COURIER SERVICE MANAGEMENT (CMS)

GROUP 20

CHANDANASRI GANDHAM – 001069916

AJITH BABU RAMESH – 001061464

NETHRA BHUPATHI SUDARSHAN RAO -001029253

Objectives

The main goal is to deliver an efficient Export/Import Management System whose main functionality includes the booking a courier, maintain employee and customer details, process shipment and maintain the company details

The purpose is to minimize manual intervention for maintaining records and build computerized process by creating the database

Design a Database Architecture that

-> Allows future upgrades to be largely non-breaking

-> Easy to implement and maintain

PROJECT HIGHLIGHTS

- Increased Productivity: Automated Process for effective dispatch, order processing and delivery management
- Improved Customer Service: Real time access to shipment information, reducing call center volume.
- Invoice Faster: Digital workflow to initiate an order.
- Go Paperless: Digital workflow that eliminates cost of paper-based processes
- Could be integrated to any domain having features of package delivery

Business Requirements

CUSTOMERS

- Can register and login
- Can create an order to export product.
- Can view his orders and products couriered.
- Can enter their unique courier tracking number and get the recorded statuses of that package & its estimated delivery.

ADMIN

- Can register and login.
- Can enter their unique courier tracking number and get the last recorded status of that particular package & its estimated delivery.
- Update the shipments recent status.
- By default, admin of branch is given privilege to approve the shipment and assign it for delivery package to delivery employees.

PRODUCT

- Product type that can be delivered should be of type- food, metals, furniture or other

DELIVERY EMPLOYEES

- Can register and login.
- Will be able to see the packages assigned to him for delivery.
- Can update the status of his package.

SHIPMENT

- Customer, Admin or Delivery Employee should be able to track status of the shipment with reference to the tracking number

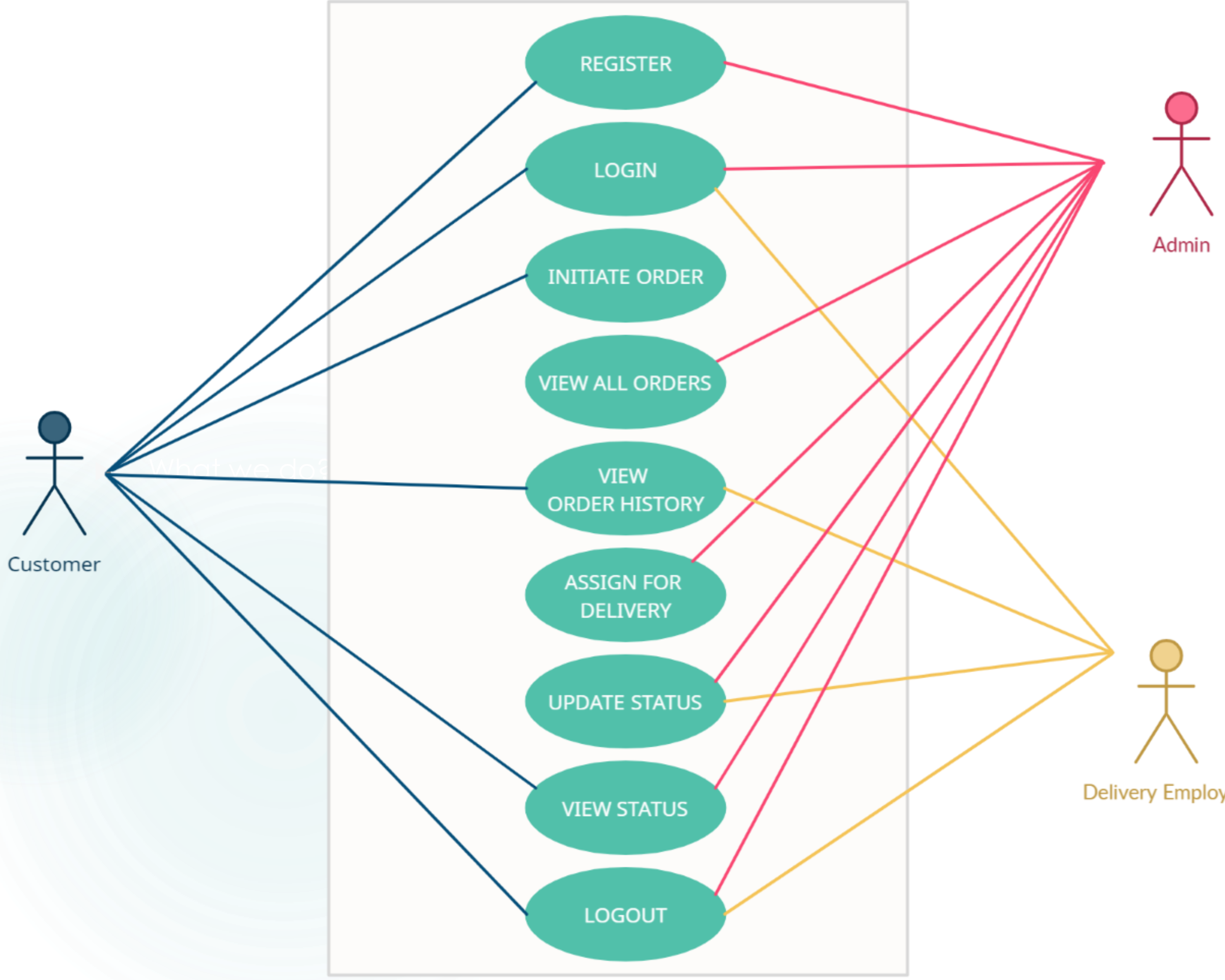
INVOICE

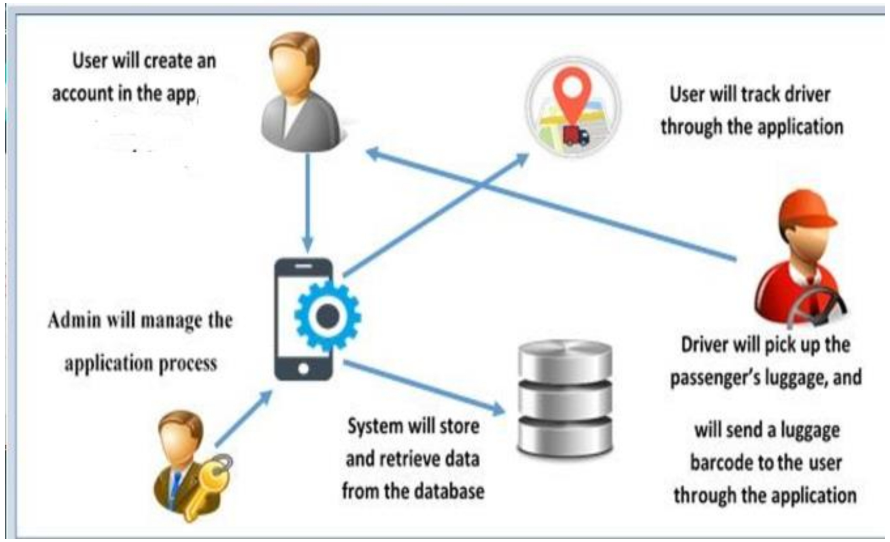
- Should contain the information about the Order and its Payment Type - Card or Coupon
- Total amount for an order should be calculated based on the below functionality
- If the total weight of the products contained in the order is less than 15 then it costs 20 per unit.
- If the total weight of the products contained in the order is between-15 to 50, then it costs 40 per unit.
- If the total weight of the products contained in the order is greater than 50, then it costs 50 per unit.

ORDER

- Should record the customer user name who is ordering.
- Source address which is customer Address, destination Address where the courier has to be sent can be given.
order type can be Express, Priority or Firstclass
- Should record the Date of the Order placed by the customer

HIGH LEVEL DESIGN





User will create an account in the app,

User will track driver through the application

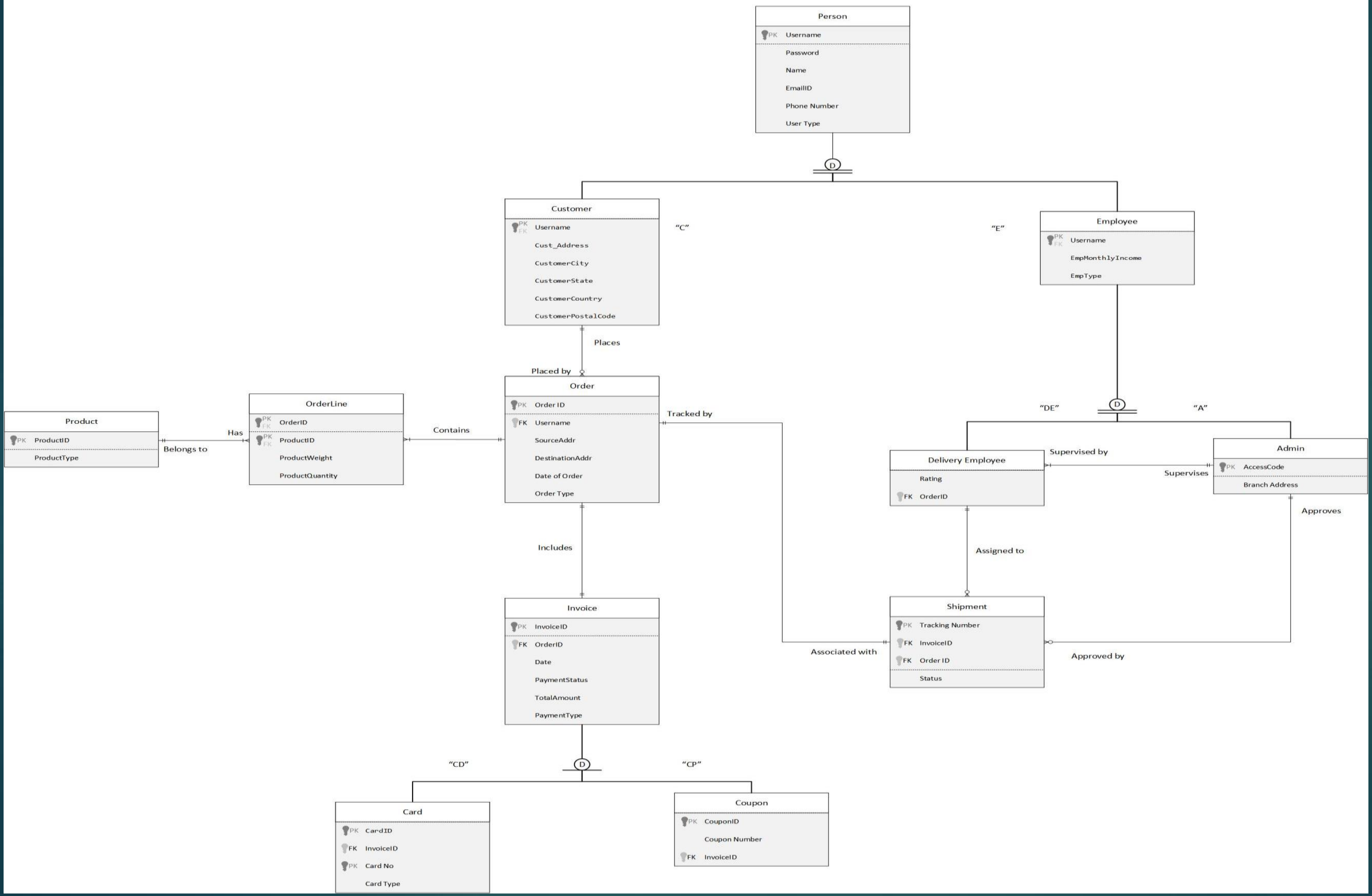
Admin will manage the application process

Driver will pick up the passenger's luggage, and will send a luggage barcode to the user through the application

System will store and retrieve data from the database

ER Diagram

COURIER MANAGEMENT SYSTEM



Person
- PK Username
- Password
- Name
- EmailID
- Phone Number
- User Type

Customer “C”
- PK FK Username
- Cust_Address
- CustomerCity
- CustomerState
- CustomerCountry
- CustomerPostalCode

Employee “E”
- PK FK Username
- EmpMonthlyIncome
- EmpType

Order
- PK Order ID
- FK Username
- SourceAddr
- DestinationAddr
- Date of Order
- Order Type

Product
- PK ProductID
- ProductType

OrderLine
- PK FK OrderID
- PK FK ProductID
- ProductWeight
- ProductQuantity

Delivery Employee “DE”
- Rating
- FK OrderID

Admin “A”
- PK AccessCode
- Branch Address

Shipment
- PK Tracking Number
- FK InvoiceID
- FK Order ID
- Status

Invoice
- PK InvoiceID
- FK OrderID
- Date
- PaymentStatus
- TotalAmount
- PaymentType

Card “CD”
- PK CardID
- FK InvoiceID
- PK Card No
- Card Type

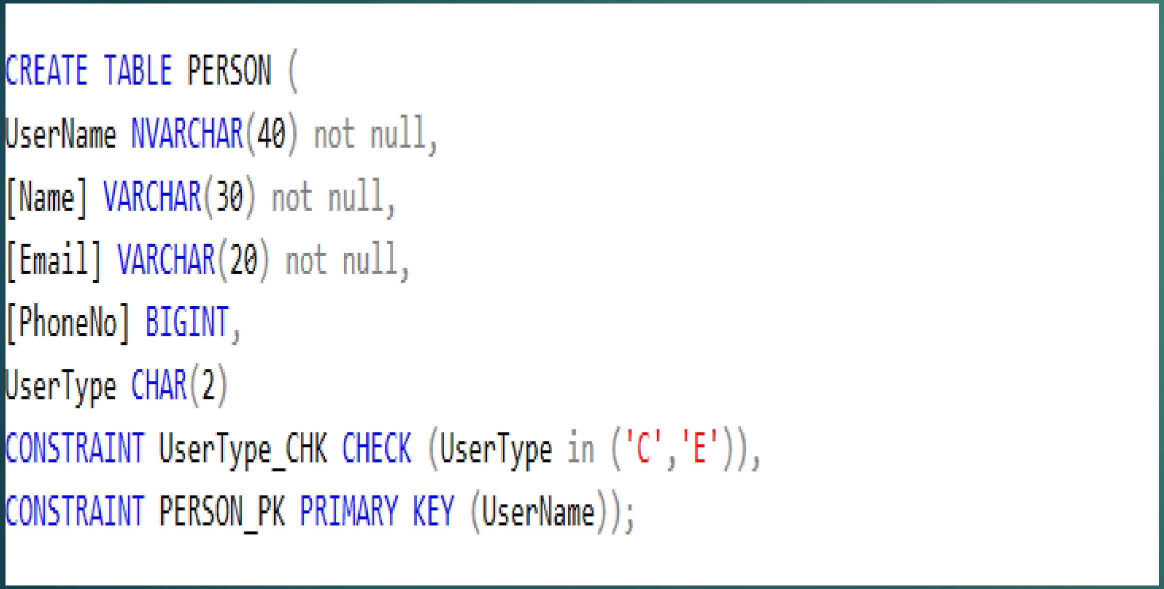
Coupon “CP”
- PK CouponID
- Coupon Number
- FK InvoiceID

Database Implementation

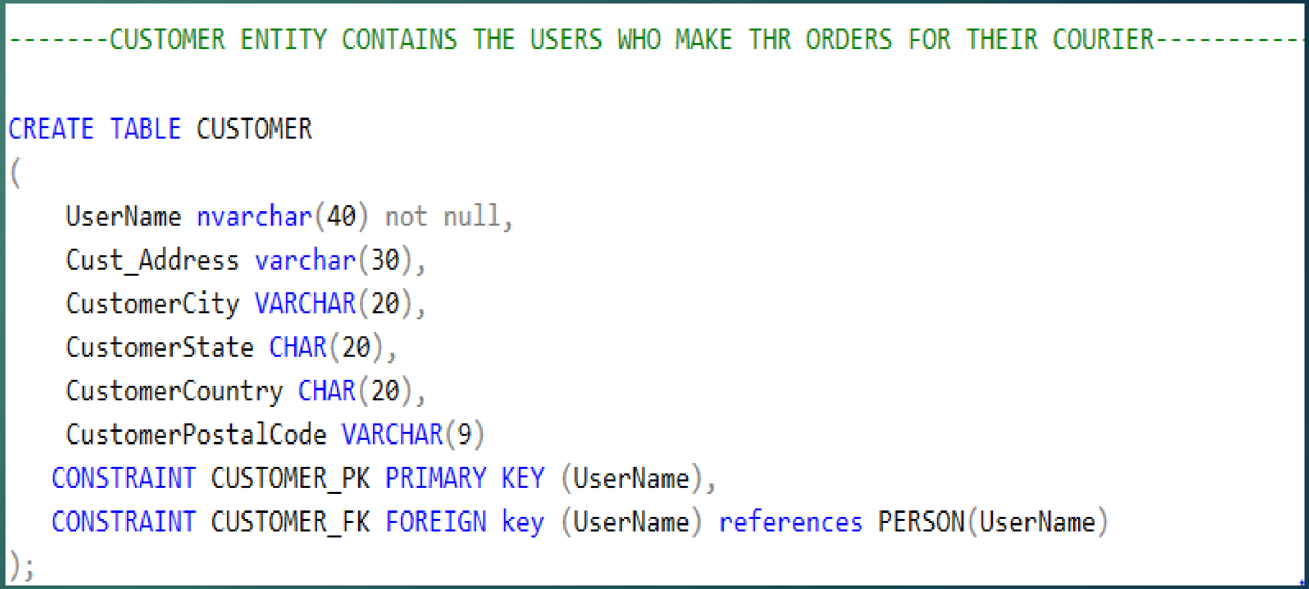
- *Implemented complete database based on CRUD operations done on UI*
- *A routine could be called by the REST API, based on the scenario*

DDL

DDL.sql

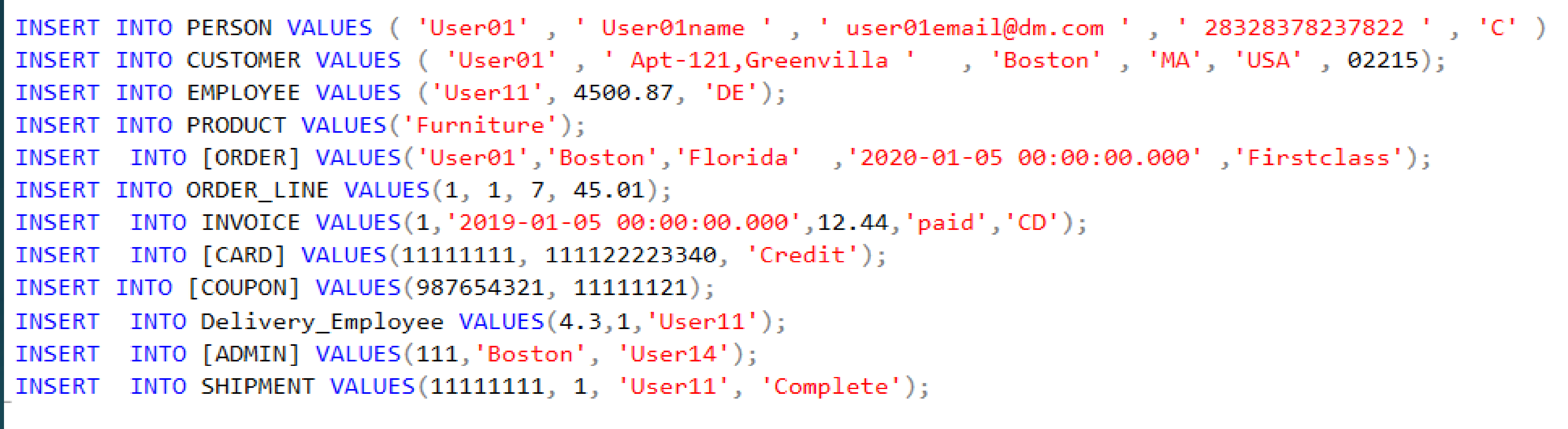


```
CREATE TABLE PERSON (
UserName NVARCHAR(40) not null,
[Name] VARCHAR(30) not null,
[Email] VARCHAR(20) not null,
[PhoneNo] BIGINT,
UserType CHAR(2)
CONSTRAINT UserType_CHK CHECK (UserType in ('C','E')),
CONSTRAINT PERSON_PK PRIMARY KEY (UserName));
```

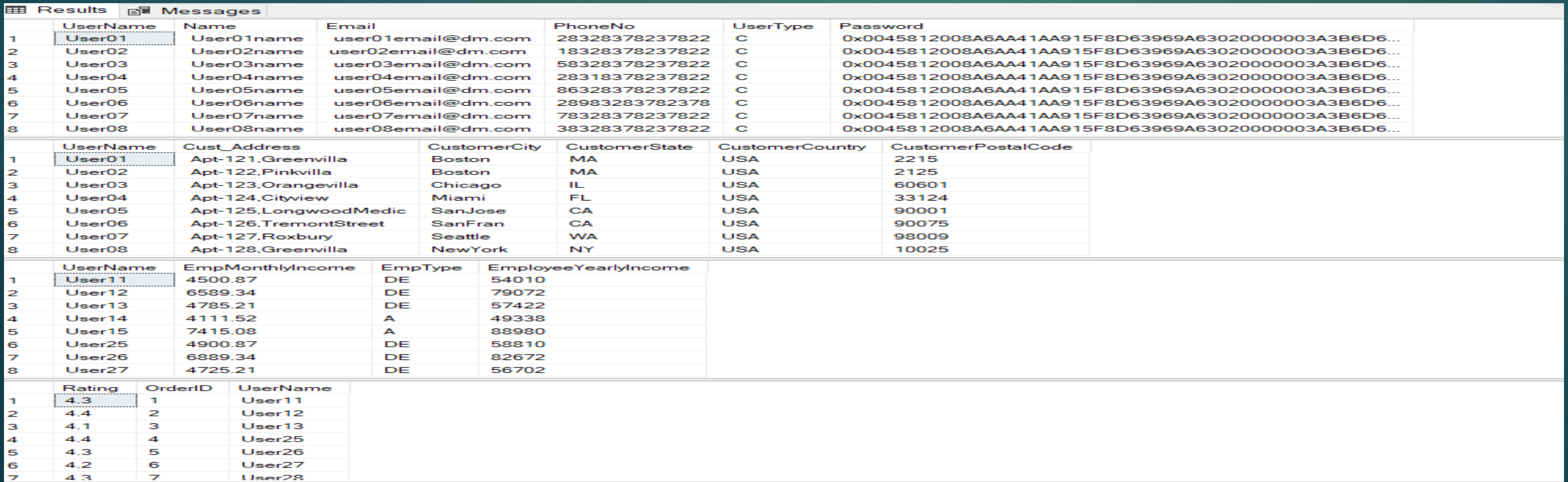


```
-------CUSTOMER ENTITY CONTAINS THE USERS WHO MAKE THR ORDERS FOR THEIR COURIER----------


CREATE TABLE CUSTOMER
(
    UserName nvarchar(40) not null,
    Cust_Address varchar(30),
    CustomerCity VARCHAR(20),
    CustomerState CHAR(20),
    CustomerCountry CHAR(20),
    CustomerPostalCode VARCHAR(9)
   CONSTRAINT CUSTOMER_PK PRIMARY KEY (UserName),
   CONSTRAINT CUSTOMER_FK FOREIGN key (UserName) references PERSON(UserName)
);
```

DML



```sql
INSERT INTO PERSON VALUES ( 'User01' , ' User01name ' , ' user01email@dm.com ' , ' 28328378237822 ' , 'C' )
INSERT INTO CUSTOMER VALUES ( 'User01' , ' Apt-121,Greenvilla '   , 'Boston' , 'MA', 'USA' , 02215);
INSERT INTO EMPLOYEE VALUES ('User11', 4500.87, 'DE');
INSERT INTO PRODUCT VALUES('Furniture');
INSERT  INTO [ORDER] VALUES('User01','Boston','Florida'  ,'2020-01-05 00:00:00.000' ,'Firstclass');
INSERT INTO ORDER_LINE VALUES(1, 1, 7, 45.01);
INSERT  INTO INVOICE VALUES(1,'2019-01-05 00:00:00.000',12.44,'paid','CD');
INSERT  INTO [CARD] VALUES(11111111, 111122223340, 'Credit');
INSERT INTO [COUPON] VALUES(987654321, 11111121);
INSERT  INTO Delivery_Employee VALUES(4.3,1,'User11');
INSERT  INTO [ADMIN] VALUES(111,'Boston', 'User14');
INSERT  INTO SHIPMENT VALUES(11111111, 1, 'User11', 'Complete');
```



	UserName	Name	Email	PhoneNo	UserType	Password
1	User01	User01name	user01email@dm.com	28328378237822	C	0x0045812008A6AA41AA915F8D63969A63020000003A3B6D6...
2	User02	User02name	user02email@dm.com	18328378237822	C	0x0045812008A6AA41AA915F8D63969A63020000003A3B6D6...
3	User03	User03name	user03email@dm.com	58328378237822	C	0x0045812008A6AA41AA915F8D63969A63020000003A3B6D6...
4	User04	User04name	user04email@dm.com	28318378237822	C	0x0045812008A6AA41AA915F8D63969A63020000003A3B6D6...
5	User05	User05name	user05email@dm.com	86328378237822	C	0x0045812008A6AA41AA915F8D63969A63020000003A3B6D6...
6	User06	User06name	user06email@dm.com	28983283782378	C	0x0045812008A6AA41AA915F8D63969A63020000003A3B6D6...
7	User07	User07name	user07email@dm.com	78328378237822	C	0x0045812008A6AA41AA915F8D63969A63020000003A3B6D6...
8	User08	User08name	user08email@dm.com	38328378237822	C	0x0045812008A6AA41AA915F8D63969A63020000003A3B6D6...

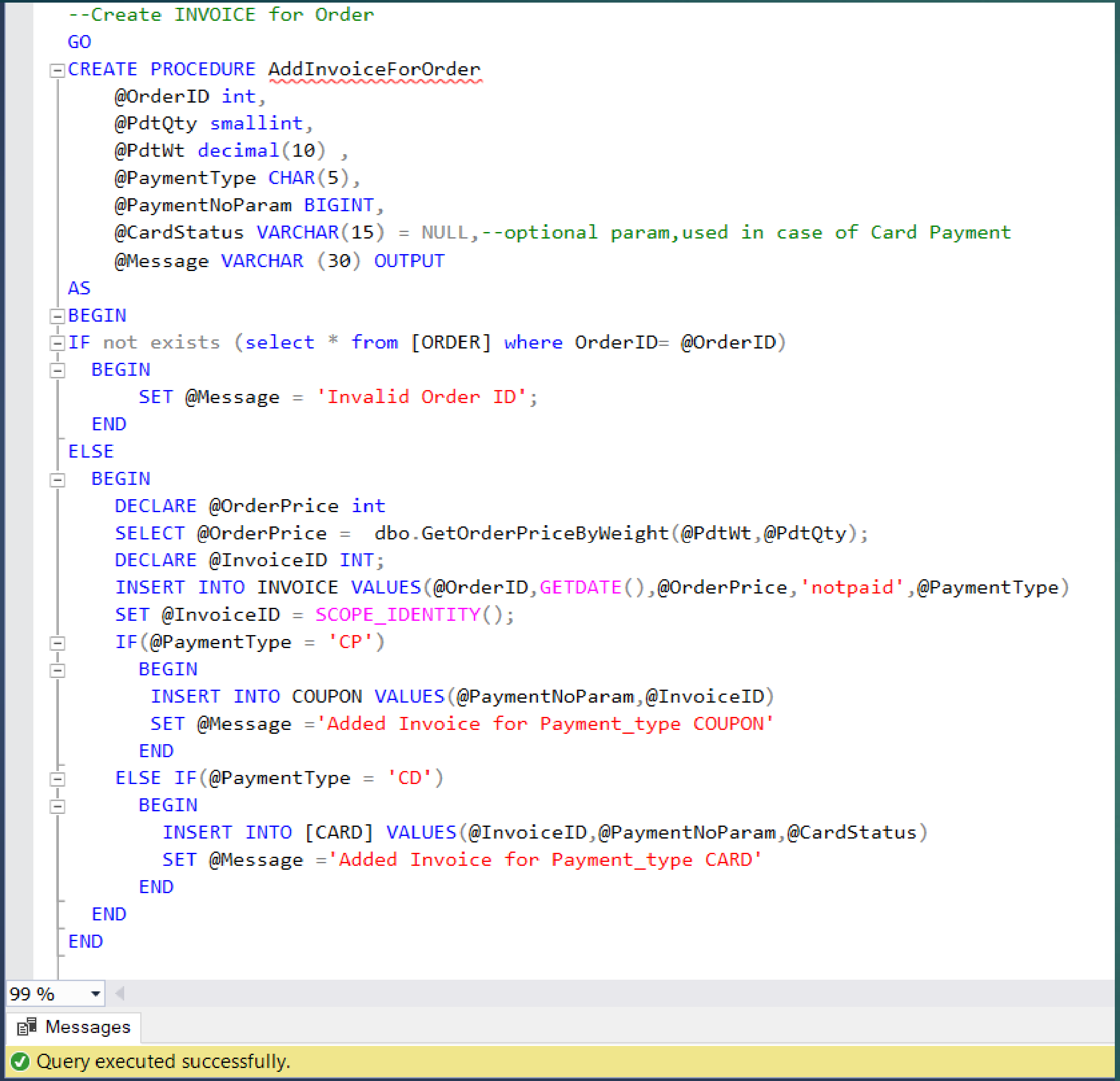
	UserName	Cust_Address	CustomerCity	CustomerState	CustomerCountry	CustomerPostalCode
1	User01	Apt-121,Greenvilla	Boston	MA	USA	2215
2	User02	Apt-122,Pinkvilla	Boston	MA	USA	2125
3	User03	Apt-123,Orangevilla	Chicago	IL	USA	60601
4	User04	Apt-124,Cityview	Miami	FL	USA	33124
5	User05	Apt-125,LongwoodMedic	SanJose	CA	USA	90001
6	User06	Apt-126,TremontStreet	SanFran	CA	USA	90075
7	User07	Apt-127,Roxbury	Seattle	WA	USA	98009
8	User08	Apt-128,Greenvilla	NewYork	NY	USA	10025

	UserName	EmpMonthlyIncome	EmpType	EmployeeYearlyIncome
1	User11	4500.87	DE	54010
2	User12	6589.34	DE	79072
3	User13	4785.21	DE	57422
4	User14	4111.52	A	49338
5	User15	7415.08	A	88980
6	User25	4900.87	DE	58810
7	User26	6889.34	DE	82672
8	User27	4725.21	DE	56702

	Rating	OrderID	UserName
1	4.3	1	User11
2	4.4	2	User12
3	4.1	3	User13
4	4.4	4	User25
5	4.3	5	User26
6	4.2	6	User27
7	4.3	7	User28

DML.sql

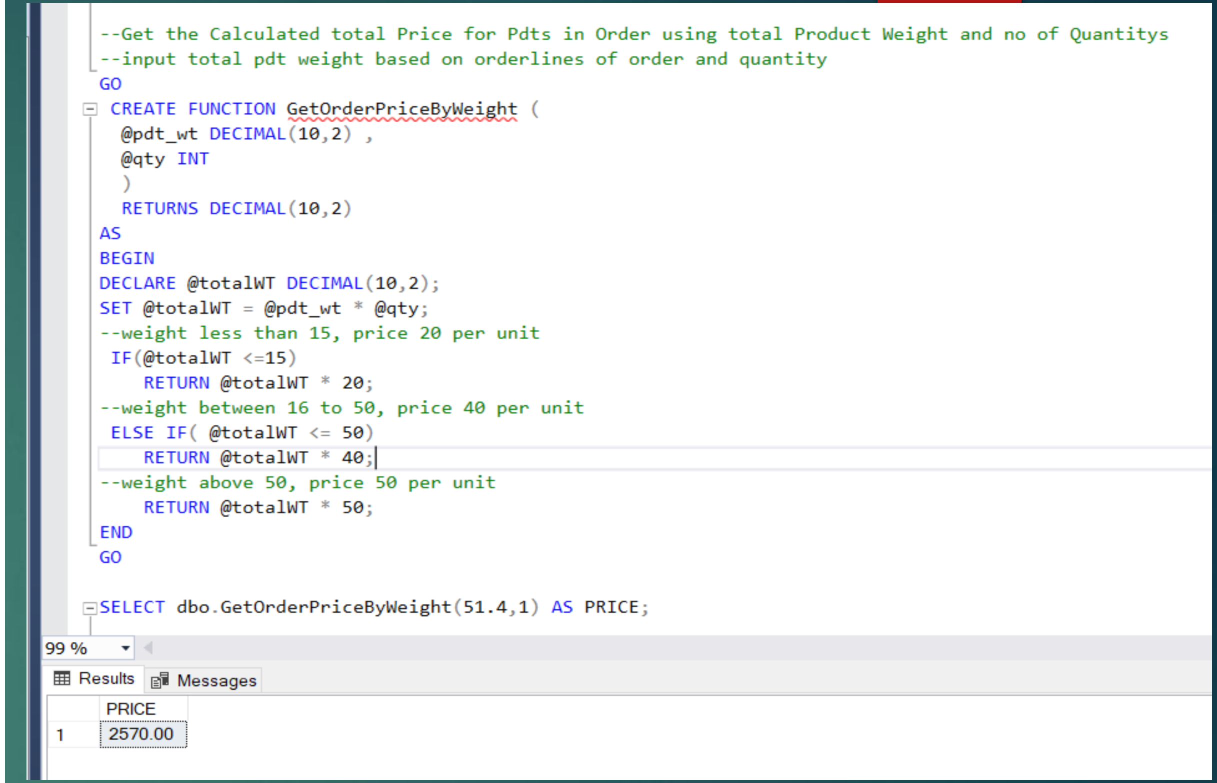
STORED PROCEDURE



```sql
--Create INVOICE for Order
GO
CREATE PROCEDURE AddInvoiceForOrder
    @OrderID int,
    @PdtQty smallint,
    @PdtWt decimal(10) ,
    @PaymentType CHAR(5),
    @PaymentNoParam BIGINT,
    @CardStatus VARCHAR(15) = NULL,--optional param,used in case of Card Payment
    @Message VARCHAR (30) OUTPUT
AS
BEGIN
IF not exists (select * from [ORDER] where OrderID= @OrderID)
  BEGIN
      SET @Message = 'Invalid Order ID';
  END
ELSE
  BEGIN
    DECLARE @OrderPrice int
    SELECT @OrderPrice =  dbo.GetOrderPriceByWeight(@PdtWt,@PdtQty);
    DECLARE @InvoiceID INT;
    INSERT INTO INVOICE VALUES(@OrderID,GETDATE(),@OrderPrice,'notpaid',@PaymentType)
    SET @InvoiceID = SCOPE_IDENTITY();
    IF(@PaymentType = 'CP')
      BEGIN
       INSERT INTO COUPON VALUES(@PaymentNoParam,@InvoiceID)
       SET @Message ='Added Invoice for Payment_type COUPON'
      END
    ELSE IF(@PaymentType = 'CD')
      BEGIN
        INSERT INTO [CARD] VALUES(@InvoiceID,@PaymentNoParam,@CardStatus)
        SET @Message ='Added Invoice for Payment_type CARD'
      END
  END
END
```

```
Query executed successfully.
```

FUNCTION



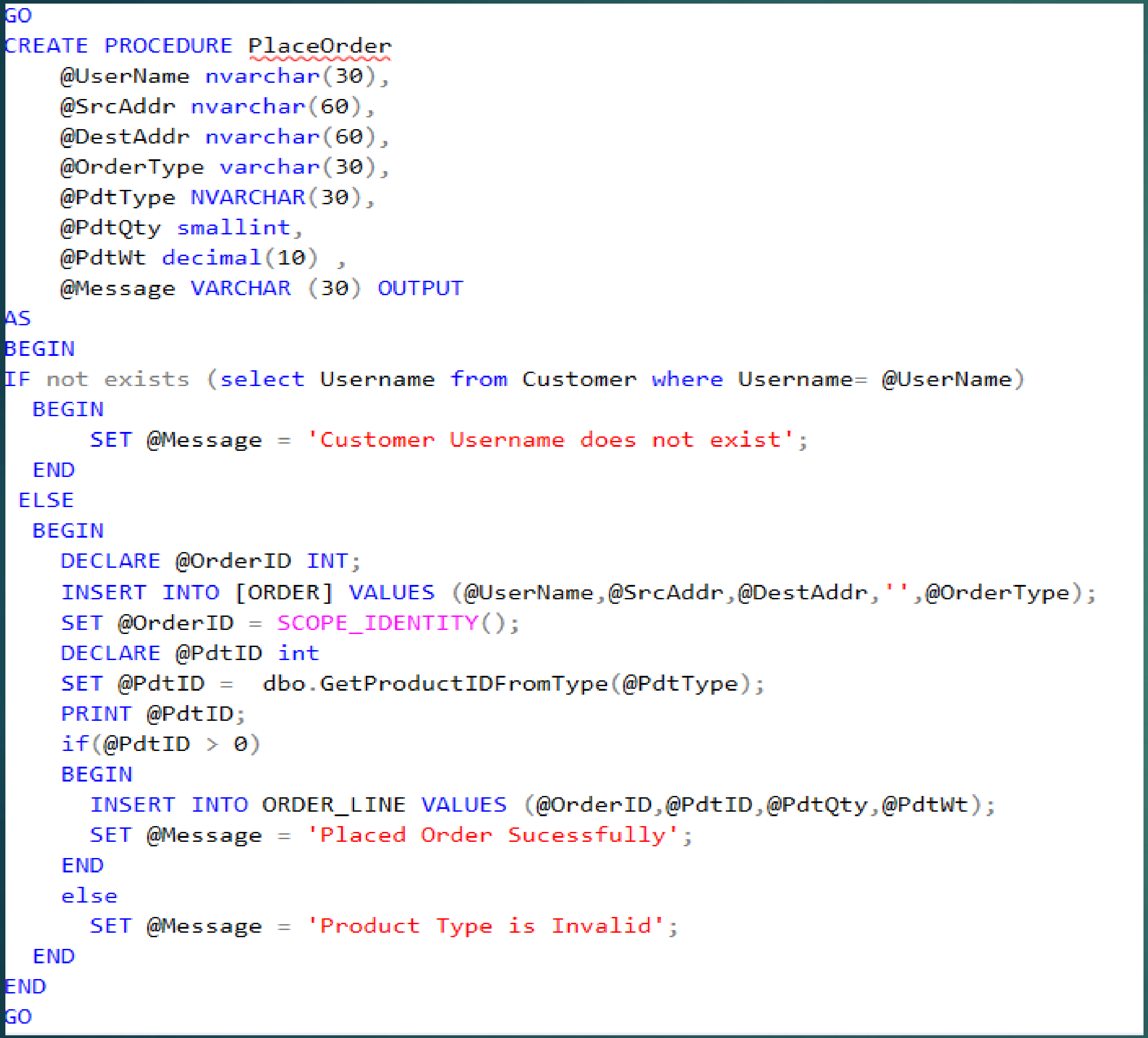
```sql
--Get the Calculated total Price for Pdts in Order using total Product Weight and no of Quantitys
--input total pdt weight based on orderlines of order and quantity
GO
 CREATE FUNCTION GetOrderPriceByWeight (
  @pdt_wt DECIMAL(10,2) ,
  @qty INT
  )
  RETURNS DECIMAL(10,2)
AS
BEGIN
DECLARE @totalWT DECIMAL(10,2);
SET @totalWT = @pdt_wt * @qty;
--weight less than 15, price 20 per unit
 IF(@totalWT <=15)
    RETURN @totalWT * 20;
--weight between 16 to 50, price 40 per unit
 ELSE IF( @totalWT <= 50)
    RETURN @totalWT * 40;
--weight above 50, price 50 per unit
    RETURN @totalWT * 50;
END
GO

SELECT dbo.GetOrderPriceByWeight(51.4,1) AS PRICE;
```

	PRICE
1	2570.00

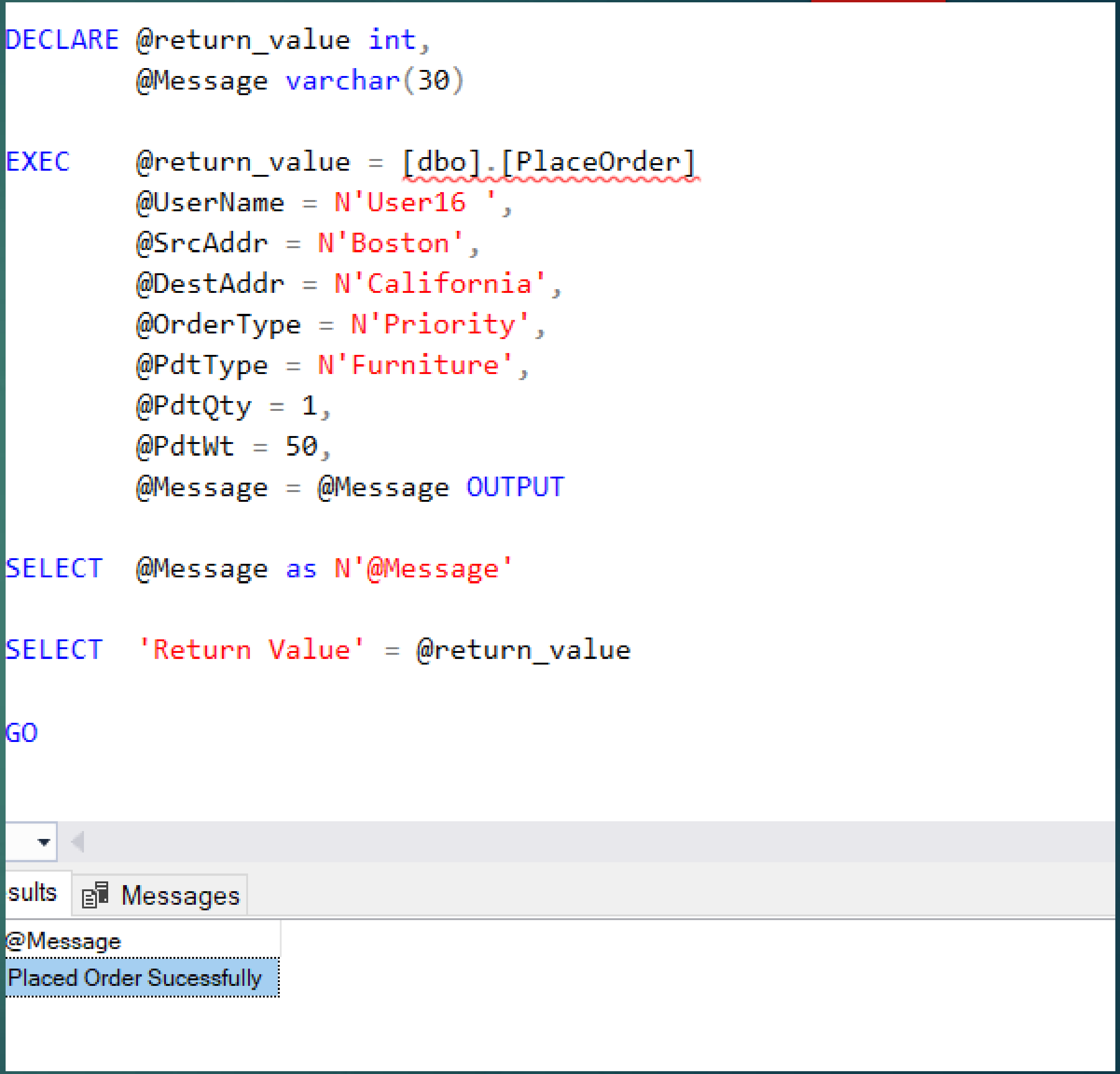
Stored Procedure to create Invoice

- Used function to calculate Price for order.
- Price calculated acc to business requirements.

Stored Procedure : Customer Placing an order

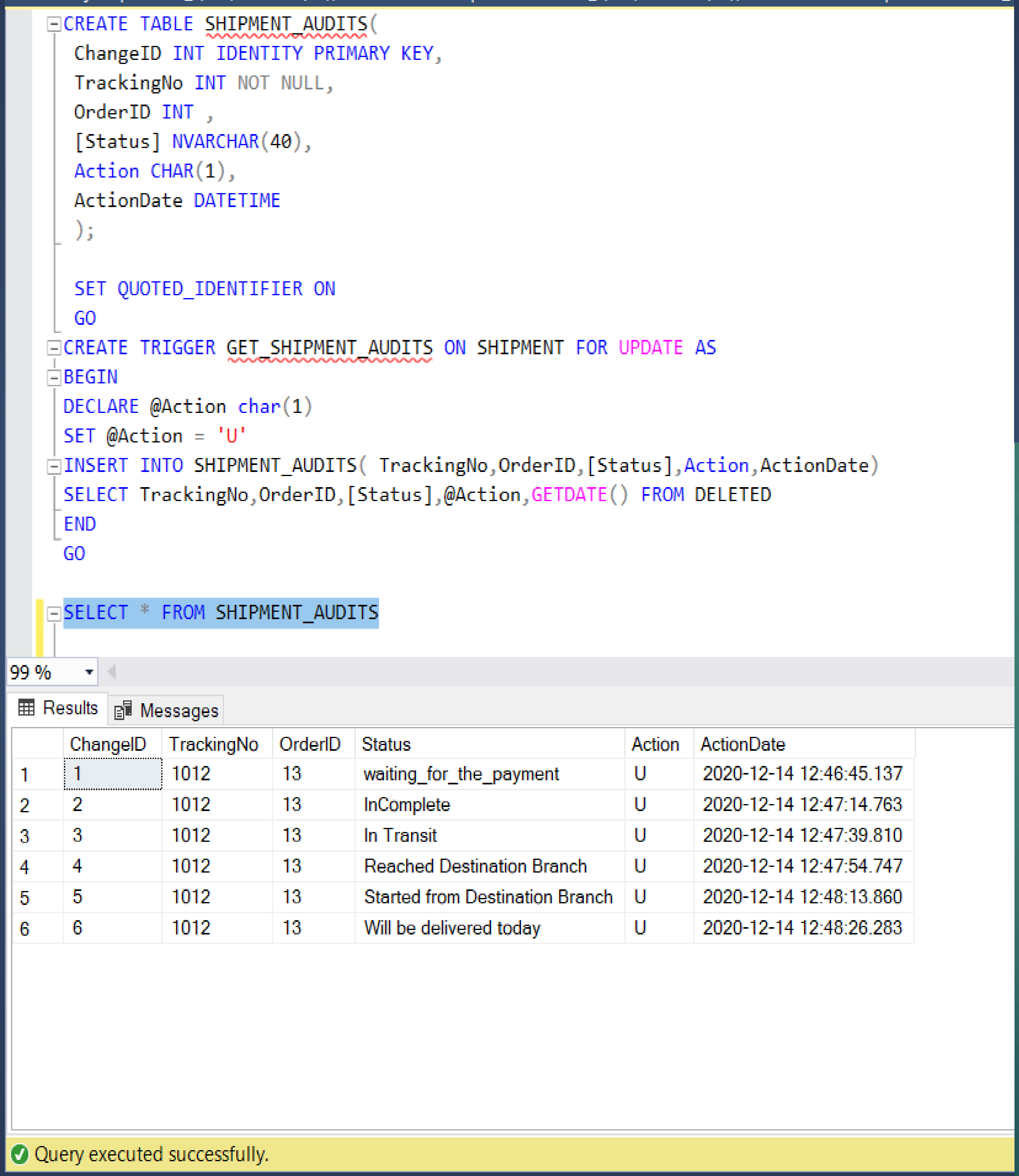


```sql
GO
CREATE PROCEDURE PlaceOrder
    @UserName nvarchar(30),
    @SrcAddr nvarchar(60),
    @DestAddr nvarchar(60),
    @OrderType varchar(30),
    @PdtType NVARCHAR(30),
    @PdtQty smallint,
    @PdtWt decimal(10) ,
    @Message VARCHAR (30) OUTPUT
AS
BEGIN
IF not exists (select Username from Customer where Username= @UserName)
  BEGIN
      SET @Message = 'Customer Username does not exist';
  END
 ELSE
  BEGIN
    DECLARE @OrderID INT;
    INSERT INTO [ORDER] VALUES (@UserName,@SrcAddr,@DestAddr,'',@OrderType);
    SET @OrderID = SCOPE_IDENTITY();
    DECLARE @PdtID int
    SET @PdtID =  dbo.GetProductIDFromType(@PdtType);
    PRINT @PdtID;
    if(@PdtID > 0)
    BEGIN
      INSERT INTO ORDER_LINE VALUES (@OrderID,@PdtID,@PdtQty,@PdtWt);
      SET @Message = 'Placed Order Sucessfully';
    END
    else
      SET @Message = 'Product Type is Invalid';
  END
END
GO
```



```sql
DECLARE @return_value int,
        @Message varchar(30)

EXEC    @return_value = [dbo].[PlaceOrder]
        @UserName = N'User16 ',
        @SrcAddr = N'Boston',
        @DestAddr = N'California',
        @OrderType = N'Priority',
        @PdtType = N'Furniture',
        @PdtQty = 1,
        @PdtWt = 50,
        @Message = @Message OUTPUT

SELECT  @Message as N'@Message'

SELECT  'Return Value' = @return_value

GO
```

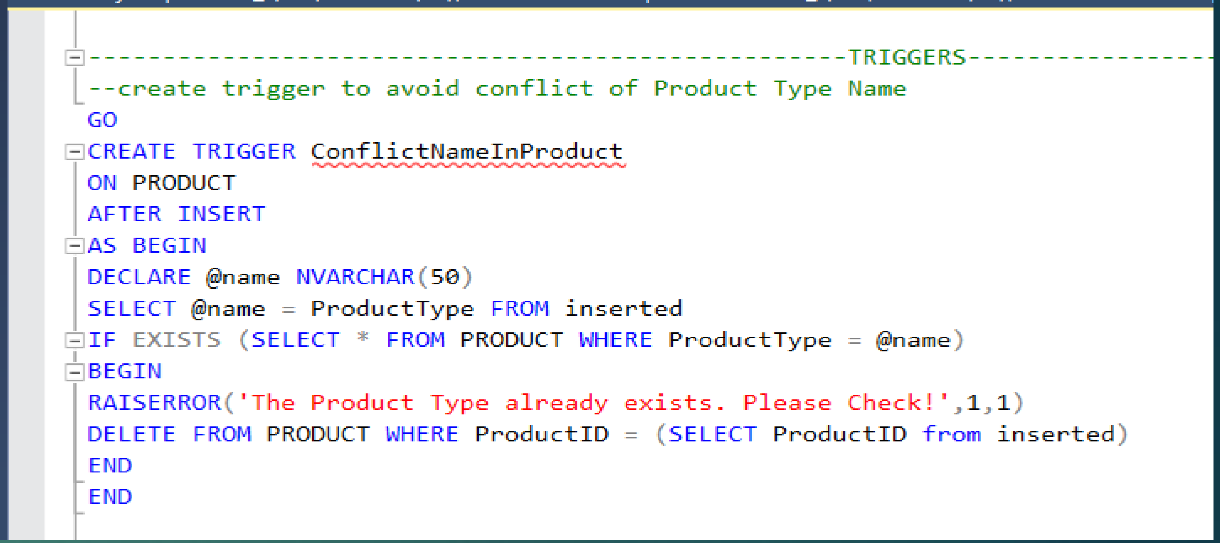
```
@Message
Placed Order Sucessfully
```



```
CREATE TABLE SHIPMENT_AUDITS(
 ChangeID INT IDENTITY PRIMARY KEY,
 TrackingNo INT NOT NULL,
 OrderID INT ,
 [Status] NVARCHAR(40),
 Action CHAR(1),
 ActionDate DATETIME
 );

 SET QUOTED_IDENTIFIER ON
 GO
CREATE TRIGGER GET_SHIPMENT_AUDITS ON SHIPMENT FOR UPDATE AS
BEGIN
DECLARE @Action char(1)
SET @Action = 'U'
INSERT INTO SHIPMENT_AUDITS( TrackingNo,OrderID,[Status],Action,ActionDate)
SELECT TrackingNo,OrderID,[Status],@Action,GETDATE() FROM DELETED
END
GO

SELECT * FROM SHIPMENT_AUDITS
```

	ChangeID	TrackingNo	OrderID	Status	Action	ActionDate
1	1	1012	13	waiting_for_the_payment	U	2020-12-14 12:46:45.137
2	2	1012	13	InComplete	U	2020-12-14 12:47:14.763
3	3	1012	13	In Transit	U	2020-12-14 12:47:39.810
4	4	1012	13	Reached Destination Branch	U	2020-12-14 12:47:54.747
5	5	1012	13	Started from Destination Branch	U	2020-12-14 12:48:13.860
6	6	1012	13	Will be delivered today	U	2020-12-14 12:48:26.283

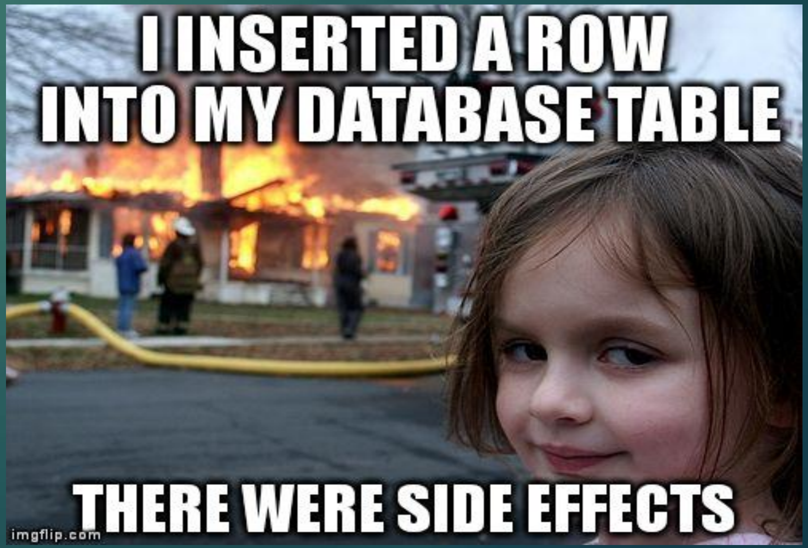
Query executed successfully.



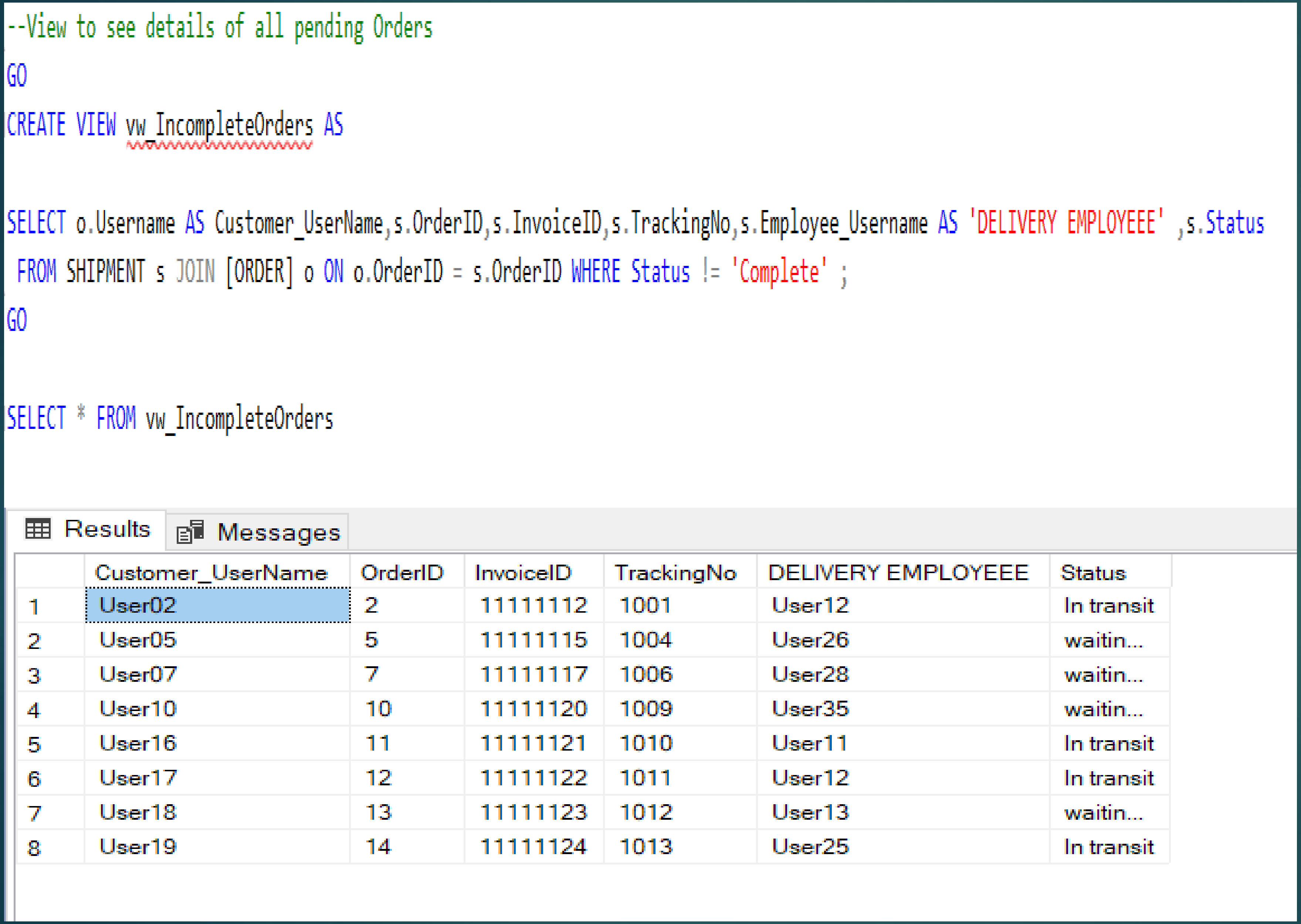
```
-----------------------------------------------------TRIGGERS----------------
--create trigger to avoid conflict of Product Type Name
GO
CREATE TRIGGER ConflictNameInProduct
ON PRODUCT
AFTER INSERT
AS BEGIN
DECLARE @name NVARCHAR(50)
SELECT @name = ProductType FROM inserted
IF EXISTS (SELECT * FROM PRODUCT WHERE ProductType = @name)
BEGIN
RAISERROR('The Product Type already exists. Please Check!',1,1)
DELETE FROM PRODUCT WHERE ProductID = (SELECT ProductID from inserted)
END
END
```

TRIGGERS

- Create Shipment_Audit to track Shipment on Update action
- To avoid insertion of duplicate Product type name



Sample View : View Incomplete Orders



```sql
--View to see details of all pending Orders
GO
CREATE VIEW vw_IncompleteOrders AS

SELECT o.Username AS Customer_UserName,s.OrderID,s.InvoiceID,s.TrackingNo,s.Employee_Username AS 'DELIVERY EMPLOYEEE' ,s.Status
 FROM SHIPMENT s JOIN [ORDER] o ON o.OrderID = s.OrderID WHERE Status != 'Complete' ;
GO

SELECT * FROM vw_IncompleteOrders
```

	Customer_UserName	OrderID	InvoiceID	TrackingNo	DELIVERY EMPLOYEEE	Status
1	User02	2	11111112	1001	User12	In transit
2	User05	5	11111115	1004	User26	waitin...
3	User07	7	11111117	1006	User28	waitin...
4	User10	10	11111120	1009	User35	waitin...
5	User16	11	11111121	1010	User11	In transit
6	User17	12	11111122	1011	User12	In transit
7	User18	13	11111123	1012	User13	waitin...
8	User19	14	11111124	1013	User25	In transit

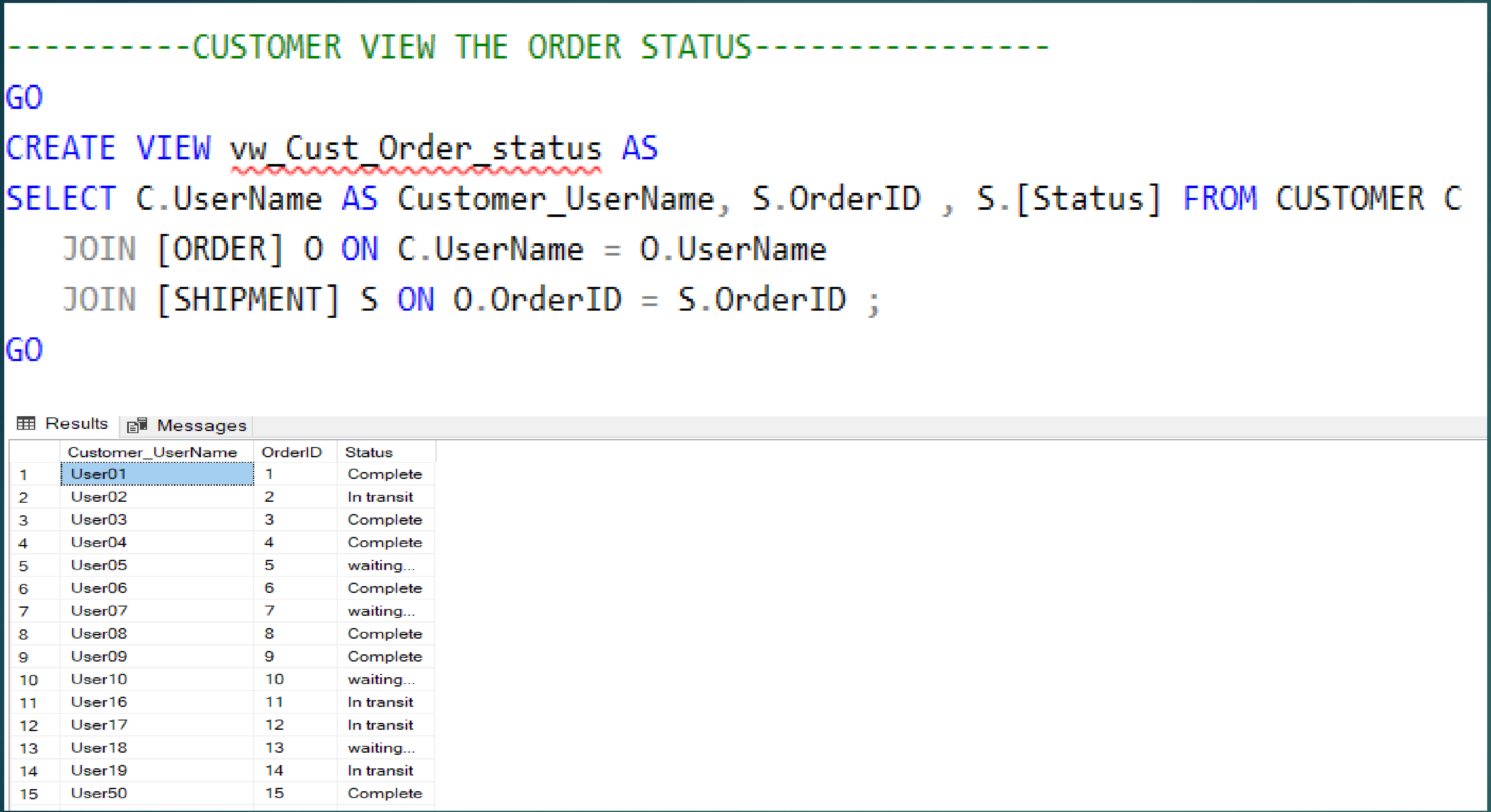
Programmability.sql

VIEWS

-> Created a VIEW to track all the Incomplete Orders

-> Retrieved result set by the Inner Join of Shipment and order table

Sample View : Customer can view the status of order



```
----------CUSTOMER VIEW THE ORDER STATUS----------------
GO
CREATE VIEW vw_Cust_Order_status AS
SELECT C.UserName AS Customer_UserName, S.OrderID , S.[Status] FROM CUSTOMER C
   JOIN [ORDER] O ON C.UserName = O.UserName
   JOIN [SHIPMENT] S ON O.OrderID = S.OrderID ;
GO
```

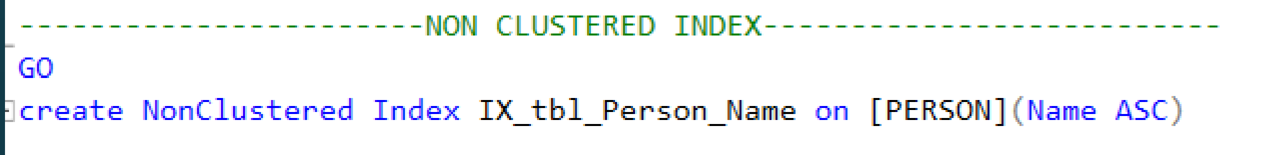
	Customer_UserName	OrderID	Status
1	User01	1	Complete
2	User02	2	In transit
3	User03	3	Complete
4	User04	4	Complete
5	User05	5	waiting...
6	User06	6	Complete
7	User07	7	waiting...
8	User08	8	Complete
9	User09	9	Complete
10	User10	10	waiting...
11	User16	11	In transit
12	User17	12	In transit
13	User18	13	waiting...
14	User19	14	In transit
15	User50	15	Complete

VIEWS

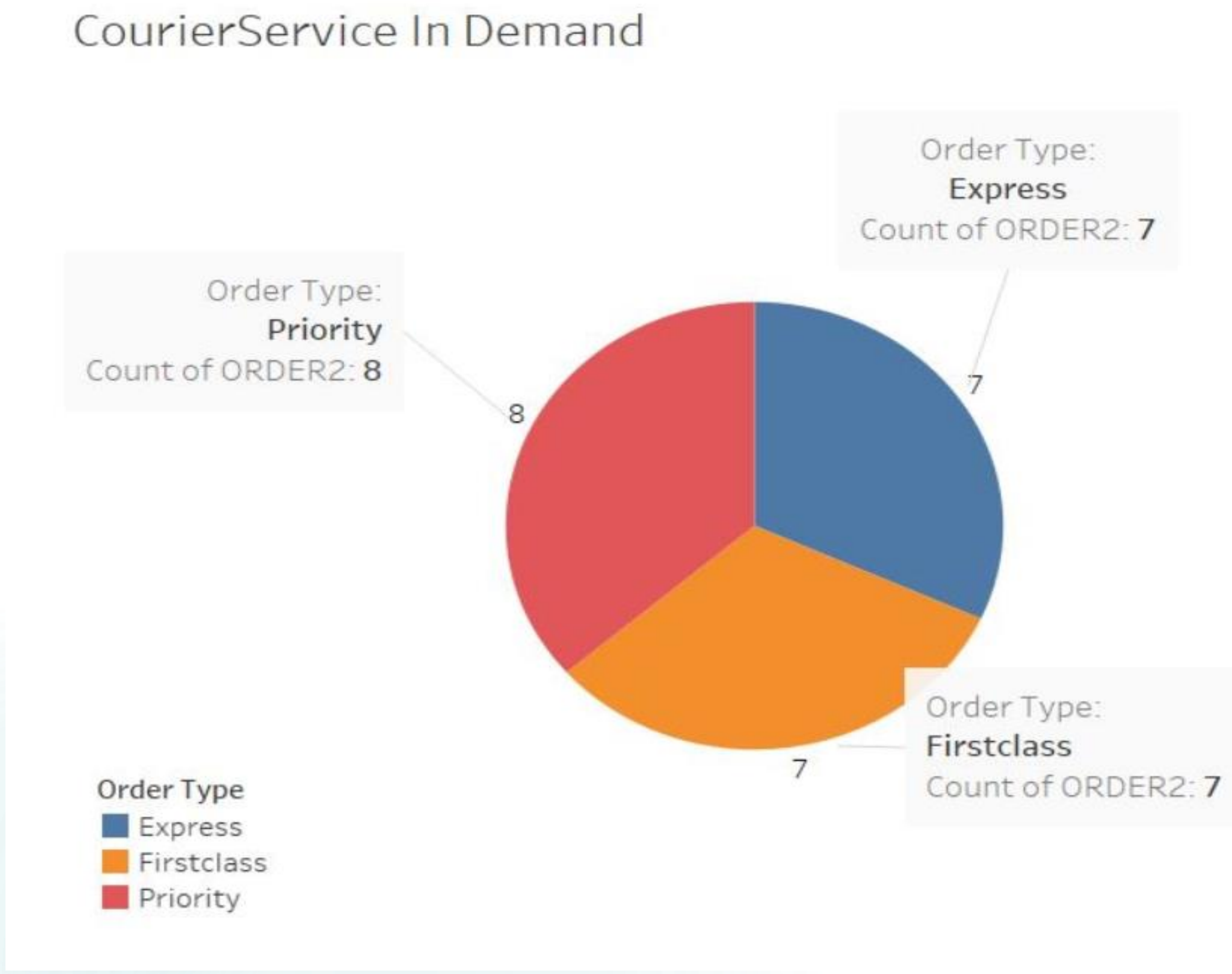
-> Created a VIEW for customer to view the order status

-> Retrieved result set by the Inner Join of customer,order and Shipment

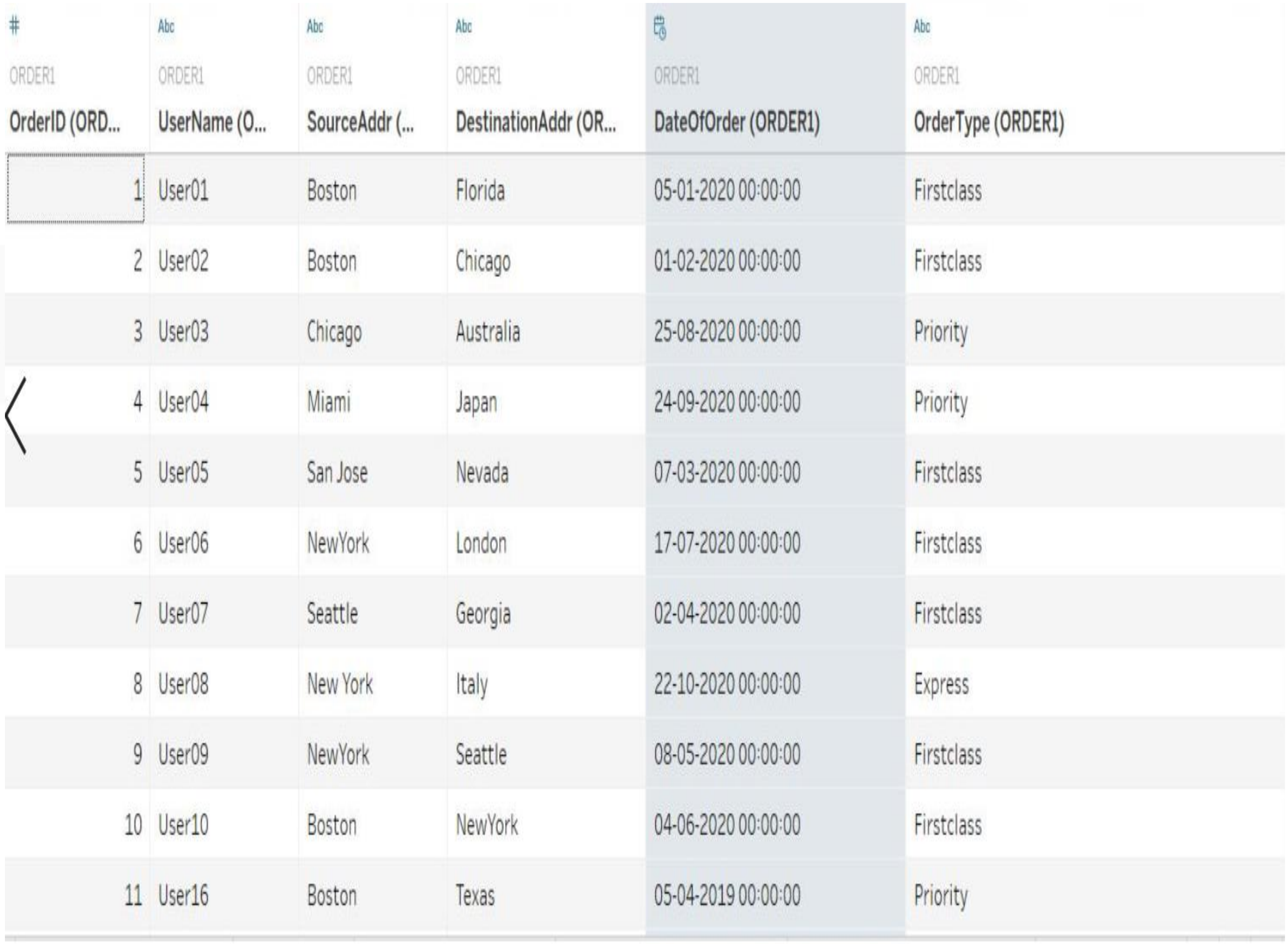
Non Clustered Index



```
-----------------------NON CLUSTERED INDEX--------------------------
GO
create NonClustered Index IX_tbl_Person_Name on [PERSON](Name ASC)
```



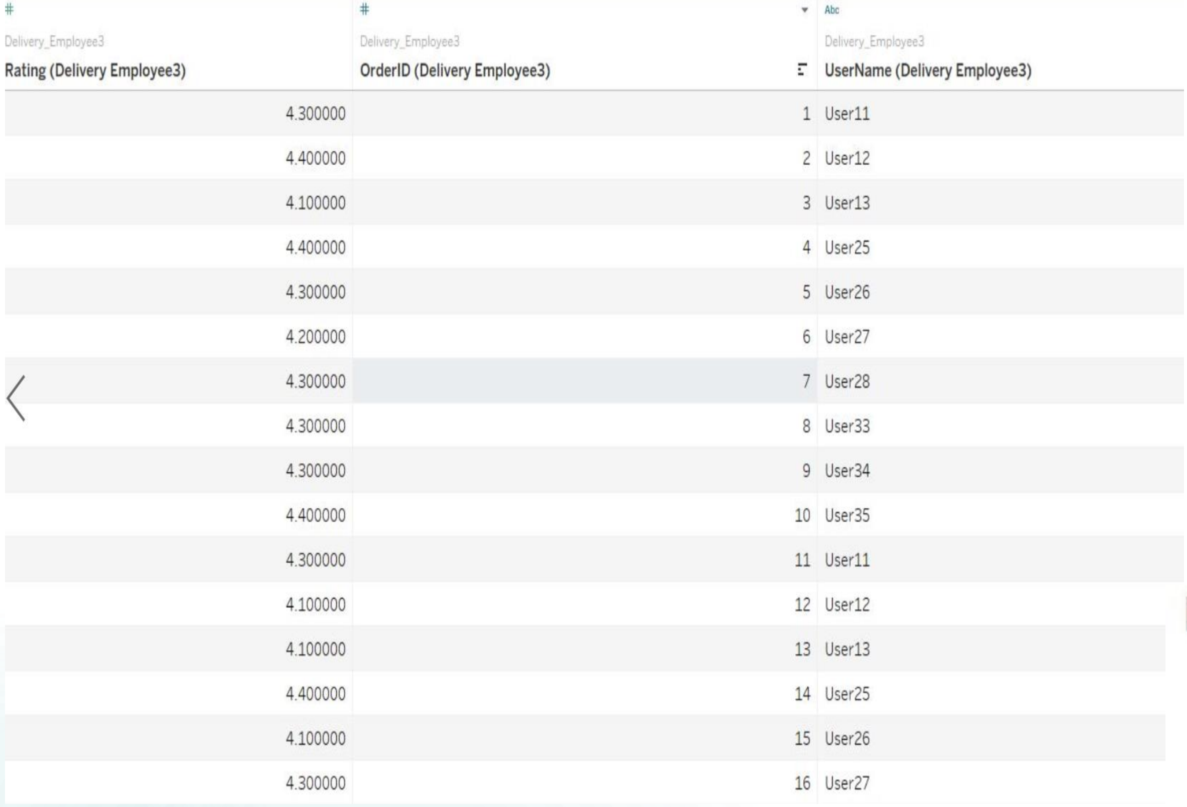
Improve the least used service



#	Abc	Abc	Abc		Abc
ORDER1	ORDER1	ORDER1	ORDER1	ORDER1	ORDER1
OrderID (ORD...	UserName (O...	SourceAddr (...	DestinationAddr (OR...	DateOfOrder (ORDER1)	OrderType (ORDER1)
1	User01	Boston	Florida	05-01-2020 00:00:00	Firstclass
2	User02	Boston	Chicago	01-02-2020 00:00:00	Firstclass
3	User03	Chicago	Australia	25-08-2020 00:00:00	Priority
4	User04	Miami	Japan	24-09-2020 00:00:00	Priority
5	User05	San Jose	Nevada	07-03-2020 00:00:00	Firstclass
6	User06	NewYork	London	17-07-2020 00:00:00	Firstclass
7	User07	Seattle	Georgia	02-04-2020 00:00:00	Firstclass
8	User08	New York	Italy	22-10-2020 00:00:00	Express
9	User09	NewYork	Seattle	08-05-2020 00:00:00	Firstclass
10	User10	Boston	NewYork	04-06-2020 00:00:00	Firstclass
11	User16	Boston	Texas	05-04-2019 00:00:00	Priority

Can be used to track the service in demand.

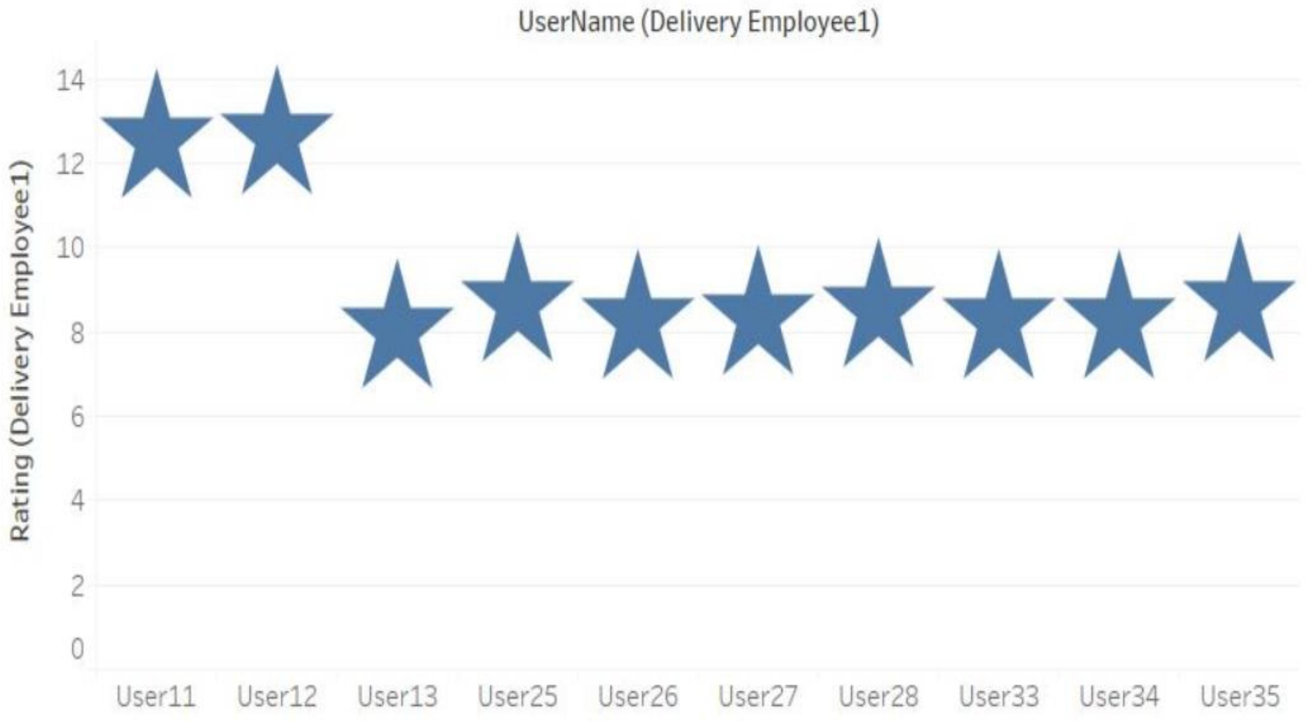
Employee Satisfaction is important!



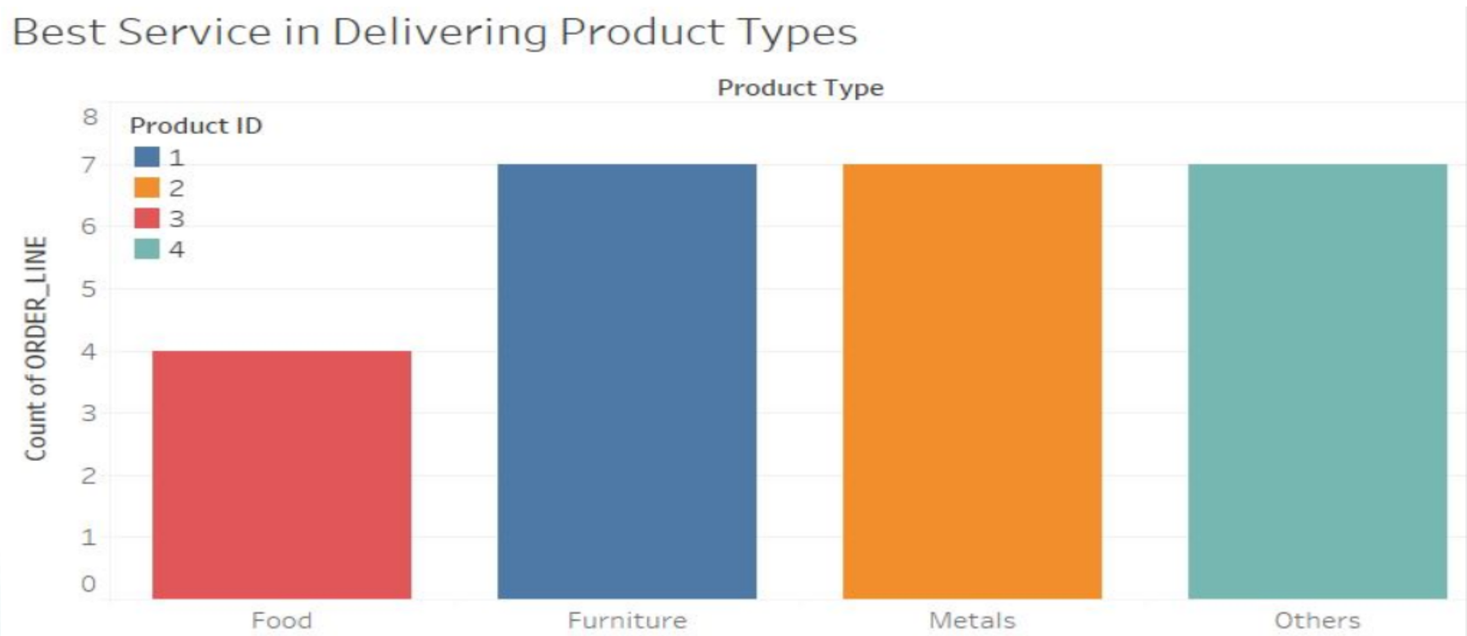
Delivery_Employee3 Rating (Delivery Employee3)	Delivery_Employee3 OrderID (Delivery Employee3)	Delivery_Employee3 UserName (Delivery Employee3)
4.300000	1	User11
4.400000	2	User12
4.100000	3	User13
4.400000	4	User25
4.300000	5	User26
4.200000	6	User27
4.300000	7	User28
4.300000	8	User33
4.300000	9	User34
4.400000	10	User35
4.300000	11	User11
4.100000	12	User12
4.100000	13	User13
4.400000	14	User25
4.100000	15	User26
4.300000	16	User27

The supervisor could recognize his star employee

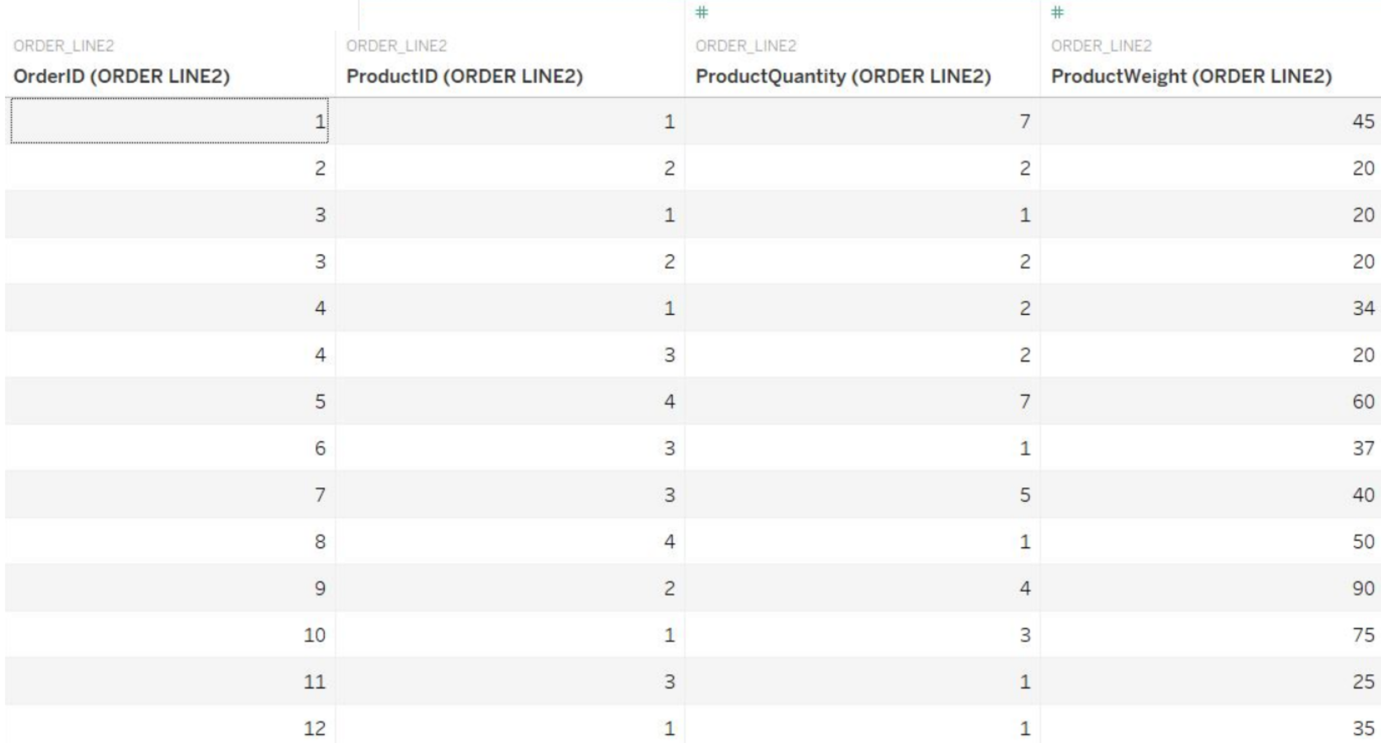
Most Rated Delivery Employee



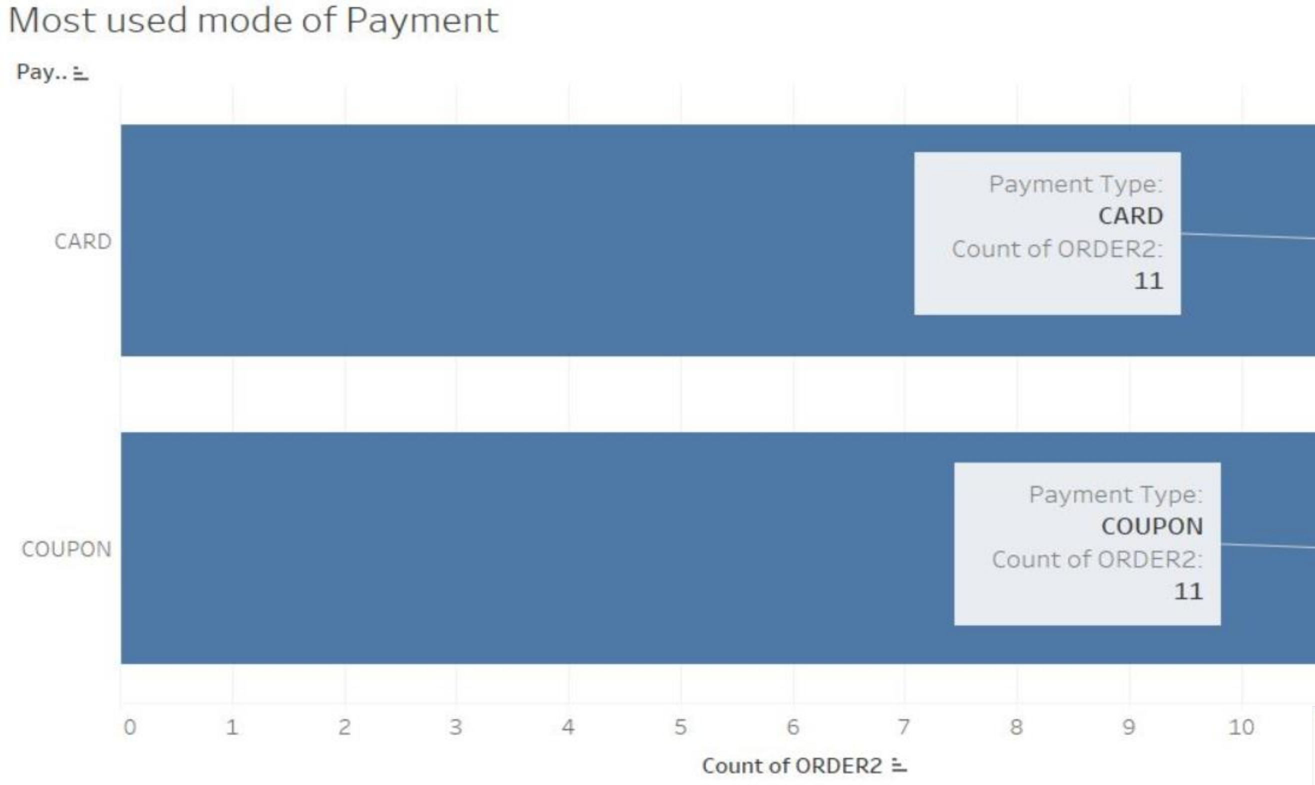
Plan to improve profits


Best Service in Delivering Product Types

Could check for the most delivered product type

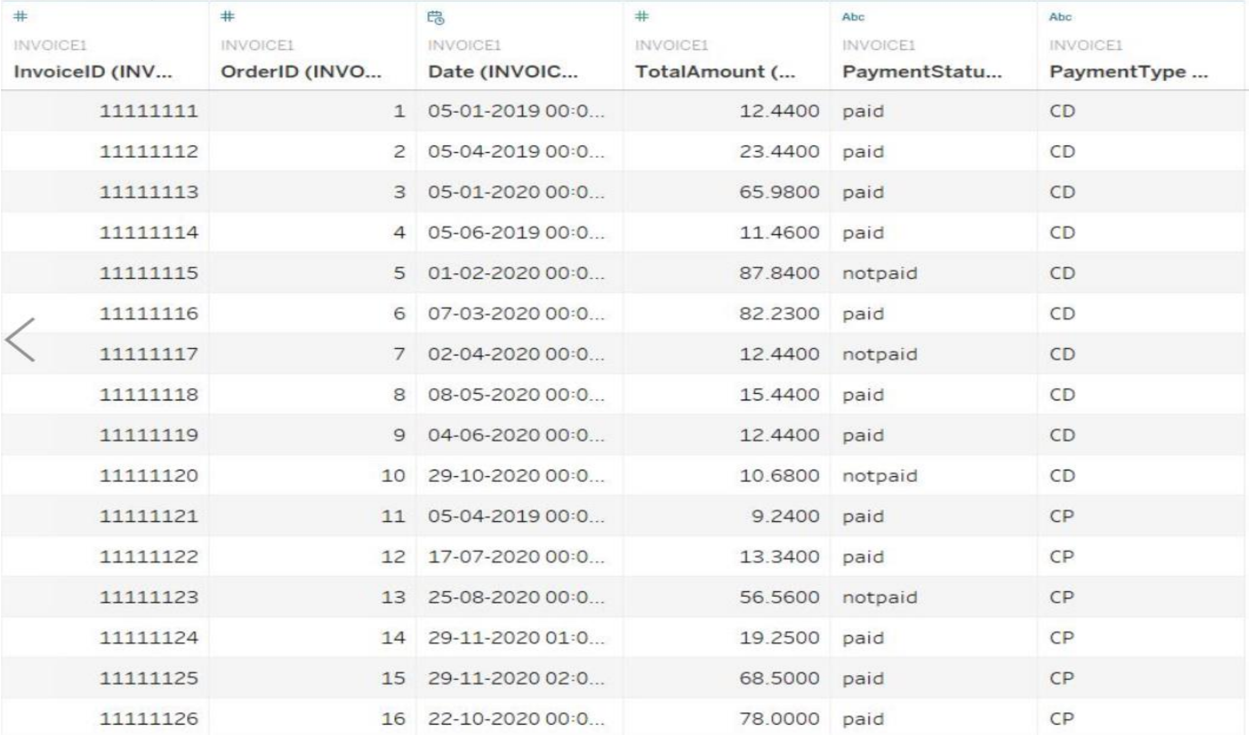


		#	#
ORDER_LINE2	ORDER_LINE2	ORDER_LINE2	ORDER_LINE2
OrderID (ORDER LINE2)	ProductID (ORDER LINE2)	ProductQuantity (ORDER LINE2)	ProductWeight (ORDER LINE2)
1	1	7	45
2	2	2	20
3	1	1	20
3	2	2	20
4	1	2	34
4	3	2	20
5	4	7	60
6	3	1	37
7	3	5	40
8	4	1	50
9	2	4	90
10	1	3	75
11	3	1	25
12	1	1	35

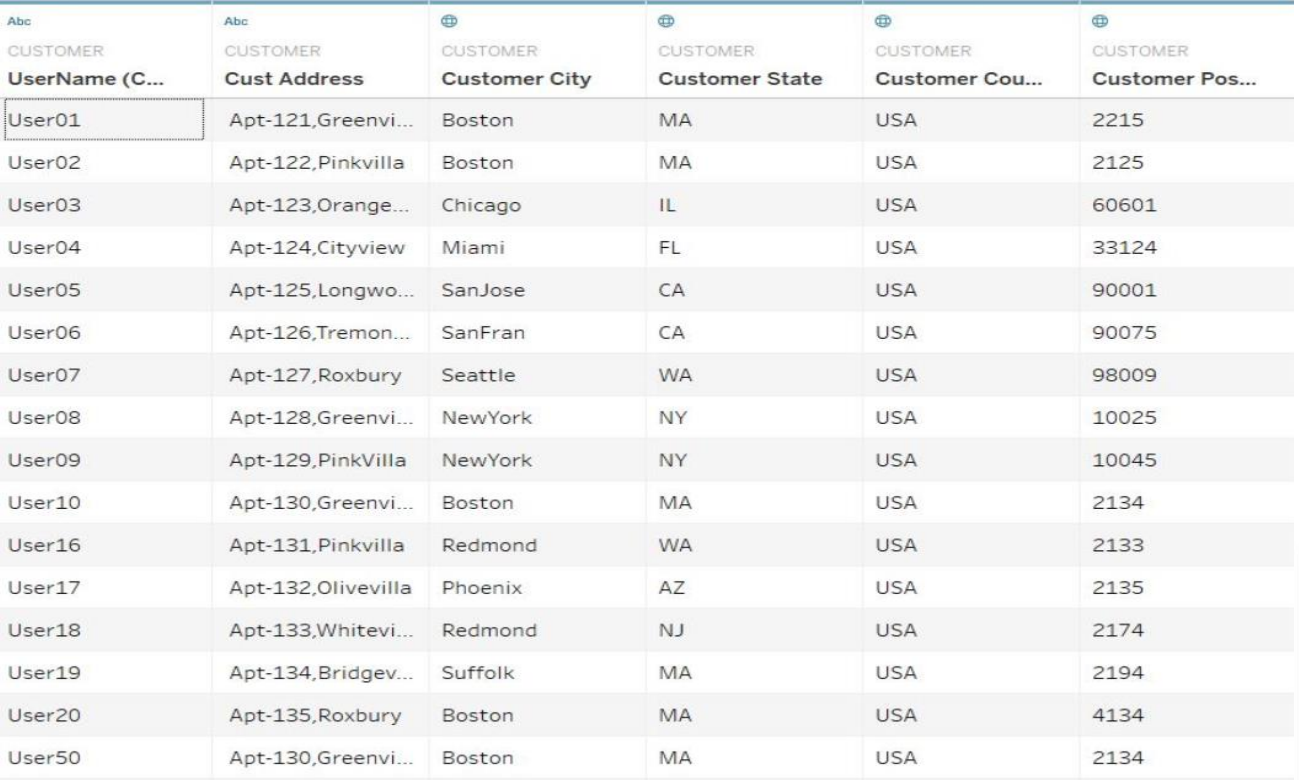


- ❑ Business can use this to Analyse more used mode of Payment and act accordingly

To depict which payment type is used better



INVOICE1 InvoiceID (INV...	INVOICE1 OrderID (INVO...	INVOICE1 Date (INVOIC...	INVOICE1 TotalAmount (...	INVOICE1 PaymentStatu...	INVOICE1 PaymentType ...
11111111	1	05-01-2019 00:0...	12.4400	paid	CD
11111112	2	05-04-2019 00:0...	23.4400	paid	CD
11111113	3	05-01-2020 00:0...	65.9800	paid	CD
11111114	4	05-06-2019 00:0...	11.4600	paid	CD
11111115	5	01-02-2020 00:0...	87.8400	notpaid	CD
11111116	6	07-03-2020 00:0...	82.2300	paid	CD
11111117	7	02-04-2020 00:0...	12.4400	notpaid	CD
11111118	8	08-05-2020 00:0...	15.4400	paid	CD
11111119	9	04-06-2020 00:0...	12.4400	paid	CD
11111120	10	29-10-2020 00:0...	10.6800	notpaid	CD
11111121	11	05-04-2019 00:0...	9.2400	paid	CP
11111122	12	17-07-2020 00:0...	13.3400	paid	CP
11111123	13	25-08-2020 00:0...	56.5600	notpaid	CP
11111124	14	29-11-2020 01:0...	19.2500	paid	CP
11111125	15	29-11-2020 02:0...	68.5000	paid	CP
11111126	16	22-10-2020 00:0...	78.0000	paid	CP

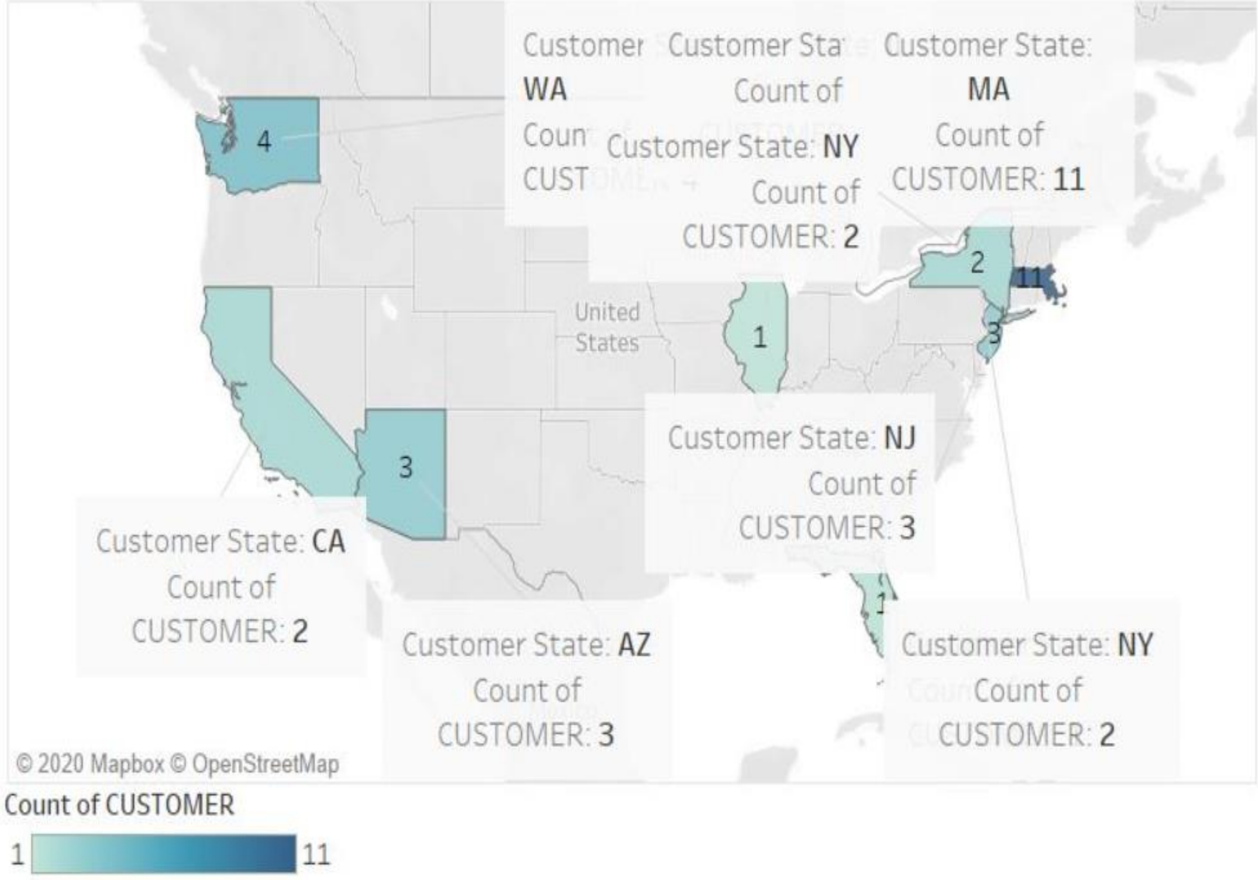


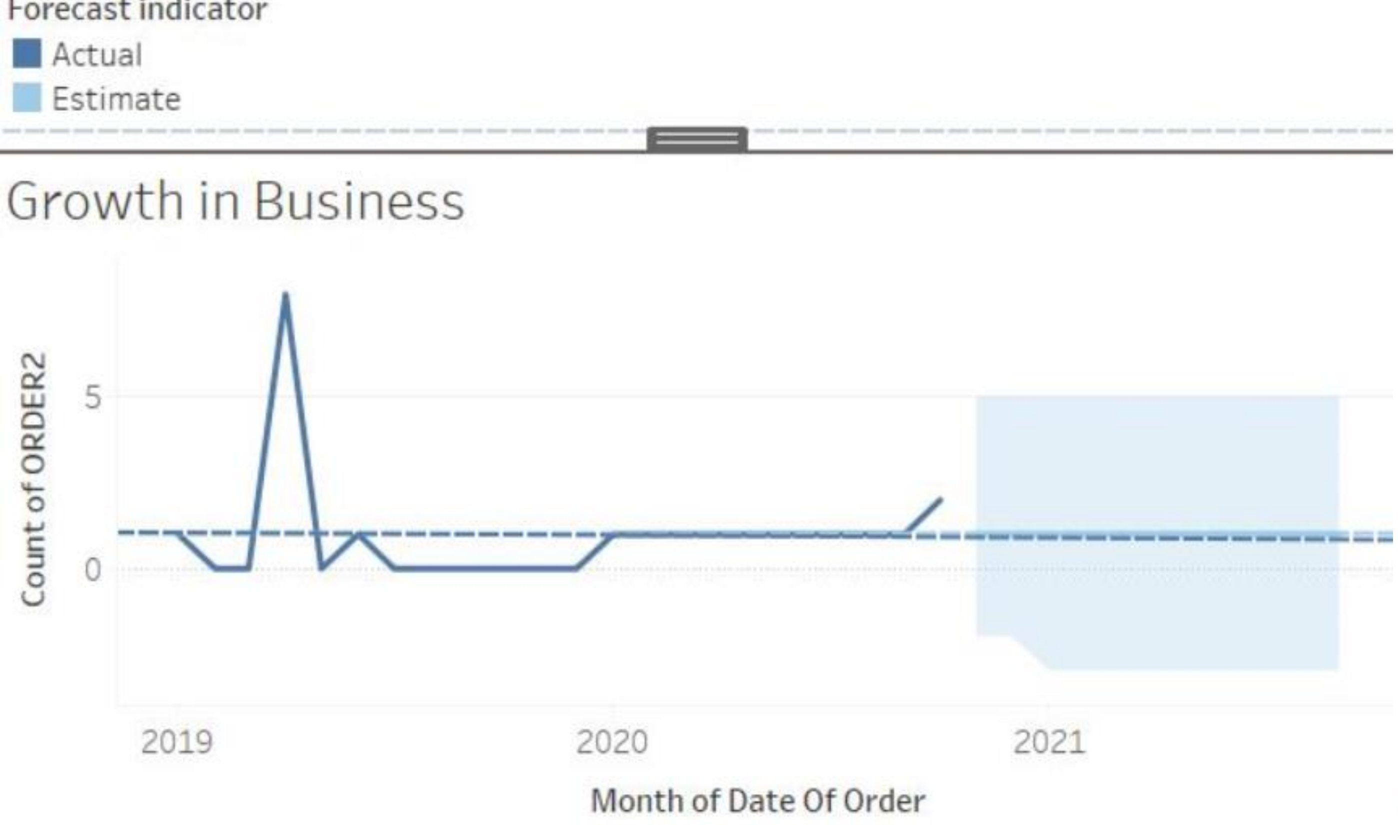
CUSTOMER	CUSTOMER	CUSTOMER	CUSTOMER	CUSTOMER	CUSTOMER
UserName (C...	**Cust Address**	**Customer City**	**Customer State**	**Customer Cou...**	**Customer Pos...**
User01	Apt-121,Greenvi...	Boston	MA	USA	2215
User02	Apt-122,Pinkvilla	Boston	MA	USA	2125
User03	Apt-123,Orange...	Chicago	IL	USA	60601
User04	Apt-124,Cityview	Miami	FL	USA	33124
User05	Apt-125,Longwo...	SanJose	CA	USA	90001
User06	Apt-126,Tremon...	SanFran	CA	USA	90075
User07	Apt-127,Roxbury	Seattle	WA	USA	98009
User08	Apt-128,Greenvi...	NewYork	NY	USA	10025
User09	Apt-129,PinkVilla	NewYork	NY	USA	10045
User10	Apt-130,Greenvi...	Boston	MA	USA	2134
User16	Apt-131,Pinkvilla	Redmond	WA	USA	2133
User17	Apt-132,Olivevilla	Phoenix	AZ	USA	2135
User18	Apt-133,Whitevi...	Redmond	NJ	USA	2174
User19	Apt-134,Bridgev...	Suffolk	MA	USA	2194
User20	Apt-135,Roxbury	Boston	MA	USA	4134
User50	Apt-130,Greenvi...	Boston	MA	USA	2134

Could be used to analyse a location of more profit.
Can be used to expand business

- ❑ Courier service everywhere with at most customer satisfaction

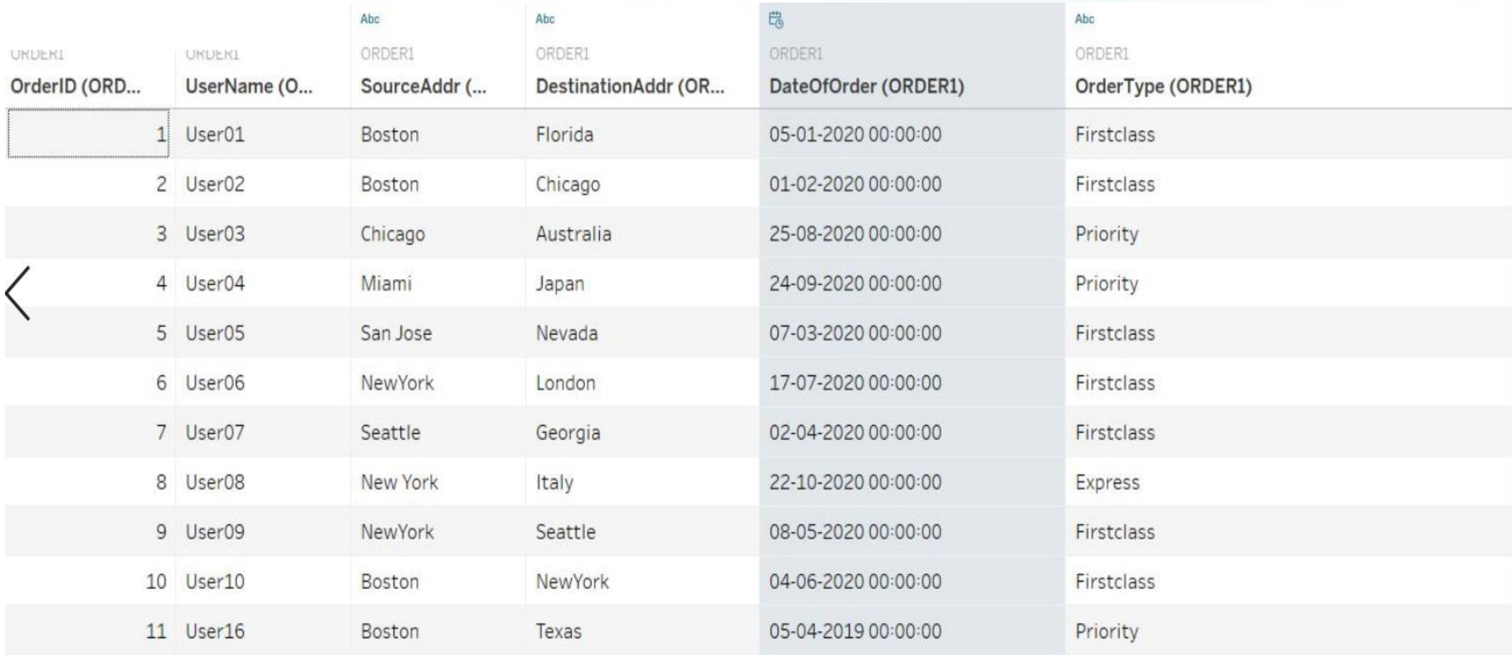
We are all over the Globe





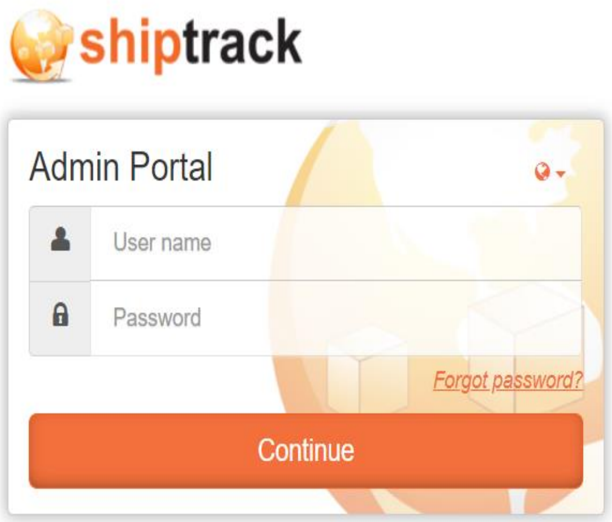
- ❑ Growth Strategy is important for companies to expand their business

Growth can be achieved by practices like adding new Locations, expanding the product lines



ORDER1	ORDER1	ORDER1	ORDER1	ORDER1	ORDER1
OrderID (ORD...	UserName (O...	SourceAddr (...	DestinationAddr (OR...	DateOfOrder (ORDER1)	OrderType (ORDER1)
1	User01	Boston	Florida	05-01-2020 00:00:00	Firstclass
2	User02	Boston	Chicago	01-02-2020 00:00:00	Firstclass
3	User03	Chicago	Australia	25-08-2020 00:00:00	Priority
4	User04	Miami	Japan	24-09-2020 00:00:00	Priority
5	User05	San Jose	Nevada	07-03-2020 00:00:00	Firstclass
6	User06	NewYork	London	17-07-2020 00:00:00	Firstclass
7	User07	Seattle	Georgia	02-04-2020 00:00:00	Firstclass
8	User08	New York	Italy	22-10-2020 00:00:00	Express
9	User09	NewYork	Seattle	08-05-2020 00:00:00	Firstclass
10	User10	Boston	NewYork	04-06-2020 00:00:00	Firstclass
11	User16	Boston	Texas	05-04-2019 00:00:00	Priority

UI MOCKS



shiptrack

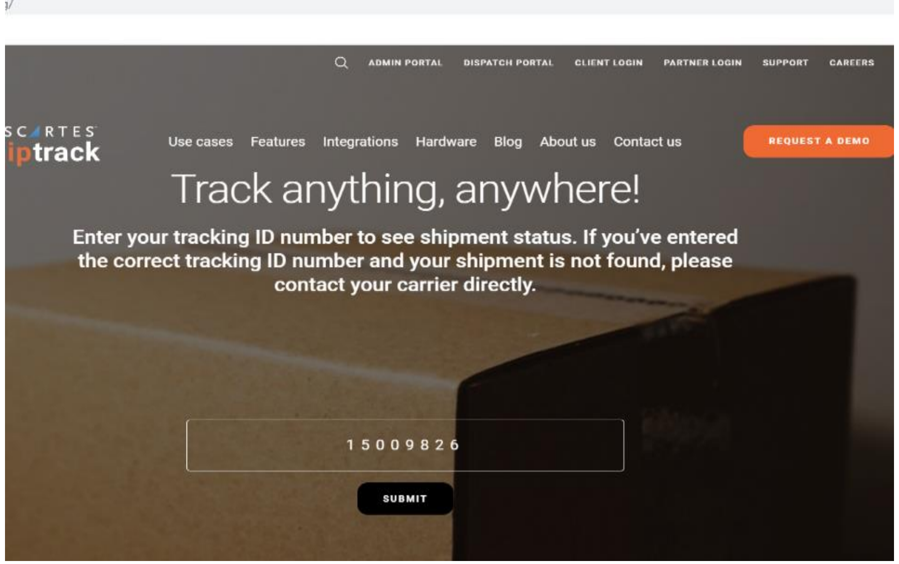
Admin Portal

User name

Password

Forgot password?

Continue



ADMIN PORTAL | DISPATCH PORTAL | CLIENT LOGIN | PARTNER LOGIN | SUPPORT | CAREERS

SCARTES iptrack

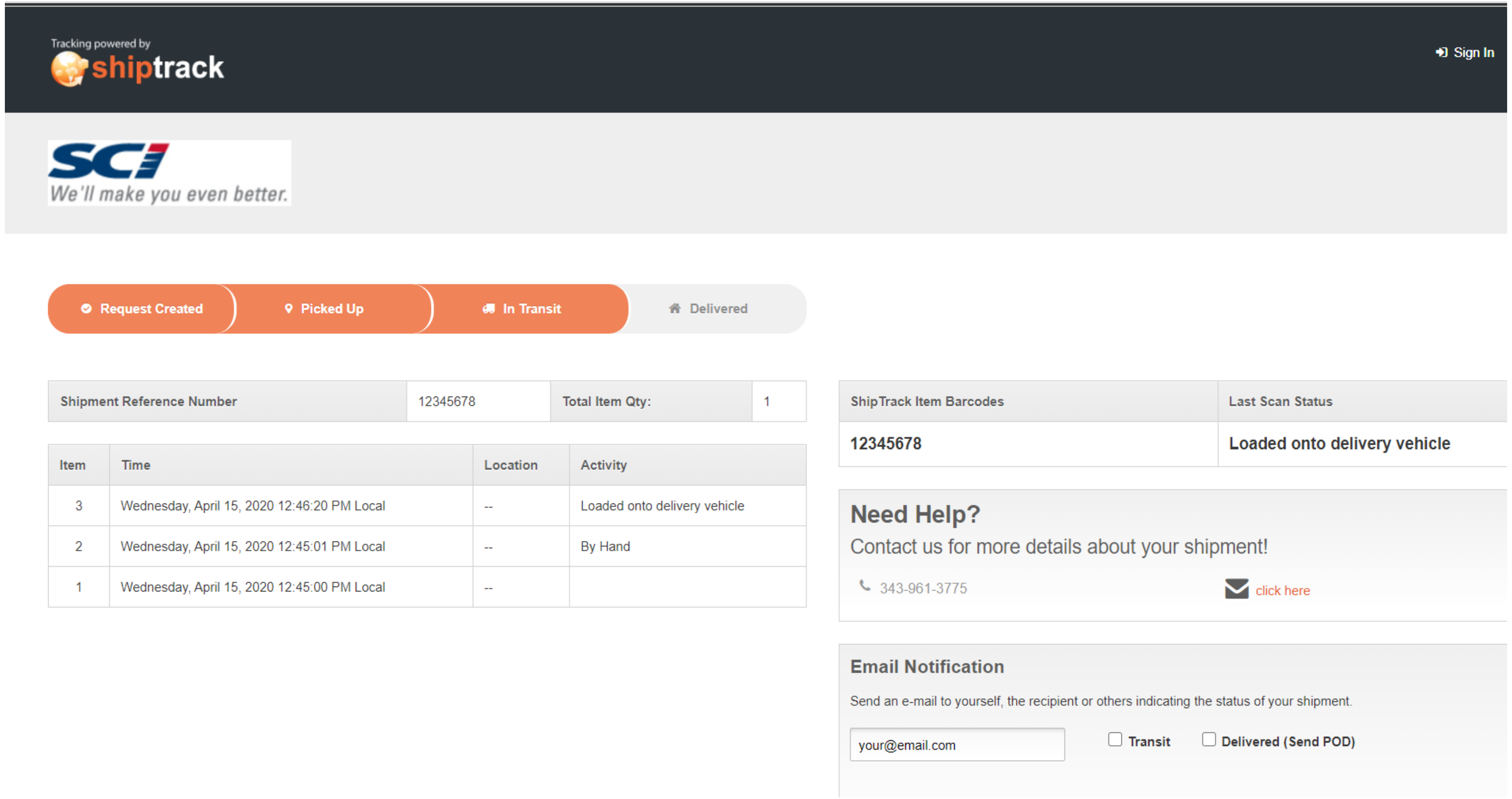
Use cases | Features | Integrations | Hardware | Blog | About us | Contact us | REQUEST A DEMO

Track anything, anywhere!

Enter your tracking ID number to see shipment status. If you've entered the correct tracking ID number and your shipment is not found, please contact your carrier directly.

15009826

SUBMIT



Tracking powered by shiptrack

Sign In

SCI We'll make you even better.

Request Created | Picked Up | In Transit | Delivered

Shipment Reference Number	12345678	Total Item Qty:	1

Item	Time	Location	Activity
3	Wednesday, April 15, 2020 12:46:20 PM Local	--	Loaded onto delivery vehicle
2	Wednesday, April 15, 2020 12:45:01 PM Local	--	By Hand
1	Wednesday, April 15, 2020 12:45:00 PM Local	--	

ShipTrack Item Barcodes	Last Scan Status
12345678	Loaded onto delivery vehicle

Need Help?

Contact us for more details about your shipment!

☎ 343-961-3775

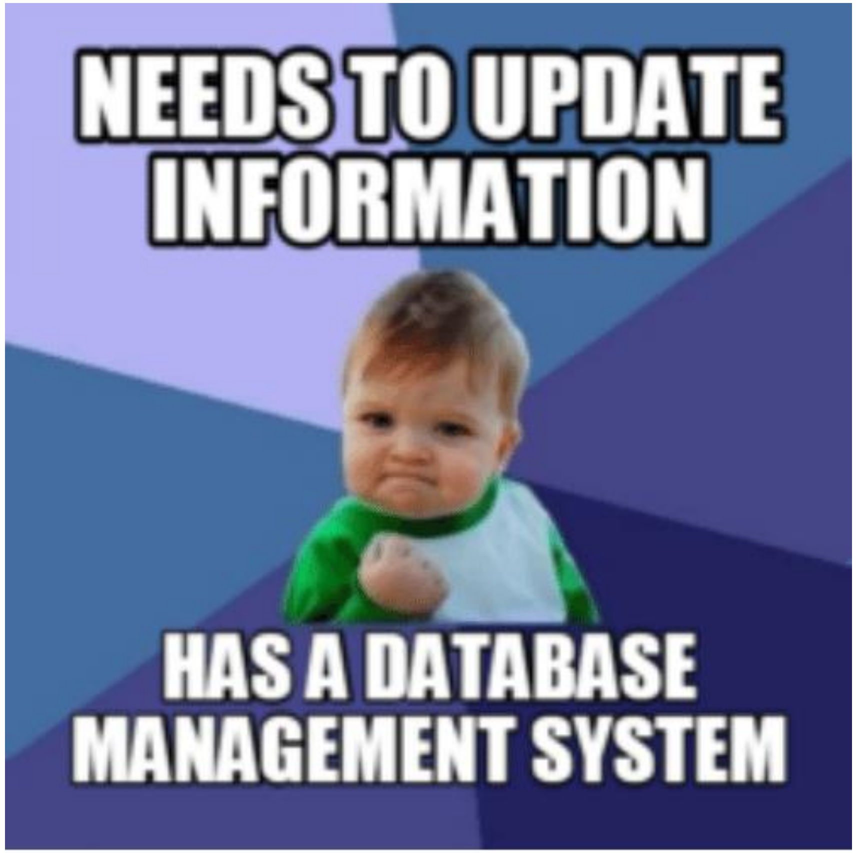
click here

Email Notification

Send an e-mail to yourself, the recipient or others indicating the status of your shipment.

your@email.com

□ Transit □ Delivered (Send POD)


NEEDS TO UPDATE INFORMATION
HAS A DATABASE MANAGEMENT SYSTEM

Thank You