Section 1

Introduction to Database

Section 2

What is Database: database is a collection of organize data that is stored in a way that allows that allow for efficient retrieval and manipulation of data.

In other words

Database is a structured set of computerized data with an accessible interface.

Examples of database

1. Phone book
2. Todo list etc

Databases are used in many applications, such as

1. Website
2. Mobile app
3. Business management systems
4. Social media platforms

Features of database

1. Data storage
2. Data retrieval
3. Data manipulation (update, delete, etc)
4. Data security

Different type of database

1. Relational database eg MySQL, PostgreSQL, Microsoft SQLserver.
2. NoSQL databases eg MongoDB
3. Cloud databases e.g AWS Aurora
   Database play a crucial role in many industries enabling efficient data management and analysis.

Relational database: is a type of database that organizes data in tables, also known as relations.

Each tables has rows (records) and columns (fields) and relationship between tables are established using keys.

Key features of relational database

1. **Tables**: data are stored in tables with well-defined structures
2. **Relationships**: tables are linked using primary and foreign keys
3. **SQL**: relational database use structured query language (SQL) for querying and managing data.

Examples

1. MySQL
2. PostgreSQL,kll;
3. Microsoft SQL Server

Relational database are suitable for complex transactions, data integrity and scalability.

Note:

Transactions is a sequence of operations performed as a single, (***all or nothing***) unit of work. Transactions ensure data consistency and integrity by:

1. **Atomicity:** transactions are treated as a single unit, either all changes are applied or none are.
2. **Consistency**: transactions maintain data consistency, adhering to predefined rules
3. **Isolation**: transactions operate independently, without interference.
4. **Durability**: once committed, transaction changes are permanent.
   Transactions are crucial for financial operation eg banking , e-commerce (order processing) and for critical data update.

Database vs Database Management System.

	Database	**Database management system**
1	Database is a collection of organized data that is stored for in way that allow for efficient retrieval and manipulation .	database management system is a software that manages and interacts with a database, providing tools for data definition, creation, querying, updating, and administration.
2	In other words database is the data itself	DBMS is the system that manages and provides access to that data. E.g. MySQL, PostgreSQL, Microsoft SQL server, MongoDB. The DBMS plays a crucial role in ensuring data security, integrity and scalability.

Section 3

CREATING DATABASES

SHOW Databases;

Creating Database: CREATE DATABASE nameOf Database;

Using Databases; USE nameOfDatabase;

SELECT Database();

Dropping Database; DROP database nameOfDatabase;

Introduction to Tables

Creating Tables:

Datatypes for creating tables

NAME	BREED	AGE
Blue	Scottish	1
rokect	Persian	3
muntiny	Tabby	ten
sam	munchkin	I am yung cat

List of different datatypes

NUMERIC TYPES	STRING TYPES	DATE TYPES
INT	CHAR	DATE
SMALLINT	VARCHAR	DATETIME
TINYINT	BINARY	TIMESTAMP
MEDIUMINT	VARBINARY	TIME
BIGINT	BLOB	YEAR
DECIMAL	TINYBLOP	
NUMERIC	MEDIUMBLOB	
FLOAT	LONGBLOB	
DOUBLE	TEXT	
BIT	TINYTEXT	
	MEDIUMTEXT	
	LONGTEXT	
	ENUM	

Two datatypes that is used frequently is **INT** and **Varchar**

Max value for Int is 4294967295

It can be negative or positive from zero to 4294967295 or negative -0

Varchar is a variable length of string and its max value is from 1 to 255 characters, it values is in quotes .

CREATING TABLES

CREATE TABLE tablename(

Column_name data_type,

Column_name data_type

);

Examples

CREATE TABLE cats(

Name VARCHAR(100),

Age INT

);

To check if your tables worked

SHOW TABLES;

Its show the tables you just created.

SHOW COLUMNS FROM tablename;

Alternatively you can also use DESC TABLES;

They both do the same thing

DROPPING TABLES: DROP TABLE cat;

Will delete the table

To check if it has been deleted you can DESC cat; or SHOW TABLE cat;

INSERTING DATA INTO TABLES

INSEERT INTO cat(name, age) VALUES('draco', 4);

To check if data has been inserted we can use this command

SELECT * FROM cat;

MULTIPLE INSERT

INSERT INTO cat(name, age)

VALUES('draco' , 4),

('blue', 2),

('lizy', 5);

SHOW WARNINGS; gives more details about an error.

Sample Customers Table

ID	Name	Age	City	Spend
1	Alice	30	New York	250
2	Bob	22	Miami	180
3	Charlie	35	New York	320
4	Diana	28	Chicago	210
5	Edward	40	Miami	300

All Filtering & Sorting Commands in MySQL

1. WHERE — Show only what you want

Use it to filter rows.

```
SELECT * FROM Customers
WHERE City = 'Miami';
```

Only shows customers from **Miami**.

2. AND / OR — Combine conditions

```
SELECT * FROM Customers
WHERE City = 'Miami' AND Spend > 200;
```

Customers from **Miami** who spent more than **$200**.

```
SELECT * FROM Customers
WHERE City = 'Chicago' OR Age < 25;
```

Customers who are either from **Chicago** OR **younger than 25**.

3. IN — Match from a list

```
SELECT * FROM Customers
WHERE City IN ('Miami', 'Chicago');
```

Shows customers in either **Miami or Chicago**.

4. BETWEEN — Filter in a range

SELECT * FROM Customers

WHERE Spend BETWEEN 200 AND 300;

Customers who spent **between $200 and $300** (inclusive).

5. LIKE — Pattern matching for text

SELECT * FROM Customers

WHERE Name LIKE 'A%';

Names that **start with "A"** (e.g., Alice)

SELECT * FROM Customers

WHERE Name LIKE '%e';

Names that **end with "e"** (e.g., Charlie)

6. IS NULL / IS NOT NULL — Check empty values

Let's say some customers didn't list their city.

SELECT * FROM Customers

WHERE City IS NULL;

Find customers with **no city listed**.

SELECT * FROM Customers

WHERE City IS NOT NULL;

Only customers **with a city**.

7. ORDER BY — Sort the results

SELECT * FROM Customers

ORDER BY Spend ASC;

Cheapest customers first.

SELECT * FROM Customers

ORDER BY Age DESC;

Oldest customers first.

8. LIMIT — Show only a few rows

```
SELECT * FROM Customers
ORDER BY Spend DESC
LIMIT 3;
```

Show the **top 3 spenders**.

You can even **skip some rows** using OFFSET:

```
SELECT * FROM Customers
ORDER BY Spend DESC
LIMIT 3 OFFSET 2;
```

Skip 2 top spenders, then show the **next 3**.

9. DISTINCT — Remove duplicate results

```
SELECT DISTINCT City FROM Customers;
```

Show each **city only once**, no repeats.

10. Combine Everything

Let's say you want:

- Customers from **New York**
- Who spent **more than $200**
- Sort them by **age**
- Show only the **top 2**

```
SELECT * FROM Customers
WHERE City = 'New York' AND Spend > 200
ORDER BY Age ASC
LIMIT 2;
```

Topic 4 Sorting and filtering

We’re gonna talk about how to **sort** and **limit/filter** results when you get data from your table (aka your big spreadsheet).

1. ORDER BY — Sorting the Results

What it does:

It tells the database how to **sort the rows** — either **smallest to biggest** (A–Z or 1–10), or **biggest to smallest** (Z–A or 10–1).

Example 1: Sort salaries from lowest to highest

SELECT * FROM employees

ORDER BY salary ASC;

"Show me all employees and sort them by **salary** from **low to high**."

- ASC = Ascending = going up

Example 2: Sort by name A to Z

SELECT * FROM employees

ORDER BY name ASC;

"Show me all employees, sorted by **name alphabetically**."

Example 3: Sort by age, oldest first

SELECT * FROM employees

ORDER BY age DESC;

"Show me employees, with the **oldest ones first**."

- DESC = Descending = going down

2. LIMIT and TOP — Show Only Some Rows

Sometimes you don’t want the whole list. You just want the **first few** rows—like the top 5 highest-paid employees.

LIMIT — for MySQL, PostgreSQL, SQLite

```
SELECT * FROM employees
ORDER BY salary DESC
LIMIT 5;
```

"Give me the **top 5 highest-paid** employees."

- ORDER BY salary DESC = sort by salary, highest first
- LIMIT 5 = only show 5 rows

TOP — for SQL Server

```
SELECT TOP 5 * FROM employees
ORDER BY salary DESC;
```

"Same thing, but this works in **SQL Server**."

- TOP 5 = only show the first 5 results

You can combine with WHERE too!

```
SELECT * FROM employees
WHERE department = 'Sales'
ORDER BY age DESC
LIMIT 3;
```

"Show me the **3 oldest Sales employees**."