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

QuartzDesk Version: 2.x

April 24, 2017

Table of Contents

1. PURPOSE 3

2. DEFINITIONS 4

3. REQUIREMENTS 5

3.1 SOFTWARE REQUIREMENTS 5
3.1.1 Browser 5
3.1.2 Operating System 5
3.1.3 Java 5
3.1.4 Application Server 5
3.1.5 Database 5
3.1.6 Database JDBC Driver 5
3.1.7 QuartzDesk Web Application Archive 6
3.2 HARDWARE REQUIREMENTS 6

4. INSTALLATION 7

4.1 DATABASE 7
4.2 JDBC DRIVER 7
4.3 JDBC PROVIDER 8
4.3.1 DB2 9
4.3.2 H2 10
4.3.3 Microsoft SQL Server 11
4.3.4 MySQL 12
4.3.5 Oracle 13
4.3.6 PostgreSQL 14
4.4 DATA SOURCE J2C AUTHENTICATION DATA 14
4.5 DATA SOURCE 15
4.5.1 DB2 16
4.5.2 H2 19
4.5.3 Microsoft SQL Server 22
4.5.4 MySQL 25
4.5.5 Oracle 28
4.5.6 PostgreSQL 31
4.6 APPLICATION WORK DIRECTORY 34
4.7 APPLICATION CONFIGURATION 35
4.8 DEPLOY APPLICATION 36
4.9 SECURITY 40
4.9.1 Enable application security 40
4.9.2 Map security roles to users / groups 40
4.10 START APPLICATION 41

5. UPGRADING 42

5.1 STOP EXISTING APPLICATION 42
5.2 BACKUP 42
5.3 REMOVE EXISTING APPLICATION 42
5.4 DEPLOY NEW APPLICATION 43
5.5 START NEW APPLICATION 43

6. CLUSTER DEPLOYMENT NOTES 44

6.1 HTTP SESSION REPLICATION AND AFFINITY 44
6.2 SHARED WORK DIRECTORY 44
6.3 LOGGING CONFIGURATION 44
6.3.1 Using Shared Log Files 45
6.3.2 Using Separate Log Files 46
6.4 INTERNAL QUARTZ SCHEDULER 47

Copyright 2013-2017 QuartzDesk.com

1. Purpose

This document describes the installation and upgrade process for the QuartzDesk web application 2.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.

Copyright 2013-2017 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 database server host.
DB_PORT	5432	QuartzDesk database server port.
DB_NAME	quartzdesk	QuartzDesk database name.
DB_SCHEMA	quartzdesk	QuartzDesk database schema.
DB_USER	quartzdesk	QuartzDesk database user.
DB_PASSWORD	quartzdesk	QuartzDesk 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	QuartzDesk work directory.

Copyright 2013-2017 QuartzDesk.com

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	10
FireFox	3.6
Internet Explorer	8
Opera	11
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) 6, 7, 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.

Copyright 2013-2017 QuartzDesk.com

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, you need to obtain the QuartzDesk web application archive (WAR). 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 runs on any physical or virtualized hardware that supports the above software requirements.

Copyright 2013-2017 QuartzDesk.com

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 database named `quartzdesk`[1] (DB_NAME) owned by the DB_USER.

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

Please refer to the database engine documentation for details on how to perform the above database operations as they are all database-specific.

Please note that you do not have to create any other database objects (tables, keys, indices etc.) in the QuartzDesk database. These objects will be automatically created by the QuartzDesk web application during the 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 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, then leave the variable as is. Otherwise, add, or adjust the variable value accordingly.

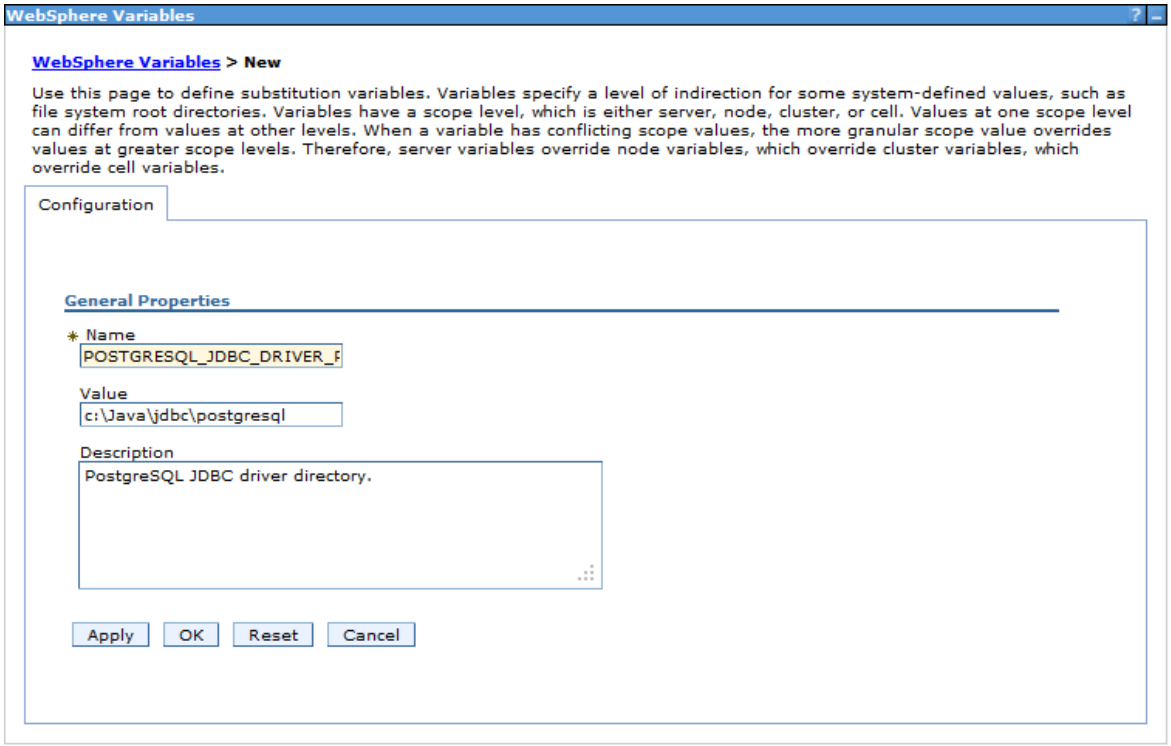
Database	WebSphere Variable Name	Default Value In WAC
DB2	DB2_JCC_DRIVER_PATH	Defined with empty value.
H2	H2_JDBC_DRIVER_PATH	Not defined.
Microsoft SQL Server	MICROSOFT_JDBC_DRIVER_PATH	Defined with empty value.
MySQL	MYSQL_JDBC_DRIVER_PATH	Not defined.

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

Copyright 2013-2017 QuartzDesk.com

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.

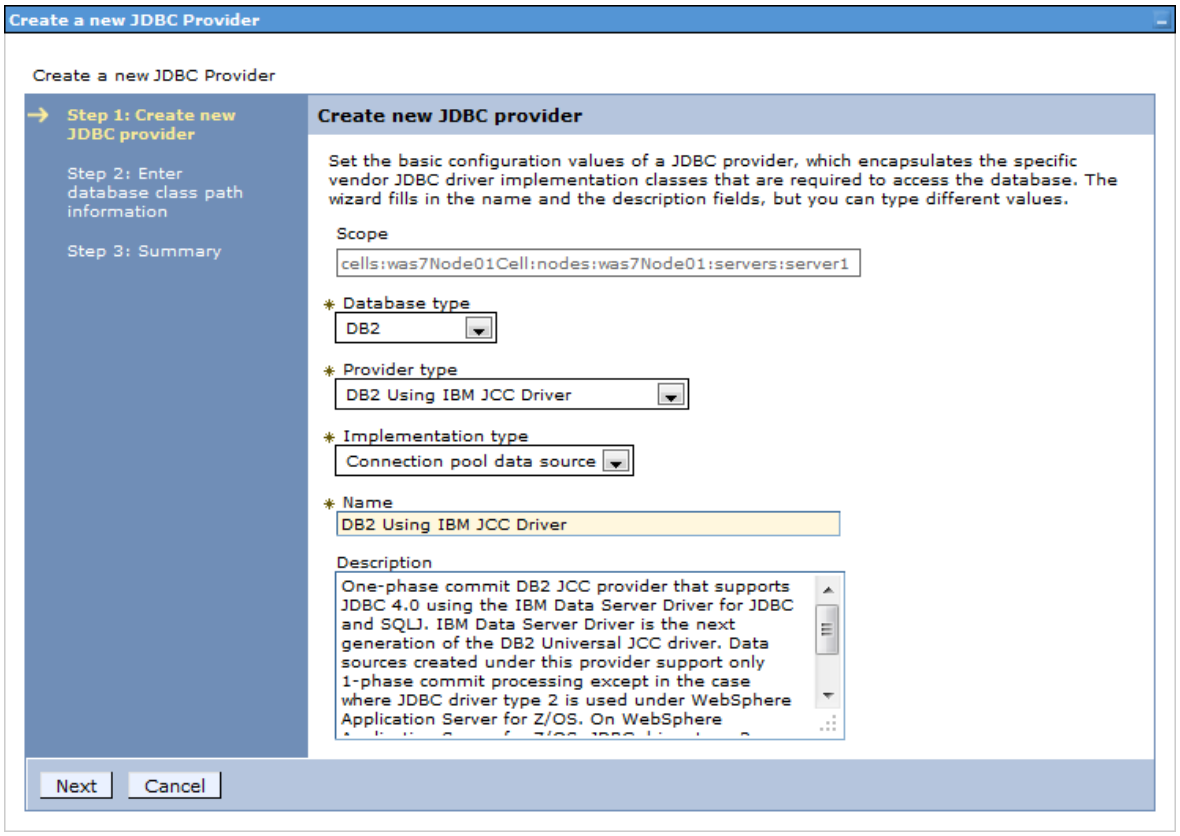


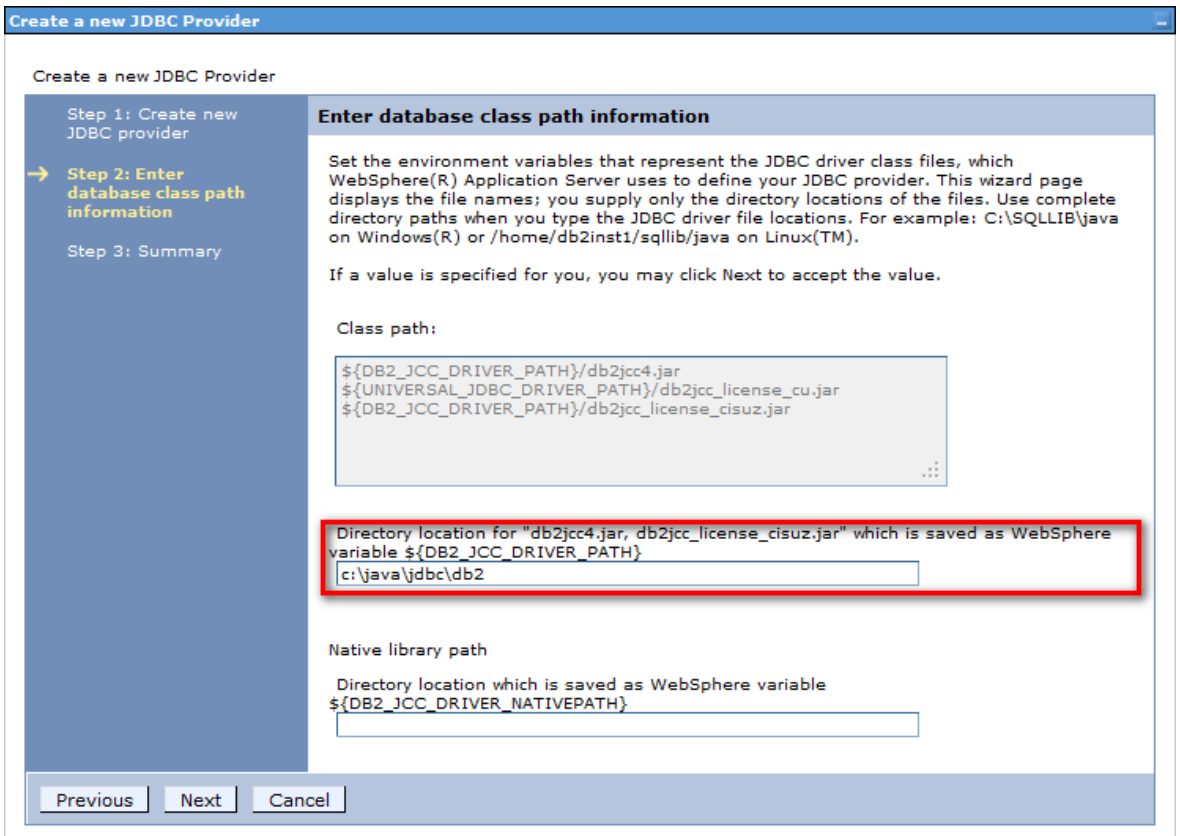
4.3 JDBC Provider

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

Copyright 2013-2017 QuartzDesk.com

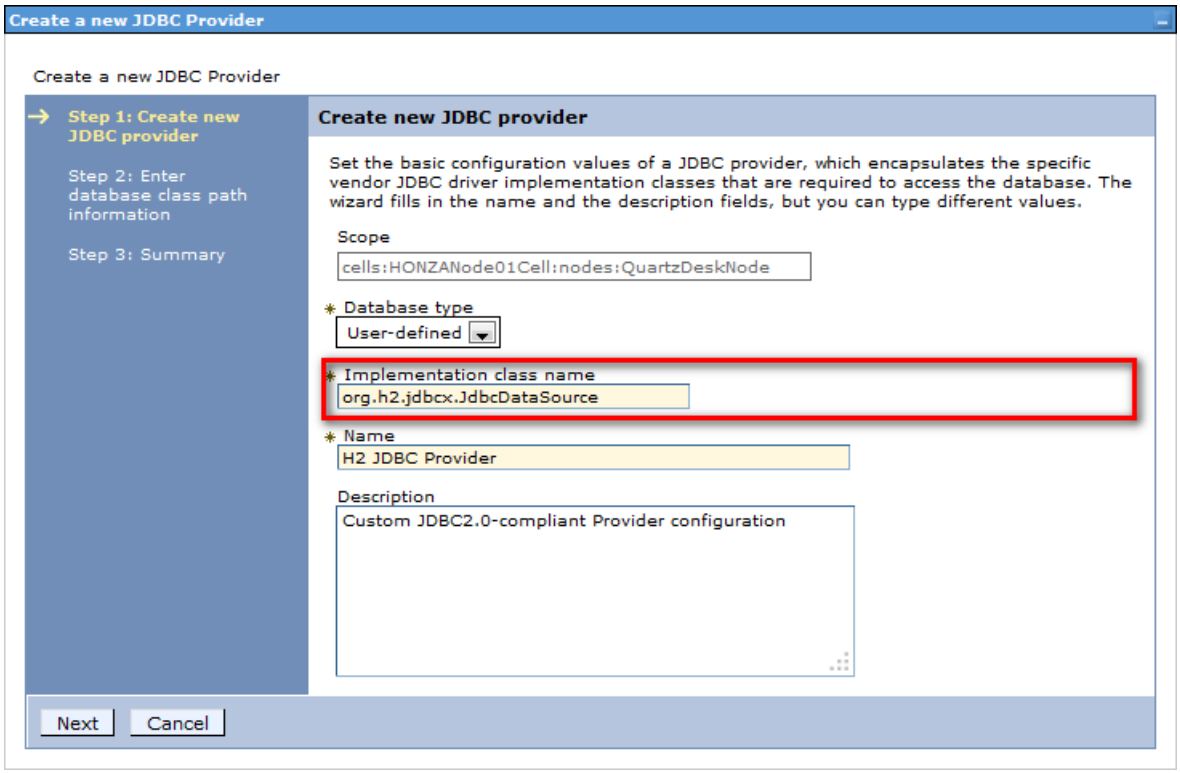
4.3.1 **DB2**



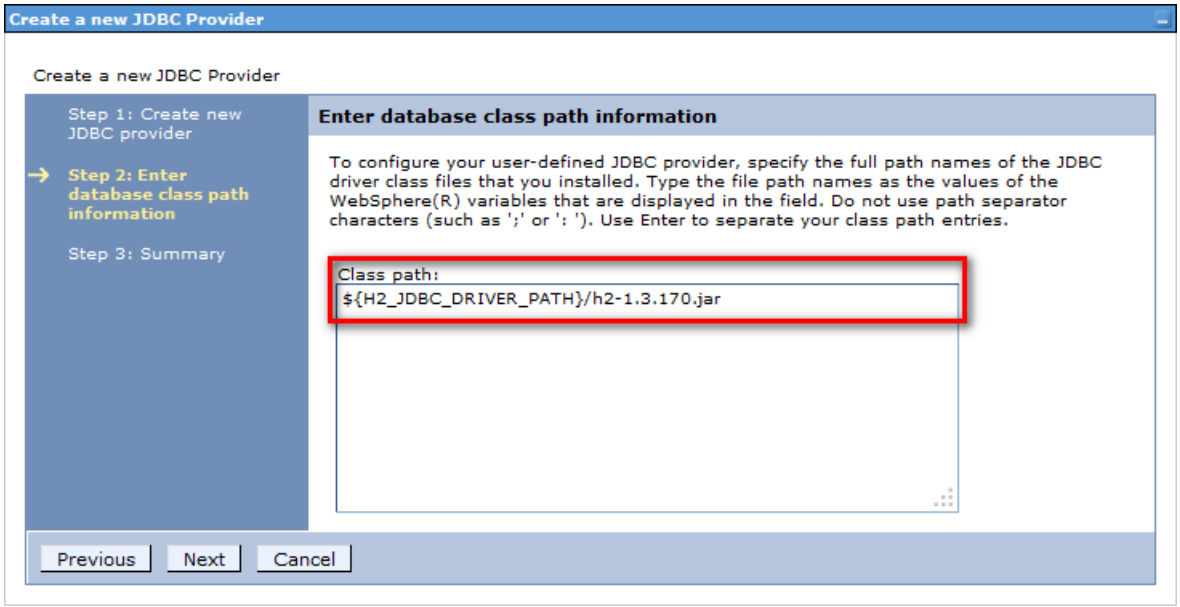


Copyright 2013-2017 QuartzDesk.com

4.3.2 **H2**



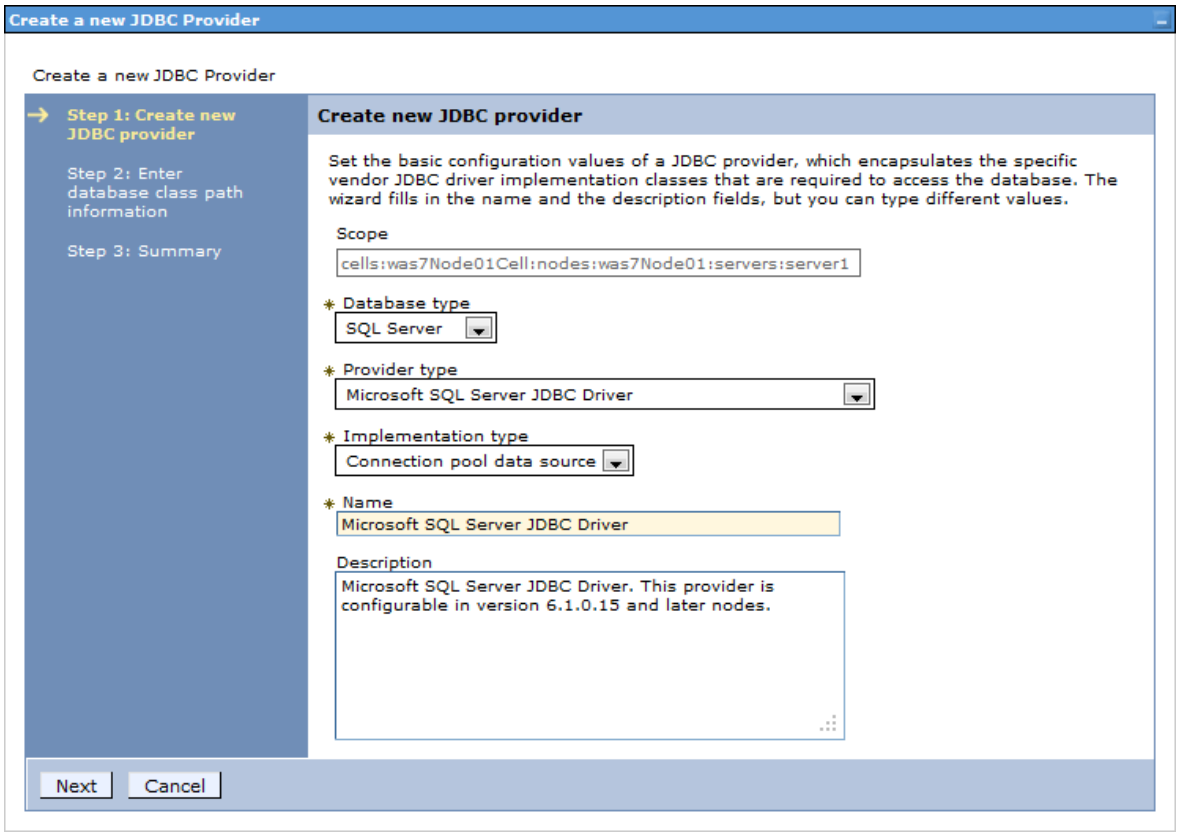
Database type: User-defined
Implementation class name: `org.h2.jdbcx.JdbcDataSource`
Name: H2 JDBC Provider

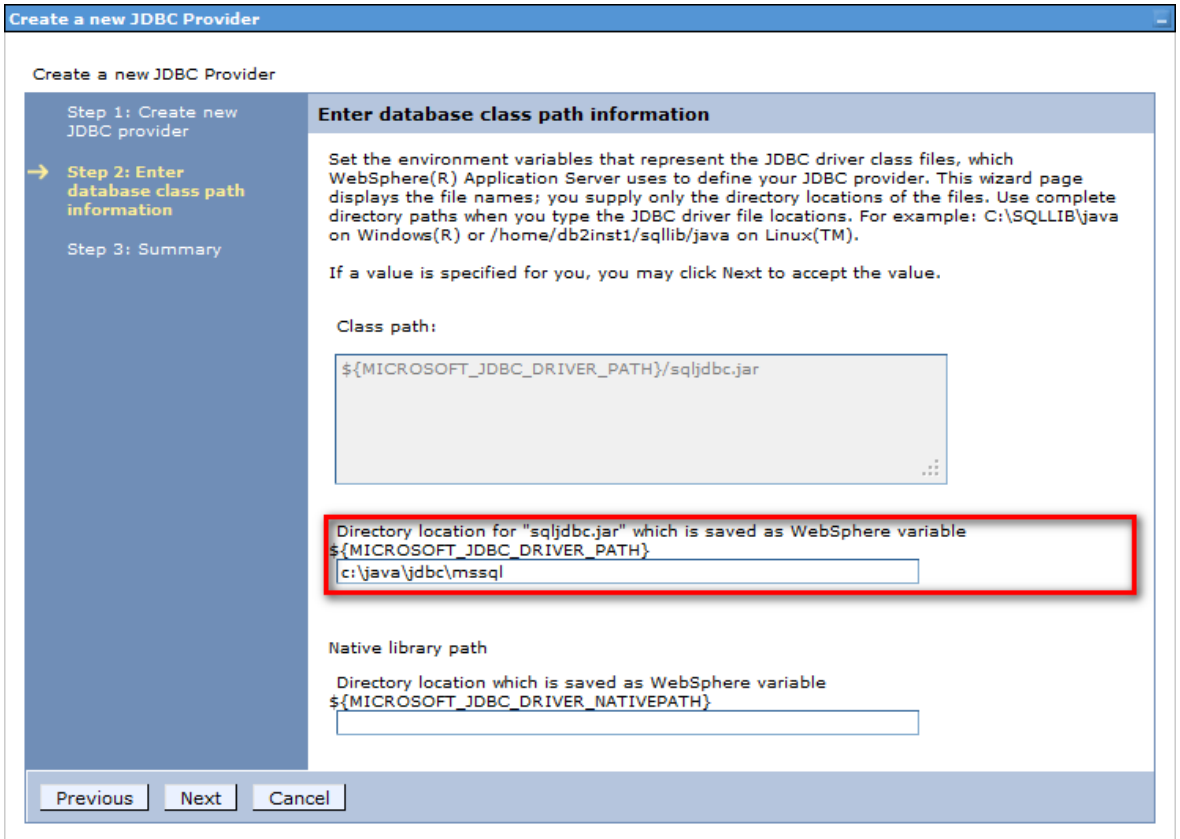


Class path: H2 JDBC driver classpath

Copyright 2013-2017 QuartzDesk.com

4.3.3 Microsoft SQL Server

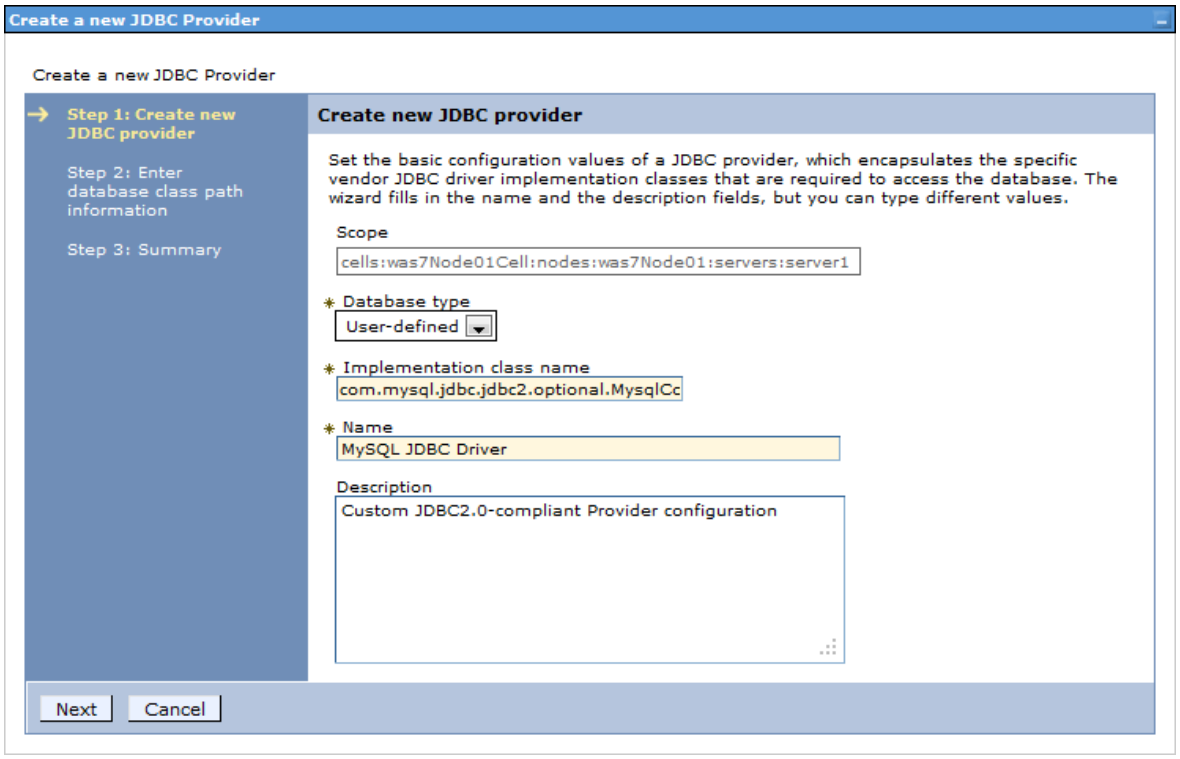




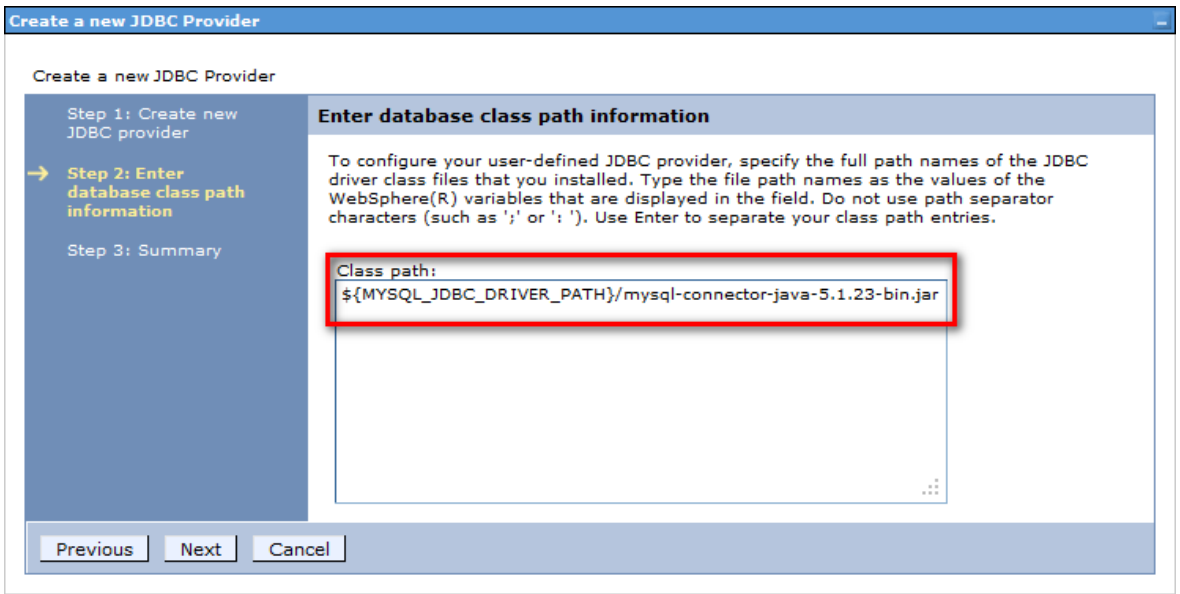
Copyright 2013-2017 QuartzDesk.com

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.

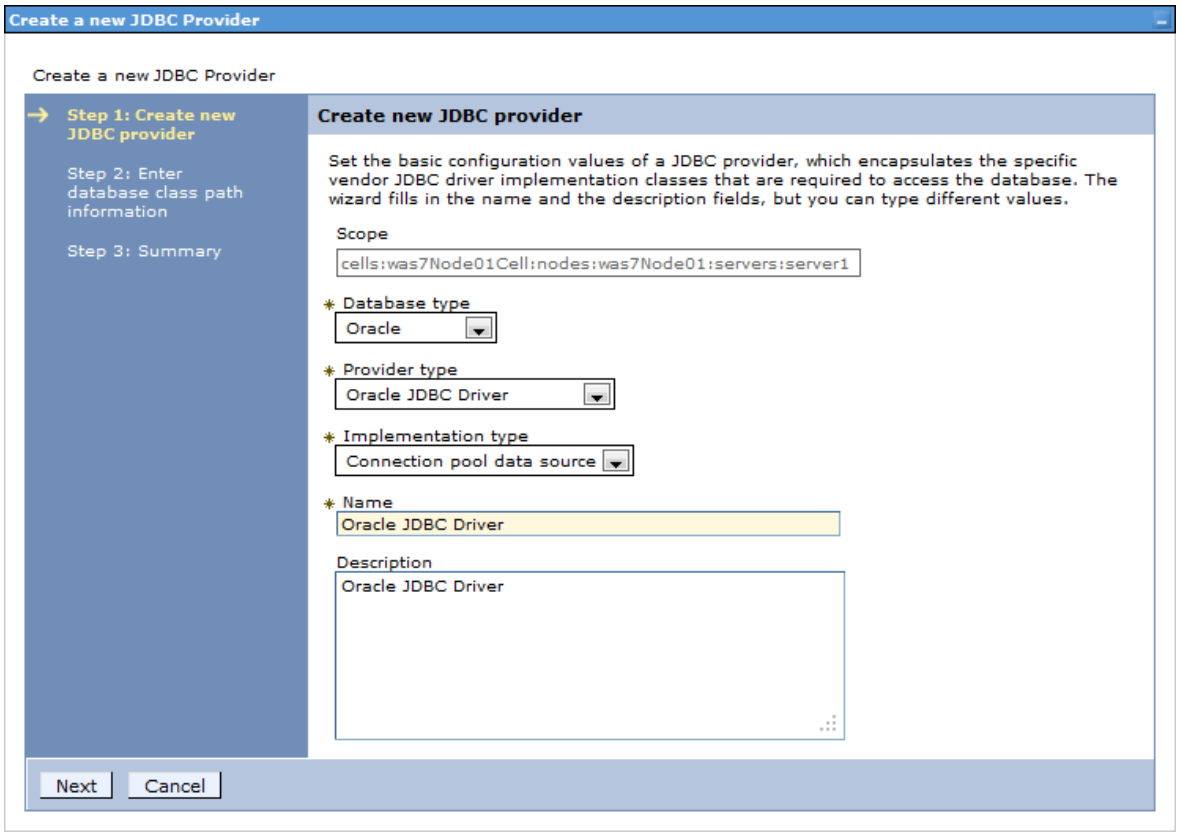
4.3.4 MySQL

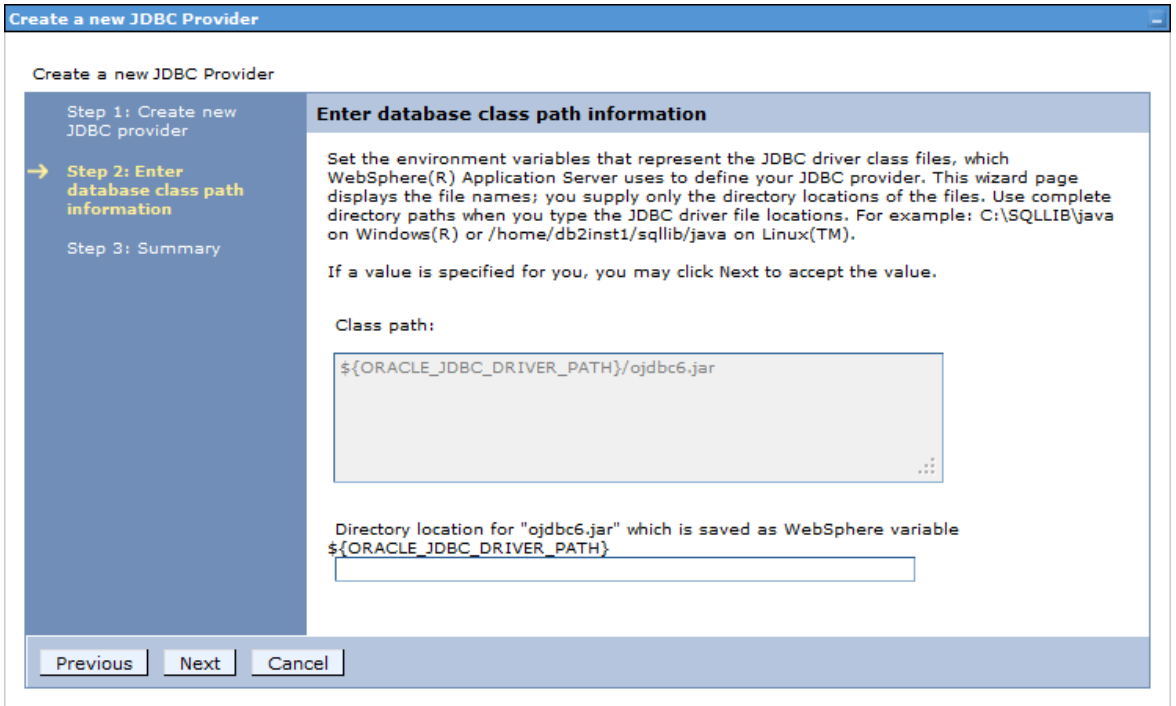


Implementation class name:
`com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoolDataSource`

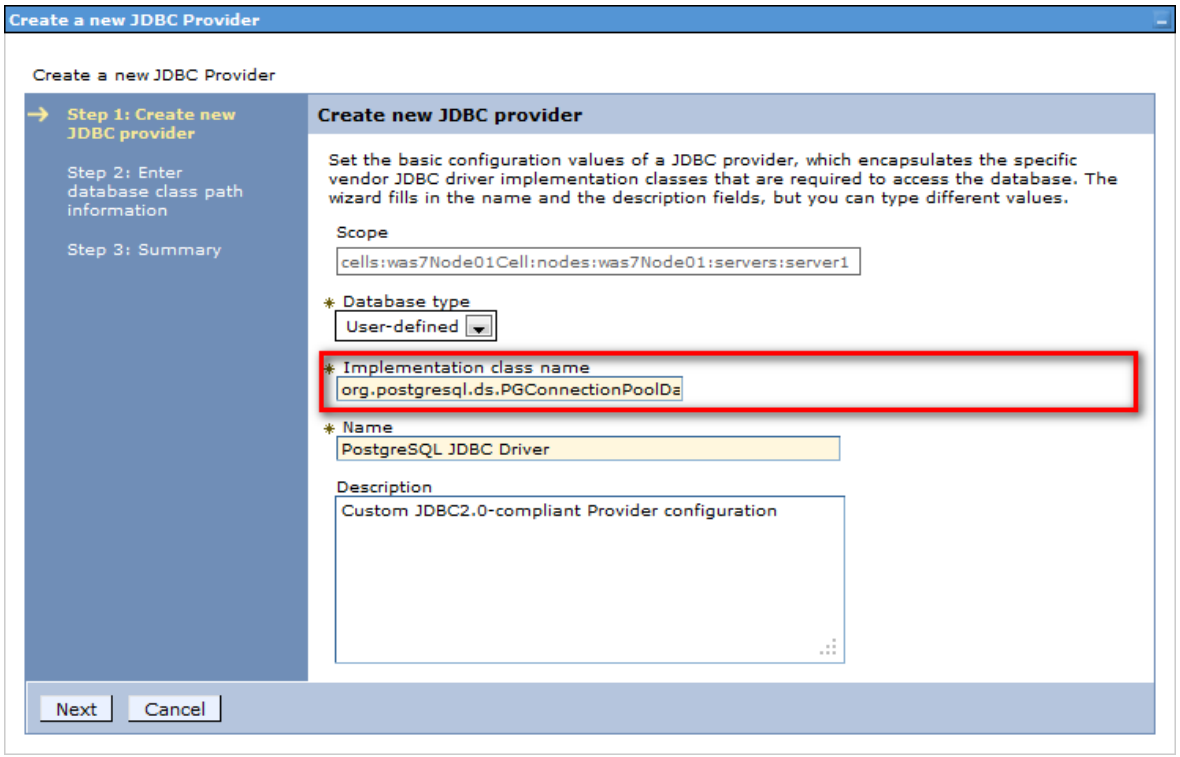


4.3.5 Oracle

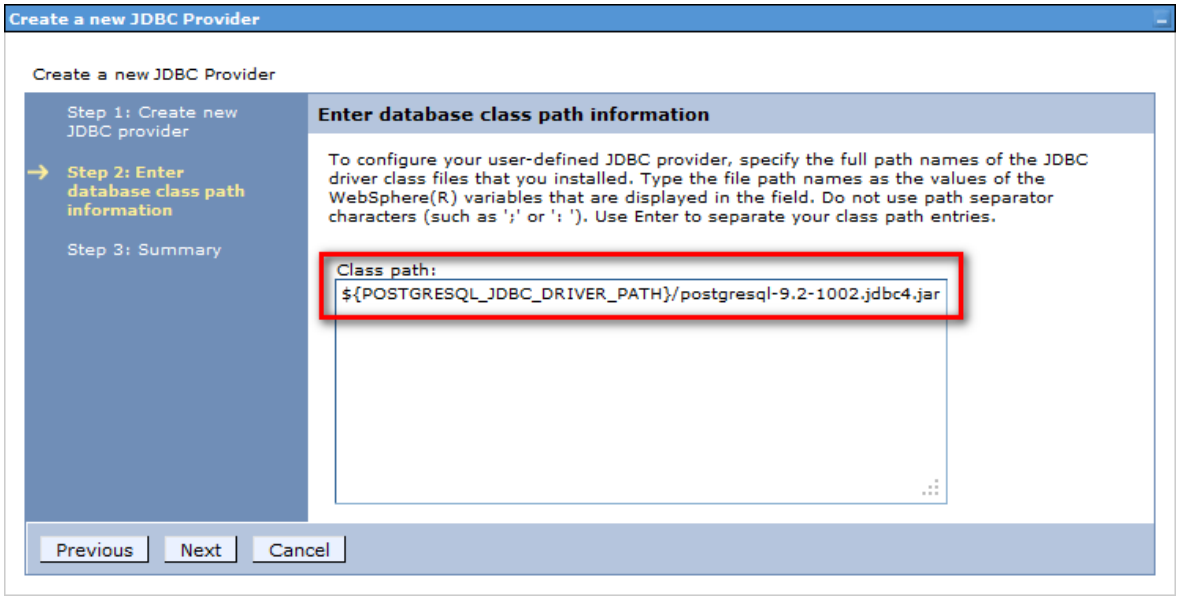

Create a new JDBC Provider
Create a new JDBC Provider
Step 1: Create new JDBC provider
Step 2: Enter database class path information
Step 3: Summary
Create new JDBC provider
Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.
Scope
cells:was7Node01Cell:nodes:was7Node01:servers:server1
Database type
Oracle
Provider type
Oracle JDBC Driver
Implementation type
Connection pool data source
Name
Oracle JDBC Driver
Description
Oracle JDBC Driver
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:
${ORACLE_JDBC_DRIVER_PATH}/ojdbc6.jar
Directory location for "ojdbc6.jar" which is saved as WebSphere variable ${ORACLE_JDBC_DRIVER_PATH}
Previous
Next
Cancel

4.3.6 PostgreSQL

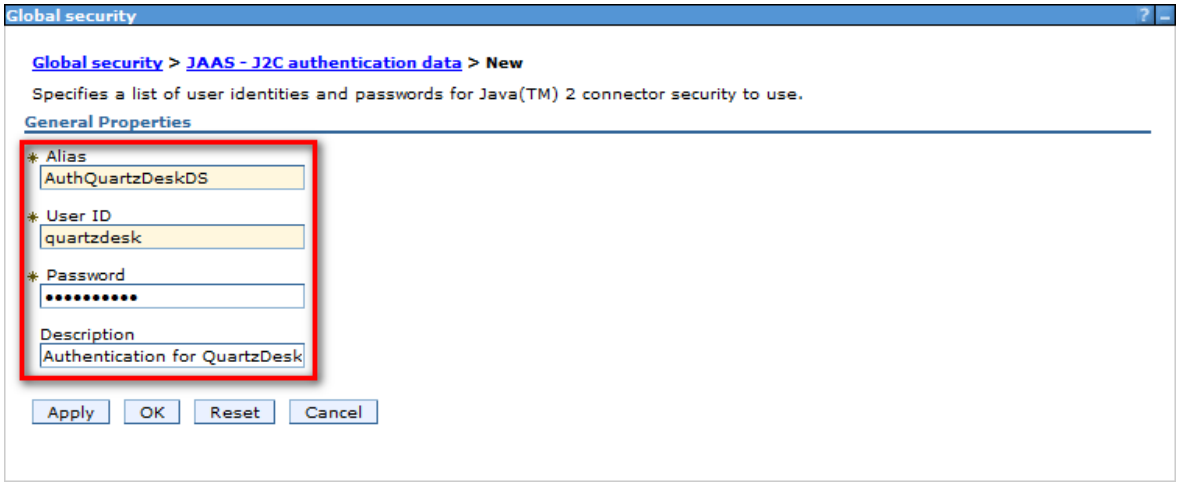


Implementation class name: `org.postgresql.ds.PGConnectionPoolDataSource`



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 database data source.



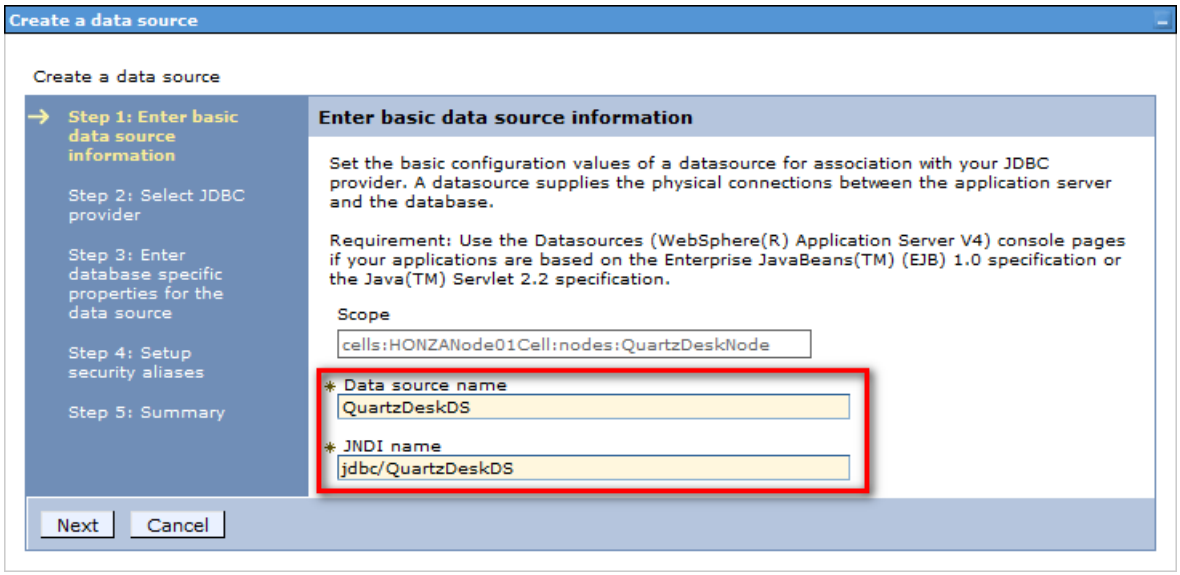
Alias: AuthQuartzDeskDS
User ID: DB_USER
Password: DB_PASSWORD

4.5 Data Source

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

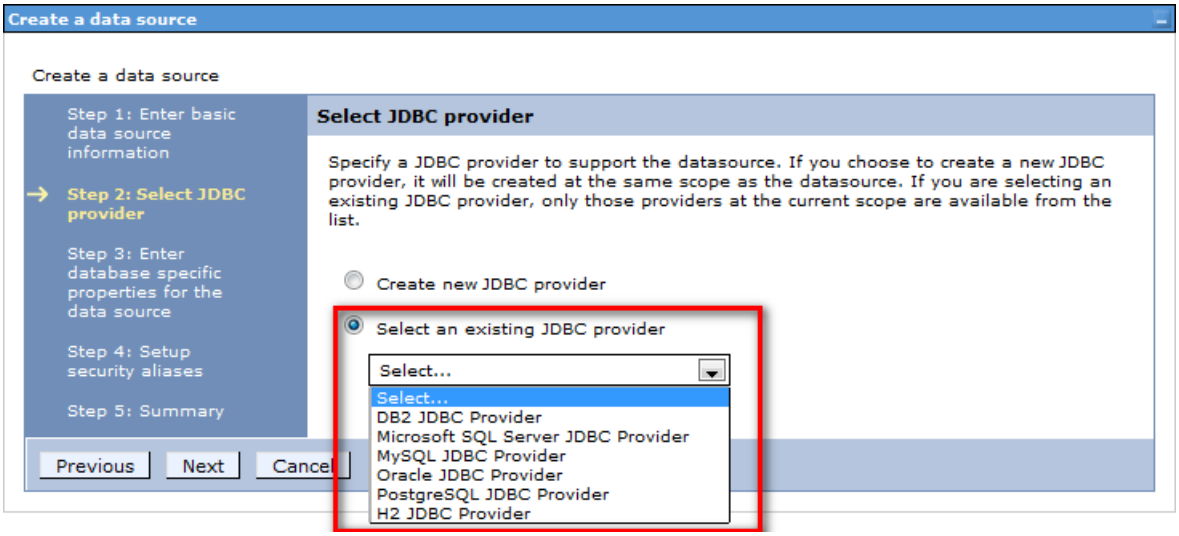
In Step 1, provide the data source name and JNDI name.

Data source name: QuartzDeskDS
JNDI name: jdbc/QuartzDeskDS



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

Copyright 2013-2017 QuartzDesk.com



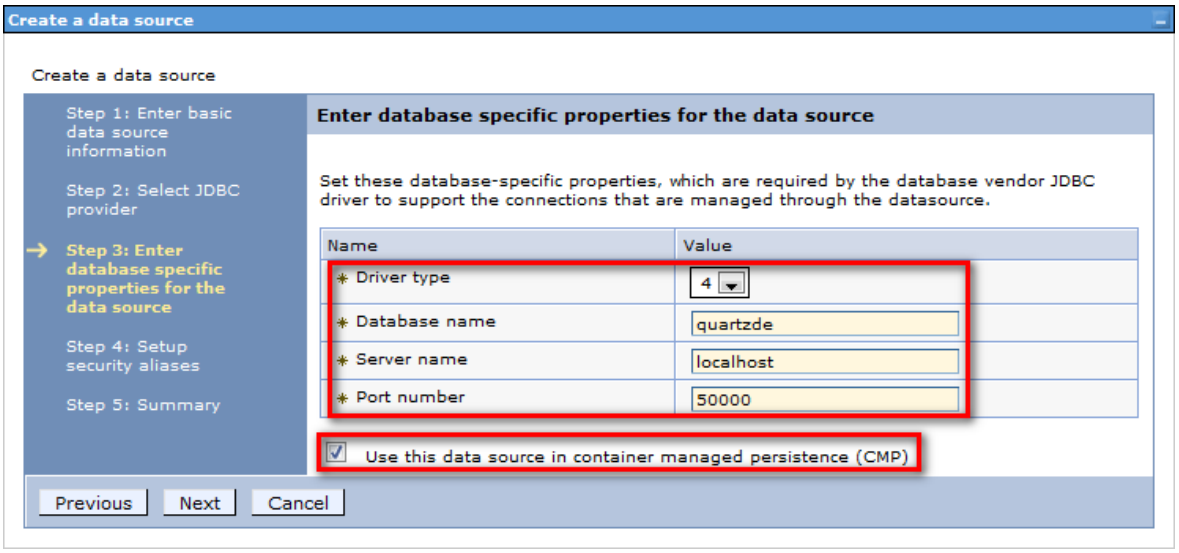
The following steps depend on the selected JDBC provider.

4.5.1 **DB2**

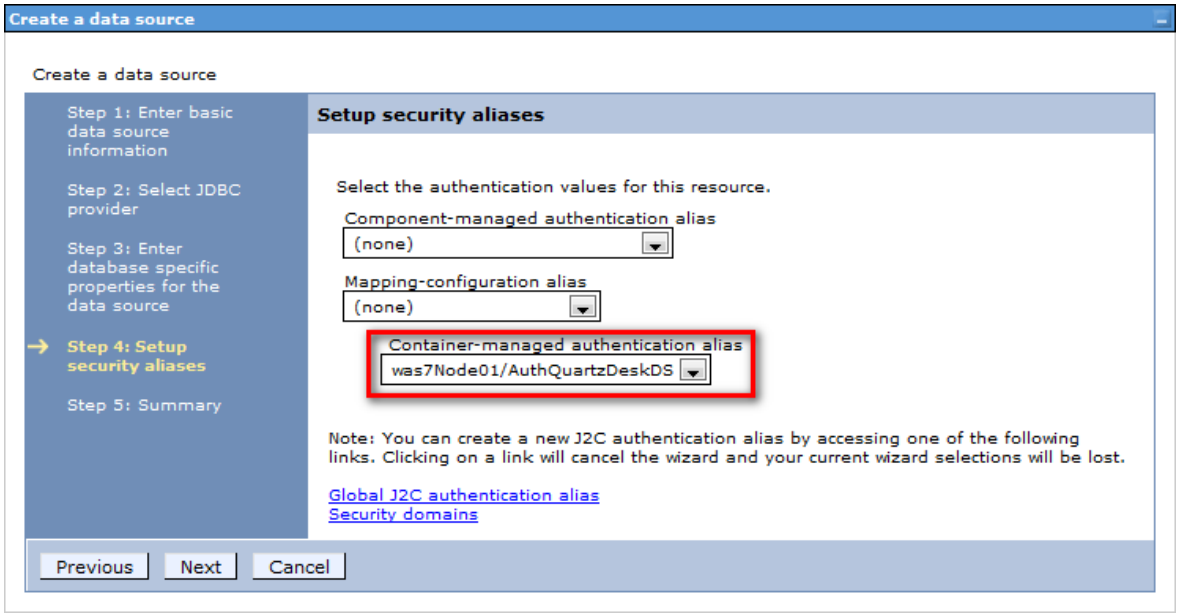
In Step 3, provide these values:

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.



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

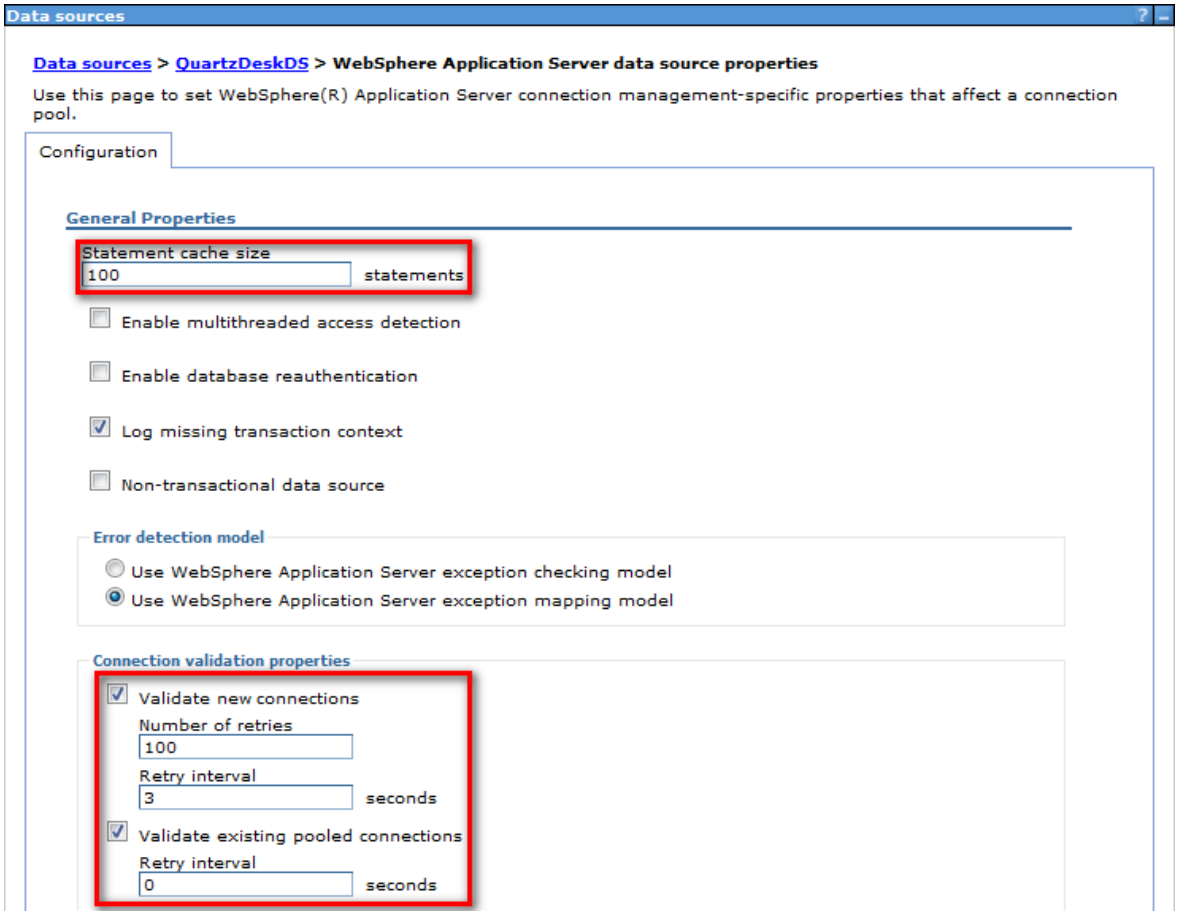
Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select “Validation by SQL query” with the following query:

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

Copyright 2013-2017 QuartzDesk.com



Apply and Save changes.

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

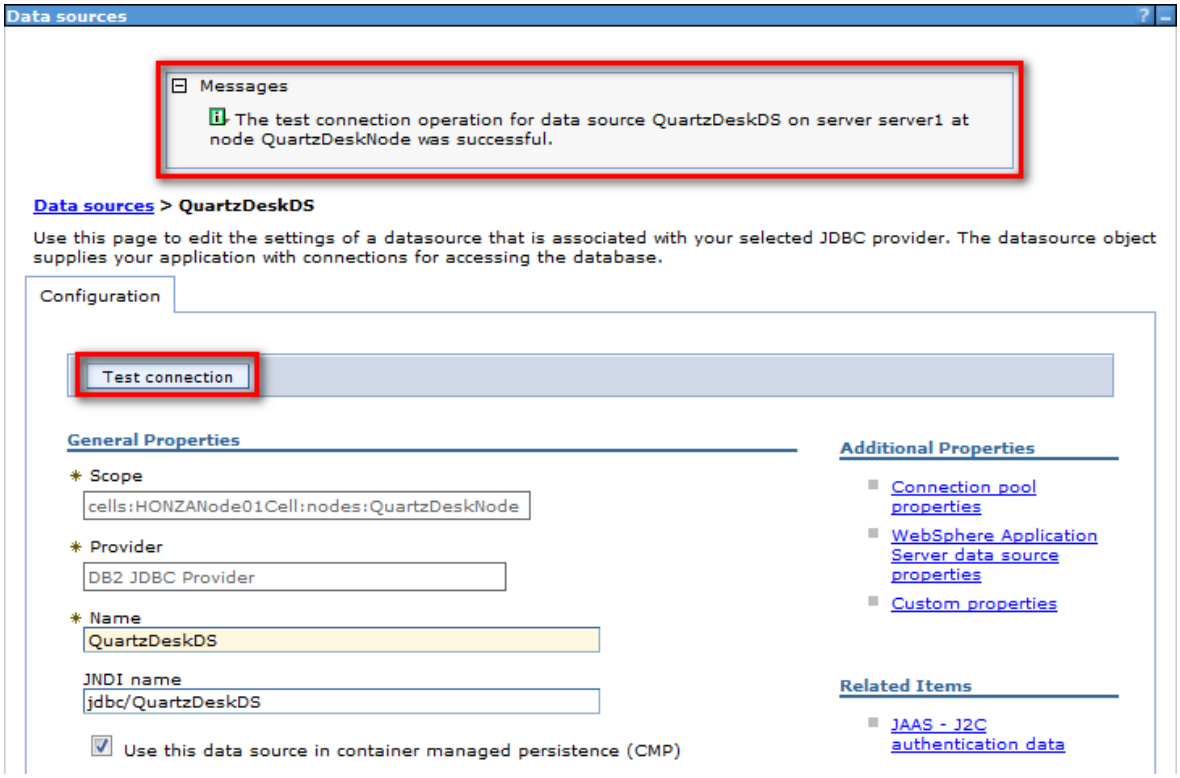
Set the following properties:

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.

Copyright 2013-2017 QuartzDesk.com



Check there are no errors displayed.

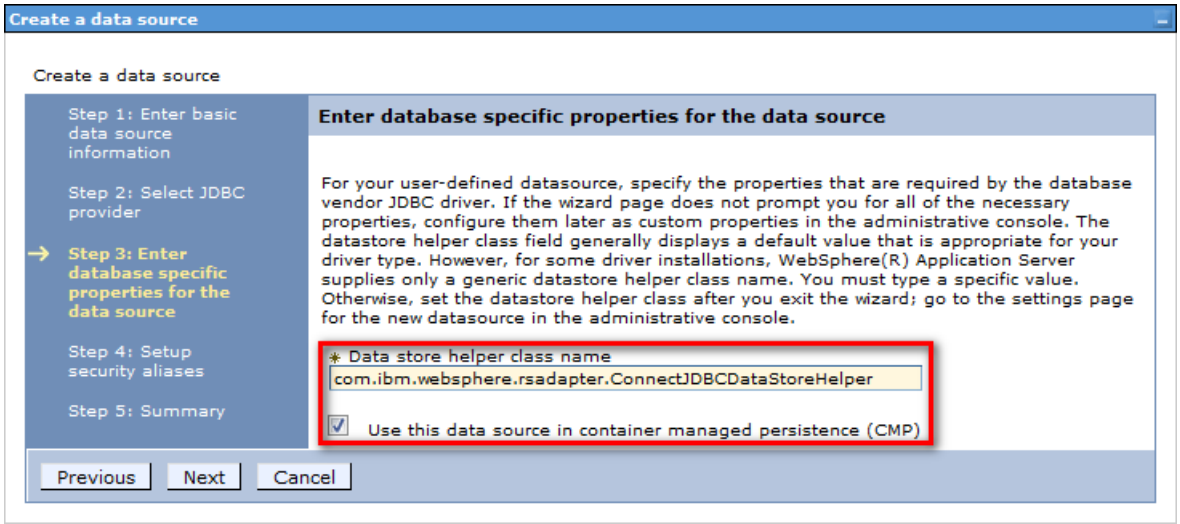
4.5.2 **H2**

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.

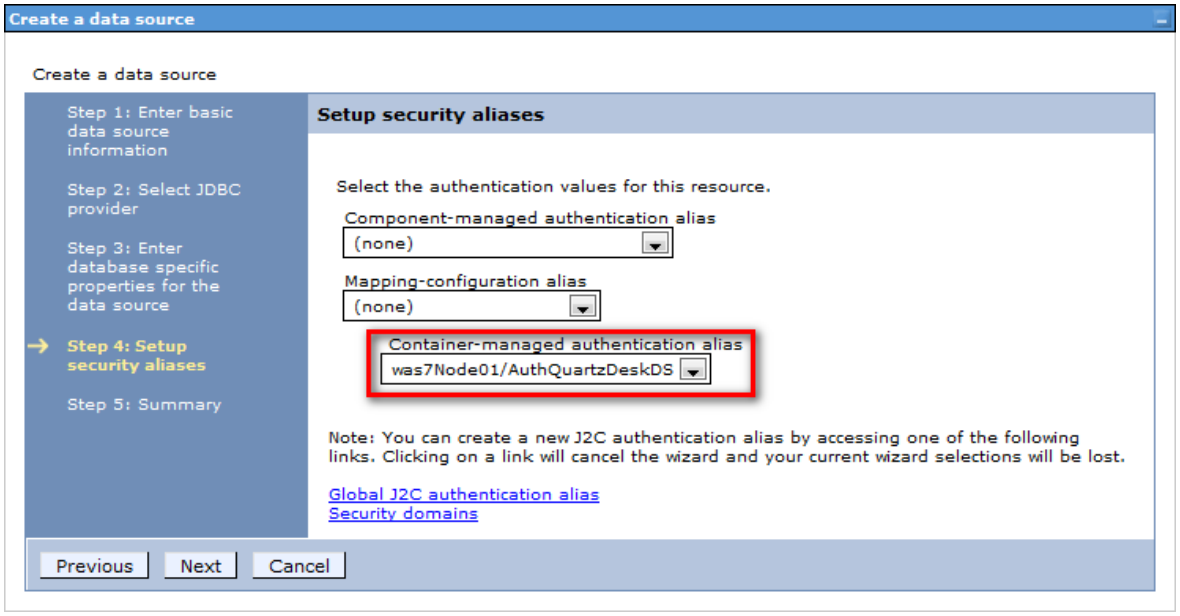
Data store helper class name:
`com.ibm.websphere.rsadapter.ConnectJDBCDataStoreHelper`

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



Copyright 2013-2017 QuartzDesk.com

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

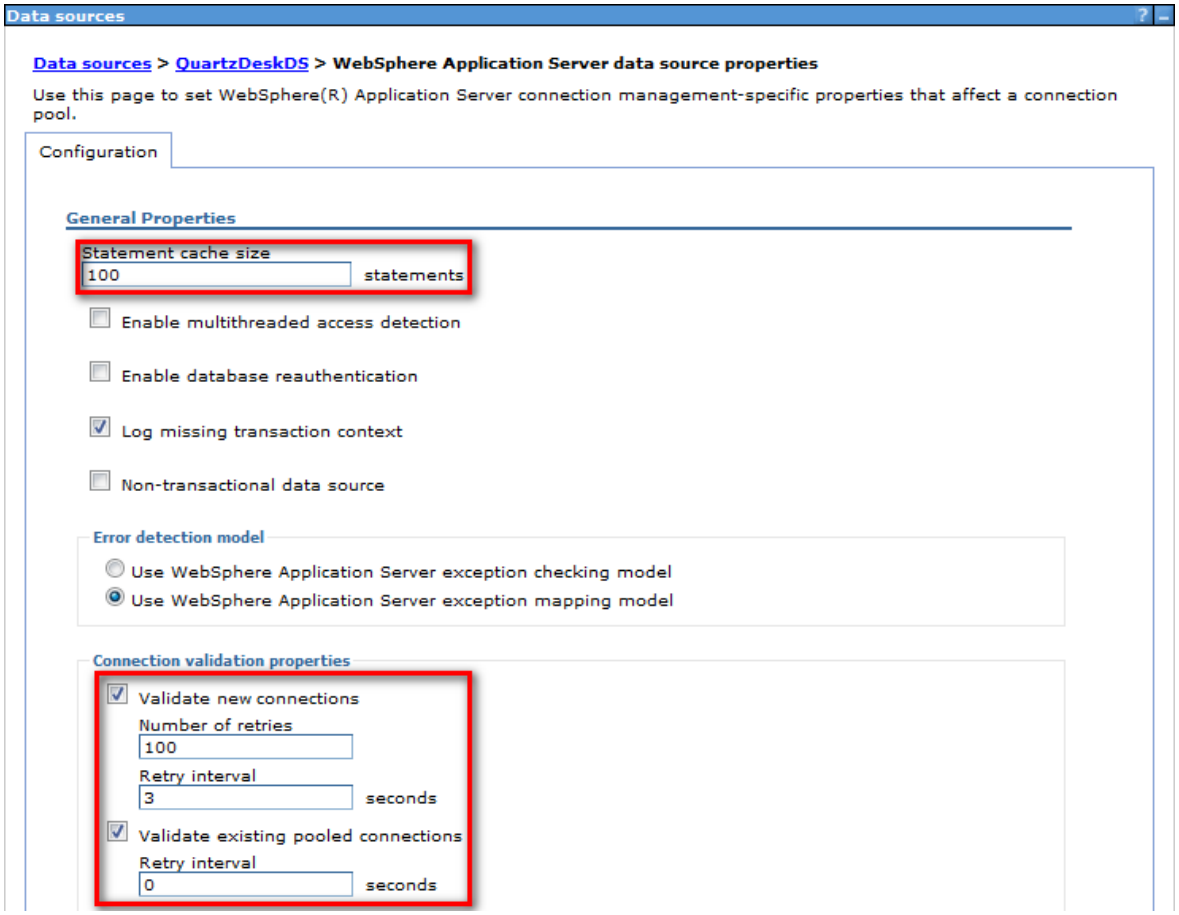
Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select “Validation by SQL query” with the following query:

```
select 1
```

Copyright 2013-2017 QuartzDesk.com



Apply and Save changes.

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

Set the following properties:

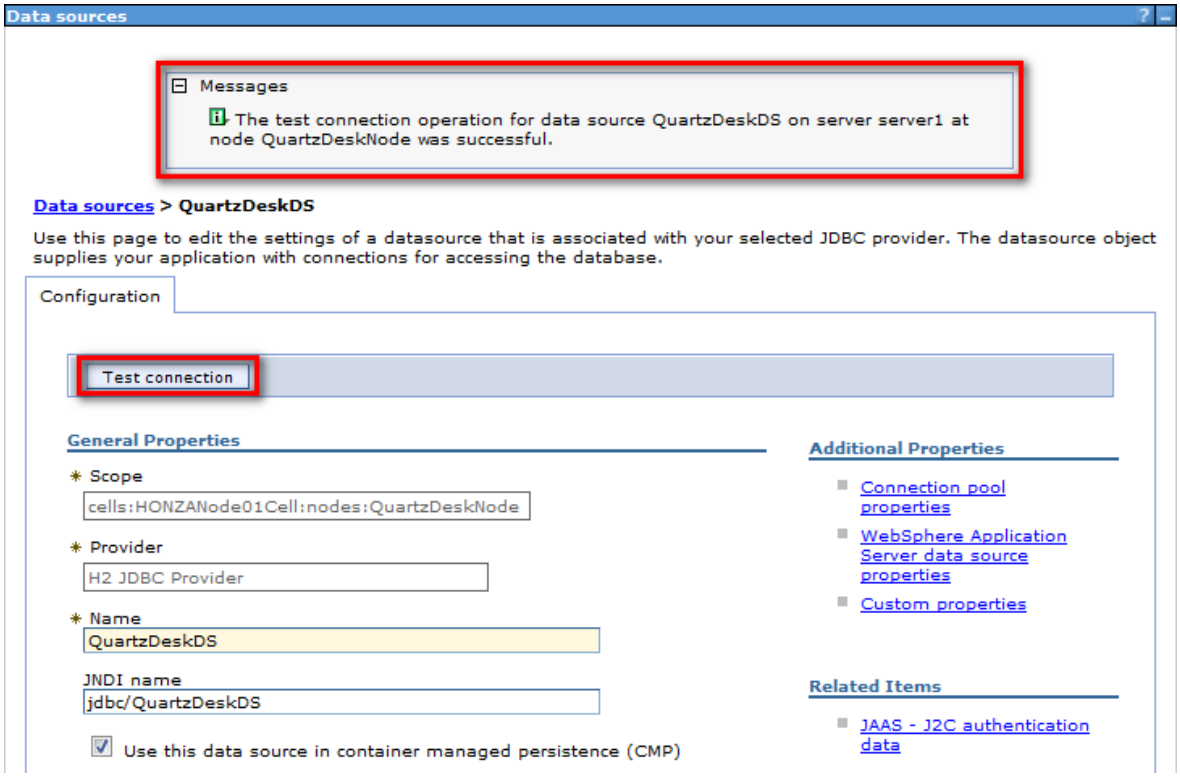
Name: URL
Value: `jdbc:h2:file:<H2_DB_FILE_PATH>`

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.

Copyright 2013-2017 QuartzDesk.com



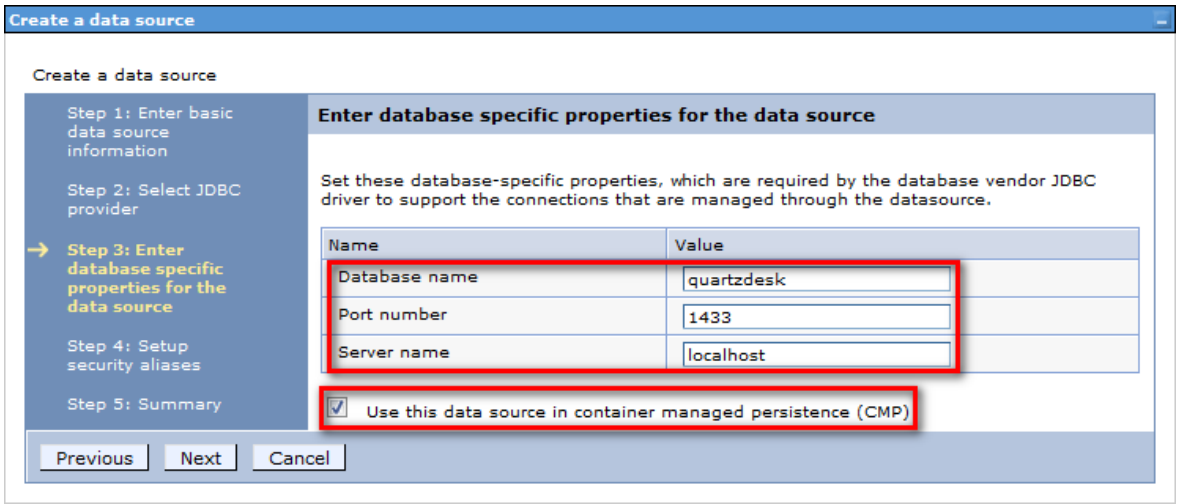
Check there are no errors displayed.

4.5.3 Microsoft SQL Server

In Step 3, provide these values:

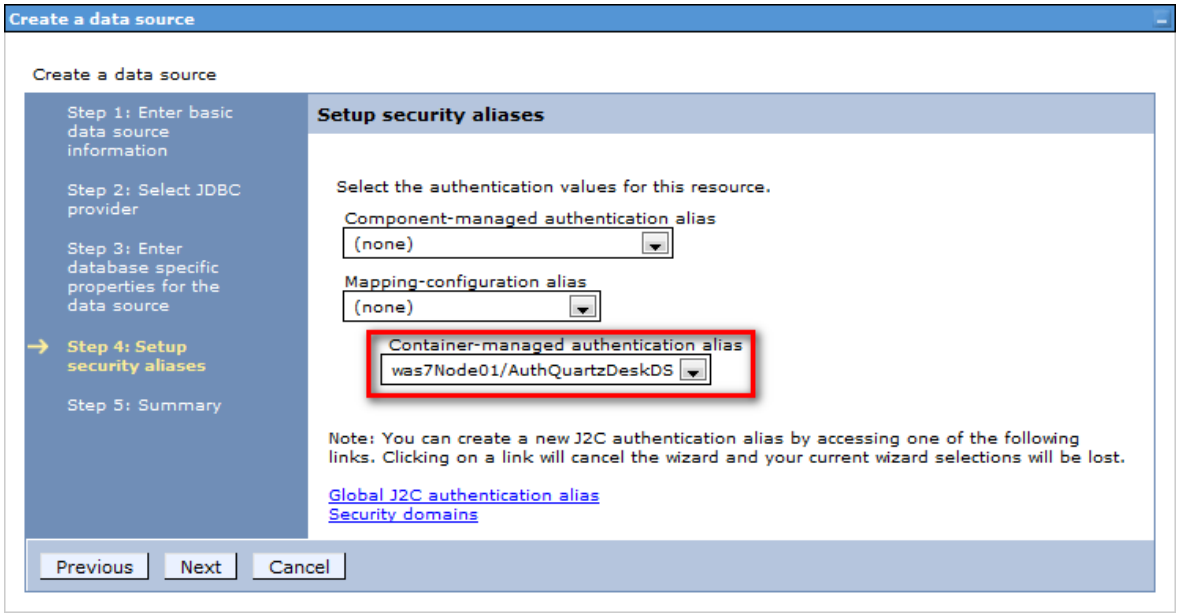
Database name: DB_NAME
Server name: DB_HOST
Port number: DB_PORT

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.

Copyright 2013-2017 QuartzDesk.com



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.

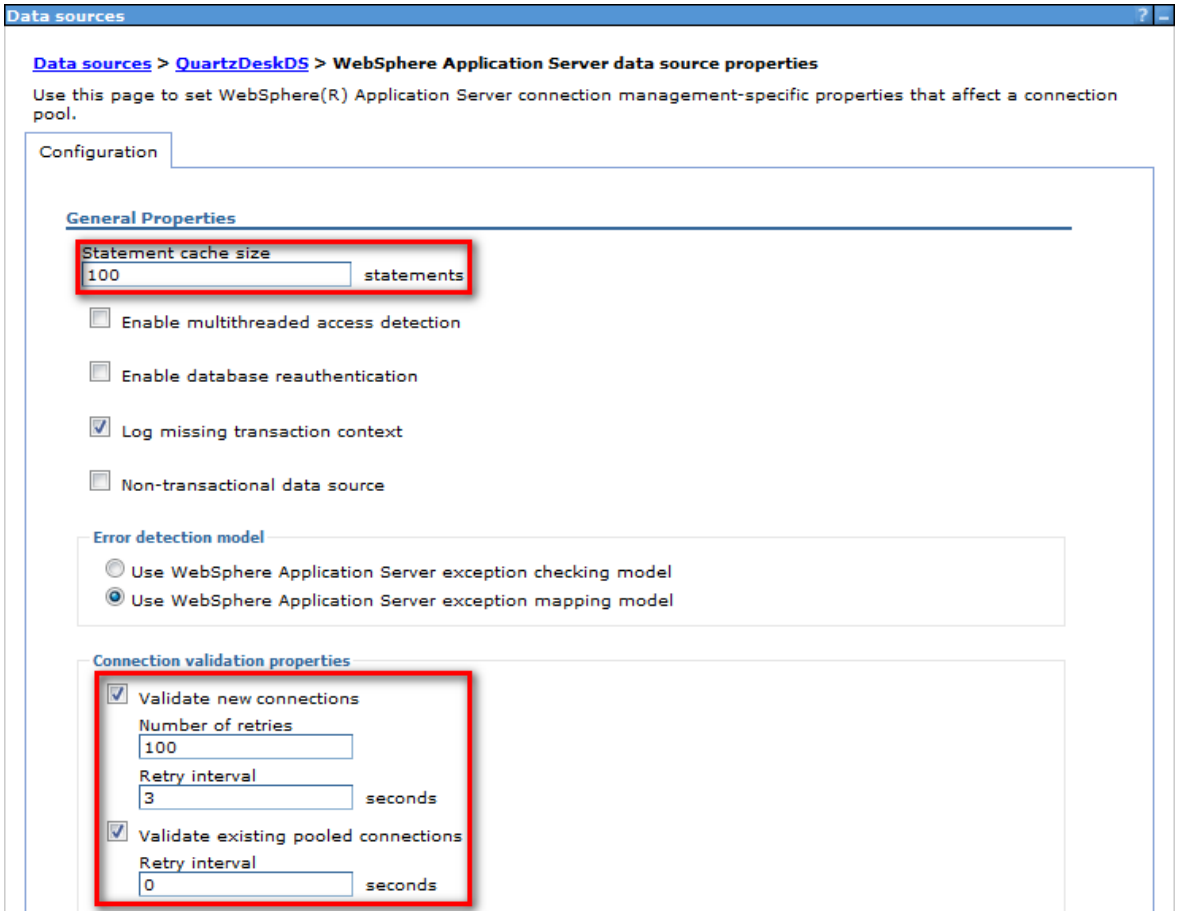
Set the following properties:
Statement cache size: 100

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
```



Apply and Save changes.

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

Set the following properties:

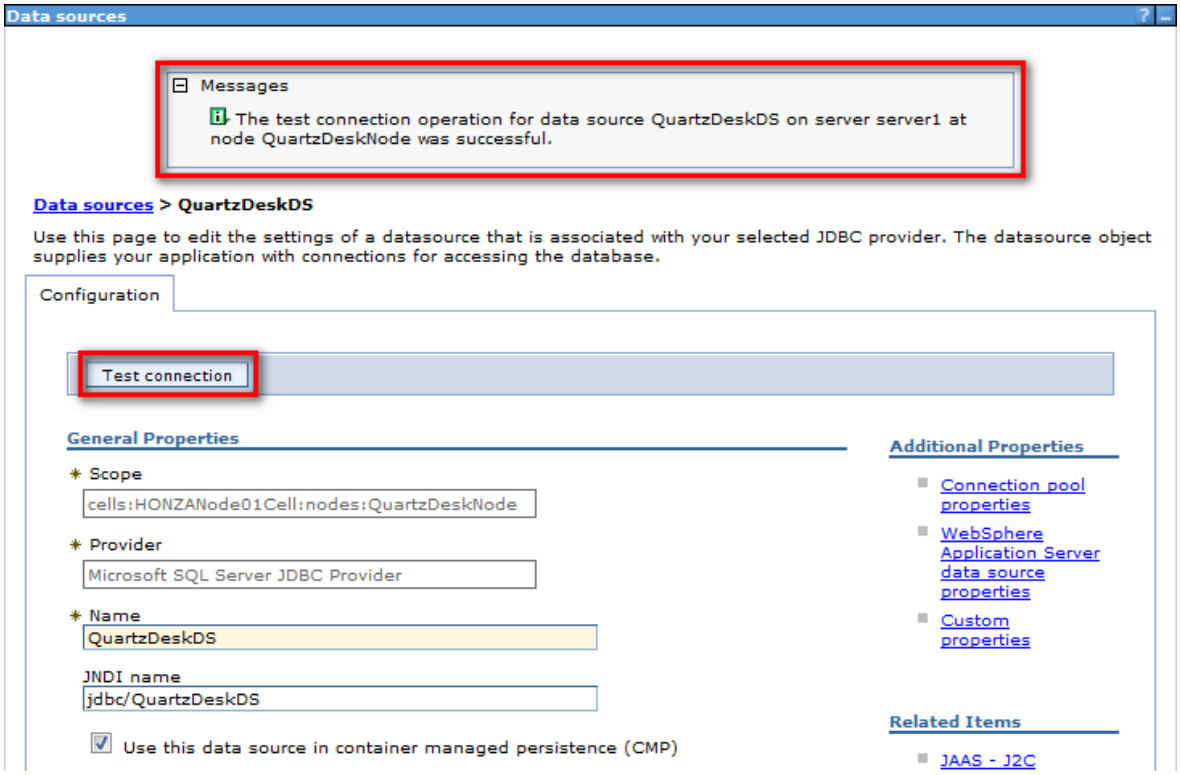
Name: applicationName
Value: QuartzDesk

Depending on your Microsoft SQL Server configuraton, 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.

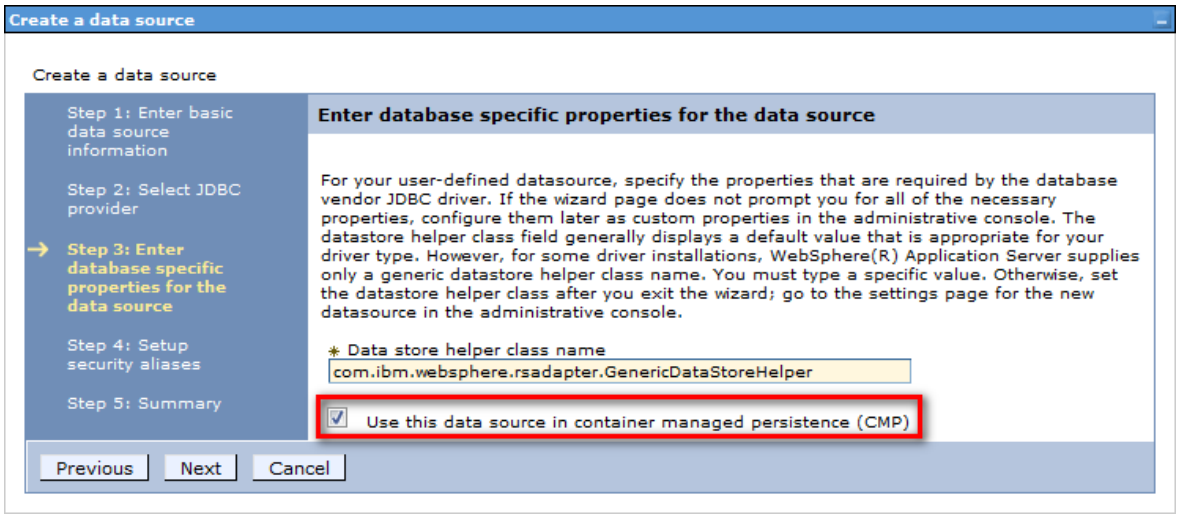
Copyright 2013-2017 QuartzDesk.com



Check there are no errors displayed.

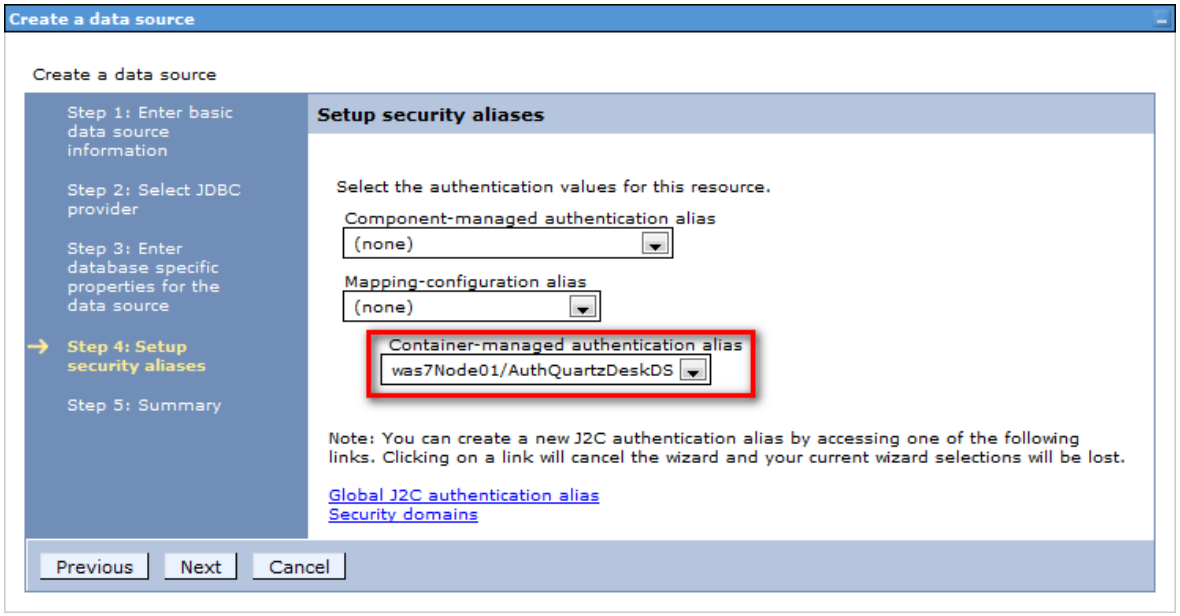
4.5.4 MySQL

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



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

Copyright 2013-2017 QuartzDesk.com



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

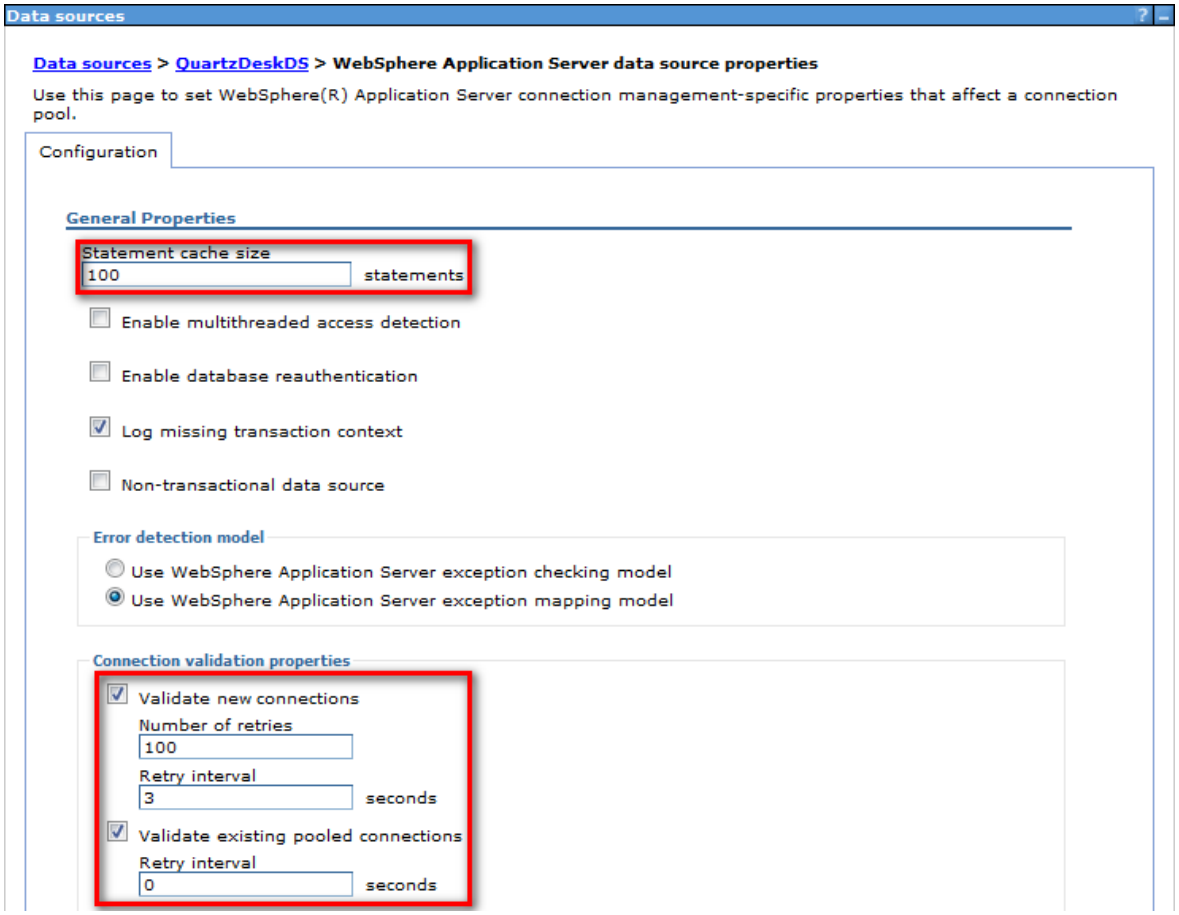
Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select “Validation by SQL query” with the following query:

```
select 1
```

Copyright 2013-2017 QuartzDesk.com



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

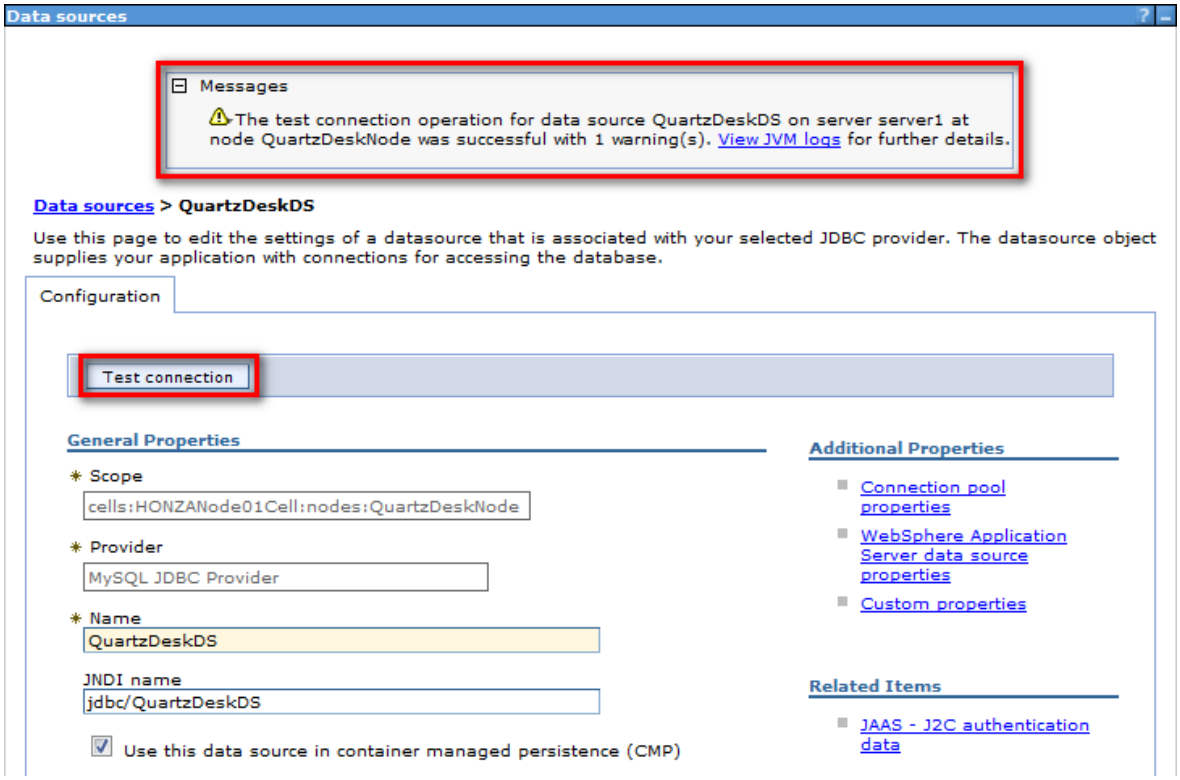
Name: serverName
Value: DB_HOST

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.

Copyright 2013-2017 QuartzDesk.com



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

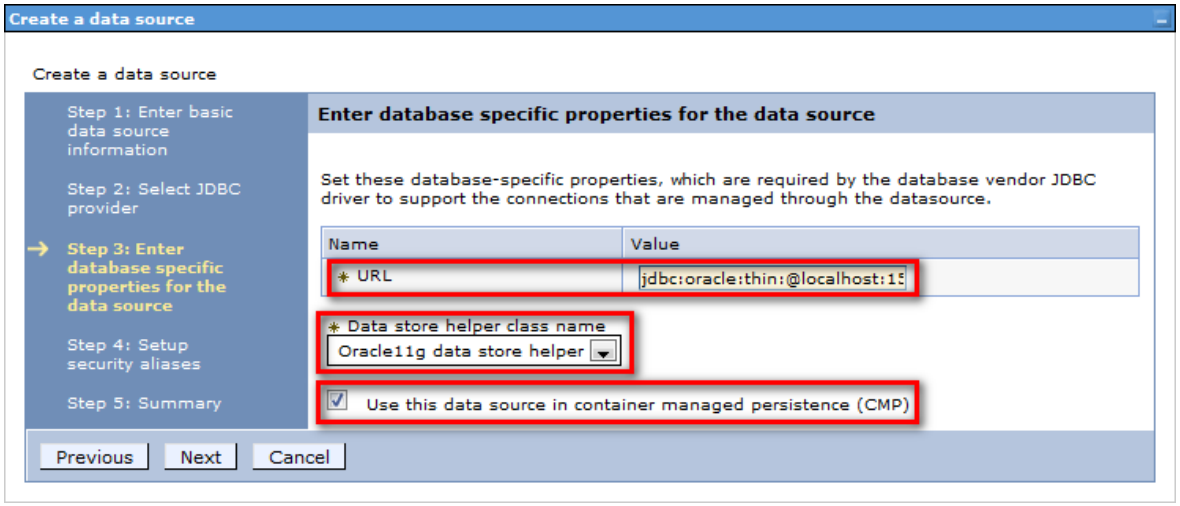
In Step 3, provide the URL value:

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

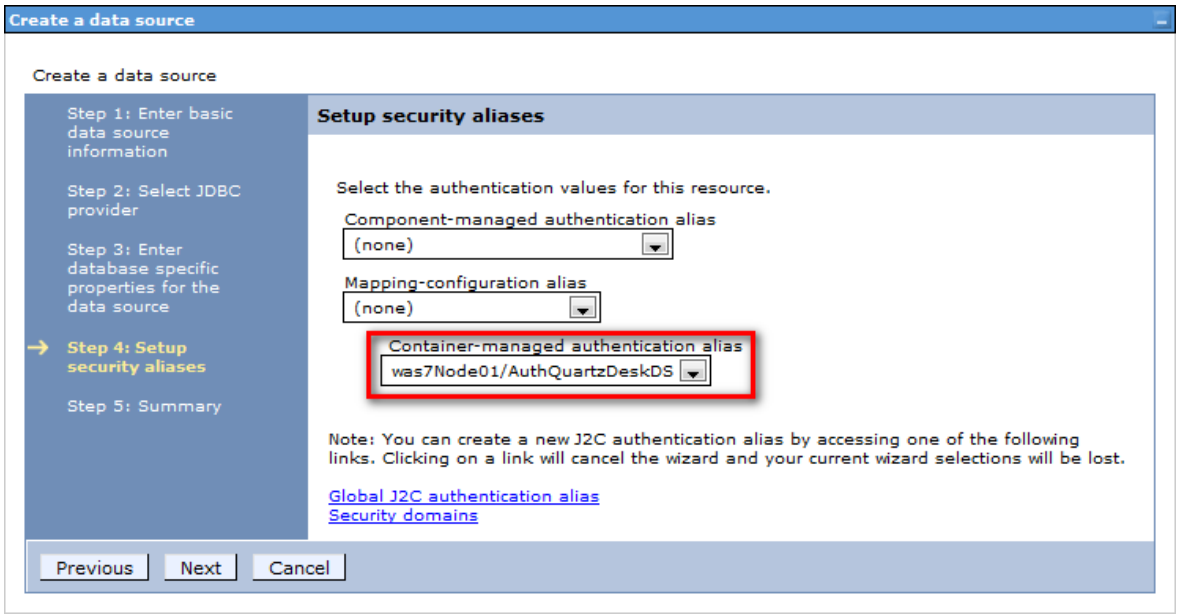
Select the “Data store helper class name” based on your Oracle database version.

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

Copyright 2013-2017 QuartzDesk.com



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

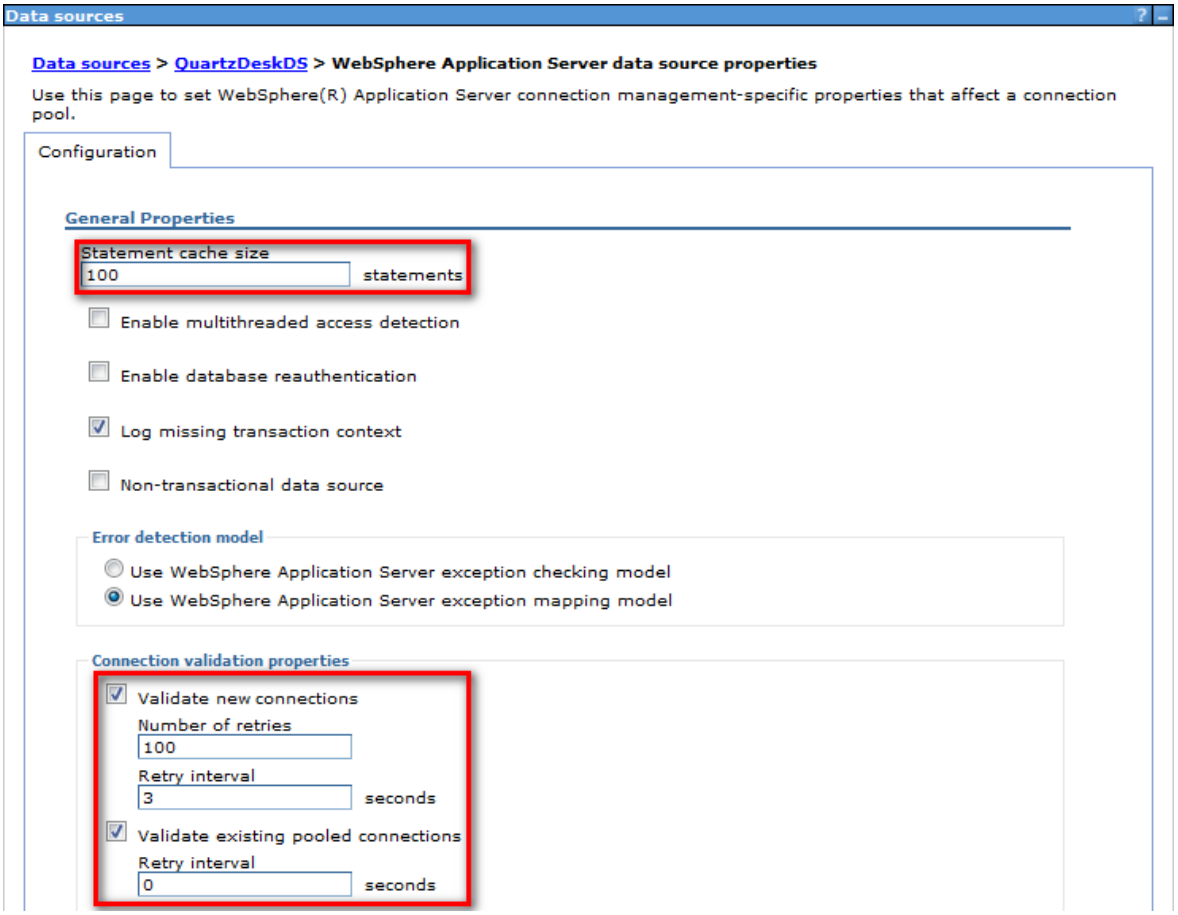
Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select “Validation by SQL query” with the following query:

```
select 1 from dual
```

Copyright 2013-2017 QuartzDesk.com



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

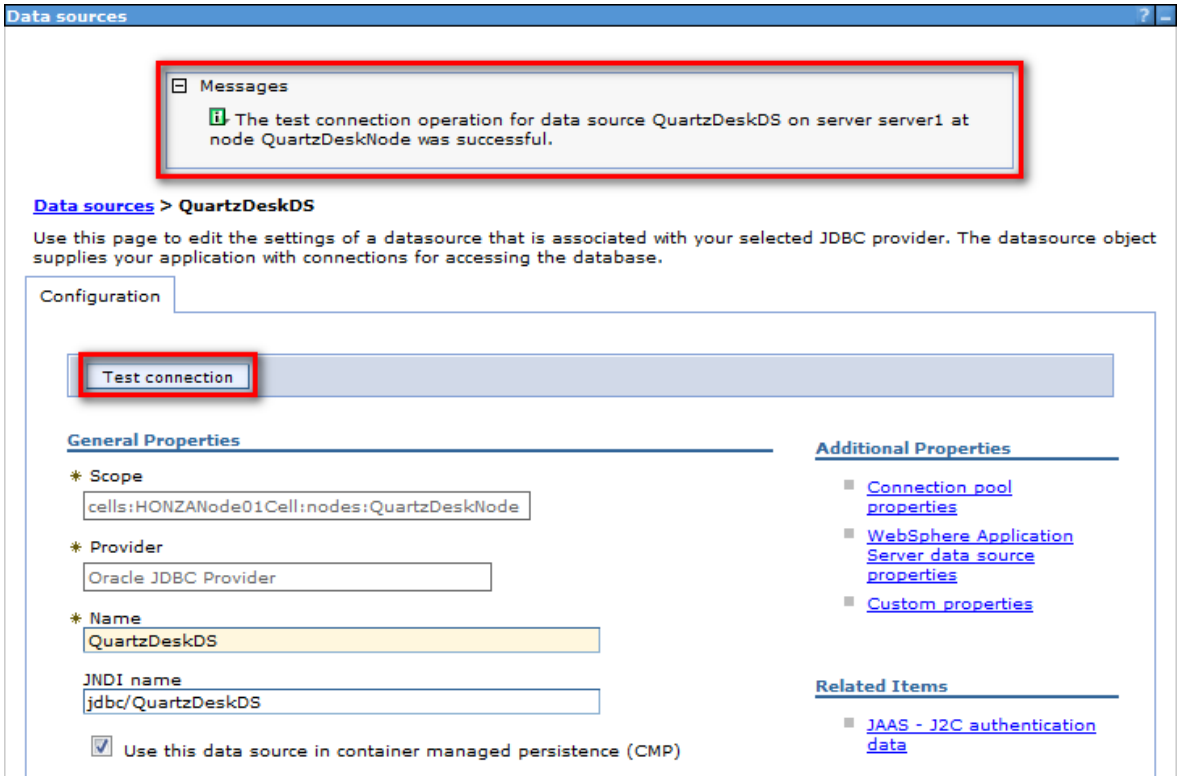
Name: serverName
Value: DB_HOST

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.

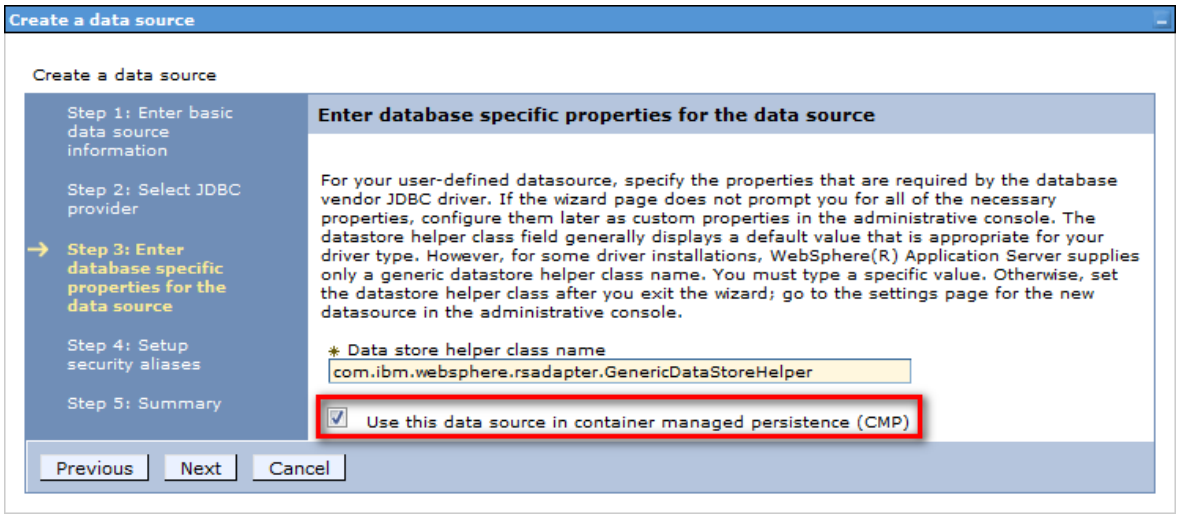
Copyright 2013-2017 QuartzDesk.com



Check there are no errors displayed.

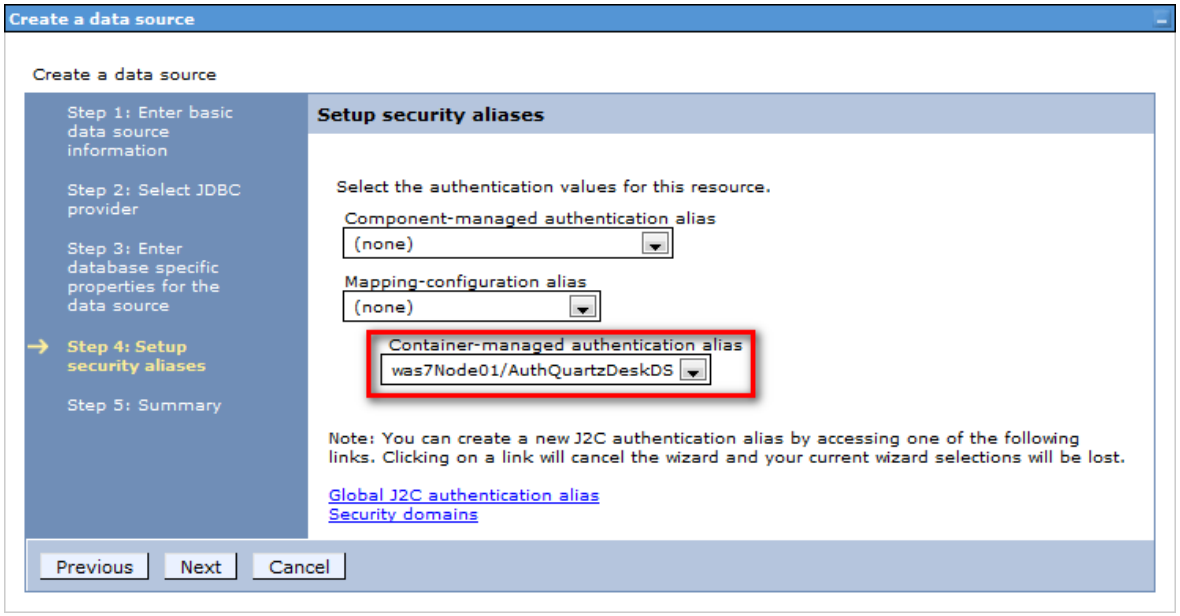
4.5.6 PostgreSQL

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.

Copyright 2013-2017 QuartzDesk.com



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

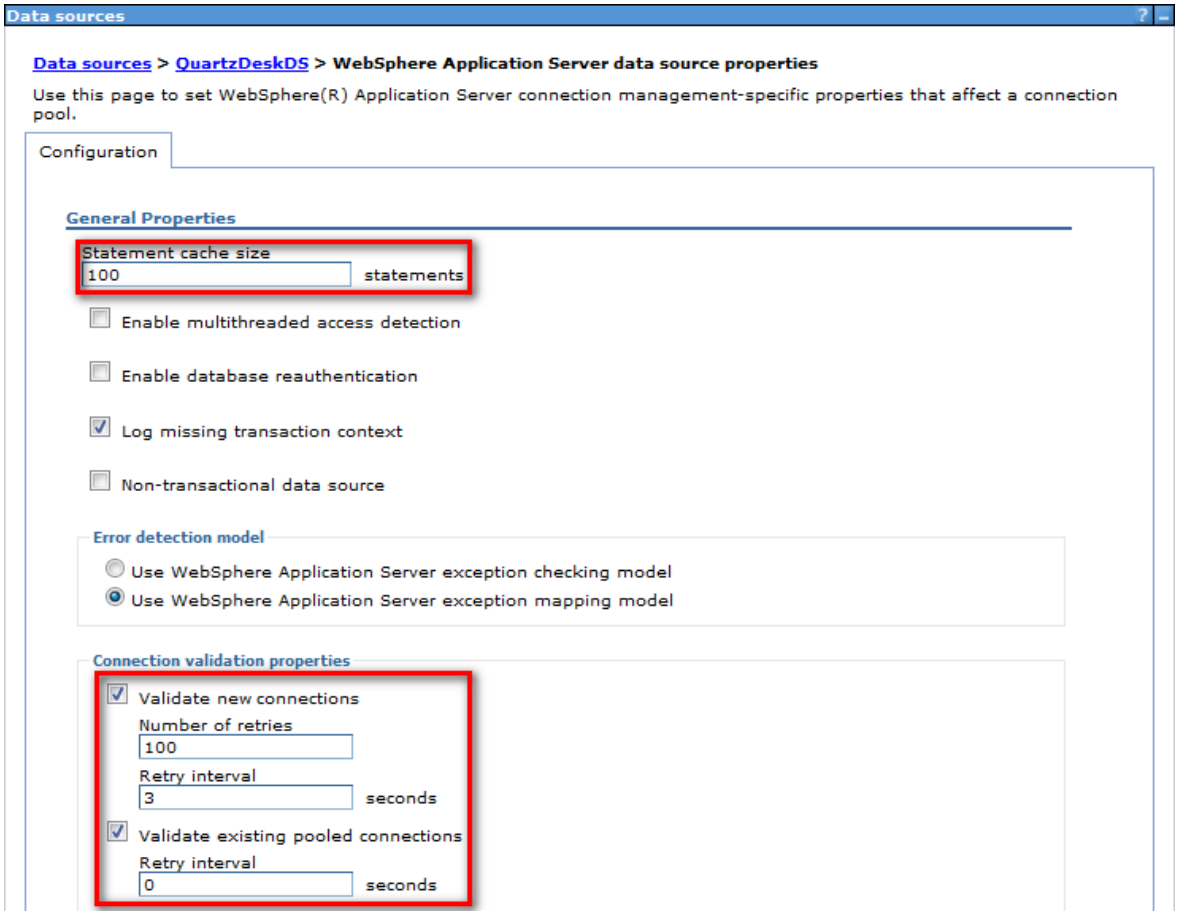
Validate new connectons: checked
Number of retries: 100
Retry Interval: 3

Validate existing pooled connections: checked
Retry interval: 0

Select “Validation by SQL query” with the following query:

```
select 1
```

Copyright 2013-2017 QuartzDesk.com



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

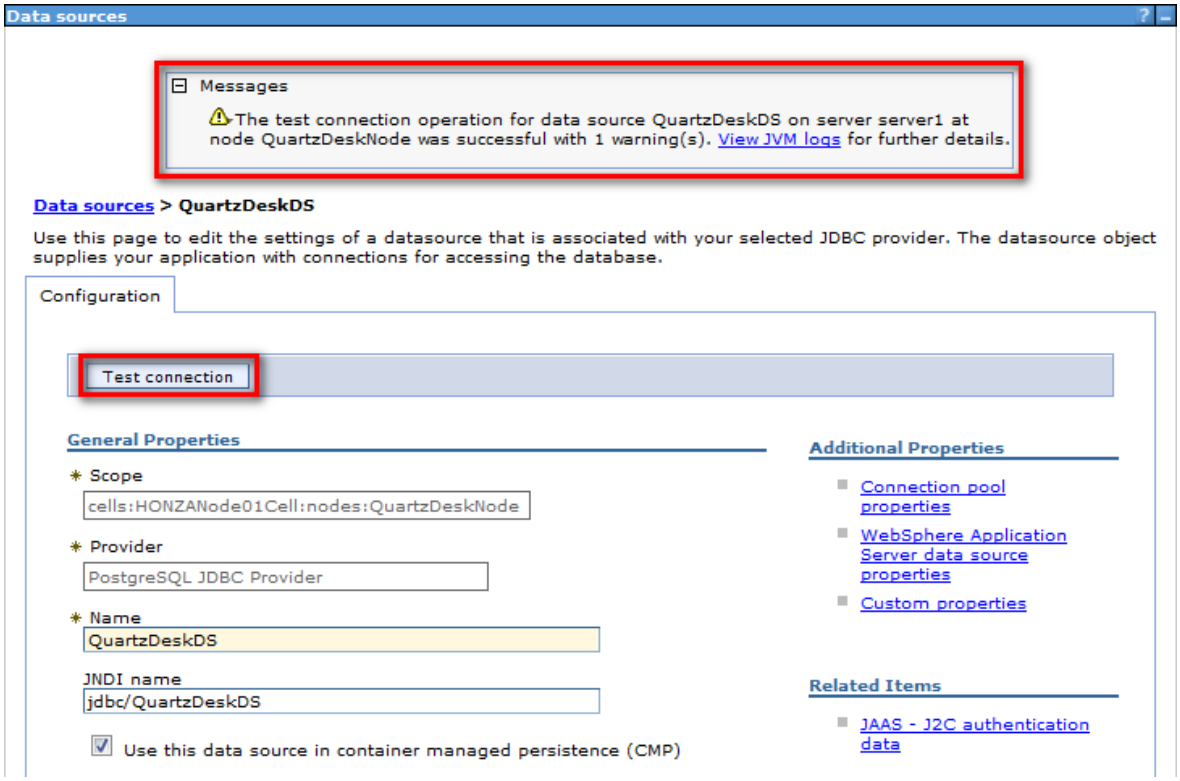
Name: serverName
Value: DB_HOST

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.

Copyright 2013-2017 QuartzDesk.com



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

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

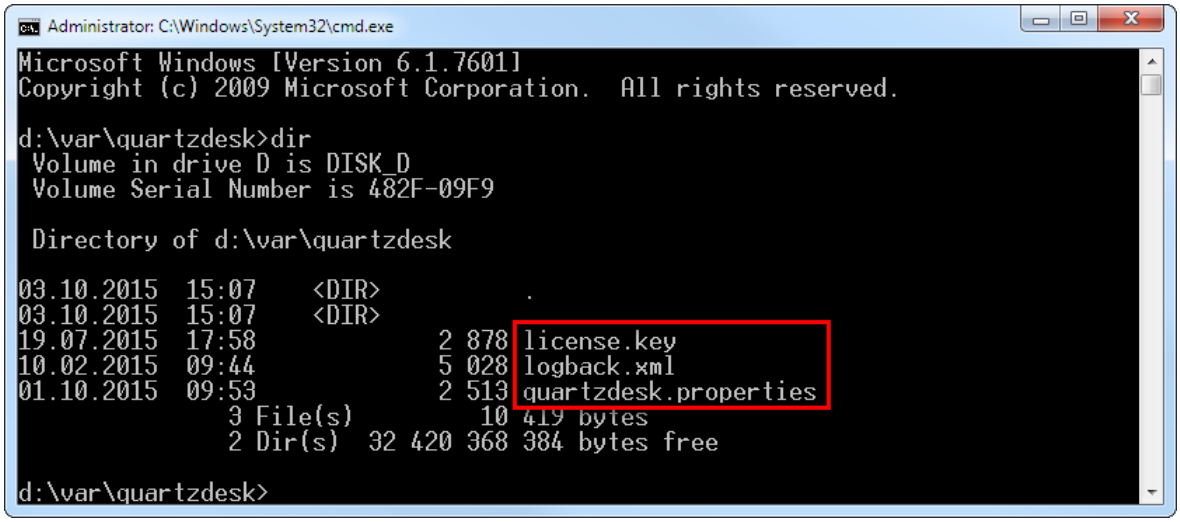
You can obtain a free 30-day trial license key at www.quartzdesk.com (open the Try / Purchase menu).

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.

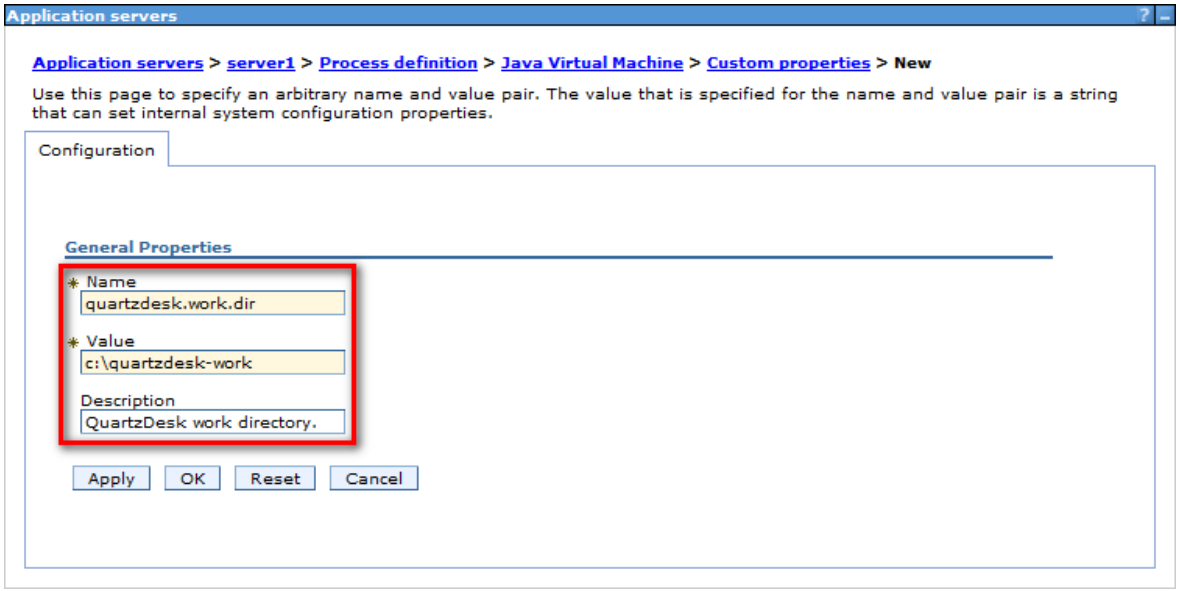
In the following figure you can see an example of a QuartzDesk work directory correctly set up on a Microsoft Windows machine.

Copyright 2013-2017 QuartzDesk.com



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.work.dir
Value: WORK_DIR



Apply and Save changes.

4.7 Application Configuration

Open the QuartzDesk configuration file `WORK_DIR/quartzdesk.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 logging parameters by editing the `WORK_DIR/logback.xml` configuration file. The default sample `logback.xml` configuration file makes QuartzDesk log under the `WORK_DIR/logs` directory that is automatically created when QuartzDesk 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.

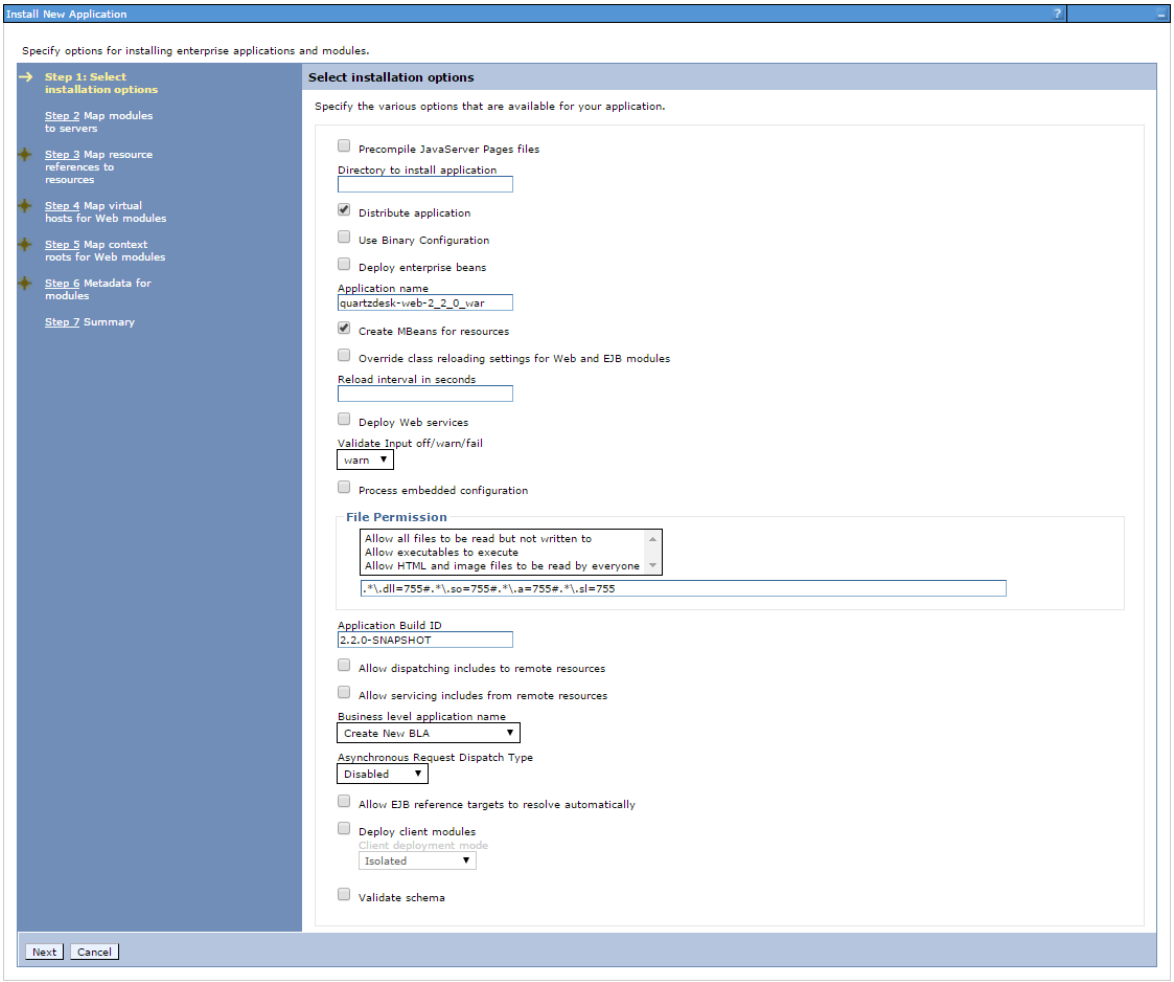
In WAC go to Applications → Application Types → WebSphere enterprise applications.

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.

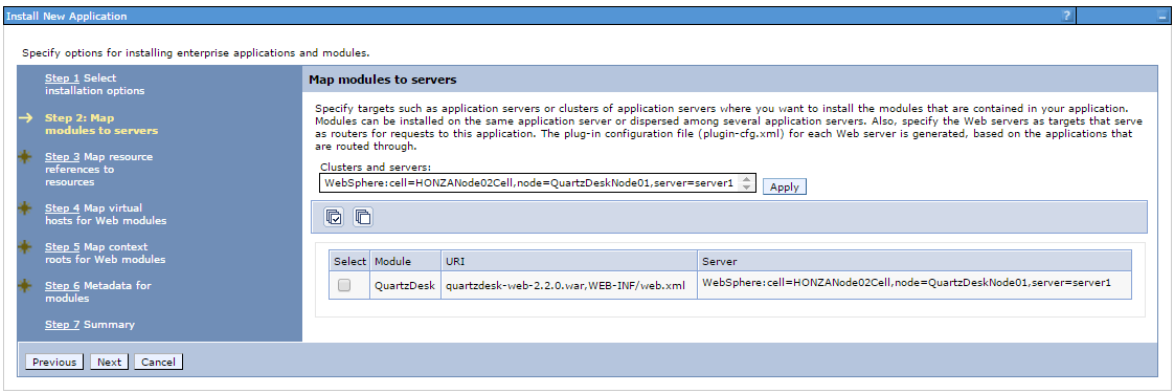
Step 1 – Select installation options

Click Next.



Step 2 – Map modules to servers

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

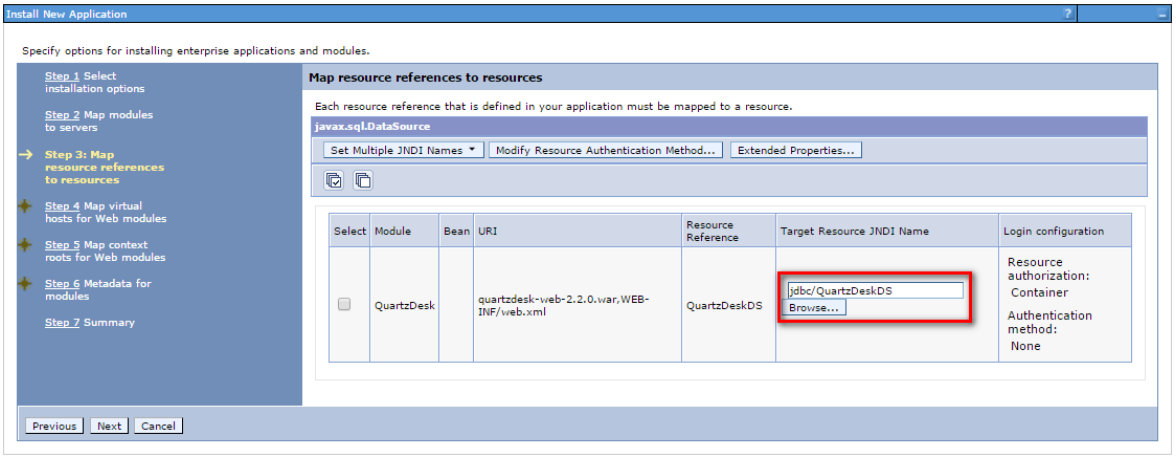


Click Next.

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.

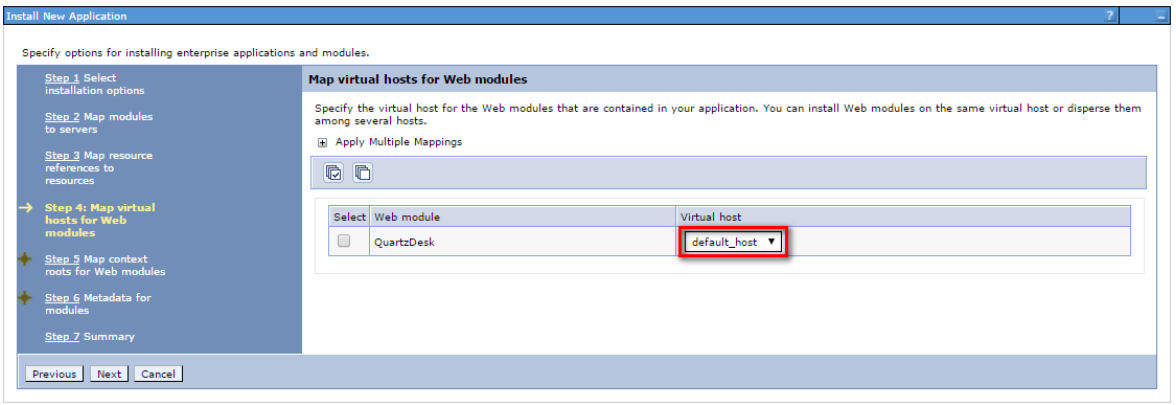
Copyright 2013-2017 QuartzDesk.com



Click Next.

Step 4 – Map virtual hosts for Web modules

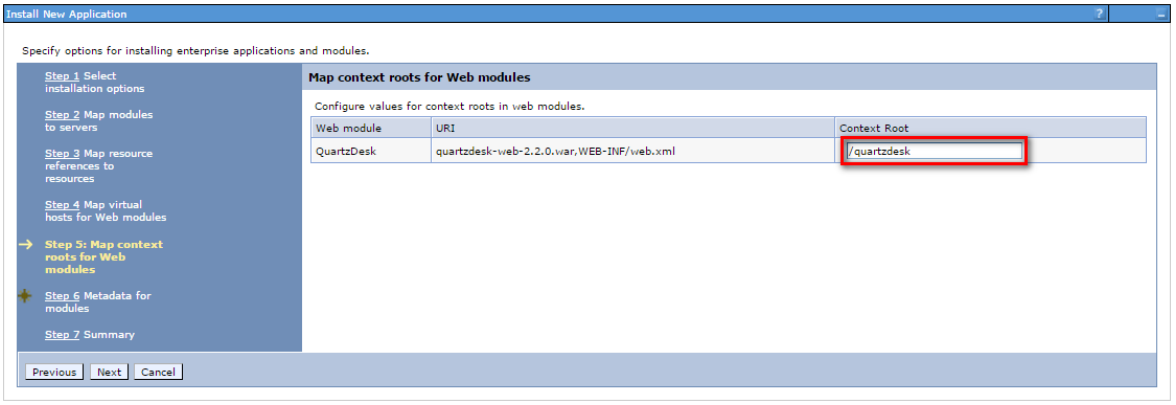
Map the QuartzDesk web module to the desired WebSphere virtual host.



Click Next.

Step 5 – Map context roots for Web modules

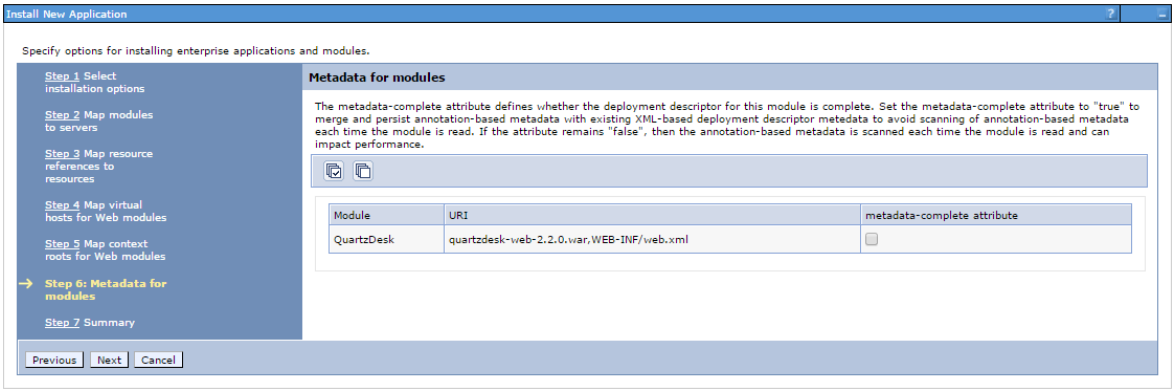
Provide the web servlet context root for the QuartzDesk web application. We recommend using "/quartzdesk" (without quotes).



Click Next.

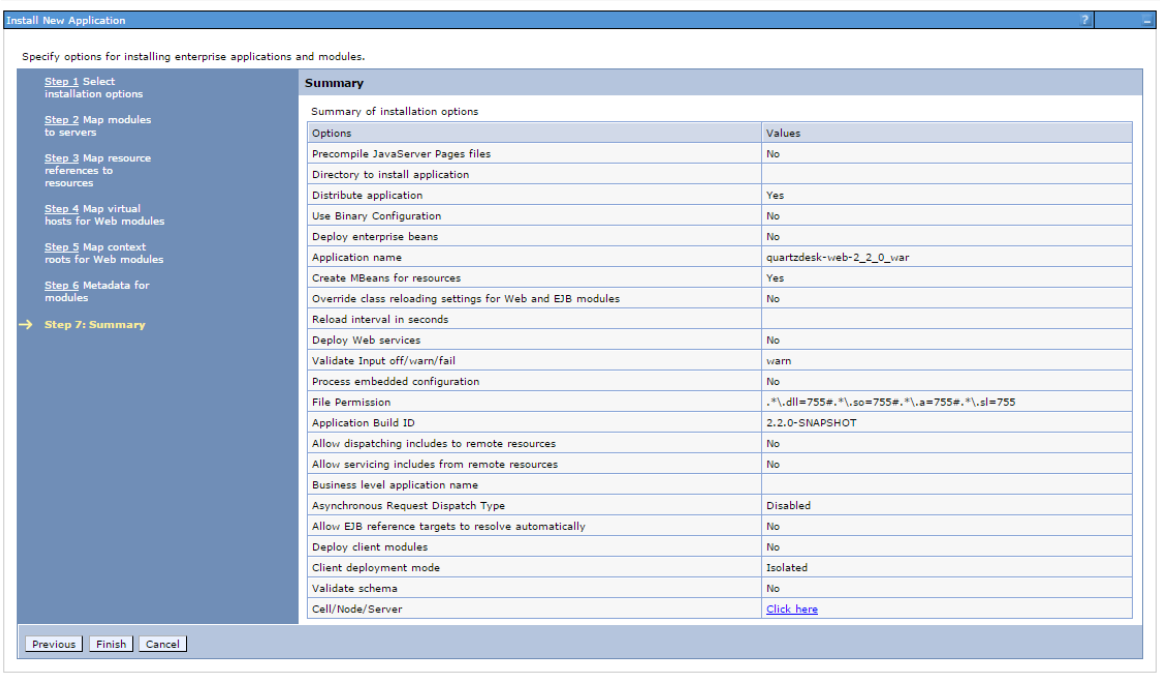
Copyright 2013-2017 QuartzDesk.com

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



Click Next.

Step 7 – Summary



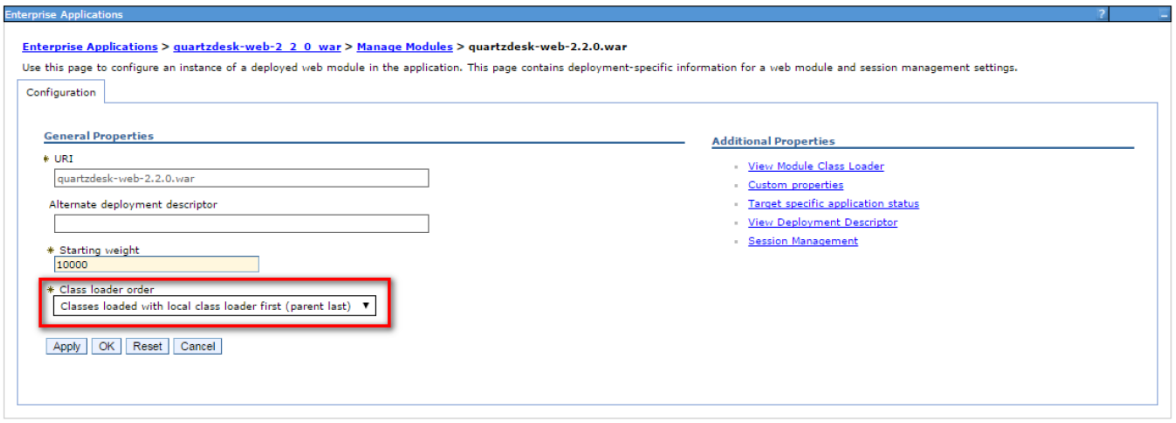
Click Finish.
When the installation completes, Save changes.

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.

Copyright 2013-2017 QuartzDesk.com

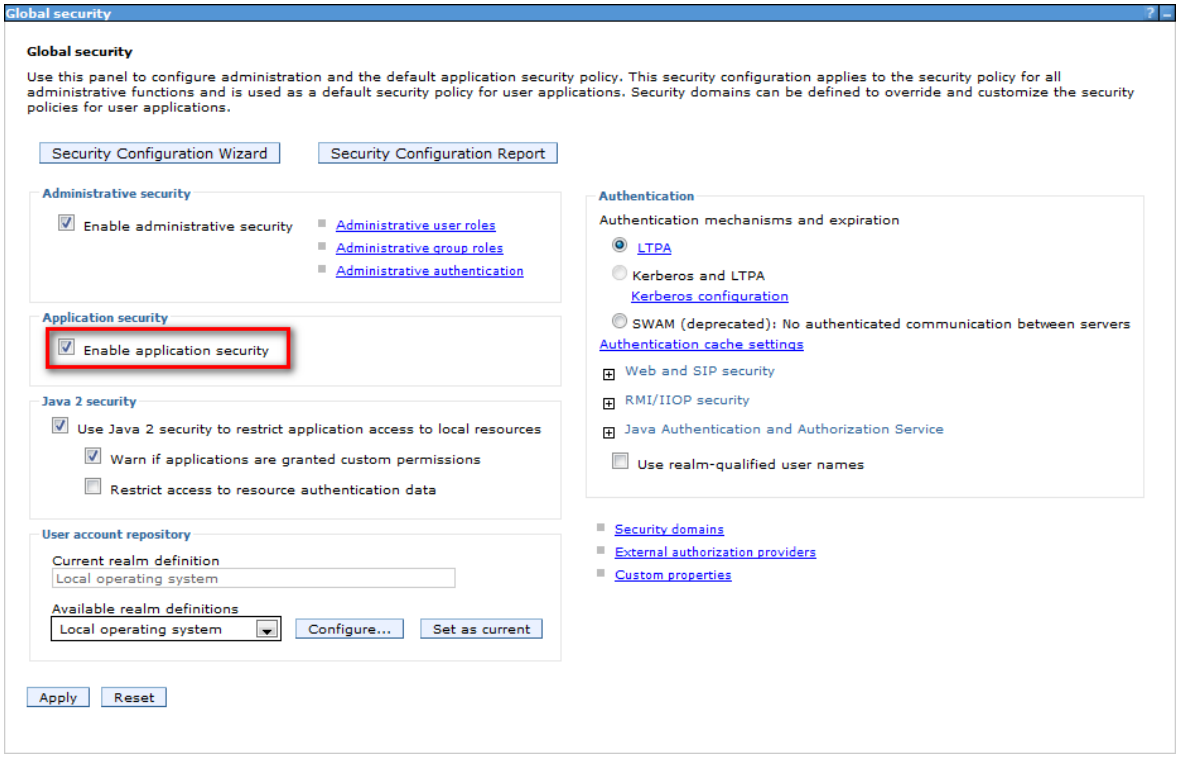


4.9 Security

QuartzDesk supports the HTTP/S Basic authentication scheme to authenticate users who access the application. To configure application security, perform the following two steps:

4.9.1 Enable application security

In WAC go to Security → Global security. Make sure the Enable application security checkbox is checked.



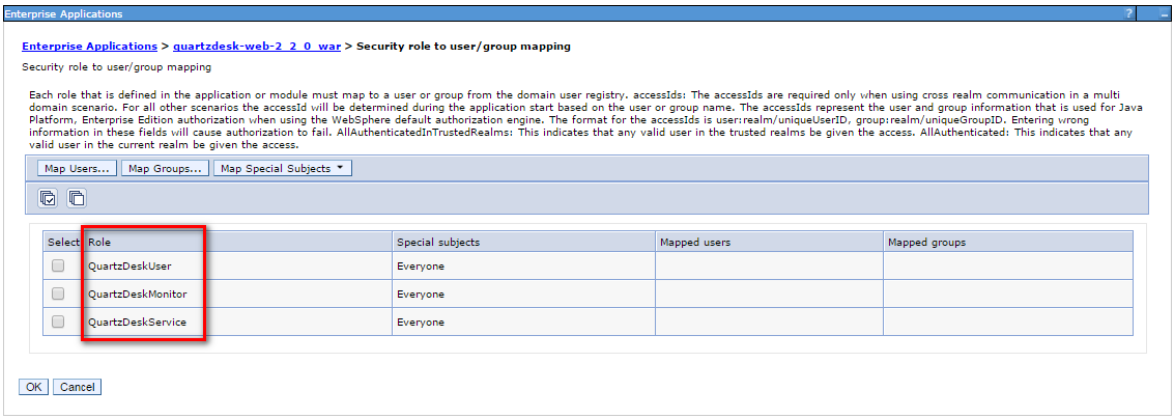
4.9.2 Map security roles to users / groups

In WAC go to Applications → Application Types → WebSphere enterprise applications → quartzdesk-web-x_y_x_war → Security role to user/group mapping and map the following 3 QuartzDesk security roles to the desired users / groups defined in WAS security realms.

Security Role	Description

QuartzDeskUser	Role required to access the QuartzDesk web application UI (QuartzDesk GUI).
QuartzDeskMonitor	Role required to access the scheduler, job and trigger monitoring URLs (REST API).
QuartzDeskService	Role required to access QuartzDesk web-services (e.g. the QuartzAnywhere web-service).

By mapping the QuartzDesk security role to the special subject Everyone, you effectively disable user authentication for that role. In the following figure user authentication has been disabled for the entire application.



Click OK. Save changes.

4.10 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 in the `WORK_DIR/logs` directory) for errors.

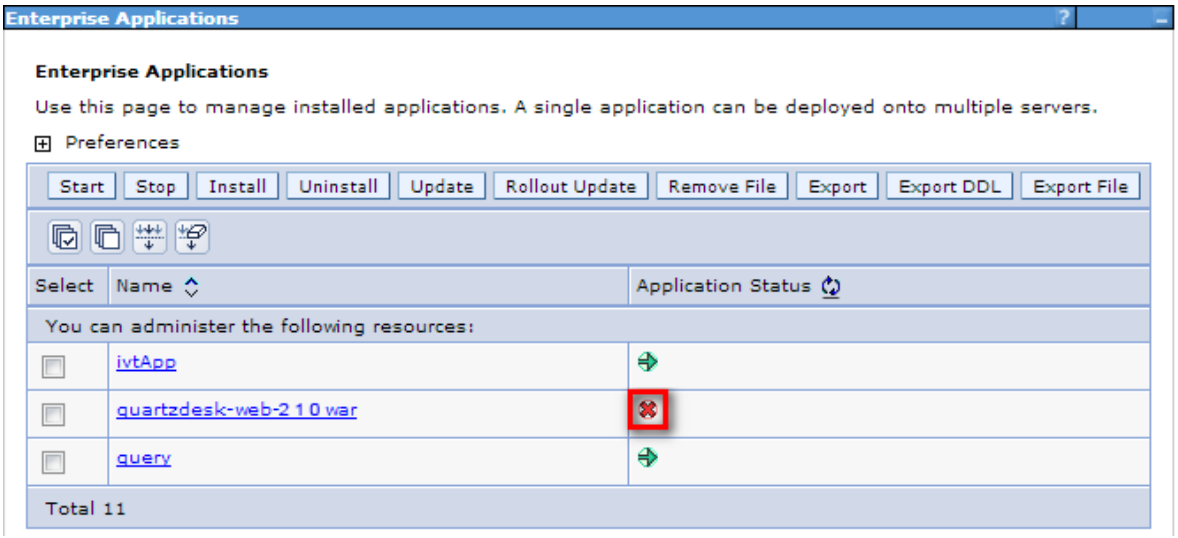
If there are no errors, point your browser to `http://WAS_HTTP_HOST:WAS_HTTP_PORT/quartzdesk` and verify that the QuartzDesk web application works.

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 database. We recommend performing a **full database backup**.

Backup the contents of the QuartzDesk 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.

Copyright 2013-2017 QuartzDesk.com

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. Continue with the security role mapping configuration in 4.9.2.

5.5 Start New Application

Start the new QuartzDesk web application by following the steps outlined in 4.10.

Check the version number of the deployed QuartzDesk web application to make sure the application has been successfully upgraded. For details on how to find out the version number of a deployed QuartzDesk web application, please refer to our FAQs at www.quartzdesk.com (click Support → FAQs and search for “find out version”).

Copyright 2013-2017 QuartzDesk.com

6. 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.

6.1 HTTP Session Replication and Affinity

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 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.

6.2 Shared Work Directory

We recommend that you put the QuartzDesk work directory, described in chapter 4.6, 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 requited 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.

6.3 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:

Copyright 2013-2017 QuartzDesk.com

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

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

6.3.1 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.log</file>
  <append>true</append>
  <prudent>true</prudent>
  ...
</appender>


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

Copyright 2013-2017 QuartzDesk.com

6.3.2 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>${logs.dir}/quartzdesk-${cluster.member.instanceId}.log</file>
  <append>true</append>

  ...

  <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <!-- daily rollover -->
    <fileNamePattern>${logs.dir}/quartzdesk-
${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>${logs.dir}/quartzdesk-${cluster.member.instanceId}-trace.log</file>
  <append>true</append>

  ...

  <rollingPolicy
class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
    <fileNamePattern>${logs.dir}/quartzdesk-${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.

Copyright 2013-2017 QuartzDesk.com

```
...

<!--
  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"/>

...
```

6.4 Internal Quartz Scheduler

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.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
`org.quartz.simpl.HostnameInstanceIdGenerator`	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. This is the default implementation used by QuartzDesk. No QuartzDesk configuration changes are necessary to use this instance ID generator.
`org.quartz.simpl.SystemPropertyInstanceIdGenerator`	This implementation is suitable for QuartzDesk web application deployments where some of the clustered QuartzDesk web application instances run on the same host. This implementation extracts the Quartz scheduler instance ID from the `org.quartz.scheduler.instanceId` JVM system property that must be explicitly set. Please refer to the WebSphere documentation for details on how to add a new JVM system property.

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.

Copyright 2013-2017 QuartzDesk.com