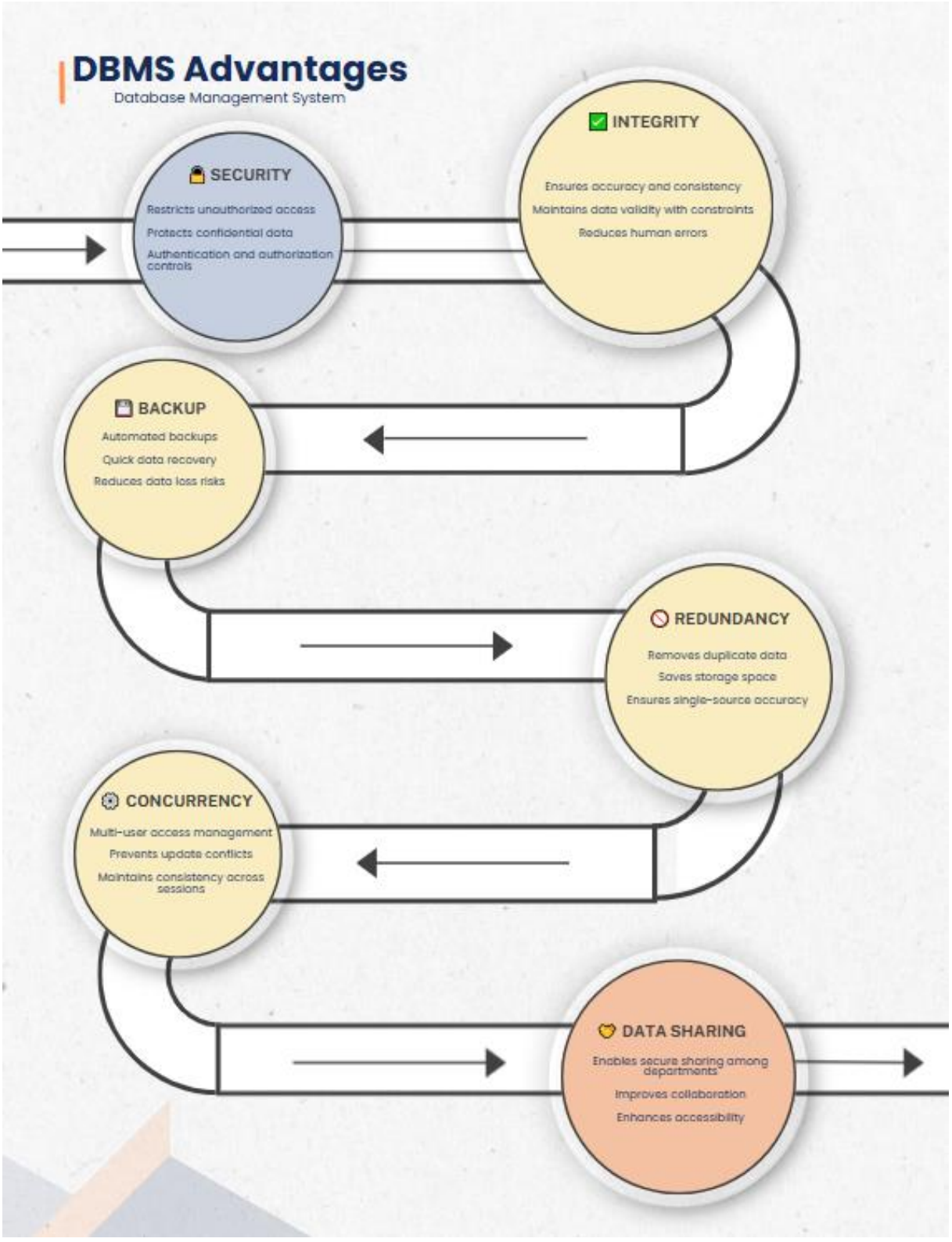
Task 1:

A **Flat File System** stores data in a simple, single file such as a text file or spreadsheet. Each line in the file usually represents a record, and the fields are separated by commas or tabs.

A Relational Database organizes data into tables with rows and columns. Each table is related to others using keys, such as primary and foreign keys.

Flat File Systems vs. Relational Databases:

Feature	Flat File System	Relational Database
Structure	Stores data in a single file like CSV or text; no relations between data.	Stores data in multiple related tables using rows and columns.
Data Redundancy	High redundancy – same data repeated in multiple files.	Low redundancy – data stored once and referenced using keys.
Relationships	No relationships between data records.	Supports relationships (one-to-one, one-to-many, many-to-many).
Example Usage	Simple logs, Excel sheets, configuration files.	Banking systems, HR systems, e-commerce platforms.
Drawbacks	Hard to update or maintain; prone to inconsistencies.	Requires setup and maintenance but ensures consistency and scalability.


DBMS Advantages
Database Management System
SECURITY
Restricts unauthorized access
Protects confidential data
Authentication and authorization controls
INTEGRITY
Ensures accuracy and consistency
Maintains data validity with constraints
Reduces human errors
BACKUP
Automated backups
Quick data recovery
Reduces data loss risks
REDUNDANCY
Removes duplicate data
Saves storage space
Ensures single-source accuracy
CONCURRENCY
Multi-user access management
Prevents update conflicts
Maintains consistency across sessions
DATA SHARING
Enables secure sharing among departments
Improves collaboration
Enhances accessibility

Task 2:

Task 3 :
Roles in a Database System:

1️⃣ System Analyst

Main Role: Acts as a bridge between business needs and technical implementation.

Responsibilities:

Gathers and analyzes user requirements.

Defines system functions and performance needs.

Prepares detailed specifications for designers and developers.

Ensures the database system supports business goals effectively.

2️⃣ Database Designer

Main Role: Structures how data will be organized and related.

Responsibilities:

Designs the logical and physical database models.

Defines tables, relationships, keys, and constraints.

Ensures efficient data flow and normalization.

Works closely with the System Analyst to align design with requirements.

3️⃣ Database Developer

Main Role: Builds and codes the actual database according to design specifications.

Responsibilities:

Implements tables, queries, and stored procedures.

Writes SQL scripts for CRUD (Create, Read, Update, Delete) operations.

Optimizes performance through indexing and query tuning.

Tests data integrity and ensures smooth integration with applications.

4️⃣ Database Administrator (DBA)

Main Role: Maintains and secures the database system.

Responsibilities:

Manages installation, configuration, and updates.

Performs backup, recovery, and user access control.

Monitors performance and prevents data breaches.

Ensures the system runs efficiently and reliably.

5️⃣ Application Developer

Main Role: Builds software applications that interact with the database.

Responsibilities:

Uses programming languages (e.g., Java, Python, C#) to connect applications with databases.

Designs user interfaces for data input and retrieval.

Implements business logic and ensures smooth user experience.

Works with Database Developers and DBAs for data integration.

6️⃣ BI (Business Intelligence) Developer

Main Role: Transforms raw data into actionable insights.

Responsibilities:

Designs dashboards and reports using tools like Power BI or Tableau.

Develops ETL (Extract, Transform, Load) processes for data warehousing.

Performs data analysis and trend visualization.

Supports strategic decision-making through analytics.

Task 4:

Types of Databases

1. Relational vs. Non-Relational Databases

- **Relational Databases (RDBMS):**

Store data in tables (rows and columns).

Use SQL (Structured Query Language) for data management.

Data has fixed schemas and relationships using primary/foreign keys.

- **Examples:** MySQL, PostgreSQL, Oracle, Microsoft SQL Server.
- **Use Case:** Banking systems, HR databases, inventory management — where data relationships are clearly defined and consistent.

Non-Relational Databases (NoSQL)

- Store data in **different formats** such as key-value pairs, documents, graphs, or wide columns.
- Are **schema-less**, flexible, and handle **unstructured or semi-structured** data.
- **Examples:**
  - **MongoDB** → document-oriented, stores data in JSON-like format.
  - **Cassandra** → wide-column store, designed for scalability and high availability.
- **Use Case:** Social media platforms, IoT data, real-time analytics — where large and flexible data sets are needed.

2. Centralized vs. Distributed vs. Cloud Databases

Centralized Database

- All data stored and managed in **one central location/server**.
- Easy to maintain but can become a **single point of failure**.
- **Use Case:** Small organizations or schools where all users access one main database.

Distributed Database

- Data is **spread across multiple locations or servers** but connected through a network.
- Offers **faster access, fault tolerance, and reliability**.
- **Use Case:** Large enterprises or global applications (e.g., airline reservation systems) where data must be synchronized across regions.

Cloud Database

- Hosted on **cloud computing platforms** like AWS, Azure, or Google Cloud.
- Provides **on-demand scalability, automatic backups, and remote accessibility**.
- **Use Case:** E-commerce or mobile applications needing global access and flexible storage without managing physical hardware.

Summary Table

Type	Description	Example	Use Case
Relational	Structured tables with SQL	MySQL, Oracle	Banking, Inventory
Non-Relational	Schema-less, NoSQL	MongoDB, Cassandra	Social media, IoT
Centralized	One server location	School DB	Small organizations
Distributed	Multiple connected databases	Google Spanner	Global enterprise systems
Cloud	Internet-hosted	AWS RDS, Firebase	E-commerce, mobile apps

Cloud Storage and Databases

1. What is Cloud Storage and How It Supports Databases

Cloud Storage refers to saving data on **remote servers** that are managed and maintained by **cloud service providers** such as **Amazon Web Services (AWS)**, **Microsoft Azure**, or **Google Cloud**.
Instead of keeping data on local computers or physical servers, it is stored online and can be accessed anytime through the internet.

Cloud storage **supports database functionality** by:

- **Hosting database servers** remotely, allowing organizations to run SQL or NoSQL databases without owning hardware.
- **Providing scalability** — databases can grow or shrink based on demand.
- **Ensuring data redundancy and backup**, so information remains safe even if one server fails.
- **Enabling global accessibility**, letting users or applications connect from any location