CS157A

Ahmed Ezzat

Project Report-Books Database

Team members: Petr Khromov, Eang Heab, Ankit Gandhi, Laura Gandhi, Riadiani Marcelita

12/6/2016

**Project Report**

This project was created for our CS157A (Database Management Systems) class. It is called the Books database, and deals mainly with making a database dealing with books. This database also handles the creation of tables related to books, the population of those tables with data, and executions of sample query statements.

On the first part of our project, we try to establish a connection to MySQL server through our Java source code. We do so by opening a connection with localhost, using port 3306 and username “root.” Our program also needs to detect the JDBC driver installed in our IDE, or whichever machine we ran the Java file from. In our case, we used Eclipse to run the Java code. Once we have established this connection, we will then be able to continue with the next step, which is creating the database.

We create the Books database by executing the statement, “CREATE DATABASE Books.” To ensure no duplication of the database creation, we initially drop the database if it exists. Once the database is created successfully, we will receive a success message, and we can proceed to the next step.

The next step is the table creations. For this project, we created four tables:

1. **authors**: table to handle authors’ information in the database. Each author has a unique authorID, and their informations are stored as their first and last names.
2. **authorsISBN**: table to handle the information of the authors’ unique ID numbers. The authors’ ID numbers are the same numbers that appears in the authors table.
3. **publishers**: table to take care of the publishers’ information. It contains the publishers’ names and publishers’ unique IDs.
4. **titles**: table to handle the book titles published. Book titles have unique ISBN, year they are published, publisherID, and price.

Once the tables are created successfully we will have a success message, and we will then be able to check if the tables were created by using the “show tables” operation/statement through MySQL. In our case, we check the existence of the tables using MySQL Workbench. A screenshot of our created tables is included in a separate printout.

The next step is to populate the table with data. We gathered data from an outer website and stored them in a .csv file. We then import the data from the file into a temporary array in our code, and we use those stored values to populate the tables with data. Now, each of our tables are populated with at least twenty-thirty lines of data.

After populating the tables, the next step is to execute some sample query statements as were stated in the project description:

1. Select all authors from the authors table. Order the information alphabetically by the author’s last name and first name.
2. Select all publishers from the publishers table.
3. Select a specific publisher and list all books published by that publisher. Include the title, year, and ISBN number. Order the information alphabetically by title.
4. Add new author.
5. Edit/update the existing information about an author.
6. Add a new title for an author.
7. Add a new publisher.
8. Edit/update the existing information about a publisher.

We executed each query statements with sample data, and all of them are run successfully. We prove this by checking the tables that we modified, and including screenshots of our query statement results on a separate printout.

Finally, after all the query statements are executed successfully, we then close the server connections and statements, and that is the end of our project.

JDBC

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import java.sql.\*;

import java.util.Scanner;

public class BookDatabase {

public static Connection conn = null;

public static Statement stmt = null;

// JDBC driver name and database URL

static final String JDBC\_DRIVER = "com.mysql.jdbc.Driver";

static final String DB\_URL = "jdbc:mysql://127.0.0.1";

// Database credentials

static final String USER = "root";

static final String PASS = "123456";

/\*\*

\* Creates a connection instance for connecting to DB

\*/

public BookDatabase() {

try {

Class.forName(JDBC\_DRIVER).newInstance();

} catch (Exception ex) {

System.out.println("JDBC driver not found");

}

try {

System.out.println("Establishing connection to database...");

conn = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306", USER, PASS);

stmt = conn.createStatement();

stmt.executeUpdate("USE books");

} catch (SQLException ex) {

System.out.println("SQL exeption thrown");

}

System.out.println("Connected to database!");

}

/\*\*

\* Initializes books schema

\*/

public void createSchema() {

String dropSchema = "DROP SCHEMA IF EXISTS books";

String schema = "CREATE SCHEMA books";

String select = "USE books";

try {

stmt.executeUpdate(dropSchema);

int result = stmt.executeUpdate(schema);

stmt.executeUpdate(select);

if (result == 1) {

System.out.println("books Schema created successfully");

} else {

System.out.println("Failed to initilize schema!");

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

/\*\*

\* Initialize all the tables in books database

\*/

public void createTables() {

String authors = "CREATE TABLE IF NOT EXISTS authors(" + "authorID INTEGER NOT NULL AUTO\_INCREMENT,"

+ "firstName CHAR(20) NOT NULL," + "lastName CHAR(20) NOT NULL," + "PRIMARY KEY(authorID)" + ")";

String publishers = "CREATE TABLE publishers " + " (publisherID INTEGER NOT NULL auto\_increment, "

+ " publisherName CHAR(100) NOT NULL, " + " PRIMARY KEY ( publisherID ))";

String titles = "CREATE TABLE titles " + " (isbn CHAR(10) NOT NULL, " + " title VARCHAR(2500) NOT NULL, "

+ " editionNumber INTEGER NOT NULL, " + " year CHAR(4) NOT NULL, " + " publisherID INTEGER NOT NULL, "

+ " price FLOAT NOT NULL, " + " PRIMARY KEY ( isbn ), "

+ " FOREIGN KEY ( publisherID ) REFERENCES publishers( publisherID ))";

String authorISBN = "CREATE TABLE IF NOT EXISTS authorISBN(" + "authorID INTEGER NOT NULL AUTO\_INCREMENT,"

+ "FOREIGN KEY (authorID) REFERENCES authors(authorID) " + "ON DELETE CASCADE " + "ON UPDATE CASCADE,"

+ "isbn CHAR(10) NOT NULL," + "FOREIGN KEY (isbn) REFERENCES titles(isbn) " + "ON DELETE CASCADE "

+ "ON UPDATE CASCADE" + ")";

try {

System.out.println("Creating tables in database....");

int result1 = stmt.executeUpdate(authors);

int result2 = stmt.executeUpdate(publishers);

int result3 = stmt.executeUpdate(titles);

int result4 = stmt.executeUpdate(authorISBN);

if (result1 == 0 && result2 == 0 && result3 == 0 && result4 == 0) {

System.out.println("All Tables initialized successfully in database!");

} else {

System.out.println("Failed to initilize table data!");

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public void populateData() throws SQLException, IOException {

FileReader file = null;

FileReader file1 = null;

FileReader file2 = null;

FileReader file3 = null;

Scanner in = null;

Scanner in1 = null;

Scanner in2 = null;

Scanner in3 = null;

try {

System.out.println("Populating authors table...");

file = new FileReader("authors.csv");

in = new Scanner(file);

in.nextLine();

while (in.hasNextLine()) {

String[] temp = in.nextLine().trim().split(",");

String insert = "INSERT INTO authors (authorID, firstName, lastName)" + "VALUES (" + temp[0] + ", '"

+ temp[1] + "', '" + temp[2] + "' )";

stmt.executeUpdate(insert);

}

/////////////////////////////////////////////////////////////

System.out.println("Populating publishers table...");

file2 = new FileReader("publishers.csv");

in2 = new Scanner(file2);

in2.nextLine();

while (in2.hasNextLine()) {

String[] temp = in2.nextLine().trim().split(",");

String insert = "INSERT INTO publishers " + "VALUES (" + temp[0] + ", '" + temp[1] + "' )";

stmt.executeUpdate(insert);

}

/////////////////////////////////////////////////////////////

System.out.println("Populating titles table...");

file1 = new FileReader("titles.csv");

in1 = new Scanner(file1);

in1.nextLine();

while (in1.hasNextLine()) {

String[] temp = in1.nextLine().trim().split(",");

String insert = "INSERT INTO titles " + "VALUES (" + temp[0] + ", '" + temp[1] + "', '" + temp[2]

+ "', '" + temp[3] + "', '" + temp[4] + "', '" + temp[5] + "' )";

stmt.executeUpdate(insert);

}

/////////////////////////////////////////////////////////////

System.out.println("Populating authorISBN table...");

file3 = new FileReader("authorISBN.csv");

in3 = new Scanner(file3);

in3.nextLine();

while (in3.hasNextLine()) {

String[] temp = in3.nextLine().trim().split(",");

String insert = "INSERT INTO authorISBN " + "VALUES (" + temp[0] + ", '" + temp[1] + "' )";

stmt.executeUpdate(insert);

}

} catch (FileNotFoundException e) {

System.err.println("Caught FileNotFoundException: " + e.getMessage());

} catch (Exception e) {

System.err.println("Caught Exception: " + e.getMessage());

} finally {

// Closing scanner and file instances

in.close();

in1.close();

in2.close();

in3.close();

file.close();

file1.close();

file2.close();

file3.close();

}

}

public void runQueries() throws SQLException {

//////////////////////////////////////////////////////////////////////////////////////////////////

/\*

\* q1, get author last name and first name by alphabetical order

\*

\* @param first name

\*

\* @param last name

\*/

String q1 = "SELECT \* from authors ORDER BY lastName, firstName ";

ResultSet authorName = stmt.executeQuery(q1);

System.out.println("Select all author in table alphabetically by last name, first name: ");

System.out.printf("%-20s%-20s%n", "Last Name", "First Name");

while (authorName.next()) {

// Retrieve by column name

String first = authorName.getString("firstName");

String last = authorName.getString("lastName");

// Display values

System.out.printf("%-20s%-20s%n", last, first);

}

System.out.println("");

///////////////////////////////////////////////////////////////////////////////////////////

/\*

\* q2, get all publisher

\*

\* @param publisherID

\*

\* @param publisherName

\*/

String q2 = "SELECT \* FROM publishers";

ResultSet publisher = stmt.executeQuery(q2);

System.out.println("Select all publishers from the publishers table");

System.out.printf("%-20s%-20s%n", "publisherID", "Publisher Name");

while (publisher.next()) {

// Retrieve by column name

int publisherID = publisher.getInt("publisherID");

String publisherName = publisher.getString("publisherName");

// Display values

System.out.printf("%-20s%-20s%n", publisherID, publisherName);

}

System.out.println("");

////////////////////////////////////////////////////////////////////////////////////////////////

/\*

\* q3, select from specific publisher and list all books published by

\* that publisher, order the books by title.

\*

\* @param title

\*

\* @param copyright

\*

\* @param isbn

\*/

String q3 = "SELECT title, year,isbn FROM titles, publishers "

+ "WHERE titles.publisherID = publishers.publisherID "

+ "AND publishers.publisherName = 'Kyla Gentry' ORDER BY title;";

ResultSet publisherLookUp = stmt.executeQuery(q3);

System.out.println("Please Select a Publisher ");

System.out.println("Kyla Gentry");

System.out.println("The following books are from publisher Kyla Gentry: \n");

System.out.printf("%-20s%-20s%-20s%n", "Title", "Year", "ISBN");

while (publisherLookUp.next()) {

// Retrieve by column name

String bookTitle = publisherLookUp.getString("title");

long isbn = publisherLookUp.getLong("isbn");

int year = publisherLookUp.getInt("year");

System.out.printf("%-20s%-20s%-20s%n", bookTitle, year, isbn);

}

System.out.println("");

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/\*

\* q4, INSERT NEW AUTHOR

\*

\* @param first name

\*

\* @param last name

\*

\* @param authorID, value set as default , auto incremented in data base

\*/

System.out.println("Insert New Author: Ankit Gandhi ");

String q4 = "insert into authors (firstName, lastName) values ('Ankit', 'Gandhi')";

int r = stmt.executeUpdate(q4);

if (r == 1) {

System.out.println("Insert completed\n");

} else {

System.out.println("Failed to insert new author");

}

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/\*

\* q5, EDIT/UPDATE THE EXISITNG INFORMATION ABOUT AN AUTHOR

\*

\* @param last name

\*

\* @param first name Update

\*

\*/

System.out.println("Updating authors Charissa Garrett to Charissa Gandhi");

String q5 = "UPDATE authors SET firstName = 'Charissa' , " + "lastName = 'Gandhi' WHERE authorID= '1'";

int r1 = stmt.executeUpdate(q5);

if (r1 == 1) {

System.out.println("Updated ");

} else {

System.out.println("Failed to update author");

}

/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/\*

\* q6, ADD A NEW TITLE FOR AN AUTHOR

\*

\* @PARAM title

\*

\* @PARAM isbn

\*

\* @PARAM publisher

\*

\* @PARAM publisherID

\*

\* @PARAM price

\*

\* @param eidtionNumber

\*

\* @param copyright

\*

\*/

// String q11 = "Insert into authors (firstName, lastName) values

// ('Yoko', 'Moana')";

String q12 = "Insert into publishers values (1936346, 'Yoko Moana')";

String q6 = "INSERT INTO titles(isbn, title, editionNumber, year, publisherID, price)"

+ "VALUES('345454545','Advance Programming In C', 2, '2011','1936346',22.3)";

// String q7 = "INSERT INTO authorISBN (isbn) VALUES

// ('345454545')";

// stmt.executeUpdate(q11);

stmt.executeUpdate(q12);

stmt.executeUpdate(q6);

// stmt.executeUpdate(q7);

System.out.println("Insert Completed\n");

//////////////////////////////////////////////////////////////////////////////////////////////////////

/\*

\* q7, ADD A NEW PUBLISHER

\*

\* @PARAM publisherID

\*

\* @PARAM publisherName

\*/

System.out.println("Insert publisher : EANG HEAB ");

String q8 = "insert into publishers (publisherID, publisherName) values ('9567144', 'EANG HEAB')";

stmt.executeUpdate(q8);

System.out.println("Insert completed\n");

//////////////////////////////////////////////////////////////////////////////////////////////////////////

/\*

\* q8, EDIT/UPDATE NEW PUBLISHER

\*

\* @PARAM publisherID

\*

\* @PARAM publisherName

\*/

System.out.println("Updating publisher Kylan Becker to Newton Tech");

String q9 = "UPDATE publishers SET publisherName='Newton Tech' WHERE publisherID = '1936116'";

stmt.executeUpdate(q9);

System.out.println("Updated ");

/////////////////////////////////////////////////////////////////////////////////////////////////////////

// STEP 6: Clean-up environment

authorName.close();

publisher.close();

publisherLookUp.close();

}

public static void main(String[] args) throws SQLException, IOException {

BookDatabase bd = new BookDatabase();

bd.createSchema();

bd.createTables();

bd.populateData();

bd.runQueries();

conn.close();

System.out.println("Have a good one!");

}

}

Snap shots of MYSQL workbench

