“Craftique”

Artisan & Handmade Goods E-commerce Website

Submitted By

Student Name	Student ID
Ramesha Rawnok Haque	0242220005101665
Kh Sadman Sakib	0242220005101951
Most. Sumaiya Islam Sithi	0242220005101915
Fahmida Akter Juthi	0242220005101926
Hafijur Rahman	0242220005101620

MINI LAB PROJECT REPORT

This Report Presented in Partial Fulfillment of the course **CSE312: Database Management System Lab in the Computer Science and Engineering Department**

DAFFODIL INTERNATIONAL UNIVERSITY

Dhaka, Bangladesh

December 10, 2024

DECLARATION

We hereby declare that this lab project has been done by us under the supervision of **Md Shah Jalal**, **Senior Lecturer**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere as lab projects.

Submitted To:

Md. Shah Jalal
Senior Lecturer
Department of Computer Science and Engineering
Daffodil International University.

Submitted by

Ramesha Rawnok Haque 0242220005101665 Dept. of CSE, DIU	
Sadman Sakib Kh Sadman Sakib 0242220005101951 Dept. of CSE, DIU	Sithi Sumaiya Islam Sithi 0242220005101915 Dept. of CSE, DIU
Fahmida Akter Juthi 0242220005101926 Dept. of CSE, DIU	Hafijur Hafijur Rahman 0242220005101620 Dept. of CSE, DIU

COURSE & PROGRAM OUTCOME

The following courses have course outcomes as following.
After completion of this course, students will be able to:

COs	CO Statements	POs	Learning Domains	Knowledge Profile	Complex Engineering Problem	Complex Engineering Activities
CO1	**Demonstrate** a comprehensive understanding of fundamental database management concepts, including the relational data model, normalization techniques, and SQL basics.	**PO1**	**C2** **A2** **P2**	**K2** **K3** **K4** **K8**	**EP1** **EP4**	
CO2	**Design, implement** and optimize relational databases, incorporating advanced SQL queries, indexing techniques and query optimization strategies.	**PO3**	**C3** **A3** **P3**	**K2** **K3** **K4** **K6** **K8**	**EP1** **EP2** **EP7**	**EA3**
CO3	Understand and **Analyze** security measures, distributed database architectures and emerging trends in database management, demonstrating an understanding of the broader context and challenges in the field.	**PO5**	**C4** **A4** **P3**	**K6**	**EP4**	

Details can be found at: [LINK]

Table of Contents

Chapter 1

Introduction

1.1 Introduction

In recent years, the demand for unique, handcrafted goods has grown significantly, as consumers increasingly value creativity, authenticity, and sustainability [1]. To bridge the gap between local artisans and a broader customer base, Craftique is envisioned as an innovative e-commerce platform dedicated to showcasing and selling artisan and handmade products. This platform allows talented creators to reach customers beyond geographical barriers, offering them an opportunity to thrive in the digital marketplace while providing buyers with access to one-of-a-kind items.

1.2 Motivation

The motivation behind this project stems from a desire to empower local artisans and support sustainable practices. Artisans often struggle to compete with large-scale manufacturers, facing challenges in marketing and distribution.[2] With Craftique, we aim to give these creators a digital space where their work can shine. Additionally, by promoting handmade goods, this platform contributes to reducing the environmental impact associated with mass production.

1.3 Objectives

The primary objectives of the *Craftique* project are as follows:

- To design and implement a robust database system that can efficiently manage customers, sellers, products, orders, and payments.
- To ensure data integrity and smooth functionality through triggers, procedures, and views.
- To provide a scalable solution that can be extended into a full-fledged e-commerce website in the future.

1.4 Project Outcome

The expected outcomes of the *Craftique* project are as follows:

- A robust and well-organized relational database system for managing all aspects of the e-commerce platform, including users, products, orders, and payments.
- Stored procedures for automating repetitive tasks such as adding, updating, and deleting records across different tables.
- Triggers to ensure data integrity and enforce business rules, such as maintaining stock levels and updating order statuses.
- Views to simplify data retrieval and provide actionable insights for managing customers, sales, and artisan performance.
- A scalable and flexible database design that can be integrated with a front-end interface in the future for a complete e-commerce solution.
- A foundation for empowering artisans and promoting their products to a wider audience through digital means.

Chapter 2

Proposed Methodology/Architecture

2.1 Requirement Analysis & Design Specification

2.1.1 Overview

The development of *Craftique* required careful planning and analysis to ensure the database supports the functional and non-functional requirements of an e-commerce platform for artisan and handmade goods. Key functionalities identified include user management (customers, sellers, delivery agents), product management (inventory, categories, discounts), and transactional features (orders, payments, reviews). Non-functional requirements included scalability, data consistency, and ease of data retrieval for reporting and analytics.

2.1.2 Proposed Methodology

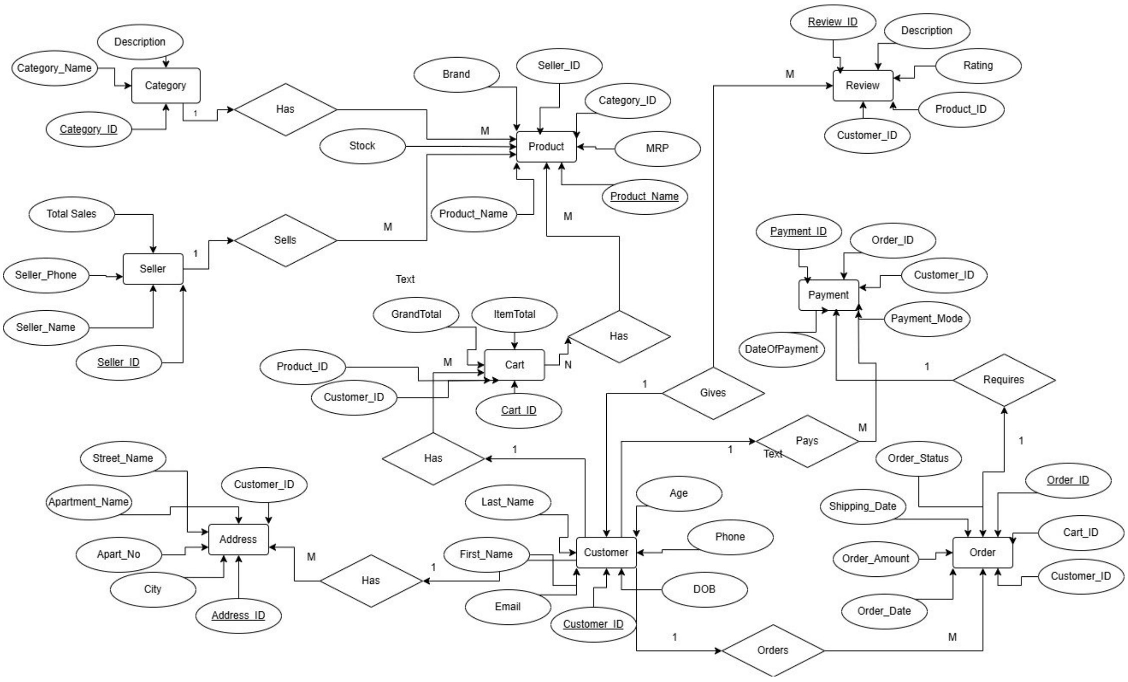
The project followed these steps to design and implement the database:

- **Requirement Analysis:**
  - Identified the key entities (e.g., customers, sellers, products, orders, etc.) and their relationships.
  - Determine the necessary operations for managing data, including adding, updating, and retrieving information.

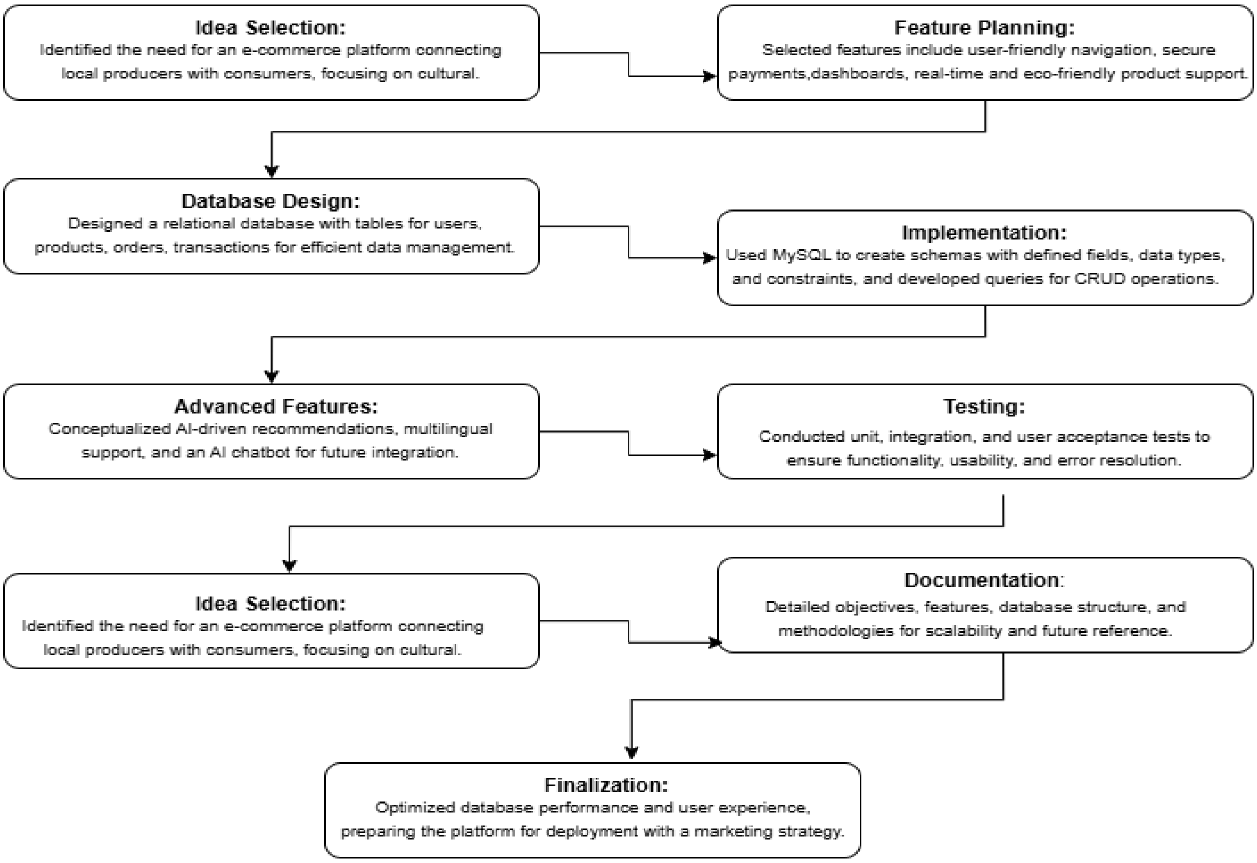
- **Database Design:**
  - Created an Entity-Relationship (ER) diagram to model the relationships between entities.
  - Converted the ER diagram into a relational schema.
  - Applied normalization to eliminate redundancy and ensure data integrity.

Here is the ER diagram we made:

ER Diagram of *Craftique* Database Management



Architecture Diagram / Flow Chart



Idea Selection:
Identified the need for an e-commerce platform connecting local producers with consumers, focusing on cultural.

Feature Planning:
Selected features include user-friendly navigation, secure payments,dashboards, real-time and eco-friendly product support.

Database Design:
Designed a relational database with tables for users, products, orders, transactions for efficient data management.

Implementation:
Used MySQL to create schemas with defined fields, data types, and constraints, and developed queries for CRUD operations.

Advanced Features:
Conceptualized AI-driven recommendations, multilingual support, and an AI chatbot for future integration.

Testing:
Conducted unit, integration, and user acceptance tests to ensure functionality, usability, and error resolution.

Idea Selection:
Identified the need for an e-commerce platform connecting local producers with consumers, focusing on cultural.

Documentation:
Detailed objectives, features, database structure, and methodologies for scalability and future reference.

Finalization:
Optimized database performance and user experience, preparing the platform for deployment with a marketing strategy.

Environment Setup as Availability of Code:

Github Repository Link of Code: To find the repository, click on the links below:

- **Kh Sadman Sakib (ID-0242220005101951):**

  https://github.com/sakibsidha/DBMS_Project_Craftique/tree/main

- **Ramesha Rawnok Haque (ID-0242220005101665):**

  https://github.com/rawnok-18/DBMS_Craftique_Project/tree/main

Tools Used: MySQL, phpMyAdmin, XAMPP.

1. MySQL Documentation. (n.d.). MySQL Reference Manual. Retrieved from https://dev.mysql.com/doc/
   - Official documentation for MySQL, providing insights into stored procedures, triggers, and views.
2. phpMyAdmin Documentation. (n.d.). phpMyAdmin Official Documentation. Retrieved from https://docs.phpmyadmin.net/
   - Guidance on using phpMyAdmin for database management.
3. XAMPP Documentation. (n.d.). XAMPP for Beginners. Retrieved from https://www.apachefriends.org/docs/
   - Instructions for setting up a local server environment

Database Implementation:

- Designed and implemented tables in MySQL using XAMPP phpMyAdmin.
- Added constraints like primary keys, foreign keys, and unique constraints for data accuracy.

- **Automation and Data Integrity:**
  - Developed stored procedures for repetitive tasks such as adding new records and generating reports.
  - Created triggers to maintain data integrity, such as updating inventory after an order or validating data changes.
  - Designed views to simplify data retrieval for reporting purposes.

- **Testing criteria:**
  - Tested all database operations, including queries, triggers, and procedures, to ensure correctness and efficiency.
  - Validated the system by simulating real-world scenarios such as order placement, payment processing, and inventory updates.

2.2 Overall Project Plan

The development of *Craftique* followed a structured and iterative approach:

1. **Phase 1:** Requirement Gathering and Analysis
    - Defined the scope of the project and identified key database features.
    - Drafted an ER diagram to visualize entity relationships.
2. **Phase 2:** Database Design and Implementation
    - Designed tables and implemented them in MySQL.
    - Wrote stored procedures for CRUD (Create, Read, Update, Delete) operations.
    - Created triggers for automated tasks like maintaining inventory and order status updates.
3. **Phase 3:** Testing and Debugging
    - Performed testing on queries, procedures, and triggers to ensure correctness.
    - Validated data consistency and accuracy in scenarios such as adding users, placing orders, and handling payments.
4. **Phase 4:** Documentation and Report Preparation
    - Documented the database schema, triggers, procedures, and views.
    - Prepared screenshots from the XAMPP server for the implementation section.

Chapter 3

Implementation and Results

.

3.1 Implementation

The implementation of the Craftique project focused on designing and deploying a database system that manages various aspects of an e-commerce platform for artisan and handmade goods. The following steps were undertaken during the implementation:

3.1.1 Database Structure

The database consists of multiple tables to manage key entities such as customers, products, sellers, orders, payments, and more. Each table is designed to ensure data consistency, avoid redundancy, and handle complex operations efficiently.

Tables:

address:

Address_ID (PK),
Customer_ID (FK),
Apartment_Name,
Apart_No,
Street_Name,
City;

cart:

Cart_ID (PK),
Customer_ID (FK),
Product_ID (FK),
Quantity,
ItemTotal,
GrandTotal;

category:

Category_ID (PK),
Category_Name,
Description;

coupon:

Coupon_ID (PK),
Coupon_Code,
Discount_Value,
Expiry_Date;

customer:

Customer_ID (PK),
First_Name,
Last_Name,
Email,

Phone,
Age,
DOB;

delivery_Agent:
Agent_ID (PK),
Agent_Name,
Phone,
Vehicle_Type;

discount:
Discount_ID (PK),
Product_ID (PK),
Discount_Percentage,
Start_Date,
End_Date;

feedback:
Feedback_ID (PK),
Customer_ID (PK),
Feedback_Text,
Feedback_Date;

inventory:
Inventory_ID (PK),
Product_ID (FK),
Location,
Quantity_In_Stock;

notification:
Notification_ID (PK),
User_ID (FK),
User_Type,
Notification_Text,
Notification_Date,
Is_Read;

order:
Order_ID (PK),
Cart_ID (FK),
Customer_ID (FK),
Order_Amount,
Order_Date,
Shipping_Date,
Order_Status;

payment:
Payment_ID (PK),
Order_ID (FK),
Customer_ID (FK),
Payment_Mode,
DateOfPayment;

product:

- Product_ID (PK),
- Product_Name,
- Category_ID (FK),
- Seller_ID (FK),
- Brand,
- Stock,
- MRP;

review:

- Review_ID (PK),
- Customer_ID (FK),
- Product_ID (FK),
- Rating,
- Description;

seller:

- Seller_ID (PK),
- Seller_Name,
- Seller_Phone,
- Total_Sales;

shipment:

- Shipment_ID (PK),
- Order_ID (FK),
- Agent_ID (FK),
- Shipment_Date,
- Expected_Delivery_Date,
- Shipment_Status;

wishlist:

- Wishlist_ID (PK),
- Customer_ID (FK),
- Product_ID (FK),
- Date_Added;

Here are some screenshots:

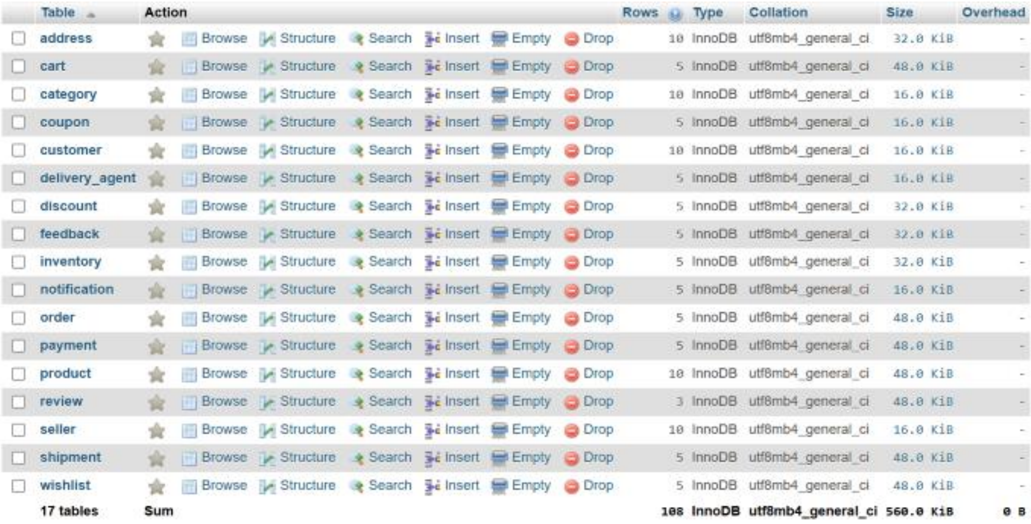
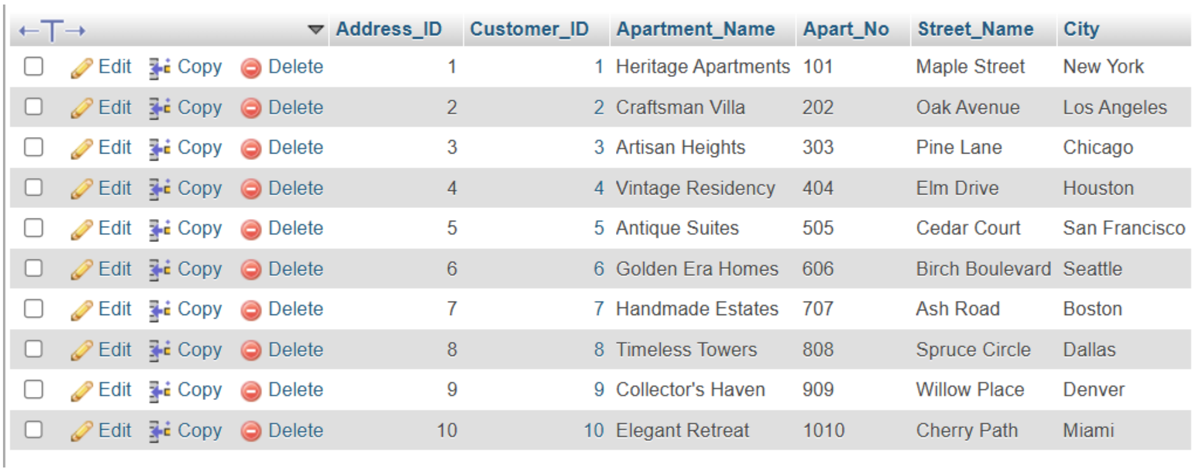


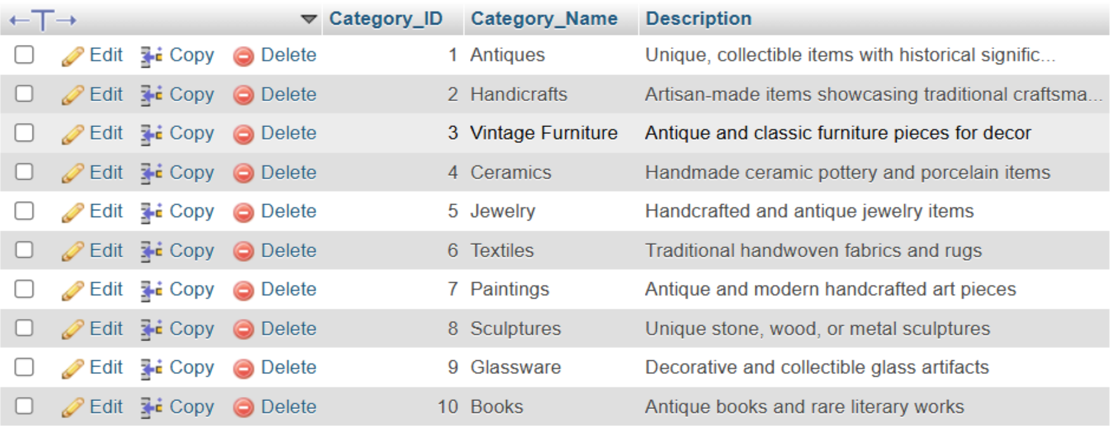
	Table	Action	Rows	Type	Collation	Size	Overhead
□	address	☆ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_general_ci	32.0 KiB	-
□	cart	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	48.0 KiB	-
□	category	☆ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	coupon	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	customer	☆ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	delivery_agent	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	discount	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	32.0 KiB	-
□	feedback	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	32.0 KiB	-
□	inventory	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	32.0 KiB	-
□	notification	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	order	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	48.0 KiB	-
□	payment	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	48.0 KiB	-
□	product	☆ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_general_ci	48.0 KiB	-
□	review	☆ Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	48.0 KiB	-
□	seller	☆ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	shipment	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	48.0 KiB	-
□	wishlist	☆ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	48.0 KiB	-
	17 tables	Sum	108	InnoDB	utf8mb4_general_ci	560.0 KiB	0 B

Fig: Tables Of Database



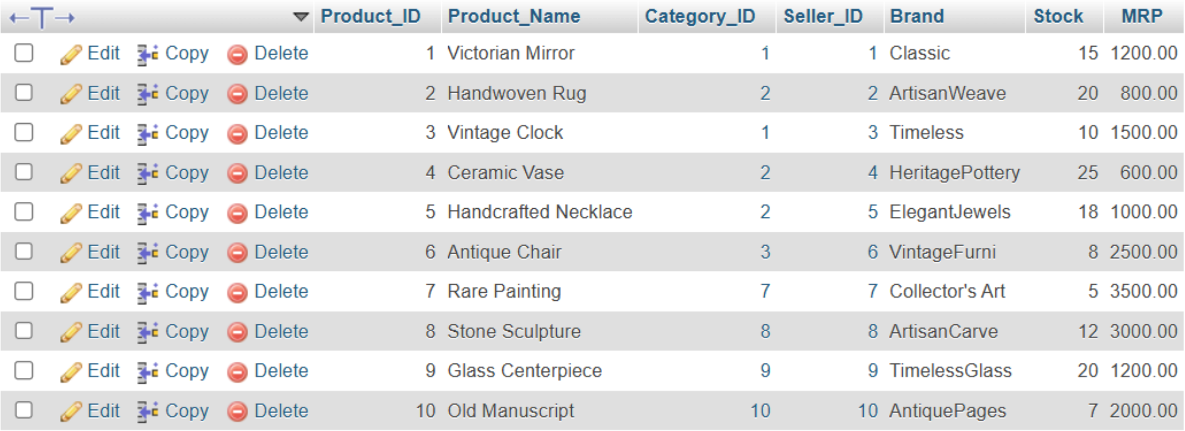
←T→				Address_ID	Customer_ID	Apartment_Name	Apart_No	Street_Name	City
□	Edit	Copy	Delete	1	1	Heritage Apartments	101	Maple Street	New York
□	Edit	Copy	Delete	2	2	Craftsman Villa	202	Oak Avenue	Los Angeles
□	Edit	Copy	Delete	3	3	Artisan Heights	303	Pine Lane	Chicago
□	Edit	Copy	Delete	4	4	Vintage Residency	404	Elm Drive	Houston
□	Edit	Copy	Delete	5	5	Antique Suites	505	Cedar Court	San Francisco
□	Edit	Copy	Delete	6	6	Golden Era Homes	606	Birch Boulevard	Seattle
□	Edit	Copy	Delete	7	7	Handmade Estates	707	Ash Road	Boston
□	Edit	Copy	Delete	8	8	Timeless Towers	808	Spruce Circle	Dallas
□	Edit	Copy	Delete	9	9	Collector's Haven	909	Willow Place	Denver
□	Edit	Copy	Delete	10	10	Elegant Retreat	1010	Cherry Path	Miami

Fig: Address Table



←T→				Category_ID	Category_Name	Description
□	Edit	Copy	Delete	1	Antiques	Unique, collectible items with historical signific...
□	Edit	Copy	Delete	2	Handicrafts	Artisan-made items showcasing traditional craftsma...
□	Edit	Copy	Delete	3	Vintage Furniture	Antique and classic furniture pieces for decor
□	Edit	Copy	Delete	4	Ceramics	Handmade ceramic pottery and porcelain items
□	Edit	Copy	Delete	5	Jewelry	Handcrafted and antique jewelry items
□	Edit	Copy	Delete	6	Textiles	Traditional handwoven fabrics and rugs
□	Edit	Copy	Delete	7	Paintings	Antique and modern handcrafted art pieces
□	Edit	Copy	Delete	8	Sculptures	Unique stone, wood, or metal sculptures
□	Edit	Copy	Delete	9	Glassware	Decorative and collectible glass artifacts
□	Edit	Copy	Delete	10	Books	Antique books and rare literary works

Fig: Category Table



←T→				Product_ID	Product_Name	Category_ID	Seller_ID	Brand	Stock	MRP
□	Edit	Copy	Delete	1	Victorian Mirror	1	1	Classic	15	1200.00
□	Edit	Copy	Delete	2	Handwoven Rug	2	2	ArtisanWeave	20	800.00
□	Edit	Copy	Delete	3	Vintage Clock	1	3	Timeless	10	1500.00
□	Edit	Copy	Delete	4	Ceramic Vase	2	4	HeritagePottery	25	600.00
□	Edit	Copy	Delete	5	Handcrafted Necklace	2	5	ElegantJewels	18	1000.00
□	Edit	Copy	Delete	6	Antique Chair	3	6	VintageFurni	8	2500.00
□	Edit	Copy	Delete	7	Rare Painting	7	7	Collector's Art	5	3500.00
□	Edit	Copy	Delete	8	Stone Sculpture	8	8	ArtisanCarve	12	3000.00
□	Edit	Copy	Delete	9	Glass Centerpiece	9	9	TimelessGlass	20	1200.00
□	Edit	Copy	Delete	10	Old Manuscript	10	10	AntiquePages	7	2000.00

Fig: Product Table

3.1.2 Stored Procedures

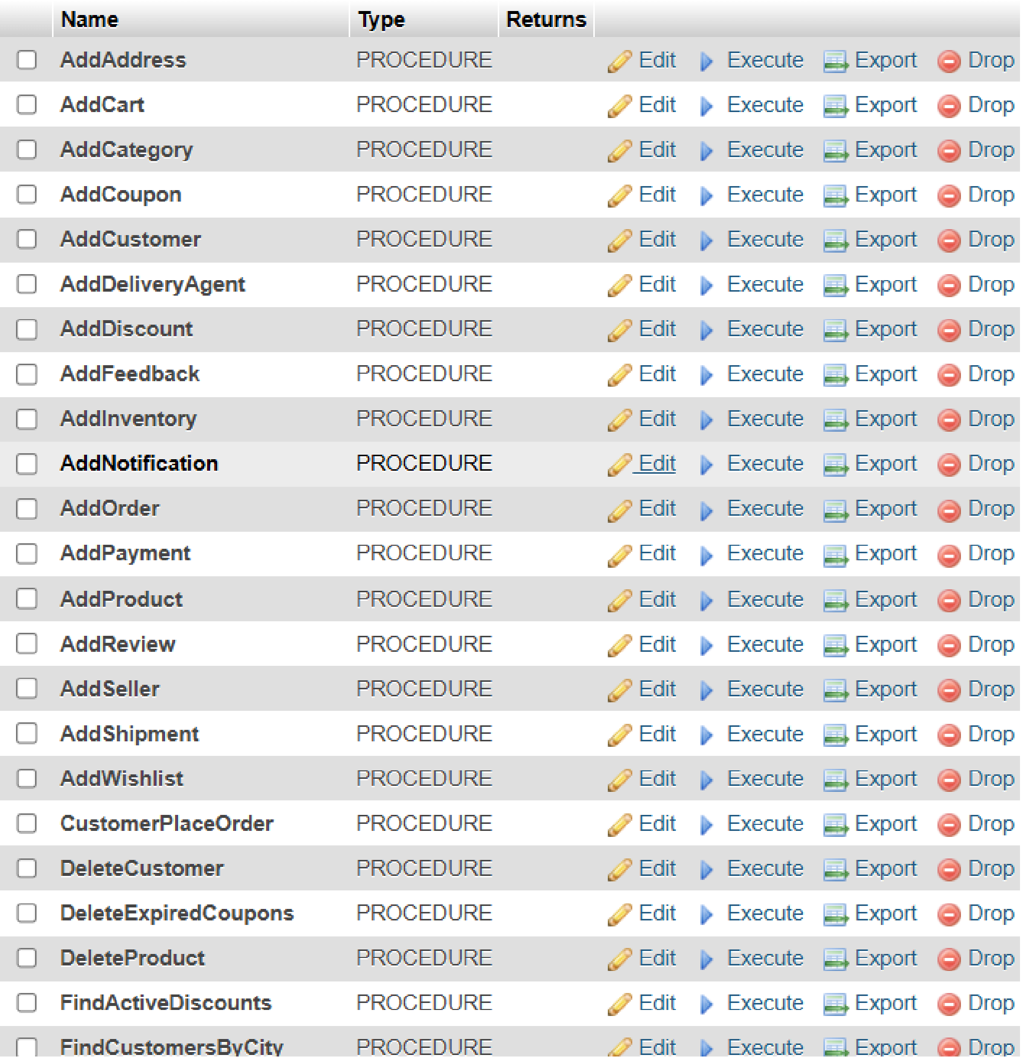
Stored procedures were created to automate repetitive tasks and simplify database operations. Some of the key procedures include:

- Adding new customers, sellers, and products.
- Updating stock after a purchase.
- Generating sales and performance reports.

For example:
The AddCustomer procedure allows easy insertion of customer data, ensuring no manual errors and maintaining uniformity across records.

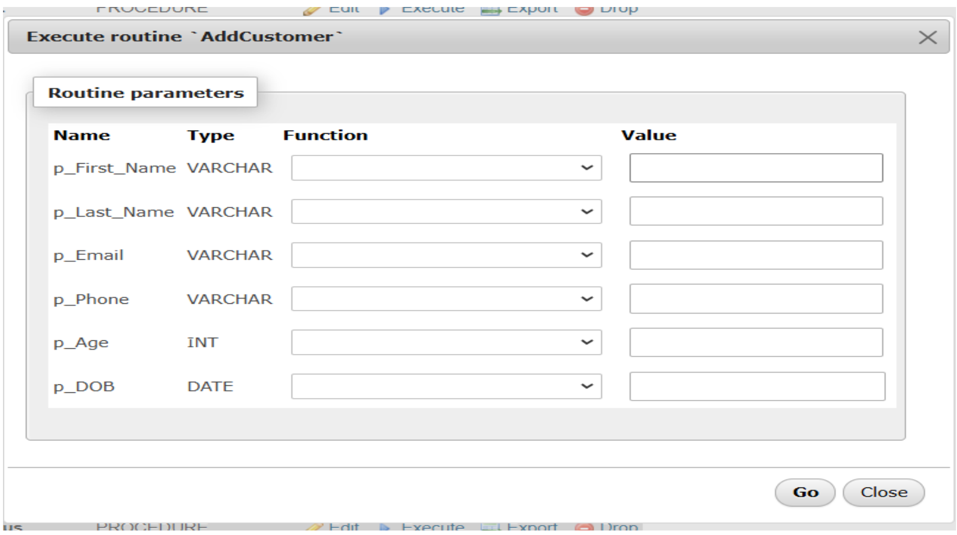
```
DELIMITER //
      CREATE PROCEDURE AddCustomer(
          IN p_FirstName VARCHAR(50),
          IN p_LastName VARCHAR(50),
          IN p_Email VARCHAR(100),
          IN p_Phone VARCHAR(15),
          IN p_Age INT,
          IN p_DOB DATE
      )
      BEGIN
          INSERT INTO customer (First_Name, Last_Name, Email, Phone,
     Age, DOB)
          VALUES (p_FirstName, p_LastName, p_Email, p_Phone, p_Age,
     p_DOB);
      END //
      DELIMITER ;
```

Here are some screenshots of the procedures we implemented.



	Name	Type	Returns	
□	AddAddress	PROCEDURE		Edit Execute Export Drop
□	AddCart	PROCEDURE		Edit Execute Export Drop
□	AddCategory	PROCEDURE		Edit Execute Export Drop
□	AddCoupon	PROCEDURE		Edit Execute Export Drop
□	AddCustomer	PROCEDURE		Edit Execute Export Drop
□	AddDeliveryAgent	PROCEDURE		Edit Execute Export Drop
□	AddDiscount	PROCEDURE		Edit Execute Export Drop
□	AddFeedback	PROCEDURE		Edit Execute Export Drop
□	AddInventory	PROCEDURE		Edit Execute Export Drop
□	AddNotification	PROCEDURE		Edit Execute Export Drop
□	AddOrder	PROCEDURE		Edit Execute Export Drop
□	AddPayment	PROCEDURE		Edit Execute Export Drop
□	AddProduct	PROCEDURE		Edit Execute Export Drop
□	AddReview	PROCEDURE		Edit Execute Export Drop
□	AddSeller	PROCEDURE		Edit Execute Export Drop
□	AddShipment	PROCEDURE		Edit Execute Export Drop
□	AddWishlist	PROCEDURE		Edit Execute Export Drop
□	CustomerPlaceOrder	PROCEDURE		Edit Execute Export Drop
□	DeleteCustomer	PROCEDURE		Edit Execute Export Drop
□	DeleteExpiredCoupons	PROCEDURE		Edit Execute Export Drop
□	DeleteProduct	PROCEDURE		Edit Execute Export Drop
□	FindActiveDiscounts	PROCEDURE		Edit Execute Export Drop
□	FindCustomersByCity	PROCEDURE		Edit Execute Export Drop

FIG: Procedure of Database

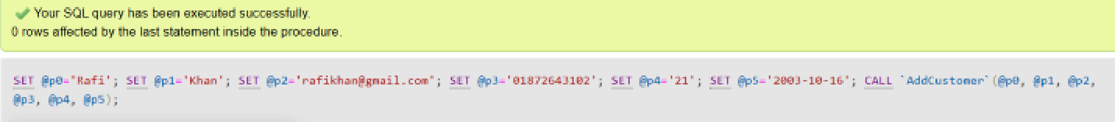


Execute routine `AddCustomer`

Routine parameters

Name	Type	Function	Value
p_First_Name	VARCHAR		
p_Last_Name	VARCHAR		
p_Email	VARCHAR		
p_Phone	VARCHAR		
p_Age	INT		
p_DOB	DATE		

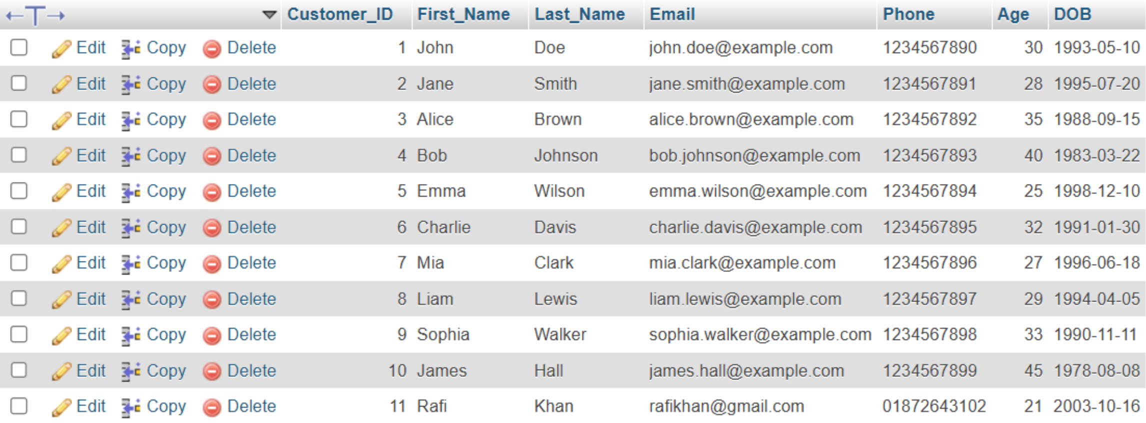
Fig: Execute routine



✓ Your SQL query has been executed successfully.
0 rows affected by the last statement inside the procedure.

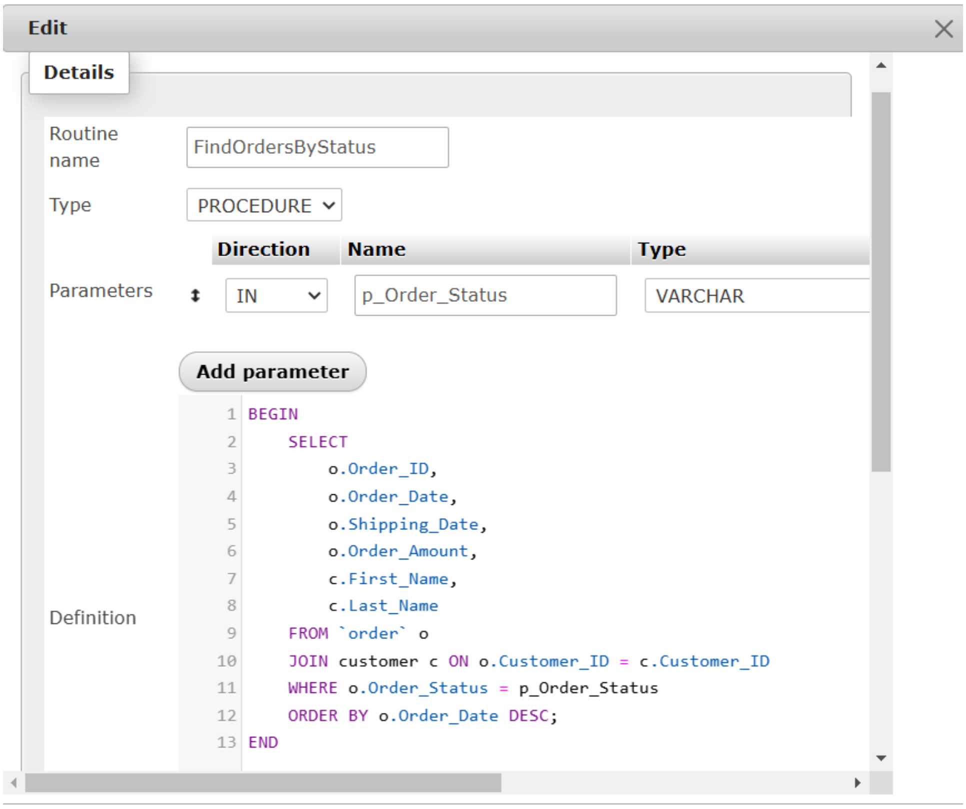
```
SET @p0='Rafi'; SET @p1='Khan'; SET @p2='rafikhan@gmail.com'; SET @p3='01872643102'; SET @p4='21'; SET @p5='2003-10-16'; CALL `AddCustomer`(@p0, @p1, @p2,
@p3, @p4, @p5);
```

Fig: Executed Statement



←T→			▼	Customer_ID	First_Name	Last_Name	Email	Phone	Age	DOB
□	Edit	Copy	Delete	1	John	Doe	john.doe@example.com	1234567890	30	1993-05-10
□	Edit	Copy	Delete	2	Jane	Smith	jane.smith@example.com	1234567891	28	1995-07-20
□	Edit	Copy	Delete	3	Alice	Brown	alice.brown@example.com	1234567892	35	1988-09-15
□	Edit	Copy	Delete	4	Bob	Johnson	bob.johnson@example.com	1234567893	40	1983-03-22
□	Edit	Copy	Delete	5	Emma	Wilson	emma.wilson@example.com	1234567894	25	1998-12-10
□	Edit	Copy	Delete	6	Charlie	Davis	charlie.davis@example.com	1234567895	32	1991-01-30
□	Edit	Copy	Delete	7	Mia	Clark	mia.clark@example.com	1234567896	27	1996-06-18
□	Edit	Copy	Delete	8	Liam	Lewis	liam.lewis@example.com	1234567897	29	1994-04-05
□	Edit	Copy	Delete	9	Sophia	Walker	sophia.walker@example.com	1234567898	33	1990-11-11
□	Edit	Copy	Delete	10	James	Hall	james.hall@example.com	1234567899	45	1978-08-08
□	Edit	Copy	Delete	11	Rafi	Khan	rafikhan@gmail.com	01872643102	21	2003-10-16

Fig: Customer Table



Routine name: FindOrdersByStatus

Type: PROCEDURE

Parameters:

Direction	Name	Type
IN	p_Order_Status	VARCHAR

Definition:

```
BEGIN
    SELECT
        o.Order_ID,
        o.Order_Date,
        o.Shipping_Date,
        o.Order_Amount,
        c.First_Name,
        c.Last_Name
    FROM `order` o
    JOIN customer c ON o.Customer_ID = c.Customer_ID
    WHERE o.Order_Status = p_Order_Status
    ORDER BY o.Order_Date DESC;
END
```

Fig: Procedure Parameters

3.1.3 Triggers

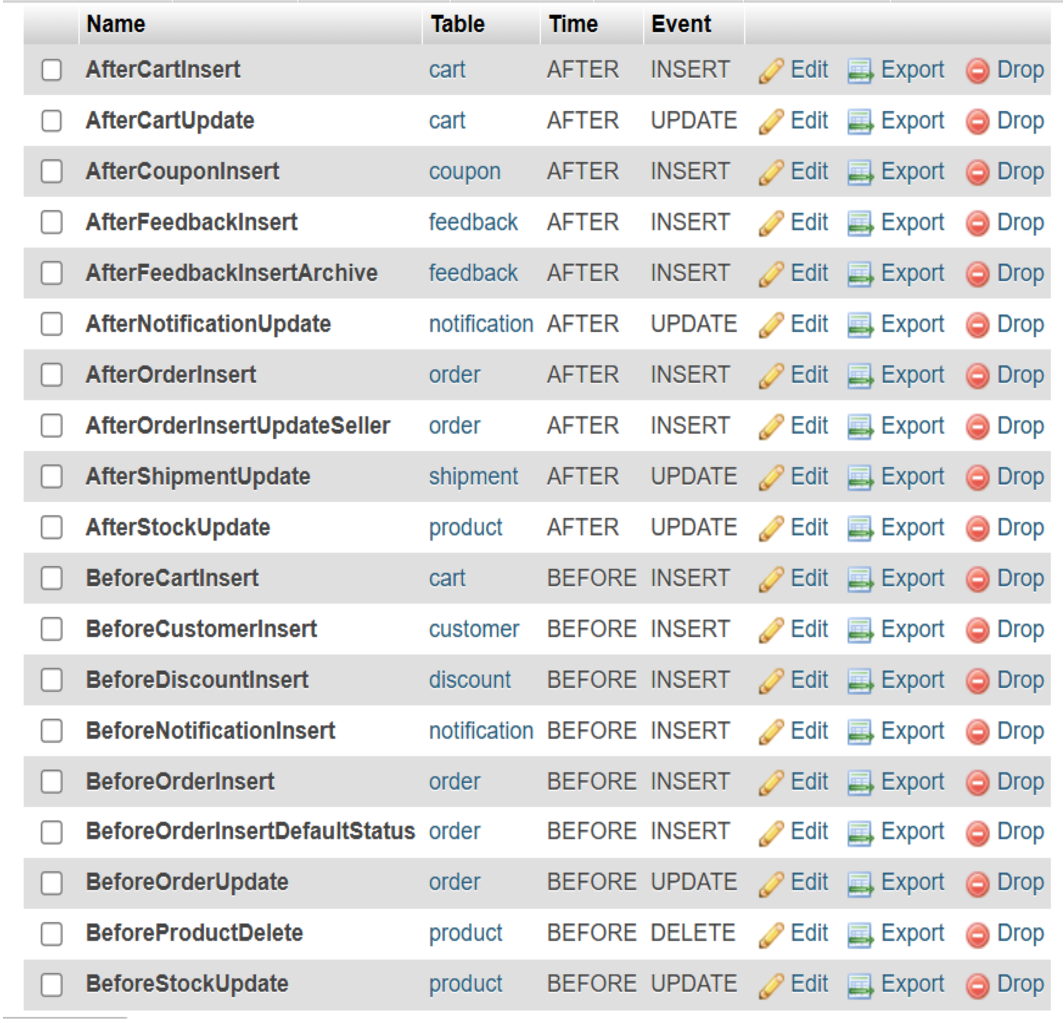
Triggers were implemented to ensure data consistency and enforce business rules. Some examples include:

Updating stock in the inventory table after a purchase is made.

Automatically calculating cart totals when items are added or updated.
Preventing deletion of a product if it is linked to an active order.
For example:
The AfterOrderInsert trigger updates inventory levels when a new order is placed.

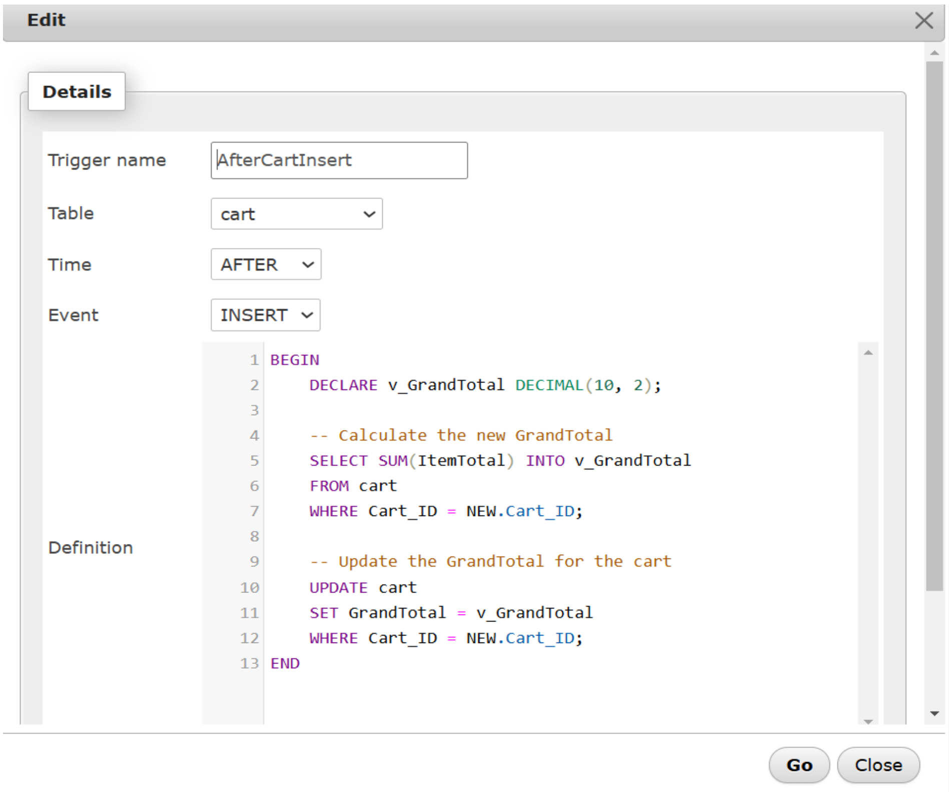
```
        DELIMITER //
        CREATE TRIGGER AfterOrderInsert
        AFTER INSERT ON order
        FOR EACH ROW
        BEGIN
            UPDATE product
            SET Stock = Stock - (SELECT Quantity FROM cart WHERE
       Product_ID = NEW.Product_ID)
            WHERE Product_ID = NEW.Product_ID;
        END //
DELIMITER ;
```

Here are some screenshots:



	Name	Table	Time	Event			
□	AfterCartInsert	cart	AFTER	INSERT	Edit	Export	Drop
□	AfterCartUpdate	cart	AFTER	UPDATE	Edit	Export	Drop
□	AfterCouponInsert	coupon	AFTER	INSERT	Edit	Export	Drop
□	AfterFeedbackInsert	feedback	AFTER	INSERT	Edit	Export	Drop
□	AfterFeedbackInsertArchive	feedback	AFTER	INSERT	Edit	Export	Drop
□	AfterNotificationUpdate	notification	AFTER	UPDATE	Edit	Export	Drop
□	AfterOrderInsert	order	AFTER	INSERT	Edit	Export	Drop
□	AfterOrderInsertUpdateSeller	order	AFTER	INSERT	Edit	Export	Drop
□	AfterShipmentUpdate	shipment	AFTER	UPDATE	Edit	Export	Drop
□	AfterStockUpdate	product	AFTER	UPDATE	Edit	Export	Drop
□	BeforeCartInsert	cart	BEFORE	INSERT	Edit	Export	Drop
□	BeforeCustomerInsert	customer	BEFORE	INSERT	Edit	Export	Drop
□	BeforeDiscountInsert	discount	BEFORE	INSERT	Edit	Export	Drop
□	BeforeNotificationInsert	notification	BEFORE	INSERT	Edit	Export	Drop
□	BeforeOrderInsert	order	BEFORE	INSERT	Edit	Export	Drop
□	BeforeOrderInsertDefaultStatus	order	BEFORE	INSERT	Edit	Export	Drop
□	BeforeOrderUpdate	order	BEFORE	UPDATE	Edit	Export	Drop
□	BeforeProductDelete	product	BEFORE	DELETE	Edit	Export	Drop
□	BeforeStockUpdate	product	BEFORE	UPDATE	Edit	Export	Drop

Fig: Trigger Table



Trigger name	AfterCartInsert
Table	cart
Time	AFTER
Event	INSERT

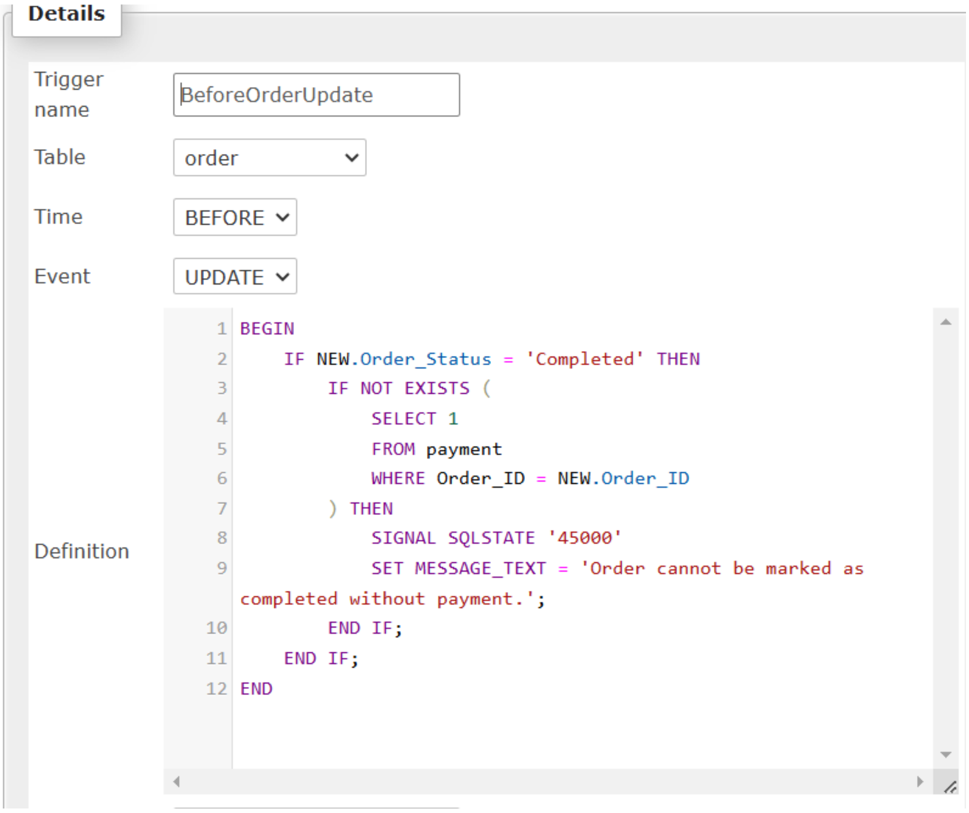
Definition

```sql
BEGIN
    DECLARE v_GrandTotal DECIMAL(10, 2);

    -- Calculate the new GrandTotal
    SELECT SUM(ItemTotal) INTO v_GrandTotal
    FROM cart
    WHERE Cart_ID = NEW.Cart_ID;

    -- Update the GrandTotal for the cart
    UPDATE cart
    SET GrandTotal = v_GrandTotal
    WHERE Cart_ID = NEW.Cart_ID;
END
```

Fig: Query Input



Trigger name	BeforeOrderUpdate
Table	order
Time	BEFORE
Event	UPDATE

Definition

```sql
BEGIN
    IF NEW.Order_Status = 'Completed' THEN
        IF NOT EXISTS (
            SELECT 1
            FROM payment
            WHERE Order_ID = NEW.Order_ID
        ) THEN
            SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Order cannot be marked as completed without payment.';
        END IF;
    END IF;
END
```

Fig: Trigger Code (SQL Input)

3.1.4 Views

Views were created to simplify data retrieval and allow efficient reporting. Key views include:

- A view to track product performance (sales and reviews).
- A view showing customer order histories.
- A view to list top-selling products for artisan performance analysis.

For example: The TopSellingProducts view displays products with the highest sales:

```
 CREATE VIEW TopSellingProducts AS
 SELECT p.Product_ID, p.Product_Name, SUM(o.Order_Amount) AS
Total_Sales
 FROM product p
 JOIN order o ON p.Product_ID = o.Product_ID
 GROUP BY p.Product_ID, p.Product_Name
 ORDER BY Total_Sales DESC;
```

Here are some screenshots of the views we implemented:

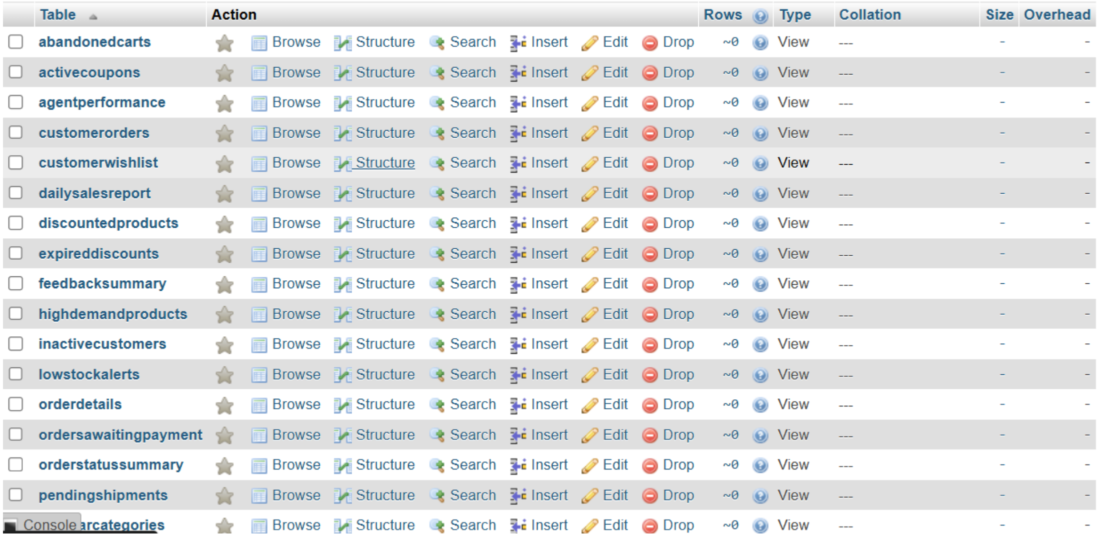
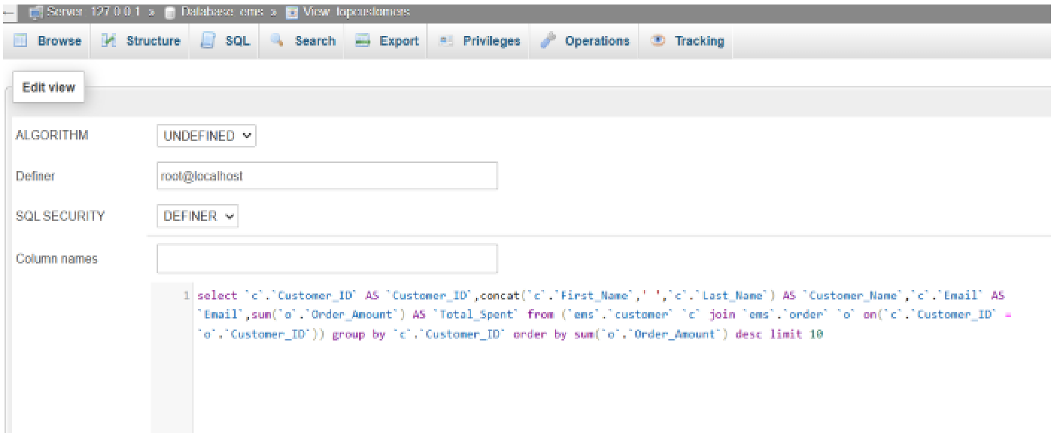


	Table	Action							Rows	Type	Collation	Size	Overhead
□	abandonedcarts	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	activecoupons	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	agentperformance	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	customerorders	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	customerwishlist	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	dailysalesreport	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	discountedproducts	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	expireddiscounts	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	feedbacksummary	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	highdemandproducts	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	inactivecustomers	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	lowstockalerts	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	orderdetails	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	ordersawaitingpayment	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	orderstatussummary	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
□	pendingshipments	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-
	Console arcategories	☆	Browse	Structure	Search	Insert	Edit	Drop	~0	View	---	-	-

Fig: View Table (SQL Output)



ALGORITHM: UNDEFINED

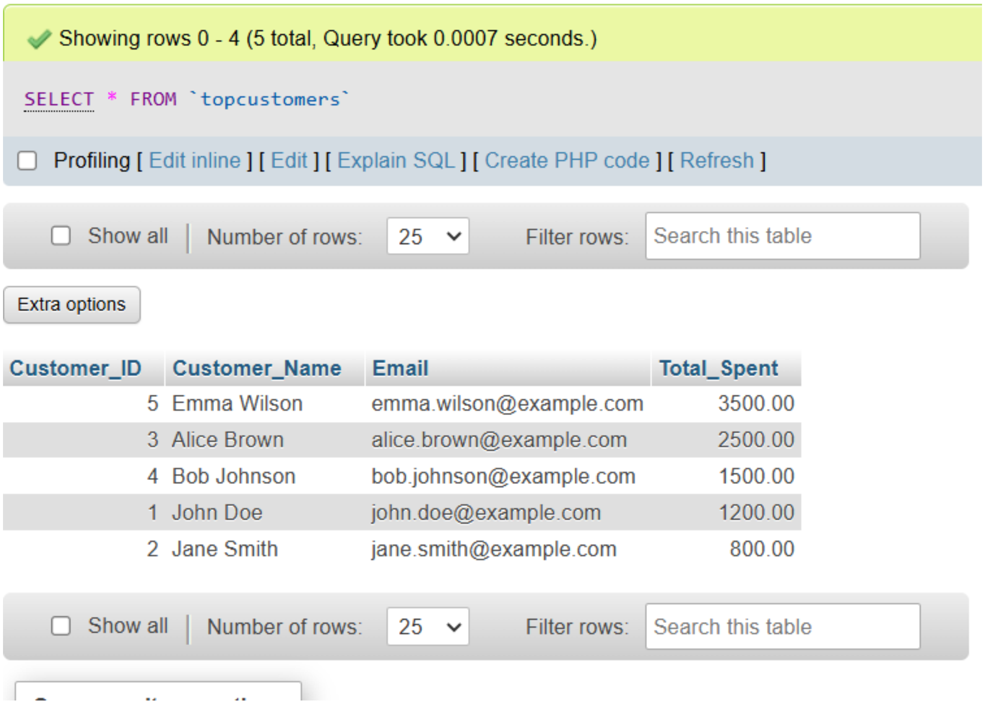
Definer: root@localhost

SQL SECURITY: DEFINER

Column names:

```
select `c`.`Customer_ID` AS `Customer_ID`,concat(`c`.`First_Name`,' ',`c`.`Last_Name`) AS `Customer_Name`,`c`.`Email` AS
`Email`,sum(`o`.`Order_Amount`) AS `Total_Spent` from (`ems`.`customer` `c` join `ems`.`order` `o` on(`c`.`Customer_ID` =
`o`.`Customer_ID`)) group by `c`.`Customer_ID` order by sum(`o`.`Order_Amount`) desc limit 10
```

Fig: View Code (Query Input)


✓ Showing rows 0 - 4 (5 total, Query took 0.0007 seconds.)

```
SELECT * FROM `topcustomers`
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
□ Show all
Number of rows: 25
Filter rows: Search this table
Extra options

Customer_ID	Customer_Name	Email	Total_Spent
5	Emma Wilson	emma.wilson@example.com	3500.00
3	Alice Brown	alice.brown@example.com	2500.00
4	Bob Johnson	bob.johnson@example.com	1500.00
1	John Doe	john.doe@example.com	1200.00
2	Jane Smith	jane.smith@example.com	800.00

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

Fig: View table

3.2 Performance Analysis

To ensure the database is efficient and scalable, performance tests were conducted on key operations, including:

- Query execution time for retrieving large datasets, such as all orders placed in a given month.
- Efficiency of procedures when inserting bulk data.
- Trigger performance under frequent inventory updates during sales events.

The results showed that:

- Indexing on primary and foreign keys significantly reduced query execution time.
- Using stored procedures minimized errors and increased efficiency.
- Proper normalization ensured smooth handling of large datasets without redundancy.

3.3 Results and Discussion

The implementation successfully delivered a robust and scalable database system for *Craftique*. Key outcomes include:

- Seamless management of customers, products, and orders using structured tables.
- Automation of common tasks through stored procedures and triggers, improving efficiency and reducing human error.
- Simplified reporting and analytics through views, aiding in decision-making.

The system was tested for various scenarios such as adding new users, placing orders, and handling payments. All tests yielded correct results, validating the system's functionality and reliability.

Chapter 4

Engineering Standards and Mapping

4.1 Impact on Society, Environment, and Sustainability

4.1.1 Impact on Life

Craftique serves as a bridge between artisans and customers, creating significant societal value:

- **Empowering Artisans**: By providing a platform to showcase their talent, artisans gain access to a broader customer base, increasing their income and livelihood opportunities.
- **Customer Benefits**: Customers can access unique, high-quality handmade goods, promoting appreciation for craftsmanship.

4.1.2 Impact on Society & Environment

- **Social Impact**:
  - Promotes local businesses and supports rural economies, creating a ripple effect in society.
  - Encourages cultural preservation by bringing traditional crafts into mainstream commerce.
- **Environmental Impact**:
  - Focuses on handmade products, which have a lower carbon footprint compared to mass-produced items.
  - Reduces reliance on industrial manufacturing and fosters sustainable practices.

4.1.3 Ethical Aspects

Craftique adheres to ethical standards to ensure fair practices, such as:

- **Fair Trade**: Artisans receive a fair price for their work, promoting equitable growth.
- **Transparency**: Customers are provided with accurate information about the origin and materials of products.
- **Data Privacy**: The platform safeguards user data through secure database practices and encryption mechanisms.

4.1.4 Sustainability Plan

The project contributes to sustainability by:

- Reducing waste through the promotion of durable, handmade goods.
- Encouraging the use of natural or eco-friendly materials in the creation of artisan products.
- Building a scalable system that can adapt to future growth without compromising resource efficiency.

4.2 Project Management and Team Work

The development of *Craftique* adhered to established project management principles to ensure timely and successful delivery. The following practices were implemented:

4.2.1 Task Breakdown and Planning

The project was divided into key milestones:

1. **Requirement Analysis**: Understanding user needs and defining database requirements.
2. **Database Design**: Creating an ER diagram, relational schema, and normalization.
3. **Implementation**: Writing SQL scripts, developing procedures, triggers, and view
4. **Testing**: Ensuring all operations function correctly under various scenarios.
5. **Documentation**: Preparing a comprehensive project report.

4.2.2 Collaboration and Team Roles

Although this was an individual effort, collaboration principles were followed, simulating team scenarios:

- **Communication**: Regularly documenting progress and challenges.
- **Role Simulation**: Taking on roles such as database designer, developer, and tester to manage different aspects of the project.
- **Peer Feedback**: Sharing progress with peers for constructive feedback.

4.2.3 Tools and Resources

- **XAMPP**: Used to set up the MySQL database and phpMyAdmin for development and testing.
- **DBMS Concepts**: Applied principles of database normalization, indexing, and data integrity.
- **Documentation Tools**: Microsoft Word for report writing and Lucidchart for designing the ER diagram.

Chapter 5

Conclusion

5.1 Summary

The *Craftique* project aimed to design and implement a database system for an e-commerce platform dedicated to artisan and handmade goods. Through systematic planning, implementation, and testing, the following were achieved:

- A robust database with well-structured tables to manage entities such as customers, products, sellers, orders, and shipments.
- Automation of critical processes using stored procedures and triggers, improving efficiency and reducing errors.
- Creation of views to facilitate reporting, analysis, and decision-making.
- Alignment with societal and environmental goals by promoting local artisans and sustainable practices.

This project not only serves as a functional database system but also demonstrates the application of database management principles in real-world scenarios.

5.2 Limitations

Despite its successful implementation, the project has some limitations:

1. **Lack of Frontend Integration**: The project focuses solely on the backend database and does not include a user interface for customers or administrators.
2. **Scalability Testing**: While the database is designed to handle growth, it has not been stress-tested under high transaction volumes.
3. **Payment Gateway Simulation**: The payment processing functionality is not integrated with real-world payment gateways, limiting its scope in actual deployment.

These limitations highlight areas for future improvement and expansion.

5.3 Future Work

The *Craftique* project provides a strong foundation for further development. Possible areas of improvement and expansion include:

- **Frontend Development**.
- **Scalability Enhancement.**
- **Integration with Payment Gateways.**
- **Advanced Analytics.**
- **Mobile Application.**

The *Craftique* project successfully demonstrates how a well-designed database system can support a niche ecommerce platform. It opens up numerous possibilities for future growth and scalability while maintaining its commitment to societal and environmental impact.

SQL File Link: [LINK]

Reference

[1] Artisan entrepreneurship: a systematic literature review and research agenda: https://www.researchgate.net/publication/325132814_Artisan_entrepreneurship_a_systematic_literature_review_and_research_agenda

[2] Entrepreneurial artisan products as regional tourism competitiveness: https://www.researchgate.net/publication/325132814_Artisan_entrepreneurship_a_systematic_literature_review_and_research_agenda

We thank you from the Team Laughter Crew.
The End.