
HIBERNATE TUTORIAL - 1

Hibernate

When we learn a new tool, we should understand;

1)What is it?

2)Why do we need it?

3)How to implement it?

4)What are the advance Concepts?

What is Persistence?

Persistence means the process of storing and managing data for long time.
Persistence is never related with RAM Memory, it is related with secondary memory like Files, HardDisks, CDs, DVDs, DataBases etc.
Persistence is never related with storing data in variables or objects because they are temporary memories

If we do not use Persistence, after the execution data will be vanished.
For example when you create a list an add elements into it, the data will be stored in RAM Memory and after sometimes it will be vanished.

Persistence Store: Databases, Files

Persistence Data: Data in files; tables, records in Databases etc.

Persistence Operations: CRUD operations

Persistence Logic: The logic which is written to perform CRUD operations.
For example; if you use Files*(text, xml)* as Persistence Store you need to use IO Stream logic.
In IO Stream you need to use Serialization and De-Serialization.
If you use DataBases as Persistence Store you need to use JDBC logic, ORM(Object Relational Mapping) logic.

Persistence Technology / Framework: We use persistence technology / framework to develop persistence logic.
JDBC is a technology *(Semi-finished product, we develop the rest and use them - frozen food)*,
Hibernate is a Framework *(Fully-finished product - McDonalds)* or
Tool *(They name it as tool as well because it makes our life easy)*

Cons of Using Files as Persistence Storage

Text files and XML files are called Flat files.

Using Text files as storage is a)No Security
b)Difficult in CRUD operations
c)SQL does not support
d) No Constraints
e) Merge and Comparison is complex

They can be used for Small Applications or Memory Critical Applications like simple mobile games, mobile applications, desktop applications but Java is used to create at least Medium or Large scale applications

Why do we need Hibernate?

1) Hibernate mainly solves the object-relational mismatch problems which arise when a relational database is connected by an application written in object oriented programming language style.

Object-relational mismatch includes mismatches arises due to data type differences, manipulative differences, transactional differences, structural differences etc.

2) In JDBC, you need to write code to map the object model's data representation to a relational model and its corresponding schema. Hibernate itself maps the Java classes to the database tables using XML configuration or by using annotations.

JDBC:

```
List<User> users = new ArrayList<User>();
while(rs.next()) {
    User user = new User();
    user.setUserId(rs.getString("UserId"));
    user.setName(rs.getString("FirstName"));
    user.setEmail(rs.getString("Email"));
    users.add(user);
}
```

Hibernate:

```
@Entity
public class UserModel {
@Id
private BigInteger id;
@Column(name= "user_email", unique=true)
private String email;
private String name;
public BigInteger getId() { return this.id; }
public void setId(BigInteger id) { this.id = id; }
public String getEmail() { return email; }
public void setEmail(String email) { this.email = email; }
}
```

In the example, you can see that by using JDBC you need to set every property of an object on fetching the data each and every time. But in Hibernate we need to map the table with the java class as mentioned.

3) Hibernate uses HQL (Hibernate Query Language) which is similar to SQL but Hibernate's HQL provides full support to polymorphic queries. HQL understands object-oriented concepts like inheritance, polymorphism, and encapsulation

4) Hibernate's code is database independent because you do not need to change the HQL queries when you change the databases like MySQL, oracle, etc and hence it is easy to migrate to a new database.

In Hibernate, It is achieved by using a dialect to communicate with the database.
The database can be specified using dialect in the Hibernate configuration XML as follows.
<property name="dialect">org.hibernate.dialect.MySQL</property> or <property name="dialect">org.hibernate.dialect.Oracle</property>

Following code works for all databases
Session.CreateQuery("SELECT E.id FROM Employee E ORDER BY E.id ASC").SetMaxResults(10).List();

In JDBC, consider the following scenario.
You need to fetch the first 10 entries of a table. How this is implemented in different databases is explained below
Mysql: SELECT column_name FROM table_name ORDER BY column_name ASC LIMIT 10;
Sql Server: SELECT TOP 10 column_name FROM table_name ORDER BY column_name ASC;

5) Hibernate minimize the code changes when we add a new column in the database table. The changes to be done is given below.

JDBC:

1. *You have to add a new field into your POJO class.*
2. *Change your JDBC method that performs the "SELECT" to include the new column.*
3. *Change your JDBC method that performs the "INSERT" to add a new value into the new column.*
4. *Change your JDBC method that performs the "UPDATE" to update an existing value in your new column.*

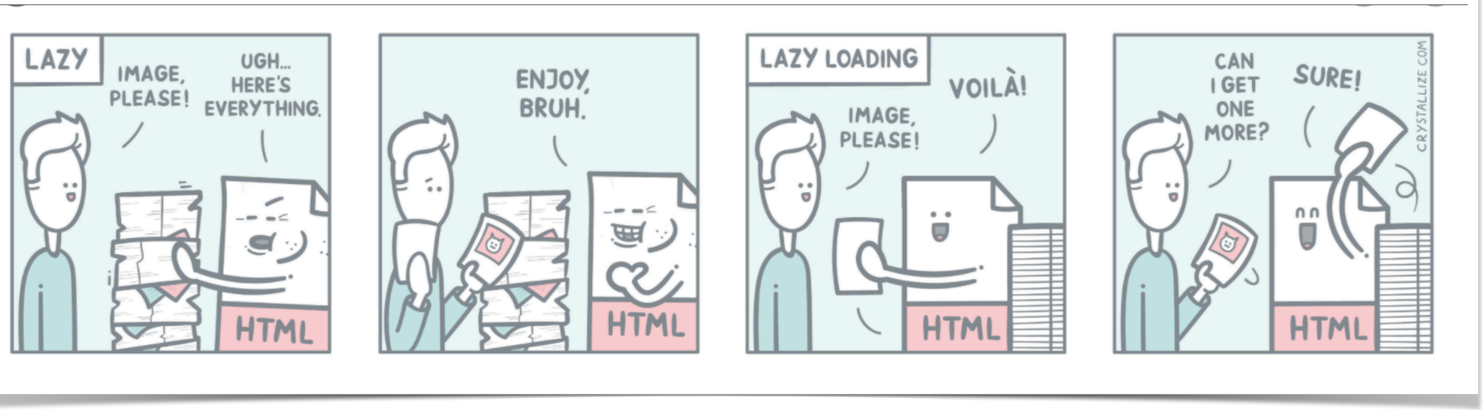
HIBERNATE:

1. *You have to add a new field into your POJO class.*
2. *Modify the Hibernate XML mapping file to include the new column*

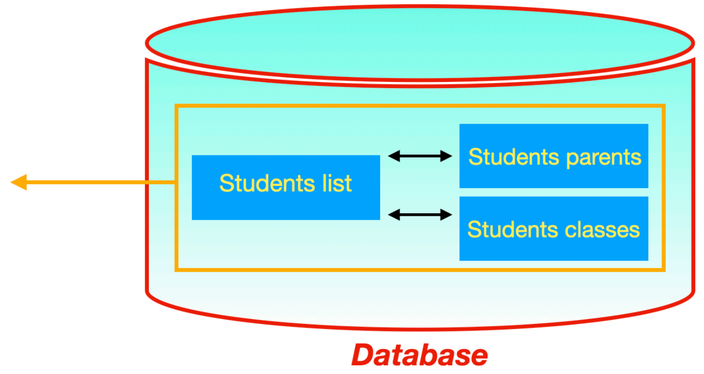
6) Hibernate reduces the amount of repeating lines of code which you can often find with JDBC.
In JDBC orange lines are repeated.

```
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con = DriverManager.getConnection("jdbc : oracle : thin : @localhost : 1521 / orcl", "hr", "oracle");
Statement st = con.createStatement();
st.setInt(1,101);
st.setString(2,"Ratan");
st.executeUpdate();
con.close();
```

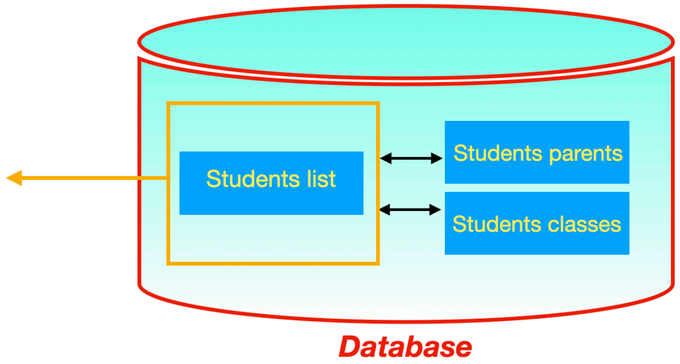
7) We can achieve lazy loading of data by using Hibernate so data can be loaded just when it is needed.



Without lazy loading, when you select Students list, Students parents and Students classes will come as well and fetching data takes time.



If you use lazy loading, when you select Students list, just Students list comes others do not.
So fetching data happens faster



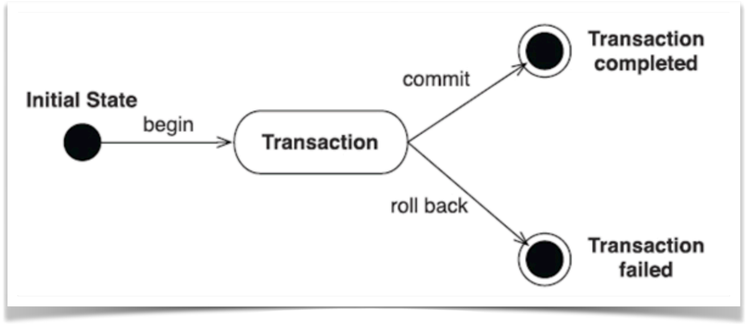
8) JDBC will throw SQL Exception which is a checked exception. So you will be writing "try-catch" blocks in your code. Hibernate handles this by converting all JDBC exceptions to unchecked exceptions. Therefore, you do not need waste your time in implementing this try catch blocks.

9) In JDBC,

If a transaction is a success you need to commit ==> con.commit()
If transaction is failed you have to rollback ==> con.rollback();

In Hibernate,

You don't have to commit and rollback these transactions as it is implicitly provided.



Transaction: *A transaction is a group of operation to be performed under one task.*

If all the operations in a group are a success then the task is finished and the transaction is successfully completed.

If one of the operation is failed then the whole task is failed and hence the transaction is also failed.

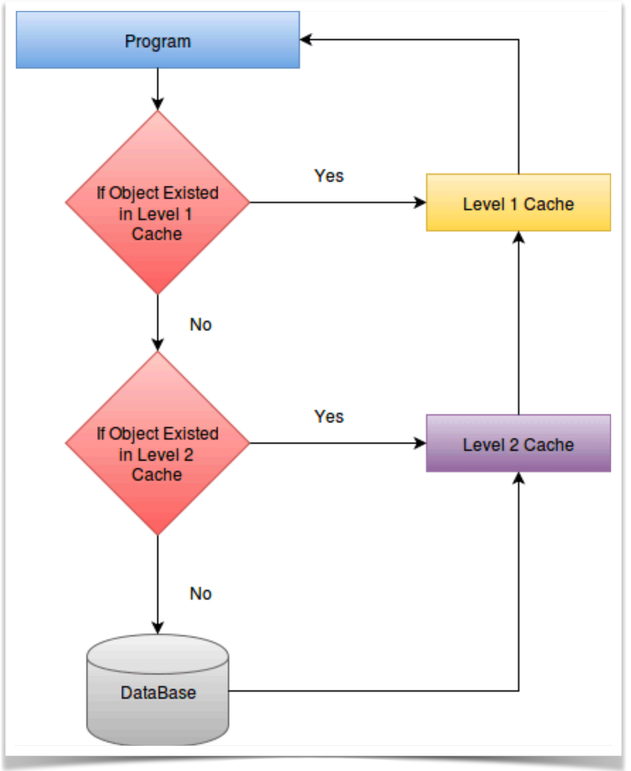
Transaction ==> Deposit money —> Update balance —> Withdraw
If Deposit money fails then Update balance and Withdraw fails.
If Update balance fails Withdraw fails.

***10) Hibernate** provides **caching mechanism which helps to reduce the number of hits** as much as possible that your application makes with the database server. This will improve the performance of your application.*

Hibernate stores the object in session which is available till the transaction is active. When a particular query is executed repeatedly, the value stored in the session is used.

When a new transaction begins, the object is fetched again from the database and is stored in the session.

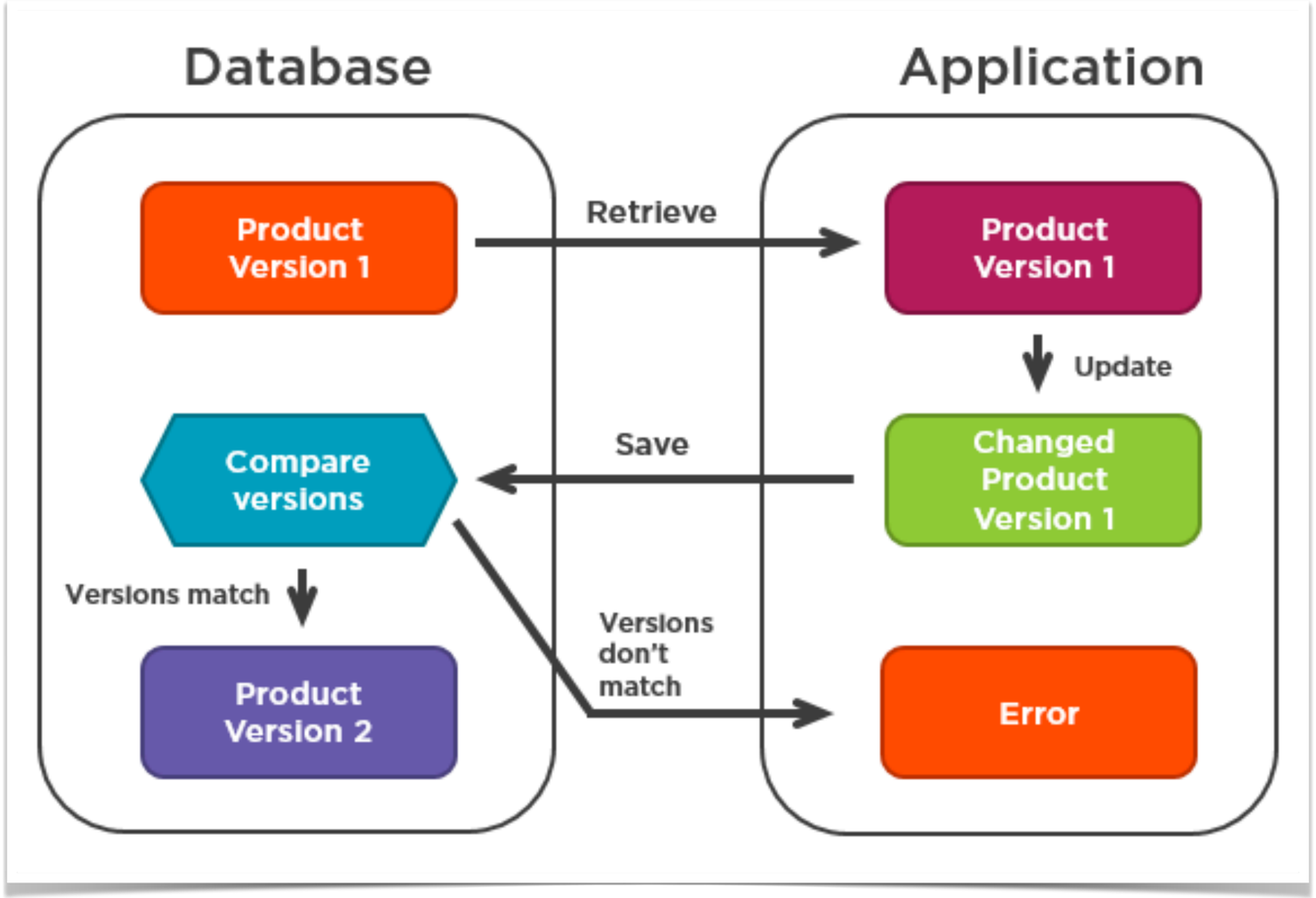
*There is **no such caching mechanism** available **in JDBC**.*



11) Hibernate enables the developer to define version type field to an application which is updated when the data is updated every time.

If two different users retrieve same data and then modify it and one user saved his modified data to the database before the other user, the version is updated. Now when the other user tries to saves his data, it will not allow to save it because this data is not an updated data.

In JDBC this check should be performed by the developer.



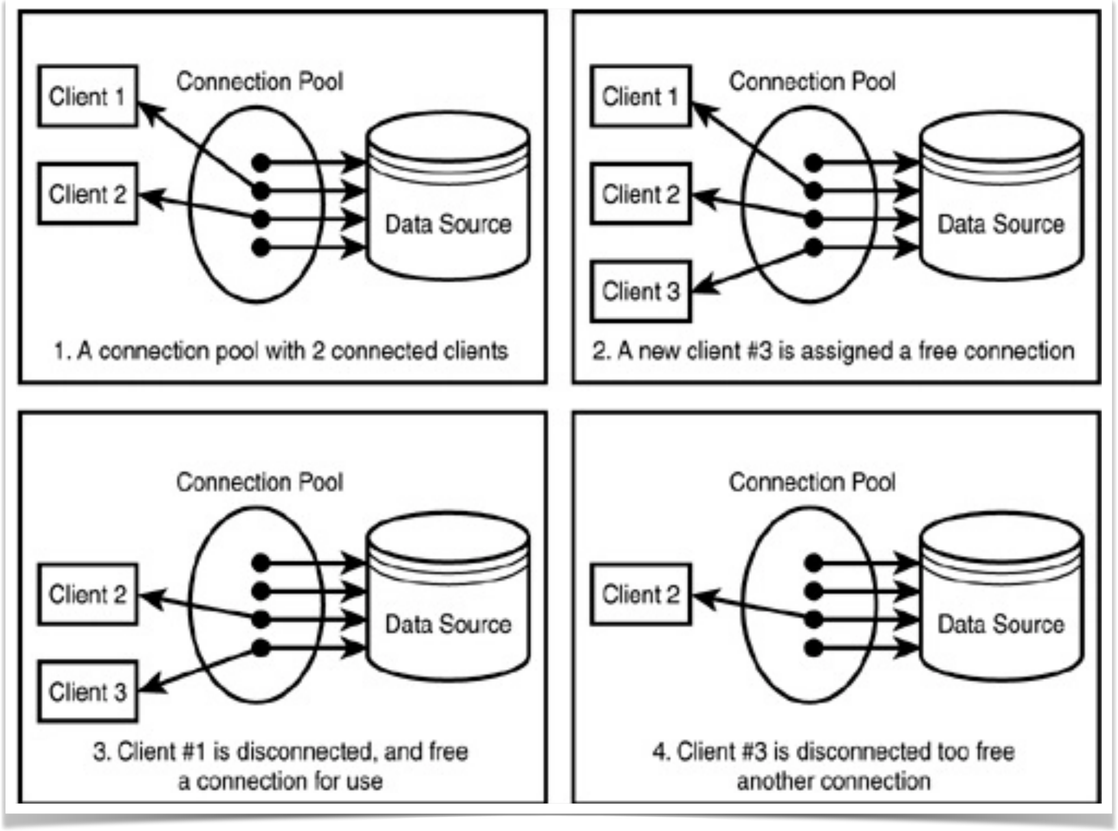
12) In an enterprise application, the application will be hit by numerous users.

If the application server establishes a new connection for every request, it will be a burden on the database server.
On the other hand, if there is a single database connection only, there will be a huge overhead of requests for query.

Hence, it is preferred to have a limited number of database connections pre-configured in Hibernate.
Hibernate framework ensures that new connections are established until the defined maximum limit is reached.

Post the limit, Hibernate reuses the database connection object.

Thus, Hibernate connection pool is a bunch of Database connection objects created for managing parallel communication with the database



What is Hibernate?

1) Hibernate is a i) Free
ii) Open Source Object-Relational Mapping(ORM) tool for Java
iii) It is designed to map Objects to an RDBMS
iv) It implements the Object-Oriented Programming concepts in a Relational Database.

2) When we type Java codes we focus on "objects", "classes", "variables", "methods" etc.
but databases use "tables", "columns", "rows".
As you see there are 2 different worlds, Hibernate is a connector to create connection between these two worlds.

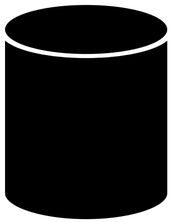
3) While creating connection between Java codes and database, Hibernate maps OOP Concept with the Database Concept

4) By using Hibernate, we can create database, we can fetch, save and modify data.

Note: *Hibernate is an ORM Tool.*
We have different ORM Tools like iBatis or TopLink but most common one is Hibernate

Application
Java Codes
Data: Variables and Objects

Hibernate



Database
Data: Tables, Records and Fields

```
class Students{
    private int id;
    private String name;
    private int accountNo;
}
```

ID	NAME	ACCOUNTNO
1001	Ali	9876543
1002	Mary	9876544

Students Table

```
Students student1 = new Students();
student1.id = 1001;
student1.name = Ali;
student1.accountNo = 9876543;
```

```
Students student2 = new Students();
student2.id = 1002;
student2.name = Mary;
student2.accountNo = 9876544;
```

Class ==> Table
Object ==> Row
Variable Names ==> Column Names
Values of Variables ==> Data

HIBERNATE Implementation Steps

1) To save an object into database we need save(<object name>) method like save(student1)

2) To fetch an object from database we need get() method

3) To be able to use save(<object name>) or get(<Class name>, <Serializable id values>) we need to create an object of Session
   Session: Session is an Interface.
   We create an object whose data type is Session Interface by typing "Session session = sf.openSession();"
   It creates communication between the Java Application and the Database Persistence Store.
   This object is used for CRUD Operations in Hibernate

4) To be able to create an object of Session we need to create an object of SessionFactory by "SessionFactory sf = con.buildSessionFactory();"
   Session Factory: As the name indicates, this interface manages for the creation and destruction of Session Objects.

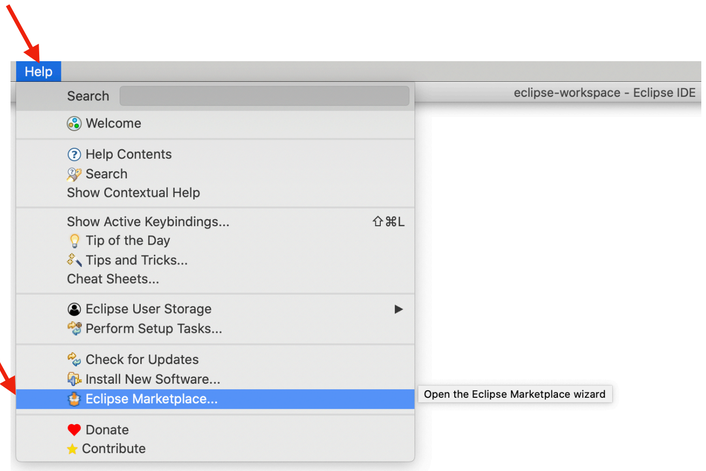
5) To be able to send data from Java Application to Database we need to begin Transaction by typing "Transaction tx = session.beginTransaction();"
   Transaction: Transaction is an Interface.
   A transaction takes care of fetching the data from Java application and delivering it to the database in the form of a query.
   A Transaction is a short-lived thread that is destroyed after every database transaction.

6) To be able to create an object of SessionFactory we need to provide configuration of database.

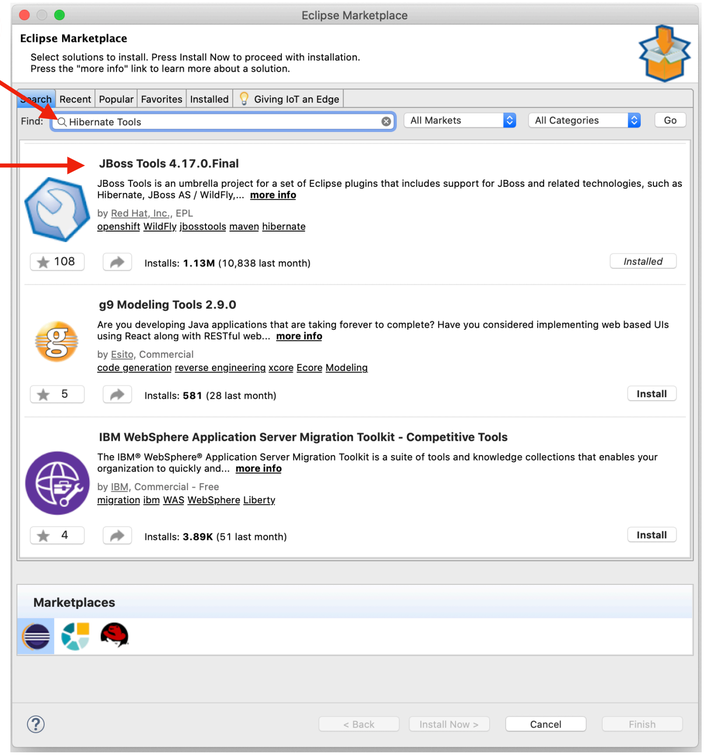
7) To configure database we need to set driver name, database url, username, password etc.

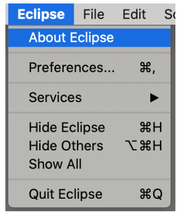
How to add Hibernate Plugin in Eclipse?

1)



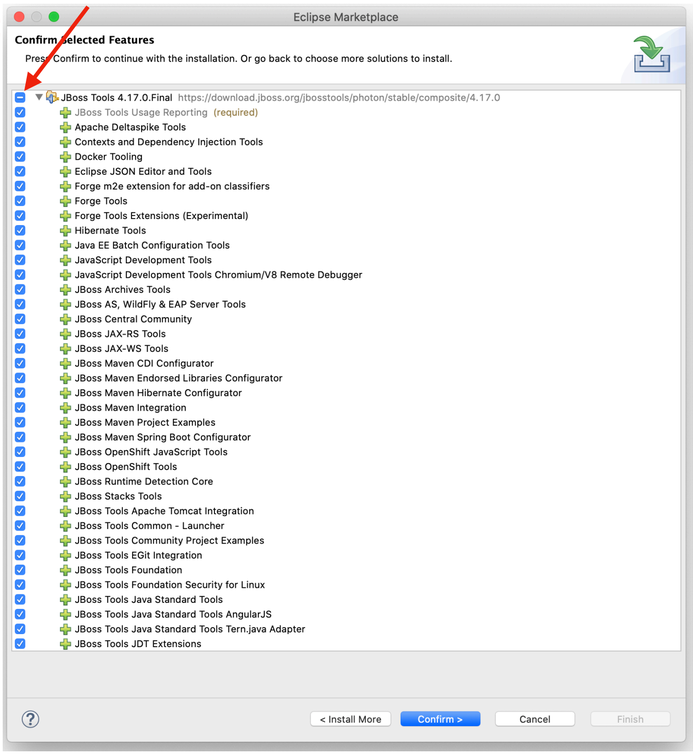
2)





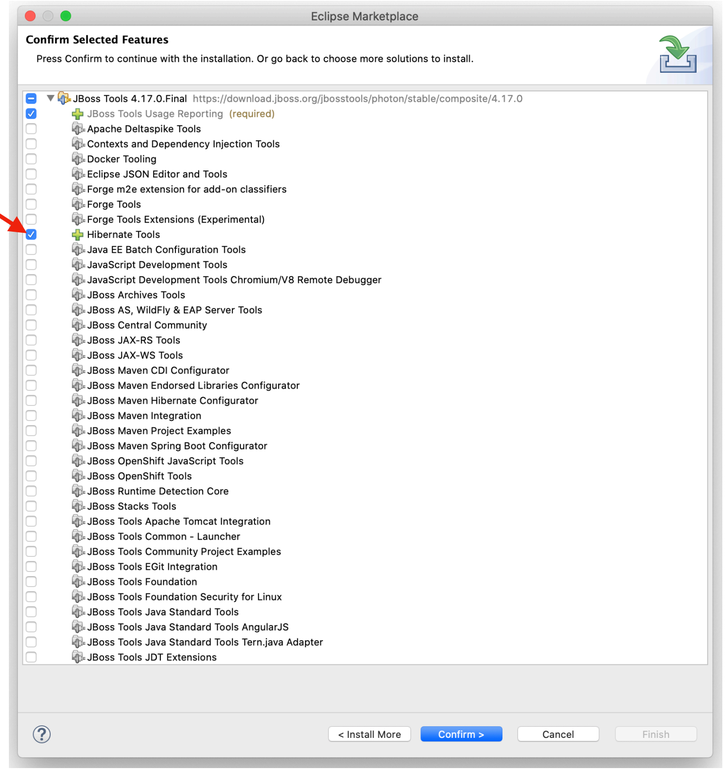
Check your Eclipse version, your Eclipse version should match with the JBoss Tools version. To see Eclipse version click on "Eclipse" then "About Eclipse"

3)

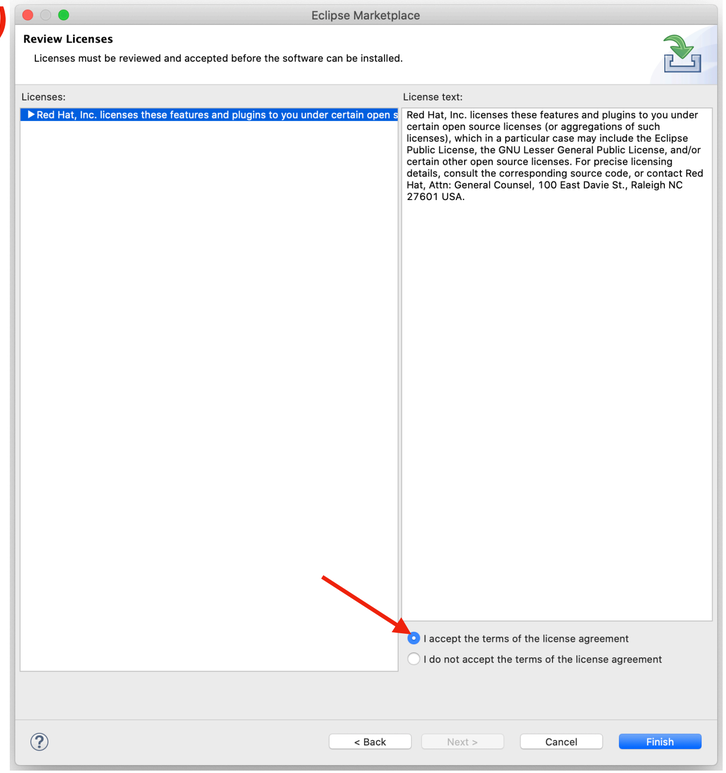


To unselect all checkboxes click on the first checkbox

4)


Eclipse Marketplace
Confirm Selected Features
Press Confirm to continue with the installation. Or go back to choose more solutions to install.
JBoss Tools 4.17.0.Final https://download.jboss.org/jbosstools/photon/stable/composite/4.17.0
JBoss Tools Usage Reporting (required)
Apache Deltaspike Tools
Contexts and Dependency Injection Tools
Docker Tooling
Eclipse JSON Editor and Tools
Forge m2e extension for add-on classifiers
Forge Tools
Forge Tools Extensions (Experimental)
Hibernate Tools
Java EE Batch Configuration Tools
JavaScript Development Tools
JavaScript Development Tools Chromium/V8 Remote Debugger
JBoss Archives Tools
JBoss AS, WildFly & EAP Server Tools
JBoss Central Community
JBoss JAX-RS Tools
JBoss JAX-WS Tools
JBoss Maven CDI Configurator
JBoss Maven Endorsed Libraries Configurator
JBoss Maven Hibernate Configurator
JBoss Maven Integration
JBoss Maven Project Examples
JBoss Maven Spring Boot Configurator
JBoss OpenShift JavaScript Tools
JBoss OpenShift Tools
JBoss Runtime Detection Core
JBoss Stacks Tools
JBoss Tools Apache Tomcat Integration
JBoss Tools Common - Launcher
JBoss Tools Community Project Examples
JBoss Tools EGit Integration
JBoss Tools Foundation
JBoss Tools Foundation Security for Linux
JBoss Tools Java Standard Tools
JBoss Tools Java Standard Tools AngularJS
JBoss Tools Java Standard Tools Tern.java Adapter
JBoss Tools JDT Extensions
< Install More
Confirm >
Cancel
Finish

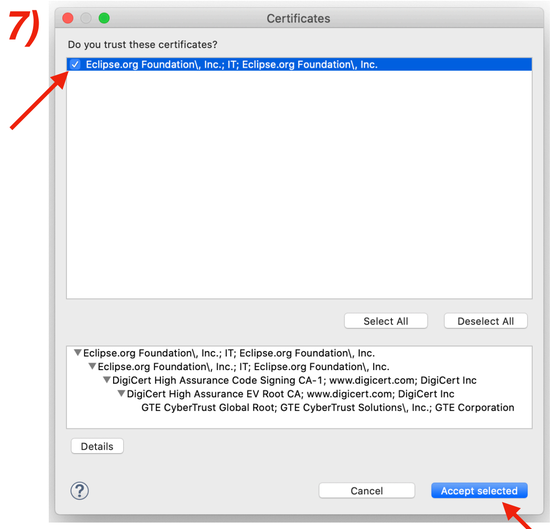
5)


Eclipse Marketplace
Review Licenses
Licenses must be reviewed and accepted before the software can be installed.
Licenses:
Red Hat, Inc. licenses these features and plugins to you under certain open s
License text:
Red Hat, Inc. licenses these features and plugins to you under certain open source licenses (or aggregations of such licenses), which in a particular case may include the Eclipse Public License, the GNU Lesser General Public License, and/or certain other open source licenses. For precise licensing details, consult the corresponding source code, or contact Red Hat, Attn: General Counsel, 100 East Davie St., Raleigh NC 27601 USA.
I accept the terms of the license agreement
I do not accept the terms of the license agreement
< Back
Next >
Cancel
Finish

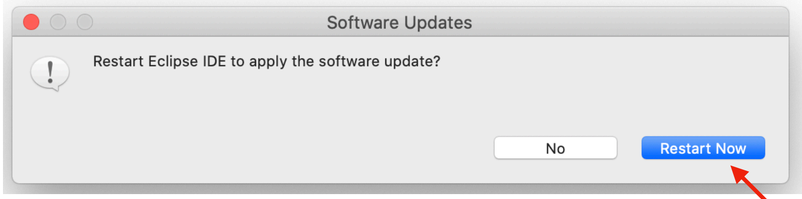
6)


Security Warning
Warning: Installing unsigned software for which the authenticity or validity cannot be established. Continue with the installation?
Install anyway
Details >>
Cancel

7)

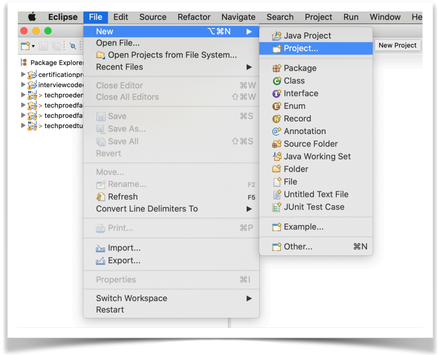

Certificates
Do you trust these certificates?
Eclipse.org Foundation\, Inc.; IT; Eclipse.org Foundation\, Inc.
Select All
Deselect All
Eclipse.org Foundation\, Inc.; IT; Eclipse.org Foundation\, Inc.
Eclipse.org Foundation\, Inc.; IT; Eclipse.org Foundation\, Inc.
DigiCert High Assurance Code Signing CA-1; www.digicert.com; DigiCert Inc
DigiCert High Assurance EV Root CA; www.digicert.com; DigiCert Inc
GTE CyberTrust Global Root; GTE CyberTrust Solutions\, Inc.; GTE Corporation
Details
Cancel
Accept selected

8)

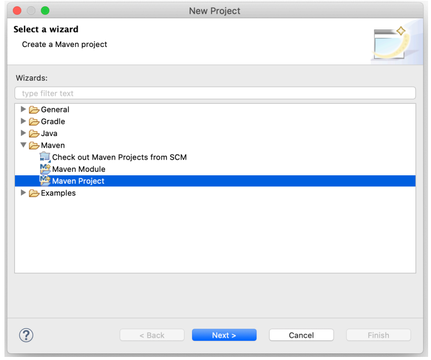

Software Updates
Restart Eclipse IDE to apply the software update?
No
Restart Now

How to create a Maven Project for Hibernate?

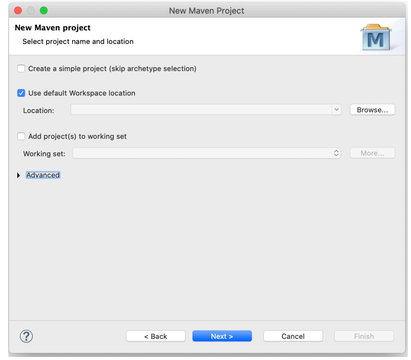
1)



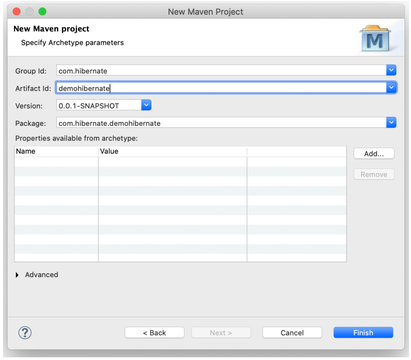
2)



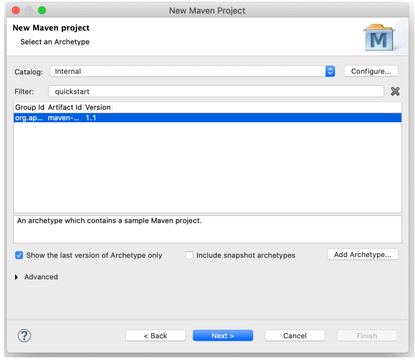
3)



4)



5)



Which dependencies to add to pom.xml file?

For SQL Connection

```
<dependency>
        <groupId>io.syndesis.connector</groupId>
        <artifactId>connector-sql</artifactId>
        <version>1.10.0</version>
        <scope>runtime</scope>
</dependency>
```

For Oracle JDBC

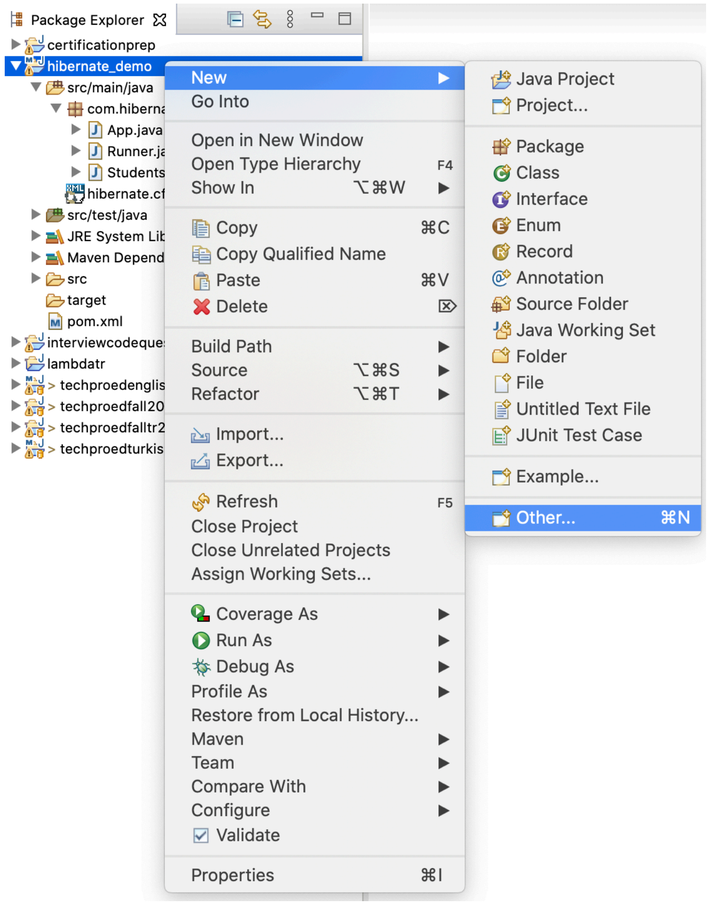
```
<dependency>
        <groupId>com.oracle.database.jdbc</groupId>
        <artifactId>ojdbc10</artifactId>
        <version>19.8.0.0</version>
    </dependency>
```

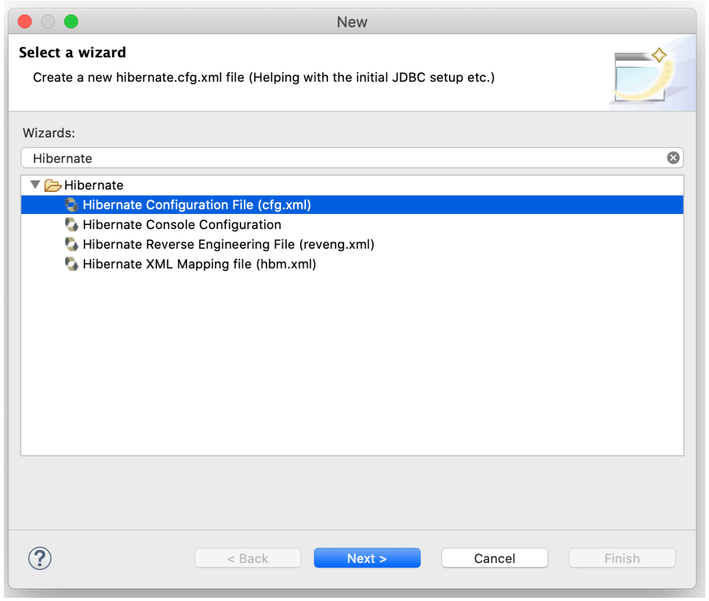
For Hibernate jar

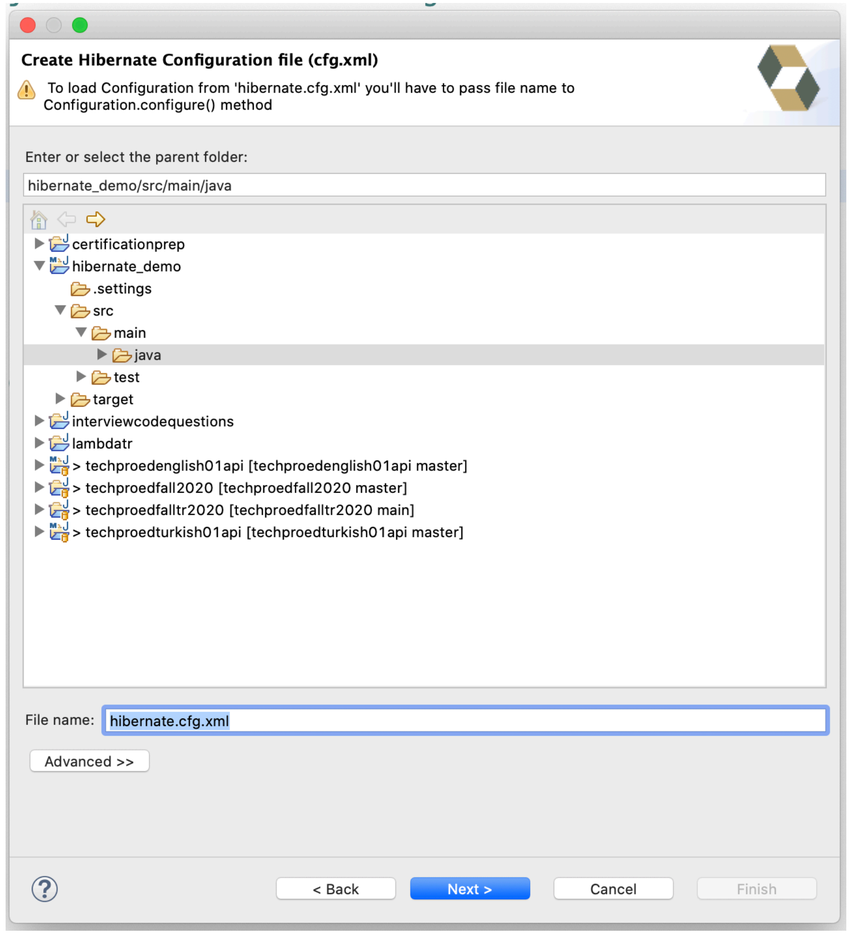
```
<dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-core</artifactId>
        <version>5.4.23.Final</version>
</dependency>
```

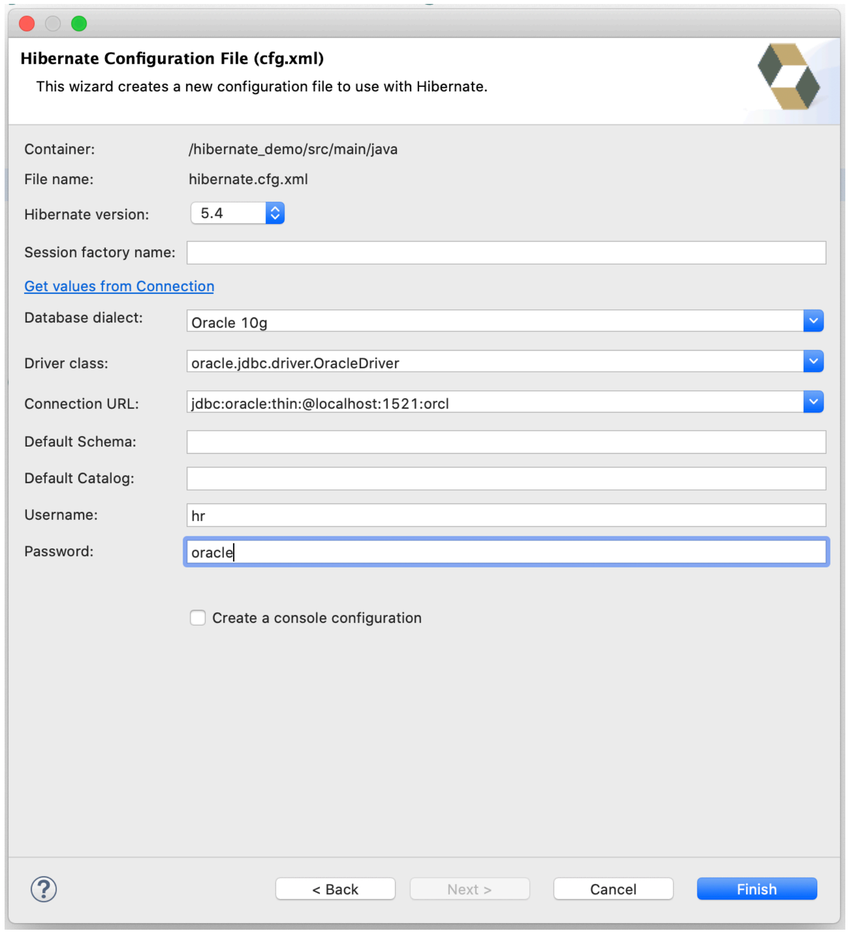
***Note:** After adding dependencies do not forget to save and update the project…*

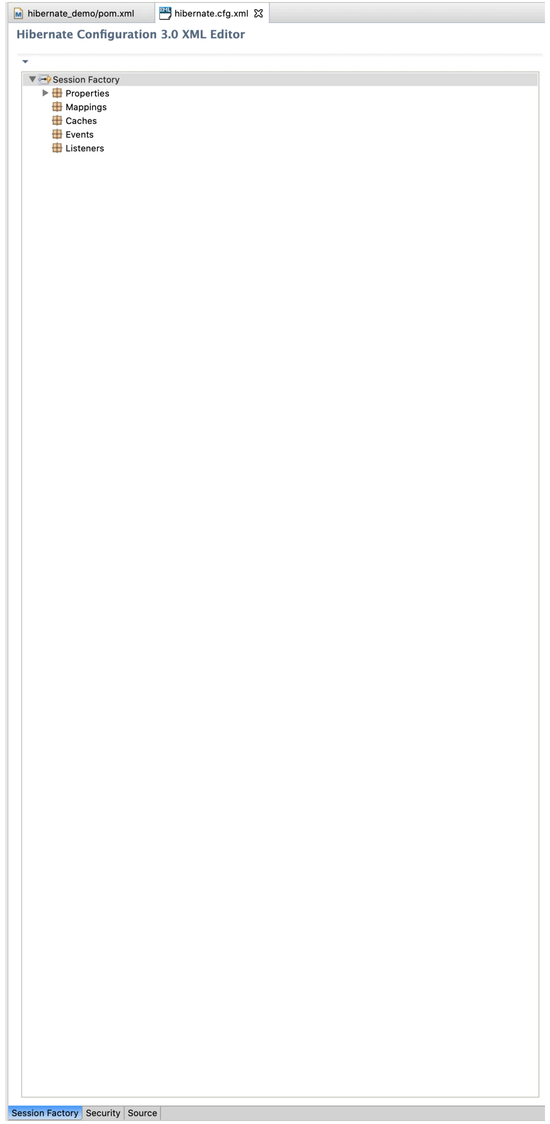
How to create hibernate.cfg.xml file?






Create Hibernate Configuration file (cfg.xml)
To load Configuration from 'hibernate.cfg.xml' you'll have to pass file name to Configuration.configure() method
Enter or select the parent folder:
hibernate_demo/src/main/java
certificationprep
hibernate_demo
.settings
src
main
java
test
target
interviewcodequestions
lambdatr
> techproedenglish01api [techproedenglish01api master]
> techproedfall2020 [techproedfall2020 master]
> techproedfalltr2020 [techproedfalltr2020 main]
> techproedturkish01api [techproedturkish01api master]
File name:
hibernate.cfg.xml
Advanced >>
< Back
Next >
Cancel
Finish


Hibernate Configuration File (cfg.xml)
This wizard creates a new configuration file to use with Hibernate.
Container:
/hibernate_demo/src/main/java
File name:
hibernate.cfg.xml
Hibernate version:
5.4
Session factory name:
Get values from Connection
Database dialect:
Oracle 10g
Driver class:
oracle.jdbc.driver.OracleDriver
Connection URL:
jdbc:oracle:thin:@localhost:1521:orcl
Default Schema:
Default Catalog:
Username:
hr
Password:
oracle
Create a console configuration
< Back
Next >
Cancel
Finish


hibernate_demo/pom.xml
hibernate.cfg.xml
Hibernate Configuration 3.0 XML Editor
Session Factory
Properties
Mappings
Caches
Events
Listeners
Session Factory
Security
Source

This is the Service Name which you created for SQL

hibernate_demo/pom.xml | hibernate.cfg.xml

```
 1 <?xml version="1.0" encoding="UTF-8"?>
 2 <!DOCTYPE hibernate-configuration PUBLIC
 3         "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
 4         "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
 5 <hibernate-configuration>
 6     <session-factory>
 7         <property name="hibernate.connection.driver_class">oracle.jdbc.driver.OracleDriver</property>
 8         <property name="hibernate.connection.password">oracle</property>
 9         <property name="hibernate.connection.url">jdbc:oracle:thin:@localhost:1521:orcl</property>
10         <property name="hibernate.connection.username">hr</property>
11         <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12     </session-factory>
13 </hibernate-configuration>
14
```

This is the username which you created for SQL

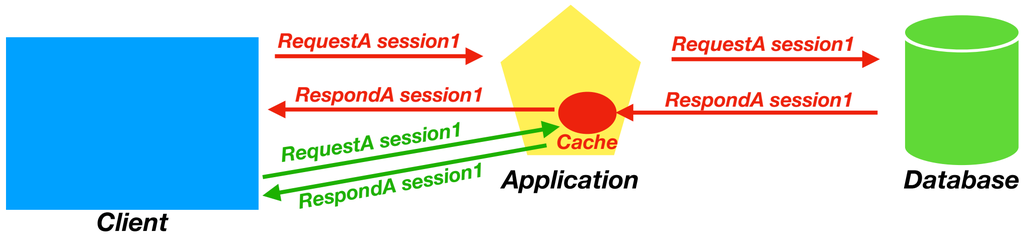
Remove “:” and put “/”

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

Type the above code between the line 11 and line 12.
If you type “”update” it adds new data to the table
If you type “create” it creates the table from scratch in every run

Hibernate Caching

What is Cache?



Caching is a mechanism to enhance the performance of a system.
Cache memory stores recently used data items in order to reduce the number of database hits as much as possible.

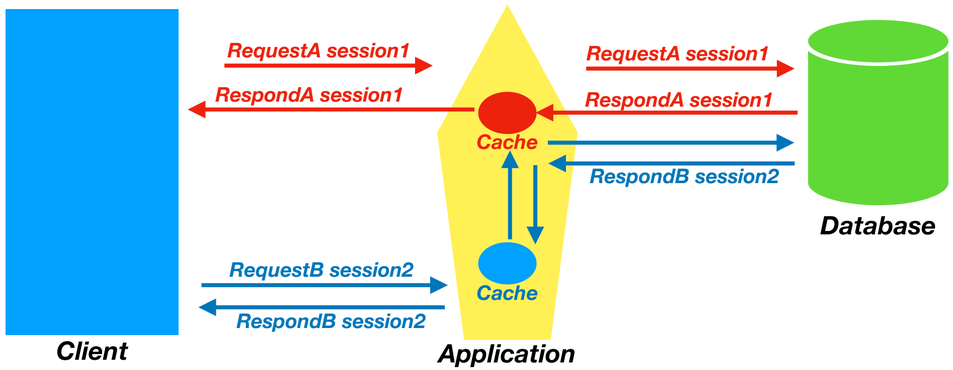
First-Level Cache

The first-level cache is the session cache and is a mandatory cache through which all requests must pass.

If you close the session, all the objects being cached are lost and either persisted or updated in the database.

Hibernate uses first-level cache by default and you have nothing to do to use first-level cache.

Second-Level Cache



The Second Level cache is an optional cache and first-level cache will always be consulted before any attempt is made to locate an object in the second-level cache.

The Second Level case is responsible for caching objects across sessions.

How to configure Second Level Cache?

We need to use third party libraries like ehcache or OS or Swam
We will use ehcache in this course

To configure ehcahce

1) Configure pom.xml file
    a) Add ehcahce dependency (This dependency provides the features of ehcache)
    b) Add Hibernate-ehcache dependency (This dependency provides the integration of ehcache and hibernate)
2) Configure hibernate.cfg.xml file
    As default Second Level Cache is disabled, if we want to use we have to enable it from the hibernate.cfg.xml file by adding properties.
3) Change @Entity class
    Because by default Entity is not cachable.
    You need to add 2 annotations a)@cachable (It makes the Entity cachable)
    b)@cache (It allows us to add caching strategies)

What is JPA (Java Persistence API) ?

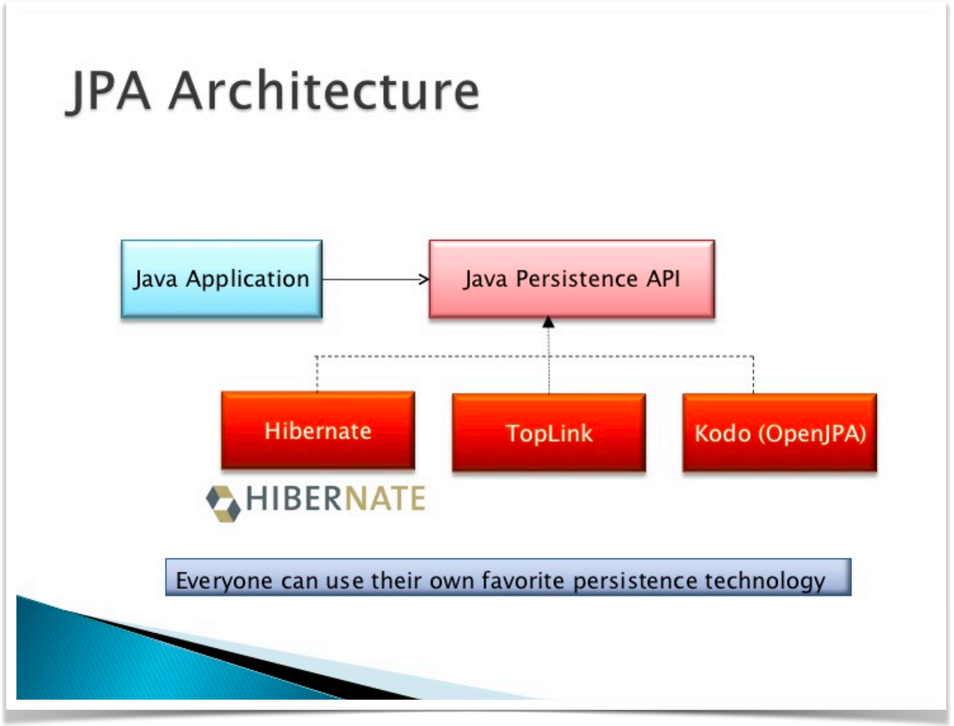
The Java Persistence API (JPA) is a specification of Java.

It is used to persist data between Java Object and Relational Database.

JPA acts as a bridge between object-oriented domain models and relational database systems.

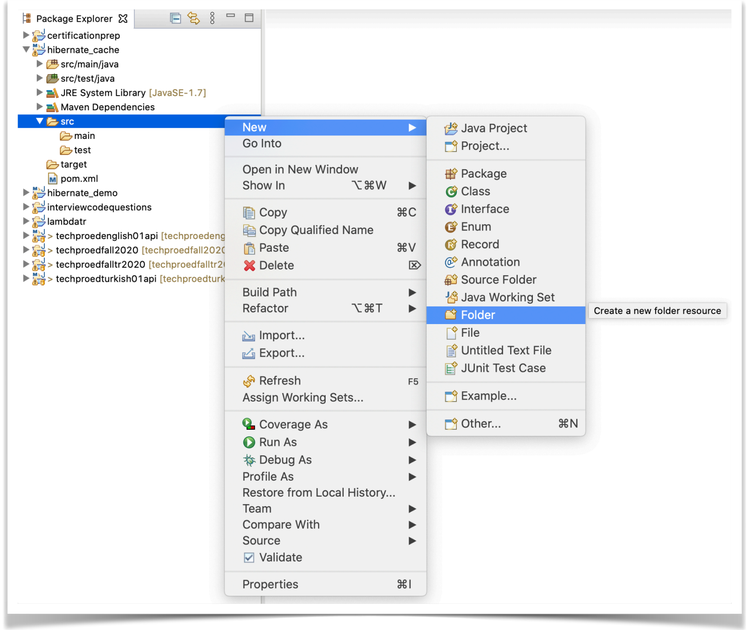
As JPA is just a specification, it doesn't perform any operation by itself.

It requires an implementation. So, ORM tools like Hibernate, TopLink and iBatis implements JPA specifications for data persistence.

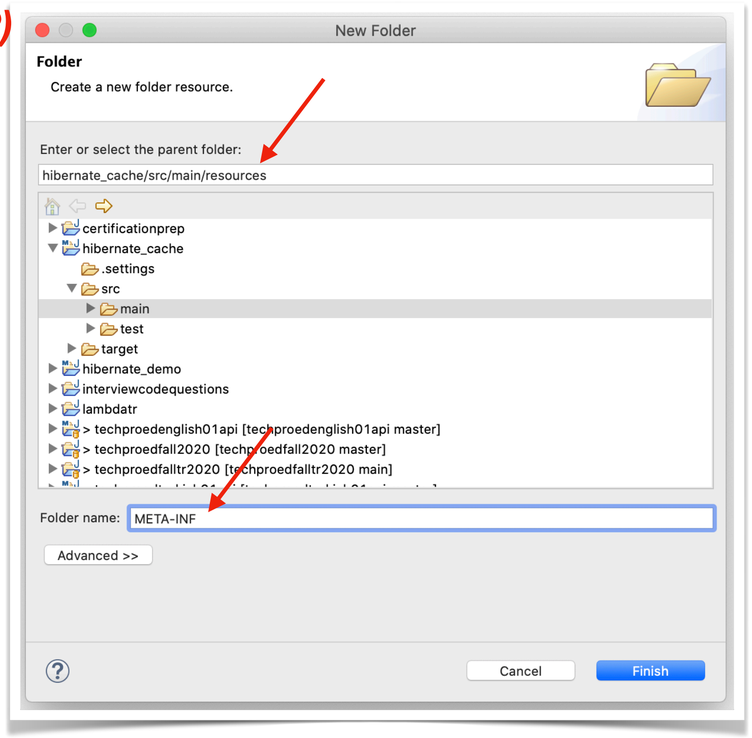


How to create persistence.xml file in Eclipse?

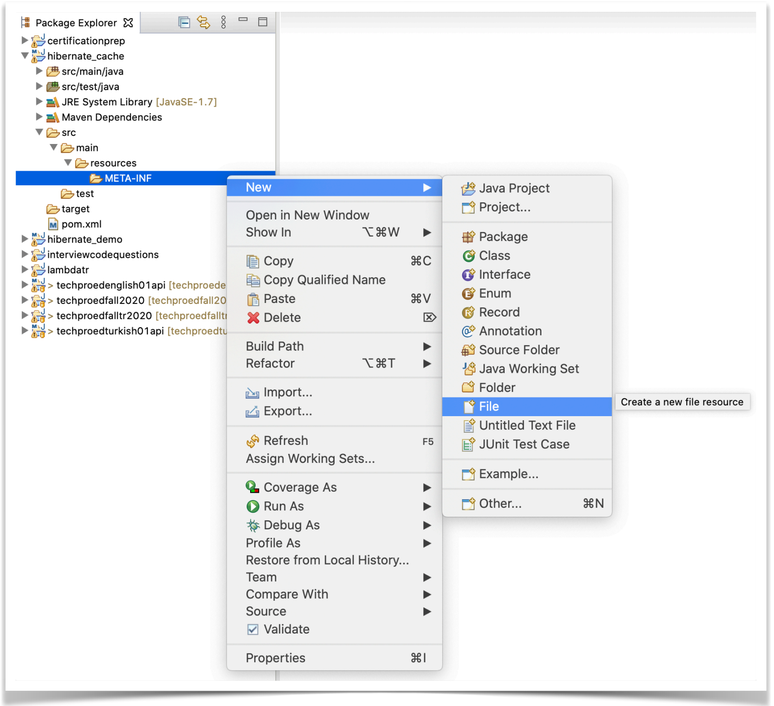
1)



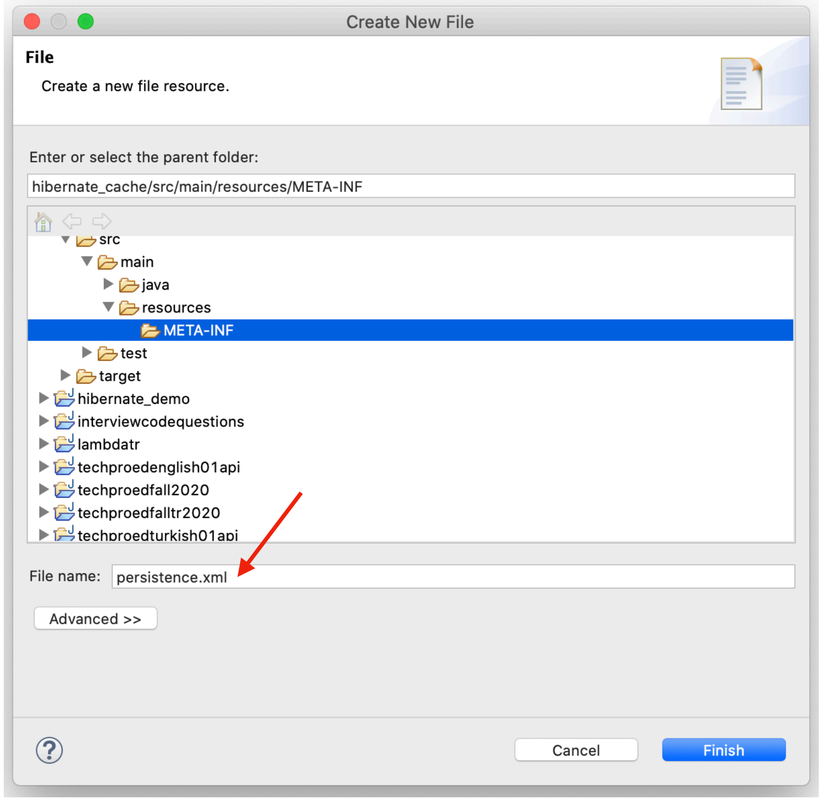
2)



3)



4)



Type the following codes into your persistence.xml file

5)

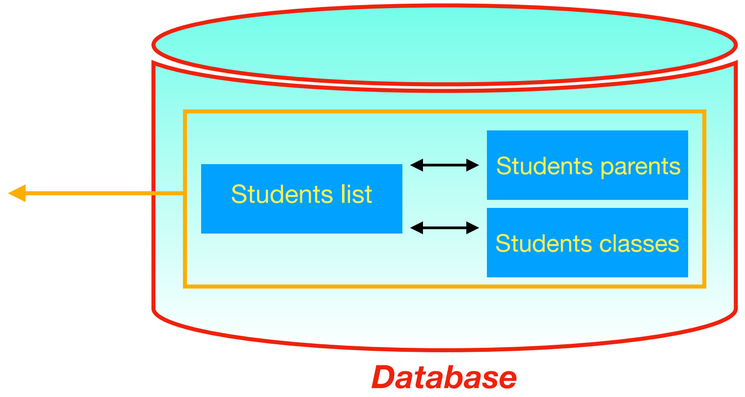
```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1"
   xmlns="http://xmlns.jcp.org/xml/ns/persistence"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
   <persistence-unit name="pu" transaction-type="RESOURCE_LOCAL">
   <class>Students13</class>
   <properties>
     <property name="javax.persistence.jdbc.driver" value="oracle.jdbc.driver.OracleDriver"/>
     <property name="javax.persistence.jdbc.url" value="jdbc:oracle:thin:@localhost:1521/orcl"/>
     <property name="javax.persistence.jdbc.user" value="hr"/>
     <property name="javax.persistence.jdbc.password" value="oracle"/>
   </properties>
 </persistence-unit>

</persistence>
```

Use your own class name, url, username, and password

7) We can achieve lazy loading of data by using Hibernate so data can be loaded just when it is needed.

Without lazy loading, when you select Students list, Students parents and Students classes will come as well and fetching data takes time.



YEDEK!!!!!!!

If you use lazy loading, when you select Students list, just Students list comes others do not.
So fetching data happens faster