**Open Ended Experiment- Enterprise Database Management System**

**Aim**: Take an example of real-world situation to and make entity- relationship diagram along with schema and convert it into a relational database.

**Conceptual Designing**

Conceptual design involves abstractly defining entities and relationships based on database requirements. Entity-Relationship (ER) modelling represents these concepts visually using diagrams. Schema design then translates the ER model into a physical database schema with tables, columns, and constraints, facilitating the actual implementation of the database system.

For conceptual designing, the entity and their respective attributes can be inferred from the following table.

|  |  |
| --- | --- |
| **Departments**   * DepartmentID (Primary Key * DepartmentName * ManagerID * Location | **Employees**   * EmployeeID (Primary Key) * FirstName * LastName * Email * PhoneNumber * HireDate * JobTitle * Salary * DepartmentID (Foreign Key referencing Department) |
| **Project**   * ProjectID (Primary Key) * ProjectName * StartDate * EndDate * Budget * DepartmentID (Foreign Key referencing Department) * ManagerID (Foreign Key referencing Employees) | **EmployeeProjects**   * EmployeeID (Foreign Key referencing Employees) * ProjectID (Foreign Key referencing Projects) * AssignmentStartDate * AssignmentEndDate |
| **EmployeeSkills**   * EmployeeID (Foreign Key referencing Employees) * Skill * ExperienceYears | **EmployeeBenefits**   * EmployeeID (Foreign Key referencing Employees) * Skill * ExperienceYears |

The following type of relationship is seen between different entities.

1. **Employees - Departments:** One-to-Many relationship: An employee belongs to one department, but a department can have many employees.
2. **Employees - Projects:** One-to-Many relationship: An employee can manage multiple projects, but a project is managed by one employee.
3. **Projects - Departments:** One-to-Many relationship: A project belongs to one department, but a department can have many projects.
4. **EmployeeProjects - Employees:** Many-to-One relationship: Many employees can be assigned to multiple projects, but each assignment is associated with one employee.
5. **EmployeeProjects - Projects:** Many-to-One relationship: Many projects can have multiple employees assigned to them, but each assignment is associated with one project.
6. **EmployeeSkills - Employees:** One-to-Many relationship: An employee can have multiple skills, but each skill is associated with one employee.
7. **EmployeeBenefits - Employees:** One-to-Many relationship: An employee can have multiple benefits, but each benefit is associated with one employee.

**ER Diagram**

**Schema**

A screenshot of a computer

Description automatically generated

**Creation of the database in PostgreSQL**

* 1. After opening pgadmin 4, right click on Databases and then select Create followed by Database…

A screenshot of a computer

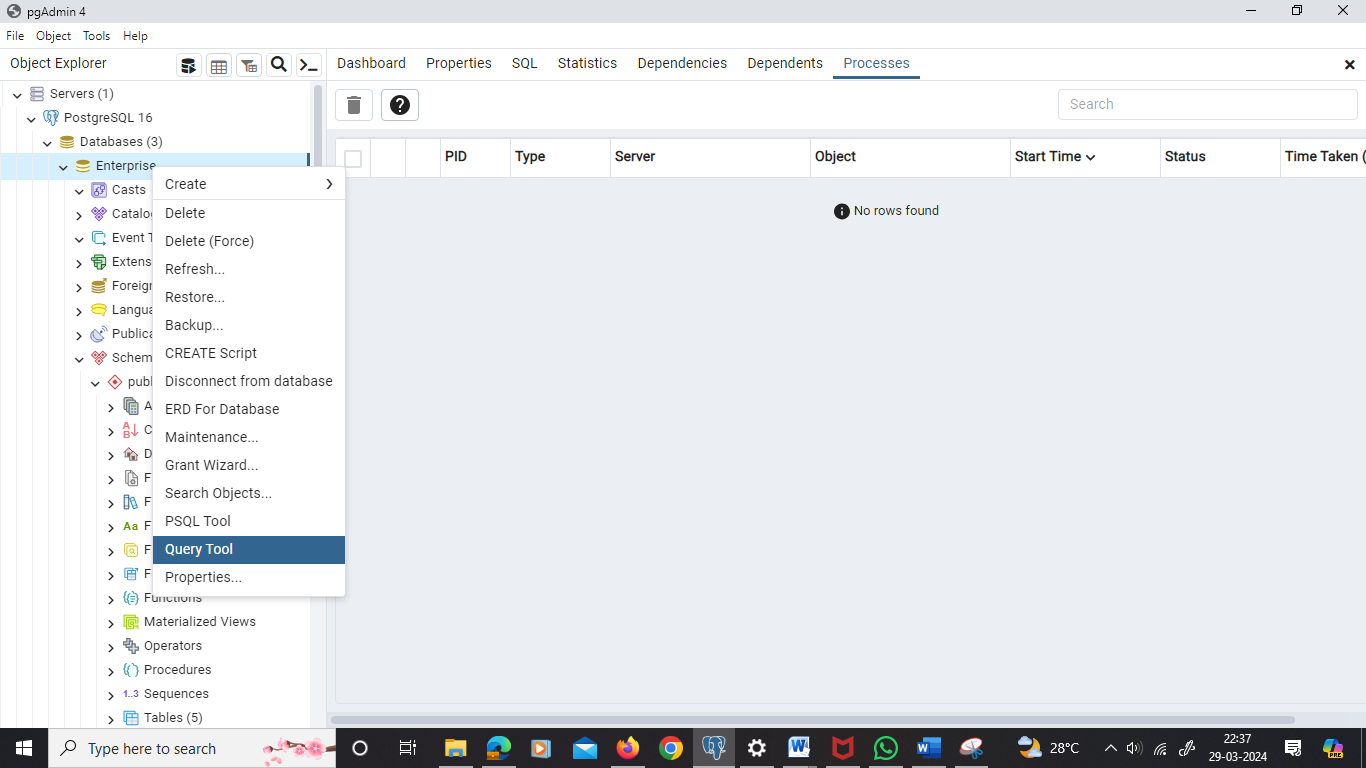
Description automatically generated

* 1. Enter the necessary information and create the database.

A screenshot of a computer

Description automatically generated

* 1. Right click on the created database and select query tool.



**Coding Commands for Relational Database**

1. Department
2. Code

A computer screen shot of a computer code

Description automatically generated

1. Output-

A screenshot of a computer

Description automatically generated

1. Employees
2. Code-

A computer screen shot of a computer code

Description automatically generated

1. Output

A screenshot of a phone number

Description automatically generated

1. Projects
2. Code

A computer code on a white background

Description automatically generated

1. Output

A screenshot of a computer

Description automatically generated

1. EmployeeProjects
2. Code

A computer screen shot of a code

Description automatically generated

1. Output

A screenshot of a computer

Description automatically generated

1. EmployeeSkills
2. Code

A computer screen shot of a computer code

Description automatically generated

1. Output

A screenshot of a computer

Description automatically generated

1. EmployeeBenefits
2. Code

A screenshot of a computer

Description automatically generated

1. Output

A screenshot of a computer

Description automatically generated

**Concepts used while coding**

1. Data Definition Language (DDL): DDL statements are used to define the structure of the database schema. Examples include CREATE TABLE statements for defining tables and their attributes, specifying primary keys, foreign keys, and constraints.
2. Data Manipulation Language (DML): DML statements would be used to interact with the database and manipulate data. However, they are not explicitly shown in the provided code. Examples of DML statements include INSERT, UPDATE, DELETE for adding, modifying, and deleting data in the tables.
3. Data Types: Data types such as VARCHAR, INT, DATE, and DECIMAL are used to define the format and size of the data stored in each column. For instance, VARCHAR(100) indicates a variable-length character string with a maximum length of 100 characters, while DECIMAL(10, 2) specifies a decimal number with a total of 10 digits, with 2 digits after the decimal point.
4. Keys: Primary keys (e.g., DepartmentID in Departments) uniquely identify each record within a table. Foreign keys (e.g., DepartmentID in Employees referencing DepartmentID in Departments) establish relationships between tables by referencing the primary key of another table.
5. Constraints: Constraints such as PRIMARY KEY, FOREIGN KEY, and UNIQUE are specified to enforce data integrity rules and maintain consistency. PRIMARY KEY constraints ensure uniqueness and non-null values for key columns, while FOREIGN KEY constraints enforce referential integrity by ensuring that values in one table match values in another table.
6. Normalization: The schema design appears to follow normalization principles, as data is organized into separate tables to minimize redundancy and maintain data integrity. This is evident from the use of separate tables for related entities and the establishment of relationships between them using foreign keys.

**Result:** The project involved the creation and study of an enterprise database management system.

**Marking Scheme:**

