Introduction to Hive

@void_io

Introduction to Hive

What is Apache Hive

Uses of Hive

Data Type

Hive CLI

Commands

Schema

Conclusion

What is HIVE?

A system for managing and querying structured data

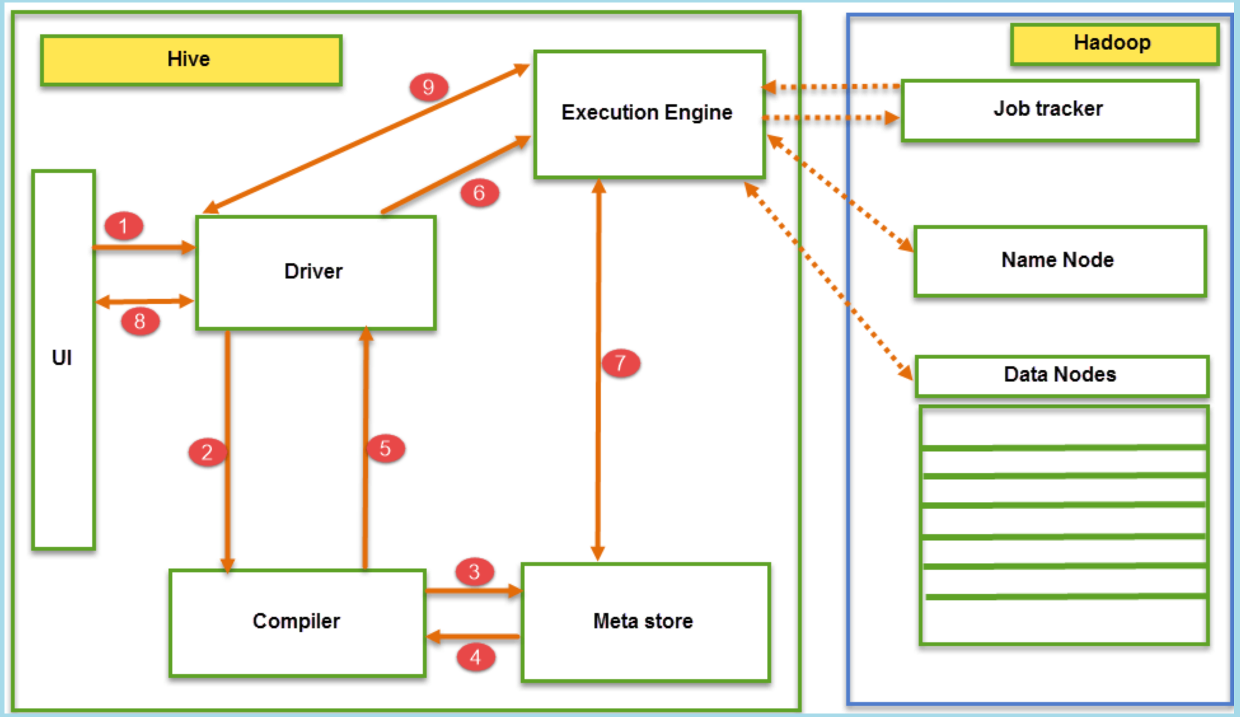
- Uses Map-Reduce for execution
- HDFS for Storage

Key Building Principles

- SQL as a familiar data warehousing tool
- Extensibility (Pluggable map/reduce scripts in the language of your choice, Rich and User Defined Data Types, User Defined Functions)
- Interoperability (Extensible Framework to support different file and data formats)
- Performance

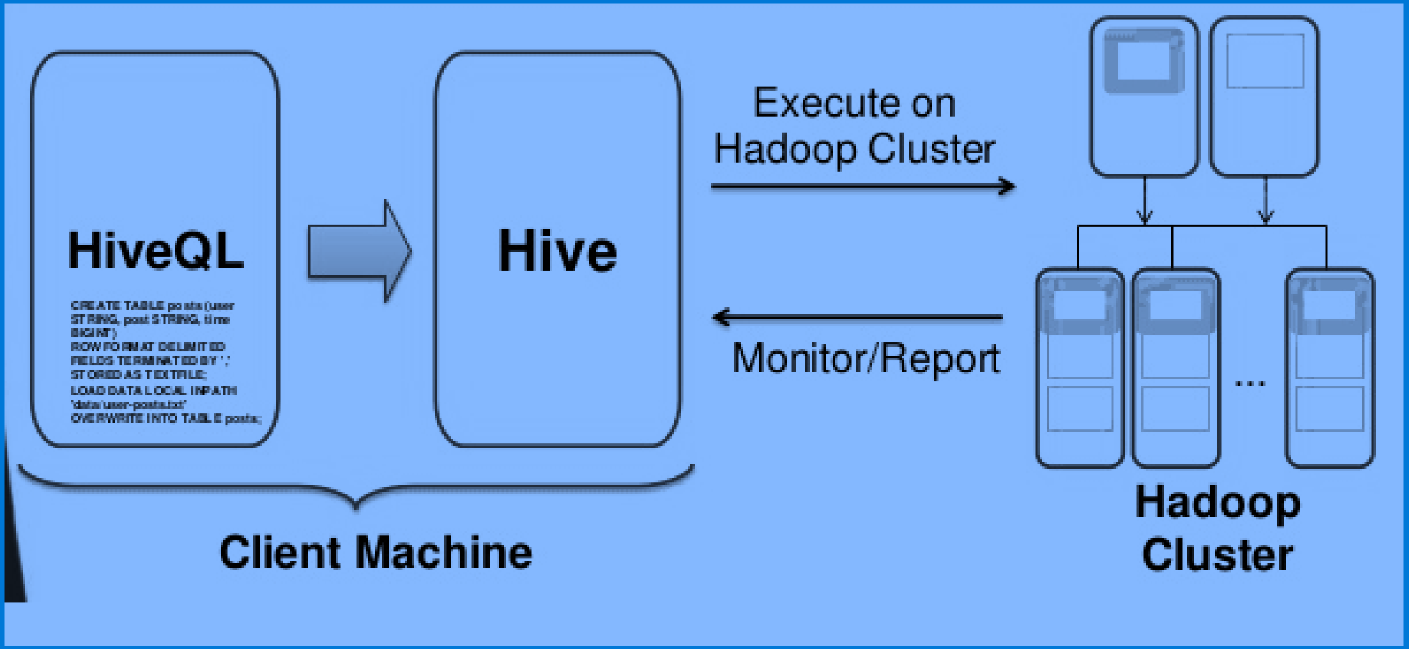
Data Flow in Hive

1. Executing Query from the UI(User Interface)
2. The driver is interacting with Compiler for getting the plan. (Here plan refers to query execution) process and its related metadata information gathering
3. The compiler creates the plan for a job to be executed. Compiler communicating with Meta store for getting metadata request
4. Meta store sends metadata information back to compiler
5. Compiler communicating with Driver with the proposed plan to execute the query
6. Driver Sending execution plans to Execution engine
7. Execution Engine (EE) acts as a bridge between Hive and Hadoop to process the query. For DFS operations:
   - EE should first contacts Name Node and then to Data nodes to get the values stored in tables.
   - EE is going to fetch desired records from Data Nodes. The actual data of tables resides in data node only. While from Name Node it only fetches the metadata information for the query.
   - It collects actual data from data nodes related to mentioned query
   - Execution Engine (EE) communicates bi-directionally with Meta store present in Hive to perform DDL (Data Definition Language) operations. Here DDL operations like CREATE, DROP and ALTERING tables and databases are done. Meta store will store information about database name, table names and column names only. It will fetch data related to query mentioned.
   - Execution Engine (EE) in turn communicates with Hadoop daemons such as Name node, Data nodes, and job tracker to execute the query on top of Hadoop file system
8. Fetching results from driver
9. Sending results to Execution engine. Once the results fetched from data nodes to the EE, it will send results back to driver and to UI (front end)



Uses of Hive

- The Apache Hive distributed storage.
- Hive provides tools to enable easy data extract/load/transform (ELT)
- It provides the structure on a variety of data formats.
- By using Hive, we can access files stored in Hadoop Distributed File System or in other data storage systems such as Apache HBase.



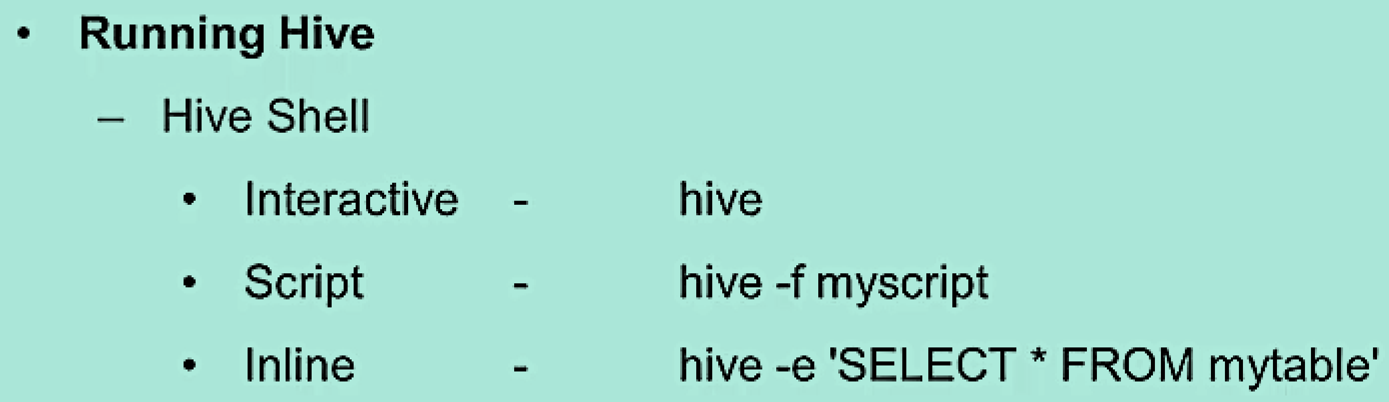
```
CREATE TABLE posts (user
STRING, post STRING, time
BIGINT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
STORED AS TEXTFILE;
LOAD DATA LOCAL INPATH
'data/user-posts.txt'
OVERWRITE INTO TABLE posts;
```

Data Type System

- Primitive types
  - Integers: TINYINT, SMALLINT, INT, BIGINT.
  - Boolean: BOOLEAN.
  - Floating point numbers: FLOAT, DOUBLE .
  - String: STRING.
- Complex types
  - Array: name['a', 'b', 'c']. `:name array<string>`
  - Struct: {a INT; b INT}. `:address struct<city:string,state:string,pin:bigint>`
  - Map: M['group']. `:student map<string,int>`

Hive CLI

- DDL:
  - create table/drop table/rename table
  - alter table add column
- Browsing:
  - show tables
  - describe table
  - cat table
- Loading Data
- Queries



- **Running Hive**
  - Hive Shell

Interactive	-	hive
Script	-	hive -f myscript
Inline	-	hive -e 'SELECT * FROM mytable'

Commands

Create Database

```
CREATE DATABASE [IF NOT EXISTS] db_name;
```

Create Tables

```
CREATE [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.] table_name
(col_name data_type [, col_name ...])
[COMMENT table_comment]
[ROW FORMAT row_format]
[STORED AS file_format];
```

Load Data

```
LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE tablename;
```

Alter Table

```
ALTER TABLE name RENAME TO new_name
ALTER TABLE name ADD COLUMNS (col_spec[, col_spec ...])
ALTER TABLE name DROP [COLUMN] column_name
ALTER TABLE name CHANGE column_name new_name new_type;
```

Commands

Projection Statement

```
SELECT [ALL | DISTINCT] select_expr, select_expr, ...
FROM table_reference
[WHERE where_condition]
[GROUP BY col_list]
[HAVING having_condition]
[CLUSTER BY col_list | [DISTRIBUTE BY col_list]
[SORT BY col_list]]
[LIMIT number];
```

SELECT is the projection operator in SQL. The points are:
•SELECT scans the table specified by the FROM clause
•WHERE gives the condition of what to filter
•GROUP BY gives a list of columns which specify how to aggregate the records
•CLUSTER BY, DISTRIBUTE BY, SORT BY specify the sort order and algorithm
•LIMIT specifies how many # of records to retrieve

Schema

Schema on Read Vs Schema on Write

In a traditional database, a table's schema is enforced at data load time. If the data being loaded doesn't conform to the schema, then it is rejected. This design is sometimes called schema on write, since the data is checked against the schema when it is written into the database.

Hive, on the other hand, doesn't verify the data when it is loaded, but rather when a query is issued. This is called schema on read.

Conclusion

Pros

- A easy way to process large scale data
- Support SQL-based queries
- Provide more user defined interfaces to extend
- Programmability
- Efficient execution plans for performance
- Interoperability with other database tools

Cons

- Accepts only a subset of SQL queries
- No easy way to append data
- Files in HDFS are immutable
- Performance comparisons with other systems would have been more appreciable

Installation Steps

1. Copy hive-0.9.0 software to HOME directory
2. `tar -xzvf hive-0.9.0`
3. `gedit .bashrc`

```
#HIVE Variables
export HIVE_HOME=/home/sohrab/hive-
0.9.0-bin
export HIVE_CONF_DIR=$HIVE_HOME/conf
export HIVE_LIB=$HIVE_HOME/lib
export PATH=$PATH:$HIVE_HOME/bin
export CLASSPATH=$CLASSPATH$HIVE_LIB
```

4. `source .bashrc`
5. Create 2 directory in HDFS

```
$ bin/hadoop fs -mkdir temp
$ bin/hadoop fs -mkdir warehouse
```

6. Grant permission to these 2 directory in HDFS

```
$ bin/Hadoop fs -chmod 755 temp
$ bin/Hadoop fs -chmod 755 temp
```

7. For connection between HDFS and Hive

```
/home/sohrab/hadoop-1.0.3/conf $ gedit hive-
site.xml
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
href="configuration.xsl"?>
<configuration>
<property>
<name>hive.metastore.warehouse.dir</name>
<value>/user/sohrab/warehouse</value></property>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:54310</value></property>
<property>
<name>mapred.job.tracker</name>
<value>hdfs://localhost:54311</value></property>
</configuration>
```

7. Open Hive terminal

```
/home/sohrab/hadoop-1.0.3/bin $ hive
```

Thank you ☺

@void_io