package cecs323jdbcproject;

import java.util.\*;

import java.sql.\*;

/\*\*

\*

\* @author Phuc Nguyen and Brett Tomita

\* 3/22/2018

\*/

public class CECS323JDBCProject

{

   static String USER;

   static String PASS;

   static String DBNAME;

   public static Scanner input = new Scanner(System.in);

   static final String JDBC\_DRIVER = "org.apache.derby.jdbc.ClientDriver";

   static String DB\_URL = "jdbc:derby://localhost:1527/JDBC\_Project;user=jdbc;password=jdbc";  //TODO

   /\*\*

    \* @param args : The main program that will execute the 9 options for the program

    \*/

   public static void main(String[] args)

   {

       // TODO code application logic here

       Connection connection = null;

       Statement stmt = null;

       try

       {

           Class.forName("org.apache.derby.jdbc.ClientDriver");

           connection = DriverManager.getConnection(DB\_URL);

           stmt = connection.createStatement();

           boolean choice = false;

           String action = "";

           int actionNum = -1;

           while(!choice)

           {

               try

               {

                   System.out.println("Welcome!\nWhat would you like to do? (Enter the number only)"

                       + "\n1. List all the writing groups"

                       + "\n2. List all the data for a group specified by the user"

                       + "\n3. List all publishers"

                       + "\n4. List all the data for a pubisher specified by the user."

                       + "\n5. List all book titles"

                       + "\n6. List all the data for a book specified by the user. "

                       + "\n7. Insert a new book"

                       + "\n8. Insert a new publisher and update all book published by one publisher to be published by the new pubisher."

                       + "\n9. Remove a book specified by the user"

                       + "\n0. Exit");

                   action = input.next();

                   actionNum = Integer.parseInt(action);

                   if(actionNum < 0 || actionNum > 9)

                       actionNum = -1;

               }

               catch(InputMismatchException i)

               {

                   System.out.println("Invalid input");

               }

               catch(Exception e)

               {

                   System.out.println("Please only enter the number.  Do not include any letters or punctuation\n");

               }

               switch(actionNum)

               {

                   case 1:

                       getAllGroups(stmt);

                       choice = false;

                       break;

                   case 2:

                       getGroupData(stmt);

                       choice = false;

                       break;

                   case 3:

                       getAllPublishers(stmt);

                       choice = false;

                       break;

                   case 4:

                       getPublishers(stmt);

                       choice = false;

                       break;

                   case 5:

                       getAllTitles(stmt);

                       choice = false;

                       break;

                   case 6:

                       getBookData(stmt);

                       choice = false;

                       break;

                   case 7:

                       addBook(stmt);

                       choice = false;

                       break;

                   case 8:

                       updatePublishers(stmt);

                       choice = false;

                       break;

                   case 9:

                       remove(stmt);

                       choice = false;

                       break;

                   case 0:

                       choice = true;

                       break;

                   default:

               }

           }

       }

       catch (SQLException s)

       {

           System.out.println("An error occurred");

       }

       catch (Exception e)

       {

           System.out.println("An error occurred");;

       }

       finally

       {

           try

           {

               if(stmt != null)

                   stmt.close();

               if(connection != null)

                   connection.close();

           }

           catch(SQLException s)

           {

               System.out.println("An error occurred");

           }

       }

   }

   public static String isNull(String input)

   {

       if (input == null || input.length() == 0)

           return "N/A";

       else

           return input;

   }

   public static boolean isListed(ArrayList list, String s)

   {

       for(int x = 0; x<list.size(); x++)

       {

           if(list.get(x).equals(s))

               return true;

       }

       return false;

   }

  /\*\*

  \* Option 1: Print all the writing groups

  \* @param stmt : the statement that will holds the values of the book to be delete

  \* @throws SQLException - catching the exception where user enters invalid input other than numbers

  \*/

   public static void getAllGroups(Statement stmt) throws SQLException

   {

       String sql;

       // SQL statement to print out tall the writting groups

       sql = "SELECT DISTINCT GroupName FROM WRITINGGROUP";

       ResultSet rs = stmt.executeQuery(sql); // execute the sql statement from stmt statement

       String groupName;

       int i = 1;

       while(rs.next())

       {

           groupName = rs.getString("GroupName");

           System.out.println(i + ".  " + groupName);

           i++;

       }

       rs.close(); // close the result set

   }

   /\*\*

    \* Option 2: Print the data of a writing group based on a specified writing group by user

   \* @param stmt : the statement that will holds the values of the book to be delete

   \* @throws SQLException - catching the exception where user enters invalid input

   \*

    \*/

   public static void getGroupData(Statement stmt) throws SQLException

   {

       ArrayList groups = new ArrayList<>();

       String data = "", response = "", name = "";

       // Get GroupName

       groups.add("GroupName");

       data += "Group Name: ";

       //Get Head Writer

           name = "HeadWriter";

           groups.add(name);

           data += " Head Writer: ";

       //Get Year Formed

           name = "yearFormed";

           groups.add(name);

           data += " YearFormed: ";

       //Get subject

       name = "Subject";

       groups.add(name);

       data += " Subject: ";

       name = "PublisherName";

       groups.add(name);

       data += " PublisherName: ";

       name = "PublisherAddress";

       groups.add(name);

       data += " PublisherAddress: ";

       name = "PublisherPhone";

       groups.add(name);

       data += " PublisherPhone: ";

       name = "PublisherEmail";

       groups.add(name);

       data += " PublisherEmail: ";

       name = "BookTitle";

       groups.add(name);

       data += " BookTitle: ";

       name = "YearPublished";

       groups.add(name);

       data += " YearPublished: ";

       name = "NumberPages";

       groups.add(name);

       data += " NumberPages: ";

       String selections = " " + groups.get(0);

       for(int i = 1; i < groups.size(); i++)

           selections += ", " + groups.get(i);

       boolean done = false; // boolean variable for testing

       while(!done)

       {

           try

           {

               System.out.println("What group would you like data on ?");

               getAllGroups(stmt);

               input.nextLine();

               String groupName = input.nextLine();

               // sql statement to get all the data for the book

               String sql = "SELECT \* From Book NATURAL JOIN Publisher Natural Join WritingGroup Where WritingGroup.GroupName = '" + groupName +"'", attribute;

               ResultSet rs = stmt.executeQuery(sql);

               if(rs.next())

                   System.out.println(data);

               while (rs.next())

               {

                 for(int i=0;i<groups.size() - 1 ;i++)

                 {

                     attribute = rs.getString((String) groups.get(i));

                     System.out.print(attribute+"  ,");

                 }

                 attribute = rs.getString((String) groups.get(groups.size() - 1));

                 System.out.print(" " + attribute);

                 System.out.println();

               }

               done = true;

               rs.close();

          }

           catch(Exception e) // catching the errors where user enter invalid information

          {

              System.out.println("An Error Occurred");

          }

       }

   }

   /\*\*

    \* Option 3: List all the publisher name

   \* @param stmt : the statement that will holds the values of the book to be delete

  \* @throws SQLException - catching the exception where user enters invalid input

    \*/

   public static void getAllPublishers(Statement stmt) throws SQLException

   {

       // Sql statement to retrive all the publisher names

       ResultSet rs = stmt.executeQuery("SELECT DISTINCT PublisherName FROM PUBLISHER");

       String pubName;

       int i = 1;

       while(rs.next())

       {

           pubName = rs.getString("PublisherName");

           System.out.println(i + ".  " + pubName);

           i++;

       }

       rs.close();

   }

   /\*\*

    \* Option 4 - print all the data for the specificed publisher

    \* @param stmt

    \*/

   public static void getPublishers(Statement stmt)

   {

       ArrayList groups = new ArrayList<>();

       String data = "", response = "", name = "", values = "";

//        // Get Publisher Name

//        name = "Publishername";

//        groups.add(name);

//        data += "Publisher Name: ";

//

       groups.add("GroupName");

       data += "Group Name: ";

       values += "GroupName, ";

       //Get Head Writer

           name = "HeadWriter";

           groups.add(name);

           data += " Head Writer: ";

           values += name + ", ";

       //Get Year Formed

           name = "yearFormed";

           groups.add(name);

           data += " YearFormed: ";

           values += name + ", ";

       //Get subject

       name = "Subject";

       groups.add(name);

       data += " Subject: ";

       values += name + ", ";

       name = "PublisherName";

       groups.add(name);

       data += " PublisherName: ";

       values += name + ", ";

       name = "PublisherAddress";

       groups.add(name);

       data += " PublisherAddress: ";

       values += name + ", ";

       name = "PublisherPhone";

       groups.add(name);

       data += " PublisherPhone: ";

       values += name + ", ";

       name = "PublisherEmail";

       groups.add(name);

       data += " PublisherEmail: ";

       values += name + ", ";

       name = "BookTitle";

       groups.add(name);

       data += " BookTitle: ";

       values += name + ", ";

       name = "YearPublished";

       groups.add(name);

       data += " YearPublished: ";

       values += name + ", ";

       name = "NumberPages";

       groups.add(name);

       data += " NumberPages: ";

       values += name;

       String selections = " " + groups.get(0);

       for(int i = 1; i < groups.size(); i++)

           selections += ", " + groups.get(i);

       boolean done = false;

       while(!done)

       {

           try

           {

               System.out.println("What Publisher Would You Like Data On?");

               getAllPublishers(stmt);

               input.nextLine();

               String publisher = input.nextLine();

              // "SELECT \* FROM publisher NATURAL JOIN book NATURAL JOIN writinggroup WHERE UPPER(publisherName) LIKE UPPER('%" + publisherName + "%')";

               String sql = "SELECT Distinct \* FROM Publisher Natural Join book Natural Join writinggroup WHERE Publisher.publisherName = '" + publisher + "'", attribute;

               //String sql = "SELECT Distinct \* FROM Publisher, Book, WritingGroup WHERE Publisher.PublisherName = '" + publisher + "'", attribute;

               //String sql = "SELECT Distinct " + values + " FROM Publisher Natural Join Book Natural Join WritingGroup WHERE Publisher.PublisherName = '" + publisher + "'", attribute;

               //String sql = "SELECT \* FROM Book Natural Join Publisher Natural Join WritingGroup WHERE Publisher.PublisherName = '" + publisher + "'", attribute;

               //sSystem.out.println("\nSQL = " + sql + "\n");

               ResultSet rs = stmt.executeQuery(sql);

               if(rs.next())

                   System.out.println(data);

               while (rs.next())

               {

                 for(int i=0;i<groups.size() - 1 ;i++)

                 {

                     attribute = rs.getString((String) groups.get(i));

                     System.out.print(attribute+"  ,");

                 }

                 attribute = rs.getString((String) groups.get(groups.size() - 1));

                 System.out.print(" " + attribute);

                 System.out.println();

               }

               done = true;

               rs.close();

          }catch(Exception e)

          {

              System.out.println("An Error Occurred");

              e.printStackTrace();

          }

       }

   }

   /\*\*

    \* Option 5 - Print all the book titles

    \* @param stmt : the statement that will holds the values of the book to be delete

  \* \* @throws SQLException - catching the exception where user enters invalid input

  \* \*/

   public static void getAllTitles(Statement stmt) throws SQLException

   {

       // SQL function to print all the book title from the book table

       ResultSet rs = stmt.executeQuery("SELECT DISTINCT BOOKTITLE FROM BOOK");

       String title;

       int i = 0;

       while(rs.next())

       {

           title = rs.getString("BOOKTITLE");

           System.out.println(i + ".  " + title);

           i++;

       }

       rs.close();

   }

   /\*\*

    \* Option 6: get all data information for a book

  \* @param stmt : the statement that will holds the values of the book to be delete

    \*/

  public static void getBookData(Statement stmt)

   {

       ArrayList groups = new ArrayList<>();

       String data = "", response = "", name = "";

       // Get Name

               groups.add("GroupName");

       data += "Group Name: ";

       //Get Head Writer

           name = "HeadWriter";

           groups.add(name);

           data += " Head Writer: ";

       //Get Year Formed

           name = "yearFormed";

           groups.add(name);

           data += " YearFormed: ";

       //Get subject

       name = "Subject";

       groups.add(name);

       data += " Subject: ";

       name = "PublisherName";

       groups.add(name);

       data += " PublisherName: ";

       name = "PublisherAddress";

       groups.add(name);

       data += " PublisherAddress: ";

       name = "PublisherPhone";

       groups.add(name);

       data += " PublisherPhone: ";

       name = "PublisherEmail";

       groups.add(name);

       data += " PublisherEmail: ";

       name = "BookTitle";

       groups.add(name);

       data += " BookTitle: ";

       name = "YearPublished";

       groups.add(name);

       data += " YearPublished: ";

       name = "NumberPages";

       groups.add(name);

       data += " NumberPages: ";

       String selections = " " + groups.get(0);

       for(int i = 1; i < groups.size(); i++)

           selections += ", " + groups.get(i);

       try

       {

           System.out.println("What Book Would You Like Data On?");

           getAllTitles(stmt);

           input.nextLine();

           String title = input.nextLine();

           String sql = "SELECT \* FROM Book NATURAL JOIN Publisher NATURAL JOIN WritingGroup WHERE BookTitle = '" + title + "'", attribute;

           //System.out.println(sql);

           ResultSet rs = stmt.executeQuery(sql);

//            if(rs.next())

//                System.out.println(data + "\ngroups.size() = " + groups.size());

           while (rs.next())

           {

                 for(int i=0;i<groups.size()-1;i++)

                 {

                     attribute = rs.getString((String) groups.get(i));

                     System.out.print(attribute+"  ,");

                 }

                 attribute = rs.getString((String) groups.get(groups.size() - 1));

                 System.out.print(attribute);

                 System.out.println();

           }

           rs.close();

      }catch(Exception e)

      {

          System.out.println("An Error Occurred");

      }

   }

   /\*\*

    \* Option 7Insert a book into the system

  \* @param stmt : the statement that will holds the values of the book to be delete

  \* @throws SQLException - catching the exception where user enters invalid input other than numbers

    \*/

   public static void addBook(Statement stmt) throws SQLException

   {

       String title = "", groupName = "", yearPublished = "", pubName = "";

       int numPages = 0;

       try

          // Prompt user to enter the information about the book that will be added

       {

           input.nextLine();

           System.out.println("Enter the title of the book:");

           title = input.nextLine();

           System.out.println("Enter the Year Published:");

           yearPublished = input.next();

           System.out.println("Enter the Number of Pages:");

           numPages = input.nextInt();

           System.out.println("\nchoose one of these registered group names");

           getAllGroups(stmt); // print all group for the user to choose

           System.out.println("\nEnter the Group Name:");

           input.nextLine();

           groupName = input.nextLine();

           getAllPublishers(stmt); // print all publisher for the user to choose

           System.out.println("\nChoose Publisher Name:");

           pubName = input.nextLine();

       }

       catch (InputMismatchException i)

       {

           System.out.println("Invalid input");

       }

       catch(Exception e)

       {

           System.out.println("An Error Occurred");

       }

       try

       {

           String sql = "INSERT INTO Book VALUES ('" + title + "', '" + yearPublished + "'," + numPages + ", '" + groupName +"', '"+ pubName +"') ";

           stmt.executeUpdate(sql);

           System.out.println("New book have been inserted");

       }

       catch(SQLException s)

       {

           System.out.println("There was a problem.  Please try again");

       }

       catch(Exception e)

       {

           System.out.println("There was a problem.  Please try again");

       }

   }

   /\*\*.

    \* Option 8 - Update an old publisher name and data with the new name

    \* @param stmt - statement to hold the SQL info after passing in values

    \* @throws SQLException : if the entered publisher name or writing group is not found

    \* numberException: If statement to check if the entered numbers is actually numbers, not letters

    \*/

   public static void updatePublishers(Statement stmt) throws SQLException

   {

       input.nextLine();

       try{

           // asking for user to enter the new publiser name

           System.out.println("What is the publisher's name to be added: ");

           String name = input.nextLine();

           // asking for the new publisher's address

           System.out.println("What is the publisher's address: ");

           String address = input.nextLine();

           boolean validPhone = false;

           String phone = "";

           // to check if the phone is valid

           while(!validPhone)

           {

               System.out.print("What is the publisher phone: ");

               phone = input.nextLine();

               //Regex to validate phone number

               String pattern = "\\d{10}|(?:\\d{3}-){2}\\d{4}|\\(\\d{3}\\)\\d{3}-?\\d{4}";

               if(phone.matches(pattern))

                   validPhone = true;

               else

                   System.out.println("Invalid Phone Number");

           }

           System.out.println("What is the publisher email: ");

           String email = input.nextLine();

           // try statement to catch the SQL and Invalid input excptions later

           try

           {

                String sql = "INSERT INTO PUBLISHER VALUES('"+name+"' , '"+address+"' , '"+phone+"' , '"+email+"')";

                // passing the value of SQL into stmt(statement)

                stmt.executeUpdate(sql);

                System.out.println("Should the books published by one Publisher now be published by the new Publisher? (Y/N)");

                String choice = input.next(), newPublisher = "";

                // allow lower case and upper case

                if(choice.equalsIgnoreCase("Y"))

                {

                    input.nextLine();

                    // recall the 3rd option to print out all the publisher name for user to choose which one will be updated

                   getAllPublishers(stmt);

                   System.out.println("What is the name of the publisher that will be updated: ");

                   newPublisher = input.nextLine();

                   String update = "update BOOK set publishername= '"+name+"' where publishername= '"+ newPublisher+"'";

                   stmt.execute(update);

                   System.out.println("Would you like to delete the old publisher? (Y/N)");

                   String answer = input.next();

                   if(answer.equalsIgnoreCase("Y"))

                   {

                       sql = "DELETE from Publisher WHERE publisherName = '" + newPublisher + "'";

                       stmt.executeUpdate(sql);

                   }

                }

           } // catching the excpetion if user enters info that was not there

           catch(SQLException e)

           {

               System.out.println("An Error Occured");

           }

       }

       catch(InputMismatchException e)

       {

           System.out.println("Invalid Input");

           updatePublishers(stmt);

       }

       catch(Exception e)

       {

           System.out.println("An Error Occured");

           updatePublishers(stmt);

       }

   }

 /\*\*

  \* Option 9: remove a book base on the user specific choices

  \* @param stmt : the statement that will holds the values of the book to be delete

  \* @throws SQLException - catching the exception where user enters invalid input that was not added to the table

  \* We added 5 option for user to delete: by book, year published, number of pages or pub name

  \* Also catch exception for invalidNumberForms and invalid input

  \*/

   public static void remove(Statement stmt) throws SQLException

   {

       System.out.println("1. Delete by Book title"

                         +"\n2. Delete by Year Published"

                         +"\n3. Delete by Number of Pages"

                         +"\n4. Delete by Group Name"

                         +"\n5. Delete by Publisher Name");

       int choice = input.nextInt();

       String attribute = null;

       String delete = null;

       int pageDelete = 0;

       input.nextLine();

       switch(choice)

       {

           // 5 options cases : this is just to retetrieve the information that will be delete

           // base on the user input comparing to the information in the table

           case 1:

               attribute = "BOOKTITLE";

               //show all the titles that can be deleted

               getAllTitles(stmt);

               System.out.println("Enter the title of the book you would like to delete");

               delete = input.nextLine();

               break;

           case 2:

               attribute = "YEARPUBLISHED";

               System.out.println("Enter the year published of the book you would like to delete");

               delete = input.nextLine();

               break;

           case 3:

               attribute = "NUMBERPAGES";

               System.out.println("Enter the year number of pages of the book you would like to delete");

               pageDelete = input.nextInt();

               break;

           case 4:

               attribute = "GROUPNAME";

               System.out.println("\nchoose one of these registered group names");

               getAllGroups(stmt);

               System.out.println("Enter the year name of the group of the book you would like to delete");

               delete = input.nextLine();

               break;

           case 5:

               attribute = "PUBLISHERNAME";

               System.out.println("\nchoose one of these registered group names");

               getAllPublishers(stmt);

               System.out.println("Enter the year publisher's name of the book you would like to delete");

               delete = input.nextLine();

               break;

           default:

               System.out.println("Invalid chose.");

               break;

       }

       // Checking if the input is not null, else catch expcetion

       if (attribute != null)

       {

           try

           {

               if (delete != null)

               {

                   // DELETION SATE WHERE IT DELETE FROM THE TABLES

                   String sql = "Delete from Book where "+attribute+"='"+delete+"'";

                   stmt.executeUpdate(sql);

                   System.out.println("Book have been deleted");

               }

               else

               {

                   String sql = "Delete from Book where "+attribute+"="+pageDelete;

                   stmt.executeUpdate(sql);

               }

           }

           catch(InputMismatchException e)

           {

               System.out.println("Invalid input");

           }

           catch(SQLException s)

           {

               System.out.println("There was an error.  Please try again");

           }

       }

   }

}