Drill Documentation Project

Goethe University
Datenbank Praktikum - Big Data (WS 2015/16)

Lecturers: Prof. Dott. Ing. Roberto V. Zicari and Todor Ivanov
Owners: Harold Bartels, Lucas Berghäuser, Daniel Orbegoso

Feb. 2016, Frankfurt am Main

Purpose:

Install Drill and run TPC-H queries

Methodology:

Download and setup Drill to use Hive as storage

Load Data in HDFS: The data was generated with a tool available online called D2F-Bench. It loaded the data in HDFS directly

Choose schema and run queries

Resources:

Cloudera Virtual Machine (online available)

5GB Ram available for the VM

Apache Drill (online available)

Data generator (online available)

Installing Drill

For installing Apache Drill in the Cloudera Virtual Machine:

A) Go to the website and follow the steps of installing drill on Linux.

https://drill.apache.org/docs/installing-drill-on-linux-and-mac-os-x/

B) Type in a command line:

wget http://getdrill.org/drill/download/apache-drill-1.1.0.tar.gz

Then, create a folder where you want to save drill: for example Drill_exercise

mkdir Drill-exercise

Move the folder to the Drill-exercise folder

mv apache-drill-1.1.0.tar.gz Drill-exercise/

Go to the Drill-exercise folder

cd Drill-exercise

Then extract the contents as follows:

tar -xvzf apache-drill-1.1.0.tar.gz

Starting Drill

Start your local host:

In your browser go to Hue by typing:

http://quickstart.cloudera:8888/

Enter the user name and password, which by default are:

cloudera

Navigate to the Drill installation directory and type

cd apache-drill-1.1.0

Type the following command to start the Drill shell:

bin/drill-embedded

The 0: jdbc:drill:zk=local> prompt appears. Now, you can run queries.

Open the browser interface of Drill:

http://localhost:8047/query

Go to Storage, Hive click in Enable and paste this configuration

```
{
  "type": "hive",
  "enabled": true,
  "configProps": {
  "hive.metastore.uris": "thrift://127.0.0.1:9083",
    "hive.metastore.sasl.enabled": "false"
  }
}
```

Go to options and find “planner.enable_decimal_data_type”

Update to “True”

Download Adjustable TPC-H Data Generator for HDFS

Go to the website

https://github.com/t-ivanov/D2F-Bench

Follow steps 1 to 5

Download Data Generator and Fill the Tables with TPCH Conform Entries (Optional)

To download a data generator go to:

http://www.tpc.org/tpch/spec/tpch_2_16_0.zip

```
cd Downloads/tpch_2_16_0/tpch_2_15_0/dbgen/
```

Make a copy of the dummy makefile

```
cp makefile.suite makefile
```

In dbgen folder find the created makefile and insert highlighted values (bold) to this file.

```
...
#################
## CHANGE NAME OF ANSI COMPILER HERE
#################
CC      = gcc
# Current values for DATABASE are: INFORMIX, DB2, TDAT (Teradata)
#                          SQLSERVER, SYBASE, ORACLE, VECTORWISE
# Current values for MACHINE are:  ATT, DOS, HP, IBM, ICL, MVS,
#                          SGI, SUN, U2200, VMS, LINUX, WIN32
# Current values for WORKLOAD are:  TPCH
DATABASE= SQLSERVER
MACHINE = LINUX
WORKLOAD = TPCH
#
...
```

In dbgen folder find the **tpcd.h** file and edit higlighted (bold) values for SQLSERVER.

```
...
#ifdef  SQLSERVER
#define GEN_QUERY_PLAN  "set showplan on\nset noexec on\ngo\n"
#define START_TRAN      "BEGIN WORK;"
#define END_TRAN        "COMMIT WORK;"
#define SET_OUTPUT      ""
#define SET_ROWCOUNT    "limit %d;\n\n"
#define SET_DBASE       "use %s;\n"
#endif
...
```

Run make command.

```
make
```

Generate the files for population. (The last numeric parameter determines the volume of data with which will be your database then populated -
I decided that 0.1 (=100MB) is fine for my purposes, since I am not interested in the database benchmark tests

```
$ ./dbgen -s 0.1
```

Indicate Drill to Use Hive to Locate Rows by Changing the Default Schema

Start Drill in embedded mode with

```
bin/drill-embedded
```

Then define the schema that contains the tables which will be used. The schema is in user/hive/warehouse, but it is not necessary to give the complete path:

```
use hive.tpch_orc_2sf;
```

Open the shell to run queries

Go to the folder where the queries are:

```
cd Drill-exercise/tpch_2_16_0/dbgen/queries/
```

Run drill embedded:

```
~/Drill-exercise/apache-drill-1.1.0/bin/drill-embedded
```

Select the schema to use:

```
0: jdbc:drill:zk=local> show schemas;
```

```
0: jdbc:drill:zk=local> use hive.tpch_orc_2sf;
```

Run queries:

```
0: jdbc:drill:zk=local> !run 1.sql
```

Times:

Pc 1

Q1: 4 rows selected (93.257 seconds)
Q1 Second time :4 rows selected (59.575 seconds)
Q5: 5 rows selected (41.194 seconds)
Q12: 2 rows selected (43.747 seconds)
Q13: 41 rows selected (35.381 seconds)
Q14: 1 row selected (26.781 seconds)

Pc 2

Q1.output:4 rows selected (34.337 seconds)
Q5.output:5 rows selected (20.752 seconds)
Q6.output:1 row selected (12.991 seconds)
Q10.output:20 rows selected (22.503 seconds)
Q12.output:2 rows selected (24.755 seconds)
Q13.output:41 rows selected (16.502 seconds)
Q14.output:1 row selected (13.35 seconds)

Pc 1
HIVE
Q1: Time taken: 245.202 seconds, Fetched: 4 row(s)
Q5: 2Time taken: 698.052 seconds, Fetched: 5 row(s)
Q12: Time taken: 476.499 seconds, Fetched: 2 row(s)
Q13: Time taken: 603.991 seconds, Fetched: 41 row(s)
Q14: Time taken: 203.082 seconds, Fetched: 1 row(s)

Query 1:
Name: tpch_query1.sql
Location: /home/cloudera/D2F-Bench/tpch/queries

```
select
	l_returnflag,
	l_linestatus,
	sum(l_quantity) as sum_qty,
	sum(l_extendedprice) as sum_base_price,
	sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
	sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
	avg(l_quantity) as avg_qty,
	avg(l_extendedprice) as avg_price,
	avg(l_discount) as avg_disc,
	count(*) as count_order
from
	lineitem
where
	l_shipdate <= '1998-09-16'
group by
	l_returnflag,
	l_linestatus
order by
	l_returnflag,
	l_linestatus;
```

Query 2:

```
SELECT
  S_ACCTBAL,
  S_NAME,
  N_NAME,
  P_PARTKEY,
  P_MFGR,
  S_ADDRESS,
  S_PHONE,
  S_COMMENT
FROM
  PART,
  SUPPLIER,
  PARTSUPP,
  NATION,
  REGION
WHERE
  P_PARTKEY = PS_PARTKEY
  AND S_SUPPKEY = PS_SUPPKEY
  AND P_SIZE = 15
  AND P_TYPE LIKE '%%BRASS'
  AND S_NATIONKEY = N_NATIONKEY
  AND N_REGIONKEY = R_REGIONKEY
  AND R_NAME = 'EUROPE'
  AND PS_SUPPLYCOST = (                    # problematic part of the query
    SELECT
      MIN (PS_SUPPLYCOST)
    FROM
      PARTSUPP,
      SUPPLIER,
      NATION,
      REGION
    WHERE
      P_PARTKEY = PS_PARTKEY
      AND S_SUPPKEY = PS_SUPPKEY
      AND S_NATIONKEY = N_NATIONKEY
      AND N_REGIONKEY = R_REGIONKEY
      AND R_NAME = 'EUROPE'
  )
ORDER BY
  S_ACCTBAL DESC,
  N_NAME,
  S_NAME,
  P_PARTKEY
```