Day 20- Facilitation Guide

Index

For (1.5 hrs) ILT

In the preceding session, we conducted a mock Java interview.During this session, we will evaluate the students' preparedness in Database Management System.

I. Interview Q&A

1. What is a DBMS, and why is it important in the world of data management?

A Database Management System (DBMS) is software that enables users to interact with databases to store, retrieve, update, and manage data. It's crucial in data management because it provides data integrity, security, and efficient data access, ensuring that information is organized, accessible, and reliable.

2.What are the key components of a typical DBMS architecture?

A typical DBMS architecture includes the following components:

- Data Storage: The database itself where data is stored.
- Query Processor: Manages query optimization and execution.
- Transaction Manager: Ensures data consistency through ACID properties.
- Concurrency Control: Handles multiple users accessing data simultaneously.
- Security and Authorization: Controls user access and permissions.
- Backup and Recovery: Manages data backup and restoration.

3. What is normalization in the context of database design, and why is it important?

Normalization is a process in database design where data is organized efficiently to minimize data redundancy and improve data integrity. It's important because it reduces anomalies in data, ensures consistency, and simplifies data maintenance. Common forms of normalization include 1NF, 2NF, and 3NF.

4. Explain the ACID properties in the context of database transactions.

ACID stands for Atomicity, Consistency, Isolation, and Durability, and it defines properties that ensure the reliability of database transactions:

- Atomicity: Transactions are treated as a single, indivisible unit, and either all changes are applied or none.
- Consistency: Transactions bring the database from one consistent state to another.
- Isolation: Transactions are isolated from each other to prevent interference.
- Durability: Once a transaction is committed, its changes are permanent and survive system failures.

5. What is a primary key, and why is it important in a database?

A primary key is a unique identifier for a record in a database table. It ensures that each record is uniquely identifiable and enforces data integrity by preventing duplicate records. The primary key is used for data retrieval and relationship establishment between tables.

6. Can you explain the differences between a relational database and a NoSQL database?

Relational databases are table-based, use structured data, and rely on SQL for querying. NoSQL databases are non-tabular, use various data models (e.g., document, key-value, graph), and are better suited for unstructured or semi-structured data. Each type has its use cases and advantages.

7. What is database normalization, and can you provide an example of the normalization process?

Database normalization is a process of organizing data in a relational database to reduce data redundancy and improve data integrity. For example, let's consider a table that stores customer information. Instead of having all customer data in one table, you can normalize it into two tables: one for customer information and another for orders. This minimizes data duplication and reduces potential inconsistencies.

8. What is a foreign key, and how does it relate to the concept of referential integrity in a database?

A foreign key is a field in a table that is used to establish a link between the data in two tables. It enforces referential integrity by ensuring that the values in the foreign key column of one table correspond to the values in the primary key column of another table. This relationship maintains consistency in the data.

9. What is SQL, and what are its main components?

SQL (Structured Query Language) is a programming language used to manage and manipulate relational databases. Its main components include Data Query Language (DQL), Data Definition Language (DDL), Data Manipulation Language (DML), and Data Control Language (DCL). DQL is used for querying data (e.g., SELECT statements), DDL for defining database structures (e.g., CREATE TABLE), DML for modifying data (e.g., INSERT, UPDATE), and DCL for managing access rights (e.g., GRANT, REVOKE).

10. Explain the difference between INNER JOIN and LEFT JOIN.

INNER JOIN returns only the rows that have matching values in both tables. LEFT JOIN returns all rows from the left table and the matching rows from the right table. If there is no match in the right table, NULL values are returned.

11. What is a subquery in SQL, and how does it differ from a JOIN?

A subquery is a query nested within another query. It is used to retrieve data that will be used in the main query. Subqueries can return a single value, a list of values, or a table. Unlike a JOIN, which combines data from multiple tables into a single result set, a subquery returns a result set that can be used by the main query.

12. Explain the difference between the WHERE clause and HAVING clause in SQL.

The WHERE clause is used to filter rows before grouping (or aggregating) in a query. HAVING, on the other hand, filters rows after grouping and aggregation. It is typically used with GROUP BY to filter aggregated results based on conditions.

13. What is a SQL injection, and how can it be prevented?

SQL injection is a type of security vulnerability in which an attacker can manipulate or inject malicious SQL code into an input field or parameter. It can lead to unauthorized

access or data loss. To prevent SQL injection, use parameterized queries or prepared statements, input validation, and sanitize user inputs. Never directly include user input in SQL queries.

14. What is the purpose of the GROUP BY clause in SQL, and how is it used?

The GROUP BY clause is used to group rows that have the same values in specified columns. It is often used with aggregate functions (e.g., COUNT, SUM, AVG) to perform calculations on groups of data. The result is a summary of data based on the grouped columns.

15. Explain the concept of database normalization and provide an example.

Database normalization is the process of organizing data in a relational database to reduce data redundancy and improve data integrity. For example, consider a table storing customer information. Instead of having all customer data in one table, you can normalize it into two tables: one for customer information and another for orders. This minimizes data duplication and reduces potential inconsistencies.

II. Research and learn about Security

The trainer will allocate case studies to students to investigate SQL security.

Case Study 1: Unauthorized Data Access

***Scenario*:** A financial institution is facing a security breach where sensitive customer data has been accessed by unauthorized users. The company's database contains customer account details, including personal and financial information.

***Challenge*:** Investigate the breach and identify how the unauthorized access occurred. Implement measures to secure the database and prevent future unauthorized data access.