```
#############################################################################
# Copyright 2017 International Business Machines Corporation and
# others. All Rights Reserved.
#############################################################################
```

How to connect to an ORACLE database via JDBC toolkit

This document describes a step by step instruction to connect to an OARACLE database via JDBC toolkit and get the SQL code and SQL message in case of any error.

1 - Download the ORACLE jdbc driver (ojdbc7.jar) from:

http://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html

2 - Create a database user in your ORACLE database

login as oracle and start the sqlplus

create a database user

create a test table

insert some data into table

check the oracle service name. you need this name for jadbUrl parameter in SPL file.

```
sqlplus / as sysdba
SQL> connect system/manager as sysdba
SQL> alter session set "_ORACLE_SCRIPT"=true;
SQL> create user streams identified by streamspw;
User created.
SQL> grant dba to stream;
Grant succeeded.

SQL> connect streams/streamspw;
Connected.

SQL> create table test (name varchar(30), id int);
Table created.

SQL> insert into test values ('jim', 1);
1 row created.

SQL> insert into test values ('kati' , 2);
1 row created.

SQL> select * from test;
NAME                                    ID
------------------------------ ----------
jim                                      1
kati                                     2
```

```
SQL> show parameter service_name
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
service_names                        string      orcl.fyre.ibm.com
```

3- Create a SPL project in your Streams server

JDBCOracle/Makefile

JDBCOracle/opt/ojdbc7.jar

JDBCOracle/application/JDBCOracle.spl

```
/*#######################################################################
# Copyright (C)2014, 2017 International Business Machines Corporation and
# others. All Rights Reserved.
#######################################################################

// *************************************************************************
// The sample SPL application JDBCOracle demonstrates how to connect to an ORACLE database
// and select data from a table using JDBCRun operator.
// It demonstrates also how to get the SQL message in case of any error.
//
// Required Streams Version = 4.1.x.x
// Required JDBC Toolkit Version = 1.2.2
// https://github.com/IBMStreams/streamsx.jdbc/releases/tag/v1.2.2
// ORACLE jdbc driver version 7 or higher (ojdbc7.jar)
// http://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-195834
//
// To connect to database, the following parameters need to be specified:
// jdbcDriverLib   : the jdbc driver libraries (download the jdbc driver file from oracle si
// and store it in the opt folder, e.g. opt/ ojdbc7.jar)
// jdbcClassName   : the class name for ORACLE jdbc driver (oracle.jdbc.driver.OracleDriver)
// jdbcUrl         : the database URL. (e.g. jdbc:oracle:thin:@<your database server>:1521/<
// dbcUser         :  the database user on whose behalf the connection is being made.
// jdbcPassword    : the user's password.
// sqlStatusAttr   : "error" ;
// isolationLevel   : "READ_COMMITTED" ;
// set the isolationLevel to "READ_COMMITTED for ORACLE database
// In the SPL sample:
// "select" operator demonstrates how to run SQL statement from stream attribute via stateme
// In this sample the JDBCRun operator connect to the database and read all rows from test t
// write them into data/output.csv
// The second output port "error" provide SQL code SQL Status and SQL message in case  of an
// *************************************************************************/


namespace application ;

use com.ibm.streamsx.jdbc::* ;
use com.ibm.streamsx.jdbc.types::* ;
/*************************************************************************
 * JDBCRunErrorPort demonstrates how to Error Port with JDBCRun operator.
 *************************************************************************/
composite JDBCOracle
{
param
  expression<rstring> $jdbcDriverLib : "opt/ojdbc7.jar" ;
  expression<rstring> $jdbcClassName : "oracle.jdbc.driver.OracleDriver" ;
  expression<rstring> $jdbcUrl : "jdbc:oracle:thin:@skipsof1.fyre.ibm.com:1521/orcl.fyre.ibm.c
  expression<rstring> $jdbcUser : "streams" ;
  expression<rstring> $jdbcPassword : "streamspw" ;
```

```
type
insertSchema = int32 ID, rstring NAME ;
rsSchema = int32 ID, rstring NAME ;
selectSchema = rstring sql ;
graph
stream<insertSchema> pulse = Beacon()
{
param
iterations : 1000u ;
initDelay : 5.0 ;
}

(stream<rsSchema> runSql ; stream<tuple<insertSchema> inTuple, JdbcSqlStatus_T error> errors
JDBCRun(pulse)
{
logic
state :
{
mutable int32 count = 0  ;
}

//mutable int32 n=0
onTuple pulse : printStringLn((rstring) count++) ;

param
jdbcDriverLib  : $jdbcDriverLib ;
jdbcClassName  : $jdbcClassName ;
jdbcUrl        : $jdbcUrl ;
jdbcUser       : $jdbcUser ;
jdbcPassword   : $jdbcPassword ;
//statement      : "SELECT * FROM TEST" ;
statement      : "SELECT * FROM TEST2" ;
sqlStatusAttr  : "error" ;
isolationLevel : "READ_COMMITTED" ;
}

() as errorprint = Custom(errors)
{
logic
onTuple errors : printStringLn("sqlCode: " +(rstring) error.sqlCode + ", sqlState: " +
error.sqlState + ", sqlMessage: " + error.sqlMessage) ;
}

() as runSqlprint = FileSink(runSql)
{
logic
onTuple runSql : printStringLn((rstring) ID + "," + NAME) ;
param
file : "output.csv" ;
}

}
```

4 - Make the SPL application

create a Makefile and run make

```
#################################################################
# Copyright (C)2014, 2017 International Business Machines Corporation and
# others. All Rights Reserved.
#################################################################
```

```
.PHONY: all clean

#SPLC_FLAGS = -t $(STREAMS_INSTALL)/toolkits/com.ibm.streamsx.jdbc  --data-directory data
SPLC_FLAGS = -t ../streamsx.jdbc/com.ibm.streamsx.jdbc  --data-directory data

SPLC = $(STREAMS_INSTALL)/bin/sc

SPL_CMD_ARGS ?=
SPL_COMP1NAME=JDBCOracle
SPL_MAIN_COMPOSITE1 = application::$(SPL_COMP1NAME)
BUILD_OUTPUT_DIR = output

all: data clean
$(SPLC) $(SPLC_FLAGS) -M  $(SPL_MAIN_COMPOSITE1) --output-dir ./$(BUILD_OUTPUT_DIR)  $(SPL_CM

data:
mkdir data
clean:
$(SPLC) $(SPLC_FLAGS) -C -M $(SPL_MAIN_COMPOSITE1) --output-dir output
-rm -rf toolkit.xml
-rm -rf data/output.csv
```

5 - Run the SPL application

Change the database credentials in SPL file with your database credentials and run

```
$> make
```

Start the application with

```
$> output/bin/standalone
```

6 - check the SQL message

Change the statement in SPL file to

```
statement : "SELECT * FROM TEST2" ;
```

The table TEST2 doesn't exist in the database

```
$> make
$> output/bin/standalone
```

The JDCBRun operator delivers the following SQL code and SQL message, because the table TEST2 does not exist.

```
sqlCode: 942, sqlState: 42000, sqlMessage: ORA-00942: table or view does not exist
Error : 942, Position : 14, Sql = SELECT * FROM TEST2, OriginalSql = SELECT * FROM TEST2, Err
```