Lab: Data Sources in DSXL

July 2018

Author: Parminder Kaur parminder.kaur@ibm.com

© Copyright IBM Corporation, 2017. All Rights Reserved.
All trademarks or registered trademarks mentioned herein are the property of their respective holders

Table of contents

Contents

Overview 1
What You Should Be Able to Do 1
Required software, access, and files 1
Part 1: Add dashdb built-in/packaged data source 1
Part 2: Add PostgreSQL custom data source 5
Part 3: Add HDFS and Hive HDP data sources 6

Overview

In this lab, you will learn how to create data sources and remote dataset definitions that can be used to pull data into DSXL.

DSXL v1.2 packages Big SQL, Db2, Db2 for z/OS, DB2 Warehouse on Cloud (dashDB), Hive, HDFS, Informix, Netezza and Oracle databases. However, customers can also work with other JDBC databases that are not prepackaged by using custom JDBC data source features. In this lab, you will learn how to work with a packaged data source as well as custom JDBC and HDP data sources.

This lab has been divided into three parts:

Part 1 - where you will add DB2 Warehouse (dashDB) and Big SQL built-in/packaged data sources to DSXL

Part 2 – where you will add a PostgreSQL custom data source to DSXL (*optional – requires admin rights*)

Part 3 – where you will add HDP data sources to DSXL

What You Should Be Able to Do

- Pull data from external data sources explicitly supported or packaged by DSXL
- Configure DSXL to work with custom JDBC data sources
- Share data sources and remote datasets definitions with project collaborators
- Browse and Preview HDP and Hive remote datasets
- Use insert-to-code to read/fetch data from defined remote datasets

Required software, access, and files

- To complete this lab, you will need access to a DSX Local cluster v1.2.0.2 or above with connectivity to your remote data sources.

Part 1: Add built-in/packaged data sources

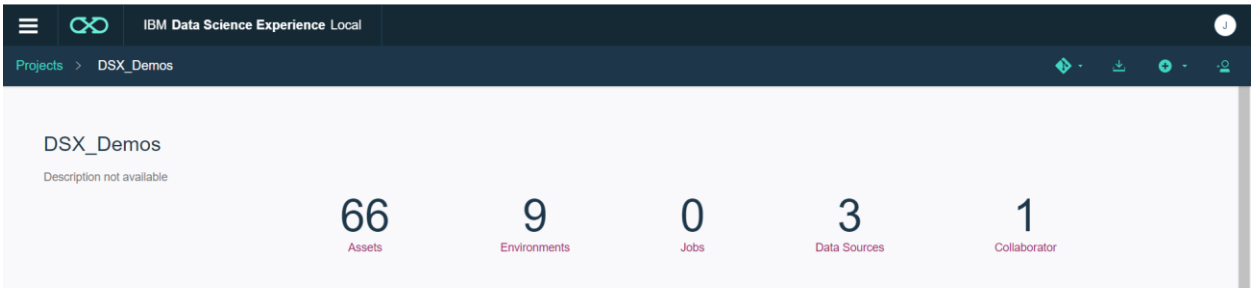
1. Create a new blank project, follow instruction in documentation https://content-dsxlocal.mybluemix.net/docs/content/local/projects.html#create-a-project.

2. Create a DB2 Warehouse and Big SQL data source and remote data set definitions by following instructions in documentation https://content-dsxlocal.mybluemix.net/docs/content/local/createdatasources.html.

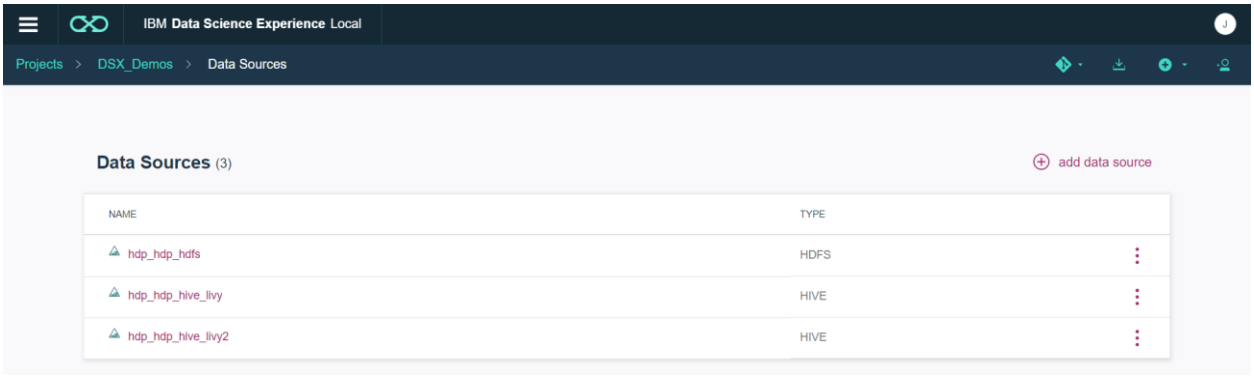
   You can use below mentioned test servers for this lab.

```
DB2 Warehouse

Data Source Type: dashDB
JDBC URL: jdbc:db2://18.206.242.141:50000/BLUDB
Username: secdemo
Password: 1BMDemo4SEC=
schema: SECDEMO
table: STOCKS

Big SQL

Data Source Type: Big SQL
JDBC URL: jdbc:db2://54.87.13.207:32051/BIGSQL
Username: secdemo
Password: 1BMDemo4SEC=
Schema: SECDEMO
Table: STOCKS
```

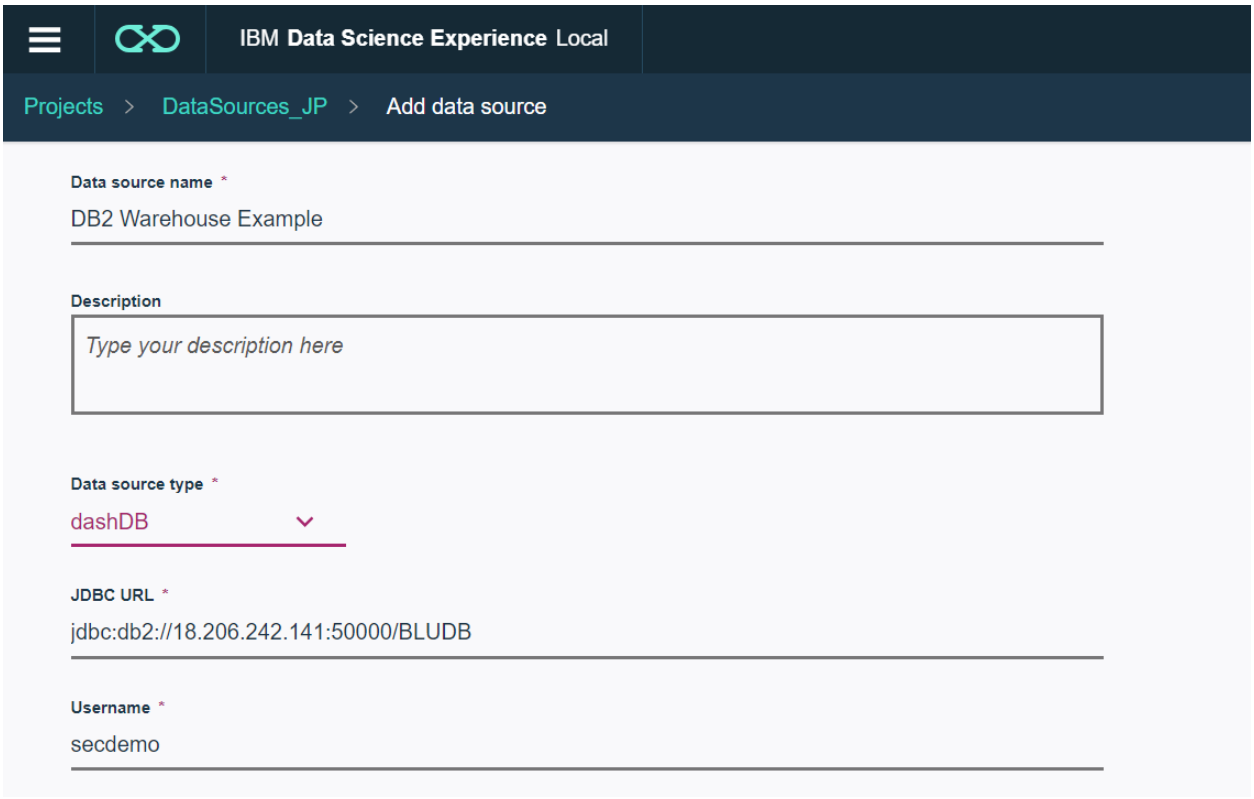
Select **Data Sources** for the project.



If you have an HDP DSXHI system configured, you may already have HDP data source configured. To add a new data source, select **add data source**



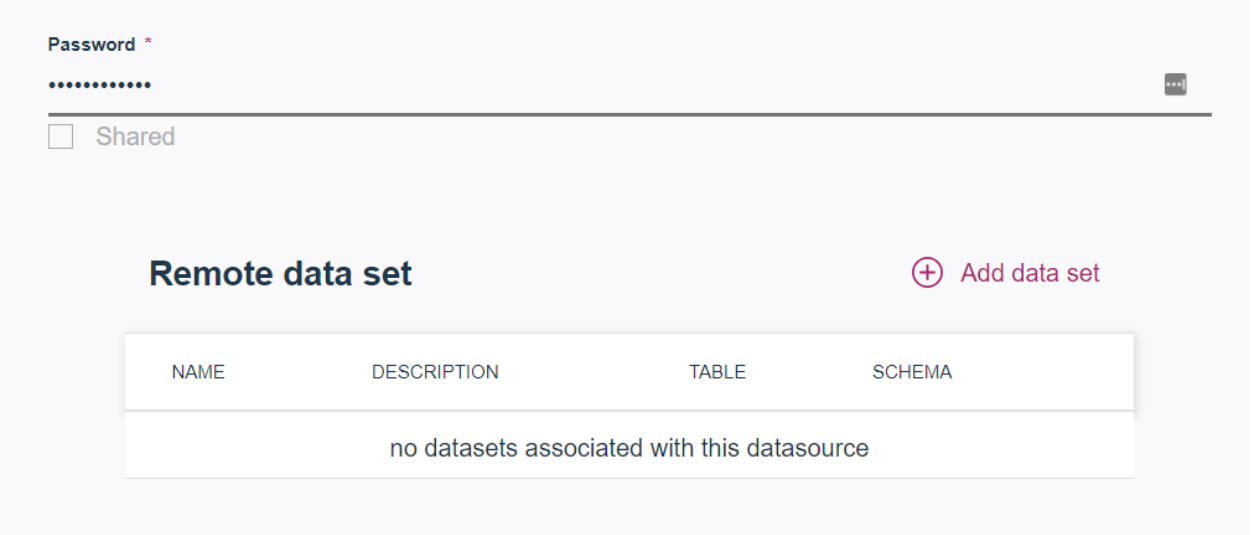
Fill in the values for each data source.



Note that, there is a **Shared** checkbox option in **Add data source** page. This option allows you to share your data source credential information with all the project collaborators. However, if you want to keep your credential private, leave it unchecked. The collaborators may see the data source definition and related dataset, but they would not see credential information and should provide a valid credential to access the data source.

3. Once you have saved the data source (or you can do this when creating it) create a data set. A Data Set is a pointer to an existing table.

   Select **Add data set**



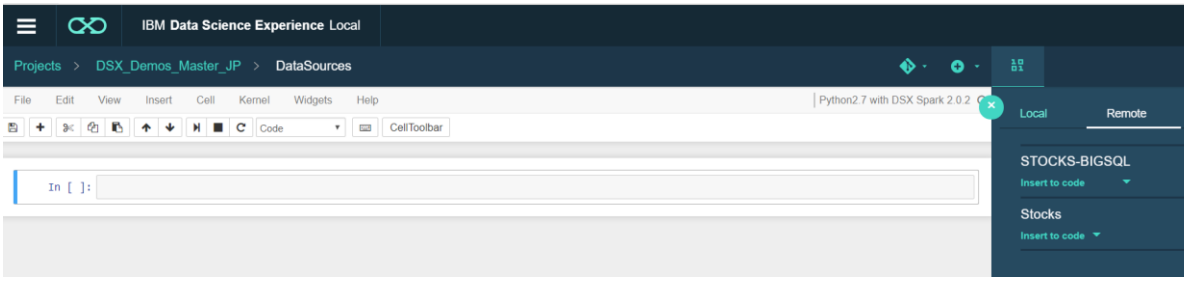
   Enter the values for the data set name, schema and table

   Save

4. Pull data from above created remote assets using insert-to-code.

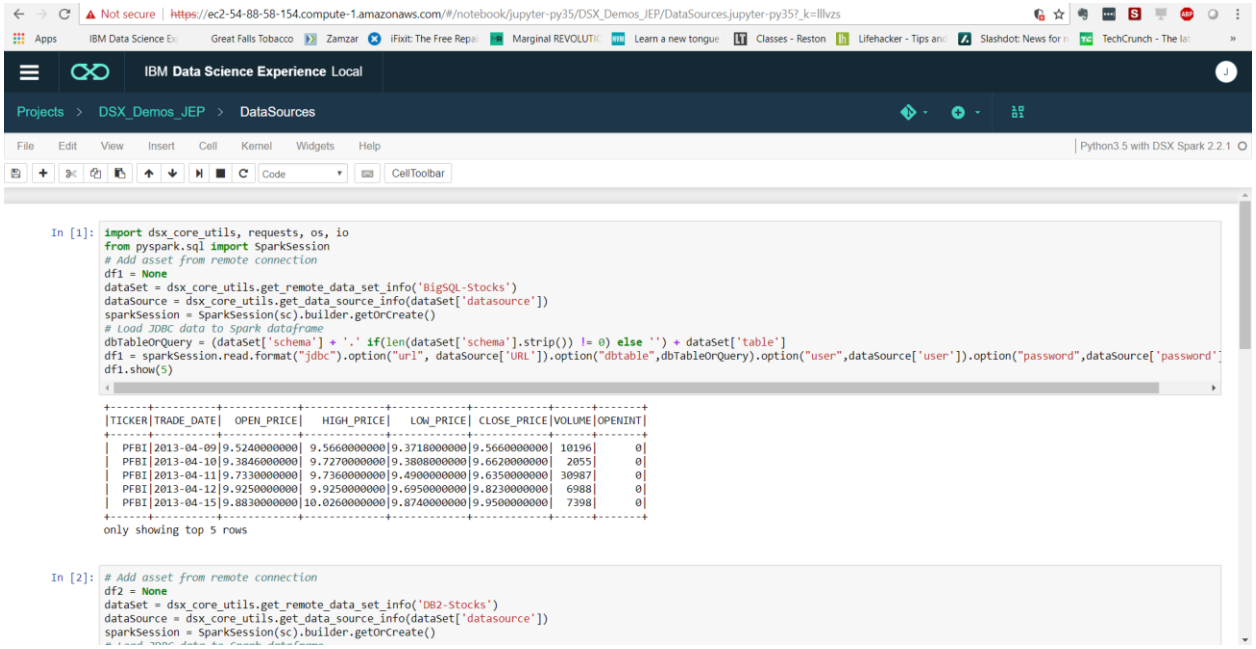
   4.1. Create a blank Jupyter notebook. Go to Notebooks -> add notebook. Select Jupyter with Python 3.5, Scala 2.11, R 3.4.3, Spark 2.2.1.

   4.2. Click on `Find data` icon  . Select `Remote`



4.3. Click Insert to code on your remote dataset. Select Insert Spark DataFrame in Python from the dropdown, this will insert a code snippet in a new Jupyter cell.

4.4. Run the cell(s) by clicking Cell -> Run All. After successful cell execution, you would see a data frame from your table as output.



4.5. As an optional, insert the HDFS Titanic data as well

Part 2: Add PostgreSQL custom data source (optional)

To work with custom JDBC, admin needs to upload a JDBC driver jar to DSXL so that DSXL users can add custom data sources and remote data sets and access data from them.

Follow instructions in documentation https://content-dsxlocal.mybluemix.net/docs/content/local/drivers.html to first import PostgresSQL driver jar, create a custom data source and a remote data set, and finally access data using `Insert Pandas DataFrame` insert-to-code option from a Jupyter with Python 2.7, Scala 2.11, R 3.4.3, Spark 2.0.2 notebook.

Part 3: Add HDFS and Hive HDP data sources (optional)

Pre-requites:

- You need to have a HDP cluster with DSXHI (DSX Hadoop Integration Service) installed on Hadoop edge node. Already covered in `Hadoop Integration` training session. Here is the documentation https://content-dsxlocal.mybluemix.net/docs/content/local/hdp.html.

- If you choose not to install DSXHI on your HDP cluster, then follow steps mentioned in documentation https://content-dsxlocal.mybluemix.net/docs/content/local/hdp.html#option-2-set-up-a-hdp-cluster-without-dsxhi. (also follow steps from 3.2)

3.1. For HDP DSXHI cluster,

1. Register DSXHI cluster to DSXL. Follow documentation https://content-dsxlocal.mybluemix.net/docs/content/local/hadoopintegration.html.

2. Once you are done with registration, the HDFS and Hive data sources will automatically get populated in your existing project or new project. You can browse HDFS and Hive data sources and preview their remote data sets.

3. Access data using `Insert Pandas DataFrame` insert-to-code option on HDFS data set from a Jupyter with Python 2.7, Scala 2.11, R 3.4.3, Spark 2.0.2 notebook. Note that insert-to-code is not supported for Hive remote data sets. Documentation on Hive https://content-dsxlocal.mybluemix.net/docs/content/local-dev/hadoop-secure_hive_data.html (beyond the scope of this lab).

3.2. For HDP non-DSXHI cluster,

1. Download topology xml file from https://ibm.box.com/s/d14hcxc8qgzur13tugwgztobpxirjq6v.

2. Replace `knox.token.verification.pem` property value with the token generated by your DSXL cluster. To retrieve token, run `curl -k https://<your-cluster-ip>/auth/jwtcert` on command line or terminal.

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
   <gateway>
      <provider>
         <role>federation</role>
         <name>JWTProvider</name>
         <enabled>true</enabled>
         <param>
            <name>knox.token.verification.pem</name>
            <value>MIIDcTCCAlmgAwIBAgIJAMbI5dCPnxnEMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMQswCQYDVQQIDAJDQTERMA8GA1UEBwwIU2FuIEpvc2UxKTAnBgNVBAoM
IEFuYWx5dGljcyBQcml2YXRlIENsb3VkIFBsYXRmb3JtMRAwDgYDVQQDDAdwZWdh
c3VzMB4XDTE4MDUzMTE4MTc0OFoXDTI4MDUyODE4MTc0OFowajELMAkGA1UEBhMC
VVMxCzAJBgNVBAgMAkNBMREwDwYDVQQHDAhTYW4gSm9zZTEpMCcGA1UECgwgQW5h
bHl0aWNzIFByaXZhdGUgQ2xvdWQgUGxhdGZvcm0xEDAOBgNVBAMMB3BlZ2FzdXMw
ggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDlPaLSNDJlQ7rjer2+Snab
6RDaEL8h8Q8UBdcY2plmSvbMEIAjT5x316SIRynhykc73/IfrEzt25w8k5lw5tNQ
hZo9+OK+DJGBIz9+1JJETYYVy402b6tnVuW3x0MRPM5RKGkLxoliaGOCsch2oj2o
214Ns/T1+c+Ma9fIv5Zana3C9BBnouz5c4v1wo5yHkMToelMEOrtgNzygf7Htpbj
kCisxYTlsAyjmUFVh6lQZKUlmPzKjgGLhe1DJhY72N5DUsHJBrPosNSfvNZQjV7P
aGHaxS28xXjO5S7PaiKQ0IdJ+KfRyoqcZdDjM/O/UTfmod5mIqWmCq754s27Z67x
AgMBAAGjGjAYMAkGA1UdEwQCMAAwCwYDVR0PBAQDAgXgMA0GCSqGSIb3DQEBBQUA
A4IBAQDdmuwB1Din2t0rTmKpExJNdM+bTY+8hEQOQKO0TcTUhT2vEdINX02d/0+H
vRUZbmx1Ctpo2vjGYscXiqyv0FDBhrQULdQI+hi4R2Icl1fXiip9LQ8T/XmCfoUE
ZFEykhiGCNVFXQZgxhWbhRbowe7/zMYtRlFm5gcj7xK61pUn0ckaKz1Rw8+hgh91
wDCI51PIelePD19ruW7YQ+VD89ntcG6kMIOenxX70u4P2Z/XUULS11uKX3ohbSWv
9CKACELM6145AKCFWng6+8Y/jrtp33a1yKx4O++EdDg7b81h8vecbeiUbL2Ywlbp
rF3837HTmvQ6g+x6jvMk7ZZyjfIF</value>
         </param>
      </provider>
```

   Give your xml file a unique name so that it does not conflict with others lab work.

3. Copy your topology xml file to HDP cluster. For this lab, use following cluster: (Note this cluster would expire on Jun14)

```
prmndr-hdp1-fyre.ibm.com
password: Temp4lab!

// ***use below command for copy***
scp ./dsx-yourname.xml root@prmndr-hdp1.fyre.ibm.com:/usr/hdp/current/knox-
server/conf/topologies
```

4. Once you have configured your HDP cluster by adding a new topology for DSX, create HDFS and Hive data sources and remote data sets manually.

```
Data source type: HDFS - HDP
```

```
HDFS host: prmndr-hdp1-fyre.ibm.com
HDFS port: 8020
WebHDFS URL: https://prmndr-hdp1.fyre.ibm.com:8443/gateway/<dsx-yourname>/webhdfs/v1
```

```
Data source type: Hive - HDP
WebHCat URL: https://prmndr-hdp1.fyre.ibm.com:8443/gateway/<dsx-
yourname>/templeton/v1
WebHDFS URL: https://prmndr-hdp1.fyre.ibm.com:8443/gateway/<dsx-yourname>/webhdfs/v1
Livy URL: https://prmndr-hdp1.fyre.ibm.com:8443/gateway/<dsx-yourname>/livy2/v1
```

5. You can browse and preview on HDFS and Hive assets. Also, you can access data from insert-to-code. Note, insert-to-code on Hive remote data set is not yet supported.