15 Lecture - CS506

Important Subjective

Certainly, here are 10 subjective short questions along with their answers related to advanced concepts in Java Database Connectivity (MoreOnJDBC):

****Question 1: What is connection pooling in JDBC, and why is it important?****

****Answer:** Connection pooling involves reusing and managing a pool of database connections. It's essential for optimizing resource usage, as creating new connections is resource-intensive. Connection pooling improves application performance by efficiently managing available connections for multiple clients.**

****Question 2: Explain the concept of a JDBC transaction.****

****Answer:** A JDBC transaction represents an atomic unit of work on a database. It consists of multiple SQL statements that are executed as a single, cohesive operation. Transactions ensure data integrity and consistency, either committing all changes if successful or rolling back if an error occurs.**

****Question 3: How does the `Connection` interface handle transactions in JDBC?****

****Answer:** The `Connection` interface provides methods like `commit()` and `rollback()` to handle transactions. Transactions can be initiated with `setAutoCommit(false)` and then explicitly committed or rolled back based on the desired outcome of the database operations.**

****Question 4: What are prepared statements, and why are they recommended over regular statements?****

****Answer:** Prepared statements are pre-compiled SQL statements that accept parameters. They are recommended over regular statements for several reasons: they offer better performance due to pre-compilation, prevent SQL injection attacks, and allow efficient execution of parameterized queries.**

Question 5: How does batch processing contribute to improving performance in JDBC?

Answer: Batch processing involves executing multiple SQL statements in a single batch, minimizing the overhead of database round-trips. This improves performance by reducing the network communication and optimizing query execution, especially for scenarios with repetitive operations.

Question 6: Describe the purpose of a stored procedure in JDBC and its benefits.

Answer: A stored procedure is a precompiled database program that can be called from a JDBC application. Benefits include encapsulating complex operations, promoting code reusability, enhancing security by preventing direct SQL exposure, and reducing network traffic by executing on the database server.

Question 7: How do you call a stored procedure using the `CallableStatement` interface in JDBC?

Answer: To call a stored procedure, you create a `CallableStatement` object using a SQL call string with placeholders for input and output parameters. Then you set parameter values using `setXXX()` methods, execute the procedure using `execute()`, and retrieve output parameters using `getXXX()` methods.

Question 8: Explain the role of `ResultSetMetaData` in JDBC.

Answer: `ResultSetMetaData` is an interface that provides metadata about a `ResultSet`. It offers methods to retrieve information about column names, types, and properties of the result set. This metadata helps in dynamically handling various query results.

Question 9: How can you handle exceptions and manage error scenarios in JDBC applications?

Answer: JDBC methods may throw exceptions related to database connectivity, query execution, and more. Proper error handling involves using try-catch blocks to catch exceptions, logging error details, and taking appropriate actions such as rolling back transactions or notifying users.

Question 10: Describe the benefits of using advanced JDBC concepts like connection pooling and transaction management in real-world applications.

Answer: Connection pooling optimizes resource utilization by reusing connections, enhancing application performance. Transaction management ensures data integrity and consistency, allowing safe execution of multiple SQL statements as a single unit of work. These concepts collectively improve application efficiency, scalability, and maintainability.