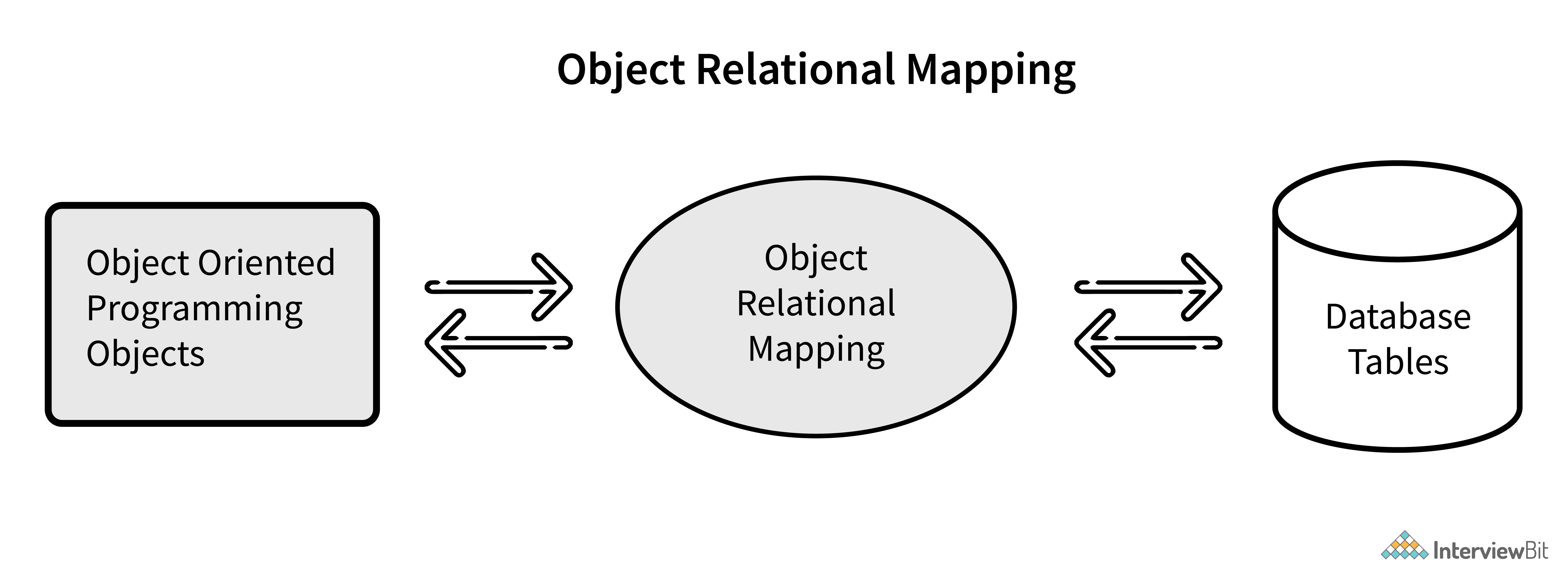
**Hibernate**

**1. What is ORM in Hibernate?**

**Ans :-** orm is Object Relational Mapping. Mainly used for converting data stored in a relational database to an object used in oop constructs. This tool greatly in simplifying data retrieval creation and manipulation.



### 2. What are the advantages of Hibernate over JDBC?

Ans :-

**Clean Readable Code**

**HQL (Hibernate Query Language):** Hibernate provides HQL which is closer to Java and is object-oriented in nature.

* **Transaction Management:** JDBC doesn't support implicit transaction management. It is upon the developer to write transaction management code using commit and rollback methods. Whereas, Hibernate implicity provides this feature.
* **Exception Handling:** Hibernate wraps the JDBC exceptions and throws unchecked exceptions like JDBCException or HibernateException. This along with the built-in transaction management system helps developers to avoid writing multiple try-catch blocks to handle exceptions. In the case of JDBC, it throws a checked exception called SQLException thereby mandating the developer to write try-catch blocks to handle this exception at compile time.

**Special Features:** Hibernate supports OOPs features like inheritance, associations and also supports collections.

**What are some of the important interfaces of Hibernate framework?**

Hibernate core interfaces are:

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

### 4. What is a Session in Hibernate?

A session is an object that maintains the connection between Java **object application** and **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.

### 5. What is a SessionFactory?

SessionFactory provides an instance of **Session** Session **objects**  on configuration  parameters in order to establish the connection to the database. It single instance of SessionFactory. The internal state of a SessionFactory which includes metadata about ORM is immutable

### Can you explain what is lazy loading in hibernate?

Lazy loading is mainly used for improving the application performance by helping to load the child objects on demand.

**8. What is the difference between first level cache and second level cache?**

Hibernate has 2 cache types. First level and second level cache for which the difference is given below:

| **First Level Cache** | **Second Level Cache** |
| --- | --- |
| This is local to the Session object and cannot be shared between multiple sessions. | This cache is maintained at the SessionFactory level and shared among all sessions in Hibernate. |
| This cache is enabled by default and there is no way to disable it. | This is disabled by default, but we can enable it through configuration. |
| The first level cache is available only until the session is open, once the session is closed, the first level cache is destroyed. | The second-level cache is available through the application’s life cycle, it is only destroyed and recreated when an application is restarted. |

### What can you tell about Hibernate Configuration File?

**hibernate.cfg.xml** is one of the most required configuration files in Hibernate.

default, this file is placed under the src/main/resource folder.

The file contains database related and session-related  configurations .

* Database connection details: Driver class, URL, username, and password.
* There must be one configuration file for each database used in the application, suppose if we want to connect with 2 databases, then we must create 2 configuration files with different names.
* Hibernate properties: Dialect, show\_sql, second\_level\_cache, and mapping file names.

### 10. How do you create an immutable class in hibernate?

In the case of using annotations, immutable classes in hibernate can also be created by using @Immutable annotation.

### 13. Explain hibernate mapping file

Hibernate mapping file is an XML file that is used for defining the entity bean fields and corresponding database column mappings.  
These files are useful when the project uses third-party classes where JPA annotations provided by hibernating cannot be used.

### Explain Hibernate architecture

* SessionFactory: This provides a factory method to get session objects and clients of ConnectionProvider. It holds a second-level cache (optional) of data.
* Session: This is a short-lived object that acts as an interface between the java application objects and database data.
  + The session can be used to generate transaction, query, and criteria objects.
  + It also has a mandatory first-level cache of data.
* Transaction: This object specifies the atomic unit of work and has methods useful for transaction management. This is optional.
* ConnectionProvider: This is a factory of JDBC connection objects and it provides an abstraction to the application from the DriverManager. This is optional.
* TransactionFactory: This is a factory of Transaction objects. It is optional.

### https://s3.ap-south-1.amazonaws.com/myinterviewtrainer-domestic/public_assets/assets/000/000/049/original/Hibernate_Framework_Objects.png?1614765478

### 16. Can you tell the difference between getCurrentSession and openSession methods?

**16. Can you tell the difference between getCurrentSession and openSession methods?**

Both the methods are provided by the Session Factory. The main differences are given below:

| **getCurrentSession()** | **openSession()** |
| --- | --- |
| This method returns the session bound to the context. | This method always opens a new session. |
| hibernate configuration file has to be modified by adding **<property name = "hibernate.current\_session\_context\_class"> thread </property>**. If not added, then using the method would throw an HibernateException. | created for each request in a multi-threaded environment. Hence, you need not configure any property to call this method. |
| This session object gets closed once the session factory is closed. | It's the developer’s responsibility to close this object once all the database operations are done. |
| In a single-threaded environment, this method is faster than openSession(). | In single threaded environment, it is slower than getCurrentSession()single-threadeda |

**. Differentiate between save() and saveOrUpdate() methods in hibernate session.**

Both the methods save records to the table in the database in case there are no records with the primary key in the table. However, the main differences between these two are listed below:

| **save()** | **saveOrUpdate()** |
| --- | --- |
| save() generates a new identifier and INSERT record into a database | Session.saveOrUpdate() can either INSERT or UPDATE based upon existence of a record. |
| The insertion fails if the primary key already exists in the table. | In case the primary key already exists, then the record is updated. |
| The return type is Serializable which is the newly generated identifier id value as a Serializable object. | The return type of the saveOrUpdate() method is void. |
| This method is used to bring only a transient object to a persistent state. | This method can bring both transient (new) and detached (existing) objects into a persistent state. It is often used to re-attach a detached object into a Session |

**Differentiate between get() and load() in Hibernate session**

These are the methods to get data from the database. The primary differences between get and load in Hibernate are given below:

| **get()** | **load()** |
| --- | --- |
| This method gets the data from the database as soon as it is called. | This method returns a proxy object and loads the data only when it is required. |
| The database is hit every time the method is called. | The database is hit only when it is really needed and this is called Lazy Loading which makes the method better. |
| The method returns null if the object is not found. | The method throws ObjectNotFoundException if the object is not found. |
| Only use when we dnot unsure method in database | Only use this method we are sure method in db |

### What is criteria API in hibernate?

Criteria API in Hibernate helps to  **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.

 The org.hibernate.Criteria interface is used for these purposes. The Session interface has createCriteria() method that  the persistent object’s class or its entity name as the parameters and returns persistence object instance the criteria query is executed.

 It can be achieved by using the add() method which accepts the org.hibernate.criterion.Criterion object representing individual restriction.

Criteria criteria= session.createCriteria(InterviewBitEmployee.class);

criteria.add(Restrictions.eq("fullName","Hibernate"));

List<InterviewBitEmployee> results = criteria.list();

### 20. What is HQL?

### Can you tell something about one to many associations and how can we use them in Hibernate?

The **one-to-many association** is the most commonly used which indicates that one object is linked/associated with multiple objects.

### What does session.lock() method in hibernate do?

**session.lock()** method is used to reattach a detached object to the session. **session.lock()** method does not check for any data synchronization between the database. Again reattachment lead to loss of data synchronization.

### 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.

### Types of Hibernate Caching

**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, JBoss Cache are some example cache providers.

### When is merge() method of the hibernate session useful?

Merge() method can be used for updating existing values. the existing values are updated, the method creates a copy from the entity object and returns it.

### Can you tell the difference between setMaxResults() and setFetchSize() of Query?

setMaxResults() the function works similar to LIMIT in SQL. Here, we set the maximum number of rows that we want to be returned. This method is implemented by all database drivers.

setFetchSize() works for optimizing how Hibernate sends the result to the caller for example: are the results buffered, are they sent in different size chunks, etc. This method is not implemented by all the database drivers.

**Does Hibernate support Native SQL Queries?**

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

Consider the example where you want to get employee data with the full name “Hibernate”. We don’t want to use HQL-based features, instead, we want to write our own SQL queries. In this case, the code would be:

Query query = session.createSQLQuery( "select \* from interviewbit\_employee ibe where ibe.fullName = :fullName")

.addEntity(InterviewBitEmployee.class)

.setParameter("fullName", "Hibernate"); //named parameters

List result = query.list();

Alternatively, native queries can also be supported when using NamedQueries.

### Can we declare the Entity class final?

No, we should not define the entity class final because hibernate uses proxy classes and objects for lazy loading of data and hits the database only when it is absolutely needed. This is achieved by extending the entity bean. If the entity class (or bean) is made final, then it cant be extended and hence lazy loading can not be supported.

### What are the states of a persistent entity?

Object ko db me save karta h hb ki help se wha kuch states me ho kar gurat h.

**Transient:** This state is the initial state jb hum ko object me value ko save kr dete h . is this a transient states. Ye heap memory ko use karta h .this object not link session.

InterviewBitEmployee employee=new InterviewBitEmployee(); //The object is in the transient state.

employee.setId(101);

employee.setFullName("Hibernate");

employee.setEmail("hibernate@interviewbit.com");

**Persistent : - is state me aate hi object session ke sath** associated  (jud ) jata h. object sath me db me save ho jata h. hum us object ko kuch change karte to wo reflect session and db pr padta h.

session.save(record);

session.persist(record);

session.update(record);

session.saveOrUpdate(record);

session.lock(record);

session.merge(record);

**Detached: object is state me jat aata h jb us ka session se connection close ho jata h . fir object persisten and detached state me rahta h object me value change hone pr db pr koi reflect nahi pata h. session fir create karne ke liye** load(), merge(), refresh(), update(), or save()  help karte h .

session.close();

session.clear();

session.detach(record);

session.evict(record);

### Explain Query Cache

Hibernate framework provides an optional feature called cache region for the queries’ resultset. Additional configurations have to be done in code in order to enable this. Query cache use karta h db ke object ko save kar leta h cache me jb java get object karta h fastly check in cache and second check in db

### Can you tell something about the N+1 SELECT problem in Hibernate?

N+1 SELECT problem is due to the result of using lazy loading and on-demand fetching strategy.Let's say you have a collection of Car objects (database rows), and each Car has a collection of Wheel objects (also rows). In other words, Car → Wheel is a 1-to-many relationship.

Now, let's say you need to iterate through all the cars, and for each one, print out a list of the wheels. The naive O/R implementation would do the following:

### How to solve N+1 SELECT problem in Hibernate?

* Subselect the fetching strategy
* As last resort, try to avoid or disable lazy loading altogether.

**Fetch Type**

Lazy :- it call getter and setter method call is lazy working

Eager :- it loading jb programme chalta h tabi

### 35. What are the concurrency strategies available in hibernate?

It responsible for storing and retrieving items from the cache.

**Transactional: -** This is used in cases of updating data that most likely

* **Read-Only:** This is used when we don't want the data to be modified and can be used for reference data only.
* **Read-Write:** Here, data is mostly read and is used when the prevention of stale data is of critical importance.
* **Non-strict-Read-Write:** Using this strategy will ensure that there wouldn't be any consistency between the database and cache. This strategy can be used when the data can be modified and stale data is not of critical concern.

### Can you tell 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 to 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.

**Code Snippet: We can define Named Query as shown below**

@NamedQueries(

{

@NamedQuery(

name = "findIBEmployeeByFullName",

query = "from InterviewBitEmployee e where e.fullName = :fullName"

)

}

)