Sri Indu
College of Engineering & Technology
UGC Autonomous Institution
Recognized under 2(f) & 12(B) of UGC Act 1956,
NAAC, Approved by AICTE &
Permanently Affiliated to JNTUH

Estd.2001

NAAC
NATIONAL ASSESSMENT AND ACCREDITATION COUNCIL

BIG DATA ANALYTICS LAB MANUAL

B.Tech.– III Year – I Semester

DEPARTMENTOFCOMPUTERSCIENCEANDENGINEERING(AI&DS)

ACADEMICYEAR :2024-25

SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY
(An Autonomous Institution under UGC, New Delhi)
Recognized under 2(f) and 12(B) of UGC Act 1956
NBA Accredited, Approved by AICTE and Permanently affiliated to JNTUH
Sheriguda (V), Ibrahimpatnam, R.R.Dist, Hyderabad - 501 510

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD
B.Tech-III Year II Sem **L T P C**

Course Code: **0 0 2 1**

BIG DATA ANALYTICTS LAB MANUAL

Course Objectives

1. The purpose of this course is to provide the students with the knowledge of Big data Analytics principles and techniques.
2. This course is also designed to give an exposure of the frontiers of Big data Analytics

Course Outcomes

1.Use Excel as an Analytical tool and visualization tool.
2. Ability to program using HADOOP and Map reduce.
3. Ability to perform data analytics using ML in R.
4. Use Cassandra to perform social media analytics.

List of Experiments

1. Implement a simple map-reduce job that builds an inverted index on the set of input documents (Hadoop)
2. Process big data in HBase
3. Store and retrieve data in Pig
4. Perform Social media analysis using cassandra
5. Buyer event analytics using Cassandra on suitable product sales data.
6. Using Power Pivot (Excel) Perform the following on any dataset
a) Big Data Analytics
b) Big Data Charting
7. Use R-Project to carry out statistical analysis of big data
8. Use R-Project for data visualization of social media data

TEXT BOOKS:

1.Big Data Analytics, Seema Acharya, Subhashini Chellappan, Wiley 2015.
2. Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today's Business, Michael Minelli, Michehe Chambers, 1st Edition, Ambiga Dhiraj, Wiely CIO Series, 2013.
3. Hadoop: The Definitive Guide, Tom White, 3rd Edition, O"Reilly Media, 2012.
4. Big Data Analytics: Disruptive Technologies for Changing the Game, Arvind Sathi, 1st Edition,
IBM Corporation, 2012.

REFERENCES:

1.Big Data and Business Analytics, Jay Liebowitz, Auerbach Publications, CRC press (2013).
2. Using R to Unlock the Value of Big Data: Big Data Analytics with Oracle R Enterprise and Oracle R Connector for Hadoop, Tom Plunkett, Mark Hornick, McGraw-Hill/Osborne Media (2013), Oracle press.
3. Professional Hadoop Solutions, Boris lublinsky, Kevin t. Smith, Alexey Yakubovich, Wiley, ISBN: 9788126551071, 2015.
4. Understanding Big data, Chris Eaton, Dirk deroos et al., McGraw Hill, 2012.
5. Intelligent Data Analysis, Michael Berthold, David J. Hand, Springer, 2007.
6. Taming the Big Data Tidal Wave: Finding Opportunities in Huge Data Streams with Advanced
Analytics, Bill Franks, 1st Edition, Wiley and SAS Business Series, 2012.

INDEX

Experiment 1: Implement a simple map-reduce job that builds an inverted index on the set of input documents (Hadoop)

Aim: To implement an Inverted index on Hadoop.
Resources: Hadoop, Java, Eclipse
Theory: Hadoop is an open-source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Hadoop runs applications using the MapReduce algorithm, where the data is processed in parallel with others. Hadoop is used to develop applications that could perform complete statistical analysis on huge amounts of data.

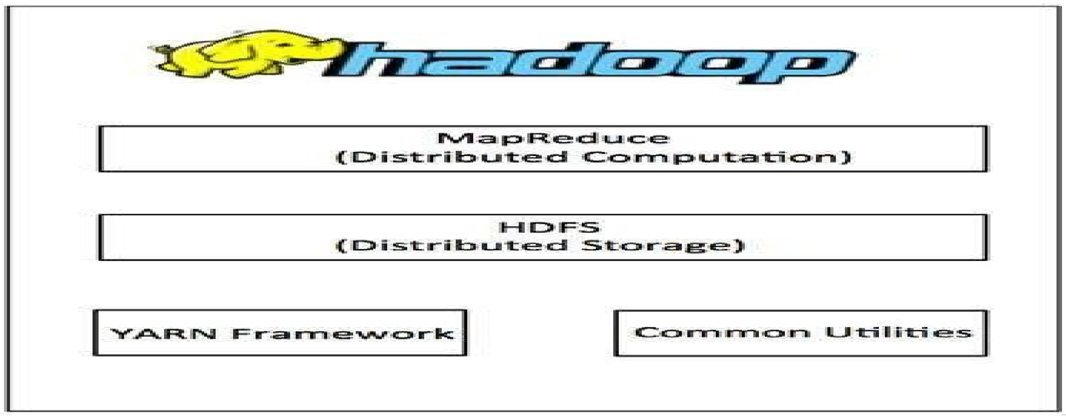
Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models. The Hadoop framework application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

Hadoop Architecture

At its core, Hadoop has two major layers namely −

- Processing/Computation layer (MapReduce), and
- Storage layer (Hadoop Distributed File System).

Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models. The Hadoop framework application works in an environment that provides distributed *storage* and *computation* across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.



MapReduce

MapReduce is a processing technique and a program model for distributed computing based on java. The MapReduce algorithm contains two important tasks, namely Map and Reduce.

Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples.

As the sequence of the name MapReduce implies, the reduce task is always performed after the map job.

The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called mappers and reducers.

Decomposing a data processing application into *mappers* and *reducers* is sometimes nontrivial. But, once we write an application in the MapReduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of machines in a cluster is merely a configuration change. This simple scalability is what has attracted many programmers to use the MapReduce model.

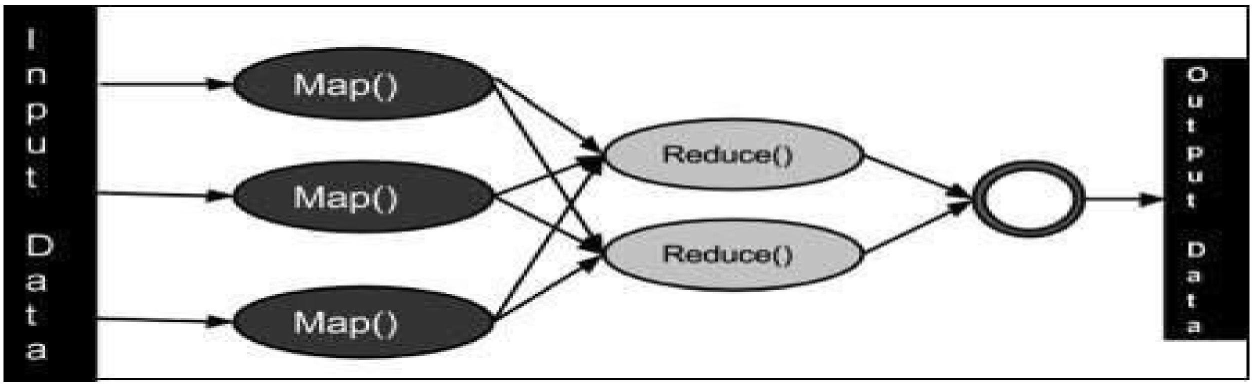
The Algorithm

- Generally MapReduce paradigm is based on sending the computer to where the data resides.
- MapReduce program executes in three stages, namely map stage, shuttle stage,and reduce stage.
    - **<u>Map stage</u>** − The map or mapper's job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.
    - **<u>Reduce stage</u>** − This stage is the combination of the **Shuffle** stage and the **Reduce** stage. The Reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.

- During a MapReduce job, Hadoop sends the Map and Reduce tasks to the appropriate servers in the cluster.

The framework manages all the details of data-passing such as issuing tasks, verifying task completion, and copying data around the cluster between the nodes.

- Most of the computing takes place on nodes with data on local disks that reduces the network traffic.
- After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.



Inputs and Outputs (Java Perspective)

The MapReduce framework operates on <key, value> pairs, that is, the framework views the input to the job as a set of <key, value> pairs and produces a set of <key, value> pairs as the output of the job, conceivably of different types.

The key and the value classes should be in serialized manner by the framework and hence, need to implement the Writable interface. Additionally, the key classes have to implement the Writable- Comparable interface to facilitate sorting by the framework. Input and Output types of a **MapReduce job** − (Input) <k1, v1> → map
→ <k2, v2> → reduce → <k3, v3>(Output).

	Input	Output
Map	<k1, v1>	list (<k2, v2>)
Reduce	<k2, list(v2)>	list (<k3, v3>)

Procedure:

<u>Steps to install Hadoop:</u>

1. Make sure java is installed.

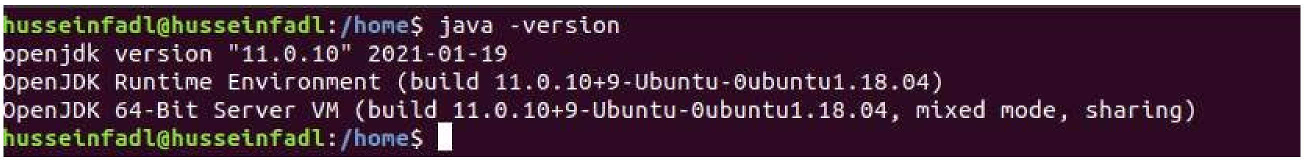
```
java -version
```

If java is not installed, then type in the following commands:

```
sudo apt-get install update
sudo apt-get update
sudo apt-get install default-jdk
```

Make sure now java is installed.

```
java -version
```



```
husseinfadl@husseinfadl:/home$ java -version
openjdk version "11.0.10" 2021-01-19
OpenJDK Runtime Environment (build 11.0.10+9-Ubuntu-0ubuntu1.18.04)
OpenJDK 64-Bit Server VM (build 11.0.10+9-Ubuntu-0ubuntu1.18.04, mixed mode, sharing)
husseinfadl@husseinfadl:/home$
```

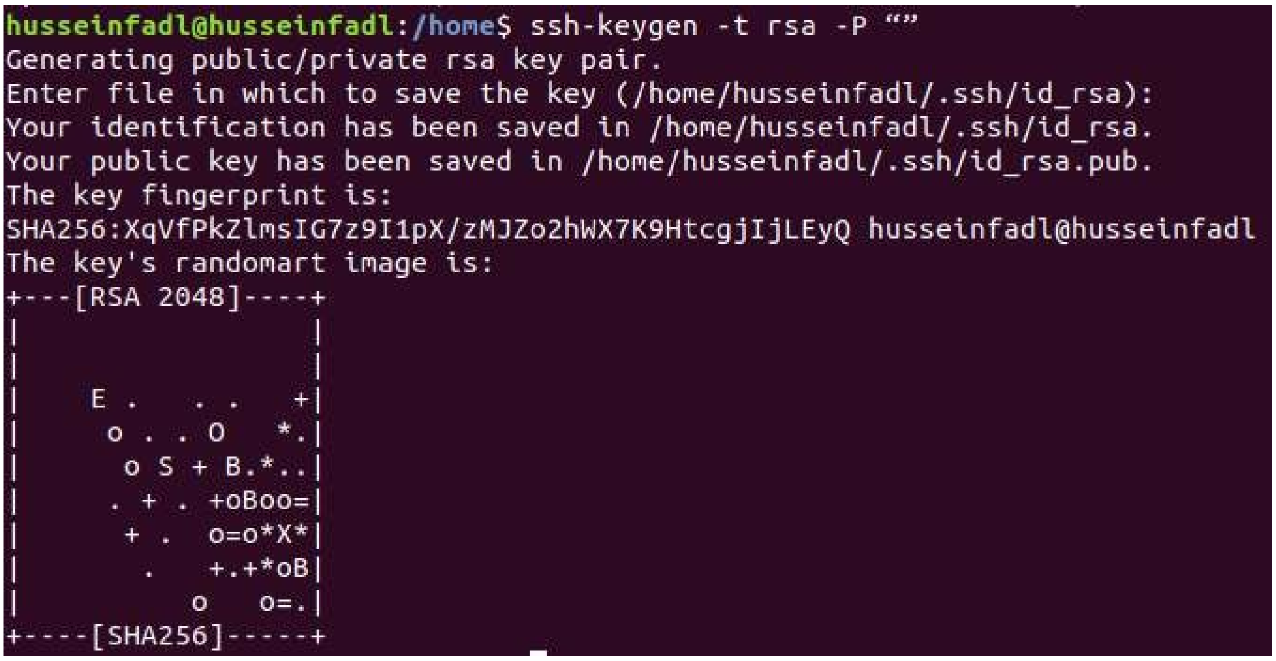
2. Install ssh server

```
sudo apt-get install ssh- server
```

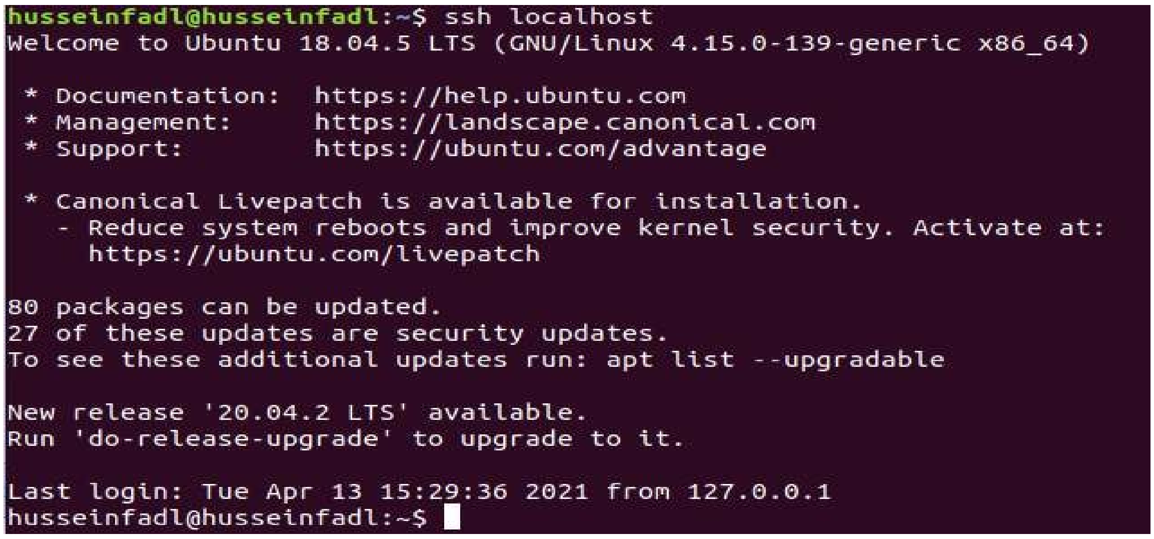
Generate public/private RSA key pair.

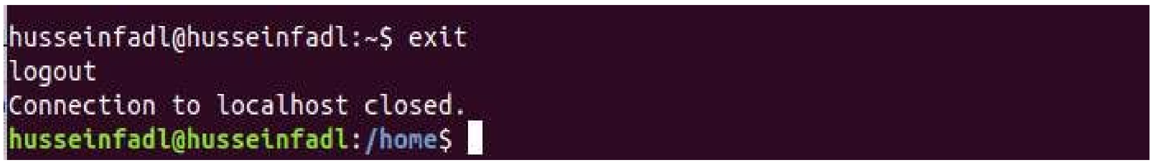
```
ssh-keygen -t rsa -P “”
```

When prompted for the file name to save the key, press Enter (leave it blank).

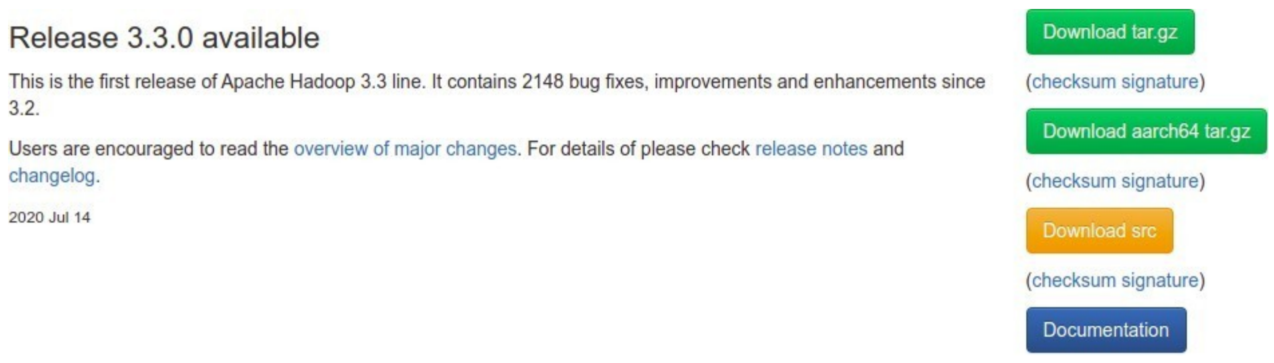

```
husseinfadl@husseinfadl:/home$ ssh-keygen -t rsa -P “”
Generating public/private rsa key pair.
Enter file in which to save the key (/home/husseinfadl/.ssh/id_rsa):
Your identification has been saved in /home/husseinfadl/.ssh/id_rsa.
Your public key has been saved in /home/husseinfadl/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:XqVfPkZlmsIG7z9I1pX/zMJZo2hWX7K9HtcgjIjLEyQ husseinfadl@husseinfadl
The key's randomart image is:
+---[RSA 2048]----+
|                 |
|                 |
|    E .    . .  +|
|     o . . O   *.|
|      o S + B.*..|
|     . + . +oBoo=|
|      + .  o=o*X*|
|        .   +.+*oB|
|           o   o=.|
+----[SHA256]-----+
```

Type the following commands:
cat $HOME/.ssh/id_rsa.pub >>
$HOME/.ssh/authorized_keys ssh localhost exit


```
husseinfadl@husseinfadl:~$ ssh localhost
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-139-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

80 packages can be updated.
27 of these updates are security updates.
To see these additional updates run: apt list --upgradable

New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Apr 13 15:29:36 2021 from 127.0.0.1
husseinfadl@husseinfadl:~$
```


```
husseinfadl@husseinfadl:~$ exit
logout
Connection to localhost closed.
husseinfadl@husseinfadl:/home$
```

3. Install Hadoop by navigating to the following link and downloading the tar.gz file for Hadoop version 3.3.0 (or a later version if you wish). (478 MB)
   https://hadoop.apache.org/release/3.3.0.html



Release 3.3.0 available

This is the first release of Apache Hadoop 3.3 line. It contains 2148 bug fixes, improvements and enhancements since 3.2.

Users are encouraged to read the overview of major changes. For details of please check release notes and changelog.

2020 Jul 14

4. Once downloaded, open the terminal and cd to the directory where it is downloaded (assume the desktop for example) and extract it as follows:

```
cd Desktop
sudo tar -xvzf hadoop-3.3.0.tar.gz
```

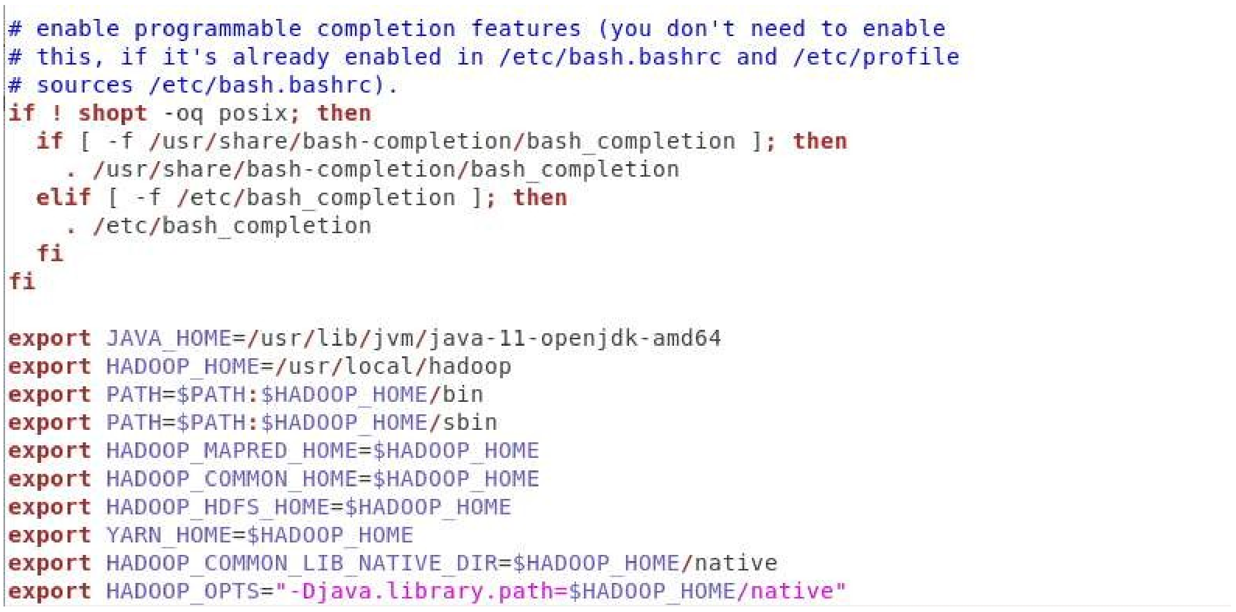
   You can now check that there is an extracted file named hadoop-3.3.0 by typing the command “ls” or by visually inspecting the files.

5. Now, we move the extracted file to the location /usr/local/hadoop

```
sudo mv hadoop-3.3.0 /usr/local/hadoop
```

6. Let’s configure the hadoop system.
   Type the following command:
   sudo gedit ~/.bashrc
   At the end of the file, add the following lines: (Note: Replace the java version with the version number you already have. You can navigate to the directory /usr/lib/jvm and check the file name java-xx-openjdk-amd64)

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/native"
```



```
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/native"
```

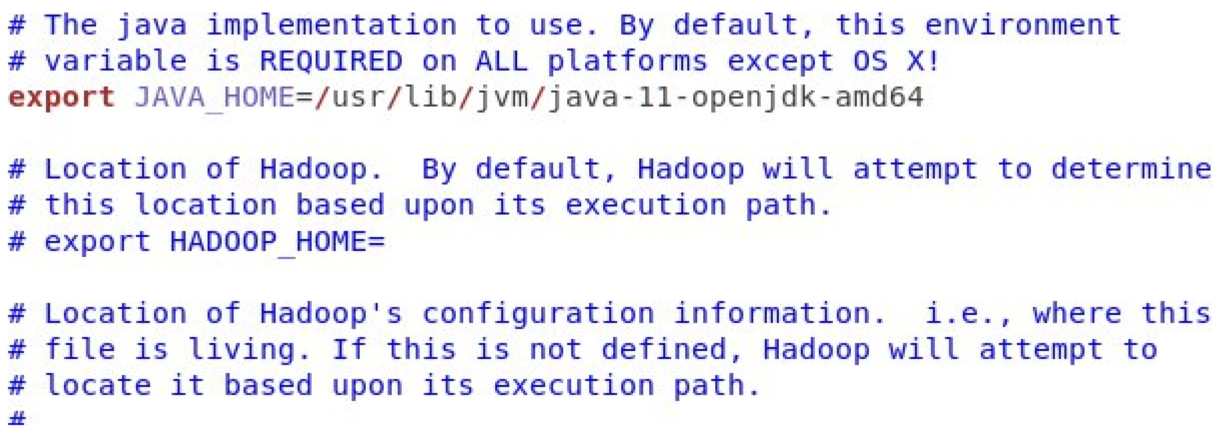
7. Save the file and close it.
8. Now from the terminal, type the following command:

   ```
   source ~/.bashrc
   ```
9. We start configuring Hadoop by opening **hadoop-env.sh** as follows:

   ```
   sudo gedit /usr/local/hadoop/etc/hadoop/hadoop-env.sh
   ```
   Search for the line starting with **export JAVA_HOME=** and replace it with the following line.

   ```
   export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
   ```

   Save the file by clicking on “Save” or (Ctrl+S)

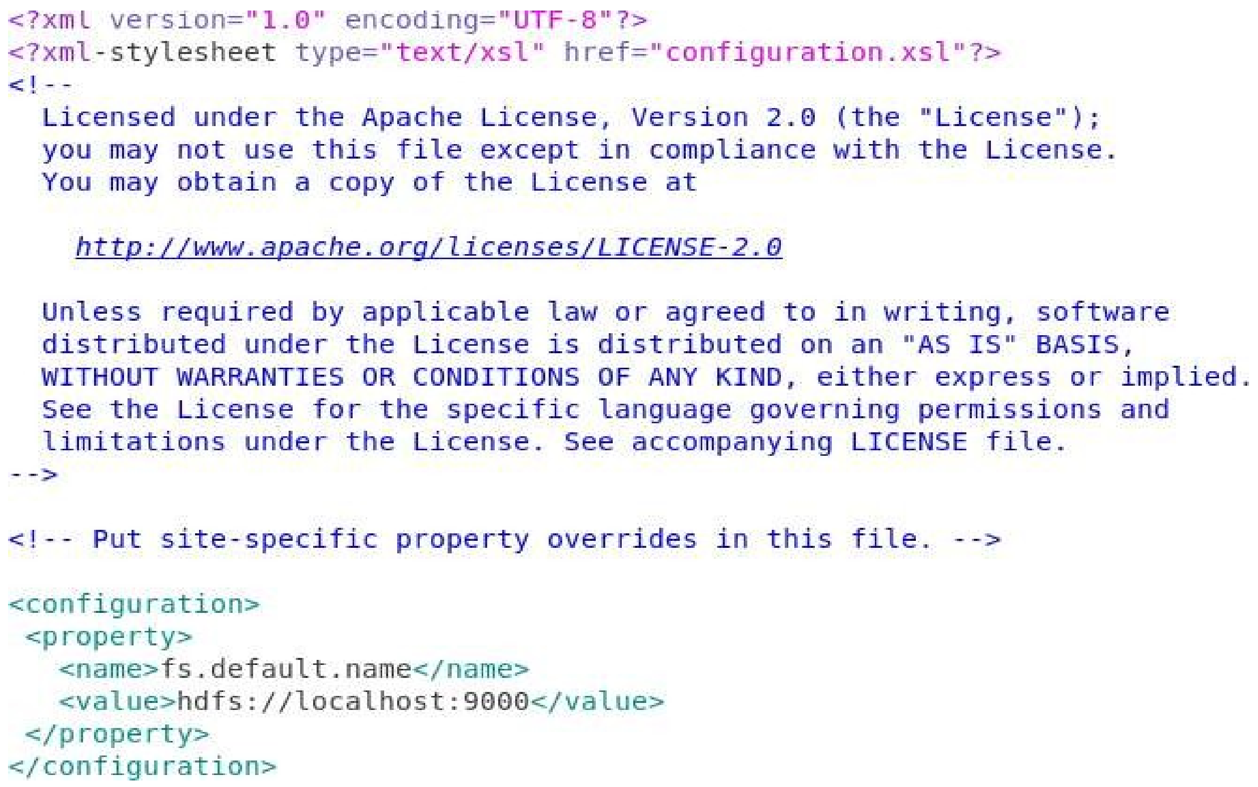


```
# The java implementation to use. By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64

# Location of Hadoop.  By default, Hadoop will attempt to determine
# this location based upon its execution path.
# export HADOOP_HOME=

# Location of Hadoop's configuration information.  i.e., where this
# file is living. If this is not defined, Hadoop will attempt to
# locate it based upon its execution path.
#
```

10. Open **core-site.xml** as follows:

   ```
   sudo gedit /usr/local/hadoop/etc/hadoop/core-site.xml
   ```
Add the following lines between the tags <configuration> and </configuration> and save it (Ctrl+S).

```
<property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:9000</value>
</property>
```



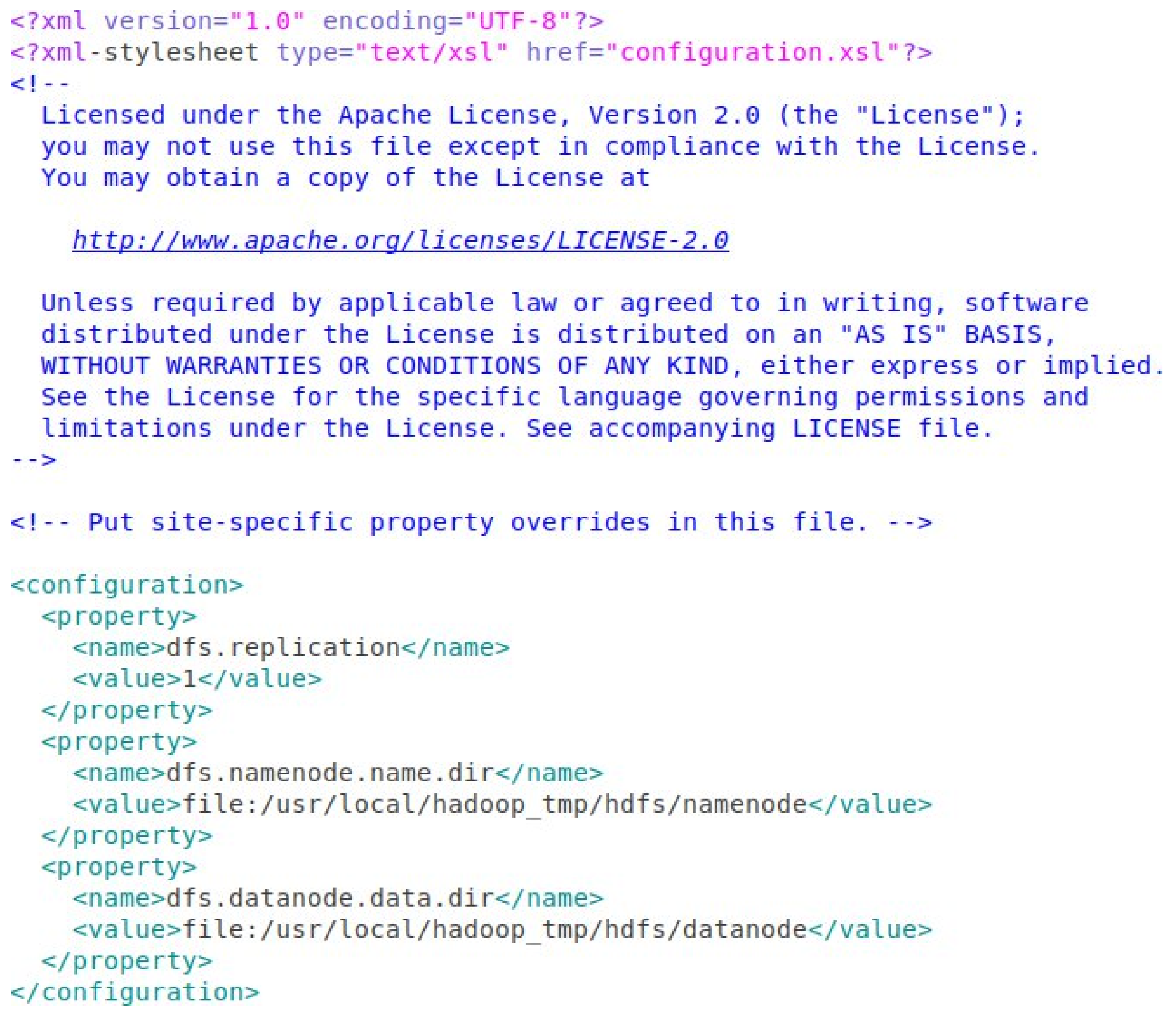
```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
 <property>
   <name>fs.default.name</name>
   <value>hdfs://localhost:9000</value>
 </property>
</configuration>
```

11. Open **hdfs-site.xml** as follows:

```
sudo  gedit  /usr/local/hadoop/etc/hadoop/hdfs-site.xml
```

Add the following lines between the tags <configuration> and </configuration> and save it (Ctrl+S).

```
<property>
   <name>dfs.replication</name>
   <value>1</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:/usr/local/hadoop_space/hdfs/namenode</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:/usr/local/hadoop_space/hdfs/datanode</value>
</property>
```



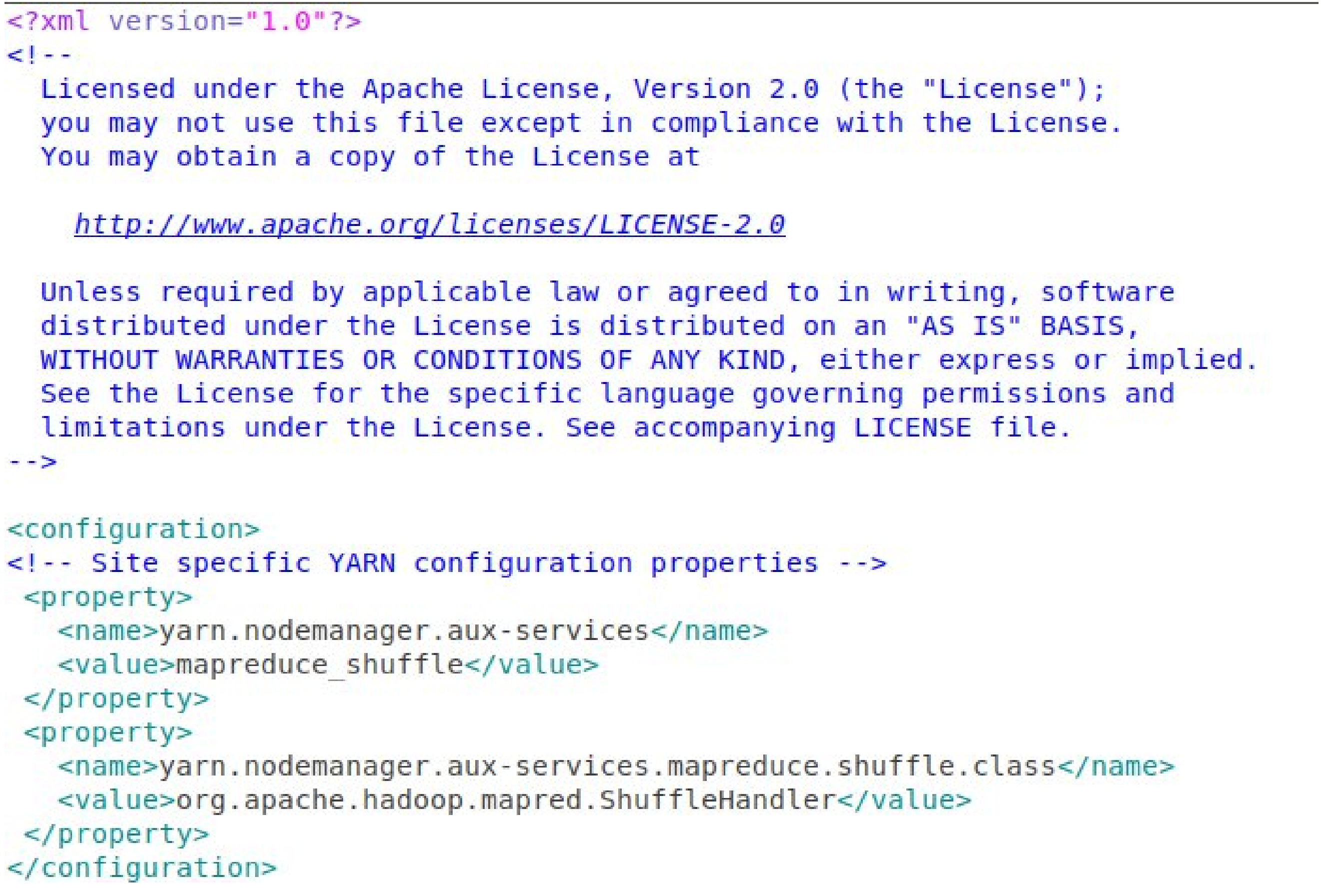
```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/usr/local/hadoop_tmp/hdfs/namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/usr/local/hadoop_tmp/hdfs/datanode</value>
  </property>
</configuration>
```

Open **yarn-site.xml** as follows:

```
sudo gedit /usr/local/hadoop/etc/hadoop/yarn-site.xml
```

Add the following lines between the tags <configuration> and </configuration> and save it (Ctrl+S)

```
 <property>
     <name>yarn.nodemanager.aux-services</name>
     <value>mapreduce_shuffle</value>
 </property>
 <property>
     <name>yarn.nodemanager.aux-
services.mapreduce.shuffle.class</n ame>
     <value>org.apache.hadoop.mapred.ShuffleHandler</value>
 </property>
```



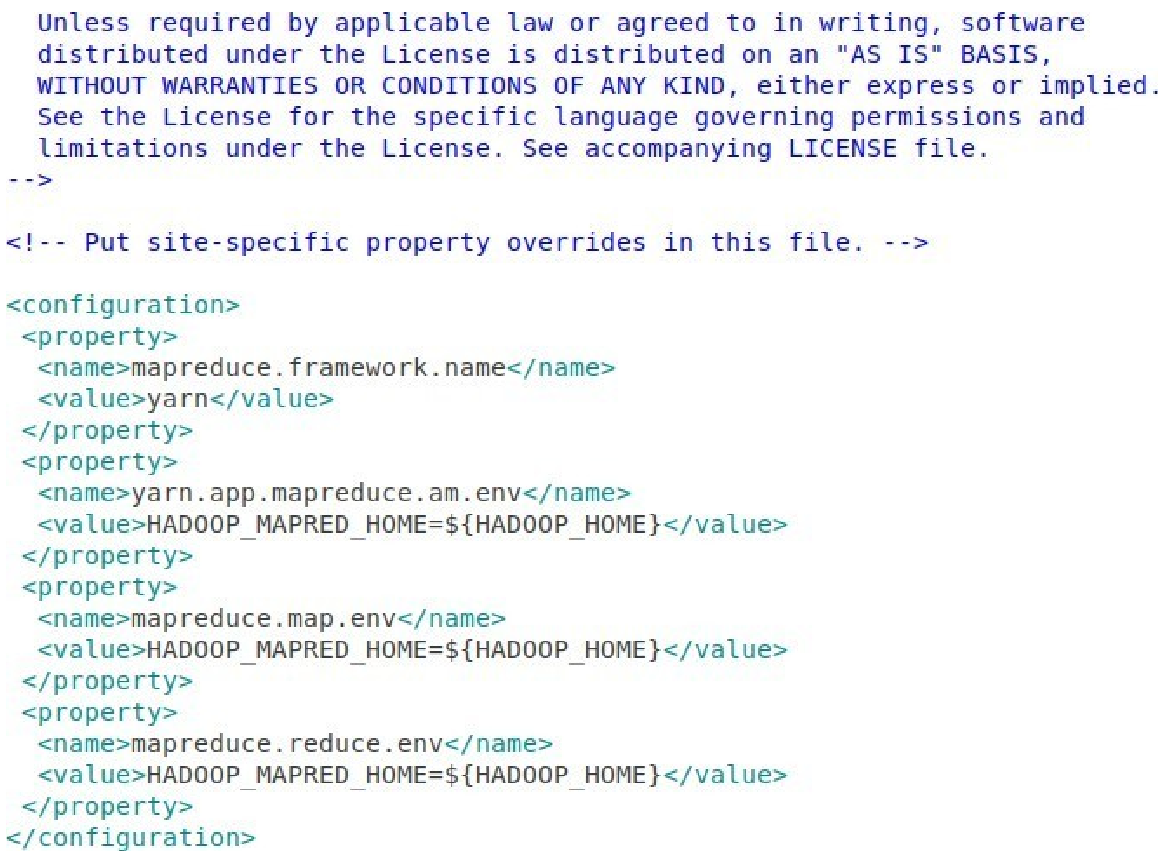
```
<?xml version="1.0"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<configuration>
<!-- Site specific YARN configuration properties -->
 <property>
   <name>yarn.nodemanager.aux-services</name>
   <value>mapreduce_shuffle</value>
 </property>
 <property>
   <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
   <value>org.apache.hadoop.mapred.ShuffleHandler</value>
 </property>
</configuration>
```

12. Open **mapred-site.xml** as follows:

```
sudo  gedit  /usr/local/hadoop/etc/hadoop/mapred-site.xml
```

Add the following lines between the tags <configuration> and </configuration> and save it (Ctrl+S)

```
 <property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
 </property>
 <property>
  <name>yarn.app.mapreduce.am.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
 <property>
  <name>mapreduce.map.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
 <property>
  <name>mapreduce.reduce.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
```



```
  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
 <property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
 </property>
 <property>
  <name>yarn.app.mapreduce.am.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
 <property>
  <name>mapreduce.map.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
 <property>
  <name>mapreduce.reduce.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
 </property>
</configuration>
```

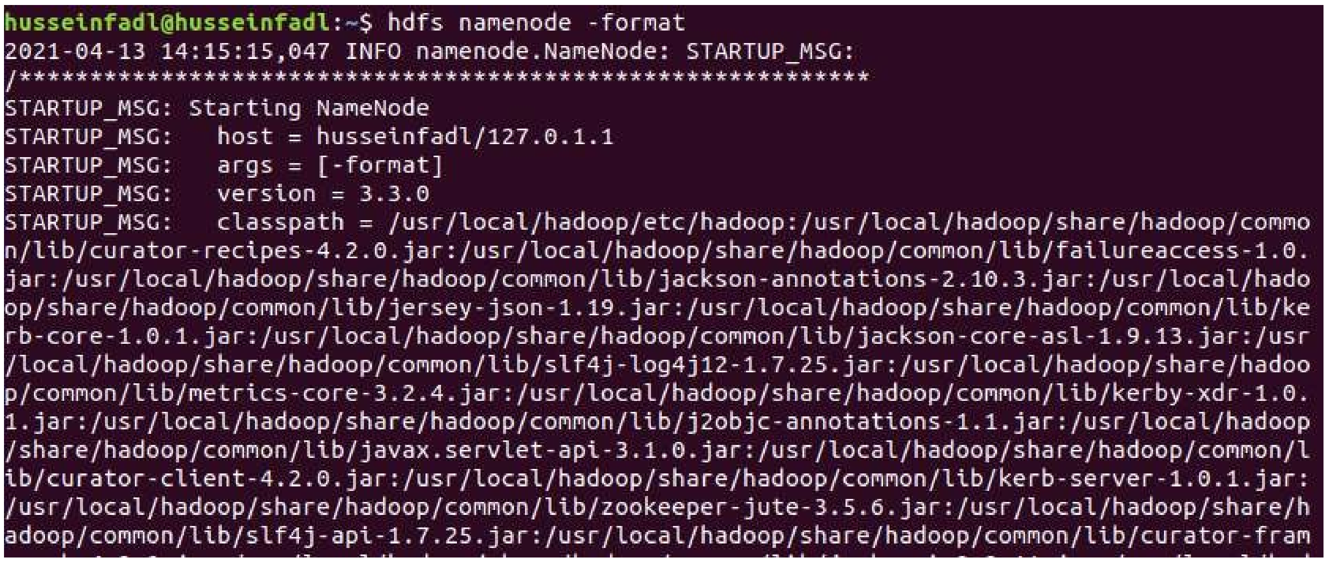
13. Now, run the following commands on the terminal to create a directory for hadoop space, name node and data node.

```
sudo mkdir -p /usr/local/hadoop_space sudo
mkdir -p
/usr/local/hadoop_space/hdfs/namenode sudo mkdir -p
/usr/local/hadoop_space/hdfs/datanode
```

Now we have successfully installed Hadoop.

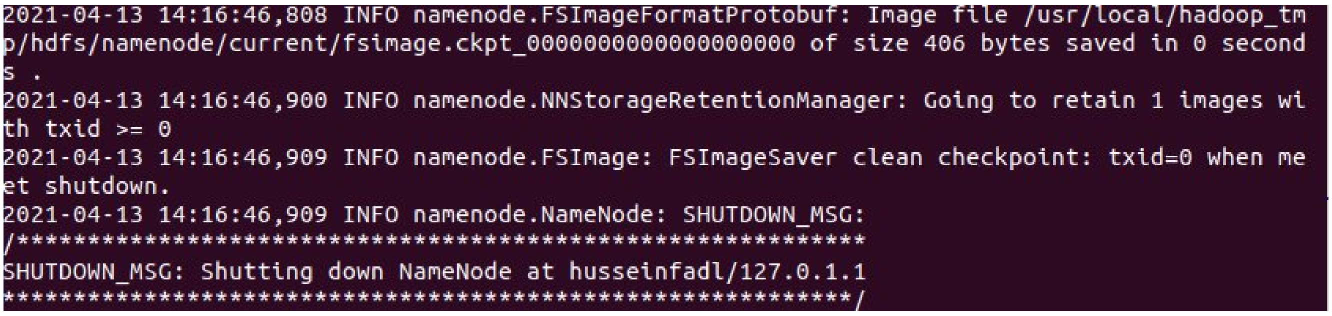
14. Format the namenode as follows:

```
hdfs namenode -format
```



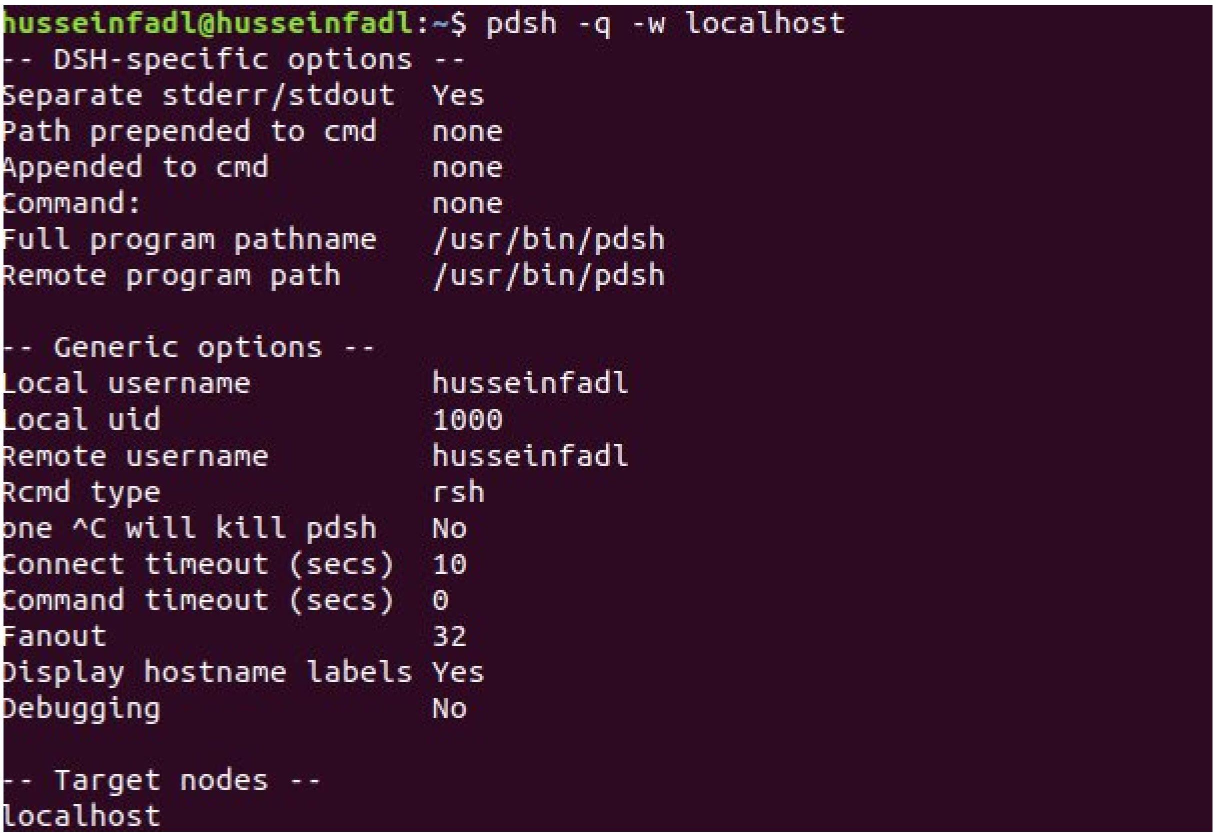
```
husseinfadl@husseinfadl:~$ hdfs namenode -format
2021-04-13 14:15:15,047 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = husseinfadl/127.0.1.1
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 3.3.0
STARTUP_MSG:   classpath = /usr/local/hadoop/etc/hadoop:/usr/local/hadoop/share/hadoop/commo
n/lib/curator-recipes-4.2.0.jar:/usr/local/hadoop/share/hadoop/common/lib/failureaccess-1.0.
jar:/usr/local/hadoop/share/hadoop/common/lib/jackson-annotations-2.10.3.jar:/usr/local/hado
op/share/hadoop/common/lib/jersey-json-1.19.jar:/usr/local/hadoop/share/hadoop/common/lib/ke
rb-core-1.0.1.jar:/usr/local/hadoop/share/hadoop/common/lib/jackson-core-asl-1.9.13.jar:/usr
/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar:/usr/local/hadoop/share/hadoo
p/common/lib/metrics-core-3.2.4.jar:/usr/local/hadoop/share/hadoop/common/lib/kerby-xdr-1.0.
1.jar:/usr/local/hadoop/share/hadoop/common/lib/j2objc-annotations-1.1.jar:/usr/local/hadoop
/share/hadoop/common/lib/javax.servlet-api-3.1.0.jar:/usr/local/hadoop/share/hadoop/common/l
ib/curator-client-4.2.0.jar:/usr/local/hadoop/share/hadoop/common/lib/kerb-server-1.0.1.jar:
/usr/local/hadoop/share/hadoop/common/lib/zookeeper-jute-3.5.6.jar:/usr/local/hadoop/share/h
adoop/common/lib/slf4j-api-1.7.25.jar:/usr/local/hadoop/share/hadoop/common/lib/curator-fram
```

This step should end by shutting down the namenode as follows:



```
2021-04-13 14:16:46,808 INFO namenode.FSImageFormatProtobuf: Image file /usr/local/hadoop_tm
p/hdfs/namenode/current/fsimage.ckpt_0000000000000000000 of size 406 bytes saved in 0 second
s .
2021-04-13 14:16:46,900 INFO namenode.NNStorageRetentionManager: Going to retain 1 images wi
th txid >= 0
2021-04-13 14:16:46,909 INFO namenode.FSImage: FSImageSaver clean checkpoint: txid=0 when me
et shutdown.
2021-04-13 14:16:46,909 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at husseinfadl/127.0.1.1
************************************************************/
```

15. Before starting the Hadoop Distributed File System (hdfs), we need to make sure that the rcmd type is “ssh” not “rsh” when we type the following command
    pdsh -q -w localhost



```
husseinfadl@husseinfadl:~$ pdsh -q -w localhost
-- DSH-specific options --
Separate stderr/stdout  Yes
Path prepended to cmd   none
Appended to cmd         none
Command:                none
Full program pathname   /usr/bin/pdsh
Remote program path     /usr/bin/pdsh

-- Generic options --
Local username          husseinfadl
Local uid               1000
Remote username         husseinfadl
Rcmd type               rsh
one ^C will kill pdsh   No
Connect timeout (secs)  10
Command timeout (secs)  0
Fanout                  32
Display hostname labels Yes
Debugging               No

-- Target nodes --
localhost
```

16. If the rcmd type is “rsh” as in the above figure, type the following commands:

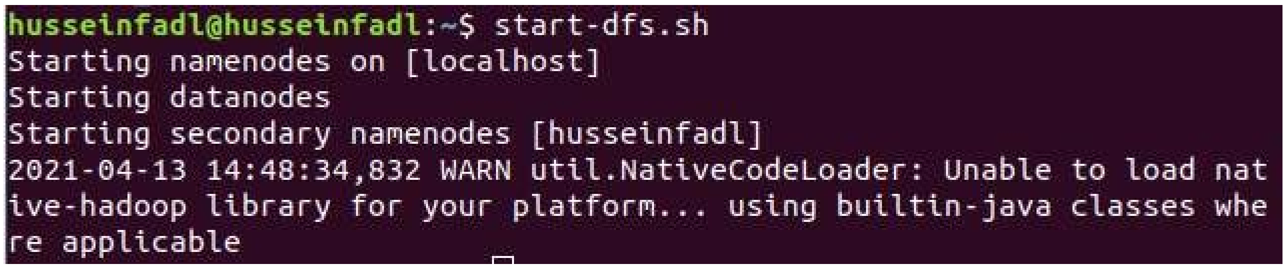
    export PDSH_RCMD_TYPE=ssh

    cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys chmod 0600 ~/.ssh/authorized_keys

    Run Step 16 again to check that the rcmd type is now ssh.
    If not, skip that step.

17. Start the HDFS System using the command.

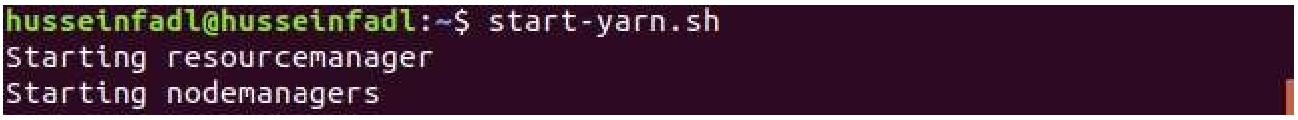
    start-dfs.sh



```
husseinfadl@husseinfadl:~$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [husseinfadl]
2021-04-13 14:48:34,832 WARN util.NativeCodeLoader: Unable to load nat
ive-hadoop library for your platform... using builtin-java classes whe
re applicable
```

18. Start the YARN using the command

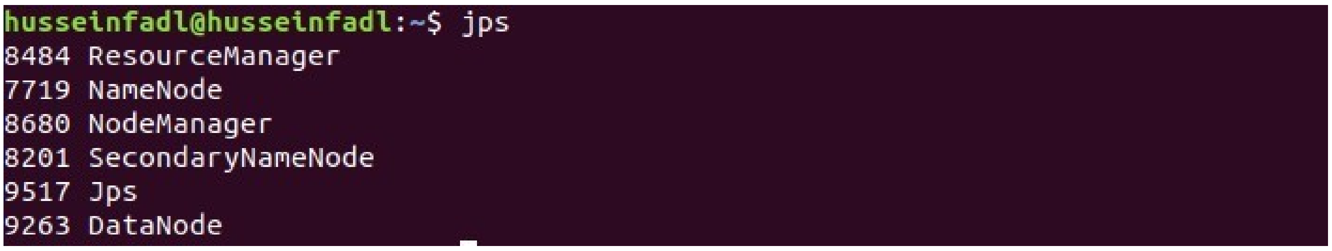
    start-yarn.sh



```
husseinfadl@husseinfadl:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
```

19. Type the following command. You should see an output similar to the one in the following figure.

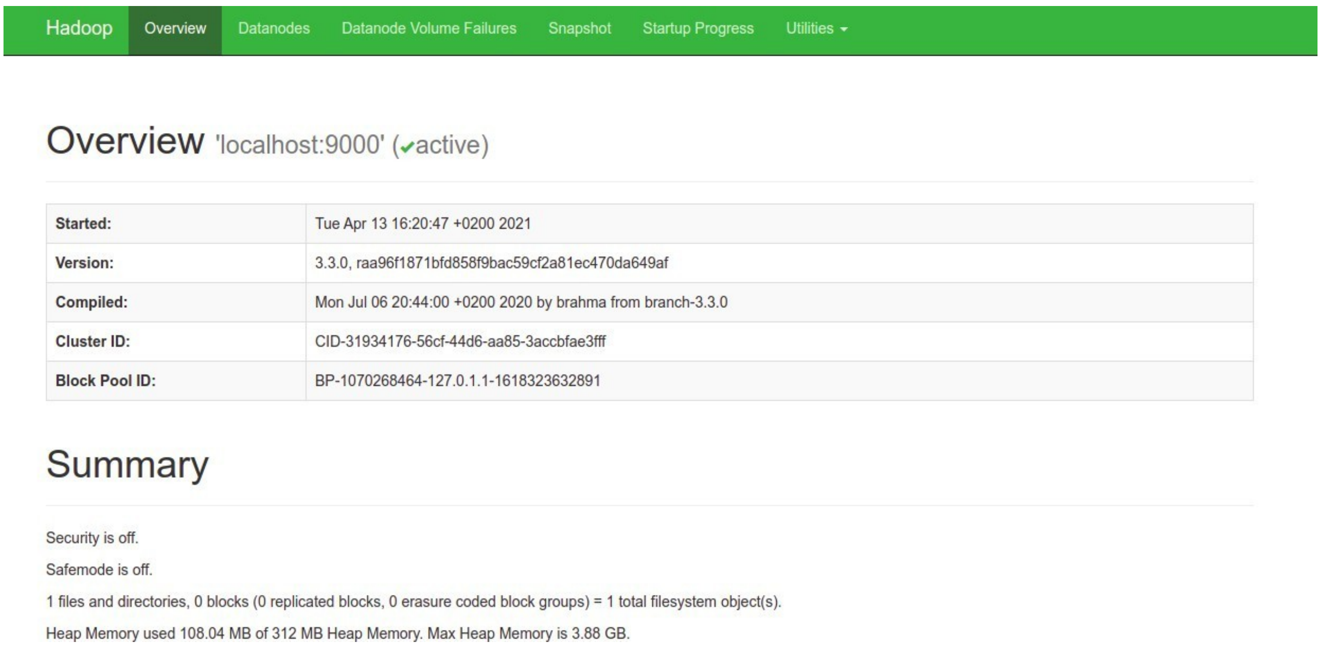
    jps



```
husseinfadl@husseinfadl:~$ jps
8484 ResourceManager
7719 NameNode
8680 NodeManager
8201 SecondaryNameNode
9517 Jps
9263 DataNode
```

Make sure these nodes are listed: (ResourceManager, NameNode, NodeManager, SecondaryNameNode, Jps and DataNode).

20. Go to localhost:9870 from the browser. You should expect the following



Overview 'localhost:9000' (✓active)

Started:	Tue Apr 13 16:20:47 +0200 2021
Version:	3.3.0, raa96f1871bfd858f9bac59cf2a81ec470da649af
Compiled:	Mon Jul 06 20:44:00 +0200 2020 by brahma from branch-3.3.0
Cluster ID:	CID-31934176-56cf-44d6-aa85-3accbfae3fff
Block Pool ID:	BP-1070268464-127.0.1.1-1618323632891

Summary

Security is off.

Safemode is off.

1 files and directories, 0 blocks (0 replicated blocks, 0 erasure coded block groups) = 1 total filesystem object(s).

Heap Memory used 108.04 MB of 312 MB Heap Memory. Max Heap Memory is 3.88 GB.

Steps to run WordCount Program on Hadoop:

1. Make sure Hadoop and Java are installed properly

    hadoop
    version javac
    - version

2. Create a directory on the Desktop named Lab and inside it create two folders; one called "Input" and the other called "tutorial_classes".
   [You can do this step using GUI normally or through terminal commands]

```
cd  Desktop

mkdir      Lab
mkdir
Lab/Input
mkdir  Lab/tutorial_classes
```

3. Add the file attached with this document “WordCount.java” in the directory Lab

4. Add the file attached with this document “input.txt” in the directory Lab/Input.

5. Type the following command to export the hadoop classpath into bash.

   ```
   export  HADOOP_CLASSPATH=$(hadoop  classpath)
   ```

   Make sure it is now exported.

   ```
   echo  $HADOOP_CLASSPATH
   ```

6. It is time to create these directories on HDFS rather than locally. Type the following commands.

   ```
   hadoop  fs  -mkdir  /WordCountTutorial hadoop  fs
   -mkdir
   /WordCountTutorial/Input
   hadoop  fs  -put  Lab/Input/input.txt  /WordCountTutorial/Input
   ```

7. Go to localhost:9870 from the browser, Open “Utilities → Browse File System” and you should see the directories and昀椀les we placed in the 昀椀le system.

8. Then, back to local machine where we will compile the WordCount.java file. Assuming we are currently in the Desktop directory.

   ```
   cd  Lab
   javac  -classpath  $HADOOP_CLASSPATH  -d
   tutorial_classes  WordCount.java
   ```

   Put the output files in one jar file (There is a dot at the end)

   ```
   jar  -cvf  WordCount.jar  -C  tutorial_classes  .
   ```

9. Now, we run the jar file on Hadoop.

   ```
   hadoop  jar  WordCount.jar  WordCount  /WordCountTutorial/Input
   /WordCountTutorial/Output
   ```

10. Output the result:

    ```
    hadoop  dfs  -cat  /WordCountTutorial/Output/*
    ```

Program:

First Create Indexmapper.java class

```
Packagemr03.inverted_index;

                                        import  org.apache.hadoop.io.LongWritable; import
                                        org.apache.hadoop.io.Text;
                                        import  org.apache.hadoop.mapreduce.Mapper;

                                        import org.apache.hadoop.mapreduce.lib.input.FileSplit;
                                      import  java.io.IOException;

                                        import  java.util.StringTokenizer;


                                        public  class  IndexMapper  extends
                                        Mapper<LongWritable,  Text,  Text,  Text>  {
                                              private  final  Text  wordAtFileNameKey  =  new  Text(); private  final
                                              Text  ONE_STRING  =  new  Text("1");


                                              @Override
                                              protected  void  map(LongWritable  key,  Text  value,
                                                                        Context  context)  throws IOException,
                                        InterruptedException  {
                                                  FileSplit  split  =  (FileSplit) context.getInputSplit();
                                                    StringTokenizer  tokenizer  =  new StringTokenizer(value.toString());
                                                    while  (tokenizer.hasMoreTokens())  { String  fileName
                                                          =
                                        split.getPath().getName().split("\\.")[0];
                                                          //remove  special  char  using
                                                         //  tokenizer.nextToken().replaceAll("[^a- zA-
                                      Z]",  "").toLowerCase()
                                                          //check  for  empty  words
                                                          wordAtFileNameKey.set(tokenizer.nextToken ()  +
                                       "@"  +  fileName);
                                                          context.write(wordAtFileNameKey, ONE_STRING);
                                                    }
                                              }
                                        }
```

IndexReducer.java

```
package mr03.inverted_index;

                                  import org.apache.hadoop.io.Text;
```

```
                    import  org.apache.hadoop.mapreduce.Reducer;

 public  class  IndexReducer  extends  Reducer<Text,  Text,  Text, Text>
 {
         private  final  Text  allFilesConcatValue  =  new  Text(); @Override
protected  void  reduce(Text  key,  Iterable<Text>  values,
                                                   Context  context)  throws
 java.io.IOException  ,InterruptedException  {
                                StringBuilder  filelist  =  new StringBuilder("");
                 for(Text  value:values)  { filelist.append(value.toString()).append(";");
                 }
                 allFilesConcatValue.set(filelist.toString()); context.write(key,
                 allFilesConcatValue);
         };
 }
```

IndexDriver.java

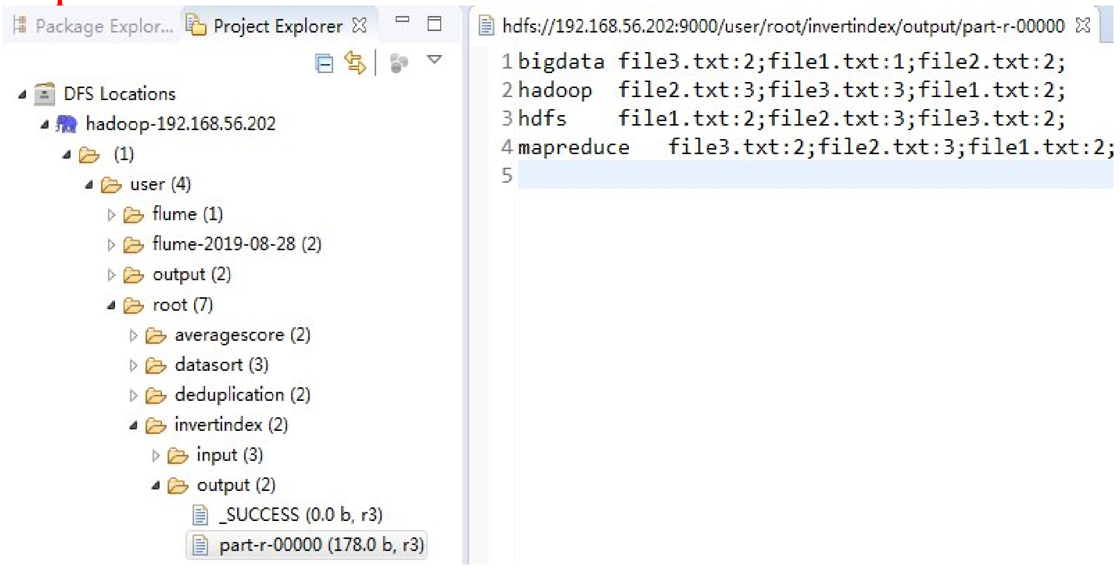
```
package mr03.inverted_index;

 import        org.apache.hadoop.fs.FileSystem;        import
 org.apache.hadoop.mapreduce.Job; import
 org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat; import
 org.apache.hadoop.conf.Configuration;
 import  org.apache.hadoop.fs.Path; import
 org.apache.hadoop.io.Text;

 public  class  IndexDriver  {

         public  static  void  main(String[]  args)  throws  Exception
         {
               if  (args.length  !=  2)  {
                         System.err.println("Usage  IndexDriver
                         <input_dir>
 <output_dir>");
                       System.exit(2);
                 }
                 Configuration  conf  =  new  Configuration(); String  input  =
               args[0];
                         String  output  = args[1];
```

```
                    FileSystem  fs  =  FileSystem.get(conf);
            boolean  exists  =  fs.exists(new Path(output));
                        if(exists)  {
                                fs.delete(new  Path(output),  true);
                    }
                    Job  job  =  Job.getInstance(conf); job.setJarByClass(IndexDriver.class);

                    job.setMapperClass(IndexMapper.class); job.setCombinerClass(IndexCombiner.class);
                    job.setReducerClass(IndexReducer.class);

                    job.setOutputKeyClass(Text.class); job.setOutputValueClass(Text.class);

                    FileInputFormat.addInputPath(job,  new  Path(input));
                    FileOutputFormat.setOutputPath(job,  new Path(output));
                    System.exit(job.waitForCompletion(true)?
                                                0:1);

                }

        }
```

IndexCombiner.java

```
package mr03.inverted_in
de  x;

                        import  java.io.IOException;  import
                        org.apache.hadoop.io.Text;
                        import  org.apache.hadoop.mapreduce.Reducer;


                        public  class  IndexCombiner  extends  Reducer<Text,  Text,  Text, Text>
                        {
                            private  final  Text  fileAtWordFreqValue  =  new  Text(); @Override
                            protected  void  reduce(Text  key,  java.lang.Iterable<Text> values,
                                                    Context  context)  throws  IOException
                        ,InterruptedException  { int  sum  =
                                0;
                                    for(Text  value:values)
                                            {
                                            sum  +=  Integer.parseInt(value.toString());
                                }
                                int  splitIndex  =  key.toString().indexOf("@");
fileAtWordFreqValue.set(key.toString().substring(splitIndex+1)
+":"
+sum);
            key.set(key.toString().substring(0,splitInde x));
            context.write(key,  fileAtWordFreqValue);
      }
}
```

Output:



```
bigdata file3.txt:2;file1.txt:1;file2.txt:2;
hadoop  file2.txt:3;file3.txt:3;file1.txt:2;
hdfs    file1.txt:2;file2.txt:3;file3.txt:2;
mapreduce   file3.txt:2;file2.txt:3;file1.txt:2;
```

Experiment 2. Process big data in HBase

Aim: To create a table and process the big data in Hbase

Resources: Hadoop, oracle virtual box, Hbase
Theory: Hbase is an open source and sorted map data built on Hadoop. It is column oriented and horizontally scalable.

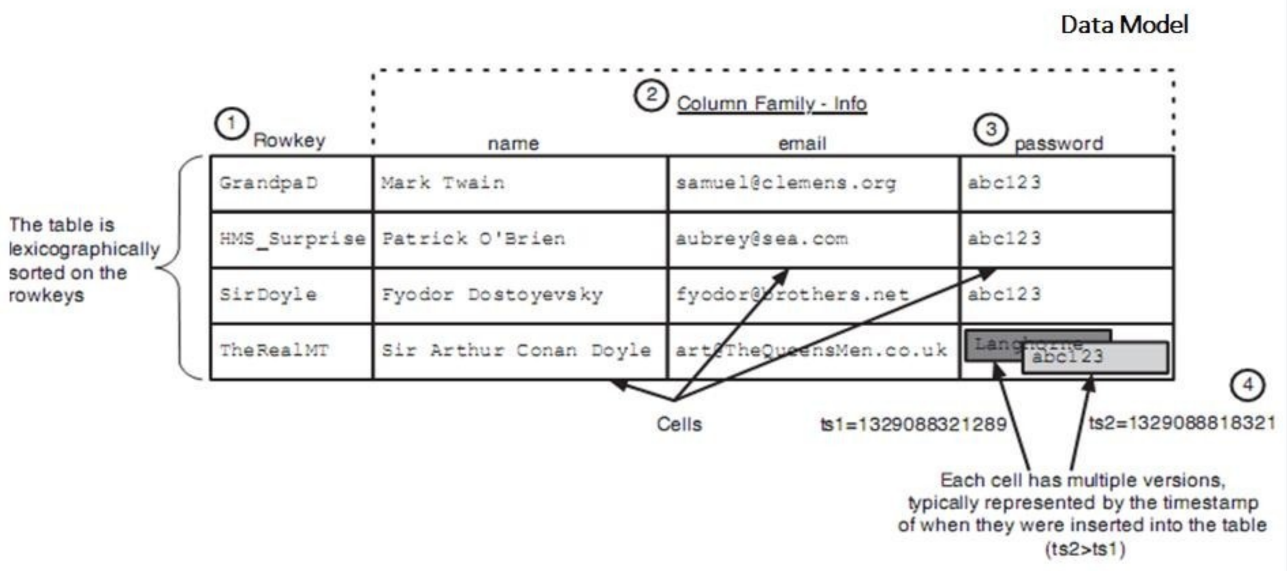
It is based on Google's Big Table. It has set of tables which keep data in key value format. Hbase is well suited for sparse data sets which are very common in big data use cases. Hbase provides APIs enabling development in practically any programming language. It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in the Hadoop File System.

- RDBMS get exponentially slow as the data becomes large
- Expects data to be highly structured, i.e. ability to fit in a well-defined schema
- Any change in schema might require a downtime
- For sparse datasets, too much of overhead of maintaining NULL values

Features of Hbase

- Horizontally scalable: You can add any number of columns anytime.
- Automatic Failover: Automatic failover is a resource that allows a system administrator to automatically switch data handling to a standby system in the event of system compromise
- Integrations with Map/Reduce framework: Al the commands and java codes internally implement Map/ Reduce to do the task and it is built over Hadoop Distributed File System.
- Sparse, distributed, persistent, multidimensional sorted map, which is indexed by row key, column key, and timestamp.
- Often referred as a key value store or column family-oriented database, or storing versioned maps of maps.
- Fundamentally, it's a platform for storing and retrieving data with random access.
- It doesn't care about data types(storing an integer in one row and a string in another for the same column).
- It doesn't enforce relationships within your data.
- It is designed to run on a cluster of computers, built using commodity hardware.

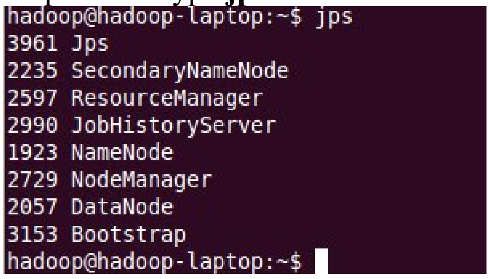
Cloudera VM is recommended as it has Hbase preinstalled on it.
Starting Hbase: Type Hbase shell in terminal to start the Hbase.



The coordinates used to identify data in an HBase table are ① rowkey, ② column family, ③ column qualifier, and ④ version.

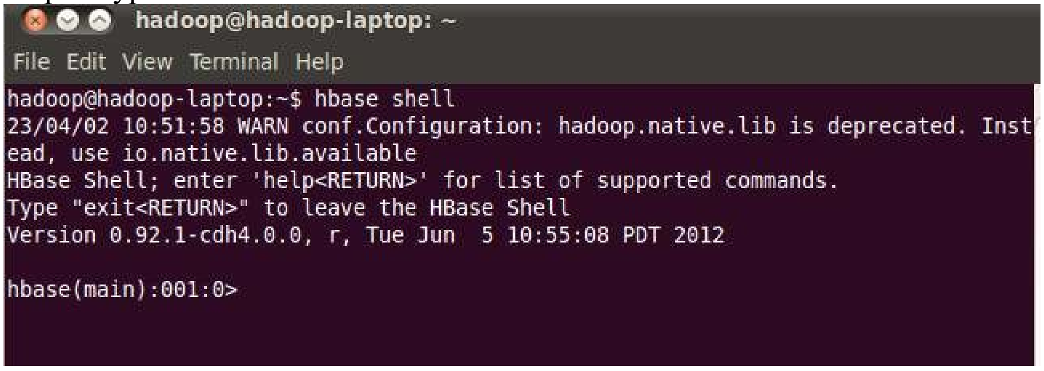
Cloudera VM is recommended as it has Hbase preinstalled on it.

Hbase commands:
Step 1:First go to terminal and type **StartCDH.sh**
Step 2:Next type **jps** command in the terminal



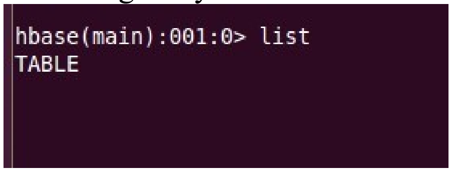
```
hadoop@hadoop-laptop:~$ jps
3961 Jps
2235 SecondaryNameNode
2597 ResourceManager
2990 JobHistoryServer
1923 NameNode
2729 NodeManager
2057 DataNode
3153 Bootstrap
hadoop@hadoop-laptop:~$
```

Step 3:Type **hbase shell**



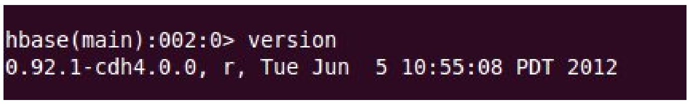
```
hadoop@hadoop-laptop:~$ hbase shell
23/04/02 10:51:58 WARN conf.Configuration: hadoop.native.lib is deprecated. Inst
ead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.92.1-cdh4.0.0, r, Tue Jun  5 10:55:08 PDT 2012

hbase(main):001:0>
```

Step 4:hbase(main):001:0> **list**
List will gives you list of tables in Hbase



```
hbase(main):001:0> list
TABLE
```

Step 5:hbase(main):001:0>**version**



```
hbase(main):002:0> version
0.92.1-cdh4.0.0, r, Tue Jun  5 10:55:08 PDT 2012
```

Version will gives you the version of hbase

Create Table Syntax

CREATE 'name_space:table_name', 'column_family’

hbase(main):011:0> create 'newtbl','knowledge'
hbase(main):011:0>describe 'newtbl'
hbase(main):011:0>status
1 servers, 0 dead, 15.0000 average load

HBase – Using PUT to Insert data to Table

To insert data into the HBase table use PUT command, this would be similar to insert statement on RDBMS but the syntax is completely different. In this article I will describe how to insert data into HBase table with examples using PUT command from the HBase shell.

HBase PUT command syntax

Below is the syntax of PUT command which is used to insert data (rows and columns) into a HBase table.

HBase PUT command syntax

Below is the syntax of PUT command which is used to insert data (rows and columns) into a HBase table.

put '<name_space:table_name>', '<row_key>' '<cf:column_name>', '<value>'

```
hbase(main):015:0> put 'newtbl','r1','knowledge:sports','cricket'
0 row(s) in 0.0150 seconds

hbase(main):016:0> put 'newtbl','r1','knowledge:science','chemistry'
0 row(s) in 0.0040 seconds

hbase(main):017:0> put 'newtbl','r1','knowledge:science','physics'
0 row(s) in 0.0030 seconds

hbase(main):018:0> put 'newtbl','r2','knowledge:economics','macroeconomics'
0 row(s) in 0.0030 seconds

hbase(main):019:0> put 'newtbl','r2','knowledge:music','songs'
0 row(s) in 0.0170 seconds
hbase(main):020:0> scan 'newtbl'
ROW                COLUMN+CELL
 r1                column=knowledge:science, timestamp=1678807827189, value
                  =physics
 r1                column=knowledge:sports, timestamp=1678807791753, value=
                  cricket
```

```
 r2                column=knowledge:economics, timestamp=1678807854590, value=macroeconomics
 r2                column=knowledge:music, timestamp=1678807877340, value=s
                  ongs
2 row(s) in 0.0250 seconds
```

To retrieve only the row1

hbase(main):023:0> **get 'newtbl', 'r1'**

Output:

```
COLUMN CELL
 knowledge:science     timestamp=1678807827189, value=physics
 knowledge:sports      timestamp=1678807791753, value=cricket
2 row(s) in 0.0150 seconds.
```

```
hbase(main):025:0> disable 'newtbl'
0 row(s) in 1.2760 seconds
```

Verification:

After disabling the table, you can still sense its existence through **list** and **exists** commands. You cannot scan it. It will give you the following error.

```
hbase(main):028:0> scan 'newtbl'
ROW       COLUMN + CELL
ERROR: newtbl is disabled.
```

is_disabled

This command is used to find whether a table is disabled. Its syntax is as follows.

```
hbase> is_disabled 'table name'
```

```
hbase(main):031:0> is_disabled 'newtbl'
true
0 row(s) in 0.0440 seconds
```

disable_all

This command is used to disable all the tables matching the given regex. The syntax for **disable_all** command is given below.

```
hbase> disable_all 'r.*'
```

Suppose there are 5 tables in HBase, namely raja, rajani, rajendra, rajesh, and raju. The following code will disable all the tables starting with **raj.**

```
hbase(main):002:07> disable_all 'raj.*'
raja
rajani
```

```
rajendra
rajesh
raju
Disable the above 5 tables (y/n)?
y
5 tables successfully disabled
```

Enabling a Table using HBase Shell

Syntax to enable a table:

```
enable ‘newtbl’
```

Example

Given below is an example to enable a table.

```
hbase(main):005:0> enable 'newtbl'
0 row(s) in 0.4580 seconds
```

Verification

After enabling the table, scan it. If you can see the schema, your table is successfully enabled.

```
hbase(main):006:0> scan 'newtbl'
```

is_enabled

This command is used to find whether a table is enabled. Its syntax is as follows:

```
hbase> is_enabled 'table name'
```

The following code verifies whether the table named **emp** is enabled. If it is enabled, it will return true and if not, it will return false.

```
hbase(main):031:0> is_enabled 'newtbl'
true
0 row(s) in 0.0440 seconds
```

describe

This command returns the description of the table. Its syntax is as follows:

```
hbase> describe 'table name'
```

```
hbase(main):006:0> describe 'newtbl'
  DESCRIPTION
    ENABLED
```

Experiment: 3 Store and retrieve data in Pig

Aim: To perform storing and retrieval of big data using Apache pig

Resources: Apache pig
Theory:

Pig is a platform that works with large data sets for the purpose of analysis. The Pig dialect is called Pig Latin, and the Pig Latin commands get compiled into MapReduce jobs that can be run on a suitable platform, like Hadoop.

Apache Pig is a platform for analyzing large data sets that consists of a high- level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

At the present time, Pig's infrastructure layer consists of a compiler that produces sequences of Map-Reduce programs, for which large-scale parallel implementations already exist (e.g., the Hadoop subproject). Pig's language layer currently consists of a textual language called Pig Latin, which has the following key properties:

- **Ease of programming.** It is trivial to achieve parallel execution of simple, "embarrassingly parallel" data analysis tasks. Complex tasks comprised of multiple interrelated data transformations are explicitly encoded as data flow sequences, making them easy to write, understand, and maintain.
- **Optimization opportunities.** The way in which tasks are encoded permits the system to optimize their execution automatically, allowing the user to focus on semantics rather than efficiency.
- **Extensibility.** Users can create their own functions to do special-purpose processing.
- Pig Latin – Relational Operations
- The following table describes the relational operators of Pig Latin.

Operator	Description
Loading and Storing	
LOAD	To Load the data from the file system (local/HDFS) into a relation.
STORE	To save a relation to the file system (local/HDFS).

Filtering	
FILTER	To remove unwanted rows from a relation.
DISTINCT	To remove duplicate rows from a relation.
FOREACH, GENERATE	To generate data transformations based on columns of data.
STREAM	To transform a relation using an external program.
Grouping and Joining	
JOIN	To join two or more relations.
COGROUP	To group the data in two or more relations.
GROUP	To group the data in a single relation.
CROSS	To create the cross product of two or more relations.
Sorting	
ORDER	To arrange a relation in a sorted order based on one or more 昀椀elds (ascending or descending).
LIMIT	To get a limited number of tuples from a relation.
Combining and Splitting	
UNION	To combine two or more relations into a single relation.
SPLIT	To split a single relation into two or more relations.
Diagnostic Operators	
DUMP	To print the contents of a relation on the console.
DESCRIBE	To describe the schema of a relation.
EXPLAIN	To view the logical, physical, or MapReduce execution plans to compute a relation.
ILLUSTRATE	To view the step-by-step execution of a series of statements.

For the given Student dataset and Employee dataset,perform Relational operations like Loading, Storing, Diagnostic Operations (Dump, Describe, Illustrate & Explain) in Hadoop Pig framework using Cloudera

Student ID	First Name	Age	City	CGPA
001	Jagruthi	21	Hyderabad	9.1
002	Praneeth	22	Chennai	8.6
003	Sujith	22	Mumbai	7.8
004	Sreeja	21	Bengaluru	9.2
005	Mahesh	24	Hyderabad	8.8
006	Rohit	22	Chennai	7.8
007	Sindhu	23	Mumbai	8.3

Employee ID	Name	Age	City
001	Angelina	22	LosAngeles
002	Jackie	23	Beijing
003	Deepika	22	Mumbai
004	Pawan	24	Hyderabad
005	Rajani	21	Chennai
006	Amitabh	22	Mumbai

Step-1: **Create a Directory**in HDFS with the name **pigdir** in the required path using **mkdir**:

```
$ hdfs dfs -mkdir /bdalab/pigdir
```

Step-2: The input 昀椀le of Pig contains each tuple/record in individual lines with the entities separated by a delimiter (“,”).

In the local file system, **create an input file student_data.txt** containing data as shown below.	In the local file system, **create an input file employee_data.txt** containing data as shown below.
001,Jagruthi,21,Hyderabad,9.1 002,Praneeth,22,Chennai,8.6 003,Sujith,22,Mumbai,7.8 004,Sreeja,21,Bengaluru,9.2 005,Mahesh,24,Hyderabad,8.8 006,Rohit,22,Chennai,7.8 007,Sindhu,23,Mumbai,8.3	001,Angelina,22,LosAngeles 002,Jackie,23,Beijing 003,Deepika,22,Mumbai 004,Pawan,24,Hyderabad 005,Rajani,21,Chennai 006,Amitabh,22,Mumbai

Step-3: **Move the file** from the local fi le system to HDFS using **put (Or) copy From Local** command and verify using -cat command

To get the path of the file student_data.txt type the below command readlink -f student_data.txt

```
$ hdfs dfs -put /home/hadoop/Desktop/student_data.txt  /bdalab/pigdir/
$ hdfs dfs -cat /bdalab/pigdir/student_data

$ hdfs dfs -put /home/hadoop/Desktop/employee_data  /bdalab/pigdir/
```

Step-4: **Apply Relational Operator – LOAD to load the data** from the file student_data.txt into Pig by executing the following Pig Latin statement in the **Grunt shell**. Relational Operators are **NOT case sensitive.**

$ pig => will direct to **grunt> shell**

```
grunt> student = LOAD ' /bdalab/pigdir/student_data.txt' USING PigStorage(',')as
( id:int, name:chararray, age:int, city:chararray, cgpa:double );

grunt>employee = LOAD ' /bdalab/pigdir/employee_data.txt'
        USING PigStorage(',')as ( id:int, name:chararray, age:int,
city:chararray);
```

Step-5: **Apply Relational Operator – STORE** to **Store the relation** in the HDFS directory "/pig_output/" as shown below.

```
grunt> STORE student INTO ' /bdalab/pigdir/pig_output/ ' USING Pig Storage (',');
grunt> STORE employee INTO ' /bdalab/pigdir/pig_output/ ' USING Pig Storage (',');
```

Step-6: **Verify the stored data** as shown below

```
$ hdfs dfs -ls /bdalab/pigdir/pig_output/

$ hdfs dfs -cat /bdalab/pigdir/pig_output/part-m-00000
```

Step-7: **Apply Relational Operator – Diagnostic Operator – DUMP to Print the contents of the relation**.

```
grunt> Dump student
grunt> Dump employee
```

Step-8: **Apply Relational Operator – Diagnostic Operator – DESCRIBE toView the schema of a relation**.

```
grunt> Describe student
grunt> Describe employee
```

Step-9: **Apply Relational Operator – Diagnostic Operator – EXPLAIN to Display the logical, physical, and MapReduce execution plans** of a relation using **Explain** operator

```
grunt> Explain student
grunt>Explain employee
```

Step-9: **Apply Relational Operator – Diagnostic Operator – ILLUSTRATE to give the step- by-step execution of a sequence of statements**

```
grunt>Illustrate student
grunt>Illustrate employee
```

Experiment 4: Perform Social media analysis using Cassandra

Aim: To perform the social media data analysis using Cassandra

Resources: Cassandra

Procedure:

- Apache Cassandra is an open-source distributed database management system designed to handle large amounts of data across many commodity servers.
- Cassandra provides high availability with no single point of failure.
- Cassandra offers robust support for clusters spanning multiple data centers, with asynchronous master-less replication allowing low latency operations for all clients.

Cassandra is a distributed database for low latency, high throughput services that handle real time workloads comprising of hundreds of updates per second and tens of thousands of reads per second.

When looking to replace a key-value store with something more capable on the real-time replication and data distribution, research on Dynamo, the CAP theorem and eventual consistency model shows Cassandra 昀椀ts this model quite well. As one learns more about data modeling capabilities, we gradually move towards decomposing data.

If one is coming from a relational database background with strong ACID semantics, then one must take the time to understand the eventual consistency model.

Understand Cassandra's architecture very well and what it does under the hood. With Cassandra 2.0 you get lightweight transaction and triggers, but they are not the same as the traditional database transactions one might be familiar with. For example, there are no foreign key constraints available – it has to be handled by one's own application. Understanding one's use cases and data access patterns clearly before modeling data with Cassandra and to read all the available documentation is a must.

Capture

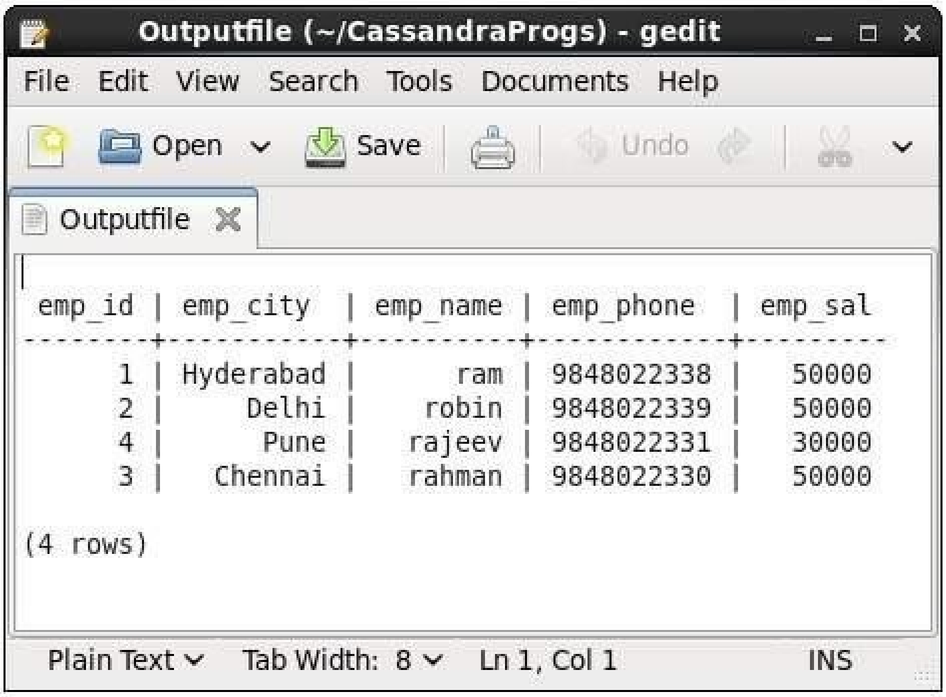
This command captures the output of a command and adds it to a file.

For example, take a look at the following code that captures the output to a file named **Outputfile**.

```
cqlsh> CAPTURE '/home/hadoop/CassandraProgs/Outputfile'
```

When we type any command in the terminal, the output will be captured by the file given. Given below is the command used and the snapshot of the output file.

```
cqlsh:tutorialspoint> select * from emp;
```


Outputfile (~/CassandraProgs) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
Outputfile
Plain Text Tab Width: 8 Ln 1, Col 1 INS

```
 emp_id | emp_city  | emp_name | emp_phone  | emp_sal
--------+-----------+----------+------------+---------
      1 | Hyderabad |      ram | 9848022338 |   50000
      2 |     Delhi |    robin | 9848022339 |   50000
      4 |      Pune |   rajeev | 9848022331 |   30000
      3 |   Chennai |   rahman | 9848022330 |   50000

(4 rows)
```

You can turn capturing off using the following command.

```
cqlsh:tutorialspoint> capture off;
```

Consistency

This command shows the current consistency level, or sets a new consistency level.

```
cqlsh:tutorialspoint> CONSISTENCY
Current consistency level is 1.
```

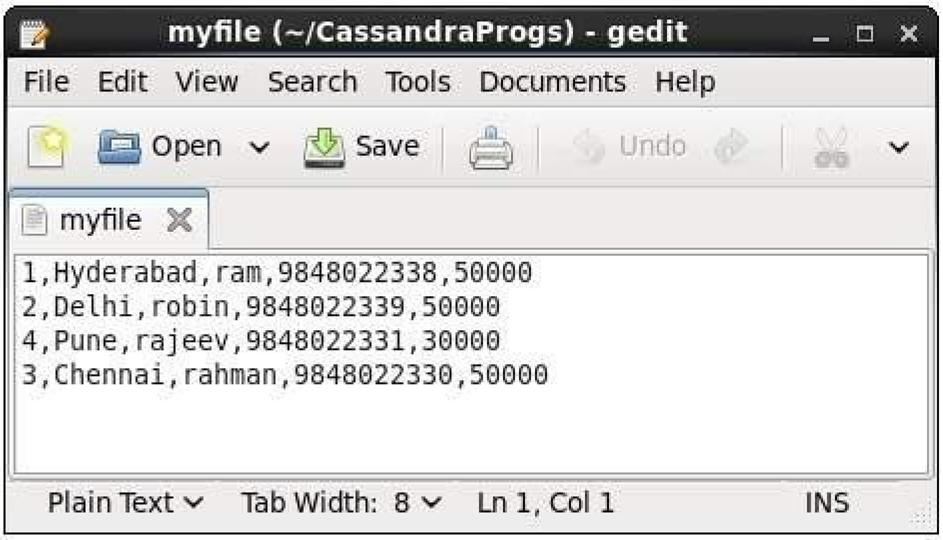
Copy

This command copies data to and from Cassandra to a file.

Given below is an example to copy the table named **emp** to the file **myfile**.

```
cqlsh:tutorialspoint> COPY emp (emp_id, emp_city, emp_name, emp_phone,emp_sal) TO
'myfile';
4 rows exported in 0.034 seconds.
```

If you open and verify the file given, you can find the copied data as shown below.


myfile (~/CassandraProgs) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
myfile
Plain Text
Tab Width: 8
Ln 1, Col 1
INS

```
1,Hyderabad,ram,9848022338,50000
2,Delhi,robin,9848022339,50000
4,Pune,rajeev,9848022331,30000
3,Chennai,rahman,9848022330,50000
```

Describe

This command describes the current cluster of Cassandra and its objects. The variants of this command are explained below.

Describe cluster − This command provides information about the cluster.

```
cqlsh:tutorialspoint> describe cluster;

Cluster: Test Cluster
Partitioner: Murmur3Partitioner

Range ownership:
-658380912249644557 [127.0.0.1]
-2833890865268921414 [127.0.0.1]
-6792159006375935836 [127.0.0.1]
```

Describe Keyspaces − This command lists all the keyspaces in a cluster. Given below is the usage of this command.

```
cqlsh:tutorialspoint> describe keyspaces;

system_traces system tp tutorialspoint
```

Describe tables − This command lists all the tables in a keyspace. Given below is the usage of this command.

```
cqlsh:tutorialspoint> describe tables;
emp
```

Describe table − This command provides the description of a table. Given below is the usage of this command.

```
cqlsh:tutorialspoint> describe table emp;

CREATE TABLE tutorialspoint.emp
( emp_id int PRIMARY KEY,
emp_city text,
emp_name text,
emp_phone varint,
emp_sal varint
) WITH bloom_filter_fp_chance = 0.01
AND caching = '{"keys":"ALL", "rows_per_partition":"NONE"}'
AND comment = ''
AND compaction = {'min_threshold': '4', 'class':
'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy',
'max_threshold': '32'}

AND compression = {'sstable_compression':
'org.apache.cassandra.io.compress.LZ4Compressor'}

AND dclocal_read_repair_chance = 0.1
AND default_time_to_live = 0
AND gc_grace_seconds = 864000
AND max_index_interval = 2048
AND memtable_flush_period_in_ms = 0
AND min_index_interval = 128
AND read_repair_chance = 0.0
AND speculative_retry = '99.0PERCENTILE';
CREATE INDEX emp_emp_sal_idx ON tutorialspoint.emp (emp_sal);
```

Describe Type

This command is used to describe a user-defined data type. Given below is the usage of this command.

```
cqlsh:tutorialspoint> describe type card_details;

CREATE TYPE tutorialspoint.card_details
( num int,
pin int,
name text,
cvv int,
phone set<int>,
mail text
);
```

Describe Types

This command lists all the user-defined data types. Given below is the usage of this command. Assume there are two user-defined data types: **card** and **card_details**.

```
cqlsh:tutorialspoint> DESCRIBE TYPES;

card_details card
```

Expand

This command is used to expand the output. Before using this command, you have to turn the expand command on. Given below is the usage of this command.

```
cqlsh:tutorialspoint> expand on;
cqlsh:tutorialspoint> select * from emp;

 @ Row 1
           +
emp_id | 1
emp_city | Hyderabad
emp_name | ram
emp_phone | 9848022338
emp_sal | 50000

 @ Row 2
           +
emp_id | 2
emp_city | Delhi
emp_name | robin
emp_phone | 9848022339
emp_sal | 50000

 @ Row 3
           +
emp_id | 4
emp_city | Pune
emp_name | rajeev
emp_phone | 9848022331
emp_sal | 30000

 @ Row 4
           +
emp_id | 3
emp_city | Chennai
emp_name | rahman
emp_phone | 9848022330
emp_sal | 50000
(4 rows)
```

Note − You can turn the expand option off using the following command.

```
cqlsh:tutorialspoint> expand off;
Disabled Expanded output.
```

Exit

This command is used to terminate the cql shell.

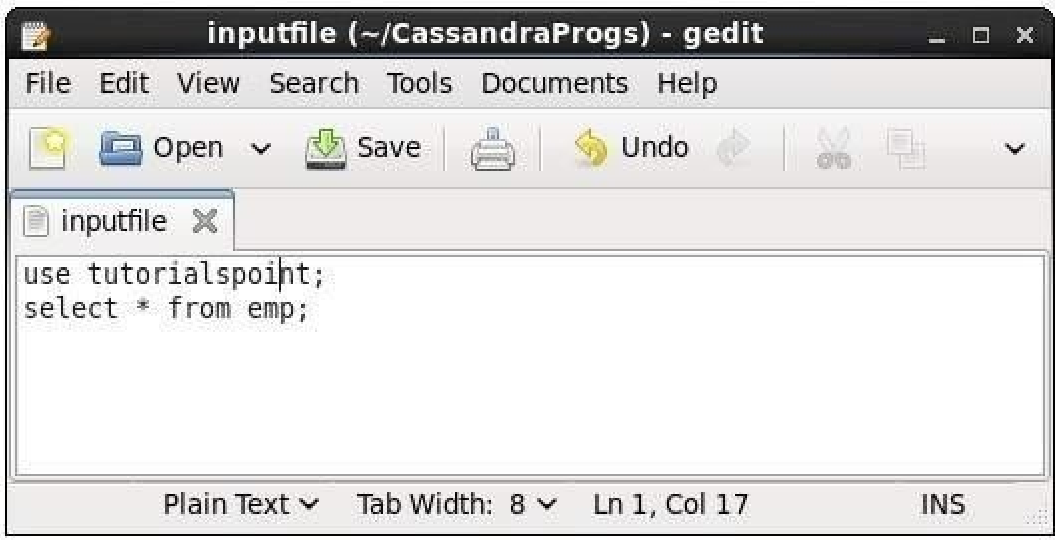
Show

This command displays the details of current cqlsh session such as Cassandra version, host, or data type assumptions. Given below is the usage of this command.

```
cqlsh:tutorialspoint> show host;
Connected to Test Cluster at 127.0.0.1:9042.

cqlsh:tutorialspoint> show version;
[cqlsh 5.0.1 | Cassandra 2.1.2 | CQL spec 3.2.0 | Native protocol v3]
```

Source

Using this command, you can execute the commands in a file. Suppose our input file is as follows –


inputfile (~/CassandraProgs) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
inputfile
Plain Text
Tab Width: 8
Ln 1, Col 17
INS

```
use tutorialspoint;
select * from emp;
```

Then you can execute the file containing the commands as shown below.

```
cqlsh:tutorialspoint> source '/home/hadoop/CassandraProgs/inputfile';

 emp_id | emp_city | emp_name | emp_phone | emp_sal
------------+-----------------+---------------+-------------------+--------------
 1 | Hyderabad | ram | 9848022338 | 50000
2 | Delhi | robin | 9848022339 | 50000
3 | Pune | rajeev | 9848022331 | 30000
4 | Chennai | rahman | 9848022330 | 50000
(4 rows)
```

Experiment 5. Buyer event analytics using Cassandra on suitable product sales data.

Aim: To perform the buyer event analysis using Cassandra on sales data
Resources Required: Apache Hadoop, Apache Cassandra

Theory:

Users can access Cassandra through its nodes using Cassandra Query Language (CQL). CQL treats the database **(Keyspace)** as a container of tables. Programmers use **cqlsh:** a prompt to work with CQL or separate application language drivers.

Clients approach any of the nodes for their read-write operations. That node (coordinator) plays a proxy between the client and the nodes holding the data.

Write Operations

Every write activity of nodes is captured by the **commit logs** written in the nodes. Later the data will be captured and stored in the **mem-table.** Whenever the mem-table is full, data will be written into the **SStable** data file. All writes are automatically partitioned and replicated throughout the cluster. Cassandra periodically consolidates the SSTables, discarding unnecessary data.

Read Operations

During read operations, Cassandra gets values from the mem-table and checks the bloom filter to find the appropriate SSTable that holds the required data.

Apache is an open-source platform. This web server delivers web-related content using the internet. It has gained huge popularity over the last few years, as the most used web server software. Cassandra is a database management system that is open-source. It has the capacity to handle a large amount of data across servers. It was first developed by Facebook for the inbox search feature and was released as an open-source project back in 2008.

The following year, Cassandra became a part of Apache incubation, and combined with Apache, it has reached new heights. To put it in simple terms, Apache Cassandra is a powerful open-source distributed database system that can work efficiently to handle a massive amount of data across multiple servers.

Considering all the features of Apache Cassandra, be it Cassandra fault-tolerance, Cassandra data migration, Cassandra enterprise support, Cassandra cluster optimization and tuning, many organizations have opted for this product. Starting from big players in the market to startups, Cassandra has changed the way of database management. Let’s consider Netflix, the largest online streaming platform. Netflix has successfully provided updated data to its users day after day. Apache Cassandra has undeniably a huge role to play in this feat.

DATA-MODELLING

The way data is modeled is a major difference between Cassandra & MySQL. .

Let us consider a platform where users can post. Now, you have commented on a post of another user. In these two databases, the information will be stored differently. In Cassandra, you can store the data in a single table. The comments for each user is stored in the form of a list(as one single row).

In MySQL, you have to make two tables with one-to-many relationships between them. As MySQL does not permit unstructured data such as a List or a Map, one-to-many relationships are required among these tables.

READ PERFORMANCE

The query to retrieve the comments made by a user(for example ‘5’) in MySQL, will look like this.

SELECT * from Users u, Comments c WHERE u.user_id=c.user_id and user_id=5;
When you utilize indexing in MySQL, it saves the data like a binary tree.
In Cassandra, it is surprisingly simple:
SELECT * from Users WHERE user_id=3;

You only have to store a single row in Cassandra for a specific user_id. It will require just one lookup.

WRITE PERFORMANCE

A search needs to be done with every INSERT/UPDATE/DELETE in MySQL. If you have to update a record with an existing primary key,

1. **It will first search for the row, and**
2. **Then update it**

Cassandra leverages an append-only model. Insert & update have no fundamental difference. If you want to insert a row that comes with the same primary key as an existing row, the row will be replaced. Or, if you update a row with a non-existent primary key, Cassandra will create the row. Cassandra is very fast and stores large swathes of data on commodity hardware without compromising the read efficiency in any way.

<u>TRANSACTIONS</u>

MySQL facilitates ACID transactions like any other Relational Database Management System

- Atomicity
- Consistency
- Isolation
- Durability

On the other hand, Cassandra has certain limitations to provide ACID transactions. Cassandra can achieve consistency if data duplication is not allowed. But, that will kill Cassandra's availability. So, the systems that require ACID transactions must avoid NoSQL databases.

Procedure:

A sample query to insert a record into an Apache Cassandra table is as follows:

```
INSERT INTO employee
      (empid, firstname, lastname,
gender) VALUES
      ('1', 'FN', 'LN', 'M')
```

The same query in MongoDB will have an implementation as follows:

```
db.employee.insert(
      {
        empid: '1',
        firstname:
        'FN', lastname:
        'LN', gender:
        'M'
      }
```

```
cqlsh>
 SELECT      TTL(name)      FROM      learn_cassandra.todo_by_user_email      WHERE
user_email='john@email.com';

 ttl(name)
------
       43

(1 rows)
cqlsh>
 SELECT        *        FROM        learn_cassandra.todo_by_user_email        WHERE
user_email='john@email.com';

 user_email | creation_date | todo_uuid | name
------+-----------+--------+-----

(0 rows)
```

Let's insert a new record:
cqlsh>

```
 INSERT INTO learn_cassandra.todo_by_user_email (user_email, creation_date, name)
VALUES(' ('john@email.com', '2021-03-14 16:07:19.622+0000', 'Insert query');

cqlsh>
 UPDATE learn_cassandra.todo_by_user_email SET
   name = 'Update query'
 WHERE    user_email    =    'john@email.com'    AND    creation_date    =    '2021-03-14
16:10:19.622+0000';
```

2 new rows appear in our table:

```
cqlsh>
 SELECT     *     FROM     learn_cassandra.todo_by_user_email     WHERE
user_email='john@email.com';

 user_email     | creation_date                   | name
___________+_______________________+__________
 john@email.com | 2021-03-14 16:10:19.622000+0000 | Update query
 john@email.com | 2021-03-14 16:07:19.622000+0000 | Insert query

(2 rows)
```

Let’s only update if an entry already exists, by using IF EXISTS:

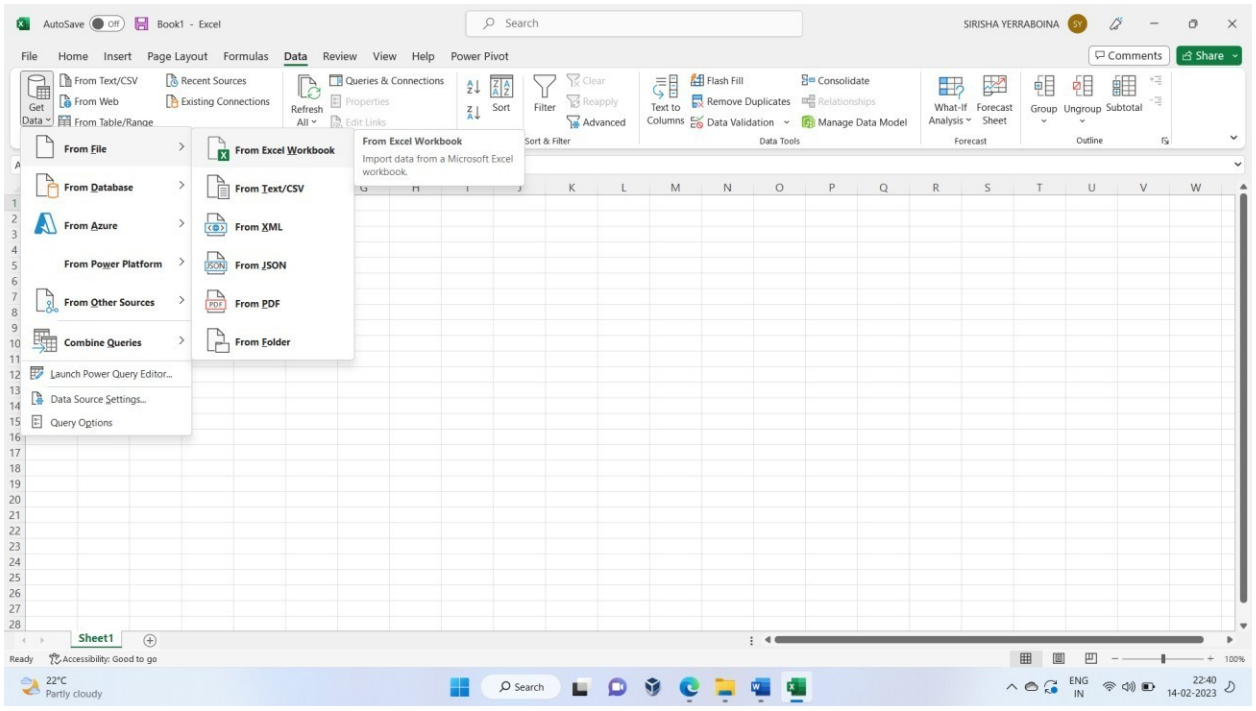
```
cqlsh>
 UPDATE learn_cassandra.todo_by_user_email SET
   name = 'Update query with LWT'
 WHERE    user_email    =    'john@email.com'    AND    creation_date    =    '2021-03-14
16:07:19.622+0000' IF EXISTS;

 [applied]
_ _True_
cqlsh>
 INSERT    INTO    learn_cassandra.todo_by_user_email    (user_email,creation_date,name)
VALUES('john@email.com', toTimestamp(now()), 'Yet another entry') IF NOT EXISTS;

 [applied]
_ _True_
```

<u>Experiment 6</u>: Using a power pivot(Excel) perform the following on any data set.
<u>Aim</u>: To perform the big data analytics using power pivot in Excel
<u>Resources</u>: Microsoft Excel
<u>Theory</u>: Power Pivot is an Excel add-in you can use to perform powerful data analysis and create sophisticated data models. With Power Pivot, you can mash up large volumes of data from various sources, perform information analysis rapidly, and share insights easily.

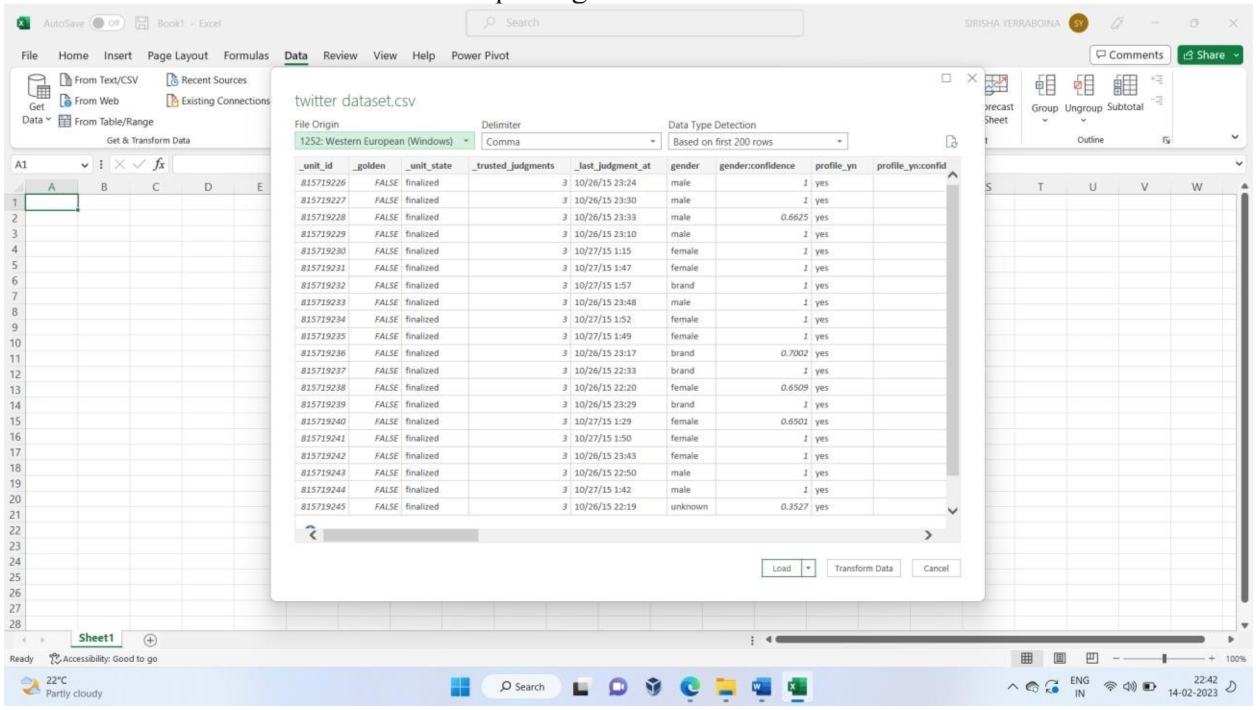
In both Excel and in Power Pivot, you can create a Data Model, a collection of tables with relationships. The data model you see in a workbook in Excel is the same data model you see in the Power Pivot window. Any data you import into Excel is available in Power Pivot, and vice versa.

<u>Procedure:</u>

Open the Microsoft Excel and go to data menu and click get data



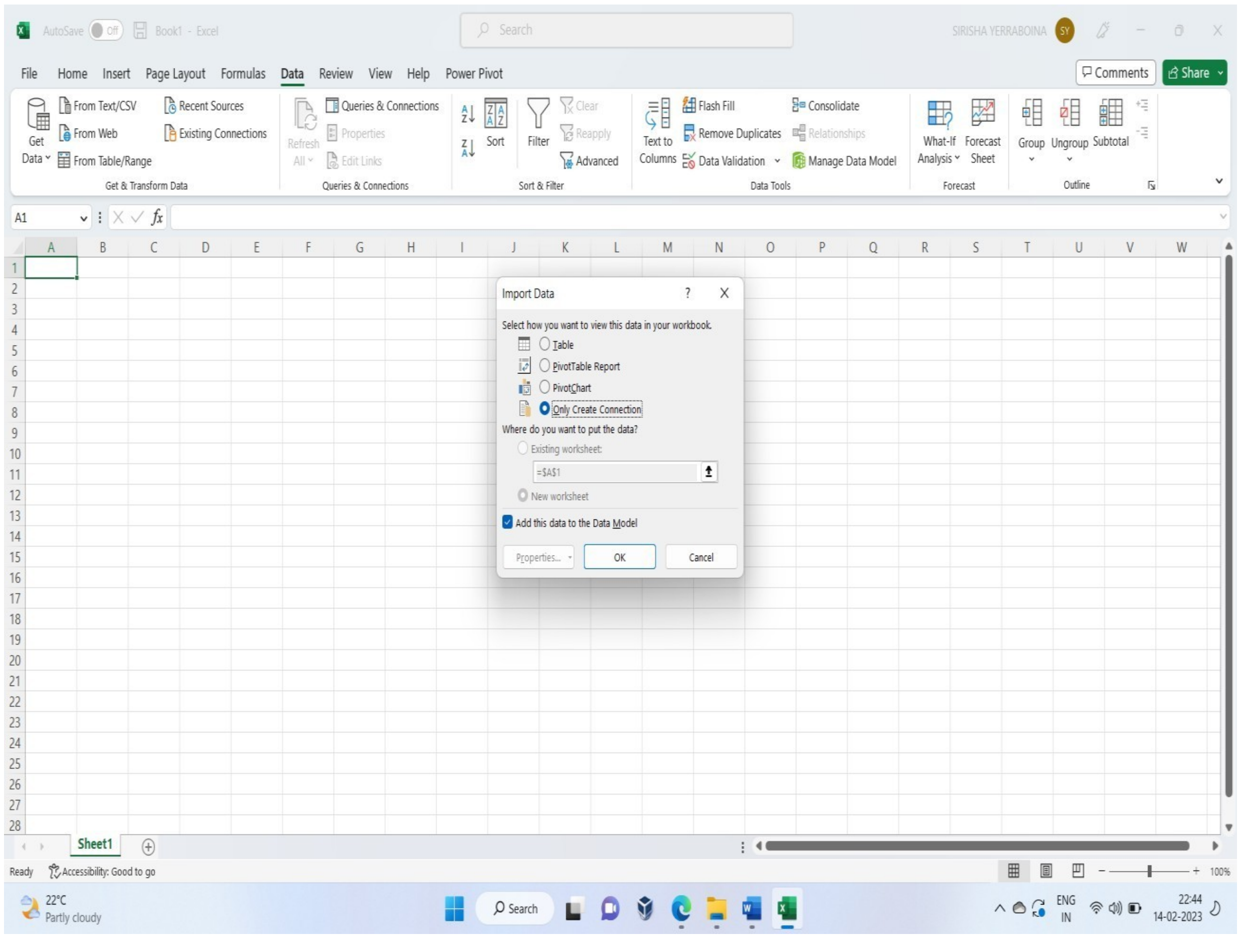
Import the Twitter data set and click load to button
Now from the excel data will starts importing


twitter dataset.csv

File Origin: 1252: Western European (Windows)
Delimiter: Comma
Data Type Detection: Based on first 200 rows

_unit_id	_golden	_unit_state	_trusted_judgments	_last_judgment_at	gender	gender:confidence	profile_yn	profile_yn:confid
815719226	FALSE	finalized	3	10/26/15 23:24	male	1	yes	
815719227	FALSE	finalized	3	10/26/15 23:30	male	1	yes	
815719228	FALSE	finalized	3	10/26/15 23:33	male	0.6625	yes	
815719229	FALSE	finalized	3	10/26/15 23:10	male	1	yes	
815719230	FALSE	finalized	3	10/27/15 1:15	female	1	yes	
815719231	FALSE	finalized	3	10/27/15 1:47	female	1	yes	
815719232	FALSE	finalized	3	10/27/15 1:57	brand	1	yes	
815719233	FALSE	finalized	3	10/26/15 23:48	male	1	yes	
815719234	FALSE	finalized	3	10/27/15 1:52	female	1	yes	
815719235	FALSE	finalized	3	10/27/15 1:49	female	1	yes	
815719236	FALSE	finalized	3	10/26/15 23:17	brand	0.7002	yes	
815719237	FALSE	finalized	3	10/26/15 22:33	brand	1	yes	
815719238	FALSE	finalized	3	10/26/15 22:20	female	0.6509	yes	
815719239	FALSE	finalized	3	10/26/15 23:29	brand	1	yes	
815719240	FALSE	finalized	3	10/27/15 1:29	female	0.6501	yes	
815719241	FALSE	finalized	3	10/27/15 1:50	female	1	yes	
815719242	FALSE	finalized	3	10/26/15 23:43	female	1	yes	
815719243	FALSE	finalized	3	10/26/15 22:50	male	1	yes	
815719244	FALSE	finalized	3	10/27/15 1:42	male	1	yes	
815719245	FALSE	finalized	3	10/26/15 22:19	unknown	0.3527	yes	

Next click create connection and click the check box add to the data model



Import Data

Select how you want to view this data in your workbook.

( ) Table
( ) PivotTable Report
( ) PivotChart
[✓ Only Create Connection]

Where do you want to put the data?

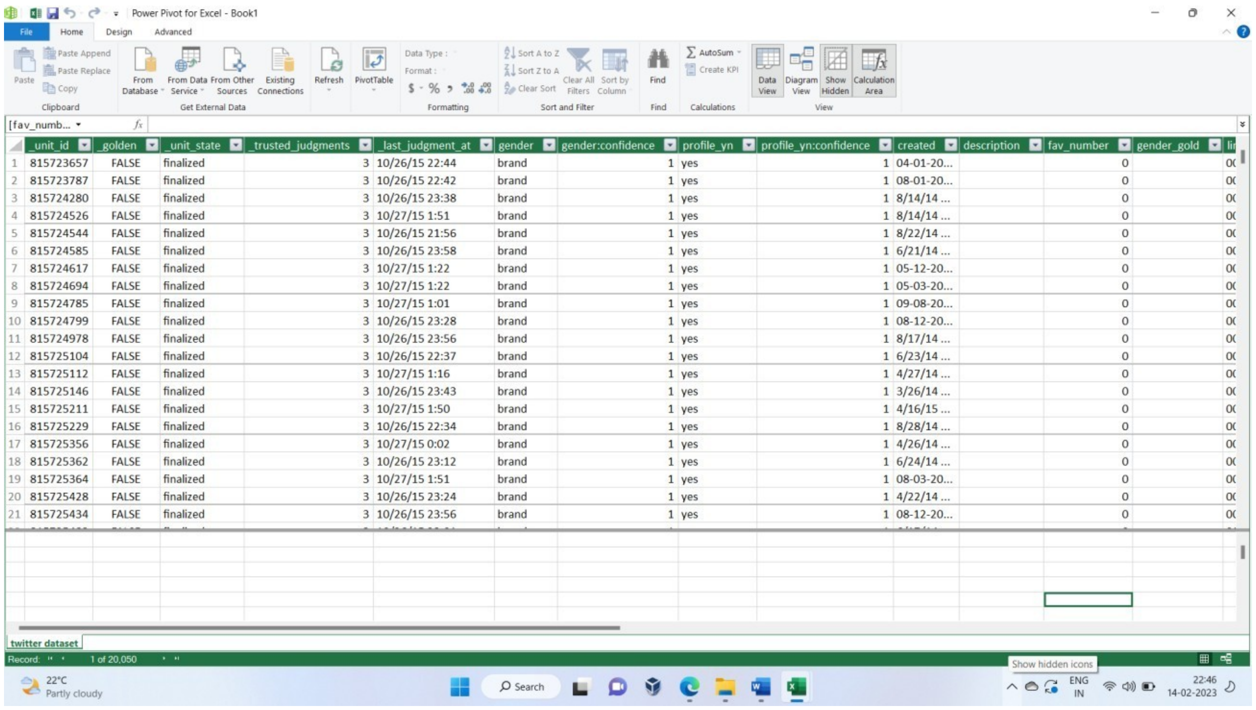
( ) Existing worksheet: =$A$1
( ) New worksheet

[✓] Add this data to the Data Model

Properties... OK Cancel

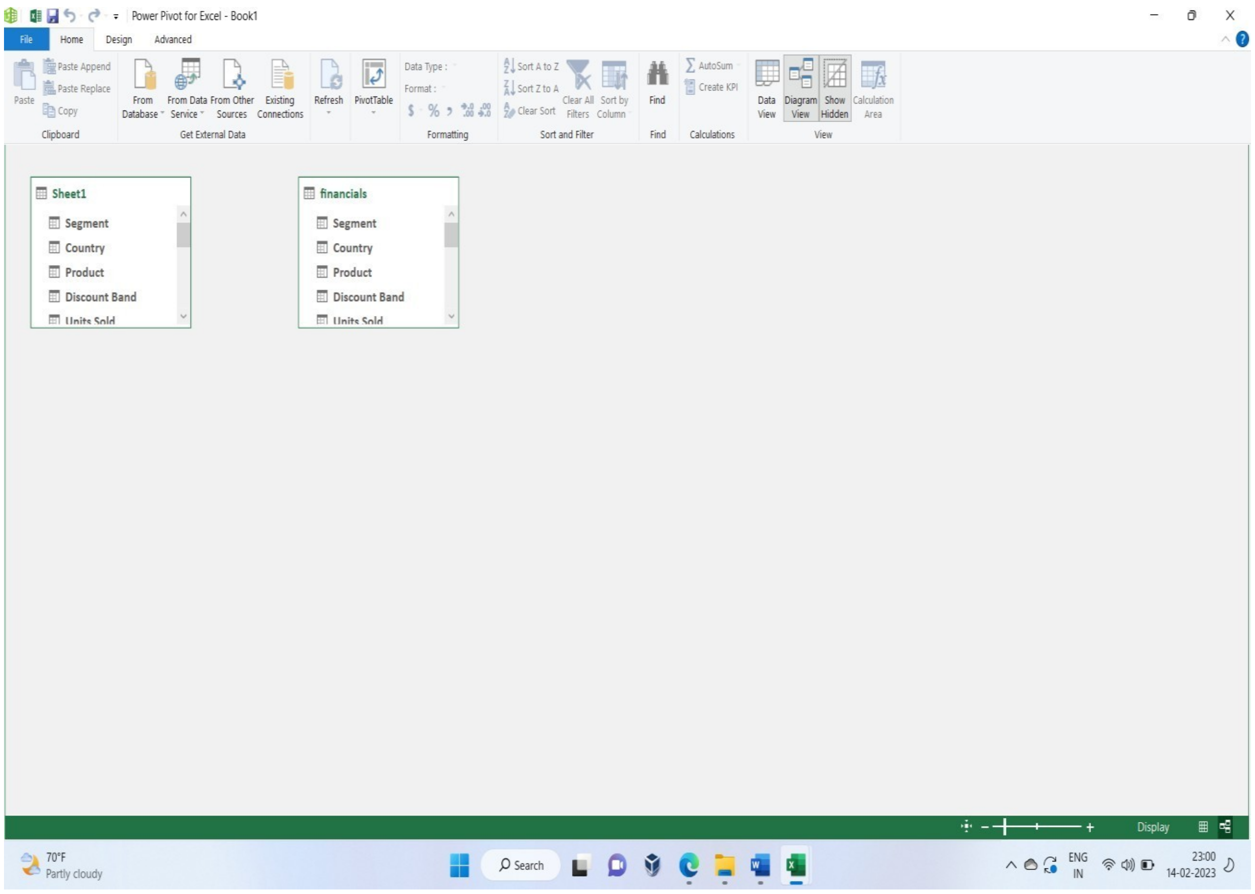
Next click manage data model and see that all the twitter data is loaded as model and close the power pivot window.

Save the excel file as sample.xls

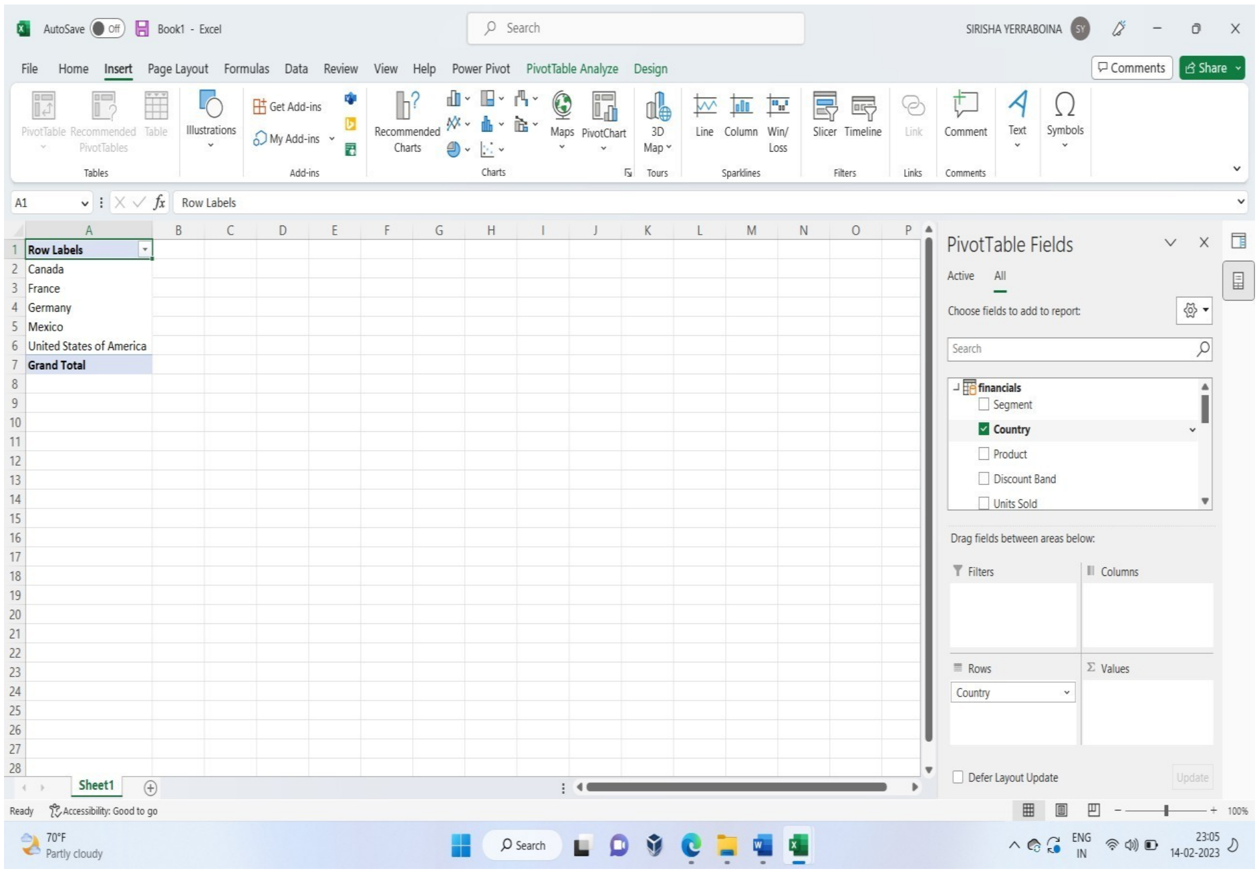


	_unit_id	_golden	_unit_state	_trusted_judgments	_last_judgment_at	gender	gender:confidence	profile_yn	profile_yn:confidence	created	description	fav_number	gender_gold	li
1	815723657	FALSE	finalized	3	10/26/15 22:44	brand	1	yes	1	04-01-20...		0		0
2	815723787	FALSE	finalized	3	10/26/15 22:42	brand	1	yes	1	08-01-20...		0		0
3	815724280	FALSE	finalized	3	10/26/15 23:38	brand	1	yes	1	8/14/14 ...		0		0
4	815724526	FALSE	finalized	3	10/27/15 1:51	brand	1	yes	1	8/14/14 ...		0		0
5	815724544	FALSE	finalized	3	10/26/15 21:56	brand	1	yes	1	8/22/14 ...		0		0
6	815724585	FALSE	finalized	3	10/26/15 23:58	brand	1	yes	1	6/21/14 ...		0		0
7	815724617	FALSE	finalized	3	10/27/15 1:22	brand	1	yes	1	05-12-20...		0		0
8	815724694	FALSE	finalized	3	10/27/15 1:22	brand	1	yes	1	05-03-20...		0		0
9	815724785	FALSE	finalized	3	10/27/15 1:01	brand	1	yes	1	09-08-20...		0		0
10	815724799	FALSE	finalized	3	10/26/15 23:28	brand	1	yes	1	08-12-20...		0		0
11	815724978	FALSE	finalized	3	10/26/15 23:56	brand	1	yes	1	8/17/14 ...		0		0
12	815725104	FALSE	finalized	3	10/26/15 22:37	brand	1	yes	1	6/23/14 ...		0		0
13	815725112	FALSE	finalized	3	10/27/15 1:16	brand	1	yes	1	4/27/14 ...		0		0
14	815725146	FALSE	finalized	3	10/26/15 23:43	brand	1	yes	1	3/26/14 ...		0		0
15	815725211	FALSE	finalized	3	10/27/15 1:50	brand	1	yes	1	4/16/15 ...		0		0
16	815725229	FALSE	finalized	3	10/26/15 22:34	brand	1	yes	1	8/28/14 ...		0		0
17	815725356	FALSE	finalized	3	10/27/15 0:02	brand	1	yes	1	4/26/14 ...		0		0
18	815725362	FALSE	finalized	3	10/26/15 23:12	brand	1	yes	1	6/24/14 ...		0		0
19	815725364	FALSE	finalized	3	10/27/15 1:51	brand	1	yes	1	08-03-20...		0		0
20	815725428	FALSE	finalized	3	10/26/15 23:24	brand	1	yes	1	4/22/14 ...		0		0
21	815725434	FALSE	finalized	3	10/26/15 23:56	brand	1	yes	1	08-12-20...		0		0

Click the diagram view and give the relation ships between the tables



Go to the Insert menu and click pivot table

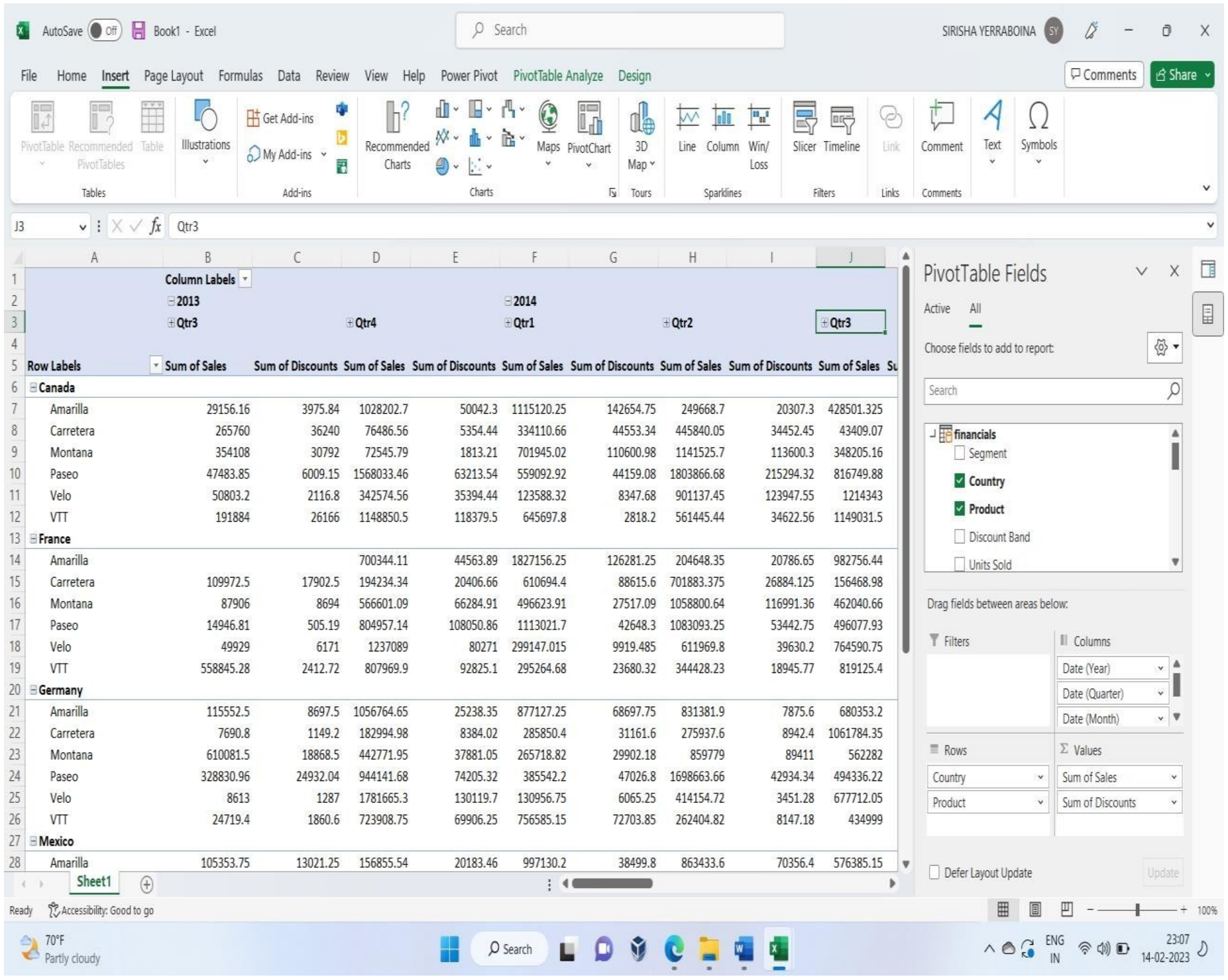

AutoSave Off Book1 - Excel
File Home Insert Page Layout Formulas Data Review View Help Power Pivot PivotTable Analyze Design
PivotTable Fields
Choose fields to add to report:
Drag fields between areas below:
Filters
Columns
Rows
Values
Defer Layout Update
Sheet1

Row Labels
Canada
France
Germany
Mexico
United States of America
Grand Total

financials
- [ ] Segment
- [✓] Country
- [ ] Product
- [ ] Discount Band
- [ ] Units Sold

Rows: Country

Select the columns and u can perform drill down and rollup operations using pivot table



	Column Labels								
	2013				2014				
	Qtr3		Qtr4		Qtr1		Qtr2		Qtr3
Row Labels	Sum of Sales	Sum of Discounts	Sum of Sales	Sum of Discounts	Sum of Sales	Sum of Discounts	Sum of Sales	Sum of Discounts	Sum of Sales
Canada									
Amarilla	29156.16	3975.84	1028202.7	50042.3	1115120.25	142654.75	249668.7	20307.3	428501.325
Carretera	265760	36240	76486.56	5354.44	334110.66	44553.34	445840.05	34452.45	43409.07
Montana	354108	30792	72545.79	1813.21	701945.02	110600.98	1141525.7	113600.3	348205.16
Paseo	47483.85	6009.15	1568033.46	63213.54	559092.92	44159.08	1803866.68	215294.32	816749.88
Velo	50803.2	2116.8	342574.56	35394.44	123588.32	8347.68	901137.45	123947.55	1214343
VTT	191884	26166	1148850.5	118379.5	645697.8	2818.2	561445.44	34622.56	1149031.5
France									
Amarilla			700344.11	44563.89	1827156.25	126281.25	204648.35	20786.65	982756.44
Carretera	109972.5	17902.5	194234.34	20406.66	610694.4	88615.6	701883.375	26884.125	156468.98
Montana	87906	8694	566601.09	66284.91	496623.91	27517.09	1058800.64	116991.36	462040.66
Paseo	14946.81	505.19	804957.14	108050.86	1113021.7	42648.3	1083093.25	53442.75	496077.93
Velo	49929	6171	1237089	80271	299147.015	9919.485	611969.8	39630.2	764590.75
VTT	558845.28	2412.72	807969.9	92825.1	295264.68	23680.32	344428.23	18945.77	819125.4
Germany									
Amarilla	115552.5	8697.5	1056764.65	25238.35	877127.25	68697.75	831381.9	7875.6	680353.2
Carretera	7690.8	1149.2	182994.98	8384.02	285850.4	31161.6	275937.6	8942.4	1061784.35
Montana	610081.5	18868.5	442771.95	37881.05	265718.82	29902.18	859779	89411	562282
Paseo	328830.96	24932.04	944141.68	74205.32	385542.2	47026.8	1698663.66	42934.34	494336.22
Velo	8613	1287	1781665.3	130119.7	130956.75	6065.25	414154.72	3451.28	677712.05
VTT	24719.4	1860.6	723908.75	69906.25	756585.15	72703.85	262404.82	8147.18	434999
Mexico									
Amarilla	105353.75	13021.25	156855.54	20183.46	997130.2	38499.8	863433.6	70356.4	576385.15

PivotTable Fields

financials: Segment, ✓ Country, ✓ Product, Discount Band, Units Sold

Columns: Date (Year), Date (Quarter), Date (Month)

Rows: Country, Product

Values: Sum of Sales, Sum of Discounts

We can load 10mllions rows of data also from multiple resources.

Experiment 6: Using Power Pivot perform the following on any data set
b) Big data Charting

Aim : To create variety of charts using Excel for the given data
Resources: Microsoft Excel
Theory:

When your data sets are big, you can use Excel Power Pivot that can handle hundreds of millions of rows of data. The data can be in external data sources and Excel Power Pivot builds a Data Model that works on a memory optimization mode. You can perform the calculations, analyze the data and arrive at a report to draw conclusions and decisions. The report can be either as a Power PivotTable or Power PivotChart or a combination of both.

You can utilize Power Pivot as an ad hoc reporting and analytics solution. Thus, it would be possible for a person with hands-on experience with Excel to perform the high-end data analysis and decision making in a matter of few minutes and are a great asset to be included in the dashboards.

Uses of Power Pivot

You can use Power Pivot for the following −

- To perform powerful data analysis and create sophisticated Data Models.
- To mash-up large volumes of data from several different sources quickly.
- To perform information analysis and share the insights interactively.
- To create Key Performance Indicators (KPIs).
- To create Power PivotTables.
- To create Power PivotCharts.

Differences between PivotTable and Power PivotTable

Power PivotTable resembles PivotTable in its layout, with the following differences −

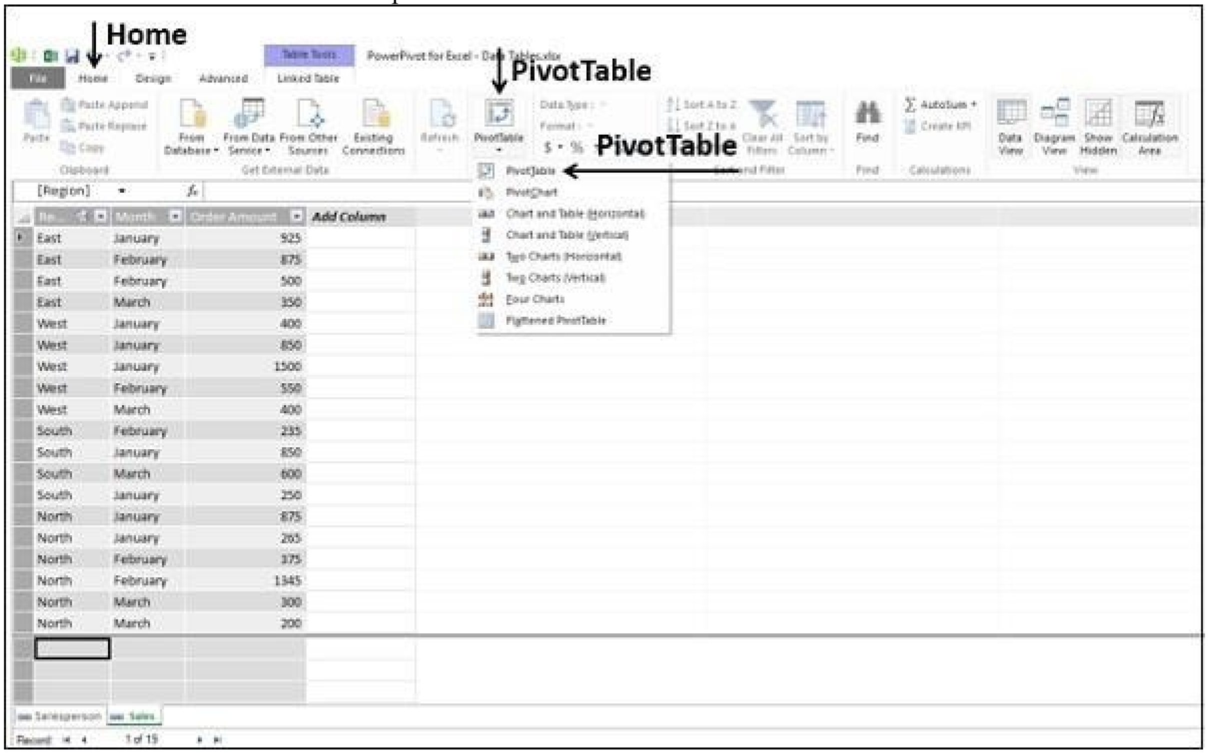
- PivotTable is based on Excel tables, whereas Power PivotTable is based on data tables that are part of Data Model.
- PivotTable is based on a single Excel table or data range, whereas
- Power PivotTable can be based on multiple data tables, provided they are added to Data Model.

PivotTable is created from Excel window, whereas Power PivotTable is created from PowerPivot window.

Creating a Power PivotTable

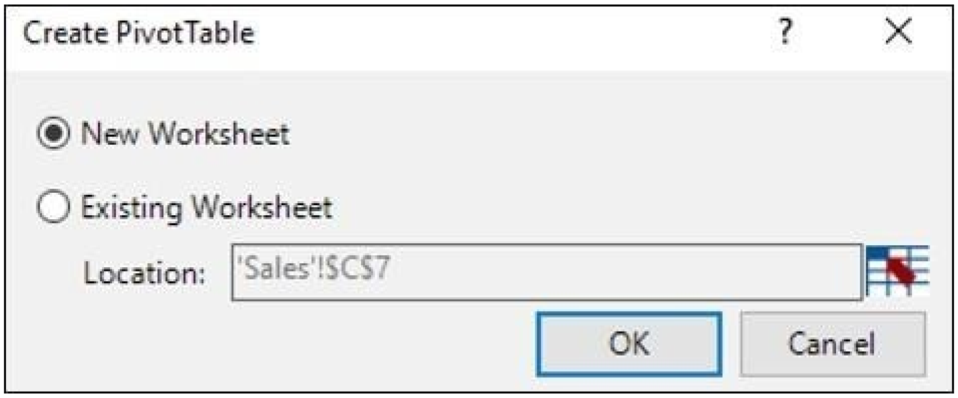
Suppose you have two data tables – Salesperson and Sales in the Data Model. To create a Power PivotTable from these two data tables, proceed as follows –

- Click on the Home tab on the Ribbon in PowerPivot window.
- Click on PivotTable on the Ribbon.
- Click on PivotTable in the dropdown list.



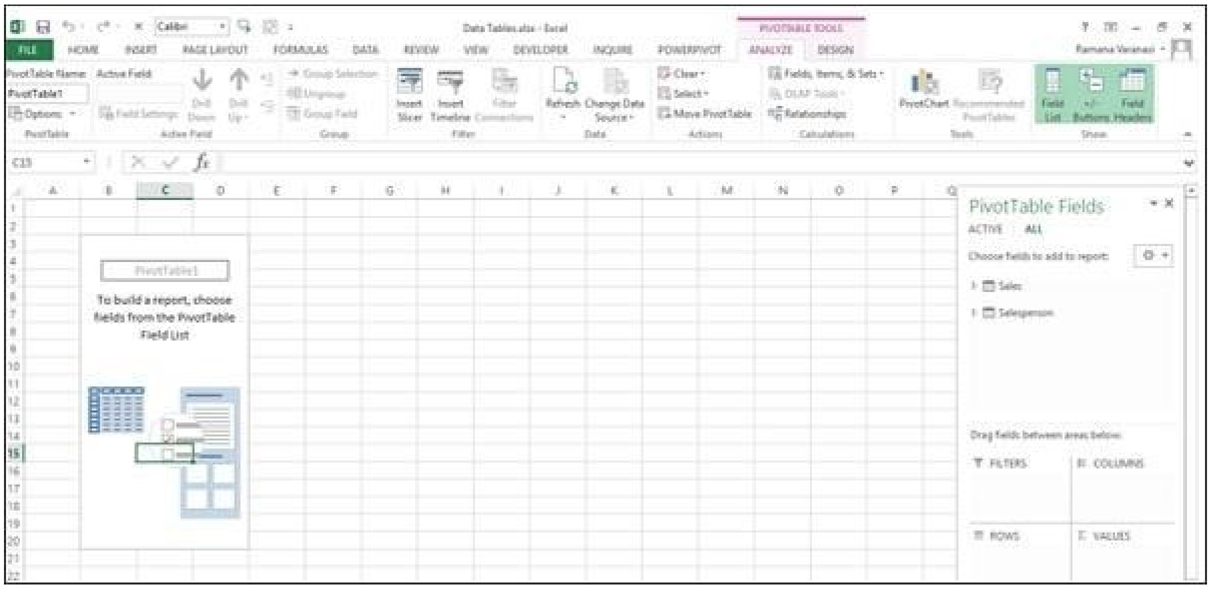
Re...	Month	Order Amount	Add Column
East	January	925	
East	February	875	
East	February	500	
East	March	350	
West	January	400	
West	January	850	
West	January	1500	
West	February	550	
West	March	400	
South	February	235	
South	January	850	
South	March	600	
South	January	250	
North	January	875	
North	January	265	
North	February	375	
North	February	1345	
North	March	300	
North	March	200	

Create PivotTable dialog box appears. Click on New Worksheet.



(●) New Worksheet
( ) Existing Worksheet
Location: 'Sales'!$C$7

Click the OK button. New worksheet gets created in Excel window and an empty Power PivotTable appears.


PivotTable Fields
ACTIVE
ALL
Choose fields to add to report:
Sales
Salesperson
Drag fields between areas below:
FILTERS
COLUMNS
ROWS
VALUES
PivotTable1
To build a report, choose fields from the PivotTable Field List

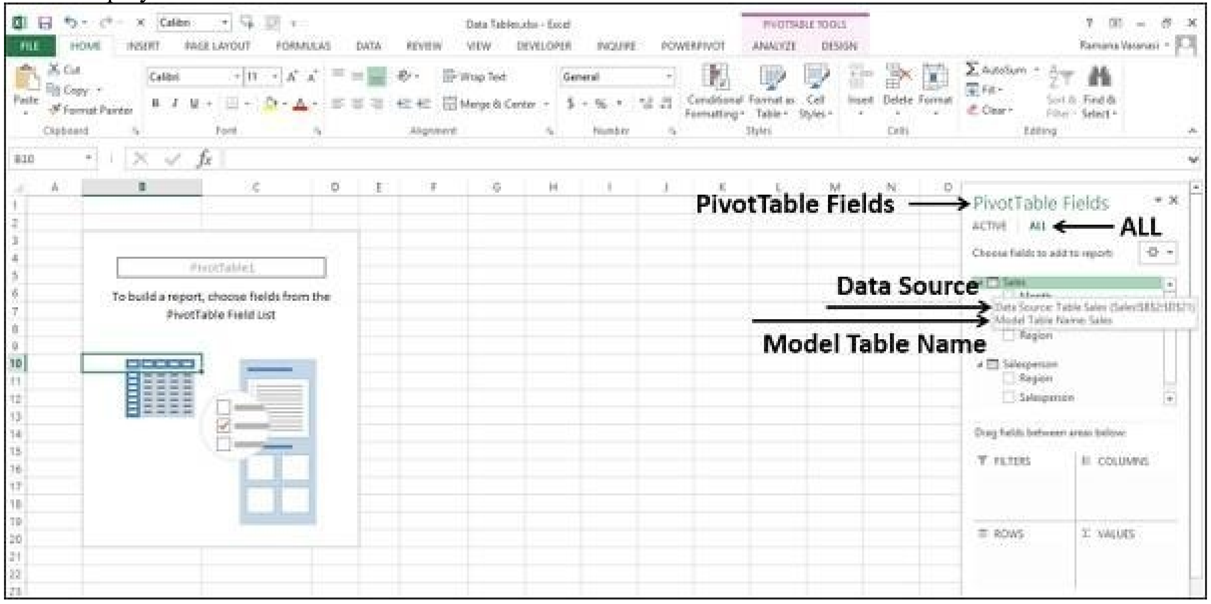
As you can observe, the layout of the Power PivotTable is similar to that of PivotTable.

The PivotTable Fields List appears on the right side of the worksheet. Here, you will 昀椀nd some differences from PivotTable. The Power PivotTable Fields list has two tabs − ACTIVE and ALL, that appear below the title and above the 昀椀elds list. ALL tab is highlighted. The ALL tab displays all the data tables in the Data Model and ACTIVE tab displays all the data tables that are chosen for the Power PivotTable at hand.

- Click the table names in the PivotTable Fields list under ALL.

The corresponding 昀椀elds with check boxes will appear.

- Each table name will have the symbol on the left side.
- If you place the cursor on this symbol, the Data Source and the Model Table Name of that data table will be displayed.

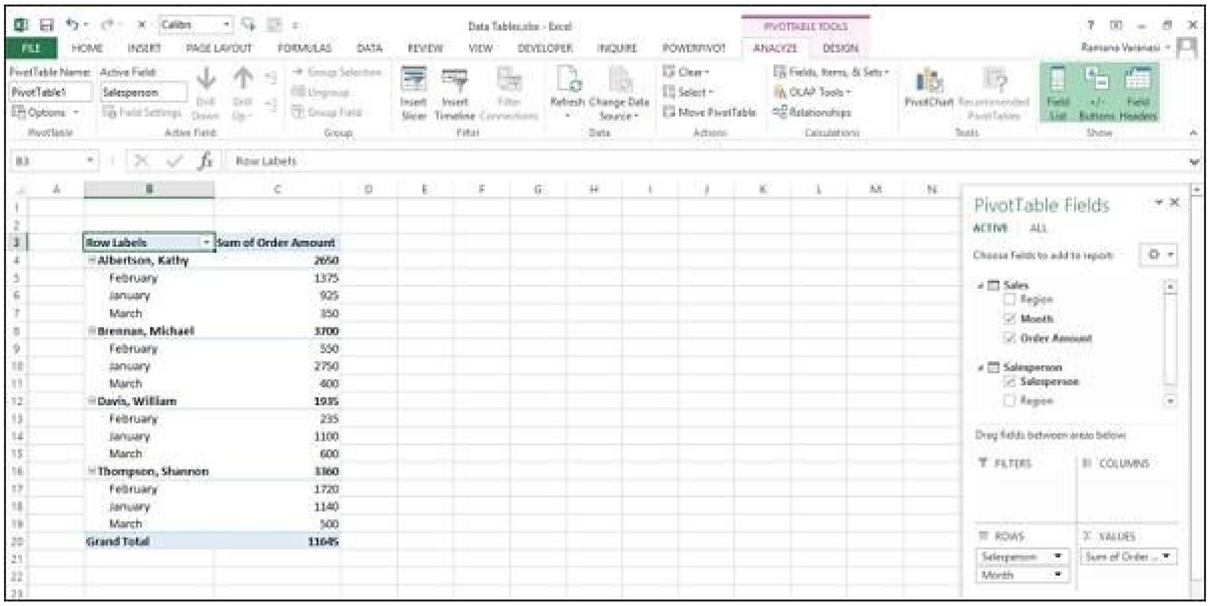


- Drag Salesperson from Salesperson table to ROWS area.
- Click on the ACTIVE tab.

The 昀椀eld Salesperson appears in the Power PivotTable and the table Salesperson appears under ACTIVE tab.

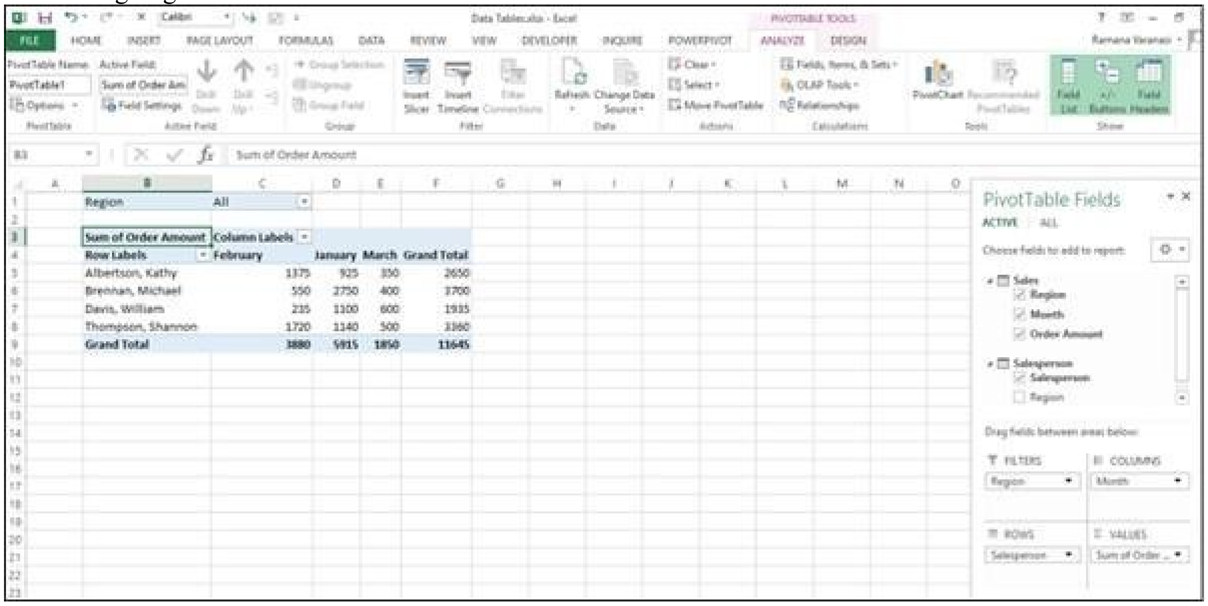
- Click on the ALL tab.
- Click on Month and Order Amount in the Sales table.
- Click on the ACTIVE tab.

Both the tables – Sales and Salesperson appear under the ACTIVE tab.



Row Labels	Sum of Order Amount
Albertson, Kathy	2650
February	1375
January	925
March	350
Brennan, Michael	3700
February	550
January	2750
March	400
Davis, William	1935
February	235
January	1100
March	600
Thompson, Shannon	3360
February	1720
January	1140
March	500
Grand Total	11645

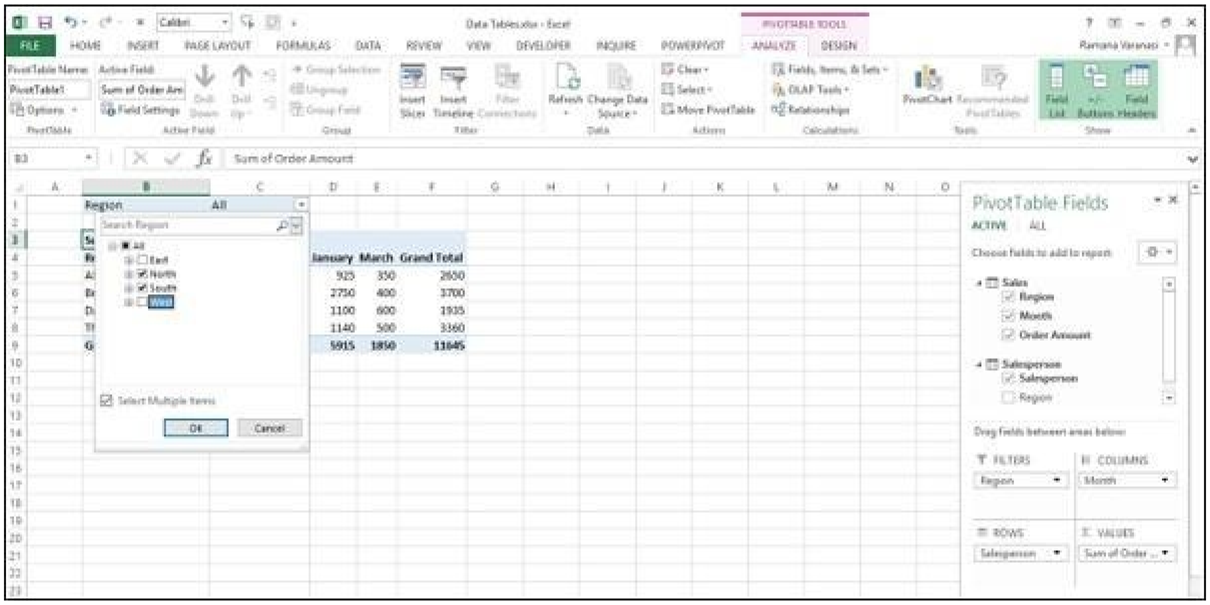
- Drag Month to COLUMNS area.
- Drag Region to FILTERS area.



Region: All

Sum of Order Amount	Column Labels			
Row Labels	February	January	March	Grand Total
Albertson, Kathy	1375	925	350	2650
Brennan, Michael	550	2750	400	3700
Davis, William	235	1100	600	1935
Thompson, Shannon	1720	1140	500	3360
Grand Total	3880	5915	1850	11645

- Click on arrow next to ALL in the Region 昀椀lter box.
- Click on Select Multiple Items.
- Click on North and South.



Region: All

- (All)
  - East
  - ☑ North
  - ☑ South
  - West

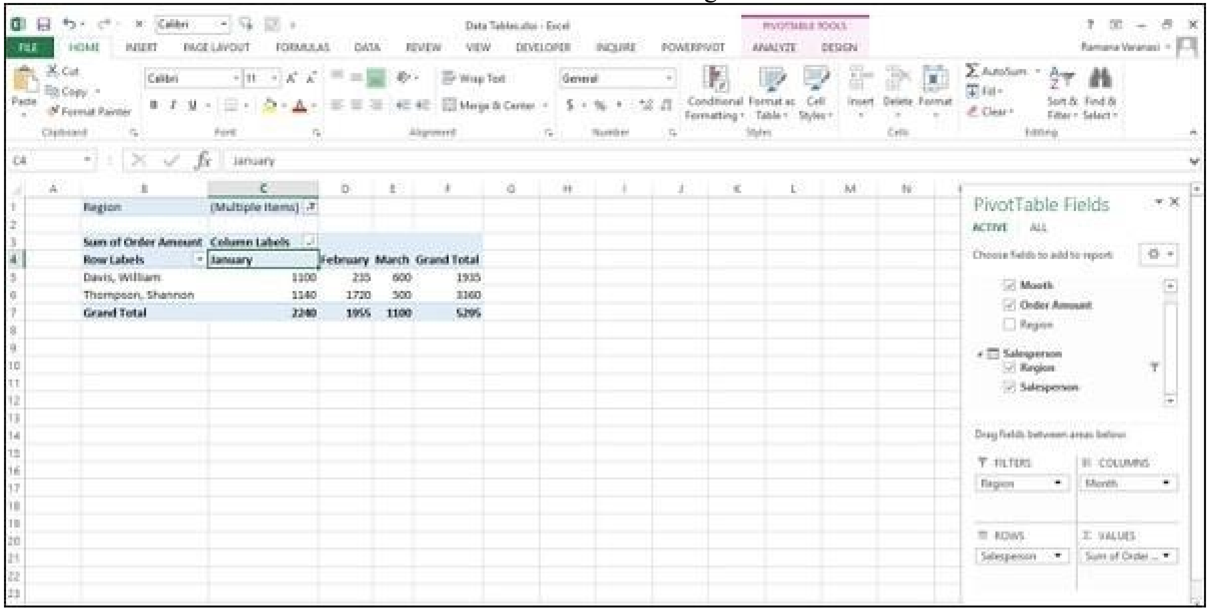
☑ Select Multiple Items

OK Cancel

January	March	Grand Total
925	350	2650
2750	400	3700
1100	600	1935
1140	500	3360
5915	1850	11645

PivotTable Fields — FILTERS: Region; COLUMNS: Month; ROWS: Salesperson; VALUES: Sum of Order ...

- Click the OK button. Sort the column labels in the ascending order.



Region: (Multiple Items)

Sum of Order Amount	Column Labels			
Row Labels	January	February	March	Grand Total
Davis, William	1100	235	600	1935
Thompson, Shannon	1140	1720	500	3360
Grand Total	2240	1955	1100	5295

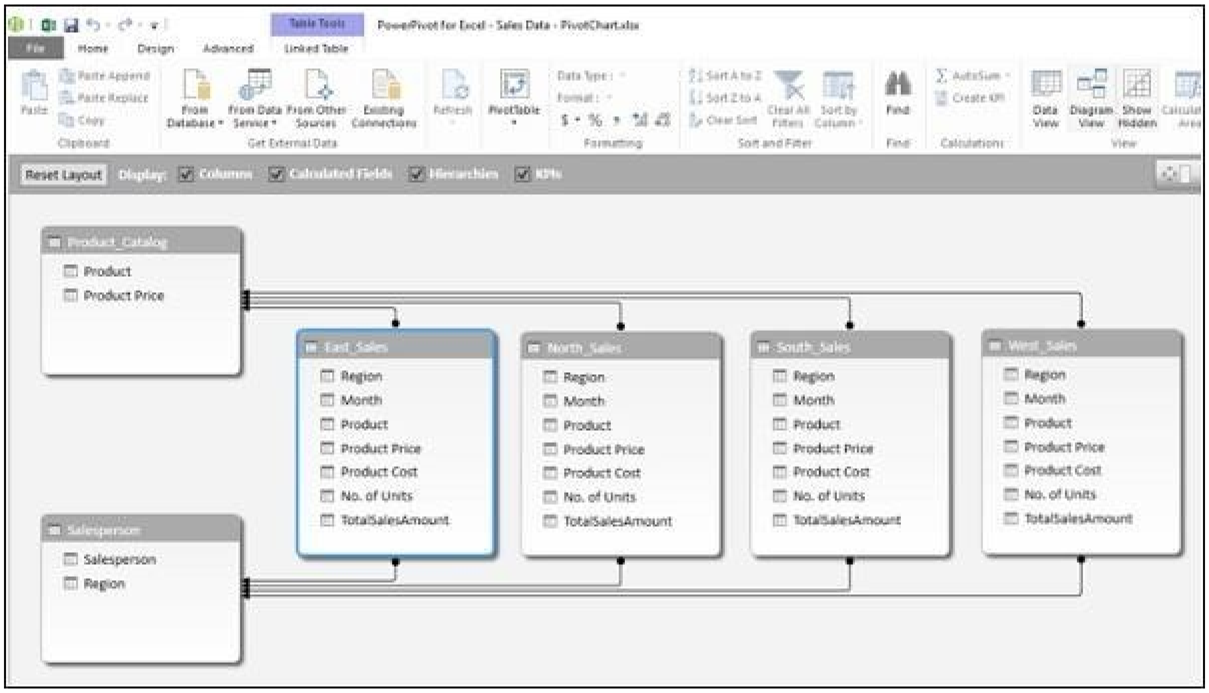
PivotTable Fields — FILTERS: Region; COLUMNS: Month; ROWS: Salesperson; VALUES: Sum of Order ...

Power PivotTable can be modi昀椀ed dynamically to explore and report data.

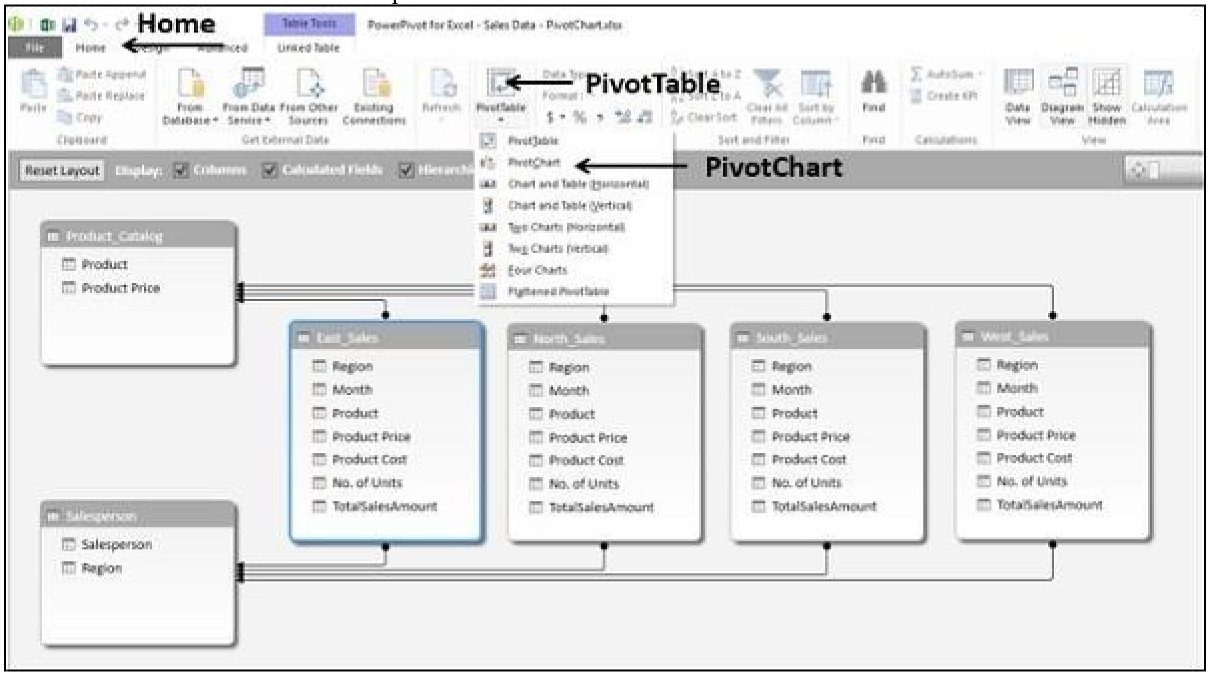
Creating a Power PivotChart

A Power PivotChart is a PivotChart that is based on Data Model and created from the Power Pivot window. Though it has some features similar to Excel PivotChart, there are other features that make it more powerful.

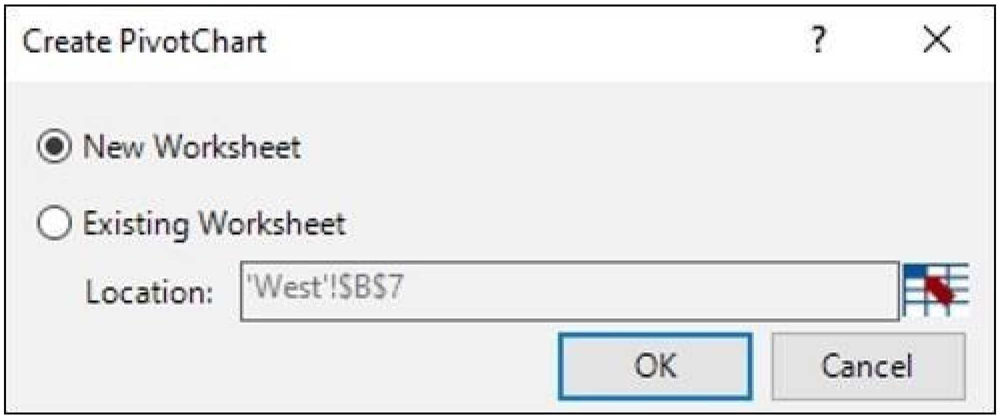
Suppose you want to create a Power PivotChart based on the following Data Model.



- Click on the Home tab on the Ribbon in the Power Pivot window.
- Click on PivotTable.
- Click on PivotChart in the dropdown list.



Create PivotChart dialog box appears. Click New Worksheet.

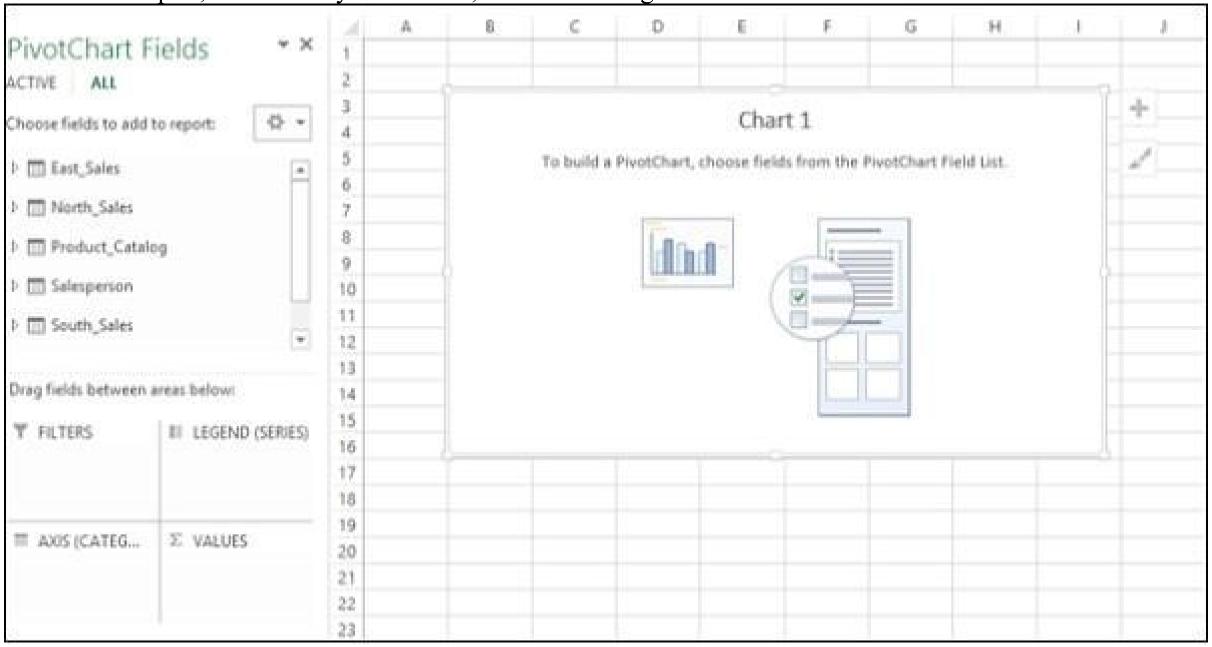

Create PivotChart

[● New Worksheet]

○ Existing Worksheet

Location: 'West'!$B$7

- Click the OK button. An empty PivotChart gets created on a new worksheet in the Excel window. In this chapter, when we say PivotChart, we are referring to Power PivotChart.


PivotChart Fields

ACTIVE ALL

Choose fields to add to report:

- East_Sales
- North_Sales
- Product_Catalog
- Salesperson
- South_Sales

Drag fields between areas below:

FILTERS | LEGEND (SERIES) | AXIS (CATEG... | VALUES

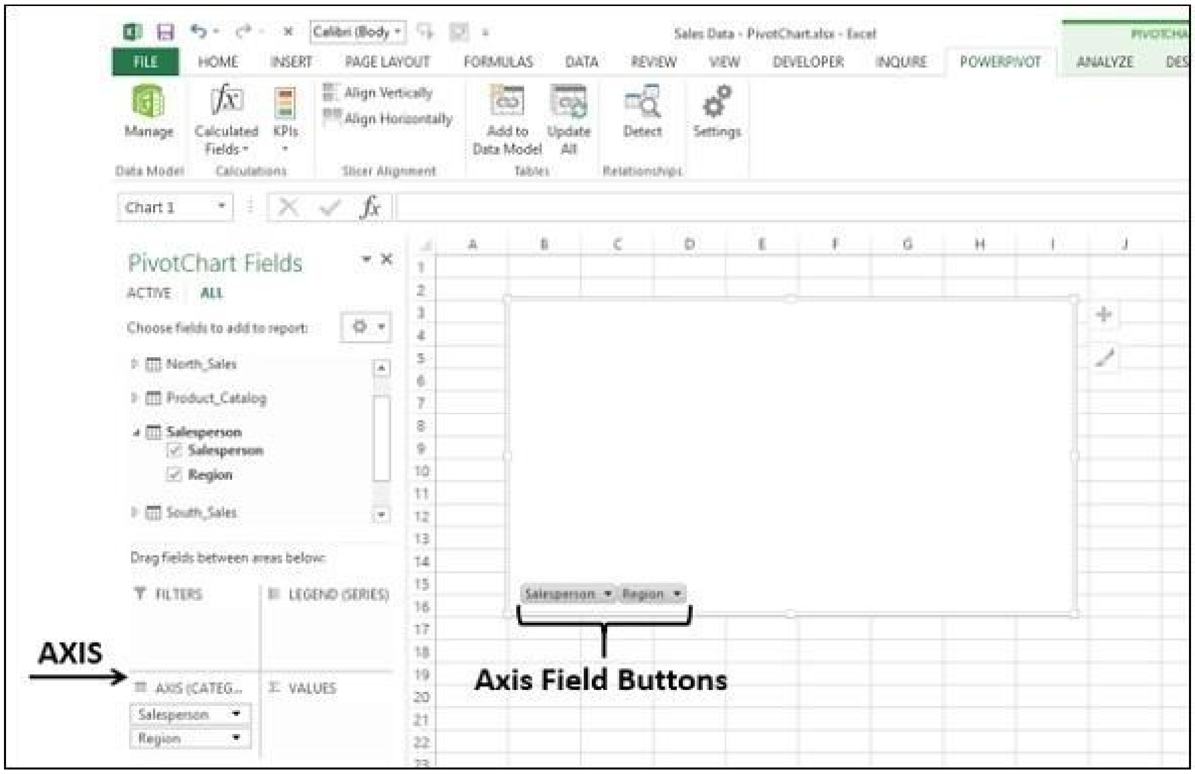
Chart 1

To build a PivotChart, choose fields from the PivotChart Field List.

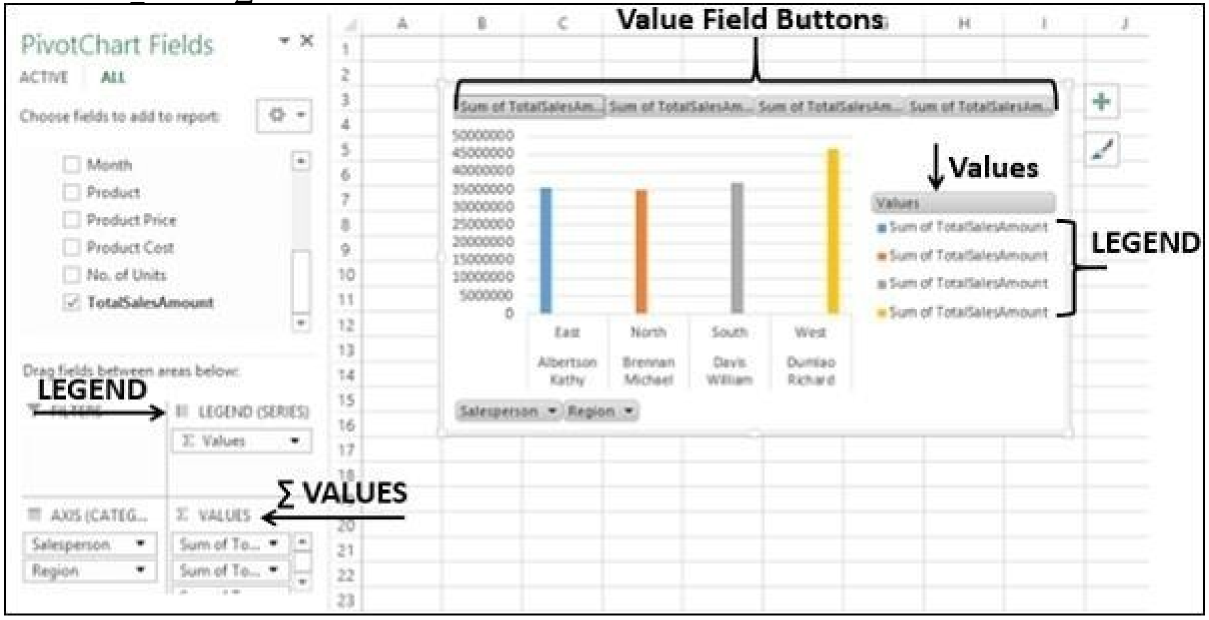
As you can observe, all the tables in the data model are displayed in the PivotChart Fields list.

- Click on the Salesperson table in the PivotChart Fields list.
- Drag the 昀椀elds – Salesperson and Region to AXIS area.

Two 昀椀eld buttons for the two selected 昀椀elds appear on the PivotChart. These are the Axis昀椀eld buttons. The use of 昀椀eld buttons is to 昀椀lter data that is displayed on the PivotChart.



- Drag TotalSalesAmount from each of the 4 tables – East_Sales, North_Sales, South_Sales and West_Sales to ∑ VALUES area.

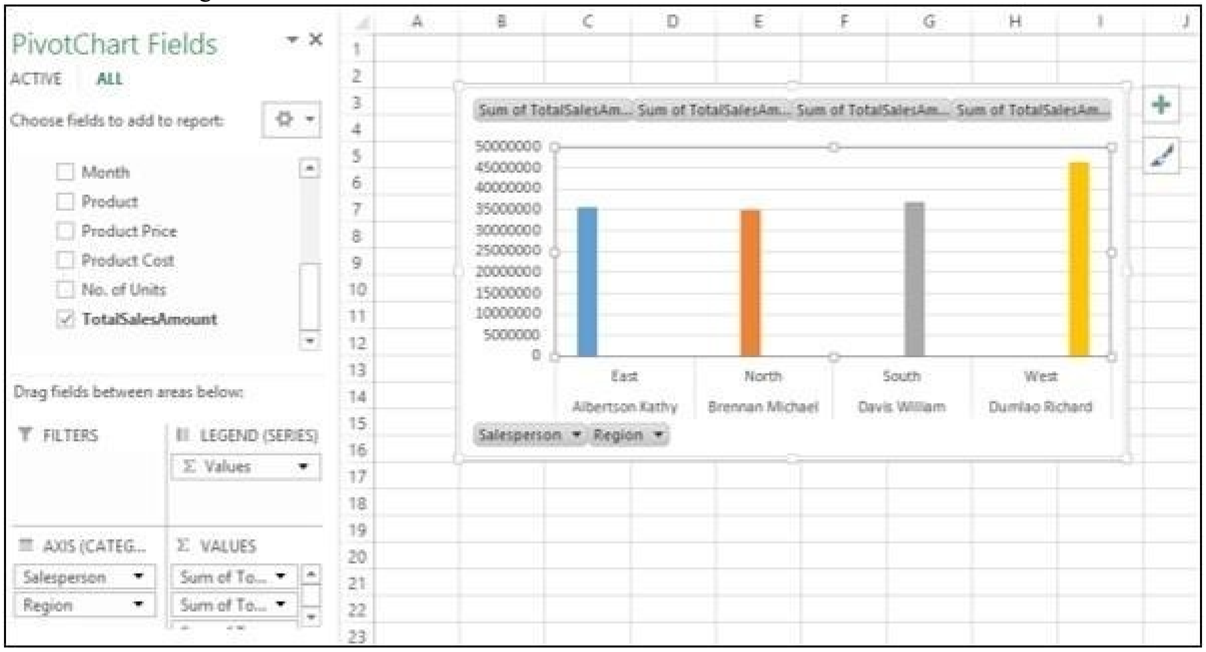


As you can observe, the following appear on the worksheet –

- In the PivotChart, column chart is displayed by default.
- In the LEGEND area, ∑ VALUES gets added.
- The Values appear in the Legend in the PivotChart, with title Values.
- The Value Field Buttons appear on the PivotChart.

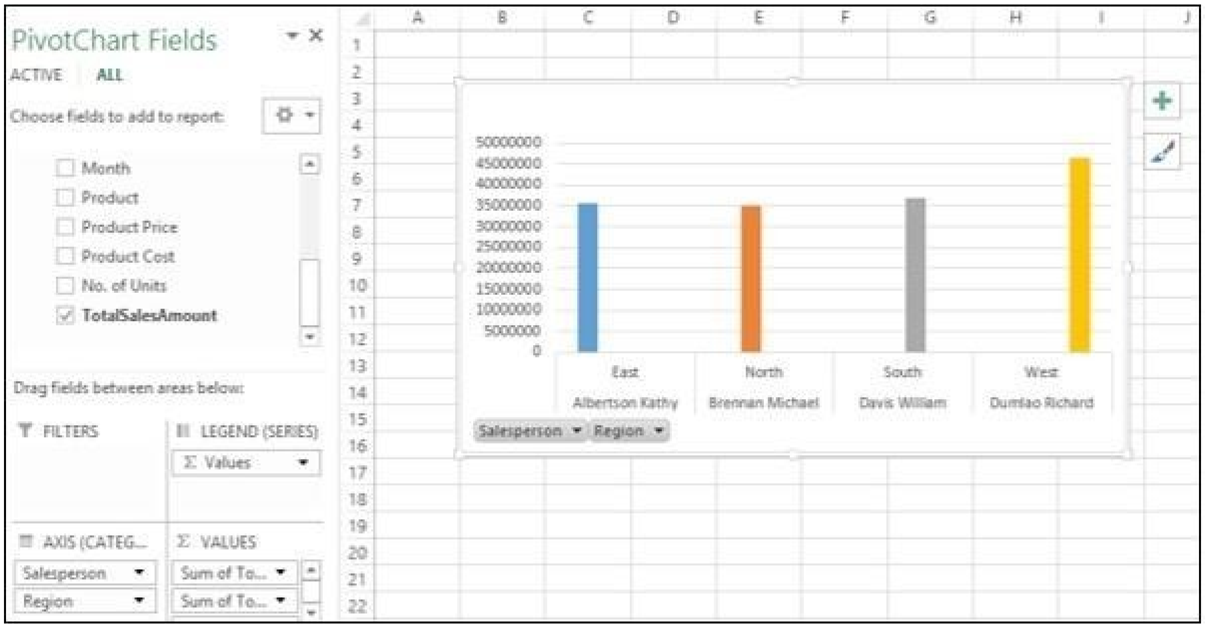
You can remove the legend and the value 昀椀eld buttons for a tidier look of the PivotChart.

- Click on the button at the top right corner of the PivotChart.
- Deselect Legend in the Chart Elements.



- Right click on the value 昀椀eld buttons.
- Click on Hide Value Field Buttons on Chart in the dropdown

list. The value 昀椀eld buttons on the chart will be



hidden.

Note that display of Field Buttons and/or Legend depends on the context of the PivotChart. You need to decide what is required to be displayed.

As in the case of Power PivotTable, Power PivotChart Fields list also contains two tabs – ACTIVE and ALL. Further, there are 4 areas –

- AXIS (Categories)

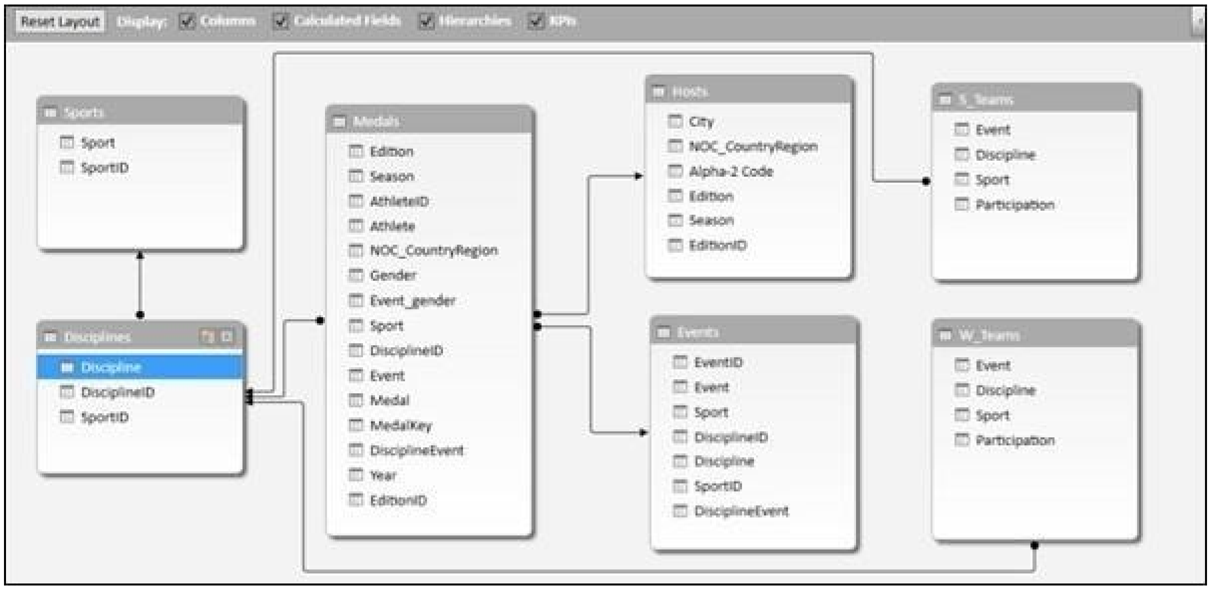
- LEGEND (Series)
- ∑ VALUES
- FILTERS

As you can observe, Legend gets populated with ∑ Values. Further, Field Buttons get added to the PivotChart for the ease of 昀椀ltering the data that is being displayed. You can click on the arrow on a Field Button and select/deselect values to be displayed in the Power PivotChart.

Table and Chart Combinations

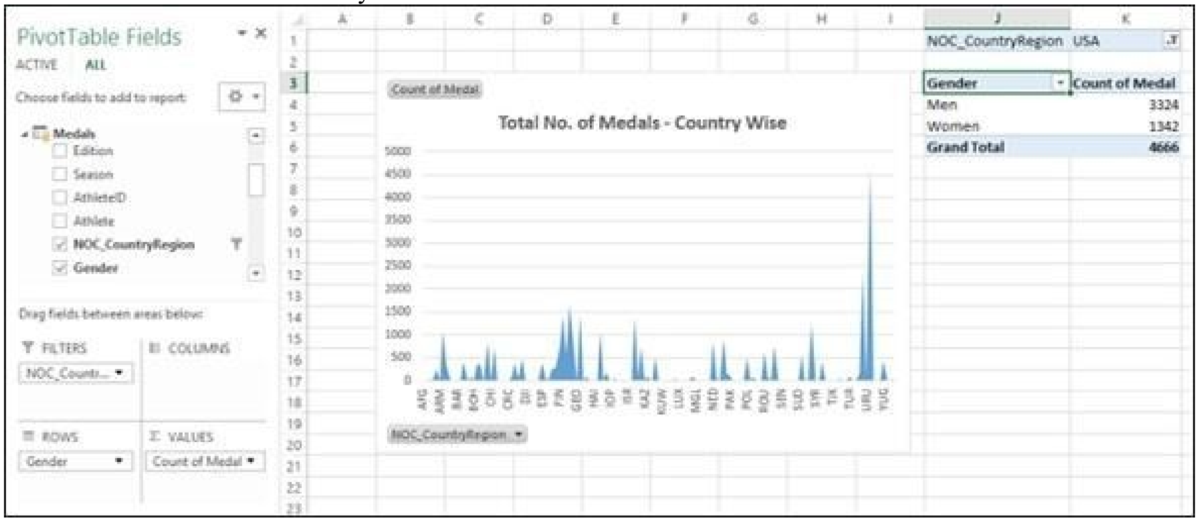
Power Pivot provides you with different combinations of Power PivotTable and Power PivotChart for data exploration, visualization and reporting.

Consider the following Data Model in Power Pivot that we will use for illustrations –



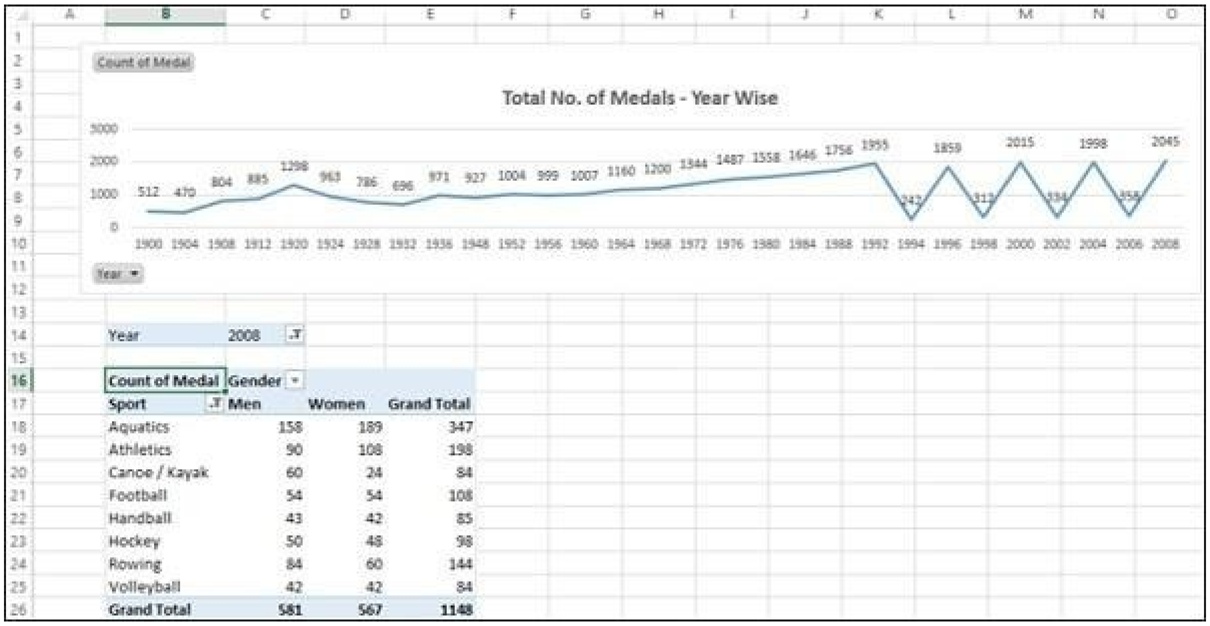
You can have the following Table and Chart Combinations in Power Pivot.

- Chart and Table (Horizontal) - you can create a Power PivotChart and a Power PivotTable, one next to another horizontally in the same worksheet.



NOC_CountryRegion	USA
Gender	Count of Medal
Men	3324
Women	1342
Grand Total	4666

Chart and Table (Vertical) - you can create a Power PivotChart and a Power PivotTable, one below another vertically in the same worksheet.



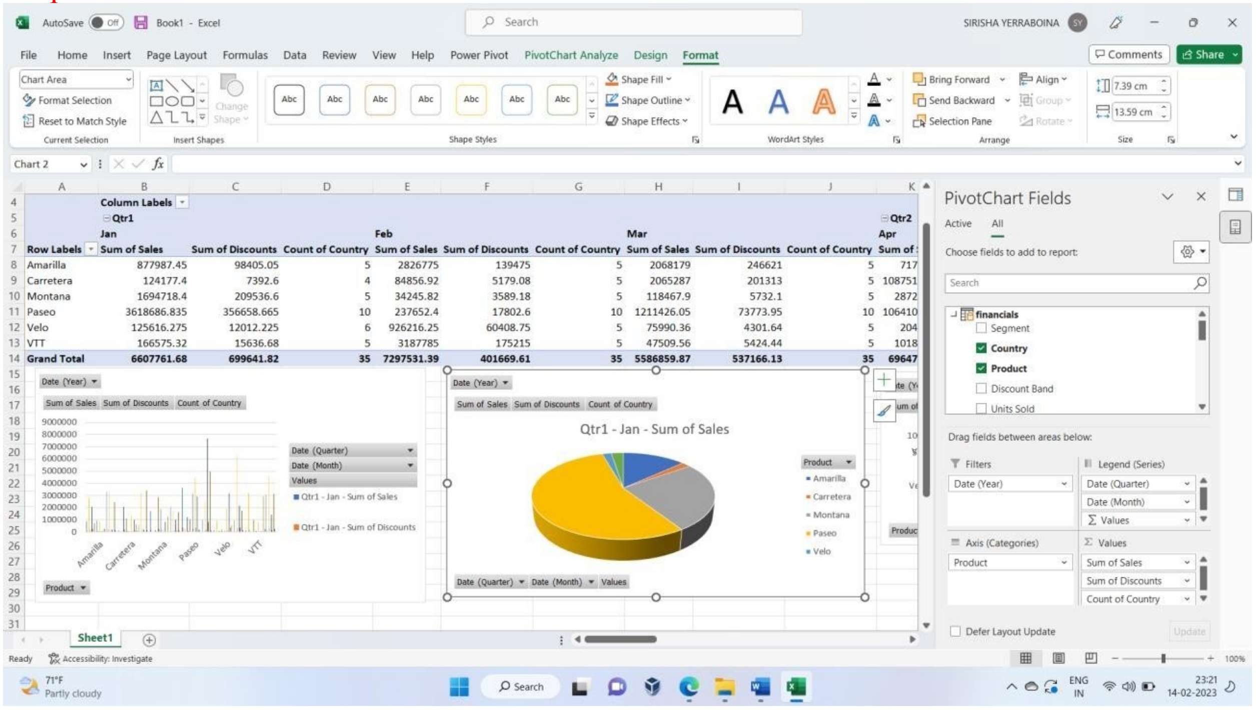
Year 2008

Count of Medal	Gender		
Sport	Men	Women	Grand Total
Aquatics	158	189	347
Athletics	90	108	198
Canoe / Kayak	60	24	84
Football	54	54	108
Handball	43	42	85
Hockey	50	48	98
Rowing	84	60	144
Volleyball	42	42	84
Grand Total	581	567	1148

These combinations and some more are available in the dropdown list that appears when you click on PivotTable on the Ribbon in the Power Pivot window.

Click on the pivot chart and can develop multiple variety of charts

Output:



	Column Labels									
	Qtr1									Qtr2
	Jan			Feb			Mar			Apr
Row Labels	Sum of Sales	Sum of Discounts	Count of Country	Sum of Sales	Sum of Discounts	Count of Country	Sum of Sales	Sum of Discounts	Count of Country	Sum of
Amarilla	877987.45	98405.05	5	2826775	139475	5	2068179	246621	5	717
Carretera	124177.4	7392.6	4	84856.92	5179.08	5	2065287	201313	5	108751
Montana	1694718.4	209536.6	5	34245.82	3589.18	5	118467.9	5732.1	5	2872
Paseo	3618686.835	356658.665	10	237652.4	17802.6	10	1211426.05	73773.95	10	106410
Velo	125616.275	12012.225	6	926216.25	60408.75	5	75990.36	4301.64	5	204
VTT	166575.32	15636.68	5	3187785	175215	5	47509.56	5424.44	5	1018
Grand Total	6607761.68	699641.82	35	7297531.39	401669.61	35	5586859.87	537166.13	35	69647

Experiment 7: Using R project to carry out statistical analysis of big data

Aim: To perform the statistical analysis of big data using R

Theory: Statistics is the science of analyzing, reviewing and conclude data.

Some basic statistical numbers include:

- Mean, median and mode
- Minimum and maximum value
- Percentiles
- Variance and Standard Devation
- Covariance and Correlation
- Probability distributions

The R language was developed by two statisticians. It has many built-in functionalities, in addition to libraries for the exact purpose of statistical analysis.

Procedure:

Installation of R and Rstudio

Step 1:

```
sudo apt-get update
sudo apt-get install r-base
```

Step 2:

Installation of R studio

https://posit.co/download/rstudio-desktop/#download

step 1:download R studio for ubuntu

step 2 :wget -c https://download1.rstudio.org/desktop/jammy/amd64/rstudio-2022.07.2-576-amd64.deb

step 2:sudo dpkg -i rstudio-2022.07.2-576-amd64.deb

step 3 :sudo apt install -f step
4:rstudio launch R studio

procedure:

-->install.packages("gapminder")
-->library(gapminder)

-->data(gapminder) output:

A tibble: 1,704 × 6

```
      country           continent        year  lifeExp              pop  gdpPercap

      <fct>             <fct>          <int>      <dbl>         <int>          <dbl>

  1  Afghanistan  Asia                  1952         28.8       8425333              779.

  2  Afghanistan  Asia                  1957         30.3       9240934              821.

  3  Afghanistan  Asia                  1962         32.0   10267083              853.

  4  Afghanistan  Asia                  1967         34.0   11537966              836.

  5  Afghanistan  Asia                  1972         36.1   13079460              740.

  6  Afghanistan  Asia                  1977         38.4   14880372              786.

  7  Afghanistan  Asia                  1982         39.9   12881816              978.

  8  Afghanistan  Asia                  1987         40.8   13867957              852.

  9  Afghanistan  Asia                  1992         41.7   16317921              649.

10  Afghanistan  Asia                  1997         41.8   22227415              635.

# … with 1,694 more rows
```

-->summary(gapminder)
summary(gapminder) output:

```
               country                      continent                    year
  Afghanistan:    12   Africa  :624Min.  1952 Albania        :      12
                  Americas:300    1st Qu.:1966 Algeria       :      12
                  Asia          :396Median  :1980
  Angola           :      12       Europe       :360          Mean           1980

  Argentina         :      12       Oceania  :  24             3rd Qu.:1993
```

```
 Australia          :    12                               Max.      :2007

 (Other)                 1632

       lifeExp                     pop                        gdpPercap

 Min.      :23.60      Min.      :6.001e+04       Min.      :     241.2

 1st Qu.:48.20         1st Qu.:2.794e+06          1st Qu.:        1202.1

 Median  :60.71        Median  :7.024e+06         Median :       3531.8

 Mean      :59.47      Mean      :2.960e+07       Mean      :     7215.3

 3rd Qu.:70.85         3rd Qu.:1.959e+07          3rd Qu.:        9325.5

 Max.      :82.60      Max.      :1.319e+09       Max.      :113523.1
```

-->x<-mean(gapminder$gdpPercap)

Type X to get mean value of gapminder

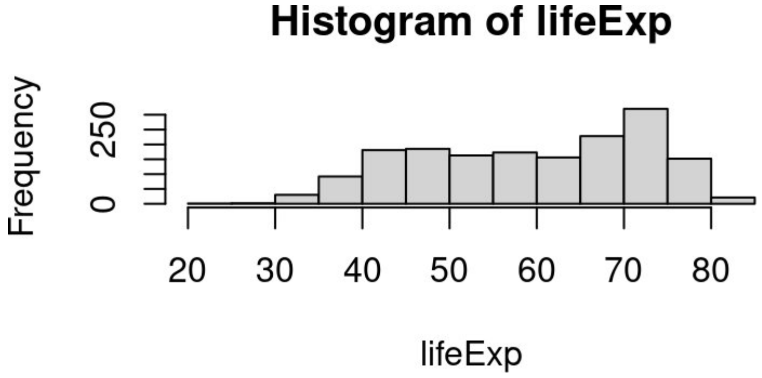
-->x

output:[1] 7215.327

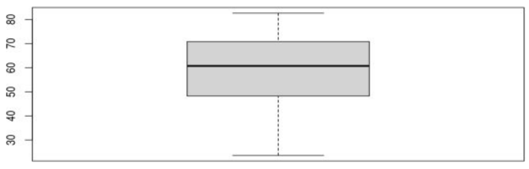
-->attach(gapminder)

**-->median(pop) output:[1]
7023596**

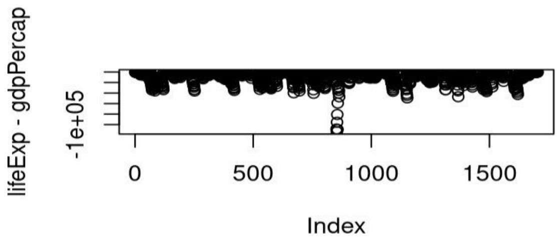
-->hist(lifeExp)



-->boxplot(lifeExp)
will plot the below images



-->plot(lifeExp - gdpPercap)



-->install.packages("dplyr")

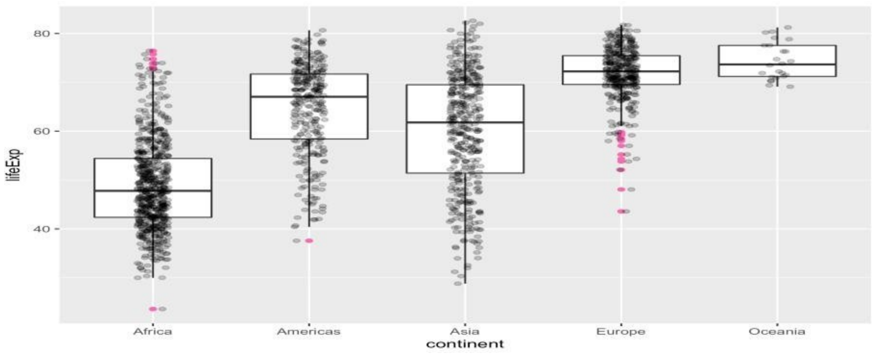
```
-->gapminder %>%
+     filter(year == 2007) %>%
+     group_by(continent) %>%
+     summarise(lifeExp = median(lifeExp))
```

output:

```
# A tibble: 5 × 2
  continent lifeExp
  <fct>       <dbl>
1 Africa       52.9
2 Americas     72.9
3 Asia         72.4
4 Europe       78.6
5 Oceania      80.7
```

```
-->install.packages("ggplot2")
--> library("ggplot2")
-->ggplot(gapminder, aes(x = continent, y = lifeExp))
  + geom_boxplot(outlier.colour = "hotpink") +
  geom_jitter(position = position_jitter(width = 0.1, height = 0), alpha = 1/4)
```

output:



-->head(country_colors, 4)
output:

```
        Nigeria           Egypt        Ethiopia
      "#7F3B08"     "#833D07"       "#873F07"
Congo, Dem. Rep.
      "#8B4107"
```

-->head(continent_colors)

mtcars

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8
Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

```
> Data_Cars <- mtcars
> dim(Data_Cars)
[1] 32 11
> names(Data_Cars)
  [1]   "mpg"  "cyl"  "disp" "hp"               "drat" "wt"          "qsec" "vs"           "am"       "gear" "car b"
> Data_Cars <- mtcars
> Data_Cars$cyl
         [1] 6 6 4 6 8 6 8 4 4 6 6  8  8  8  8  8   8 4 4   4  4  8  8  8  8   4 4  4  8  6   8 4
> Data_Cars <- mtcars
> sort(Data_Cars$cyl)
         [1] 4 4 4 4 4 4 4 4 4 4 4  6  6  6  6  6   6 6 8   8  8  8  8  8  8   8 8  8  8  8   8 8
> Data_Cars <- mtcars
>
> summary(Data_Cars)

       mpg                 cyl                 disp                hp                   drat
   Min.     :10.40     Min.     :4.000     Min.     : 71.1     Min.     : 52.0     Min.     :2.760

  1st Qu.:15.43       1st Qu.:4.000      1st Qu.:120.8      1st Qu.: 96.5      1st Qu.:3.080
  Median :19.20       Median :6.000      Median :196.3      Median :123.0      Median :3.695
  Mean     :20.09     Mean     :6.188    Mean     :230.7    Mean     :146.7    Mean     :3.597
  3rd Qu.:22.80       3rd Qu.:8.000      3rd Qu.:326.0      3rd Qu.:180.0      3rd Qu.:3.920
  Max.     :33.90     Max.     :8.000    Max.     :472.0    Max.     :335.0    Max.     :4.930
           wt                 qsec                vs                  a                    gear
                                                                      m

  Min.     :1.513     Min.     :14.50    Min.     :0.0000   Min.     :0.0000   Min.     :3.00
0                     1st Qu.:16.89      1st Qu.:0.0000     1st Qu.:0.0000     1st Qu.:3.00
  1st Qu.:2.581       Median :17.71      Median :0.0000     Median :0.0000     Median :4.00
0
  Median :3.325
```

```
0
  Mean      :3.217        Mean      :17.85        Mean      :0.4375        Mean      :0.4062        Mean      :3.68
8
  3rd Qu.:3.610          3rd Qu.:18.90          3rd Qu.:1.0000          3rd Qu.:1.0000          3rd Qu.:4.00
0
  Max.      :5.424        Max.      :22.90        Max.      :1.0000        Max.      :1.0000        Max.      :5.00
0
          carb
  Min.   :1.000    1st
  Qu.:2.000
  Median :2.000
  Mean   :2.812   3rd
  Qu.:4.000
  Max.   :8.000
> Data_Cars <- mtcars
>
> max(Data_Cars$hp)
[1] 335
> min(Data_Cars$hp)
[1] 52
> Data_Cars <- mtcars
>
> which.max(Data_Cars$hp)
[1] 31
> which.min(Data_Cars$hp)
[1] 19

> Data_Cars <- mtcars
> rownames(Data_Cars)[which.max(Data_Cars$hp)]
[1] "Maserati Bora"
> rownames(Data_Cars)[which.min(Data_Cars$hp)]
[1] "Honda Civic"
> median(Data_Cars$wt)
[1] 3.325
> names(sort(-table(Data_Cars$wt)))[1] [1]
"3.44"
> Data_Cars <- mtcars
> an(Data_Cars$wt)
[1] 3.21725

Data_Cars <- mtcars
median(Data_Cars$wt)

    [1]  3.325
Data_Cars  <-  mtcars

names(sort(-table(Data_Cars$wt)))[1]

Data_Cars  <-  mtcars

#   c()   specifies   which   percentile   you   want
quantile(Data_Cars$wt, c(0.75)) 75%
3.61
>Data_Cars <- mtcars
> quantile(Data_Cars$wt)
        0%        25%        50%        75%       100%
               1.51300 2.58125 3.32500 3.61000 5.42400
```

Regression analysis using R

Regression analysis is a very widely used statistical tool to establish a relationship model between two variables. One of these variable is called predictor variable whose value is gathered through experiments. The other variable is called response variable whose value is derived from the predictor variable.

In Linear Regression these two variables are related through an equation, where exponent (power) of both these variables is 1. Mathematically a linear relationship represents a straight line when plotted as a graph. A non-linear relationship where the exponent of any variable is not equal to 1 creates a curve.

The general mathematical equation for a linear regression is −

```
y = ax + b
```

Following is the description of the parameters used −

- **y** is the response variable.
- **x** is the predictor variable.
- **a** and **b** are constants which are called the coefficients.

Steps to Establish a Regression

A simple example of regression is predicting weight of a person when his height is known. To do this we need to have the relationship between height and weight of a person.

The steps to create the relationship is −

- Carry out the experiment of gathering a sample of observed values of height and corresponding weight.
- Create a relationship model using the **lm()** functions in R.
- Find the coefficients from the model created and create the mathematical equation using these

- Get a summary of the relationship model to know the average error in prediction. Also called **residuals**.
- To predict the weight of new persons, use the **predict()** function in R.

Input Data

Below is the sample data representing the observations −

```
# Values of height
151, 174, 138, 186, 128, 136, 179, 163, 152, 131

# Values of weight.
63, 81, 56, 91, 47, 57, 76, 72, 62, 48
```

lm() Function

This function creates the relationship model between the predictor and the response vari- able.

Syntax

The basic syntax for **lm()** function in linear regression is −

```
lm(formula,data)
```

Following is the description of the parameters used −

- **formula** is a symbol presenting the relation between x and y.
- **data** is the vector on which the formula will be applied.

Create Relationship Model & get the Coefficient

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.
relation <- lm(y~x)

print(relation)
```

Result:

```
Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)          x
  -38.4551      0.6746
```

To get the summary of the relation ships

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.
relation <- lm(y~x)

print(summary(relation))
```

Result:

```
Call:
lm(formula = y ~ x)

Residuals:
   Min      1Q    Median     3Q     Max
-6.3002   -1.6629  0.0412    1.8944  3.9775

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -38.45509    8.04901  -4.778  0.00139 **
x             0.67461    0.05191  12.997 1.16e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.253 on 8 degrees of freedom
Multiple R-squared:  0.9548,    Adjusted R-squared: 0.9491
F-statistic: 168.9 on 1 and 8 DF,  p-value: 1.164e-06
```

predict() Function

Syntax

The basic syntax for predict() in linear regression is −

```
predict(object, newdata)
```

Following is the description of the parameters used −

- **object** is the formula which is already created using the lm() function.
- **newdata** is the vector containing the new value for predictor variable.

Predict the weight of new persons

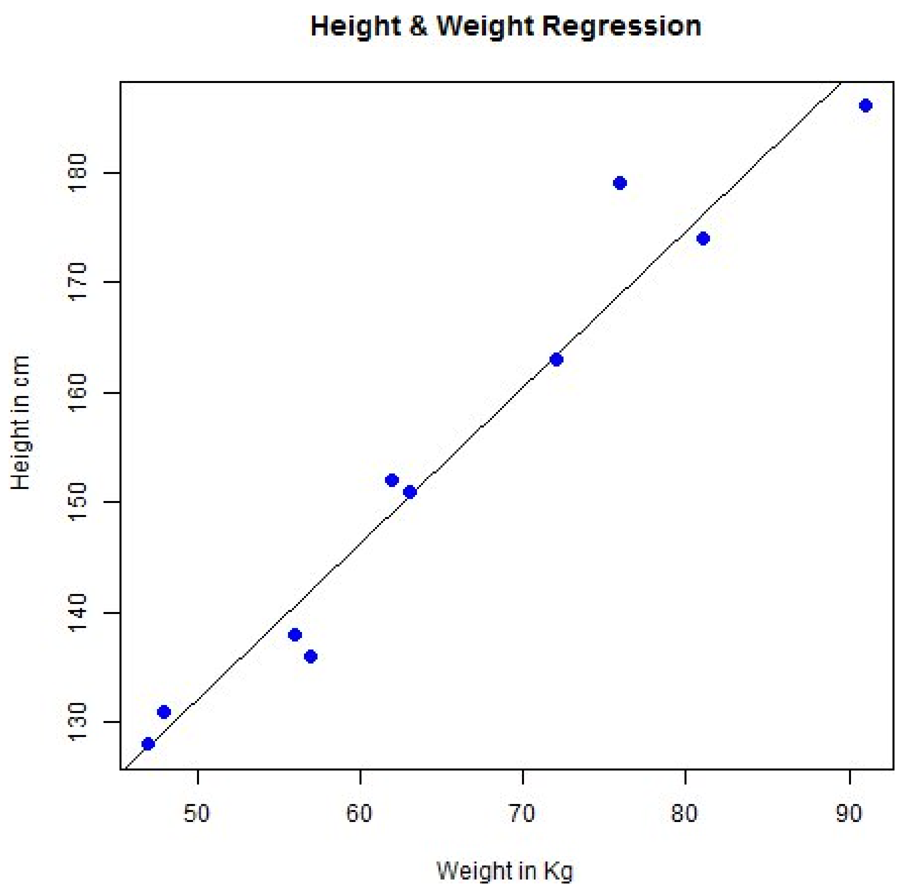
```
# The predictor vector.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

# The response vector.
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.
relation <- lm(y~x)

# Find weight of a person with height 170.
a <- data.frame(x = 170)
result <- predict(relation,a)
print(result)
```

Result:

```
      1
76.22869
```

Visualize the Regression Graphically

```
# Create the predictor and response variable.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
relation <- lm(y~x)

# Give the chart file a name.
png(file =
"linearregression.png")

# Plot the chart.
plot(y,x,col = "blue",main = "Height S Weight Regression",
abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in cm")

# Save the file.
```


Height & Weight Regression
Height in cm
Weight in Kg
130
140
150
160
170
180
50
60
70
80
90

Experiment 8: **Using R project for data visualization of social media**
Aim: To perform data visualization using R programming
Theory:

Data visualization is the technique used to deliver insights in data using visual cues such as graphs, charts, maps, and many others. This is useful as it helps in intuitive and easy understanding of the large quantities of data and thereby make better decisions regarding it.

Data Visualization in R Programming Language
The popular data visualization tools that are available are Tableau, Plotly, R, Google Charts, Infogram, and Kibana. The various data visualization platforms have different capabilities, functionality, and use cases. They also require a different skill set. This article discusses the use of R for data visualization.

R is a language that is designed for statistical computing, graphical data analysis, and scientific research. It is usually preferred for data visualization as it offers flexibility and minimum required coding through its packages.

Types of Data Visualizations
Some of the various types of visualizations offered by R are:

Bar Plot

There are two types of bar plots- horizontal and vertical which represent data points as horizontal or vertical bars of certain lengths proportional to the value of the data item. They are generally used for continuous and categorical variable plotting. By setting the **horiz** parameter to true and false, we can get horizontal and vertical bar plots respectively.

Bar plots are used for the following scenarios:

- To perform a comparative study between the various data categories in the data set.
- To analyze the change of a variable over time in months or years.

Histogram

A histogram is like a bar chart as it uses bars of varying height to represent data distribution. However, in a histogram values are grouped into consecutive intervals called bins. In a Histogram, continuous values are grouped and displayed in these bins whose size can be varied.

For a histogram, the parameter **xlim** can be used to specify the interval within which all values are to be displayed.
Another parameter **freq** when set to *TRUE* denotes the frequency of the various values in the histogram and when set to *FALSE*, the probability densities are represented on the y-axis such that they are of the histogram adds up to one.
Histograms are used in the following scenarios:

- To verify an equal and symmetric distribution of the data.
- To identify deviations from expected values.

Box Plot

The statistical summary of the given data is presented graphically using a boxplot. A boxplot depicts information like the minimum and maximum data point, the median value, first and third quartile, and interquartile range.

Box Plots are used for:

- To give a comprehensive statistical description of the data through a visual cue.
- To identify the outlier points that do not lie in the inter-quartile range of data.

Scatter Plot

A scatter plot is composed of many points on a Cartesian plane. Each point denotes the value taken by two parameters and helps us easily identify the relationship between them.

Scatter Plots are used in the following scenarios:

- To show whether an association exists between bivariate data.
- To measure the strength and direction of such a relationship.

Heat Map

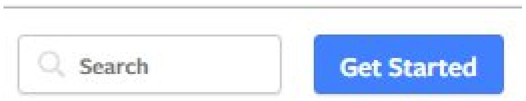
Heatmap is defined as a graphical representation of data using colors to visualize the value of the matrix. heatmap() function is used to plot heatmap.

Syntax: *heatmap(data)*
Parameters: *data: It represent matrix data, such as values of rows and columns*
Return: *This function draws a heatmap.*

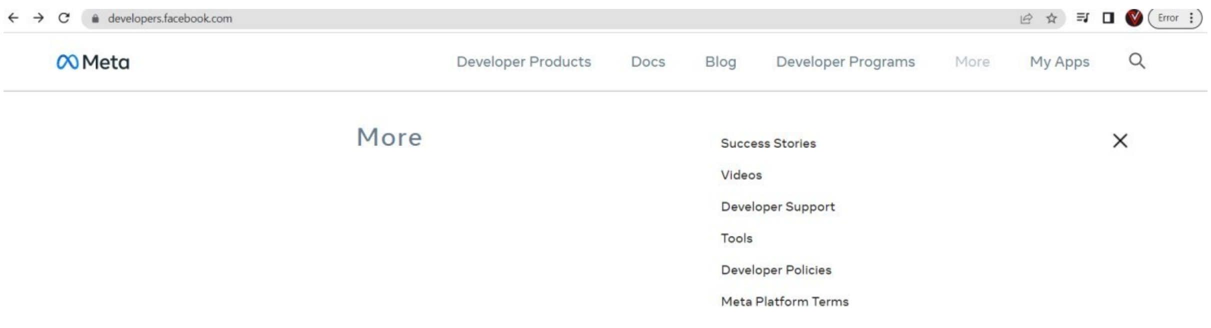
Procedure:

Step 1: Facebook Developer Registration

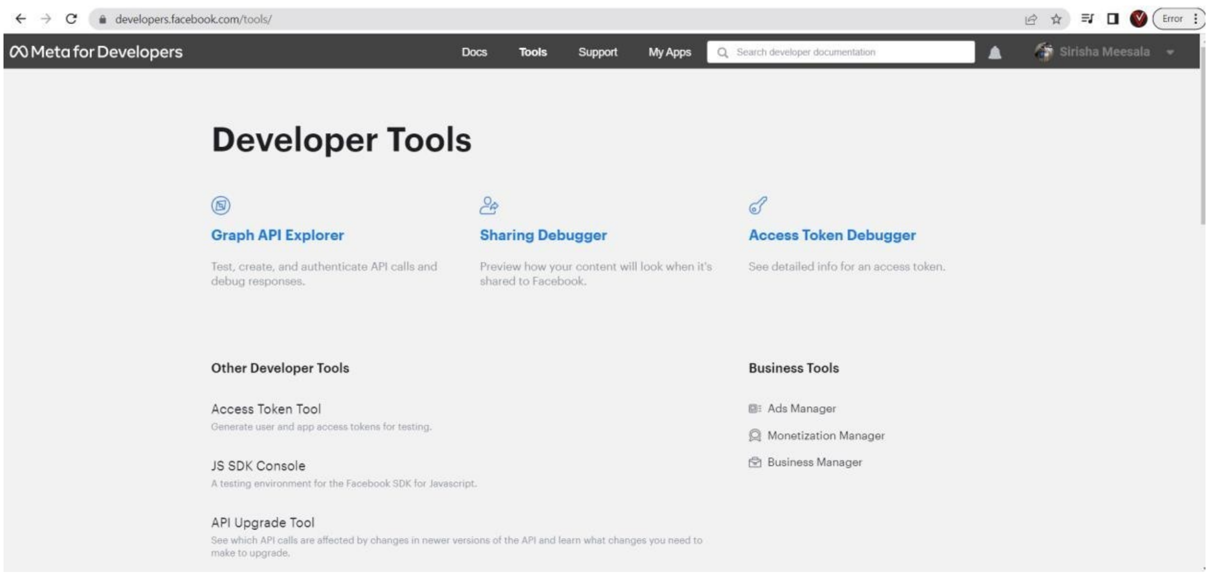
Go to https://developers.facebook.com and register yourself by clicking on **Get Started** button at the top right of page (See the snapshot below). After it would open a form for registration which you need to 昀椀ll it to get yourself registered.



<u>Step 2</u>: Click on tools



<u>Step 3</u>: Click on graph Api explorer


developers.facebook.com/tools/
Meta for Developers
Docs
Tools
Support
My Apps
Search developer documentation
Sirisha Meesala

Developer Tools

Graph API Explorer
Test, create, and authenticate API calls and debug responses.

Sharing Debugger
Preview how your content will look when it's shared to Facebook.

Access Token Debugger
See detailed info for an access token.

Other Developer Tools

Access Token Tool
Generate user and app access tokens for testing.

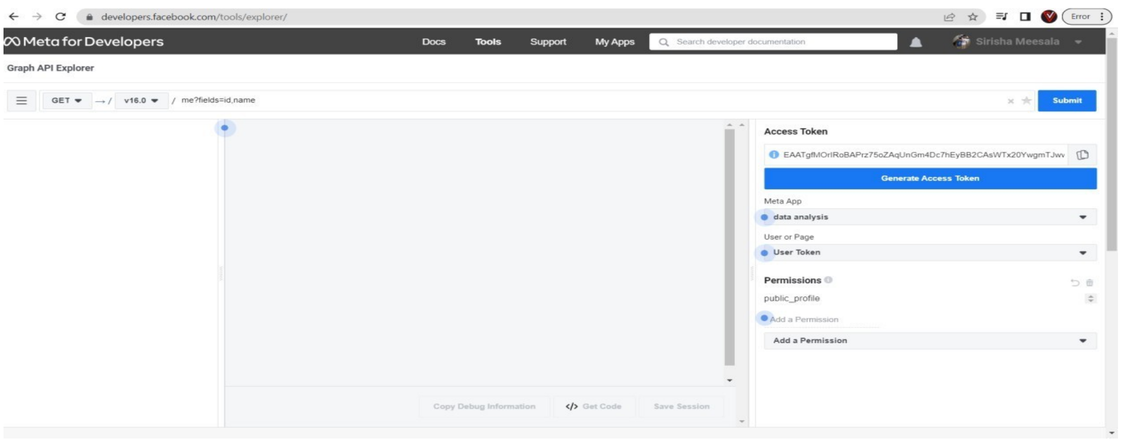
JS SDK Console
A testing environment for the Facebook SDK for Javascript.

API Upgrade Tool
See which API calls are affected by changes in newer versions of the API and learn what changes you need to make to upgrade.

Business Tools

Ads Manager
Monetization Manager
Business Manager

<u>Step 4</u>: Copy the access token



GET → / v16.0 / me?fields=id,name

Access Token
EAATgfMOrIRoBAPrz75oZAqUnGm4Dc7hEyBB2CAsWTx20YwgmTJwv

Meta App: data analysis

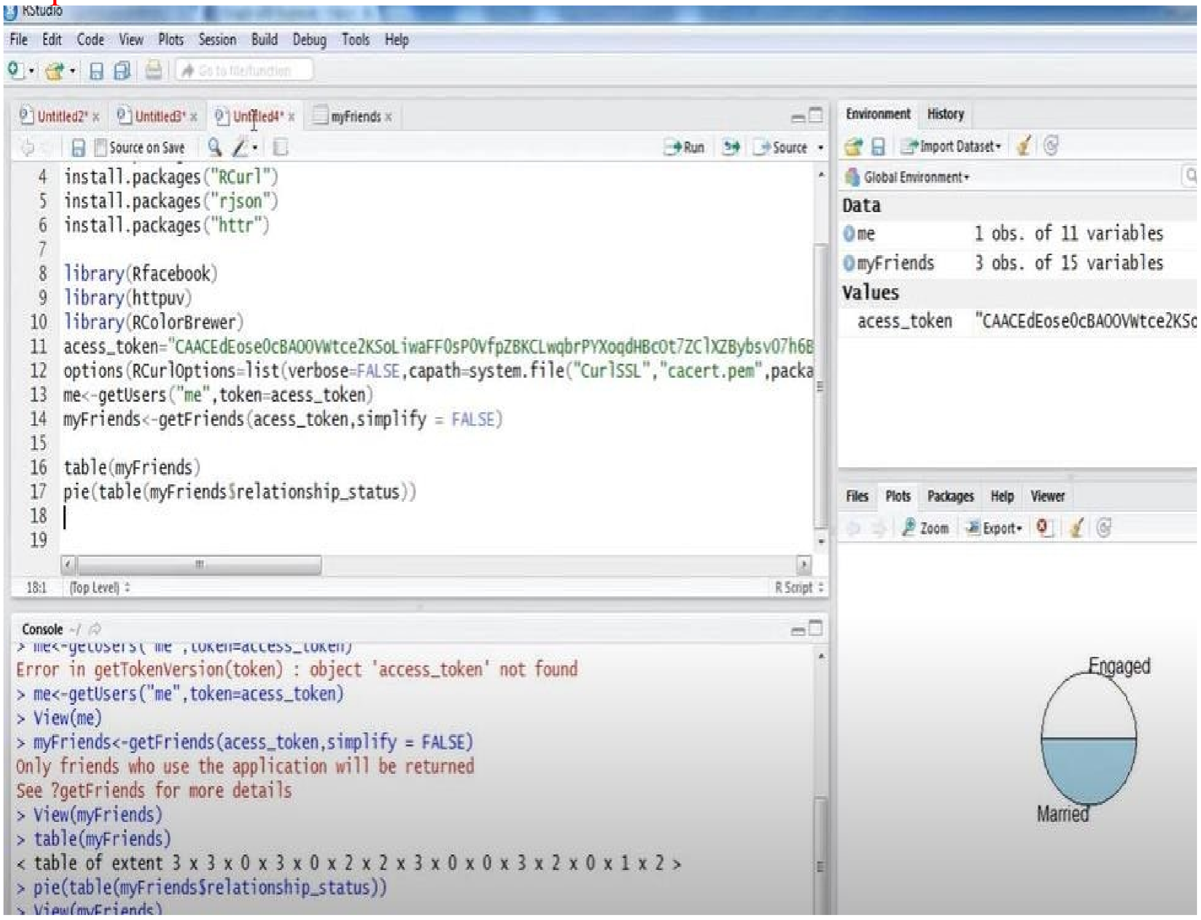
User or Page: User Token

Permissions
public_profile

Copy the access token
Go to R studio and write this Script

```
install.packages("httpuv")
install.packages("Rfacebook")
install.packages("RcolorBrewer")
install.packages("Rcurl")
install.packages("rjson")
install.packages("httr")


library(Rfacebook)
library(httpuv)
library(RcolorBrewer)
acess_token="EAATgfMOrIRoBAOR9XUl3VGzbLMuWGb9FqGkTK3PFBu
RyUVZA                    WAL7ZBw0xN3AijCsPiZBylucovck4YUhU昀欛
WLMZBo640k2ZAupKgsaKog9736lec
P8E52qkl5de8M963oKG8KOCVUXqqLiRcI7yIbEONeQt0eyLI6LdoeZA65Hy
xf8so1 UMbywAdZCZAQBpNiZAPPj7G3UX5jZAvUpRLZCQ5SIG"

options(RCurloptions=list(verbose=FALSE,capath=system.昀椀
le("CurlSSL","cacert. pem",package = "Rcurl"),ssl.verifypeer=FALSE))
me<-getUsers("me",token=acess_token)
View(me)
myFriends<-getFriends(acess_token,simplify = FALSE)
table(myFriends)

pie(table(myFriends$gender))
```

output



```
install.packages("RCurl")
install.packages("rjson")
install.packages("httr")

library(Rfacebook)
library(httpuv)
library(RColorBrewer)
acess_token="CAACEdEose0cBAOOVWtce2KSoLiwaFF0sP0VfpZBKCLwqbrPYXoqdHBcOt7ZClXZBybsv07h6B
options(RCurlOptions=list(verbose=FALSE,capath=system.file("CurlSSL","cacert.pem",packa
me<-getUsers("me",token=acess_token)
myFriends<-getFriends(acess_token,simplify = FALSE)

table(myFriends)
pie(table(myFriends$relationship_status))
```

```
Error in getTokenVersion(token) : object 'access_token' not found
> me<-getUsers("me",token=acess_token)
> View(me)
> myFriends<-getFriends(acess_token,simplify = FALSE)
Only friends who use the application will be returned
See ?getFriends for more details
> View(myFriends)
> table(myFriends)
< table of extent 3 x 3 x 0 x 3 x 0 x 2 x 2 x 3 x 0 x 0 x 3 x 2 x 0 x 1 x 2 >
> pie(table(myFriends$relationship_status))
```

```
Data
me          1 obs. of 11 variables
myFriends   3 obs. of 15 variables
Values
acess_token   "CAACEdEose0cBAOOVWtce2KSo
```