UNIVERSITY INSTITUTE of COMPUTING
Asia's Fastest Growing University

NAAC GRADE A+ ACCREDITED UNIVERSITY

Worksheet-1

Name: TANUJ JOSHI
UID: 25MCA20081
Branch: MCA General
Section/Group: 25MCA_KAR-1
Semester: II
Date of Performance: 15-01-26
Subject Name: Technical Training
Subject Code: 25CAP-652

Aim of the Session:

The aim of this practical session is to implement and analyze SQL commands including Data Definition Language (DDL), Data Manipulation Language (DML), Data Query Language (DQL), and Data Control Language (DCL) in PostgreSQL.
This practical helps in understanding how database objects are created, data is manipulated, retrieved, and how user access control is managed in a relational database system.

Tools Used:

PostgreSQL — Powerful open-source relational database for storing and managing data efficiently

Objective of the Session:

The objectives of this practical session are:

- To create and modify database tables using DDL commands
- To apply constraints such as PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK, and DEFAULT
- To insert, update, and delete records using DML commands
- To retrieve records using DQL commands
- To create database roles and manage permissions using DCL commands
- To understand role-based access control in PostgreSQL

Practical / Experiment Steps:

Step 1: Database and Table Creation (DDL)

- Create department, employee, and project tables
- Apply necessary constraints and foreign key relationships

Step 2: Table Modification

- Add and remove columns using ALTER TABLE
- Apply default values to columns

Step 3: Data Manipulation (DML)

- Insert records into tables
- Update existing records
- Delete unwanted records

Step 4: Data Retrieval (DQL)

- Retrieve records using SELECT statements

Step 5: Data Control (DCL)

- Create a database role
- Grant and revoke permissions
- Verify access using restricted user

Procedure of the Practical:

1. Start the system and log in to the computer.
2. Open the PostgreSQL client tool (psql / pgAdmin).
3. Create the **Department table** with primary key, unique constraint, and check constraint.
   CREATE TABLE department(
   department_id INT PRIMARY KEY,
   department_name VARCHAR(20) NOT NULL UNIQUE,
   salary FLOAT CHECK(salary>=0)
   );

Data Output | Messages | Notifications

SQL

	department_id [PK] integer	department_name character varying (20)	salary double precision
1	101	Manager	90000
2	102	HR	70000
3	103	Employee	50000

4. Create the **Employee table** with foreign key reference and date constraints.

```
CREATE TABLE employee(
employee_id INT PRIMARY KEY,
employee_name VARCHAR(20) NOT NULL,
department_id INT NOT NULL REFERENCES department(department_id),
employee_contact VARCHAR(20),
join_date DATE NOT NULL,
end_date DATE CHECK(end_date>=join_date)
);
```

Data Output | Messages | Notifications

	employee_id [PK] integer	employee_name character varying (20)	department_id integer	employee_contact character varying (20)	join_date date	end_date date	status character varying (20)
1	1	Rahul	101	8888888888	2001-04-12	2010-07-13	active
2	2	Anuj	102	7777777777	2003-06-10	2004-05-11	active
3	4	Naman	103	5555555555	2006-06-25	2009-08-11	active
4	5	Karan	103	4444444444	2006-03-12	2009-05-11	active

5. Modify the employee table using ALTER TABLE.

```
ALTER TABLE employee ADD work_location VARCHAR(20);
ALTER TABLE employee DROP work_location;
ALTER TABLE employee ADD status VARCHAR(20) DEFAULT 'active';
```

6. Create the **Project table** with foreign key reference.

```
CREATE TABLE project(
project_id INT PRIMARY KEY,
project_name VARCHAR(20) NOT NULL UNIQUE,
department_id INT NOT NULL REFERENCES department(department_id),
start_date DATE NOT NULL,
end_date DATE CHECK(end_date>=start_date)
);
```

Data Output | Messages | Notifications

	project_id [PK] integer	project_name character varying (20)	department_id integer	start_date date	end_date date
1	11	P1	103	2025-08-14	2025-09-14
2	12	P2	103	2025-08-14	2025-08-30

7. Insert records into tables and perform update and delete operations.

```
INSERT INTO department
```

```
VALUES
(101,'Manager',90000),
(102,'HR',70000),
(103,'EMPLOYEE',50000);

INSERT INTO department
VALUES
(104,'DEVELOPER',-30000);

INSERT INTO department
VALUES
(104,'DEVELOPER',30000);

INSERT INTO employee
VALUES
(1,'Rahul',101,8888888888,'2001-04-12','2010-07-13'),
(2,'Anuj',102,7777777777,'2003-06-10','2004-05-11'),
(3,'Aman',103,6666666666,'2006-05-20','2009-09-11'),
(4,'Naman',103,5555555555,'2006-06-25','2009-08-11'),
(5,'Karan',103,4444444444,'2006-03-12','2009-05-11');

INSERT INTO project
VALUES
(11,'P1',103,'2025-08-14','2025-09-14'),
(12,'P2',103,'2025-08-14','2025-08-30');

UPDATE department SET department_name='Employee' WHERE department_id=103;
DELETE FROM department WHERE department_id=104;
DELETE FROM employee WHERE employee_id=3;
```

8. Retrieve records using SELECT queries.

```
SELECT * FROM department;
SELECT * FROM employee;
SELECT * FROM project;
```

9. Create a role and manage permissions using DCL commands.

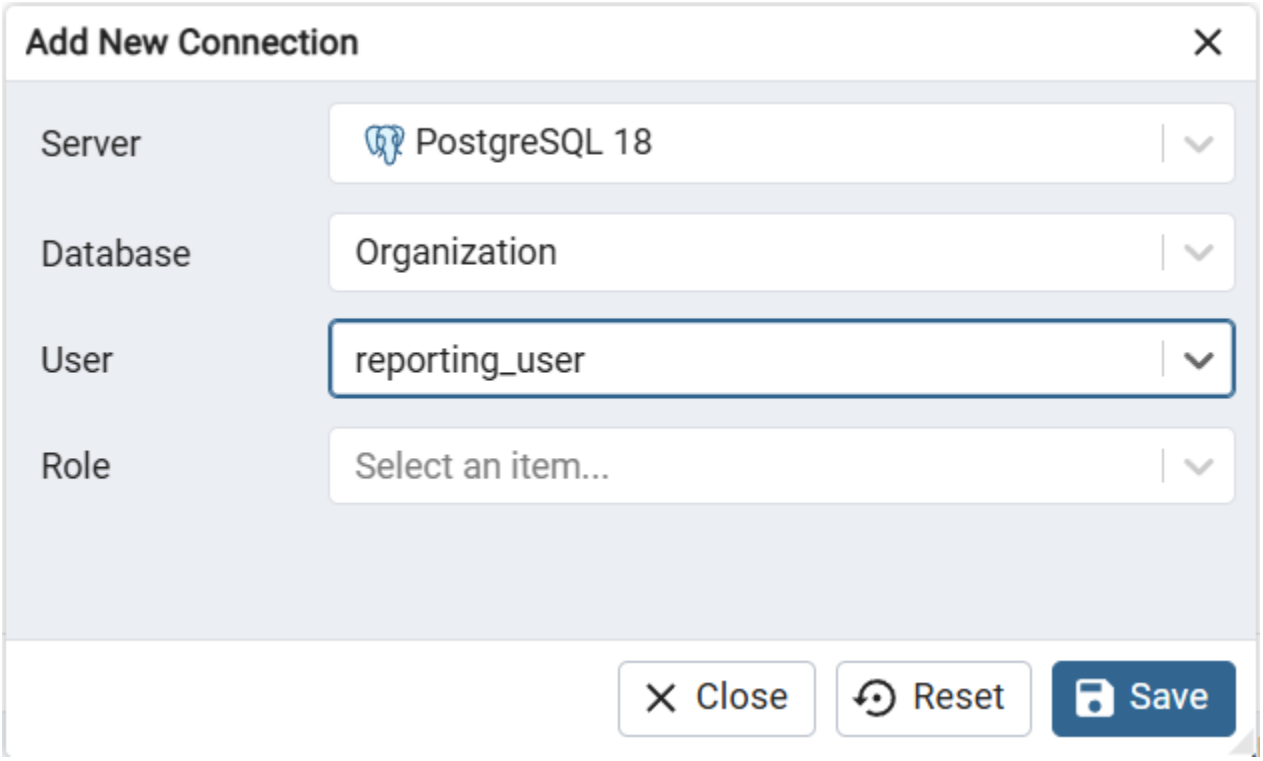
```
CREATE ROLE reporting_user
```

```
LOGIN
PASSWORD
'user123';

GRANT SELECT ON department TO reporting_user;
REVOKE SELECT ON department FROM reporting_user;
GRANT SELECT ON project TO reporting_user;
REVOKE CREATE ON SCHEMA PUBLIC FROM reporting_user;
```

10. Verify data access using the reporting user.

```
SELECT * FROM project;
```



I/O Analysis:

Input Provided

- SQL DDL commands (CREATE, ALTER)
- SQL DML commands (INSERT, UPDATE, DELETE)
- SQL DQL commands (SELECT)
- SQL DCL commands (CREATE ROLE, GRANT, REVOKE)

Output Generated

- Tables created successfully with constraints
- Records inserted, updated, and deleted
- Data retrieved from all tables
- Controlled access to tables using role-based permissions

Learning Outcomes:

- Understanding of DDL, DML, DQL, and DCL commands
- Knowledge of constraints and referential integrity
- Practical experience in modifying database schemas
- Ability to manage database users and permissions
- Improved confidence in handling real-world SQL database operations