The following SPL sample demonstrates how to connect to a Teradata database and select data from a table using **JDBCRun** operator.

Required Streams Version = 4.1.x.x

Required JDBC Toolkit Version = 1.2.0

https://github.com/IBMStreams/streamsx.jdbc

To connect to database, the following parameters need to be specified:

jdbcDriverLib: the jdbc driver libraries.

download the jdbc driver files from: http://downloads.teradata.com/download/connectivity/jdbc-driver

and store them in opt folder, write the path and the jar file names comma separated in one string e.g. ("opt/terajdbc4.jar, opt/tdgssconfig.jar")

jdbcClassName: the class name for teradata jdbc driver (com.teradata.jdbc.TeraDriver)

jdbcUrl: the database URL. (e.g. jdbc:teradata://your-db-host/db-name)

jdbcUser: the database user on whose behalf the connection is being made.

jdbcPassword: the user's password.

In the SPL sample:

The "**select**" operator demonstrates how to run an SQL statement from stream attribute via statementAttr parameter.

In this sample the **JDBCRun** operator connect to the database and read all table names from database and write them into a text file data/output.txt

```
namespace application;
use com.ibm.streamsx.jdbc::* ;

composite Main
{
expression<rstring> $jdbcDriverLib : getSubmissionTimeValue("jdbcDriverLib", "opt/terajdbc4.j
expression<rstring> $jdbcClassName : getSubmissionTimeValue("jdbcClassName", "com.teradata.jd
expression<rstring> $jdbcUrl : getSubmissionTimeValue("jdbcUrl", "jdbc:teradata://your-db-hos
expression<rstring> $jdbcUser : getSubmissionTimeValue("jdbcUser", "your-db-userName");
expression<rstring> $jdbcPassword : getSubmissionTimeValue("jdbcPassword", "your-db-password"
expression<rstring> $statement : getSubmissionTimeValue("statement", "SELECT TableName FROM d

graph


        stream<int32 counter> pulse = Beacon() {
param
iterations : 1u ;
output
pulse : counter = 1;
}
```

```
stream<rstring TableName> select = JDBCRun(pulse) {
param
jdbcDriverLib: $jdbcDriverLib;
jdbcClassName: $jdbcClassName;
jdbcUrl: $jdbcUrl;
jdbcUser: $jdbcUser;
jdbcPassword: $jdbcPassword;
statement:  $statement;
}

() as WriteToFile = FileSink(select) {

logic
state :
{
mutable int64 counter = 0;
}

onTuple select :
{
printStringLn((rstring)counter++ + " TableName = " +(rstring) TableName) ;
}

param
      file: "output.txt";
format: line ;
flush: 1u;/*** flush the output file after 1 tuple */
}
 } 
```