Actividad UT2_03: JDBC y MySQL. Base de datos de empleados

Utilizando MySQL Workbench, crea una base de datos de nombre "**empresa**".

Crea un proyecto en NetBeans, de nombre UT3_ACT03_MYSQL_EMPLE, en el que vas a desarrollar una aplicación para gestionar la base de datos "empresa", utilizando JDBC. Debes tener en cuenta las siguientes consideraciones:

- En todos los métodos se debe mostrar por consola un mensaje de éxito o error.
- Se debe comentar el código, identificando claramente cada bloque y lo que hace.
- Además, para poder seguir la ejecución del programa, se deben añadir mensajes por consola que indiquen el método que se está ejecutando.

MÉTODOS A IMPLEMENTAR

- **private static String creaUrl ():** crea y devuelve la cadena de conexión a la base de datos. Utiliza un objeto de la clase **Properties** para definir todas las propiedades de la conexión: usuario, contraseña, nombre de la base de datos, el uso de Unicode y codificación UTF-8. Los valores de estos parámetros se guardan en un **fichero de extensión .properties dentro del paquete**.

- **private static Connection abreConexion (String url):** recibe una cadena de conexión a una base de datos y abre la conexión a la base de datos especificada.

- **private static void cierraConexion (Connection con):** cierra la conexión a la base de datos controlando todas las excepciones que se puedan producir.

- **private static void verInfoBD (Connection con):** recibe una conexión y muestra por pantalla el nombre del sistema gestor de la base de datos, nombre del driver, la URL de la conexión y usuario conectado.

- **private static void crearTablaDepartamentos (Connection con, String tname):** recibe una conexión y el nombre de una tabla, y crea en la base de datos una tabla de departamentos con ese nombre (si no existe) y con la siguiente estructura:

  TABLA de departamentos:

  - **dep_id:** identificador del departamento. Tipo entero, clave primaria **autogenerada.**
  - **dep_nombre:** nombre del departamento. Tipo texto de 15 caracteres. No puede estar vacío.

- **private static void crearTablaEmpleados (Connection con, String tEmple, String tDep):** recibe una conexión, el nombre de la tabla de empleados y el nombre de la tabla de departamentos, y crea en la base de datos una tabla de empleados con ese nombre (si no existe) y con la siguiente estructura:

  TABLA de empleados:

  - **emp_id:** identificador del empleado. Tipo entero, clave primaria **autogenerada.**.
  - **apellido:** Tipo texto de 15 caracteres. No puede estar vacío.
  - **salario:** tipo FLOAT. No puede estar vacío.
  - **dep_id:** identificador del departamento del empleado. No puede estar vacío. **Clave ajena que referencia a la clave primaria de departamentos.**

- **private static void verClaves (Connection con, String tname):** Muestra las claves primarias, claves importadas y claves exportadas de la tabla cuyo nombre recibe como parámetro.

- **private static void insertarDepartamentos (Connection con, String tname, String fich):** recibe una conexión, el nombre de la tabla de departamentos y el nombre de un fichero, con su ruta. Inserta en la tabla de departamentos todos los departamentos que están en el fichero. El fichero es un fichero de texto en el que cada línea contiene un nombre de departamento. Utiliza el método executeUpdate() de la interfaz PreparedStatement, y sus métodos setXXX para parametrizar las sentencias SQL correspondientes.

- **private static ResultSet consultarDepartamentos (Connection con, String tdep):** recibe una conexión y el nombre de la tabla de departamentos y devuelve un ResultSet con todos los datos de la tabla de departamentos.

- **private static void insertarEmple (Connection con, String apellidos, int salario, int dep_id, String tname)**: recibe una conexión, los datos de un empleado y el nombre de una tabla. Inserta en la tabla el empleado con los datos especificados. Muestra el número de registros que se han insertados.

- **private static ResultSet consultarEmpleados (Connection con, String tDep, String tEmple):** recibe una conexión y los nombre de las tablas de departamentos y empleados, y devuelve un ResultSet con el id, apellido, salario y **nombre** del departamento de todos los empleados. Utiliza una única sentencia SQL que se ejecuta con un PreparedStatement.

- **private static void mostrarRegistros (ResultSet resul):** recibe un ResultSet y muestra todos sus datos por consola con formato tabulado. Si el ResultSet está vacío se indica con un mensaje. Para procesar el ResultSet hay que obtener los metadatos necesarios, ya que no se conoce a priori la estructura del mismo.

- **private static void modificarSalarios (Connection con, String tname):** recibe una conexión y el nombre de la tabla de empleados. Pide por consola un valor de porcentaje (entre 3% y 10 %) y sube en ese porcentaje el salario de todos los empleados que tienen un salario inferior a 1200. Utiliza PreparedStatement y sus métodos setXXX para parametrizar las sentencias SQL correspondientes. Muestra el número de registros que se han actualizado.

- **private static void eliminarEmplePorId (Connection con, String tname):** recibe una conexión y el nombre de la tabla de empleados y borra un empleado con un id determinado, que se pedirá por consola. Utiliza el método executeUpdate() de la interfaz PreparedStatement, y sus métodos setXXX para parametrizar las sentencias SQL correspondientes. Si no existe el id de empleado se indicará con un mensaje.

MÉTODO main

En el método main, escribir el código necesario para, utilizando estos métodos, hacer las siguientes operaciones:

1) Abrir la conexión a la base de datos (método abreConexion).

2) Mostrar información de la base de datos: nombre del sistema gestor de la base de datos, nombre del driver, usuario y url utilizados en la cadena de conexión (método verInfoBD).

3) Crear las tablas de departamentos y empleados (métodos crearTablaDepartamentos y crearTablaEmpleados)

4) Mostrar las claves primarias, importadas y exportadas de las tablas de empleados y departamentos (método verClaves).

5) Insertar 3 registros en la tabla de departamentos. Los nombres de los departamentos son ventas, marketing y finanzas, y están guardados, en este orden, en un fichero de texto que se encuentra en el directorio “src/datos” del proyecto (crea el fichero de la forma que quieras). Muestra el número de registros insertados. (método insertarDepartamentos).

6) Mostrar los datos de todos los departamentos (métodos consultarDepartamentos y mostrarRegistros).

7) Insertar los siguientes registros en las tabla de empleados (método insertarEmple):

apellido	salario	dep_id
Moreno	2300	1
Val	1000	2
Amigo	1900	1
Lucas	1100	2
Torres	1600	3

8) Mostrar los siguientes datos de todos los empleados: id, apellido, salario y nombre del departamento (métodos consultarEmpleados y mostrarRegistros).

9) Subir el salario a los empleados cuyo salario es inferior a 1200, en un porcentaje que se pide al usuario (entre 3% y 10%). Muestra el número de registros actualizados (método modificarSalarios). A continuación vuelve a mostrar los datos de los empleados.

10) Eliminar un empleado cuyo id se pide al usuario. Muestra un mensaje si el id no existe, o de éxito se elimina el registro (método eliminarEmplePorId).

11) Cerrar la conexión a la base de datos (método cierraConexion)