# IBM Fluid Query for PureData Analytics

- Sanjit Chakraborty

# IBM Fluid Query for PureData Analytics

## Introduction

IBM Fluid Query is the capability that unifies data access across the Logical Data Warehouse. Users and analytic applications need access to data in a variety of data repositories and platforms without concern for the data's location or access method or the need to rewrite a query. IBM Fluid Query is the capability for a data store to route a query to the correct data store within the logical data warehouse so that the query can flow to the data, not the data flow to the query.

No matter where a user connects within the logical data warehouse, they can access all data through the same, standard API/SQL access. IBM Fluid Query powers the Logical Data Warehouse by giving users the ability to combine their data even if spread across various sources in a fast, agile manner to drive analytics and deeper insight, without understanding how to connect multiple data stores, use different syntax or APIs or change their application.

IBM Fluid Query allows access to data in Hadoop or Spark from IBM PureData System for Analytics appliances. Fluid Query enables the fast movement of data between Hadoop and IBM PureData System for Analytics appliances. Enabling query and data movement, IBM Fluid Query connects those appliances to common Hadoop systems like IBM BigInsights for Apache™ Hadoop®, Cloudera and Hortonworks with the ability to access Spark through a Spark SQL connector. This provides an additional level of flexibility when accessing data residing on the Hadoop framework. With IBM Fluid Query, you can query against PureData System for Analytics, Hadoop or both by combining results from PureData System for Analytics database tables and Hadoop data sources thus creating powerful analytic combinations.

## Objectives

In the following workshop, you will be exploring following areas:

- Get necessary libraries, and client configuration files from IBM BigInsights environment for work with Fluid Query
- Configure Fluid Query to connect to BigSQL in IBM BigInsights environment
- Move data between IBM PureData System for Analytics and BigSQL
- Query data from IBM PureData System for Analytics that resides in BigSQL
- Retrieve data from BigSQL in IBM BigInsights environment to PureData System

[Netezza and BigInsights VMs must be up and running prior to start this workshop]

# Start of the workshop

## 1. Set the Fluid Query environment on PDA

1.1. Access the Netezza Virtual Machine (Netezza VM) by double click on the following desktop icon. Use user name and password as '**nz**' for login to the Netezza VM:


1.2. Change to the directory that containing the Fluid Query scripts

```
cd /nz/export/ae/products/fluidquery
```

1.3. Download the required Hadoop, Hive, BigSQL libraries, and client configuration files from the BigInsights to the Netezza system.

```
./fqDownload.sh --host bigdata --user bigsql --service bigsql
```

Each time you are prompted for a password, type **passw0rd**. When asked if you want to continue connecting, type **yes**

| | |
|---|---|
| **Tips** | The **fqDownload.sh** will fail with following error if /nz/export/ae/products/fluidquery/libs/ibm/fqdm directory not empty:<br><br>***ERROR:***<br>***The download directory /nz/export/ae/products/fluidquery/libs/ibm/fqdm is not empty***<br><br>Delete all file from the /nz/export/ae/products/fluidquery/libs/ibm/fqdm directory using following command:<br><br>`rm -f /nz/export/ae/products/fluidquery/libs/ibm/fqdm/*` |

## 2. Configuring the Data Movement Utility on PDA

2.1. Change to the directory containing the configuration file to be edited

```
cd  /nz/export/ae/products/fluidquery/conf
```

2.2. Using vi, open the file fq-remote-conf.xml

| **Notes** | If you would rather not do the configuration yourself, you can use the pre-populated version with the following command and skip to step 2.4.:<br><br>`cp fq-remote-conf.xml.completed fq-remote-conf.xml` |
|---|---|

2.3. Locate the following sections and edit them accordingly.

2.3.1 Change the location to archive the data to be under the bigsql home directory

<name>**nz.fq.output.path**</name>
<value>/**home**/**bigsql**/**nzbackup**/**sls_sales_fact_archive**</value>
<description>**Directory in HDFS where transferred data is stored**</description>

2.3.2 Specify the name of the Netezza system, where the data is stored.

<name>**nz.fq.nps.server**</name>
<value>**netezza**</value>
<description>**The wall IP address or fully qualified hostname of the Netezza server**</description>

2.3.3 Verify the credentials that will be used to connect to the Netezza system. For the purposes of this lab, the database user account is **admin** and password is **password**.

<name>**nz.fq.nps.user**</name>
<value>**admin**</value>
<description>**The Netezza database user account name for access to the database**</description>
</property>
<property>
<name>**nz.fq.nps.password**</name>
<value>**password**</value>
<description>**The password for the Netezza database user account**</description>

2.3.4 Provide the following details for the Hadoop environment:

<name>**nz.fq.sql.server**</name>

```
<value>bigdata</value>
<description>SQL server address on Hadoop where imported table will be created</description>
```

2.3.5 Provide the BigSQL port

```
<name>nz.fq.sql. port</name>
<value>32051</value>
<description>SQL server port number on Hadoop</description>
```

2.3.6 Setting the type property as BigSQL

```
<name>nz.fq.sql.type</name>
<value>bigsql</value>
<description>Supported types: hive1, hive2, bigsql. Hive2 is preferred, Hive1 should be used on old systems where Hive2 is not available.</description>
```

2.3.7 Provide connection credentials

```
<name>nz.fq.sql.user</name>
<value>bigsql</value>
<description>User name. Required if User/Password authentication should be used. </description>
<name>nz.fq.sql.password</name>
<value>bigsql</value>
<description>Password. Required if user name was provided.</description>
```

2.4. Change back to the directory containing the scripts

```
cd /nz/export/ae/products/fluidquery
```

2.5. Using the configuration file, we just edited, create a connection to the BigInsights/Hadoop environment for the Data Movement utility.

```
./fqConfigure.sh --service fqdm --provider ibm --fqdm-conf conf/fq-remote-conf.xml --database gosalesdw --config fqdm
```

This process will take few minutes. Press **Y**, if it asks for overwrite the “fqdm” configuration file.

Make sure all checks from ‘fqConfigure.sh’ command returns “OK” before you proceed to next step.

| | |
|---|---|
| **Tips** | Following is a sample output from 'fqConfigure.sh' command:<br><br>log4j:WARN No appenders could be found for logger (com.ibm.nz.fq.FqConfiguration).<br>log4j:WARN Please initialize the log4j system properly.<br>log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.<br>Database set to: gosalesdw<br>FluidQuery version: 1.7.0.0 [Build 160420-201]<br>SLF4J: Class path contains multiple SLF4J bindings.<br>SLF4J: Found binding in<br>[jar:file:/nz/export/ae/products/fluidquery/libs/ibm/fqdm/slf4j-log4j12.jar!/org/slf4j/impl/StaticLoggerBinder.class]<br>SLF4J: Found binding in<br>[jar:file:/nz/export/ae/products/fluidquery/libs/ibm/fqdm/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]<br>SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.<br>SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory] Checking access to HDFS for user: bigsql.<br>Using following url: hdfs://bigdata.ibm.com:8020.....[OK]<br>Checking connection to warehouse<br>jdbc:netezza://netezza:5480/GOSALESDW.....[OK]<br>Checking map-reduce jobs.....[OK]<br>Init destination database engine .....[OK]<br>Checking BigSQL connection.<br>jdbc:db2://bigdata.localdomain:32051/BIGSQL.....[OK]<br>Done<br>Connection configuration success. |

2.6. Register UDFs that will be used to move data between the systems.

```
./fqRegister.sh --db gosalesdw --config fqdm --udtf tohadoop,fromhadoop
```

| | |
|---|---|
| **Notes** | If the above command generates messages "**Existing function with same name already found.**" Press **Y** to register and overwrite each existing function. |

## 3. Configuring Access To BigSQL

Copy the JDBC driver that will be used to query data from BigSQL, along with the necessary connections.

3.1. Change to the directory designated to hold the bigsql JDBC driver.

```
cd /nz/export/ae/products/fluidquery/libs/ibm/bigsql
```

3.2. Copy the driver from the BigInsights/Hadoop environment. When prompted for the password enter "**passw0rd**".

```
scp bigsql@bigdata:/usr/ibmpacks/bigsql/4.1/db2/java/db2jcc.jar .
```

3.3. Change the directory to

```
cd /nz/export/ae/products/fluidquery
```

3.4. Next configure the connection to BigSQL that you will be used for queries. When prompted for your password, enter "**passw0rd**". Press **Y**, if it asks for overwrite the "pda2bigsql" configuration file.

```
./fqConfigure.sh --host bigdata --port 32051 --username bigsql --config
pda2bigsql --provider ibm --service bigsql --database gosalesdw
```

3.5. Register the UDFs that will be used to read data. When it will ask form client password use **passw0rd**.

```
./fqRegister.sh --db gosalesdw --config pda2bigsql --udtf readbigsql
```

## 4. Copying Data to BigSQL

In previous steps you configured the PDA and BigInsights Hadoop environments. Now let's copy the oldest month of data from PDA to BigInsights Hadoop environment.

4.1. In **gosalesdw** database, the earliest month of data in the **sls_sales_fact** table is from January 2004. You can use the following SQL using **nzsql** to figure it out:

```
nzsql -d gosalesdw -c 'SELECT MIN(order_day_key) FROM sls_sales_fact;'
```

4.2. To move the full year of data, specify a range of 20040101 through 20041231. You will use the **tohadoop** UDF that you registered in section 2.6.

| | |
|---|---|
| **Notes** | The parameters for the UDF are as follows:<br><br>`toHadoop('nps_db', 'nps_table', 'hive_schema', 'hive_table', 'option', …);`<br><br>You will use the following parameters :<br>nps_db – Name of the PDA database from which we are copying the data.<br>nps_table – Name of the PDA table from which we are copying the data.<br>hive_schema – Name of the schema in which the data will reside in BigInsights Hadoop.<br>hive_table – Name of the table in which the data will reside in BigInsights Hadoop. |

| | |
|---|---|
| | option – All of the other arguments needed to move data to the Hadoop cluster are taken from the edited fq-remote-conf.xml file. The file parameters can be overridden using command arguments starting in this section. In this lab you will use the **nz.fq.nps.where** as a parameter to include a WHERE clause to filter data. |

Run the following SQL from **nzsql** to move the full year of data from 2004. This step will take few minutes and once successful will return '**Fluid Query Data Movement finished successfully**'.

```
nzsql -d gosalesdw -c "CALL tohadoop('gosalesdw', 'sls_sales_fact',
'gosalesdw', 'sls_sales_fact_archive', 'nz.fq.nps.where=order_day_key
BETWEEN 20040101 and 20041231'); "
```

4.3. Now let's check the data in BigSQL. Double click on the following desktop icon to access the BigInsights Virtual Machine (BigInsights VM):


The above will open a terminal session as user **root**. Use '**su - bigsql**' to login as user **bigsql**.

4.4. Run following commands one at a time, to check the data that you copied to BigSQL:

```
db2 CONNECT TO bigsql

db2 "SELECT COUNT(*) FROM gosalesdw.sls_sales_fact_archive"

db2 "SELECT MIN(order_day_key), MAX(order_day_key) FROM
gosalesdw.sls_sales_fact_archive"
```

Following are sample outputs from the above commands:

| | |
|---|---|
| **Tips** | `$ db2 connect to bigsql`<br><br>`   Database Connection Information`<br><br>` Database server        = DB2/LINUXX8664 10.6.3`<br>` SQL authorization ID   = BIGSQL`<br>` Local database alias   = BIGSQL` |

| | |
|---|---|
| **Tips** | `$ db2 "SELECT COUNT(*) FROM gosalesdw.sls_sales_fact_archive"`<br><br>`1`<br>`-----------`<br>`84474.` |

```
  1 record(s) selected.
```

**Tips**

```
$ db2 "SELECT MIN(order_day_key), MAX(order_day_key) FROM
gosalesdw.sls_sales_fact_archive"

1           2
----------- -----------
   20040112    20041217

  1 record(s) selected.
```

## 5. Querying BigSQL/Hadoop Data from PDA

In this section you will query data that is stored in BigSQL from PDA.

5.1. Go back to the "**Netezza**" PuTTY session that you previously using.

5.2. Use the UDF registered in step 3.5, to count the rows of data in the archive table and see the data ranges, execute the following query from **nzsql**:

```
nzsql –d gosalesdw –c "SELECT * FROM TABLE WITH FINAL
(readbigsql('','','SELECT COUNT(*) FROM
gosalesdw.sls_sales_fact_archive'));"
```

The above SQL should validate the same number of rows that found in the step 4.4.

5.3. You can query both PDA and BigSQL at once. Execute this query from **nzsql**. This may take a minute or so.

```
nzsql –d gosalesdw –c "select * from
((select cast(order_day_key/10000 as int) sales_year,sum(sale_total) from
sls_sales_fact group by sales_year)
union all
(select *
from table with final (readbigsql('','','select cast(order_day_key/10000 as
int) sales_year ,sum(sale_total) from gosalesdw.sls_sales_fact_archive
group by cast(order_day_key/10000 as int)'))))zed
order by sales_year;"
```

## 6. Retrieving Data from BigSQL/Hadoop

While querying data in Hadoop from PDA is fine for occasional or ad hoc use, for cases where performance is important, you would want to temporarily move data back into PDA. For example, let's bring the 2010 sales data back into PDA for run some ad hoc report.

6.1. The 2005 data is stored a table called GOSALESDW.SLS_SALES_FACT. It has multiple years of data. But we only want to pull back the 2005 data and store it in PDA. First, at Netezza side drop the SLS_SALES_FACT_2005 table, if it exists.

```
nzsql –d gosalesdw –c "DROP TABLE sls_sales_fact_2005"
```

6.2. Retrieve the data from BigSQL and store it in PDA (This may take a minute or so):

```
nzsql –d gosalesdw –c "CREATE TABLE sls_sales_fact_2005 AS SELECT * FROM
TABLE WITH FINAL (readbigsql('','','select * from gosalesdw.sls_sales_fact
where order_day_key between 20050101 and 20051231')) DISTRIBUTE ON RANDOM;"
```

6.3. Confirm amount of data retrieved:

```
nzsql –d gosalesdw –c "SELECT COUNT(*) FROM sls_sales_fact_2005;"
```

## Workshop Complete

Thank you for your time to walk through this workshop.

Please perform following before starting a new workshop or leaving:

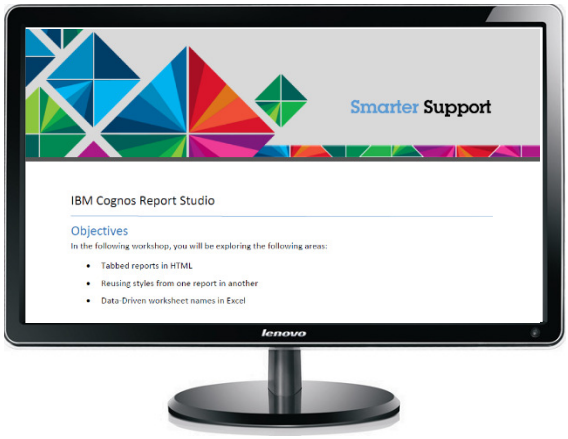
- From the "**Netezza**" terminal session, run **/nzscratch/backup/cleanup.sh**
- Close the terminal session that connect to '**Netezza**'
- Close the terminal session that connect to '**BigInsights**'
- Close this workshop

## Customer Experience

When a customer sits down to one of the lab machine, there will be a **monitor, laptop and mouse**. On one of the monitors, they will be presented with our Reset Application.

The customer will click on one of the workshop icon from the Reset Application. At this point a **PDF of the workshop** will display on one screen and an **app or a browser** screen will pop up on the other screen depending on the starting point of the workshop.


App or Browser




Workshop