PRACTICA/PROYECTO 1

BASES DE DATOS

Curso 2017-18

CREACIÓN Y CONSULTAS A UNA BASE DE DATOS USANDO JAVA

1. Objetivos Generales

- Crear una base de datos en base a un diagrama E-R e insertar el contenido mediante la creación de un programa inicial de carga de datos.
- Acceder a una Base de Datos y realizar consultas SQL mediante un programa implementado en código Java.
- Manejar el conector JDBC (Java DataBase Conectivity), librería de clases para el acceso a un SGBD y para la realización de consultas SQL desde Java.

2. Objetivos específicos

El alumno será capaz de:

- Realizar conexiones a un SGBD desde un programa Java.
- Realizar consultas simples y complejas SQL manejando las clases adecuadas del conector JDBC.
- Tratamiento de los resultados de las consultas SQL dentro del código Java.
- Manejo de las excepciones en la gestión de consultas.
- Implementación de un sistema basado en el concepto de transacción usando código Java.

3. Metodología

El alumno dispone de la Máquina Virtual del curso en la que se ha configurado la misma para su utilización como gestor (servidor) de una Base de Datos MySQL. Desde el software MySQL Workbench el alumno también puede importar una base de datos cargándola como si fuera un script y ejecutando las sentencias correspondientes así como realizar las tareas de implementación y consulta que considere oportunas.

La Máquina Virtual del curso está disponible para su descarga en la página especificada en el apartado 8. En el *Anexo 1* se describe cómo utilizar este paquete.

Adicionalmente la Máquina Virtual del curso dispone de una *máquina virtual java* y de la herramienta de compilación y ejecución *Eclipse*. Desde la Máquina Virtual del curso y con la

herramienta *Eclipse* se implementará el código fuente java pedido en este proyecto. La herramienta Eclipse, instalada en la máquina virtual del curso, dispone en la carpeta "*C:\eclipse\CONECTOR_MYSQL*" del conector JDBC "*mysql-connector-java-5.1.38-bin.jar*" que se deberá usar para la realización de este proyecto. Asimismo tiene configurada la carpeta "C:\eclipse\proys" como la ubicación por defecto de los proyectos *java*.

Se recuerda que el uso de la máquina virtual para esta práctica aunque no es obligatorio si es recomendable, y que toda duda que surja por la no utilización de la máquina virtual, es responsabilidad del alumno buscar una solución.

4. Práctica a realizar

Se desea realizar un programa que trabaje con información sobre enfermedades y sus síntomas. El objetivo del programa es el de tener un sistema que permita tanto realizar el diagnóstico de enfermedades en base a los síntomas como el de, dada una enfermedad, listar cuáles son sus síntomas, códigos o tipos semánticos de los síntomas así como mostrar una serie de estadísticas básicas del contenido de la BD. En este sentido, las funcionalidades básicas que se requiere que tenga el programa (y su puntuación asociada) son:

1. Creación de la base de datos en base al esquema E-R y carga de los datos del fichero .data (ambos proporcionados). [3 puntos]: Dado el fichero de datos (ver sección 6 para más información), se debe cargar su contenido usando código Java, procesarlo y en base a la estructura del E-R y los datos contenidos, insertar la información en la base de datos. Debe tenerse en cuenta el diagrama E-R para entender como son los datos y como deben por lo tanto almacenarse. Obligatoriamente, debe ejecutarse la creación y carga de todos los datos como si fuera una única transacción, de tal forma que cualquier fallo intermedio de lugar a deshacer por completo los cambios anteriores.

2. Capacidad de realizar diagnósticos en base a la introducción de síntomas contenidos en la base de datos [2 puntos]: El programa debe pedir por teclado que síntomas quiere tener en cuenta para realizar el diagnóstico. Los síntomas deben solicitarse para ser introducidos por teclado, y debe previamente mostrarse la lista de síntomas disponibles, para que quien use el programa pueda introducirlos usando sus códigos identificativos (no a través del nombre del síntoma). El sistema debe devolver aquellas enfermedades que **tenga asociados todos los síntomas** que se hayan introducido.

3. Capacidad de listar [1.5 puntos]:

    a. Los síntomas de una enfermedad de la base de datos [0.5 puntos]: Debe mostrar cuales son las enfermedades que hay en la base de datos, y el usuario debe introducir el ID que corresponda para seleccionar la enfermedad.

    b. Listado de enfermedades de la base de datos y sus códigos asociados [0.5 puntos]: El programa debe mostrar las enfermedades que contiene la base de datos, y para cada enfermedad que códigos tiene (y tipo de código).

c. Listado de síntomas disponible en la base de datos y sus tipos semánticos asociados [0.5 puntos]: El programa debe mostrar los síntomas que contiene la base de datos y sus tipos semánticos asociados.

4. Estadísticas de la base de datos y su contenido. La funcionalidad de estadísticas debe proporcionar [2 puntos]:

    a. Número de enfermedades [0.5 puntos]: Un conteo del número total de enfermedades que hay en la base de datos.

    b. Número de síntomas [0.5 puntos]: Un conteo del número total de síntomas que hay en la base de datos.

    c. Enfermedad con más síntomas, con menos síntomas y número medio de síntomas [0.5 puntos]: Debe indicar cuales son las enfermedades con más y menos síntomas y cuál es el número medio de síntomas asociados a las enfermedades.

    d. Tipos de semantic types en los síntomas y distribución de cada semantic type (cuantos síntomas hay de cada semantic type) [0.5 puntos]: Se debe indicar cuales son los semantic types que hay en la base de datos, y cuantos síntomas hay de cada semantic type.

Se pide que el código sea lo más limpio y ordenado posible, así como la salida que muestre el programa. Los 1.5 puntos restantes se valorarán en función de la limpieza del código y otros aspectos que el profesor pueda considerar relevantes (gestión correcta de errores, optimización del código, comentarios, etc.).

El programa **debe conectarse cuando se vaya a realizar una consulta/operación contra la BD por primera vez** y **no debe desconectarse hasta que el programa finalice**. El programa debe finalizar siempre llamando al método exit(), por lo tanto, la desconexión de la base de datos y operaciones de cierre **deben** realizarse dentro de dicho método.

La opción de creación de la base de datos y carga de los datos debe realizarse solo en el caso de que se detecte que la base de datos no exista. En caso de existir, debe devolver un error indicando que ya existe y que no procede su creación/introducción de datos. Adicionalmente y de forma optativa se podría pedir por teclado la opción de borrar la base de datos existente para cargarla de nuevo.

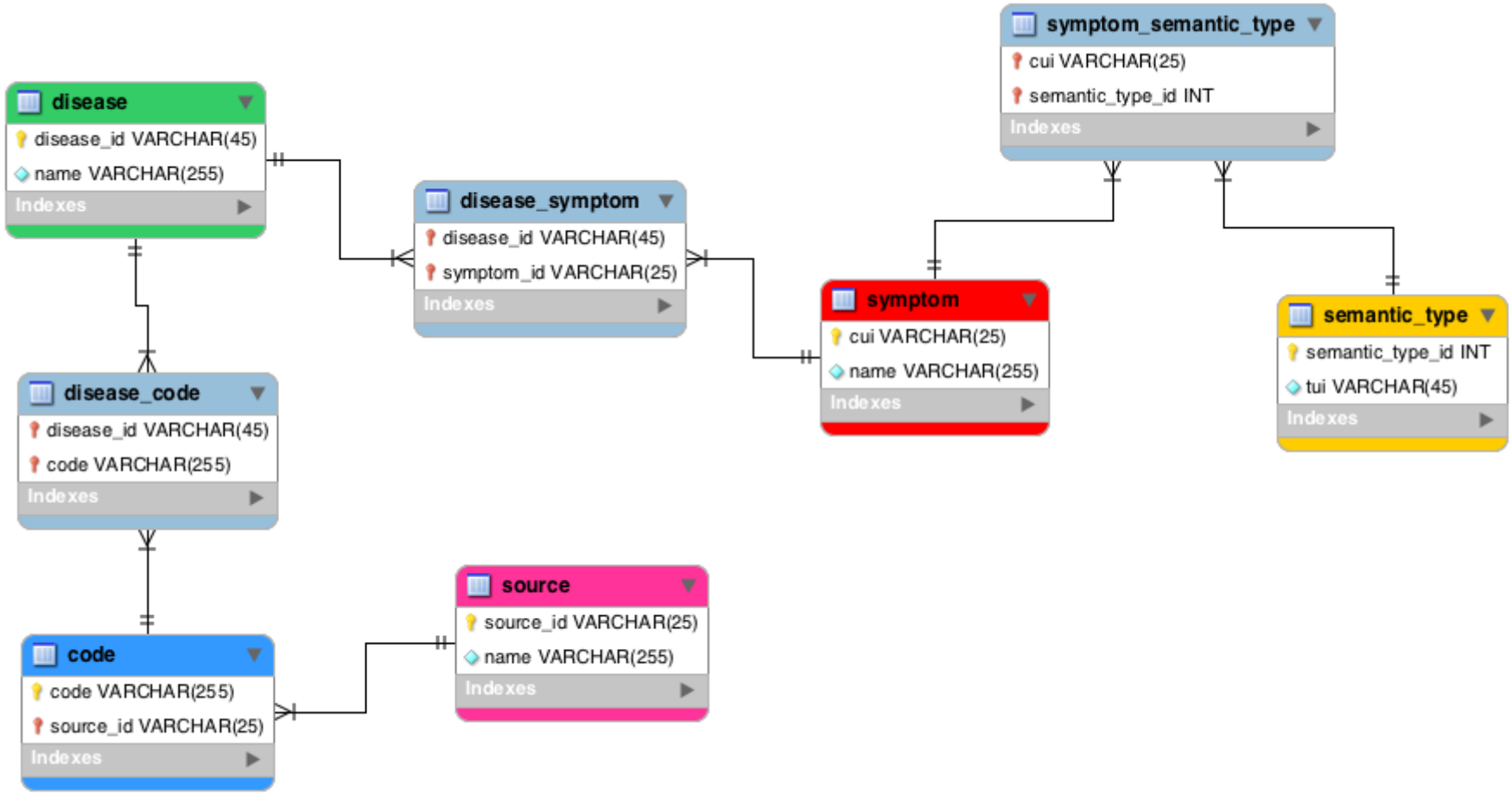

Imagen 1. Diagrama entidad relación de la base de datos “diagnostico”

5. Consideraciones sobre fichero Diagnostico.java

El fichero Diagnostico.java es un fichero que está ya listo para ser ejecutado (contiene el método main y llama directamente al menú de ejecución del programa). El menú está preparado para que se ejecuten las diferentes operaciones que se pueden escoger (de la 1 a la 7) y que tras ello vuelva a mostrarse el menú. Solo se sale del menú (y del programa) escogiendo la opción 7 que es la de salir.

El fichero tiene implementados dos métodos que los alumnos deben usar para obtener datos del usuario: readInt (para leer valores enteros) y readString (para leer cadenas).

El método exit() se llamará cuando se finalice el programa y debe introducirse el código necesario para desconectar de la base de datos y demás operaciones de cierre.

Así mismo existen ya creados los diferentes métodos de las 6 operaciones que se piden implementar, con lo que el alumno simplemente debe rellenar el código en dichos métodos.

El método conectar() debe ser implementado para realizar la conexión a la base de datos cuando corresponda según lo indicado en el enunciado.

También existe implementado ya el método readData(), que devuelve una lista de String (LinkedList<String>) que contiene, en cada String, cada una de las líneas del fichero de datos. El fichero de datos está establecido para estar, obligatoriamente (no se puede modificar), dentro de la carpeta “data”, y el nombre, es el que ya se ha proporcionado.

6. Consideraciones sobre fichero disease_data.data

El fichero disease_data.data contiene las enfermedades, síntomas, códigos, etc. que deben ser añadidos en la base de datos. Si se observa el código de la clase Diagnostico, como se ha mencionado previamente, hay un método (readData()) que ya está implementado y que permite leer todas las líneas del fichero, para que el alumno pueda luego procesarlo y extraer la información que debe insertar en las tablas correspondientes. La estructura del fichero contiene la información en base a las siguientes características:

- Cada línea es una enfermedad.
- Las enfermedades contienen:
    - Su nombre

- Sus códigos (cada enfermedad puede tener un código identificativo en diferentes vocabularios. Es decir, podemos tener un código para el vocabulario OMIM, otro para MedlinePlus, otro para DiseasesDB, etc.). No todas las enfermedades tienen el mismo número de códigos disponible.

- Síntomas (con su nombre, código y semantic type).

- La estructura de la información de cada enfermedad sigue el siguiente patrón:

Nombre enfermedad:codigo1@vocabulario1;codigo2@vocabulario2;….;codigoN@vocabularioN=Sintoma1:codigoSintoma1:semanticTypeSintoma1; Sintoma2:codigoSintoma2:semanticTypeSintoma2; …. ;SintomaN:codigoSintomaN:semanticTypeSintomaN

Para el procesado de la información, se recomienda al alumno que revise la documentación de Java, concretamente del método split() de la clase String[1].

7. Evaluación y otras indicaciones

Es importante que se cumplan los requisitos establecidos en la práctica, incluyendo:

- El método conectar() debe ser el encargado de: cargar el driver y realizar la conexión. Es obligatorio que se implemente esta funcionalidad dentro de dicho método. Parámetros:

  - Servidor: localhost:3306
  - Usuario: bddx
  - Password: bddx_pwd
  - Nombre base de datos: diagnostico

- Que sean los métodos ya dados los que, al menos, se encarguen de proporcionar el resultado final. El alumno puede generar otros métodos adicionales si lo considera relevante, pero el método asignado a cada ejercicio debe devolver el resultado.

- El fichero de la clase Java debe llamarse obligatoriamente “Diagnostico.java” tal y como se entrega. Es decir, debe usarse en realidad ese fichero, no se debe crear uno nuevo. El código ha sido creado para que no haya una estructura de paquetes, de modo que no debe enviarse el fichero .java con ninguna línea que haga referencia a paquetes. Solo puede entregarse dicho fichero .java. En caso de necesitar crear objetos extra para el procesamiento de datos, deben hacerse como clases inner[2].

[1] https://docs.oracle.com/javase/7/docs/api/java/lang/String.html

[2] https://www.tutorialspoint.com/java/java_innerclasses.htm

- La práctica se evaluará de 0 a 10. Cada ejercicio propuesto tiene una puntuación asociada y todos suman un total de 8.5 puntos. El punto y medio restante lo establecerá el profesor en función de criterios como:
    - Limpieza de código.
    - Uso correcto de excepciones y elementos similares.
    - Consultas optimizadas.
    - Otros indicadores a estimar por el profesor durante la corrección.
    - Salida de datos ordenada y limpia.

- El profesor se reserva el derecho de citar a cualquier grupo a defender la práctica si lo considerara necesario.
- La base de datos a ser creada debe llamarse también obligatoriamente “diagnostico”.

8. Enlace Descarga Software

https://drive.upm.es/index.php/s/7tRvUY2h6tApctE

9. Entrega de la Práctica

El grupo de prácticas deberá **entregar mediante la plataforma Moodle en una tarea habilitada para ello un único fichero comprimido (*.zip, *.rar) cuyo nombre sea (Grupo_N_JDBC.rar) (donde N es el número de grupo correspondiente) que contenga:**

1. Clase Diagnostico.java con la implementación solicitada.
2. Opcionalmente un documento en PDF que contenga comentarios acerca de los problemas surgidos al hacer la práctica o cualquier otro comentario que los alumnos estimen oportuno.

La entrega debe ser realizada por **SOLO** uno de los miembros del grupo (preferiblemente el representante).

La fecha límite para la entrega de esta práctica es el día jueves 3 de Mayo de 2017 (10:00).

10. Consultas acerca del desarrollo del ejercicio práctico

Cualquier duda acerca del desarrollo del ejercicio práctico se podrá resolver en el despacho 4302 (Alejandro Rodríguez González) o mediante email en el correo alejandro.rg@upm.es. Si un alumno desea resolver alguna duda de manera presencial en el despacho debe concertar cita con el profesor previamente.

A1. 1 Paquete XAAMP

Con el fin de facilitar la tarea de la realización de las prácticas propuestas en la asignatura de Bases de Datos se proporciona el software "*PAQUETE XAMPP PRÁCTICAS SQL*". Este paquete es una versión del software *XAMPP* de libre distribución (GNU General Public License).

Este paquete software contiene un gestor de *Bases de Datos Mysql* que se propone para hacer las prácticas de la asignatura. La ventaja de utilizar este paquete es que el alumno no tiene que instalar ningún servicio ni ninguna aplicación en su sistema.

Este paquete aparece integrado en la Máquina Virtual del curso. En este caso el paquete ya está configurado y listo para ser usado.

En este momento alumno podrá utilizar las aplicaciones incluidas en el paquete *XAMPP*. Para la práctica propuesta el alumno deberá pinchar en primer lugar el botón "*start*" de la aplicación "*Apache*". Si todo ha ido bien deberá aparecer la palabra "*running*" resaltada de color verde a la altura del nombre de la aplicación como se aprecia en la figura A4. A continuación deberá pinchar el botón "*start*" de la aplicación "*Mysql*". Si todo ha ido bien deberá también aparecer la palabra "*running*" resaltada de color verde a la altura del nombre de la aplicación como se aprecia en la figura A1.1.

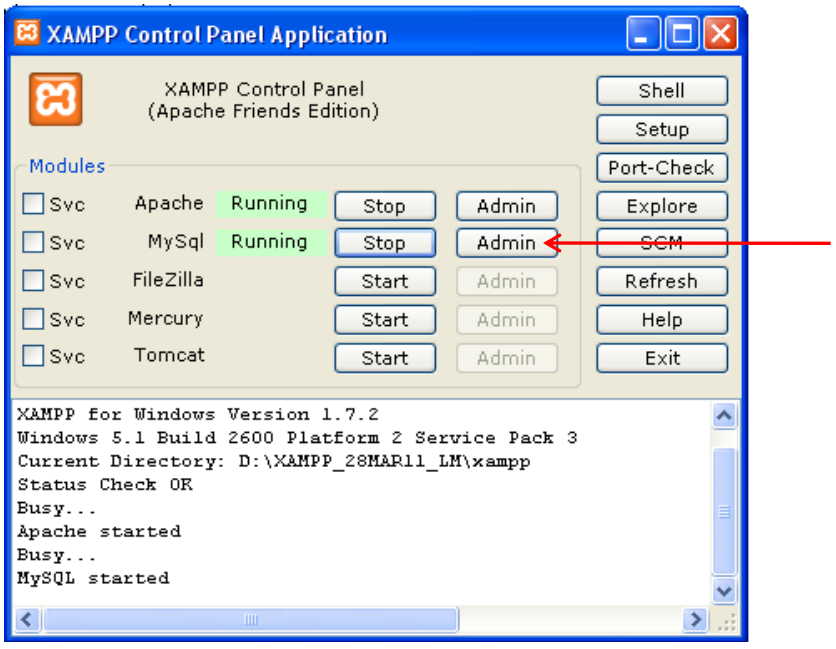

Figura A1.1

Ahora, el alumno puede pinchar el botón "*Admin*" del panel de control del paquete *XAMPP* (ver Figura A4). De este modo el alumno accede al entorno denominado "*phpMyAdmin*" el cual es

una aplicación web de acceso administrativo al gestor de Base de Datos *MySql* . Desde este entorno (ver Figura A1.2) el alumno puede por ejemplo exportar una Base de Datos, ver el diagrama de esquema de una Base de Datos, crear tablas, hacer consultas de distintas tablas, etc ….

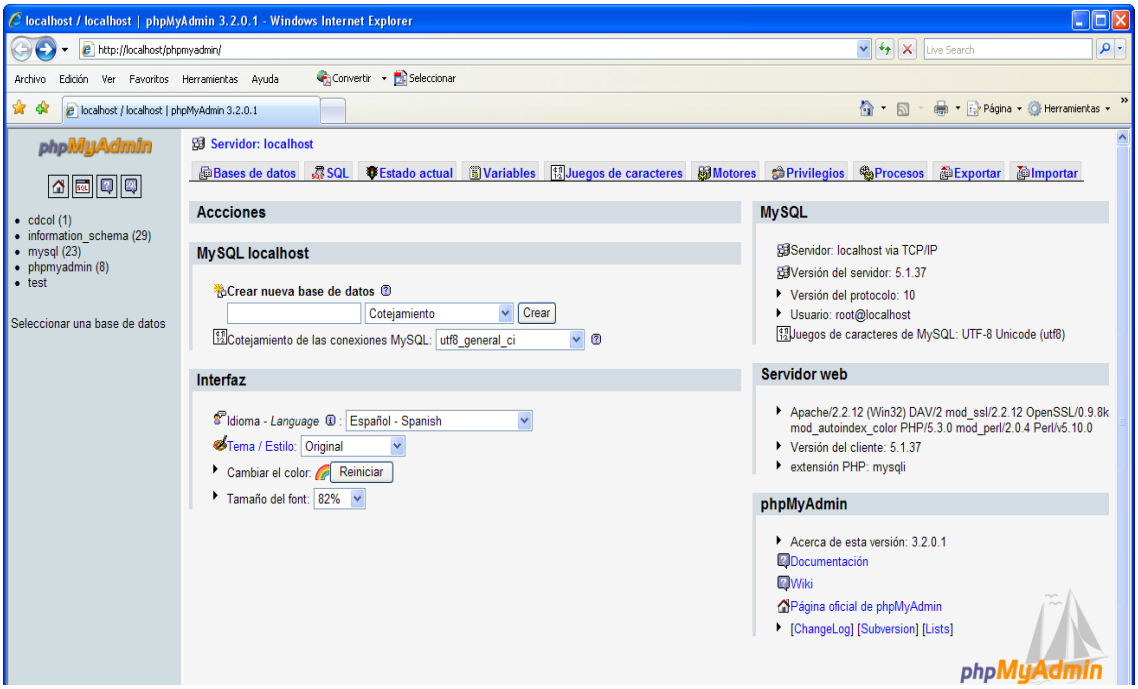

Figura A1.2: Entorno Administración "*phpMyAdmin*"

Se recomienda, no obstante, que el alumno utilice un cliente, por ejemplo, el *"MySQL Workbench"* para conectarse al Servidor *MySQL* y realizar consultas DDL (Data Definition Language) y DML (Data Manipulation Language), dejando el entorno "*phpMyAdmin*" para tareas de administración (P.e. exportar una Base de Datos y ver el diagrama de esquema de una Base de Datos. (ver Figura A1.3).

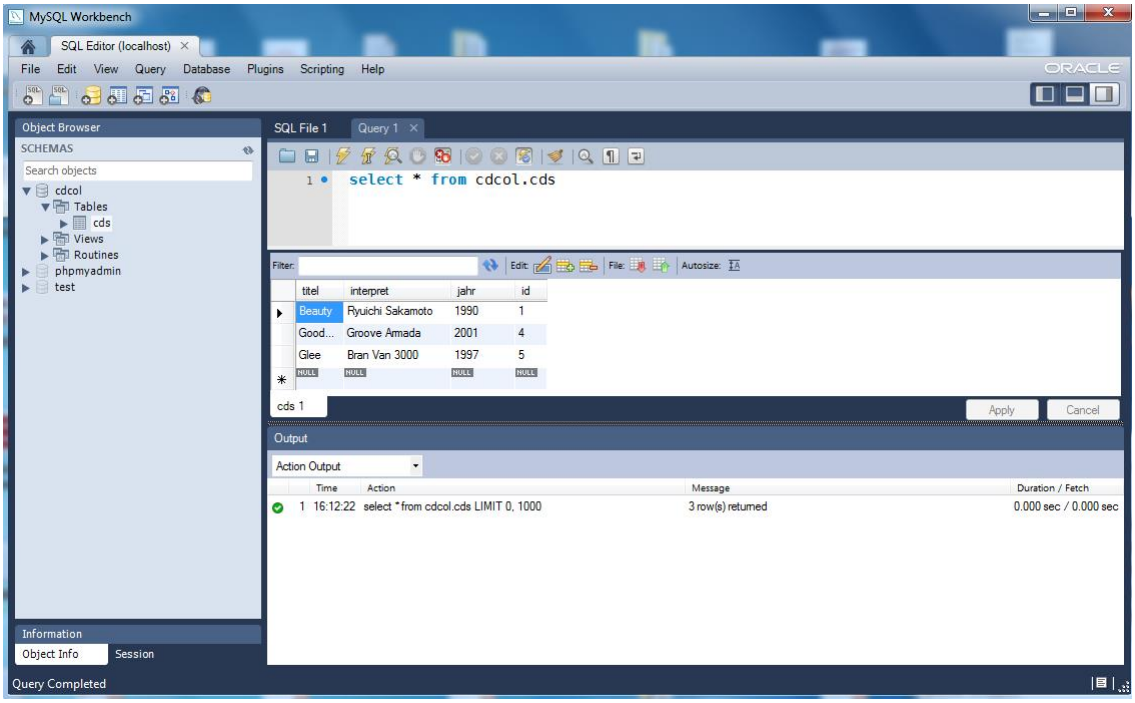

Figura A1.3: Cliente *Workbench*