NAGARJUNA COLLEGE OF ENGINEERING AND TECHNOLOGY

Department of CSE (Data Science)

Big Data Analytics Lab(21CDL66)

PREPARED BY
Mrs. Geetha M, MTech,(PhD)
ASSISTANT PROFESSORDEPARTMENT OF CSE (DATA SCIENCE)

1. Installation of Hadoop Framework, it‘s components and study the HADOOPecosystem

Aim: Installation of Hadoop Framework, it‘s components and study the HADOOPecosystem

Hadoop is an open-source framework that allows to store and process big data in a distributedenvironment across clusters of computers using simple programming models. It is designed toscale up from single servers to thousands of machines, each offering localcomputation andstorage.

Hadoop Architecture:

The Apache Hadoop framework includes following four modules:

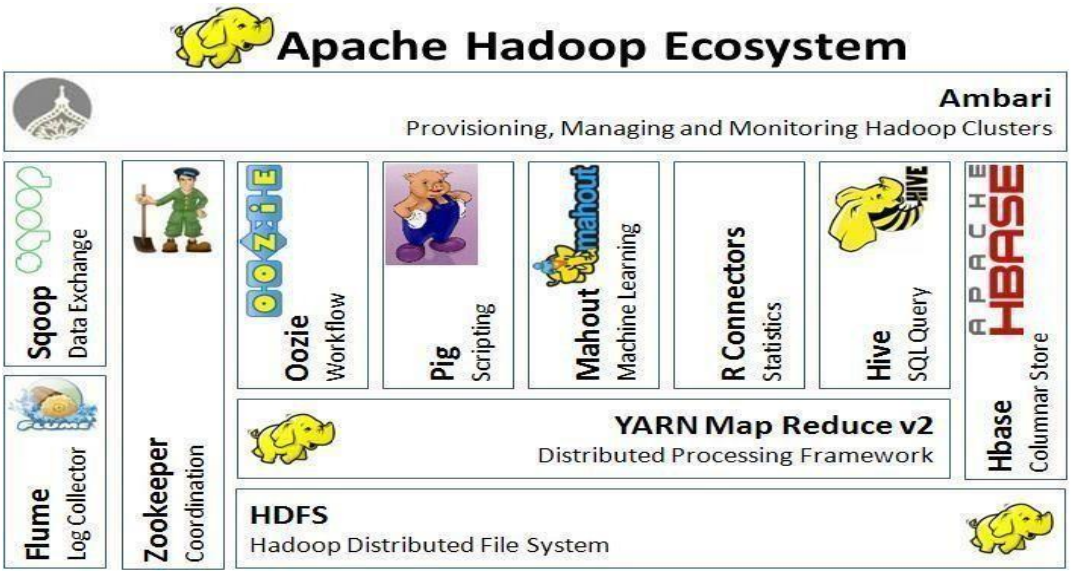
Hadoop Common: Contains Java libraries and utilities needed by other Hadoop modules. These libraries give file system and OS level abstraction and comprise of the essential Java files and scripts that are required to start Hadoop.

Hadoop Distributed File System (HDFS): A distributed file-system that provides high- throughput access to application data on the community machines thus providing very high aggregate bandwidth across the cluster.

Hadoop YARN: A resource-management framework responsible for job scheduling and cluster resource management.

Hadoop MapReduce: This is a YARN- based programming model for parallel processing oflarge data sets.

Hadoop Ecosystem:



Hadoop has gained its popularity due to its ability of storing, analyzing and accessing large amount of data, quickly and cost effectively through clusters of commodity hardware. It won‘tbe wrong if we say that Apache Hadoop is actually a collection of several components and notjust a single product.

With Hadoop Ecosystem there are several commercial along with an open source products which are broadly used to make Hadoop laymen accessible and more usable.

MapReduce

Hadoop MapReduce is a software framework for easily writing applications which process bigamounts of data in-parallel on large clusters of commodity hardware in a reliable, fault- tolerantmanner. In terms of programming, there are **two functions** which are most common in MapReduce.

- **The Map Task:** Master computer or node takes input and convert it into divide it into smallerparts and distribute it on other worker nodes. All worker nodes solve their own small problemand give answer to the master node.
- **The Reduce Task:** Master node combines all answers coming from worker node andformsit in some form of output which is answer of our big distributed problem.

Generally both the input and the output are reserved in a file-system. The framework is responsible for scheduling tasks, monitoring them and even re-executes the failed tasks.

Hadoop Distributed File System (HDFS)

HDFS is a distributed file-system that provides high throughput access to data. When data ispushed to HDFS, it automatically splits up into multiple blocks and stores/replicates the datathus ensuring high availability and fault tolerance.

Note: A file consists of many blocks (large blocks of 64MB and above).

Here are the **main components of HDFS:**

- **Name Node:** It acts as the master of the system. It maintains the name systemi.e.,directoriesand files and manages the blocks which are present on the Data Nodes.
- **Data Nodes:** They are the slaves which are deployed on each machine and provide theactual storage. They are responsible for serving read and write requests for the clients.
- **Secondary Name Node:** It is responsible for performing periodic checkpoints. In theevent of Name Node failure, you can restart the Name Node using the checkpoint.

Hive Click on New and add the bin directory path of Hadoop and Java in it.

Hive is part of the Hadoop ecosystem and provides an SQL like interface to Hadoop. It is a data warehouse system for Hadoop that facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in Hadoop compatible file systems.

HBase (Hadoop DataBase)

HBase is a distributed, column oriented database and uses HDFS for the underlying storage. As said earlier, HDFS works on write once and read many times pattern, but this isn‘t a casealways. We may require real time read/write random access for huge dataset; this is where HBase comes into the picture. HBase is built on top of HDFS and distributed on column- oriented database.

Hadoop is powerful because it is extensible and it is easy to integrate with any component. Itspopularity is due in part to its ability to store, analyze and access large amounts of data, quicklyand cost effectively across clusters of commodity hardware. Apache Hadoop is not actually asingle product but instead a collection of several components. When all these components are merged, it makes the Hadoop very user friendly.

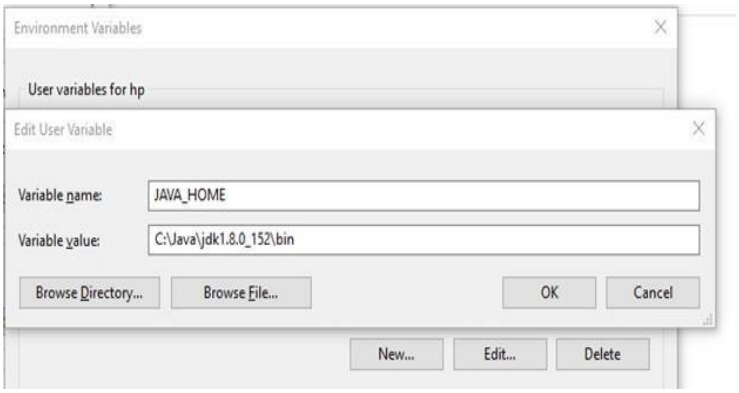
PREREQUISITE: TO INSTALL HADOOP, YOU SHOULD HAVE JAVA VERSION 1.8 IN YOUR SYSTEM.

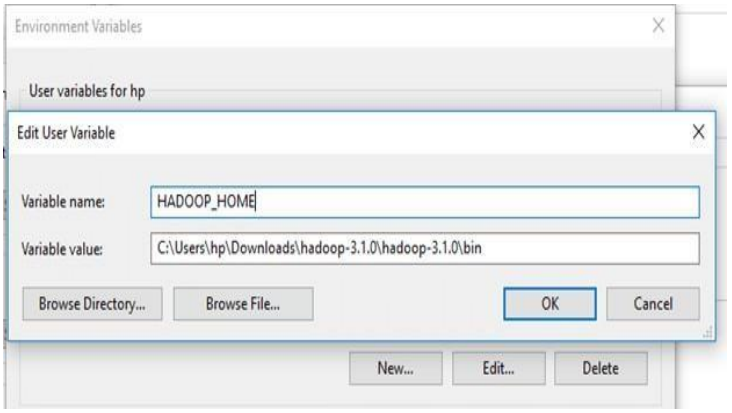
Check your java version through this command on command prompt

Java -version

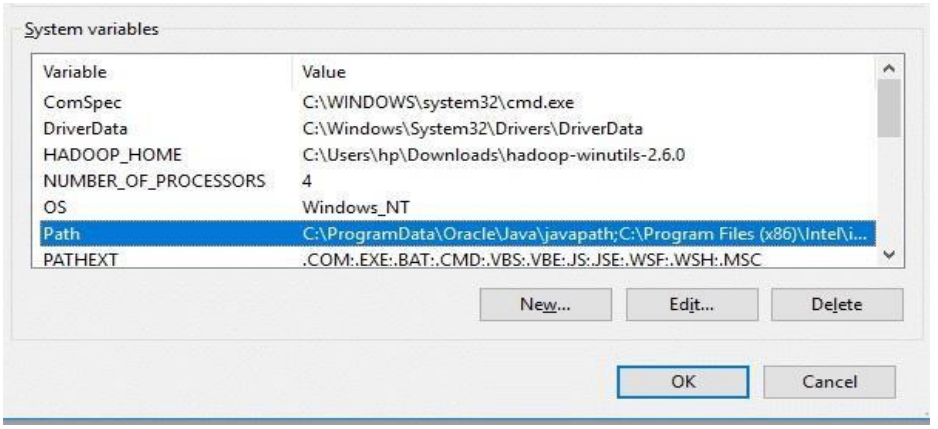
Create a new user variable. Put the Variable_name as HADOOP_HOME and Variable_value as the path of the bin folder where you extracted hadoop.

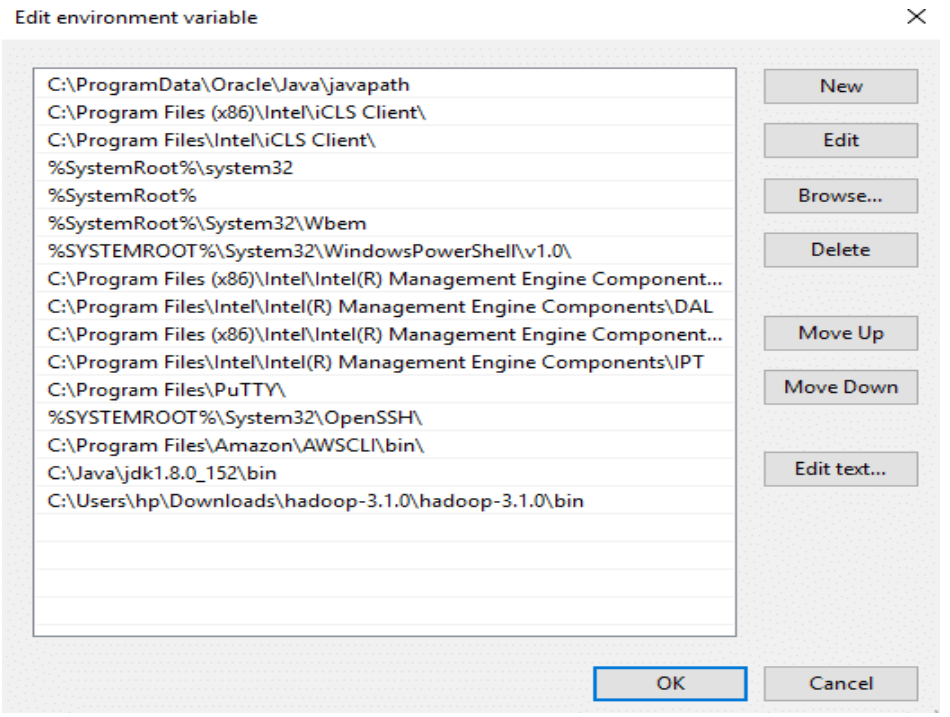
Likewise, create a new user variable with variable name as JAVA_HOME and variable value as the path of the bin folder in the Java directory.





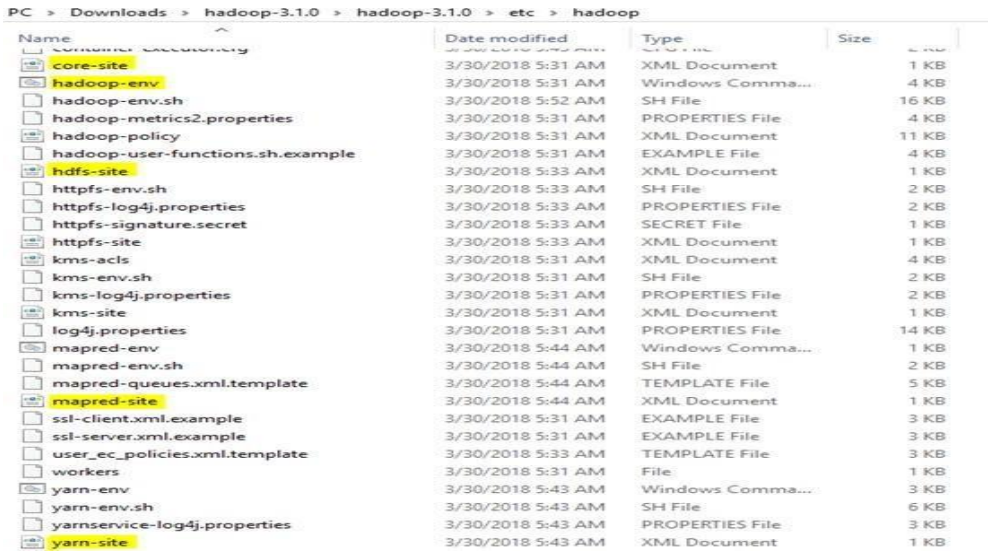
Now we need to set Hadoop bin directory and Java bin directory path in system variable path.Edit Path in system variable.





GUI CONFIGURATIONS

Now we need to edit some files located in the hadoop directory of the etc folder where we installed hadoop. The files that need to be edited have been highlighted.



1. **Edit core site.xml file and copy this property in the configuration.**

   <configuration>

   <property>

   <name>fs.defaultFS</name>

   <value>hdfs://localhost:9000</value>

   </property>

   </configuration>

2. Edit mapred-site.xml and copy this property in the configuration

   <configuration>

   <property>

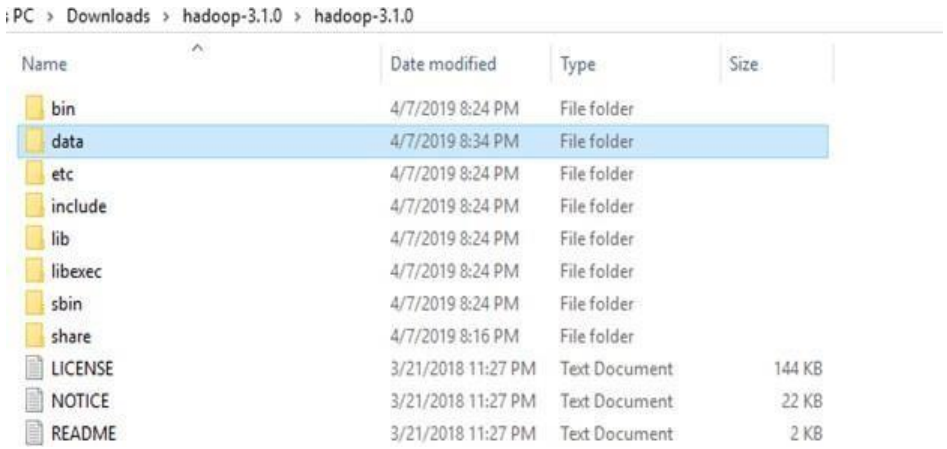
   <name>mapreduce.framework.name</name>

   <value>yarn</value>

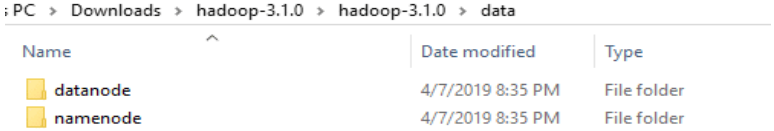
   </property>

   </configuration>

3. **Create a folder 'data' in the hadoop directory**



   Create a folder with the name 'datanode' and a folder 'namenode' inthis data directory



4. **Edit the file hdfs-site.xml and add below property in the configuration**

Note: The path of namenode and datanode across value would be the path of the datanodeand namenode folders you just created.

<configuration>

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

<property>

<name>dfs.namenode.name.dir</name>

<value>C:\Users\hp\Downloads\hadoop-3.1.0\hadoop-3.1.0\data\namenode</value>

</property>

<property>

<name>dfs.datanode.data.dir</name>

<value> C:\Users\hp\Downloads\hadoop-3.1.0\hadoop-3.1.0\data\datanode</value>

</property>

</configuration>

5. **Edit the file yarn-site.xml and add below property in the configuration**

<configuration>
<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce_shuffle</value>

</property>

<property>

<name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>

<value>org.apache.hadoop.mapred.ShuffleHandler</value>

</property>

</configuration>

6. **Edit hadoop-env.cmd and replace %JAVA_HOME% with the path of thejava folder where your jdk 1.8 is installed**

hadoop-env - Notepad

File Edit Format View Help

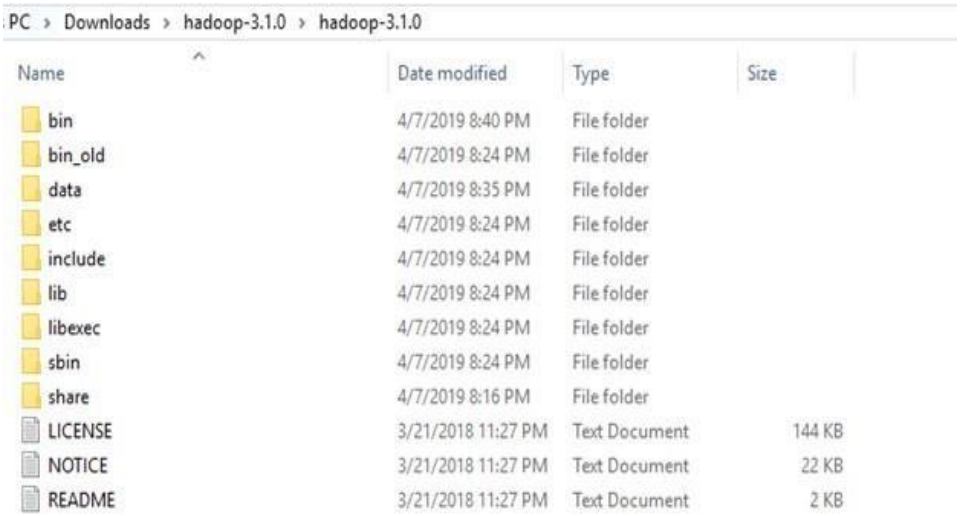
```
@rem Set Hadoop-specific environment variables here.

@rem The only required environment variable is JAVA_HOME.  All others are
@rem optional.  When running a distributed configuration it is best to
@rem set JAVA_HOME in this file, so that it is correctly defined on
@rem remote nodes.

@rem The java implementation to use.  Required.
set JAVA_HOME=C:\Java\jdk1.8.0_152

@rem The jsvc implementation to use. Jsvc is required to run secure datanodes.
@rem set JSVC_HOME=%JSVC_HOME%

@rem set HADOOP_CONF_DIR=

@rem Extra Java CLASSPATH elements.  Automatically insert capacity-scheduler.
if exist %HADOOP_HOME%\contrib\capacity-scheduler (
  if not defined HADOOP_CLASSPATH (
    set HADOOP_CLASSPATH=%HADOOP_HOME%\contrib\capacity-scheduler\*.jar
  ) else (
    set HADOOP_CLASSPATH=%HADOOP_CLASSPATH%;%HADOOP_HOME%\contrib\capacity-scheduler\*.jar
  )
)
```

7. **Hadoop needs windows OS specific files which does not come withdefault download of hadoop.**

   To include those files, replace the bin folder in hadoop directory withthe bin folder provided in this github link.

   https://github.com/s911415/apache-hadoop-3.1.0-winutils

   Download it as zip file. Extract it and copy the bin folder in it. If you want to save the old bin folder, rename it like bin_old and paste the copied bin folder in that directory.



   Check whether hadoop is successfully installed by running this command on cmd.

   hadoop –version

Format the NameNode

Formatting the NameNode is done once when hadoop is installed and not for runninghadoop filesystem, else it will delete all the data inside HDFS.
Run this command

hdfs namenode –format

Now change the directory in cmd to sbin folder of hadoop directory with this command,Start

namenode and datanode with this command –

start-dfs.cmd

Two more cmd windows will open for NameNode and DataNode Now start yarn throughthis

command-

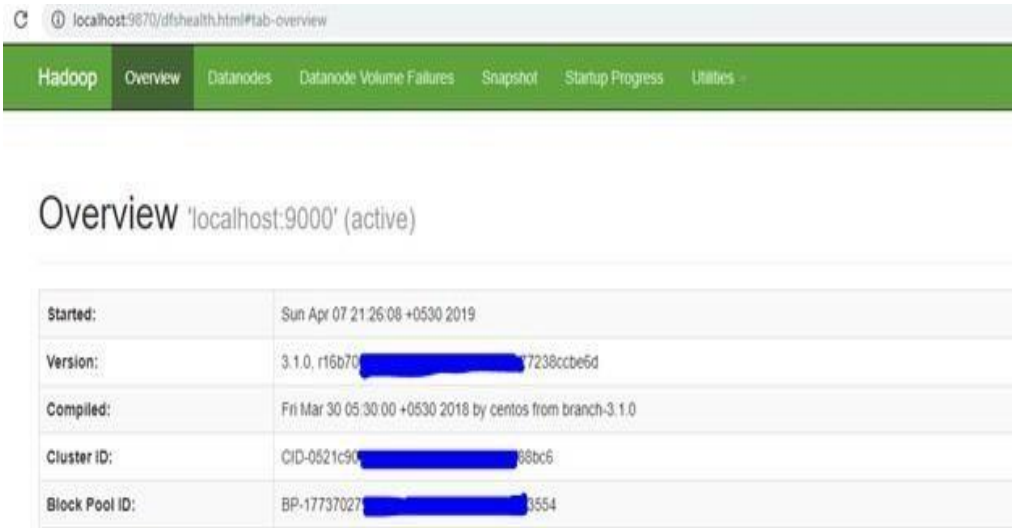
start-yarn.cmd

Note: Make sure all the 4 Apache Hadoop Distribution windows are up n running. If they are not running, you will see an error or a shutdown message. In that case, you needto debug the error.

To access information about resource manager current jobs, successful and failed jobs,go to this link in browser-

http://localhost:8088/cluster

To check the details about the hdfs (namenode and datanode),http://localhost:9870/



2. Hive: introduction, creation of database and table ,Hive Partition,Hive built in Function and Operators ,High View and Index.

Introduction

Hive **allows users to read, write, and manage petabytes of data using SQL**. Hive is built on top of Apache Hadoop, which is an open-source framework used to efficiently store and process large datasets. As a result, Hive is closely integrated with Hadoop, and is designed to work quickly on petabytes of data.

Hive is a data warehouse system which is used to analyze structured data. It is built on the top of Hadoop. It was developed by Facebook.

Hive provides the functionality of reading, writing, and managing large datasets residing in distributed storage. It runs SQL like queries called HQL (Hive query language) which gets internally converted to MapReduce jobs.

Using Hive, we can skip the requirement of the traditional approach of writing complex MapReduce programs. Hive supports Data Definition Language (DDL), Data Manipulation Language (DML), and User Defined Functions (UDF).

Features of Hive

- Hive is fast and scalable.
- It provides SQL-like queries (i.e., HQL) that are implicitly transformed to MapReduce or Spark jobs.
- It is capable of analyzing large datasets stored in HDFS.
- It allows different storage types such as plain text, RCFile, and HBase.
- It uses indexing to accelerate queries.
- It can operate on compressed data stored in the Hadoop ecosystem.
- It supports user-defined functions (UDFs) where user can provide its functionality.

Limitations of Hive

- Hive is not capable of handling real-time data.
- It is not designed for online transaction processing.
- Hive queries contain high latency.
- Hive is a database technology that can define databases and tables to analyze structured data. The theme for structured data analysis is to store the data in a tabular manner, and pass queries to analyze it. Create Database Statement

- **Creation of database**

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
```

Hive - Partitioning

Hive organizes tables into partitions. It is a way of dividing a table into related parts based on the values of partitioned columns such as date, city and department. Using partition, it is easy to query a portion of the data.

Tables or partitions are sub-divided into **buckets,** to provide extra structure to the data that may be used for more efficient querying. Bucketing works based on the value of hash function of some column of a table.

For example, a table named **Tab1** contains employee data such as id, name, dept, and yoj (i.e., year of joining). Suppose you need to retrieve the details of all employees who joined in 2012. A query searches the whole table for the required information. However, if you partition the employee data with the year and store it in a separate file, it reduces the query processing time.

The following example shows how to partition a file and its data:

The following file contains employeedata table.

/tab1/employeedata/file1 id, name, dept, yoj

1, gopal, TP, 2012

2, kiran, HR, 2012

3, kaleel,SC, 2013

4, Prasanth, SC, 2013

The above data is partitioned into two files using year.

/tab1/employeedata/2012/file2 1, gopal, TP, 2012

2, kiran, HR, 2012

/tab1/employeedata/2013/file3 3, kaleel,SC, 2013

Hive - Built-in Functions

Return Type	Signature	Description
BIGINT	round(double a)	It returns the rounded BIGINT value of the double.
BIGINT	floor(double a)	It returns the maximum BIGINT value that is equal or less than the double.
BIGINT	ceil(double a)	It returns the minimum BIGINT value that is equal or greater than the double.
Double	rand(), rand(int seed)	It returns a random number that changes from row to row.
String	concat(string A, string B,...)	It returns the string resulting from concatenating B after A.
String	substr(string A, int start)	It returns the substring of A starting from start position till the end of string A.
String	substr(string A, int start, int length)	It returns the substring of A starting from start position with the given length.
String	upper(string A)	It returns the string resulting from converting all characters of A to upper case.
String	lower(string A)	It returns the string resulting from converting all characters of B to lower case.
String	trim(string A)	It returns the string resulting from trimming spaces from both ends of A.
String	ltrim(string A)	t returns the string resulting from trimming spaces from the beginning (left hand side) of A.
String	rtrim(string A)	rtrim(string A) It returns the string resulting from trimming spaces from the end (right hand side) of A.
String	regexp_replace(string A, string B, string C)	returns the string resulting from replacing all substrings in B that match the Java regular expression syntax with C.
Int	size(Map<K.V>)	It returns the number of elements in the map type.
Int	size(Array<T>)	It returns the number of elements in the array type.
Int	month(string date)	It returns the month part of a date or a timestamp string: month("1970-11-01 00:00:00") = 11, month("1970-11-01") = 11
Int	day(string date)	It returns the day part of a date or a timestamp string: day("1970- 11-01 00:00:00") = 1, day("1970-11-01") = 1
String	get_json_object(string json_string, string path)	t extracts json object from a json string based on json path specified, and returns son string of the extracted json object. It returns NULL if the input json string is invalid.

Hive - View and Indexes

Views are generated based on user requirements. You can save any result set data as a view. The usage of view in Hive is same as that of the view in SQL. It is a standard RDBMS concept. We can execute all DML operations on a view.

Creating a View

You can create a view at the time of executing a SELECT statement. The syntax is as follows:

```
CREATE VIEW [IF NOT EXISTS] view_name [(column_name
[COMMENT column_comment], ...) ]

[COMMENT table_comment]

AS SELECT ...
```

Example

Let us take an example for view. Assume employee table as given below, with the fields Id, Name, Salary, Designation, and Dept. Generate a query to retrieve the employee details who earn a salary of more than Rs 30000. We store the result in a view named **emp_30000.**

```
+                    ---- ---------- --------- ------------- ------
| ID | Name | Salary | Designation | Dept |
+                    ---- +--------- +-------- +------------ +-----+
|1201 | Gopal | 45000 | Technical manager | TP |
|1202 | Manisha | 45000 | Proofreader | PR |
|1203 | Masthanvali | 40000 | Technical writer | TP |
|1204 | Krian | 40000 | Hr Admin | HR |
|1205 | Kranthi | 30000 | Op Admin | Admin |
+                    ---- +--------- +-------- +------------ +-----+
```

```
hive> CREATE VIEW emp_30000 AS SELECT * FROM employee

WHERE salary>30000;
```

Creating an Index

An Index is nothing but a pointer on a particular column of a table. Creating an index means creating a pointer on a particular column of a table. Its syntax is as follows:

```
CREATE INDEX index_name

ON TABLE base_table_name (col_name, ...) AS 'index.handler.class.name'

[WITH DEFERRED REBUILD]

[IDXPROPERTIES (property_name=property_value, ...)] [IN TABLE index_table_name]

[PARTITIONED BY (col_name, ...)] [

[ ROW FORMAT ...] STORED AS ...

| STORED BY ...]

[LOCATION hdfs_path] [TBLPROPERTIES (...)]
```

Example

Let us take an example for index. Use the same employee table that we have used earlier with the fields Id, Name, Salary, Designation, and Dept. Create an index named index_salary on the salary column of the employee table.

```
hive> CREATE INDEX index_salary ON TABLE employee(salary) AS

'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler';
```


It is a pointer to the salary column. If the column is modified, the changes are stored using an index value.

3.Pig Latin Basic Shell ,Pig Data Types ,Creating a Pig Data Model, Reading and Storing Data ,Pig Operations.

Pig is an open-source high level data flow system. It provides a simple language called Pig Latin, for queries and data manipulation, which are then compiled in to MapReduce jobs that run on Hadoop.

Pig is important as companies like Yahoo, Google and Microsoft are collecting huge amounts of data sets in the form of click streams, search logs and web crawls. Pig is also used in some form of ad-hoc processing and analysis of all the information.

Why Do you Need Pig?

- It's easy to learn, especially if you're familiar with SQL.
- Pig's multi-query approach reduces the number of times data is scanned. This means 1/20th the lines of code and 1/16th the development time when compared to writing raw MapReduce.
- Performance of Pig is in par with raw MapReduce
- Pig provides data operations like filters, joins, ordering, etc. and nested data types like tuples, bags, and maps, that are missing from MapReduce.
- Pig Latin is easy to write and read.

Why was Pig Created?

Pig was originally developed by Yahoo in 2006, for researchers to have an ad-hoc way of creating and executing MapReduce jobs on very large data sets. It was created to reduce the development time through its multi-query approach. Pig is also created for professionals from non-Java background, to make their job easier.

Pig can be used under following scenarios:

- When data loads are time sensitive.
- When processing various data sources.
- When analytical insights are required through sampling.

Pig Latin – Basics

Pig Latin is the language used to analyze data in Hadoop using Apache Pig.

Pig Latin – Data Model

The data model of Pig is fully nested. A **Relation** is the outermost structure of the Pig Latin data model. And it is a **bag** where −

- A bag is a collection of tuples.

- A tuple is an ordered set of fields.
- A field is a piece of data.

Pig Latin – Statemets

While processing data using Pig Latin, **statements** are the basic constructs.

- These statements work with **relations**.
- They include **expressions** and **schemas**.
- Every statement ends with a semicolon (;).
- We will perform various operations using operators provided by Pig Latin, through statements.
- Except LOAD and STORE, while performing all other operations, Pig Latin statements take a relation as input and produce another relation as output.
- As soon as you enter a **Load** statement in the Grunt shell, its semantic checking will be carried out. To see the contents of the schema, you need to use the **Dump** operator. Only after performing the **dump** operation, the MapReduce job for loading the data into the file system will be carried out.

Pig Data Types

Apache Pig supports many data types. A list of Apache Pig Data Types with description and examples are given below.

Type	Description	Example
Int	Signed 32 bit integer	2
Long	Signed 64 bit integer	15L or 15l
Float	32 bit floating point	2.5f or 2.5F
Double	32 bit floating point	1.5 or 1.5e2 or 1.5E2
charArray	Character array	hello students
byteArray	BLOB(Byte array)	
Tuple	Ordered set of fields	(12,43)
Bag	Collection f tuples	{(12,43),(54,28)}
Map	collection of tuples	[open#apache]

Apache Pig - Reading Data

In general, Apache Pig works on top of Hadoop. It is an analytical tool that analyzes large datasets that exist in the Hadoop File System. To analyze data using Apache Pig, we have to initially load the data into Apache Pig. This chapter explains how to load data to Apache Pig from HDFS.

Preparing HDFS

In MapReduce mode, Pig reads (loads) data from HDFS and stores the results back in HDFS. Therefore, let us start HDFS and create the following sample data in HDFS.

Step 1: Verifying Hadoop

First of all, verify the installation using Hadoop version command, as shown below.

```
$ hadoop version
```

Step 2: Starting HDFS

Browse through the **sbin** directory of Hadoop and start **yarn** and Hadoop dfs (distributed file system) as shown below.

cd /$Hadoop_Home/sbin/

Step 3: Create a Directory in HDFS

In Hadoop DFS, you can create directories using the command **mkdir**. Create a new directory in HDFS with the name **Pig_Data** in the required path as shown below.

```
$cd /$Hadoop_Home/bin/

$ hdfs dfs -mkdir hdfs://localhost:9000/Pig_Data
```

Step 4: Placing the data in HDFS

The input file of Pig contains each tuple/record in individual lines. And the entities of the record are separated by a delimiter (In our example we used **“,”**).

In the local file system, create an input file **student_data.txt** containing data as shown below.
001,Rajiv,Reddy,9848022337,Hyderabad 002,siddarth,Battacharya,9848022338,Kolkata

003,Rajesh,Khanna,9848022339,Delhi

004,Preethi,Agarwal,9848022330,Pune

005,Trupthi,Mohanthy,9848022336,Bhuwaneshwar
006,Archana,Mishra,9848022335,Chennai.

Now, move the file from the local file system to HDFS using **put** command as shown below. (You can use **copyFromLocal** command as well.)

```
$ cd $HADOOP_HOME/bin

$ hdfs dfs -put /home/Hadoop/Pig/Pig_Data/student_data.txt
dfs://localhost:9000/pig_data/
```


Verifying the file

You can use the **cat** command to verify whether the file has been moved into the HDFS, as shown below.

```
$ cd $HADOOP_HOME/bin

$ hdfs dfs -cat hdfs://localhost:9000/pig_data/student_data.txt
```

001,Rajiv,Reddy,9848022337,Hyderabad 002,siddarth,Battacharya,9848022338,Kolkata

003,Rajesh,Khanna,9848022339,Delhi 004,Preethi,Agarwal,9848022330,Pune

005,Trupthi,Mohanthy,9848022336,Bhuwaneshwar 006,Archana,Mishra,9848022335,Chennai

Apache Pig - Storing Data

You can store the loaded data in the file system using the **store** operator.

Syntax

Given below is the syntax of the Store statement.
STORE Relation_name INTO ' required_directory_path ' [USING function];

Example

Assume we have a file **student_data.txt** in HDFS with the following content.

001,Rajiv,Reddy,9848022337,Hyderabad

002,siddarth,Battacharya,9848022338,Kolkata

003,Rajesh,Khanna,9848022339,Delhi

004,Preethi,Agarwal,9848022330,Pune

005,Trupthi,Mohanthy,9848022336,Bhuwaneshwar

006,Archana,Mishra,9848022335,Chennai.

And we have read it into a relation **student** using the LOAD operator as shown below.

```
grunt> student = LOAD 'hdfs://localhost:9000/pig_data/student_data.txt'
  USING PigStorage(',')
  as ( id:int, firstname:chararray, lastname:chararray, phone:chararray,
  city:chararray );
```

Now, let us store the relation in the HDFS directory **"/pig_Output/"** as shown below.

```
grunt> STORE student INTO ' hdfs://localhost:9000/pig_Output/ ' USING
PigStorage (',');
```


After executing the **store** statement, you will get the following output. A directory is created with the specified name and the data will be stored in it.

4.Write a program to implement Twitter Sentiment Analysis System.

Sentiment Analysis is the process of 'computationally' determining whether a piece of writing is positive, negative or neutral. It's also known as opinion mining, deriving the opinion or attitude of a speaker.

Why sentiment analysis?
Business: In marketing field companies use it to develop their strategies, to understand customers' feelings towards products or brand, how people respond to their campaigns or product launches and why consumers don't buy some products.
Politics: In political field, it is used to keep track of political view, to detect consistency and inconsistency between statements and actions at the government level. It can be used to predict election results as well!
Public Actions: Sentiment analysis also is used to monitor and analyse social phenomena, for the spotting of potentially dangerous situations and determining the general mood of the blogosphere.

Tweepy: tweepy is the python client for the official Twitter API.

Install it using following pip command:

pip install tweepy

TextBlob: textblob is the python library for processing textual data. Install it using following pip command:

pip install textblob

Also, we need to install some NLTK corpora using following command:

python -m textblob.download_corpora

(Corpora is nothing but a large and structured set of texts.)

Authentication: In order to fetch tweets through Twitter API, one needs to register an App through their

twitter account.

Follow these steps for the same:

Open this link and click the button: 'Create New App'

Fill the application details. You can leave the callback url field empty.

Once the app is created, you will be redirected to the app page.

Open the 'Keys and Access Tokens' tab.

Copy 'Consumer Key', 'Consumer Secret', 'Access token' and 'Access Token Secret'.

Code:

```
import re
import tweepy
from tweepy import OAuthHandler
from textblob import TextBlob

class TwitterClient(object):
    '''
    Generic Twitter Class for sentiment analysis.
    '''
    def __init__(self):
        '''
        Class constructor or initialization method.
        '''
        # keys and tokens from the Twitter Dev Console
        consumer_key = 'XXXXXXXXXXXXXXXXXXXXXXXXX'
        consumer_secret = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
        access_token = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
        access_token_secret = 'XXXXXXXXXXXXXXXXXXXXXXXXX'

        # attempt authentication
        try:
            # create OAuthHandler object
            self.auth = OAuthHandler(consumer_key, consumer_secret)
            # set access token and secret
            self.auth.set_access_token(access_token, access_token_secret)
            # create tweepy API object to fetch tweets
            self.api = tweepy.API(self.auth)
        except:
            print("Error: Authentication Failed")

    def clean_tweet(self, tweet):
        '''
        Utility function to clean tweet text by removing links, special characters
        using simple regex statements.
        '''
        return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t]) |(\w+:\/\/\S+)", " ", tweet).split())

    def get_tweet_sentiment(self, tweet):
        '''
        Utility function to classify sentiment of passed tweet
        using textblob's sentiment method
        '''
        # create TextBlob object of passed tweet text
        analysis = TextBlob(self.clean_tweet(tweet))
        # set sentiment
        if analysis.sentiment.polarity > 0:
            return 'positive'
        elif analysis.sentiment.polarity == 0:
            return 'neutral'
        else:
            return 'negative'

    def get_tweets(self, query, count = 10):
        '''
        Main function to fetch tweets and parse them.
        '''
        # empty list to store parsed tweets
        tweets = []
```

```
        try:
            # call twitter api to fetch tweets
            fetched_tweets = self.api.search(q = query, count = count)

            # parsing tweets one by one
            for tweet in fetched_tweets:
                # empty dictionary to store required params of a tweet
                parsed_tweet = {}

                # saving text of tweet
                parsed_tweet['text'] = tweet.text
                # saving sentiment of tweet
                parsed_tweet['sentiment'] = self.get_tweet_sentiment(tweet.text)

                # appending parsed tweet to tweets list
                if tweet.retweet_count > 0:
                    # if tweet has retweets, ensure that it is appended only once
                    if parsed_tweet not in tweets:
                        tweets.append(parsed_tweet)
                else:
                    tweets.append(parsed_tweet)

            # return parsed tweets
            return tweets

        except tweepy.TweepError as e:
            # print error (if any)
            print("Error : " + str(e))

def main():
    # creating object of TwitterClient Class
    api = TwitterClient()
    # calling function to get tweets
    tweets = api.get_tweets(query = 'Donald Trump', count = 200)

    # picking positive tweets from tweets
    ptweets = [tweet for tweet in tweets if tweet['sentiment'] == 'positive']
    # percentage of positive tweets
    print("Positive tweets percentage: {} %".format(100*len(ptweets)/len(tweets)))
    # picking negative tweets from tweets
    ntweets = [tweet for tweet in tweets if tweet['sentiment'] == 'negative']
    # percentage of negative tweets
    print("Negative tweets percentage: {} %".format(100*len(ntweets)/len(tweets)))
    # percentage of neutral tweets
    print("Neutral tweets percentage: {} % \
        ".format(100*(len(tweets) -(len( ntweets )+len( ptweets)))/len(tweets)))

    # printing first 5 positive tweets
    print("\n\nPositive tweets:")
    for tweet in ptweets[:10]:
        print(tweet['text'])

    # printing first 5 negative tweets
    print("\n\nNegative tweets:")
    for tweet in ntweets[:10]:
        print(tweet['text'])

if __name__ == "__main__":
    # calling main function
    main()
```

Output:

```
Positive tweets percentage: 22 %
Negative tweets percentage: 15 %
```

5.Write a program to implement word count program using MapReduce.

Objective

In MapReduce word count example, we find out the frequency of each word. Here, the role of Mapper is to map the keys to the existing values and the role of Reducer is to aggregate the keys of common values. So, everything is represented in the form of Key-value pair.

In Hadoop, Map Reduce is a computation that decomposes large manipulation jobs into individual tasks that can be executed in parallel across a cluster of servers. The results of tasks can be joined together to compute final results.

Example of parallel workflow:

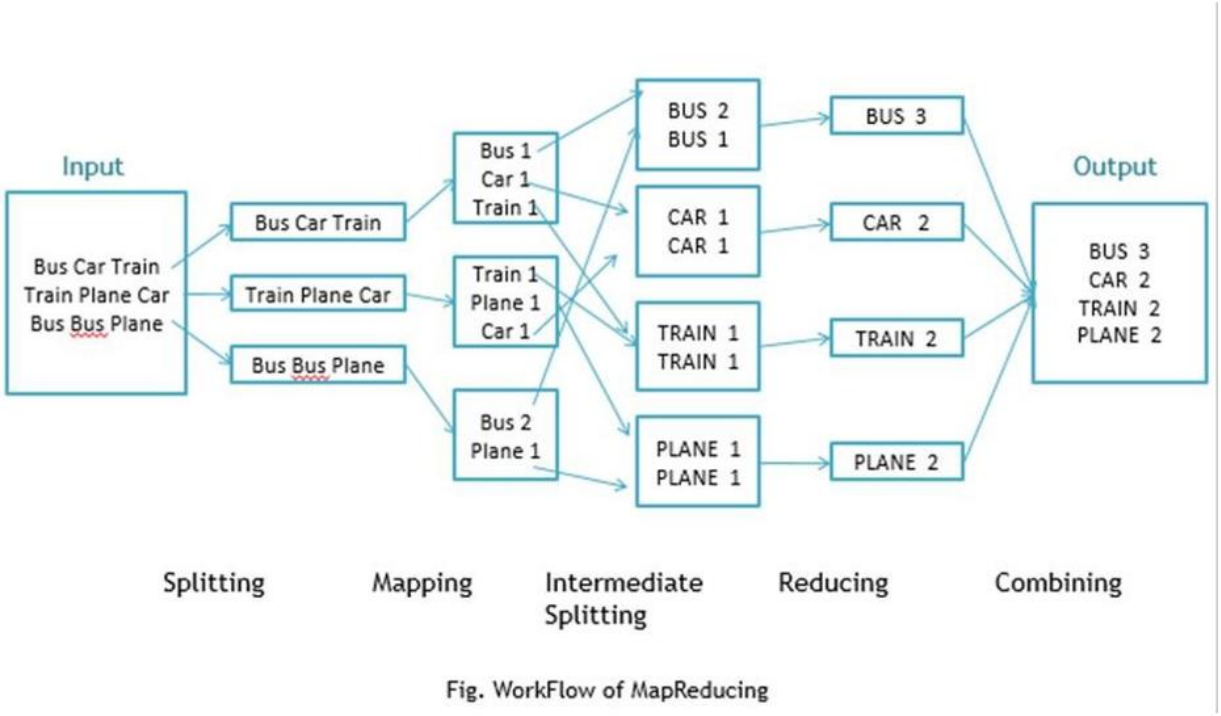


Fig. WorkFlow of MapReducing

Step 1: Create a file with the name word_count_data.txt and add some data to it.

```
cd Documents/                    # to change the directory to /Documents
touch word_count_data.txt        # touch is used to create an empty file
nano word_count_data.txt         # nano is a command line editor to edit the file
cat word_count_data.txt              # cat is used to see the content of the file
```

Step 2: Create a mapper.py file that implements the mapper logic. It will read the data from STDIN and will split the lines into words, and will generate an output of each word with its individual count.

```
cd Documents/                        # to change the directory to /Documents
touch mapper.py                  # touch is used to create an empty file
cat mapper.py                # cat is used to see the content of the file
```

code: mapper.py

```
#!/usr/bin/env python
import sys
for line in sys.stdin:
   line = line.strip()
   words = line.split()
 for word in words:
       print '%s\t%s' % (word, 1)
```

#! is known as shebang and used for interpreting the script.

1. **To test our mapper.py locally that it is working fine or not.**

cat word_count_data.txt | python mapper.py

Step 3: Create a reducer.py file that implements the reducer logic. It will read the output of mapper.py from STDIN(standard input) and will aggregate the occurrence of each word and will write the final output to STDOUT.

```
cd Documents/                    # to change the directory to /Documents
touch reducer.py                 # touch is used to create an empty file
```

code:reducer.py

```
#!/usr/bin/env python

from operator import itemgetter
import sys

current_word = None
current_count = 0
word = None

for line in sys.stdin:
    line = line.strip()
    word, count = line.split('\t', 1)
    try:
        count = int(count)
    except ValueError:
        continue

    if current_word == word:
        current_count += count
    else:
        if current_word:
            print '%s\t%s' % (current_word, current_count)
        current_count = count
        current_word = word

if current_word == word:
    print '%s\t%s' % (current_word, current_count)
```

2. To check reducer code reducer.py with mapper.py is it working properly or not with the help of the below command.

cat word_count_data.txt | python mapper.py | sort -k1,1 | python reducer.py

OUTPUT:

1.For the data in input file mapper shows each word with one count as output in key value pair.

2.Final output will be total count of each unique appearances of words.

6. Implement Page Rank algorithm using Map Reduce.

Aim: Implementing Page Rank Algorithm Using Map-Reduce.

Theory:

PageRank is a way of measuring the importance of website pages. PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.

In the general case, the PageRank value for any page u can be expressed as:

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)}$$

i.e. the PageRank value for a page u is dependent on the PageRank values for each page v contained in the set Bu (the set containing all pages linking to page u), divided by the number L(v) of links from page v.

Suppose consider a small network of four web pages: A, B, C and D. Links from a page to itself, or multiple outbound links from one single page to another single page, are ignored. PageRank is initialized to the same value for all pages. In the original form of PageRank, the sum of PageRank over all pages was the total number of pages on the web at that time, so each page in this example would have an initial value of 1.

$$PR(A) = \frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)}.$$

The damping factor (generally set to 0.85) is subtracted from 1 (and in some variations of the algorithm, the result is divided by the number of documents (N) in the collection) and this term is then added to the product of the damping factor and the sum of the incoming PageRank scores. That is,

$$PR(A) = \frac{1-d}{N} + d\left(\frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \cdots\right).$$

So any page's PageRank is derived in large part from the PageRanks of other pages.
The damping factor adjusts the derived value downward.

CODE:

```
import numpy as np
import scipy as sc
import pandas as pd
from fractions import Fraction
def display_format(my_vector, my_decimal):
  return np.round((my_vector).astype(np.float), decimals=my_decimal)
my_dp = Fraction(1,3)
Mat = np.matrix([[0,0,1],
      [Fraction(1,2),0,0],
      [Fraction(1,2),1,0]])
Ex = np.zeros((3,3))

Ex[:] = my_dp
beta = 0.7
Al = beta * Mat + ((1-beta) * Ex)
r = np.matrix([my_dp, my_dp, my_dp])
r = np.transpose(r)
previous_r = r
for i in range(1,100):
  r = Al * r
  print (display_format(r,3))
  if (previous_r==r).all():
    break
  previous_r = r
print ("Final:\n", display_format(r,3))
print ("sum", np.sum(r))
```

OUTPUT:

```
[[0.333]
 [0.217]
 [0.45 ]]
[[0.415]
 [0.217]
 [0.368]]
[[0.358]
 [0.245]
 [0.397]]
...
//Reduce upper matrix if need to
...
[[0.375]
 [0.231]
 [0.393]]
```

FINAL:

```
[[0.375]
 [0.231]
 [0.393]]
sum 0.9999999999999951
```

7.Visualization: Connect to data, build charts and analyze Data, create dashboard.

Big data visualization often goes beyond the typical techniques used in normal visualization, such as pie charts, histograms and corporate graphs. It instead uses more complex representations, such as heat maps and fever charts. Big data visualization requires powerful computer systems to collect raw data, process it and turn it into graphical representations that humans can use to quickly draw insights.
While big data visualization can be beneficial, it can pose several disadvantages to organizations.
They are as follows:
• To get the most out of big data visualization tools, a visualization specialist must be hired. This specialist must be able to identify the best data sets and visualization styles to guarantee organizations are optimizing the use of their data.
• Big data visualization projects often require involvement from IT, as well as management, since the visualization of big data requires powerful computer hardware, efficient storage systems and even a move to the cloud.
• The insights provided by big data visualization will only be as accurate as the information being visualized. Therefore, it is essential to have people and processes in place to govern and control the quality of corporate data, metadata and data sources

Individuals could utilize data visualization to present data in a more effective manner and companies turn to dashboards to report their performance metrics in real-time. Dashboards are effective data visualization tools for tracking and visualizing data from multiple data sources, providing visibility into the effects of specific behaviors by a team or an adjacent one on performance.
Dashboards include common visualization techniques, such as:

1. Tables: This consists of rows and columns used to compare variables.
2. Pie charts and stacked bar charts: These graphs are divided into sections that represent parts of a whole.
3. Line graphs and area charts: These visuals show change in one or more quantities by plotting a series of data points over time. Line graphs utilize lines to demonstrate these changes while area charts connect data points with line segments, stacking variables on top of one another and using color to distinguish between variables.
4. Histograms: This graph plots a distribution of numbers using a bar chart representing the quantity of data that falls within a particular range.
5. Scatter plots: These visuals are beneficial in reveling the relationship between two variables, and they are commonly used within regression data analysis.
6. Heat maps: These graphical displays are helpful in visualizing behavioral Data Visualization data by location.
7. Tree maps: Display hierarchical data as a set of nested shapes, typically rectangles. Treemaps are great for comparing the proportions between categories via their area size.
8. Population pyramids: This technique uses a stacked bar graph to display the complex social narrative of a population.

DATA VISUALIZATION TOOLS AND VENDORS

Data visualization tools can be used in a variety of ways. The most common use today is as a business intelligence (BI) reporting tool. Users can set up visualization tools to generate automatic dashboards that track company performance across key performance indicators (KPIs) and visually interpret the results.

While Microsoft Excel continues to be a popular tool for data visualization, others have been created that provide more sophisticated abilities:

• IBM Cognos Analytics

• Qlik Sense and QlikView

Microsoft Power BI

• Oracle Visual Analyzer

• SAP Lumira

• SAS Visual Analytics

• Tibco Spotfire

• Zoho Analytics

• D3.js

• Jupyter

• MicroStrategy

• Google Charts

TABLEAU

Tableau is a visual analysis solution that allows people to explore and analyze data with simple drag and drop operations. It has a user-friendly interface that creates reports that look great right at the beginning.

Tableau is a rapid BI Software.

Great Visualizations: Allows to connect to the data, visualize and creative interactive, sharable dashboards in a few clicks.

Ease of use: It's easy enough that any excel user can learn it, but powerful enough to satisfy even the most complex analytical problems

Fast: We can create interactive dashboards, quick filters and calculations.

Three main stages in Tableau

Connect data source: Connect Tableau to any data source like MS-Excel, MySQL, and Oracle. Tableau connects data in two ways Live connect and extract.

Analyze & Visualize: Analyze the data by filtering, sorting and Visualize Data Visualization the data using the relevant chart provided. Tableau automatically analyzes the data as follows: Dimensions: Tableau treats any field containing qualitative, categorical information as dimension. All dimensions are indicated by "Blue" color.

Measures: Tableau treats any field containing numeric information as a measure. All measures are indicated by "Green" color Tableau suggests the recommended charts based on the dimensions and measures.

Share: Tableau Dashboards can be shared as word documents, pdf files, images.

Connecting to the data in Tableau Public Tableau Public has a graphical user interface (GUI) that was designed to enable users to load data sources without having to write code. Since the only place to save

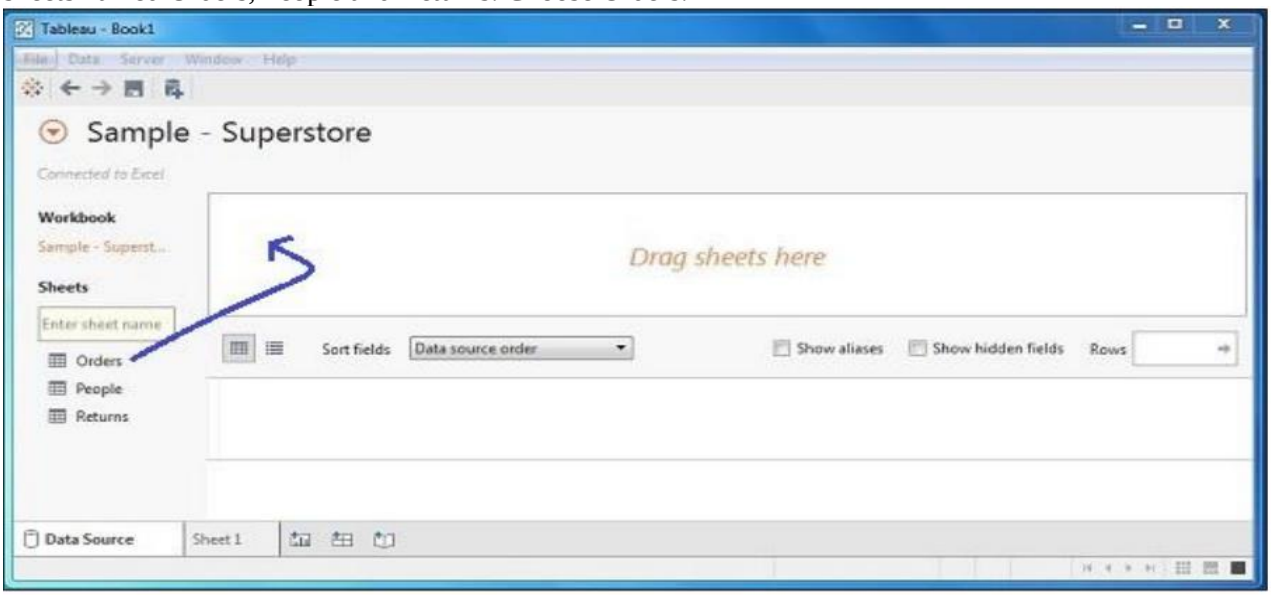
Tableau Public documents is in Tableau's Cloud, data sources are automatically extracted and packaged with the workbook.

There are three basic steps involved in creating any Tableau data analysis report.
These three steps are:

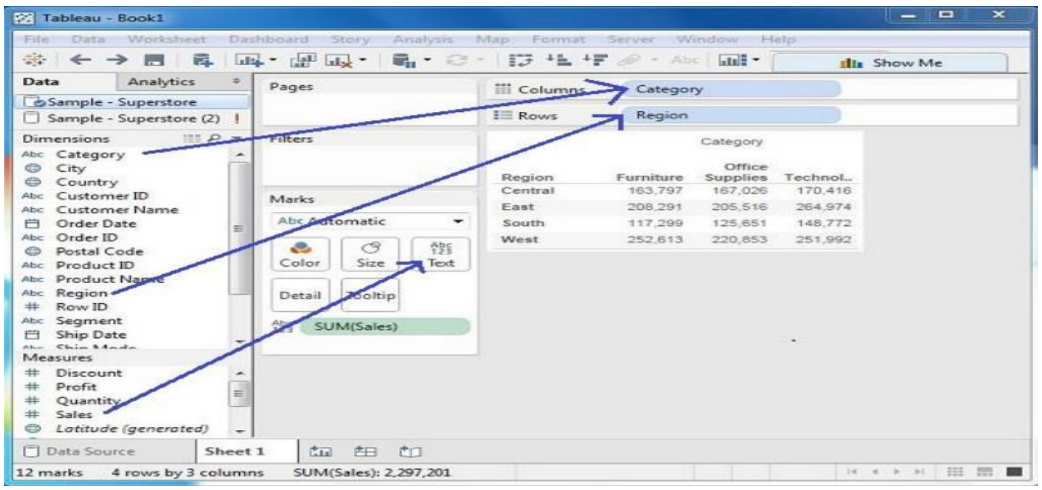
- Connect to a data source: It involves locating the data and using an appropriate type of connection to read the data.
- Choose dimensions and measures: This involves selecting the required columns from the source data for analysis.
- Apply visualization technique: This involves applying required visualization methods, such as a specific chart or graph type to the data being analyzed.

For convenience, let's use the sample data set that comes with Tableau installation named sample – superstore.xls

Connect to a Data Source On opening Tableau, you will get the start page showing various data sources. Under the header "Connect", you have options to choose a file or server or saved data source. Under Files, choose excel. Then navigate to the file "Sample – Superstore.xls" as mentioned above. The excel file has three sheets named Orders, People and Returns. Choose Orders.

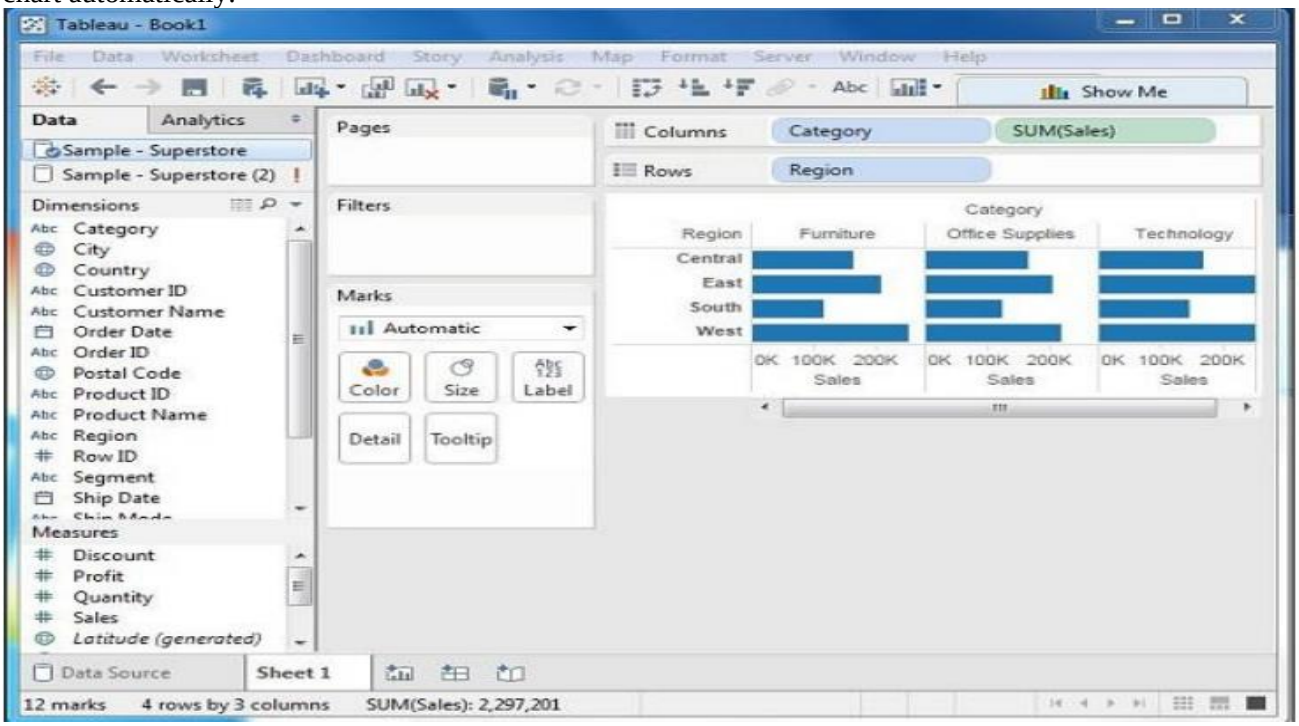


Choose the Dimensions and Measures Next, choose the data to be analyzed by deciding on the dimensions and measures. Dimensions are the descriptive data while measures are numeric data. When put together, they help visualize the performance of the dimensional data with respect to the data which are measures.
Choose Category and Region as the dimensions and Sales as the measure. Drag and drop them as shown in the following screenshot. The result shows the total sales in each category for each region.

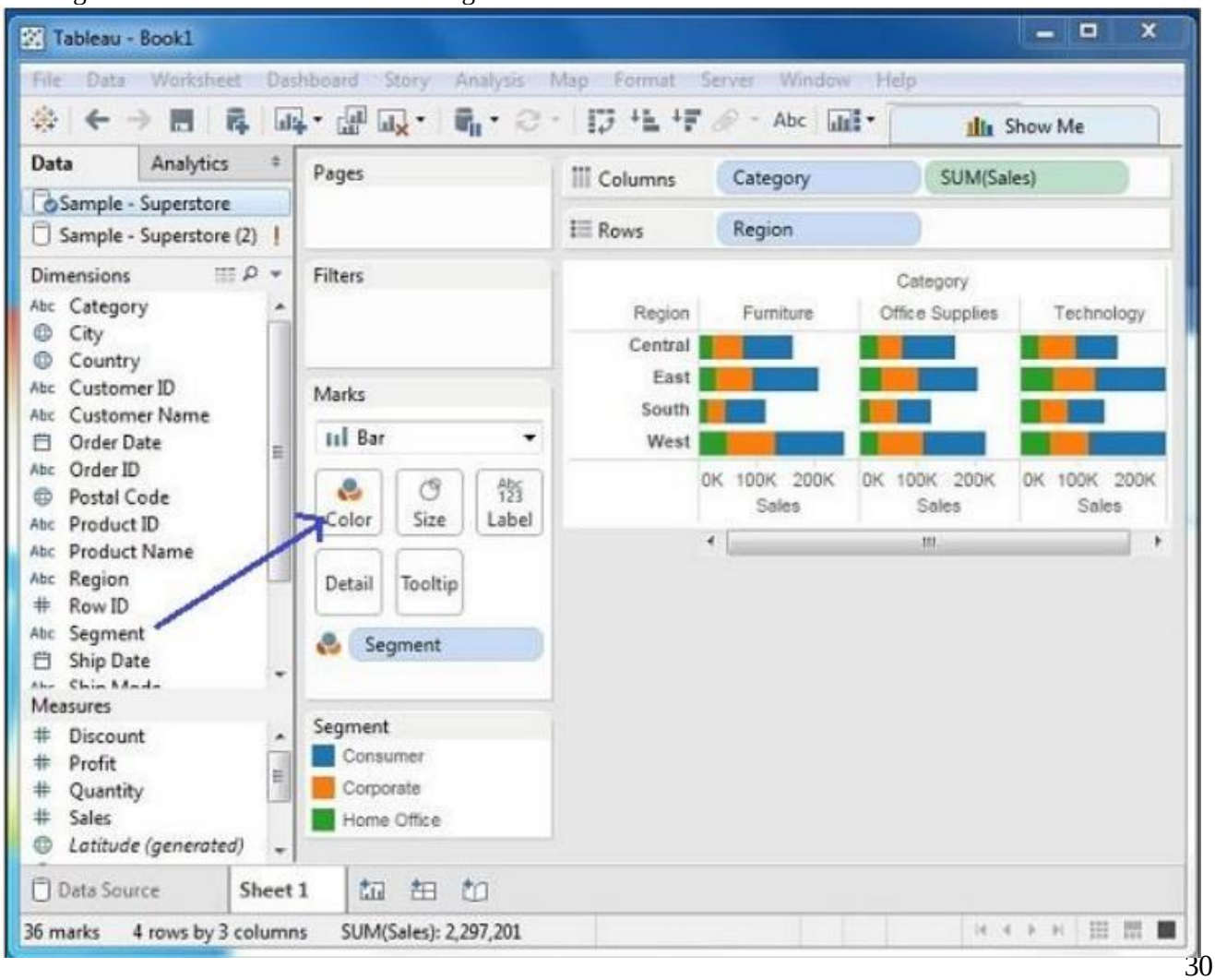


Apply Visualization Technique In the previous step, you can see that the data is available only as numbers. You have to read and calculate each of the values to judge the performance. However, you can see them as graphs or charts with different colors to make a quicker judgment. We drag and drop the sum (sales) column from the

Marks tab to the Columns shelf. The table showing the numeric values of sales now turns into a bar chart automatically.



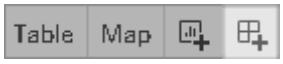
You can apply a technique of adding another dimension to the existing data. This will add more colors to the existing bar chart as shown in the following screenshot.



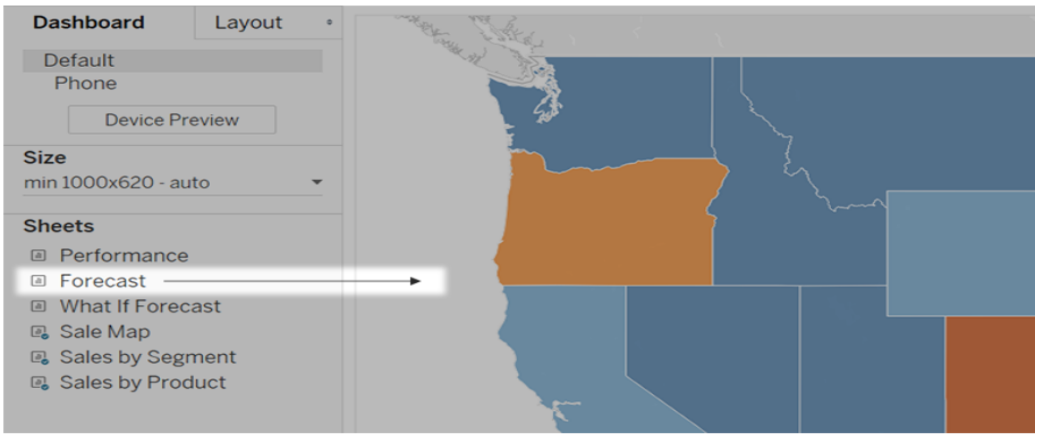
Creating Dashboards:

A dashboard is a collection of several views, letting you compare a variety of data simultaneously. For example, if you have a set of views that you review every day, you can create a dashboard that displays all the views at once, rather than navigate to separate worksheets.

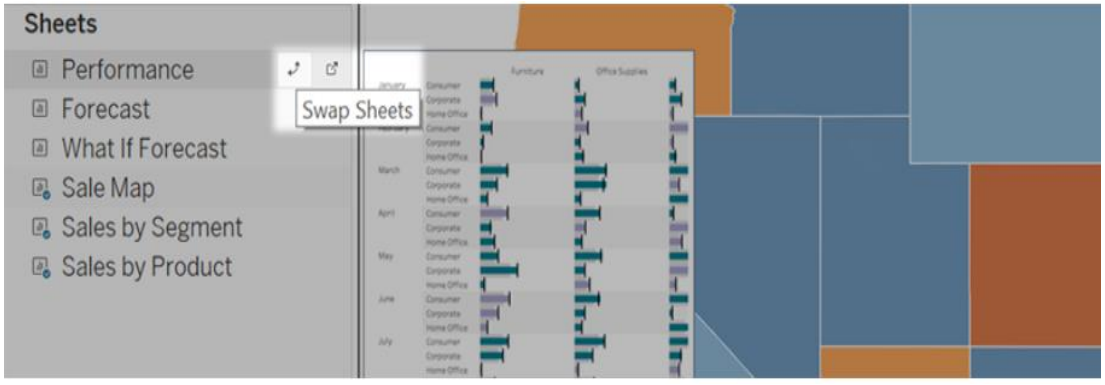
Like worksheets, you access dashboards from tabs at the bottom of a workbook. Data in sheets and dashboards is connected; when you modify a sheet, any dashboards containing it change, and vice versa. Both sheets and dashboards update with the latest available data from the data source.
After you've created one or more sheets, you can combine them in a dashboard, add interactivity, and much more.
Create a dashboard, and add or replace sheets
You create a dashboard in much the same way you create a new worksheet.
1.At the bottom of the workbook, click the New Dashboard icon:



2.From the Sheets list at left, drag views to your dashboard at right



3.To replace a sheet, select it in the dashboard at right. In the Sheets list at left, hover over the replacement sheet, and click the Swap Sheets button.



****