TCS Interview Question of DBMS

Question on Trigeers.

1) what are Trigeers ?

ans :

- A trigger is a stored procedure in a database that automatically invokes whenever a special event in the database occurs.
- For example, a trigger can be invoked when a row is inserted into a specified table or when specific table columns are updated.
- In simple words, a trigger is a collection of SQL statements with particular names that are stored in system memory.
- Because a trigger cannot be called directly, unlike a stored procedure, it is referred to as a special procedure.
- Trigger Structure :

```
create trigger [trigger_name]
//Creates or replaces an existing trigger with the      trigger_name.
[before | after]
//This specifies when the trigger will be executed.
{insert | update | delete}
//This specifies the DML operation.
on [table_name]
//This specifies the name of the table associated with the trigger.
[for each row]
//This specifies a row-level trigger, i.e., the trigger will be executed for each affected row.
[trigger_body]
//This provides the operation to be performed as the trigger is fired
```

2) What are the different types of Trigger ?

Ans :

1. DDL Trigger
2. DML Trigger
3. Logon Triggers

1.DDL Trigger:

The Data Definition Language (DDL) command events such as Create_table, Create_view, drop_table, Drop_view, and Alter_table cause the DDL triggers to be activated.

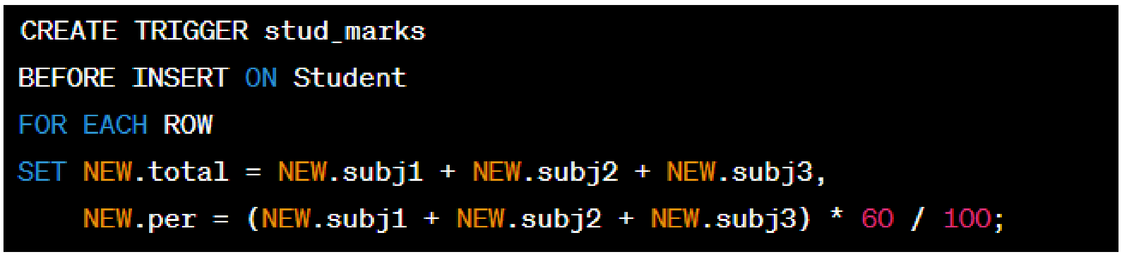
Trigger Example :

```
create trigger deep
on emp
for
insert,update ,delete
as
print 'you can not insert,update and delete this table i'
rollback;
```

2. **DML Trigger**

The Data uses manipulation Language (DML) command events that begin with Insert, Update, and Delete set off the DML triggers. corresponding to insert_table, update_view, and delete_table.

Given Student Report Database, in which student marks assessment is recorded. In such a schema, create a trigger so that the total and percentage of specified marks are automatically inserted whenever a record is inserted.



```
CREATE TRIGGER stud_marks
BEFORE INSERT ON Student
FOR EACH ROW
SET NEW.total = NEW.subj1 + NEW.subj2 + NEW.subj3,
    NEW.per = (NEW.subj1 + NEW.subj2 + NEW.subj3) * 60 / 100;
```

Above SQL statement will create a trigger in the student database in which whenever subjects marks are entered, before inserting this data into the database, the trigger will compute those two values and insert them with the entered values. I.e.

3 Logon Triggers:

- When a user session is created with a SQL Server instance after the authentication process of logging is finished but before establishing a user session, the LOGON event takes place.
- these triggers can be used to track login activity or set a limit on the number of sessions that a given login can have in order to audit and manage server sessions.

Q3. What are the advantages and Disadvantage of Trigger ?

Ans ; Advantages :

1. Database object rules are established by triggers, which cause changes to be undone if they are not met.

2.The trigger will examine the data and, if necessary, make changes.

3.We can enforce data integrity thanks to triggers.

4.Data is validated using triggers before being inserted or updated.

5.Triggers assist us in maintaining a records log.

Disadvantages :

1. Only triggers permit the use of extended validations.

2. Automatic triggers are used, and the user is unaware of when they are being executed. Consequently, it is difficult to troubleshoot issues that arise in the database layer.

3. The database server's overhead may increase as a result of triggers.

Question on Stored Procedure :

4. What is difference between store procedure and Trigger?

Ans :

Trigger cannot be called directly, unlike a stored procedure, it is referred to as a special procedure. A trigger is automatically called whenever a data modification event against a table takes place, which is the main distinction between a trigger and a procedure. On the other hand, a stored procedure must be called directly.

The following are the key differences between triggers and stored procedures:

1. Triggers cannot be manually invoked or executed.
2. There is no chance that triggers will receive parameters.
3. A transaction cannot be committed or rolled back inside a trigger.

5. What is Stored procedure .

ans :

Definition:

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.

So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.

We can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.

Syntax:

```
CREATE PROCEDURE procedure_name
AS
sql_statement
GO;
```

Execute a Stored Procedure:

```
EXEC procedure_name;
```

Example :

```
CREATE PROCEDURE SelectAllCustomers
AS
SELECT * FROM Customers
GO;
EXEC SelectAllCustomers;
```

6 .Why we use store procedure in sql.

Ans :

Stored procedures in SQL are used for several important reasons, offering a range of benefits for database management, performance optimization, security, and maintainability.

Reduced Network Traffic: Stored procedures execute on the server, minimizing the amount of data sent between the server and the client.

Execution Plan Reuse: The database server can cache and reuse the execution plan of a stored procedure, improving performance.

- **SQL Injection Prevention**: Using parameters in stored procedures can help prevent SQL injection attacks, as they separate data from code.

7 .What is Sql injection?

Ans :

SQL injection is the placement of malicious code in SQL statements, via web page input.

SQL injection is one of the most common web hacking techniques.

SQL injection is a code injection technique that might destroy your database.

It occurs when an attacker is able to insert or "inject" malicious SQL code into a query, potentially allowing them to view, modify, or delete data in the database, or even execute administrative operations on the database.

Example :

```
SELECT * FROM Users WHERE username = 'user_input';

/* If the application concatenates the user input directly into the SQL query without proper
sanitization, an attacker could input something like: */

' OR '1'='1

The resulting SQL query would be:

SELECT * FROM Users WHERE username = '' OR '1'='1';
```

This query is always true and would return all rows in the Users table, potentially exposing sensitive data.

8. How to prevent data from Sql Injection?

Ans :

Parameterized Queries / Prepared Statements: Use parameterized queries to ensure user input is treated as data, not executable code.

Example :

```
import sqlite3

connection = sqlite3.connect('example.db')
cursor = connection.cursor()

# Using a parameterized query
cursor.execute("SELECT * FROM Users WHERE username = ?", (user_input,))
```

2 Stored Procedures: Use stored procedures, which can help to ensure that input is treated as parameters.

Example :

```
CREATE PROCEDURE GetUserByUsername
   @username NVARCHAR(50)
AS
BEGIN
   SELECT * FROM Users WHERE username = @username;
END;
```

3. **Input Validation and Sanitization**: Validate and sanitize all user inputs to ensure they conform to expected formats and do not contain malicious code.

3. Question on Normalization:

9. What is database normalization, and why is it important in database design?

Ans :

Database normalization is the process of organizing data to minimize data redundancy and dependency, resulting in a more efficient and well-structured database schema.

10 .What is the First Normal Form (1NF) in database normalization?

Answer: 1NF requires that each column in a table holds only atomic (indivisible) values and that all values in a column are of the same data type.

11. What is an atomic value, and how does it relate to 1NF?

Answer: An atomic value is a value that cannot be divided further. In 1NF, each column must contain atomic values, ensuring that data is not grouped or combined in a single field.

12. What is an atomic value, and how does it relate to 1NF?

Answer: An atomic value is a value that cannot be divided further. In 1NF, each column must contain atomic values, ensuring that data is not grouped or combined in a single field.

13. What is the Second Normal Form (2NF), and why is it an improvement over 1NF?

Answer: 2NF builds upon 1NF by adding a requirement that all non-key attributes (columns) are fully functionally dependent on the entire primary key. It eliminates partial dependencies.

14.What is the Third Normal Form (3NF), and how does it further refine the database schema?

Answer: 3NF extends 2NF by eliminating transitive dependencies. It ensures that non-key attributes depend only on the primary key, not on other non-key attributes.

15.. What is the Boyce-Codd Normal Form (BCNF), and when is it applied in database design?

Answer: BCNF is a more stringent form of normalization that addresses situations where there may be multiple candidate keys in a table. It ensures that non-prime attributes are functionally dependent on candidate keys.

16. Explain the concept of a candidate key in the context of BCNF.

Answer: A candidate key is a set of attributes that can uniquely identify a tuple (row) in a table. In BCNF, non-prime attributes must be functionally dependent on candidate keys.

17.What are prime and non-prime attributes in the context of BCNF?

Answer: Prime attributes are part of the candidate key(s), while non-prime attributes are not. In BCNF, non-prime attributes must be functionally dependent on candidate keys.

18. What is the purpose of the Fourth Normal Form (4NF), and when is it applied in database design?

Answer: 4NF addresses multi-valued dependencies, ensuring that data can be split into separate tables to eliminate redundancy while preserving information integrity.

19. What is Data Anamolies

Ans:

Anomalies in the relational model refer to inconsistencies or errors that can arise when working with relational databases, specifically in the context of data insertion, deletion, and modification. There are different types of anomalies that can occur in referencing and referenced relations which can be discussed as:

These anomalies can be categorized into three types:

- Insertion Anomalies
- Deletion Anomalies
- Update Anomalies.

20. Advantages of Data Anamolies ?

Ans :**Data Integrity:** Ensure data Accurancy

Scalability: highly scalable and can handle large amounts of data without sacrificing performance**.**

21.How Anomalies can be removed?

Ans : Anomalies can be removed with the process of Normalization. Normalization involves organizing data into tables and applying rules to ensure data is stored in a consistent and efficient manner.

22. What is view in the database .

Ans :

Views in SQL are a kind of virtual table. A view also has rows and columns as they are on a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain conditions.

23.What are aggregate and scalar functions?

Ans :

Aggregation Function:

Aggregate functions perform a calculation on a set of values and return a single value. They are often used to summarize or group data.

Scalar Function:

Scalar functions operate on a single value and return a single value. They are used to perform operations on individual data values

23. What is the Difference between Aggregate function and Scalar function.

Ans :

Aggregate Function:

Operate on a set of values (multiple rows).

Output: Return a single value that summarizes the input set.

Usage: Commonly used with GROUP BY clauses to group rows that share a property so that an aggregate function can be applied to each group.

Examples:

- COUNT(): Counts the number of rows.
- SUM(): Calculates the total sum of a numeric column.
- AVG(): Computes the average value of a numeric column.
- MIN(): Finds the minimum value in a column.
- MAX(): Finds the maximum value in a column.

Scalar Function :

The primary difference between aggregate functions and scalar functions in SQL lies in how they operate on data and the type of result they produce. Here's a detailed comparison:

Aggregate Functions

Aggregate functions perform a calculation on a set of values and return a single value. They are often used to summarize or group data.

Key Characteristics:

1. **Input**: Operate on a set of values (multiple rows).
2. **Output**: Return a single value that summarizes the input set.
3. **Usage**: Commonly used with GROUP BY clauses to group rows that share a property so that an aggregate function can be applied to each group.
4. **Examples**:
   - **COUNT()**: Counts the number of rows.
   - **SUM()**: Calculates the total sum of a numeric column.
   - **AVG()**: Computes the average value of a numeric column.
   - **MIN()**: Finds the minimum value in a column.
   - **MAX()**: Finds the maximum value in a column.

Example:

sql

Copy code

```
SELECT Department, COUNT(*) AS NumberOfEmployees

FROM Employees

GROUP BY Department;
```

This query counts the number of employees in each department.

Scalar Functions

Scalar functions operate on a single value and return a single value. They are used to perform operations on individual data values.

Key Characteristics:

1. **Input**: Operate on a single value (one row or one column value).
2. **Output**: Return a single value for each input value.
3. **Usage**: Often used to transform or manipulate individual data values within a query.
4. **Examples**:
   - **UPPER()**: Converts a string to uppercase.
   - **LOWER()**: Converts a string to lowercase.
   - **LEN()**: Returns the length of a string.
   - **ROUND()**: Rounds a numeric value to a specified number of decimal places.

24. What is a subquery?

Ans :

Subquery is a query that is embedded in the WHERE clause of another SQL query.

25.What are the different operators available in SQL?

Ans : There are three operators available in SQL namely:

1. Arithmetic Operators
2. Logical Operators
3. Comparison Operators

26. What are Union, minus, and Interact commands?

Ans :
Set Operations in SQL eliminate duplicate tuples and can be applied only to the relations which are union compatible.
UNION Operation: This operation includes all the tuples which are present in either of the relations. For example: To find all the customers who have a loan or an account or both in a bank.

```
SELECT CustomerName FROM Depositor
 UNION
 SELECT CustomerName FROM Borrower ;
```

INTERSECT Operation: This operation includes the tuples which are present in both of the relations. For example: To find the customers who have a loan as well as an account in the bank:

```
SELECT CustomerName FROM Depositor
 INTERSECT
 SELECT CustomerName FROM Borrower ;
```

EXCEPT for Operation:This operation includes tuples that are present in one relationship but should not be present in another relationship. For example:To find customers who have an account but no loan at the bank

```
SELECT CustomerName FROM Depositor
 EXCEPT
 SELECT CustomerName FROM Borrower ;
```

27. What is a Cursor?

Ans : The cursor is a Temporary Memory or Temporary Work Station. It is Allocated by Database Server at the Time of Performing DML operations on the Table by the User. Cursors are used to store Database Tables.

28. Write down various types of relationships in SQL?

Ans :

There are various relationships, namely:

- One-to-One Relationship.
- One to Many Relationships.
- Many to One Relationship.
- Self-Referencing Relationship.

29.Name different types of case manipulation functions available in SQL.

Ans :1) Lower()

2)Upper()

3)INITCAP: This function return the first letter to the uppercase .

30.**Name the function which is used to remove spaces at the end of a string?**

Ans : In SQL, the spaces at the end of the string are removed by a trim function.

Syntax:

Trim(string) ,

31.Which operator is used in queries for pattern matching?

Ans : LIKE operator: It is used to fetch filtered data by searching for a particular pattern in the where clause.

32. What is the SQL query to display the current date?

Ans:CURRENT_DATE returns to the current date. This function returns the same value if it is executed more than once in a single statement, which means that the value is fixed, even if there is a long delay between fetching rows in a cursor.

Syntax:

CURRENT_DATE or CURRENT DATE

33. Name the operator which is used in the query for appending two strings?

Ans :
In SQL for appending two strings, the " Concentration operator" is used and its symbol is " || ".

Keys/Constraint in SQL

34. What is Constraints ?

Ans :

Constraints are the rules that we can apply to the type of data in a table. That is, we can specify the limit on the type of data that can be stored in a particular column in a table using constraints

35. What do you mean by foreign key?

Ans :
A Foreign key is a field that can uniquely identify each row in another table. And this constraint is used to specify a field as a Foreign key. That is this field points to the primary key of another table. This usually creates a kind of link between the two tables.

```
CREATE TABLE Orders
(
O_ID int NOT NULL,
ORDER_NO int NOT NULL,
C_ID int,
PRIMARY KEY (O_ID),
FOREIGN KEY (C_ID) REFERENCES Customers(C_ID)
)
```

36.What is the primary key?

Ans :

A Primary Key is one of the candidate keys. One of the candidate keys is selected as the most important and becomes the primary key. There cannot be more than one primary key in a table.

37. What is a Default constraint?

Ans :

The DEFAULT constraint is used to fill a column with default and fixed values. The value will be added to all new records when no other value is provided.

38. What is the difference between primary key and unique constraints?

Ans:
The primary key cannot have NULL values, the unique constraints can have NULL values. There is only one primary key in a table, but there can be multiple unique constraints. The primary key creates the clustered index automatically but the unique key does not.

39. What is Auto Increment?

Ans :
Sometimes, while creating a table, we do not have a unique identifier within the table, hence we face difficulty in choosing Primary Key. So as to resolve such an issue, we've to manually provide unique keys to every record, but this is often also a tedious task. So we can use the Auto-Increment feature that automatically generates a numerical Primary key value for every new record inserted. The Auto Increment feature is supported by all the Databases.

Join Question

40. What is a join in SQL? What are the types of joins?

Ans :

An SQL Join statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are:

- **INNER JOIN**: The INNER JOIN keyword selects all rows from both tables as long as the condition is satisfied. This keyword will create the result set by combining all rows from both the tables where the condition satisfies i.e. the value of the common field will be the same.
- **LEFT JOIN**: This join returns all the rows of the table on the left side of the join and matching rows for the table on the right side of the join. For the rows for which there is no matching row on the right side, the result set will be null. LEFT JOIN is also known as LEFT OUTER JOIN
- **RIGHT JOIN**: RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of the join. For the rows for which there is no matching row on the left side, the result set will contain null. RIGHT JOIN is also known as RIGHT OUTER JOIN.
- **FULL JOIN**: FULL JOIN creates the result set by combining the results of both LEFT JOIN and RIGHT JOIN. The result set will contain all the rows from both tables. For the rows for which there is no matching, the result set will contain NULL values.

Clause In Sql

41. Define SQL Order by the statement

Ans :

The ORDER BY statement in SQL is used to sort the fetched data in either ascending or descending according to one or more columns.

By default ORDER BY sorts the data in ascending order.

We can use the keyword DESC to sort the data in descending order and the keyword ASC to sort in ascending order.

42. Explain SQL Having statement?

Ans : HAVING is used to specify a condition for a group or an aggregate function used in the select statement. The WHERE clause selects before grouping. The HAVING clause selects rows after grouping. Unlike the HAVING clause, the WHERE clause cannot contain aggregate functions.

43.What is the Difference between Where and Having Clause .

Ans : **WHERE Clause:**

The `WHERE` clause is used to filter rows before any grouping or aggregation takes place. It operates on individual rows and is used to specify conditions that the rows must meet to be included in the result set.

Cannot be used with aggregate functions like `SUM()`, `AVG()`, `COUNT()`, etc., directly in the condition.

Filters rows before grouping and aggregation.

Operates on individual rows.

Appears before the `GROUP BY` clause.

Having Clause :

The `HAVING` clause is used to filter groups after the `GROUP BY` clause has been applied. It operates on aggregated data and is typically used in combination with aggregate functions.

Used only with `GROUP BY` to filter groups based on aggregate functions.

Cannot be used without a `GROUP BY` clause (unless in a context with aggregate functions).

Filters groups after grouping and aggregation.Filters groups after grouping and aggregation.

Operates on groups.

Appears after the Group By clause .

43. What is the Difference between SQL and NoSQL.

Ans :

1) Sql has predefine Structure and fix Schema Whereas No SQL does not have Fix Schema and predefine Structure.
2) SQL uses Relational Database model Whereas NoSQL uses various data model like Document Oriented ,Key-Value pair ,graph.
3) Sql is vertically scalable whereas NoSQL is Horizontal Scalabe
4) SQL follows ACID properties whereas NoSQL support CAP(Consistency ,Avalibility,Partition tolerance)
5) Generally Slower for the large scale read and Write operation Whereas NOSQL is Faster in read and Write operation in the large database.
6) SQL is less flexible vs NoSQL is more Flexible .