```
// *********************************************************************************
// * Copyright (C) 2018 International Business Machines Corporation
// * All Rights Reserved
// *********************************************************************************
```

How to connect to a DB2 database on IBM Cloud via JDBC toolkit

This document describes a step by step instruction to connect to a DB2 database on IBM cloud

via **JDBC** toolkit.

1 - Download the DB2 jdbc driver (db2jcc4.jar) from:

http://www-01.ibm.com/support/docview.wss?uid=swg21363866

and copy it in **opt** directory in your project workspace:

```
JDBCDb2Cloud/opt/db2jcc4.jar
JDBCDb2Cloud/Makefile
JDBCDb2Cloud/application/JDBCDb2Cloud.spl
```

2 - Create a DB2 service on IBM Cloud and create a service credential

Create a db2 service on IBM cloud.

https://console.bluemix.net/catalog/?search=db2

Create a service credential for DB2 service on IBM cloud.

The following sample shows a DB2 service credentials:

```
{
   "port": 50000,
    "db": "BLUDB",
    "username": "dash15202",
    "ssljdbcurl": "jdbc:db2://dashdb-entry-yp-dal9-08.services.dal.bluemix.net:50001/BLUDB:ss
    "host": "dashdb-entry-yp-dal9-08.services.dal.bluemix.net",
    "https_url": "https://dashdb-entry-yp-dal9-08.services.dal.bluemix.net:8443",
    "dsn": "DATABASE=BLUDB;HOSTNAME=dashdb-entry-yp-dal9-08.services.dal.bluemix.net;PORT=500
    "hostname": "dashdb-entry-yp-dal9-08.services.dal.bluemix.net",
    "jdbcurl": "jdbc:db2://dashdb-entry-yp-dal9-08.services.dal.bluemix.net:50000/BLUDB",
    "ssldsn": "DATABASE=BLUDB;HOSTNAME=dashdb-entry-yp-dal9-08.services.dal.bluemix.net;PORT=
    "uri": "db2://dash15202:gvQ0O8_u_CcQ@dashdb-entry-yp-dal9-08.services.dal.bluemix.net:500
    "password": "your-cloud-db2-password"
}
```

The jdbcurl in this sample DB2 service credentials is:

```
"jdbcurl": "jdbc:db2://dashdb-entry-yp-dal9-08.services.dal.bluemix.net:50000/BLUDB",
```

Copy "jdbcurl" "username" and "password" from IBM Cloud DB2 service credential and put it in the following spl file

```
// *****************************************************************************
// * Copyright (C) 2018 International Business Machines Corporation
// * All Rights Reserved
// *****************************************************************************
namespace application;
use com.ibm.streamsx.jdbc::* ;
// *****************************************************************************
// JDBCDb2Cloud.spl demonstrates how to
// create a table
// insert data into table
// select data from table
// and drop table
// via JDBCRun operator.
//
// To connect to database, the following parameters need to be specified:
// jdbcDriverLib : the jdbc driver library (download the jdbc driver and store it in opt fold
// e.g.   opt/db2jcc4.jar)
// jdbcClassName : the class name for jdbc driver (e.g. com.ibm.db2.jcc.DB2Driver)
// jdbcUrl       : the database URL. (e.g. jdbc:db2://<server:port>/<database>)
// jdbcUser      : the database user on whose behalf the connection is being made.
// jdbcPassword  : the database user's password.
//
// In the SPL sample:
// create Creates a table on databce
// insert operator demonstrates how to run SQL statement with parameter markers via statement
// select operator demonstrates how to run SQL statement from stream

// *****************************************************************************
composite JDBCDb2Cloud
{
param
expression<rstring> $jdbcDriverLib : getSubmissionTimeValue("jdbcDriverLib", "opt/db2jcc4.jar
expression<rstring> $jdbcClassName : getSubmissionTimeValue("jdbcClassName", "com.ibm.db2.jcc
expression<rstring> $jdbcUrl       : getSubmissionTimeValue("jdbcUrl","jdbc:db2://dashdb-entr
expression<rstring> $jdbcUser      : getSubmissionTimeValue("jdbcUser","dash15202") ;
expression<rstring> $jdbcPassword  : getSubmissionTimeValue("jdbcPassword","your-cloud-db2-pa

type
insertSchema = int32 ID, rstring FNAME, rstring LNAME, int32 AGE,
rstring GENDER, float32 SCORE, float64 TOTAL ;
rsSchema = int32 ID, rstring FNAME, rstring LNAME, int32 AGE, rstring GENDER,
float32 SCORE, float64 TOTAL ;
selectSchema = rstring sql ;
graph
stream<insertSchema> pulse = Beacon()
{
param
iterations : 1u ;
output
pulse : ID = 1, FNAME = "Mike", LNAME = "Ward", AGE = 31, GENDER = "M",
SCORE = 33.3w, TOTAL = 912.3l ;
}

stream<insertSchema> create = JDBCRun(pulse)
{
param
jdbcDriverLib : $jdbcDriverLib ;
jdbcClassName : $jdbcClassName ;
jdbcUrl : $jdbcUrl ;
jdbcUser : $jdbcUser ;
jdbcPassword : $jdbcPassword;
statement : "CREATE TABLE JDBCRUN_SAMPLE (ID INTEGER NOT NULL, FNAME CHAR(10), LNAME CHAR(10)

}

stream<insertSchema> insert = JDBCRun(create)
{
param
jdbcDriverLib: $jdbcDriverLib ;
jdbcClassName: $jdbcClassName ;
jdbcUrl: $jdbcUrl ;
jdbcUser: $jdbcUser ;
jdbcPassword: $jdbcPassword;
statement: "INSERT INTO JDBCRUN_SAMPLE (ID, FNAME, LNAME, AGE, GENDER, SCORE, TOTAL)
VALUES (?, ?, ?, ?, ?, ?, ?)" ;
statementParamAttrs : "ID, FNAME, LNAME, AGE, GENDER, SCORE, TOTAL" ;
}

stream<selectSchema> genSelect = Functor(insert)
{
output
genSelect : sql =
"SELECT ID, FNAME, LNAME, AGE, GENDER, SCORE, TOTAL FROM JDBCRUN_SAMPLE" ;
}

stream<rsSchema> select = JDBCRun(genSelect)
{
param
jdbcDriverLib: $jdbcDriverLib ;
jdbcClassName: $jdbcClassName ;
jdbcUrl: $jdbcUrl ;
jdbcUser: $jdbcUser ;
jdbcPassword: $jdbcPassword;
statementAttr: sql ;
}

() as printer = Custom(select)
{
logic
onTuple select :
printStringLn((rstring) ID + "," + FNAME + "," + LNAME + "," +
(rstring) AGE + "," + GENDER + "," +(rstring) SCORE + "," +(rstring)TOTAL) ;
}

/**
* delay 20 secounds to check the database table
*/
            stream<rsSchema> delayed = Delay(select)
            {
param delay : 8.0;
}


stream<rsSchema> drop = JDBCRun(delayed)
{
param
jdbcDriverLib : $jdbcDriverLib ;
jdbcClassName : $jdbcClassName ;
jdbcUrl : $jdbcUrl ;
jdbcUser : $jdbcUser ;
jdbcPassword : $jdbcPassword;
statement : "DROP TABLE JDBCRUN_SAMPLE" ;
}

}
```

4 - Make the SPL application

Create a Makefile and run make

```
######################################################################
# Copyright (C)2014, 2018 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 = $(STREAMS_INSTALL)/bin/sc
SPL_CMD_ARGS ?=
SPL_COMP1NAME=JDBCDb2Cloud
SPL_MAIN_COMPOSITE1 = application::$(SPL_COMP1NAME)
BUILD_OUTPUT_DIR = output

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

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

Be aware of tabs in Makefile

5 - Run the SPL application

Change the database credentials in SPL file with your IBM DB2 database credentials and run

```
$> make
```

Start the application with

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

Or you can submit the job on your local Streams server with:

```
$ streamtool submitjob output/application.JDBCDb2Cloud.sab
```

6 - Submit the spl application on IBM Streams Cloud

Create a Streaming Analytics on IBM Cloud

https://console.bluemix.net/catalog/?search=streams

Start the service

Lunch the application

It starts the IBM Streams console.

Now it is possible here to submit a SAB file as job

The SAB file is located in your project output directory:

```
output/application.JDBCDb2Cloud.sab
```