SOUTHEAST MISSOURI
STATE UNIVERSITY · 1873®

CS440 FINAL PROJECT

The SQL Squad

FALL 2024:

Thabo Adams
Joe Samples
Noah Schuchardt
Kyle Schwartz

Contents

1) Abstract

Every organization typically requires some form of database system. If an organization operates online, the database becomes an essential aspect of the company. A database system in a commercial setting is the backbone of an organization, providing information on everything from customer data, order information, and inventory status. This is why much care is taken in the development of such systems. Different companies design their e-commerce websites differently depending on the available resources and security needed within an organization. Such systems go through rigorous evaluations to ensure they are functioning up to the standards of the organization. In this project, we developed a bookstore management database capable of handling customer registration, order processing, product information, and inventory management. The developmental process is long and requires attention to detail to ensure that the system is adequate to meet the needs of the organization. Our team was able to create a database that meets these needs. Although we did not have time to implement all the technologies a bookstore would need to open an e-commerce business, we did establish the basics of what would be required to develop the necessary applications to manage such a business.

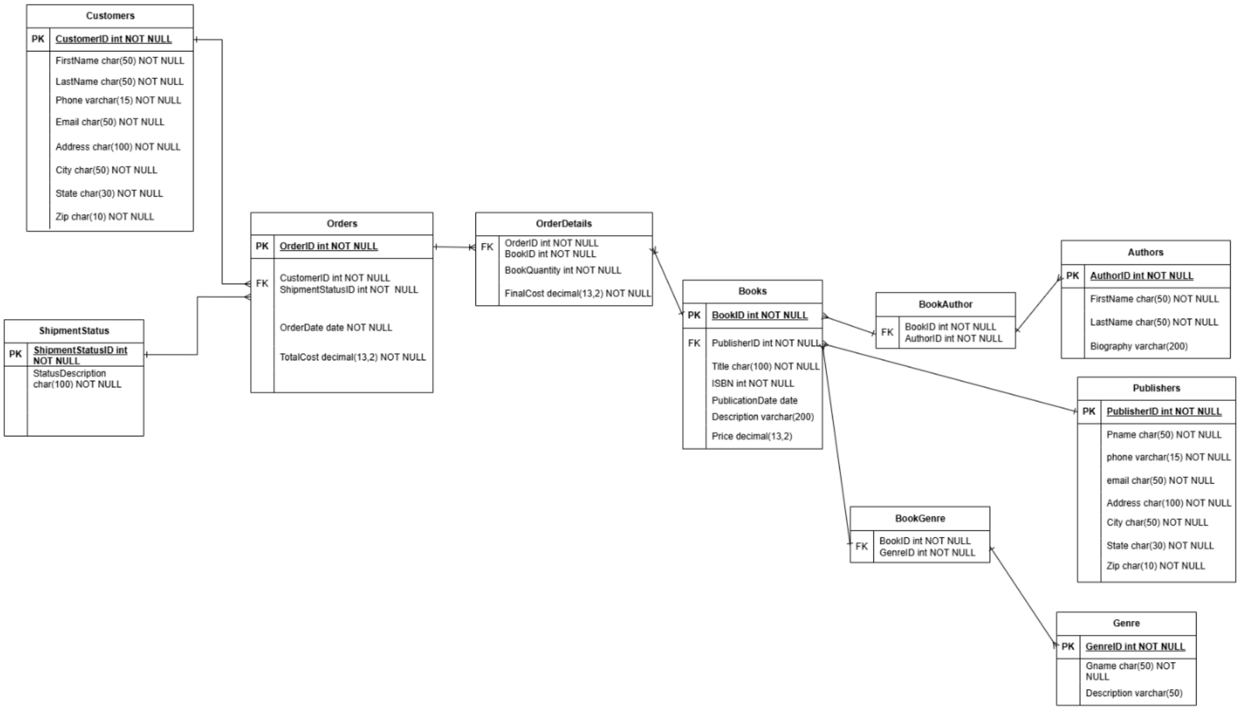
2) Introduction

For this project, we decided to develop a bookstore database like the one that may be used at organizations such as Barnes and Noble. We wanted a database design that could help track customer, shipping, product, and order information. Most book retailers, large and small, experience the challenge of having to maintain a massive inventory while ensuring that their customers receive what they order in a timely fashion. Because of this, creating and maintaining an efficient database system is essential to handle the complex relationships between customers, orders, books, and the metadata that is associated with them, such as publishers, authors, genres, and order details. This project is focused around designing a relational database that encapsulates these entities and how they interact, providing a robust foundation for managing shipments and the other operational processes that come along with being a book retailer.

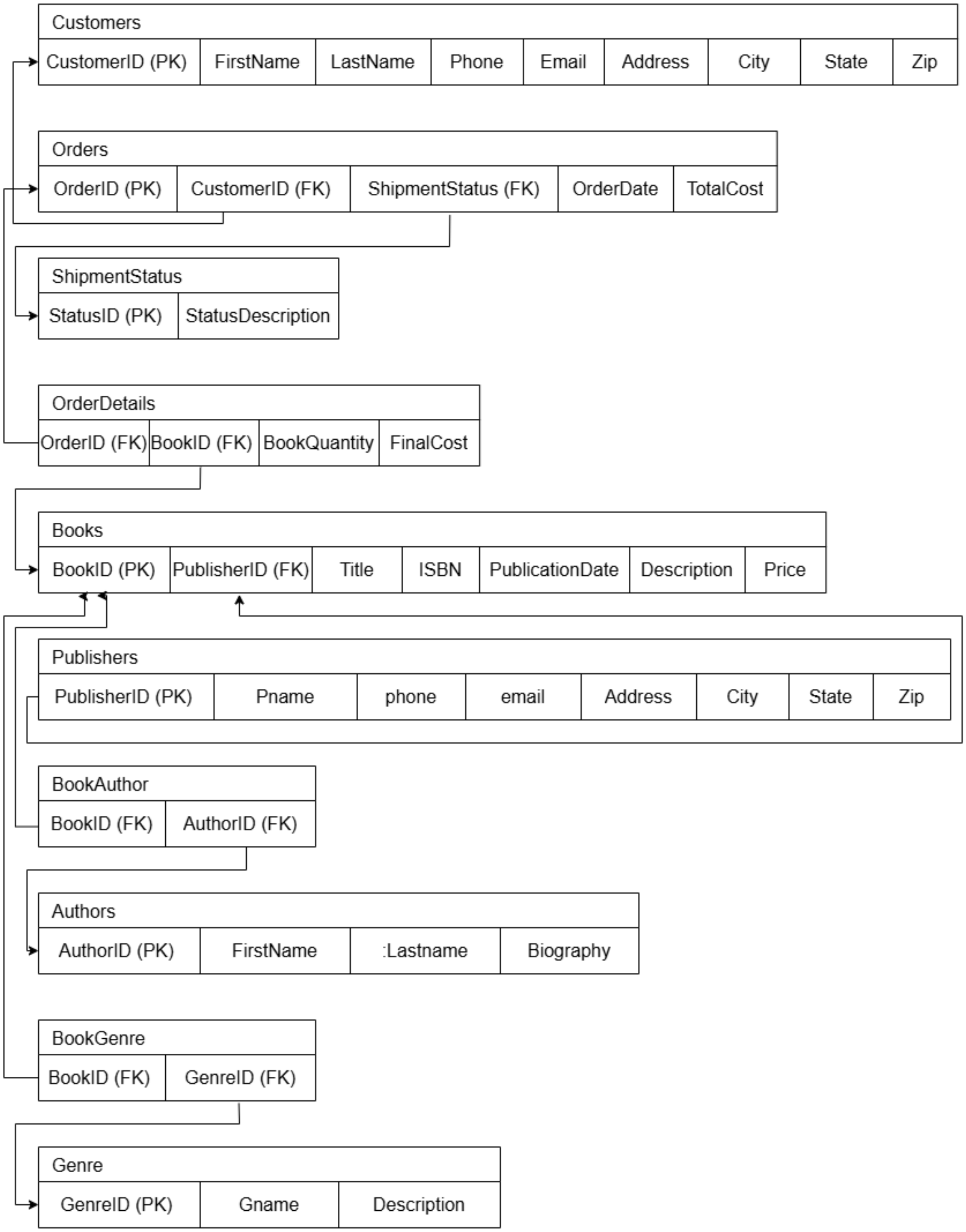
This database is designed with the purpose of helping the book retailer develop a better grasp of how they can expand their business by completing practical tasks, such as identifying the books that are selling well, tracking the order histories of customers, and seeing how certain publishers and genres perform with the average customer. In addition, the database will allow for the detailed management of inventory and shipping in order to ensure that operations run smoothly, even as the business expands. Through the leveraging of SQL, the database will enable the precise and complex retrieval of data, helping book retailers to optimize their workflows and create a high-quality customer experience.

The project follows a structured approach to database design, development, and implementation, with each step being guided by a set of deliverables and deadlines. The intended outcome is a database that supports the primary needs of a book retailer in the areas of operational management and strategic planning.

3) E-R Diagram


Customers
PK
CustomerID int NOT NULL
FirstName char(50) NOT NULL
LastName char(50) NOT NULL
Phone varchar(15) NOT NULL
Email char(50) NOT NULL
Address char(100) NOT NULL
City char(50) NOT NULL
State char(30) NOT NULL
Zip char(10) NOT NULL
Orders
PK
OrderID int NOT NULL
FK
CustomerID int NOT NULL
ShipmentStatusID int NOT NULL
OrderDate date NOT NULL
TotalCost decimal(13,2) NOT NULL
OrderDetails
FK
OrderID int NOT NULL
BookID int NOT NULL
BookQuantity int NOT NULL
FinalCost decimal(13,2) NOT NULL
ShipmentStatus
PK
ShipmentStatusID int NOT NULL
StatusDescription char(100) NOT NULL
Books
PK
BookID int NOT NULL
FK
PublisherID int NOT NULL
Title char(100) NOT NULL
ISBN int NOT NULL
PublicationDate date
Description varchar(200)
Price decimal(13,2)
BookAuthor
FK
BookID int NOT NULL
AuthorID int NOT NULL
Authors
PK
AuthorID int NOT NULL
FirstName char(50) NOT NULL
LastName char(50) NOT NULL
Biography varchar(200)
Publishers
PK
PublisherID int NOT NULL
Pname char(50) NOT NULL
phone varchar(15) NOT NULL
email char(50) NOT NULL
Address char(100) NOT NULL
City char(50) NOT NULL
State char(30) NOT NULL
Zip char(10) NOT NULL
BookGenre
FK
BookID int NOT NULL
GenreID int NOT NULL
Genre
PK
GenreID int NOT NULL
Gname char(50) NOT NULL
Description varchar(50)

4) Relational Schema



Customers

CustomerID (PK)	FirstName	LastName	Phone	Email	Address	City	State	Zip

Orders

OrderID (PK)	CustomerID (FK)	ShipmentStatus (FK)	OrderDate	TotalCost

ShipmentStatus

StatusID (PK)	StatusDescription

OrderDetails

OrderID (FK)	BookID (FK)	BookQuantity	FinalCost

Books

BookID (PK)	PublisherID (FK)	Title	ISBN	PublicationDate	Description	Price

Publishers

PublisherID (PK)	Pname	phone	email	Address	City	State	Zip

BookAuthor

BookID (FK)	AuthorID (FK)

Authors

AuthorID (PK)	FirstName	:Lastname	Biography

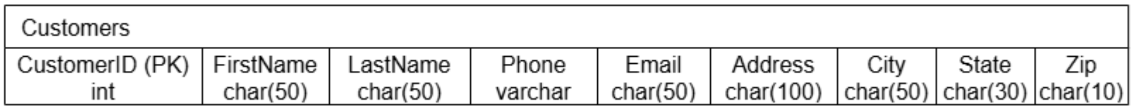
BookGenre

BookID (FK)	GenreID (FK)

Genre

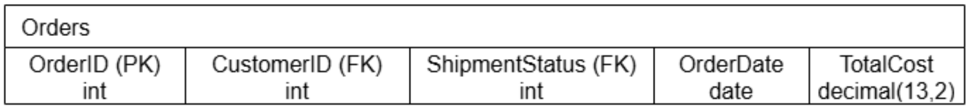
GenreID (PK)	Gname	Description

5) Data Types



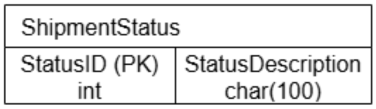
Customers

CustomerID (PK)	FirstName	LastName	Phone	Email	Address	City	State	Zip
int	char(50)	char(50)	varchar	char(50)	char(100)	char(50)	char(30)	char(10)



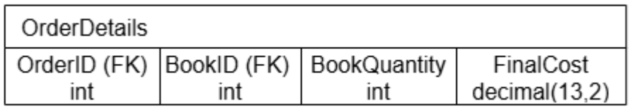
Orders

OrderID (PK)	CustomerID (FK)	ShipmentStatus (FK)	OrderDate	TotalCost
int	int	int	date	decimal(13,2)



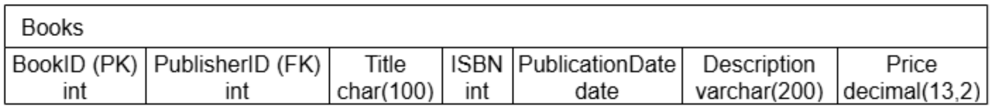
ShipmentStatus

StatusID (PK)	StatusDescription
int	char(100)



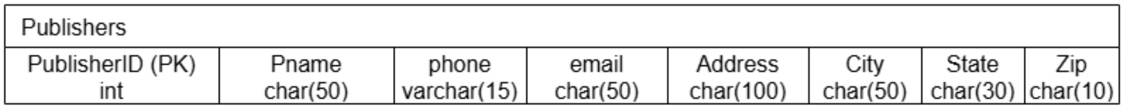
OrderDetails

OrderID (FK)	BookID (FK)	BookQuantity	FinalCost
int	int	int	decimal(13,2)



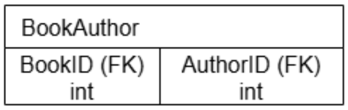
Books

BookID (PK)	PublisherID (FK)	Title	ISBN	PublicationDate	Description	Price
int	int	char(100)	int	date	varchar(200)	decimal(13,2)



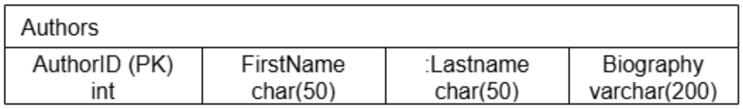
Publishers

PublisherID (PK)	Pname	phone	email	Address	City	State	Zip
int	char(50)	varchar(15)	char(50)	char(100)	char(50)	char(30)	char(10)



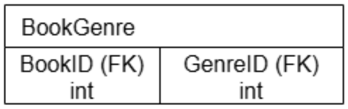
BookAuthor

BookID (FK)	AuthorID (FK)
int	int



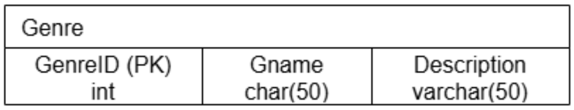
Authors

AuthorID (PK)	FirstName	:Lastname	Biography
int	char(50)	char(50)	varchar(200)



BookGenre

BookID (FK)	GenreID (FK)
int	int



Genre

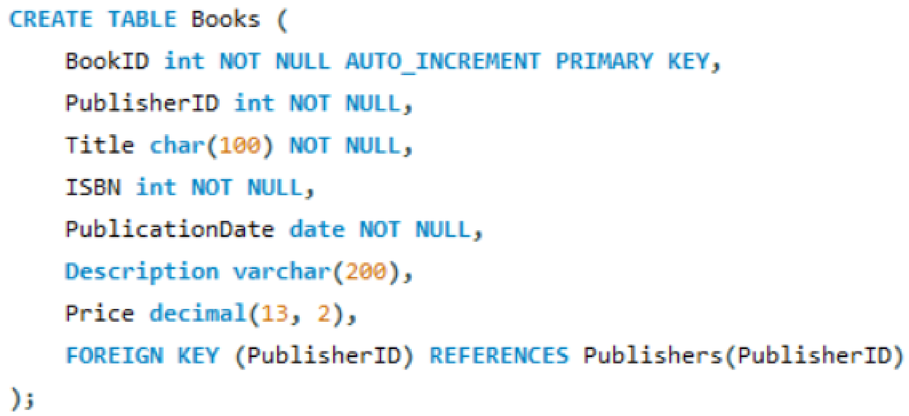
GenreID (PK)	Gname	Description
int	char(50)	varchar(50)

6) CRUD Matrix: Form versus Tables

	Customer	Author	Bookstore Employee	Bookstore Manager	Warehouse Employee	Warehouse Manager	Delivery Driver	Publisher
Customers	CR		CRU	CRUD				
Orders	CR		R	RU	CR	RU		
Order Details			R	RU	RU	RU		
Books	R	R	CRU	CRUD	R	RU		CRU
BookAuthor		R	R	RUD				RU
Authors		RU	R	CRUD				
Publishers			R	CRUD				
BookGenre	R		R	RUD				
Genres	R		R	CRUD				
Shipment Status	R		R	RU	CRU	RU		

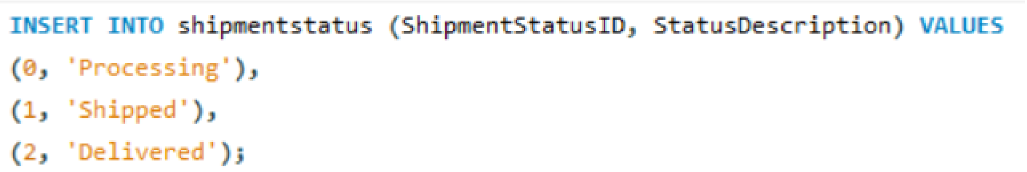
7) Table Description

Example SQL code used to create the books table in the database (other CREATE TABLE code was left out to keep the document more organized).



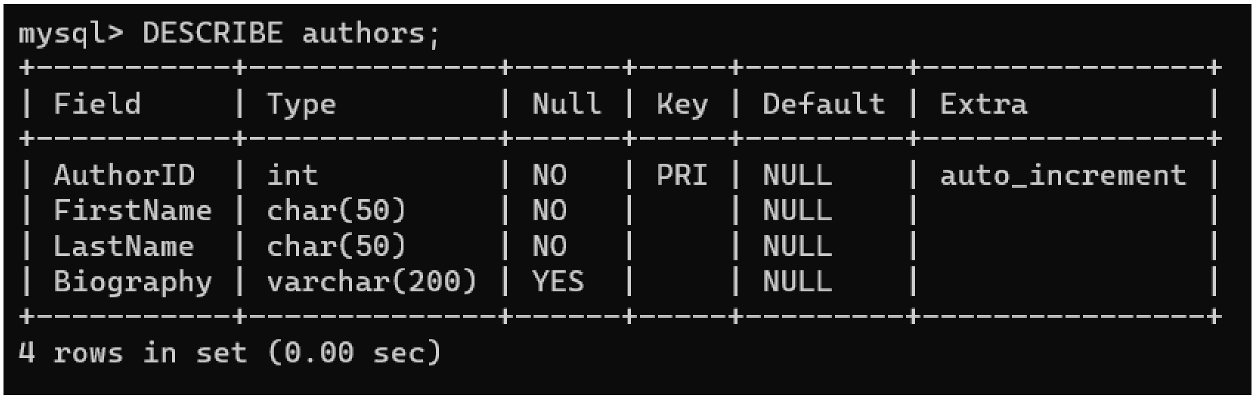
```
CREATE TABLE Books (
    BookID int NOT NULL AUTO_INCREMENT PRIMARY KEY,
    PublisherID int NOT NULL,
    Title char(100) NOT NULL,
    ISBN int NOT NULL,
    PublicationDate date NOT NULL,
    Description varchar(200),
    Price decimal(13, 2),
    FOREIGN KEY (PublisherID) REFERENCES Publishers(PublisherID)
);
```

Example INSERT Statement used to add data to the reference table ShipmentStatus.



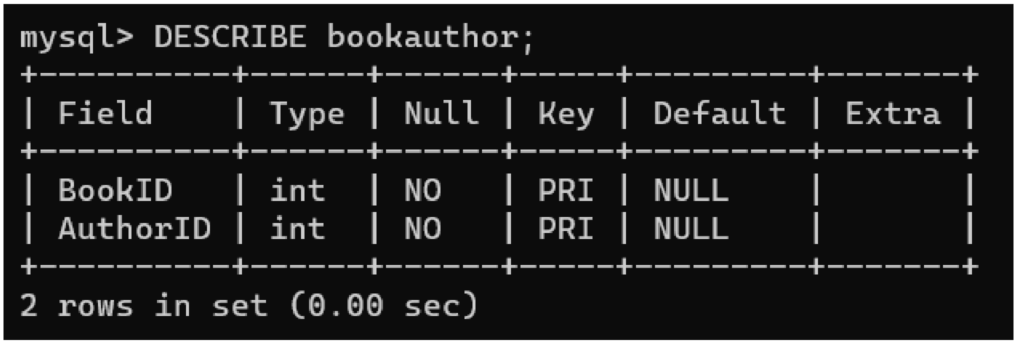
```
INSERT INTO shipmentstatus (ShipmentStatusID, StatusDescription) VALUES
(0, 'Processing'),
(1, 'Shipped'),
(2, 'Delivered');
```

Authors Description



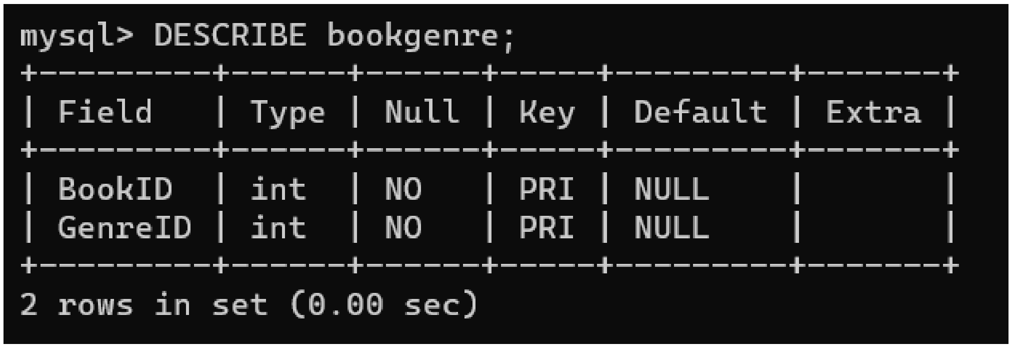
```
mysql> DESCRIBE authors;
+-----------+--------------+------+-----+---------+----------------+
| Field     | Type         | Null | Key | Default | Extra          |
+-----------+--------------+------+-----+---------+----------------+
| AuthorID  | int          | NO   | PRI | NULL    | auto_increment |
| FirstName | char(50)     | NO   |     | NULL    |                |
| LastName  | char(50)     | NO   |     | NULL    |                |
| Biography | varchar(200) | YES  |     | NULL    |                |
+-----------+--------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

BookAuthor Description



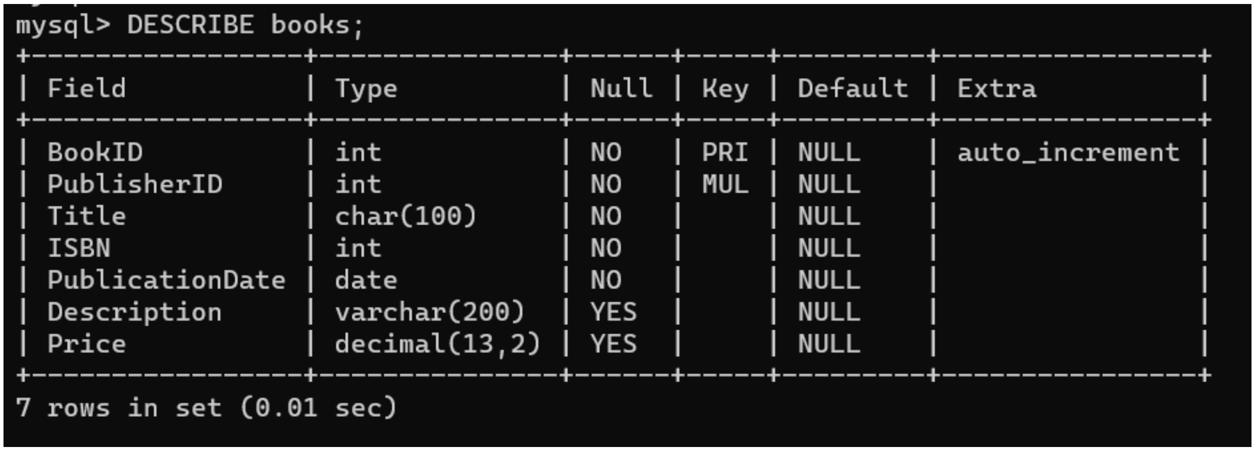
```
mysql> DESCRIBE bookauthor;
+----------+------+------+-----+---------+-------+
| Field    | Type | Null | Key | Default | Extra |
+----------+------+------+-----+---------+-------+
| BookID   | int  | NO   | PRI | NULL    |       |
| AuthorID | int  | NO   | PRI | NULL    |       |
+----------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

BookGenre Description



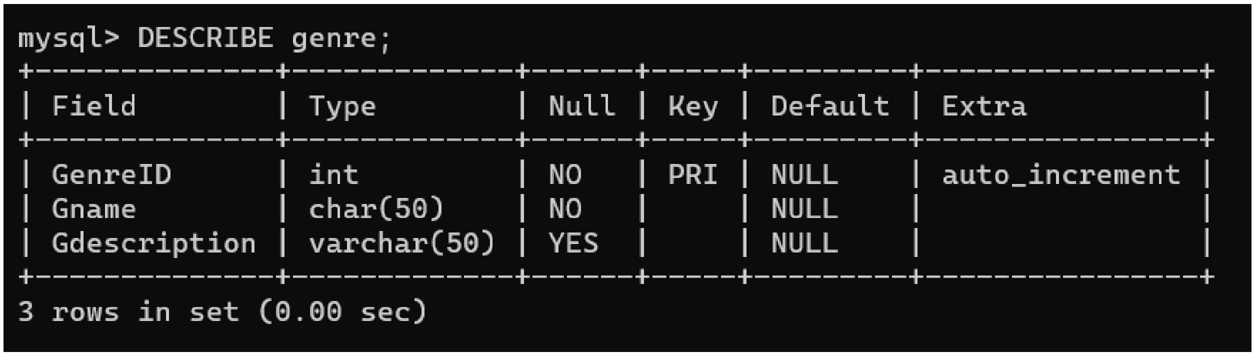
```
mysql> DESCRIBE bookgenre;
+---------+------+------+-----+---------+-------+
| Field   | Type | Null | Key | Default | Extra |
+---------+------+------+-----+---------+-------+
| BookID  | int  | NO   | PRI | NULL    |       |
| GenreID | int  | NO   | PRI | NULL    |       |
+---------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Book Description



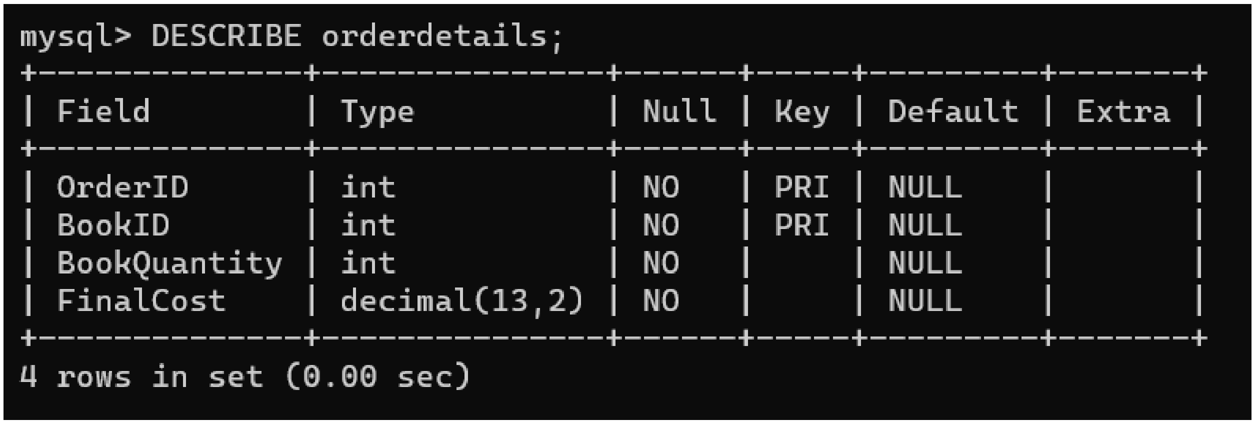
```
mysql> DESCRIBE books;
+-----------------+---------------+------+-----+---------+----------------+
| Field           | Type          | Null | Key | Default | Extra          |
+-----------------+---------------+------+-----+---------+----------------+
| BookID          | int           | NO   | PRI | NULL    | auto_increment |
| PublisherID     | int           | NO   | MUL | NULL    |                |
| Title           | char(100)     | NO   |     | NULL    |                |
| ISBN            | int           | NO   |     | NULL    |                |
| PublicationDate | date          | NO   |     | NULL    |                |
| Description     | varchar(200)  | YES  |     | NULL    |                |
| Price           | decimal(13,2) | YES  |     | NULL    |                |
+-----------------+---------------+------+-----+---------+----------------+
7 rows in set (0.01 sec)
```

Genre Description



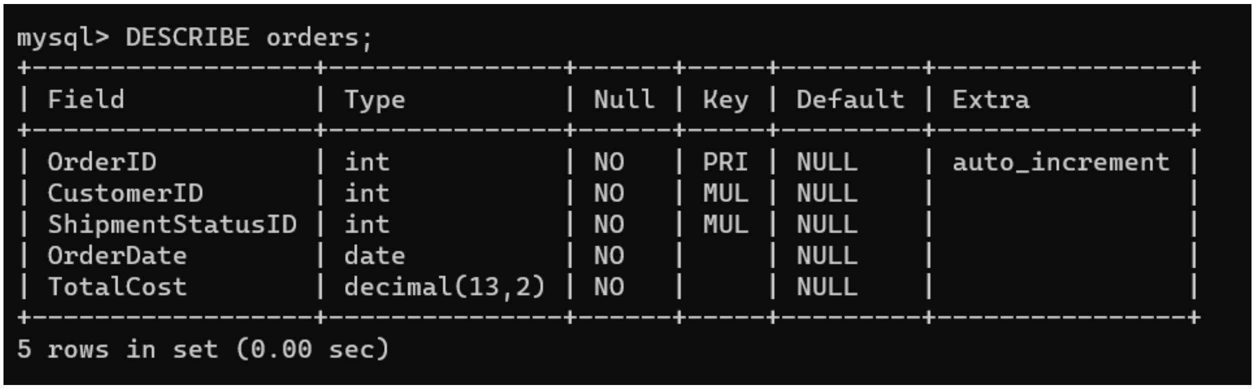
```
mysql> DESCRIBE genre;
+--------------+-------------+------+-----+---------+----------------+
| Field        | Type        | Null | Key | Default | Extra          |
+--------------+-------------+------+-----+---------+----------------+
| GenreID      | int         | NO   | PRI | NULL    | auto_increment |
| Gname        | char(50)    | NO   |     | NULL    |                |
| Gdescription | varchar(50) | YES  |     | NULL    |                |
+--------------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

OrderDetails Description



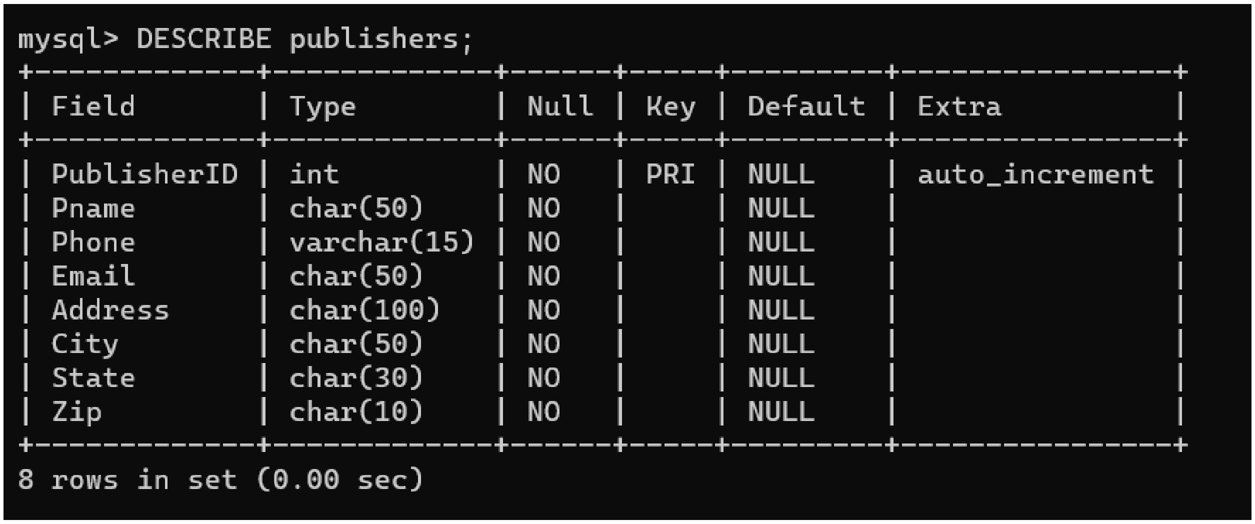
```
mysql> DESCRIBE orderdetails;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| OrderID      | int           | NO   | PRI | NULL    |       |
| BookID       | int           | NO   | PRI | NULL    |       |
| BookQuantity | int           | NO   |     | NULL    |       |
| FinalCost    | decimal(13,2) | NO   |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Orders Description



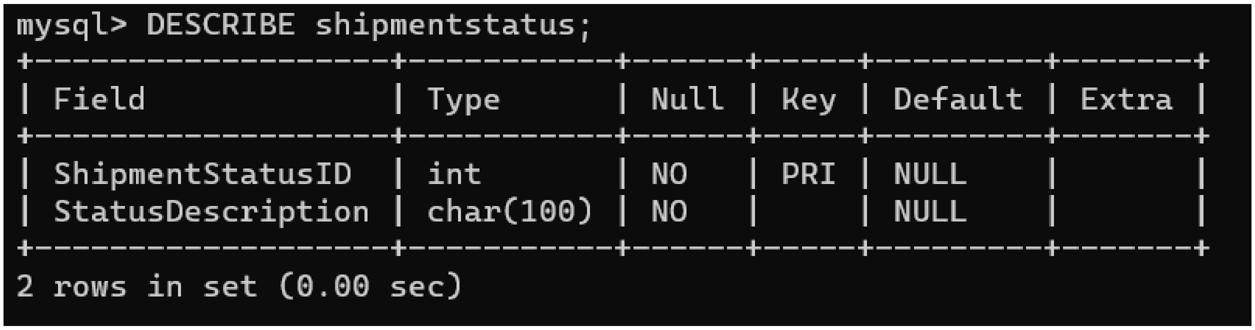
```
mysql> DESCRIBE orders;
+------------------+---------------+------+-----+---------+----------------+
| Field            | Type          | Null | Key | Default | Extra          |
+------------------+---------------+------+-----+---------+----------------+
| OrderID          | int           | NO   | PRI | NULL    | auto_increment |
| CustomerID       | int           | NO   | MUL | NULL    |                |
| ShipmentStatusID | int           | NO   | MUL | NULL    |                |
| OrderDate        | date          | NO   |     | NULL    |                |
| TotalCost        | decimal(13,2) | NO   |     | NULL    |                |
+------------------+---------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

Publishers Description



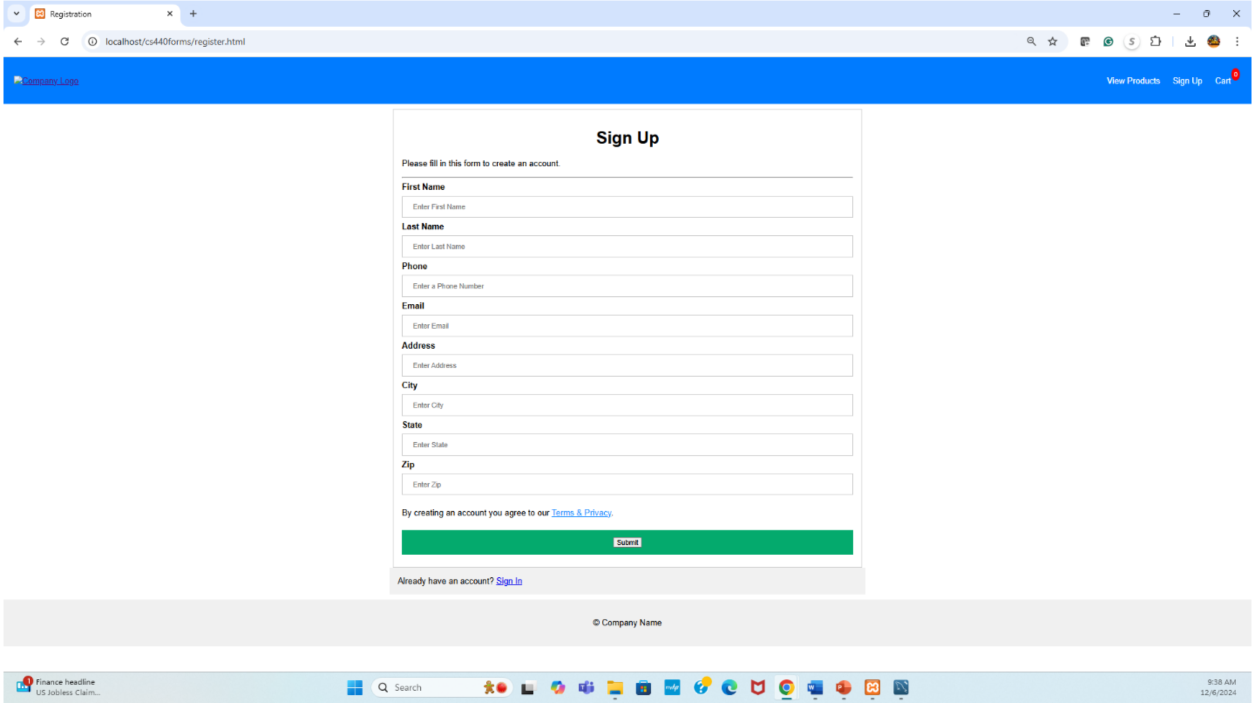
```
mysql> DESCRIBE publishers;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| PublisherID | int         | NO   | PRI | NULL    | auto_increment |
| Pname       | char(50)    | NO   |     | NULL    |                |
| Phone       | varchar(15) | NO   |     | NULL    |                |
| Email       | char(50)    | NO   |     | NULL    |                |
| Address     | char(100)   | NO   |     | NULL    |                |
| City        | char(50)    | NO   |     | NULL    |                |
| State       | char(30)    | NO   |     | NULL    |                |
| Zip         | char(10)    | NO   |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
8 rows in set (0.00 sec)
```

ShipmentStatus Description



```
mysql> DESCRIBE shipmentstatus;
+-------------------+-----------+------+-----+---------+-------+
| Field             | Type      | Null | Key | Default | Extra |
+-------------------+-----------+------+-----+---------+-------+
| ShipmentStatusID  | int       | NO   | PRI | NULL    |       |
| StatusDescription | char(100) | NO   |     | NULL    |       |
+-------------------+-----------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

8) Forms



Sign Up

Please fill in this form to create an account.

First Name

Enter First Name

Last Name

Enter Last Name

Phone

Enter a Phone Number

Email

Enter Email

Address

Enter Address

City

Enter City

State

Enter State

Zip

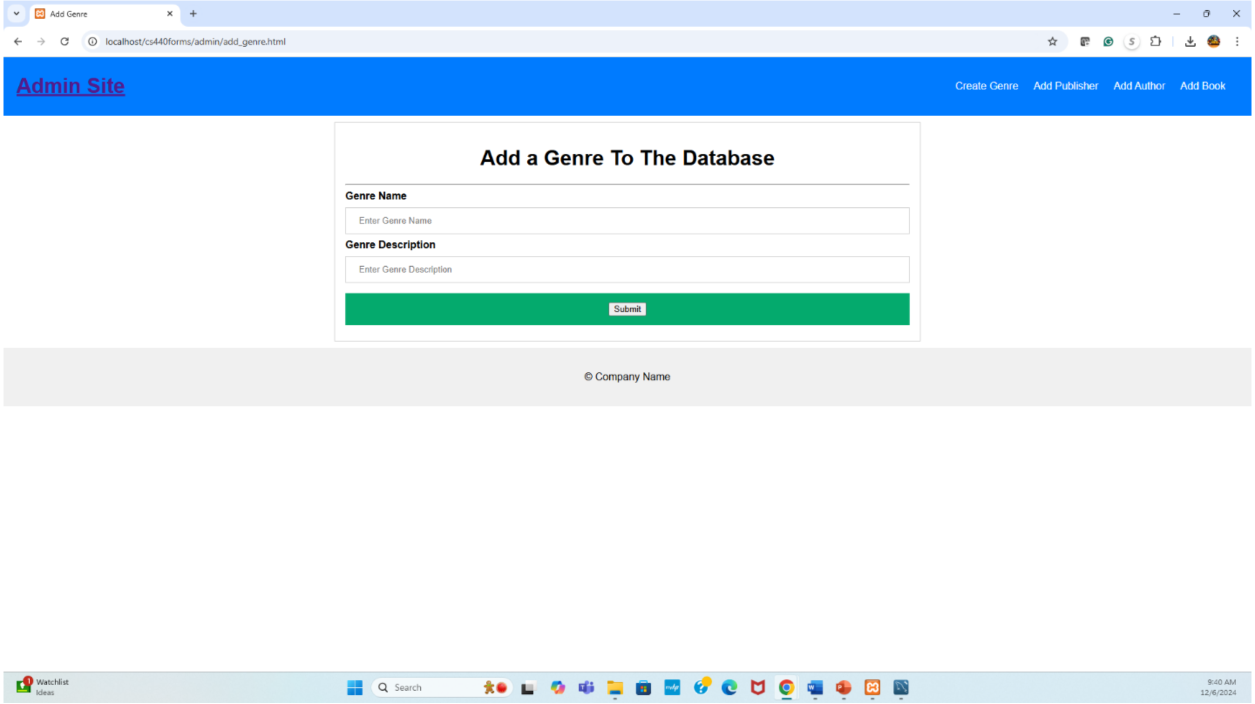
Enter Zip

By creating an account you agree to our Terms & Privacy.

Submit

Already have an account? Sign In

© Company Name



Add a Genre To The Database

Genre Name

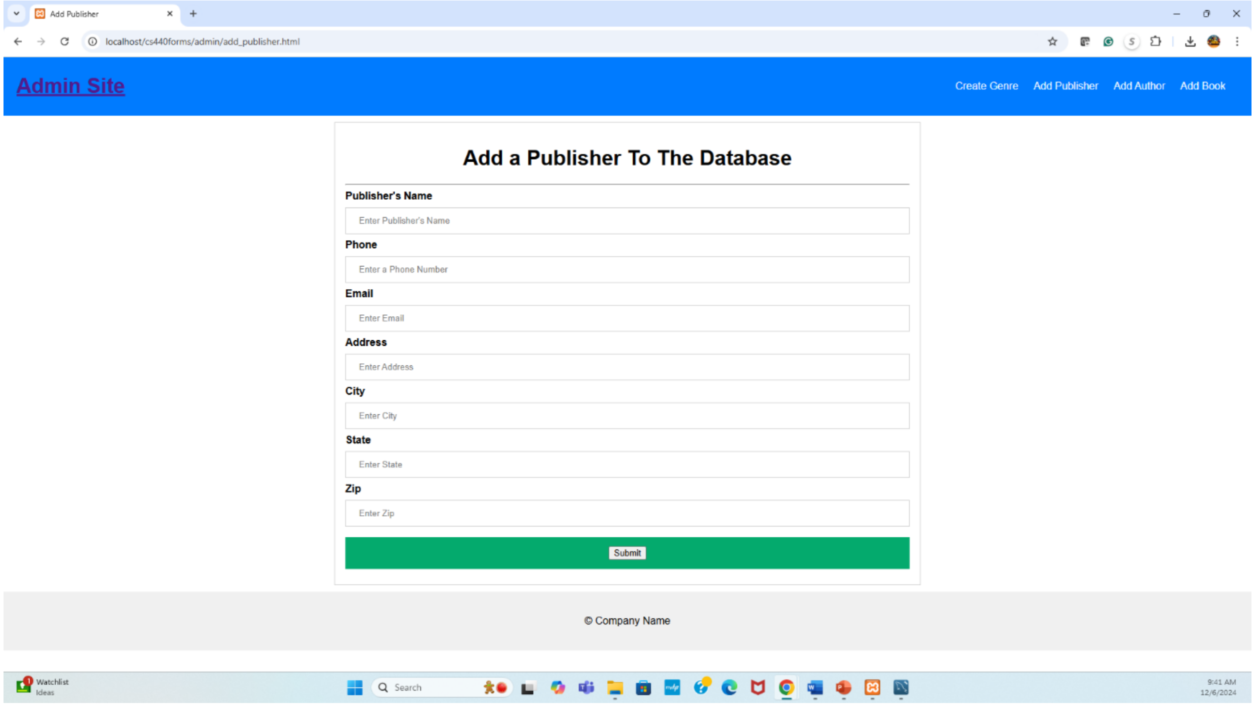
Enter Genre Name

Genre Description

Enter Genre Description

Submit

© Company Name



Add a Publisher To The Database

Publisher's Name

Enter Publisher's Name

Phone

Enter a Phone Number

Email

Enter Email

Address

Enter Address

City

Enter City

State

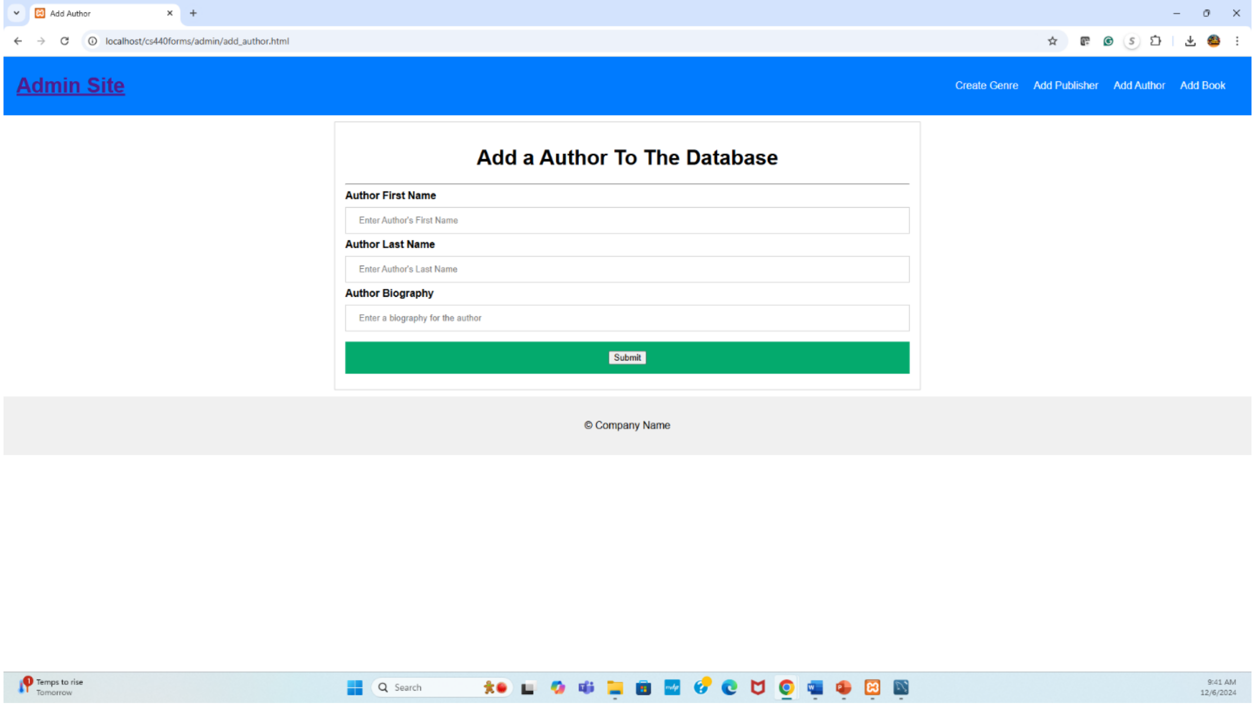
Enter State

Zip

Enter Zip

Submit

© Company Name



Add a Author To The Database

Author First Name

Enter Author's First Name

Author Last Name

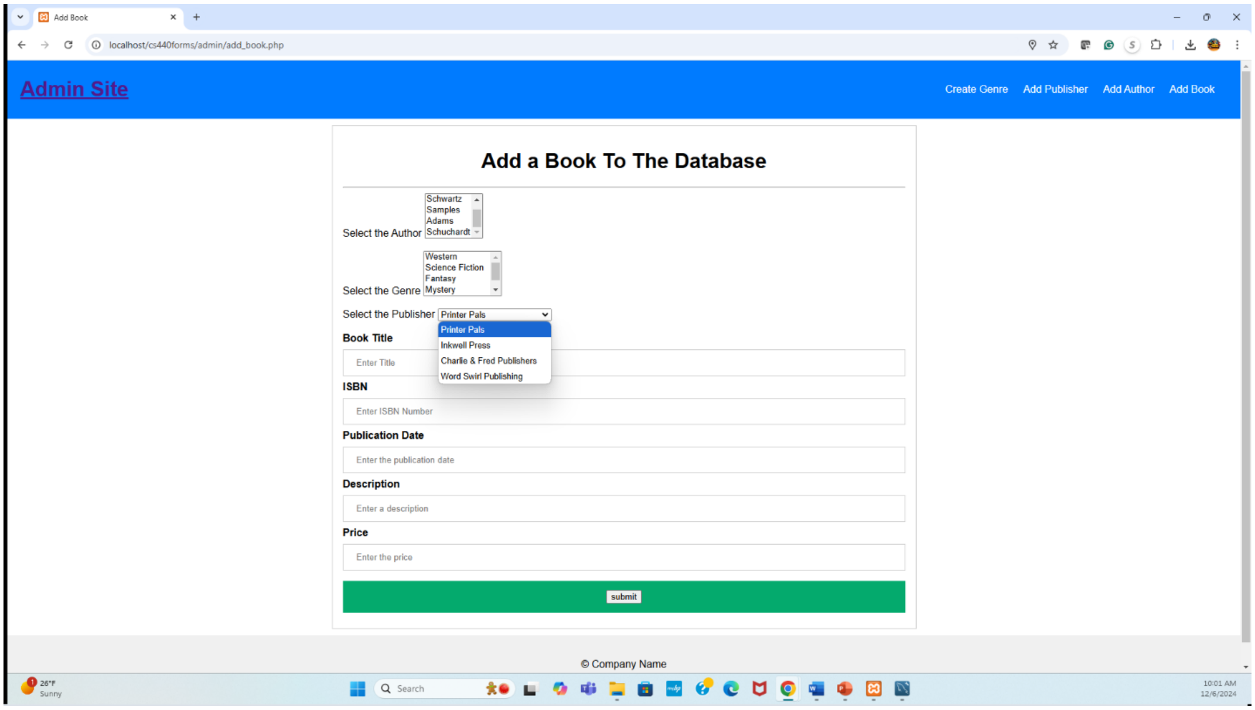
Enter Author's Last Name

Author Biography

Enter a biography for the author

Submit

© Company Name



Add Book

localhost/cs440forms/admin/add_book.php

Admin Site

Create Genre Add Publisher Add Author Add Book

Add a Book To The Database

Select the Author: Schwartz, Samples, Adams, Schuchardt

Select the Genre: Western, Science Fiction, Fantasy, Mystery

Select the Publisher: Printer Pals

- Printer Pals
- Inkwell Press
- Charlie & Fred Publishers
- Word Swirl Publishing

Book Title

Enter Title

ISBN

Enter ISBN Number

Publication Date

Enter the publication date

Description

Enter a description

Price

Enter the price

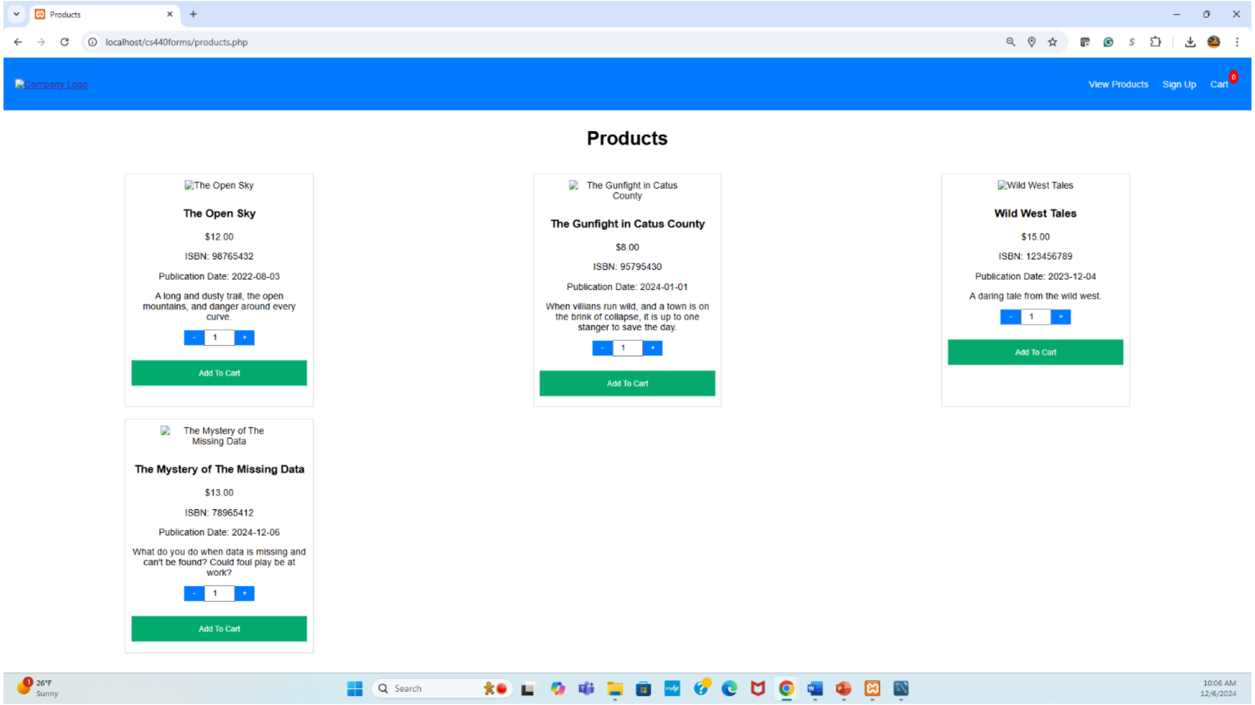
submit

© Company Name

26°F Sunny

Search

10:01 AM 12/6/2024



Products

localhost/cs440forms/products.php

Company Logo

View Products Sign Up Cart 0

Products

The Open Sky

The Open Sky

$12.00

ISBN: 98765432

Publication Date: 2022-08-03

A long and dusty trail, the open mountains, and danger around every curve.

- 1 +

Add To Cart

The Gunfight in Catus County

The Gunfight in Catus County

$8.00

ISBN: 95795430

Publication Date: 2024-01-01

When villians run wild, and a town is on the brink of collapse, it is up to one stanger to save the day.

- 1 +

Add To Cart

Wild West Tales

Wild West Tales

$15.00

ISBN: 123456789

Publication Date: 2023-12-04

A daring tale from the wild west.

- 1 +

Add To Cart

The Mystery of The Missing Data

The Mystery of The Missing Data

$13.00

ISBN: 78965412

Publication Date: 2024-12-06

What do you do when data is missing and can't be found? Could foul play be at work?

- 1 +

Add To Cart

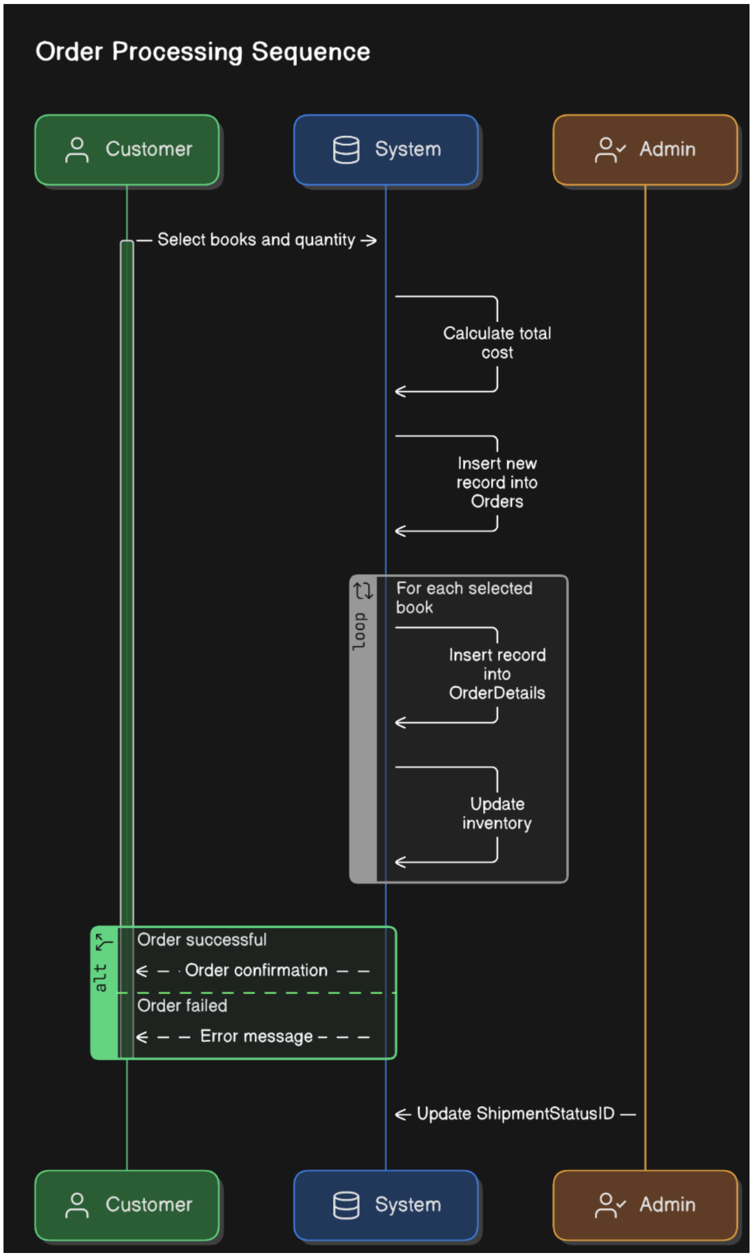
26°F Sunny

Search

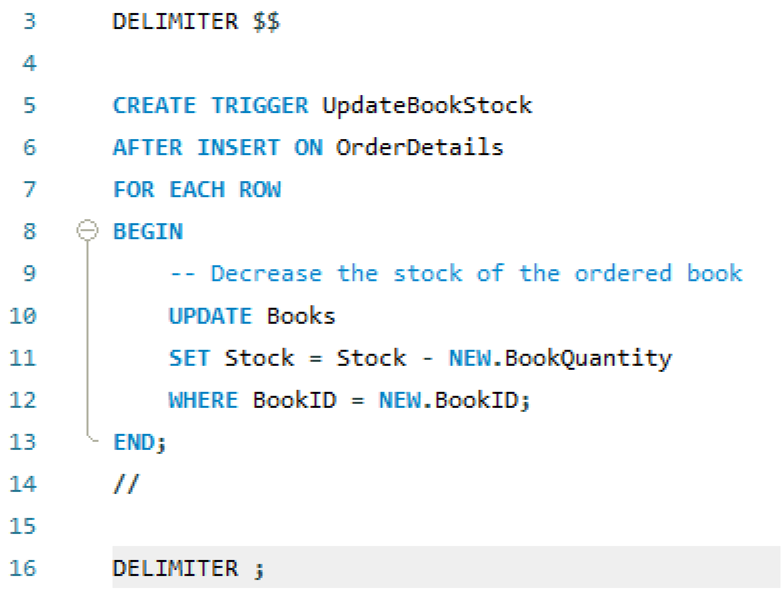
10:06 AM 12/6/2024

9) Sequence and Triggers

Sequence:



This is a trigger to update book stock. When a new order is placed, this trigger subtracts the total stock of the book.

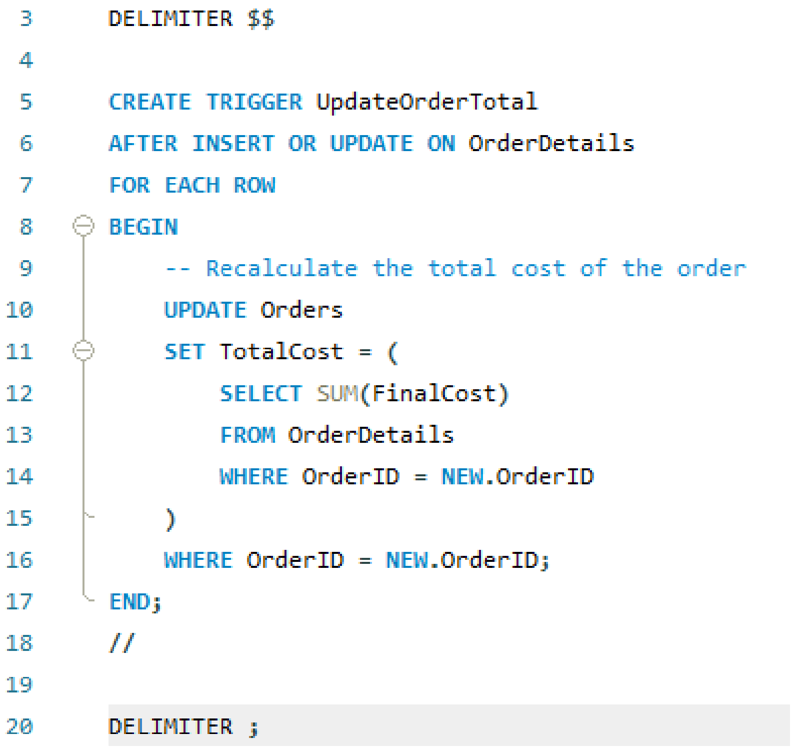


```sql
DELIMITER $$

CREATE TRIGGER UpdateBookStock
AFTER INSERT ON OrderDetails
FOR EACH ROW
BEGIN
    -- Decrease the stock of the ordered book
    UPDATE Books
    SET Stock = Stock - NEW.BookQuantity
    WHERE BookID = NEW.BookID;
END;
//

DELIMITER ;
```

This trigger updates the TotalCost in the Orders table whenever an entry is added or updates.

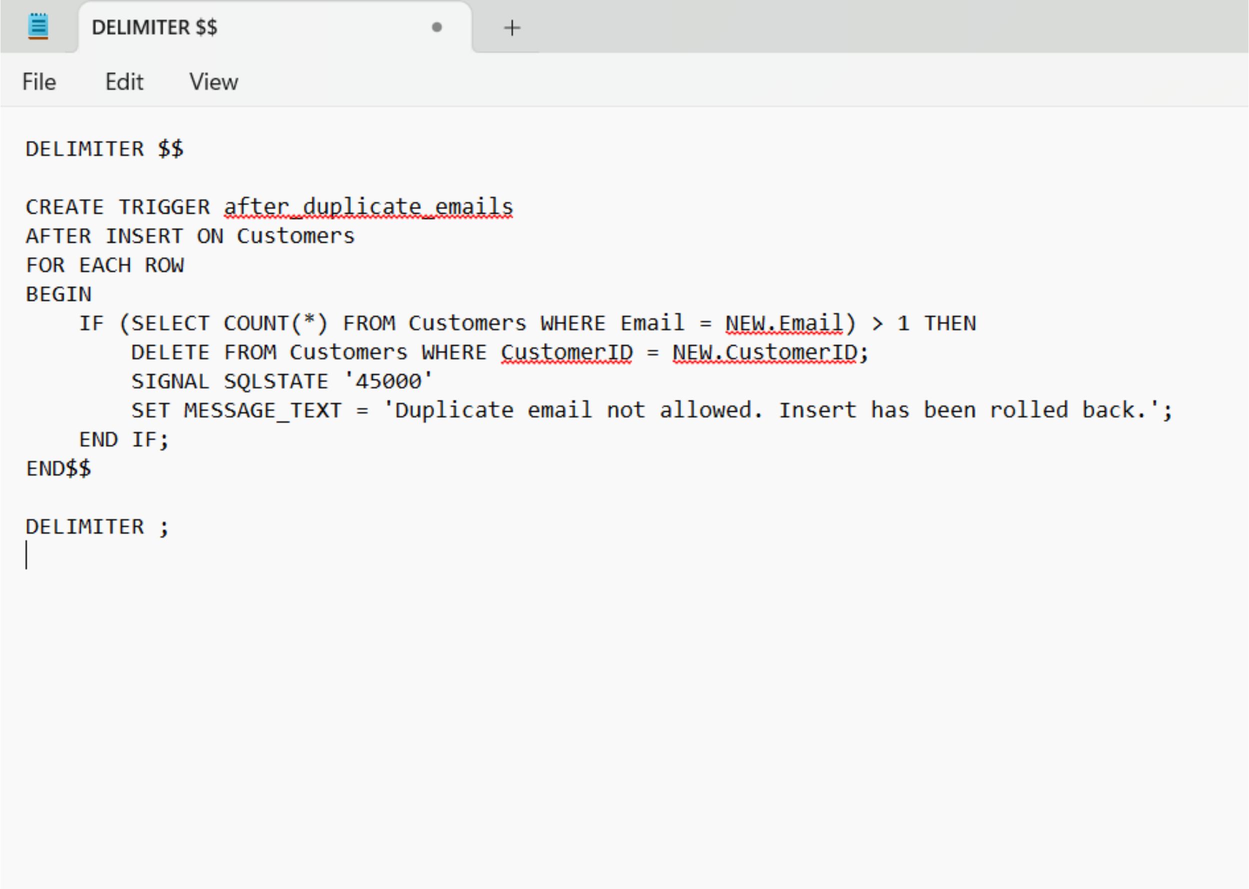


```sql
DELIMITER $$

CREATE TRIGGER UpdateOrderTotal
AFTER INSERT OR UPDATE ON OrderDetails
FOR EACH ROW
BEGIN
    -- Recalculate the total cost of the order
    UPDATE Orders
    SET TotalCost = (
        SELECT SUM(FinalCost)
        FROM OrderDetails
        WHERE OrderID = NEW.OrderID
    )
    WHERE OrderID = NEW.OrderID;
END;
//

DELIMITER ;
```

This trigger prevents duplicate emails in the Customers table

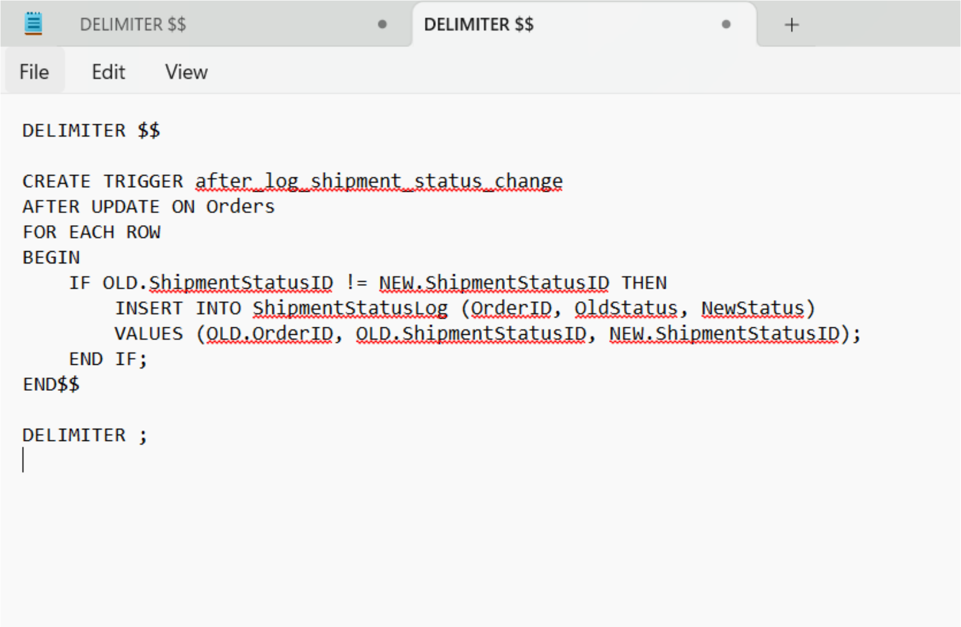


```
DELIMITER $$

CREATE TRIGGER after_duplicate_emails
AFTER INSERT ON Customers
FOR EACH ROW
BEGIN
    IF (SELECT COUNT(*) FROM Customers WHERE Email = NEW.Email) > 1 THEN
        DELETE FROM Customers WHERE CustomerID = NEW.CustomerID;
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Duplicate email not allowed. Insert has been rolled back.';
    END IF;
END$$

DELIMITER ;
```

This trigger logs changes to ShipmentStatusID in the Orders table



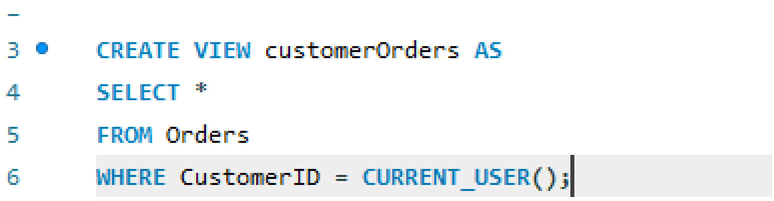
```
DELIMITER $$

CREATE TRIGGER after_log_shipment_status_change
AFTER UPDATE ON Orders
FOR EACH ROW
BEGIN
    IF OLD.ShipmentStatusID != NEW.ShipmentStatusID THEN
        INSERT INTO ShipmentStatusLog (OrderID, OldStatus, NewStatus)
        VALUES (OLD.OrderID, OLD.ShipmentStatusID, NEW.ShipmentStatusID);
    END IF;
END$$

DELIMITER ;
```

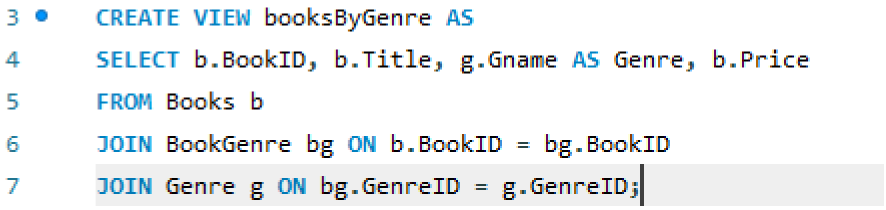
10) Views

This view allows a customer to see all the orders they have placed through the **customerOrders** view.



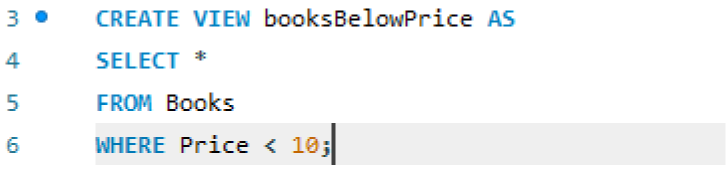
```
CREATE VIEW customerOrders AS
SELECT *
FROM Orders
WHERE CustomerID = CURRENT_USER();
```

The **booksByGenre** view displays the books categorized by genre



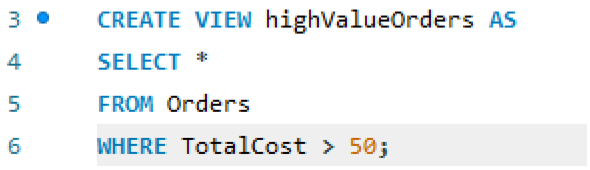
```
CREATE VIEW booksByGenre AS
SELECT b.BookID, b.Title, g.Gname AS Genre, b.Price
FROM Books b
JOIN BookGenre bg ON b.BookID = bg.BookID
JOIN Genre g ON bg.GenreID = g.GenreID;
```

The booksBelowPrice view show all books that cost less than $10.



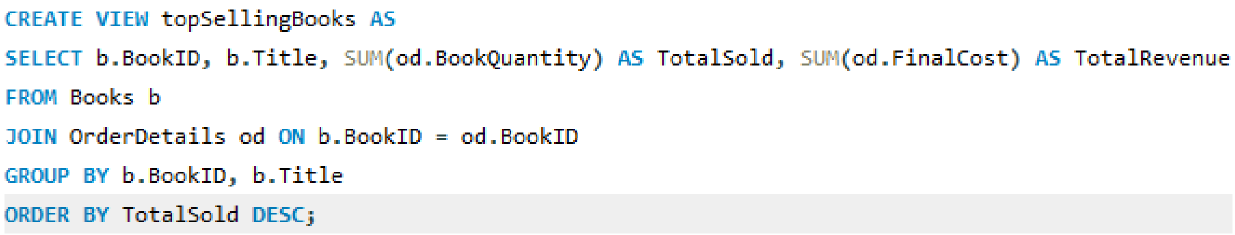
```
CREATE VIEW booksBelowPrice AS
SELECT *
FROM Books
WHERE Price < 10;
```

The highValueOrders view displays all orders larger than $50.



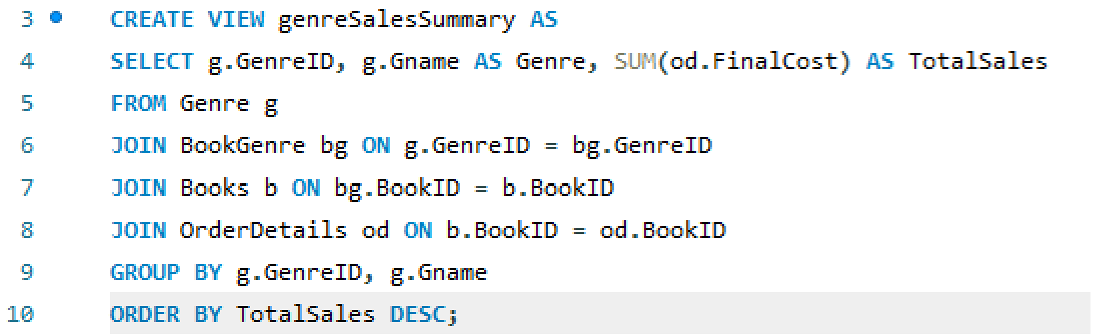
```sql
CREATE VIEW highValueOrders AS
SELECT *
FROM Orders
WHERE TotalCost > 50;
```

The topSellingBooks view calculates the total quantity sold for each book and displays them in order.



```sql
CREATE VIEW topSellingBooks AS
SELECT b.BookID, b.Title, SUM(od.BookQuantity) AS TotalSold, SUM(od.FinalCost) AS TotalRevenue
FROM Books b
JOIN OrderDetails od ON b.BookID = od.BookID
GROUP BY b.BookID, b.Title
ORDER BY TotalSold DESC;
```

The genreSalesSummary view provides a summary of the total sales for each genre of books



```sql
CREATE VIEW genreSalesSummary AS
SELECT g.GenreID, g.Gname AS Genre, SUM(od.FinalCost) AS TotalSales
FROM Genre g
JOIN BookGenre bg ON g.GenreID = bg.GenreID
JOIN Books b ON bg.BookID = b.BookID
JOIN OrderDetails od ON b.BookID = od.BookID
GROUP BY g.GenreID, g.Gname
ORDER BY TotalSales DESC;
```

11) Conclusion and Summary

In conclusion, this project is successful in implementing a relational database system for the bookstore. Our design principles are based off the consideration of the bookstore's functional needs: Order processing, customer management, and inventory tracking. Through several different functions, the database is enhanced to meet those needs. The triggers, views, and schema are some of the main functions of this project:

The relational schema is comprised of multiple entities such as books, publishers, orders, and customers. These tables reduce overall redundancy with the incorporation of foreign and primary keys to create sensible and clear relationships.

- Customers: handles customer details such as address and contact. Is assigned with a CustomerID key
- Authors: handles details on the name of the book author to the "Books" tables and stores to the AuthorID key
- Books: Stores book detail such as price and title with the BookID key.
- Genre: Categorically separates books into sections with GenreID

Triggers work to automate key operations in the database, decreasing error and increasing reliability. Two key triggers are:

- Stock update: Changes the inventory automatically in real time based on when a new order is placed.
- Cost update: Adjusts and recomputes the TotalCost function located in the Orders table whenever a new entry is edited or added.

Views used for this project simplify access to the database and increase efficiency for commonly needed queries. Here are just a few examples of their efficiency:

- CustomerOrders: Shows the customer that made the order and the order details
- TopSellingBooks: Sorts the books by total units sold.
- GenreSalesSummary: Sales summary information of each of the different book genres.

This project highlights the importance of an effective database in a modern business environment. The schema, triggers, and views combine to create a practical system that will meet the needs of the bookstore.

In the forms section, the database is fully connected to the front-end website forms aside from the orders table. In the future, we could complete our project by adding the necessary code in the website to add items to the orders and orderdetails tables based on items that are added to a cart stored in the browser's session storage.

12) Acknowledgements

Joe Samples – Database Design, ER Diagram Drafting, ER Diagram Revision, CRUD Matrix Design, Sequence Diagram Design, Database Triggers, Miscellaneous Work on Paper – Introduction

Kyle Schwartz – ER Diagram Revision, Relational Schema Drafting, Data Types Diagram Drafting, Table Creation With SQL, Table Descriptions, Miscellaneous Work on Paper – Abstract

Thabo Adams - Miscellaneous Work on Paper – Conclusion, Document Formatting, Video editing and formatting

Noah Schuchardt – Database Triggers, Database Views, Forms creation and debugging, ER Diagram Revisions.