COURSEWORK ASSIGNMENT **UNIVERSITY OF EAST ANGLIA**

School of Computing Sciences

UNIT: CMP-7025A Database Manipulation

ASSIGNMENT TITLE: SQL and Java Coursework Programming Exercise

DATE SET	**:**	see attached	
DATE & TIME OF SUBMISSION	**:**	see attached	
RETURN DATE	**:**	see attached	
ASSIGNMENT VALUE	**:**	see attached	
SET BY	**:**	B. de la Iglesia	**SIGNED:**
CHECKED BY	**:**	J. Chin	**SIGNED:**

Aim:

Assess knowledge of database design, programming and testing.

Learning outcomes:

Ability to complete a partial design.

Ability to write and test SQL queries.

Ability to produce an application program using SQL statements from within a Java Program using the Java JDBC interface.

Assessment criteria

Appropriate use of SQL to complete design.

Appropriate use of SQL statements to implement specified tasks.

Correct style of Java and JDBC programming.

SQL and Java program testing.

Readiness to demonstrate.

Submission of requested documentation.

Description of assignment:

See attached

Required:

See attached

Handing in procedure:

Please submit your piece of coursework electronically following instructions posted on Blackboard. Submissions will be time stamped by the system. Any late submission (after 3 pm on the day of submission) will be penalised.

Plagiarism:

Plagiarism is the copying or close paraphrasing of published or unpublished work, including the work of another student without the use of quotation marks and due acknowledgement. Plagiarism is regarded a serious offence by the University and all cases will be reported to the Board of Examiners. Work that contains even small fragments of plagiarised material will be penalised.

UNIVERSITY OF EAST ANGLIA

School of Computing Sciences

CMP-7025A Database Manipulation

SQL and Java Coursework Exercise

Set on : Thursday 10th October 2019
To be completed by : Wednesday 11th December 2019, 3 pm
Returned by : Thursday 9th January 2020
Value : 40%

Introduction

Libro plc is a bookshop that sells both over the Internet and through retail shops. Libro maintains a central database containing details of customers (both individuals and retail bookshops), books and current orders. The central database is held on a server computer that processes transactions coming from Internet users or from shops. Transactions are held in an input queue and processed by an application program on the server. The results of processing the transactions are placed in an output queue for despatch to Internet users or shops.

For this coursework exercise you are required to set up a database for Libro, write SQL statements for the transactions and develop a Java application program running the transactions through the JDBC interface. Naturally the exercise is greatly simplified compared to a real business. A detailed specification of the task to be undertaken and the deliverables to be produced for assessment is given below.

You will set up your 'Libro database' as a set of tables in your default schema. You are required to write the Libro server application program and *not* the client software for use on the Internet or in the shops. For this exercise you will simulate the queue of input transactions by reading a stream of transactions from a text file. Similarly, the results of the transactions will be sent to a text file that simulates the output queue.

You may, of course, use your own facilities to develop your program but the final version **must use PostgreSQL and run in the CMP labs.**

Application Program Functionality

The database comprises the following tables:

```
book (bno, title, author, category, price, sales)
customer (cno, name, address, balance)
bookOrder (cno, bno, orderTime, qty)
```

Notes:

The `book` table holds data on each book that Libro offers for sale.
The `customer` table holds details of all Libro customers.
The `bookOrder` table holds details of current orders placed by Libro customers. The order details are archived periodically and the bookOrder table is then emptied. Each order is for a single customer and for a single title but may be for multiple copies of the title.

`bno` is a Libro's six-digit book number used by them to identify a book offered for sale.
`title` is the title of the book.
`author` is the author(s) of the book.
`category` is one of Science, Lifestyle, Arts or Leisure.
`price` is the selling price of the book.
`sales` is a count of the number of copies of the book which have been sold. This is zero when a new book is inserted into the database.
`cno` is a six-digit number used to identify a customer.
`name` is the name of the customer.
`address` is the address of the customer.
`balance` is the amount of money owed to Libro by the customer for books ordered. This is always zero when a new customer is inserted into the database.
`orderTime` is a timestamp recording the instant in time when an order was put on the Libro database.
`qty` is the number of copies of a specific book ordered by a customer.

The transactions of interest to us in this exercise are:

A. Insert a new book.

B. Delete a book.

C. Insert a customer.

D. Delete a customer.

E. Place an order for a customer for a specified number of copies of a book. The copies ordered are assumed to be sold and will have to be paid for. Books are not supplied on 'sale or return' terms of business.

F. Record a payment by a customer. The payment is subtracted from the customer's balance.

G. Find details of customers who have current orders for a book with a given text fragment in the book title. For example, find customers with orders for books with 'Java' in the title. This transaction produces a report with lines showing the full title of a book ordered, the customer name and the customer address relevant to the order. The report is to be sorted by title and then by customer name.

H. Find details of books ordered by a specified customer. The report will show the name of the customer followed by, for each book, the book number, title and author, sorted by book number.

I. Produce a book report by category. This report shows, for each category, the number of books sold and the total value of these sales. The total value calculation assumes that the currently held price is used and any earlier changes in the price of a book since it was inserted into the database are ignored.

J. Produce a customer report. This report shows, for each customer, the customer number, customer name and a count of the number of copies of books on order (if any). This report is to be in customer number order.

K. Daily sales. This transaction returns the total number of copies of books sold today.

X. A transaction sent to close down the server application program.

Note that for each transaction the program should send a reply to the output queue (a text file). Transactions A to F and X should, if successful, send a simple message showing the transaction type (A, B, …) and confirming that it has been completed. For example : `'X. Libro application closing down'.`

Reports are sent to the output queue and comprise the application type followed by the relevant data in a simple tabular format. You are not expected to implement report headers, report footers, page breaks, control breaks, fancy fonts, colour, company logos and so on, as might be found in a 'real' report. We will assume that the appropriate client software does this type of presentation dressing.

A transaction that contains errors or would cause errors should send to the output queue its transaction type and a user friendly error message.

Assignment

1. Database definition and loading (approx. 20% of marks)

Prepare an SQL program to create your copy of the Libro database. This should take, as a starting point, the table definitions given in Appendix 1 but you also need to prepare additional SQL clauses and/or statements to complete the definition of the database by specifying primary keys, domain constraints, entity and referential integrity constraints, etc. Note that you should NOT modify the name and type of the attributes (i.e. the information you have been given). Save all your Data Definition Language (DDL) statements in a text file.

Load a reasonable volume of test data into the tables for use in your testing. The test data should be sufficient to test all of the queries with their expected output and should provide a suitable environment in which to test normal operation as well as abnormal conditions.

- Document this stage with a copy of your complete DDL statements in SQL (including any table definitions, views, triggers, comments, etc.) You should bring a printed copy to the demonstration.

- Produce also a copy of the test data you loaded for testing and bring along to the demonstration or be ready to show it in screen as a text file.

- Separate marks will be awarded for the completeness/quality of your online submission.

2. Interactive SQL version of the transactions (approx. 40% of marks)

Prepare and test interactive SQL statements for the various types of tasks defined above. Test these statements using the SQL editor in PostgreSQL. The purpose is to test your SQL statements before using them in your Java program. You may need more than one execution of some of the task types to demonstrate the correctness of your work (e.g. test primary keys, referential integrity, correct and incorrect execution, etc). Your .sql files need to be ready to be loaded (copied/pasted) during the demo so if you fail to demonstrate your Java program working you can at least demonstrate your prototype SQL statements through the interactive interface.

- Document this stage with a copy of the SQL statements. Your SQL statements **need** to be accessible as text files through the demonstration for copy/paste.

- **Evidence of testing** of each SQL statement (e.g. copy of the output of running the query) at least once but you should perform further testing. Any SQL statement which has not been submitted or is submitted **with evidence of testing** will receive 0 marks.

- Separate marks will be awarded for the completeness/quality of your online submission.

3. Java application program (approx. 20% of marks)

Deploy suitable versions of your SQL produced in stage 2 in a Java application program using the JDBC interface. The program will read a queue of transaction details from a text file and send the results and/or error messages to the output queue (another text file). The

precise input format of transactions is described below. The client software will have ensured that the values of individual fields in transactions are of the correct type and format so this does not have to be checked in your program. However, the program has to make checks against the database to ensure that transaction details are valid. For example, a customer number field will be a valid integer number but may not correspond to an actual customer in the database. Where possible, the program should handle errors by reporting them and then continue with the next transaction.

The deliverables for this part of the assignment are:

- A commented program source listing (your .java file/s) which demonstrate the use of SQL in Java
- Printed copies of input and output files used to show that the program works as specified.
- Separate marks will be awarded for the completeness/quality of your online submission.

The objective is to demonstrate the good use of SQL and database features and not complex Java coding. Source programs should be consistently laid out and printed in a form that is easy to read

4. Week 12 (beginning on the 9th December) Demonstrations (5% of marks)

Please keep your tables, test data, interactive SQL statements and program after completing the assignment so that you can give a demonstration to show your Java application working. Demonstrations will be scheduled for week 12 and marks will be awarded for the demonstration. Demonstrating your work is therefore a mandatory part of the assessment procedure and those that do not come for their scheduled demonstration time will receive a mark of zero for the assignment unless they have obtained an extension to the deadline. The demonstration will involve loading a set of provided test data and carrying out a standard set of tests. The SQL and program demonstrated must match the version submitted!

Summary of Deliverables (Approx. 15% of marks)

- Electronic submission to Blackboard (All files to be collected in a folder named with your student number, zipped and submitted following instructions in Blackboard):
    - Part 1: A copy of your SQL data definition statements, annotated as necessary, together with the test data.
    - Part 2: Your SQL data manipulation statements (annotated with comments if necessary) for each of the requirements plus evidence of testing.
    - Part 3: Your Java source code files (.java) and input and output files.
- Attend and participate in a demonstration in week 12

Important notes

- Your demonstration should be well prepared as per instructions issued nearer the time and should run smoothly.
- The document you submit should be complete and neatly formatted to ease reading.
- This is an individual piece of coursework NOT a group project. Collusion/plagiarism checks may be carried out.
- As you will be given a SQL script in week 12 to load test data into your tables, it is vital that you do not change the table names, field names, field types etc. from what is described below.
- The demonstration must take place on a CMP lab machine using a Java desktop application as the client communicating with the CMP PostgreSQL database server. We will not accept other systems, languages, technologies or machines.

Appendix

Transaction formats

Each transaction comprises one or more lines, each terminated by an end-of-line code. The first line always comprises a single character giving the transaction type. Where the transaction type requires more details to be input, these are on separate lines following the transaction type. The text file simulating the input queue will comprise a series of such transactions terminating with a type X transaction.

Transaction Type	Fields	Example
Insert book (A)	book number, title, author, category, price[1]	A 123456 More Java Smith, A Science 25.99
Delete book (B)	book number	B 123456
Insert customer (C)	customer number, name, address[2]	C 678901 Jones, B 1 High St., Diss
Delete customer (D)	customer number	D 678901
Place order (E)	customer number, book number, quantity[3]	E 678901 123456 10
Record payment (F)	customer number, amount	F 678901 35.99
Find customer (G)	title text fragment	G Java
Find books (H)	customer number	H 678901
Book report (I)	(none)	I
Customer report (J)	(none)	J
Daily sales (K)	(none)	K
End of requests (X)	(none)	X

[1] When a new book is inserted the `sales` column is always zero so no value for sales is given here.

[2] When a new customer is inserted the `balance` is always zero, so no value for balance is given here.

[3] When a new order is inserted the `orderTime` should be taken from the system so no value e is given here.

Minimal database definition[4]

The following SQL program provides a minimal definition of the Libro database for use in part 1. A copy of this text can be found in the file `CgTableDef.txt` in Blackboard. To do part 1 of the assignment you will probably want to *add* SQL clauses or statements. However to enable my testing of your program during the demonstration you should use the same tables, columns and column data types in your version.

```
CREATE TABLE book (
    bno         INTEGER,
    title       VARCHAR(20),
    author      VARCHAR(20),
    category    CHAR(10),
    price       DECIMAL(6,2),
    sales       INTEGER);

CREATE TABLE customer (
    cno         INTEGER,
    name        VARCHAR(20),
    address     VARCHAR(30),
    balance     DECIMAL(8,2));

CREATE TABLE bookOrder (
    cno         INTEGER,
    bno         INTEGER,
    orderTime   TIMESTAMP,
    qty         INTEGER);
```

Help!

If you do not know where to start in the Java programming part, then there is a 'starter program' which might help. This is on Blackboard as `starter.java`. This program reads a text file, calls a method and writes to a text file. It does not do any database processing. The method called is just a 'stub' but the program gives a possible working framework to start your own development.

[4] Of course, the number of columns and their lengths is not realistic. The objective is to have enough data to show the principles of database programming and not to show a realistic design for a bookshop.