Database Search and Reporting

Submitted by: Deena Al Busaidi

Submission Date: 16th June 2025

Course: Data Science

Contents

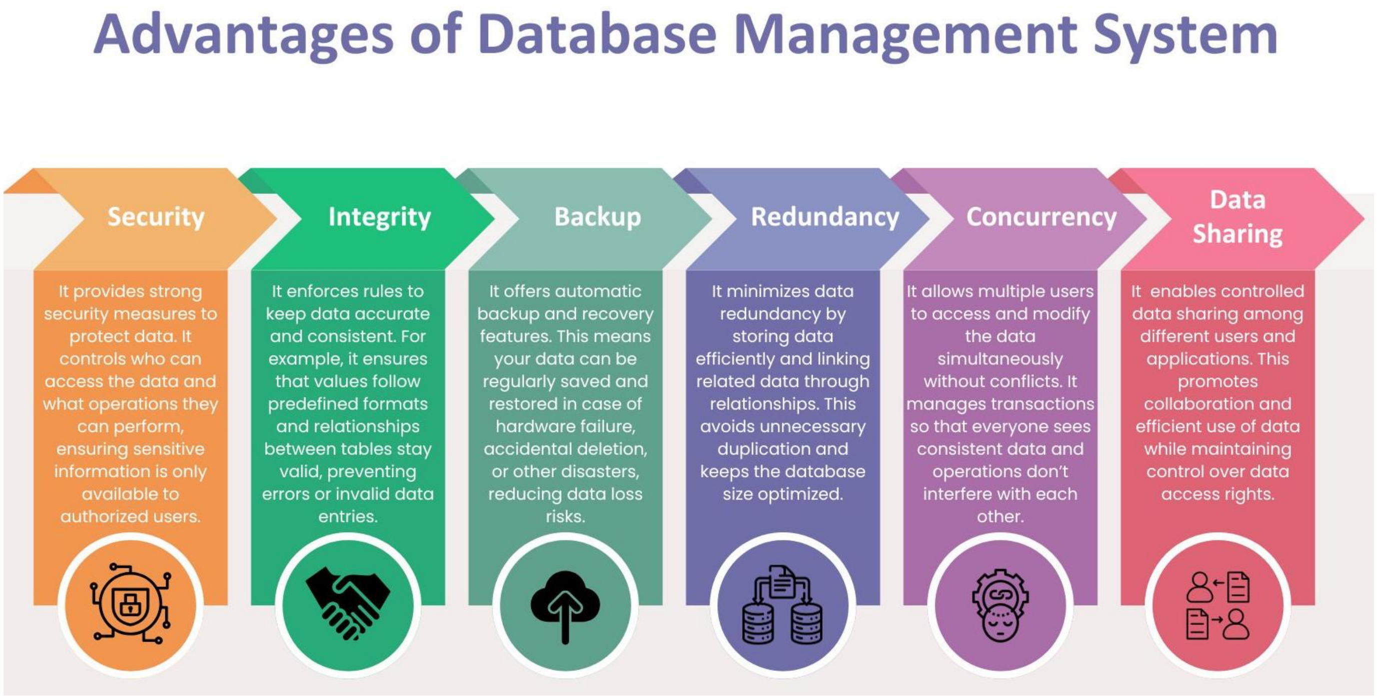
Flat File Systems vs. Relational Databases

	Flat File System	Relational Database
Structure	• Single table/file format. • Data is stored in rows and columns in a plain text file (like .csv or .txt). • There is no internal linking between different sets of data. • Every file is independent and self-contained.	• Data is split into multiple related tables**.** • Each table stores data about one type of things such as (students, classes, teachers). • Tables are linked by keys such as (StudentID, ClassID) to avoid duplication and organize data better. • Managed by a Database Management System (DBMS) like MySQL, PostgreSQL, or SQLite.
Data Redundancy	• High data redundancy (lots of repeated data). • Same information (e.g., class name, teacher) is repeated in multiple records. • Any update must be made in every place where the data appears. • Increases the risk of errors and inconsistencies. • Uses more storage space due to duplication.	• Low data redundancy (minimal repetition). • Shared information is stored once in separate, related tables. • Updates are made in one place and reflected everywhere through relationships. • Reduces chances of inconsistencies and data entry errors. • More efficient in terms of storage and maintenance.
Relationships	• Do not support relationships between data. • All information is stored in a single file, even if it's related to multiple subjects (e.g., students, teachers, classes). • Cannot link data across different files or tables. • Difficult to represent complex data structures or associations. • Makes it harder to organize or update related data efficiently.	• Support relationships between different tables using keys (e.g., primary and foreign keys). • Data is separated into tables based on type (e.g., Students, Courses, Instructors), and linked logically. • Makes it easy to connect, query, and manage related data. • Useful for representing real-world relationships (e.g., one student enrolled in multiple classes). • Improves data integrity and organization in large or complex systems.

Example Usage	• Small-scale data storage needs. • Simple applications where data relationships are not important. • Storing contact lists, product catalogs, or employee records in CSV or Excel files. • Exporting or importing data between systems in a basic format. • Ideal for one-time or read-only data tasks.	• Complex systems with large volumes of data. • Applications that require structured relationships (e.g., one-to-many, many-to-many). • Used in websites, mobile apps, banking systems, hospital management, inventory control, and more. • Suitable for multi-user access, reporting, and dynamic queries. • Essential for any system where data integrity, relationships, and scalability matter.
Drawbacks	• High data redundancy due to repeated information. • No support for relationships, making complex data management difficult. • Poor scalability — not ideal for large or growing data sets. • Harder to maintain and update data consistently. • Limited querying and filtering capabilities. • No built-in data validation or security features.	• More complex setup and structure — requires planning and design. • Needs database software (e.g., MySQL, PostgreSQL). • Requires knowledge of SQL or a query language to use effectively. • Can be overkill for very small or simple tasks. • May require more system resources (memory, processing) compared to flat files.
Examples	**Example 1:** Name, Age, Department, Manager Alice, 30, Sales, John Bob, 28, Sales, John Carol, 35, HR, Mike **Example 2:** StudentName, Subject, Grade Tom, Math, A Tom, Science, B Sara, Math, A	**Example 1:** **Library Management System** • Books Table: BookID, Title, AuthorID • Authors Table: AuthorID, Name • Borrowers Table: BorrowerID, Name • Loans Table: LoanID, BookID, BorrowerID, Date **Example 2:** **University Database System** • Students Table: StudentID, Name, Age • Courses Table: CourseID, Name, InstructorID • Enrollments Table: StudentID, CourseID

Database Management System (DMS)

Advantages



Advantages of Database Management System

Security

It provides strong security measures to protect data. It controls who can access the data and what operations they can perform, ensuring sensitive information is only available to authorized users.

Integrity

It enforces rules to keep data accurate and consistent. For example, it ensures that values follow predefined formats and relationships between tables stay valid, preventing errors or invalid data entries.

Backup

It offers automatic backup and recovery features. This means your data can be regularly saved and restored in case of hardware failure, accidental deletion, or other disasters, reducing data loss risks.

Redundancy

It minimizes data redundancy by storing data efficiently and linking related data through relationships. This avoids unnecessary duplication and keeps the database size optimized.

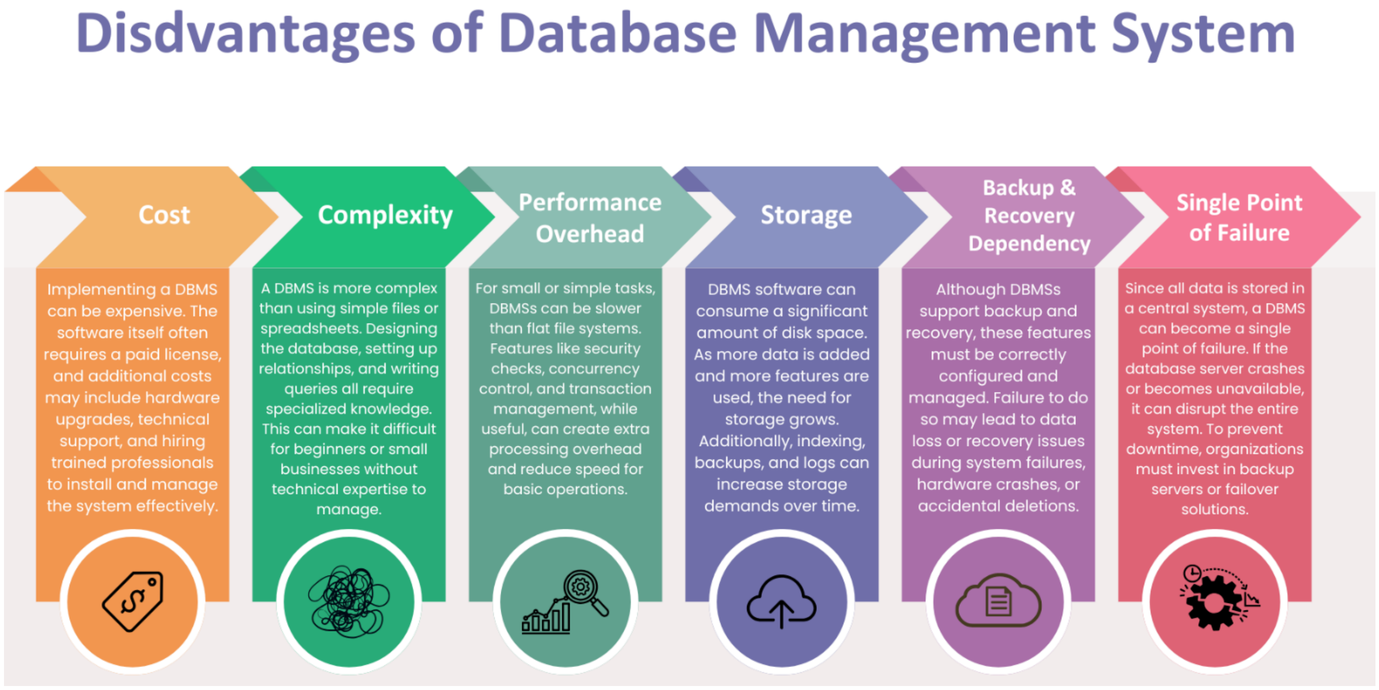
Concurrency

It allows multiple users to access and modify the data simultaneously without conflicts. It manages transactions so that everyone sees consistent data and operations don't interfere with each other.

Data Sharing

It enables controlled data sharing among different users and applications. This promotes collaboration and efficient use of data while maintaining control over data access rights.

Disadvantages



Disdvantages of Database Management System

Cost

Implementing a DBMS can be expensive. The software itself often requires a paid license, and additional costs may include hardware upgrades, technical support, and hiring trained professionals to install and manage the system effectively.

Complexity

A DBMS is more complex than using simple files or spreadsheets. Designing the database, setting up relationships, and writing queries all require specialized knowledge. This can make it difficult for beginners or small businesses without technical expertise to manage.

Performance Overhead

For small or simple tasks, DBMSs can be slower than flat file systems. Features like security checks, concurrency control, and transaction management, while useful, can create extra processing overhead and reduce speed for basic operations.

Storage

DBMS software can consume a significant amount of disk space. As more data is added and more features are used, the need for storage grows. Additionally, indexing, backups, and logs can increase storage demands over time.

Backup & Recovery Dependency

Although DBMSs support backup and recovery, these features must be correctly configured and managed. Failure to do so may lead to data loss or recovery issues during system failures, hardware crashes, or accidental deletions.

Single Point of Failure

Since all data is stored in a central system, a DBMS can become a single point of failure. If the database server crashes or becomes unavailable, it can disrupt the entire system. To prevent downtime, organizations must invest in backup servers or failover solutions.

Roles In a Database System

System Analyst

- **A system analyst connects business needs with technical solutions by gathering user requirements, analyzing workflows, and helping design a high-level plan for the database system to improve efficiency and solve problems.**

Database Designer

- **A database designer plans the database structure by defining tables, relationships, data types, and constraints to create an efficient, organized, and normalized design that meets business needs.**

Database Developer

- **A database developer creates the actual database using SQL code to build tables, views, and procedures, and develops automated functions, focusing on the technical implementation of the database design.**

Database Adminstrator

- **A Database Administrator (DBA) manages the database's performance, security, and reliability by handling user access, backups, recovery, and system monitoring to ensure smooth and secure operations.**

Application Developer

- **An application developer builds software or web apps that interact with the database, allowing users to access and manage data, while ensuring smooth integration and a user-friendly experience.**

BI Developer

- **The BI developer focuses on analyzing data and generating insights. They design and build dashboards, reports, and data visualizations using tools like Power BI, Tableau, or SQL-based reporting tools. Their goal is to help decision-makers understand the data and use it to guide business strategy.**

Types of Databases

1. Relational vs. Non-Relational Databases

	Rational Database	Non-Rational Database
Definition	Relational databases store data in tables (rows and columns), similar to spreadsheets. Each table contains records, and each record has fields with defined data types. These tables can be connected through keys (primary and foreign keys), allowing relationships between data to be maintained. Relational databases use SQL (Structured Query Language) for creating, reading, updating, and deleting data.	Non-relational databases don't use the traditional table format. Instead, they are more flexible, storing data in various formats like documents, key-value pairs, graphs, or wide-column stores. They allow for schema-less design, making them ideal for handling unstructured or rapidly changing data. These databases often scale better horizontally and are used in high-performance environments.
Examples	• **MySQL:** Popular open-source database for web applications. • **PostgreSQL:** Open-source and advanced relational DB. • **Oracle Database:** Often used in large enterprise systems. • **Microsoft SQL Server:** Common in Windows-based environments.	• **Document-Based:** MongoDB – stores data in JSON-like documents, flexible schema. • **Wide-column store:** Cassandra – stores data across many machines; great for massive, distributed systems. • **Key-value stores**: Redis – stores simple key-value pairs, used for caching. • **Graph databases**: Neo4j – manages data through nodes and relationships, perfect for network data.
Use Cases	• **Banking systems:** strict consistency and relationships needed. • **School management systems:** organized data like students, courses, and grades. • **Inventory systems:** linking products, suppliers, and transactions.	• **Social media apps:** handle diverse user data and large volumes. • **Real-time analytics:** fast and flexible data structures. • **IoT data:** semi-structured sensor data from many devices.

2. Centralized vs. Distributed vs. Cloud Databases

	Centralized Database	Distributed Database	Cloud Databases
Definition	A centralized database is stored and managed in a single location, often on one server or in one data center. Users from different locations access this central system. It's simpler to manage and maintain but comes with risks — like downtime if the server fails.	In a distributed database, data is stored on multiple servers or locations. These servers may be in different parts of the world but work together as a single system. Distributed databases provide fault tolerance, better performance, and high availability.	Cloud databases are hosted online using cloud services like AWS (Amazon Web Services), Google Cloud Platform, or Microsoft Azure. These databases can be relational (like Amazon RDS for SQL) or non-relational (like Firebase or Amazon DynamoDB). Cloud databases offer scalability, automatic updates, backup, and global access.
Advantages	• Easier security management. • Consistent performance due to centralized control.	• Reduces downtime (if one server fails, others still work). • Faster access based on user location. • Can handle more users and larger datasets.	• Easy to scale up or down. • Accessible from anywhere with the internet. • Reduces the need for in-house infrastructure.
Drawback	• Single point of failure. • Slower access for users far from the central location.	• More complex to manage and synchronize data. • Requires robust networking and error handling.	• Dependent on internet connectivity. • May involve ongoing subscription costs. • Security depends on cloud provider's configuration.
Use Cases	• Small companies with limited data. • Internal systems that don't need global access.	• E-commerce platforms (like Amazon). • Global enterprise systems and financial institutions.	• Web and mobile applications. • SaaS platforms (e.g., Zoom, Trello). • Online learning and collaboration tools.

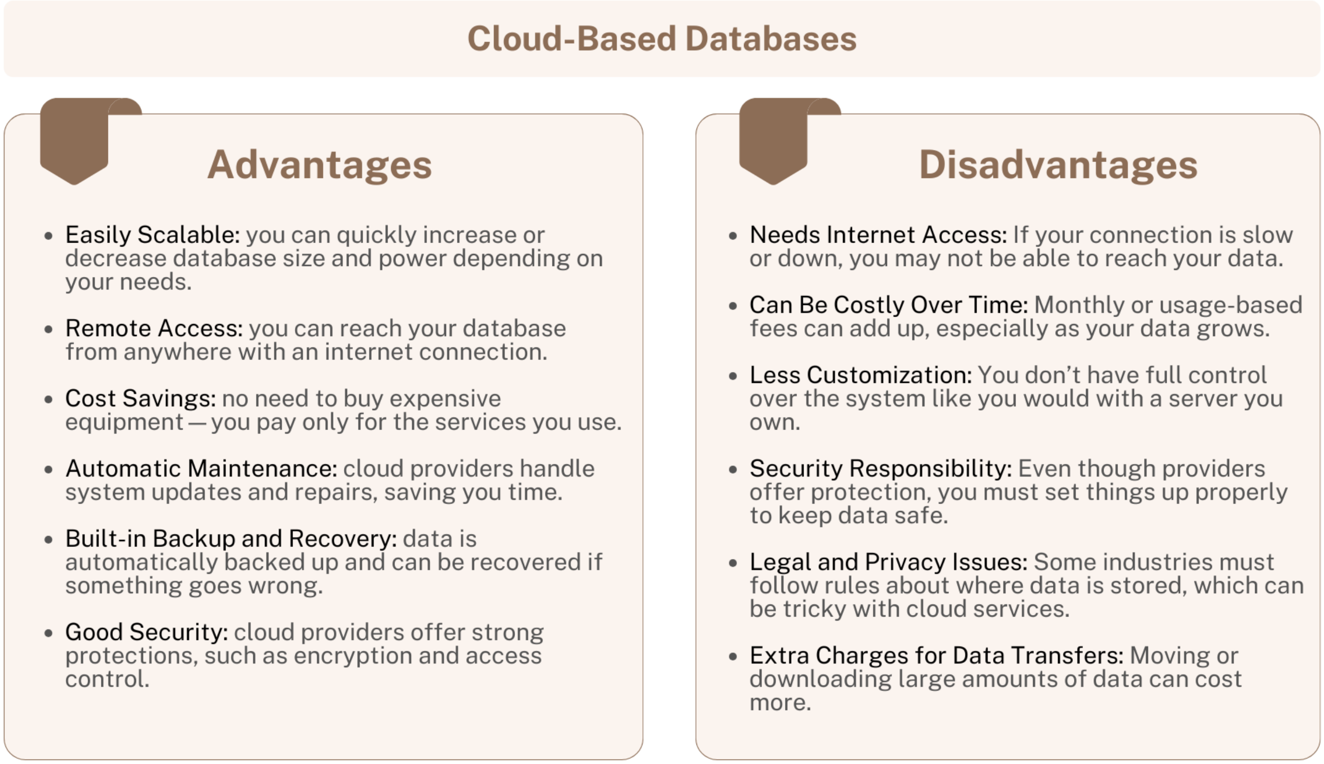
Cloud Storage and Databases

1. Cloud Storage and How It Helps Databases

Cloud storage means saving data on the internet instead of on your personal device or office server. This data is stored on powerful computers managed by companies like Amazon (AWS), Microsoft (Azure), or Google Cloud.

While cloud storage is often used for saving files and backups, it also supports databases by giving them space and resources to run. When a database runs on the cloud, it's called a cloud database. These databases benefit from the cloud's flexibility, fast access, and backup systems.

2. Advantages & Disadvantages of using cloud-based databases



Cloud-Based Databases

Advantages

- Easily Scalable: you can quickly increase or decrease database size and power depending on your needs.
- Remote Access: you can reach your database from anywhere with an internet connection.
- Cost Savings: no need to buy expensive equipment—you pay only for the services you use.
- Automatic Maintenance: cloud providers handle system updates and repairs, saving you time.
- Built-in Backup and Recovery: data is automatically backed up and can be recovered if something goes wrong.
- Good Security: cloud providers offer strong protections, such as encryption and access control.

Disadvantages

- Needs Internet Access: If your connection is slow or down, you may not be able to reach your data.
- Can Be Costly Over Time: Monthly or usage-based fees can add up, especially as your data grows.
- Less Customization: You don't have full control over the system like you would with a server you own.
- Security Responsibility: Even though providers offer protection, you must set things up properly to keep data safe.
- Legal and Privacy Issues: Some industries must follow rules about where data is stored, which can be tricky with cloud services.
- Extra Charges for Data Transfers: Moving or downloading large amounts of data can cost more.

References

Amazon Web Services (AWS). (n.d.). *What is cloud storage?* Retrieved June 16, 2025, from https://aws.amazon.com/what-is/cloud-storage/

Microsoft Azure. (n.d.). *What is Azure SQL Database?* Retrieved June 16, 2025, from https://azure.microsoft.com/en-us/products/azure-sql/

Google Cloud. (n.d.). *Cloud Spanner documentation*. Retrieved June 16, 2025, from https://cloud.google.com/spanner/docs

Oracle. (n.d.). *Advantages and disadvantages of cloud databases*. Retrieved June 16, 2025, from https://www.oracle.com/database/cloud/

IBM Cloud Education. (2022). *Cloud databases explained*. Retrieved from https://www.ibm.com/cloud/learn/cloud-databases

Red Hat. (n.d.). *What is cloud storage?* Retrieved June 16, 2025, from https://www.redhat.com/en/topics/cloud-computing/what-is-cloud-storage

Amazon Web Services (AWS). (n.d.). *Amazon RDS – Features*. Retrieved June 16, 2025, from https://aws.amazon.com/rds/features/

Cisco. (2021). *Understanding cloud computing risks and challenges*. Retrieved from https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-cybersecurity-report/acr-2021.html

Deloitte. (2023). *Cloud migration and vendor lock-in*. Retrieved from https://www2.deloitte.com/global/en/pages/risk/articles/cloud-computing-risks.html

Gartner. (2022). *Cloud cost management and optimization*. Retrieved from https://www.gartner.com/en/articles/optimizing-cloud-costs