"E-Commerce Online Order Management"

An SQL Project Report is submitted

In accordance with requirement of degree of

BACHELOR OF TECHNOLOGY

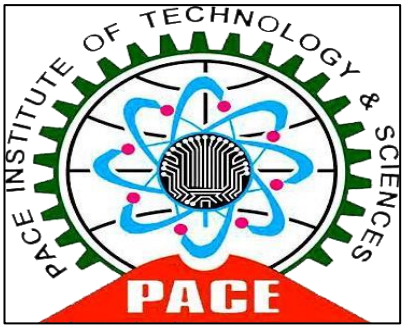
In

ELECTRICAL AND ELECTRONICS ENGINEERING

Submitted by

SYED SADHIK

22KQ1A0263



DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

PACE INSTITUTE OF TECHNOLOGY AND SCIENCES (AUTONOMOUS)

(Affiliated to Jawaharlal Nehru Technological University Kakinada, Kakinada &

Accredited by NAAC 'A' grade, An ISO 9001-2015 Certified Institution)

NH-16, Valluru Post, Prakasam District, A.P – 523272.

(2022-2026)

Abstract

This document presents the design of a robust **E-Commerce Order Management System (OMS)** implemented using a relational database schema. The system models a comprehensive **order-to-fulfillment workflow**, from capturing purchaser details through managing inventory, processing payments, and coordinating shipping logistics. Key entities include Purchaser, OnlineOrders, OrderItems, Products, Stockpile, Shipping_Address, Payments, and associated service components like Feedback, Add_to_wishlist, and After_Sales_Care. Through carefully defined relationships and normalization, the schema ensures data integrity and supports seamless operation.

The structured approach enables:

- **Accurate user profiling and payment processing**, connecting personal, credit, and debit card data to customer records.
- **Inventory tracking across multiple stock locations** and real-time product availability.
- **Clear processing paths for orders**, including item-level breakdowns and linking to shipping details.
- **Post-purchase engagement**, including feedback loops, wishlist maintenance, and vendor or service oversight.

This OMS blueprint aligns with industry-standard objectives—efficient order handling, fulfillment accuracy, and elevated customer experience—providing a scalable foundation for modern e-commerce platforms. It bridges operational reliability and analytical potential, making it a powerful backend for real-time commerce workflows.

Introduction

An **Order Management System (OMS)** is a vital software platform for e-commerce businesses, responsible for managing the complete lifecycle of a customer order—from entry through processing, inventory handling, fulfillment, and delivery to post-sales support. It ensures seamless order flow, visibility, and operational efficiency

This SQL schema is architected as a **relational operational database**—optimized for real-time transaction processing and data integrity (also known as an OLTP system)—making it well-suited to support an OMS.

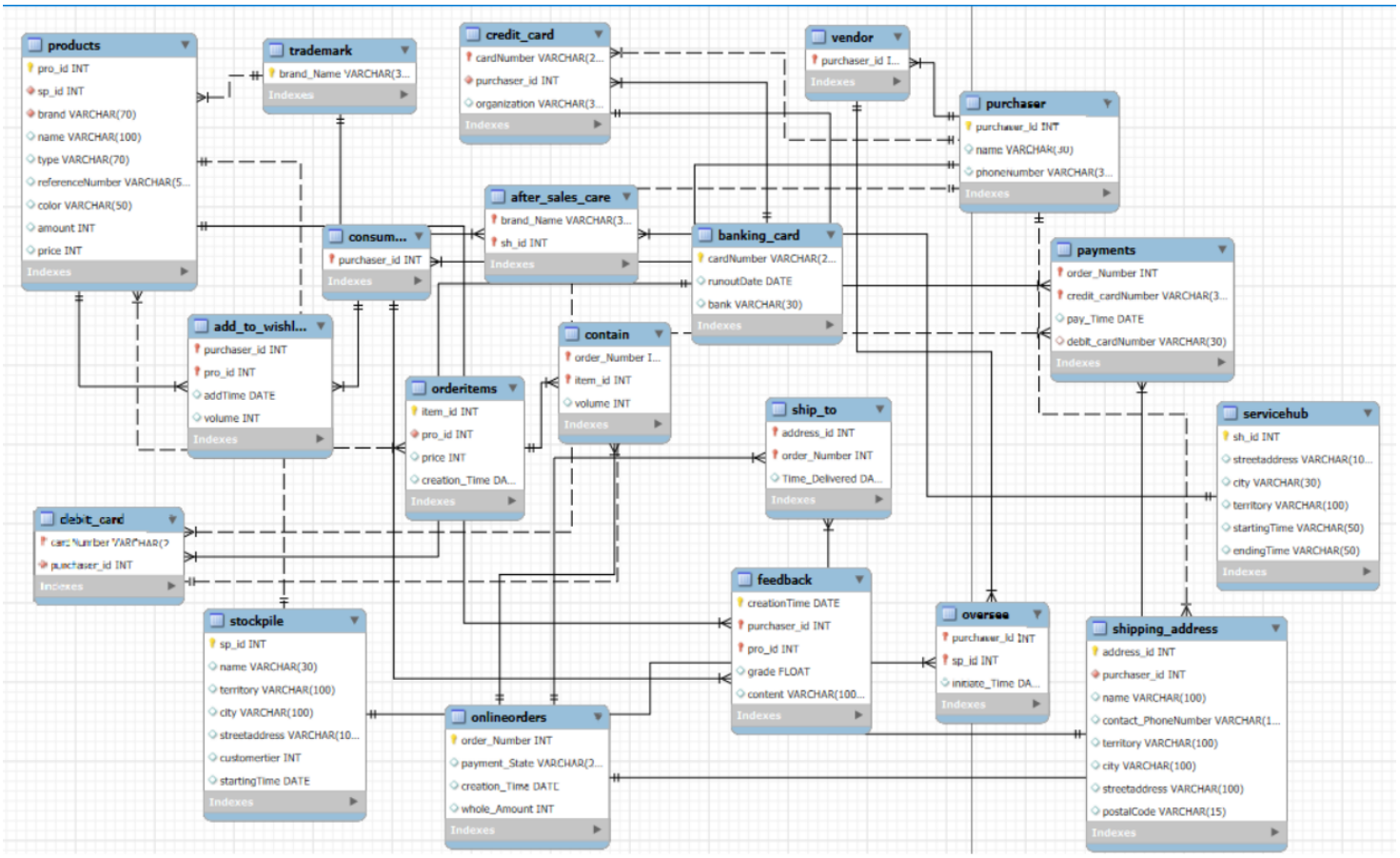
Key Functional Areas Represented in Schema

- **Customer and Payment Handling**
  The tables **Purchaser, Consumer, Vendor,** alongside **Banking_Card, Credit_Card,** and **Debit_Card**, capture user data and link payments to accountable entities—a foundation for order entry and payment tracking—core functionalities of an OMS.
- **Order Capture and Processing**
  Entities such as **Onlineorders, OrderItems,** and Contain facilitate the detailed breakdown of orders and their processing, aligning with OMS needs for order entry and verification.
- **Inventory and Fulfillment Support**
  Inclusion of **Products, Stockpile, Shipping_Address,** and **Ship_To** supports inventory tracking across locations and logical order fulfillment routes—key in multi-warehouse operations supported by modern OMS platforms.
- **Customer Experience and Service Integration**
  Feedback mechanisms (**Feedback, Add_to_wishlist**) and after-sales management (**After_Sales_Care, Oversee**) enable customer engagement, satisfaction tracking, and service integration—enhancing the functional scope of your OMS towards post-purchase workflows.

Methodology & Design Approach

- **Requirement Analysis**—Identified core entities and user roles (Consumer, Vendor).
- **Conceptual Modeling**—Created an ER diagram mapping relationships (e.g., Purchaser → Consumer/Vendor, Orders, Payments, Shipping).
- **Logical Schema Design**—Defined tables with PKs, FKs, and verifying normalization.
- **Implementation**—Used SQL DDL scripts to create and test database schema.

ER Diagram:-



E-Commerce Order Management System (OMS)

- **Project Purpose**—Designing a comprehensive relational database for an e-commerce OMS.
- **Key Components**—Entities like Purchaser, Products, Orders, Payments, Shipping, Feedback, Vendors, and Service Hubs.

CREATING THE DATABASE IN MYSQL:-

```
1   CREATE DATABASE e_Commerce;
2
3
4
5
```

Output

Action Output

#	Time	Action	Message
1	20:15:39	CREATE DATABASE e_Commerce	1 row(s) affected

CREATING TABLES:-

1.Purchaser:

```
1   CREATE DATABASE e_Commerce;
2 • USE e_Commerce;
3 • CREATE TABLE Purchaser
4   (
5       purchaser_id INT NOT NULL
6       ,name VARCHAR(30)
7       ,phoneNumber VARCHAR(30)
8       ,PRIMARY KEY(purchaser_id)
9   );
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	20:15:39	CREATE DATABASE e_Commerce	1 row(s) affected	0.016 sec
2	20:25:43	USE e_Commerce	0 row(s) affected	0.000 sec
3	20:26:55	CREATE TABLE Purchaser (purchaser_id INT NOT NULL ,name VARCHAR(30) ,phoneNumber VARCHAR(...	0 row(s) affected	0.047 sec

2.Consumer:

```
11
12 •  CREATE TABLE Consumer
13    (
14        purchaser_id INT NOT NULL
15        ,PRIMARY KEY(purchaser_id)
16        ,FOREIGN KEY(purchaser_id) REFERENCES purchaser(purchaser_id)
17    );
18
```

3.Vendor:

```
19 •  CREATE TABLE Vendor
20    (
21        purchaser_id INT NOT NULL
22        ,PRIMARY KEY(purchaser_id)
23        ,FOREIGN KEY(purchaser_id) REFERENCES purchaser(purchaser_id)
24    );
25
26
```

Output

Action Output

#	Time	Action	Message
1	20:54:02	CREATE TABLE Vendor (purchaser_id INT NOT NULL ,PRIMARY KEY(purchaser_id) ,FOREIGN KEY(p...	0 row(s) affected

4.Banking_Card:-

```
CREATE TABLE Banking_Card
(
    cardNumber VARCHAR(20) NOT NULL
    ,runoutDate DATE
    ,bank VARCHAR(30)
    ,PRIMARY KEY(cardNumber)
);
```

```
Output
Action Output
#  Time      Action                                                                                                  Message
1  20:57:10  CREATE TABLE Banking_Card (  cardNumber VARCHAR(20) NOT NULL  ,runoutDate DATE  ,bank VAR...  0 row(s) affected
```

5. Credit_Card:-

```
CREATE TABLE Credit_Card
(   cardNumber VARCHAR(20) NOT NULL
    ,purchaser_id INT NOT NULL
    ,organization VARCHAR(30)
    ,PRIMARY KEY(cardNumber)
    ,FOREIGN KEY(cardNumber) REFERENCES Banking_Card(cardNumber)
    ,FOREIGN KEY(purchaser_id) REFERENCES purchaser(purchaser_id)
);
```

```
Output
Action Output
#  Time      Action                                                                                                  Message
1  21:00:17  CREATE TABLE Credit_Card (  cardNumber VARCHAR(20) NOT NULL  ,purchaser_id INT NOT NULL  ,org...  0 row(s) affected
```

6.Debit_Card:-

```
CREATE TABLE Debit_Card
(
    cardNumber VARCHAR(20) NOT NULL
    ,purchaser_id INT NOT NULL
    ,PRIMARY KEY(cardNumber)
    ,FOREIGN KEY(cardNumber) REFERENCES Banking_Card(cardNumber)
    ,FOREIGN KEY(purchaser_id) REFERENCES purchaser(purchaser_id)
);
```

```
Output
Action Output
#  Time      Action                                                                                                  Message
1  21:02:34  CREATE TABLE Debit_Card (  cardNumber VARCHAR(20) NOT NULL  ,purchaser_id INT NOT NULL  ,P...  0 row(s) affected
```

7.Shipping_Address:-

```
CREATE TABLE Shipping_Address (address_id INT NOT NULL,purchaser_id INT NOT NULL,name VARCHAR(100)
    ,contact_PhoneNumber VARCHAR(15) ,territory VARCHAR(100),city VARCHAR(100)
    ,streetaddress VARCHAR(100),postalCode VARCHAR(15)
    ,PRIMARY KEY(address_id)
    ,FOREIGN KEY(purchaser_id) REFERENCES Purchaser(purchaser_id)
);
```

```
Output
Action Output
#  Time      Action                                                                                                  Message
1  21:06:25  CREATE TABLE Shipping_Address (address_id INT NOT NULL,purchaser_id INT NOT NULL,name VARCHAR(...  0 row(s) affected
```

8. Stockpile:-

```
CREATE TABLE Stockpile (sp_id INT NOT NULL,name VARCHAR(30),territory VARCHAR(100),
    city VARCHAR(100),streetaddress VARCHAR(100),customertier INT
    ,startingTime DATE
    ,PRIMARY KEY(sp_id)
);
```

```
Output
Action Output
#  Time      Action                                                                                                  Message
1  21:09:28  CREATE TABLE Stockpile (sp_id INT NOT NULL,name VARCHAR(30),territory VARCHAR(100),  city VARCHA...  0 row(s) affected
```

9. Trademark:-

```
CREATE TABLE Trademark
(
    brand_Name VARCHAR(30) NOT NULL
    ,PRIMARY KEY (brand_Name)
);
```

```
Output
Action Output
#  Time      Action                                                                                                  Message
1  21:13:22  CREATE TABLE Trademark (  brand_Name VARCHAR(30) NOT NULL  ,PRIMARY KEY (brand_Name) )  0 row(s) affected
```

10. Products:-

```
CREATE TABLE Products (pro_id INT NOT NULL,sp_id INT NOT NULL,brand VARCHAR(70) NOT NULL,name VARCHAR(100)
    ,type VARCHAR(70),referenceNumber VARCHAR(50),color VARCHAR(50),amount INT,price INT
    ,PRIMARY KEY(pro_id)
    ,FOREIGN KEY(sp_id) REFERENCES Stockpile(sp_id)
    ,FOREIGN KEY(brand) REFERENCES Trademark(brand_Name)
);
```

Output

Action Output

#	Time	Action	Message
1	21:19:37	CREATE TABLE Products (pro_id INT NOT NULL,sp_id INT NOT NULL,brand VARCHAR(70) NOT NULL,name ...	0 row(s) affected

11. OrderItems:-

```
CREATE TABLE OrderItems(item_id INT NOT NULL,pro_id INT NOT NULL
    ,price INT,creation_Time DATE
    ,PRIMARY KEY(item_id)
    ,FOREIGN KEY(pro_id) REFERENCES Products(pro_id)
);
```

Output

Action Output

#	Time	Action	Message
1	21:22:02	CREATE TABLE OrderItems(item_id INT NOT NULL,pro_id INT NOT NULL ,price INT,creation_Time DATE ...	0 row(s) affected

12. Onlineorders:-

```
CREATE TABLE Onlineorders (order_Number INT NOT NULL,payment_State VARCHAR(20)
    ,creation_Time DATE,whole_Amount INT
    ,PRIMARY KEY (order_Number)
);
```

Output

Action Output

#	Time	Action	Message
1	21:24:47	CREATE TABLE Onlineorders (order_Number INT NOT NULL,payment_State VARCHAR(20) ,creation_Time D...	0 row(s) affected

13. Servicehub:-

```
CREATE TABLE Servicehub (sh_id INT NOT NULL,streetaddress VARCHAR(100),city VARCHAR(30)
    ,territory VARCHAR(100),startingTime VARCHAR(50),endingTime VARCHAR(50)
    ,PRIMARY KEY(sh_id)
);
```

Output

Action Output

#	Time	Action	Message
1	21:27:08	CREATE TABLE Servicehub (sh_id INT NOT NULL,streetaddress VARCHAR(100),city VARCHAR(30) ,territory...	0 row(s) affected

14. Feedback:-

```
CREATE TABLE Feedback (creationTime DATE NOT NULL,purchaser_id INT NOT NULL
    ,pro_id INT NOT NULL,grade FLOAT,content VARCHAR(1000)
    ,PRIMARY KEY(creationTime,purchaser_id,pro_id)
    ,FOREIGN KEY(purchaser_id) REFERENCES Consumer(purchaser_id)
    ,FOREIGN KEY(pro_id) REFERENCES Products(pro_id)
);
```

Output

Action Output

#	Time	Action	Message
1	21:29:05	CREATE TABLE Feedback (creationTime DATE NOT NULL,purchaser_id INT NOT NULL ,pro_id INT NOT N...	0 row(s) affected

15. Add_to_wishlist:-

```
CREATE TABLE Add_to_wishlist (purchaser_id INT NOT NULL,pro_id INT NOT NULL,addTime DATE,volume INT
    ,PRIMARY KEY (purchaser_id,pro_id)
    ,FOREIGN KEY(purchaser_id) REFERENCES Consumer(purchaser_id)
    ,FOREIGN KEY(pro_id) REFERENCES Products(pro_id)
);
```

Output

Action Output

#	Time	Action	Message
1	21:31:19	CREATE TABLE Add_to_wishlist (purchaser_id INT NOT NULL,pro_id INT NOT NULL,addTime DATE,volume I...	0 row(s) affected

16. Contain:-

```
CREATE TABLE Contain (order_Number INT NOT NULL,item_id INT NOT NULL,volume INT
    ,PRIMARY KEY (order_Number,item_id)
    ,FOREIGN KEY(order_Number) REFERENCES Onlineorders(order_Number)
    ,FOREIGN KEY(item_id) REFERENCES OrderItems(item_id)
);
```

Output

Action Output

#	Time	Action	Message
1	21:34:40	CREATE TABLE Contain (order_Number INT NOT NULL,item_id INT NOT NULL,volume INT ,PRIMARY KEY ...	0 row(s) affected

17. Payments:-

```
CREATE TABLE Payments (order_Number INT NOT NULL,credit_cardNumber VARCHAR(30) NOT NULL,pay_Time DATE
    ,PRIMARY KEY(order_Number,credit_cardNumber)
    ,FOREIGN KEY(order_Number) REFERENCES Onlineorders(order_Number)
    ,FOREIGN KEY(credit_cardNumber) REFERENCES Credit_Card(cardNumber)
);
```

Output

Action Output

#	Time	Action	Message
1	21:37:18	CREATE TABLE Payments (order_Number INT NOT NULL,credit_cardNumber VARCHAR(30) NOT NULL,pay_T...	0 row(s) affected

18. Ship_To:-

```
CREATE TABLE Ship_To (address_id INT NOT NULL,order_Number INT NOT NULL
    ,Time_Delivered DATE
    ,PRIMARY KEY(address_id,order_Number)
    ,FOREIGN KEY(address_id) REFERENCES Shipping_Address(address_id)
    ,FOREIGN KEY(order_Number) REFERENCES Onlineorders(order_Number)
);
```

Output

Action Output

#	Time	Action	Message
1	21:39:19	CREATE TABLE Ship_To (address_id INT NOT NULL,order_Number INT NOT NULL ,Time_Delivered DATE ...	0 row(s) affected

19. Oversee:-

```
CREATE TABLE Oversee (purchaser_id INT NOT NULL,sp_id INT NOT NULL
    ,initiate_Time DATE
    ,PRIMARY KEY(purchaser_id,sp_id)
    ,FOREIGN KEY(purchaser_id) REFERENCES Vendor(purchaser_id)
    ,FOREIGN KEY(sp_id) REFERENCES Stockpile(sp_id)
);
```

Output

Action Output

#	Time	Action	Message
1	21:42:01	CREATE TABLE Oversee (purchaser_id INT NOT NULL,sp_id INT NOT NULL ,initiate_Time DATE ,PRIMA...	0 row(s) affected

20. After_Sales_Care:-

```
CREATE TABLE After_Sales_Care (brand_Name VARCHAR(30) NOT NULL,sh_id INT NOT NULL
    ,PRIMARY KEY(brand_Name, sh_id)
    ,FOREIGN KEY(brand_Name) REFERENCES Trademark(brand_Name)
    ,FOREIGN KEY(sh_id) REFERENCES Servicehub(sh_id)
    );
```

Output

Action Output

#	Time	Action	Message
1	21:44:01	CREATE TABLE After_Sales_Care (brand_Name VARCHAR(30) NOT NULL,sh_id INT NOT NULL ,PRIMARY...	0 row(s) affected

Inserting Data:-

1. Purchaser:-

```
INSERT INTO Purchaser (purchaser_id, name, phoneNumber) VALUES (1, 'Aarav Sharma', '+91-987-543-10'),(2, 'Priya Patel', '+91-912-456-80'),
(3, 'Rahul Singh', '+91-998-776-55'),(4, 'Ananya Mukherjee', '+91-923-567-91'),
(5, 'Vikram Nair', '+91-935-787-02'),(6, 'Deepika Rao', '+91-945-789-13'),
(7, 'Karan Mehta', '+91-956-890-24'),(8, 'Sneha Verma', '+91-967-901-35'),
(9, 'Rohit Desai', '+91-978-012-46'),(10, 'Isha Gupta', '+91-989-123-57'),
(11, 'Aditya Joshi', '+91-990-123-56'),(12, 'Neha Kapoor', '+91-991-234-67'),
(13, 'Sanjay Kumar', '+91-992-345-78'),(14, 'Vandana Iyer', '+91-993-456-89'),
(15, 'Manish Tiwari', '+91-994-567-90'),(16, 'Leela Menon', '+91-995-578-01'),
(17, 'Arjun Reddy', '+91-996-789-12'),(18, 'Smita Singh', '+91-997-890-23'),
(19, 'Gaurav Jain', '+91-998-901-34'),(20, 'Anita Chawla', '+91-999-012-45'),
(21, 'Ritesh Bhatia', '+91-989-112-33'),(22, 'Swathi Iyer', '+91-972-723-44'),
(23, 'Pradeep Das', '+91-967-334-55'),(24, 'Meher Kaur', '+91-956-445-66'),(25, 'Farhan Khan', '+91-945-556-77');
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:13:40	INSERT INTO Purchaser (purchaser_id, name, phoneNumber) VALUES (1, 'Aarav Sharma', '+91-987-543-10'),(2, ...	25 row(s) affected Records: 25 Duplicates: 0 Warnings: 0	0.016 sec

2. Consumer:-

```
INSERT INTO Consumer (purchaser_id) VALUES (1);
INSERT INTO Consumer (purchaser_id) VALUES (2);
INSERT INTO Consumer (purchaser_id) VALUES (3);
INSERT INTO Consumer (purchaser_id) VALUES (4);
INSERT INTO Consumer (purchaser_id) VALUES (5);
INSERT INTO Consumer (purchaser_id) VALUES (6);
INSERT INTO Consumer (purchaser_id) VALUES (7);
INSERT INTO Consumer (purchaser_id) VALUES (8);
INSERT INTO Consumer (purchaser_id) VALUES (9);
INSERT INTO Consumer (purchaser_id) VALUES (10);
INSERT INTO Consumer (purchaser_id) VALUES (11);
INSERT INTO Consumer (purchaser_id) VALUES (12);
INSERT INTO Consumer (purchaser_id) VALUES (13);
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
5	22:16:45	INSERT INTO Consumer (purchaser_id) VALUES (5)	1 row(s) affected	0.000 sec
6	22:16:45	INSERT INTO Consumer (purchaser_id) VALUES (6)	1 row(s) affected	0.000 sec
7	22:16:45	INSERT INTO Consumer (purchaser_id) VALUES (7)	1 row(s) affected	0.015 sec
8	22:16:45	INSERT INTO Consumer (purchaser_id) VALUES (8)	1 row(s) affected	0.000 sec
9	22:16:45	INSERT INTO Consumer (purchaser_id) VALUES (9)	1 row(s) affected	0.016 sec
10	22:16:45	INSERT INTO Consumer (purchaser_id) VALUES (10)	1 row(s) affected	0.000 sec
11	22:16:45	INSERT INTO Consumer (purchaser_id) VALUES (11)	1 row(s) affected	0.000 sec
12	22:16:45	INSERT INTO Consumer (purchaser_id) VALUES (12)	1 row(s) affected	0.000 sec
13	22:16:45	INSERT INTO Consumer (purchaser_id) VALUES (13)	1 row(s) affected	0.016 sec

3.Vendor:-

```
INSERT INTO Vendor (purchaser_id) VALUES (14);
INSERT INTO Vendor (purchaser_id) VALUES (15);
INSERT INTO Vendor (purchaser_id) VALUES (16);
INSERT INTO Vendor (purchaser_id) VALUES (17);
INSERT INTO Vendor (purchaser_id) VALUES (18);
INSERT INTO Vendor (purchaser_id) VALUES (19);
INSERT INTO Vendor (purchaser_id) VALUES (20);
INSERT INTO Vendor (purchaser_id) VALUES (21);
INSERT INTO Vendor (purchaser_id) VALUES (22);
INSERT INTO Vendor (purchaser_id) VALUES (23);
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (14)	1 row(s) affected	0.015 sec
2	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (15)	1 row(s) affected	0.000 sec
3	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (16)	1 row(s) affected	0.000 sec
4	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (17)	1 row(s) affected	0.000 sec
5	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (18)	1 row(s) affected	0.000 sec
6	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (19)	1 row(s) affected	0.000 sec
7	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (20)	1 row(s) affected	0.000 sec
8	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (21)	1 row(s) affected	0.000 sec
9	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (22)	1 row(s) affected	0.016 sec
10	22:19:41	INSERT INTO Vendor (purchaser_id) VALUES (23)	1 row(s) affected	0.000 sec

4. Banking_Card:-

```
INSERT INTO Banking_Card (cardNumber, runoutDate, bank) VALUES
('ICICI1234567890','2027-12-31','ICICI Bank'),('HDFC0987654321','2028-05-15','HDFC Bank'),
('SBI1111222233334444','2026-11-30','State Bank of India'),('AXIS5555666677778888','2029-01-20','Axis Bank'),
('PNB9999000011112222','2027-03-10','Punjab National Bank'),('KOTA333344445555666','2028-07-25','Kotak Mahindra Bank'),
('YES7777888899990000','2026-09-05','Yes Bank'),('IDBI2222333344445555','2027-10-11','IDBI Bank'),
('BOB66667777888889999','2029-12-01','Bank of Baroda'),('CAN1111222233334445','2028-04-18','Canara Bank'),
('UCO55556666777778889','2027-08-22','UCO Bank'),('IOB9999000011112223','2029-06-30','Indian Overseas Bank');
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:26:28	INSERT INTO Banking_Card (cardNumber, runoutDate, bank) VALUES ('ICICI1234567890','2027-12-31','ICICI Bank'...	12 row(s) affected Records: 12 Duplicates: 0 Warnings: 0	0.031 sec

5. Credit_Card:-

```
INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('ICICI1234567890',1,'ICICI');
INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('HDFC0987654321',2,'HDFC');
INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('SBI1111222233334444',3,'SBI');
INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('AXIS5555666677778888',4,'AXIS');
INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('PNB9999000011112222',5,'PNB');
INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('KOTA3333444455556666',6,'Kotak');
INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('YES7777888899990000',7,'Yes');
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:38:08	INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('ICICI1234567890',1,'ICICI')	1 row(s) affected	0.015 sec
2	22:38:08	INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('HDFC0987654321',2,'HDFC')	1 row(s) affected	0.000 sec
3	22:38:08	INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('SBI1111222233334444',3,'SBI')	1 row(s) affected	0.016 sec
4	22:38:09	INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('AXIS5555666677778888',4,'AXIS')	1 row(s) affected	0.000 sec
5	22:38:09	INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('PNB9999000011112222',5,'PNB')	1 row(s) affected	0.000 sec
6	22:38:09	INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('KOTA3333444455556666',6,'Kot...	1 row(s) affected	0.000 sec
7	22:38:09	INSERT INTO Credit_Card (cardNumber, purchaser_id, organization) VALUES ('YES7777888899990000',7,'Yes')	1 row(s) affected	0.000 sec

6.Debit_Card:-

```
INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('IDBI2222333344445555',8);
INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('BOB66667777888889999',9);
INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('CAN1111222233334445',10);
INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('UCO55556666777778889',11);
INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('IOB9999000011112223',12);
INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('AXIS5555666677778888',13);
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:41:49	INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('IDBI2222333344445555',8)	1 row(s) affected	0.031 sec
2	22:41:49	INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('BOB66667777888889999',9)	1 row(s) affected	0.016 sec
3	22:41:49	INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('CAN1111222233334445',10)	1 row(s) affected	0.000 sec
4	22:41:49	INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('UCO55556666777778889',11)	1 row(s) affected	0.000 sec
5	22:41:49	INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('IOB9999000011112223',12)	1 row(s) affected	0.015 sec
6	22:41:49	INSERT INTO Debit_Card (cardNumber, purchaser_id) VALUES ('AXIS5555666677778888',13)	1 row(s) affected	0.000 sec

7. Shipping_Address:-

```
INSERT INTO Shipping_Address
(address_id, purchaser_id, name, contact_PhoneNumber, territory, city, streetaddress, postalCode) VALUES
(101,1,'Aarav Sharma','+91-9876543210','Telangana','Secunderabad','4-158/9 5th Cross Road, Sainikpuri','500094'),
(102,2,'Priya Patel','+91-9123456780','Gujarat','Ahmedabad','Plot 22, Navrangpura Main Road, Navrangpura','380009'),
(103,3,'Rahul Singh','+91-9988776655','Delhi','New Delhi','Flat 408, Tagore Road, Near Khan Market','110001'),
(104,4,'Ananya Mukherjee','+91-9234567891','Karnataka','Bangalore','32/1 2nd Main Road, 8th Block, Jayanagar','560082'),
(105,5,'Vikram Nair','+91-9345678902','Maharashtra','Mumbai','Apte Building, Dadabhai Naoroji Road, Fort','400001'),
(106,6,'Deepika Rao','+91-9456789013','West Bengal','Kolkata','15 Sudder Street, Esplanade Area','700016'),
(107,7,'Karan Mehta','+91-9567890124','Tamil Nadu','Chennai','15/A Anna Salai, Near Government Museum','600002'),
(108,8,'Sneha Verma','+91-9678901235','Maharashtra','Mumbai','Altamount Road, Near Antilia','400026'),
(109,9,'Rohit Desai','+91-9789012346','Uttar Pradesh','Lucknow','House No. 56/78, 14th A Cross, JP Nagar Phase 7','226001'),
(110,10,'Isha Gupta','+91-9890123457','Madhya Pradesh','Bhopal','Flat No. 402, Rosewood Apartments, MP Nagar','462011'),
(111,11,'Aditya Joshi','+91-9900123456','Punjab','Chandigarh','Sector 22-C, House No. 134','160022'),
(112,12,'Neha Kapoor','+91-9911234567','Rajasthan','Jaipur','E-8, Malviya Nagar, Opposite Malviya Nagar Metro Station','302017'),
(113,13,'Sanjay Kumar','+91-9922345678','Haryana','Gurgaon','A-75, Sector 14, DLF City','122001'),
(114,14,'Vandana Iyer','+91-9933456789','Kerala','Kochi','Flat No. 202, Seafront Apartments, Marine Drive','682031'),
(115,15,'Manish Tiwari','+91-9944567890','Odisha','Bhubaneswar','Plot 101, IRC Village, Nayapalli','751015'),
(116,16,'Leela Menon','+91-9955678901','Karnataka','Mysore','No. 12, Lakshmi Road, Opposite City Market','570024'),
(117,17,'Arjun Reddy','+91-9966789012','Telangana','Hyderabad','House No. 23, Road No. 12, Banjara Hills','500034'),
(118,18,'Smita Singh','+91-9977890123','Chhattisgarh','Raipur','C-45, Mowa Road, Sector 1, Shankar Nagar','492007'),
(119,19,'Gaurav Jain','+91-9988901234','Jharkhand','Ranchi','Flat 15B, Nirmala Apartment, Kanke Road','834008'),
(120,20,'Anita Chawla','+91-9990012345','Uttarakhand','Dehradun','Prime Standard Apartments, Near Clock Tower','248001'),
(121,21,'Ritesh Bhatia','+91-9891112233','Assam','Guwahati','House No. 9, Suren Das Road, Fancy Bazaar','781001'),
(122,22,'Swathi Iyer','+91-9782223344','Tamil Nadu','Coimbatore','No. 45, Race Course Road, Gandhipuram','641018'),
(123,23,'Pradeep Das','+91-9673334455','West Bengal','Howrah','Plot 56, Netaji Subhash Road, Shibpur','711102'),
(124,24,'Meher Kaur','+91-9564445566','Punjab','Amritsar','House No. 12, Ranjit Avenue, Near Golden Temple','143001'),
(125,25,'Farhan Khan','+91-9455556677','Karnataka','Hubli','1st Floor, Kumar Plaza, Airport Road','580030');
```

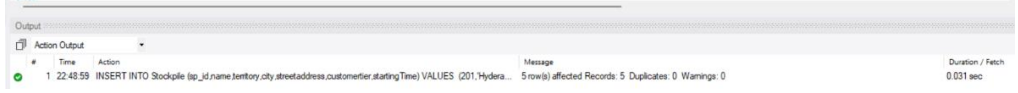
Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:45:18	INSERT INTO Shipping_Address (address_id, purchaser_id, name, contact_PhoneNumber, territory, city, streetad...	25 row(s) affected Records: 25 Duplicates: 0 Warnings: 0	0.031 sec

8. Stockpile:-

```
INSERT INTO Stockpile (sp_id,name,territory,city,streetaddress,customertier,startingTime) VALUES
(201,'Hyderabad Depot','Telangana','Hyderabad','Begumpet Airport Rd',1,'2024-01-01'),
(202,'Mumbai Hub','Maharashtra','Mumbai','Andheri East Industrial Area',2,'2024-02-15'),
(203,'Delhi Warehouse','Delhi','New Delhi','Okhla Industrial Estate',1,'2024-03-10'),
(204,'Bengaluru Centre','Karnataka','Bengaluru','Whitefield EPIP Zone',3,'2024-04-20'),
(205,'Chennai Store','Tamil Nadu','Chennai','Guindy Industrial Estate',2,'2024-05-30');
```

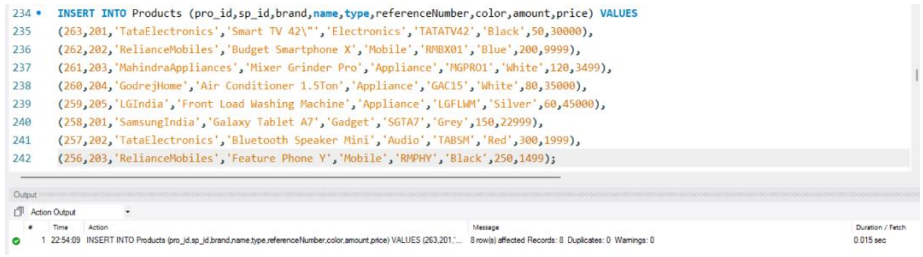


9. Trademark:-

```
INSERT INTO Trademark (brand_Name) VALUES ('TataElectronics'),
('RelianceMobiles'),('MahindraAppliances'),('GodrejHome'),('LGIndia'),('SamsungIndia');
```

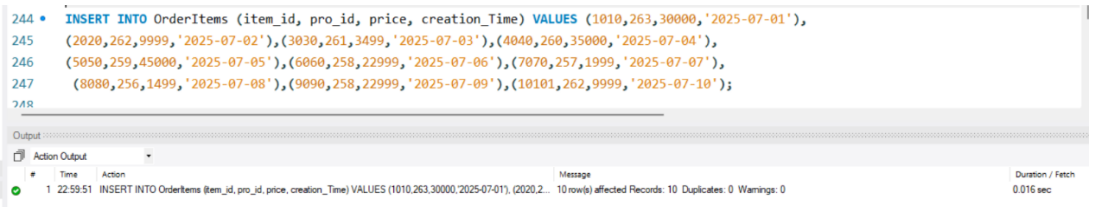
10. Products:-

```
INSERT INTO Products (pro_id,sp_id,brand,name,type,referenceNumber,color,amount,price) VALUES
(263,201,'TataElectronics','Smart TV 42\"','Electronics','TATATV42','Black',50,30000),
(262,202,'RelianceMobiles','Budget Smartphone X','Mobile','RMBX01','Blue',200,9999),
(261,203,'MahindraAppliances','Mixer Grinder Pro','Appliance','MGPRO1','White',120,3499),
(260,204,'GodrejHome','Air Conditioner 1.5Ton','Appliance','GAC15','White',80,35000),
(259,205,'LGIndia','Front Load Washing Machine','Appliance','LGFLWM','Silver',60,45000),
(258,201,'SamsungIndia','Galaxy Tablet A7','Gadget','SGTA7','Grey',150,22999),
(257,202,'TataElectronics','Bluetooth Speaker Mini','Audio','TABSM','Red',300,1999),
(256,203,'RelianceMobiles','Feature Phone Y','Mobile','RMPHY','Black',250,1499);
```



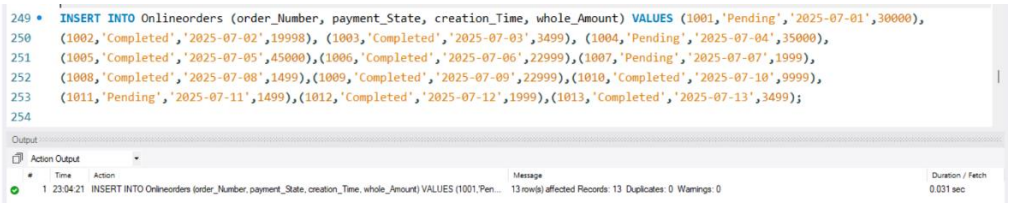
11. OrderItems:-

```
INSERT INTO OrderItems (item_id, pro_id, price, creation_Time) VALUES (1010,263,30000,'2025-07-01'),
(2020,262,9999,'2025-07-02'),(3030,261,3499,'2025-07-03'),(4040,260,35000,'2025-07-04'),
(5050,259,45000,'2025-07-05'),(6060,258,22999,'2025-07-06'),(7070,257,1999,'2025-07-07'),
(8080,256,1499,'2025-07-08'),(9090,258,22999,'2025-07-09'),(10101,262,9999,'2025-07-10');
```



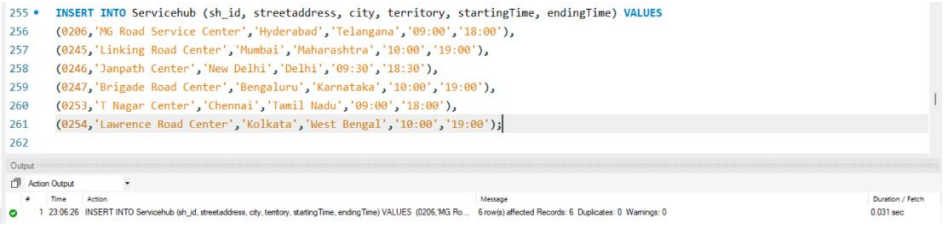
12. Onlineorders:-

```
INSERT INTO Onlineorders (order_Number, payment_State, creation_Time, whole_Amount) VALUES (1001,'Pending','2025-07-01',30000),
(1002,'Completed','2025-07-02',19998), (1003,'Completed','2025-07-03',3499), (1004,'Pending','2025-07-04',35000),
(1005,'Completed','2025-07-05',45000),(1006,'Completed','2025-07-06',22999),(1007,'Pending','2025-07-07',1999),
(1008,'Completed','2025-07-08',1499),(1009,'Completed','2025-07-09',22999),(1010,'Completed','2025-07-10',9999),
(1011,'Pending','2025-07-11',1499),(1012,'Completed','2025-07-12',1999),(1013,'Completed','2025-07-13',3499);
```



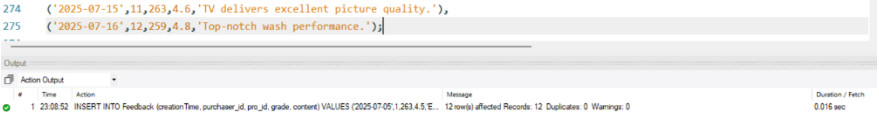
13. Servicehub:-

```
INSERT INTO Servicehub (sh_id, streetaddress, city, territory, startingTime, endingTime) VALUES
(0206,'MG Road Service Center','Hyderabad','Telangana','09:00','18:00'),
(0245,'Linking Road Center','Mumbai','Maharashtra','10:00','19:00'),
(0246,'Janpath Center','New Delhi','Delhi','09:30','18:30'),
(0247,'Brigade Road Center','Bengaluru','Karnataka','10:00','19:00'),
(0253,'T Nagar Center','Chennai','Tamil Nadu','09:00','18:00'),
(0254,'Lawrence Road Center','Kolkata','West Bengal','10:00','19:00');
```



14. Feedback:-

```
INSERT INTO Feedback (creationTime, purchaser_id, pro_id, grade, content) VALUES
('2025-07-05',1,263,4.5,'Excellent picture clarity and sound quality.'),
('2025-07-06',2,262,4.0,'Good value phone for the price.'),
('2025-07-07',3,261,4.2,'Very useful mixer grinder with long run blades.'),
('2025-07-08',4,260,3.8,'AC cools well but is a bit noisy.'),
('2025-07-09',5,259,4.7,'Washing machine is very quiet and efficient.'),
('2025-07-10',6,258,4.1,'Tablet performs smoothly, battery lasts long.'),
('2025-07-11',7,257,4.3,'Great small speaker with good bass.'),
('2025-07-12',8,256,3.9,'Basic phone, battery lasts long.'),
('2025-07-13',9,258,4.0,'Tablet is fine, display is sharp.'),
('2025-07-14',10,262,4.5,'Affordable phone with good features.'),
('2025-07-15',11,263,4.6,'TV delivers excellent picture quality.'),
('2025-07-16',12,259,4.8,'Top-notch wash performance.');
```



15. Add_to_wishlist:-

```
INSERT INTO OrderItems (item_id, pro_id, price, creation_Time) VALUES (1010,263,30000,'2025-07-01'),
(2020,262,9999,'2025-07-02'),(3030,261,3499,'2025-07-03'),(4040,260,35000,'2025-07-04'),
(5050,259,45000,'2025-07-05'),(6060,258,22999,'2025-07-06'),(7070,257,1999,'2025-07-07'),
(8080,256,1499,'2025-07-08'),(9090,258,22999,'2025-07-09'),(10101,262,9999,'2025-07-10');
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:59:51	INSERT INTO OrderItems (item_id, pro_id, price, creation_Time) VALUES (1010,263,30000,'2025-07-01'), (2020,2...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec

16. Contain:-

```
INSERT INTO Contain (order_Number, item_id, volume) VALUES (1001,1010,1),
(1002,2020,2),(1003,3030,1),(1004,4040,1),(1005,5050,1),(1006,6060,1),
(1007,7070,1),(1008,8080,1),(1009,9090,1),(1010,10101,1),(1011,8080,1),
(1012,7070,1),(1013,3030,1),(1002,10101,1),(1003,6060,1),(1004,2020,1),(1005,1010,1);
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	23:22:43	INSERT INTO Contain (order_Number, item_id, volume) VALUES (1001,1010,1), (1002,2020,2),(1003,3030,1),(10...	17 row(s) affected Records: 17 Duplicates: 0 Warnings: 0	0.016 sec

17. Payments:-

```
INSERT INTO Payments (order_Number, credit_cardNumber, pay_Time) VALUES (1002,'ICICI1234567890','2025-07-02'),
(1003,'HDFC0987654321','2025-07-03'),(1005,'SBI1111222233334444','2025-07-05'),(1006,'AXIS5555666677778888','2025-07-06'),
(1008,'PNB9999000011112222','2025-07-08'),(1009,'KOTA3333444455556666','2025-07-09'),(1010,'YES7777888899990000','2025-07-10');
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	23:26:35	INSERT INTO Payments (order_Number, credit_cardNumber, pay_Time) VALUES (1002,'ICICI1234567890','2025...	7 row(s) affected Records: 7 Duplicates: 0 Warnings: 0	0.016 sec

18. Ship_To:-

```
INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (101,1001,'2025-07-04');
INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (102,1002,'2025-07-05');
INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (103,1003,'2025-07-06');
INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (104,1004,'2025-07-07');
INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (105,1005,'2025-07-08');
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	23:28:26	INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (101,1001,'2025-07-04')	1 row(s) affected	0.032 sec
2	23:28:26	INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (102,1002,'2025-07-05')	1 row(s) affected	0.000 sec
3	23:28:26	INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (103,1003,'2025-07-06')	1 row(s) affected	0.000 sec
4	23:28:26	INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (104,1004,'2025-07-07')	1 row(s) affected	0.000 sec
5	23:28:26	INSERT INTO Ship_To (address_id, order_Number, Time_Delivered) VALUES (105,1005,'2025-07-08')	1 row(s) affected	0.000 sec

19. Oversee:-

```
INSERT INTO Oversee (purchaser_id, sp_id, initiate_Time) VALUES (14,201,'2025-07-01'),
(15,202,'2025-07-02'),(16,203,'2025-07-03'),(17,204,'2025-07-04'),(18,205,'2025-07-05');
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	23:31:09	INSERT INTO Oversee (purchaser_id, sp_id, initiate_Time) VALUES (14,201,'2025-07-01'), (15,202,'2025-07-02'),...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0	0.015 sec

20. After_Sales_care:-

```
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('TataElectronics',0206);
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('RelianceMobiles',0245);
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('MahindraAppliances',0246);
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('GodrejHome',0247);
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('LGIndia',0253);
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('SamsungIndia',0254);
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('TataElectronics',0246);
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES('RelianceMobiles',0247);
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('MahindraAppliances',0253);
INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('GodrejHome',0254);
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('TataElectronics',0206)	1 row(s) affected	0.016 sec
2	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('RelianceMobiles',0245)	1 row(s) affected	0.000 sec
3	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('MahindraAppliances',0246)	1 row(s) affected	0.000 sec
4	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('GodrejHome',0247)	1 row(s) affected	0.015 sec
5	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('LGIndia',0253)	1 row(s) affected	0.000 sec
6	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('SamsungIndia',0254)	1 row(s) affected	0.000 sec
7	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('TataElectronics',0246)	1 row(s) affected	0.000 sec
8	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES('RelianceMobiles',0247)	1 row(s) affected	0.016 sec
9	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('MahindraAppliances',0253)	1 row(s) affected	0.000 sec
10	23:33:40	INSERT INTO After_Sales_care (brand_Name, sh_id) VALUES ('GodrejHome',0254)	1 row(s) affected	0.000 sec

WRITING QUERIES FOR ABOVE FUNCTIONALITES:-

1. List all_purchaser names and their cities

```
319 •  SELECT p.purchaser_id,p.name,sa.city
320    FROM Purchaser AS p
321    JOIN shipping_address AS sa ON p.name=sa.name;
322
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

purchaser_id	name	city
1	Aarav Sharma	Secunderabad
2	Priya Patel	Ahmedabad
3	Rahul Singh	New Delhi
4	Ananya Mukherjee	Bangalore
5	Vikram Nair	Mumbai
6	Deepika Rao	Kolkata
7	Karan Mehta	Chennai
8	Sneha Verma	Mumbai
9	Rohit Desai	Lucknow
10	Isha Gupta	Bhopal
11	Aditya Joshi	Chandigarh
12	Neha Kapoor	Jaipur
13	Sanjay Kumar	Gurgaon
14	Vandana Iyer	Kochi
15	Manish Tiwari	Bhubaneswar
16	Leela Menon	Mysore
17	Arjun Reddy	Hyderabad
18	Smita Singh	Raipur

Explanation:-

This SQL statements **JOIN**, by default, is the same as **INNER JOIN**—meaning only rows with matching values in both tables are included. It combines rows from the **Purchaser** table (aliased p) with those from **shipping_address** (aliased sa), using the condition that the name column matches in both tables.The result includes p.purchaser_id, p.name, and sa.city

2.Find all banking cards issued by 'HDFC Bank'

```
323 •  SELECT cardNumber,runoutDate
324    FROM Banking_Card
325    WHERE Bank='HDFC Bank';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

cardNumber	runoutDate
HDFC0987654321	2028-05-15
NULL	NULL

Explanation:-

This SQL statements Retrieves two columns—**cardNumber** and **runoutDate**—from the **Banking_Card** table.Filters the results to only include rows where the **Bank** column equals 'HDFC Bank'.

3.Get all debit card number and the names of the purchasers who own them

```
336 •  SELECT cardNumber,name
337    FROM Debit_Card AS dc
338    JOIN purchaser AS p ON dc.purchaser_id=p.purchaser_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

cardNumber	name
IDBI2222333344445555	Sneha Verma
BOB6666777788889999	Rohit Desai
CAN1111222233334445	Isha Gupta
UCO5555666677778889	Aditya Joshi
IOB9999000011112223	Neha Kapoor
AXIS5555666677778888	Sanjay Kumar

Explanation:-

This SQL statements Performs an **INNER JOIN** between the Debit_Card table (dc) and the purchaser table (p) using the shared key purchaser_id.
Retrieves two columns in the result:

- cardNumber from the Debit_Card table.
- name from the purchaser table.

4.Update the Phonenumber of Shipping_Address with purchaser phoneNumber

```
327 •  UPDATE Shipping_Address AS s, purchaser AS p
328    SET s.contact_PhoneNumber =p.phoneNumber
329    WHERE s.purchaser_id=p.purchaser_id;
330
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	23:47:47	UPDATE Shipping_Address AS s, purchaser AS p SET s.contact_PhoneNumber =p.phoneNumber WHERE s.pur...	0 row(s) affected Rows matched: 25 Changed: 0 Warnings: 0	0.016 sec

Explanation:-

This SQL statement, It updates the Shipping_Address table, setting its contact_PhoneNumber column to match the phoneNumber value from the purchaser table.

How it works: For each pair of rows where s.purchaser_id = p.purchaser_id, the statement copies p.phoneNumber into s.contact_PhoneNumber

5. Show Shipping_Address

```
331 •  SELECT * FROM Shipping_Address;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

address_id	purchaser_id	name	contact_PhoneNumber	territory	city	streetaddress	postalCode
101	1	Aarav Sharma	+91-987-543-10	Telangana	Secunderabad	4-158/9 5th Cross Road, Sainikpuri	500094
102	2	Priya Patel	+91-912-456-80	Gujarat	Ahmedabad	Plot 22, Navrangpura Main Road, Navrangpura	380009
103	3	Rahul Singh	+91-998-776-55	Delhi	New Delhi	Flat 408, Tagore Road, Near Khan Market	110001
104	4	Ananya Mukherjee	+91-923-567-91	Karnataka	Bangalore	32/1 2nd Main Road, 8th Block, Jayanagar	560082
105	5	Vikram Nair	+91-935-787-02	Maharashtra	Mumbai	Apte Building, Dadabhai Naoroji Road, Fort	400001
106	6	Deepika Rao	+91-945-789-13	West Bengal	Kolkata	15 Sudder Street, Esplanade Area	700016
107	7	Karan Mehta	+91-956-890-24	Tamil Nadu	Chennai	15/A Anna Salai, Near Government Museum	600002
108	8	Sneha Verma	+91-967-901-35	Maharashtra	Mumbai	Altamount Road, Near Antilia	400026
109	9	Rohit Desai	+91-978-012-46	Uttar Pradesh	Lucknow	House No. 56/78, 14th A Cross, JP Nagar Phase 7	226001
110	10	Isha Gupta	+91-989-123-57	Madhya Pradesh	Bhopal	Flat No. 402, Rosewood Apartments, MP Nagar	462011
111	11	Aditya Joshi	+91-990-123-56	Punjab	Chandigarh	Sector 22-C, House No. 134	160022
112	12	Neha Kapoor	+91-991-234-67	Rajasthan	Jaipur	E-8, Malviya Nagar, Opposite Malviya Nagar Me...	302017
113	13	Sanjay Kumar	+91-992-345-78	Haryana	Gurgaon	A-75, Sector 14, DLF City	122001
114	14	Vandana Iyer	+91-993-456-89	Kerala	Kochi	Flat No. 202, Seafront Apartments, Marine Drive	682031
115	15	Manish Tiwari	+91-994-567-90	Odisha	Bhubaneswar	Plot 101, IRC Village, Nayapalli	751015
116	16	Leela Menon	+91-995-578-01	Karnataka	Mysore	No. 12, Lakshmi Road, Opposite City Market	570024
117	17	Arjun Reddy	+91-996-789-12	Telangana	Hyderabad	House No. 23, Road No. 12, Banjara Hills	500034
118	18	Smita Singh	+91-997-890-23	Chhattisgarh	Raipur	C-45, Mowa Road, Sector 1, Shankar Nagar	492007
119	19	Gaurav Jain	+91-998-901-34	Jharkhand	Ranchi	Flat 15B, Nirmala Apartment, Kanke Road	834008

Shipping_Address 3

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	23:47:47	UPDATE Shipping_Address AS s, purchaser AS p SET s.contact_PhoneNumber =p.phoneNumber WHERE s.p...	0 row(s) affected Rows matched: 25 Changed: 0 Warnings: 0	0.016 sec
2	23:54:17	SELECT * FROM Shipping_Address LIMIT 0, 1000	25 row(s) returned	0.000 sec / 0.000 sec

Explanation:-

This SQL statement

- SELECT * retrieves all columns from the specified table.
- FROM Shipping_Address indicates that you're pulling data from the Shipping_Address table.

Put together, this query returns every column and every row in the Shipping_Address table — essentially a full dump of its contents**.**

6. Display the purchaser id and name of all purchasers who have a debit card and show their credit card details if available

```
346 •  SELECT p.purchaser_id,p.name
347    FROM Purchaser AS p
348    LEFT JOIN Credit_Card AS cc ON p.purchaser_id= cc.purchaser_id
349    JOIN Debit_Card AS dc ON p.purchaser_id = dc.purchaser_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

purchaser_id	name
8	Sneha Verma
9	Rohit Desai
10	Isha Gupta
11	Aditya Joshi
12	Neha Kapoor
13	Sanjay Kumar

Explanation:-

This SQL statement,

LEFT JOIN Credit_Card,Returns **all** purchasers, whether or not they have a matching row in the Credit_Card table. If there's no match, the cc columns will be NULL for those purchasers. This preserves all rows from the Purchaser table.

INNER JOIN Debit_Card Filters the results further to include only purchasers who *do* have a corresponding entry in the Debit_Card table. This means only purchasers with a debit card will be included in the final output—even if they have no credit card data.

7. List all shipping address for Purchaser ID=5

```
346 •  SELECT streetaddress,city,postalcode
347    FROM shipping_address
348    WHERE purchaser_id=5;
349
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

streetaddress	city	postalcode
Apte Building, Dadabhai Naoroji Road, Fort	Mumbai	400001

Explanation:-

This SQL statement Retrieves only the streetaddress, city, and postalcode columns. Specifies the data comes from the shipping_address table.Applies a filter to return only those rows where the purchaser_id is exactly **5**

8. Find all purchasers whose debit cards expires before 2028

```
360 •  SELECT DISTINCT p.name,bc.runoutDate
361    FROM Debit_Card as dc
362    JOIN Banking_Card AS bc ON dc.cardNumber=bc.cardNumber
363    JOIN purchaser as p ON dc.purchaser_id = p.purchaser_id
364    WHERE bc.runoutDate < '2028-01-01';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	runoutDate
Sneha Verma	2027-10-11
Aditya Joshi	2027-08-22

Explanation:-

This SQL statement Ensures the output lists only unique combinations of purchaser name (p.name) and card expiration date (bc.runoutDate), eliminating exact duplicate row pair. Debit_Card joins Banking_Card via matching cardNumber.Then it joins purchaser based on purchaser_id.Filters to include only those cards whose runoutDate is **mentioned.**

9. Get purchaser details along with their bank name for all credit card holders

```
351 •  SELECT p.name,bc.bank,cc.organization
352    FROM purchaser AS p
353    JOIN Credit_Card AS cc ON p.purchaser_id= cc.purchaser_id
354    JOIN Banking_Card AS bc ON cc.cardNumber= bc.cardNumber;
355
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	bank	organization
Ananya Mukherjee	Axis Bank	AXIS
Priya Patel	HDFC Bank	HDFC
Aarav Sharma	ICICI Bank	ICICI
Deepika Rao	Kotak Mahindra Bank	Kotak
Vikram Nair	Punjab National Bank	PNB
Rahul Singh	State Bank of India	SBI
Karan Mehta	Yes Bank	Yes

Explanation:-

This SQL statements, The purchaser table (p) is joined to the Credit_Card table (cc) via a matching purchaser_id.Then, Credit_Card links to Banking_Card (bc) using the common cardNumber.

10. Find the total number of shipping address per city.

```
361 •  SELECT city, COUNT(*) AS total_address
362    FROM Shipping_address
363    GROUP BY city;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

city	total_address
Secunderabad	1
Ahmedabad	1
New Delhi	1
Bangalore	1
Mumbai	2
Kolkata	1
Chennai	1
Lucknow	1
Bhopal	1
Chandigarh	1
Jaipur	1
Gurgaon	1
Kochi	1
Bhubaneswar	1
Mysore	1
Hyderabad	1
Raipur	1

Explanation:-

This SQL statements,This clause groups all rows in the Shipping_address table by the value in the city column so each distinct city forms its own group.For each city group, the COUNT(*) function calculates how many records (rows) exist. The result is labeled as total_address.

11.List the bank names with the total number of cards issued(cc+dc)

```
371 •  SELECT bc.bank,COUNT(*) AS total_cards
372    FROM Banking_Card AS bc
373    LEFT JOIN Credit_Card AS cc ON bc.cardNumber = cc.cardNumber
374    LEFT JOIN Debit_Card AS dc ON bc.cardNumber = dc.cardNumber
375    GROUP BY bc.bank;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

bank	total_cards
Axis Bank	1
Bank of Baroda	1
Canara Bank	1
HDFC Bank	1
ICICI Bank	1
IDBI Bank	1
Indian Overseas Bank	1
Kotak Mahindra Bank	1
Punjab National Bank	1
State Bank of India	1
UCO Bank	1
Yes Bank	1

Explanation:-

This SQL statements

- **LEFT JOIN Credit_Card & Debit_Card Tables**

These joins ensure **all rows** from Banking_Card are included, even if there's no matching data in Credit_Card or Debit_Card. Non-matching right-side fields appear as NULL

- **GROUP BY bc.bank**

Groups results by each distinct bank, enabling you to compute counts per bank

- **COUNT(*) AS total_cards**

Counts all Banking_Card records for each bank, regardless of whether they have corresponding credit or debit entries.

12. Find the maximum and minimum card expiry dates for each bank

```
377 •  SELECT bank, MIN(runoutDate) AS earliest_expiry, MAX(runoutDate) AS latest_expiry
378    FROM Banking_Card
379    GROUP BY bank;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

bank	earliest_expiry	latest_expiry
Axis Bank	2029-01-20	2029-01-20
Bank of Baroda	2029-12-01	2029-12-01
Canara Bank	2028-04-18	2028-04-18
HDFC Bank	2028-05-15	2028-05-15
ICICI Bank	2027-12-31	2027-12-31
IDBI Bank	2027-10-11	2027-10-11
Indian Overseas Bank	2029-06-30	2029-06-30
Kotak Mahindra Bank	2028-07-25	2028-07-25
Punjab National Bank	2027-03-10	2027-03-10
State Bank of India	2026-11-30	2026-11-30
UCO Bank	2027-08-22	2027-08-22
Yes Bank	2026-09-05	2026-09-05

Explanation:-

This SQL statements

- **MIN(runoutDate)**: Finds the **earliest** expiration date for each bank.
- **MAX(runoutDate)**: Finds the **latest** expiration date for each bank.
- **GROUP BY bank**: Groups the rows based on the bank column so that the aggregate functions (MIN, MAX) operate within each bank group.

In essence, for every bank in your Banking_Card table, the query returns both the soonest and the furthest card expiration dates

13. Find the average number of cards per purchaser.

```
381 •  SELECT AVG(card_count) AS avg_cards
382    FROM (
383        SELECT purchaser_id, COUNT(DISTINCT cardNumber) AS card_count
384        FROM (
385            SELECT purchaser_id, cardNumber FROM Credit_Card
386            UNION
387            SELECT purchaser_id, cardNumber FROM Debit_Card
388        ) AS all_cards
389        GROUP BY purchaser_id
390    ) AS card_stats;
391
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

avg_cards
1.0000

Explanation:-

This SQL statements

1. **Combines Credit and Debit Card Data via UNION**
   - The inner UNION merges all rows from Credit_Card and Debit_Card, ensuring each (purchaser_id, cardNumber) pair is unique **by default**.
2. **Counts Distinct Cards per Purchaser**
   - The intermediate query groups by purchaser_id and counts each purchaser's **unique** card numbers, storing that in card_count.
3. **Computes the Overall Average**
   - The outer query takes the average (AVG) of all card_count values□i.e., the average number of distinct cards held per purchaser.

This query calculates the **average number of distinct cards** (credit or debit) per customer. It ensures no duplicate card numbers are counted for the same purchaser, even if a card appears in both credit and debit tables.

Conclusions and Recommendations

Summation:

- The schema supports full OMS workflows and ensures data integrity.
- It's scalable for features like returns, discounts, dynamic inventory allocation, or analytics dashboards.
- **Recommendation**: Extend with performance enhancements (indexes), UI layer, and automated operations (procedures/triggers) in future iterations.