



PRÁCTICA 3. DESFASE OBJETO-RELACIONAL

2º DAM

Jesús Del Pino Martínez, José Abel Viñolo Galdeano

Contenido

<u>Contenido.....</u>	<u>2</u>
<u>Creación de la BD.....</u>	<u>3</u>
<u>Mapeo Objeto Relacional.....</u>	<u>4</u>
<u>Conexión con la BD.....</u>	<u>4</u>
<u>Realización del mapeo objeto relacional.....</u>	<u>8</u>
<u>Consultas.....</u>	<u>26</u>
<u>Consulta 1.....</u>	<u>26</u>
<u>Consulta 2.....</u>	<u>26</u>
<u>Consulta 3.....</u>	<u>27</u>
<u>Consulta 4.....</u>	<u>28</u>
<u>Sistema gestor Oracle.....</u>	<u>29</u>
<u>Instalación del sistema gestor y creación de la BD multas.....</u>	<u>29</u>
<u>Mapeo utilizando Hibernate.....</u>	<u>35</u>
<u>Consultas.....</u>	<u>49</u>
<u>Consulta 1.....</u>	<u>49</u>
<u>Consulta 2.....</u>	<u>49</u>
<u>Anexo.....</u>	<u>52</u>
<u>MySQL.....</u>	<u>52</u>
<u>ORACLE.....</u>	<u>55</u>

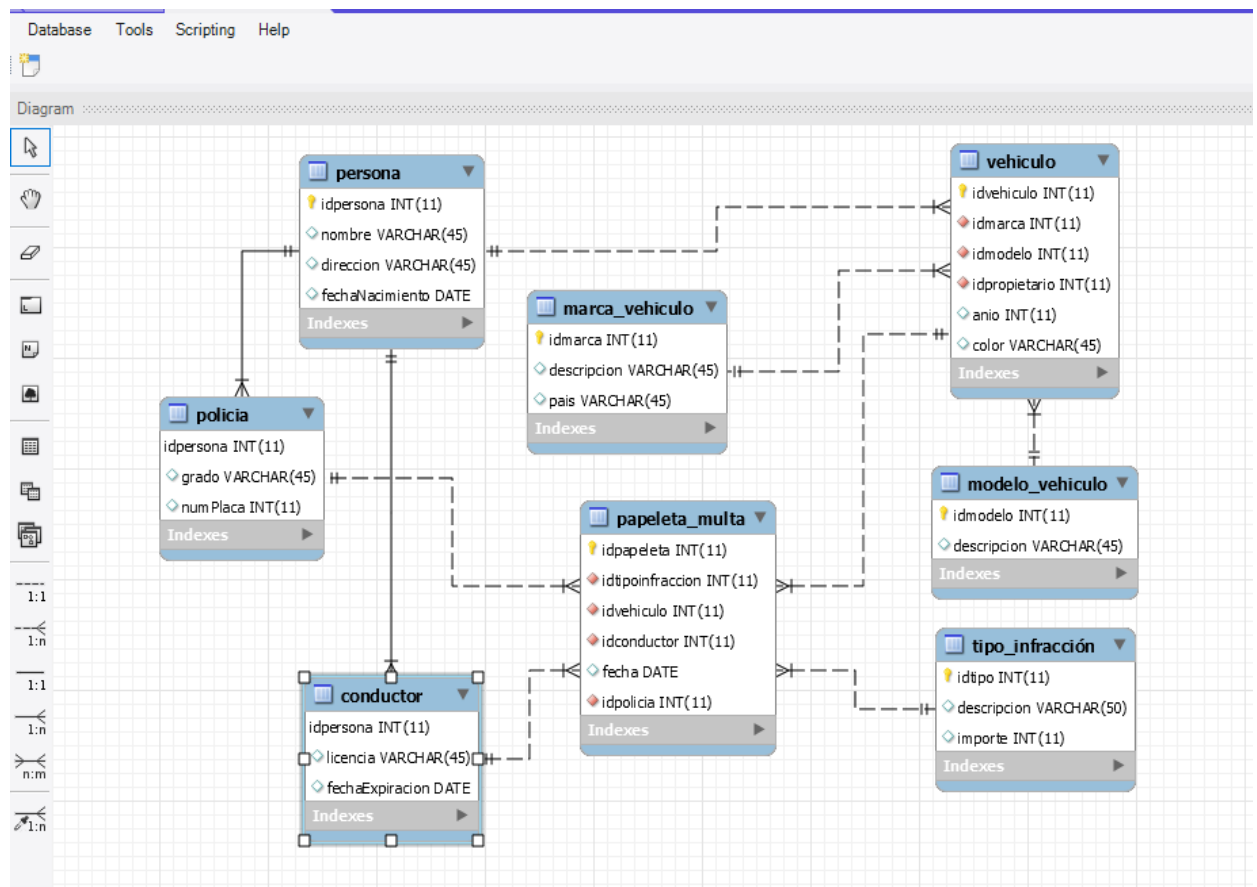
En esta práctica crearemos una base de datos para la gestión las infracciones de tráfico cometidas por los conductores en la provincia de Almería.

Creación de la BD

La creación de la base de datos la realizaremos en el entorno gráfico MySQL Workbench.

Creamos la BD de nombre bdMulta y creamos las tablas de la BD. Dado que en MySQL no es posible utilizar las secuencias así que utilizaremos AUTO_INCREMENT.

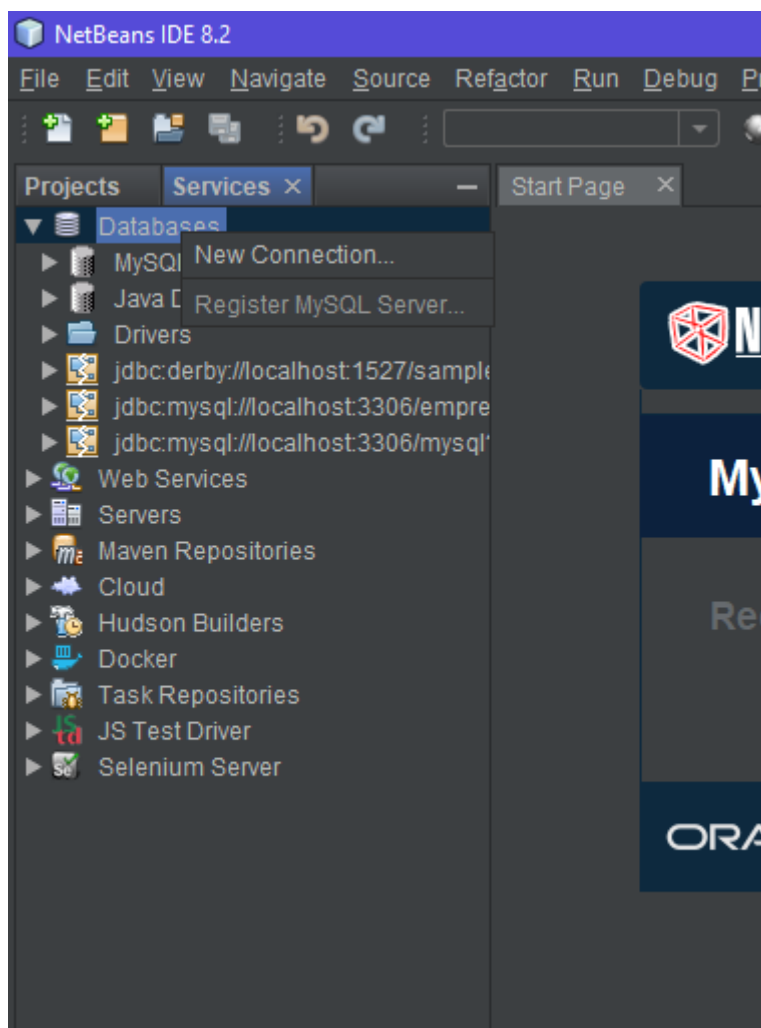
La BD resultante es la siguiente:



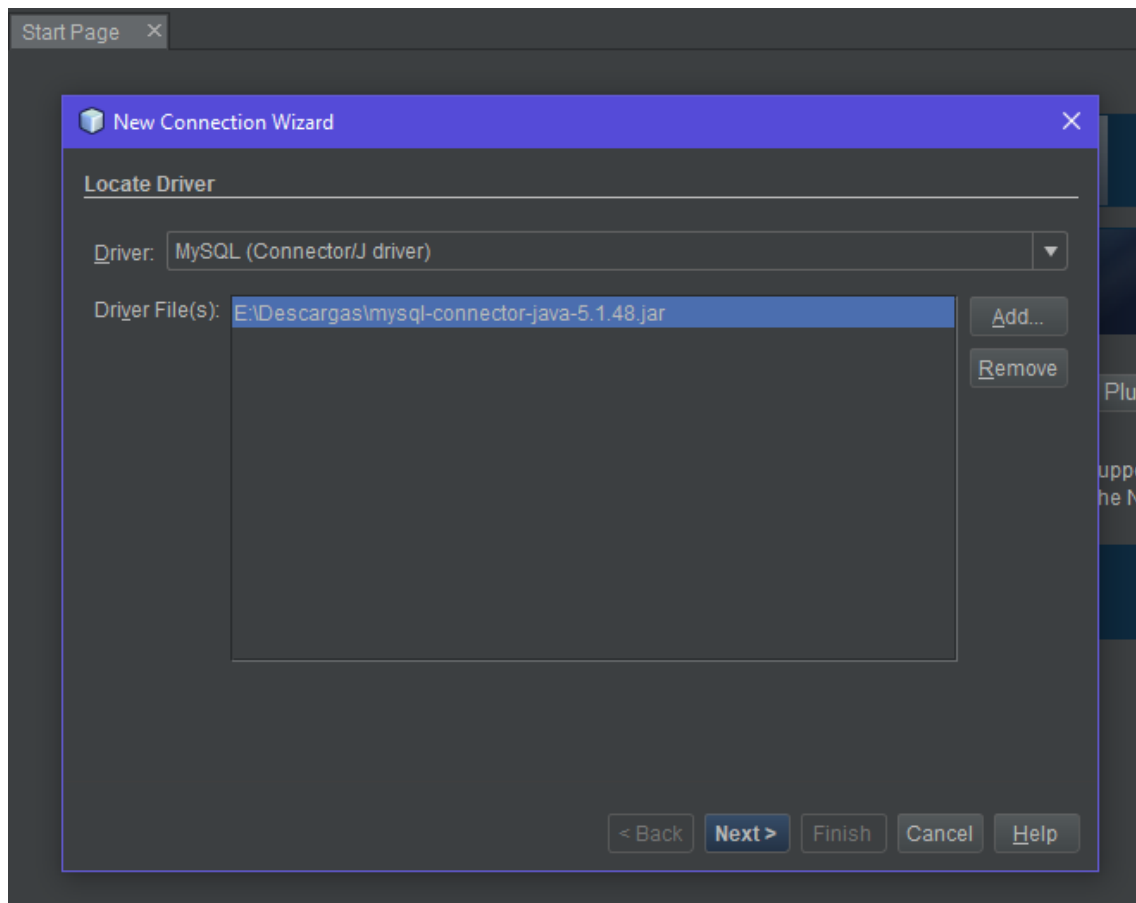
Mapeo Objeto Relacional

Conexión con la BD

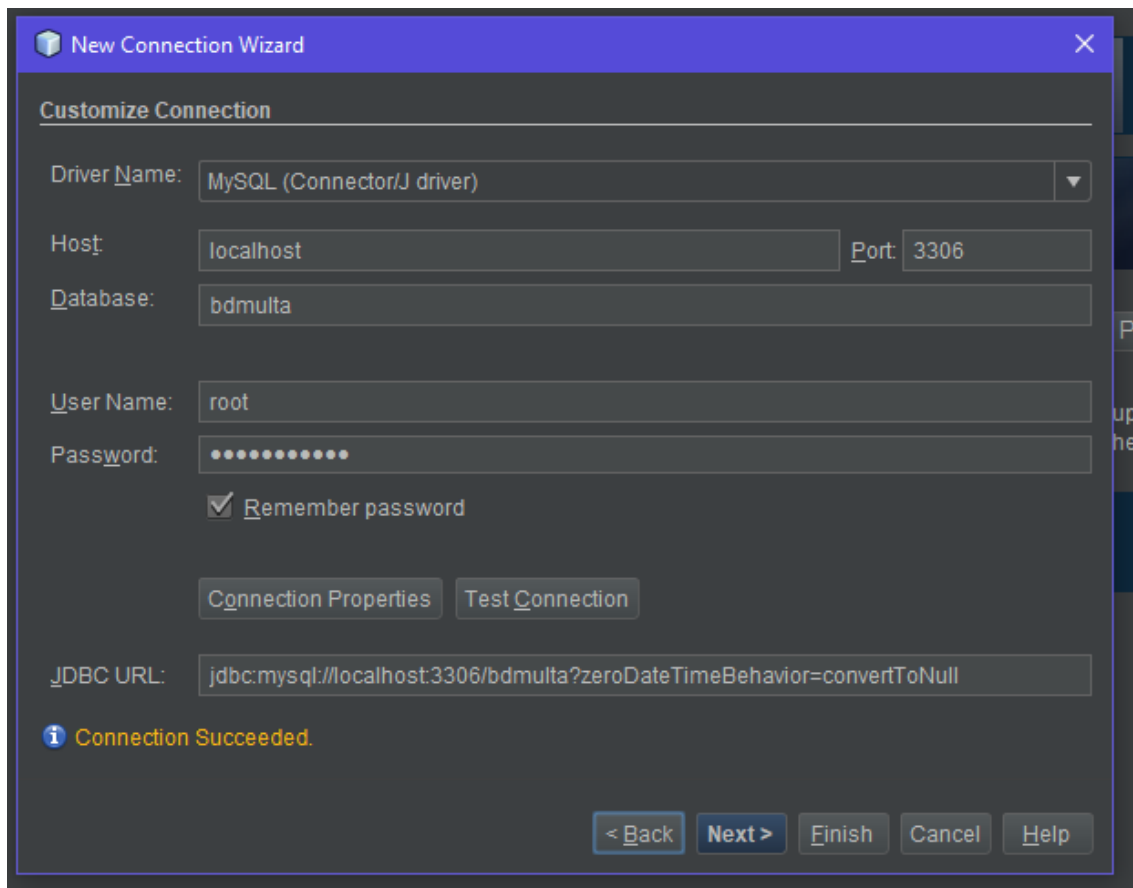
Lo primero que tendremos que hacer para llevar a cabo el mapeo será conectar con la base de datos desde NetBeans, para ello, nos vamos a la pestaña de Servicios y creamos una nueva conexión.



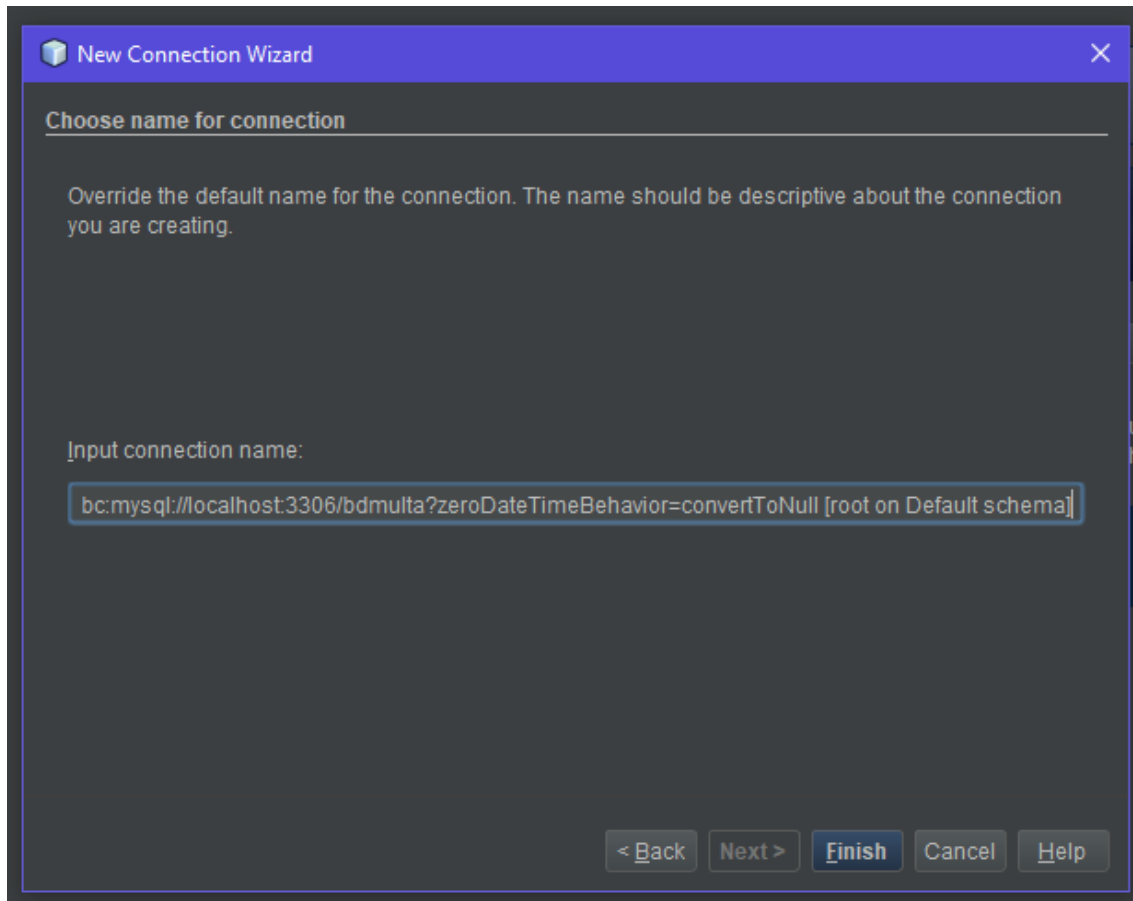
Localizamos el driver para realizar la conexión:



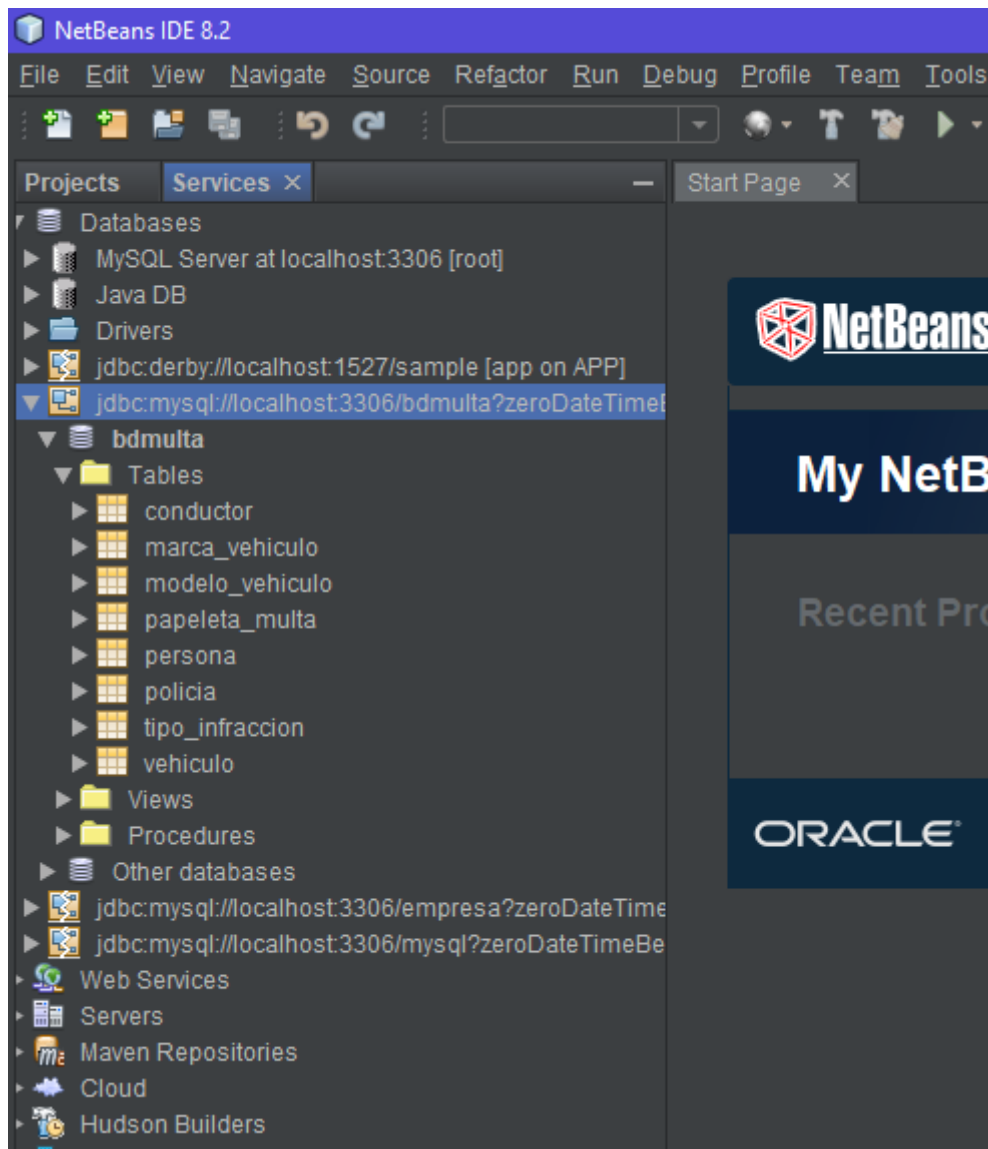
A continuación, introducimos el nombre de nuestra base de datos, el usuario con el que accederemos y su contraseña. Hacemos click en el botón de “Probar conexión” para ver si se conecta correctamente:



Por último, haremos click en Finish y ya tendremos nuestra conexión a la BD preparada.

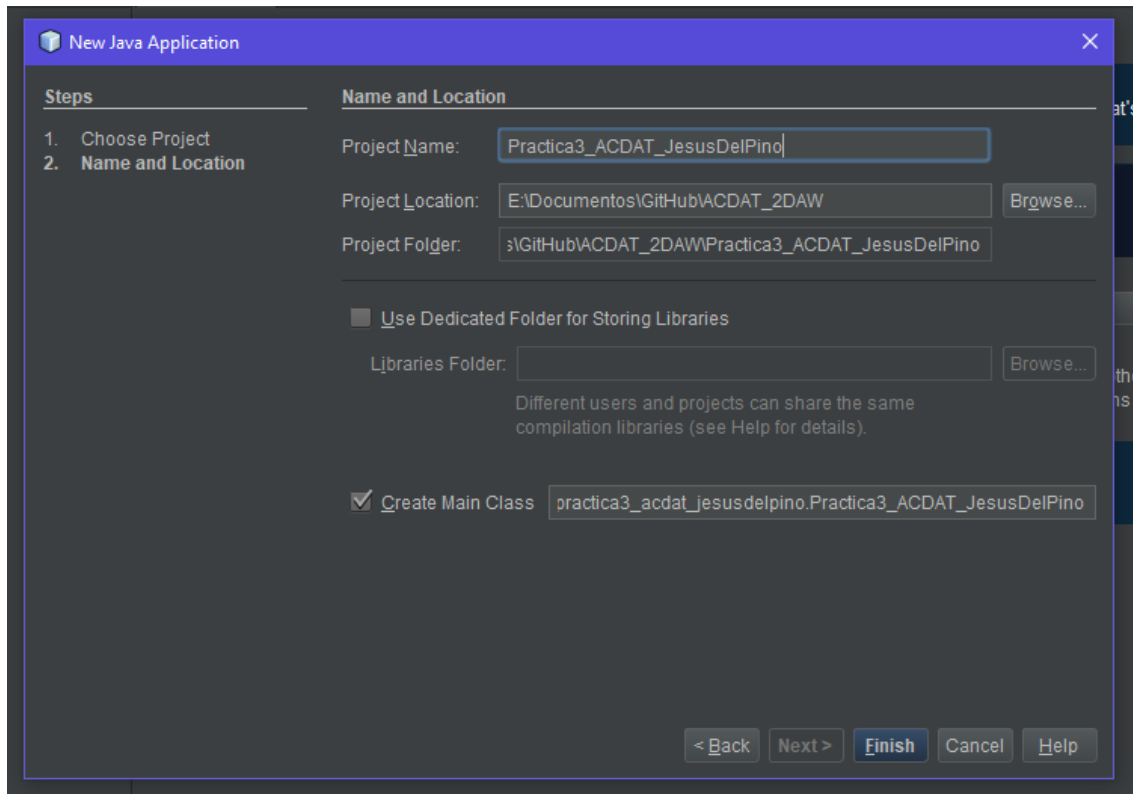


Como podemos comprobar se muestran todas las tablas de nuestra BD:

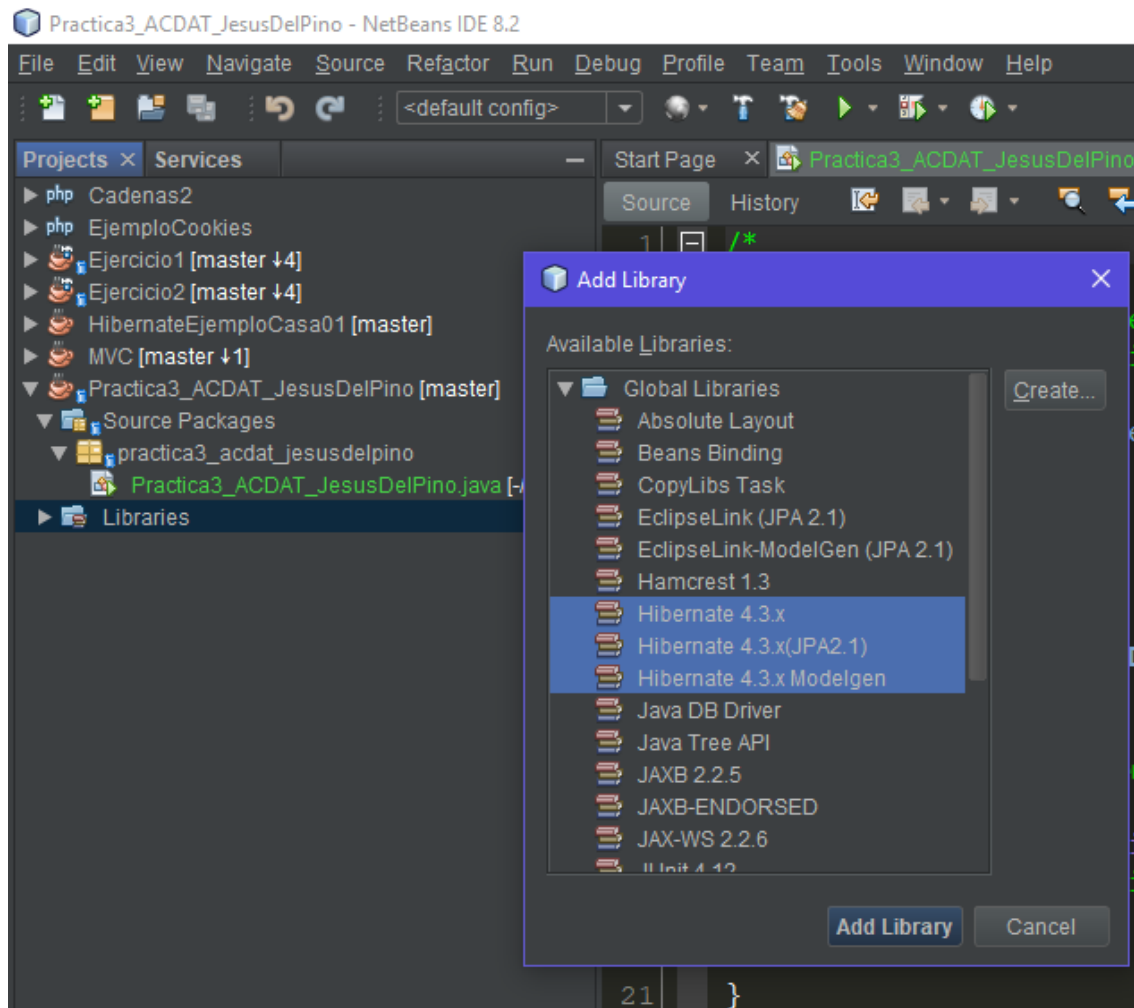


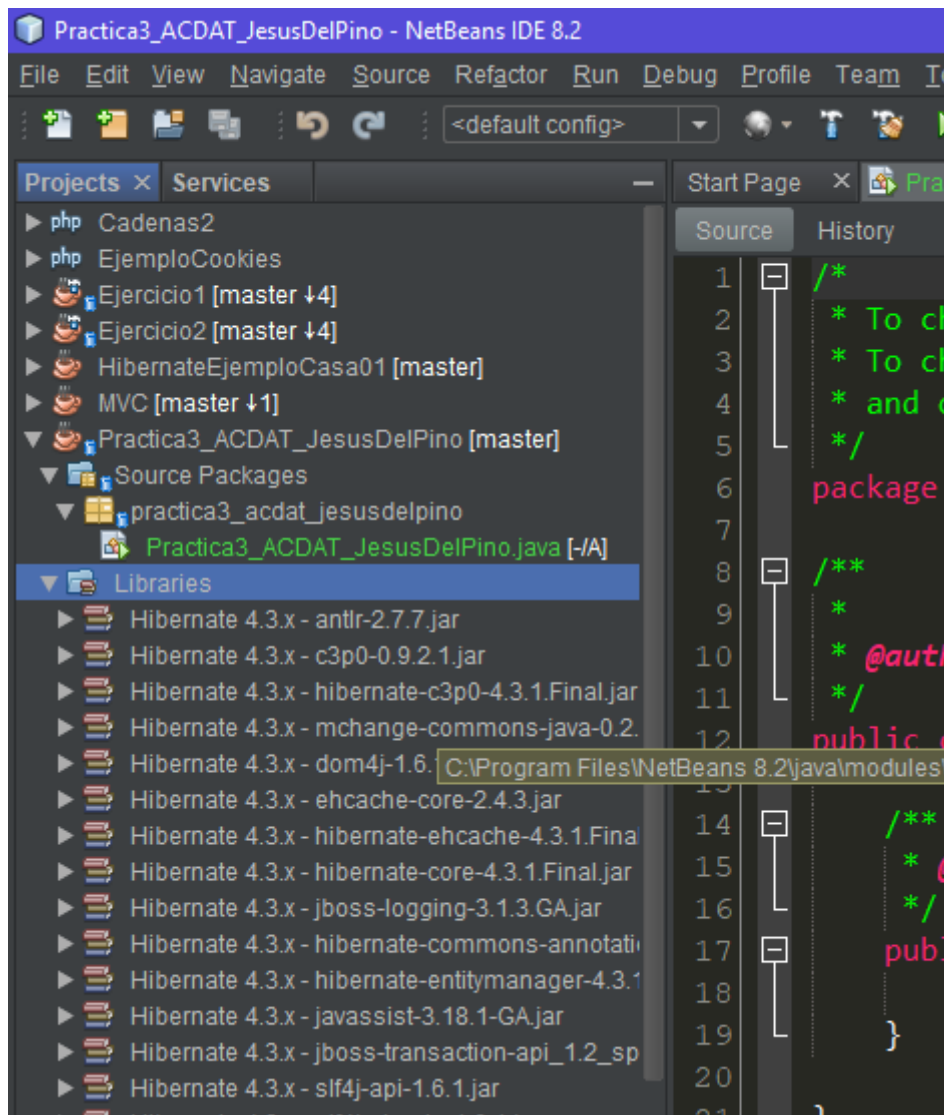
Realización del mapeo objeto relacional

Creamos un nuevo proyecto, en mi caso lo llamaré Practica3_ACDAT_JesusDelPino.



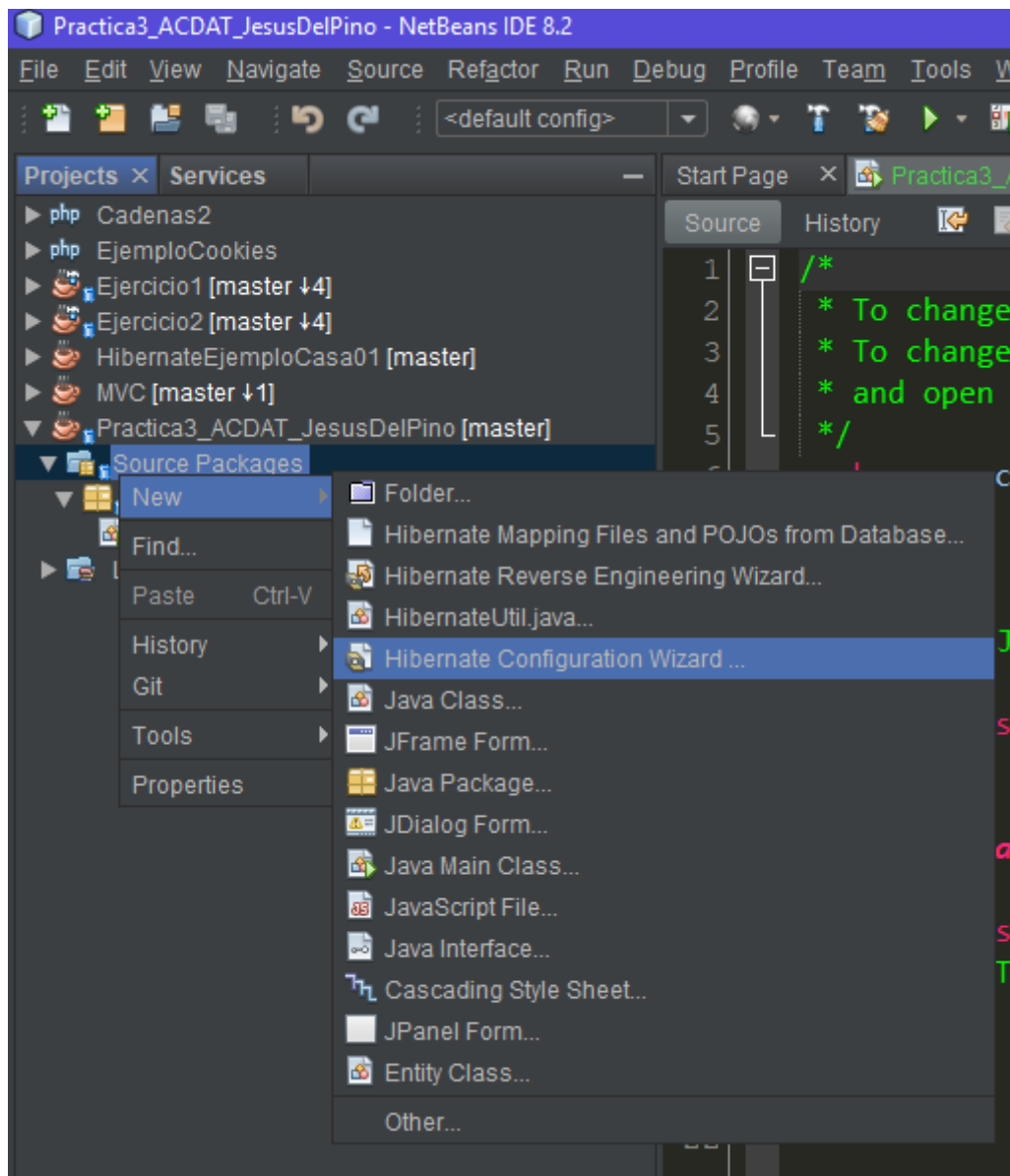
Para poder utilizar Hibernate tendremos que importar sus librerías, para ello, haremos click derecho sobre la carpeta Libraries de nuestro proyecto y Add Libraries:



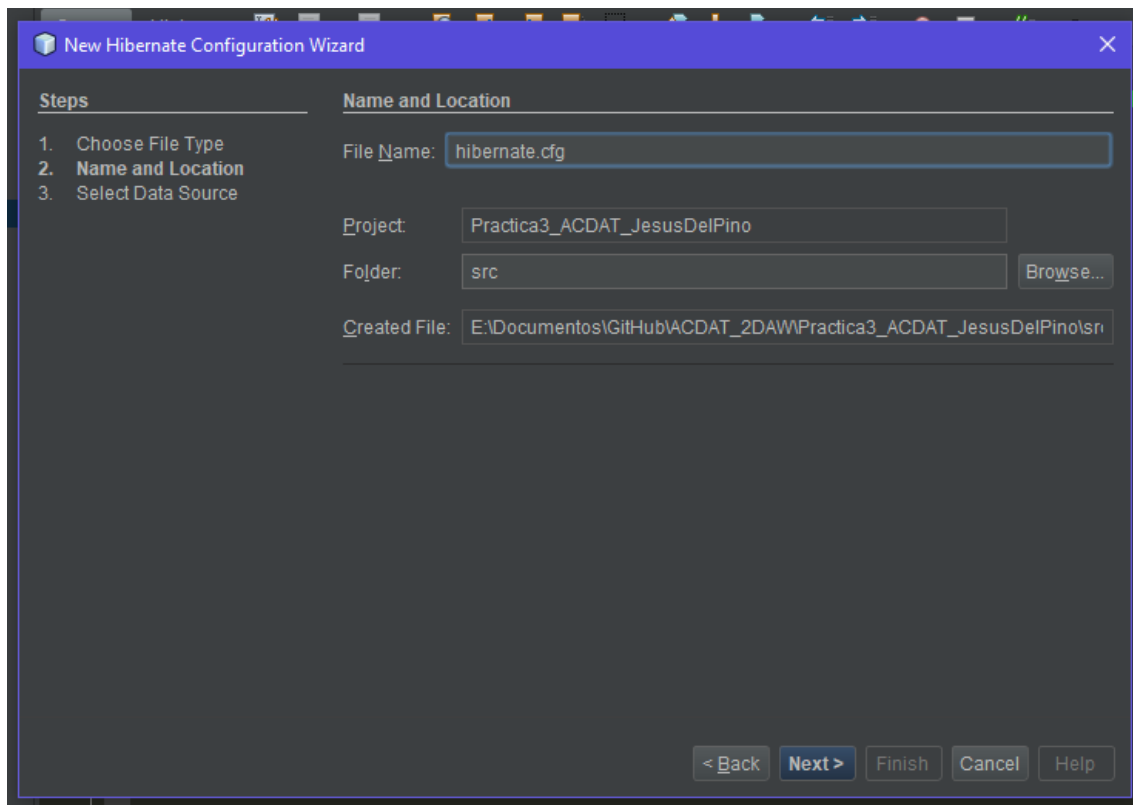


Lo siguiente, será crear el archivo de configuración de Hibernate (hibernate.cfg.xml), que contiene información sobre la conexión de la base de datos, los recursos de mapeo y otras propiedades de la conexión.

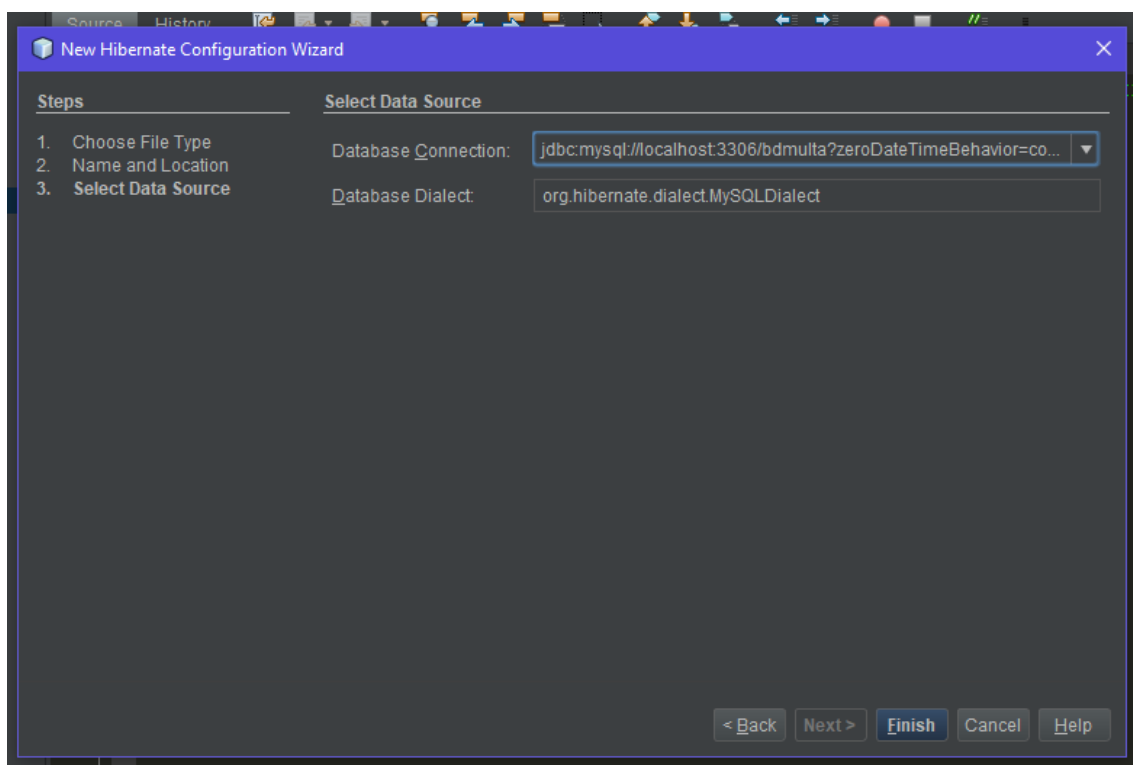
Para ello, hacemos click derecho sobre el nodo Source packages, nuevo y elegimos el asistente de configuración:



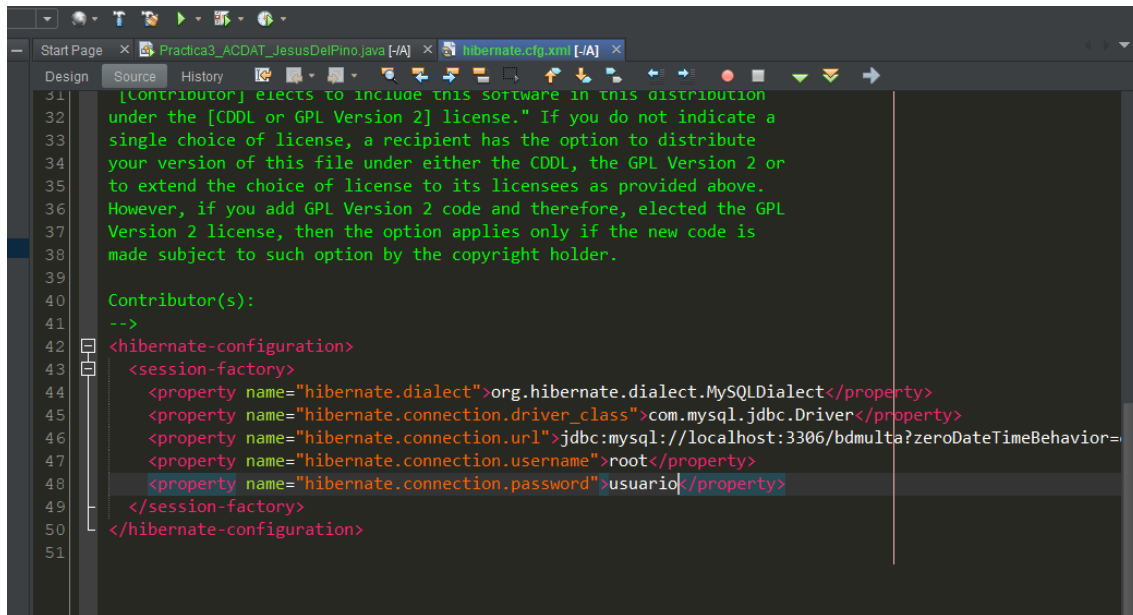
Dejamos el nombre por defecto:



Escogemos la conexión a la BD que creamos anteriormente:



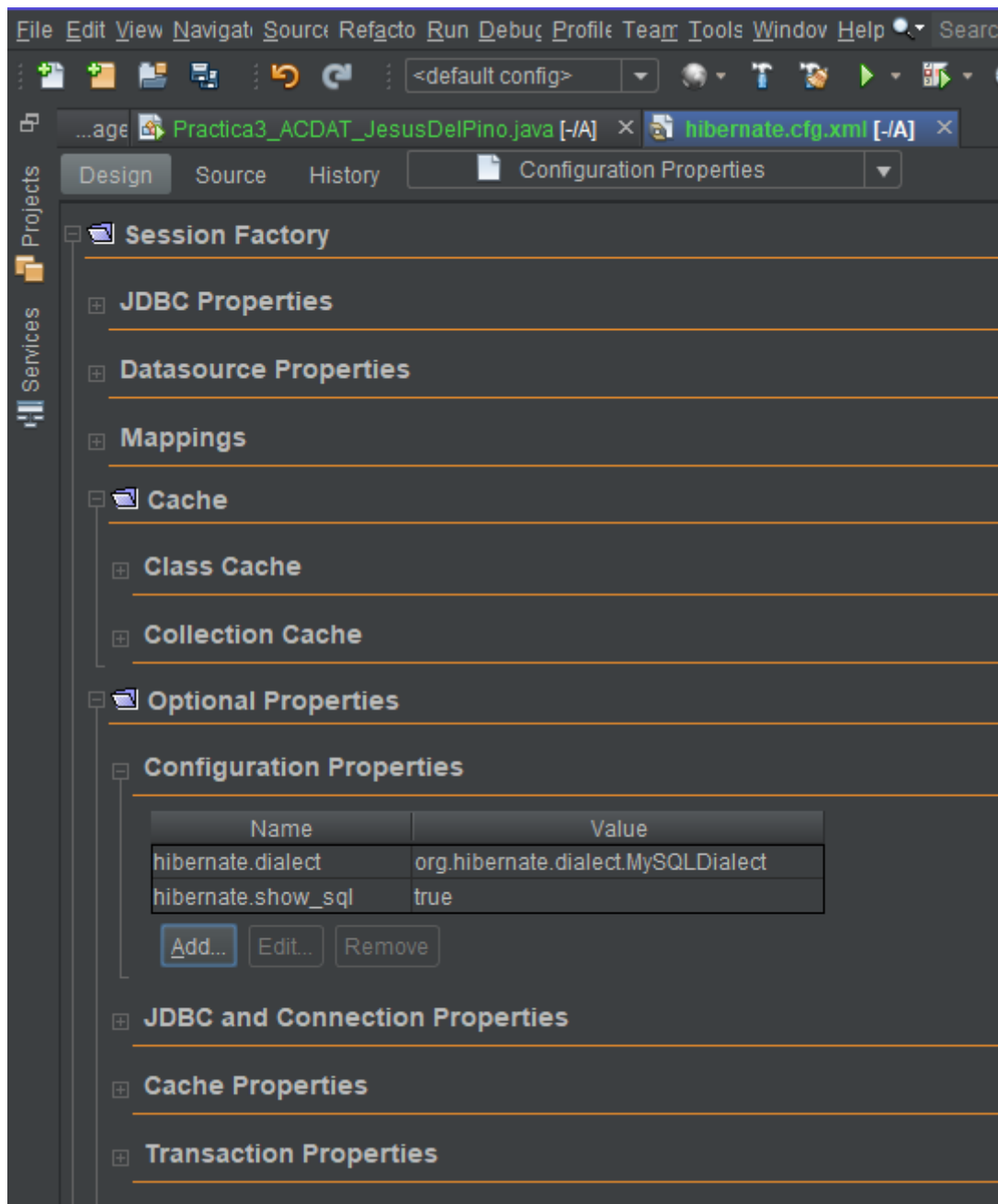
Ya tenemos creado nuestro archivo de configuración:



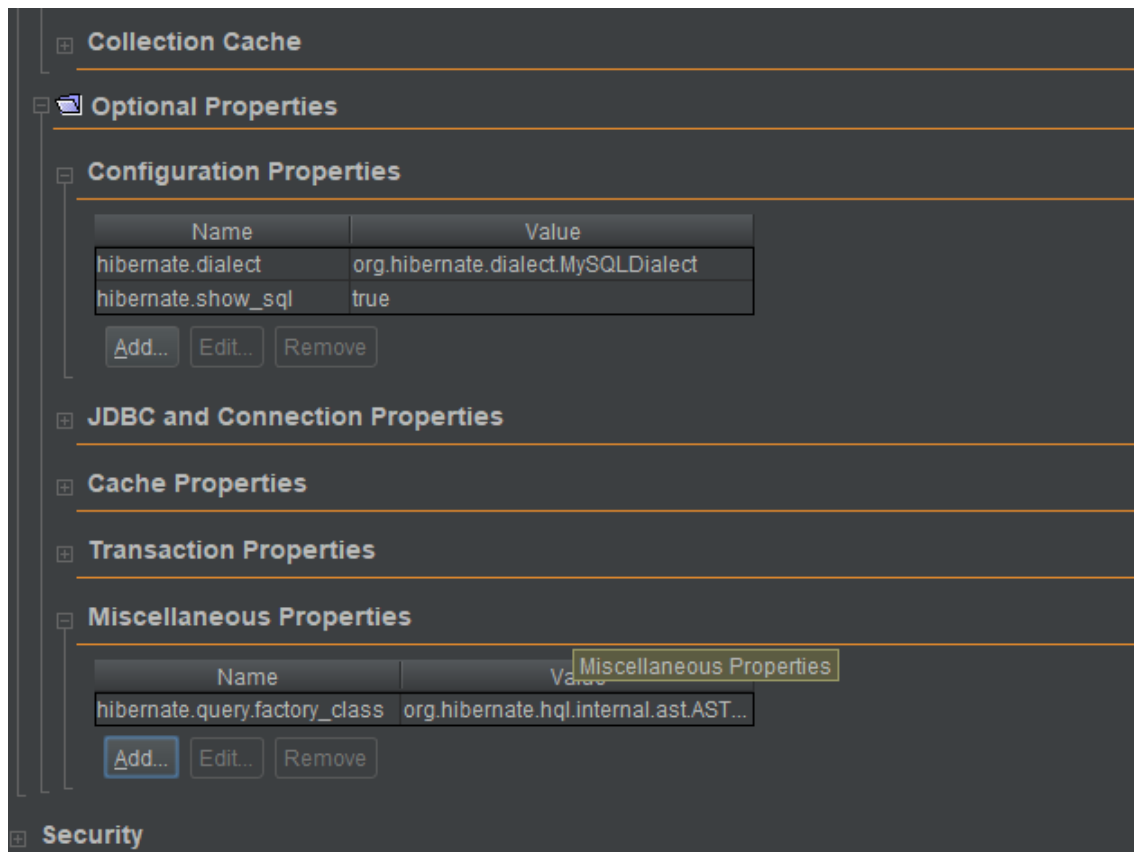
```
31 [Contributor] elects to include this software in this distribution
32 under the [CDDL or GPL Version 2] license." If you do not indicate a
33 single choice of license, a recipient has the option to distribute
34 your version of this file under either the CDDL, the GPL Version 2 or
35 to extend the choice of license to its licensees as provided above.
36 However, if you add GPL Version 2 code and therefore, elected the GPL
37 Version 2 license, then the option applies only if the new code is
38 made subject to such option by the copyright holder.
39
40 Contributor(s):
41 -->
42 <hibernate-configuration>
43   <session-factory>
44     <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
45     <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
46     <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/bdmulta?zeroDateTimeBehavior=
47     <property name="hibernate.connection.username">root</property>
48     <property name="hibernate.connection.password">usuariok</property>
49   </session-factory>
50 </hibernate-configuration>
51
```

Ahora tendremos que modificar el fichero de configuración hibernate.cfg.xml para habilitar la depuración de consultas SQL.

En la vista de diseño abrimos la sección Optional properties/Configuration properties y añadimos lo siguiente:

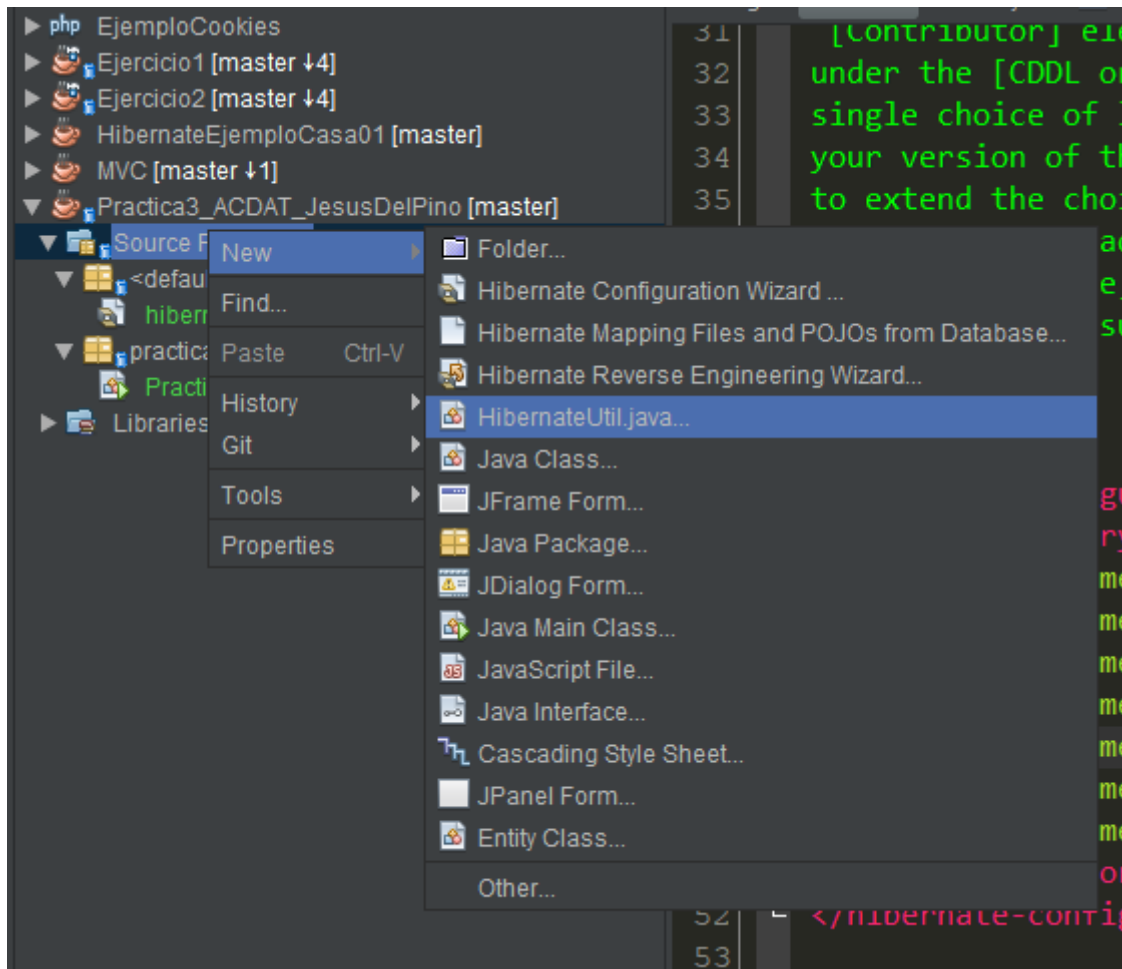


A continuación, seleccionamos el nodo de “Miscellaneous Properties” y agregamos hibernate.query.factory_class

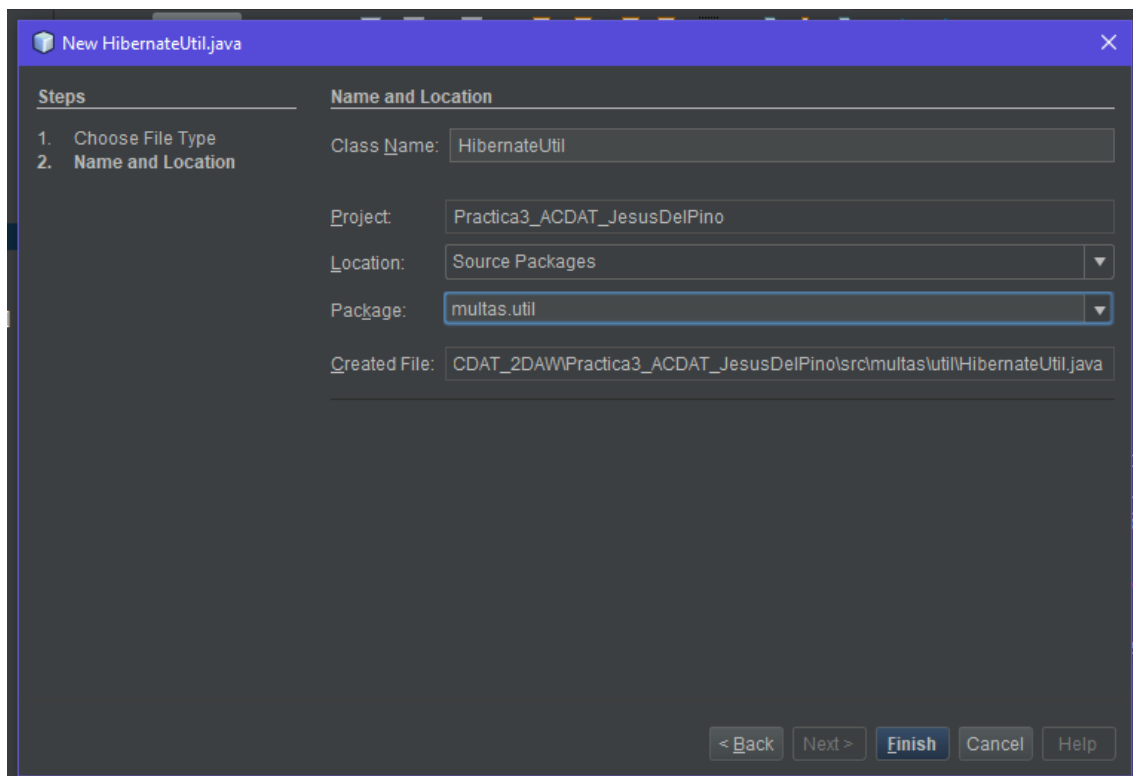


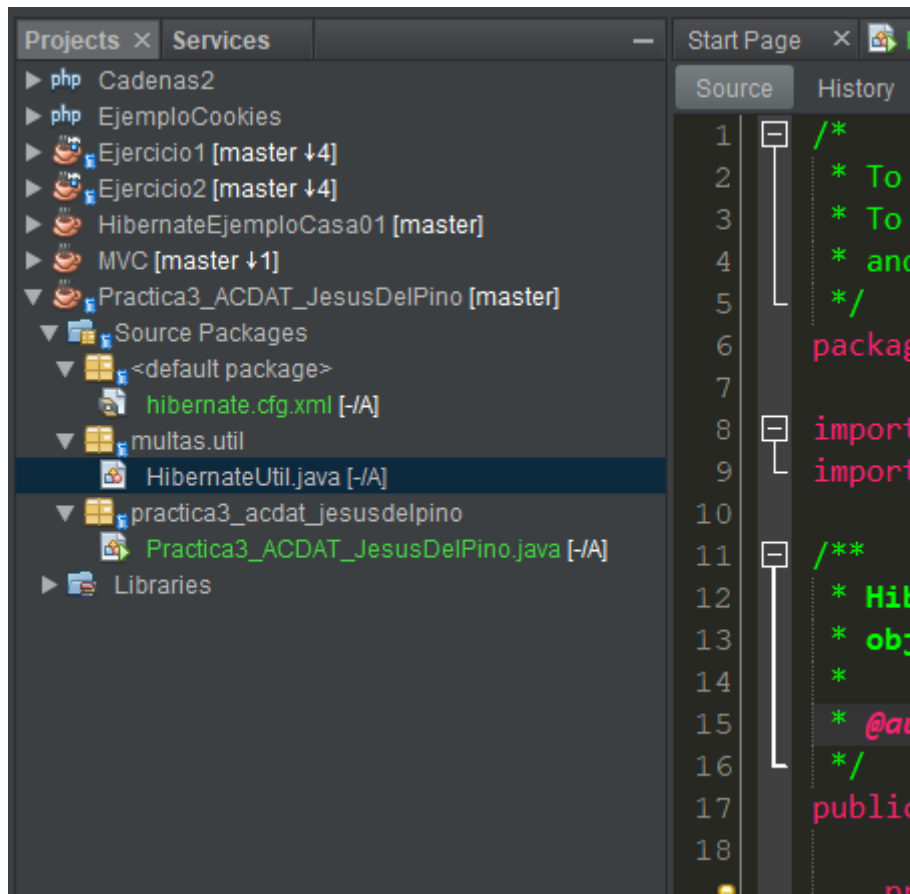
Lo siguiente será crear la clase `HibernateUtil.java` HelperFile, esta clase llama al método de Hibernate **`configure()`**, que selecciona el fichero **`hibernate.cfg.xml`** y a partir de él construye *SessionFactory* para obtener el objeto *Session*.

En el nodo “Source packages” hacemos click derecho, nuevo y seleccionamos `HibernateUtil.java`:



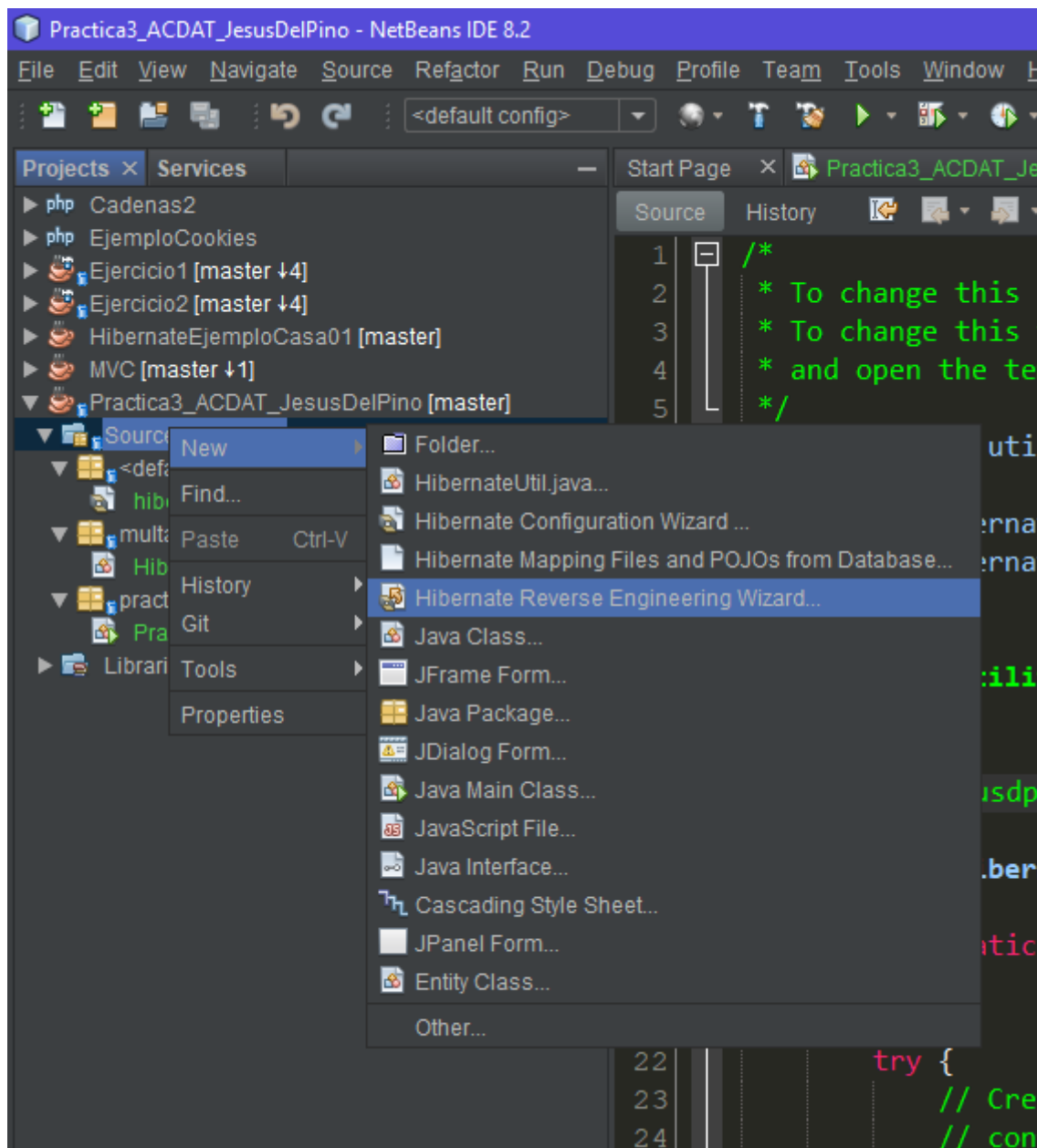
De nombre le pondremos HibrenateUtil y el paquete será multas.util:



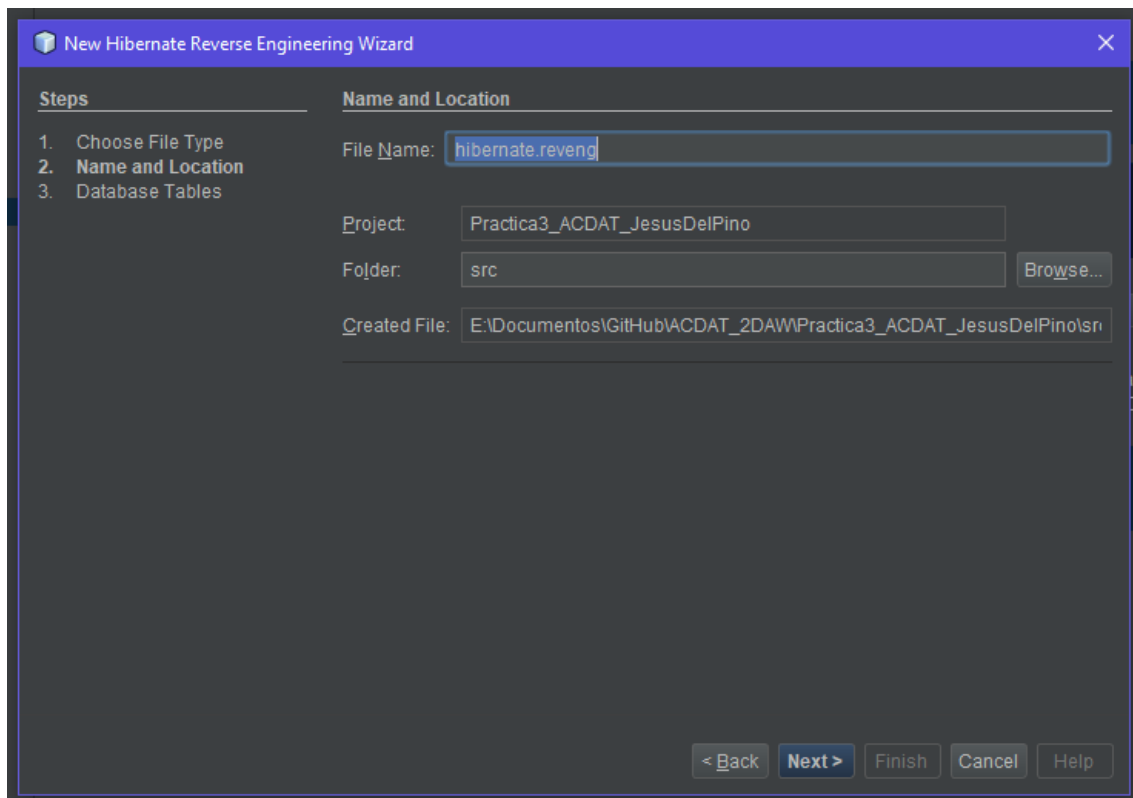


Ahora, utilizaremos el asistente de Ingeniería Inversa y el Mapeo de Archivos de Hibernate y POJO desde una base de datos y crearemos múltiples POJO's y archivos de mapeo, basados en las tablas de la base de datos que estemos seleccionando.

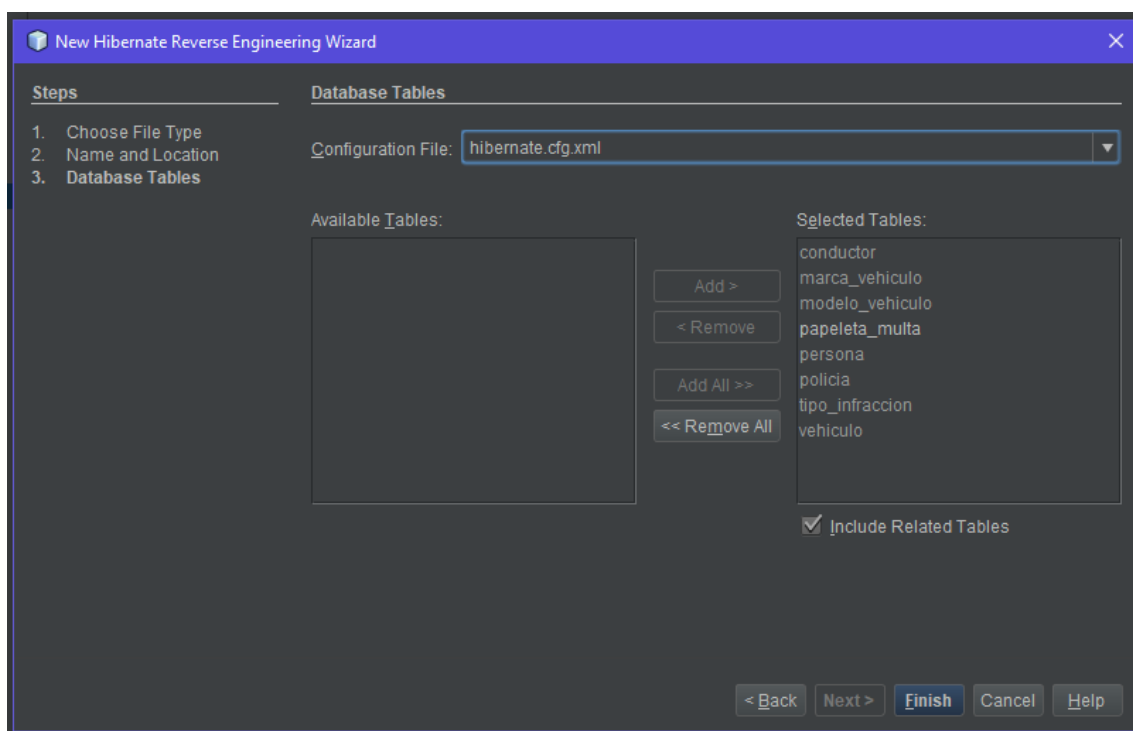
Para ello, haremos click derecho sobre el nodo "Source packages", nuevo y seleccionamos el asistente de ingeniería inversa de Hibernate.

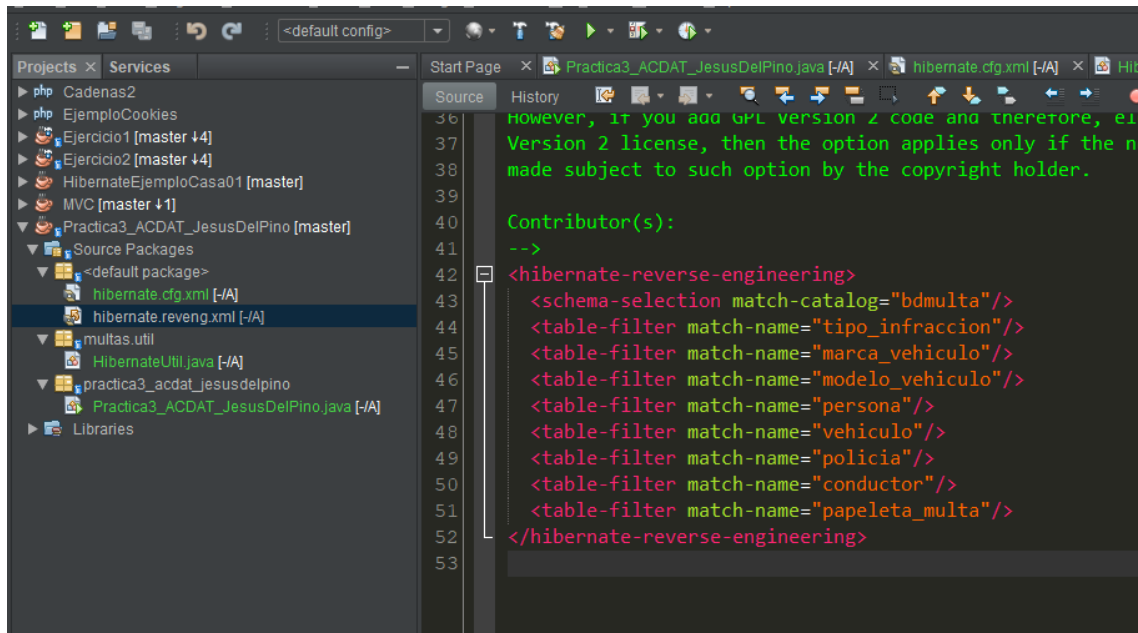


Dejamos el nombre por defecto:



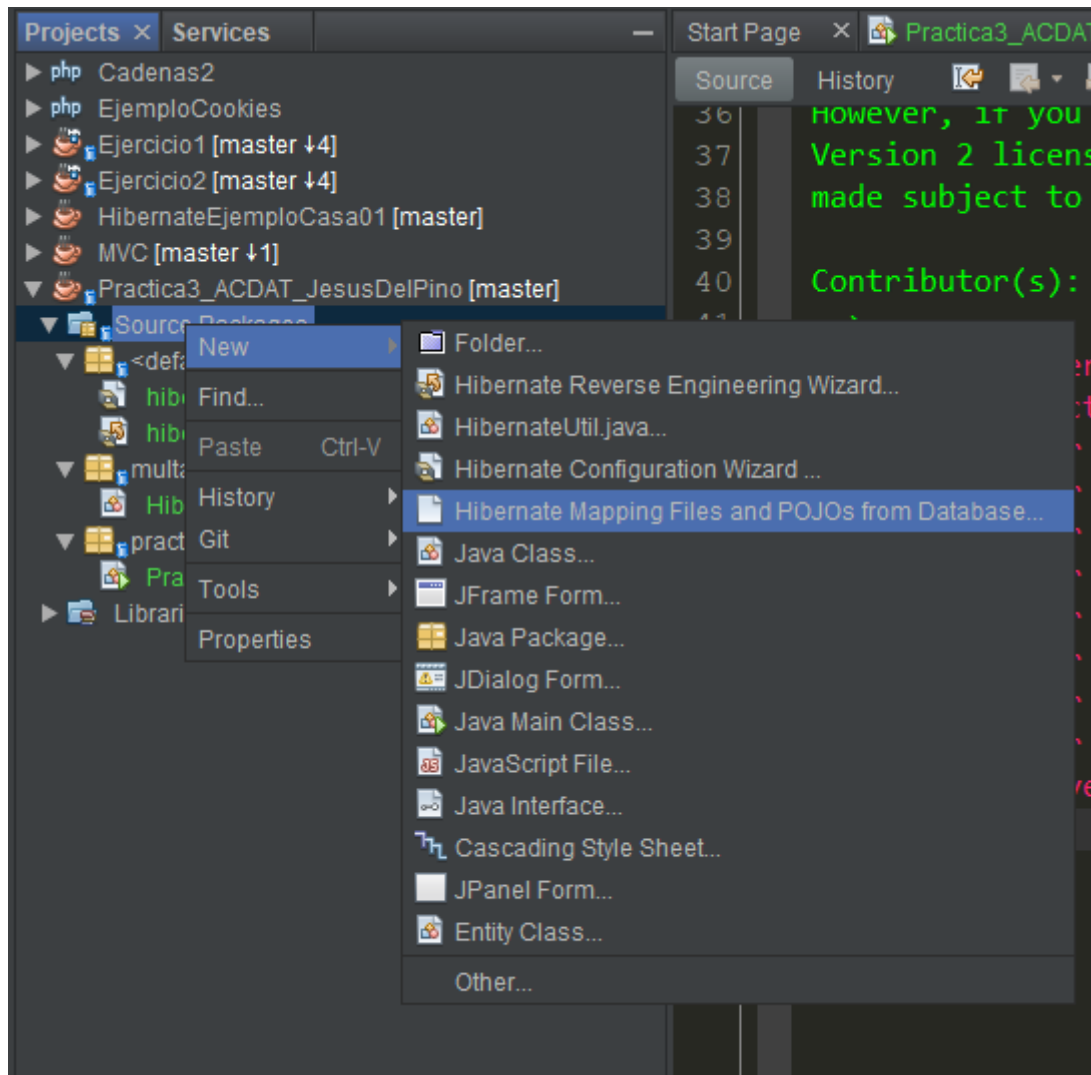
En la siguiente pantalla seleccionamos todas nuestras tablas y hacemos click en terminar:



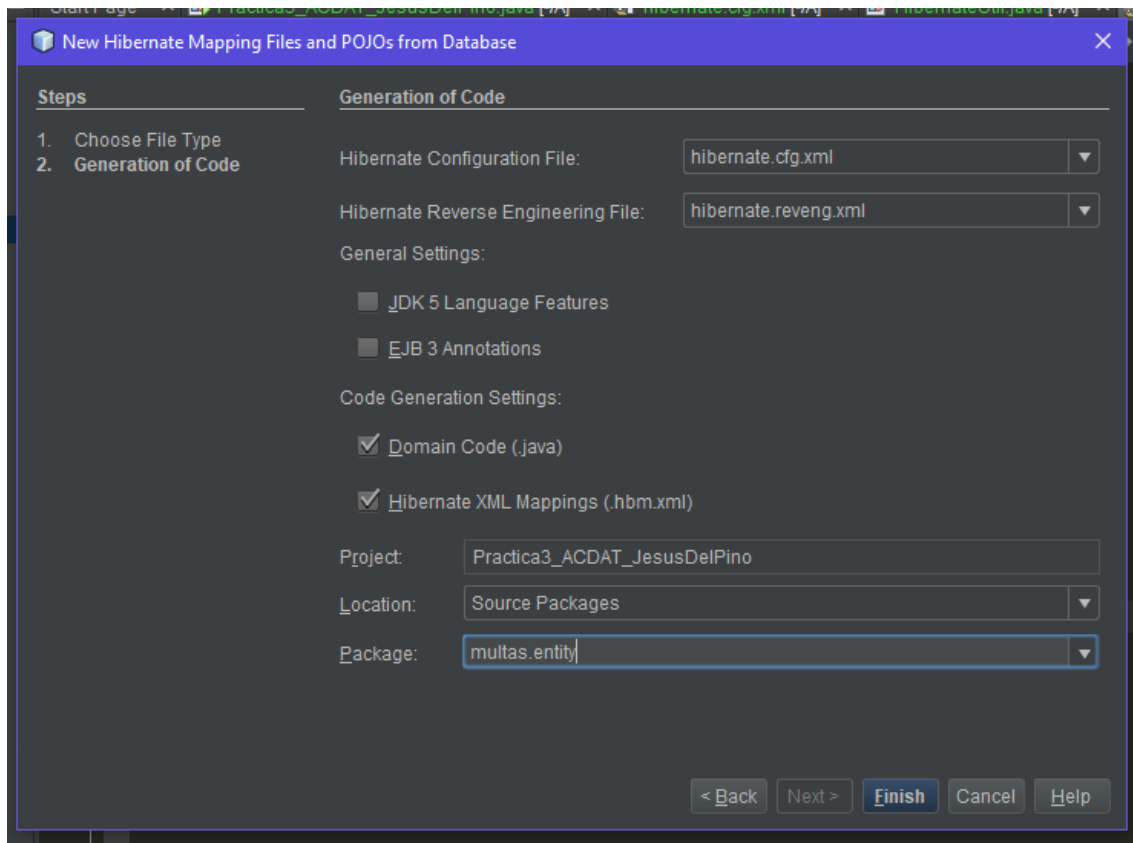


El asistente de archivos de mapeo y POJOs desde una base de datos, permite generar ficheros basados en tablas de la base de datos. Cuando usamos el asistente, NetBeans genera POJOs y archivos de mapeo basados en las tablas de la base de datos especificadas en **hibernate.reveng.xml** y los añade a las entradas de mapeo de *hibernate.cfg.xml*.

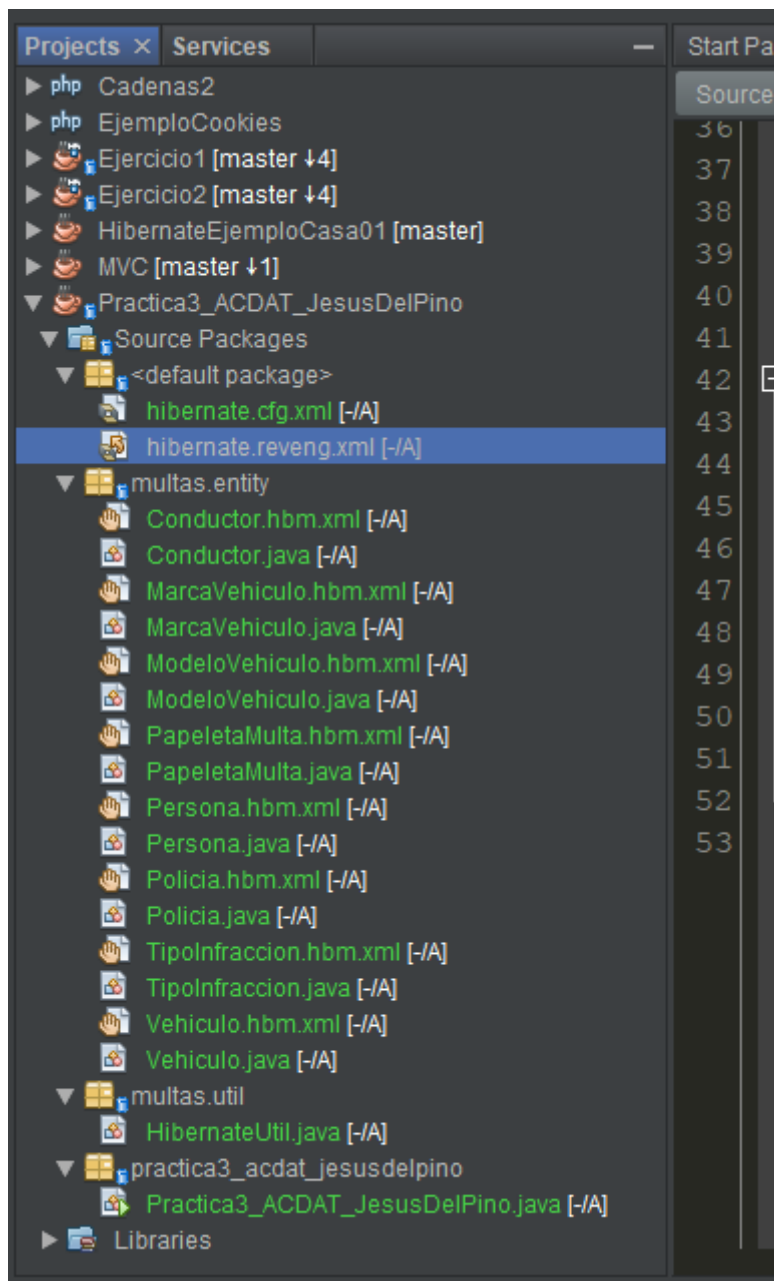
Para crear los archivos de mapeo hacemos click sobre el nodo "Source Packages", nuevo y seleccionamos la siguiente opción:



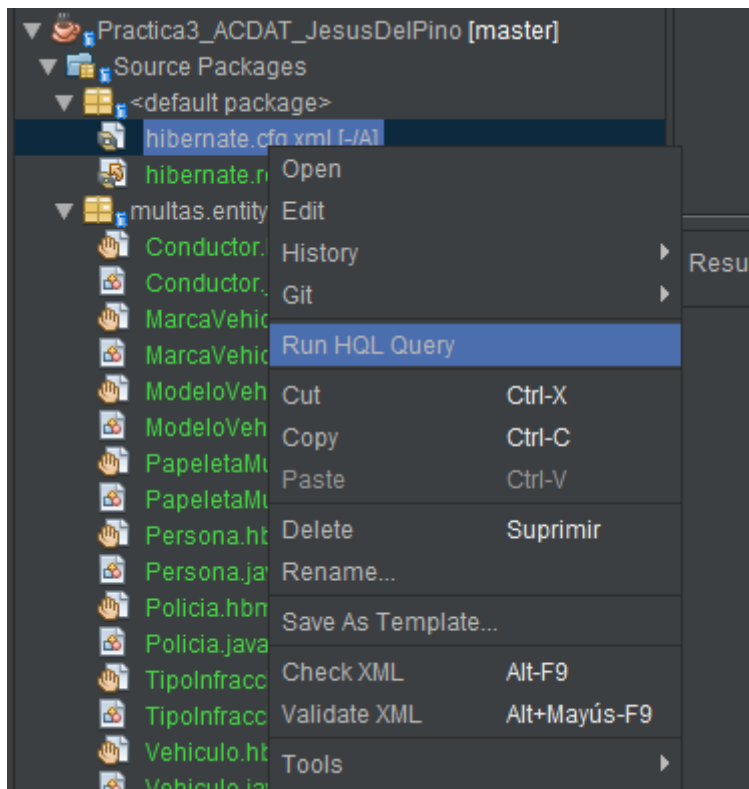
Dejamos todo por defecto y como nombre del paquete escribimos `multas.entity`:



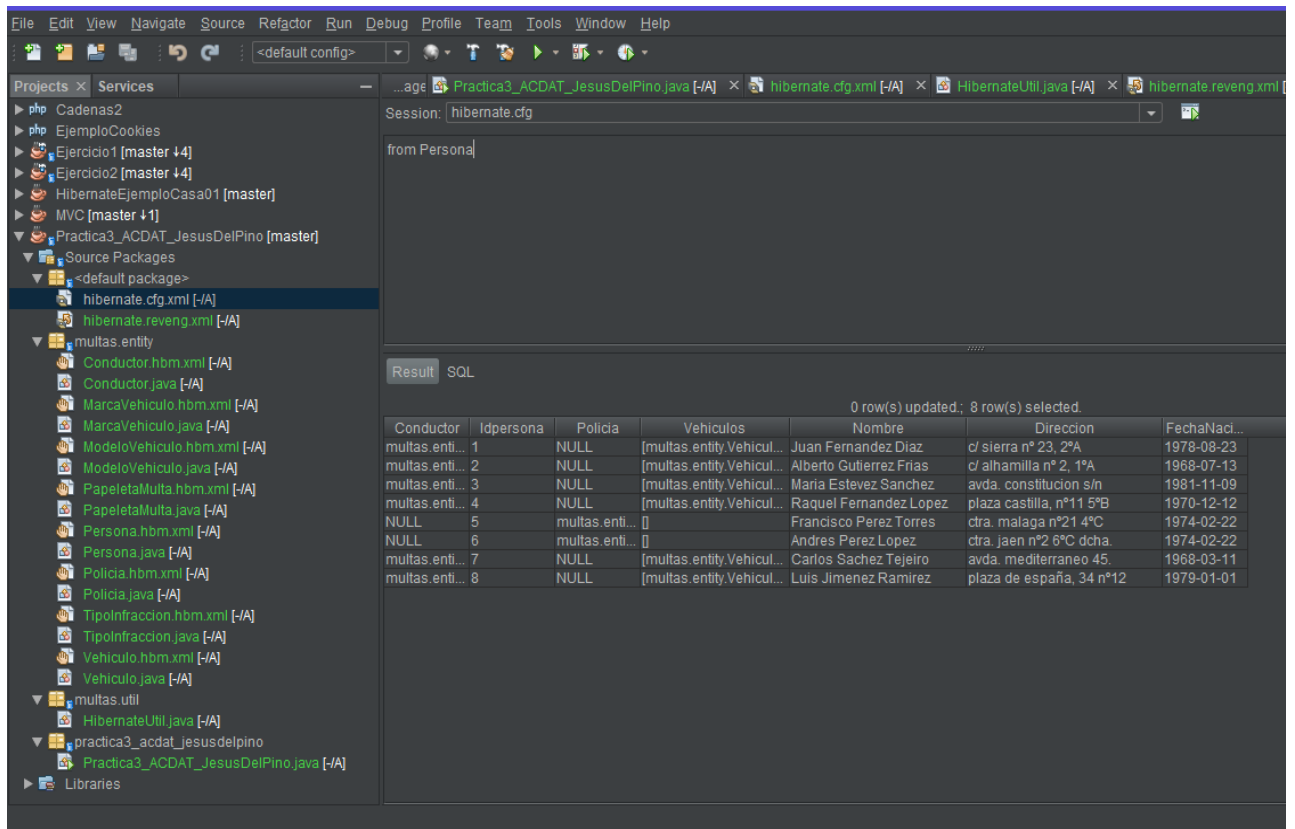
Hacemos click en terminar y se nos generarán las clases Java de todas nuestras tablas y los archivos de mapeo:



Para comprobar que la conexión a la base de datos funciona correctamente realizaremos consultas en HQL. Para ello, hacemos click derecho sobre hibernate.cfg.xml y seleccionamos “Run HQL query”.



Ejecutamos la consulta “from Personas”, que muestra todas las filas de la tabla Personas:



Consultas

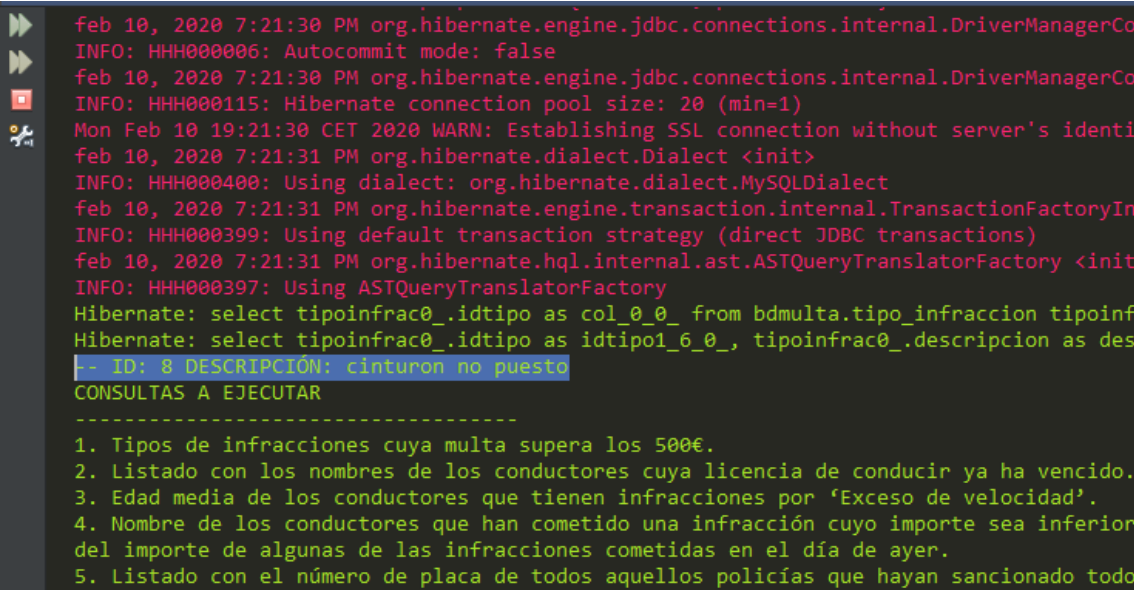
Consulta 1

En la primera consulta, obtendremos los tipos de infracciones cuya multa supera los 500€.

El código que hemos realizado para esta consulta es el siguiente:

```
TipoInfraccion inf = new TipoInfraccion();
SessionFactory sfactory = HibernateUtil.getSessionFactory();
Session session = sfactory.openSession();
Query q = session.createQuery("from TipoInfraccion where
importe>500");
Iterator<?> iter = q.iterate();
while (iter.hasNext()) {
    inf = (TipoInfraccion) iter.next();
    System.out.println("-- ID: "+inf.getIdtipo()+"
DESCRIPCIÓN: "+inf.getDescripcion());
}
```

Resultado obtenido tras la ejecución de la consulta:



```

feb 10, 2020 7:21:30 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerCo
INFO: HHH000006: Autocommit mode: false
feb 10, 2020 7:21:30 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerCo
INFO: HHH000115: Hibernate connection pool size: 20 (min=1)
Mon Feb 10 19:21:30 CET 2020 WARN: Establishing SSL connection without server's identi
feb 10, 2020 7:21:31 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
feb 10, 2020 7:21:31 PM org.hibernate.engine.transaction.internal.TransactionFactoryIn
INFO: HHH000399: Using default transaction strategy (direct JDBC transactions)
feb 10, 2020 7:21:31 PM org.hibernate.hql.internal.ast.ASTQueryTranslatorFactory <init
INFO: HHH000397: Using ASTQueryTranslatorFactory
Hibernate: select tipoinfrac0_.idtipo as col_0_0_ from bdmulta.tipoinfrac0_ tipoinfrac0_
Hibernate: select tipoinfrac0_.idtipo as idtipo1_6_0_, tipoinfrac0_.descripcion as des
-- ID: 8 DESCRIPCIÓN: cinturón no puesto
CONSULTAS A EJECUTAR
-----
1. Tipos de infracciones cuya multa supera los 500€.
2. Listado con los nombres de los conductores cuya licencia de conducir ya ha vencido.
3. Edad media de los conductores que tienen infracciones por 'Exceso de velocidad'.
4. Nombre de los conductores que han cometido una infracción cuyo importe sea inferior
del importe de algunas de las infracciones cometidas en el día de ayer.
5. Listado con el número de placa de todos aquellos policías que hayan sancionado todo

```

Consulta 2

En esta consulta listaremos los nombres de los conductores cuya licencia de conducir ya haya vencido.

El código que hemos realizado para esta consulta es el siguiente:

```

Conductor cond = new Conductor();
SessionFactory sfactory = HibernateUtil.getSessionFactory();
Session session = sfactory.openSession();
Query q = session.createQuery("from Conductor where
fechaExpiracion < now()");
Iterator<?> iter = q.iterate();
while (iter.hasNext()) {
    cond = (Conductor) iter.next();
    System.out.println("-- NOMBRE CONDUCTOR:
"+cond.getPersona().getNombre());
}

```

Resultado obtenido tras la ejecución de la consulta:

```

-----
INTRODUZCA UNA OPCIÓN
2
Hibernate: select conductor0_.idpersona as col_0_0_ from bdmulta.conductor conductor0_ where
Mon Feb 10 19:30:07 CET 2020 WARN: Establishing SSL connection without server's identity ver
Hibernate: select conductor0_.idpersona as idperson1_0_0_, conductor0_.licencia as licencia2
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nombre as nombre2_4_0_, p
-- NOMBRE CONDUCTOR: Juan Fernandez Diaz
Hibernate: select conductor0_.idpersona as idperson1_0_0_, conductor0_.licencia as licencia2
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nombre as nombre2_4_0_, p
-- NOMBRE CONDUCTOR: Alberto Gutierrez Frias
Hibernate: select conductor0_.idpersona as idperson1_0_0_, conductor0_.licencia as licencia2
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nombre as nombre2_4_0_, p
-- NOMBRE CONDUCTOR: Maria Estevez Sanchez
Hibernate: select conductor0_.idpersona as idperson1_0_0_, conductor0_.licencia as licencia2
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nombre as nombre2_4_0_, p
-- NOMBRE CONDUCTOR: Raquel Fernandez Lopez
Hibernate: select conductor0_.idpersona as idperson1_0_0_, conductor0_.licencia as licencia2
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nombre as nombre2_4_0_, p
-- NOMBRE CONDUCTOR: Carlos Sachez Tejeiro
Hibernate: select conductor0_.idpersona as idperson1_0_0_, conductor0_.licencia as licencia2
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nombre as nombre2_4_0_, p
-- NOMBRE CONDUCTOR: Luis Jimenez Ramirez
CONSULTAS A EJECUTAR
-----
1. Tipos de infracciones cuya multa supera los 500€.
2. Listado con los nombres de los conductores cuya licencia de conducir ya ha vencido.

```

Consulta 3

En esta consulta obtendremos la edad media de los conductores que tienen infracciones por 'Exceso de velocidad'.

El código que hemos realizado para esta consulta es el siguiente:

```

SessionFactory sfactory = HibernateUtil.getSessionFactory();
Session session = sfactory.openSession();
Query q = session.createQuery("select avg(datediff (now(),
fechaNacimiento)/365) from Persona where idpersona in (select
idpersona from Conductor where idpersona in (select conductor from
PapeletaMulta where idtipoinfraccion in (select idtipo from
TipoInfraccion where descripcion='exceso de velocidad'))))");

Double result = (Double) q.uniqueResult();

```

```
Long resultF = Math.round(result);
System.out.println("-- EDAD MEDIA: "+resultF);
```

Resultado obtenido tras la ejecución de la consulta:

```
1. Tipos de infracciones cuya multa supera los 500€.
2. Listado con los nombres de los conductores cuya licencia de conducir ya ha vencido
3. Edad media de los conductores que tienen infracciones por 'Exceso de velocidad'.
4. Nombre de los conductores que han cometido una infracción cuyo importe sea inferior
del importe de algunas de las infracciones cometidas en el día de ayer.
5. Listado con el número de placa de todos aquellos policías que hayan sancionado a
de infracciones.
6. Salir

-----
INTRODUZCA UNA OPCIÓN
3
Hibernate: select avg(datediff(now(), persona0_.fechaNacimiento)/365) as col_0_0_ f
Mon Feb 10 19:35:46 CET 2020 WARN: Establishing SSL connection without server's ide
-- EDAD MEDIA: 52
CONSULTAS A EJECUTAR
-----
1. Tipos de infracciones cuya multa supera los 500€.
2. Listado con los nombres de los conductores cuya licencia de conducir ya ha vencido
3. Edad media de los conductores que tienen infracciones por 'Exceso de velocidad'.
4. Nombre de los conductores que han cometido una infracción cuyo importe sea inferior
del importe de algunas de las infracciones cometidas en el día de ayer.
```

Consulta 4

En esta consulta obtendremos el nombre de los conductores que han cometido una infracción cuyo importe sea inferior al del importe de algunas de las infracciones cometidas en el día de ayer.

El código que hemos realizado para esta consulta es el siguiente:

```
Persona pers = new Persona();
SessionFactory sfactory = HibernateUtil.getSessionFactory();
Session session = sfactory.openSession();
Query q = session.createQuery("from Persona as per where
idpersona in (select conductor from PapeletaMulta where tipoInfraccion
in (select idtipo from TipoInfraccion where importe < (select
max(importe) from TipoInfraccion where idtipo in (select
tipoInfraccion from PapeletaMulta where datediff(now(),
fecha)=1 )))");
Iterator<?> iter = q.iterate();
while (iter.hasNext()) {
pers = (Persona) iter.next();
System.out.println("-- NOMBRE: "+pers.getNombre());
}
```

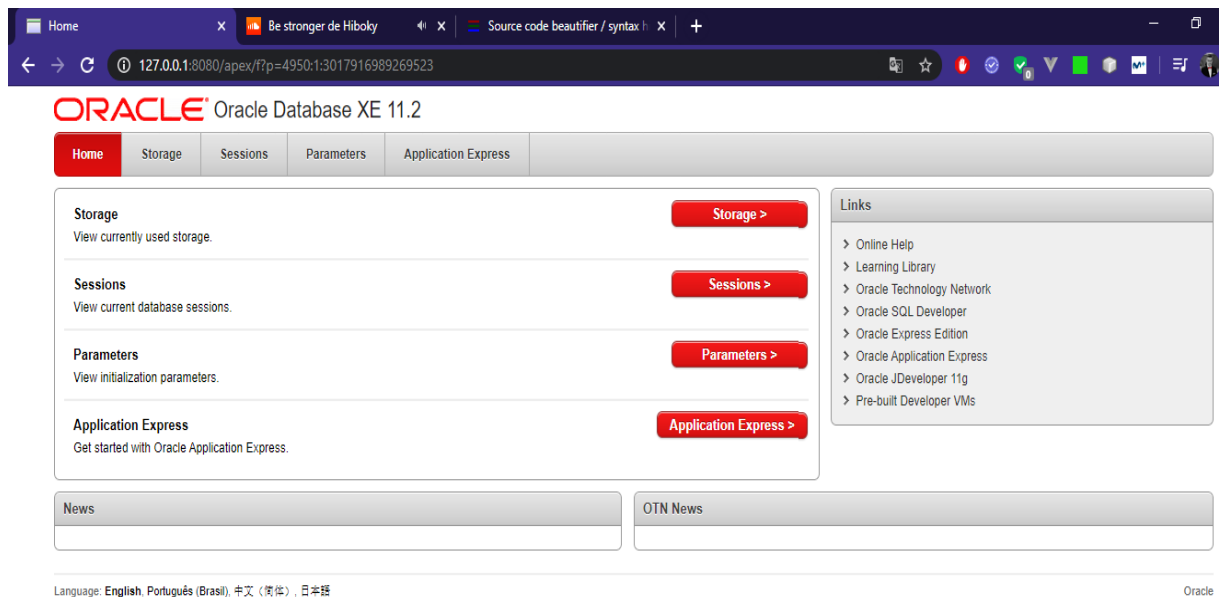
Resultado obtenido tras la ejecución de la consulta:

```
Output - Practica3_ACDAT_JesusDelPino (run)
6. Salir
-----
INTRODUZCA UNA OPCIÓN
4
Hibernate: select persona0_.idpersona as col_0_0_ from bdmulta.persona
Mon Feb 10 19:41:59 CET 2020 WARN: Establishing SSL connection without
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nomb
-- NOMBRE: Alberto Gutierrez Frias
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nomb
-- NOMBRE: Raquel Fernandez Lopez
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nomb
-- NOMBRE: Carlos Sacher Tejeiro
Hibernate: select persona0_.idpersona as idperson1_4_0_, persona0_.nomb
-- NOMBRE: Juan Fernandez Diaz
CONSULTAS A EJECUTAR
-----
1. Tipos de infracciones cuya multa supera los 500€.
2. Listado con los nombres de los conductores cuya licencia de conducir
3. Edad media de los conductores que tienen infracciones por 'Exceso de
4. Nombre de los conductores que han cometido una infracción cuyo import
del importe de algunas de las infracciones cometidas en el día de ayer
```

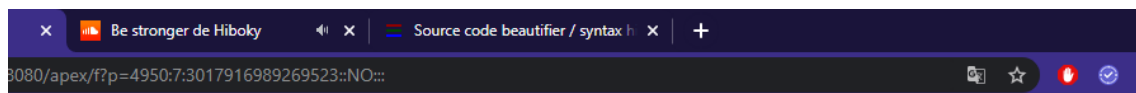
Sistema gestor Oracle

Instalación del sistema gestor y creación de la BD multas

Cuando ejecutamos el instalador nos pedirá que ingresemos una contraseña y una vez instalado nos proporciona un acceso directo que nos lleva a la siguiente dirección:



Para empezar a utilizar el sistema gestor haremos click en Application Express. A continuación, nos pedirá que iniciemos sesión, utilizaremos el usuario system y la contraseña que introdujimos durante la instalación:



Login

Username

system

Password

.....

Login

Login as a database user which has been granted the DBA database role (for example, SYSTEM).

Nos llevará a la siguiente pantalla, en la que tendremos que crear un usuario para la base de datos con su contraseña o, en el caso de que ya lo hayamos creado nos registraremos mediante la opción que aparece a la derecha de la pantalla:

PRÁCTICA 3. DESFASE OBJETO-RELACIONAL | Jesús Del Pino, José Abel Viñolo

The screenshot shows the Oracle Database XE 11.2 Application Express interface. The browser address bar displays `127.0.0.1:8080/apex/www_flow.accept`. The page title is "ORACLE® Oracle Database XE 11.2". The navigation bar includes links for Home, Storage, Sessions, Parameters, and Application Express. The main content area is titled "Create Application Express Workspace" and contains the following fields:

- Database User:** Radio buttons for "Create New" (selected) and "Use Existing".
- * Database Username:** Text field containing "Multas".
- * Application Express Username:** Text field containing "root".
- * Password:** Text field with masked characters (*****). A red error message below it states "Passwords do not match."
- * Confirm Password:** Text field with masked characters (*****). A red error message below it states "Confirm Password must have some value."

Buttons for "Cancel" and "Create Workspace" are located at the top right of the form. To the right of the form is a "Getting Started" section with a link "Already have an account? Login Here" and a paragraph explaining how to get started with Oracle Application Express, including a list of required information: Database Username, Application Express Username, and Password.

Iniciamos sesión para entrar al sistema gestor de Oracle:

The screenshot shows the Oracle Application Express login page. The browser address bar displays `127.0.0.1:8080/apex/f?p=4550:1:1401719560764930`. The page title is "ORACLE® Application Express". The main content area is titled "Enter Application Express workspace and credentials." and contains the following fields:

- Workspace:** Text field containing "MULTAS".
- Username:** Text field containing "ROOT".
- Password:** Text field with masked characters (*****).

A "Login" button is located below the password field. Below the login fields is a link "Click here to learn how to get started". At the bottom of the page, there is a paragraph describing Oracle Application Express as a rapid Web application development tool. Below this paragraph is a language selection dropdown menu showing "Language: English, Português (Brasil), 中文 (简体), 日本語". At the bottom of the page, there are three columns of links: "Workspace" (Reset Password, Find My Workspace, Administration), "Getting Started" (Learn ..., Oracle Technology Network, apex.oracle.com, Oracle by Example's), and "Community" (Discussion Forum, Packaged Applications, Partners, BLOGs).

The screenshot shows the Oracle Application Express Administration page for workspace MULTAS. The top navigation bar includes Home, Application Builder, SQL Workshop, Team Development, and Administration. The main content area has four icons: Manage Service, Manage Users and Groups, Monitor Activity, and Dashboards. Below these are three tables: Service, Users, and Activity.

Service	
Workspace Name	MULTAS
Applications	1
Application Pages	23
Web sheets	0
SQL Scripts	0
Schemas	1
Open Requests	0

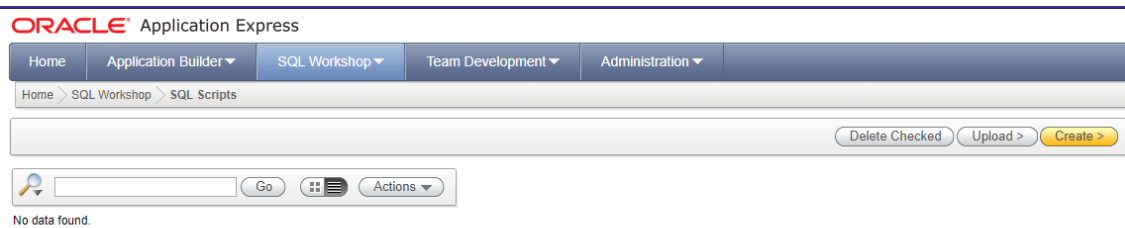
Users	
Users	1
Workspace Administrators	1
Application Developers	1
Worksheet Developers	0
End Users	0
Created Last 24 Hours	1
Created Last Week	1

Activity	
Reporting Timeframe	
Page Events	
Median Page Time	
Distinct Applications	
Distinct Users	
Distinct Sessions	
Errors	

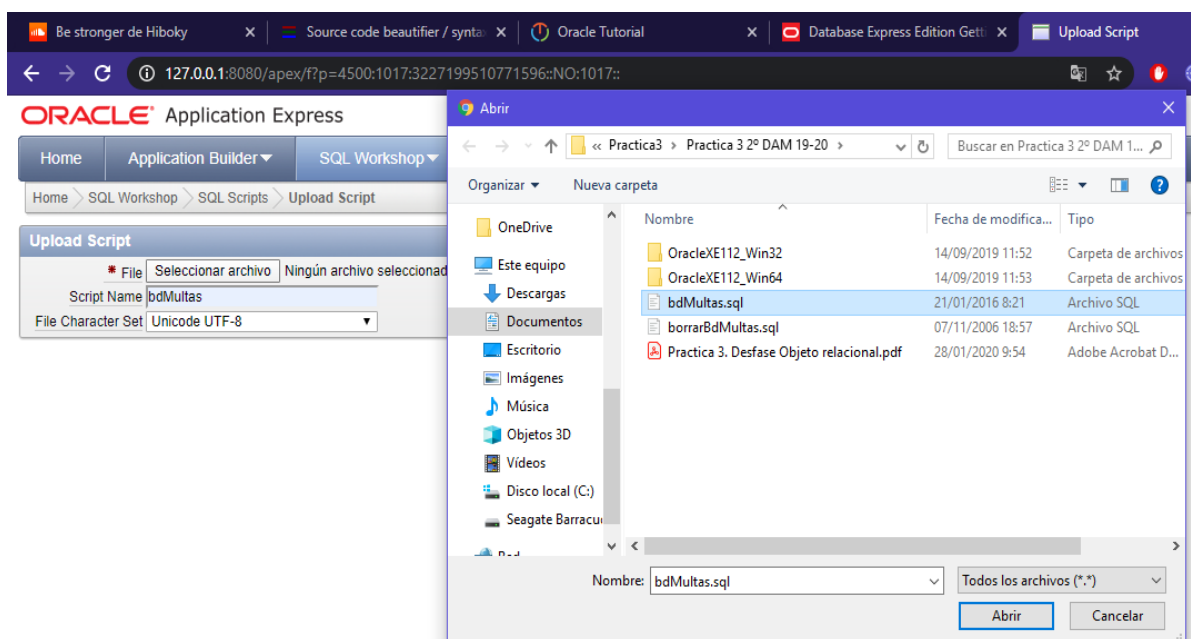
Ahora vamos a insertar el script bdMultas. Para ello nos vamos a la barra de herramientas y hacemos click en SQL Workshop/SQL Scripts:

The screenshot shows the Oracle Application Express SQL Workshop page. The top navigation bar includes Home, Application Builder, SQL Workshop, and Team Development. The main content area has four icons: Object Browser, SQL Commands, SQL Scripts, and Query Builder. A dropdown menu is open over the SQL Scripts icon, showing the following options: Object Browser, SQL Commands, SQL Scripts (highlighted), Query Builder, Utilities, - Data Workshop, - Object Reports, and - User Interface Defaults. Below the icons is a section titled 'Recently Created Tables' with two entries: APEX\$_WS_WEBPG_SECTION_HISTORY and APEX\$_WS_FILES.

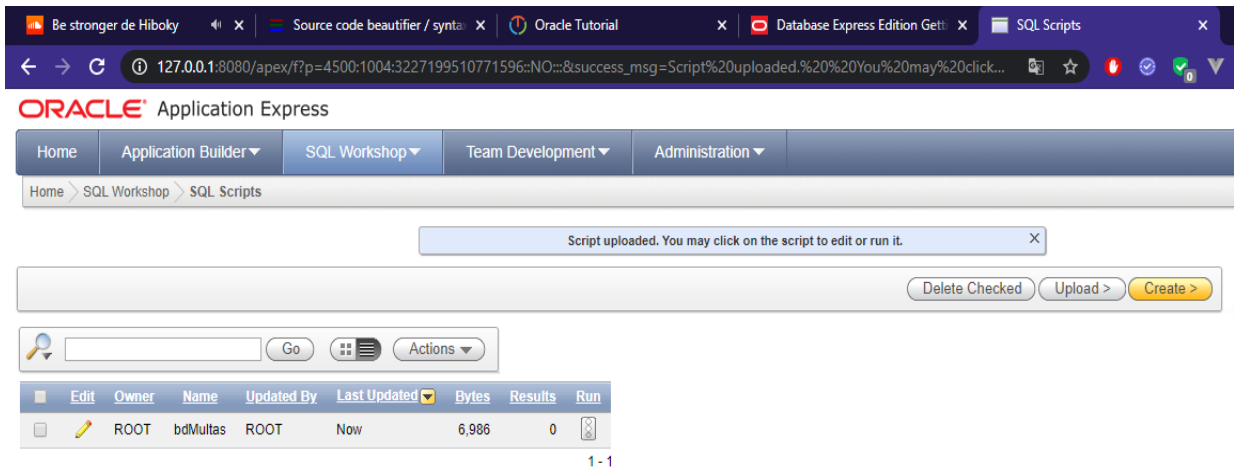
Hacemos click en upload:



Seleccionamos nuestro script:



Ya tendríamos nuestro script cargado y sería simplemente pulsar el botón de Run:



Oracle Application Express

Home Application Builder SQL Workshop Team Development Administration

Home > SQL Workshop > SQL Scripts

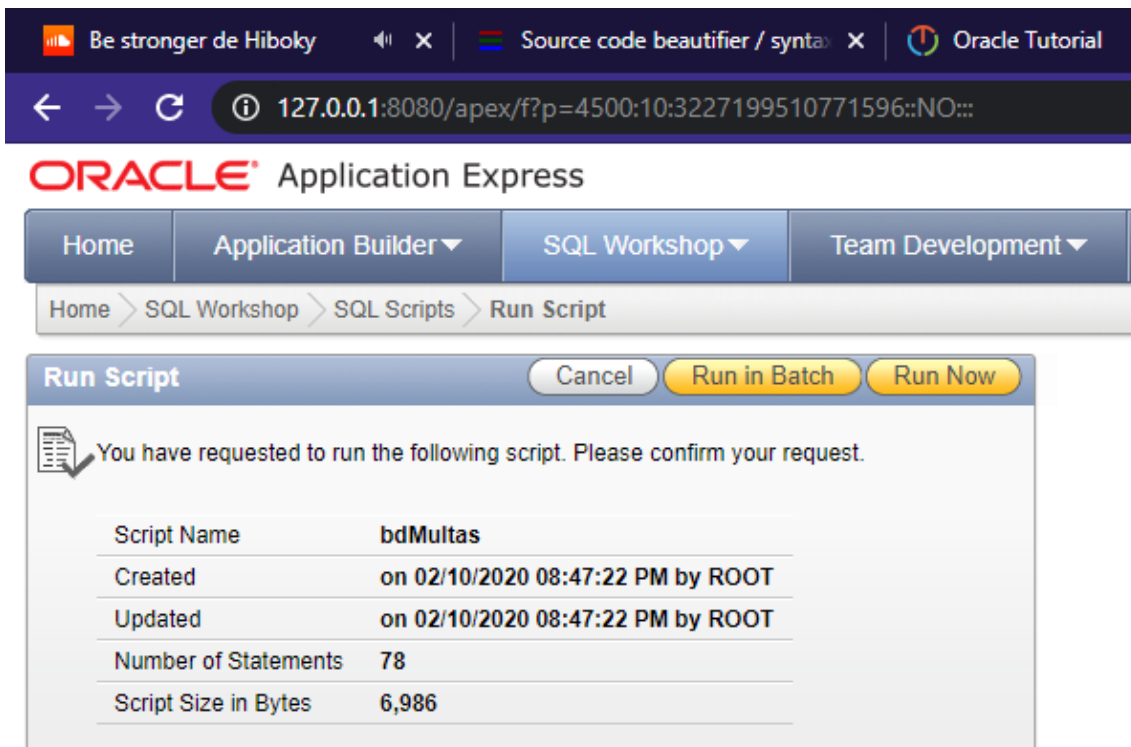
Script uploaded. You may click on the script to edit or run it.

Delete Checked Upload > Create >

Go Actions

	Edit	Owner	Name	Updated By	Last Updated	Bytes	Results	Run
<input type="checkbox"/>		ROOT	bdMultas	ROOT	Now	6,986	0	

1 - 1



Oracle Application Express

Home Application Builder SQL Workshop Team Development

Home > SQL Workshop > SQL Scripts > Run Script

Run Script Cancel Run in Batch Run Now

You have requested to run the following script. Please confirm your request.

Script Name	bdMultas
Created	on 02/10/2020 08:47:22 PM by ROOT
Updated	on 02/10/2020 08:47:22 PM by ROOT
Number of Statements	78
Script Size in Bytes	6,986

Ya se han creado nuestras tablas:

The screenshot shows the Oracle Application Express interface. The browser address bar indicates the URL: 127.0.0.1:8080/apex/f?p=4500:1225:3227199510771596::NO::P1224_RESULT_ID:4858320225749984. The Oracle logo and 'Application Express' text are visible. The navigation menu includes 'Home', 'Application Builder', 'SQL Workshop', 'Team Development', and 'Administration'. The breadcrumb trail shows 'Home > SQL Workshop > SQL Scripts > Results'.

Script: **bdMultas** Status: **Complete**
View: ☐ Detail ☒ Summary Rows: 15

Number	Elapsed	Statement	Feedback	Rows
1	0.06	create table tipo_infraccion (idtipo smallint not null	Table created.	0
2	0.01	create sequence tipolnf increment by 1 minvalue 1 nomaxva	Sequence created.	0
3	0.00	insert into tipo_infraccion values(tipolnf.nextval,'exceso d	1 row(s) inserted.	1
4	0.00	insert into tipo_infraccion values(tipolnf.nextval,'mal apar	1 row(s) inserted.	1
5	0.01	insert into tipo_infraccion values(tipolnf.nextval,'cinturon	1 row(s) inserted.	1
6	0.00	insert into tipo_infraccion values(tipolnf.nextval,'exceso d	1 row(s) inserted.	1
7	0.00	insert into tipo_infraccion values(tipolnf.nextval,'semaforo	1 row(s) inserted.	1
8	0.00	COMMIT	Statement processed.	0
9	0.01	create table persona (idpersona smallint not null prima	Table created.	0
10	0.01	create sequence person increment by 1 minvalue 1 nomaxval	Sequence created.	0
11	0.00	insert into persona values(person.nextval,'Juan Fernandez Di	ORA-01843: not a valid month	-
12	0.00	insert into persona values(person.nextval,'Alberto Gutierrez	ORA-01843: not a valid month	-
13	0.00	insert into persona values(person.nextval,'Maria Estevez San	1 row(s) inserted.	1
14	0.00	insert into persona values(person.nextval,'Raquel Fernandez	1 row(s) inserted.	1
15	0.00	insert into persona values(person.nextval,'Francisco Perez T	ORA-01843: not a valid month	-

row(s) 1 - 15 of 78

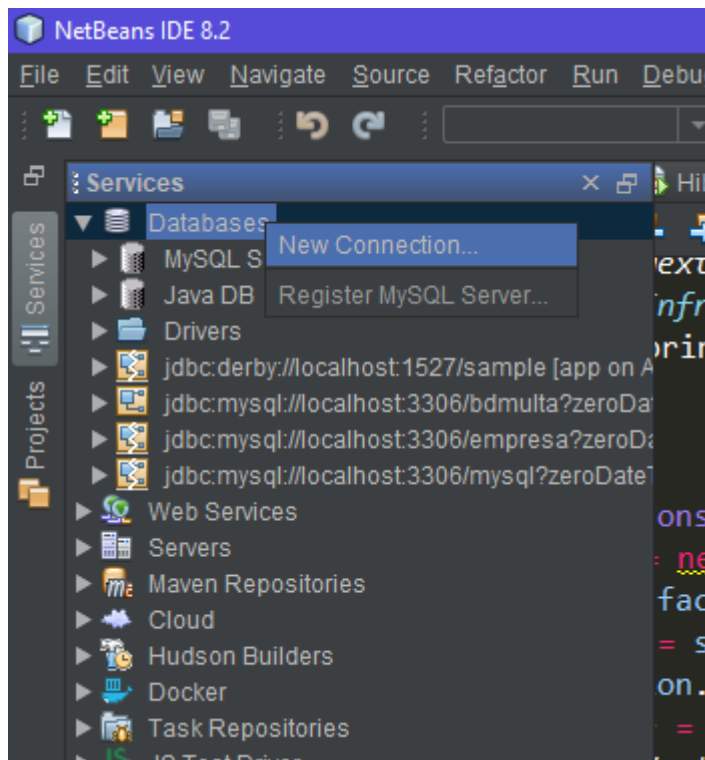
[Download](#)

Statements Processed | 78

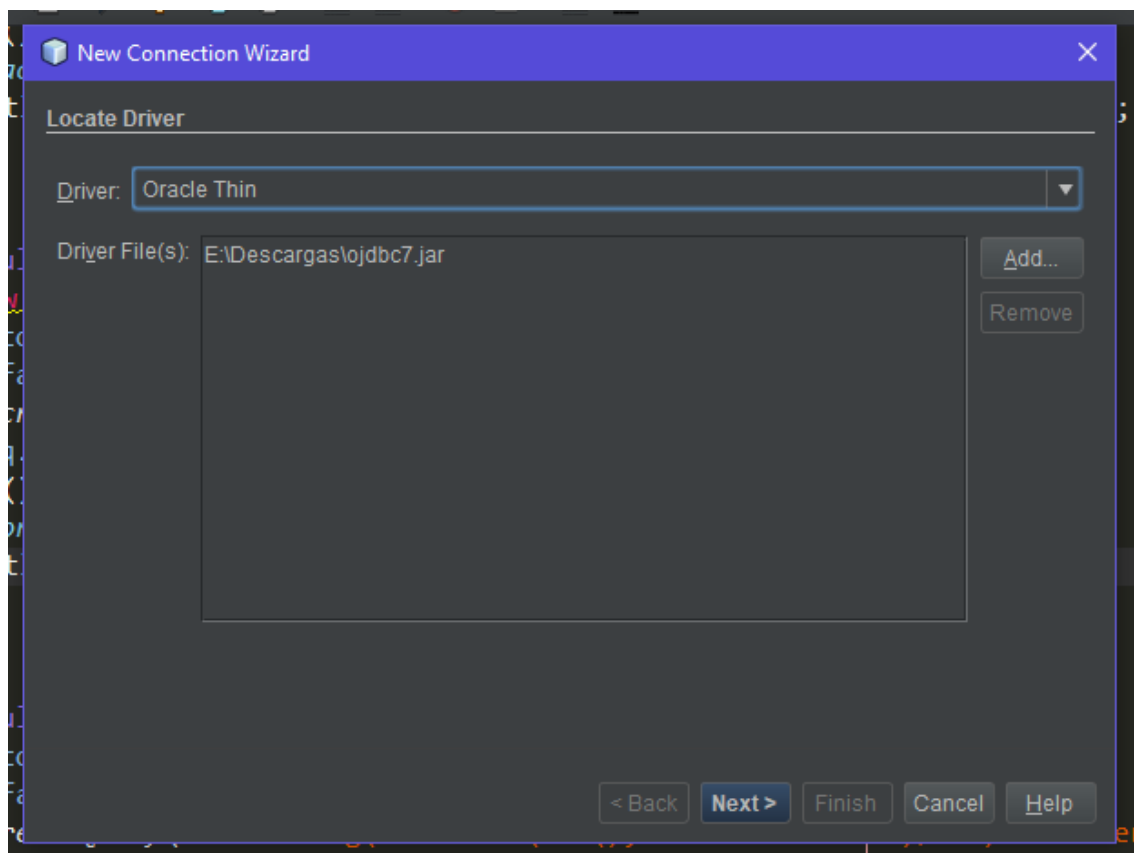
Mapeo utilizando Hibernate

Para el mapeo con Hibernate lo haremos del mismo modo que anteriormente, la diferencia estará en que crearemos una conexión a una BD Oracle utilizando el correspondiente conector:

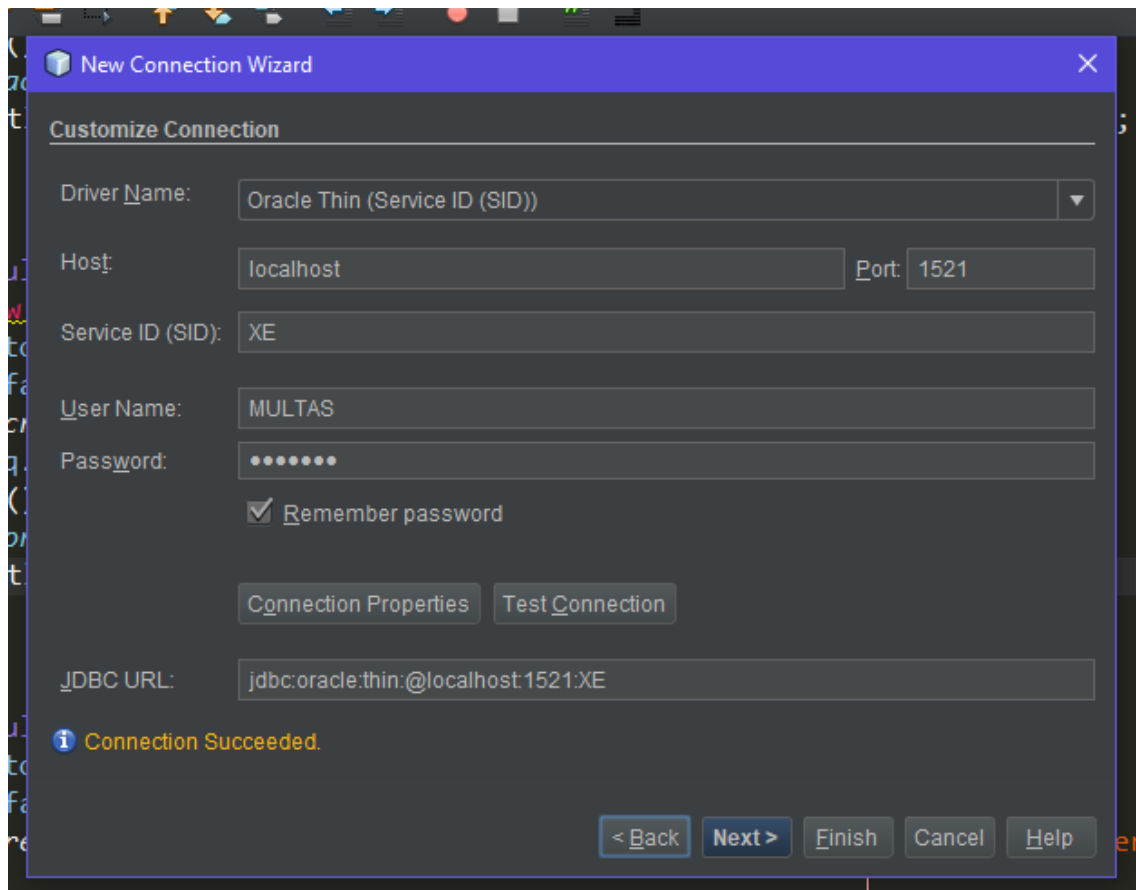
Lo primero, será crear la conexión a la BD, para ello Nos vamos a la pestaña servicios y creamos una nueva conexión:



Escogemos la opción de Oracle Thin y cargamos nuestro conector:

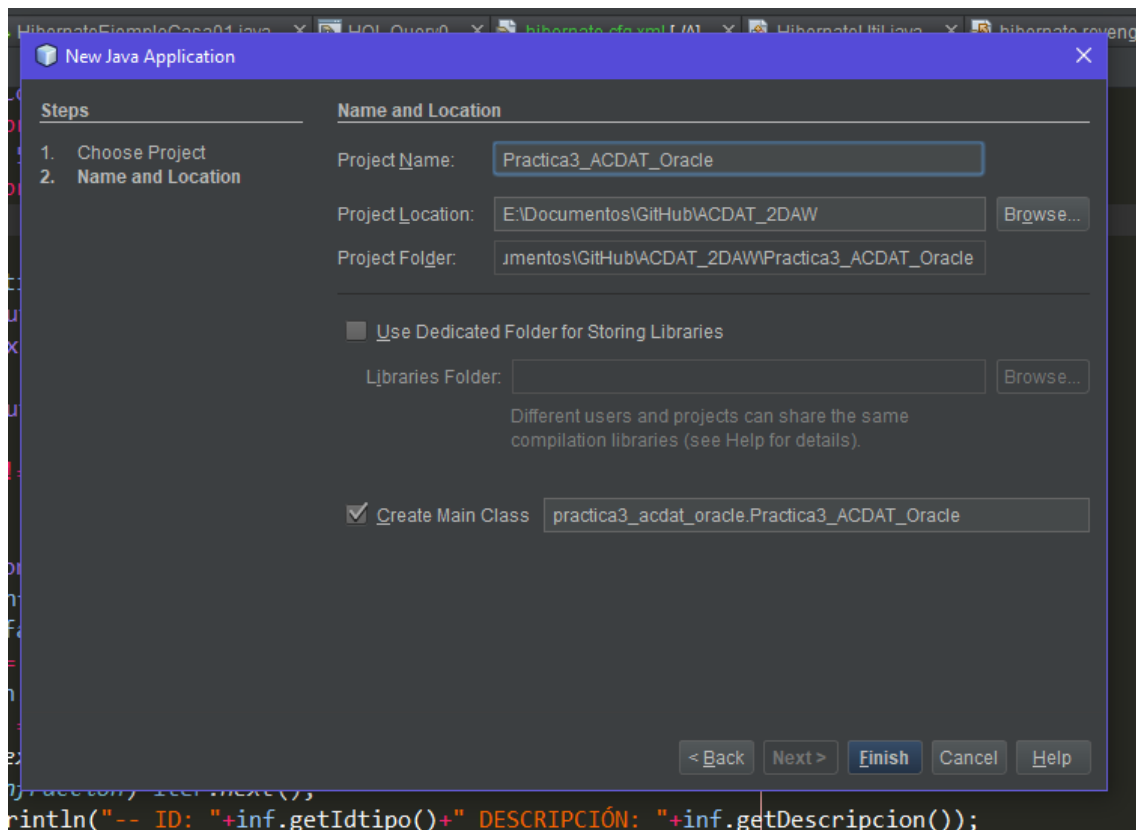


Rellenamos los campos y probamos la conexión para ver si funciona correctamente:

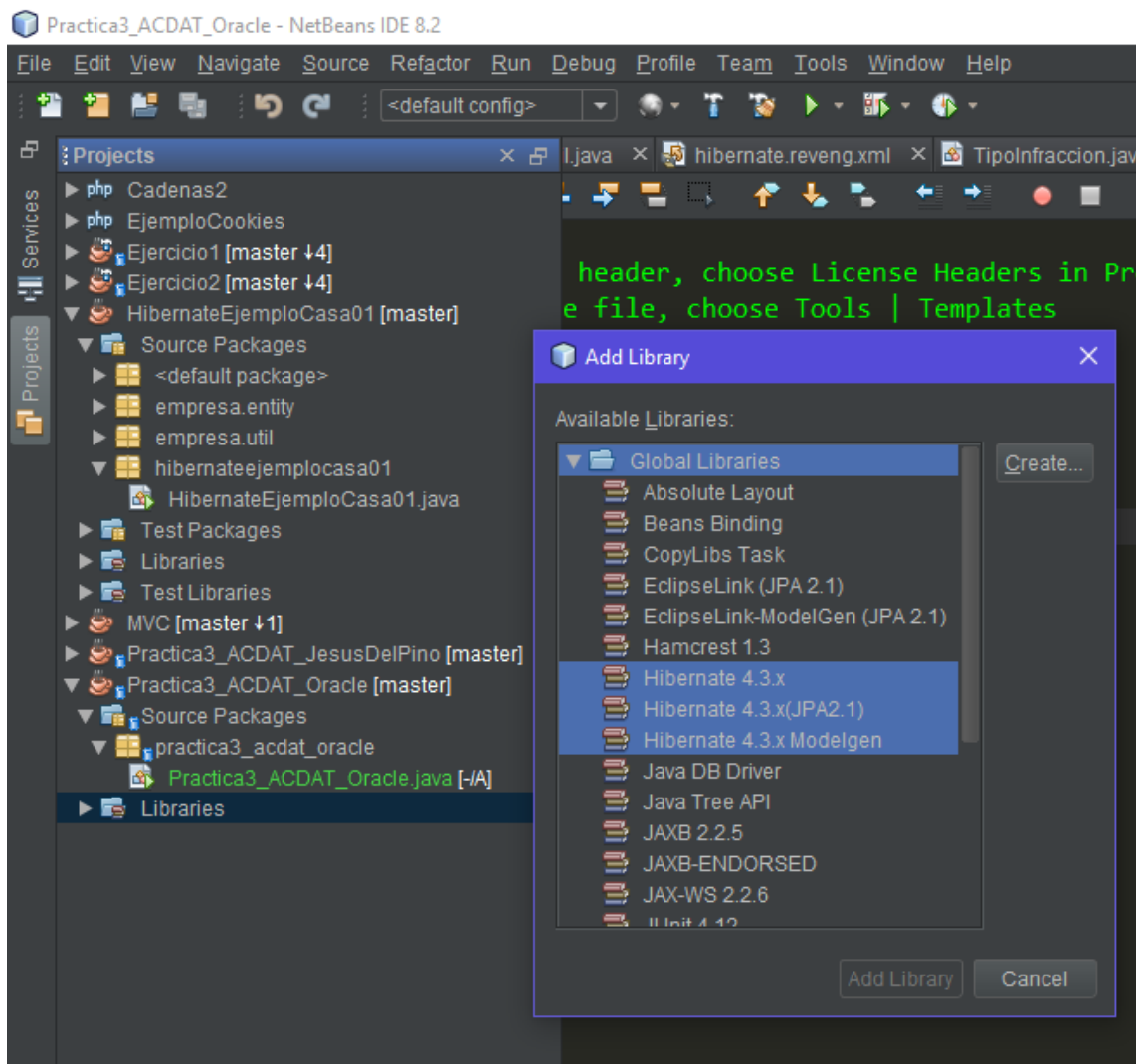


Pulsamos en Finish y ya tenemos nuestra conexión preparada.

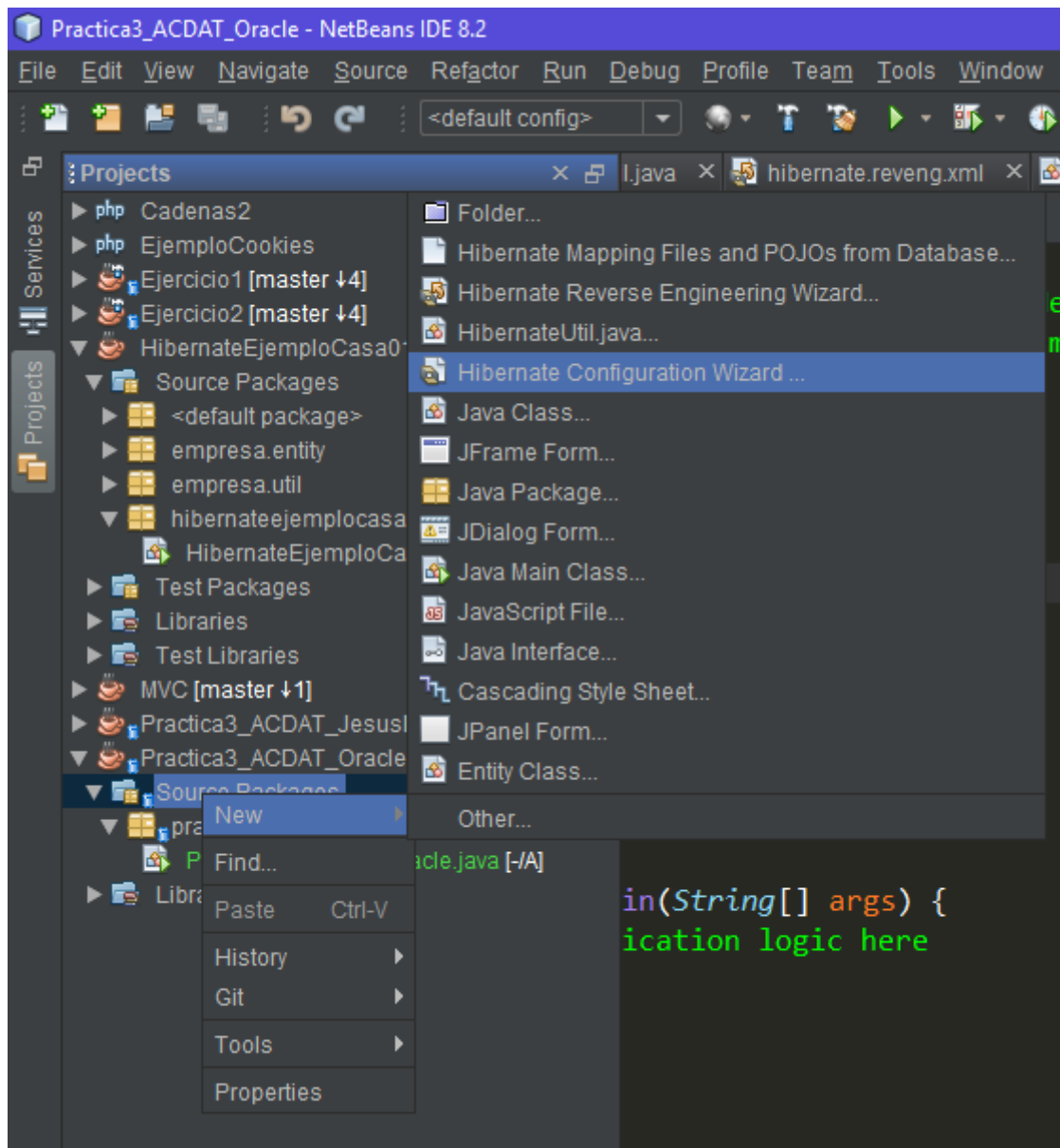
Vamos a crear un nuevo proyecto, en mi caso lo llamaré Practica3_ACDAT_Oracle:



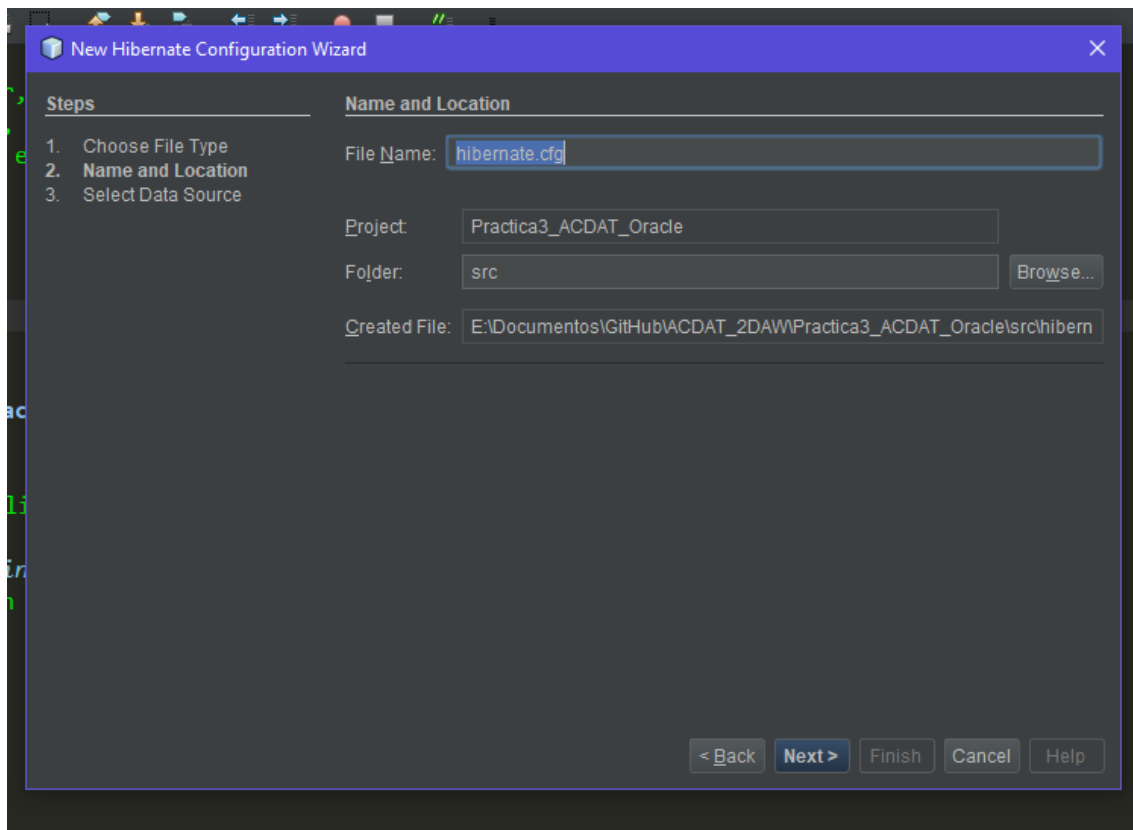
Añadimos las librerías de hibernate a nuestro proyecto:



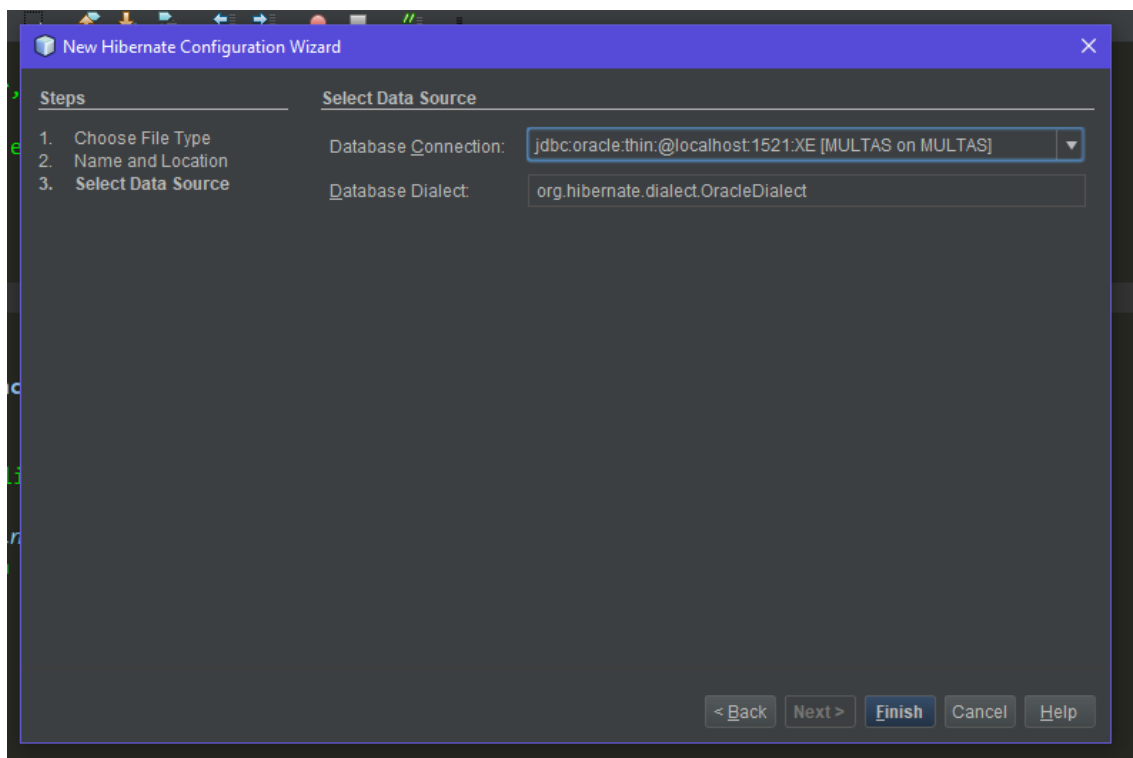
Ahora vamos a crear el fichero de configuración de Hibernate hibernate.cfg.xml. Para ello, hacemos click derecho sobre Source Packages y nos vamos a Nuevo/Asistente de configuración de Hibernate:



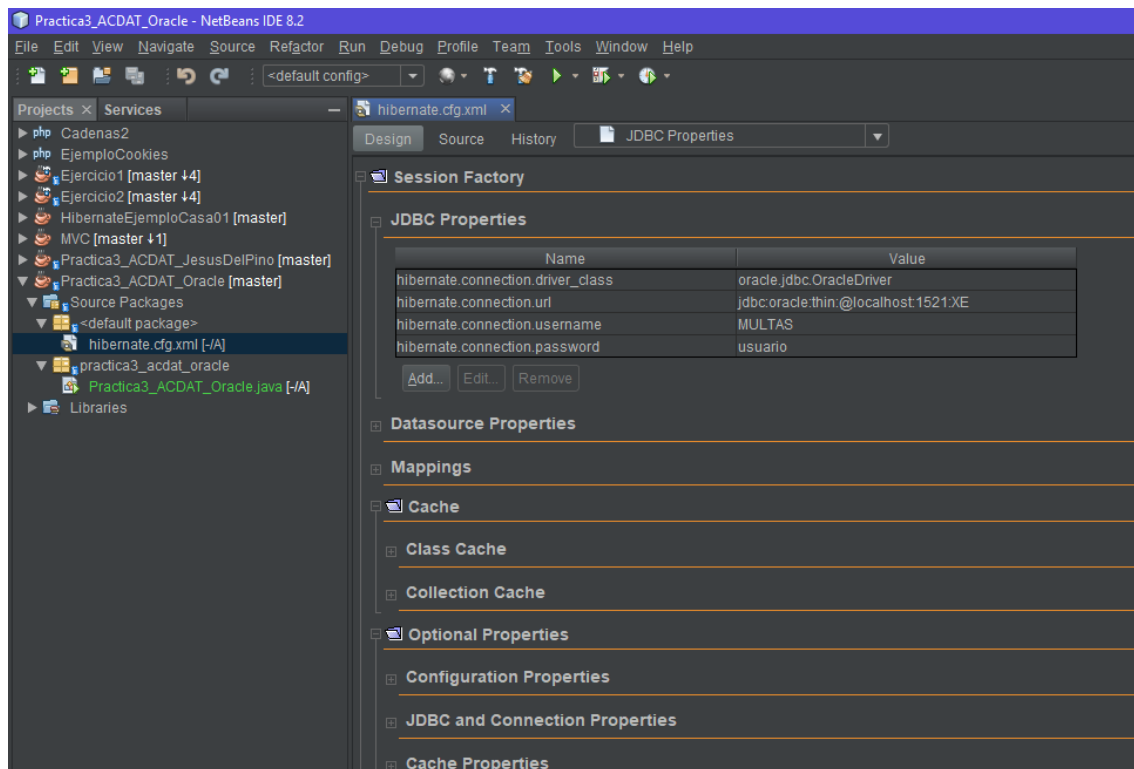
Dejamos el nombre por defecto:



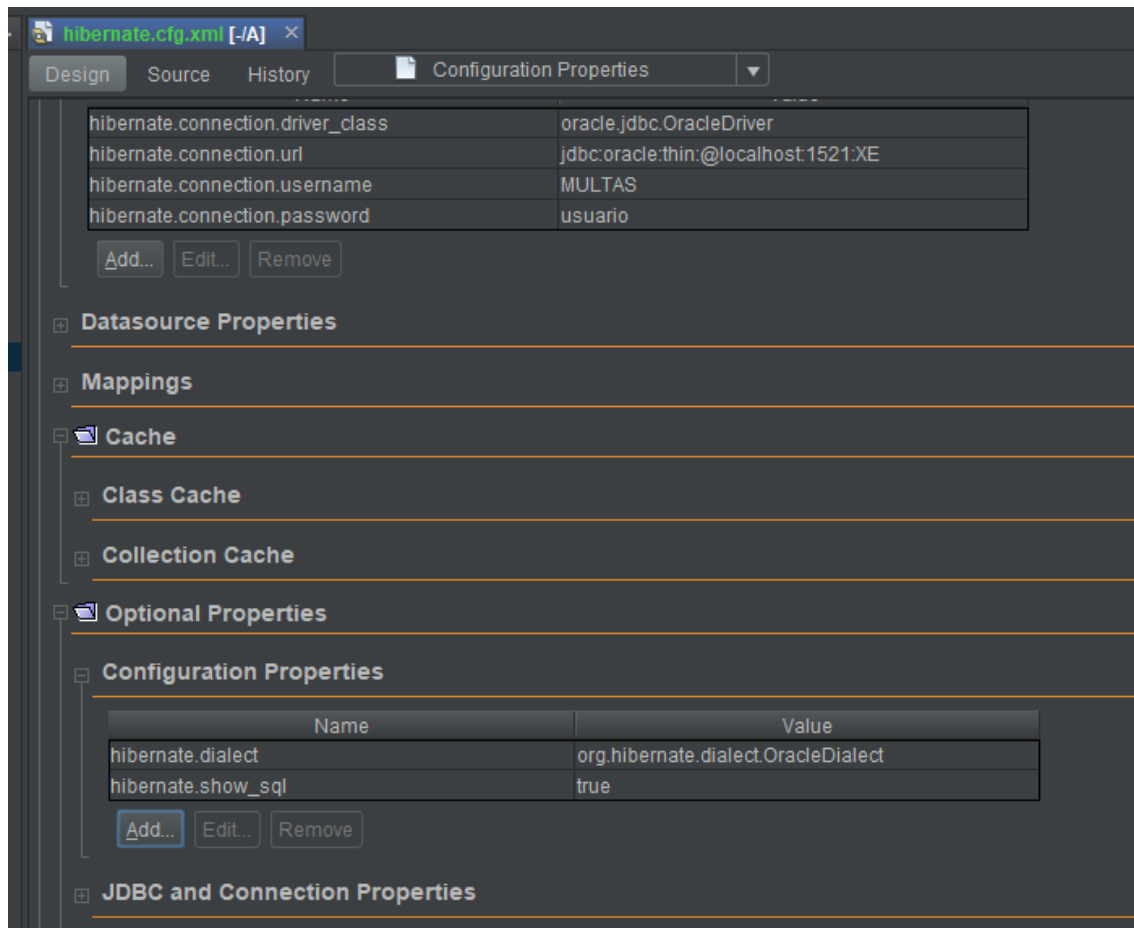
Seleccionamos la conexión que creamos anteriormente y pulsamos Finish:



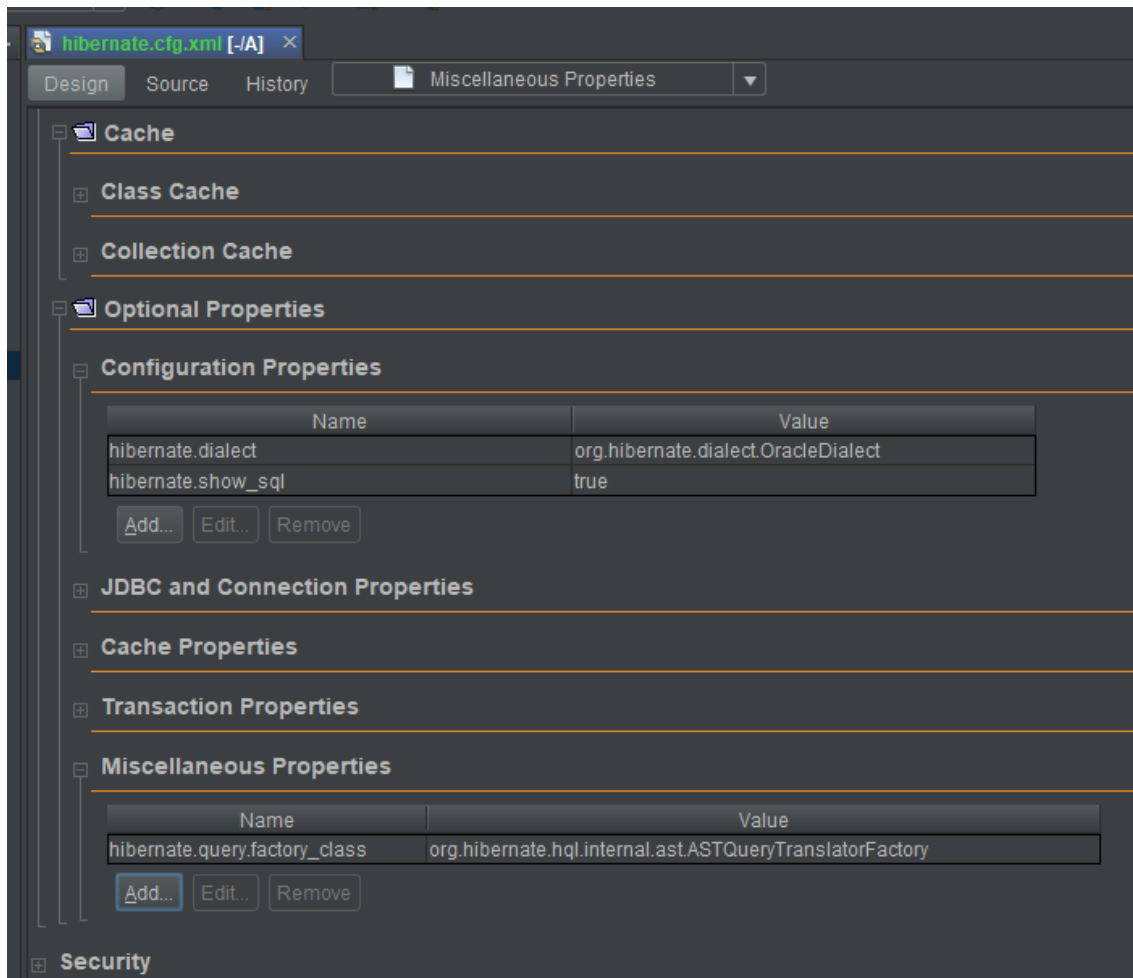
Abrimos nuestro fichero de configuración y nos vamos a la pestaña de diseño:



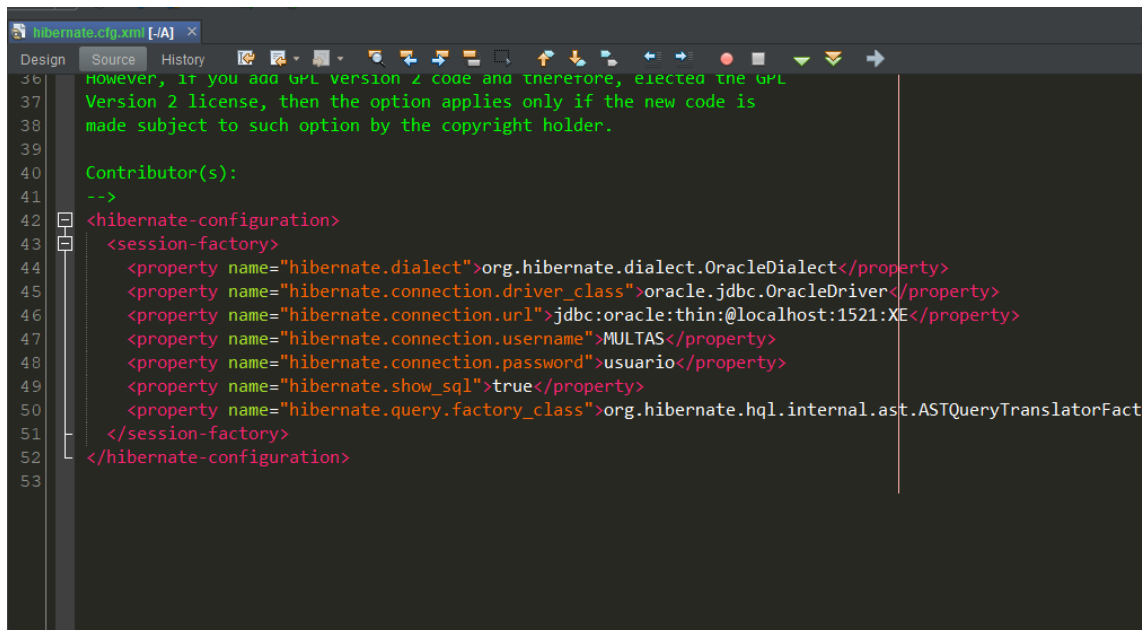
Nos vamos al apartado de Propiedades Opcionales/Propiedades de configuración, agregamos la propiedad Hibernate y habilitamos el seguimiento de la depuración de las consultas SQL



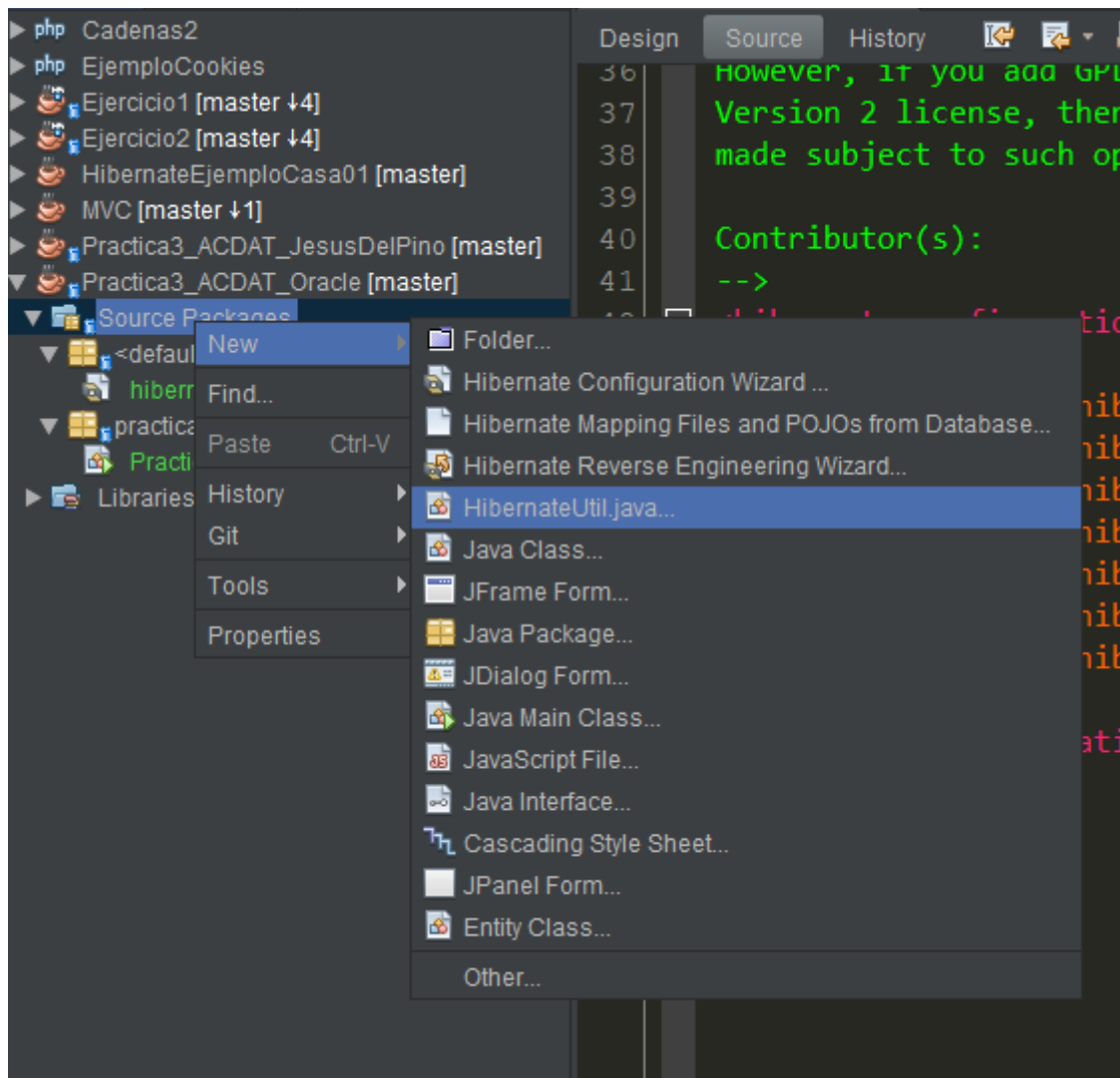
Ahora en Propiedades varias agregamos hibernate.query.factory_class:



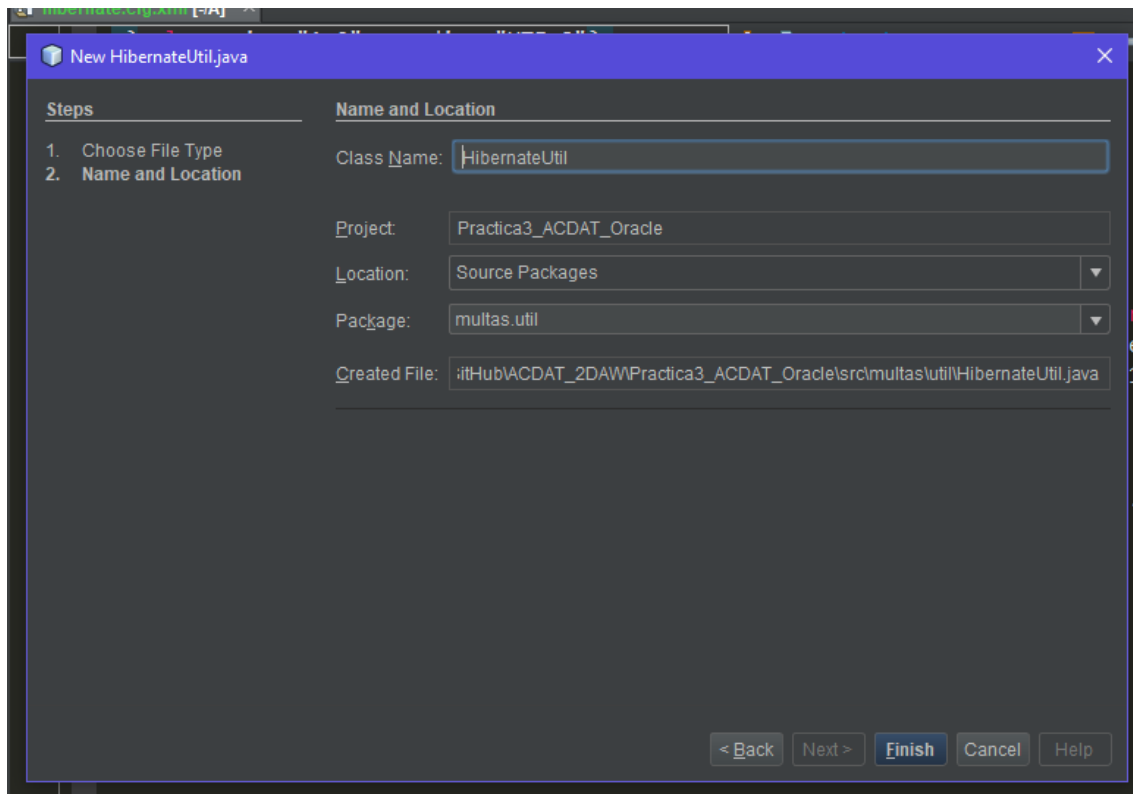
Vemos como queda nuestro fichero de configuración:



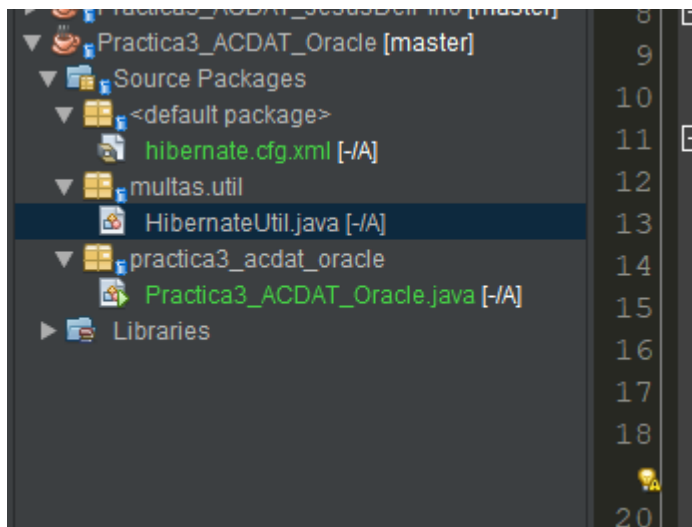
El siguiente paso es crear la clase `HibernateUtil.javaHelperFile`. Para ello, nos vamos a Source Packages, hacemos click derecho, Nuevo/HibernateUtil.java:



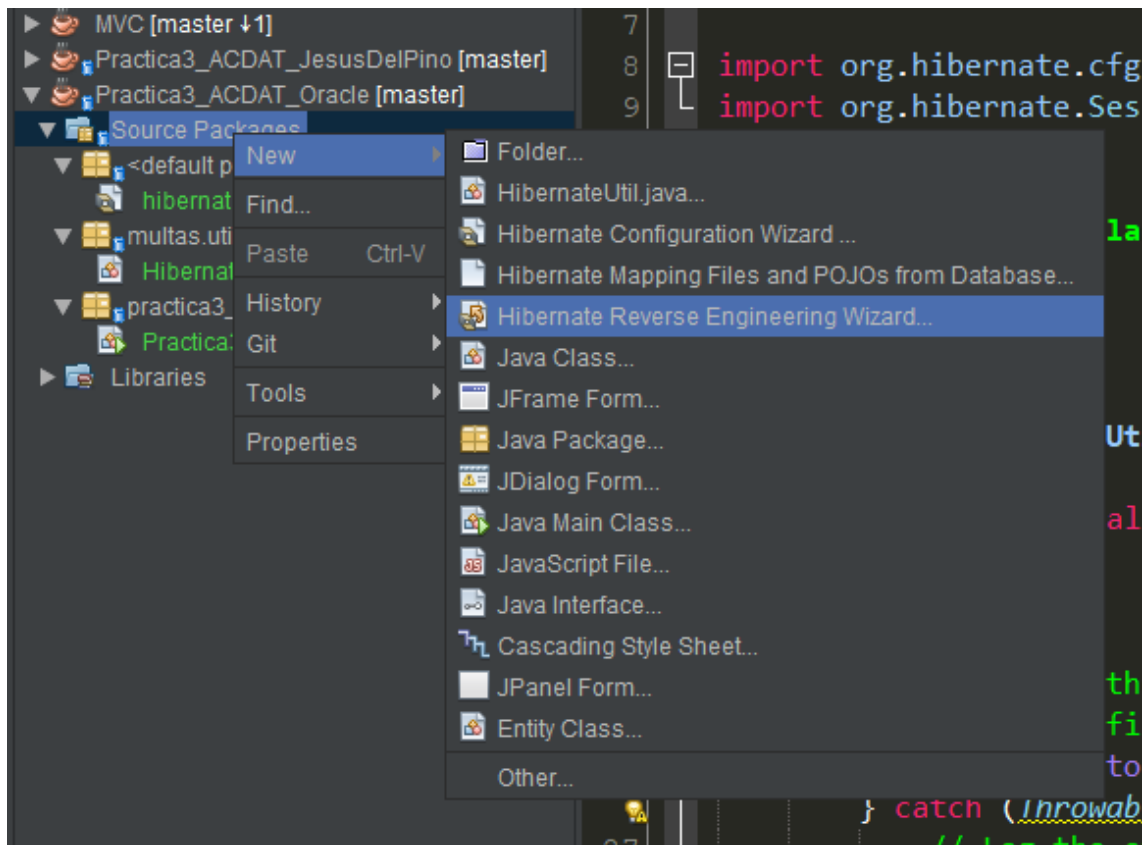
De nombre le pondremos HibernateUtil y en el paquete multas.util:



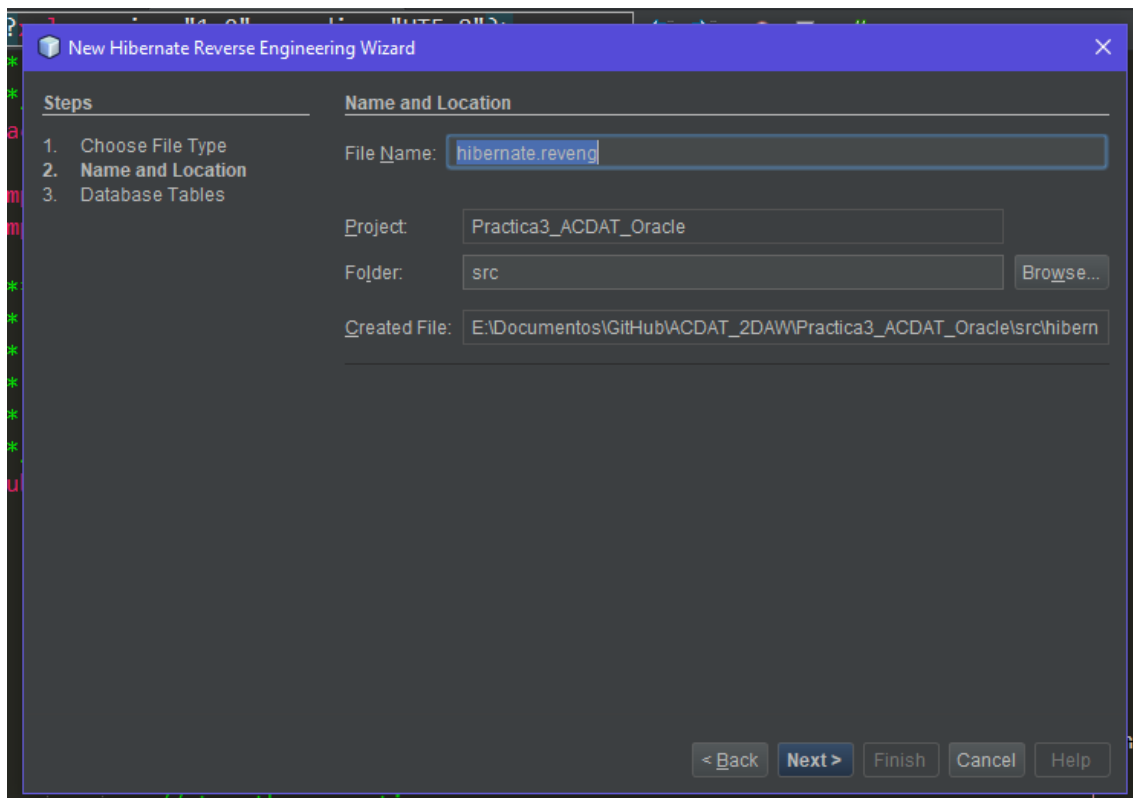
Le damos a Finish.



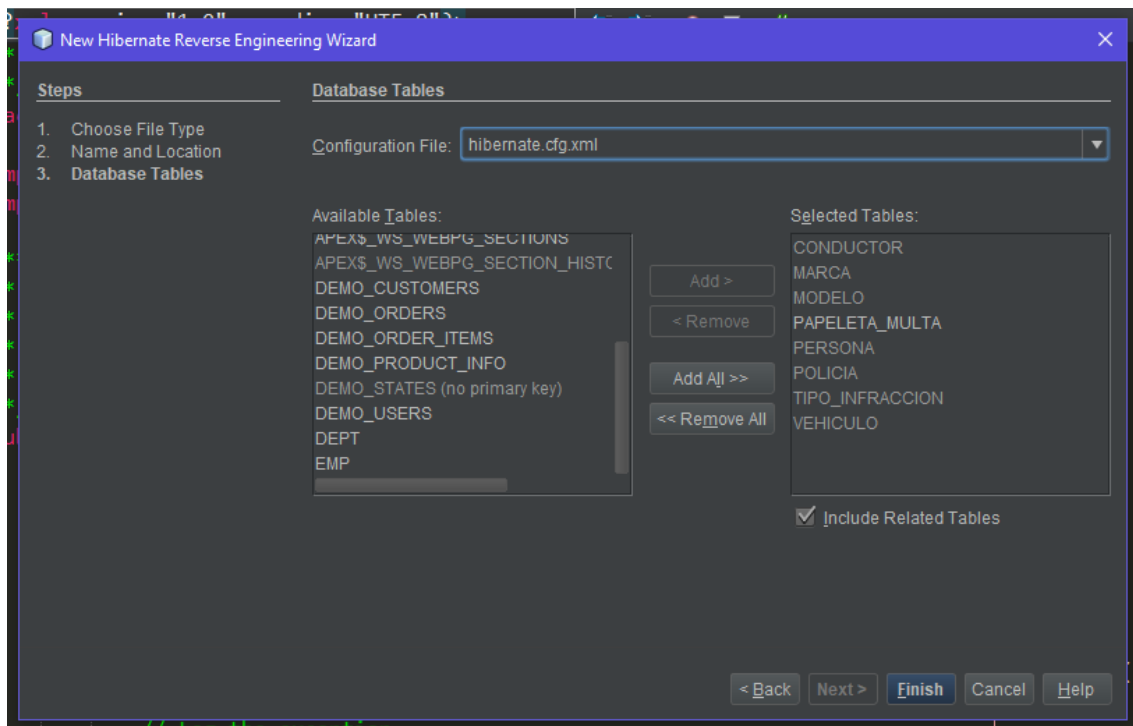
El siguiente paso será crear el archivo de ingeniería inversa. Esta opción la encontraremos haciendo click derecho sobre Source Packages y en Nuevo:



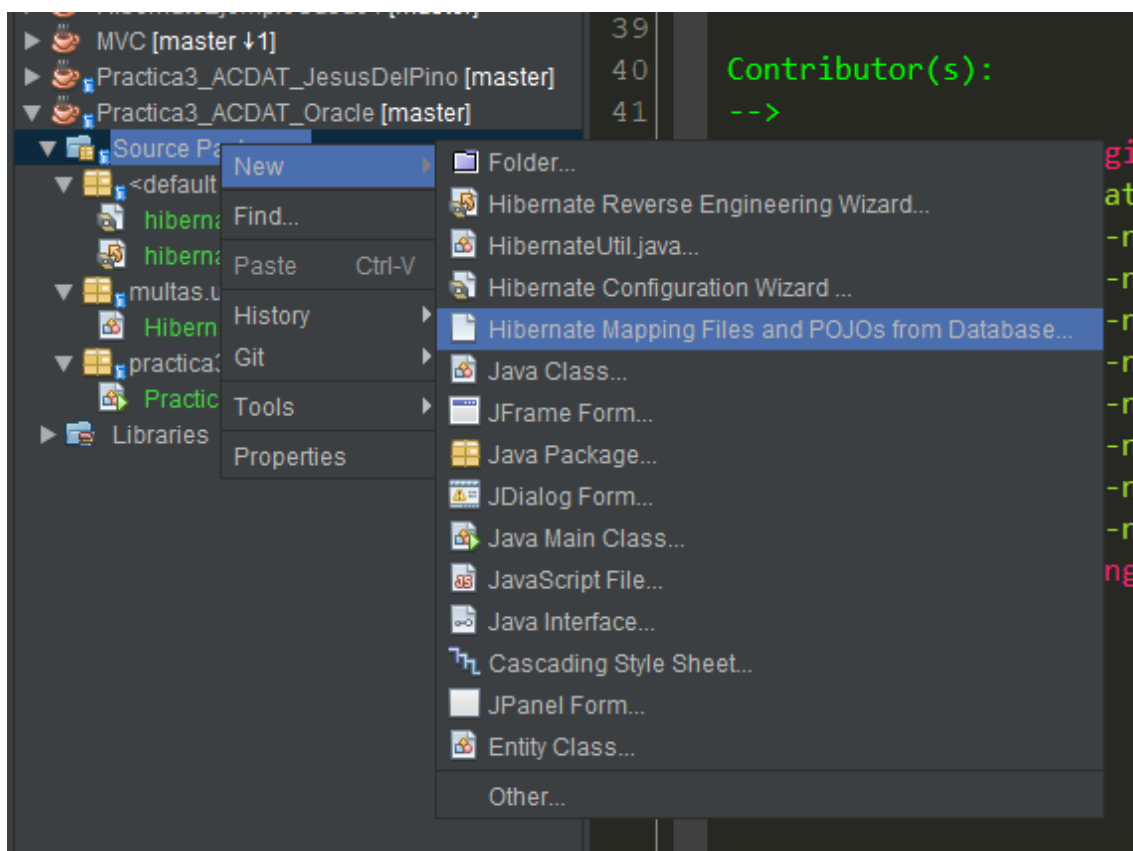
Dejamos todo por defecto:



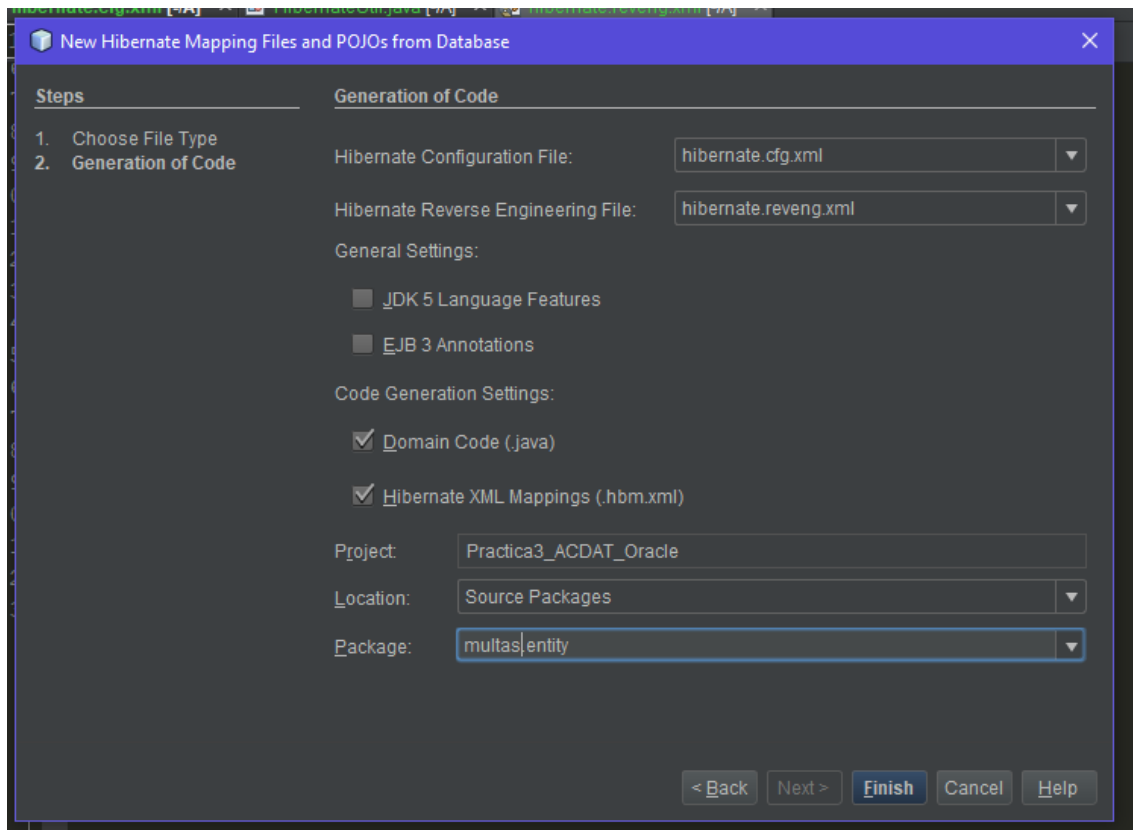
En el siguiente paso seleccionamos nuestras tablas y las añadimos:



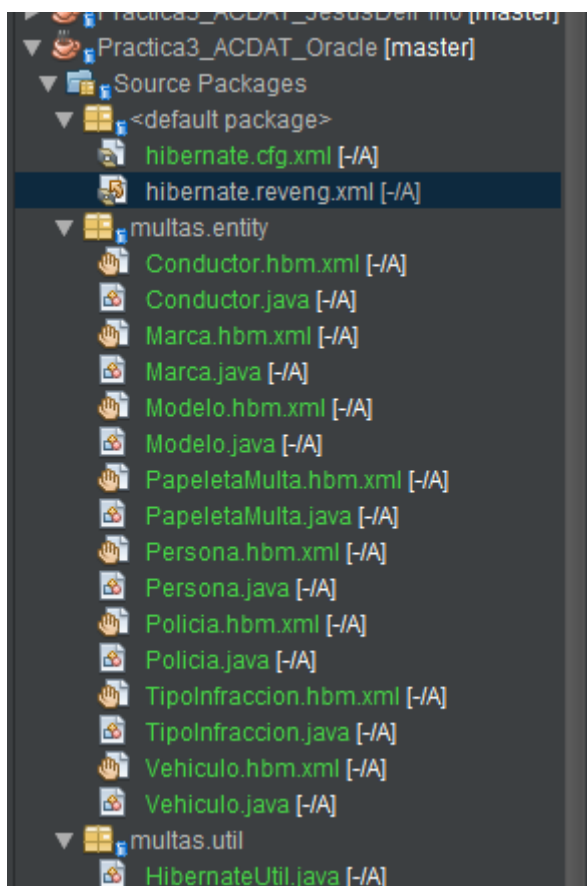
Por último, vamos a crear los archivos de mapeo hibernate y POJOs. Para ello, click derecho sobre Source Packages, Nuevo y seleccionamos la opción de Archivos de mapas de Hibernate y POJOs:



Dejamos todo por defecto excepto el nombre del paquete que lo pondremos como multas.entity y pulsamos en Finish:



Se nos genera todo:



Consultas

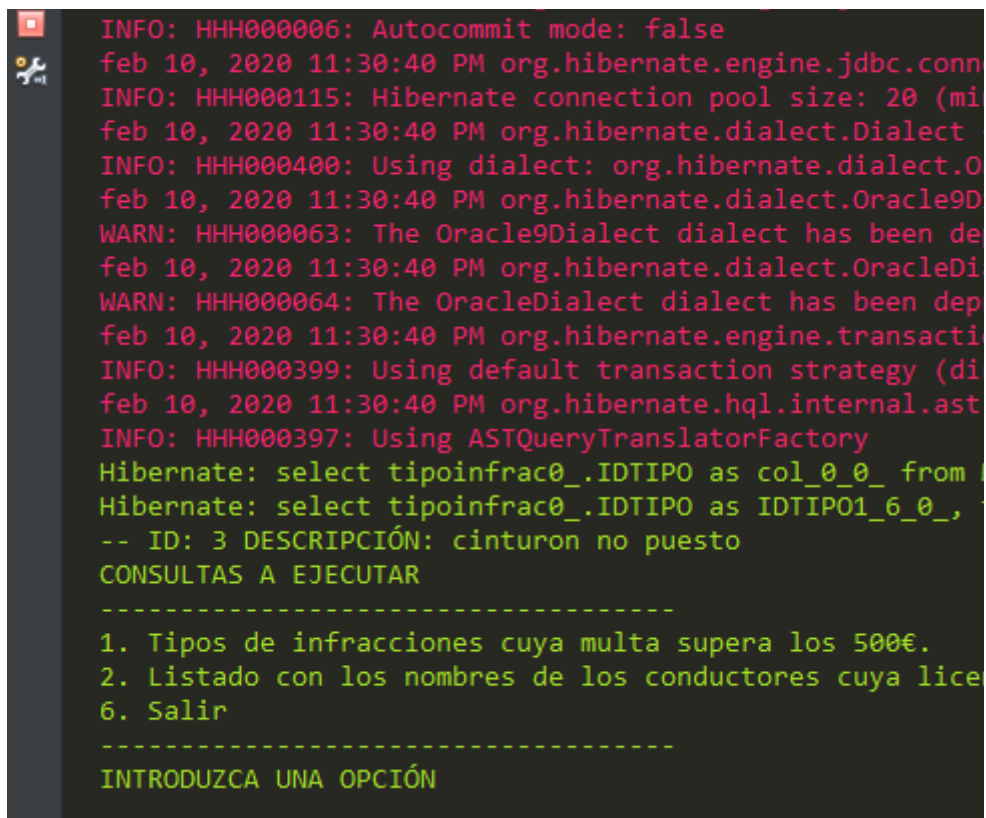
Consulta 1

En la primera consulta, obtendremos los tipos de infracciones cuya multa supera los 500€.

El código que hemos realizado para esta consulta es el siguiente:

```
TipoInfraccion inf = new TipoInfraccion();
SessionFactory sfactory = HibernateUtil.getSessionFactory();
Session session = sfactory.openSession();
Query q = session.createQuery("from TipoInfraccion where
importe>500");
Iterator<?> iter = q.iterate();
while (iter.hasNext()) {
    inf = (TipoInfraccion) iter.next();
    System.out.println("-- ID: "+inf.getIdtipo()+"
DESCRIPCIÓN: "+inf.getDescripcion());
}
```

Resultado obtenido tras la ejecución de la consulta:



```
INFO: HHH000006: Autocommit mode: false
feb 10, 2020 11:30:40 PM org.hibernate.engine.jdbc.conn
INFO: HHH000115: Hibernate connection pool size: 20 (mi
feb 10, 2020 11:30:40 PM org.hibernate.dialect.Dialect
INFO: HHH000400: Using dialect: org.hibernate.dialect.O
feb 10, 2020 11:30:40 PM org.hibernate.dialect.Oracle9D
WARN: HHH000063: The Oracle9Dialect dialect has been de
feb 10, 2020 11:30:40 PM org.hibernate.dialect.OracleDi
WARN: HHH000064: The OracleDialect dialect has been dep
feb 10, 2020 11:30:40 PM org.hibernate.engine.transacti
INFO: HHH000399: Using default transaction strategy (di
feb 10, 2020 11:30:40 PM org.hibernate.hql.internal.ast
INFO: HHH000397: Using ASTQueryTranslatorFactory
Hibernate: select tipoinfrac0_.IDTIPO as col_0_0_ from
Hibernate: select tipoinfrac0_.IDTIPO as IDTIPO1_6_0_,
-- ID: 3 DESCRIPCIÓN: cinturón no puesto
CONSULTAS A EJECUTAR
-----
1. Tipos de infracciones cuya multa supera los 500€.
2. Listado con los nombres de los conductores cuya lice
6. Salir
-----
INTRODUZCA UNA OPCIÓN
```

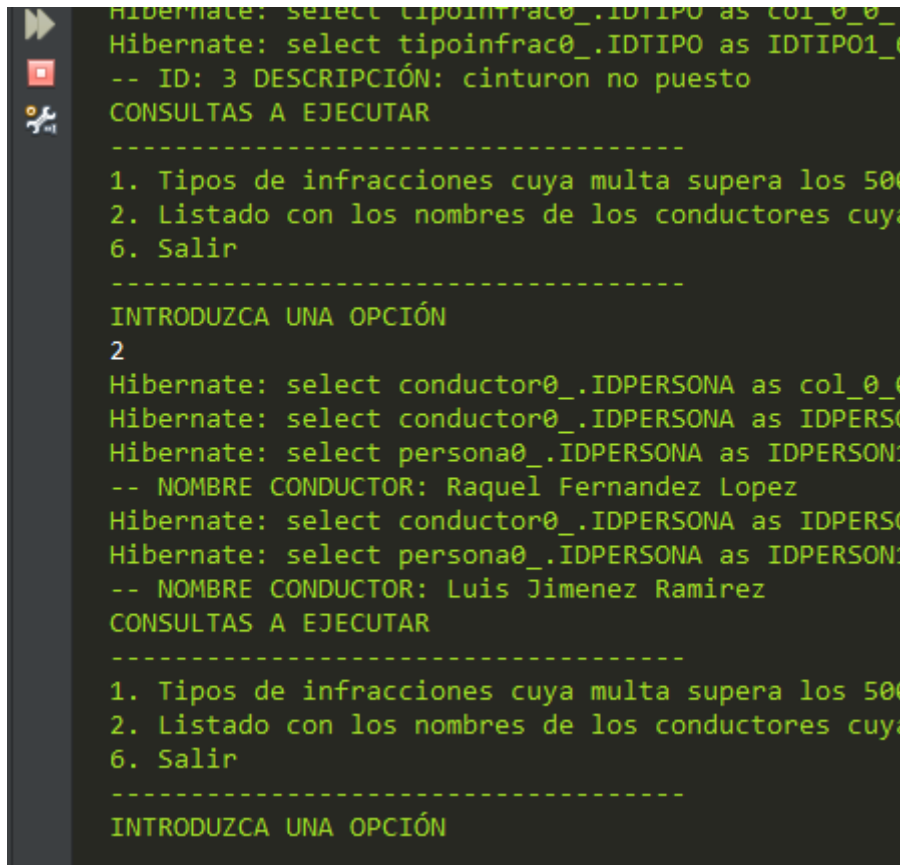
Consulta 2

En esta consulta listaremos los nombres de los conductores cuya licencia de conducir ya haya vencido.

El código que hemos realizado para esta consulta es el siguiente:

```
Conductor cond = new Conductor();
SessionFactory sfactory = HibernateUtil.getSessionFactory();
Session session = sfactory.openSession();
Query q = session.createQuery("from Conductor where
fechaExpiracion < sysdate()");
Iterator<?> iter = q.iterate();
while (iter.hasNext()) {
    cond = (Conductor) iter.next();
    System.out.println("-- NOMBRE CONDUCTOR:
"+cond.getPersona().getNombre());
}
```

Resultado obtenido tras la ejecución de la consulta:



```
Hibernate: select tipoinfrac0_.IDTIPO as col_0_0_
Hibernate: select tipoinfrac0_.IDTIPO as IDTIPO01_0
-- ID: 3 DESCRIPCIÓN: cinturon no puesto
CONSULTAS A EJECUTAR
-----
1. Tipos de infracciones cuya multa supera los 500
2. Listado con los nombres de los conductores cuya
6. Salir
-----
INTRODUZCA UNA OPCIÓN
2
Hibernate: select conductor0_.IDPERSONA as col_0_0_
Hibernate: select conductor0_.IDPERSONA as IDPERSONA01_0
Hibernate: select persona0_.IDPERSONA as IDPERSONA02_0
-- NOMBRE CONDUCTOR: Raquel Fernandez Lopez
Hibernate: select conductor0_.IDPERSONA as IDPERSONA03_0
Hibernate: select persona0_.IDPERSONA as IDPERSONA04_0
-- NOMBRE CONDUCTOR: Luis Jimenez Ramirez
CONSULTAS A EJECUTAR
-----
1. Tipos de infracciones cuya multa supera los 500
2. Listado con los nombres de los conductores cuya
6. Salir
-----
INTRODUZCA UNA OPCIÓN
```

- **¿Es posible usar Hibernate para el mapeo? Justifica la respuesta. Si no lo es ¿existe alguna otra herramienta que si lo permita para sistemas gestores como oracle? Indica cuál y describe su funcionamiento y principales características.**

Si, se puede, Hibernate es una herramienta de mapeo objeto-relacional (ORM) bajo licencia GNU LGPL para Java, que facilita el mapeo de atributos en una base de datos tradicional, y el modelo de objetos de una aplicación mediante archivos declarativos o anotaciones en los beans de las entidades que permiten establecer estas relaciones. Resumiendo, agiliza la relación entre la aplicación y nuestra base de datos SQL (Oracle, entre otros).

- **Si no es posible el mapeo justificar la razón y responde a la siguiente pregunta según tu criterio.**

Si, es posible hacer el mapeo.

- **¿Tiene sentido que una aplicación Java no pueda acceder a los mismos datos almacenados en diferentes sistemas gestores de bases datos?**

No tiene sentido, porque en una aplicación java no creo que sea recomendable utilizar distintos métodos para acceder a la base de datos, puesto que, se estaría desperdiciando código y no se reutilizaría. A demás puede ocasionar problemas a hora de trabajar con los datos de la base de datos.

Anexo

MySQL

En el main lo que hemos hecho es llamar a nuestro método menú:

```
public static void main(String[] args) {
    // TODO code application logic here
    menu();
}
```

En el método menú lo que hacemos es inicializar la variables option que será la que iremos modificando cuando volvamos a listar las opciones de nuestro menú, aquí podemos elegir del 1 al 5 y el número 6 sirve para salir del programa. Según el número seleccionado hará la consulta indicada en la lista anteriormente mostrada, en caso de seleccionar un número que no esté comprendido entre el 1 y el 6 dará error y tendremos la posibilidad de poder elegir otro número.

```
public static void menu() {
    int option = 0;
    Scanner teclado = new Scanner(System.in);
    do {
        System.out.println("CONSULTAS A EJECUTAR");

        System.out.println("-----");
        System.out.println("1. Tipos de infracciones cuya multa supera los 500€.");
        System.out.println("2. Listado con los nombres de los conductores cuya licencia de conducir ya ha vencido.");
        System.out.println("3. Edad media de los conductores que tienen infracciones por 'Exceso de velocidad'.");
        System.out.println("4. Nombre de los conductores que han cometido una infracción cuyo importe sea inferior al\n"
            + "del importe de algunas de las infracciones cometidas en el día de ayer.");
        System.out.println("5. Listado con el número de placa de todos aquellos policías que hayan sancionado todos los tipos\n"
            + "de infracciones.");
        System.out.println("6. Salir");

        System.out.println("-----");
        System.out.println("INTRODUZCA UNA OPCIÓN");
        option = teclado.nextInt();
        if(option >=1 && option <=5) {

            switch(option) {
                case 1:
                    Consultal();
                    break;
                case 2:
```

```

        Consulta2();
        break;
    case 3:
        Consulta3();
        break;
    case 4:
        Consulta4();
        break;
    case 5:
        break;
    }

    } else if(option == 6){
        System.out.println("Hasta luego!!");
        System.exit(0);
    } else {
        System.out.println("Opción no válida!!");
    }
} while (option != 6);
}

```

En el método Consulta1 lo que hacemos es crear el objeto TipoInfraccion, iniciamos la sesión, escribimos la consulta y la ejecutamos. En el iterator guardamos los datos de las filas obtenidas tras realizar la consulta. Con el while recorremos el iterator y devolvemos los datos que han sido pedidos.

```

public static void Consulta1() {
    TipoInfraccion inf = new TipoInfraccion();
    SessionFactory sfactory = HibernateUtil.getSessionFactory();
    Session session = sfactory.openSession();
    Query q = session.createQuery("from TipoInfraccion where
importe>500");
    Iterator<?> iter = q.iterate();
    while (iter.hasNext()) {
        inf = (TipoInfraccion) iter.next();
        System.out.println("-- ID: "+inf.getIdtipo()+"
DESCRIPCIÓN: "+inf.getDescripcion());
    }
}

```

En este método inicializamos el objeto Conductor, iniciamos la sesión con la base de datos y realizamos la consulta. Realiza lo mismo que en el método anterior sacando los datos pertinentes de esta.

```

public static void Consulta2() {
    Conductor cond = new Conductor();
    SessionFactory sfactory = HibernateUtil.getSessionFactory();
    Session session = sfactory.openSession();
    Query q = session.createQuery("from Conductor where
fechaExpiracion < now()");
    Iterator<?> iter = q.iterate();
    while (iter.hasNext()) {
        cond = (Conductor) iter.next();
    }
}

```

```

        System.out.println("-- NOMBRE CONDUCTOR:
"+cond.getPersona().getNombre());
    }
}

```

En esta consulta como pedimos directamente los datos desde el select no necesitamos utilizar el iterator, simplemente cuando realizamos la consulta guardamos el resultado único obtenido en nuestra variable de tipo double, luego la redondeamos y finalmente la mostramos por pantalla.

```

public static void Consulta3() {
    SessionFactory sfactory = HibernateUtil.getSessionFactory();
    Session session = sfactory.openSession();
    Query q = session.createQuery("select avg(datediff (now(),
fechaNacimiento)/365) from Persona where idpersona in (select
idpersona from Conductor where idpersona in (select conductor from
PapeletaMulta where idtipoinfraccion in (select idtipo from
TipoInfraccion where descripcion='exceso de velocidad'))"));

    Double result = (Double) q.uniqueResult();
    Long resultF = Math.round(result);
    System.out.println("-- EDAD MEDIA: "+resultF);
}

```

En la consulta 4 utilizaremos la clase persona para guardar los datos del objeto obtenido tras la consulta, finalmente muestra el resultado obtenido en la consulta.

```

public static void Consulta4() {
    Persona pers = new Persona();
    SessionFactory sfactory = HibernateUtil.getSessionFactory();
    Session session = sfactory.openSession();
    Query q = session.createQuery("from Persona as per where
idpersona in (select conductor from PapeletaMulta where tipoInfraccion
in (select idtipo from TipoInfraccion where importe < (select
max(importe) from TipoInfraccion where idtipo in (select
tipoInfraccion from PapeletaMulta where datediff(now(),
fecha)=1) )))");
    Iterator<?> iter = q.iterate();
    while (iter.hasNext()) {
        pers = (Persona) iter.next();
        System.out.println("-- NOMBRE: "+pers.getNombre());
    }
}

```

ORACLE

```
public static void main(String[] args) {
    menu();
}
```

En el main lo que hacemos es llamar al método menú-

```
public static void menu() {
    int option = 0;
    Scanner teclado = new Scanner(System.in);
    do {
        System.out.println("CONSULTAS A EJECUTAR");

        System.out.println("-----");
        System.out.println("1. Tipos de infracciones cuya multa
supera los 500€.");
        System.out.println("2. Listado con los nombres de los
conductores cuya licencia de conducir ya ha vencido.");
        System.out.println("6. Salir");

        System.out.println("-----");
        System.out.println("INTRODUZCA UNA OPCIÓN");
        option = teclado.nextInt();
        if(option >=1 && option <=2) {

            switch(option) {
                case 1:
                    Consulta1();
                    break;
                case 2:
                    Consulta2();
                    break;
            }

        } else if(option == 6){
            System.out.println("Hasta luego!!");
            System.exit(0);
        } else {
            System.out.println("Opción no válida!!");
        }
    } while (option != 6);
}
```

El método menú funciona de igual forma que el de MySQL solo que en vez de tener 6 posibles elecciones este tiene 3.

```
public static void Consulta1() {
    TipoInfraccion inf = new TipoInfraccion();
    SessionFactory sfactory = HibernateUtil.getSessionFactory();
    Session session = sfactory.openSession();
```



```

        Query q = session.createQuery("from TipoInfraccion where
importe>500");
        Iterator<?> iter = q.iterate();
        while (iter.hasNext()) {
            inf = (TipoInfraccion) iter.next();
            System.out.println("-- ID: "+inf.getIdtipo()+"
DESCRIPCIÓN: "+inf.getDescripcion());
        }
    }
}

```

En la consulta 1 lo que hacemos es crear un objeto de la clase TipoInfraccion para poder inicializarlo después con los objetos obtenidos en la consulta guardados en el iterator y poder utilizar sus métodos a la hora de mostrar los datos de la consulta.

```

public static void Consulta2() {
    Conductor cond = new Conductor();
    SessionFactory sfactory = HibernateUtil.getSessionFactory();
    Session session = sfactory.openSession();
    Query q = session.createQuery("from Conductor where
fechaExpiracion < sysdate()");
    Iterator<?> iter = q.iterate();
    while (iter.hasNext()) {
        cond = (Conductor) iter.next();
        System.out.println("-- NOMBRE CONDUCTOR:
"+cond.getPersona().getNombre());
    }
}

```

En esta última consulta lo que hacemos es usar la clase Conductor para sacar los datos tras la consulta.