**Question 4c**

Database administrator security measures are as follows;

1. **Use Strong authentication and access control**: This involves enforcing the use of strong, unique passwords for each user and regularly changing them. Additionally, employing multi-factor authentication can provide an extra layer of security, requiring users to verify their identity using a combination of something they know (i.e., password) and something they have (i.e., a mobile device).
2. **Regularly updating and patching software:** Keeping your database software up to date is crucial for avoiding vulnerabilities that cybercriminals can exploit. Developers frequently release software updates and patches to fix security flaws identified in previous versions. By regularly updating and patching your database software, you can ensure that you are benefiting from the latest security enhancements and protections.
3. **Encrypting sensitive data:** Encrypting sensitive data is another vital precautionary measure to safeguard your database. Encryption converts data into an unreadable format, ensuring that even if a breach occurs, the stolen data remains unintelligible to unauthorized individuals. It's recommended to employ strong encryption algorithms and implement encryption at rest as well as in transit to protect data both when stored and transmitted.
4. **Regularly Backup your database:** Encrypting sensitive data is another vital precautionary measure to safeguard your database. Encryption converts data into an unreadable format, ensuring that even if a breach occurs, the stolen data remains unintelligible to unauthorized individuals. It's recommended to employ strong encryption algorithms and implement encryption at rest as well as in transit to protect data both when stored and transmitted.

Question 5a

The principle of determination is related to the concept of functional dependence because the principle of determination anchors on one set attribute determining the value of another set of attributes.

Question 5b

Question 6a

A **Transaction Log**, often known as a database log, is a critical component of many databases. It is a history of actions that the DBMS executes to ensure data integrity and to allow for recovery from failures. It details all changes made to the database, along with providing the ability to roll back transactions or reproduce them for recovery purposes.

**Importance of this information to DBMS:**

The use of Transaction Logs aids in preserving database integrity by ensuring that transactions on a database follow the ACID (Atomicity, Consistency, Isolation, Durability) properties. They also provide a crucial recovery mechanism in case of unexpected database failures.

Question 6b

**Properties of Transaction**

There are four properties which can be denote with the acronym ACID which stand for Atomicity, Consistency, Isolation, Durability

* **Atomicity:**  This concept ensures that all operations within a transaction are either fully executed or completely undone. In other words, a transaction cannot be partially completed. Each transaction is treated as a single unit, similar to an atom. Atomicity is achieved through commit and rollback operations. This means that changes are only made to the database if all operations related to a transaction are successfully completed. If a transaction is interrupted, any changes made are rolled back using the rollback operation to return the database to its previous state.
* **Consistency:**  This property of a transaction ensures that the database remains consistent before and after the transaction is completed. Any transaction must guarantee that after it is executed, the database is either in its previous stable state or a new stable state.
* **Isolation:** In a database system where multiple transactions are executed simultaneously and in parallel, the isolation property ensures that each transaction is carried out as if it is the only transaction in the system. This means that no transaction will impact the existence of any other transaction.
* **Durability:** This property ensures that any changes made to the database after a transaction is fully executed are permanent. It signifies that successful execution of a transaction results in lasting changes. In the event of system failures or crashes, the consistent state achieved after the completion of a transaction remains unchanged. The recovery subsystem of the DBMS is responsible for enforcing this property.

Single-user databases and multi-user databases can ensure that these properties are in place when factored while designing the system.

Question 6ci

A schematic representation of hierarchical database structure



6cii.

Hierarchical databases excel in scenarios where data retrieval follows a specific, well-defined path. The structure allows for quick navigation from the root to the desired child node because the path is predefined and direct.

6ciii.

Each parent node connects directly to its child nodes, ensuring efficient data access. Hierarchical databases excel in applications with clear parent-child relationships. The Model provides fast data retrieval and easy comprehension due to its one-to-many relationships.

Question 7a

A database is said to be in a consistent state when all the data integrity constraints are satisfied. This process is achieved by ensuring data written to the database must be valid according to all defined rules, including constraints, cascades, triggers, or any combination.

Question 7B

A Database Management System (DBMS) is responsible for accurate data persistence and processing. However, it's true that a DBMS does not inherently guarantee that the semantic meaning of a transaction is equivalent to the real-world event it represents. Semantics represents the meaning, whereas DBMS operates on syntax.

The consequences of this limitation can be critical, depending upon the context. In healthcare, for instance, if a pharmacy system allows a transaction where a lethal dose of a drug is ordered due to the lack of suitable semantic rules, real-world implications could be life-threatening.

Question 7c

Concurrency Control ensures that simultaneous transactions on a DBMS do not interfere with each other. The main aim is to prevent conflicts and maintain the integrity of data when multiple processes seek to alter stored values at the same time.

**importance:**

It also prevents common issues like dirty reads, lost updates, and non-repeatable reads. The key features include deadlock avoidance, ensuring serializability of transactions, and maintaining data integrity.