File-based vs. Database Systems

File-Based Systems

- Stores data in individual files within the file system of an operating system. Each application accesses its data by reading and writing directly to these files.
- **Characteristics**:
    - **Decentralized Data Storage**: Each application manages its own files, often leading to isolated data storage.
        - Example: A payroll application may store employee data in a separate file from a performance evaluation application.
    - **Fixed File Formats**: Data is often stored in flat files or predefined formats like .txt, .csv, or .xml.
    - **Minimal Data Abstraction**: Files are accessed directly without an intermediate layer like a DBMS.
- **Advantages**:
    - **Simplicity**: Easy to implement for small-scale systems.
        - Example: A single text file storing customer orders in a small retail shop.
    - **Low Overhead**: No need for specialized software like a DBMS; the file system handles storage.
    - **Direct File Access**: Applications can directly read or modify data without querying through a database layer.
- **Disadvantages**:
    - **Data Redundancy and Inconsistency**: Multiple applications might store duplicate data, leading to inconsistencies.
        - Example: An employee's address being updated in one application but not in another.
    - **Poor Data Security**: No centralized control for enforcing access restrictions.
    - **Limited Data Sharing**: Sharing data between applications is difficult without manual intervention.
    - **Complex Data Retrieval**: Searching, filtering, or aggregating data is not as efficient as in database systems.
    - **Scalability Issues**: Becomes inefficient as the volume of data grows.

CCINFOM Fundamentals of Databases

Database Systems

- Stores data in a structured manner, typically managed by a Database Management System (DBMS), which acts as an intermediary between applications and the stored data.
- **Characteristics**:
    - **Centralized Data Management**: All data is stored in a centralized database, reducing redundancy and ensuring consistency.
    - **Schema-Based Design**: Data is organized according to a predefined schema that defines tables, relationships, and constraints.
    - **Query Language**: Use of SQL (Structured Query Language) to retrieve, update, and manage data.
- **Advantages**:
    - **Reduced Redundancy and Improved Consistency**: Centralized storage eliminates duplication of data.
        - Example: A customer's contact details updated once are reflected across all applications.
    - **Efficient Data Retrieval**: Optimized query processing enables quick searches and aggregations.
    - **Enhanced Security**: Access permissions and roles are enforced by the DBMS.
    - **Data Sharing**: Multiple applications can access the same database simultaneously without conflicts.
    - **Scalability**: Supports large-scale data with features like indexing, partitioning, and replication.
    - **Backup and Recovery**: Built-in mechanisms ensure data is recoverable in case of failures.
- **Disadvantages**:
    - **Complexity**: Setting up and maintaining a database system requires expertise.
    - **Higher Cost**: Licensing, hardware, and maintenance costs for DBMS can be significant.
    - **Overhead**: The DBMS layer introduces computational overhead compared to direct file access.