How is inheritance implemented in SQL?

Inheritance is specified via the **UNDER** keyword

Example:

CREATE TYPE Manager_type UNDER Emp_type AS (dept_managed CHAR (20));

Manager_type **inherits** all features of Emp_type and it has an additional attribute called dept_managed

What is pre-processing? What is the need of this step while building Data Warehouse?

Data pre-processing refers to cleaning, transforming, and integrating data in order to make it ready for analysis.

Steps of data pre-processing:

- **Data Cleaning:** This involves identifying and correcting errors or inconsistencies in the data.
- **Data Integration:** This involves combining data from multiple sources to create a unified dataset.
- **Data Transformation:** This involves converting the data into a suitable format for analysis.
- **Data Reduction:** This involves reducing the size of the dataset while preserving the important information.

Need of data pre-processing:

- **Improves Data Quality:** Corrects errors to improve accuracy and consistency of data.
- **Enhances data integration:** Resolves discrepancies and harmonizes data from various sources.
- **Optimizes Data Storage and Performance:** Reduces the volume of data and enhances the efficiency of queries.
- **Facilitates Better Data Analysis:** Performs normalization and aggregation of data.

How does the concept of an object in the object-oriented model differ from the concept of an entity in the E-R model?

Entity	Object
Entity is a real time object that can be distinguished from other objects.	Object is an entity that has all the attributes and the actions required to be taken.

Entity has attributes.	Object has life span and object identifier.
Entity is a uniquely identifiable object.	An object can be identified using its identifier.
Every entity has a primary key.	Object does not have a primary key.
Entity is a part of relational database.	Object is a part of object oriented database.
Entities are represented in rectangular shape using E-R diagram.	Objects aren't represented graphically.
Attributes is a property of entity.	Inheritance, Encapsulation, Polymorphism and Abstraction are properties of object.
Example: Computer, Software.	Example: Minimum age to vote is 18.

Suppose a new class is created as a subclass of a class 'C'. What changes may have to be made to the database objects.

Table Creation: If the new class introduces new attributes, a new table may need to be created to represent it.

Table Modification: If the new class inherits attributes from its superclass, the table representing the subclass should include these inherited attributes along with any additional attributes specific to the subclass.

Primary Key Constraints: Ensure that primary key constraints are defined for the new table representing the subclass.

Foreign Key Constraints: If the subclass references other tables as part of its definition, foreign key constraints may need to be updated or added to maintain referential integrity.

Data Migration: If there are existing instances of the superclass in the database, data may need to be migrated to accommodate instances of the subclass.

Permissions: Ensure that appropriate permissions are set for the new table representing the subclass to restrict access as necessary.

Security: Review and update security policies to ensure that users have the appropriate access rights to the subclass.

Documentation and Metadata: Update database documentation and metadata to reflect the changes introduced by the subclass.

Discuss the relative advantages of centralized and distributed database.

Advantages of centralized database:

- It is easier to access data.
- It has minimal data redundancy.
- It is cheaper in comparison to all other databases available.
- It is easier to enforce security policies
- It provides higher data integrity and consistency
- It simplifies backup and recovery processes

Advantages of distributed database:

- It can be easily expanded as data is spread across different physical locations.
- It can be easily accessed from different networks.
- It is more secure in comparison to a centralized database.
- It can handle large amounts of data
- It has higher fault tolerance

Explain the differences between Fragmentation transparency, Replication transparency and Location transparency

Aspect	**Fragmentation Transparency**	**Replication Transparency**	**Location Transparency**
Definition	Hides the details of data fragmentation across different locations.	Hides the existence of multiple copies (replicas) of data.	Hides the physical location of data in the distributed system.
Purpose	Allows users to interact with the data without knowing about fragmentation.	Allows users to interact with data without knowing about replicas.	Allows users to interact with data using logical identifiers, not physical locations.
Types	Horizontal, vertical and hybrid fragmentation.	Full and partial replication.	Data can be stored in various physical locations.

Why is it not desirable to force users to make an explicit choice of a query-processing strategy? Explain in detail.

It is undesirable due to the following reasons:

- **User Expertise:** Selecting an appropriate query-processing strategy requires deep knowledge of database internals, which most users lack.
- **Increased Complexity:** Requiring users to choose a query-processing strategy adds significant complexity to user tasks.
- **Query Optimizer Role:** Modern DBMSs have sophisticated optimizers that automatically select the best execution plan.
- **Suboptimal Choices:** Users might make poor choices, leading to inefficient performance.

- **Resource Management:** The DBMS manages resources more efficiently than individual users.
- **Evolving Database Systems:** Manual strategies require frequent updates, increasing maintenance.

Explain why a non-recoverable schedule results in a loss of transaction atomicity?

A non-recoverable schedule results in a loss of transaction atomicity because it allows transactions to commit based on the uncommitted changes of other transactions. If an uncommitted transaction fails or is rolled back, it becomes impossible to ensure the atomicity of all dependent transactions, leading to an inconsistent state.

Example: Consider the following transactions:

- Transaction T1:
  1) Write X.
  2) (Not yet committed or rolled back)

- Transaction T2:
  1) Read X (the value written by T1).
  2) Commit.

In a non-recoverable schedule, T2 commits based on the value of X written by T1 before T1 itself commits. If T1 later aborts, the changes made by T2 are based on a value that never should have existed, leading to inconsistency. Since T2 has committed, we cannot roll it back to rectify the inconsistency caused by T1's abort. This breaks the atomicity of T2 because part of its operations depend on an invalid state.

What is the purpose of Distributed Transaction Modelling and concurrency control?

Purpose of Distributed Transaction Modelling:

- Ensure Atomicity: Ensure that a distributed transaction either completes entirely or has no effect at all.
- Maintain Consistency: Ensure that a distributed transaction brings the database from one consistent state to another, even when multiple databases are involved.
- Support Isolation: Ensure that concurrent transactions do not interfere with each other.
- Durability: Ensure that once a transaction has been committed, its effects are permanently recorded in the database, even in case of a system failure.

Purpose of Concurrency Control:

- Prevent Conflicts: Prevents conflicts that arise from concurrent transactions accessing shared data, ensuring that the database remains in a consistent state.
- Maintain Isolation Levels: Provides different levels of isolation (e.g., read committed, repeatable read, serializable) to balance the trade-off between consistency and performance according to application requirements.
- Avoid Anomalies: Prevents anomalies such as lost updates, dirty reads, non-repeatable reads, and phantom reads, ensuring data integrity and correctness.