Hibernate Interview Questions

Hibernate interview questions are asked to the students because it is a widely used ORM tool. The important list of top 20 hibernate interview questions and answers for freshers and professionals are given below.

1) What is hibernate?

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

more details...

2) What is ORM?

ORM is an acronym for Object/Relational mapping. It is a programming strategy to map object with the data stored in the database. It simplifies data creation, data manipulation, and data access.

3) Explain hibernate architecture?

Hibernate architecture comprises of many interfaces such as Configuration, SessionFactory, Session, Transaction, etc.

more
details…

Core objects of Hibernate Framework   Internal API
Used by
Hibernate
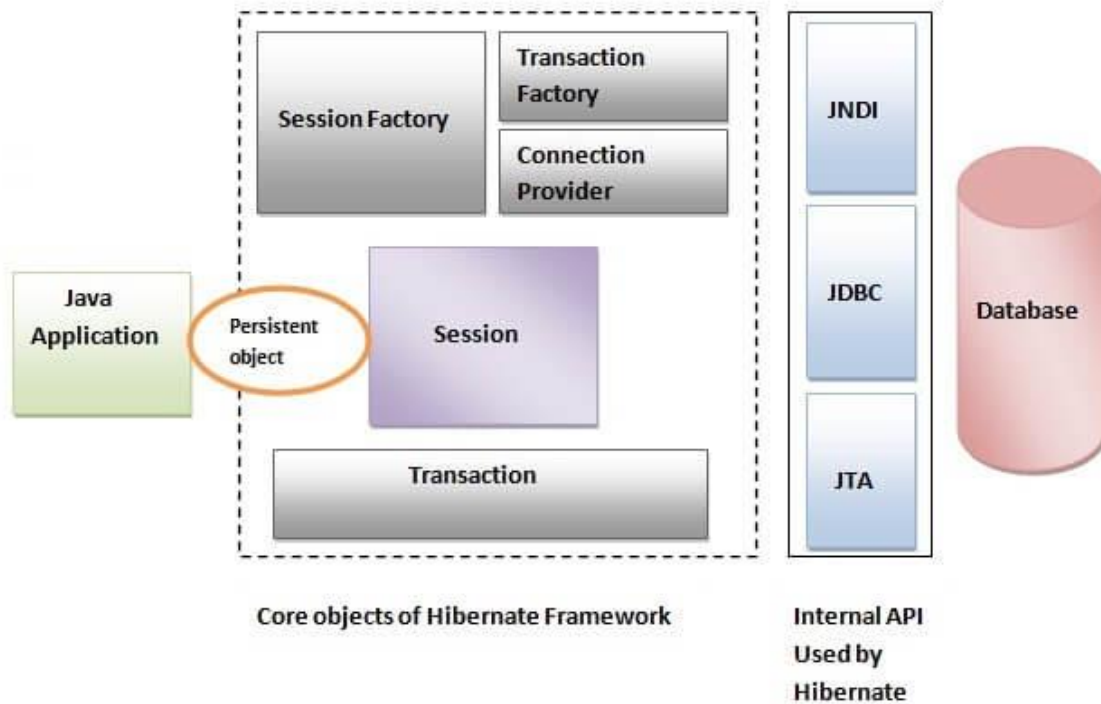
4) What are the core interfaces of Hibernate?

The core interfaces of Hibernate framework are:

- o Configuration
- o SessionFactory
- o Session
- o Query
- o Criteria
- o Transaction

5) Mention some of the advantages of using ORM over JDBC.

ORM has the following advantages over JDBC:

- o Application development is fast.
- o Management of transaction.

o   Generates key automatically.

o   Details of SQL queries are hidden.

6) Define criteria in terms of Hibernate.

The objects of criteria are used for the creation and execution of the object-oriented criteria queries.

7) List some of the databases supported by Hibernate.

Some of the databases supported by Hibernate are:

o   DB2

o   MySQL

o   Oracle

o   Sybase SQL Server

o   Informix Dynamic Server

o   HSQL

o   PostgreSQL

o   FrontBase

8) List the key components of Hibernate.

Key components of Hibernate are:

o   Configuration

o   Session

o   SessionFactory

o   Criteria

o   Query

o   Transaction

9) Mention two components of Hibernate configuration object.

Database Connection

Class Mapping Setup

10) How is SQL query created in Hibernate?

The SQL query is created with the help of the following syntax:

Session.createSQLQuery

11) What does HQL stand for?

Hibernate Query Language

12) How is HQL query created?

The HQL query is created with the help of the following syntax:

Session.createQuery

13) How can we add criteria to a SQL query?

A criterion is added to a SQL query by using the Session.createCriteria.

14) Define persistent classes.

Classes whose objects are stored in a database table are called as persistent classes.

15) What is SessionFactory?

SessionFactory provides the instance of Session. It is a factory of Session. It holds the data of second level cache that is not enabled by default.

more details...

16) Is SessionFactory a thread-safe object?

Yes, SessionFactory is a thread-safe object, many threads cannot access it simultaneously.

---

17) What is Session?

It maintains a connection between the hibernate application and database.

It provides methods to store, update, delete or fetch data from the database such as persist(), update(), delete(), load(), get() etc.

It is a factory of Query, Criteria and Transaction i.e. it provides factory methods to return these instances.

more details...

---

18) Is Session a thread-safe object?

No, Session is not a thread-safe object, many threads can access it simultaneously. In other words, you can share it between threads.

---

19) What is the difference between session.save() and session.persist() method?

No.	save()	persist()
1)	returns the identifier (Serializable) of the instance.	Return nothing because its return type is void.
2)	Syn: public Serializable save(Object o)	Syn: public void persist(Object o)

20) What is the difference between get and load method?

The differences between get() and load() methods are given below.

No.	get()	load()
1)	Returns **null** if an object is not found.	Throws **ObjectNotFoundException** if an object is not found.
2)	get() method always **hit the database**.	load() method **doesn't hit** the database.
3)	It returns the real object, not the proxy.	It returns **proxy object.**
4)	It should be used if **you are not sure** about the existence of instance.	It should be used if **you are sure** that instance exists.

21) What is the difference between update and merge method?

The differences between update() and merge() methods are given below.

No.	The update() method	merge() method
1)	Update means to edit something.	Merge means to combine something.
2)	update() should be used if the session doesn't contain an already persistent state with the same id. It means an update should be used inside the session only. After closing the session, it will throw the error.	merge() should be used if you don't know the state of the session, means you want to make the modification at any time.

Let's try to understand the difference by the example given below:

1. ...
2. SessionFactory factory = cfg.buildSessionFactory();
3. Session session1 = factory.openSession();

4.

5. Employee e1 = (Employee) session1.get(Employee.**class**, Integer.valueOf(101));
   //passing id of employee
6. session1.close();

7.

8. e1.setSalary(70000);

9.

10. Session session2 = factory.openSession();
11. Employee e2 = (Employee) session1.get(Employee.**class**, Integer.valueOf(101));
    //passing same id

12.

13. Transaction tx=session2.beginTransaction();
14. session2.merge(e1);

15.

16. tx.commit();
17. session2.close();

After closing session1, e1 is in detached state. It will not be in the session1 cache. So if you call update() method, it will throw an error.

Then, we opened another session and loaded the same Employee instance. If we call merge in session2, changes of e1 will be merged in e2.

22) What are the states of the object in hibernate?

There are 3 states of the object (instance) in hibernate.

1. **Transient**: The object is in a transient state if it is just created but has no primary key (identifier) and not associated with a session.
2. **Persistent**: The object is in a persistent state if a session is open, and you just saved the instance in the database or retrieved the instance from the database.
3. **Detached**: The object is in a detached state if a session is closed. After detached state, the object comes to persistent state if you call lock() or update() method.

23) What are the inheritance mapping strategies?

There are 3 ways of inheritance mapping in hibernate.

1. Table per hierarchy
2. Table per concrete class
3. Table per subclass

more details...

24) How to make an immutable class in hibernate?

If you mark a class as mutable="false", the class will be treated as an immutable class. By default, it is mutable="true".

25) What is automatic dirty checking in hibernate?

The automatic dirty checking feature of Hibernate, calls update statement automatically on the objects that are modified in a transaction.

Let's understand it by the example given below:

1. ...
2. SessionFactory factory = cfg.buildSessionFactory();
3. Session session1 = factory.openSession();
4. Transaction tx=session2.beginTransaction();
5.
6. Employee e1 = (Employee) session1.get(Employee.**class**, Integer.valueOf(101));

7.
8. e1.setSalary(70000);
9.
10. tx.commit();
11. session1.close();

Here, after getting employee instance e1 and we are changing the state of e1.

After changing the state, we are committing the transaction. In such a case, the state will be updated automatically. This is known as dirty checking in hibernate.

26) How many types of association mapping are possible in hibernate?

There can be 4 types of association mapping in hibernate.

1. One to One
2. One to Many
3. Many to One
4. Many to Many

27) Is it possible to perform collection mapping with One-to-One and Many-to-One?

No, collection mapping can only be performed with One-to-Many and Many-to-Many.

28) What is lazy loading in hibernate?

Lazy loading in hibernate improves the performance. It loads the child objects on demand.

Since Hibernate 3, lazy loading is enabled by default, and you don't need to do lazy="true". It means not to load the child objects when the parent is loaded.

29) What is HQL (Hibernate Query Language)?

Hibernate Query Language is known as an object-oriented query language. It is like a structured query language (SQL).

The main advantage of HQL over SQL is:

1. You don't need to learn SQL
2. Database independent
3. Simple to write a query

30) What is the difference between first level cache and second level cache?

No.	First Level Cache	Second Level Cache
1)	First Level Cache is **associated with Session**.	Second Level Cache is associated with **SessionFactory**.
2)	It is **enabled** by default.	It is **not enabled** by default.

31) What is the difference between Lazy vs Eager Loading?

Lazy and Eager are two types of data loading strategies in ORMs such as hibernate and eclipse Link. These data loading strategies we used when one entity class is having references to other Entities like Employee and Phone (phone in the employee).

Lazy Loading – Associated data loads only when we explicitly call getter or size method.

- Use Lazy Loading when you are using one-to-many collections.
- Use Lazy Loading when you are sure that you are not using related entities.

Egare Loading – Data loading happens at the time of their parent is fetched.

- Use Eager Loading when the relations are not too much. Thus, Eager Loading is a good practice to reduce further queries on the Server.
- Use Eager Loading when you are sure that you will be using related entities with the main entity everywhere.

Sr. No.	Key	Lazy	Eager
1	Fetching strategy	In Lazy loading, associated data loads only when we explicitly call getter or size method.	In Eager loading, data loading happens at the time of their parent is fetched
2	Default Strategy in ORM Layers	ManyToMany and OneToMany associations used lazy loading strategy by default.	ManyToOne and OneToOne associations used lazy loading strategy by default.

Sr. No.	Key	Lazy	Eager
3	Loading Configuration	It can be enabled by using the annotation parameter : fetch = FetchType.LAZY	It can be enabled by using the annotation parameter : fetch = FetchType.EAGER
4	Performance	Initial load time much smaller than Eager loading	Loading too much unnecessary data might impact performance