1. What is Hibernate?

Hibernate is an Object-Relational Mapping ORM solution for JAVA and it raised as an open source persistent framework. Hibernate maps Java classes

to database tables and from Java data types to SQL data types and relieve the developer from 95% of common data persistence related programming tasks.

2.Difference between FetchType LAZY and EAGER in Java Persistence API?

FetchType.LAZY = This does not load the relationships unless you invoke it via the getter method.

Lazy initialization improves performance by avoiding unnecessary computation and reduce memory requirements.

FetchType.EAGER = This loads all the relationships.

Eager initialization takes more memory consumption and processing speed is slow.

The Lazy Fetch type is by default selected by Hibernate unless you explicitly mark Eager Fetch type.

In JPA the defaults are:

OneToMany: LAZY

ManyToOne: EAGER

ManyToMany: LAZY

OneToOne: EAGER

All data is fetched when eager marked data in the object when session is connected. However, in case of lazy loading strategy,

lazy loading marked object does not retrieve data if session is disconnected (after session.close() statement).

All that can be made by hibernate proxy. Eager strategy lets data to be still available after closing session.

3. Difference between get() vs load() method in Hibernate?

The key difference between get() and load() method is that load() will throw an exception if an object with id passed to them is not found,

But get() will return null. Another important difference is that load() can return proxy without hitting the database unless required (when you access any attribute other than id) but get() always go to the database, so sometimes using load() can be faster than the get() method.

It makes sense to use the load() method if you know the object exists but get() method if you are not sure about object's existence.

4. What are the advantages of Hibernate over JDBC?

1) Caching

2) Lazy Loading

3)Hibernate takes care of mapping Java classes to database tables using XML files and without writing any line of code.

Provides simple APIs for storing and retrieving Java objects directly to and from the database.

If there is change in Database or in any table then the only need to change XML file

properties.

Abstract away the unfamiliar SQL types and provide us to work around familiar Java Objects.

Hibernate does not require an application server to operate.

Manipulates Complex associations of objects of your database.

Minimize database access with smart fetching strategies.

Hibernate supports almost all the major RDBMS.

5. What is the difference between save() and persist() method in Hibernate?

First difference between save and persist is there return type. The return type of persist method is void while return type of save method is Serializable object. But bot of them also INSERT records into database

Another difference between persist and save is that both methods make a transient object to persistent state. However, persist() method doesn’t guarantee that the identifier value will be assigned to the persistent state immediately, the assignment might happen at flush time.

Third difference between save and persist method in Hibernate is behavior on outside of transaction boundaries. persist() method will not execute an insert query if it is called outside of transaction boundaries. Because save() method returns an identifier so that an insert query is executed immediately to get the identifier, no matter if it are inside or outside of a transaction.

Fourth difference between save and persist method in Hibernate: persist method is called outside of transaction boundaries, it is useful in long-running conversations with an extended Session context. On the other hand save method is no

6. **What is different between Session and Sessionfactory in Hibernate?**(detailed answer)  
This is another popular Hibernate interview question, mostly at a telephonic round of interviews. The main difference between Session and SessionFactory is that former is a single-threaded, short-lived object while later is Immutable and shared by all Session. It also lives until the Hibernate is running. Another difference between Session and SessionFactory is that former provides first level cache while SessionFactory provides the Second level cache.

**7. What is criterion query in hibernate?**   
Criteria is a simplified API for retrieving entities by composing Criterion objects also known as Criterion query. This is a very convenient approach for functionality like "search" screens where you can filter data on multiple conditions as shown in the following example:

List books = session.createCriteria(Book.class)

.add(Restrictions.like("name", "java%") )

.add(Restrictions.like("published\_year", "2015"))

.addOrder(Order.asc("name") )

.list();

8. Which cache is used by Session Object in Hibernate? First level or second level cache?

A Session object uses the first-level cache. The second level cache is used at SessionFactory level.

9. What is the difference between the transient, persistent and detached state in Hibernate?  
ans : New objects created in Java program but not associated with any hibernate Session are said to be in the transient state. On the other hand, an object which is associated with a Hibernate session is called Persistent object. While an object which was earlier associated with Hibernate session but currently it's not associate is known as a detached object. You can call save() or persist() method to store those object into the database and bring them into the Persistent state. Similarly, you can re-attach a detached object to hibernate sessions by calling either update() or saveOrUpdate() method.

10. What are the three states of a Hibernate Persistence object can be?   
The Hibernate persistent or entity object can live in following three states:  
1) transient  
2) persistent  
3) detached

11. **How do you log SQL queries issued by the Hibernate framework in Java application?**  
You can use the show\_sql property to log SQL queries issued by the Hibernate framework, Just add the following line in your Hibernate configuration file:  
<property name=”show\_sql”>true</property>

12. **The difference between sorted and ordered collection in Hibernate?**

The main difference between sorted and ordered collection is that sorted collection sort the data in JVM's heap memory using Java's collection framework sorting methods while ordered collection is sorted using order by clause in the database itself. A sorted collection is more suited for small dataset but for a large dataset, it's better to use ordered collection to avoid [OutOfMemoryError in Java](http://java67.blogspot.com/2013/08/guide-of-javalangoutofmemoryerror-java-heap-space-tomcat-eclipse-minecraft-jboss.html) application.

### 13. What is the difference between update and merge method?

Update : Update means to edit something.update() should be used if session doesn't contain an already persistent state with same id. It means update should be used inside the session only. After closing the session it will throw error.

Merge() : Merge means to combine something.merge() should be used if you don't know the state of the session, means you want to make modification at any time.

1. ...
2. SessionFactory factory = cfg.buildSessionFactory();
3. Session session1 = factory.openSession();
5. Employee e1 = (Employee) session1.get(Employee.**class**, Integer.valueOf(101));//passing id of employee
6. session1.close();
8. e1.setSalary(70000);
10. Session session2 = factory.openSession();
11. Employee e2 = (Employee) session1.get(Employee.**class**, Integer.valueOf(101));//passing same id
13. Transaction tx=session2.beginTransaction();
14. session2.merge(e1);
16. tx.commit();
17. session2.close();

After closing session1, e1 is in detached state. It will not be in 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.

### 14) What is HQL (Hibernate Query Language)?

Hibernate Query Language is known as an object oriented query language. It is like 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 query

15. **What is difference between getCurrentSession() and openSession() in Hibernate?**(  
https://java2blog.com/difference-opensession-getcurrentsession-hibernate/

**16. What is the difference between save() and saveOrUpdate() method of Hibernate?**([detailed answer](http://javarevisited.blogspot.com/2012/09/difference-hibernate-save-vs-persist-and-saveOrUpdate.html))  
Though both save() and saveOrUpdate() method is used to store object into Database, the key difference between them is that save can only INSERT records but saveOrUpdate() can either INSERT or UPDATE records.

17.**What are other ORM frameworks? Any alternative of Hibernate?**  
This is a general question, sometimes asked to start the conversation and other times to finish the interview. EJB and TopLink from Oracle are two of the most popular alternative to Hibernate framework.

### 18) What are the core interfaces of Hibernate?

The core interfaces of Hibernate framework are:

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

**19. What’s the usage of callback interfaces in hibernate?**  
Callback interfaces of hibernate are useful in receiving event notifications from objects. For example, when an object is loaded or deleted, an event is generated and notification is sent using callback interfaces.

12. why we dialect in hibernate configuration file?

This property makes Hibernate generate the appropriate SQL for the chosen database. Hibernate uses "dialect" configuration to know which database you are using so that it can switch to the database specific SQL generator code wherever/whenever necessary.

**What is the requirement for a Java object to become Hibernate entity object?**

It should not be final and must provide a default, no-argument constructor.

Explanation:

In order to use a proxy in place of real class, your hibernate persistence class must be either non-final or the implementation of an interface that declares all of the public methods. Why? because [you cannot extend a final class in Java](http://javarevisited.blogspot.com/2011/12/final-variable-method-class-java.html), and to stand up as a proxy, proxy class must satisfy the **IS-A** relation, which comes either by extending a class using "extends", or implementing an interface using "implements". By the way, it doesn't mean that you cannot persist your final entity class, you can, but this will limit Hibernate's ability to use proxies for lazy association fetching, which will affect the performance of Java application,

# Difference Between Merge And Update Methods In Hibernate

As we know that update() and merge() methods in hibernate are used to convert the object which is in detached state into persistence state. But there are different situation where we should be used update() and where should be used merge() method in hibernate, let us see below snippet of codes.

Employee emp1 = new Employee();

emp1.setEmpId(100);

emp1.setEmpName("Dinesh");

//create session

Session session1 = createNewHibernateSession();

session1.saveOrUpdate(emp1);

session1.close();

//emp1 object in detached state now

emp1.setEmpName("Dinesh Rajput");//Updated Name

//Create session again

Session session2 = createNewHibernateSession();

Employee emp2 =(Employee)session2.get(Employee.class, 100);

//emp2 object in persistent state with id 100

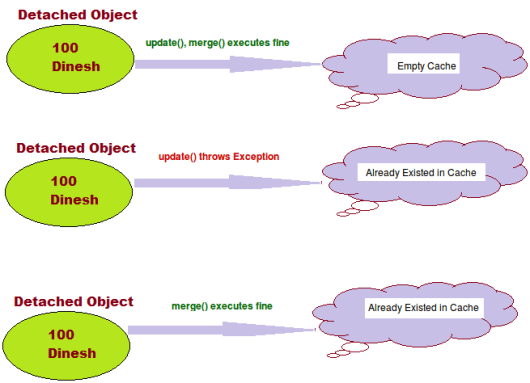
//below we try to make on detached object with id 100 to persistent state by using update method of hibernate

session2.update(emp1);//It occurs the exception NonUniqueObjectException because emp2 object is having employee with same empid as 100. In order //to avoid this exception we are using merge like given below instead of session.update(emp1);

session2.merge(emp1); //it merge the object state with emp2

session2.update(emp1); //Now it will work with exception

In the hibernate session we can maintain only one employee object in persistent state with same primary key, while converting a detached object into persistent, if already that session has a persistent object with the same primary key then hibernate throws an Exception whenever update() method is called to reattach a detached object with a session. In this case we need to call merge() method instead of update() so that hibernate copies the state changes from detached object into persistent object and we can say a detached object is converted into a persistent object.



Hibernate handles persisting any changes to objects in the session when the session is flushed. update can fail if an instance of the object is already in the session. Merge should be used in that case. It merges the changes of the detached object with an object in the session, if it exists.

**Update:** Suppose we are dealing with any employee object in the same session then we should use update() or saveOrUpdate() method.

**Update:** if you are sure that the session does not contains an already persistent instance with the same identifier,then use update to save the data in hibernate

**Merge:** Suppose we are creating a session and load an employee object. Now object in session cache. If we close the session at this point and we edit state of object and tried to save using update() it will throw exception. To make object persistent we need to open another session. Now we load same object again in current session. So if we want to update present object with previous object changes we have to use merge() method. Merge method will merge changes of both states of object and will save in database.

**Merge:** if you want to save your modifications at any time with out knowing about the state of an session, then use merge() in hibernate.

Difference between openSession and currentSession

|  |  |  |
| --- | --- | --- |
| **Parameter** | **openSession** | **getCurrentSession** |
| Session object | It always create new Session object | It creates a new Session if not exists , else use same session which is in current hibernate context |
| Flush and close | You need to explicitly flush and close session objects | You do not need to flush and close session objects, it will be automatically taken care by Hibernate internally |
| Performance | In single threaded environment , It is slower than getCurrentSession | In single threaded environment , It is faster than getOpenSession |
| Configuration | You do not need to configure any property to call this method | You need to configure additional property “hibernate.current\_session\_context\_class” to call getCurrentSession method, otherwise it will throw exceptions. |

**Hibernate save()** can be used to save entity to database. We can invoke this method outside a transaction, that’s why I don’t like this method to save data. If we use this without transaction and we have cascading between entities, then only the primary entity gets saved **unless we flush the session**.

**flush():** Forces the session to flush. It is used to synchronize session data with database.

When you call session.flush(), the statements are executed in database but it will not committed.  
If you don’t call session.flush() and if you call session.commit() , internally commit() method executes the statement and commits.

So **commit()= flush+commit.**  
So seesion.flush() just executes the statements in database (but not commits) and statements are NOT IN MEMORY anymore. It just forces the session to flush.

Few important points:

* We should avoid save outside transaction boundary, otherwise mapped entities will not be saved causing data inconsistency. It’s very normal to forget flushing the session because it doesn’t throw any exception or warnings.
* By default, Hibernate will flush changes automatically for you:

Generaor tag

Optional tag under id tag of mapping file, it specifires to be used by hibernate to generate primary key.

Hibernate generator clases are :

Assigned : developer has to provide the primary key information.

Increment : hibernate responsible for providing primary key.

Identity : DB Server is responsible for providing primary key.

Native :

Hilo

Uuid …………….

Hibernate.cfg.xml :

The default file name for the configuration file is hibernate.cfg.xml and default locaton is “src” i.e project class path

However this configuration file can have ny name and can be present anywhere in world.

Hibernate expects one configuration file per database i.e if our apps has more then one DB those many configuration files should be there.

Note : if we commit the Transcation then its not necessary to flush the session , because commit internally invokes flush() method.

* If we add property by name “transcation.auto\_close\_session” with the value “true” in the hibernate configuration file then its not necessary to close the session. Session will be automatically closed after the completion of transcation.
* <property name=” transcation.auto\_close\_session”>true</property>
* In case of JDBC by default “auto-commit” mode is set to true. But incase of hibernate it is set to false.Hence to work with hibernate we must make use of transcations however if we set property by name “connection.auto\_commit” to “true” then we need not to use transcations even in case of hibernate.