Database Course Documentation

Table of Contents

List of Tables

List of Figures

1. Flat File Systems vs. Relational Databases

The following table presents a comparison between Flat File Systems and Relational Databases based on structure, redundancy, relationships, typical use cases, and limitations.

Table 1 Flat File System vs Relational Database

	Flat File System	**Relational Databases**
Structure	Stores data in a single file or table, usually plain text, without relationships between records.	Supports multiple related tables using rows, columns, primary and foreign keys.
Data Redundancy	High redundancy due to lack of normalization and relationships.	Minimal redundancy through normalization and relational integrity.
Relationships	No relationships; each record stands alone.	Supports complex relationships using keys and joins.
Example Usage	Simple storage like spreadsheets and config files.	Complex systems like CRM, inventory, banking, and enterprise applications.
Drawbacks	• High redundancy and poor scalability • Data inconsistency • Hard to maintain • No support for large or complex queries	• Hard to scale for extremely large data • Complex design and maintenance • Fixed schema • High cost • Not ideal for unstructured data • Concurrency issues • Limited real-time analytics

Flat file systems are suitable for simple, small-scale data storage, but they lack scalability, flexibility, and support for data realtionships .Relational databases, while more complex and costly, offer powerful tools for managing structured and interconnected data in large-scale applications.

2. DBMS Advantages

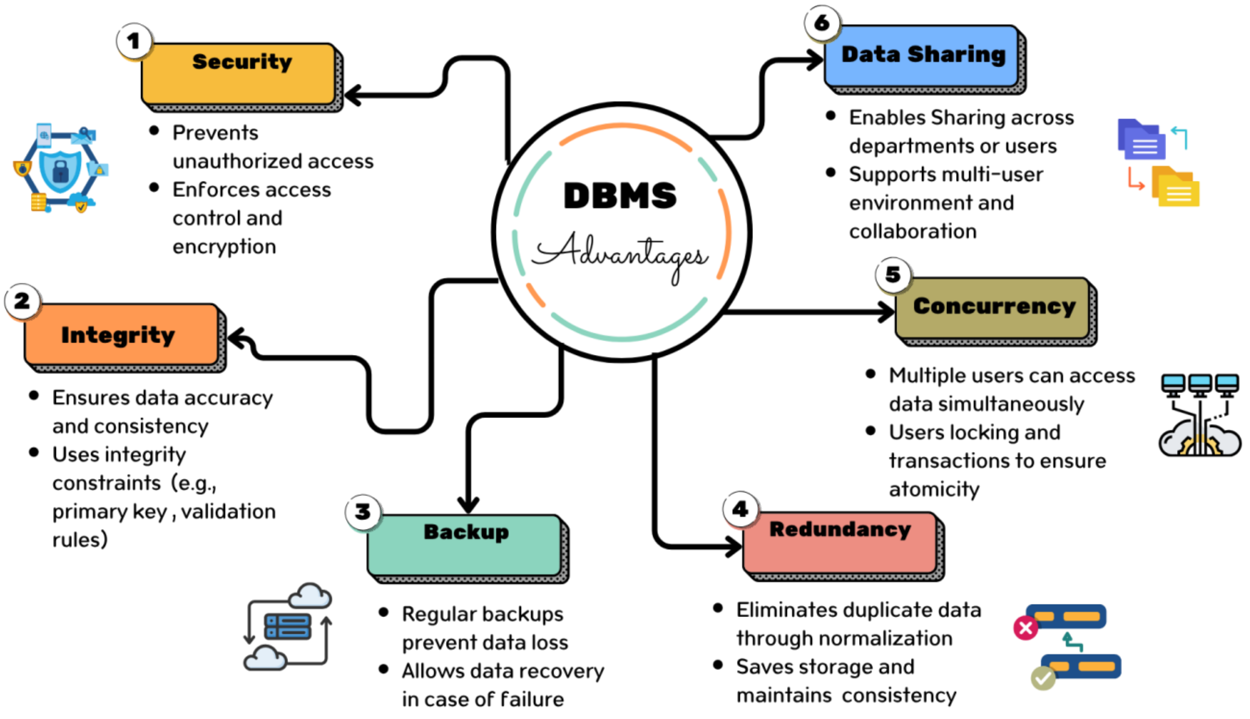
DBMS
Advantages
1
Security
Prevents unauthorized access
Enforces access control and encryption
2
Integrity
Ensures data accuracy and consistency
Uses integrity constraints (e.g., primary key , validation rules)
3
Backup
Regular backups prevent data loss
Allows data recovery in case of failure
4
Redundancy
Eliminates duplicate data through normalization
Saves storage and maintains consistency
5
Concurrency
Multiple users can access data simultaneously
Users locking and transactions to ensure atomicity
6
Data Sharing
Enables Sharing across departments or users
Supports multi-user environment and collaboration

Figure 1 DBMS Advantages

3. Roles in a Database System

In a database project, several specialized roles work together to design, implement, maintain, and use the database effectively. Below are the key roles and their responsibilities:

3.1. System Analyst:

- Gathers and defines user requirements for the database system.
- Analyzes existing processes and identifies areas for improvement.
- Bridges the gap between business needs and IT solutions.
- Creates detailed system specifications and documentation to guide development.
- Collaborates with stakeholders, designers, and developers to ensure the requirements are met throughout the project lifecycle

3.2. Database Designer:

- Designs the logical and physical structure of databases, including tables, relationships, keys, and indexes.
- Creates data models (conceptual, logical, and physical) to accurately represent business data and its relationships.

- Ensures data integrity, normalization, and efficient storage.
- Collaborates with system analysts and developers to implement database solutions based on organizational needs.
- Documents database designs and provides technical guidance.

3.3. Database Developer:

- Implements and develops the actual database systems (writing SQL, stored procedures, functions, triggers, etc.).
- Structures and manages company data within the database.
- Optimizes database performance and ensures data integrity.
- Assists in migrating, integrating, and transforming data between systems.
- Creates user documentation and provides training or support for end-users and administrators.
- Ensures backup, security, and data protection features are in place

3.4. Database Administrator (DBA):

- Installs, configures, and maintains database management systems.
- Manages database security, backups, restores, and disaster recovery.
- Monitors database performance, tunes the system, and troubleshoots issues.
- Sets user roles, permissions, and ensures compliance with security protocols.
- Establishes and enforces policies, standards, and best practices for database use and data integrity

3.5. Application Developer:

- Designs and develops software applications that interact with the database.
- Writes code to access, manipulate, and display data based on user and business requirements.
- Works with database designers and DBAs to ensure efficient and secure data access.
- Tests, debugs, and maintains applications to guarantee proper integration with the database

3.6. BI (Business Intelligence) Developer:

- Creates, develops, and maintains data analytics and reporting solutions.
- Designs ETL (Extract, Transform, Load) processes to collect and prepare data from various sources.
- Builds dashboards, reports, and visualizations to support decision-making.
- Works closely with stakeholders to transform raw data into actionable insights for the business.
- Optimizes data models and queries to ensure efficient analysis

4. **Types of Databases**
   Briefly research and describe:
   4.1. Relational vs. Non-Relational Databases

Relational Database:

Relational databases store data in structured tables with rows and columns. Each table has a fixed schema, and data is normalized to reduce redundancy. Relationships between data are enforced using primary and foreign keys, allowing for complex queries using SQL. These systems are ideal where data consistency, integrity, and structured querying are essential.

Best Uses:

- Financial and banking systems
- ERP (Enterprise Resource Planning)
- Healthcare records

Non-Relational Databases (NoSQL):

Non-relational databases offer flexible data models such as document, key-value, column-family, or graph formats. They do not rely on a fixed schema, making them more suitable for dynamic or unstructured data. NoSQL databases are designed for high scalability and performance, especially in distributed systems.

Best Uses:

- Social media and messaging apps
- IoT data collection
- Content management systems
- Real-time analytics

4.2. Example Non-Relational DBs: MongoDB, Cassandra

MongoDB:

A document-oriented NoSQL database that stores data in flexible, JSON-like documents. This allows for dynamic, nested data structures and makes it easy to store complex, hierarchical information without needing a fixed schema. It is widely used in applications that require flexibility, scalability, and quick iteration, such as content management systems, e-commerce platforms, and real-time analytics.

Apache Cassandra:

A wide-column NoSQL database built for high availability and horizontal scalability. It is optimized for storing and managing large volumes of structured data across multiple nodes in a distributed environment. Cassandra is ideal for use cases such as Internet of Things (IoT), recommendation engines, and systems that require fast, real-time data access across multiple geographic locations.

4.3. Centralized vs. Distributed vs. Cloud Databases

Databases can be deployed in different architectures depending on organizational needs. The table below compares centralized, distributed, and cloud databases based on their key features:

Table 2 Centralized vs. Distributed vs. Cloud Databases

Feature	Centralized Database	Distributed Database	Cloud Database
Definition	Entire database is stored at a single physical location.	Database is split across multiple physical locations but managed as one.	Database is hosted on cloud infrastructure by third-party providers.
Access	Accessed via a local network or intranet.	Accessed via network; supports multi-site access.	Accessed globally over the Internet.
Scalability	Limited scalability; upgrading requires hardware changes.	Highly scalable via horizontal scaling across nodes.	Very scalable; resources can be increased or decreased as needed.
Availability	Risk of single point of failure.	Higher availability; if one site fails, others can take over.	Built-in redundancy and high availability by design.
Usage	Suitable for small to medium-sized organizations with basic needs.	Used by global apps and large enterprises for performance and reliability.	Ideal for modern applications and enterprises needing global, flexible solutions.

4.4. Use Case Examples for each type

Table 3 Use Case Example

Database Type	Example Use Cases
Centralized	University student records, hospital database, small business inventory system
Distributed	Airline booking systems, global banking networks, multinational retail chains
Cloud	E-commerce websites, mobile apps, global analytics platforms, SaaS applications

5. Cloud Storage and Databases

5.1. What is Cloud Storage and how does it support database functionality?

Cloud storage is a service that allows data to be stored on remote servers accessed via the internet, typically managed by cloud providers such as Amazon Web Services (AWS), Google Cloud, or Microsoft Azure. It eliminates the need for on-premises infrastructure by offering scalable, redundant, and cost-effective storage solutions.

In the context of databases, cloud storage serves as the foundation for cloud-based databases. It enables these databases to:

- Store data in distributed environments
- Ensure high availability through replication
- Scale automatically to meet growing demands
- Allow users to access data from any location with internet connectivity

While cloud storage handles the physical and networking layers, cloud databases manage logical data structure, retrieval, and transactional operations. This combination makes cloud-based database solutions highly reliable, scalable, and accessible.

5.2. Advantages of using cloud-based databases (e.g., Azure SQL, Amazon RDS, Google Cloud Spanner)

- **Scalability:** Automatically scale up (vertical) or out (horizontal) depending on application demand.

- **High Availability & Disaster Recovery:** Built-in features like geo-replication, automated backups, and failover ensure data resilience.
- **Cost Efficiency:** Pay-as-you-go pricing eliminates upfront infrastructure costs and minimizes IT overhead.
- **Managed Services:** Cloud providers handle routine database management tasks like patching, updates, and backups.
- **Global Accessibility:** Access databases securely from anywhere, making them ideal for distributed teams and applications.
- **Strong Security:** Cloud platforms offer data encryption (at rest and in transit), role-based access control, and compliance with international security standards.

5.3. Disadvantages or challenges with cloud-based databases

- **Latency & Network Dependency**: Internet-based access can introduce delays and make uptime dependent on network stability.
- **Vendor Lock-In**: Switching cloud providers may be difficult due to service dependencies, tools, and data formats.
- **Security & Compliance Risks**: Even with provider safeguards, organizations must ensure compliance with privacy laws and protect sensitive data.
- **Cost Management**: Without proper monitoring, usage spikes and poor resource management can lead to unexpected expenses.
- **Limited Customization**: Some cloud database services restrict access to low-level settings and fine-tuning options.
- **Hybrid Complexity**: Integrating cloud and on-premises systems introduces challenges in data syncing, latency, and configuration.

6. References

1. Admin. (2025, April 29). *RDBMS vs Flat File Database: Which One Should You Choose?*https://www.encoders.co.in/blog/database/rdbms-vs-flat-file-database-which-one-should-you-choose
2. Baker College. (2025, May 30). *The Power of Database Administration: The DBA's Role in Today's Data-Driven Landscape*. https://www.baker.edu/about/get-to-know-us/blog/role-of-a-database-administrator/
3. *Flat File Database: Definition, uses, and benefits*. (2025, July 21). Airbyte. https://airbyte.com/data-engineering-resources/flat-file-database
4. Ibm. (2025a, July 30). Topics. *IBM*. https://www.ibm.com/cloud/learn/database-types
5. MongoDB. (n.d.). *What is MongoDB?* https://www.mongodb.com/what-is-mongodb
6. *What is the difference between a flat-file database and a relational database? | TutorChase*. (n.d.). https://www.tutorchase.com/answers/a-level/computer-science/what-is-the-difference-between-a-flat-file-database-and-a-relational-database
7. GeeksforGeeks. (2025, July 12). *Advantages of database management System*. GeeksforGeeks. https://www.geeksforgeeks.org/dbms/advantages-of-database-management-system/
8. *What is Cloud Storage & How Does it Work? | Google Cloud | Google Cloud*. (n.d.). Google Cloud. https://cloud.google.com/learn/what-is-cloud-storage
9. *What is Cloud Storage? - Cloud Storage Explained - AWS*. (n.d.). Amazon Web Services, Inc. https://aws.amazon.com/what-is/cloud-storage/