Data Analysis Using Hadoop Hive 1

1 Outline

- Course 1
  - Why we learn Hive
  - Hive introduction
  - How Hive works
  - HQL programming
  - Demo
- Course 2
  - Best Practices for high performance
  - Optimization Hive Usage
  - Most asked interview questions

2. Why we learn Hive

2.1 Let's learn Hive

- Hive can deal with most processes of big data case:
  A. Loaded into file system
  B. Some schema must be applied
  C. Transformed
  D. Analyzed
  E. Visualized
- Hive is a mostly commonly used tool for Big Data.
- A must-have skill for DS, DE, BA, and BI.

Hive originated as a Facebook initiative before becoming a sub-project of Hadoop. In 2008, Facebook started to manage lots of data with Hive, and they publish "Hive a Warehousing Solution Over a MapReduce Framework" Paper in 2009. Before that time, the data was stored in Oracle database every night, and the ELS was performed on data. With the exponential data growth (by 2006 1 TB/day; by 2010 10 TB/day). Therefore, there was a need to find some way to manage the data effectively. Compare with the traditional Oracle database, Hive convert SQL query into a series of MR job, and separation of data & schema. Hive is currently an open source volunteer top-level project under the Apache Software Foundation.

2.2 Motivation

- MapReduce is really powerful, but there are some limitations:
  - Have to use M/R model
  - Not reusable
  - Long development type/overhead
  - For complex jobs: multiple M/R stages
- There are some bright sides of SQL and MapReduce:
  - MapReduce is scalable
  - SQL has huge user base
  - SQL is easy to code

Then we have a good motivation to combine MapReduce and SQL.

2.3 Intuitive

According to the advantage and disadvantage of MapReduce and SQL, there are some intuitive for Hive:

- Make unstructured data looks like tables regardless how it really lays out
- SQL-like query can operate directly against these tables
- Less development time
- Easy for adhoc analysis
- Place for multi-user, multi-session
- User-friendly
- Operates on tables instead of manipulating on file levels.
- Ad-hoc analysis and multi-user, multi-session are basic for data warehousing,

2.4 What is Hive

- Hive is a data warehouse infrastructure built on top of Hadoop for providing data summarization, query and analysis.
- What is Data Warehouse?
  - Is a database specific for analysis and reporting purposes?
- Designed for OLAP
- Provides SQL type language for querying called Hive QL
- Familiar, fast, scalable and extensible

- Syntax very similar to traditional SQL

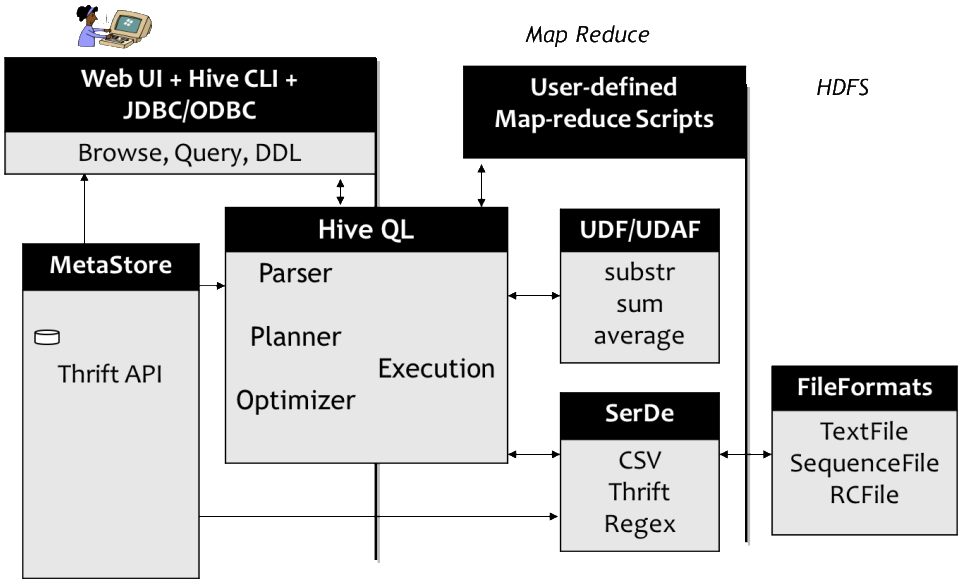
2.5 What Hive is not

- Not work with small data set effective (high latency)
- Not designed for online transaction processing
- Not offer real-time queries (high latency)
- Not work as row level query

3 How Hive Works

- Hive built on top of Hadoop
- Think HDFS and MapReduce
- Hive stored data in the HDFS (Querying and managing large datasets residing in distributed storage)
- Hive compile SQL queries into MapReduce jobs and run the jobs in the Hadoop cluster (syntax very similar to traditional SQL)

3.1 Hive Architecture



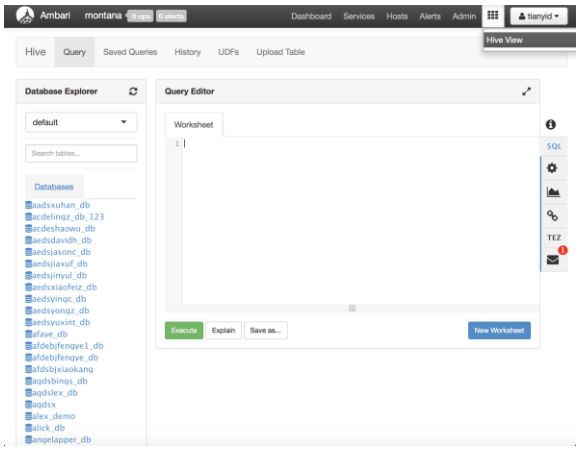
- We can visit Hive by those three ways:
  - Web UI: such as Cloudera, Ambari (Hortonworks), those can build a visual interface by browse, and those can make report or save result to HDFS.
  - Hive CLI: Command Line Interface (such as Beeline)
  - JDBC/ODBC: those are a kind of language independents. (such as Java, Python).

- Hive QL :
  - Parser
  - Planner
  - Optimizer
  - Execution

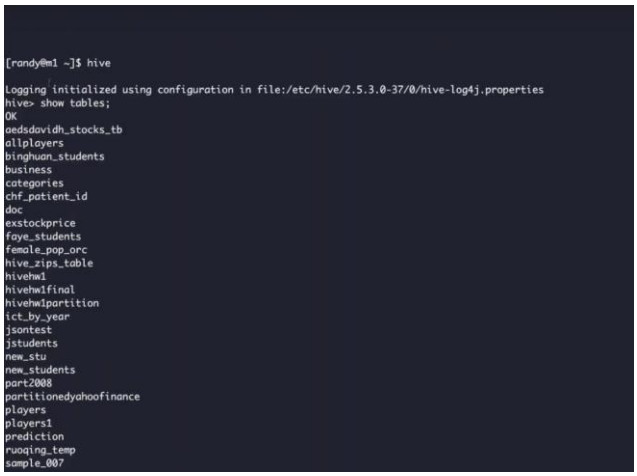
- UDF/UDAF
  - Substr
  - Sum
  - Average
- Serde
  - CSV
  - Thrift
  - Regex

3.2 Demo: Connect to Hive

Log in to Ambari, and click "Hive view" to going to the operator interface of Hive. (Web UI)



Connect to Hive with Terminal:



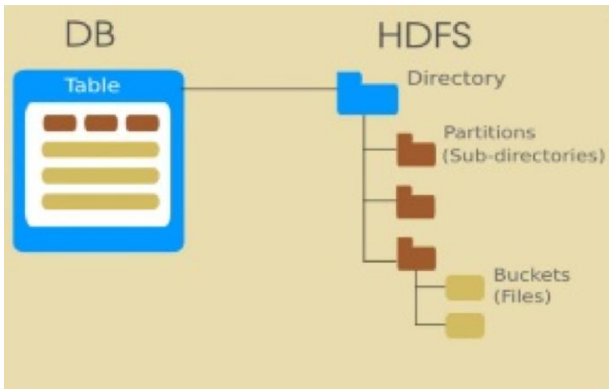
Connect to Hive with Beeline: (Hive CLI)

```
[randy@m1 ~]$ beeline
Beeline version 1.2.1000.2.5.3.0-37 by Apache Hive
beeline> show tables;
No current connection
beeline> !connect jdbc:hive2://m3.mt.dataapplab.com:10000/default
Connecting to jdbc:hive2://m3.mt.dataapplab.com:10000/default
Enter username for jdbc:hive2://m3.mt.dataapplab.com:10000/default: randy
Enter password for jdbc:hive2://m3.mt.dataapplab.com:10000/default: *********
Connected to: Apache Hive (version 1.2.1000.2.5.3.0-37)
Driver: Hive JDBC (version 1.2.1000.2.5.3.0-37)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://m3.mt.dataapplab.com:10000/de> show tables;
+-----------------------------+--+
|          tab_name           |
+-----------------------------+--+
| aedsdavidh_stocks_tb        |
| allplayers                  |
| binghuan_students           |
| business                    |
| categories                  |
| chf_patient_id              |
| doc                         |
| exstockprice                |
| faye_students               |
| female_pop_orc              |
| hive_zips_table             |
| hivehw1                     |
| hivehw1final                |
| hivehw1partition            |
| ict_by_year                 |
```

4 HQL Programming

4.1 Hive Data Model

Hive Structure data into a well-defined database concept:



- **Tables**
  - Basic type columns: int, float, boolean...
  - Complex types: array, map, struct...
  - `CREATE TABLE employee(id INT, name STRING);`
- **Partitions**
  - Such as range partition tables by date:
  - `CREATE TABLE sales(id INT, iterms STRING) PARTITIONED BY (ds STRING);`
- **Buckets**
  - Useful for sampling
  - `CREATE TABLE sales(id INT, iterms STRING)`

```
PARTITIONED BY (ds STRING)

CLUSTERED BY (id) INTO 32 BUCKETS;

SELECT id FROM sales TABLESAMPLE(BUCKET 1 OUT OF 32
);
```

4.2 Metadata

Metadata is data that describes other data. Metadata summarizes basic information about data, which can make finding and working with particular instances of data easier:

- Database
  - Namespace containing a set of tables
- Tabledefinitions
  - Contains list of columns and their types and SerDe Info
  - Schema info, physical location on HDFS
- Partition
  - Each partition can have its own columns and storage info
- Statistics
  - Info about the databases

4.3 Hive Physical Layout

- Warehouse directory in HDFS
- Table row data is stored in warehouse sub-directories
- Partition creates sub-directories within table directories

4.4 Demo for Basic Hive Operation

4.4.1 Creation of a table on hive - Beginner practice

1. Create tables, load data
2. Query and analyze data

Step 1. create a text file in local system

```
mkdir -p /home/randy/demo/students_tb/students/
vi /home/randy/demo/students_tb/students/students.txt
```

Step 2. paste below rows into the file

```
1,Nic,Raboy,Merced,California
2,Jane,Doe,Newark,New York
3,John,Lee,Las Vegas,Nevada
```

```
4,Maria,Campos,Modesto,California
```

Step 3. Create an empty table for students

```
CREATE TABLE students(id INT, first_name STRING, last_name STRING,
 city STRING, state STRING) ROW FORMAT DELIMITED FIELDS TERMINATED
 BY ',';

-- show tables;

-- describe tablename;
```

Step 4. Load the text file into this table

```
LOAD DATA LOCAL INPATH '/home/randy/demo/students_tb/students/stud
ents.txt' OVERWRITE INTO TABLE students;
```

Step 5. query this table to validate it

```
-- command line interface

SELECT * FROM students;


-- Web UI (Ambari)

SELECT * FROM students where state="California";
```

4.5 Load data into a Hive Table

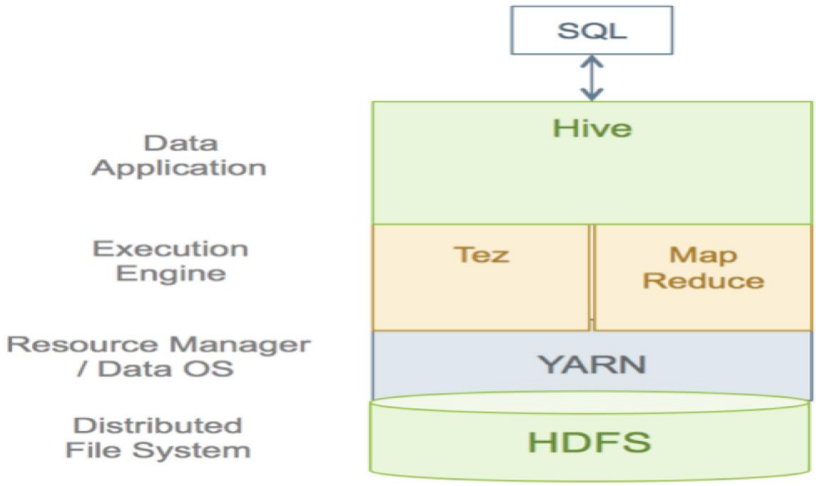
- Hive does not do any transformation while loading data into tables
- Load operations are currently pure copy/move operations that move data file s into locations corresponding to hive tables
- `LOAD DATA LOCAL INPATH '/demo/students_tb/students/students. txt' OVERWRITE INTO TABLE students;`

  `---describe formatted students;`

4.6 Hive Execution

- 3 ways to check tables and connect with Hive:
  - CLI: Hive CLI & Beeline
  - Web UI: Hue, Ambari
  - JDBC/ODBC
- Run one query
  - hive -e：
    - `hive -e 'SELECT DISTINCT username FROM temp.Twitter Exampletextexample LIMIT 10'`
- Run a Hive query file
  - hive -f：
    - `hive -f /tmp/demo_hive.sql`

4.7 SQL Query Execution Process



In the relational engine, a query is parsed and then processed by the query optimizer, which generates an execution plan. When any query reaches SQL Server, the first place it goes to is the relational engine.

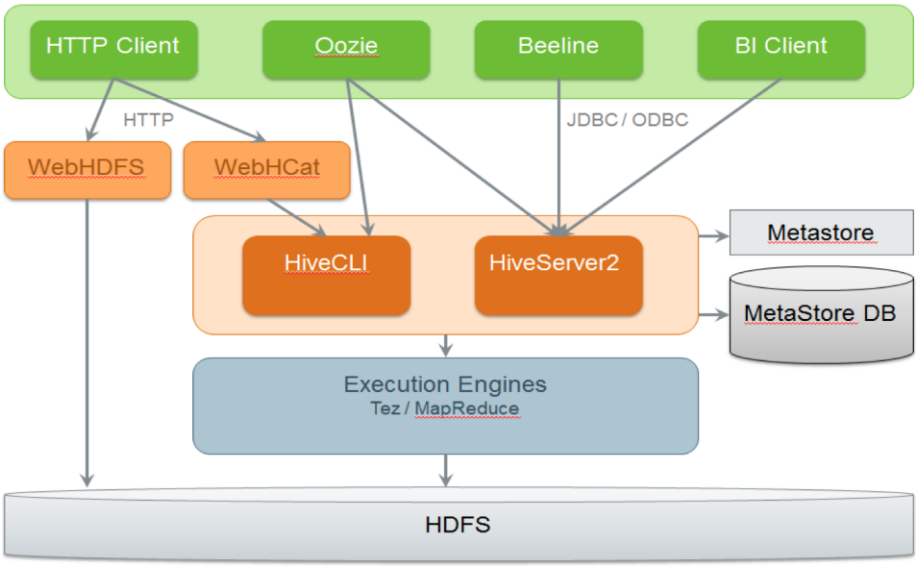
4.8 Data Type

- Integer Type:
  - TINYINT

    - SMALLINT
    - INT
    - BIGINT
- BOOLEAN
- Decimal Type
    - FLOAT
    - DOUBLE
    - BIGDECIMAL
- STRING
- BINARY
- TIMESTAMP

4.9 Hive Stack

The conjunction part of HiveQL process Engine and MapReduce is Hive Execution Engine. Execution engine processes the query and generates results as same as MapReduce results. It uses the flavor of MapReduce. Hadoop distributed file system is the data storage techniques to store data into file system.



4.10 Hive Database Features

- All about files
- Schema on read
- Fast when load data into DB

- Touch data only when run query
- Don’ t support delete/update

4.11 Data Units

- **Database**
  - CREATE
  - USE
  - DROP (delete)
    - **CREATE** (DATABASE|SCHEMA) [IF **NOT** EXISTS] database_name
    - [COMMENT database_comment]
    - [LOCATION hdfs_path]
    - [WITH DBPROPERTIES (property_name=property_value, ...)];
- **Table**
  - External vs internal
  - DROP (delete)

```
CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_name-- (Note: TEMPORARY available in Hive 0.14.0 and later)

  [(col_name data_type [COMMENT col_comment], ...)]

  [COMMENT table_comment]

  [PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)]

  [CLUSTERED BY (col_name, col_name, ...) [SORTED BY (col_name [ASC|DESC], ...)] INTO num_buckets BUCKETS]

  [SKEWED BY (col_name, col_name, ...)]-- (Note: Available in Hive 0.10.0 and later)

     ON ((col_value, col_value, ...), (col_value, col_value, ...), ...)

     [STORED AS DIRECTORIES]

  [

   [ROW FORMAT row_format]
```

```
   [STORED AS file_format]

     | STORED BY 'storage.handler.class.name' [WITH SERDEPROPERTIE
S (...)]-- (Note: Available in Hive 0.6.0 and later)

  ]

  [LOCATION hdfs_path]

  [TBLPROPERTIES (property_name=property_value, ...)]-- (Note: Ava
ilable in Hive 0.6.0 and later)

  [AS select_statement];-- (Note: Available in Hive 0.5.0 and late
r; not supported for external tables)
```

- **Partition** – are done on columns
    - `CREATE TABLE students_part(ID INT, Name STRING) PARTITIONED BY (dept STRING)`
    - Above command create a sub-directory for each value of the partition column
        - `/user/hive/warehouse/stutents_part/dept=cs/`
    - Queries with partition columns in WHERE clause will scan through only a subset of data
- Basic Queries
    - Select
    - Count
    - Join
    - Group
    - *LOAD DATA INPATH 'hdfs_file_or_directory_path' [OVERWRITE] INTO TABLE tablename [PARTITION (partcol1=val1, partcol2=val2 ...)]*