Employee Management System

Designing a database for an employee management system involves defining the structure, relationships, and functionalities of the database to efficiently store and manage employee data. Here are some of the key functionalities and considerations for designing such a database:

- Employee Information:
- Department Information:
- Salary :
- Project Information:

We will be using MySQL as the DBMS to create the database and execute its related operations.

1. Introduction to MySQL

MySQL is an open-source relational database management system (RDBMS) that uses structured query language (SQL) to manage and manipulate data in a database. It is widely used for various applications, from small web applications to large enterprise systems.

MySQL's key features include:

• Scalability: Capable of handling large amounts of data and concurrent connections.

• Flexibility: Supports various data types and storage engines.

• Performance: Optimized for speed and efficiency.

• Reliability: Known for its stability and robustness.

2. Installation of MySQL

MySQL can be installed on various operating systems, including Windows, macOS, and Linux. Here are the general steps to install MySQL in Windows system:

• Download the MySQL installer from the official website.

https://dev.mysql.com/downloads/installer/

• Run the installer and follow the on-screen instructions.

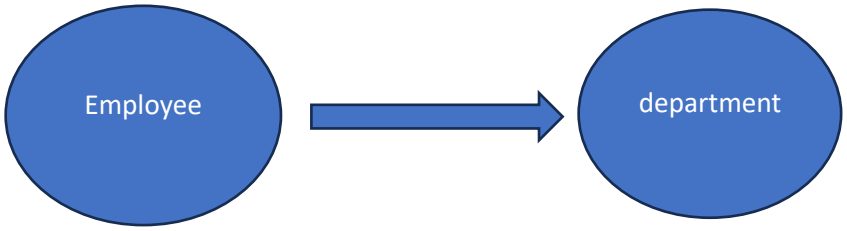
• Choose the installation type (Typical, Complete, or Custom). Recommended Custom.

• Set a root password for the MySQL server.

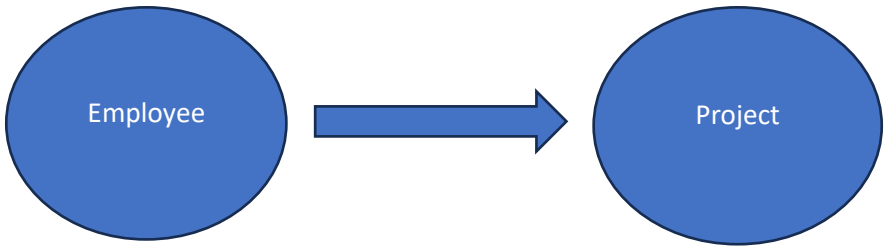
3. E-R Diagram (ERD)

An Entity-Relationship Diagram (ERD) is a visual representation of the data model that shows the entities, attributes, relationships between entities, and cardinality. ERDs are commonly used in database design to help developers and stakeholders understand the structure and relationships within a database.

One-to-One: When each entity in each entity set can take part only once in the relationship, the cardinality is one-to-one. Let us assume that a male can marry one female and a female can marry one male. So the relationship will be one-to-one.



One-to-Many: In one-to-many mapping as well each entity can be related to more than one relationship and the total number of tables that can be used in this is 2. Let us assume that one surgeon department can accommodate many doctors. So the Cardinality will be 1 to M. It means one department has many Doctors.



Many-to-One: When entities in one entity set can take part only once in the relationship set and entities in other entity sets can take part more than once in the relationship set, cardinality is many to one. Let us assume that a student can take only one course but one course can be taken by many students.

Many-to-Many: When entities in all entity sets can take part more than once in the relationship cardinality is many to many.

1. Employee

Attributes:

1. **EmployeeID** (Primary Key):
   An auto-incremented unique identifier for each employee.
2. **First_name**:
   The first name of the employee.
3. **Last_name**:
   The last name of the employee.
4. **Address**:
   The address or physical location of the employee.
5. **Salary**:
   The salary of the employee.
6. **Gender**:
   The gender of the employee.
7. **DepartmentID** (Foreign Key):
   A reference to the department to which the employee belongs, establishing a many-to-one relationship with the "Department" table.

Relationships:

Many **Employees** can enroll in one **Department**(Many-to-One)
One Employee have one department (One-To-One)

2. Department

Attributes:

1. **DepartmentID** (Primary Key):
   An auto-incremented unique identifier for each department.
2. **Name**:
   The name of the department.

Relationships:

One Department can belong to Many Employees.

3. Salary

Attributes:

- **SalaryID** (Primary Key):
  - An auto-incremented unique identifier for each salary record.
- **EmployeeID** (Foreign Key):
  - A reference to the employee for whom the salary record is maintained, establishing a one-to-many relationship with the "Employee" table.
- **SalaryAmount**:
  - The amount of the salary for a specific period.
- **StartDate**:
  - The start date of the salary period.
- **EndDate**:
  - The end date of the salary period.

4. Project

Attributes:

- **ProjectID** (Primary Key):
  - An auto-incremented unique identifier for each project.
- **Name**:
  - The name of the project.
- **Budget**:
  - The budget allocated to the project.
- **Location**:
  - The location or place where the project is based.

Tables:

1. *Employee*

Field	Type	Null	Key	Default	Extra
EmployeeID	int	NO	PRI	NULL	auto_increment
First_name	varchar(255)	YES		NULL	
Last_name	varchar(255)	YES		NULL	
Address	varchar(255)	YES		NULL	
Salary	int	YES		NULL	
gender	varchar(255)	YES		NULL	
DepartmentID	int	YES	MUL	NULL	

2. *Department*

Field	Type	Null	Key	Default	Extra
DepartmentID	int	NO	PRI	NULL	auto_increment
Name	varchar(255)	NO		NULL	

3. *Salary*

Field	Type	Null	Key	Default	Extra
SalaryID	int	NO	PRI	NULL	auto_increment
EmployeeID	int	YES	MUL	NULL	
SalaryAmount	decimal(10,2)	YES		NULL	
StartDate	date	YES		NULL	
EndDate	date	YES		NULL	

4. *Project*

Field	Type	Null	Key	Default	Extra
ProjectID	int	NO	PRI	NULL	auto_increment
Name	varchar(255)	YES		NULL	
Budget	decimal(10,2)	YES		NULL	
Location	varchar(255)	YES		NULL	