HIBERNATE

1. What is Hibernate Framework?

* Hibernate is an open-source and lightweight ORM tool that is used to store, manipulate, and retrieve data from the database.

Object-relational mapping or ORM is the programming technique to map application domain model objects to the relational database tables. Hibernate is a Java-based ORM tool that provides a framework for mapping application domain objects to the relational database tables and vice versa.

Hibernate provides a reference implementation of Java Persistence API, which makes it a great choice as an ORM tool with the benefits of loose coupling. We can use the Hibernate persistence API for CRUD operations. Hibernate framework provides an option to map plain old java objects to traditional database tables with the use of JPA annotations as well as XML-based configuration.

Similarly, hibernate configurations are flexible and can be done from an XML configuration file as well as programmatically.

1. What are the important benefits of using Hibernate Framework?

Some of the important benefits of using hibernate framework are:

* Hibernate eliminates all the boiler-plate code that comes with JDBC and takes care of managing resources, so we can focus on business logic.
* Hibernate framework provides support for XML as well as JPA annotations, which makes our code implementation independent.
* Hibernate provides a powerful query language (HQL) that is similar to SQL. However, HQL is fully object-oriented and understands concepts like inheritance, polymorphism, and association.
* Hibernate is an open-source project from Red Hat Community and is used worldwide. This makes it a better choice than others because the learning curve is small and there are tons of online documentation and help easily available in forums.
* Hibernate is easy to integrate with other Java EE frameworks, it’s so popular that Spring Framework provides built-in support for integrating hibernate with Spring applications.
* Hibernate supports lazy initialization using proxy objects and performs actual database queries only when it’s required.
* Hibernate cache helps us in getting better performance.
* For database vendor-specific features, hibernate is suitable because we can also execute native SQL queries.

Overall hibernate is the best choice in the current market for ORM tools, it contains all the features that you will ever need in an ORM tool.

1. Explain Hibernate architecture.

Hibernate has a layered architecture that helps the user to operate without having to know the underlying APIs. Hibernate makes use of the database and configuration data to provide persistence services (and persistent objects) to the application. It includes many objects such as a persistent object, session factory, transaction factory, connection factory, session, transaction, etc.

The Hibernate architecture is categorized into four layers.

* Java application layer
* Hibernate framework layer
* Backhand API layer
* Database layer

1. Get() method:

* Returns null if the object is not found.
* get() method always hit the database.
* It returns a real object, not a proxy.
* It should be used if you are not sure about the existence of an instance.

1. Load() method:

* Throws ObjectNotFoundException if an object is not found.
* load() method doesn’t hit the database.
* It returns a proxy object.
* It should be used if you are sure that the instance exists.

1. What is ORM in Hibernate?

Hibernate ORM stands for Object Relational Mapping. This is a mapping tool pattern mainly used for converting data stored in a relational database to an object used in object-oriented programming constructs. This tool also helps greatly in simplifying data retrieval, creation, and manipulation.

1. What are the advantages of Hibernate over JDBC?

- The advantages of Hibernate over JDBC are listed below:

* Application development is fast.
* Management of transactions.
* Generates key automatically.
* Details of SQL queries are hidden.

1. What is a Session in Hibernate?

A session is an object that maintains the connection between the Java object application and the database. Session also has methods for storing, retrieving, modifying or deleting data from database using methods like persist(), load(), get(), update(), delete(), etc. Additionally, It has factory methods to return Query, Criteria, and Transaction objects.

1. What is a SessionFactory?

SessionFactory provides an instance of Session. It is a factory class that gives the Session objects based on the configuration parameters to establish the connection to the database.

As a good practice, the application generally has a single instance of SessionFactory. The internal state of a SessionFactory which includes metadata about ORM is immutable, i.e once the instance is created, it cannot be changed.

This also provides the facility to get information like statistics and metadata related to a class, query executions, etc. It also holds second-level cache data if enabled.

1. What can you tell me about Hibernate Configuration File?

* hibernate.cfg.xml
* hibernate.properties
* hibernate.cfg.yml

1. Is hibernate prone to SQL injection attacks?

SQL injection attack is a serious vulnerability in terms of web security wherein an attacker can interfere with the queries made by an application/website to its database thereby allowing the attacker to view sensitive data which are generally irretrievable. It can also give the attacker to modify/ remove the data resulting in damages to the application behavior.

Hibernate does not provide immunity to SQL Injection. However, following good practices avoids SQL injection attacks. It is always advisable to follow any of the below options:

* Incorporate Prepared Statements that use Parameterized Queries.
* Use Stored Procedures.
* Ensure data sanity by doing input validation

1. What are the criteria for API in hibernate?

Criteria API in Hibernate helps developers to build dynamic criteria queries on the persistence database. Criteria API is a more powerful and flexible alternative to HQL (Hibernate Query Language) queries for creating dynamic queries.

This API allows for the programmatic development of criteria query objects. The org. hibernate. The criteria interface is used for these purposes. The Session interface of hibernate framework has the createCriteria() method that takes the persistent object’s class or its entity name as the parameters and returns the persistence object instance the criteria query is executed.

It also makes it very easy to incorporate restrictions to selectively retrieve data from the database. It can be achieved by using the add() method which accepts the org.hibernate.criterion.Criterion object represents individual restriction.

1. What is HQL?

Hibernate Query Language (HQL) is used as an extension of SQL. It is very simple, efficient, and very flexible for performing complex operations on relational databases without writing complicated queries. HQL is the object-oriented representation of query language, i.e instead of using table name, we make use of the class name which makes this language independent of any database.

1. Does Hibernate support Native SQL Queries?

Yes, it does. Hibernate provides the create SQL query() method to let a developer call the native SQL statement directly and return a Query object.

1. What is hibernate caching?

Hibernate caching is the strategy for improving the application performance by pooling objects in the cache so that the queries are executed faster. Hibernate caching is particularly useful when fetching the same data that is executed multiple times. Rather than hitting the database, we can just access the data from the cache. This results in reduced throughput time of the application.

First Level Cache:

* This level is enabled by default.
* The first level cache resides in the hibernate session object.
* Since it belongs to the session object, the scope of the data stored here will not be available to the entire application as an application can make use of multiple session objects.

Second Level Cache:

* Second level cache resides in the SessionFactory object and due to this, the data is accessible by the entire application.
* This is not available by default. It has to be enabled explicitly.
* EH (Easy Hibernate) Cache, Swarm Cache, OS Cache, and JBoss Cache are some examples of cache providers.

1. Can you tell me something about Named SQL Query?  
   A named SQL query is an expression represented in the form of a table. Here, SQL expressions to select/retrieve rows and columns from one or more tables in one or more databases can be specified. This is like using aliases for the queries.

In Hibernate, we can make use of @NameQueries and @NameQuery annotations:

* @NameQueries annotation is used for defining multiple named queries.
* @NameQuery annotation is used for defining a single named query.

1. What are the benefits of NamedQuery?

To understand the benefits of NamedQuery, let's first understand the disadvantage of HQL and SQL. The main disadvantage of having HQL and SQL scattered across data access objects is that it makes the code unreadable. Hence, as good practice, it is recommended to group all HQL and SQL codes in one place and use only their reference in the actual data access code. To achieve this, Hibernate gives us named queries.

A named query is a statically defined query with a predefined unchangeable query string. They are validated when the session factory is created, thus making the application fail fast in case of an error.

1. What are the core interfaces of Hibernate?

The core interfaces of Hibernate framework are:

* Configuration
* SessionFactory
* Session
* Query
* Criteria
* Transaction

1. List some of the databases supported by Hibernate.

Some of the databases supported by Hibernate are:

* DB2
* MySQL
* Oracle
* Sybase SQL Server
* Informix Dynamic Server
* HSQL
* PostgreSQL
* FrontBase