**2. CREATION OF DB COLLECTION AND DISPLAY**

**AIM :** - To Create a database Collection and display listof database and collection.

**Procedure**

Create database : MongoDB USe DATABASE\_NAME is used to create database. The Command will Create a new database if it doesnt exist. otherwise it will retesn the existing

data base

• Syntax: Use DATABASE \_NAME

• Examble: > use Student

Switched to db Student

ii) To Create Collections:: MongoDB db. createCollection (namer options)

isused to Create collection.In the command, name is the name of Collection to be created. Options is a document and is used to specify configuration of collection

• Syntax: db. CreateCollection (name. options)

• Example: db. CreateCollection (" mycollection")

iii) To display the list of datas : It use want to check our databases list. use the Command.

Show db S

• syntar: Show dbs

test> **use student**

switched to db student

student> db.createCollection ("My Collection")

{ ok: 1 }

student> show dbs

admin 40.00 KiB

config 108.00 KiB

local 72.00 KiB

student 8.00 KiB

student> show collections

My Collection

student>

**3.INSERTION AND UPDATION OF DOCS AND COLLECTION**

**Insert many**

student> db.student.insertMany([{Firstname:"Jack",Lastname:"Son",Reg:"4012008"},{Firstname:"Sharma",Lastname:"gill"}])

{

acknowledged: true,

insertedIds: {

'0': ObjectId("643ead539d855c7558a14418"),

'1': ObjectId("643ead539d855c7558a14419")

}

}

**Update many**

student> db.student.updateMany({Firstname: "Jack"}, {$set:{Lastname: "Jin"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

student>

**4.MONGODB COMMAND TO QUERY THE DOCUMENT**

**Find() method -** To Query data fom Mongodb collection we need to use mongodb find() method

**Example:**db.student.find()

**O/p**

[

{ \_id: ObjectId("643eab339d855c7558a14417"), Name: 'Jack' },

{

\_id: ObjectId("643ead539d855c7558a14418"),

Firstname: 'Jack',

Lastname: 'Jin',

Reg: '4012008'

},

{

\_id: ObjectId("643ead539d855c7558a14419"),

Firstname: 'Sharma',

Lastname: 'gill'

}]

**Pretty() -**  to display the results in a formatted way to use pretty() meythod.

**Example** > db.student.find().pretty()

O/p

[

{ \_id: ObjectId("643eab339d855c7558a14417"), Name: 'Jack' },

{

\_id: ObjectId("643ead539d855c7558a14418"),

Firstname: 'Jack',

Lastname: 'Jin',

Reg: '4012008'

},

{

\_id: ObjectId("643ead539d855c7558a14419"),

Firstname: 'Sharma',

Lastname: 'gill'

}

]

**Findone() method:**

Returns only one method

Example:db.student.findone{Firstname:”jack”})

**4.AND**

Query: db.student.find({$and:[{“by”:”Hello”},{“title”:”MongoDB Overview”}]}).Pretty()

Example:db.student.find({$and:[{“by”:”Hello”},{“title”:”MongoDB Overview”}]}).Pretty()

{

“\_id”:ObjectId(“5dd4e2cc0821d344”),

“title”:”MongoDB Overview”,

“description”:”NoSQL Database”,

“by”:”Hello”,

“tags”:[

“mongodb”,

“database”,

]

“likes”:100

}

**5.OR**

Query: db.student.find({$or:[{“by”:”Hello”},{“title”:”MongoDB Overview”}]}).Pretty()

Example: db.student.find({$or:[{“by”:”Hello”},{“title”:”MongoDB Overview”}]}).Pretty()

**6.AND and OR together**

Example: db.student.find({“likes”.{$gt:10},$or:[{“by”:”Hello”},

{“title”:”MongoDB Overview”}]}.Pretty()

**7.NOR**

Example: db.empDetails.find({$nor:[{first\_name:”Radhika”},{last\_name:”christoper”}]}).Pretty()

**8.NOT**

Example: db.empDetails.find({“Age”:{$not:{$not:{$gt:25}}}).Pretty()

**5.LIMITING RECORD AND SORTING**

**Limit() method**

Query: db.mycol.find({“title”:1,\_id:0}).limit(2)

Example: db.mycol.find({“title”:1,\_id:0}).limit(2)

{“title”: “Mongodb Overview”}

{“title”: “NoSQL Overview”}

**Skip() method**

Query: db.mycol.find({“title”:1,\_id:0}) .limit(1).skip(1)

Example: db.mycol.find({“title”:1,\_id:0}).limit().skip(1)

{“title”: “Mongodb Overview”}

**Sort() method**

Query: db.mycol.find({“title”:1,\_id:0}).sort({title:-1})

Example: db.mycol.find().sort({key:1})

{“title”: “Mongodb Overview”}

{“title”: “Mongodb page”}

**6.AGGREGATION AND INDEXING**

**Aim :** to implement indexing and aggregation commands in mongo db

**Procedure :**

**1.create under()method**

Syntax: db.COLLECTION\_NAME.create index({key:1})

Query : db.mycol.create index({“title”: ”des”: 1})

**2. dropindex() method**

Syntax : Db.COLLECTION\_NAME.dropindex({key1})

Query : Db.mycol\_drop index ({“title” : 1})

**3.get index() method**

Syntax : Db.COLLECTION\_NAME.get index()

Query : Db.mycol.get index()

**4.aggregate() method**:

db

Syntax : Db.COLLECTION\_NAME.aggregate

Query : Db.mycol.aggregate((8 group: {-id: “$by uses”num\_tutorial{$sum : 1}}}])

**7.FILE MANAGEMENT TASK IN HADOOP**

AIM: To execeite file management Commands Hadoop

Procedure: -

- Various HaDoop Commands -aSk

in Hadoop

1. Create a directory ni HDfs at the given path –

Syntax : Hadoop fs – mkdir/Path |directory name .

Example: hadoop f s - mkdir/new Dataflair

2 List the content of the directory-

Syntax: hadoop fs – ls/Path

Example: Hadoop fs ls/Path

fs. -Is | newpalaflair.

3 Upload and Download a file in HDFS.

Upload syntax: hadoop fs - Put<Local src> (destination)

Example : hadoop fs - Put /home/ file.txt/uses input

Download syntax: hadoop fs -get<src> (local destinations )

Example: hadoop fs - get /user /output-// home /Hadoop-tp/

4.Set theContent of a file

syntax: hadoop fs - Cat /Path

Example: hadoop fs - cat/user / output /outfile

5. Copy a file from sousce to destination –

syntax: hadoop fs -cp<src> <destinations>

Example: hadoop fs - cp /grekst geeks -copied a file

6. Copy from local To local file system to HDFS.

syntax : hadoop fs - copyfromlocal <locar src> <hadoop destinations>

Example: hadoopfs -copy fomLocal/Desktop /uses .txt /geeKs

S y n t a x :Hadoop fs -copyTolocal <Hadoop src> <local destinations>

Example: Hadoop fs – CopyTolocal/geeks /DesKtop /Helo

7) Move file from Source to destination: - mv <Source> <destinations

syntax : Hadoop fs mv <Source> <destinations>

Example: hadoop fs –mv geeks /myfill. txt/geeks. copied

8.> Remove a file of directory in HoFs -

Syntax: Hadoop fs- rm<files

Example: Hadoop fs -rm -r /user/test /sample. txt

9.) Move From local to HDFS/to local from HDFS

syntax:hadoop fs –movefromlocalK<src><dest>

Example: hadoop fs move fromlocal.../Desktop/Cet.txt-/geeks.

syntax: hadwop fs move tolocal <Sre> < local dest>

10. Display last few lines of a file -

fs -tail

syntax: Hadoop -f<files>

Exampie: hadoop fs –tail/in/appendfile

**12.WORDCOUNT PROGRAM**

**PROCEDURE**

C:\Users\jeber>hdfs namenode -format

C:\hadoop-2.8.0\sbin>start-all.cmd

C:\hadoop-2.8.0\sbin>jps

3472 DataNode

21896 Jps

7688 ResourceManager

26492 NameNode

C:\>hadoop fs -mkdir /input\_dir

C:\>hadoop fs -put E:\input\_file.txt /input\_dir

C:\>hadoop fs -ls /input\_dir/

Found 1 items

-rw-r--r-- 1 jeber supergroup 283 2023-01-27 21:19 /input\_dir/input\_file.txt

C:\>hadoop dfs -cat /input\_dir/input\_file.txt

C:\>hadoop jar C:\hadoop-2.8.0\share\hadoop\mapreduce\hadoop-mapreduce-examples-2.8.0.jar wordcount /input\_dir/input\_file.txt /output\_dir

C:\>hadoop dfs -cat /output1\_dir/\*

DEPRECATED: Use of this script to execute hdfs command is deprecated.

Instead use the hdfs command for it.

1987 1

2001 1

Chennai, 1

College 1

Engineering 1

India. 1

Institute 1

It 1

Jeppiaar 1

Nadu, 1

Sathyabama 3

Science 1

Tamil 1

Technology, 1

University, 1

a 1

and 2

as 1

at 1

be 2

by 1

deemed 2

formerly 1

founded 1

in 2

is 1

its 1

late 1

of 1

received 1

situated 1

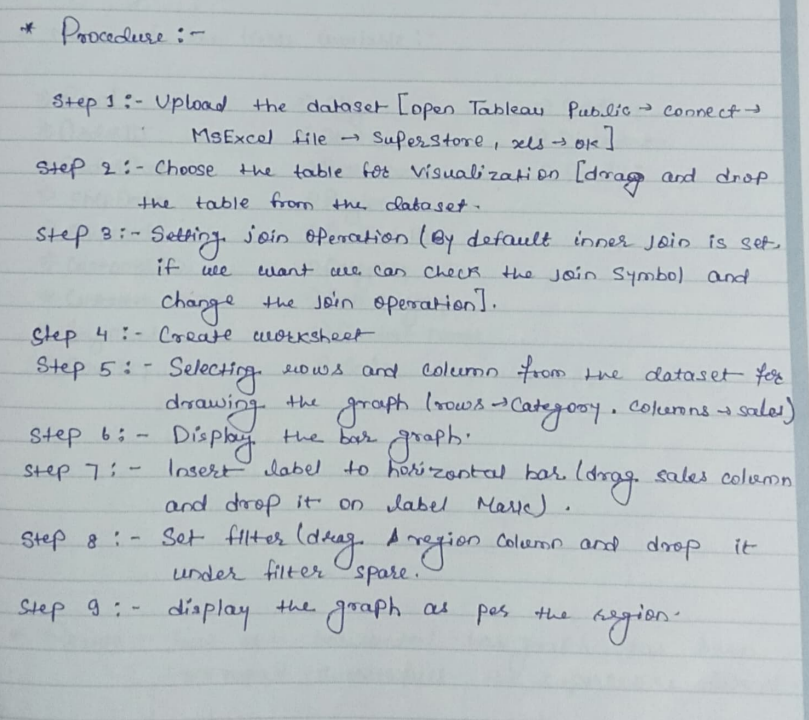
status 1

the 1

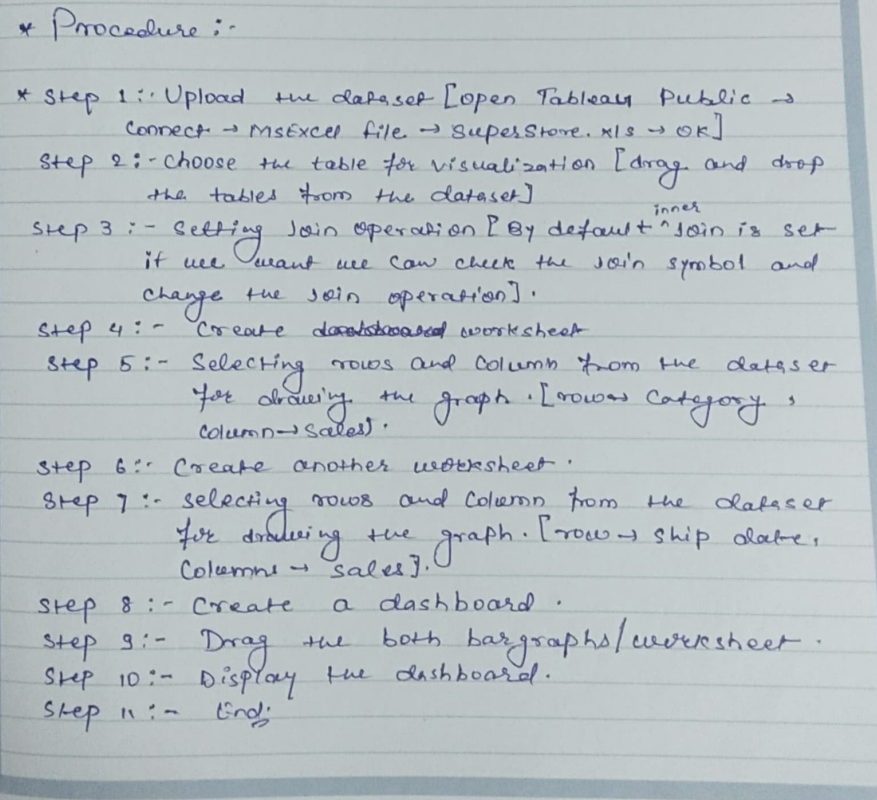
to 2

university 1

university, 1 was 1

**8. HORIZONTAL GRAPH BAR TOTAL SALES / 9.DATAWISE REPORT**

**10.DASHBOARD OF REGION DATAWISE**

****