20CSC393	Big Data Architecture & Core Technologies Lab	L	T	P	S	C	CH
Version 1.00		0	0	2	0	1	2
Pre-requisites/ Exposure	Big Data Engineering						
Co-requisites	20CST332						

COURSE OBJECTIVES

The Course aims to:

1. Demonstrate experiments based on HDP , Hadoop , HDFS and Apache Spark in the Hadoop ecosystem.
2. Develop the understanding of IBM Cognos.
3. To demonstrate the use of databases in IBM DB2 using Ambari and command Line.

COURSE OUTCOMES

On completion of this course, the students shall be able to :

I. Get prepared for different big data core technologies.

II. Design and identify Hadoop cluster functions.

III. Design and employ the Hadoop as a big data analytics tool.

IV. Implementation of IBM cognos as a big data analytics tool.

V. Study and design the databases using DB2 .

COURSE DESCRIPTION

This course is an advanced course for Big Data Analytics Program that will enable student to understand thefunctionality and the implementation of the Big Data Tools and technologies and there working.

Learning Material(s)

Reference Books					
Sr No	Title of the Book	Author Name	Volume/Edition	Publish Hours	Years
1	Big Data Analytics	Shankarmani	2nd	Wiley	2017
Text Books					
1	IBM AP Skills	IBM	2020	IBM	2020

COURSE CONTENT

Unit – 1 8 contact hours

1. Elucidate the function of the NameNode and DataNodes in a Hadoop cluster.
2. Write a program to implement file management tasks in Hadoop HDFS and perform Hadoop commands.
3. Write a program to implement a word count application using the MapReduce programming mode.
4. Write a program to develop a MapReduce application and implement a program that analyses weather data.

Unit -2 6 contact hours

5. Create A Sample Report that lists all of the sales representatives and the revenue they have generated to date. The report should include their name, position, city, and country. Sort the report by revenue, in descending order, and display revenue.
6. Explore a dimensionally-modeled relational data source and create a report that enables you to drill down to a lower level of detail.
7. The Vice President of Sales has requested a report that shows sales performance in each country for 2012. He wants to see the performance for representatives in Southern Europe.

Unit-3 6 contact hours

8. Perform the following: a. Viewing all databases, Creating a Database, Viewing all Tables in a Database, Creating Tables (With and Without Constraints), Inserting/Updating/Deleting Records in a Table, Saving (Commit) and Undoing (rollback).
9. Perform the following: a. Altering a Table, Dropping/Truncating/Renaming Tables, Backing up / Restoring a Database.
10. For a given set of relation schemes, create tables and perform the following Simple Queries, Simple Queries with Aggregate functions, Queries with Aggregate functions (group by and having clause), Queries involving- Date Functions, String Functions , Math Functions Join Queries- Inner Join, Outer Join Subqueries- With IN clause, With EXISTS clause.

MODE OF EVALUATION: The performance of students is evaluated as follows:

Unit – 1

Experiment – 1

CO Mapped : CO1

Aim :

Elucidate the function of the NameNode and DataNodes in a Hadoop cluster.

Procedure:

An HDFS cluster has two types of nodes operating in a master−slave pattern:

1. NameNode (the master) and

2. DataNodes (slaves/workers).

HDFS NameNode

1. NameNode is the main central component of HDFS architecture framework.

2. NameNode is also known as Master node.

3. HDFS Namenode stores meta-data i.e. number of data blocks, file name, path, Block IDs, Block location, no. of replicas, and also Slave related configuration. This meta-data is available in memory in the master for faster retrieval of data.

4. NameNode keeps metadata related to the file system namespace in memory, for quicker response time. Hence, more memory is needed. So NameNode configuration should be deployed on reliable configuration.

5. NameNode maintains and manages the slave nodes, and assigns tasks to them.

6. NameNode has knowledge of all the DataNodes containing data blocks for a given file.

7. NameNode coordinates with hundreds or thousands of data nodes and serves the requests coming from client applications.

Two files 'FSImage' and the 'EditLog' are used to store metadata information.

FsImage: It is the snapshot the file system when Name Node is started. It is an "Image file". FsImage contains the entire filesystem namespace and stored as a file in the NameNode's local file system. It also contains a serialized form of all the directories and file inodes in the filesystem. Each inode is an internal representation of file or directory's metadata.

EditLogs: It contains all the recent modifications made to the file system on the most recent FsImage. NameNode receives a create/update/delete request from the client. After that this request is first recorded to edits file.

Functions of NameNode in HDFS

1. It is the master daemon that maintains and manages the DataNodes (slave nodes).

2. It records the metadata of all the files stored in the cluster, e.g. The location of blocks stored, the size of the files, permissions, hierarchy, etc.

3. It records each change that takes place to the file system metadata. For example, if a file is deleted in HDFS, the NameNode will immediately record this in the EditLog.

4. It regularly receives a Heartbeat and a block report from all the DataNodes in the cluster to ensure that the DataNodes are live.

5. It keeps a record of all the blocks in HDFS and in which nodes these blocks are located.

6. The NameNode is also responsible to take care of the replication factor of all the blocks.

7. In case of the DataNode failure, the NameNode chooses new DataNodes for new replicas, balance disk usage and manages the communication traffic to the DataNodes.

HDFS DataNode

1. DataNode is also known as Slave node.

2. In Hadoop HDFS Architecture, DataNode stores actual data in HDFS.

3. DataNodes responsible for serving, read and write requests for the clients.

4. DataNodes can deploy on commodity hardware.

5. DataNodes sends information to the NameNode about the files and blocks stored in that node and responds to the NameNode for all filesystem operations.

6. When a DataNode starts up it announce itself to the NameNode along with the list of blocks it is responsible for.

7. DataNode is usually configured with a lot of hard disk space. Because the actual data is stored in the DataNode.

Functions of DataNode in HDFS

1. These are slave daemons or process which runs on each slave machine.

2. The actual data is stored on DataNodes.

3. The DataNodes perform the low-level read and write requests from the file system's clients.

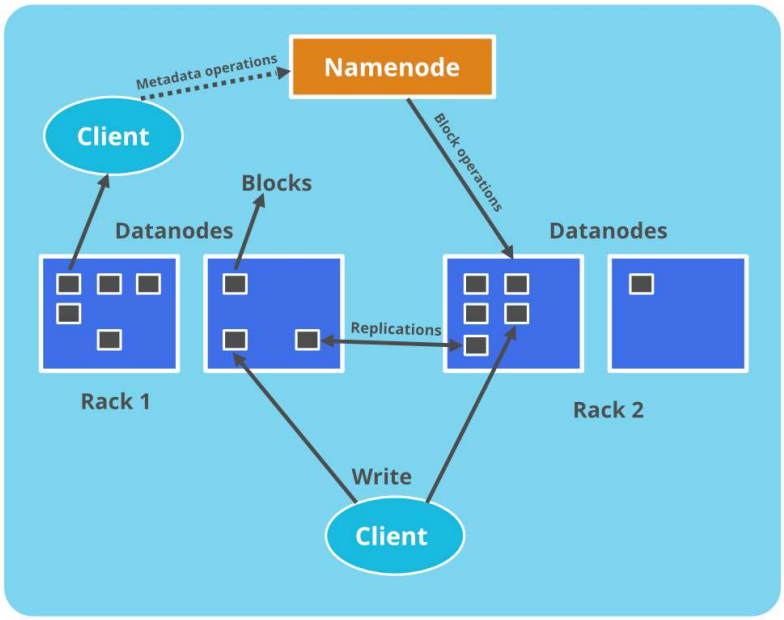
4. Every DataNode sends a heartbeat message to the Name Node every 3 seconds and conveys that it is alive. In the scenario when Name Node does not receive a heartbeat from a Data Node for 10 minutes, the Name Node considers that particular Data Node as dead and starts the process of Block replication on some other Data Node..

5. All Data Nodes are synchronized in the Hadoop cluster in a way that they can communicate with one another and make sure of

i. Balancing the data in the system

ii. Move data for keeping high replication

iii. Copy Data when required



Hardware Configuration

Hardware configuration of nodes varies from cluster to cluster and it depends on the usage of the cluster. In Some Hadoop clusters the velocity of data growth is high, in that instance more importance is given to the storage capacity. If the SLAs for the job executions are important and can not be missed then more importance is give to the processing power of nodes.

Often the term “Commodity Computers” is misunderstood. Commodity Computers or Nodes does not mean cheap or less powerful hardware, it just means in-expensive computer and deemphasize the need for specialized hardware.

Here is a sample configuration for NameNode and DataNode hardware configuration.

Name Node Configuration

Processors: 2 Quad Core CPUs running @ 2 GHz
RAM: 128 GB
Disk: 6 x 1TB SATA
Network: 10 Gigabit Ethernet

Data Node Configuration

Processors: 2 Quad Core CPUs running @ 2 GHz
RAM: 64 GB
Disk: 12-24 x 1TB SATA
Network: 10 Gigabit Ethernet

Experiment – 2

CO Mapped : CO1 and CO2

Aim : Write a program to implement file management tasks in Hadoop HDFS and perform Hadoop commands.

PROCEDURE :

With growing data velocity, the data size easily outgrows the storage limit of a machine. A solution would be to store the data across a network of machines. Such filesystems are called distributed filesystems. Since data is stored across a network all the complications of a network come in.This is where Hadoop comes in. It provides one of the most reliable filesystems. HDFS (Hadoop Distributed File System) is a unique design that provides storage for :

extremely largefiles with streaming data access pattern and it runs on commodity hardware. Let's elaborate the terms: Extremely large files: Here we are talking about the data in range of petabytes (1000 TB).

Streaming Data Access Pattern: HDFS is designed on principle of write-once and read-many-times. Once data is written large portions of dataset can be processed any number times.

Commodity hardware: Hardware that is inexpensive and easily available in the market. This is one of feature which specially distinguishes HDFS from other file system.

Syntax and Commands to Add, Retrieve and Delete Data From HDFS:

Adding Files and Directories to HDFS: Before you'll run Hadoop programs on data stored in HDFS, you 'll got to put the info into HDFS first. Let 's creates a directory and put a enter it. HDFS features a default working directory of /user/$USER, where $USER is your login user name. This directory isn't automatically created for you, though, so let 's creates it with the mkdir command. For the aim of illustration, we use chuck. you ought to substitut hadoopfs -put example.txte your user name within the example commands.

hadoopfs -put example.txt /user/chuck

Hadoop fs -mkdir /user/chuck

Retrieving Files from HDFS: The Hadoop command get copies files from HDFS back to thenative filesystem. To retrieve example.txt, we are ready to run the next command

hadoopfs -cat example.txt

Deleting Files from HDFS hadoopfs -rm example.txt. Command for creating a directory inhdfs is

"hdfsdfs –mkdir /lendicse".

Adding directory is done through the command "hdfsdfs –put lendi_english /".

Copying Data from NFS to HDFS:

Copying from directory command is "hdfsdfs –copyFromLocal/home/lendi/Desktop/shakes/glossary /lendicse/"

View the file by using the command "hdfsdfs –cat /lendi_english/glossary".

Command for listing of things in Hadoop is “hdfsdfs –lshdfs://localhost:9000/”. Command for Deleting files is “hdfsdfs –rm r/example”

Hadoop Commands:

1) ls: This command is employed to list all the files. Use lsr for recursive approach. it'suseful once we need a hierarchy of a folder.Syntax: bin/hdfsdfs -lsExample: bin/hdfsdfs -ls / It will print all the directories present in HDFS. bin directory contains executables so,bin/hdfs means we would like the executables of hdfs particularly dfs (DistributedFile System) commands.

suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -ls /

19/01/31 10:35:11 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic able

Found 4 items

1 suraj supergroup 13965969 2019-01-31 00:13 /input

drwxr-xr-x - suraj supergroup © 2019-01-31 01:30 /output

drwx------ - suraj supergroup © 2019-01-31 00:15 /tmp

drwxr-xr-x - suraj supergroup

2) mkdir: To create a directory. In Hadoopdfs there's no home directory by default. So,let’s first create it.Syntax:bin/hdfsdfs -mkdircreating home directory:hdfs/bin -mkdir /userhdfs/bin -mkdir /user/username

Example:bin/hdfsdfs -mkdir /geeks => '/' means absolute path

bin/hdfsdfs -mkdir geeks2 => Relative path -> the folder are going to be created relative tothe house directory.

19/01/31 10:53:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic able

suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -ls /19/01/31 10:53:56 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Found 5 items

drwxr-xr-x - suraj supergroup -rw-r--r-.1 suraj supergroup 0 2019-01-31 10:53 /geeks

13965969 2019-01-31 00:13 /inputdrwxr-xr-x - suraj supergroup 02019-01-31 01:30 /output

drwx------ - suraj supergroup 8 2019-01-31 08:15 /tmp

drwxr-xr-x suraj supergroup

3) touchz:It creates an empty file.

Syntax: bin/hdfsdfs -touchz

Example:bin/hdfsdfs -touchz /geeks/myfile.txt

4) copyFromLocal (or) put: To copy files/folders from local filing system to hdfs store.this is often the foremost important command. Local filesystem means the filespresent on the OS.

Syntax: bin/hdfsdfs -copyFromLocal

Example: Let's suppose we've a file AI.txt on Desktop which we would like to repeat to folder geeks present on hdfs.

bin/hdfsdfs -copyFromLocal ../Desktop/AI.txt /geek

5) cat: To print file contents.

Syntax: bin/hdfsdfs -cat <path>

Example:// print the content of AI.txt present// inside geeks folder.

bin/hdfsdfs -cat /geeks/AI.txt ->

6) copyToLocal (or) get: To copy files/folders from hdfs store to local file system.

Syntax: bin/hdfsdfs -copyToLocal<<srcfile(on hdfs)><local file dest>

Example: bin/hdfsdfs -copyToLocal /geeks ../Desktop/hero

(OR)

bin/hdfsdfs -get /geeks/myfile.txt ../Desktop/hero

myfile.txt from geeks folder will be copied to folder hero present on Desktop.

7) moveFromLocal: This command will move file from local to hdfs.

Syntax: bin/hdfsdfs -moveFromLocal<local src><dest(on hdfs)>

Example: bin/hdfsdfs -moveFromLocal ../Desktop/cutAndPaste.txt /geeks

8) cp: This command is used to copy files within hdfs. Lets copy folder geeks to geeks_copied.

Syntax: bin/hdfsdfs -cp<src(on hdfs)><dest(on hdfs)>

Example: bin/hdfs -cp /geeks /geeks_copied.

9) mv: This command is used to move files within hdfs. Lets cut-paste a file myfile.txt from geeks folder to geeks_copied.

Syntax: bin/hdfsdfs -mv <src(on hdfs)><src(on hdfs)>

Example: bin/hdfs -mv /geeks/myfile.txt /geeks_copied.

10) rmr: This command deletes a file from HDFS recursively. It is very useful command when you want to delete a non-empty directory.

Syntax : bin/hdfsdfs -rmr<filename/directoryName>

Example: bin/hdfsdfs -rmr /geeks_copied -> It will delete all the content inside the directory and then the directory itself.

Experiment – 3

CO mapped : CO1 and CO3

Write a program to implement a word count application using the MapReduce programming mode.

PROCEDURE :

Steps to execute MapReduce word count example

1. Create a text file in your local machine and write some text into it. $ nano data.txt

2. Check the text written in the data.txt file. $ cat data.txt

3. Create a directory in HDFS, where to kept text file. $ hdfs dfs -mkdir /test

4. Upload the data.txt file on HDFS in the specific directory. $ hdfs dfs -put /home/codegyani/data.txt /test

5. Write the MapReduce program using eclipse(any IDE).

6. Create the jar file of this program and name it countworddemo.jar.(Right Click on Project-> Click on Export-> Select export destination as Jar File-> Name the jar File(WordCount.jar) -> Click on next -> at last Click on Finish. Now copy this file into the Workspace directory of Cloudera)

7. Run the jar file Hadoop jar /home/codegyani/wordcountdemo.jar com.javatpoint.WC_Runner /test/data.txt /r_output

8. The output is stored in /r_output/part-00000

9. Now execute the command to see the output. hdfs dfs -cat /r_output/part-00000.

Output :

Experiment – 4

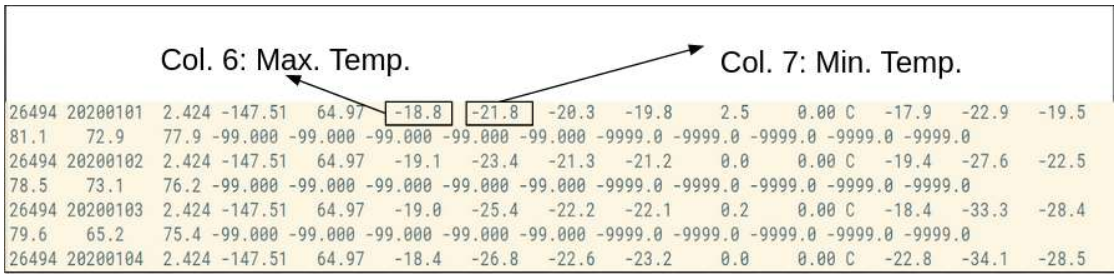
CO Mapped CO1 and CO3

Aim:

To develop a MapReduce application and implement a program that analyses weather data.

PROCEDURE :

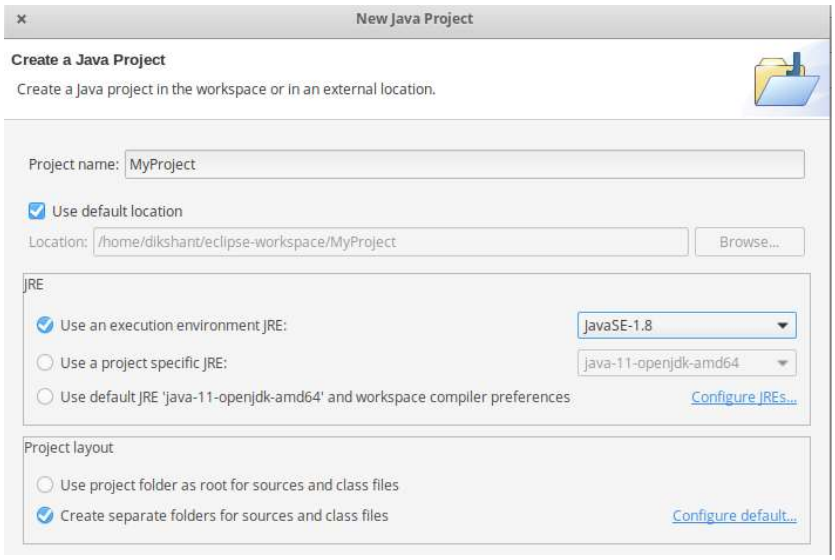
Step 1: Download the weather dataset



Columns 6 and 7 showing maximum and minimum temperatures

Step 2: Make a project in Eclipse

First Open **Eclipse** -> then select **File -> New -> Java Project** ->Name it **MyProject** -> then select **use an execution environment** -> choose **JavaSE-1.8** then **next** -> **Finish**.



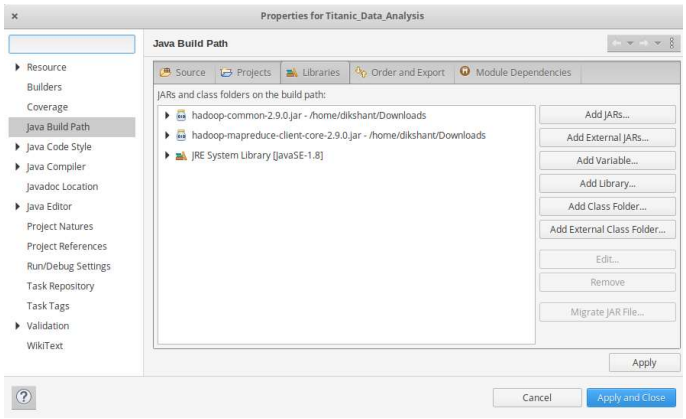
Code :

```
// importing Libraries

import java.io.IOException;

import java.util.Iterator;

import org.apache.hadoop.fs.Path;

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

```
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.conf.Configuration;

public class MyMaxMin {
    // Mapper

    /*MaxTemperatureMapper class is static and extends Mapper abstract class
    * having four Hadoop generics type  LongWritable, Text, Text, Text.        */
    public static class MaxTemperatureMapper extends
                    Mapper<LongWritable, Text, Text, Text> {
    public static final int MISSING = 9999;
            @Override
            public void map(LongWritable arg0, Text Value, Context context)
                            throws IOException, InterruptedException {
            String line = Value.toString();
                    if (!(line.length() == 0)) {
                            String date = line.substring(6, 14);
                                    float temp_Max = Float.parseFloat(line.substring(39, 45).trim());
                            float temp_Min = Float.parseFloat(line.substring(47, 53).trim());
                            if (temp_Max > 30.0) {
                                    context.write(new Text("The Day is Hot Day :" + date),
                                    new Text(String.valueOf(temp_Max)));                                    }
                            if (temp_Min < 15) {
                                    context.write(new Text("The Day is Cold Day :" + date),
```

```
                    new Text(String.valueOf(temp_Min)));        } } } }

public static class MaxTemperatureReducer extends
        Reducer<Text, Text, Text, Text> {
            public void reduce(Text Key, Iterator<Text> Values, Context context)
            throws IOException, InterruptedException {
        String temperature = Values.next().toString();
        context.write(Key, new Text(temperature));
    }       }

public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = new Job(conf, "weather example");
    job.setJarByClass(MyMaxMin.class);
    job.setMapOutputKeyClass(Text.class);
    job.setMapOutputValueClass(Text.class);
    job.setMapperClass(MaxTemperatureMapper.class);
    job.setReducerClass(MaxTemperatureReducer.class);
    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);
    Path OutputPath = new Path(args[1]);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    OutputPath.getFileSystem(conf).delete(OutputPath);
    System.exit(job.waitForCompletion(true) ? 0 : 1);    } }
```

Step 3 :
Now we add these external jars to our **MyProject**. Right Click on **MyProject** -> then select **Build Path**-> Click on **Configure Build Path** and select **Add External jars….** and add jars from it's download location then click -> **Apply and Close**.



- Now export the project as jar file. Right-click on **MyProject** choose **Export..** and go to **Java -> JAR file** click -> **Next** and choose your export destination then click -> **Next**.
choose Main Class as **MyMaxMin** by clicking -> **Browse** and then click -> **Finish** -> **Ok**.

Step 4:

Start our Hadoop Daemons

start-dfs.sh

start-yarn.sh

Step 5:

Move your dataset to the Hadoop HDFS.

Syntax:

hdfs dfs -put /file_path /destination

In below command / shows the root directory of our HDFS.

hdfs dfs -put /home/dikshant/Downloads/CRND0103-2020-AK_Fairbanks_11_NE.txt /

Check the file sent to our HDFS.

hdfs dfs -ls /

```
dikshant@dikshant-Inspiron-5567:~$ hdfs dfs -put /home/dikshant/Downloads/CRND0103-2020-AK_Fairbanks_11_NE
.txt /
dikshant@dikshant-Inspiron-5567:~$ hdfs dfs -ls /
Found 4 items
-rw-r--r--   1 dikshant supergroup      39711 2020-07-04 09:39 /CRND0103-2020-AK_Fairbanks_11_NE.txt
drwxrwxr-x+  - dikshant supergroup          0 2020-06-23 14:23 /Hadoop_File
drwxrwxrwx   - dikshant supergroup          0 2020-06-14 21:43 /tmp
drwxr-xr-x   - dikshant supergroup          0 2020-06-14 21:43 /user
dikshant@dikshant-Inspiron-5567:~$
```

Step 6:

Now Run your Jar File with below command and produce the output in **MyOutput** File.
Syntax:
hadoop jar /jar_file_location /dataset_location_in_HDFS /output-file_name

Command:
hadoop jar /home/dikshant/Documents/Project.jar /CRND0103-2020-AK_Fairbanks_11_NE.txt /MyOutput

```
dikshant@dikshant-Inspiron-5567:~$ hadoop jar /home/dikshant/Documents/Project.jar /CRND0103-2020-AK_Fairb
anks_11_NE.txt /MyOutput
20/07/04 09:44:40 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.sessi
on-id
20/07/04 09:44:40 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
20/07/04 09:44:41 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Im
```

Step 7:

Now Move to *localhost:50070/*, under utilities select *Browse the file system* and download **part-r-00000** in **/MyOutput** directory to see result.

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	dikshant	supergroup	38.78 KB	Jul 04 09:39	1	122.07 MB	CRND0103-2020-AK_Fairbanks_11_NE.txt
drwxrwxr-x+	dikshant	supergroup	0 B	Jun 23 14:23	0	0 B	Hadoop_File
drwxr-xr-x	dikshant	supergroup	0 B	Jul 04 09:44	0	0 B	MyOutput
drwxrwxrwx	dikshant	supergroup	0 B	Jun 14 21:43	0	0 B	tmp
drwxr-xr-x	dikshant	supergroup	0 B	Jun 14 21:43	0	0 B	user

Step 8:

See the result in the Downloaded File.

```
1 The Day is Cold Day :20200101   -21.8
2 The Day is Cold Day :20200102   -23.4
3 The Day is Cold Day :20200103   -25.4
4 The Day is Cold Day :20200104   -26.8
5 The Day is Cold Day :20200105   -28.8
6 The Day is Cold Day :20200106   -30.0
7 The Day is Cold Day :20200107   -31.4
8 The Day is Cold Day :20200108   -33.6
9 The Day is Cold Day :20200109   -26.6
10 The Day is Cold Day :20200110   -24.3
```

In the above image, you can see the top 10 results showing the cold days. The second column is a day in yyyy/mm/dd format. For Example, **20200101** means
year = 2020

month = 01

Date = 01

Unit – 2

Experiment – 5

Co Mapped : CO1 and CO4

Create A Sample Report that lists all of the sales representatives and the revenue they have generated to date. The report should include their name, position, city, and country. Sort the report by revenue, in descending order, and display revenue.

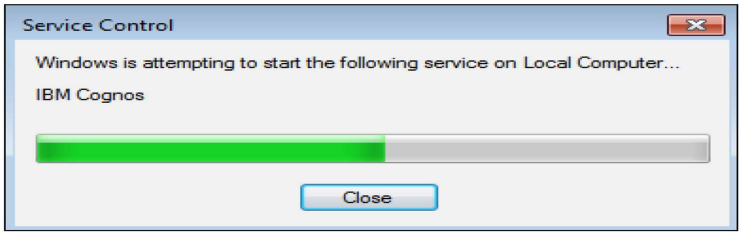
Country	City	Last name	First name	Position name	Revenue
Switzerland	Genève	Bruno	Fausta	Level 3 Sales Representative	$79,955,838.92
Switzerland	Genève	Giordano	Fiorenza	Level 3 Sales Representative	$72,784,594.30
Switzerland	Genève	Chambers	Warren	Level 3 Sales Representative	$62,843,459.76
Finland	Kuopio	Lindholm	Helena	Level 3 Sales Representative	$59,799,153.93
Korea	Seoul	Kim	Chang-ho	Level 3 Sales Representative	$59,422,592.32
United States	Los Angeles	Laurel	Charles	Level 3 Sales Representative	$59,406,874.73
Switzerland	Genève	Bichot	Lotta	Level 3 Sales Representative	$54,436,904.60
Netherlands	Amsterdam	Jansen-Velasquez	Belinda	Level 3 Sales Representative	$52,822,234.19

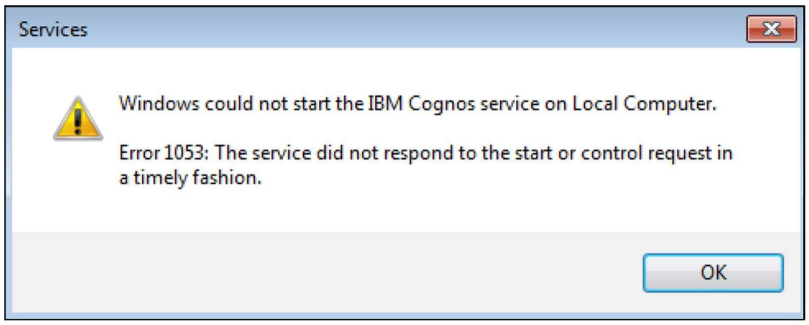
PROCEDURE :

Task 1. Maunally Start The IBM Cognos Service.

1. From the **Windows** desktop, locate the **Services** button in the task bar, at the bottom of the
2. Click **Services**, to view the list of Windows
3. In the list, scroll down until you locate the **IBM Cognos**
4. Click on the **IBM Cognos** service, to select.
5. From the Services menu bar, click **Start Service** .

After a moment, a progress bar appears - showing the Service is being started. This may take a few seconds, to several minutes, to complete.
You may also see the following message, which is normal:





6. Click **OK**, then continue to monitor and check that the Service is

7. Periodically check to see if the **IBM Cognos** service has started, by clicking **Refresh**  .

   Once the service has started, you will see a status of Started:

8. Once you have confirmed that the Service has started, close the **Services** window.

9. You may now proceed to Task

Task 2. Open IBM Cognos Analytics- Reporting And Then Choose A Report Template.

1. Open your internet browser, in the address box type **http://vclassbase:9300/bi**, and then press **Enter**.

2. On the IBM Cognos Analytics page, in the **User ID** box, type **brettonf**, in the **Password** box, type **Education1**, and then click **Sign in**. The Welcome to IBM Cognos Analytics page appears.

3. From the side panel on the left, click **New**  , and then click **Report**.
4. From the **New** dialog, select the **Blank** template, and then click **OK**.

Task 3. Add a data source and a list.

1. From the side panel, click **Data**  . .

2. Under the **Source** tab, click **Add report data**.

3. Navigate to: **Samples\Models\GO data warehouse (query)**. The results appear as follows:

4. Click **Open**.
5. On the **Application** bar, click **Page views** , and then click **Page preview**.
6. Click **Add** in the center of the This will bring up a set of data container choices.
7. Click the **List** template, and then click **OK** to accept the default query

Task 4. Add data to the list.

1. On the side panel, **Source** tab, expand the **Sales and Marketing (query)** folder , expand the **Sales (query)** namespace , and then expand the **Employee by region** query subject .
2. Double-click the **Country** query item to add it to the list report The list report objectnow has one column.
3. Double-click **City** to add it to the list report object. City is automatically added to theend of the
4. Right-click **Last name**, and then click **Properties**.
   The Properties dialog box appears, with details about the item.
5. Click **Close**.
6. Click **First name**, and then Ctrl-click **Last name**, **Employee level** and Position name.
7. Right-click **Position name**, and then click **Insert**.

The items are added to the list in the order in which they are selected.A section of the results appear as follows:

Country	City	First name	Last name	Employee level	Position name
Switzerland	Genève	Aaghie	Heiman	4	Information Technology Manager
Switzerland	Genève	Aaghie	Heiman	5	Software Engineer
Switzerland	Genève	Aaltje	Hansen	6	Level 1 Sales Representative
Brazil	São Paulo	Abel	Antunes	4	Product Manager
Switzerland	Genève	Abram	Ruiz	6	Level 2 Sales Representative
Italy	Milano	Ada	Morales	6	Warehouse Worker
Italy	Milano	Adara	Cruz	5	Accountant 2

8. From the **Source** tab, expand the **Sales fact** query subject, and then click and drag **Revenue** to add it to the end of the list (you should see a black line inside of a white bar, indicating the correct drop zone).

   If you place the query item outside of the list report object you will receive a message indicating that you have created a singleton. You instead want the new query item to be added to the end of the List.

   You would like to see Last name appearing before First name.

9. In the work area, click the data for **Last name** (not the header) to select the list column body, and then drag it to the left of the **First name** list column

   A flashing black bar appears when the item is over a drop zone.

   Note: Make sure that the list column body is selected by clicking any one of the cells in the column, not the column header. To check to see what element of the report you have selected, check the title bar of the Properties pane.

   Now that you have built the report you can view the data items in the query.

Task 5. View the data items in the query.

1. On the side panel, click **Navigate**  , then on the content pane, click **Query explorer**  , and then click **Query1**.

   The data items you added to the list appear in the Data Items pane for the query. The names of the data items correspond to the column titles in the report layout.

2. In the **Data Items** pane, click **Position name**.

   You want to view information about the data the Position name data item retrieves from the data source.

3. From the **Application** bar, click **Show properties**  .
4. In the **Properties** pane, double-click the **Expression** property**.**

   In the Data item expression dialog box, you can see that this data item retrieves data from the Position name query item, in the Employee by region query subject, in the Sales (query) namespace.

5. Click **OK**, and then in the **Data Items** pane, click **Last name**.
6. In the **Properties** pane, double-click the **Expression**

   The Data item expression dialog box appears. You can see that this data item retrieves data from the Last name query item, in the Employee by region query subject, in the Sales (query) namespace.

7. Click **OK** to close the dialog

8. In the content pane, click **Page explorer**  , and then click **Page1** to return to the work

Task 6. Remove a column from the report

It has been decided that Employee level in the list report object is not needed in the report. You will remove it from the list.

1. In the list report object, click the data cell for **Employee level** (not the header).

2. From the now visible container toolbar, click **More**  , and then click **Cut**  . Thecolumn is removed from the list

3. On the content toolbar, click **Query explorer**, and then click **Query1**.

The Employee level data item still appears in the Data Items pane. Although you removed the Employee level data item from the report layout in Page Explorer, thedata item has not been removed from the query. However, keeping the data item inthe query can be useful for other tasks such as creating a calculation.

Other examples of where you would keep a data item in the query, but remove it from the report layout are: creating an expression based on the query item, or, using this item when sorting or formatting data in the list.

4. On the content toolbar, click **Page explorer**, and then click **Page1** to return to the work

5. On the **Application** bar, click **Undo**  .

6. With the **Employee level** data column still selected, on the container toolbar, click **More**, and then click **Delete**  .

7. On the content toolbar, click **Query explorer**, and then click **Query1**.

   The Employee level data item has been removed from the report layout and the query and no longer appears in the Data Items pane.

Task 7. Format and sort the data, and then run the report.

1. On the content toolbar, click **Page explorer**, and then click **Page1**.

2. In the list report object, click the data cell for **Revenue**, in the list column body (not the column title).

   The Revenue cells are highlighted to show that they are selected. The Properties pane shows the properties for this column.

3. On the container toolbar, click **Sort**  , and then click **Descending**. Our sales reps will now be ranked starting with our top performers.

4. With the **Revenue** column still selected, in the **Properties** pane, under the **DATA** category, click **Data format**, and then click the **ellipsis**  . The Data Format dialog box appears.

5. In the **Format type** list, select **Currency**.

6. Under **Properties**, click **Currency**, click the down arrow button in the column to the right of **Currency**, and then select **$ (USD) - United States of America, dollar** from the

Revenue will now be displayed in American dollars. By default, it will use a comma as a Thousands separator, and two decimal places.

Note: Changing the currency will not perform a currency conversion (for example, it will not convert one currency into the value of another). It will simply show the value with a different currency symbol, thousands separator, decimal place, and so on. If you want to see data displayed in a particular currency, the data must be stored in the data source in that currency.

7. Click **OK**.

8. Click **Page views**  , and then click **Page design**. The results appear as follows:

Country	City	Last name	First name	Position name	Revenue
<Country>	<City>	<Last name>	<First name>	<Position name>	<Revenue>
<Country>	<City>	<Last name>	<First name>	<Position name>	<Revenue>
<Country>	<City>	<Last name>	<First name>	<Position name>	<Revenue>

Page design mode is an alternate way to edit report objects such as the List.

9. On the main toolbar, click **Run options**  , and then click **Run HTML.**

A section of the results appear as follows:

Country	City	Last name	First name	Position name	Revenue
Switzerland	Genève	Bruno	Fausta	Level 3 Sales Representative	$79,955,838.92
Switzerland	Genève	Giordano	Fiorenza	Level 3 Sales Representative	$72,784,594.30
Switzerland	Genève	Chambers	Warren	Level 3 Sales Representative	$62,843,459.76
Finland	Kuopio	Lindholm	Helena	Level 3 Sales Representative	$59,799,153.93
Korea	Seoul	Kim	Chang-ho	Level 3 Sales Representative	$59,422,592.32
United States	Los Angeles	Laurel	Charles	Level 3 Sales Representative	$59,406,874.73
Switzerland	Genève	Bichot	Lotta	Level 3 Sales Representative	$54,436,904.60
Netherlands	Amsterdam	Jansen-Velasquez	Belinda	Level 3 Sales Representative	$52,822,234.19

You can see that revenue is sorted in descending order.

Experiment : 6

CO Mapped : CO1 and CO4.

Aim: . Explore a dimensionally-modeled relational data source and create a report that enables you to drill down to a lower level of detail.

2011	Canada	Star Dome	Quantity
Q1 2011	Canada	Star Dome	621
Q2 2011	Canada	Star Dome	531
Q3 2011	Canada	Star Dome	586
Q4 2011	Canada	Star Dome	665

PROCEDURE :

Task 1. Examine a dimensionally modelled relational data source.

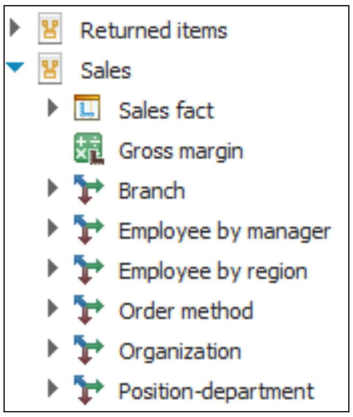
1. From the side panel, click **New**  , and then click **Report**.
2. Select **Blank**, and then click **OK**.
3. Using Data > Source > Add report data, navigate to Team content > Samples > Models > GO data warehouse (analysis).
4. Click **Open**.

   The Source tab on the left, displays the folders available in the package. Notice the folder symbols  .

5. Click **Add**  in the center of the screen, click **List**, and then click **OK**.
6. Expand **Sales and Marketing (analysis)**.
   You see the namespaces in the Sales and Marketing (analysis) folder. Notice the namespace symbols  .

7. Expand the **Sales**

   A section of the results appear as follows:



The available measures and dimensions are displayed in the data tree. Notice the measures query subject and the dimensions .

Task 2. Continue examining the data source.

1. Expand the **Sales fact** measures query

   You see all the measures available in the Sales fact measures query. Notice the measures .

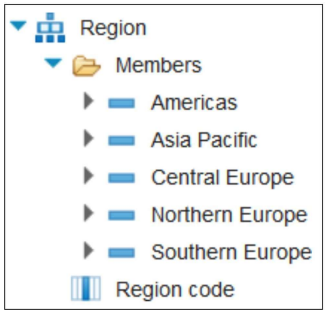
2. Expand the **Retailers** dimension , and then expand the **Retailers** hierarchy .

   You see a Members folder and five levels. Notice the level .

3. Expand the **Region** level .

   The Members folder and the Region code query item display in the data tree. Notice the query item .

4. Expand the **Members** folder (under Region) to see the five sales The results appear as follows:



Task 3. Add items to the list report object.

You need to create a report that shows the quantity of Star Dome tents sold in Canada in 2011. Because this is dimensionally-modeled relational data, you can drill down to a greater level of detail than in a relational model.

1. Expand the **Time** dimension, **Time** hierarchy, **Year** level, and **Members**.

2. Drag **2011** to the list report object, into the work

Notice how you can add specific members to a report, instead of having all years added and filtering for only the years you want (as in relational data sources).

3. Under the **Retailers** dimension, **Retailers** hierarchy, **Region** level, **Members** folder, expand the **Americas** member, and then drag the **Canada** member to the list report
4. Expand the **Products** dimension, **Products** hierarchy, **Product line** level, **Members** folder, **Camping Equipment** member, **Tents** member, and then drag **Star Dome** to the listreport object
5. Expand **Sales fact** measures (if necessary), and then drag the **Quantity** measure to the list report object. The results appear as follows:

2011	Canada	Star Dome	Quantity
<2011>	<Canada>	<Star Dome>	<Quantity>
<2011>	<Canada>	<Star Dome>	<Quantity>
<2011>	<Canada>	<Star Dome>	<Quantity>

Task 4. Allow drill-up and drill-down on the report.

1. From the side panel, click **Navigate**.
2. Under **Find**, click **Report**  , and then on the **Application** bar, click Show properties.
3. Under **DATA**, change the **Drill-up and drill-down** property to **Yes**.
4. Click the **Page explorer**, and then click **Page 1**.
5. Select the entire List data container by clicking the selection  handle in the upper lef.
6. If necessary, open the **Properties**
7. Click **Select Ancestor**  (next to the List panel header title) in the upper left of the Properties panel, and then click **Report**.
8. Under **RUNNING & VALIDATING**, change the property for Run with full interactivityto No.
9. On the **Application** bar, click **Run options**, and then click **Run HTML**. You see that2,403 Star Dome tents were sold in Canada, in 2011.

2011	Canada	Star Dome	Quantity
2011	Canada	Star Dome	2,403

10. Click **2011** to drill-down.

The results appear as follows:

2011	Canada	Star Dome	Quantity
Q1 2011	Canada	Star Dome	621
Q2 2011	Canada	Star Dome	531
Q3 2011	Canada	Star Dome	586
Q4 2011	Canada	Star Dome	665

Experiment – 7

CO Mapped : CO1 and CO4

Aim : The Vice President of Sales has requested a report that shows sales performance in each country for 2012. He wants to see the performance for representatives in Southern Europe.

PROCEDURE :

Task 1. Create the list.

1. Add the following query items to a new list template:

- Employee by region: **Country**, **City**, **First name**, **Last name**, **Position name**
- Sales fact: **Revenue**

Country	City	First name	Last name	Position name	Revenue
<Country>	<City>	<First name>	<Last name>	<Position name>	<Revenue>
<Country>	<City>	<First name>	<Last name>	<Position name>	<Revenue>
<Country>	<City>	<First name>	<Last name>	<Position name>	<Revenue>

2. Ctrl-click **<Country>** and **<City>**, and then on the list toolbar, click Group / Ungroup.
3. Click **<Country>**, on the list toolbar click **Headers & footers**, and then click Createheader.
4. With **<Country>** still selected, press the **Delete** key to delete theredundant **<Country>** list column body.
5. Click the **<Revenue>** list column body, on the list toolbar click **Summarize**, and thenclick **Total**.
6. Click the **<Revenue>** list column body, on the list toolbar click **Sort**, and then

click **Descending**.

7. Run the report in **HTML**.

   A section of the results appear as follows:

City	First name	Last name	Position name	Revenue
Australia				
Melbourne	Donald	Ward	Level 2 Sales Representative	19,815,234.63
	Jackie	Fulford	Level 2 Sales Representative	19,456,734.01
	Alice	Walter	Level 3 Sales Representative	19,040,701.32
	Dave	Smythe	Level 1 Sales Representative	16,652,383.41
	John	Sinden	Level 2 Sales Representative	4,965,193.22
	Jake	Cartel	Level 1 Sales Representative	4,283,418.14
	Jonathan	Farrel	Level 1 Sales Representative	2,260,515.45
	Donald	Neely	Level 1 Sales Representative	1,089,148.84
Melbourne - Total				**87,563,329.02**
Australia - Total				**87,563,329.02**

8. Close rendered report tab.

Task 2. Add a filter to show sales from 2012.

1. Select the list data container by clicking  in the upper left corner of the list.
2. On the list toolbar, click **Filters**  , and then click **Edit Filters**.

   The Filters - Query 1 dialog box appears. There are two tabs: one for creating filters at the detail level, and one for creating filters at the summary level.

3. With the **Detail Filters** tab selected, click **Add**  , click **Advanced**, and then click **OK**.
4. Under **Available Components**, from the **Source** tab, expand **Sales and Marketing (query)**, expand **Sales (query)**, and then expand **Time**.
5. Create and validate the following expression. (using the Hint outlined below, you cancreate the expression differently): **[Sales (query)].[Time].[Year]=2012**

   Hint:

   Drag Year from the Time query subject, into the Expression Definition There aredifferent ways of creating filters to achieve the same result:

- create the expression [Sales (query)].[Time].[Date]between 2012-01-01 and 2012-12-31
- create filters by adding operators and conditions to query items using SQL syntax
- Click Validate  to check the syntax of the expression.

6. Click **OK** to close the **Detail filter expression** dialog box, and then click **OK** to closethe **Filters - Query1** dialog
7. Run the report in **HTML**.
8. At the bottom of the page, click **Bottom** to navigate to the end of the A section of theresults appear as follows:

Seattle	George	Harrows	Level 3 Sales Representative	17,924,373.12
	Bart	Scott	Level 2 Sales Representative	14,538,997.37
	Audrey	Lastman	Level 3 Sales Representative	13,535,227.17
	Melanie	White	Level 1 Sales Representative	6,906,978.7
Seattle - Total				**52,905,576.36**
United States - Total				**164,986,189.21**
Overall - Total				**1,495,891,100.9**

Only 2012 sales are included in the report. On the last page of the report, the Overall - Total revenue is $1,495,891,100.90 for 2012.

9. Close the rendered report tab.

Task 3. Filter data to show only Southern European countries.

The Southern European countries consist of Austria, Italy, and Spain.

1. Select the list data container, on the list toolbar, click **Filters**, and then click Edit Filters.

   The Filters - Query 1 dialog box appears showing the detail filter you just created. Youwill create another detail filter.

2. Click **Add**, ensure that **Country** is selected under **Custom based on data item**, and then click **OK**.
3. In the **Values** section, ensure that **Specific values** is selected from the Text pattern matching is also available and includes:

- Starts with
- Ends with
- Contains
- Matches SQL pattern

Advanced search options are also available.

4. From the **Values** list, click **Austria**, and then Ctrl+click **Italy**.
5. Click the **arrow**  to add the items to the **Selected values**
6. Click **Page down**  , click **Spain**, and then add it into the **Selected values** pane.
7. Click **OK** to close the **Filter condition** dialog box, and then click **OK** to close the **Filters** dialog box.
8. Run the report in **HTML**.

A section of the results appear as follows:

City	First name	Last name	Position name	Revenue
Austria				
Wien	Sabine	Grüner	Level 3 Sales Representative	12,193,198.67
	Jutta	Shulz	Level 2 Sales Representative	9,938,792.37
	Thomas	Schirmer	Level 1 Sales Representative	6,216,976.62
Wien - Total				**28,348,967.66**
Austria - Total				**28,348,967.66**
Italy				
Milano	Mario	Esposito	Level 2 Sales Representative	11,284,621.77
	Sergio	Ferrari	Level 1 Sales Representative	9,590,004.91
	Alessandra	Torta	Level 3 Sales Representative	9,049,090.7
	Alberto	Pera	Level 1 Sales Representative	6,603,296.71
	Silvano	Allessori	Level 2 Sales Representative	4,859,409.87
	Rolando	Giordano	Level 2 Sales Representative	4,235,729.57
Milano - Total				**45,622,153.53**
Italy - Total				**45,622,153.53**
Spain				
Bilbao	Tomás	Iglesias	Level 3 Sales Representative	11,769,059.22
	Yolanda	Torres	Level 3 Sales Representative	11,611,178.39
	Agatha	Reyes	Level 2 Sales Representative	7,475,301.46
	Anica	Torres	Level 1 Sales Representative	5,401,311.8
	Lara	Broschat	Level 1 Sales Representative	5,210,721.27
Bilbao - Total				**41,467,572.14**
Spain - Total				**41,467,572.14**
Overall - Total				**115,438,693.33**

You can see that in 2012, Italy generated the most revenue of Southern European countries, and Sabine Grüner from Austria earned the top sales rep award.

Unit-3

Experiment 8

CO Mapped : CO1 and CO5

Aim : Perform the following: a. Viewing all databases, Creating a Database, Viewing all Tables in a Database, Creating Tables (With and Without Constraints), Inserting/Updating/Deleting Records in a Table, Saving (Commit) and Undoing (rollback).

Consider the Company database with following tables

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
	Research	5	333445555	1988-05-22
	Administration	4	987654321	1995-01-01
	Headquarters	1	888665555	1981-06-19

Perform the following:

1. Create company database
2. Viewing all databases
3. Viewing all Tables in a Database,
4. Creating Tables (With and Without Constraints)
5. Inserting/Updating/Deleting Records in a Table
6. Saving (Commit) and Undoing (rollback)

PROCEDURE :

1. Creating a Database
   CREATE DATABASE Company;

2. Viewing all databases
   SHOW DATABASES;

3. Viewing all Tables in a Database,
   SHOW tables;

4. Creating Tables (With and Without Constraints)

```
CREATE TABLE DEPARTMENT
(DNO VARCHAR2 (20) PRIMARY KEY,
DNAME VARCHAR2 (20),
MGRSTARTDATE DATE);

CREATE TABLE EMPLOYEE
(SSN VARCHAR2 (20) PRIMARY KEY,
FNAME VARCHAR2 (20),
LNAME VARCHAR2 (20),
ADDRESS VARCHAR2 (20),
SEX CHAR (1),
SALARY INTEGER,
SUPERSSN REFERENCES EMPLOYEE (SSN),
DNO REFERENCES DEPARTMENT (DNO));
```

NOTE: Once DEPARTMENT and EMPLOYEE tables are created we must alter department table to add foreign constraint MGRSSN using sql command

```
ALTER TABLE DEPARTMENT
ADD MGRSSN REFERENCES EMPLOYEE (SSN);
```

5. Inserting/Updating/Deleting Records in a Table,

```
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSECE01‘,‘JOHN‘,‘SCOTT‘,‘BANGALORE‘,‘M‘,  450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSCSE01‘,‘JAMES‘,‘SMITH‘,‘BANGALORE‘,‘M‘, 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSCSE02‘,‘HEARN‘,‘BAKER‘,‘BANGALORE‘,‘M‘, 700000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSCSE03‘,‘EDWARD‘,‘SCOTT‘,‘MYSORE‘,‘M‘, 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSCSE04‘,‘PAVAN‘,‘HEGDE‘,‘MANGALORE‘,‘M‘, 650000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSCSE05‘,‘GIRISH‘,‘MALYA‘,‘MYSORE‘,‘M‘, 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSCSE06‘,‘NEHA‘,‘SN‘,‘BANGALORE‘,‘F‘, 800000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSACC01‘,‘AHANA‘,‘K‘,‘MANGALORE‘,‘F‘, 350000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSACC02‘,‘SANTHOSH‘,‘KUMAR‘,‘MANGALORE‘,‘M‘, 300000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSISE01‘,‘VEENA‘,‘M‘,‘MYSORE‘,‘M‘, 600000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES (_RNSIT01‘,‘NAGESH‘,‘HR‘,‘BANGALORE‘,‘M‘, 500000);


INSERT INTO DEPARTMENT VALUES (_1‘,‘ACCOUNTS‘,‘01-JAN-
01‘,‘RNSACC02‘);
INSERT INTO DEPARTMENT VALUES (_2‘,‘IT‘,‘01-AUG-16‘,‘RNSIT01‘);
```

```
INSERT INTO DEPARTMENT VALUES (_3‘,‘ECE‘,‘01-JUN-08‘,‘RNSECE01‘);
INSERT INTO DEPARTMENT VALUES (_4‘,‘ISE‘,‘01-AUG-15‘,‘RNSISE01‘);
INSERT INTO DEPARTMENT VALUES (_5‘,‘CSE‘,‘01-JUN-02‘,‘RNSCSE05‘);
```

Update

```
UPDATE EMPLOYEE SET DNO=‘5‘, SUPERSSN=‘RNSCSE06‘ WHERE
SSN=‘RNSCSE05‘;
```

Delete entries of employee table where DNO =1;

```
DELETE FROM EMPLOYEE WHERE DNO=1;
```

6. COMMIT and ROLLBACK
   Before concluding this section on Data Manipulation Language commands there are two further commands, which are very useful. Changes made to the database by INSERT, UPDATE and DELETE commands are temporary until explicitly committed. This is performed by the command:

   ```
   COMMIT;
   ```

   On execution of this command all changes to the database made by you are made permanent and cannot be undone.
   - A COMMIT is automatically executed when you exit normally from SQL*Plus. However, it does no harm to occasionally issue a COMMIT command.
   - A COMMIT does not apply to any SELECT commands as there is nothing to commit.
   - A COMMIT does not apply to any DDL commands (eg CREATE TABLE, CREATE INDEX, etc). These are automatically committed and cannot be rolled back.
   - If you wished to rollback (ie undo) any changes made to the database since the last commit, you can issue the command:

   ```
   ROLLBACK;
   ```

   A group of related SQL commands that all have to complete successfully or otherwise be rolled back, is called a transaction. Part of your research for Outcome 3 includes investigating transaction processing and the implications of rollback and commit.

Experiment – 9

CO Mapped : CO1 and CO5

Perform the following: a. Altering a Table, Dropping/Truncating/Renaming Tables, Backing up / Restoring a Database.

Consider Dept table

DEPTNO	DNAME	LOC

Perform the following:

1. Rename the table dept as department
2. Add a new column PINCODE with not null constraints to the existing table DEPT
3. All constraints and views that reference the column are dropped automatically, along with the column.
4. Rename the column DNAME to DEPT_NAME in dept table
5. Change the data type of column loc as CHAR with size 10
6. Delete table

SOLUTION:

Create Table

SQL> CREATE TABLE DEPT(DEPTNO INTEGER, DNAME VARCHAR(10),LOC VARCHAR(4), PRIMARY KEY(DEPTNO));

1. Rename the table dept as department

   SQL> ALTER TABLE DEPT RENAME TO DEPARTMENT;
   Table altered.

2. Add a new column PINCODE with not null constraints to the existing table DEPT

   SQL> ALTER TABLE DEPARTMENT ADD(PINCODE NUMBER(6) NOT NULL);

   Table altered.

   SQL> DESC DEPARTMENT;

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER(38)
DNAME		VARCHAR2(10)
LOC		VARCHAR2(4)
PINCODE	NOT NULL	NUMBER(6)

3. All constraints and views that reference the column are dropped automatically, along with the column.

   SQL> ALTER TABLE DEPARTMENT DROP column LOC CASCADE CONSTRAINTS;

   Table altered.

   SQL> desc dept

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER(38)
DNAME		VARCHAR2(10)
PINCODE	NOT NULL	NUMBER(6)

4. Rename the column DNAME to DEPT_NAME in dept table

   SQL> ALTER TABLE DEPT RENAME COLUMN DNAME TO DEPT_NAME ;

   Table altered.
   SQL> DESC DEPARTMENT;

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER(38)
DEPT_NAME		VARCHAR2(10)
LOC		VARCHAR2(4)
PINCODE	NOT NULL	NUMBER(6)

5. Change the datatype of colunm loc as CHAR with size 10

   SQL> ALTER TABLE DEPARTMENT MODIFY LOC CHAR(10) ;
   Table altered.
   SQL> DESC DEPARTMENT;

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER(38)
DEPT_NAME		VARCHAR2(10)
LOC		CHAR(10)
PINCODE	NOT NULL	NUMBER(6)

6. Delete table

   SQL> DROP TABLE DEPARTMENT;
   Table dropped.

Experiment 10.

CO Mapped : CO1 and CO5

Aim : For a given set of relation schemes, create tables and perform the following Simple Queries, Simple Queries with Aggregate functions, Queries with Aggregate functions (group by and having clause), Queries involving- Date Functions, String Functions , Math Functions Join Queries- Inner Join, Outer Join Subqueries- With IN clause, With EXISTS clause.

For a given tables

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
	Research	5	333445555	1988-05-22
	Administration	4	987654321	1995-01-01
	Headquarters	1	888665555	1981-06-19

Create tables and perform the following

1. How the resulting salaries if every employee working on the 'Research' Departments is given a 10 percent raise.
2. Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department
3. Retrieve the name of each employee Controlled by department number 5 (use EXISTS operator).
4. Retrieve the name of each dept and number of employees working in each department which has at least 2 employees
5. Retrieve the name of employees who born in the year 1990's
6. Retrieve the name of employees and their dept name (using JOIN)

PROCEDURE:

```
SQL> CREATE TABLE DEPARTMENT(
        DNO VARCHAR2 (20) PRIMARY KEY,
        DNAME VARCHAR2 (20),
        MGRSTARTDATE DATE);

SQL> DESC DEPARTMENT;
        Name                          Null?    Type
        ----------------------------------------- -------- ----------------------------

        DNO                           NOT NULL VARCHAR2(20)
```

```
DNAME                                     VARCHAR2(20)
MGRSTARTDATE                              DATE

SQL> CREATE TABLE EMPLOYEE(
        SSN VARCHAR2 (20) PRIMARY KEY,
        FNAME VARCHAR2 (20),
        LNAME VARCHAR2 (20),
        ADDRESS VARCHAR2 (20),
        SEX CHAR (1),
        SALARY INTEGER,
        SUPERSSN REFERENCES EMPLOYEE (SSN),
        DNO REFERENCES DEPARTMENT (DNO));

SQL> DESC EMPLOYEE;
        Name                      Null?                 Type

        ----------------------------------------------------------------------
        SSN                       NOT NULL     VARCHAR2(20)
        FNAME                                  VARCHAR2(20)
        LNAME                                  VARCHAR2(20)
        ADDRESS                                VARCHAR2(20)
        SEX                                    CHAR(1)
        SALARY                                 NUMBER(38)
        SUPERSSN                               VARCHAR2(20)
        DNO                                    NUMBER(38)

SQL> ALTER TABLE DEPARTMENT
        2  ADD MGRSSN REFERENCES EMPLOYEE (SSN);

Table altered.

SQL> DESC DEPARTMENT;
        Name                      Null?                 Type
        ----------------------------------------- ---------- ---------------------------

        DNO                       NOT NULL     VARCHAR2(20)
        DNAME                                  VARCHAR2(20)
```

MGRSTARTDATE	DATE
MGRSSN	VARCHAR2(20)

```
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSECE01','JOHN','SCOTT','BANGALORE','M', 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSCSE01','JAMES','SMITH','BANGALORE','M', 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSCSE02','HEARN','BAKER','BANGALORE','M', 700000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSCSE03','EDWARD','SCOTT','MYSORE','M', 500000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSCSE04','PAVAN','HEGDE','MANGALORE','M', 650000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSCSE05','GIRISH','MALYA','MYSORE','M', 450000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSCSE06','NEHA','SN','BANGALORE','F', 800000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSACC01','AHANA','K','MANGALORE','F', 350000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSACC02','SANTHOSH','KUMAR','MANGALORE','M', 300000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSISE01','VEENA','M','MYSORE','M', 600000);
INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, SEX, SALARY)
VALUES ('RNSIT01','NAGESH','HR','BANGALORE','M', 500000);

INSERT INTO DEPARTMENT VALUES (1,'ACCOUNTS','01-JAN-
01','RNSACC02');
INSERT INTO DEPARTMENT VALUES (2,'IT','01-AUG-16','RNSIT01');
INSERT INTO DEPARTMENT VALUES (3,'ECE','01-JUN-08','RNSECE01');
INSERT INTO DEPARTMENT VALUES (4,'ISE','01-AUG-15','RNSISE01');
INSERT INTO DEPARTMENT VALUES (5,'CSE','01-JUN-02','RNSCSE05');
```

Note: update entries of employee table to fill missing fields SUPERSSN and DNO

```
UPDATE EMPLOYEE SET SUPERSSN=NULL, DNO='3' WHERE
SSN='RNSECE01';

UPDATE EMPLOYEE SET SUPERSSN='RNSCSE02', DNO='5' WHERE
SSN='RNSCSE01';
UPDATE EMPLOYEE SET SUPERSSN='RNSCSE03', DNO='5' WHERE
SSN='RNSCSE02';
UPDATE EMPLOYEE SET SUPERSSN='RNSCSE04', DNO='5' WHERE
SSN='RNSCSE03';
```

UPDATE EMPLOYEE SET DNO='5', SUPERSSN='RNSCSE05'
WHERE SSN='RNSCSE04'; UPDATE EMPLOYEE SET DNO='5',
SUPERSSN='RNSCSE06'WHERE SSN='RNSCSE05';
UPDATE EMPLOYEE SET DNO='5', SUPERSSN=NULL WHERE
SSN='RNSCSE06';
UPDATE EMPLOYEE SET DNO='1', SUPERSSN='RNSACC02' WHERE
SSN='RNSACC01';
UPDATE EMPLOYEE SET DNO='1', SUPERSSN=NULL WHERE
SSN='RNSACC02';
UPDATE EMPLOYEE SET DNO='4', SUPERSSN=NULL WHERE
SSN='RNSISE01';
UPDATE EMPLOYEE SET DNO='2', SUPERSSN=NULL WHERE
SSN='RNSIT01';

1. How the resulting salaries if every employee working on the 'Research'Departments is given a 10 percent raise.

```
SQL> SELECT E.FNAME,E.LNAME, 1.1*E.SALARY
  AS INCR_SAL2  FROM EMPLOYEE1
  E,DEPARTMENT D,EMPLOYEE1 W
  3  WHERE E.SSN=W.SSN
  4  AND E.DNO=D.DNUMBER
  5  AND D.DNAME='research';
```

FNAME	LNAME	SALARY	DNO	DNUMBER	INC_SAL
john	smith	30000	5	5	33000
franklin	wong	40000	5	5	44000
ramesh	narayan	780000	5	5	858000
joyce	english	25000	5	5	27500

2. Find the sum of the salaries of all employees of the 'Accounts' department, as wellas the maximum salary, the minimum salary, and the average salary in this department

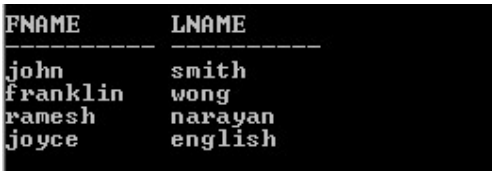
```
SQL> SELECT
SUM(E.SALARY),MAX(E.SALARY),MIN(E.SALARY),
AVG(E.SALARY)FROM EMPLOYEE1 E,DEPARTMENT
D WHERE E.DNO=D.DNUMBER AND
D.DNAME='RESEARCH';
```

SUM(E.SALARY)	MAX(E.SALARY)	MIN(E.SALARY)	AVG(E.SALARY)
875000	780000	25000	218750

3. Retrieve the name of each employee Controlled by department number 5 (useEXISTS operator).

```
SQL > SELECT E.FNAME , E.LNAME FROM EMPLOYEE E WHERE EXITS

1 (SELECT DNO. FROM EMPLOYEEE1 WHERE
```

2 E.DNO=5);



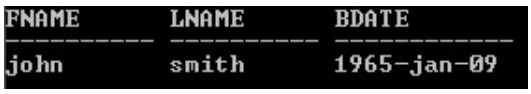
FNAME	LNAME
john	smith
franklin	wong
ramesh	narayan
joyce	english

4. Retrieve the name of each dept and number of employees working in eachdepartment which has at least 2 employees

```
SELECT DNAME, COUNT(*)
FROM EMPLOYEE E,
DEPARTMENT DWHERE
D.DNO=E.DNO
AND D.DNO IN
(SELECT E1.DNO
FROM EMPLOYEE
E1
GROUP BY E1.DNO
Having count(*)>2) ORDER BY DNO.
```

5. Retrieve the name of employees who born in the year 1990's

```
SELECT E.FNAME,E.LNAME,E.BDATE FROM EMPLOYEE1 E WHERE
BDATE LIKE '196%';
```



FNAME	LNAME	BDATE
john	smith	1965-jan-09

6. Retrieve the name of employees and their dept name (using JOIN)

```
SELECT E.FNAME, E.LNAME, DNAME
FROM EMPLOYEE E NATURAL JOIN DEPARTMENT D ON
E,DNO=D.DNO;
```