📘 Module 14: Database Basics

🔷 What is SQL?

SQL stands for **Structured Query Language**.

It is a standard programming language for managing and interacting with **relational databases**.

📌 Key Capabilities:

- Create and modify databases and tables
- Insert, update, and delete records
- Retrieve and filter data
- Set user permissions and ensure data security
- Aggregate, sort, and organise data for analysis

💡 SQL Trivia & Facts

Trivia	Details
Pronunciation	Often pronounced as **“Sequel”**
Universal	Used across **all major databases** (MySQL, SQL Server, Oracle, PostgreSQL)
Paid + Free	SQL engines may be **open-source (e.g., MySQL)** or **paid (e.g., Oracle)**
Security	Supports **role-based access**, **user permissions**, and **data encryption**
Compatibility	Works with popular tools like **Python**, **Excel**, **Power BI**, and **Java**
Declarative	You tell **what** to do, not **how** (unlike most programming languages)

👥 Who Uses SQL?

SQL is used by a wide range of professionals and industries:

User Type	Role
📊 **Data Analyst**	Uses SQL to extract and analyze data for insights
💼 **Business Analyst**	Generates reports and dashboards using SQL
👩‍💻 **Software Developer**	Integrates SQL into apps for storing and accessing data
🛠️ **Database Administrator (DBA)**	Manages database structure, users, backups, and performance
📈 **Data Scientist**	Uses SQL to prepare and filter data before applying ML models
🧪 **Tester / QA Engineer**	Verifies data correctness and performs backend testing
📚 **Students / Learners**	Learn SQL to build a strong foundation in data handling

🗄️ What is a Database?

A **database** is an organised collection of **data** that can be easily accessed, managed, and updated.

Databases help store information digitally for quick retrieval and processing.

🔷 **Example**:

A school's database might store:

- Students' names and IDs
- Subjects and marks
- Attendance records

🔗 What is a Relational Database?

A **Relational Database** stores data in the form of **tables**, where data is related to other data using **keys**.

Each table represents a different **entity**, and relationships are defined using **Primary** and **Foreign Keys**.

🔷 Example:

- `Students` Table
- `Marks` Table
- Linked by `StudentID`

✅ Data is:

- Organized
- Easy to update
- Non-redundant

📋 Tables: Structure & Terminology

Term	Description	Example
Table	Grid to store related data	`Students` , `Employees` , `Orders`
Attribute (Column)	Field name	`Name` , `Gender` , `Age`
Record (Row)	One entry of data	`('Piya','Female', 25)`
Cell	Intersection of row and column	`25` in column `Age` for `Piya`

📌 **Each table has a unique name**

📌 **Each column has a data type** (e.g., INT, VARCHAR, DATE)

Tables

Name	Gender	Age	State	Country
Piya	Female	25	Bangalore	India
Tarun	Male	32	Delhi	India
Harsh	Male	27	Kolkata	India

🧠 What is a DBMS?

DBMS = **Database Management System**

A **DBMS** is software used to store, retrieve, and manage databases efficiently.

Feature	Description
Storage	Manages how and where data is saved
Querying	Allows users to ask questions using SQL
Security	Controls access (users, roles, privileges)
Backup/Recovery	Ensures data is safe in case of failures
Multi-user Access	Many users can work at the same time

🔍 Examples of Popular DBMS

DBMS	Type	Notes
MySQL	Open-source	Popular for web apps
SQL Server	Paid	Microsoft-developed
PostgreSQL	Open-source	Advanced & feature-rich
Oracle	Paid	Enterprise-grade
SQLite	Lightweight	Used in mobile apps

📂 Types of SQL Commands

SQL commands are grouped into **5 main categories**, based on their purpose. Each category serves different roles in the database system.

🔧 1. DDL – Data Definition Language

DDL commands are used to **define and modify the structure** of database objects like tables, schemas, indexes, etc.

🔷 Common DDL Commands:

Command	Description
`CREATE`	Create a new table or database
`ALTER`	Modify table structure
`DROP`	Delete a table or database
`TRUNCATE`	Remove all records (structure remains)

👷 Used by:

- **Database Architects**
- **Designers**
- **DBAs (Database Admins)** during initial design and schema changes

---

🛠️ 2. DML – Data Manipulation Language

DML commands are used to **manipulate data** stored in tables (insert, update, delete records).

🔷 Common DML Commands:

Command	Description
`INSERT`	Add new data
`UPDATE`	Modify existing data
`DELETE`	Remove data

👤 Used by:

- **Data Analysts**
- **Application Developers**
- **Data Entry Operators**

🔎 3. DQL – Data Query Language

DQL is used to **query and retrieve** data from databases.

🔷 Common DQL Command:

Command	Description
`SELECT`	Fetch data from tables

👤 Used by:

- **Data Analysts**
- **BI Professionals**
- **Students**
- **Data Scientists**

🔐 4. DCL – Data Control Language

DCL is used to **control access and permissions** in a database.

🔷 Common DCL Commands:

Command	Description
`GRANT`	Give access/privileges to users
`REVOKE`	Remove access/privileges

👤 Used by:

- **Database Administrators (DBAs)**
- **Security Engineers**

🔄 5. TCL – Transaction Control Language

TCL commands are used to **manage changes** made by DML statements. They help maintain **data integrity** and allow **rollback** in case of errors.

🔷 Common TCL Commands:

Command	Description
`COMMIT`	Save changes
`ROLLBACK`	Undo changes
`SAVEPOINT`	Set a point to rollback to
`BEGIN` / `END`	Start/close a transaction block

👤 Used by:

- **Application Developers**
- **Backend Engineers**
- **DBAs**

🧭 Summary Table

Type	Full Form	Purpose	Users
DDL	Data Definition Language	Define/modify schema	DB Designers, Architects, DBAs
DML	Data Manipulation Language	Add/change/delete data	Developers, Analysts
DQL	Data Query Language	Fetch/read data	Analysts, Students, Scientists
DCL	Data Control Language	Control access/permissions	DBAs, Security Teams
TCL	Transaction Control Language	Manage transaction consistency	Developers, DBAs

⚡ Quick Tips – SQL & Database Basics

- ✅ **Start with** `SELECT` – It's the most common and useful SQL command.
- 🔍 Use `WHERE` to **filter specific rows** from a table.
- 🔗 Learn `JOIN` early – it's essential for working with multiple tables.
- 🧱 Always define **Primary Keys** to ensure data uniqueness.
- 🗃️ Use `CREATE TABLE` carefully – design your schema before inserting data.
- ✏️ Use `UPDATE` and `DELETE` with a `WHERE` clause to **avoid accidental data loss**.
- 🔐 Use `GRANT` and `REVOKE` to **control who can access or modify data**.
- 🔄 Use `ROLLBACK` if something goes wrong in a transaction – **TCL saves your data**.
- 📂 Keep your **DBMS backed up** regularly to prevent data loss.
- 🧠 Practice DDL, DML, and DQL together for complete understanding.