Hibernate Interview Questions

1) What is Hibernate?

Hibernate is a powerful, high performance object/relational persistence and query service. This lets the users to develop persistent classes following object-oriented principles such as association, inheritance, polymorphism, composition, and collections.

2) What is ORM?

ORM stands for Object/Relational mapping. It is the programmed and translucent perseverance of objects in a Java application in to the tables of a relational database using the metadata that describes the mapping between the objects and the database. It works by transforming the data from one representation to another.

3) What does an ORM solution comprises of?

- It should have an API for performing basic CRUD (Create, Read, Update, Delete) operations on objects of persistent classes
- Should have a language or an API for specifying queries that refer to the classes and the properties of classes
- An ability for specifying mapping metadata
- It should have a technique for ORM implementation to interact with transactional objects to perform dirty checking, lazy association fetching, and other optimization functions

4) What are the different levels of ORM quality?

There are four levels defined for ORM quality.

i. Pure relational
ii. Light object mapping
iii. Medium object mapping
iv. Full object mapping

5) What is a pure relational ORM?

The entire application, including the user interface, is designed around the relational model and SQL-based relational operations.

6) What is a meant by light object mapping?

The entities are represented as classes that are mapped manually to the relational tables. The code is hidden from the business logic using specific design patterns. This approach is successful for applications with a less number of entities, or applications with common, metadata-driven data models. This approach is most known to all.

7) What is a meant by medium object mapping?

The application is designed around an object model. The SQL code is generated at build time. And the associations between objects are supported by the persistence mechanism, and queries are specified using an object-oriented expression language. This is best suited for medium-sized applications with some complex transactions. Used when the mapping exceeds 25 different database products at a time.

8) What is meant by full object mapping?

Full object mapping supports sophisticated object modeling: composition, inheritance, polymorphism and persistence. The persistence layer implements transparent persistence; persistent classes do not inherit any special base class or have to implement a special interface. Efficient fetching strategies and caching strategies are implemented transparently to the application.

9) What are the benefits of ORM and Hibernate?

There are many benefits from these. Out of which the following are the most important one.

i. **Productivity –** Hibernate reduces the burden of developer by providing much of the functionality and let the developer to concentrate on business logic. **Maintainability –** As hibernate provides most of the functionality, the LOC for the application will be reduced and it is easy to maintain. By automated object/ relational persistence it even reduces the LOC. **Performance –** Hand-coded persistence provided greater performance than automated one. But this is not true all the times. But in hibernate, it provides more optimization that works all the time there by increasing the performance. If it is automated persistence then it still increases the performance. **Vendor independence –** Irrespective of the different types of databases that are there, hibernate provides a much easier way to develop a cross platform application.

10) How does hibernate code looks like?

```
Session session = getSessionFactory().openSession();
Transaction tx = session.beginTransaction();
MyPersistanceClass mpc = new MyPersistanceClass ("Sample App");
session.save(mpc);
tx.commit();
session.close();
```

The Session and Transaction are the interfaces provided by hibernate. There are many other interfaces besides this.

11) What is a hibernate xml mapping document and how does it look like?

In order to make most of the things work in hibernate, usually the information is provided in an xml document. This document is called as xml mapping document. The document

defines, among other things, how properties of the user defined persistence classes' map to the columns of the relative tables in database.

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "http://hibernate.sourceforge.net/hibernate-map

<hibernate-mapping>
   <class name="sample.MyPersistanceClass" table="MyPersitaceTable">
      <id name="id" column="MyPerId">
        <generator class="increment"/>
      </id>
      <property name="text" column="Persistance_message"/>
      <many-to-one name="nxtPer" cascade="all" column="NxtPerId"/>
   </class>
</hibernate-mapping>
```

Everything should be included under tag. This is the main tag for an xml mapping document.

12) Show Hibernate overview?

13) What the Core interfaces are of hibernate framework?

There are many benefits from these. Out of which the following are the most important one.

i. **Session Interface –** This is the primary interface used by hibernate applications. The instances of this interface are lightweight and are inexpensive to create and destroy. Hibernate sessions are not thread safe.
ii. **SessionFactory Interface –** This is a factory that delivers the session objects to hibernate application. Generally there will be a single SessionFactory for the whole application and it will be shared among all the application threads.
iii. **Configuration Interface –** This interface is used to configure and bootstrap hibernate. The instance of this interface is used by the application in order to specify the location of hibernate specific mapping documents.
iv. **Transaction Interface –** This is an optional interface but the above three interfaces are mandatory in each and every application. This interface abstracts the code from any kind of transaction implementations such as JDBC transaction, JTA transaction.
v. **Query and Criteria Interface –** This interface allows the user to perform queries and also control the flow of the query execution.

14) What are Callback interfaces?

These interfaces are used in the application to receive a notification when some object events occur. Like when an object is loaded, saved or deleted. There is no need to implement callbacks in hibernate applications, but they're useful for implementing certain kinds of generic functionality.

15) What are Extension interfaces?

When the built-in functionalities provided by hibernate is not sufficient enough, it provides a way so that user can include other interfaces and implement those interfaces for user desire functionality. These interfaces are called as Extension interfaces.

16) What are the Extension interfaces that are there in hibernate?

There are many extension interfaces provided by hibernate.

- **ProxyFactory** interface - used to create proxies
- **ConnectionProvider** interface – used for JDBC connection management
- **TransactionFactory** interface – Used for transaction management
- **Transaction** interface – Used for transaction management
- **TransactionManagementLookup** interface – Used in transaction management.
- **Cahce** interface – provides caching techniques and strategies
- **CacheProvider** interface – same as Cache interface
- **ClassPersister** interface – provides ORM strategies
- **IdentifierGenerator** interface – used for primary key generation
- **Dialect** abstract class – provides SQL support

17) What are different environments to configure hibernate?

There are mainly two types of environments in which the configuration of hibernate application differs.

i. **Managed environment –** In this kind of environment everything from database connections, transaction boundaries, security levels and all are defined. An example of this kind of environment is environment provided by application servers such as JBoss, Weblogic and WebSphere.
ii. **Non-managed environment –** This kind of environment provides a basic configuration template. Tomcat is one of the best examples that provide this kind of environment.

18) What is the file extension you use for hibernate mapping file?

The name of the file should be like this : filenam**.hbm.xml**
The filename varies here. The extension of these files should be ".hbm.xml".
This is just a convention and it's not mandatory. But this is the best practice to follow this extension.

19) What do you create a SessionFactory?

```
Configuration cfg = new Configuration();
cfg.addResource("myinstance/MyConfig.hbm.xml");
cfg.setProperties( System.getProperties() );
SessionFactory sessions = cfg.buildSessionFactory();
```

First, we need to create an instance of Configuration and use that instance to refer to the location of the configuration file. After configuring this instance is used to create the SessionFactory by calling the method buildSessionFactory().

20) What is meant by Method chaining?

Method chaining is a programming technique that is supported by many hibernate interfaces. This is less readable when compared to actual java code. And it is not mandatory to use this format. Look how a SessionFactory is created when we use method chaining.

```
SessionFactory sessions = new Configuration()
    .addResource("myinstance/MyConfig.hbm.xml")
    .setProperties( System.getProperties() )
    .buildSessionFactory();
```

21) What does hibernate.properties file consist of?

This is a property file that should be placed in application class path. So when the Configuration object is created, hibernate is first initialized. At this moment the application will automatically detect and read this hibernate.properties file.

```
hibernate.connection.datasource = java:/comp/env/jdbc/AuctionDB
hibernate.transaction.factory_class = net.sf.hibernate.transaction.JTATransactionF
hibernate.transaction.manager_lookup_class = net.sf.hibernate.transaction.JBossTra
hibernate.dialect = net.sf.hibernate.dialect.PostgreSQLDialect
```

22) What should SessionFactory be placed so that it can be easily accessed?

As far as it is compared to J2EE environment, if the SessionFactory is placed in JNDI then it can be easily accessed and shared between different threads and various components that are hibernate aware. You can set the SessionFactory to a JNDI by configuring a property **hibernate.session_factory_name** in the hibernate.properties file.

23) What are POJOs?

POJO stands for plain old java objects. These are just basic JavaBeans that have defined setter and getter methods for all the properties that are there in that bean. Besides they can also have some business logic related to that property. Hibernate applications works efficiently with POJOs rather then simple java classes.

24) What is object/relational mapping metadata?

ORM tools require a metadata format for the application to specify the mapping between classes and tables, properties and columns, associations and foreign keys, Java types and SQL types. This information is called the object/relational mapping metadata. It defines the transformation between the different data type systems and relationship representations.

25) What is HQL?

HQL stands for Hibernate Query Language. Hibernate allows the user to express queries in its own portable SQL extension and this is called as HQL. It also allows the user to express in native SQL.

26) What are the different types of property and class mappings?

- Typical and most common property mapping

```
<property name="description" column="DESCRIPTION" type="string"/>
Or
<property name="description" type="string">
<column name="DESCRIPTION"/>
</property>
```

- Derived properties

```
<property name="averageBidAmount" formula="( select AVG(b.AMOUNT) from BID k
```

- Typical and most common property mapping

```
<property name="description" column="DESCRIPTION" type="string"/>
```

- Controlling inserts and updates

```
<property name="name" column="NAME" type="string" insert="false" update="fal
```

27) What is Attribute Oriented Programming?

XDoclet has brought the concept of attribute-oriented programming to Java. Until JDK 1.5, the Java language had no support for annotations; now XDoclet uses the Javadoc tag format (@attribute) to specify class-, field-, or method-level metadata attributes. These attributes are used to generate hibernate mapping file automatically when the application is built. This kind of programming that works on attributes is called as Attribute Oriented Programming.

28) What are the different methods of identifying an object?

There are three methods by which an object can be identified.

i. **Object identity –** Objects are identical if they reside in the same memory location in the JVM. This can be checked by using the = = operator.
ii. **Object equality –** Objects are equal if they have the same value, as defined by the equals( ) method. Classes that don't explicitly override this method inherit the implementation defined by java.lang.Object, which compares object identity.
iii. **Database identity –** Objects stored in a relational database are identical if they represent the same row or, equivalently, share the same table and primary key value.

29) What are the different approaches to represent an inheritance hierarchy?

i. Table per concrete class.
ii. Table per class hierarchy.
iii. Table per subclass.

30) What are managed associations and hibernate associations?

Associations that are related to container management persistence are called managed associations. These are bi-directional associations. Coming to hibernate associations, these are unidirectional.