**Hibernate Q&A**

1. What is ORM ?

2. What is Hibernate?

3. What is the advantage of Hibernate over jdbc?

**4. How do you implemente Hibernate in the application development ?**

5. What are the important tags of hibernate.cfg.xml?

6. What are the Core interfaces are of Hibernate framework?

7. SessionFactory vs Session

8. What is Hibernate Query Language (HQL)?

9. How do you map Java Objects with Database tables?

10. What is the difference between load() and get()?

11. What are different types of fetch ?

12. What is lazy loading ?

13. Difference between fetch type and fetch mode ?

14. What are different fetch modes ?

15. What is the difference between and merge and update ?

16. What are the different states of the Hibernate obejects ?

17. Define cascade and inverse option in one-many mapping?

**18. What are different Collection types in Hibernate ?**

19. What do you mean by fetching strategy ?

20. What are different inheritance strategies in Hibernate?

**21. What are the different associations?**

22. What is first level and second level cache in Hibernate ?

**23. Have implemented any second level caching in Hibernate ?**

24. What is Query Cache ?

25. What is Criteria API ?

**26. What is bidirectional in Hibernate associations ?**

27. What is automatic dirty checking?

Hibernate is an open source, lightweight, [ORM (Object Relational Mapping)](http://en.wikipedia.org/wiki/Object-relational_mapping) tool.

1. **What is ORM ?**

**An ORM (Object Relational Mapper) is a piece/layer of software that helps map your code Objects to your database.**

An ORM tool simplifies the data creation, data manipulation and data access. It is a programming technique that maps the object to the data stored in the database.



Advantages of Hibernate Framework

**Opensource and Lightweight:**

**Database Independent query:** HQL (Hibernate Query Language) is the object-oriented version of SQL. It generates the database independent queries. So you don't need to write database specific queries.

**Automatic table creation:** Hibernate framework provides the facility to create the tables of the database automatically. So there is no need to create tables in the database manually.

**Simplifies complex join:** To fetch data form multiple tables is easy in hibernate framework.

1. What is Hibernate?

Hibernate is an open source, lightweight, [ORM (Object Relational Mapping)](http://en.wikipedia.org/wiki/Object-relational_mapping) tool.

Hibernate framework simplifies the development of java application to interact with the database.

1. What is the advantage of Hibernate over jdbc?

1.Hibernate is data base independent, same code will work for all data bases like ORACLE,MySQL ,SQLServer etc.

In case of JDBC query must be data base specific.

2)As Hibernate is set of Objects , you don't need to learn SQL language.

You can treat TABLE as a Object .

In case of JDBC you need to learn SQL.

3) Development fast in case of Hibernate because you don't need to write queries.

4) You can load your objects on start up using lazy=false in case of Hibernate.

JDBC Don't have such support.

5) You will get benefit of Cache. Hibernate support two level of cache. First level and 2nd level. So you can store your data into Cache for better performance.

In case of JDBC you need to implement your java cache .

1. How do you implemente Hibernate in the application development ?
2. What are the important tags of hibernate.cfg.xml?

This file can be used to provide the database information like driverclass name, url, database usename, database password, dialect, connection pooling mapping file and so on

Ex:

<hibernate-configuration>

<session-factory>

<property name=*"hbm2ddl.auto"*>update</property>

<property name=*"dialect"*>org.hibernate.dialect.MySQLDialect</property>

<property name=*"connection.url"*>jdbc:mysql://localhost:3306/poc</property>

<property name=*"connection.username"*>root</property>

<property name=*"connection.password"*>system</property>

<property name=*"connection.driver\_class"*>com.mysql.jdbc.Driver</property>

<property name=*"hibernate.current\_session\_context\_class"*>thread</property>

<!-- <property name="connection.pool\_size">2</property> -->

<mapping resource=*"user.hbm.xml"*/>

</session-factory>

</hibernate-configuration>

1. What are the Core interfaces are of Hibernate framework?

The five core interfaces are used in just about every Hibernate application. Using these interfaces, you can store and retrieve persistent objects and control transactions.

* Session interface
* SessionFactory interface
* Configuration interface
* Transaction interface
* Query and Criteria interfaces

Ex;

//creating configuration object

Configuration cfg=**new** Configuration();

cfg.configure("hibernate.cfg.xml");//populates the data of the configuration file

//creating seession factory object

SessionFactory factory=cfg.buildSessionFactory();

//creating session object

Session session=factory.openSession();

//creating transaction object

Transaction t=session.beginTransaction();

Employee e1=**new** Employee();

e1.setId(157);

e1.setFirstName("s");

e1.setLastName("jaiswal");

session.persist(e1);//persisting the object

t.commit();//transaction is commited

session.close();

1. SessionFactory vs Session

* SessionFactory objects are one per application and Session objects are one per client.
* SessionFactory is to create and manage Sessions. Session is to provide a CRUD interface for mapped classes, and also access to the more versitile Criteria API.
* SessionFactory is thread safe where as Session is not thread safe

8. What is Hibernate Query Language (HQL)?

Hibernate offers a query language that embodies a very powerful and flexible mechanism to query, store, update, and retrieve objects from a database. This language, the Hibernate query Language (HQL), is an object-oriented extension to SQL.

9.How do you map Java Objects with Database tables?

* First we need to write Java domain objects (beans with setter and getter).
* Write hbm.xml, where we map java class to table and database columns to Java class variables.

**Example** :

<hibernate-mapping>  
  <class name="com.test.User"  table="user">  
   <property  column="USER\_NAME" length="255"   
      name="userName" not-null="true"  type="java.lang.String"/>  
   <property  column="USER\_PASSWORD" length="255"  
 name="userPassword" not-null="true"  type="java.lang.String"/>  
 </class>  
</hibernate-mapping>

10.load() vs. get() :-

|  |  |
| --- | --- |
| **load()** | **get()** |
| Only use the load() method if you are sure that the object exists. | If you are not sure that the object exists, then use one of the get()methods. |
| load() method will throw an exception if the unique id is not found in the database. | get() method will return null if the unique id is not found in the database. |
| load() just returns a proxy by default and database wonï¿½t be hit until the proxy is first invoked. | get() will hit the database immediately. |

11. What are different types of fetch ?

Hibernate uses a fetching strategy to retrieve associated objects if the application needs to navigate the association. Fetch strategies can be declared in the O/R mapping metadata, or over-ridden by a particular HQL or Criteria query.

Hibernate3 defines the following fetching strategies:

**Join fetching**: Hibernate retrieves the associated instance or collection in the same SELECT, using an OUTER JOIN.

**Select fetching:** a second SELECT is used to retrieve the associated entity or collection. Unless you explicitly disable lazy fetching by specifying lazy="false", this second select will only be executed when you access the association.

**Subselect fetching:** a second SELECT is used to retrieve the associated collections for all entities retrieved in a previous query or fetch. Unless you explicitly disable lazy fetching by specifying lazy="false", this second select will only be executed when you access the association.

**Batch fetching:** an optimization strategy for select fetching. Hibernate retrieves a batch of entity instances or collections in a single SELECT by specifying a list of primary or foreign keys.

<https://www.mkyong.com/hibernate/hibernate-fetching-strategies-examples/>

## **Fetching Strategies**

There are four fetching strategies

1. fetch-“join” = Disable the lazy loading, always load all the collections and entities.  
2. fetch-“select” (default) = Lazy load all the collections and entities.  
3. batch-size=”N” = Fetching up to ‘N’ collections or entities, \*Not record\*.  
4. fetch-“subselect” = Group its collection into a sub select statement.

12. What is lazy loading ?

**Lazy loading** is a design pattern commonly used in computer programming to defer initialization of an object until the point at which it is needed. It can contribute to efficiency in the program's operation if properly and appropriately used. The opposite of **lazy loading** is eager **loading**.

In hiberante

**Lazy loading /fetching** decides whether to **load** child objects while **loading** the Parent Object. You need to do this setting respective **hibernate** mapping file of the parent class.**Lazy** = true (means not to **load** child) By default the **lazy loading** of the child objects is true.

13. Difference between fetch type and fetch mode ?

Fetch type (lazy/eager) tells when Hibernate fetchs the association, whether in advance when it fetches the entity (eager), or whether it waits for the code to ask for the association (lazy).  
  
Fetch mode (select/join) refers to how Hibernate will fetch the association, i.e. does it use an extra SELECT statement, or does it use a join.

14. What are different fetch modes ?/samse as 11

15. What is the difference between and merge and update ?

The both update() and merge() methods are used to change the objects states in hibernate. We can call update() and merge() methods to transfer an object from detached state to persistent state.

A detached state object can be made persist by reattaching to a session. If previous session is closed then we can open a new session and we can reattach to a new session. To reattach we can use update() or merge() methods. Both are doing the same functionality, but there are few differences internally.

**but the major difference between update and merge is that update method cannot be used when the same object exists in the session.**

16. What are the different states of the Hibernate obejects ?

**Transient:**

One newly created object,with out having any relation with the database, means never persistent, not associated with any Session object

**Persistent:**

Having the relation with the database, associated with a unique Session object

**Detached:**

previously having relation with the database [persistent ], now not associated with any Session

see the next sessions for the better understanding of the life cycle states of pojo class object(s) the hibernate

17. Define cascade and inverse option in one-many mapping?

1. inverse: This is used to decide which side is the relationship owner to manage the relationship (insert or update of the foreign key column).

2. cascade: In cascade, after one operation (save, update and delete) is done, it decide whether it need to call other operations (save, update and delete) on another entities which has relationship with each other.

Conclusion: In short, the “inverse” is decide which side will update the foreign key, while “cascade” is decide what’s the follow by operation should execute. Both are look quite similar in relationship, but it’s totally two different things. Hibernate developers are worth to spend time to research on it, because misunderstand the concept or misuse it will bring serious performance or data integrity issue in your application.

18. What are different Collection types in Hibernate ?

19. What do you mean by fetching strategy ?

<https://dzone.com/articles/hibernate-facts-importance>

20. What are different inheritance strategies in Hibernate?

a)table per hierarchy

2.table per concrete

3.table per subclass

**21. What are the different associations?**

22. What is first level and second level cache in Hibernate ?

**The first-level cache:**The first level cache type is the **session cache**. The session cache caches object within the current session but this is not enough for long level i.e. session factory scope.

**The second-level cache:** The second-level cache is called 'second-level' because there is already a cache operating for you in Hibernate for the duration you have a session open. A Hibernate Session is a transaction-level cache of persistent data. It is possible to configure a SessionFactory-level cache on a class-by-class and collection-by-collection basis.

<http://www.dineshonjava.com/p/cacheing-in-hibernate-first-level-and.html>

**23. Have implemented any second level caching in Hibernate ?**

24. What is Query Cache ?

The Hibernate second level cache is an application level cache for storing entity data. The query cache is a separate cache that stores query results only. ... When well used these caches provide improved performance in a transparent way, by reducing the number of SQL statements that hit the database.

25. What is Criteria API ?

**Hibernate** provides alternate ways of manipulating objects and in turn data available in RDBMS tables. ... The **Hibernate** Session interface provides createCriteria() method which can be used to create a **Criteria** object that returns instances of the persistence object's class when your application executes a **criteria query**

**26. What is bidirectional in Hibernate associations ?**

27. What is automatic dirty checking?

**Dirty checking concept :**

Dirty checking is a concept to avoid time consuming database write actions. By this concept, all necessary updating and changes are done without affecting the other fields.  
Only the changed fields of database are updated and the remaining unchanged fields are left untouched.

**Hibernate Dirty Checking :**

Hibernate allows dirty checking feature. It saves developer's time and effort in updating of database when states of objects are modified inside a transaction.   
Hibernate automatically detects the object states whenever changed and synchronized with the database in order to update.