310241 - Database Management System

(2019 Pattern) (Semester - V)

Unit 1	Introduction to Database Management Systems and ER Model	Marks
Introduction, Purpose of Database Systems, Database-System Applications, View of Data, Database Languages, Database System Structure, Data Models. Database Design and ER Model: Entity, Attributes, Relationships, Constraints, Keys, Design Process, Entity-Relationship Model, ER Diagram, Design Issues, Extended E-R Features, converting ER and EER diagram into tables.		**15**

1. Introduction to DBMS

A Database Management System (DBMS) is software that enables users to define, create, manage, and manipulate databases. It stores data in a structured way, ensures data integrity, and allows multiple users to access the same database securely. DBMS helps reduce data redundancy, enhances data sharing, and provides a centralized view of information across an organization.

- DBMS stores data in tables, which consist of rows (records) and columns (fields).
- It separates the logical view of data from the physical storage.
- Offers security mechanisms to protect data from unauthorized access.
- Supports multiple users simultaneously using concurrency control.
- Maintains data integrity through constraints and validation rules.

2. Purpose of Database Systems

The purpose of a database system is to overcome the limitations of file-based systems by organizing data efficiently. Traditional file systems lack flexibility, lead to duplication, and make it difficult to manage data consistency and security. A DBMS provides a unified framework to handle large amounts of data with minimal redundancy and high reliability.

- Helps avoid duplicate and inconsistent data by centralizing control.
- Provides a standard method to store, retrieve, and update data.
- Supports complex queries and reporting for business decision-making.
- Ensures transaction safety using the ACID properties.
- Simplifies backup, recovery, and audit processes.

3. Database-System Applications

Databases are used in almost every sector today. They serve as the backbone of many applications that require storage, retrieval, and analysis of data. From small businesses to large enterprises, databases power daily operations.

- **Banking Systems:** Manage accounts, customer info, loans, and transactions.
- **Airlines:** Handle schedules, bookings, and ticketing.
- **Educational Institutes:** Manage student records, marks, fees, and courses.
- **E-commerce Platforms:** Store product catalogs, orders, payments, and customer data.
- **Healthcare Systems:** Maintain patient records, prescriptions, and treatment histories.
- **Telecom:** Billing systems, user data, and plan subscriptions.

4. View of Data

A DBMS provides data abstraction through three levels of views. This abstraction helps different users interact with the database without worrying about the complexities of data storage.

- **Physical Level:** Describes how data is stored physically on disks, using file systems, indexes, and storage blocks.
- **Logical Level:** Describes what data is stored and the relationships among data. It hides the physical details from users.
- **View Level (External Level):** Describes only part of the database relevant to a particular user. Different users can have different views of the same data.

Example: A student user may only see their own profile, while an admin can see all student records.

5. Database Languages

DBMS offers a set of languages to define database structure, manage records, control user access, and manage transactions. These languages are part of SQL (Structured Query Language).

- **DDL (Data Definition Language):** Used to define and modify database structure.
  - Commands: CREATE TABLE, ALTER TABLE, DROP TABLE

- **DML (Data Manipulation Language):** Deals with data manipulation like retrieval and update.
  - Commands: SELECT, INSERT, UPDATE, DELETE
- **DCL (Data Control Language):** Controls user access to data.
  - Commands: GRANT, REVOKE
- **TCL (Transaction Control Language):** Ensures safe execution of a group of SQL commands.
  - Commands: COMMIT, ROLLBACK, SAVEPOINT

6. Database System Structure

A DBMS has multiple components that work together to perform different operations like query processing, transaction management, and data storage. Understanding its structure helps us know how queries are handled internally.

- **Query Processor:** Translates SQL queries into low-level instructions and optimizes their execution.
- **Storage Manager:** Responsible for storing data on disk, managing files and indexes.
- **Transaction Manager:** Ensures transactions are completed successfully or rolled back if any failure occurs.
- **Buffer Manager:** Controls the data transfer between disk storage and main memory (RAM).
- **Authorization and Integrity Manager:** Manages access control and enforces constraints (e.g., NOT NULL, UNIQUE).

7. Data Models

A data model defines the structure of a database and determines how data is stored, connected, and manipulated. It provides a blueprint for database designers and helps users understand the data.

- **Relational Model:** Represents data in tables (relations) with rows and columns. Most widely used model.
- **Entity-Relationship (ER) Model:** Uses ER diagrams to show entities, attributes, and relationships visually.
- **Object-Based Model:** Based on object-oriented programming; supports classes, objects, and inheritance.

- **Hierarchical Model:** Data is stored in a tree structure. Each parent can have many children, but each child has only one parent.
- **Network Model:** Similar to the hierarchical model but allows many-to-many relationships using graph-like structures.
- **Semi-structured Model:** Data doesn't follow a rigid schema, used in XML and JSON formats.

8. Database Design and ER Model

Database design involves determining what data to store and how to organize it. The ER model helps in designing databases at a high level by showing entities and their relationships before implementation.

- **Entity:** Any real-world object (e.g., Student, Employee) that has a distinct existence.
- **Attribute:** Property of an entity (e.g., Name, RollNo). Each entity has multiple attributes.
- **Relationship:** A logical connection between two or more entities (e.g., Student ENROLLS in Course).

Types of Attributes:

- **Simple Attribute:** Cannot be divided further (e.g., Age).
- **Composite Attribute:** Can be broken down into sub-parts (e.g., Name → First Name, Last Name).
- **Single-valued Attribute:** Contains only one value (e.g., Roll Number).
- **Multivalued Attribute:** Can have multiple values (e.g., Phone Numbers).
- **Derived Attribute:** Computed from other attributes (e.g., Age from Date of Birth).

Types of Relationships:

- **One-to-One (1:1):** One student has one ID card.
- **One-to-Many (1:N):** One teacher teaches many students.
- **Many-to-Many (M:N):** Many students enroll in many courses.

Constraints:

- **Cardinality:** Specifies the number of instances an entity can be related to another entity (1:1, 1:N, M:N).
- **Participation Constraint:** Indicates whether all entity instances must participate in the relationship.
  - Total Participation: Every entity must participate.
  - Partial Participation: Some entities may not participate.

Keys:

- **Primary Key:** Uniquely identifies each record in a table.
- **Candidate Key:** A set of fields that can qualify as a primary key.
- **Super Key:** Any combination of fields that uniquely identifies a record.
- **Foreign Key:** A field in one table that refers to the primary key of another table.

9. Design Process

Database design must follow a step-by-step method to ensure the system meets user requirements and functions efficiently.

1. **Requirement Analysis:** Identify what the users need. Understand business processes and the data involved.
2. **Conceptual Design:** Use ER diagrams to describe entities, attributes, and relationships.
3. **Logical Design:** Convert the ER diagram into relational schemas (tables with keys).
4. **Schema Refinement:** Apply normalization to eliminate data redundancy and improve structure.
5. **Physical Design:** Choose storage methods, indexes, and access paths.
6. **Security Design:** Set up user roles, permissions, and security constraints.

10. ER Diagram

An ER diagram is a graphical representation used in database design. It shows entities, their attributes, and relationships among entities.

Symbols Used:

- **Rectangle:** Entity (e.g., Student)
- **Oval:** Attribute (e.g., Name, Age)
- **Underlined Oval:** Key attribute (e.g., RollNo)

- **Double Oval:** Multivalued attribute (e.g., PhoneNumber)
- **Dashed Oval:** Derived attribute (e.g., Age from DOB)
- **Diamond:** Relationship (e.g., Enrolls)
- **Lines:** Connect attributes to entities and entities to relationships.

11. Extended ER Features (EER)

EER enhances the basic ER model to represent more complex relationships such as hierarchy and shared relationships.

- **Specialization:** Creating sub-entities from a general entity based on some differences. (e.g., Employee → Engineer, Manager)
- **Generalization:** Combining multiple entities into a generalized super-entity. (e.g., Car and Bike → Vehicle)
- **Aggregation:** Treating a relationship as an entity for participating in other relationships. Useful for modeling complex scenarios.

12. Converting ER to Tables

To implement a database using SQL, we must convert ER diagrams into relational tables.

- **Entity to Table:** Each entity becomes a separate table.
- **Attributes to Columns:** Entity attributes become table columns.
- **Primary Key:** Becomes the main identifier in the table.
- **1:1 Relationship:** Add foreign key to one of the tables.
- **1:N Relationship:** Add foreign key to the N-side table.
- **M:N Relationship:** Create a new table with foreign keys from both related tables.
- **Weak Entity:** Include primary key of related strong entity as a foreign key.
- **Specialization/Generalization:** Use a separate table for each subclass or use a type attribute.

Summary Points

- DBMS is used to store, retrieve, and manage data securely.
- It solves the problems of data redundancy and inconsistency in file systems.
- Three levels of data abstraction: physical, logical, and view level.
- Data models help organize and visualize the structure of databases.
- ER diagrams are useful for designing databases before implementation.
- EER adds concepts like specialization, generalization, and aggregation.
- ER models must be mapped into relational tables for implementation.