ASSIGNMENT – 01

Name – RAVI SHEKHAR

ROLL – 22CS3075

Q. 1. Define database and how it is different from database management system.

Ans: A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system.

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data.

A database typically requires a comprehensive database software program known as a database management system (DBMS). A DBMS serves as an interface between the database and its end users or programs, allowing users to retrieve, update, and manage how the information is organized and optimized. A DBMS also facilitates oversight and control of databases, enabling a variety of administrative operations such as performance monitoring, tuning, and backup and recovery.

Some examples of popular database software or DBMSs include MySQL, Microsoft Access, Microsoft SQL Server, and Oracle Database.

Q. 2. Write down the differences between database management system and file management system with proper examples.

Ans: File Management System (FMS): In FMS, data is organized in files, which are essentially collections of records. Each file contains records, and these records contain fields.

- ✓ Example: In a simple text document, each line can be considered a record, and the various pieces of information on that line (e.g., name, age, address) are fields.

Database Management System (DBMS): In DBMS, data is organized in a more structured way using tables. Each table consists of rows (records) and columns (fields).

● Example: In a relational database, you might have a table for "Customers" with columns for "CustomerID," "Name," "Address," etc.

Data Redundancy:

● FMS: Redundancy is common in FMS, as the same data may be stored in multiple files. For example, if the address of a customer changes, it needs to be updated in every file where that customer's information is stored.

● DBMS: DBMS minimizes redundancy through normalization. Data is stored in tables in a way that minimizes duplication, and relationships between tables are used to link related information.

Data Independence:

● FMS: Changes to the structure of a file may require modifying application programs that use that file. This lack of independence can lead to maintenance challenges.

● DBMS: DBMS provides data independence. Changes to the database structure (schema) generally do not affect the application programs that use the data. This separation of data and application logic is achieved through the use of a data definition language (DDL) and a data manipulation language (DML).

Data Integrity:

● FMS: Ensuring data integrity (e.g., constraints, relationships) is the responsibility of application programs. FMS doesn't provide built-in mechanisms for enforcing data integrity rules.

● DBMS: DBMS offers mechanisms to enforce data integrity through constraints (e.g., primary keys, foreign keys), ensuring that the data in the database remains accurate and consistent.

Concurrency Control:

● FMS: FMS may lack sophisticated mechanisms for handling concurrent access by multiple users, leading to potential data inconsistency and conflicts.

● DBMS: DBMS includes robust concurrency control mechanisms to manage multiple users accessing and modifying the data simultaneously, ensuring consistency and preventing conflicts.

Example Systems:

● FMS: Traditional file-based systems like flat file systems where data is stored in simple text files or CSV files.

● DBMS: Examples include MySQL, Oracle, Microsoft SQL Server, and PostgreSQL, which are relational database management systems (RDBMS).

In summary, while File Management Systems are suitable for simple data storage and retrieval, Database

Management Systems provide a more organized, efficient, and scalable approach for managing and

manipulating data in a structured and controlled manner.

Q.3. Explain the role of data model in DBMS and explain its all types.

Ans: In a Database Management System (DBMS), a data model plays a crucial role in defining the structure

and organization of the data within the database. A data model serves as a conceptual framework that

facilitates the understanding, representation, and manipulation of data. There are several types of data

models, each offering a different way to represent and organize data. Here are the main types of data

models:

Hierarchical Data Model:

● Description: In this model, data is organized in a tree-like structure with a top-down hierarchy. Each parent node can have multiple child nodes, and each child node can have only one parent.

● Example: An organizational chart where each employee reports to a higher-level employee, forming a hierarchical structure.

Network Data Model:

● Description: Similar to the hierarchical model, but with the ability for a child node to have multiple parent nodes. It represents complex relationships more flexibly than the hierarchical model.

● Example: A university database where a student can be associated with multiple courses, and each course can have multiple instructors.

Relational Data Model:

● Description: This model represents data as tables consisting of rows and columns. Tables can have relationships with each other, and data is stored without any predefined hierarchy.

● Example: A database for a library where you have tables for "Books," "Authors," and "Publishers," with relationships established between them.

Entity-Relationship Model (ER Model):

● Description: The ER model is a high-level, abstract representation that uses entities, attributes, and relationships to describe the data and its structure.

● Example: In a university database, entities might include "Student," "Course," and "Instructor," with relationships like "enrolls in" and "teaches."

Object-Oriented Data Model:

● Description: This model extends the concepts of object-oriented programming to databases, representing data as objects that encapsulate attributes and methods.

● Example: In a multimedia database, an "Image" object might have attributes like "file name" and "resolution" and methods like "display" or "resize."

Object-Relational Data Model:

● Description: This model combines features of both relational and object-oriented data models. It allows for the representation of complex data types, inheritance, and encapsulation.

● Example: Extending a relational database to include object-oriented features, such as user-defined data types or methods associated with data.

NoSQL Data Models:

● Description: NoSQL databases use various data models that diverge from the traditional relational model. Examples include document stores, key-value stores, column-family stores, and graph databases.

● Example: MongoDB, a document store, represents data as JSON-like documents, while Neo4j, a graph database, models data as nodes and relationships.

The choice of a data model depends on the specific requirements of the application and the nature of the

data to be stored. Each data model has its strengths and weaknesses, and the selection of the

appropriate model is crucial for designing an efficient and effective database system.

Q.4. Differentiate entity, entity type and entity set.

Ans: In the context of database design and the Entity-Relationship (ER) model, the terms "entity," "entity type," and "entity set" are related concepts but have distinct meanings:

1. **Entity:**
    - An entity represents a real-world object or concept that has data to be stored in a database. Examples of entities can include a person, place, event, or thing. Each entity has a set of attributes that describe its properties.
2. **Entity Type:**
    - An entity type is a collection or category of entities that share common characteristics or attributes. It defines a set of entities that have the same structure, meaning they have the same attributes and relationships. For

example, "Person" can be an entity type, and individual people are instances of this entity type.

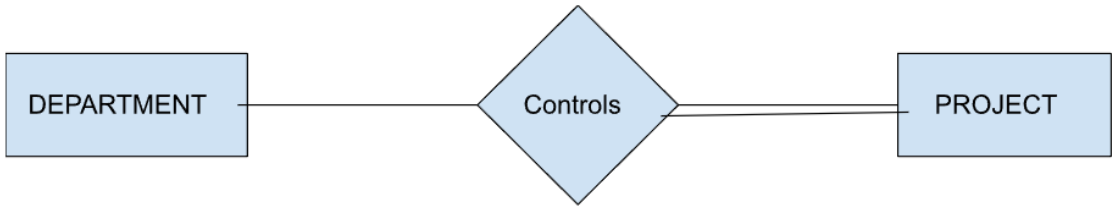
3. **Entity Set:**
    - An entity set is a collection of instances of an entity type. In other words, it's the set of all entities of a particular type that exists in the database at a given point in time. Continuing with the example, if "Person" is an entity type, then the set of all individual persons stored in the database is the "Person" entity set.

In summary, an entity is a specific occurrence or object in the real world, an entity type is a category or class of similar entities, and an entity set is the collection of all instances of a particular entity type in a database. The relationships between entities, their attributes, and the way they interact are key elements in designing a database using the ER model.

Q.5. Draw the ER diagram for the following task:

- A department controls a number of projects, each of which has a unique name, a unique number, and a single location.

Ans:



ER DIAGRAM OF A DEPARTMENT

Entities:

1. Department
2. Project

Attributes:

1. Department:
    - DepartmentID (Primary Key)
    - DepartmentName
2. Project:

- ProjectID (Primary Key)
- ProjectName (Unique)
- ProjectNumber (Unique)
- Location

Relationship:

1. A department controls a number of projects.
    - Relationship: Controls (1:N)

Q.6. Draw the ER diagram for the following description;

We store each employee's name (first, last, MI), Social Security number (SSN), street address, salary, sex (gender), and birth date. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. We keep track of the current number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each employee (who is another employee).

Ans:

ER DIAGRAM OF A COMPANY

Entities:

1. Employee
2. Department
3. Project

Attributes:

1. Employee:
    - EmployeeID (Primary Key)
    - FirstName
    - LastName
    - MiddleInitial
    - SSN
    - StreetAddress
    - Salary
    - Sex
    - BirthDate
2. Department:
    - DepartmentID (Primary Key)
    - DepartmentName
3. Project:
    - ProjectID (Primary Key)
    - ProjectName

Relationships:

1. An employee is assigned to one department.
    - Relationship: WorksIn (1:N)
2. An employee may work on several projects.
    - Relationship: WorksOn (M:N)
    - Associative Entity: WorkAssignment
        - HoursPerWeek
3. An employee has a direct supervisor.
    - Relationship: Supervises (1:N)