartzDeskDS on server server1 at node QuartzDeskNode was successful.' Below this, the configuration for 'QuartzDeskDS' is shown. A 'Test connection' button is highlighted. The 'General Properties' section includes: Scope (cells:HONZANode01Cell:nodes:QuartzDeskNode), Provider (Microsoft SQL Server JDBC Provider), Name (QuartzDeskDS), and JNDI name (jdbc/QuartzDeskDS). A checkbox 'Use this data source in container managed persistence (CMP)' is checked. On the right, 'Additional Properties' includes links for 'Connection pool properties', 'WebSphere Application Server data source properties', and 'Custom properties'. 'Related Items' includes a link for 'JAAS - J2C'.

Check there are no errors displayed.

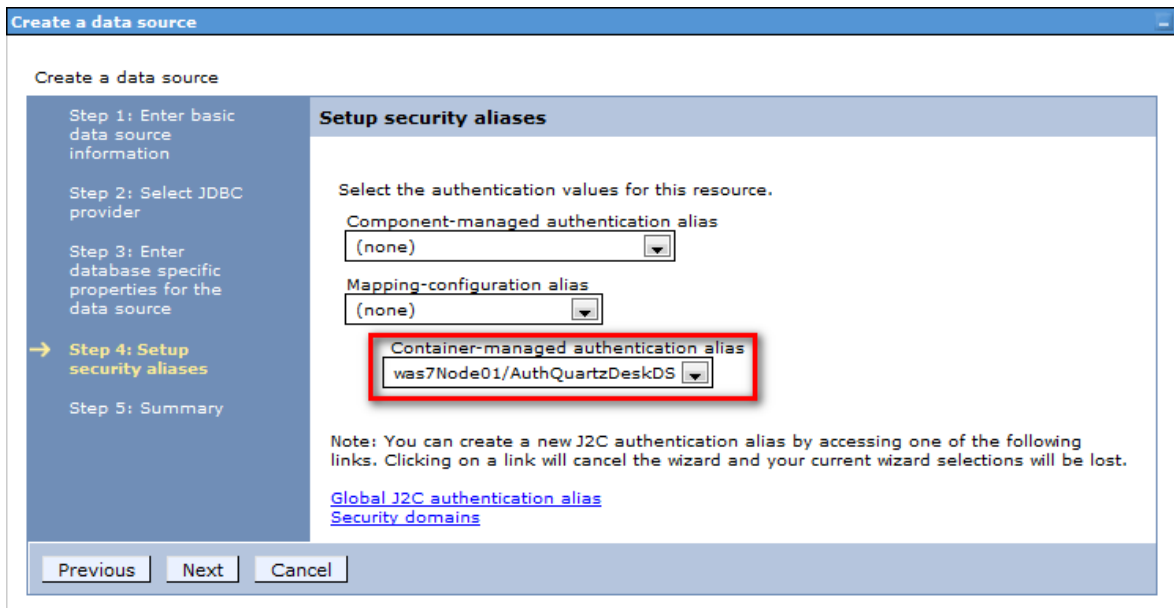
4.5.4 MySQL

In Step 3, leave the “Use this data source in container managed persistence (CMP)” checkbox checked.



The screenshot shows the 'Create a data source' wizard. The current step is 'Step 3: Enter database specific properties for the data source'. The wizard is divided into two main sections. The left section shows the progress: Step 1: Enter basic data source information, Step 2: Select JDBC provider, Step 3: Enter database specific properties for the data source (highlighted with a yellow arrow), Step 4: Setup security aliases, and Step 5: Summary. The right section, titled 'Enter database specific properties for the data source', contains instructions: 'For your user-defined datasource, specify the properties that are required by the database vendor JDBC driver. If the wizard page does not prompt you for all of the necessary properties, configure them later as custom properties in the administrative console. The datastore helper class field generally displays a default value that is appropriate for your driver type. However, for some driver installations, WebSphere(R) Application Server supplies only a generic datastore helper class name. You must type a specific value. Otherwise, set the datastore helper class after you exit the wizard; go to the settings page for the new datasource in the administrative console.' Below the instructions, there is a field for '* Data store helper class name' with the value 'com.ibm.websphere.rsadapter.GenericDataStoreHelper'. A checkbox 'Use this data source in container managed persistence (CMP)' is checked and highlighted with a red box. At the bottom, there are 'Previous', 'Next', and 'Cancel' buttons.

In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Click Next and then Finish. Save changes.

Edit the data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:

Statement cache size: 100

Validate new connectons: checked

Number of retries: 100

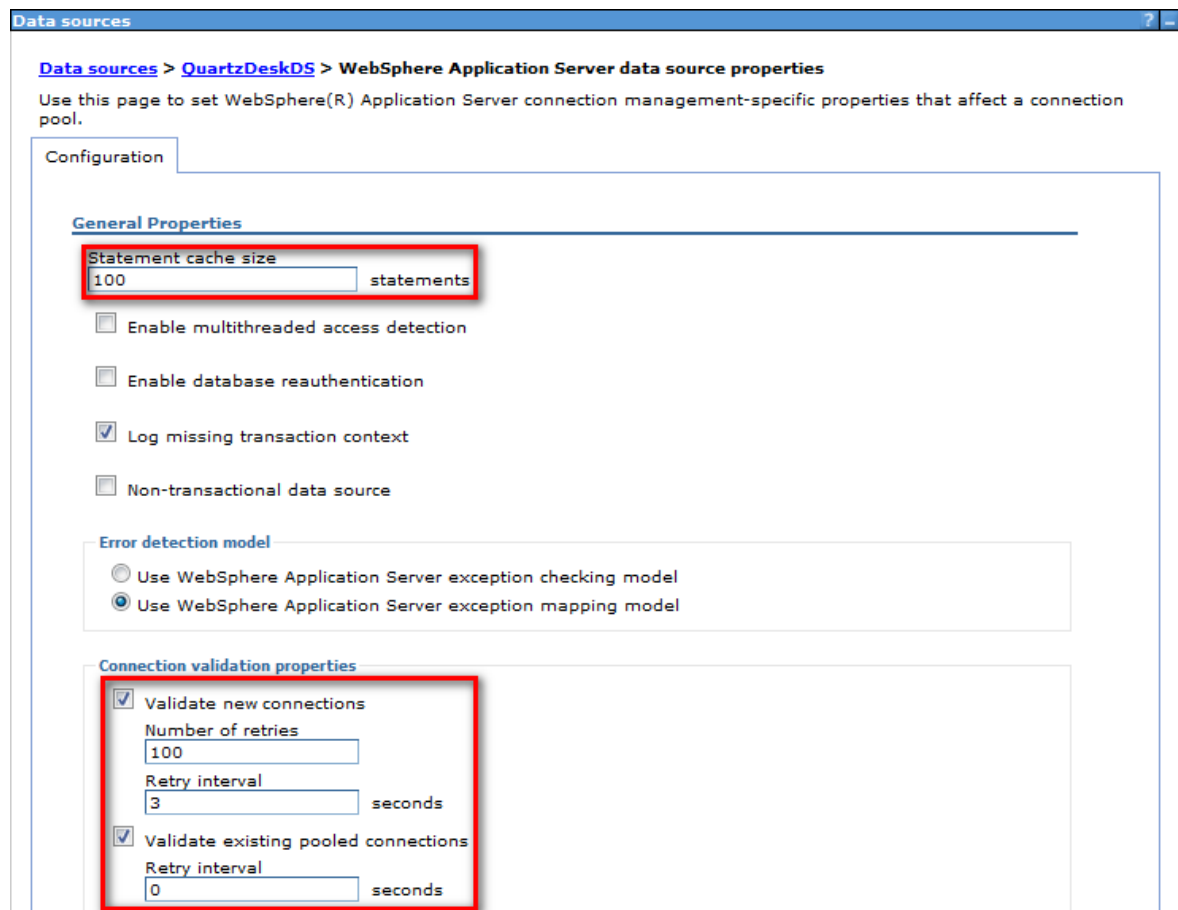
Retry Interval: 3

Validate existing pooled connections: checked

Retry interval: 0

Select "Validation by SQL query" with the following query:

```
select 1
```



Data sources > **QuartzDeskDS** > **WebSphere Application Server data source properties**

Use this page to set WebSphere(R) Application Server connection management-specific properties that affect a connection pool.

Configuration

General Properties

Statement cache size
100 statements

Enable multithreaded access detection

Enable database reauthentication

Log missing transaction context

Non-transactional data source

Error detection model

Use WebSphere Application Server exception checking model

Use WebSphere Application Server exception mapping model

Connection validation properties

Validate new connections

Number of retries
100

Retry interval
3 seconds

Validate existing pooled connections

Retry interval
0 seconds

Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

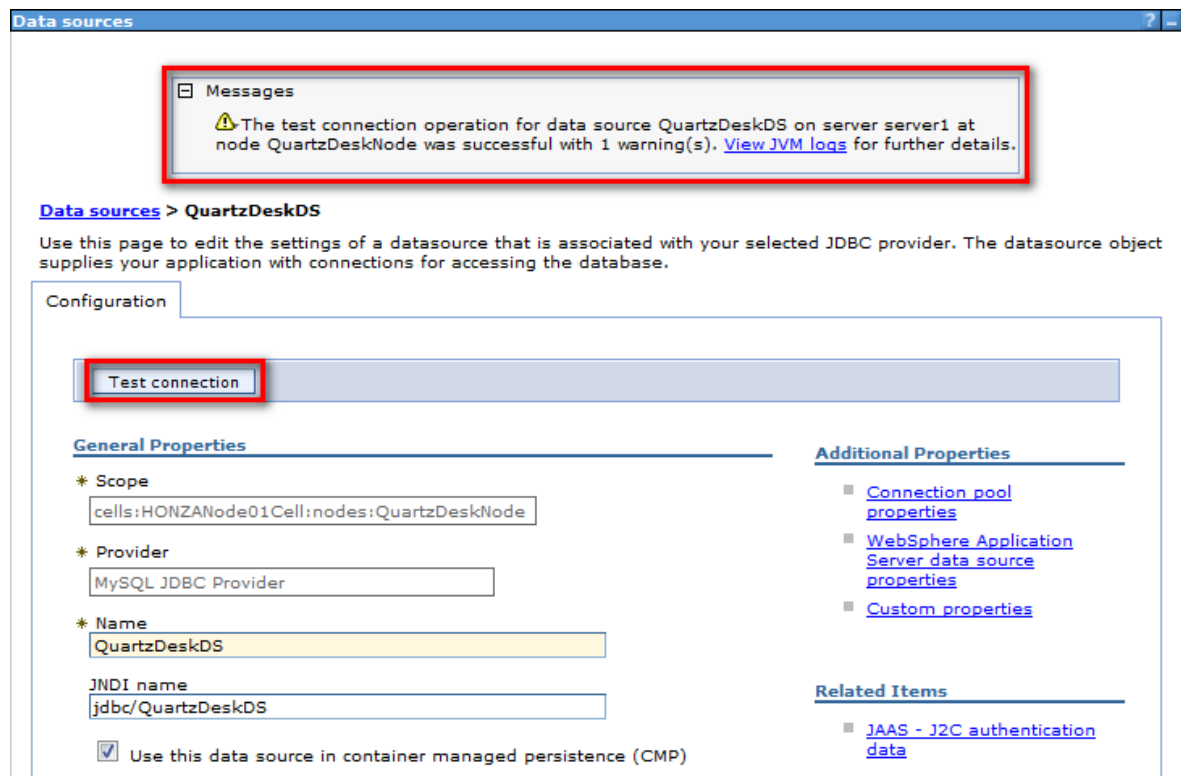
Name: databaseName
Value: DB_NAME

Name: serverName
Value: DB_HOST

Name: portNumber
Value: DB_PORT

Apply and Save changes.

Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



Check there are no errors displayed.



When testing the connection of a data source using a JDBC provider with the “Database type” option set to “User type”, there may be a warning message displayed. In the JVM logs, the following message is logged:

```
DSRA0174W: Warning: GenericDataStoreHelper is being used.
```

You can ignore this warning safely.

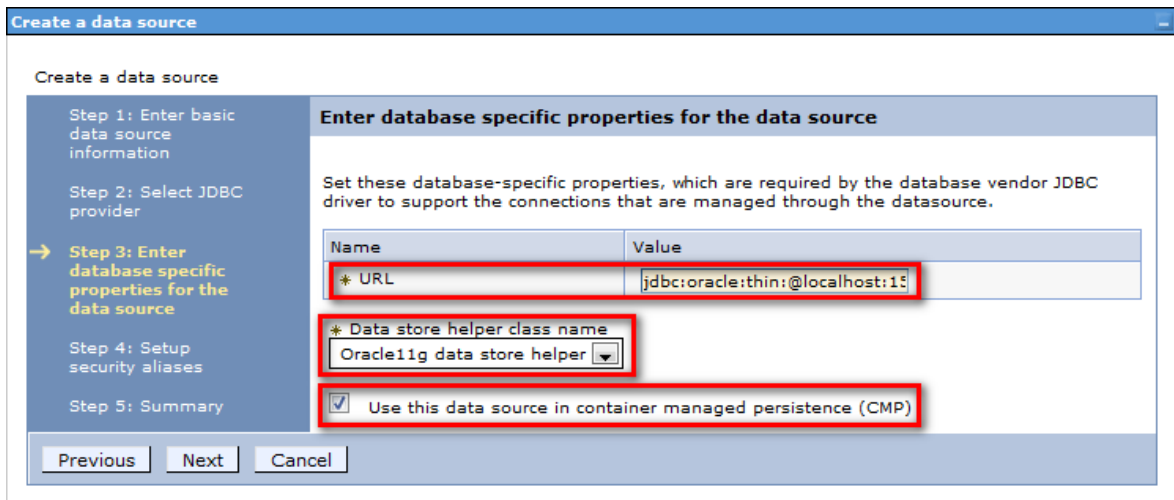
4.5.5 Oracle

In Step 3, provide the URL value:

URL: jdbc:oracle:thin:@DB_HOST:DB_PORT

Select the “Data store helper class name” based on your Oracle database version.

Leave the “Use this data source in container managed persistence (CMP)” checkbox checked.



Create a data source

Step 1: Enter basic data source information
Step 2: Select JDBC provider
→ **Step 3: Enter database specific properties for the data source**
Step 4: Setup security aliases
Step 5: Summary

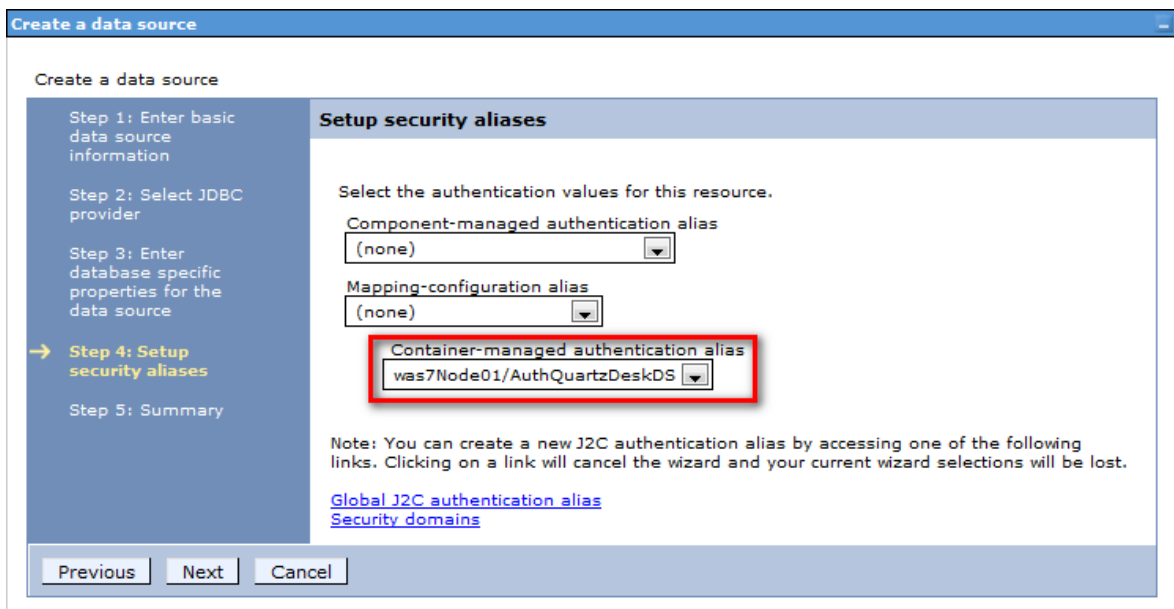
Enter database specific properties for the data source

Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource.

Name	Value
* URL	jdbc:oracle:thin:@localhost:15
* Data store helper class name	Oracle11g data store helper
<input checked="" type="checkbox"/> Use this data source in container managed persistence (CMP)	

Previous Next Cancel

In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Create a data source

Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 3: Enter database specific properties for the data source
→ **Step 4: Setup security aliases**
Step 5: Summary

Setup security aliases

Select the authentication values for this resource.

Component-managed authentication alias
(none)

Mapping-configuration alias
(none)

**Container-managed authentication alias
was7Node01/AuthQuartzDeskDS**

Note: You can create a new J2C authentication alias by accessing one of the following links. Clicking on a link will cancel the wizard and your current wizard selections will be lost.

[Global J2C authentication alias](#)
[Security domains](#)

Previous Next Cancel

Click Next and then Finish. Save changes.

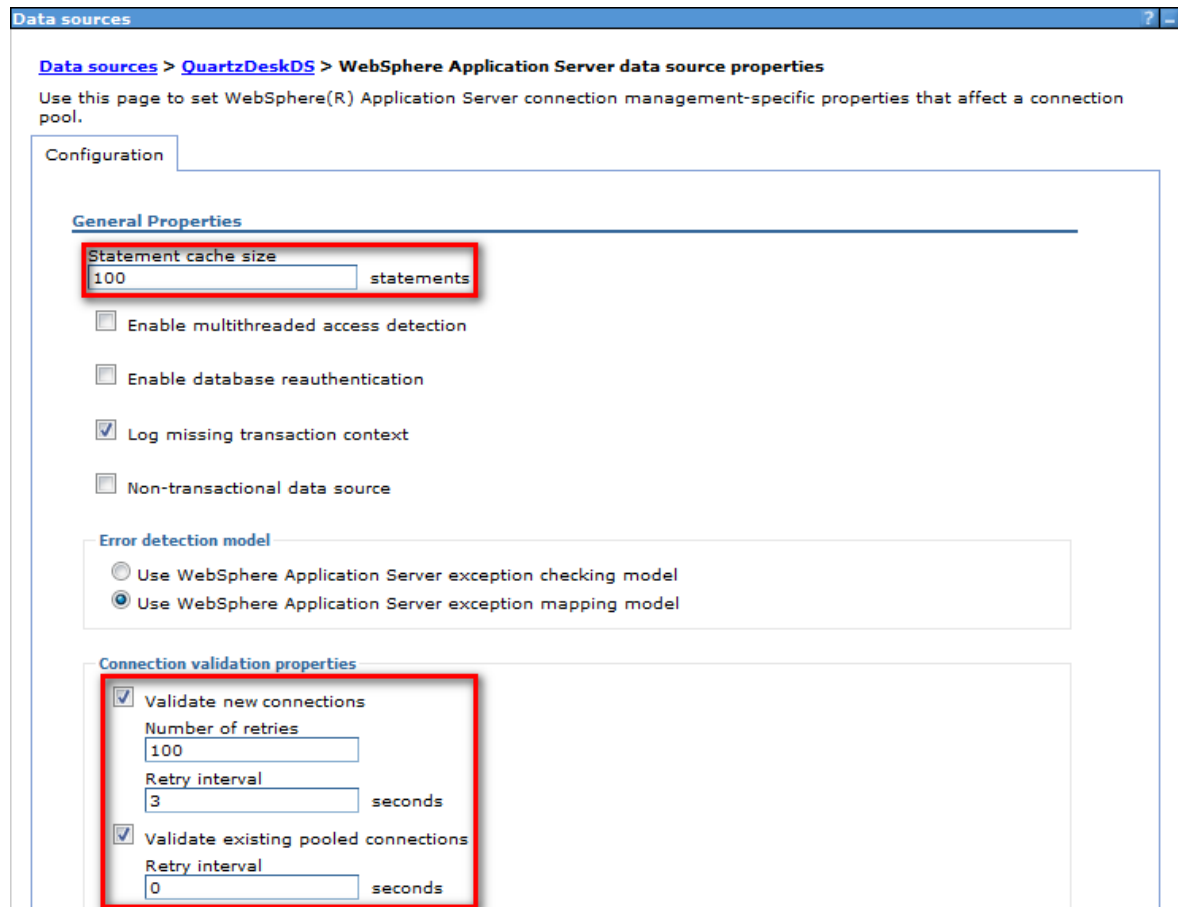
Edit the created data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:
Statement cache size: 100

Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select "Validation by SQL query" with the following query:
select 1 from dual



Data sources > **QuartzDeskDS** > **WebSphere Application Server data source properties**

Use this page to set WebSphere(R) Application Server connection management-specific properties that affect a connection pool.

Configuration

General Properties

Statement cache size
100 statements

Enable multithreaded access detection

Enable database reauthentication

Log missing transaction context

Non-transactional data source

Error detection model

Use WebSphere Application Server exception checking model

Use WebSphere Application Server exception mapping model

Connection validation properties

Validate new connections

Number of retries
100

Retry interval
3 seconds

Validate existing pooled connections

Retry interval
0 seconds

Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

Name: driverType
Value: thin

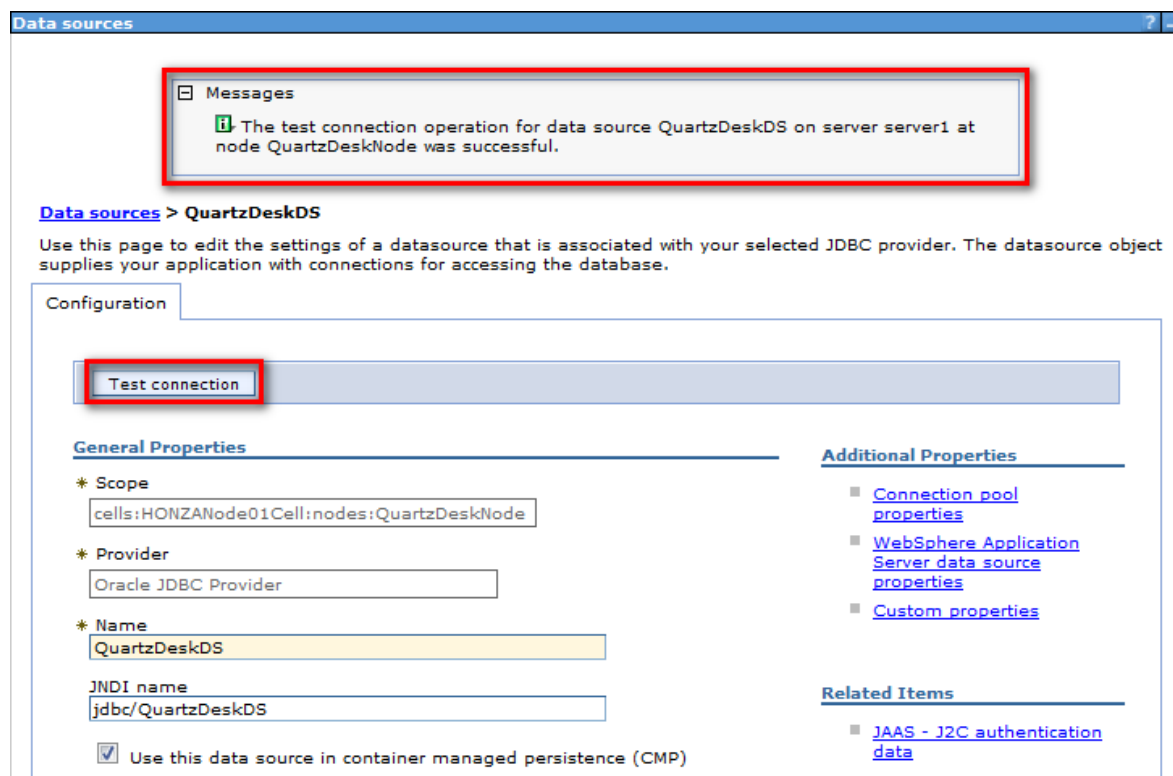
Name: databaseName
Value: DB_NAME

Name: serverName
Value: DB_HOST

Name: portNumber
Value: DB_PORT

Apply and Save changes.

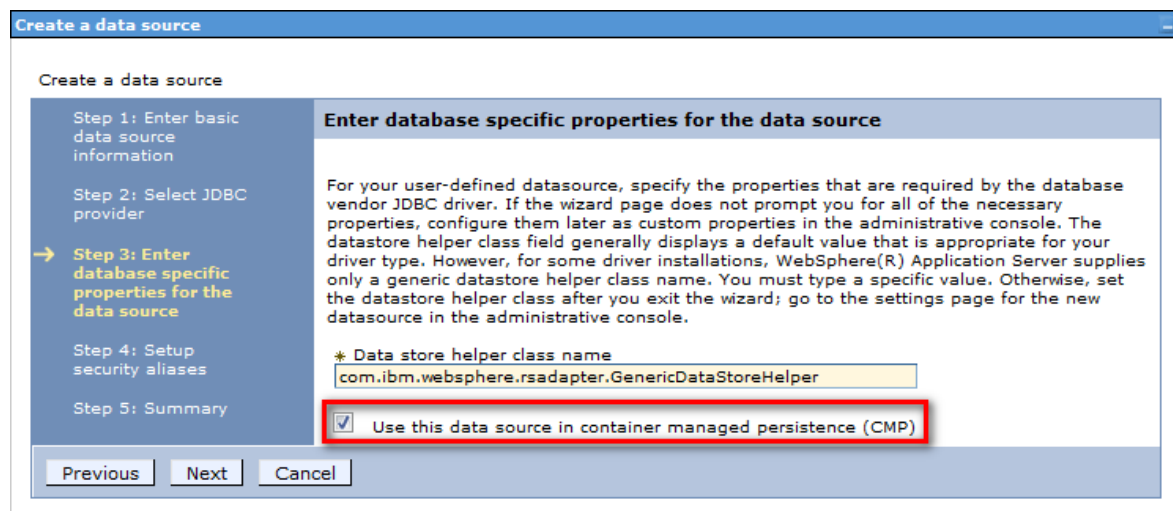
Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



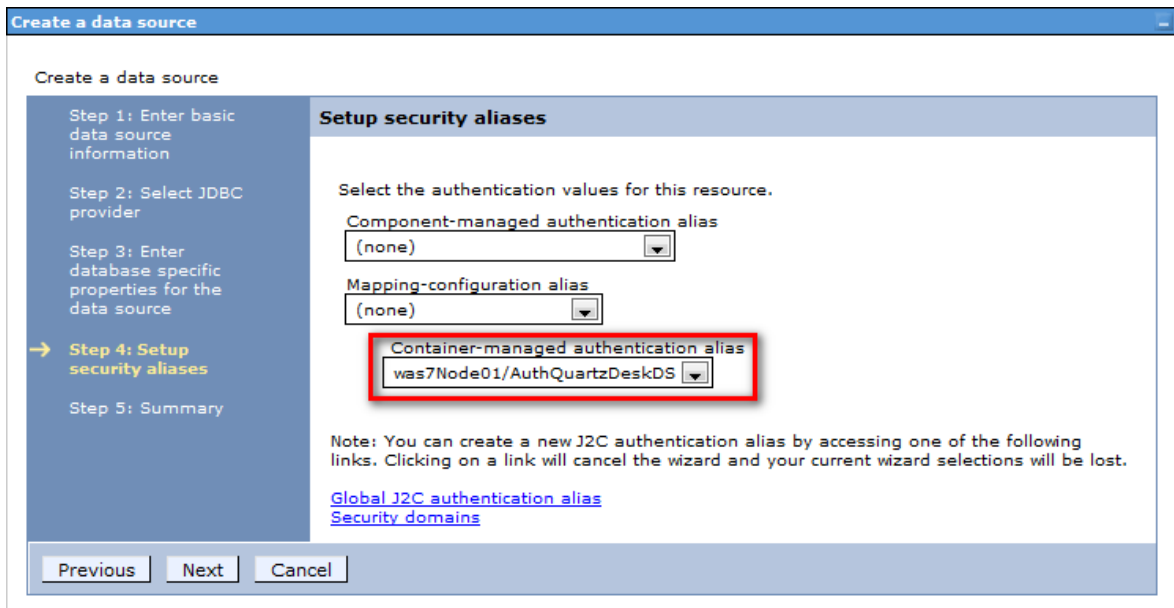
Check there are no errors displayed.

4.5.6 PostgreSQL

In Step 3, leave the “Use this data source in container managed persistence (CMP)” checkbox checked.



In Step 4, select the J2C authentication alias created in 4.4 as the Container-managed authentication alias.



Create a data source

Step 1: Enter basic data source information

Step 2: Select JDBC provider

Step 3: Enter database specific properties for the data source

→ Step 4: Setup security aliases

Step 5: Summary

Setup security aliases

Select the authentication values for this resource.

Component-managed authentication alias
(none)

Mapping-configuration alias
(none)

Container-managed authentication alias
was7Node01/AuthQuartzDeskDS

Note: You can create a new J2C authentication alias by accessing one of the following links. Clicking on a link will cancel the wizard and your current wizard selections will be lost.

[Global J2C authentication alias](#)
[Security domains](#)

Previous Next Cancel

Click Next and then Finish. Save changes.

Edit the data source properties by going to Resources → JDBC → Data sources → QuartzDeskDS → WebSphere Application Server data source properties.

Set the following properties:

Statement cache size: 100

Validate new connectons: checked

Number of retries: 100

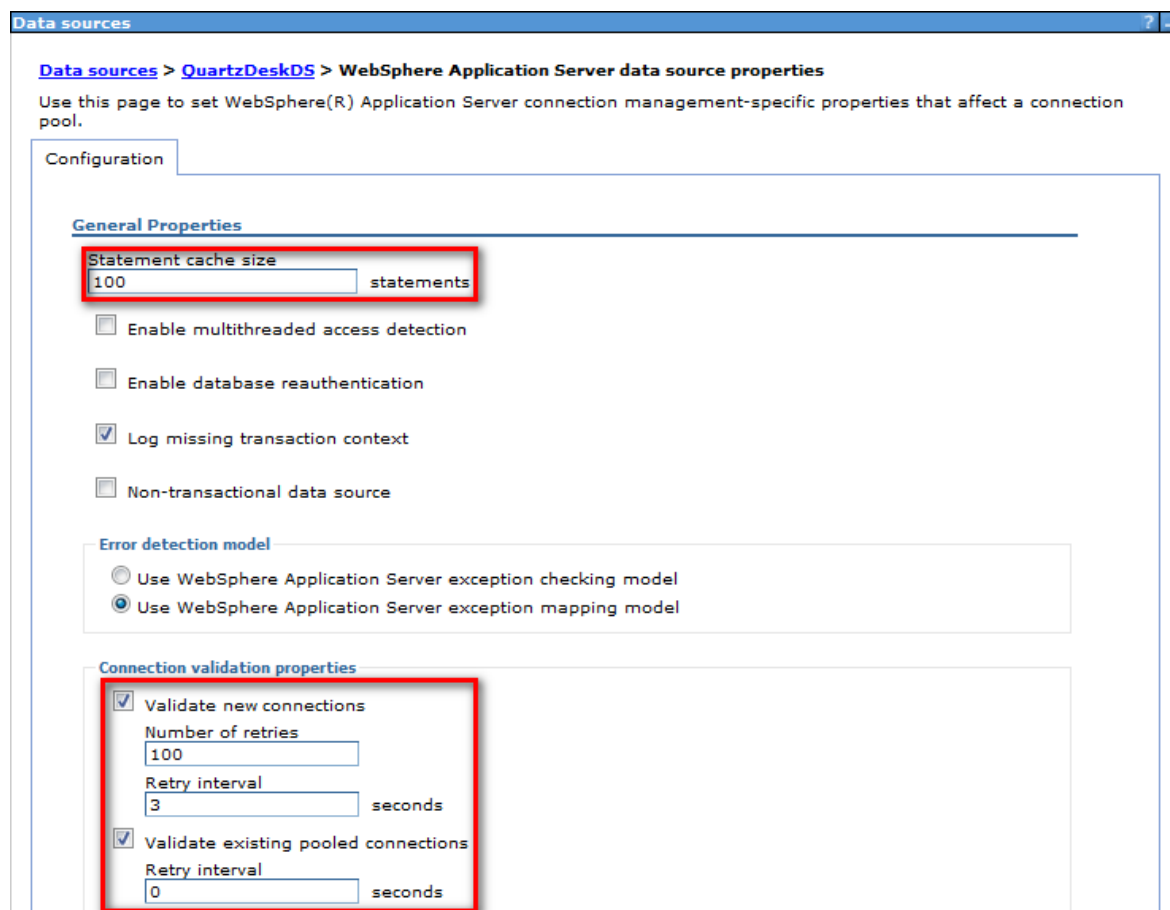
Retry Interval: 3

Validate existing pooled connections: checked

Retry interval: 0

Select "Validation by SQL query" with the following query:

```
select 1
```



Data sources > **QuartzDeskDS** > **WebSphere Application Server data source properties**

Use this page to set WebSphere(R) Application Server connection management-specific properties that affect a connection pool.

Configuration

General Properties

Statement cache size
100 statements

Enable multithreaded access detection

Enable database reauthentication

Log missing transaction context

Non-transactional data source

Error detection model

Use WebSphere Application Server exception checking model

Use WebSphere Application Server exception mapping model

Connection validation properties

Validate new connections

Number of retries
100

Retry interval
3 seconds

Validate existing pooled connections

Retry interval
0 seconds

Apply and Save changes.

Edit the data source custom properties by going to Resources → JDBC → Data sources → QuartzDeskDS → Custom properties.

Set the following properties:

Name: applicationName
Value: QuartzDesk

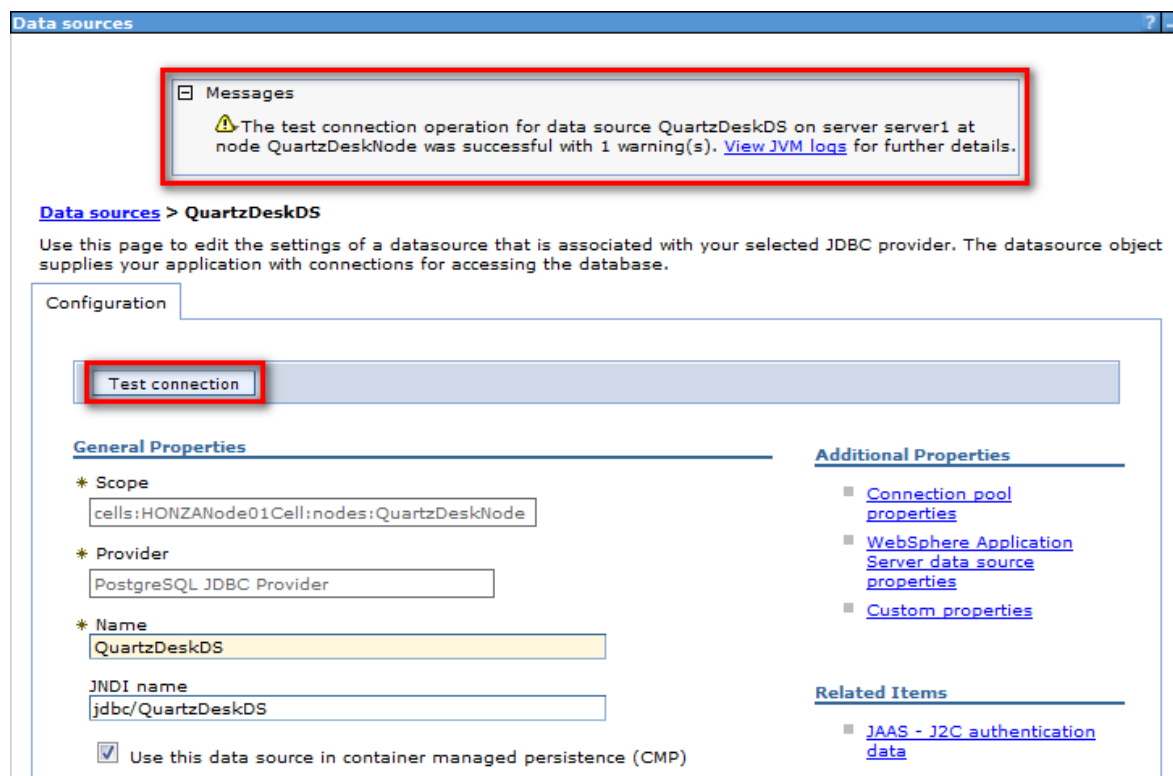
Name: databaseName
Value: DB_NAME

Name: serverName
Value: DB_HOST

Name: portNumber
Value: DB_PORT

Apply and Save changes.

Test the created data source by clicking the “Test connection” button under Resources → JDBC → Data sources → QuartzDeskDS.



Data sources

Messages

⚠ The test connection operation for data source QuartzDeskDS on server server1 at node QuartzDeskNode was successful with 1 warning(s). [View JVM logs](#) for further details.

[Data sources](#) > QuartzDeskDS

Use this page to edit the settings of a datasource that is associated with your selected JDBC provider. The datasource object supplies your application with connections for accessing the database.

Configuration

Test connection

General Properties

- * Scope: cells:HONZANode01Cell:nodes:QuartzDeskNode
- * Provider: PostgreSQL JDBC Provider
- * Name: QuartzDeskDS
- JNDI name: jdbc/QuartzDeskDS
- Use this data source in container managed persistence (CMP)

Additional Properties

- Connection pool properties
- WebSphere Application Server data source properties
- Custom properties

Related Items

- JAAS - J2C authentication data

Check there are no errors displayed.



When testing connection of a data source using a JDBC provider with the “Database type” option set to “User type”, there may be a warning message displayed. In the JVM logs, the following message is logged:

DSRA0174W: Warning: GenericDataStoreHelper is being used.

You can ignore this warning safely.

4.6 Application Work Directory

Create a QuartzDesk web application work directory (WORK_DIR) anywhere on the local file system. The directory must be readable and writable by the user the WAS process runs under.

Copy your QuartzDesk license key file (`license.key`) to WORK_DIR.



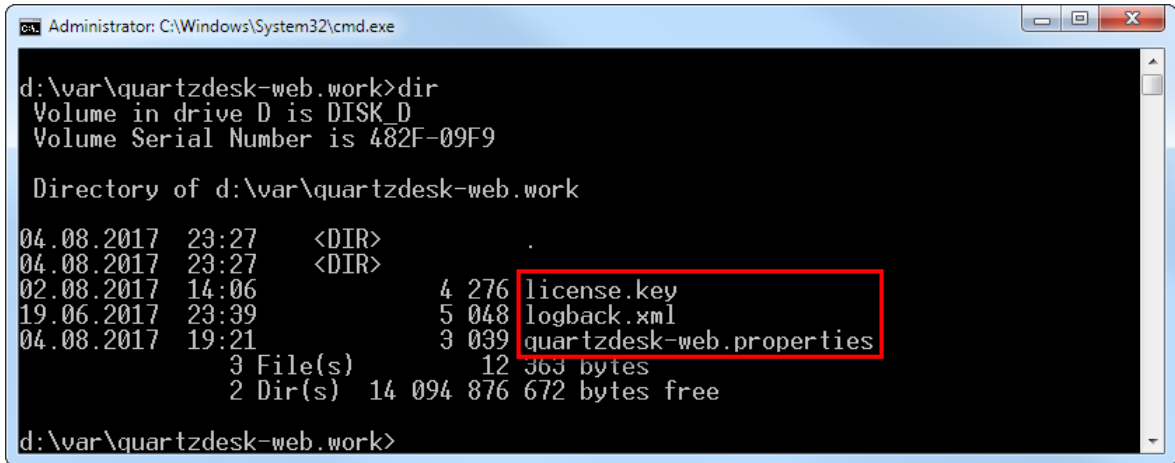
You can obtain a free 30-day trial license key at www.quartzdesk.com (go to Try / Purchase > Get Trial License Key).

Open the QuartzDesk web application archive (`quartzdesk-web-x.y.z.war`) and copy all files from the `extras/work` directory into WORK_DIR.



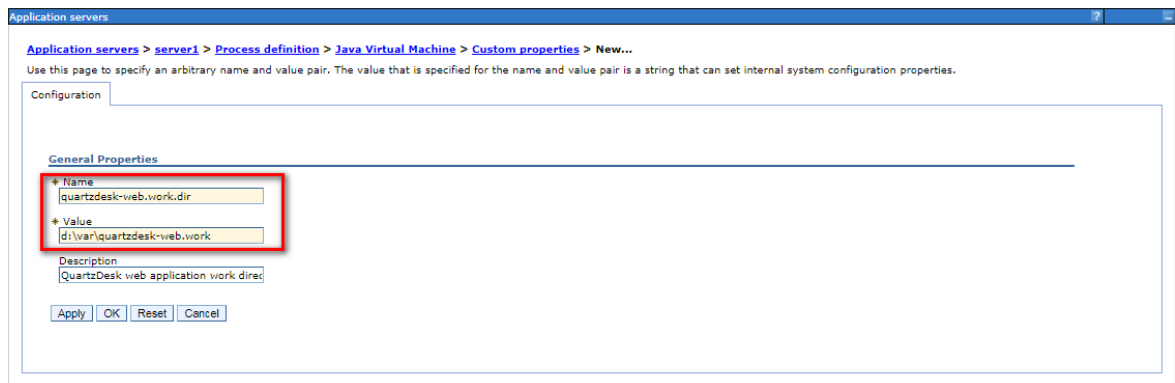
If you cannot open the WAR file directly, rename it to `*.zip`. Do not forget to rename the file back to `*.war` once you have extracted the required files.

In the following figure you can see an example of a QuartzDesk web application work directory correctly set up on a Microsoft Windows machine.



In WAC open Servers → Server Types → WebSphere application servers → WAS_SERVER_NAME → Java and Process Management → Process Definition → Java Virtual Machine → Custom Properties. Add a new JVM system property:

Name: quartzdesk-web.work.dir
Value: WORK_DIR



Apply and Save changes.

4.7 Application Configuration

Open the QuartzDesk web application configuration file WORK_DIR/quartzdesk-web.properties.

Based on the type and version of the database created in step 4.1, change the value of the db.profile configuration property according to the following table.

Database	Database Version	db.profile Value
DB2	>= 10.0	db2
H2	>= 1.3.170	h2
Microsoft SQL Server	>= 2008	mssql
MySQL (MyISAM)	>= 5.6	mysql
MySQL (InnoDB)	>= 5.6	mysql_innodb
Oracle	== 8i	oracle8
Oracle	>= 9i	oracle9
PostgreSQL	== 8.1	postgres81

PostgreSQL

>= 8.2

postgres82

Optionally, you can adjust the QuartzDesk web application logging parameters by editing the `WORK_DIR/logback.xml` configuration file. The default sample `logback.xml` configuration file makes QuartzDesk web application log under the `WORK_DIR/logs` directory that is automatically created when the web application starts. Please refer to the [Logback Manual](#) for Logback configuration details.

4.8 Deploy Application



Before deploying QuartzDesk web application, please remove the `WEB-INF/lib/jboss-transaction-api_<version>.jar` library from the `quartzdesk-web-x.y.z.war` file. This is required to avoid class loading conflicts between the bundled JTA API and the JTA API provided by WAS runtime.

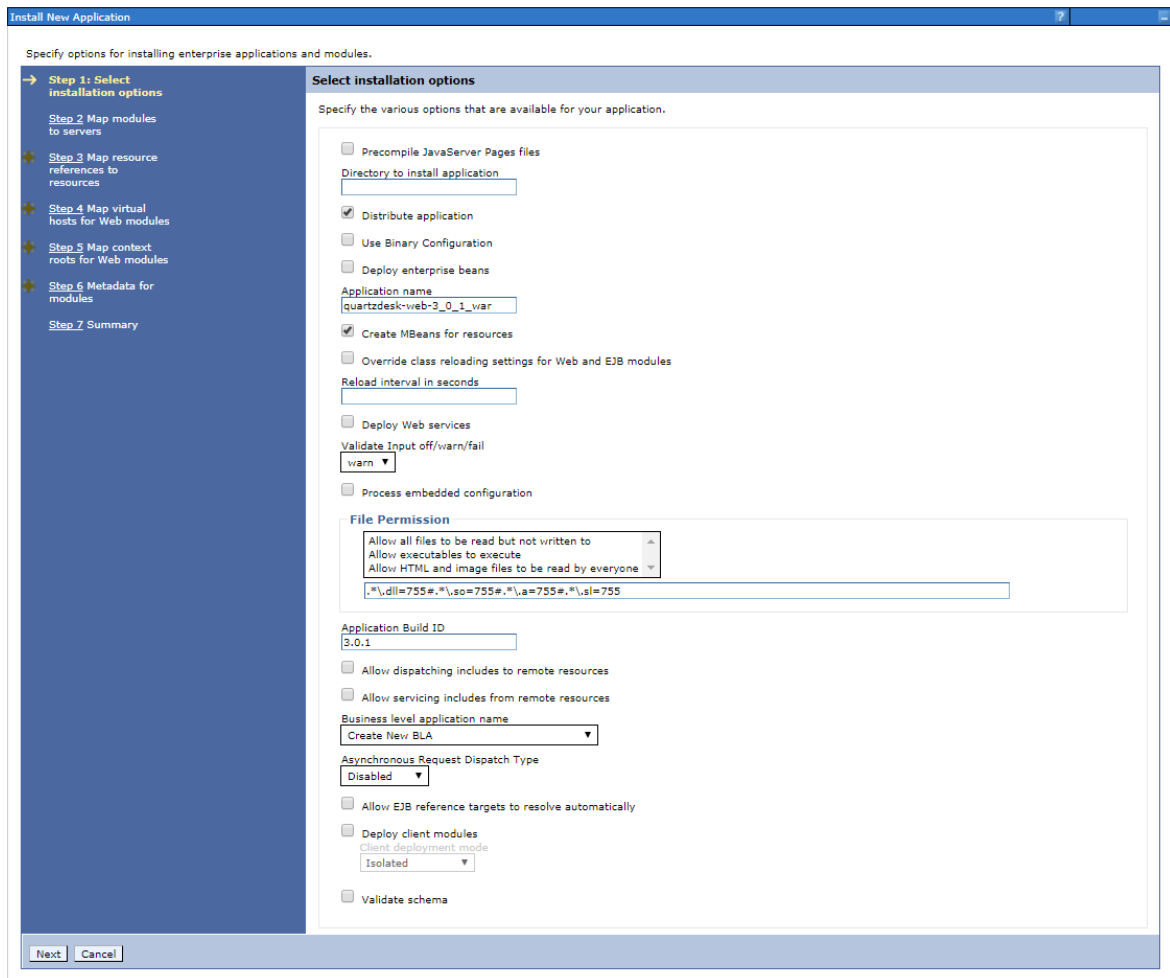
In WAC go to Applications → Application Types → WebSphere enterprise applications.

Click the Install button and select the `quartzdesk-web-x.y.z.war` file.
Click Next.

Leave the “Fast Path” radio button selected. Click Next.

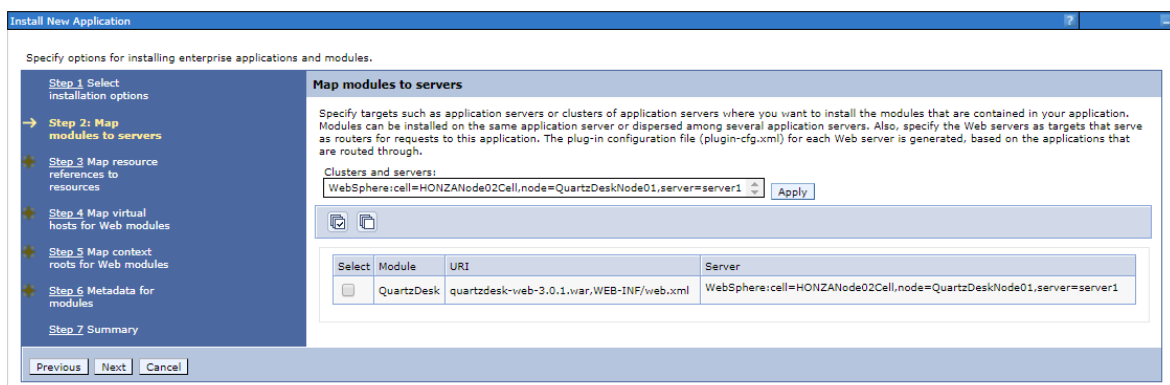
Step 1 – Select installation options

Click Next.



Step 2 – Map modules to servers

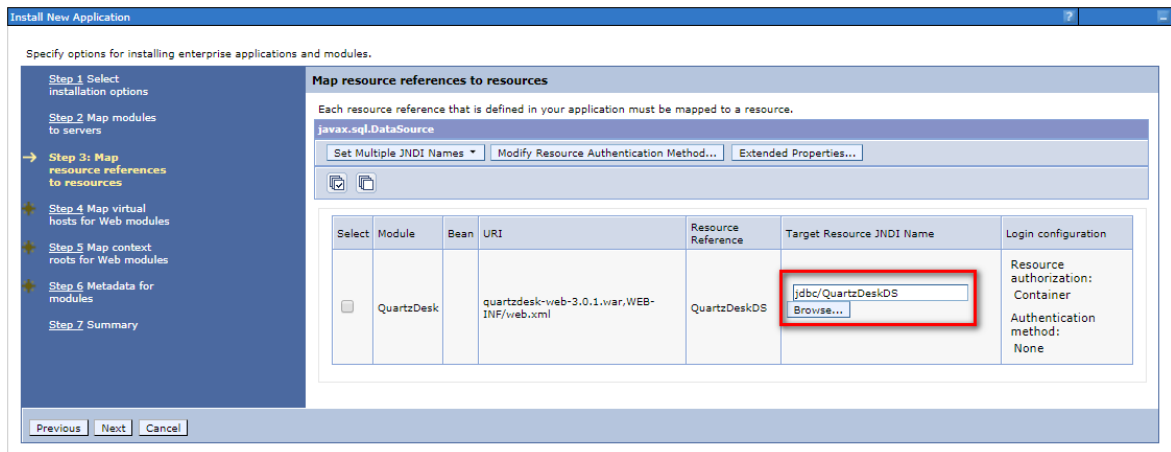
Map the QuartzDesk web module to the desired application server(s).



Click Next.

Step 3 – Map resource references to resources

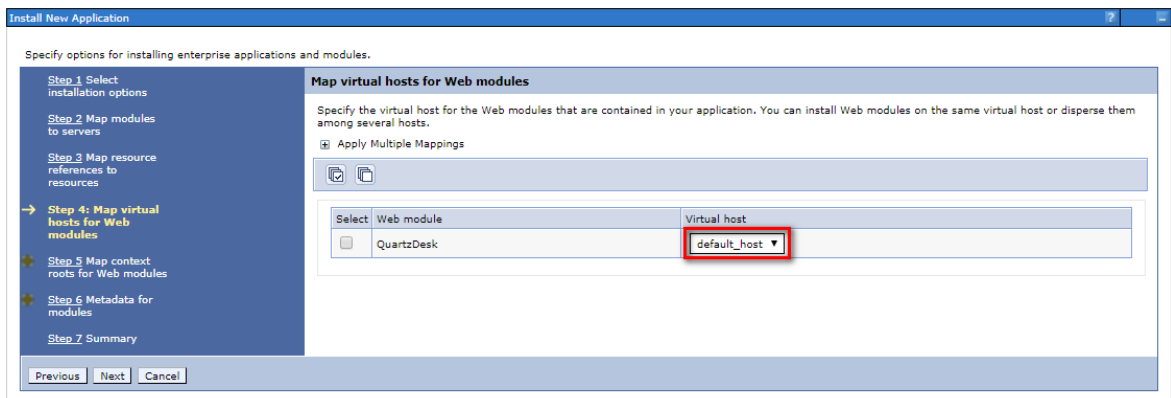
Map the QuartzDeskDS data source reference to the JNDI name of the QuartzDesk data source created in 4.5.



Click Next.

Step 4 – Map virtual hosts for Web modules

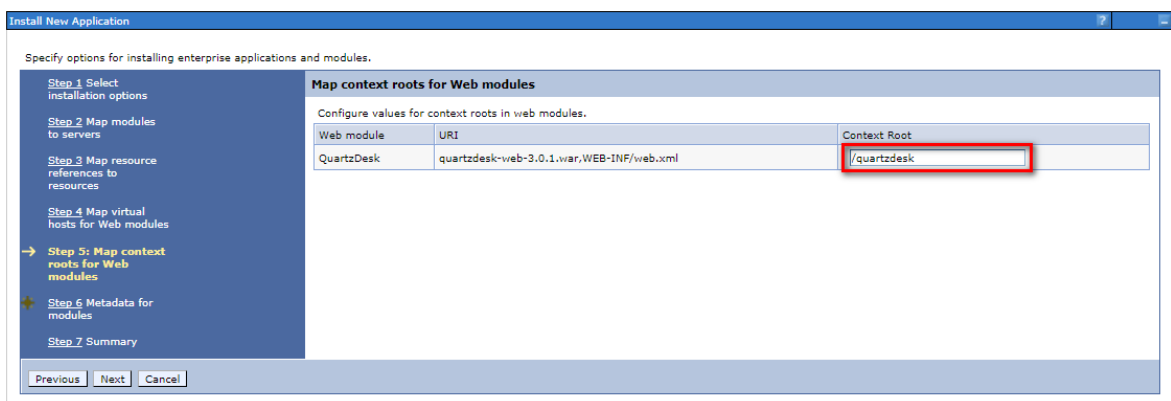
Map the QuartzDesk web module to the desired WebSphere virtual host.



Click Next.

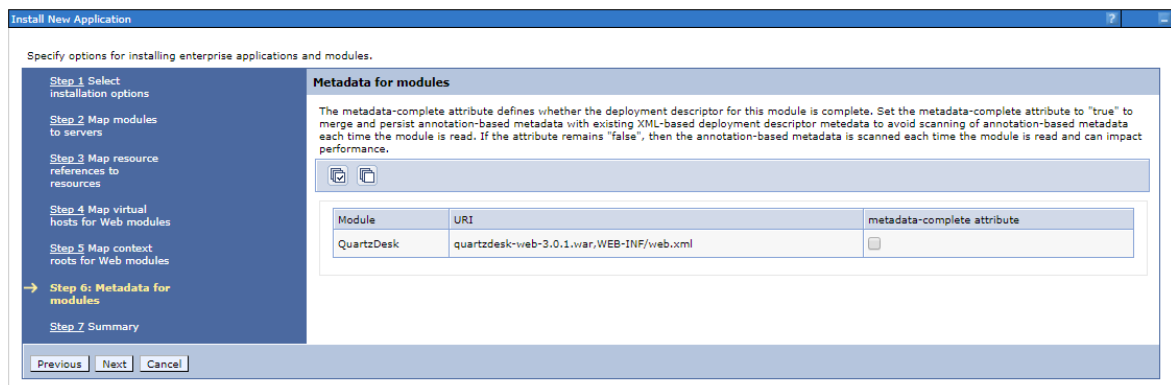
Step 5 – Map context roots for Web modules

Provide the web servlet context root for the QuartzDesk web application. We recommend using “/quartzdesk” (without quotes).



Click Next.

Step 6 – Metadata for modules (only WAS >= 8.0)



Specify options for installing enterprise applications and modules.

Step 6: Metadata for modules

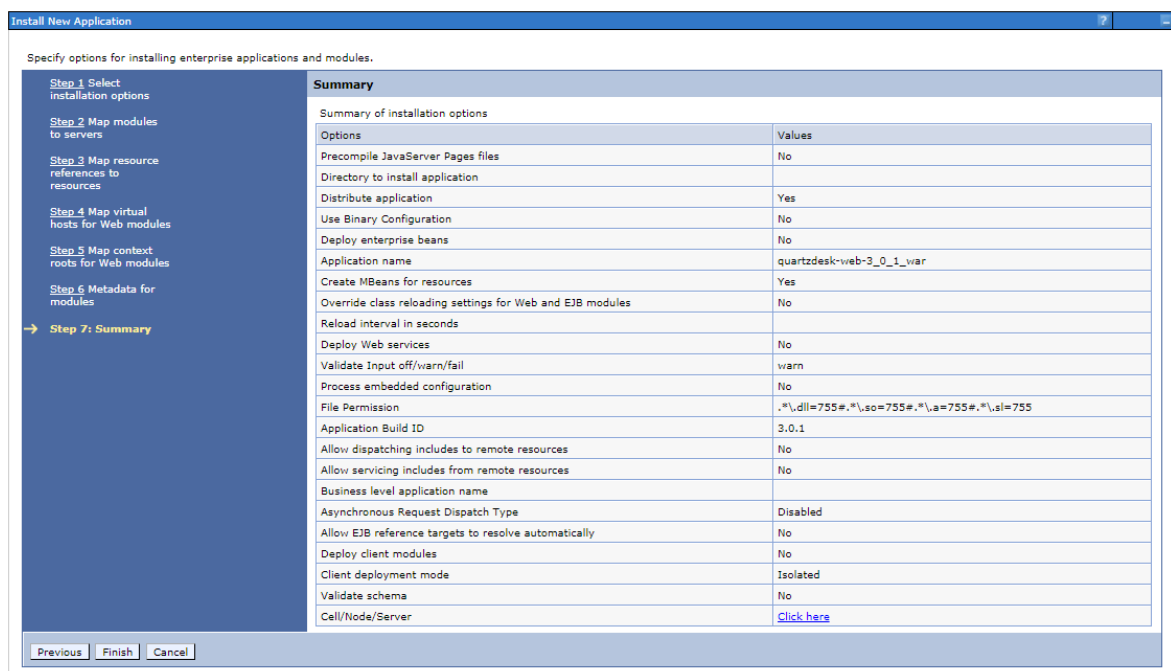
The metadata-complete attribute defines whether the deployment descriptor for this module is complete. Set the metadata-complete attribute to "true" to merge and persist annotation-based metadata with existing XML-based deployment descriptor metadata to avoid scanning of annotation-based metadata each time the module is read. If the attribute remains "false", then the annotation-based metadata is scanned each time the module is read and can impact performance.

Module	URI	metadata-complete attribute
QuartzDesk	quartzdesk-web-3.0.1.war,WEB-INF/web.xml	<input type="checkbox"/>

Previous Next Cancel

Click Next.

Step 7 – Summary



Specify options for installing enterprise applications and modules.

Step 7: Summary

Summary of installation options

Options	Values
Precompile JavaServer Pages files	No
Directory to install application	
Distribute application	Yes
Use Binary Configuration	No
Deploy enterprise beans	No
Application name	quartzdesk-web-3_0_1_war
Create MBeans for resources	Yes
Override class reloading settings for Web and EJB modules	No
Reload interval in seconds	
Deploy Web services	No
Validate Input off/warn/fail	warn
Process embedded configuration	No
File Permission	.*\,dl=755#.*\,so=755#.*\,a=755#.*\,sl=755
Application Build ID	3.0.1
Allow dispatching includes to remote resources	No
Allow servicing includes from remote resources	No
Business level application name	
Asynchronous Request Dispatch Type	Disabled
Allow EJB reference targets to resolve automatically	No
Deploy client modules	No
Client deployment mode	Isolated
Validate schema	No
Cell/Node/Server	Click here

Previous Finish Cancel

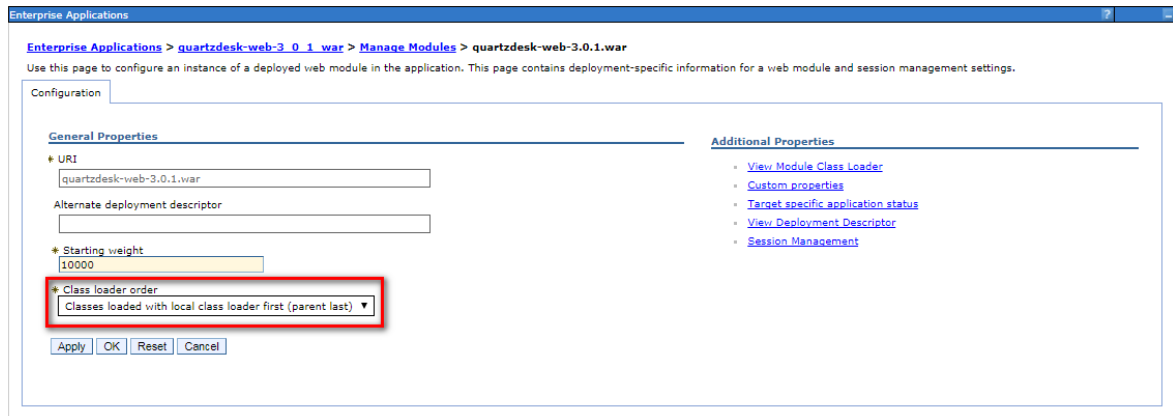
Click Finish.

When the installation completes, Save changes.

In WAC go to Applications → Application Types → WebSphere enterprise applications → quartzdesk-web-x_y_z_war.

Click on Manage Modules → QuartzDesk.

In “Class loader order” select the “Classes loaded with local class loader first (parent last)” option.



4.9 Start Application

In WAC go to Applications → Application Types → WebSphere enterprise applications. Select the checkbox next to the QuartzDesk web application in the Enterprise Applications list. Click the Start button and wait for the startup procedure to complete.

Check WAS logs under `WAS_SERVER_PROFILE/logs` for errors.

Check the QuartzDesk web application logs (by default located in the `WORK_DIR/logs` directory) for errors.

If there are no errors, point your browser to http://WAS_HTTP_HOST:WAS_HTTP_PORT/quartzdesk/ and verify that the QuartzDesk web application GUI is accessible.

Check the version number of the deployed QuartzDesk web application.



To log in, use the default administrator login credentials:

Username: admin

Password: admin123

Once logged in, you can go to Settings > Users to manage users with access to the QuartzDesk web application GUI. Users can be assigned different access permissions based on their intended roles.

In Settings > Groups, you can manage groups and assign access permissions to these groups. A group can contain users (members) who inherit access permissions of the group. A user can be a member of any number of groups.

Effective access permissions of a user are permissions associated directly with the user plus access permissions of all groups the user is a member of.

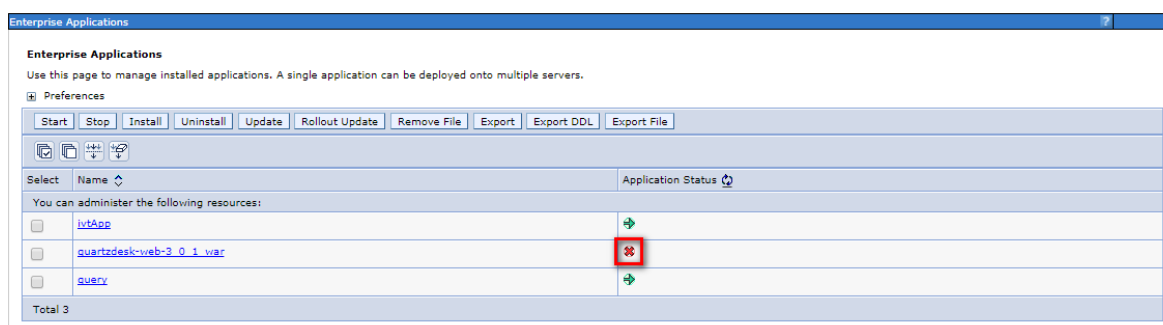


5. Upgrading

5.1 Stop Existing Application

In WAC go to Applications → Application Types → WebSphere enterprise applications. Select the checkbox next to the QuartzDesk web application in the Enterprise Applications list. Click the Stop button at the top of the list. Wait for the action to complete.

Upon successful stopping, the Application Status flag, shown next the existing QuartzDesk web application in the Deployments list, indicates that the applications has been stopped.



5.2 Backup

Backup your QuartzDesk web application database. We recommend performing a **full database backup**.

Backup the contents of the QuartzDesk web application work directory.

Backup the QuartzDesk web application in WAC by going to Applications → Application Types → WebSphere enterprise applications. Select the QuartzDesk web application by selecting the checkbox on the line. Click the Export button and wait for the export to complete. Download the exported WAR file.

Store the backup files in a safe place so that you can restore the original QuartzDesk web application version if the need arises.

5.3 Remove Existing Application

In WAC go to Applications → Application Types → WebSphere enterprise applications. Select the checkbox next to the QuartzDesk web application in the Enterprise Applications list. Click the Uninstall button at the top of the list. Wait for the action to complete and Save the changes when prompted.

Upon successful removal, the QuartzDesk web application disappears from the Enterprise Applications list.

5.4 Deploy New Application

Deploy the new version of the QuartzDesk web application by following the deployment steps outlined in 4.8.

5.5 Start New Application

Start the new version of the QuartzDesk web application by following the steps outlined in 4.9.



6. QuartzDesk 2.x to 3.x Migration Notes

To upgrade QuartzDesk web application 2.x to 3.x, follow the upgrade steps outlined in [5Error! Reference source not found.](#)

Before deploying the new QuartzDesk web application WAR file (quartzdesk-web-x.y.z.war), as outlined in step 5.4, make sure you have implemented changes described in this chapter.

6.1 Minimum Required Java Version

QuartzDesk web application 3.x requires Java 7 or higher. Java 6 is no longer supported.

Make sure WAS is configured to use Java 7 or higher.

6.2 Rename Configuration File

The name of the QuartzDesk web application 3.x configuration file has changed from `quartzdesk.properties` to `quartzdesk-web.properties`.

Rename the existing configuration file `quartzdesk.properties` located in the QuartzDesk web application work directory.

6.3 Rename Log Files

The names of QuartzDesk web application 3.x log files have changed.

Original Log File Name (2.x)	New Log File Name (3.x)
<code>quartzdesk.log</code>	<code>quartzdesk-web.log</code>
<code>quartzdesk-trace.log</code>	<code>quartzdesk-web-trace.log</code>

To use these new log file names, edit the QuartzDesk web application logging configuration file (`WORK_DIR/logback.xml`) and change the following lines:

```

Lister - [d:\var\quartzdesk-web.work\logback.xml]
File Edit Options Encoding Help
<?xml version="1.0" encoding="UTF-8"?>
<!--
~ Copyright (c) 2011-2014 QuartzDesk.com. All Rights Reserved.
~ QuartzDesk.com PROPRIETARY/CONFIDENTIAL. Use is subject to license terms.
-->

<configuration scan="true" scanPeriod="60 seconds" debug="false">

  <!--
  Registers the MBean for logback management in the JMX server under the given context name.
  -->
  <contextName>quartzdesk-web</contextName>
  <JMXConfigurator/>

  <!--
  Logback context property logback.config.dir is set by the LogbackInitContextListener
  to point to the parent directory of the logback configuration file (logback.xml).
  -->
  <property name="logs.dir" value="${logback.config.dir:-./logs}/>

  <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <file>${logs.dir}/quartzdesk-web.log</file>
    <append>true</append>

    <filter class="ch.qos.logback.classic.filter.ThresholdFilter">
      <level>INFO</level>
    </filter>

    <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
      <!-- daily rollover -->
      <fileNamePattern>${logs.dir}/quartzdesk-web.log.%d{yyyy-MM-dd}</fileNamePattern>
      <!-- keep 10 days' worth of history -->
      <maxHistory>10</maxHistory>
    </rollingPolicy>

    <encoder>
      <charset>UTF-8</charset>
      <pattern>[%date] %-1level [%thread] [%mdc] [%logger:%line] - %msg%n</pattern>
    </encoder>
  </appender>

  <appender name="TRACE_FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <file>${logs.dir}/quartzdesk-web-trace.log</file>
    <append>true</append>

    <filter class="ch.qos.logback.classic.filter.ThresholdFilter">
      <level>TRACE</level>
    </filter>

    <rollingPolicy class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
      <fileNamePattern>${logs.dir}/quartzdesk-web-trace.log.%i</fileNamePattern>
      <minIndex>1</minIndex>
      <maxIndex>5</maxIndex>
    </rollingPolicy>

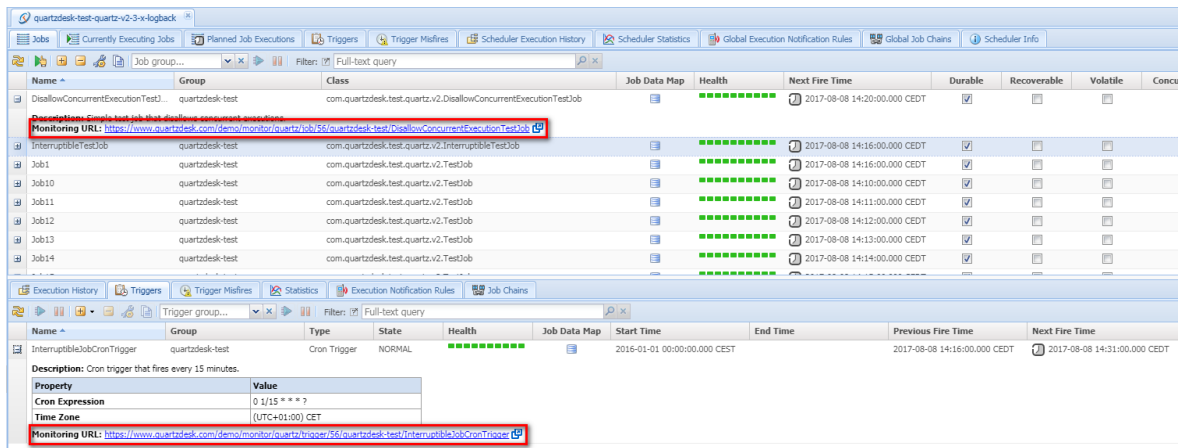
    <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
      <maxFileSize>2MB</maxFileSize>
    </triggeringPolicy>

    <encoder>
      <charset>UTF-8</charset>
      <pattern>[%date] %-1level [%thread] [%mdc] [%logger:%line] - %msg%n</pattern>
    </encoder>
  </appender>
  
```

Alternatively, extract the default `logback.xml` configuration file from the QuartzDesk web application 3.x WAR (`quartzdesk-web-x.y.z.war/extras/work/logback.xml`) and copy it to `WORK_DIR`.

6.4 Access to Monitoring URLs (REST API)

In QuartzDesk web application 2.x, the monitoring REST API URLs could be accessed by users with the QuartzDeskMonitor J2EE security role. In QuartzDesk web application 3.x, these monitoring URLs can be accessed by all authenticated users.



We recommend that you create a dedicated user account to access these monitoring URLs. The user account can be created in Settings → Users in the QuartzDesk GUI.



All monitoring URLs in QuartzDesk 3.x support the HTTP Basic authentication scheme where the user's authentication credentials are passed in the `Authorization` HTTP header. Please note that the same authentication scheme was used by monitoring URLs in QuartzDesk 2.x.

6.5 Access to JAX-WS Endpoints

In QuartzDesk web application 2.x, all JAX-WS web service endpoints could be accessed by users with the QuartzDeskService J2EE security role. In QuartzDesk web application 3.x, these web service end points can only be accessed by authenticated users with particular access permissions.

The following table lists all JAX-WS web services and the security permissions that are required to access these web services.

JAX-WS Service	Required Permission
Connection Service	WS_CONNECTION
Security Service	WS_SECURITY
Quartz Service	WS_QUARTZ
Quartz Execution History Service	WS_QUARTZ_EXEC_HISTORY
Quartz Execution Notification Rule Service	WS_QUARTZ_EXEC_NOTIF_RULE
Quartz Job Chain Service	WS_QUARTZ_JOB_CHAIN

We recommend that you create a dedicated user account to access these JAX-WS endpoints. The user account can be created in Settings → Users in the QuartzDesk GUI. Do not forget to assign the user the relevant permission(s).



All JAX-WS web service endpoints in QuartzDesk 3.x support the HTTP Basic authentication scheme where the user's authentication credentials are passed in the `Authorization` HTTP header. Please note that the same authentication scheme was used by JAX-WS endpoints in QuartzDesk 2.x.

7. Cluster Deployment Notes

When deploying the QuartzDesk web application to a WebSphere cluster you need to follow the configuration steps described in preceding chapters. In addition to these, there are several extra configuration steps that must be performed for a cluster deployment.

7.1 HTTP Session Replication and Affinity

The QuartzDesk web application makes use of HTTP sessions and to store some short-lived and user-specific data. To achieve high-availability (HA), it is necessary to make the session data available on all application cluster members so that when one cluster member becomes unavailable, the remaining cluster members can take over and handle user requests without the user noticing any service interruption. To make the session data available on all application cluster members, the HTTP session replication process must be enabled on the cluster.



The amount of data stored by the QuartzDesk web application in an HTTP session is kept at the absolute minimum to reduce the session replication overhead. The total size of data stored in the session does not exceed 1KB.

When configuring session replication, we recommend that you also enable session affinity (sticky-sessions) on the load-balancer so that all user requests are preferably passed to the WebSphere instance that handled the first user request that established the session.

Please refer to the WebSphere and load-balancer documentation for details on how to configure session replication and session affinity because the actual steps may vary depending on the WebSphere cluster topology and configuration.

7.2 Shared Work Directory

We recommend that you put the QuartzDesk web application work directory, described in chapter 4.4, on a shared drive and make this work directory available to all cluster members. Not only does this make application and configuration upgrading easier, it is actually required by all “Save” (for example, Save Log, Save Chart etc.) actions provided by the QuartzDesk web application GUI. These actions trigger two subsequent HTTP requests where the first request prepares the data and stores it in the `WORK_DIR/tmp` directory and the second request downloads the data and makes the browser open the Save As dialog.

During a fail-over or if the session affinity is not enabled, it can easily happen that the first request is handled by cluster member A and the second request is handled by cluster member B. If A and B are not configured to use the same `WORK_DIR/tmp` directory, then B will fail to serve the data prepared by A during the preceding request because the data will not be found.

7.3 Logging Configuration

If you set up your cluster to use a shared QuartzDesk web application work directory, as described in the previous chapter, you will need to edit the QuartzDesk web application logging configuration file `WORK_DIR/logback.xml` and decide where QuartzDesk web application instances running on individual cluster members should log. There are two options:

- 1) Logging into the same (shared) log files.
- 2) Logging into separate log files.

The QuartzDesk web application uses two log files – quartzdesk-web.log and quartzdesk-web-trace.log that are stored in WORK_DIR/logs directory. The following chapters discuss these two options.

7.3.1 Using Shared Log Files

In order to make individual QuartzDesk web application instances log into the same log files, you must enable the prudent mode on both file appenders used in the WORK_DIR/logback.xml configuration file:

```
...

<appender name="FILE"
class="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-web.log</file>
  <append>true</append>
  <prudent>true</prudent>
  ...
</appender>

<appender name="TRACE_FILE"
class="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-web-trace.log</file>
  <append>true</append>
  <prudent>true</prudent>
  ...

<!--
  We must use the TimeBasedRollingPolicy because the
  FixedWindowRollingPolicy is not supported in prudent mode!
-->
<rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
  <!-- daily rollover -->
  <fileNamePattern>${logs.dir}/quartzdesk-web.log.%d{yyyy-MM-
dd}</fileNamePattern>
  <!-- keep 10 days' worth of history -->
  <maxHistory>10</maxHistory>
</rollingPolicy>

<!--
  The SizeBasedTriggeringPolicy removed because it is used only in
  conjunction with the FixedWindowRollingPolicy.
-->

<encoder>
  <charset>UTF-8</charset>
  <pattern>[%date] %.-1level [%thread] [%mdc] [%logger:%line] -
%msg%n</pattern>
</encoder>
</appender>

...
```

For details on the Logback prudent mode, please refer to <http://logback.qos.ch/manual/appenders.html#FileAppender>.



Because prudent mode relies on exclusive file locks to manage concurrent access to the log files and these locks can have negative impact on the QuartzDesk web application's performance, we generally discourage using the prudent mode and shared log files.

7.3.2 Using Separate Log Files

In order to make individual QuartzDesk web application instances log into separate log files, you can use a JVM system property set on all cluster member JVMs. The value of this property must be unique for all cluster members. The property can be referred to from the `WORK_DIR/logback.xml` logging configuration file.

The following examples assume the use of the `cluster.member.instanceId` JVM system property, but any JVM system property name can be used.

There are two common approaches as to where the separate log files produced by individual QuartzDesk web application instances are stored:

- 1) Log files created under a common log root directory.

```
...
<appender name="FILE"
class="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-web--${cluster.member.instanceId}.log</file>
  <append>true</append>
...
  <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <!-- daily rollover -->
    <fileNamePattern>${logs.dir}/quartzdesk-web-
-${cluster.member.instanceId}.log.%d{yyyy-MM-dd}</fileNamePattern>
    <!-- keep 10 days' worth of history -->
    <maxHistory>10</maxHistory>
  </rollingPolicy>
...
</appender>

<appender name="TRACE_FILE"
class="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>${logs.dir}/quartzdesk-web--${cluster.member.instanceId}-
trace.log</file>
  <append>true</append>
...
  <rollingPolicy
class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
  <fileNamePattern>${logs.dir}/quartzdesk-web-
-${cluster.member.instanceId}-trace.log.%i</fileNamePattern>
  <minIndex>1</minIndex>
  <maxIndex>5</maxIndex>
  </rollingPolicy>
...
</appender>
...
```

2) Log files created in separate (cluster member specific) log root directories.

```

...
<!--
  Logback context property logback.config.dir is set by the
  LogbackInitContextListener to point to the parent directory of the Logback
  configuration file (logback.xml).
-->
<property name="logs.dir" value="${logback.config.dir:-
  .}/${cluster.member.instanceId}/logs"/>
...
  
```

7.4 Internal Quartz Scheduler

The QuartzDesk web application ships with an embedded Quartz scheduler to periodically execute its internal jobs. When deploying the QuartzDesk web application to a cluster, it is necessary to **assign unique instance IDs to Quartz scheduler instances** running in the clustered QuartzDesk web application instances.

For these purposes the QuartzDesk web application configuration (`quartzdesk-web.properties` file) provides the `scheduler.org.quartz.scheduler.instanceIdGenerator.class` configuration property. The value of this property must be a fully-qualified class name of a Java class that implements the `org.quartz.spi.InstanceIdGenerator` Quartz API interface. Quartz API provides two out of the box implementations suitable for clustered QuartzDesk web application deployments:

Implementation	Description
<code>org.quartz.simpl.HostnameInstanceIdGenerator</code>	<p>This implementation is suitable for QuartzDesk web application deployments where individual clustered QuartzDesk web application instances run on distinct hosts and each of these hosts is assigned a unique hostname.</p> <p>This is the default implementation used by the QuartzDesk web application. No configuration changes are necessary to use this instance ID generator.</p>
<code>org.quartz.simpl.SystemPropertyInstanceIdGenerator</code>	<p>This implementation is suitable for QuartzDesk web application deployments where some of the clustered QuartzDesk web application instances run on the same host.</p> <p>This implementation extracts the Quartz scheduler instance ID from the <code>org.quartz.scheduler.instanceId</code> JVM system property that must be explicitly set.</p> <p>Please refer to the WebSphere documentation for details on how to add a new JVM system property.</p>

Please refer to the table above and optionally modify the value of the `scheduler.org.quartz.scheduler.instanceIdGenerator.class` configuration property according to the cluster configuration.

