Assignment - 8 (Java)

Q1.What is ORM in Hibernate?

Answer:ORM (Object-Relational Mapping) in Hibernate is a technique that allows developers to map Java objects to relational database tables. It eliminates the need for writing complex SQL queries by providing a layer of abstraction. Hibernate automatically handles the mapping between object-oriented concepts and relational database concepts, simplifying database operations. It provides features like automatic data persistence, caching, and transaction management. Developers can work with Java objects directly, while Hibernate takes care of translating the data to and from the underlying database tables.

Q2.What are the advantages of Hibernate over JDBC?

Answer: Hibernate offers several advantages over JDBC:

1. Simplified Data Access: Hibernate eliminates the need for writing repetitive JDBC code, reducing development time and effort.
2. Object-Relational Mapping: Hibernate maps Java objects to database tables, providing a seamless integration between the object-oriented and relational paradigms.
3. Database Independence: Hibernate supports multiple databases, allowing developers to switch databases without changing the code.
4. Caching and Performance: Hibernate provides caching mechanisms, improving application performance by reducing database hits.
5. Transparent Persistence: Hibernate automatically manages the persistence of objects, simplifying CRUD operations and ensuring data integrity.

Q3.What are some of the important interfaces of Hibernate framework?

Answer: Some important interfaces in the Hibernate framework are:

1. Session: It represents a single-threaded unit of work and acts as a factory for creating database operations and managing transactions.
2. SessionFactory: It is responsible for creating and managing sessions. It is typically instantiated once per application and provides a cache of compiled mappings.
3. Transaction: It represents a unit of work that is performed within a session. It provides methods for transaction management, such as commit and rollback.
4. Query: It allows the execution of HQL (Hibernate Query Language) or SQL queries and provides methods for retrieving query results.
5. Criteria: It provides a programmatic way to build type-safe queries using criteria-based restrictions and projections.

Q4.What is a Session in Hibernate?

Answer: In Hibernate, a Session represents a single-threaded unit of work between the application and the database. It acts as a factory for creating database operations and manages the interaction with the database. The Session is obtained from the SessionFactory and serves as a persistent context for entities. It provides methods to save, update, delete, and retrieve objects from the database. The Session also manages transaction boundaries and provides caching mechanisms for improved performance. It represents a logical connection to the database and encapsulates the underlying JDBC connections and transactions.

Q5.What is a SessionFactory?

Answer: In Hibernate, a SessionFactory is a thread-safe factory for creating Session objects. It is responsible for initializing and configuring Hibernate, as well as providing a cache of compiled mappings for efficient object-to-relational mapping. The SessionFactory is typically instantiated once during application startup and shared across the application. It is built based on the configuration settings and metadata mappings provided by the application. The SessionFactory represents the database connection and provides a Session whenever needed, allowing efficient management of database interactions and transactions.

Q6.What is HQL?

Answer:HQL (Hibernate Query Language) is a powerful object-oriented query language provided by Hibernate. It is similar to SQL but operates on persistent objects and their properties rather than database tables and columns. HQL allows developers to write database-independent queries, making it easier to switch between different database systems. It supports various query operations such as filtering, sorting, joining, and aggregating data. HQL queries are translated by Hibernate into SQL queries, enabling efficient interaction with the underlying database while working with object-oriented concepts.

Q7.What are Many to Many associations?

Answer: Many-to-many associations refer to a relationship between two entities where multiple instances of one entity can be associated with multiple instances of another entity. It involves the use of a join table to establish the association between the two entities. This type of association is commonly used when there is a need for a many-to-many relationship, such as students enrolling in multiple courses and courses having multiple students. In Hibernate, many-to-many associations can be mapped using appropriate annotations or XML configurations to define the relationship between entities.

Q8.What is hibernate caching?

Answer: Hibernate caching refers to the mechanism of storing frequently accessed data in memory to improve application performance. Hibernate provides several levels of caching, including the first-level cache (session cache) and second-level cache.

The first-level cache is associated with a session and stores the objects retrieved from the database during a session. It ensures that subsequent requests for the same objects are served from memory, reducing database hits.

The second-level cache is shared among sessions and stores objects across sessions, providing a global cache for improved performance and scalability. It can be configured to use various cache providers like Ehcache or Infinispan.

Q9.What is the difference between first level cache and second level cache?
Answer: The first-level cache (session cache) in Hibernate is associated with a specific Hibernate session. It is enabled by default and stores objects retrieved from the database during the session. It ensures that subsequent requests for the same objects are served from memory, reducing database hits within a single session.

On the other hand, the second-level cache is a shared cache that spans across multiple sessions. It stores objects across sessions, providing a global cache for improved performance and scalability. It reduces the number of database hits and enhances overall application performance by caching data at a higher level.

Q10.What can you tell about Hibernate Configuration File?

Answer: The Hibernate configuration file, typically named `hibernate.cfg.xml`, is an essential file in Hibernate. It contains configuration settings required to establish the database connection, define mapping metadata, and configure various Hibernate properties. It specifies the database connection details, including the driver class, URL, username, and password. Additionally, it can define the mapping resources, dialect, cache settings, and other Hibernate-specific configurations. The configuration file is read during Hibernate initialization and serves as a crucial configuration source for the Hibernate framework to function properly.