Session 2

1 - categories of SQL commands according to ANSI SQL standards

1. **DDL (Data Definition Language)** – Commands that define or modify the structure of database objects like tables, databases, etc.
   - **Create**: Make new tables or databases
   - **Drop**: Delete tables or databases
   - **Alter**: Modify structure of tables
   - **Truncate**: Remove all rows from a table quickly
2. **DML (Data Manipulation Language)** – Commands to manipulate the **data inside tables**.
   - **Insert**: Add new rows
   - **Update**: Modify existing rows
   - **Delete**: Remove rows
3. **DCL (Data Control Language)** – Commands for **permissions and access control**.
   - **Grant**: Give permissions to users
   - **Revoke**: Remove permissions
4. **TCL (Transaction Control Language)** – Commands to manage **transactions** and ensure database consistency.
   - **Commit**: Save changes permanently
   - **Rollback**: Undo changes
   - **Savepoint**: Mark a point to roll back to within a transaction
5. **DQL (Data Query Language)** – Commands to **query the data**.
   - **Select**: Retrieve data from tables

---

2 - Difference Between DROP,DELETE and TRUNCATE

Command	What it does	Can it remove the table itself?	Can it have a WHERE clause?	Transaction-safe / Rollback?	Speed	Notes
DROP	Removes a table (or database) completely, including all its data and structure (columns, indexes, constraints).	**Yes**	No	Cannot rollback in some systems (depends on DBMS)	Fast	After dropping, the table no longer exists. You'd need to recreate it to use it again.
DELETE	Removes rows from a table, based on a condition if specified.	**No**	**Yes**	Yes, can be rolled back if inside a transaction	Slower for large tables	Fires triggers; can delete specific rows. If no `WHERE`, deletes all rows but keeps table structure.
TRUNCATE	Removes all rows from a table, **but keeps the table structure**.	No	**No**	Sometimes cannot rollback (DBMS-dependent)	Very fast	Resets auto-increment counters in some DBMS; usually doesn't fire triggers.

3 – Steps to Download PostgreSQL (Windows)

✅ 1. Download & Install PostgreSQL (includes pgAdmin optional)

Step 1 — Download

Go to the official website:
https://www.postgresql.org/download/windows/ ↗

Click **Download the installer** (EDB installer).

Step 2 — Run Installer

During setup, you will see:

- **PostgreSQL Server**
- **pgAdmin 4**
- StackBuilder
  ✔ Keep all selected.

Step 3 — Set the Password

You will be asked to set a password for the default PostgreSQL superuser:

```makefile
Username: postgres
Password: <your_password>
```

👉 **Remember this**, pgAdmin will use it.

Step 4 — Complete Installation

After installation, PostgreSQL service will automatically run in the background.

✅ 2. Open pgAdmin & Add Your PostgreSQL Server

Step 1 — Open pgAdmin

Search in Start Menu:

```nginx
pgAdmin 4
```

It will open in your browser.

Step 2 — Set a Master Password

This is for pgAdmin only.
It **does NOT** have to match the PostgreSQL password.

Step 3 — Connect pgAdmin to PostgreSQL

Left side → **Servers (1)** → you will see:

```scss
PostgreSQL 16 (or 15)
```

Click it → pgAdmin will ask for password.

Enter the password you created during PostgreSQL install:

```pgsql
postgres user password
```

Now pgAdmin will connect to your local PostgreSQL server.

✔ Now PostgreSQL + pgAdmin Are Linked!

You can now create databases:

Right-click **Databases → Create → Database**

🔧 3. (Optional) How to Check PostgreSQL from Terminal (psql)

Open command prompt:

```nginx
psql -U postgres
```

If you installed PostgreSQL 16, the path is automatically added to your system.

If not, you can do:

```arduino
"C:\Program Files\PostgreSQL\16\bin\psql.exe" -U postgres
```

---

4 - Steps on mac

🍏 1. Install PostgreSQL on macOS

You have **two options**:
✔ Using the **official installer (EASIEST)**
✔ Using **Homebrew** (developers prefer)

✅ Option A — Install using Official Installer (recommended)

Step 1 — Download

Visit:
https://www.postgresql.org/download/macosx/ ↗

Download the **EDB macOS Installer**.

Step 2 — Run the Installer

It includes:

- PostgreSQL Server
- pgAdmin (optional)
- StackBuilder

✔ Keep everything selected.

Step 3 — Set the postgres Password

When prompted:

```makefile
Username: postgres
Password: <your_password>
```

Step 4 — Finish Installation

PostgreSQL server starts automatically.

✅ Option B — Install PostgreSQL using Homebrew

If you prefer Homebrew:

Install PostgreSQL:

```bash
brew install postgresql
```

Start PostgreSQL service:

```bash
brew services start postgresql
```

Check version:

```bash
psql --version
```

This method **does NOT install pgAdmin**, so install it separately (next section).

🍏 2. Install pgAdmin on macOS

Option A — Download the pgAdmin App

Go to:
https://www.pgadmin.org/download/pgadmin-4-macos/ ↗

Download → open `.dmg` → drag **pgAdmin 4** to Applications.

Option B — Using Homebrew

```bash
brew install --cask pgadmin4
```

🍏 3. Open pgAdmin & Link it to PostgreSQL

Step 1 — Open pgAdmin

Open from **Applications** → pgAdmin 4
or:

```bash
open -a pgAdmin\ 4
```

Step 2 — Set a Master Password

This is only for pgAdmin (not PostgreSQL).

Step 3 — Add PostgreSQL Server

Left sidebar → Right-click **Servers → Create → Server**

Then:

⚙️ **GENERAL Tab**

- Name: `Local PostgreSQL`

⚙️ **CONNECTION Tab**

- Host: `localhost`
- Port: `5432` (default)
- Username: `postgres`
- Password: (the password you set during installation)

Click **Save**.

🎉 pgAdmin is now connected to your PostgreSQL server.

🍏 4. Test Connection Using Terminal (psql)

If installed using official installer:

```bash
/Library/PostgreSQL/16/bin/psql -U postgres
```

If installed using Homebrew:

```bash
psql -U postgres
```

If a database prompt appears:

```makefile
postgres=#
```

You're connected.

🍏 5. Common macOS Issues

Issue: "psql: command not found"

Your PATH is missing PostgreSQL.

Fix by adding this to your shell:

```bash
echo 'export PATH="/Library/PostgreSQL/16/bin:$PATH"' >> ~/.zshrc
source ~/.zshrc
```

Issue: pgAdmin can't connect

Check PostgreSQL service:

Homebrew installation

```bash
brew services list
brew services start postgresql
```

EDB installer installation

```bash
sudo /Library/PostgreSQL/16/scripts/pg_ctl start
```
