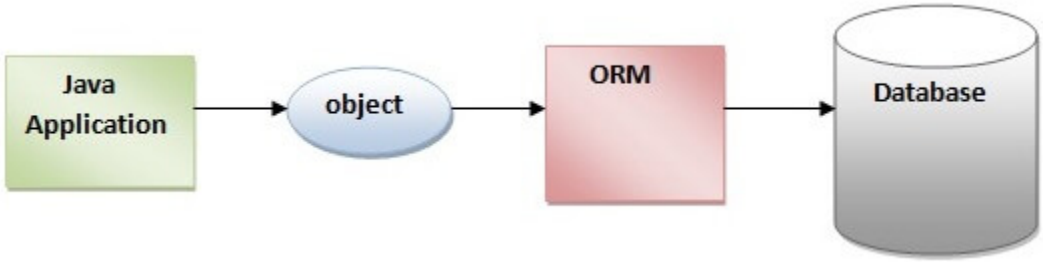
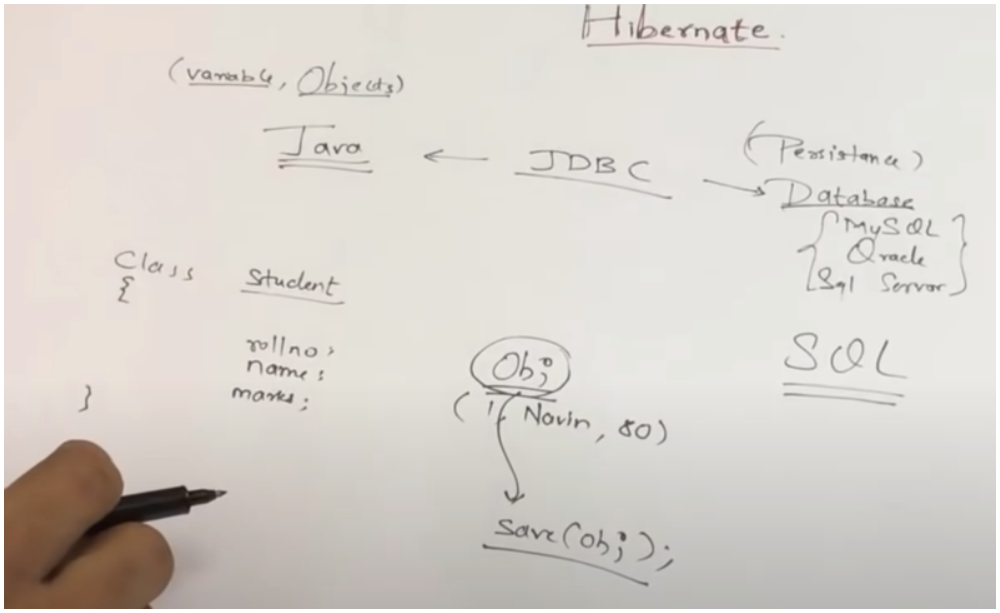
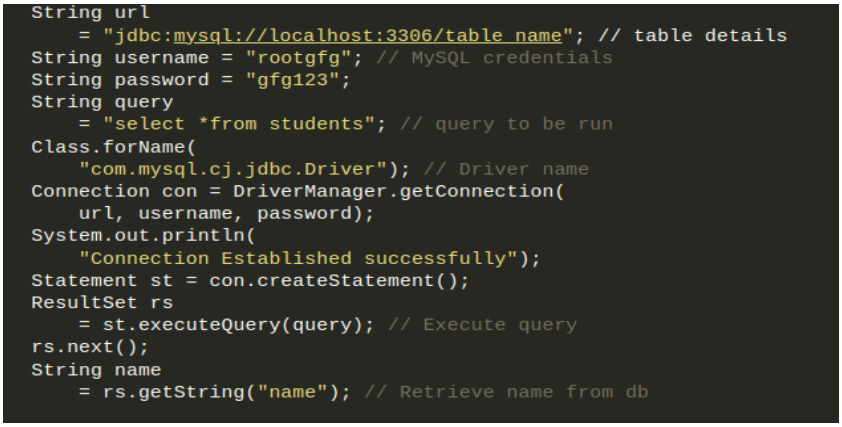
- Hibernate
    - Hibernate is one of the best ORM (Object Relational Mapping) frameworks available in the market. Hibernate is a Java framework that simplifies the development of Java applications to interact with the database. It is an open-source, lightweight, ORM (Object Relational Mapping) tool. Hibernate implements the specifications of JPA (Java Persistence API) for data persistence.
- By using Hibernate, we directly access the database without writing SQL queries. If you don't want to execute SQL query, that case your ORM framework comes to the picture.
- ORM Tool
    - An ORM tool simplifies the data creation, data manipulation, and data access. It is a programming technique that maps the object to the data stored in the database.

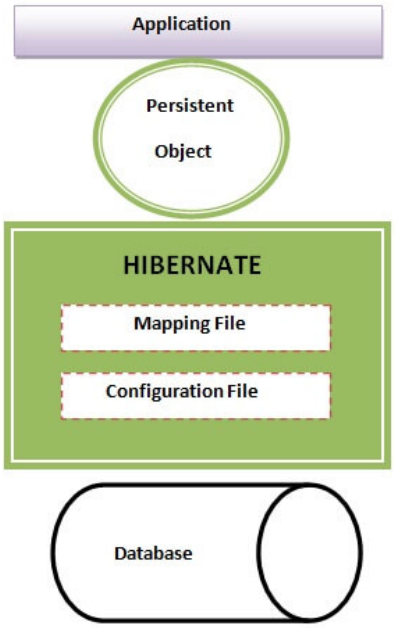



- Suppose we have one class Student and it contains some variable, here class represents as table name and the variable represents an attribute in the table.
- Usually, when we want to connect Java applications with databases we need "JDBC" that connects both, when we need to deal with databases we need to know SQL language, everyone is not comfortable with SQL, so that case ORM framework comes into the picture.

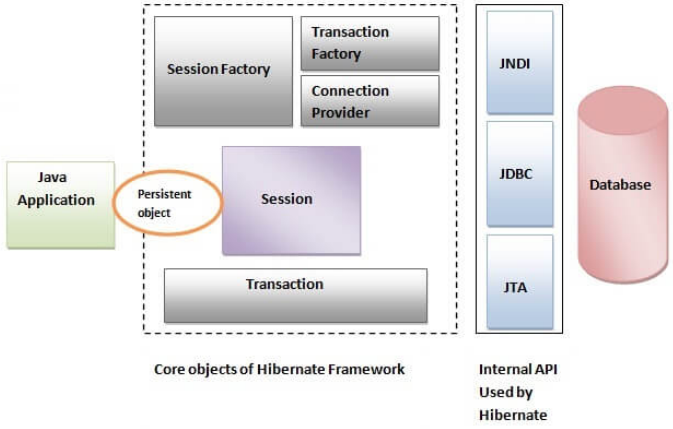



- When we want to save object data into a table, we just need to "save(obj)", that's it.

- To implement ORM, we have ORM tools, one of the tools is Hibernate, IBatis, and Toplinks
- By using the “save()” method, we can save objects into the database, but the question is how to use the save method for that
    - We need to create “Session” obj s → s.save(obj)
    - For creating a session we need a session factory, A session factory contains the configuration of a database, normally database needs to be configured such as the URL, database name, user name, and password of the database.
    - 
```
String url
    = "jdbc:mysql://localhost:3306/table_name"; // table details
String username = "rootgfg"; // MySQL credentials
String password = "gfg123";
String query
    = "select *from students"; // query to be run
Class.forName(
    "com.mysql.cj.jdbc.Driver"); // Driver name
Connection con = DriverManager.getConnection(
    url, username, password);
System.out.println(
    "Connection Established successfully");
Statement st = con.createStatement();
ResultSet rs
    = st.executeQuery(query); // Execute query
rs.next();
String name
    = rs.getString("name"); // Retrieve name from db
```
    - Now using the Session Factory Interface you can Create a Session object and from the session object, you can save the thing into DB, and get data from DB as well.
- Hibernate Architecture:
    - The Hibernate architecture includes many objects such as persistent objects, session factory, transaction factory, connection factory, session, transaction, etc.
    - The Hibernate architecture is categorized into four layers.
        - Java application layer
        - Hibernate framework layer
        - Backhand API layer
        - Database layer




This is the high level architecture of Hibernate with mapping file and configuration file.




- 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).

- Example:

- Let's consider a simple entity class `Book` representing books in a library:

```java
import javax.persistence.*;

@Entity
@Table(name = "books")
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String title;
    private String author;
    private int year;

    // Getters and setters
    // Constructors
}
```

- Next, you'd create a Hibernate configuration file (hibernate.cfg.xml) to set up the database connection and other configurations:

```xml
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <!-- Database connection settings -->
        <property name="hibernate.connection.driver_class">your_database_driver</property>
        <property name="hibernate.connection.url">jdbc:your_database_url</property>
        <property name="hibernate.connection.username">your_username</property>
        <property name="hibernate.connection.password">your_password</property>

        <!-- Mapping the Book entity -->
        <mapping class="com.example.Book"/>
    </session-factory>
</hibernate-configuration>
```

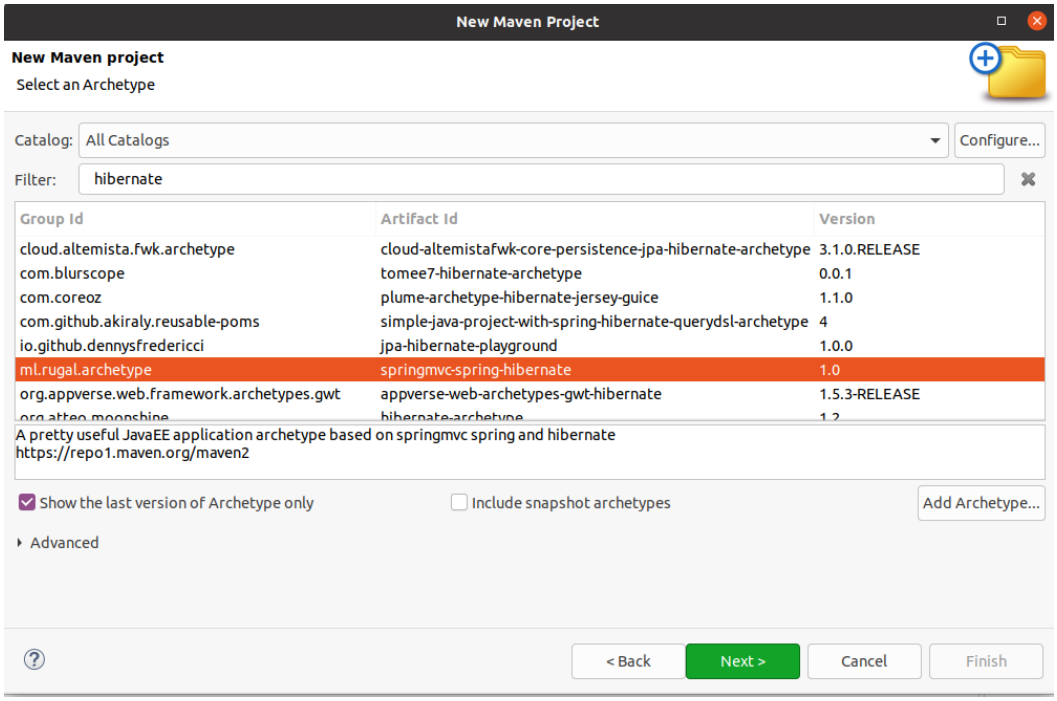
- Now, let's use these entities and configurations to perform CRUD operations using Hibernate:

```java
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

public class Main {
    public static void main(String[] args) {
        // Create a Hibernate configuration instance
        Configuration configuration = new Configuration().configure("hibernate.cfg.xml");

        // Create a SessionFactory
        SessionFactory sessionFactory = configuration.buildSessionFactory();

        // Open a session
        try (Session session = sessionFactory.openSession()) {
            // Begin a transaction
            Transaction transaction = session.beginTransaction();

            // Creating a new Book instance and saving it to the database
            Book book = new Book();
            book.setTitle("Hibernate Guide");
            book.setAuthor("John Doe");
            book.setYear(2020);

            session.save(book); // Save the book

            // Commit the transaction
            transaction.commit();

            // Retrieve a book by its ID
            Book retrievedBook = session.get(Book.class, 1L);
            System.out.println("Retrieved Book: " + retrievedBook.getTitle());

            // Update the retrieved book
            retrievedBook.setTitle("Hibernate Updated Guide");
            transaction = session.beginTransaction();
            session.update(retrievedBook);
            transaction.commit();

            // Delete the book
            transaction = session.beginTransaction();
            session.delete(retrievedBook);
            transaction.commit();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            // Close the session and the session factory
            sessionFactory.close();
        }
    }
}
```

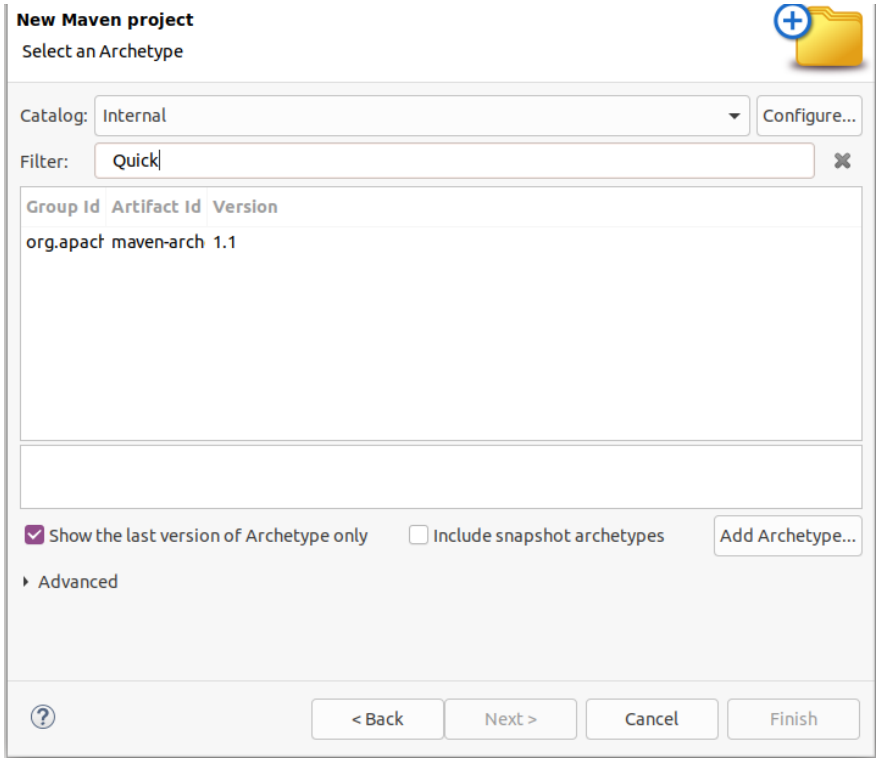
- In this example:
  - We create a SessionFactory using the Hibernate configuration file.
  - Open a Session to perform database operations.
  - Begin a Transaction.
  - Create, read, update, and delete operations (save, get, update, delete) on a Book entity within a transaction.
  - Commit the transaction to persist changes or rollback in case of exceptions.
  - Finally, close the Session and SessionFactory.

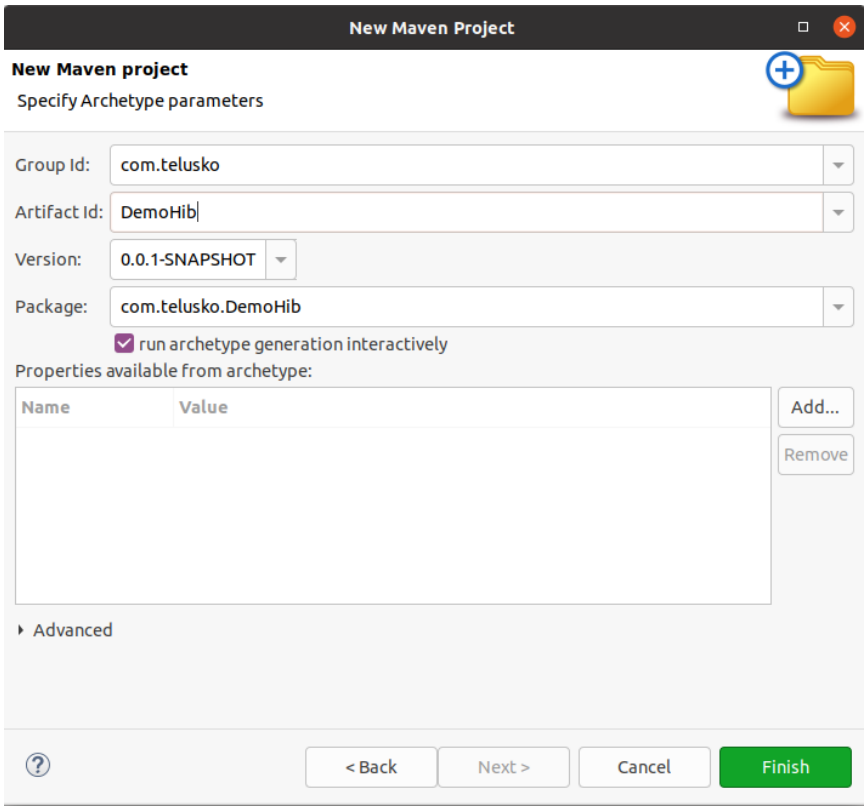
## Implementation

- Open Eclipse and you can create a Normal Project or Maven Project.
- Configure automatically with the standalone application or Web application you can select “All catalog” → hibernate MVC spring:

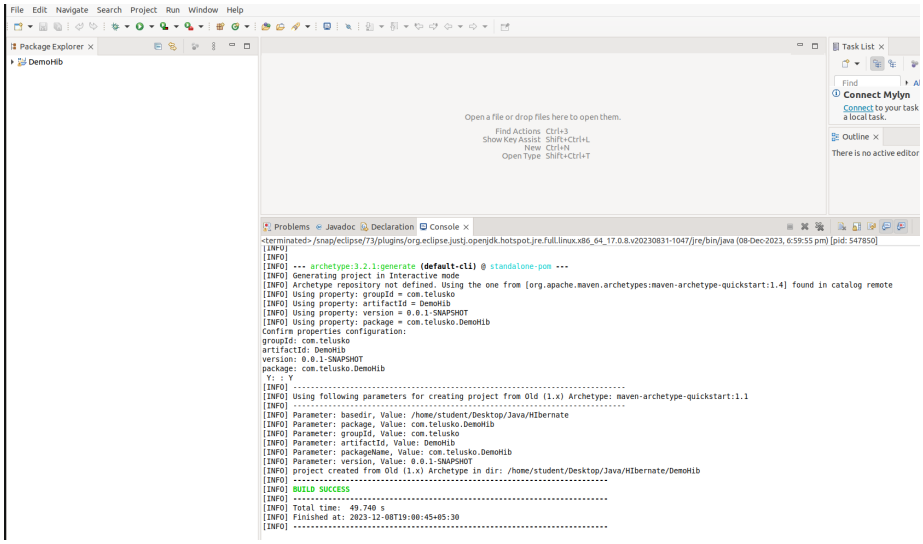


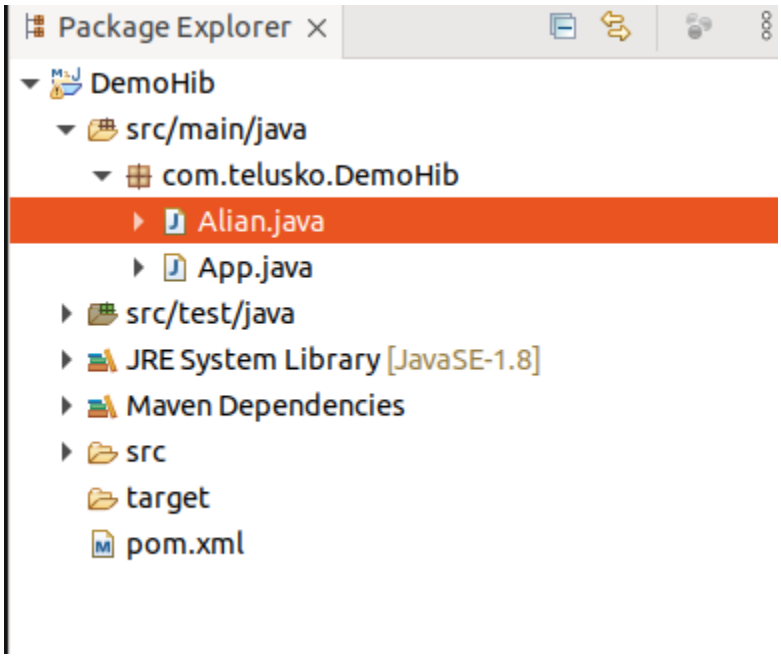

- Here I am using the “Internal” catalog for learning purposes → Quick start.
- Here we are using “Quick Start” which means we developing a core Java application.
    - GroupId → any
    - Artifact Id → Project Name
-

- 

- 

- Once you click on finish, It creates a project, first, it download all dependencies that required for hibernate

- 

- Open that project and create new class “Alian.java”
- 

App.java | Alian.java

```java
package com.telusko.DemoHib;

public class Alian   //pojo
{
    private int aid;
    private String aname;
    private String color;
    public int getAid() {
        return aid;
    }
    public void setAid(int aid) {
        this.aid = aid;
    }
    public String getAname() {
        return aname;
    }
    public void setAname(String aname) {
        this.aname = aname;
    }
    public String getColor() {
        return color;
    }
    public void setColor(String color) {
        this.color = color;
    }


}
```

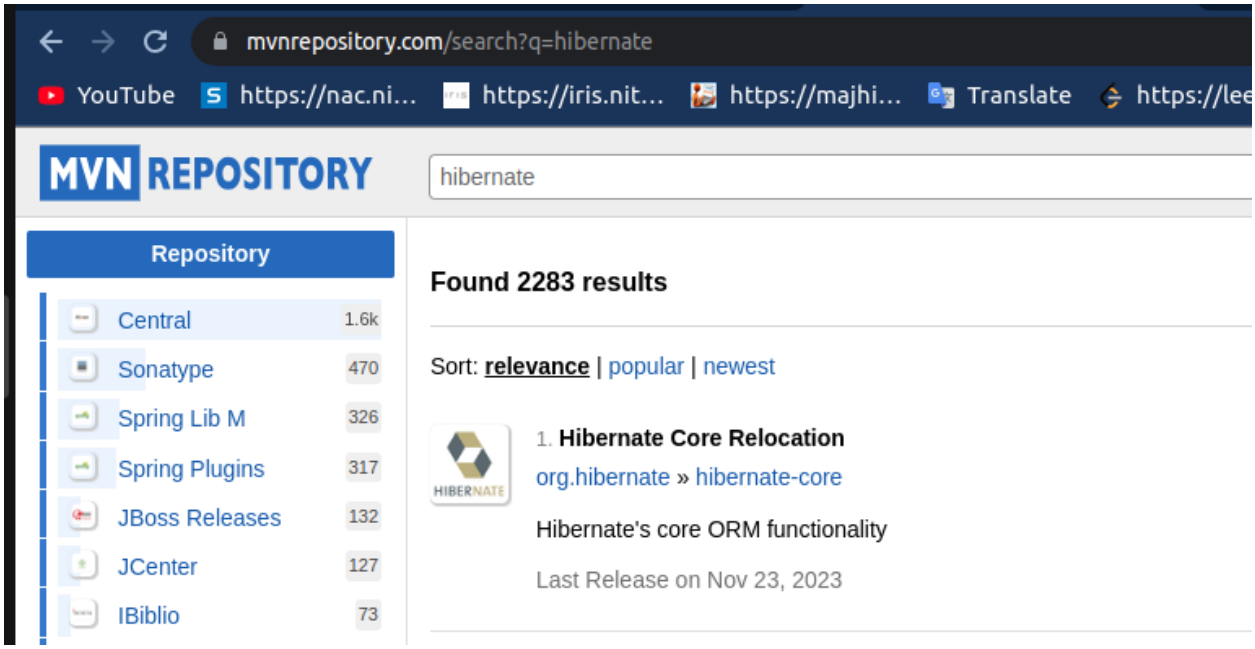
App.java | Alian.java

```java
package com.telusko.DemoHib;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alian telusko = new Alian();
        telusko.setAid(101);
        telusko.setAname("navin");
        telusko.setColor("Green");


    }
}

```

- That data stored in the object is now present in the application not in a persistence state like a database, for that we have to import all libraries required for hibernate.
- Open → pom.xml → add dependency

App.java | Alian.java | DemoHib/pom.xml ×

```
http://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation with catalog)
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.telusko</groupId>
  <artifactId>DemoHib</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>DemoHib</name>
  <url>http://maven.apache.org</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

- 

Maven | Gradle | Gradle (Short) | Gradle (Kotlin) | SBT | Ivy | Grape | Leiningen | Buildr

```
<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>5.6.15.Final</version>
</dependency>
```

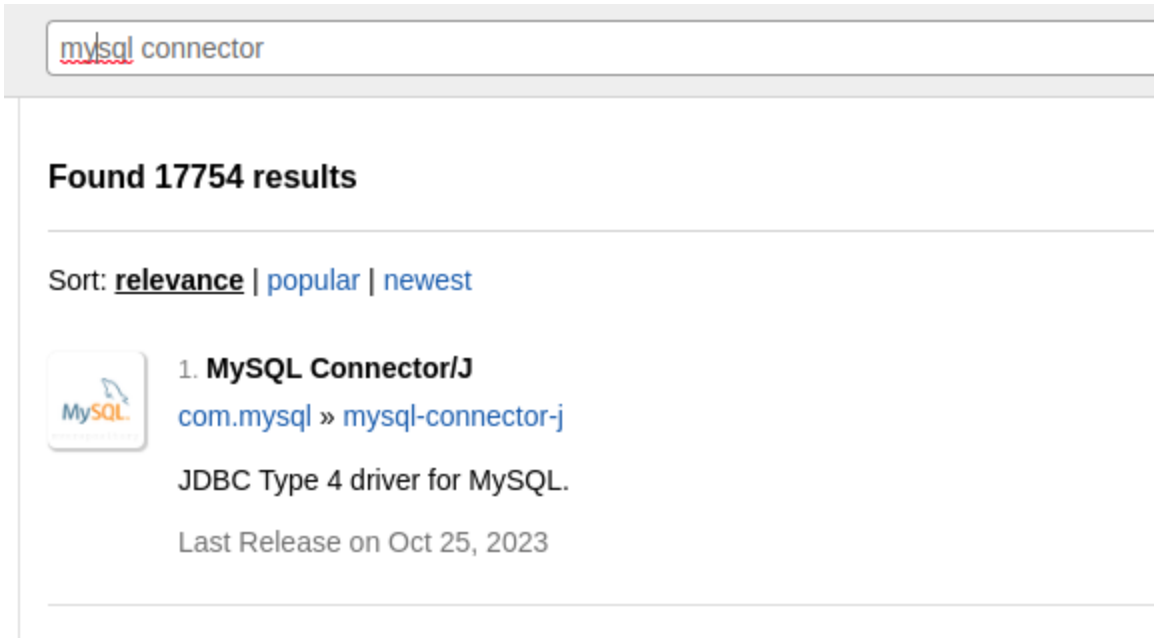
☑ Include comment with link to declaration

- 
- Copy that and paste in the pom.xml file

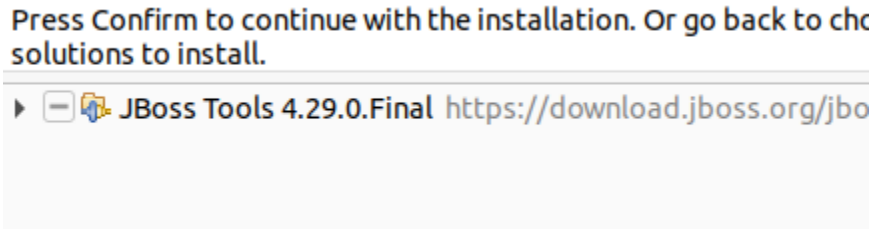
App.java | Alian.java | DemoHib/pom.xml ×

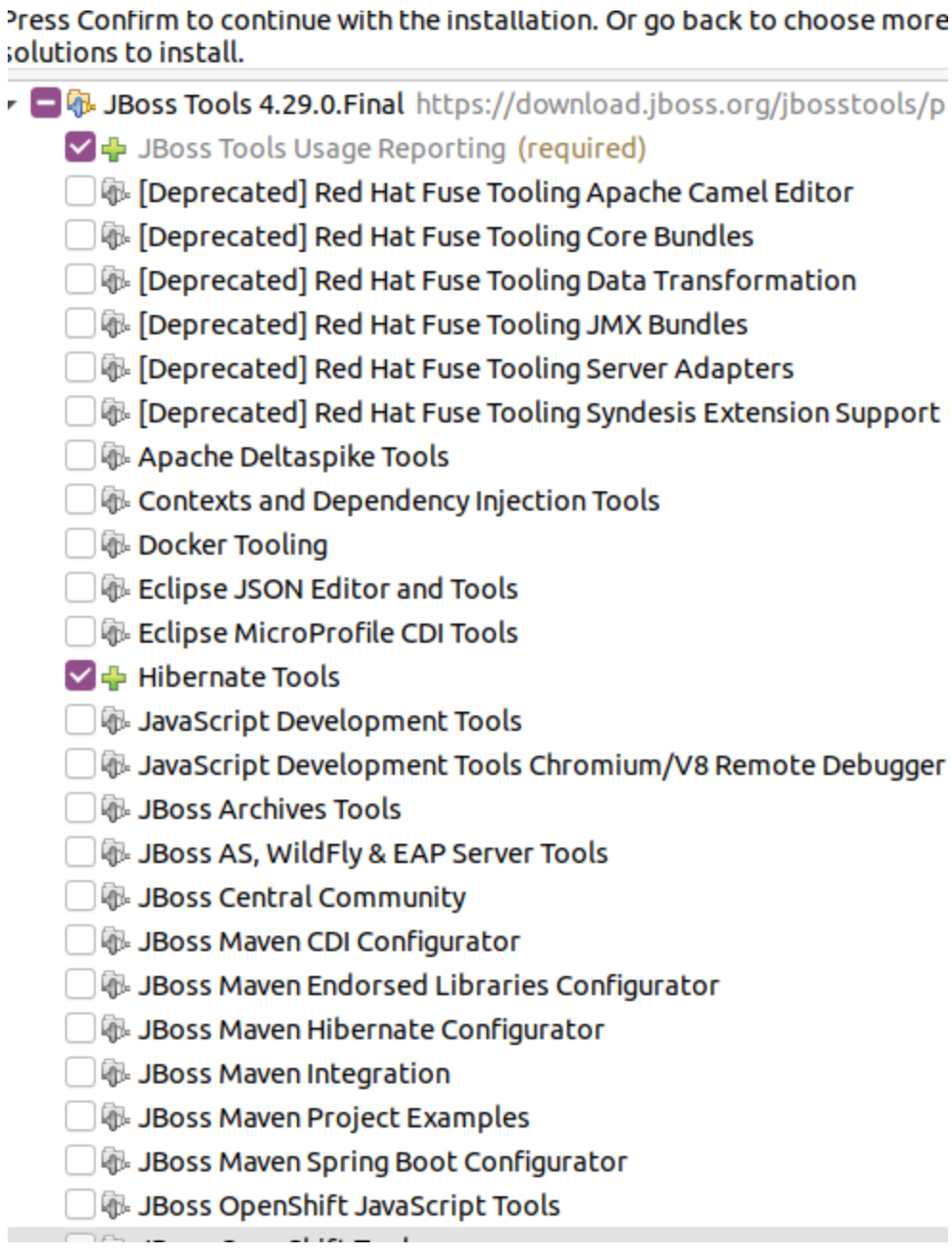
```
  http://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation with catalog)
1 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
3   <modelVersion>4.0.0</modelVersion>
4
5   <groupId>com.telusko</groupId>
6   <artifactId>DemoHib</artifactId>
7   <version>0.0.1-SNAPSHOT</version>
8   <packaging>jar</packaging>
9
10  <name>DemoHib</name>
11  <url>http://maven.apache.org</url>
12
13  <properties>
14    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15  </properties>
16
17  <dependencies>
18    <dependency>
19      <groupId>junit</groupId>
20      <artifactId>junit</artifactId>
21      <version>3.8.1</version>
22      <scope>test</scope>
23    </dependency>
24
25    <!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->
26 <dependency>
27    <groupId>org.hibernate</groupId>
28    <artifactId>hibernate-core</artifactId>
29    <version>5.6.15.Final</version>
30 </dependency>
31
32  </dependencies>
33 </project>
34
```

- 

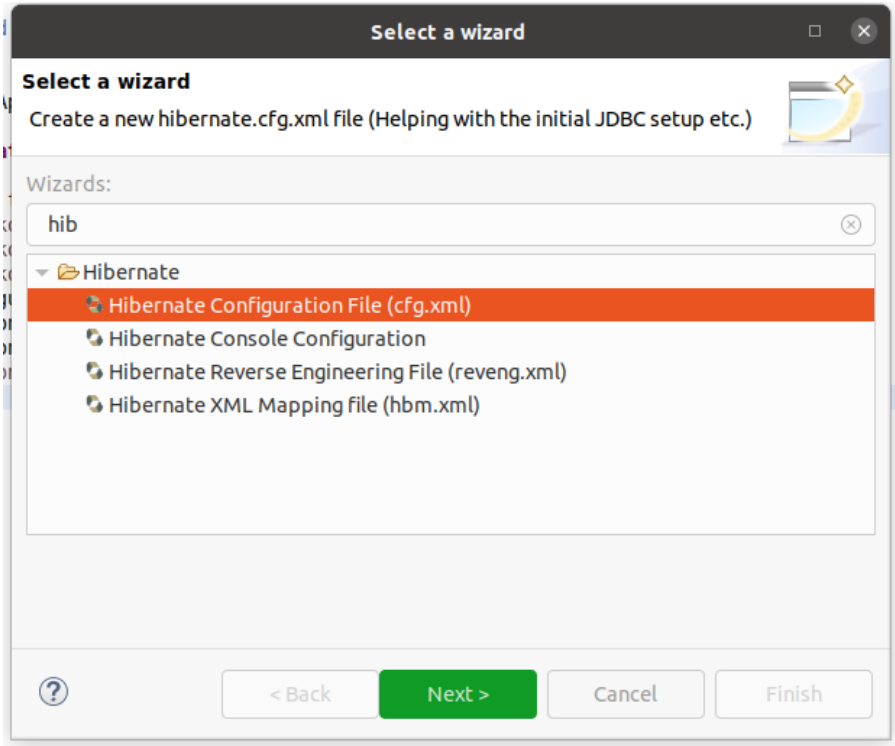



- Copy the dependency and paste in the pom.xml file
- Now we ready with all the dependency
- The flow of code”
    - `Configuration con = new Configuration();`
    - `SessionFactory sf = con.buildSessionFactory();`
    - `Session session = sf.openSession();  // it is an interface`
    - `session.save(telusko);`
- We have to do configuration with the database, like which RDBMS, which database, username, and password.
    - For this, we need to install the hibernate plugin
    - Eclipes → marketplace →serach “hibernate” → jBoss tool(it contains many tool select only hibernate)

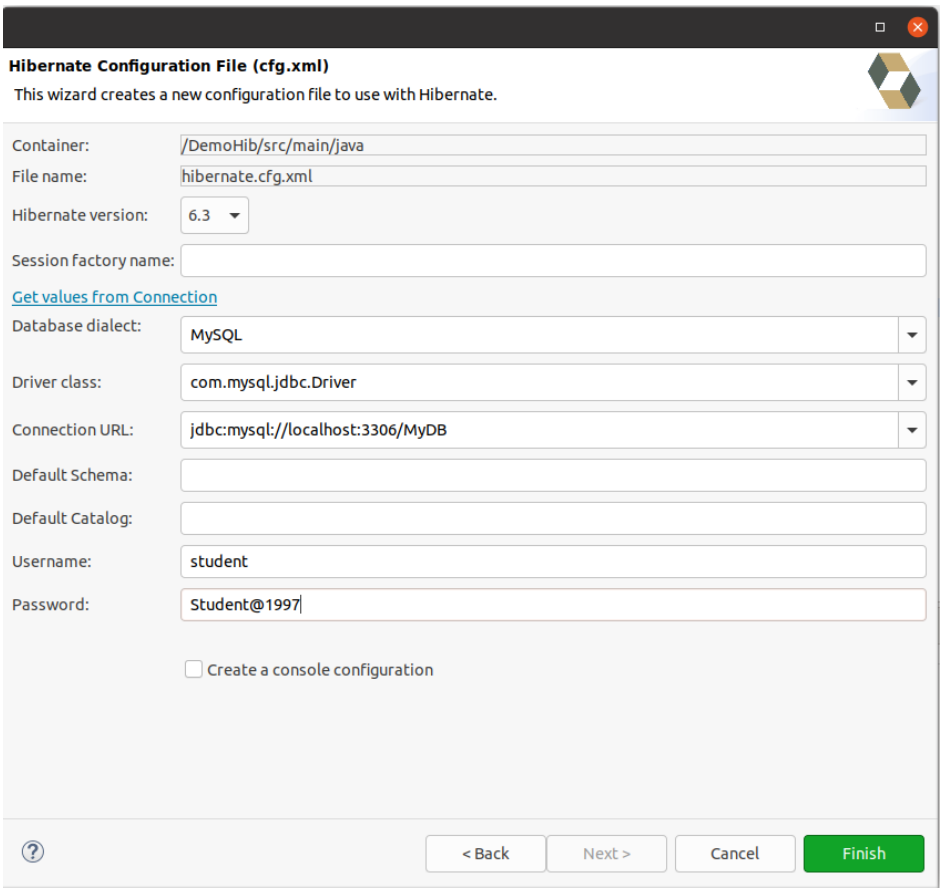
      
      




- 
- Now you get hibernate in your eclipse.
- For database configuration, we store it into XML file
  - Right-click on your project → search for hibernate configuration XML file




- Configure database:



- We got our cfg file

```
-//Hibernate/Hibernate Configuration DTD 3.0//EN (doctype with catalog)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.password">Student@1997</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/MyDB</property>
        <property name="hibernate.connection.username">student</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
    </session-factory>
</hibernate-configuration>
```

-

```
package com.telusko.DemoHib;

import org.hibernate.Session;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alian telusko = new Alian();
        telusko.setAid(101);
        telusko.setAname("navin");
        telusko.setColor("Green");
        Configuration con = new Configuration().configure();  // in configure file set configuration file name,
                                                               //where database connect store
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        session.save(telusko);

    }
}
```

-

- But still, we have problem, we didn't allow the class to be stored in database, so we can you Annotation //Entity

```
package com.telusko.DemoHib;

import javax.persistence.Entity;

@Entity
public class Alian   //pojo
{
    private int aid;
    private String aname;
    private String color;
    public int getAid() {
        return aid;
    }
    public void setAid(int aid) {
        this.aid = aid;
    }
    public String getAname() {
        return aname;
    }
    public void setAname(String aname) {
        this.aname = aname;
    }
    public String getColor() {
        return color;
    }
    public void setColor(String color) {
        this.color = color;
    }


}
```

  - 

- We have to mention in Configuration obj which class you are annotation use for,

*App.java × | Alian.java | DemoHib/pom.xml | hibernate.cfg.xml

```
1  package com.telusko.DemoHib;
2
3  import org.hibernate.Session;
6
7  /**
8   * Hello world!
9   *
10  */
11 public class App
12 {
13     public static void main( String[] args )
14     {
15         Alian telusko = new Alian();
16         telusko.setAid(101);
17         telusko.setAname("navin");
18         telusko.setColor("Green");
19         Configuration con = new Configuration().configure().addAnnotatedClass(Alian.class);  // in
20                                                                 //where database connect store
21         SessionFactory sf = con.buildSessionFactory();
22         Session session = sf.openSession();
23         session.save(telusko);
24
25     }
26 }
27
```

- 
- Note, that we have all the variables, in the database there should be a primary key, so we have to make one of the variables as Primary Key for that we use another annotation “ID”.

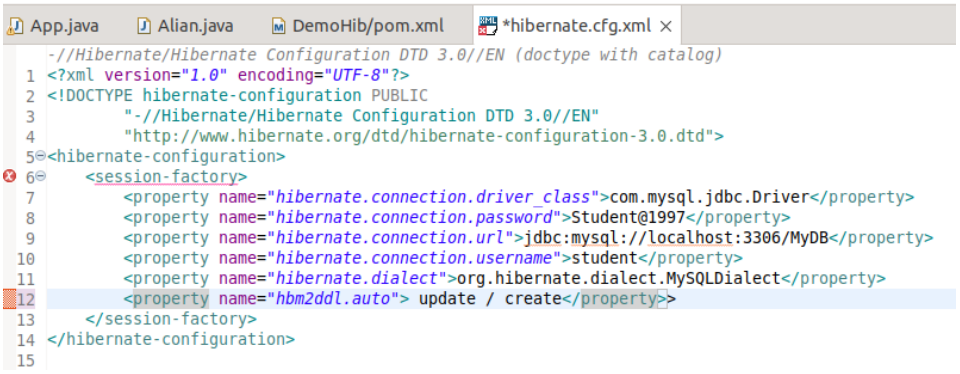
```
package com.telusko.DemoHib;

import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Alian   //pojo
{
    @Id
    private int aid;
    private String aname;
    private String color;
    public int getAid() {
        return aid;
    }
    public void setAid(int aid) {
        this.aid = aid;
    }
    public String getAname() {
        return aname;
    }
    public void setAname(String aname) {
        this.aname = aname;
    }
    public String getColor() {
        return color;
    }
    public void setColor(String color) {
        this.color = color;
    }


}
```

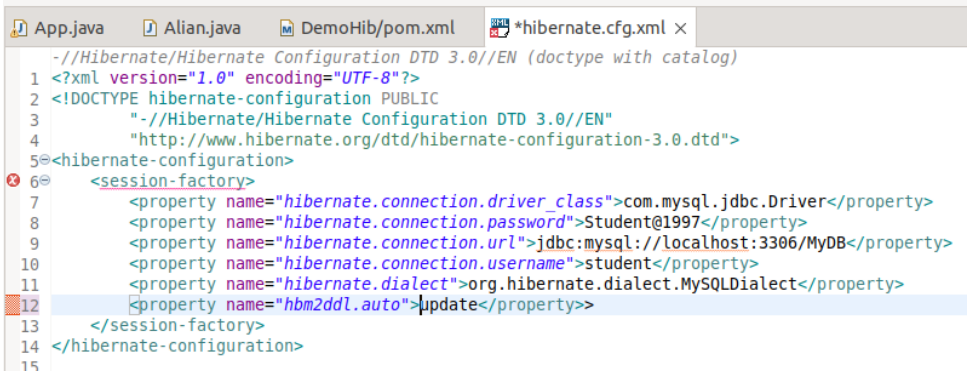
- 

- Here we have created a table and manipulated it with a table, so it should follow ACID property, for that, we have used “Transaction”.

```
 */
public class App
{
    public static void main( String[] args )
    {
        Alian telusko = new Alian();
        telusko.setAid(101);
        telusko.setAname("navin");
        telusko.setColor("Green");
        Configuration con = new Configuration().configure().addAnnotatedClass(Alian.class);
                                                                    //where database connect st
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        session.save(telusko);
        tx.commit();

    }
}
```

- This code does not directly create a table we have to make some changes in the configuration.xml file or create by userself manual in database
- Here you can use create/update. If you use create it create a new table every time, if you use update it check table is there, if there then update the entry or create a new table and update entries.




“MyDB” is the schema Name, following is correct ss:
Create a table manually in MYSQL Workbench, then run the Java code

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.password">Student@1997</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/new_schema</property>
        <property name="hibernate.connection.username">student</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property name="hibernate.hbm2ddl.auto">create</property>
        <property name="hibernate.show_sql">true</property>
    </session-factory>
</hibernate-configuration>
```

- In XML. the configuration file, we use “update” or “create”, which means that hibernate somewhere using SQL for creating or updating SQL table. If you want to see that you can use “show_sql” true.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.password">Student@1997</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/new_schema</property>
        <property name="hibernate.connection.username">student</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property name="hibernate.hbm2ddl.auto">create</property>
        <property name="hibernate.show_sql">true</property>
    </session-factory>
</hibernate-configuration>
```

- 

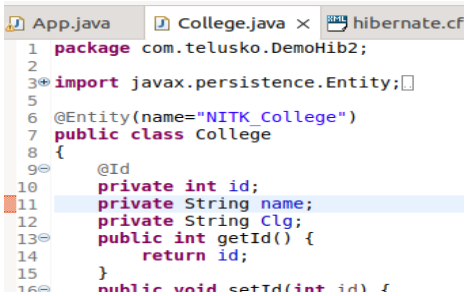
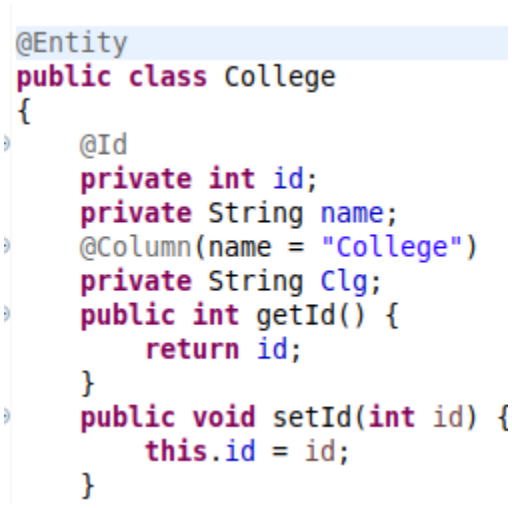
```
Dec 11, 2023 1:47:09 PM org.hibernate.resource.transaction.backend.
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [
Dec 11, 2023 1:47:09 PM org.hibernate.engine.transaction.jta.platfo
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.e
Hibernate: insert into College (Clg, name, id) values (?, ?, ?)
```

- 

## Table is not created by code, so following is correct configuration.xml file:

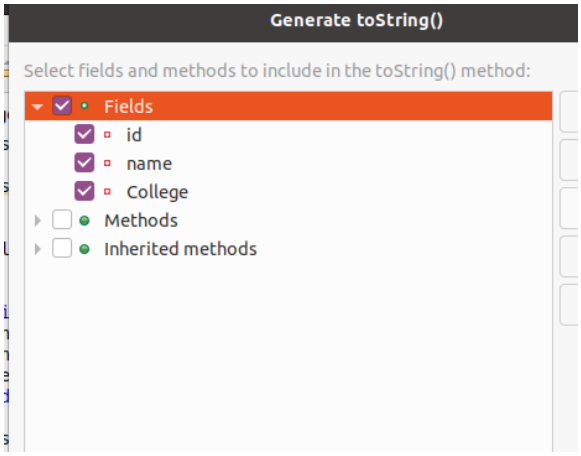
```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.password">Student@1997</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/new_schema</property>
        <property name="hibernate.connection.username">student</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property name="hibernate.hbm2ddl.auto">create</property>
        <property name="hibernate.show_sql">true</property>
    </session-factory>
</hibernate-configuration>
```

**Annotation used in Hibernate**

- Suppose you don't want table name as the class name; you want your custom name that case you have to use @Enity(name="College_Table")
- 
```
App.java   College.java ×   hibernate.cf
 1 package com.telusko.DemoHib2;
 2
 3 import javax.persistence.Entity;
 5
 6 @Entity(name="NITK_College")
 7 public class College
 8 {
 9     @Id
10     private int id;
11     private String name;
12     private String Clg;
13     public int getId() {
14         return id;
15     }
16     public void setId(int id) {
```
- Suppose you don't want a variable name as column name, you want to specify your name, that case you use @Column(name="Column_Name")
- 
```
@Entity
public class College
{
    @Id
    private int id;
    private String name;
    @Column(name = "College")
    private String Clg;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
```
- If you do not want to push variable name as column name in table that case you can use
    - "@Transient" annotation -> below variable name not include as column name.

**Fetching Data**

- For fetching , hibernate fetch object so, we have to use "toString()" method to convert object into String.

- 


- App.java | *College.java | hibernate.cfg.xml | DemoHib2/pom.xml

```java
package com.telusko.DemoHib2;

import javax.persistence.Column;

@Entity
public class College
{
    @Id
    private int id;
    private String name;
    private String College;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getClg() {
        return College;
    }
    public void setClg(String clg) {
        College = clg;
    }
    @Override
    public String toString() {
        return "College [id=" + id + ", name=" + name + ", College=" + College + "]";
    }

}
```

- *App.java | College.java | hibernate.cfg.xml | DemoHib2/pom.xml

```java
package com.telusko.DemoHib2;

import javax.imageio.spi.ServiceRegistry;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        College obj = new College();
        Configuration con = new Configuration().configure().addAnnotatedClass(College.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session =  sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        obj = (College)session.get(College.class, 1);  // get method return object so We have to typecast
        tx.commit();
        System.out.println(obj);
    }
}
```

- Problems | Javadoc | Declaration | Console

```
<terminated> App (1) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (11-Dec-2023, 2:38:02 pm – 2
INFO: HHH10001003: Autocommit mode: false
Dec 11, 2023 2:38:02 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 11, 2023 2:38:02 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Dec 11, 2023 2:38:03 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcC
Dec 11, 2023 2:38:03 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select college0_.id as id1_0_0_, college0_.College as college2_0_0_, college0_.name as name3_0_0_ from College college0_ where college0_.id=?
College [id=1, name=Nilesh, College=NITK]
```

**How to use Embeddable Object**

- **Suppose you want have Full name in database, for that you can create Fname, Mname, Name in same file, instead of you can create new class file with these 3 variable and use getter setter as well, pass that class name as variable name.**

App.java | College.java | FullName.java

```java
package com.telusko.DemoHib2;

public class FullName
{
    private String Fname;
    private String Mname;
    private String Lname;
    public String getFname() {
        return Fname;
    }
    public void setFname(String fname) {
        Fname = fname;
    }
    public String getMname() {
        return Mname;
    }
    public void setMname(String mname) {
        Mname = mname;
    }
    public String getLname() {
        return Lname;
    }
    public void setLname(String lname) {
        Lname = lname;
    }

}
```

- App.java | *College.java | FullName.java

```java
import javax.persistence.Column;

@Entity
public class College
{
    @Id
    private int id;
    private FullName name;
    private String College;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public FullName getName() {
        return name;
    }

    public void setName(FullName name) {
        this.name = name;
    }

    public String getCollege() {
        return College;
    }

    public void setCollege(String college) {
        College = college;
    }

    @Override
    public String toString() {
        return "College [id=" + id + ", name=" + name + ", College=" + College + "]";
```

Problem is here, we have two classes with for variable, so my possible it create two table, or hibernate will get confused which table I have to create, so for that , I don't want to create "FullName" again, I want that variable insert in same table , that case in "FullName" class add one annotation call "Embaddle".

App.java College.java FullName.java ×

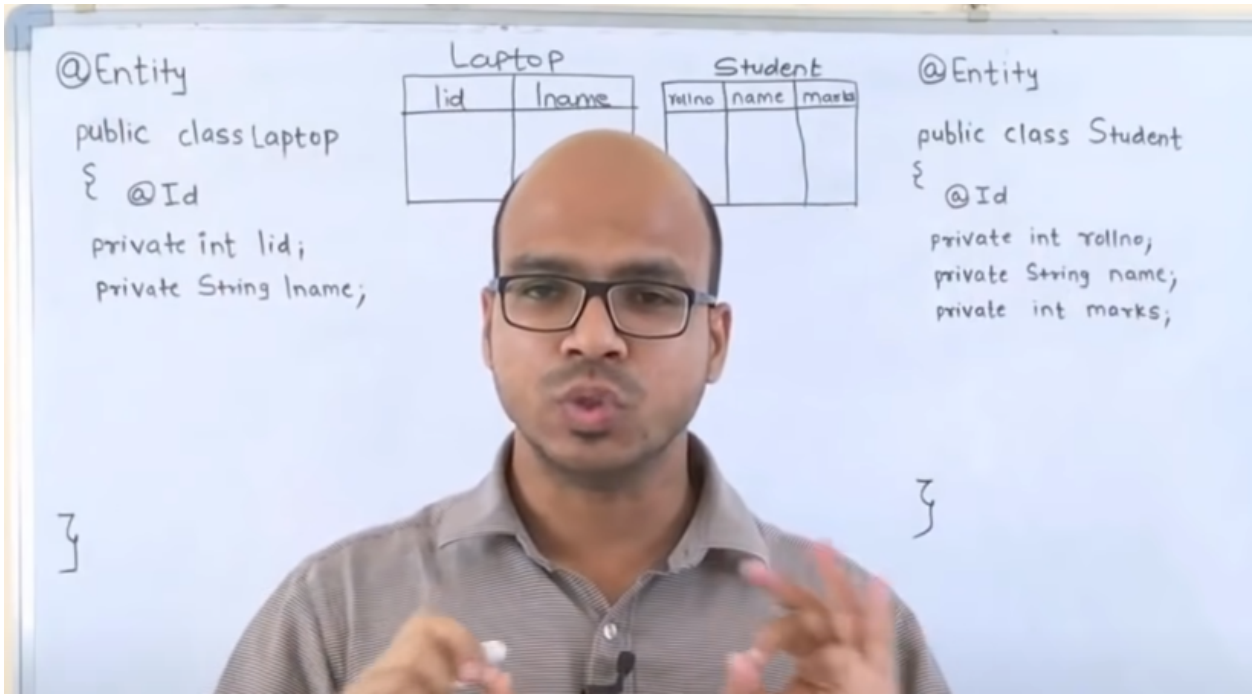
```java
package com.telusko.DemoHib2;

import javax.persistence.Embeddable;

@Embeddable
public class FullName
{
    private String Fname;
    private String Mname;
    private String Lname;
    public String getFname() {
        return Fname;
    }
    public void setFname(String fname) {
        Fname = fname;
    }
    public String getMname() {
        return Mname;
    }
    public void setMname(String mname) {
        Mname = mname;
    }
    public String getLname() {
        return Lname;
    }
    public void setLname(String lname) {
        Lname = lname;
    }

}
```

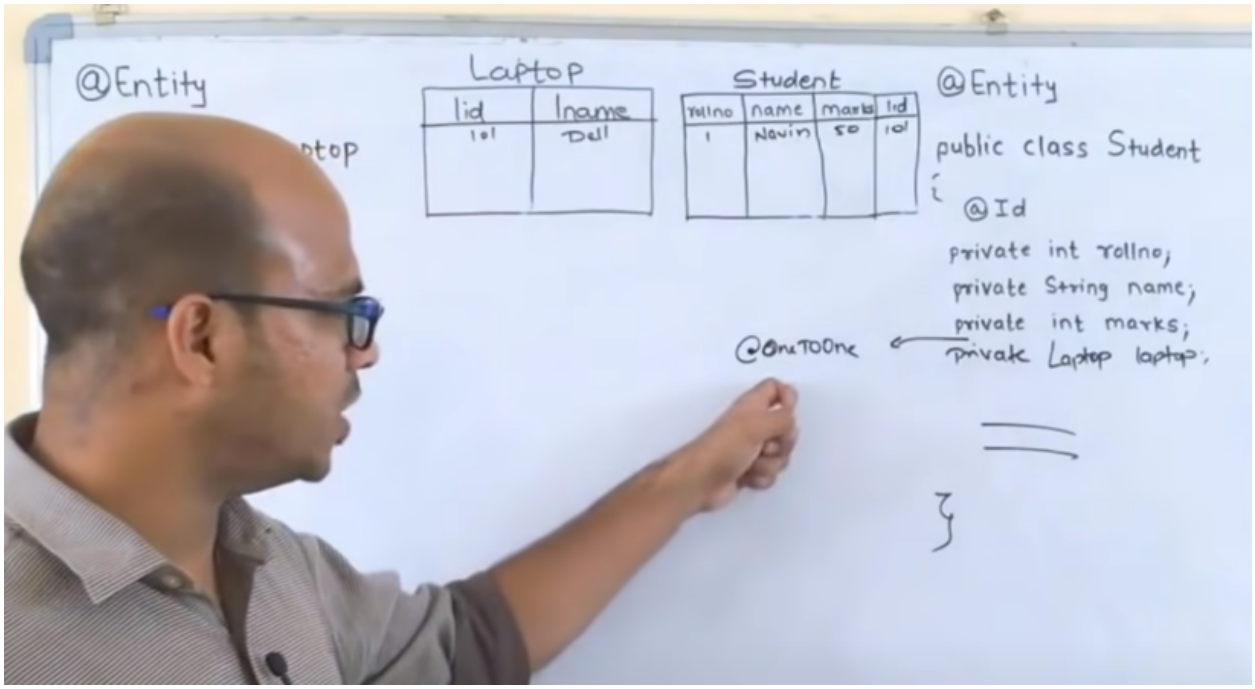
Problems Javadoc Declaration Console ×

```
<terminated> App (1) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.lin
Dec 11, 2023 2:58:10 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnecti
INFO: HHH10001003: Autocommit mode: false
Dec 11, 2023 2:58:10 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnecti
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 11, 2023 2:58:10 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Dec 11, 2023 2:58:11 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransact
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.
Dec 11, 2023 2:58:11 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformIn
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.pla
Hibernate: insert into College (College, Fname, Lname, Mname, id) values (?, ?, ?, ?, ?)
```

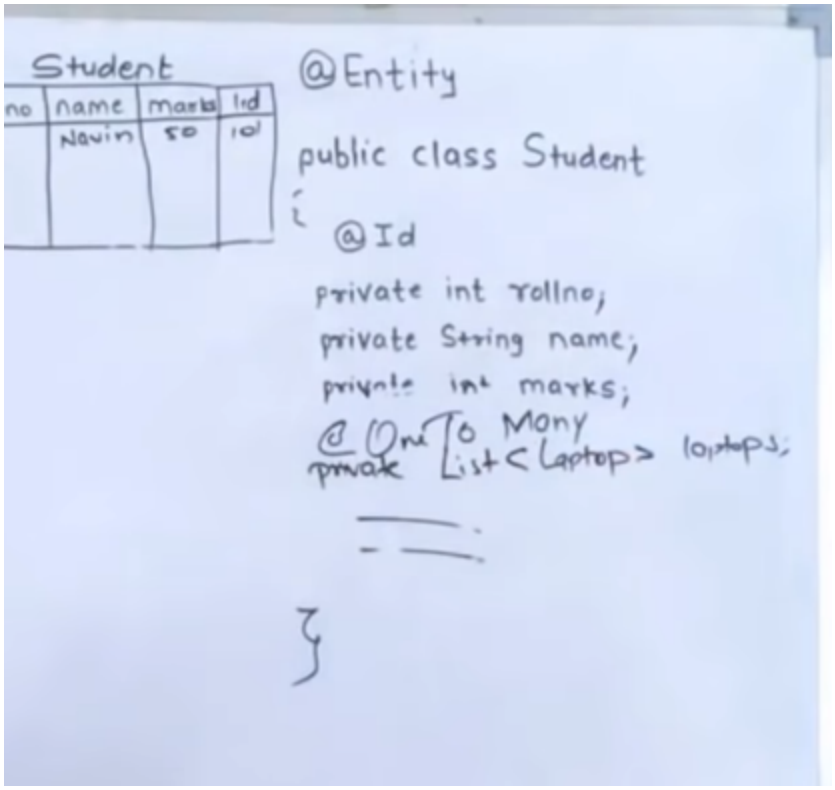
**Hibernate Relation**

- In sql we have 1-1,M-M,1-M,M-1 how we can implement this in hibernate.
- Take example, we have two classes,and both are “Entity”: it will create two separate tables.

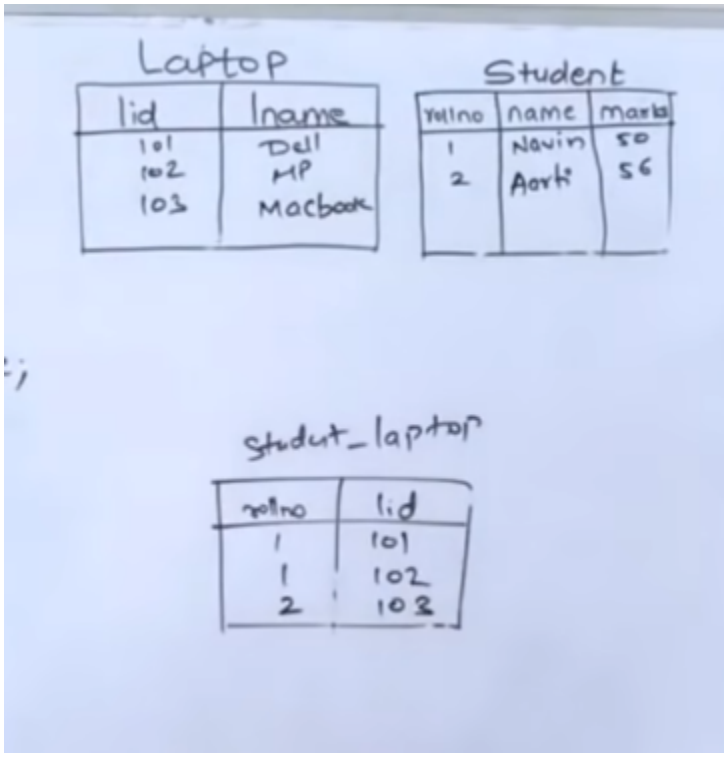
- 

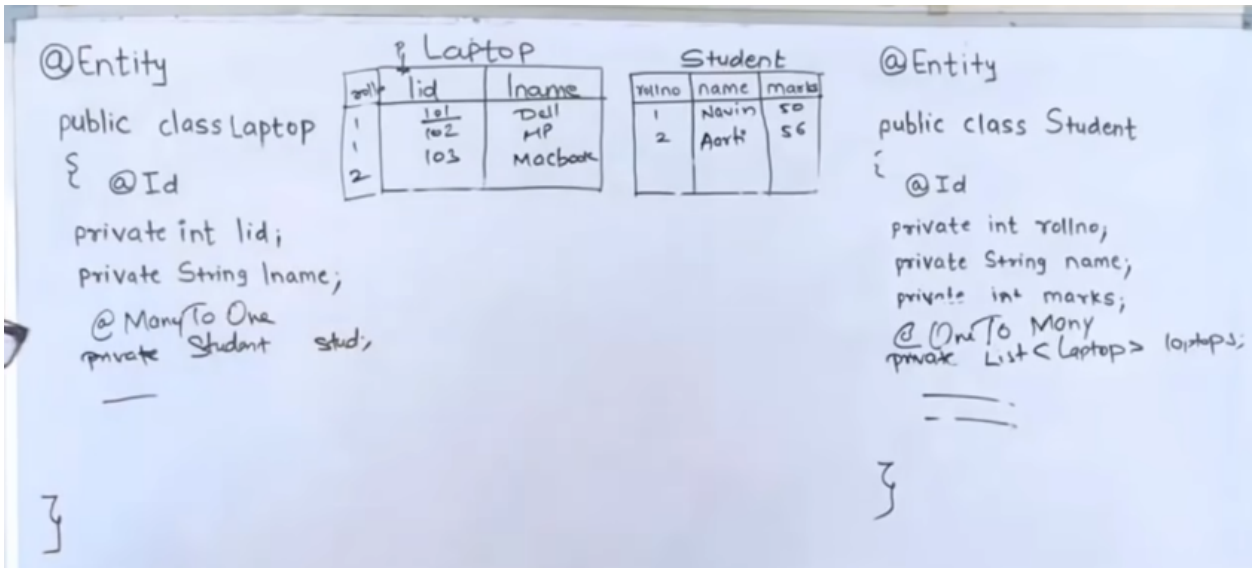
- One student have laptop, so we have to create object of “Laptop” class into student class for mapping 1-1 relation, and add one annotation (OneToOne) just below the object of “Laptop” class.
- 

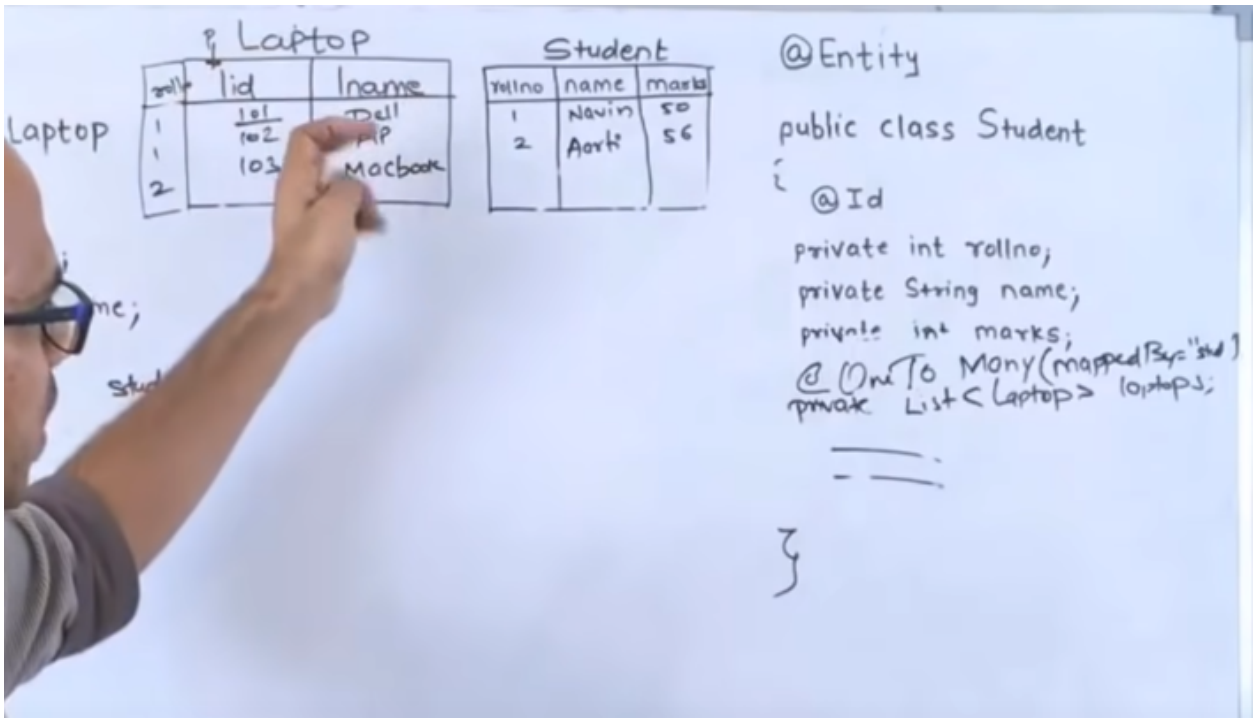
- Suppose student may have many laptops, that case you have to use “OneToMany” annotation



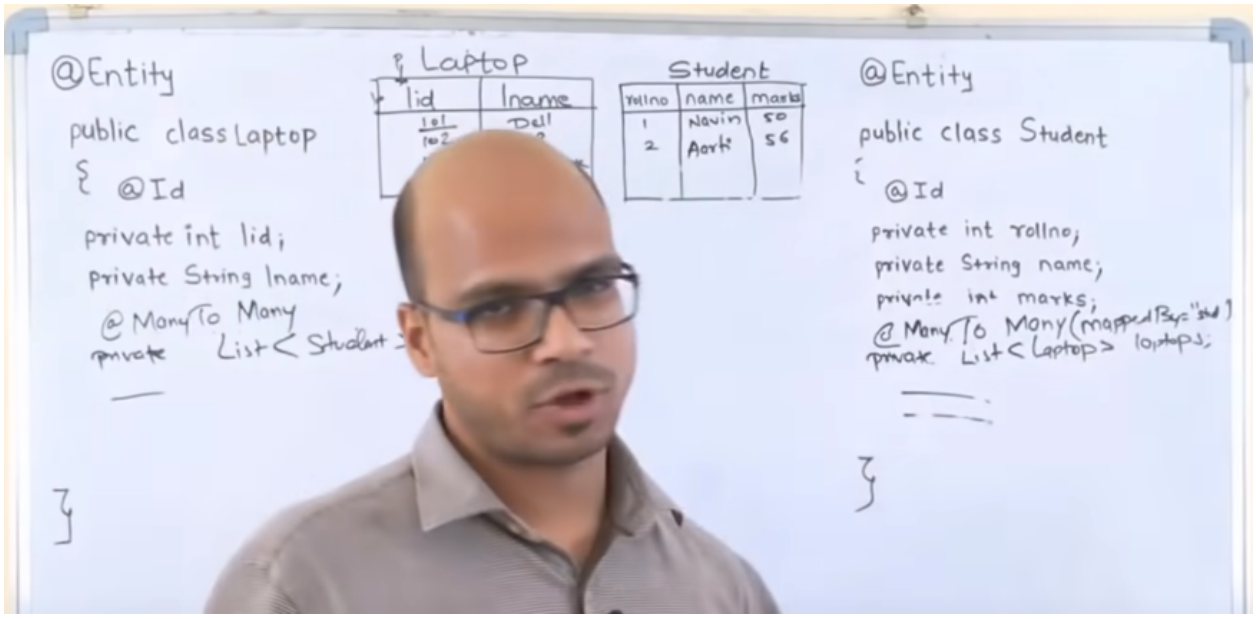

- But here we can not insert laptop directly in student table, row is atomic, and we can not make new entry with same roll number, id is primary, that case new table created with name “Student_Laptop”.



- Suppose I don't want to create new table , that case I want add new column in "Laptop" table, that case we should create object of "Student" table in Laptop class and Laptap class know student that case we have to use "ManyToOne" annotation in Laptop class. It mean that this laptop class responsibility to create column in "Laptop" table.
- 

- But still here new table is created because "Student" class have "OneToMany" annotation , so deleaning with this ,we have add "OneToMany(mappedBy="std"), mapping here done by laptop.
- 

- Many to Many relationship, suppose we "one student may have many laptop and multiple laptop belongs to one student", we can use "ManyToMany" annotation .

- 


For referring relational mapping point refere “DemoHib3” project.

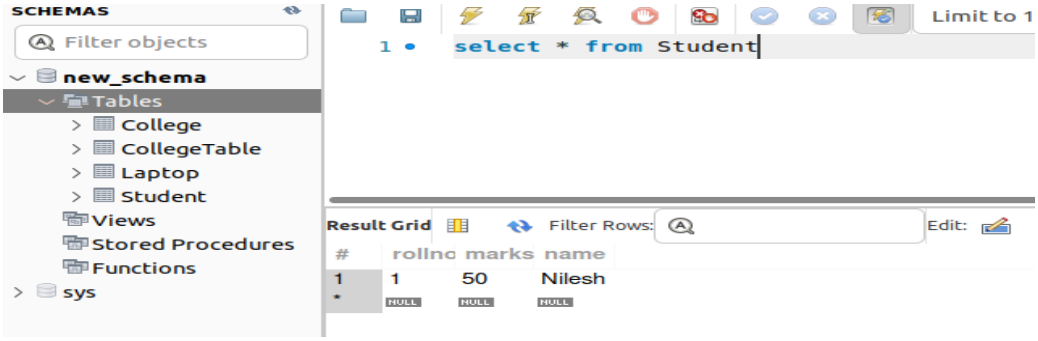
- I create two classes “Student” and “Laptop”
- Here not that I have to add both class for notation following is code

```
public static void main( String[] args )
{
    Laptop laptop = new Laptop();
    laptop.setLid(101);
    laptop.setLname("Dell");

    Student s = new Student();
    s.setRollno(1);;
    s.setMarks(50);
    s.setName("Nilesh");
    Configuration con = new Configuration().configure().addAnnotatedClass(Student.class).addAnnotatedClass(Laptop.class);
    SessionFactory sf = con.buildSessionFactory();
    Session session =  sf.openSession();
    org.hibernate.Transaction tx = session.beginTransaction();
    session.save(laptop);
    session.save(s);
    session.getTransaction().commit();
```

```
-//Hibernate/Hibernate Configuration DTD 3.0//EN (doctype with catalog)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.password">Student@1997</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/new_schema</property>
        <property name="hibernate.connection.username">student</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property name="hibernate.hbm2ddl.auto">create</property>
        <property name="hibernate.show_sql">true</property>
    </session-factory>
</hibernate-configuration>
```

- Make sure that in “Student” and ”Laptop” class you should use (@Entity, toString() and @Id)
- When I run this code, It will create two table “Student” and “Laptop”

```
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Hibernate: drop table if exists Laptop
Dec 12, 2023 7:12:56 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$Connect
Hibernate: drop table if exists Student
Hibernate: create table Laptop (lid integer not null, lname varchar(255), primary key (lid)) engine=InnoDB
Dec 12, 2023 7:12:56 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$Connect
Hibernate: create table Student (rollno integer not null, marks integer not null, name varchar(255), primary key (rollno)) engine=InnoDB
Dec 12, 2023 7:12:57 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: insert into Laptop (lname, lid) values (?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
```

- 

- Case 1 (OneToOne): For mapping we have to create object of laptop in student class, and create getter and setter for laptop as well, in App.java file create obj of laptop and assign it to student.,
  - Directly running this code we got an error

```
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 12, 2023 7:19:36 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PoolState stop
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/new_schema]
Exception in thread "main" org.hibernate.MappingException: Could not determine type for: com.telusko.DemoHib3.Laptop, at table: Student, for columns: [org.hiber
        at org.hibernate.mapping.SimpleValue.getType(SimpleValue.java:515)
        at org.hibernate.mapping.SimpleValue.isValid(SimpleValue.java:482)
        at org.hibernate.mapping.Property.isValid(Property.java:231)
        at org.hibernate.mapping.PersistentClass.validate(PersistentClass.java:627)
        at org.hibernate.mapping.RootClass.validate(RootClass.java:267)
        at org.hibernate.boot.internal.MetadataImpl.validate(MetadataImpl.java:359)
        at org.hibernate.internal.SessionFactoryImpl.<init>(SessionFactoryImpl.java:314)
```

  - Here Student table don’t know relationship between laptop and student, so I have to mentioned this in student class

```
App.java | Student.java | Laptop.java

1  package com.telusko.DemoHib3;
2
3  import javax.persistence.Entity;
4  import javax.persistence.Id;
5  import javax.persistence.OneToOne;
6
7  @Entity
8  public class Student
9  {
10     @Id
11     private int rollno;
12     private String name;
13     private int marks;
14     @OneToOne
15     Laptop laptop;
16     public int getRollno() {
17         return rollno;
18     }
19     public void setRollno(int rollno) {
20         this.rollno = rollno;
21     }
```

- 
```
Dec 12, 2023 7:29:33 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 12, 2023 7:29:33 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Hibernate: alter table Student drop foreign key FKbt5lbpcekxv81bs28nm4odw71
Dec 12, 2023 7:29:33 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectio
Hibernate: drop table if exists Laptop
Hibernate: drop table if exists Student
Hibernate: create table Laptop (lid integer not null, lname varchar(255), primary key (lid)) engine=InnoDB
Dec 12, 2023 7:29:33 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectio
Hibernate: create table Student (rollno integer not null, marks integer not null, name varchar(255), laptop_lid integer, primary key (rollno)) engine=InnoDB
Hibernate: alter table Student add constraint FKbt5lbpcekxv81bs28nm4odw71 foreign key (laptop_lid) references Laptop (lid)
Dec 12, 2023 7:29:33 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: insert into Laptop (lname, lid) values (?, ?)
Hibernate: insert into Student (laptop_lid, marks, name, rollno) values (?, ?, ?, ?)
```
- 

Result Grid | Filter Rows:

| # | rollno | marks | name | laptop_lic |
|---|---|---|---|---|
| 1 | 1 | 50 | Nilesh | 101 |
| * | NULL | NULL | NULL | NULL |

- Case 2 (OneToMany → one student have many laptop)
  - We have create “Laptop” object as ArrayList<Laptop>, change getter and setter as well for this “List”.
  - 

App.java | Student.java | Laptop.java | hibernate.cfg.xml

```
8  import javax.persistence.OneToMany;
9  import javax.persistence.OneToOne;
10
11 @Entity
12 public class Student
13 {
14     @Id
15     private int rollno;
16     private String name;
17     private int marks;
18     @OneToMany
19     private List<Laptop> laptop = new ArrayList<>();
20     public int getRollno() {
21         return rollno;
22     }
23     public void setRollno(int rollno) {
```

  - 

```
    private int marks;
    @OneToMany
    private List<Laptop> laptop = new ArrayList<>();
    public int getRollno() {
        return rollno;
    }
```

  - 

```
public List<Laptop> getLaptop() {
    return laptop;
}
public void setLaptop(List<Laptop> laptop) {
    this.laptop = laptop;
}
```

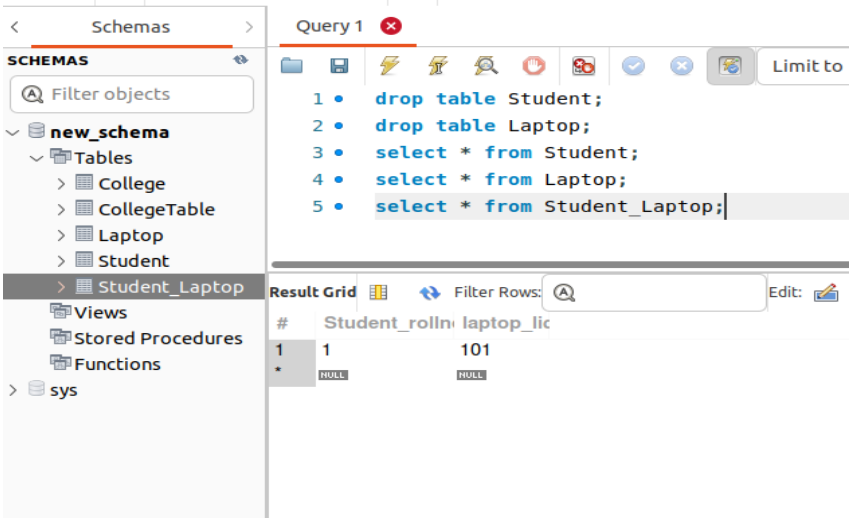
App.java | Student.java | Laptop.java | hibernate.cfg.xml

```
3  import org.hibernate.Session;
4  import org.hibernate.SessionFactory;
5  import org.hibernate.cfg.Configuration;
6
7  /**
8   * Hello world!
9   *
10  */
11 public class App
12 {
13     public static void main( String[] args )
14     {
15         Laptop laptop = new Laptop();
16         laptop.setLid(101);
17         laptop.setLname("Dell");
18         Student s = new Student();
19         s.setRollno(1);;
20         s.setMarks(50);
21         s.setName("Nilesh");
22         s.getLaptop().add(laptop);
23         Configuration con = new Configuration().configure().adc
24         SessionFactory sf = con.buildSessionFactory();
25         Session session =  sf.openSession();
26         org.hibernate.Transaction tx = session.beginTransactior
27         session.save(laptop);
28         session.save(s);
29         session.getTransaction().commit();
30
31     }
32 }
```

- It will give error because hibernate not able to delete student because now it have foreign key of Laptop, so delete manually.
- After delete it will work:

```
<terminated> App (2) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (12-Dec-2
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$Connecti
Hibernate: alter table Student_Laptop drop foreign key FKf50k4j1dephx1see92301ln6c
Hibernate: drop table if exists Laptop
Hibernate: drop table if exists Student
Hibernate: drop table if exists Student_Laptop
Hibernate: create table Laptop (lid integer not null, lname varchar(255), primary key (lid)) engine=InnoDB
Dec 12, 2023 7:37:28 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$Connecti
Hibernate: create table Student (rollno integer not null, marks integer not null, name varchar(255), primary key (rollno)) engine=InnoDB
Hibernate: create table Student_Laptop (Student_rollno integer not null, laptop_lid integer not null) engine=InnoDB
Hibernate: alter table Student_Laptop add constraint UK_bh2bn978np3lskg6p95ao4gxw unique (laptop_lid)
Hibernate: alter table Student_Laptop add constraint FKfandba4qlts6tnwlixspu0sly foreign key (laptop_lid) references Laptop (lid)
Hibernate: alter table Student_Laptop add constraint FKf50k4j1dephx1see92301ln6c foreign key (Student_rollno) references Student (rollno)
Dec 12, 2023 7:37:28 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: insert into Laptop (lname, lid) values (?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
Hibernate: insert into Student_Laptop (Student_rollno, laptop_lid) values (?, ?)
```

- Here we got new table because OneToMany



- Case 3: previous example one student may have multiple laptop, Student know the laptop, but laptop don't know studen, we want to make sure that laptop also know many laptop own by one student;
    - For that make "Student" object in Laptop class and getter setter and annotation (@ManyToOne)

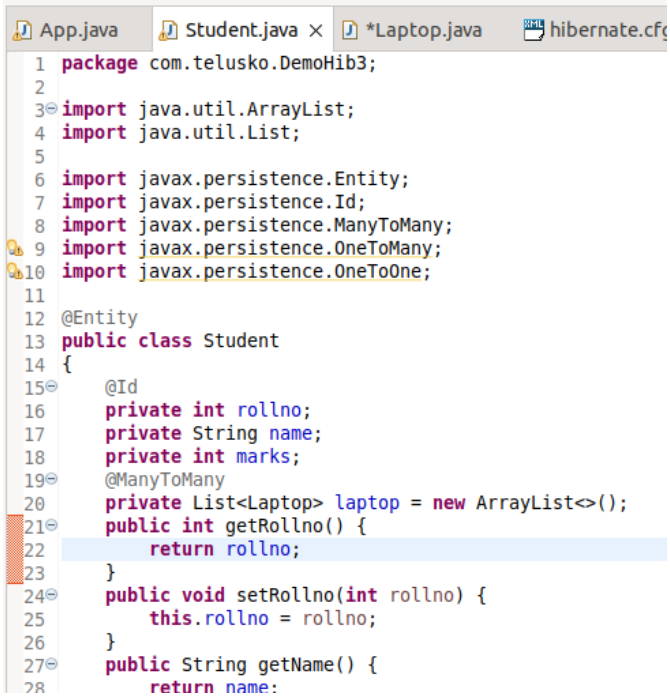
App.java | Student.java | Laptop.java | hibernate.cfg.x

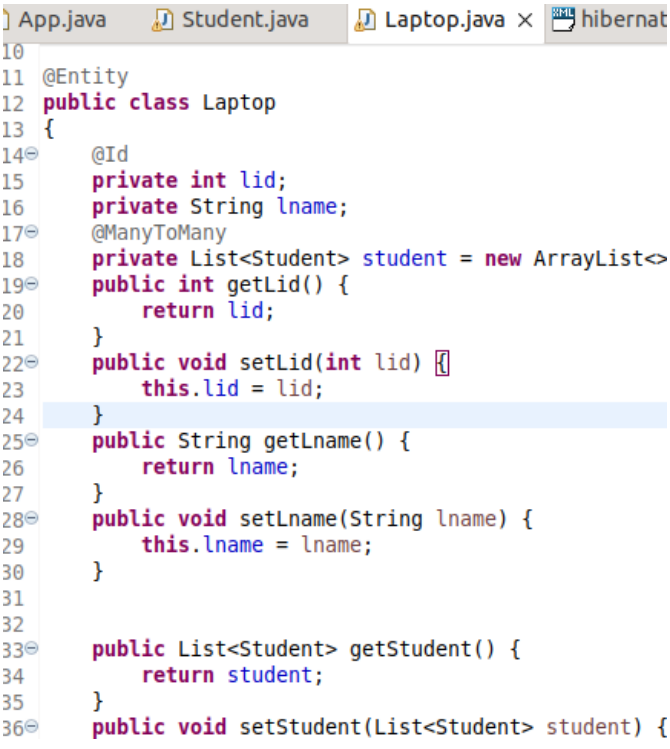
```java
package com.telusko.DemoHib3;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.ManyToOne;

@Entity
public class Laptop
{
    @Id
    private int lid;
    private String lname;
    @ManyToOne
    Student student;
    public int getLid() {
        return lid;
    }
    public void setLid(int lid) {
        this.lid = lid;
    }
    public String getLname() {
        return lname;
    }
    public void setLname(String lname) {
        this.lname = lname;
    }

    public Student getStudent() {
        return student;
    }
```

- 
    - But still it is creating new table  that "Roll number as ID and Lid", student class creating new table again so avoid that you add parameter in "Student" annotation .
    - Here it won't create new table,

```
Hibernate: drop table if exists Laptop
Hibernate: drop table if exists Student
Hibernate: create table Laptop (lid integer not null, lname varchar(255), student_rollno integer, primary key (lid)) engine=InnoDB
Dec 12, 2023 8:10:56 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$Connect
Hibernate: create table Student (rollno integer not null, marks integer not null, name varchar(255), primary key (rollno)) engine=InnoDB
Hibernate: alter table Laptop add constraint FKn8pdrsdaxhe8jlkf4vjtj9xne foreign key (student_rollno) references Student (rollno)
Dec 12, 2023 8:10:56 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: insert into Laptop (lname, student_rollno, lid) values (?, ?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
```

    -

- Case 4: ManyToMany, multiple student have multiple laptop, I make list of student and laptop in both classes and make getter setter as for both. In main class I added laptop object. Similarly I changes Anotation to ManyToMany for both.
    - 

    - 

App.java | Student.java | Laptop.java | hibernate.cfg.xm

```java
/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Laptop laptop = new Laptop();
        laptop.setLid(101);
        laptop.setLname("Dell");

        Student s = new Student();
        s.setRollno(1);;
        s.setMarks(50);
        s.setName("Nilesh");

        s.getLaptop().add(laptop);
        laptop.getStudent().add(s);
        Configuration con = new Configuration().configure(
        SessionFactory sf = con.buildSessionFactory();
        Session session =  sf.openSession();
        org.hibernate.Transaction tx = session.beginTransa
        session.save(laptop);
        session.save(s);
        session.getTransaction().commit();

    }
}
```

- Now I run this code, I will create 4 tables and some error also getting:

```
        ... 14 more

Hibernate: drop table if exists Laptop
Hibernate: drop table if exists Laptop_Student
Hibernate: drop table if exists Student
Hibernate: drop table if exists Student_Laptop
Hibernate: create table Laptop (lid integer not null, lname varchar(255), primary key (lid)) engine=InnoDB
Dec 12, 2023 8:19:34 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConn
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator
Hibernate: create table Laptop_Student (Laptop_lid integer not null, student_rollno integer not null) engine=InnoDB
Hibernate: create table Student (rollno integer not null, marks integer not null, name varchar(255), primary key (rollno)) engine
Hibernate: create table Student_Laptop (Student_rollno integer not null, laptop_lid integer not null) engine=InnoDB
Hibernate: alter table Laptop_Student add constraint FKjsrlk0ioysn605fe862ivvvpp foreign key (student_rollno) references Student
Hibernate: alter table Laptop_Student add constraint FKpu1u7kdpkouegh3qbob69h436 foreign key (Laptop_lid) references Laptop (lid)
Hibernate: alter table Student_Laptop add constraint FKfandba4qlts6tnwlixspu0sly foreign key (laptop_lid) references Laptop (lid)
Hibernate: alter table Student_Laptop add constraint FKf50k4j1dephx1see92301ln6c foreign key (Student_rollno) references Student
Dec 12, 2023 8:19:34 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: insert into Laptop (lname, lid) values (?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
Hibernate: insert into Laptop_Student (Laptop_lid, student_rollno) values (?, ?)
Hibernate: insert into Student_Laptop (Student_rollno, laptop_lid) values (?, ?)
```

- Both class thinking , there responsibility to map the table, so in student class I write statement and tell don't map it.

App.java | Student.java × | Laptop.java | hibernate.

```
 7 import javax.persistence.Id;
 8 import javax.persistence.ManyToMany;
 9 import javax.persistence.OneToMany;
10 import javax.persistence.OneToOne;
11
12 @Entity
13 public class Student
14 {
15     @Id
16     private int rollno;
17     private String name;
18     private int marks;
19     @ManyToMany(mappedBy = "student")
20     private List<Laptop> laptop = new ArrayList<>();
21     public int getRollno() {
22         return rollno;
23     }
24     public void setRollno(int rollno) {
25         this.rollno = rollno;
26     }
```

```
Hibernate: drop table if exists Laptop
Hibernate: drop table if exists Laptop_Student
Hibernate: drop table if exists Student
Hibernate: create table Laptop (lid integer not null, lname varchar(255), primary key (lid)) engine=InnoDB
Dec 12, 2023 8:22:35 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$Connecti
Hibernate: create table Laptop_Student (laptop_lid integer not null, student_rollno integer not null) engine=InnoDB
Hibernate: create table Student (rollno integer not null, marks integer not null, name varchar(255), primary key (rollno)) engine=InnoDB
Hibernate: alter table Laptop_Student add constraint FKjsrlk0ioysn605fe862ivvvpp foreign key (student_rollno) references Student (rollno)
Hibernate: alter table Laptop_Student add constraint FK6h2b87ga5tvhb6r0ipxrlg3n9 foreign key (laptop_lid) references Laptop (lid)
Dec 12, 2023 8:22:35 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: insert into Laptop (lname, lid) values (?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
Hibernate: insert into Laptop_Student (laptop_lid, student_rollno) values (?, ?)
```

- Here we got 3 tables.

## Fetch → Eager or Lazy (Fer DemoHib4 project)

- Lazy meaning, do task at deadline
- Eager means, doing task at that time

Note : In object you can save at a time one instance.

- Here I want to fetch data from data base, and following data present in database;

```
1 • Select * from Alian;
```

Result Grid | Filter Rows:

| # | aid | sname |
|---|---|---|
| 1 | 1 | Nilesh |
| 2 | 2 | Rahul |
| 3 | 3 | Priyanshu |
| * | NULL | NULL |

```
Select * from Laptop;
```

Result Grid | Filter Rows: | Edit:

| # | lid | brand | price | alian_aid |
|---|---|---|---|---|
| 1 | 101 | Dell | 1000 | 1 |
| 2 | 102 | Apple | 2000 | 3 |
| 3 | 103 | Asus | 800 | 1 |
| 4 | 104 | Acer | 1500 | 3 |
| 5 | 105 | Samsung | 1400 | 1 |
| * | NULL | NULL | NULL | NULL |

- Classes of this table is following

Alian.java | Laptop.java | App.java | hibernate.cfg.xml

```java
import java.util.ArrayList;
import java.util.Collection;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.OneToMany;
@Entity
public class Alian
{
    @Id
    private int aid;
    private String sname;
    @OneToMany(mappedBy = "alian")
    private Collection<Laptop> laps = new ArrayList<Laptop>();

    public Collection<Laptop> getLaps() {
        return laps;
    }
    public void setLaps(Collection<Laptop> laps) {
        this.laps = laps;
    }
    public int getAid() {
        return aid;
    }
    public void setAid(int aid) {
        this.aid = aid;
    }
    public String getSname() {
        return sname;
    }
    public void setSname(String sname) {
        this.sname = sname;
    }
    @Override
    public String toString() {
        return "Alian [aid=" + aid + ", sname=" + sname + ", laps=" + laps + "]";
    }

}
```

- Alian.java | Laptop.java | App.java | hibernate.cfg.xml

```java
package com.telusko.DemoHib4;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.ManyToOne;

@Entity
public class Laptop
{
    @Id
    private int lid;
    private String brand;
    private int price;
    @ManyToOne
    private Alian alian;

    public Alian getAlian() {
        return alian;
    }
    public void setAlian(Alian alian) {
        this.alian = alian;
    }
    public int getLid() {
        return lid;
    }
    public void setLid(int lid) {
        this.lid = lid;
    }
    public String getBrand() {
        return brand;
    }
    public void setBrand(String brand) {
        this.brand = brand;
    }
    public int getPrice() {
        return price;
    }
    public void setPrice(int price) {
        this.price = price;
```

- Alian.java | Laptop.java | App.java | hibernate.cfg.xml

```java
package com.telusko.DemoHib4;

import javax.transaction.Transaction;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {

        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class).addAnnot
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Alian al = session.get(Alian.class, 1);
        session.getTransaction().commit();
    }
}
```

- After running this query you will get following output in consol.

```
Problems  Javadoc  Declaration  Console ×  Progress
<terminated> App (3) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/j
Dec 13, 2023 3:30:52 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001001: Connection properties: {password=****, user=student}
Dec 13, 2023 3:30:52 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Dec 13, 2023 3:30:52 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnect
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 13, 2023 3:30:53 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 13, 2023 3:30:53 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIso
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironment
Dec 13, 2023 3:30:53 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.sname as sname2_0_0_ from Alian alian0_ where alian0_.aid=?
```

- Now hare, you can see when we call alien in code, it only fire Alian query not laptop.
- But I change in code following

Alian.java  Laptop.java  App.java ×  hibernate.cfg.xml

```java
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {

        Configuration con = new Configuration().configure().addAnnotatedClass
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Alian al = session.get(Alian.class, 1);
        System.out.println(al.getSname());
        Collection<Laptop> laps = al.getLaps();
        for(Laptop l : laps)
        {
            System.out.println(l.getBrand());
        }
        session.getTransaction().commit();
    }
}
```

- Now we using laptop than only it fire laptop query.

```
Problems  Javadoc  Declaration  Console ×  Progress
<terminated> App (3) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (13-Dec-202
Dec 13, 2023 3:35:36 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001005: using driver [com.mysql.jdbc.Driver] at URL [jdbc:mysql://localhost:3306/new_schema]
Dec 13, 2023 3:35:36 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001001: Connection properties: {password=****, user=student}
Dec 13, 2023 3:35:36 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Dec 13, 2023 3:35:36 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 13, 2023 3:35:36 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 13, 2023 3:35:36 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$Connectior
Dec 13, 2023 3:35:36 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.sname as sname2_0_0_ from Alian alian0_ where alian0_.aid=?
Nilesh
Hibernate: select laps0_.alian_aid as alian_ai4_1_0_, laps0_.lid as lid1_1_0_, laps0_.lid as lid1_1_1_, laps0_.alian_aid as alian_ai4_1_1_,
Dell
Asus
Samsung
```

- It fire query of Laptop as well if you want, If you commnet that Laptop part then it only fire query of Alian only even Alian have multiple laptop.
- This scenario called “Lazy”.
- Now problem is, you fire query two time, one after another, so it give some delay here. It should fire query at same time. If you want to make it “Eager” than you have to only one change following.
- In “Alian.class” mentioned fetch type “Eager/Lazy” by default it Lazy, I selected “Eager”

Alian.java | Laptop.java | App.java | hibernate.cfg.xml

```java
package com.telusko.DemoHib4;

import java.util.ArrayList;
import java.util.Collection;

import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.Id;
import javax.persistence.OneToMany;
@Entity
public class Alian
{
    @Id
    private int aid;
    private String sname;
    @OneToMany(mappedBy = "alian", fetch = FetchType.EAGER)
    private Collection<Laptop> laps = new ArrayList<Laptop>();

    public Collection<Laptop> getLaps() {
        return laps;
    }
```

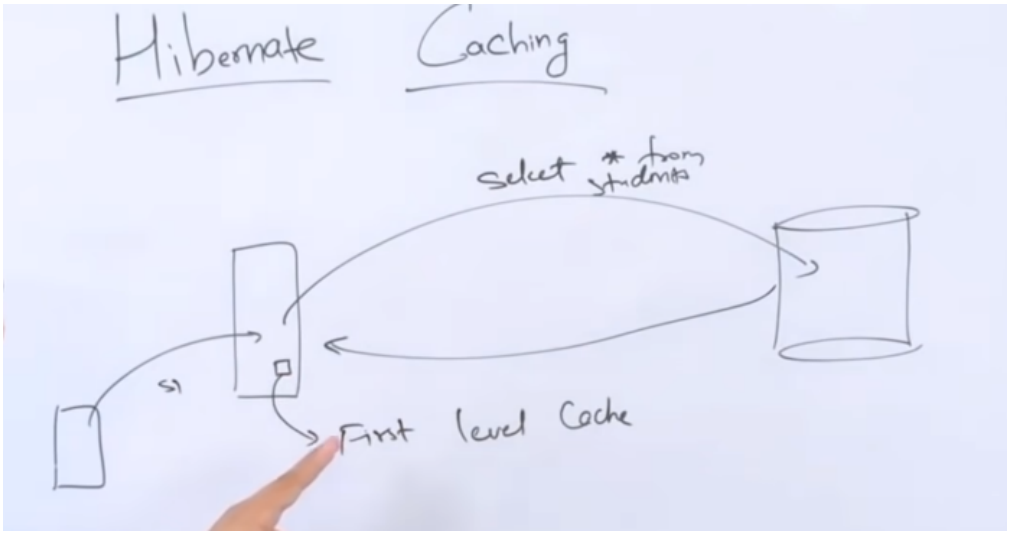
- After running code again, it fire only single query.

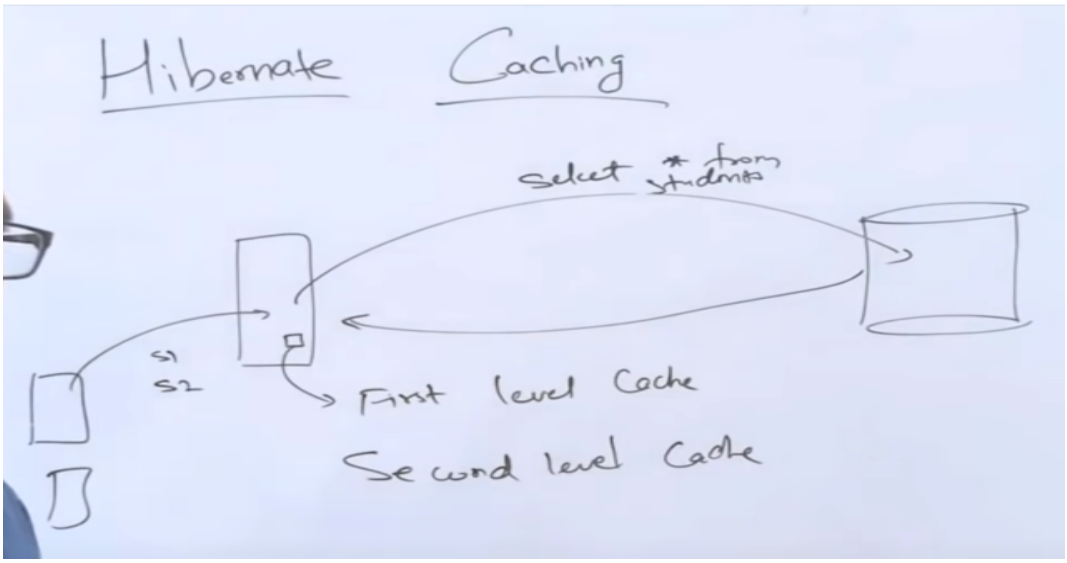
```
Dec 13, 2023 3:44:01 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001005: using driver [com.mysql.jdbc.Driver] at URL [jdbc:mysql://localhost:3306/new_schema]
Dec 13, 2023 3:44:01 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001001: Connection properties: {password=****, user=student}
Dec 13, 2023 3:44:01 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Dec 13, 2023 3:44:01 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 13, 2023 3:44:02 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 13, 2023 3:44:02 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@2bc
Dec 13, 2023 3:44:02 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.sname as sname2_0_0_, laps1_.alian_aid as alian_ai4_1_1_, laps1_.lid as lid1_1_1_, laps1_.lid as lid1_1_2_, laps1_.alian
Nilesh
Dell
Asus
Samsung
```

## Hibernate Caching

- It is one of the best feature of hibernate, Instead of hitting database every time, you can use cache memory for data.
- Suppose we have app, application server and DB
    - Your app fire query (select * from student) than application sever request it to DB, DB give data to application server and from that app receive that data, this select quer may have 1000 records or 10000 records or many, but second time app

again fire same query that time application server fetch data from DB, it's time consuming, that case Hibernate have "Cache" memory that store recent query result, so next time same query fire that case app will have to use cache, it's call "First Level Cache". For that particular session  will have first level cache by default.

- We can say First session have "First level Cache".
- 

- Suppose another user create session s2, he fire same query that case that 2nd user don't have First Level Cache. He have to fire query with DB because First Level Cache occupied by Session 1.
- Solution is that, he s2 can use "Second Level Cache"
- 

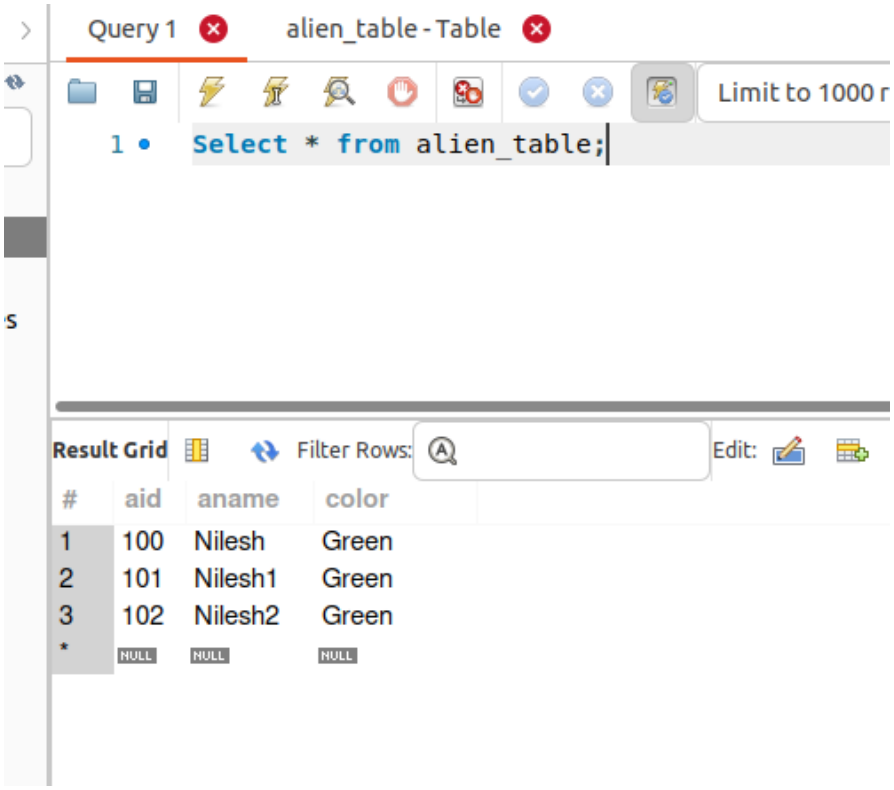
- Basically hibernate give two level cache.
- Hibernate give First Level Cache bydefault.
- For Second level cache you have to configure and all the session except first will use common second level cache.

- For second level, we have to use 3rd party library, following is cache provider library
  - eh-Cache (mostly used)
  - OS-cache
  - Swam cache
- EH-CACHE configuration
  - Pom.xml → dependency for eh-cache → provide features
  - Hibernate ehcache jar file → integration (communicate)
  - Hibernate.cgf.xml file → enable 2nd level cache here by adding property
  - Change your “Entity” because every Entity is not cachable, for that we have to add two annotation
    - @Cachable
    - @Cache

- L1-Cache Implemetation (refer DemoHib5)
- 

Alian.java × App.java

```
1  package com.telusko.DemoHib5;
2
3  import javax.persistence.Entity;
4  import javax.persistence.Id;
5  import javax.persistence.Table;
6
7  @Entity
8  @Table(name="alien_table")
9  public class Alian
10 {
11     @Id
12     private int aid;
13     private String aname;
14     private String color;
15     public int getAid() {
16         return aid;
17     }
18     public void setAid(int aid) {
19         this.aid = aid;
20     }
21     public String getAname() {
22         return aname;
23     }
24     public void setAname(String aname) {
25         this.aname = aname;
26     }
27     public String getColor() {
28         return color;
29     }
30     public void setColor(String color) {
31         this.color = color;
32     }
33     @Override
34     public String toString() {
```

- 

- Alian.java | App.java

```java
package com.telusko.DemoHib5;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alian a = null;
        Configuration con = new Configuration().configure().addAnnotatedClass(Alian.cla
        SessionFactory sf = con.buildSessionFactory();
        Session session1 = sf.openSession();
        Transaction tx = session1.beginTransaction();
        a = (Alian) session1.get(Alian.class, 100);
        System.out.println(a);


        session1.getTransaction().commit();
    }
}
```

- If I run this code following output will get print, it fire one query for id=100 and give output

```
Problems  Javadoc  Declaration  Console  Progress
<terminated> App (4) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (13-Dec-2023, 4:19:46 pm – 4:19:
INFO: HHH10001001: Connection properties: {password=****, user=student}
Dec 13, 2023 4:19:46 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Dec 13, 2023 4:19:46 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 13, 2023 4:19:47 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 13, 2023 4:19:47 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConne
Dec 13, 2023 4:19:47 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.aname as aname2_0_0_, alian0_.color as color3_0_0_ from alien_table alian0_ where alian0_.aid=?
Alian [aid=100, aname=Nilesh, color=Green]
```

- If I want another alien details, following code, it will fire two query

Alian.java  *App.java

```java
package com.telusko.DemoHib5;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alian a = null;
        Configuration con = new Configuration().configure().addAnnotatedClass(Alian.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session1 = sf.openSession();
        Transaction tx = session1.beginTransaction();
        a = (Alian) session1.get(Alian.class, 100);
        System.out.println(a);
        a = (Alian) session1.get(Alian.class, 101);
        System.out.println(a);

        session1.getTransaction().commit();
    }
}
```

```
Problems  Javadoc  Declaration  Console  Progress
<terminated> App (4) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (13-Dec-2023, 4:22:01 pm -
INFO: HHH10001003: Autocommit mode: false
Dec 13, 2023 4:22:02 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 13, 2023 4:22:02 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 13, 2023 4:22:02 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbc
Dec 13, 2023 4:22:02 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.aname as aname2_0_0_, alian0_.color as color3_0_0_ from alien_table alian0_ where alian0_.aid=?
Alian [aid=100, aname=Nilesh, color=Green]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.aname as aname2_0_0_, alian0_.color as color3_0_0_ from alien_table alian0_ where alian0_.aid=?
Alian [aid=101, aname=Nilesh1, color=Green]
```

- Here it actually fire two query for 100 and 101, what if I used 100 in both case , it will fire only one query. Because it first check in cache then only move for DB, because we already fire of 100 than that entry present in cache, so next time for 100 it will fetch data from cache.

- Alian.java | App.java

```java
package com.telusko.DemoHib5;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alian a = null;
        Configuration con = new Configuration().configure().addAnnotatedClass(Alian.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session1 = sf.openSession();
        Transaction tx = session1.beginTransaction();
        a = (Alian) session1.get(Alian.class, 100);
        System.out.println(a);
        a = (Alian) session1.get(Alian.class, 100);
        System.out.println(a);

        session1.getTransaction().commit();
    }
}
```

- Problems | Javadoc | Declaration | Console | Progress

```
<terminated> App (4) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (13-Dec-2023, 4:23:24 p
Dec 13, 2023 4:23:25 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Dec 13, 2023 4:23:25 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 13, 2023 4:23:25 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 13, 2023 4:23:25 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJ
Dec 13, 2023 4:23:25 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.aname as aname2_0_0_, alian0_.color as color3_0_0_ from alien_table alian0_ where alian0_.aid=?
Alian [aid=100, aname=Nilesh, color=Green]
Alian [aid=100, aname=Nilesh, color=Green]
```

- Now if we change the session , now we create session2
- You can see in output , it is going for DB 2 time, because we are using two different session. And First level cache available for only first session
- If we fetch data 2 time in 2nd level cache than first level cache is user:

Alian.java | App.java

```
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alian a = null;
        Configuration con = new Configuration().configure().addAnnotatedClass(Alian.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session1 = sf.openSession();
        Transaction tx = session1.beginTransaction();
        a = (Alian) session1.get(Alian.class, 100);
        System.out.println(a);
        session1.getTransaction().commit();
        session1.close();

        Session session2 = sf.openSession();
        session2.beginTransaction();
        a = (Alian) session2.get(Alian.class, 100);
        System.out.println(a);

        a = (Alian) session2.get(Alian.class, 100);
        System.out.println(a);
        session2.getTransaction().commit();



    }
}
```

Problems | Javadoc | Declaration | Console | Progress

```
<terminated> App (4) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (13-Dec-2023, 4:32:49 pm
Dec 13, 2023 4:32:49 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 13, 2023 4:32:50 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 13, 2023 4:32:50 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdb
Dec 13, 2023 4:32:50 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.aname as aname2_0_0_, alian0_.color as color3_0_0_ from alien_table alian0_ where alian0_.aid=?
Alian [aid=100, aname=Nilesh, color=Green]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.aname as aname2_0_0_, alian0_.color as color3_0_0_ from alien_table alian0_ where alian0_.aid=?
Alian [aid=100, aname=Nilesh, color=Green]
Alian [aid=100, aname=Nilesh, color=Green]
```

- .

Alian.java | App.java

```
package com.telusko.DemoHib5;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alian a = null;
        Configuration con = new Configuration().configure().addAnnotatedClass(Alian.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session1 = sf.openSession();
        Transaction tx = session1.beginTransaction();
        a = (Alian) session1.get(Alian.class, 100);
        System.out.println(a);
        session1.getTransaction().commit();
        session1.close();

        Session session2 = sf.openSession();
        session2.beginTransaction();
        a = (Alian) session2.get(Alian.class, 100);
        session2.getTransaction().commit();
        System.out.println(a);


    }
}
```

```
Problems  Javadoc  Declaration  Console ×  Progress
<terminated> App (4) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (13-Dec-2023, 4:28:44 p
INFO: HHH10001003: Autocommit mode: false
Dec 13, 2023 4:28:44 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 13, 2023 4:28:44 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 13, 2023 4:28:45 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJ
Dec 13, 2023 4:28:45 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.aname as aname2_0_0_, alian0_.color as color3_0_0_ from alien_table alian0_ where alian0_.aid=?
Alian [aid=100, aname=Nilesh, color=Green]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.aname as aname2_0_0_, alian0_.color as color3_0_0_ from alien_table alian0_ where alian0_.aid=?
Alian [aid=100, aname=Nilesh, color=Green]
```

- L2-cache
  - Download dependency “ehcache” paste in pom.xml →net.ef cache
  - Download dependency “hibernate-ehcache” in pom.xml

```
    <version>5.6.15.Final</version>
</dependency>
<!-- https://mvnrepository.com/artifact/com.mysql/mysql-connector-j -->
<dependency>
    <groupId>com.mysql</groupId>
    <artifactId>mysql-connector-j</artifactId>
    <version>8.0.33</version>
</dependency>
<!-- https://mvnrepository.com/artifact/net.sf.ehcache/ehcache -->
<dependency>
    <groupId>net.sf.ehcache</groupId>
    <artifactId>ehcache</artifactId>
    <version>2.10.6</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-ehcache -->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-ehcache</artifactId>
    <version>4.3.11.Final</version>
</dependency>

  </dependencies>
</project>
```

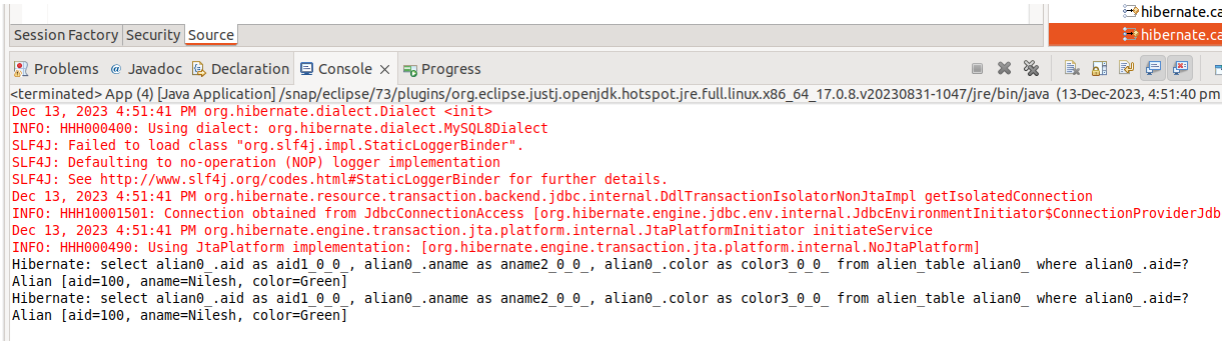
  - Add property in hibernate.cfg.xml file

```
Alian.java  App.java  hibernate.cfg.xml ×  DemoHib5/pom.xml
-//Hibernate/Hibernate Configuration DTD 3.0//EN (doctype with catalog)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.password">Student@1997</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/new_schema</property>
        <property name="hibernate.connection.username">student</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property name="hibernate.hbm2ddl.auto">update</property>
        <property name="hibernate.show_sql">true</property>
        <property name="hibernate.cache.use_second_level_cache">true</property>
        <property name="hibernate.cache.region.factory_class">org.hibernate.cache.ehcache.EhCacheRegionFactory</property>
        <property name="hibernate.cache.provider_class">net.sf.ehcache.hibernate.EhCacheProvider</property>
    </session-factory>
</hibernate-configuration>
```

- Following code, is run, than output is:

Alian.java | App.java | hibernate.cfg.xml | DemoHib5/pom.xml

```java
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alian a = null;
        Configuration con = new Configuration().configure().addAnnotatedClass(Alian.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session1 = sf.openSession();
        Transaction tx = session1.beginTransaction();
        a = (Alian) session1.get(Alian.class, 100);
        System.out.println(a);
        session1.getTransaction().commit();
        session1.close();

        Session session2 = sf.openSession();
        session2.beginTransaction();
        a = (Alian) session2.get(Alian.class, 100);
        System.out.println(a);
        session2.getTransaction().commit();
        session2.close();



    }
```




- Now here , we can see two query is fired, because all the Entity is not cachable , we have to specified explicitly which entity is going to use 2nd Level Cache.
- We use two notation
    - Cachable
    - Cache → here you have to mentioned which strtegy going to use, readonly or readwrite. → read only means cache is only for read not for update.

*Alian.java × App.java hibernate.cfg.xml DemoHib5/pom.xml

```java
package com.telusko.DemoHib5;

import javax.persistence.Cacheable;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

import org.hibernate.annotations.Cache;
import org.hibernate.annotations.CacheConcurrencyStrategy;

@Entity
@Table(name="alien_table")
@Cacheable
@Cache(usage = CacheConcurrencyStrategy.READ_ONLY)
public class Alian
{
    @Id
    private int aid;
    private String aname;
    private String color;
    public int getAid() {
        return aid;
    }
    public void setAid(int aid) {
        this.aid = aid;
    }
    public String getAname() {
        return aname;
    }
    public void setAname(String aname) {
        this.aname = aname;
    }
```

- Now again we run same code, you can see only one query is fired.

Problems Javadoc Declaration Console × Progress

```
App (4) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (13-Dec-2023, 4:58:32 pm) [pid: 140517]
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Dec 13, 2023 4:58:32 PM org.hibernate.cache.ehcache.internal.EhcacheRegionFactory createCache
WARN: HHH90001006: Missing cache[com.telusko.DemoHib5.Alian] was created on-the-fly. The created cache will use a provider-specific default configurati
Dec 13, 2023 4:58:32 PM org.hibernate.cache.spi.support.EntityReadOnlyAccess <init>
WARN: HHH90001003: Read-only caching was requested for mutable entity [NavigableRole[com.telusko.DemoHib5.Alian]]
Dec 13, 2023 4:58:33 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdb
Dec 13, 2023 4:58:33 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_0_, alian0_.aname as aname2_0_0_, alian0_.color as color3_0_0_ from alien_table alian0_ where alian0_.aid=?
Alian [aid=100, aname=Nilesh, color=Green]
Alian [aid=100, aname=Nilesh, color=Green]
```

- Even we have two different session , only one query is fired because both session sharing 2nd Level Cache.

## Hibernate Cache with query

- Now previous in caching, data we fetch with the help of hibernate , what if we want to fetch data with the help of query. E.g. HQL (Hibernate Query Language)
- Instead of fetch data using get method we use query
- 

```java
package com.telusko.DemoHib5;

import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alian a = null;
        Configuration con = new Configuration().configure().addAnnotatedClass(Alian.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session1 = sf.openSession();
        Transaction tx = session1.beginTransaction();
        Query q1 = session1.createQuery("from Alian where aid=100");
        a = (Alian)q1.uniqueResult();
        System.out.println(a);
        session1.getTransaction().commit();
        session1.close();

        Session session2 = sf.openSession();
        session2.beginTransaction();
        Query q2 = session2.createQuery("from Alian where aid=100");
        a = (Alian)q2.uniqueResult();
        System.out.println(a);
        session2.getTransaction().commit();
        session2.close();
```

- 

```
Problems  Javadoc  Declaration  Console  Progress
App (4) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (13-Dec-2023, 6:36:33 pm) [pid:
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Dec 13, 2023 6:36:34 PM org.hibernate.cache.ehcache.internal.EhcacheRegionFactory createCache
WARN: HHH90001006: Missing cache[com.telusko.DemoHib5.Alian] was created on-the-fly. The created cache will use a provider-specific default conf
Dec 13, 2023 6:36:34 PM org.hibernate.cache.spi.support.EntityReadOnlyAccess <init>
WARN: HHH90001003: Read-only caching was requested for mutable entity [NavigableRole[com.telusko.DemoHib5.Alian]]
Dec 13, 2023 6:36:34 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProv
Dec 13, 2023 6:36:34 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_, alian0_.aname as aname2_0_, alian0_.color as color3_0_ from alien_table alian0_ where alian0_.aid=100
Alian [aid=100, aname=Nilesh, color=Green]
Hibernate: select alian0_.aid as aid1_0_, alian0_.aname as aname2_0_, alian0_.color as color3_0_ from alien_table alian0_ where alian0_.aid=100
Alian [aid=100, aname=Nilesh, color=Green]
```

- 
- Here we got two query q1 and q2, for same query we get two query, even in cfg.congiguration we enable 2nd Level Cache, because this configuration is hibernate get method not for hibernate query, for that we have to add one more configuration “use_query_cache” by default it is false you make it true. And Query (q1,q2) object specific q1.setCachable(true).

Alian.java | App.java | hibernate.cfg.xml | DemoHib5/pom.xml

```
-//Hibernate/Hibernate Configuration DTD 3.0//EN (doctype with catalog)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.password">Student@1997</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/new_schema</property>
        <property name="hibernate.connection.username">student</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property name="hibernate.hbm2ddl.auto">update</property>
        <property name="hibernate.show_sql">true</property>
        <property name="hibernate.cache.use_second_level_cache">true</property>
        <property name="hibernate.cache.region.factory_class">org.hibernate.cache.ehcache.EhCacheRegionFactory</property>
        <property name="hibernate.cache.provider_class">net.sf.ehcache.hibernate.EhCacheProvider</property>
        <property name="hibernate.cache.use_query_cache"> true </property>
    </session-factory>
</hibernate-configuration>
```

```
public static void main( String[] args )
{
    Alian a = null;
    Configuration con = new Configuration().configure().addAnnotatedClass(Alian.class);
    SessionFactory sf = con.buildSessionFactory();
    Session session1 = sf.openSession();
    Transaction tx = session1.beginTransaction();
    Query q1 = session1.createQuery("from Alian where aid=100");
    q1.setCacheable(true);
    a = (Alian)q1.uniqueResult();
    System.out.println(a);
    session1.getTransaction().commit();
    session1.close();

    Session session2 = sf.openSession();
    session2.beginTransaction();
    Query q2 = session2.createQuery("from Alian where aid=100");
    q2.setCacheable(true);
    a = (Alian)q2.uniqueResult();
    System.out.println(a);
    session2.getTransaction().commit();
    session2.close();
```

- Now I run this project , only one time query is fire for both session

Problems | Javadoc | Declaration | Console | Progress

```
App (4) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-1047/jre/bin/java (13-Dec-2023, 6:42:44 pm) [pid: 1
Dec 13, 2023 6:42:45 PM org.hibernate.cache.ehcache.internal.EhcacheRegionFactory createCache
WARN: HHH90001006: Missing cache[default-query-results-region] was created on-the-fly. The created cache will use a provider-specific default con
Dec 13, 2023 6:42:45 PM org.hibernate.cache.ehcache.internal.EhcacheRegionFactory createCache
WARN: HHH90001006: Missing cache[com.telusko.DemoHib5.Alian] was created on-the-fly. The created cache will use a provider-specific default confi
Dec 13, 2023 6:42:45 PM org.hibernate.cache.spi.support.EntityReadOnlyAccess <init>
WARN: HHH90001003: Read-only caching was requested for mutable entity [NavigableRole[com.telusko.DemoHib5.Alian]]
Dec 13, 2023 6:42:45 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProvi
Dec 13, 2023 6:42:45 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alian0_.aid as aid1_0_, alian0_.aname as aname2_0_, alian0_.color as color3_0_ from alien_table alian0_ where alian0_.aid=100
Alian [aid=100, aname=Nilesh, color=Green]
Alian [aid=100, aname=Nilesh, color=Green]
```

## Hibernate Query Language (ref DemoHib6 project)

In hibernate we can do save,update,drop,update but why we need this extra language, because if you remember “session.get(Alian.class,101)”, it fetch entire row, what is we want some specific value from this row, not entire record, that case we need HQL.

- SQL vs HQL
- 

- Suppose you want to fetch entire table how SQL vs HQL
- 

- In SQL we fire query result will get in result set, but in HQL we get result in List of object
- 


**Note: You can use SQL in hiber it is called Native Query**

- I creating Class Student , from that I put 50 entries:

App.java | Student.java | hibernate.cfg.xml

```
-//Hibernate/Hibernate Configuration DTD 3.0//EN (doctype with catalog)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.password">Student@1997</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/new_schema</property>
        <property name="hibernate.connection.username">student</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property name="hibernate.hbm2ddl.auto">create</property>
        <property name="hibernate.show_sql">true</property>
    </session-factory>
</hibernate-configuration>
```

App.java | Student.java | hibernate.cfg.xml

```
import javax.persistence.Id;

@Entity
public class Student
{
    @Id
    private int rollno;
    private String name;
    private int marks;
    public int getRollno() {
        return rollno;
    }
    public void setRollno(int rollno) {
        this.rollno = rollno;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getMarks() {
        return marks;
    }
    public void setMarks(int marks) {
        this.marks = marks;
    }
    @Override
    public String toString() {
        return "Student [rollno=" + rollno + ", name=" + name + ", marks=" + marks + "]";
    }

}
```

App.java | Student.java | hibernate.cfg.xml

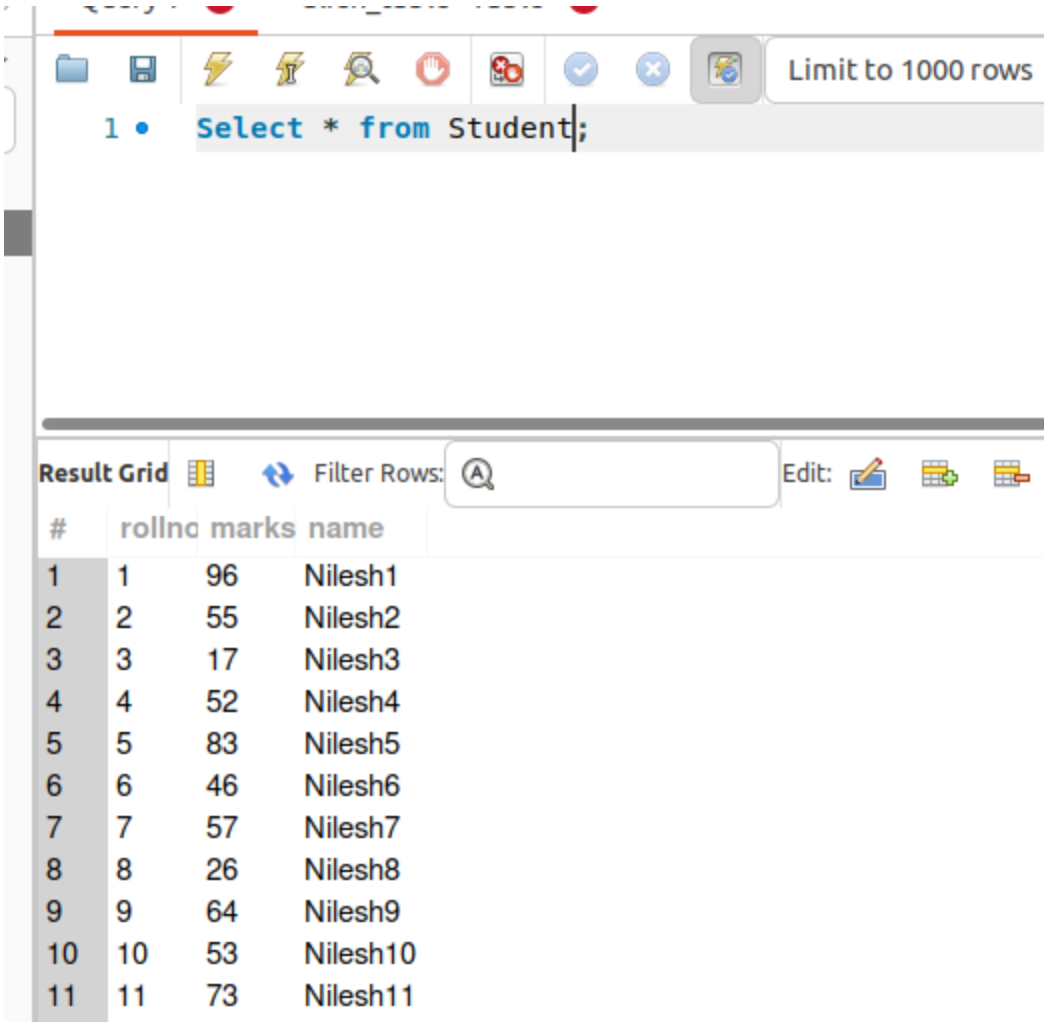
```java
package com.telusko.DemoHib6;

import java.util.Random;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;


/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Random r = new Random();
        for(int i=1;i<=50;i++)
        {
            Student s = new Student();
            s.setRollno(i);
            s.setName("Nilesh"+i);
            s.setMarks(r.nextInt());
            session.save(s);
        }
        session.getTransaction().commit();
    }
}
```

- 

Problems | Javadoc | Declaration | Console | Progress

```
<terminated> App (5) [Java Application] /snap/eclipse/73/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.8.v20230831-104
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmer
Hibernate: create table Student (rollno integer not null, marks integer not null, name varchar(255), primary key (rollr
Dec 13, 2023 7:04:12 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIs
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmer
Dec 13, 2023 7:04:12 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatfor
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
Hibernate: insert into Student (marks, name, rollno) values (?, ?, ?)
```

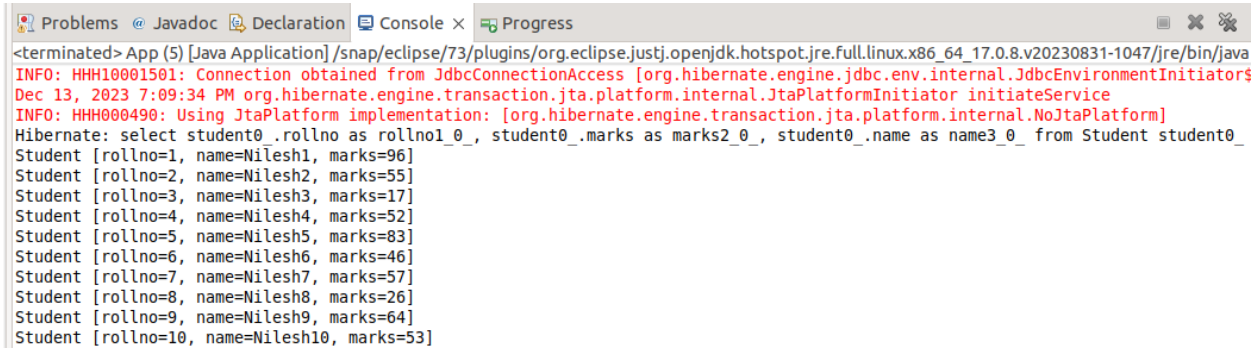
- 

- 


| # | rollno | marks | name |
|---|---|---|---|
| 1 | 1 | 96 | Nilesh1 |
| 2 | 2 | 55 | Nilesh2 |
| 3 | 3 | 17 | Nilesh3 |
| 4 | 4 | 52 | Nilesh4 |
| 5 | 5 | 83 | Nilesh5 |
| 6 | 6 | 46 | Nilesh6 |
| 7 | 7 | 57 | Nilesh7 |
| 8 | 8 | 26 | Nilesh8 |
| 9 | 9 | 64 | Nilesh9 |
| 10 | 10 | 53 | Nilesh10 |
| 11 | 11 | 73 | Nilesh11 |

- **Now we want to print all this record**

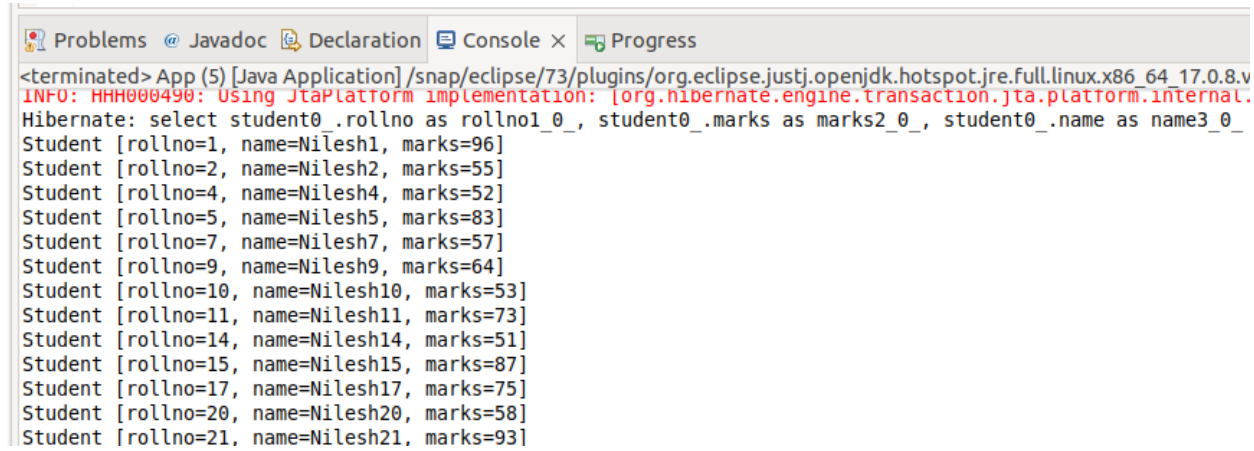
```
14  */
15 public class App
16 {
17     public static void main( String[] args )
18     {
19         Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
20         SessionFactory sf = con.buildSessionFactory();
21         Session session = sf.openSession();
22         org.hibernate.Transaction tx = session.beginTransaction();
23         Query q1 = session.createQuery("from Student");
24         List<Student> student = q1.list();
25         for(Student s : student)
26         {
27             System.out.println(s);
28         }
29         session.getTransaction().commit();
30     }
31 }
32
```




- 
- **Suppose I want student who score more than 50**

```
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Query q1 = session.createQuery("from Student where marks > 50");
        List<Student> student = q1.list();
        for(Student s : student)
        {
            System.out.println(s);
        }
        session.getTransaction().commit();
    }
}
```

- 




- 
- **I want student who roll number is 7**
- This query give one single values, so it not return list

```
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Query q1 = session.createQuery("from Student where rollno = 7");
        Student student = (Student) q1.uniqueResult();
        System.out.println(student);
        session.getTransaction().commit();
    }
}
```

```
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 13, 2023 7:13:51 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnec
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$
Dec 13, 2023 7:13:51 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select student0_.rollno as rollno1_0_, student0_.marks as marks2_0_, student0_.name as name3_0_ from Student student0_ w
Student [rollno=7, name=Nilesh7, marks=57]
```

Now what id we want to fetch specific column, in query we directly not use column

```
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Query q1 = session.createQuery("select rollno,marks,name from Student where rollno = 7");
        Student student = (Student) q1.uniqueResult();
        System.out.println(student);
        session.getTransaction().commit();
    }
}
```

```
Dec 13, 2023 7:16:32 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJ
Dec 13, 2023 7:16:32 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select student0_.rollno as col_0_0_, student0_.marks as col_1_0_, student0_.name as col_2_0_ from Student student0_ where student0_.rollno=
Exception in thread "main" java.lang.ClassCastException: class [Ljava.lang.Object; cannot be cast to class com.telusko.DemoHib6.Student ([Ljava.lang.C
        at com.telusko.DemoHib6.App.main(App.java:24)
```

- Why it showing this error, because when you specifying “Select” it mean that your fetching the column name, that case it not returning entire list, it return you object, object can be rollno, name or marks, if all 3 are there than that case you have to create array of object
- It will give me all 3 column value separately

```java
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Query q1 = session.createQuery("select rollno,marks,name from Student where rollno = 7");
        Object[] object = (Object[])q1.uniqueResult();
        for(Object o: object)
        {
            System.out.println(o);
        }
        session.getTransaction().commit();
    }
}
```

```
Dec 13, 2023 7:20:25 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select student0_.rollno as col_0_0_, student0_.marks as col_1_0_, student0_.name as col_2_0_ from Student student0_ where student0_.rollno=7
7
57
Nilesh7
```

- Now what if where clause is remove, that case it will give entire table of that 3 columns so, it return list of object, here we get list so uniqueresult not possible

```java
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Query q1 = session.createQuery("select rollno,marks,name from Student");
        List<Object[]> objects = (List<Object[]>)q1.list();

        for(Object[] obj: objects)
        {
            System.out.println(obj[0]+" : "+obj[1]+" : "+obj[2]);
        }
        session.getTransaction().commit();
    }
}
```

```
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiat
Dec 13, 2023 7:25:29 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select student0_.rollno as col_0_0_, student0_.marks as col_1_0_, student0_.name as col_2_0_ from Student student0_
1 : 96 : Nilesh1
2 : 55 : Nilesh2
3 : 17 : Nilesh3
4 : 52 : Nilesh4
5 : 83 : Nilesh5
6 : 46 : Nilesh6
7 : 57 : Nilesh7
8 : 26 : Nilesh8
9 : 64 : Nilesh9
10 : 53 : Nilesh10
```

- Marks > 60

```java
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Query q1 = session.createQuery("select rollno,marks,name from Student s where s.marks>60");
        List<Object[]> objects = (List<Object[]>)q1.list();

        for(Object[] obj: objects)
        {
            System.out.println(obj[0]+" : "+obj[1]+" : "+obj[2]);
        }
        session.getTransaction().commit();
    }
}
```

- 

```
11 : 73 : Nilesh11
15 : 87 : Nilesh15
17 : 75 : Nilesh17
21 : 93 : Nilesh21
22 : 67 : Nilesh22
26 : 73 : Nilesh26
28 : 84 : Nilesh28
31 : 68 : Nilesh31
35 : 88 : Nilesh35
38 : 77 : Nilesh38
47 : 63 : Nilesh47
49 : 93 : Nilesh49
50 : 77 : Nilesh50
```

- 
- Sum of marks, here we know it return single value that case we can also use uniqreResult(), we know object is type Integer, we can explicitly specify Integer object

```java
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Query q1 = session.createQuery("select sum(marks) from Student s where s.marks>60");
        List<Object> objects = (List<Object>)q1.list();

        for(Object obj: objects)
        {
            System.out.println(obj);
        }
        session.getTransaction().commit();
    }
}
```

- 

```
Dec 13, 2023 7:29:41 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator in
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.inte
Hibernate: select sum(student0_.marks) as col_0_0_ from Student student0_ where student0_.marks>60
1261
```

- 

- If you given where clause input from user, like prepared statement, remember bydefault hibernate giving prepared statement

```
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        int b = 60;
        org.hibernate.Transaction tx = session.beginTransaction();

        Query q1 = session.createQuery("select sum(marks) from Student s where s.marks> :b");
        q1.setParameter("b", b);
        List<Object> objects = (List<Object>)q1.list();

        for(Object obj: objects)
        {
            System.out.println(obj);
        }
        session.getTransaction().commit();
    }
```

-
```
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.Jdb
Dec 13, 2023 7:38:18 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiate
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.N
Hibernate: select sum(student0_.marks) as col_0_0_ from Student student0_ where student0_.marks>?
1261
```

-

- **If you want use SQL in HQL // this sql called here Native Query**
- We want all student marks > 60, here we are fetching entire column, so we know list is going to type “Student”, in addEntity we mentioned “Student.class”

```
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Student.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        int b = 60;
        org.hibernate.Transaction tx = session.beginTransaction();

        SQLQuery q1 = session.createSQLQuery("Select * from Student where marks > 60");
        q1.addEntity(Student.class);
        List<Student> student = q1.list();
        for(Student s : student)
        {
            System.out.println(s);
        }
        session.getTransaction().commit();
    }
```

-

```
<terminated> App (5) [Java Application] /snap/eclipse/73/plugins/org.eclipse
Dec 13, 2023 7:50:40 PM org.hibernate.engine.transaction.jta.pla
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernat
Hibernate: Select * from Student where marks > 60
Student [rollno=1, name=Nilesh1, marks=96]
Student [rollno=5, name=Nilesh5, marks=83]
Student [rollno=9, name=Nilesh9, marks=64]
Student [rollno=11, name=Nilesh11, marks=73]
Student [rollno=15, name=Nilesh15, marks=87]
Student [rollno=17, name=Nilesh17, marks=75]
Student [rollno=21, name=Nilesh21, marks=93]
Student [rollno=22, name=Nilesh22, marks=67]
Student [rollno=26, name=Nilesh26, marks=73]
Student [rollno=28, name=Nilesh28, marks=84]
```

- Suppose We want only name and marks, that case we are not fetch entire columns so, we can't use "Student" list and similarly we can't mentioned Entity type "Student".
- Here have to convert our result in Map <key,value>, that way you can access it.

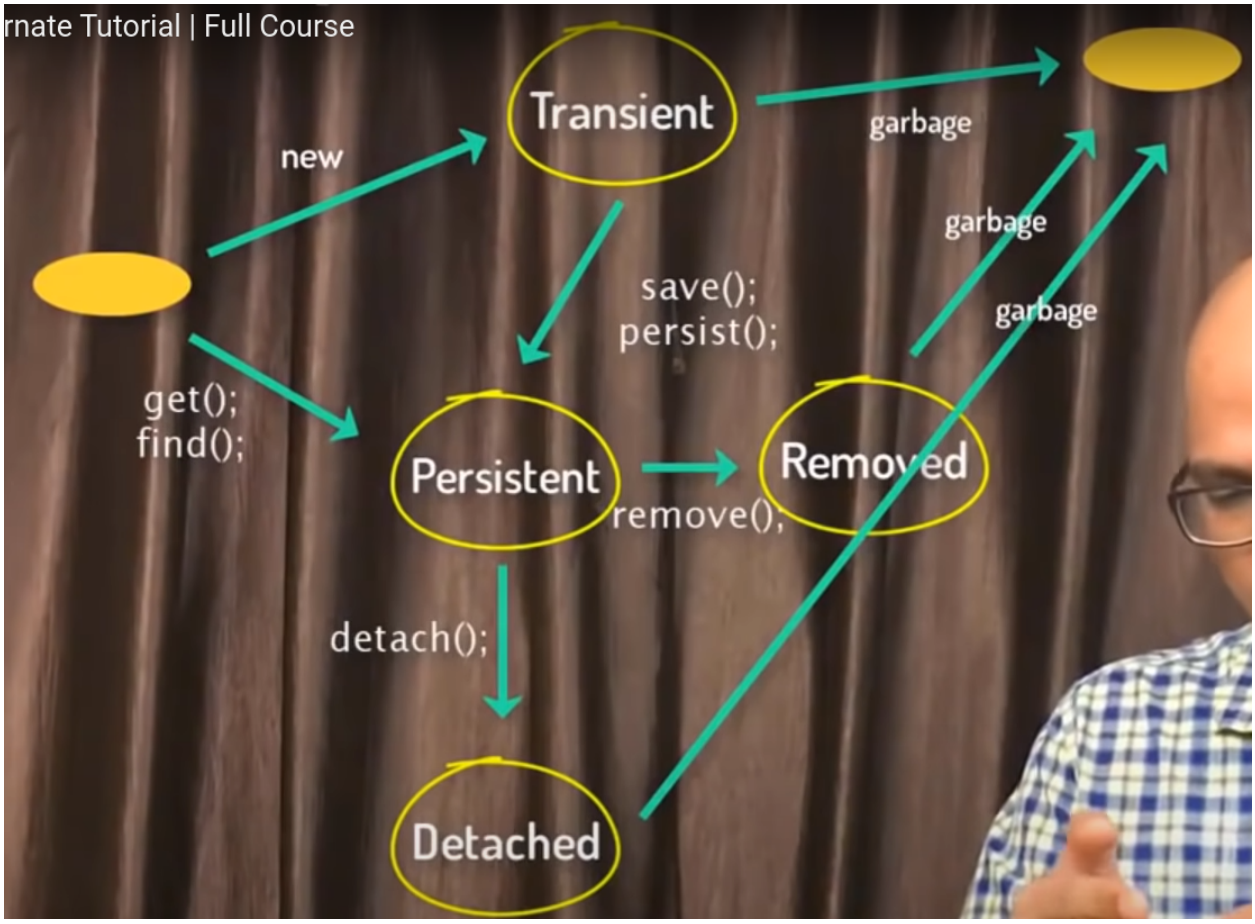
```
        org.hibernate.Transaction tx = session.beginTransaction();

        SQLQuery q1 = session.createSQLQuery("Select name,marks from Student where marks > 60");
        q1.setResultTransformer(Criteria.ALIAS_TO_ENTITY_MAP); // convert obj --> into map
        List students = q1.list();
        for(Object o : students)
        {
            Map m = (Map) o; //typecast
            System.out.println(m.get("name")+" : "+m.get("marks"));
        }
        session.getTransaction().commit();
    }
}
```

```
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine
Hibernate: Select name,marks from Student where marks > 60
Nilesh1 : 96
Nilesh5 : 83
Nilesh9 : 64
Nilesh11 : 73
Nilesh15 : 87
Nilesh17 : 75
Nilesh21 : 93
Nilesh22 : 67
Nilesh26 : 73
```

## HIbernate Object state / Persistence Life Cycle (ref DemoHib7)

Whenever we working with object, we do save, update, delete it mean object changing the state.

1. When we create object then it is in **“New”** state
2. When object contain data, if you directly close the program than object hold data will gone so that state called **“Transient”** state
3. If you want to store object data permanently that state called **“Persistence”** state, In persistence state whatever you do with object it always store in DB
4. Suppose you are in persistence state, that mean any change in object that reflect on DB as well, but you don’t want to store that changes as of now in DB, that case you can push your object in “Detach” state.
5. You are in a persistent state and you remove data from DB by using remove() method, so data is present in java object but not in DB.
6. If you are in “new” state and you want to fetch data from DB, so you no need to go in a persistent state, you can directly use two methods from the “new” state.
    a. get()
    b. find()



Laptop.java | hibernate.cfg.xml | App.java ×

```java
package com.telusko.DemoHib7;

import java.util.Random;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Random rv = new Random();
        for(int i=1;i<=50;i++)
        {
            Laptop lp = new Laptop();
            lp.setLid(i);;
            lp.setBrand("Dell"+i);
            int x = rv.nextInt(1000);
            lp.setPrice(x);
            session.save(lp);
        }

        session.getTransaction().commit();
    }
}

```

Laptop.java | hibernate.cfg.xml × | App.java

```xml
-//Hibernate/Hibernate Configuration DTD 3.0//EN (doctype with catalog)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.password">Student@1997</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/new_schema</property>
        <property name="hibernate.connection.username">student</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
    <property name="hibernate.hbm2ddl.auto">update</property>
    <property name="hibernate.show_sql">true</property>
    </session-factory>
</hibernate-configuration>
```

Laptop.java × hibernate.cfg.xml App.java

```java
package com.telusko.DemoHib7;

import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Laptop
{
    @Id
    private int lid;
    private String brand;
    private int price;
    public int getLid() {
        return lid;
    }
    public void setLid(int lid) {
        this.lid = lid;
    }
    public String getBrand() {
        return brand;
    }
    public void setBrand(String brand) {
        this.brand = brand;
    }
    public int getPrice() {
        return price;
    }
    public void setPrice(int price) {
        this.price = price;
    }
    @Override
    public String toString() {
        return "Laptop [lid=" + lid + ", brand=" + brand + ", price=" + price + "]";
    }

}
```

Result Grid  Filter Rows:

| # | lid | brand | price |
|---|---|---|---|
| 1 | 1 | Dell1 | 314 |
| 2 | 2 | Dell2 | 983 |
| 3 | 3 | Dell3 | 699 |
| 4 | 4 | Dell4 | 535 |
| 5 | 5 | Dell5 | 740 |
| 6 | 6 | Dell6 | 296 |
| 7 | 7 | Dell7 | 847 |
| 8 | 8 | Dell8 | 485 |
| 9 | 9 | Dell9 | 577 |
| 10 | 10 | Dell10 | 541 |
| 11 | 11 | Dell11 | 399 |
| 12 | 12 | Dell12 | 997 |
| 13 | 13 | Dell13 | 322 |
| 14 | 14 | Dell14 | 894 |
| 15 | 15 | Dell15 | 763 |

Here i have 1 to 50 entries, Now suppose I want to store 51 entry in DB,

```
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lp = new Laptop();
        lp.setLid(51);
        lp.setBrand("Sony");
        lp.setPrice(700);

        session.getTransaction().commit();
    }
}
```

Now our object is in Transient state, because it not store yet, when  I Stop the application I will loss data.

```
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lp = new Laptop();
        lp.setLid(51);
        lp.setBrand("Sony");
        lp.setPrice(700);
        session.save(lp);
        session.getTransaction().commit();
    }
}
```

Here When I save object in DB than it goes to persistence state, but only when you made it commit.

```
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lp = new Laptop();
        lp.setLid(51);
        lp.setBrand("Sony");
        lp.setPrice(700);
        session.save(lp);
        lp.setPrice(500);
        session.getTransaction().commit();
    }
}
```

Now here, after saving the object we again set price of that object, now question is what is saved values, you can say 700, because 700 value is save not 500, but actually it store 500, because our object is in “Persistant” state now, whatever you make change it will store in DB. look at the consol query, it fire 2 query insert and update

```
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.h
Dec 21, 2023 7:52:09 PM org.hibernate.engine.transaction.jta.platform.i
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engin
Hibernate: insert into Laptop (brand, price, lid) values (?, ?, ?)
Hibernate: update Laptop set brand=?, price=? where lid=?
```

Suppose you want to make some change in program but not store int DB, look following example, we can use detach(), here 9000 value not get store in DB

```
13  */
14 public class App
15 {
16     public static void main( String[] args )
17     {
18         Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
19         SessionFactory sf = con.buildSessionFactory();
20         Session session = sf.openSession();
21         org.hibernate.Transaction tx = session.beginTransaction();
22         Laptop lp = new Laptop();
23         lp.setLid(53);
24         lp.setBrand("Jio");
25         lp.setPrice(700);
26         session.save(lp);
27         lp.setPrice(500);
28 
29         session.detach(lp);
30         lp.setPrice(9000);
31         session.getTransaction().commit();
32     }
33 }
34 
```

Here detach use before commit, so no any value will get store in DB, we have to detach object after commit, now 9000 value only value not get store in DB

```
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lp = new Laptop();
        lp.setLid(53);
        lp.setBrand("Jio");
        lp.setPrice(700);
        session.save(lp);
        lp.setPrice(500);
        session.getTransaction().commit();
        session.detach(lp);
        lp.setPrice(9000);
    }
}
```

| | | | |
|---|---|---|---|
| 50 | 50 | Dell50 | 238 |
| 51 | 51 | Sony | 700 |
| 52 | 52 | Sony | 500 |
| * | NULL | NULL | NULL |

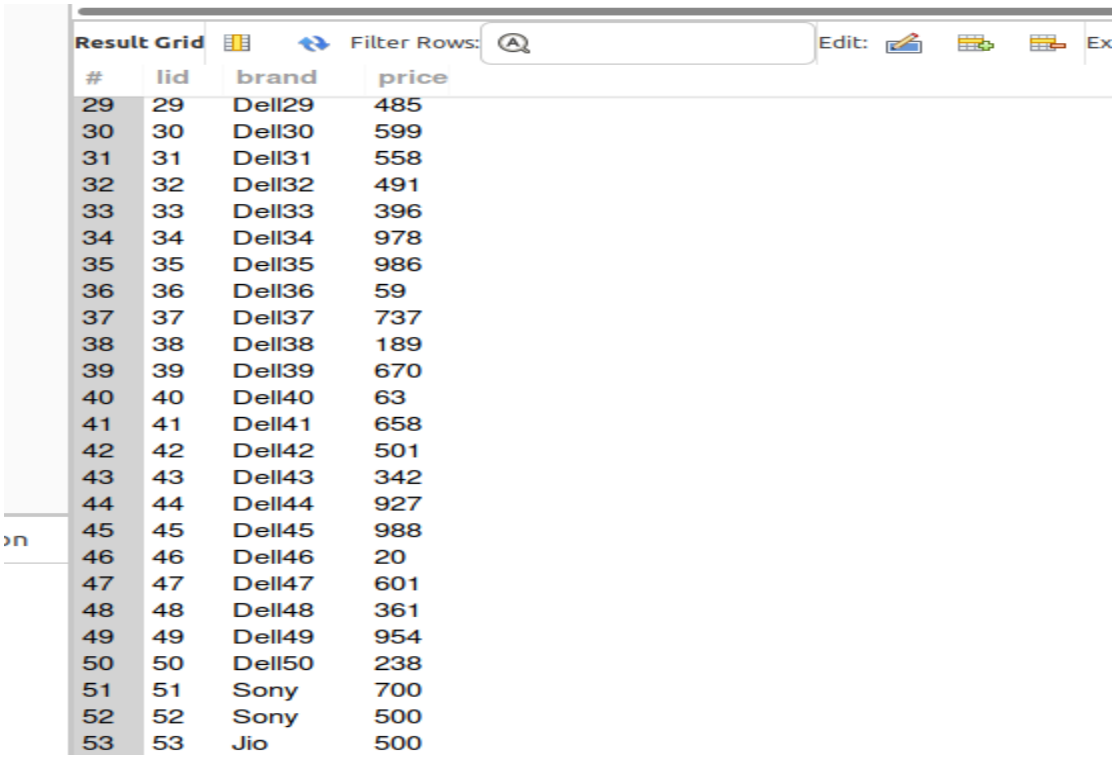
Now when you sait, session.remove(), that object goes to remove state, no entry in DB

```
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lp = new Laptop();
        lp.setLid(54);
        lp.setBrand("Jio");
        lp.setPrice(7000);
        session.save(lp);
        session.remove(lp);
        session.getTransaction().commit();

    }
}
```

| | | | |
|---|---|---|---|
| 51 | 51 | Sony | 700 |
| 52 | 52 | Sony | 500 |
| 53 | 53 | Jio | 500 |
| * | NULL | NULL | NULL |

# Hibernate Get vs Load (ref DemoHib7.. Continue )

Suppose I want to fetch data,  following data store in DB



Result Grid | Filter Rows: | Edit:

| # | lid | brand | price |
|---|---|---|---|
| 29 | 29 | Dell29 | 485 |
| 30 | 30 | Dell30 | 599 |
| 31 | 31 | Dell31 | 558 |
| 32 | 32 | Dell32 | 491 |
| 33 | 33 | Dell33 | 396 |
| 34 | 34 | Dell34 | 978 |
| 35 | 35 | Dell35 | 986 |
| 36 | 36 | Dell36 | 59 |
| 37 | 37 | Dell37 | 737 |
| 38 | 38 | Dell38 | 189 |
| 39 | 39 | Dell39 | 670 |
| 40 | 40 | Dell40 | 63 |
| 41 | 41 | Dell41 | 658 |
| 42 | 42 | Dell42 | 501 |
| 43 | 43 | Dell43 | 342 |
| 44 | 44 | Dell44 | 927 |
| 45 | 45 | Dell45 | 988 |
| 46 | 46 | Dell46 | 20 |
| 47 | 47 | Dell47 | 601 |
| 48 | 48 | Dell48 | 361 |
| 49 | 49 | Dell49 | 954 |
| 50 | 50 | Dell50 | 238 |
| 51 | 51 | Sony | 700 |
| 52 | 52 | Sony | 500 |
| 53 | 53 | Jio | 500 |

1. Get() method

```
 */
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lap = new Laptop();
        session.get(Laptop.class, 50);
        System.out.println(lap);
        session.getTransaction().commit();

    }
}
```

```
Dec 21, 2023 8:12:43 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select laptop0_.lid as lid1_0_0_, laptop0_.brand as brand2_0_0_, laptop0_.price as price3_0_0_ from Laptop laptop0_ where laptop0_.lid=?
Laptop [lid=0, brand=null, price=0]
```

2. Load() method

```
*/
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lap = new Laptop();
        Laptop l = (Laptop) session.load(Laptop.class, 50);
        System.out.println(l);
        session.getTransaction().commit();

    }
```

```
Dec 21, 2023 8:17:26 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select laptop0_.lid as lid1_0_0_, laptop0_.brand as brand2_0_0_, laptop0_.price as price3_0_0_ from Laptop laptop0_ where laptop0_.lid=?
Laptop [lid=50, brand=Dell50, price=238]
```

It also give same result, but why we need both if both are giving same result

Main difference between both method is performance and type of exception handle by both.

In **get() method** , it hit database everytime even you use it or not, see following example we are not printing object means we are not actually using it

```
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lap = new Laptop();
        Laptop l = (Laptop) session.get(Laptop.class, 50);
        //System.out.println(l);
        session.getTransaction().commit();

    }
}
```

```
Dec 21, 2023 8:20:35 PM org.hibernate.resource.transaction.backend.jdbc.internal.JdbcIsolationDelegate$JdbcIsolationNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProvider
Dec 21, 2023 8:20:35 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select laptop0_.lid as lid1_0_0_, laptop0_.brand as brand2_0_0_, laptop0_.price as price3_0_0_ from Laptop laptop0_ where laptop0_.lid=?
```

You can see, it still fire query,so it will impact the performance.

But in **load() Method** , it will fire query only when you are using that object. See the following examples

It fire query, see output

```java
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lap = new Laptop();
        Laptop l = (Laptop) session.load(Laptop.class, 50);
        System.out.println(l);
        session.getTransaction().commit();

    }
}
```

```
Dec 21, 2023 8:22:19 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select laptop0_.lid as lid1_0_0_, laptop0_.brand as brand2_0_0_, laptop0_.price as price3_0_0_ from Laptop laptop0_ where laptop0_.lid=?
Laptop [lid=50, brand=Dell50, price=238]
```

In the following example you can see now, if I don’t print the object then it not fire the query

```java
public class App
{
    public static void main( String[] args )
    {
        Configuration con = new Configuration().configure().addAnnotatedClass(Laptop.class);
        SessionFactory sf = con.buildSessionFactory();
        Session session = sf.openSession();
        org.hibernate.Transaction tx = session.beginTransaction();
        Laptop lap = new Laptop();
        Laptop l = (Laptop) session.load(Laptop.class, 50);
        session.getTransaction().commit();

    }
}
```

```
Dec 21, 2023 8:23:27 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 21, 2023 8:23:27 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolat
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentIni
Dec 21, 2023 8:23:27 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
```

## 1. session.load()

- It will always return a “**proxy**” (Hibernate term) without hitting the database. In Hibernate, proxy is an object with the given identifier value, its properties are not initialized yet, it just look like a temporary fake object.
- If no row found , it will throws an **ObjectNotFoundException**.

## 2. session.get()

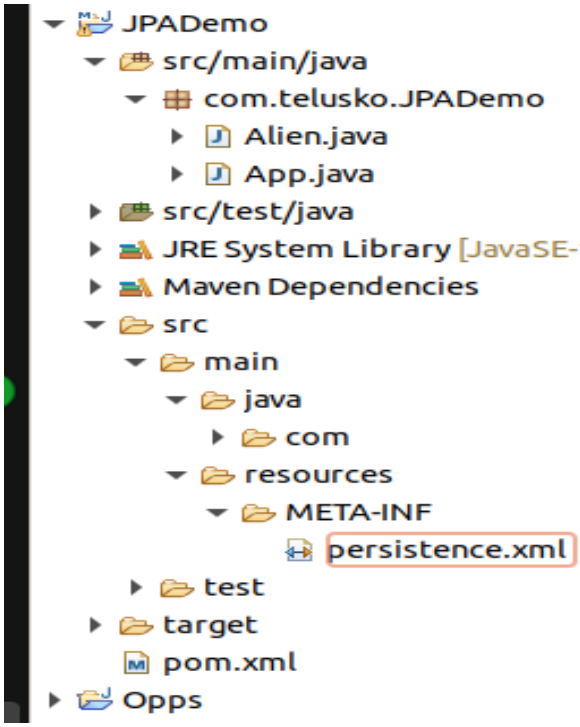
- It always **hit the database** and return the real object, an object that represent the database row, not proxy.
- If no row found , it return **null**.

# JPA in Hibernate (ref JPADemo)

- Java Persistence API (JPA)
- Some time you close application and still you want to store that data, there are so many way you can save that data
    - RDBMS
    - File
- But storing data here, there is some problem because in java everything is store in object, so need to store object. Storing object can be done using ORM tool like hibernate, iBatis or TopLink.
- But here problem is , now suppose company used hibernate for building application, after some year company need to access some feature that don't present in hibernate but available in iBatis. That case you need switch between Hibernate to iBatis, but switching now so easy here, because every ORM tool have there independent standard, to solving this problem we can use JPA.
- JPA give standards to all this ORM tools, so they all tool implemented in JPA specification. So here changing between tool become easy, here we need to just need to change tool not specification, because all are using same JPA specification.

Following configuration need for this project

1. Normal mevan project, import hibernate and mysql connect.
2. Follow industry standard for creating configuration file location so, src→main I create resources/META-INF folder and create there persistence.xml file for Database configuration.




This file structure import, else it show error, file not found, write exact same

We are using persistence API, so configuration of file is difference, se property tags.
You can see we are using persistence unit, here we create “pu” unit, you can create as many unit as you want “pu1”, “pu2” or so many…
Select any one of them according to you need.

Alien.java | App.java | persistence.xml

```
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_2.xsd (xsi:schemaLocation with catalog)
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.2"
     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_2.xsd">

    <persistence-unit name="pu">
         <properties>
            <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver" />
            <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/new_schema" />
            <property name="javax.persistence.jdbc.user" value="student" />
            <property name="javax.persistence.jdbc.password" value="Student@1997" />

            <property name="hibernate.dialect" value="org.hibernate.dialect.MySQLDialect" />
            <property name="hibernate.dialect" value="org.hibernate.dialect.MySQL8Dialect" />
            <property name="hibernate.show_sql" value="true" />
        </properties>
    </persistence-unit>
</persistence>
```

I already have table “Alien” it contains data.

Query 3

Limit to 1000 rows

```
select * from Alien;
INSERT INTO Alien (aid, aname, tech)  VALUES (5, "james", "Python");
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| # | aid | aname | tech |
|---|---|---|---|
| 1 | 1 | ABHIRAM | Java |
| 2 | 2 | Nilesh | C++ |
| 3 | 3 | Priyanshu | C |
| 4 | 4 | Krish | javaScript |
| 5 | 5 | james | Python |
| * | NULL | NULL | NULL |

It is hibernate so, class name == table name

Alien.java × persistence.xml

```java
package com.telusko.JPADemo;

import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Alien
{
    @Id
    private int aid;
    private String aname;
    private String tech;
    public int getAid() {
        return aid;
    }
    public void setAid(int aid) {
        this.aid = aid;
    }
    public String getAname() {
        return aname;
    }
    public void setAname(String aname) {
        this.aname = aname;
    }
    public String getTech() {
        return tech;
    }
    public void setTech(String tech) {
        this.tech = tech;
    }
    @Override
    public String toString() {
        return "Alien [aid=" + aid + ", aname=" + aname + ", tech=" + tech + "]";
    }

}
```

1. This way you can fetch the values.

Alien.java App.java ×

```java
package com.telusko.JPADemo;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("pu"); // unit name mentioned
                                                                                  // in persistence.xml file
        EntityManager em = emf.createEntityManager();
        Alien al = em.find(Alien.class, 2);
        System.out.println(al);
    }
}
```

```
Dec 22, 2023 7:38:24 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Dec 22, 2023 7:38:24 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL8Dialect
Dec 22, 2023 7:38:24 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionP
Dec 22, 2023 7:38:24 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select alien0_.aid as aid1_0_0_, alien0_.aname as aname2_0_0_, alien0_.tech as tech3_0_0_ from Alien alien0_ where alien0_.aid=?
Alien [aid=2, aname=Nilesh, tech=C++]
```

2. Save the values
    a. Here we are saving into DB, we have begin() and commit() the transaction then only it save the data in DB, else not.

Alien.java | App.java | persistence.xml

```java
package com.telusko.JPADemo;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        Alien al = new Alien();
        al.setAid(6);
        al.setAname("KKM");
        al.setTech("Blockchain");
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("pu"); // unit name mentioned
                                                                                  // in persistence.xml file
        EntityManager em = emf.createEntityManager();
        em.getTransaction().begin();
        em.persist(al);
        em.getTransaction().commit();
    }
}
```

Result Grid | Filter Rows:

| # | aid | aname | tech |
|---|---|---|---|
| 1 | 1 | ABHIRAM | Java |
| 2 | 2 | Nilesh | C++ |
| 3 | 3 | Priyanshu | C |
| 4 | 4 | Krish | javaScript |
| 5 | 5 | james | Python |
| 6 | 6 | KKM | Blockchain |
| 7 | 7 | Jaimin | ML |
| * | NULL | NULL | NULL |

Using JPA, you no need to change source code for changing ORM tool, simply change dependency.