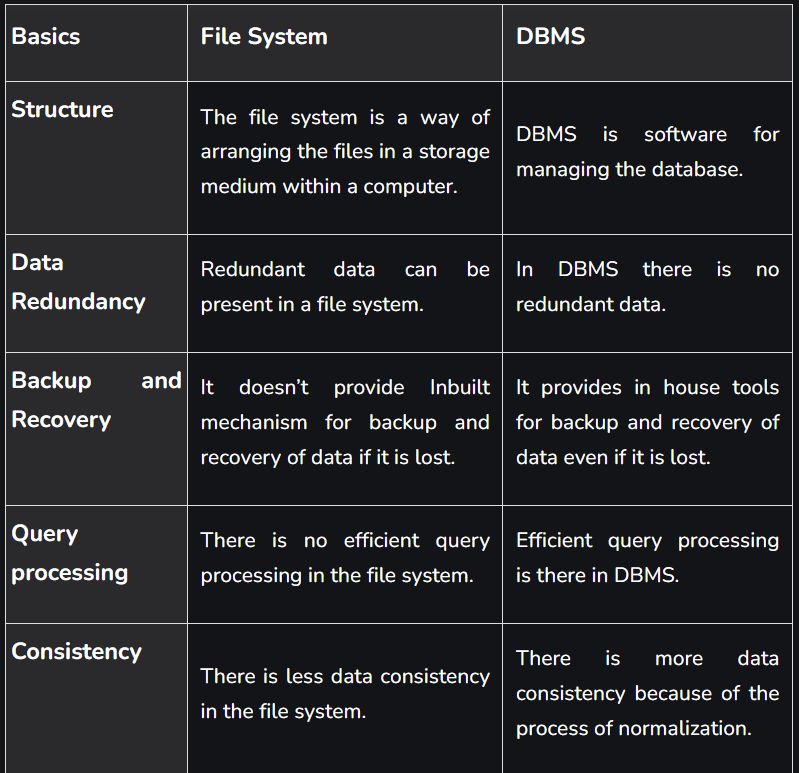
important topics:

Why do we need DBMS

* One of the primary needs for a DBMS is data organization and management.
* DBMSs allow data to be stored in a structured manner, which helps in easier retrieval and analysis.
* A well-designed database schema enables faster access to information, reducing the time required to find relevant data.
* A DBMS also provides features like indexing and searching, which make it easier to locate specific data within the database.
* This allows organizations to manage their data more efficiently and effectively.

File management system vs DBMS



Data abstraction

* **Data Abstraction** is a process of hiding unwanted or irrelevant details from the end user.
* The database systems consist of complicated data structures and relations.
* For users to access the data easily, these complications are kept hidden, and only the relevant part of the database is made accessible to the users through data abstraction.

 three levels of abstraction for DBMS

Physical or Internal Level

* It is the lowest level of abstraction for DBMS which defines how the data is actually stored, it defines data-structures to store data and access methods used by the database.
* Actually, it is decided by developers or database application programmers how to store the data in the database.

Logical or Conceptual Level

* Logical level is the intermediate level or next higher level.
* It describes what data is stored in the database and what relationship exists among those data.
* It tries to describe the entire or whole data because it describes what tables to be created and what are the links among those tables that are created.

View or External Level

* It is the highest level.
* In view level, there are different levels of views and every view only defines a part of the entire data.
* It also simplifies interaction with the user and it provides many views or multiple views of the same database.

ER model, relational model

* The Entity Relational Model is a model for identifying entities to be represented in the database and representation of how those entities are related.
* The ER data model specifies enterprise schema that represents the overall logical structure of a database graphically.
* The Entity Relationship Diagram explains the relationship among the entities present in the database.
* In short, the ER Diagram is the structural format of the database.

Why use ER models

* It is easily converted into relations (tables).
* ER diagrams require no technical knowledge and no hardware support.
* These diagrams are very easy to understand and easy to create even for a naive user.

A chart with symbols and symbols

Description automatically generated

Normalization basic definitions and usage

<https://www.guru99.com/database-normalization.html>

Transactions (Lock based protocol for concurrency control)

ACID properties

A diagram of a diagram

Description automatically generated

SQL vs NoSQL

Relational Databases

* A relational database or relational database management system(RDBMS) stores data in the form of a table.
* The table consists of rows and columns, and in relational database rows are referred to as records, and columns are referred to as fields.
* The most widely and popular way of interacting with a relational database is SQL(Structural Query Language), which allows access, filter, and modify data.
* Examples of Relational Databases: MySQL, Oracle.

Advantages of Relational Databases

* Acid properties
* Normalization
* High Security

Disadvantages of Relational Databases

* slow Performance

Non-Relational Database

* Non-Relational Databases are also called No-SQL databases, that doesn't require any table, fields, or records.
* NoSQL databases are completely different from SQL databases and work differently. It has to deal with semi-structured or unstructured data.
* Rather than containing tables, it consists of files within various folders.
* They can possess any kind of data, whether JSON, XML, etc.
* So, creating and managing data in NoSQL is easy and faster.

Types

* Documents Databases.
* Graph Databases.
* Wide Column Databases.
* Key-value Databases

Examples of Non-Relational Databases: MongoDB, Apache Cassandra.

Advantages of Non-Relational Databases

* Unstructured Data
* Large data and cheap
* Fast performance

Disadvantages

* Acid
* Backup
* No standrand rules

Deadlock in DBMS

In a database, a deadlock is an unwanted situation in which two or more transactions are waiting indefinitely for one another to give up locks.

Deadlock Avoidance:

* When a database is stuck in a deadlock, It is always better to avoid the deadlock rather than restarting or aborting the database.
* The deadlock avoidance method is suitable for smaller databases whereas the deadlock prevention method is suitable for larger databases. .
* Methods like Wait-For graph can be used in smaller databases to detect deadlocks

Deadlock Detection

* When a transaction waits indefinitely to obtain a lock,
* The database management system should detect whether the transaction is involved in a deadlock or not.
* Wait-for-graph is one of the methods for detecting the deadlock situation.
* This method is suitable for smaller databases.
* In this method, a graph is drawn based on the transaction and its lock on the resource
* If the graph created has a closed loop or a cycle, then there is a deadlock.

A diagram of a diagram

Description automatically generated

Deadlock prevention

* For a large database, the deadlock prevention method is suitable.
* A deadlock can be prevented if the resources are allocated in such a way that a deadlock never occurs.
* The DBMS analyzes the operations whether they can create a deadlock situation or not, If they do, that transaction is never allowed to be executed.

Prevention schemes

A black and white text on a black background

Description automatically generated

Starvation

* In a database management system (DBMS), starvation occurs when a transaction or process is not able to get the resources it needs to proceed and is continuously delayed or blocked.
* This can happen when other transactions or processes are given priority over the one that is experiencing starvation.
* In DBMSs, resources such as locks, memory, and CPU time are typically shared among multiple transactions or processes. If some transactions or processes are given priority over others, it is possible for one or more transactions or processes to experience starvation.

DBMSs typically use various techniques to prevent or mitigate starvation, such as:

* Resource allocation policies: DBMSs can use policies to allocate resources in a fair manner, ensuring that no transaction or process is consistently given priority over others.
* Priority-based scheduling: DBMSs can use scheduling algorithms that take into account the priority of transactions or processes, ensuring that high-priority transactions or processes are executed before low-priority ones.
* Timeout mechanisms: DBMSs can use timeout mechanisms to prevent transactions or processes from being blocked indefinitely, by releasing resources if a transaction or process waits for too long.
* Resource management: DBMSs can also use techniques such as resource quotas and limits to prevent any single transaction or process from monopolizing resources, thus reducing the likelihood of starvation.

**Questions on what database to be used**

**Scaling in databases**

Database scalability is the ability to expand or contract the capacity of system resources in order to support the changing usage of your application. This can refer both to increasing and decreasing usage of the application.

**Cache and in-memory database**

**Cache** - By definition means it is stored in memory. Any data stored in memory (RAM) for faster access is called cache. Examples: Ehcache, Memcache Typically you put an object in cache with String as Key and access the cache using the Key. It is very straight forward.

**In Memory Database**- It has all the features of a Cache plus come processing/querying capabilities.

* For the in-memory cache to work, it sets aside some part of the RAM that will work as the cache. Before an application reads data from storage, it first checks to see if the data is available in the cache.
* If it finds it, it reads it from the cache but if it doesn’t, it reads it from the source. Once the data is retrieved from the source, it is written on the cache so that it’s available next time it’s retrieved.
* This process of using in memory cache makes sure the recently accessed data is available for fast retrieval, thus eliminating the long process taken to read data from storage.
* In-memory database, on the other hand, makes data available all the time it’s needed.
* It is also called the main memory database because it’s found in the main memory. Because the data is not located in a hard disk, it eliminates the need for I/O tasks before accessing and processing data.
* The process reduces the time taken to retrieve or store data.

Also, practice standard interview questions from:

Most Standard topics : SQL queries, Normalization, transactions [<https://prepinsta.com/database-management-system-dbms/> ]Revision:

<https://www.geeksforgeeks.org/last-minute-notes-dbms/>

Interview questions:

<https://www.geeksforgeeks.org/commonly-asked-dbms-interview-questions/>

<https://www.geeksforgeeks.org/commonly-asked-dbms-interview-questions-set-2/>

DonneMartin book:

<https://github.com/donnemartin/system-design-primer#database>