Session 6

Assignment 3

|  |  |
| --- | --- |
| **Prepared For:** | AcadGild |
|  |  |
| **Document Approval:** | **AcadGild** |
|  |  |
|  |  |
|  |  |
|  |  |
| **Project Title:** | Session 6 – Assignment 3 |
|  |  |
| **Prepared By:** | Duncan Burgess |
|  |  |
|  | dburgess@duncb.com |
|  |  |
| **Primary Engineer:** | Duncan Burgess |
|  |  |
| **Document Reference:** | **Session 6 – Assignment 3** |
|  |  |
| **Start Date:** | 18/09/2017 |
|  |  |
|  |  |

# 

# Contents

[Contents 2](#_Toc493496208)

[Change History 3](#_Toc493496209)

[1. Problem Statement 4](#_Toc493496210)

[2. Solution 5](#_Toc493496211)

[2.1. Hadoop Hive Architecture 5](#_Toc493496212)

[2.2. Major Components of Hive 7](#_Toc493496213)

# Change History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Document Revision** | **Date** | **Authored By** | **Authorised By** | **Sections Affected** | **Reason for Change** |
| Rev 01 | 18/09/2017 | Duncan Burgess |  | All | Initial release. |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Problem Statement

**Problem Statement:**

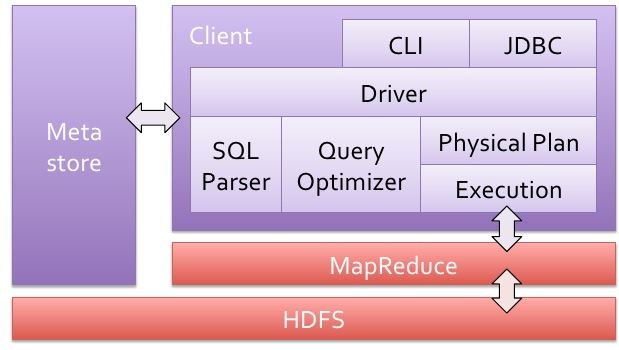
* Explain Hive Architecture in Brief.
* Explain Hive Components in Brief.

# Solution

## Hadoop Hive Architecture

Hive is one of the most important components of Hadoop, this document about Hadoop Hive Architecture.

**Hadoop Hive Architecture**

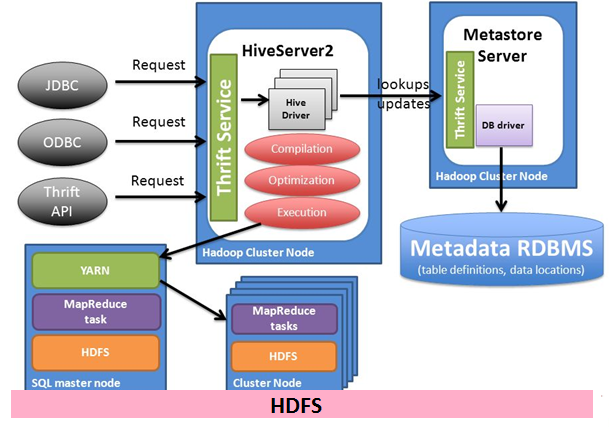


The above diagram shows the basic Hadoop Hive architecture. Primarily The diagram represents CLI (Command Line Interface),JDBC/ODBC and Web GUI (Web Graphical User Interface ).

This represents when user connects to the CLI(Hive Terminal) it directly connected to Hive Drivers, When User uses JDBC/ODBC(JDBC Program) this time using the API(Thrift Server) it connected to Hive driver and when the user connects with Web GUI (Ambari Server) it directly connected to Hive Driver.

The hive driver receives the tasks (Queries) from user and sends it to Hadoop architecture. The Hadoop architecture uses Namenode, Datanode, Node Manager and Resource Manager - Yarn for receiving and dividing the work what Hive sends to Hadoop (Mapreduce architecture).

**Detailed Hadoop Hive Architecture**



The following steps show how a typical query flows through the system.

Step 1 :- The UI calls the execute interface to the Driver

Step 2 :- The Driver creates a session handle for the query and sends the query to the compiler to generate an execution plan

Step 3&4 :- The compiler needs the metadata so send a request for getMetaData and receives the sendMetaData request from MetaStore.

Step 5 :- This metadata is used to type check the expressions in the query tree as well as to prune partitions based on query predicates. The plan generated by the compiler is a DAG of stages with each stage being either a map/reduce job, a metadata operation or an operation on HDFS. For map/reduce stages, the plan contains map operator trees (operator trees that are executed on the mappers) and a reduce operator tree (for operations that need reducers).

Step 6 :- The execution engine submits these stages to appropriate components (steps 6, 6.1, 6.2 and 6.3). In each task (mapper/reducer) the de-serializer associated with the table or intermediate outputs is used to read the rows from HDFS files and these are passed through the associated operator tree. Once the output generate it is written to a temporary HDFS file though the serializer. The temporary files are used to provide the subsequent map/reduce stages of the plan. For DML operations the final temporary file is moved to the table’s location

Step 7&8&9 :- For queries, the contents of the temporary file are read by the execution engine directly from HDFS as part of the fetch call from the Driver

## Major Components of Hive

**UI** :- UI means User Interface, The user interface for users to submit queries and other operations to the system.

**Driver** :- The Driver is used for receives the quires from UI .This component implements the notion of session handles and provides execute and fetch APIs modelled on JDBC/ODBC interfaces.

**Compiler** :- The component that parses the query, does semantic analysis on the different query blocks and query expressions and eventually generates an execution plan with the help of the table and partition metadata looked up from the metastore.

**MetaStore** :- The component that stores all the structure information of the various tables and partitions in the warehouse including column and column type information, the serializers and deserializers necessary to read and write data and the corresponding HDFS files where the data is stored.

**Execution Engine** :- The component which executes the execution plan created by the compiler. The plan is a DAG of stages. The execution engine manages the dependencies between these different stages of the plan and executes these stages on the appropriate system components.