import java.sql.\*;

import java.util.Scanner;

/\*\*

\* Kyle Pamintuan

\* March 14, 2016

\* CECS 323

\* Professor Dave Brown

\*/

public class JDBC\_Project

{

//This is the specification for the printout that I'm doing:

//each % denotes the start of a new field.

//The - denotes left justification.

//The number indicates how wide to make the field.

//The "s" denotes that it's a string.

static String displayFormat;

// JDBC driver name and database URL

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

static String DB\_URL = "jdbc:derby://localhost:1527/JDBC Project Database;user=APP;password=008830924";

public static void main(String[] args)

{

Connection conn = null; //initialize the connection

Statement stmt = null; //initialize the statement that we're using

//user input scanners

Scanner in = new Scanner(System.in);

Scanner in2 = new Scanner(System.in);

Scanner in3 = new Scanner(System.in);

Scanner in4 = new Scanner(System.in);

//Variables associated with SQL

String sql;

PreparedStatement pstmt;

ResultSet rs;

//Prompt the user for a function

System.out.println("Choose a Function:");

System.out.println("1. List all album titles");

System.out.println("2. List all data for a particular album");

System.out.println("3. Insert a new album");

System.out.println("4. Insert a new studio");

System.out.println("5. Remove a particular album");

System.out.println();

int function = in.nextInt();

System.out.println();

//Execute the function chosen

if (function == 1)

{

//Prints all album titles -------------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT album\_title FROM albums";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-20s\n";

System.out.println("===== Albums =====");

System.out.printf(displayFormat, "Album Title");

while (rs.next())

{

//Retrieve by column name

String albumTitle = rs.getString("album\_title");

//Display values

System.out.printf(displayFormat, albumTitle);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

}

else if (function == 2)

{

//Get album title from user

System.out.println("Which album?");

//Prints all album titles -------------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT album\_title FROM albums";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-20s\n";

System.out.println("===== Albums =====");

System.out.printf(displayFormat, "Album Title");

while (rs.next())

{

//Retrieve by column name

String albumTitle = rs.getString("album\_title");

//Display values

System.out.printf(displayFormat, albumTitle);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

//Get user input

System.out.println();

String album = in2.nextLine();

System.out.println();

//Prints all info from album specified by user --------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

System.out.println("Creating statement...");

sql = "SELECT \* FROM albums INNER JOIN recording\_groups USING(group\_name) INNER JOIN recording\_studios USING(studio\_name) WHERE album\_title = '" + album + "'";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-30s%-20s%-20s%-20s%-30s%-20s%-20s%-20s%-30s%-20s%-20s%-20s\n";

System.out.println("===== Albums, Recording Groups, and Recording Studios =====");

System.out.printf(displayFormat, "Album Title", "Date Recorded", "Length (mins)", "Number Of Songs", "Group Name", "Lead Singer", "Year Formed", "Genre", "Studio Name", "Studio Address", "Studio Owner", "Studio Phone");

while (rs.next())

{

//Retrieve by column name

String albumTitle = rs.getString("album\_title");

String groupName = rs.getString("group\_name");

String studioName = rs.getString("studio\_name");

Date dateRecorded = rs.getDate("date\_recorded");

int length = rs.getInt("length");

int numberOfSongs = rs.getInt("number\_of\_songs");

String leadSinger = rs.getString("lead\_singer");

int yearFormed = rs.getInt("year\_formed");

String genre = rs.getString("genre");

String studioAddress = rs.getString("studio\_address");

String studioOwner = rs.getString("studio\_owner");

String studioPhone = rs.getString("studio\_phone");

//Display values

System.out.printf(displayFormat, albumTitle, dateRecorded, length, numberOfSongs, groupName, leadSinger, yearFormed, genre, studioName, studioAddress, studioOwner, studioPhone);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

}

else if (function == 3)

{

//Get album information from user

System.out.println("Album Title");

String title = in2.nextLine();

System.out.println();

System.out.println("Group Name");

String group = in2.nextLine();

int count = 0;

System.out.println();

//If the group name that the user inputed is not in the table, show them the list of groups so that they may choose a group name/studio name

//Group Name and Studio Name must be paired consistently because of pk/fk constraints

//Find the count of how many instances of the group name that the user inputed... ----------------------------------------------------------------------------------

//...is in the albums table. If the count is > 0, then it exits. If the count = 0, it does exist.

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT COUNT(group\_name) AS Count FROM recording\_groups WHERE group\_name = ?";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, group);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

while (rs.next())

{

//Retrieve by column name

count = rs.getInt("Count");

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

while(count == 0) //While the group name that the user inputs does NOT exist, give them a list of existing group names

{

count = 1;

System.out.println("ERROR: That group does not exist");

System.out.println("Enter a group name from the the table below:");

//Print all group names from the recording\_groups table ---------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT group\_name FROM recording\_groups";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-20s\n";

System.out.printf(displayFormat, "Group Names");

while (rs.next())

{

//Retrieve by column name

String groupName = rs.getString("group\_name");

//Display values

System.out.printf(displayFormat, groupName);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

//Get the group name that the user inputed again

System.out.println();

group = in2.nextLine();

}

System.out.println();

System.out.println("Studio Name");

String studio = in2.nextLine();

int count2 = 0;

System.out.println();

//Find the count of how many instances of the studio name that the user inputed... ----------------------------------------------------------------------------------

//...is in the recording\_studios table. If the count is > 0, then it exits. If the count = 0, it does exist.

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT COUNT(studio\_name) AS Count FROM recording\_studios WHERE studio\_name = ?";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, studio);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

while (rs.next())

{

//Retrieve by column name

count2 = rs.getInt("Count");

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

while(count2 == 0) //While that sudio does NOT exist, give the user the list of studio names that DO exist

{

count2 = 1;

System.out.println("ERROR: That studio does not exist");

System.out.println("Enter a studio name from the the table below:");

//Print all group names from the recording\_groups table ---------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT studio\_name FROM recording\_studios";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-20s\n";

System.out.printf(displayFormat, "Studio Names");

while (rs.next())

{

//Retrieve by column name

String studioName = rs.getString("studio\_name");

//Display values

System.out.printf(displayFormat, studioName);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

//Get the name of the studio that the user inputed again

System.out.println();

studio = in2.nextLine();

}

System.out.println();

System.out.println("Date Recorded (Year-Month-Day)");

String date = in2.nextLine();

System.out.println();

System.out.println("Length (mins)");

int length = in2.nextInt();

System.out.println();

System.out.println("Number Of Songs");

int num = in2.nextInt();

System.out.println();

//Inserts new album info into the albums table --------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

System.out.println("Creating statement...");

sql = "INSERT INTO albums VALUES ('" + title + "','" + group + "','" + studio + "','" + date + "'," + length + "," + num + ")";

pstmt = conn.prepareStatement(sql);

pstmt.executeUpdate();

//STEP 6: Clean-up environment

pstmt.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();

//Prints all info from the albums table ------------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT \* FROM albums";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-30s%-30s%-30s%-20s%-20s%-20s\n";

System.out.println("===== Albums =====");

System.out.printf(displayFormat, "Album Title", "Group Name", "Studio Name", "Date Recorded", "Length(mins)", "Number Of Songs");

while (rs.next())

{

//Retrieve by column name

String albumTitle = rs.getString("album\_title");

String groupName = rs.getString("group\_name");

String studioName = rs.getString("studio\_name");

String dateRecorded = rs.getString("date\_recorded");

int length2 = rs.getInt("length");

int numberOfSongs = rs.getInt("number\_of\_songs");

//Display values

System.out.printf(displayFormat, albumTitle, groupName, studioName, dateRecorded, length2, numberOfSongs);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

}

else if (function == 4)

{

//Get studio information from user

System.out.println("Studio Name");

String studio = in2.nextLine();

System.out.println();

System.out.println("Studio Address");

String address = in2.nextLine();

System.out.println();

System.out.println("Studio Owner");

String owner = in2.nextLine();

System.out.println();

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

String phone = in2.nextLine();

//Inserts new studio info into the recording\_studios table --------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

System.out.println("Creating statements...");

sql = "INSERT INTO recording\_studios VALUES ('" + studio + "','" + address + "','" + owner + "','" + phone + "')";

pstmt = conn.prepareStatement(sql);

pstmt.executeUpdate();

//STEP 6: Clean-up environment

pstmt.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();

//Ask user which studio they want to replace with new studio

System.out.println("Now, type the name of the studio (associated with albums) you wish to replace with the new studio");

String studio2 = "";

//Print all group names from the recording\_groups table ---------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT DISTINCT studio\_name FROM albums";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-20s\n";

System.out.printf(displayFormat, "Studio Names");

while (rs.next())

{

//Retrieve by column name

String studioName = rs.getString("studio\_name");

//Display values

System.out.printf(displayFormat, studioName);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

//Get the studio name from user

System.out.println();

studio2 = in2.nextLine();

System.out.println();

//Update new studio info into the albums table --------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

System.out.println("Creating statements...");

sql = "UPDATE albums SET studio\_name = '" + studio + "' WHERE studio\_name = '" + studio2 + "'";

pstmt = conn.prepareStatement(sql);

pstmt.executeUpdate();

//STEP 6: Clean-up environment

pstmt.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();

//Prints all info from recording studios --------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT \* FROM recording\_studios";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-30s%-30s%-30s%-30s\n";

System.out.println("===== Recording Studios =====");

System.out.printf(displayFormat, "Studio Name", "Studio Address", "Studio Owner", "Studio Phone Number");

while (rs.next())

{

//Retrieve by column name

String studioName = rs.getString("studio\_name");

String studioAddress = rs.getString("studio\_address");

String studioOwner = rs.getString("studio\_owner");

String studioPhone = rs.getString("studio\_phone");

//Display values

System.out.printf(displayFormat, studioName, studioAddress, studioOwner, studioPhone);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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();

//Prints info fromt the albums table --------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT \* FROM albums";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-30s%-30s%-30s%-20s%-20s%-20s\n";

System.out.println("===== Albums =====");

System.out.printf(displayFormat, "Album Title", "Group Name", "Studio Name", "Date Recorded", "Length(mins)", "Number Of Songs");

while (rs.next())

{

//Retrieve by column name

String albumTitle = rs.getString("album\_title");

String groupName = rs.getString("group\_name");

String studioName = rs.getString("studio\_name");

String dateRecorded = rs.getString("date\_recorded");

int length = rs.getInt("length");

int numberOfSongs = rs.getInt("number\_of\_songs");

//Display values

System.out.printf(displayFormat, albumTitle, groupName, studioName, dateRecorded, length, numberOfSongs);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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 --------------------------------------------------------------------------------------------------

}

else if (function == 5)

{

//Ask user which album info to delete

System.out.println("Which album do you want to remove?");

System.out.println();

//Prints all album titles -------------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT album\_title FROM albums";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-20s\n";

System.out.println("===== Albums =====");

System.out.printf(displayFormat, "Album Title");

while (rs.next())

{

//Retrieve by column name

String albumTitle = rs.getString("album\_title");

//Display values

System.out.printf(displayFormat, albumTitle);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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();

String album = in2.nextLine();

//Delete specified album from the albums table --------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

System.out.println("Creating statements...");

sql = "DELETE FROM albums WHERE album\_title = '" + album + "'";

pstmt = conn.prepareStatement(sql);

pstmt.executeUpdate();

//STEP 6: Clean-up environment

//rs.close();

pstmt.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();

//Prints all info from the albums table --------------------------------------------------------------------------------

try

{

//STEP 2: Register JDBC driver

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

//STEP 3: Open a connection

conn = DriverManager.getConnection(DB\_URL);

//STEP 4: Execute a query

sql = "SELECT \* FROM albums";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

System.out.println();

//STEP 5: Extract data from result set

displayFormat ="%-30s%-30s%-30s%-20s%-20s%-20s\n";

System.out.println("===== Albums =====");

System.out.printf(displayFormat, "Album Title", "Group Name", "Studio Name", "Date Recorded", "Length(mins)", "Number Of Songs");

while (rs.next())

{

//Retrieve by column name

String albumTitle = rs.getString("album\_title");

String groupName = rs.getString("group\_name");

String studioName = rs.getString("studio\_name");

String dateRecorded = rs.getString("date\_recorded");

int length2 = rs.getInt("length");

int numberOfSongs = rs.getInt("number\_of\_songs");

//Display values

System.out.printf(displayFormat, albumTitle, groupName, studioName, dateRecorded, length2, numberOfSongs);

}

//STEP 6: Clean-up environment

rs.close();

pstmt.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();

System.out.println();

}//end main

}//end FirstExample}