## Why Object Relational Mapping (ORM)?

When we work with an object-oriented system, there is a mismatch between the object model and the relational database. **RDBMSs represent data** in a tabular format whereas **object-oriented languages**, such as Java or C# represent it as an interconnected graph of objects.

1. Hibernate is a Java framework that helps Java application to interact with the database.

**2. Hibernate** **is an ORM (Object Relational Mapping) tool.**

3. Hibernate implements the specifications of **JPA (Java Persistence API) for data persistence.**

**ORM tool**

It is a **programming technique** that maps the object to the data stored in the database.



The ORM tool internally uses the **JDBC** API to interact with the database.

What is JPA?

It provides a platform to work directly with **objects instead of using SQL statements.**

Java Persistence API (JPA) is a Java specification **that provides certain functionality and standard** to ORM tools. The **javax.persistence** package contains the JPA classes and interfaces.



Hibernate framework uses many objects such as session factory, session, transaction etc. along with existing Java API such as JDBC (Java Database Connectivity), JTA (Java Transaction API) and JNDI (Java Naming Directory Interface).

#### **SessionFactory**

The SessionFactory is a factory of instance of **session** and client of **ConnectionProvider.** It holds second level cache (optional) of data. The org.hibernate.SessionFactory interface

#### **Session**

It maintains a connection between the hibernate application and database**. It is factory of Transaction, Query and Criteria.** It holds a first-level cache (mandatory) of data.

#### **Transaction**

The transaction object specifies the **atomic unit of work**. It is optional. The org.hibernate.Transaction interface provides methods for transaction management.

#### **ConnectionProvider**

It is a factory of JDBC connections. It abstracts the application from DriverManager or DataSource. It is optional.

#### **TransactionFactory**

It is a factory of Transaction. It is optional.

Example:

1. SessionFactory factory = meta.getSessionFactoryBuilder().build();
2. Session session = factory.openSession();
3. Transaction t = session.beginTransaction();

**If your application is referring to multiple databases, then you need to create one SessionFactory per database.**

Program:

1.Create the Persistent class

2.Create the mapping file for Persistent class

3.Create the Configuration file

4.Create the class that retrieves or stores the persistent object

1.Create the Persistent class

**A no-arg constructor:**

**Provide an identifier property:**  It is better to assign an attribute as i**id**

**Declare getter and setter methods:**

**Prefer non-final class:**

2. Create the mapping file for Persistent class(employee.hbm.xml)

<hibernate-mapping>

  <**class** name="com.javatpoint.mypackage.Employee" table="emp1000">

    <id name="id">

     <generator **class**="assigned"></generator>

    </id>

     <property name="firstName"></property>

    <property name="lastName"></property>

  </**class**>

</hibernate-mapping>

#### 3.Create the Configuration file(**hibernate.cfg.xml**)

<hibernate-configuration>

    <session-factory>

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

        <property name="dialect">org.hibernate.dialect.Oracle9Dialect</property>

        <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>

        <property name="connection.username">system</property>

        <property name="connection.password">oracle</property>

<property name="connection.driver\_class">oracle.jdbc.driver.OracleDriver</propert>

    <mapping resource="employee.hbm.xml"/>

   </session-factory>

 </hibernate-configuration>

4. Create the class that retrieves or stores the persistent object(Dao.java)

1.SessionFactory factory = meta.getSessionFactoryBuilder().build();

2.Session session = factory.openSession();

3.Transaction t = session.beginTransaction();

"hbm2ddl.auto//create table automatically ddl-data definition language

Table configuration present in .hbm file

**<generator>**

The <generator> class is a sub-element of id. It is used to **generate the unique** identifier **for the objects** of persistent class.

**org.hibernate.id.IdentifierGenerator**[**interface**](https://www.javatpoint.com/interface-in-java).

1. Assigned// The **assigned generator** is a default generator.
2. increment
3. sequence
4. hilo
5. native
6. identity
7. seqhilo etc…

Example:

<**class** ...>

    <id ...>

     <generator **class**="assigned"></generator>

</id>

**Dialect**:

The dialect **specifies the type of database** used in hibernate so that hibernate generate appropriate type of SQL statements. For connecting any hibernate application with the database.

<property name="dialect">org.hibernate.dialect.Oracle9Dialect</property>

There are three **inheritance mapping strategies** defined in the hibernate:

1. Table Per Hierarchy: **single table is required to map the whole hierarchy,** an extra column (known as discriminator column) is added to identify the class.
2. Table Per Concrete class:  tables are created as per class. But duplicate column is added in subclass tables.
3. Table Per Subclass: tables are created as per class but related by foreign key. So there are no duplicate columns

The core interfaces/components of Hibernate framework are:

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

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

Session.createSQLQuery

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

Session.createQuery

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

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

Session is **not a thread-safe** object, many threads can access it simultaneously.

For hibernate mapping, the file name should be like **filename.hbm.xml**.

For hibernate configuration, the file name should be like **hibernate.cfg.xml**.

**Session Methods:**

session maintains a connection between the hibernate application and database.

It provides methods

Session.

### 1. beginTransaction()

### 2. save()

### 3. persist()

### 4. update()

### 5. saveOrUpdate()

### 6. createQuery()

### 7. createSQLQuery()

### 8. merge()

### 9. flush()

### 10. delete()

|  |  |  |
| --- | --- | --- |
| **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 |

### **States of the object in hibernate**

*1. transient:* object never persistent, not associated with any Session  
*2.persistent:* object associated with a unique Session  
*3.detached:* object previously persistent, not associated with any Session

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

The HCQL provides methods to add criteria:**public** Criteria createCriteria(Class c)

There are two ways to perform **logging using log4j:**

1. By log4j.xml file (**or**)
2. By log4j.properties file

### **Mention two components of Hibernate configuration object.**

**Database Connection**

**Class Mapping Setup**

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 | | |