UNIVERSITI TEKNOLOGI MALAYSIA

SEMESTER 3 SESSION 2023/2024

PROJECT 3

Project Proposal & Database Requirements

SECD2523: Database

SECTION 01

NO.	NAME	MATRIC NO
1	ANG CHUN WEI	A23CS0046
2	CHIN PEI WEN	A23CS0065
3	WELSON WOONG LU BIN	A23CS0196
4	TAN ZHAO HONG	A23CS0188

COURSE : SECBH

LECTURER'S NAME : Dr. Haslina binti Hashim

Table of content

1.0 Introduction

In today's fast-paced retail environment, operational efficiency, accurate inventory management, and customer satisfaction are critical to the success of any business, including small-scale operations like minimarts. BESTARI Mini Mart has relied on manual processes to manage its operations. This includes everything from tracking stock levels and managing sales to handling customer interactions and maintaining supplier relationships. However, such manual methods, while cost-effective in the short term, are prone to inefficiencies and errors. As the business has grown in size and scope, these processes have become increasingly time-consuming, leading to stock mismanagement, delays in reordering supplies, and difficulties in tracking financial performance.

To address these challenges and improve the minimart's overall operational efficiency, the development of a comprehensive BESTARI Minimart Database System has been proposed. The system will digitise and automate key aspects of the business, ensuring real-time data availability and better management of resources. By implementing this database system, BESTARI Minimart aims to transform its operations and enhance its ability to serve its customers effectively. A database system allows businesses to store, manage, and retrieve data efficiently. In the case of BESTARI Minimart, the database system will serve as the backbone of the store's daily operations, automating and integrating several key functions such as inventory management, financial tracking, staff scheduling, and supplier interactions.

In conclusion, by automating key processes such as inventory management, financial tracking, staff scheduling, and customer engagement, the system will drastically reduce the time and effort required to run the store. It will also provide management with the tools and insights they need to make informed decisions, improving the store's overall performance and profitability. As the minimart continues to grow, the database system will play a crucial role in ensuring that it can meet the demands of its customers efficiently and effectively. Ultimately, the BESTARI Minimart Database System will serve as a foundation for the store's long-term success.

2.0 Overview of project

In the database logical design phase for the minimart database system, we will convert the conceptual ERD developed in Project 2 into logical ERD. The adjustments will be made in this phase to address the unique complexities of managing minimart operations. This structured approach will lay the foundation for building an efficient and scalable database. Each entity in the logical ERD will be converted into a relations; schema with attributes defined and primary keys assigned to each table. For a minimart system, key entities such as staff, customer, transaction, supplier, and products must be clearly defined and interlinked.

Normalization will also be conducted during this stage to eliminate redundancies and dependencies in the relational schema. For the minimart database, normalization ensures that product details are not duplicated, sales transactions are accurately associated with customers and products, and overall data integrity is preserved. Simultaneously, the data dictionary will be updated to reflect any changes brought about by normalization to ensure that all database components are documented accurately.

Finally, SQL statements will be proposed to validate the logical ERD against the system's transaction requirements, focusing on interface design and usability. This step is critical to ensure the database structure supports essential operations such as sales processing, inventory management and supplier transactions. For the minimart database system, the responsiveness and reliability of the database significantly impact the business efficiency and customer satisfaction to make this validation process a key aspect the design phase.

3.0 Database Conceptual Design

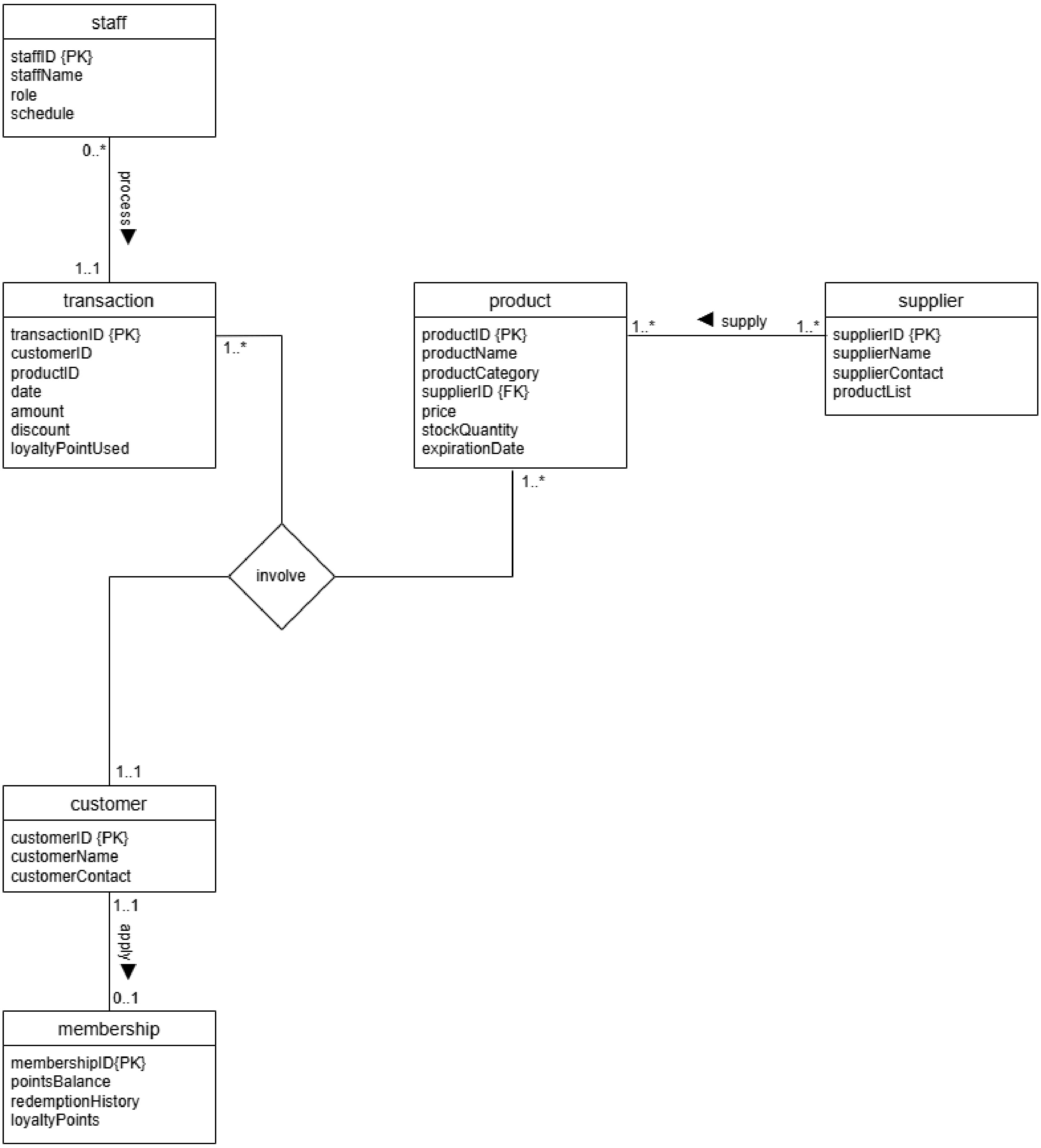
3.1 Updated Business Rules

1. Customer
    - Each customer must have a unique customer ID to identify their purchase history, preferences and loyalty points.
    - Customer data including name, contact details and membership status should be securely stored and updated as needed.
    - Customer enrolled in the loyalty program earn points for eligible purchases which can be redeemed according to the program's policies
    - Only one active loyalty membership is allowed per customer and points expire after a designed period if unused.

2. Staff
    - Each staff member must have a unique employee ID and records must include their role, work schedule and attendance history.
    - Employee performance and attendance records should be regularly reviewed and used as part of the staff evaluation process.

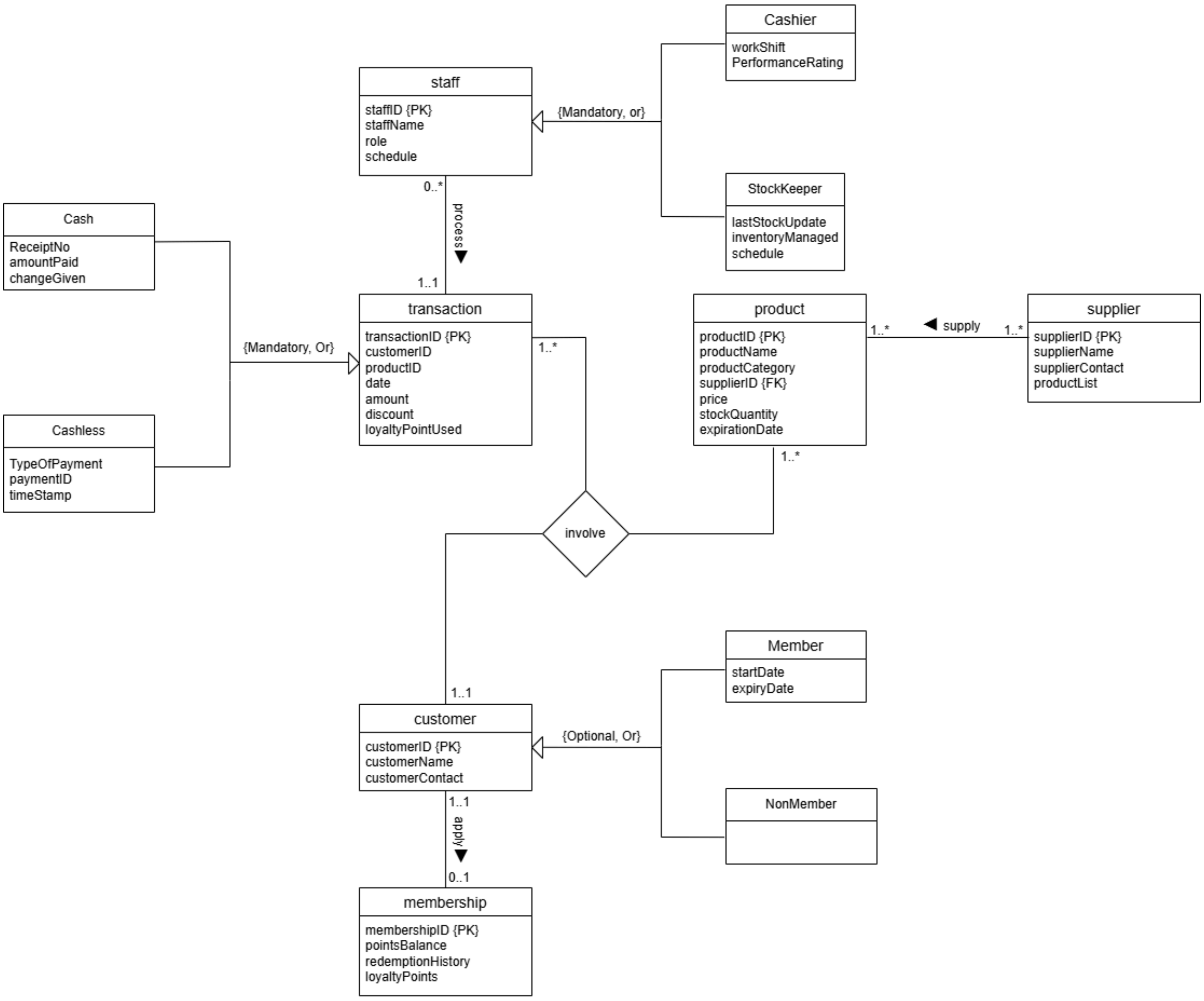
3. Supplier
    - Each supplier must have a unique supplier ID and the database should store contact information, supply terms and the list of products supplied.
    - Each of the deliveries must have an associated order number.
    - Payments to suppliers should only be made once products are received and verified as accurate against the order details.

3.2 Conceptual Entity-Relational Model (ERD)



Conceptional Entity-Relational Model (ERD)

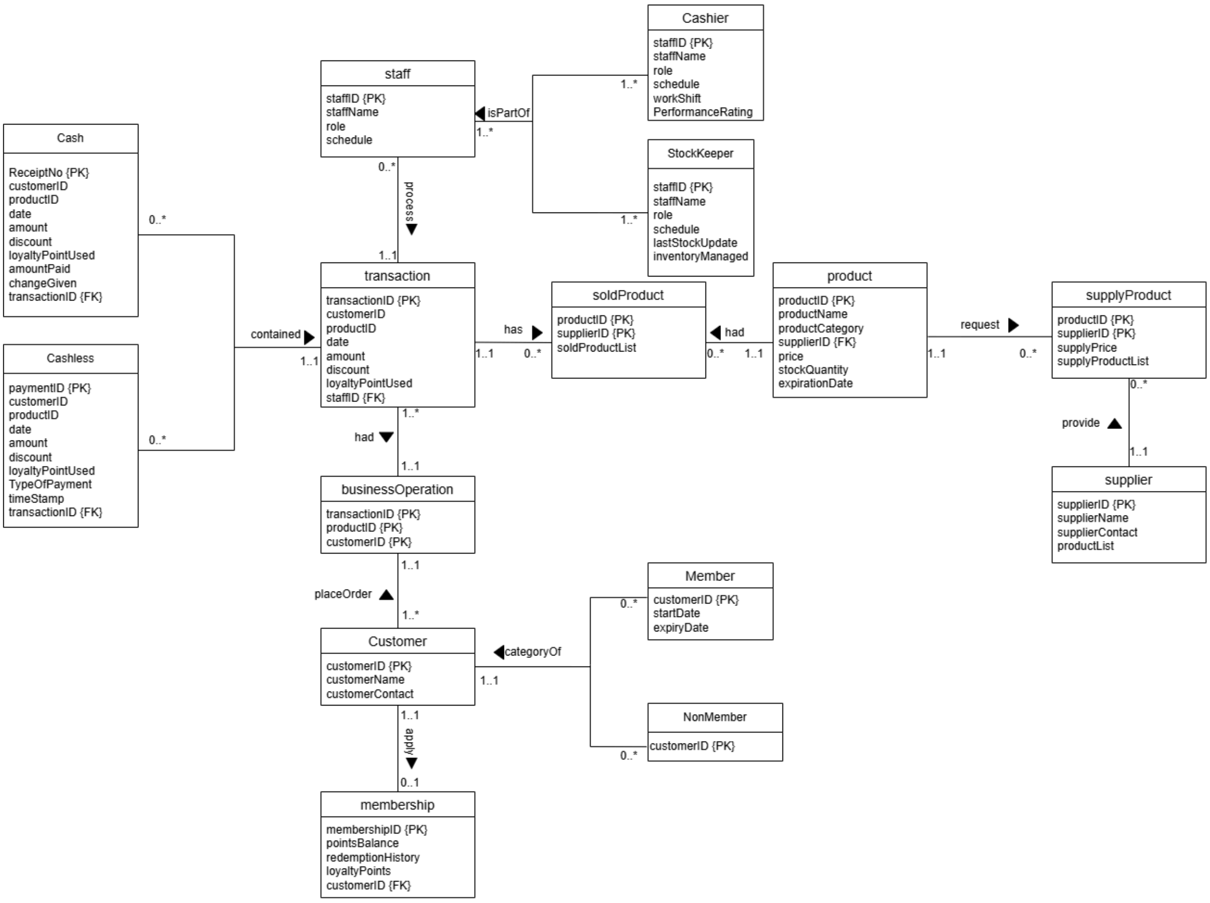
3.3 Enhanced Entity-Relational Model (EERD)



Enhanced Entity-Relational Model (EERD)

4.0 Database Logical Design

4.1 Logical ERD



4.2 Updated data dictionary

4.2.1 Description of Entity

Entity	**Description**	**Occurrence**
Customer	General term describing all the customers of the minimarket	Each customer can only apply one membership
Membership	General term describing the membership details for customer	Each membership only has one customerID
Member	General term describing the customer who applied for membership	Each member has a customerID and membership details
NonMember	General term describing customer without membership	Each nonMember has a customerID but does not have membership details
Staff	General term describing all the staff works in the minimarket	Each staff member can has multiple roles such as cashier and stock keeper
Cashier	General term describing staff assigned to handle customer purchases	Each cashiers can handle multiple transactions
StockKeeper	General term describing a specific type of staff responsible for managing inventory and stock	Each stockKeeper can manage multiple managing updates
Transaction	General term describing the purchases made by customers	Each transaction records each of the detail of purchase
SoldProduct	General term describing the details of products sold by minimarket	Each product has only one soldProduct to be sold to the customer
SupplyProduct	General term describing the supply details of products provided by supplier	Each supply has a supplier to one or more products they provide
BusinessOperation	General term describing the order placed by customers for products	Each operation occurs once per customer order for a product

Cash	General term describing the details of cash payment	Each cash transaction occurs once for a specific product and is linked to a customer and transaction
Cashless	General term describing the details of cashless payment	Each cashless payment occurs once for a specific product and transaction
Product	General term describing the inventory of products available in the store	Each product information is recorded with details about quantity, price and supplier
Supplier	General term describing the supplier who provide the product to the store	Each supplier linked to multiple products and supply details

4.2.2 Description of Relationship

Entity	**Multiplicity**	**Relationship**	**Multiplicity**	**Entity**
Staff	0…*	process	1…1	Transaction
Supplier	1…1	provide	0…*	SupplyProduct
Customer	1…1	apply	0…1	Membership
Cashier	1…*	isPartOf	1…*	Staff
StockKeeper	1…*	isPartOf	1…*	Staff
Cash	0…*	contained	1…1	Transaction
Cashless	0…*	contained	1…1	Transaction
Transaction	1…*	had	1…1	BusinessOperation
Customer	1…*	placeOrder	1…1	BusinessOperation
Member	0…*	categoryOf	1…1	Customer
NonMember	0…*	categoryOf	1…1	Customer
Product	1…1	request	0…*	SupplyProduct
Transaction	1…1	has	0…*	SupplyProduct
Transaction	1…1	has	0…*	SoldProduct
Product	1…1	had	0…*	SoldProduct

4.2.3 Description Attributes

Entity	Attributes	Description	Data Type	Null	Multi-valued
Staff	staffID	Uniquely identify a staff (PK)	INT	No	No
	staffName	Full name of staff	VARCHAR(100)	No	No
	role	Role of a staff	VARCHAR(50)	No	No
	schedule	Time of schedule	TEXT	No	No
Transaction	transactionID	Uniquely identify a transaction(PK)	INT	No	No
	customerID	Foreign key of Customer that uniquely identify a customer (FK)	INT	No	No
	productID	Foreign key of Product that uniquely identify a product (FK)	INT	No	No
	date	Date of transaction made	DATE	No	No
	amount	Amount of transaction	INT	No	No
	discount	Discount of product	INT	Yes	No
	loyaltyPointUsed	Point used for each transaction	INT	Yes	No
	staffID	Uniquely identify a staff (FK)	INT	No	No
SoldProduct	productID	Uniquely identify a product (PK)	INT	No	No
	supplierID	Uniquely identify a supplier (PK)	INT	No	Yes

	soldProductList	List of product sold by minimarket	VARCHAR(50)	No	No
SupplyProduct	productID	Uniquely identify a product (PK)	INT	No	No
	supplierID	Uniquely identify a supplier (PK)	INT	No	Yes
	supplyPrice	Price of the supply products	DECIMAL(10,2)	No	No
	supplyProductList	List of products supplied by supplier	VARCHAR(50)	No	No
BusinessOperation	transactionID	Uniquely identify a transaction(PK)	INT	No	No
	productID	Uniquely identify a product (PK)	INT	No	No
	customerID	Uniquely identify a customer (PK)	INT	No	No
Customer	customerID	Uniquely identify a customer (PK)	INT	No	No
	customerName	Name of customer	VARCHAR(100)	No	No
	customerContact	Contact number of customer	VARCHAR(15)	No	Yes
Membership	membershipID	Uniquely identify a member	INT	Yes	No
	pointsBalance	Balance of membership points after a transaction	INT	Yes	No
	RedemptionHistory	History of point redemption	VARCHAR (20)	Yes	No
	loyaltyPoints	Points collected	INT	Yes	No
	customerID	Uniquely identify a	INT	No	No

		customer (FK)			
Product	productID	Uniquely identify a product (PK)	INT	No	No
	productName	Name of product	VARCHAR (50)	No	No
	productCategory	Category of product	VARCHAR (30)	No	No
	supplierID	Foreign Key of Supplier that uniquely identify a supplier (FK)	INT	No	No
	price	Price of product	DECIMAL(10,2)	No	No
	stockQuantity	Quantity of stock	INT	No	No
	expirationDate	Expired date of a product	DATE	No	No
Supplier	supplierID	Uniquely identify a supplier (PK)	INT	No	Yes
	supplierName	Name of supplier	VARCHAR (100)	No	No
	supplierContact	Contact number of supplier	VARCHAR(15)	No	No
	productList	List of product	VARCHAR (50)	No	No
Cashier	staffID	Uniquely identify a staff (PK)	INT	No	No
	staffName	Full name of staff	VARCHAR(100)	No	No
	role	Role of a staff	VARCHAR(20)	No	No
	schedule	Time of schedule	TEXT	No	No
	workShift	Work shift of the cashier	VARCHAR(50)	No	No
	performanceRati	Rating for staff	DECIMAL(5,2)	No	No

	ng	performance			
stockKeeper	staffID	Uniquely identify a staff (PK)	INT	No	No
	staffName	Full name of staff	VARCHAR(100)	No	No
	role	Role of a staff	VARCHAR(20)	No	No
	lastStockUpdate	Date that stock updated	DATE	No	No
	inventoryManaged	Amount of inventory	TEXT	No	No
	schedule	Time of schedule	TEXT	No	No
Cash	receiptNo	Number of receipt (PK)	VARCHAR(50)	No	No
	customerID	Uniquely identify a customer (FK)	INT	No	No
	productID	Uniquely identify a product (FK)	INT	No	No
	date	Date of transaction made	DATE	No	No
	amount	Amount of transaction	DECIMAL(10,2)	No	No
	discount	Discount of product	DECIMAL(10,2)	Yes	No
	loyaltyPointUsed	Point used for each transaction	INT	Yes	No
	amountPaid	Amount paid for transaction	DECIMAL(10,2)	No	No
	changeGiven	Change given for the transaction	DECIMAL(10,2)	No	No
	transactionID	Uniquely identify a transaction (FK)	INT	No	No

Cashless	paymentID	ID of payment (PK)	VARCHAR(50)	No	No
	customerID	Uniquely identify a customer (FK)	INT	No	No
	productID	Uniquely identify a product (FK)	INT	No	No
	date	Date of transaction made	DATE	No	No
	amount	Amount of transaction	DECIMAL(10,2)	No	No
	discount	Discount of product	DECIMAL(10,2)	Yes	No
	loyaltyPointUsed	Point used for each transaction	INT	Yes	No
	typeOfPayment	Payment method	VARCHAR (50)	No	No
	timeStamp	Time that make payment	TIMESTAMP	No	No
	transactionID	Uniquely identify a transaction (FK)	INT	No	No
Member	customerID	Uniquely identify a customer (PK)	INT	No	No
	startDate	Date of registered as a member	DATE	No	No
	expiryDate	Expired date of membership	DATE	No	No
NonMember	customerID	Uniquely identify a customer (PK)	INT	No	No

4.3 Normalization

1. Staff (staffID, staffName, role, schedule)
   FD1: staffID → staffName, role, schedule
   1NF & 2NF & 3NF & BCNF:
   Staff (staffID, staffName, role, schedule)

2. Transaction (transactionID, customerID, productID, date, amount, discount, loyaltyPointUse, staffID)
   FD1: transactionID → customerID, date, staffID
   FD2: transactionID, productID → amount, discount, loyaltyPointUse
   1NF & 2NF & 3NF & BCNF:
   Transaction (transactionID, customerID, date, staffID)
   TransactionDetails (transactionID, productID, amount, discount, loyaltyPointUse)

3. Product (productID, productName, productCategory, supplierID, price, stockQuantity, expirationDate)
   FD1: productID → productName, productCategory, supplierID, price, stockQuantity, expirationDate
   1NF & 2NF & 3NF & BCNF:
   Product (productID, productName, productCategory, supplierID, price, stockQuantity, expirationDate)

4. Supplier (supplierID, supplierName, supplierContact, productList)
   FD1: supplierID → supplierName, supplierContact
   FD2: supplierID → productList
   1NF & 2NF & 3NF & BCNF:
   Supplier (supplierID, supplierName, supplierContact)
   SupplierProduct(supplierID, productID)

5. Customer (customerID, customerName, customerContact)
   FD1: customerID → customerName, customerContact
   1NF & 2NF & 3NF & BCNF:
   Customer (customerID, customerName, customerContact)

6. Membership (membershipID, pointsBalance, redemptionHistory, loyaltyPoint, customerID)
   FD1: membershipID → pointsBalance, customerID
   FD2: membershipID → redemptionHistory
   1NF & 2NF & 3NF & BCNF:
   Membership(membershipID, pointsBalance, customerID)
   RedemptionHistory(redemptionID, membershipID, date, pointsUsed)

7. Cash (transactionID, customerID, productID, date, amount, discount, loyaltyPointUse, staffID, receiptNo, amountPaid, changeGiven)
   FD1: transactionID → receiptNo, date, amount, amountPaid, changeGiven
   1NF & 2NF & 3NF & BCNF:
   Cash(transactionID, receiptNo, date, amount, amountPaid, changeGiven)

8. Cashless (transactionID, customerID, productID, date, amount, discount, loyaltyPointUse, staffID, typeOfPayment, paymentID, timeStamp)
   FD1:transactionID → paymentID, date, amount, typeOfPayment, timeStamp
   1NF & 2NF & 3NF & BCNF:
   Cashless(transactionID, paymentID, typeOfPayment, date, amount, timeStamp)

9. Cashier (staffID, staffName, role, schedule, workshift, performanceRating)
   FD1: staffID → workshift, performanceRating
   1NF & 2NF & 3NF & BCNF:
   Cashier(staffID, workshift, performanceRating)

10. stockKeeper (staffID, staffName, role, schedule, lastStockUpdate, inventoryManaged, schedule)
    FD1: staffID → lastStockUpdate, inventoryManaged, schedule
    1NF & 2NF & 3NF & BCNF:
    StockKeeper(staffID, lastStockUpdate, inventoryManaged, schedule)

11. Member (customerID, startDate, expiryDate)
    FD1: customerID → startDate, expiryDate
    1NF & 2NF & 3NF & BCNF:
    Member(customerID, startDate, expiryDate)

12. nonMember (customerID)
    FD1: customerID → NULL
    1NF & 2NF & 3NF & BCNF:
    NonMember(customerID)

13. supplyProduct (productID, supplierID, supplyPrice, supplyProductList)
    FD1: productID, supplierID → supplyPrice
    FD2: productID, supplierID → supplyProductList
    1NF & 2NF & 3NF & BCNF:
    SupplyProduct(productID, supplierID, supplyPrice)
    suppliedProduct (productID, supplierID, suppliedQuantity)

14. soldProduct (transactionID, productID, soldProductList)
    FD1: transactionID,productID → soldProductList
    1NF & 2NF & 3NF & BCNF:
    SoldProduct (transactionID, productID, soldQuantity)

15. businessOperation (transactionID, productID, customerID)
    FD1: transactionID, productID → customerID
    1NF & 2NF & 3NF & BCNF:
    BusinessOperationDetails (transactionID, productID)

5.0 Relational Database Schemas (after Normalization)

Staff (staffID, staffName, role, schedule)

Transaction (transactionID, customerID, date, staffID)

TransactionDetails (transactionID, productID, amount, discount, loyaltyPointUse)

Product (productID, productName, productCategory, supplierID, price, stockQuantity, expirationDate)

Supplier (supplierID, supplierName, supplierContact)

SupplierProduct (supplierID, productID)

Customer (customerID, customerName, customerContact)

Membership (membershipID, pointsBalance, customerID)

RedemptionHistory (redemptionID, membershipID, date, pointsUsed)

Cash (transactionID, receiptNo, date, amount, amountPaid, changeGiven)

Cashless (transactionID, paymentID, typeOfPayment, date, amount, timeStamp)

Cashier (staffID, workshift, performanceRating, schedule)

StockKeeper (staffID, lastStockUpdate, inventoryManaged, schedule)

Member (customerID, startDate, expiryDate)

NonMember (customerID)

SupplyProduct (productID, supplierID, supplyPrice)

suppliedProduct (productID, supplierID, suppliedQuantity)

SoldProduct (transactionID, productID, soldQuantity)

BusinessOperationDetails (transactionID, productID)

Staff

staffID	staffName	role	schedule

Transaction

transactionID	customerID	date	staffID

TransactionDetails

transactionID	productID	amount	discount	loyaltyPointUse

Product

productID	productName	productCategory	supplierID	price	stockQuantity	expirationDate

Supplier

supplierID	supplierName	supplierContact

SupplierProduct

supplierID	productID

Customer

customerID	customerName	customerContact

Membership

membershipID	pointsBalance	customerID

RedemptionHistory

redemptionID	membershipID	date	pointsUsed

Cash

transactionID	receiptNo	date	amount	amountPaid	changeGiven

Cashless

transactionID	paymentID	date	amount	typeOfPayment	timeStamp

Cashier

staffID	workshift	performanceRating	schedule

StockKeeper

staffID	lastStockUpdate	inventoryManaged	schedule

Member

customerID	startDate	expiryDate

NonMember

customerID

SupplyProduct

productID	supplierID	supplyPrice

SuppliedProduct

productID	supplierID	suppliedQuantity

SoldProduct

transactionID	productID	soldQuantity

BusinessOperationDetails

transactionID	productID

6.0 SQL Statements (DDL & DML)

6.1 DDL

DDL:

```
-- Create Customer table
CREATE TABLE Customer (
   customerID INT PRIMARY KEY,
   customerName VARCHAR(100),
   customerContact VARCHAR(15)
);

-- Create Staff table
CREATE TABLE Staff (
   staffID INT PRIMARY KEY,
   staffName VARCHAR(100),
   role VARCHAR(50),
   schedule TEXT
);

-- Create Product table
CREATE TABLE Product (
   productID INT PRIMARY KEY,
   productName VARCHAR(100),
   productCategory VARCHAR(50),
   supplierID INT,
   price DECIMAL(10, 2),
   stockQuantity INT,
   expirationDate DATE
);

-- Create Supplier table
```

```
CREATE TABLE Supplier (
   supplierID INT PRIMARY KEY,
   supplierName VARCHAR(100),
   supplierContact VARCHAR(15)
);

-- Create SupplierProduct table
CREATE TABLE SupplierProduct (
   supplierID INT,
   productID INT,
   PRIMARY KEY (supplierID, productID),
   FOREIGN KEY (supplierID) REFERENCES Supplier(supplierID),
   FOREIGN KEY (productID) REFERENCES Product(productID)
);

-- Create SupplyProduct table
CREATE TABLE SupplyProduct (
   productID INT,
   supplierID INT,
   supplyPrice DECIMAL(10, 2),
   PRIMARY KEY (productID, supplierID),
   FOREIGN KEY (productID) REFERENCES Product(productID),
   FOREIGN KEY (supplierID) REFERENCES Supplier(supplierID)
);

-- Create SuppliedProduct table
CREATE TABLE SuppliedProduct (
   productID INT,
   supplierID INT,
    suppliedQuantity INT,
    PRIMARY KEY (productID, supplierID),
    FOREIGN KEY (productID) REFERENCES Product(productID),
    FOREIGN KEY (supplierID) REFERENCES Supplier(supplierID)
);

-- Create Membership table
CREATE TABLE Membership (
    membershipID INT PRIMARY KEY,
    pointsBalance INT,
    customerID INT,
    FOREIGN KEY (customerID) REFERENCES Customer(customerID)
);

-- Create Member table
CREATE TABLE Member (
    customerID INT PRIMARY KEY,
    startDate DATE,
    expiryDate DATE,
    FOREIGN KEY (customerID) REFERENCES Customer(customerID)
);

-- Create NonMember table
CREATE TABLE NonMember (
    customerID INT PRIMARY KEY,
    FOREIGN KEY (customerID) REFERENCES Customer(customerID)
);

-- Create Transaction table
```

```
CREATE TABLE Transaction (
    transactionID INT PRIMARY KEY,
    customerID INT,
    date DATE,
    staffID INT,
    FOREIGN KEY (customerID) REFERENCES Customer(customerID),
    FOREIGN KEY (staffID) REFERENCES Staff(staffID)
);

-- Create TransactionDetails table
CREATE TABLE TransactionDetails (
    transactionID INT,
    productID INT,
    amount DECIMAL(10, 2),
    discount DECIMAL(10, 2),
    loyaltyPointUse INT,
    PRIMARY KEY (transactionID, productID),
    FOREIGN KEY (transactionID) REFERENCES Transaction(transactionID),
    FOREIGN KEY (productID) REFERENCES Product(productID)
);

-- Create RedemptionHistory table
CREATE TABLE RedemptionHistory (
    redemptionID INT PRIMARY KEY,
    membershipID INT,
    date DATE,
    pointsUsed INT,
    FOREIGN KEY (membershipID) REFERENCES Membership(membershipID)
);
```

```
-- Create Cash table
CREATE TABLE Cash (
    transactionID INT PRIMARY KEY,
    receiptNo VARCHAR(50),
    date DATE,
    amount DECIMAL(10, 2),
    amountPaid DECIMAL(10, 2),
    changeGiven DECIMAL(10, 2),
    FOREIGN KEY (transactionID) REFERENCES Transaction(transactionID)
);

-- Create Cashless table
CREATE TABLE Cashless (
    transactionID INT PRIMARY KEY,
    paymentID VARCHAR(50),
    typeOfPayment VARCHAR(50),
    date DATE,
    amount DECIMAL(10, 2),
    timeStamp TIMESTAMP,
    FOREIGN KEY (transactionID) REFERENCES Transaction(transactionID)
);

-- Create SoldProduct table
CREATE TABLE SoldProduct (
    transactionID INT,
    productID INT,
    soldQuantity INT,
    PRIMARY KEY (transactionID, productID),
    FOREIGN KEY (transactionID) REFERENCES Transaction(transactionID),
    FOREIGN KEY (productID) REFERENCES Product(productID)
);

-- Create BusinessOperationDetails table
CREATE TABLE BusinessOperationDetails (
    transactionID INT,
    productID INT,
    PRIMARY KEY (transactionID, productID),
    FOREIGN KEY (transactionID) REFERENCES Transaction(transactionID),
    FOREIGN KEY (productID) REFERENCES Product(productID)
);

-- Create Cashier table
CREATE TABLE Cashier (
    staffID INT PRIMARY KEY,
    workshift VARCHAR(50),
    performanceRating DECIMAL(5, 2),
    schedule TEXT,
    FOREIGN KEY (staffID) REFERENCES Staff(staffID)
);
-- Create StockKeeper table
CREATE TABLE StockKeeper (
    staffID INT PRIMARY KEY,
    lastStockUpdate DATE,
    inventoryManaged TEXT,
    schedule TEXT,
    FOREIGN KEY (staffID) REFERENCES Staff(staffID)
);
```

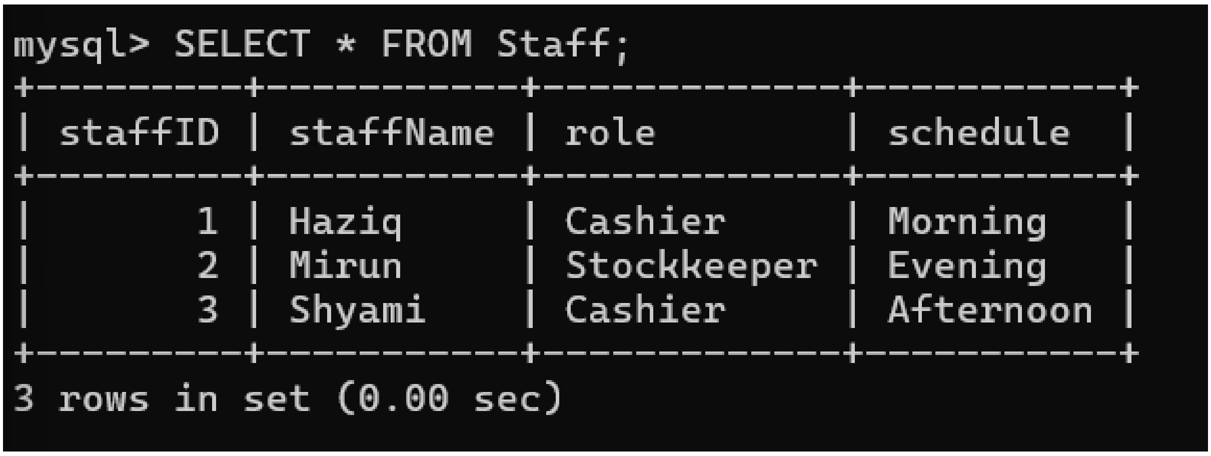
6.2 DML & Test Query

DML:
-- Insert data into Staff table

INSERT INTO Staff (staffID, staffName, role, schedule)
VALUES (1, 'Haziq', 'Cashier', 'Morning');

INSERT INTO Staff (staffID, staffName, role, schedule)
VALUES (2, 'Mirun', 'StockKeeper', 'Evening');

INSERT INTO Staff (staffID, staffName, role, schedule)
VALUES (3, 'Shyami', 'Cashier', 'Afternoon');



```
mysql> SELECT * FROM Staff;
+---------+-----------+-------------+-----------+
| staffID | staffName | role        | schedule  |
+---------+-----------+-------------+-----------+
|       1 | Haziq     | Cashier     | Morning   |
|       2 | Mirun     | Stockkeeper | Evening   |
|       3 | Shyami    | Cashier     | Afternoon |
+---------+-----------+-------------+-----------+
3 rows in set (0.00 sec)
```

Diagram 6.2.1: Output for Staff Table

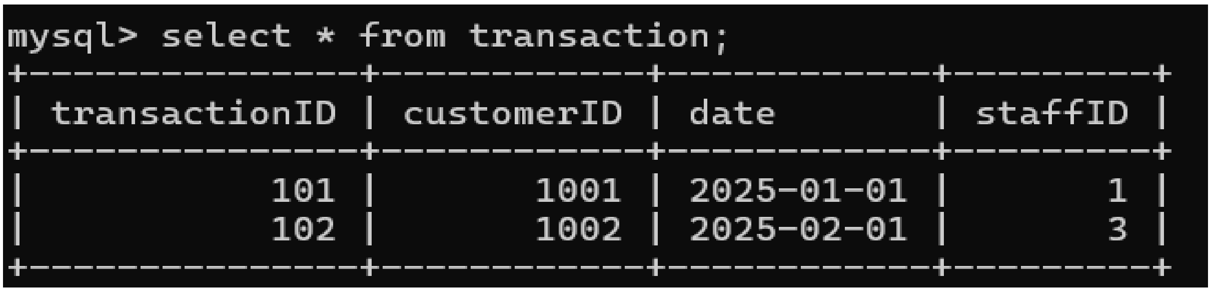
-- Insert data into Transaction table

INSERT INTO `Transaction` (transactionID, customerID, date, staffID)

VALUES (101, 1001, '2025-01-01', 1);

INSERT INTO `Transaction` (transactionID, customerID, date, staffID)

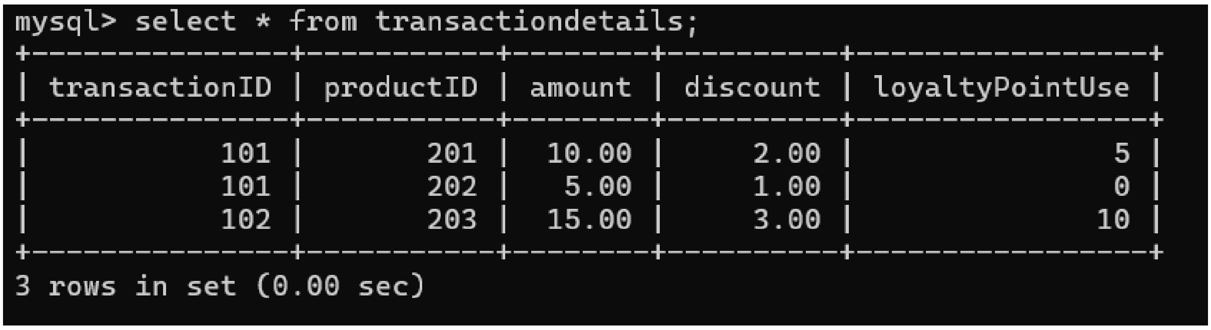
VALUES (102, 1002, '2025-02-01', 3);



```
mysql> select * from transaction;
+---------------+------------+------------+---------+
| transactionID | customerID | date       | staffID |
+---------------+------------+------------+---------+
|           101 |       1001 | 2025-01-01 |       1 |
|           102 |       1002 | 2025-02-01 |       3 |
+---------------+------------+------------+---------+
```

Diagram 6.2.2: Output for Transaction Table

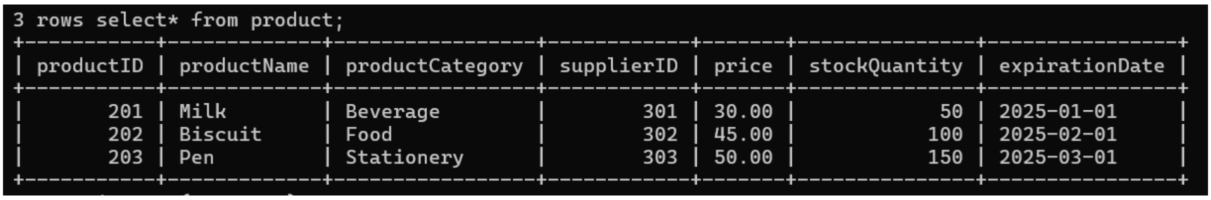
```sql
-- Insert data into TransactionDetails table

INSERT INTO TransactionDetails (transactionID, productID, amount, discount, loyaltyPointUse)
VALUES (101, 201, 10, 2, 5);

INSERT INTO TransactionDetails (transactionID, productID, amount, discount, loyaltyPointUse)
VALUES (101, 202, 5, 1, 0);

INSERT INTO TransactionDetails (transactionID, productID, amount, discount, loyaltyPointUse)
VALUES (102, 203, 15, 3, 10);
```



```
mysql> select * from transactiondetails;
+---------------+-----------+--------+----------+-----------------+
| transactionID | productID | amount | discount | loyaltyPointUse |
+---------------+-----------+--------+----------+-----------------+
|           101 |       201 |  10.00 |     2.00 |               5 |
|           101 |       202 |   5.00 |     1.00 |               0 |
|           102 |       203 |  15.00 |     3.00 |              10 |
+---------------+-----------+--------+----------+-----------------+
3 rows in set (0.00 sec)
```

Diagram 6.2.3: Output for TransactionDetails Table

```sql
-- Insert data into Product table

INSERT INTO Product (productID, productName, productCategory, supplierID, price, stockQuantity, expirationDate)

VALUES (201, 'Milk', 'Beverage', 301, 30.00, 50, '2025-01-01');

INSERT INTO Product (productID, productName, productCategory, supplierID, price, stockQuantity, expirationDate)

VALUES (202, 'Biscuit', 'Food', 302, 45.00, 100, '2025-02-01');

INSERT INTO Product (productID, productName, productCategory, supplierID, price, stockQuantity, expirationDate)

VALUES (203, 'Pen', 'Stationery', 303, 50.00, 150, '2025-03-01');
```



```
3 rows select* from product;
+-----------+-------------+-----------------+------------+-------+---------------+----------------+
| productID | productName | productCategory | supplierID | price | stockQuantity | expirationDate |
+-----------+-------------+-----------------+------------+-------+---------------+----------------+
|       201 | Milk        | Beverage        |        301 | 30.00 |            50 | 2025-01-01     |
|       202 | Biscuit     | Food            |        302 | 45.00 |           100 | 2025-02-01     |
|       203 | Pen         | Stationery      |        303 | 50.00 |           150 | 2025-03-01     |
+-----------+-------------+-----------------+------------+-------+---------------+----------------+
```

Diagram 6.2.4: Output for Product Table

-- Insert data into Supplier table

INSERT INTO Supplier (supplierID, supplierName, supplierContact)

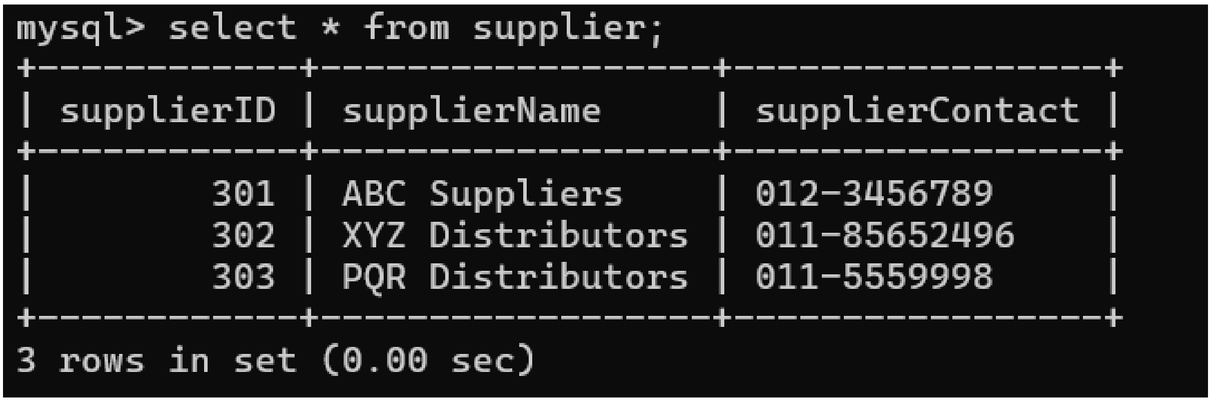
VALUES (301, 'ABC Suppliers', '012-3456789');

INSERT INTO Supplier (supplierID, supplierName, supplierContact)

VALUES (302, 'XYZ Distributors', '011-85652496');

INSERT INTO Supplier (supplierID, supplierName, supplierContact)

VALUES (303, 'PQR Distributors', '011-5559998');



```
mysql> select * from supplier;
+------------+------------------+-----------------+
| supplierID | supplierName     | supplierContact |
+------------+------------------+-----------------+
|        301 | ABC Suppliers    | 012-3456789     |
|        302 | XYZ Distributors | 011-85652496    |
|        303 | PQR Distributors | 011-5559998     |
+------------+------------------+-----------------+
3 rows in set (0.00 sec)
```

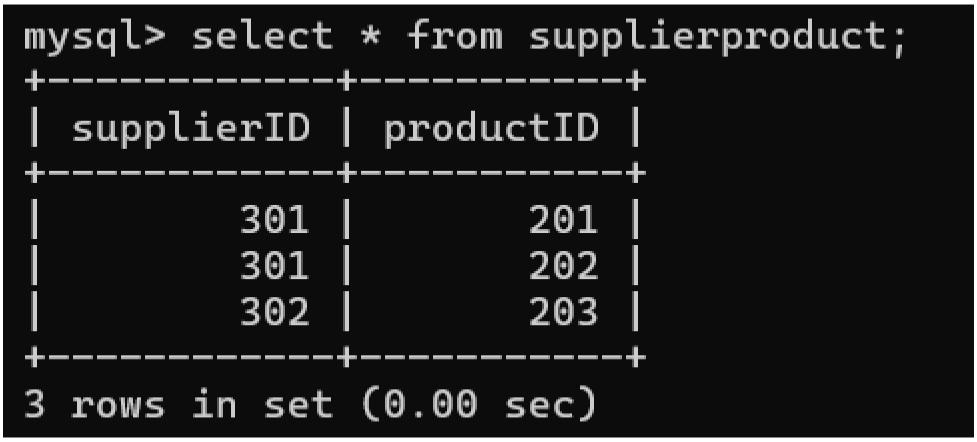
Diagram 6.2.5: Output for Supplier Table

-- Insert data into SupplierProduct table

INSERT INTO SupplierProduct (supplierID, productID)
VALUES (301, 201);

INSERT INTO SupplierProduct (supplierID, productID)
VALUES (301, 202);

INSERT INTO SupplierProduct (supplierID, productID)
VALUES (302, 203);



```
mysql> select * from supplierproduct;
+------------+-----------+
| supplierID | productID |
+------------+-----------+
|        301 |       201 |
|        301 |       202 |
|        302 |       203 |
+------------+-----------+
3 rows in set (0.00 sec)
```

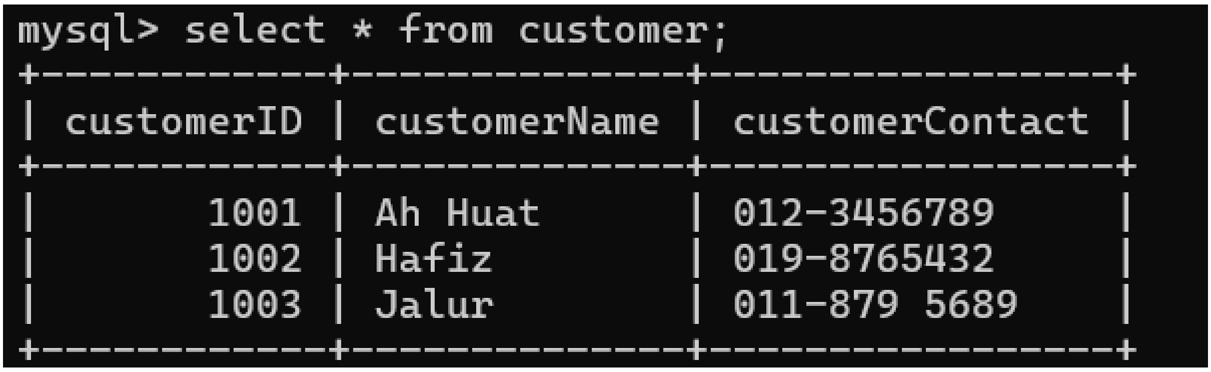
Diagram 6.2.6: Output for SupplierProduct Table

-- Insert data into Customer table

INSERT INTO Customer (customerID, customerName, customerContact)
VALUES (1001, 'Ah Huat', '012- 345 6789');

INSERT INTO Customer (customerID, customerName, customerContact)
VALUES (1002, 'Hafiz', '019-876 5432');

INSERT INTO Customer (customerID, customerName, customerContact)
VALUES (1003, 'Jalur', '011-879 5689');



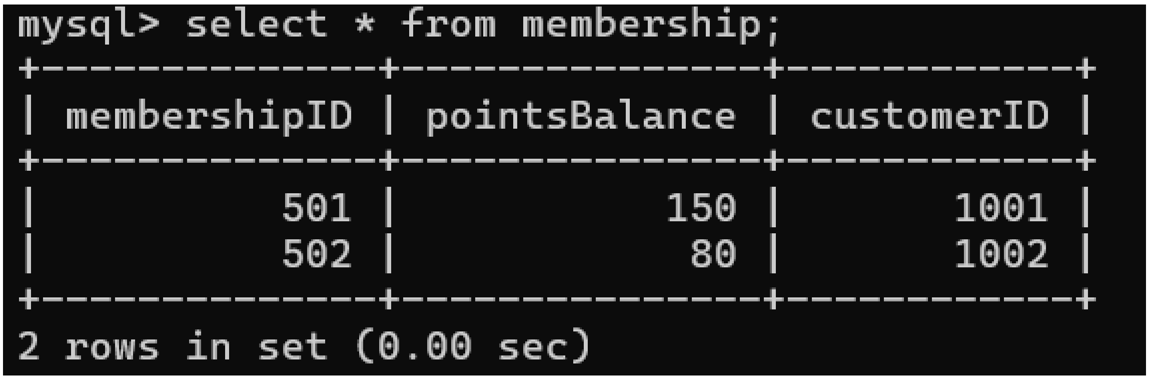
```
mysql> select * from customer;
+------------+--------------+-----------------+
| customerID | customerName | customerContact |
+------------+--------------+-----------------+
|       1001 | Ah Huat      | 012-3456789     |
|       1002 | Hafiz        | 019-8765432     |
|       1003 | Jalur        | 011-879 5689    |
+------------+--------------+-----------------+
```

Diagram 6.2.7: Output for Customer Table

–Insert data into Membership table

INSERT INTO Membership (membershipID, pointsBalance, customerID)
VALUES (501, 150, 1001);

INSERT INTO Membership (membershipID, pointsBalance, customerID)
VALUES (502, 80, 1002);



```
mysql> select * from membership;
+--------------+---------------+------------+
| membershipID | pointsBalance | customerID |
+--------------+---------------+------------+
|          501 |           150 |       1001 |
|          502 |            80 |       1002 |
+--------------+---------------+------------+
2 rows in set (0.00 sec)
```

Diagram 6.2.8: Output for Membership Table

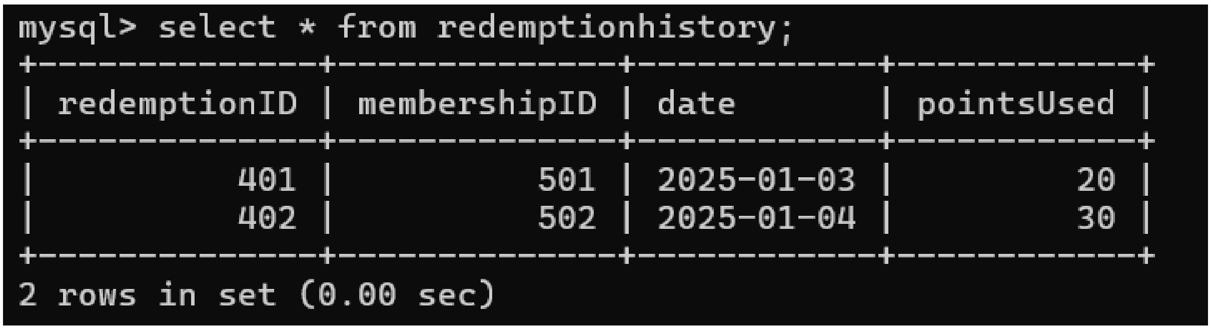
-- Insert data into RedemptionHistory table

INSERT INTO RedemptionHistory (redemptionID, membershipID, date, pointsUsed)

VALUES (401, 501, '2025-01-03', 20);

INSERT INTO RedemptionHistory (redemptionID, membershipID, date, pointsUsed)

VALUES (402, 502, '2025-01-04', 30);

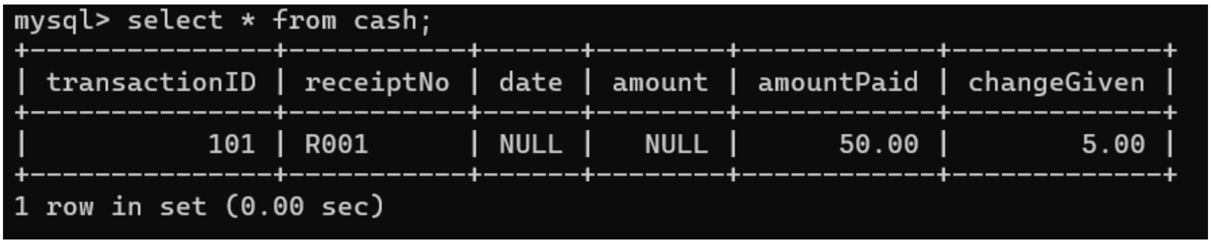


```
mysql> select * from redemptionhistory;
+--------------+--------------+------------+------------+
| redemptionID | membershipID | date       | pointsUsed |
+--------------+--------------+------------+------------+
|          401 |          501 | 2025-01-03 |         20 |
|          402 |          502 | 2025-01-04 |         30 |
+--------------+--------------+------------+------------+
2 rows in set (0.00 sec)
```

Diagram 6.2.9: Output for RedemptionHistory Table

-- Insert data into Cash table

INSERT INTO Cash (transactionID, receiptNo, amountPaid, changeGiven)
VALUES (101, 'R001', 50.00, 5.00);

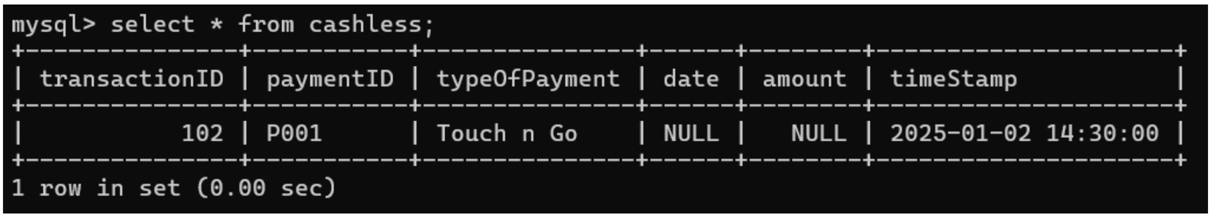


```
mysql> select * from cash;
+---------------+-----------+------+--------+------------+-------------+
| transactionID | receiptNo | date | amount | amountPaid | changeGiven |
+---------------+-----------+------+--------+------------+-------------+
|           101 | R001      | NULL |   NULL |      50.00 |        5.00 |
+---------------+-----------+------+--------+------------+-------------+
1 row in set (0.00 sec)
```

Diagram 6.2.10: Output for Cash Table

-- Insert data into Cashless table

INSERT INTO Cashless (transactionID, paymentID, typeOfPayment, timeStamp)
VALUES (102, 'P001', 'Touch n Go', '2025-01-02 14:30:00');



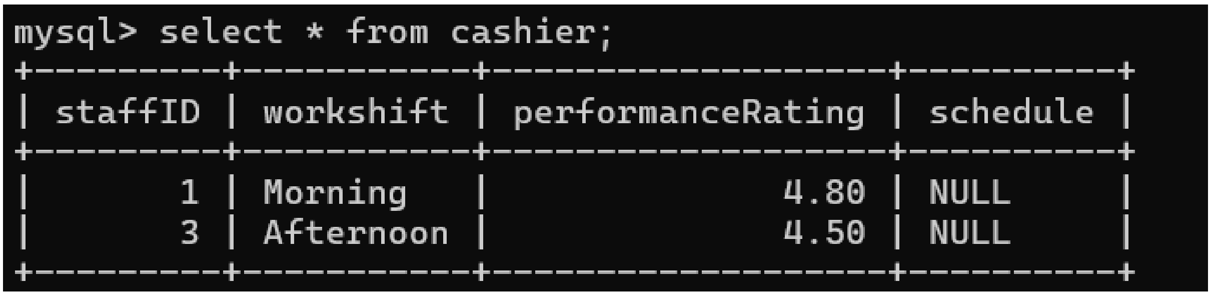
```
mysql> select * from cashless;
+---------------+-----------+---------------+------+--------+---------------------+
| transactionID | paymentID | typeOfPayment | date | amount | timeStamp           |
+---------------+-----------+---------------+------+--------+---------------------+
|           102 | P001      | Touch n Go    | NULL |   NULL | 2025-01-02 14:30:00 |
+---------------+-----------+---------------+------+--------+---------------------+
1 row in set (0.00 sec)
```

Diagram 6.2.11: Output for Cashless Table

-- Insert data into Cashier table

INSERT INTO Cashier (staffID, workshift, performanceRating)
VALUES (1, 'Morning', 4.8);

INSERT INTO Cashier (staffID, workshift, performanceRating)
VALUES (3, 'Afternoon', 4.5);



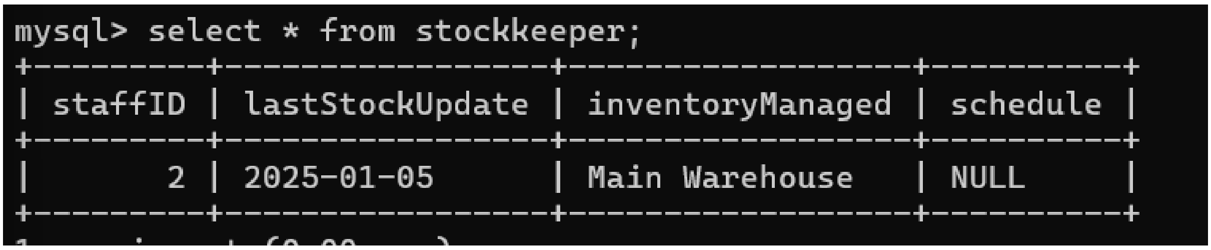
```
mysql> select * from cashier;
+---------+-----------+-------------------+----------+
| staffID | workshift | performanceRating | schedule |
+---------+-----------+-------------------+----------+
|       1 | Morning   |              4.80 | NULL     |
|       3 | Afternoon |              4.50 | NULL     |
+---------+-----------+-------------------+----------+
```

Diagram 6.2.12: Output for Cashier Table

-- Insert data into StockKeeper table

INSERT INTO StockKeeper (staffID, lastStockUpdate, inventoryManaged)

VALUES (2, '2025-01-05', 'Main Warehouse');

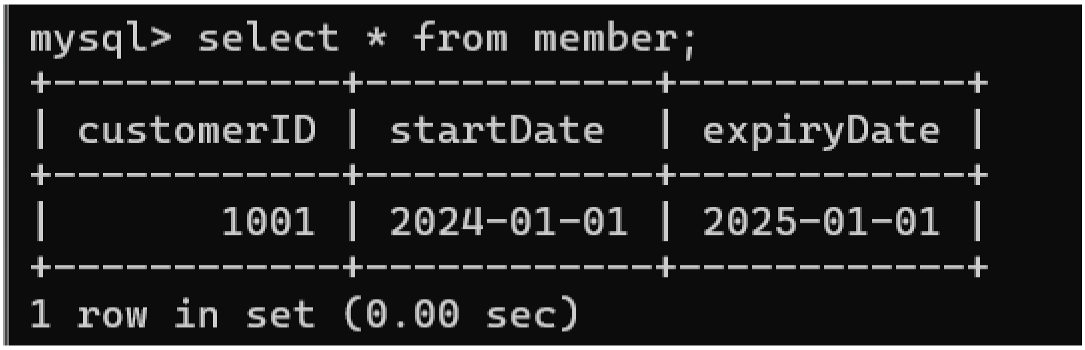


```
mysql> select * from stockkeeper;
+---------+-----------------+------------------+----------+
| staffID | lastStockUpdate | inventoryManaged | schedule |
+---------+-----------------+------------------+----------+
|       2 | 2025-01-05      | Main Warehouse   | NULL     |
+---------+-----------------+------------------+----------+
```

Diagram 6.2.13: Output for StockKeeper Table

-- Insert data into Member table

INSERT INTO Member (customerID, startDate, expiryDate)
VALUES (1001, '2024-01-01', '2025-01-01');

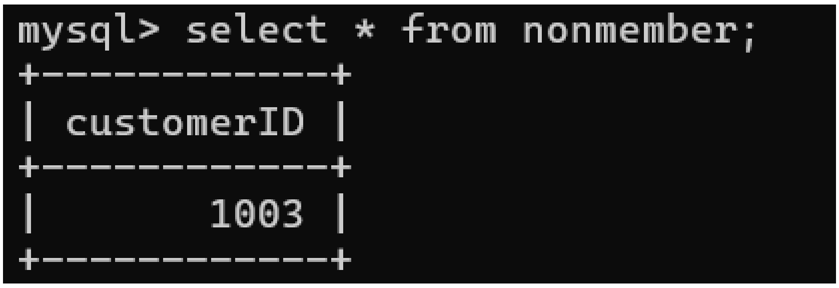


```
mysql> select * from member;
+------------+------------+------------+
| customerID | startDate  | expiryDate |
+------------+------------+------------+
|       1001 | 2024-01-01 | 2025-01-01 |
+------------+------------+------------+
1 row in set (0.00 sec)
```

Diagram 6.2.14: Output for Member Table

-- Insert data into NonMember table

INSERT INTO NonMember (customerID)
VALUES (1002);



```
mysql> select * from nonmember;
+------------+
| customerID |
+------------+
|       1003 |
+------------+
```

Diagram 6.2.15: Output for NonMember Table

-- Insert data into SupplyProduct table

INSERT INTO SupplyProduct (productID, supplierID, supplyPrice)

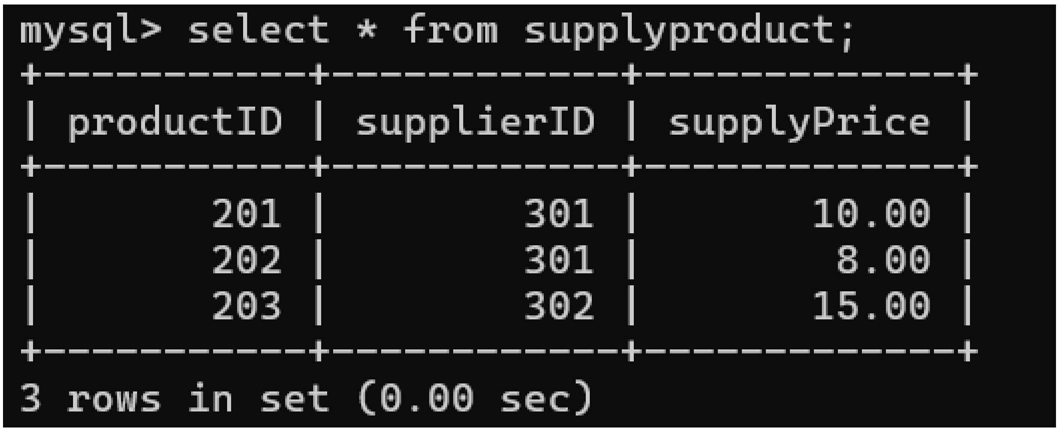
VALUES (201, 301, 10.00);

INSERT INTO SupplyProduct (productID, supplierID, supplyPrice)

VALUES (202, 301, 8.00);

INSERT INTO SupplyProduct (productID, supplierID, supplyPrice)

VALUES (203, 302, 15.00);



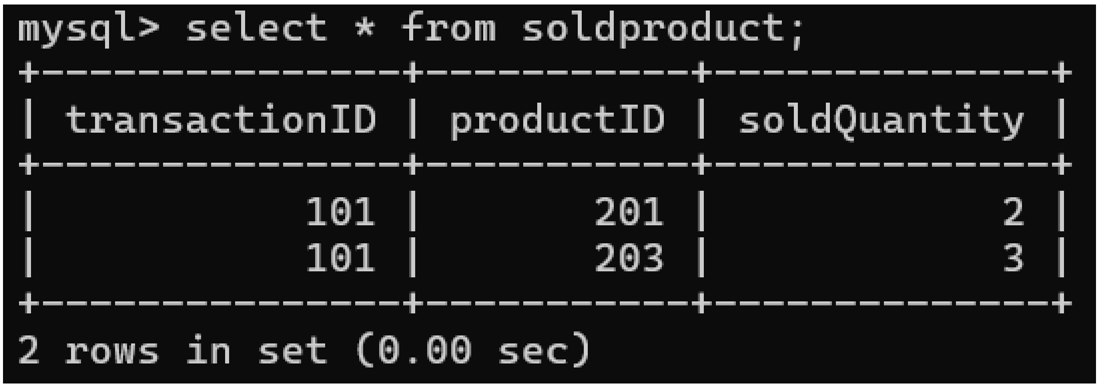
```
mysql> select * from supplyproduct;
+-----------+------------+-------------+
| productID | supplierID | supplyPrice |
+-----------+------------+-------------+
|       201 |        301 |       10.00 |
|       202 |        301 |        8.00 |
|       203 |        302 |       15.00 |
+-----------+------------+-------------+
3 rows in set (0.00 sec)
```

Diagram 6.2.16: Output for SupplyProduct Table

-- Insert data into SoldProduct table

INSERT INTO SoldProduct (transactionID, productID, soldProduct)
VALUES (101, 201, 2);

INSERT INTO SoldProduct (transactionID, productID, soldProduct)
VALUES (101, 203, 3);



```
mysql> select * from soldproduct;
+---------------+-----------+--------------+
| transactionID | productID | soldQuantity |
+---------------+-----------+--------------+
|           101 |       201 |            2 |
|           101 |       203 |            3 |
+---------------+-----------+--------------+
2 rows in set (0.00 sec)
```

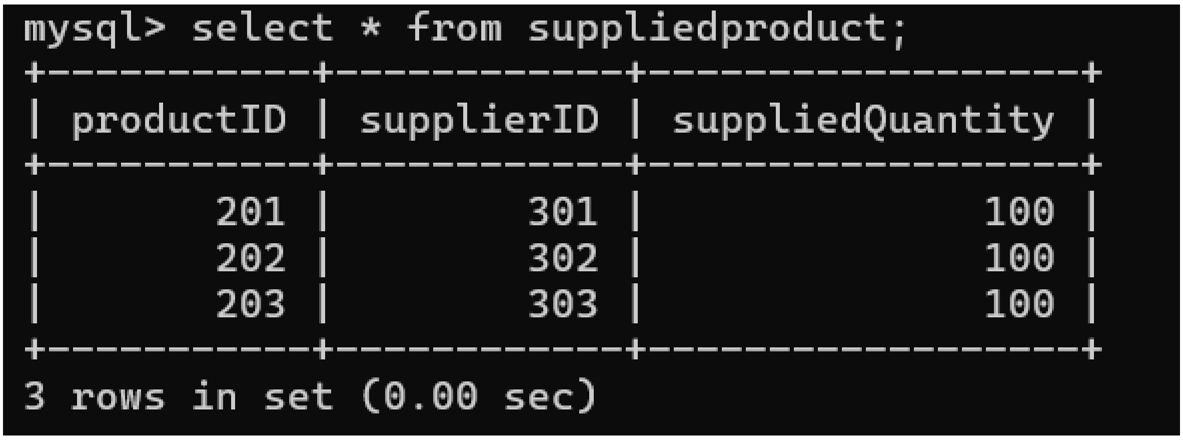
Diagram 6.2.17: Output for SoldProduct Table

-- Insert data into suppliedProduct table

INSERT INTO suppliedProduct (productID, supplierID, suppliedQuantity)
VALUES (201, 301, 100);

INSERT INTO suppliedProduct (productID, supplierID, suppliedQuantity)
VALUES (202, 302, 100);

INSERT INTO suppliedProduct (productID, supplierID, suppliedQuantity)
VALUES (203, 303, 100);



```
mysql> select * from suppliedproduct;
+-----------+------------+------------------+
| productID | supplierID | suppliedQuantity |
+-----------+------------+------------------+
|       201 |        301 |              100 |
|       202 |        302 |              100 |
|       203 |        303 |              100 |
+-----------+------------+------------------+
3 rows in set (0.00 sec)
```

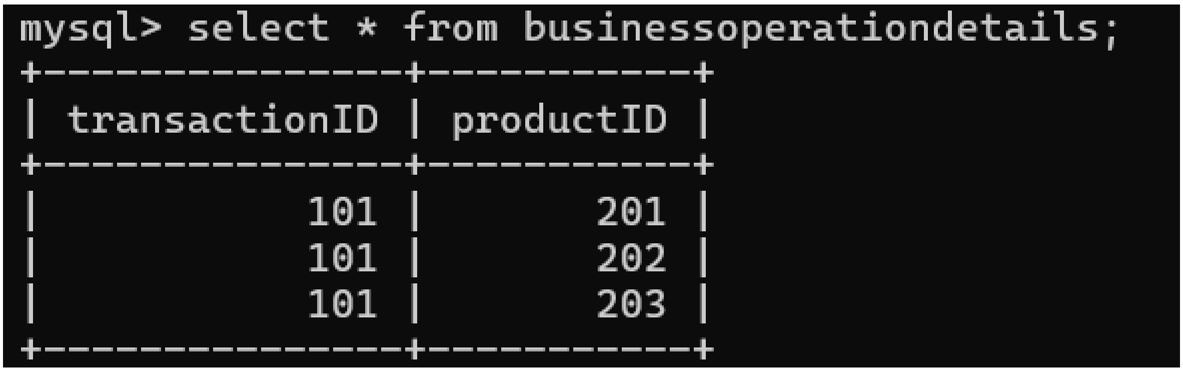
Diagram 6.2.18: Output for SuppliedProduct Table

-- Insert data into BusinessOperationDetails table

INSERT INTO BusinessOperationDetails (transactionID, productID)
VALUES (101, 201);

INSERT INTO BusinessOperationDetails (transactionID, productID)
VALUES (101, 202);

INSERT INTO BusinessOperationDetails (transactionID, productID)
VALUES (101, 203);

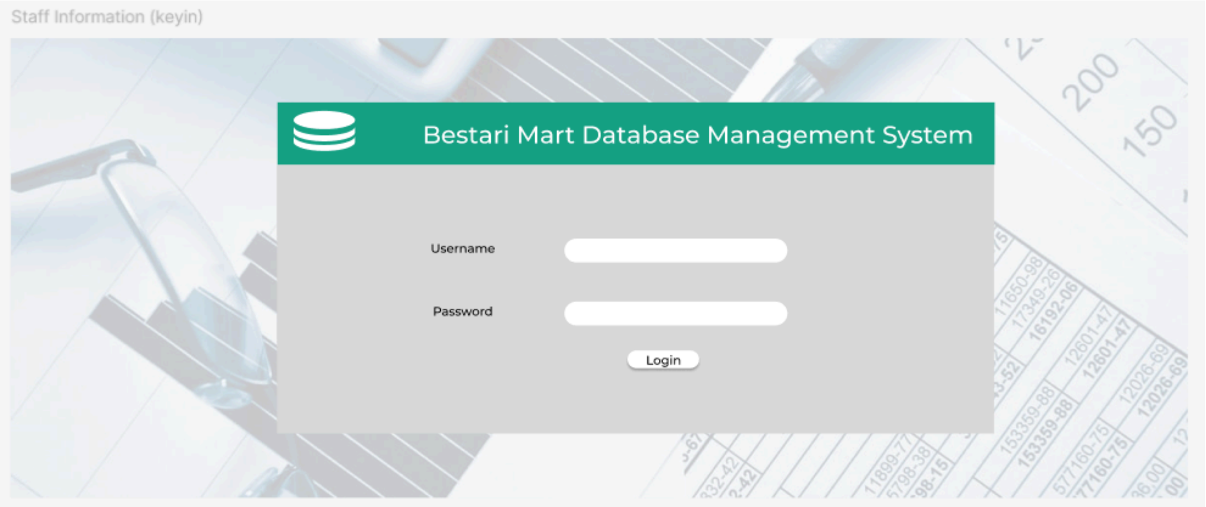


```
mysql> select * from businessoperationdetails;
+---------------+-----------+
| transactionID | productID |
+---------------+-----------+
|           101 |       201 |
|           101 |       202 |
|           101 |       203 |
+---------------+-----------+
```

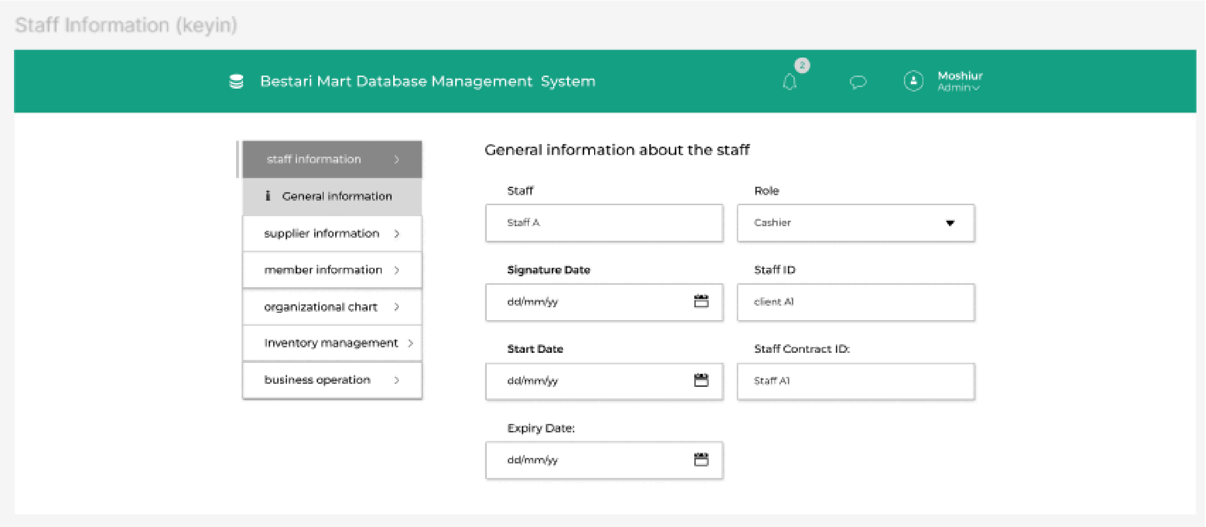
Diagram 6.2.19: Output for BusinessOperationDetails Table

7.0 Interfaces

7.1 Staff Login


Staff Information (keyin)
Bestari Mart Database Management System
Username
Password
Login

7.2 Staff Information



General information about the staff

Staff: Staff A

Role: Cashier ▾

Signature Date: dd/mm/yy

Staff ID: client A1

Start Date: dd/mm/yy

Staff Contract ID: Staff A1

Expiry Date: dd/mm/yy

7.3 Supplier List

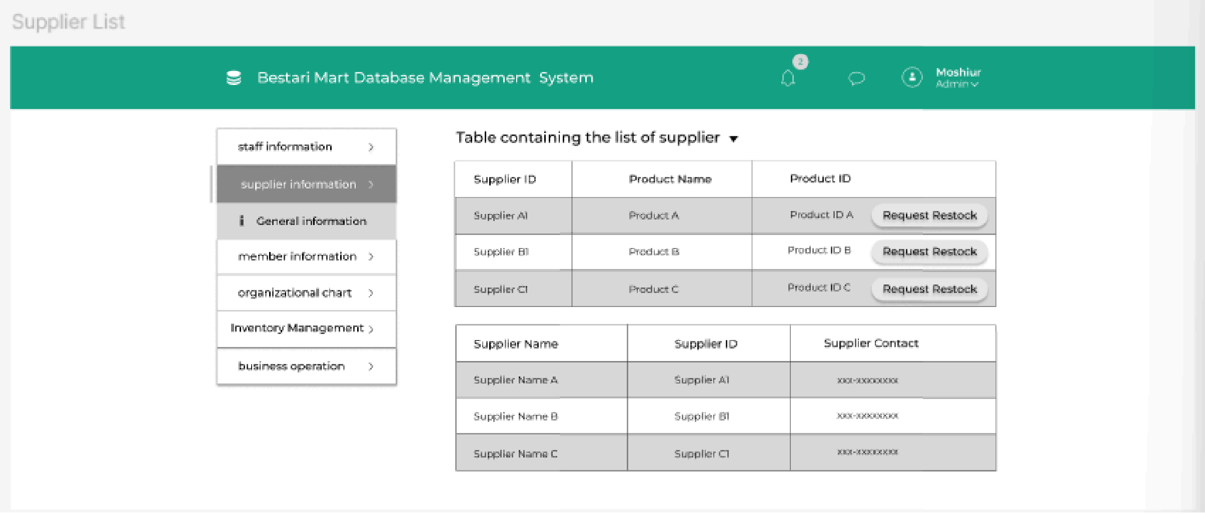


Table containing the list of supplier ▾

Supplier ID	Product Name	Product ID
Supplier A1	Product A	Product ID A Request Restock
Supplier B1	Product B	Product ID B Request Restock
Supplier C1	Product C	Product ID C Request Restock

Supplier Name	Supplier ID	Supplier Contact
Supplier Name A	Supplier A1	xxx-xxxxxxxx
Supplier Name B	Supplier B1	xxx-xxxxxxxx
Supplier Name C	Supplier C1	xxx-xxxxxxxx

7.4 Member List

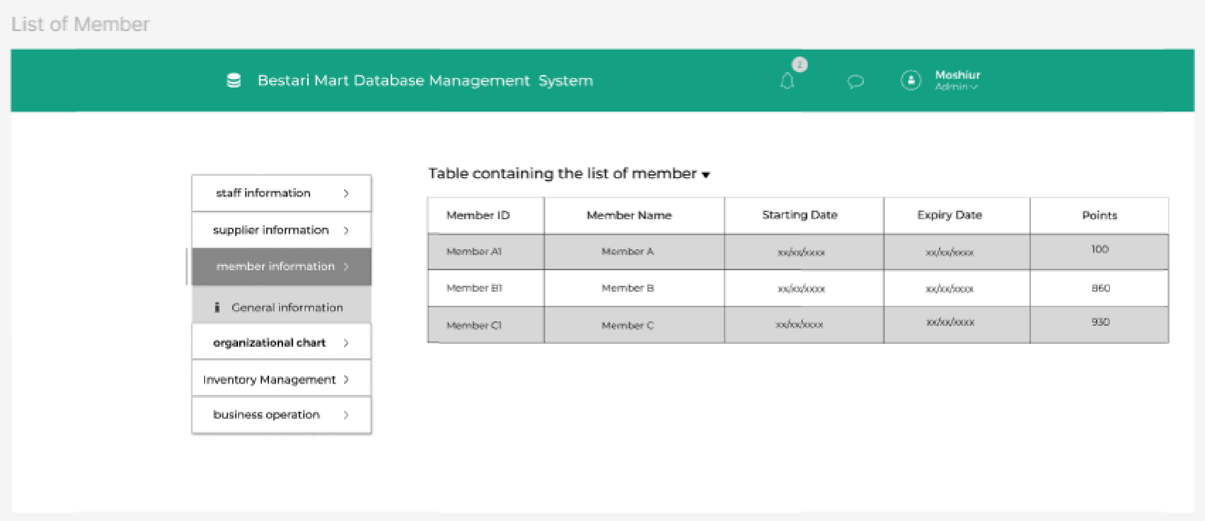
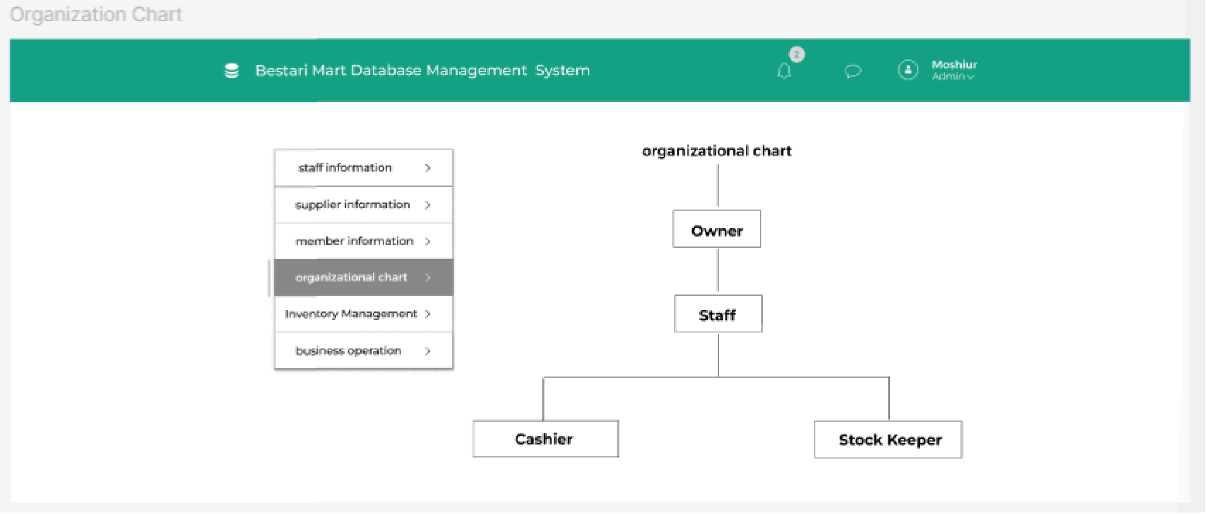


Table containing the list of member ▾

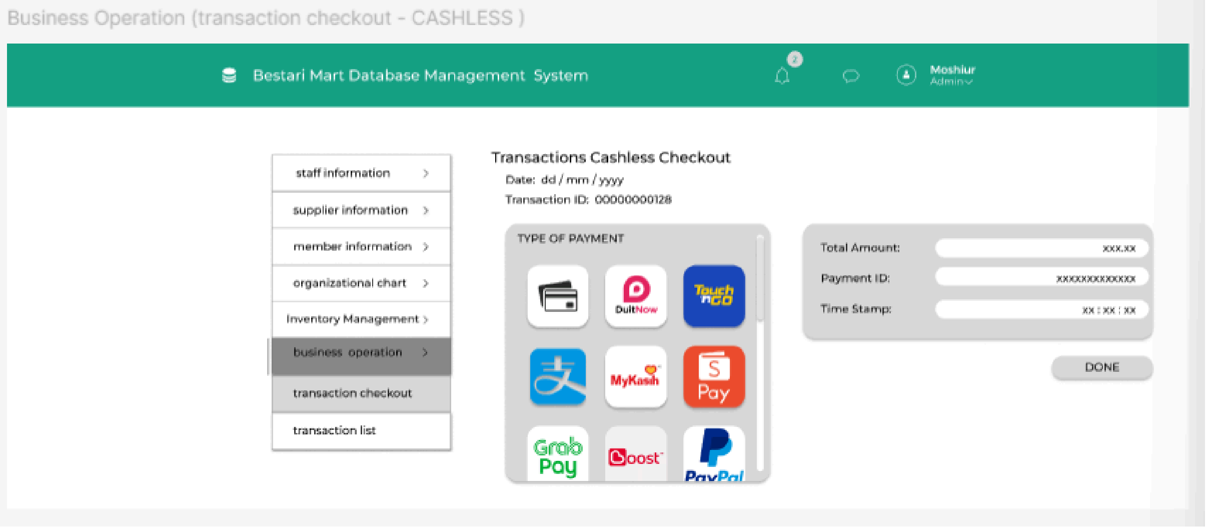
Member ID	Member Name	Starting Date	Expiry Date	Points
Member A1	Member A	xx/xx/xxxx	xx/xx/xxxx	100
Member B1	Member B	xx/xx/xxxx	xx/xx/xxxx	860
Member C1	Member C	xx/xx/xxxx	xx/xx/xxxx	930

7.5 Organization Chart



7.6 Business Operation

7.6.1 Transaction Checkout - Cashless



Transactions Cashless Checkout

Date: dd / mm / yyyy

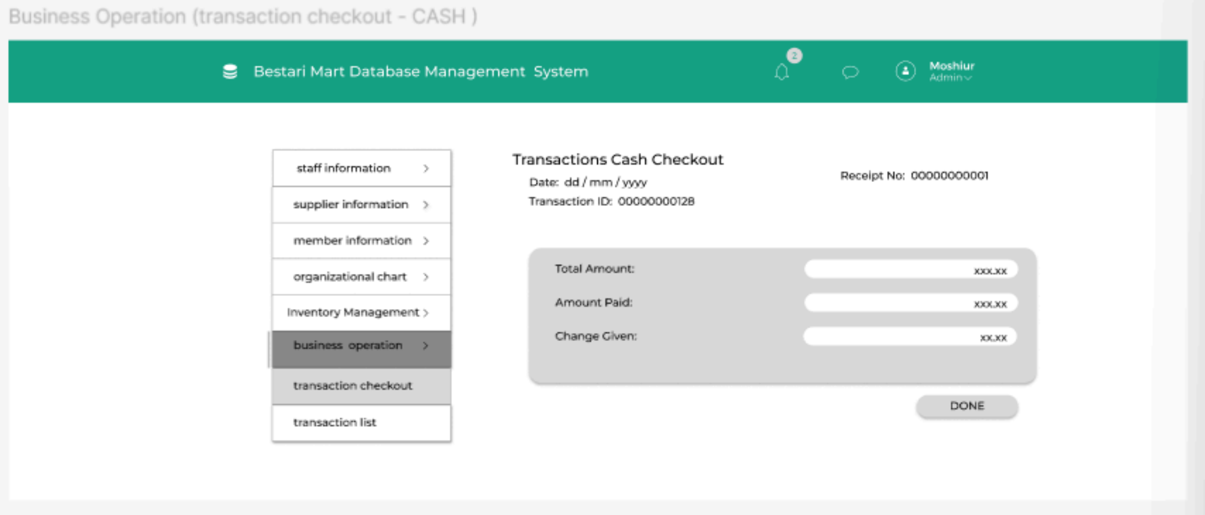
Transaction ID: 00000000128

Total Amount: xxx.xx

Payment ID: xxxxxxxxxxxxx

Time Stamp: xx : xx : xx

7.6.2 Transaction Checkout - Cash



Transactions Cash Checkout

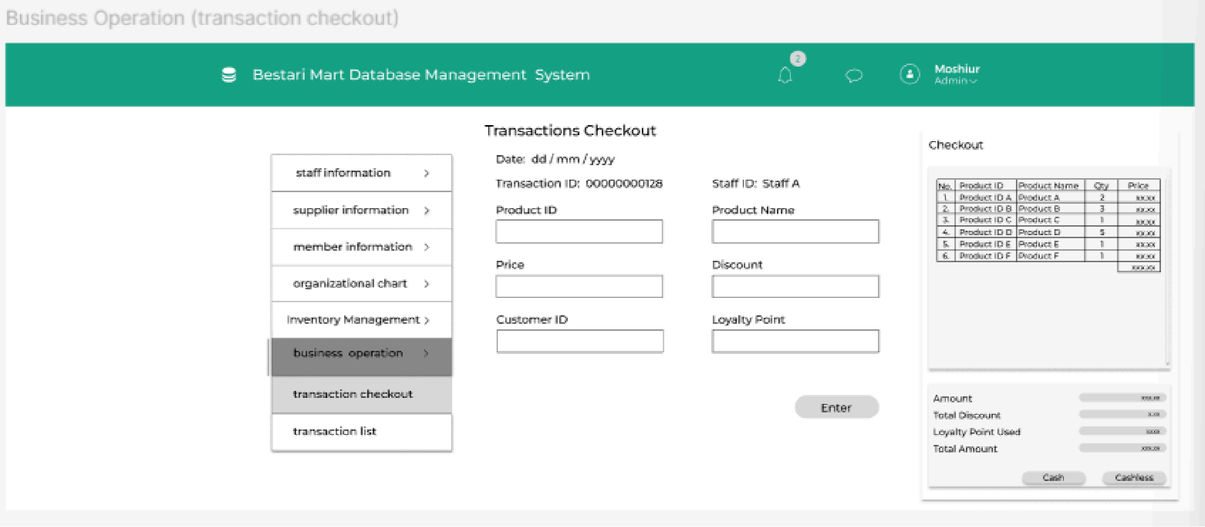
Date: dd / mm / yyyy

Transaction ID: 00000000128

Receipt No: 00000000001

Total Amount:	xxx.xx
Amount Paid:	xxx.xx
Change Given:	xx.xx

7.6.3 Items Checkout



Transactions Checkout

Date: dd / mm / yyyy

Transaction ID: 00000000128

Staff ID: Staff A

Product ID

Product Name

Price

Discount

Customer ID

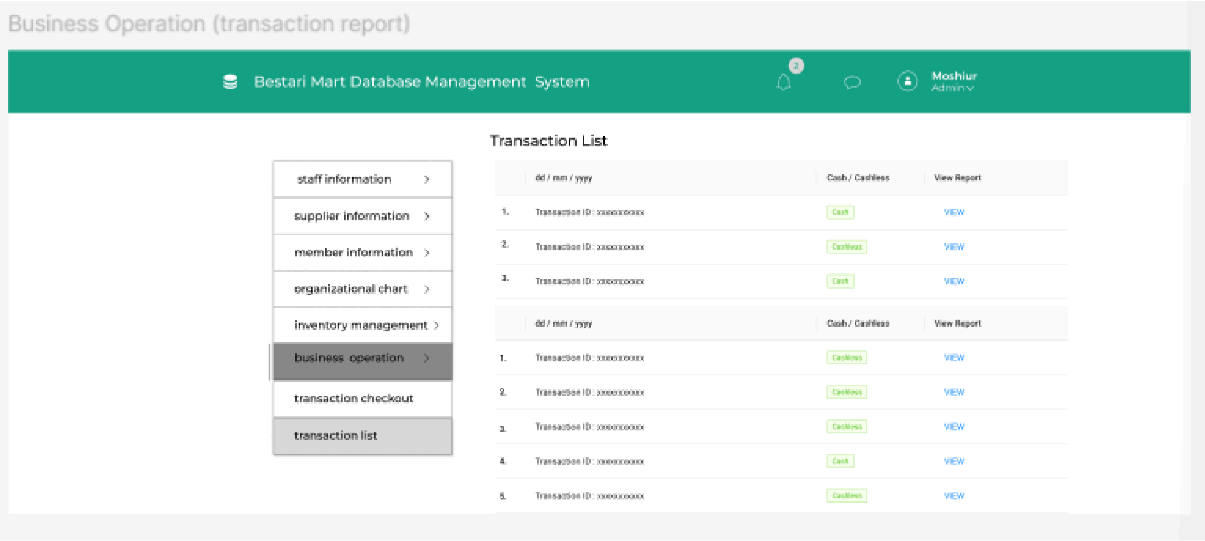
Loyalty Point

Checkout

No.	Product ID	Product Name	Qty	Price
1.	Product ID A	Product A	2	xx.xx
2.	Product ID B	Product B	3	xx.xx
3.	Product ID C	Product C	1	xx.xx
4.	Product ID D	Product D	5	xx.xx
5.	Product ID E	Product E	1	xx.xx
6.	Product ID F	Product F	1	xx.xx
				xxx.xx

Amount	xxx.xx
Total Discount	x.xx
Loyalty Point Used	xxxx
Total Amount	xxx.xx

7.6.4 Transaction List

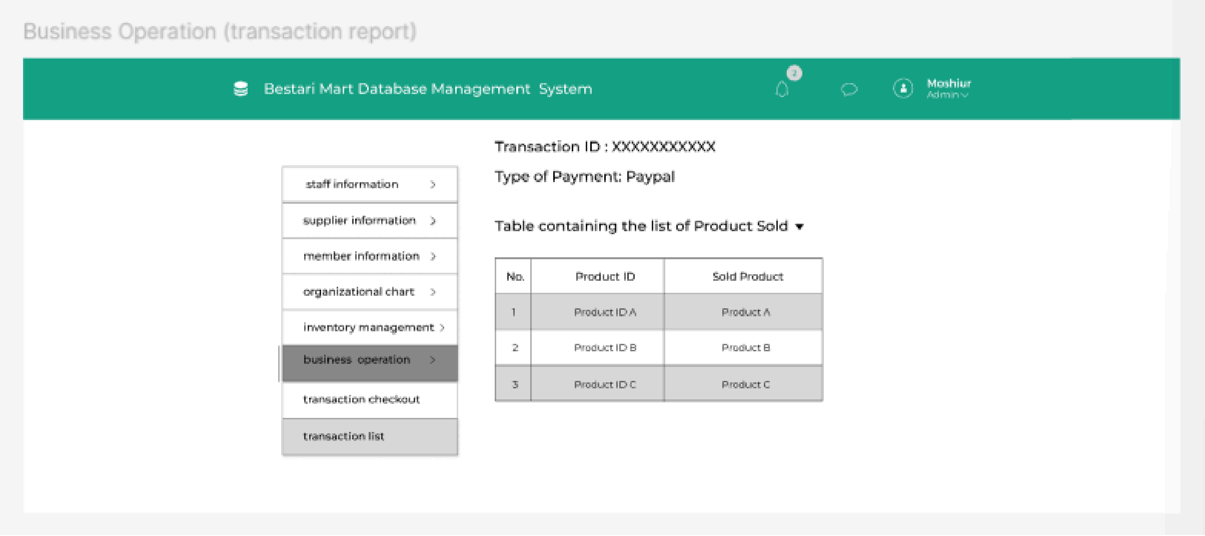


Transaction List

	dd / mm / yyyy	Cash / Cashless	View Report
1.	Transaction ID : xxxxxxxxxx	Cash	VIEW
2.	Transaction ID : xxxxxxxxxx	Cashless	VIEW
3.	Transaction ID : xxxxxxxxxx	Cash	VIEW

	dd / mm / yyyy	Cash / Cashless	View Report
1.	Transaction ID : xxxxxxxxxx	Cashless	VIEW
2.	Transaction ID : xxxxxxxxxx	Cashless	VIEW
3.	Transaction ID : xxxxxxxxxx	Cashless	VIEW
4.	Transaction ID : xxxxxxxxxx	Cash	VIEW
5.	Transaction ID : xxxxxxxxxx	Cashless	VIEW

7.6.5 Transaction Details



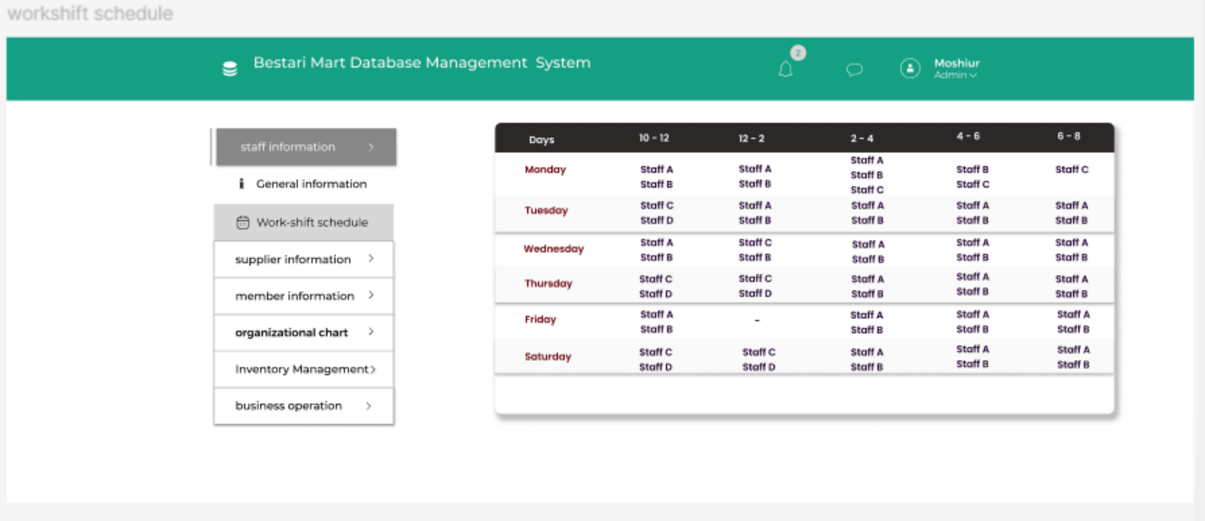
Transaction ID : XXXXXXXXXXX

Type of Payment: Paypal

Table containing the list of Product Sold ▾

No.	Product ID	Sold Product
1	Product ID A	Product A
2	Product ID B	Product B
3	Product ID C	Product C

7.7 WorkShift Schedule

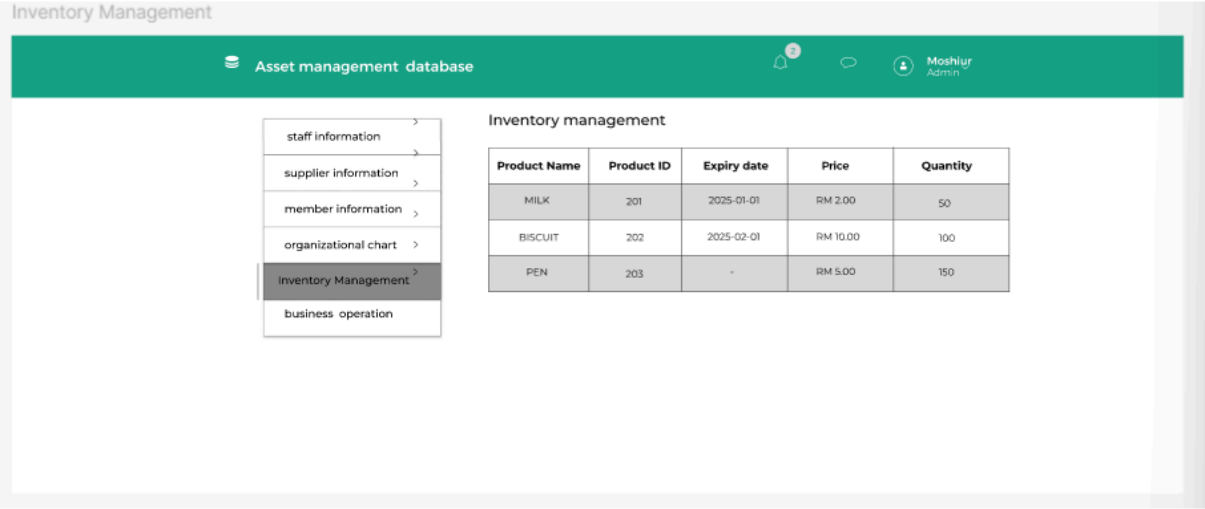


workshift schedule

Bestari Mart Database Management System

Days	10 - 12	12 - 2	2 - 4	4 - 6	6 - 8
Monday	Staff A Staff B	Staff A Staff B	Staff A Staff B Staff C	Staff B Staff C	Staff C
Tuesday	Staff C Staff D	Staff A Staff B	Staff A Staff B	Staff A Staff B	Staff A Staff B
Wednesday	Staff A Staff B	Staff C Staff B	Staff A Staff B	Staff A Staff B	Staff A Staff B
Thursday	Staff C Staff D	Staff C Staff D	Staff A Staff B	Staff A Staff B	Staff A Staff B
Friday	Staff A Staff B	-	Staff A Staff B	Staff A Staff B	Staff A Staff B
Saturday	Staff C Staff D	Staff C Staff D	Staff A Staff B	Staff A Staff B	Staff A Staff B

7.8 Inventory Management



Inventory Management

Asset management database

Inventory management

Product Name	Product ID	Expiry date	Price	Quantity
MILK	201	2025-01-01	RM 2.00	50
BISCUIT	202	2025-02-01	RM 10.00	100
PEN	203	-	RM 5.00	150

8.0 Summary

During this phase, we focused on transforming our Conceptual Entity Relationship Diagram (ERD) into a Logical ERD, adapting it to meet the practical needs of managing a minimart efficiently. This meant refining the relationships between entities, removing any non-relational features, and ensuring that the database design followed proper relational principles.

We worked on converting key entities like staff, customers, transactions, suppliers, and products into well-structured tables, each with clearly defined attributes and primary keys. To make the database more efficient, we applied normalization to remove redundancies and dependencies, ensuring data accuracy and reliability. This step helped us maintain the integrity of important information such as inventory details, customer purchases, and supplier records.

The final Logical ERD we created is a clear representation of the database structure we designed. It shows how all the entities are connected and provides a solid foundation for managing data efficiently. At the same time, we updated the data dictionary to reflect the changes and to ensure everything stayed consistent and accurate.

Overall, this phase taught us how to turn conceptual ideas into a well-organized database design. By creating a reliable and efficient database system, we're confident that our solution will help BESTARI Minimart improve operations, manage inventory better, handle transactions more smoothly, and deliver a better experience for customers. This project has been a valuable learning experience, and we're excited about how this system can support the minimart's future growth.