PracticaJDBC2021

Estructura de Tablas

Dadas las tablas

- *Clientes (idCliente(PK), DNI)*
  en la que cada fila representa la información de un cliente de un banco. DNI es UNIQUE NOT NULL (clave candidata).
- *Cuentas (idCta(PK), idOficina, fechaCreacion, saldo)*
  en la que cada fila representa la información de una cuenta corriente.
- *Cuentas_Clientes (idCta, idCli)*
  que representa la relación varios a varios que hay entre cuentas y clientes autorizados (un cliente puede estar en varias cuentas como autorizado, una cuenta puede temer varios clientes autorizados)

El script que crea las tablas está en `src/lsi/ubu/PracticaJDBC2021/script_PracticaJDBC2021.sql` y el *main* ya lo corre automáticamente al llamar a *`pruebaAltaAutorizado()`*, que a su vez llama a *`creaTablas()`*.

Descripción General

En `lsi.ubu.PracticaJDBC2021.PracticaJDBC2021.java` está el fichero que se te pide completar. Del mismo, se pide implementar el método:

```
public static void altaAutorizado( int cta, int dni) throws
SQLException_PracticaJDBC2021
```

que dado un número de cuenta `cta`, y un DNI de un cliente (argumento `dni`) inserta en la tabla *Cuentas_Clientes* la fila correspondiente para que ese cliente figure como autorizado de esa cuenta.

Algoritmo

Para ello, se sugiere seguir estos pasos (pero se puede hacer de otras formas):

1. Consultar en la BD los clientes con DNI igual al pasado por parámetro.
2. Si existe ese cliente
   1. insertar en la tabla de *Cuentas_Clientes* la fila que conecta a la cuenta pasada por parámetro con el *idCliente* (ojo, no es el DNI) obtenido en la consulta del paso 1.
3. Si no
   1. Lanzar la excepción correspondiente que indique que no existe el cliente con ese DNI.

El método tomará una conexión del pool al principio, y cometerá la transacción. Al finalizar cerrará todos los recursos utilizados.

Excepciones

El método lanza sólo excepciones del tipo `SQLException_PracticaJDBC2021`.

Hay dos excepciones posibles:

1. La primera excepción recogería el problema de que **no existiera el DNI del cliente que se pretende autorizar.** Su código será el "-1" y el mensaje de error `"No existe cliente con ese DNI"`. Ver `lsi.ubu.PracticaJDBC2021.SQLException_PracticaJDBC2021.java`.
2. La segunda excepción recogerá el problema de que al intentar autorizar la cuenta **no existiera la cuenta corriente objeto de la autorización, dando un problema de violación de clave ajena**. Su código será el "-2" y el mensaje de error `"No existe ese nro de cuenta corriente"`. Ver también `lsi.ubu.PracticaJDBC2021.SQLException_PracticaJDBC2021.java`.

   Para detectar este problema la clase `lsi.ubu.OracleTableError.java` tiene codificado el error ***2291*** de Oracle como ***`FK_VIOLATED`***, que precisamente es el error que ocurre cuando se viola una clave ajena por insertar una fila sin padre (en este caso el padre es la cuenta corriente y el hijo la fila que estamos insertando en la tabla de *cuentas_clientes*). Se valora que tu solución no dependa de que este error sea codificado como 2291, usando para ello el método `checkExceptionToCode` y la clase `OracleTableError.java`.
3. La tercera excepción recogerá el problema de que al intentar autorizar la cuenta para el cliente con ese DNI **ya existiera dicha autorización, dando un problema de violación de clave primaria de la tabla *Cuentas_Clientes*.**

   Para detectar este problema la clase `lsi.ubu.OracleTableError.java` tiene codificado el error ***1*** de Oracle como ***`UNQ_VIOLATED`***, que precisamente es el error que ocurre cuando se viola una clave primaria compuesta por insertar una fila con valores repetidos (ya existe en la tabla esa combinación cuenta-DNI). Se valora que tu solución no dependa de que este error sea codificado como 1, usando para ello el método `checkExceptionToCode` y la clase `OracleTableError.java`.

Nota: A diferencia de otro enunciado de este curso las excepciones han sido codificadas con números negativos precisamente para evitar que alguno se codificara como "1" (positivo), lo que haría que ese código coincidiese con el error – también – 1 de Oracle correspondiente a ***`UNQ_VIOLATED`***.

El resto de excepciones serán tratadas dentro del propio método.

Toda excepción sea del tipo que sea:

1. retrocederá la transacción y
2. dejará el mensaje con `l.error(...)` del error en el *logger*.

Si además, la excepción que estamos tratando es del tipo `SQLException_PracticaJDBC2021` , esta se lanzará o propagará al *main* que hace la llamada al método. El resto de excepciones serán tratadas dentro del propio método (basta en ese caso con el ya mencionado mensaje en *logger).*

Pruebas automáticas

Las pruebas automáticas se invocan desde el *main* y están en el método `pruebaAltaAutorizado`. Aunque no hayas empezado a hacer nada el fichero `PracticaJDBC2021.java` compila y ya te muestra los mensajes de las pruebas aún no superadas. Esas pruebas hacen 3 cosas:

1. La primera prueba intenta dar de alta como autorizado un cliente cuyo DNI (el 44444) no existe en una cuenta que si que existe según el script sql (la cuenta 1). Por tanto provoca el error `SQLException_PracticaJDBC2021` número -1 (por eso, de momento, al ejecutarlo, te sale `Cliente inexistente MAL`).

2. La segunda prueba intenta dar de alta como autorizado un cliente cuyo DNI (el 12345678) si que existe según el script sql, pero cuya cuenta (la cuenta 2) no existe. Por tanto provoca el error por violación de clave ajena ya comentado que acarrea la `SQLException_PracticaJDBC2021` número -2 (por eso, de momento, al ejecutarlo, te sale `Cuenta inexistente MAL`).
3. La tercera prueba da de alta exitosamente como autorizado un cliente cuyo DNI (el 12345678) y cuenta (la cuenta 1) si que existen según el script sql. Como todavía no has implementado la transacción te sale `Caso todo correcto MAL`.
4. Finalmente, repetimos el mismo alta que en el paso 3. Si el paso 3 se ejecutó correctamente ahora debería de saltar la excepción por violación de la clave primaria de la tabla *Cuentas_Clientes* ya comentada, que acarrea la `SQLException_PracticaJDBC2021` número -3. Como todavía no has implementado la transacción te sale `Autorización repetida MAL`.

El mensaje que sale cuando todo funciona1 es:

```
          -- Mensajes de SQL*Plus creando y borrando tablas que omitimos--
ERROR lsi.ubu.PracticaJDBC2021.SQLException_PracticaJDBC2021 <init>   No existe
cliente con ese DNI
Cliente inexistente OK
-------------------------------
ERROR lsi.ubu.PracticaJDBC2021.SQLException_PracticaJDBC2021 <init>   No existe
ese nro de cuenta
corriente
Cuenta inexistente OK
-------------------------------
Caso todo correcto OK
-------------------------------
ERROR lsi.ubu.PracticaJDBC2021.SQLException_PracticaJDBC2021 <init>   Ese cliente
ya había sido
autorizado para esa cuenta
Autorización repetida OK
FIN.............
```