Database Management System

Lecture 1

- **Introduction to Databases**
  - A database is a collection of related data with meaning, stored systematically for specific purposes.
  - Traditional applications include banking and hotel reservations, while modern applications cover social media and cloud storage.
  - Database users interact with databases in various tasks such as online shopping and inventory management.

- **Database Characteristics**
  - Databases reflect changes in the real world they model.
  - Designed for specific users and purposes, ensuring data integrity and coherence.

- **Database Management Systems (DBMS)**
  - A DBMS handles data storage, manipulation (queries and updates), and sharing across users.
  - It manages multiuser access, ensuring data consistency and protecting against unauthorized access.
  - DBMS systems offer protection against hardware/software failures and secure access.

- **Database Design**
  - The design process includes gathering requirements, conceptual design (e.g., ER models), logical design (e.g., relational models), and physical design (storage and access methods).

Lecture 2

- **DBMS Architecture**
  - DBMS has evolved from tightly integrated systems to modular architectures with client-server setups.
  - In client-server architecture, the client handles user interaction while the server manages data storage and query processing.
  - Distributed DBMS is used for handling massive data, such as in cloud computing environments.

- **Data Models**
    - Conceptual models represent high-level data structures (e.g., ER models).
    - Implementation models like the relational model hide storage details but can be implemented in DBMSs.
    - Physical models deal with how data is stored at the hardware level, using techniques like indexing and hashing.

- **Three-Schema Architecture**
    - The internal schema describes the physical storage of the database.
    - The conceptual schema describes the overall structure of the database for users.
    - External schemas provide different views for user groups, hiding the complexity of the database.
    - Logical data independence allows changes to the conceptual schema without affecting user views. Physical data independence allows changes to the internal schema without affecting the conceptual schema.

- **Database Languages**
    - DDL (Data Definition Language) defines the structure of the database schema.
    - SDL (Storage Definition Language) specifies how data is stored, using file organization and indexing.
    - DML (Data Manipulation Language) is used for data operations like insertion, deletion, and querying (e.g., SQL).

Lecture 3

- **Evolution of Data Models**
    - The hierarchical model uses a tree structure, good for one-to-many relationships but limited in flexibility.
    - The network model allows for many-to-many relationships by introducing multiple parent nodes.
    - The ER model represents real-world entities, their attributes, and relationships in a visual, easy-to-understand format.
    - The relational model represents data in rows (records) and columns (attributes), using primary and foreign keys for relationships.
    - Object-oriented models represent entities as objects, which encapsulate both data and behavior, allowing for complex data types like multimedia.

- **Entity-Relationship (ER) Model**
    - Entities are real-world objects with attributes (e.g., Faculty with ID and Name).
    - Relationships represent associations between entities (e.g., Faculty works in a Department).
    - ER diagrams visually represent entities, relationships, and attributes, making database design easier.

- **Relational Model**
    - Data is organized in tables with rows (records) and columns (attributes).
    - SQL is the standard language used for querying and managing relational databases.

- **Object-Oriented Data Models**
    - Data and its relationships are stored together in objects, allowing for easier handling of complex data types such as images and videos.

- **ER Model Design Process**
    - Entities, attributes, and relationships are mapped from real-world situations to the database using ER diagrams.
    - Constraints like cardinality (one-to-one, one-to-many) and participation (total, partial) help define relationships between entities.

- **Composite and Multivalued Attributes**
    - Composite attributes can be divided into subparts (e.g., Faculty Name can be divided into First Name and Last Name).
    - Multivalued attributes hold multiple values (e.g., Faculty Phone Numbers).