

Introduction to Databases-Final Project

Group 2, August 3, 2018

Mirnes Salkic and Prasad Vaidya

Objective:

Design a web-based solution for a bookstore with the objective to **retrieve, filter, and update** customer purchase information. The bookstore offers a 'credit system' to its customers- every time a person buys a book he/she earns a specified amount of credits, which can be used in one of the future purchases of books.

Target market:

This project is inspired by a bookstore in Peterborough, ON which currently uses a paperbased catalogue for their 'credit system'. This software will enable them to retrieve, filter, and update their customer data more efficiently.

Includes Files: The files included, and their functionality are as follows

- 1) db_create.php: To create the database
- 2) tables_create.php: To create the tables in the database.
- 3) Index.html: Front end of the database
- 4) Insert.php: PHP file that connects the front-end form to the database.
- 5) Customer_Info.php: PHP file for summary table of customer info.
- 6) Profit_Summary.php: PHP file for Profit summary.
- 7) Stock_Summary.php: PHP file for stock summary.
- 8) book.jpeg: Used in Background picture of index.html

Assumptions:

- Books are bought from publishers.
- Every order must be done by a registered customer in a database. If a customer does not want to register, we assume that there is a dummy customer id that is being used to store data for all those who do not wish to participate in the "credits program". That is a workaround to the constraint we set.

On the vocabulary used:

- We use purchase to refer to the bookstore owners buying books from publishers to update their stock.
- Total_purchase_price – this is the total price including taxes (13%) paid in one particular purchase of isbn's. For example, we purchase 30 books for Databases for 20 dollars, and 40 books for Parallel Computing for 50 dollars from the same or different publishers.

Total purchase price would be the total price paid for this entire transaction at a particular time.

- Purchase_price_of_a_book – is the price paid for a particular ISBN at a given time. This in combination with purchase_quantity help us calculate the avg_cost for isbn which in turn will help us form our selling price.
- Avg_cost – this is the weighted average cost for a particular ISBN. We assume that a particular ISBN is always bought from the same publisher, but it can be bought at different prices at different times. Thus, the avg_cost will be the weighted cost average for a particular ISBN. In the backend, we add 25% to this cost of a ISBN to obtain the selling price of a book.
- Credits – dollar amounts of credits given to customers. When an account is created the customer gets \$0.00 credits. After each purchase 5% of the total price [before tax] is given to the customer in form of credits. At each future transaction the customer may choose to use credits available or not. If the customer chooses to use the credits in one of his/her future transactions he can only use up to 10% of the total order price.
- Discount – is the actual amount of credits applied for a transaction.
- Final price – is the final price of the order after the discount and tax have been considered.

Normalization:

We started with having in mind three larger tables:

PURCHASE (purchase_id, date_time, total_purchase_price (PURCHASE DETAIL purchase_detail_id, isbn, purchase_quantity, purchase_price_per_book))

STOCK (isbn, num_copies, title, year, avg_cost, price, pub_id, pub_name, pub_address, pub_phone)

CUSTOMER (customer_id, first_name, middle_name, last_name, phone, email, credits (CUSTOMER_ORDER order_id, customer_id, datetime, total_price, discount, final_price (ORDER_DETAIL order_detail_id, isbn, quantity_ordered)))

1st normal form

- Each unique purchase can have multiple purchase details (e.g. multiple purchase quantities (e.g. we buy 20 books of one kind, and 30 books of another). Thus we split the purchase into two tables: purchase and purchase detail.
- Each customer can have multiple orders in the bookstore. Thus customer order type of information goes into another table.
- Similarly, each customer order may have multiple order details (e.g. a quantity ordered of isbn A, and b quantity ordered of isbn B).

PURCHASE (purchase_id, date_time, total_purchase_price)

PURCHASE DETAIL (purchase_detail_id, purchase_id, isbn, purchase_quantity, purchase_price_per_book)

STOCK (isbn, num_copies, title, year, avg_cost, pub_name, pub_address, pub_phone)

CUSTOMER (customer_id, first_name, middle_name, last_name, phone, email, credits)

CUSTOMER_ORDER (order_id, customer_id, date_time, total_price, discount, final_price)

ORDER_DETAIL (order_detail_id, isbn, order_id, quantity_ordered)

2nd normal form

all non-key attributes must be fully functionally dependent on the entire primary key

PURCHASE table:

purchase_id → {date_time, total_purchase_price} – OK!

PURCHASE (<u>purchase_id</u> , date_time, total_purchase_price)
--

PURCHASE DETAIL table:

purchase_detail_id → {purchase_id, isbn, purchase_quantity, purchase_price_per_book} – OK!

PURCHASE DETAIL (<u>purchase_detail_id</u> , purchase_id, isbn, purchase_quantity, purchase_price_per_book)
--

STOCK table:

isbn → {num_copies, title, year, avg_cost, pub_name, pub_address, pub_phone} – OK!

STOCK (<u>isbn</u> , num_copies, title, year, avg_cost, pub_id, pub_name, pub_address, pub_phone)
--

CUSTOMER table:

customer_id → {first_name, middle_name, last_name, phone, email, credits} – OK!

CUSTOMER (<u>customer_id</u> , first_name, middle_name, last_name, phone, email, credits)
--

CUSTOMER_ORDER table:

order_id → {customer_id, date_time, total_price, discount, final_price) – OK!

CUSTOMER_ORDER (<u>order_id</u> , customer_id, date_time, total_price, discount, final_price)
--

ORDER_DETAIL table:

{order_detail_id} → {order_id, isbn, quantity_ordered}- OK!

ORDER_DETAIL (<u>order_detail_id</u> , order_id, isbn, quantity_ordered)

3rd normal form

#all non-key attributes of a table must be functionally dependent on every candidate key i.e. there can be no interdependencies among non-key attributes.

PURCHASE table:

PURCHASE (<u>purchase_id</u> , date_time, total_purchase_price)
--

PURCHASE_DETAIL table:

- Here one might be tempted to think that if one knows the isbn he/she will also know the purchase_price of a particular order but, that would not be correct as we can purchase the same isbn from the same publisher for different prices at different times!
- We have two candidate keys of which one is the primary; all other attributes are fully dependent on the candidate keys; there is no interdependency among non-key attributes.
- The table is in 3rd NF.

purchase_detail_id → {purchase_id, isbn, purchase_quantity, purchase_price_per_book}
{purchase_id, isbn} → {purchase_detail_id, purchase_quantity, purchase_price_per_book} #a
candidate key – still ok according to our book!

PURCHASE_DETAIL (<u>purchase_detail_id</u> , purchase_id, isbn, purchase_quantity, purchase_price_per_book)
--

STOCK table:

- Here we can see that pub_name and pub_address are transitively dependent on the primary key through pub_phone. Hence, the table is not in 3nf. It is important to note that each publisher has a unique phone number and it is mandatory to have this data to be registered as a customer.
- pub_phone becomes the foreign key in the STOCK table
- recall, that we assumed that a unique ISBN is bought from the same publisher. So, if we know the isbn we know the publisher.

isbn \rightarrow {num_copies, title, year, avg_cost, pub_name, pub_address, pub_phone}

pub_phone \rightarrow {pub_name, pub_address}

STOCK (<u>isbn</u> , num_copies, title, year, avg_cost, pub_phone)

PUBLISHER (<u>pub_phone</u> , pub_name, pub_address)

CUSTOMER table:

- here we have three candidate keys: phone, email, and customer_id (which is the primary key)
- all other attributes are fully dependent on the candidate keys
- there are no interdependencies between non-candidate keys

customer_id \rightarrow {first_name, middle_name, last_name, phone, email, credits}

phone \rightarrow {customer_id, first_name, middle_name, last_name, email, credits}

email \rightarrow {customer_id, first_name, middle_name, last_name, phone, credits}

CUSTOMER (<u>customer_id</u> , first_name, middle_name, last_name, phone, email, credits)
--

CUSTOMER_ORDER table:

- if we know the total price, discount, and tax [13%] we can determine the final price thus the final_price is transitively dependent on the primary key
- the table is not in 3rd NF
- We created a new key for the new table receipt_id instead of using in order to ensure that each order has a unique receipt.

order_id \rightarrow {customer_id, date_time, total_price, discount, final_price}

{total_price, discount, tax} \rightarrow {final_price}

CUSTOMER_ORDER (<u>order_id</u> , customer_id, date_time, total_price, discount, receipt_id)

RECEIPT (<u>receipt_id</u> , final_price)
--

ORDER_DETAIL table:

- this table is in 3rd NF

{order_detail_id} \rightarrow {order_id, isbn, quantity_ordered} - OK!

{order_id, isbn} \rightarrow {quantity_ordered} – this is a candidate key - OK!

ORDER_DETAIL (order_detail_id, order_id, isbn, quantity_ordered)
--

Explanation of the ER – diagram:

- The ER diagram can be read in this order:
 Purchase ->has-> Purchase_detail->makes up -> Stock->published by/bought from ->Publisher
 Stock ->part of -> Order_detail -> belongs to-> Customer_order-> has-> Receipt
 Customer_order -> belongs to -> Customer
- In other words, Each purchase (transaction at a particular time made by the bookstore) can have many purchase details.
- Many purchase details [referring to a particular ISBN] make up a single stock of a unique ISBN.
- Each stock [unique ISBN] is bought from/published by a single publisher.
- A stock [unique ISBN] is part of one order detail. Many order details belong to a customer order. A customer order has one receipt. Many customer orders can belong to a single customer.
- Purchase details cannot exist without a purchase which make it a weak entity. Similarly an order detail and receipt cannot exist without a customer order and a customer order cannot exist without a customer. This makes, order detail, customer order and receipt weak entities. Total purchase price and average cost are derived from other attributes.

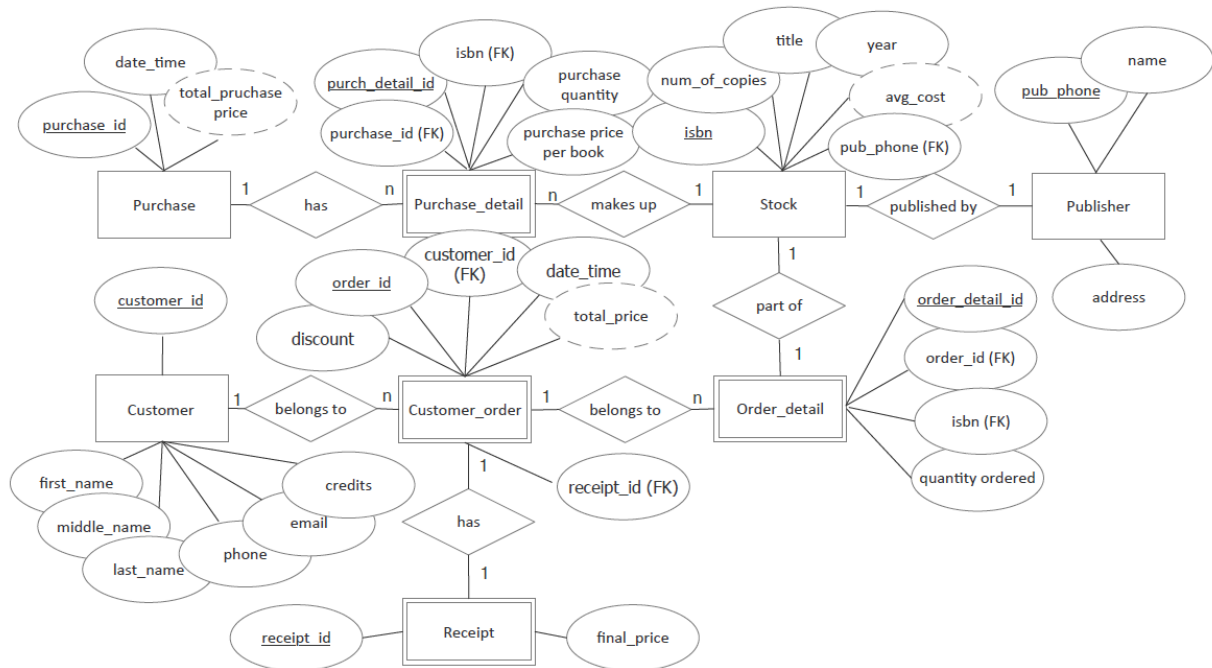
If we could go back would you do anything differently?

- Perhaps, writing the code for backend turned out to be about 700 lines of code; but, later we realized that a portion of the code was repetitive. Hence, if we had more time we would have created useful general functions/methods and make some of the code reusable and easier to follow.
- If we had more time we would have created a few more functionalities, such as canceling order.
- We believe the third normal form made our life more complicated since it would not allow for example total price, discount, tax and final price to be in the same table which we believe would be logical.

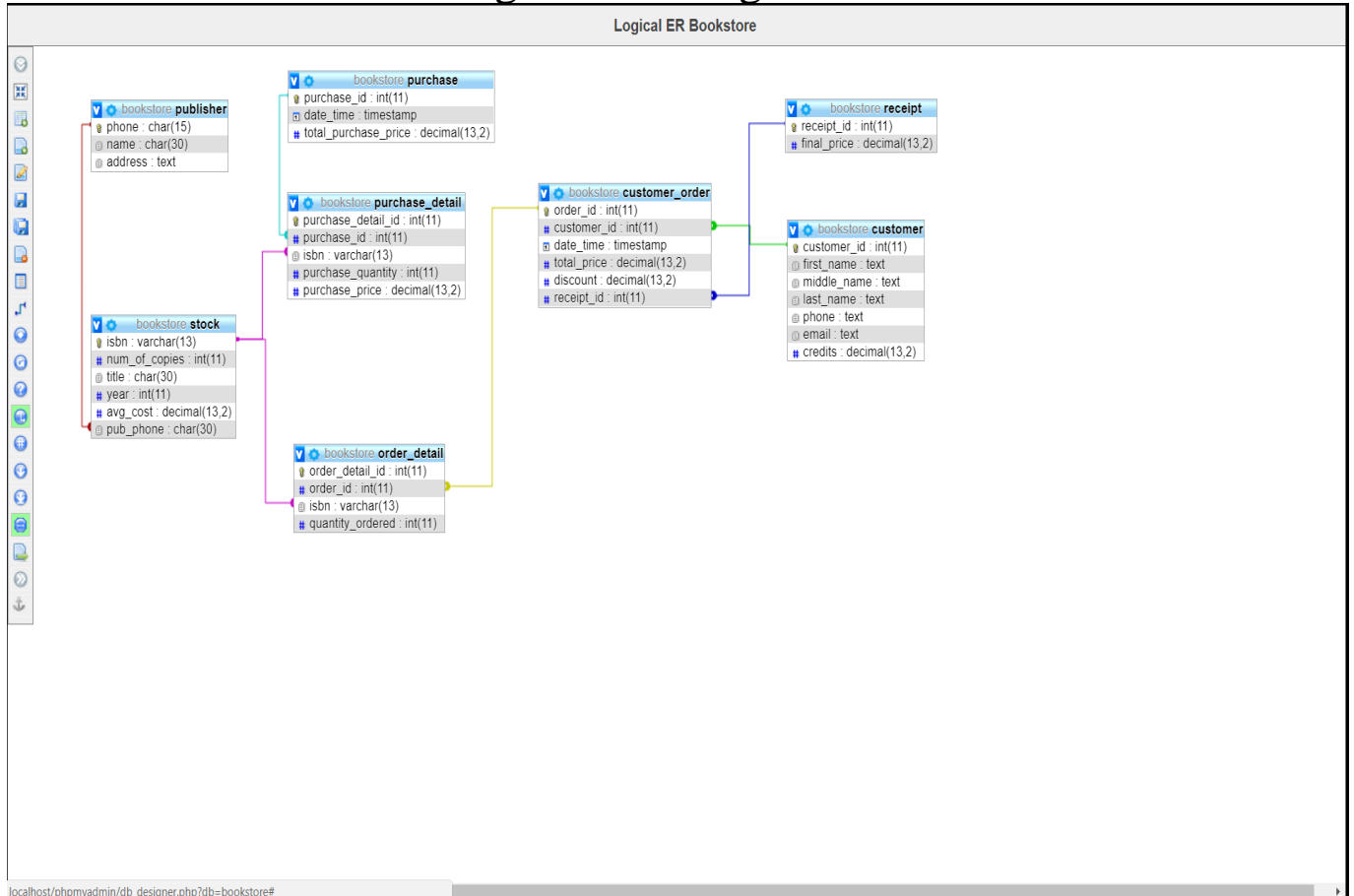
What was the most difficult?

- The most difficult was implementing some of the complex logic we challenged ourselves to implement.

ER Diagram for a Bookstore



Logical ER Diagram



Database

1) Customer Table

Server: 127.0.0.1 » Database: bookstore » Table: customer

Browse

Structure

SQL

Search

Insert

Export

Import

Privileges

Operations

Tracking

Tri

Table structure

Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	customer_id	int(11)		No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/>	2	first_name	text		No	None			Change Drop More
<input type="checkbox"/>	3	middle_name	text		Yes	None			Change Drop More
<input type="checkbox"/>	4	last_name	text		No	None			Change Drop More
<input type="checkbox"/>	5	phone	text		No	None			Change Drop More
<input type="checkbox"/>	6	email	text		No	None			Change Drop More
<input type="checkbox"/>	7	credits	decimal(13,2)		Yes	None			Change Drop More

Server: 127.0.0.1 » Database: bookstore » Table: customer

Browse

Structure

SQL

Search

Insert

Export

Import

Privileges

Operations

Tracking

Show query box

✓ Showing rows 0 - 0 (1 total, Query took 0.0014 seconds.)

```
SELECT * FROM `customer` WHERE 1
```

☐ Profiling [\[Edit in\]](#)

☐ Show all

Number of rows: 25

Filter rows:

+ Options

Edit Copy Delete

customer_idfirst_namemiddle_namelast_namephoneemailcredits

☐ 1prasadV Vaidya1234567891prasadvaidya@trentu.ca5.56

☐ Check all

With selected: Edit Copy Delete Export

☐ Show all

Number of rows: 25

Filter rows:

Query results operations

2) Customer Order Table

Server: 127.0.0.1 » Database: bookstore » Table: customer_order

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	order_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	customer_id	int(11)			No	None			Change Drop More
3	date_time	timestamp		on update CURRENT_TIMESTAMP	No	CURRENT_TIMESTAMP		ON UPDATE CURRENT_TIMESTAMP	Change Drop More
4	total_price	decimal(13,2)			No	None			Change Drop More
5	discount	decimal(13,2)			Yes	None			Change Drop More
6	receipt_id	int(11)			No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Fulltext Add to central columns Remove from central columns

Server: 127.0.0.1 » Database: bookstore » Table: customer_order

Showing rows 0 - 0 (1 total, Query took 0.0013 seconds.)

SELECT * FROM `customer_order`

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code]

Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

	order_id	customer_id	date_time	total_price	discount	receipt_id
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	2018-08-03 20:58:50	111.25	0.00	1

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 | Filter rows: Search this table

3) Order Detail Table

Server: 127.0.0.1 » Database: bookstore » Table: order_detail

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	order_detail_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	order_id	int(11)			No	None			Change Drop More
3	isbn	varchar(13)	utf8mb4_general_ci		No	None			Change Drop More
4	quantity_ordered	int(11)			No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Fulltext Add to central columns Remove from central columns

Showing rows 0 - 0 (1 total, Query took 0.0014 seconds.)

SELECT * FROM `order_detail`

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

	order_detail_id	order_id	isbn	quantity_ordered
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	1	1

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 | Filter rows: Search this table

4) Publisher table

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	phone	char(15)	utf8mb4_general_ci		No	None			Change Drop More
2	name	char(30)	utf8mb4_general_ci		No	None			Change Drop More
3	address	text	utf8mb4_general_ci		No	None			Change Drop More

Check all With selected: [Browse](#) [Change](#) [Drop](#) [Primary](#) [Unique](#) [Index](#) [Fulltext](#) [Add to central columns](#) [Remove from central columns](#)

Server: 127.0.0.1 » Database: bookstore » Table: publisher

Showing rows 0 - 0 (1 total, Query took 0.0013 seconds.)

`SELECT * FROM `publisher``

Profiling [\[Edit inline\]](#) [\[Edit\]](#) [\[Explain SQL\]](#) [\[Create PHP c](#)

Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

phone name address

Edit Copy Delete	1234567891	kjdbwkd	Ahj
--	------------	---------	-----

Check all With selected: [Edit](#) [Copy](#) [Delete](#) [Export](#)

Show all | Number of rows: 25 | Filter rows: Search this table

5) Purchase table

Server: 127.0.0.1 » Database: bookstore » Table: purchase

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	purchase_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	date_time	timestamp		on update CURRENT_TIMESTAMP	No	CURRENT_TIMESTAMP		ON UPDATE CURRENT_TIMESTAMP	Change Drop More
3	total_purchase_price	decimal(13,2)			No	None			Change Drop More

Check all With selected: [Browse](#) [Change](#) [Drop](#) [Primary](#) [Unique](#) [Index](#) [Fulltext](#) [Add to central columns](#) [Remove from central columns](#)

Server: 127.0.0.1 » Database: bookstore » Table: purchase

Showing rows 0 - 1 (2 total, Query took 0.0013 seconds.)

`SELECT * FROM `purchase``

Profiling [\[Edit inline\]](#) [\[Edit\]](#) [\[Explain SQL\]](#) [\[Create PHP cc](#)

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

	purchase_id	date_time	total_purchase_price
Edit Copy Delete	2	2018-08-03 20:48:08	1005.70
Edit Copy Delete	3	2018-08-03 21:05:25	6297.49

Check all With selected: [Edit](#) [Copy](#) [Delete](#) [Export](#)

6) Purchase detail table

Server: 127.0.0.1 » Database: bookstore » Table: purchase_detail

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	purchase_detail_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	purchase_id	int(11)			No	None			Change Drop More
3	isbn	varchar(13)	utf8mb4_general_ci		No	None			Change Drop More
4	purchase_quantity	int(11)			No	None			Change Drop More
5	purchase_price	decimal(13,2)			No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Fulltext Add to central columns Remove from central columns

Showing rows 0 - 5 (6 total, Query took 0.0013 seconds.)

SELECT * FROM `purchase_detail`

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code]

Show all Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	purchase_detail_id	purchase_id	isbn	purchase_quantity	purchase_price
<input type="checkbox"/> Edit Copy Delete	3	2	1	10	89.00
<input type="checkbox"/> Edit Copy Delete	4	3	1	10	89.00
<input type="checkbox"/> Edit Copy Delete	5	3	2	20	34.00
<input type="checkbox"/> Edit Copy Delete	6	3	3	6	78.00
<input type="checkbox"/> Edit Copy Delete	7	3	4	19	45.00
<input type="checkbox"/> Edit Copy Delete	8	3	5	40	67.00

Check all With selected: Edit Copy Delete Export

7) Receipt table

Server: 127.0.0.1 » Database: bookstore » Table: receipt

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	receipt_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	final_price	decimal(13,2)			No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index Fulltext Add to central columns Remove from central columns

Showing rows 0 - 0 (1 total, Query took 0.0016 seconds.)

SELECT * FROM `receipt`

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refr

Show all Number of rows: 25 Filter rows: Search this table

+ Options

	receipt_id	final_price
<input type="checkbox"/> Edit Copy Delete	1	125.71

Check all With selected: Edit Copy Delete Export

Show all Number of rows: 25 Filter rows: Search this table

8) Stock table

Server: 127.0.0.1 » Database: bookstore » Table: stock

[Browse](#) [Structure](#) [SQL](#) [Search](#) [Insert](#) [Export](#) [Import](#) [Privileges](#) [Operations](#) [Tracking](#) [Triggers](#)

[Table structure](#) [Relation view](#)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1 isbn	varchar(13)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/>	2 num_of_copies	int(11)			No	None			Change Drop More
<input type="checkbox"/>	3 title	char(30)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/>	4 year	int(11)			No	None			Change Drop More
<input type="checkbox"/>	5 avg_cost	decimal(13,2)			No	None			Change Drop More
<input type="checkbox"/>	6 pub_phone	char(30)	utf8mb4_general_ci		No	None			Change Drop More

⬆ ☐ Check all With selected: [Browse](#) [Change](#) [Drop](#) [Primary](#) [Unique](#) [Index](#) [Fulltext](#) [Add to central columns](#) [Remove from central columns](#)

Server: 127.0.0.1 » Database: bookstore » Table: stock

[Browse](#) [Structure](#) [SQL](#) [Search](#) [Insert](#) [Export](#) [Import](#) [Privileges](#) [Operations](#) [Tracking](#) [Triggers](#)

✓ Showing rows 0 - 4 (5 total, Query took 0.0015 seconds)

[SELECT * FROM `stock`](#)

☐ Profiling [\[Edit inline\]](#) [\[Edit\]](#) [\[Explain SQL\]](#) [\[Create PHP code\]](#) [\[Refresh\]](#)

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

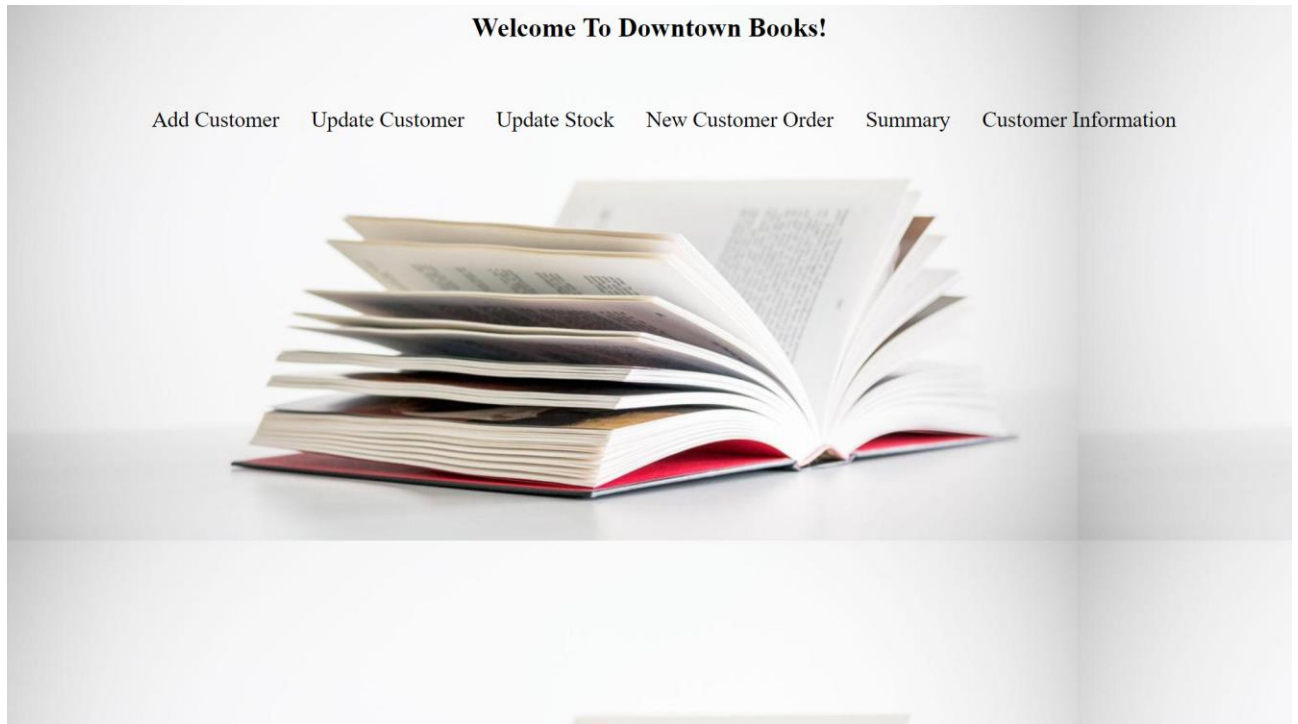
+ Options

		isbn	num_of_copies	title	year	avg_cost	pub_phone
<input type="checkbox"/>	Edit Copy Delete	1	19	NHY	2000	89.00	1234567891
<input type="checkbox"/>	Edit Copy Delete	2	20	Book2	2002	34.00	1234567891
<input type="checkbox"/>	Edit Copy Delete	3	6	Book3	2003	78.00	1234567891
<input type="checkbox"/>	Edit Copy Delete	4	19	Book4	2004	45.00	1234567891
<input type="checkbox"/>	Edit Copy Delete	5	40	Book5	2005	67.00	1234567891

⬆ ☐ Check all With selected: [Edit](#) [Copy](#) [Delete](#) [Export](#)

FRONT END

- 1) **Main page:** In the home page there are four forms and two summary tables. On clicking on each text respective element will open.




- 2) **Add Customer:** In Add customer form a new customer can be added where the FirstName, last name, Email , phone are required fields. Phone should have ten digits only numeric. Email should have “@”.


Welcome To Downtown Books!

Add Customer Update Customer Update Stock New Customer Order Summary Customer Information

Add Customer

First Name*: Prasad Middle Name*: Last Name*: Vaidya Phone*: 1234567 Email*: pras

 Please match the requested format.





Welcome To Downtown Books!

Add Customer Update Customer Update Stock New Customer Order Summary Customer Information

Add Customer

First Name*: Prasad Middle Name*: Last Name*: Vaidya Phone*: 1234567891 Email*: prad

 Please include an '@' in the email address. 'pras' is missing an '@'.



A new customer record for **Prasad Vaidya** with customer id: **1** created. Prasad Vaidya has **\$0.00** credits.


- 3) **Update Customer:** Customer Info can be updated in this form. If a customer doesnot exist there will be a message. This is checked based on customer id.

Welcome To Downtown Books!

Add Customer Update Customer Update Stock New Customer Order Summary Customer Information

Update Customer Information

CustID*: 3 First Name*: prasad Middle Name: V Last Name*: Vaidya Phone*: 1234567891 Email*:
prasadvaidya@trentu.ca




Error - cannot update Customer Info with customer id: **3**. No existing customer with customer id: **3**.

Welcome To Downtown Books!

Add Customer Update Customer Update Stock New Customer Order Summary Customer Information

Update Customer Information

CustID*: 1 First Name*: prasad Middle Name: V Last Name*: Vaidya Phone*: 1234567891 Email*:
prasadvaidya@trentu.ca



Record belonging to **prasad Vaidya** with customer id: **1** successfully updated.

- 4) **Update Stock:** Stock can be update in this form. All fields are required to have value. One can also add multiple Books in the same form by clicking on **Add more** text besides the field. Also for the consequent fields the publisher details can be fetched from the first record by double clicking on the publisher field space.

Welcome To Downtown Books!

Add Customer Update Customer Update Stock New Customer Order Summary Customer Information

Update Stock

ISBN*: 1	Title*: NHY	Year*: 2000	Quantity*: 10	Cost per Book*: 89
Publisher name*: kjdbwkd	Publisher phone*: 1234567891	Publisher address*: Ahj	Add More	

SubmitReset

After the new purchase we have: 10 copies of ISBN: 1 (title: NHY). in stock.

Welcome To Downtown Books!

Add Customer Update Customer Update Stock New Customer Order Summary Customer Information

Update Stock

ISBN*: 1	Title*: NHY	Year*: 2000	Quantity*: 10	Cost per Book*: 89
Publisher name*: kjdbwkd	Publisher phone*: 1234567891	Publisher address*: Ahj	Add More	

ISBN*: <input type="text"/>	Title*: <input type="text"/>	Year*: <input type="text"/>	Quantity*: <input type="text"/>	Cost per Book*: <input type="text"/>
Publisher name*: kjdbwkd	Publisher phone*: 1234567891	Publisher address*: <input type="text"/>	<input type="text" value="X"/>	

Please fill out this field

SubmitReset

- 5) **New customer order:** New customer order can be added from this form. Customer ID must be present in the database. Existing customer can also choose to apply credit or not to his/her purchase. All fields are required to have value. One can also add multiple Books in the same form by clicking on **Add more** text besides the field. Also for the consequent fields the publisher details can be fetched from the first record by double clicking on the publisher field space.

Welcome To Downtown Books!

Add Customer Update Customer Update Stock New Customer Order Summary Customer Information

New Customer Order

Customer ID*: ISBN*: Quantity*: Apply Credits*: ☐ Yes ☐ No [Add More](#)

Submit Reset

WHEN CUSTOMER IS NOT PRESENT IN DB

Welcome To Downtown Books!

Add Customer Update Customer Update Stock New Customer Order Summary Customer Information

New Customer Order

Customer ID*: 3 ISBN*: 78 Quantity*: 1 Apply Credits*: ☒ Yes ☐ No [Add More](#)

Submit Reset

Error - cannot make purchase. No existing customer with customer id: 3.


WHEN CUSTOMER IS PRESENT IN DB

Welcome To Downtown Books!

[Add Customer](#) [Update Customer](#) [Update Stock](#) [New Customer Order](#) [Summary](#) [Customer Information](#)

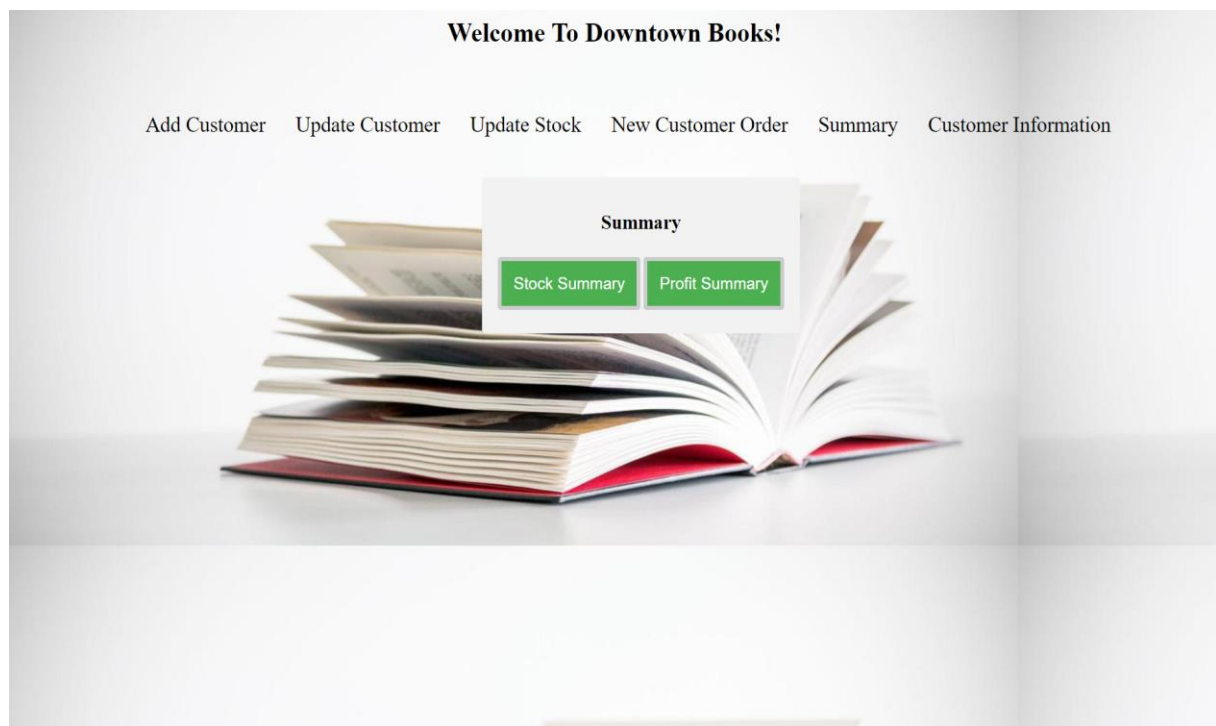
New Customer Order

Customer ID*: ISBN*: Quantity*: Apply Credits*: ☒ Yes ☐ No [Add More](#)



A new customer order with receipt id: 1 added for customer id: 1. The total price including taxes is \$125.71. The customer has \$5.56 credits left.

- 6) **Summary:** This form has two buttons which gives the Profit summary for the past one month from the time of submit and Stock Summary of the books in inventory by ascending order of stock to help in better inventory management.



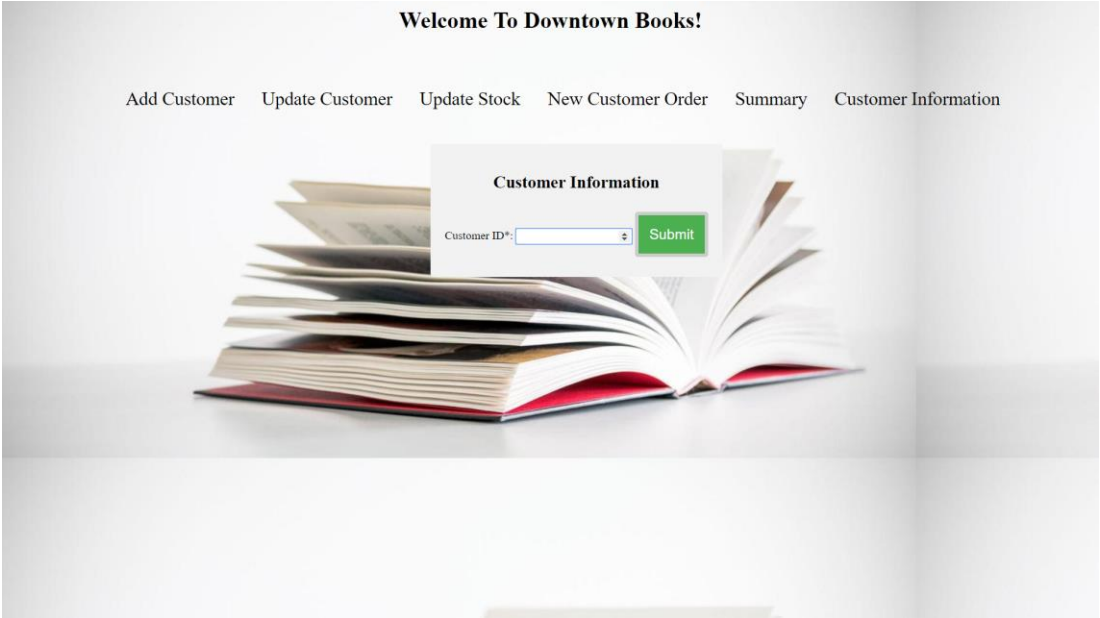
STOCK INFORMATION

ISBN	Title	Year	Stock
3	Book3	2003	6
1	NHY	2000	19
4	Book4	2004	19
2	Book2	2002	20
5	Book5	2005	40

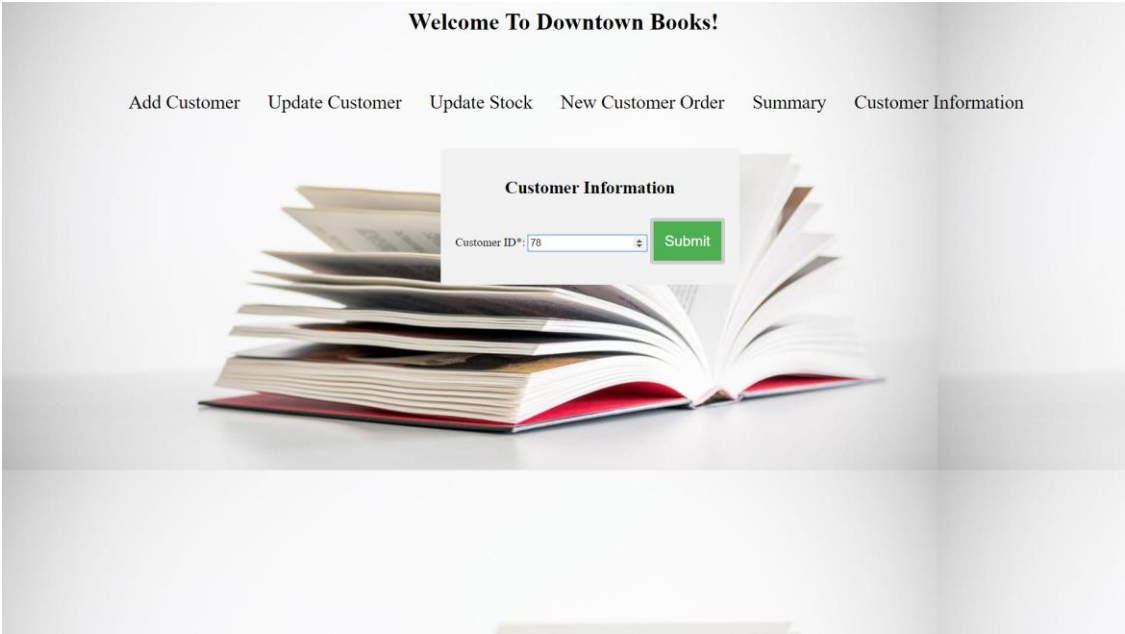
PROFIT FROM PREVIOUS MONTH

Revenue	Cost	Profit
125.71	89.00	36.71

7) **Customer Information:** This form gives the information about customer. Like the first name, last name, Email, phone and credits. If the customer ID doesn't exist in the table, it will give an empty table as an output.



WHEN CUSTOMER_ID DOESN'T EXIST IN DB



CUSTOMER INFORMATION

Result

Customer ID	First Name	Middle Name	Last Name	Phone	Email	Credit
-------------	------------	-------------	-----------	-------	-------	--------


WHEN CUSTOMER_ID EXIST IN DB

Welcome To Downtown Books!

Add Customer Update Customer Update Stock New Customer Order Summary Customer Information

Customer Information

Customer ID*:



CUSTOMER INFORMATION

Customer ID	First Name	Middle Name	Last Name	Phone	Email	Credit
1	prasad	V	Vaidya	1234567891	prasadvaidya@trentu.ca	5.56