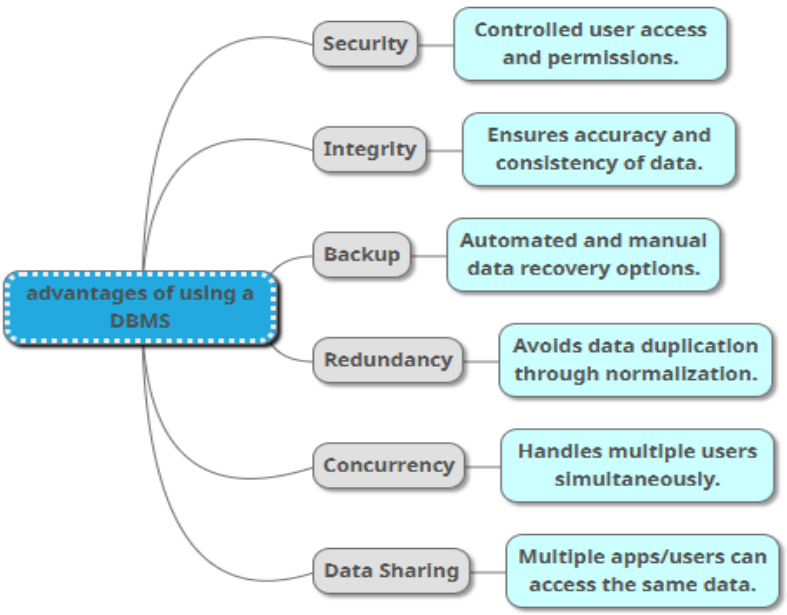
Task 1 (Report) Database Search and Reporting Task

✅ 1. Comparison: Flat File Systems vs Relational Databases

Feature	Flat File Systems	Relational Databases
Structure	Single, unstructured file (e.g., CSV)	Organized into tables with rows and columns
Data Redundancy	High – duplicate data across files	Low – uses relationships to avoid repetition
Relationships	No relationships	Supports foreign keys and joins
Example Usage	Storing logs, simple spreadsheets	Online banking, inventory systems
Drawbacks	Hard to update, prone to errors, not scalable	Complex setup, requires DBMS knowledge

DBMS Advantages Mind



👨‍💼 3. Roles in a Database System

Role	Description
System Analyst	Gathers user requirements and defines database specifications.
Database Designer	Designs the database structure and schema.
Database Developer	Implements the design and builds the database.
DBA (Admin)	Manages, secures, and optimizes the database.
Application Developer	Builds apps that connect to and use the database.
BI Developer	Creates reports and dashboards from data for decision-making.

Additional Research Topics

4. Types of Databases

- **Relational:** Structured, uses SQL (e.g., MySQL, PostgreSQL)
- **Non-Relational:** Flexible schema, used for big data (e.g., MongoDB, Cassandra)
- **Centralized:** Stored in a single location
- **Distributed:** Spread across multiple systems
- **Cloud:** Hosted on cloud platforms (e.g., AWS, Azure)

Use Cases:

- MongoDB – Real-time analytics
- MySQL – Web applications
- Google Cloud Spanner – Global-scale apps

5. Cloud Storage and Databases

- **Cloud Storage:** Storing data over the internet instead of on local devices.
- **Relation to Databases:** Cloud platforms host databases, providing easy scaling and backups.

Advantages:

- Accessible, scalable, reduced hardware costs

Disadvantages:

- Internet dependency, data privacy concerns

Examples: Azure SQL, Amazon RDS, Google Cloud Spanner

6. Database Engines and Languages

- **Database Engine:** Core service that stores, processes, and secures data.
- **Examples:** SQL Server, MySQL, Oracle, PostgreSQL
- **Languages Used:**
    - SQL Server → T-SQL
    - Oracle → PL/SQL
    - MySQL/PostgreSQL → ANSI SQL

Cross-compatibility: Some SQL syntax works across engines, but advanced features may differ.

7. Database Migration Between Engines

Is it Possible? Yes, using migration tools or scripts.

Challenges:

- Data type mismatch
- Stored procedures and triggers may not translate well
- Indexing and performance tuning differences

Considerations: Compatibility, testing, downtime, security

8. Logical vs Physical Schema

- **Logical Schema:** Abstract design (e.g., entities, relationships)
- **Physical Schema:** Actual implementation in a database system (e.g., table structures, data types)

Why It Matters: Understanding both ensures a system is both well-planned and efficiently implemented.

Example:

- **Logical Entity:** Student(Name, ID, Courses)
- **Physical:** Table: `students` with columns `student_id INT`, `name VARCHAR(50)`, `course_id INT`