UNIVERSITY INSTITUTE of COMPUTING
Asia's Fastest Growing University

NAAC GRADE A+

Student Name: Sahil Hans **UID: 25MCI10088**
Branch: M.C.A(A.I & M.L) **Section/Group: MAM-1 A**
Semester: Second Sem **Date of Performance:12/01/2026**
Subject Name: TECHINCAL SKILLS **Subject Code:**

WORKSHEET 1.1

AIM: To design and implement a sample database system using DDL, DML, and DCL commands, including database creation, data manipulation, schema modification, and role-based access control to ensure data integrity and secure, read-only access for authorized users.

S/W Requirement: Oracle Database Express Edition and pgAdmin

OBJECTIVES:

To gain practical experience in implementing Data Definition Language (DDL), Data Manipulation Language (DML), and Data Control Language (DCL) operations in a real database environment. This will also include implementing role-based privileges to secure data.

Given:

An organization wants to design a **sample database system** to manage **Departments, Employees, and Projects**. The database must ensure **data integrity**, **controlled access**, and **proper privilege management** for different users.

1. **Database Design**

Create multiple tables such as **Department**, **Employee**, and **Project**.

Define appropriate **PRIMARY KEY** and **FOREIGN KEY** constraints.

Enforce **NOT NULL**, **UNIQUE**, and **CHECK** constraints where necessary.

Querry:

```
CREATE TABLE Departments (
    dept_id INT PRIMARY KEY,
    dept_name VARCHAR(100) NOT NULL UNIQUE,
    location VARCHAR(100) NOT NULL
);
```

```
CREATE TABLE Employees (
    emp_id INT PRIMARY KEY,
    fullname VARCHAR(100) NOT NULL,
    salary DECIMAL(10,2) NOT NULL,
    dept_id INT NOT NULL
        REFERENCES Departments(dept_id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE,
    role VARCHAR(50) NOT NULL
);
CREATE TABLE Projects (
    project_id INT PRIMARY KEY,
    proj_name VARCHAR(150) NOT NULL,
    start_date DATE NOT NULL,
    end_date DATE,
    dept_id INT NOT NULL
        REFERENCES Departments(dept_id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE
);
```

2. Data Manipulation

Insert sample records into all tables.

Querry:

```
INSERT INTO Projects (project_id, proj_name, start_date, end_date, dept_id) VALUES
(201, 'Payroll System', '2025-01-10', '2025-06-30', 1),
(202, 'E-Commerce App', '2025-02-01', '2025-12-31', 2),
(203, 'Budget Analysis', '2025-03-15', NULL, 3);
```

Data Output Messages Notifications

	project_id [PK] integer	proj_name character varying (150)	start_date date	end_date date	dept_id integer
1	201	Payroll System	2025-01-10	2025-06-30	1
2	202	E-Commerce App	2025-02-01	2025-12-31	2
3	203	Budget Analysis	2025-03-15	[null]	3

INSERT INTO Employees (emp_id, fullname, salary, dept_id, role) VALUES

(101, 'Amit Sharma', 50000, 1, 'HR Executive'),

(102, 'Neha Verma', 70000, 2, 'Software Engineer'),

(103, 'Rohit Mehta', 90000, 2, 'Project Manager'),

(104, 'Priya Singh', 60000, 3, 'Accountant');

Data Output Messages Notifications

	emp_id [PK] integer	fullname character varying (100)	salary numeric (10,2)	dept_id integer	role character varying (50)
1	101	Amit Sharma	50000.00	1	HR Executive
2	102	Neha Verma	70000.00	2	Software Engineer
3	103	Rohit Mehta	90000.00	2	Project Manager
4	104	Priya Singh	60000.00	3	Accountant

INSERT INTO Departments (dept_id, dept_name, location) VALUES

(1, 'HR', 'Mumbai'),

(2, 'IT', 'Bangalore'),

(3, 'Finance', 'Delhi');

Data Output Messages Notifications

	dept_id [PK] integer	dept_name character varying (100)	location character varying (100)
1	1	HR	Mumbai
2	2	IT	Bangalore
3	3	Finance	Delhi

Perform **UPDATE** operations to modify existing records.

Querry:

Change an employee's department

UPDATE Employees

SET dept_id = 3

WHERE emp_id = 101;

4	101	Amit Sharma	50000.00	3	HR Executive

Extend a project deadline

UPDATE Projects

SET end_date = '2026-03-31'

WHERE project_id = 202;

3	202	E-Commerce App	2025-02-01	2026-03-31	2

Perform **DELETE** operations while maintaining referential integrity.

Querry:

DELETE FROM Employees

WHERE dept_id = 2;

	emp_id [PK] integer	fullname character varying (100)	salary numeric (10,2)	dept_id integer	role character varying (50)
1	104	Priya Singh	60000.00	3	Accountant
2	101	Amit Sharma	50000.00	3	HR Executive

DELETE FROM Projects

WHERE dept_id = 2;

	project_id [PK] integer	proj_name character varying (150)	start_date date	end_date date	dept_id integer
1	201	Payroll System	2025-01-10	2025-06-30	1
2	203	Budget Analysis	2025-03-15	[null]	3

DELETE FROM Departments

WHERE dept_id = 2;

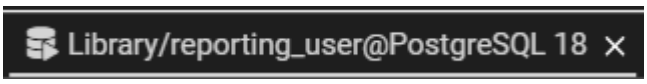
	dept_id [PK] integer	dept_name character varying (100)	location character varying (100)
1	1	HR	Mumbai
2	3	Finance	Delhi

3. **Access Control & Security**

Create a **role/user** for a reporting staff member.

Querry:

CREATE ROLE reporting_user LOGIN PASSWORD 'report123';



Grant **ONLY SELECT privilege** on required tables to this role/user.

Querry:

GRANT SELECT ON Departments TO reporting_user;

GRANT SELECT ON Employees TO reporting_user;

GRANT SELECT ON Projects TO reporting_user;

Explicitly **REVOKE CREATE privilege** so that the user cannot create any database objects.

Querry:

REVOKE CREATE ON SCHEMA public FROM reporting_user;

Ensure the user has **read-only access** to the database.

Querry:

REVOKE INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA public FROM reporting_user;

4. **Schema Modification**

Use **ALTER TABLE** to add or modify a column.

Querry:

ALTER TABLE Employees

ADD COLUMN email VARCHAR(100);

	emp_id [PK] integer	fullname character varying (100)	salary numeric (10,2)	dept_id integer	role character varying (50)	email character varying (100)
1	104	Priya Singh	60000.00	3	Accountant	[null]
2	101	Amit Sharma	50000.00	3	HR Executive	[null]

Drop a table that is no longer required using **DROP TABLE**.

Querry:

DROP TABLE Projects;

Learning Outcomes:

1. Understood the basics of **relational database design** using tables, keys, and relationships.
2. Learned to apply **primary key and foreign key constraints** to maintain data integrity.
3. Gained hands-on experience with **INSERT, UPDATE, and DELETE** operations safely.
4. Understood how **roles and privileges** control access to database objects.
5. Learned to use **GRANT and REVOKE** for implementing **read-only users**.
6. Practiced **ALTER TABLE and DROP TABLE** for managing database changes.