**Experiment no – 1**

1.Practice DDL & DML commands execute basic utilities used to interact with Oracle DBMS /mysql

**Program:-**

CREATE table person(person\_id int primary key,FirstName varchar(50),LastName varchar(50));

ALTER table person ADD Email varchar(60); ALTER table person

ADD PhoneNumber int; select \* from person;

ALTER TABLE person DROP COLUMN PhoneNumber; ALTER TABLE person

MODIFY Lastname varchar(150); rename person to staff; ALTER TABLE STAFF DROP COLUMN Email; SELECT \* FROM

staff; truncate table staff;

SELECT \* FROM staff;create table Teacher(teacher\_id int primary key,teacher\_name varchar(50),teacher\_age int,teacher\_salary int);

INSERT into Teacher values(1,'pratiksha',20,80000);

INSERT into Teacher values(2,'tiksha',21,77000);

INSERT into Teacher values(3,'shravni',20,90000);

INSERT into Teacher values(4,'prachi',20,100000);

INSERT into Teacher values(5,'anjali',20,80000);

INSERT into Teacher values(6,'diksha',20,50000);

INSERT into Teacher values(7,'harshad',20,50000);

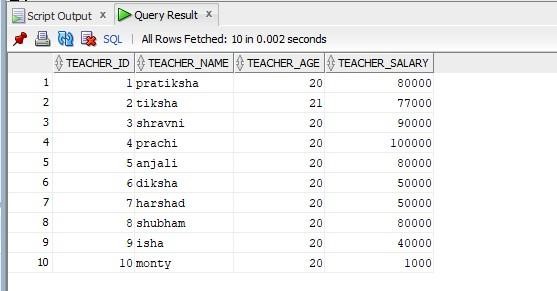
INSERT into Teacher values(8,'shubham',20,80000);

INSERT into Teacher values(9,'isha',20,40000);

INSERT into Teacher values(10,'monty',20,1000);

SELECT \* FROM Teacher;

**Output:-**



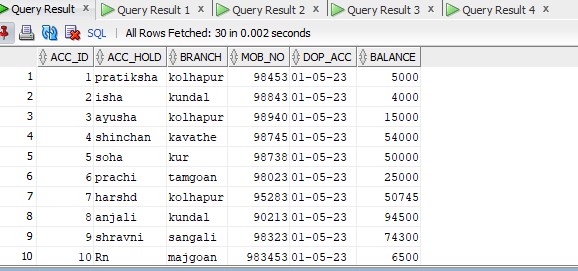
**Experiment No. :-2**

**Title** :- Design and Implement the Fragmentation schema and the Replication scheme for the social networking wesites /online e-shopping/e-learning wesites .

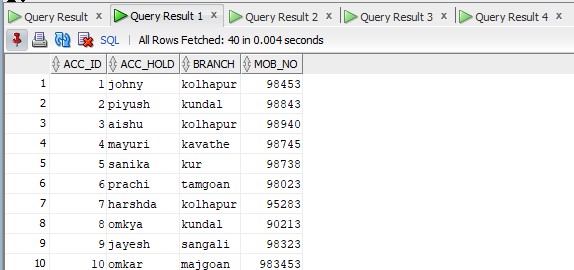
**SQL Queries :-**

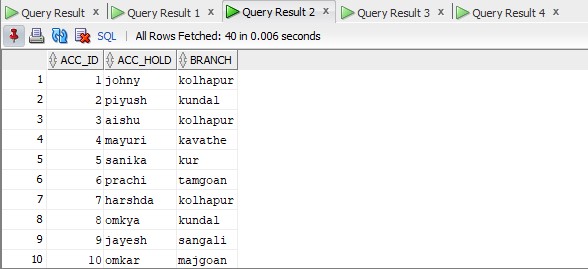
CREATE TABLE BOB (Acc\_ID int,Acc\_hold varchar(100),Branch varchar(100),Mob\_no INT,DOP\_acc DATE,Balance int); CREATE TABLE Alice (Acc\_ID int,Acc\_hold varchar(100),Branch varchar(100),Mob\_no INT,DOP\_acc DATE,Balance int); insert INTO Alice VALUES (1,'shubham','kolhapur',98453,'1-5-2023',5000); insert INTO Alice VALUES (2,'isha','kundal',98843,'1-5-2023',4000); insert INTO Alice VALUES (3,'vishal','kolhapur',98940,'1-5-2023',15000); insert INTO Alice VALUES (4,'shinchan','kavathe',98745,'1-5-2023',54000); insert INTO Alice VALUES (5,'soha','kur',98738,'1-5-2023',50000); insert INTO Alice VALUES (6,'soham','tamgoan',98023,'1-5-2023',25000); insert INTO Alice VALUES (7,'harshd','kolhapur',95283,'1-5-2023',50745); insert INTO Alice VALUES (8,'anjali','kundal',90213,'1-5-2023',94500); insert INTO Alice VALUES (9,'shravni','sangali',98323,'1-5-2023',74300); insert INTO Alice VALUES (10,'Rn','majgoan',983453,'1-5-2023',6500); select \*from Alice; create table Alice2 as select Acc\_ID,Acc\_hold,Branch,Mob\_no from BOB ; select \*from Alice2; create table Alice3 as select Acc\_ID,Acc\_hold,Branch from BOB ; select \*from Alice3; create table Alice4 as select\*from Alice where Branch='kolhapur'; select \*from Alice4; create table Alice5 as select\*from Alice where Balance>=15000; select \*from Alice5;

**Output :-**

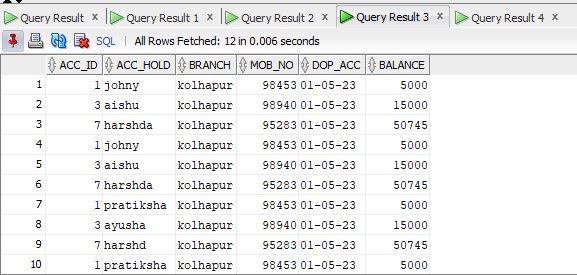
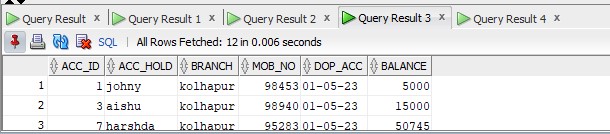


**1.Vertical Fragmentation :-**





**2. Horizontal Fragmentation :-**



**Experiment 3**

**Title: Implementation of 2 Phase Commit protocol for distributed databases.**

**Server:**

//Two Phase Commit Protocol SERVER package twophase;

import java.io.\*; import java.net.\*; import java.util.\*;

public class Server { boolean closed = false, inputFromAll = false; List<ClientThread> thread;

List<String> data;

Server() { thread = new ArrayList<ClientThread>(); data = new ArrayList<String>();

}

public static void main(String args[]) { Socket clientSocket

= null; ServerSocket serverSocket = null; int

port\_number = 1111;

Server server = new Server(); try { serverSocket = new ServerSocket(port\_number);

} catch (IOException e) { System.out.println(e);

}

while (!server.closed) { try { clientSocket = serverSocket.accept();

ClientThread clientThread = new ClientThread(server, clientSocket); (server.thread).add(clientThread); System.out.println("\nNow Total clients are : " + (server.thread).size()); (server.data).add("NOT\_SENT"); clientThread.start();

} catch (IOException e) {

}

}

try {

serverSocket.close();

} catch (Exception e1) {

}

}

}

class ClientThread extends Thread { DataInputStream is = null;

String line;

String destClient = ""; String name;

PrintStream os = null; Socket clientSocket = null; String clientIdentity; Server server;

public ClientThread(Server server, Socket clientSocket) { this.clientSocket = clientSocket; this.server = server;

}

@SuppressWarnings("deprecation") public void run() { try { is = new DataInputStream(clientSocket.getInputStream()); os = new

PrintStream(clientSocket.getOutputStream()); os.println("Enter your name."); name = is.readLine(); clientIdentity = name; os.println("Welcome " + name + " to this 2 Phase Application.\nYou will receive a vote Request now..."); os.println("VOTE\_REQUEST\nPlease enter COMMIT or ABORT to proceed : "); for (int i = 0; i <

(server.thread).size(); i++) {

if ((server.thread).get(i) != this) {

((server.thread).get(i)).os.println("---A new user " + name + " entered the Appilcation---");

}

}

while (true) { line = is.readLine();

if (line.equalsIgnoreCase("ABORT")) { System.out.println("\nFrom '" + clientIdentity

+ "' : ABORT\n\nSince aborted we will not wait for inputs from other clients."); System.out.println("\nAborted .... ");

for (int i = 0; i < (server.thread).size(); i++) {

((server.thread).get(i)).os.println("GLOBAL\_ABORT"); ((server.thread).get(i)).os.close(); ((server.thread).get(i)).is.close();

}

break;

}

if (line.equalsIgnoreCase("COMMIT")) { System.out.println("\nFrom '" + clientIdentity

+ "' : COMMIT"); if ((server.thread).contains(this)) {

(server.data).set((server.thread).indexOf(this), "COMMIT"); for (int j = 0; j <

(server.data).size(); j++) { if (!(((server.data).get(j)).equalsIgnoreCase("NOT\_SENT"))) { server.inputFromAll = true; continue;

} else { server.inputFromAll = false;

System.out.println("\nWaiting for inputs from other clients."); break;

}

}

if (server.inputFromAll) { System.out.println("\n\nCommited

.......................................................... ");

for (int i = 0; i < (server.thread).size(); i++) {

((server.thread).get(i)).os.println("GLOBAL\_COMMIT"); ((server.thread).get(i)).os.close(); ((server.thread).get(i)).is.close();

}

break;

}

} // if thread.contains

} // commit

} // while server.closed = true; clientSocket.close();

} catch (IOException e) {

}

}

}// end class thread

/\*

\*

\*

* Coordinator Cohorts
* QUERY TO COMMIT
* >
* VOTE YES/NO prepare/abort
* <
* commit/abort COMMIT/ROLLBACK
* >
* ACKNOWLEDGMENT commit/abort
* <
* end

\*

\*

* Two Phases :

\*

* 1.Prepare and Vote Phase
* 2. Commit or Abort Phase

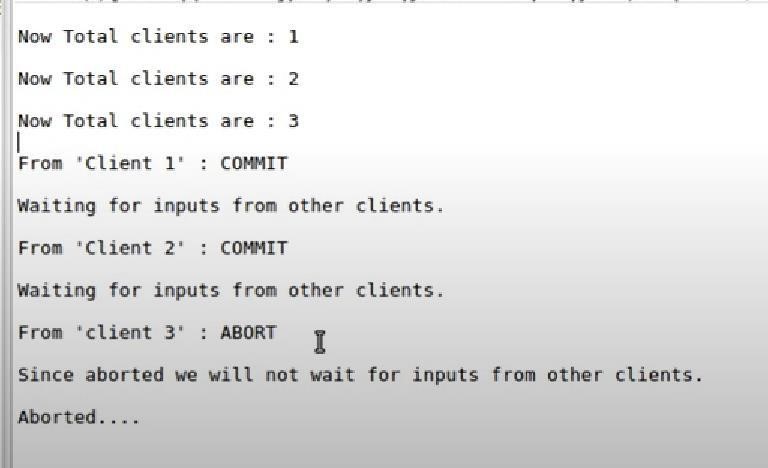
\*

* "Either All Commit Or All RollBack."

\*

\*

\*/



# Client.java

//Two Phase Commit Protocol CLIENT package twophase;

import java.io.\*; import java.net.\*;

public class Client implements Runnable { static Socket clientSocket = null;

static PrintStream os = null; static DataInputStream is

= null;

static BufferedReader inputLine = null; static boolean closed = false;

public static void main(String[] args) { int port\_number =

1111;

String host = "localhost"; try { clientSocket = new Socket(host, port\_number);

inputLine = new BufferedReader(new InputStreamReader(System.in)); os = new PrintStream(clientSocket.getOutputStream()); is = new DataInputStream(clientSocket.getInputStream());

} catch (Exception e) {

System.out.println("Exception occurred : " + e.getMessage());

}

if (clientSocket != null && os != null && is != null) { try { new Thread(new Client()).start(); while (!closed) {

os.println(inputLine.readLine());

}

os.close();

is.close(); clientSocket.close();

} catch (IOException e) { System.err.println("IOException: " + e);

}

}

}

@SuppressWarnings("deprecation") public void run() {

String responseLine; try { while ((responseLine = is.readLine()) != null) { System.out.println("\n" +

responseLine);

if (responseLine.equalsIgnoreCase("GLOBAL\_COMMIT") == true

|| responseLine.equalsIgnoreCase("GLOBAL\_ABORT") == true) { break;

}

}

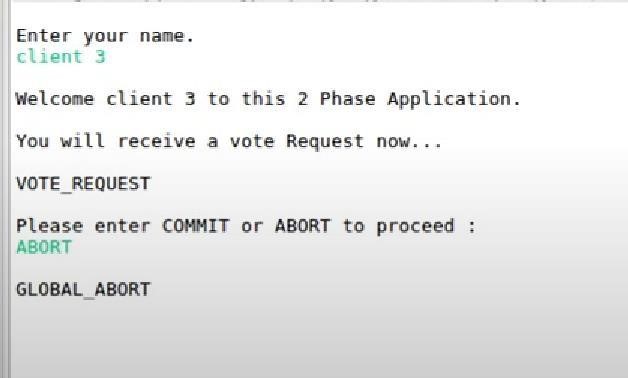
closed = true;

} catch (IOException e) { System.err.println("IOException: " + e);

}

}

} // end client



# Experiment No – 4

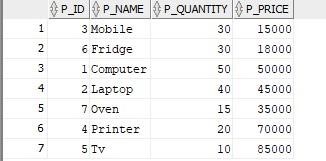
**Partition by range:**  create table product\_range(p\_id number,p\_name varchar(30),p\_quantity number,p\_price number) partition by range(p\_price)

(partition p1 values less than(20001), partition p2 values less than(50001), partition p3 values less than(100000)

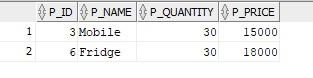
);

insert into product\_range values(1,'Computer',50,50000); insert into product\_range values(2,'Laptop',40,45000); insert into product\_range values(3,'Mobile',30,15000); insert into product\_range values(4,'Printer',20,70000); insert into product\_range values(5,'Tv',10,85000); insert into product\_range values(6,'Fridge',30,18000); insert into product\_range values(7,'Oven',15,35000);

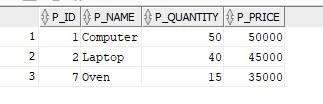
select \* from product\_range;

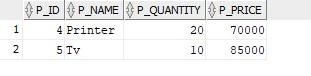


select \* from product\_range partition(p1);



select \* from product\_range partition(p2);

 select \* from product\_range partition(p3);



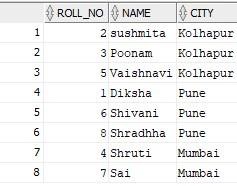
**Partition by list:**

create table student\_list(Roll\_no number,Name varchar(30),city varchar(20))

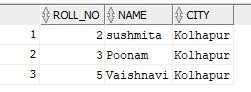
partition by list(city) (partition l1 values('Kolhapur'), partition l2 values('Pune'), partition l3 values('Mumbai'));

insert into student\_list values(1,'Diksha','Pune'); insert into student\_list values(2,'sushmita','Kolhapur'); insert into student\_list values(3,'Poonam','Kolhapur'); insert into student\_list values(4,'Shruti','Mumbai'); insert into student\_list values(5,'Vaishnavi','Kolhapur'); insert into student\_list values(6,'Shivani','Pune'); insert into student\_list values(7,'Sai','Mumbai'); insert into student\_list

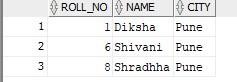
values(8,'Shradhha','Pune'); select \* from student\_list;



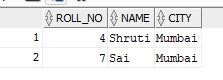
select \* from student\_list partition(l1);



select \* from student\_list partition(l2);



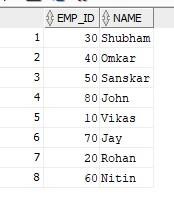
select \* from student\_list partition(l3);



**Partition by hash:**

create table emp\_hash(emp\_id number,name varchar(20)) partition by hash(emp\_id) partitions 4; insert into emp\_hash values(10,'Vikas'); insert into emp\_hash values(20,'Rohan'); insert into emp\_hash values(30,'Shubham'); insert into emp\_hash values(40,'Omkar'); insert into emp\_hash values(50,'Sanskar'); insert into emp\_hash values(60,'Nitin'); insert into emp\_hash values(70,'Jay'); insert into emp\_hash

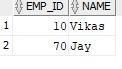
values(80,'John'); select \* from emp\_hash;



select \* from emp\_hash partition(SYS\_P948);



select \* from emp\_hash partition(SYS\_P949);



select \* from emp\_hash partition(SYS\_P950);



select \* from emp\_hash partition(SYS\_P951);



## Experiment No.5

**Q. Implementation of Oracle Synonyms and Sequence.** create SEQUENCE seq\_student

start with 1 increment by 2 nocycle nocache;



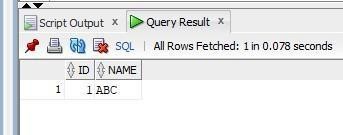
create table student\_info(id number, name varchar(50))



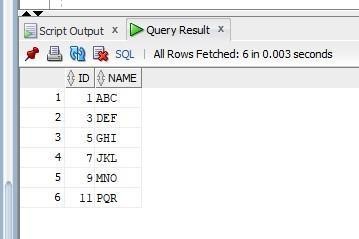
insert into student\_info Values(seq\_student.nextval,'ABC');



SELECT \* FROM student\_info;



insert into student\_info Values(seq\_student.nextval,'DEF'); insert into student\_info Values(seq\_student.nextval,'GHI'); insert into student\_info Values(seq\_student.nextval,'JKL'); insert into student\_info Values(seq\_student.nextval,'MNO'); insert into student\_info Values(seq\_student.nextval,'PQR'); SELECT \* FROM student\_info;



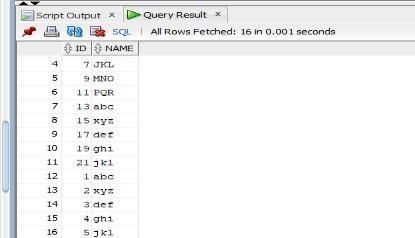
create SEQUENCE seq\_value start with 1 minvalue 1 maxvalue 5 increment by 1 cycle cache 4;



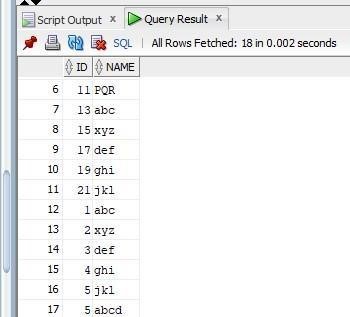
insert into student\_info Values(seq\_student.nextval,'abc');



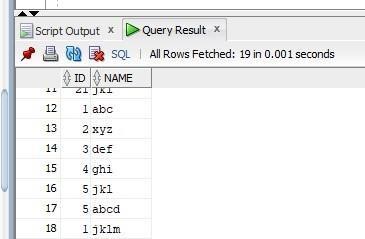
insert into student\_info Values(seq\_value.nextval,'abc'); insert into student\_info Values(seq\_value.nextval,'xyz'); insert into student\_info Values(seq\_value.nextval,'def'); insert into student\_info Values(seq\_value.nextval,'ghi'); insert into student\_info Values(seq\_value.nextval,'jkl');



insert into student\_info Values(seq\_value.currval,'abcd');



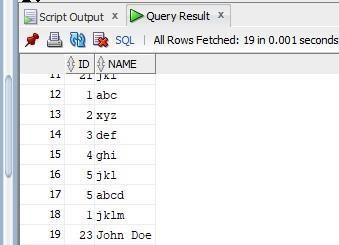
insert into student\_info Values(seq\_value.nextval,'jklm');



CREATE SYNONYM student\_id\_seq FOR seq\_student;



INSERT INTO student\_info (id, name) VALUES (student\_id\_seq.nextval, 'John Doe'); SELECT \* FROM student\_info;



# Experiment No – 6

**1. Functions :**

create or replace function square\_area(side number) return number is pi constant number(7,3) := 3.141; area number(7,3); begin area := side\*side; return area; end;

/



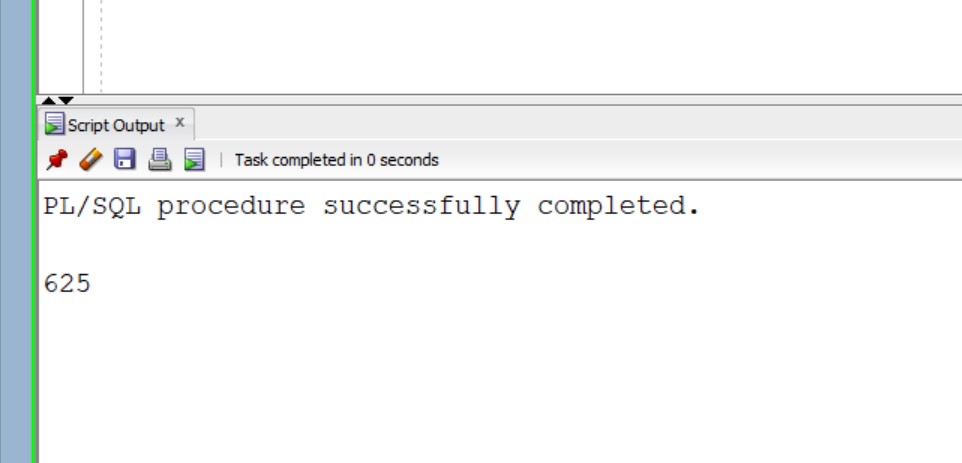
set SERVEROUTPUT ON;

begin

DBMS\_OUTPUT.PUT\_LINE(SQUARE\_AREA(25));

end;

/



**2.Procedures :**

CREATE OR REPLACE PROCEDURE P1

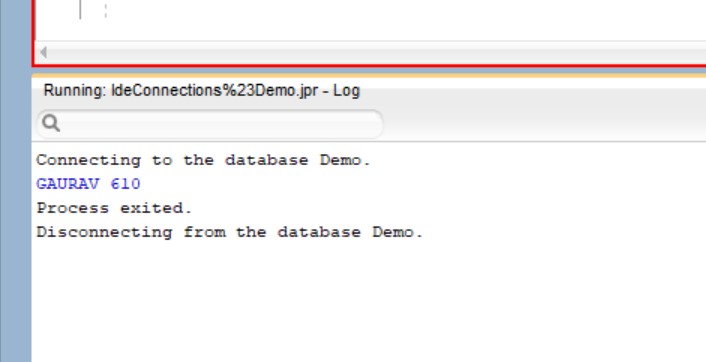
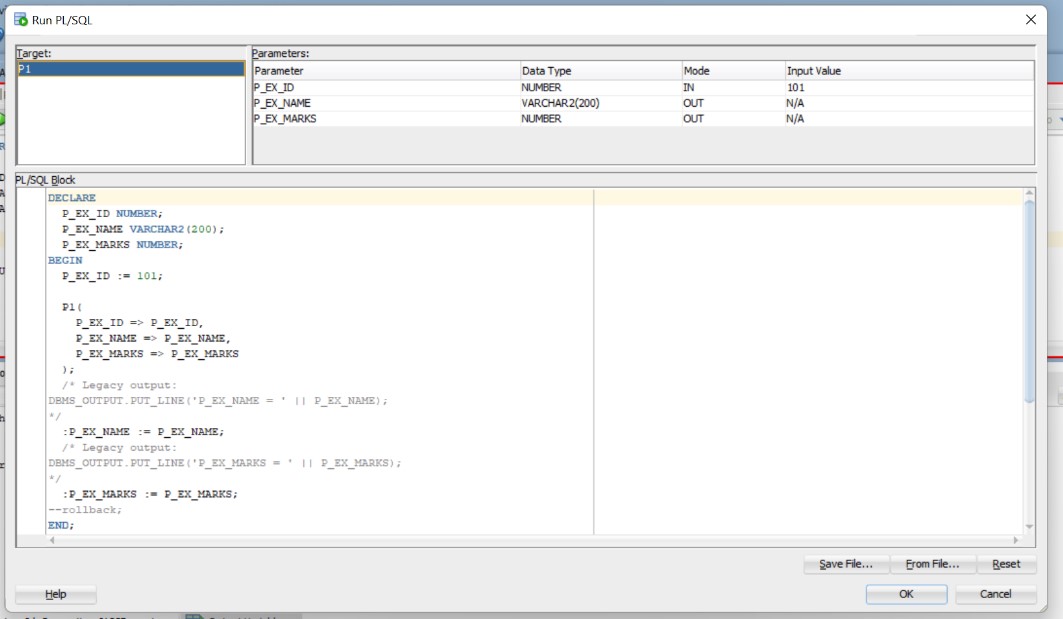
(P\_EX\_ID IN NUMBER , P\_EX\_NAME OUT VARCHAR2 , P\_EX\_MARKS OUT NUMBER ) AS BEGIN

SELECT EX\_NAME,EX\_MARKS INTO P\_EX\_NAME,P\_EX\_MARKS FROM EXAM WHERE EX\_ID = P\_EX\_ID;

DBMS\_OUTPUT.PUT\_LINE(P\_EX\_NAME || ' ' || P\_EX\_MARKS); END P1;

**3.**

**Cursors :**



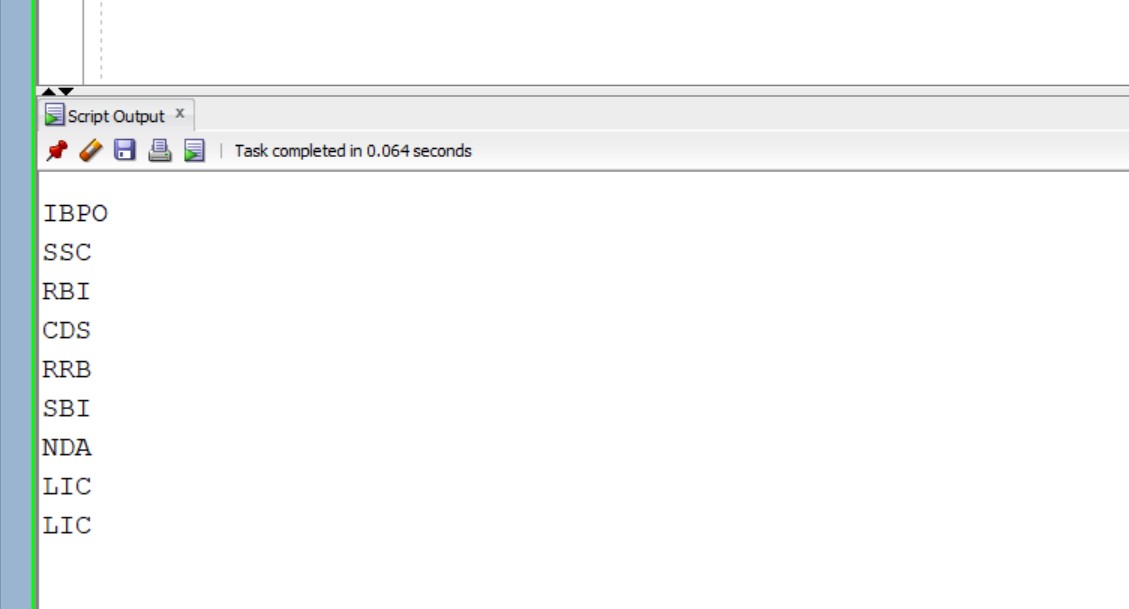
set SERVEROUTPUT ON; DECLARE v\_name varchar(30); cursor emp\_cursor is select ex\_name from exam where ex\_id<110; begin open emp\_cursor;

loop fetch emp\_cursor into v\_name;

DBMS\_OUTPUT.PUT\_LINE(v\_name);

exit when emp\_cursor%notfound;

end loop; close emp\_cursor; end;



1. **Triggers :**

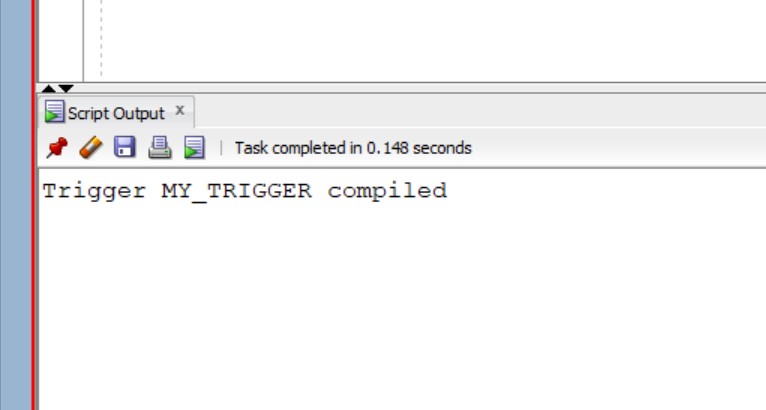
CREATE OR REPLACE TRIGGER MY\_TRIGGER

BEFORE UPDATE ON EXAM

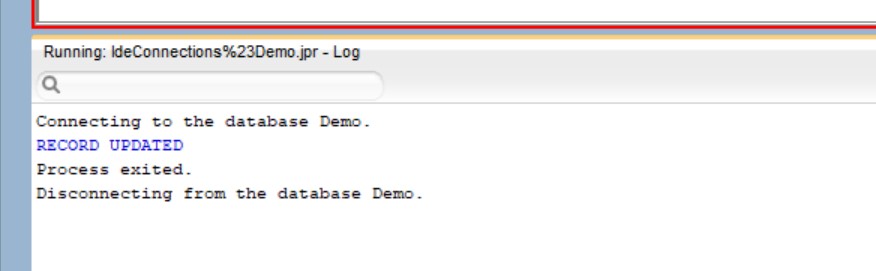
BEGIN

DBMS\_OUTPUT.PUT\_LINE('RECORD UPDATED');

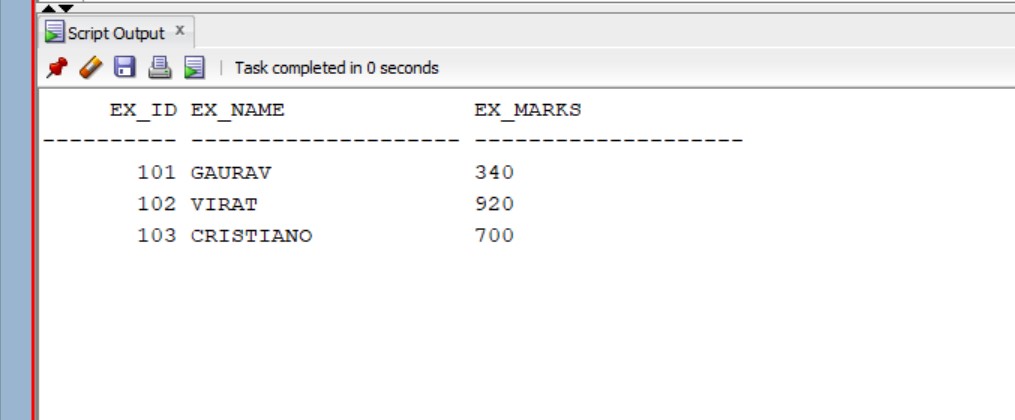
END;



UPDATE EXAM SET EX\_MARKS = '340' WHERE EX\_ID = '101';



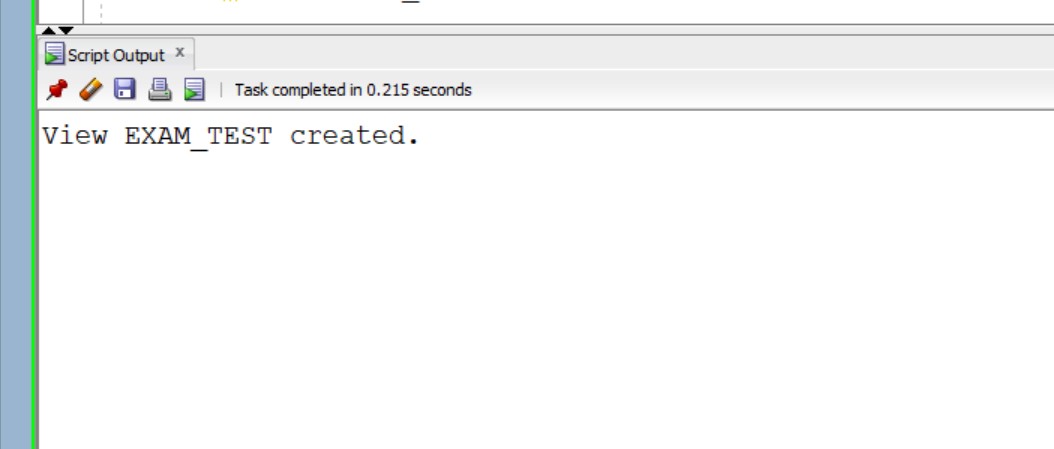
select \* from exam;



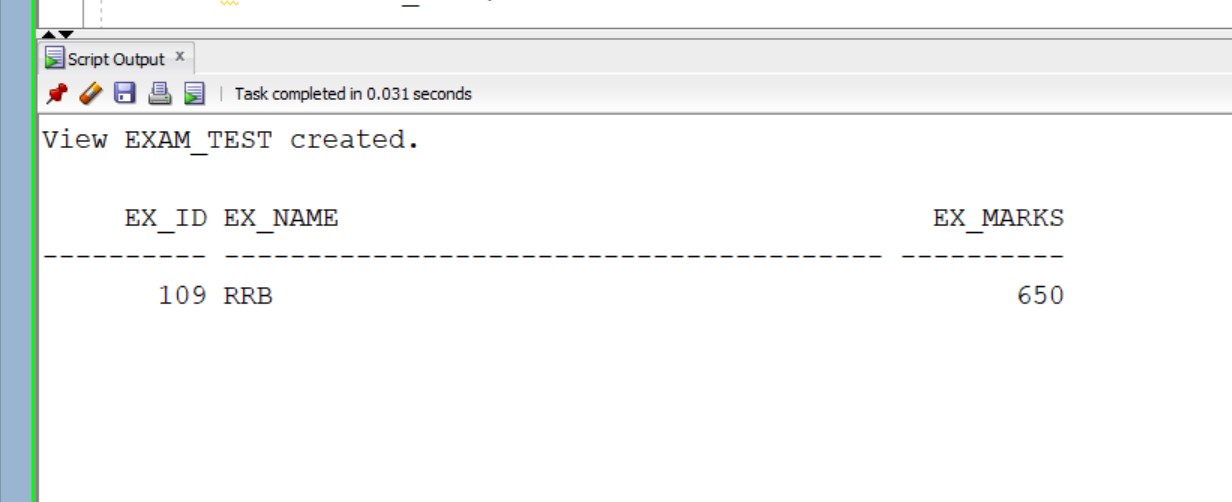
1. **Views :**

**a. Creating views :**

create view exam\_test as select ex\_id,ex\_name,ex\_marks from exam where ex\_id = 109 with check option constraint exam\_test\_cnst;

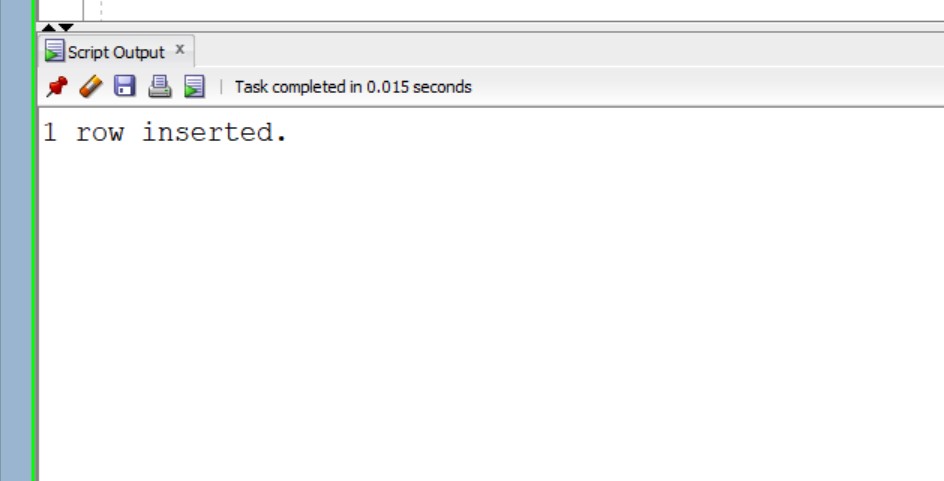


select \* from exam\_test;



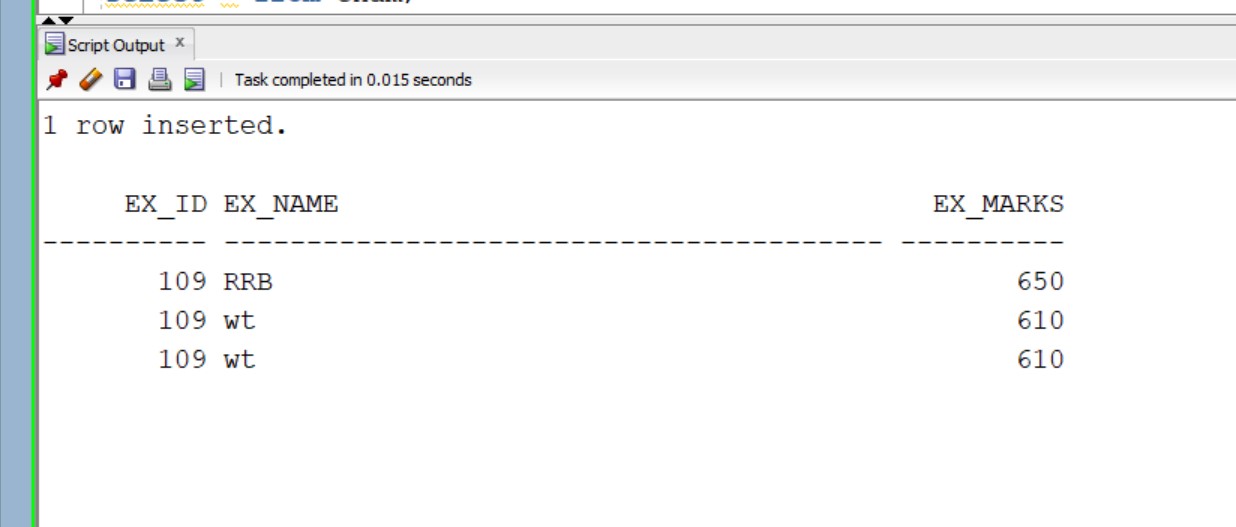
insert into exam\_test(ex\_id,ex\_name,ex\_marks) values(109,'wt',610);

select \* from exam\_test;



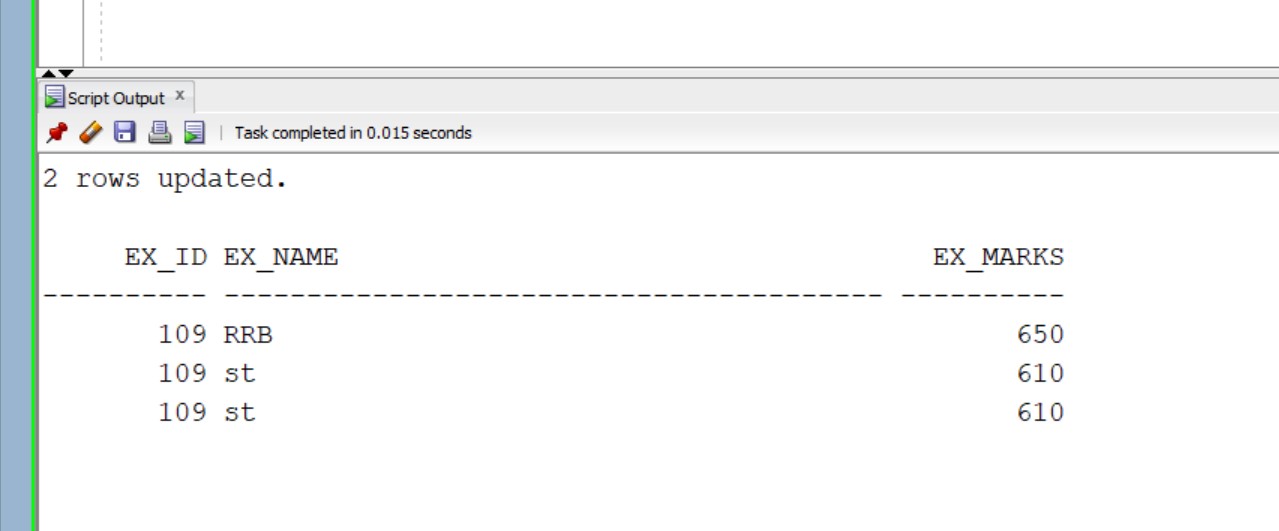
**b.**

**Updating views :**



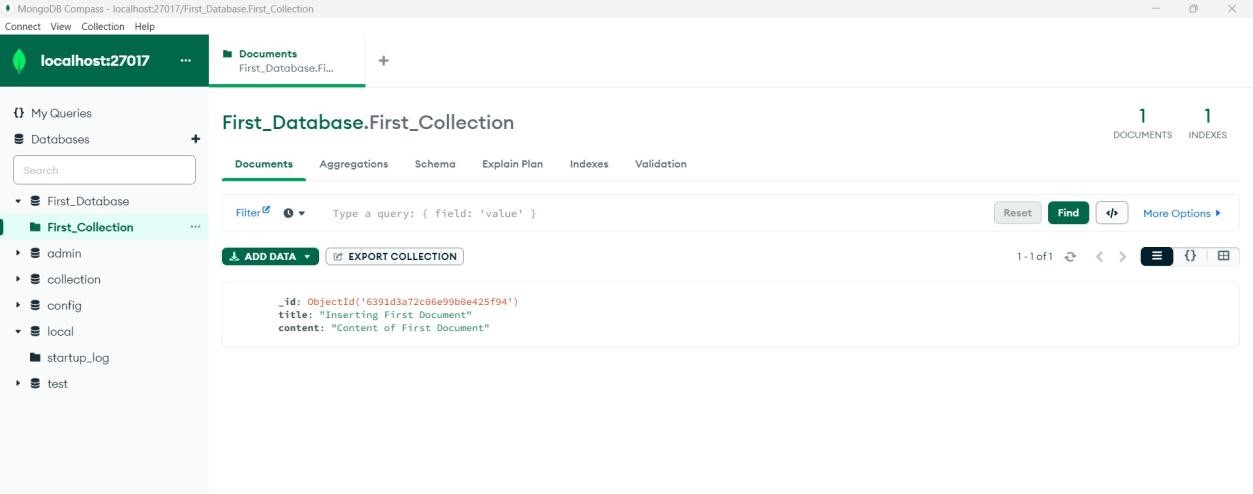
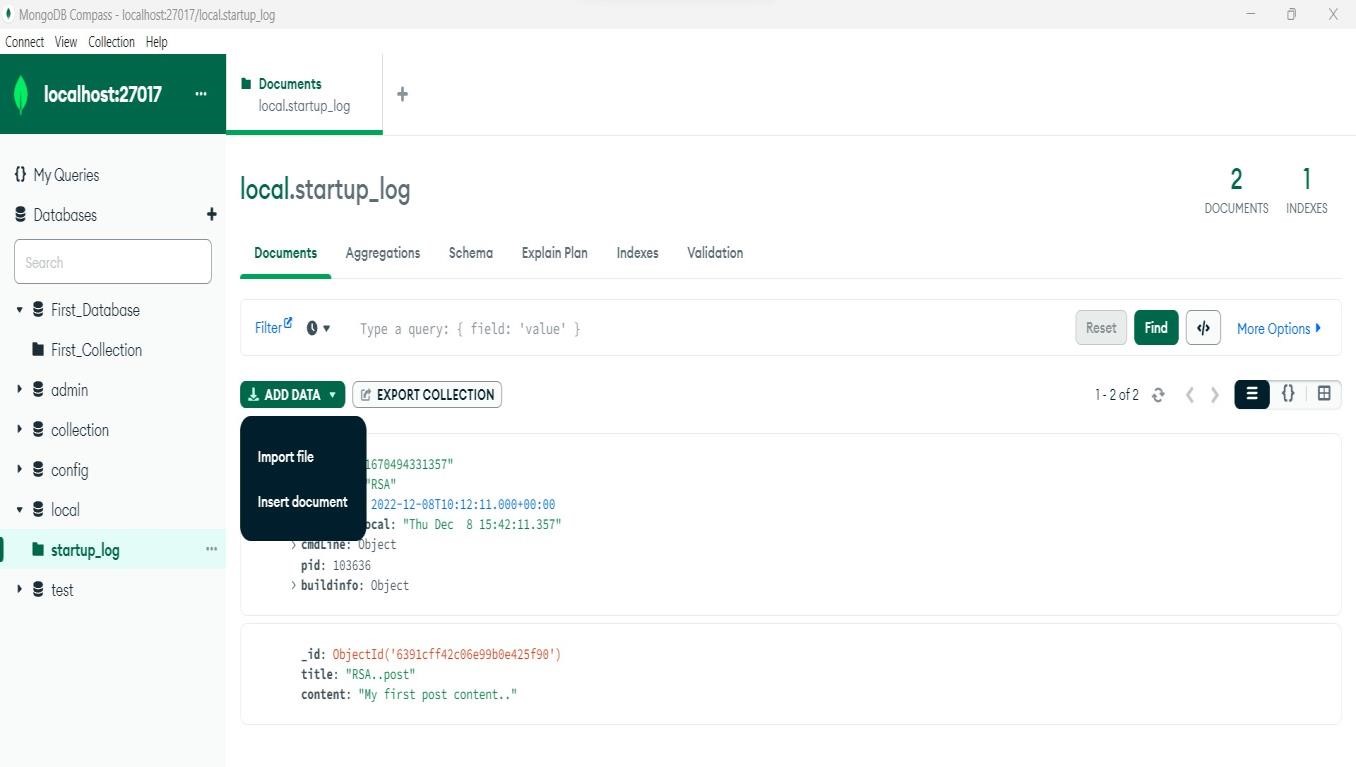
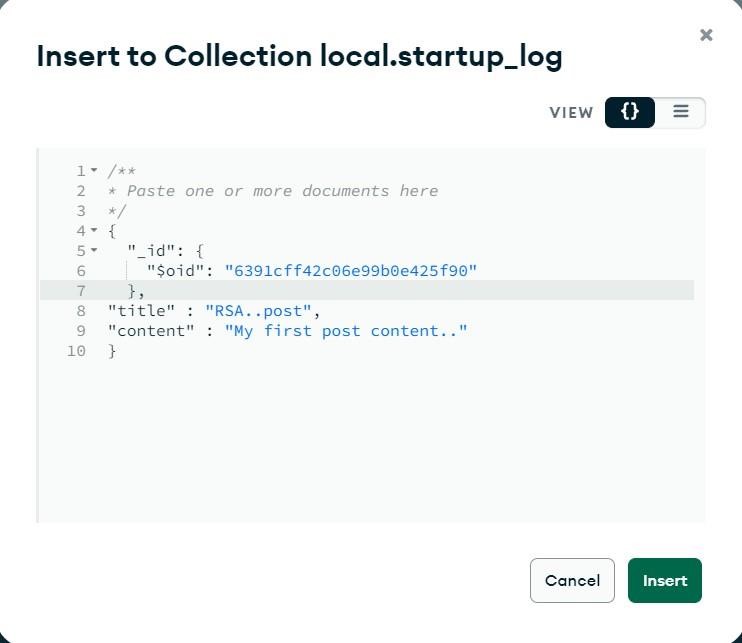
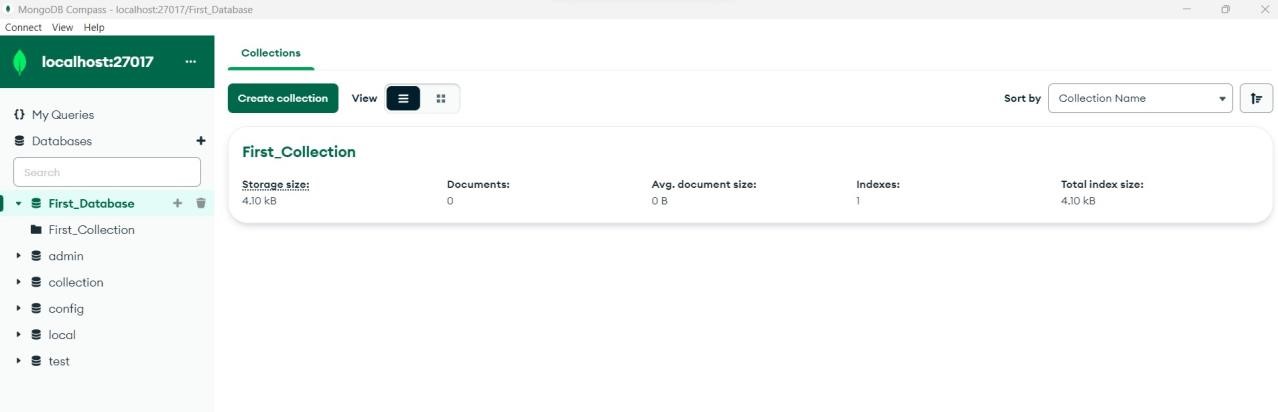
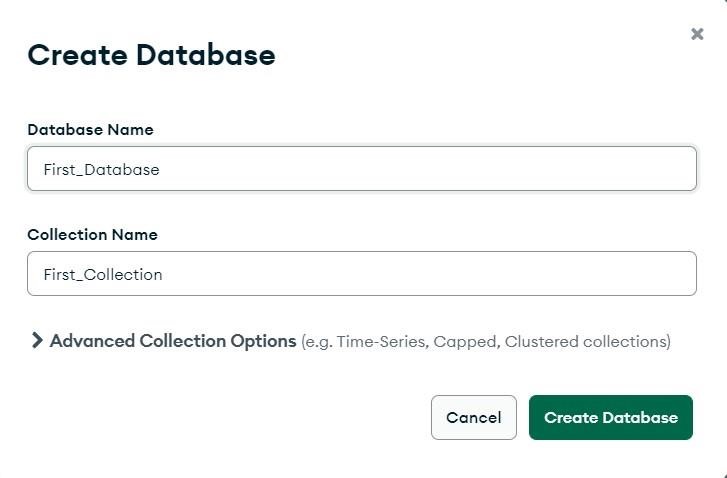
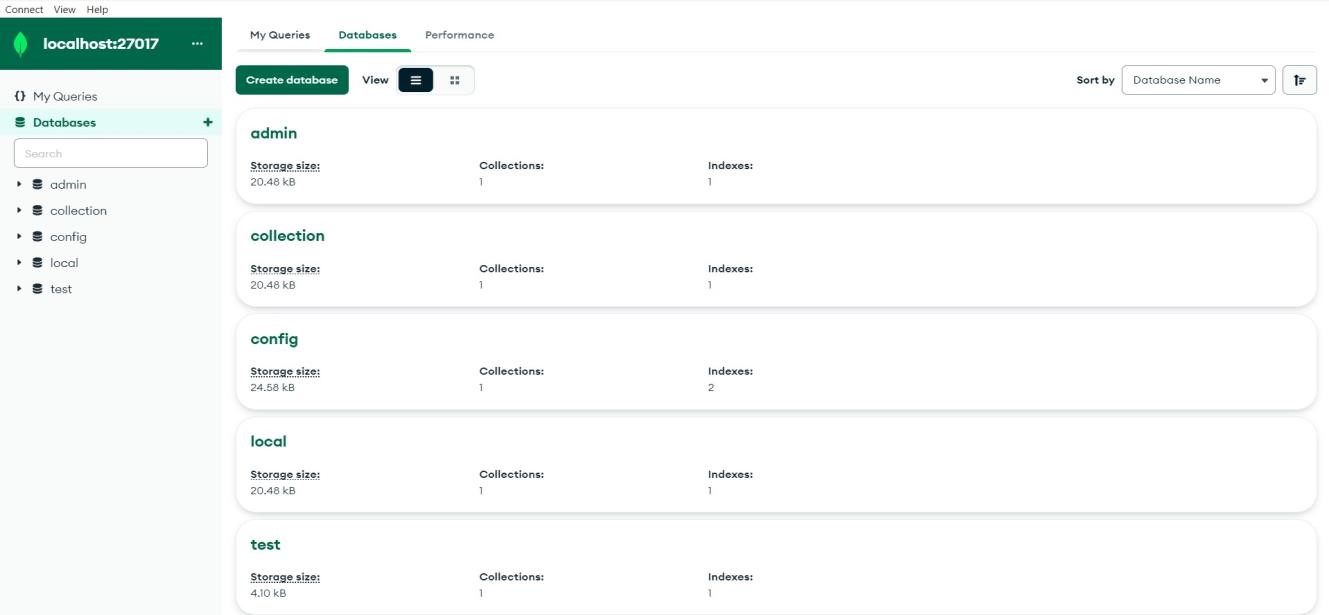
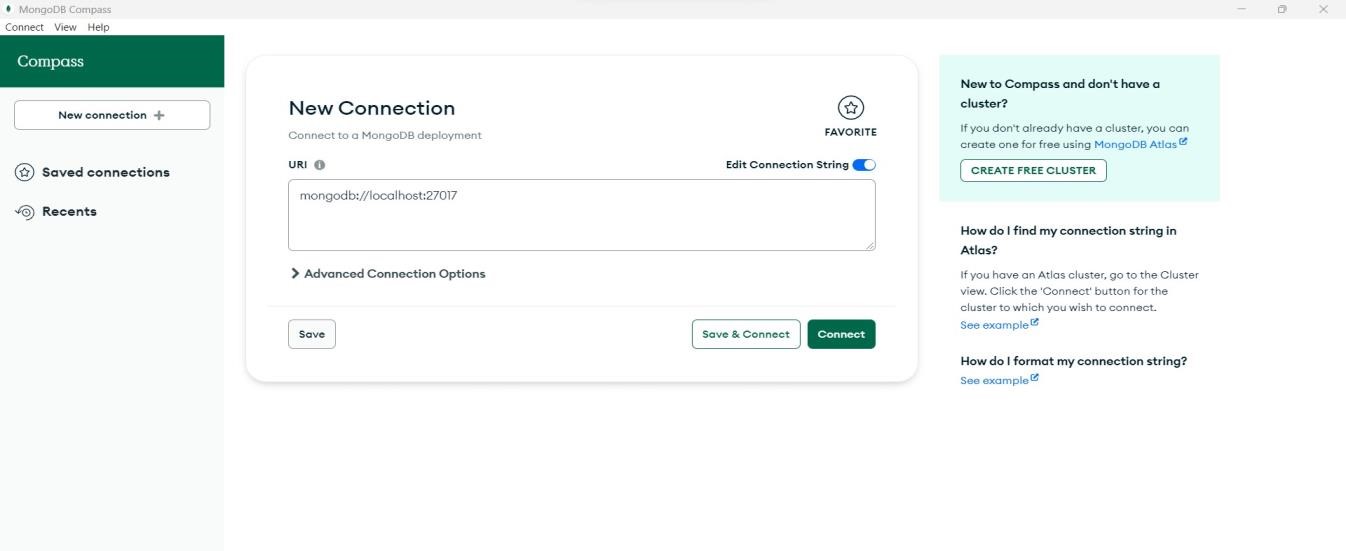
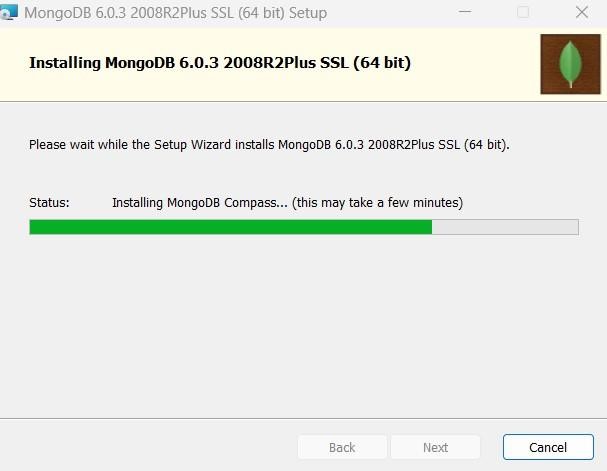
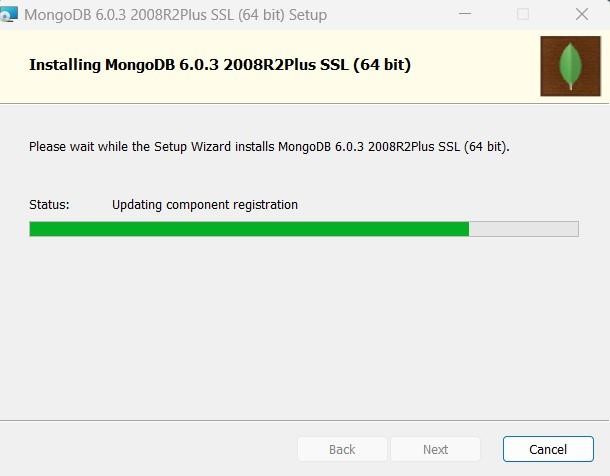
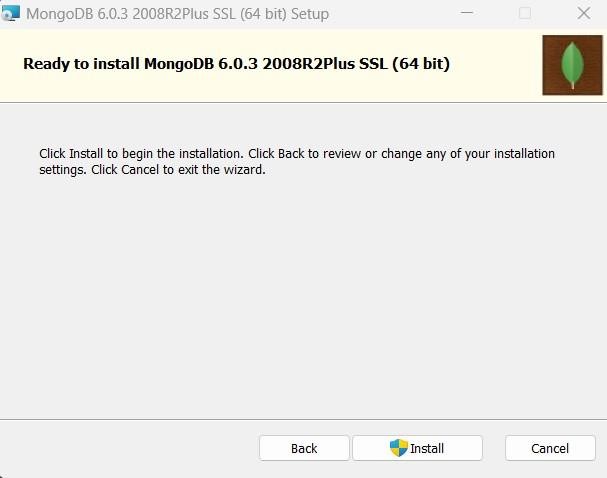
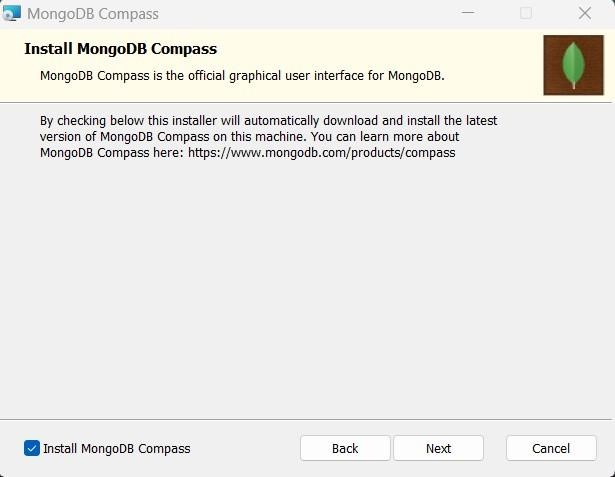
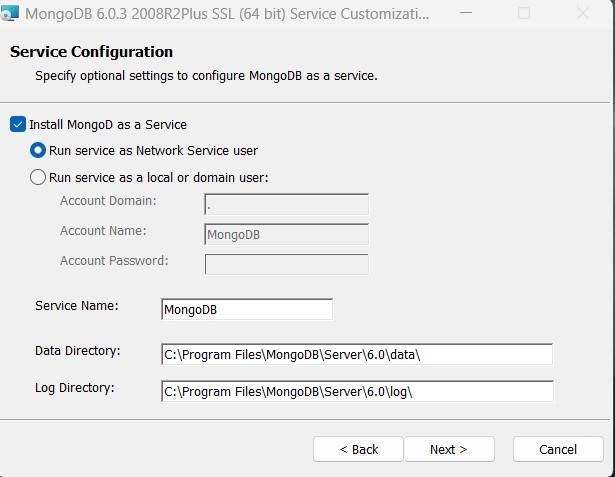
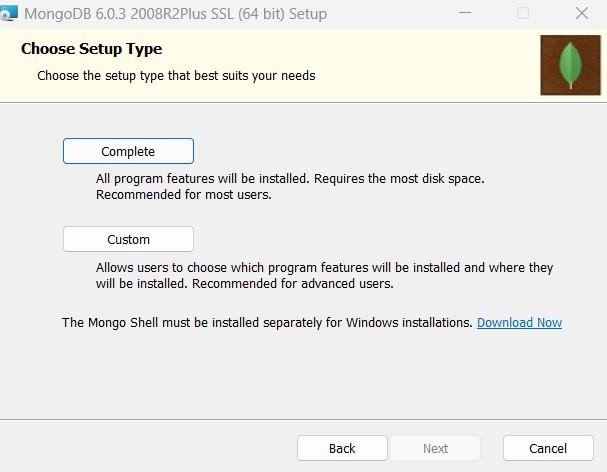
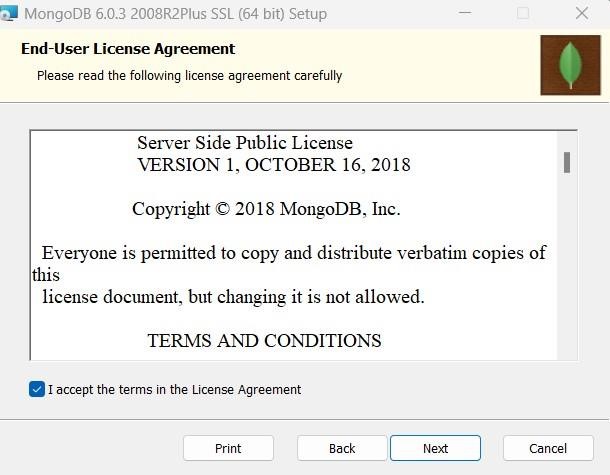
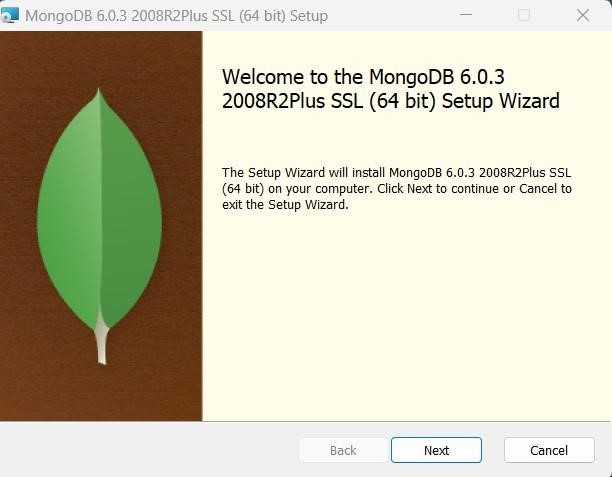
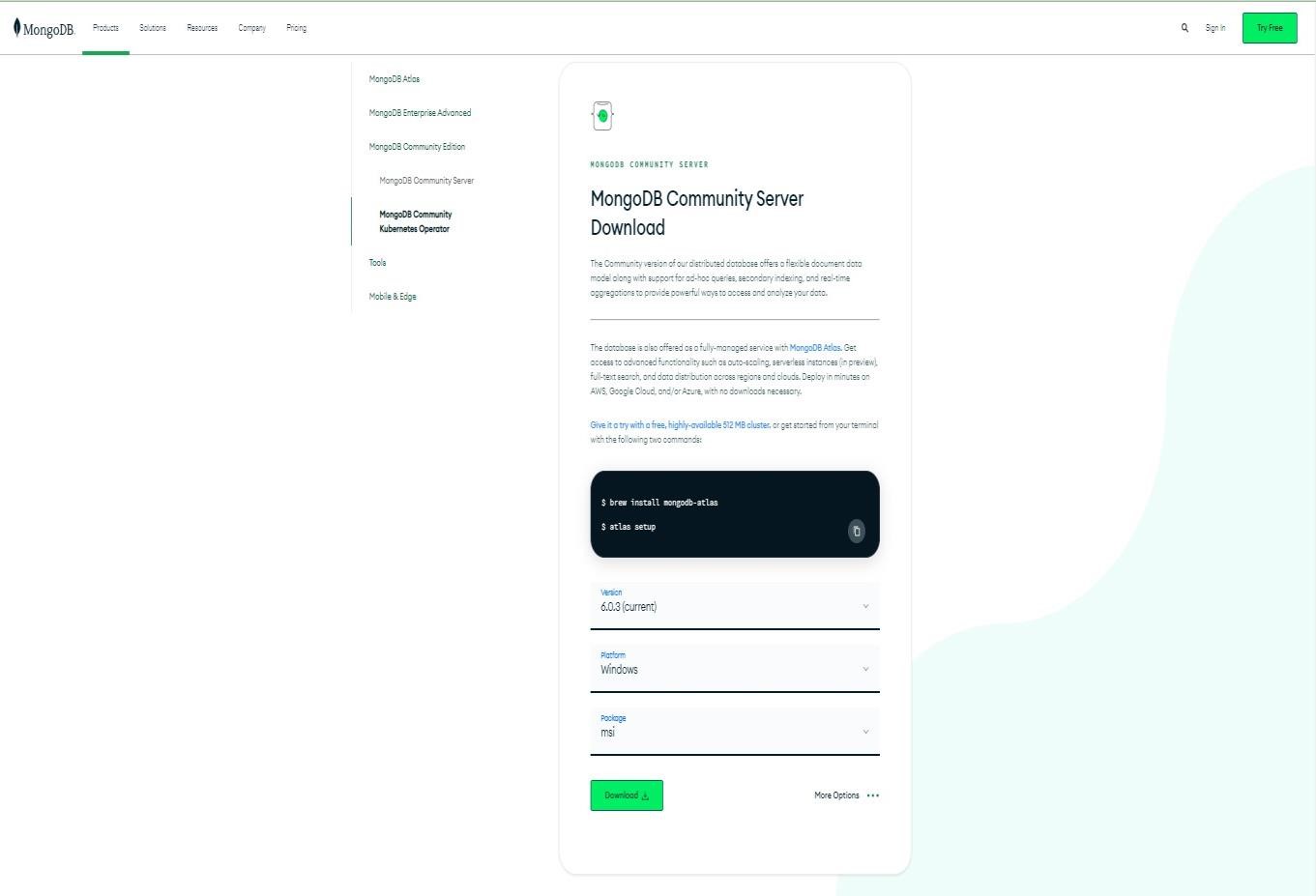
update exam\_test set ex\_name ='st' where Ex\_marks = 610;

select \* from EXAM\_TEST;



**Experiment no.7**

MongoDB Installation:



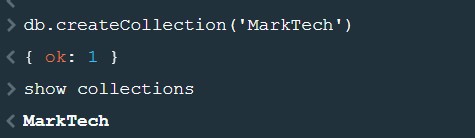
# EXPERIMENT NO. 9

**CRUD in MongoDB:**

**Create Operations**

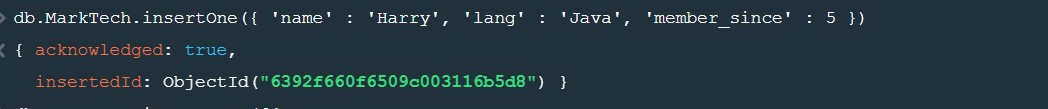
**createCollection():**

db.createCollection('MarkTech')



**insertOne():**

db.MarkTech.insertOne({ 'name' : 'Harry', 'lang' : 'Java', 'member\_since' : 5 })



**insertMany():**

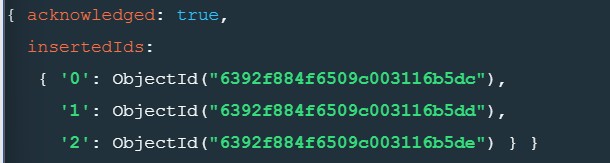
db.MarkTech.insertMany([

{'name':'Vikas','lang':'JavaScript','member\_since':1},

{'name':'Rohan','lang':'Python','member\_since':6},

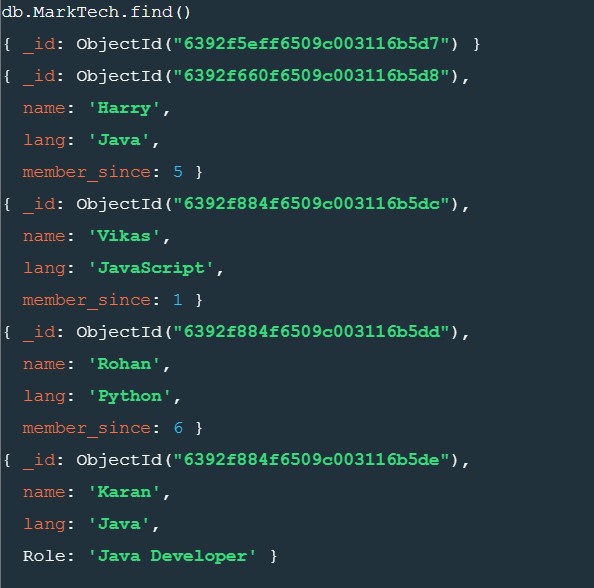
{'name':'Karan','lang':'Java','Role':'Java Developer'}

])

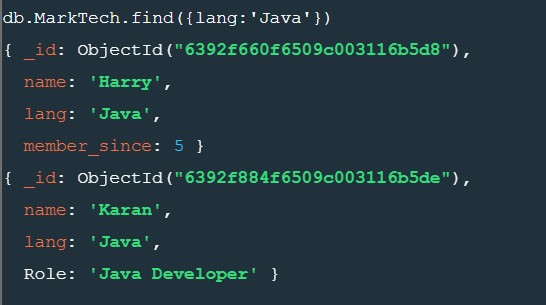


Read Operations

*find()* db.MarkTech.find()

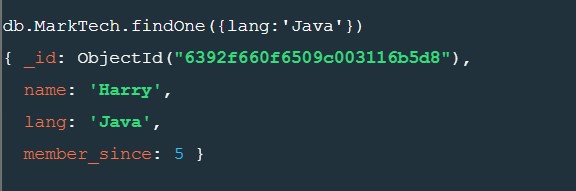


db.MarkTech.find({lang:'Java'})



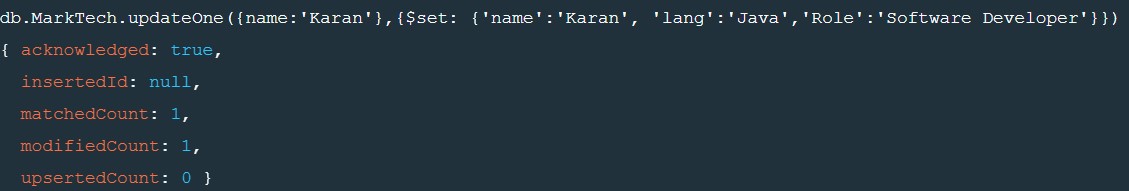
## *findOne()*

db.MarkTech.findOne({lang:'Java'})



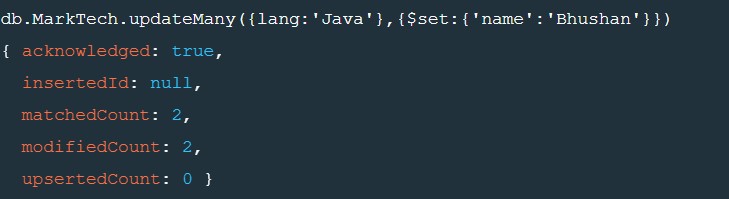
Update Operations*updateOne()*

db.MarkTech.updateOne({name:'Karan'},{$set: {'name':'Karan', 'lang':'Java','Role':'Software Developer'}})



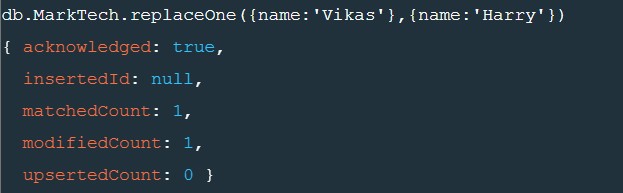
**updateMany()**

db.MarkTech.updateMany({lang:'Java'},{$set:{'name':'Bhushan'}})



**replaceOne()**

db.MarkTech.replaceOne({name:'Vikas'},{name:'Harry'})



**Delete Operations**

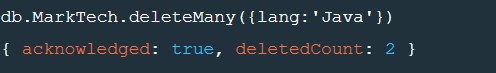
**deleteOne()**

db.MarkTech.deleteOne({name:'Harry'})



**deleteMany()**

db.MarkTech.deleteMany({lang:'Java'})

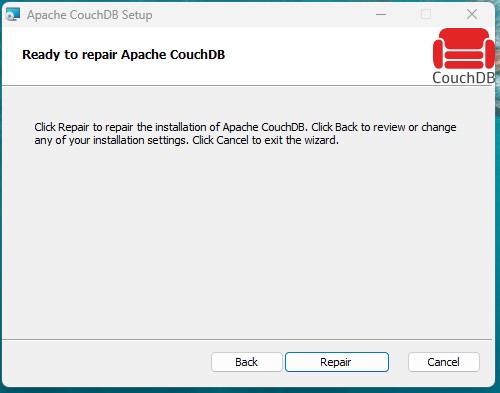


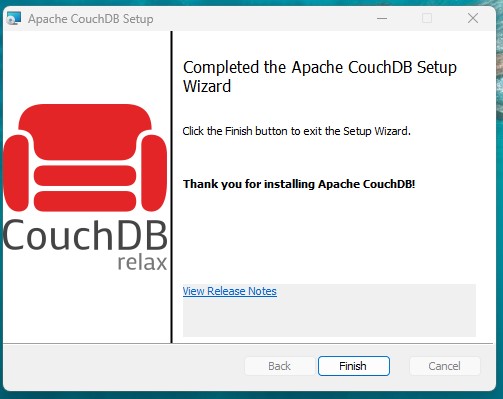
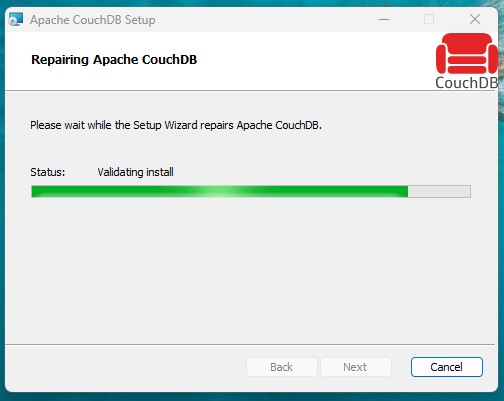
**EXPERIMENT NO. 9**

**CouchDB**

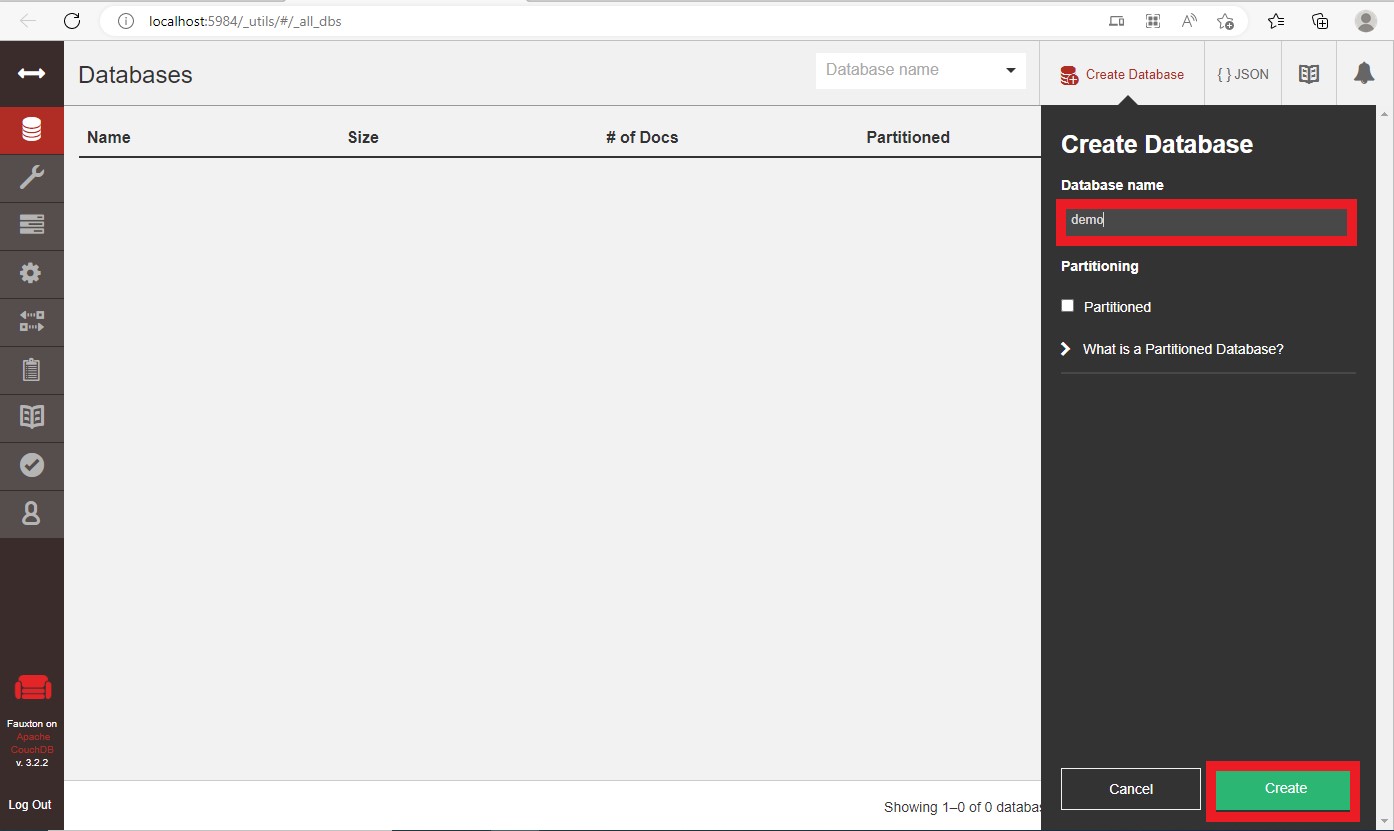
Installation:

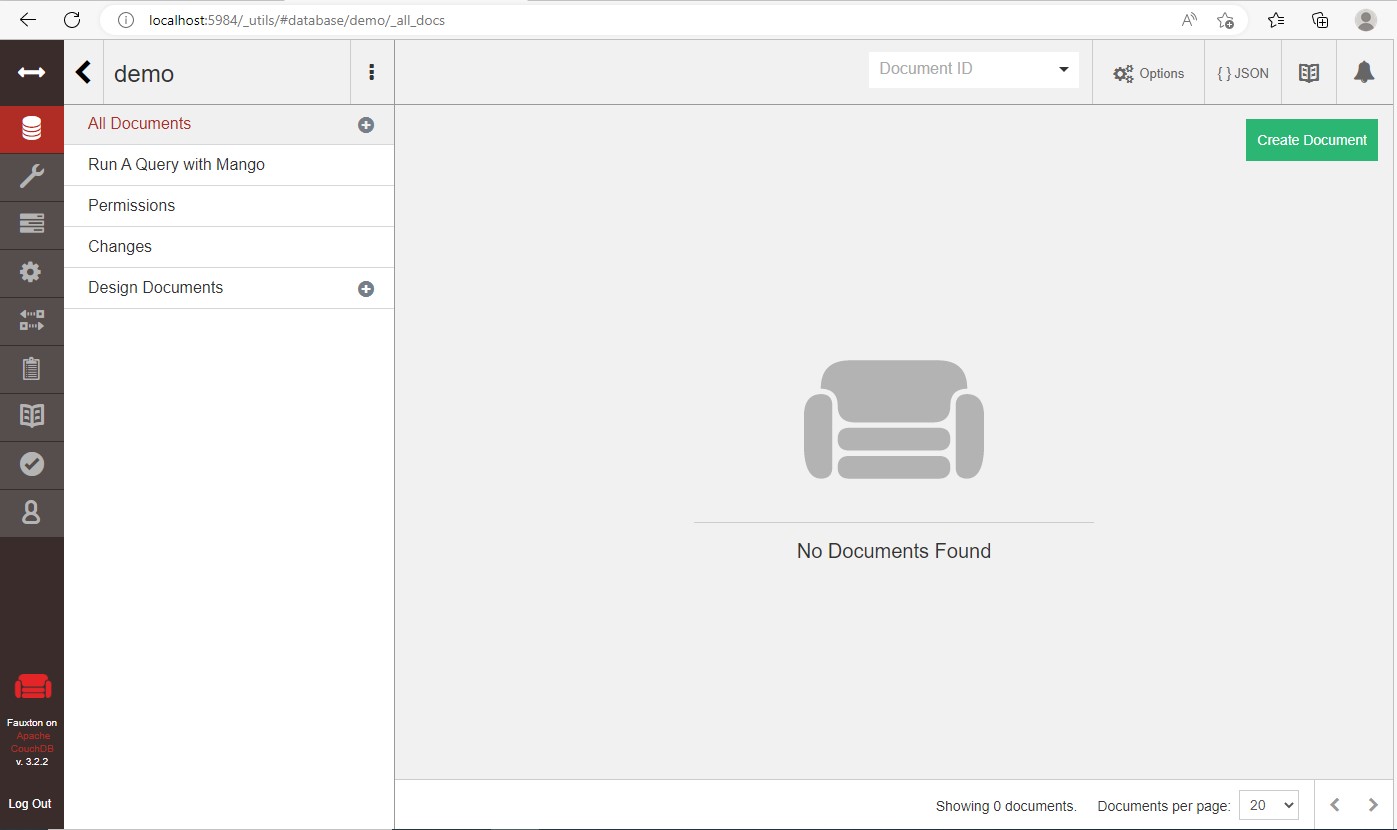




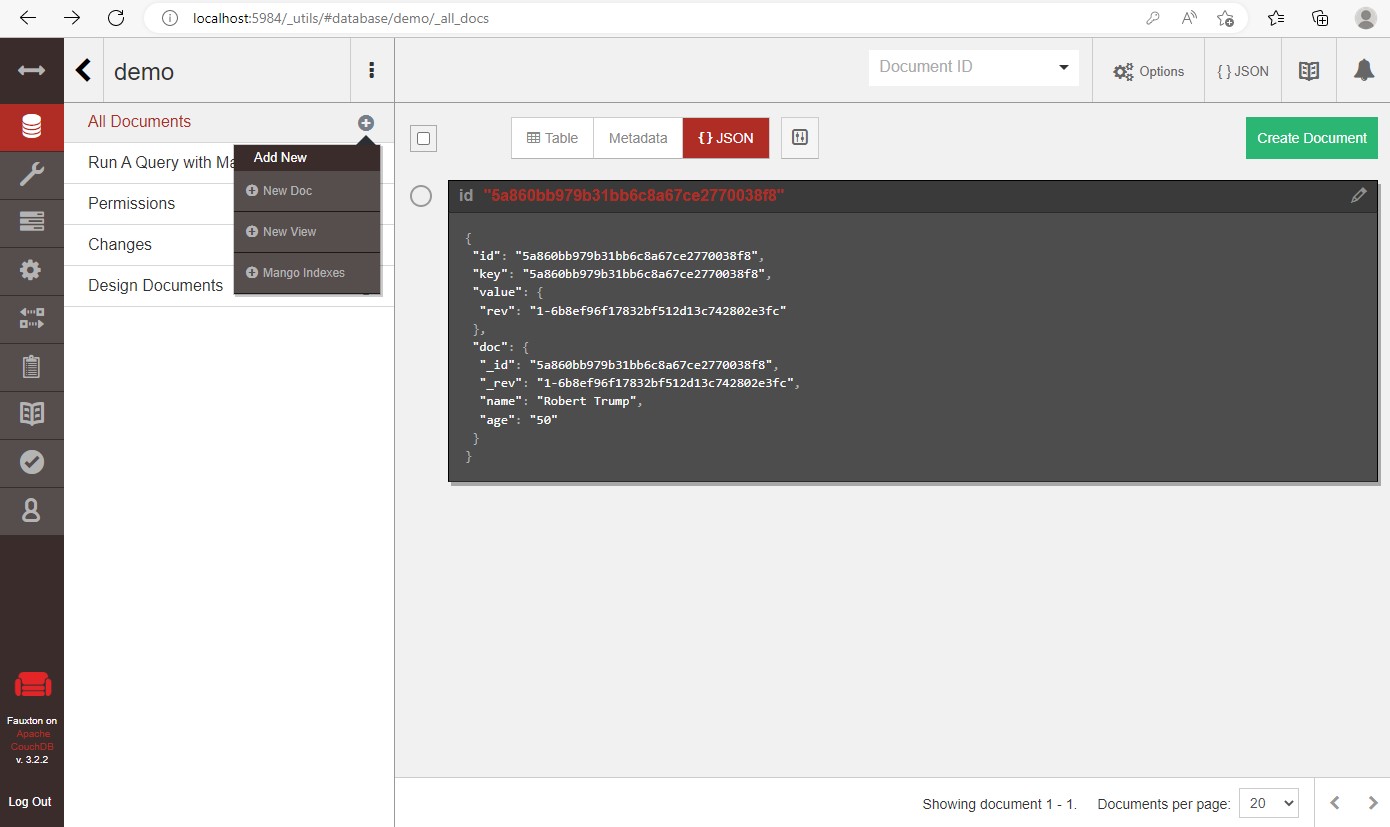


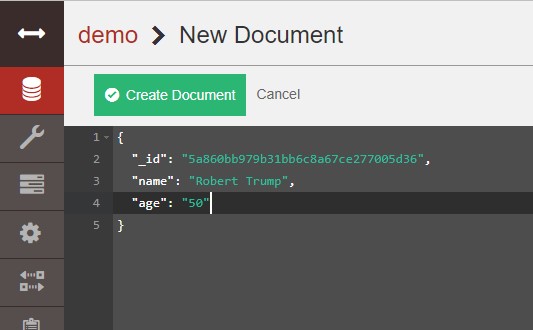
Create Database:

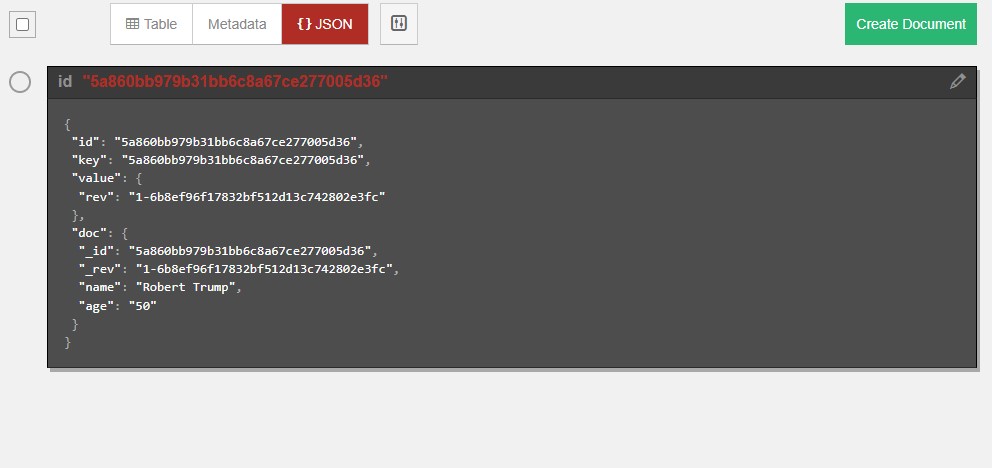




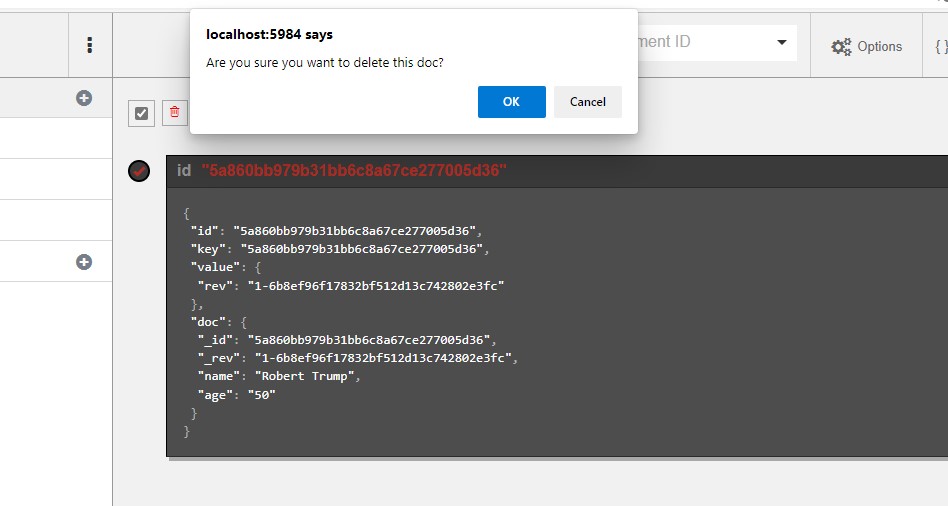
Create Document:

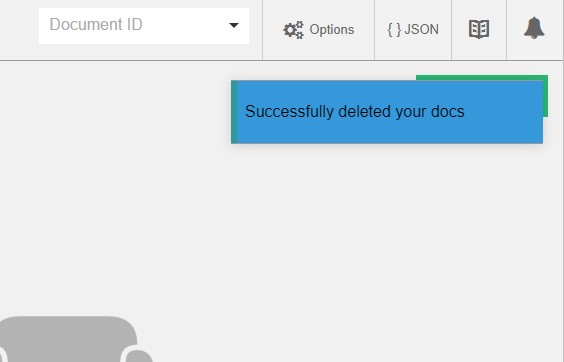




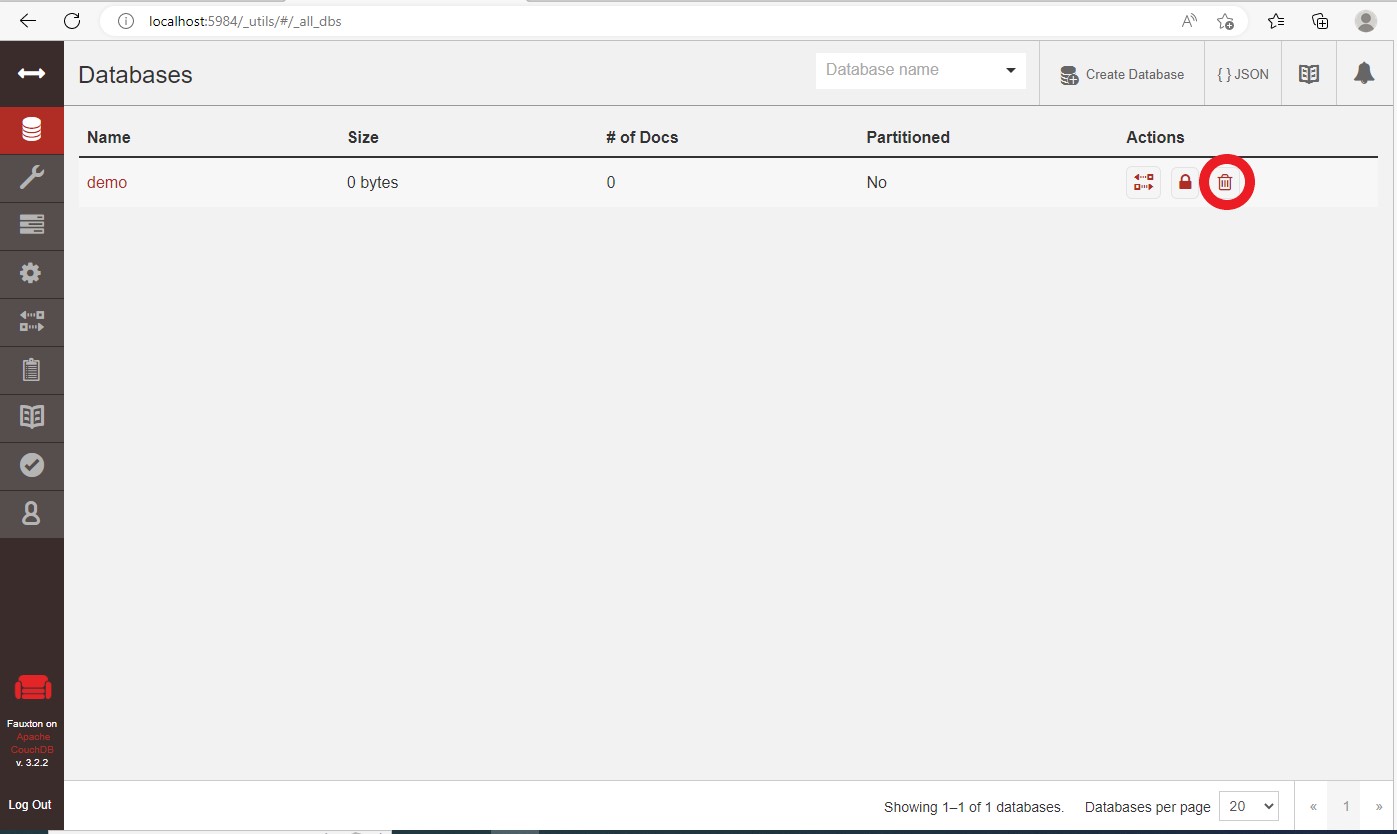


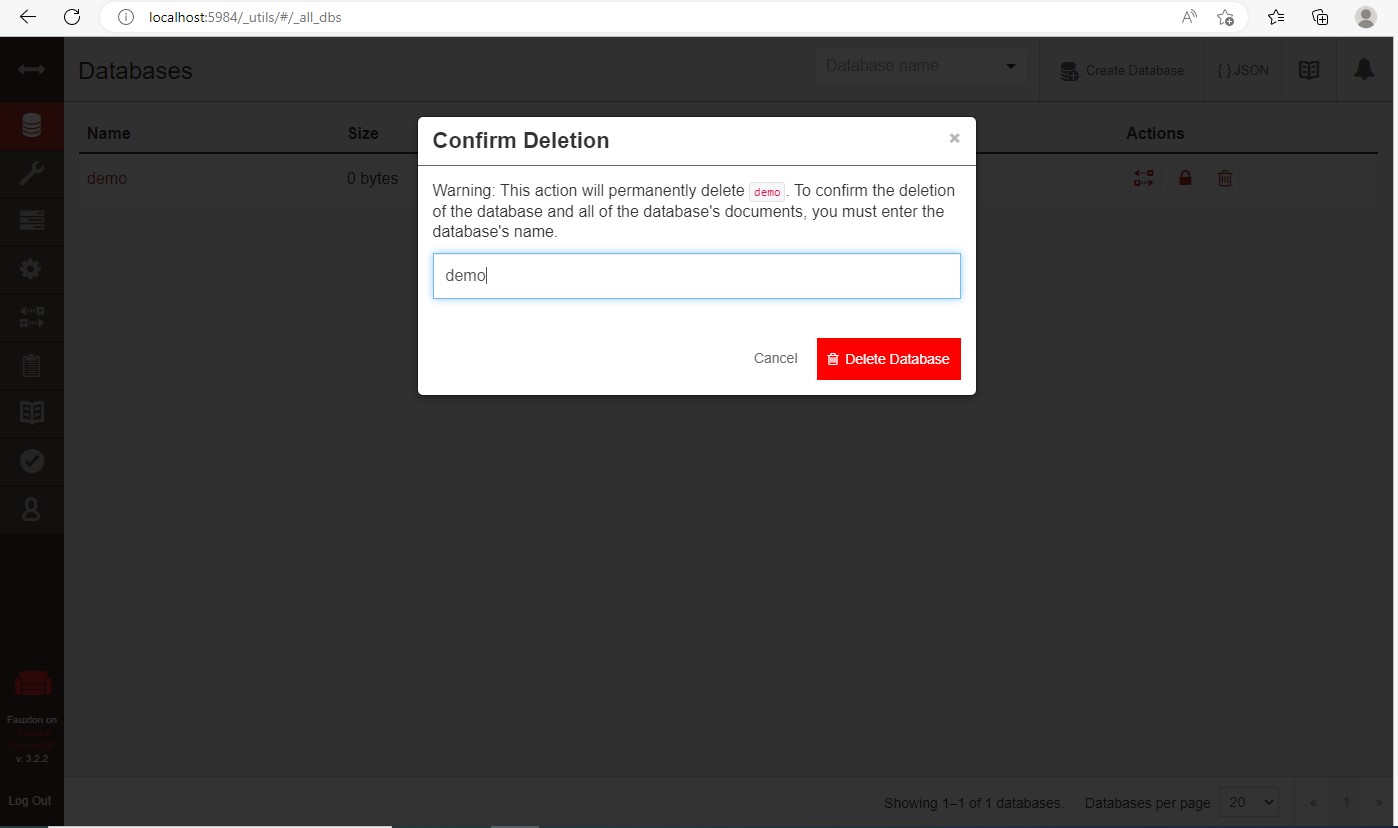
Delete Document:

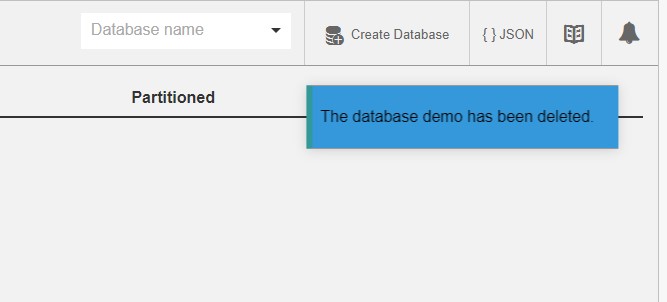




Delete Database:







### Experiment No . 10

Title : Case study of Oracle Database Administration, Security & study of database administrator’s responsibilities.

--to create user :

CREATE USER c##intro\_user IDENTIFIED BY mypassword;

--to grant connect privilege to user : GRANT CONNECT TO c##intro\_user;

--to grant session to user and any kind of privilege:

GRANT CREATE SESSION , GRANT ANY PRIVILEGE TO c##intro\_user;

--

GRANT UNLIMITED TABLESPACE TO c##intro\_user;

--to grant create table privilege to user:

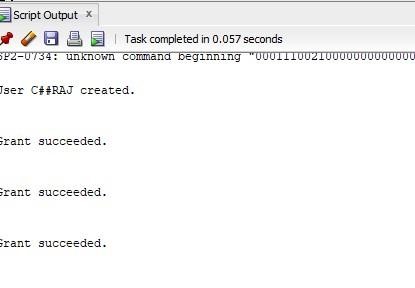
GRANT CREATE TABLE TO c##intro\_user;

--To see all the users : -- SELECT \* FROM all\_users;

--to revoke permission of other user by intro\_user :

REVOKE ALL on table\_name from user\_name;

**Output:**



SQL\*Plus: Release 12.2.0.1.0 Production on Tue Oct 14 12:51:58 2025

Copyright (c) 1982, 2016, Oracle. All rights reserved.

Enter user-name: c##raj

Enter password:

Connected to:

Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production

SQL> create table student12( int ID ,varchar(55) Name)

SQL> insert into student122( int ID ,varchar(55) Name) values (1, 'Alice'), (2, 'Bob'),(3, 'Charlie'); insert into student122( int ID ,varchar(55) Name) values (1, 'Alice'),

SQL> insert into student122( int ID ,varchar(55) Name) values (1, 'Alice')

SQL> insert into student122( int ID ,varchar(55) Name) values (2, 'BOb')

SQL> insert into student122( int ID ,varchar(55) Name) values (3, 'Marie')

SQL> Select \* from student12;

SQL> CREATE TABLE student12 ( ID INT, Name VARCHAR(55) ); Table created.

SQL> INSERT INTO student12 (ID, Name) VALUES (1, 'Alice'); 1 row created.

SQL> INSERT INTO student12 (ID, Name) VALUES (2, 'Bob'); 1 row created.

SQL> INSERT INTO student12 (ID, Name) VALUES (3, 'Charlie');

1 row created.

SQL> SELECT \* FROM student12;

ID NAME

---------- -------------------------------------------------------

1. Alice
2. Bob
3. Charlie

**Experiment No: 11**

**Title: Study of CASE concept and tools.**

**Computer aided software engineering (CASE)** is the implementation of computer facilitated tools and methods in software development. CASE is used to ensure a high-quality and defect-free software. CASE ensures a check-pointed and disciplined approach and helps designers, developers, testers, managers and others to see the project milestones during development.

CASE can also help as a warehouse for documents related to projects, like business plans, requirements and design specifications. One of the major advantages of using CASE is the delivery of the final product, which is more likely to meet real-world requirements as it ensures that customers remain part of the process.

CASE illustrates a wide set of labor-saving tools that are used in software development. It generates a framework for organizing projects and to be helpful in enhancing productivity. There was more interest in the concept of CASE tools years ago, but less so today, as the tools have morphed into different functions, often in reaction to software developer needs. The concept of CASE also received a heavy dose of criticism after its release.

**CASE Tools:**

The essential idea of CASE tools is that in-built programs can help to analyse developing systems in order to enhance quality and provide better outcomes. Throughout the 1990, CASE tool became part of the software lexicon, and big companies like IBM were using these kinds of tools to help create software.

Various tools are incorporated in CASE and are called CASE tools, which are used to support different stages and milestones in a software development life cycle.

**Types of CASE Tools:**

1. **Diagramming Tools:**

It helps in diagrammatic and graphical representations of the data and system processes. It represents system elements, control flow and data flow among different software components and system structure in a pictorial form.

For example, Flow Chart Maker tool for making state-of-the-art flowcharts.

1. **Computer Display and Report Generators:**

It helps in understanding the data requirements and the relationships involved.

1. **Analysis Tools:**

It focuses on inconsistent, incorrect specifications involved in the diagram and data flow. It helps in collecting requirements, automatically check for any irregularity, imprecision in the diagrams, data redundancies or erroneous omissions.

For example,

* + (i) Accept 360, Accompa, CaseComplete for requirement analysis.

* + (ii) Visible Analyst for total analysis.

1. **Central Repository:**

It provides the single point of storage for data diagrams, reports and documents related to project management.

1. **Documentation Generators:**

It helps in generating user and technical documentation as per standards. It creates documents for technical users and end users.

For example, Doxygen, DrExplain, Adobe RoboHelp for documentation.

1. **Code Generators:**

It aids in the auto generation of code, including definitions, with the help of the designs, documents and diagrams.

**Uses in Databases**

CASE tools can serve many functions in database design, including:

* Collecting and analyzing data.
* Designing a data model.
* Feasibility analysis.
* Requirements definition.
* Implementing the database.
* Prototyping.
* Data conversion.
* Generating application code.
* Generating reports.
* Programming and testing.
* Maintenance.

## Case Tool

**Microsoft Visio** is a diagramming tool to create simple as well as complex diagrams and vector graphics. It enables you to create detailed organization charts, floor plans and pivot diagrams according to your business needs. It comes with inbuilt templates which helps you virtually create diagrams of any complexity like shape-based artwork or complex drawing.

**Features of Microsoft Visio:**

* VISIO provides diagramming capabilities to prepare various business engineering Software, and DatabaseArchitecture easily.
* Prepare BPMN Models, Maps, and Diagrams. Prepare Flow Charts, Capture Brain Storming discussions, etc.
* Visio Standard Online and Visio Professional both are downloadable products. You can use these products as long as they’re compatible with your device.
* The tool allows you to capture the information in ways that are valuable for you and your business.
* Provide support for creating varieties of diagramming shapes.
* Visio Pro offers features like automatic updates and multiple installations.

**Visio File Types**

Now in this MS Visio tutorial, we will learn about different Visio File Types. Following are important file types available with Visio software:

* **VSD (Visio Drawing):** This file extension is associated with Visio binary file format. It is used for storing flowcharts and diagram document files.
* **VSS (Visio Stencil):** This file extension is associated with Microsoft Visio. The file contains smart shapes (stencils). The file is in binary Visio document format.
* **VST (Visio Template):** The VST file extension is associated with Microsoft Visio templates.
* **VDW (Visio Web drawing):** The VDW extension is associated with Microsoft Office Visio. It contains web drawing which is created in Microsoft Office Visio.

**Download and Install Visio**

**Notes:** You require to have an active Visio license to download Visio. You need to run the “Click-toRun” version of Microsoft Office before installing Visio. The Click-to-Run version is installed from [https://www.office.com.](https://www.office.com/)

**How to start Visio**

2.

Click on

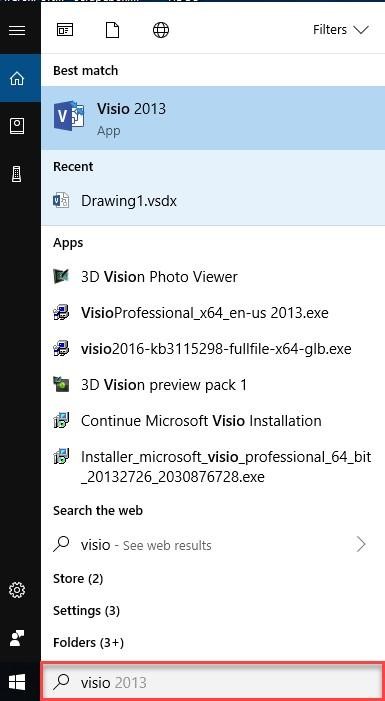
“

Visio

2013

”

menu.



**Step 1)**



Click on

“

start

”

button.



1.



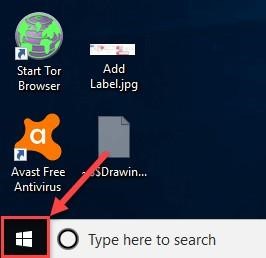
Type Visio in a

“

search

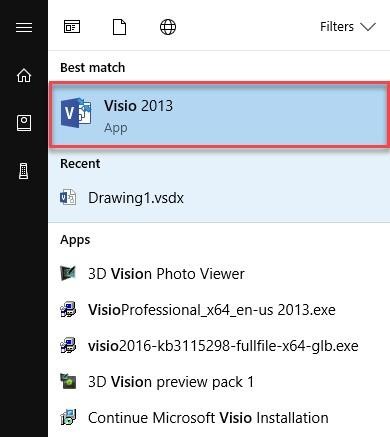
”

box.

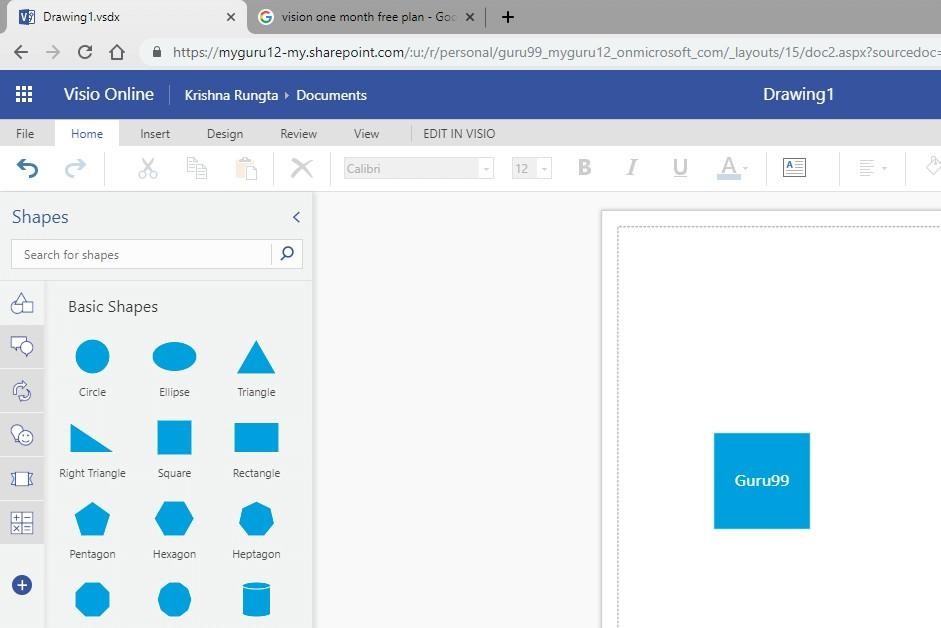


**You**

**can also register with Office 365 and access Microsoft Visio Online.**



And create diagrams



**How to use Microsoft Visio**

Now in this Visio tutorials guide, we will learn how to use Microsoft Visio:

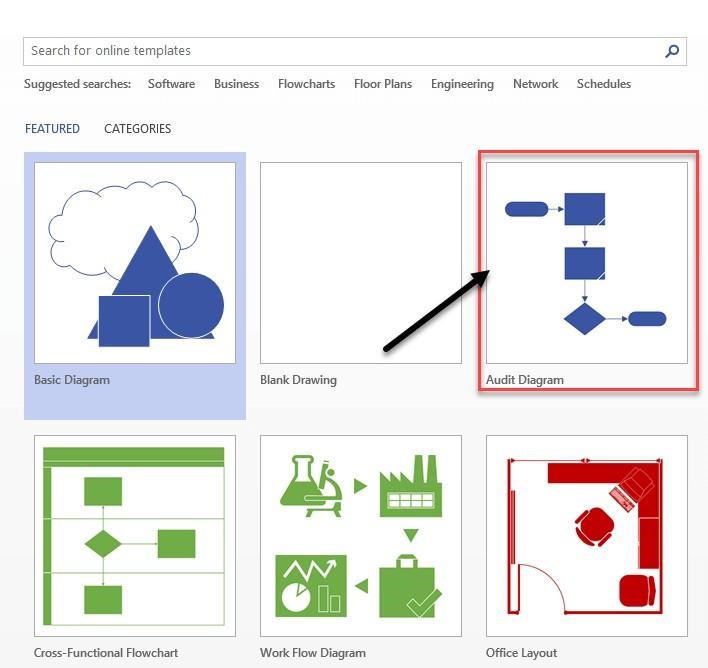
Here, are steps to plan and draw a basic Visio chart:

**Step 1)** Gather the suitable detail.

Decide what you’re trying to accomplish and gather the suitable detail requirements to draw your diagram or chart.

**Step 2)** Determine chart type.

Determine which chart type is best suited to illustrate the information you want to convey. In our case, we will create a login flowchart.



**Step 3)** Open the template.

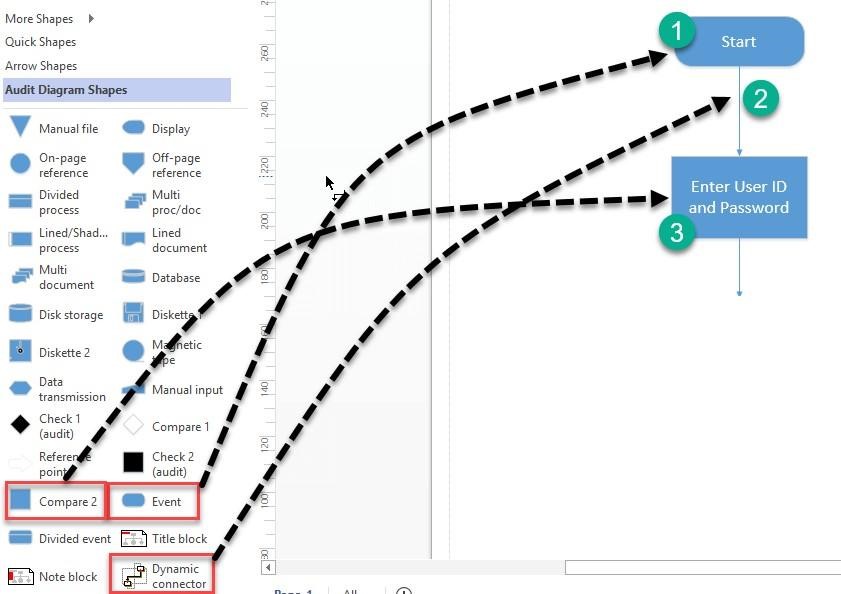
In Visio, select and open the template according to your chart type.

Each shape represents different things, which ranges from simple rectangles, squares, ovals, and arrows to hundreds of highly specialized shapes and symbols.

**Step 4)** Connect shapes.

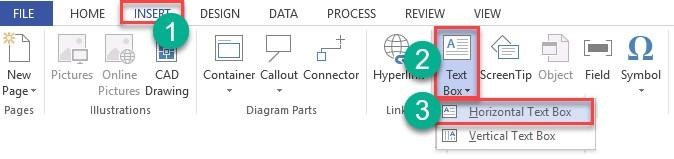
Drag and connect shapes or symbols to illustrate the items and their flow.

1. Drag terminator shape to drawing area.
2. Drag Compare2 shape.
3. Drag dynamic connector to drawing area and connect these two shapes.

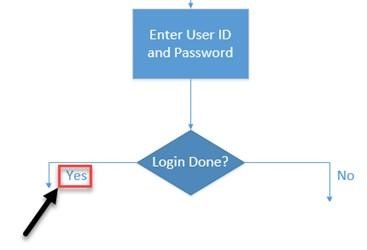


**Step 5)** Add a label.

To add a label, go to Insert tab then click on “Text Box” and select horizontal Text Box.



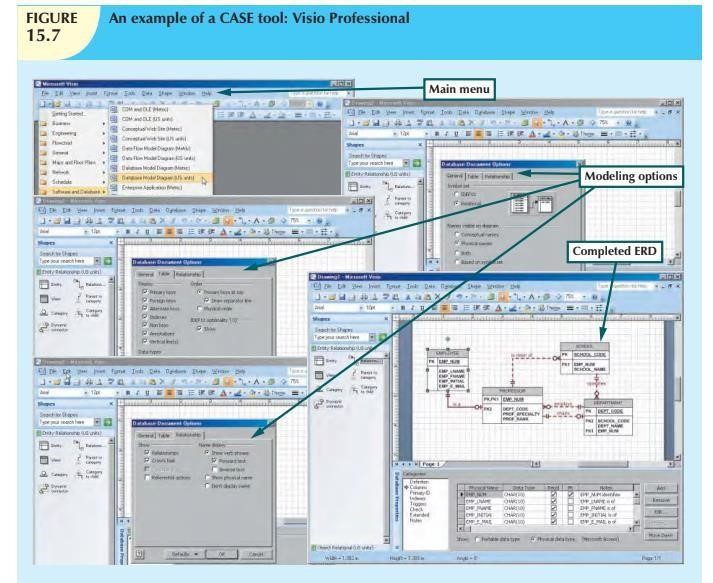
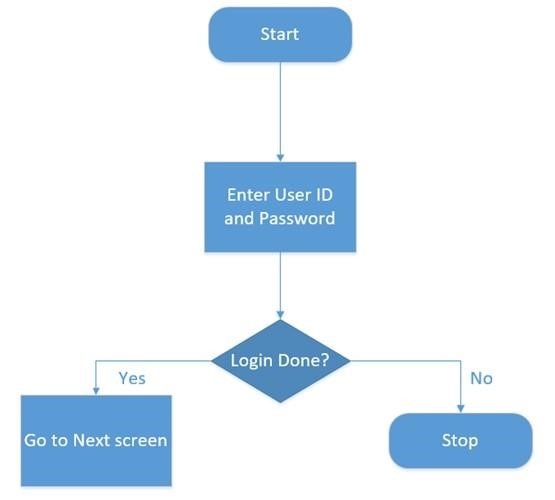
Click on drawing area to add label.



**Step 6)** Review your diagram.

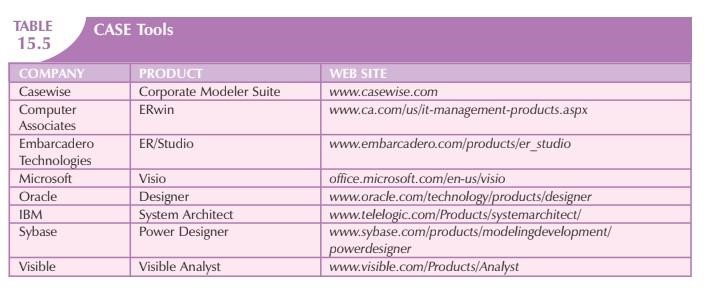
Lastly, review your diagram and compare with your rough design.

Below is an example of a flowchart showing a login process?



**Figure: illustrates how Microsoft Visio Professional can be used to produce an ER diagram.**

**Shows a short list of the many available CASE tool vendors**



**Applications of Microsoft Visio**

Here, are essential applications and functions of Microsoft Visio:

|  |  |
| --- | --- |
| **Usage** | **Description** |
| Business Process  Diagrams | It is one of the most popular used of Visio. It helps you to illustrate business processes. |
| Organization  Charts | Organization charts are frequently used in the diagram for your business. |
| Brainstorming  Diagram | A brainstorming diagram allows you to record and develop any set of related ideas or information like new strategies for business, book outlines, travel notes, meeting minutes, etc. |
| Building Plan | You can also build the most accurate building plans in Visio. This diagram also provides an overhead view of walls, doors, windows, and furniture. |
| Flowchart | A flowchart can be used to document and analyze a process; standardize a process for efficiency and quality. It also uses for training or understanding by other parts of the organization. |
| Analysis | Visio offers focus approach to create professional Data flow diagrams (DFDs) for all types of projects. |
| In business | You can draw Business Process Modeling and Notation shortly known as (BPMN). It is targeted at participants and other stakeholders in a business process. |
| In Chemical  Engineering | Visio allows you to draw a Process Flow Diagram (PFD), which is a type of flowchart that explains the relationships between various components at an industrial plant. |

**EXPERIMENT NO. 12**

create table sales( continent varchar(20), country varchar(20), city varchar(20), units\_sold int

);

insert into sales values('North America','Canada','Toronta',10000); insert into sales values('North America','Canada','Motreal',5000); insert into sales values('North America','Canada','Vancouver',15000); insert into sales values('Asia','China','Hong Kong',7000); insert into sales values('Asia','China','Shanghai',3000); insert into sales values('Asia','Japan','Tokyo',5000); insert into sales values('Europe','UK','London',6000); insert into sales values('Europe','UK','Manchester',12000); insert into sales values('Europe','France','Paris',5000);

select \* from sales order by continent, country, city;

//select continent, sum(units\_sold)

//from sales

//group by continent;

//select country, sum(units\_sold)

//from sales

//group by country;

//select city, sum(units\_sold)

//from sales

//group by city;

-- drill down :

select continent, country, city, sum(units\_sold) from sales group by grouping sets(continent, country, city);

-- RollUP :

select continent, country, city, sum(units\_sold) from sales group by rollup(continent, country, city);

-- cube :

select continent, country, city, sum(units\_sold) from sales group by cube(continent, country, city);

