1. Comparison Assignment: Flat File Systems vs. Relational Databases

Flat file system:

Structure: simple data storage method where all data is stored in a single table, with no connections between various data sets.

Data Redundancy: High - Since flat file databases contain only one table, multiple entries with the same information may result in data redundancy.

Relationships: There are no relationships with other data elements because there are only two dimensions: rows and columns.

Example Usage: commonly used for small-scale storage tasks such as logs, configuration files, contact lists, and simple data exports. Formats like CSV and plain text are often used to transfer data between systems or store lightweight information in a human-readable format.

Drawbacks: lack data integrity features and do not support relationships between data elements. They are prone to data redundancy and inconsistency as the data grows. Searching, updating, and maintaining large amounts of information becomes inefficient, and they are not suitable for complex querying or multi-user environments.

Relational Databases:

Structure: Tables with rows and columns, organized into schemas. Supports constraints and data types.

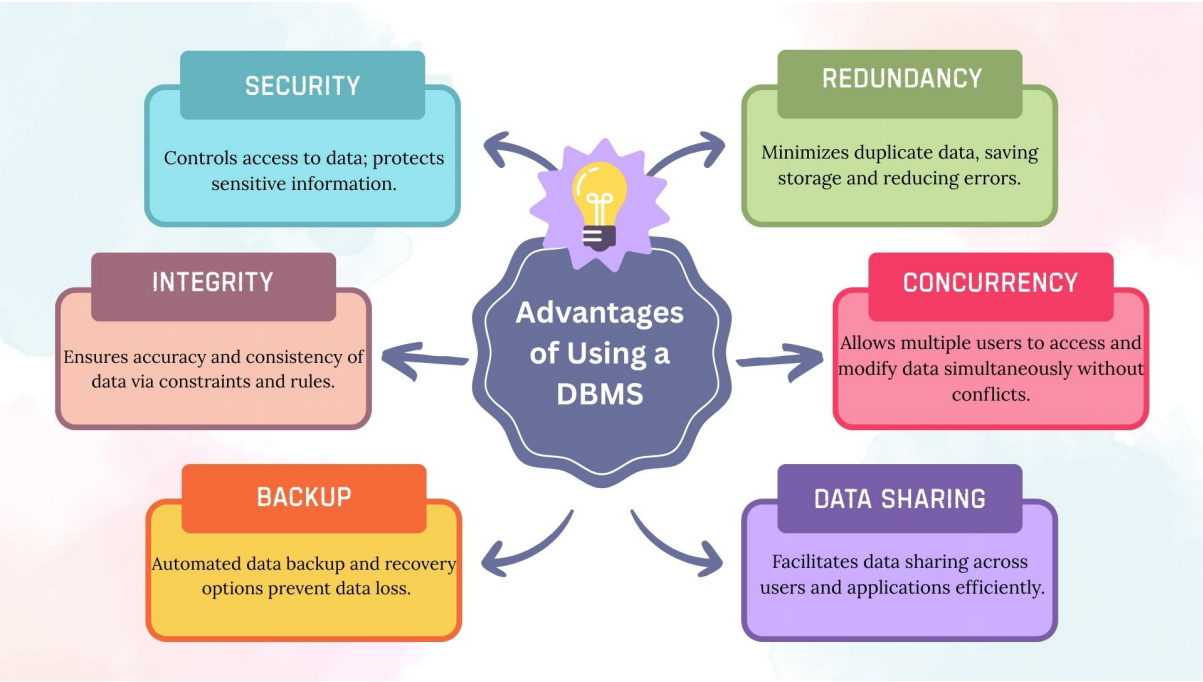
Data Redundancy: Low - normalization reduces duplication using keys and relationships.

Relationships: Relational databases organize data into tables that can be linked to one another through relationships. These relationships are defined using keys (primary and foreign), allowing data to be connected logically across multiple tables without redundancy.

Example Usage: Some of the most well-known RDBMSs include MySQL, PostgreSQL, MariaDB, Microsoft SQL Server, and Oracle Database. These systems are widely used in industries for managing structured data with relationships.

Drawbacks: require a more complex setup, including the use of a Database Management System (DBMS). They have a learning curve for SQL and relational schema design. Additionally, they can be less flexible when dealing with unstructured or hierarchical data and may face challenges in scalability and adapting to evolving data models.

2. DBMS Advantages – Mind Map



3. Roles in a Database System

System Analyst: The system analyst talks with clients to understand what they need and then writes down the requirements clearly for the team.

Database Designer: The database designer plans how the data will be organized and creates diagrams to show how different parts relate to each other.

Database Developer: The database developer builds the actual database based on the design and writes the SQL code to store and manage the data.

Database Administrator (DBA): The DBA takes care of the database's security, makes sure backups are done, and keeps everything running smoothly.

Application Developer: The application developer creates the apps that users interact with and connects those apps to the database.

Business Intelligence (BI) Developer: The BI developer analyzes data, builds reports and dashboards, and helps turn raw data into useful insights.

4. Types of Databases

Relational vs. Non-Relational Databases

- **Relational Databases**
  Relational databases store data in structured tables with rows and columns. They use SQL to query and manage the data. Relationships between tables help reduce duplication and keep things organized.
  Use case: Ideal for banking systems, inventory, or any application where data is highly structured and linked.
  Examples: MySQL, PostgreSQL, Oracle.

- **Non-Relational Databases (NoSQL)**
  These databases are more flexible — they don't use tables, and are great for handling large amounts of unstructured or semi-structured data.
  Use case: Used in social media apps, real-time analytics, or big data projects.
  Examples: MongoDB

Centralized vs. Distributed vs. Cloud Databases

- **Centralized Databases**
  All data is stored in one single location, usually on one server. It's easier to manage but can become a bottleneck or point of failure.
  Use case: Small businesses or internal applications with a limited number of users.

- **Distributed Databases**
  Data is split across multiple machines or locations. This helps improve performance and reliability, especially for global systems.
  Use case: Large e-commerce websites, global services like Netflix or Amazon.
  Example: Google Spanner, Apache Cassandra.

- **Cloud Databases**
  These are hosted online using cloud providers. They are scalable, accessible from anywhere, and maintained by the cloud provider.
  Use case: Startups, mobile apps, and modern Software as a Service platforms.
  Examples: Amazon RDS, Firebase, Google Cloud SQL.

5. Cloud Storage and Databases

Cloud storage is a service where data is saved on remote servers and accessed over the internet. Instead of storing data on a personal computer or company server, it's kept in secure data centers managed by cloud providers.This storage supports **cloud-based databases** by offering flexible, scalable space to store and manage data. It also makes databases accessible from anywhere with an internet connection.

Advantages of Using Cloud-Based Databases

- **Scalability**: Easily increase or decrease storage and performance based on needs.
- **Accessibility**: Access the database from anywhere with an internet connection.
- **Cost-Efficient**: Pay only for what you use, no need for expensive hardware.
- **Automatic Backups**: Cloud services handle backups, updates, and maintenance.
- **High Availability**: Most cloud databases offer strong uptime and reliability.

Disadvantages of Using Cloud-Based Databases

- **Internet Required**: You need a stable connection to access the database.
- **Less Control**: You depend on the provider for performance and security settings.
- **Ongoing Costs**: Monthly or usage-based fees can add up over time.
- **Data Privacy Concerns**: Sensitive data must be handled carefully to meet regulations.