

Extras/Working with Oracle Database

xtra Tutorial: Working with Oracle Database

READY

This tutorial was built for BDCS-CE version 17.3.3-20 as part of the New Data Lake User Journey: here (<https://github.com/oracle/learning-library/tree/master/workshops/journey2-new-data-lake>). Questions and feedback about the tutorial: david.bayard@oracle.com (<mailto:david.bayard@oracle.com>)

Contents

- Identifying your Oracle Database Cloud Service connection details
- Setting up an Access Rule for DBCS to allow BDCS-CE to connect
- Using the Zeppelin JDBC Interpreter to query the Oracle Database
- Examples with the JDBC Interpreter
- Using Spark to query the Oracle Database
- Examples with Spark SQL
- Using Spark to write to the Oracle Database
- Examples writing to the Oracle Database

Identifying your Oracle Database Cloud Service Connection Details

READY

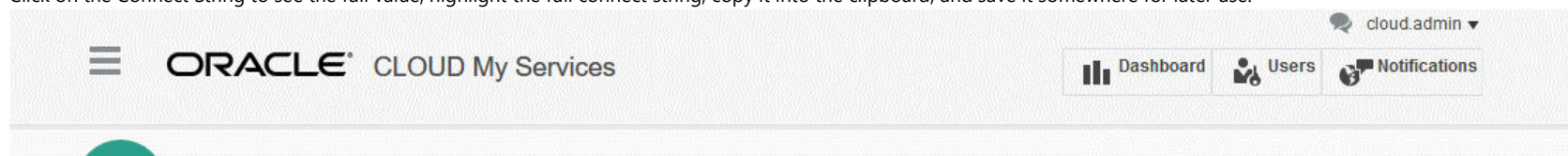
This tutorial will assume you are using an Oracle Database running in the Oracle Database Cloud Service. To connect from BDCS-CE, we need to:


- Identify the database connect string (which embeds the database hostname and service name)
- Ensure that an Access Rule allows network traffic from BDCS-CE to the Database server


In general, a very useful resource will be the “Oracle Database Cloud - Database as a Service Quick Start” which can be found here (http://www.oracle.com/webfolder/technetwork/tutorials/obe/cloud/dbaas/obe_dbaas_QS/oracle_database_cloud_service_dbaas_quick_start.html). In particular, the topic “Finding the Connection Details for your Database Instance” provides the necessary details. For simplicity, we will repeat the steps here:

Follow these steps:

- Navigate to the Oracle Database Cloud Service page for your DBCS instance.
- Click on the Connect String to see the full value, highlight the full connect string, copy it into the clipboard, and save it somewhere for later use.




Oracle Database Cloud Service / CD-DBCS-ORCL12



Overview

1
Node

Administration

1
Patches available

0
Snapshots available

Service Overview

As of Jun 30, 2017 7:37:01 PM UTC

1
Nodes

2
OCPUs

15
GB
Memory

185
GB
Storage

Status: Ready

Version: 12.1.0.2

Connect String: CD-DBCS-ORCL12:1521/PDB1....

Edition: Enterprise Edition - High Performance

Backup Destination: Both Cloud Storage and...

Cloud Storage Container: https://gse00010212.stora...

PDB Name: PDB1

Container Name: ORCL12

Show more...

Resources

Host Name: CD-DBCS-ORCL12

Public IP: 141.144.29.38

SID: ORCL12

OCPUs: 2

Memory: 15 GB

Storage: 185 GB

Shell command to identify the private IP address of your BDCS-CE instance (used by next paragraph)

READY

```
%sh
ifconfig eth0
```

Setting up an Access Rule for DBCS to allow BDCS-CE to connect

READY

We need to ensure that your BDCS-CE instance can communicate with the database listener (port 1521) in your DBCS instance. If you defined an Association between your BDCS-CE instance and your DBCS database when you created your BDCS-CE instance, then this access rule was created for you.

Most likely, you probably did NOT define an association between your DBCS database and your BDCS-CE instance at the time you provisioned the BDCS-CE instance. Therefore, you will probably need to manually define a new access rule. For this manually defined rule, you will need to use the private IP address of your BDCS-CE instance, which can be looked up via running the shell paragraph above.

For reference, review the “Oracle Database Cloud - Database as a Service Quick Start” which can be found here (http://www.oracle.com/webfolder/technetwork/tutorials/obe/cloud/dbaas/obe_dbaas_QS/oracle_database_cloud_service_dbaas_quick_start.html). In particular, the topic “Enabling Secure Network Access to your Database Instance” provides the necessary details.

Here is an animation showing you how to:

- Identify the private IP address of your BDCS-CE instance
- Navigate to the DBCS instance console
- Add a new Access Rule to the DBCS instance

ORACLE[®] Big Data Cloud - Compute Edition Console

bdcscs_admin

BDCSCE-lab

Overview

Jobs

Notebook

Data Stores

Settings

Host Name: CD-DBCS-ORCL12

Public IP: 141.144.29.38

SID: ORCL12

OCPUs: 2

Memory: 15 GB

Storage: 185 GB

Resources

CD-DBCS-ORCL12

CD-DBCS-ORCL12

Task 1

Shell command to identify the private IP address of your BDCS-CE instance (used by next paragraph)

FINISHED

Task 2

Setting up an Access Rule for DBCS to allow BDCS-CE to connect

FINISHED

Task 3

Script to set the Zeppelin JDBC settings for Oracle

READY

%sh

ifconfig eth0

Task 1 seconds. Last updated by anonymous at time Jul 1, 2017 9:28:42 AM. (outdated)

Setting up an Access Rule for DBCS to allow BDCS-CE to connect

We need to ensure that your BDCS-CE instance can communicate with the database listener (port 1521) in your DBCS instance. If you defined an Association between your BDCS-CE instance and your DBCS database when you created your BDCS-CE instance, then this access rule was created for you.

Most likely, you probably did NOT define an association between your DBCS database and your BDCS-CE instance at the time you provisioned the BDCS-CE instance. Therefore, you will probably need to manually define a new access rule. For this manually defined rule, you will need to use the private IP address of your BDCS-CE instance, which can be looked up via running the shell paragraph above.

For reference, review the “Oracle Database Cloud - Database as a Service Quick Start” which can be found here: http://www.oracle.com/webfolder/technetwork/tutorials/obe/cloud/dbaas/obe_dbaas_QS/oracle_database_cloud_service_dbaas_quick_start.html. In particular, the topic “Enabling Secure Network Access to your Database Instance” provides the necessary details.

Here is an animation showing you how to:

- Identify the private IP address of your BDCS-CE instance
- Navigate to the DBCS instance console
- Add a new Access Rule to the DBCS instance

Task 2 seconds. Last updated by anonymous at time Jul 1, 2017 9:28:42 AM. (outdated)

Script to set the Zeppelin JDBC settings for Oracle

%sh

Task 3 seconds. Last updated by anonymous at time Jul 1, 2017 9:28:42 AM. (outdated)

Example of JDBC(oracle)

READY

```
%jdbc(oracle)
select table_name from user_tables
```

JDBC Interpreter in action

READY

```
%jdbc(oracle)
select * from emp
```

Another example

READY

```
%jdbc(oracle)
select deptno, count(*) from emp group by deptno
```

Using Spark to query the Oracle Database

READY

In this section of the tutorial, we will show you working code examples that define a Spark Dataframe as an Oracle SQL query. We then define a Spark SQL temporary table against that Dataframe to show you how you can further use Spark SQL to filter and manipulate your selected data.

To run the spark code, you should **first edit the code and insert your specific database connect string, username, and password.**

To learn more about how Spark Data Frames work with JDBC data sources, check out here (<https://spark.apache.org/docs/2.1.0/sql-programming-guide.html#jdbc-to-other-databases>).

Spark code to query the Oracle Database and define results and Spark SQL tables

READY

```
%spark

// BEFORE RUNNING THIS, YOU WILL NEED TO EDIT THIS
// 1.Insert your database Connect String
// 2.Insert your database user name
// 3.Insert your database password

//define URL for the Oracle JDBC driver
println(">>>>>>Defining url for Oracle JDBC")

val url="jdbc:oracle:thin:@/" + "CD-DBCS-ORCL12:1521/PDB1.gse00010212.oraclecloud.internal"

//define the username and password as properties
println(">>>>>>Defining Oracle JDBC username and password")
val prop = new java.util.Properties

prop.setProperty("user","scott")
prop.setProperty("password","tiger")

prop.setProperty("driver","oracle.jdbc.OracleDriver") //the driver is needed to be defined with Spark 1.6.1 due to https://issues.apache.org/jira/browse/SPARK-14204

//now you can use JDBC commands like: val movies = sqlContext.read.jdbc(url,"movie",prop)
val emp = sqlContext.read.jdbc(url,"emp",prop)
//emp.explain()
emp.printSchema()
emp.show()

//register the emp dataframe as a SparkSQL table
```

SparkSQL Example against our Oracle Database-based Data Frame

READY

```
%sql
select * from emp_sparksql
where deptno=10
```

Using Spark to write to the Oracle Database

READY

Now we will show a working example of writing back to the Oracle Database from Spark. If you observed above, we created a Spark dataframe called emp_query. In the following example, we will write this dataframe back to the Oracle Database as a new table called emp_query.

For this example, we will use Spark to read in some Citibike data and write that data into an Oracle table.

Spark Scala to read CSV and register as Spark SQL temporary table

READY

```
%spark

//a previous tutorial placed the csv file into /var/lib/zeppelin/bikes/201612-citibike-tripdata.csv

val df = sqlContext.read.format("com.databricks.spark.csv").option("header", "true").load("file:/var/lib/zeppelin/citibike/201612-citibike-tripdata.csv")

//cache the data frame for performance
df.cache()

println("Here is the schema detected from the CSV")
df.printSchema()
println("..<")

println("# of rows: %s".format(
  df.count()
))
println("..<")

df.registerTempTable("bike_trips_csvtemp")
println("done")
```

Spark to write a DataFrame to an Oracle table

READY

```
%spark
```



<pre>%jdbc(orcl) select * from CITIBIKE_ORCL</pre>	READY
--	-------

	READY
--	-------