SQL
PART 01

Jayathma Chathurangani
ejc@ucsc.cmb.ac.lk

LESSON SHOULD COVER..

- Structured Query Language (SQL)
  - Introduction to SQL standards
- Creating SQL Databases and Tables
- Selecting Data
- Data Insertion, Updating and Deletion
- Data Views
  - What is a view?
  - Views using SQL
    - Creating view
    - Dropping view
  - View Updatability and WITH CHECK OPTION in SQL
  - View Materialization
- Stored Procedures
- Triggers

OUTLINE

- ✓ Introduction to SQL
- ✓ Relational Databases
- ✓ SQL Basics

1

INTRODUCTION

1.1 BACKGROUND?

- The Structured Query Language (SQL), widely recognized as SQL, originated from the evolution of the relational model.
- In 1986, the American National Standards Institute (ANSI) established a standard for SQL, which was later internationally recognized by the International Organization for Standardization (ISO) in 1987.
- Today, SQL is supported by over a hundred database management systems (DBMSs) across a variety of hardware platforms, from personal computers to mainframes. Over time, SQL has become the dominant language for relational databases.
- It is now both a formal and de facto international standard for creating, managing, and manipulating relational databases (ISO, 1992, 2011a).

1.2 OBJECTIVES OF A DATABASE LANGUAGE?

- Ideally, a database language should allow a user to:
  - create the database and relation structures;
  - perform basic data management tasks, such as the insertion, modification, and deletion of data from the relations;
  - perform both simple and complex queries.
- A database language must perform these tasks with minimal user effort, and its command structure and syntax must be relatively easy to learn.
- Finally, **the language must be portable**; that is, it must conform to some recognized standard so that we can use the same command structure and syntax when we move from one DBMS to another.

1.3 WHAT IS SQL?

- SQL, short for Structured Query Language, is designed to meet the fundamental requirements of a database language. It is a transform-oriented language, meaning it uses relations to process input and produce the desired output.
- The ISO SQL standard defines two primary components:
- **Data Definition Language (DDL):** Used for creating and managing the database structure while controlling access to the data.
- **Data Manipulation Language (DML):** Focused on retrieving and modifying data within the database.

- Additionally, SQL includes:
- **Data Control Language (DCL):** Manages user permissions and access.
- **Transaction Control Language (TCL):** Ensures the consistency and reliability of database transactions.

- SQL can be utilized in two distinct ways:
- **Interactive Mode:** Users directly enter SQL statements through a terminal interface.
- **Embedded Mode:** SQL commands are integrated into procedural programming languages.

1.4 SQL IS A RELATIVELY EASY LANGUAGE TO LEARN..

- SQL is a nonprocedural language, meaning you specify the data you need without defining how to retrieve it. In essence, SQL abstracts the underlying data access methods.
- Like most modern programming languages, SQL is free-format, allowing flexibility in how commands ar structured, as they don't need to follow rigid formatting rules.
- SQL commands use intuitive English-like syntax, such as:
  - CREATE TABLE for defining tables.
  - INSERT for adding records.
  - SELECT for querying data.
- Examples:

```
CREATE TABLE Staff (staffNo VARCHAR(5), lName VARCHAR(15), salary DECIMAL(7,2));

INSERT INTO Staff VALUES ('SG16', 'Brown', 8300);

SELECT staffNo, lName, salary FROM Staff WHERE salary > 10000;
```

- SQL is versatile and can be used by various professionals, including:
  - **Database Administrators (DBA):** For managing and maintaining databases.
  - **Application Developers:** For building data-driven applications.
  - **Management Personnel:** For retrieving critical business insights.
  - **End Users:** For ad-hoc data queries.

2

RDBMS

2.1 DBMS VS. RDBMS

Feature	DBMS	RDBMS
Data Organization	Flexible data organization (structured, semi-structured, unstructured)	Structured data (tables, rows, columns)
Complexity	Lower complexity	Higher complexity
Support for Relationships	No inherent support for relationships	Strong relationships defined through keys
ACID Compliance	Not typically ACID compliant	Fully ACID compliant ensuring reliable transactions
Use Cases	Document management, email archiving, unstructured data	Transactional databases, financial records, customer management, complex analysis

ACID PROPERTIES IN RELATIONAL DATABASES

- **Atomicity:** It ensures that a transaction is treated as a single unit of work, and either all the steps in a transaction must be completed successfully, or none of them must be executed at all. In other words, if a transaction fails, it is rolled back to the original state.

- **Consistency:** It ensures that the data is always consistent before and after a transaction. In other words, a transaction cannot leave the database in an inconsistent state.

- **Isolation:** It ensures that each transaction is isolated from other concurrent transactions. This means that a transaction sees the state of the database before any concurrent transaction has modified it. Transactions does not effect each other.

- **Durability:** It ensures that the changes made by a transaction are permanent and will not be lost even if the system crashes. That is successfully written data will not be lost.

BASE PROPERTIES IN NOSQL DATABASES

- **Basically Available:** It means that the system is always available, even if there is a network partition or a node failure.
- **Soft state:** It means that the state of the system can change over time, even without input.
- **Eventually Consistent:** It means that the system will eventually become consistent, although there may be some inconsistency in the meantime.

The ACID model is ideal for systems that require transactional integrity (Eg- Banking systems, financial systems, airline reservation systems), while the BASE model is suitable for systems that require high availability and scalability (Eg-Social media platforms, e-commerce websites).

Understanding the differences between these two models is essential for building robust and reliable database systems.

2.2 WHAT IS RDBMS?

- A **relational database management system (RDBMS**) is a collection of programs and capabilities that enable IT teams and others to create, update, administer and otherwise interact with a relational database.
- In other words **A relational database management system (RDBMS) is a program used to create, update, and manage relational databases**.
- RDBMSes store data in the form of tables, with most commercial relational database management systems using Structured Query Language (SQL) to access the database.
- However, since SQL was invented after the initial development of the relational model, it is not necessary for RDBMS use.

2.3 EXAMPLES OF RDBMS?

- Some of the most well-known RDBMSs include

MySQL, PostgreSQL, MariaDB, Microsoft SQL Server, and Oracle Database.

- Cloud-based relational databases include

Cloud SQL, Cloud Spanner and AlloyDB.

Amazon Aurora

3

SQL BASICS

3.1 WRITING SQL COMMANDS

- An SQL statement is composed of the following elements:
  - **Reserved Words:** These are predefined parts of the SQL language with fixed meanings. They must be spelled precisely as defined and cannot span multiple lines.
  - **User-Defined Words:** These are custom identifiers created by the user, adhering to specific syntax rules. They represent database objects like tables, columns, views, and indexes.

- Statement Terminator: SQL statements typically end with a statement terminator, often a semicolon (;), to signify the end of the command.

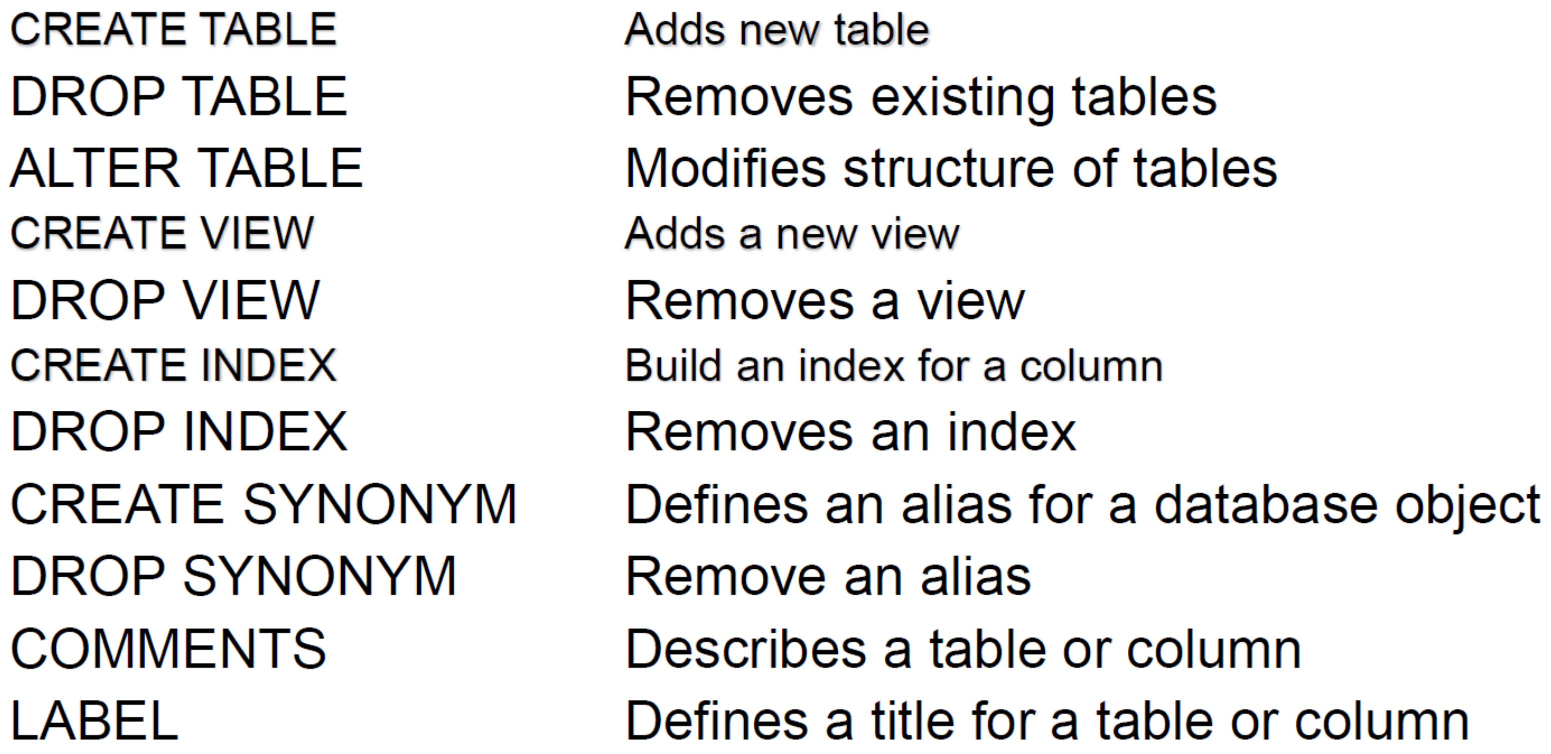
- Case Sensitivity: Most SQL components are case-insensitive, meaning commands like SELECT and select are treated the same. However, literal character data (e.g., Smith) must match the database exactly.

Example: Searching for the name “Colombo” requires typing it as stored in the database.

- Readability: hile SQL is free-format, meaning commands don’t need specific alignment, proper indentation and formatting improve readability and maintainability of SQL statements.

3.2 DATA DEFINITION LANGUAGE (DDL)

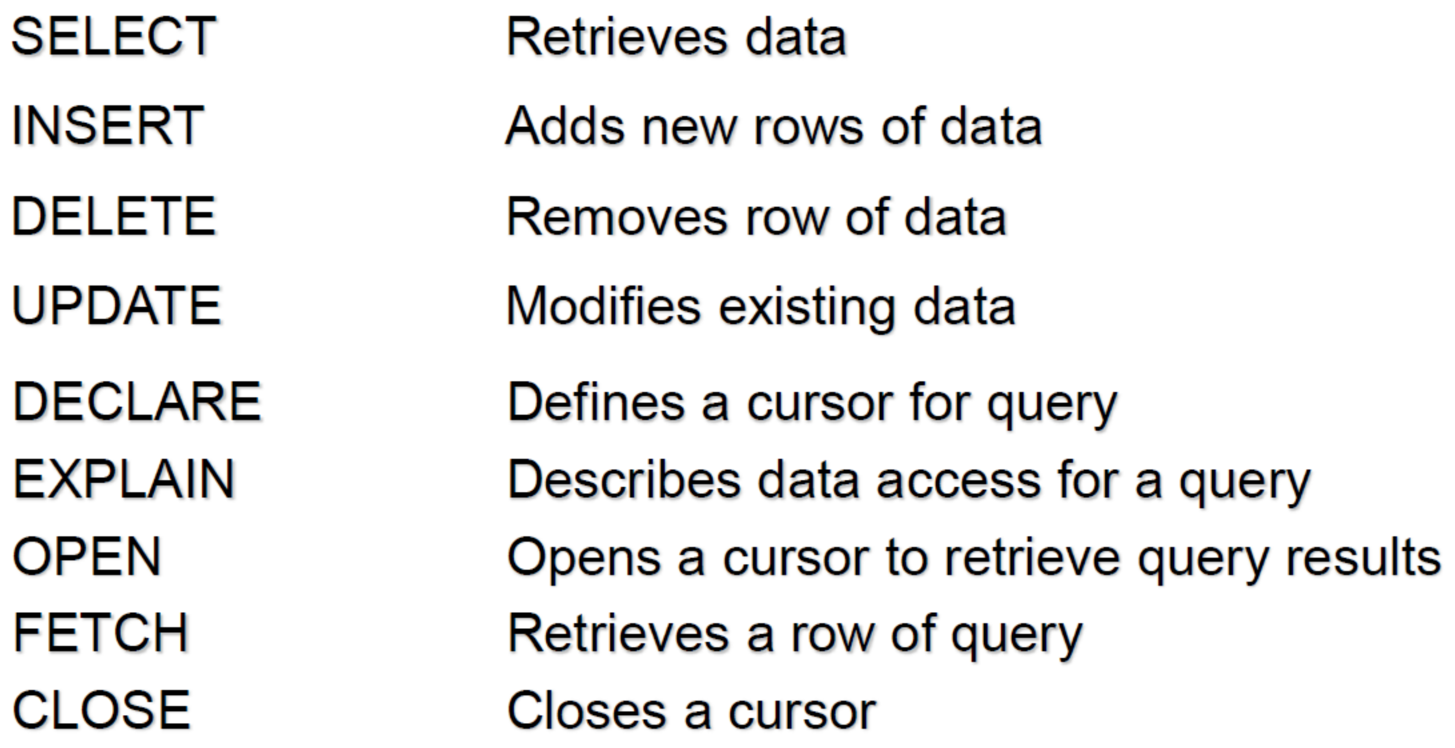
- DDL defines the database: Physical Design.
- A set of statements that allow the user to define or modify data structures and objects such as data tables
- Example:



CREATE TABLE	Adds new table
DROP TABLE	Removes existing tables
ALTER TABLE	Modifies structure of tables
CREATE VIEW	Adds a new view
DROP VIEW	Removes a view
CREATE INDEX	Build an index for a column
DROP INDEX	Removes an index
CREATE SYNONYM	Defines an alias for a database object
DROP SYNONYM	Remove an alias
COMMENTS	Describes a table or column
LABEL	Defines a title for a table or column

3.4 DATA MANIPULATION LANGUAGE (DML)

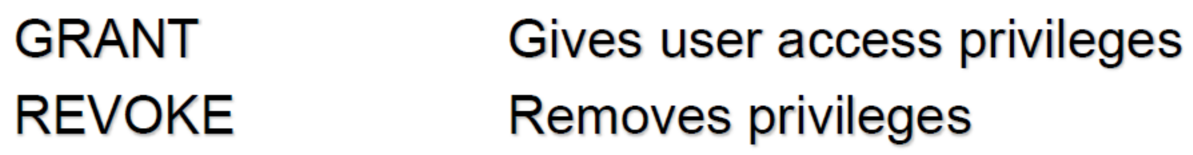
- DML load the database: Implementation
- Through a set of operations, this SQL syntax allows you to manipulate existing data objects
- Example:



SELECT	Retrieves data
INSERT	Adds new rows of data
DELETE	Removes row of data
UPDATE	Modifies existing data
DECLARE	Defines a cursor for query
EXPLAIN	Describes data access for a query
OPEN	Opens a cursor to retrieve query results
FETCH	Retrieves a row of query
CLOSE	Closes a cursor

3.5 DATA CONTROL LANGUAGE (DCL)

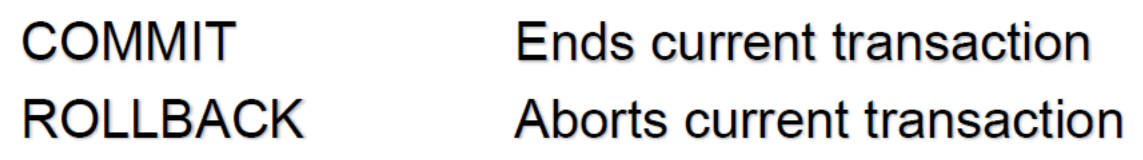
- DCL control the database: Maintenance
- an SQL syntax that allows you to manage users' rights in a database through a pair of commands you can implement.
- Moreover, people who have complete rights to a database are database administrators who can manage user access.
- Example:



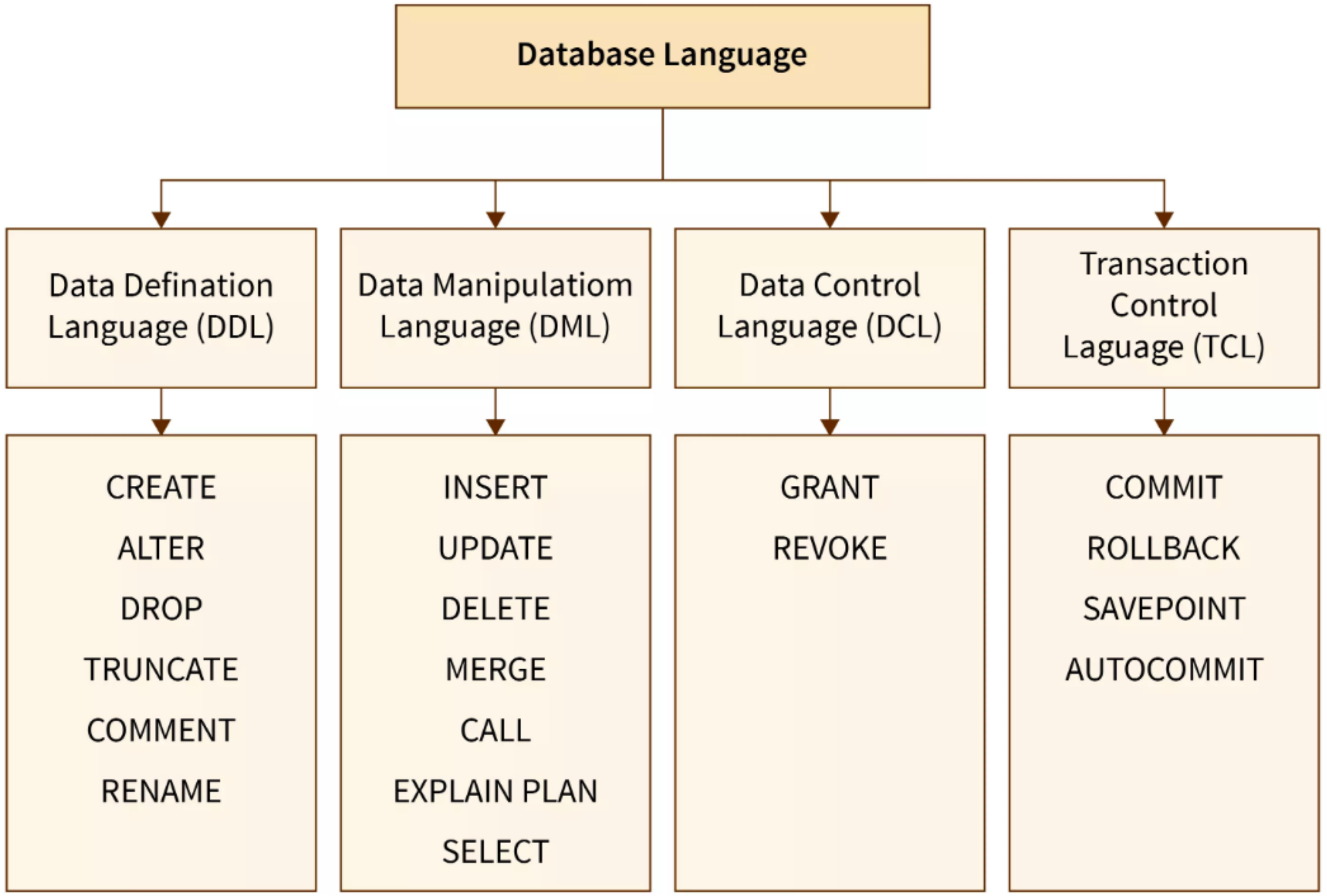
GRANT	Gives user access privileges
REVOKE	Removes privileges

3.5 TRANSACTION CONTROL LANGUAGE (TCL)

- A set of special commands that deal with the transactions within the database.
- Basically, they are used to manage transactions within the database.
- TCL commands are also used for maintaining the consistency of the database.
- Example:



COMMIT	Ends current transaction
ROLLBACK	Aborts current transaction


Database Language
Data Defination Language (DDL)
Data Manipulatiom Language (DML)
Data Control Language (DCL)
Transaction Control Laguage (TCL)
CREATE
ALTER
DROP
TRUNCATE
COMMENT
RENAME
INSERT
UPDATE
DELETE
MERGE
CALL
EXPLAIN PLAN
SELECT
GRANT
REVOKE
COMMIT
ROLLBACK
SAVEPOINT
AUTOCOMMIT

WRAP UP

THANK YOU!