e-Journal

on

ADVANCED DATABASE SYSTEMS

SUBMITTED BY

**KALLIL RAHUL RAVIDNRAN**

ROLL NO: 05

Submitted in partial fulfillment of the requirement for

Qualifying

M.Sc. Part I Semester II Examination

2018-19

Department of Information Technology

Ramniranjan Jhunjhunwala College

Station Road, Ghatkopar (w), Mumbai-86



**CERTIFICATE**

This is to certify that Mr. KALLIL RAHUL RAVINDRAN with Seat No. 05 has successfully completed the necessary course of experiments in the subject of **ADVANCED DATABASE SYSTEMS** during the academic year **2018 – 2019** complying with the requirements of **RAMNIRANJAN JHUNJHUNWALA COLLEGE OF ARTS, SCIENCE AND COMMERCE**, for the course of **M.Sc. (IT)** semester -II.

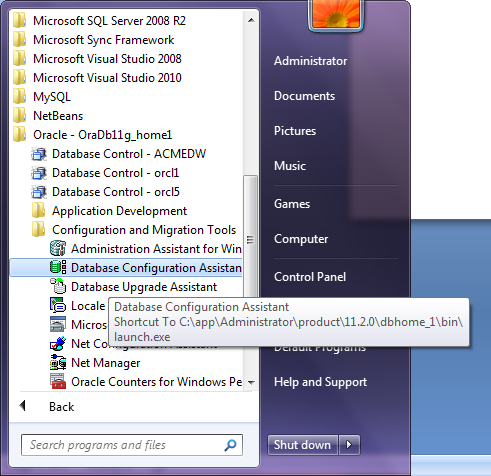
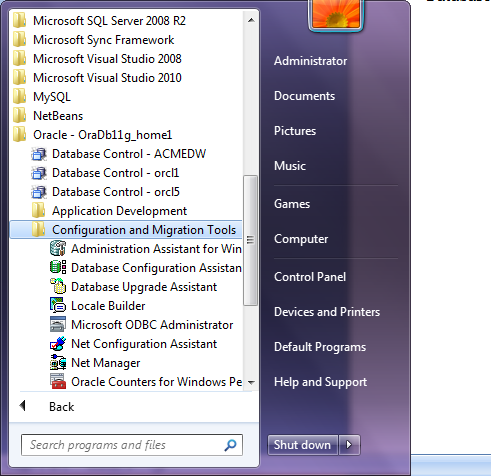
Internal Examiner Date:

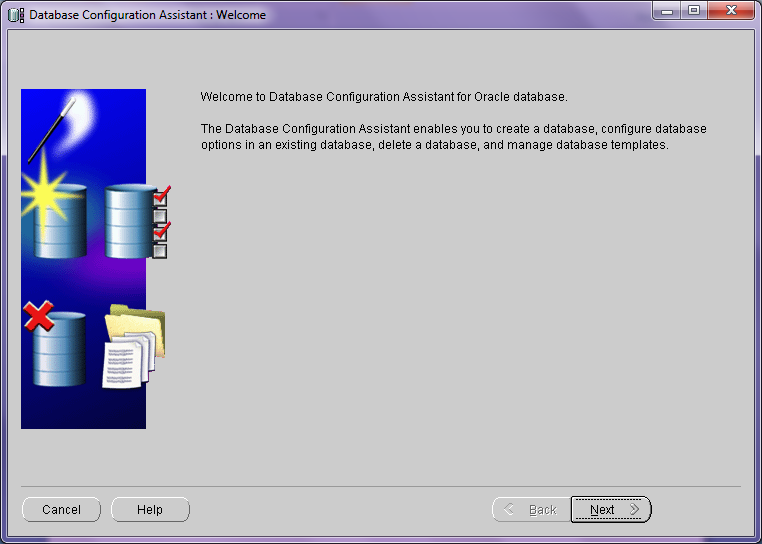
Head of Department College Seal External Examiner

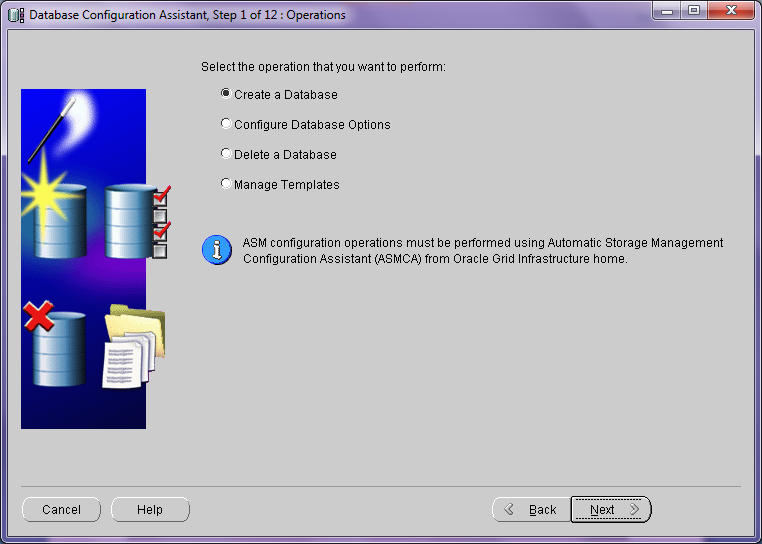
**CONTENTS**

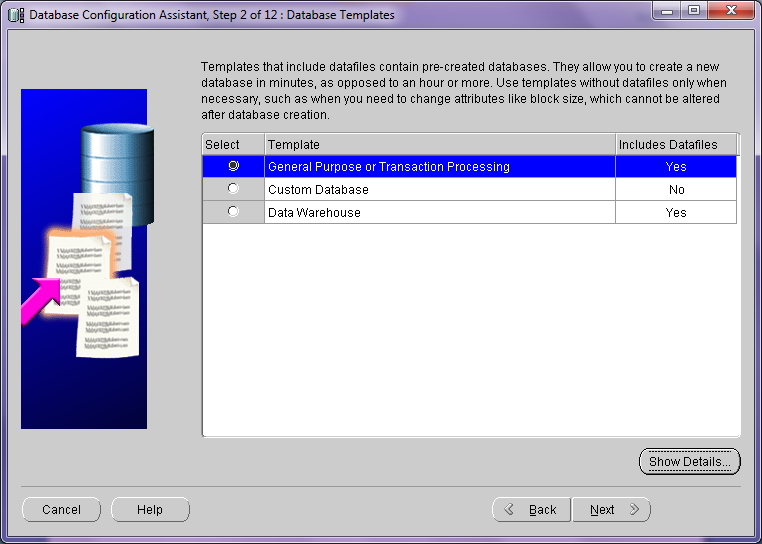
|  |  |  |
| --- | --- | --- |
| **PRACTICAL NUMBER** | **PRACTICAL TITLE** | **PAGE NUMBER** |
| 1 | Horizontal fragmentation of database | 12 |
| 2 | Vertical fragmentation of database | 15 |
| 3 | Creating Replica of database | 19 |
| 4 | Create ORDBMS Application | 24 |
| 5 | XML Application | 31 |
| 6 | Implement Active database using Triggers | 39 |
| 7 | Create Temporal Database | 45 |
| 8 | Implement and retrieve records from a Spatial Database | 48 |
| 9 | Implement Prolog Programming | 60 |
| 10 | Implement XML Parsing & Using XML DOM Traverse XML Document | 66 |
| 11 | Inserting and Retrieving Multimedia Objects in Database (Image/audio/video) | 69 |

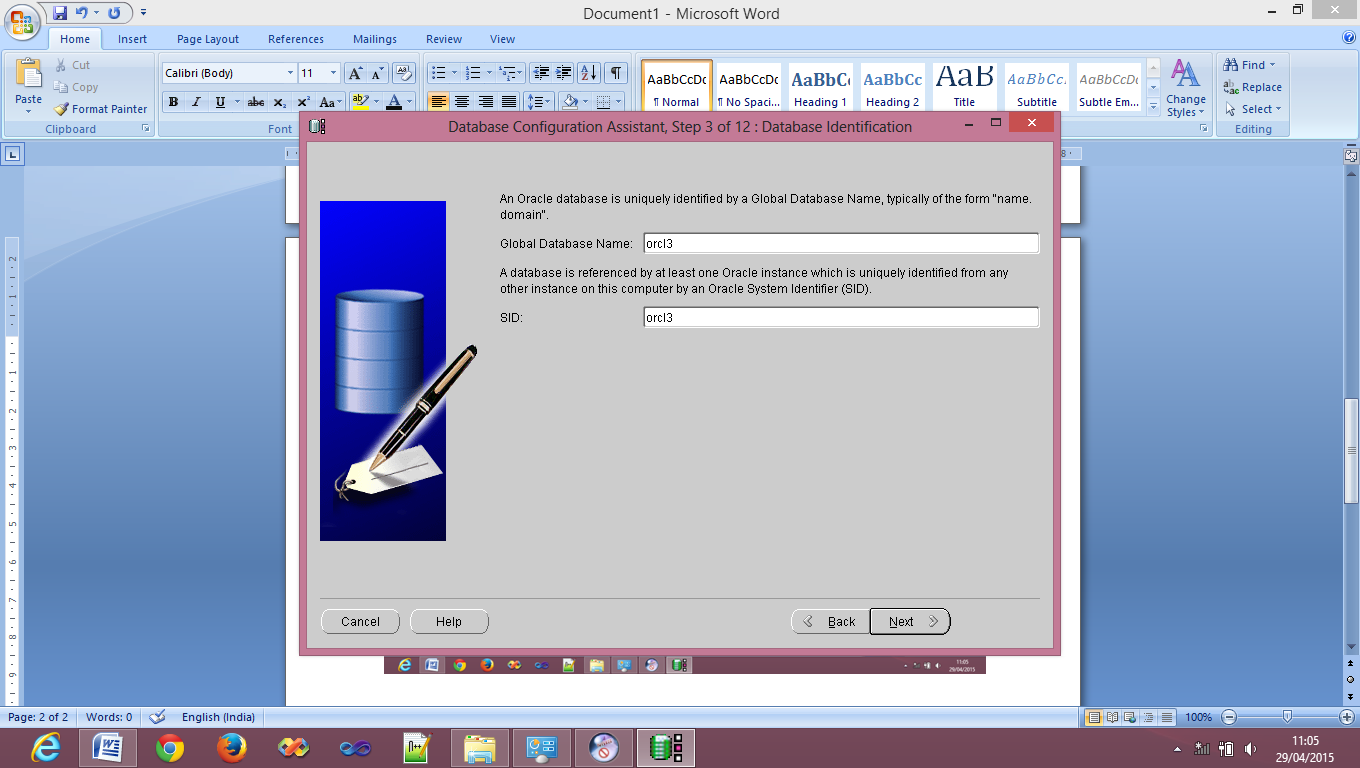
**Database Creation Steps:**

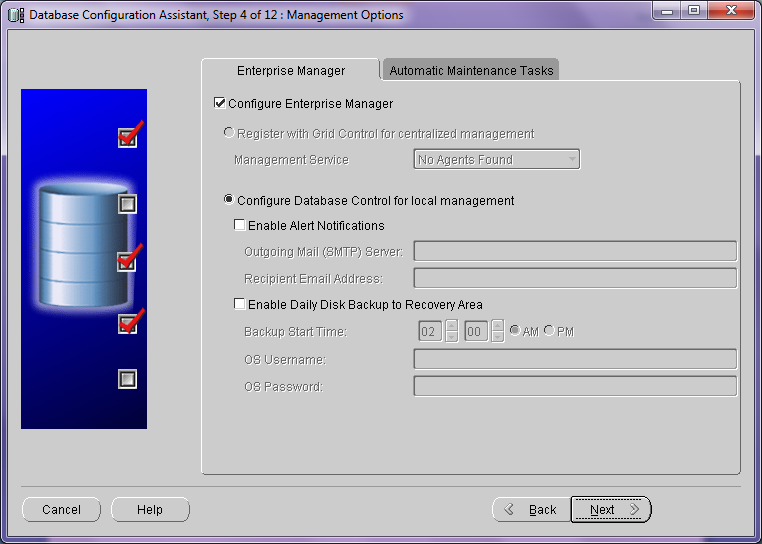
-

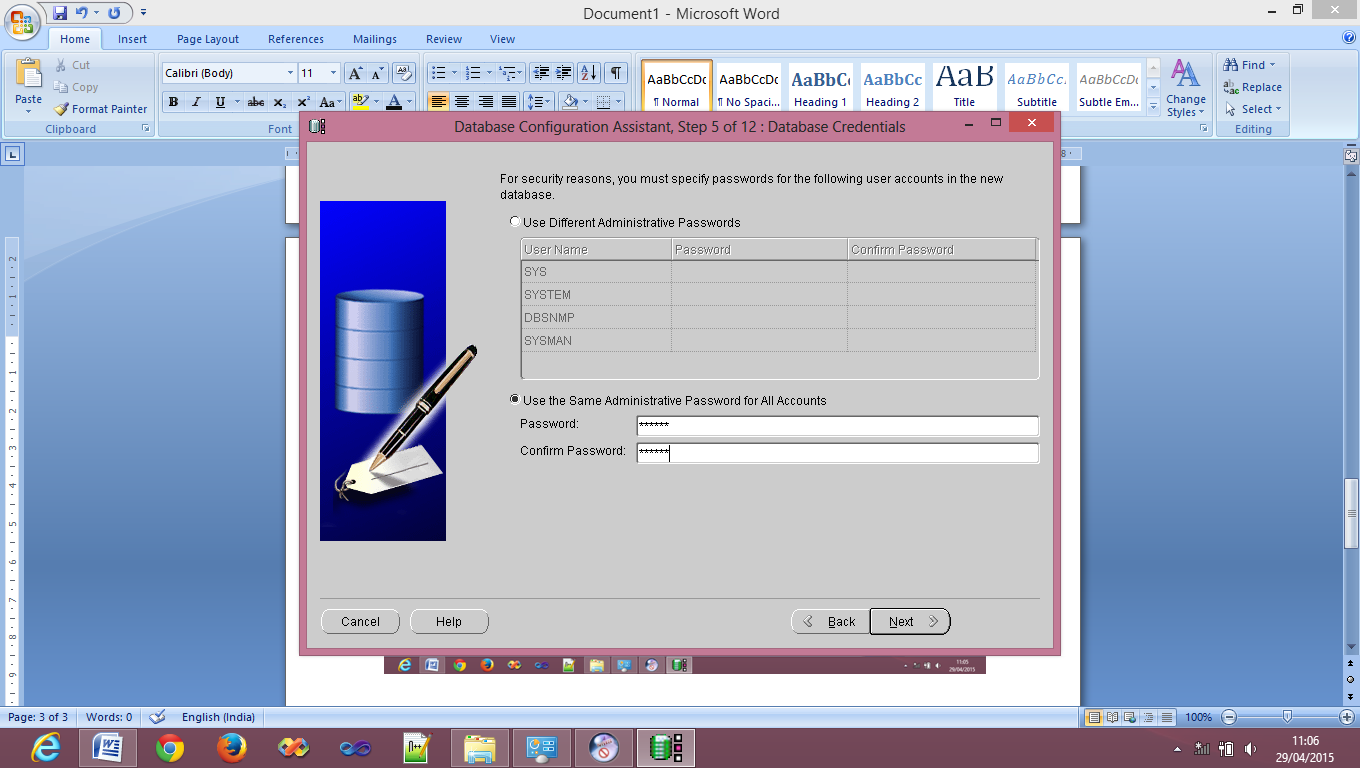




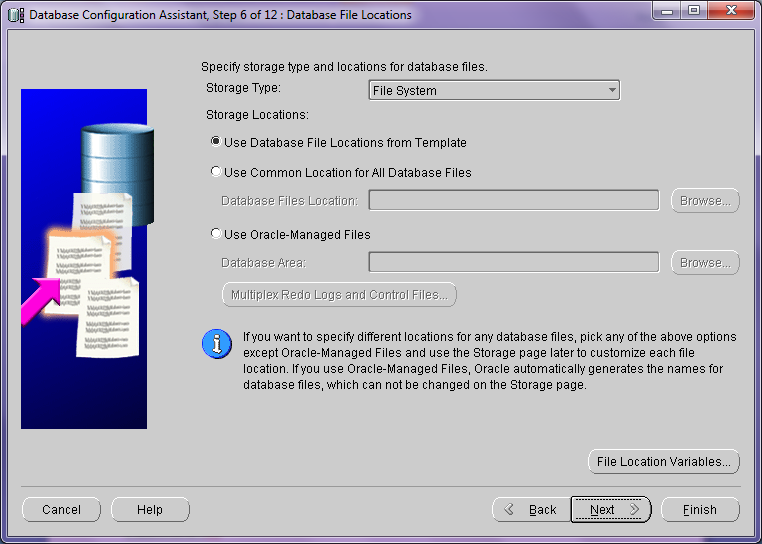


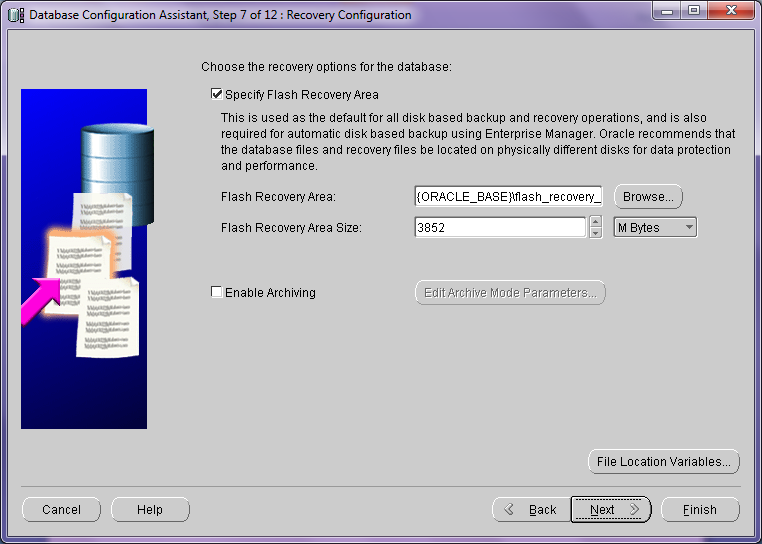
****

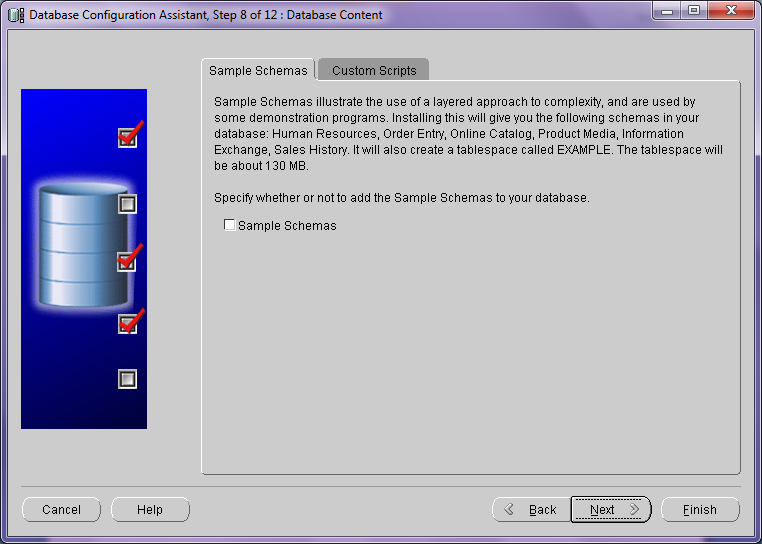


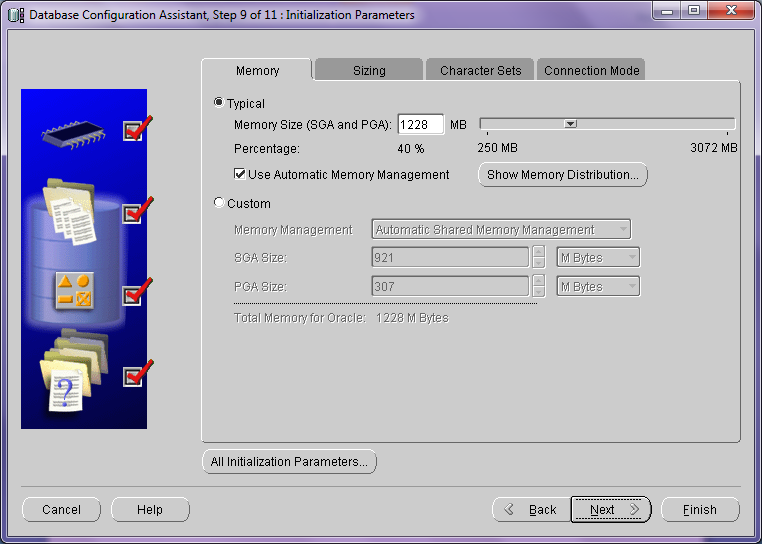
****

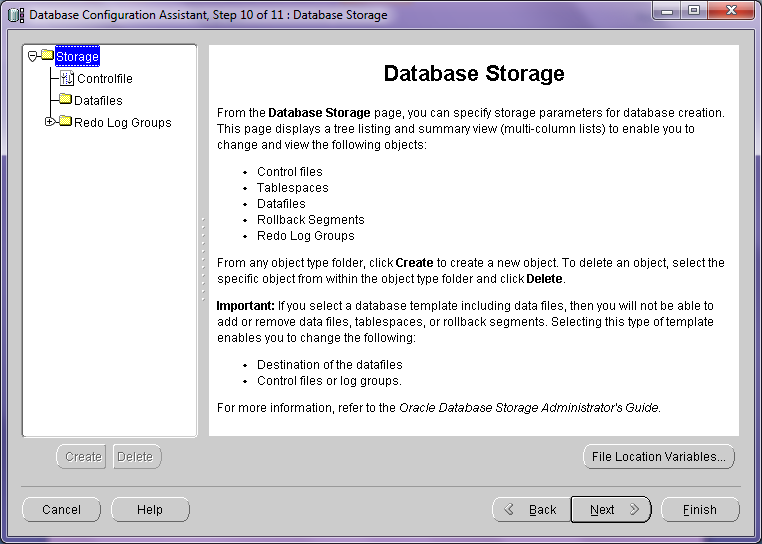
**Password:**

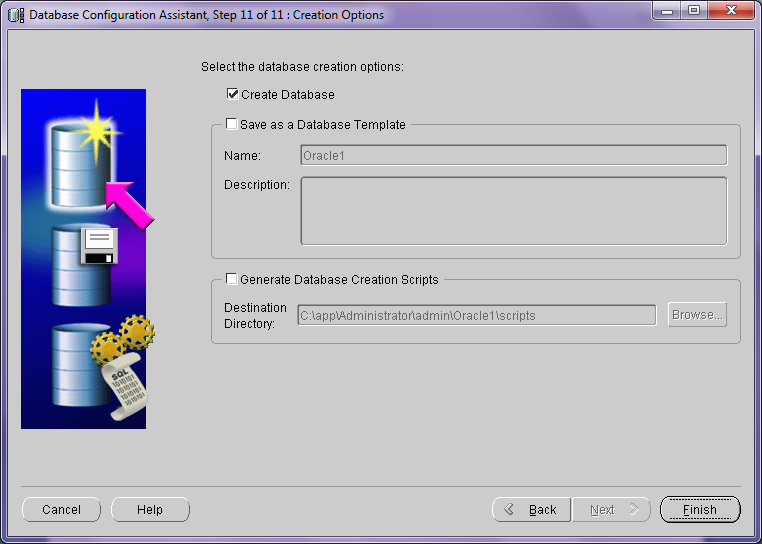




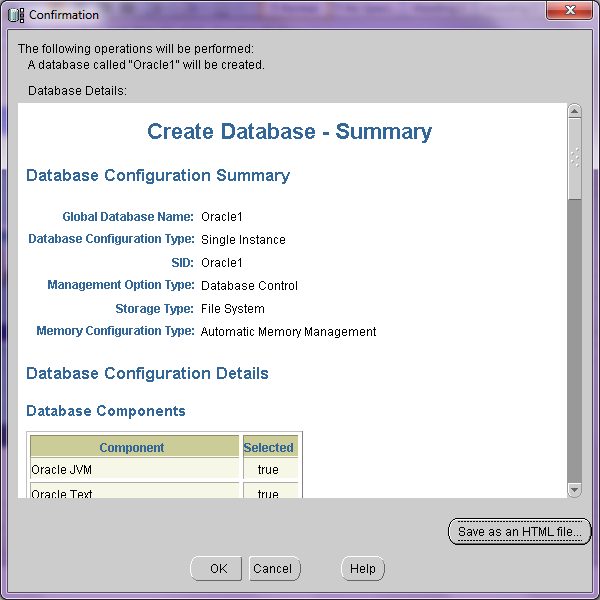




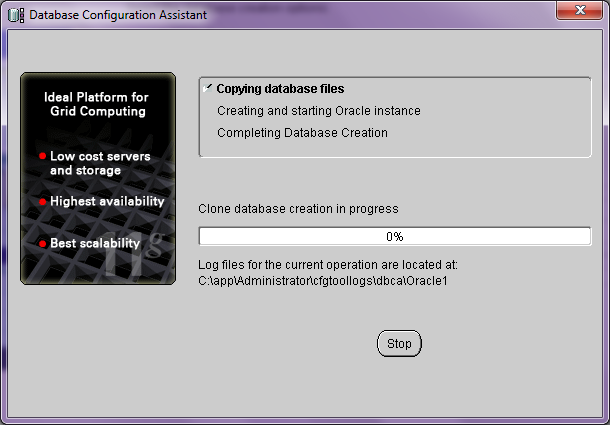


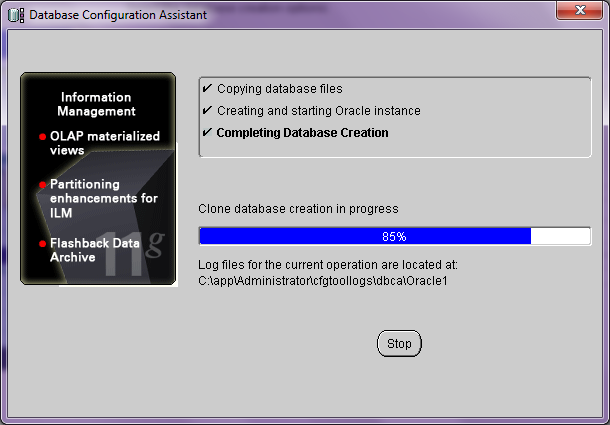


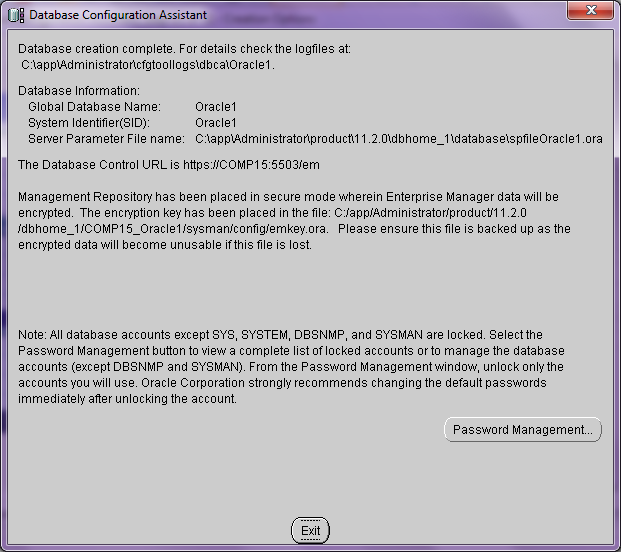
Click Finish



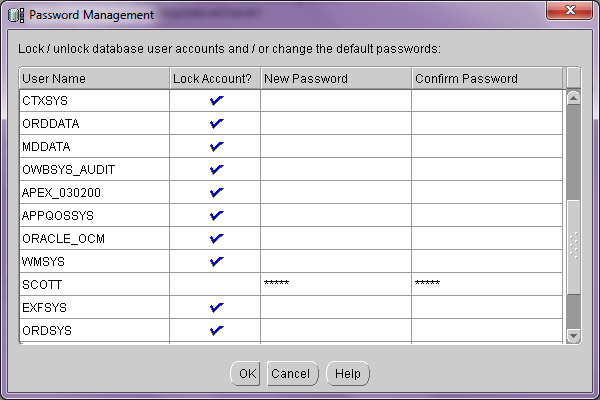
Click OK then you wil get above screen

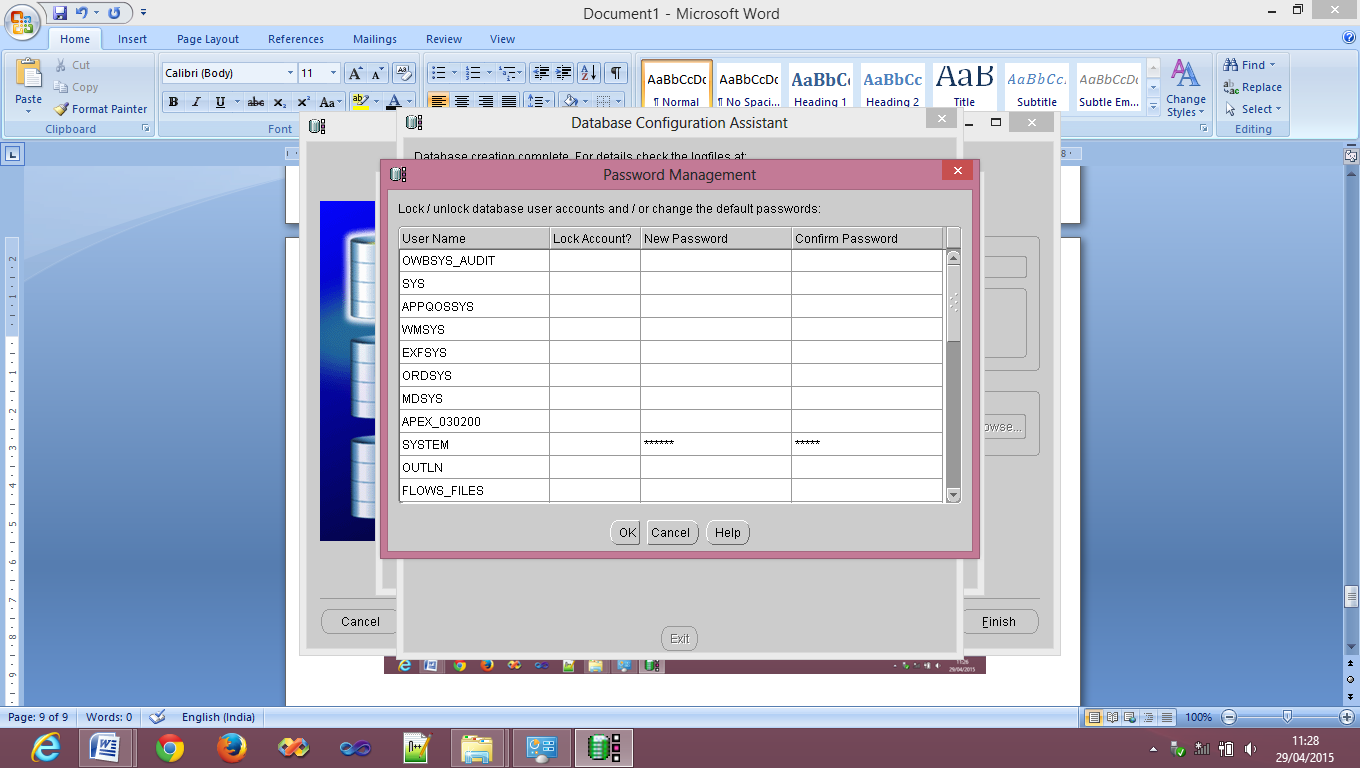






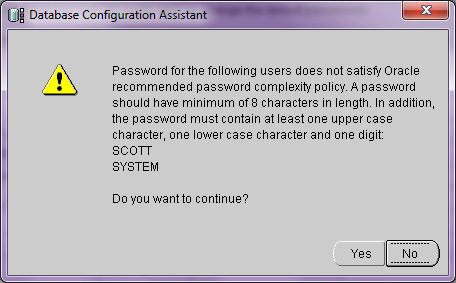
Click on password management



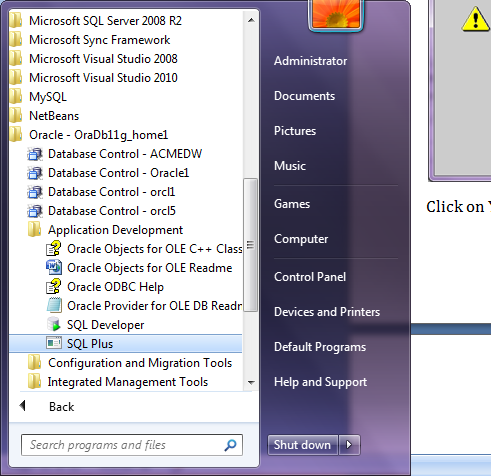
****

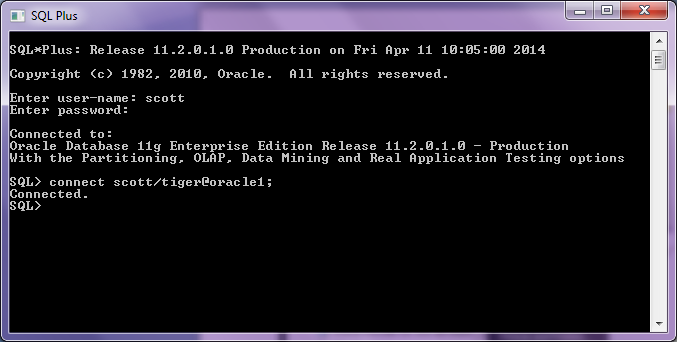
Uncheck everything as shown above write password t=at System

Click Ok



Click on Yes and then Exit





**Practical No.1**

**Horizontal fragmentation of database.**

Question: Create a global conceptual schema Emp (Eno, Ename, Address, Email, Salary) and insert 10 records. Divide Emp into horizontal fragments using the condition that Emp contains tuples with salary < 5000 and Emp with 5000 < salary < 20000 on two different nodes. Fire the following queries:

a) Find the salary of all employees.

b) Find the Email of all employees where salary=4000.

c) Find the employee name and Email where employee number is known.

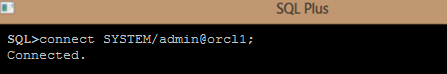
d) Find the employee name and address where employee number is known.

**Similarly create database orcl2**

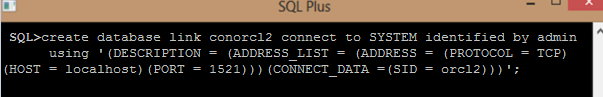
Open sqlplus command prompt and login as SYSTEM (USERNAME ) and

admin (PASSWORD)

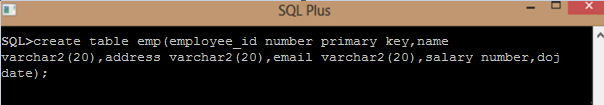
**Open ORCL1 Database:**



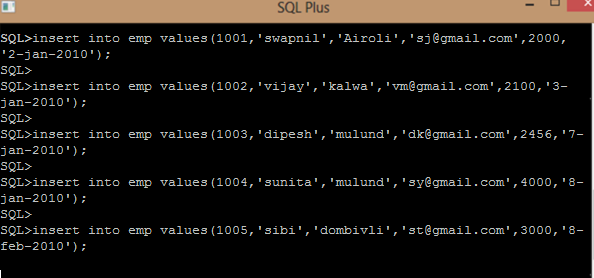
**Creating Link In ORCL1:**



**Creating Table emp:**



**Inserting Values Into Table emp:**

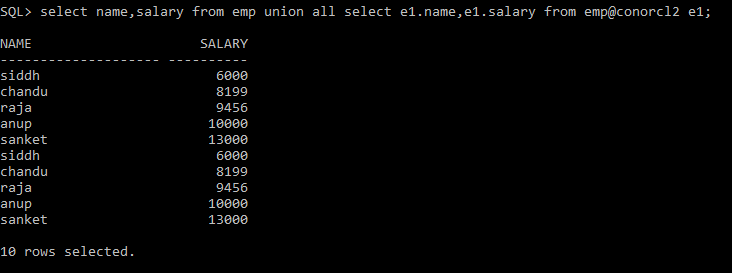


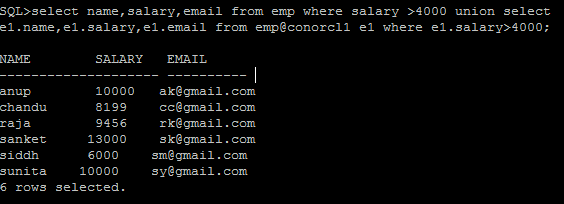
**Creating view hrz\_view:**



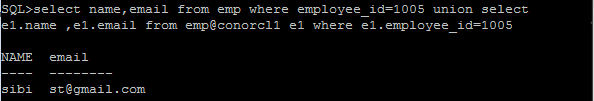
**Output Queries:**

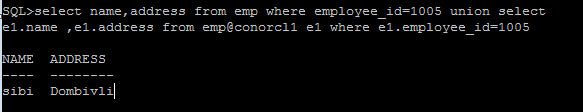
**1) Find the salary of all employees.**



**2) Find the Email of all employees where salary=4000.**

**3 ) Find the employee name and Email where employee number is known**





**Practical No.2**

**Aim: Vertical fragmentation of database.**

Question: Create a global conceptual schema Emp ( Employee\_id, first\_name,last\_name,email,address1,address2,address3,doj,Salary,designation) and insert 10 records. Divide Emp into vertical fragments Emp(Employee\_id,first\_name,last\_name,email) and emp(Employee\_id ,address1,address2,address3),Emp(Employee\_id,doj,Salary ,designation) on two different nodes.

Fire the following queries:

a) Find the salary of an Employee where employee number is known.

b) Find the Email where the employee name is known.

c) Find the employee name and Email where employee number is known.

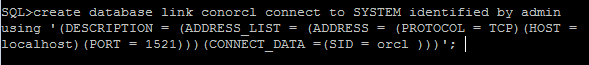
d) Find the employee name whose salary is > 200.

**Solution : create databases orcl,orcl1,orcl2 as shown above**

**Open ORCL Database:**



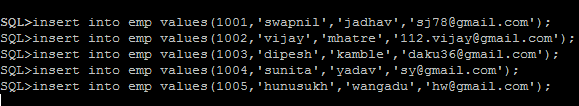
**Creating link in ORCL:**



**Creating Table emp:**



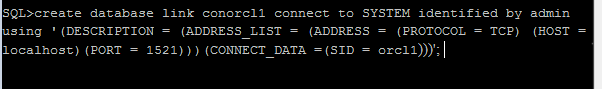
**Inserting Values Into Table emp:**



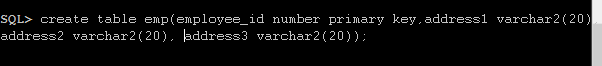
**Open ORCL1 Database:**



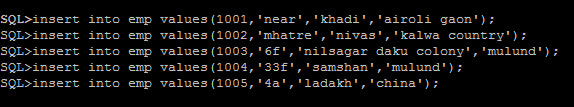
**Creating Link in ORCL1:**



**Creating Table emp:**



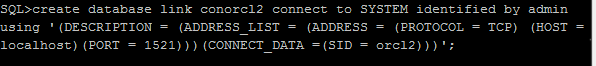
**Inserting Values Into Table emp:**



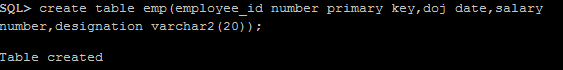
**Open ORCL2 Database:**



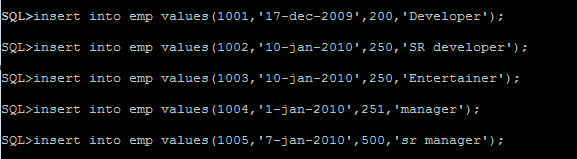
**Creating Link In ORCL2:**



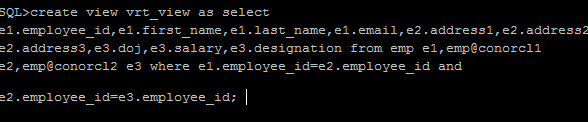
**Creating Table emp:**



**Inserting Values Into Table emp:**

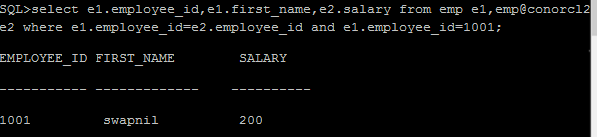


**Creating view vrt\_view:**

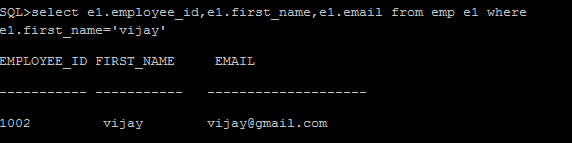


**Output Queries:**

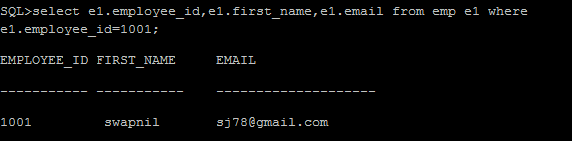
**1) Find the salary of an Employee where employee number is known.**



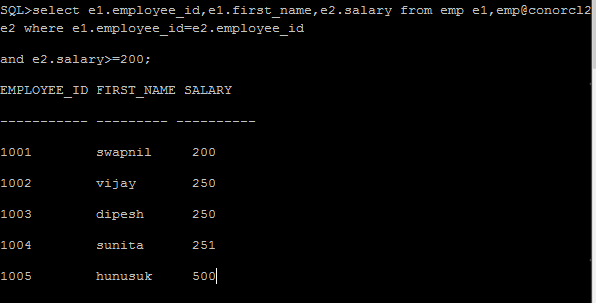
**2) Find the Email where the employee name is known.**



**3) Find the employee name and Email where employee number is known.**



**4) Find the employee name whose salary is > 200.**



**Practical No.3**

**Creating Replica of database.**

Question: Create global conceptual schema Emp (Eno ,Ename, Address, Email, Salary) and insert 10 records. Store the replication of Emp into two different nodes and

Fire The Following Queries.

a) Find the salary of all employees.

b) Find the email of all employees where salary = 15000.

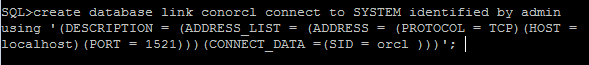
c) Find the employee name and email where employee number is known.

d) Find the employee name and address where employee number is known.

**Open ORCL Database:**



**Creating link in ORCL:**



**Creating Table emp:**

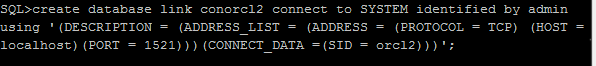
create table emp (eno number, ename varchar2(20), address varchar(20), email varchar2(20), salary number);

**Open ORCL2 Database:**

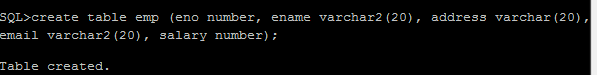
**Open ORCL2 Database:**



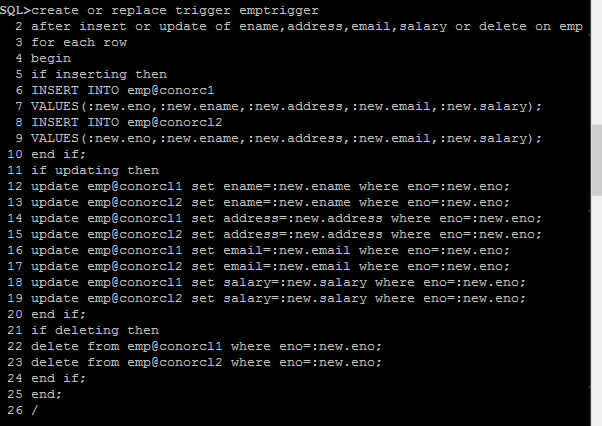
**Creating Link In ORCL2:**



**Creating Table emp:**

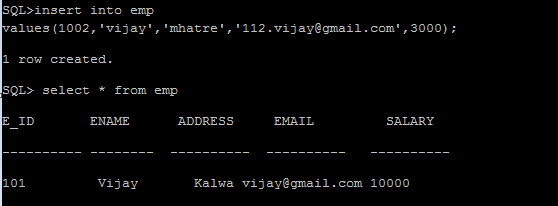


**Creating Trigger emptrigger:**

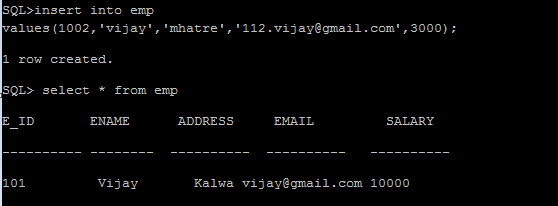


**Output Queries:**

**A) Inserting values into employee table:**

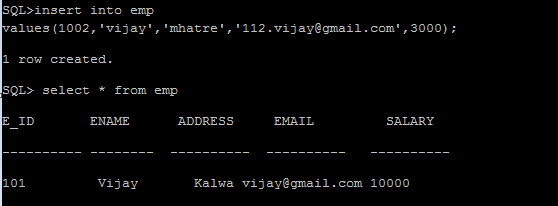


**Connect to orcl1 database:**

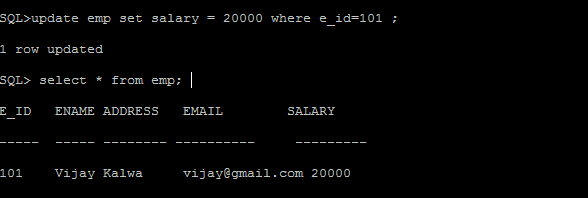


**Connect to orcl2 database:**



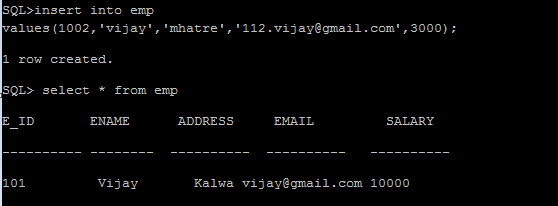


**B) Updating the values into employee table:**



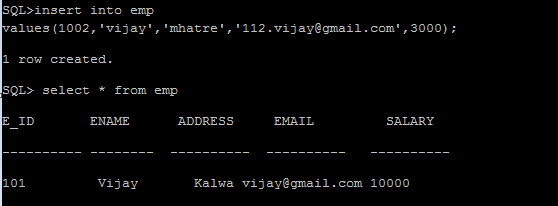
**Connect to orcl1 database:**



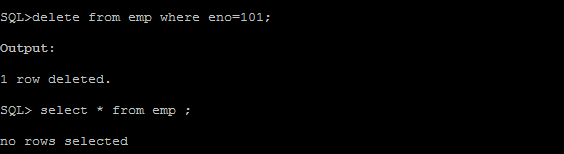


**Connect to orcl2 database:**





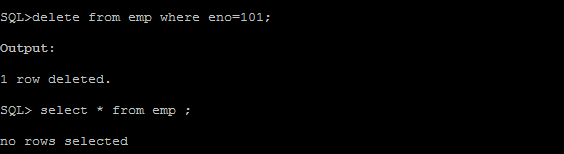
**C) Deleting the values into employee table:**



**Connect to orcl1 database:**



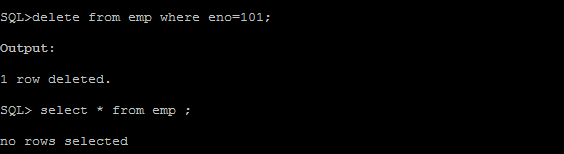
**Fire the query:**



**Connect to orcl2 database:**



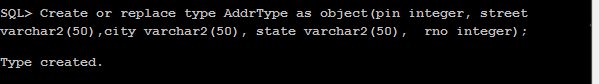
**Fire the query:**



**Practical No. 4**

**Implement ORDBMS Application.**

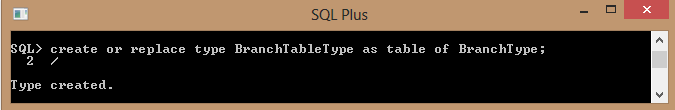
Create or replace type AddrType as object(pin integer, street varchar2(50), city varchar2(50), state varchar2(50), rno integer);



create or replace type BranchType as object(address AddrType1,phone1 integer,phone2 integer);



create or replace type BranchTableType as table of BranchType;



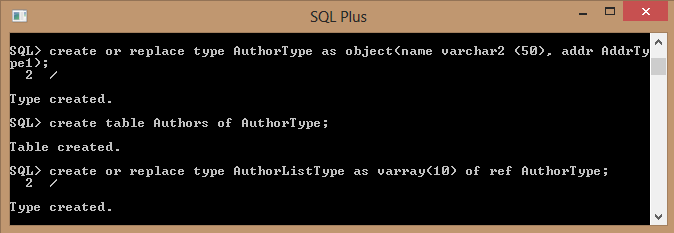
create or replace type AuthorType as object(name varchar2 (50), addr AddrType1);

/

create table Authors of AuthorType;

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

/

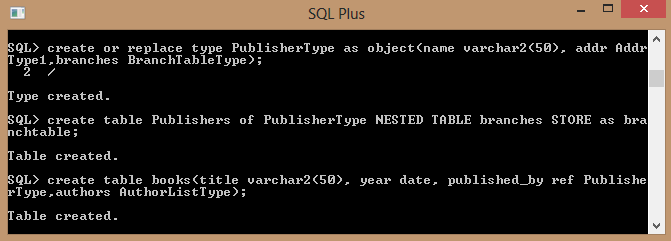


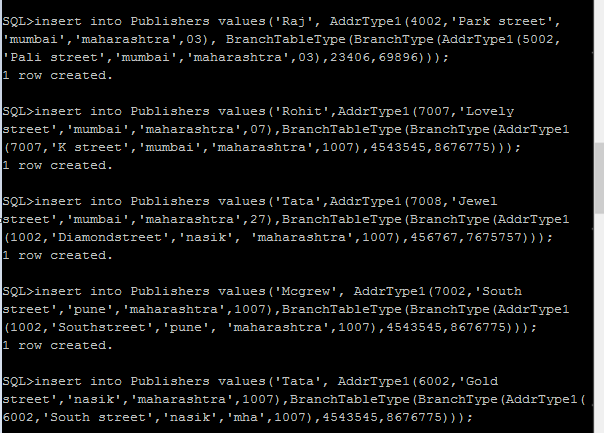
create or replace type PublisherType as object(name varchar2(50), addr AddrType1,branches BranchTableType);

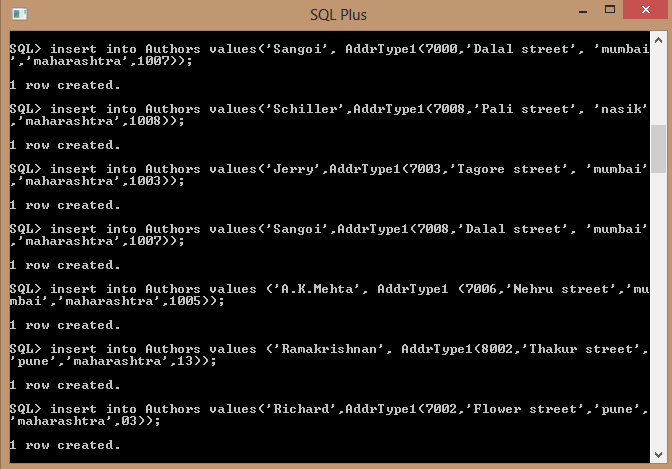
/

create table Publishers of PublisherType NESTED TABLE branches STORE as branchtable;

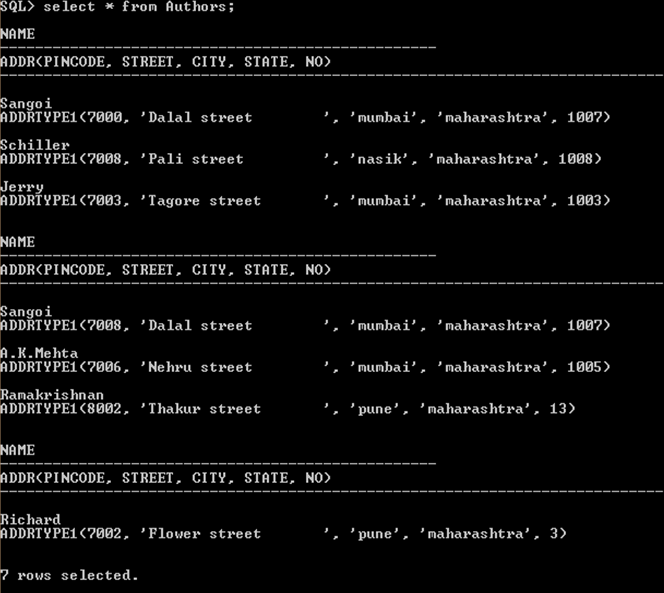
create table books(title varchar2(50), year date, published\_by ref PublisherType,authorsAuthorListType);

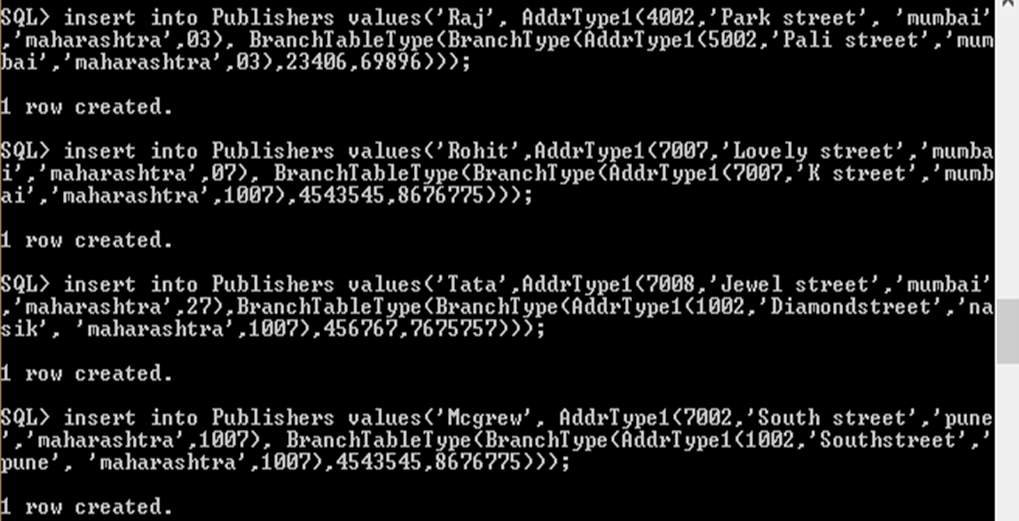




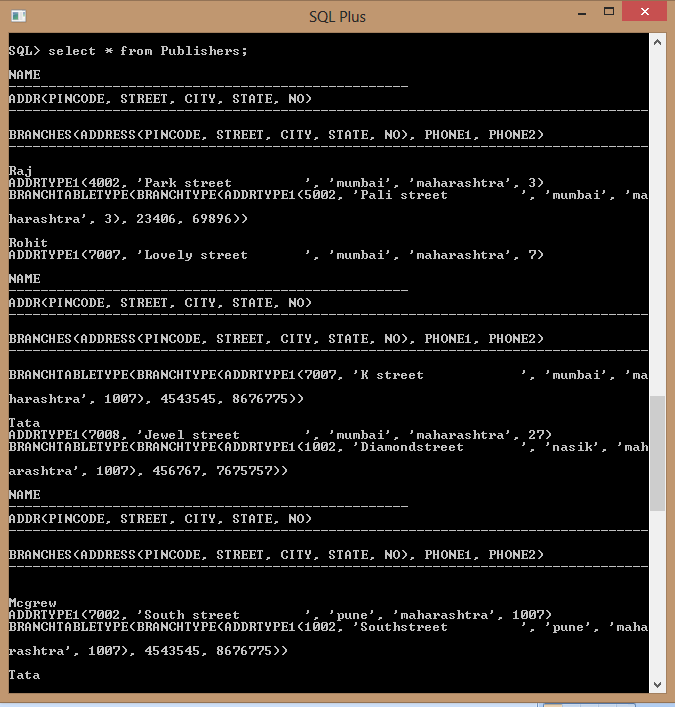


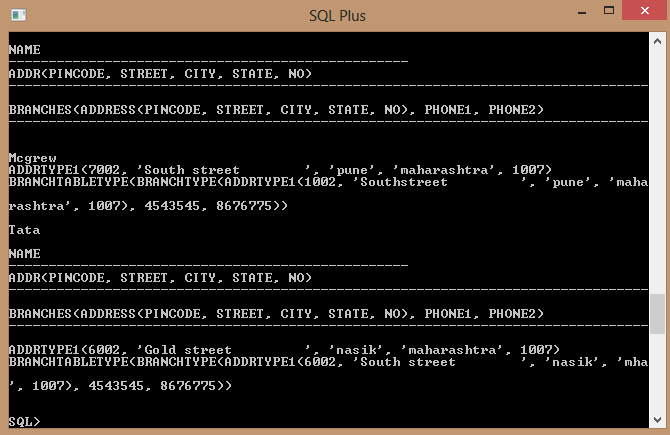
select \* from Authors;





select \* from Publishers;



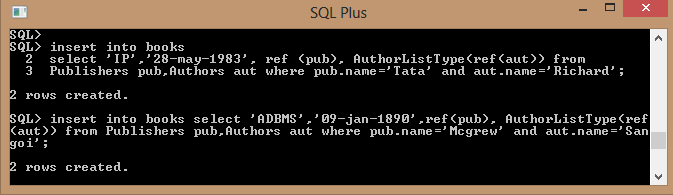


insert into books

select 'IP','28-may-1983', ref (pub), AuthorListType(ref(aut)) from

Publishers pub,Authorsaut where pub.name='Tata' and aut.name='Richard';

insert into books select 'ADBMS','09-jan-1890',ref(pub), AuthorListType(ref(aut)) from Publishers pub,Authorsaut where pub.name='Mcgrew' and aut.name='Sangoi';

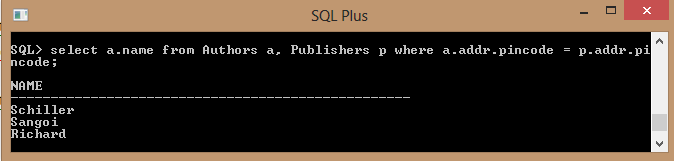


insert into books

select 'c prog','25-may-1983', ref (pub),AuthorListType(ref(aut)) from

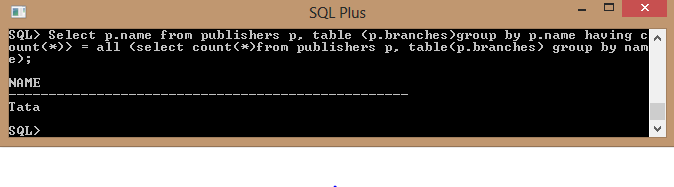
Publishers pub,Authorsaut where pub.name='Raj' and aut.name='Ramkrishnan.';

select a.name from Authors a, Publishers p where a.addr.pincode = p.addr.pincode;



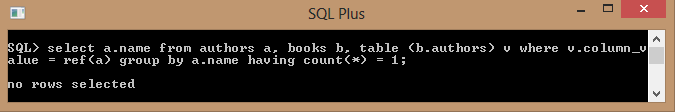
List the name of the publisher that has the most branches:

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



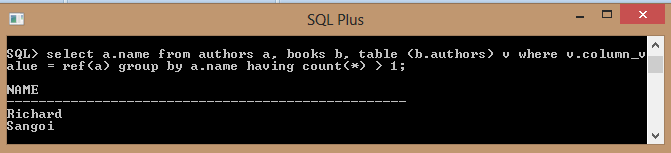
List all the authors who have published more than one book & Name of authors who have published books with atleast two different publishers

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;



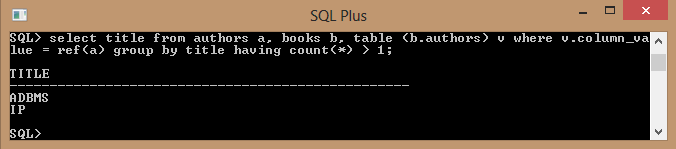
List all the authors who have published more than one book & Name of authors who have published books with atleast two different publishers:

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;



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 isunique in a list of authors has not been specified):

select title from authors a, books b, table (b.authors) vwherev.column\_value = ref(a) group by title having count(\*) > 1;



**Practical No. 5**

**Aim: Implement XML Application**

A]Create an XML Application using XML as database and ASP.

B] Create a table employee having dept\_id as number datatype and employee\_spec as XML datatype(XM\_Type). The employee\_spec is a schema with attributes emp\_id, name, email, acc\_no, managerEmail, dataOfJoning. Insert 10 tuples into employee table.

Fire the following queries on XML database.

a) Retrieve the names of employee.

b) Retrieve the acc\_no of employees.

c) Retrieve the names, acc\_no, email of employees.

d) Update the 3rd record from the table and display the name of an employee.

e) Delete 4th record from the table.

**A] Create an XML Application using XML as database and ASP.**

**VerifyPerson.asp:**

<%@ Language=JScript%>

<%Server.ScriptTimeout=21478836%>

<%Response.Buffer=false%>

// 1) To check whether the user has already pressed the submit button

<%

var submit = Request.Form("submit").Count;

if( submit > 0 ){

// 2) Create the needed to variables to store data from the form

var name = Request.Form("Name");

var age = Request.Form("Age");

var gender = Request.Form("Gender");

varpcode = Request.Form("PostalCode");

var city = Request.Form("City");

// 3) First check whether the user has entered anything or not !

var error = "";

if ( name == "" )

error = "Name ";

if ( age == "" )

error += "Age ";

if ( pcode == "")

error += "PostalCode ";

if ( city == "")

error += "City ";

// 4) We have found that the user didnt entered anything

if(error!=""){

Response.Write("<center>");

Response.Write("<font fac=verdana size=2>");

Response.Write("Please enter the following data:<br>");

Response.Write("<b>");

Response.Write(error);

Response.Write("</b>");

Response.Write("</font>");

Response.Write("</center>");

}

//Response.Write(name+" has been added to your Friend list.");

// 5) Load the xmlDoc and create the required elements/nodes

varxmlDoc=Server.CreateObject("MICROSOFT.FreeThreadedXMLDOM");

xmlDoc.async="false";

xmlDoc.load(Server.MapPath("Person.xml"));

varnodeList = xmlDoc.getElementsByTagName("PersonList");

if(nodeList.length> 0){

varparentNode = nodeList(0) ;

varpersonNode = xmlDoc.createElement("Person");

varnameNode = xmlDoc.createElement("Name");

varageNode = xmlDoc.createElement("Age");

vargenderNode = xmlDoc.createElement("Gender");

varpcodeNode = xmlDoc.createElement("PostalCode");

varcityNode = xmlDoc.createElement("City");

nameNode.text = name;

ageNode.text = age;

genderNode.text= gender;

pcodeNode.text = pcode;

cityNode.text = city;

// 6) Attach the nodes to the parent node (Person)

parentNode.appendChild(personNode);

personNode.appendChild(nameNode);

personNode.appendChild(ageNode);

personNode.appendChild(genderNode);

personNode.appendChild(pcodeNode);

personNode.appendChild(cityNode);

// 7) Now save the nodes to the file

xmlDoc.save(Server.MapPath("Person.xml"));

}

}

}

%>

<html><body>

<TD width="69"><FONT face=Verdana size=2><STRONG style="BACKGROUND-COLOR: silver">Age</STRONG></FONT></TD>

<TD width="112"><FONT face=Verdana size=2><STRONG style="BACKGROUND-COLOR: silver">Gender</STRONG></FONT></TD>

<TD width="115"><FONT face=Verdana size=2><STRONG style="BACKGROUND-COLOR: silver">Postal Code</STRONG></FONT></TD>

<TD width="115"><FONT face=Verdana size=2><STRONG style="BACKGROUND-COLOR: silver">City</STRONG></FONT></TD>

</TR>

<%

// This part is used to display the data in a table via XSL

varobjXMLDoc = Server.CreateObject("MICROSOFT.FreeThreadedXMLDOM");

objXMLDoc.async = false;

objXMLDoc.load(Server.MapPath("person.xml"));

varxsl=Server.CreateObject("MICROSOFT.FreeThreadedXMLDOM");

xsl.async = false;

xsl.load(Server.MapPath("person.xsl"));

varxmlQuery="//Person";

vardocHeadlines=objXMLDoc.documentElement.selectNodes(xmlQuery);

varnumNodes;

numNodes=docHeadlines.length;

varnn;

for(vari=0;i<numNodes;i++){

nn = docHeadlines.nextNode();

Response.Write(nn.transformNode(xsl));

}

%>

</table>

</body>

</html>

**Person.xml :**

<?xml version="1.0" encoding="ISO-8859-1"?>

<PersonList>

<Person><Name>Sonu</Name><Age>25</Age><Gender>Male</Gender><PostalCode>99999</PostalCode><City>Thane</City></Person>

<Person><Name>vijay</Name><Age>23</Age><Gender>Male</Gender><PostalCode>121</PostalCode><City>mumbai</City></Person></PersonList>

Person.xsl :

<xsl:stylesheetxmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="Person">

<tr>

<td><font face="verdana" size="2"><xsl:value-of select="Name"/></font></td>

<td><font face="verdana" size="2"><xsl:value-of select="Age"/></font></td>

<td><font face="verdana" size="2"><xsl:value-of select="Gender"/></font></td>

<td><font face="verdana" size="2"><xsl:value-of select="PostalCode"/></font></td>

<td><font face="verdana" size="2"><xsl:value-of select="City"/></font></td>

</tr>

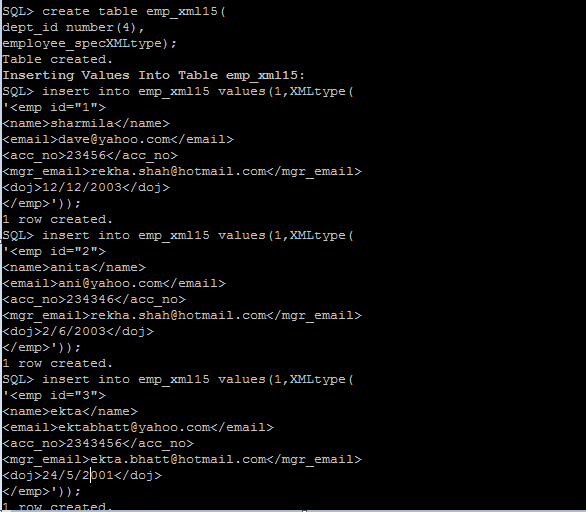
</xsl:template>

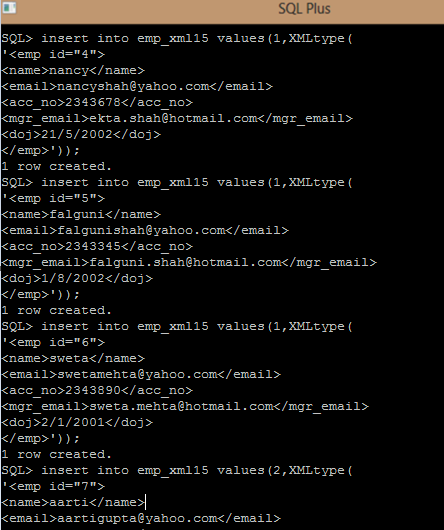
</xsl:stylesheet>

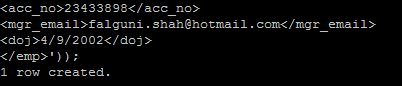
****

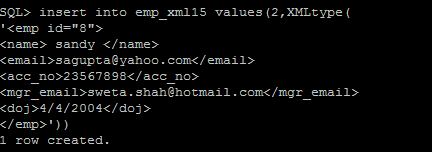
**B] XML Database**

**Creating Table emp\_xml15**



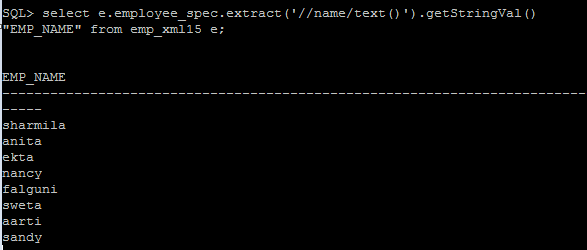




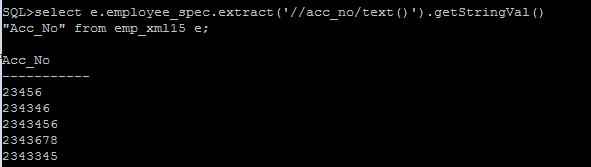


**Queries:**

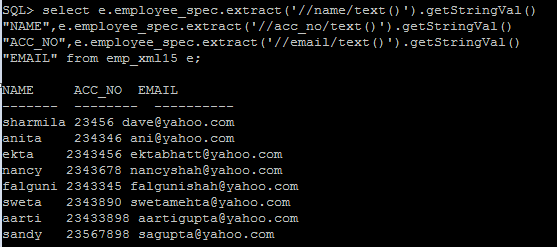
**a) Retrieve the names of employee:**



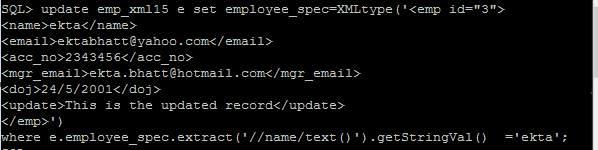
**b) Retrieve the acc\_no of employees:**

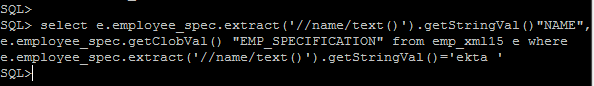


**c) Retrieve the names, acc\_no, email of employees:**



**d) Update the 3rd record from the table and display the name of an employee:**





**Output:**

**NAME**

**-------------------------------------------------------------------------------**

**EMP\_SPECIFICATION**

**-------------------------------------------------------------------------------**

**ekta**

**<emp id="3">**

**<name>ekta</name>**

**<email>ektabhatt@yahoo.com</email>**

**<acc\_no>2343456</acc\_no>**

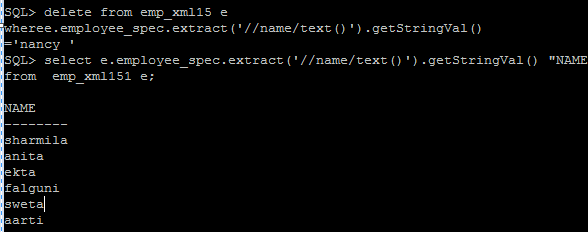
**<mgr\_email>ekta.bhatt@hotmail.com</mgr\_email>**

**<doj>24/5/2001</doj>**

**<update>This is the updated record</update>**

**</emp>**

**e) Delete the 4th record from the table:**



**Practical No.6**

**Implement Active database using Triggers.**

Question: Create table emptab (eno, ename, hrs, pno, super\_no) and project (pname, pno, thrs, head\_no) where thrs is the total hours and is the derived attribute. Its value is the sum of all employees working on that project. eno and pno are primary keys, head\_no is foreign key to emp relation.

Insert 10 tuples and write triggers to do the following.

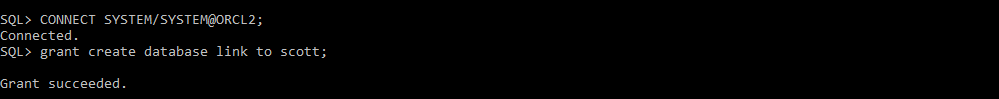
a) Creating a trigger to insert new employee tuple and display the new total hours from project table.

b) Creating a trigger to change the hrs of existing employee and display the new total hours from project table.

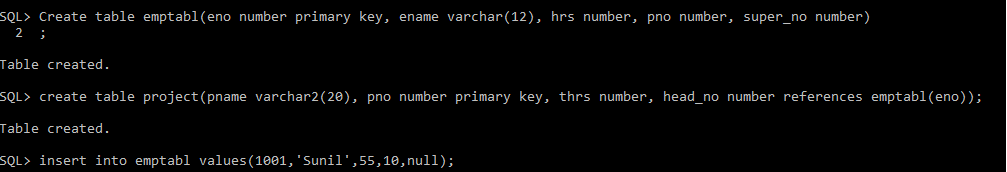
c) Cr eating a trigger to change the project of an employee and display the new total hours from project table.

d) Creating a trigger to delete the project of an employee.

Create table emptabl(eno number primary key, enamevarchar(12), hrs number, pno number, super\_no number)



create table project(pname varchar2(20), pno number primary key, thrs number, head\_no number references emptabl(eno));



insert into emptabl values(1001,'Sunil',55,10,null);

insert into emptabl values(1002,'vijaya',155,20,1001);

insert into emptabl values(1003,'dipti',175,40,1001);

insert into emptabl values(1004,'sonu',455,30,null);

insert into emptabl values(1005,'anupam',155,10,1003);

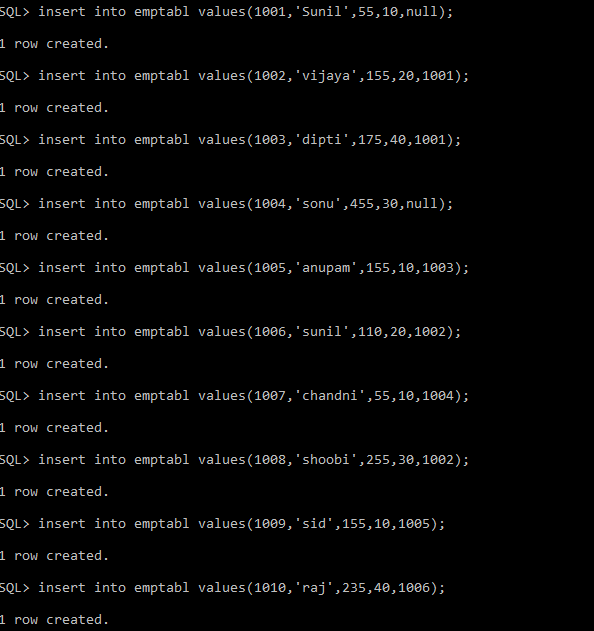
insert into emptabl values(1006,'sunil',110,20,1002);

insert into emptabl values(1007,'chandni',55,10,1004);

insert into emptabl values(1008,'shoobi',255,30,1002);

insert into emptabl values(1009,'sid',155,10,1005);

insert into emptabl values(1010,'raj',235,40,1006);

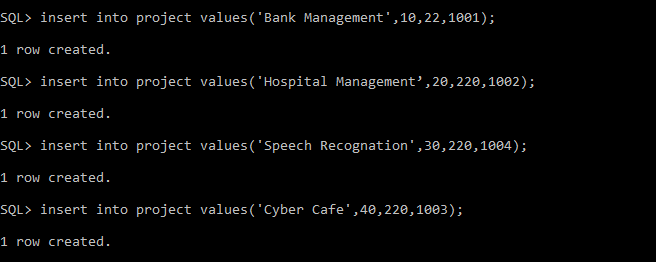


insert into project values('Bank Management',10,22,1001);

insert into project values('Hospital Management’,20,220,1002);

insert into project values('Speech Recognation',30,220,1004);

insert into project values('Cyber Cafe',40,220,1003);

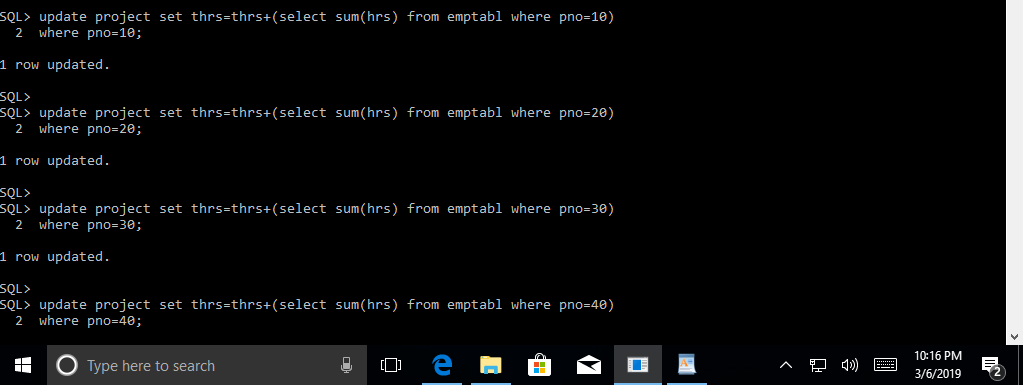


update project set thrs=thrs+(select sum(hrs) from emptabl where pno=10) wherepno=10;

update project set thrs=thrs+(select sum(hrs) from emptabl where pno=20) wherepno=20;

update project set thrs=thrs+(select sum(hrs) from emptabl where pno=30)wherepno=30;

update project set thrs=thrs+(select sum(hrs) from emptabl where pno=40) wherepno=40;



create or replace trigger emptrigg

after insert on emptabl

for each row

when(New.pno!=0)

begin

update project

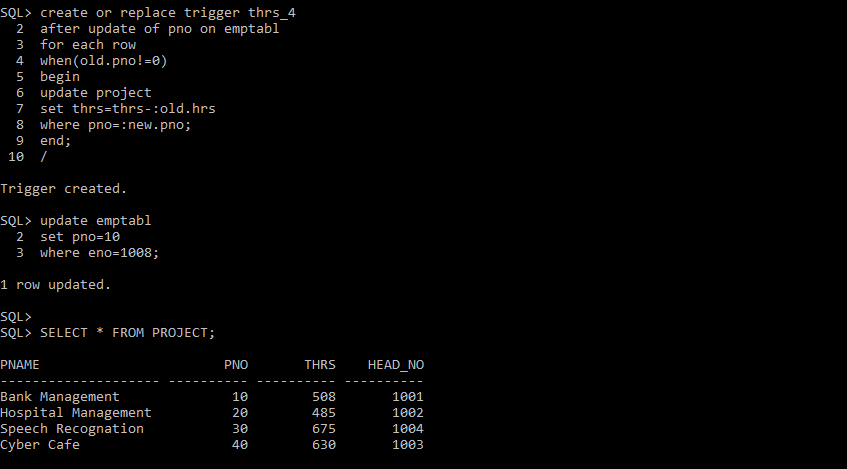
setthrs=thrs+:new.hrs

wherepno=:new.pno;

end;

/

insert into emptabl values(1011,'sameer',21,10,1001);



create or replace trigger emptrigg1

after update on emptabl

for each row

when(New.pno!=0)

begin

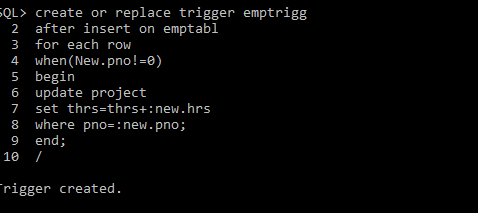
update project

setthrs=thrs-:old.hrs+:new.hrs

wherepno=:new.pno;

end;

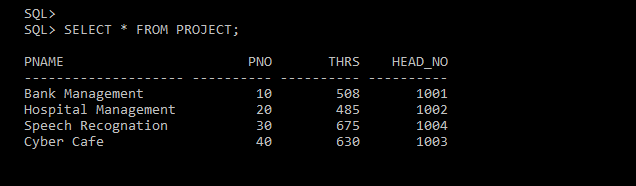
/



UPDATE EMPTABl

SET HRS=100

WHERE ENO=1001;



create or replace trigger emptrigg2

after update of pno on emptabl

for each row

when(New.pno!=0)

begin

update project

setthrs=thrs+:new.hrs

wherepno=:new.pno;

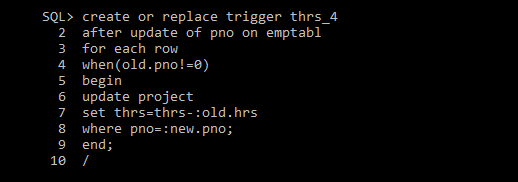
update project

setthrs=thrs-:old.hrs

wherepno=:old.pno;

end;

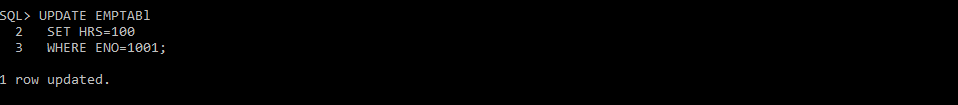
/

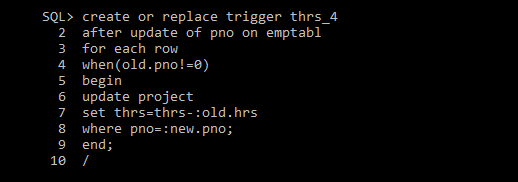


updateemptabl

setpno=10

whereeno=1001;





create or replace trigger thrs\_4

after update of pno on emptabl

for each row

when(old.pno!=0)

begin

update project

setthrs=thrs-:old.hrs

wherepno=:new.pno;

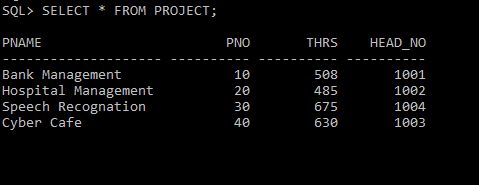
end;

/

updateemptabl

setpno=10

whereeno=1008;



**Practical No.7**

**Aim: Create Temporal Database.**

A] Create a table tbl\_shares, which stores the, name of company, number of shares, and price per share at transaction time. Insert 10 records and fire the following queries.

1) Find all the names of a company whose share price is more than Rs.100 at 11:45 A.M.

2)Find the name of company which has highest shares price at 5.00 P.M.

B] Create a table employees, which stores the, employee\_id, name, department, salary at transaction time using Time DB .insert 3 records and fire the following queries.

1) Find all the details where employee\_id=10;

A)

Open ORCL1 Database:



Create table tbl\_shares15

(

cname varchar2(20),

nofshares number(5),

pricepshare number(5),

transtime timestamp(6)

)

insert into tbl\_shares15 values(‘Cap Gemini’,250,25,’17-dec-94 11.55.00.000000 am’);

insert into tbl\_shares15 values('Tata',205,20,'05-jun-04 11.45.00.000000 am');

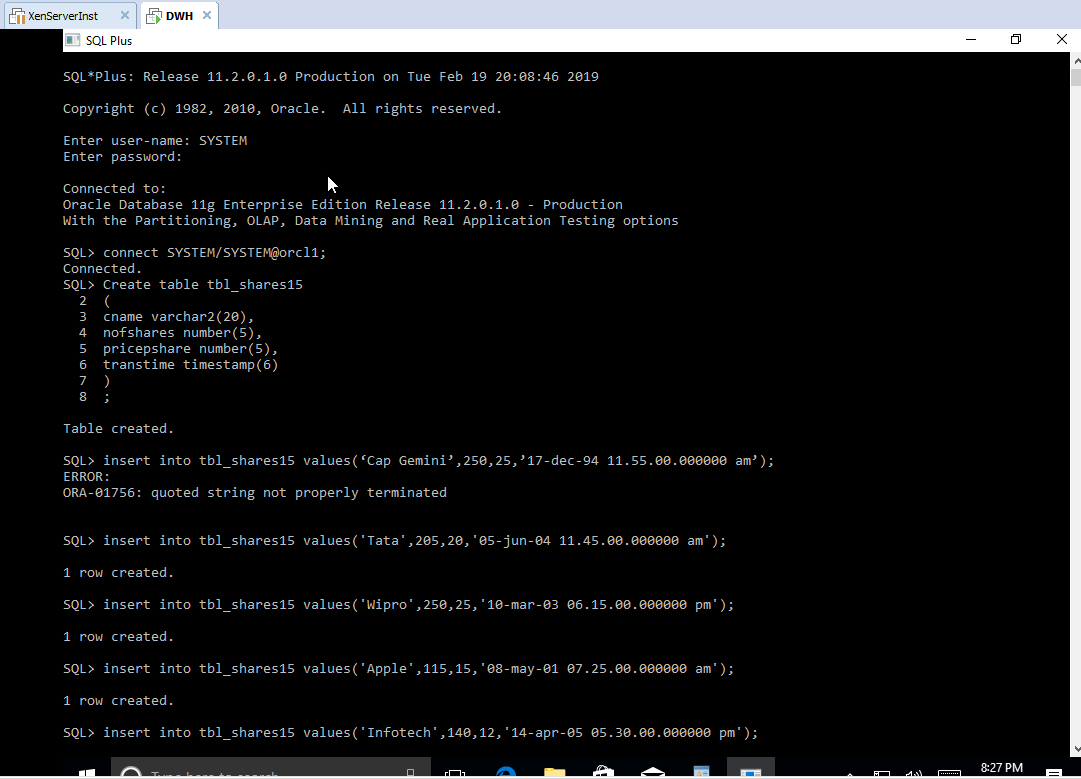
insert into tbl\_shares15 values('Wipro',250,25,'10-mar-03 06.15.00.000000 pm');

insert into tbl\_shares15 values('Apple',115,15,'08-may-01 07.25.00.000000 am');

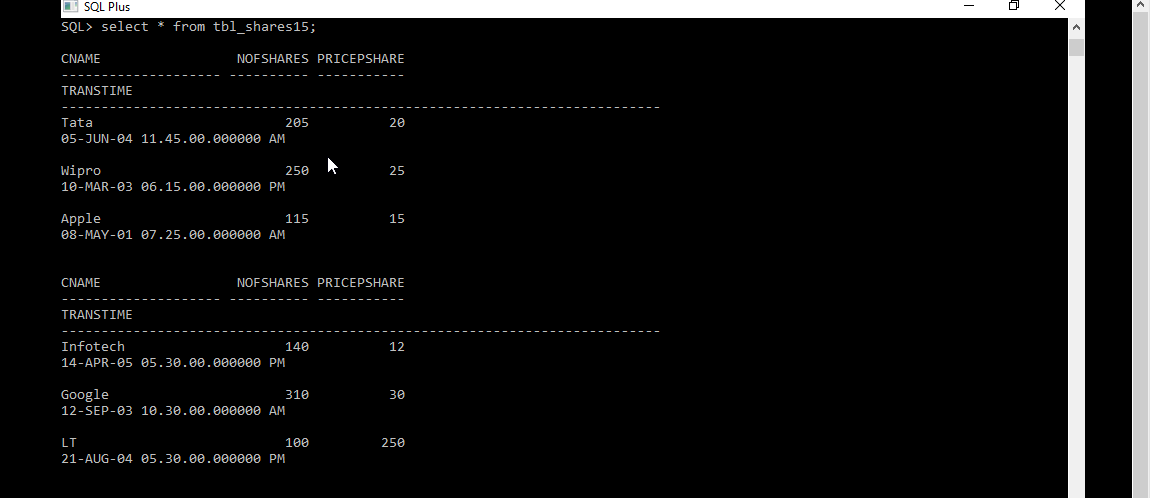
insert into tbl\_shares15 values('Infotech',140,12,'14-apr-05 05.30.00.000000 pm');

insert into tbl\_shares15 values('Google',310,30,'12-sep-03 10.30.00.000000 am');

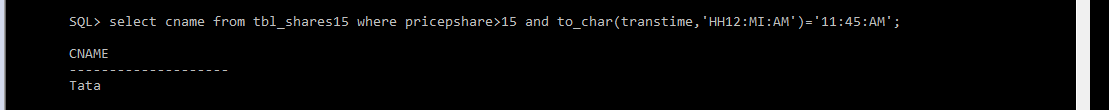
insert into tbl\_shares15 values('LT',100,250,'21-aug-04 05.30.00.000000 pm')



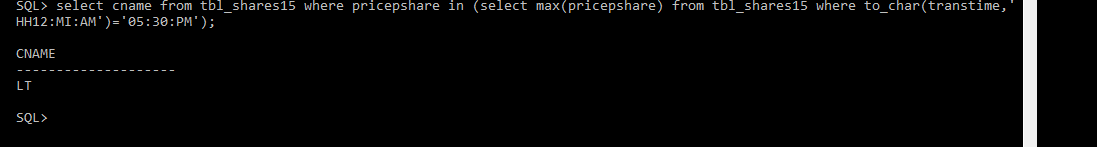
select \* from tbl\_shares15;



selectcname from tbl\_shares15 where pricepshare>15 and to\_char(transtime,'HH12:MI:AM')='11:45:AM';



selectcname from tbl\_shares15 where pricepshare in (select max(pricepshare) from tbl\_shares15 where to\_char(transtime,'HH12:MI:AM')='05:30:PM');



**Practical No.8**

**Implement and retrieve records from a Spatial Database.**

**A] Spatial Database**

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

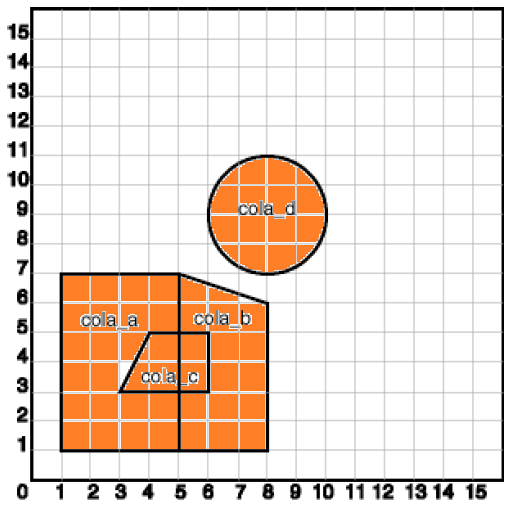
1. Find the topological intersection of two geometries.

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

3. Find the areas of all direction locations.

4. Find the area of only one location.

5. Find the distance between two geometries.



**B] Spatial Database**

Create a spatial database that stores the number, name and location, which consists of different areas within the university campus including the main gate, the playground, the arts and science college buildings, the lad and print facility building. Fire the following queries.

1. Display area for each object.

2. Find out the distance of the main gate from all other objects

3. Find the intersection area of lab and print facility

4. Distance between arts and science building

5. Find the spatial relationship between canteen and print facility

6. Find the distance between college buildings and canteen.

|  |
| --- |
|  |
|  |

A)

CREATE TABLE cola\_markets1 (

mkt\_id NUMBER PRIMARY KEY,

name VARCHAR2(32),

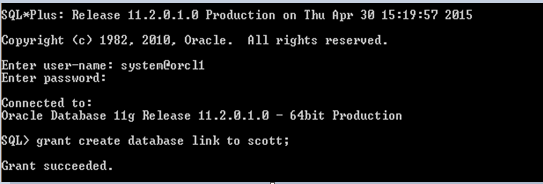
shape MDSYS.SDO\_GEOMETRY);

INSERT INTO cola\_markets1 VALUES(1,'abc',MDSYS.SDO\_GEOMETRY(2003,NULL,NULL,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),MDSYS.SDO\_ORDINATE\_ARRAY(1,1, 5,7) ));

INSERT INTO cola\_markets1 VALUES(2,'pqr',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)));

INSERT INTO cola\_markets1 VALUES(3,'mno',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)));

INSERT INTO cola\_markets1 VALUES(4,'xyz',MDSYS.SDO\_GEOMETRY(2003, NULL,NULL,MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,4),MDSYS.SDO\_ORDINATE\_ARRAY(8,7, 10,9, 8,11)));



INSERT INTO USER\_SDO\_GEOM\_METADATA

VALUES ('cola\_markets1','shape',

MDSYS.SDO\_DIM\_ARRAY(

MDSYS.SDO\_DIM\_ELEMENT('X', 0, 20, 0.005),

MDSYS.SDO\_DIM\_ELEMENT('Y', 0, 20, 0.005)

),NULL

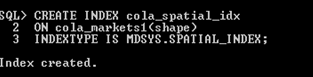
);



CREATE INDEX cola\_spatial\_idx

ON cola\_markets1(shape)

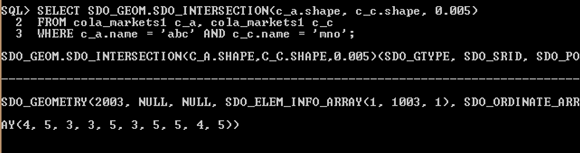
INDEXTYPE IS MDSYS.SPATIAL\_INDEX;



SELECT SDO\_GEOM.SDO\_INTERSECTION(c\_a.shape, c\_c.shape, 0.005)

FROM cola\_markets1 c\_a, cola\_markets1 c\_c

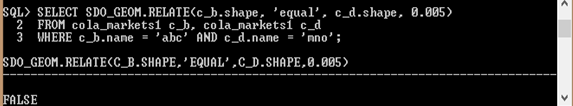
WHERE c\_a.name = 'abc' AND c\_c.name = 'mno';



SELECT SDO\_GEOM.RELATE(c\_b.shape, 'equal', c\_d.shape, 0.005)

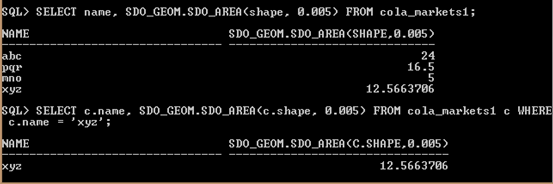
FROM cola\_markets1 c\_b, cola\_markets1 c\_d

WHERE c\_b.name = 'abc' AND c\_d.name = 'mno';



SELECT name, SDO\_GEOM.SDO\_AREA(shape, 0.005) FROM cola\_markets1;

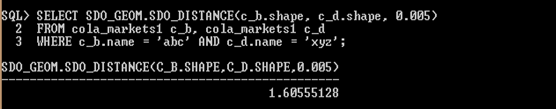
SELECT c.name, SDO\_GEOM.SDO\_AREA(c.shape, 0.005) FROM cola\_markets1 c WHERE c.name = 'xyz';



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM cola\_markets1 c\_b, cola\_markets1 c\_d

WHERE c\_b.name = 'abc' AND c\_d.name = 'xyz';



B)

CREATE TABLE university\_camp (

mkt\_id NUMBER PRIMARY KEY,

name VARCHAR2(32),

shape MDSYS.SDO\_GEOMETRY);

INSERT INTO university\_campVALUES(

1,'a',

MDSYS.SDO\_GEOMETRY(

2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(1,5, 2,8)

)

);

INSERT INTO university\_campVALUES(

4,'d',

MDSYS.SDO\_GEOMETRY(

2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(8,6,10,9)

)

);

INSERT INTO university\_campVALUES(

5,'e',

MDSYS.SDO\_GEOMETRY(

2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(8,4,10,7)

)

);

INSERT INTO university\_campVALUES(

6,'f',

MDSYS.SDO\_GEOMETRY(

2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,3),

MDSYS.SDO\_ORDINATE\_ARRAY(9,1,11,3)

)

);

INSERT INTO university\_campVALUES(

3,'c',

MDSYS.SDO\_GEOMETRY(

2003,NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,1),

MDSYS.SDO\_ORDINATE\_ARRAY(8,10,11,10,11,13,6,13,8,11,8,10)

)

);

INSERT INTO university\_campVALUES(

7,'g',

MDSYS.SDO\_GEOMETRY(

2003, NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,1),

MDSYS.SDO\_ORDINATE\_ARRAY(4,2,7,2,8,3,7,4,4,4,4,2)

)

);

INSERT INTO university\_campVALUES(

2,'b',

MDSYS.SDO\_GEOMETRY(

2003, NULL,NULL,

MDSYS.SDO\_ELEM\_INFO\_ARRAY(1,1003,4),

MDSYS.SDO\_ORDINATE\_ARRAY(3,7, 5,5,5,9)

)

);

INSERT INTO USER\_SDO\_GEOM\_METADATA

VALUES ('university\_camp','shape',

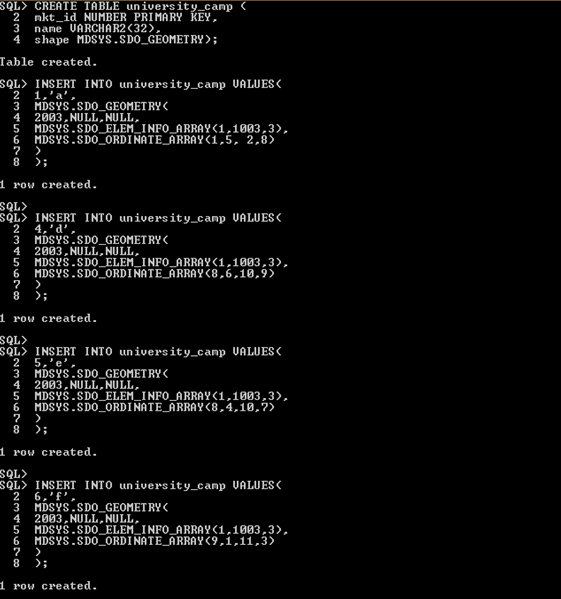
MDSYS.SDO\_DIM\_ARRAY(

MDSYS.SDO\_DIM\_ELEMENT('X', 0, 20, 0.005),

MDSYS.SDO\_DIM\_ELEMENT('Y', 0, 20, 0.005)

),NULL

);

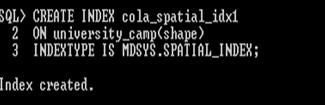




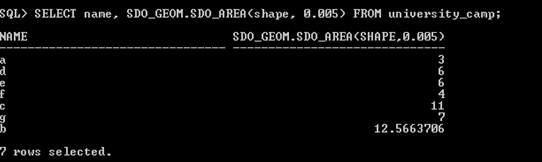
CREATE INDEX cola\_spatial\_idx1

ON university\_camp(shape)

INDEXTYPE IS MDSYS.SPATIAL\_INDEX;



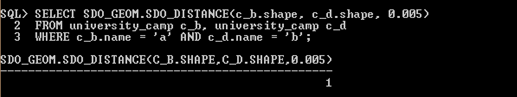
SELECT name, SDO\_GEOM.SDO\_AREA(shape, 0.005) FROM university\_camp;



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

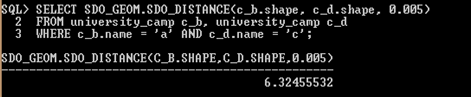
WHERE c\_b.name = 'a' AND c\_d.name = 'b';



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

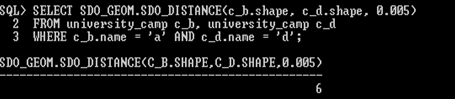
WHERE c\_b.name = 'a' AND c\_d.name = 'c';



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

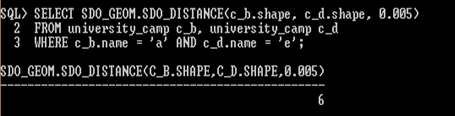
WHERE c\_b.name = 'a' AND c\_d.name = 'd';



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

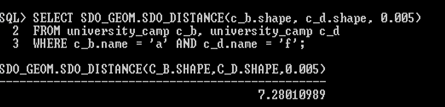
WHERE c\_b.name = 'a' AND c\_d.name = 'e';



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

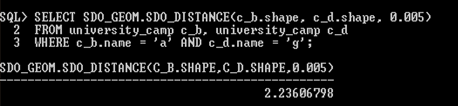
WHERE c\_b.name = 'a' AND c\_d.name = 'f';



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

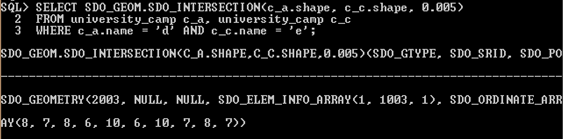
WHERE c\_b.name = 'a' AND c\_d.name = 'g';



SELECT SDO\_GEOM.SDO\_INTERSECTION(c\_a.shape, c\_c.shape, 0.005)

FROM university\_campc\_a, university\_campc\_c

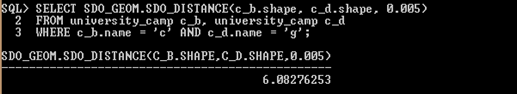
WHERE c\_a.name = 'd' AND c\_c.name = 'e';



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

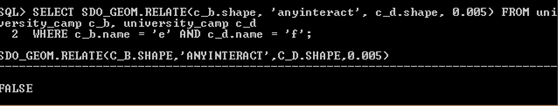
WHERE c\_b.name = 'c' AND c\_d.name = 'g';



SELECT SDO\_GEOM.RELATE(c\_b.shape, 'anyinteract', c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

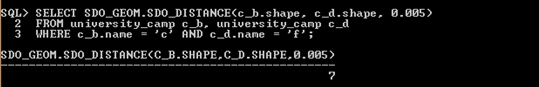
WHERE c\_b.name = 'e' AND c\_d.name = 'f';



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

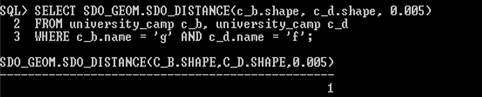
WHERE c\_b.name = 'c' AND c\_d.name = 'f';



SELECT SDO\_GEOM.SDO\_DISTANCE(c\_b.shape, c\_d.shape, 0.005)

FROM university\_campc\_b, university\_campc\_d

WHERE c\_b.name = 'g' AND c\_d.name = 'f';



**Practical No.9**

**Implement Prolog Programming.**

a) Map colorings.

b) Two factorial definitions.

c) Towers of Hanoi puzzle.

d) Tree data and relations.

e) Animal identification game.

A] Map colorings

Source Code:

adjacent(1,2). adjacent(2,1).

adjacent(1,3). adjacent(3,1).

adjacent(1,4). adjacent(4,1).

adjacent(1,5). adjacent(5,1).

adjacent(2,3). adjacent(3,2).

adjacent(2,4). adjacent(4,2).

adjacent(3,4). adjacent(4,3).

adjacent(4,5). adjacent(5,4).

color(1,red,a). color(1,red,b).

color(2,blue,a). color(2,blue,b).

color(3,green,a). color(3,green,b).

color(4,yellow,a). color(4,blue,b).

color(5,blue,a). color(5,green,b).

conflict(Coloring) :-

adjacent(X,Y),

color(X,Color,Coloring),

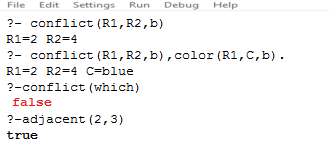
color(Y,Color,Coloring).

conflict(R1,R2,Coloring) :-

adjacent(R1,R2),

color(R1,Color,Coloring),

color(R2,Color,Coloring).



B] Two factorial definitions :

Source Code:

factorial(0,1).

factorial(N,F) :-

N>0,

N1 is N-1,

factorial(N1,F1),

F is N \* F1.

factorial(0,F,F).

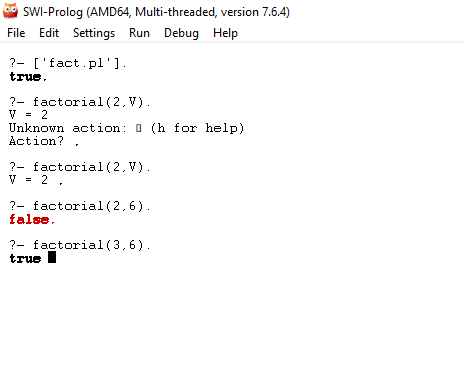
factorial(N,A,F) :-

N > 0,

A1 is N\*A,

N1 is N -1,

factorial(N1,A1,F).

****

**C] Towers of Hanoi puzzle**

**Source Code:**

move(1,X,Y,\_) :-

write('Move top disk from '),

write(X),

write(' to '),

write(Y),

nl.

move(N,X,Y,Z) :-

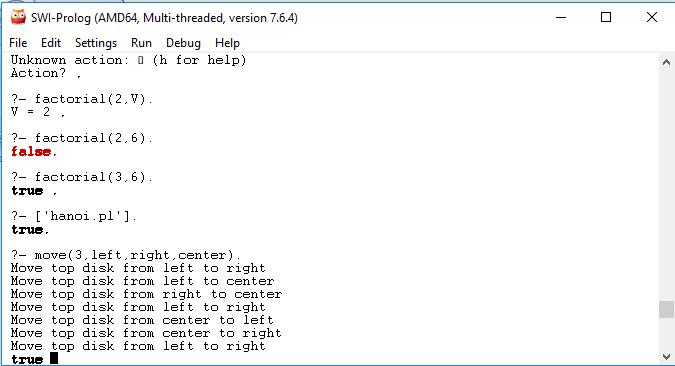
N>1,

M is N-1,

move(M,X,Z,Y),

move(1,X,Y,\_),

move(M,Z,Y,X).

****

**D] Tree data and relations**

**Source Code:**

:- op(500,xfx,'is\_parent').

ais\_parent b. c is\_parent g. f is\_parent l. j is\_parent q.

ais\_parent c. c is\_parent h. f is\_parent m. j is\_parent r.

ais\_parent d. c is\_parenti. h is\_parent n. j is\_parent s.

bis\_parent e. d is\_parent j. iis\_parent o. m is\_parent t.

bis\_parent f. e is\_parent k. iis\_parent p.

/\* X and Y are siblings \*/

:- op(500,xfx,'is\_sibling\_of').

X is\_sibling\_ofY :- Z is\_parent X, Z is\_parent Y, X \== Y.

/\* X and Y are on the same level in the tree. \*/

:-op(500,xfx,'is\_same\_level\_as').

X is\_same\_level\_asX .

X is\_same\_level\_asY :- W is\_parent , Z is\_parent Y, W is\_same\_level\_as Z.

/\* Depth of node in the tree. \*/

:- op(500,xfx,'has\_depth').

ahas\_depth 0 :- !.

Node has\_depthD :- Mother is\_parent Node, Mother has\_depth D1,

D is D1 + 1.

/\* Locate node by finding a path from root down to the node. \*/

locate(Node) :- path(Node), write(Node), nl.

path(a). /\* Can start at a. \*/

path(Node) :- Mother is\_parent Node, /\* Choose parent, \*/

path(Mother), /\* find path and then \*/

write(Mother),

write(' --> ').

/\* Calculate the height of a node, length of longest path to

a leaf under the node. \*/

height(N,H) :- setof(Z,ht(N,Z),Set), /\* See section 2.8 for 'setof'. \*/

max(Set,0,H).

ht(Node,0) :- leaf(Node), !.

ht(Node,H) :- Node is\_parent Child,

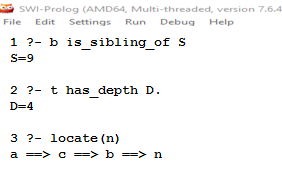
ht(Child,H1),

H is H1 +1.

leaf(Node) :- not(is\_parent(Node,Child)). /\* Node grounded \*/

max([],M,M).

max([X|R],M,A) :- (X > M -> max(R,X,A) ; max(R,M,A)).



**E] Animal identification game**

**Source Code:**

/\* start with ?- go. \*/

go :- hypothesize(Animal), write('I guess that the animal is: '), write(Animal), nl, undo.

/\* hypotheses to be tested \*/

hypothesize(cheetah) :- cheetah, !.

hypothesize(tiger) :- tiger, !.

hypothesize(giraffe) :- giraffe, !.

hypothesize(zebra) :- zebra, !.

hypothesize(ostrich) :- ostrich, !.

hypothesize(penguin) :- penguin, !.

hypothesize(albatross) :- albatross, !.

hypothesize(unknown). /\* no diagnosis \*/

/\* animal identification rules \*/

cheetah :- mammal,

carnivore,

verify(has\_tawny\_color),

verify(has\_dark\_spots).

tiger :- mammal,

carnivore,

verify(has\_tawny\_color),

verify(has\_black\_stripes).

giraffe :- ungulate,

verify(has\_long\_neck),

verify(has\_long\_legs).

zebra :- ungulate,

verify(has\_black\_stripes).

ostrich :- bird,

verify(does\_not\_fly),

verify(has\_long\_neck).

penguin :- bird,

verify(does\_not\_fly),

verify(swims),

verify(is\_black\_and\_white).

albatross :- bird,

verify(appears\_in\_story\_Ancient\_Mariner),

verify(flys\_well).

/\* classification rules \*/

mammal :- verify(has\_hair), !.

mammal :- verify(gives\_milk).

bird :- verify(has\_feathers), !.

bird :- verify(flys),

verify(lays\_eggs).

carnivore :- verify(eats\_meat), !.

carnivore :- verify(has\_pointed\_teeth),

verify(has\_claws),

verify(has\_forward\_eyes).

ungulate :- mammal,

verify(has\_hooves), !.

ungulate :- mammal,

verify(chews\_cud).

/\* how to ask questions \*/

ask(Question) :-

write('Does the animal have the following attribute: '),

write(Question),

write('? '),

read(Response),

nl,

( (Response == yes ; Response == y)

->

assert(yes(Question)) ;

assert(no(Question)), fail).

:- dynamic yes/1,no/1.

/\* How to verify something \*/

verify(S) :-

(yes(S)

->

true ;

(no(S)

->

fail ;

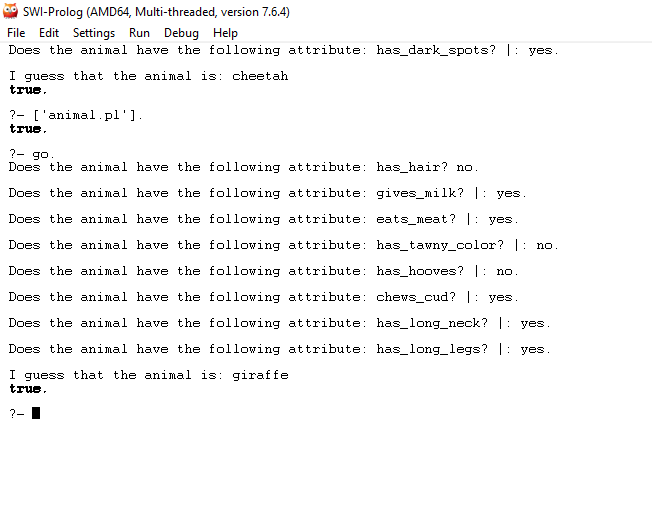
ask(S))).

/\* undo all yes/no assertions \*/

undo :- retract(yes(\_)),fail.

undo :- retract(no(\_)),fail.

undo.

****

**Practical No.10**

**Aim:-**

**1. Create XML Parser**

**2. Using XML DOM Traverse XML Document**

**Source Code:-**

XML file for parsing in Java

Here is xml file Stocks.xml which contains some stocks and there price, quantity we will use this in our xml parsing example in Java.

<?xml version="1.0" encoding="UTF-8"?>

<stocks>

       <stock>

              <symbol>Citibank</symbol>

              <price>100</price>

              <quantity>1000</quantity>

       </stock>

       <stock>

              <symbol>Axis bank</symbol>

              <price>90</price>

              <quantity>2000</quantity>

       </stock>

</stocks>

Code Example of Parsing XML File in Java using DOM Parser

Here is a code example of parsing above xml file in Java using DOM parser:

import java.io.File;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import org.w3c.dom.Document;

import org.w3c.dom.Element;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

public class DOMExampleJava {

public static void main(String args[]) {

try {

File stocks = new File("Stocks.xml");

DocumentBuilderFactorydbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilderdBuilder = dbFactory.newDocumentBuilder();

Document doc = dBuilder.parse(stocks);

doc.getDocumentElement().normalize();

System.out.println("root of xml file" + doc.getDocumentElement().getNodeName());

NodeList nodes = doc.getElementsByTagName("stock");

System.out.println("==========================");

for (int i = 0; i<nodes.getLength(); i++) {

Node node = nodes.item(i);

if (node.getNodeType() == Node.ELEMENT\_NODE) {

Element element = (Element) node;

System.out.println("Stock Symbol: " + getValue("symbol", element));

System.out.println("Stock Price: " + getValue("price", element));

System.out.println("Stock Quantity: " + getValue("quantity", element));

}

}

} catch (Exception ex) {

ex.printStackTrace();

}

}

private static String getValue(String tag, Element element) {

NodeList nodes = element.getElementsByTagName(tag).item(0).getChildNodes();

Node node = (Node) nodes.item(0);

return node.getNodeValue();

}

}

Output:

root of xml file stocks

==========================

Stock Symbol: Citibank

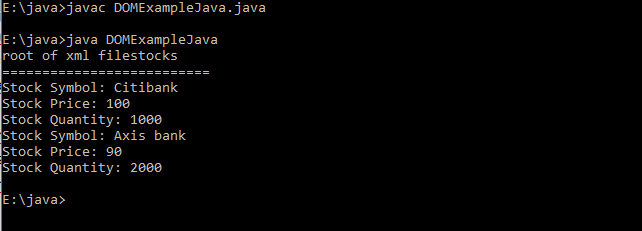
Stock Price: 100

Stock Quantity: 1000

Stock Symbol: Axis bank

Stock Price: 90

Stock Quantity: 2000



**Practical No.11**

**Inserting and Retrieving Multimedia Objects in Database (Image/audio/video).**

Coding:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Data.SqlClient;

using System.IO;

namespace ImageSaveToSQLServer

{

publicpartialclassForm1 : Form

{

SqlConnection conn = newSqlConnection(@"Data Source=VAIO\SQLEXPRESS;Initial Catalog=MultimediaDB;Integrated Security=True");

SqlCommand command;

string imgLoc = "";

public Form1()

{

InitializeComponent();

}

privatevoid Form1\_Load(object sender, EventArgs e)

{

}

privatevoid buttonBrowse\_Click(object sender, EventArgs e)

{

try

{

OpenFileDialog dlg = newOpenFileDialog();

dlg.Filter = "JPG Files(\*.jpg)|\*.jpg|GIF Files(\*.gif)|\*.gif|All Files(\*.\*)|\*.\*";

dlg.Title = "Select employee picture";

if (dlg.ShowDialog() == DialogResult.OK)

{

imgLoc = dlg.FileName.ToString();

picEmp.ImageLocation = imgLoc;

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

privatevoid buttonSave\_Click(object sender, EventArgs e)

{

try

{

byte[] img = null;

FileStream fs = newFileStream(imgLoc, FileMode.Open, FileAccess.Read);

BinaryReader br = newBinaryReader(fs);

img = br.ReadBytes((int)fs.Length);

string sql = "insert into Employee(EID,FIRST\_NAME,LAST\_NAME,IMAGE)values(" + textBoxEID.Text + ",'" + textBoxFName.Text + "','" + textBoxLName.Text + "',@img)";

if (conn.State != ConnectionState.Open)

conn.Open();

command = newSqlCommand(sql, conn);

command.Parameters.Add(newSqlParameter("@img", img));

int x = command.ExecuteNonQuery();

conn.Close();

MessageBox.Show(x.ToString() + "record(s) saved.");

textBoxEID.Text = "";

textBoxFName.Text = "";

textBoxLName.Text = "";

picEmp.Image = null;

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show(ex.Message);

}

}

privatevoid buttonShow\_Click(object sender, EventArgs e)

{

try

{

string sql="SELECT FIRST\_NAME,LAST\_NAME,IMAGE FROM Employee WHERE EID="+textBoxEID.Text+"";

if (conn.State != ConnectionState.Open)

conn.Open();

command = newSqlCommand(sql, conn);

SqlDataReader reader = command.ExecuteReader();

reader.Read();

if (reader.HasRows)

{

textBoxFName.Text = reader[0].ToString();

textBoxLName.Text = reader[1].ToString();

byte[] img = (byte[])(reader[2]);

if (img == null)

picEmp.Image = null;

else

{

MemoryStream ms = newMemoryStream(img);

picEmp.Image = Image.FromStream(ms);

}

}

else

{

MessageBox.Show("This does not Exist.");

}

conn.Close();

}

catch (Exception ex)

{

conn.Close();

MessageBox.Show(ex.Message);

}

}

}

}

