SEGUNDA TAREA OBLIGATORIA - BASES DE DATOS 2
CURSO 2019 - TECNÓLOGO INFORMÁTICO

Objetivo

Desarrollar un Sistema de Información a través de un ejemplo de la vida real, basado en la programación con acceso a base de datos.

Metodología

Para gestionar la base de datos se utilizará el DBMS relacional: “PostgreSQL”.
El desarrollo del sistema se realizará con el lenguaje de programación: “JAVA”.
Se aplicará la metodología de programación de base de datos de biblioteca de funciones utilizando la API “JDBC” y la metodología de procedimientos almacenados.

Sistema a desarrollar

Se debe desarrollar un Sistema para gestionar la información de una Agencia de Viajes, donde deberán definirse las siguientes entidades:

- Empleado.
  Representa a los empleados de la agencia de viajes. Los mismos son identificados por su cédula de identidad y sus datos personales básicos.
  La cédula de identidad no podrá ser utilizada como clave primaria.
- Cliente.
  Representa a los clientes de la agencia. Se identifican de la misma forma que los empleados pero deben almacenarse por separado.
- Aerolínea.
  Corresponde a la empresa encargada del transporte de pasajeros por avión.
- Ciudad.
  Son los posibles puntos de origen y de destino con los que la agencia opera.
- Vuelo.
  Refiere al recorrido de un avión perteneciente a determinada aerolínea desde una ciudad de origen a una ciudad de destino. Además debe tener como atributos la fecha/hora de inicio del mismo y la duración del vuelo medida en horas con formato: “hh:mm”.
- Forma_de_Pago.
  Es la forma en que el cliente paga el viaje adquirido.
- Viaje_Vendido.
  El viaje consiste en el desplazamiento de un determinado pasajero desde una ciudad origen a otra de destino. El viaje desde estar asociado con al menos un vuelo. En caso de tener escalas tendrá que estar asociado a más de un vuelo.
  Además deberá registrarse el empleado que hizo la venta, el cliente comprador del viaje, las fecha de compra del viaje, el monto pagado por el mismo y la forma de pago.
  La fecha/hora de inicio del viaje corresponde a la de su primer vuelo.

Restricciones:

1. Las claves primarias son de tipo numérico auto-numerado.
2. Cumplimiento estricto de las relaciones foráneas de acuerdo al nombre de la entidad, nombres de sus atributos y su relación con otras entidades o atributos.

Consignas

Las consignas de la tarea son las siguientes:

I. A lo largo de la toda la tarea se trabajará con el DBMS instalado localmente en el mismo equipo donde se realizará el desarrollo del sistema.
Instalar el DBMS relacional "PostgreSQL", desde la versión 9.x en adelante.
Luego instalar el entorno para programación con lenguaje Java, mediante el JDK (Java Development Kit).
Además se deberá descargar la librería de la API de "JDBC" provista en el sitio oficial del fabricante del DBMS y luego registrarla en el entorno de desarrollo.

II. En ninguna parte de la tarea se podrá utilizar el usuario de instalación del DBMS. Deberá definirse un solo usuario como "DBA", otros dos usuarios con permisos para "CREATE", otros dos con permisos para "INSERT, UPDATE, DELETE" y otros tres usuarios con permisos para "SELECT".
Tanto la creación de usuarios como la asignación de permisos deberá realizarse mediantes sentencias SQL.
En las siguientes partes de la tarea que requieren ejecución de consultas se deberá justificar la elección del usuario utilizado según el nivel de seguridad necesario para determinada operación (u operaciones).

III. Diseñar las tablas necesarias para almacenar la información de todas las entidades definidas para el sistema.

IV. Desarrollar un programa en lenguaje "java" para cada de una de las siguientes operaciones:

- **A.** Creación de la base de datos
- **B.** Creación de todas las tablas definidas en la parte 3 junto a todas las restricciones necesarias.
- **C.** Ejecutar cuatro sentencias de "INSERT", "DELETE", "UPDATE" y "SELECT".
- **D.** Probar la ejecución de dos "INSERT", "DELETE" y "UPDATE" no permitidos y analizar sus respectivos mensajes de error.
- **E.** Ejecutar desde uno de los usuarios autorizados solo para "SELECT" tres operaciones de modificación y analizar los errores producidos.
- **F.** Ingresar cinco nuevas ventas de viajes y luego eliminar dos de ellas. Dos de los viajes deben poseer dos escalas, otros dos viajes con una sola escala y un solo viaje es un vuelo directo.
  Ejecutar en dos terminales el mismo programa al mismo tiempo y analizar los efectos.

G. Investigar el concepto de transacción.
Realizar mediante transacción la parte “F”, ejecutar en dos terminales el mismo programa al mismo tiempo y comparar con la parte “F”.

H. Mediante bucles del lenguaje de programación sin utilizar transacción realizar en orden las operaciones:

1. Ingresar 200 clientes
2. Ingresar 50 empleados
3. Ingresar 50 aerolíneas
4. Ingresar 100 ciudades
5. Ingresar 15 formas de pago
6. Ingresar 360 viajes vendidos (80 viajes son vuelos directos, 100 poseen una sola escala, 100 tienen dos escalas y 80 poseen tres escalas).
7. Modificar las ciudades de destino y los clientes de la mitad de los viajes.
8. Eliminar la otra mitad de los viajes que no fueron modificados.
9. Medir los tiempos de duración con “java”.

I. Realizar la parte “H” utilizando transacciones.
Medir los tiempos de duración con sentencias de “java” y comparar con parte “H”.

J. Desplegar un listado con todas las filas y atributos de los viajes vendidos que fueron ingresados en las partes anteriores, ordenado por el nombre de la ciudad de origen.
No deben aparecer los valores de las claves primarias, sino los valores de los nombres asignados en toda entidad involucrada en la consulta.
Medir los tiempos de duración con “java”.

K. Investigar concepto de índice, analizar cómo se puede aplicar en la tabla de viajes vendidos y luego agregarlo en el diseño.
El agregado de índice debe hacerse con sentencias SQL desde el mismo programa.

L. Realizar el mismo listado de la parte “J” pero utilizando índices, medir los tiempos y comparar con la parte “J”.

V. Implementar un Store Procedure en el DBMS que realice lo mismo de la parte “I” del ejercicio “IV”.
Invocarlo desde la interfaz de línea de comando del DBMS (“psql”), midiendo los tiempos de ejecución mediante sentencias SQL y comparar con las partes anteriores.

VI. Desarrollar un programa desde “java” que invoque el Store Procedure de la parte “V”, medir los tiempos de duración con sentencias de “java” y comparar con las partes anteriores.

VII. Crear Triggers para auditar las operaciones de “INSERT”, “DELETE” y “UPDATE” de la tabla de viajes vendidos y de otras dos tablas a elección.

VIII. Desarrollar un programa desde “java” que invoque sentencias que verifiquen el funcionamiento de todos los Triggers implementados en la parte “VII”.

La compilación y ejecución de los programas podrá realizarse solamente desde la interfaz de línea de comando.
En el anexo de la letra de la tarea se expone un esqueleto de programa en lenguaje “java” que accede a la base de datos y los comandos para la compilación/ejecución del mismo.

Entregables
La entrega consiste de:

1. El informe principal que documenta la totalidad del trabajo realizado en las ocho partes de la tarea, tanto de texto como de copias de pantalla.
2. Los códigos fuentes de programación de las partes que involucraron código fuente.
3. Los ficheros conteniendo las sentencias SQL (o pgSQL) utilizadas en las partes que no involucraron código fuente.

Modalidad de Entrega
La entrega se realizará por e-mail, enviando un archivo comprimido que contenga todos los artefactos solicitados en cada una de las partes que componen la tarea.

Plazo de Entrega
El plazo de entrega es el lunes 17 de junio a la 23:59 hs.

Consideraciones
El trabajo debe ser realizado por cada uno de los equipos que fueron conformados y publicados en la plataforma a finales del mes de marzo.

Todo tipo de conflicto que pueda surgir con algún integrante, debe ser inmediatamente informado al profesor. No se permitirá que se separen integrantes para formar otro equipo, y en caso de ocurrir no será aceptado lo entregado por los integrantes que se separaron produciéndose la desaprobación automática del trabajo y por ende la pérdida del curso para los mismos.

ANEXOS

Ejemplo de código de creación de base de datos utilizando JDBC:

//
/// Nombre de fichero: “JDBCExample.java”

```
//STEP 1. Import required packages
import java.sql.*;

public class JDBCExample {
  // JDBC driver name and database URL
  static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
  static final String DB_URL = "jdbc:mysql://localhost/";

  //  Database credentials
  static final String USER = "user";
  static final String PASS = "pass";

  public static void main(String[] args) {
    Connection conn = null;
    Statement stmt = null;
    try{
      //STEP 2: Register JDBC driver

      Class.forName("com.mysql.jdbc.Driver");


      //STEP 3: Open a connection
      System.out.println("Connecting to servidor localhost mysql...");
      conn = DriverManager.getConnection(DB_URL, USER, PASS);

      //STEP 4: Execute a query
      System.out.println("Creating database...");
      stmt = conn.createStatement();

      String sql = "CREATE DATABASE Nuevadb3";
      stmt.executeUpdate(sql);

      System.out.println("Database created successfully...");

    }catch(SQLException se){
      //Handle errors for JDBC
      se.printStackTrace();
    }catch(Exception e){
      //Handle errors for Class.forName
      e.printStackTrace();
    }finally{
      //finally block used to close resources
      try{
        if(stmt!=null)
          stmt.close();
      }catch(SQLException se2){
      }// nothing we can do
      try{
        if(conn!=null)
          conn.close();
      }catch(SQLException se){
        se.printStackTrace();
      }//end finally try
    }//end try
    System.out.println("Goodbye!");
  }//end main
}//end JDBCExample
```

//

Ejemplo código de ejecución de sentencias SQL utilizando JDBC:

//

/// Nombre de fichero: “FirstExample.java”

```
import java.sql.*;

public class FirstExample {
  // JDBC driver name and database URL
  static final String JDBC_DRIVER = "org.postgresql.Driver";
  static final String DB_URL = "jdbc:postgresql://localhost:5432/ejer2";

  //  Database credentials
  static final String USER = "postgres";
  static final String PASS = "passj";

  public static void main(String[] args) {
    Connection conn = null;
    Statement stmt = null;
    try{
      //STEP 2: Register JDBC driver

      Class.forName("org.postgresql.Driver");

      //STEP 3: Open a connection
      System.out.println("Connecting to database...");
      conn = DriverManager.getConnection(DB_URL,USER,PASS);

      //STEP 4: Execute a query
      System.out.println("Creating statement...");
      stmt = conn.createStatement();

      String sql;

      sql = "INSERT INTO proveedores values (default,'proveedor1','direccion1');";
      stmt.executeUpdate(sql);

      sql = "Select idp,nombrep,direccion from proveedores";
      ResultSet rs = stmt.executeQuery(sql);

      //STEP 5: Extract data from result set
      while(rs.next()){
        //Retrieve by column name
        int idp  = rs.getInt("idp");
        String nombrep = rs.getString("nombrep");
        String direccion = rs.getString("direccion");

        //Display values
        System.out.print("ID: " + idp);
        System.out.print(", nombrep: " + nombrep);
        System.out.println(", direccion: " + direccion);

      }
      //STEP 6: Clean-up environment

      rs.close();

      sql = "delete from proveedores";
      stmt.executeUpdate(sql);
      stmt.close();
      conn.close();

    }catch(SQLException se){
      //Handle errors for JDBC
      se.printStackTrace();
    }catch(Exception e){
      //Handle errors for Class.forName
      e.printStackTrace();
    }finally{
```

```
      //finally block used to close resources
      try{
         if(stmt!=null)
            stmt.close();
      }catch(SQLException se2){
      }// nothing we can do
      try{
         if(conn!=null)
            conn.close();
      }catch(SQLException se){
         se.printStackTrace();
      }//end finally try
    }//end try
    System.out.println("Goodbye!");
  } //end main

}//end FirstExample
```

//

Compilación en java desde línea de comandos:

java FirstExample.java ///// Compilación
javac FirstExample ///// Ejecución del objeto recién compilado

Ejemplo de registro de la librería en el entorno:

Existen varias formas de registrar la librería, a continuación se muestra una de ellas que consiste en agregarla a las variables de entorno:

export CLASSPATH=/bdatos/prod_acceso_db/postgresql-42.2.5.jre7.jar:$CLASSPATH