**Task 1 ( Report )**

**Database Search and Reporting Task**

**Objective:**

To develop research and analytical reporting skills by exploring key database concepts

**Make a new git hub repo ( Database Course documentation ) and add the following requirements to a one report and upload it**

1. **Comparison Assignment**

Create a comparison between **Flat File Systems** and **Relational Databases** covering:

* + Structure
  + Data Redundancy
  + Relationships
  + Example usage
  + Drawbacks

**The Answer:**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Flat File System** | **Relational Database** |
| **Structure** | Simple text or CSV files | Tables with rows and columns (schema-based) |
| **Redundancy** | High – duplicate data common | Low – normalization reduces redundancy |
| **Relationships** | Not inherently supported | Supports relations via keys (PK, FK) |
| **Usage** | Excel sheets, config files | Enterprise apps, banking systems |
| **Drawbacks** | Hard to query, no constraints | More complex setup and management |

1. **DBMS Advantages Mind Map**

Draw a mind map (or use online tools like [MindMup)](https://www.mindmup.com/) illustrating the **advantages of using a DBMS**. Include short descriptions or icons for:

* + Security
  + Integrity
  + Backup
  + Redundancy
  + Concurrency
  + Data sharing

**The Answer:**

**Security**  
Keeps data safe by controlling who can access or change it.

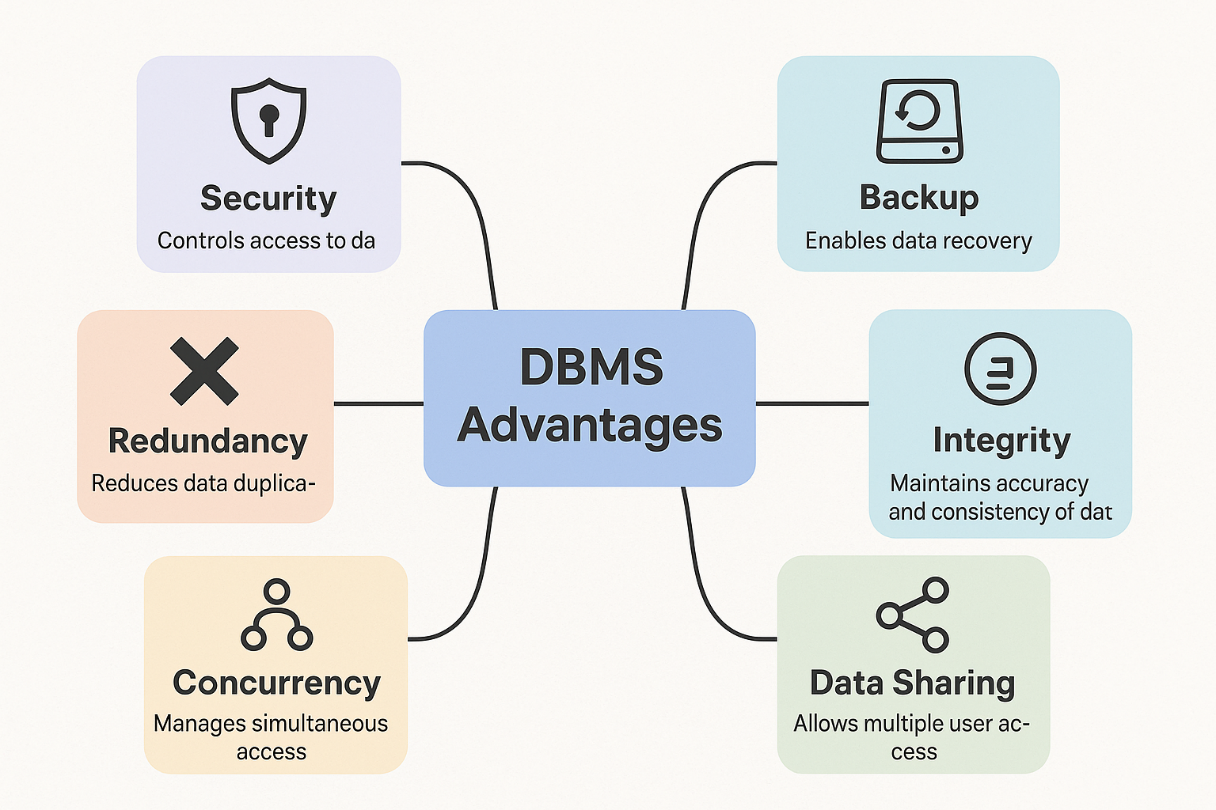
**Integrity**  
Makes sure the data is correct and stays consistent.

**Backup**  
Saves copies of data so it can be restored if something goes wrong.

**Redundancy**  
Avoids storing the same data in multiple places.

**Concurrency**  
Lets many users work with the data at the same time without problems.

**Data Sharing**  
Allows different users and programs to use the same data easily



1. **Roles in a Database System**

Explaining each of the following roles:

* + System Analyst
  + Database Designer
  + Database Developer
  + DBA (Admin)
  + Application Developer
  + BI Developer

**The Answer:**

|  |  |
| --- | --- |
| **Role** | **Description** |
| **System Analyst** | Gathers user requirements and plans system architecture |
| **Database Designer** | Designs schema, ensures normalization and integrity |
| **Database Developer** | Implements schema, writes stored procedures and queries |
| **DBA (Admin)** | Manages performance, backups, security |
| **Application Developer** | Connects frontend/backend to DB using queries |
| **BI Developer** | Builds reports, dashboards from database data |

-----------------------------------------------------------------------------------------------------------------------------------------------------------

**Additional Research Topics to Include in the Report:**

These topics are often overlooked but important for understanding the ecosystem of databases in modern applications.

**Types of Databases**

Search and briefly describe:

**Relational** vs **Non-Relational** (e.g., MongoDB, Cassandra)

* + Centralized vs Distributed vs Cloud Databases
  + Use case examples

**The Answer:**

**Types of Databases**

**• Relational vs non-relational**

* **Relational Databases** (e.g., MySQL, PostgreSQL): Store data in tables with rows and columns. Use SQL for queries. Ideal for structured data.
* **Non-Relational Databases** (e.g., MongoDB, Cassandra): Store data in flexible formats like documents, key-value pairs, or graphs. Great for unstructured or semi-structured data.

**Example Use Cases:**

* Relational: Banking systems, HR software
* Non-Relational: Social media apps, IoT data storage

**• Centralized vs Distributed vs Cloud Databases**

* **Centralized:** All data stored on one server. Easier control but risky if server fails.
* **Distributed:** Data spread across multiple locations or servers. More reliable and scalable.
* **Cloud Databases:** Hosted on cloud platforms (e.g., AWS, Azure). Accessible from anywhere, scalable, managed by providers.

**Cloud Storage and Databases**

* + What is **Cloud Storage** and how does it relate to databases?
  + Advantages and Disadvantages of using cloud-based databases (e.g., Azure SQL, Amazon RDS, Google Cloud Spanner)

**The Answer:**

**Cloud Storage and Databases**

**• What is Cloud Storage?**

A service that stores data online, accessible over the internet. It supports databases by providing scalable, remote data storage.

**• Advantages of Cloud Databases**

* Accessible globally
* Scalable and flexible
* Managed backups and updates
* Lower upfront cost

**• Disadvantages**

* Security and compliance risks
* Internet dependency
* Potential vendor lock-in

**Examples:**

* **Azure SQL Database** (Microsoft)
* **Amazon RDS** (AWS)
* **Google Cloud Spanner**

**Database Engines and Languages** Search about:

* + What is a **Database Engine**?
  + Examples: **SQL Server**, **MySQL**, **Oracle**, **PostgreSQL**
  + What languages do they use? (e.g., T-SQL, PL/SQL, ANSI SQL) • Is there a relationship between the **engine** and the **language**?
  + Can one language work across different engines?

**The Answer:**

**What is a Database Engine?**

A software component that stores, retrieves, and manages database data. It executes SQL commands and handles performance tasks like indexing, transactions, and locking.

|  |  |
| --- | --- |
| **Engine** | **Language Used** |
| SQL Server | T-SQL |
| Oracle | PL/SQL |
| MySQL | ANSI SQL |
| PostgreSQL | PL/pgSQL, SQL |

Relationship Between Engine and Language

* The engine determines which SQL dialect is supported.
* Some SQL code is portable, but engine-specific features (e.g., procedures or triggers) may not be.

• Can One Language Work Across Engines?

* Yes, basic SQL works across many engines, but advanced features and syntax differ.

**Can We Transfer a Database Between Engines?**

Search and answer:

* + Is it possible to migrate a database from **SQL Server to MySQL**, or **Oracle to PostgreSQL**?
  + What are the challenges of engine-to-engine migration?
  + What should we consider before transferring (data types, triggers, stored procedures, etc.)?

**The Answer:**

**Is Migration Possible?**

Yes, you can migrate databases between engines like SQL Server ➝ MySQL or Oracle ➝ PostgreSQL.

**• Challenges**

* Different data types
* Incompatible SQL dialects
* Rewriting stored procedures, triggers, and constraints
* Application dependencies

**• What to Consider Before Migrating**

* Data structure compatibility
* Referential integrity and constraints
* Function and procedure logic
* Tools like AWS Schema Conversion Tool or DBConvert can help

**Logical vs. Physical Schema**

* + What is the **Logical Schema** in database design?
  + What is the **Physical Schema**?
  + What’s the difference between them?
  + Why is it important to understand both?
  + Example: Show how one entity (e.g., Student) would appear in both logical and physical schemas.

**The Answer:**

**• What is a Logical Schema?**

An abstract design of the database—defines tables, fields, relationships without considering physical implementation.

**• What is a Physical Schema?**

Describes how data is stored on disk—indexes, storage formats, partitions, etc.

The different:

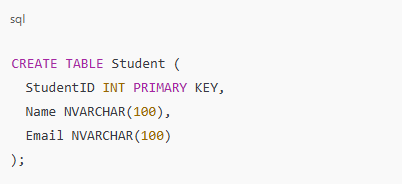
|  |  |
| --- | --- |
| **Logical Schema** | **Physical Schema** |
| Conceptual model | Implementation model |
| Focuses on structure | Focuses on performance |
| Tech-agnostic | Engine-specific |

**Why It’s Important**

Understanding both ensures efficient design and optimized performance in real-world systems.

**Example: “Student” Entity**

* **Logical Schema:**
  + Table: Student
  + Fields: StudentID, Name, Email
* **Physical Schema (SQL Server):**



--------------------------------------------------------------------------------------------------------------------------------------

**Evaluation Criteria:**

|  |  |
| --- | --- |
| **Criteria** | **Marks** |
| Content Accuracy and Depth | 10 |
| Research Effort | 10 |
| Organization & Clarity | 5 |
| Visuals and Mind Map | 5 |
| GitHub Submission | 5 |
| **Total** | **35 marks** |