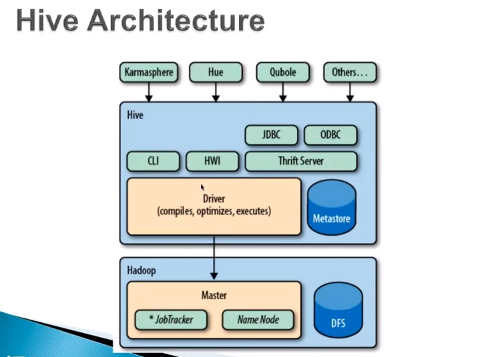
**Quick ref: Hive**

**Architecture - 1**



Metastore – will be used to store logical structure

|  |  |  |
| --- | --- | --- |
| **S.No** | **Topic** | **Desc** |
|  | **Web Reference** |  |
|  |  | <https://www.safaribooksonline.com/library/view/programming-hive/9781449326944/ch04.html> |
|  | Both Hive & Impala | <https://www.youtube.com/watch?v=yrXCp9VXyUA&index=18&list=PLf0swTFhTI8pOZ4VBSGerKUmF9USWL6vd> |
|  |  |  |
|  |  |  |
|  | **General Info** |  |
|  | **Architecture** | 1. **CLI – Command Line Interface** 2. **HWI – Hive Web Interface** 3. **ThriftServer -** HiveServer2 (HS2) is a server interface that enables remote clients to execute queries against Hive and retrieve the results. The current implementation, based on Thrift RPC, is an improved version of [HiveServer](https://cwiki.apache.org/confluence/display/Hive/HiveServer) and supports multi-client concurrency and authentication. It is designed to provide better support for open API clients like JDBC and ODBC |
| **1** | **Difference between Hive, Impala and HBase** | **Hive, Impala:**   1. Used for both reading & writing data into DB. But mainly used for reading data from DB. 2. Hive will use MapReduce 3. Hive is used for batch processing   **Impala:**   1. Used for both reading & writing data into DB. But mainly used for reading data from DB. 2. Impala will NOT use MapReduce 3. Impala is for kind of adhoc query   **HBase:**   1. Used for both reading & writing data into DB. But mainly used for writing data into DB 2. HBase will NOT use MapReduce |
| **2** | **>hive** | To login to hive |
| **2.1** | **Hive log file** | For every user a directory will be created under /tmp |
| **3** | **Table name is a directory** | # Hive Table name is a directory   1. Based on the table name it will create the directory 2. It stores table name, column name and data type in Metastore and actual data will be in HDFS (actual data will not be in Hive layer) 3. **Hive is completely logical. It has structure on top of data from HDFS. Hive itself will never store data** |
| **3.1** | **Pre-request** | # pre-request   1. HDFS – To store the data 2. MapReduce – To generate MR program automatically 3. Resource Manager – To process the MR & data. 4. MySQL, <any RDBMS> -- To maintain metastore |
| **4** | **Three Componenets** | **# To make it more simple, we can say Hive has 3 componenets**  a. DDL or Physical Modeling  b. Copying data (LOAD or INSERT)  c. Querying data (Hive QL) |
| **5** | **HiveQL** | # Things can be done using HiveQL   1. Aggregation |
| **6** | There is a common point between Hive & Impala: Need to identify | There is a common point between Hive & Impala: Need to identify |
| **7** | **Hive: User defined Fucntions (UDF)** | #hive API used to create user defined function.  #Get Jar file and Classname from Developer. Ex: lca.lca\_to\_date is the java class name for hive UDF  #To make it consistent to all the jobs triggered by the user. you need to update .hiverc file  >add jar /home/hduser/demo/hive/lca\_hive.jar;  >Create temporary function lca\_to\_date as 'lca.lca\_to\_date';  Now developer can use function.  #To make it consistent to all the jobs triggered by the user. you need to update .hiverc file  > vi .hiverc  use the function name  Get the steps from developer to understand how to validate. so that you can validate and deliver to developer |
| **8** | **Hive: Increase spliter size , so that block size will be increase for only Hive** | # If default block size if 128 MB then MapReduce will use 128 mb. But Hive will use 256 MB based on below parameter. Setting this to 1GB may increaes the performance   * set mapreduce.input.fileinputformat.split.maxsize = 256000000; |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  | **General commands/Info** |  |
| 1.1 | **Hive Configuration**  **<Metastore – MySQL>**  **<pre-request>** | # In RDBMS, we have one DB and multiple schema where as in MySQL we can create as many DB as we want  >mysql –u root –p  Enter pwd:  mysql >show databases;  Rman – created to maintain CDH & CM  mysql>create database if not exists hive;  mysql>create user ‘hive’ identified by ‘hadoop20’;  mysql>grant all on hive.\* to hive; |
| 1.2 | **Hive Configuration**  **<Cloudera Manager>** | # Gateway: by default all the nodes  #What will happen as soon as you configure hive in your node? Some internal steps   1. CM will create metastore database tables 2. CM will create hive user directory in HDFS. >haoop fs –ls /user/hive 3. CM will create hive data warehouse directory under. /user/hive/warehouse |
| 1.3 | **Hive Configuration**  **<Temporary>** | # Apart from hive-site.xml, if you want to alter some parameter during runtime then create a hidden file called ‘.hiverc’  Go to corresponding path  >vi .hiverc |
| 2 | **Hive - Database – Manage**  **<Database creation>** | # Create database  hive> SHOW DATABASES;  default  financials  hive> CREATE DATABASE if not exists human\_resources;  hive> SHOW DATABASES;  default  financials  human\_resources  hive> DROP DATABASE IF EXISTS financials CASCADE;  Using the RESTRICT keyword instead of CASCADE is equivalent to the default behavior, where existing tables must be dropped before dropping the database. When a database is dropped, its directory is also deleted. |
| 2.1 | **Hive – Database – Manage** | # After you create you DB, the new DB will be available in the below path  HDFS>Hadoop fs –ls /user/hive/warehouse  or  Hive> dfs -ls /user/hive/warehouse  You can see the DB “human\_resoures.db”;  Note: The below command shows empty folder as no table created now  HDFS>Hadoop fs –ls /user/hive/warehouse/human\_resource.db  or  Hive> dfs -ls /user/hive/warehouse/human\_resource.db |
| 2.3 | **To select tables** | >select count(\*) from table1; -- Will use MapReduce to get no of rec |
| 3 | **Three Components** | **# To make it more simple, we can say Hive has 3 components**  a. DDL or Physical Modeling  b. Copying data (LOAD or INSERT)  c. Querying data (Hive QL) |
| 3.1 | **DDL or Physical Modeling**  **Data Type**   * *1. Integral Types (TINYINT, SMALLINT, INT, BIGINT)* * *2. Strings.* * *3. Varchar.* * *4. Char.* * *5. Timestamps.* * *6. Dates. Casting Dates.* * *7. Decimals. Decimal Literals. Decimal Type Incompatibilities between Hive 0.12.0 and 0.13.0. Upgrading Pre-Hive 0.13.0 Decimal Columns.* * *8. Union Types.* | **# Creating hive table**  Note: by default, hive login to default DB. So change the DB  Hive>use human\_resoures; -- To create table under this DB  Hive> CREATE TABLE large\_deck (  COLOR string,  SUIT string,  PIP string)  ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'  STORED AS TEXTFILE;  Note: The below command shows empty folder as table just created and no data loaded  HDFS>Hadoop fs –ls /user/hive/warehouse/human\_resource.db/large\_deck  or  Hive> dfs -ls /user/hive/warehouse/human\_resource.db/large\_deck |
| 3.2 | **Create table with parameter, Load data**  ***<Parameterized table creation and load data>*** | # To create the parameterized table (tablename)  CREATE EXTERNAL TABLE ${tablename}(username STRING, profileLocation STRING, tweetId STRING, content STRING, fav STRING, retweet STRING, retcount STRING, CreatedAt STRING)  Comment ‘This is the Twitter table’  ROW FORMAT DELIMITED  FIELDS TERMINATED BY ‘\t’  LINES TERMINATED BY ‘\n’  STORED AS TEXTFILE;  # To load the parameterized table (input)  LOAD DATA INPATH “${input}” OVERWRITE INTO TABLE ${tablename};  # To put the data in parameterized directory (output)  INSERT OVERWRITE DIRECTORY ‘${output}’ SELECT profilelocation, COUNT(content) as count FROM ${tablename} GROUP BY profilelocation ORDER BY count desc limit 10; |
| 3.1.1 | **Describe table** | >describe large\_deck;  >describe extended large\_deck ; *-- To get more details*  *>describe formatted large\_deck; -- Formated output of desc* |
| 3.2 | **Load Data**  **<To load data from local to HDFS>** | # It will not validate, just dump the data. But shows error while query in case of issue with file format vs table format  # LOAD will work like ‘hadoop fs –put local\_path hdfs\_path’  **# To load data from local path:**  LOAD DATA LOCAL INPATH '/home/hduser/data/cards/largedeck.txt' INTO TABLE large\_deck;  **# To load data from HDFS path:**  LOAD DATA INPATH '/home/hduser/data/cards/largedeck.txt' INTO TABLE large\_deck; |
| 3.3 | **Insert Data** | # Insert will run the mapreduce behind the scene  # You can change the delimiter while loading data (won’t dump just like load)  CREATE TABLE deck\_of\_cards\_pby\_pip (  COLOR string,  SUIT string)  PARTITIONED BY (PIP string)  ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'  STORED AS TEXTFILE;  INSERT OVERWRITE TABLE deck\_of\_cards\_pby\_pip PARTITION (pip) SELECT \* FROM small\_deck;  # Consider the ‘partitioned’ column PIP has value of 1,2,3 then instead of single directory, it will create three directory as follows  Deck\_of\_cards\_pby\_pip/pip=1  Deck\_of\_cards\_pby\_pip/pip=2  Deck\_of\_cards\_pby\_pip/pip=3 |
| 3.4 | Querying Data (Hive QL) | # Simple select  >select \* from table\_name limit 10; -- \* will not use mapreduce  >select col1, col2 from table\_name; col1 will use mapreduce  # Group by   * SELECT PIP, SUIT, count(1) FROM deck\_of\_cards\_pby\_pip GROUP BY PIP, SUIT; * Note: the above group by query will show a http path, copy paste it in WebUI to see the progress * In general, there is no WebUI for Hive |
| 4 | Hive – Functions | # Hive functions  Hdfs>hive –e “show functions”; -- without login to hive  Hive>show functions; -- after login to hive   1. Predefined – As part of hive configuration, hive will automatically create predefined functions 2. Sum, min, max 3. Like, not like 4. Or , and 5. Date\_diff, date\_add 6. Nvl 7. Trim, upper, lower 8. Reverse 9. Sin, tan 10. UDF – 11. Developer can create user defined function in java and provide jar file to admin 12. Developer will also provide create command to create function 13. Admin has to update .hiverc file |
| 5 | Hive commands from HDFS  <Connect from HDFS to hive> | **# Using “hive –e”.** Without login to hive. You can execute from hdfs. It will execute the command “show functions” and come out of hive immediately  Hdfs>hive –e “show functions”;  **# Using “hive –f”.** As part of automation, Create a bash shell with set of commands like create table, run query, etc. we need to use “hive –f”  Hdfs>vi hive\_test.hql  Show functions  :wq  Hdfs>ls –ltr  Hdfs>hive –f hive\_test.hql;  It will show all functions;  **# Set Parameter - # Pending**  >set – parameter temporarly  **# Hive conf- # Pending**  >hive -hiveconf |
| 6 | HDFS commands from Hive  <opposite to above steps>  <Connect from HDFS to hive> | # dfs from hive. Dfs is an alias for ‘hadoop fs’  Hive> dfs -ls /user/hive/warehouse |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

1. **Database:**
2. **Create table:**

**CREATE** **TABLE** IF **NOT** **EXISTS** mydb.employees (

name STRING **COMMENT** 'Employee name',

salary FLOAT **COMMENT** 'Employee salary',

subordinates ARRAY<STRING> **COMMENT** 'Names of subordinates',

deductions **MAP**<STRING, FLOAT>

**COMMENT** 'Keys are deductions names, values are percentages',

address STRUCT<street:STRING, city:STRING, **state**:STRING, zip:INT>

**COMMENT** 'Home address')

**COMMENT** 'Description of the table'

TBLPROPERTIES ('creator'='me', 'created\_at'='2012-01-02 10:00:00', ...)

**LOCATION** '/user/hive/warehouse/mydb.db/employees';

WARNING:If you use IF NOT EXISTS and the existing table has a different schema than the schema in the CREATE TABLE statement, Hive will ignore the discrepancy.

1. **Database and table path:**

Finally, you can optionally specify a location for the table data (as opposed to metadata, which the metastore will always hold). In this example, we are showing the default location that Hive would use, /user/hive/warehouse/mydb.db/employees, where /user/hive/warehouse is the default “warehouse” location (as discussed previously), mydb.db is the database directory, and employees is the table directory.

By default, Hive always creates the table’s directory under the directory for the enclosing database. The exception is the default database. It doesn’t have a directory under /user/hive/warehouse, so a table in the default database will have its directory created directly in /user/hive/warehouse (unless explicitly overridden).

1. **Your developers request that you enable them to use Hive on your Hadoop cluster. What do install and/or configure?**

Install the Hive interpreter on the client machines only, and configure a shared remote Hive Metastore.

The Hive Interpreter runs on a client machine

1. dsf