Hamromaster
सुनौलो भविष्य हातमा **Hamromaster** को साथमा

Hamromaster

COMPLETE NOTES

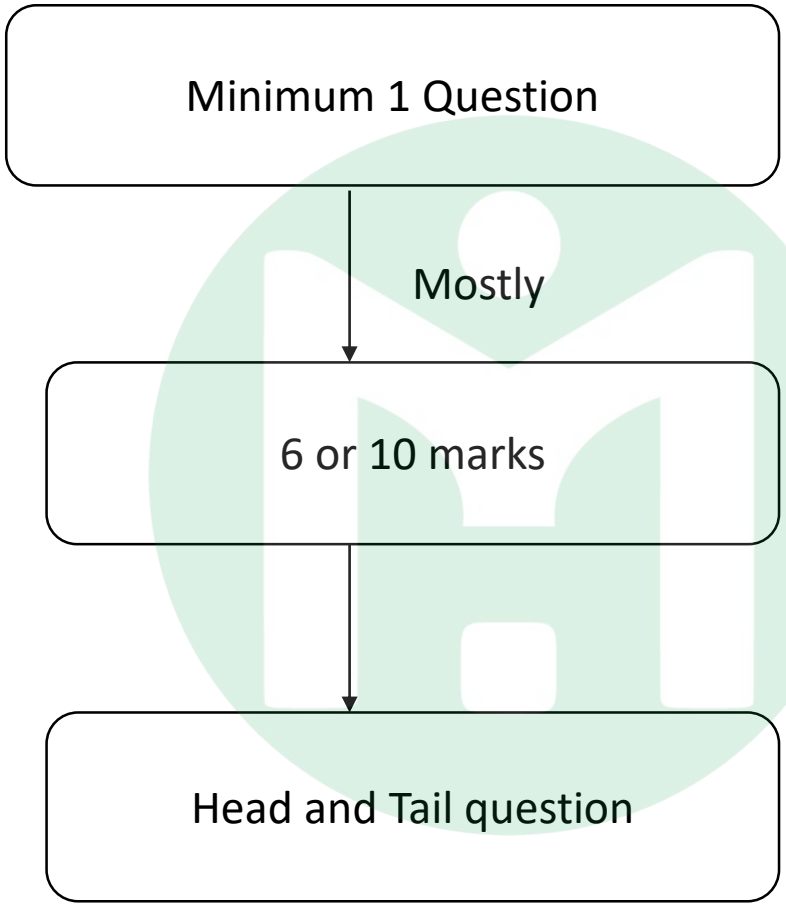
For Hamromaster Online Classes

Whatsapp: 9840842566

Hamromaster

Hamromaster

Unit 6: Database Management System



Unit 6: Database Management System

- What is Database?
- What is Database Management System?
- Advantages of using Database Approach
- Database applications
- Introduction to Database Models
- Introduction to data warehousing, Data mining, and Data Mart
- Computational Nano Science
- Space Data
- Computational Biology

What is Database?

A **database** is an organized collection of data that can be easily **accessed, managed, and updated**. It stores information in a structured format, typically in tables made up of rows and columns, allowing users and applications to retrieve and manipulate the data efficiently.

Key Points:

- Data: Refers to facts or information stored (e.g., names, dates, prices).
- Database Management System (DBMS): Software like MySQL, or Oracle that helps manage and interact with databases.
- Uses: Found in nearly every domain, from banking and education to social media and healthcare.

Key Characteristics:

- **Structured:** Data is organized into tables (relational databases), documents (NoSQL), or other formats.
- **Persistent:** Stored electronically for long-term use.
- **Managed:** Controlled via schemas defining data types, relationships, and constraints.

Example:
A student database might have a table like:

StudentID	Name	Age	Major
100	Ram	25	Biology
101	Shyam	26	Chemistry

What is Database Management System (DBMS)?

A **DBMS** is a software system that acts as an interface between users (or applications) and the database.
It manages data, the database engine, and the database schema to facilitate the organization and manipulation of data.

Functions of a DBMS:

- ❑ Data Storage: Efficiently stores large volumes of data.
- ❑ Data Retrieval: Provides query languages (e.g., SQL) to fetch specific data.
- ❑ Data Manipulation: Allows insertion, deletion, and updating of data.
- ❑ Data Security: Controls access to data using authentication and authorization mechanisms.
- ❑ Backup and Recovery: Ensures data is safe from loss and can be recovered if needed.
- ❑ Concurrency Control: Manages simultaneous access by multiple users.
- ❑ Data Integrity: Maintains accuracy and consistency through constraints and rules.

Advantages of DBMS:

- ❑ Improved data sharing and data security.
- ❑ Reduced data redundancy and inconsistency.
- ❑ Easier data access through query languages like SQL.
- ❑ Better backup and recovery mechanisms.
- ❑ Support for multi-user access and transactions.

Types of DBMS:

Relational DBMS (RDBMS):	Uses tables with rows and columns. E.g., MySQL, PostgreSQL, Oracle.
NoSQL DBMS:	For unstructured or semi-structured data. E.g., MongoDB, Cassandra.
Hierarchical DBMS:	Organizes data in a tree-like structure.
Network DBMS:	Uses graph structures with more complex relationships.
Object-oriented DBMS:	Stores data as objects, like in OOP languages.

Popular DBMS Examples:

MySQL – Open-source RDBMS used in web applications.

PostgreSQL – Advanced RDBMS with strong support for standards.

Oracle Database – Enterprise-grade RDBMS.

Advantages of using Database Approach

1. Reduced Data Redundancy
- ❑ Data is stored in a centralized location.
- ❑ Eliminates duplicate copies across files.
- ❑ Example: A customer's name is stored only once and reused wherever needed.

2. Improved Data Consistency
- ❑ When data is updated in one place, it reflects everywhere.
- ❑ Ensures uniformity across the system.
- ❑ Example: Updating an address updates it across all relevant tables.

3. Better Data Integrity
- ❑ Built-in rules and constraints (e.g., primary keys, foreign keys).
- ❑ Helps maintain accuracy and validity of data.

4. Enhanced Data Security
- ❑ Access can be restricted using user roles and permissions.
- ❑ Protects sensitive data from unauthorized access.
- ❑ Example: HR staff can access employee records but not salary details.

5. Efficient Data Access and Querying
- ❑ Use of SQL and other query languages for quick data retrieval.
- ❑ Complex queries can be handled efficiently.

6. Improved Data Sharing
- ❑ Multiple users and applications can access the data simultaneously.
- ❑ Useful in collaborative or enterprise environments.

7. Backup and Recovery
- ❑ Most DBMSs provide automatic backup and restore features.
- ❑ Ensures data is safe even in case of failures.

8. Concurrency Control
- ❑ Supports multiple users accessing and modifying the database at the same time.
- ❑ Maintains consistency even in concurrent environments.

9. Data Independence
- ❑ The application logic is separate from data structure.
- ❑ Changes to data format or structure don’t necessarily require rewriting the whole application.

10. Scalability and Flexibility
- ❑ Easily handles growing amounts of data and users.
- ❑ Adaptable to new requirements and technologies.

Database applications

1. Banking and Finance

Use Cases: Customer account management, Transactions (withdrawals, deposits, transfers) and Loan processing and credit history

Why databases?

- ❑ High-volume data processing
- ❑ Secure, real-time access and auditing
- ❑ Example: Core banking systems (CBS), ATM networks

2. Education

Use Cases: Student information systems (SIS), Attendance tracking and Course and grade management

Why databases?

- ❑ Centralized management of student data
- ❑ Secure handling of academic records
- ❑ Example: University database systems for results, registration, and timetables

3. Healthcare

Use Cases: Patient records (Electronic Health Records – EHR), Billing systems and Laboratory and pharmacy databases

Why databases?

- ❑ Accurate, confidential storage
- ❑ Easy retrieval for emergency care
- ❑ Example: Hospital Management Information Systems (HMIS)

4. E-Commerce and Retail

Use Cases: Inventory management, Customer data and order processing and Recommendation systems

Why databases?

- ❑ Real-time updates on stock and sales
- ❑ Personalized marketing using customer data
- ❑ Example: Amazon, Flipkart – product catalog and order tracking systems

5. Government

Use Cases: Citizen databases (passports, IDs), Tax records and Public service delivery systems

Why databases?

- ❑ Efficient handling of large populations
- ❑ Policy planning through data analysis
- ❑ Example: National identity databases, voter registration systems

6. Transportation

Use Cases: Vehicle tracking, Ticketing and booking systems and Route optimization

Why databases?

- ❑ Real-time information
- ❑ Efficient operations and customer service
- ❑ Example: Airline reservation systems, Uber/Lyft backend systems

Introduction to Database Models

A database model defines the logical structure of a database, how data is stored, organized, and accessed.

Why?

- ❑ Provide a blueprint for designing and implementing databases.
- ❑ Help manage data relationships, efficiency, and integrity.
- ❑ Different models suit different types of applications and data types.

Types of Database Models:

- ❑ Hierarchical Model
- ❑ Network Model
- ❑ Relational Model
- ❑ Entity-Relationship (ER) Model

1. Hierarchical Model

Structure: Tree-like structure with parent-child relationships.
Each parent can have multiple children, but each child has only one parent.
Data is stored in records connected through links.

Example:
A company database:

Company
├── Department
│ ├── Employee

Advantages:

- ❑ Simple and fast for one-to-many relationships.
- ❑ Easy to navigate if data structure is predictable.

Limitations:

- ❑ Rigid structure.
- ❑ Poor support for many-to-many relationships.
- ❑ Difficult to reorganize.

2. Network Database Model

Description:

- ❑ Similar to hierarchical but more flexible.
- ❑ Allows each child to have multiple parents.
- ❑ Data is represented using graphs.

Example:
College system:

- ❑ A student can enroll in multiple courses
- ❑ A course can have multiple students

Use Cases:
Complex relationships (e.g., manufacturing, telecom)

3. Relational Database Model (RDBMS)

Description:

- ❑ Most widely used model today.
- ❑ Data is stored in tables (relations) with rows and columns.
- ❑ Tables are connected using keys (primary and foreign keys).
- ❑ Uses Structured Query Language (SQL) for interaction.

Use Cases:

Web applications, banking, e-commerce, universities

Examples of RDBMS:

MySQL, PostgreSQL, Oracle, SQL Server

Example:

Students Table

StudentID	Name	Major
401	Ram	Physics

Courses Table

CourseID	Title
P401	Database

Enrollments Table

StudentID	CourseID
401	P401

4. Entity-Relationship (ER) Model

An ER Model uses ER diagrams to describe:

- ❑ Entities (things or objects),
- ❑ Attributes (properties of entities),
- ❑ Relationships (connections between entities).
- ❑ It provides a blueprint for designing relational databases.

Components

Entities:

- ❑ Represent real-world objects or concepts (e.g.,Teacher, Employee, Course).
- ❑ Represented as rectangles in ER diagrams.

Attributes:

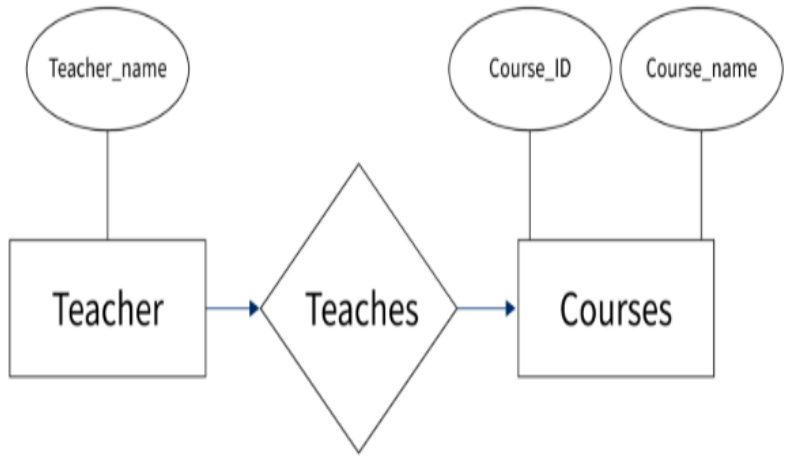
- ❑ Properties or characteristics of entities (e.g., Name, Age, Course_name).
- ❑ Represented as ovals in ER diagrams.
- ❑ Primary Key: A unique identifier for an entity (e.g., Course_ID).

Relationships:

- ❑ Connections between entities (e.g., Teaches).
- ❑ Represented as diamonds in ER diagrams.

Types of relationships: One-to-One, One-to-Many and Many-to-Many

Example



Introduction to data warehousing, Data mining, and Data Mart

1. Data Warehousing

A Data Warehouse is a centralized repository that stores large volumes of integrated, historical, and subject-oriented data from multiple sources. It is optimized for querying and analysis, not transaction processing.

Key Features:

- ❑ Integrates data from multiple sources
- ❑ Stores historical data for trend analysis
- ❑ Supports decision-making and business intelligence (BI)

Advantages:

- ❑ Better data quality and consistency
- ❑ Faster reporting and analytics
- ❑ Improved business intelligence
- ❑ Enables time-based analysis (e.g., sales over years)

Example:

A retail chain's data warehouse may combine sales, customer, and inventory data from all stores for company wide performance analysis.

2. Data Mining

Data Mining is the process of discovering patterns, correlations, trends, or anomalies from large datasets using techniques from statistics, machine learning, and AI.

Key Techniques:

- ❑ Classification
- ❑ Clustering
- ❑ Association rule learning (e.g., Market Basket Analysis)
- ❑ Regression
- ❑ Anomaly detection

Advantages:

- ❑ Helps predict customer behavior
- ❑ Identifies hidden patterns and opportunities
- ❑ Supports risk management
- ❑ Drives targeted marketing and personalization

Example:

A bank uses data mining to detect fraudulent transactions by identifying patterns that deviate from normal behavior.

3. Data Mart

A Data Mart is a subset of a data warehouse, focused on a specific business line or department, like sales, marketing, or HR.

Types:

Dependent Data Mart: Extracted from an enterprise data warehouse
Independent Data Mart: Built directly from operational data

Advantages:

- ❑ Faster access for departmental users
- ❑ Easier to maintain and less expensive
- ❑ Custom-tailored to department-specific needs

Example:

A marketing data mart may include campaign performance, customer demographics, and engagement data for marketing analysis only.

Feature / Aspect	**Data Warehouse**	**Data Mining**	**Data Mart**
Definition	A centralized repository that stores integrated, historical data for analysis	The process of discovering patterns, trends, and insights from large datasets	A subset of a data warehouse focused on a specific business area
Primary Purpose	Storage and consolidation of large volumes of data	Extracting hidden patterns and knowledge from data	Provide specific, focused data access to departments or teams
Function	Data storage and organization for analytics	Data analysis and pattern discovery	Departmental-level data retrieval and reporting
Scope	Enterprise-wide (all departments)	Analytical processes across data sources	Limited to a single department (e.g., sales, HR, finance)
Usage Example	A retailer consolidates all stores' sales data for strategic planning	A bank uses data mining to detect fraudulent transactions	A marketing team uses a data mart to analyze campaign performance
Performance Focus	High-speed querying and reporting	Pattern recognition, forecasting, and classification	Quick, focused access for specific tasks
Complexity	High, needs strong data integration and architecture	Very high, involves advanced algorithms and modeling	Lower, easy to build and maintain for specific needs
Maintenance	Requires regular updates and scalability management	Continuous refinement of models and techniques	Easier to update and scale

Computational Nano Science

Computational Nanoscience is the use of computer simulations and models to study the behavior and properties of materials at the nanoscale (1–100 nanometers). It helps scientists understand atomic and molecular interactions without conducting expensive or complex experiments.

Applications:

- ❑ Designing nanomaterials for electronics or medicine
- ❑ Simulating drug delivery using nanoparticles
- ❑ Studying quantum effects in nanostructures

Tools/Methods:

- ❑ Density Functional Theory (DFT)
- ❑ Molecular Dynamics (MD) Simulations
- ❑ Monte Carlo Methods

Example:
Simulating the behavior of carbon nanotubes to develop lightweight, strong materials for aerospace.

Space Data (Computational Space Science)

Space Data refers to information collected from space missions, satellites, and telescopes. It involves processing and analyzing large volumes of data about planets, stars, black holes, and Earth's atmosphere.

Applications:

- ❑ Monitoring climate change using satellite images
- ❑ Mapping galaxies and studying the cosmic microwave background
- ❑ Supporting space exploration missions (Mars rovers, Artemis)

Tools/Technologies:

- ❑ Remote sensing
- ❑ Machine learning for astronomy
- ❑ High-performance computing (HPC)

Example:

NASA uses space data to predict solar storms that can impact Earth's communication systems.

Computational Biology

Computational Biology uses algorithms, modeling, and simulations to understand biological systems and processes. It bridges biology with computer science and data analysis.

Applications:

- ❑ Drug discovery and protein structure prediction
- ❑ Modeling the spread of infectious diseases

Tools/Methods:

- ❑ Bioinformatics tools (BLAST, FASTA)
- ❑ Molecular dynamics
- ❑ Machine learning in genomics

Example:

AlphaFold by DeepMind predicted protein structures with high accuracy, revolutionizing biology research.

Database Schema

A Database Schema is the blueprint or structure that defines how data is organized in a database.

It includes:

- ❑ Tables
- ❑ Fields/Columns
- ❑ Data types
- ❑ Relationships
- ❑ Constraints (e.g., primary keys, foreign keys)

Think of a schema as the design or layout of a building, it defines what exists but not the actual content.

Advantages	**Disadvantages**
Provides a clear structure and organization of data	Rigid structure – changes to schema can be complex and affect existing data
Helps enforce data integrity and consistency	Requires careful design upfront
Useful for database design and documentation	Not flexible for frequent schema changes
Supports data validation through constraints	Complex schemas may lead to higher development time

Database Instance

A Database Instance is the actual data stored in a database at a given point in time. It is a snapshot of the database content based on the schema.

If the schema is the layout, the instance is the furniture and people currently in the building.

Advantages	**Disadvantages**
Represents the real-time data in the database	Data can become inconsistent or outdated if not properly managed
Can be queried and updated easily	Large instances may affect performance
Reflects the current state of the database	Needs backup and recovery mechanisms

Example:
For the Student schema above, an instance would be:

StudentID	Name	Age	Major
401	Ram	25	Biology
402	Lakshman	24	Physics

Feature / Type	**Centralized Database**	**Client/Server Database**	**Distributed Database**
Location of Data	All data is stored in one single central server	Server stores data; clients request/query via network	Data is stored across multiple geographically spread sites
Architecture	Single-tier	Two-tier or three-tier architecture	Multi-node system
Performance	Can slow down under high load	Better performance via load distribution	High performance through parallel processing
Reliability	Low (if server fails, system stops)	Moderate (some failure tolerance)	High (if one site fails, others can still operate)
Maintenance	Easy to manage and control centrally	Moderate – requires network setup	Complex – requires synchronization and coordination
Example Use Case	Small office or single-location company	Web-based applications	Global organizations, telecom systems
Cost	Low initial cost	Moderate cost	High cost due to setup and coordination

Hamromaster
सुनौलो भविष्य हातमा **Hamromaster** को साथमा

Hamromaster

COMPLETE NOTES

For Hamromaster Online Classes

Whatsapp: 9840842566

Hamromaster

Hamromaster