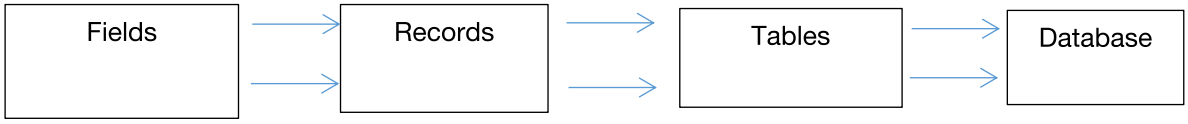
What is a Database?

A **database** is an organized collection of structured information or data, typically stored electronically in a computer system. Databases allow users to efficiently retrieve, insert, and manage data, which makes them essential in almost every modern application.

Similar Name of Row and Column:

- **Row:** Also called a **Record, Tuple and Sample**.
- **Column:** Also known as a **Field, Attribute and Features.**.



Points Breakdown:

1. **Collection of Fields are Records:**
A **field** represents a single piece of information within a table, like a name, age, or ID. When you collect multiple fields together (for example, a name, age, and ID for one person), this creates a **record**.

2. **Collection of Records are Tables:**
A **table** is made up of rows (records) and columns (fields). For example, a table might contain a list of employees, where each row represents a different employee and each column represents a piece of information like name, age, or department.

3. **Collection of Tables are Databases:**
A **database** holds multiple tables that store related information. For example, in a company's database, there might be one table for employee information, another for sales data, and another for inventory.

1. Similar Name of Database and Table:

- Database is similar to a Schema in some contexts. A Schema represents the overall structure and organization of the database, including tables, views, and other objects. It acts as a blueprint of how the database is organized.

- Table is similar to an Entity in the context of database design. An Entity represents a real-world object or concept (like "Employee" or

"Product") that you want to store data about, and this data is organized in a Table.

What is a Table?

A **table** is a structured format that organizes data into rows and columns. Each row in the table is a record, and each column is a field or attribute. Tables allow for organized storage of data.

What is a Column?

A **column** represents a specific field or attribute in a table. It defines the type of data stored for each record in that column. For example, in a table of employees, there might be a column for "Name" and another for "Salary."

What is DBMS (Database Management System)?

A **DBMS** is software used to store, manage, and retrieve data in databases. It allows users to perform various operations like creating, reading, updating, and deleting data.

- **Example of DBMS:**
  Microsoft Access, FileMaker, dbase,FoxPro.

What is RDBMS (Relational Database Management System)?

An **RDBMS** is a type of DBMS that organizes data into tables that are related to each other based on shared fields or keys. It follows the relational model, meaning it stores data in rows and columns and allows relationships between tables.

- **Example of RDBMS:**
  MySQL, PostgreSQL, Oracle DB (relational use), SQL Server.

Difference Between DBMS and RDBMS (Tabular Form):

Feature	DBMS (Database Management System)	RDBMS (Relational Database Management System)
Data Storage Format	Data is stored in files, typically hierarchical or navigational in nature.	Data is stored in tables (rows and columns) following the relational model.
Client-Server Architecture	Does not support client-server architecture. DBMS is typically single-user and desktop-oriented.	Supports client-server architecture, allowing multiple users to interact with the database over a network.
Normalization	Data normalization (reducing redundancy) is not supported.	Supports normalization, which organizes data to reduce redundancy and dependency.
Data Structure	Hierarchical or navigational structure for data. Examples include tree-like structures.	Data is stored in a tabular format (rows and columns), and relationships between tables are maintained via keys.
User Access	Allows only one user to access the system at a time.	Multiple users can access the database simultaneously, ensuring data consistency and availability.
Software & Hardware Requirements	Requires less complex software and lower hardware specifications. Typically used in smaller systems or single-user applications.	Requires more advanced hardware and software due to its complexity and the need to support multiple users and larger datasets.
ACID Properties	Does not support ACID (Atomicity, Consistency, Isolation, Durability) properties, leading to potential data integrity issues.	Supports ACID properties, ensuring safe transactions and reliable data processing.
Data Redundancy	Higher chances of data redundancy (duplicate data) because there is no support for normalization or structured relationships between data.	Data redundancy is minimized through normalization and proper database design.
Data Relationships	DBMS does not support relationships between data (e.g., primary-foreign key relationships).	RDBMS supports relationships between tables using foreign keys, enabling powerful data querying and integrity checks.
Examples	Examples include Microsoft Access, dBase, and FileMaker.	Examples include MySQL, Oracle, PostgreSQL, and Microsoft SQL Server.

SQL Commands Types:

SQL commands are classified into different categories based on the operation they perform on the database. Below is a structured format categorizing the types of SQL commands:

---

1. DDL (Data Definition Language)

DDL commands are used to define and manage the structure of database objects such as tables, schemas, indexes, and views.

Command	Description	Example
CREATE	Used to create new database objects (tables, views, etc.)	CREATE TABLE Employees (ID INT, Name VARCHAR(50));
ALTER	Modifies an existing database object (table, column, etc.)	ALTER TABLE Employees ADD Age INT;
DROP	Deletes an entire database object (table, view, etc.)	DROP TABLE Employees;
TRUNCATE	Removes all records from a table, but not the table itself	TRUNCATE TABLE Employees;
RENAME	Changes the name of an existing database object	RENAME TABLE Employees TO Staff;

---

2. DML (Data Manipulation Language)

DML commands are used to manipulate data stored in the database tables.

Command	Description	Example
INSERT	Adds new data (rows) to a table	INSERT INTO Employees (ID, Name) VALUES (1, 'John');
UPDATE	Modifies existing data in a table	UPDATE Employees SET Name = 'Jane' WHERE ID = 1;
DELETE	Removes data from a table based on a condition.	DELETE FROM Employees WHERE ID = 1;
SELECT	Retrieves data from one or more tables	SELECT * FROM Employees;

---

3. DCL (Data Control Language)

DCL commands are used to control access to data within the database.

Command	Description	Example
GRANT	Gives specific user(s) permission to perform actions	GRANT SELECT ON Employees TO User1;
REVOKE	Removes user permissions to perform specific actions	REVOKE SELECT ON Employees FROM User1;

---

4. TCL (Transaction Control Language)

TCL commands are used to manage transactions in a database. They help in making changes permanent or undoing them.

Command	Description	Example
COMMIT	Saves the current transaction's changes permanently	COMMIT;
ROLLBACK	Reverts the database to the last committed state	ROLLBACK;
SAVEPOINT	Sets a point within a transaction to which you can roll back	SAVEPOINT SP1;
SET TRANSACTION	Sets the transaction properties like isolation level	SET TRANSACTION READ ONLY;

---

5. DQL (Data Query Language)

DQL commands are specifically used to query or retrieve data from the database.

Command	Description	Example
SELECT	Fetches data from one or more tables	SELECT Name, Age FROM Employees WHERE Age > 25;

---

6. DQL Clauses (Extensions of SELECT)

These clauses are used in conjunction with the `SELECT` statement to filter and organize the result set.

Clause	Description	Example
WHERE	Filters records based on a	SELECT * FROM Employees WHERE Age >

Clause	Description	Example
	condition	30;
ORDER BY	Sorts the result set based on one or more columns	SELECT * FROM Employees ORDER BY Name ASC;
GROUP BY	Groups rows that share a property into summary rows	SELECT COUNT(ID), Department FROM Employees GROUP BY Department;
HAVING	Filters groups based on a condition	SELECT Department, COUNT(ID) FROM Employees GROUP BY Department HAVING COUNT(ID) > 5;
JOIN	Combines rows from two or more tables based on a related column	SELECT Employees.Name, Department.Name FROM Employees JOIN Department ON Employees.DeptID = Department.ID;
LIMIT	Limits the number of returned rows	SELECT * FROM Employees LIMIT 10;

---

7. Index Commands

These commands are used to create, modify, and manage indexes, which are used to improve query performance.

Command	Description	Example
CREATE INDEX	Creates an index on a table column	CREATE INDEX idx_emp_name ON Employees(Name);
DROP INDEX	Removes an index from the table	DROP INDEX idx_emp_name;

MYSQL Data Types:

In MySQL, data types define the type of data that can be stored in a column. These data types are grouped into several categories:

1. Numeric Data Types

MySQL supports different numeric types for storing integers and floating-point numbers.

Data Type	Description	Example Value
TINYINT	A very small integer (-128 to 127 or 0 to	127

Data Type	Description	Example Value
	255 if unsigned)	
SMALLINT	A small integer (-32,768 to 32,767 or 0 to 65,535 if unsigned)	32,767
MEDIUMINT	A medium-sized integer (-8,388,608 to 8,388,607 unsigned)	8,388,607
INT or INTEGER	A standard integer (-2,147,483,648 to 2,147,483,647 unsigned)	2,147,483,647
BIGINT	A large integer (-9,223,372,036,854,775,808 to large positive)	9,223,372,036,854,775,807
FLOAT(M,D)	A small floating-point number (precision of 24)	3.14
DOUBLE(M,D)	A large floating-point number (precision of 53)	3.14159
DECIMAL(M,D)	A fixed-point number (exact value) where M is precision and D is scale	123.45

- **M** represents the total number of digits.
- **D** represents the number of digits after the decimal point.

2. String Data Types:

MySQL provides various string types for text and binary data.

Data Type	Description	Example Value
CHAR(M)	A fixed-length string, right-padded with spaces (1 to 255 characters)	'Hello '
VARCHAR(M)	A variable-length string (0 to 65,535 characters, depending on encoding)	'Hello World'
TINYTEXT	A small text string (up to 255 characters)	'Short text'
TEXT	A text string (up to 65,535 characters)	'Longer paragraph'
MEDIUMTEXT	A medium-length text string (up to 16,777,215 characters)	'Medium amount of text'
LONGTEXT	A large text string (up to 4,294,967,295 characters)	'Large amount of text'
BINARY(M)	A fixed-length binary string	b'101010'
VARBINARY(M)	A variable-length binary string	b'101010110101'

ENUM('value1', 'value2', ...)	A string object with one value from a list of allowed values	'value1' or 'value2'
SET('value1', 'value2', ...)	A string object with zero or more values from a list of allowed values	'value1,value2'

3. Date and Time Data Types

MySQL offers several data types to store date and time information.

Data Type	Description	Example Value
DATE	A date in the format YYYY-MM-DD	2024-10-03
TIME	A time in the format HH:MM:SS	12:30:45
DATETIME	A combination of date and time in YYYY-MM-DD HH:MM:SS	2024-10-03 12:30:45
TIMESTAMP	Stores the number of seconds since the Unix epoch (1970-01-01 00:00:01 UTC)	2024-10-03 12:30:45
YEAR(M)	A year in YYYY format. (Can be 2 or 4 digits, but 4 digits are recommended)	2024

4. Binary Data Types

Binary data types are used for storing binary information, such as images, audio, or other types of files.

Data Type	Description	Example Value
BLOB	A Binary Large Object, used to store binary data (up to 65,535 bytes)	Binary data
TINYBLOB	A very small binary object (up to 255 bytes)	Binary data
MEDIUMBLOB	A medium-sized binary object (up to 16,777,215 bytes)	Binary data
LONGBLOB	A large binary object (up to 4,294,967,295 bytes)	Binary data

5. Spatial Data Types

These types are used to store geographical or geometric data.

Data Type	Description	Example Value
GEOMETRY	A spatial data type that can store any type of spatial data	Geometrical data

Data Type	Description	Example Value
POINT	A point in 2D space (X, Y)	POINT(45.1234, -93.1234)
LINESTRING	A line between two or more points	LINESTRING(1 1, 2 2, 3 3)
POLYGON	A polygon defined by multiple points	POLYGON((1 1, 1 2, 2 2, 1 1))

Summary of Categories

- **Numeric:** Used for storing numbers (integers, decimals, floats).
- **String:** Used for storing text or binary strings.
- **Date and Time:** Used for storing date, time, or timestamp data.
- **Binary:** Used for storing large binary objects such as files.
- **Spatial:** Used for geographic or geometrical data.

Each data type in MySQL is chosen based on the kind of data you need to store. For example, you would use `INT` for whole numbers, `VARCHAR` for variable-length text, and `BLOB` for binary files like images.