

Mapea las tablas utilizando Hibernate con NetBeans y realiza un proyecto Java llamado HibernateOracle que obtenga lo siguiente:

1. Crea la base de datos.
2. Configura y crea la ORM Hibernate.
3. Realiza una inserción y un borrado sobre la tabla EMP.
4. Obtener un listado sobre las tablas EMP y DEPT que visualice empno, ename, sal, dname y loc.
5. Redactar un documento donde se explique el proceso seguido para la realización de la práctica.

Instalación de la base de datos NetBeans: 1 punto.

Configuración de Hibernate: 2 puntos.

Sentencias DML: 3 puntos.

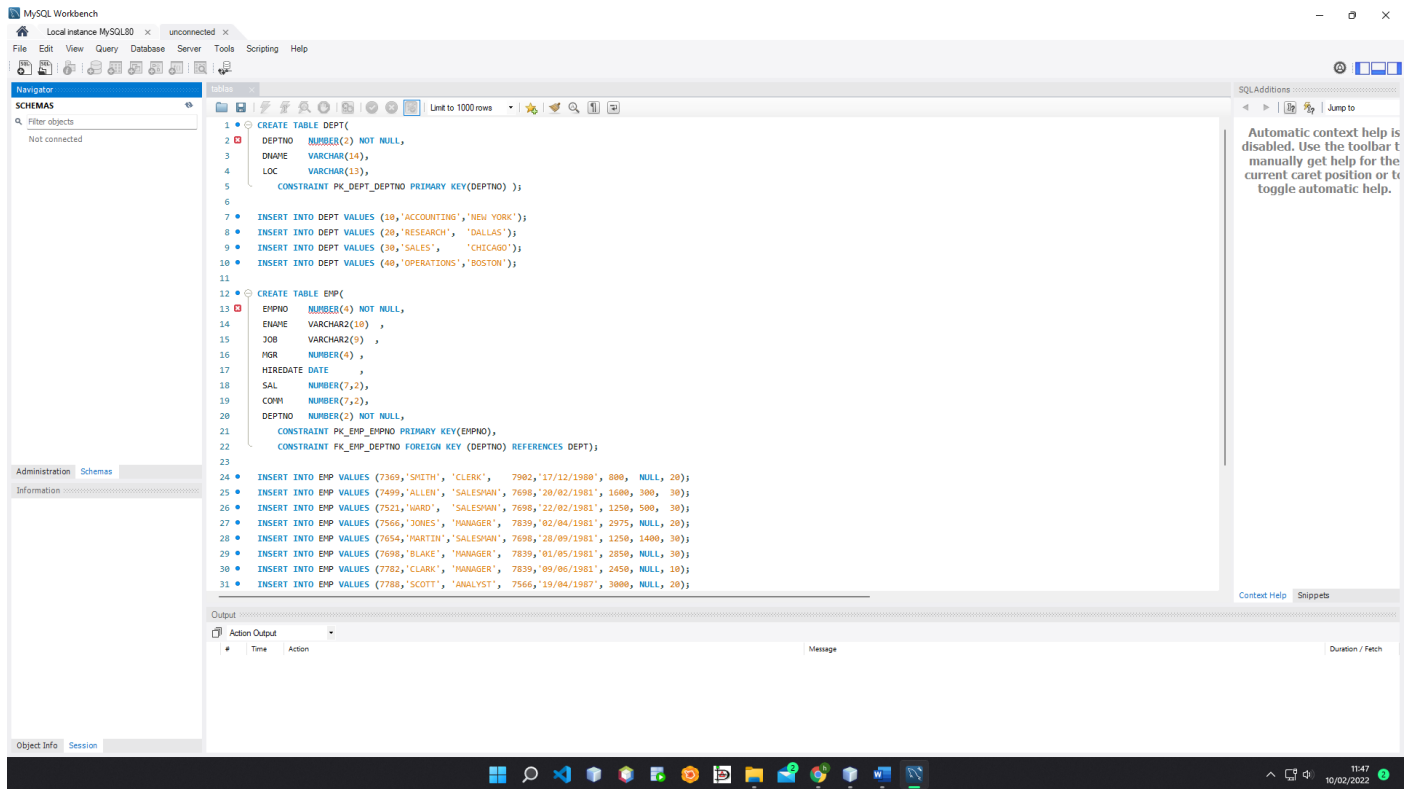
Realización de la join: 3 puntos.

Explicación de la tarea: 1 punto.

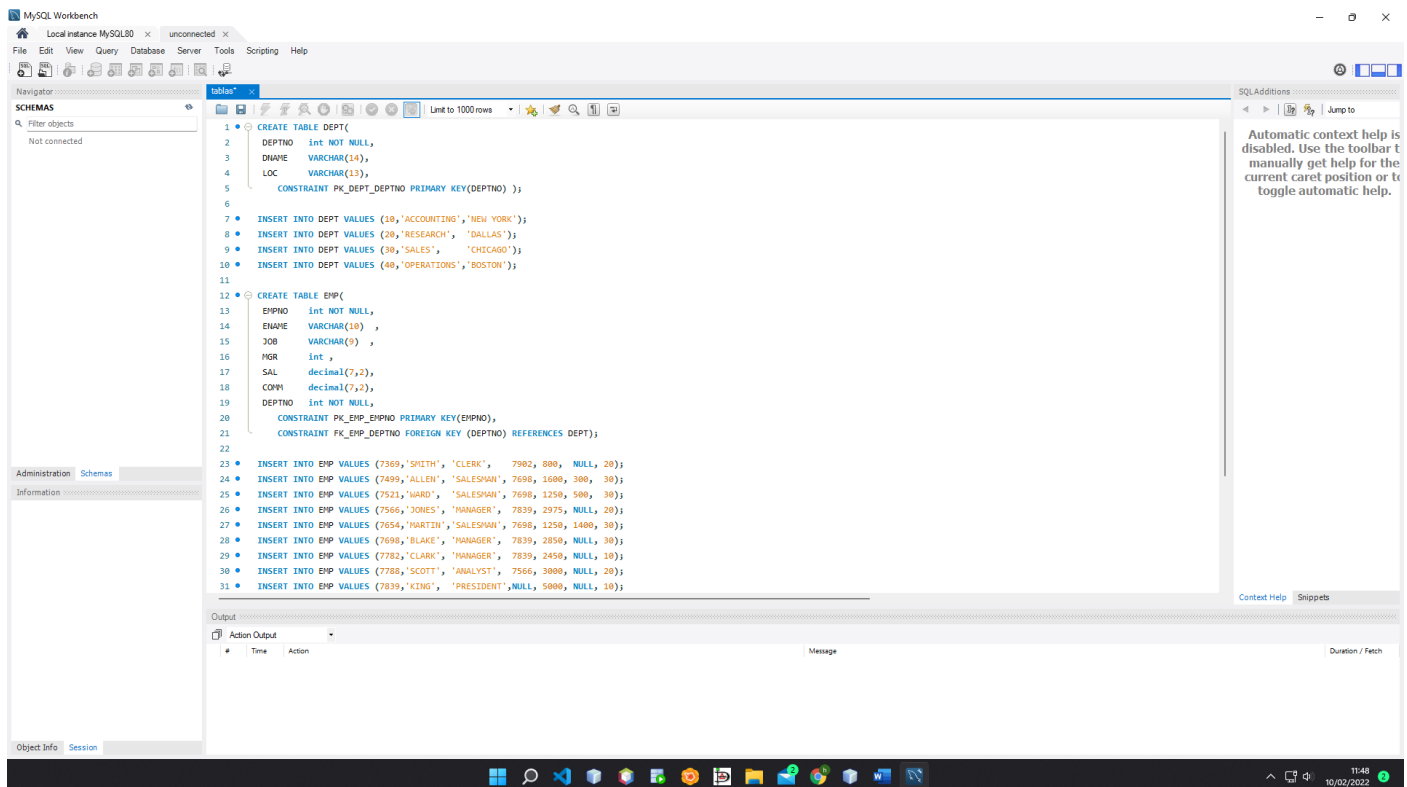
*Antes de empezar con la documentación, quiero aclarar que tuve muchos problemas durante el proceso de los cuales no tengo capturas, así que he usado otro proyecto a modo de ilustración para simular el proceso que hice para poder realizar la practica satisfactoriamente.

Lo primero que hice fue llevarme la tabla a Oracle, pero me daba error Netbeans al conectar con la base de datos, errores que no pude capturar, pero desde problemas con las versiones de las contraseñas hasta problemas con las versiones del ojdbc. Probe en Netbeans versiones 12, 12.6, 8.2 y 8.0.2 y en cada uno me salían distintos errores.

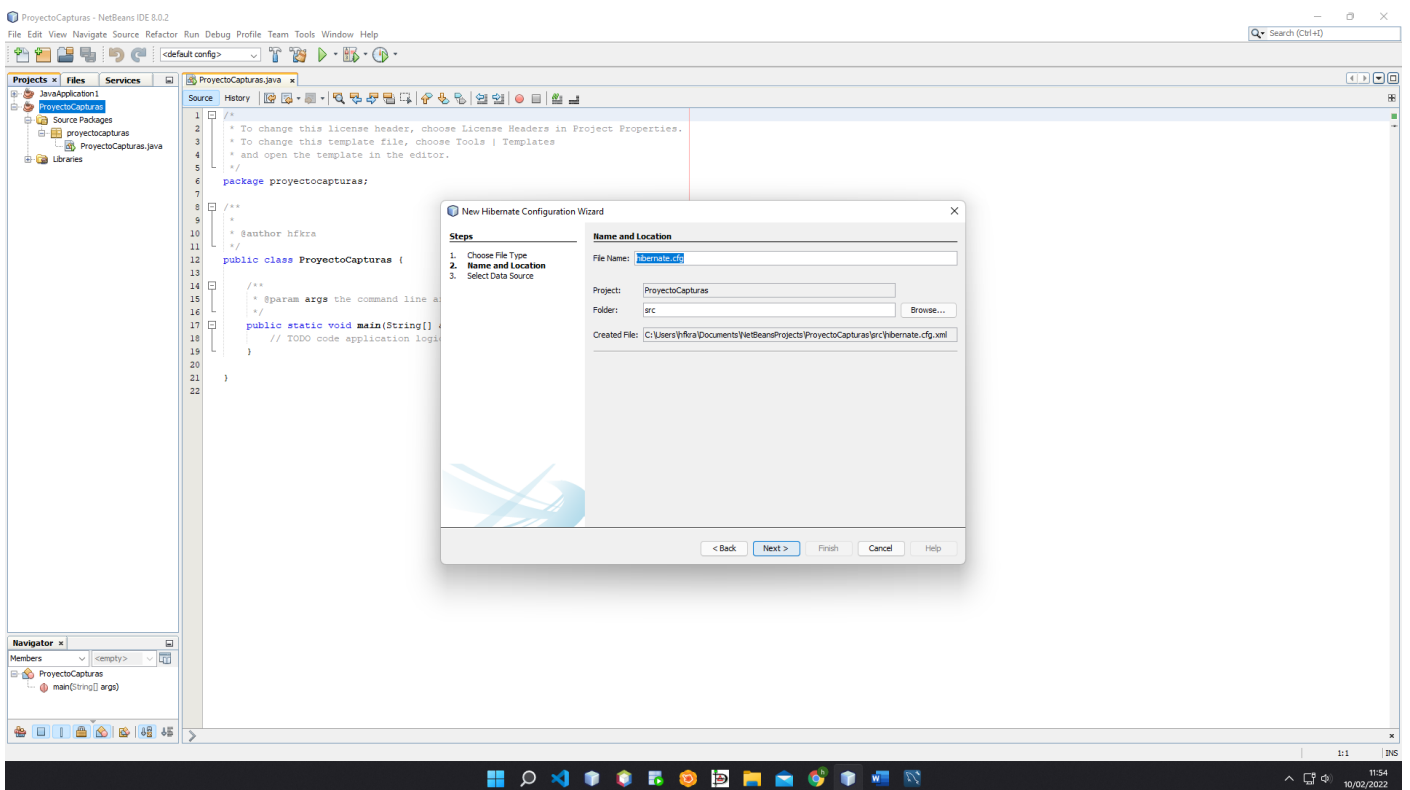
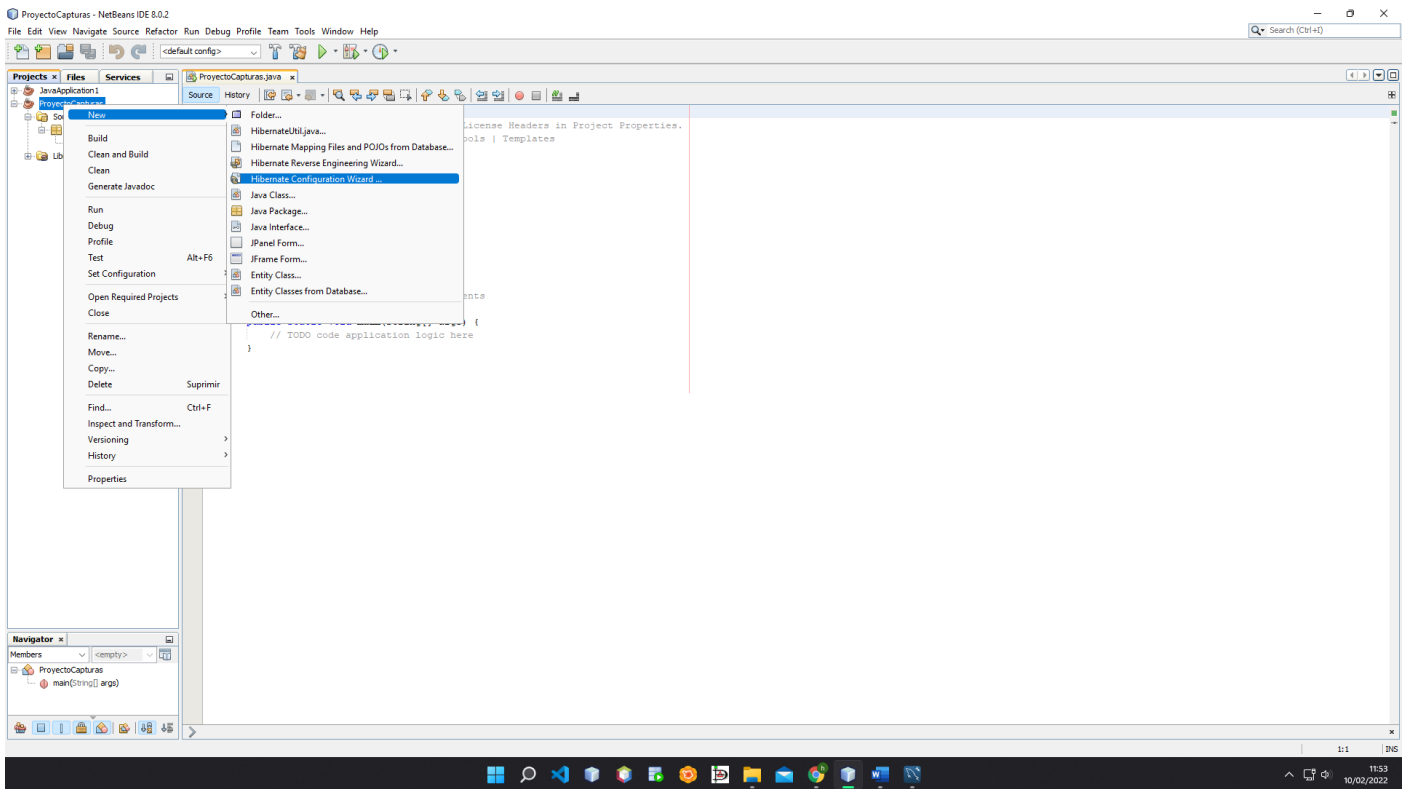
Ante la imposibilidad de hacerlo funcionar decidí probar MySQL, aquí esta la base de datos suministrada.



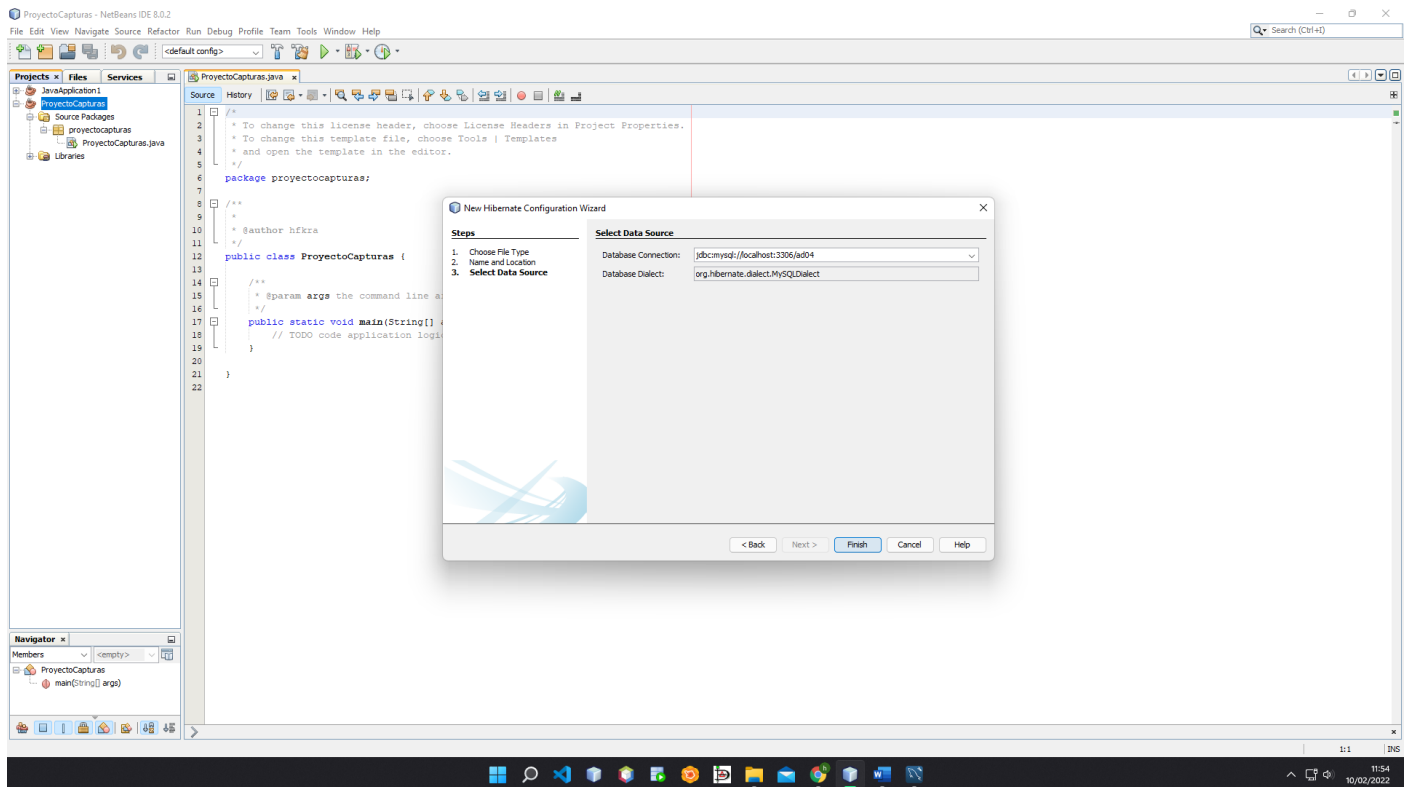
*Tuve que hacer ciertas modificaciones para poder crearla: cambiar NUMBER por INT, VARCHAR2 por VARCHAR, NUMBER por DECIMAL y tuve que borrar el campo DATE de la tabla EMP y de los INSERT (me daba error al guardar los datos, probé varios formatos 'dd-MM-yyyy','dd-MM-yyyy hh:mm:ss','dd/MM/yyyy','dd/MM/yyyy hh:mm:ss' pero no conseguí nada)



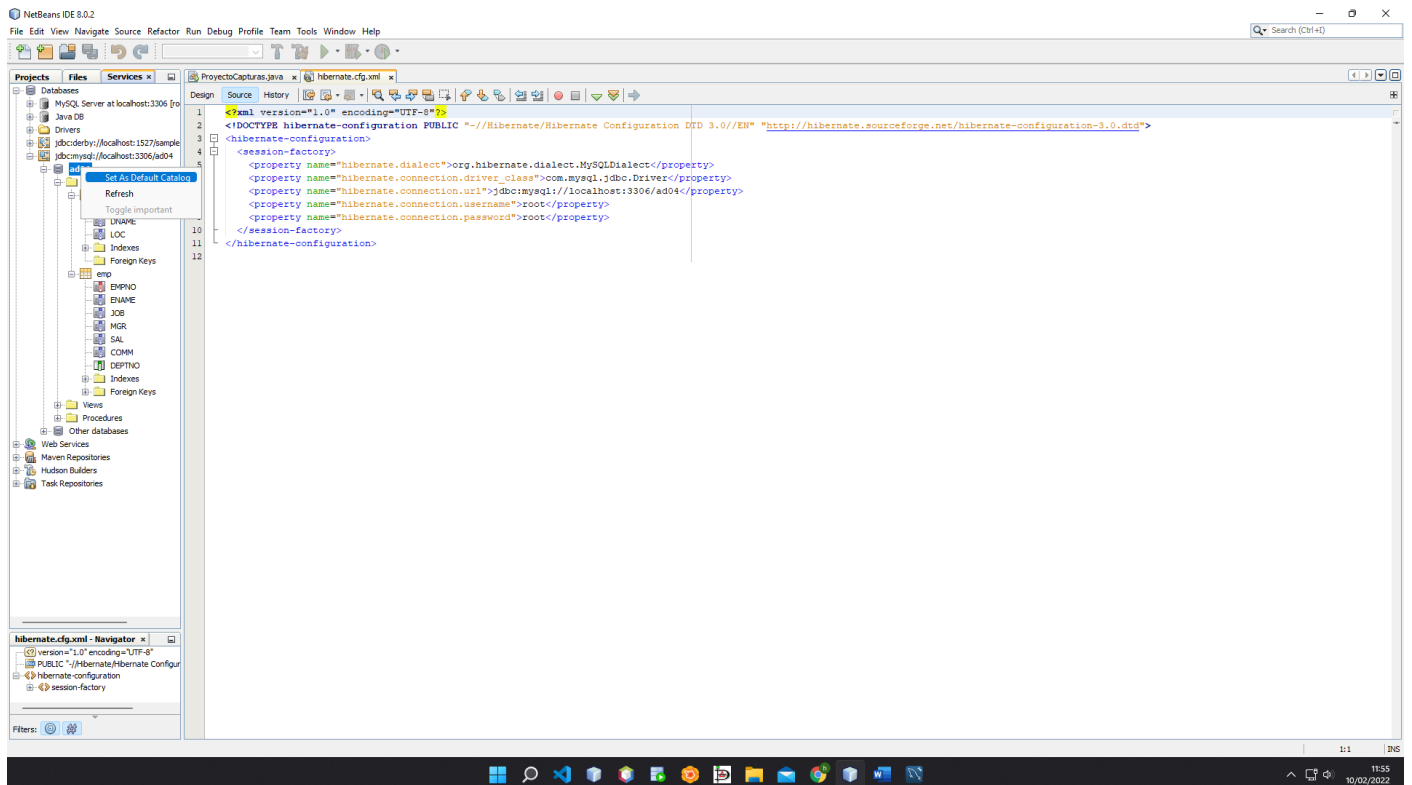
Luego, ya en Netbeans, cree un proyecto y me dispuse a hacer la conexión con MySQL creando el hibernate.cfg.xml



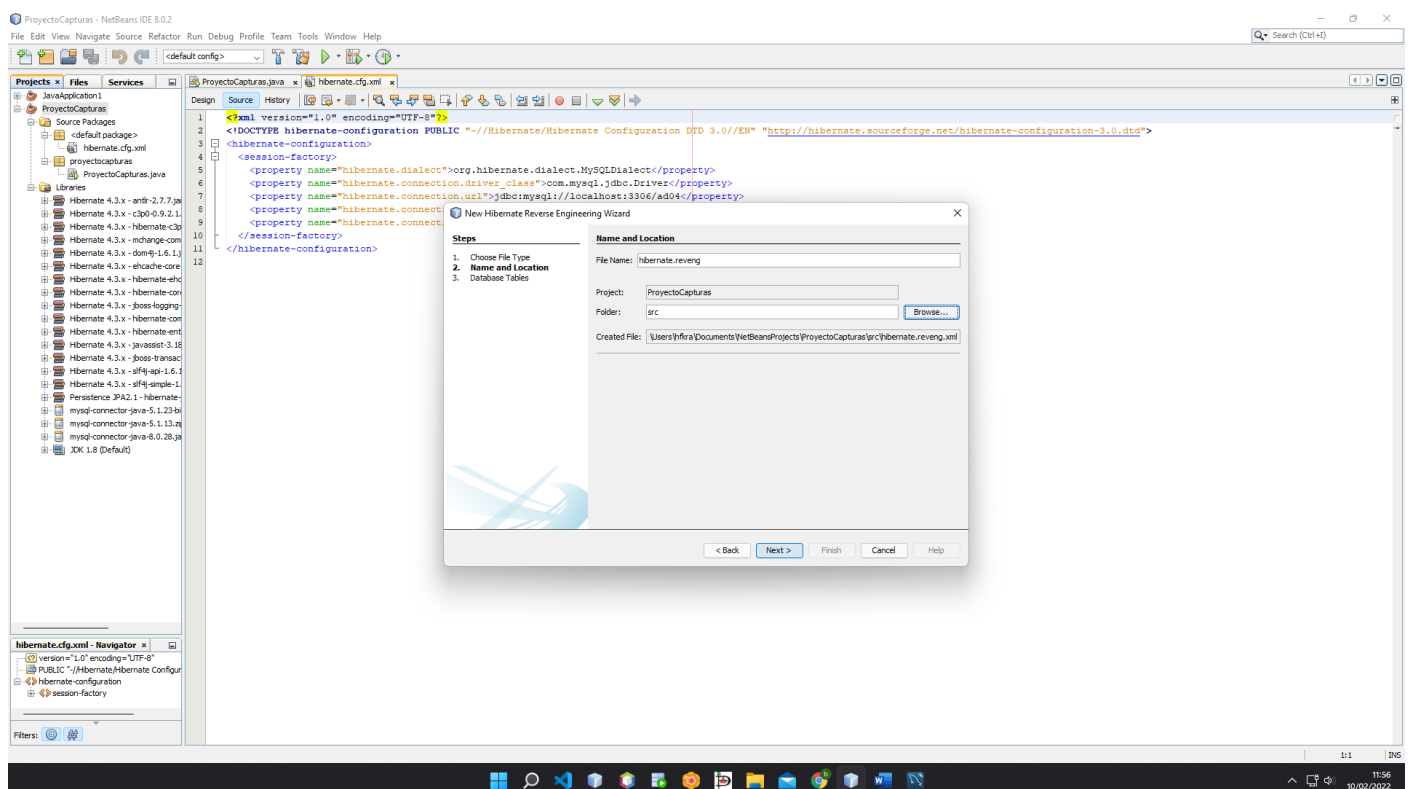
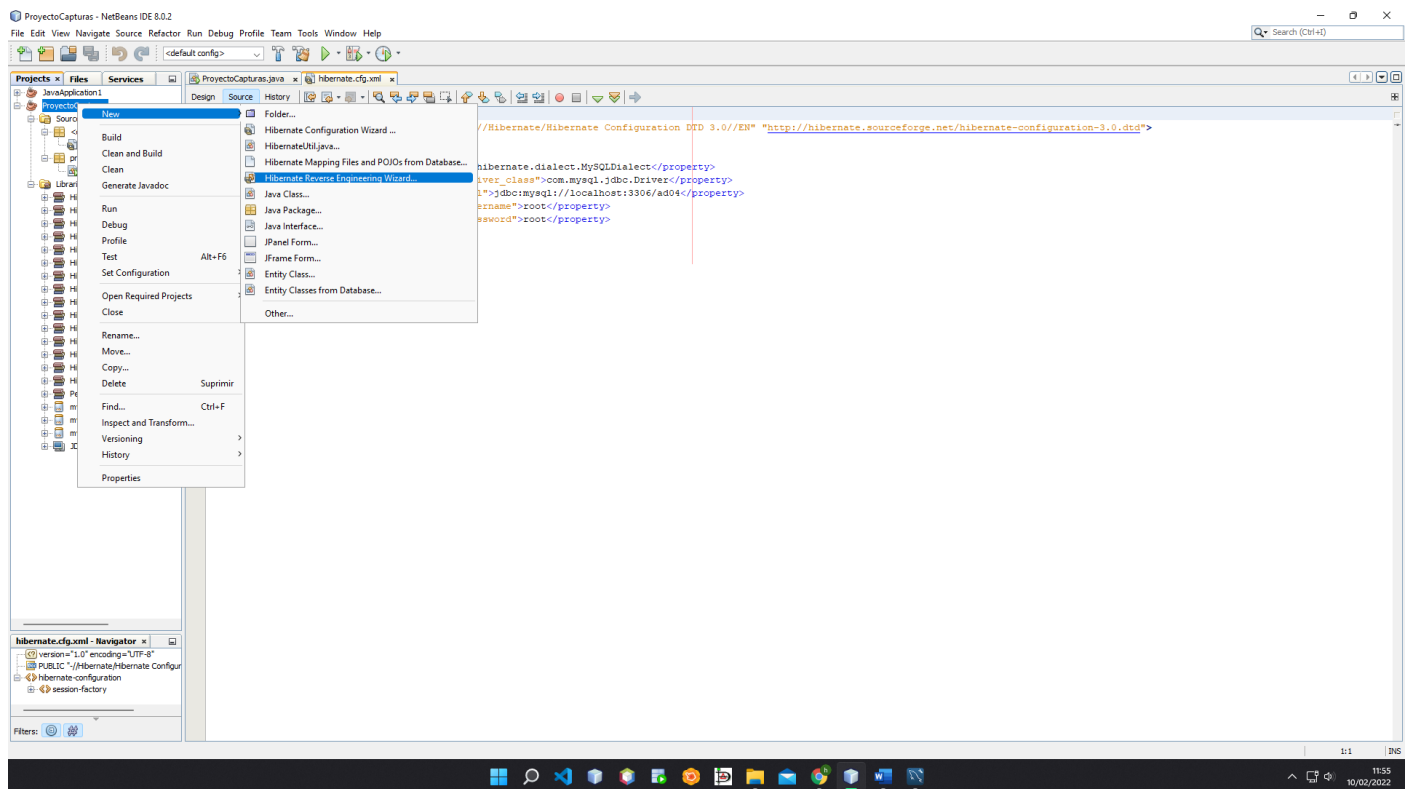
Despues de este apartado, viene la selección del driver para conectar, probe el 5.1.13 y el 8.0 con el que al final me estableció la conexión. Aunque, en la siguiente captura, se ve como utilizo la conexión que ya había hecho antes (la captura es posterior, por ilustrar el proceso).



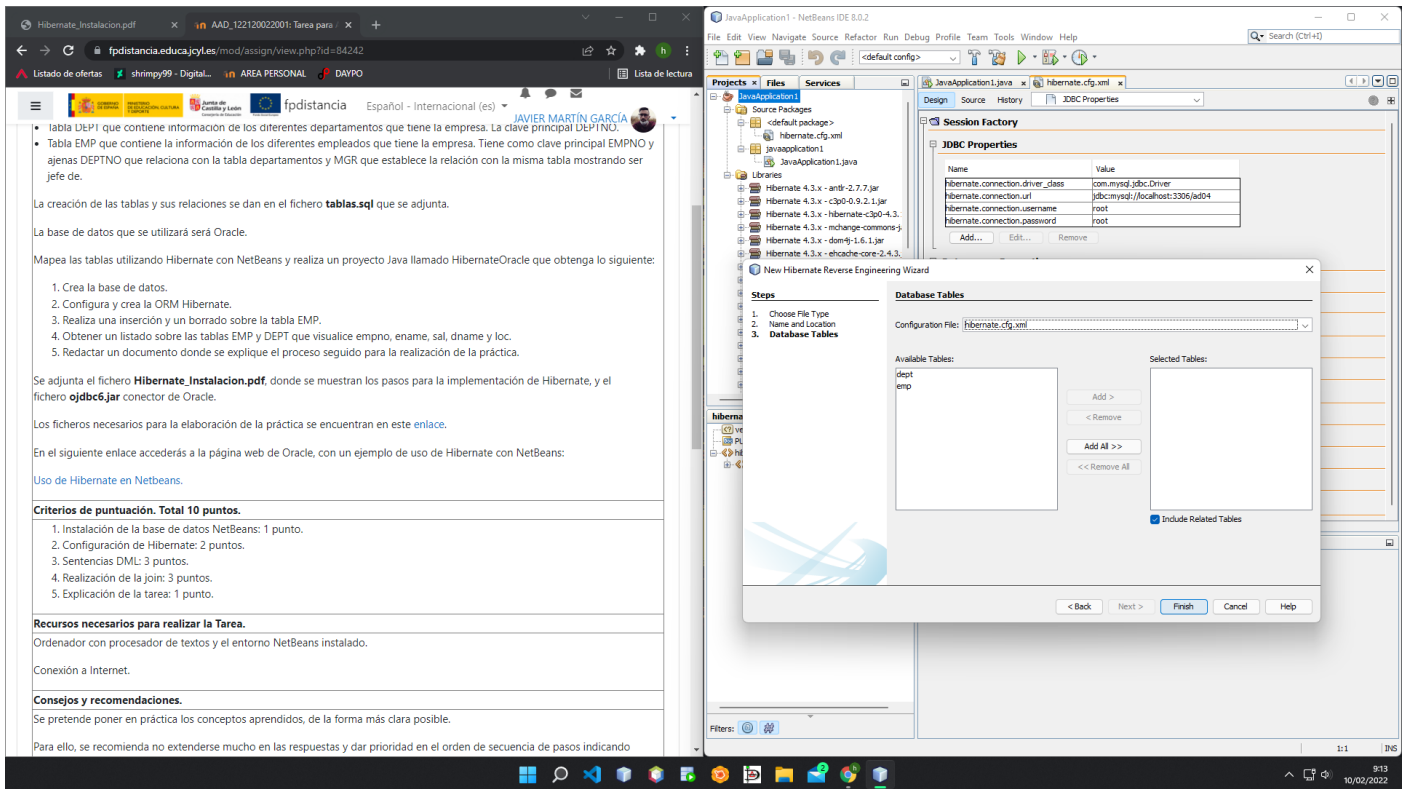
Con la conexión establecida, y al intentar crear el reverse engineering, me daban errores desde que no me aparecía en el paquete el hibernate.cfg, errores de que no eran validos los parametros e incluso probé a construirlo de nuevo. Despues de todo, lo que me funcionó fue lo siguiente, click derecho doble la base de datos y 'Set as Default Catalog', a partir de ahí, ya pude resolver la practica a buen ritmo.



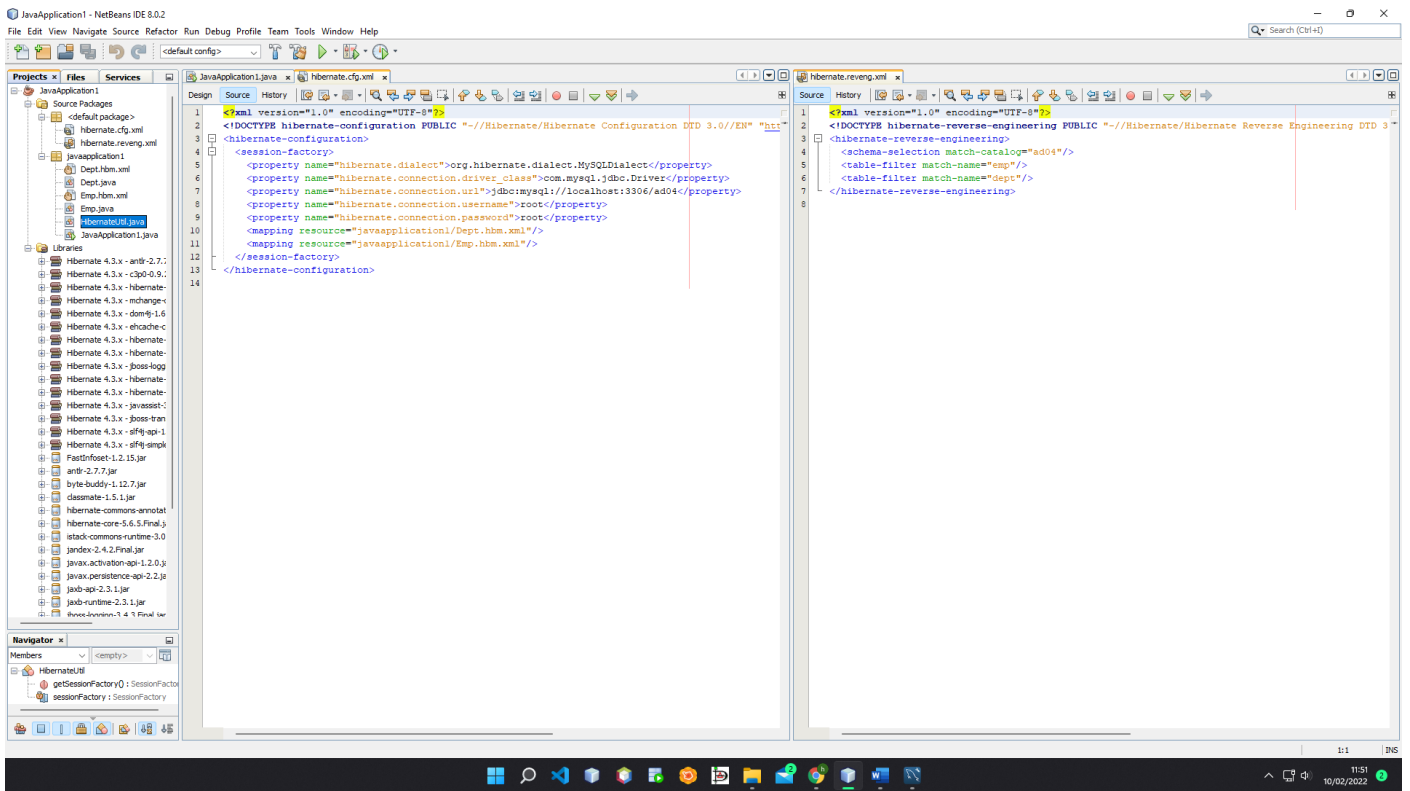
Despues de tener el hibernate.config, procedemos a crear el reverse engineering.



Cuando carga, el sistema nos reconoce correctamente las tablas de nuestra base de datos y nos las muestra para poder crearlo.



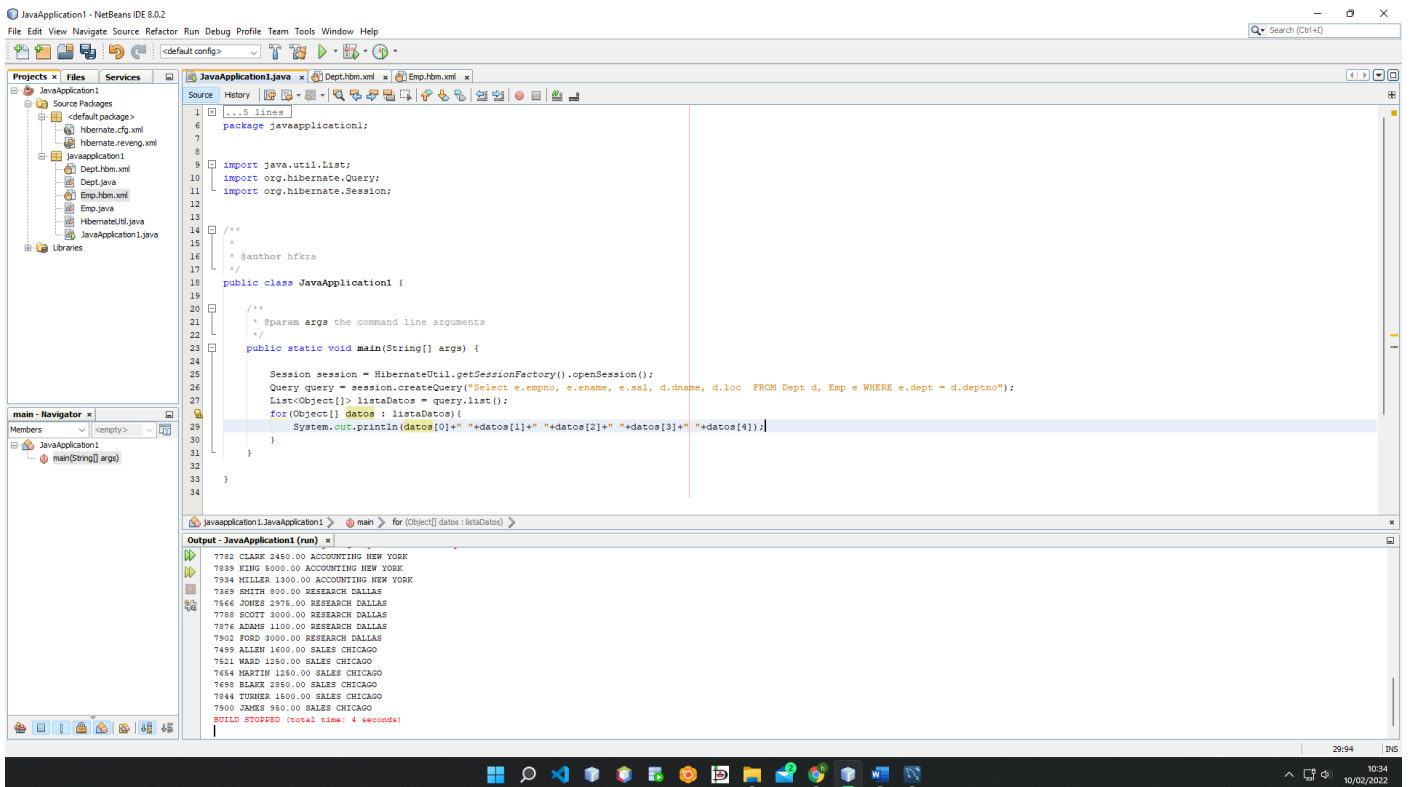
Aquí una muestra de ambos archivos abiertos. Se pueden apreciar en el proyecto los POJOS.



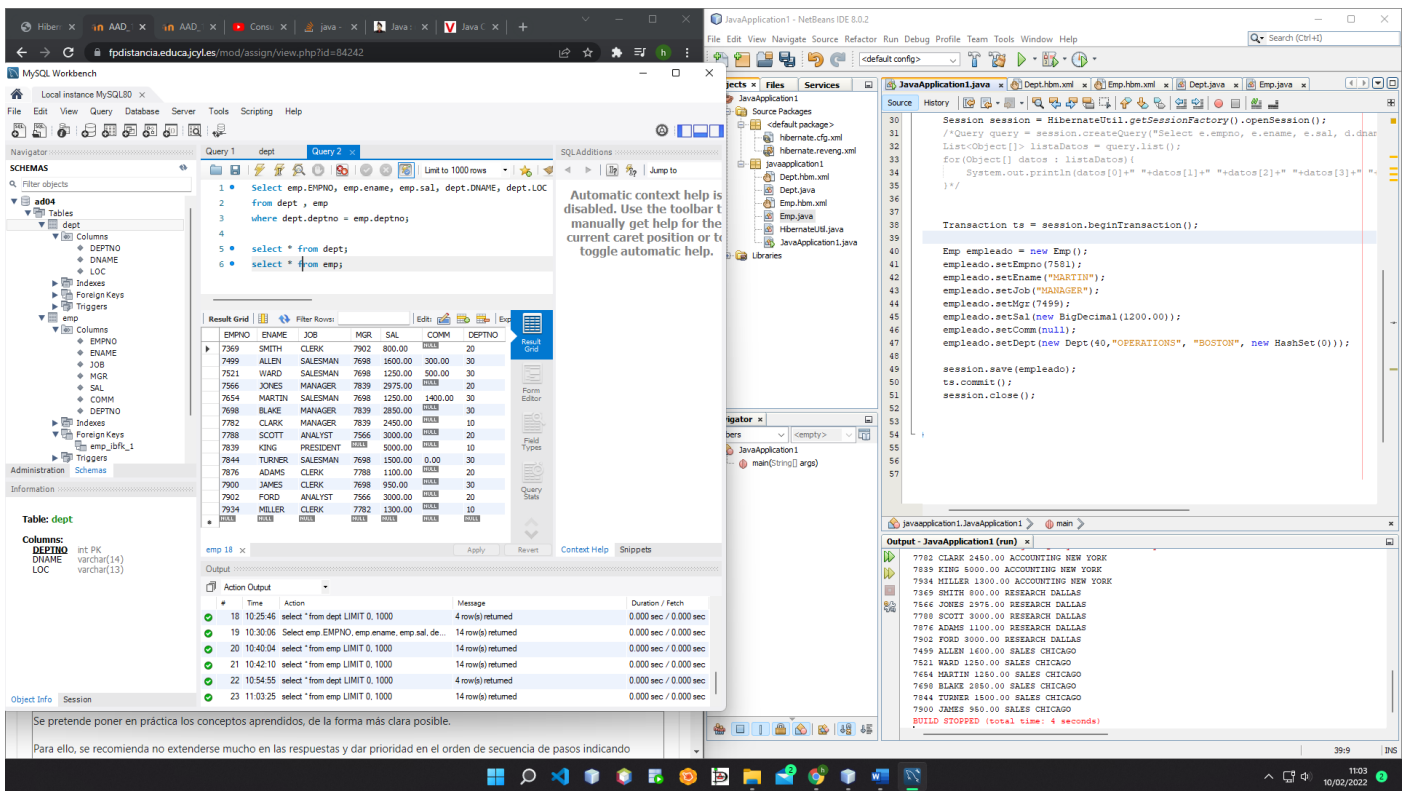
Una vez establecidas todas las configuraciones necesarias para que hibernate funcione correctamente, procedo a hacer lo que se pedía con las tablas:

Lo primero que hago es el join (lo hago con la clausula WHERE, me es mas facil de esta manera). Así hago la selección que se pide en el punto 4.

Obtener un listado sobre las tablas EMP y DEPT que visualice empno, ename, sal, dname y loc, aquí el resultado de la consulta:



Luego hago el apartado 3, realiza una inserción y un borrado sobre la tabla EMP. Lo que hice fue crear un empleado con cada uno de sus parametros (pude usar el constructor con parámetros, pero creí que así se vería más claro) y lo guardo. Como se ve en la imagen, solo hay un "MARTIN" que ya estaba y no coincide con el empleado que yo estoy haciendo.



Aquí se ve, despues de ejecutar el programa, como el nuevo empleado está añadido a la base de datos.

The screenshot shows two windows. On the left is MySQL Workbench with a query window open. The query is:

```
1. Select emp.EMPNO, emp.ename, emp.sal, dept.DNAME, dept.LOC
2. from dept, emp
3. where dept.deptno = emp.deptno;
4.
5. select * from dept;
6. select * from emp;
```

The 'emp' table is selected in the 'Table: dept' dropdown. The 'Columns' list shows: EMPNO (int PK), ENAME (varchar(14)), and LOC (varchar(13)). The 'Result Grid' shows the following data:

EMPNO	ENAME	JOB	MGR	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	800.00		20
7499	ALLEN	SALESMAN	7698	1600.00	300.00	30
7521	WARD	SALESMAN	7698	1250.00	500.00	30
7566	JONES	MANAGER	7839	2975.00		20
7581	MARTIN	MANAGER	7499	1200.00		40
7654	MARTIN	SALESMAN	7698	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	2850.00		30
7782	CLARK	MANAGER	7839	2450.00		10
7788	SCOTT	ANALYST	7566	3000.00		20
7839	KING	PRESIDENT		5000.00		10
7844	TURNER	SALESMAN	7698	1500.00	0.00	30
7876	ADAMS	CLERK	7788	1100.00		20
7900	JAMES	CLERK	7698	950.00		30
7902	FORD	ANALYST	7566	3000.00		20
7934	MILLER	CLERK	7782	1300.00		10

On the right is NetBeans IDE showing the Java code for 'JavaApplication1.java'. The code is as follows:

```
30 Session session = HibernateUtil.getSessionFactory().openSession();
31 //Query query = session.createQuery("Select e.empno, e.ename, e.sal, d.dname
32 List<Object[]> listadatos = query.list();
33 for(Object[] datos : listadatos){
34     System.out.println(datos[0]+" "+datos[1]+" "+datos[2]+" "+datos[3]+" "+
35 }
36
37
38 Transaction ts = session.beginTransaction();
39
40 Emp empleado = new Emp();
41 empleado.setEmpno(7581);
42 empleado.setName("MARTIN");
43 empleado.setJob("MANAGER");
44 empleado.setMgr(7499);
45 empleado.setSal(new BigDecimal(1200.00));
46 empleado.setComm(null);
47 empleado.setDept(new Dept(40,"OPERATIONS", "BOSTON", new HashSet(0)));
48
49 session.save(empleado);
50 ts.commit();
51 session.close();
52
53
54
55
56
57
```

The 'Output' window shows the following log messages:

```
Feb 10, 2022 11:04:12 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnecti
INFO: HH0000401: using driver (com.mysql.jdbc.Driver) at URL jdbc:mysql://localhost:3306/ad0
Feb 10, 2022 11:04:12 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnecti
INFO: HH000046: Connection properties: (user=root, password=****)
Feb 10, 2022 11:04:12 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnecti
INFO: HH000046: Autocommit mode: false
Feb 10, 2022 11:04:12 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnecti
INFO: HH000015: Hibernate connection pool size: 20 (min=1)
Feb 10, 2022 11:04:12 AM org.hibernate.dialect.Dialect<init>
INFO: HH000040: Using dialect: org.hibernate.dialect.MySQLDialect
Feb 10, 2022 11:04:12 AM org.hibernate.engine.transaction.internal.TransactionFactoryInitia
INFO: HH000039: Using default transaction strategy (direct JDBC transactions)
Feb 10, 2022 11:04:12 AM org.hibernate.hql.internal.ast.ASTQueryTranslatorFactory<init>
INFO: HH000037: Using ASTQueryTranslatorFactory
```

Por ultimo, para borrarlo, ejecuté el código que aparece en la siguiente captura y como se puede ver por consola, me reconoce que hay una coincidencia y lo borra en la base de datos, donde ya no hay dos "MARTIN", siendo borrado el elemento que yo había agregado anteriormente por la id.

The screenshot shows two windows. On the left is MySQL Workbench with a query window open. The query is:

```
1. Select emp.EMPNO, emp.ename, emp.sal, dept.DNAME, dept.LOC
2. from dept, emp
3. where dept.deptno = emp.deptno;
4.
5. select * from dept;
6. select * from emp;
7. select * from emp where empno = 7581;
```

The 'emp' table is selected in the 'Table: dept' dropdown. The 'Columns' list shows: EMPNO (int PK), ENAME (varchar(14)), and LOC (varchar(13)). The 'Result Grid' shows the following data:

EMPNO	ENAME	JOB	MGR	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	800.00		20
7499	ALLEN	SALESMAN	7698	1600.00	300.00	30
7521	WARD	SALESMAN	7698	1250.00	500.00	30
7566	JONES	MANAGER	7839	2975.00		20
7654	MARTIN	SALESMAN	7698	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	2850.00		30
7782	CLARK	MANAGER	7839	2450.00		10
7788	SCOTT	ANALYST	7566	3000.00		20
7839	KING	PRESIDENT		5000.00		10
7844	TURNER	SALESMAN	7698	1500.00	0.00	30
7876	ADAMS	CLERK	7788	1100.00		20
7900	JAMES	CLERK	7698	950.00		30
7902	FORD	ANALYST	7566	3000.00		20
7934	MILLER	CLERK	7782	1300.00		10

On the right is NetBeans IDE showing the Java code for 'JavaApplication1.java'. The code is as follows:

```
33 for(Object[] datos : listadatos){
34     System.out.println(datos[0]+" "+datos[1]+" "+datos[2]+" "+datos[3]+" "+
35 }
36
37
38 Transaction ts = session.beginTransaction();
39
40 Emp empleado = new Emp();
41 empleado.setEmpno(7581);
42 empleado.setName("MARTIN");
43 empleado.setJob("MANAGER");
44 empleado.setMgr(7499);
45 empleado.setSal(new BigDecimal(1200.00));
46 empleado.setComm(null);
47 empleado.setDept(new Dept(40,"OPERATIONS", "BOSTON", new HashSet(0)));
48
49 session.save(empleado);
50 ts.commit();
51 session.close();
52
53
54 Transaction ts = session.beginTransaction();
55 String hql = "DELETE FROM Emp e WHERE e.empno = ?";
56 Query query = session.createQuery(hql);
57 query.setInteger(0, 7581);
58 System.out.println(query.executeUpdate());
59 ts.commit();
60
61
62
```

The 'Output' window shows the following log messages:

```
Feb 10, 2022 11:43:36 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnecti
INFO: HH000040: Autocommit mode: false
Feb 10, 2022 11:43:36 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnecti
INFO: HH000015: Hibernate connection pool size: 20 (min=1)
Feb 10, 2022 11:43:36 AM org.hibernate.dialect.Dialect<init>
INFO: HH000040: Using dialect: org.hibernate.dialect.MySQLDialect
Feb 10, 2022 11:43:36 AM org.hibernate.engine.transaction.internal.TransactionFactoryInitia
INFO: HH000039: Using default transaction strategy (direct JDBC transactions)
Feb 10, 2022 11:43:36 AM org.hibernate.hql.internal.ast.ASTQueryTranslatorFactory<init>
INFO: HH000037: Using ASTQueryTranslatorFactory
Feb 10, 2022 11:43:36 AM org.hibernate.hql.internal.ast.HqlSqlWalker generatePositionalParam
WARN: [DEPRECATION] Encountered positional parameter near line 1, column 62. Positional para
BUILD STOPPED (total time: 5 seconds)
```