DATABASE

"Learning databases is an investment in the future, as they are the backbone of all modern technology."

Done By
Quds AL-Jabri

Table of Contents

Relational vs Flat File Databases **3**

Database Management System (DBMS) **5**

Meet the Database Dream Team **6**

Types of Databases **8**

Relational vs. Non-Relational Databases 8

Centralized vs. Distributed vs. Cloud Databases 10

Centralized Databases 10

Distributed Databases 11

Cloud Databases 11

Cloud Storage and Databases **12**

Relationship Between Cloud Storage and Databases 12

References **14**

Relational vs Flat File Databases

Who Wins the Data Game?

🧠 Let's Start with a Riddle:

"I am like a well-organized library with sections, cross-references, and catalog numbers. My cousin, however, is like a big messy drawer full of pages. Who am I?"
Answer: The Relational Database!

📖 The definitions:

🧩 **A relational database** is a type of database that stores and organizes data in a collection of tables. These tables are related to each other by a common field known as a primary key. Relational databases are used to store, organize and retrieve data quickly and efficiently. They are the most common type of database used in business applications

🗃️ **A flat file,** also known as a text database, stores data in plain text format and is organized as a single table with no relationships between tables. This type of database was first developed and implemented by IBM in the early 1970s

Features	🧩 Relational DB	🗃️ Flat File DB
Structure	Multiple linked tables	A single giant table
Redundancy	Low (shared data is stored once and referenced)	High (same data repeated)
Relationships	Strong (tables connect using primary/foreign keys)	None, no way to link data properly
Speed & Efficiency	Optimized for large-scale data retrieval	Slows down as data grows
Example	A database with separate tables for customers, orders, products	A .csv of all customer orders
First Use	Introduced later but now dominates modern business systems	IBM in the early 1970s

In summary, both relational databases and flat files are used to store, retrieve and organize data, both are file-based storage and can be used with a variety of platforms and programming languages and both have backup and restore capabilities. However, relational databases are more powerful, flexible, and efficient than flat files, providing advanced data management capabilities, better performance, scalability, and security

Database Management System (DBMS)

Database Management System (DBMS) is a collection of interrelated data and a set of software tools/programs that access, process, and manipulate data.

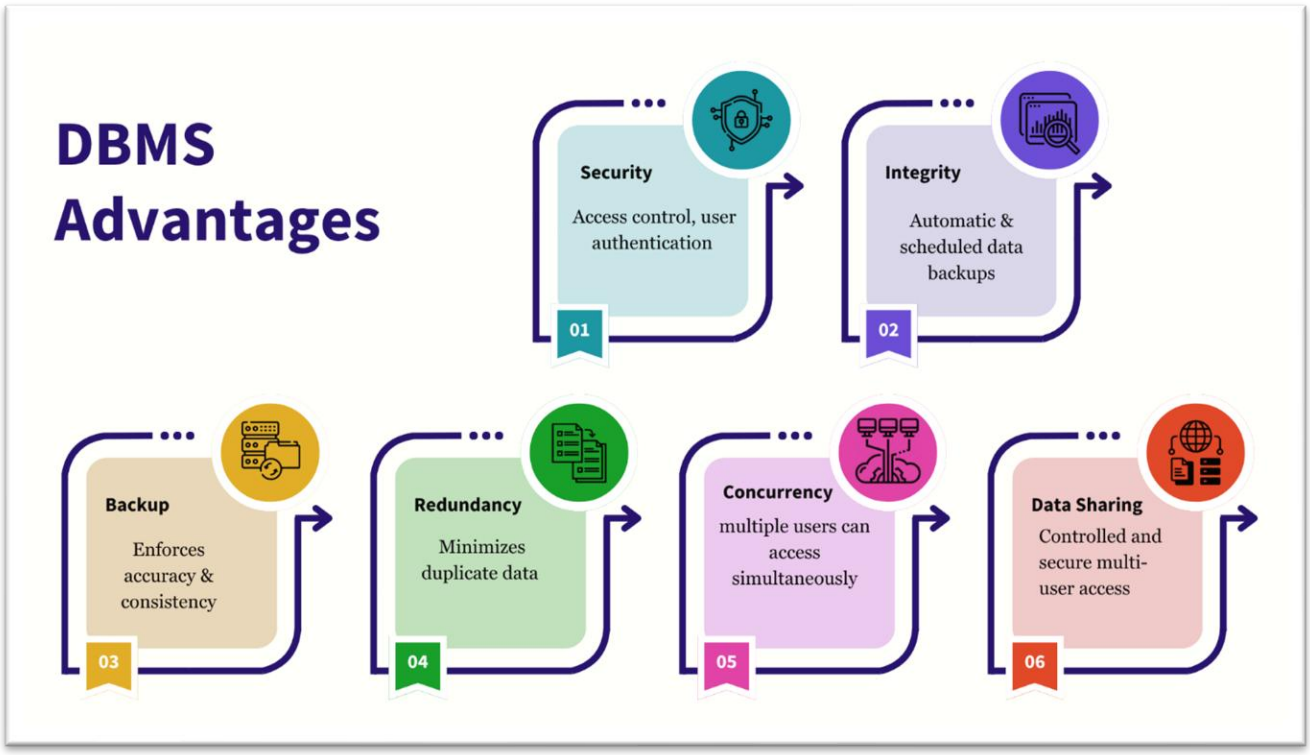

Figure 1: Advantage of DBMS

1. **Data Security:** DBMS enhances data security through access control and encryption. It enforces privacy policies and prevents unauthorized access. As user numbers grow, it mitigates the associated risks of breaches.
2. **Data integration:** DBMS unifies data from different sources into a centralized system. This provides a coherent organizational view of operations. It helps to keep track of how one segment of the company affects another segment.

3. **Improved Data Backup and Recovery**: DBMS offers automated, reliable backup and point-in-time recovery. Data can be restored after failure, maintaining consistency and availability. It supports disaster recovery strategies, ensuring business continuity. Traditional file systems often require manual, error-prone backups.

4. **Reduction in data Redundancy:** DBMS avoids data duplication by enforcing unique constraints. It removes unnecessary repetitive entries in databases. This ensures efficient use of storage and improves consistency. for e.g. - If there are two same students in different rows, then one of the duplicate data will be deleted.

5. **Concurrency and maintained Atomicity**: That means, if some operation is performed on one particular table of the database, then the change must be reflected for the entire database. DBMS allows concurrent access to multiple users by using the synchronization technique.

6. **Data sharing:** A DBMS provides a platform for sharing data across multiple applications and users, which can increase productivity and collaboration.

Meet the Database Dream Team 🎭

Think of a database project like a movie production. Each role has a key part to play from writing the script to editing the final cut!

🎯 1. System Analyst (The Director)

"Lights, camera, data!"

- Translates business goals into system requirements.
- Ensures everyone understands the *story* (what the database must do).
- Coordinates between business people and tech folks.

  🧠 *Asks the right questions to frame the big picture.*

🧱 2. Database Designer (The Architect)

"Blueprints before bricks."

- Designs the structure: tables, fields, relationships.
- Plans how data is stored, connected, and organized.
- Creates ER diagrams and normalizes tables.
  📐 *Thinks like a city planner clean, efficient design.*

👷 3. Database Developer (The Builder)

"Bringing the design to life."

- Writes SQL queries, stored procedures, and triggers.
- Implements data logic and rules.
- Works hands-on with real data to make things run.
  🛠️ *Turns the blueprint into a working system.*

🔐 4. Database Administrator (DBA) (The Security Chief)

"No data left behind."

- Manage installation, backups, performance, and access control.
- Protects the system from crashes and intruders.
- Ensures data availability 24/7.
  🛡️ *Keeps the database healthy and safe.*

💻 5. Application Developer (The Frontline Storyteller)

"Putting data in the hands of users."

- Builds web/mobile apps that talk to the database.
- Ensures smooth user experience and accurate data display.
- Integrates backend and frontend.
  📲 *Makes sure users love the app, not just the data.*

📊 6. BI Developer (The Data Detective)

"Turning numbers into knowledge."

- Creates dashboards, charts, and data stories.
- Helps businesses make smart decisions with clean insights.
- Works with tools like Power BI, Tableau, or Looker.
  🔎 *Sees the patterns others miss.*

Types of Databases

Relational vs. Non-Relational Databases 🆚

Choosing the Right Engine for Your Data Journey

In the world of data, not all engines are built the same. Some offer structure and precision like an orchestra; others thrive in chaos, adapting fluidly like jazz. Below is a side-by-side comparison that demystifies the two dominant database types: Relational (SQL) and Non-Relational (NoSQL).

Feature	Relational Database (SQL)	Non-Relational Database (NoSQL)
Data Structure	Tables (rows and columns)	Collections, documents, key-value pairs, graphs
Schema	Fixed and pre-defined	Dynamic and flexible
Best For	Structured data, transactional systems (e.g., finance, HR)	Unstructured or evolving data (e.g., social media, IoT, real-time apps)
Examples	MySQL, PostgreSQL, Oracle, SQL Server	MongoDB, Cassandra, Redis, DynamoDB
Query Language	SQL (Structured Query Language)	Varies MongoDB uses JSON-like queries, Redis uses commands, etc.
Scalability	Vertical scaling (scale-up: more CPU/RAM)	Horizontal scaling (scale-out: more nodes/machines)
Data Integrity	High – supports ACID (Atomicity, Consistency, Isolation, Durability)	Eventual consistency – often trades strict rules for speed and agility
Performance on Large Data Volumes	Can slow down with huge unstructured data	Optimized for high volume, velocity, and variety
Storage Style	Relational (normalized) – minimal redundancy	Denormalized – redundancy is acceptable for faster access
Use Case Analogy	🧱 Brick building – structured, solid, every block in place	🌱 Vine growth – organic, spreads out, adapts as it grows

🧠 When to Use What?

Choose a Relational Database when:

- You need reliable transactions (e.g., banking).
- Data has clear relationships (e.g., customers → orders).
- You require strict schema enforcement.

Choose a Non-Relational Database when:

- You deal with big data or real-time analytics.
- Your data structure evolves frequently.
- You need extreme scalability and availability.

🎯 The Bottom Line

Both relational and non-relational databases are powerful but like choosing between a DSLR and a GoPro, the right choice depends on the environment. In fast-moving, high-volume systems, flexibility and speed matter. In critical systems where precision is everything, structure wins.

The smartest systems today? They often use both choosing each engine where it excels.

Centralized vs. Distributed vs. Cloud Databases 🌎

Understanding the Architecture Behind Data Access and Control

In the evolving landscape of data management, how and **where** data is stored is just as critical as what data is stored. Database architecture plays a fundamental role in determining **performance**, **resilience**, and **accessibility**. Here's a deep dive into three major architectures: **Centralized**, **Distributed**, and **Cloud-based** databases.

Centralized Databases 🏢

"All under one roof."

A centralized database stores all data in a single location — typically one server or data center. This architecture is simple, cost-efficient for small-scale operations, and easy to manage.

✅ **Advantages:**

- Simplified data management and security
- Consistent data versioning
- Easier backup and maintenance

⚠️ **Limitations:**

- Single point of failure system crashes can bring everything down
- Latency issues for geographically distant users
- Limited scalability hardware constraints can become bottlenecks

💼 **Best For:**
Internal systems, small businesses, or environments where data integrity and control are top priorities.

Distributed Databases 🌐

"United but spread out."

A **distributed database** stores data across **multiple nodes** or locations. These systems behave as a unified database to users, but data is physically stored across a network.

✅ **Advantages:**

- **Fault tolerance** system continues running even if one node fails
- **Faster access** for users near data nodes
- **Scalable horizontally** just add more nodes

⚠️ **Challenges:**

- Complexity in synchronization and consistency
- Higher setup and maintenance effort
- More complicated security enforcement

💼 **Best For:**
Enterprises with global operations, content delivery networks, or applications that demand high availability.

Cloud Databases ☁️

"Data on demand, powered by the cloud."

Cloud databases are managed by third-party providers like AWS, Azure, or Google Cloud, and accessible over the internet. These databases can be relational or non-relational, and offer highly scalable, flexible, and cost-efficient solutions.

✅ **Advantages:**

- On-demand scalability
- No physical infrastructure needed
- Built-in redundancy and disaster recovery
- Global accessibility

⚠️ **Challenges:**

- Ongoing subscription costs
- Vendor lock-in switching providers can be difficult
- Compliance and data sovereignty concerns

💼 **Best For:**
Modern apps, startups, global SaaS platforms, or any system requiring agility, uptime, and scalability.

Cloud Storage and Databases 🌩️

Relationship Between Cloud Storage and Databases 🔗

- **Cloud Storage** is the foundation for storing data in the cloud.
- **Databases** (like Azure SQL, Amazon RDS, Google Cloud Spanner) use cloud storage to:
    1. Store structured data
    2. Maintain backups and logs
    3. Enable high availability and disaster recovery
    4. Scale dynamically with demand

☑ Advantages of Cloud-Based Databases

- Scalability: Easily scale resources up or down based on demand.
- Security: Built-in encryption, access control, and compliance support.
- Managed Services: Automatic updates, backups, and maintenance reduce admin overhead.
- Global Access: Access your database from anywhere with internet connectivity.
- Cost Efficiency: Pay-as-you-go pricing reduces upfront investment.

⚠ Challenges of Cloud-Based Databases

- Latency: Network delays can affect performance, especially for real-time applications.
- Vendor Lock-In: Switching providers can be complex and costly.
- Limited Customization: Some low-level configurations may not be accessible.
- Downtime Risks: Cloud outages, though rare, can impact availability.
- Cost Management: Unexpected usage spikes can lead to high bills.

References

1- Admin. (2024, July 16). Relational Database vs Flat File (Differences & Similarities). *DatabaseTown*.

2- GeeksforGeeks. (2024, September 30). *Advantages of database management System*. GeeksforGeeks.

3- Staff, E. (2024, May 13). *Flat File vs. Relational Databases: Securing LBM Customer Data*. ECI Software Solutions.

4- Lemonaki, D. (2022, April 18). *Relational VS Nonrelational Databases – The difference between a SQL DB and a NoSQL DB*. freeCodeCamp.org.

5- Ibm. (2025, April 16). Cloud database. *What is a cloud database?*

6- GeeksforGeeks. (2025, April 15). *Types of databases*. GeeksforGeeks.