# **RDBMS LAB**

# **CLASS: I MCA D**

## Part A - Question 1:

DEPARTMENT (dept\_no, dept\_name, location)

1. Create the Simple DEPARTMENT Table.
2. Display structure of department table.
3. Insert below records into Department Table

|  |  |  |
| --- | --- | --- |
| Detp\_no | Dept\_name | Location |
| 10 | Account | NY |
| 20 | HR | NY |
| 30 | Production | DL |
| 40 | Sales | NY |
| 50 | EDP | MU |
| 60 | TRG |  |
| 110 | RND | AH |

1. Display all records of Department table
2. Display all department belonging to location 'NY'
3. Display details of Department 10
4. List all department names starting with 'A'
5. List all departments whose number is between 1 and 100
6. Delete 'TRG' department
7. Change department name 'EDP' to 'IT

## Part A - Answer 1:

## DEPARTMENT Table

## (MySQL Server)

### Create the DEPARTMENT Table (MySQL Syntax)

CREATE TABLE DEPARTMENT (

dept\_no INT PRIMARY KEY,

dept\_name VARCHAR(50) NOT NULL UNIQUE,

location VARCHAR(50)

);

### Display Structure of the DEPARTMENT Table

DESC DEPARTMENT;

### Insert Records into the DEPARTMENT Table

### INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (10, 'Account', 'NY');

### INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (20, 'HR', 'NY');

### INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (30, 'Production', 'DL');

### INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (40, 'Sales', 'NY');

### INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (50, 'EDP', 'MU');

### INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (60, 'TRG', NULL);

### INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (110, 'RND', 'AH');

COMMIT;

### Display All Records of the DEPARTMENT Table

SELECT \* FROM DEPARTMENT;

### Display All Departments Belonging to Location 'NY'

SELECT \* FROM DEPARTMENT WHERE location = 'NY';

### Display Details of Department 10

SELECT \* FROM DEPARTMENT WHERE dept\_no = 10;

### List All Department Names Starting with 'A'

SELECT \* FROM DEPARTMENT WHERE dept\_name LIKE 'A%';

### List All Departments Whose Number is Between 1 and 100

SELECT \* FROM DEPARTMENT WHERE dept\_no BETWEEN 1 AND 100;

### Delete 'TRG' Department

DELETE FROM DEPARTMENT WHERE dept\_name = 'TRG';

### Change Department Name 'EDP' to 'IT'

UPDATE DEPARTMENT SET dept\_name = 'IT' WHERE dept\_name = 'EDP';

## Part A - Question 2:

EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email)

DEPARTMENT (dept\_no, dept\_name, location)

Create the EMP Table with all necessary constraints such as

In EMP TABLE: Employee id should be primary key, Department no should be Foreign key, employee age (birth\_date) should be greater than 18 years, salary shouldbe greater than zero, email should have (@anddot) sign in address, designation of employee can be "manager", "clerk", "leader", "analyst", "designer", "coder","tester".

* 1. Create DEPT table with necessary constraint such as
  2. Department no should be primary key, department name should be unique.
  3. After creation of above tables, modify Employee table by adding the constraints as
  4. 'Male' or 'Female' in gender field and display the structure.
  5. Insert proper data (at least 5 appropriate records) in all the tables.
  6. Describe the structure of table created
  7. List all records of each table in ascending order.
  8. Delete the department whose loction is Ahmedabad.
  9. Display female employee list
  10. Display Departname wise employee Names
  11. Find the names of the employee who has salary less than 5000 and greater than 2000.
  12. Display the names and the designation of all female employee in descending order.
  13. Display the names of all the employees who names starts with 'A' ends with 'A'.
  14. Find the name of employee and salary for those who had obtain minimum salary.
  15. Add 10% raise in salary of all employees whose department is 'IT'.
  16. Count total number of employees of 'IT' department.
  17. List all employees who born in the current month.
  18. Print the record of employee and dept table as "Employee works in department MBA'.
  19. List names of employees who are fresher's (less than 1 year of experience).
  20. List department wise names of employees who has more than 5 years of experience.
  21. Create Sequence to generate department ID
  22. List department having no employee

## Part A - Answer 2:

## EMPLOYEE and DEPARTMENT Tables (MySQL Server)

### Create the DEPARTMENT Table with Constraints

* + dept\_no is **Primary Key**
  + dept\_name is **Unique**

CREATE TABLE DEPARTMENT (

dept\_no INT PRIMARY KEY,

dept\_name VARCHAR(50) UNIQUE NOT NULL,

location VARCHAR(50)

);

### Create the EMPLOYEE Table with Constraints

CREATE TABLE EMPLOYEE (

emp\_id INT PRIMARY KEY,

emp\_name VARCHAR2(50) NOT NULL,

birth\_date DATE NOT NULL,

gender VARCHAR2(10) CHECK (gender IN ('Male', 'Female')) NOT NULL,

dept\_no INT,

address VARCHAR2(255),

designation VARCHAR2(20) CHECK (designation IN ('Manager', 'Clerk', 'Leader', 'Analyst', 'Designer', 'Coder', 'Tester')) NOT NULL,

salary DECIMAL(10,2) CHECK (salary > 0),

experience INT CHECK (experience >= 0),

email VARCHAR2(100) NOT NULL,

CONSTRAINT fk\_dept FOREIGN KEY (dept\_no) REFERENCES DEPARTMENT(dept\_no) ON DELETE CASCADE,

CONSTRAINT ck\_email CHECK (email LIKE '%@%.%')

);

--Trigger for Age Restriction

CREATE OR REPLACE TRIGGER trg\_emp\_age\_check

BEFORE INSERT OR UPDATE ON EMPLOYEE

FOR EACH ROW

BEGIN

IF ADD\_MONTHS(:NEW.birth\_date, 18 \* 12) > SYSDATE THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Employee must be at least 18 years old.');

END IF;

END;

/

To Create Auto Increment for emp\_id in oracle (Not mandatary)

-- Oracle **11g** does **not** support the IDENTITY column (introduced in **Oracle 12c**). Instead, we need to use a **SEQUENCE** and a **BEFORE INSERT TRIGGER** to achieve auto-increment functionality.

CREATE SEQUENCE emp\_id\_seq

START WITH 1

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- **Trigger** to automatically assign values from the sequence to emp\_id

CREATE OR REPLACE TRIGGER trg\_emp\_id\_auto

BEFORE INSERT ON EMPLOYEE

FOR EACH ROW

BEGIN

IF :NEW.emp\_id IS NULL THEN

SELECT emp\_id\_seq.NEXTVAL INTO :NEW.emp\_id FROM DUAL;

END IF;

END;

/

### Insert Sample Data into DEPARTMENT

INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (10, 'Accounts', 'NY');

INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (20, 'HR', 'NY');

INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (30, 'Production', 'DL');

INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (40, 'Sales', 'NY');

INSERT INTO DEPARTMENT (dept\_no, dept\_name, location) VALUES (50, 'IT', 'MU');

COMMIT;

### Insert Sample Data into EMPLOYEE

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(101, 'Alice', TO\_DATE('12-MAY-1990', 'DD-MON-YYYY'), 'Female', 10, '123 Main St', 'Manager', 60000, 10, 'alice@example.com');

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(102, 'Bob', TO\_DATE('23-SEP-1995', 'DD-MON-YYYY'), 'Male', 20, '456 Elm St', 'Clerk', 35000, 5, 'bob@example.com');

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(103, 'Charlie', TO\_DATE('11-FEB-1998', 'DD-MON-YYYY'), 'Male', 30, '789 Oak St', 'Analyst', 45000, 3, 'charlie@example.com');

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(104, 'David', TO\_DATE('30-JUL-2000', 'DD-MON-YYYY'), 'Male', 40, '321 Maple St', 'Coder', 40000, 2, 'david@example.com');

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(105, 'Eve', TO\_DATE('15-OCT-1992', 'DD-MON-YYYY'), 'Female', 50, '654 Pine St', 'Designer', 50000, 8, 'eve@example.com');

COMMIT;

### Display Structure of the Tables

DESC EMPLOYEE; DESC DEPARTMENT;

### List All Records in Ascending Order

SELECT \* FROM EMPLOYEE ORDER BY emp\_name ASC;

SELECT \* FROM DEPARTMENT ORDER BY dept\_name ASC;

### Delete the Department Whose Location is 'Ahmedabad'

DELETE FROM DEPARTMENT WHERE location = 'AH';

COMMIT;

### Display Female Employees

SELECT \* FROM EMPLOYEE WHERE gender = 'Female';

### Display Department-Wise Employee Names

SELECT d.dept\_name, e.emp\_name FROM EMPLOYEE e

JOIN DEPARTMENT d ON e.dept\_no = d.dept\_no ORDER BY d.dept\_name;

### Find Employees with Salary Between 2000 and 5000

SELECT emp\_name FROM EMPLOYEE WHERE salary BETWEEN 2000 AND 5000;

### Display Female Employees in Descending Order

SELECT emp\_name, designation FROM EMPLOYEE WHERE gender = 'Female' ORDER BY emp\_name DESC;

### Employees Whose Name Starts and Ends with 'A'

SELECT emp\_name FROM EMPLOYEE WHERE emp\_name LIKE 'A%A';

### Employees with Minimum Salary

SELECT emp\_name, salary FROM EMPLOYEE

WHERE salary = (SELECT MIN(salary) FROM EMPLOYEE);

### Increase Salary by 10% for IT Department Employees

UPDATE EMPLOYEE

SET salary = salary \* 1.10

WHERE dept\_no = (SELECT dept\_no FROM DEPARTMENT WHERE dept\_name = 'IT');

COMMIT;

### Count Total Employees in IT Department

SELECT COUNT(\*) FROM EMPLOYEE

WHERE dept\_no = (SELECT dept\_no FROM DEPARTMENT WHERE dept\_name = 'IT');

### List Employees Born in the Current Month

SELECT \* FROM EMPLOYEE

WHERE EXTRACT(MONTH FROM birth\_date) = EXTRACT(MONTH FROM SYSDATE);

### Print Employee and Department Details in Sentence Format

SELECT emp\_name || ' works in department ' || dept\_name AS employee\_info

FROM EMPLOYEE

JOIN DEPARTMENT ON EMPLOYEE.dept\_no = DEPARTMENT.dept\_no;

### List Freshers (Experience < 1 Year)

SELECT emp\_name FROM EMPLOYEE WHERE experience < 1;

### List Department-Wise Employees with >5 Years Experience

SELECT d.dept\_name, e.emp\_name FROM EMPLOYEE e

JOIN DEPARTMENT d ON e.dept\_no = d.dept\_no WHERE e.experience > 5;

1. Create a Sequence to Generate Department ID

(Creates a sequence named dept\_id\_seq that starts at 51 and increments by 1 for each subsequent value generated.)

CREATE SEQUENCE dept\_id\_seq START WITH 51 INCREMENT BY 1;

### List Departments with No Employees

SELECT d.dept\_name FROM DEPARTMENT d

LEFT JOIN EMPLOYEE e ON d.dept\_no = e.dept\_no WHERE e.emp\_id IS NULL;

## Part A - Question 3:

STUDENT (rollno, name, class, birthdate)

COURSE (courseno, coursename, max\_marks, pass marks)

SC (rollno, courseno, marks)

1. Create the above three tables along with key constraints.
2. Write an Insert script for insertion of rows with substitution variables and insertappropriate data.
3. Add a constraint that the marks entered should strictly be between 0 and 100.
4. While creating SC table, composite key constraint was forgotten. Add the compositekeynow.
5. Display details of student who takes 'Database Management System' course.
6. Display the names of students who have scored more than 70% in Computer Networksand have not failed in any subject.
7. Display the average marks obtained by each student.
8. Select all courses where passing marks are more than 30% of average maximum mark.
9. Display details of students who are born in 1980 or 1982.
10. Create a view that displays student courseno and its corresponding marks

## Part A - Answer 3:

## STUDENT, COURSE and SC Tables (MySQL Server)

### Create the Tables with Constraints

#### STUDENT Table

* + rollno is **Primary Key**
  + birthdate must not be **NULL**

CREATE TABLE STUDENT ( rollno INT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

class VARCHAR(20) NOT NULL,

birthdate DATE NOT NULL

);

#### COURSE Table

* + courseno is **Primary Key**
  + max\_marks must be **greater than 0**
  + pass\_marks must be **greater than 0** and **less than or equal to max\_marks**

CREATE TABLE COURSE (

courseno INT PRIMARY KEY, coursename VARCHAR(100) NOT NULL,

max\_marks INT CHECK (max\_marks > 0), pass\_marks INT,

CHECK (pass\_marks > 0 AND pass\_marks <= max\_marks)

);

#### SC Table (Student-Course Relationship Table)

* + **Composite Primary Key** (rollno, courseno)
  + marks must be **between 0 and 100**
  + rollno is a **Foreign Key** referencing STUDENT(rollno)
  + courseno is a **Foreign Key** referencing COURSE(courseno)

CREATE TABLE SC (

rollno INT,

courseno INT,

marks INT CHECK (marks BETWEEN 0 AND 100),

PRIMARY KEY (rollno, courseno),

FOREIGN KEY (rollno) REFERENCES STUDENT(rollno) ON DELETE CASCADE,

FOREIGN KEY (courseno) REFERENCES COURSE(courseno) ON DELETE CASCADE

);

### Insert Sample Data into the Tables

#### Insert Students

INSERT INTO STUDENT (rollno, name, class, birthdate) VALUES

(1, 'Alice', 'MCA', TO\_DATE('10-MAY-1998', 'DD-MON-YYYY'));

INSERT INTO STUDENT (rollno, name, class, birthdate) VALUES

(2, 'Bob', 'MCA', TO\_DATE('15-AUG-1997', 'DD-MON-YYYY'));

INSERT INTO STUDENT (rollno, name, class, birthdate) VALUES

(3, 'Charlie', 'BSc', TO\_DATE('20-FEB-1999', 'DD-MON-YYYY'));

COMMIT;

Insert Courses

INSERT INTO COURSE (courseno, coursename, max\_marks, pass\_marks) VALUES

INSERT INTO COURSE (courseno, coursename, max\_marks, pass\_marks) VALUES

(101, 'Database Management System', 100, 40);

INSERT INTO COURSE (courseno, coursename, max\_marks, pass\_marks) VALUES

(102, 'Computer Networks', 100, 35);

INSERT INTO COURSE (courseno, coursename, max\_marks, pass\_marks) VALUES

(103, 'Data Structures', 100, 50);

COMMIT;

Insert Student-Course Marks

INSERT INTO SC (rollno, courseno, marks) VALUES

(1, 101, 85);

INSERT INTO SC (rollno, courseno, marks) VALUES

(1, 102, 72);

INSERT INTO SC (rollno, courseno, marks) VALUES

(2, 101, 40);

INSERT INTO SC (rollno, courseno, marks) VALUES

(2, 103, 55);

INSERT INTO SC (rollno, courseno, marks) VALUES

(3, 102, 30);

INSERT INTO SC (rollno, courseno, marks) VALUES

(3, 103, 90);

COMMIT;

### Add Constraint to Ensure Marks are Between 0 and 100

(Already added in the SC table using CHECK (marks BETWEEN 0 AND 100))

### Add Composite Key Constraint to SC Table

(Already defined in SC table: PRIMARY KEY (rollno, courseno))

### Display Details of Students Taking 'Database Management System'

SELECT s.rollno, s.name, s.class, s.birthdate, c.coursename, sc.marks

FROM STUDENT s

JOIN SC sc ON s.rollno = sc.rollno

JOIN COURSE c ON sc.courseno = c.courseno

WHERE c.coursename = 'Database Management System';

### Display Students Who Scored More Than 70% in 'Computer Networks' and Have

Not Failed in Any Subject

SELECT s.rollno, s.name

FROM STUDENT s

JOIN SC sc ON s.rollno = sc.rollno

JOIN COURSE c ON sc.courseno = c.courseno

WHERE c.coursename = 'Computer Networks' AND sc.marks > (c.max\_marks \* 0.7)

AND s.rollno NOT IN (

SELECT sc.rollno

FROM SC sc

JOIN COURSE c ON sc.courseno = c.courseno

WHERE sc.marks < c.pass\_marks

);

### Display the Average Marks Obtained by Each Student

SELECT s.rollno, s.name, AVG(sc.marks) AS avg\_marks FROM STUDENT s

JOIN SC sc ON s.rollno = sc.rollno GROUP BY s.rollno, s.name;

### Select All Courses Where Passing Marks Are More Than 30% of Maximum Marks

SELECT \* FROM COURSE WHERE pass\_marks > (max\_marks \* 0.3);

### Display Details of Students Born in 1980 or 1982

SELECT \*

FROM STUDENT

WHERE TO\_CHAR(birthdate, 'YYYY') IN ('1980', '1982');

### Create a View That Displays Student Roll Number, Course Number, and Marks

CREATE VIEW Student\_Course\_Marks AS SELECT rollno, courseno, marks FROM SC;

--To fetch data from the view:

SELECT \* FROM Student\_Course\_Marks;

## Part A - Question 4:

Create the database COMPANY and create given tables with all necessary constraints such asprimary key, foreign key, unique key, not null and check constraints.

EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept no, address, designation, salary, experience, email)

DEPART (dept no, dept\_name, total\_employees, location)

PROJECT (proj\_id, type\_of\_project, status, start\_date, emp\_id)

Insert proper data (at least 5 appropriate records) in all the tables.

1. Delete the department whose total number of employees less than 1.
2. Display the names and the designation of all female employee in descending order.
3. Display the names of all the employees who names starts with 'A' ends with 'A'.
4. Find the name of employee and salary for those who had obtain minimum salary.
5. Add 10% raise in salary of all employees whose department is CIVIL.
6. Count total number of employees of 'MCA' department.
7. List all employees who born in the current month.
8. Print the record of employee and dept table as "Employee works in department 'CE'.
9. List names of employees who are fresher's(less than 1 year of experience).
10. List department wise names of employees who has more than 5 years of experience.

## Part A - Answer 4: COMPANY Database (MySQL Server)

### Create the DEPART Table

* + dept\_no is **Primary Key**
  + dept\_name is **Unique**

**CREATE TABLE DEPART (**

**dept\_no INT PRIMARY KEY,**

**dept\_name VARCHAR2(50) UNIQUE NOT NULL,**

**total\_employees INT CHECK (total\_employees >= 0),**

**location VARCHAR2(50)**

**);**

### Create the EMPLOYEE Table with Constraints

* + emp\_id is **Primary Key**
  + dept\_no is **Foreign Key** referencing DEPART(dept\_no)
  + birth\_date should ensure **age > 18 years**
  + salary should be **greater than 0**
  + email should contain **@ and .**
  + designation should be one of: **'Manager', 'Clerk', 'Leader', 'Analyst', 'Designer', 'Coder', 'Tester'**
  + gender should be **'Male' or 'Female'**
  + **Oracle 11g doesn't directly support AUTO\_INCREMENT or ENUM types in the same way as MySQL.**

**CREATE TABLE EMPLOYEE (**

**emp\_id INT PRIMARY KEY,**

**emp\_name VARCHAR2(50) NOT NULL,**

**birth\_date DATE NOT NULL,**

**gender VARCHAR2(10) CHECK (gender IN ('Male', 'Female')) NOT NULL,**

**dept\_no INT,**

**address VARCHAR2(255),**

**designation VARCHAR2(20) CHECK (designation IN ('Manager', 'Clerk', 'Leader', 'Analyst', 'Designer', 'Coder', 'Tester')) NOT NULL,**

**salary DECIMAL(10,2) CHECK (salary > 0),**

**experience INT CHECK (experience >= 0),**

**email VARCHAR2(100) NOT NULL,**

**CONSTRAINT fk\_e2\_dept\_no FOREIGN KEY (dept\_no) REFERENCES DEPARTMENT(dept\_no) ON DELETE CASCADE,**

**CONSTRAINT ck\_email\_check CHECK (email LIKE '%@%.%')**

**);**

**-- Trigger for Age Restriction**

**CREATE OR REPLACE TRIGGER trg\_emp\_age\_check1**

**BEFORE INSERT OR UPDATE ON EMPLOYEE**

**FOR EACH ROW**

**BEGIN**

**IF ADD\_MONTHS(:NEW.birth\_date, 18 \* 12) > SYSDATE THEN**

**RAISE\_APPLICATION\_ERROR(-20001, 'Employee must be at least 18 years old.');**

**END IF;**

**END;**

**/**

### Create the PROJECT Table

* + proj\_id is **Primary Key**
  + emp\_id is **Foreign Key** referencing EMPLOYEE(emp\_id)

CREATE TABLE PROJECT (

proj\_id INT PRIMARY KEY,

type\_of\_project VARCHAR2(100) NOT NULL,

status VARCHAR2(50),

start\_date DATE NOT NULL,

emp\_id INT,

FOREIGN KEY (emp\_id) REFERENCES EMPLOYEE(emp\_id) ON DELETE CASCADE

);

### Insert Sample Data

#### Insert Departments

INSERT INTO DEPART (dept\_no, dept\_name, total\_employees, location) VALUES

(10, 'Accounts', 5, 'NY');

INSERT INTO DEPART (dept\_no, dept\_name, total\_employees, location) VALUES

(20, 'HR', 3, 'NY');

INSERT INTO DEPART (dept\_no, dept\_name, total\_employees, location) VALUES

(30, 'Production', 8, 'DL');

INSERT INTO DEPART (dept\_no, dept\_name, total\_employees, location) VALUES

(40, 'Sales', 6, 'NY');

INSERT INTO DEPART (dept\_no, dept\_name, total\_employees, location) VALUES

(50, 'CIVIL', 4, 'MU');

COMMIT;

#### Insert Employees

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(1, 'Alice', TO\_DATE('1990-05-12', 'YYYY-MM-DD'), 'Female', 10, '123 Main St', 'Manager', 60000, 10, 'alice@example.com');

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(2, 'Bob', TO\_DATE('1995-09-23', 'YYYY-MM-DD'), 'Male', 20, '456 Elm St', 'Clerk', 35000, 5, 'bob@example.com');

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(3, 'Charlie', TO\_DATE('1998-02-11', 'YYYY-MM-DD'), 'Male', 30, '789 Oak St', 'Analyst', 45000, 3, 'charlie@example.com');

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(4, 'David', TO\_DATE('2000-07-30', 'YYYY-MM-DD'), 'Male', 40, '321 Maple St', 'Coder', 40000, 2, 'david@example.com');

INSERT INTO EMPLOYEE (emp\_id, emp\_name, birth\_date, gender, dept\_no, address, designation, salary, experience, email) VALUES

(5, 'Eve', TO\_DATE('1992-10-15', 'YYYY-MM-DD'), 'Female', 50, '654 Pine St', 'Designer', 50000, 8, 'eve@example.com');

COMMIT;

### Delete Departments Where total\_employees < 1

DELETE FROM DEPART WHERE total\_employees < 1;

COMMIT;

### Display Female Employees in Descending Order

SELECT emp\_name, designation FROM EMPLOYEE WHERE gender = 'Female' ORDER BY emp\_name DESC;

### Display Employees Whose Name Starts and Ends with 'A'

SELECT emp\_name FROM EMPLOYEE WHERE emp\_name LIKE 'A%';

### Find Employee with Minimum Salary

SELECT emp\_name, salary FROM EMPLOYEE

WHERE salary = (SELECT MIN(salary) FROM EMPLOYEE);

### Increase Salary by 10% for CIVIL Department Employees

UPDATE EMPLOYEE

SET salary = salary \* 1.10

WHERE dept\_no = (SELECT dept\_no FROM DEPART WHERE dept\_name

= 'CIVIL');

COMMIT;

### Count Total Employees in MCA Department

SELECT COUNT(\*) FROM EMPLOYEE

WHERE dept\_no = (SELECT dept\_no FROM DEPART WHERE dept\_name

= 'MCA');

### List Employees Born in the Current Month

SELECT \* FROM EMPLOYEE4 WHERE EXTRACT(MONTH FROM birth\_date) = EXTRACT(MONTH FROM SYSDATE);

or

SELECT \* FROM EMPLOYEE4 WHERE TO\_CHAR(birth\_date, 'MM') = TO\_CHAR(SYSDATE, 'MM');

### Print Employee and Department Details in Sentence Format

SELECT E.emp\_name || ' works in department ''' || D.dept\_name || '''' AS Employee\_Department FROM EMPLOYEE4 E JOIN DEPART4 D ON E.dept\_no = D.dept\_no;

### List Freshers (Experience < 1 Year)

SELECT emp\_name FROM EMPLOYEE WHERE experience < 1;

### List Department-Wise Employees with >5 Years Experience

SELECT d.dept\_name, e.emp\_name FROM EMPLOYEE e

JOIN DEPART d ON e.dept\_no = d.dept\_no WHERE e.experience > 5;

## Part A - Question 5:

Create the database STUD and create given tables with all necessary constraints such as primary key, foreign key, unique key, not null and check constraints.

HOSTEL (HNO, HNAME, HADDR, TOTAL\_CAPACITY, WARDEN)

ROOM (HNO, RNO, RTYPE, LOCATION, NO\_OF\_STUDENTS, STATUS) CHARGES (HNO, RTYPE, CHARGES)

STUDENT (SID, SNAME, MOBILE-NO, GENDER, FACULTY, DEPT, CLASS, HNO, RNO)

FEES (SID, FDATE, FAMOUNT)

The STATUS field tells us whether the room is occupied or vacant. The charges represent the term fees to be paid half yearly. A student can pay either the annual fees at one time or the half yearly fees twice a year.

Insert proper data (at least 5 appropriate records) in all the tables.

1. Display the total number of rooms that are presently vacant.
2. Display number of students of each faculty and department wise staying in eachhostel.
3. Display hostels, which have at least one single-seated room.
4. Display the warden name and hostel address of students of Computer Sciencedepartment.
5. Display those hostel details where single seated or double- seated rooms are vacant.
6. Display details of hostels occupied by medical students.
7. Display hostels, which are totally occupied to its fullest capacity.
8. List details about students who are staying in the double-seated rooms of Chanakya Hostel.
9. Display the total number of students staying in each room type of each hostel.
10. Display details about students who have paid fees in the month of Nov. 2017.
11. For those hostels where total capacity is more than 300, display details of studentsstudying in Science faculty.
12. Display hostel details where there are at least 10 vacant rooms.
13. Display details of students who have still not paid fees.
14. Display those hostels where single-scated room is the costliest.