

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr no. | Topic | Page Number | Date |
| 1. | Practical 1- 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. | 4-27 | 25/09/2023 |
| 2. | Practical 2- Place the replication of global conceptual schema on different nodes and execute queries that will demonstrate distributed databases environment. | 28-33 | 09/10/2023 |
| 3. | Practical 3- CRUD operation using MongoDB. | 34-37 | 18/10/2023 |
| 4. | Practical 4- 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. Fire suitable queries on them. | 38-46 | 23/10/2023 |
| 5. | Practical 5- Create a temporal database and issue queries on it. | 47-51 | 30/10/2023 |
| 6. | Practical 6- Create a table that stores the special data and issue queries on it. | 52-58 | 13/11/2023 |
| 7. | Practical 7- Create a table employee having dept\_id as number datatype and employee\_spec as XML data type (XML\_Type). The employee\_spec is a schema with attributes emp\_id, name, email, acc\_no, managerEmail, dataOf Joning. Insert 10 tuples into employee table. Fire the following queries on XML database. | 59-67 | 16/11/2023 |

**PRACTICAL 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.

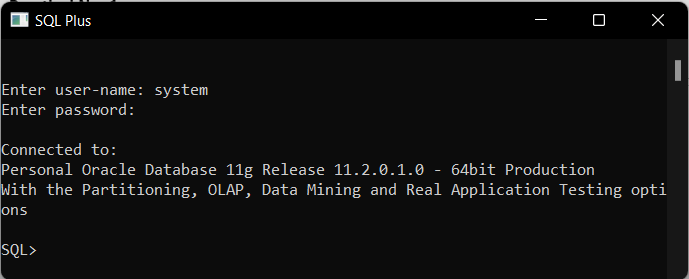
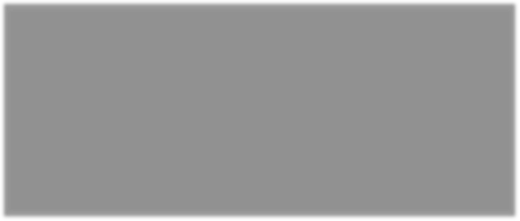
Theory:

A conceptual schema is an abstract representation of the entire database. It provides a high-level view of the data and its organization, focusing on the logical structure and relationships between data entities. It abstracts away the details of how data is physically stored or accessed and focuses on the way data is perceived by users and applications.

The global conceptual schema, in the context of distributed databases, extends the concept of the conceptual schema to encompass the entire distributed database environment. It defines the structure and organization of data across multiple interconnected databases, which may be geographically distributed or managed by different organizations.

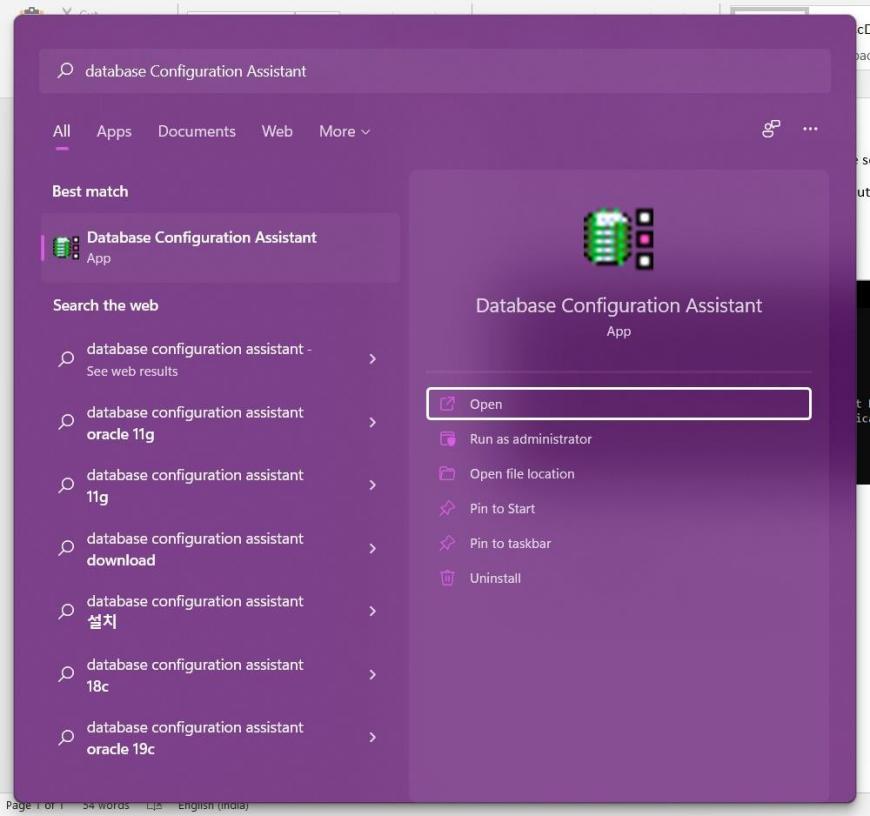
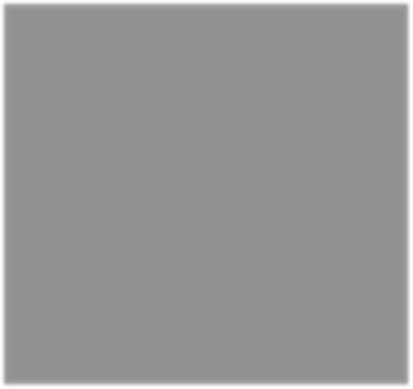
Software Requirement:

Oracle Database 11g.

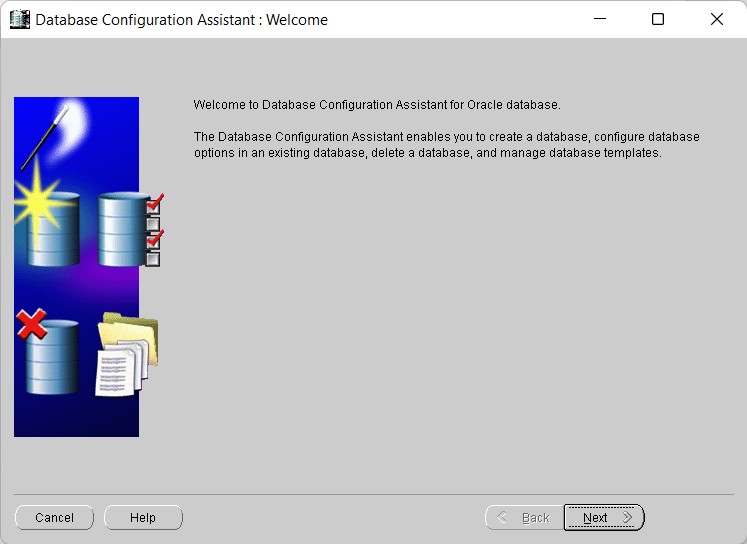
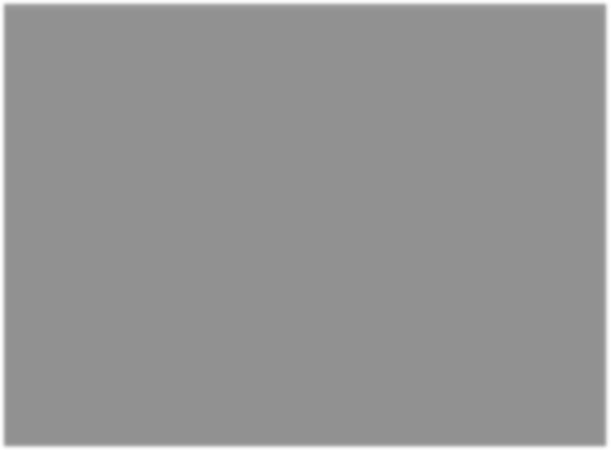


Steps to Create Database db1 and db2:

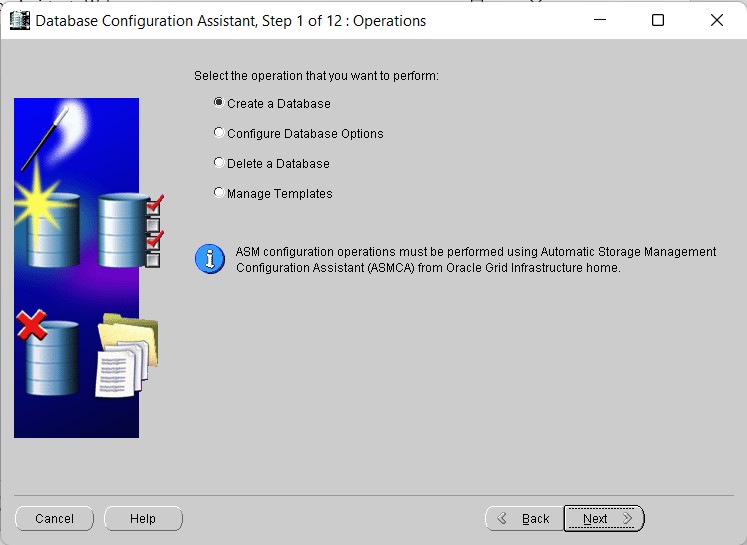
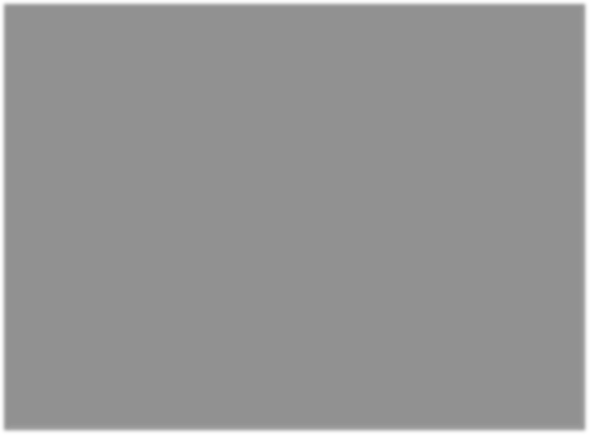
**Step 1:-** Open Start Menu on Window Explorer Go to Database Configuration Assistant.



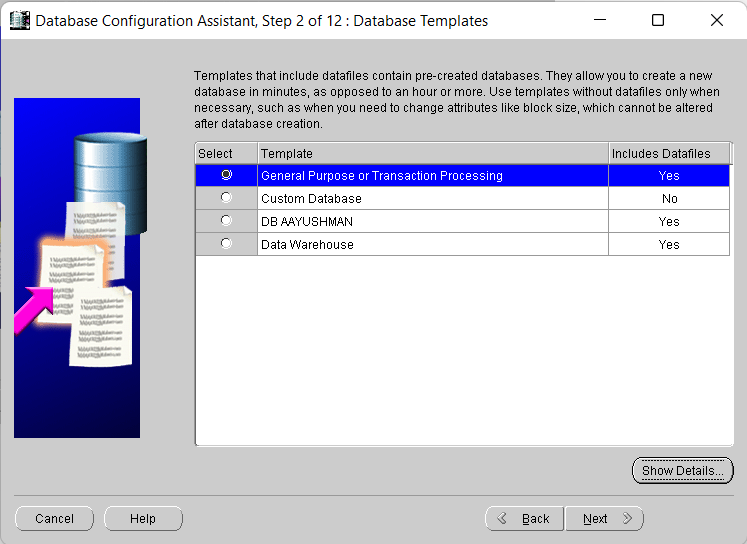
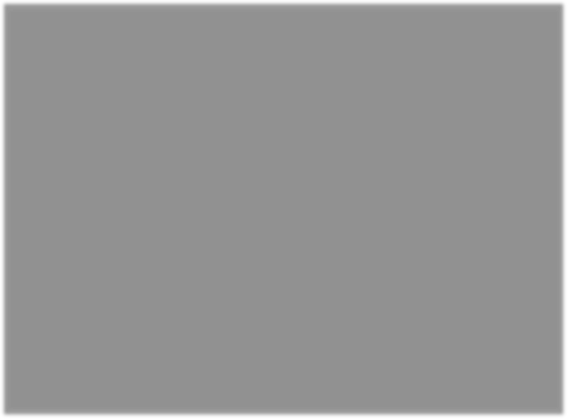
**Step 2:** Click on Next.



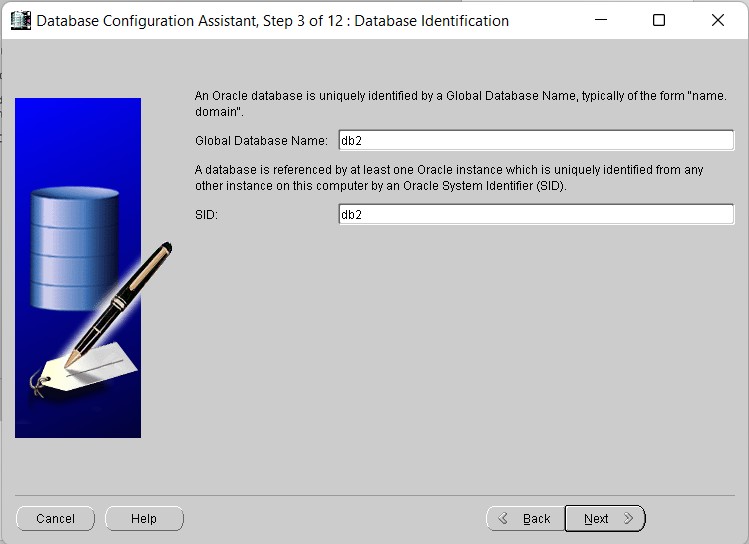
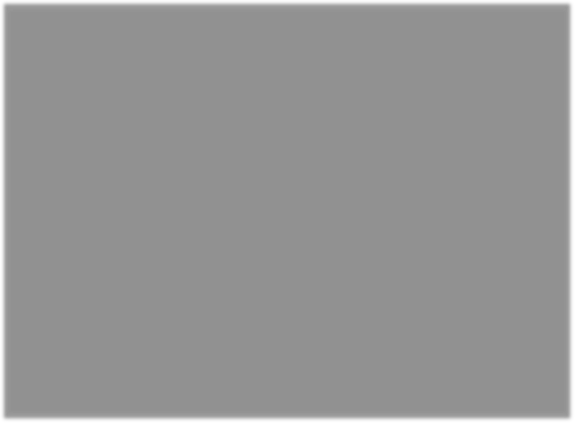
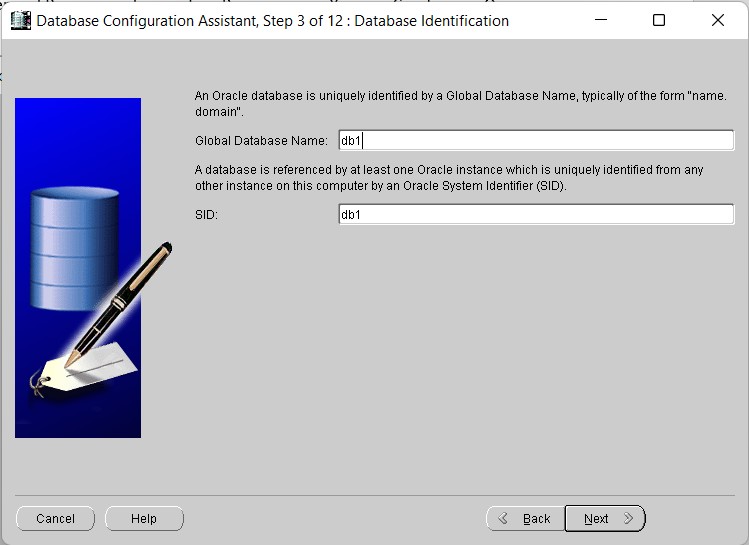
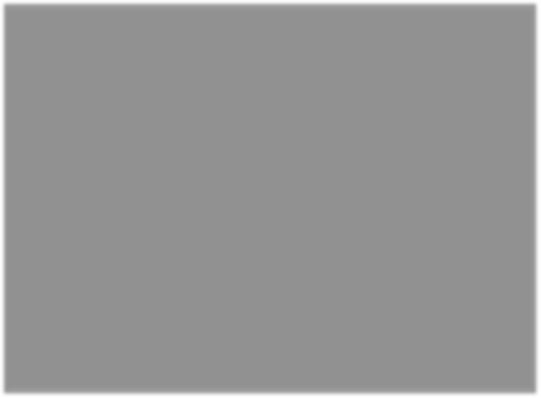
**Step 3:** Select Option Create a Database.



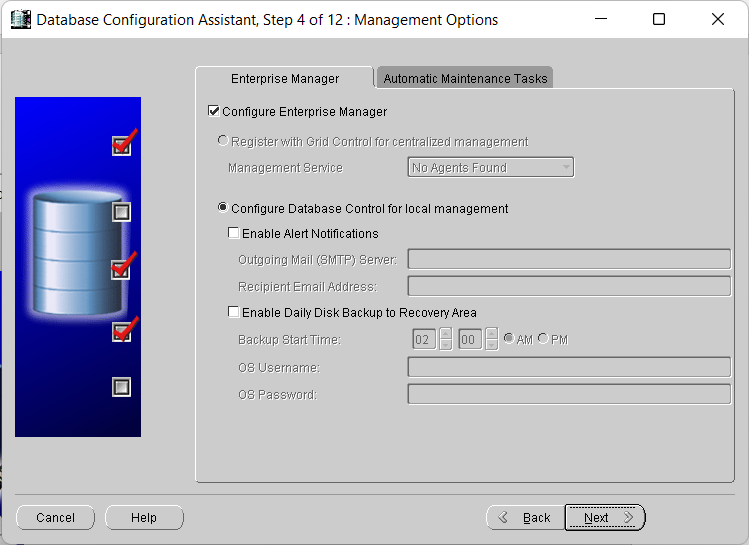
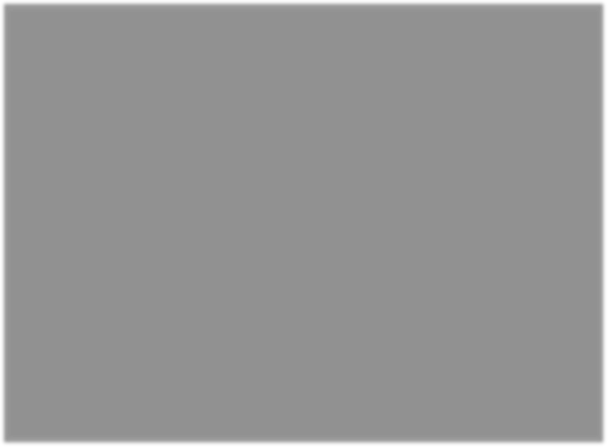
**Step 4:** Select Option General Purpose or Transaction Processing or You can Create your Own Custom Database.



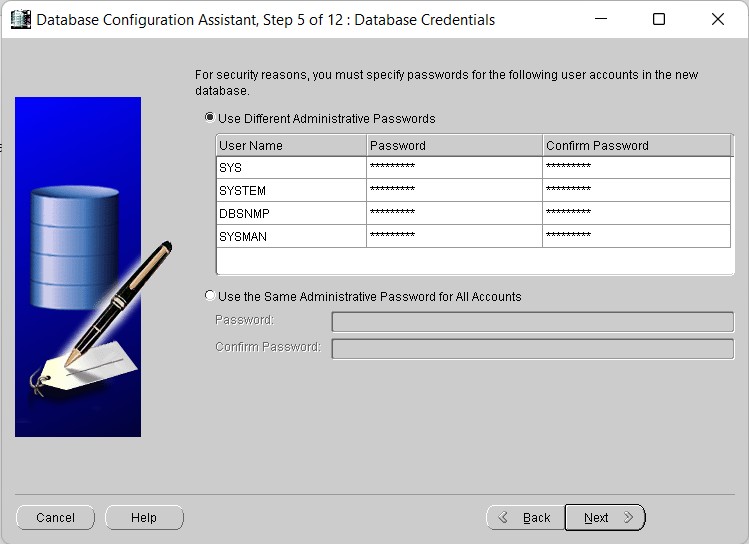
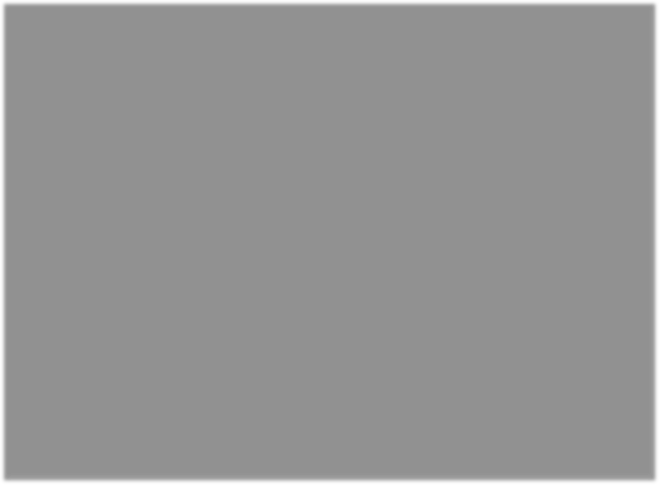
**Step 5:** Give Database Name as db1 (of your own choice).



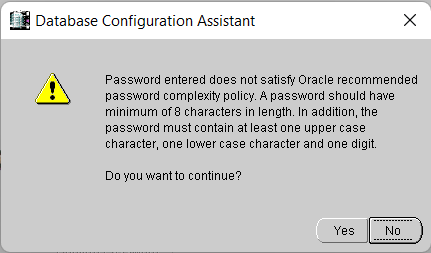
**Step 6:** No changes Needed, Click on Next.



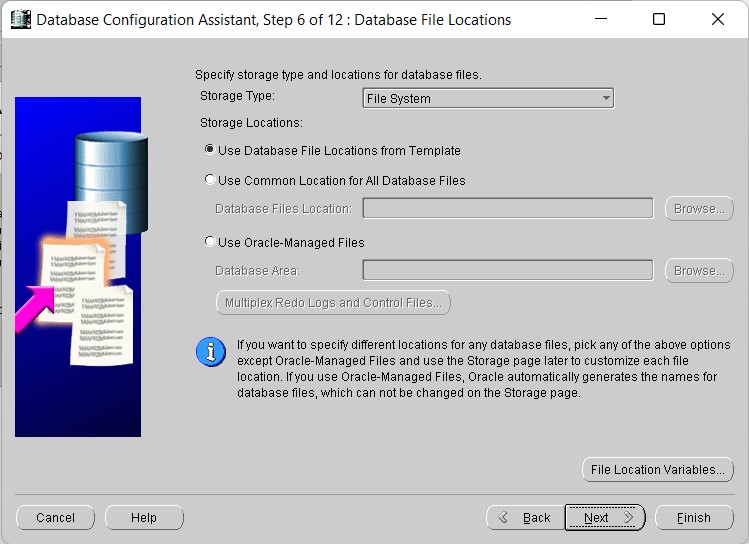
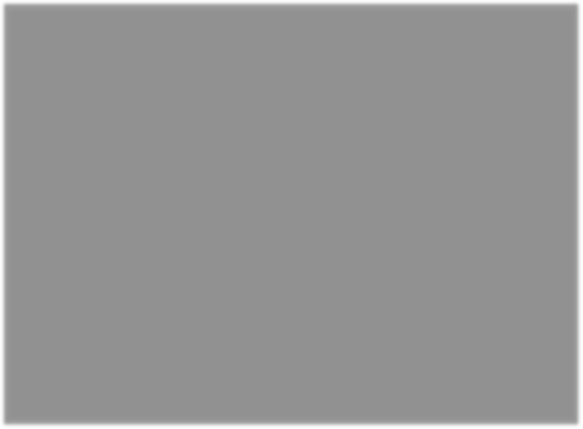
**Step 7:** Input Password of your choice for Each Fields or Else use your Administrator Credentials for all Profile.



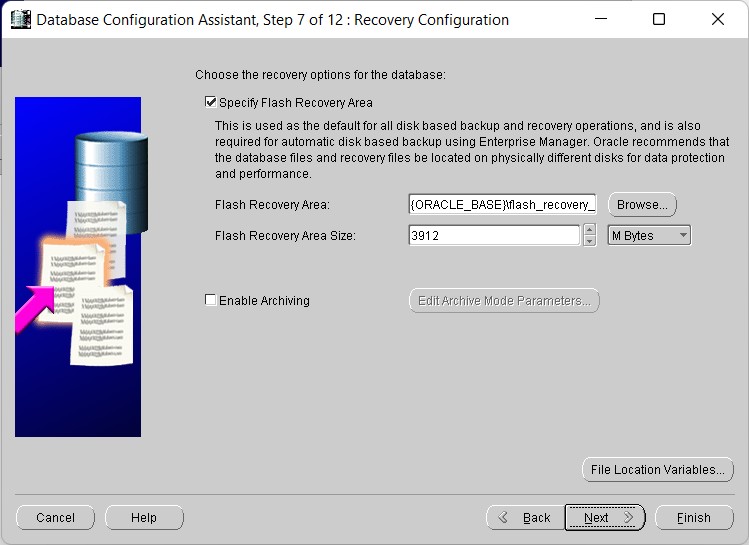
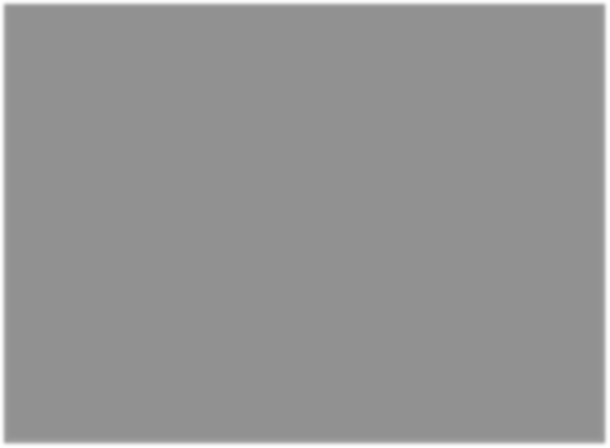
Checks for Password Confirmation, Just Click Yes.



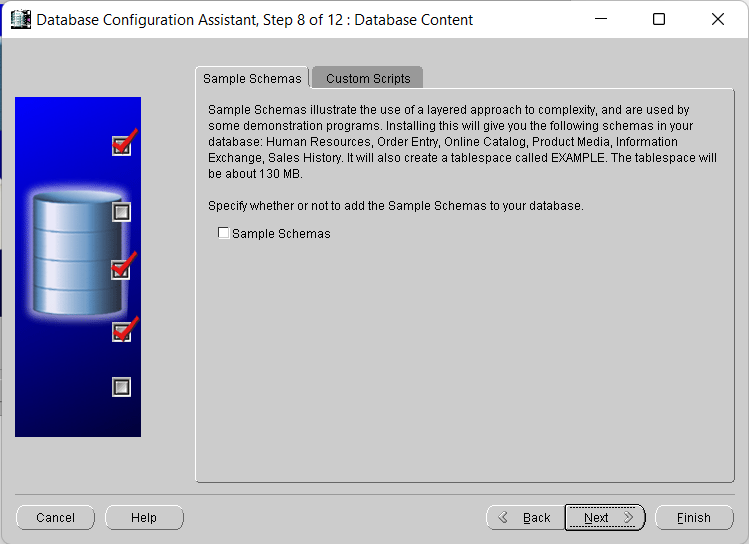
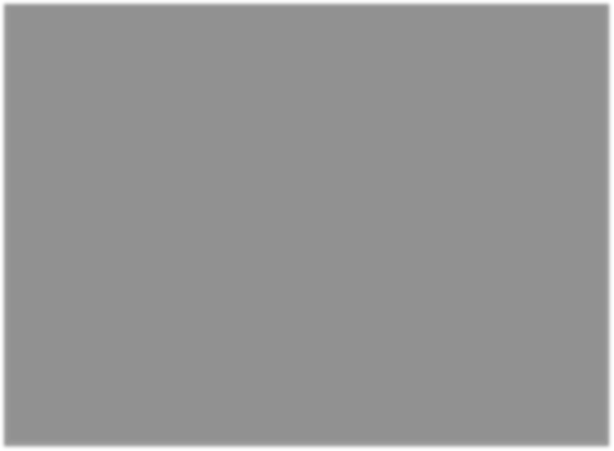
**Step 8:** No changes Needed, Click on Next.



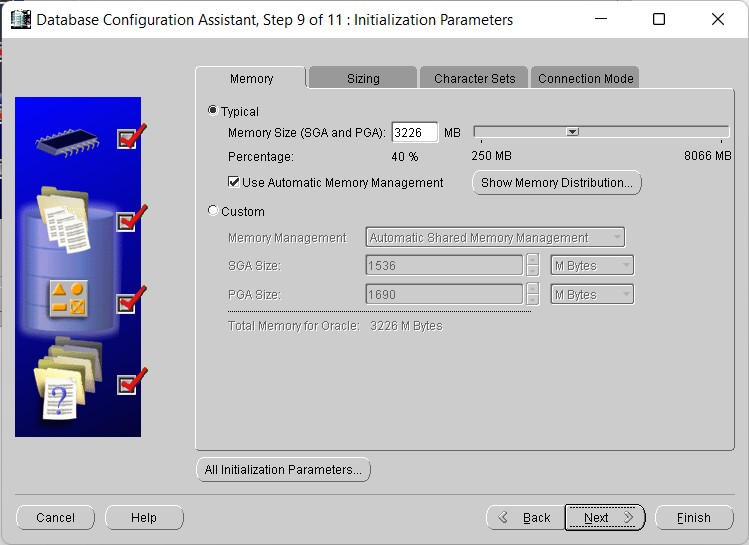
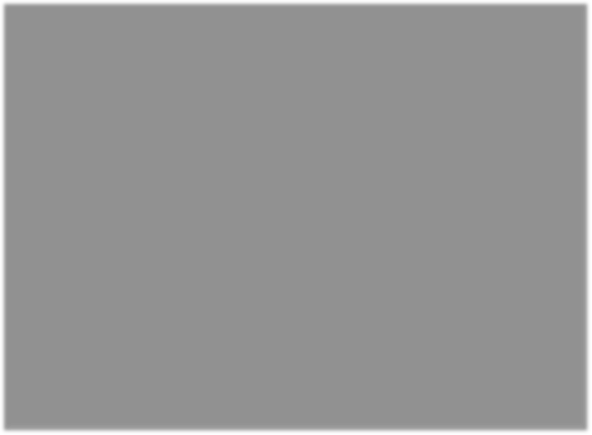
**Step 9:** No changes Needed, Click on Next.



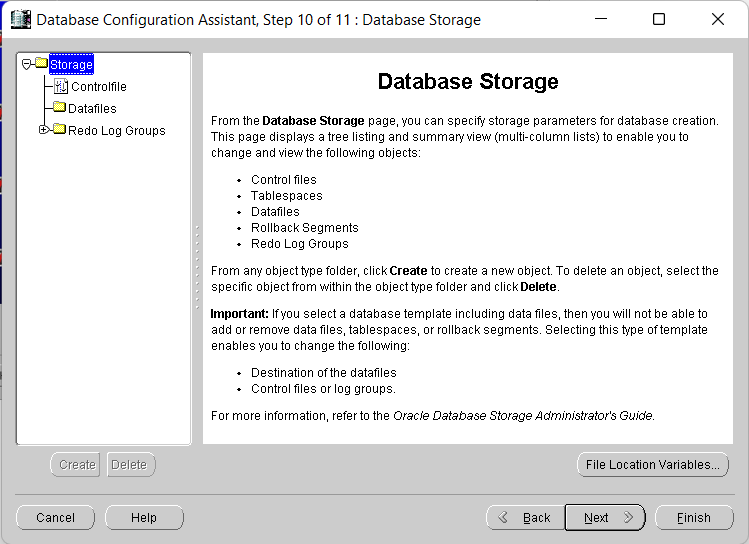
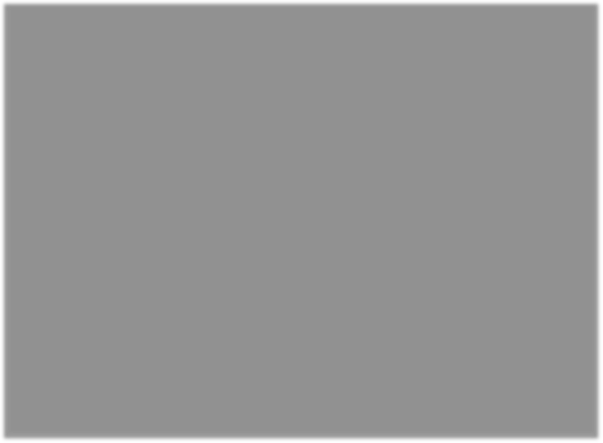
**Step 10:** No changes Needed, Click on Next



**Step 11:** No changes Needed, Click on Next.

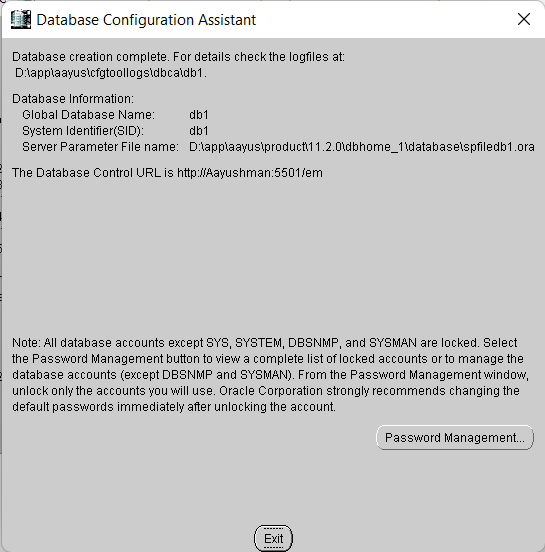
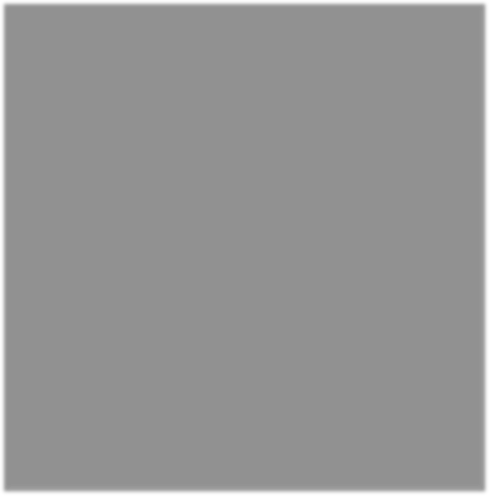
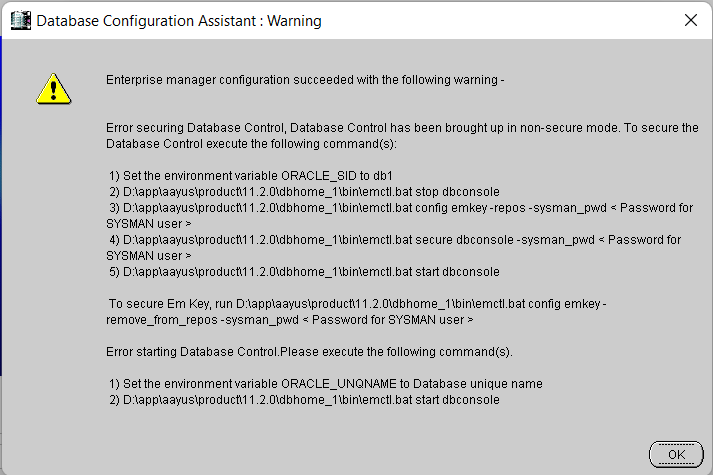
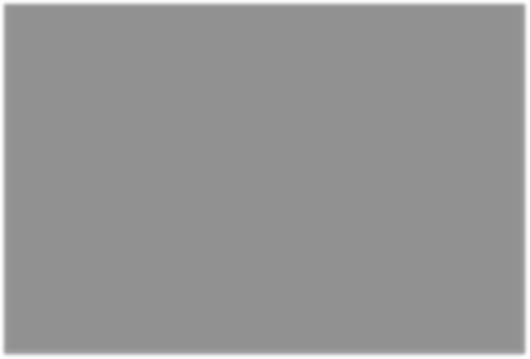
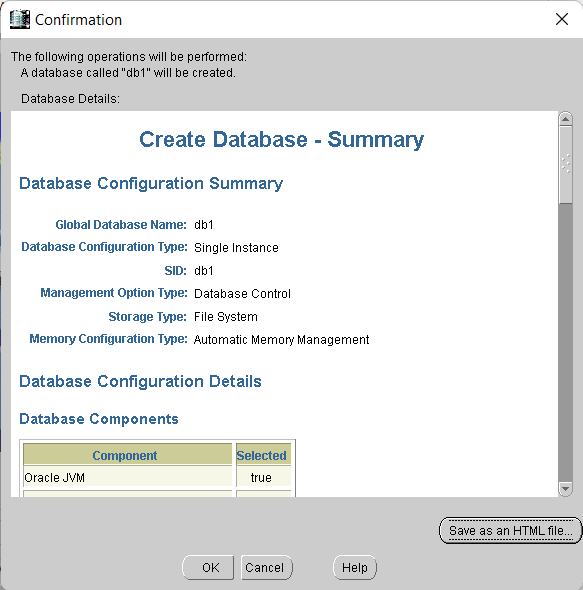
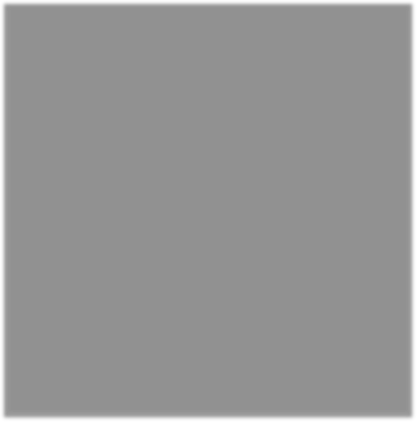
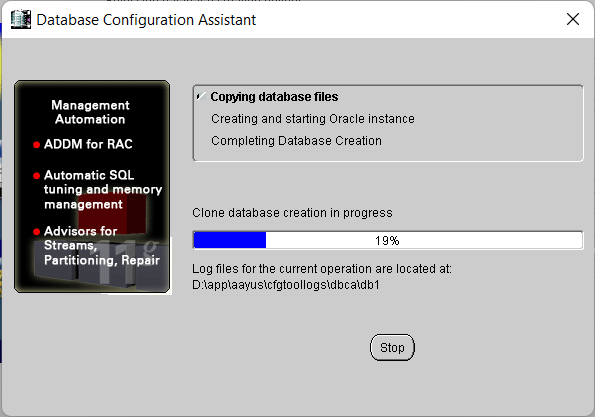
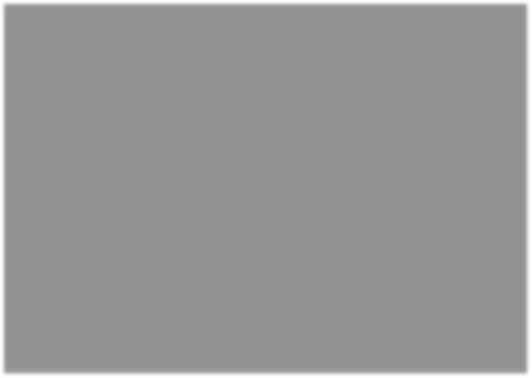
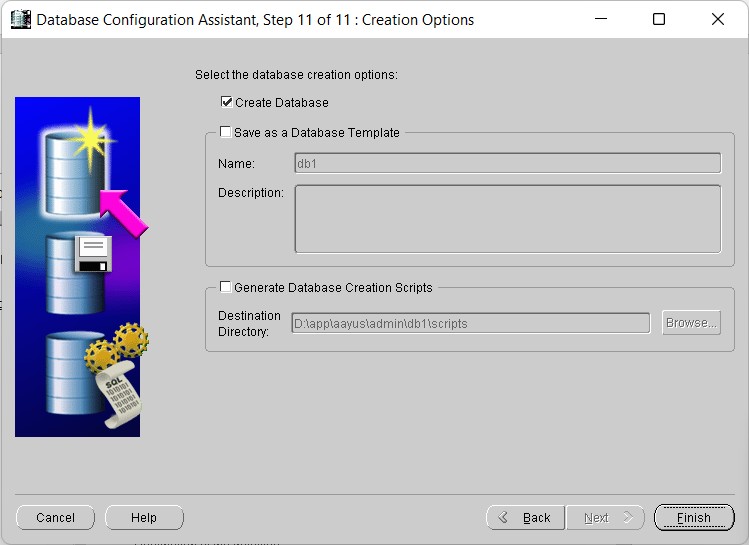
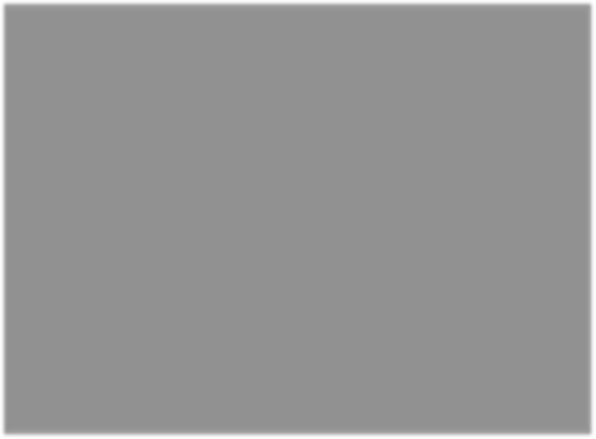


**Step 12:** No changes Needed, Click on Next..



**Step 13:** No changes Needed, Click on Finish.

Confirmation of Creating Database, You can Save it as well for your database details. Incase you forget credentials for your database, you can take help of this file to get access of your database.



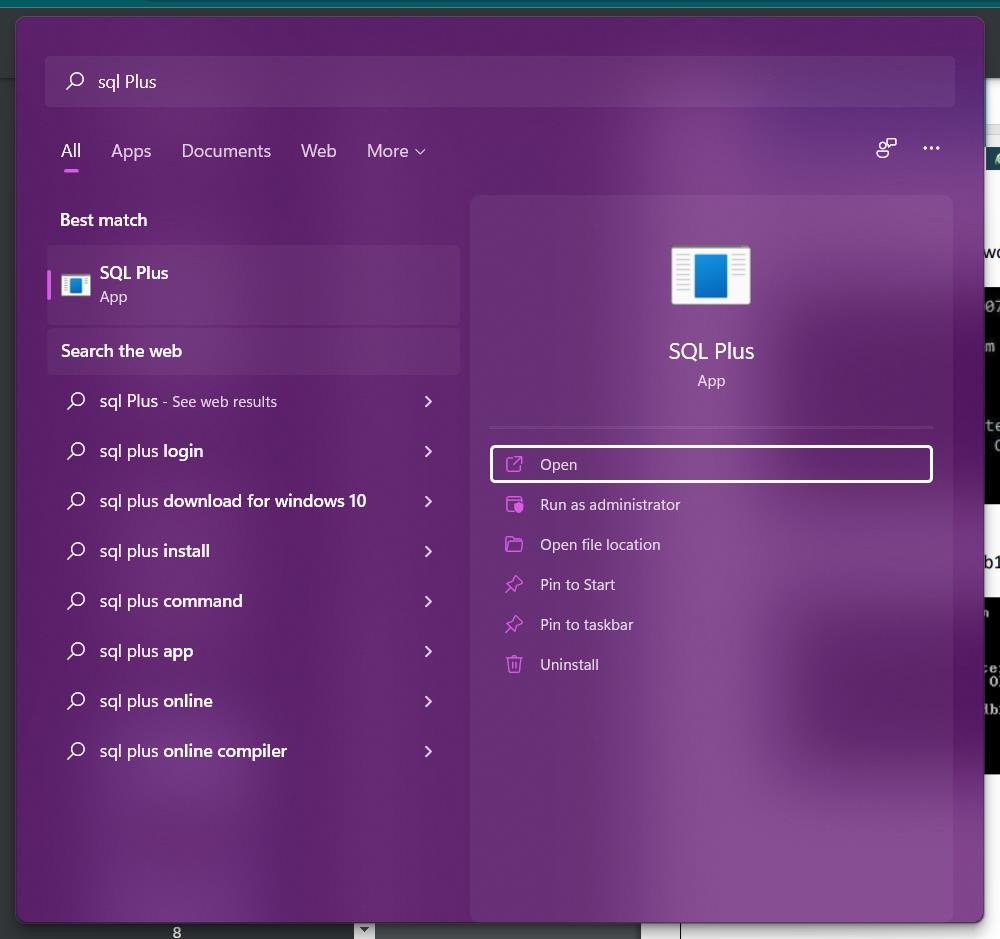
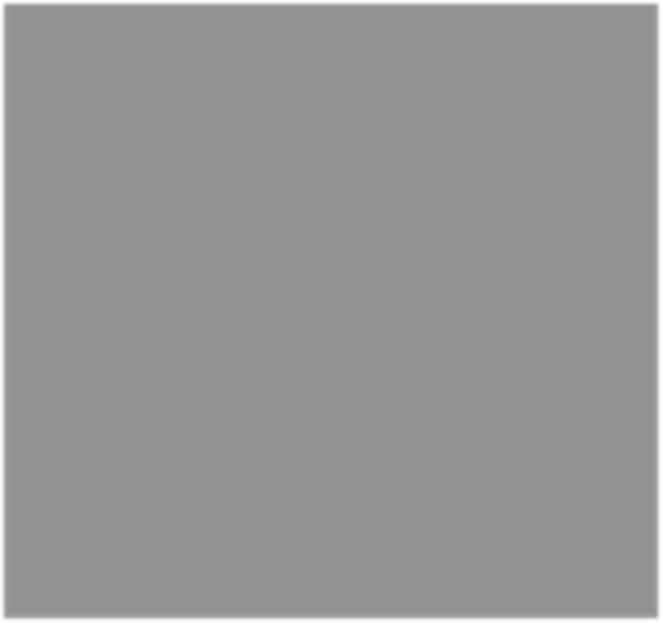
Click on Exit and Done…..

### Follow the Same Steps to create db2,

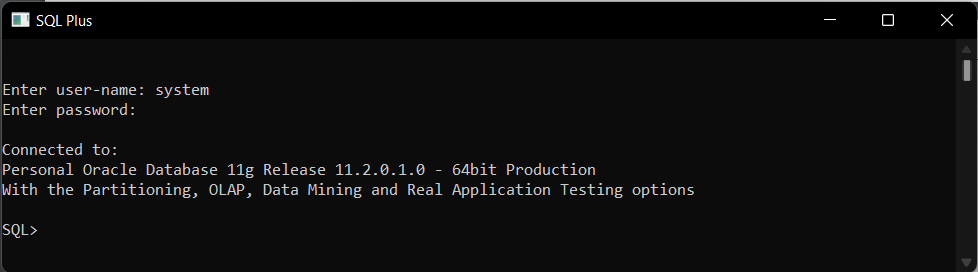
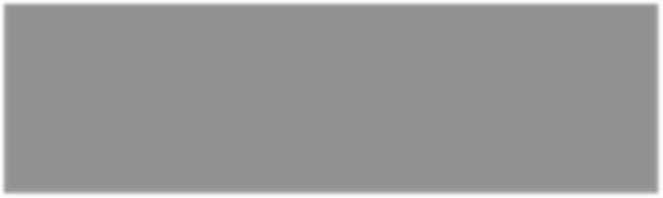
Once done with Creating db1 and db2 .

### Practical Implementation Steps:

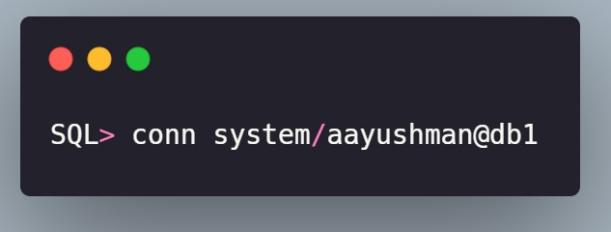
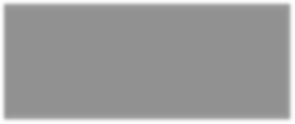
**Step 1:-** Open SQLPlus.



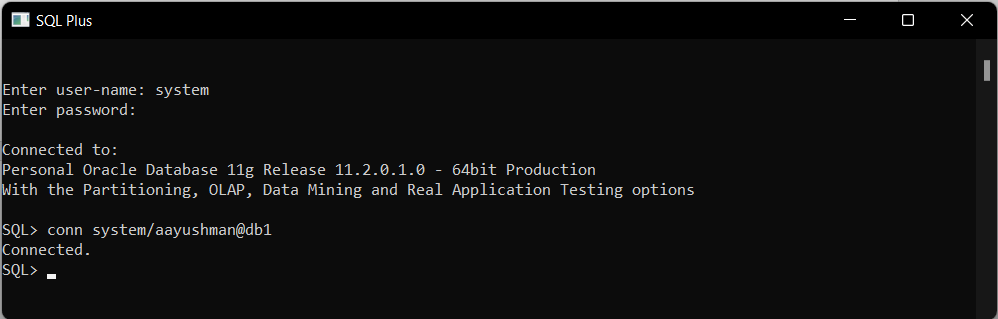
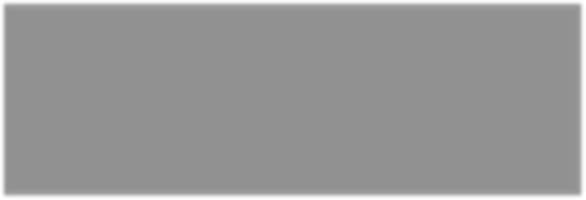
**Step 2:** Connect to Your Database .



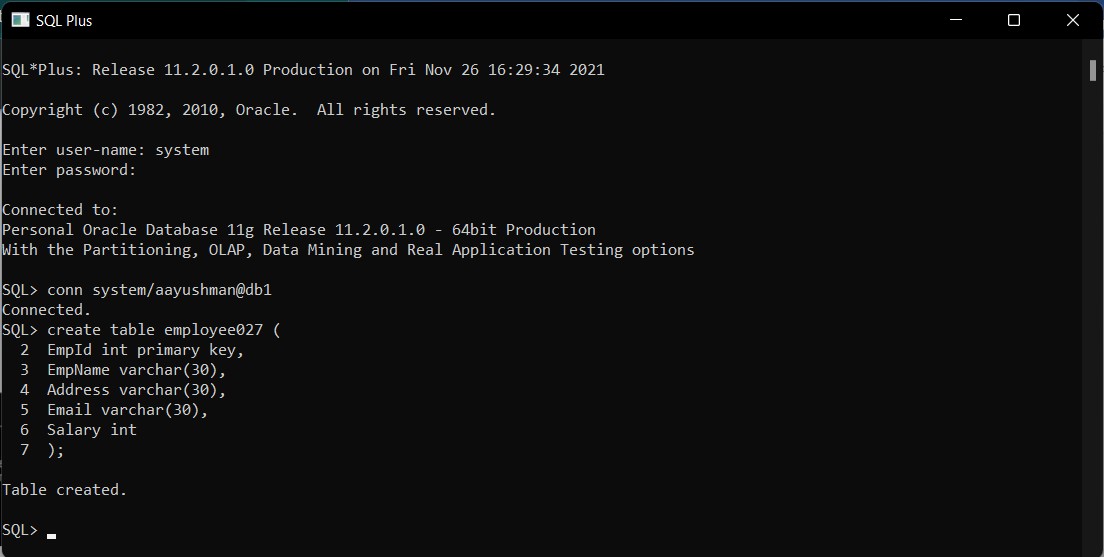
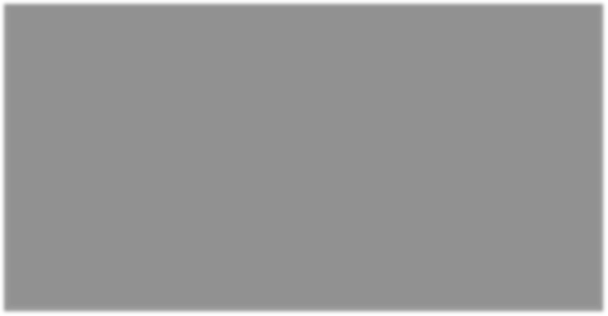
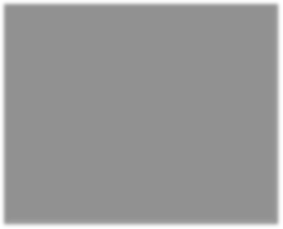
**Step 3:** Connect your db1 While executing the Command



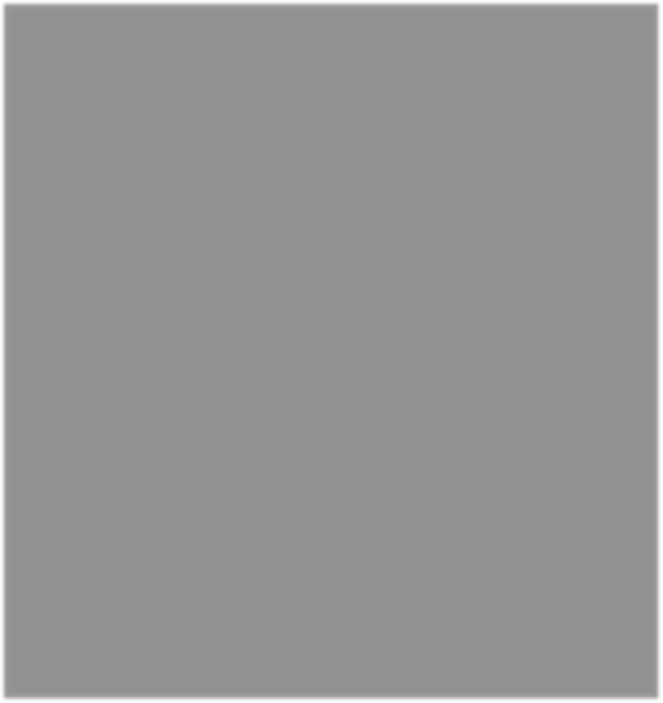
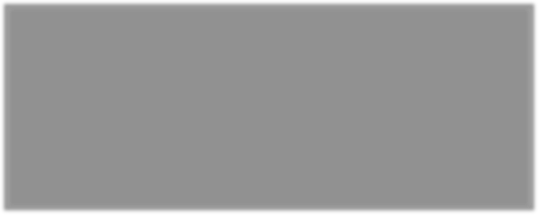
[Where “aayushman” is password of your database, and “db1” is database name].



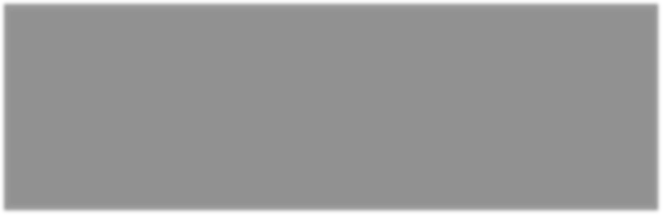
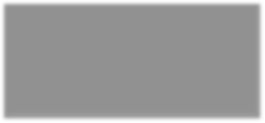
**Step 4:** Create one table in database db1.



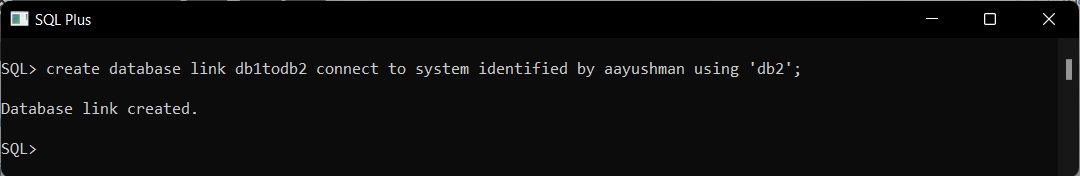
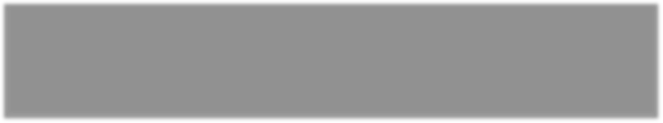
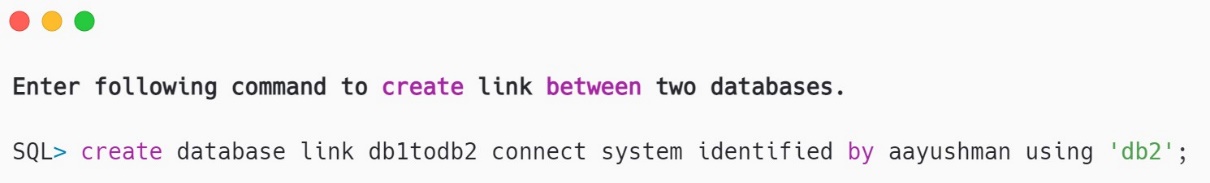
**Step 5:** Insert Some values in Created Table.



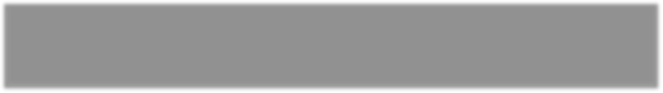
### Step 6:



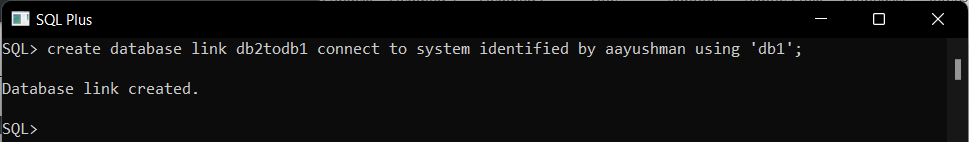
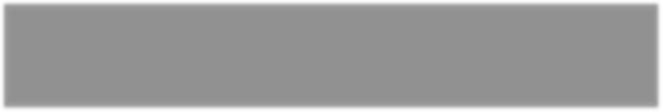
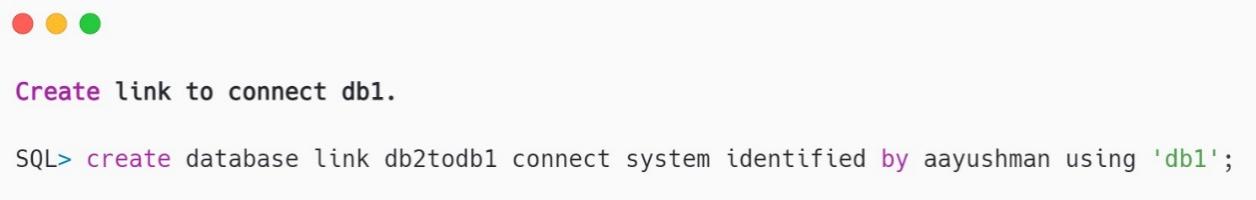
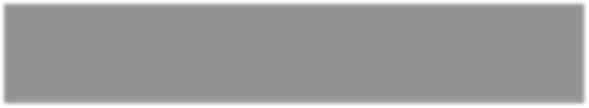
**Step 7:** Enter following command to create link between two databases.



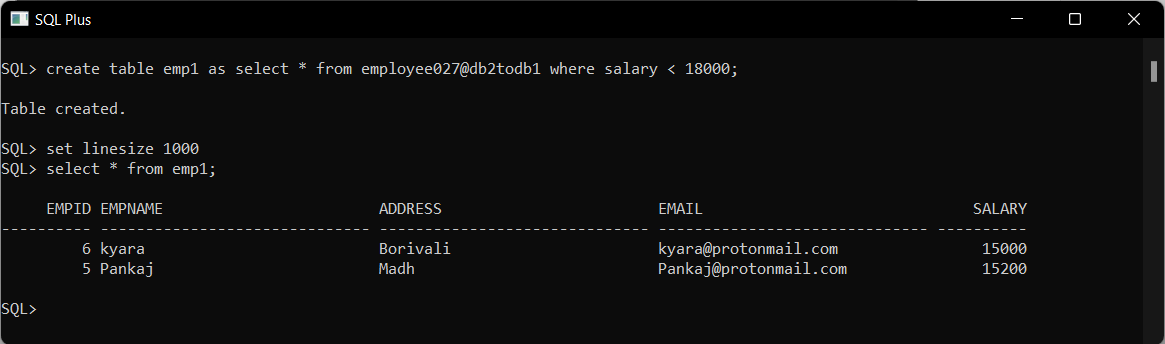
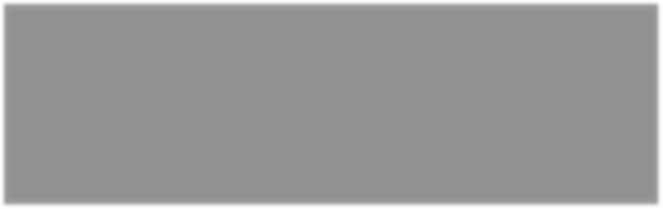
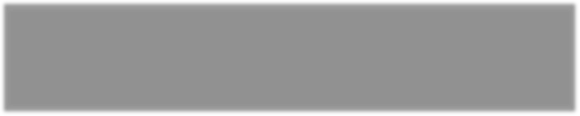
**Step 8:** Connect to Db2.



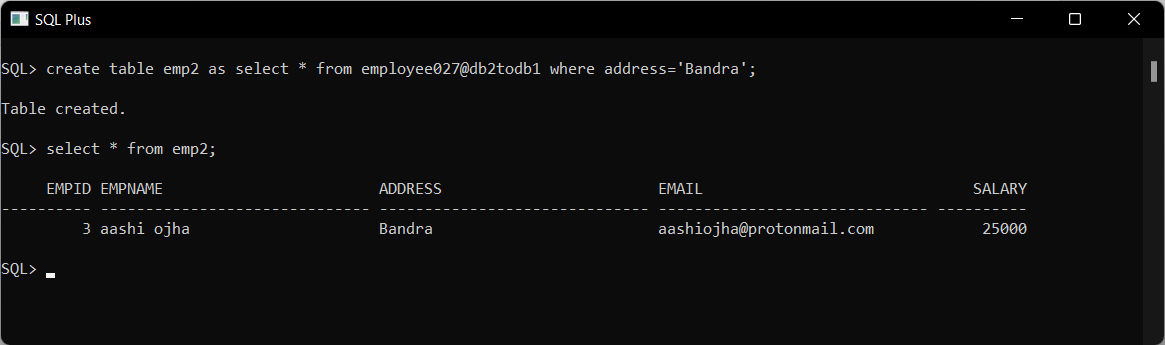
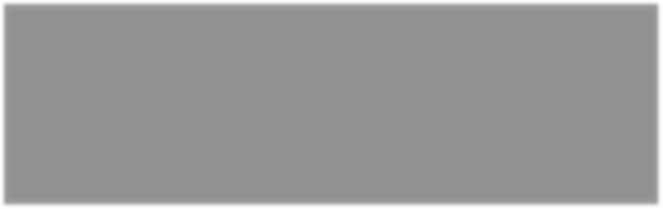
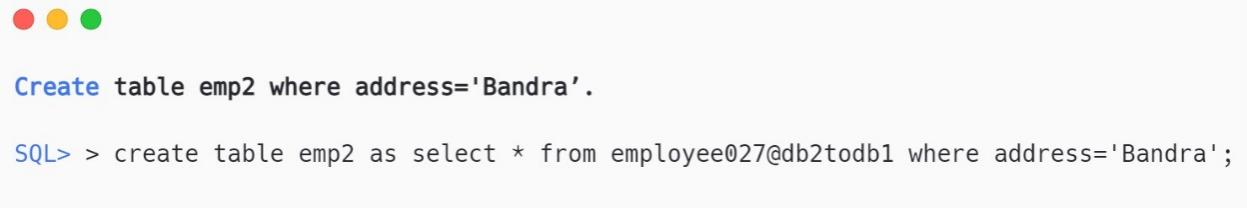
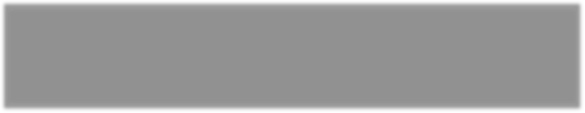
**Step 9:** Create link to connect db1.



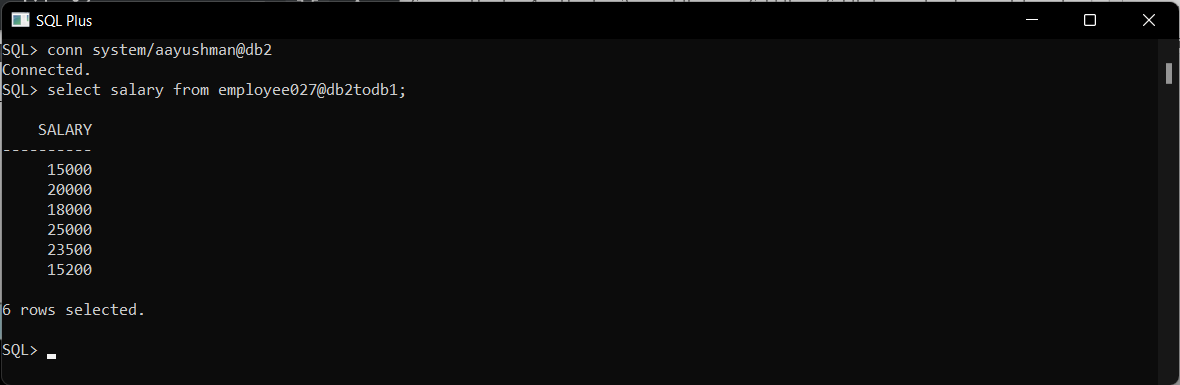
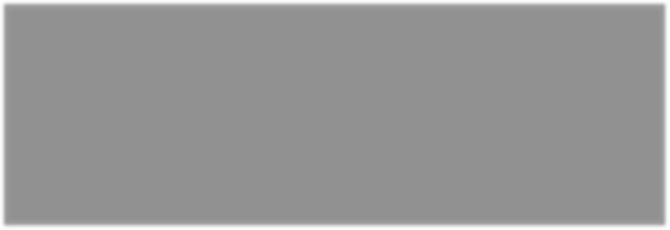
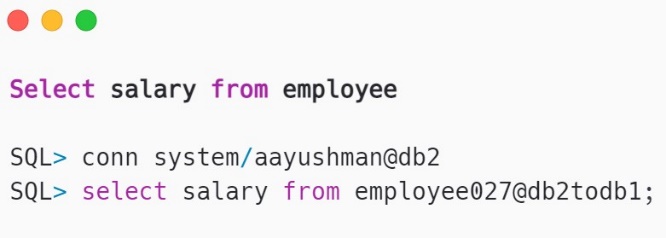
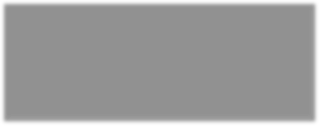
**Step 10:** Create emp1 select where salary<18000.



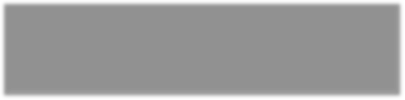
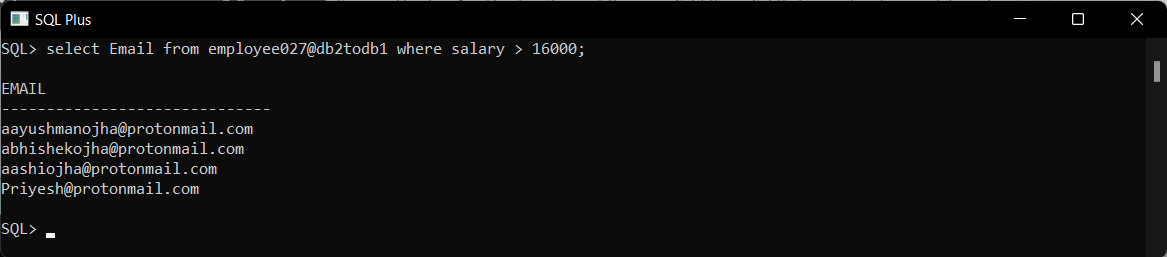
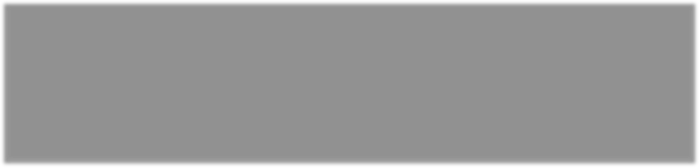
**Step 11:** Create table emp2 where address='Bandra’.



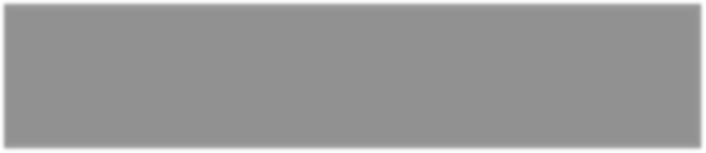
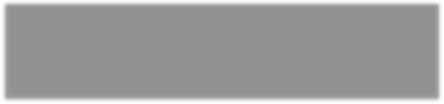
**Step 12:** Select salary from employee.



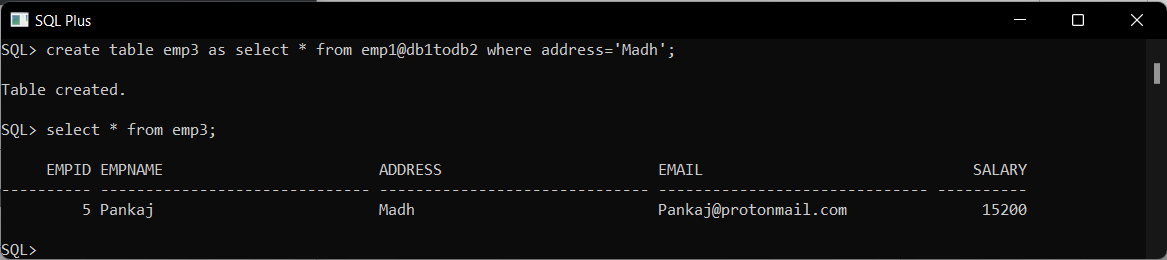
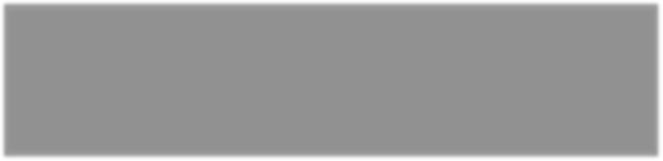
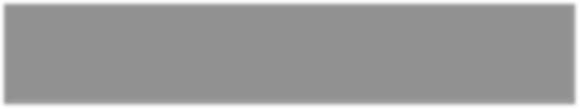
**Step 13:** Select mail whose salary>16000.



**Step 14:** Select Employee Name and Email from Employee table where eid=2.



**Step 15:** Create table emp3 where address='Madh'.



Conclusion:

Successfully executed Schema into horizontal and vertical Fragmentation on different nodes in Distributed Database Environment.

**PRACTICAL 2**

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

Theory:

The global conceptual schema, in the context of distributed databases, extends the concept of the conceptual schema to encompass the entire distributed database environment. It defines the structure and organization of data across multiple interconnected databases, which may be geographically distributed or managed by different organizations.

The global conceptual schema defines a unified view of data across multiple databases, taking into account data distribution, data replication, and data synchronization among the various components of the distributed system.

It is used to provide a consistent and coherent view of the data for applications and users, even when data is distributed across different physical locations or systems.

Software Requirements:

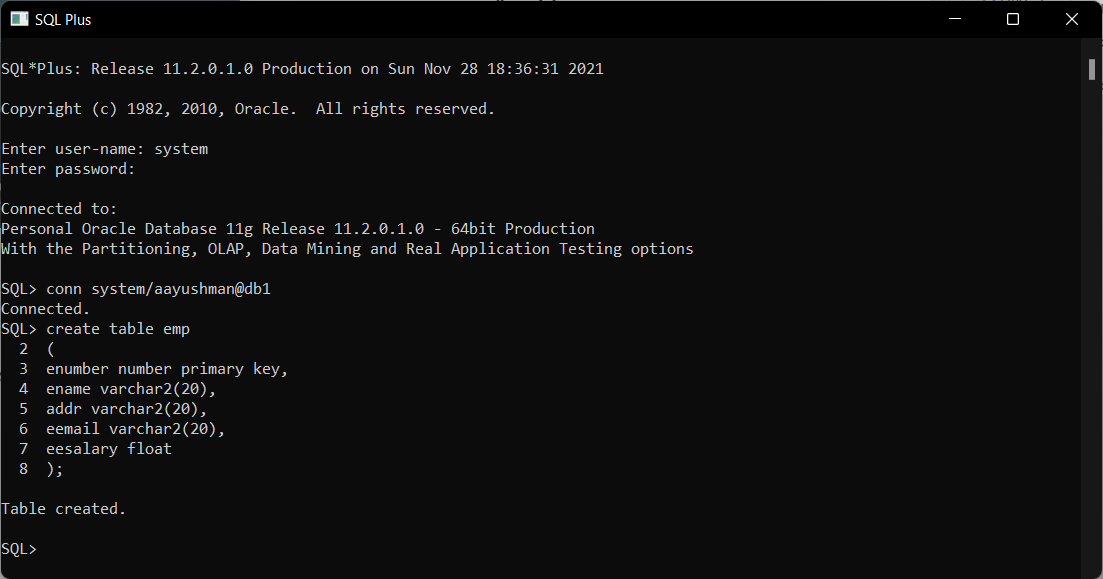
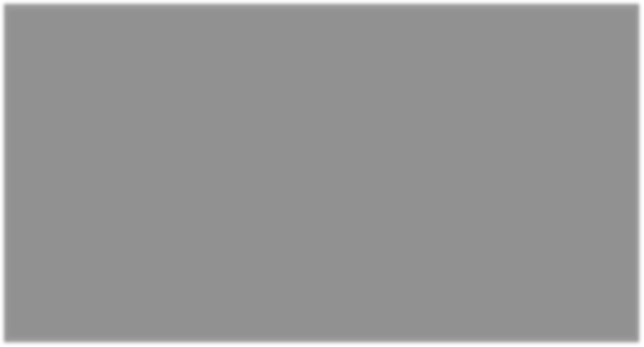
Oracle 11g.

### Query:

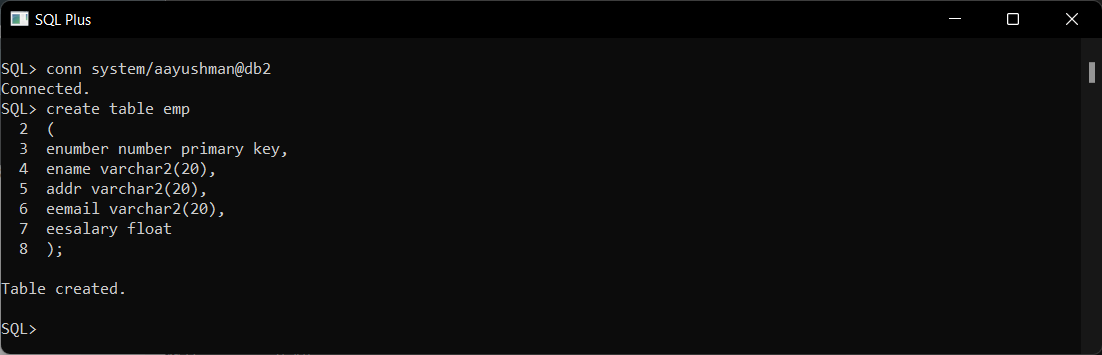
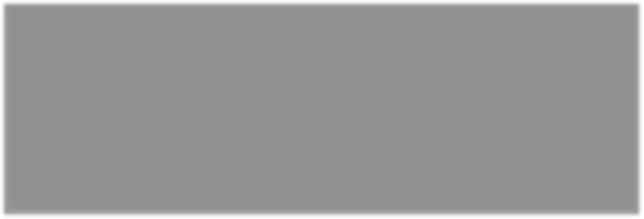
1. Update any record in db1 & show in db2
2. Delete any record in db1 & show in db2.
3. Find the salary of all employees.
4. Find the email of all employees where salary = 15000.
5. Find the employee name and email where employee number is known.
6. Find the employee name and address where employee number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

### Steps:

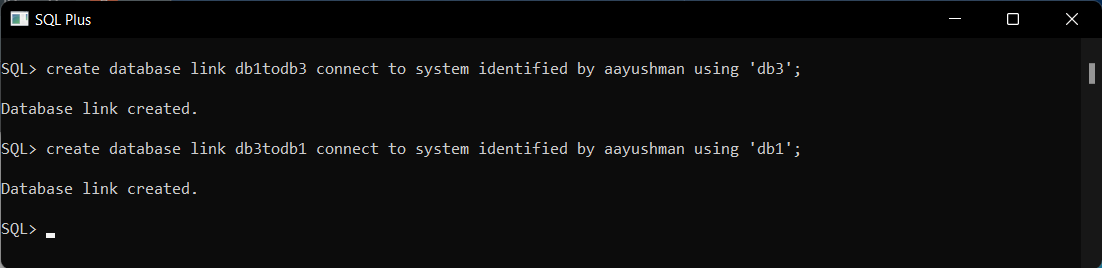
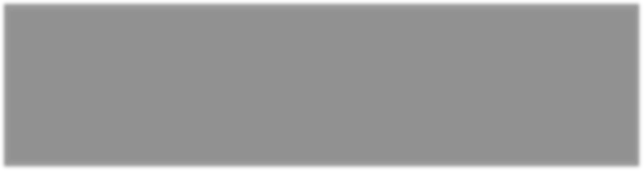
**Step 1:** Create Table in db1.



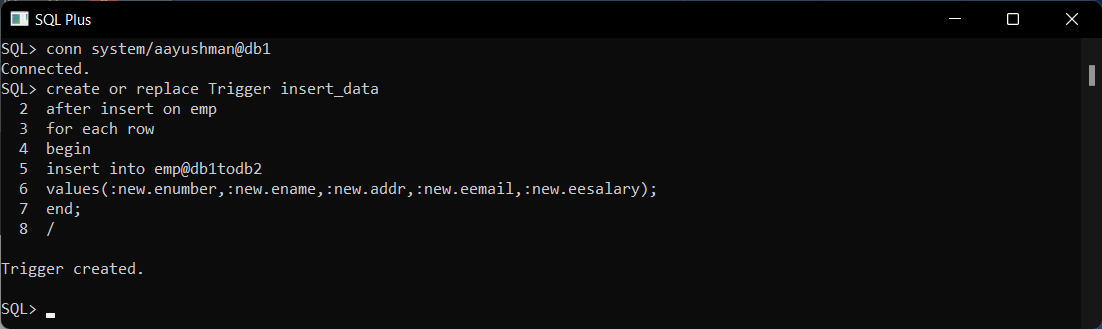
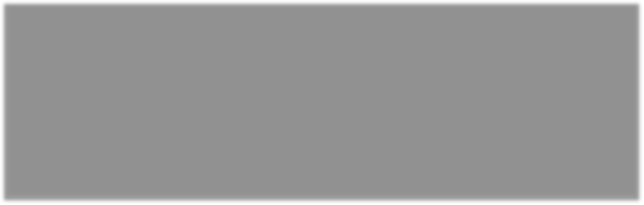
**Step 2:** Create Table in db2.



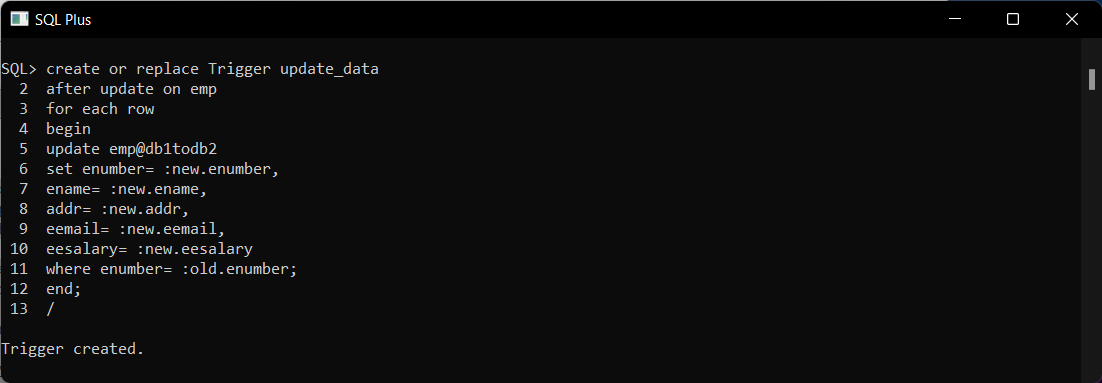
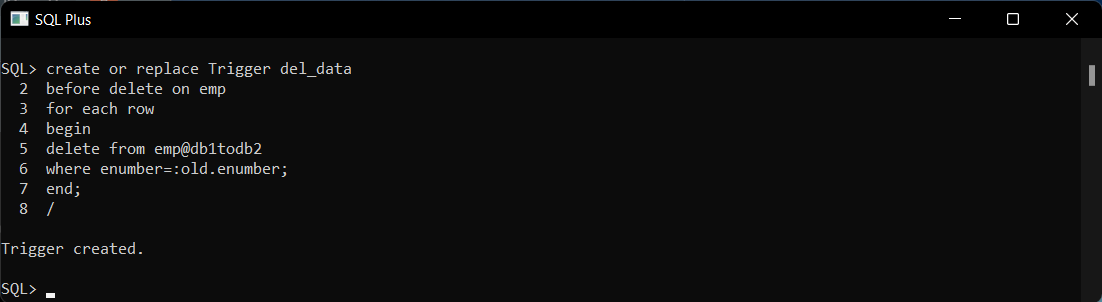
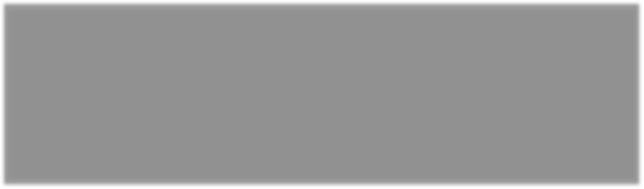
**Step 3:** Create Database link.



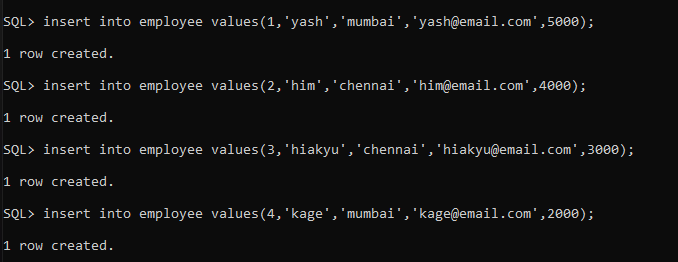
**Step 4:** Create Trigger to Insert Data.



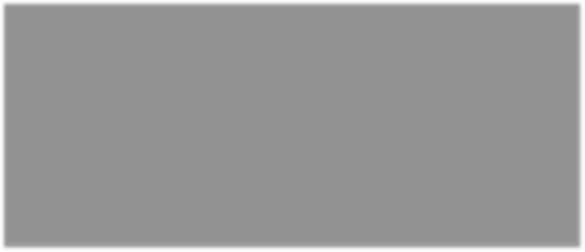
**Step 5:** Create Trigger to Update Data in Table.



**Step 6:** Insert Values in Created Table.

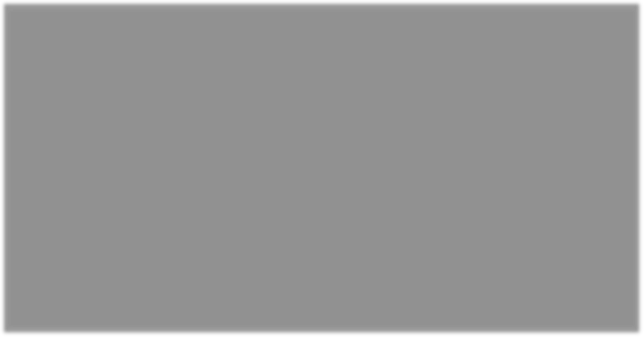


Show Create Tables.

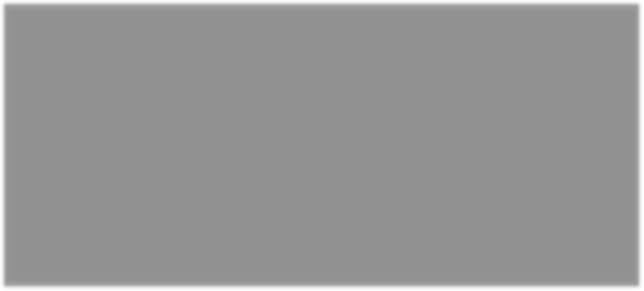


### Query

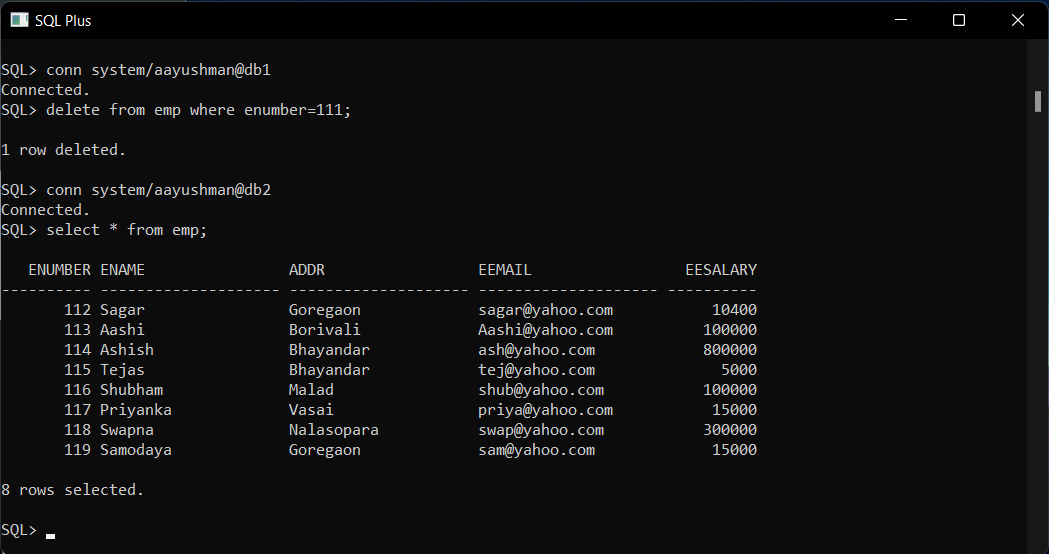
1. **Update any record in db1 & show in db2.**



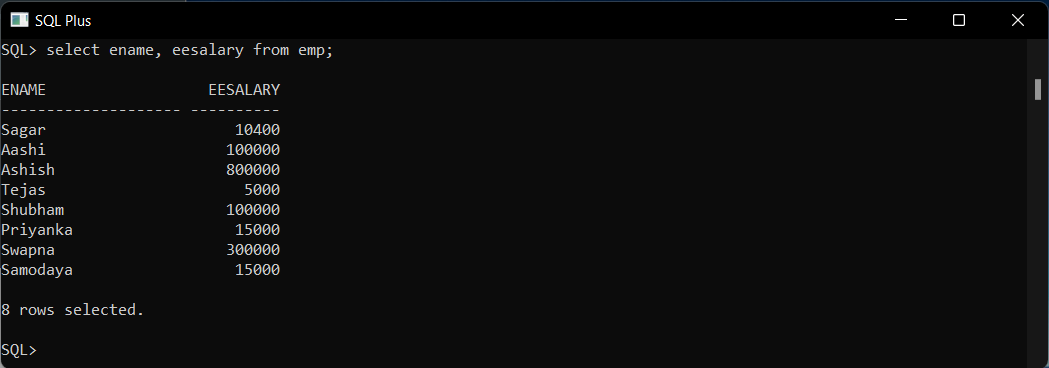
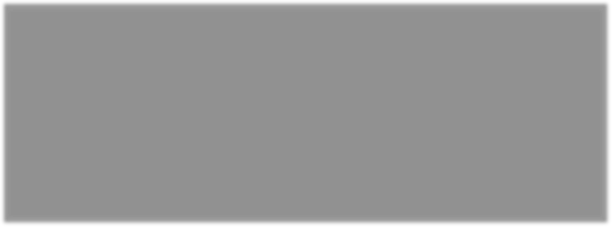
Show Updated Table in db2.



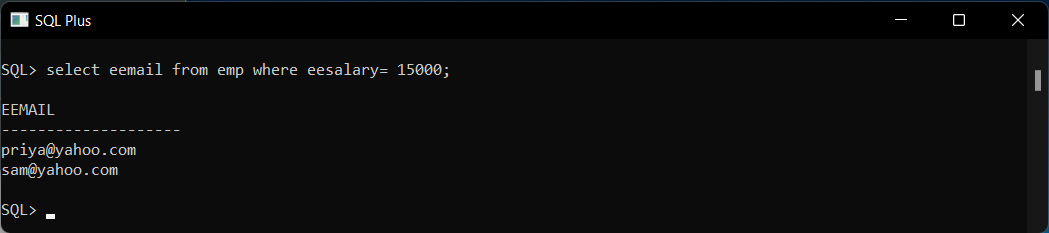
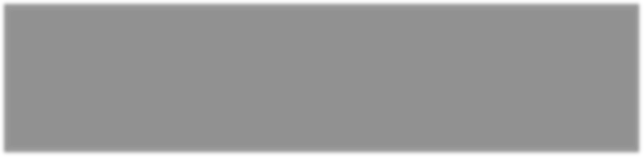
### Delete any record in db1 & show in db2.



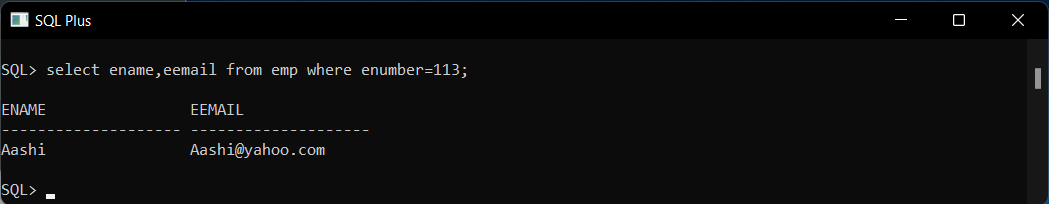
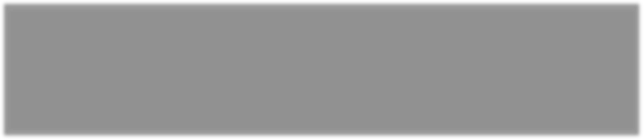
1. **Find the salary of all employees.**



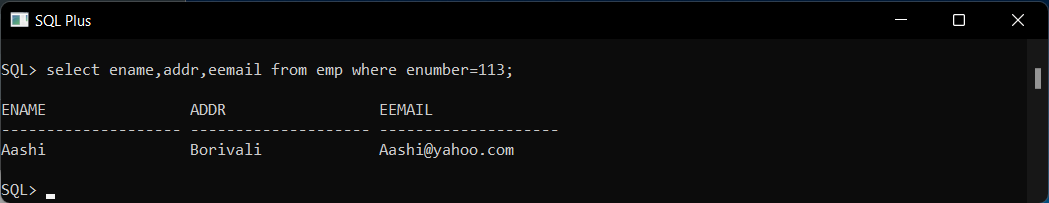
### Find the email of all employees where salary = 15000.



1. **Find the employee name and email where employee number is known.**



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



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

**PRACTICAL 3**

***Aim:-*** CRUD operation using MongoDB.

Theory:

MongoDB is a popular, open-source NoSQL database management system that is designed to store and manage large volumes of unstructured or semi-structured data. MongoDB is part of a class of databases known as document-oriented databases, which are designed to be flexible and highly scalable.

CRUD operations are fundamental actions used to manage data in a database:

1) Create (C): It involves adding new data records to the database. This is done using an "insert" operation, where new data is added to the database.

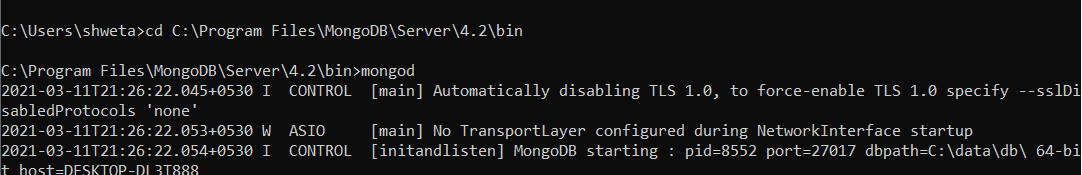
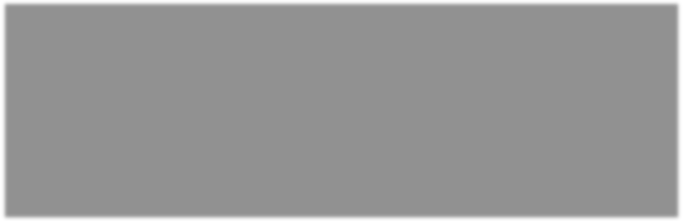
2) Read (R): It involves retrieving or reading data from the database. This is done using a "select" operation to retrieve data from the database.

3) Update (U): It involves modifying or updating existing data in the database. This is done using an "update" operation to change the values of existing records.

4) Delete (D): It involves removing data from the database. This is done using a "delete" operation to remove data records from the database.

Steps:

To run mongoDb from your command prompt:

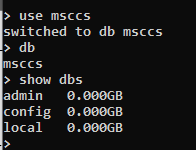


By performing the above command you are going to on your mongo DB serverNow to on your mongoDb shell :open new cmd

### mongo

Creating and selecting database;

### use msccs



To check the current connected database : **db**To see the list of all database : **show dbs**

### C : CREATING COLLECTION AND INSERTING VALUES :

Creating a collection and inserting values can be done together. Here we have or collection name as ‘student ‘

### db.student.insert({name:"sds"})

* **db.student.insert({no:2,name:"yash",course:{coursename:"bsc",duration:"3yrs"**

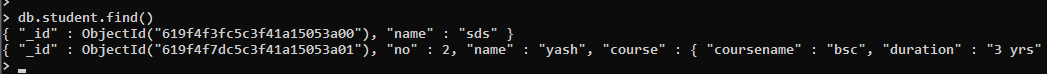
### },address:{city:"mumbai",state:"maharashtra",country:"india"}})



**R : READ DATA FROM THE COLLECTION :**

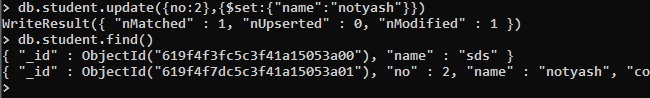
To retrieve the inserted document,

### db.student.find()



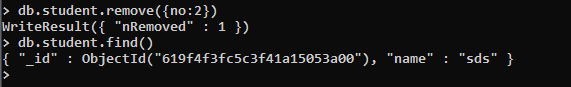
**U : UPDATING A DOCUMENT IN A COLLECTION :**

### db.student.update({no:2},{$set:{"name":"notyash"}})



**D : REMOVING AN ENTRY FROM THE COLLECTION (DELETE)**

### db.student.remove({no:2})



Conclusion: We have successfully performed CRUD operations.

**PRACTICAL 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.

Using Object Oriented databases create the following types:

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: BranchTableTypee) AuthorListType as varray, which is a reference to AuthorType

Next create the following tables:

1. BranchTableType of BranchType
2. authors of AuthorType
3. books(title: varchar, year : date,

published\_by ref PublisherType,authorsAuthorListType)

1. Publishers of PublisherType

Insert 10 records into the above tables and fire the following queries:

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. Name of authors who have not published a book
5. List all authors who have published more than one book:
6. Name of authors who have published books with at least two different publishers
7. 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 beenspecified).

Theory:

1) Insert: The "insert" operation is used to add new data or records to a database. It creates new entries in the database with specified values.

2) Update: The "update" operation is used to modify existing data or records in the database. It allows you to change the values of specific fields within a record.

3) Delete: The "delete" operation is used to remove data or records from the database. It deletes entries that meet specified criteria, effectively removing them from the database.

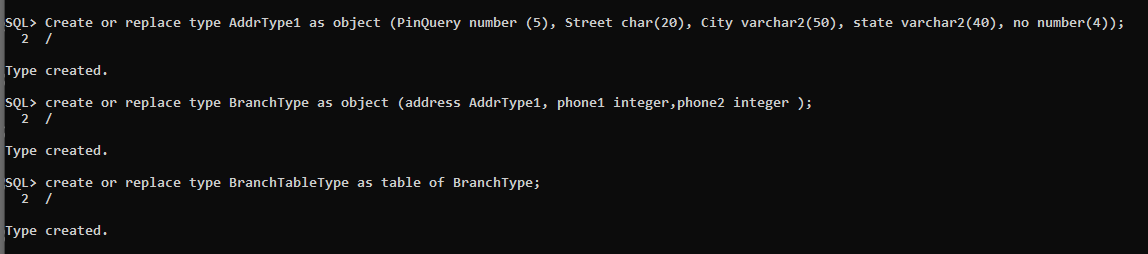
Steps:

SQL> **Create or replace type AddrType1 as object (PinQuery number (5), Street char(20), City varchar2(50), state varchar2(40), no number(4));**

SQL> **create or replace type BranchType as object (address AddrType1, phone1integer,phone2 integer**

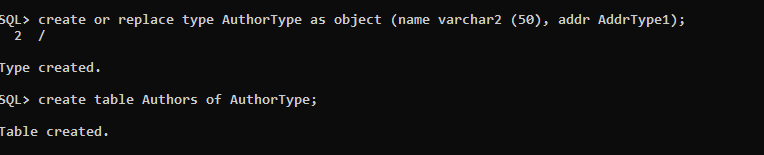
**);**

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

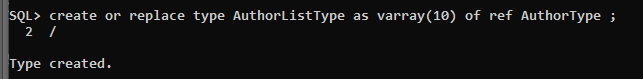


SQL> **create or replace type AuthorType as object (name varchar2 (50), addr AddrType1);**

SQL> **create or replace table Authors of AuthorType;**

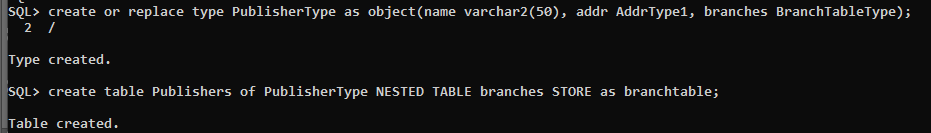


SQL> **create or replace type AuthorListType as varray(10) of ref AuthorType ;**

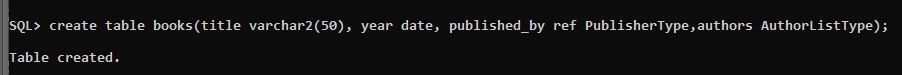


SQL> **create or replace type PublisherType as object(name varchar2(50), addr AddrType1,branches BranchTableType);**

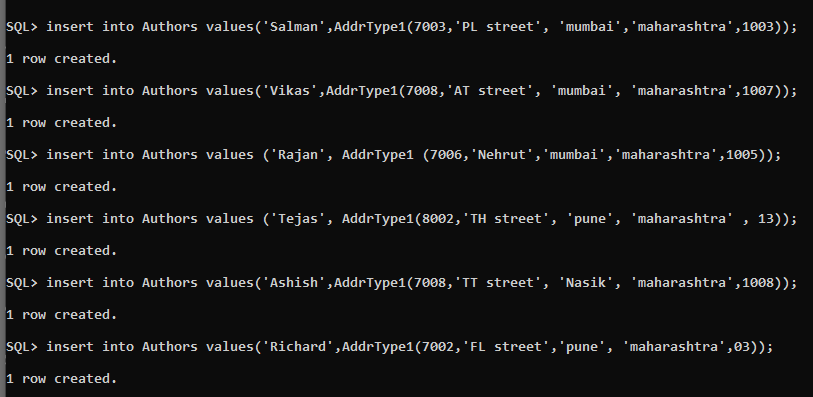
SQL> **create table Publishers of PublisherType NESTED TABLE branches STORE asbranchtable;**



SQL> **create table books(title varchar2(50), year date, published\_by ref PublisherType,authorsAuthorListType);**



* insert into Authors values('Aakash', AddrType1(7000,'AT street','mumbai','maharashtra',1007));
* insert into Authors values(‘abc’, AddrType1(7000,'AT street','mumbai','maharashtra',1007));
* insert into Authors values('Salman',AddrType1(7003,'PL street','mumbai','maharashtra',1003));
* insert into Authors values('Vikas',AddrType1(7008,'AT street', 'mumbai','maharashtra',1007));
* insert into Authors values ('Rajan', AddrType1(7006,'Nehrut','mumbai','maharashtra',1005));
* insert into Authors values ('Tejas', AddrType1(8002,'THstreet', 'pune', 'maharashtra' , 13));
* insert into Authors values('Ashish',AddrType1(7008,'TT street', 'Nasik','maharashtra',1008));
* insert into Authors values('Richard',AddrType1(7002,'FL street','pune','maharashtra',03));



insert into Publishers values ('Aakash', AddrType1 (4002,'PK street', 'mumbai','maharashtra',03), BranchTableType(BranchType (AddrType1(5002,'PL street','mumbai', 'maharashtra', 03), 23406,69896)));

insert into Publishers

values('McGraw',AddrType1(7007,'LJstreet','mumbai' ,'maharashtra',07), BranchTableType(BranchType ( AddrType1 (7007,'K street','mumbai', 'maharashtra',1007), 4543545,8676775)));

insert into Publishers values ('Tata',AddrType1(7008,'JW street','mumbai', 'maharashtra',27), BranchTableType (BranchType (AddrType1(1002,'DM street','nasik','maharashtra',1007), 456767,7675757)));

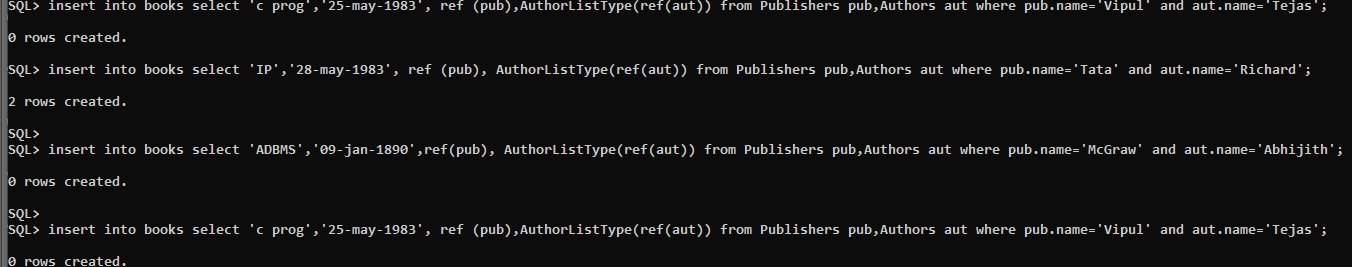
insert into Publishers values ('Nurali', AddrType1(7002,'ST street','pune','maharashtra',1007), BranchTableType (BranchType (AddrType1(1002,'SGstreet','pune', 'maharashtra',1007), 4543545,8676775)));

insert into Publishers values('Tata', AddrType1(6002,'Gold street','nasik', 'maharashtra',1007),BranchTableType (BranchType(AddrType1(6002,'South street','nasik','mha',1007), 4543545,8676775)));

insert into books select 'IP','28-may-1983', ref (pub), AuthorListType(ref(aut)) fromPublishers pub,Authors aut where pub.name='Tata' and aut.name='Richard';

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='Abhijith';

insert into books select 'c prog','25-may-1983', ref (pub),AuthorListType(ref(aut)) fromPublishers pub,Authors aut where pub.name='Vipul' and aut.name='Tejas';



### Firing Queries on the tables.

1. **List all of the authors that have the same pin Query as their publisher:**

Query:

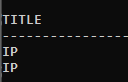
select a.name from Authors a, Publishers pwhere a.addr.pinQuery = p.addr.pinQuery;



1. **List all books that have 2 or more authors**

Query:

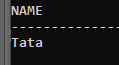
Select title from books b where 1 <= (select count(\*) from table(b.authors));



1. **List the name of the publisher that has the most branches**

Query:

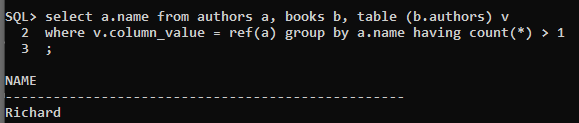
Select p.name from publishers p, table (p.branches) group by p.name having count(\*)> = all (selectcount(\*)from publishers p, table(p.branches) group by name);



1. **List all authors who have published more than one bookQuery:**

select a.name from authors a, books b, table (b.authors) v

where v.column\_value = ref(a) group by a.name having count(\*) > 1;



1. **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).**

**Query:**

select title from authors a, books b, table (b.authors) v wherev.column\_value = ref(a) group by title having count(\*)

> 1;

**Conclusion**:

Successfully implemented different attributes and special data types.

**PRACTICAL 5**

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

Theory:

A temporal database is a type of database system that is designed to handle and manage data with an explicit notion of time. In temporal databases, data is not just stored with a current timestamp; it also includes information about when data was valid, how it changed over time, and historical records of past states. Temporal databases are especially useful for applications that need to track and analyze changes in data over time, such as historical records, versioning systems, and financial systems.

Steps:

**Create table:**

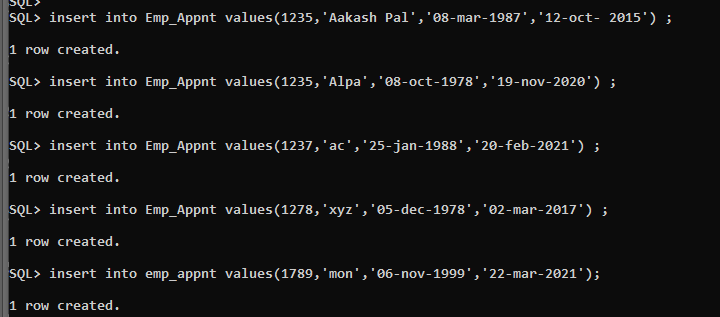
SQL> create table Emp\_Appnt( Acc\_Nonumber(10),Name varchar2(10), RECDate date, RETDate date);



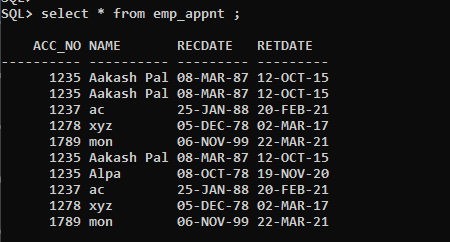
### Inserting rows :

insert into Emp\_Appnt values(1235,'Aakash Pal','08-mar-1987','12-oct-2015') ; insert into Emp\_Appnt values(1235,'Alpa','08-oct-1978','19-nov-2020') ;

insert into Emp\_Appnt values(1237,'ac','25-jan-1988','20-feb-2021') ; insert into Emp\_Appnt values(1278,'xyz','05-dec-1978','02-mar-2017') ; insert into emp\_appnt values(1789,'mon','06-nov-1999','22-mar-2021');

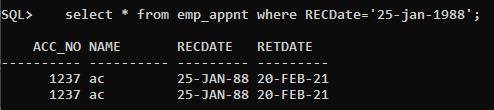


SQL> select \* from emp\_appnt ;



### Queries:

* 1. Show the employee whose record date is 25th Jan-1988. select \* from emp\_appnt where RECDate='25-jan-1988';



* 1. Show the employee whose record date is 22th mar-2021. select \* from emp\_appnt where RETDate='22-mar-2021';



* 1. Create a new table named as tbl\_shares.

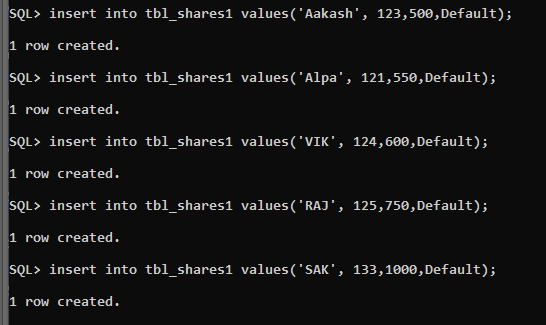
create table tbl\_shares1(

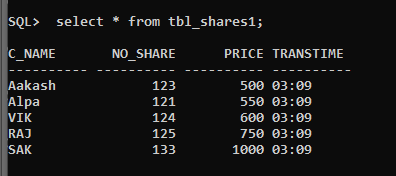
C\_Name varchar2(10), No\_Share Number(10), Price number(10), TransTime varchar2(10) Default To\_char(sysdate,'HH:MI'));

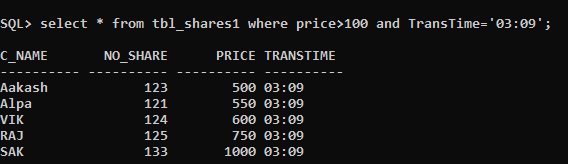


### Inserting rows :

insert into tbl\_shares1 values('Aakash', 123,500,Default); insert into tbl\_shares1 values('Alpa', 121,550,Default); insert into tbl\_shares1 values('VIK', 124,600,Default); insert into tbl\_shares1 values('RAJ', 125,750,Default); insert into tbl\_shares1 values('SAK', 133,1000,Default);

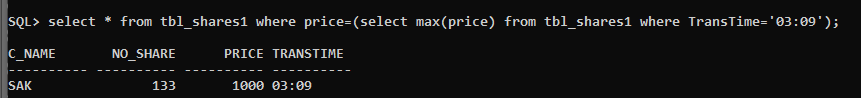


* 1.  Display all the records you have entered in table. select \* from tbl\_shares;
  2. Display records where price>100 and TransTime='01:24'. select \* from tbl\_shares where price>100 and TransTime='01:24';



* 1. Display the recordswhere price=(select max(price) from tbl\_shares where TransTime='02:04');

select \* from tbl\_shares1 where price=(select max(price) from tbl\_shareswhere TransTime='01:25');

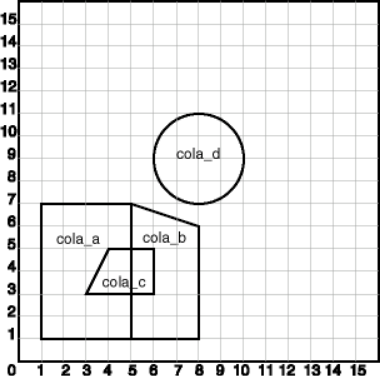


Conclusion: We have successfully created a temporal database.

**PRACTICAL 6**

***Aim:-*** Create a table that stores spatial data and issue queries on it.

Theory:

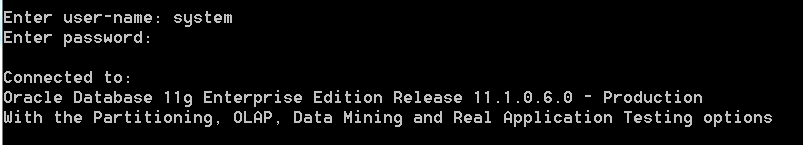
Spatial data, also known as geospatial data, is information that is associated with specific geographic locations on the Earth's surface. It represents data that has a spatial or geographical component, which can be expressed in terms of coordinates, shapes, boundaries, or distances. Spatial data is used to describe and analyze the physical world, including the locations of objects, features, and phenomena in the context of geography and cartography.

Query:

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

Fire the following queries:

* + 1. Find the topological intersection of two geometries.
    2. Find whether two geometric figures are equivalent to each other.
    3. Find the areas of all different locations.
    4. Find the area of only one location.
    5. Find the distance between two geometries.

Steps:

**Query for Creating Table**

SQL>create table cola\_mrp(

mkt\_id number primary key, name varchar(20), shapeMDSYS.SDO\_Geometry

);

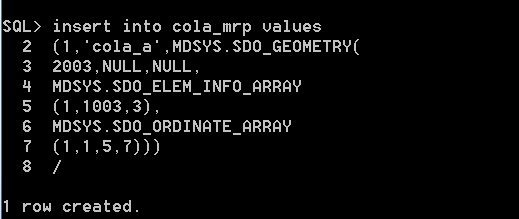
### Queries for inserting rows :

1)

SQL> insert into cola\_mrp values (1,'cola\_a',MDSYS.SDO\_GEOMETRY( 2003,NULL,NULL, MDSYS.SDO\_ELEM\_INFO\_ARRAY (1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY (1,1,5,7)))

/

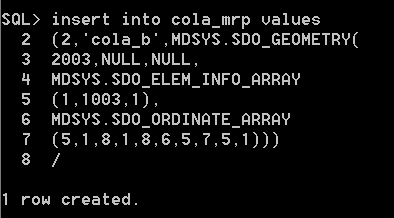


2)

SQL>insert into cola\_mrp values (2,'cola\_b',MDSYS.SDO\_GEOMETRY( 2003,NULL,NULL, MDSYS.SDO\_ELEM\_INFO\_ARRAY (1,1003,1),

MDSYS.SDO\_ORDINATE\_ARRAY (5,1,8,1,8,6,5,7,5,1)))

/

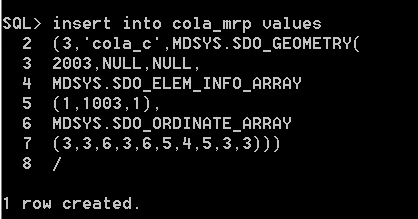


3)

SQL>insert into cola\_mrp values (3,'cola\_c',MDSYS.SDO\_GEOMETRY( 2003,NULL,NULL, MDSYS.SDO\_ELEM\_INFO\_ARRAY (1,1003,1),

MDSYS.SDO\_ORDINATE\_ARRAY (3,3,6,3,6,5,4,5,3,3)))

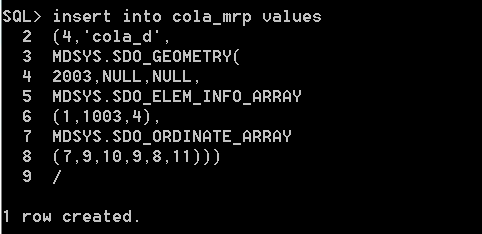
/



4)

SQL>insert into cola\_mrp values (4,'cola\_d', MDSYS.SDO\_GEOMETRY( 2003,NULL,NULL, MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,4), MDSYS.SDO\_ORDINATE\_ARRAY (7,9,10,9,8,11)))

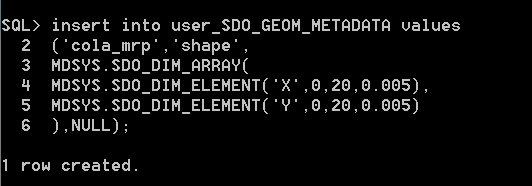
/



### Creating Metadata information:

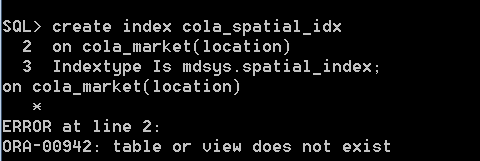
SQL>insert into user\_SDO\_GEOM\_METADATA values('cola\_mrp','shape', MDSYS.SDO\_DIM\_ARRAY( MDSYS.SDO\_DIM\_ELEMENT('X',0,20,0.005), MDSYS.SDO\_DIM\_ELEMENT('Y',0,20,0.005)

),NULL);



### Query for creating index :

SQL>create index cola\_spatial\_idx oncola\_market(location) Indextype Is mdsys.spatial\_index;

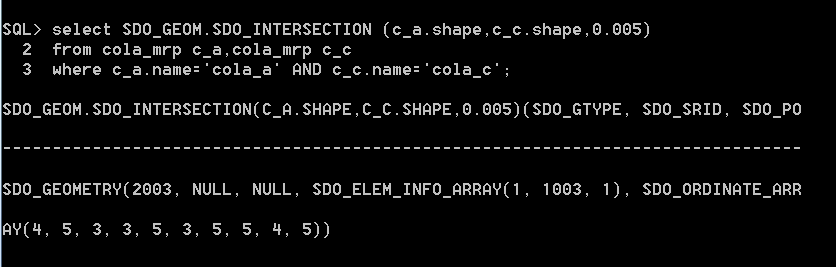


### Queries :

1. Find the topological intersection of two geometries.

SQL>select SDO\_GEOM.SDO\_INTERSECTION (c\_a.shape,c\_c.shape,0.005) fromcola\_mrpc\_a,cola\_mrpc\_c

where c\_a.name='cola\_a' AND c\_c.name='cola\_c';

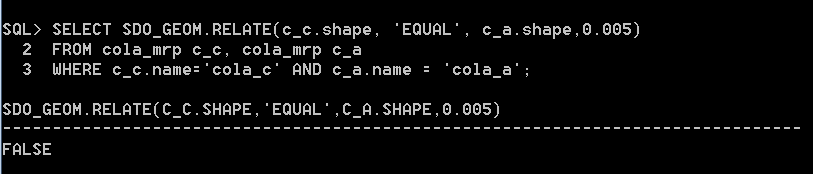


1. Find whether two geometric figures are equivalent to each other.

SQL>SELECT SDO\_GEOM.RELATE(c\_c.shape, 'EQUAL', c\_a.shape,0.005)

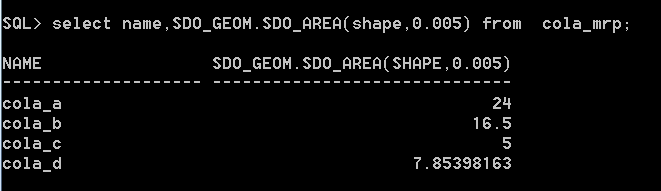
FROM cola\_mrpc\_c, cola\_mrpc\_a

WHERE c\_c.name='cola\_c' AND c\_a.name = 'cola\_a';



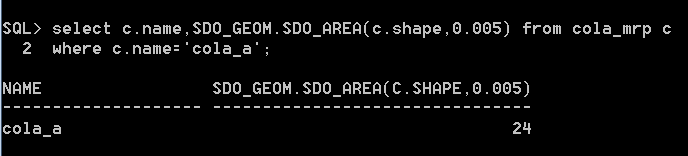
1. Find the areas of all different locations

SQL>select name,SDO\_GEOM.SDO\_AREA(shape,0.005) from cola\_mrp;



1. Find the area of only one location.

SQL>select c.name,SDO\_GEOM.SDO\_AREA(c.shape,0.005) from cola\_mrp cwhere c.name='cola\_a';



1. Find the distance between two geometries.

SQL>select SDO\_GEOM.SDO\_DISTANCE(c\_b.shape,c\_d.shape,0.005) fromcola\_mrpc\_b,cola\_mrpc\_d

where c\_b.name= 'cola\_b' AND c\_d.name ='cola\_d';



**Conclusion**:

We have successfully created a Spatial database.

**PRACTICAL 7**

***Aim:-*** Create a table employee having dept\_id as number datatype and employee\_spec as XML data type (XML\_Type). The employee\_spec is a schema with attributes emp\_id, name, email, acc\_no, managerEmail, dataOf Joning. Insert 10 tuples into employee table. Fire the following queries on XML database:

Insert 10 tuples into employee table. Fire the following queries on XML database.

1. Retrieve the names of employee.
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

Theory:

An XML database is a type of database management system (DBMS) designed specifically for the storage, retrieval, and management of XML (eXtensible Markup Language) data. XML is a text-based data format commonly used for representing structured and semi-structured data, making it suitable for various applications, including web services, data interchange, and document representation.

Steps:

Create a table:

create table emp (emp\_id int, emp\_spec xmltype);

Insert records:

Insert into emp values (1,xmltype('<?xml version="1.0"?>

<employee id="be130">

<firstname>William</firstname>

<lastname>Defoe</lastname>

<title>Accountant</title>

<division>Accts Payable</division>

<building>326</building>

<room>14a</room>

</employee>’));

Insert into emp values (2,xmltype('<?xml version="1.0"?>

<employee id="be129">

<firstname>Jane</firstname>

<lastname>Doe</lastname>

<title>Engineer</title>

<division>Materials</division>

<building>327</building>

<room>19</room>

<supervisor>be131</supervisor>

</employee>’));

Insert into emp values (3,xmltype('<?xml version="1.0"?>

<employee id="be129">

<firstname>Jane</firstname>

<lastname>Doe</lastname>

<title>Engineer</title>

<division>Materials</division>

<building>327</building>

<room>19</room>

<supervisor>be131</supervisor>

</employee>’));

Insert into emp values (4,xmltype('<?xml version="1.0"?>

<employee id="be132">

<firstname>Sandra</firstname>

<lastname>Rogers</lastname>

<title>Engineering</title>

<division>Materials</division>

<building>327</building>

<room>22</room>

</employee>’));

Insert into emp values (5,xmltype('<?xml version="1.0"?>

<employee id="be133">

<firstname>Steve</firstname>

<lastname>Casey</lastname>

<title>Engineering</title>

<division>Materials</division>

<building>327</building>

<room>24</room>

</employee>’));

Insert into emp values (6,xmltype('<?xml version="1.0"?>

<employee id="be135">

<firstname>Michelle</firstname>

<lastname>Michaels</lastname>

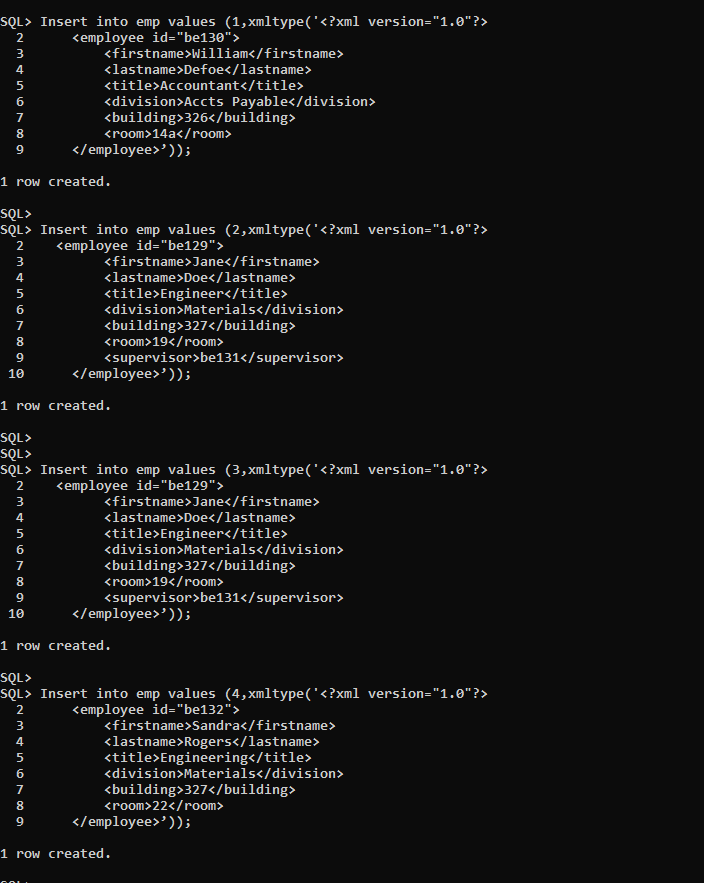
<title>COO</title>

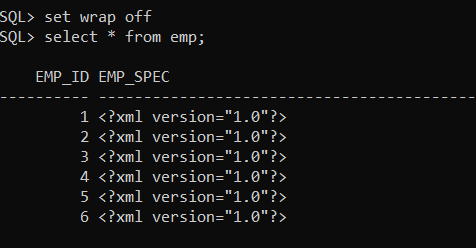
<division>Management</division>

<building>216</building>

<room>264</room>

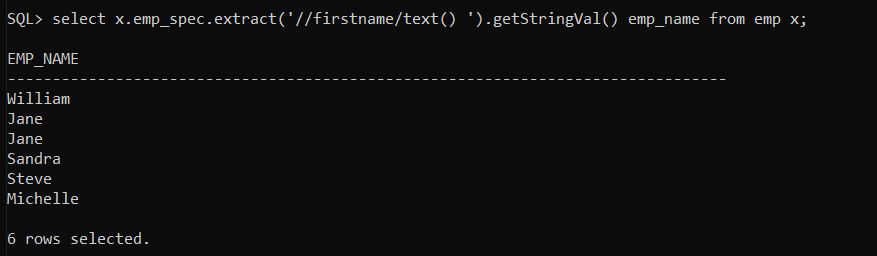
</employee>’));





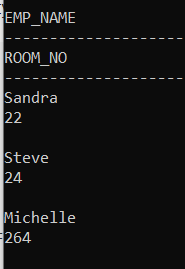
Get the first name:

select x.emp\_spec.extract('//firstname/text() ').getStringVal() from emp x;



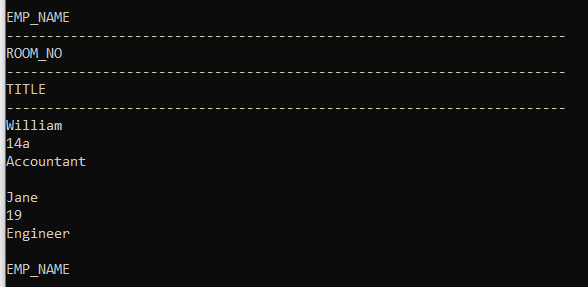
## Get the first name and room number:

select x.emp\_spec.extract('//firstname/text() ').getStringVal() emp\_name, x.emp\_spec.extract('//room/text()').getStringVal() room\_No from emp x;



## Get the first name and room number and title:

select x.emp\_spec.extract('//firstname/text() ').getStringVal() emp\_name, x.emp\_spec.extract('//room/text()').getStringVal() room\_No from emp x;



## Update 3rd record from the table:

Update emp set emp\_spec=xmltype('<?xml version="1.0"?>

<employee id="be135">

<firstname>NotMichelle</firstname>

<lastname>NotMichaels</lastname>

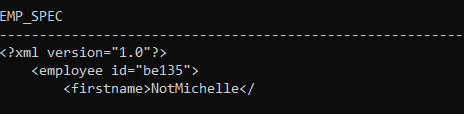
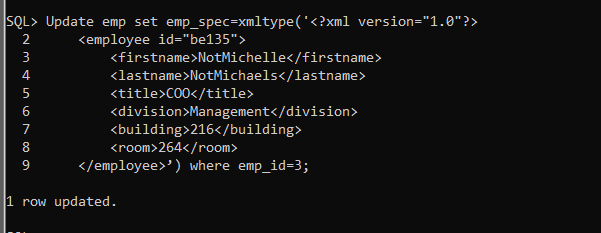
<title>COO</title>

<division>Management</division>

<building>216</building>

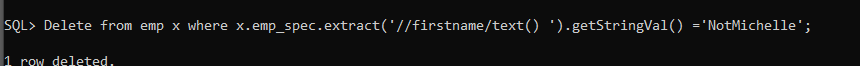
<room>264</room>

</employee>’) where emp\_id=3;



## Delete a record from the table:

Delete from emp where x.emp\_spec.extract('//firstname/text() ').getStringVal() =”NotMichelle”;



Conclusion: We have successfully created an XML database.