Clothing E-commerce

SQL Database

Group 3

Krishna Yashwanth Tummala

Rituja Vishwanath Mahajan

Vinotha Subramaniyan

Maitreya Dayanand Babar

Project Objectives

- Create a SQL database for multiple small business clothing vendors to offer their products
- Provide a safe and secure platform for customers to browse and purchase clothes
- Provide a platform for customers to give their feedback on purchased products
- Provide a platform for vendors to connect with shipping contractors

Project Design

MARKET ANALYSIS

Gives an idea about the best-selling products and the view segments which showcases the growth of the business.

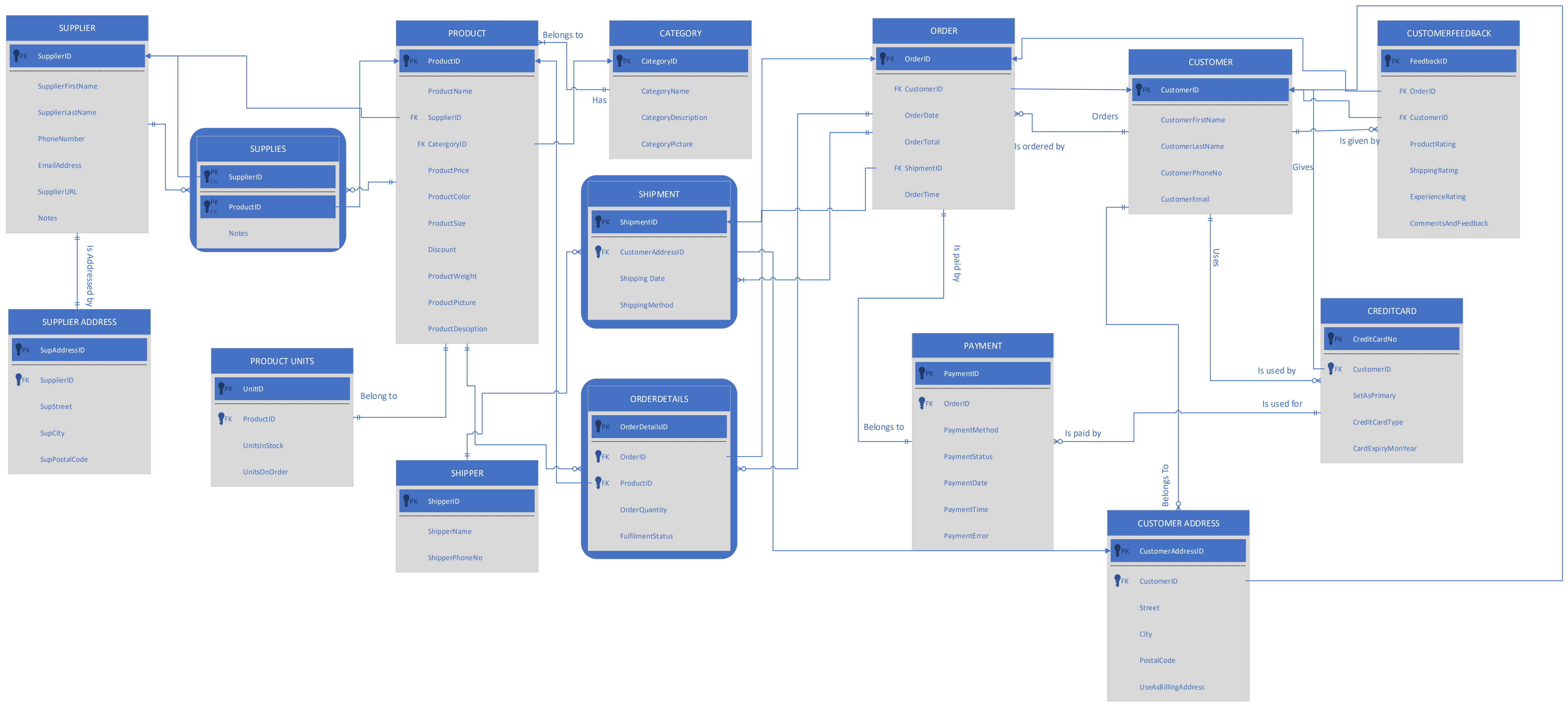
TECHNICAL ANALYSIS

To maintain data integrity and derive unique insights from customer behavior for further analysis

FINANCIAL ANALYSIS

The data shared between departments involves a deeper analysis than simple accounting work.

E-R Diagram


SUPPLIER
PK SupplierID
SupplierFirstName
SupplierLastName
PhoneNumber
EmailAddress
SupplierURL
Notes
Is Addressed by
SUPPLIER ADDRESS
PK SupAddressID
FK SupplierID
SupStreet
SupCity
SupPostalCode
SUPPLIES
PK FK SupplierID
PK FK ProductID
Notes
PRODUCT UNITS
PK UnitID
FK ProductID
UnitsInStock
UnitsOnOrder
Belong to
PRODUCT
PK ProductID
ProductName
FK SupplierID
FK CatergoryID
ProductPrice
ProductColor
ProductSize
Discount
ProductWeight
ProductPicture
ProductDescipction
Belongs to
Has
CATEGORY
PK CategoryID
CategoryName
CategoryDescription
CategoryPicture
SHIPMENT
PK ShipmentID
FK CustomerAddressID
Shipping Date
ShippingMethod
SHIPPER
PK ShipperID
ShipperName
ShipperPhoneNo
ORDERDETAILS
PK OrderDetailsID
FK OrderID
FK ProductID
OrderQuantity
FulfilmentStatus
ORDER
PK OrderID
FK CustomerID
OrderDate
OrderTotal
FK ShipmentID
OrderTime
Is ordered by
Orders
Is paid by
Belongs to
PAYMENT
PK PaymentID
FK OrderID
PaymentMethod
PaymentStatus
PaymentDate
PaymentTime
PaymentError
Is paid by
CUSTOMER
PK CustomerID
CustomerFirstName
CustomerLastName
CustomerPhoneNo
CustomerEmail
Gives
Is given by
Uses
CUSTOMERFEEDBACK
PK FeedbackID
FK OrderID
FK CustomerID
ProductRating
ShippingRating
ExperienceRating
CommentsAndFeedback
CREDITCARD
PK CreditCardNo
FK CustomerID
SetAsPrimary
CreditCardType
CardExpiryMonYear
Is used by
Is used for
Belongs To
CUSTOMER ADDRESS
PK CustomerAddressID
FK CustomerID
Street
City
PostalCode
UseAsBillingAddress

Database Objects

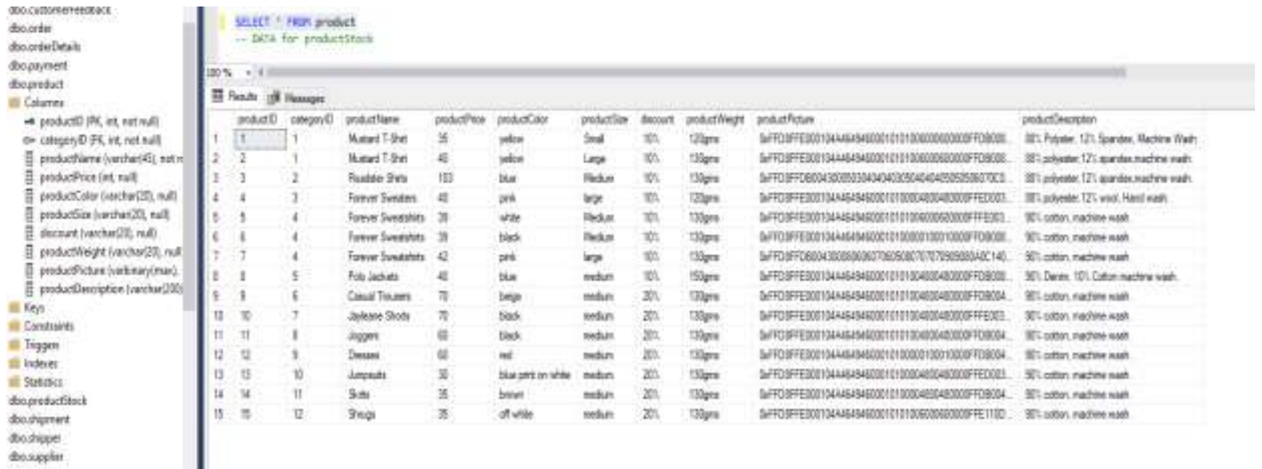
- DDL for Product Table

Table: product

```sql
/* CREATE table product */
CREATE TABLE [dbo].[product] (
  [productID] int NOT NULL,
  [categoryID] int NOT NULL,
  [productName] varchar(45) NOT NULL,
  [productPrice] int,
  [productColor] varchar(20),
  [productSize] varchar(20),
  [discount] varchar(20),
  [productWeight] varchar(20),
  [productPicture] varbinary(max),
  [productDescription] varchar(200)

  CONSTRAINT prim_Key_product PRIMARY KEY CLUSTERED ([productID] ASC),
  )
ON [PRIMARY]

-- Add CHECK CONSTRAINT FOREIGN KEY for product Table--
ALTER TABLE [dbo].[product] WITH CHECK ADD  CONSTRAINT foreign_key_categoryID
FOREIGN KEY ([categoryID]) REFERENCES [dbo].[category] ([categoryID])

--Add a CHECK for productID in product Table --
ALTER TABLE [dbo].[product] WITH CHECK ADD CONSTRAINT CHK_productID CHECK (productID > 0
);
```

- Inserting data into Product Table

```sql
INSERT INTO product (productID, productName, categoryID, productPrice, productColor,
productSize, discount, productWeight, productPicture, productDescription)
VALUES (1,'Mustard T-Shirt', 1, 35,'yellow','Small','10%','120gms', (SELECT * FROM
OPENROWSET (BULK N'D:Saved Pictures\MustardT.jpg', SINGLE_BLOB)image),'88% Polyster, 12%
Spandex, Machine Wash');
INSERT INTO product (productID, productName, categoryID, productPrice, productColor,
productSize, discount, productWeight, productPicture, productDescription)
VALUES (2,'Mustard T-Shirt', 1, 40,'yellow','Large','10%','130gms', (SELECT * FROM
OPENROWSET (BULK N'D:Saved Pictures\MustardT.jpg', SINGLE_BLOB)image),'88% polyester,12%
spandex,machine wash.');
INSERT INTO product (productID, productName, categoryID, productPrice, productColor,
productSize, discount, productWeight, productPicture, productDescription)
VALUES (3,'Roadster Shirts', 2, 103,'blue','Medium','10%','130gms', (SELECT * FROM
OPENROWSET (BULK N'D:Saved Pictures\RoadsterShirt.jpg', SINGLE_BLOB)image),'88%
polyester,12% spandex,machine wash.');
INSERT INTO product (productID, productName, categoryID, productPrice, productColor,
productSize, discount, productWeight, productPicture, productDescription)
VALUES (4,'Forever Sweaters', 3, 40,'pink','large','10%','120gms', (SELECT * FROM
OPENROWSET (BULK N'D:Saved Pictures\ForeverSweater.jpg', SINGLE_BLOB)image),'88%
polyester,12% wool, Hand wash.');
INSERT INTO product (productID, productName, categoryID, productPrice, productColor,
productSize, discount, productWeight, productPicture, productDescription)
VALUES (5,'Forever Sweatshirts', 4, 39,'white','Medium','10%','130gms', (SELECT * FROM
OPENROWSET (BULK N'D:Saved Pictures\ForeverSweatshirtWhite.jpg', SINGLE_BLOB)image),'90%
cotton, machine wash.');
```



```
dbo.customerFeedback
dbo.order
dbo.orderDetails
dbo.payment
dbo.product
  Columns
    productID (PK, int, not null)
    categoryID (FK, int, not null)
    productName (varchar(45), not n
    productPrice (int, null)
    productColor (varchar(20), null)
    productSize (varchar(20), null)
    discount (varchar(20), null)
    productWeight (varchar(20), null
    productPicture (varbinary(max),
    productDescription (varchar(200)
  Keys
  Constraints
  Triggers
  Indexes
  Statistics
dbo.productStock
dbo.shipment
dbo.shipper
dbo.supplier
```

```sql
SELECT * FROM product
-- DATA for productStock
```

100 %

Results | Messages

	productID	categoryID	productName	productPrice	productColor	productSize	discount	productWeight	productPicture	productDescription
1	1	1	Mustard T-Shirt	35	yellow	Small	10%	120gms	0xFFD8FFE000104A46494600010101006000600000FFDB00...	88% Polyster, 12% Spandex, Machine Wash
2	2	1	Mustard T-Shirt	40	yellow	Large	10%	130gms	0xFFD8FFE000104A46494600010101006000600000FFDB00...	88% polyester,12% spandex,machine wash.
3	3	2	Roadster Shirts	103	blue	Medium	10%	130gms	0xFFD8FFDB004300050304040403050404040505050607 0C...	88% polyester,12% spandex,machine wash.
4	4	3	Forever Sweaters	40	pink	large	10%	120gms	0xFFD8FFE000104A46494600010100004800480000FFED003...	88% polyester,12% wool, Hand wash.
5	5	4	Forever Sweatshirts	39	white	Medium	10%	130gms	0xFFD8FFE000104A46494600010101006000600000FFFE003...	90% cotton, machine wash
6	6	4	Forever Sweatshirts	39	black	Medium	10%	130gms	0xFFD8FFE000104A46494600010100000100010000FFDB00...	90% cotton, machine wash
7	7	4	Forever Sweatshirts	42	pink	large	10%	130gms	0xFFD8FFDB00430006060607060508070707090908 0A0C140...	90% cotton, machine wash
8	8	5	Polo Jackets	40	blue	medium	10%	150gms	0xFFD8FFE000104A46494600010101004800480000FFDB00...	90% Denim, 10% Cotton machine wash.
9	9	6	Casual Trousers	70	beige	medium	20%	130gms	0xFFD8FFE000104A46494600010101004800480000FFDB004...	90% cotton, machine wash
10	10	7	Jayleane Shorts	70	black	medium	20%	130gms	0xFFD8FFE000104A46494600010101004800480000FFFE003...	90% cotton, machine wash
11	11	8	Joggers	60	black	medium	20%	130gms	0xFFD8FFE000104A46494600010101004800480000FFDB004...	90% cotton, machine wash
12	12	9	Dresses	60	red	medium	20%	130gms	0xFFD8FFE000104A46494600010100000100010000FFDB004...	90% cotton, machine wash
13	13	10	Jumpsuits	30	blue print on white	medium	20%	130gms	0xFFD8FFE000104A46494600010100004800480000FFED003...	90% cotton, machine wash
14	14	11	Skirts	35	brown	medium	20%	130gms	0xFFD8FFE000104A46494600010100004800480000FFDB004...	90% cotton, machine wash
15	15	12	Shrugs	35	off white	medium	20%	130gms	0xFFD8FFE000104A46494600010101006000600000FFE110D...	90% cotton, machine wash

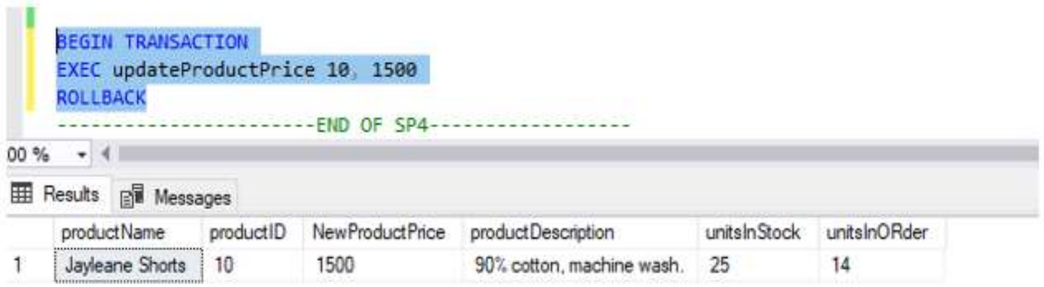
Database Objects

• Stored Procedures

Explanation: Gets productID and new productPrice as parameters and UPDATES PRODUCT PRICE.

```
CREATE PROCEDURE
updateProductPrice @product_ID INT, @new_product_Price VARCHAR(10)
AS
BEGIN
DECLARE @currProductPrice VARCHAR(10);
SET @currProductPrice = (SELECT productPrice from product where productID = @product_ID);
Update product SET productPrice = @new_product_Price where productID = @product_ID;
SELECT productName, p.productID, productPrice AS NewProductPrice, productDescription,
unitsInStock, unitsInORder FROM product p JOIN
productStock ON [productStock].[productID] = @product_ID and p.[productID] = @product_ID;
END
```

RESULT:



```
BEGIN TRANSACTION
EXEC updateProductPrice 10, 1500
ROLLBACK
-----------------------END OF SP4------------------
```

	productName	productID	NewProductPrice	productDescription	unitsInStock	unitsInORder
1	Jayleane Shorts	10	1500	90% cotton, machine wash.	25	14

• Triggers

Explanation:
This trigger is called on update of the product price. Check if the product price is not less than 0 and not greater that specified limit.

```
CREATE TRIGGER
CheckProductPriceChanges
ON product
AFTER UPDATE
AS
    DECLARE @productPrice INT
    SET @productPrice=(select productPrice from inserted)
    IF( @productPrice < 0)
       BEGIN
              UPDATE product SET productPrice = 0
       END
    IF(@productPrice > 10000)
       BEGIN
              UPDATE product SET productPrice=10000
       END
```

a. When the product price is given below 0, for example, say -5, the price gets updated as 0. This is because of the trigger "CheckProductPriceChanges" which checks the update on the price change of the product.

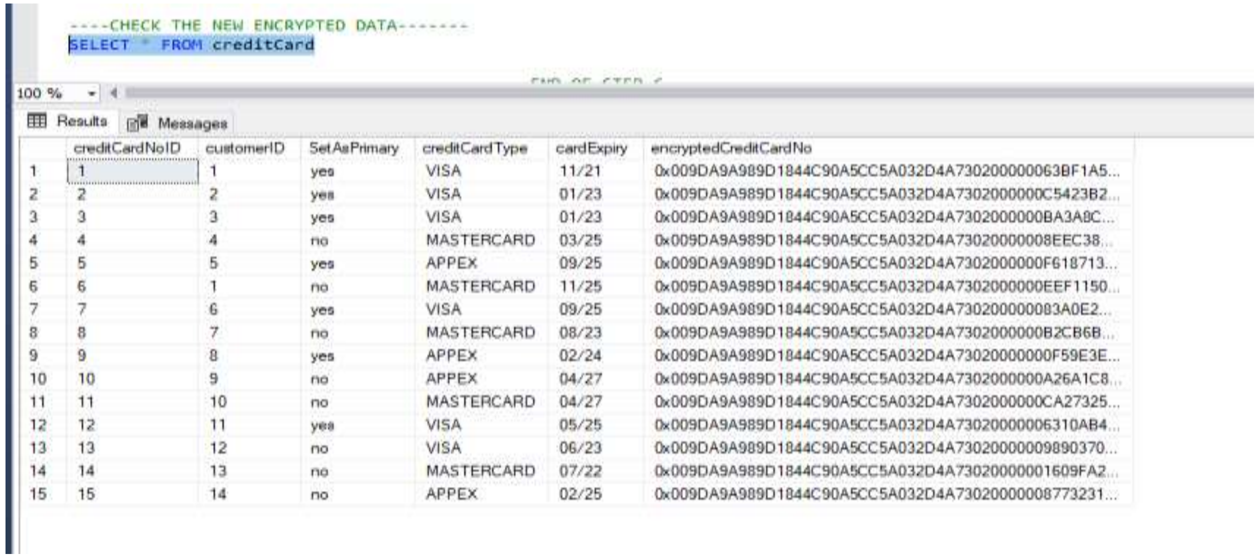
```
EXEC updateProductPrice 3,-5
```

	productName	productPrice	productDescription	unitsInStock	unitsInORder
1	[illegible]	0	[illegible]	[illegible]	[illegible]

Database Objects

- Encryption

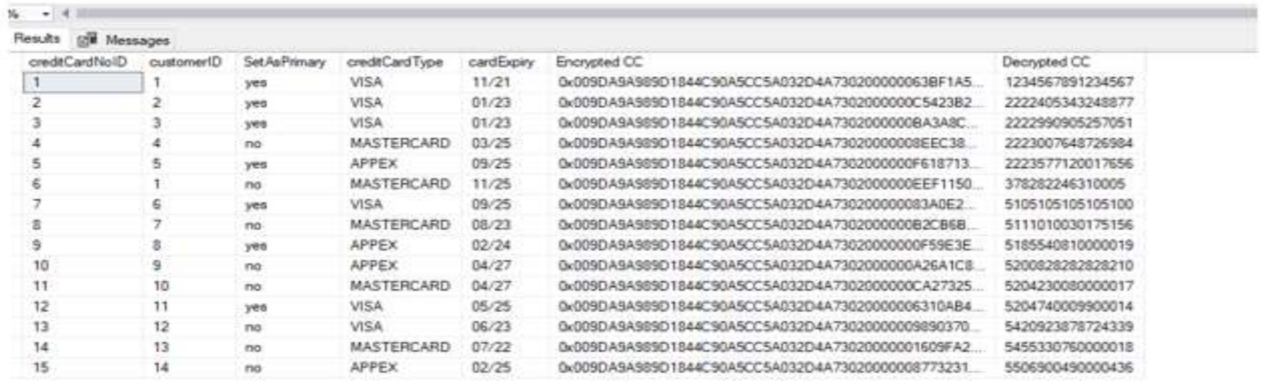
```
USE DMDDP4

GO

-- Create database Key
CREATE MASTER KEY
ENCRYPTION BY PASSWORD = 'DMDDP4Encrypt';

--verify that master key has been created
SELECT name KeyName,
symmetric_key_id KeyID,
key_length KeyLength,
algorithm_desc KeyAlgorithm
FROM sys.symmetric_keys;

-- Create self-signed certificate
USE DMDDP4;
GO
CREATE CERTIFICATE CreditCardNumber
WITH SUBJECT = 'EncryptCreditCardData';
GO
-- Create symmetric Key
CREATE SYMMETRIC KEY CustCC_SM
 WITH ALGORITHM = AES_256
 ENCRYPTION BY CERTIFICATE CreditCardNumber;

--ADD new column for encrypted data
 ALTER TABLE creditcard
 ADD encryptedCreditCardNo varbinary(MAX)

-- Opens the symmetric key for use
OPEN SYMMETRIC KEY CustCC_SM
DECRYPTION BY CERTIFICATE CreditCardNumber;


-- Populating encrypted credit card no into new column
UPDATE dbo.creditcard
SET encryptedCreditCardNo = EncryptByKey (Key_GUID('CustCC_SM'), creditCardNo)
FROM dbo.creditcard;
GO

-- Closing the symmetric key
CLOSE SYMMETRIC KEY CustCC_SM;
GO

---DROPPING CreditCardNo----
ALTER TABLE creditCard
DROP COLUMN creditCardNo;
GO
```



```
----CHECK THE NEW ENCRYPTED DATA-------
SELECT * FROM creditCard
```

	creditCardNoID	customerID	SetAsPrimary	creditCardType	cardExpiry	encryptedCreditCardNo
1	1	1	yes	VISA	11/21	0x009DA9A989D1844C90A5CC5A032D4A730200000063BF1A5...
2	2	2	yes	VISA	01/23	0x009DA9A989D1844C90A5CC5A032D4A7302000000C5423B2...
3	3	3	yes	VISA	01/23	0x009DA9A989D1844C90A5CC5A032D4A7302000000BA3A8C...
4	4	4	no	MASTERCARD	03/25	0x009DA9A989D1844C90A5CC5A032D4A73020000008EEC38...
5	5	5	yes	APPEX	09/25	0x009DA9A989D1844C90A5CC5A032D4A7302000000F618713...
6	6	1	no	MASTERCARD	11/25	0x009DA9A989D1844C90A5CC5A032D4A7302000000EEF1150...
7	7	6	yes	VISA	09/25	0x009DA9A989D1844C90A5CC5A032D4A730200000083A0E2...
8	8	7	no	MASTERCARD	08/23	0x009DA9A989D1844C90A5CC5A032D4A7302000000B2CB6B...
9	9	8	yes	APPEX	02/24	0x009DA9A989D1844C90A5CC5A032D4A73020000000F59E3E...
10	10	9	no	APPEX	04/27	0x009DA9A989D1844C90A5CC5A032D4A7302000000A26A1C8...
11	11	10	no	MASTERCARD	04/27	0x009DA9A989D1844C90A5CC5A032D4A7302000000CA27325...
12	12	11	yes	VISA	05/25	0x009DA9A989D1844C90A5CC5A032D4A73020000006310AB4...
13	13	12	no	VISA	06/23	0x009DA9A989D1844C90A5CC5A032D4A73020000009890370...
14	14	13	no	MASTERCARD	07/22	0x009DA9A989D1844C90A5CC5A032D4A73020000001609FA2...
15	15	14	no	APPEX	02/25	0x009DA9A989D1844C90A5CC5A032D4A73020000008773231...

```
---Decrypting Credit Card No-
OPEN SYMMETRIC KEY CustCC_SM
DECRYPTION BY CERTIFICATE CreditCardNumber;
SELECT creditCardNoID, customerID, SetAsPrimary, creditCardType, cardExpiry, encryptedCreditCardNo AS
'Encrypted CC', CONVERT(varchar(50), DecryptByKey(encryptedCreditCardNo)) AS 'Decrypted CC'
FROM creditcard
```



creditCardNoID	customerID	SetAsPrimary	creditCardType	cardExpiry	Encrypted CC	Decrypted CC
1	1	yes	VISA	11/21	0x009DA9A989D1844C90A5CC5A032D4A730200000063BF1A5...	1234567891234567
2	2	yes	VISA	01/23	0x009DA9A989D1844C90A5CC5A032D4A7302000000C5423B2...	2222405343248877
3	3	yes	VISA	01/23	0x009DA9A989D1844C90A5CC5A032D4A7302000000BA3A8C...	2222990905257051
4	4	no	MASTERCARD	03/25	0x009DA9A989D1844C90A5CC5A032D4A73020000008EEC38...	2223007648726984
5	5	yes	APPEX	09/25	0x009DA9A989D1844C90A5CC5A032D4A7302000000F618713...	2223577120017656
6	1	no	MASTERCARD	11/25	0x009DA9A989D1844C90A5CC5A032D4A7302000000EEF1150...	378282246310005
7	6	yes	VISA	09/25	0x009DA9A989D1844C90A5CC5A032D4A730200000083A0E2...	5105105105105100
8	7	no	MASTERCARD	08/23	0x009DA9A989D1844C90A5CC5A032D4A7302000000B2CB6B...	5111010030175156
9	8	yes	APPEX	02/24	0x009DA9A989D1844C90A5CC5A032D4A73020000000F59E3E...	5185540810000019
10	9	no	APPEX	04/27	0x009DA9A989D1844C90A5CC5A032D4A7302000000A26A1C8...	5200828282828210
11	10	no	MASTERCARD	04/27	0x009DA9A989D1844C90A5CC5A032D4A7302000000CA27325...	5204230080000017
12	11	yes	VISA	05/25	0x009DA9A989D1844C90A5CC5A032D4A73020000006310AB4...	5204740009900014
13	12	no	VISA	06/23	0x009DA9A989D1844C90A5CC5A032D4A73020000009890370...	5420923878724339
14	13	no	MASTERCARD	07/22	0x009DA9A989D1844C90A5CC5A032D4A73020000001609FA2...	5455330760000018
15	14	no	APPEX	02/25	0x009DA9A989D1844C90A5CC5A032D4A73020000008773231...	5506900490000436

Database Objects

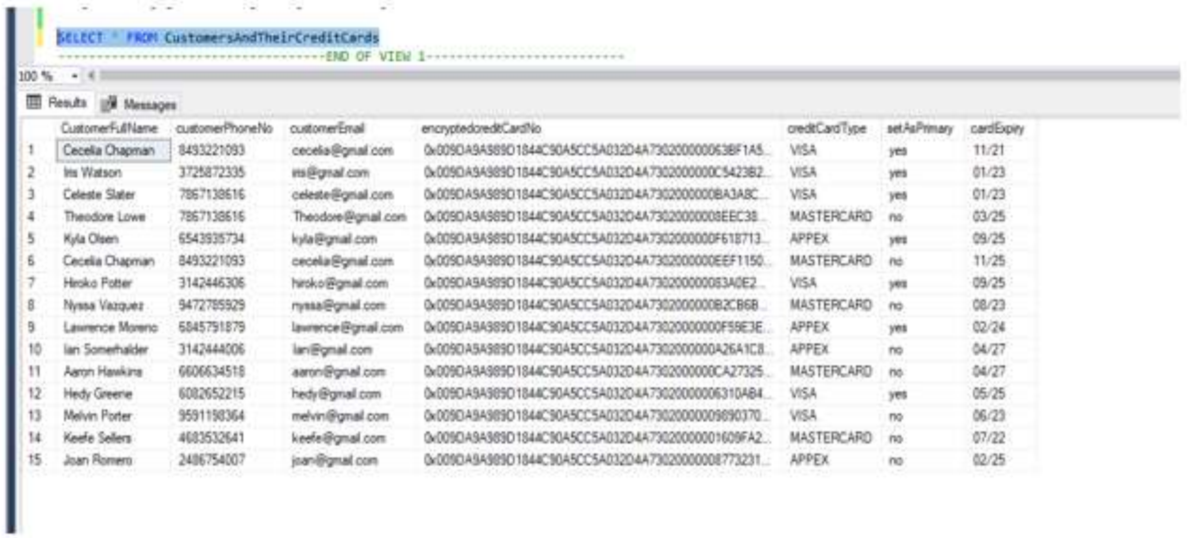
- Views

VIEW 1:

Explanation: This view displays all customer information and their credit card information.

```sql
CREATE VIEW
CustomersAndTheirCreditCards
AS
Select dbo.CustomerFullName (customer.customerID) AS CustomerFullName, customerPhoneNo,
customerEmail, C.encryptedcreditCardNo, creditCardType, setAsPrimary, cardExpiry
FROM Customer JOIN creditCard C
ON [Customer].[customerID] = C.[customerID];
```

Result:



```sql
SELECT * FROM CustomersAndTheirCreditCards
-----------------------------------END OF VIEW 1-------------------------
```

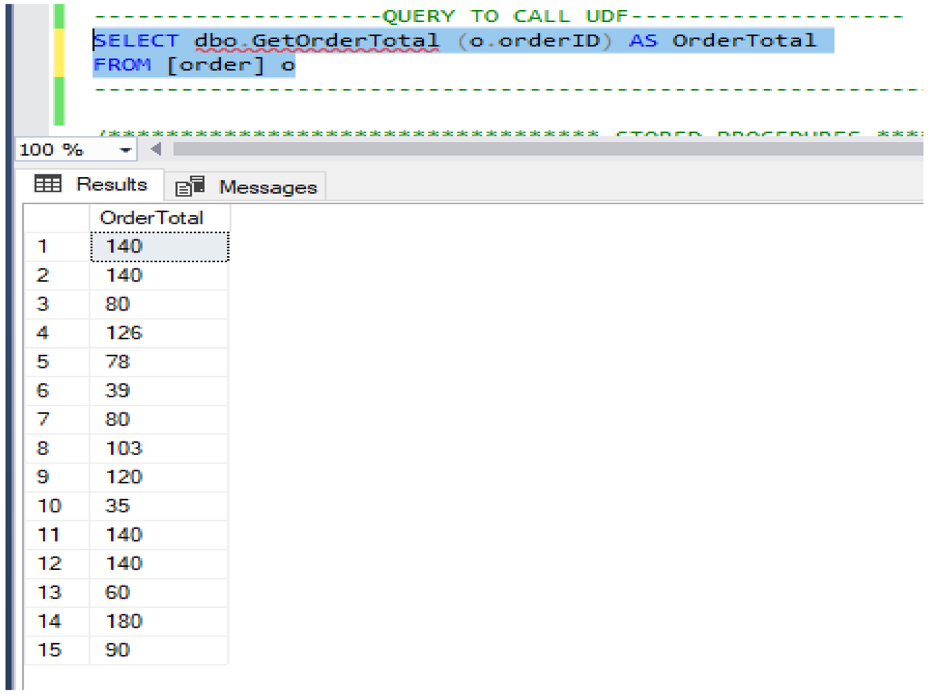
100 % | Results | Messages

	CustomerFullName	customerPhoneNo	customerEmail	encryptedcreditCardNo	creditCardType	setAsPrimary	cardExpiry
1	Cecelia Chapman	8493221093	cecelia@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A7302000000638F1A5...	VISA	yes	11/21
2	Iris Watson	3725872335	iris@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A730200000C5423B2...	VISA	yes	01/23
3	Celeste Slater	7867138616	celeste@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A730200000BA3A8C...	VISA	yes	01/23
4	Theodore Lowe	7867138616	Theodore@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A7302000008EEC38...	MASTERCARD	no	03/25
5	Kyla Olsen	6543935734	kyla@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A730200000F618713...	APPEX	yes	09/25
6	Cecelia Chapman	8493221093	cecelia@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A730200000EEF1150...	MASTERCARD	no	11/25
7	Hiroko Potter	3142446306	hiroko@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A73020000083A0E2...	VISA	yes	09/25
8	Nyssa Vazquez	9472785929	nyssa@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A730200000B2CB6B...	MASTERCARD	no	08/23
9	Lawrence Moreno	6845791879	lawrence@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A7302000000F59E3E...	APPEX	yes	02/24
10	Ian Somerhalder	3142444006	ian@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A730200000A26A1C8...	APPEX	no	04/27
11	Aaron Hawkins	6606634518	aaron@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A730200000CA27325...	MASTERCARD	no	04/27
12	Hedy Greene	6082652215	hedy@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A73020000006310AB4...	VISA	yes	05/25
13	Melvin Porter	9591198364	melvin@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A7302000009890370...	VISA	no	06/23
14	Keefe Sellers	4683532641	keefe@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A73020000001609FA2...	MASTERCARD	no	07/22
15	Joan Romero	2486754007	joan@gmail.com	0x009DA9A989D1844C90A5CC5A032D4A7302000008773231...	APPEX	no	02/25

- User Defined Function

Explanation: Take OrderID as input and returns Order Total

```sql
CREATE FUNCTION
GetOrderTotal (@orderID int)
RETURNS Float
AS
BEGIN
DECLARE @OrderTotal float
SELECT @OrderTotal = SUM ((o.orderQuantity) * (p.productPrice))
FROM orderDetails o JOIN product p
ON orderID = @orderID AND o.productID = p.productID
RETURN @OrderTotal
END
```



```sql
--------------------QUERY TO CALL UDF--------------------
SELECT dbo.GetOrderTotal (o.orderID) AS OrderTotal
FROM [order] o
------------------------------------------------------------

/********************************** STORED PROCEDURES ***
```

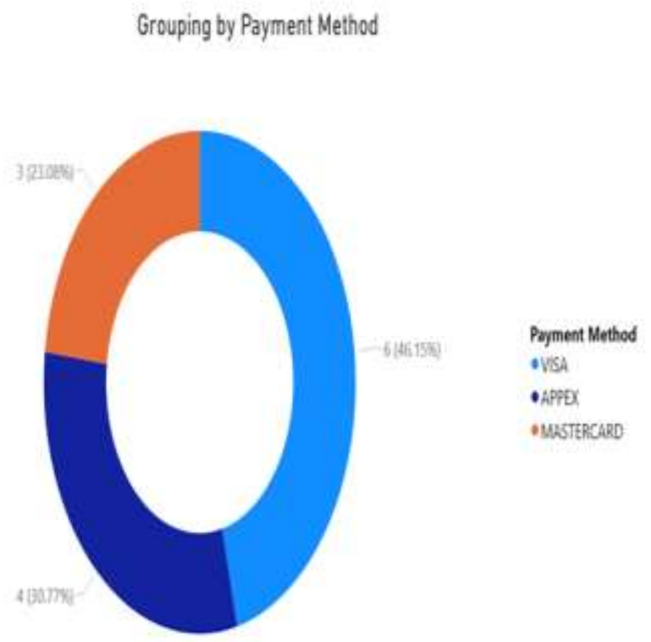
100 % | Results | Messages

	OrderTotal
1	140
2	140
3	80
4	126
5	78
6	39
7	80
8	103
9	120
10	35
11	140
12	140
13	60
14	180
15	90

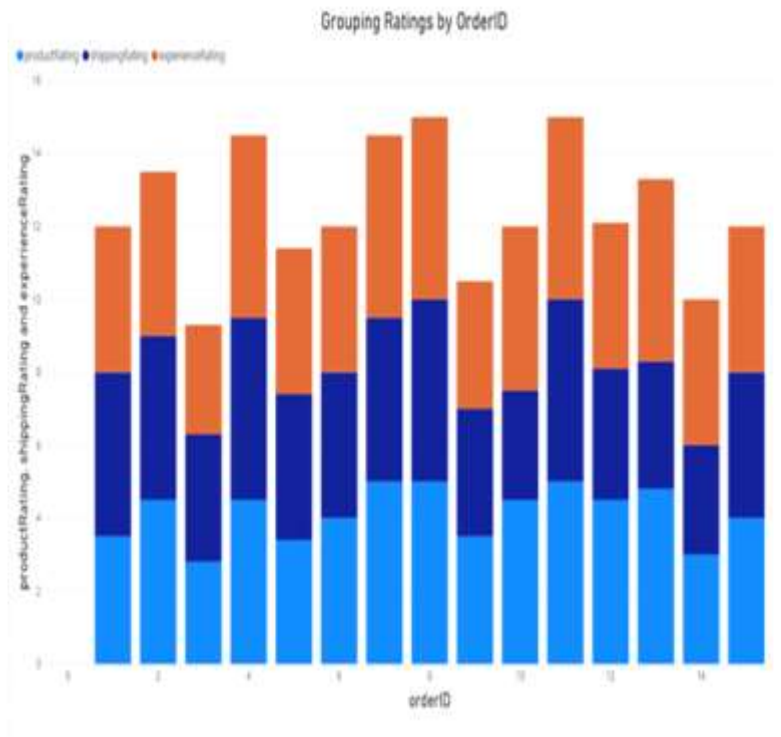
Project Analysis

- Power BI analysis

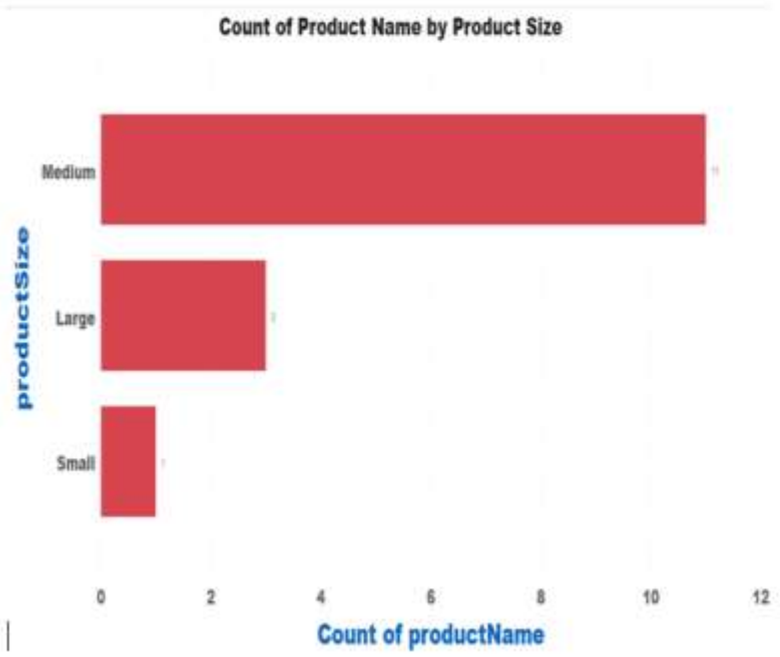
Visualization 1: Displaying the percentage of the payment methods (VISA, APPEX and MASTERCARD. This shows that most of the payments are made by VISA.



Visualization 2: This bar chart helps us to know the different ratings (according to product, shipping and experience) given by the customer , which is grouped by the orderID



Visualization 3: This bar chart helps us to know the count of the products grouped by the product size. We see that the medium size has the highest number (11) of products as per the given data.



Thank You