DBMS Practical

**Q** – Making a student table

**Ans –**

CREATE TABLE Student (

StudentId NUMBER(4) PRIMARY KEY,

Student\_name VARCHAR2(40) NOT NULL,

Address1 VARCHAR2(300),

Gender VARCHAR2(15),

Course VARCHAR2(8)

);

**Q** – Making a course table

**Ans –**

CREATE TABLE Course (

DeptNo NUMBER(2) PRIMARY KEY,

Dname VARCHAR2(20),

Location VARCHAR2(10)

);

**Q** – Making a Customer & Orders table

**Ans –**

CREATE TABLE CUSTOMER (

SID NUMBER PRIMARY KEY,

Last\_Name VARCHAR2(50),

First\_Name VARCHAR2(50)

);

CREATE TABLE ORDERS (

Order\_ID NUMBER PRIMARY KEY,

Order\_Date DATE,

Customer\_SID NUMBER,

Amount NUMBER CHECK (Amount > 20000),

FOREIGN KEY (Customer\_SID) REFERENCES CUSTOMER(SID)

);

**Q** - Data Insertion

**Ans** –

INSERT INTO Student VALUES (7369, 'Raj Sharma', '789 Pune Avenue', 'Male', 'MCA');

INSERT INTO Student VALUES (7777, 'Priya Singh', '321 Kolkata Lane', 'Female', 'BSc');

INSERT INTO Student VALUES (2233, 'Sarah Lee', '456 Mumbai Street', 'Female', 'BCA');

INSERT INTO CUSTOMER VALUES (1, 'Patel', 'Amit');

INSERT INTO CUSTOMER VALUES (2, 'Gupta', 'Neha');

**Q** - Basic SELECT Queries

**Ans** –

**List all students**

SELECT \* FROM Student;

**Students in MCA course**

SELECT \* FROM Student WHERE Course = 'MCA';

**Specific student IDs**

SELECT Student\_name FROM Student WHERE StudentId IN (7369, 7777, 2233);

**Names starting with 'S'**

SELECT Student\_name FROM Student WHERE Student\_name LIKE 'S%';

**Names ending with 's'**

SELECT Student\_name FROM Student WHERE Student\_name LIKE '%s';

**Second character is 'k'**

SELECT Student\_name FROM Student WHERE Student\_name LIKE '\_k%';

**Q** - Filtering & Conditions

**Ans** –

SELECT Student\_name FROM Student

WHERE StudentId NOT IN (

SELECT StudentId FROM Course WHERE DeptNo IN (10, 40)

);

SELECT Student\_name FROM Student WHERE Course IS NULL;

SELECT \* FROM ORDERS WHERE Amount BETWEEN 21000 AND 30000;

**Q** - Sorting & Ordering

**Ans** –

SELECT \* FROM Student ORDER BY Course ASC;

SELECT Student\_name FROM Student ORDER BY Student\_name ASC;

**Q** - Constraints

**Ans** –

-- Primary Key

CREATE TABLE Department (

Dept\_ID NUMBER(4) PRIMARY KEY,

Dept\_Name VARCHAR2(50)

);

-- NOT NULL

CREATE TABLE Employee (

Emp\_ID NUMBER(6) NOT NULL,

Emp\_Name VARCHAR2(50) NOT NULL

);

-- UNIQUE

CREATE TABLE Login (

User\_ID NUMBER(6) PRIMARY KEY,

Username VARCHAR2(30) UNIQUE,

Email VARCHAR2(50) UNIQUE

);

-- Foreign Key

CREATE TABLE Enrollment (

Enrollment\_ID NUMBER(6) PRIMARY KEY,

DeptNo NUMBER(2),

FOREIGN KEY (DeptNo) REFERENCES Course(DeptNo)

);

**Q** - Joins

**Ans** –

-- INNER JOIN

SELECT c.SID, c.First\_Name, c.Last\_Name, o.Amount

FROM CUSTOMER c

INNER JOIN ORDERS o ON c.SID = o.Customer\_SID;

-- LEFT JOIN

SELECT \* FROM s4 LEFT JOIN s5 ON s4.common\_column = s5.common\_column;

-- RIGHT JOIN

SELECT \* FROM s4 RIGHT JOIN s5 ON s4.common\_column = s5.common\_column;

-- FULL OUTER JOIN

SELECT \* FROM s4 FULL OUTER JOIN s5 ON s4.common\_column = s5.common\_column;

-- SELF JOIN (Employee-Manager)

SELECT e.First\_Name AS Employee\_Name, m.First\_Name AS Manager\_Name

FROM Employee e

LEFT JOIN Employee m ON e.Manager\_ID = m.Emp\_ID;

**Q** - Set Operations

**Ans** –

-- UNION

SELECT \* FROM s4

UNION

SELECT \* FROM s5;

-- INTERSECT

SELECT \* FROM s4

INTERSECT

SELECT \* FROM s5;

**Q** - Data Modification

**Ans** –

-- UPDATE with calculation

UPDATE ORDERS SET Amount = Amount + 500;

-- Display with alias

SELECT Order\_ID, Amount AS new\_amount FROM ORDERS;