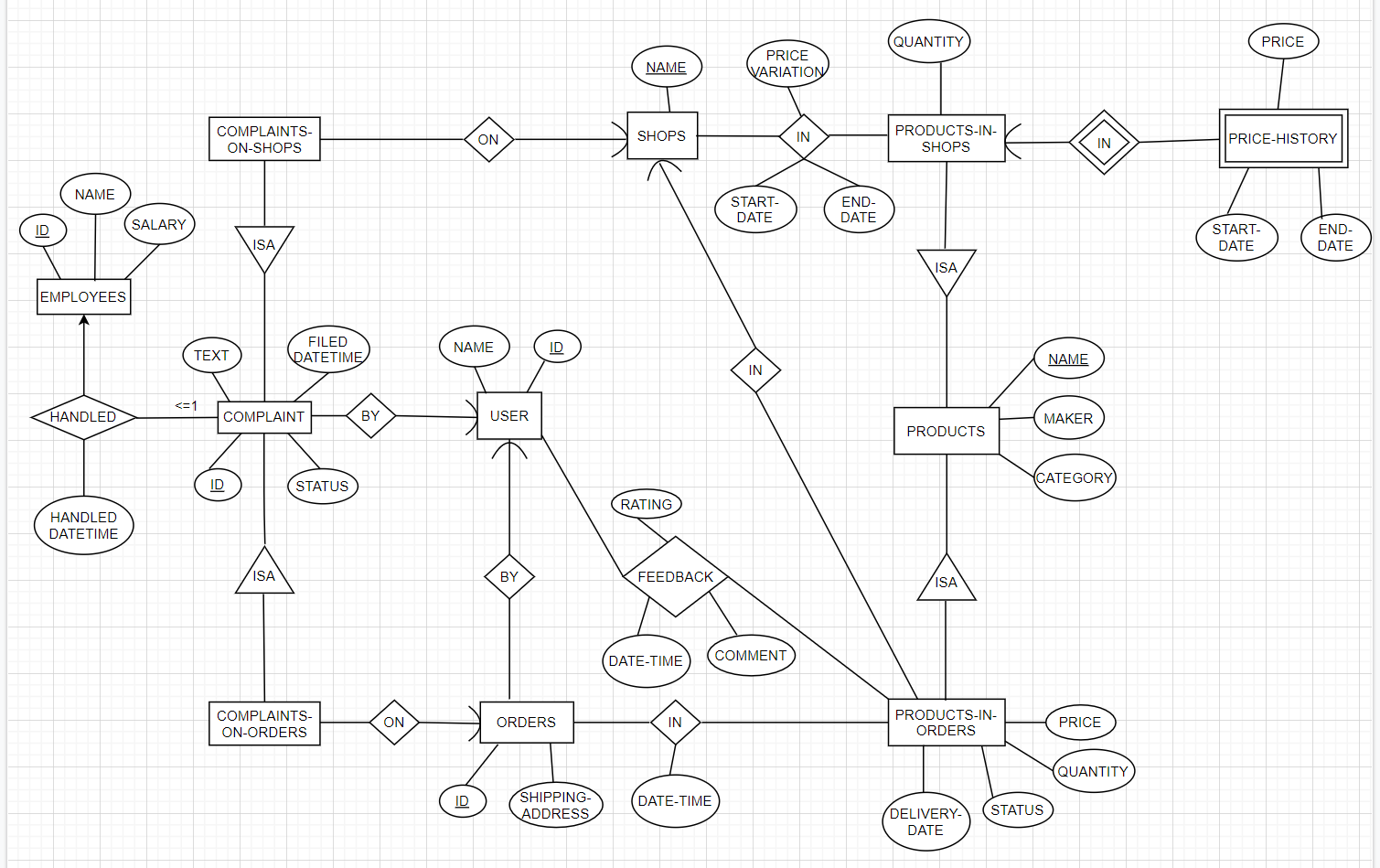
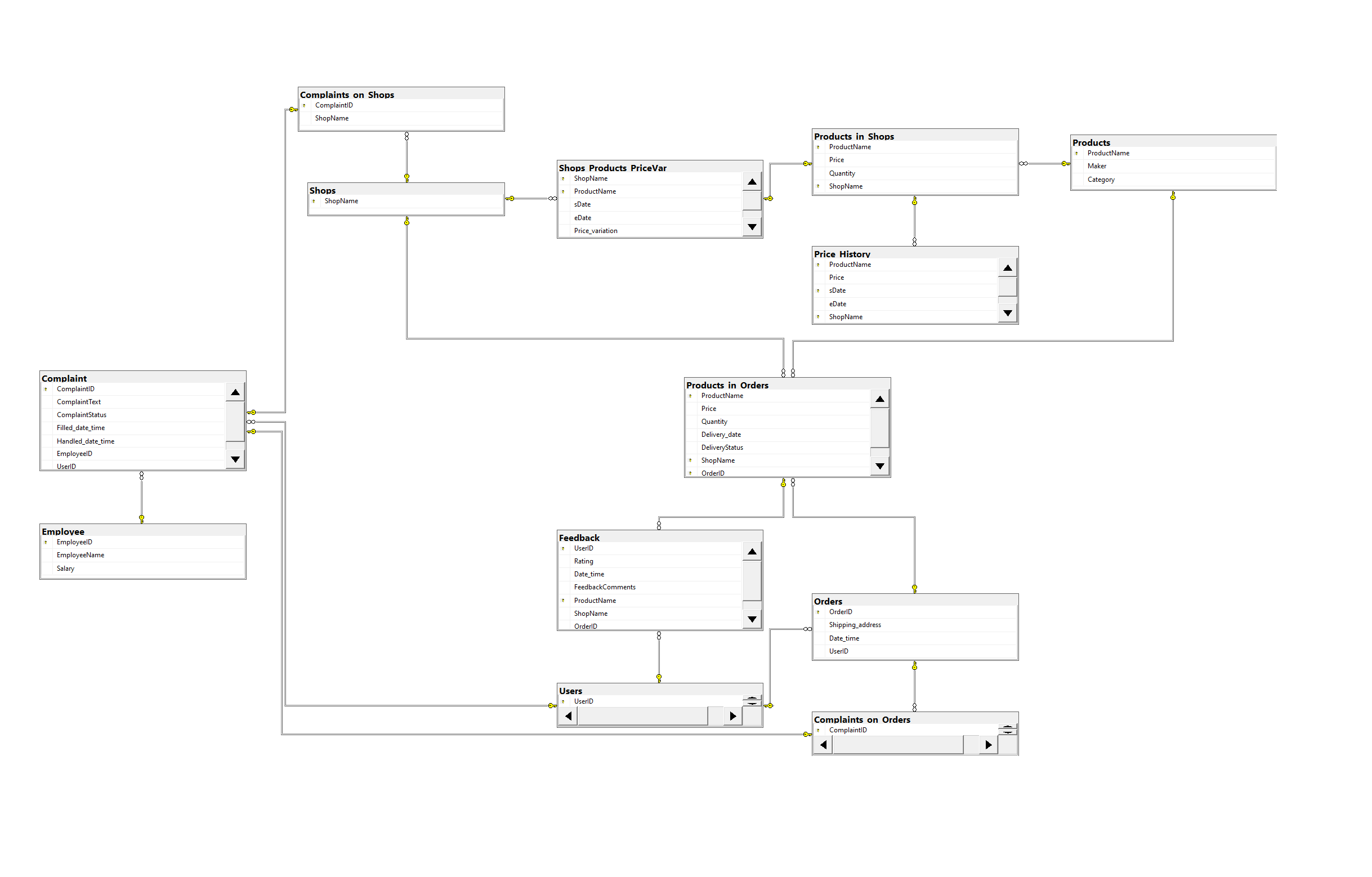
# **Database Diagram**

****

MS SQL Generated ERD

****

# **Relation Schemas**

**1** Employees (EmployeeID, EmployeeName, Salary)

**2** Complaint (ComplaintID, ComplaintText, ComplaintStatus, Filled\_date\_time,

Handled\_date\_time, EmployeeID, UserID)

**3** Complaints\_on\_Shops (ComplaintID, ShopName)

**4** Complaints\_on\_Orders (ComplaintID, OrderID)

**5** Orders (OrderID, Shipping\_address, Date\_time, UserID)

**6** User (UserID, UserName)

**7** Feedback (UserID, Rating, Date\_time, FeedbackComments, ProductName,

ShopName, OrderID)

**8** Product (ProductName, Maker, Category)

**9** Product\_in\_Orders (ProductName, Price, Quantity, Delivery\_date,

DeliveryStatus, ShopName, OrderID)

**10** Product\_in\_Shops (ProductName, Price, Quantity, ShopName)

**11** Shops (ShopName)

**12** Price\_History (ProductName, Price, sDate, eDate, ShopName)

**13** Shops\_Products\_PriceVar (ShopName, ProductName, sDate, eDate,

Price\_variation)

# **1. Employee**

**Employee** (EmployeeID, EmployeeName, Salary)

**Primary Key:** EmployeeID

**Foreign Key:** -

CREATE TABLE **Employee**(  
 **EmployeeID** int NOT NULL IDENTITY (1,1),

**EmployeeName** varchar(255) NOT NULL,

**Salary** float NOT NULL,

PRIMARY KEY (**EmployeeID**),

);

# **2. Complaint**

**Complaint** (ComplaintID, ComplaintText, ComplaintStatus, Filled\_date\_time, Handled\_date\_time, EmployeeID, UserID)

**Primary Key:** ComplaintID

**Foreign Key:** EmployeeID -> Employee(EmployeeID), UserID -> Users(UserID)

CREATE TABLE **Complaint**(  
 **ComplaintID** int NOT NULL IDENTITY(1,1),

**ComplaintText** varchar(255) NOT NULL,

**ComplaintStatus** varchar(255) NOT NULL DEFAULT 'pending',

**Filled\_date\_time** datetime NOT NULL,

**Handled\_date\_time** datetime,

**EmployeeID** int,

**UserID** int NOT NULL,

PRIMARY KEY (**ComplaintID**),

FOREIGN KEY (**EmployeeID**) REFERENCES **Employee**(**EmployeeID**)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY (**UserID**) REFERENCES **Users**(**UserID**)

ON UPDATE CASCADE

ON DELETE CASCADE

);

# **3. Complaints\_on\_Shops**

**Complaints\_on\_Shops** (ComplaintID, ShopName)

**Primary Key:** ComplaintID

**Foreign Key:** ComplaintID -> Complaint(ComplaintID), ShopName -> Shops(ShopName)

CREATE TABLE **Complaints\_on\_Shops**(  
 **ComplaintID** int NOT NULL,

**ShopName** varchar(255) NOT NULL,

PRIMARY KEY (**ComplaintID**),

FOREIGN KEY (**ComplaintID**) REFERENCES **Complaint**(**ComplaintID**)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY (**ShopName**) REFERENCES **Shops**(**ShopName**)

ON UPDATE CASCADE

ON DELETE CASCADE

);

# **4. Complaints\_on\_Orders**

**Complaints\_on\_Orders** (ComplaintID, OrderID)

**Primary Key:** ComplaintID

**Foreign Keys:** ComplaintID -> Complaint(ComplaintID), OrderID -> Orders(OrderID)

CREATE TABLE **Complaints\_on\_Orders** (  
 **ComplaintID** int NOT NULL,

**OrderID** int NOT NULL,

PRIMARY KEY (**ComplaintID**),

FOREIGN KEY (**ComplaintID**) REFERENCES **Complaint**(**ComplaintID**)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

FOREIGN KEY (**OrderID**) REFERENCES **Orders**(**OrderID**)

ON UPDATE NO ACTION

ON DELETE NO ACTION

);

# **5. Orders**

**Orders** (OrderID, Shipping\_address, Date\_time, UserID)

**Primary Key:** OrderID

**Foreign Key:** UserID -> Users(UserID)

CREATE TABLE **Orders** (  
 **OrderID** int NOT NULL IDENTITY(1,1),

**Shipping\_address** varchar(255) NOT NULL,

**Date\_time** datetime NOT NULL,

**UserID** int NOT NULL,

PRIMARY KEY (**OrderID**),

FOREIGN KEY (**UserID**) REFERENCES **Users**(**UserID**)

ON UPDATE CASCADE

ON DELETE CASCADE

);

# **6. User**

**User** (UserID, UserName)

**Primary Key:** UserID

**Foreign Key:** -

CREATE TABLE **User** (  
 **UserID** int NOT NULL IDENTITY(1,1),

**UserName** varchar(255) NOT NULL,

PRIMARY KEY (**UserID**)  
);

# **7. Feedback**

**Feedback** (UserID, Rating, Date\_time, FeedbackComments, ProductName, ShopName, OrderID)

**Primary Keys:** UserID, ProductName

**Foreign Keys:** UserID -> Users(UserID), ProductName, OrderID, ShopName -> Products\_in\_Orders(ProductName, OrderID, ShopName)

CREATE TABLE **Feedback**(  
 **UserID** int NOT NULL,

**Rating** int NOT NULL,

**Date\_time** datetime NOT NULL,

**FeedbackComments** varchar(255),

**ProductName** varchar(255) NOT NULL,

**ShopName** varchar(255) NOT NULL,

**OrderID** int NOT NULL,

PRIMARY KEY (**UserID**, **ProductName**),

FOREIGN KEY (**UserID**) REFERENCES **Users**(**UserID**)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY (**ProductName**, **OrderID**, **ShopName**) REFERENCES **Products\_in\_Orders**(**ProductName**, **OrderID**, **ShopName**)

ON UPDATE NO ACTION

ON DELETE NO ACTION  
 );

# **8. Product**

**Products** (ProductName, Maker, Category)

**Primary Key:** ProductName

**Foreign Key: -**

CREATE TABLE **Products** (  
 **ProductName** varchar(255) NOT NULL,

**Maker** varchar(255) NOT NULL,

**Category** varchar(255) NOT NULL,

PRIMARY KEY (**ProductName**)

);

# **9. Product\_in\_Orders**

**Product\_in\_Orders** (ProductName, Price, Quantity, Delivery\_date, DelivertyStatus

ShopName, OrderID)

**Primary Key:** ProductName, OrderID, ShopName

**Foreign Key:** ProductName -> Products(ProductName), OrderID -> Orders(OrderID), ShopName -> Shops(ShopName)

CREATE TABLE **Product\_in\_Orders** (  
 **ProductName** varchar(255) NOT NULL,

**Price** float NOT NULL,

**Quantity** int NOT NULL,

**Delivery\_date** datetime,

**DeliveryStatus** varchar(255) NOT NULL DEFAULT 'being processed',

**ShopName** varchar(255) NOT NULL,

**OrderID** int NOT NULL,

PRIMARY KEY (**ProductName**, **OrderID**, **ShopName**),

FOREIGN KEY (**ProductName**) REFERENCES **Products**(**ProductName**)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY (**OrderID**) REFERENCES **Orders**(**OrderID**)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY (**ShopName**) REFERENCES **Shops**(**ShopName**)

ON UPDATE CASCADE

ON DELETE CASCADE  
);

# **10. Product\_in\_Shops**

**Product\_in\_Shops** (ProductName, Price, Quantity, ShopName)

**Primary Key:** ProductName, ShopName

**Foreign Key:** ProductName -> Products(ProductName)

CREATE TABLE **Product\_in\_Shops** (  
 **ProductName** varchar(255) NOT NULL,

**Price** float NOT NULL,

**Quantity** int NOT NULL,

**ShopName** varchar(255) NOT NULL,

PRIMARY KEY (**ProductName**, **ShopName**),

FOREIGN KEY (**ProductName**) REFERENCES **Products**(**ProductName**)

ON UPDATE CASCADE

ON DELETE CASCADE

);

# **11. Shops**

**Shops** (ShopName)

**Primary Key:** ShopName

**Foreign Key:** -

CREATE TABLE **Shops** (  
 **ShopName** varchar(255) NOT NULL,

PRIMARY KEY (**ShopName**)  
);

# **12. Price\_History**

**Price\_History** (ProductName, Price, sDate, eDate, ShopName)

**Primary Key:** ProductName, ShopName, sDate

**Foreign Key:** ProductName, ShopName->Products\_in\_Shops (ProductName, ShopName)

CREATE TABLE **Price\_History** (  
 **ProductName** varchar(255) NOT NULL,

**Price** float NOT NULL,

**sDate** date NOT NULL,

**eDate** date,

**ShopName** varchar(255) NOT NULL,

PRIMARY KEY (**ProductName**, **ShopName**, **sDate**),

FOREIGN KEY (**ProductName**, **ShopName**) REFERENCES **Products\_in\_Shops**(**ProductName**, **ShopName**)

ON UPDATE CASCADE

ON DELETE CASCADE

);

# **13. Shops\_Products\_PriceVar**

**Shops\_Products\_PriceVar** (ShopName, ProductName, sDate, eDate,

Price\_variation)

**Primary Key:** ProductName, ShopName

**Foreign Key:** ProductName, ShopName->Products\_in\_Shops (ProductName, ShopName), ShopName -> Shops(ShopName)

CREATE TABLE **Shops\_Products\_PriceVar** (  
 **ShopName** varchar(255) NOT NULL,

**ProductName** varchar(255) NOT NULL,

**sDate** date NOT NULL,

**eDate** date,

**Price\_variation** float NOT NULL,

PRIMARY KEY (**ProductName**, **ShopName**),

FOREIGN KEY (**ProductName**, **ShopName**) REFERENCES **Products\_in\_Shops**(**ProductName**, **ShopName**)

ON UPDATE CASCADE

ON DELETE CASCADE,

FOREIGN KEY (**ShopName**) REFERENCES **Shops**(**ShopName**)

ON UPDATE CASCADE

ON DELETE CASCADE  
);

# **Queries**

**1** Find the average price of “iPhone Xs” on Sharkee from 1 August 2020 to 31 August 2020.

**SELECT** AVG([Price]) **AS** Average

**FROM** [SS11\_Group1].[dbo].[Products\_in\_Orders]

**WHERE** [Productname] = 'iPhone Xs' AND [Delivery\_date] BETWEEN '2020-08-01' and '2020-08-31';

**OUTPUT:**



**2** Find products that received at least 100 ratings of “5” in August 2020, and order them by their average ratings.

**SELECT** [ProductName] **AS** Rating100of5

**INTO** Rating100

**FROM** [SS11\_Group1].[dbo].[Feedback]

**WHERE** [Date\_time] **BETWEEN** '2020-08-01' **AND** '2020-08-31' **AND** [Rating] = 5

**GROUP BY** [ProductName]

**HAVING** COUNT(\*) >= 100;

**SELECT** ProductName **AS** ProductName,Avg(Rating) **AS** Ratings\_Avg

**INTO** AvgRatings

**FROM** [SS11\_Group1].[dbo].[Feedback]

**GROUP BY** [ProductName];

**SELECT** Rating100.Rating100of5, Ratings\_Avg

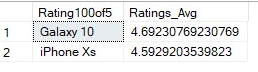
**FROM** Rating100

**INNER JOIN** AvgRatings **ON**  Rating100.Rating100of5 = AvgRatings.ProductName

**DROP TABLE** Rating100;

**DRop TABLE** AvgRatings;

**OUTPUT:**

****

**3** For all products purchased in June 2020 that have been delivered, find the average time from the ordering date to the delivery date.

**SELECT AVG(DATEDIFF** ( day , Orders.[Date\_time] , Products\_in\_Orders.[Delivery\_date] ) ) **AS** Avg\_Delivery\_Jun

**FROM** [SS11\_Group1].[dbo].[Products\_in\_Orders], [SS11\_Group1].[dbo].[Orders]

**WHERE** [SS11\_Group1].[dbo].[Orders].[Date\_time] **BETWEEN** '2020-06-01' **AND** '2020-06-30';

**OUTPUT:**

****

**4** Let us define the “latency” of an employee by the average that he/she takes to process a complaint. Find the employee with the smallest latency.

**SELECT AVG(DATEDIFF** ( day , [Filled\_date\_time] , [Handled\_date\_time] )) **AS** Lowest\_latency\_Emp\_ID

**FROM** [SS11\_Group1].[dbo].[Complaint]

**WHERE** [Handled\_date\_time] **IS NOT NULL**

**GROUP BY** EmployeeID

**ORDER BY** Lowest\_latency\_Emp\_ID **DESC**

OFFSET **0 ROWS**

**FETCH NEXT 1 ROWS ONLY;**

**OUTPUT:**

****

**5** Produce a list that contains (i) all products made by Samsung, and (ii) for each of them, the number of shops on Sharkee that sell the product.

**SELECT** [ProductName] **AS** SamsungProducts

**FROM** [SS11\_Group1].[dbo].[Products]

**WHERE** [Maker] = 'Samsung'

**SELECT** COUNT([ProductName]) **AS** SamsungProductsCount

**FROM** [SS11\_Group1].[dbo].[Products\_in\_Shops]

**WHERE** [ProductName] IN (

**SELECT** [ProductName] as SamsungProducts

**FROM**  [SS11\_Group1].[dbo].[Products]

**WHERE** [Maker] = 'Samsung')

**GROUP BY** [ProductName];

**OUTPUT:**

****

**6** Find shops that made the most revenue in August 2020.

**SELECT** PO.ShopName, SUM(PO.Price \* PO.Quantity) **as** Rev

**FROM** Products\_in\_Orders as PO, Orders as O

**WHERE** Date\_time BETWEEN '2020-08-01' AND '2020-08-31'

**GROUP BY** PO.ShopName

**ORDER BY** Rev **DESC**

**OFFSET** 0 **ROWS**

**FETCH NEXT** 1 ROWS ONLY;

**OUTPUT:**

****

**7** For users that made the most amount of complaints, find the most expensive products he/she has ever purchased.

**SELECT** U.UserID, COUNT(C.ComplaintID) **AS**'Num\_Of\_Complaints'

**INTO** Num\_Of\_User\_Complaints

**FROM** Users **AS** U, Complaint **AS** C

**WHERE** U.UserID = C.UserID

**GROUP BY** U.UserID

**SELECT \***

**INTO** Max\_Num\_Of\_Complaints

**FROM** Num\_Of\_User\_Complaints **AS** T1

**WHERE** T1.Num\_Of\_Complaints **IN (SELECT MAX**(Num\_Of\_Complaints)

**FROM** Num\_Of\_User\_Complaints**);**

**SELECT** O.OrderID, T2.UserID

**INTO** Most\_Complaints\_User\_Orders

**FROM** Orders **AS** O, Max\_Num\_Of\_Complaints **AS** T2

**WHERE** O.UserID = T2.UserID;

**SELECT** T2.UserID, P.ProductName, P.Price

**INTO** Most\_Complaints\_User\_Order\_Products

**FROM** Products\_in\_Orders **AS** P, Max\_Num\_Of\_Complaints **AS** T2, Most\_Complaints\_User\_Orders **AS** T3

**WHERE** P.OrderID = T3.OrderID;

**SELECT** UserID, ProductName, MAX(Price) **AS** 'Most\_Expensive'

**INTO** Most\_Expensive\_Bought

**FROM** Most\_Complaints\_User\_Order\_Products **AS** T4

**GROUP BY** UserID, ProductName;

**SELECT** U.UserName, T5.ProductName, T5.Most\_Expensive

**FROM** Users **AS** U, Most\_Expensive\_Bought **AS** T5

**WHERE** U.UserID = T5.UserID;

**DROP TABLE** Num\_Of\_User\_Complaints;

**DROP TABLE** Max\_Num\_Of\_Complaints;

**DROP TABLE** Most\_Complaints\_User\_Orders;

**DROP TABLE** Most\_Complaints\_User\_Order\_Products;

**DROP TABLE** Most\_Expensive\_Bought;

**OUTPUT:**

****

**8** Find products that have never been purchased by some users, but are the top 5 most purchased products by other users in August 2020.

**SELECT** O.UserID, PO.ProductName

**INTO** User\_Bought

**FROM** Orders **as** O, Products\_in\_Orders **as** PO

**WHERE** O.OrderID = PO.OrderID

**SELECT** User\_Bought.UserID, P.ProductName

**INTO** User\_Not\_Bought

**FROM** User\_Bought, Products **as** P

**WHERE** P.ProductName IN (

**SELECT** ProductName

**FROM** User\_Bought

**WHERE** User\_Bought.ProductName <> P.ProductName)

**EXCEPT**

**SELECT** \*

**FROM** User\_Bought

**SELECT** \*

**FROM** User\_Not\_Bought

**DROP TABLE** User\_Bought

**SELECT TOP** 5 PO.ProductName, COUNT(PO.ProductName) **as** 'Num\_Purchased'

**INTO** Aug\_Top

**FROM** Products\_in\_Orders **as** PO, Orders **as** O

**WHERE** PO.OrderID = O.OrderID

**AND** O.Date\_time BETWEEN '2020-08-01 00:00:00' AND '2020-08-31 23:59:59'

**AND** O.UserID = **SOME** (**SELECT** UserID **FROM** Users)

**GROUP BY** PO.ProductName

**ORDER BY** 'Num\_Purchased' **DESC**;

**SELECT**\*

**FROM** Aug\_Top;

**SELECT** UserID, User\_Not\_Bought.ProductName

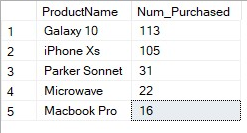
**FROM** Aug\_Top, User\_Not\_Bought

**WHERE** Aug\_Top.ProductName = User\_Not\_Bought.ProductName

**DROP TABLE** User\_Not\_Bought;

**DROP TABLE** Aug\_Top;

**OUTPUT:**

****

**9** Find products that are increasingly being purchased over at least 3 months.

**SELECT** PO.ProductName, DATEADD(MONTH, DATEDIFF(MONTH, 0, O.Date\_time), 0) **as** 'Year\_Month', **SUM**(Quantity) as 'Total\_Sold'

**INTO** Monthly\_Sold

**FROM** Orders **as** O

**JOIN** Products\_in\_Orders **as** PO ON O.OrderID = PO.OrderID

**GROUP BY** DATEADD(MONTH, DATEDIFF(MONTH, 0, O.Date\_time), 0), PO.ProductName;

**SELECT** Monthly\_Sold.ProductName, FORMAT(Year\_Month, 'yyyy-MM') **as** 'Increasing\_3\_Month'

**FROM** Monthly\_Sold

**WHERE** Monthly\_Sold.ProductName **IN** (**SELECT** MS2.ProductName

**FROM** Monthly\_Sold MS2

**WHERE** Monthly\_Sold.ProductName = MS2.ProductName

**AND** DATEDIFF(MONTH, Monthly\_Sold.Year\_Month, MS2.Year\_Month) = 1

**AND** Monthly\_Sold.Year\_Month < MS2.Year\_Month

**AND** Monthly\_Sold.Total\_Sold < MS2.Total\_Sold

**AND** MS2.ProductName **IN** (**SELECT** MS3.ProductName

**FROM** Monthly\_Sold as MS3

**WHERE** MS2.ProductName = MS3.ProductName

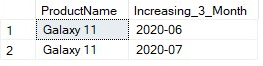
**AND** DATEDIFF(MONTH, MS2.Year\_Month, MS3.Year\_Month) = 1

**AND** MS2.Year\_Month < MS3.Year\_Month

**AND** MS2.Total\_Sold < MS3.Total\_Sold))

**DROP TABLE** Monthly\_Sold;

**OUTPUT:**

****

# **Triggers**

**CREATE TRIGGER** CheckRefundEligible **ON**  Products\_in\_Orders

**AFTER UPDATE**

**AS**

**BEGIN**

**IF** EXISTS(**SELECT** x.Delivery\_date **FROM** inserted x

**WHERE** x.DeliveryStatus = 'returned' **AND** DATEDIFF(DAY, x.Delivery\_date, GETDATE()) > 30

**GROUP BY** x.Delivery\_date

**HAVING** COUNT(\*) > 0)

**ROLLBACK TRANSACTION**

**END**

# **Constraints**

**ALTER TABLE** Employee

**ADD CHECK** (Salary>0);

**ALTER TABLE** Feedback

**ADD CHECK** (Rating BETWEEN 1 AND 5);

**ALTER TABLE** Products\_in\_Orders

**ADD CHECK** (Price >= 0 and Quantity >= 0);

**ALTER TABLE** Products\_in\_Shops

**ADD CHECK** (Price >= 0 and Quantity >= 0);