Report on Project 1: Books Database

I. Assignment Overview

The current project implements the basic functionalities of MySql database and SELECT statements for storing books. The focus is on implementing a book database, populating it, and then executing different SQL statements to query or manipulate the Books database. I used MySQL Workbench to be a graphical tool for working with MySQL servers and the Book database.

II. Technical Impression

The current project consists of Book.java that considers the following aspects:

(1) Create the Books database tables as specified in the schema definition in the assignment (see *createTable()* method).
(2) If tables for the Books database already exist, drop them all before creating (see *dropTable()* method);
(3) Initialize the four tables (at least 15 entries per table) with no NULL data for each field;
(4) Sample data file is available (see BooksDataNEW.txt) for populating the database.

III. How to run the project

To run the Book.java, please include the classpath of the mysql connector jar before typing Book.java, and then add the following four arguments: <File pathname> <The connection URL for the mysql database> <mysql username> <mysql password>
The 1st argument is to put the file path of BookDataNEW.txt;
The 2nd argument is the connection URL for the mysql database;
The 3rd argument is your username of the mysql database in order to connect to the MySQL server;
The 4th argument is your password of the mysql database
For example, in your command line interface, type the following:

```
java -classpath ~/Downloads/mysql-connector-java-8.0.19.jar Book.java
src/BookDataNEW.txt jdbc:mysql://localhost:3306/sys root abc123
```

I have included the connector jar in the same directory of Book.java. Please specify the pathname of the jar accordingly.

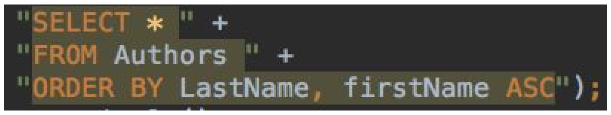
IV. Result of SQL Query

Eight SQL queries are executed. Results of queries 1 - 3 are straightforward because the results need to be printed out. For queries 4 - 8 which are editing or updating the database, additional sql queries are created and executed in order to check if the queries 4 - 8 are executed as desired For example, query 4 adds a new author. In order to check if the author is inserted correctly, additional SQL query is implemented (as seen in insertAuthorOK() method).

SQL syntax for the desired queries and the corresponding results are shown as follows:

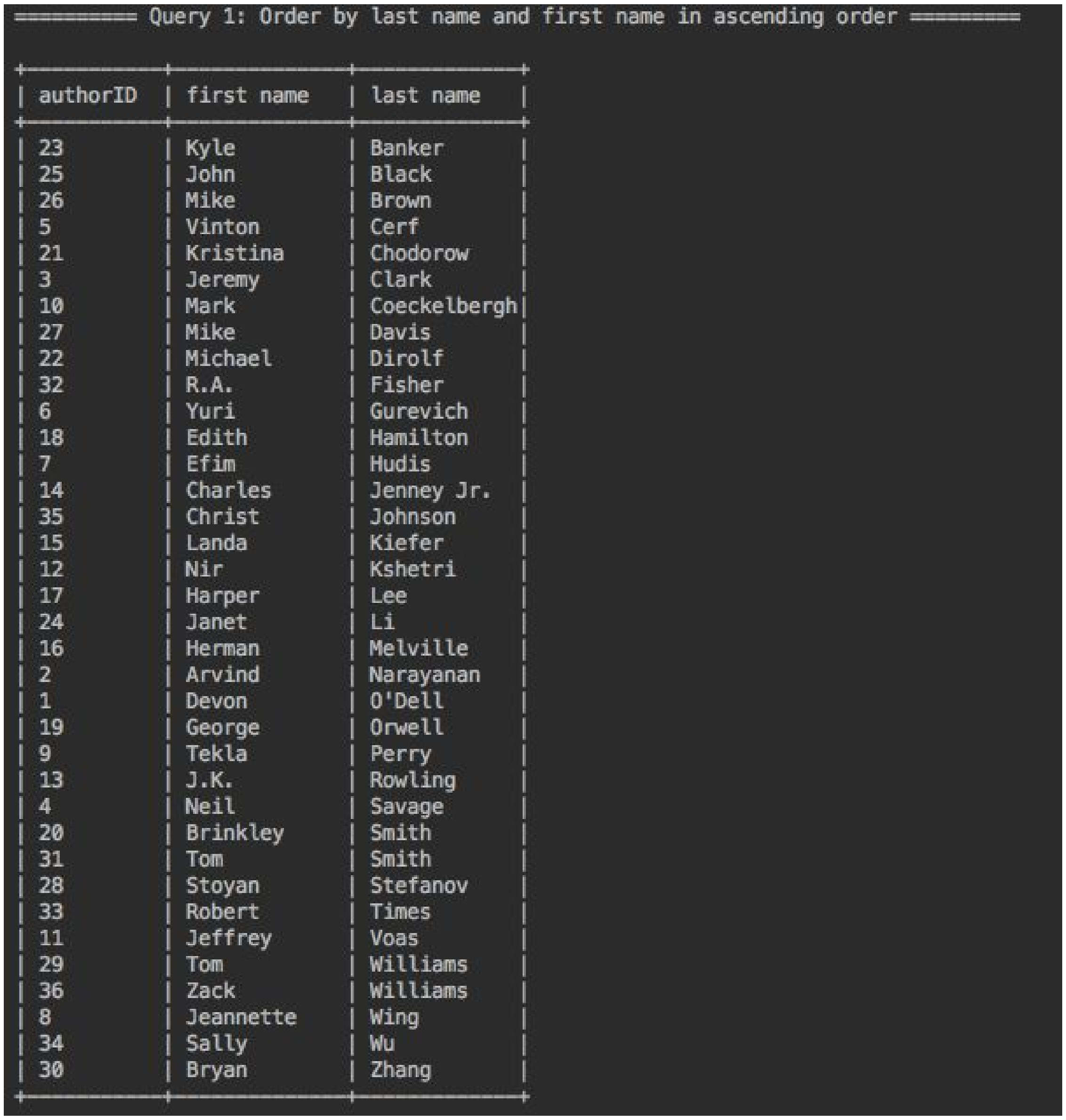
a. Select all authors from the authors table. Order the information * alphabetically by the author's last name and first name.

Syntax:



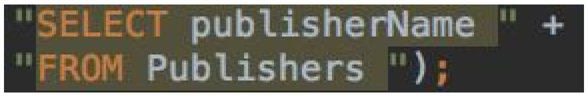
```
"SELECT * " +
"FROM Authors " +
"ORDER BY LastName, firstName ASC");
```

Result:



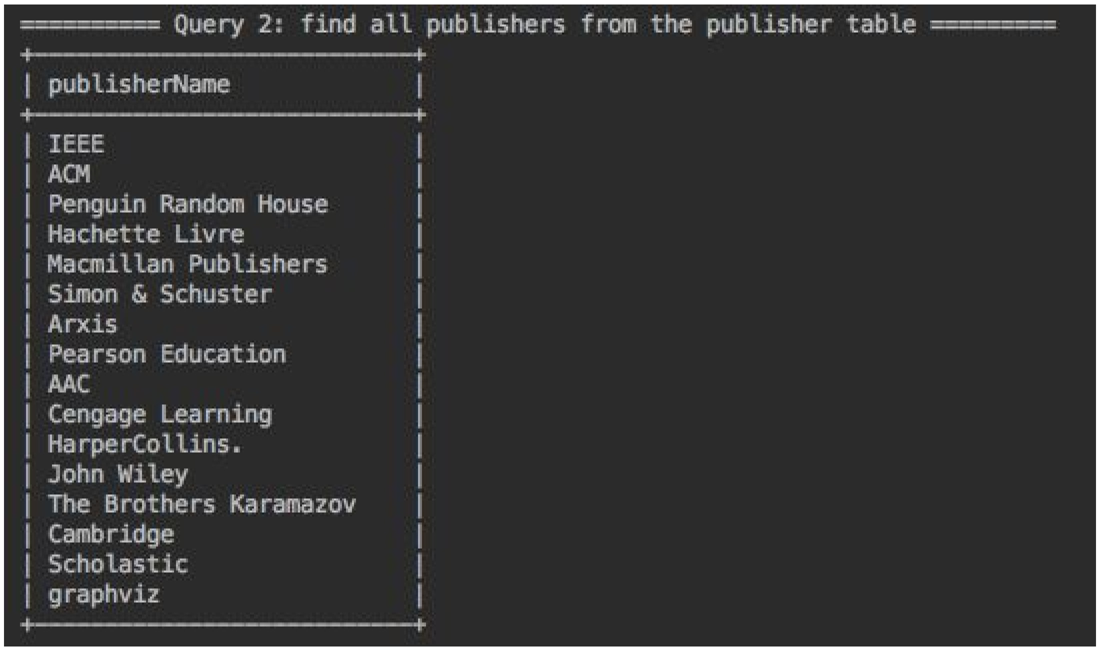
```
========== Query 1: Order by last name and first name in ascending order =========

+------------+---------------+--------------+
| authorID   | first name    | last name    |
+------------+---------------+--------------+
| 23         | Kyle          | Banker       |
| 25         | John          | Black        |
| 26         | Mike          | Brown        |
| 5          | Vinton        | Cerf         |
| 21         | Kristina      | Chodorow     |
| 3          | Jeremy        | Clark        |
| 10         | Mark          | Coeckelbergh|
| 27         | Mike          | Davis        |
| 22         | Michael       | Dirolf       |
| 32         | R.A.          | Fisher       |
| 6          | Yuri          | Gurevich     |
| 18         | Edith         | Hamilton     |
| 7          | Efim          | Hudis        |
| 14         | Charles       | Jenney Jr.   |
| 35         | Christ        | Johnson      |
| 15         | Landa         | Kiefer       |
| 12         | Nir           | Kshetri      |
| 17         | Harper        | Lee          |
| 24         | Janet         | Li           |
| 16         | Herman        | Melville     |
| 2          | Arvind        | Narayanan    |
| 1          | Devon         | O'Dell       |
| 19         | George        | Orwell       |
| 9          | Tekla         | Perry        |
| 13         | J.K.          | Rowling      |
| 4          | Neil          | Savage       |
| 20         | Brinkley      | Smith        |
| 31         | Tom           | Smith        |
| 28         | Stoyan        | Stefanov     |
| 33         | Robert        | Times        |
| 11         | Jeffrey       | Voas         |
| 29         | Tom           | Williams     |
| 36         | Zack          | Williams     |
| 8          | Jeannette     | Wing         |
| 34         | Sally         | Wu           |
| 30         | Bryan         | Zhang        |
+------------+---------------+--------------+
```

b. Select all publishers from the publishers table.
   Syntax:



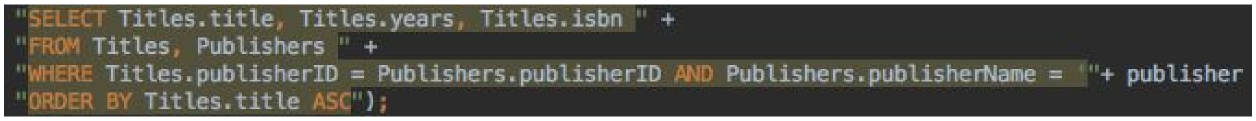
```
"SELECT publisherName " +
"FROM Publishers ");
```

   Result:



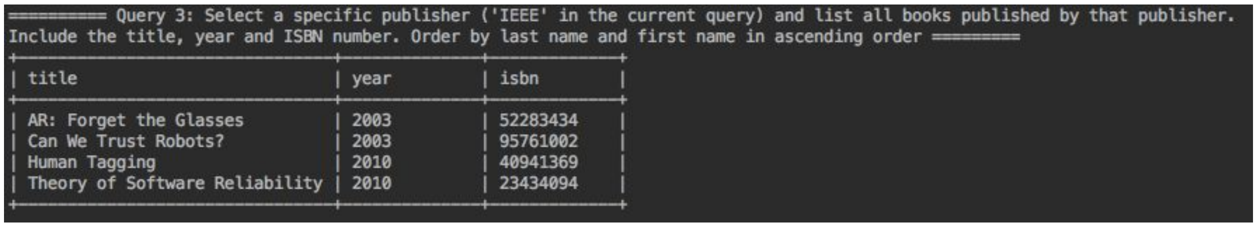
```
========== Query 2: find all publishers from the publisher table =========
+--------------------------------+
| publisherName                  |
+--------------------------------+
| IEEE                           |
| ACM                            |
| Penguin Random House           |
| Hachette Livre                 |
| Macmillan Publishers           |
| Simon & Schuster               |
| Arxis                          |
| Pearson Education              |
| AAC                            |
| Cengage Learning               |
| HarperCollins.                 |
| John Wiley                     |
| The Brothers Karamazov         |
| Cambridge                      |
| Scholastic                     |
| graphviz                       |
+--------------------------------+
```

c. 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
Syntax:



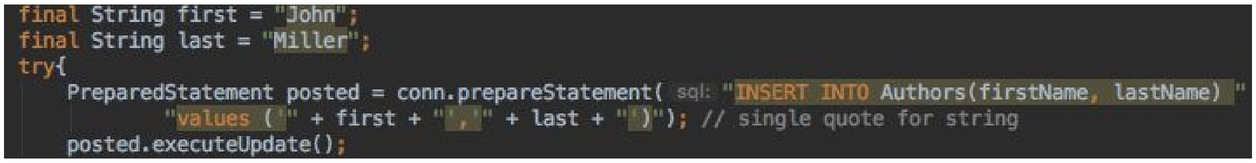
```
"SELECT Titles.title, Titles.years, Titles.isbn " +
"FROM Titles, Publishers " +
"WHERE Titles.publisherID = Publishers.publisherID AND Publishers.publisherName = '"+ publisher
"ORDER BY Titles.title ASC");
```

Result:



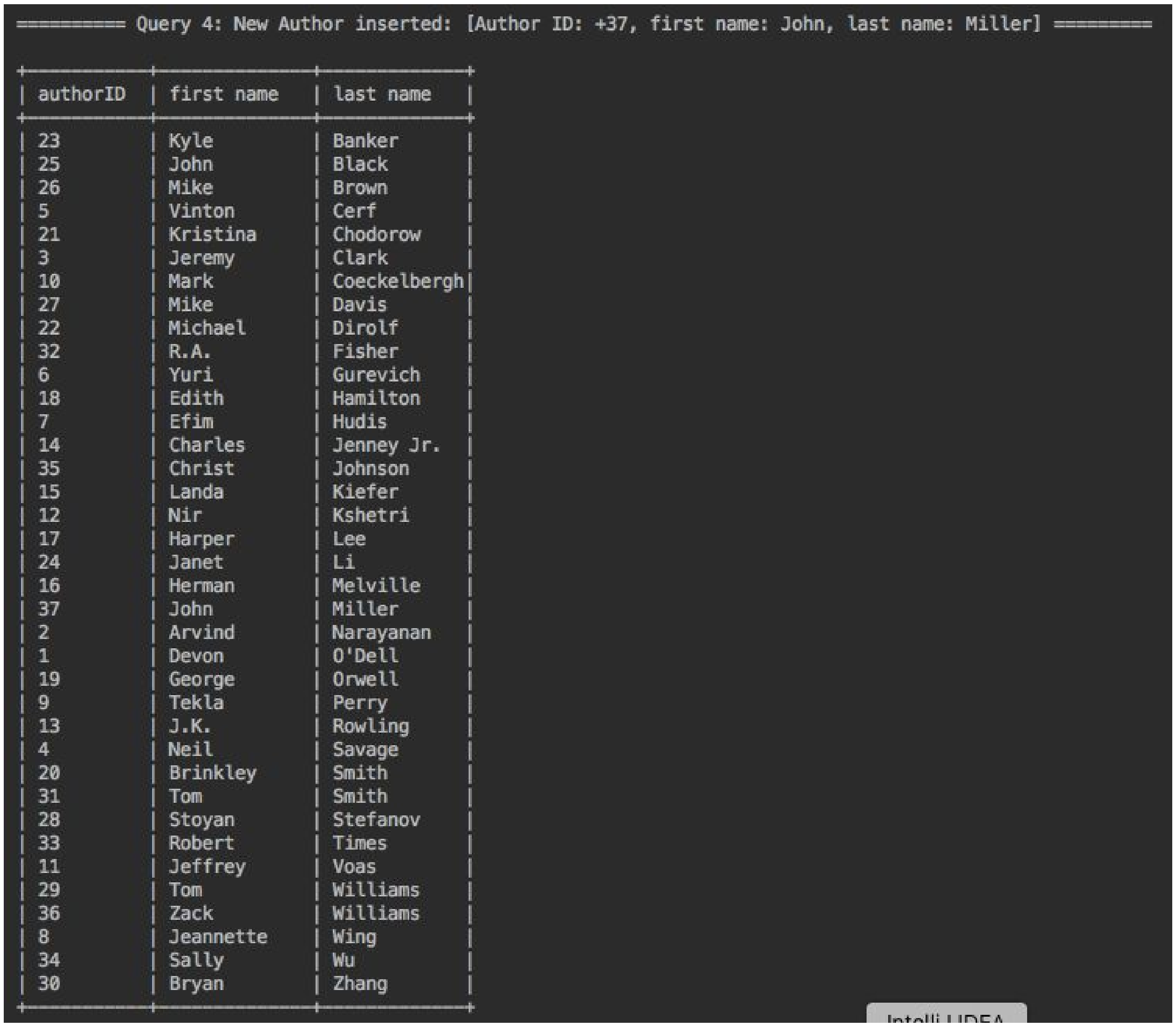
```
========== Query 3: Select a specific publisher ('IEEE' in the current query) and list all books published by that publisher.
Include the title, year and ISBN number. Order by last name and first name in ascending order =========
+---------------------------------+-------------+--------------+
| title                           | year        | isbn         |
+---------------------------------+-------------+--------------+
| AR: Forget the Glasses          | 2003        | 52283434     |
| Can We Trust Robots?            | 2003        | 95761002     |
| Human Tagging                   | 2010        | 40941369     |
| Theory of Software Reliability  | 2010        | 23434094     |
+---------------------------------+-------------+--------------+
```

d. Add new Author
Syntax:



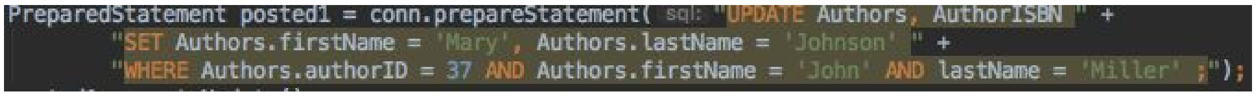
```
final String first = "John";
final String last = "Miller";
try{
    PreparedStatement posted = conn.prepareStatement( sql: "INSERT INTO Authors(firstName, lastName) "
            "values ('" + first + "','" + last + "')"); // single quote for string
    posted.executeUpdate();
```

Result (order by author's last name and first name):



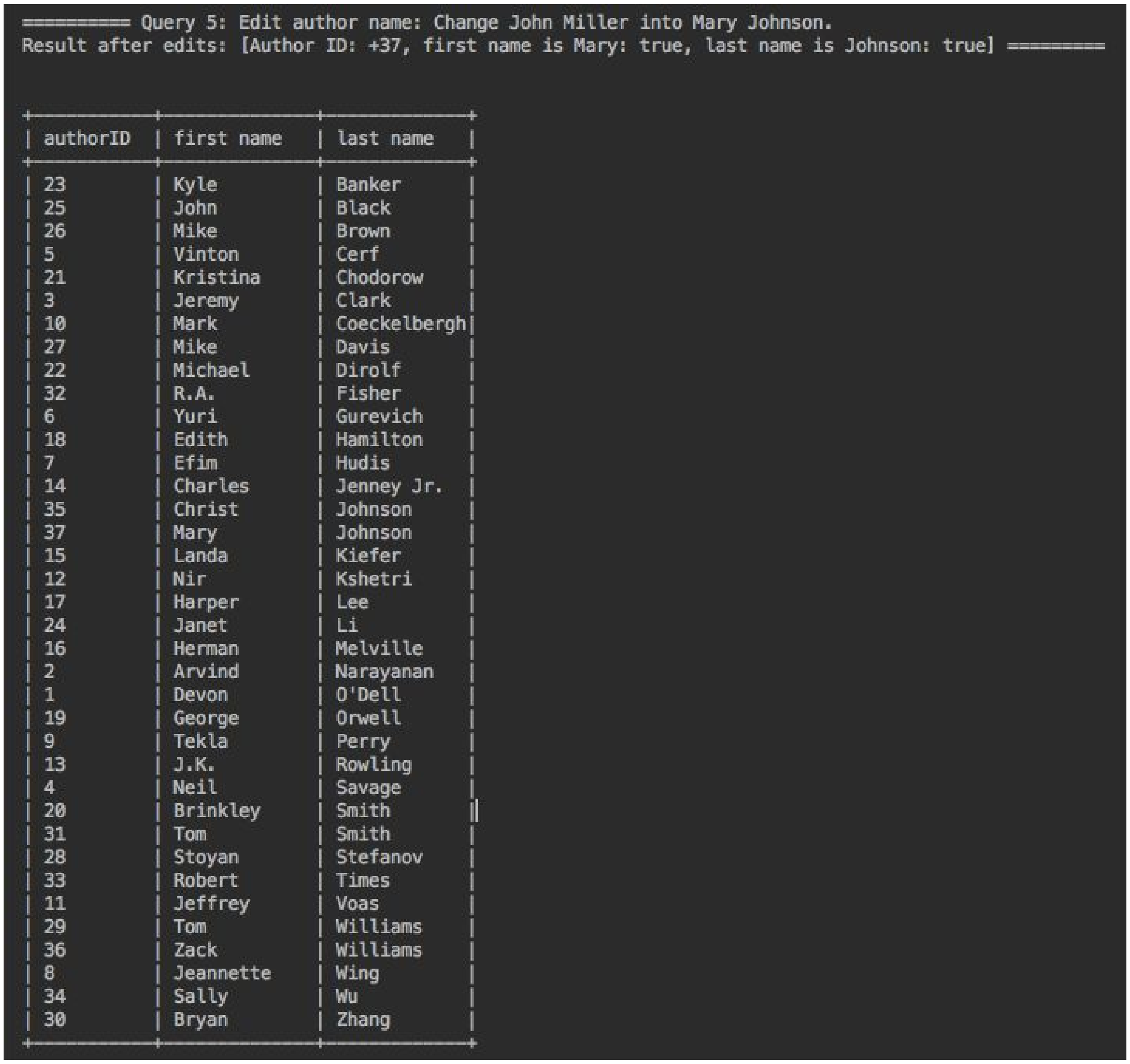
```
========== Query 4: New Author inserted: [Author ID: +37, first name: John, last name: Miller] =========

+------------+---------------+-------------+
| authorID   | first name    | last name   |
+------------+---------------+-------------+
| 23         | Kyle          | Banker      |
| 25         | John          | Black       |
| 26         | Mike          | Brown       |
| 5          | Vinton        | Cerf        |
| 21         | Kristina      | Chodorow    |
| 3          | Jeremy        | Clark       |
| 10         | Mark          | Coeckelbergh|
| 27         | Mike          | Davis       |
| 22         | Michael       | Dirolf      |
| 32         | R.A.          | Fisher      |
| 6          | Yuri          | Gurevich    |
| 18         | Edith         | Hamilton    |
| 7          | Efim          | Hudis       |
| 14         | Charles       | Jenney Jr.  |
| 35         | Christ        | Johnson     |
| 15         | Landa         | Kiefer      |
| 12         | Nir           | Kshetri     |
| 17         | Harper        | Lee         |
| 24         | Janet         | Li          |
| 16         | Herman        | Melville    |
| 37         | John          | Miller      |
| 2          | Arvind        | Narayanan   |
| 1          | Devon         | O'Dell      |
| 19         | George        | Orwell      |
| 9          | Tekla         | Perry       |
| 13         | J.K.          | Rowling     |
| 4          | Neil          | Savage      |
| 20         | Brinkley      | Smith       |
| 31         | Tom           | Smith       |
| 28         | Stoyan        | Stefanov    |
| 33         | Robert        | Times       |
| 11         | Jeffrey       | Voas        |
| 29         | Tom           | Williams    |
| 36         | Zack          | Williams    |
| 8          | Jeannette     | Wing        |
| 34         | Sally         | Wu          |
| 30         | Bryan         | Zhang       |
+------------+---------------+-------------+
```

e. Edit/Update the existing information about an author
   Syntax (change John Miller into Mary Johnson):



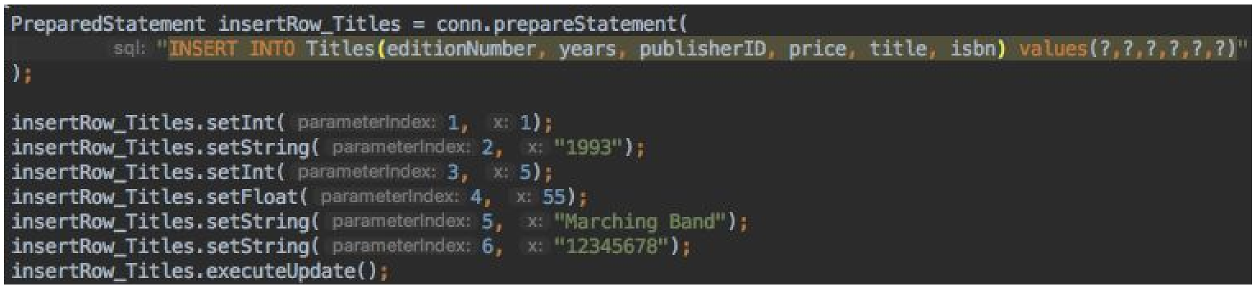
```
PreparedStatement posted1 = conn.prepareStatement( sql: "UPDATE Authors, AuthorISBN " +
        "SET Authors.firstName = 'Mary', Authors.lastName = 'Johnson' " +
        "WHERE Authors.authorID = 37 AND Authors.firstName = 'John' AND lastName = 'Miller' ;");
```

Result:



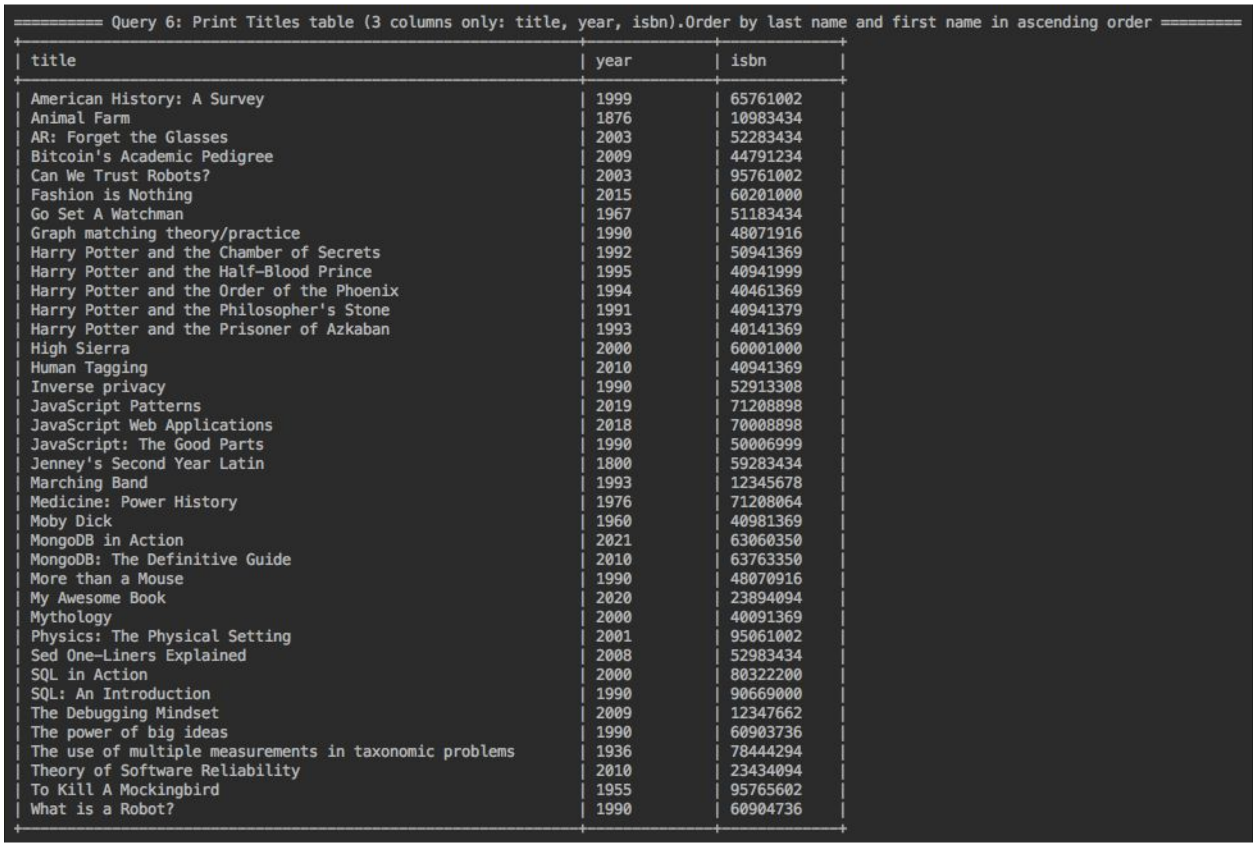
```
========== Query 5: Edit author name: Change John Miller into Mary Johnson.
Result after edits: [Author ID: +37, first name is Mary: true, last name is Johnson: true] =========


+-----------+--------------+-------------+
| authorID  | first name   | last name   |
+-----------+--------------+-------------+
| 23        | Kyle         | Banker      |
| 25        | John         | Black       |
| 26        | Mike         | Brown       |
| 5         | Vinton       | Cerf        |
| 21        | Kristina     | Chodorow    |
| 3         | Jeremy       | Clark       |
| 10        | Mark         | Coeckelbergh|
| 27        | Mike         | Davis       |
| 22        | Michael      | Dirolf      |
| 32        | R.A.         | Fisher      |
| 6         | Yuri         | Gurevich    |
| 18        | Edith        | Hamilton    |
| 7         | Efim         | Hudis       |
| 14        | Charles      | Jenney Jr.  |
| 35        | Christ       | Johnson     |
| 37        | Mary         | Johnson     |
| 15        | Landa        | Kiefer      |
| 12        | Nir          | Kshetri     |
| 17        | Harper       | Lee         |
| 24        | Janet        | Li          |
| 16        | Herman       | Melville    |
| 2         | Arvind       | Narayanan   |
| 1         | Devon        | O'Dell      |
| 19        | George       | Orwell      |
| 9         | Tekla        | Perry       |
| 13        | J.K.         | Rowling     |
| 4         | Neil         | Savage      |
| 20        | Brinkley     | Smith       |
| 31        | Tom          | Smith       |
| 28        | Stoyan       | Stefanov    |
| 33        | Robert       | Times       |
| 11        | Jeffrey      | Voas        |
| 29        | Tom          | Williams    |
| 36        | Zack         | Williams    |
| 8         | Jeannette    | Wing        |
| 34        | Sally        | Wu          |
| 30        | Bryan        | Zhang       |
+-----------+--------------+-------------+
```

f. Add a new title for an author
   Syntax (the whole Titles table has added data for every field (i.e., a new row) if we want to add a new title):



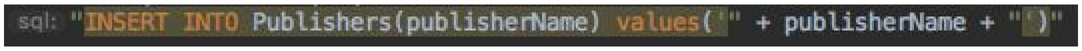
```
PreparedStatement insertRow_Titles = conn.prepareStatement(
        sql: "INSERT INTO Titles(editionNumber, years, publisherID, price, title, isbn) values(?,?,?,?,?,?)"
);

insertRow_Titles.setInt( parameterIndex: 1,  x: 1);
insertRow_Titles.setString( parameterIndex: 2,  x: "1993");
insertRow_Titles.setInt( parameterIndex: 3,  x: 5);
insertRow_Titles.setFloat( parameterIndex: 4,  x: 55);
insertRow_Titles.setString( parameterIndex: 5,  x: "Marching Band");
insertRow_Titles.setString( parameterIndex: 6,  x: "12345678");
insertRow_Titles.executeUpdate();
```

Result:



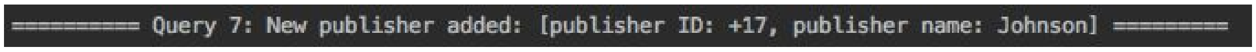
```
========== Query 6: Print Titles table (3 columns only: title, year, isbn).Order by last name and first name in ascending order =========
+--------------------------------------------------------------+--------------+-------------+
| title                                                        | year         | isbn        |
+--------------------------------------------------------------+--------------+-------------+
| American History: A Survey                                   | 1999         | 65761002    |
| Animal Farm                                                  | 1876         | 10983434    |
| AR: Forget the Glasses                                       | 2003         | 52283434    |
| Bitcoin's Academic Pedigree                                  | 2009         | 44791234    |
| Can We Trust Robots?                                         | 2003         | 95761002    |
| Fashion is Nothing                                           | 2015         | 60201000    |
| Go Set A Watchman                                            | 1967         | 51183434    |
| Graph matching theory/practice                               | 1990         | 48071916    |
| Harry Potter and the Chamber of Secrets                      | 1992         | 50941369    |
| Harry Potter and the Half-Blood Prince                       | 1995         | 40941999    |
| Harry Potter and the Order of the Phoenix                    | 1994         | 40461369    |
| Harry Potter and the Philosopher's Stone                     | 1991         | 40941379    |
| Harry Potter and the Prisoner of Azkaban                     | 1993         | 40141369    |
| High Sierra                                                  | 2000         | 60001000    |
| Human Tagging                                                | 2010         | 40941369    |
| Inverse privacy                                              | 1990         | 52913308    |
| JavaScript Patterns                                          | 2019         | 71208898    |
| JavaScript Web Applications                                  | 2018         | 70008898    |
| JavaScript: The Good Parts                                   | 1990         | 50006999    |
| Jenney's Second Year Latin                                   | 1800         | 59283434    |
| Marching Band                                                | 1993         | 12345678    |
| Medicine: Power History                                      | 1976         | 71208064    |
| Moby Dick                                                    | 1960         | 40981369    |
| MongoDB in Action                                            | 2021         | 63060350    |
| MongoDB: The Definitive Guide                                | 2010         | 63763350    |
| More than a Mouse                                            | 1990         | 48070916    |
| My Awesome Book                                              | 2020         | 23894094    |
| Mythology                                                    | 2000         | 40091369    |
| Physics: The Physical Setting                                | 2001         | 95061002    |
| Sed One-Liners Explained                                     | 2008         | 52983434    |
| SQL in Action                                                | 2000         | 80322200    |
| SQL: An Introduction                                         | 1990         | 90669000    |
| The Debugging Mindset                                        | 2009         | 12347662    |
| The power of big ideas                                       | 1990         | 60903736    |
| The use of multiple measurements in taxonomic problems       | 1936         | 78444294    |
| Theory of Software Reliability                               | 2010         | 23434094    |
| To Kill A Mockingbird                                        | 1955         | 95765602    |
| What is a Robot?                                             | 1990         | 60904736    |
+--------------------------------------------------------------+--------------+-------------+
```

g. Add new publisher
Syntax:



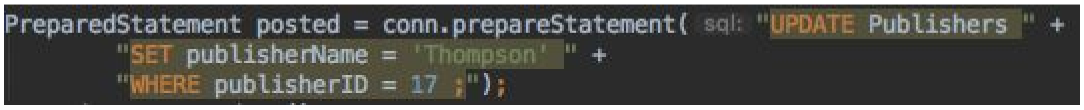
```
sql: "INSERT INTO Publishers(publisherName) values('" + publisherName + "')"
```

Result:



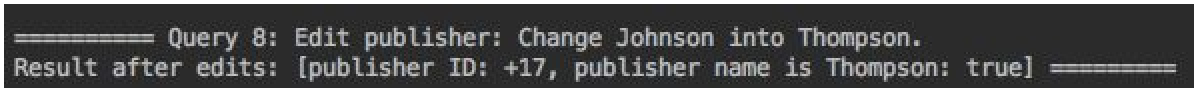
```
========== Query 7: New publisher added: [publisher ID: +17, publisher name: Johnson] =========
```

h. Edit/Update the existing information about a publisher
   Syntax:



```
PreparedStatement posted = conn.prepareStatement( sql: "UPDATE Publishers " +
        "SET publisherName = 'Thompson' " +
        "WHERE publisherID = 17 ;");
```

Result:



```
=========== Query 8: Edit publisher: Change Johnson into Thompson.
Result after edits: [publisher ID: +17, publisher name is Thompson: true] =========
```