INDEX (Section II)

Sr. No.	Topic	Page No.	Signature
1	Distributed Databases(Vertical)	2	
2	Distributed Databases(Horizontal)	2	
3	Distributed Databases(Replication)	2	
4	Object Oriented Databases	12	
5	Multimedia Databases	19	
6	Temporal Databases	23	
7	Active Databases	26	
8	XML Databases	31	
9	Spatial Databases	35	

Practical No. 1, 2, 3

Topic: Distributed Databases

Problem Statement:

1. Create a global conceptual schema Emp (Eno; Ename; Address; Email; Salary) and insert 5 records. Divide Emp into vertical fragments Emp1 (Eno; Ename; Address) and Emp2 (Eno; Email; Salary) on two different nodes. Fire the following 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 = 20,000.

2. Create a global conceptual schema Emp (Eno; Ename; Address; Email; Salary) and insert 5 records. Divide Emp into horizontal fragments on two different nodes. Fire the following queries:
    1. Find the salary of all employees.
    2. Find the Email of all employees where salary = 20,000
    3. Find the employee name and Email where employee number is known.
    4. Find the employee name and address where employee number is known.

3. Create a 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:
    1. Find the salary of all employees.
    2. Find the Email of all employees where salary = 20,000
    3. Find the employee name and Email where employee number is known.
    4. Find the employee name and address where employee number is known.

Description:

Creation of Database Links*:* Create database links using the CREATE DATABASE LINK statement. The table gives examples of SQL statements that create database links in a local database to the remote sales.us.americas.acme_auto.com database:

SQL Statement	Connects To Database	Connects As	Link Type
CREATE DATABASE LINK sales.us.americas.acme_auto.com USING 'sales_us';	sales using net service name sales_us	Connected user	Private connected user
CREATE DATABASE LINK foo CONNECT TO CURRENT_USER USING 'am_sls';	sales using service name am_sls	Current global user	Private current user
CREATE DATABASE LINK sales.us.americas.acme_auto.com CONNECT TO scott IDENTIFIED BY tiger USING 'sales_us';	sales using net service name sales_us	scott using password tiger	Private fixed user
CREATE PUBLIC DATABASE LINK sales CONNECT TO scott IDENTIFIED BY tiger USING 'rev';	sales using net service name rev	scott using password tiger	Public fixed user

CREATE SHARED PUBLIC DATABASE LINK sales.us.americas.acme_auto.com CONNECT TO scott IDENTIFIED BY tiger AUTHENTICATED BY anupam IDENTIFIED BY bhide USING 'sales';	sales using net service name sales	scott using password tiger, authenticated as anupam using password bhide	Shared public fixed user

Vertical Fragmentation

Creating Emp_1 at Node1

create table emp_1 (eno varchar2(4) PRIMARY KEY,ename varchar2(10),address varchar2(50));

SQL> insert into emp_1 values ('&eno','&ename','&address');
Enter value for eno: e001
Enter value for ename: Salil
Enter value for address: Andheri
old 1: insert into emp1 values ('&eno','&ename','&address')
new 1: insert into emp1 values ('e001','Salil','Andheri')

1 row created.

SQL> /
Enter value for eno: e002
Enter value for ename: Divyata
Enter value for address: Bandra
old 1: insert into emp1 values ('&eno','&ename','&address')
new 1: insert into emp1 values ('e002','Divyata','Bandra')

1 row created.

SQL> /
Enter value for eno: e003
Enter value for ename: Crima
Enter value for address: Matunga
old 1: insert into emp1 values ('&eno','&ename','&address')
new 1: insert into emp1 values ('e003','Crima','Matunga')

1 row created.

SQL> /
Enter value for eno: e004
Enter value for ename:Harshada
Enter value for address: Wadala
old 1: insert into emp_1 values ('&eno','&ename','&address')
new 1: insert into emp_1 values ('e004','Harshada','Wadala')

1 row created.

```
SQL> /
Enter value for eno: e005
Enter value for ename: Haneet
Enter value for address: Andheri
old   1: insert into emp_1 values ('&eno','&ename','&address')
new   1: insert into emp_1 values ('e005','Haneet','Andheri')

1 row created.

SQL> select * from emp_1;

ENO  ENAME      ADDRESS
---- ---------- ----------------------
e001  Salil     Andheri
e002  Divyata      Bandra
e003  Crima   Matunga
e004  Harshada       Wadala
e005  Haneet         Andheri
```

Creating Emp_2 at Node2

```
create table emp_2 (eno varchar2(4) PRIMARY KEY,email varchar2(50),salary
number(7,2));
insert into emp_2 values ('&eno','&email',&salary);
Enter value for eno: e001
Enter value for email: my_name_Salil@rediffmail.com
Enter value for salary: 25000
old   1: insert into emp2 values ('&eno','&email',&salary)
new   1: insert into emp2 values ('e001','my_name_Salil@rediffmail.com',25000)

1 row created.

SQL> /
Enter value for eno: e002
Enter value for email: Divyata_Dal@rediffmail.com
Enter value for salary: 20000
old   1: insert into emp2 values ('&eno','&email',&salary)
new   1: insert into emp2 values ('e002','Divyata_Dal@rediffmail.com',20000)

1 row created.

SQL> /
Enter value for eno: e003
Enter value for email: crima@rediffmail.com
Enter value for salary: 24000
old   1: insert into emp2 values ('&eno','&email',&salary)
new   1: insert into emp2 values ('e003','crima@rediffmail.com',24000)

1 row created.

SQL> /
```

```
Enter value for eno: e004
Enter value for email: harshada@rediffmail.com
Enter value for salary: 10000
old   1: insert into emp2 values ('&eno','&email',&salary)
new   1: insert into emp2 values ('e003', 'harshada@rediffmail.com ',10000)

1 row created.

SQL> /
Enter value for eno: e005
Enter value for email: haneet@yahoo.co.in
Enter value for salary: 16000
old   1: insert into emp2 values ('&eno','&email',&salary)
new   1: insert into emp2 values ('e003',' haneet@yahoo.co.in ',16000)

1 row created.

SQL> select * from emp_2@prac1_link;

ENO  EMAIL                                          SALARY
---- ---------------------------------------------- ----------
e001 my_name_Salil@rediffmail.com                    25000
e002 Divyata@yahoo.com                               20000
e003 Crima@yahoo.co.in                               24000
e004 harshada@rediffmail.com                         10000
e005 haneet@yahoo.co.in                              16000
```

Connecting as System User

```
SQL> connect
Enter user-name: system
Enter password: *****
Connected.
```

Creating Public Database Links at the Central Machine

```
SQL> create public database link "prac1_link" connect to "scott" identified by "tiger" using
'orcl14';
```

1) To Find the salary of an employee where employee number is known

```
SQL> select salary from emp_2@prac1_link where eno='e001';

   SALARY
   ----------
    20000
```

2) To Find the Email where the employee name is known

```
SQL> select email from emp_2@prac1_link e2,emp_1@orcl e1 where e2.eno=e1.eno and
e1.ename='HARSHADA';

EMAIL
--------------------------------------------------
harshada@rediffmail.com
```

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

```
SQL> select ename,email from emp_1 e1,emp_2@prac1_link e2 where e1.eno=e2.eno and
e1.eno='e005';

ENAME      EMAIL
---------- --------------------------------------------------
HANEET      haneet@yahoo.co.in
```

4) To Find the employee name whose salary is =20000.

```
SQL> select ename from emp_1 e1,emp_2@prac1_link e2 where e1.eno=e2.eno and
e2.salary=20000;

ENAME
----------
DIVYATA
```

Horizontal Fragmentation

Creating Emp_21 at Node1

create table emp_21 (eno varchar2(4) PRIMARY KEY,ename varchar2(10),address varchar2(50), email varchar2(50),salary number(7,2));

SQL> SELECT * from emp_21;

ENO	ENAME	ADDRESS	EMAIL	SALARY
e001	SALIL	ANDHERI	my_name_Salil@rediffmail.com	20000
e003	CRIMA	BANDRA	Crima@yahoo.co.in	20000
e005	HANEET	ANDHERI	haneet@yahoo.co.in	16000

Creating Emp_22 at Node2

create table emp_22 (eno varchar2(4) PRIMARY KEY,ename varchar2(10),address varchar2(50), email varchar2(50),salary number(7,2));

SQL> select * from emp_22@prac1_link;

ENO	ENAME	ADDRESS	EMAIL	SALARY
e002	DIVYATA	BANDRA	Divyata@yahoo.com	10000
e004	HARSHADA	WADALA	harshada@rediffmail.com	10000

Creating Public Database Links at the Central Machine

SQL> connect
Enter user-name: system
Enter password: *****
Connected.

SQL> create public database link "prac1_link" connect to "scott" identified by "tiger" using 'orcl14';

1) To Find the salary of all employees.

```
SQL> select salary from emp_21 union select salary emp_22@prac1_link;

SALARY
----------
20000
20000
16000
10000
10000
```

2) To Find the Email of all employees where salary = 20000

```
SQL> select email from emp_21 where salary=20000  union
select ename, email from emp22@ prac1_link where salary=20000;

EMAIL
--------------------------------------------------
my_name_Salil@rediffmail.com
Crima@yahoo.co.in
```

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

```
SQL> select ename,email from emp_22@prac1_link e2 where e2.eno='e002'
union select ename,email from emp_21 e1 where e1.eno='e002';

ENAME      EMAIL
---------- --------------------------------------------------
DIVYATA     Divyata@yahoo.com
```

4) To Find the employee name and address where employee number is known

```
SQL> select ename,address from emp_21 e1 where e1.eno='e001' union select ename,email
from emp_22@prac1_link e2 where e2.eno='e001';

ENAME      ADDRESS
---------- --------------------------------------------------
SALIL     ANDHERI
```

Replication

Creating Emp_3 at Node1

SQL> create table emp_3 (eno varchar2(4) PRIMARY KEY,ename varchar2(10),address varchar2(50), email varchar2(50),salary number(7,2));

SQL> select * from emp_3;

ENO	ENAME	ADDRESS	EMAIL	SALARY
e001	SALIL	ANDHERI	my_name_Salil@rediffmail.com	20000
e002	DIVYATA	BANDRA	Divyata@yahoo.com	10000
e003	CRIMA	BANDRA	Crima@yahoo.co.in	20000
e004	HARSHADA	WADALA	harshada@rediffmail.com	10000
e005	HANEET	ANDHERI	haneet@yahoo.co.in	16000

Creating Emp_3 at Node2

SQL> select * from emp_3;

ENO	ENAME	ADDRESS	EMAIL	SALARY
e001	SALIL	ANDHERI	my_name_Salil@rediffmail.com	20000
e002	DIVYATA	BANDRA	Divyata@yahoo.com	10000
e003	CRIMA	BANDRA	Crima@yahoo.co.in	20000
e004	HARSHADA	WADALA	harshada@rediffmail.com	10000
e005	HANEET	ANDHERI	haneet@yahoo.co.in	16000

Creating Public Database Links at the Central Machine

SQL> connect
Enter user-name: system
Enter password: *****
Connected.

SQL> create public database link "prac1_link" connect to "scott" identified by "tiger" using 'orcl14';

1) To Find the salary of all employees.

```
SQL> select salary from emp_3@prac1_link;

    SALARY
----------
     20000
     10000
     20000
     10000
     16000

SQL> select salary from emp_3;

    SALARY
----------
     20000
     10000
     20000
     10000
     16000
```

2) To Find the Email of all employees where salary = 20000

```
SQL> select email from emp_3 where salary=15000;

EMAIL
--------------------------------------------------
my_name_Salil@rediffmail.com
Crima@yahoo.co.in

SQL> select email from emp_3@prac1_link e3 where e3.salary=15000;

EMAIL
--------------------------------------------------
my_name_Salil@rediffmail.com
Crima@yahoo.co.in
```

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

```
SQL> select email,ename from emp_3@prac1_link e3 where e3.eno='e002';

EMAIL                                    ENAME
---------------------------------------- ----------
Divyata@yahoo.com                        DIVYATA

SQL> select email,ename from emp_3 where eno='e002';

EMAIL                                    ENAME
---------------------------------------- ----------
```

Divyata@yahoo.com DIVYATA

4) To Find the employee name and address where employee number is known

SQL> select ename,address from emp_3 where eno='e004';

ENAME ADDRESS
---------- --
HARSHADA WADALA

SQL> select ename,address from emp_3@prac1_link e3 where e3.eno='e004';

ENAME ADDRESS
---------- --
HARSHADA WADALA

Practical No. 4

Topic: Object Oriented Databases

Problem Statement:

Using Object Oriented databases create the following types:

a) AddrType (Pincode: number, Street :char, City : char, state :char)
b) BranchType (address: AddrType, phone1: integer,phone2:integer)
c) AuthorType (name:char,addr AddrType)
d) PublisherType (name: char, addr: AddrType, branches: BranchTableType
e) AuthorListType as varray, which is a reference to AuthorType

Next create the following tables:

f) BranchTableType of BranchType
g) authors of AuthorType
h) books(title:varchar, year:date, published_by ref PublisherType,authors AuthorListType)
i) Publishers of PublisherType

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

a) List all of the authors that have the same pin code as their publisher:
b) List all books that have 2 or more authors:
c) List the name of the publisher that has the most branches
d) Name of authors who have not published a book
e) List all authors who have published more than one book:
f) Name of authors who have published books with at least two different publishers
g) 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).

Overview: An object oriented database management system (OODBMS) is the coupling of database and object oriented concepts. OODBMS concepts have their origin in object oriented programming language that contains abstract data types, encapsulation, information hiding methods, class/type hierarchies, inheritance etc.

An object has state (value) and behavior (operation).

For Example: An Employee can be an object with state as fields eno, ename, bdate, salary and behavior may be finding age operation to find the age of an employee using bdate.

1) Create different Types and Tables

```
create type addrtype as object (pincode number(6),street varchar2(20),city varchar2(20),state
varchar2(20));
2 /

create type branchtype as object (address addrtype,phone1 number(8),phone2 number(8));
  2  /

create type authortype as object (name varchar2(10), addr addrtype);
  2  /

create type publishertype as object (name varchar2(30), addr addrtype, branches
branchtabletype);
  2  /

create type authorlisttype as varray(2) of ref authortype;
  2  /

create type branchtabletype as table of branchtype;
  2  /

create table authors of authortype;

create table books (title varchar2(10), year date ,published_by ref publishertype, authors
authorlisttype);

create table publishers of publishertype nested table branches store as branchtable;
```

2) Insertion of different values in the tables

Table Name: Authors
Format : Authors(name, addr:<pincode,street,city,state,no>)

/*Here addr is the name of an object-valued attribute, which has 5 components.*/

```
insert into authors values ('&name',addrtype(&pincode,'&street','&city','&state'));
Enter value for name: Aho
Enter value for pincode: 400068
Enter value for street: s.v.rd.
Enter value for city: Mumbai
Enter value for state: Maharashtra
old   1: insert into authors values ('&name',addrtype(&pincode,'&street','&city','&state'))
new   1: insert into authors values ('Aho',addrtype(400068,'s.v.rd.','Mumbai','Maharashtra'))

1 row created.

SQL> /
Enter value for name: Stallings
Enter value for pincode: 400052
Enter value for street: r.t.rd.
Enter value for city: Pune
Enter value for state: Maharashtra
old   1: insert into authors values ('&name',addrtype(&pincode,'&street','&city','&state'))
new   1: insert into authors values ('Stallings',addrtype(400052,'r.t.rd.','Pune','Maharashtra'))

1 row created.

SQL> /
Enter value for name: Jerry
Enter value for pincode: 400092
Enter value for street: j.s.rd.
Enter value for city: Nashik
Enter value for state: Maharashtra
old   1: insert into authors values ('&name',addrtype(&pincode,'&street','&city','&state'))
new   1: insert into authors values ('Jerry',addrtype(400092,'j.s.rd.','Nashik','Maharashtra'))

1 row created.

SQL> /
Enter value for name: Provakis
Enter value for pincode: 500012
Enter value for street: l.t.rd.
Enter value for city: Baroda
Enter value for state: Gujarat
old   1: insert into authors values ('&name',addrtype(&pincode,'&street','&city','&state'))
new   1: insert into authors values ('Provakis',addrtype(500012,'l.t.rd.','Baroda','Gujarat'))

1 row created.
```

```
SQL> /
Enter value for name: Rabiner
Enter value for pincode: 500014
Enter value for street: s.v.rd.
Enter value for city: Nagpur
Enter value for state: Maharashtra
old   1: insert into authors values ('&name',addrtype(&pincode,'&street','&city','&state'))
new   1: insert into authors values
('Rabiner',addrtype(500014,'s.v.rd.','Nagpur','Maharashtra'))

1 row created.
```

Table name : Publishers
Format : Publishers(name, addr:<pincode,street,city,state,no>, branches:set of<address:<pincode,street,city,state,no>,phone1,phone2>)

/*where addr is an object-valued attribute. Branches is a complex-valued attribute, in this case a nested table where each element in the table has 3 parts: an address and two phones*/

```
Insert into publishers values('BPB publication',
addrtype(400068,'s.v.rd.','Mumbai','Maharashtra'),branchtabletype(branchtype(addrtype(4000
68,'s.v.rd.','Mumbai','Maharashtra'),28964401,28946601)));

Insert into publishers values('Navneet Prakashan',
addrtype(400068,'r.m.rd','Mumbai','Maharashtra'),branchtabletype(branchtype(addrtype(4000
68,'r.m.rd','mumbai','maharashtra'),28952210,28952211)));

Insert into publishers values('Sheth',
addrtype(500014,'s.v.rd','Nagpur','Maharashtra'),branchtabletype(branchtype(addrtype(50001
4,'s.v.rd','Nagpur','Maharashtra'),28952210,28952211)));

Insert into publishers values('Chetna Publication',
addrtype(500012,'l.t.rd','Baroda','Gujarat'),branchtabletype(branchtype(addrtype(500012,'l.t.r
d','Baroda','Gujarat'),28952210,28952211)));
```

Table Name : books
Format : books(title,year,published_by: ref<Publishers>, authors:list of ref Author)

/*Where published_by is a reference to elements of the table Publishers; authors is a complex-valued attribute, in this case a list of references to objects of the type author(managed by table 'Author'.*/

insert into books select 'java','20-may-1983',ref(pub),authorlisttype(ref(aut)) from publishers pub,authors aut where pub.name='BPB publication' and aut.name='Stallings';

insert into books select 'compiler','19-june-1980',ref(pub),authorlisttype(ref(aut)) from publishers pub,authors aut where pub.name='Navneet Prakashan' and aut.name='Aho';

insert into books select 'Simulation','15-august-1970',ref(pub),authorlisttype(ref(aut)) from publishers pub,authors aut where pub.name='Sheth' and aut.name='Jerry';

insert into books select 'DSP','09-december-1965',ref(pub),authorlisttype(ref(aut)) from publishers pub,authors aut where pub.name='Chetna Publication' and aut.name='Provakis';

3) Firing queries on the created tables

a) List all of the authors that have the same address as their publisher

select a.name from authors a, publishers p where a.addr = p.addr;

NAME

Aho

b) List all of the authors that have the same pin code as their publisher

select a.name from authors a,publishers p where p.addr.pincode=a.addr.pincode;

NAME

Aho
Aho
Provakis
Jerry

c) List all books that have 2 or more authors

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

no rows selected

d) List the title of the book that has the most authors

Select title from books b, table(b.authors) group by title having count(*) = (select max(count(*)) from books b, table(b.authors) group by title);

TITLE

DSP
Simulation
compiler
java

e) 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);

NAME

BPB publication
Chetna Publication
Navneet Prakashan
Sheth

f) Name of authors who have not published a book

select (a.name) from authors a minus select a.name from books b, table(b.authors) b1, authors a where b1.column_value=ref(a);

NAME

Rabiner

g) Move all the branches that belong to the publisher 'tata' to the publisher ‘‘joshi'

insert into table(select branches from publishers where name = 'Navneet Prakashan') select b.* from publishers p, table(p.branches) b where name = 'Sheth';

/*Here all the branches belonging to 'tata' is moved(copied) to joshi's branch i.e joshi will have its own branch with 3 more branches of tata that tata has*/

h) List all authors who have published more than one book

select a.name from authors a, books b, table(b.authors) b1 where b1.column_value =ref(a) group by a.name having count(*) > 1;

no rows selected

/*This query will give the names of those authors who have published a book*/

select a.name from Authors a,books b,table(b.authors) v where v.column_value=ref(a) group by a.name;

i) 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).

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

no rows selected

Practical No. 5

Topic: Multimedia Databases

Problem Statement:

Create a table Emp with the attributes Eno as employee number, Ename as employee name, Eaddress as employee address and photo as an employee picture. Also create a table Company with attributes Eno,designation,age. Fire the following queries

a) Find name and designation of all the employees
b) Find name and age of all the employees

[B] Create a table Singer with the attributes sno as singer number, Sname as singer name, Saddress as singer address and audio as an audio clip. Also create a table Company with attributes Sno,age. Fire the following queries

a) Find name and age of all the singer
b) Retrieving the Content Length of a audio file

[C] Create a table Singer with the attributes sno as singer number, Sname as singer name, Saddress as singer address and video as an audio clip. Also create a table Company with attributes Sno,age. Fire the following queries

a) Find name and age of all the singer
b) Retrieving the Content Length of a video file

Storing an Image File:

Create or replace directory photo as 'd:\Salil\pracs\oracle pracs\';

[C:\MSCI\Oracle]

Directory created.

SQL> Create table emp1(eno number(4) PRIMARY KEY, ename varchar2(20), photo ORDSYS.ORDImage);

Table created.

SQL> Create table compimg(eno number(4) REFERENCES emp1(eno) ON DELETE CASCADE , desg varchar2(20),age number(2));

Table created.

SQL> Insert into emp1 values(1,'Salil',ORDSYS.ORDImage.init('file','photo','1.jpeg'))

1 row created.

SQL> Insert into compimg values (1,'Software Engineer',25);

1 row created.

a) Find name and designation of all the employees

SQL> select e.ename,c.desg from emp1 e, compimg c where e.eno=c.eno;

```
ENAME                DESG
-------------------- --------------------
Salil                Software Engineer
```

b) Find name and age of all the employees

SQL> select e.ename, c.age from emp1 e, compimg c where e.eno=c.eno;

```
ENAME                    AGE
-------------------- ----------
Salil                    25
```

Storing a Audio File:

Create or replace directory AUDIO as 'd://Salil//pracs//oracle pracs//';

Directory created.

SQL> Create table singer(sno number(4) PRIMARY KEY, sname varchar2(20), aud ORDSYS.ORDAudio);

Table created.

SQL> Create table compaud(sno number(4) REFERENCES singer(sno) ON DELETE CASCADE ,age number(2));

Table created.

SQL> Insert into singer values (1,'Salil',ORDSYS.ORDAudio.init('file' , 'AUDIO' , '1.mp3'));

1 row created.

SQL> Insert into compaud values (1,25);

1 row created.

a) Find name and age of all the singer

SQL> select s.sname, c.age from singer s, compaud c where s.sno=c.sno;

```
SNAME                AGE
-------------------- ----------
Salil                25
```

b) Retrieving the Content Length of a audio file

SQL> select sno,s.aud.getContentLength() from singer s;

```
    SNO    S.AUD.GETCONTENTLENGTH()
---------- ----------------------------------------
      1                2032118
```

Storing a Video File:

```
SQL> Create or replace directory VIDEO as 'd://Salil//pracs//oracle pracs//';

Directory created.

SQL> Create table singer1(sno number(4) PRIMARY KEY, sname varchar2(20), vid
ORDSYS.ORDVideo);

Table created.

SQL> Create table compvid(sno number(4) REFERENCES singer(sno) ON DELETE
CASCADE ,age number(2));

Table created.

SQL> Insert into singer1 values (1,'Salil',ORDSYS.ORDVideo.init('file' , 'VIDEO' , '1.wmv'
));

1 row created.

SQL> Insert into compvid values (1,25);

1 row created.
```

a) Find name and age of all the singer

```
SQL> select s.sname, c.age from singer1 s, compvid c where s.sno=c.sno;

SNAME                AGE
-------------------- ----------
Salil                25
```

b) Retrieving the Content Length of a video file

```
SQL> select sno,s.vid.getContentLength() from singer1 s;

   SNO     S.VID.GETCONTENTLENGTH()
---------- ----------------------------------------
         1                  467574
```

Practical No. 6

Topic: Temporal Databases

Problem Statement:

[A]Create a table tblEmp_Appnt, which stores the account number, name, and valid time say, recruitment date and retirement date. Insert 10 records and fire the following queries

a) Find all the employees who join the company on 2/3/2001
b) Find all the employees who will retired on 2/3/2001

[B] 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

a) Find all the names of a company whose share price is more than Rs. 1000 at 2:21 P.M.
b) Find the name of company which has highest share price at 2.24 P.M.

A) Creating the tables

```
Create table tblEmp_Appnt (empno number(3),empname varchar2(20),recruit_date
date,retire_date date);

insert into tblEmp_Appnt values(1,'Salil',to_date(sysdate,'DD-MM-YYYY'), to_date('18-
10-2005','DD-MM-YYYY'));

insert into tblEmp_Appnt values(2,'Muddu', to_date('18-11-1999','DD-MM-
YYYY'),to_date(sysdate,'DD-MM-YYYY'));

insert into tblEmp_Appnt values(3,'Harshada',to_date('2-3-2001','DD-MM-YYYY'),
to_date('2-10-2005','DD-MM-YYYY'));

insert into tblEmp_Appnt values(4,'Shails',to_date('4-1-1990','DD-MM-YYYY'),
to_date('2-3-2001','DD-MM-YYYY'));

insert into tblEmp_Appnt values(5,'Crima',to_date('5-12-1999','DD-MM-YYYY'),
to_date('18-10-2006','DD-MM-YYYY'));

SQL> select * from tblEmp_Appnt;

EMPNO EMPNAME    RECRUIT_DATE   RETIRE_DATE
---------- -------------------- --------- ---------------------------------
    1      Salil           18-OCT-05          18-OCT-10
    2      Muddu             18-NOV-99          18-OCT-05
    3      Harshada            02-MAR-01          02-OCT-05
    4      Shails            04-JAN-90          02-MAR-01
    5      Crima           05-DEC-99          18-OCT-06
```

Firing Queries:

a) Find all the employees who join the company on 2/3/2001

```
SQL> select empname from tblEmp_Appnt where recruit_date=to_date('2-3-2001','DD-MM-
YYYY');

EMPNAME
--------------------
Harshada
```

b) Find all the employees who will retired on 2/3/2001

```
SQL> select empname from tblEmp_Appnt where retire_date=to_date('2-3-2001','DD-MM-
YYYY');

EMPNAME
--------------------
Shails
```

B) Creating Tables:

```
Create table tbl_shares (name varchar2(20),no_shares number(5),price
number(7,2),trans_time date);

Insert into tbl_shares values ('Infosys',12,100,to_date(sysdate,'HH24:Mi'));

Insert into tbl_shares values ('Reliance Infocom',10,80,to_date(sysdate,'HH24:Mi'));

Insert into tbl_shares values ('Gati',100,40,to_date(sysdate,'HH24:Mi'));

Insert into tbl_shares values ('Patni',12,10,to_date(sysdate,'HH24:Mi'));

Insert into tbl_shares values ('Uniphos',10,15,to_date(sysdate,'HH24:Mi'));

SQL> select * from tbl_shares;

NAME              NO_SHARES    PRICE   TRANS_TIME
-------------------------------------------------------------------------
Infosys             12           100      14:21
Reliance Infocom    10           80       14:21
Gati                10           40       14:23
Patni               12           10       14:24
Uniphos             10           15       14:24
```

```
SQL>alter session
set nls_date_format = 'HH24:MI';
```

Firing Queries:

a) Find all the names of a company whose share price is more than Rs. 1000 at 14:21 A.M.

```
SQL> select name from tbl_shares where (no_shares*price)>1000 and
trans_time=to_date('14:21','HH24:Mi');

NAME
--------------------
Infosys
```

b)Find the name of company which has highest share price at 2.24 P.M.

```
SQL> select max(no_shares*price) from tbl_shares where
trans_time=to_date('14:24','HH24:Mi');

select name from  tbl_shares  where no_shares*price= (select max(no_shares*price) from
tbl_shares where trans_time=to_date('02:06','HH24:Mi'))
/

MAX(NO_SHARES*PRICE)
---------------------------------------
              150
```

Practical No. 7

Topic: Active Databases

Problem Statement: Formulate a database using active rules using row and statement level Triggers.

Description:

- Trigger Syntax:

```
CREATE TRIGGER <TRIGGER NAME>
(BEFORE|AFTER)<TRIGGERING EVENT> ON <TABLE NAME>
FOR EACH (ROW|STATEMENT)
[WHEN (CONDITION)]
PL/SQL BLOCK.
```

Consider the following relations:
Emp2

Field Name	Data Type
Eno (PK)	Number(3)
Ename	Varchar(25)
Hrs	Number
Pno	Number
Super_no	Number

Product2

Field Name	Data Type
Pno (PK)	Number
Pname	Varchar2(20)
Thrs	Number
Head_no	Number

Create table Emp2

Create table emp2(eno number(4) PRIMARY KEY,ename varchar2(20), hrs number(2),pno number(4), super_no number(4) UNIQUE);

Create table Project2

Create table project2(pno number(4) PRIMARY KEY,pname varchar2(20),thrs number(3), head_no number(4) REFERENCES emp2(super_no));

SQL> select * from emp2;

ENO	ENAME	HRS	PNO	SUPER_NO
1	Salil	20	1	102
2	Jayesh	10	1	101
3	Arun	15	1	104
4	Ram	12	2	103
5	Shyam	8	2	105

SQL> select * from project2;

PNO	PNAME	THRS	HEAD_NO
1	Compiler	45	101
2	KBC	20	103

Create trigger according to events specified.

Event1: Insert a new employee tuple

Event: New Employee is assigned to a project
Condition: pno value is not Null when emp is assigned to a project
Action: update the total thrs of project by adding hrs to old thrs.

Trigger:

```
SQL> create or replace Trigger thrs
  2    after insert on emp2
  3     for each row
  4        when(New.pno IS NOT NULL)
  5    begin
  6     update project2 set thrs=thrs+:New.hrs
  7     where pno=:New.pno;
  8    end;
  9  /
```

SQL> insert into emp2 values(6,'Krishna',10,2,106);

1 row created.

SQL> select * from project2;

PNO	PNAME	THRS	HEAD_NO
1	Compiler	45	101
2	KBC	30	103

Event2: Changing the hrs of existing employee.

Event: employee is assigned new hrs
Condition: pno value is not NULL when emp is assigned new hrs.
Action: Update thrs of project by adding hrs to thrs & subtract old hrs from it.

Trigger:

```
SQL> create or replace Trigger thrs1
  2  after update of hrs on emp2
  3  for each row
  4  when(New.pno IS NOT NULL)
  5  begin
  6   update project2
  7   set thrs=thrs+:New.hrs-:Old.hrs
  8   where pno=:New.pno;
  9  end;
 10  /
```

SQL> update emp2 set hrs=13 where eno=1;

1 row updated.

SQL> select * from emp2;

ENO	ENAME	HRS	PNO	SUPER_NO
1	Salil	13	1	102
2	Jayesh	10	1	101
3	Arun	15	1	104
4	Ram	12	2	103
5	Shyam	8	2	105
6	Krishna	10	2	106

SQL> select * from project2;

PNO	PNAME	THRS	HEAD_NO
1	Compiler	38	101
2	KBC	30	103

Event3: Changing the project of an employee.

Event: Change of Project
Condition: No condition
Action: Update the thrs of new project by adding new hrs & subtract old hrs from thrs from old project.

Trigger:

```
create or replace Trigger thrs2
  2   after update of pno on emp2
  3    for each row
  4      when(New.pno IS NOT NULL)
  5        begin
  6           update project2 set thrs=thrs+:New.hrs-:Old.hrs
           where pno=:New.pno;
  7        end;
  8  /


Try it

update project2 set thrs=thrs-:Old.hrs
          where pno=:Old.pno;


 SQL> update emp2 set pno=2 where eno=2;

1 row updated.

SQL> select * from project2;

   PNO    PNAME        THRS    HEAD_NO
---------- -------------------- ---------- ----------
     1      Compiler         38       101
     2      KBC              30       103

SQL> select * from emp2;

 ENO     ENAME        HRS     PNO   SUPER_NO
---------- -------------------- ---------- ---------- ----------
     1      Salil            13          1        102
     2      Jayesh            10         2        101
     3      Arun              15         1        104
     4      Ram               12         2        103
     5      Shyam              8         2        105
     6      Krishna           10         2        106

6 rows selected.
```

Event4: Deleting 1 or more project of an employee

.

Event: Deleting the project.
Condition: OLD.pno is not NULL.
Action: update the thrs by subtracting old hrs.

Trigger:

```
SQL> create or replace Trigger thrs3
  2       after delete on emp2
  3       for each row
  4       when (Old.pno IS NOT NULL)
  5       begin
  6       update project2 set thrs=thrs-:Old.hrs where pno=:Old.pno;
  7       end;
  8  /

SQL> delete from emp2 where pno=1;

2 rows deleted.

SQL> select * from project2;

  PNO    PNAME              THRS    HEAD_NO
---------- -------------------- ---------- ----------
    1       Compiler             25        101
    2       KBC                  30        103

SQL> select * from emp2;

   ENO   ENAME        HRS  PNO   SUPER_NO
---------- -------------------- ---------- ---------- ----------
     2       Jayesh           10        2        101
     3       Arun             15        1        104
     4       Ram              12        2        103
     5       Shyam             8        2        105
     6       Krishna          10        2        106
```

Practical No. 8

Topic: XML Databases

Problem Statement:

Create a table employee having (dept_id : number, employee_spec : XMLType).
The employee_spec is a schema with attributes emp id, name, email, acc_no, managerEmail, dateOf Joning. 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 4 th record from the table.

Solution:

1) Creating table with XMLType

Create a table called “employee” having dept_id as “number” datatype and employee_spec as XML datatype (XMLType).The employee_spec is a schema with attributes emp id, name, email, acc_no, managerEmail, dateOf Joning .We create employee_spec like any other user-defined type.

Create table emp1(dept_id number(4), emp_spec XMLType);

2) Inserting values into “employee table”

```
Insert into emp1 values(1001,XMLType(
  2     '<employees>
  3     <emp id="001">
  4    <name>Salil Gala</name>
  5     <email>my_name_Salil@yahoo.com</email>
  6     <acc_no>101</acc_no>
  7      <dateofjoining>1994-12-24</dateofjoining>
  8     </emp>
  9   </employees>'));

Insert into emp1 values(1001,XMLType(
  2  '<employees>
  3     <emp id="002">
  4    <name>Rahul Suryawanshi</name>
  5    <email>rahul_pari_123@yahoo.com</email>
  6     <acc_no>102</acc_no>
  7     <dateofjoining>1995-10-10</dateofjoining>
  8    </emp>
  9    </employees>'));
```

```
Insert into emp1 values(1002,XMLType(
  2     '<employees>
  3     <emp id="003">
  4     <name>Ram Joshi</name>
  5     <email>r_joshi@yahoo.com</email>
  6     <acc_no>103</acc_no>
  7     <dateofjoining>1996-2-10</dateofjoining>
  8     </emp>
  9     </employees>'));
```

3) Firing Queries on XML database.

a) Retrieve the names of employee from employee table

Name of an employee is a child of emp and emp is a child of employees in employee specification.

The query is fired as follows:

```
select e.emp_spec.extract('/employees/emp/name/text()').getStringVal() "names" from emp1
e;

names
--------------------------------------------------------------------------------
Salil Gala
Rahul Suryawanshi
Ram Joshi
```

b) Retrieve the acc_no of employee from employee table

Acc_no of an employee is a child of emp and emp is a child of employees in employee specification.

The query is fired as follows:

```
select e.emp_spec.extract('/employees/emp/acc_no/text()').getStringVal() "acc_no" from
emp1 e;

acc_no
--------------------------------------------------------------------------------
101
102
103
```

c) Retrieve the names, acc_no, email of employees from employee table

The query is fired as follows:

```
select e.emp_spec.extract('/employees/emp/name/text()').getStringVal()
"names",e.emp_spec.extract('/employees/emp/acc_no/text()').getStringVal()
"acc_no",e.emp_spec.extract('/employees/emp/email/text()').getStringVal() "email" from
emp1 e;

names
--------------------------------------------------------------------------------
acc_no
--------------------------------------------------------------------------------
email
--------------------------------------------------------------------------------
Salil Gala
101
my_name_Salil@yahoo.com

Rahul Suryawanshi
102
rahul_pari_123@yahoo.com

names
--------------------------------------------------------------------------------
acc_no
--------------------------------------------------------------------------------
email
--------------------------------------------------------------------------------

Ram Joshi
103
r_joshi@yahoo.com
```

4) Updating XMLType

To update an XML document, you can execute a standard SQL UPDATE statement. You need to bind an `XMLType` instance, as follows:

```
update emp1 e set emp_spec=XMLType(
  2  '<employees>
  3  <emp id="002">
  4   <name>Shyam Singh</name>
  5  </emp>
  6  </employees>')
  7  where
  8  e.emp_spec.extract('/employees/emp/@id').getStringVal()='002';

1 row updated.

SQL> select e.emp_spec.extract('/employees/emp/name/text()').getStringVal() "names" from
emp1 e;

names
--------------------------------------------------------------------------------
Salil Gala
Shyam Singh
Ram Joshi
```

5) Deleting an XMLType Column Row

You can use `extract()` and `existsNode()` functions to identify rows to delete as well. For example to delete all warehouse rows for which the warehouse building is leased, you can write a statement such as:

```
delete from emp1 e where  e.emp_spec.extract('/employees/emp/@id').getStringVal()='003';

1 row deleted.

SQL> select e.emp_spec.extract('/employees/emp/acc_no/text()').getStringVal() "acc_no"
from emp1 e;

acc_no
--------------------------------------------------------------------------------
101
102
```

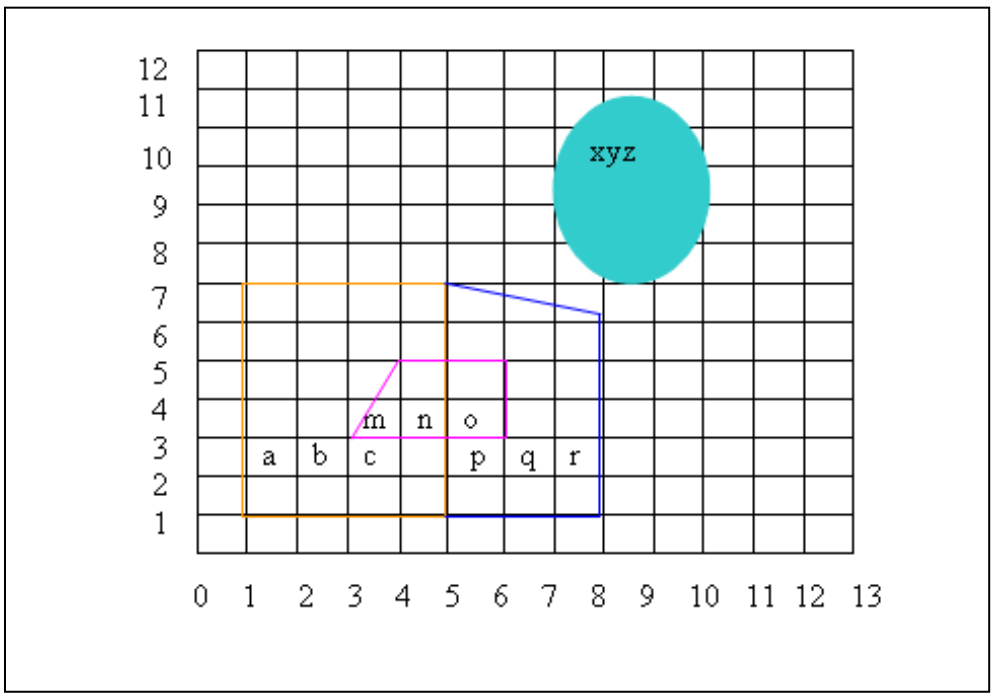
Practical No. 9

Topic: Spatial Databases

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

a) Find the topological intersection of two geometries.
b) Find whether two geometric figures are equivalent to each other.
c) Find the areas of all different locations.
d) Find the area of only one location.
e) Find the distance between two geometries.

Description:
This database consists of a location which consists of the following four areas. They are abc, pqr, mno and xyz. We consider these four as different geographical locations



1) Creating tables

This query creates a table market.

Create table market (mkt_no number(3) primary key,mkt_name varchar2(20),mkt_shape MDSYS.SDO_GEOMETRY);

2) Inserting Values In The Tables

1) This insert query inserts the abc shape in the table.

```
INSERT INTO market VALUES(
        1,
        'abc',
        MDSYS.SDO_GEOMETRY(
                2003, -- 2-dimensional polygon
                NULL,
                NULL,
                MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,3), -- one rectangle (1003
                =exterior)
                MDSYS.SDO_ORDINATE_ARRAY(1,1, 5,7)
                -- only 2 points needed to
                -- define rectangle (lower left and upper right) with
                -- Cartesian-coordinate data
        )
)
/
```

2) This insert query inserts the pqr shape in the table.

```
INSERT INTO market VALUES(
        2,
        'pqr',
        MDSYS.SDO_GEOMETRY(
                2003, -- 2-dimensional polygon
                NULL,
                NULL,
                MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,1),
                -- one polygon (exterior polygon ring)
                MDSYS.SDO_ORDINATE_ARRAY(5,1, 8,1, 8,6, 5,7, 5,1)
        )
)
/
```

3) This insert query inserts the mno shape in the table.

```
INSERT INTO market VALUES(
        3,
        'mno',
        MDSYS.SDO_GEOMETRY(
```

```
            2003, -- 2-dimensional polygon
            NULL,
            NULL,
            MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,1),
            -- one polygon (exterior polygon ring)
            MDSYS.SDO_ORDINATE_ARRAY(3,3, 6,3, 6,5, 4,5, 3,3)
        )
)
/
```

4) This insert query inserts the xyz shape in the table.

```
INSERT INTO market VALUES(
        4,
        'xyz',
        MDSYS.SDO_GEOMETRY(
            2003, -- 2-dimensional polygon
            NULL,NULL,MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,4), -- one circle
            MDSYS.SDO_ORDINATE_ARRAY(8,7, 10,9, 8,11)
        )
)
/
```

To create a table called university with an indexed spatial column, the following SQL statements are needed.

```
INSERT INTO USER_SDO_GEOM_METADATA VALUES (
        'university',
        'shape',
        MDSYS.SDO_DIM_ARRAY(
            MDSYS.SDO_DIM_ELEMENT('X', 0, 20, 0.005),
            MDSYS.SDO_DIM_ELEMENT('Y', 0, 20, 0.005)
        ),
        NULL -- SRID
)
```

The last statement creates the spatial index, which is required for spatial queries
Do this only once for each layer (that is, table-column combination; here: university and SHAPE).

```
CREATE INDEX university_spatial_idx
ON university(shape)
INDEXTYPE IS MDSYS.SPATIAL_INDEX;
```

3) Firing Queries:

1) Topological intersection of two geometries.

```
SQL> SELECT SDO_GEOM.SDO_INTERSECTION(c_a.mkt_shape, c_c.mkt_shape,
0.005)
  2  FROM market c_a, market c_c
  3  WHERE c_a.mkt_name = 'abc' AND c_c.mkt_name = 'mno';

SDO_GEOM.SDO_INTERSECTION(C_A.MKT_SHAPE,C_C.MKT_SHAPE,0.005)(SDO_
GTYPE, SDO_SRID
--------------------------------------------------------------------------------
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 1),
SDO_ORDINATE_ARR
AY(4, 5, 3, 3, 5, 3, 5, 5, 4, 5))
```

2) Do two geometries have any spatial relationship?

```
SELECT SDO_GEOM.RELATE(c_b.shape, 'anyinteract', c_d.shape, 0.005)
FROM university c_b, university c_d
WHERE c_b.name = 'pqr' AND c_d.name = 'xyz';

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

FALSE

SELECT SDO_GEOM.RELATE(c_b.shape, 'anyinteract', c_a.shape, 0.005)
FROM university c_b, university c_a
WHERE c_b.name = 'pqr' AND c_a.name = 'abc';

SDO_GEOM.RELATE(C_B.SHAPE,'ANYINTERACT',C_A.SHAPE,0.005)
----------------------------------------------------------

TRUE
```

3) Return the areas of all different locations.

```
SELECT name, SDO_GEOM.SDO_AREA(shape, 0.005) FROM university;

NAME                           SDO_GEOM.SDO_AREA(SHAPE,0.005)
------------------------------ ------------------------------
abc                                                        24
pqr                                                      16.5
mno                                                         5
xyz                                                12.5663706
```

4) Return the area of just abc.

```
SELECT c.name, SDO_GEOM.SDO_AREA(c.shape, 0.005) FROM university c
WHERE c.name = 'abc';

NAME                           SDO_GEOM.SDO_AREA(C.SHAPE,0.005)
------------------------------ --------------------------------
abc                                                          24
```

5) Return the distance between two geometries.

```
SELECT SDO_GEOM.SDO_DISTANCE(c_b.shape, c_d.shape, 0.005)
FROM university c_b, university c_d
WHERE c_b.name = 'pqr' AND c_d.name = 'xyz';

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

6) Is a geometry valid?

```
SELECT c.name, SDO_GEOM.VALIDATE_GEOMETRY_WITH_CONTEXT(c.shape,
0.005)
FROM university c WHERE c.name = 'mno';

NAME
--------------------------------
SDO_GEOM.VALIDATE_GEOMETRY_WITH_CONTEXT(C.SHAPE,0.005)
--------------------------------------------------

mno
TRUE
```