
**Database Systems**
**Final Report**

# Software as a Service (SaaS) Platform

**Submitted by:** Alishbah Bashir
**Submitted to:** Sir Irfan Hameed
**Dated:** 16-12-2024

**Pakistan Institute of Engineering and Applied Sciences, Lehtrar Road, Nilore, Islamabad**

## Introduction:

Our Software-as-a-Service (SaaS) platform is a comprehensive solution designed to streamline subscription management for businesses and their customers. This system empowers organizations to efficiently handle subscription plans, customers, payments, and discounts in a structured and scalable manner.

---

## Key Features:

1. Subscription Plans
2. Customer Management
3. Payment Tracking
4. Discount Management
5. Plan Components (PlanLines)
6. Flexible Payment Methods
7. Subscription Tracking

---

## Use Case (our scenario):

A platform offering different photo and video editing tools (such as canva, adobe photoshop, adobe illustrator, capcut pro, kapwing, blender etc) in the form of plans to users/businesses, where they can get subscriptions based on plans they opt. Different plans have different levels such as basic, intermediate and pro, in which access to certain features of tools/softwares is given in every plan. Users are also given dicounts on the basis of duration for which they buy plans.

---

## Entities and their Attributes:

**1. Entities:**

1. **Plan:**
    - Attributes: Plan_id (Primary Key), PlanName, Price_per_month.
2. **Customer:**
    - Attributes: Customer_id (Primary Key), Name, Email, Cell_no., Account_No., City, Country.
3. **Payment:**
    - Attributes: Payment_id (Primary Key), Subscription_id, Method_id, Amount, Date & Time.
4. **Subscription:**
    - Attributes: Subscription_ID (Primary Key), Customer_id, Plan_id , Percent_discount, Duration.

5. **PlanLines:**
    - Attributes: Plan_line_id (Primary Key), Plan_id, S_T_id, Access_level.
6. **Payment_Method:**
    - Attributes: Method_id (Primary Key), Method_name.
7. **Softwares/Tools_Offered:**
    - Attributes: S_T_iD (Primary Key), S_T_name, Description.

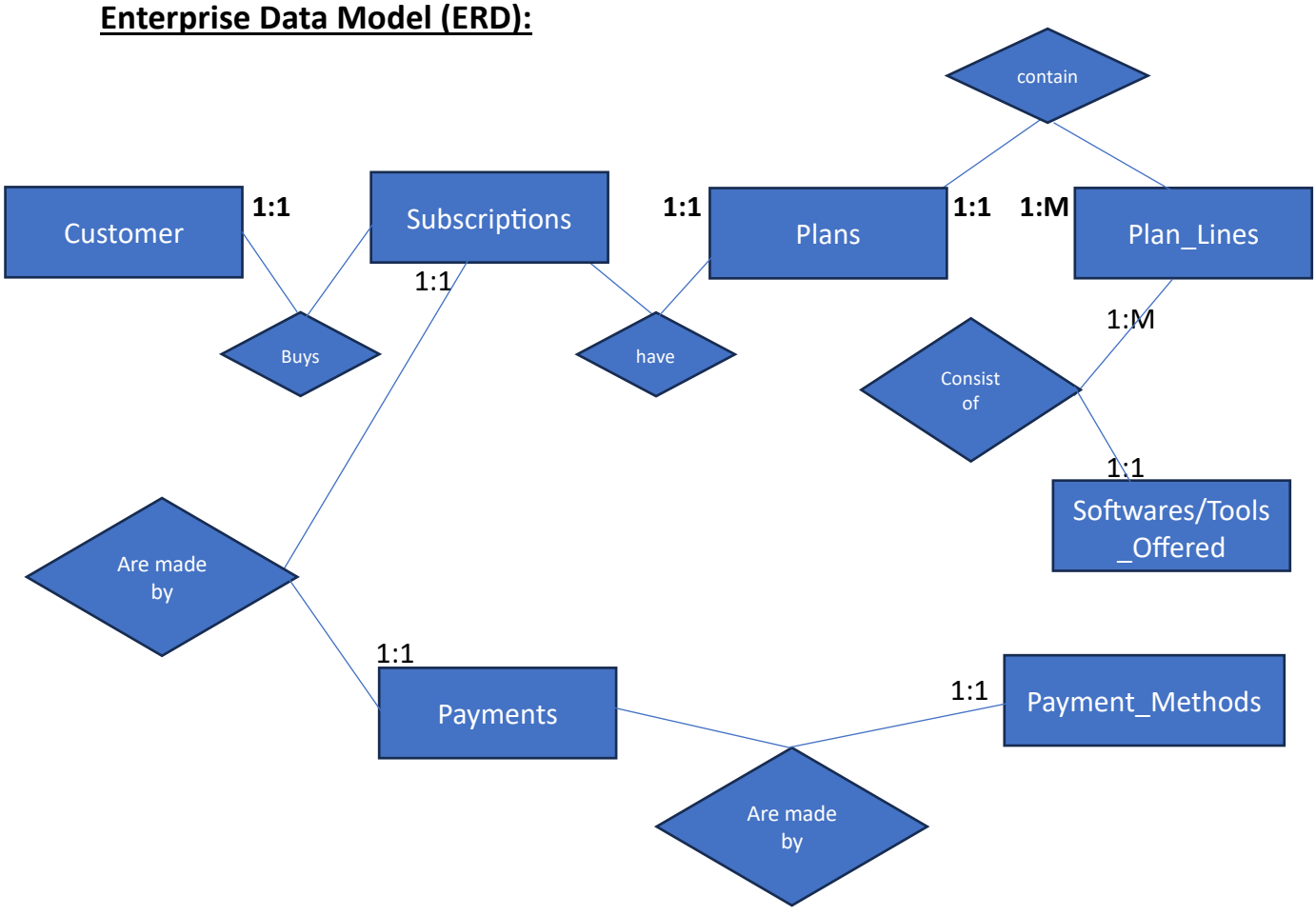
---

## Business Rules:

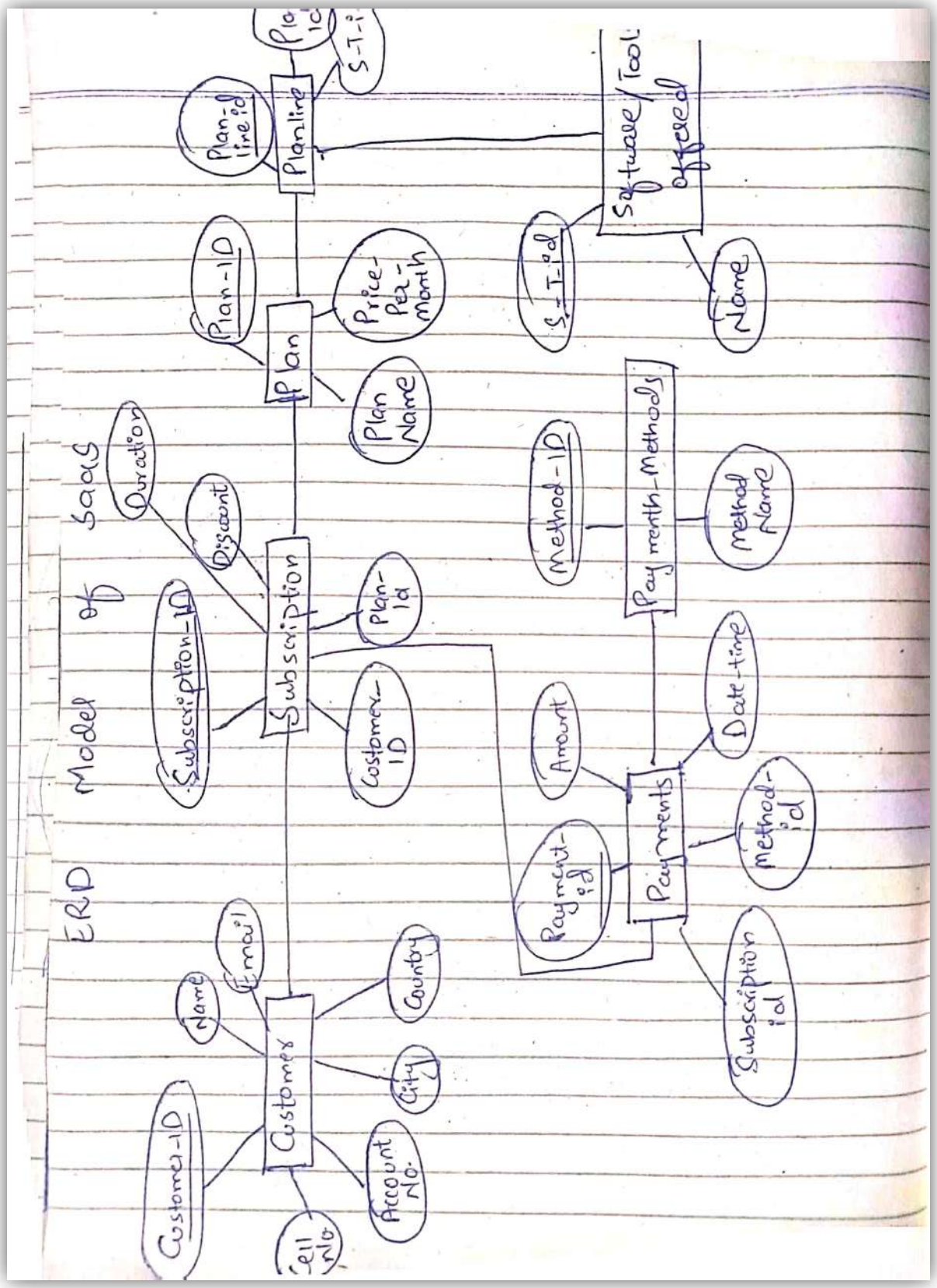
1. A customer can have multiple subscriptions (max), and min 0 subscriptions, a subscription can belong to only single customer.
2. Subscription includes max or min a single plan, but a plan can come under multiple subscriptions (max) and min 0 subscriptions.
3. Plans can have max multiple plan lines, and min 1 plan line, while a plan line can belong to one and only one plan, (in both min max cases).
4. Plan lines can have max and min 1 tools/softwares offered, while softwares/tools can belong to max multiple plan lines and min 1 plan line.
5. Subscription can have max 1 payment, and payment can belong to only 1 subscription (both min and max).
6. Payment is made only single method, while a payment method can be used by mutiple payments (max), and minimum 0 payment.

---

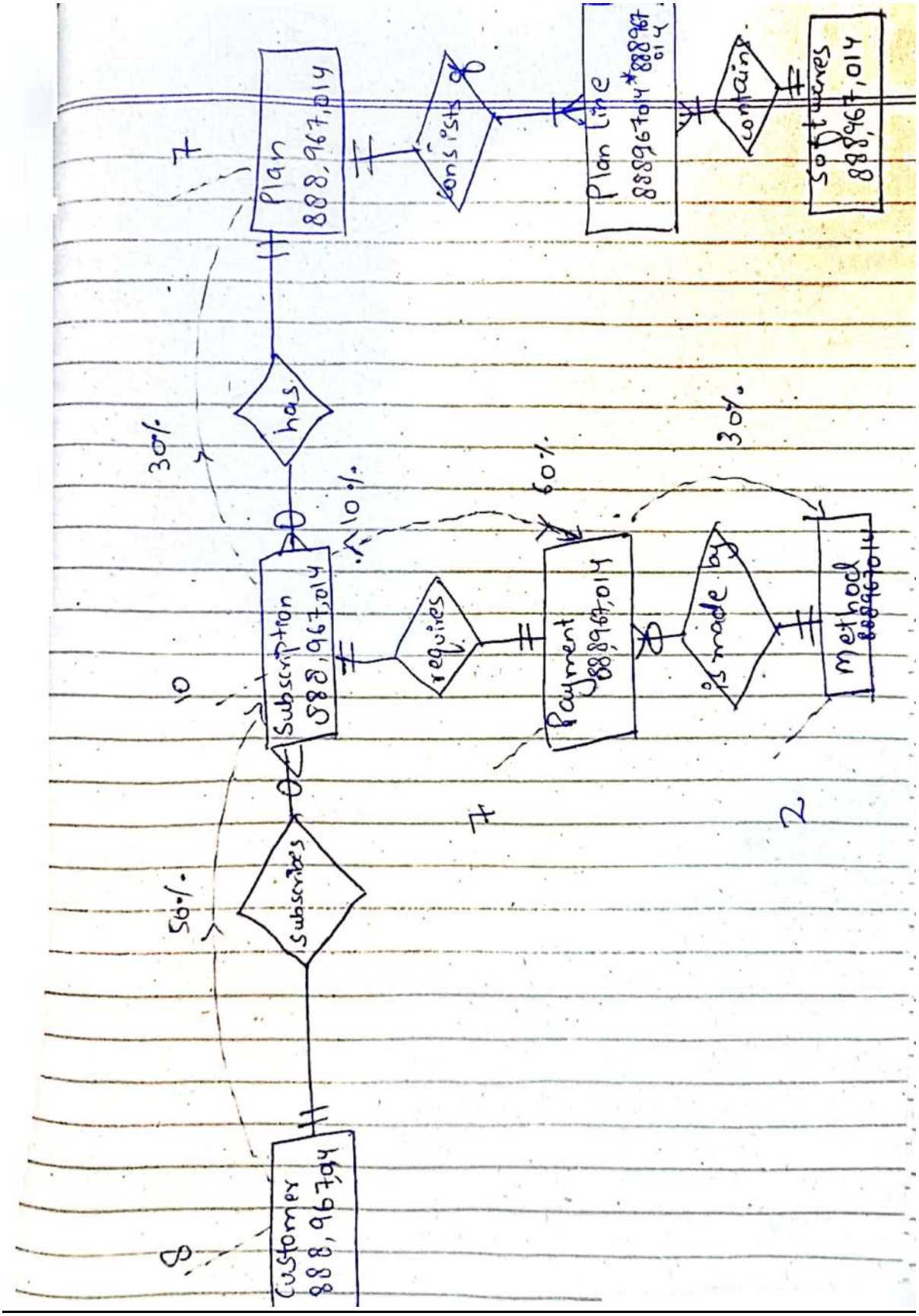
## Enterprise Data Model (ERD):



**Entity-Relationship Diagram:**



**Composite Usage Model:**



**Relational Schema:**

**Plan:**

| **Plan_ID** | Plan_Name | Price_Per_Month |
|---|---|---|

**Plan_Lines:**

| **Plan_Line_ID** | Plan_ID | S_T_id | Access_level |
|---|---|---|---|

**Subscription:**

| **Subscription_ID** | Customer_ID | Plan_ID | Discount | Duration |
|---|---|---|---|---|

**Customer**:

| **Customer_ID** | Name | Email | Cell_No | Account_No | City | Country |
|---|---|---|---|---|---|---|

**Software_Tools_Offered:**

| **S_T_ID** | Name | Description |
|---|---|---|

**Payment:**

| **Payment_ID** | Subscription_id | Method_ID | Amout | Date_Time |
|---|---|---|---|---|

**Payment_Method:**

| **Method_ID** | Method_Name |
|---|---|

## MetaData of Entities:

### Customer Table:

| Column Name | Data Type | Description |
|---|---|---|
| Customer_ID | VARCHAR(10) | Primary key |
| Name | VARCHAR(100) | Name of the customer |
| Email | VARCHAR(150) | Email address |
| Cell_No | VARCHAR(15) | Phone number |
| Account_No | VARCHAR(50) | Account number |
| City | VARCHAR(50) | City of the customer |
| Country | VARCHAR(50) | Country of the customer |

### Plan Table:

| Column Name | Data Type | Description |
|---|---|---|
| Plan_ID | VARCHAR(10) | Primary key |
| Plan_Name | VARCHAR(100) | Name of the plan |
| Price_Per_Month | Decimal (10,2) | Monthly price of the plan |

### Plan_line Table:

| Column Name | Data Type | Description |
|---|---|---|
| Plan_Line_ID | VARCHAR(10) | Primary key |
| Plan_ID | VARCHAR(10) | Primary key |
| S_T_ID | VARCHAR(10) | Foreign key (Software_Tools_Offered Table) |
| Access_Level | INT | Access to softwares offered against each plan |

### Subscription Table:

| Column Name | Data Type | Description |
|---|---|---|
| Subscription_ID | VARCHAR(10) | Primary key |
| Customer_ID | VARCHAR(10) | Foreign key (Customer) |
| Plan_ID | VARCHAR(10) | Foreign key (Plan) |
| Percent_Discount | Float | Discount applied to the plan (%) |
| Duration | Integer | Subscription duration (in months) |

## Software_Tools_Offered Table:

| Column Name | Data Type | Description |
|---|---|---|
| S_T_ID | VARCHAR(10) | Primary key |
| Name | VARCHAR(100) | Name of the software/tool |
| Description | VARCHAR (150) | Decsription of software offered |

## Payment Table:

| Column Name | Data Type | Description |
|---|---|---|
| Payment_ID | VARCHAR(10) | Primary key |
| Subscription_ID | VARCHAR(10) | Foreign key (Subscription) |
| Method_ID | VARCHAR(10) | Foreign key (Payment_Method) |
| Amount | Decimal | Payment amount |
| Date_Time | Date & Time | Date and Time at which payment was made |

## Payment_Method Table:

| Column Name | Data Type | Description |
|---|---|---|
| Method_ID | VARCHAR(10) | Primary key |
| Method_Name | VARCHAR(100) | Name of the payment method |

## Constraints:

| Table Name | Attribute Name | Data Type | Constraints |
|---|---|---|---|
| **Customer** | Customer_ID | VARCHAR(10) | PRIMARY KEY, NOT NULL |
| | Name | VARCHAR(100) | NOT NULL |
| | Email | VARCHAR(100) | UNIQUE, NOT NULL |
| | Cell_No | VARCHAR(15) | UNIQUE |
| | Account_No | VARCHAR(20) | UNIQUE, NOT NULL |
| | City | VARCHAR(20) | NONE |
| | Country | VARCHAR(20) | NONE |
| **Subscription** | Subscription_ID | VARCHAR(10) | PRIMARY KEY, NOT NULL |
| | Customer_ID | VARCHAR(10) | FOREIGN KEY REFERENCES Customer(CustomerID), NOT NULL |
| | Plan_ID | VARCHAR(10) | FOREIGN KEY REFERENCES Plan(PlanID), NOT NULL |
| | Discount | INT | NOT NULL |
| | Duration | INT | NOT NULL |
| **Plan** | Plan_ID | VARCHAR(10) | PRIMARY KEY, NOT NULL |

| | | | |
|---|---|---|---|
| | PlanName | VARCHAR(100) | NOT NULL |
| | Price_Per_Month | DECIMAL(10, 2) | NOT NULL |
| **Plan_Line** | Plan_Line_ID | VARCHAR(10) | PRIMARY KEY, NOT NULL |
| | Plan_ID | VARCHAR(10) | FOREIGN KEY REFERENCES Plan(PlanID), NOT NULL |
| | S_T_ID | VARCHAR(10) | FOREIGN KEY REFERENCES Tool(S_T_ID), NOT NULL |
| | Access_Level | INT | NOT NULL |
| **Software_Tools_Offered** | S_T_ID | VARCHAR(10) | PRIMARY KEY, NOT NULL |
| | Name | VARCHAR(100) | NOT NULL |
| | Description | TEXT | None |
| **Payment** | Payment_ID | VARCHAR(10) | PRIMARY KEY, NOT NULL |
| | Subscription_ID | VARCHAR(10) | UNIQUE, FOREIGN KEY REFERENCES Subscription(SubscriptionID), NOT NULL |
| | Method_ID | VARCHAR(10) | FOREIGN KEY REFERENCES PaymentMethod(PaymentMethodID), NOT NULL |
| | Amount | DECIMAL(10, 2) | NOT NULL |
| | Date | DATE | NOT NULL |
| **PaymentMethod** | Method_ID | VARCHAR(10) | PRIMARY KEY, NOT NULL |
| | MethodName | VARCHAR(100) | NOT NULL |

## Normalization:

All tables are already in normal form upto 3NF.

## Creating DB through SQL commands:

## Creating Tables:

**Creating Customer Table**

```
CREATE TABLE Customer (
   Customer_ID INT PRIMARY KEY NOT NULL,
   Name VARCHAR(100) NOT NULL,
   Email VARCHAR(100) UNIQUE NOT NULL,
   Cell_No VARCHAR(15) UNIQUE,
   Account_No VARCHAR(20) UNIQUE NOT NULL,
   City VARCHAR(20),
```

```
    Country VARCHAR(20)
);
```

**Creating Plan Table**

```
CREATE TABLE Plan (
    Plan_ID INT PRIMARY KEY NOT NULL,
    PlanName VARCHAR(100) NOT NULL,
    Price_Per_Month DECIMAL(10, 2) NOT NULL
);
```

**Creating Subscription Table**

```
CREATE TABLE Subscription (
    Subscription_ID INT PRIMARY KEY NOT NULL,
    Customer_ID INT NOT NULL,
    Plan_ID INT NOT NULL,
    Discount INT NOT NULL,
    Duration INT NOT NULL,
    CONSTRAINT fk_Customer FOREIGN KEY (Customer_ID) REFERENCES
Customer(Customer_ID),
    CONSTRAINT fk_Plan FOREIGN KEY (Plan_ID) REFERENCES Plan(Plan_ID)
);
```

**Creating Software_Tools_Offered Table**

```
CREATE TABLE Software_Tools_Offered (
    S_T_ID INT PRIMARY KEY NOT NULL,
    Name VARCHAR(100) NOT NULL,
    Description CLOB
);
```

**Creating Plan_Line Table**

```
CREATE TABLE Plan_Line (
    Plan_Line_ID INT PRIMARY KEY NOT NULL,
    Plan_ID INT NOT NULL,
    S_T_ID INT NOT NULL,
    Access_Level INT NOT NULL,
    CONSTRAINT fk_Plan_Line_Plan FOREIGN KEY (Plan_ID) REFERENCES Plan(Plan_ID),
    CONSTRAINT fk_Plan_Line_Software FOREIGN KEY (S_T_ID) REFERENCES
Software_Tools_Offered(S_T_ID)
);
```

**Creating PaymentMethod Table**

```
CREATE TABLE PaymentMethod (
    Method_ID INT PRIMARY KEY NOT NULL,
    MethodName VARCHAR(100) NOT NULL
);
```

**Creating Payment Table**

```
CREATE TABLE Payment (
   Payment_ID INT PRIMARY KEY NOT NULL,
   Subscription_ID INT NOT NULL,
   Method_ID INT NOT NULL,
   Amount DECIMAL(10, 2) NOT NULL,
   Payment_Date DATE NOT NULL,
   CONSTRAINT fk_Subscription FOREIGN KEY (Subscription_ID) REFERENCES
Subscription(Subscription_ID),
   CONSTRAINT fk_PaymentMethod FOREIGN KEY (Method_ID) REFERENCES
PaymentMethod(Method_ID)
);
```

```
-- Customer Table
CREATE TABLE Customer (
    Customer_ID INTEGER PRIMARY KEY,
    Name VARCHAR(100),
    Email VARCHAR(150),
    Cell_No VARCHAR(15),
    Account_No VARCHAR(50),
    City VARCHAR(50),
    Country VARCHAR(50)
);

-- Plan Table
CREATE TABLE Plan (
    Plan_ID INTEGER PRIMARY KEY,
    Plan_Name VARCHAR(100),
```

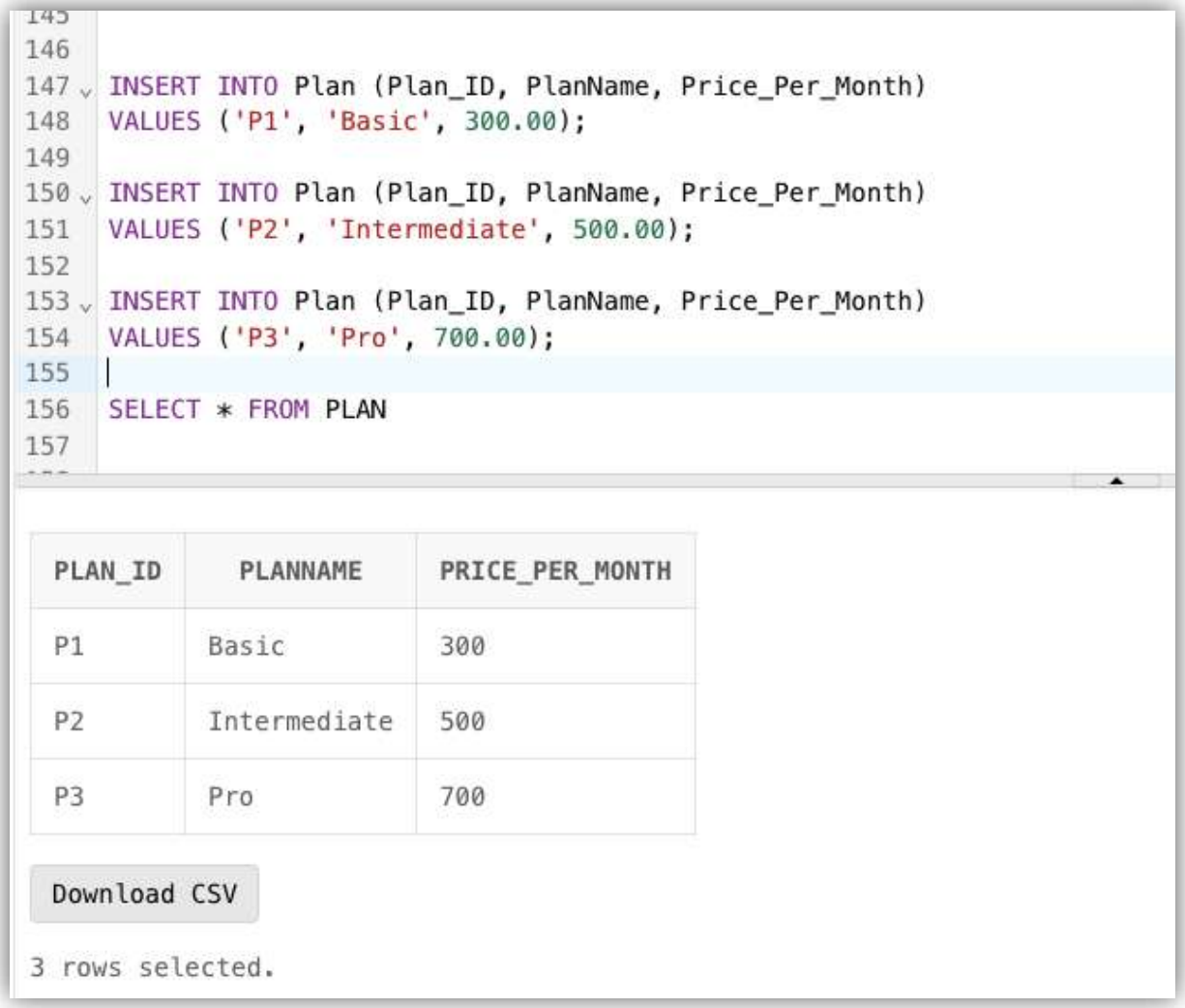
```
Table created.

Table created.

Table created.

Table created.

Table created.

Table created.

Table created.
```

## Inserting Values:

**INSERT INTO Plan Table:**

```
INSERT INTO Plan (Plan_ID, PlanName, Price_Per_Month)
VALUES ('P1', 'Basic', 300.00);

INSERT INTO Plan (Plan_ID, PlanName, Price_Per_Month)
VALUES ('P2', 'Intermediate', 500.00);
```

INSERT INTO Plan (Plan_ID, PlanName, Price_Per_Month)
VALUES ('P3', 'Pro', 700.00);



```
146
147 INSERT INTO Plan (Plan_ID, PlanName, Price_Per_Month)
148 VALUES ('P1', 'Basic', 300.00);
149
150 INSERT INTO Plan (Plan_ID, PlanName, Price_Per_Month)
151 VALUES ('P2', 'Intermediate', 500.00);
152
153 INSERT INTO Plan (Plan_ID, PlanName, Price_Per_Month)
154 VALUES ('P3', 'Pro', 700.00);
155
156 SELECT * FROM PLAN
157
```

| PLAN_ID | PLANNAME | PRICE_PER_MONTH |
|---|---|---|
| P1 | Basic | 300 |
| P2 | Intermediate | 500 |
| P3 | Pro | 700 |

Download CSV

3 rows selected.

**INSERT INTO Customer Table:**

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C101', 'John Doe', 'john.doe@example.com', '1234567890', 'AC101', 'New York', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C102', 'Jane Smith', 'jane.smith@example.com', '1234567891', 'AC102', 'Los Angeles', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C103', 'Alice Johnson', 'alice.johnson@example.com', '1234567892', 'AC103', 'Chicago', 'USA');

```
INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C104', 'Bob Brown', 'bob.brown@example.com', '1234567893', 'AC104', 'Houston',
'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C105', 'Charlie White', 'charlie.white@example.com', '1234567894', 'AC105',
'Miami', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C106', 'Daisy Green', 'daisy.green@example.com', '1234567895', 'AC106', 'Seattle',
'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C107', 'Ethan Black', 'ethan.black@example.com', '1234567896', 'AC107', 'Austin',
'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C108', 'Fiona Grey', 'fiona.grey@example.com', '1234567897', 'AC108', 'Boston',
'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C109', 'George Blue', 'george.blue@example.com', '1234567898', 'AC109',
'Denver', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C110', 'Hannah Red', 'hannah.red@example.com', '1234567899', 'AC110', 'Dallas',
'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C111', 'Ian Gold', 'ian.gold@example.com', '1234567880', 'AC111', 'Orlando',
'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C112', 'Jasmine Pink', 'jasmine.pink@example.com', '1234567881', 'AC112', 'Las
Vegas', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C113', 'Kevin Silver', 'kevin.silver@example.com', '1234567882', 'AC113', 'San
Francisco', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C114', 'Laura Violet', 'laura.violet@example.com', '1234567883', 'AC114', 'Phoenix',
'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C115', 'Michael Orange', 'michael.orange@example.com', '1234567884', 'AC115',
'San Diego', 'USA');
```

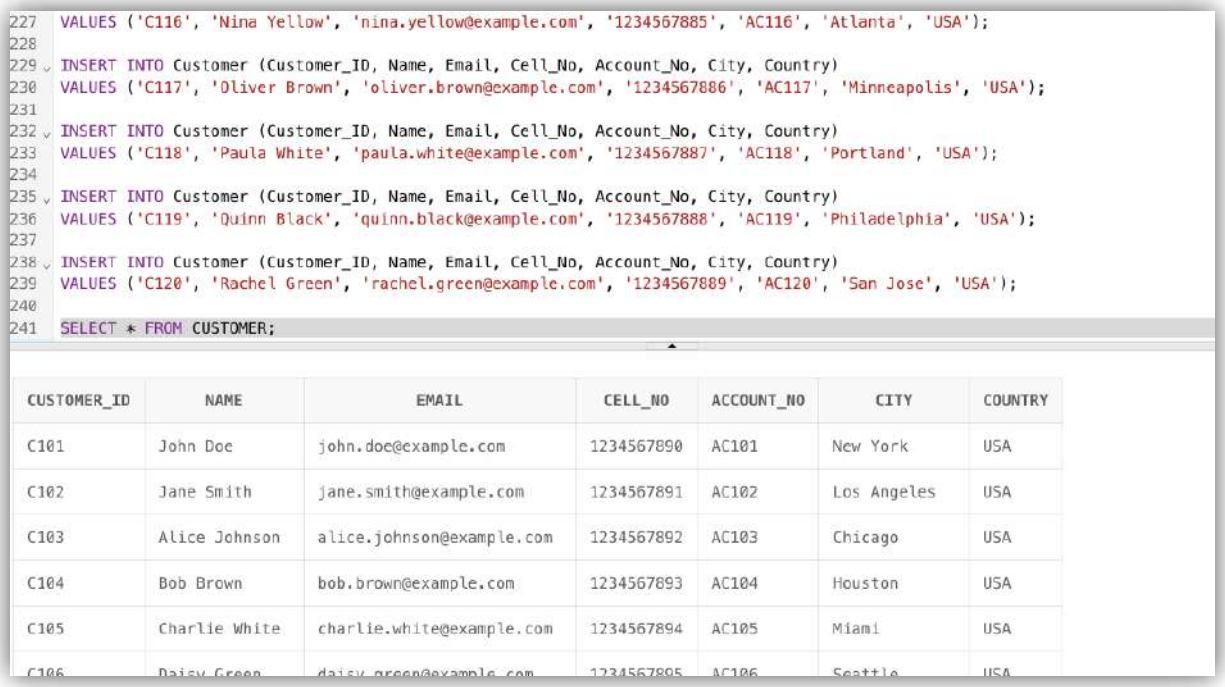
INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C116', 'Nina Yellow', 'nina.yellow@example.com', '1234567885', 'AC116', 'Atlanta', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C117', 'Oliver Brown', 'oliver.brown@example.com', '1234567886', 'AC117', 'Minneapolis', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C118', 'Paula White', 'paula.white@example.com', '1234567887', 'AC118', 'Portland', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C119', 'Quinn Black', 'quinn.black@example.com', '1234567888', 'AC119', 'Philadelphia', 'USA');

INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
VALUES ('C120', 'Rachel Green', 'rachel.green@example.com', '1234567889', 'AC120', 'San Jose', 'USA');



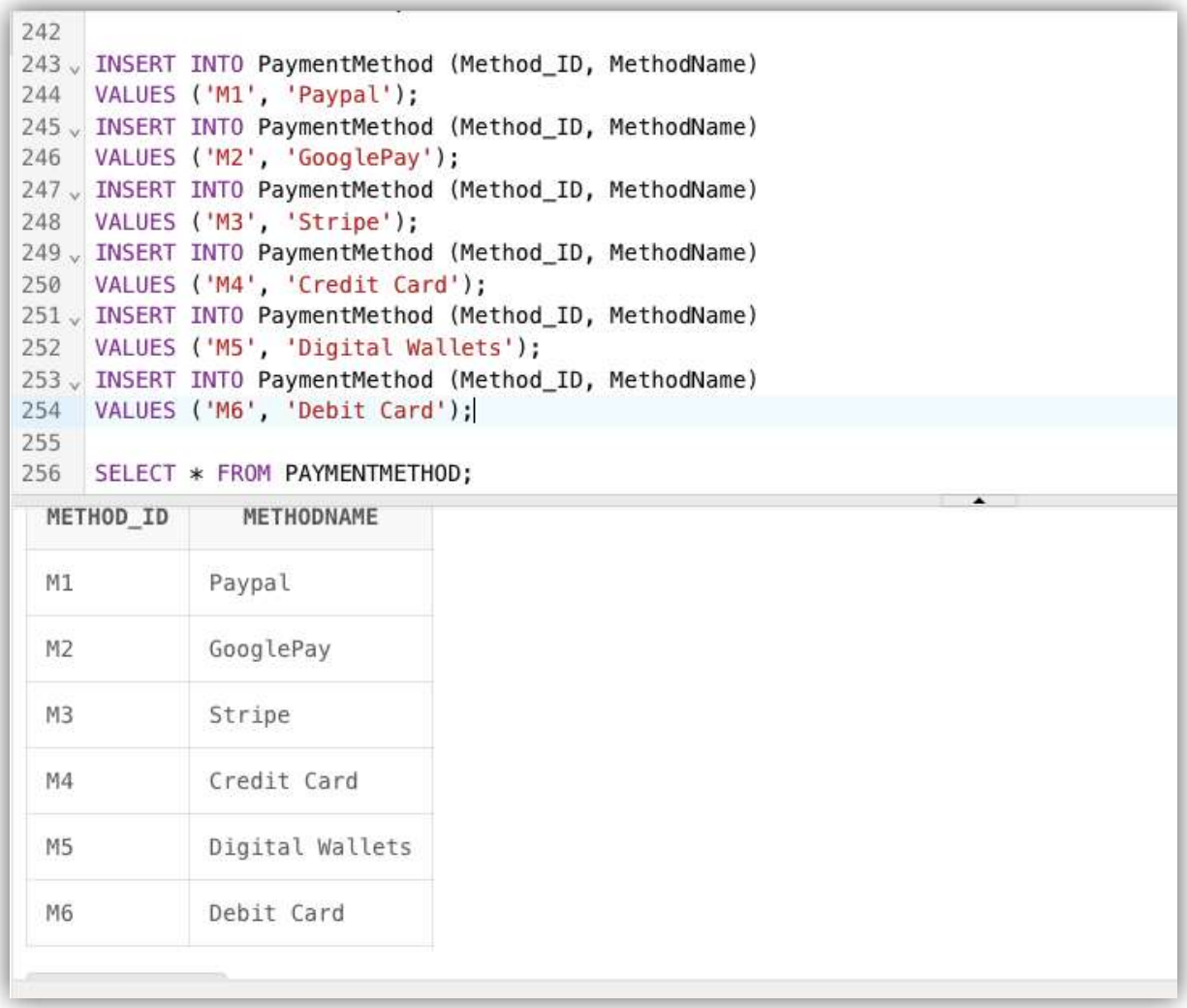
```
227 VALUES ('C116', 'Nina Yellow', 'nina.yellow@example.com', '1234567885', 'AC116', 'Atlanta', 'USA');
228
229 INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
230 VALUES ('C117', 'Oliver Brown', 'oliver.brown@example.com', '1234567886', 'AC117', 'Minneapolis', 'USA');
231
232 INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
233 VALUES ('C118', 'Paula White', 'paula.white@example.com', '1234567887', 'AC118', 'Portland', 'USA');
234
235 INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
236 VALUES ('C119', 'Quinn Black', 'quinn.black@example.com', '1234567888', 'AC119', 'Philadelphia', 'USA');
237
238 INSERT INTO Customer (Customer_ID, Name, Email, Cell_No, Account_No, City, Country)
239 VALUES ('C120', 'Rachel Green', 'rachel.green@example.com', '1234567889', 'AC120', 'San Jose', 'USA');
240
241 SELECT * FROM CUSTOMER;
```

| CUSTOMER_ID | NAME | EMAIL | CELL_NO | ACCOUNT_NO | CITY | COUNTRY |
|---|---|---|---|---|---|---|
| C101 | John Doe | john.doe@example.com | 1234567890 | AC101 | New York | USA |
| C102 | Jane Smith | jane.smith@example.com | 1234567891 | AC102 | Los Angeles | USA |
| C103 | Alice Johnson | alice.johnson@example.com | 1234567892 | AC103 | Chicago | USA |
| C104 | Bob Brown | bob.brown@example.com | 1234567893 | AC104 | Houston | USA |
| C105 | Charlie White | charlie.white@example.com | 1234567894 | AC105 | Miami | USA |
| C106 | Daisy Green | daisy.green@example.com | 1234567895 | AC106 | Seattle | USA |

**INSERT INTO PaymentMethod Table:**

INSERT INTO PaymentMethod (Method_ID, MethodName)
VALUES ('M1', 'Paypal');
INSERT INTO PaymentMethod (Method_ID, MethodName)

VALUES ('M2', 'GooglePay');
INSERT INTO PaymentMethod (Method_ID, MethodName)
VALUES ('M3', 'Stripe');
INSERT INTO PaymentMethod (Method_ID, MethodName)
VALUES ('M4', 'Credit Card');
INSERT INTO PaymentMethod (Method_ID, MethodName)
VALUES ('M5', 'Digital Wallets');
INSERT INTO PaymentMethod (Method_ID, MethodName)
VALUES ('M6', 'Debit Card');



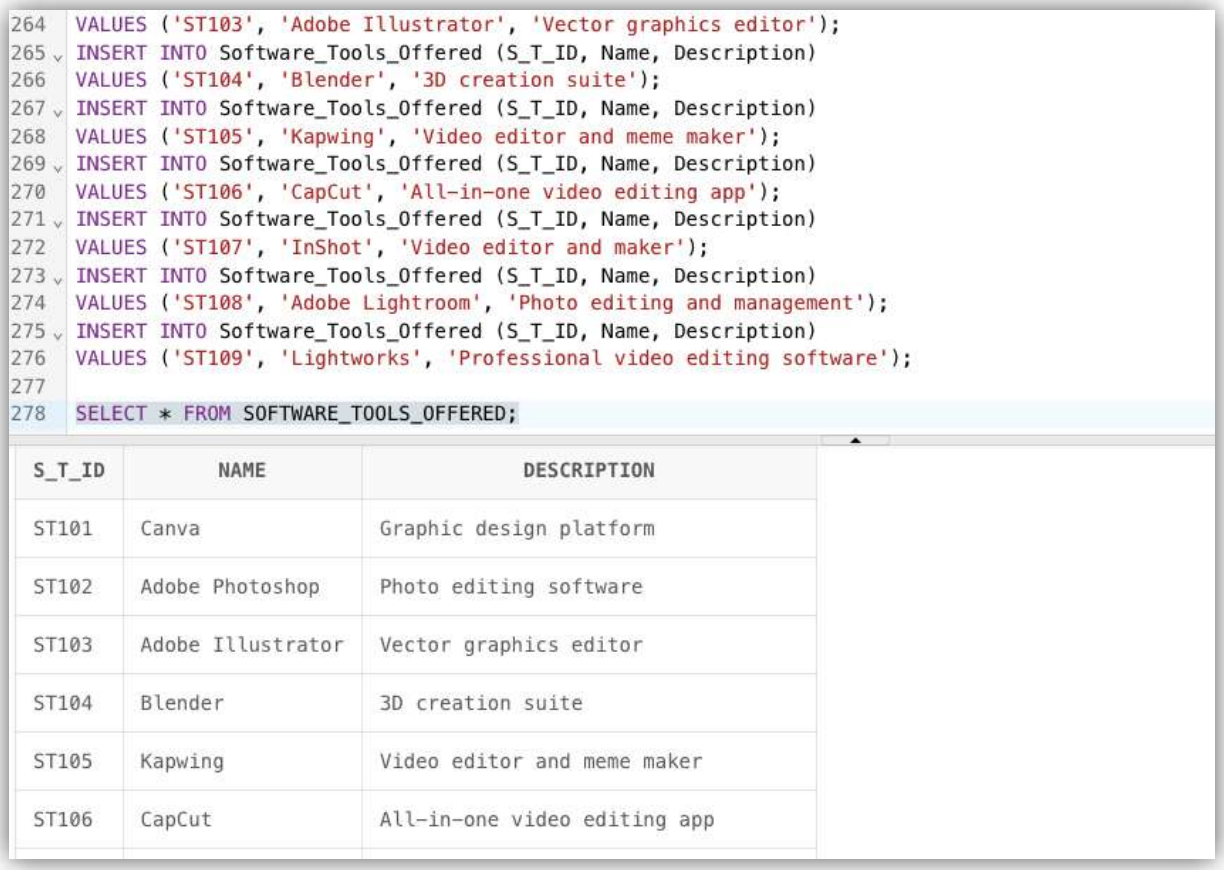
```
242
243 INSERT INTO PaymentMethod (Method_ID, MethodName)
244 VALUES ('M1', 'Paypal');
245 INSERT INTO PaymentMethod (Method_ID, MethodName)
246 VALUES ('M2', 'GooglePay');
247 INSERT INTO PaymentMethod (Method_ID, MethodName)
248 VALUES ('M3', 'Stripe');
249 INSERT INTO PaymentMethod (Method_ID, MethodName)
250 VALUES ('M4', 'Credit Card');
251 INSERT INTO PaymentMethod (Method_ID, MethodName)
252 VALUES ('M5', 'Digital Wallets');
253 INSERT INTO PaymentMethod (Method_ID, MethodName)
254 VALUES ('M6', 'Debit Card');
255
256 SELECT * FROM PAYMENTMETHOD;
```

| METHOD_ID | METHODNAME |
|---|---|
| M1 | Paypal |
| M2 | GooglePay |
| M3 | Stripe |
| M4 | Credit Card |
| M5 | Digital Wallets |
| M6 | Debit Card |

**INSERT INTO Software_Tools_Offered Table:**

INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST101', 'Canva', 'Graphic design platform');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST102', 'Adobe Photoshop', 'Photo editing software');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST103', 'Adobe Illustrator', 'Vector graphics editor');

```
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST104', 'Blender', '3D creation suite');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST105', 'Kapwing', 'Video editor and meme maker');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST106', 'CapCut', 'All-in-one video editing app');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST107', 'InShot', 'Video editor and maker');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST108', 'Adobe Lightroom', 'Photo editing and management');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST109', 'Lightworks', 'Professional video editing software');
```



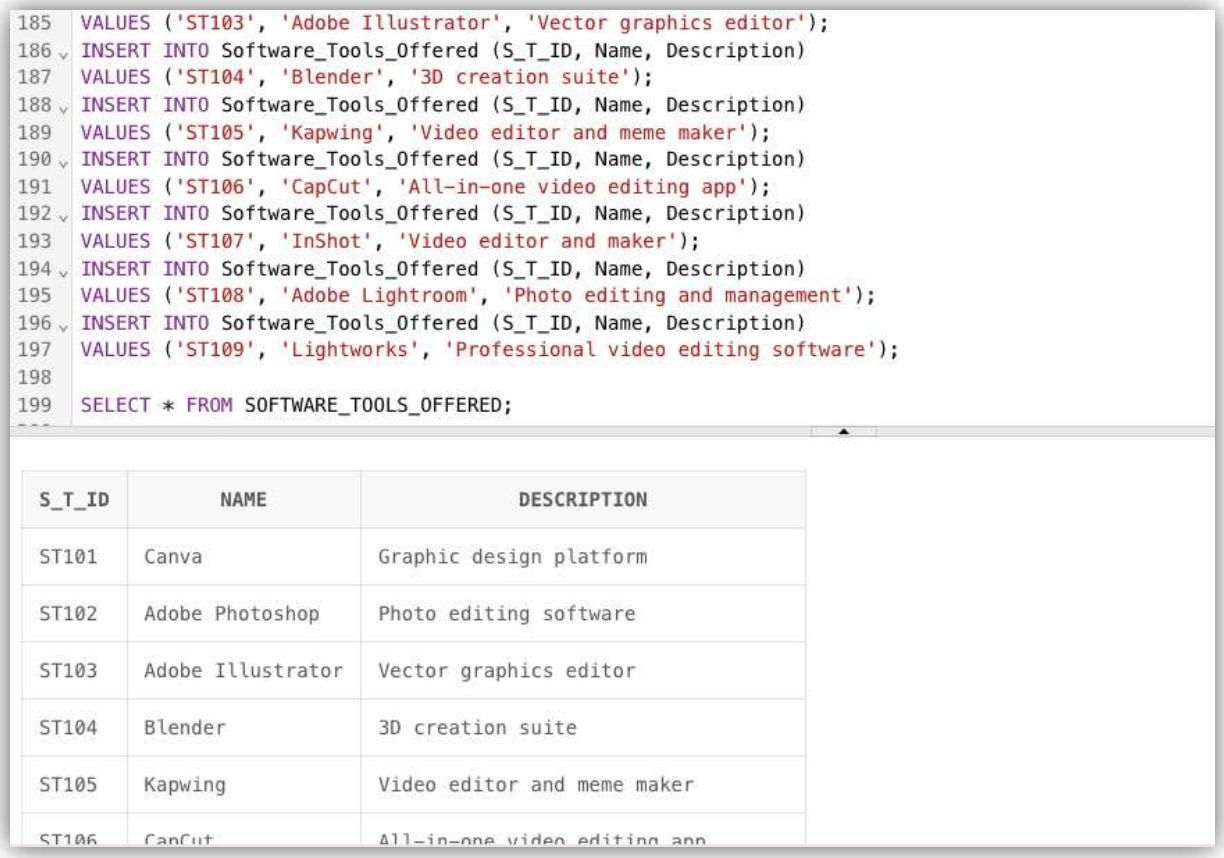
```
264  VALUES ('ST103', 'Adobe Illustrator', 'Vector graphics editor');
265  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
266  VALUES ('ST104', 'Blender', '3D creation suite');
267  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
268  VALUES ('ST105', 'Kapwing', 'Video editor and meme maker');
269  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
270  VALUES ('ST106', 'CapCut', 'All-in-one video editing app');
271  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
272  VALUES ('ST107', 'InShot', 'Video editor and maker');
273  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
274  VALUES ('ST108', 'Adobe Lightroom', 'Photo editing and management');
275  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
276  VALUES ('ST109', 'Lightworks', 'Professional video editing software');
277
278  SELECT * FROM SOFTWARE_TOOLS_OFFERED;
```

| S_T_ID | NAME | DESCRIPTION |
|---|---|---|
| ST101 | Canva | Graphic design platform |
| ST102 | Adobe Photoshop | Photo editing software |
| ST103 | Adobe Illustrator | Vector graphics editor |
| ST104 | Blender | 3D creation suite |
| ST105 | Kapwing | Video editor and meme maker |
| ST106 | CapCut | All-in-one video editing app |

**INSERT INTO Software_Tools_Offered Table:**

```
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST101', 'Canva', 'Graphic design platform');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST102', 'Adobe Photoshop', 'Photo editing software');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST103', 'Adobe Illustrator', 'Vector graphics editor');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST104', 'Blender', '3D creation suite');
```

```
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST105', 'Kapwing', 'Video editor and meme maker');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST106', 'CapCut', 'All-in-one video editing app');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST107', 'InShot', 'Video editor and maker');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST108', 'Adobe Lightroom', 'Photo editing and management');
INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
VALUES ('ST109', 'Lightworks', 'Professional video editing software');
```



```
185  VALUES ('ST103', 'Adobe Illustrator', 'Vector graphics editor');
186  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
187  VALUES ('ST104', 'Blender', '3D creation suite');
188  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
189  VALUES ('ST105', 'Kapwing', 'Video editor and meme maker');
190  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
191  VALUES ('ST106', 'CapCut', 'All-in-one video editing app');
192  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
193  VALUES ('ST107', 'InShot', 'Video editor and maker');
194  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
195  VALUES ('ST108', 'Adobe Lightroom', 'Photo editing and management');
196  INSERT INTO Software_Tools_Offered (S_T_ID, Name, Description)
197  VALUES ('ST109', 'Lightworks', 'Professional video editing software');
198
199  SELECT * FROM SOFTWARE_TOOLS_OFFERED;
```

| S_T_ID | NAME | DESCRIPTION |
|---|---|---|
| ST101 | Canva | Graphic design platform |
| ST102 | Adobe Photoshop | Photo editing software |
| ST103 | Adobe Illustrator | Vector graphics editor |
| ST104 | Blender | 3D creation suite |
| ST105 | Kapwing | Video editor and meme maker |
| ST106 | CapCut | All-in-one video editing app |

**INSERT INTO Subscriptions Table:**

```
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S101', 'C101', 'P1', 0, 1);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S102', 'C102', 'P2', 20, 3);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S103', 'C103', 'P3', 30, 6);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S104', 'C104', 'P1', 0, 1);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S105', 'C102', 'P2', 20, 3);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S106', 'C105', 'P3', 30, 6);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S107', 'C106', 'P1', 0, 1);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S108', 'C106', 'P2', 20, 3);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S109', 'C108', 'P3', 30, 6);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S110', 'C110', 'P1', 0, 1);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S111', 'C111', 'P2', 20, 3);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S112', 'C113', 'P3', 30, 6);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S113', 'C115', 'P1', 0, 1);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S114', 'C115', 'P2', 20, 3);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S115', 'C117', 'P3', 30, 6);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S116', 'C117', 'P1', 0, 1);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S117', 'C118', 'P2', 20, 3);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S118', 'C119', 'P3', 30, 6);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S119', 'C119', 'P1', 0, 1);
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S120', 'C120', 'P2', 20, 3);
```

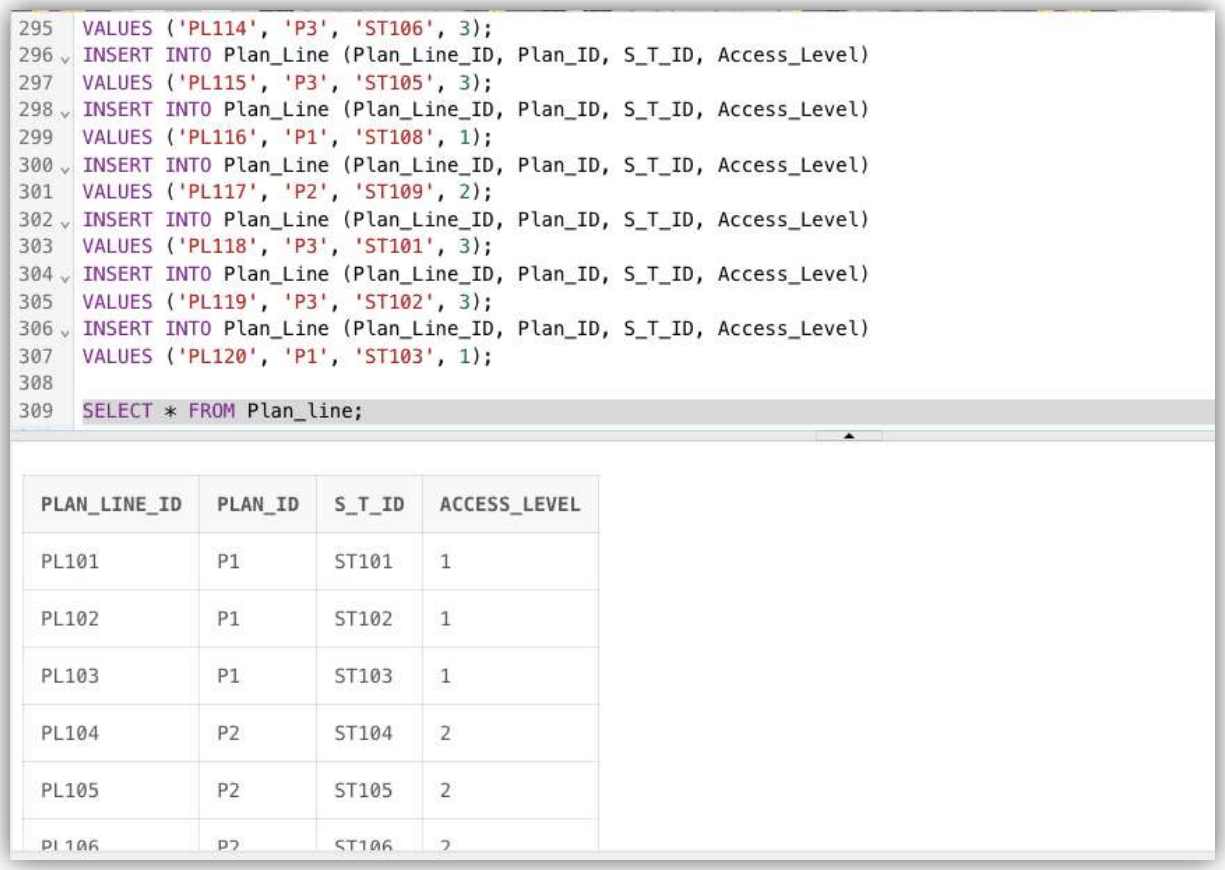
```
253 INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
254 VALUES ('S115', 'C117', 'P3', 30, 6);
255 INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
256 VALUES ('S116', 'C117', 'P1', 0, 1);
257 INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
258 VALUES ('S117', 'C118', 'P2', 20, 3);
259 INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
260 VALUES ('S118', 'C119', 'P3', 30, 6);
261 INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
262 VALUES ('S119', 'C119', 'P1', 0, 1);
263 INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
264 VALUES ('S120', 'C120', 'P2', 20, 3);
265
266 SELECT * FROM Subscription;
267
```

| SUBSCRIPTION_ID | CUSTOMER_ID | PLAN_ID | DISCOUNT | DURATION |
|---|---|---|---|---|
| S101 | C101 | P1 | 0 | 1 |
| S102 | C102 | P2 | 20 | 3 |
| S103 | C103 | P3 | 30 | 6 |
| S104 | C104 | P1 | 0 | 1 |
| S105 | C102 | P2 | 20 | 3 |

**INSERT INTO Plan_line Table:**

INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL101', 'P1', 'ST101', 1);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL102', 'P1', 'ST102', 1);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL103', 'P1', 'ST103', 1);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL104', 'P2', 'ST104', 2);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL105', 'P2', 'ST105', 2);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL106', 'P2', 'ST106', 2);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL107', 'P3', 'ST107', 3);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL108', 'P3', 'ST108', 3);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL109', 'P3', 'ST109', 3);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL110', 'P1', 'ST104', 1);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)

```
VALUES ('PL111', 'P2', 'ST102', 2);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL112', 'P2', 'ST103', 2);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL113', 'P1', 'ST107', 1);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL114', 'P3', 'ST106', 3);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL115', 'P3', 'ST105', 3);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL116', 'P1', 'ST108', 1);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL117', 'P2', 'ST109', 2);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL118', 'P3', 'ST101', 3);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL119', 'P3', 'ST102', 3);
INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
VALUES ('PL120', 'P1', 'ST103', 1);
```



```
295 VALUES ('PL114', 'P3', 'ST106', 3);
296 INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
297 VALUES ('PL115', 'P3', 'ST105', 3);
298 INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
299 VALUES ('PL116', 'P1', 'ST108', 1);
300 INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
301 VALUES ('PL117', 'P2', 'ST109', 2);
302 INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
303 VALUES ('PL118', 'P3', 'ST101', 3);
304 INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
305 VALUES ('PL119', 'P3', 'ST102', 3);
306 INSERT INTO Plan_Line (Plan_Line_ID, Plan_ID, S_T_ID, Access_Level)
307 VALUES ('PL120', 'P1', 'ST103', 1);
308
309 SELECT * FROM Plan_line;
```

| PLAN_LINE_ID | PLAN_ID | S_T_ID | ACCESS_LEVEL |
|---|---|---|---|
| PL101 | P1 | ST101 | 1 |
| PL102 | P1 | ST102 | 1 |
| PL103 | P1 | ST103 | 1 |
| PL104 | P2 | ST104 | 2 |
| PL105 | P2 | ST105 | 2 |
| PL106 | P2 | ST106 | 2 |

**INSERT INTO Payment Table:**

```
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY101', 'S101', 'M1', 10.00, TO_DATE('2024-01-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY102', 'S102', 'M2', 48.00, TO_DATE('2024-02-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY103', 'S103', 'M3', 126.00, TO_DATE('2024-03-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY104', 'S104', 'M4', 10.00, TO_DATE('2024-04-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY105', 'S105', 'M5', 48.00, TO_DATE('2024-05-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY106', 'S106', 'M6', 126.00, TO_DATE('2024-06-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY107', 'S107', 'M1', 10.00, TO_DATE('2024-07-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY108', 'S108', 'M2', 48.00, TO_DATE('2024-08-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY109', 'S109', 'M3', 126.00, TO_DATE('2024-09-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY110', 'S110', 'M4', 10.00, TO_DATE('2024-10-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY111', 'S111', 'M5', 48.00, TO_DATE('2024-11-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY112', 'S112', 'M6', 126.00, TO_DATE('2024-12-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY113', 'S113', 'M1', 10.00, TO_DATE('2025-01-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY114', 'S114', 'M2', 48.00, TO_DATE('2025-02-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY115', 'S115', 'M3', 126.00, TO_DATE('2025-03-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY116', 'S116', 'M4', 10.00, TO_DATE('2025-04-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY117', 'S117', 'M5', 48.00, TO_DATE('2025-05-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY118', 'S118', 'M6', 126.00, TO_DATE('2025-06-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY119', 'S119', 'M1', 10.00, TO_DATE('2025-07-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY120', 'S120', 'M2', 48.00, TO_DATE('2025-08-01', 'YYYY-MM-DD'));
```

```
VALUES ('PY114', 'S114', 'M2', 48.00, TO_DATE('2025-02-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY115', 'S115', 'M3', 126.00, TO_DATE('2025-03-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY116', 'S116', 'M4', 10.00, TO_DATE('2025-04-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY117', 'S117', 'M5', 48.00, TO_DATE('2025-05-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY118', 'S118', 'M6', 126.00, TO_DATE('2025-06-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY119', 'S119', 'M1', 10.00, TO_DATE('2025-07-01', 'YYYY-MM-DD'));
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY120', 'S120', 'M2', 48.00, TO_DATE('2025-08-01', 'YYYY-MM-DD'));

SELECT * FROM PAYMENT;
```

| PAYMENT_ID | SUBSCRIPTION_ID | METHOD_ID | AMOUNT | PAYMENT_DATE |
|---|---|---|---|---|
| PY101 | S101 | M1 | 10 | 01-JAN-24 |
| PY102 | S102 | M2 | 48 | 01-FEB-24 |
| PY103 | S103 | M3 | 126 | 01-MAR-24 |
| PY104 | S104 | M4 | 10 | 01-APR-24 |
| PY105 | S105 | M5 | 48 | 01-MAY-24 |
| PY106 | S106 | M6 | 126 | 01-JUN-24 |

## Queries:

### 1. Query to retrieve all subscriptions with customer details

```
SELECT
   C.Name AS Customer_Name,
   C.Email AS Customer_Email,
   P.PlanName AS Subscription_Plan,
   S.Discount AS Subscription_Discount,
   S.Duration AS Subscription_Duration
FROM
   Subscription S
JOIN
   Customer C ON S.Customer_ID = C.Customer_ID
JOIN
   Plan P ON S.Plan_ID = P.Plan_ID
ORDER BY
   S.Subscription_ID;
```

```
354
355 SELECT
356     C.Name AS Customer_Name,
357     C.Email AS Customer_Email,
358     P.PlanName AS Subscription_Plan,
359     S.Discount AS Subscription_Discount,
360     S.Duration AS Subscription_Duration
361 FROM
362     Subscription S
363 JOIN
364     Customer C ON S.Customer_ID = C.Customer_ID
365 JOIN
366     Plan P ON S.Plan_ID = P.Plan_ID
367 ORDER BY
368     S.Subscription_ID;
```

| CUSTOMER_NAME | CUSTOMER_EMAIL | SUBSCRIPTION_PLAN | SUBSCRIPTION_DISCOUNT | SUBSCRIPTION_DURATION |
|---|---|---|---|---|
| John Doe | john.doe@example.com | Basic | 0 | 1 |
| Jane Smith | jane.smith@example.com | Intermediate | 20 | 3 |
| Alice Johnson | alice.johnson@example.com | Pro | 30 | 6 |
| Bob Brown | bob.brown@example.com | Basic | 0 | 1 |
| Jane Smith | jane.smith@example.com | Intermediate | 20 | 3 |
| Charlie White | charlie.white@example.com | Pro | 30 | 6 |

## 2. Query to list all software tools offered under each subscription plan

```
SELECT
   P.PlanName AS Subscription_Plan,
   ST.Name AS Software_Tool,
   ST.Description AS Tool_Description,
   PL.Access_Level AS Tool_Access_Level
FROM
   Plan_Line PL
JOIN
   Plan P ON PL.Plan_ID = P.Plan_ID
JOIN
   Software_Tools_Offered ST ON PL.S_T_ID = ST.S_T_ID
ORDER BY
   P.PlanName, PL.Plan_Line_ID;
```

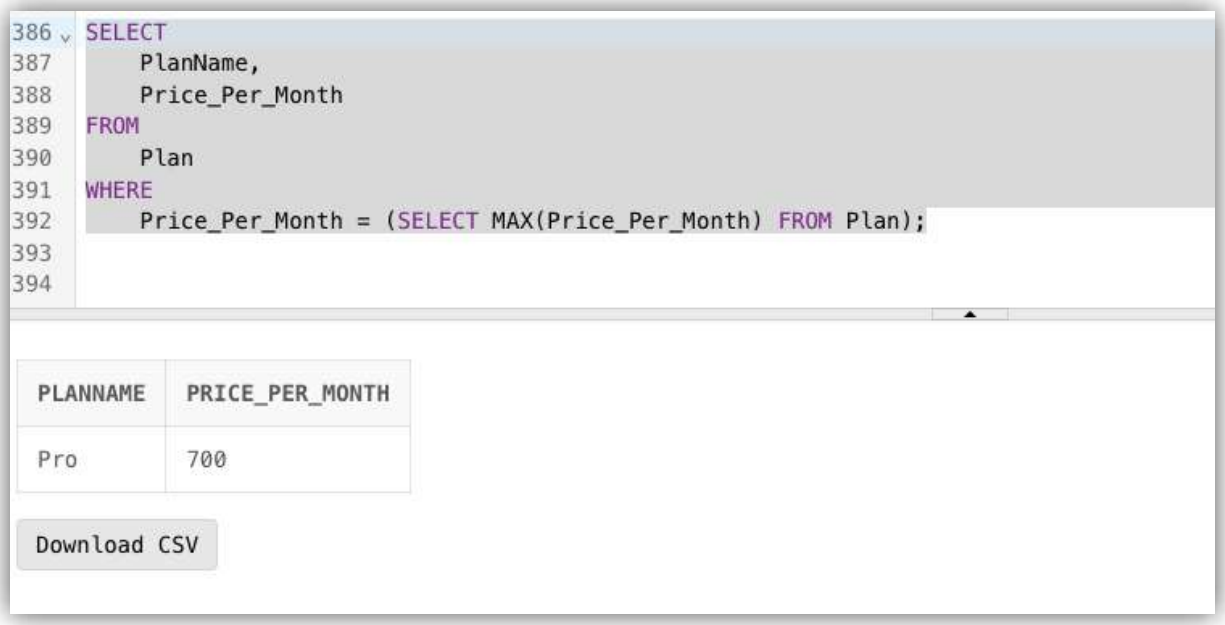
```
370
371  SELECT
372      P.PlanName AS Subscription_Plan,
373      ST.Name AS Software_Tool,
374      ST.Description AS Tool_Description,
375      PL.Access_Level AS Tool_Access_Level
376  FROM
377      Plan_Line PL
378  JOIN
379      Plan P ON PL.Plan_ID = P.Plan_ID
380  JOIN
381      Software_Tools_Offered ST ON PL.S_T_ID = ST.S_T_ID
382  ORDER BY
383      P.PlanName, PL.Plan_Line_ID;
384
```

| SUBSCRIPTION_PLAN | SOFTWARE_TOOL | TOOL_DESCRIPTION | TOOL_ACCESS_LEVEL |
|---|---|---|---|
| Basic | Canva | Graphic design platform | 1 |
| Basic | Adobe Photoshop | Photo editing software | 1 |
| Basic | Adobe Illustrator | Vector graphics editor | 1 |
| Basic | Blender | 3D creation suite | 1 |
| Basic | InShot | Video editor and maker | 1 |
| Basic | Adobe Lightroom | Photo editing and management | 1 |

| | | | |
|---|---|---|---|
| Basic | Adobe Lightroom | Photo editing and management | 1 |
| Basic | Adobe Illustrator | Vector graphics editor | 1 |
| Intermediate | Blender | 3D creation suite | 2 |
| Intermediate | Kapwing | Video editor and meme maker | 2 |
| Intermediate | CapCut | All-in-one video editing app | 2 |
| Intermediate | Adobe Photoshop | Photo editing software | 2 |
| Intermediate | Adobe Illustrator | Vector graphics editor | 2 |

### 3. Query to find the most expensive plan:

```
SELECT
   PlanName,
   Price_Per_Month
FROM
   Plan
WHERE
   Price_Per_Month = (SELECT MAX(Price_Per_Month) FROM Plan);
```



```
SELECT
    PlanName,
    Price_Per_Month
FROM
    Plan
WHERE
    Price_Per_Month = (SELECT MAX(Price_Per_Month) FROM Plan);
```

| PLANNAME | PRICE_PER_MONTH |
|---|---|
| Pro | 700 |

Download CSV

4. **Query to retrieve all customers who subscribed to Pro plan**

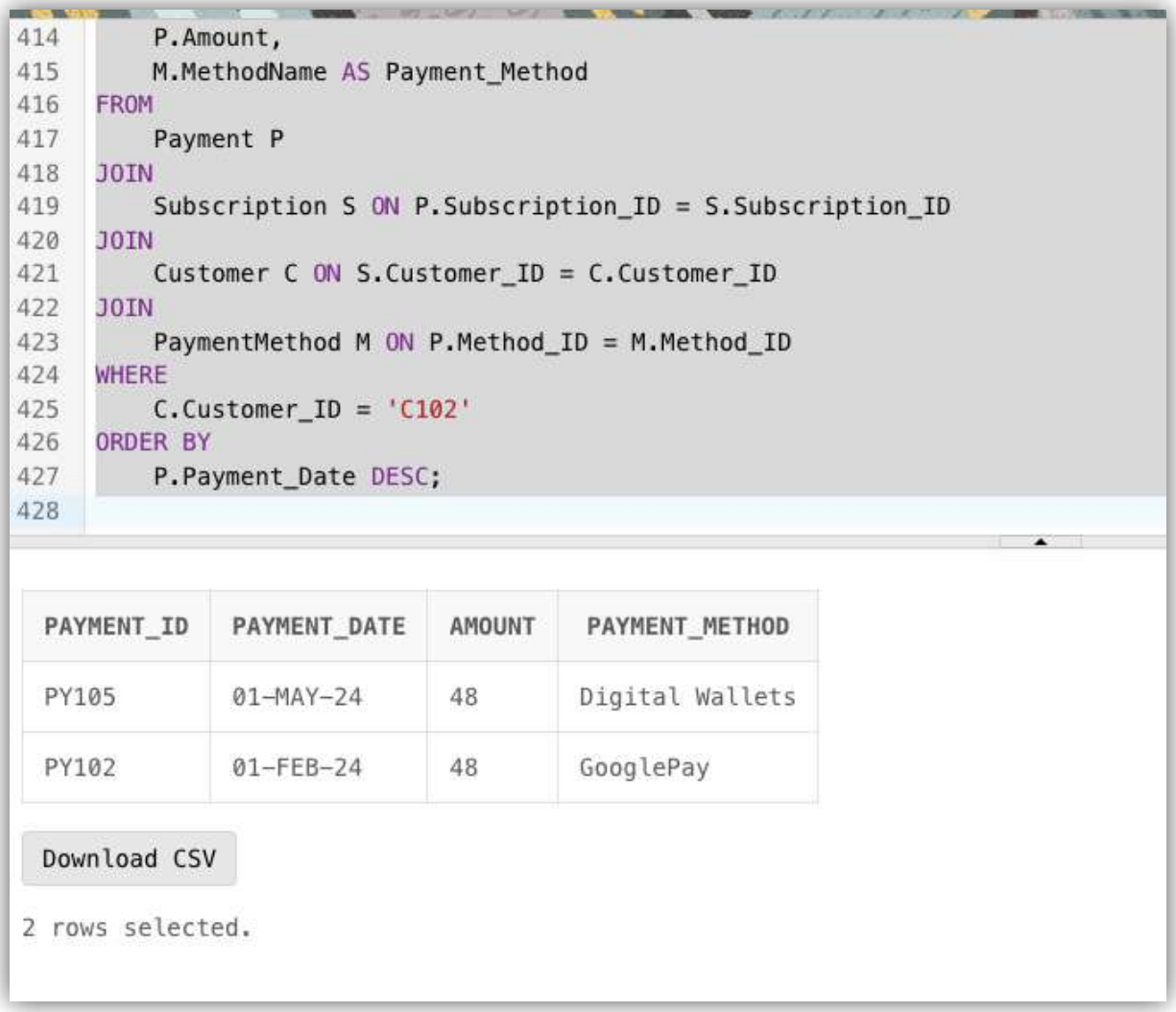
```
SELECT
   C.Name AS Customer_Name,
   C.Email AS Customer_Email,
   C.City AS Customer_City,
   C.Country AS Customer_Country
FROM
   Subscription S
JOIN
   Customer C ON S.Customer_ID = C.Customer_ID
JOIN
   Plan P ON S.Plan_ID = P.Plan_ID
WHERE
   P.PlanName = 'Pro';
```

```sql
SELECT
    C.Name AS Customer_Name,
    C.Email AS Customer_Email,
    C.City AS Customer_City,
    C.Country AS Customer_Country
FROM
    Subscription S
JOIN
    Customer C ON S.Customer_ID = C.Customer_ID
JOIN
    Plan P ON S.Plan_ID = P.Plan_ID
WHERE
    P.PlanName = 'Pro';
```

| CUSTOMER_NAME | CUSTOMER_EMAIL | CUSTOMER_CITY | CUSTOMER_COUNTRY |
|---|---|---|---|
| Alice Johnson | alice.johnson@example.com | Chicago | USA |
| Charlie White | charlie.white@example.com | Miami | USA |
| Fiona Grey | fiona.grey@example.com | Boston | USA |
| Kevin Silver | kevin.silver@example.com | San Francisco | USA |
| Oliver Brown | oliver.brown@example.com | Minneapolis | USA |
| Quinn Black | quinn.black@example.com | Philadelphia | USA |

## 5. Query to find payment history for a customer C 102

```sql
SELECT
   P.Payment_ID,
   P.Payment_Date,
   P.Amount,
   M.MethodName AS Payment_Method
FROM
   Payment P
JOIN
   Subscription S ON P.Subscription_ID = S.Subscription_ID
JOIN
   Customer C ON S.Customer_ID = C.Customer_ID
JOIN
   PaymentMethod M ON P.Method_ID = M.Method_ID
WHERE
   C.Customer_ID = 'C102'
ORDER BY
   P.Payment_Date DESC;
```



```
414     P.Amount,
415     M.MethodName AS Payment_Method
416 FROM
417     Payment P
418 JOIN
419     Subscription S ON P.Subscription_ID = S.Subscription_ID
420 JOIN
421     Customer C ON S.Customer_ID = C.Customer_ID
422 JOIN
423     PaymentMethod M ON P.Method_ID = M.Method_ID
424 WHERE
425     C.Customer_ID = 'C102'
426 ORDER BY
427     P.Payment_Date DESC;
428
```

| PAYMENT_ID | PAYMENT_DATE | AMOUNT | PAYMENT_METHOD |
|---|---|---|---|
| PY105 | 01-MAY-24 | 48 | Digital Wallets |
| PY102 | 01-FEB-24 | 48 | GooglePay |

Download CSV

2 rows selected.

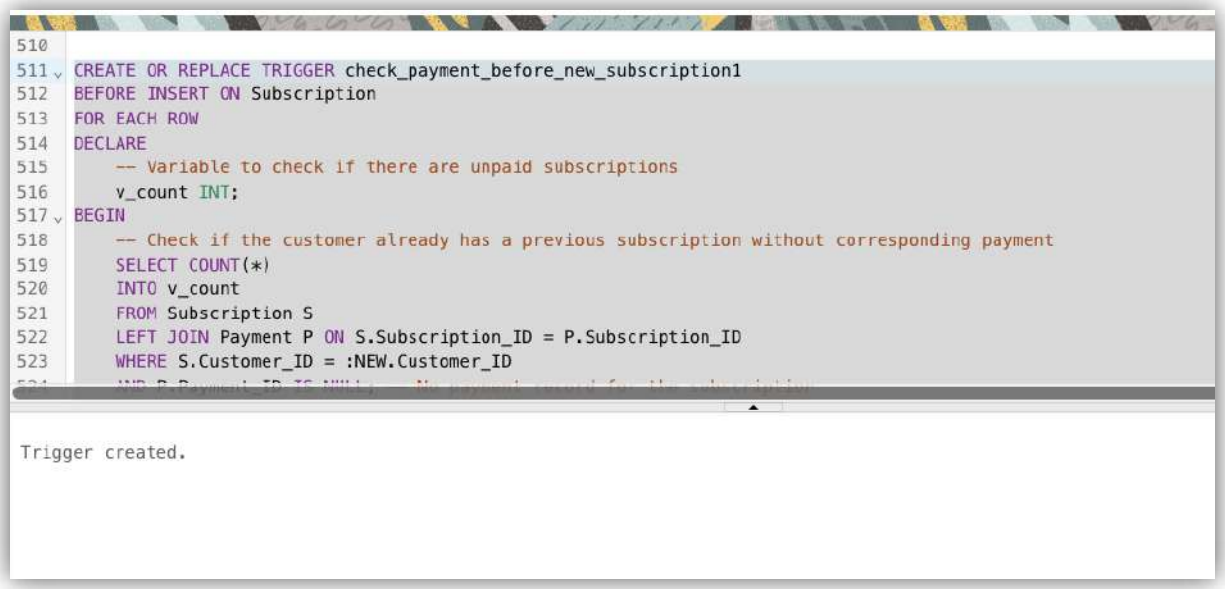
## 6. Trigger Creation and Testing:

I have created a trigger that ensures a customer cannot purchase a new subscription if they have an existing subscription that has not been paid. The trigger will check if there is a corresponding payment record for the customer's previous subscription. If no payment is found, the new subscription will not be allowed to be inserted.

## Trigger Code:

```
CREATE OR REPLACE TRIGGER check_payment_before_new_subscription1
BEFORE INSERT ON Subscription
FOR EACH ROW
DECLARE
   -- Variable to check if there are unpaid subscriptions
   v_count INT;
BEGIN
   -- Check if the customer already has a previous subscription without corresponding payment
   SELECT COUNT(*)
```

```
    INTO v_count
    FROM Subscription S
    LEFT JOIN Payment P ON S.Subscription_ID = P.Subscription_ID
    WHERE S.Customer_ID = :NEW.Customer_ID
    AND P.Payment_ID IS NULL; -- No payment record for the subscription

    -- If there are unpaid subscriptions, raise an error and prevent the insertion of the new
   subscription
    IF v_count > 0 THEN
        RAISE_APPLICATION_ERROR(-20001, 'Customer has an unpaid subscription. Cannot
   purchase a new subscription until the previous one is paid.');
    END IF;
END check_payment_before_new_subscription;
/
```




**Testing above trigger:**

**Inserting a record that had not had subscription previously: (allowed)**

```
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S121', 'C107', 'P1', 0, 1);
```

**C107 trying to buy another subscription without paying for the previous subscription: (denied)**

```
INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S125', 'C107', 'P1', 0, 1);
```

```
526     -- If there are unpaid subscriptions, raise an error and prevent the insertion of the new subscription
527     IF v_count > 0 THEN
528         RAISE_APPLICATION_ERROR(-20001, 'Customer has an unpaid subscription. Cannot purchase a new subscription until the previo
529     END IF;
530 END check_payment_before_new_subscription;
531 /
532
533
534 -- Insert a new subscription for the customer C107 without payment
535 INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
536 VALUES ('S125', 'C107', 'P1', 0, 1);
537
```

```
ORA-20001: Customer has an unpaid subscription. Cannot purchase a new subscription until the previous one is paid. ORA-06512: at
"SQL_RTRTOHGAGVQMFROJNNHSEJCCU.CHECK_PAYMENT_BEFORE_NEW_SUBSCRIPTION1", line 15
ORA-06512: at "SYS.DBMS_SQL", line 1721

More Details: https://docs.oracle.com/error-help/db/ora-20001
```

## Insert a payment for the previous subscription (S123) for the customer C107 first: (allowed)

INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY121', 'S123', 'M1', 10.00, TO_DATE('2024-01-01', 'YYYY-MM-DD'));

```
548
549 SELECT * FROM PAYMENT;
550
551 INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
552 VALUES ('PY121', 'S123', 'M1', 10.00, TO_DATE('2024-01-01', 'YYYY-MM-DD'));
553
554
555
```

```
1 row(s) inserted.
```

## Try inserting again another subscription 125 after making the payment BEGIN for C107: (now allowed)

INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
VALUES ('S125', 'C107', 'P3', 20, 6);

```
550
551 INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
552 VALUES ('PY121', 'S123', 'M1', 10.00, TO_DATE('2024-01-01', 'YYYY-MM-DD'));
553 SELECT * FROM SUBSCRIPTION;
554
555
556 INSERT INTO Subscription (Subscription_ID, Customer_ID, Plan_ID, Discount, Duration)
557 VALUES ('S125', 'C107', 'P3', 20, 6);
558
```

```
1 row(s) inserted.
```

So the trigger is tested accurately.

## 7. **Trigger to Prevent Negative Payment Amounts:**

```
CREATE OR REPLACE TRIGGER prevent_negative_payment
BEFORE INSERT ON Payment
FOR EACH ROW
BEGIN
   -- Check if the payment amount is negative
   IF :NEW.Amount < 0 THEN
      RAISE_APPLICATION_ERROR(-20004, 'Payment amount cannot be negative.');
   END IF;
END;
/
```


## **Testing Trigger:**

```
INSERT INTO Payment (Payment_ID, Subscription_ID, Method_ID, Amount, Payment_Date)
VALUES ('PY123', 'S127', 'M1', -20.00, TO_DATE('2024-01-06', 'YYYY-MM-DD'));
```

## 8. Update a Record Transaction:

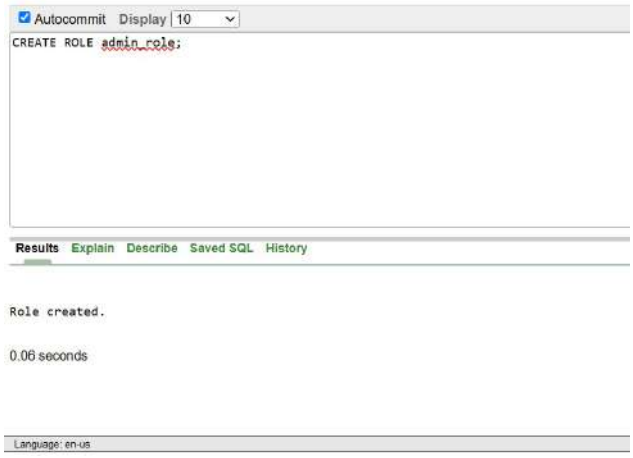
```
BEGIN
  UPDATE Payment
  SET Amount = Amount + 10
  WHERE Payment_ID = 'PY123';

  COMMIT;

  DBMS_OUTPUT.PUT_LINE('Payment amount updated successfully.');
END;
```

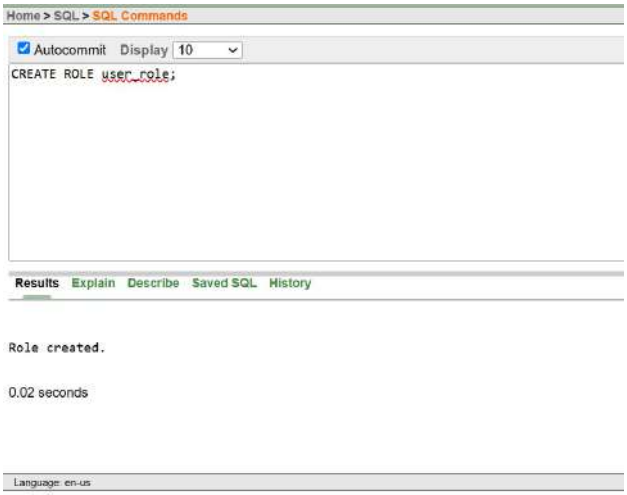
```
683  BEGIN
684    -- Start transaction
685    UPDATE Payment
686    SET Amount = Amount + 10
687    WHERE Payment_ID = 'PY123';
688
689    -- Commit the transaction if the update is successful
690    COMMIT;
691
692    DBMS_OUTPUT.PUT_LINE('Payment amount updated successfully.');
693  END;
694
```

```
Statement processed.
Payment amount updated successfully.
```

## 9. Creating Role:

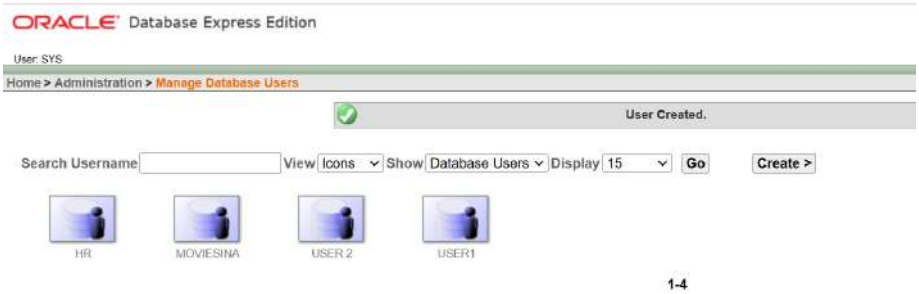
```
CREATE ROLE admin_role;

CREATE ROLE user_role;
```






## 10. Creating Users:

Creating users user1 and user2 as we created a pvfc account.





### 11. Granting Roles to Users:

```
GRANT admin_role TO user1;
GRANT user_role TO user2;
```

### 12. Grant System Privileges:

```
GRANT CREATE SESSION TO admin_role;
GRANT CREATE TABLE TO admin_role;
GRANT CREATE VIEW TO admin_role;
GRANT SELECT ON employees TO user_role;
```

### 13. Grant Object Privileges:

```
GRANT SELECT, INSERT, UPDATE ON Subscription TO user1;
GRANT SELECT ON Subscription TO user2;
```

### 14. Revoke Roles from Users:

```
REVOKE user_role FROM user2;
```

### 15. Revoke System Privileges:

```
REVOKE CREATE SESSION FROM user1;
REVOKE CREATE TABLE FROM admin_role;
```

### 16. Grant All Privileges on a Table to a User:

```
GRANT ALL PRIVILEGES ON Payment TO user1;
```

### 17. View Table Permissions Granted by You:

```
SELECT * FROM user_tab_privs_made
```

### 18. View Table Permissions Granted to You:

```
SELECT * FROM user_tab_privs_recd
```

### 19. View Column-Level Privileges Granted to You:

```
SELECT * FROM user_col_privs_recd
```

### 20. View Column-Level Privileges Granted by You

SELECT * FROM user_col_privs_made

### 21. View Privileges Granted to a Specific Role

SELECT * FROM role_sys_privs

## Size Estimation of Rows, Tables and DB:

Below are the common sizes for the data types:

1. **INT**: 4 bytes
2. **VARCHAR(n)**: Size varies based on the length of the string; for estimation, we use the maximum size (n bytes + 1 byte for length storage).
3. **DECIMAL(p, s)**: Approximately (p/2 + 1) bytes.
4. **DATE**: 3 bytes
5. **CLOB**: Variable size, let's assume an average of 500 bytes for estimation.

### Attribute Size Calculation:

### Customer Table:

| Attribute | Data Type | Size (Bytes) |
|---|---|---|
| Customer_ID | VARCHAR(10) | 11 |
| Name | VARCHAR(100) | 101 |
| Email | VARCHAR(100) | 101 |
| Cell_No | VARCHAR(15) | 16 |
| Account_No | VARCHAR(20) | 21 |
| City | VARCHAR(20) | 21 |
| Country | VARCHAR(20) | 21 |

**Single Record Size**: 11+101+101+16+21+21+21=292 bytes
**Total Size (20 rows)**: 292×20=5,840 bytes

### Plan Table:

| Attribute | Data Type | Size (Bytes) |
|---|---|---|
| Plan_ID | VARCHAR(10) | 11 |
| PlanName | VARCHAR(100) | 101 |
| Price_Per_Month | DECIMAL(10, 2) | 6 |

**Single Record Size**: 11+101+6=118 bytes
**Total Size (3 rows)**: 118×3=354 bytes

### Subscription Table:

| Attribute | Data Type | Size (Bytes) |
|---|---|---|
| Subscription_ID | VARCHAR(10) | 11 |
| Customer_ID | VARCHAR(10) | 11 |
| Plan_ID | VARCHAR(10) | 11 |
| Discount | INT | 4 |

| | | |
|---|---|---|
| Duration | INT | 4 |

**Single Record Size**: 11+11+11+4+4=41 bytes
**Total Size (20 rows)**: 41×20=820 bytes

## Software_Tools_Offered Table

| Attribute | Data Type | Size (Bytes) |
|---|---|---|
| S_T_ID | VARCHAR(10) | 11 |
| Name | VARCHAR(100) | 101 |
| Description | CLOB | 500 |

**Single Record Size**: 11+101+500=612 bytes
**Total Size (9 rows)**: 612×9=5,508 bytes

## Plan_Line Table:

| Attribute | Data Type | Size (Bytes) |
|---|---|---|
| Plan_Line_ID | VARCHAR(10) | 11 |
| Plan_ID | VARCHAR(10) | 11 |
| S_T_ID | VARCHAR(10) | 11 |
| Access_Level | INT | 4 |

**Single Record Size**: 11+11+11+4=37 bytes
**Total Size (20 rows)**: 37×20=740 bytes

## PaymentMethod Table:

| Attribute | Data Type | Size (Bytes) |
|---|---|---|
| Method_ID | VARCHAR(10) | 11 |
| MethodName | VARCHAR(100) | 101 |

**Single Record Size**: 11+101=112 bytes
**Total Size (6 rows)**: 112×6=672 bytes

## Payment Table:

| Attribute | Data Type | Size (Bytes) |
|---|---|---|
| Payment_ID | VARCHAR(10) | 11 |
| Subscription_ID | VARCHAR(10) | 11 |
| Method_ID | VARCHAR(10) | 11 |
| Amount | DECIMAL(10, 2) | 6 |
| Payment_Date | DATE | 3 |

**Single Record Size**: 11+11+11+6+3=42 bytes
**Total Size (20 rows)**: 42×20=840 bytes

## Final Table Sizes:

| Table | Single Record Size (Bytes) | Total Records | Total Size (Bytes) |
|---|---|---|---|
| Customer | 292 | 20 | 5,840 |
| Plan | 118 | 3 | 354 |
| Subscription | 41 | 20 | 820 |
| Software_Tools_Offered | 612 | 9 | 5,508 |
| Plan_Line | 37 | 20 | 740 |
| PaymentMethod | 112 | 6 | 672 |
| Payment | 42 | 20 | 840 |

## Total Database Size:

**Current DB Size**: 840+354+820+5,508+740+672+840 = 14,774 bytes (14.77 KB

## Maximum Database Size (Long Run):

Assuming all VARCHAR fields are fully utilized:

1. **Customer Table**: 292×1,000,000 = 292MB
2. **Plan Table**: 118×100,000 = 11.8MB
3. **Subscription Table**: 41×1,000,000 = 41MB
4. **Software_Tools_Offered**: 612×1,000,000 = 612MB
5. **Plan_Line Table**: 37×1,000,000 = 37MB
6. **PaymentMethod**: 112×100,000 = 11.2MB
7. **Payment Table**: 42×1,000,000 = 42MB

**Maximum DB Size**: 1,047 MB (approximately 1.05 GB).