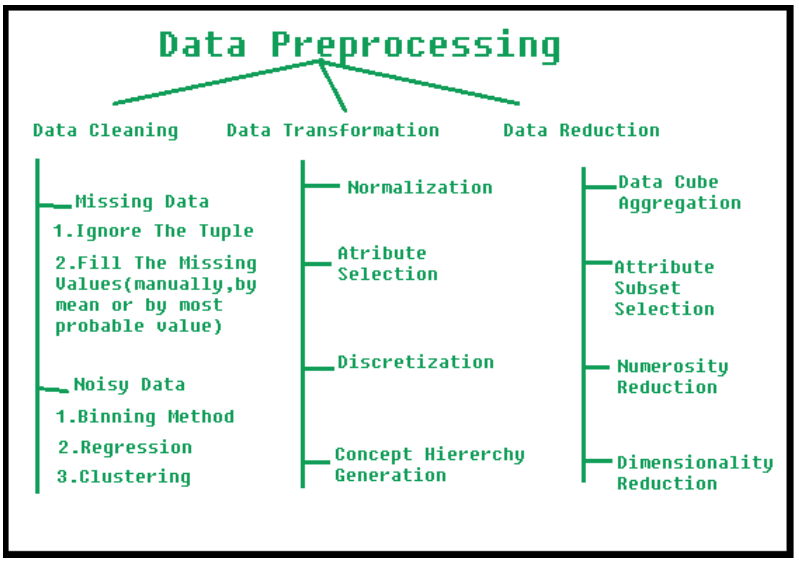
Assignment - 01 - Data Preprocessing

Data preprocessing is a step in the data mining and data analysis process that takes raw data and transforms it into a format that can be understood and analyzed by computers and machine learning.



1. Data Cleaning:

The data can have many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling of missing data, noisy data etc.

- **(a). Missing Data:**

This situation arises when some data is missing in the data. It can be handled in various ways. Some of them are:

1. **Ignore the tuples:**

This approach is suitable only when the dataset we have is quite large and multiple values are missing within a tuple.

2. **Fill the Missing values:**

There are various ways to do this task. You can choose to fill the missing values manually, by attribute mean or the most probable value.

- **(b). Noisy Data:**

Noisy data is a meaningless data that can't be interpreted by machines.It can be generated due to faulty data collection, data entry errors etc. It can be handled in following ways :

1. **Binning Method:**

This method works on sorted data in order to smooth it. The whole data is divided into segments of equal size and then various methods are performed to complete the task. Each segmented is handled separately. One can replace all data in a segment by its mean or boundary values can be used to complete the task.

2. **Regression:**

Here data can be made smooth by fitting it to a regression function.The regression used may be linear (having one independent variable) or multiple (having multiple independent variables).

3. **Clustering:**

This approach groups the similar data in a cluster. The outliers may be undetected or it will fall outside the clusters.

2. Data Transformation:

This step is taken in order to transform the data in appropriate forms suitable for mining process. This involves following ways:

1. **Normalization:**

It is done in order to scale the data values in a specified range (-1.0 to 1.0 or 0.0 to 1.0)

2. **Attribute Selection:**

In this strategy, new attributes are constructed from the given set of attributes to help the mining process.

3. **Discretization:**

This is done to replace the raw values of numeric attribute by interval levels or conceptual levels.

4. **Concept Hierarchy Generation:**

Here attributes are converted from lower level to higher level in hierarchy. For Example-The attribute “city” can be converted to “country”.

3. Data Reduction: Since data mining is a technique that is used to handle huge amount of data. While working with huge volume of data, analysis became harder in such cases. In order to get rid of this, we uses data reduction technique. It aims to increase the storage efficiency and reduce data storage and analysis costs.

The various steps to data reduction are:

1. **Data Cube Aggregation:**

Aggregation operation is applied to data for the construction of the data cube.

2. **Attribute Subset Selection:**

The highly relevant attributes should be used, rest all can be discarded. For performing attribute selection, one can use level of significance and p- value of the attribute.the attribute having p-value greater than significance level can be discarded.

3. **Numerosity Reduction:**

This enable to store the model of data instead of whole data, for example: Regression Models.

4. **Dimensionality Reduction:**

This reduce the size of data by encoding mechanisms.It can be lossy or lossless. If after reconstruction from compressed data, original data can be retrieved, such reduction are called lossless reduction else it is called lossy reduction. The two effective methods of dimensionality reduction are:Wavelet transforms and PCA (Principal Component Analysis).

Assignment - 02 - Kmeans Algo

→ K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science.

→ It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

→ It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

→It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

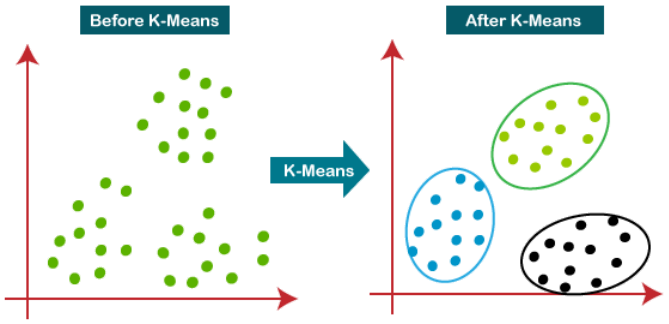
→ The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

→ The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be different from the input dataset).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

Step-5: Repeat the third step, which means re-assigning each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

Step-7: The model is ready.

Assignment - 03 - Installation Hadoop

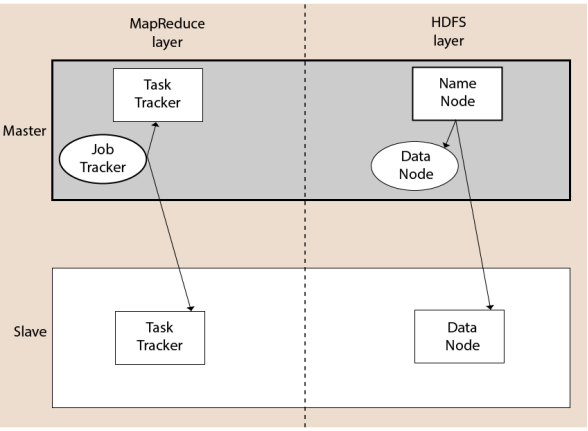
Hadoop is an open source framework from Apache and is used to store process and analyze data which are very huge in volume. Hadoop is written in Java and is not OLAP (online analytical processing). It is used for batch/offline processing.It is being used by Facebook, Yahoo, Google, Twitter, LinkedIn and many more. Moreover it can be scaled up just by adding nodes in the cluster.

Modules of Hadoop

1. **HDFS:** Hadoop Distributed File System. Google published its paper GFS and on the basis of that HDFS was developed. It states that the files will be broken into blocks and stored in nodes over the distributed architecture.
2. **Yarn:** Yet another Resource Negotiator is used for job scheduling and manage the cluster.
3. **Map Reduce:** This is a framework which helps Java programs to do the parallel computation on data using key value pair. The Map task takes input data and converts it into a data set which can be computed in Key value pair. The output of Map task is consumed by reduce task and then the out of reducer gives the desired result.
4. **Hadoop Common:** These Java libraries are used to start Hadoop and are used by other Hadoop modules.

Hadoop Architecture

The Hadoop architecture is a package of the file system, MapReduce engine and the HDFS (Hadoop Distributed File System). The MapReduce engine can be MapReduce/MR1 or YARN/MR2. A Hadoop cluster consists of a single master and multiple slave nodes. The master node includes Job Tracker, Task Tracker, NameNode, and DataNode whereas the slave node includes DataNode and TaskTracker.

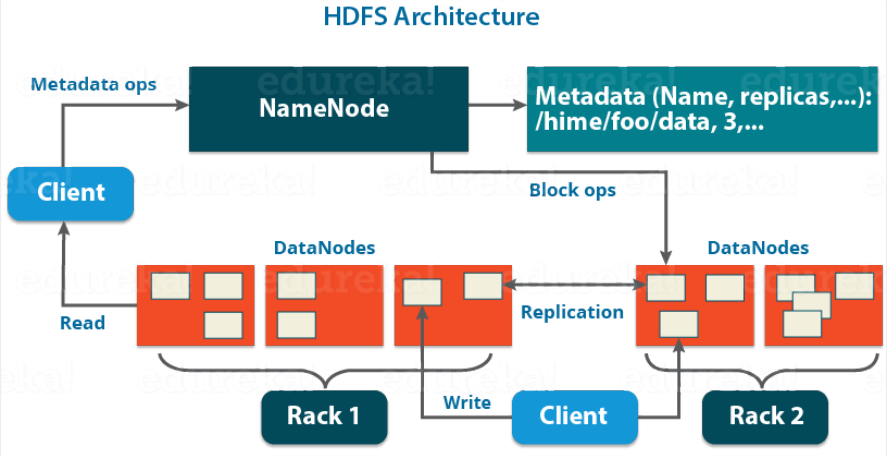


Advantages of Hadoop

- **Fast:** In HDFS the data distributed over the cluster and are mapped which helps in faster retrieval. Even the tools to process the data are often on the same servers, thus reducing the processing time. It is able to process terabytes of data in minutes and Peta bytes in hours.
- **Scalable:** Hadoop cluster can be extended by just adding nodes in the cluster.
- **Cost Effective:** Hadoop is open source and uses commodity hardware to store data so it really cost effective as compared to traditional relational database management system.
- **Resilient to failure:** HDFS has the property with which it can replicate data over the network, so if one node is down or some other network failure happens, then Hadoop takes the other copy of data and use it. Normally, data are replicated thrice but the replication factor is configurable.

Assignment - 04 - File Management in Hadoop

HDFS (Hadoop Distributed File System) is the primary storage system used by Hadoop applications. This open source framework works by rapidly transferring data between nodes. It's often used by companies who need to handle and store big data. HDFS is a key component of many Hadoop systems, as it provides a means for managing big data, as well as supporting big data analytics.



A solution would be to store the data across a network of machines. Such filesystems are called *distributed filesystems*. Since data is stored across a network all the complications of a network come in.

This is where Hadoop comes in. It provides one of the most reliable filesystems. HDFS (Hadoop Distributed File System) is a unique design that provides storage for *extremely large files* with streaming data access pattern and it runs on *commodity hardware*. Let's elaborate the terms:

- ***Extremely large files***: Here we are talking about the data in range of petabytes(1000 TB).
- ***Streaming Data Access Pattern***: HDFS is designed on principle of *write-once and read-many-times*. Once data is written large portions of dataset can be processed any number times.
- ***Commodity hardware:*** Hardware that is inexpensive and easily available in the market. This is one of feature which specially distinguishes HDFS from other file system.

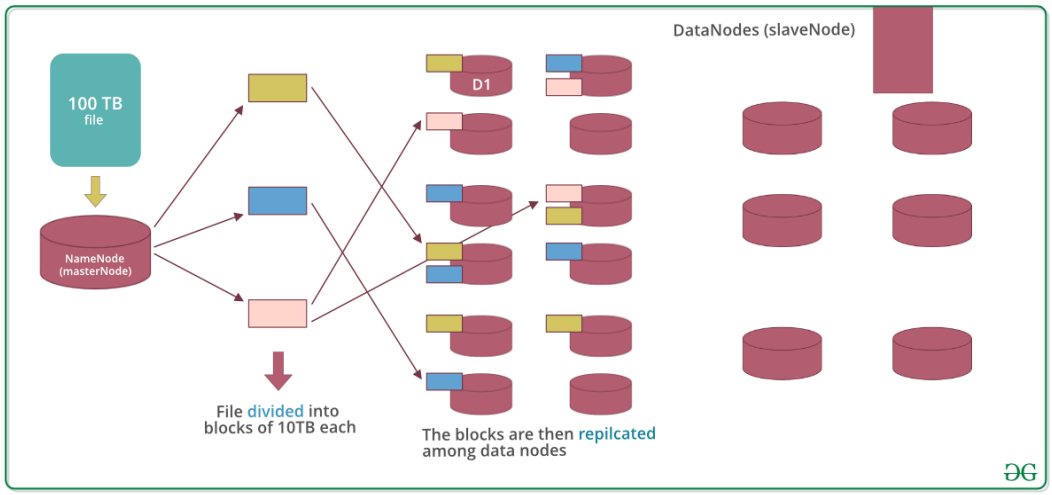
Nodes: Master-slave nodes typically forms the HDFS cluster.

1. **NameNode(MasterNode):**
    - Manages all the slave nodes and assign work to them.
    - It executes filesystem namespace operations like opening, closing, renaming files and directories.
    - It should be deployed on reliable hardware which has the high config. not on commodity hardware.
2. **DataNode(SlaveNode):**
    - Actual worker nodes, who do the actual work like reading, writing, processing etc.
    - They also perform creation, deletion, and replication upon instruction from the master.
    - They can be deployed on commodity hardware.

HDFS daemons: Daemons are the processes running in background.

- **Namenodes:**
    - Run on the master node.
    - Store metadata (data about data) like file path, the number of blocks, block Ids. etc.
    - Require high amount of RAM.
    - Store meta-data in RAM for fast retrieval i.e to reduce seek time. Though a persistent copy of it is kept on disk.
- **DataNodes:**
    - Run on slave nodes.
    - Require high memory as data is actually stored here.

Data storage in HDFS: Now let's see how the data is stored in a distributed manner.



Lets assume that 100TB file is inserted, then masternode(namenode) will first *divide* the file into blocks of 10TB (default size is *128 MB* in Hadoop 2.x and above). Then these blocks are stored across different datanodes(slavenode). Datanodes(slavenode)*replicate* the blocks among themselves and the information of what blocks they contain is sent to the master. Default replication factor is *3* means for each block 3 replicas are created (including itself). In hdfs.site.xml we can increase or decrease the replication factor i.e we can edit its configuration here. **Note:** MasterNode has the record of everything, it knows the location and info of each and every single data nodes and the blocks they contain, i.e. nothing is done without the permission of masternode.

Why divide the file into blocks? *Answer:* Let's assume that we don't divide, now it's very difficult to store a 100 TB file on a single machine. Even if we store, then each read and write operation on that *whole file* is going to take very high seek time. But if we have multiple blocks of size 128MB then its become easy to perform various read and write operations on it compared to doing it on a whole file at once. So we divide the file to have faster data access i.e. reduce seek time.

Why replicate the blocks in data nodes while storing? *Answer:* Let's assume we don't replicate and only one yellow block is present on datanode D1. Now if the data node D1 crashes we will lose the block and which will make the overall data inconsistent and faulty. So we replicate the blocks to achieve *fault-tolerance.*

Terms related to HDFS:

- ***HeartBeat*** : It is the signal that datanode continuously sends to namenode. If namenode doesn't receive heartbeat from a datanode then it will consider it dead.
- ***Balancing*** : If a datanode is crashed the blocks present on it will be gone too and the blocks will be *under-replicated* compared to the remaining blocks. Here master node(namenode) will give a signal to datanodes containing replicas of those lost blocks to replicate so that overall distribution of blocks is balanced.
- ***Replication:***: It is done by datanode.

Note: No two replicas of the same block are present on the same datanode.

Features:

- Distributed data storage.
- Blocks reduce seek time.
- The data is highly available as the same block is present at multiple datanodes.
- Even if multiple datanodes are down we can still do our work, thus making it highly reliable.
- High fault tolerance.

Limitations: Though HDFS provide many features there are some areas where it doesn't work well.

- **Low latency data access**: Applications that require low-latency access to data i.e in the range of milliseconds will not work well with HDFS, because HDFS is designed keeping in mind that we need high-throughput of data even at the cost of latency.
- **Small file problem**: Having lots of small files will result in lots of seeks and lots of movement from one datanode to another datanode to retrieve each small file, this whole process is a very inefficient data access pattern.

Command	Description
-rm	Removes file or directory
-ls	Lists files with permissions and other details
-mkdir	Creates a directory named path in HDFS
-cat	Shows contents of the file
-rmdir	Deletes a directory
-put	Uploads a file or folder from a local disk to HDFS
-rmr	Deletes the file identified by path or folder and subfolders
-get	Moves file or folder from HDFS to local file
-count	Counts number of files, number of directory, and file size
-df	Shows free space
-getmerge	Merges multiple files in HDFS

-chmod	Changes file permissions
-copyToLocal	Copies files to the local system
-Stat	Prints statistics about the file or directory
-head	Displays the first kilobyte of a file
-usage	Returns the help for an individual command
-chown	Allocates a new owner and group of a file

Assignment - 05 - Map Reduce in Hadoop

MapReduce is a programming paradigm that enables massive scalability across hundreds or thousands of servers in a Hadoop cluster. As the processing component, MapReduce is the heart of Apache Hadoop. The term "MapReduce" refers to two separate and distinct tasks that Hadoop programs perform. The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs).

The reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce job is always performed after the map job.

MapReduce programming offers several benefits to help you gain valuable insights from your big data:

- Scalability. Businesses can process petabytes of data stored in the Hadoop Distributed File System (HDFS).
- Flexibility. Hadoop enables easier access to multiple sources of data and multiple types of data.
- Speed. With parallel processing and minimal data movement, Hadoop offers fast processing of massive amounts of data.
- Simple. Developers can write code in a choice of languages, including Java, C++ and Python.

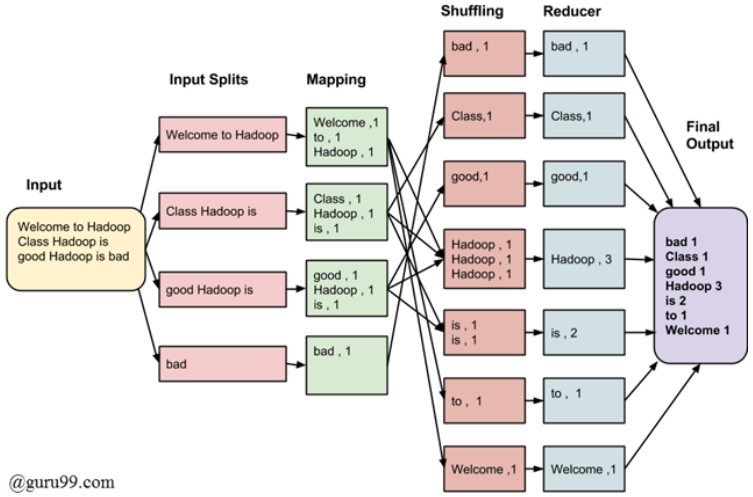
MapReduce Architecture in Big Data explained with Example

The whole process goes through four phases of execution namely, splitting, mapping, shuffling, and reducing.

Now in this MapReduce tutorial, let's understand with a MapReduce example–

Consider you have following input data for your MapReduce in Big data Program

```
Welcome to Hadoop Class
Hadoop is good
Hadoop is bad
```



MapReduce Architecture

The final output of the MapReduce task is

bad	1
Class	1
good	1
Hadoop	3
is	2
to	1
Welcome	1

The data goes through the following phases of MapReduce in Big Data

Input Splits:

An input to a MapReduce in Big Data job is divided into fixed-size pieces called **input splits** Input split is a chunk of the input that is consumed by a single map

Mapping

This is the very first phase in the execution of the map-reduce program. In this phase data in each split is passed to a mapping function to produce output values. In our example, a job of mapping phase is to count a number of occurrences of each word from input splits (more details about input-split is given below) and prepare a list in the form of <word, frequency>

Shuffling

This phase consumes the output of the Mapping phase. Its task is to consolidate the relevant records from Mapping phase output. In our example, the same words are clubbed together along with their respective frequency.

Reducing

In this phase, output values from the Shuffling phase are aggregated. This phase combines values from the Shuffling phase and returns a single output value. In short, this phase summarises the complete dataset.

In our example, this phase aggregates the values from the Shuffling phase i.e., calculates total occurrences of each word.

Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It resides on top of Hadoop to summarize Big Data, and makes querying and analyzing easy.

Initially Hive was developed by Facebook, later the Apache Software Foundation took it up and developed it further as an open source under the name Apache Hive. It is used by different companies. For example, Amazon uses it in Amazon Elastic MapReduce.

Hive is not

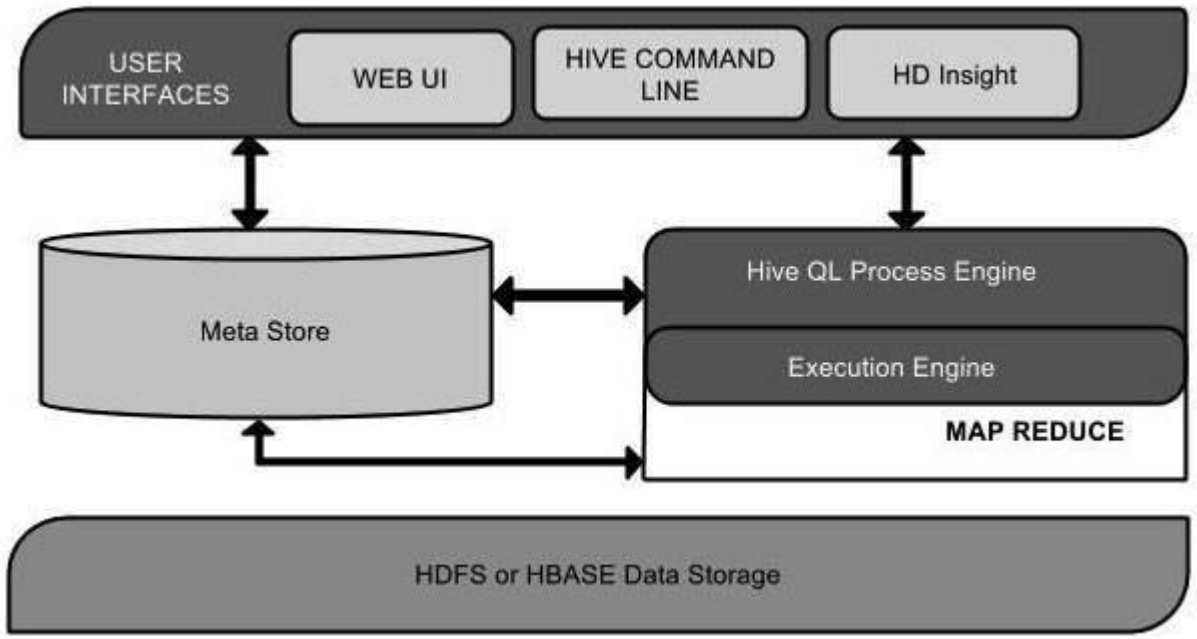
- A relational database
- A design for OnLine Transaction Processing (OLTP)
- A language for real-time queries and row-level updates

Features of Hive

- It stores schema in a database and processed data into HDFS.
- It is designed for OLAP.
- It provides SQL type language for querying called HiveQL or HQL.
- It is familiar, fast, scalable, and extensible.

Architecture of Hive

The following component diagram depicts the architecture of Hive:



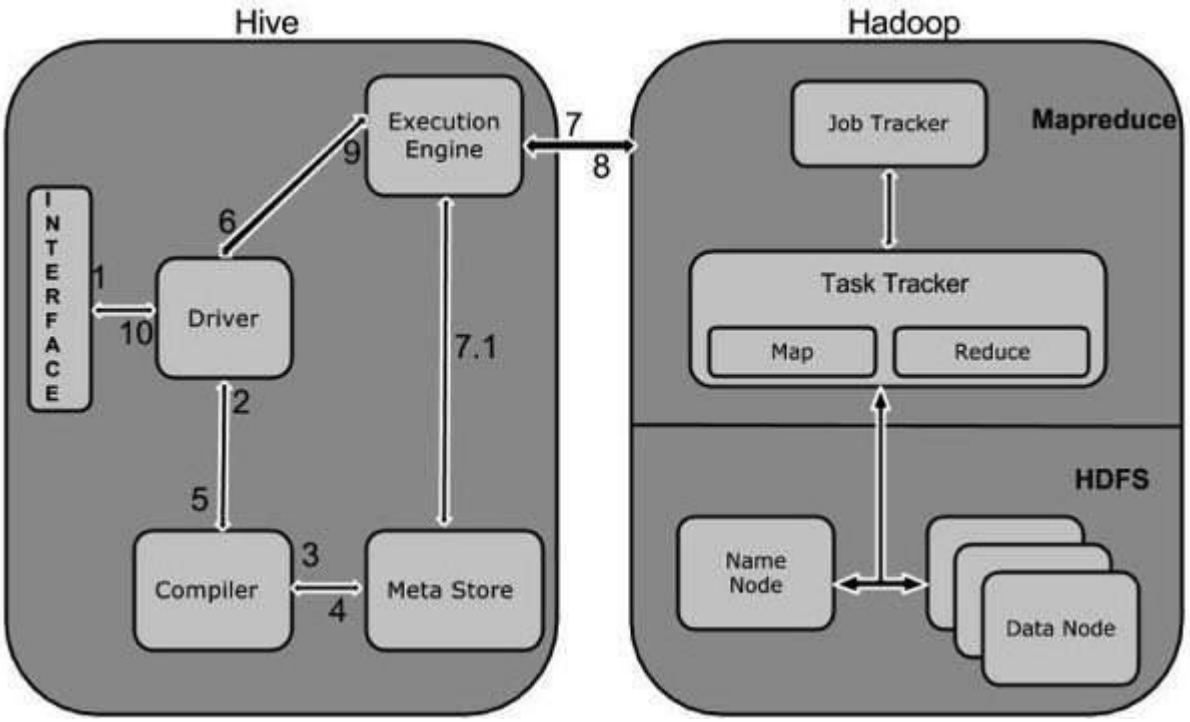
This component diagram contains different units. The following table describes each unit:

Unit Name	Operation

User Interface	Hive is a data warehouse infrastructure software that can create interaction between user and HDFS. The user interfaces that Hive supports are Hive Web UI, Hive command line, and Hive HD Insight (In Windows server).
Meta Store	Hive chooses respective database servers to store the schema or Metadata of tables, databases, columns in a table, their data types, and HDFS mapping.
HiveQL Process Engine	HiveQL is similar to SQL for querying on schema info on the Metastore. It is one of the replacements of traditional approach for MapReduce program. Instead of writing MapReduce program in Java, we can write a query for MapReduce job and process it.
Execution Engine	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.
HDFS or HBASE	Hadoop distributed file system or HBASE are the data storage techniques to store data into file system.

Working of Hive

The following diagram depicts the workflow between Hive and Hadoop.



The following table defines how Hive interacts with Hadoop framework:

Step No.	Operation
1	Execute Query The Hive interface such as Command Line or Web UI sends query to Driver (any database driver such as JDBC, ODBC, etc.) to execute.
2	Get Plan The driver takes the help of query compiler that parses the query to check the syntax and query plan or the requirement of query.
3	Get Metadata The compiler sends metadata request to Metastore (any database).
4	Send Metadata Metastore sends metadata as a response to the compiler.

5	Send Plan The compiler checks the requirement and resends the plan to the driver. Up to here, the parsing and compiling of a query is complete.
6	Execute Plan The driver sends the execute plan to the execution engine.
7	Execute Job Internally, the process of execution job is a MapReduce job. The execution engine sends the job to JobTracker, which is in Name node and it assigns this job to TaskTracker, which is in Data node. Here, the query executes MapReduce job.
7.1	Metadata Ops Meanwhile in execution, the execution engine can execute metadata operations with Metastore.
8	Fetch Result The execution engine receives the results from Data nodes.
9	Send Results The execution engine sends those resultant values to the driver.
10	Send Results The driver sends the results to Hive Interfaces.

Create Database Statement

Create Database is a statement used to create a database in Hive. A database in Hive is a namespace or a collection of tables. The syntax for this statement is as follows:

```
CREATE DATABASE|SCHEMA [IF NOT EXISTS] <database name>
```

Here, IF NOT EXISTS is an optional clause, which notifies the user that a database with the same name already exists. We can use SCHEMA in place of DATABASE in this command. The following query is executed to create a database named userdb:

```
hive> CREATE DATABASE [IF NOT EXISTS] userdb;
```

or

```
hive> CREATE SCHEMA userdb;
```

The following query is used to verify a databases list:

```
hive> SHOW DATABASES;
default
userdb
```

Drop Database Statement

Drop Database is a statement that drops all the tables and deletes the database. Its syntax is as follows:

```
DROP DATABASE StatementDROP (DATABASE|SCHEMA) [IF EXISTS] database_name
[RESTRICT|CASCADE];
```

The following queries are used to drop a database. Let us assume that the database name is userdb.

```
hive> DROP DATABASE IF EXISTS userdb;
```

The following query drops the database using CASCADE. It means dropping respective tables before dropping the database.

```
hive> DROP DATABASE IF EXISTS userdb CASCADE;
```

The following query drops the database using SCHEMA.

```
hive> DROP SCHEMA userdb;
```

Create Table Statement

Create Table is a statement used to create a table in Hive. The syntax and example are as follows:

Syntax

```
CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.] table_name

[(col_name data_type [COMMENT col_comment], ...)]
[COMMENT table_comment]
[ROW FORMAT row_format]
[STORED AS file_format]
```

Example

Let us assume you need to create a table named employee using CREATE TABLE statement. The following table lists the fields and their data types in employee table:

Sr.No	Field Name	Data Type
	e	g
	ry	t
	gnation	g

The following data is a Comment, Row formatted fields such as Field terminator, Lines terminator, and Stored File type.

```
COMMENT 'Employee details'
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
STORED IN TEXT FILE
```

The following query creates a table named employee using the above data.

```
hive> CREATE TABLE IF NOT EXISTS employee ( eid int, name String,
salary String, destination String)
```

```
COMMENT 'Employee details'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

If you add the option IF NOT EXISTS, Hive ignores the statement in case the table already exists.

On successful creation of table, you get to see the following response:

```
OK
Time taken: 5.905 seconds
hive>
```

Alter Table Statement

It is used to alter a table in Hive.

Syntax

The statement takes any of the following syntaxes based on what attributes we wish to modify in a 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
ALTER TABLE name REPLACE COLUMNS (col_spec[, col_spec ...])
```

Rename To… Statement

The following query renames the table from employee to emp.

```
hive> ALTER TABLE employee RENAME TO emp;
```

Drop Table Statement

The syntax is as follows:

```
DROP TABLE [IF EXISTS] table_name;
```

The following query drops a table named employee:

```
hive> DROP TABLE IF EXISTS employee;
```

On successful execution of the query, you get to see the following response:

```
OK
Time taken: 5.3 seconds
hive>
```