Big Data Analytics and Reasoning - Practice 03

Giuseppe Mazzotta

HIVE

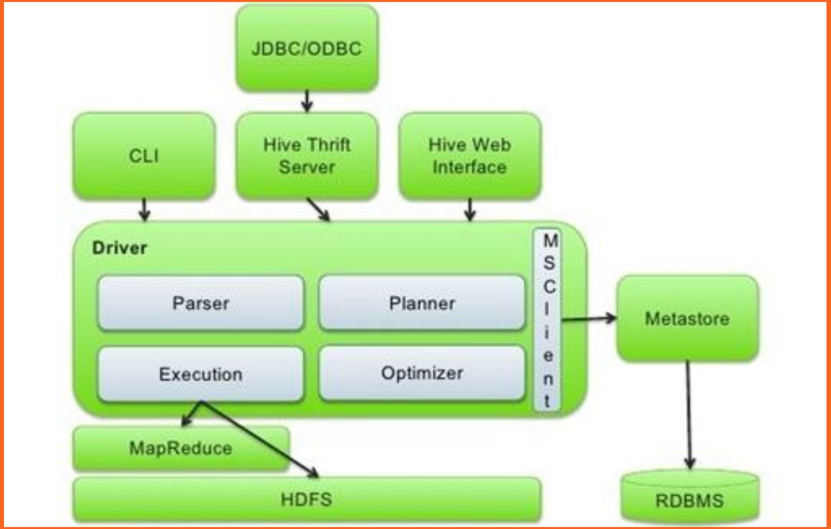
Hive is distributed data warehouse framework

Store structured data as files into hdfs

Designed to be used with an SQL-like syntax

Use internally mapreduce

Use RDBMS to store tables metadata



Download and Install Hive

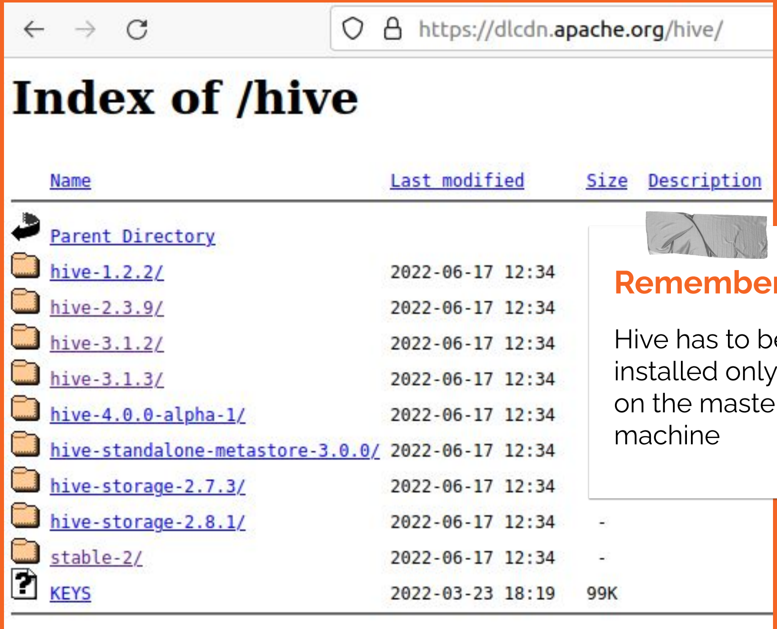
Download the binary archive of the hive distribution from the official website in the master machine

Used version 3.1.3

Unfold the archive and export into .bashrc:

HIVE_HOME

PATH : ${HIVE_HOME}/bin



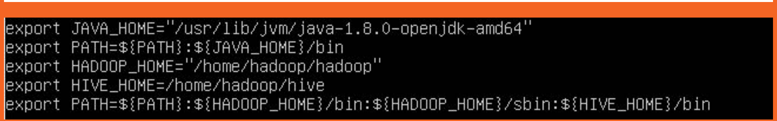
https://dlcdn.apache.org/hive/

Index of /hive

Name	Last modified	Size	Description
Parent Directory			
hive-1.2.2/	2022-06-17 12:34		
hive-2.3.9/	2022-06-17 12:34		
hive-3.1.2/	2022-06-17 12:34		
hive-3.1.3/	2022-06-17 12:34		
hive-4.0.0-alpha-1/	2022-06-17 12:34		
hive-standalone-metastore-3.0.0/	2022-06-17 12:34		
hive-storage-2.7.3/	2022-06-17 12:34		
hive-storage-2.8.1/	2022-06-17 12:34	-	
stable-2/	2022-06-17 12:34	-	
KEYS	2022-03-23 18:19	99K	

Remember

Hive has to be installed only on the master machine



```
export JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk-amd64"
export PATH=${PATH}:${JAVA_HOME}/bin
export HADOOP_HOME="/home/hadoop/hadoop"
export HIVE_HOME=/home/hadoop/hive
export PATH=${PATH}:${HADOOP_HOME}/bin:${HADOOP_HOME}/sbin:${HIVE_HOME}/bin
```

MySQL configuration for Hive

- Install mysql-server on one machine
- Create a user on mysql with all privileges to be used by hive
    a. *create user ‘username’@’%’ identified by ‘password’*
    b. *grant all privileges on *.* to ‘username’@’%’*
    c. *flush privileges*
- Install mysql connector - libmysql-java package or download from https://dev.mysql.com/downloads/connector/j/
- Disable ssl on mysql service:
    a. *sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf*
    b. Append to the file: *skip_ssl*
    c. *sudo service mysql restart*
    d. Log into mysql and run: *show variables like ‘%ssl%’*
    e. Expected output:

```
have_openssl        DISABLED
have_ssl            DISABLED
```

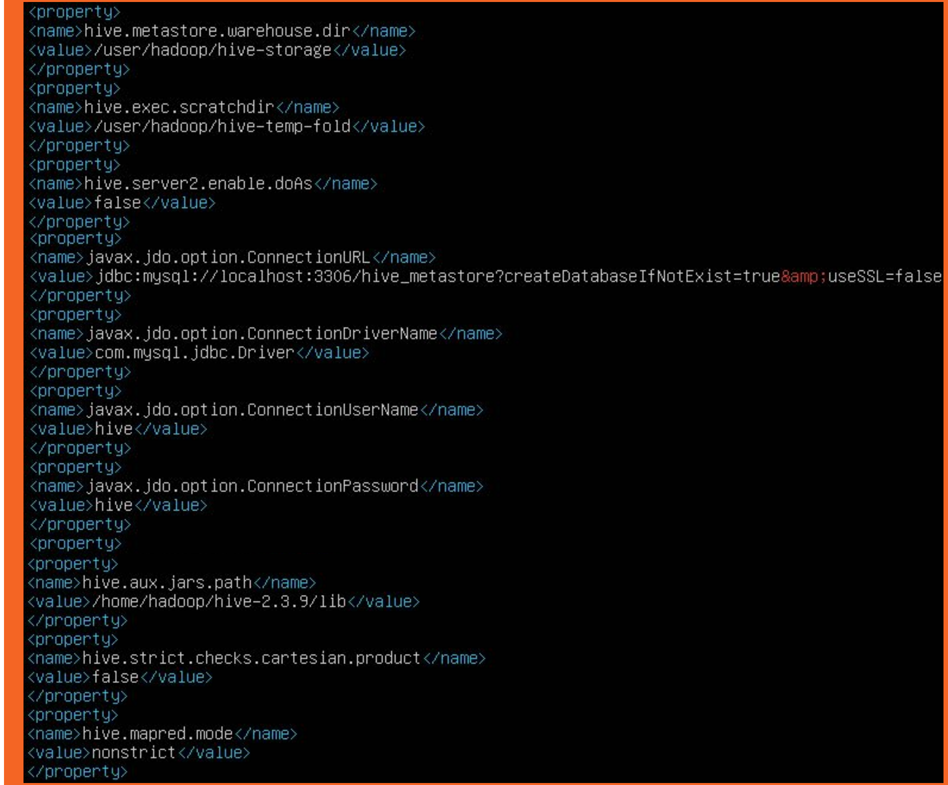
Configure Hive

Hive configuration file is located into ${HIVE_HOME}/conf

Main configuration:

- Hive warehouse location
- Hive anonymous access
- MySQL configuration

Remark: Hive doesn’t have slave services then it has to be configured only on the master machine.



```
<property>
<name>hive.metastore.warehouse.dir</name>
<value>/user/hadoop/hive-storage</value>
</property>
<property>
<name>hive.exec.scratchdir</name>
<value>/user/hadoop/hive-temp-fold</value>
</property>
<property>
<name>hive.server2.enable.doAs</name>
<value>false</value>
</property>
<property>
<name>javax.jdo.option.ConnectionURL</name>
<value>jdbc:mysql://localhost:3306/hive_metastore?createDatabaseIfNotExist=true&amp;useSSL=false
</property>
<property>
<name>javax.jdo.option.ConnectionDriverName</name>
<value>com.mysql.jdbc.Driver</value>
</property>
<property>
<name>javax.jdo.option.ConnectionUserName</name>
<value>hive</value>
</property>
<property>
<name>javax.jdo.option.ConnectionPassword</name>
<value>hive</value>
</property>
<property>
<property>
<name>hive.aux.jars.path</name>
<value>/home/hadoop/hive-2.3.9/lib</value>
</property>
<property>
<name>hive.strict.checks.cartesian.product</name>
<value>false</value>
</property>
<property>
<name>hive.mapred.mode</name>
<value>nonstrict</value>
</property>
```

Starting hive server

Initialization step

- ***schematool -dbType mysql --initSchema***

Starting hiveserver2

- ***hiveserver2***

Client will use **beeline** to open a CLI with hiveserver

- ***beeline -u jdbc:hive2://master:10000***
- ***show databases*** #query example to test hive connectivity

Tip

Share the hive configuration with the client

Suggestion: open a screen session to run hiveserver2

1. Hive Data Types

- **Primitive**

  Primitive types in hive are string, int, float, boolean, date, timestamp and more

- **Complex**

  Are built on top of the primitive types
  Allow the nesting of primitive and complex types

  - **Array**

    List of items of the same type

  - **Map**

    Set of key value pairs

  - **Struct**

    User-defined structure of any number of typed fields

2. Basic Concepts

→ **Database**

Collection of tables that are used for similar purpose
Represented by a directory into hdfs - *default* database

→ **Tables**

Collection of data that share the same schema - It belongs to a database - Represented as a subfolder of the database folder

→ **Partitions**

Extra columns that divide data into different subfolders of the table folder

→ **Buckets**

Existing columns that divide data into a fixed number of file (buckets) according to an hash function

→ **Views**

Logical data structures used to queries - Are defined in metastore only - Do not reflect changes on original table after view creation

2. Tables

Hive tables are almost the same of relational tables:

➔ **Fixed schema**

Collection of homogeneous data - Each row has the same attributes (columns) - Different row formats - Schema-on-read

➔ **Managed, External and Temporary**

Managed tables are fully handled by hive
External tables define a schema for data already stored into hdfs
Temporary tables lives in a user session - stored in hive.exec.scratchdir

➔ **File Formats**

Data are stored as files into the table directory - Plain Text, Parquet, ORC and more

Managed vs External

Managed tables are completely handle by hive; data will be stored as files into the table folder; Once we drop the table, files containing data will be deleted from HDFS.

External tables are commonly used when we want to define a schema for files already stored into hdfs; If we drop the table the data remains into the hdfs.

Tip

Managed tables often used as intermediate tables

External tables are often used as read-only tables

CTAS: Created Table As Select

File Formats

Plain Text is the default file format:

Human readable format;
Data are not indexed

Parquet is a columnar file format:

Compressed format
Designed to work well on top of the hdfs

ORC is a fully indexed file format

uses type specific readers and writers (lightweight compression)

supports projection for reading only required bytes for a given column

Tip

The best file format depends on your needs

Parquet Format

Given a table of N columns

Rows are grouped in M row groups

Data are stored in a matrix-like format NxM

For each row group columns are stored sequentially together with column metadata

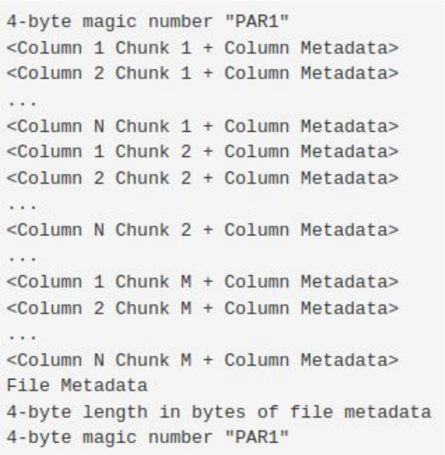
The file metadata contains the columns locations

It supports a fast sequential reading

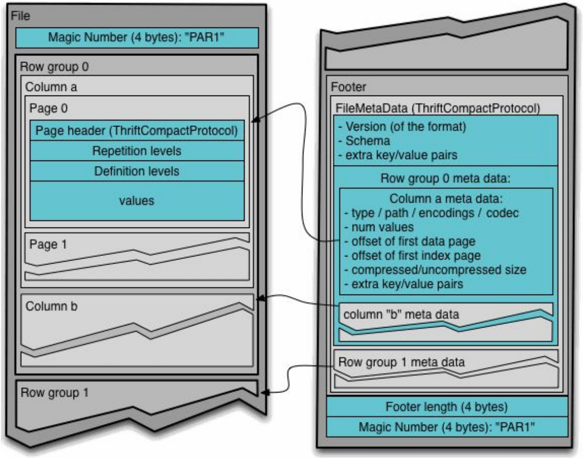
Metadata follows the data:

Writers directly append metadata after data

Readers read metadata first and then can easily access data



```
4-byte magic number "PAR1"
<Column 1 Chunk 1 + Column Metadata>
<Column 2 Chunk 1 + Column Metadata>
...
<Column N Chunk 1 + Column Metadata>
<Column 1 Chunk 2 + Column Metadata>
<Column 2 Chunk 2 + Column Metadata>
...
<Column N Chunk 2 + Column Metadata>
...
<Column 1 Chunk M + Column Metadata>
<Column 2 Chunk M + Column Metadata>
...
<Column N Chunk M + Column Metadata>
File Metadata
4-byte length in bytes of file metadata
4-byte magic number "PAR1"
```


File
Magic Number (4 bytes): "PAR1"
Row group 0
Column a
Page 0
Page header (ThriftCompactProtocol)
Repetition levels
Definition levels
values
Page 1
Column b
Row group 1
Footer
FileMetaData (ThriftCompactProtocol)
- Version (of the format)
- Schema
- extra key/value pairs
Row group 0 meta data:
Column a meta data:
- type / path / encodings / codec
- num values
- offset of first data page
- offset of first index page
- compressed/uncompressed size
- extra key/value pairs
column "b" meta data
Row group 1 meta data
Footer length (4 bytes)
Magic Number (4 bytes): "PAR1"

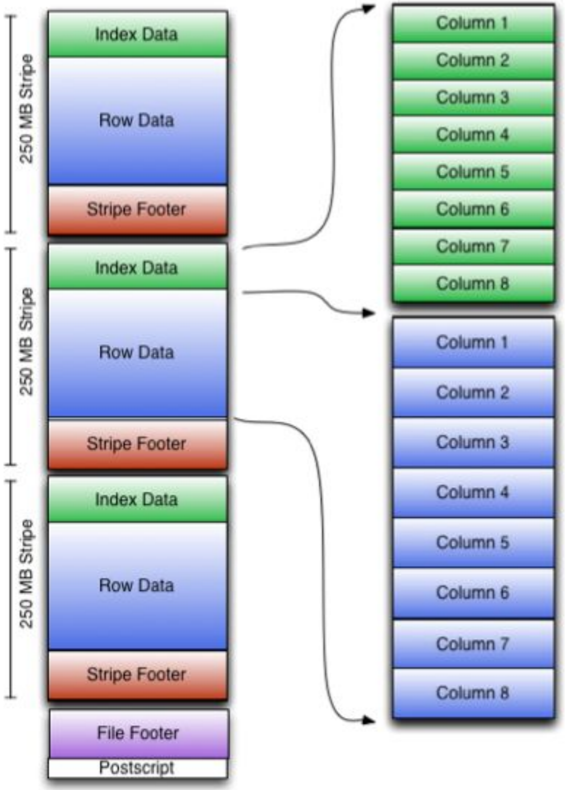
ORC Format

ORC - Optimized Row Columnar (https://orc.apache.org/specification/ORCv1/)

ORC files are stored as binary files -> not human readable

File structure:

- N stripes of 250 MB size, independent of each other
  Stripe structure:
  - Index data -> stores min and max for each column and the row positions for each column
  - Row Data -> contains a block of row stored in a columnar way
  - Stripe footer -> contains stream locations
- File footer -> contains the description of the file content; number of rows, columns data types, statistics about each column and the list of the stripes
- Postscript -> contains information to interpret the file, length of the file footer, compression parameters and more



Let us practice with Hive!

Tip

HQL references

https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DDL

https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DML

https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Select

https://cwiki.apache.org/confluence/display/Hive/LanguageManual+UDF