**Key Topics to Prepare**

1. **Basic SQL Concepts**:
   * Difference between DELETE, TRUNCATE, and DROP.
   * Understanding of JOIN operations (INNER, LEFT, RIGHT, FULL OUTER).
   * SQL constraints (e.g., PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK).
   * Data types in SQL.
2. **Advanced SQL Queries**:
   * Subqueries and Correlated Subqueries.
   * Common Table Expressions (CTEs).
   * Window functions (e.g., ROW\_NUMBER(), RANK(), DENSE\_RANK(), PARTITION BY).
   * Aggregate functions (SUM, COUNT, AVG, etc.) and GROUP BY clause.
   * HAVING vs. WHERE clause.
3. **Database Design and Normalization**:
   * Understanding of Normal Forms (1NF, 2NF, 3NF, BCNF).
   * How to design a normalized database schema.
   * Pros and cons of denormalization.
4. **Indexes and Performance Optimization**:
   * Types of indexes (Clustered vs. Non-Clustered).
   * How indexes work and their impact on query performance.
   * Explain plans and query optimization techniques.
5. **Transactions and Concurrency Control**:
   * ACID properties (Atomicity, Consistency, Isolation, Durability).
   * Types of transaction isolation levels (e.g., Read Uncommitted, Read Committed, Repeatable Read, Serializable).
   * Locking mechanisms and deadlock handling.
6. **SQL Functions and Stored Procedures**:
   * Writing and using SQL functions (Scalar, Aggregate).
   * Creating and using Stored Procedures and Triggers.
   * Differences between Functions and Procedures.
7. **Practical Problem Solving**:
   * Writing SQL queries to solve real-world problems (e.g., finding the second-highest salary, calculating running totals).
   * Debugging and optimizing SQL code.

**Example Interview Questions**

1. **Basic Questions**:
   * What is the difference between JOIN and UNION?
   * How can you retrieve unique records from a table?
2. **Intermediate Questions**:
   * Write a query to find employees who have the second-highest salary.
   * How would you delete duplicate rows in a table?
3. **Advanced Questions**:
   * Explain the concept of window functions with an example.
   * Given a large table, how would you optimize a slow-running query?
4. **Scenario-Based Questions**:
   * How would you design a database for a library management system?
   * Describe a time when you optimized a complex SQL query. What steps did you take?

Would you like more in-depth examples or explanations on any of these topics?