ACCESO A BASES DE DATOS

EJERCICIO 1

Se dispone del archivo **usuariosDB.sql** que crea la tabla usuarios con los campos necesarios y del interfaz de usuario en el proyecto **ejercicio1**, completar los siguientes métodos de la clase **AccesoDatos**:

1. El método **autenticar** que dado un nombre de usuario y una contraseña establece los atributos **login y tipo_usuario** según si los datos de usuario son correctos o no. Si son correctos el usuario se considera autenticado y listo para hacer las acciones pertinentes según el tipo de usuario.
2. El método **mostrarDatos** que devuelve un **ArrayList<String[]>** con los datos para rellenar una JTable. El primer elemento del ArrayList deben las columnas y el resto los datos de usuarios. Si el usuario está autenticado y es tipo normal devuelve la información de ese usuario y si es admin devuelve la información completa de todos. Si no está autenticado lanzar una excepción.
3. El método **nuevoUsuario** solo un **admin** puede que permite crear nuevos usuarios a partir del nombre y pidiendo 2 veces el **pass** donde el tipo es normal por defecto comprobando que el nombre de usuario no existe previamente y poniendo la fecha actual como fecha de alta. Solo un usuario de tipo **admin** puede realizar esta acción. En caso de no poder crearlo lanzar una excepción.
4. El método **borrarUsuario** que permite borrar un usuario recibiendo el nombre de usuario. Solo un usuario de tipo **admin** puede realizar esta acción. En caso de no poder borrarlo lanzar una excepción.
5. El método **modificarPassword** que permite modificar el password de un usuario que ya está autenticado previamente, recibiendo la contraseña anterior para comprobarla previamente y de la contraseña nueva. En caso de no poder lanzar una excepción.

EJERCICIO 2

Usar el archivo **sucursales.sql** que crea las tablas necesarias para guardar información sobre sucursales bancarias y las cuentas abiertas por clientes, además del interfaz de usuario en el proyecto **ejercicio2**, completar los métodos de la clase **AccesoDatos**:

1. Completar el método **verPensiones** que devuelve **ArrayList<String[]>** con toda la información de todas las cuentas de la entidad que son de tipo pensión. Si no hay cuentas tipo pensión lanzar una excepción.

2. Completar el método **cuentasSucursal** que devuelve **ArrayList<String[]>** con el propietario, saldo y tipo de todas las cuentas de la entidad de una sucursal concreta a partir del nombre de la sucursal. Si no hay cuentas lanzar una excepción.
3. Completar el método **nuevaSucursal** que crea una nueva sucursal a partir de sus datos comprobando previamente que no existe la denominación ya en la base de datos, en otro caso lanzar una excepción para informar de dicho suceso.
4. Completar el método **aumentarCuentas** que sube un % las cuentas de todas las sucursales. Ese tanto por ciento se pasa como parámetro.
5. Completar el método **mejorDirector** que devuelve un String con el nombre del director/a y la denominación de la sucursal con la cuenta con mayor saldo.

EJERCICIO 3

Se dispone del archivo **liga.sql** que crea las tablas necesarias para guardar información sobre equipos y jugadores de la liga española, además del interfaz de usuario en el proyecto **ejercicio3**, completar los siguientes métodos de la clase **AccesoDatos**:

1. El método **verEquipos** que devuelve un **ArrayList<String[]>** con los equipos de la base de datos ordenados por fecha de fundación. Si no hay equipos lanzar excepción.
2. El método **crearEquipo** que añade un equipo nuevo en la base de datos comprobando antes que el nombre no está cogido, devolviendo una excepción en caso contrario.
3. El método **verJugadores** que devuelve un **ArrayList<String[]>** con todos los jugadores de un equipo a partir del nombre equipo. Si el club no tiene jugadores lanzar excepción.
4. El método **verPichichi** que devuelve un **String** con el nombre del jugador que ha marcado más goles en la liga y el nombre del equipo al que pertenece.
5. El método **verResumen** que devuelve un **String** con todos los nombres de todos los equipos que hay en la liga junto con el total de goles marcados por cada uno.

EJERCICIO 4

Se dispone del archivo **club.sql** que crea las tablas necesarias para guardar información sobre los socios de un club y los eventos que se celebran en él, además del interfaz de usuario en el proyecto **ejercicio4**, completar los siguientes métodos de la clase **AccesoDatos** (se deben controlar las posibles excepciones que se puedan producir):

1. El método **apuntarseEvento** que dado el nombre de un socio y el nombre de un evento inscribe al socio en dicho evento.
2. El método **eventosSocio** que dado el nombre de un socio devuelve un **ArrayList<String[]>** con los eventos a los que está apuntado.

3. El método **sociosEvento** que dado el nombre de un evento devuelve un **ArrayList<String[]>** con los socios que están apuntados a dicho evento.
4. El método **eventoMultitudinario** que devuelve el nombre del evento con más asistentes que ha tenido o tendrá el club.
5. El método **sinSocios** que devuelve un String con los nombres de los eventos a los cuales no se ha apuntado ningún socio.

EJERCICIO 5

Se dispone del archivo **tienda.sql** que crea las tablas necesarias para guardar información sobre productos, clientes y ventas que se realizan en una tienda online, además del interfaz de usuario en el proyecto ejercicio5, completar los siguientes métodos de la clase **AccesoDatos** (se deben controlar las posibles excepciones que se puedan producir)**:**

1. El método **añadirVenta** que, a partir del nombre de un producto, un cliente y una cantidad de unidades que se lleva añade una venta a la tabla con la fecha actual.
2. El método **ultimaVenta** que devuelve un String con el nombre del cliente y del producto de la última venta realizada en la tienda.
3. El método **masVendido** devuelve el nombre de producto (String) con mayor cantidad de ventas en total de todos los clientes de la tienda.
4. El método **sinVentas** que nos devuelve un String con los nombres de los productos de los cuales no se ha realizado ninguna venta todavía.
5. El método **sinCompras** que nos devuelve un String con los nombres de los clientes de los cuales no se han comprado nada todavía.