Madhur Mehta

• SQL Interview Questions(Mixed)

- **Question:** Explain a scenario where you'd use a stored procedure over a regular query.
- **Answer:** Stored procedures can encapsulate complex logic, accept parameters, and return values. They are useful for repetitive tasks, like processing business logic on the database side, or for batch processing.

- **Question:** Describe a situation where you would use a SQL trigger.
- **Answer:** Triggers can automatically execute actions in response to specific events on a table, like insertions or updates. For instance, you might use a trigger to maintain an audit log every time a record in a table is modified.

- **Question:** How do you handle errors in a stored procedure?
- **Answer:** Many SQL dialects provide error handling mechanisms like **TRY...CATCH** blocks. Inside these blocks, you can log errors, rollback transactions, or take other corrective actions.

- **Question:** Can you give an example of a **CASE** statement in SQL?
- **Answer:**

```
SELECT name,
CASE WHEN age < 18 THEN 'Minor'
WHEN age BETWEEN 18 AND 65 THEN 'Adult'
ELSE 'Senior'
END AS age_group
FROM people;
```

- **Question:** Explain the difference between **UNION** and **JOIN**.
- **Answer: UNION** combines rows from two or more tables based on a common column, eliminating duplicates. **JOIN** relates two or more tables based on a related column between them.

- **Question:** Describe a scenario where a recursive stored procedure might be useful.
- **Answer:** Recursive stored procedures call themselves. They can be useful for hierarchical data, such as retrieving an employee's complete reporting structure in an organization.

- **Question:** What is a transaction in SQL and why is it important?
- **Answer:** A transaction groups one or more SQL statements into a single unit of work. It ensures data integrity by making sure all operations within it either fully complete or fully fail, maintaining the database's consistent state.

- **Question:** Describe a scenario where you would use the **ROLLBACK** command.
- **Answer:** If you're performing multiple related changes and an error occurs in one of them, you'd use **ROLLBACK** to revert all changes made during the transaction to maintain data integrity.

- **Question:** How would you combine the results of two tables that have similar structures but can't guarantee there are no duplicate records between them?

- **Answer: UNION ALL** to combine the results. If you want to remove duplicates between the two tables, simply use **UNION**.

```
SELECT *
FROM table1
UNION ALL
SELECT * FROM table2;
```

- **Question:** Can you give an example of an **AFTER INSERT** trigger?
- **Answer:**

```
CREATE TRIGGER after_insert_trigger
AFTER INSERT
ON employees
FOR EACH ROW
INSERT INTO audit_log (action, table_name, record_id, change_date)
VALUES ('INSERT', 'employees', NEW.id, NOW());
```

- **Question:** What's the difference between a temp table and a table variable?
- **Answer:** Temp tables are temporary and are stored in tempdb. They can have indexes and statistics. Table variables are stored in memory, typically have a smaller scope, and don't have statistics.

- **Question:** What is the purpose of the **COALESCE** function and how does it differ from the **NULLIF** function?
- **Answer: COALESCE** returns the first non-NULL value in a list.
- **NULLIF** returns NULL if two specified values are equal, otherwise it returns the first value.

- **Question:** Describe a scenario where you'd use the **MERGE** statement.
- **Answer: MERGE** (or **UPSERT**) is used to insert, update, or delete records based on a source. For example, synchronizing two tables where you'd like to update records if they exist, insert if they don't, and delete if they're no longer present in the source.

- **Question:** How can you ensure a stored procedure is recompiled the next time it's run?
- **Answer:** In SQL Server, you can use the **WITH RECOMPILE** option when executing the stored procedure. This ensures the procedure is recompiled, which might be necessary if the data landscape has changed significantly.

- **Question:** What is a correlated subquery?
- **Answer:** It's a subquery that references columns from the outer query. Execution depends on the row being processed in the outer query. For example, fetching employees who earn above the average salary in their department would use a correlated subquery.

- **Question:** How can you retrieve the last record from a table without using the **ORDER BY** clause?
  Answer: One approach is to use the **LIMIT** clause with an offset equal to the count of rows minus one. Another is to use a subquery to match the maximum ID or date value, depending on the table structure.

- **Question:** Explain the difference between a function and a stored procedure.
- **Answer:** Both encapsulate logic but functions return a single value or table and can be used in a SELECT statement. Stored procedures can return multiple values, can have output parameters, and don't return tables directly in a SELECT.

- **Question:** Describe a scenario where you would use a **CROSS JOIN**.
- **Answer: CROSS JOIN** returns the Cartesian product of two tables. Useful when you want combinations of all rows from two tables, such as combining colors and sizes for product options.

- **Question:** How do you handle a deadlock?
- **Answer:** Deadlocks can be minimized using best practices like accessing resources in a consistent order, keeping transactions short, and using the appropriate transaction isolation level. When they occur, databases often have mechanisms to detect and resolve them, like rolling back one of the transactions.

- **Question:** Explain how you can optimize a stored procedure
- **Answer:** Optimization can include ensuring the logic is efficient, using indexes appropriately, avoiding cursors if possible, analyzing and optimizing the query plan, and periodically reviewing and refactoring the code for improvements.