

**S.I.E.S College of Arts, Science and Commerce (Autonomous) Sion(W), Mumbai – 400 022.**

**CERTIFICATE**

This is to certify that **Ms. Shreya Janardhan Shettigar** Roll No. **FMDS2122007** has successfully completed the necessary course of experiments in the subject of **Advanced Database Management Systems** during the academic year **2021 – 2022** complying with the requirements for thecourse of **M.Sc. Data Science Part – I [Semester 1]**

|  |  |
| --- | --- |
| Prof. In-Charge | Head of Department |
| **Dr. Abuzar Ansari** | **Prof. Manoj Singh** |

Examination Date: Examiner’s Signature & Date:

College Seal &

Date

**Index**

|  |  |  |
| --- | --- | --- |
| SR. NO. | TITLE | DATE |
| **1** | **Practical 1**  Distributed Database – Vertical Fragmentation | |
| **2** | **Practical 2**  Distributed Database – Horizontal Fragmentation | |
| **3** | **Practical 3**  Distributed Database – Replication | |
| **4** | **Practical 4**  Object Oriented Database – I | |
| **5** | **Practical 5**  Object Oriented Database – II | |
| **6** | **Practical 6**  Multimedia Database | |
| **7** | **Practical 7**  Temporal Database – I | |
| **8** | **Practical 8**  Temporal Database – I | |
| **9** | **Practical 9**  Active Database | |
| **10** | **Practical 10**  XML Database | |

# Practical 1: Vertical Fragmentation

**Aim:** To demonstrate Vertical Fragmentation in a Distributed Database Environment and implement it using Oracle Database

## Exercise A:

Create a global conceptual schema Emp ( Eno, Ename, Address, Email, Salary) and insert 10 records. Divide Emp into vertical fragments emp\_prac1 ( eno, ename, address) and emp\_prac1( 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 > 10000

## Code:

SQL> -- Practical 1A -- SQL> --ORCL 1

SQL> CONNECT scott@orcl1/tiger; Connected.

SQL> CREATE TABLE emp\_prac1(eno NUMBER(5), ename VARCHAR2(10), address VARCHAR2(10));

Table created.

SQL> INSERT INTO emp\_prac1 VALUES (1, 'Inbasagar', 'Antophill');

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (2, 'Ashwini', 'Powai');

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (3, 'Sonia', 'Dharavi');

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (4, 'Angelica', 'Sion');

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (5, 'Sanjeevani', 'Vashi');

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (6, 'Praveen', 'Dharavi');

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (7, 'Barthan', 'Dharavi');

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (8, 'Alvin', 'Thane');

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (9, 'Pranav', 'Bhandup');

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (10, 'Yash', 'Bhandup');

1 row created.

SQL> -- ORCL 2

SQL> CONNECT scott@orcl2/tiger; Connected.

SQL> CREATE TABLE emp\_prac1 (eno NUMBER(5), email VARCHAR2(20), salary VARCHAR2(10));

Table created.

SQL> INSERT INTO emp\_prac1 VALUES (1, 'Inbasagar@gmail.com', 99999);

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (2, 'Ashwini@gmail.com', 88888);

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (3, 'Sonia@yahoo.com', 99999);

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (4, 'Angelica@gmail.com', 86868);

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (5, 'Sanjeevani@gmail.com', 42042);

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (6, 'Praveen@gmail.com', 78945);

1 row created.

SQL> INSERT INTO emp\_prac1 VALUES (7, 'Barthan@yahoo.com', 65412);

|  |
| --- |
| 1 row created.  SQL> INSERT INTO emp\_prac1 VALUES (8, 'Alvin@yahoo.com', 78945);  1 row created.  SQL> INSERT INTO emp\_prac1 VALUES (9, 'Pranav@gmail.com', 32145);  1 row created.  SQL> INSERT INTO emp\_prac1 VALUES (10,'Yash@yahoo.com', 78241);  1 row created.  SQL>  SQL> --ORCL 1  SQL> CONNECT scott@orcl1/tiger; Connected.  SQL> -- Salary of employe where employee number is known SQL> SELECT salary FROM emp\_prac1@orcl2 WHERE eno = 1;  SALARY |
| 99999  SQL>  SQL> -- Email where employee name is known  SQL> SELECT email FROM emp\_prac1 e1, emp\_prac1@orcl2 e2 WHERE e1.eno = e2.eno AND e1.ename = 'Inbasagar';  EMAIL |
| [Inbasagar@gmail.com](mailto:Inbasagar@gmail.com)  SQL>  SQL> -- Employee name and email where employee number is known  SQL> SELECT ename, email FROM emp\_prac1 e1, emp\_prac1@orcl2 e2 WHERE e1.eno = e2.eno AND e1.eno = 1;  ENAME EMAIL |
| Inbasagar [Inbasagar@gmail.com](mailto:Inbasagar@gmail.com)  SQL>  SQL> -- Employee name whose salary is greater than 10000  SQL> SELECT ename FROM emp\_prac1 e1, emp\_prac1@orcl2 e2 WHERE e1.eno = e2.eno@orcl2 AND salary > 10000;  ENAME |
| Inbasagar Ashwini Sonia Angelica Sanjeevani Praveen Barthan Alvin Pranav Yash  10 rows selected.  SQL> |

**Exercise B:**

Create a global conceptual schema product\_master(product\_id, product\_name, product\_desc, cost, profit) and insert 10 records.Divide product\_master into vertical fragments product(product\_id, product\_name, product\_desc) and product@orcl2(product\_id, cost, profit) on two different nodes.

1. Display cost and profit of each product
2. Display product name where profit is less than Rs.20
3. Display product name, details where cost is between 200 to 500
4. Display product name beginning with ‘la’ and profit is 10% of product cost

## Code:

SQL> -- Practical 1B -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE TABLE product\_master(

1. pro\_id VARCHAR2(5),
2. pro\_name VARCHAR2(20),
3. pro\_desc VARCHAR2(50),
4. pro\_cost NUMBER(10),
5. pro\_profit NUMBER(10)
6. );

Table created. SQL>

SQL> CREATE TABLE product(

1. pro\_id VARCHAR2(5),
2. pro\_name VARCHAR2(20),
3. pro\_desc VARCHAR2(50)
4. );

Table created. SQL>

SQL> -- ORCL 2

SQL> CONNECT scott@orcl2/tiger Connected.

SQL> CREATE TABLE product(

1. pro\_id VARCHAR2(5),
2. pro\_cost NUMBER(10),
3. pro\_profit NUMBER(10)
4. );

Table created. SQL>

SQL> --ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE OR REPLACE TRIGGER product\_master\_insert

1. AFTER INSERT ON product\_master
2. FOR EACH ROW
3. BEGIN
4. INSERT INTO product VALUES(:NEW.pro\_id,:NEW.pro\_name,:NEW.pro\_desc);
5. INSERT INTO product@orcl2 VALUES(:NEW.pro\_id,:NEW.pro\_cost,:NEW.pro\_profit);
6. END;
7. /

Trigger created.

SQL> INSERT INTO product\_master VALUES('p1','lakme','face wash',10,1);

1 row created.

SQL> INSERT INTO product\_master VALUES('p2','garnier','conditioner',100,30);

1 row created.

SQL> INSERT INTO product\_master VALUES('p3','ravlon','shampoo',200,60);

1 row created.

SQL> INSERT INTO product\_master VALUES('p4','intel','processor',8000,1000);

1 row created.

SQL> INSERT INTO product\_master VALUES('p5','intel','mouse',500,50);

1 row created.

SQL> INSERT INTO product\_master VALUES('p6','lakme','foundation',200,30);

1 row created.

SQL> INSERT INTO product\_master VALUES('p7','livon','conditioner',100,10);

1 row created.

SQL> INSERT INTO product\_master VALUES('p8','aquaguard','water purifier',10000,1000);

1. row created.

SQL> INSERT INTO product\_master VALUES('p9','nokia','handset 6600',9000,2000);

1. row created.

SQL> INSERT INTO product\_master VALUES('p10','nokia','handset 6610',6000,1500);

1. row created.

#### SQL>

SQL> -- Cost and profit of each product

SQL> SELECT pro\_profit, pro\_cost FROM product@orcl2;

#### PRO\_PROFIT PRO\_COST

1 10

30 100

60 200

1000 8000

50 500

|  |  |
| --- | --- |
| 30 | 200 |
| 10 | 100 |
| 1000 | 10000 |
| 2000 | 9000 |
| 1500 | 6000 |

10 rows selected.

#### SQL>

SQL> -- Product name where profit is less than Rs 30

SQL> SELECT pro\_name FROM product p1, product@orcl2 p2 WHERE p1.pro\_id = p2.pro\_id AND pro\_profit < 30;

#### PRO\_NAME

lakme livon

#### SQL>

SQL> -- Product name and description where 200 <= cost <= 500

SQL> SELECT pro\_name, pro\_desc FROM product p1, product@orcl2 p2 WHERE p1.pro\_id = p2.pro\_id AND pro\_cost >= 200 AND pro\_cost <= 500;

#### PRO\_NAME PRO\_DESC

ravlon shampoo

intel mouse

lakme foundation

#### SQL>

SQL> -- Product name beginning with 'la' and profit is 10% of product cost

SQL> SELECT pro\_name, pro\_desc FROM product p1, product@orcl2 p2 WHERE p1.pro\_id = p2.pro\_id AND pro\_name LIKE 'la%' AND pro\_profit = pro\_cost \* 0.1;

#### PRO\_NAME PRO\_DESC

lakme face wash

SQL>

# Practical 2: Horizontal Fragmentation

**Aim:** To demonstrate Horizontal Fragmentation in a Distributed Database Environment and implement it using Oracle Database

## Exercise A:

Create a global conceptual schema emp\_master (eno, ename, address, email, salary) and insert 10 records. Divide emp\_master into horizontal fragments using the condition that emp\_lt\_10k contains tuples with salary < 10000 and emp\_ge\_10k with 10000 < salary < 20000 on two different nodes. Fire the following queries:

1. Find the salary of all employees
2. Find the Email of all employees where salary=15000
3. Find the employee name and Email where employee number is know
4. Find the employee name and address where employee number is known

## Code:

SQL> -- Practical 2A -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE TABLE emp\_master(emp\_no VARCHAR2(5), ename VARCHAR2(25), addr VARCHAR2(50), email VARCHAR2(25), sal NUMBER );

Table created.

SQL> CREATE TABLE emp\_lt\_10k(emp\_no VARCHAR2(5), ename VARCHAR2(25), addr VARCHAR2(50), email VARCHAR2(25), sal NUMBER );

Table created. SQL>

SQL> --ORCL 2

SQL> CONNECT scott@orcl2/tiger Connected.

SQL> CREATE TABLE emp\_ge\_10k(emp\_no VARCHAR2(5), ename VARCHAR2(25), addr VARCHAR2(50), email VARCHAR2(25), sal NUMBER );

Table created. SQL>

SQL> --ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE OR REPLACE TRIGGER emp\_master\_insert

1. AFTER INSERT ON emp\_master
2. FOR EACH ROW
3. WHEN(NEW.sal IS NOT NULL)
4. BEGIN
5. IF :NEW.sal<10000 THEN
6. INSERT INTO emp\_lt\_10k VALUES(:NEW.emp\_no,:NEW.ename,:NEW.addr,:NEW.email,:NEW.sal);
7. ELSE
8. INSERT INTO emp\_ge\_10k@orcl2 VALUES(:NEW.emp\_no, :NEW.ename, :NEW.addr, :NEW.email, :NEW.sal);
9. END IF;
10. END;
11. /

Trigger created. SQL>

|  |  |
| --- | --- |
| SQL> INSERT INTO emp\_master VALUES('e01', 'Inbasagar', 'Sion', 'inbasagar@gmail.com', 100000);  1 row created.  SQL> INSERT INTO emp\_master VALUES('e02', 'Chotu', 'Kurla', 'chotu@yahoo.com', 100);  1 row created.  SQL> INSERT INTO emp\_master VALUES('e03', 'Ambani', 'Sion', 'ambani@jio.com', 15000);  1 row created.  SQL>  SQL> -- Find salary of all employees  SQL> SELECT ename, sal FROM emp\_lt\_10k UNION ALL SELECT ename, sal FROM emp\_ge\_10k@orcl2;  ENAME SAL | |
| Chotu 100  Inbasagar 100000  Ambani 15000  SQL>  SQL> -- Find the email of all employees where salary = 15000  SQL> SELECT ename, email FROM emp\_lt\_10k WHERE sal = 15000 UNION ALL SELECT ename, email FROM emp\_ge\_10k@orcl2 WHERE sal = 15000;  ENAME EMAIL | |
| Ambani  SQL> -- OR  SQL> SELECT ename, email  ENAME | [ambani@jio.com](mailto:ambani@jio.com)  FROM emp\_ge\_10k@orcl2 WHERE sal = 15000; EMAIL |
| Ambani [ambani@jio.com](mailto:ambani@jio.com)  SQL>  SQL> -- Find the employee name and email where employee number is known  SQL> SELECT ename, email FROM emp\_lt\_10k WHERE emp\_no='e01' UNION ALL SELECT ename, email FROM emp\_ge\_10k@orcl2 WHERE emp\_no='e01';  ENAME EMAIL | |
| Inbasagar [inbasagar@gmail.com](mailto:inbasagar@gmail.com)  SQL>  SQL> -- Find the employee name and adddress where employee number is known  SQL> SELECT ename, addr FROM emp\_lt\_10k WHERE emp\_no='e01' UNION ALL SELECT ename, addr FROM emp\_ge\_10k@orcl2 WHERE emp\_no='e01';  ENAME ADDR | |
| Inbasagar  SQL> | Sion |

**Exercise B:**

Create a global conceptual schema cust\_pdtls (cust\_id, cust\_name, cust\_addr) and insert 10 records. Create two more schemas cust\_bill(cust\_id, cust\_mobile, cust\_billamt) and cust\_totbill(cust\_id, cust\_totalamt) on two different nodes. Fire the following queries:

1. List out the customer name operating more than 2 mobiles
2. Display the customer name where the total bill is greater than 2000
3. Display the total bill for all the customers
4. Display the customer name who is with us for the last 4 months

## Code:

SQL> -- Pracitical 2B -- SQL> -- ORCL1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE TABLE cust\_pdtls(cust\_id VARCHAR2(10) PRIMARY KEY, cust\_name VARCHAR2(20), cust\_addr VARCHAR2(20));

Table created. SQL>

SQL> -- ORCL2

SQL> CONNECT scott@orcl2/tiger Connected.

SQL> CREATE TABLE cust\_bill (cust\_id VARCHAR2(5), cust\_mobile NUMBER(10), cust\_billamt NUMBER(10)); Table created.

SQL>

SQL> -- ORCL3

SQL> CONNECT scott@orcl3/tiger Connected.

SQL> CREATE TABLE cust\_totbill(cust\_id VARCHAR2(10), cust\_totalamt NUMBER(20)); Table created.

SQL>

SQL> -- ORCL1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE OR REPLACE TRIGGER init\_totbill

1. AFTER INSERT ON cust\_pdtls
2. FOR EACH ROW
3. BEGIN
4. INSERT INTO cust\_totbill@orcl3 VALUES(:NEW.cust\_id, 0);
5. END;
6. /

Trigger created. SQL>

SQL> -- ORCL2

SQL> CONNECT scott@orcl2/tiger Connected.

SQL> CREATE OR REPLACE TRIGGER insert\_cust\_bill

1. AFTER INSERT ON cust\_bill
2. FOR EACH ROW
3. BEGIN
4. UPDATE cust\_totbill@orcl3 SET cust\_totalamt = cust\_totalamt + :NEW.cust\_billamt WHERE cust\_id =

:NEW.cust\_id;

1. END;
2. /

Trigger created.

#### SQL>

SQL> CREATE OR REPLACE TRIGGER update\_cust\_bill

1. AFTER UPDATE ON cust\_bill

#### FOR EACH ROW

1. BEGIN
2. UPDATE cust\_totbill@orcl3 SET cust\_totalamt = cust\_totalamt + :NEW.cust\_billamt -

:OLD.cust\_billamt WHERE cust\_id = :NEW.cust\_id;

#### END;

1. /

Trigger created. SQL>

SQL> CREATE OR REPLACE TRIGGER delete\_cust\_bill

1. AFTER DELETE ON cust\_bill

#### FOR EACH ROW

1. BEGIN
2. UPDATE cust\_totbill@orcl3 SET cust\_totalamt = cust\_totalamt - :OLD.cust\_billamt WHERE cust\_id =

:OLD.cust\_id;

#### END;

1. /

Trigger created. SQL>

#### SQL> -- ORCL1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> INSERT INTO cust\_pdtls VALUES ('c001', 'Inbasagar', 'Sion');

1 row created.

SQL> INSERT INTO cust\_pdtls VALUES ('c002', 'Dhanasekhar', 'Tuticorin');

1 row created.

SQL> INSERT INTO cust\_pdtls VALUES ('c003', 'Sudarshan', 'Titwala');

1 row created.

SQL> INSERT INTO cust\_pdtls VALUES ('c004', 'Ashwini', 'Powai');

1 row created. SQL>

SQL> INSERT INTO cust\_bill@orcl2 VALUES('c001', 9137100000, 1500);

1 row created.

SQL> INSERT INTO cust\_bill@orcl2 VALUES('c001', 9137100000, 1500);

1 row created.

SQL> INSERT INTO cust\_bill@orcl2 VALUES('c001', 9137100000, 555);

1 row created.

SQL> INSERT INTO cust\_bill@orcl2 VALUES('c001', 9137100000, 599);

1 row created.

SQL> INSERT INTO cust\_bill@orcl2 VALUES('c002', 9322700000, 499);

1 row created.

SQL> INSERT INTO cust\_bill@orcl2 VALUES('c002', 8097300000, 399);

1 row created.

SQL> INSERT INTO cust\_bill@orcl2 VALUES('c003', 7021900000, 599);

1 row created.

SQL> INSERT INTO cust\_bill@orcl2 VALUES('c003', 9962100000, 1500);

1 row created.

SQL> INSERT INTO cust\_bill@orcl2 VALUES('c004', 9619000000, 359);

1 row created.

#### SQL>

SQL> -- Customers with more than 2 mobile numbers

SQL> SELECT cust\_id "ID", COUNT(\*) "Number of devices" FROM (SELECT cust\_id FROM cust\_bill@orcl2 GROUP BY cust\_id, cust\_mobile) GROUP BY cust\_id HAVING COUNT(\*) >=2;

ID Number of devices

c002 2

c003 2

SQL>

SQL> -- Customer with total bill > 2000

SQL> SELECT cust\_name "Name", cust\_totalamt "Amount" FROM cust\_pdtls a, cust\_totbill@orcl3 b WHERE a.cust\_id = b.cust\_id AND cust\_totalamt > 2000;

Name

Amount

Inbasagar

Sudarshan

4154

2099

SQL>

SQL> -- Total bill for all the customers

SQL> SELECT cust\_name "Name", cust\_totalamt "Amount" FROM cust\_pdtls a, cust\_totbill@orcl3 b WHERE a.cust\_id = b.cust\_id;

SQL>

SQL> -- Customer who is with us for the last 4 months

SQL> SELECT a.cust\_name, b.months FROM cust\_pdtls a, (SELECT cust\_id, COUNT(\*) AS months FROM cust\_bill@orcl2 b GROUP BY(cust\_id)) b WHERE a.cust\_id = b.cust\_id;

CUST\_NAME

MONTHS

Dhanasekhar

Inbasagar Ashwini Sudarshan

2

4

1

2

SQL>

|  |  |
| --- | --- |
| Name | Amount |
| Inbasagar | 4154 |
| Dhanasekhar | 898 |
| Sudarshan | 2099 |
| Ashwini | 359 |

# Practical 3: Data Replication

**Aim:** To demonstrate Data Replicatoin in a Distributed Database Environment and implement it using Oracle Database

## Exercise:

Create a global conceptual schema emp(eno; ename; address; email; salary) and insert 10 records. Store the replication of emp in different node and fire the following queries:

1. Find the salary of all employees
2. Find the email of all employees where salary = 15000
3. Find the employee name and email where employee number is known
4. Find the employee name and address where employee number is known

**Code:**

SQL> -- Pracitical 3 -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE TABLE emp\_prac3(eno NUMBER(5), ename VARCHAR2(20), address VARCHAR2(25), email VARCHAR2(25), salary NUMBER(10));

Table created. SQL>

SQL> -- ORCL 2

SQL> CONNECT scott@orcl2/tiger Connected.

SQL> CREATE TABLE emp\_prac3(eno NUMBER(5), ename VARCHAR2(20), address VARCHAR2(25), email VARCHAR2(25), salary NUMBER(10));

Table created. SQL>

SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE OR REPLACE TRIGGER emp\_prac3\_insert

1. AFTER INSERT ON emp\_prac3
2. FOR EACH ROW
3. BEGIN
4. INSERT INTO emp\_prac3@orcl2 VALUES (:NEW.eno, :NEW.ename, :NEW.address, :NEW.email, :NEW.salary);
5. END;
6. /

Trigger created. SQL>

SQL> INSERT INTO emp\_prac3 VALUES(1,'Inbasagar','Antophill','inbasagar@gmail.com',22000);

1 row created.

SQL> INSERT INTO emp\_prac3 VALUES(2,'Sudarshan','Titwala','sudu1424@gmail.com',22000);

1 row created.

SQL> INSERT INTO emp\_prac3 VALUES(3,'Sonia','Dharavi','sonia@yahoo.com',20000);

1 row created.

SQL> INSERT INTO emp\_prac3 VALUES(4,'Angelica','Sion','angelica@gmail.com',20000);

1 row created.

SQL> INSERT INTO emp\_prac3 VALUES(5,'Ashwini','Powai','ashwini@yahoo.com',15000);

1 row created.

SQL> INSERT INTO emp\_prac3 VALUES(6,'Sanjeevani','Vashi','sanjeevani@yahoo.com',10000);

1 row created.

SQL>

SQL> -- Salary of all employees

SQL> SELECT ename, salary FROM emp\_prac3 UNION SELECT ename, salary FROM emp\_prac3@orcl2;

ENAME

SALARY

Angelica

Ashwini Inbasagar

20000

15000

22000

SQL> SELECT email FROM emp\_prac3 WHERE salary = 15000 UNION SELECT email FROM emp\_prac3@orcl2 WHERE

salary = 15000; EMAIL

[ashwini@yahoo.com](mailto:ashwini@yahoo.com)

SQL>

SQL> -- Employee name and email where employee number is known

SQL> SELECT ename, email FROM emp\_prac3 WHERE eno = 1 UNION SELECT ename, email FROM emp\_prac3@orcl2 WHERE eno = 1;

ENAME

EMAIL

Inbasagar

[inbasagar@gmail.com](mailto:inbasagar@gmail.com)

SQL>

SQL> -- Employee name and address where employee number is known

SQL> SELECT ename, address FROM emp\_prac3 WHERE eno = 1 UNION SELECT ename, address FROM emp\_prac3@orcl2 WHERE eno = 1;

ENAME

ADDRESS

Inbasagar

Antophill

SQL>

|  |  |
| --- | --- |
| Sanjeevani | 10000 |
| Sonia | 20000 |
| Sudarshan | 22000 |
| 6 rows selected. |  |
| SQL>  SQL> -- Email of all | employees where salary = 15000 |

# Practical 4: Object Oriented Database – I

**Aim:** To demonstrate Object Oriented Database and implement it using Oracle Database

## Exercise:

Using Object Oriented databases create the following types:

1. AddrType1 (Pincode: number, Street: char, City: char, State: char, No: number)
2. BranchType (address: AddrType1, phone1: integer, phone2: integer)
3. AuthorType (name: char, addr AddrType1)
4. PublisherType (name: char, addr: AddrType1, branches: BranchTableType)
5. AuthorListType as varray, which is reference to AuthorType Next create the following tables:
6. BranchTableType of BranchType
7. authors of AuthorType
8. books (title: varchar, year: date, published\_by ref PublisherType, authors AuthorListType)
9. Publishers of PublisherType

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

1. List all of the authors that have the same address as their publisher
2. List all the authors that have the same pin code as their publisher
3. List all books that have 2 or more authors
4. List the title of the book that has the most authors
5. List the name of the publisher that has the most branches
6. Name of authors who have not published more than a book.
7. all the branches that belong to the publisher 'tata' to the publisher „joshi'
8. List all the authors who have published more than one book
9. 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)

## Code:

SQL> -- Practical 4 -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL>

SQL> CREATE OR REPLACE TYPE AddrType AS OBJECT (pincode NUMBER(6), street CHAR(20), city VARCHAR2(50), state VARCHAR2(40), no NUMBER(4));

2 /

Type created. SQL>

SQL> CREATE OR REPLACE TYPE BranchType AS OBJECT (address AddrType, phone1 INTEGER, phone2 INTEGER);

2 /

Type created.

#### SQL>

SQL> CREATE OR REPLACE TYPE AuthorType AS OBJECT (name CHAR(20), addr AddrType);

2 /

Type created. SQL>

SQL> CREATE OR REPLACE TYPE BranchTableType AS TABLE OF BranchType;

2 /

Type created. SQL>

SQL> CREATE OR REPLACE TYPE PublisherType AS OBJECT (name CHAR(20), addr AddrType, branches BranchTableType);

2 /

Type created. SQL>

SQL> CREATE OR REPLACE TYPE AuthorListType AS VARRAY(10) OF REF AuthorType;

2 /

Type created. SQL>

SQL> CREATE TABLE authors OF AuthorType; Table created.

SQL> CREATE TABLE publishers OF PublisherType NESTED TABLE branches STORE AS branchtable; Table created.

SQL> CREATE TABLE books (title VARCHAR2(50), year DATE, published\_by REF PublisherType, authors AuthorListType);

Table created. SQL>

SQL> INSERT INTO authors VALUES('Rabiner', AddrType(5002,'sstreet','pune','mha',04));

1 row created.

SQL> INSERT INTO authors VALUES('Schiller', AddrType(7008,'nstreet','nasik','mha',08));

1 row created.

SQL> INSERT INTO authors VALUES('Tom', AddrType(7003,'dstreet','mumbai','mha',1003));

1 row created.

SQL> INSERT INTO authors VALUES('Paulraj', AddrType(7008,'sstreet','mumbai','mha',1007));

1 row created.

SQL> INSERT INTO authors VALUES('Elmasri', AddrType(7006,'nstreet','mumbai','mha',1006));

1 row created.

SQL> INSERT INTO authors VALUES('Ramakrishnan', AddrType(8002,'dstreet','pune','mha',1003));

1 row created.

SQL> INSERT INTO authors VALUES('Jerry', AddrType(8003,'dstreet','pune','mha',1003));

1 row created.

SQL> INSERT INTO authors VALUES('Jalebi', AddrType(8003,'dstreet','pune','mha',1003));

1 row created.

#### SQL>

SQL> INSERT INTO publishers VALUES('Pearson', AddrType(4002, 'rstreet', 'mumbai', 'mha', 03), BranchTableType(BranchType(AddrType(5002, 'fstreet', 'mumbai', 'mha', 03), 23406, 69896)));

1. row created.

SQL> INSERT INTO publishers VALUES('Joshi', AddrType(7008, 'sstreet', 'mumbai', 'mha', 1007), BranchTableType(BranchType(AddrType(1002, 'sstreet', 'nasik', 'mha', 1007), 456767, 7675757)));

1. row created.

SQL> INSERT INTO publishers VALUES('Wiley', AddrType(6002, 'sstreet', 'nasik', 'mha', 1007), BranchTableType(BranchType(AddrType(6002, 'sstreet', 'nasik', 'mha', 1007), 4543545, 8676775)));

|  |  |  |
| --- | --- | --- |
| 1. row created.   SQL> INSERT INTO publishers VALUES(   1. 'Ekta', 2. AddrType(7002,'sstreet','pune','mha',1007), 3. BranchTableType( 4. BranchType(AddrType(1002,'sstreet','pune','mha',1007),4543545,8676775), 5. BranchType(AddrType(7007,'sstreet','mumbai','mha',1007),4543545,8676775) 6. ) 7. );   1 row created.  SQL>  SQL> INSERT INTO books SELECT 'DSP', '28-may-1983', REF(pub) ,AuthorListType(REF(aut)) FROM publishers pub, authors aut WHERE pub.name='Joshi' AND aut.name='Elmasri';  1 row created.  SQL> INSERT INTO books SELECT 'compiler', '09-jan-1890', REF(pub), AuthorListType(REF(aut)) FROM publishers pub, authors aut WHERE pub.name='Wiley' AND aut.name='Jerry';  1 row created.  SQL> INSERT INTO books SELECT 'c', '25-may-1983', REF(pub), AuthorListType(REF(aut), REF(aut2)) FROM publishers pub, authors aut, authors aut2 WHERE pub.name='Pearson' AND aut.name='Paulraj' AND aut2.name='Schiller';  1 row created.  SQL> INSERT INTO books SELECT 'Algo', '25-may-1983', REF(pub), AuthorListType(REF(aut)) FROM publishers pub, authors aut WHERE pub.name='Pearson' AND aut.name='Paulraj';  1 row created.  SQL>  SQL> -- Select author name where publisher and author address are the same SQL> SELECT a.name from authors a, publishers p WHERE a.addr = p.addr;  NAME | | |
| Paulraj  SQL>  SQL> -- Select author name where publisher and author pincode are the same  SQL> SELECT a.name from authors a, publishers p WHERE a.addr.pincode = p.addr.pincode;  NAME | | |
| Schiller | |  |
| Paulraj | |  |
| SQL> | |  |
| SQL> -- Select books with 2 or more authors | |  |
| SQL> SELECT \* FROM books b where (SELECT COUNT(\*) FROM TABLE(b.authors)) >= 2; | |  |
| TITLE YEAR | |  |
| PUBLISHED\_BY |  |  |
| AUTHORS | |  |
| c 25-MAY-83 | |  |
| 00002202088F00331499504316B24BFF863537F49ADCE7C724926F4FBEAE97D8C059948781 | |  |
| AUTHORLISTTYPE(00002802098D55532C8CCB4BFFB75555C9AB1BD58EE2D2DD7245C347A89212994 | |  |
| 176A9BDDB010000EF0003, 00002802094F26B8F096AA4B4BAD77B582C6CF2FB5E2D2DD7245C347A | |  |
| 89212994176A9BDDB010000EF0001) | |  |
| SQL> | |  |
| SQL> -- Select books with max number of authors | |  |
| SQL> 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 |  |
| c  SQL>  SQL> -- Select name of publisher wih most branches  SQL> SELECT p.name FROM publishers p, TABLE(p.branches) GROUP BY p.name HAVING COUNT(\*) = (SELECT MAX(COUNT(\*)) FROM publishers p, TABLE(p.branches) GROUP BY p.name);  NAME | |
| Ekta  SQL>  SQL> -- Author with less than equal to 1 book  SQL> SELECT name FROM authors a WHERE ref(a) NOT IN (SELECT VALUE(aut) FROM books b, TABLE (b.authors) aut GROUP BY VALUE(aut) HAVING COUNT(\*) > 1);  NAME | |
| Rabiner Schiller Tom Elmasri  Ramakrishnan Jerry Jalebi  7 rows selected.  SQL>  SQL> -- All the branches that belong to the publisher tata and joshi  SQL> SELECT p.name, p.branches FROM publishers p, TABLE(p.branches) WHERE p.name = 'Ekta' OR p.name = 'Joshi';  NAME | |
| BRANCHES(ADDRESS(PINCODE, STREET, CITY, STATE, NO), PHONE1, PHONE2) | |
| Joshi  BRANCHTABLETYPE(BRANCHTYPE(ADDRTYPE(1002, 'sstreet ', 'nasik', 'mha'  , 1007), 456767, 7675757))  Ekta  BRANCHTABLETYPE(BRANCHTYPE(ADDRTYPE(1002, 'sstreet ', 'pune', 'mha', 1007), 4543545, 8676775), BRANCHTYPE(ADDRTYPE(7007, 'sstreet ', 'mu  mbai', 'mha', 1007), 4543545, 8676775))  Ekta  BRANCHTABLETYPE(BRANCHTYPE(ADDRTYPE(1002, 'sstreet ', 'pune', 'mha', 1007), 4543545, 8676775), BRANCHTYPE(ADDRTYPE(7007, 'sstreet ', 'mu  mbai', 'mha', 1007), 4543545, 8676775))  SQL>  SQL> -- Author with more than one book  SQL> SELECT DEREF(VALUE(aut)).name FROM books b, TABLE(b.authors) aut GROUP BY VALUE(aut) HAVING COUNT (\*) > 1;  DEREF(VALUE(AUT)).NA | |
| Paulraj  SQL>  SQL> -- Select book with repeated authors  SQL> INSERT INTO books SELECT 'ERR','25-may-1983',REF(pub),AuthorListType(REF(aut), REF(aut2)) from publishers pub, authors aut, authors aut2 where pub.name='Pearson' and aut.name='Schiller' and aut2.name='Schiller'; | |

1 row created.

SQL> SELECT b.title FROM books b, TABLE(b.authors) aut GROUP BY VALUE(aut), b.title HAVING COUNT(\*) > 1;

TITLE

ERR SQL>

# Practical 5: Object Oriented Database – II

**Aim:** To demonstrate Object Oriented Database and implement it using Oracle Database

## Exercise:

Using Object Oriented databases, create the following types:

* 1. state61(st\_code: number, st\_name: varchar2, st\_district: varchar2, st\_pincode: number)
  2. contact\_detail61(residence\_no: number, office\_no: number, email: varchar2, fax: number, mobile: number)
  3. address61(road\_no: varchar2, road\_name: varchar2, landmark:varchar, state: state61, contact: contact\_detail61)
  4. staff61(staff\_id: number, staff\_name: varchar2, staff\_address: address61, staff\_deptno: number, staff\_sal: number, staff\_other: varchar2, dob: date) define method getAge() to calculate age using dob
  5. dept61(dept\_id: number, location: varchar2, dept\_name: varchar2,emp: staffTableType)

Next create the following tables:

1. staffTableType of staff61
2. dpt\_refernce of dept61 with nested relation (emp) Insert records into the above tables and fire the following queries:
3. Display staff ID and department name of all employees.
4. How many workers are in particular department
5. Find department name for particular staff name
6. Display department-wise report
7. Display age and birth date of particular employee

## Code:

SQL> -- Practical 5 -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE OR REPLACE TYPE state61 AS OBJECT (st\_code NUMBER, st\_name VARCHAR2(25), st\_district VARCHAR2(25), st\_pincode NUMBER);

2 /

Type created.

SQL> CREATE OR REPLACE TYPE contact\_detail61 AS OBJECT (residence\_no NUMBER, office\_no NUMBER, email VARCHAR2(50), fax NUMBER, mobile NUMBER);

2 /

Type created.

SQL> CREATE OR REPLACE TYPE address61 AS OBJECT (road\_no VARCHAR2(10), road\_name VARCHAR2(50),

landmark VARCHAR2(50), state state61, contact contact\_detail61);

2 /

Type created.

SQL> CREATE OR REPLACE TYPE staff61 AS OBJECT (staff\_id NUMBER, staff\_name VARCHAR2(25), staff\_address address61, staff\_deptno NUMBER, staff\_sal NUMBER, staff\_other VARCHAR2(50), dob DATE, MEMBER FUNCTION getAge RETURN NUMBER);

2 /

Type created.

SQL> CREATE OR REPLACE TYPE BODY staff61 as MEMBER FUNCTION getAge RETURN NUMBER AS

#### BEGIN

1. RETURN TRUNC(MONTHS\_BETWEEN(sysdate,dob)/12);
2. END getAge;

#### END;

1. /

Type body created.

SQL> CREATE OR REPLACE TYPE staff\_table\_type AS TABLE OF staff61;

2 /

Type created.

SQL> CREATE OR REPLACE TYPE dept61 AS OBJECT (dept\_id NUMBER, location VARCHAR2(20), dept\_name VARCHAR2(20), emp staff\_table\_type);

2 /

Type created.

SQL> CREATE TABLE dpt\_reference OF dept61 NESTED TABLE emp STORE AS NTrelation; Table created.

SQL> INSERT INTO dpt\_reference VALUES (

2 1,

1. 'Mumbai',
2. 'Programming',
3. staff\_table\_type(
4. staff61(
5. 1,
6. 'Inbasagar',
7. address61(
8. '21',
9. 'SM Road',
10. 'Near Antophill Church',
11. state61(1, 'Maharashtra', 'Mumbai', 400037),
12. contact\_detail61(9137000000, 9137111111, 'm.inbasagarnadar@gmail.com', 9137222222, 9137333333)
13. ),
14. 1,
15. 100000,
16. 'Senior',
17. '10-sep-1999'
18. ),
19. staff61(
20. 2,
21. 'Azoni',
22. address61(
23. '11',
24. '90ft Road',
25. 'Near Kamaraj School',
26. state61(1, 'Maharashtra', 'Mumbai', 400022),
27. contact\_detail61(9930000000, 9930111111, 'azoni@gmail.com', 9930222222, 9930333333)
28. ),
29. 1,
30. 100000,
31. 'Senior',

34 '7-may-1996'

35 )

36 )

37 );

1 row created.

#### SQL>

SQL> INSERT INTO dpt\_reference VALUES (

2 2,

1. 'Mumbai',
2. 'Marketing',
3. staff\_table\_type(
4. staff61(
5. 3,
6. 'Kajal',
7. address61(
8. '7',
9. '7th street',
10. 'Near Milan Subway',
11. state61(1, 'Maharashtra', 'Mumbai', 400054),
12. contact\_detail61(9167000000, 9167111111, 'kajal@gmail.com', 9167222222, 9167333333)
13. ),
14. 1,
15. 65000,
16. 'PR Head',

19 '22-sep-1999'

20 ),

1. staff61(
2. 4,
3. 'Ashwini',
4. address61(
5. '78',
6. 'Padmavati Devi Marg',
7. 'Near Travancore cafe',
8. state61(1, 'Maharashtra', 'Mumbai', 400076),
9. contact\_detail61(9619000000, 9619111111, 'ashanilkumar@gmail.com', 9619222222, 9619333333)
10. ),
11. 1,
12. 70000,
13. 'Social Media Head',
14. '11-nov-1999'
15. )
16. )
17. );

1 row created.

#### SQL> SQL>

SQL> INSERT INTO dpt\_reference VALUES (

2 3,

1. 'Punjab',
2. 'Sponsorship',
3. staff\_table\_type(
4. staff61(
5. 5,
6. 'Annie',
7. address61(
8. '9',
9. '9th street',
10. 'Opp to gurudwara',
11. state61(2, 'Punjab', 'Chandigarh', 160001),
12. contact\_detail61(8879000000, 8879111111, 'annie@gmail.com', 8879222222, 8879333333)
13. ),
14. 1,
15. 65000,
16. 'Head',
17. '30-nov-1999'
18. )
19. )
20. );

1 row created.

SQL>

SQL> -- Display staff ID and department name of all employees

SQL> SELECT d.dept\_name, e.staff\_id FROM dpt\_reference d, TABLE(d.emp) e;

DEPT\_NAME

STAFF\_ID

SQL>

SQL> -- How many workers are in a particular department

SQL> SELECT d.dept\_id, d.dept\_name, COUNT(e.staff\_id) as employee\_count FROM dpt\_reference d, TABLE(d.emp) e WHERE d.dept\_id=1 GROUP BY d.dept\_id, d.dept\_name;

DEPT\_ID DEPT\_NAME

EMPLOYEE\_COUNT

1 Programming

2

SQL>

SQL> -- Find department name for a particular staff name

SQL> SELECT d.dept\_id, d.dept\_name FROM dpt\_reference d, TABLE(d.emp) e WHERE e.staff\_name = 'Inbasagar';

DEPT\_ID DEPT\_NAME

1 Programming

SQL>

SQL> -- Display department-wise report

SQL> SELECT d.dept\_id, d.dept\_name, COUNT(\*) as employee\_count, SUM(e.staff\_sal) as total\_salary FROM dpt\_reference d, TABLE(d.emp) e GROUP BY d.dept\_id, d.dept\_name;

DEPT\_ID DEPT\_NAME

EMPLOYEE\_COUNT TOTAL\_SALARY

SQL>

SQL> -- Display age and dob of a particualr employee

SQL> SELECT e.staff\_name, e.getAge() as age, e.dob FROM dpt\_reference d, TABLE(d.emp) e WHERE e.staff\_id = 1;

STAFF\_NAME

AGE DOB

Inbasagar

21 10-SEP-99

SQL>

|  |  |
| --- | --- |
| Programming | 1 |
| Programming | 2 |
| Marketing | 3 |
| Marketing | 4 |
| Sponsorship | 5 |

|  |  |  |
| --- | --- | --- |
| 1 Programming | 2 | 200000 |
| 3 Sponsorship | 1 | 65000 |
| 2 Marketing | 2 | 135000 |

# Practical 6: Multimedia Database

**Aim:** To demonstrate the use of Multimedia Database and implement it using Oracle Database

## Exercise:

Create a table employee with attributes employee\_id, first\_name, last\_name, email, hire\_date, job\_id, salary, resume as clob and picture as blob to insert an employee‟s picture. Fire the following queries:

* 1. For appending data into clob datatype
  2. Selecting CLOB values by Using SQL
  3. Use of SUBSTR() and INSTR() functions
  4. Use of OUTPUT.PUT\_LINE()
  5. Removing LOBs

**Code:**

SQL> -- Practical 6 -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE TABLE employee\_prac6 (employee\_id NUMBER, first\_name VARCHAR2(25), last\_name VARCHAR2(25), email VARCHAR(25), hire\_date DATE, job\_id VARCHAR(20), salary NUMBER, resume CLOB, picture BLOB);

Table created.

SQL> INSERT INTO employee\_prac6 VALUES (1, 'Inbasagar', 'Nadar', 'inbasagarnadar@gmail.com', '10-sep- 2015' , 'Project Manager', 100000, EMPTY\_CLOB(), EMPTY\_BLOB());

1 row created.

SQL> UPDATE employee\_prac6 SET resume='Date of Birth: 10 Sep 1999' WHERE employee\_id = 1;

1 row updated.

SQL> INSERT INTO employee\_prac6 VALUES (2, 'Sudarshan', 'Jadhav', 'sudarshan@gmail.com', '14-nov-2015'

, 'Photography', 90000, EMPTY\_CLOB(), EMPTY\_BLOB());

1 row created.

SQL> UPDATE employee\_prac6 SET resume='Date of Birth: 14 Nov 1999' WHERE employee\_id = 2;

1 row updated.

SQL> INSERT INTO employee\_prac6 VALUES (3, 'Alvin', 'Nadar', 'alvin@gmail.com', '5-jan-2015' , 'Finance', 80000, EMPTY\_CLOB(), EMPTY\_BLOB());

1 row created.

SQL> UPDATE employee\_prac6 SET resume='Date of Birth: 05 Jan 1999' WHERE employee\_id = 3;

1 row updated.

SQL> INSERT INTO employee\_prac6 VALUES (4, 'Ashwini', 'Anilkumar', 'ashwini@gmail.com', '11-nov-2015'

, 'PR', 80000, EMPTY\_CLOB(), EMPTY\_BLOB());

1 row created.

SQL> UPDATE employee\_prac6 SET resume='Date of Birth: 11 Nov 1999' WHERE employee\_id = 4;

1 row updated.

SQL> INSERT INTO employee\_prac6 VALUES (5, 'Sonia', 'Saravanan', 'sonia@gmail.com', '1-sep-2015' , 'Marketing', 80000, EMPTY\_CLOB(), EMPTY\_BLOB());

1. row created.

SQL> UPDATE employee\_prac6 SET resume='Date of Birth: 01 Sep 1999' WHERE employee\_id = 5;

1. row updated.

SQL>

SQL> -- Append data to clob

SQL> -- appending using DBMS\_LOB.WRITE() SQL> DECLARE

1. resume\_lob CLOB;
2. new\_text VARCHAR2(100):='Resigned: 5 Aug 2020';
3. amount NUMBER;
4. offset NUMBER;

|  |  |  |
| --- | --- | --- |
| 1. BEGIN 2. SELECT resume INTO resume\_lob FROM employee\_prac6 WHERE employee\_id = 3 FOR UPDATE; 3. offset:= DBMS\_LOB.GETLENGTH(resume\_lob) + 2; --Date of Birth: 10 sep 1999 4. amount:= LENGTH(new\_text); 5. DBMS\_LOB.WRITE(resume\_lob, amount, offset, new\_text); 6. COMMIT; 7. END; 8. /   PL/SQL procedure successfully completed. SQL>  SQL> SELECT resume FROM employee\_prac6 WHERE employee\_id = 3;  RESUME | | |
| Date of Birth: 05 Jan 1999 Resigned: 5 Aug 2020  SQL>  SQL> -- appending using DBMS\_LOB.WRITEAPPEND() SQL> DECLARE   1. resume\_lob CLOB; 2. new\_text VARCHAR2(100):=' Resigned: 30 November 2020'; 3. amount NUMBER; 4. BEGIN 5. SELECT resume INTO resume\_lob FROM employee\_prac6 WHERE employee\_id = 2 FOR UPDATE; 6. amount:=DBMS\_LOB.GETLENGTH(resume\_lob); 7. DBMS\_LOB.WRITEAPPEND(resume\_lob, amount, new\_text); 8. COMMIT; 9. END; 10. /   PL/SQL procedure successfully completed.  SQL> SELECT resume FROM employee\_prac6 WHERE employee\_id = 2; RESUME | | |
| Date of Birth: 14 Nov 1999 Resigned: 30 November 202  SQL>  SQL> -- Selecting CLOB Values using sql  SQL> SELECT first\_name, resume FROM employee\_prac6;  FIRST\_NAME | |  |
| RESUME |  |
| Inbasagar  Date of Birth: 10 Sep 1999  Sudarshan  Date of Birth: 14 Nov 1999 Resigned: 30 November 202  Alvin  Date of Birth: 05 Jan 1999 Resigned: 5 Aug 2020  Ashwini  Date of Birth: 11 Nov 1999  Sonia  Date of Birth: 01 Sep 1999  SQL>  SQL> -- Use of SUBSTR and INSTR  SQL> SELECT DBMS\_LOB.SUBSTR(resume, 10, 15) "Substring", DBMS\_LOB.INSTR(resume,'Resigned') "Instring" FROM employee\_prac6; | | |

|  |
| --- |
| Substring |
| Instring |
| 10 Sep 19  0  14 Nov 19  28  05 Jan 19  28  11 Nov 19  0  01 Sep 19  0  SQL>  SQL> -- Use of DBMS\_OUTPUT.PUT\_LINE SQL> SET SERVEROUTPUT ON  SQL>  SQL> DECLARE   1. output\_text VARCHAR(100); 2. BEGIN 3. SELECT resume INTO output\_text FROM employee\_prac6 WHERE employee\_id = 1; 4. DBMS\_OUTPUT.PUT\_LINE('Resume Content: '||output\_text); 5. END; 6. /   Resume Content: Date of Birth: 10 Sep 1999  PL/SQL procedure successfully completed. SQL>  SQL> DECLARE   1. output\_text VARCHAR(100); 2. resume\_lob CLOB; 3. amount NUMBER:=100; 4. offset NUMBER:=1; 5. BEGIN 6. SELECT resume INTO resume\_lob FROM employee\_prac6 WHERE employee\_id = 1; 7. DBMS\_LOB.READ(resume\_lob, amount, offset, output\_text); 8. DBMS\_OUTPUT.PUT\_LINE('Resume Content: '||output\_text); 9. END; 10. /   Resume Content: Date of Birth: 10 Sep 1999  PL/SQL procedure successfully completed. SQL>  SQL> -- Removing LOB  SQL> UPDATE employee\_prac6 SET resume = EMPTY\_CLOB() WHERE employee\_id = 3;  1 row updated.  SQL> SELECT first\_name, resume FROM employee\_prac6 WHERE employee\_id = 3; FIRST\_NAME |
| RESUME |
| Alvin  SQL> |

# Practical 7: Temporal Database – I

**Aim:** To demonstrate the use of Temporal Database and implement it using Oracle Database

## Exercise A:

Create a table empant\_prac7a, which stores the account number,name, and valid time say, recruitment data retirement date. Insert couple of records and fire the following queries:

1. Find all the employees who join the company on 2/3/2001
2. Find all the employees who will retired on 2/3/2001

## Code:

|  |  |
| --- | --- |
| SQL> -- Practical 7A -- |  |
| SQL> -- ORCL 1 |  |
| SQL> CONNECT scott@orcl1/tiger | |
| Connected. |  |
| SQL> CREATE TABLE empant\_prac7a (acc\_no NUMBER(10), aname VARCHAR2(20), join\_date DATE, retire\_date | |
| DATE); |  |
| Table created. |  |
| SQL> |  |
| SQL> INSERT INTO empant\_prac7a VALUES(101, 'Inbasagar', '10-Sep-1999', '21-Mar-2004'); | |
| 1 row created. |  |
| SQL> INSERT INTO empant\_prac7a VALUES(102, 'Dhanasekhar', '10-Sep-1999', '10-May-2005'); | |
| 1 row created. |  |
| SQL> INSERT INTO empant\_prac7a VALUES(103, 'Ashwini', '11-Nov-1999', '10-June-2002'); | |
| 1 row created. |  |
| SQL> INSERT INTO empant\_prac7a VALUES(104, 'Sudarhsan', '14-Nov-1999', '14-May-2006'); | |
| 1 row created. |  |
| SQL> INSERT INTO empant\_prac7a VALUES(105, 'Praveen', '10-Aug-1995', '9-Aug-2005'); | |
| 1 row created. |  |
| SQL> INSERT INTO empant\_prac7a VALUES(107, 'Sonia', '01-Sep-1999', '20-Mar-2005'); | |
| 1 row created. |  |
| SQL> INSERT INTO empant\_prac7a VALUES(108, 'Angelica', '17-Nov-1999', '14-May-2006'); | |
| 1 row created. |  |
| SQL> INSERT INTO empant\_prac7a VALUES(109, 'Barathan', '12-Sep-2000', '14-Apr-2007'); | |
| 1 row created. |  |
| SQL> INSERT INTO empant\_prac7a VALUES(110, 'Soumya', '28-Sep-1999', '17-May-2004'); | |
| 1 row created. |  |
| SQL> |  |
| SQL> -- Employees who join the company on 10 September 1999 | |
| SQL> SELECT \* FROM empant\_prac7a WHERE join\_date = '10-Sep-1999'; | |
| ACC\_NO ANAME JOIN\_DATE RETIRE\_DA | |
| 101 Inbasagar 10-SEP-99 21-MAR-04 | |
| 102 Dhanasekhar 10-SEP-99 10-MAY-05 | |
| SQL> |  |
| SQL> -- Employees who retired on 14 May 2006 | |
| SQL> SELECT \* FROM empant\_prac7a WHERE retire\_date = '14-May-2006'; | |
| ACC\_NO ANAME JOIN\_DATE RETIRE\_DA | |
| 104 Sudarhsan  108 Angelica | 14-NOV-99 14-MAY-06  17-NOV-99 14-MAY-06 |
| SQL> |

**Exercise B:**

Create a table shares, which stores the, name of company, number of shares, and price per share at transaction time. Insert couple of records and fire the following queries:

1. Find the names of companies whose share price is more than Rs. 100 at 5:30 PM
2. Find the name of the company which has highest shares price at 5.00 PM

**Code:**

|  |  |
| --- | --- |
| SQL> -- Practical 7B -- SQL> -- ORCL 1  SQL> CONNECT scott@orcl1/tiger Connected.  SQL> CREATE TABLE shares\_prac7b (cname VARCHAR2(20), nofshares NUMBER(5), pricepshare NUMBER(5), transtime TIMESTAMP(6));  Table created. SQL>  SQL> INSERT INTO shares\_prac7b VALUES('Reliance Infocom',250,25,SYSTIMESTAMP);  1 row created.  SQL> INSERT INTO shares\_prac7b VALUES('Tata',205,20,'05-jun-04 11.45.00.000000 am');  1 row created.  SQL> INSERT INTO shares\_prac7b VALUES('Wipro',250,25,'10-mar-03 06.15.00.000000 pm');  1 row created.  SQL> INSERT INTO shares\_prac7b VALUES('Patni',115,15,'08-may-01 07.25.00.000000 am');  1 row created.  SQL> INSERT INTO shares\_prac7b VALUES('TCS',140,12,'14-apr-05 05.30.00.000000 pm');  1 row created.  SQL> INSERT INTO shares\_prac7b VALUES('Google',310,30,'12-sep-03 10.30.00.000000 am');  1 row created.  SQL> INSERT INTO shares\_prac7b VALUES('Hero Honda',100,250,'21-aug-04 05.30.00.000000 pm');  1 row created.  SQL>  SQL> -- Companies with share price > 100 at 5:30PM  SQL> SELECT cname FROM shares\_prac7b WHERE pricepshare > 100 AND TO\_CHAR(transtime,'HH12:MIAM')='05:30PM';  CNAME | |
| Hero Honda  SQL>  SQL> -- Company with highest share price at 5:30PM  SQL> SELECT cname FROM shares\_prac7b WHERE pricepshare IN (SELECT MAX(pricepshare) FROM shares\_prac7b WHERE TO\_CHAR(transtime,'HH12:MIAM')='05:30PM');  CNAME | |
| Hero Honda  SQL> |  |

# Practical 8: Temporal Database – II

**Aim:** To demonstrate the use of Temporal Database and implement it using Oracle Database

## Exercise:

Create a table employee which stores the employee number, employee name, email, address and salary. Create a table log\_employee which stores employee number, old salary, updated salary and date.

Create the following triggers :

1. On insert of an employee record in the employee table, the corresponding values must be entered in the log\_employee table
2. On update of any record in the employee table, the corresponding record must be entered in the log\_employee table

Insert 10 records and fire the following queries:

1. Display the latest salary of all the employees
2. Display employee name that has got more than 2 user events
3. Display employee name that has got an increment of 5000 in one increment
4. Display employee name and salary of all the employees at second increment
5. Display employee name, total salary and total increment

## Code:

SQL> -- Practical 8 -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE TABLE employee\_prac8 (emp\_no VARCHAR2(5), ename VARCHAR2(25), addr VARCHAR2(50), email VARCHAR2(50), sal NUMBER);

Table created. SQL>

SQL> CREATE TABLE employee\_update\_prac8 (emp\_no VARCHAR2(5), old\_sal NUMBER, up\_sal NUMBER, e\_date

DATE);

Table created. SQL>

SQL>

SQL> CREATE OR REPLACE TRIGGER employee\_prac8\_insert

1. AFTER INSERT ON employee\_prac8
2. FOR EACH ROW
3. BEGIN
4. INSERT INTO employee\_update\_prac8 VALUES(:NEW.emp\_no,NULL,:NEW.sal,SYSDATE);
5. END;
6. /

Trigger created. SQL>

SQL> CREATE OR REPLACE TRIGGER employee\_prac8\_update

1. AFTER UPDATE ON employee\_prac8
2. FOR EACH ROW
3. BEGIN
4. INSERT INTO employee\_update\_prac8 VALUES(:NEW.emp\_no,:OLD.sal,:NEW.sal,SYSDATE);

#### 6 END;

7 /

Trigger created. SQL>

SQL> INSERT INTO employee\_prac8 VALUES('a01', 'Inbasagar','Sion','inbasagar@gmail.com',15000);

1 row created.

SQL> INSERT INTO employee\_prac8 VALUES('a02', 'Dhanasekhar','Tuticorin','dhanasekhar@gmail.com',8900);

1 row created.

SQL> INSERT INTO employee\_prac8 VALUES('a03', 'Sudarhsan','Titwala','sudu@gmail.com',15000);

1 row created.

SQL> INSERT INTO employee\_prac8 VALUES('a04', 'Ashwini','Powai','ashwini@gmail.com',16000);

1 row created.

SQL> INSERT INTO employee\_prac8 VALUES('a05', 'Sonia','Dharavi','sonia@gmail.com',12000);

1 row created.

SQL> INSERT INTO employee\_prac8 VALUES('a06', 'Angelica','Sion','angelica@gmail.co.in',11500);

1 row created.

SQL> INSERT INTO employee\_prac8 VALUES('a07', 'Sanjeevani','Vashi','sanjeevani@gmail.com',10900);

1 row created.

SQL> INSERT INTO employee\_prac8 VALUES('a08', 'Praveen','Dharavi','praveen@gmail.com',19000);

1 row created.

SQL> INSERT INTO employee\_prac8 VALUES('a09', 'Balaji','Dharavi','balaji@gmail.co.in',7000);

1 row created.

SQL> INSERT INTO employee\_prac8 VALUES('a10', 'Barathan','Dharavi','barathan@gmail.com',16500);

1 row created.

#### SQL> SQL>

SQL> UPDATE employee\_prac8 SET sal=20000 WHERE emp\_no='a01';

1 row updated.

SQL> UPDATE employee\_prac8 SET sal=22000 WHERE emp\_no='a01';

1 row updated.

SQL> UPDATE employee\_prac8 SET sal=25000 WHERE emp\_no='a01';

1 row updated.

SQL> UPDATE employee\_prac8 SET sal=10000 WHERE emp\_no='a02';

1. row updated.

SQL> UPDATE employee\_prac8 SET sal=12000 WHERE emp\_no='a02';

1. row updated.

#### SQL>

SQL> -- Latest salaray of all employees SQL> SELECT ename, sal FROM employee\_prac8;

|  |  |
| --- | --- |
| ENAME | SAL |
| Inbasagar | 25000 |
| Dhanasekhar | 12000 |
| Sudarhsan | 15000 |
| Ashwini | 16000 |
| Sonia | 12000 |
| Angelica | 11500 |
| Sanjeevani | 10900 |
| Praveen | 19000 |
| Balaji | 7000 |
| Barathan | 16500 |
| 10 rows selected. |  |

#### SQL>

SQL> -- Employee name where more than 2 user events

SQL> SELECT e.ename FROM employee\_prac8 e, employee\_update\_prac8 e\_u WHERE e.emp\_no = e\_u.emp\_no GROUP BY e\_u.emp\_no, e.ename HAVING COUNT (\*) > 2;

#### ENAME

Inbasagar Dhanasekhar

SQL>

SQL> -- Employee name that has hot an increment of 5000 in one increment;

SQL> SELECT ename FROM employee\_prac8 e, employee\_update\_prac8 e\_u WHERE e.emp\_no = e\_u.emp\_no AND (e\_u.up\_sal - e\_u.old\_sal) = 5000;

ENAME

Inbasagar SQL>

SQL> -- Employee name and salary of all the employees at second increment (3 because 1 record for

insert and 2 for update)

SQL> SELECT ename, sal FROM employee\_prac8 e, employee\_update\_prac8 e\_u WHERE e.emp\_no = e\_u.emp\_no GROUP BY e\_u.emp\_no, e.ename, e.sal HAVING COUNT (\*) = 3;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ENAME | SAL | | | |
| Dhanasekhar | 12000 | | | |
| SQL>  SQL> -- Employee name and total salary and total increment  SQL> SELECT ename, MIN(up\_sal) "Initial", MAX(up\_sal) "Latest", MAX(up\_sal)-MIN(up\_sal)"Increment", COUNT(\*)-1 "Increments" FROM employee\_prac8 e, employee\_update\_prac8 e\_u WHERE e.emp\_no = e\_u.emp\_no GROUP BY e\_u.emp\_no, e.ename; | | | | |
| ENAME | Initial | Latest | Increment | Increments |
| Inbasagar | 15000 | 25000 | 10000 | 3 |
| Praveen | 19000 | 19000 | 0 | 0 |
| Sudarhsan | 15000 | 15000 | 0 | 0 |
| Ashwini | 16000 | 16000 | 0 | 0 |
| Sanjeevani | 10900 | 10900 | 0 | 0 |
| Barathan | 16500 | 16500 | 0 | 0 |
| Balaji | 7000 | 7000 | 0 | 0 |
| Dhanasekhar | 8900 | 12000 | 3100 | 2 |
| Angelica | 11500 | 11500 | 0 | 0 |
| Sonia | 12000 | 12000 | 0 | 0 |
| 10 rows selected. |  |  |  |  |
| SQL> |  |  |  |  |

# Practical 9: Active Database

**Aim:** To demonstrate the use of Active Database and implement it using Oracle Database

## Exercise A:

Create table emp (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:

* 1. Creating a trigger to insert new employee tuple and display the new total hours from project table
  2. Creating a trigger to change the hrs of existing employee and display the new total hours from project table
  3. Creating a trigger to change the project of an employee and display the new total hours from project table
  4. Creating a trigger to delete the project of an employee

## Code:

SQL> -- Practical 9A -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE TABLE emp\_prac9a (eno NUMBER(8) PRIMARY KEY, ename VARCHAR(20), hrs NUMBER, pno NUMBER,

super\_no NUMBER CONSTRAINT sup UNIQUE); Table created.

SQL>

SQL> CREATE TABLE project\_prac9a (pno NUMBER PRIMARY KEY, pname VARCHAR(20), thrs NUMBER (8), super\_no NUMBER CONSTRAINT supfk REFERENCES emp\_prac9a(super\_no));

Table created. SQL>

SQL> INSERT INTO emp\_prac9a VALUES(1, 'Inbasagar', 7, 10, 1001);

1 row created.

SQL> INSERT INTO emp\_prac9a VALUES(2, 'Dhanasekhar', 5, 20, 1002);

1 row created.

SQL> INSERT INTO emp\_prac9a VALUES(3, 'Sudarhsan', 3, 10, 1003);

1 row created.

SQL> INSERT INTO emp\_prac9a VALUES(4, 'Ashwini', 1, 20, 1004);

1 row created.

SQL> INSERT INTO emp\_prac9a VALUES(5, 'Annie', 5, 30, 1005);

1 row created.

SQL> INSERT INTO emp\_prac9a VALUES(6, 'Praveen', 8, 40, 1006);

1 row created.

SQL> INSERT INTO emp\_prac9a VALUES(7, 'Sonia', 3, 30, 1007);

1 row created.

SQL> INSERT INTO emp\_prac9a VALUES(8, 'Angelica', 12, 40, 1008);

1 row created.

SQL>

SQL> INSERT INTO project\_prac9a VALUES(10, 'Hyperloop', 10, 1001);

1 row created.

SQL> INSERT INTO project\_prac9a VALUES(20, 'Metrorail', 15, 1002);

1 row created.

SQL> INSERT INTO project\_prac9a VALUES(30, 'Monorail', 8, 1005);

1 row created.

|  |  |  |
| --- | --- | --- |
| SQL> INSERT INTO project\_prac9a VALUES(40,  1 row created.  SQL>  SQL> SELECT \* FROM project\_prac9a;  PNO PNAME THRS |  | 'Local', 20, 1008);  SUPER\_NO |
| 10 Hyperloop 10 |  | 1001 |
| 20 Metrorail 15 |  | 1002 |
| 30 Monorail 8 |  | 1005 |
| 40 Local 20  SQL>  SQL> -- Trigger to update project thrs on SQL> CREATE OR REPLACE TRIGGER emp\_insert   1. AFTER INSERT ON emp\_prac9a 2. FOR EACH ROW 3. WHEN(NEW.pno IS NOT NULL) -- yeh line 4. BEGIN 5. UPDATE project\_prac9a SET thrs = thrs 6. END; 7. / |  | 1008  ew employee added  waise toh zaroori nahi hai  + :NEW.hrs WHERE pno = :NEW.pno; |
| SQL>  SQL> INSERT INTO emp\_prac9a VALUES(9, 'Sanje  1 row created.  SQL> INSERT INTO emp\_prac9a VALUES(10, 'Ba  1 row created.  SQL> SELECT \* FROM project\_prac9a;  PNO PNAME THRS |  | evani', 1, 10, 1009);  athan', 9, 20, 1010);  SUPER\_NO |
| 10 Hyperloop 11 |  | 1001 |
| 20 Metrorail 24 |  | 1002 |
| 30 Monorail 8 |  | 1005 |
| 40 Local 20  SQL>  SQL> -- Trigger to update project thrs on SQL> CREATE OR REPLACE TRIGGER emp\_hrs\_upd   1. AFTER UPDATE ON emp\_prac9a 2. FOR EACH ROW 3. WHEN (NEW.pno = OLD.pno AND NEW.hrs != 4. BEGIN 5. UPDATE project\_prac9a SET thrs = thrs 6. END; 7. /   Trigger created. SQL>  SQL> UPDATE emp\_prac9a SET hrs = hrs + 5 WH  1 row updated.  SQL> UPDATE emp\_prac9a SET hrs = 1 WHERE en  1 row updated.  SQL> SELECT \* FROM project\_prac9a;  PNO PNAME THRS |  | 1008  pdate of employee hrs te  OLD.hrs)  - :OLD.hrs + :NEW.hrs WHERE pno = :NEW.pno;  ERE eno = 1;  o = 2;  SUPER\_NO |

n

Trigger created.

r

u a

|  |  |
| --- | --- |
|  |  |
| 10 Hyperloop 16 1001  20 Metrorail 20 1002  30 Monorail 8 1005  40 Local 20 1008  SQL>  SQL> -- Trigger to transfer employee hrs from old project to new project SQL> CREATE OR REPLACE TRIGGER emp\_pno\_update   1. AFTER UPDATE ON emp\_prac9a 2. FOR EACH ROW 3. WHEN (NEW.pno != OLD.pno) 4. BEGIN 5. UPDATE project\_prac9a SET thrs = thrs - :OLD.hrs WHERE pno = :OLD.pno; 6. UPDATE project\_prac9a SET thrs = thrs + :NEW.hrs WHERE pno = :NEW.pno; 7. END; 8. /   Trigger created. SQL>  SQL> UPDATE emp\_prac9a SET pno = 10 WHERE eno = 2;  1 row updated.  SQL> SELECT \* FROM project\_prac9a;  PNO PNAME THRS SUPER\_NO | |
| 10 Hyperloop 17 1001  20 Metrorail 19 1002  30 Monorail 8 1005  40 Local 20 1008  SQL>  SQL> -- Trigger to remove deduct employee hrs from project thrs when deleting employee SQL> CREATE OR REPLACE TRIGGER emp\_delete   1. AFTER DELETE ON emp\_prac9a 2. FOR EACH ROW 3. WHEN (OLD.pno IS NOT NULL) 4. BEGIN 5. UPDATE project\_prac9a SET thrs = thrs - :OLD.hrs WHERE pno = :OLD.pno; 6. END; 7. /   Trigger created. SQL>  SQL> DELETE FROM emp\_prac9a WHERE eno = 10;  1 row deleted.  SQL> SELECT \* FROM project\_prac9a;  PNO PNAME THRS SUPER\_NO | |
| 10  20  30  40  SQL> | Hyperloop 17 1001  Metrorail 10 1002  Monorail 8 1005  Local 20 1008 |

## Exercise B:

Create table stud (roll\_no, name) and stud@orcl2 (roll\_no, name) . Create a trigger such that when a student record is inserted into the table stud@orcl1, the same record should be inserted into the table stud@orcl2

## Code:

SQL> -- Practical 9B -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE TABLE stud\_prac9b (roll\_no NUMBER, sname VARCHAR2(50)); Table created.

#### SQL>

SQL> -- ORCL 2

SQL> CONNECT scott@orcl2/tiger Connected.

SQL> CREATE TABLE stud\_prac9b (roll\_no NUMBER, sname VARCHAR2(50)); Table created.

#### SQL>

SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE OR REPLACE TRIGGER stud\_insert

1. AFTER INSERT ON stud\_prac9b

#### FOR EACH ROW

1. BEGIN
2. INSERT INTO stud\_prac9b@orcl2 VALUES(:NEW.roll\_no, :NEW.sname);

#### END;

1. /

Trigger created. SQL>

SQL> INSERT INTO stud\_prac9b VALUES (1, 'Inbasagar');

1 row created.

SQL> INSERT INTO stud\_prac9b VALUES (2, 'Dhanasekhar');

1 row created.

SQL> INSERT INTO stud\_prac9b VALUES (3, 'Ashwini');

1 row created.

SQL> INSERT INTO stud\_prac9b VALUES (4, 'Sudarshan');

1 row created.

#### SQL>

SQL> -- Select from local database SQL> SELECT \* FROM stud\_prac9b;

#### ROLL\_NO SNAME

1. Inbasagar
2. Dhanasekhar
3. Ashwini
4. Sudarshan

SQL>

|  |  |
| --- | --- |
| SQL> -- Select from remote database SQL> SELECT \* FROM stud\_prac9b@orcl2;  ROLL\_NO SNAME | |
| 1 | Inbasagar |
| 2 | Dhanasekhar |
| 3 | Ashwini |
| 4 | Sudarshan |
| SQL> |  |

## Exercise C:

Create a table emp(dept\_no, eno, ename, salary) and a table dept(dept\_no, total\_sal) where the employee table stores the list of employees belonging to which department and their respective salaries. The dept table shows the total salary given to all the employees belonging to the same department.

Insert some tuples and write triggers to do the following:

1. Create a trigger such that on insert of record in the emp table the salaries of employees belonging to the same department should get added in the dept table
2. Create a trigger such that if a record is deleted from the emp table then the salary of the respective employee belonging to a specific department should get deducted from the dept table

**Code:**

SQL> -- Practical 9C -- SQL> -- ORCL 1

SQL> CONNECT scott@orcl1/tiger Connected.

SQL> CREATE TABLE emp\_prac9c (dept\_no VARCHAR2(5), eno VARCHAR2(5), ename VARCHAR2(20), salary NUMBER);

Table created. SQL>

SQL> CREATE TABLE dept\_prac9c (dept\_no VARCHAR(5), total\_sal NUMBER);

Table created. SQL>

SQL> INSERT INTO dept\_prac9c VALUES ('d01', 0);

1. row created.

SQL> INSERT INTO dept\_prac9c VALUES ('d02', 0);

1. row created.

SQL>

SQL> -- Trigger to add salary of employee to department on adding employee SQL> CREATE OR REPLACE TRIGGER emp\_prac9c\_insert

1. AFTER INSERT ON emp\_prac9c
2. FOR EACH ROW
3. WHEN (NEW.salary IS NOT NULL AND NEW.dept\_no IS NOT NULL)
4. BEGIN
5. UPDATE dept\_prac9c SET total\_sal = total\_sal + :NEW.salary WHERE dept\_no = :NEW.dept\_no;
6. END;
7. /

Trigger created. SQL>

SQL> INSERT INTO emp\_prac9c VALUES ('d01', 'e01', 'Inbasagar', 100000);

1 row created.

SQL> INSERT INTO emp\_prac9c VALUES ('d02', 'e02', 'Dhanasekhar', 50000);

1 row created.

SQL> INSERT INTO emp\_prac9c VALUES ('d01', 'e03', 'Sudarhsan', 60000);

1 row created.

SQL> INSERT INTO emp\_prac9c VALUES ('d02', 'e04', 'Ashwini', 70000);

1 row created.

SQL> INSERT INTO emp\_prac9c VALUES ('d02', 'e05', 'Yash', 80000);

1 row created.

SQL> SELECT \* FROM dept\_prac9c; DEPT\_ TOTAL\_SAL

d01

d02

160000

200000

SQL>

SQL> -- Trigger to subtract salary of employee from department on removing employee SQL> CREATE OR REPLACE TRIGGER emp\_prac9c\_delete

1. AFTER DELETE ON emp\_prac9c
2. FOR EACH ROW
3. WHEN (OLD.dept\_no IS NOT NULL)
4. BEGIN
5. UPDATE dept\_prac9c SET total\_sal = total\_sal - :OLD.salary WHERE dept\_no = :OLD.dept\_no;
6. END;
7. /

Trigger created. SQL>

SQL> DELETE emp\_prac9c WHERE eno = 'e05';

1 row deleted.

SQL> SELECT \* FROM dept\_prac9c; DEPT\_ TOTAL\_SAL

d01

d02

160000

120000

SQL>

# Practical 10: XML Database

**Aim:** To demonstrate the use of XML database and implement it using Oracle Database

## Exercise A:

Create a table emp\_xml 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, manager\_email, data\_of\_joning. Insert some tuples into employee table. Fire the following queries on XML database:

1. Retrieve the names of employee.
2. Retrieve the acc\_no of employees.
3. Retrieve the names, acc\_no, email of employees.
4. Update the row where employee id is 4 and display the name of an employee.
5. Delete row with employee whose id is 4

## Code:

**SQL> -- Practical 10A -- SQL> -- ORCL 1**

**SQL> CONNECT scott@orcl1/tiger Connected.**

**SQL> CREATE TABLE emp\_xml\_prac10a(dept\_id NUMBER, employee\_spec XMLTYPE); Table created.**

**SQL>**

**SQL> INSERT INTO emp\_xml\_prac10a VALUES(**

**2 1,**

1. **XMLTYPE(**
2. **'<emp id="1">**
3. **<name>Inbasagar</name>**
4. **<email>inbasagar@gmail</email>**
5. **<acc\_no>1234</acc\_no>**
6. **<mgr\_email>**[**manager1@gmail.com**](mailto:manager1@gmail.com)**</mgr\_email>**
7. **<doj>10/09/2015</doj>**
8. **</emp>'**
9. **)**
10. **);**

**1 row created.**

**SQL> INSERT INTO emp\_xml\_prac10a VALUES(**

**2 1,**

1. **XMLTYPE(**
2. **'<emp id="2">**
3. **<name>Sudarhsan</name>**
4. **<email>sudarshan@gmail</email>**
5. **<acc\_no>2345</acc\_no>**
6. **<mgr\_email>**[**manager1@gmail.com**](mailto:manager1@gmail.com)**</mgr\_email>**
7. **<doj>14/11/2015</doj>**
8. **</emp>'**
9. **)**
10. **);**

**1 row created.**

**SQL> INSERT INTO emp\_xml\_prac10a VALUES(**

**2 2,**

1. **XMLTYPE(**
2. **'<emp id="3">**
3. **<name>Ashwini</name>**
4. **<email>ashwini@gmail</email>**
5. **<acc\_no>3456</acc\_no>**
6. **<mgr\_email>**[**manager2@gmail.com**](mailto:manager2@gmail.com)**</mgr\_email>**
7. **<doj>11/11/2015</doj>**
8. **</emp>'**
9. **)**
10. **);**

**1 row created.**

### SQL>

**SQL> INSERT INTO emp\_xml\_prac10a VALUES(**

**2 2,**

### XMLTYPE(

1. **'<emp id="4">**
2. **<name>Angelica</name>**
3. **<email>angelica@gmail</email>**
4. **<acc\_no>4567</acc\_no>**
5. **<mgr\_email>**[**manager2@gmail.com**](mailto:manager2@gmail.com)**</mgr\_email>**
6. **<doj>17/11/2015</doj>**
7. **</emp>'**
8. **)**
9. **);**

**1 row created.**

### SQL>

**SQL> INSERT INTO emp\_xml\_prac10a VALUES(**

**2 3,**

### XMLTYPE(

1. **'<emp id="5">**
2. **<name>Sonia</name>**
3. **<email>angelica@gmail</email>**
4. **<acc\_no>4567</acc\_no>**
5. **<mgr\_email>**[**manager3@gmail.com**](mailto:manager3@gmail.com)**</mgr\_email>**
6. **<doj>01/09/2015</doj>**
7. **</emp>'**
8. **)**
9. **);**
10. **row created.**

### SQL>

**SQL> -- Select names for all employee**

**SQL> SELECT e.employee\_spec.EXTRACT('//name/text()').GETSTRINGVAL() "Employee Name" FROM emp\_xml\_prac10a e;**

**Employee Name**

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

**Inbasagar Sudarhsan Ashwini Angelica Sonia**

### SQL>

**SQL> SELECT e.employee\_spec.EXTRACT('//name/text()').GETSTRINGVAL() "Employee Name" FROM emp\_xml\_prac10a e;**

**Employee Name**

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

**Inbasagar Sudarhsan Ashwini Angelica Sonia**

### SQL>

**SQL>**

**SQL> -- Select account number of employees**

**SQL> SELECT e.employee\_spec.EXTRACT('//acc\_no/text()').GETSTRINGVAL() "Account Number" FROM emp\_xml\_prac10a e;**

**Account Number**

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

**1234**

**2345**

**3456**

**4567**

**4567**

### SQL>

**SQL> -- Select name, account and**

**SQL> SELECT e.employee\_spec.EXTRACT('//name/text()').GETSTRINGVAL() "Name", e.employee\_spec.EXTRACT('//acc\_no/text()').GETSTRINGVAL() "Account Number", e.employee\_spec.EXTRACT('//email/text()').GETSTRINGVAL() "Email" FROM emp\_xml\_prac10a e;**

**Name**

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

**Account Number**

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

**Email**

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

**Inbasagar 1234**

**inbasagar@gmail**

**Sudarhsan 2345**

**sudarshan@gmail**

**Ashwini 3456**

**ashwini@gmail**

**Angelica 4567**

**angelica@gmail**

**Sonia 4567**

**angelica@gmail**

### SQL>

**SQL> -- Update 3rd record from the table SQL> UPDATE emp\_xml\_prac10a e**

1. **SET employee\_spec = XMLTYPE(**
2. **'<emp id="3">**
3. **<name>Ashwini</name>**
4. **<email>ashwini@gmail</email>**
5. **<acc\_no>4567</acc\_no>**
6. **<mgr\_email>**[**manager2@gmail.com**](mailto:manager2@gmail.com)**</mgr\_email>**
7. **<doj>11/11/2015</doj>**
8. **<update>Record Updated</update>**
9. **</emp>'**
10. **)**
11. **WHERE e.employee\_spec.EXTRACT('//@id').GETSTRINGVAL() = '3';**

**1 row updated.**

### SQL>

**SQL> SELECT e.employee\_spec.GETSTRINGVAL() FROM emp\_xml\_prac10a e WHERE e.employee\_spec.EXTRACT('//@id').GETSTRINGVAL()='3';**

### E.EMPLOYEE\_SPEC.GETSTRINGVAL()

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

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

**<name>Ashwini</name>**

**<email>ashwini@gmail</email>**

**<acc\_no>4567</acc\_no>**

**<mgr\_email**[**>m**](mailto:manager2@gmail.com)**a**[**nager2@gmail.com**](mailto:manager2@gmail.com)**</mgr\_email>**

**<doj>11/11/2015</doj>**

**<update>Record Updated</update>**

**</emp>**

**SQL>**

**SQL> -- Delete the row with employee id 4**

**SQL> DELETE FROM emp\_xml\_prac10a e WHERE e.employee\_spec.EXTRACT('//@id').GETSTRINGVAL() = '4';**

**1 row deleted.**

**SQL> SELECT e.employee\_spec.EXTRACT('//name/text()').GETSTRINGVAL() "Employee Name" FROM emp\_xml\_prac10a e;**

**Employee Name**

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

**Inbasagar Sudarhsan Ashwini Sonia**

**SQL>**

## Exercise B:

Create a table candidate having cand\_id as varchar2 datatype and biodata as XML datatype ( XML type). The biodata is a schema with attributes Name, address, skill (compskill and language) networking, experience (programming and project manager), objectives. Fire the following queries on XML database:

1. Display candidate name who is good in java and having experience more than 5 years
2. Display candidate having project manager level experience
3. Display name and skill of all candidates
4. Delete record for address = Borivali
5. Update experience of a particular candidate

## Code:

**SQL> -- Practical 10B -- SQL> -- ORCL 1**

**SQL> CONNECT scott@orcl1/tiger Connected.**

**SQL> CREATE TABLE candidate\_prac10b (cand\_id VARCHAR2(5) PRIMARY KEY, biodata XMLTYPE); Table created.**

**SQL>**

**SQL> INSERT INTO candidate\_prac10b VALUES('C01', XMLTYPE(**

1. **'<cand>**
2. **<name>Inbasagar</name>**
3. **<address>Sion</address>**
4. **<skill>**
5. **<compskill>**
6. **<lang>Java</lang>**
7. **<networkskill>Windows 10</networkskill>**
8. **</compskill>**
9. **</skill>**
10. **<expr>**
11. **<programmer>7</programmer>**
12. **<projectmgr>7</projectmgr>**
13. **</expr>**
14. **<objectives>To be an Enterpreneur</objectives>**
15. **</cand>'**
16. **));**
17. **row created.**

### SQL>

**SQL> INSERT INTO candidate\_prac10b VALUES('C02', XMLTYPE(**

1. **'<cand>**
2. **<name>Ashwini</name>**
3. **<address>Powai</address>**
4. **<skill>**
5. **<compskill>**
6. **<lang>Python</lang>**
7. **<networkskill>Windows 7</networkskill>**
8. **</compskill>**
9. **</skill>**
10. **<expr>**
11. **<programmer>2</programmer>**
12. **<projectmgr>2</projectmgr>**
13. **</expr>**
14. **<objectives>To build company</objectives>**
15. **</cand>'**
16. **));**
17. **row created.**

### SQL>

**SQL> INSERT INTO candidate\_prac10b VALUES('C03', XMLTYPE(**

1. **'<cand>**
2. **<name>Sudarshan</name>**
3. **<address>Titwala</address>**
4. **<skill>**
5. **<compskill>**
6. **<lang>Java</lang>**
7. **<networkskill>Windows 8</networkskill>**
8. **</compskill>**
9. **</skill>**
10. **<expr>**
11. **<programmer>2</programmer>**
12. **</expr>**
13. **<objectives>To build pc</objectives>**
14. **</cand>'**
15. **));**
16. **row created.**

### SQL>

**SQL> INSERT INTO candidate\_prac10b VALUES('C04', XMLTYPE(**

1. **'<cand>**
2. **<name>Jalebi</name>**
3. **<address>Borivali</address>**
4. **<skill>**
5. **<compskill>**
6. **<lang>VB</lang>**
7. **<networkskill>Windows XP</networkskill>**
8. **</compskill>**
9. **</skill>**
10. **<expr>**
11. **<programmer>1</programmer>**
12. **</expr>**
13. **<objectives>To build pc</objectives>**
14. **</cand>'**
15. **));**

**1 row created.**

### SQL>

**SQL> -- Candidate with more than 5 years experience in java**

**SQL> SELECT c.biodata.EXTRACT('//name/text()') "Candidate Name" FROM candidate\_prac10b c WHERE c.biodata.EXTRACT('//lang/text()').GETSTRINGVAL() = 'Java' AND c.biodata.EXTRACT('//programmer/text()').GETSTRINGVAL() > 5;**

**Candidate Name**

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

**Inbasagar**

### SQL>

**SQL> -- Candidate having progect manager level experience**

**SQL> SELECT EXTRACT(biodata, '//name/text()') "Candidate Name" FROM candidate\_prac10b WHERE EXISTSNODE(biodata,'//projectmgr') = 1;**

**Candidate Name**

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

**Inbasagar Ashwini**

**SQL> -- Or alternatively**

**SQL> SELECT c.biodata.EXTRACT('//name/text()') "Candidate Name" FROM candidate\_prac10b c WHERE c.biodata.EXISTSNODE('//projectmgr') = 1;**

**Candidate Name**

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

**Inbasagar Ashwini**

### SQL>

**SQL> -- Display candidate name and skill of all candidate**

**SQL> SELECT EXTRACT(biodata, '//name/text()') "Name", EXTRACT(biodata, '//lang/text()') "Language", EXTRACT(biodata, '//networkskill/text()') "Network" FROM candidate\_prac10b;**

**Name**

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

**Language**

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

**Network**

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

**Inbasagar Java Windows 10**

**Ashwini Python Windows 7**

**Sudarshan Java Windows 8**

**Jalebi VB**

**Windows XP**

### SQL>

**SQL> -- Delete record for candidate having address borivali**

**SQL> DELETE FROM candidate\_prac10b WHERE EXTRACT(biodata,'//address/text()').GETSTRINGVAL() = 'Borivali';**

**1 row deleted.**

**SQL>**