**Basic SQL Questions:**

1. **What is SQL?**
   * Answer: SQL (Structured Query Language) is a standard language for managing and manipulating relational databases. It is used to perform tasks like querying, updating, and managing data in a database.
2. **What are the different types of SQL statements?**
   * Answer: SQL statements are divided into five categories:
     + **DDL (Data Definition Language):** CREATE, ALTER, DROP, TRUNCATE
     + **DML (Data Manipulation Language):** SELECT, INSERT, UPDATE, DELETE
     + **DQL (Data Query Language):** SELECT
     + **TCL (Transaction Control Language):** COMMIT, ROLLBACK, SAVEPOINT
     + **DCL (Data Control Language):** GRANT, REVOKE
3. **What is a Primary Key?**
   * Answer: A Primary Key is a column or set of columns that uniquely identifies each row in a table. A table can have only one Primary Key, and it cannot have NULL values.
4. **What is the difference between DELETE, TRUNCATE, and DROP?**
   * DELETE: Removes rows based on a condition, can be rolled back, and triggers are activated.
   * TRUNCATE: Removes all rows from a table without logging each row deletion, cannot be rolled back, and does not activate triggers.
   * DROP: Deletes the table itself along with its structure and all data.
5. **What is a Foreign Key?**
   * Answer: A Foreign Key is a column or set of columns that references the Primary Key of another table. It helps maintain referential integrity between tables.
6. **What is the difference between WHERE and HAVING?**
   * WHERE: Used to filter rows before aggregation.
   * HAVING: Used to filter rows after aggregation, typically used with GROUP BY.

**Intermediate SQL Questions:**

1. **What is a Join? What are the different types of Joins?**
   * Answer: A Join is used to combine rows from two or more tables based on a related column.
     + **INNER JOIN:** Returns only the matching rows from both tables.
     + **LEFT (OUTER) JOIN:** Returns all rows from the left table and matching rows from the right table. Non-matching rows from the right table are NULL.
     + **RIGHT (OUTER) JOIN:** Returns all rows from the right table and matching rows from the left table. Non-matching rows from the left table are NULL.
     + **FULL (OUTER) JOIN:** Returns all rows when there is a match in either table, and NULL for non-matching rows.
     + **CROSS JOIN:** Returns the Cartesian product of both tables.
2. **What is the difference between UNION and UNION ALL?**
   * UNION: Combines the result sets of two queries and removes duplicates.
   * UNION ALL: Combines the result sets of two queries without removing duplicates.
3. **What is a subquery, and what are the different types?**
   * Answer: A subquery is a query within another query. It can be used in SELECT, INSERT, UPDATE, or DELETE statements.
     + **Correlated Subquery:** Depends on the outer query for its value.
     + **Non-Correlated Subquery:** Independent of the outer query.
4. **What is the difference between RANK(), DENSE\_RANK(), and ROW\_NUMBER()?**
   * RANK(): Assigns a rank to each row, with gaps in ranking for ties.
   * DENSE\_RANK(): Assigns a rank without gaps for ties.
   * ROW\_NUMBER(): Assigns a unique sequential number to each row.
5. **Explain the use of the GROUP BY clause in SQL.**
   * Answer: The GROUP BY clause is used to group rows that have the same values in specified columns. It is often used with aggregate functions (COUNT(), SUM(), AVG(), MAX(), MIN()).
6. **What is a View in SQL, and why is it used?**
   * Answer: A View is a virtual table based on the result of an SQL query. It is used to simplify complex queries, improve security, and reuse queries.

**Advanced SQL Questions:**

1. **What is a CTE (Common Table Expression), and how does it differ from a subquery?**
   * Answer: A CTE is a temporary result set defined within the execution scope of a SELECT, INSERT, UPDATE, or DELETE statement. It can be referenced multiple times within the query and improves readability. Unlike subqueries, CTEs are more modular and reusable.
2. **What are Window Functions in SQL?**
   * Answer: Window functions operate over a subset of rows (window) and perform calculations across rows without collapsing them into a single value. Examples include RANK(), ROW\_NUMBER(), LEAD(), LAG(), SUM() OVER(), and AVG() OVER().
3. **What is a Trigger, and when would you use one?**
   * Answer: A Trigger is a stored procedure that automatically executes in response to certain events on a table, such as INSERT, UPDATE, or DELETE. Triggers are used for enforcing business rules, data integrity, and auditing changes in a database.
4. **Explain ACID properties in SQL.**
   * Answer: ACID properties ensure reliable processing of transactions:
     + **Atomicity:** Transactions are all-or-nothing.
     + **Consistency:** Transactions leave the database in a valid state.
     + **Isolation:** Transactions are isolated from one another.
     + **Durability:** Once a transaction is committed, it persists even after a system crash.
5. **What is a Stored Procedure, and why is it useful?**
   * Answer: A Stored Procedure is a precompiled set of one or more SQL statements that can be executed as a single unit. They improve performance (due to precompilation), modularize code, and allow code reuse and better security.
6. **What is Normalization? Explain different Normal Forms.**
   * Answer: Normalization is the process of organizing data in a database to minimize redundancy. The key normal forms are:
     + **1NF:** Ensures atomicity of data.
     + **2NF:** Removes partial dependencies (no non-key attribute is dependent on a part of the Primary Key).
     + **3NF:** Removes transitive dependencies (no non-key attribute is dependent on another non-key attribute).
7. **What is the difference between an Index and a Clustered Index?**
   * **Index:** A structure that improves the speed of data retrieval.
   * **Clustered Index:** Defines the physical order of data in the table. A table can have only one clustered index.
8. **What are Transactions, and how do you manage them in SQL?**
   * Answer: A transaction is a unit of work that is either completed in its entirety or not at all. Transactions are managed using BEGIN TRANSACTION, COMMIT, and ROLLBACK.

**Scenario-based Questions:**

1. **Write a SQL query to find the second-highest salary in a table Employees.**

sql

Copy code

SELECT MAX(salary)

FROM Employees

WHERE salary < (SELECT MAX(salary) FROM Employees);

1. **How would you optimize a slow SQL query?**
   * Check for indexes on the columns used in WHERE and JOIN clauses.
   * Avoid SELECT \* and retrieve only necessary columns.
   * Review execution plans.

### Consider query restructuring or using partitioning, indexing, or caching techniques. 1. SQL Server Architecture:

* Can you explain the architecture of SQL Server?
* How does SQL Server handle memory management?
* What is the difference between a clustered and a non-clustered index?

**2. Indexing:**

* What is the purpose of an index in SQL Server?
* How would you decide whether to use a clustered or non-clustered index?
* Explain the concept of a covering index.
* How do you identify and resolve index fragmentation in SQL Server?

**3. Performance Tuning:**

* How do you analyze and optimize slow-running queries?
* What are the different ways to improve query performance in SQL Server?
* Explain the use of the **Query Execution Plan** and how you interpret it.
* What is **SQL Profiler**, and how can it be used to tune performance?

**4. Transactions:**

* What is a transaction, and how do you ensure atomicity in SQL Server?
* Explain **ACID properties** in SQL Server.
* What is the difference between **COMMIT** and **ROLLBACK**?
* What is a **deadlock** and how do you handle or prevent it?

**5. Joins and Subqueries:**

* What are the different types of joins available in SQL Server?
* Can you explain the difference between an **INNER JOIN** and a **LEFT OUTER JOIN**?
* How do correlated subqueries work, and when would you use one?

**6. Stored Procedures & Functions:**

* What are stored procedures, and how are they different from user-defined functions?
* How do you optimize a stored procedure for performance?
* Explain the concept of **temporary tables** and **table variables** in SQL Server.

**7. Error Handling:**

* How do you implement error handling in SQL Server?
* What are **TRY...CATCH** blocks in SQL Server?
* How do you handle deadlock scenarios in stored procedures?

**8. Concurrency Control:**

* What is the difference between **pessimistic** and **optimistic** concurrency control?
* How does SQL Server handle locking? Explain the difference between **lock escalation** and **lock hints**.

**9. Backup and Recovery:**

* How do you perform a full backup, differential backup, and transaction log backup in SQL Server?
* What are the differences between **full recovery mode** and **simple recovery mode**?
* How do you restore a database from a backup in SQL Server?

**10. SQL Server Security:**

* What are the various types of authentication modes available in SQL Server?
* How do you grant and revoke permissions to a user in SQL Server?
* What is the difference between **server-level** and **database-level roles**?

**11. Data Types and Constraints:**

* What are some common data types in SQL Server, and how do you choose the right one for your columns?
* Explain the difference between **CHAR** and **VARCHAR**.
* What are primary key, foreign key, and unique constraints?

**12. TempDB:**

* What is **TempDB** and how is it used in SQL Server?
* How do you monitor and troubleshoot TempDB performance issues?

**13. Dynamic SQL:**

* What is **dynamic SQL**, and how do you execute it in SQL Server?
* What are the potential risks or disadvantages of using dynamic SQL?

**14. Partitioning:**

* What is **table partitioning**, and when would you use it?
* How do you implement partitioning in SQL Server?

**15. CTEs and Window Functions:**

* What is a **Common Table Expression (CTE)**, and how does it differ from a derived table?
* Can you explain how **window functions** work in SQL Server (e.g., **ROW\_NUMBER()**, **RANK()**, **LEAD()**, **LAG()**)?

**16. Normalization and Denormalization:**

* Explain **normalization** and its importance.
* When would you consider denormalizing a database, and what are the trade-offs?

**17. Triggers:**

* What are triggers, and when would you use one in SQL Server?
* What is the difference between an **AFTER TRIGGER** and an **INSTEAD OF TRIGGER**?

**18. SQL Server Agent Jobs:**

* How do you schedule tasks in SQL Server using **SQL Server Agent**?
* How would you troubleshoot a failed SQL Server Agent job?