M.Sc C.S - I
SEM I
E-Journal

Roll No.	**006**
Name	**HEMAN SHAKTHI MOHAN UDAIYAR**
Subject	**ADVANCED DATABASE SYSTEMS**

Thakur Educational Trust's (Regd.)

Thakur College of Science & Commerce

UGC Recognised • Affiliated to University Of Mumbai
(NAAC Accredited with Grade "A" [3rd Cycle] & ISO 9001:2015 Certified)

CERTIFICATE

This is here to certify that Mr./~~Ms~~. HEMAN SHAKTHI MOHAN UDAIYAR, Seat Number 006 of M.Sc. I Computer Science, has satisfactorily completed the required number of experiments prescribed by the UNIVERSITY OF MUMBAI during the academic year 2021 – 2022.

Date:
Place: Mumbai

Teacher In-Charge   Head of Department

External Examiner

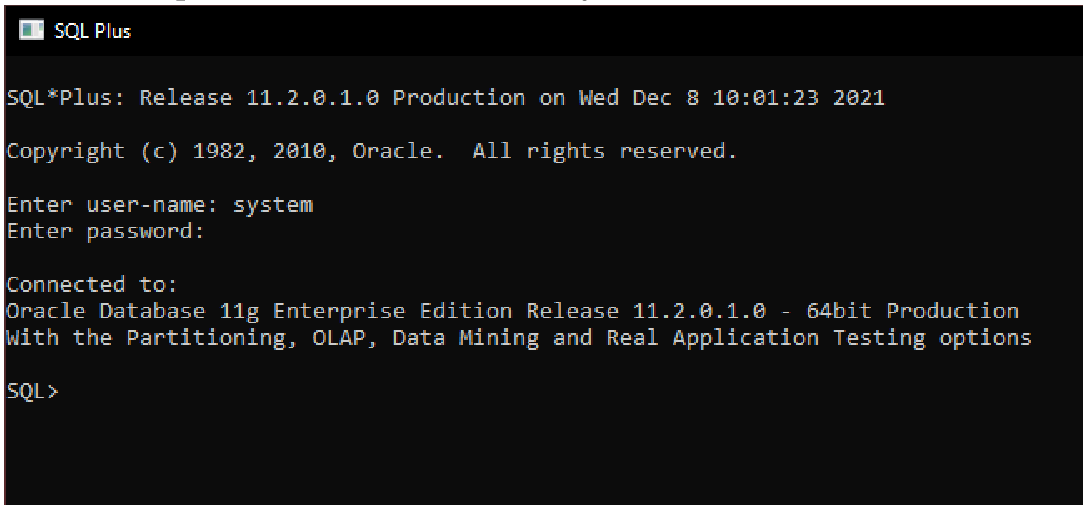
INDEX

Sr. No.	Practical Name	Date
1	For a given global conceptual schema, divide the schema into horizontal and vertical fragmentation and place them on different nodes. Execute queries on these fragments that will demonstrate a distributed database environment.	25-09-2021
2	Place the replication of global conceptual schema on different nodes and execute queries that will demonstrate a distributed database environment.	09-10-2021
3	CRUD operation using MongoDB.	18-10-2021
4	Create different types that include attributes and methods. Define tables for these types by adding a sufficient number of tuples. Demonstrate insert, update and delete operations on these tables. Fire suitable queries on them.	23-10-2021
5	Create a temporal database and issue queries on it.	30-10-2021
6	Create a table that stores the special data and issue queries on it.	13-11-2021
7	Create a table employee having dept_id as number data type and employee_spec as XML data type (XM_Type). The employee_spec is a schema with attributes emp_id, name, email, acc_no, managerEmail, dataOf Joining. Insert 10 tuples into the employee table. Fire the following queries on the XML database.	16-11-2021

PRACTICAL NO: 1

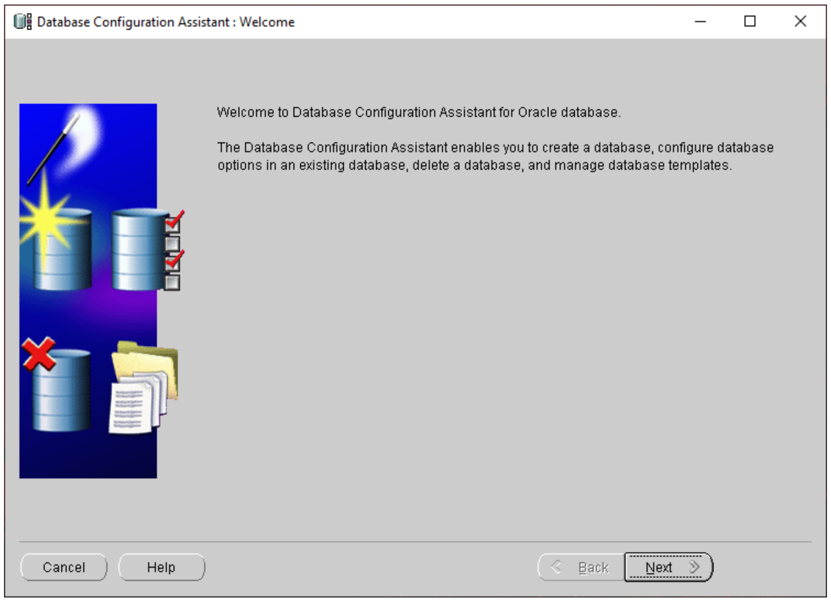
Aim: For a given a global conceptual schema, divide the schema into horizontal and vertical fragmentation and place them on different nodes. Execute queries on these fragments that will demonstrate distributed databases environment.

Software Requirement: Oracle Database 11g.



```
SQL Plus

SQL*Plus: Release 11.2.0.1.0 Production on Wed Dec 8 10:01:23 2021

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

Enter user-name: system
Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL>
```

How to Create Two Database
Steps to Create Database db1 and db2
Step 1:- Open Start Menu on Window Explorer Go to Database Configuration Assistant.

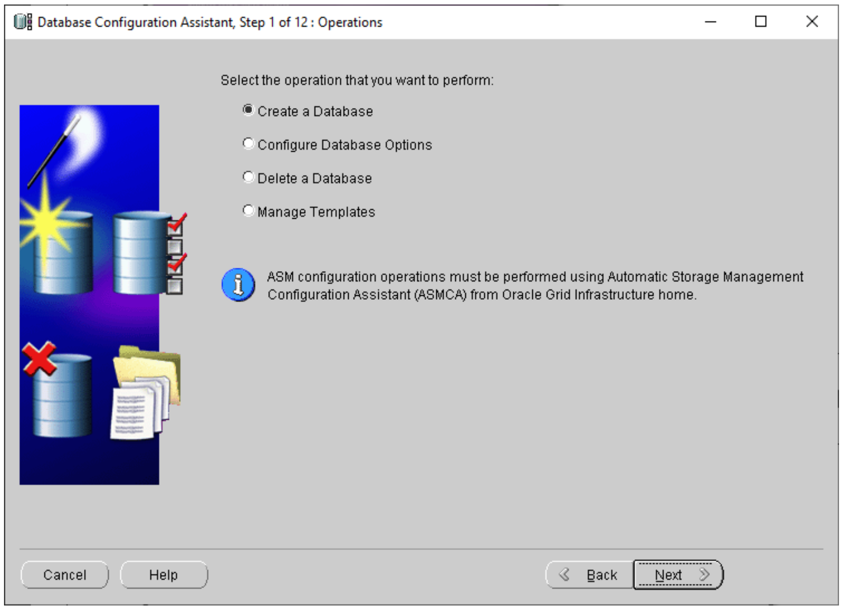


Database Configuration Assistant : Welcome

Welcome to Database Configuration Assistant for Oracle database.

The Database Configuration Assistant enables you to create a database, configure database options in an existing database, delete a database, and manage database templates.

Step 2: Select Option Create a Database.

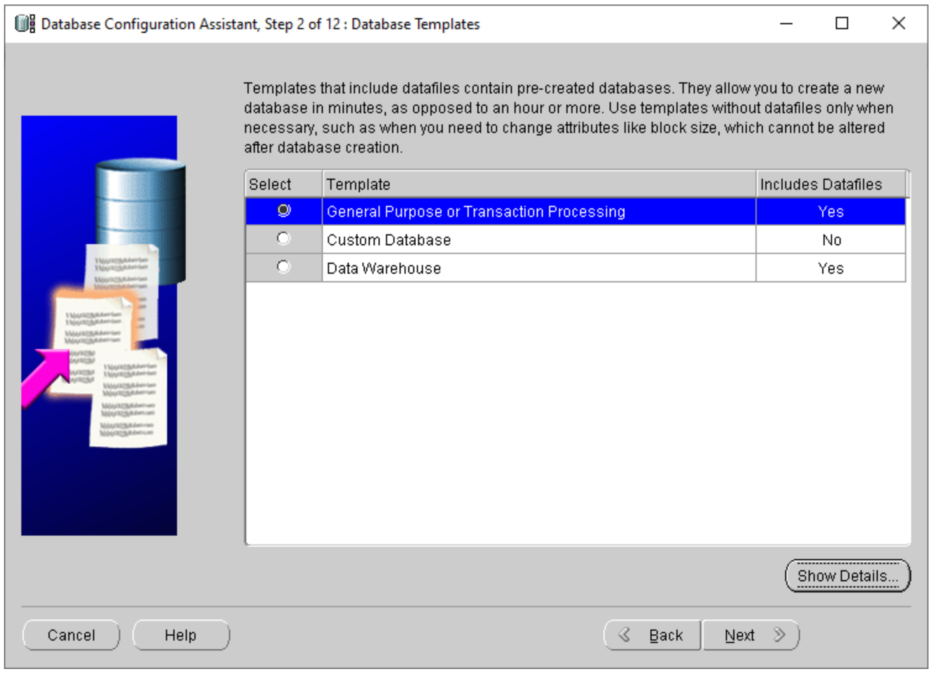

Database Configuration Assistant, Step 1 of 12 : Operations

Select the operation that you want to perform:

- [✓ Create a Database]
- Configure Database Options
- Delete a Database
- Manage Templates

ASM configuration operations must be performed using Automatic Storage Management Configuration Assistant (ASMCA) from Oracle Grid Infrastructure home.

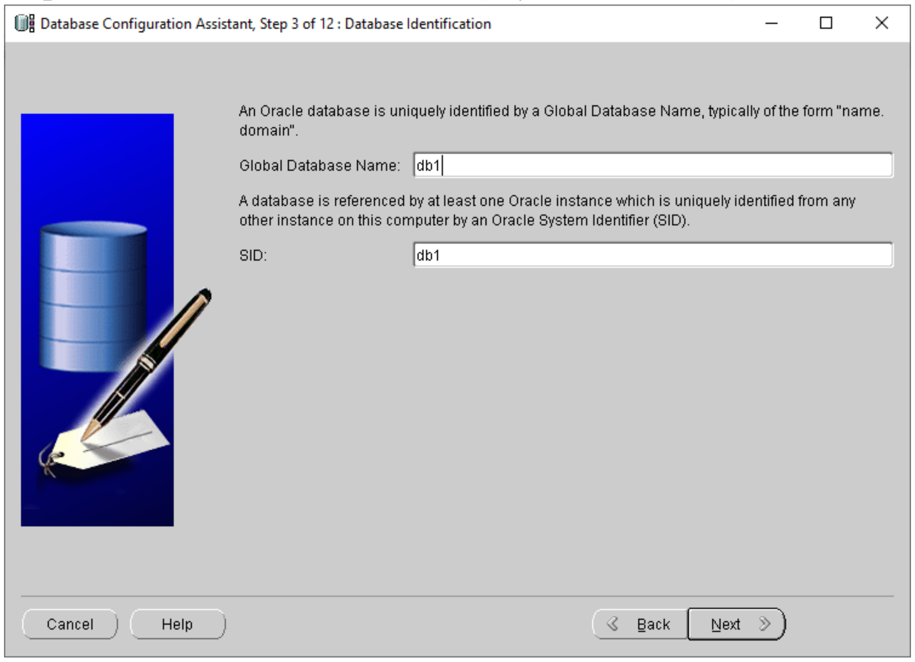
Step 3: Select Option General Purpose or Transaction Processing or You can Create your Own Custom Database.


Database Configuration Assistant, Step 2 of 12 : Database Templates

Templates that include datafiles contain pre-created databases. They allow you to create a new database in minutes, as opposed to an hour or more. Use templates without datafiles only when necessary, such as when you need to change attributes like block size, which cannot be altered after database creation.

Select	Template	Includes Datafiles
●	General Purpose or Transaction Processing	Yes
○	Custom Database	No
○	Data Warehouse	Yes

Step 4: Give Database Name as db1 (of your own choice).


Database Configuration Assistant, Step 3 of 12 : Database Identification
Cancel
Help
Back
Next

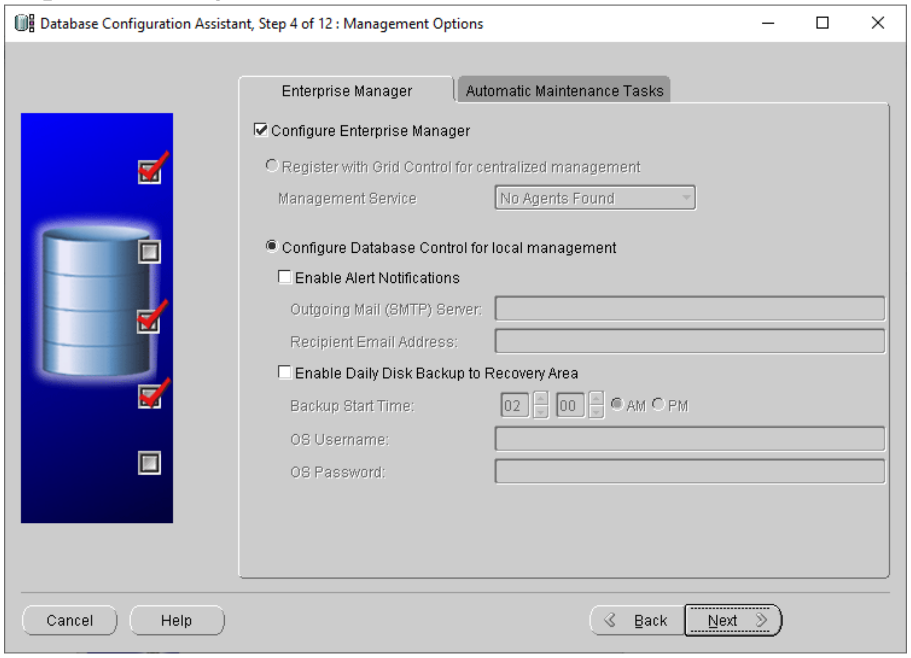
An Oracle database is uniquely identified by a Global Database Name, typically of the form "name.domain".

Global Database Name: db1

A database is referenced by at least one Oracle instance which is uniquely identified from any other instance on this computer by an Oracle System Identifier (SID).

SID: db1

Step 5: No changes Needed, Click on Next.


Database Configuration Assistant, Step 4 of 12 : Management Options
Enterprise Manager
Automatic Maintenance Tasks
Cancel
Help
Back
Next

[✓] Configure Enterprise Manager

○ Register with Grid Control for centralized management

Management Service: No Agents Found

● Configure Database Control for local management

[ ] Enable Alert Notifications

Outgoing Mail (SMTP) Server:

Recipient Email Address:

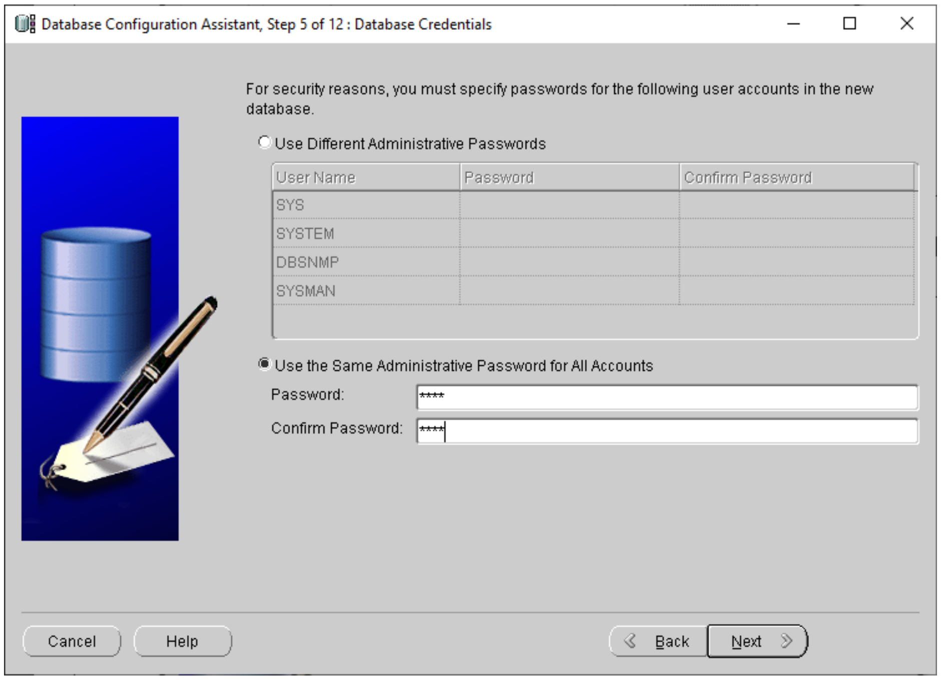
[ ] Enable Daily Disk Backup to Recovery Area

Backup Start Time: 02 00 ● AM ○ PM

OS Username:

OS Password:

Step 6 : Select "Use the same Administrative Password for All Accounts" and enter the password



Database Configuration Assistant, Step 5 of 12 : Database Credentials

For security reasons, you must specify passwords for the following user accounts in the new database.

( ) Use Different Administrative Passwords

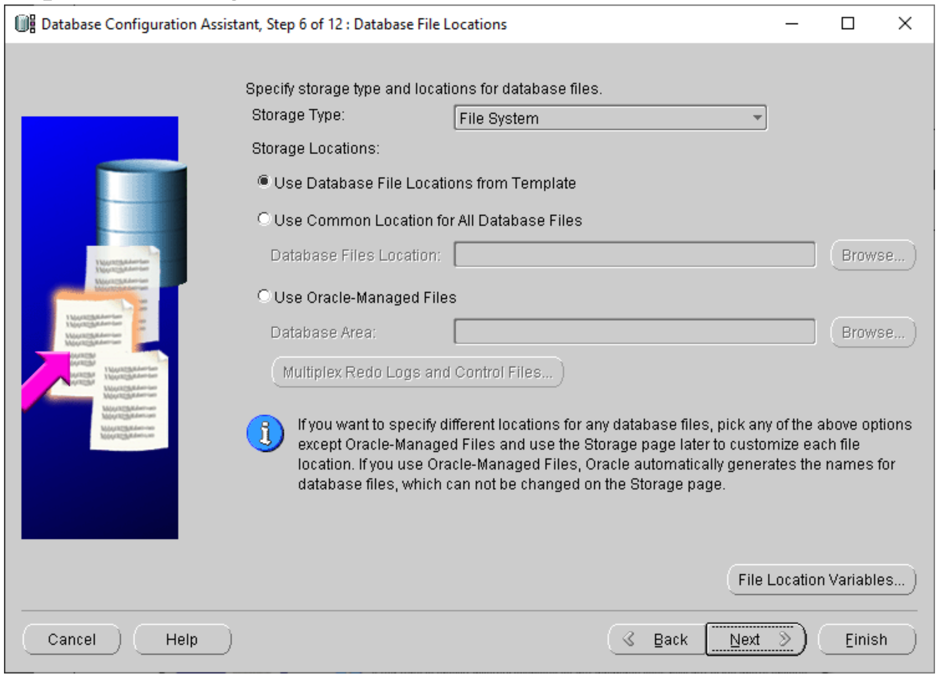
User Name	Password	Confirm Password
SYS		
SYSTEM		
DBSNMP		
SYSMAN		

(●) Use the Same Administrative Password for All Accounts

Password: ****

Confirm Password: ****

Step 7 : No changes Needed, Click on Next.



Database Configuration Assistant, Step 6 of 12 : Database File Locations

Specify storage type and locations for database files.

Storage Type: File System

Storage Locations:

(●) Use Database File Locations from Template

( ) Use Common Location for All Database Files

Database Files Location:

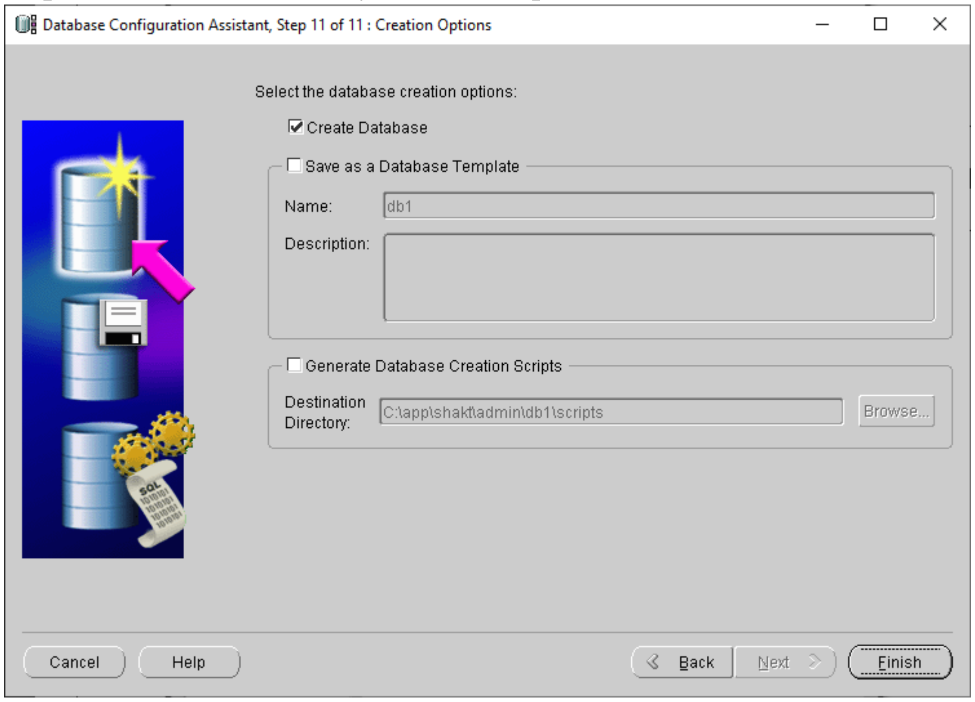
( ) Use Oracle-Managed Files

Database Area:

If you want to specify different locations for any database files, pick any of the above options except Oracle-Managed Files and use the Storage page later to customize each file location. If you use Oracle-Managed Files, Oracle automatically generates the names for database files, which can not be changed on the Storage page.

Browse...
Browse...
Multiplex Redo Logs and Control Files...
File Location Variables...
Cancel
Help
Back
Next
Finish

Step 8 : Click on next until you reach Step11 and Click Finish


Database Configuration Assistant, Step 11 of 11 : Creation Options

Select the database creation options:

[✓] Create Database

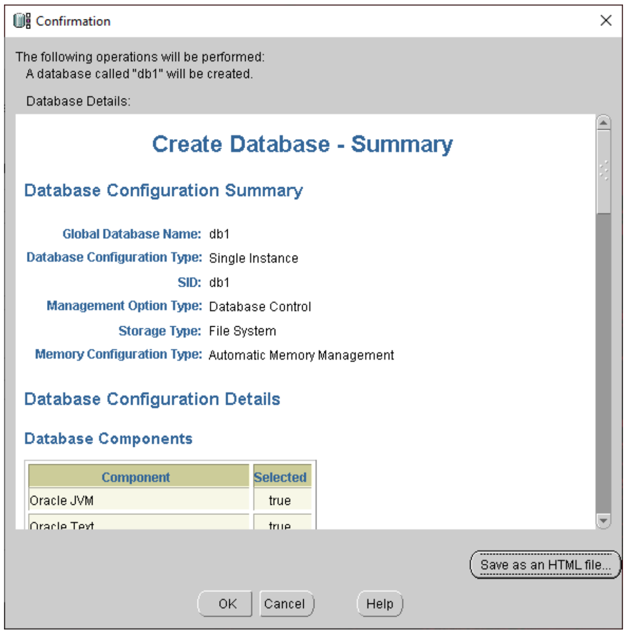
[ ] Save as a Database Template

Name: db1

Description:

[ ] Generate Database Creation Scripts

Destination Directory: C:\app\shakt\admin\db1\scripts


Confirmation

The following operations will be performed:
A database called "db1" will be created.

Database Details:

Create Database - Summary

Database Configuration Summary

Global Database Name:	db1
Database Configuration Type:	Single Instance
SID:	db1
Management Option Type:	Database Control
Storage Type:	File System
Memory Configuration Type:	Automatic Memory Management

Database Configuration Details

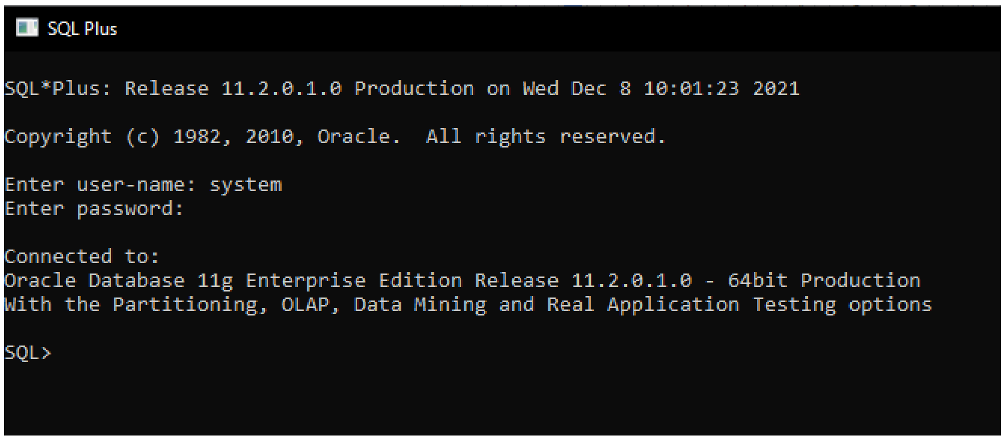
Database Components

Component	Selected
Oracle JVM	true
Oracle Text	true

After Clicking “OK” the database creating process will start and then Click on “Exit”

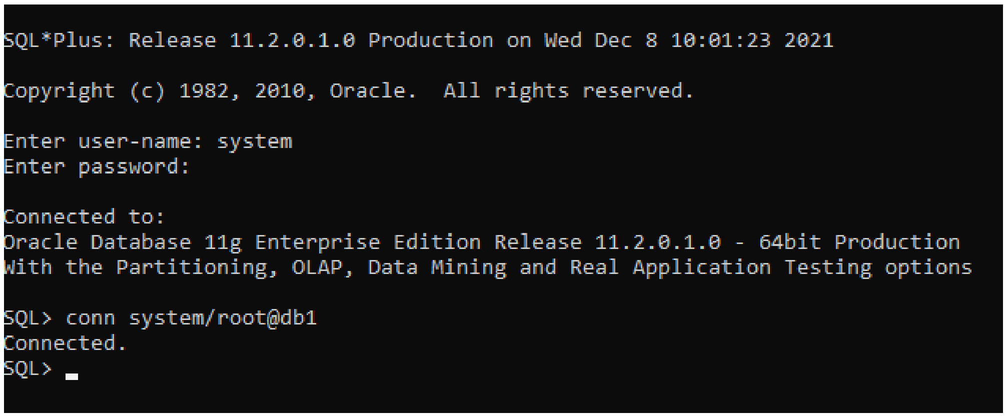
Practical Implementation Steps:

Step 1: Open SQLPlus and Connect to Your Database .



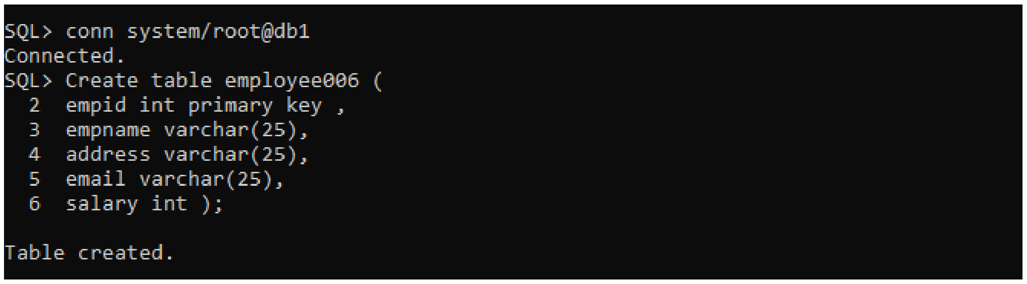
```
SQL Plus

SQL*Plus: Release 11.2.0.1.0 Production on Wed Dec 8 10:01:23 2021

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

Enter user-name: system
Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL>
```

Step 2 : Connect the Database



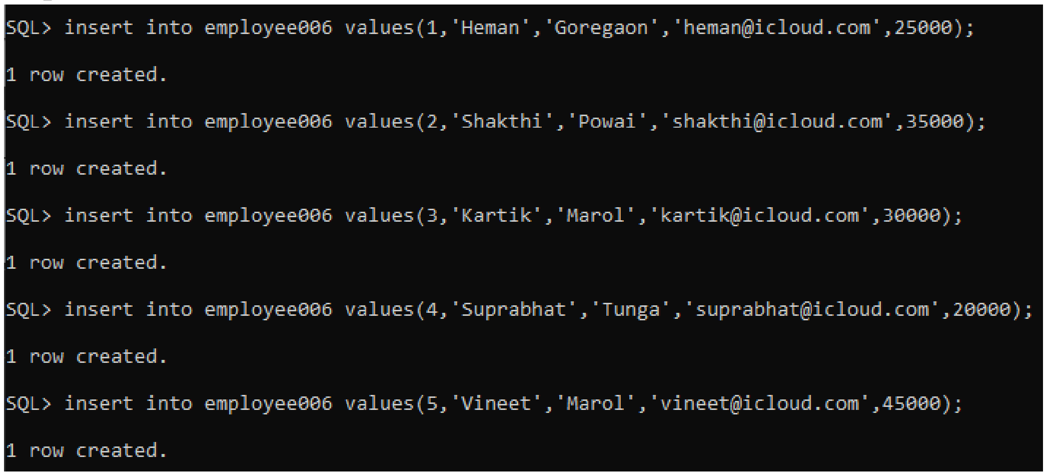
```
SQL*Plus: Release 11.2.0.1.0 Production on Wed Dec 8 10:01:23 2021

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

Enter user-name: system
Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> conn system/root@db1
Connected.
SQL>
```

Step 3 : Create a Table



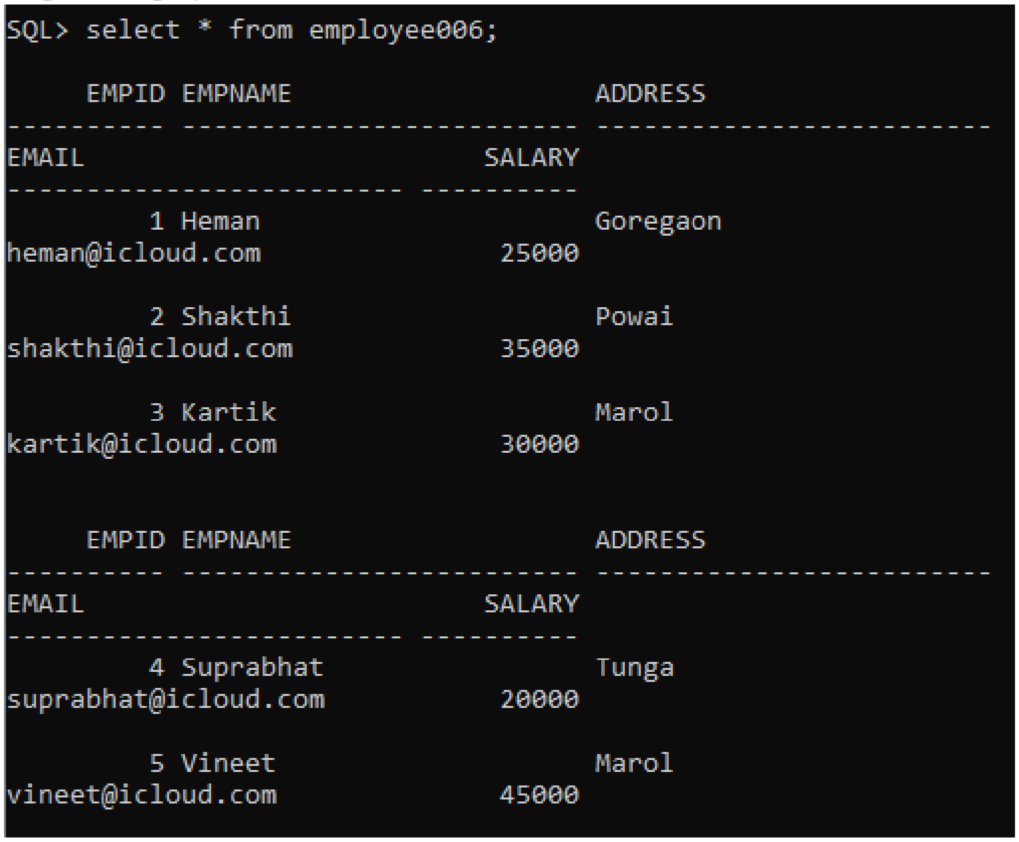
```
SQL> conn system/root@db1
Connected.
SQL> Create table employee006 (
  2  empid int primary key ,
  3  empname varchar(25),
  4  address varchar(25),
  5  email varchar(25),
  6  salary int );

Table created.
```

Step 4 : Insert values in the table



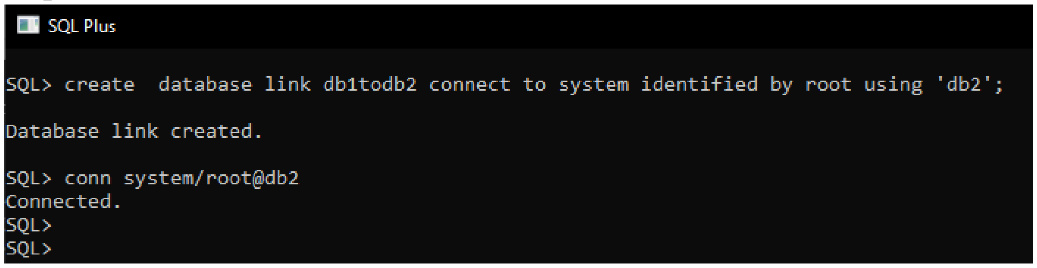
```
SQL> insert into employee006 values(1,'Heman','Goregaon','heman@icloud.com',25000);

1 row created.

SQL> insert into employee006 values(2,'Shakthi','Powai','shakthi@icloud.com',35000);

1 row created.

SQL> insert into employee006 values(3,'Kartik','Marol','kartik@icloud.com',30000);

1 row created.

SQL> insert into employee006 values(4,'Suprabhat','Tunga','suprabhat@icloud.com',20000);

1 row created.

SQL> insert into employee006 values(5,'Vineet','Marol','vineet@icloud.com',45000);

1 row created.
```

Step 5 : Display the inserted values



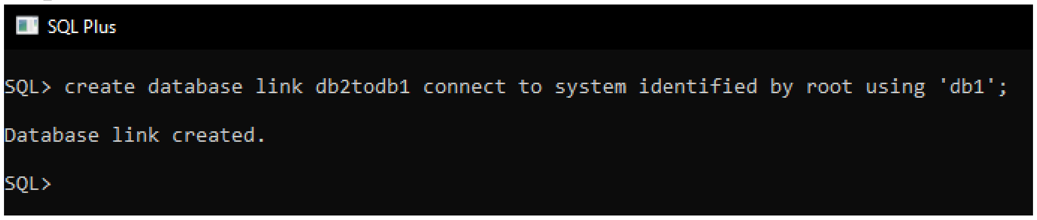
```
SQL> select * from employee006;

     EMPID EMPNAME                  ADDRESS
---------- ------------------------ -------------------------
EMAIL                         SALARY
------------------------- ----------
         1 Heman                    Goregaon
heman@icloud.com               25000

         2 Shakthi                  Powai
shakthi@icloud.com             35000

         3 Kartik                   Marol
kartik@icloud.com              30000


     EMPID EMPNAME                  ADDRESS
---------- ------------------------ -------------------------
EMAIL                         SALARY
------------------------- ----------
         4 Suprabhat                Tunga
suprabhat@icloud.com           20000

         5 Vineet                   Marol
vineet@icloud.com              45000
```

Step 6 : Create a link between two databases and then connect to db2



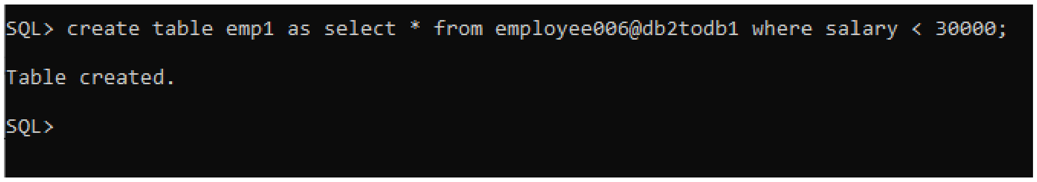
```
SQL Plus

SQL> create  database link db1todb2 connect to system identified by root using 'db2';

Database link created.

SQL> conn system/root@db2
Connected.
SQL>
SQL>
```

Step 7 : Create link to db1

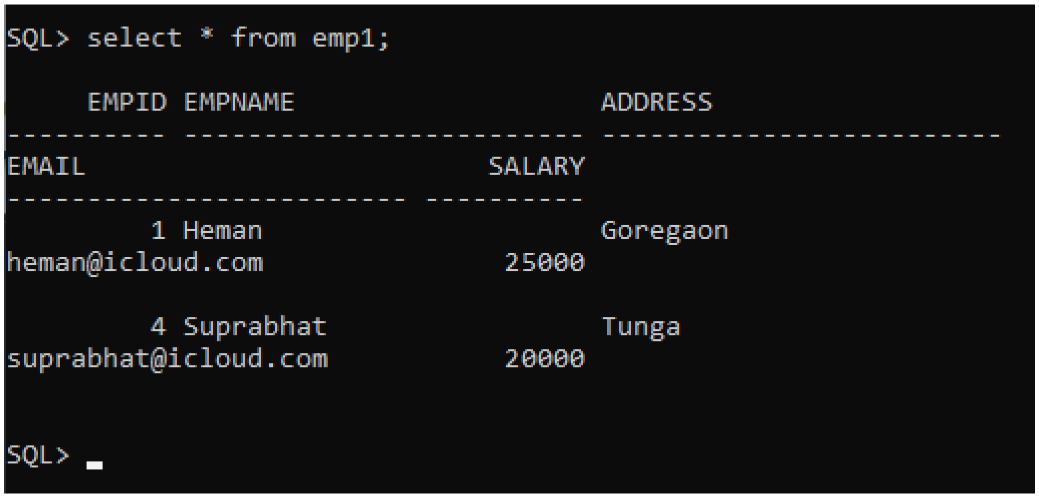


```
SQL Plus

SQL> create database link db2todb1 connect to system identified by root using 'db1';

Database link created.

SQL>
```

Step 8: Create emp1 select where salary is more than 30,000.

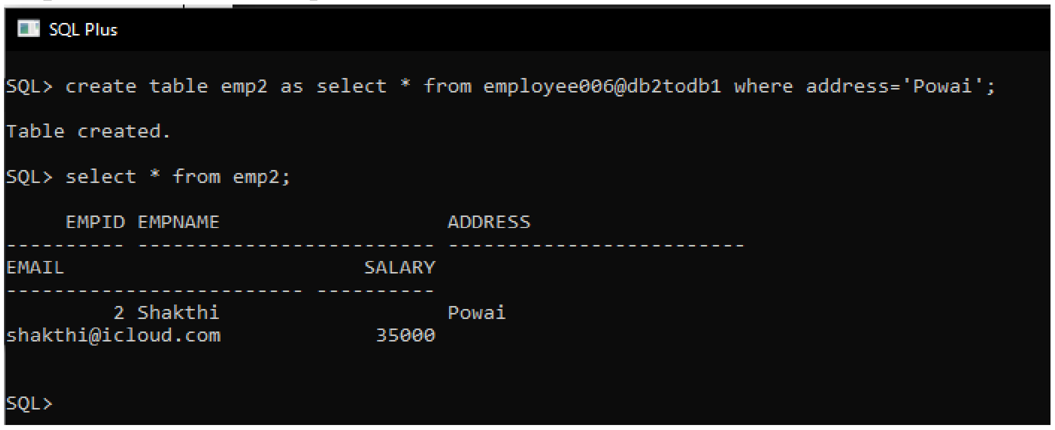


```
SQL> create table emp1 as select * from employee006@db2todb1 where salary < 30000;

Table created.

SQL>
```



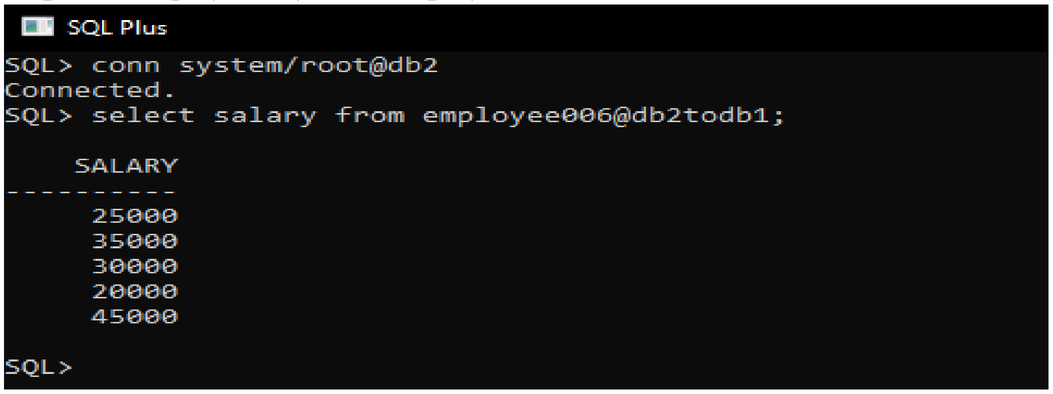
```
SQL> select * from emp1;

     EMPID EMPNAME                   ADDRESS
---------- ------------------------- -------------------------
EMAIL                         SALARY
------------------------- ----------
         1 Heman                     Goregaon
heman@icloud.com               25000

         4 Suprabhat                 Tunga
suprabhat@icloud.com           20000


SQL> _
```

Step 9 : Create table emp2 where address='Powai'.



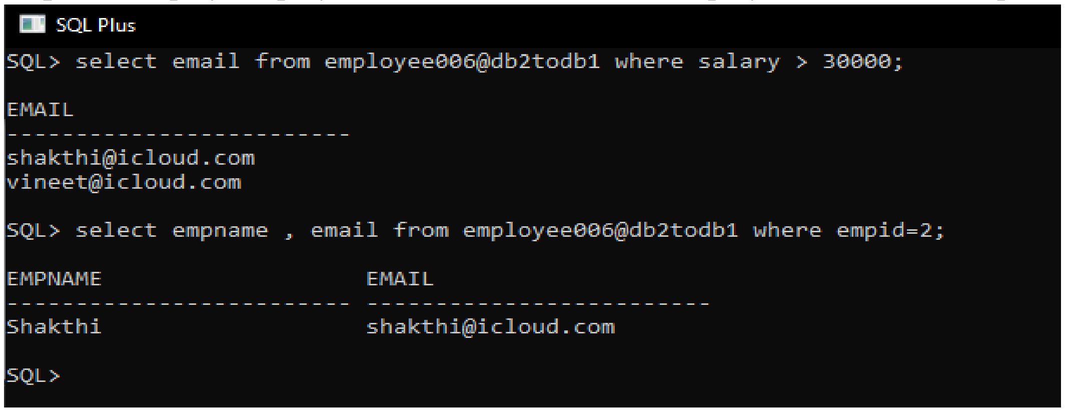
```
SQL Plus

SQL> create table emp2 as select * from employee006@db2todb1 where address='Powai';

Table created.

SQL> select * from emp2;

     EMPID EMPNAME                   ADDRESS
---------- ------------------------- -------------------------
EMAIL                          SALARY
------------------------- ----------
         2 Shakthi                   Powai
shakthi@icloud.com              35000


SQL>
```

Step 10 : Display salary from employee table



```
SQL Plus
SQL> conn system/root@db2
Connected.
SQL> select salary from employee006@db2todb1;

    SALARY
----------
     25000
     35000
     30000
     20000
     45000

SQL>
```

Step 11 : Display Employee Name and Email from Employee table where empid=2.



```
SQL Plus
SQL> select email from employee006@db2todb1 where salary > 30000;

EMAIL
-------------------------
shakthi@icloud.com
vineet@icloud.com

SQL> select empname , email from employee006@db2todb1 where empid=2;

EMPNAME                   EMAIL
------------------------- -------------------------
Shakthi                   shakthi@icloud.com

SQL>
```

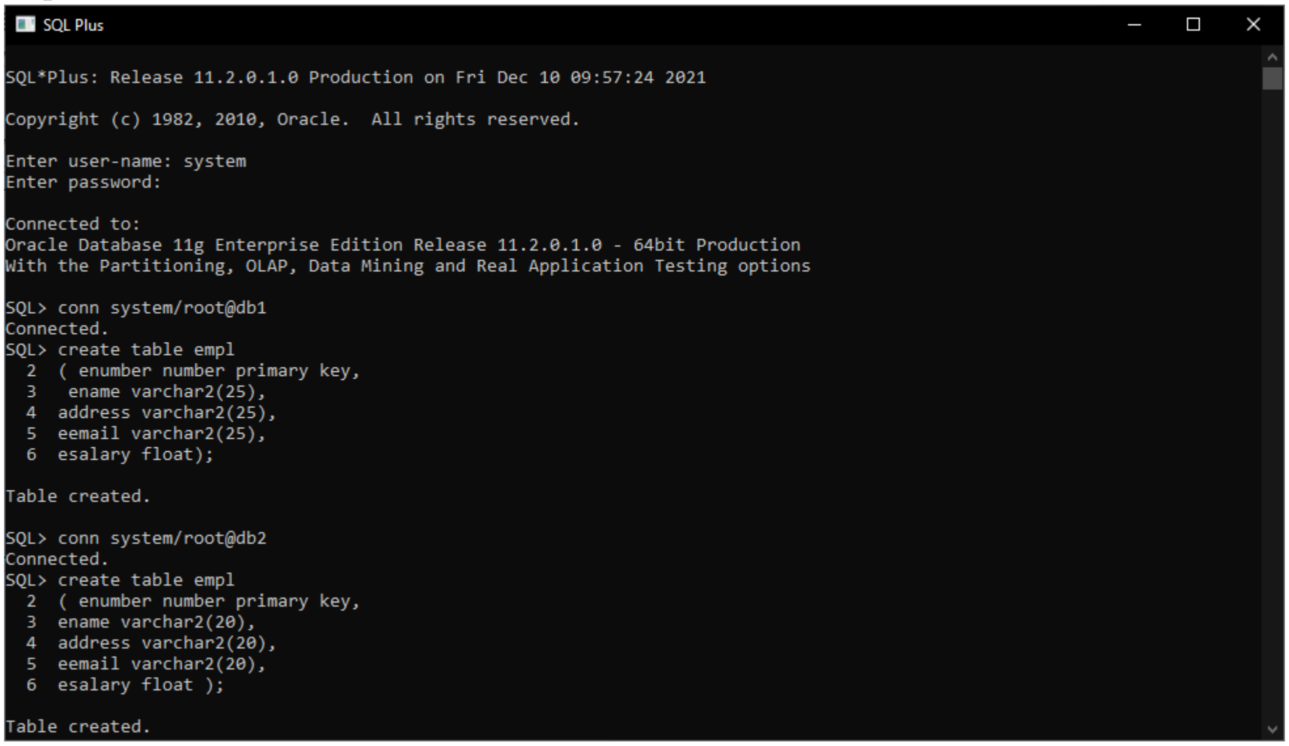
Conclusion: Successfully Execution of Schema into horizontal and vertical Fragmentation on different nodes in the Distributed Database Environment.

PRACTICAL NO: 2

Aim: Place the replication of global conceptual schema on different nodes and execute queries that will demonstrate a distributed database environment.

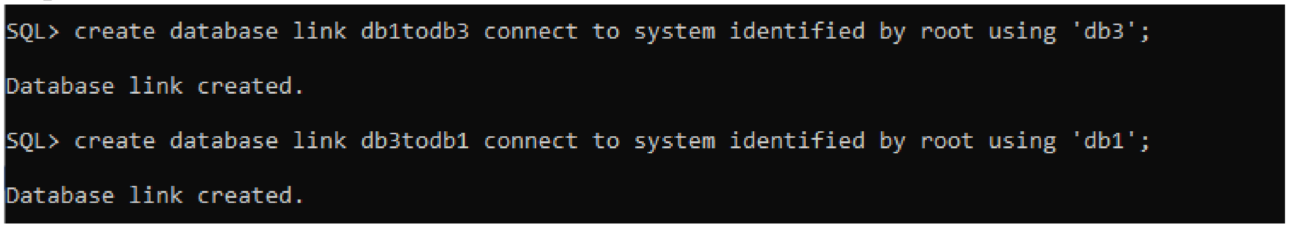
Software Requirement: Oracle 11g.
Query:
1. Update any record in db1 & show in db2
2. Delete any record in db1 & show in db2.
3. Find the salary of all employees.
4. Find the email of all employees where salary = 15000.
5. Find the employee name and email where the employee number is known.
6. Find the employee name and address where the employee number is known.

Step 1: Create a Table in both db1 and db2



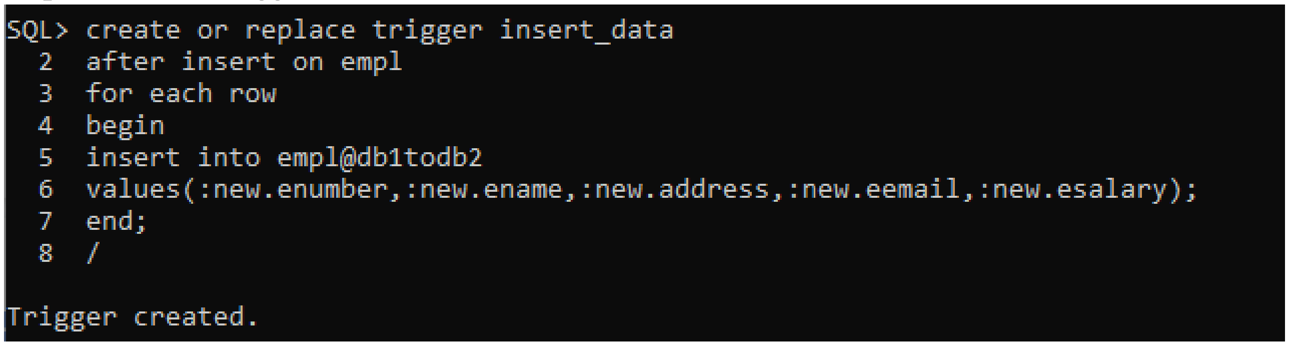
```
SQL*Plus: Release 11.2.0.1.0 Production on Fri Dec 10 09:57:24 2021

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

Enter user-name: system
Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> conn system/root@db1
Connected.
SQL> create table empl
  2  ( enumber number primary key,
  3   ename varchar2(25),
  4  address varchar2(25),
  5  eemail varchar2(25),
  6  esalary float);

Table created.

SQL> conn system/root@db2
Connected.
SQL> create table empl
  2  ( enumber number primary key,
  3  ename varchar2(20),
  4  address varchar2(20),
  5  eemail varchar2(20),
  6  esalary float );

Table created.
```

Step 2: Create Database link



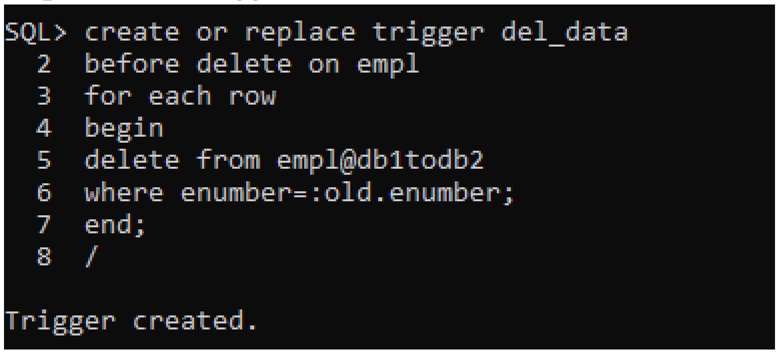
```
SQL> create database link db1todb3 connect to system identified by root using 'db3';

Database link created.

SQL> create database link db3todb1 connect to system identified by root using 'db1';

Database link created.
```

Step 3: Create a Trigger to insert data



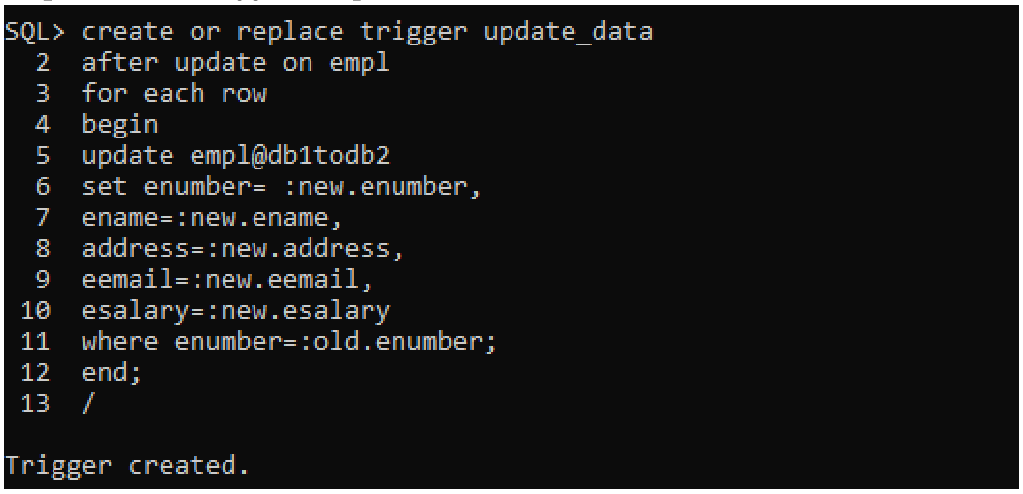
```
SQL> create or replace trigger insert_data
after insert on empl
for each row
begin
insert into empl@db1todb2
values(:new.enumber,:new.ename,:new.address,:new.eemail,:new.esalary);
end;
/

Trigger created.
```

Step 4: Create Trigger to Delete Data in Table.



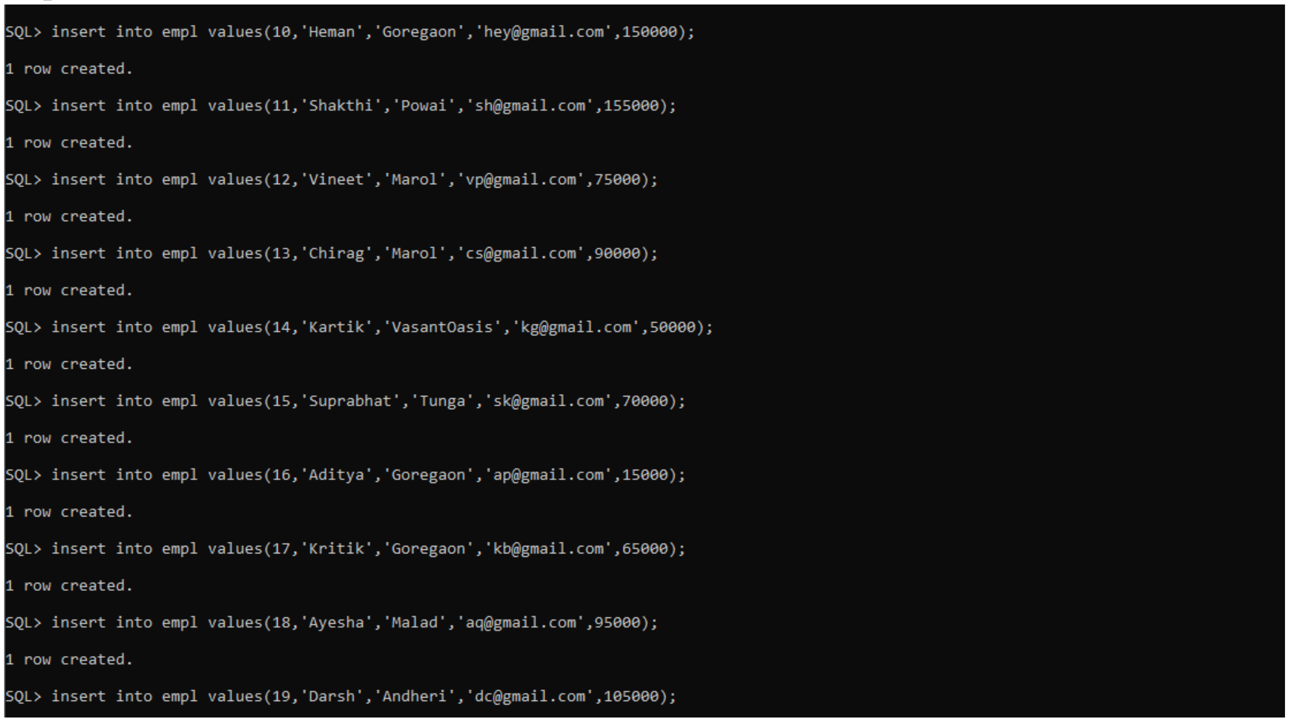
```
SQL> create or replace trigger del_data
before delete on empl
for each row
begin
delete from empl@db1todb2
where enumber=:old.enumber;
end;
/

Trigger created.
```

Step 5: Create Trigger to Update Data in Table.



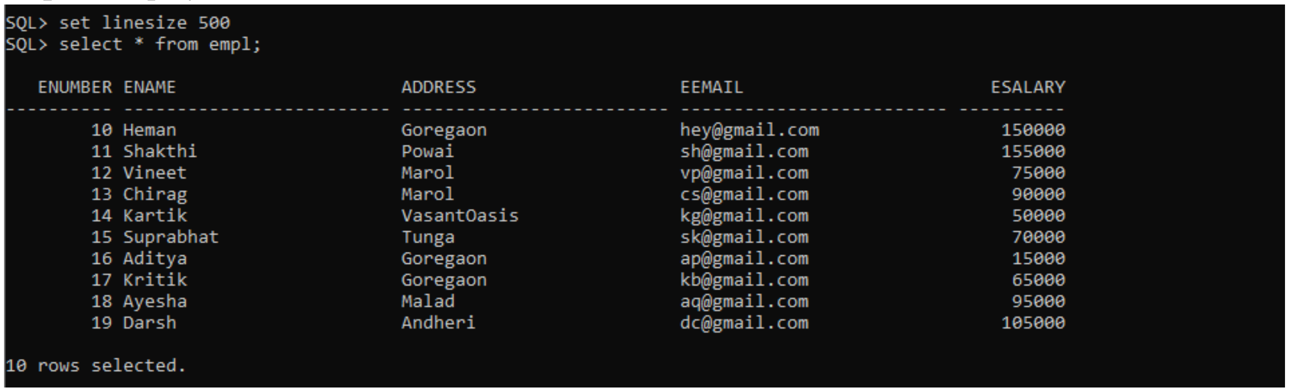
```
SQL> create or replace trigger update_data
after update on empl
for each row
begin
update empl@db1todb2
set enumber= :new.enumber,
ename=:new.ename,
address=:new.address,
eemail=:new.eemail,
esalary=:new.esalary
where enumber=:old.enumber;
end;
/

Trigger created.
```

Step 6: Insert data in the Table



```
SQL> insert into empl values(10,'Heman','Goregaon','hey@gmail.com',150000);

1 row created.

SQL> insert into empl values(11,'Shakthi','Powai','sh@gmail.com',155000);

1 row created.

SQL> insert into empl values(12,'Vineet','Marol','vp@gmail.com',75000);

1 row created.

SQL> insert into empl values(13,'Chirag','Marol','cs@gmail.com',90000);

1 row created.

SQL> insert into empl values(14,'Kartik','VasantOasis','kg@gmail.com',50000);

1 row created.

SQL> insert into empl values(15,'Suprabhat','Tunga','sk@gmail.com',70000);

1 row created.

SQL> insert into empl values(16,'Aditya','Goregaon','ap@gmail.com',15000);

1 row created.

SQL> insert into empl values(17,'Kritik','Goregaon','kb@gmail.com',65000);

1 row created.

SQL> insert into empl values(18,'Ayesha','Malad','aq@gmail.com',95000);

1 row created.

SQL> insert into empl values(19,'Darsh','Andheri','dc@gmail.com',105000);
```

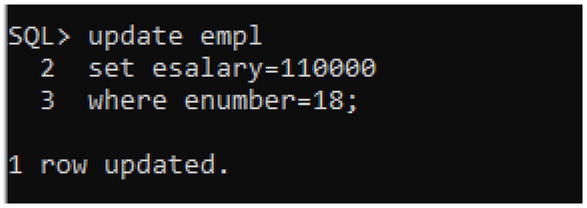
Step 7: Display inserted records



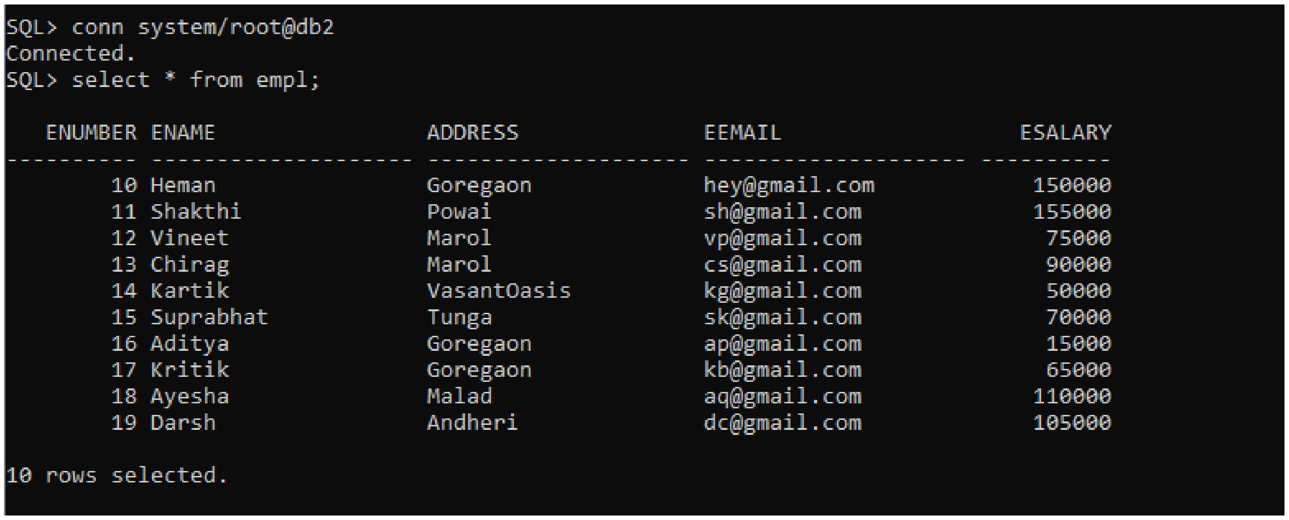
```
SQL> set linesize 500
SQL> select * from empl;

   ENUMBER ENAME                     ADDRESS                   EEMAIL                       ESALARY
---------- ------------------------- ------------------------- ------------------------- ----------
        10 Heman                     Goregaon                  hey@gmail.com                 150000
        11 Shakthi                   Powai                     sh@gmail.com                  155000
        12 Vineet                    Marol                     vp@gmail.com                   75000
        13 Chirag                    Marol                     cs@gmail.com                   90000
        14 Kartik                    VasantOasis               kg@gmail.com                   50000
        15 Suprabhat                 Tunga                     sk@gmail.com                   70000
        16 Aditya                    Goregaon                  ap@gmail.com                   15000
        17 Kritik                    Goregaon                  kb@gmail.com                   65000
        18 Ayesha                    Malad                     aq@gmail.com                   95000
        19 Darsh                     Andheri                   dc@gmail.com                  105000

10 rows selected.
```

QUERY

1. Update any record in db1 & show in db2.

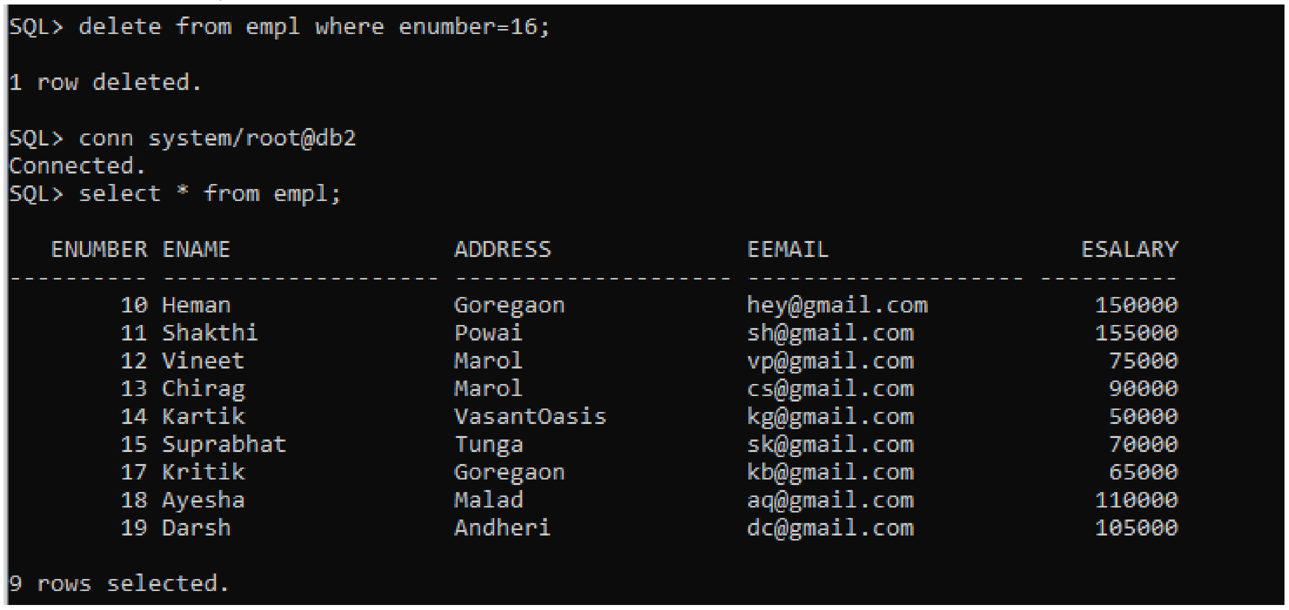


```
SQL> update empl
  2  set esalary=110000
  3  where enumber=18;

1 row updated.
```



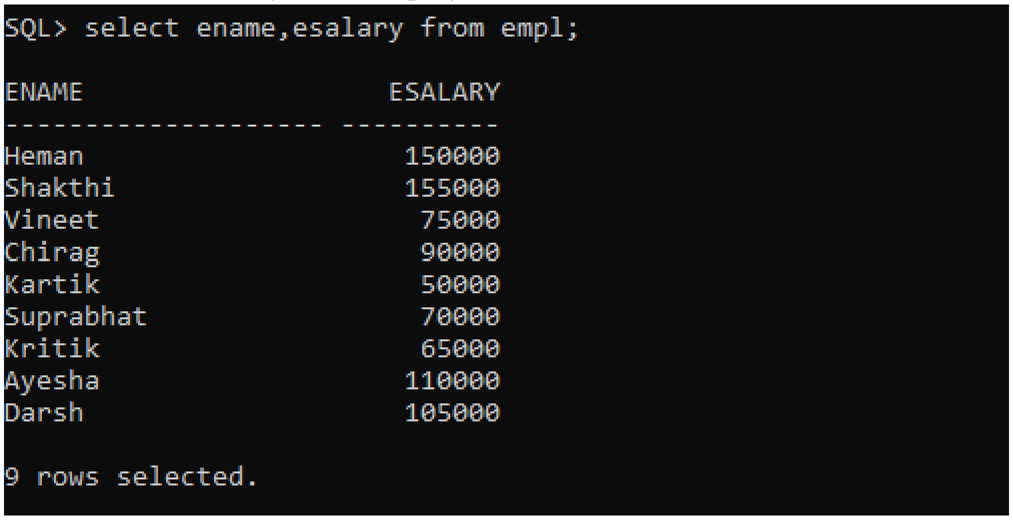
```
SQL> conn system/root@db2
Connected.
SQL> select * from empl;

   ENUMBER ENAME                ADDRESS              EEMAIL                  ESALARY
---------- -------------------- -------------------- -------------------- ----------
        10 Heman                Goregaon             hey@gmail.com            150000
        11 Shakthi              Powai                sh@gmail.com             155000
        12 Vineet               Marol                vp@gmail.com              75000
        13 Chirag               Marol                cs@gmail.com              90000
        14 Kartik               VasantOasis          kg@gmail.com              50000
        15 Suprabhat            Tunga                sk@gmail.com              70000
        16 Aditya               Goregaon             ap@gmail.com              15000
        17 Kritik               Goregaon             kb@gmail.com              65000
        18 Ayesha               Malad                aq@gmail.com             110000
        19 Darsh                Andheri              dc@gmail.com             105000

10 rows selected.
```

2. Delete any record in db1 & show in db2



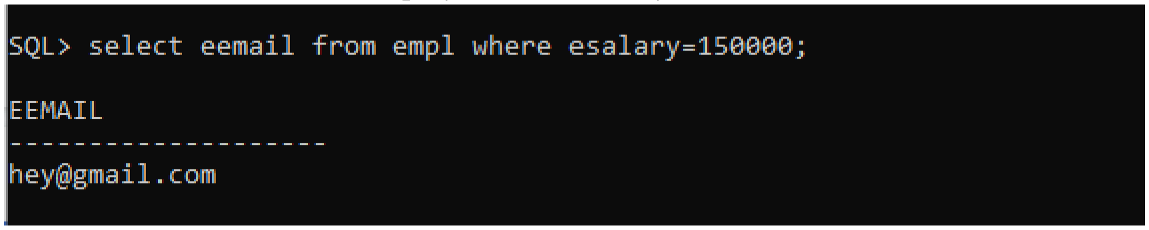
```
SQL> delete from empl where enumber=16;

1 row deleted.

SQL> conn system/root@db2
Connected.
SQL> select * from empl;

   ENUMBER ENAME                ADDRESS              EEMAIL                  ESALARY
---------- -------------------- -------------------- -------------------- ----------
        10 Heman                Goregaon             hey@gmail.com            150000
        11 Shakthi              Powai                sh@gmail.com             155000
        12 Vineet               Marol                vp@gmail.com              75000
        13 Chirag               Marol                cs@gmail.com              90000
        14 Kartik               VasantOasis          kg@gmail.com              50000
        15 Suprabhat            Tunga                sk@gmail.com              70000
        17 Kritik               Goregaon             kb@gmail.com              65000
        18 Ayesha               Malad                aq@gmail.com             110000
        19 Darsh                Andheri              dc@gmail.com             105000

9 rows selected.
```

3. Find the salary of all employees.



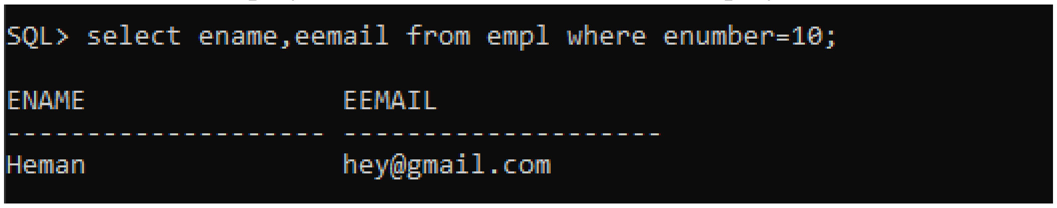
```
SQL> select ename,esalary from empl;

ENAME                    ESALARY
-------------------- ----------
Heman                    150000
Shakthi                  155000
Vineet                    75000
Chirag                    90000
Kartik                    50000
Suprabhat                 70000
Kritik                    65000
Ayesha                   110000
Darsh                    105000

9 rows selected.
```

4. Find the email of all employees where salary = 150000.



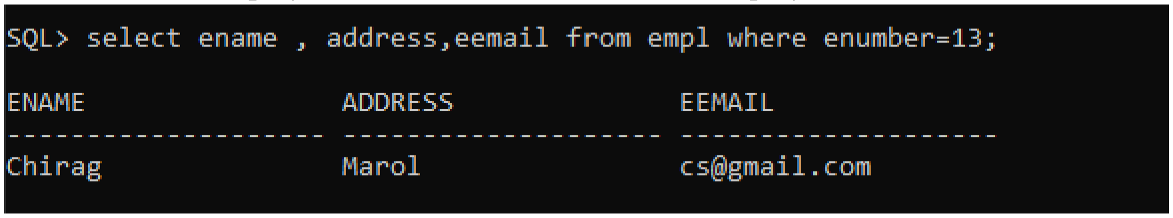
```
SQL> select eemail from empl where esalary=150000;

EEMAIL
--------------------
hey@gmail.com
```

5. Find the employee name and email where the employee number is known.



```
SQL> select ename,eemail from empl where enumber=10;

ENAME                EEMAIL
-------------------- --------------------
Heman                hey@gmail.com
```

6. Find the employee name and address where the employee number is known.



```
SQL> select ename , address,eemail from empl where enumber=13;

ENAME                ADDRESS              EEMAIL
-------------------- -------------------- --------------------
Chirag               Marol                cs@gmail.com
```

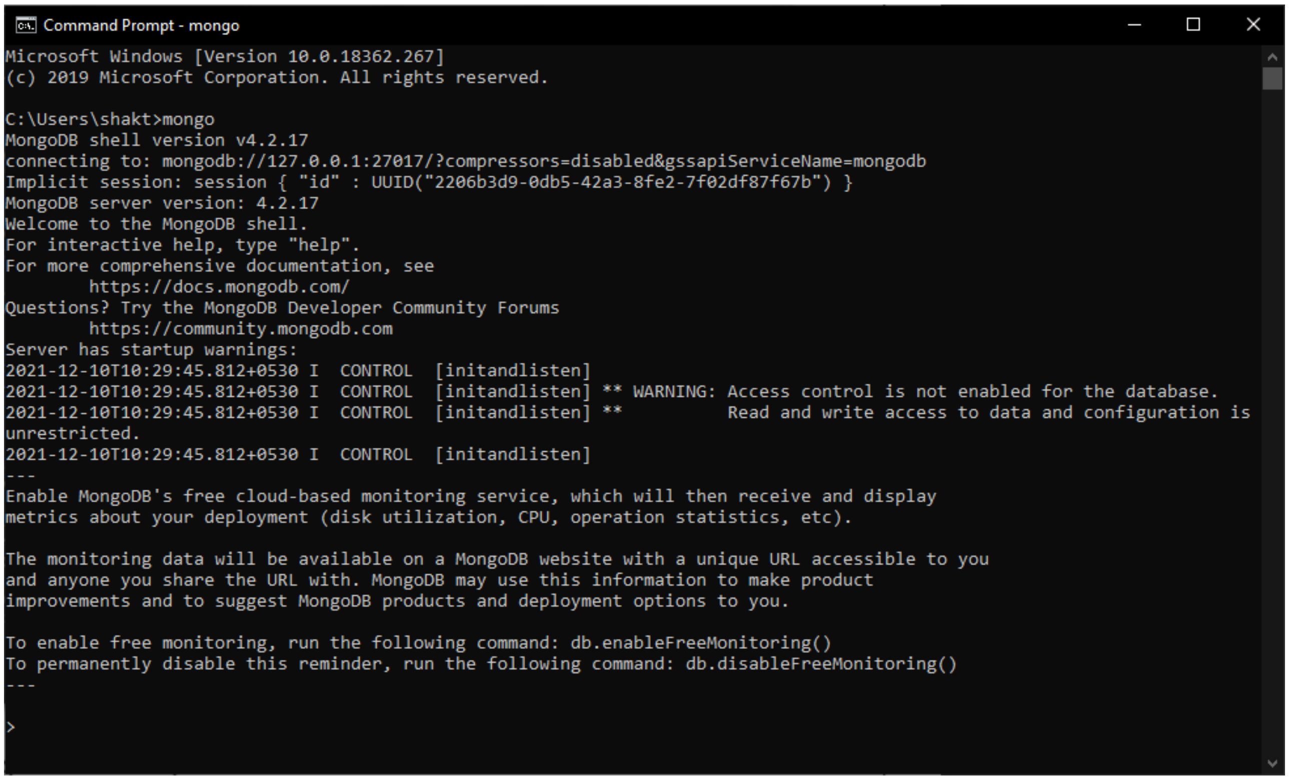
Conclusion: Successfully Created Triggers and Performed Different Queries on them.

PRACTICAL NO: 3

Aim: To perform CRUD Operation using MongoDB.
Software Requirement: MongoDB.
Practical Implementation Steps :
Step 1:

- Open CMD and hit command "Mongo" [To directly run MongoDB from Command Prompt we need to First Set the Environment Variable for MongoDB].

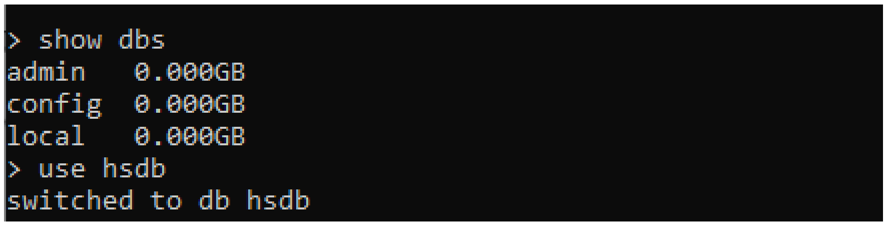
- To set Environment Variable Follow the Steps:
  Open C drive -> Program Files -> MongoDB -> server -> 5.0 -> bin
  C:\Program Files\MongoDB\Server\5.0\bin [Copy the Path].
  Start -> Search For "Edit the System Environment Variable" -> Open.
  Add the Copied Path in System Variable and done.



```
Microsoft Windows [Version 10.0.18362.267]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\shakt>mongo
MongoDB shell version v4.2.17
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("2206b3d9-0db5-42a3-8fe2-7f02df87f67b") }
MongoDB server version: 4.2.17
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
        https://community.mongodb.com
Server has startup warnings:
2021-12-10T10:29:45.812+0530 I  CONTROL  [initandlisten]
2021-12-10T10:29:45.812+0530 I  CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2021-12-10T10:29:45.812+0530 I  CONTROL  [initandlisten] **          Read and write access to data and configuration is
unrestricted.
2021-12-10T10:29:45.812+0530 I  CONTROL  [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---

>
```

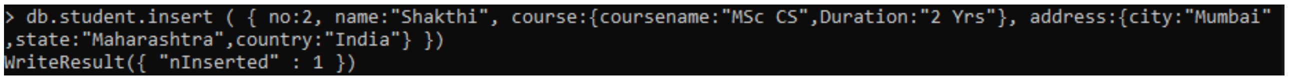
Step 2: Creating and selecting database

Command : use hsdb [hsdb is Database Name].
Note: To list all Databases use the command : Show dbs.



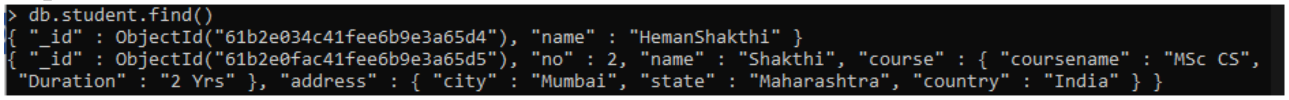
```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> use hsdb
switched to db hsdb
```

Step 3:
Creating Collections and Inserting Values [C - Create]
Creating a collection and inserting values can be done together. Here we have our collection name as ‘student ‘.



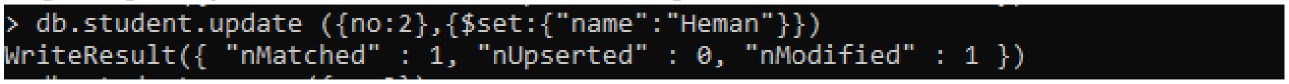
```
> db.student.insert ( { no:2, name:"Shakthi", course:{coursename:"MSc CS",Duration:"2 Yrs"}, address:{city:"Mumbai"
,state:"Maharashtra",country:"India"} })
WriteResult({ "nInserted" : 1 })
```

Step 4: Read Data from the Collections [R - Read] To retrieve the inserted document.



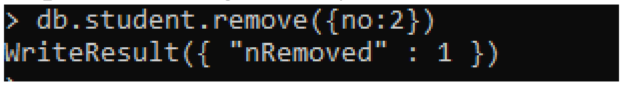
```
> db.student.find()
{ "_id" : ObjectId("61b2e034c41fee6b9e3a65d4"), "name" : "HemanShakthi" }
{ "_id" : ObjectId("61b2e0fac41fee6b9e3a65d5"), "no" : 2, "name" : "Shakthi", "course" : { "coursename" : "MSc CS",
 "Duration" : "2 Yrs" }, "address" : { "city" : "Mumbai", "state" : "Maharashtra", "country" : "India" } }
```

Step 5: Updating a Document in a Collection [U - Update].



```
> db.student.update ({no:2},{$set:{"name":"Heman"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Step 6: Removing an Entry From the Collection[D- Delete].



```
> db.student.remove({no:2})
WriteResult({ "nRemoved" : 1 })
```

Conclusion: Successfully Performed and Implemented the CRUD Operation Using MongoDB.

PRACTICAL NO: 4

Aim: Create different types that include attributes and methods. Define tables for these types by adding sufficient number of tuples. Demonstrate insert, update and delete operations on these tables. Execute queries on them.
Software Requirement: Oracle 11g.
Steps:

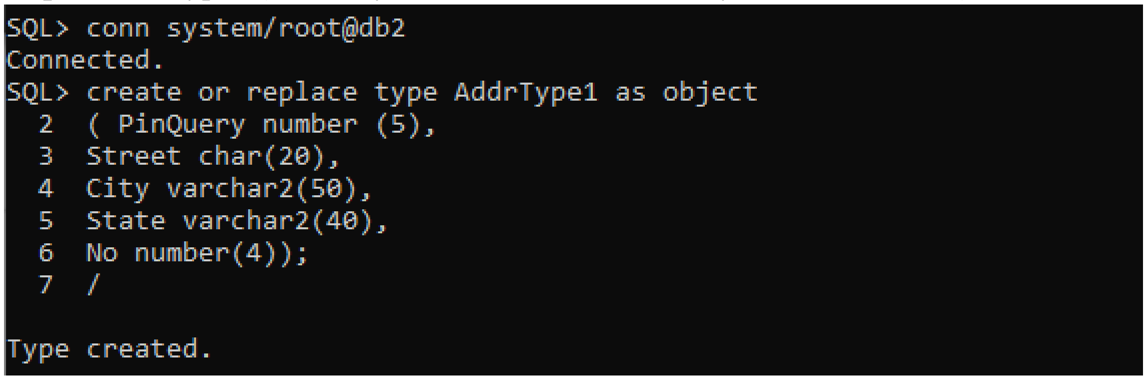
1. AddrType1 (PinQuery: number, Street :char, City : char, state :char) .
2. BranchType (address: AddrType1, phone1: integer,phone2: integer).
3. AuthorType (name:char,,addr AddrType1).
4. PublisherType (name: char, addr: AddrType1, branches: BranchTableType.
5. books(title: varchar, year : date, published_by ref PublisherType,authorsAuthorListType).
6. Insert some records into the above tables and fire the following queries.

Query:

1. List all of the authors that have the same pin Query as their publisher.
2. List all books that have 2 or more authors.
3. List the name of the publisher that has the most branches.
4. List all authors who have published more than one Book.
5. List all books (title) where the same author appears more than once on the list of authors(assuming that an integrity constraint requiring that the name of an author is unique in a list of authors has not been specified).

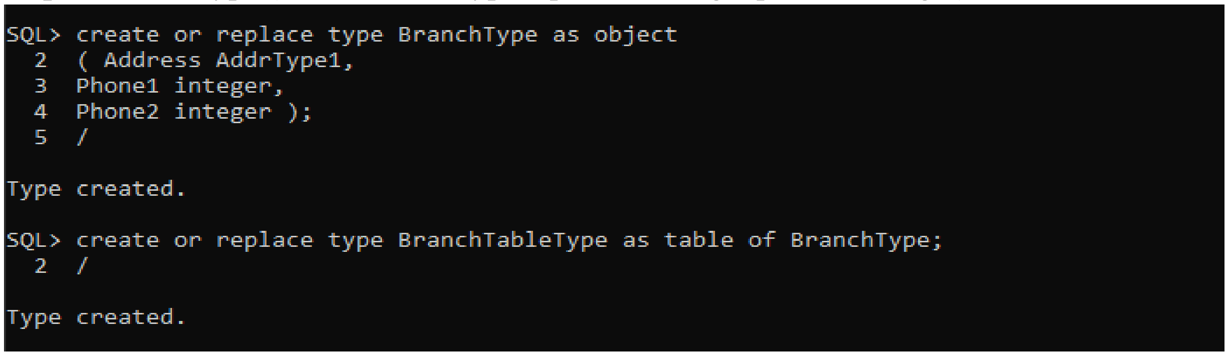
Practical Implementation Steps :

Step 1: AddrType1 (PinQuery: number, Street :char, City : char, state :char).



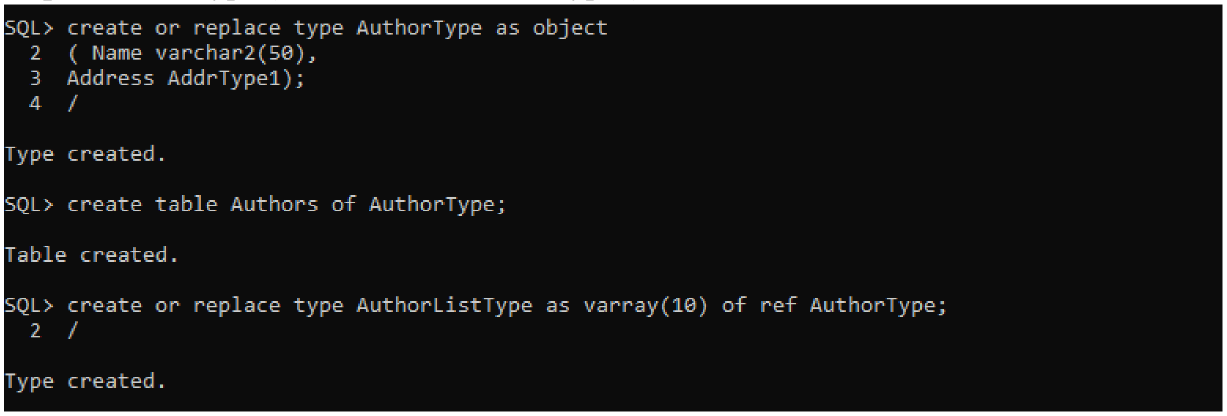
```
SQL> conn system/root@db2
Connected.
SQL> create or replace type AddrType1 as object
  2  ( PinQuery number (5),
  3  Street char(20),
  4  City varchar2(50),
  5  State varchar2(40),
  6  No number(4));
  7  /

Type created.
```

Step 2: BranchType (address: AddrType1, phone1: integer,phone2: integer).



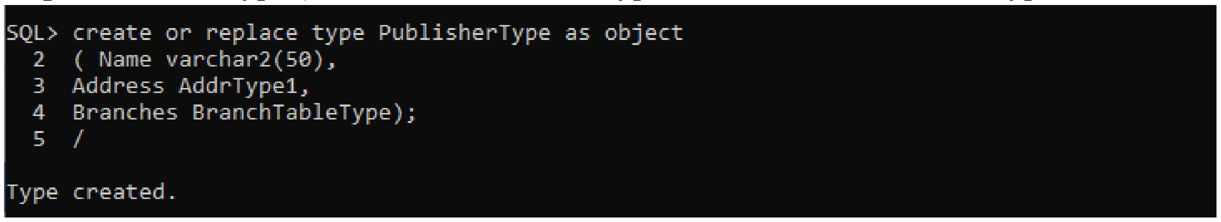
```
SQL> create or replace type BranchType as object
  2  ( Address AddrType1,
  3  Phone1 integer,
  4  Phone2 integer );
  5  /

Type created.

SQL> create or replace type BranchTableType as table of BranchType;
  2  /

Type created.
```

Step 3: AuthorType (name:char,,addr AddrType1).

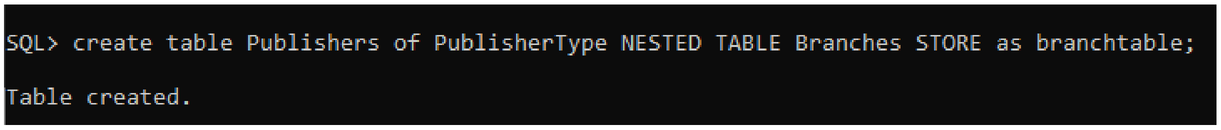


```
SQL> create or replace type AuthorType as object
  2  ( Name varchar2(50),
  3  Address AddrType1);
  4  /

Type created.

SQL> create table Authors of AuthorType;

Table created.

SQL> create or replace type AuthorListType as varray(10) of ref AuthorType;
  2  /

Type created.
```

Step 4: PublisherType (name: char, addr: AddrType1, branches: BranchTableType.

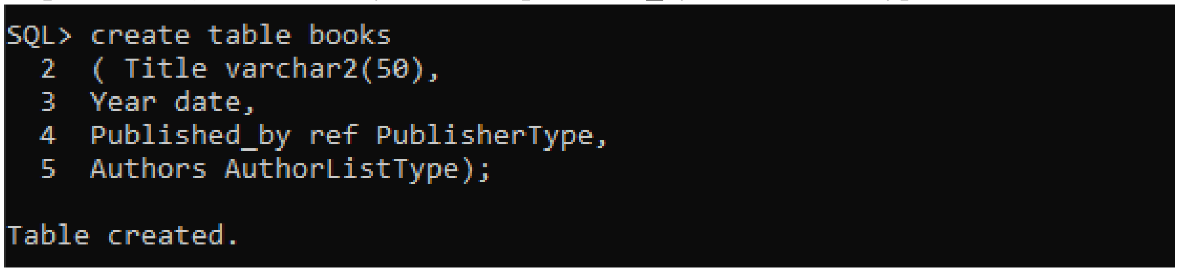


```
SQL> create or replace type PublisherType as object
  2  ( Name varchar2(50),
  3  Address AddrType1,
  4  Branches BranchTableType);
  5  /

Type created.
```



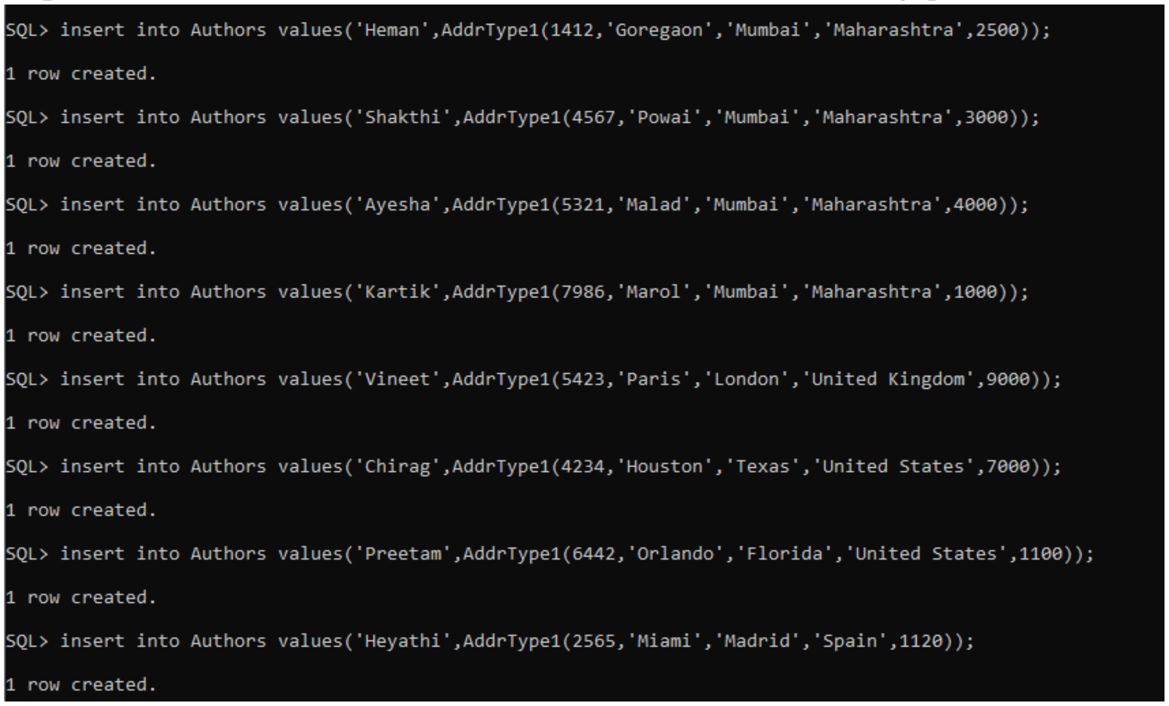
```
SQL> create table Publishers of PublisherType NESTED TABLE Branches STORE as branchtable;

Table created.
```

Step 5: Books(title: varchar, year : date, published_by ref PublisherType,authorsAuthorListType).



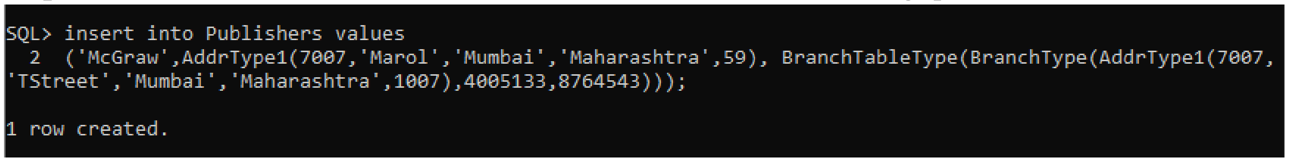
```
SQL> create table books
( Title varchar2(50),
Year date,
Published_by ref PublisherType,
Authors AuthorListType);

Table created.
```

Step 6: Insert some records into the above tables and fire the following queries.

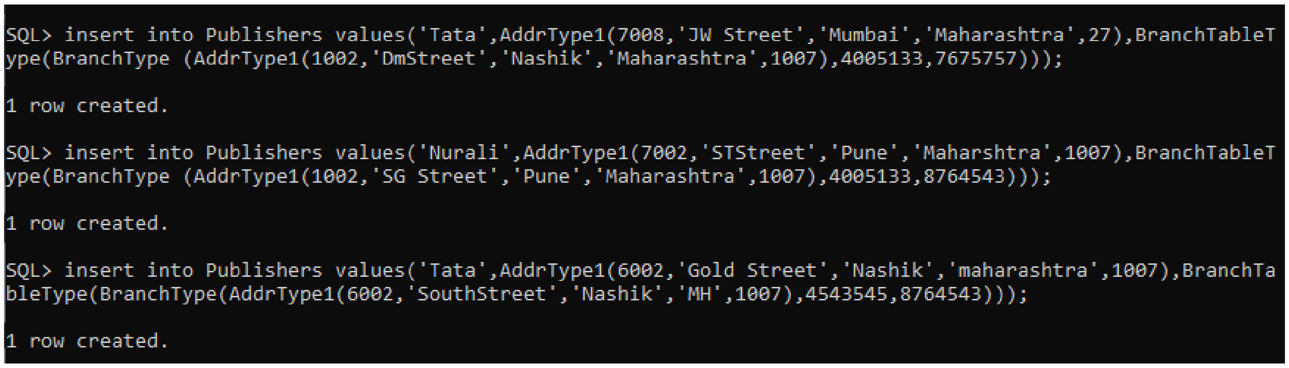


```
SQL> insert into Authors values('Heman',AddrType1(1412,'Goregaon','Mumbai','Maharashtra',2500));

1 row created.

SQL> insert into Authors values('Shakthi',AddrType1(4567,'Powai','Mumbai','Maharashtra',3000));

1 row created.

SQL> insert into Authors values('Ayesha',AddrType1(5321,'Malad','Mumbai','Maharashtra',4000));

1 row created.

SQL> insert into Authors values('Kartik',AddrType1(7986,'Marol','Mumbai','Maharashtra',1000));

1 row created.

SQL> insert into Authors values('Vineet',AddrType1(5423,'Paris','London','United Kingdom',9000));

1 row created.

SQL> insert into Authors values('Chirag',AddrType1(4234,'Houston','Texas','United States',7000));

1 row created.

SQL> insert into Authors values('Preetam',AddrType1(6442,'Orlando','Florida','United States',1100));

1 row created.

SQL> insert into Authors values('Heyathi',AddrType1(2565,'Miami','Madrid','Spain',1120));

1 row created.
```

Step 7: Insert Some records into the above tables and fire the following queries.

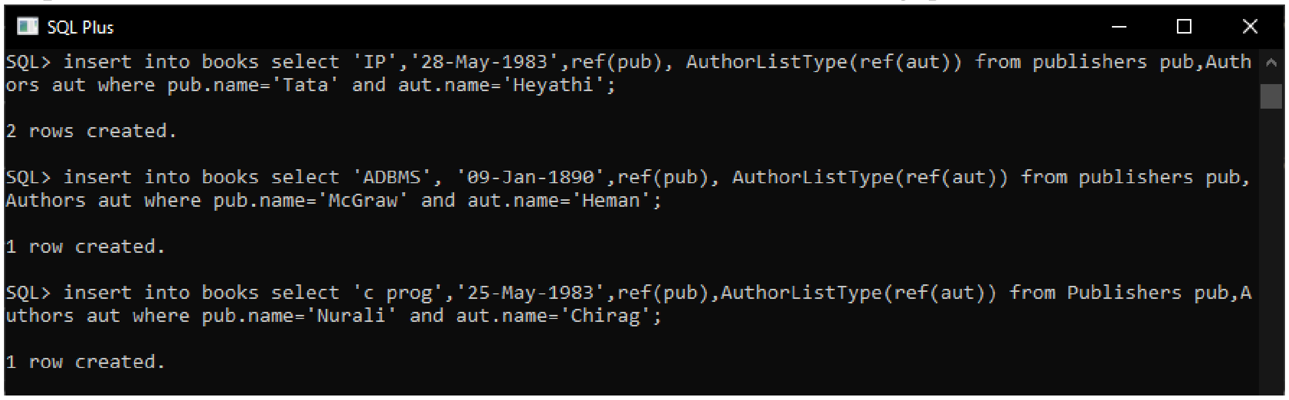


```
SQL> insert into Publishers values
('McGraw',AddrType1(7007,'Marol','Mumbai','Maharashtra',59), BranchTableType(BranchType(AddrType1(7007,
'TStreet','Mumbai','Maharashtra',1007),4005133,8764543)));

1 row created.
```



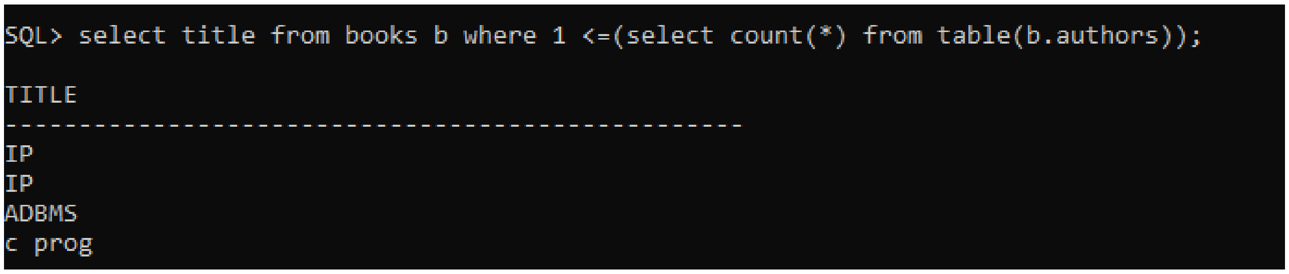
```
SQL> insert into Publishers values('Tata',AddrType1(7008,'JW Street','Mumbai','Maharashtra',27),BranchTableT
ype(BranchType (AddrType1(1002,'DmStreet','Nashik','Maharashtra',1007),4005133,7675757)));

1 row created.

SQL> insert into Publishers values('Nurali',AddrType1(7002,'STStreet','Pune','Maharshtra',1007),BranchTableT
ype(BranchType (AddrType1(1002,'SG Street','Pune','Maharashtra',1007),4005133,8764543)));

1 row created.

SQL> insert into Publishers values('Tata',AddrType1(6002,'Gold Street','Nashik','maharashtra',1007),BranchTa
bleType(BranchType(AddrType1(6002,'SouthStreet','Nashik','MH',1007),4543545,8764543)));

1 row created.
```

Step 8: Insert some records into the above tables and fire the following queries



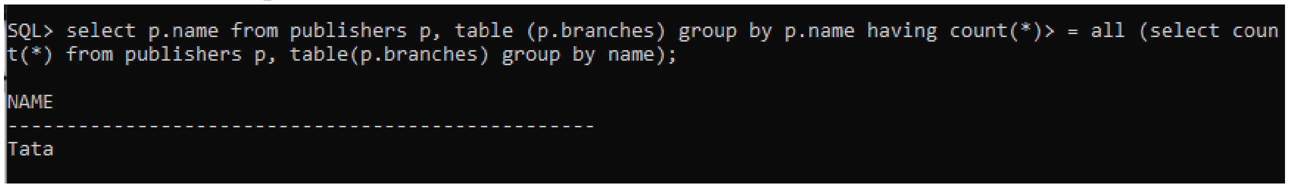
```
SQL> insert into books select 'IP','28-May-1983',ref(pub), AuthorListType(ref(aut)) from publishers pub,Auth
ors aut where pub.name='Tata' and aut.name='Heyathi';

2 rows created.

SQL> insert into books select 'ADBMS', '09-Jan-1890',ref(pub), AuthorListType(ref(aut)) from publishers pub,
Authors aut where pub.name='McGraw' and aut.name='Heman';

1 row created.

SQL> insert into books select 'c prog','25-May-1983',ref(pub),AuthorListType(ref(aut)) from Publishers pub,A
uthors aut where pub.name='Nurali' and aut.name='Chirag';

1 row created.
```

List all books that have 2 or more authors



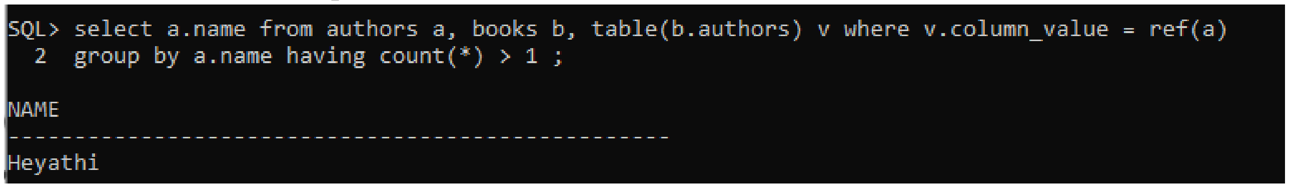
```
SQL> select title from books b where 1 <=(select count(*) from table(b.authors));

TITLE
--------------------------------------------------
IP
IP
ADBMS
c prog
```

List the name of the publisher that has the most branches



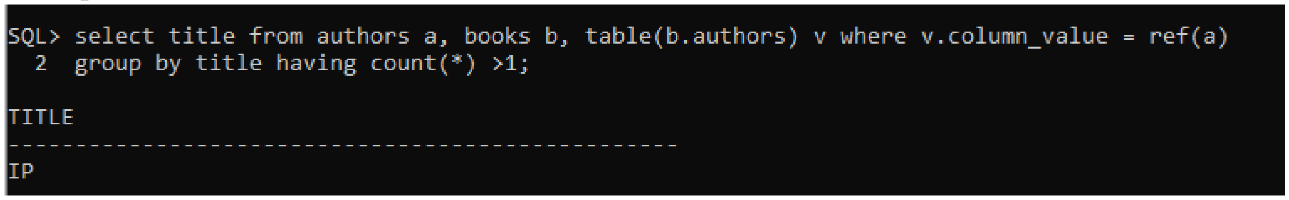
```
SQL> select p.name from publishers p, table (p.branches) group by p.name having count(*)> = all (select coun
t(*) from publishers p, table(p.branches) group by name);

NAME
--------------------------------------------------
Tata
```

List all authors who have published more than one Book



```
SQL> select a.name from authors a, books b, table(b.authors) v where v.column_value = ref(a)
  2  group by a.name having count(*) > 1 ;

NAME
--------------------------------------------------
Heyathi
```

List all books (title) where the same author appears more than once on the list of authors (assuming that an integrity constraint requiring that the name of an author is unique in a list of authors has not been specified).



```
SQL> select title from authors a, books b, table(b.authors) v where v.column_value = ref(a)
  2  group by title having count(*) >1;

TITLE
--------------------------------------------------
IP
```

Conclusion : Successfully Demonstrated insert, update and delete operations on Type

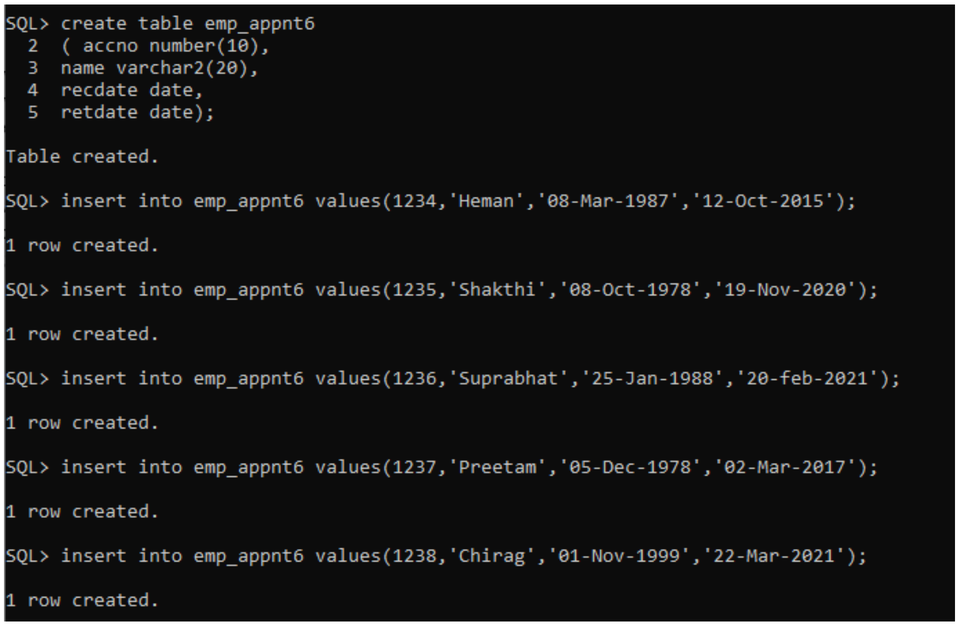
PRACTICAL NO: 5

Aim: Create a temporal database and issue queries on it.

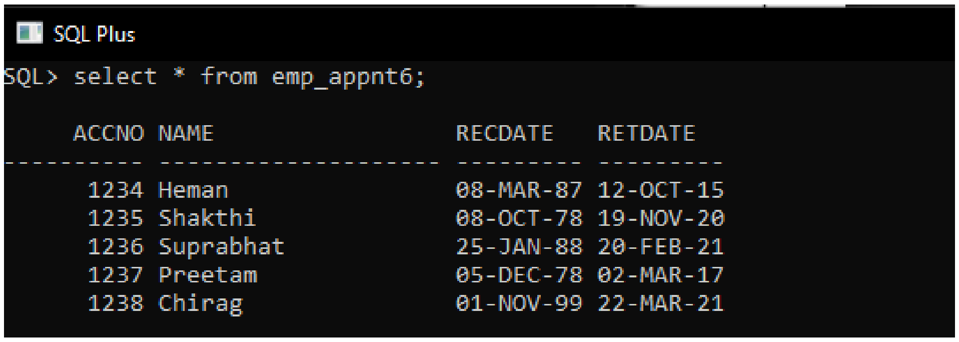
Software Requirement: MongoDB.

Query:

1. Show the Employee Whose Record Date is 08-Mar-1987.
2. Show the Employee Whose Retired Date is 22-Mar-2021 .
3. Create a new table named as tbl_shares1.
4. Insert Some Row in Table tbl_shares1.
5. Display all the records you have entered in table.
6. Display records where price>100 and TransTime='01:09'.
7. Display the records where price=(select max(price) from tbl_shares1 where TransTime='02:04');.

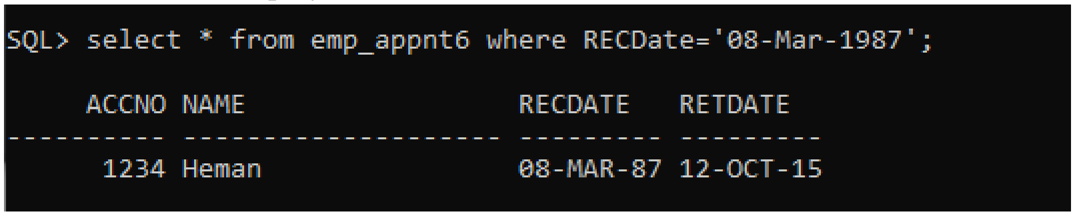


```
SQL> create table emp_appnt6
  2  ( accno number(10),
  3  name varchar2(20),
  4  recdate date,
  5  retdate date);

Table created.

SQL> insert into emp_appnt6 values(1234,'Heman','08-Mar-1987','12-Oct-2015');

1 row created.

SQL> insert into emp_appnt6 values(1235,'Shakthi','08-Oct-1978','19-Nov-2020');

1 row created.

SQL> insert into emp_appnt6 values(1236,'Suprabhat','25-Jan-1988','20-feb-2021');

1 row created.

SQL> insert into emp_appnt6 values(1237,'Preetam','05-Dec-1978','02-Mar-2017');

1 row created.

SQL> insert into emp_appnt6 values(1238,'Chirag','01-Nov-1999','22-Mar-2021');

1 row created.
```



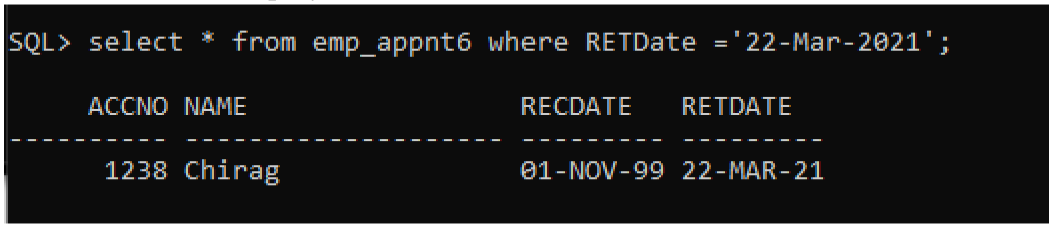
```
SQL> select * from emp_appnt6;

     ACCNO NAME                 RECDATE   RETDATE
---------- -------------------- --------- ---------
      1234 Heman                08-MAR-87 12-OCT-15
      1235 Shakthi              08-OCT-78 19-NOV-20
      1236 Suprabhat            25-JAN-88 20-FEB-21
      1237 Preetam              05-DEC-78 02-MAR-17
      1238 Chirag               01-NOV-99 22-MAR-21
```

1. Show the Employee Whose Record Date is 08-Mar-1987.



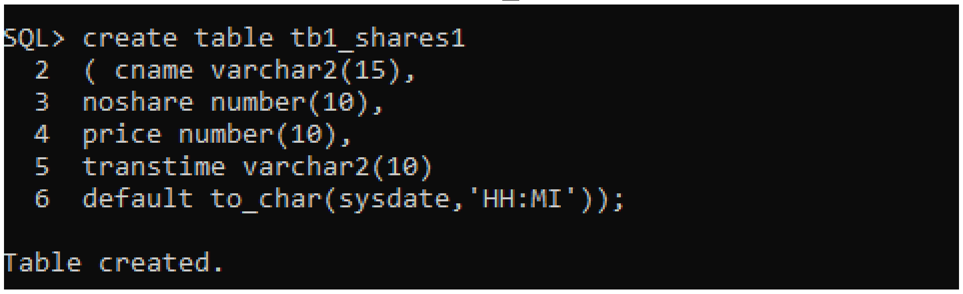
```
SQL> select * from emp_appnt6 where RECDate='08-Mar-1987';

     ACCNO NAME                       RECDATE   RETDATE
---------- -------------------- --------- ---------
      1234 Heman                      08-MAR-87 12-OCT-15
```

2. Show the Employee Whose Retired Date is 22-Mar-2021.



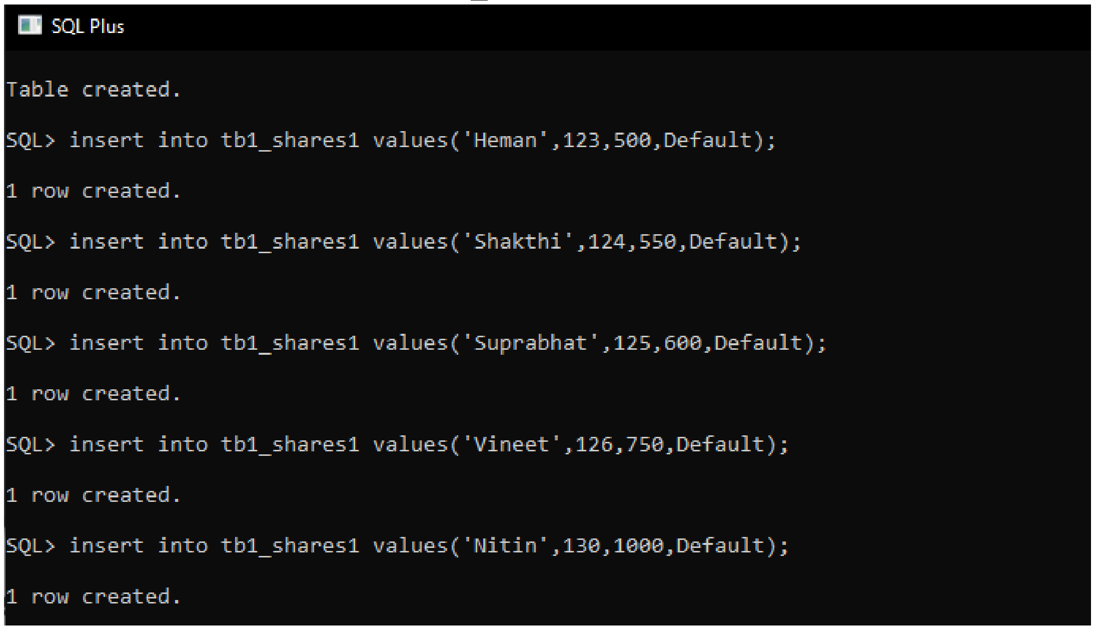
```
SQL> select * from emp_appnt6 where RETDate ='22-Mar-2021';

     ACCNO NAME                       RECDATE   RETDATE
---------- -------------------- --------- ---------
      1238 Chirag                     01-NOV-99 22-MAR-21
```

3. Create a new table named as tbl_shares1.



```
SQL> create table tb1_shares1
  2  ( cname varchar2(15),
  3  noshare number(10),
  4  price number(10),
  5  transtime varchar2(10)
  6  default to_char(sysdate,'HH:MI'));

Table created.
```

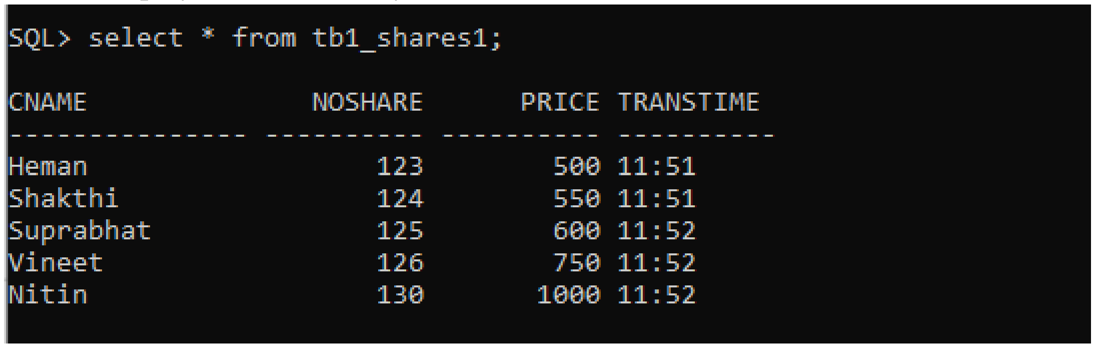
4. Insert Some Row in Table tbl_shares1.



SQL Plus

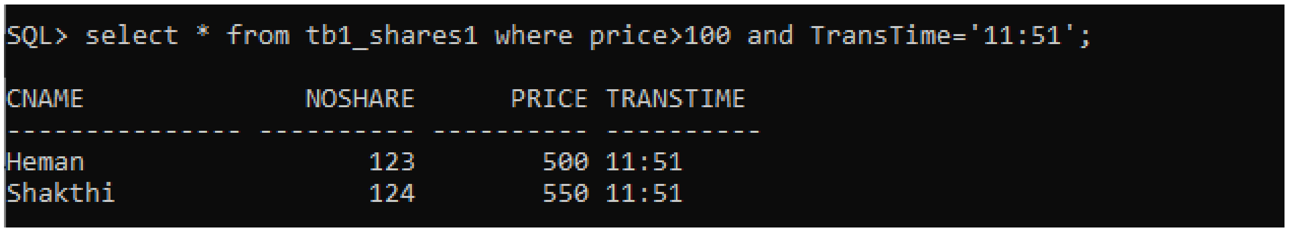
```
Table created.

SQL> insert into tb1_shares1 values('Heman',123,500,Default);

1 row created.

SQL> insert into tb1_shares1 values('Shakthi',124,550,Default);

1 row created.

SQL> insert into tb1_shares1 values('Suprabhat',125,600,Default);

1 row created.

SQL> insert into tb1_shares1 values('Vineet',126,750,Default);

1 row created.

SQL> insert into tb1_shares1 values('Nitin',130,1000,Default);

1 row created.
```

5. Display all the records you have entered in table.



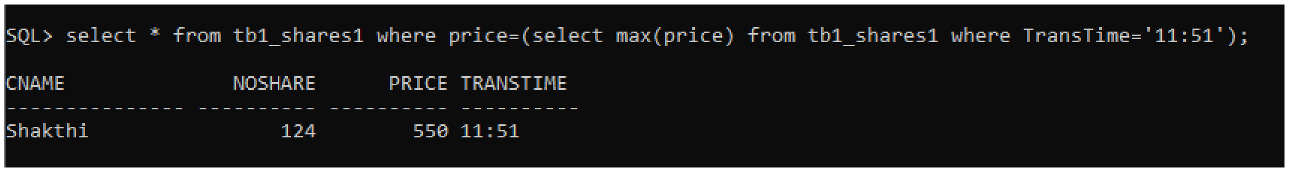
```
SQL> select * from tb1_shares1;

CNAME              NOSHARE      PRICE TRANSTIME
--------------- ---------- ---------- ----------
Heman                  123        500 11:51
Shakthi                124        550 11:51
Suprabhat              125        600 11:52
Vineet                 126        750 11:52
Nitin                  130       1000 11:52
```

6. Display records where price>100 and TransTime='11:51''.



```
SQL> select * from tb1_shares1 where price>100 and TransTime='11:51';

CNAME              NOSHARE      PRICE TRANSTIME
--------------- ---------- ---------- ----------
Heman                  123        500 11:51
Shakthi                124        550 11:51
```

7. Display the records where price=(select max(price) from tbl_shares1 where TransTime='02:04');

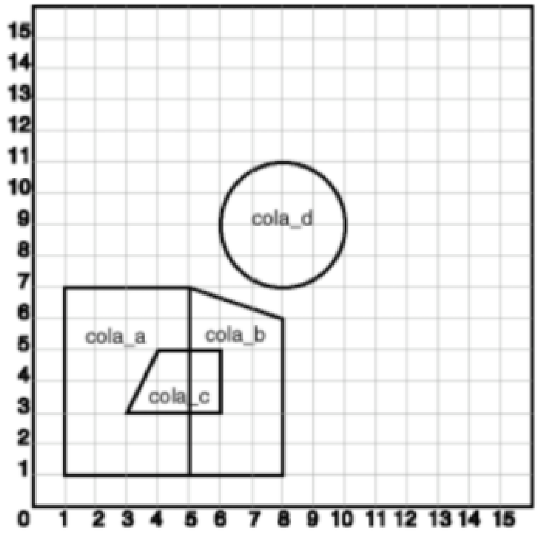


```
SQL> select * from tb1_shares1 where price=(select max(price) from tb1_shares1 where TransTime='11:51');

CNAME              NOSHARE      PRICE TRANSTIME
--------------- ---------- ---------- ----------
Shakthi                124        550 11:51
```

Conclusion : Successfully Performed and Implemented the temporal database and issued queries on Oracle Database.

PRACTICAL NO: 6

Aim: Create a table that stores spatial data and issues queries on it.



Software Requirement: Oracle 11g.

Query:

Create a spatial database table that stores the number, name and location, which consists of four different areas say abc, pqr, mno and xyz.

Fire the following queries:

a) Find the topological intersection of two geometries.

b) Find whether two geometric figures are equivalent to each other.

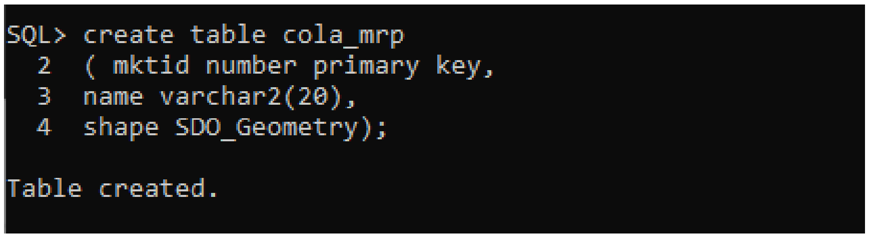
c) Find the areas of all different locations.

d) Find the area of only one location.

e) Find the distance between two geometries.

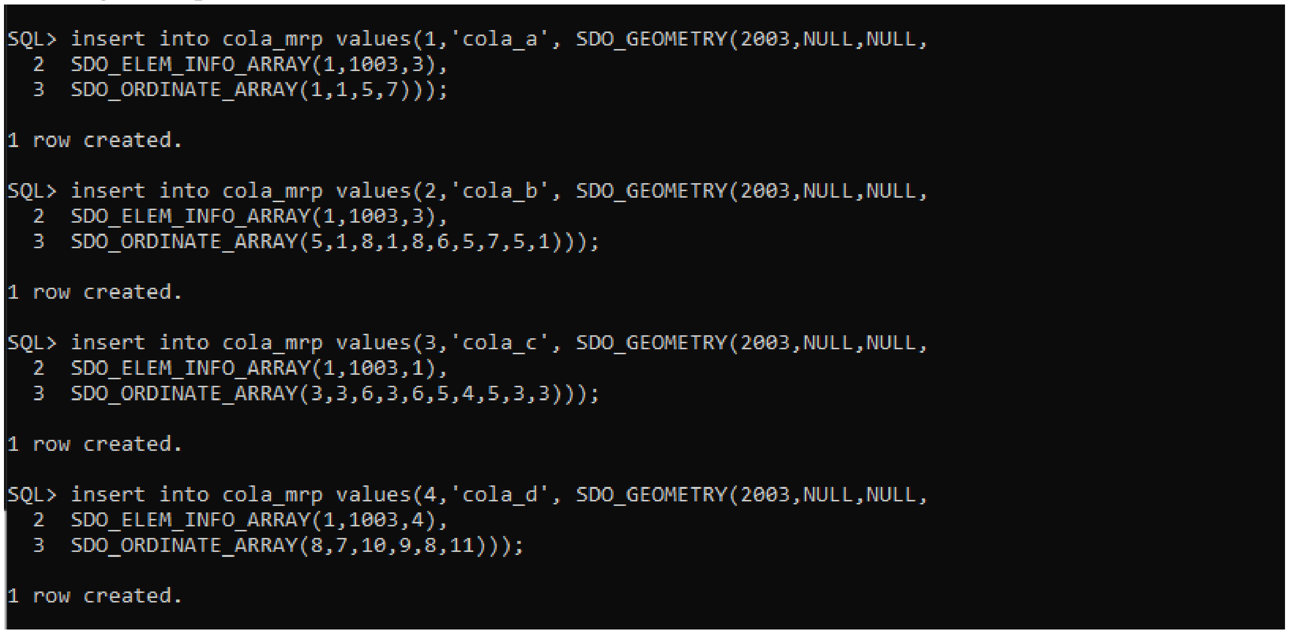
Practical Implementation:

1. Create a table for cola (soft drink) markets in a given geography (such as city or state). Each row will be an area of interest for a specific cola (for example, where the cola is most preferred by residents, where the manufacturer believes the cola has growth potential, and so on). (For restrictions on spatial table and column names, see



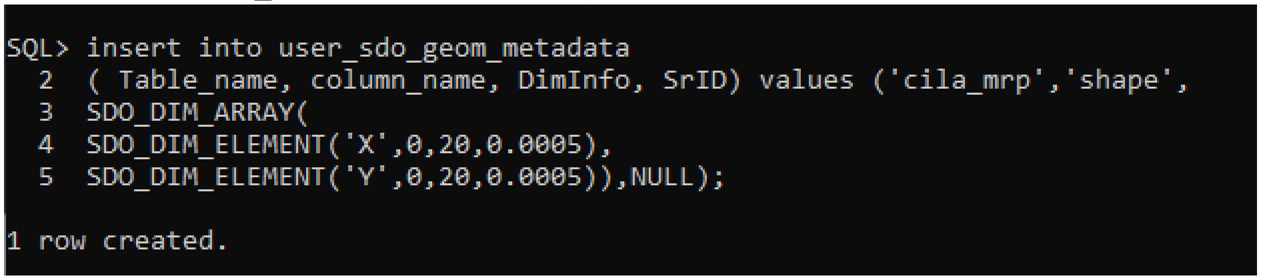
```
SQL> create table cola_mrp
  2  ( mktid number primary key,
  3  name varchar2(20),
  4  shape SDO_Geometry);

Table created.
```

2. The next INSERT statement creates an area of interest for Cola A. This area happens to be a rectangle. The area could represent any user-defined criterion: for example, where Cola A is the preferred drink, where Cola A is under competitive pressure, where Cola A has strong growth potential, and so on.



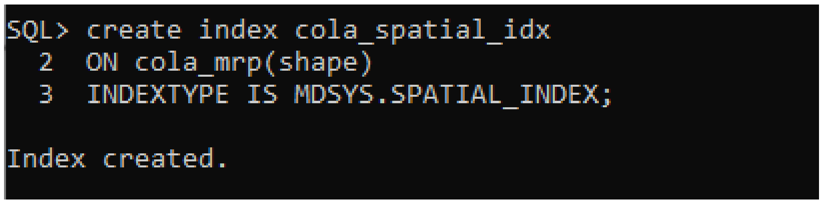
```
SQL> insert into cola_mrp values(1,'cola_a', SDO_GEOMETRY(2003,NULL,NULL,
  2  SDO_ELEM_INFO_ARRAY(1,1003,3),
  3  SDO_ORDINATE_ARRAY(1,1,5,7)));

1 row created.

SQL> insert into cola_mrp values(2,'cola_b', SDO_GEOMETRY(2003,NULL,NULL,
  2  SDO_ELEM_INFO_ARRAY(1,1003,3),
  3  SDO_ORDINATE_ARRAY(5,1,8,1,8,6,5,7,5,1)));

1 row created.

SQL> insert into cola_mrp values(3,'cola_c', SDO_GEOMETRY(2003,NULL,NULL,
  2  SDO_ELEM_INFO_ARRAY(1,1003,1),
  3  SDO_ORDINATE_ARRAY(3,3,6,3,6,5,4,5,3,3)));

1 row created.

SQL> insert into cola_mrp values(4,'cola_d', SDO_GEOMETRY(2003,NULL,NULL,
  2  SDO_ELEM_INFO_ARRAY(1,1003,4),
  3  SDO_ORDINATE_ARRAY(8,7,10,9,8,11)));

1 row created.
```

3. UPDATE METADATA VIEW
   Update the USER_SDO_GEOM_METADATA view. This is required before the spatial index can be created. Do this only once for each layer (that is, table-column combination; here: COLA_MARKETS and SHAPE).



```
SQL> insert into user_sdo_geom_metadata
  2  ( Table_name, column_name, DimInfo, SrID) values ('cila_mrp','shape',
  3  SDO_DIM_ARRAY(
  4  SDO_DIM_ELEMENT('X',0,20,0.0005),
  5  SDO_DIM_ELEMENT('Y',0,20,0.0005)),NULL);

1 row created.
```

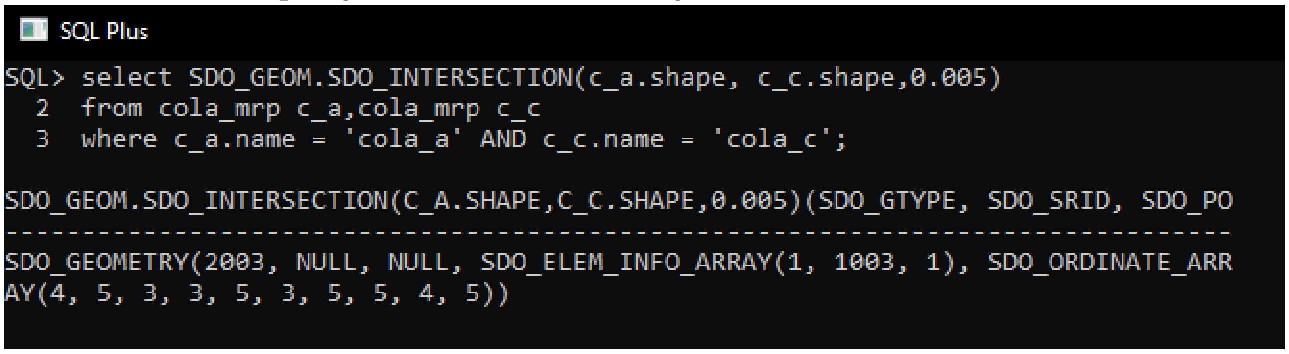
4. CREATE THE SPATIAL INDEX.



```
SQL> create index cola_spatial_idx
  2  ON cola_mrp(shape)
  3  INDEXTYPE IS MDSYS.SPATIAL_INDEX;

Index created.
```

5. PERFORM SOME SPATIAL QUERIES

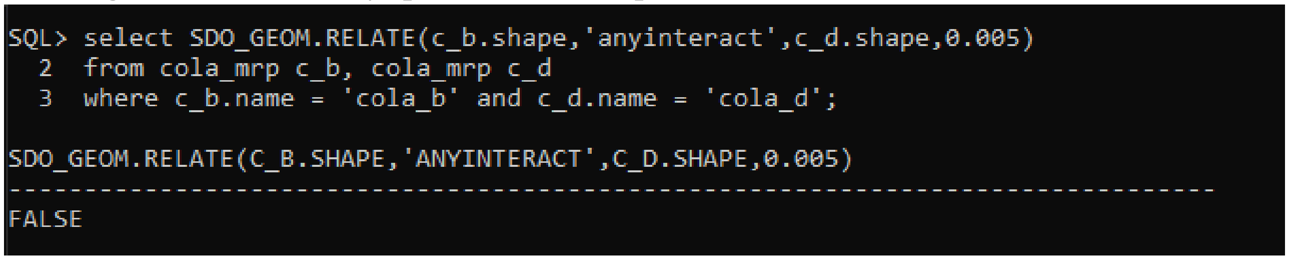
Return the topological intersection of two geometries.



SQL Plus

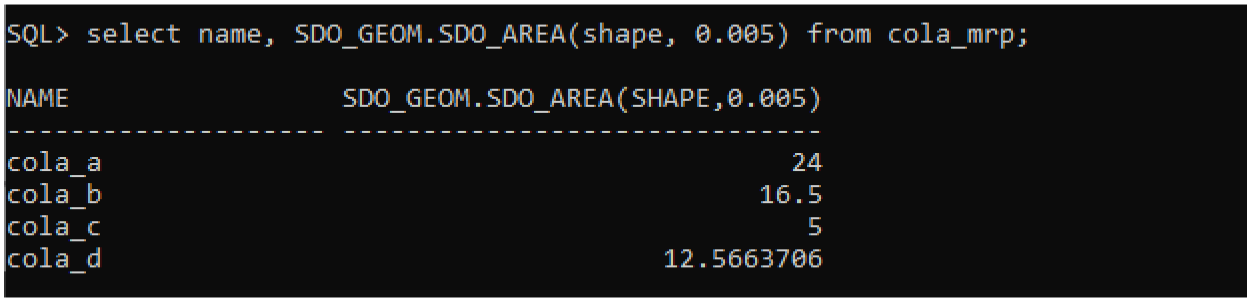
```
SQL> select SDO_GEOM.SDO_INTERSECTION(c_a.shape, c_c.shape,0.005)
  2  from cola_mrp c_a,cola_mrp c_c
  3  where c_a.name = 'cola_a' AND c_c.name = 'cola_c';

SDO_GEOM.SDO_INTERSECTION(C_A.SHAPE,C_C.SHAPE,0.005)(SDO_GTYPE, SDO_SRID, SDO_PO
--------------------------------------------------------------------------------
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1), SDO_ORDINATE_ARR
AY(4, 5, 3, 3, 5, 3, 5, 5, 4, 5))
```

Do two geometries have any spatial relationship?



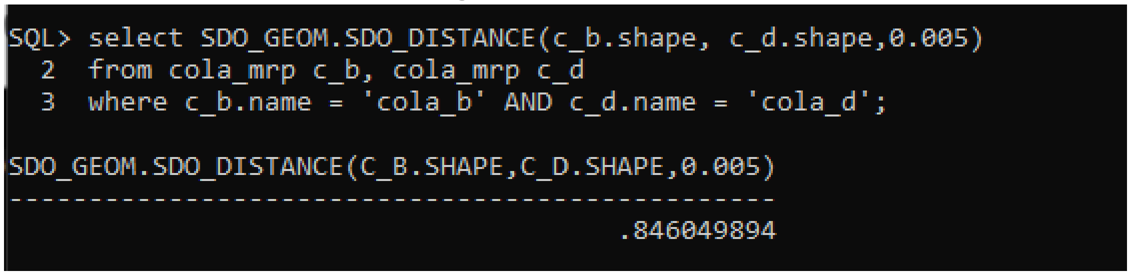
```
SQL> select SDO_GEOM.RELATE(c_b.shape,'anyinteract',c_d.shape,0.005)
  2  from cola_mrp c_b, cola_mrp c_d
  3  where c_b.name = 'cola_b' and c_d.name = 'cola_d';

SDO_GEOM.RELATE(C_B.SHAPE,'ANYINTERACT',C_D.SHAPE,0.005)
--------------------------------------------------------------------------------
FALSE
```

Return the areas of all cola markets.



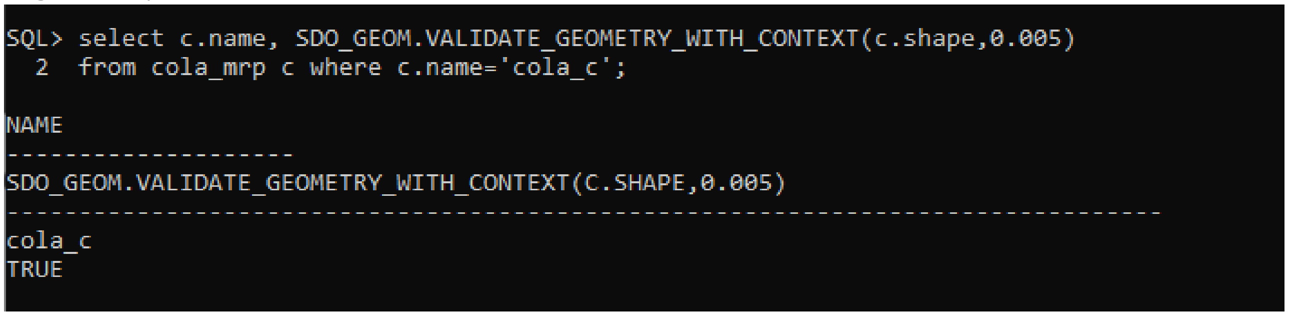
```
SQL> select name, SDO_GEOM.SDO_AREA(shape, 0.005) from cola_mrp;

NAME                 SDO_GEOM.SDO_AREA(SHAPE,0.005)
-------------------- ------------------------------
cola_a                                           24
cola_b                                         16.5
cola_c                                            5
cola_d                                   12.5663706
```

Return the distance between two geometries.



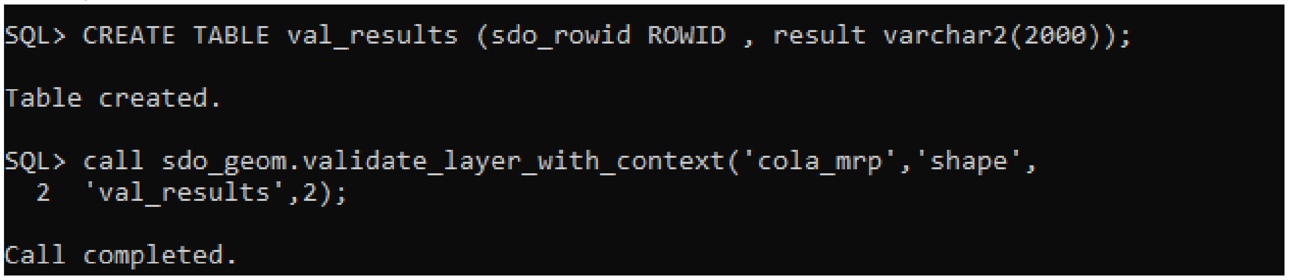
```
SQL> select SDO_GEOM.SDO_DISTANCE(c_b.shape, c_d.shape,0.005)
  2  from cola_mrp c_b, cola_mrp c_d
  3  where c_b.name = 'cola_b' AND c_d.name = 'cola_d';

SDO_GEOM.SDO_DISTANCE(C_B.SHAPE,C_D.SHAPE,0.005)
------------------------------------------------
                                       .846049894
```

Is geometry valid?

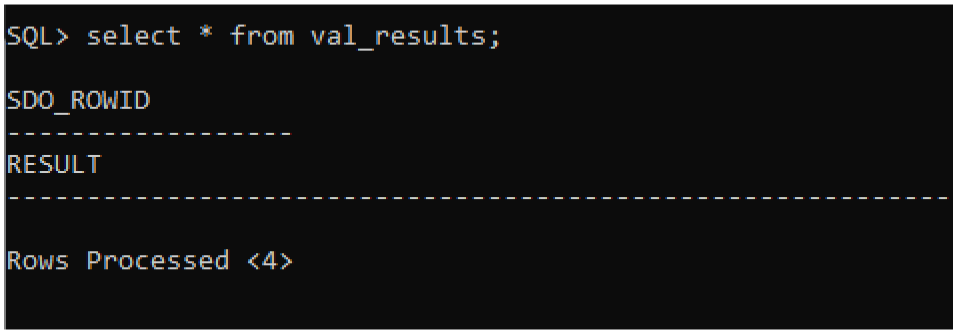


```
SQL> select c.name, SDO_GEOM.VALIDATE_GEOMETRY_WITH_CONTEXT(c.shape,0.005)
  2  from cola_mrp c where c.name='cola_c';

NAME
--------------------
SDO_GEOM.VALIDATE_GEOMETRY_WITH_CONTEXT(C.SHAPE,0.005)
--------------------------------------------------------------------------------
cola_c
TRUE
```

Is a layer valid? (First, create the results table).



```
SQL> CREATE TABLE val_results (sdo_rowid ROWID , result varchar2(2000));

Table created.

SQL> call sdo_geom.validate_layer_with_context('cola_mrp','shape',
  2  'val_results',2);

Call completed.
```



```
SQL> select * from val_results;

SDO_ROWID
------------------
RESULT
------------------------------------------------------------

Rows Processed <4>
```

Conclusion : Successfully Performed the Spatial Data Queries on Oracle Database.

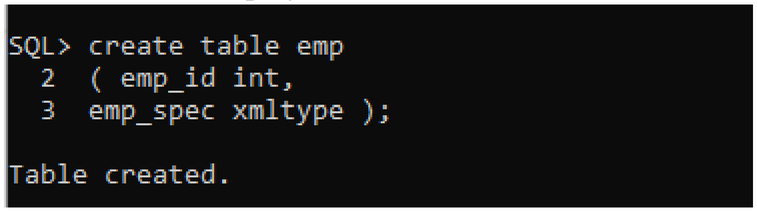
PRACTICAL NO: 7

Aim: Create a table employee having dept_id as number data type and employee_spec as XML data type (XM_Type). The employee_spec is a schema with attributes emp_id, name, email, acc_no, managerEmail, dataOf Joining. Insert 10 tuples into the employee table. Fire the following queries on the XML database.

Query:
1. Retrieve the names of employees.
2. Retrieve the acc_no of employees.
3. Retrieve the names, acc_no, and email of employees.
4. Update the 3rd record from the table and display the name of an employee.
5. Delete 4th record from the table.

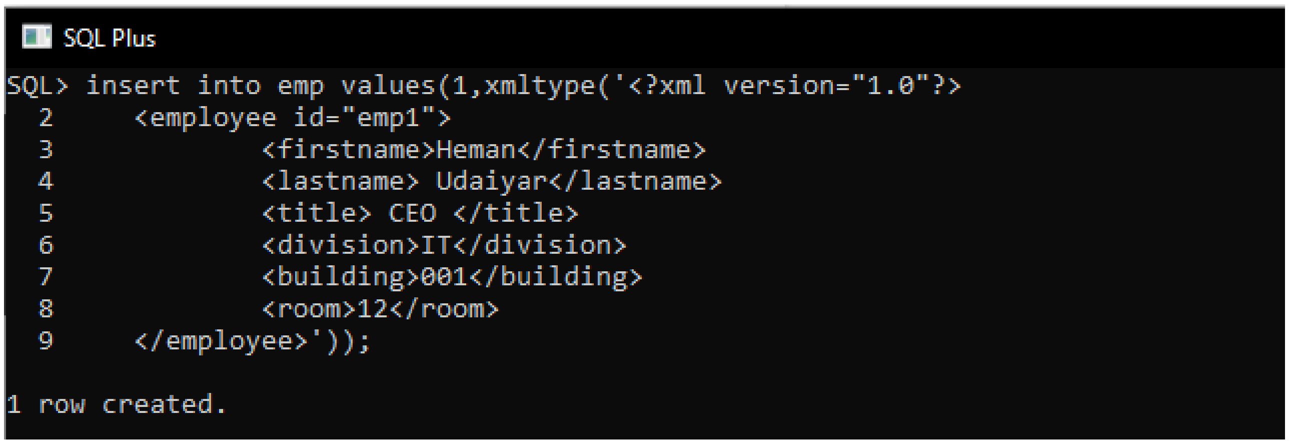
Software Requirements:
Oracle 11g Express Edition, Any browser.

Practical Implementation:
1. Create Table Employees.

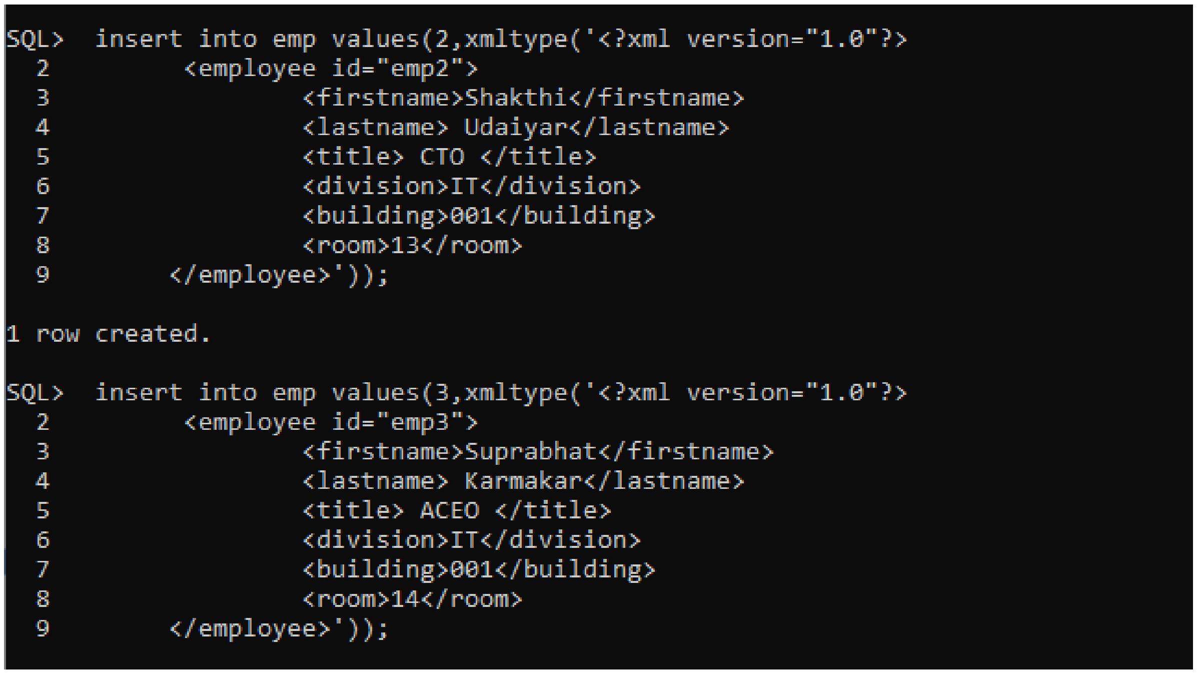


```
SQL> create table emp
  2  ( emp_id int,
  3  emp_spec xmltype );

Table created.
```

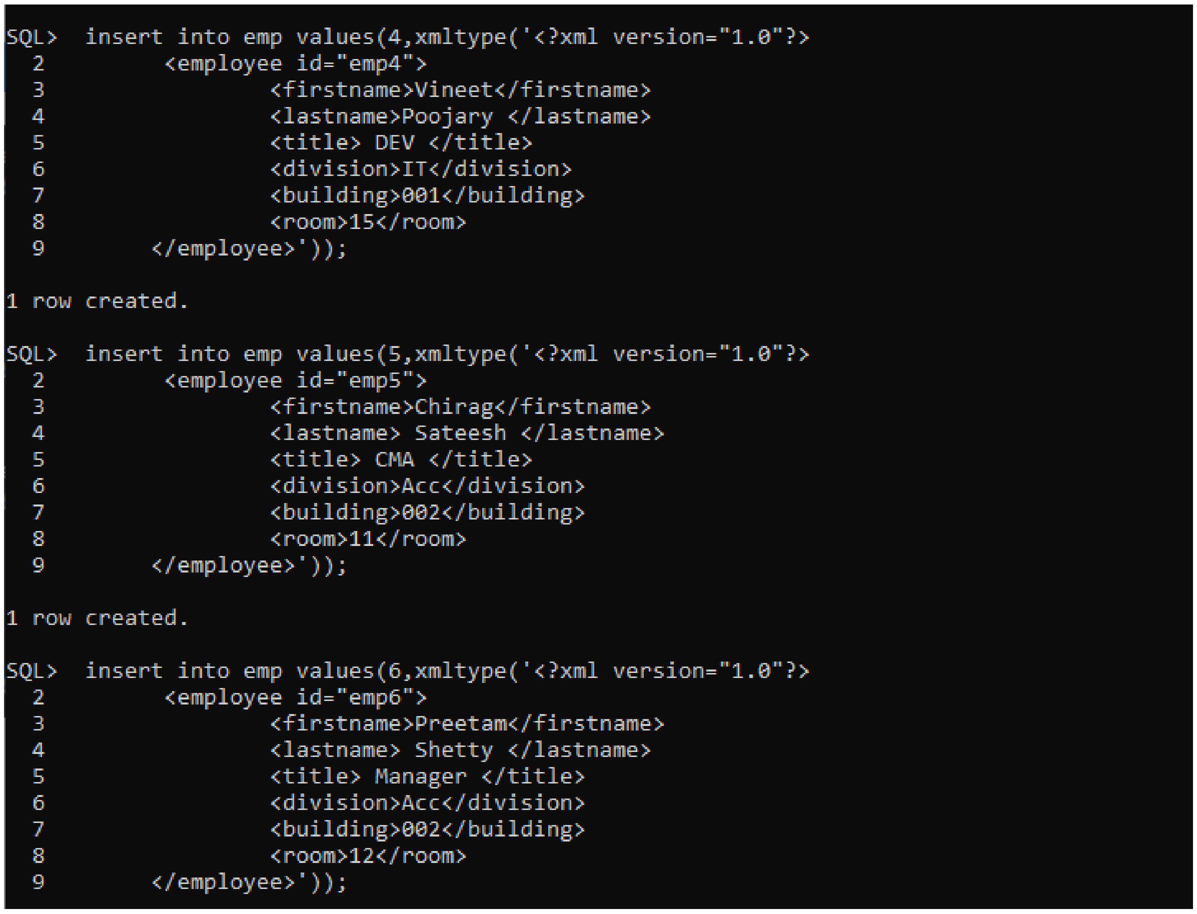
2. Insert Some Records in Created Table.



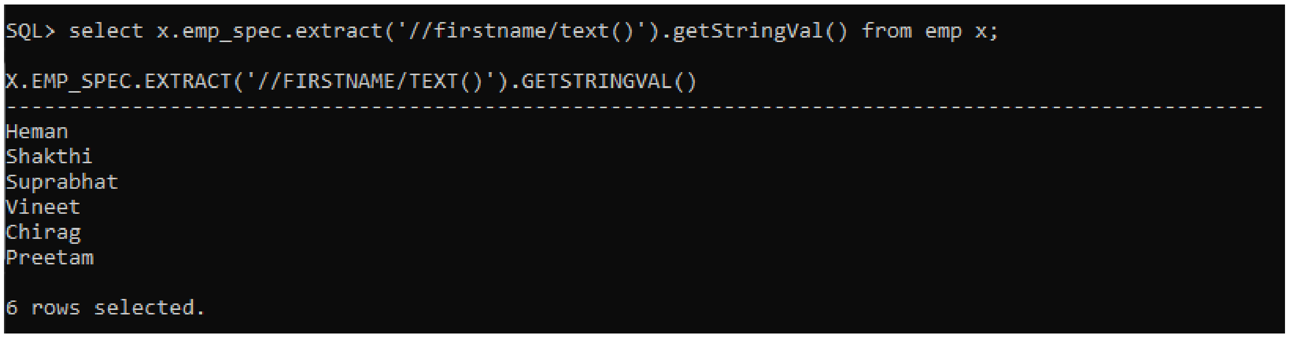
```
SQL Plus
SQL> insert into emp values(1,xmltype('<?xml version="1.0"?>
  2     <employee id="emp1">
  3             <firstname>Heman</firstname>
  4             <lastname> Udaiyar</lastname>
  5             <title> CEO </title>
  6             <division>IT</division>
  7             <building>001</building>
  8             <room>12</room>
  9     </employee>'));

1 row created.
```



```
SQL>  insert into emp values(2,xmltype('<?xml version="1.0"?>
  2         <employee id="emp2">
  3                 <firstname>Shakthi</firstname>
  4                 <lastname> Udaiyar</lastname>
  5                 <title> CTO </title>
  6                 <division>IT</division>
  7                 <building>001</building>
  8                 <room>13</room>
  9        </employee>'));

1 row created.

SQL>  insert into emp values(3,xmltype('<?xml version="1.0"?>
  2         <employee id="emp3">
  3                 <firstname>Suprabhat</firstname>
  4                 <lastname> Karmakar</lastname>
  5                 <title> ACEO </title>
  6                 <division>IT</division>
  7                 <building>001</building>
  8                 <room>14</room>
  9        </employee>'));
```

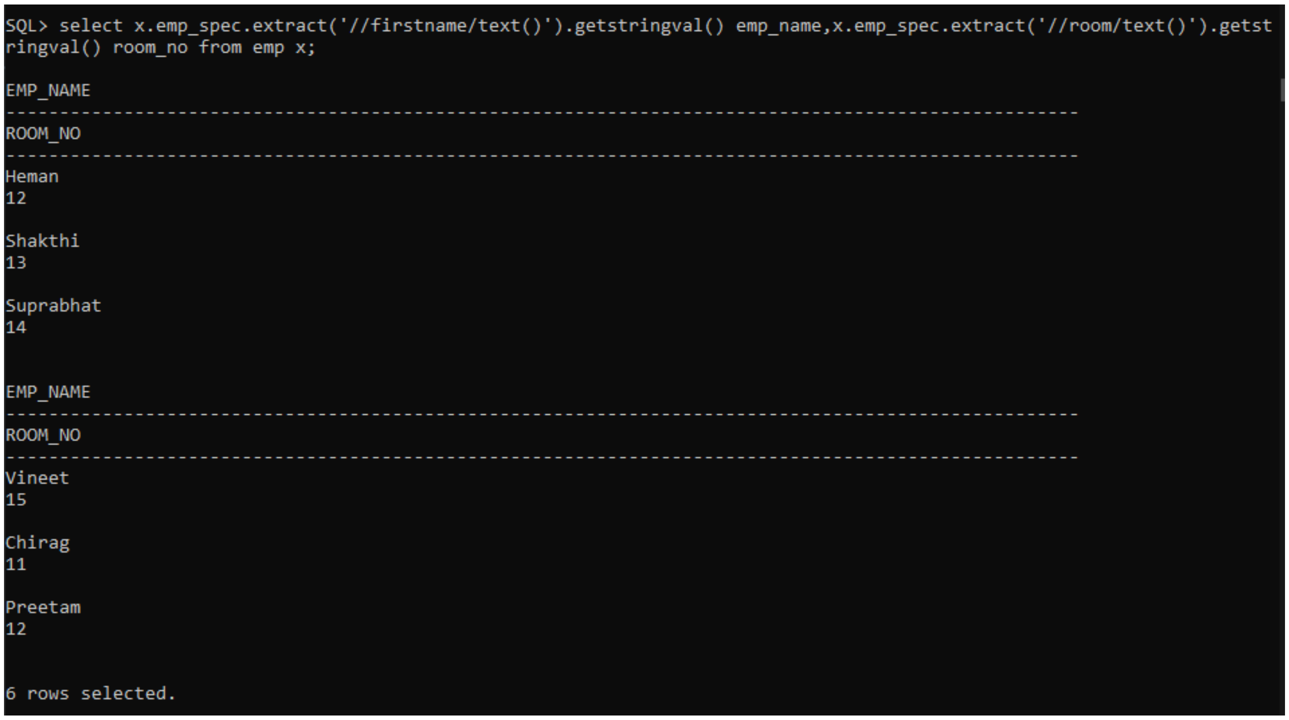


```
SQL>  insert into emp values(4,xmltype('<?xml version="1.0"?>
  2        <employee id="emp4">
  3                <firstname>Vineet</firstname>
  4                <lastname>Poojary </lastname>
  5                <title> DEV </title>
  6                <division>IT</division>
  7                <building>001</building>
  8                <room>15</room>
  9       </employee>'));

1 row created.

SQL>  insert into emp values(5,xmltype('<?xml version="1.0"?>
  2        <employee id="emp5">
  3                <firstname>Chirag</firstname>
  4                <lastname> Sateesh </lastname>
  5                <title> CMA </title>
  6                <division>Acc</division>
  7                <building>002</building>
  8                <room>11</room>
  9       </employee>'));

1 row created.

SQL>  insert into emp values(6,xmltype('<?xml version="1.0"?>
  2        <employee id="emp6">
  3                <firstname>Preetam</firstname>
  4                <lastname> Shetty </lastname>
  5                <title> Manager </title>
  6                <division>Acc</division>
  7                <building>002</building>
  8                <room>12</room>
  9       </employee>'));
```



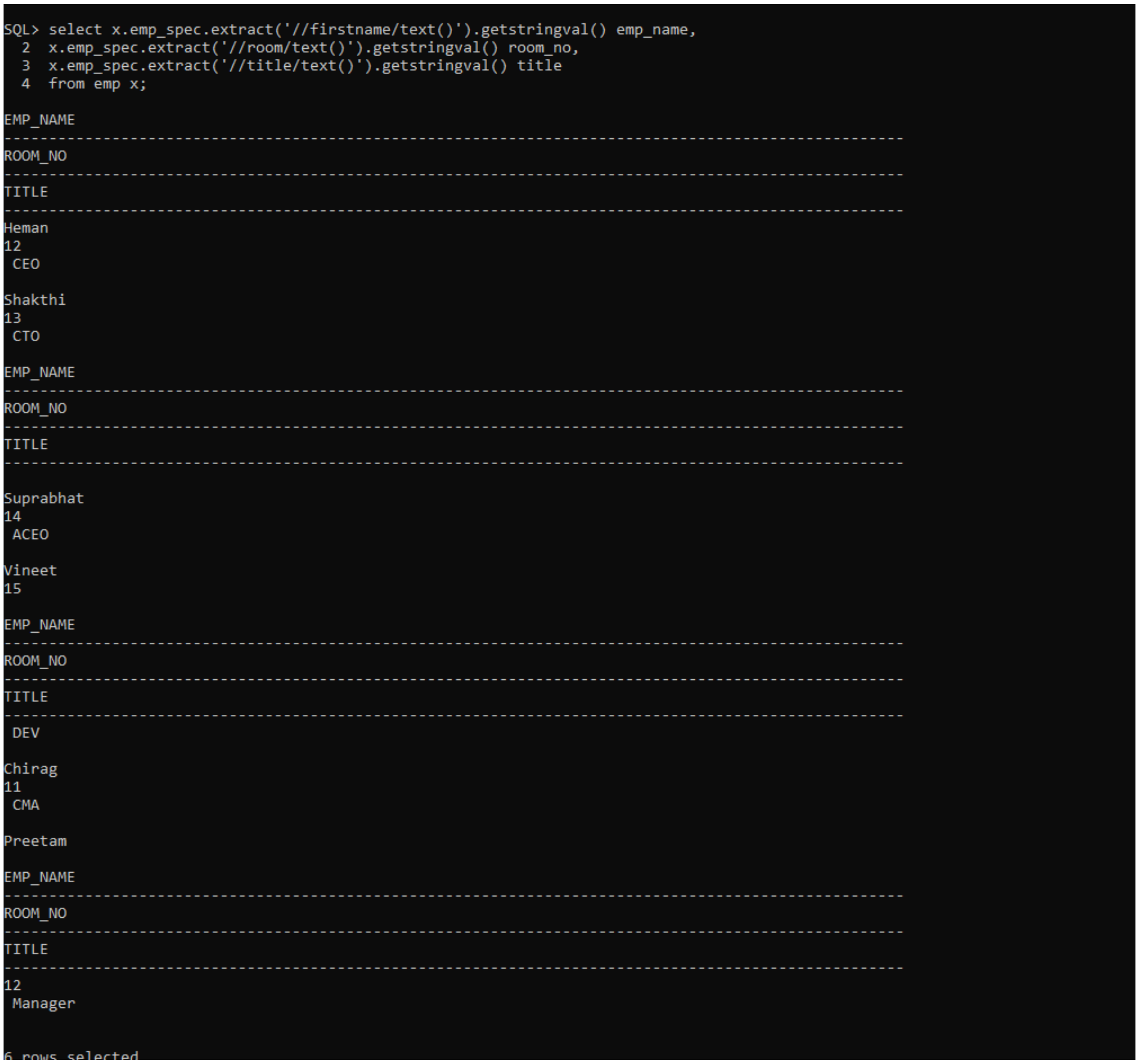
```
SQL> select x.emp_spec.extract('//firstname/text()').getStringVal() from emp x;

X.EMP_SPEC.EXTRACT('//FIRSTNAME/TEXT()').GETSTRINGVAL()
----------------------------------------------------------------------------------------------------
Heman
Shakthi
Suprabhat
Vineet
Chirag
Preetam

6 rows selected.
```

Get the first name and room number.



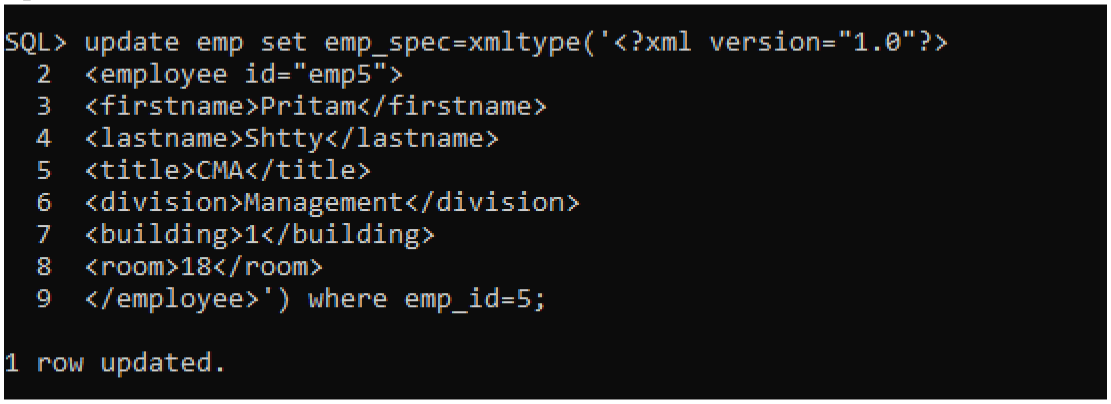
```
SQL> select x.emp_spec.extract('//firstname/text()').getstringval() emp_name,x.emp_spec.extract('//room/text()').getst
ringval() room_no from emp x;

EMP_NAME
----------------------------------------------------------------------------------------------------
ROOM_NO
----------------------------------------------------------------------------------------------------
Heman
12

Shakthi
13

Suprabhat
14


EMP_NAME
----------------------------------------------------------------------------------------------------
ROOM_NO
----------------------------------------------------------------------------------------------------
Vineet
15

Chirag
11

Preetam
12


6 rows selected.
```

Get the first name and room number and title.



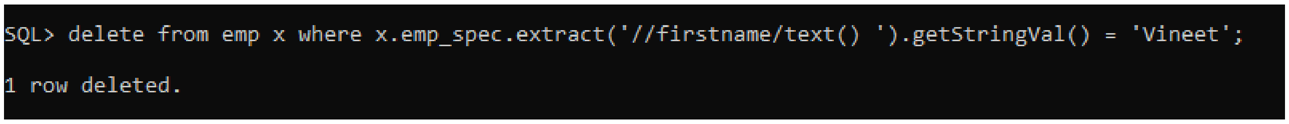
```
SQL> select x.emp_spec.extract('//firstname/text()').getstringval() emp_name,
  2  x.emp_spec.extract('//room/text()').getstringval() room_no,
  3  x.emp_spec.extract('//title/text()').getstringval() title
  4  from emp x;

EMP_NAME
----------------------------------------------------------------------------------------------------
ROOM_NO
----------------------------------------------------------------------------------------------------
TITLE
----------------------------------------------------------------------------------------------------
Heman
12
 CEO

Shakthi
13
 CTO

EMP_NAME
----------------------------------------------------------------------------------------------------
ROOM_NO
----------------------------------------------------------------------------------------------------
TITLE
----------------------------------------------------------------------------------------------------

Suprabhat
14
 ACEO

Vineet
15

EMP_NAME
----------------------------------------------------------------------------------------------------
ROOM_NO
----------------------------------------------------------------------------------------------------
TITLE
----------------------------------------------------------------------------------------------------
 DEV

Chirag
11
 CMA

Preetam

EMP_NAME
----------------------------------------------------------------------------------------------------
ROOM_NO
----------------------------------------------------------------------------------------------------
TITLE
----------------------------------------------------------------------------------------------------
12
 Manager


6 rows selected
```

Update 6th record from the table:



```
SQL> update emp set emp_spec=xmltype('<?xml version="1.0"?>
<employee id="emp5">
<firstname>Pritam</firstname>
<lastname>Shtty</lastname>
<title>CMA</title>
<division>Management</division>
<building>1</building>
<room>18</room>
</employee>') where emp_id=5;

1 row updated.
```

Delete a record from the table:



```
SQL> delete from emp x where x.emp_spec.extract('//firstname/text() ').getStringVal() = 'Vineet';

1 row deleted.
```

Conclusion :
Successfully Performed Operation like Create, Read, Update and Delete on XML Database.