FinTech 512 Midterm Spring 2023: Bookshop

Name: Haoclan Yang

NetID: hy2049

There are four questions, with the point values as shown below. You have 1 hour and 15 minutes. There are 100 points on the exam. Pace yourself accordingly. This exam must be individual work. You may not collaborate with your fellow students. You may use the two sheets of notes you prepared. You may not use any other external resources, e.g., electronic devices. Points will be deducted if you provide more examples than the number listed.

I certify that the work shown on this exam is my own work, and that I have neither given nor received improper assistance of any form in the completion of this work.

Signature: Haoclan Yang

You've been contacted by a local entrepreneur looking for technical help with adapting their IT systems for their music shop, which they are transforming into a bookstore. Here is a recent email the entrepreneur has sent you:

> Hello <your name>,
>
> I really appreciate that you have signed on to help with this project and I trust that you find the terms in the contract acceptable.
>
> As we do not have the budget for full-time technical staff and we focus on customer experience, we would like a single system that can manage books from the time we purchase them through their stocking in the store to their invoicing, delivery, and settlement of the customer's credit card transactions. Since our staff is comfortable with the Flask application a previous consultant built to help us sell new and used music albums, we would like to use the same application while having it modified to work with books. The application interfaces with our Point-Of-Sale system which handles cash, check, and credit-card sales. As we do with the album sales, we would like to review weekly, monthly, and annual sales trends to see what the most popular books and book genres are, and who the most active book buyers and sellers are. We have a loyalty program offering discounts to frequent music buyers, and we are working on the rules for a similar program for book sellers and buyers.
>
> We were impressed with your work on the Chinook database queries and we've heard that you have skills in database design and website development. We are looking forward to your technical advice on how to add books to our existing chinook database.

NetID: lyou

Question 1: Requirements

a. (12 pts) Identify three functional requirements (FR) and one non-functional requirement (NFR) in the above email. State the requirement as a sentence and indicate its type (FR, NFR.)

1. FR: a system that can handle management of books during purchasing, stocking, invoicing, delivery and settlement.

2. FR: the system has to be on a Flask application.

3. FR: the system should be able to interface with their Point-of-Sales system.

~~(FR: should be able to give weekly, monthly and annual sales trends)~~

1. NFR: the system should have good performance as they "focus on customer experience." and should be efficient & availble as they "do not have budget for full-time technical staff".)

b. (8 pts) Given the quality attributes listed below, identify the three highest priority quality attributes for the entrepreneur's use case. Explain in a sentence or two why you ranked them in the way that you did.

   a. Quality attributes: performance, efficiency, reliability, security, availability, functionality, modifiability, maintainability, usability, portability, testability, reusability.

① efficient. ② reliability ③ security.

1. it should be efficient, as they say they "focus on customer experience" but "do not have budget for full-time technical staff". so it makes sense for them to make the most out of this software. Software should not waste their source resources.

2. it should be reliable, as they are switch from music shop to bookstore and they wish to use a similar system as before, and it should interface with their own original system. It should work correctly under such complicated situation.

3. it should be secure, as they wish the system can process sales data, and more importantly, access buyer's information to offer discounts. so the software should be able to protect their customers' information.

NetID: ly2040

Question 2: Database Design

a way to integrate invoice_items to make it accomodate both trackid(music) and titleid(book) is have (item_id, item_type) in invoice_items, and item_type have type 'music' and 'book'.

We have been keeping our records of book sales in a spreadsheet. Here are the first five rows. Can you help us work out what tables we need to add to chinook to support book sales?

Customer	Title	Pages	Price	Authors
Fernanda Ramos	Harry Potter and the Sorcerer's Stone	309	19.95	J.K. Rowling
Mark Philips	The Hobbit	320	9.95	J.R.R. Tolkien
Helena Holy	Exploring Requirements: Quality Before Design	300	39.95	Donald C. Gause, Gerald M. Weinberg .
Jimmie Lenz	The Intelligent Investor	640	17.98	Benjamin Graham
Francois Tremblay	Why Zebras Don't Get Ulcers	560	22.99	Robert Sapolsky

[Update the chinook schema (last page) to track the 'spreadsheet' data and link it to the invoice. If you need to modify an existing table, describe the changes to be made to those table(s)]

a. (20 pts) Normalize the data in the given data rows, to third normal form. Show your table (relation) names, field names, and the reformatted data.

customerid	fn	ln
1	Fernanda	Ramos
2	Mark	Philips
3	Helena	Holy
4	Jimmie	Lenz
5	Francois	Tremblay

titleid	Title
1	Harry Potter and the Sorcerer's Stone
2	The Hobbit
3	Exploring Requirements: Quality Before Design
4	The Intelligent Investor
5	Why Zebras Don't Get Ulcers

customerid	titleid
1	1
2	2
3	3
4	4
5	5

titleid	Pages
1	309
2	320
3	300
4	640
5	560

titleid	Price
1	19.95
2	9.95
3	39.95
4	17.98
5	22.99

Authorid	Authors
1	J.K Rowling
2	J.R.R. Tolkien
3	Donald C. Gause
4	Gerald M.
5	Weinberg
6	Benjamin Graham
7	Robert Sapolsky

Authorid	Titleid
1	1
2	2
3	3
4	3
5	3
6	4
7	5

b. (10 pts) Write the SQL DDL 'CREATE TABLE' statement for one of the relations you defined in part a.

```
create table customer ( customerid int not null primary key, fn varchar(30), ln varchar(30) );
insert into customer ( customerid, fn, ln ) values ( 1, "Fernanda", "Ramos");
insert into customer ( customerid, fn, ln ) values ( 2, "Mark", "Philips");
insert into customer ( customerid, fn, ln ) values ( 3, "Helena", "Holy");
insert into customer ( customerid, fn, ln ) values ( 4, "Jimmie", "Lenz");
insert into customer ( customerid, fn, ln ) values ( 5, "Francois", "Tremblay");
```

NetID: tyzap

Question 3: Testing

The previous contractor developed the code for calculating loyalty program discounts, Python function 'loyalty_discount', **shown on the second to the last page of the exam**. Given that a significant amount of both money and goodwill are at stake in the proper execution of this code, we would like an assessment of how thoroughly the code has been tested.

Here are the test cases the contractor provided:
Invoice has 3 items, invoice total is 110, customer has 2 orders, no onsale items
Invoice has 6 items, invoice total is 150, customer has no orders, no onsale items

a. (10 pts) For the code and the tests given above, what level of coverage (path, condition, statement, none of the above) is achieved?

none of above is achieved.

b. (10 pts) What test cases need to be added to achieve full (statement, condition, and path) coverage? (You can give just the parameter values, as in the test case examples above)

items = n, invoice total = t, order_count = l, on_sale_item = S.

l = 2, n = 6, t = 150, S = 2.

l = 2, n = 6, t = 200, S = 11.

l = 2, n = 3, t = 150, S = 2

l = 30, n = 6, t = 150, S = 2

l = 30, n = 6, t = 200, S = 11

l = 30, n = 3, t = 150, S = 2

l = 0, n = 6, t = 150, S = 2

l = 0, n = 6, t = 200, S = 11

l = 0, n = 3, t = 150, S = 2

c. (5 pts) What difficulties exist in testing the code as it is currently written?

The condition (branches) and codes are not really defined, too many conditions clustered in the if-else statement, makes it difficult to identify equivalent class.

NetID: hy2010

Question 4: Refactoring

a. (10 pts) We've heard that 'Code Smells' are bad; does the loyalty program code have any? (Name three code smells you identify)

① mysterious name. (what is n, t, s & l ? they are not related to invoice item closely...)

② long function (the function is long, including getting quantity / onsale amount ... ; calculating discount rate and etc. too many responsibilities here)

③ divergent change (the function to decide the discount rate is very rigid, say if the shop decide to add some ranks of discount for customers / merge some ranks, we need to rewrite the entire code)

b. (15 pts) Can you make suggestions about how to improve the code, given that we intend to apply the same logic to books as well as music? (List three refactorings you could apply to address the code smells identified in part a. Be specific, e.g., if you apply 'Extract Function', give the function name and its parameters.) Write code, noting the name of the refactoring in a comment.

① mysterious name ∈ rename variable : n => quantity ; t => totalPrice; s => amountOnSale
d => discountRate

② split function / extract function

```
def get_invoice_info( invoice_id, customer_id, onsale )
    ( line 2-8 the same as it was ... )
    return invoice_items, order_count.
```

③ make '5', '100', '10', '200' in line 20 & line 21 to be functions/variables that can get its value from somewhere else, such as 'upper-limit'

NetID: hyzap

```
1  def loyalty_discount(invoice_id, customer_id, onsale):
2      invoice_items = get_db().execute("SELECT itemid, unitPrice, quantity "
3                                       "  FROM INVOICE_ITEMS "
4                                       "  WHERE invoiceId = ?",
5                                      (invoice_id,)).fetchall()
6      order_count = get_db().execute("SELECT count(*) FROM invoices "
7                                     "  WHERE customerId = ?",
8                                    (customer_id,)).fetchone()
9
10     n = t = s = 0
11     for item in invoice_items:
12         n += item["quantity"]
13         t += item["quantity"] * item["unitPrice"]
14         for id in onsale:
15             if itemid == id:
16                 s += 1
17
18     l = max(order_count/10.0,3)
19       d = 0.0
20     if n > 5 and t > 100.00:
21         if s < 10 or t < 100.00:
22             d = .25
23         else:
24             d = .30
25         d = min(d,d * (l*.5))
26     else:
27         d = 0.033 * l
28
29     return d
```

n=3 t=110 l=3 s=0

n=6 t=150 l=3 s=0

2.

(n>5 & t>100)

(s<10 | t<200)

(s≥10 & t≥200)

2

(n≤5 | t≤100)

l

3

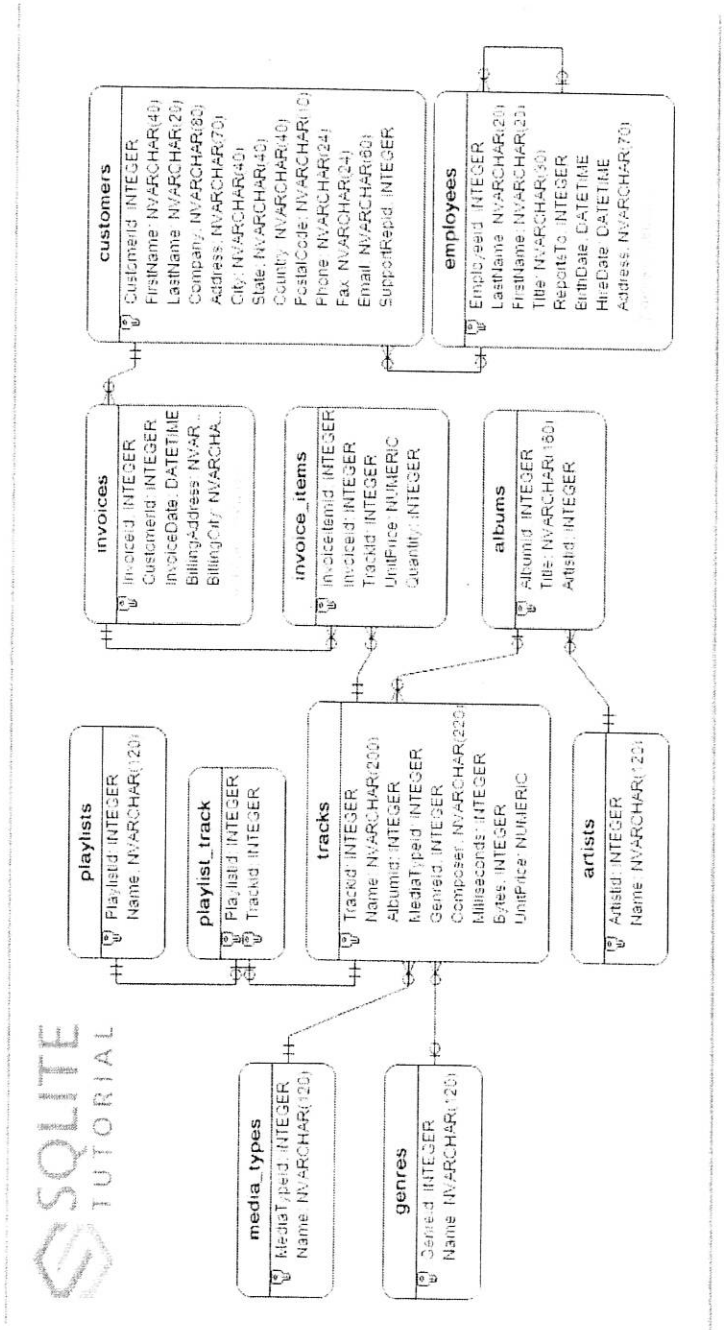
order_cnt/10

Statement coverage:
1. test_statement_contractor1 - < 5 items, no on sale items, 2 customer orders, discount = 0.099
2. test_statement_contractor2 - > 5 items, no on sale items, no orders, total > 100, discount = 0.25
3. test_statement_onsale1 - > 10 invoice items, > 10 on sale, invoice total > 100, previous customer orders, discount = 0.25
4. test_statement_onsale2 - > 10 invoice items, 10 on sale, invoice total > 200, previous customer orders, discount = 0.3

Branch/condition coverage:
5. test_branch_1_false_invoice_empty - No invoice, no onsale items, discount = 0.099
6. onsale items that aren't on the invoice, discount = 0.25

Path coverage:
7. onsale items not on invoice, total < 100, discount = 0.099
8. onsale items on invoice, total < 100, discount = 0.099

NetID: hy208


SQLITE
TUTORIAL
playlists
PlaylistId INTEGER
Name NVARCHAR(120)
playlist_track
PlaylistId INTEGER
TrackId INTEGER
tracks
TrackId INTEGER
Name NVARCHAR(200)
AlbumId INTEGER
MediaTypeId INTEGER
GenreId INTEGER
Composer NVARCHAR(220)
Milliseconds INTEGER
Bytes INTEGER
UnitPrice NUMERIC
media_types
MediaTypeId INTEGER
Name NVARCHAR(120)
genres
GenreId INTEGER
Name NVARCHAR(120)
artists
ArtistId INTEGER
Name NVARCHAR(120)
albums
AlbumId INTEGER
Title NVARCHAR(160)
ArtistId INTEGER
invoice_items
InvoiceLineId INTEGER
InvoiceId INTEGER
TrackId INTEGER
UnitPrice NUMERIC
Quantity INTEGER
invoices
InvoiceId INTEGER
CustomerId INTEGER
InvoiceDate DATETIME
BillingAddress NVAR...
BillingCity NVARCHA...
customers
CustomerId INTEGER
FirstName NVARCHAR(40)
LastName NVARCHAR(20)
Company NVARCHAR(80)
Address NVARCHAR(70)
City NVARCHAR(40)
State NVARCHAR(40)
Country NVARCHAR(40)
PostalCode NVARCHAR(10)
Phone NVARCHAR(24)
Fax NVARCHAR(24)
Email NVARCHAR(60)
SupportRepId INTEGER
employees
EmployeeId INTEGER
LastName NVARCHAR(20)
FirstName NVARCHAR(20)
Title NVARCHAR(30)
ReportsTo INTEGER
BirthDate DATETIME
HireDate DATETIME
Address NVARCHAR(70)