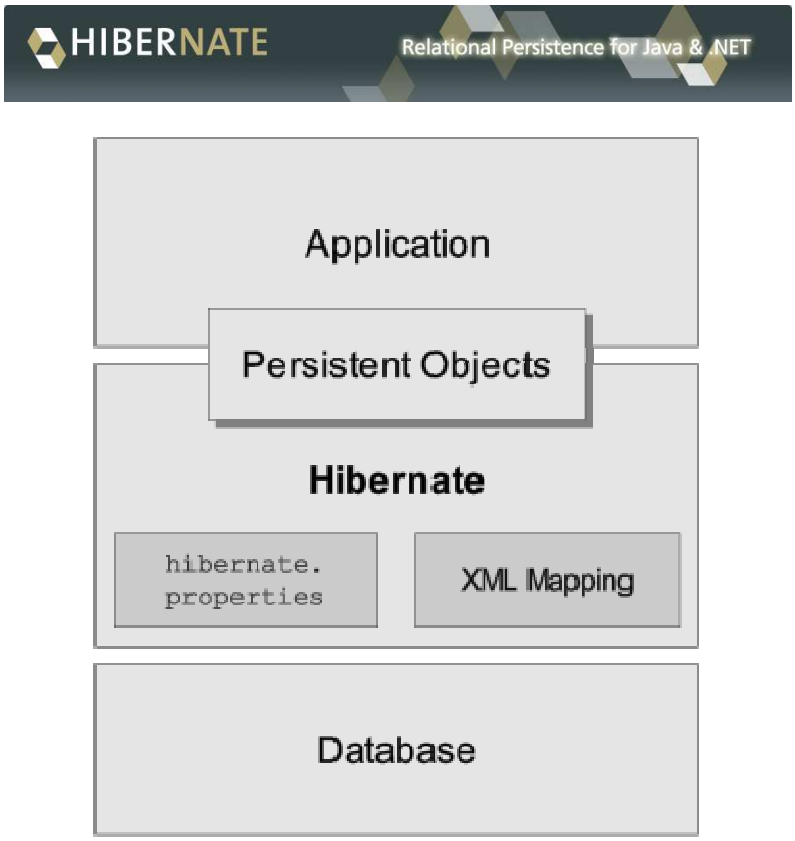
Pt5b Hibernate 2.0

Enginyeria inversa (REVerse ENGineering)

Enunciat

Amb Hibernate la forma normal de treballar és generar uns fitxers d'extensió .hbm.xml en els que es defineixen de certa forma les taules de la base de dades, les seves relacions i les classes de Java associades a aquestes taules. Un cop definits, utilitzant Hibernate i aquests fitxers de configuració, es poden crear automàticament les taules a la base de dades. Quan volem realitzar el procés contrari, es a dir, a partir de taules ja creades a la base de dades, generar els fitxers de mapatge (.hbm.xml), es parla de enginyeria inversa (REVerse ENGineering).

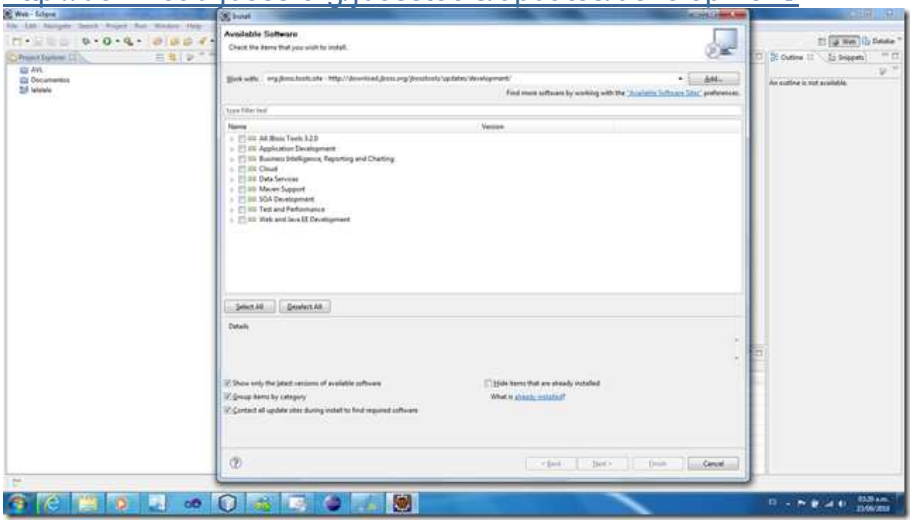


Passos a seguir

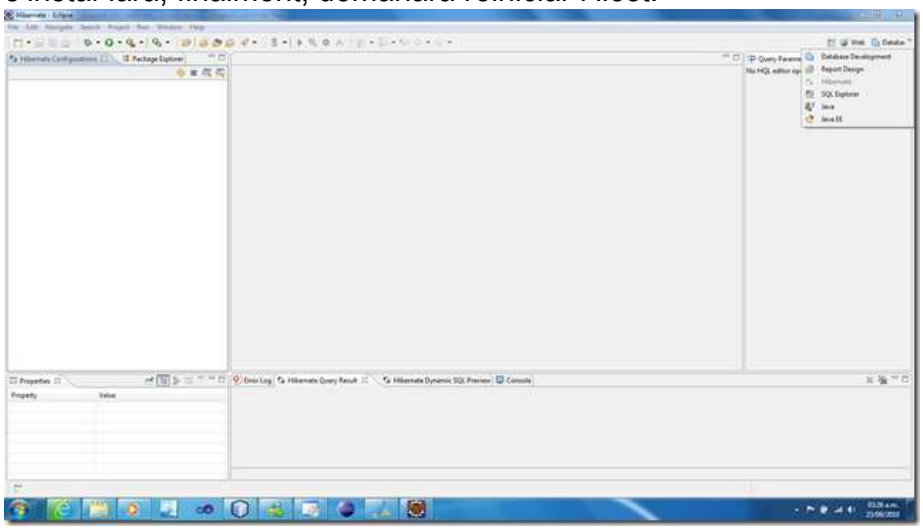
Per poder utilitzar Hibernate al teu Eclipse:

1. Instal·lació de Hibernate Tools a Eclipse.

    a. Al menú d'Eclipse, anem a `Help>Install new software`, i a l'espai on s'indica `Works with ...`, escriurem http://download.jboss.org/jbosstools/updates/development/

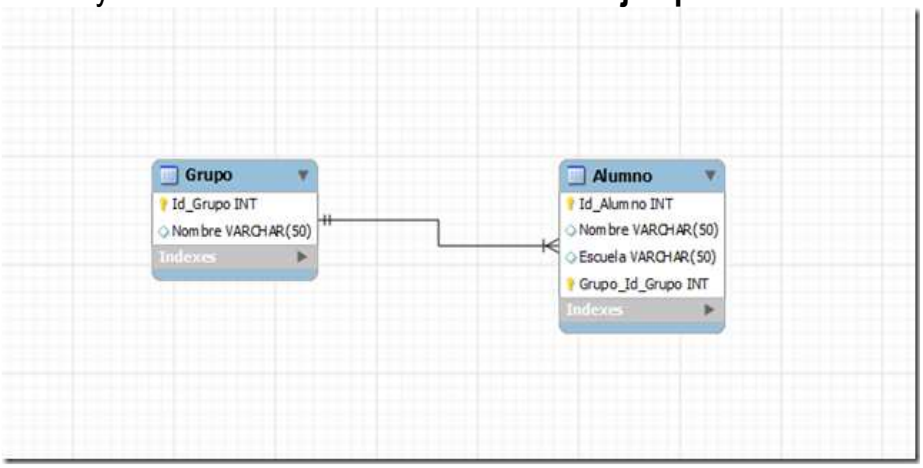


    b. Després anem on diu `Data Services>Hibernate Tools`, seleccionem, fem clic a `Next`, ens donarà la llicència i s'instal·larà, finalment, demanarà reiniciar i llest.



2. Creació de la base de dades:

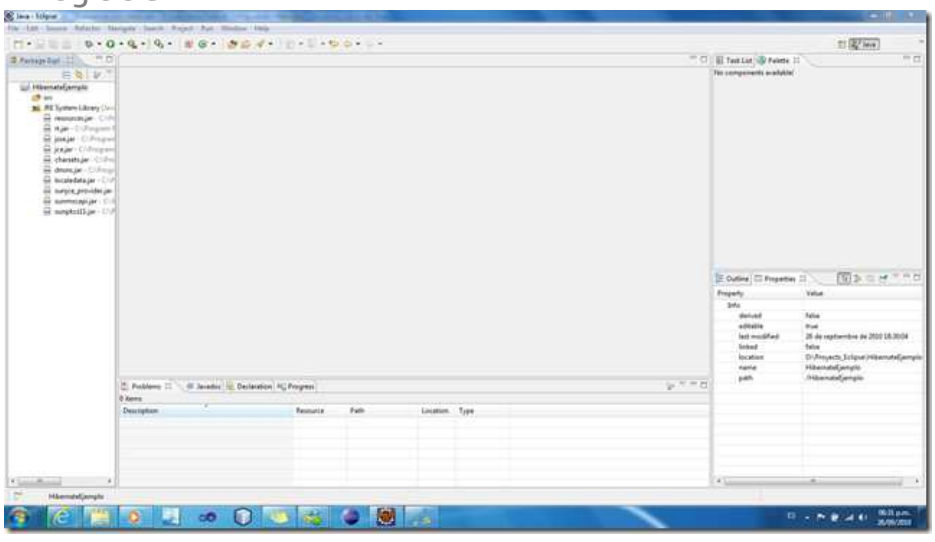
   a. Dissenyem una base de dades anomenada **Ejemplo**.



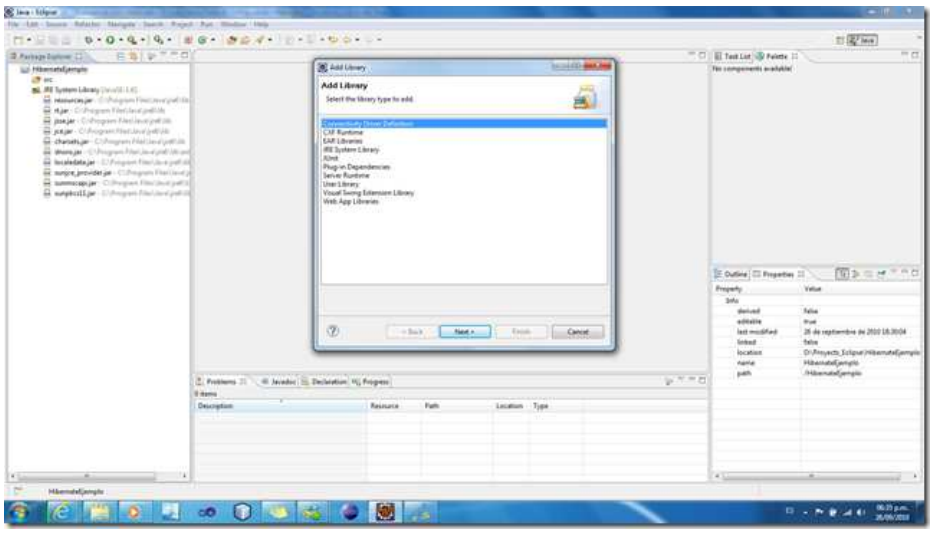
   b. Executa al SGBD MySQL l'script Pt5b.sql

3. Connectar l'Eclipse amb la base de dades per tal de crear les classes i que ens creï les classes corresponents a cada taula de la BD **Ejemplo**:

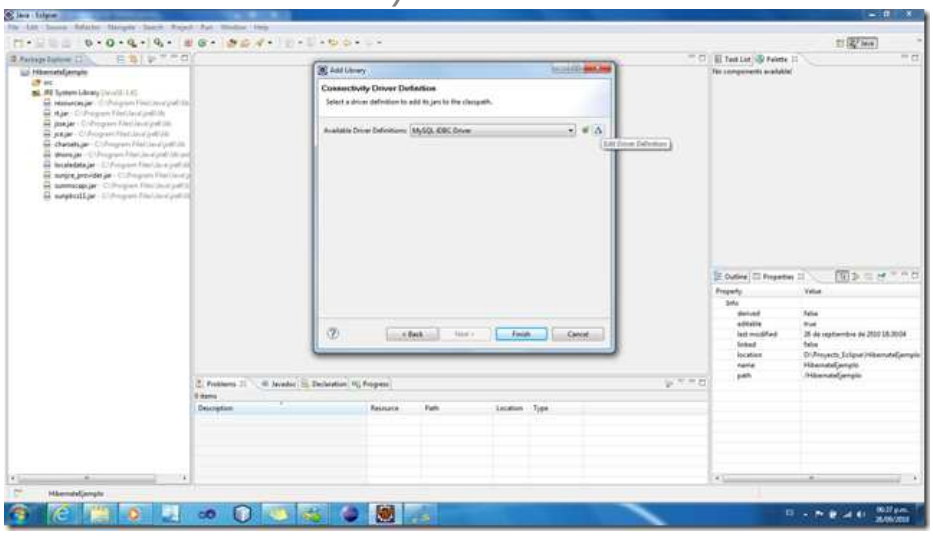
   a. Creem un nou projecte amb Eclipse: `File>New>Java Project`.



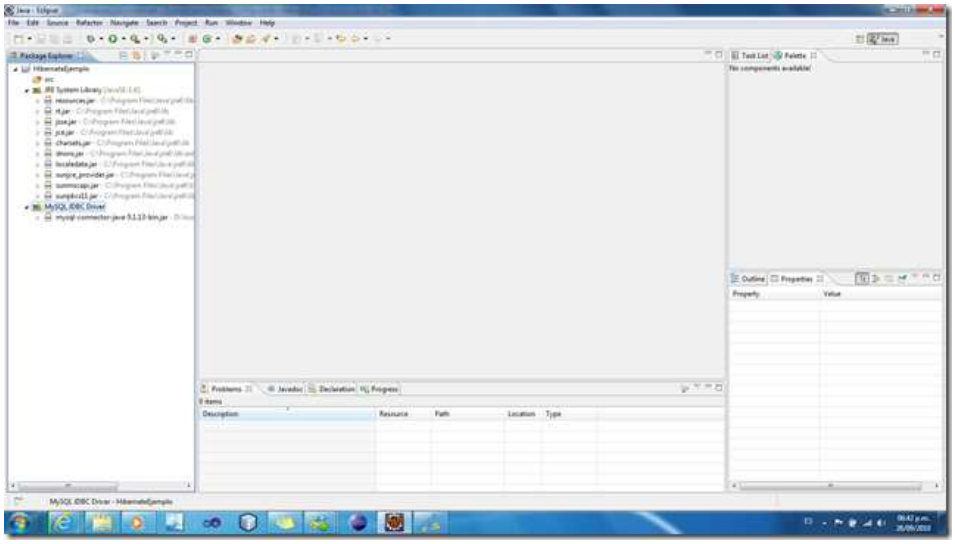
   b. **Afegim al projecte el driver de MySQL**, que tens a \\MARTE\Recursos\AMS2\M06. Seleccionem el projecte, fem clic amb botó dret anem a `Build Paths>Add Libraries`. Apareixerà una finestra com aquesta:



c. Seleccionem `Connectivity Driver Definition`, i al pas següent ens apareixen les opcions de connexió a una BD. Seleccionem **MySQL JDBC Driver** i fem clic al símbol `Δ (Edit Driver Definition)`.

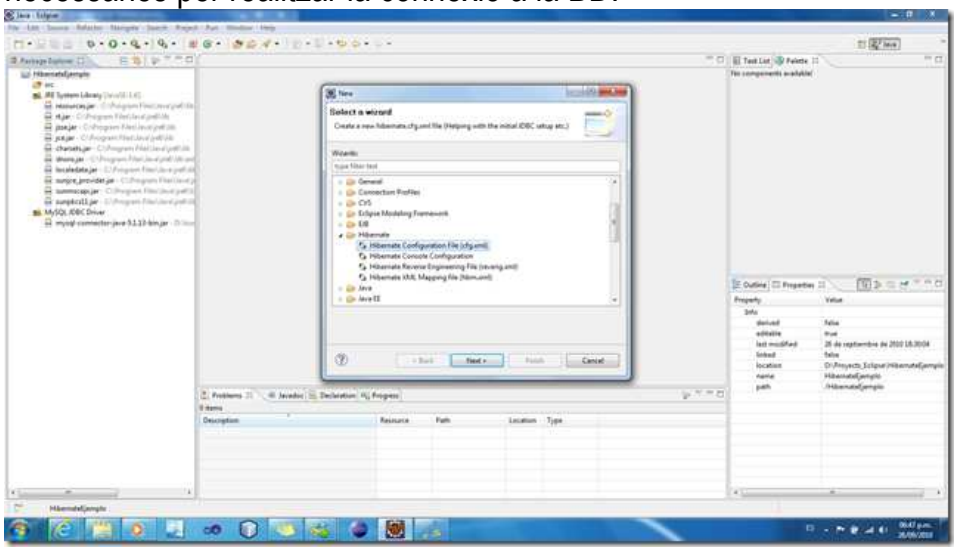


d. Ara ens apareix la finestra de configuració. Hem d'anar a la pestanya `Jar List>Add Jar/ZIP` i seleccionar el driver descarregat anteriorment. Només queda clicar a `Finish` i al projecte ha d'aparèixer quelcom semblant:

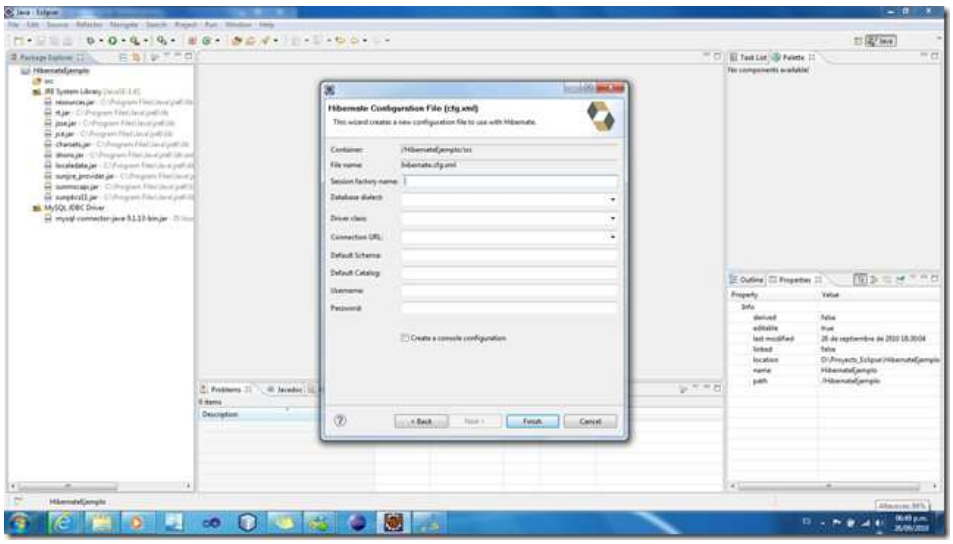


4. **Configuració del fitxer de configuració** (cfg.xml):

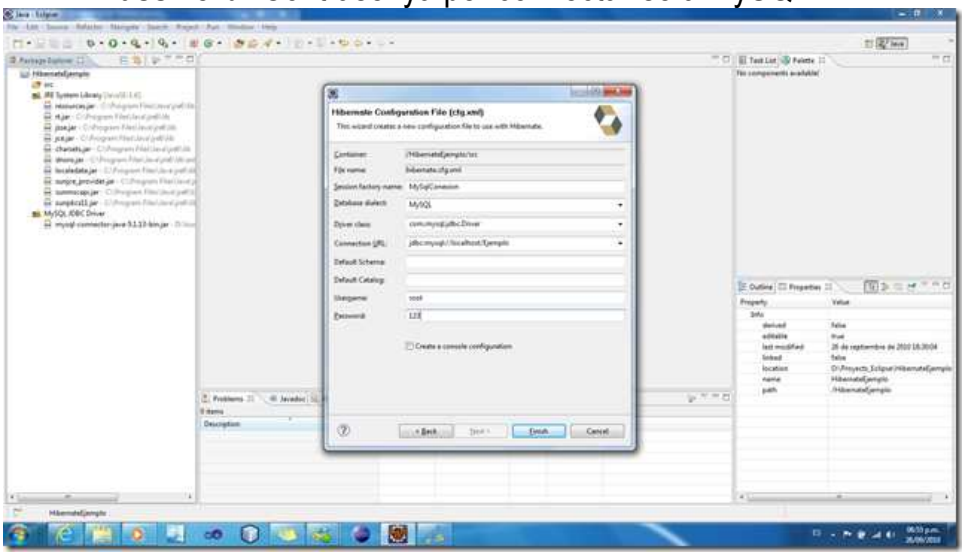
a. Seleccionem el projecte, fem clic amb el botó dret i anem a `New>Other>Hibernate Configuration File (cfg.xml)`. Aquest arxiu és un fitxer **xml** que conté les dades necessàries per realitzar la connexió a la BD.



b. Fem clic a `Next` i ens demanarà la ubicació on crear aquest arxiu, i indicarem que serà la carpeta **default (src)**, `Next` novament i passarem a indicar les dades per poder realitzar la connexió a la BD.



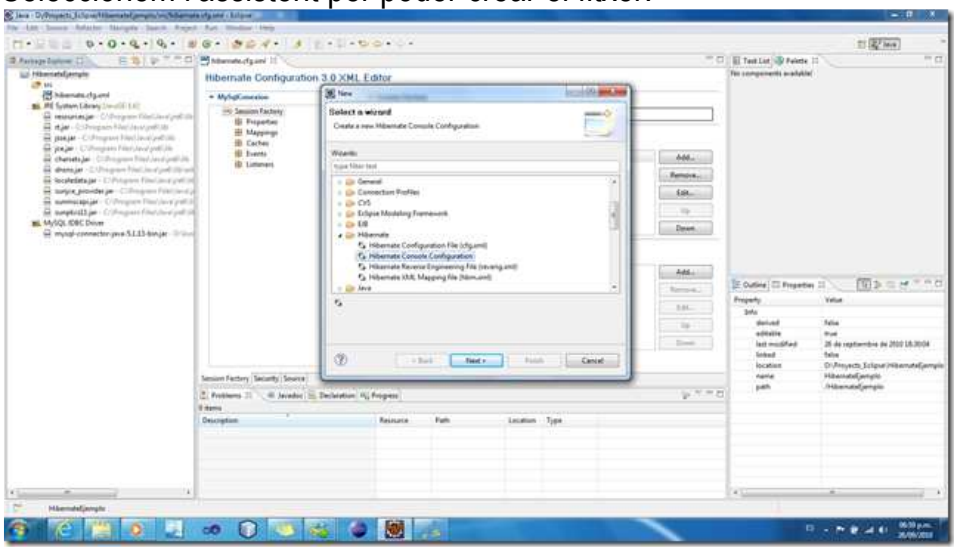
c. **Omplirem les dades** necessàries per poder establir la connexió a la BD:
   i. **Session Factory Name**: Nom de la nostra connexió a la BD.
   ii. **Database Dialect**: Com es comunicarà JDBC amb la base de dades.
   iii. **Database Class**[1]: La classe del JDBC que s'utilitzarà per la connexió.
   iv. **Connection URL:** Ruta d'accés a la BD.
   v. **Username:** Usuari per connectar-se a MySQL.
   vi. **Password:** Contrasenya per connectar-se a MySQL.



[1] Si quan a l'obrir la finestra no apareix en aquest apartat com.mysql.jdbc.Driver significa que el driver JDBC no està ben instal·lat.

5. Creació del fitxer Hibernate Console Configuration:

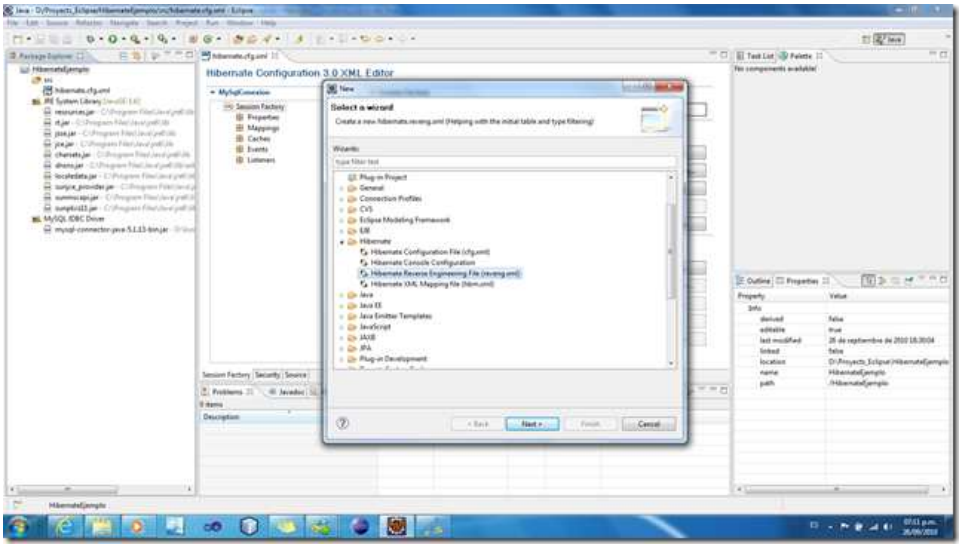
a. Seleccionem l'assistent per poder crear el fitxer.



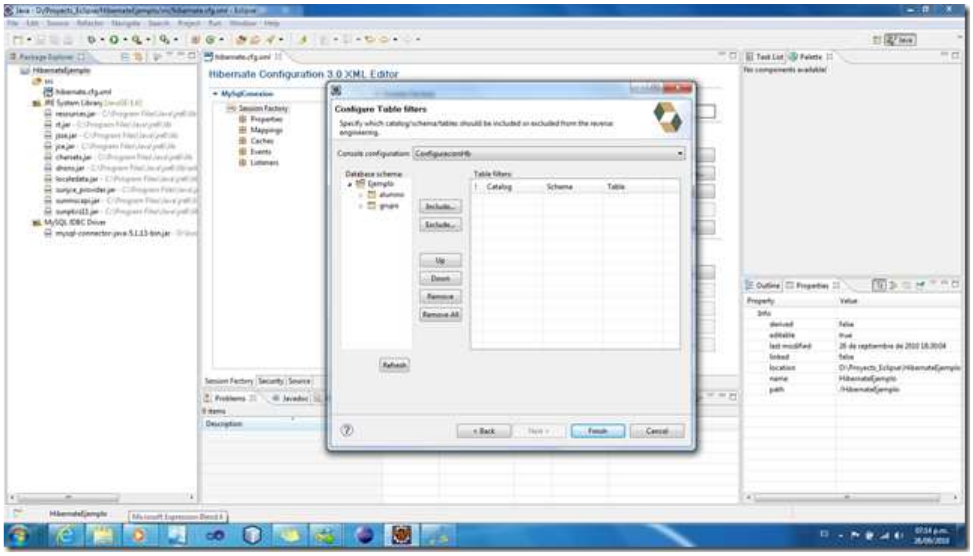
b. Fem clic a `Next`, a la nova finestra, a l'apartat **Configuration File** ja ve seleccionat el nostre fitxer de configuració si ha estat creat amb anterioritat. En cas de no haver-ne, li posem un nom a la nostra configuració, i premem `Finish`.

6. Crear l'Hibernate Reverse Engineering (reveng.xml). Aquest arxiu s'encarrega de crear classes a partir de les taules de MySQL.

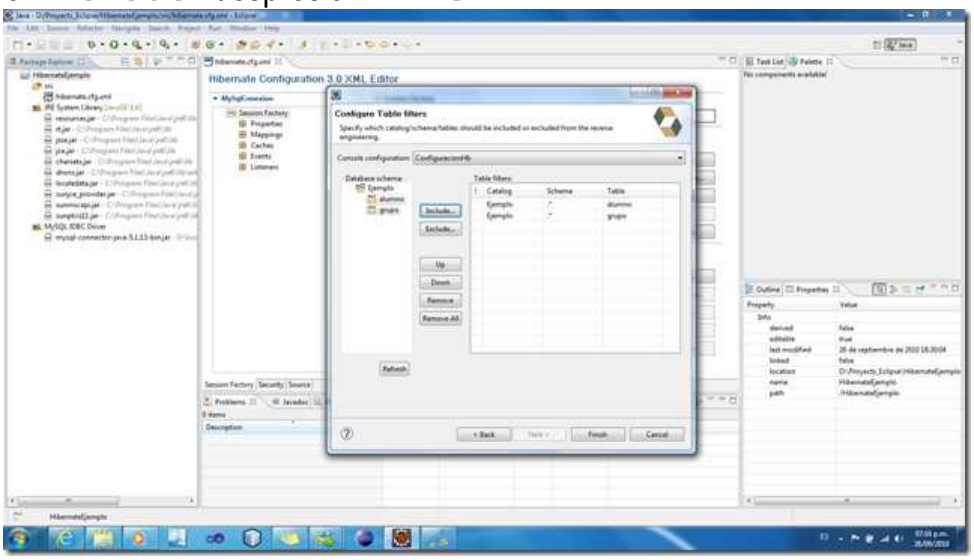
a. Seleccionem la opció que ens permetrà crear **reveng.xml** i fem clic a `Next`.



b. Indicarem on es guardarà el fitxer, que ha de ser al mateix paquet on hem desat el fitxer **cfg.xml**, i apareixerà una finestra on indicarem les taules que volem mapejar cliquem `Refresh`, i si tot ha sortit bé mostrarà la base de dades i les seves taules.



c. Les taules podem ser afegides una per una o totes de cop i farem clic a `include` i després a `finish`.

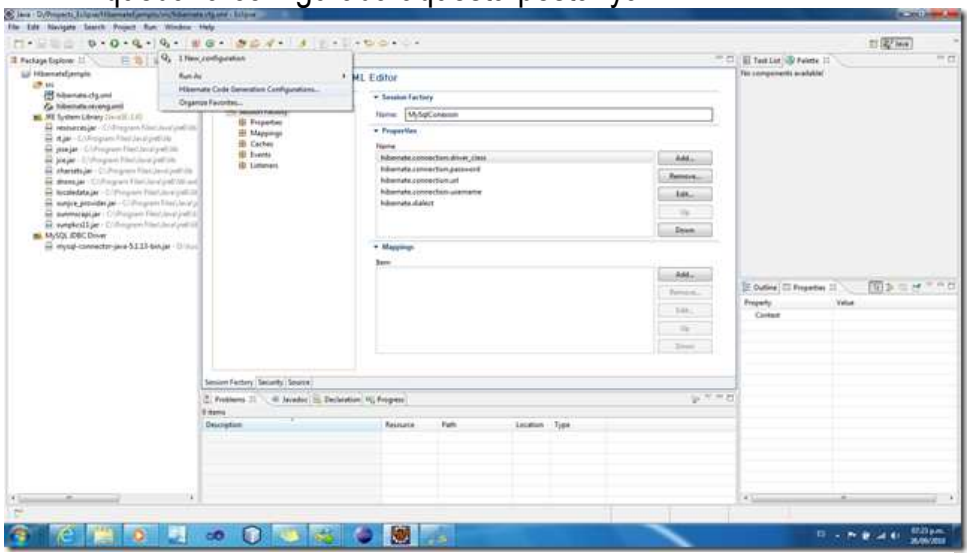


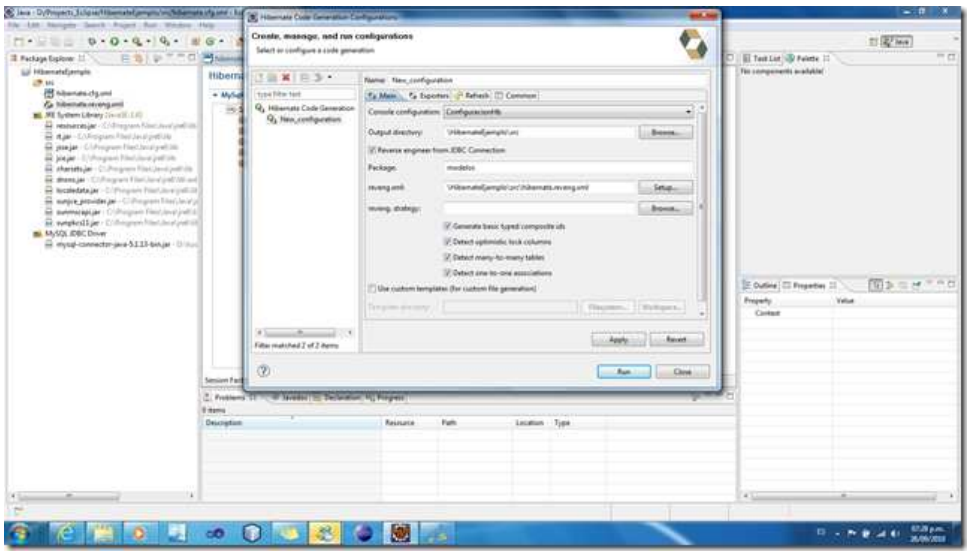
d. El codi font generat del fitxer reveng.xml és el següent:

```
<?xml version="1.0" encoding="UTF-8"?>
<hibernate-reverse-engineering>
<table-filter match-catalog-name="Ejemplo" match-name="alumno"/>
<table-filter match-catalog-name="Ejemplo" match-name="grupo"/>
</hibernate-reverse-engineering>
```

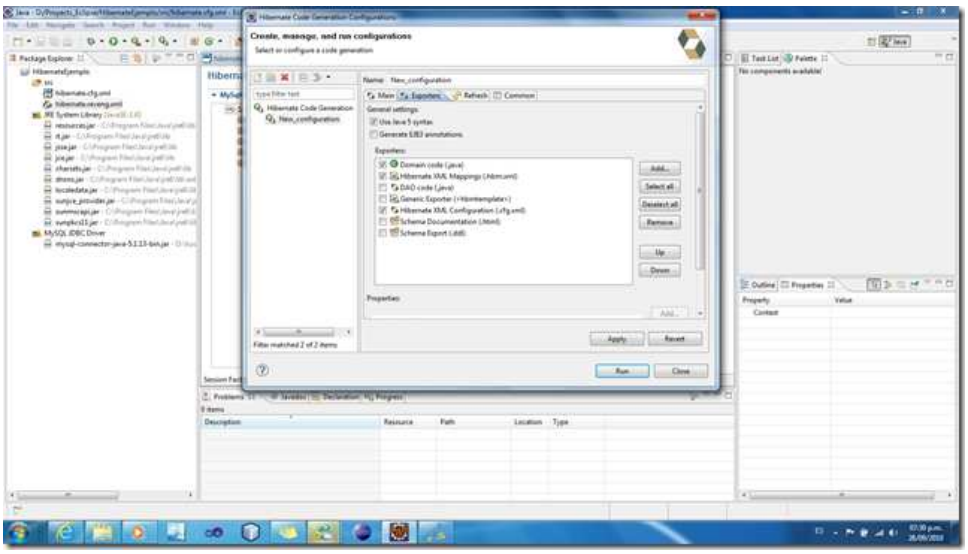
7. Generació de les classes.

a. Seleccionem `Run>Hibernate Code Generation Configurations`. Apareixerà una finestra de configuració on seleccionarem la pestanya Main, i especificarem els següents valors:
   i. **Console Configuration**: ConfiguracioHB.
   ii. **Output directory:** ha de ser la mateixa carpeta de codis.
   iii. **Package:** models.
   iv. **Reveng.xml:** ja ha estat creat amb autoritat, i mostra com quedaria configurada aquesta pestanya.

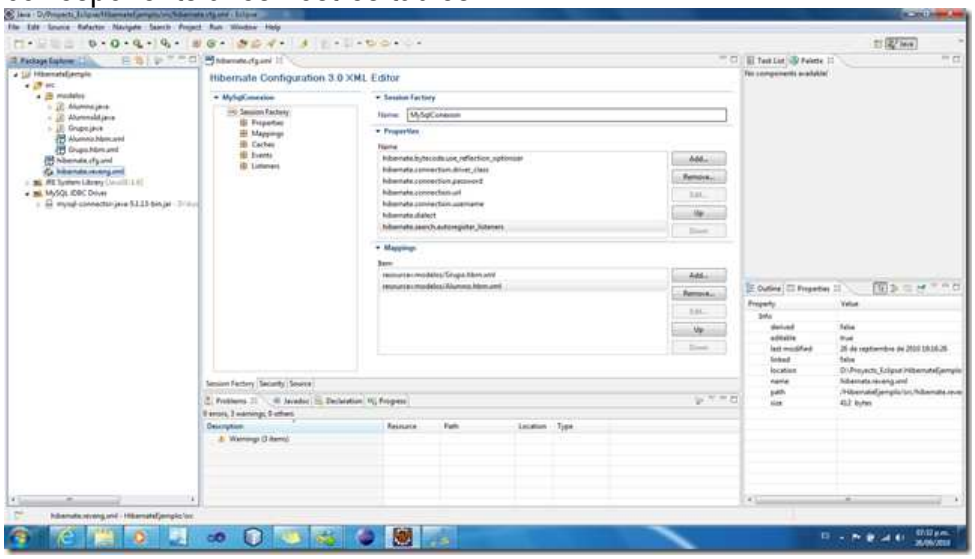




b. Anem a la pestanya `Exporters` i seleccionem els arxius que volem generar.

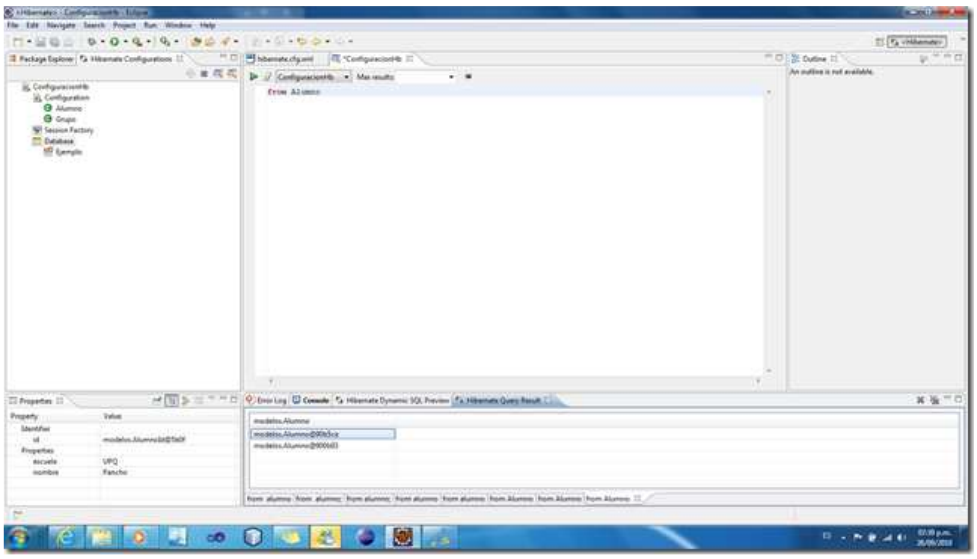


c. Un cop seleccionat, fem clic a `Apply`, després a `Run`... i si tot surt bé es generarà un paquet anomenat **models** amb les classes corresponents a les nostres taules.



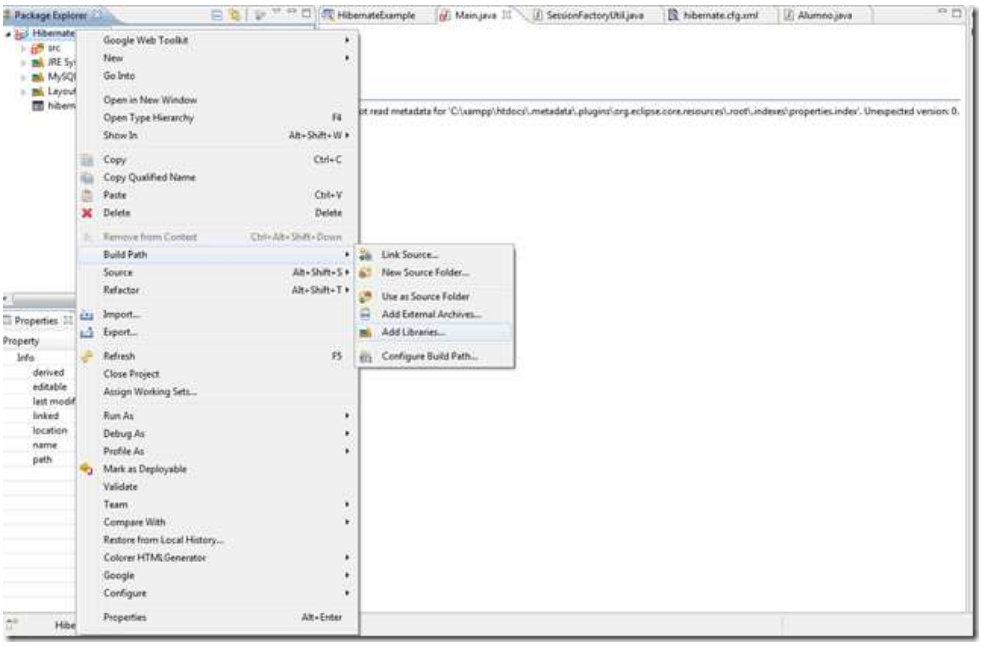
d. Per comprovar que s'ha realitzat tot correctament, anem a la perspectiva d'Hibernate, seleccionem la nostra configuració a la pestanya `Hibernate Configurations`, i fem clic al botó dret a `Database>HQL Editor` i executem una sentència HQL de prova, de la que podrem comprovar el resultat al panell `Hibernate Query Result`.

```
from Alumno
```



8. **Iniciar el projecte**. Amb totes aquestes tasques realitzades, encara no es pot començar a programar.

    a. Afegeix, tal i com es veu al vídeo-tutorial de Java amb SQL, al projecte el connector JDBC amb el SGBD.
    b. Afegeix com llibreries externes les APIs contingudes a \\marte\recursos\ams2\m06\required. Si tot ha estat correcte ja es pot treballar amb Hibernate.



9. **Treballar amb Hibernate**. Hibernate treballa amb sessions, creant una instància a la base de dades per tal de poder treballar amb ella. Una errada força comuna és iniciar una sessió per cada classe, la qual cosa pot generar errades i un consum excessiu de memòria. Es recomana crear una única instància a utilitzar per tota la nostra aplicació, i això s'aconsegueix creant un **singleton**[2].

    a. Creem una classe anomenada **HibernateUtil.java** en el paquet **models** del nostre projecte. Aquesta classe permetrà accedir a la sessió des de qualsevol Amb a i escrivim el següent codi:

```
package models;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.AnnotationConfiguration;

public class HibernateUtil{

private static final SessionFactory sf;

static{
    try{
        sf=new annotationConfiguration().
           configure(new File("hibernate.cfg.xml")).buildSessionFactory();
    }catch(Throwable ex){
        System.err.println("SessionFactory no creada "+ex);
        throw new ExceptionInInitializerError(ex);
    }
}

public static SessionFactory getSessionFactory(){return sf;}

}
```

    b. Crearen una classe Main.java que serà la que permetrà fer córrer el projecte amb Hibernate.

```
package models;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.AnnotationConfiguration;

public class Main{

public static void main (String[] args){
    SessionFactory session=HibernateUtil.getSessionFactory();
}

}
```

[2] Un singleton està dissenyat per garantir que una clase només tindrà una instancia i s'ofereix un accés global a aquesta.