TASK -1

Create DataBase

Creates a new database

syntax : Create DataBase DataBaseName

* Create Tables

Creates a new tables in a DataBase

Example:

```
CREATE TABLE Teachers (
    TeacherID INT PRIMARY KEY,
    TeacherName VARCHAR(50)
);
```

Definition of Relationships

A relationship in MySQL is a connection between two tables so that the data is linked and organized properly.

It is created using a Primary Key (unique ID) in one table and a Foreign Key in another table.

Why We Use Relationships

1.To avoid duplicate data.

2.To connect related information.

3.To keep data accurate (data integrity).

Types of RelationShips

1. One-to-One (1:1)

   One record in Table A is linked to only one record in Table B, and vice versa.

   Like a one-to-one match.

   Daigram:

   Person (1) —— (1) Passport

2. One-to-Many (1:N)

   One record in Table A can be linked to many records in Table B,

   But each record in Table B links back to only one record in Table A.

   Daigarm:

   Teacher (1) ——< Students (Many)

3. Many-to-Many (M:N)

   Many records in Table A are connected to many records in Table B.

   This needs a third table (called a junction table) to manage the relationship.

Daigarm:

Students (Many) ——< StudentCourses >—— (Many) Courses

Schema

A schema in MySQL is a logical collection of database objects such as tables, views, and relationships. It defines the structure and organization of the database.

Key Points About Schema

1.Schema = Database in MySQL.

2.It contains:

Tables

Relationships (Primary Key, Foreign Key)

* Views

* Stored Procedures

* Triggers

3.It does not hold actual data itself,

It just defines how data will be stored.

Example:

Creating a schema (database) named SchoolDB

```
CREATE SCHEMA SchoolDB;

-- OR

CREATE DATABASE SchoolDB;

USE SchoolDB;
```

Why Schema is Important

1.Organizes data logically.

2.Helps avoid confusion when there are many tables.

3.Defines relationships between tables.

4.Improves data management and security.

1. What is an ER Diagram?

An **ER Diagram** is a **visual representation of a database**.
It shows:

Entities (tables)

Attributes (columns)

Relationships (how tables are connected)

1. Entity

Definition:
An **entity** is a **real-world object or concept** about which data is stored in a database.
In MySQL, an **entity usually becomes a table**.

Key Points:

Represents **something that exists**.

Has **attributes (columns)** that describe it.

Must have a **primary key** to uniquely identify each record.

2. Attribute

Definition:
An **attribute** is a **property or characteristic** of an entity.
In MySQL, an **attribute usually becomes a column** in a table.

Key Points:

Describes the entity.

Can be of different types: text, number, date, etc.

Some attributes are **primary keys (PK)** or **foreign keys (FK)**.

ER Diagram for School Example

Entities:

1.Teachers – TeacherID (PK), TeacherName

2.Students – StudentID (PK), StudentName, TeacherID (FK).

3.Courses – CourseID (PK), CourseName

4.StudentCourses – StudentID (FK), CourseID (FK)

Relationships:

Teacher **1** → **N** Students

Students **M** → **N** Courses (via StudentCourses)

ER Diagram:

Teachers (1) ————< Students
TeacherID PK StudentID PK
TeacherName StudentName
TeacherID FK

Students (M) ————< StudentCourses >————(M) Courses
StudentID PK StudentID FK CourseID PK
StudentName CourseID FK CourseName

Benefits of ER Diagram

1.Visualizes **database structure**

2.Helps in **database design** before creating tables

3.Shows **relationships** clearly

4.Makes it easier to **communicate with developers or clients**

InterView Questions

1. **1.**

What is normalization?
Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity.

It divides large tables into smaller, related tables.

Example:

Separating customer and order information into two different tables

2.Explain primary vs foreign key

Primary Key: A column (or set of columns) that uniquely identifies each row in a table.
Example: `StudentID` in a Students table.

Foreign Key: A column that establishes a relationship between two tables by referencing a primary key in another table.

3.What are constraints?

Constraints are rules applied to columns in a table to maintain data integrity.
Types of constraints:

```
1.NOT NULL

2.UNIQUE

3.PRIMARY KEY

4FOREIGN KEY

5.    Check
6.    default
```

4.What is a surrogate key?

A surrogate key is a system-generated unique identifier (usually a number) used as a primary key when there is no natural unique key.
Example: `EmployeeID` generated automatically.

5.How do you avoid data redundancy?

1.Apply **normalization** rules.

2.Use **foreign keys** to establish relationships instead of duplicating data.

Example: Store customer data only once in the `Customers` table and reference it u using a foreign key in the `Orders` table.

6.What is an ER diagram?

An **Entity-Relationship (ER) Diagram** visually represents entities (tables), attributes (columns), and relationships between entities in a database.

7.What are the types of relationships in DBMS?

1.One-to-One (1:1)

One record in Table A is linked to exactly one record in Table B, and vice versa.

2. **One-to-Many (1:N)**

One record in Table A is linked to many records in Table B,
but each record in Table B links back to only one record in Table A.

3. **Many-to-Many (M:N)**

Many records in Table A are linked to many records in Table B.

8.Explain the purpose of AUTO_INCREMENT.

`AUTO_INCREMENT` automatically generates a unique number for each new record inserted into a table.
Example:

```
CREATE TABLE Students (
    StudentID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(50)
);
```

9.What is the default storage engine in MySQL?

The default storage engine is **InnoDB**, which supports transactions, foreign keys, and row-level locking.

10.What is a composite key?

A composite key is a combination of two or more columns used together to uniquely identify a record in a table.
Example: A `CourseID` and `StudentID` together can uniquely identify a record in a `Registrations` table.