Database Search and Reporting

Name: Sundus Alrashdi

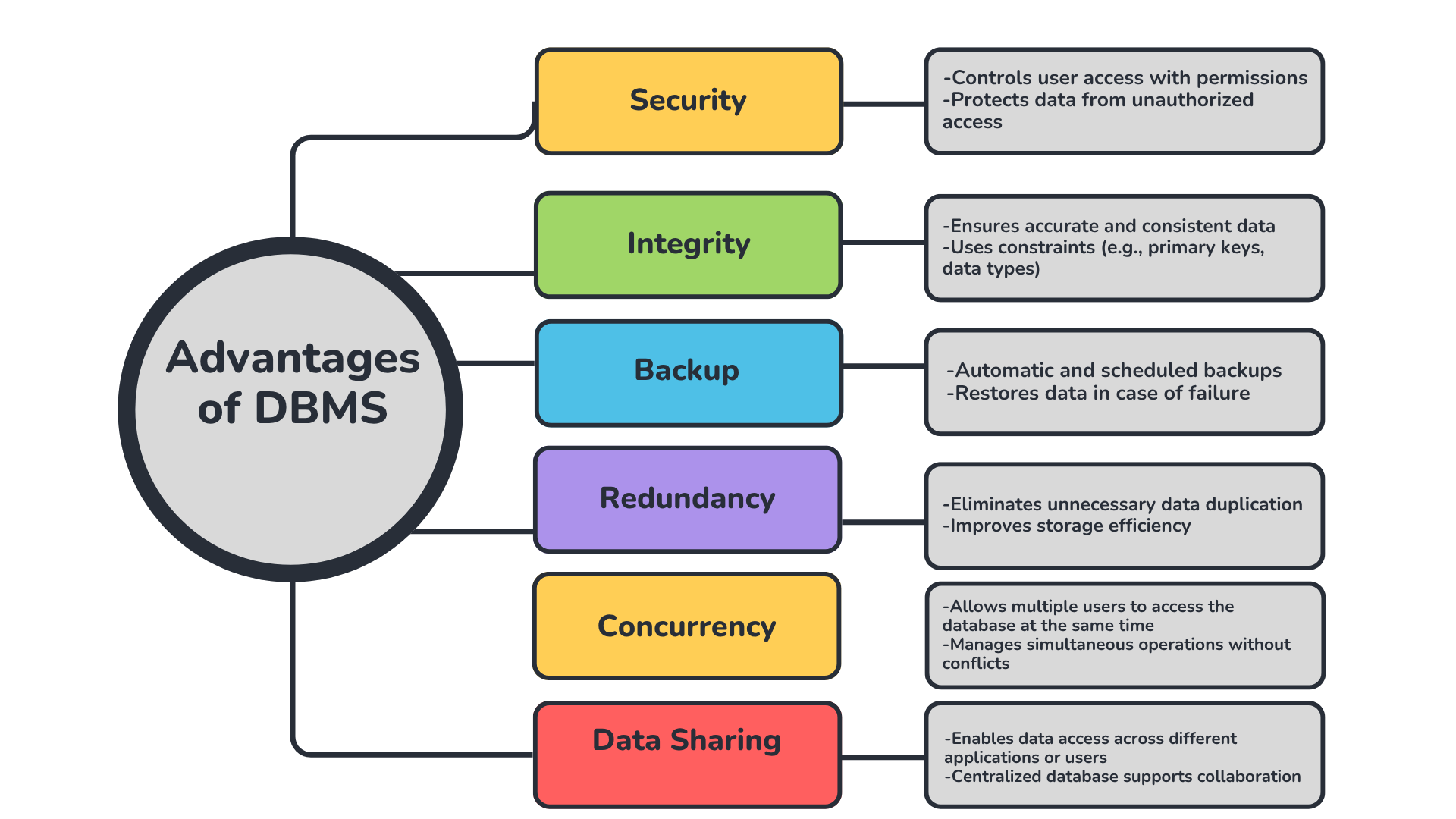
**Introduction**

This report explores key database concepts, comparing flat file systems and relational databases. It also covers the advantages of DBMS, important roles in database projects, types of databases, and the role of cloud storage. The goal is to provide a clear understanding of these topics through research and visual aids.

1. Flat File Systems vs. Relational Databases:

|  |  |  |
| --- | --- | --- |
|  | Flat File System | Relational Database (RDBMS) |
| Structure | Data is stored in a single file (e.g., .txt, .csv) with a simple format | Data is stored in tables with rows and columns; follows a defined schema |
| Data Redundancy | (High redundancy) the data is often repeated across multiple files | (Low Redundancy) it is reduced using keys and references |
| Relationships | No support for defining or managing relationships between data sets | Supports complex relationships using foreign keys and joins |
| Example Usage | Excel sheets, log files, Simple configurations | School databases, banking systems, inventory systems |
| Drawbacks | Hard to search, update, or scale; lacks integrity checks | Needs setup and maintenance; more resource-intensive |

1. **Mind Map – Advantages of DBMS:**



1. **Roles in a Database System:**

| **Role** | **Description** |
| --- | --- |
| System Analyst | - Works closely with stakeholders to gather and analyze system requirements - Defines the database needs based on business goals - Acts as a bridge between users and technical team |
| Database Designer | - Creates the **conceptual**, **logical**, and **physical** design of the database - Chooses appropriate data types, relationships, and structures - Ensures the design supports scalability and performance |
| Database Developer | - Implements the database design by writing **SQL queries**, **stored procedures**, **triggers**, and **views** - Builds database logic for data manipulation and access |
| Database Administrator | - Manages the database system after it’s developed - Handles **backup**, **recovery**, **performance tuning**, **security**, and **user permissions** - Ensures system uptime and data safety |
| Application Developer | - Develops applications or interfaces that interact with the database (e.g., web apps, mobile apps) - Uses APIs or queries to read/write data - Works closely with database developers to ensure efficient integration |
| BI Developer | - Extracts data from databases and builds **dashboards**, **reports**, and **data visualizations** - Uses tools like **Power BI**, **Tableau**, or **SQL-based analytics** - Helps organizations make data-driven decisions |

4- **Types of Databases:**

- Relational vs. Non-Relational Databases:

**Relational Databases** store data in structured tables made up of rows and columns. They use a predefined schema and enforce relationships through primary and foreign keys. These databases are best for applications that require consistency, complex queries, and transactional integrity. Examples include **MySQL**, **PostgreSQL**, and **Oracle**.

**Non-Relational Databases** (NoSQL) store data in flexible formats such as documents, key-value pairs, graphs, or wide-column stores. They are optimized for large-scale, unstructured data and provide high scalability and performance. Common examples include:

MongoDB – A document-based database where data is stored in JSON-like format.

Cassandra – A wide-column store designed for high availability and handling large volumes of data.

Centralized vs. Distributed vs. Cloud Databases:

**Centralized Databases** keep all data on a single server or location. This setup is simpler and easier to maintain, but it may become a bottleneck or single point of failure. Suitable for small businesses or local systems.

**Distributed Databases** store data across multiple servers or locations, offering better fault tolerance, performance, and availability. They are ideal for large-scale or global applications that require data access from multiple regions.

**Cloud Databases** are hosted and managed by cloud providers such as **Amazon Web Services (RDS)**, **Microsoft Azure (SQL Database)**, and **Google Cloud Spanner**. They offer automatic scaling, backup, high availability, and reduce the burden of manual database administration.

**Use Case Examples**

| **Database Type** | **Example Use Case** |
| --- | --- |
| Relational | Banking systems, university records, payroll systems |
| Non-Relational | E-commerce product catalogs, user activity logs, real-time analytics |
| Centralized | Small retail store inventory, local hospital database |
| Distributed | Global social media platforms, video streaming services like Netflix |
| Cloud | Online education platforms, SaaS applications, collaborative tools like Slack |

1. **Cloud Storage and Databases:**

**Cloud storage** refers to storing data on remote servers hosted by cloud service providers (such as AWS, Microsoft Azure, or Google Cloud), which users can access over the internet. Instead of keeping data on local machines or physical servers, organizations upload and retrieve their data from the cloud.

Cloud storage supports **database functionality** by providing scalable, reliable, and always-available infrastructure for hosting databases. Cloud platforms offer **Database as a Service (DBaaS)**, allowing users to deploy and manage databases without worrying about hardware, physical maintenance, or complex setup. These platforms can run both **relational** (like Azure SQL or Amazon RDS) and **non-relational** (like DynamoDB or Firebase) databases.

| **Advantages** | **Disadvantages** |
| --- | --- |
| 🔹 **Scalability** – Easily adjust resources based on demand | **Internet Dependency** – Requires constant internet access |
| 🔹 **High Availability** – Automatic backups, replication, and failover | **Security Concerns** – Risk of data breaches or compliance issues |
| 🔹 **Cost Efficiency** – Pay only for what you use | **Vendor Lock-In** – Difficult to migrate or switch providers |
| 🔹 **Managed Services** – Providers handle updates, maintenance, and security | **Limited Customization** – Less control over configurations and optimizations |
| 🔹 **Global Accessibility** – Access from anywhere with internet | **Performance Variability** – May be affected by shared cloud infrastructure |