

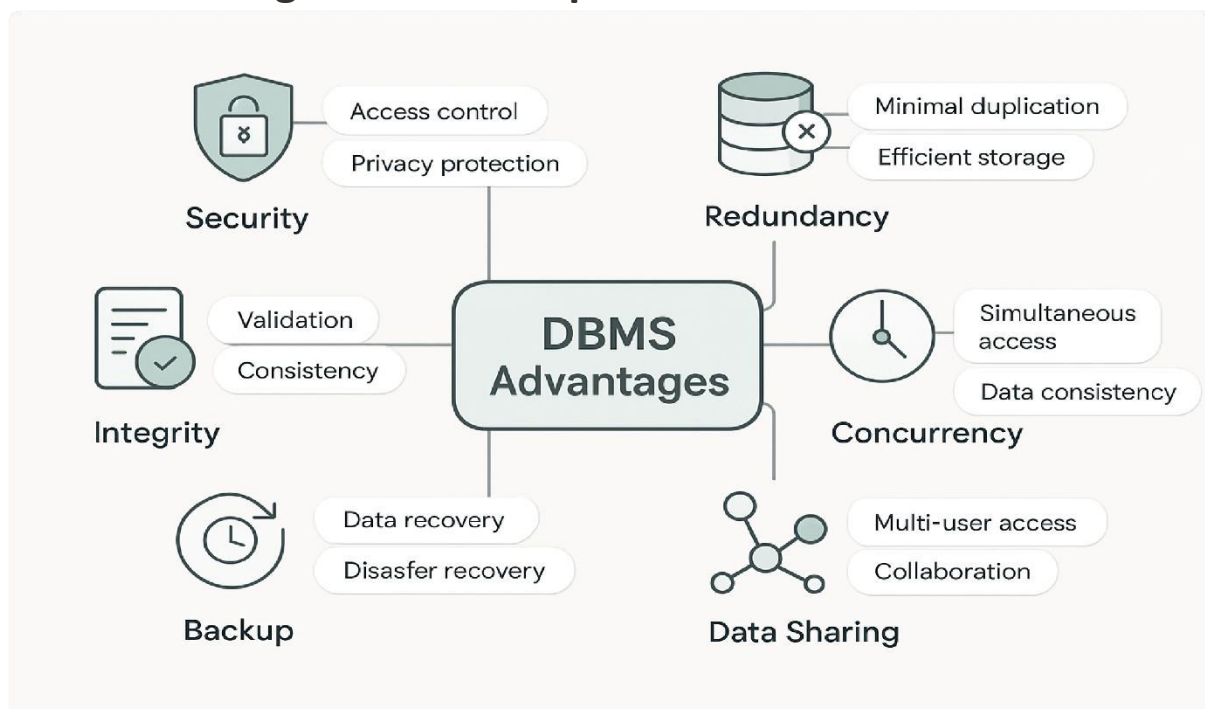


Database Course Documentation

1. Flat File Systems vs. Relational Databases

 FLAT FILE SYSTEMS		 RELATIONAL DATABASES	
Structure	Data stored in plain text files	Data organized into tables with rows and columns	
Data Redundancy	Higher – duplicate data across files	Lower – minimal duplication of data	
Relationships	No support for data relationships	Supports relationships between tables	
Example Usage	Storing configuration or log files	Business applications, customer data	
Drawbacks	Limited functionality, harder to manage	Greater complexity, requires more resources	

2. DBMS Advantages – Mind Map



3. Roles in a Database System

System Analyst

The System Analyst gathers requirements from users, analyzes existing systems, and designs new systems. Their role is in defining what the database system needs to achieve from a business perspective.

Database Designer

The Database Designer is responsible for the logical and physical design of the database. Includes creating the schema, defining tables, relationships, constraints, and data types. They ensure the database structure is efficient, consistent, and meets the requirements gathered by the System Analyst.

Database Developer

The Database Developer implements the database design. They write SQL queries, stored procedures, functions, and triggers. They are also involved in data migration, performance tuning, and ensuring the database integrates seamlessly with applications.

Database Administrator (DBA)

The Database Administrator (DBA) is responsible for the overall health, performance, and security of the database system. Their tasks include installation, configuration, monitoring, backup and recovery, user management, and troubleshooting. They ensure the database is always available and performing optimally.

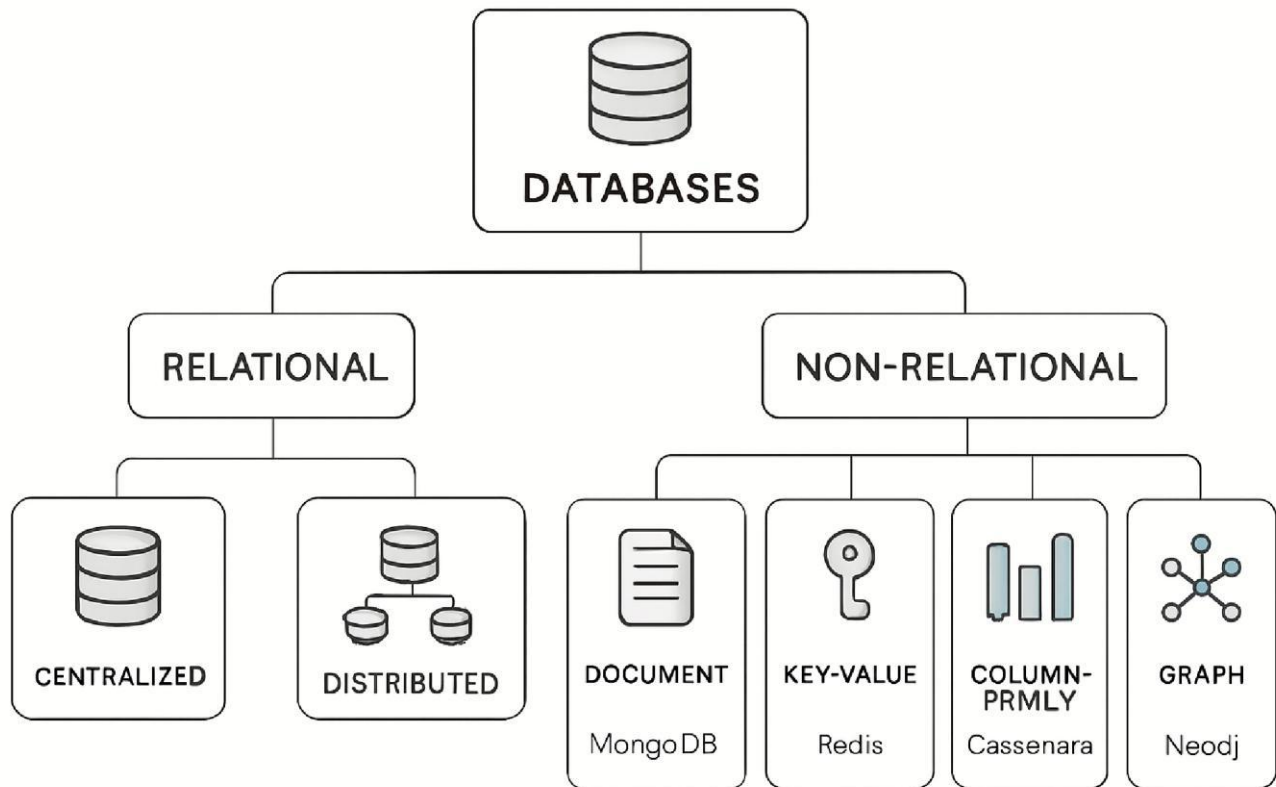
Application Developer

The Application Developer builds the software applications that interact with the database. They write code to retrieve, store, and manipulate data, ensuring a smooth user experience.

BI (Business Intelligence) Developer

The BI Developer focuses on extracting, transforming, and loading (ETL) data from various sources into data warehouses or data marts. They design and build dashboards, reports, and analytical tools to help businesses make decisions based on their data. Turning raw data into actionable insights.

4. Types of Databases



Relational vs. Non-Relational Databases

Relational Databases: These databases store data in a structured format using tables, rows, and columns. They are based on the relational model, where data is organized into related tables and relationships are defined using primary and foreign keys.

SQL (Structured Query Language) is typically used to manage and query data in relational databases.

Non-Relational Databases (NoSQL): These databases provide a mechanism for storing and retrieving data that is modeled in means other than the tabular relations used in relational databases. They are often used for large-scale data storage, real-time web applications, and applications.

Examples include:

- **MongoDB:** A document-oriented database that stores data in flexible, JSON-like documents.
- **Cassandra:** A wide-column store NoSQL database designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. Suitable for applications requiring high write throughput.

Centralized vs. Distributed vs. Cloud Databases

Centralized Databases: In a centralized database system, all data is stored and maintained in a single location, typically on a single server. All users access this central server to retrieve or update data. This setup simplifies management and ensures data consistency but can be a single point of failure and may face scalability limitations.

Distributed Databases: A distributed database system stores data across multiple interconnected computers or nodes, which may be located in different physical locations. The system appears as a single logical database to the user. This approach offers improved scalability, availability, and fault tolerance compared to centralized systems, but introduces complexities in data synchronization and consistency.

Cloud Databases: Cloud databases are databases that run on a cloud computing platform. They offer the benefits of cloud computing, such as scalability, high availability, and managed services, without the need for users to set up and maintain their own hardware and software. Cloud providers offer various database services, including relational (e.g., Azure SQL, Amazon RDS, Google Cloud Spanner) and NoSQL options. They are highly flexible and can be scaled up or down based on demand.

Use Case Examples for Each Type

Relational Databases: E-commerce platforms (managing product catalogs, customer orders), banking systems (transaction processing, account management), inventory management systems.

Non-Relational Databases:

- **MongoDB:** Content management systems, mobile applications, real-time analytics, user profiles.

- **Cassandra:** IoT data ingestion, fraud detection, messaging systems, social media analytics.
- **Centralized Databases:** Small business applications, single-server web applications, personal databases.
- **Distributed Databases:** Large-scale web services, global e-commerce platforms, big data analytics.
- **Cloud Databases:** Any application requiring high scalability, global reach, disaster recovery, or reduced operational overhead. For example: SaaS applications, gaming, and streaming services.

5. Cloud Storage and Databases

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

Cloud storage refers to storing digital data in logical pools, where the physical storage spans multiple servers, and the physical environment is typically owned and managed by a hosting provider. This provider is responsible for keeping the data available and accessible, and the physical environment protected and running. Cloud storage supports database functionality by providing a highly scalable, durable, and available infrastructure for storing database files, backups, and related data.

Instead of managing local storage hardware, databases can leverage cloud storage for their underlying data persistence, benefiting from the cloud's inherent elasticity and reliability.

Advantages of using cloud-based databases

- **Scalability:** Cloud databases can easily scale up or down based on demand, allowing businesses to handle fluctuating workloads without over-provisioning resources.

- **Cost-Effectiveness:** They eliminate the need for significant upfront investments in hardware and infrastructure. Users pay only for the resources they consume, converting capital expenditures into operational expenditures.
- **High Availability and Durability:** Cloud providers offer built-in redundancy, automated backups, and disaster recovery mechanisms, ensuring high availability and durability of data even in the event of failures.
- **Managed Services:** Cloud providers handle routine administrative tasks such as patching, backups, and maintenance, freeing up database administrators to focus on more strategic tasks.
- **Global Reach:** Cloud databases can be deployed in various geographical regions, enabling applications to serve users worldwide with low latency.
- **Security:** Cloud providers invest heavily in security measures, offering robust security features, compliance certifications, and threat detection capabilities.

Disadvantages or challenges with cloud-based databases

- **Security Concerns:** While cloud providers offer robust security, shared responsibility models mean users are still responsible for securing their data within the cloud environment. Data privacy and compliance can also be complex.
- **Latency:** Depending on the application's location relative to the cloud database, network latency can sometimes be a concern, impacting performance for highly sensitive applications.
- **Vendor Lock-in:** Migrating from one cloud database provider to another can be challenging due to proprietary technologies and APIs, potentially leading to vendor lock-in.

- **Cost Management:** While cost-effective for many, managing cloud costs can become complex, especially with dynamic scaling and various pricing models. Unexpected costs can arise if resources are not properly monitored and optimized.
- **Performance Variability:** Performance can sometimes be inconsistent due to shared resources in a multi-tenant cloud environment, although providers offer dedicated options to mitigate this.
- **Data Transfer Costs:** Ingress and egress data transfer costs can accumulate, especially for applications with high data movement between the cloud database and other services or onpremises systems.