SamXpress Grocery Ordering System

Database Design and Development

Table of Contents

Contents

Introduction

Sam, store ordering system in Myanmar, provides store orders across a variety of products to customers across Myanmar. It is empowering thousands of shopkeepers and sellers to connect with millions of customers and provides immediate and easy access to 500,000 products in more than 100 categories.

Task 1

Description of Business

Scenario

SamXpress is one of the **grocery ordering** systems in Myanmar. It is one of the entrepreneur businesses in Myanmar. They are increased in number of customers in 2021.

They want the database that is more systematic than other systems. The database must help their order, payment, delivery etc. The system must be trusted by company and customer for ordering and purchasing. The system is not only for retail but also for wholesale.

This grocery ordering database was created based on the traditional grocery ordering requirements. The system can encode customer information when purchasing / ordering. The administrator can have access to both customer information and their transactions. They can process the data needed to manage information and record customer requests.

The data will get many problem between department if the database is not exists. The functions included in the ER system diagram include security and monitoring of seller information and the status of customer information. These functions were also listed and recorded in feedback that served as a history of transactions made in the system.

Products are sold by shops. Shops are both retail and wholesale shop. Customers can buy the products via orders.

The customers can make their orders. The orders will be delivered by delivery staff. Orders can have one or more items. An order details contains one or more products. Orders will be connected with payment.

Customers gives rating on a products and services. So, products will get many rating from customers. Products belong to the categories. Products will be classified by Categories. Also, the shops will be classified by the shop type.

Form Design

Table 1. Products are sold by Shops

ShopType	Shop Name	Phone number	Address	Category Name	Product Name
Wholesale	MyanTea	+953847597	Main Rd, Kalaw Tsp, Shan State	Tea	Yee Mon pickled tea leaves
Wholesale	MyanTea	+953847597	Main Rd, Hlaing Tsp, Yangon	Tea	Yee Mon dry tea leaves
Retail	Linn	+956847532	34th St, Latha Tsp, Yangon	Milk	Silver Pearl milk
Retail	Alex	+954752846	Nan St, Insein Tsp	Oil	Organic Soybean Oil
Wholesale	Coca Cola	+952734537	Nandar St, ShwePyiThar Tsp, Yangon	Drink	Coca Cola zero
Wholesale	Loi Hein	+954758438	WarTan Rd, Alone Tsp, Yangon	Drink	Shark energy drink
Wholesale	Wilmar	+956378463	Main Rd, Hmawbi Tsp, Yangon	Oil	Meizin Oil

Retail	LoThaYa	+954758374	NaungDone St, Sancahung Tsp,Yangon	Snack	KitKat Chocolate
Retail	Platinum	+952847563	Dangon Tsp, Yangon	Tea	Milk Tea powder
Wholesale	ShweBo	+954678263	ShweBo District, Sagaing	Rice	Shwe Bo Paw San rice
Wholesale	Ready	+952683464	Pinlone Rd, North Dangon Tsp, Yangon	Ready-made can	Ready Chittee chicken curry
Retail	Shwe Store	+955638472	PyiHtaungSu Rd, Bago District,Bago	Tissue	Orchid Tissue
Wholesale	Swel	+958736453	HaKha District, Chin State	Snack	Swel sunflower seeds

Table 2. Customers Sale Order Voucher

"SamXpress" Grocery Ordering Company Limited

Customer Name: Kyaw Moe Tun Order Date: 22/3/2021

Phone Number: +9594857495

Customer Address: Tamwe, Yangon

Email Address: kmtun997@gmail.com

Gender: Male

PostCode: 20023

No	Ordered Grocery Name	Amount	Quantity	Amount
1	Shwe Linn Yone Pork Sausage	3350	15	50,250
2	Grey Cheese Crackers	1900	20	38,000
3	Silver Pearl milk	700	100	70,000
4	Mayzin Soybean Oil 1.8L	6900	10	69,000
5	Tan Choon Hwa chili sauce	800	25	20,000
6	Ready Fish cakes in curry	1200	10	12,000
7	Maggi Oyster sauce 210G	1300	15	19,500
8	SUNSILK Shampoo Black Shine	3600	10	36,000
			Total	314,750
			Tax - 5%	-
			Charge	314,750
			Promotion	-

Table 3. Payment Receipt

Payment Date: 22/3/2021 Payment ID: Pa 0021 Customer ID: Cu 0030 Order ID: Or 0034

Amount: 350,000

Promotion: 2,000

Payment Method: Bank Transfer

Signature: may

"I (the customer) accept that all amount must be paid before delivering the products."

Table 4. Order and Delivery List

Order ID	Customer ID	Payment ID	Order Date	Delivery ID	Shipping Methods	Delivery Staff Name	Deliver Date
Or0095	Cu0101	Pa0095	20.2.2021	De0095	DHL	Naing Ye Thu	21.2.2021
Or0096	Cu0092	Pa0096	20.2.2021	De0096	Shop Express	Ye Min Aung	23.2.2021
Or0097	Cu0051	Pa0097	21.2.2021	De0097	Brought from warehouse		22.2.2021

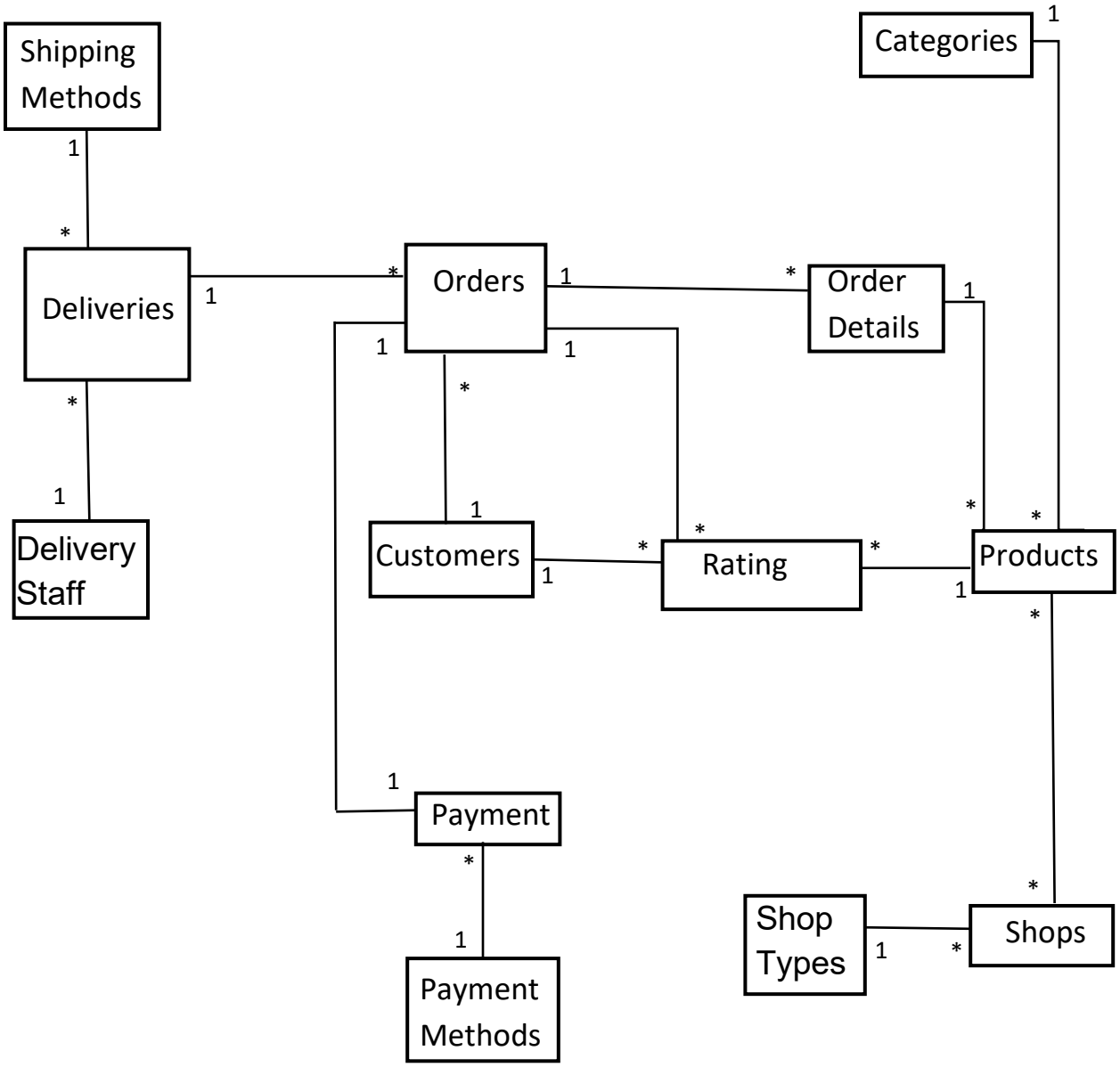
Or0098	Cu0079	Pa0098	21.2.2021	De0098	DHL	Nanda Min	25.2.2021
Or0099	Cu0014	Pa0099	21.2.2021	De0099	Royal Express	Kalayar Pyae	24.2.2021
Or0100	Cu0044	Pa0100	22.2.2021	De0100	Royal Express	Linn Myat	27.2.2021
Or0101	Cu0050	Pa0101	23.2.2021	De0101	Shop Express	Lu Maw	27.2.2021
Or0102	Cu0063	Pa0102	23.2.2021	De0102	FedEx	Min Din	26.2.2021
Or0103	Cu0105	Pa0103	24.2.2021	De0103	Brought from warehouse		25.2.2021
Or0104	Cu0092	Pa0104	25.2.2021	De0104	Shop Express	Nandar Khin	1.3.2021
Or0105	Cu0024	Pa0104	25.2.2021	De0105	FedEx	Win Naing	27.2.2021

Task 2

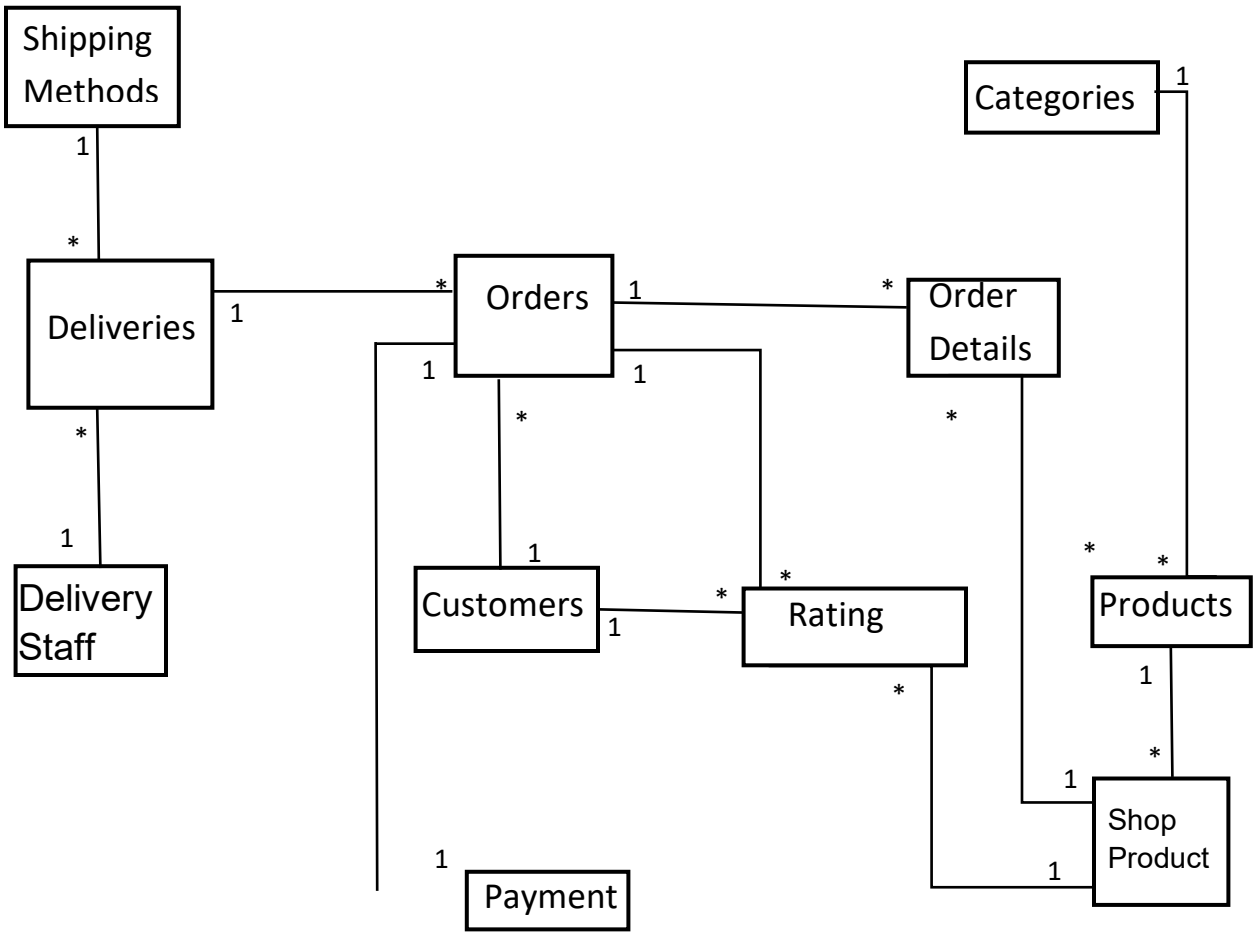
ER and Data dictionary

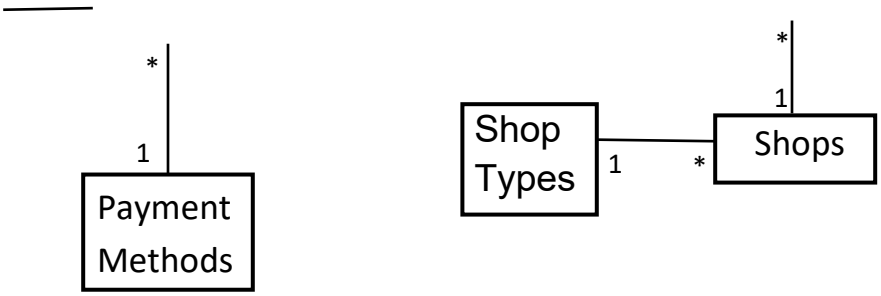
Entity Relationship Diagram

Entity Relationship, Diagram, ER Diagram or ER model, is a type of structural diagram for use in database design. The Diagram shows entities and their relationships. The ERD contains the connectors and symbols. ERD is one of the main parts of database system.



ERD without dummy table





ERD with dummy table

According to the scenario, the ERD is built to get appropriate system. The system is to connect between shopkeepers and customers. Customer will buy the products from shopkeepers. So, Entities about Customers, Order, products and shop are created. Shop will have shop types and it have one to many relationships. Products and shop connect as many to many relationships. Then there is a dummy table between them. Products are classified as categories and there is a rating for products and shops. They are connected by one to many relationships. Customer can pay with many payment methods and also have payment entities to pay orders. Orders and payment will one to one to get more clear entities. Orders will deliver by delivery staff.

Data Dictionary

A data dictionary is a collection of names, definitions, and attributes of data elements that are used or recorded in a database, information system, or part of a research project. Data Dictionary is used to help avoid inconsistencies in project data and define the conventions to be used in the

project, Ensure consistency in the collection and use of data among several members of the research team, simplify data analysis and Data Standards Enforcement. (Anon., 2021)

Look at the ERD and plan for data dictionary. Attributes are important in data dictionary. Attributes are created according to the system. Then connected by primary key and foreign key.

Categories

Entity : Categories

Primary Key: CategoryID

Foreign Key: -

Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
CategoryID	varchar	10	PK	Start with letter Ca-0000 and followed by sequential number	Unique ID for Category
CategoryName	varchar	30			Name for Category.

Products

Entity : Products

Primary Key: ProductID Foreign Key: CategoryID					
Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
ProductID	varchar	10	PK	Start with letter Pr-0000 and followed by sequential number	Unique ID for Product
ProductName	varchar	30			Name of Product.
CategoryID	varchar	10	FK	Start with letter Ca-0000 and followed by sequential number	Unique ID of Category

Shop_Types

Entity : Shop_Types

Primary Key: ShopTypeID Foreign Key: -					
Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
ShopType ID	varchar	10	PK	Start with letter St-0000 and followed by sequential number	Unique ID of Shop Type
ShopType	varchar	30			Name for ShopType

Shops

Entity : Shops Primary Key: ShopID Foreign Key: ShopTypeID					
Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
ShopID	varchar	10	PK	Start with letter Sh-0000 and followed by sequential number	Unique ID of Shops
ShopName	varchar	50			Name of Seller.
ShopTypeID	varchar	10	FK	Start with letter St-0000 and followed by sequential number	Unique ID of Shop Type
PhoneNumber	varchar	10			Contact Info Shops

ShopAddress	varchar	100			Address of Shops

Orders

Entity : Orders Primary Key: OrderID Foreign Key: CustomerID, DeliveryID					
Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
OrderID	varchar	10	PK	Start with letter Or-0000 and followed by sequential number	Unique ID of Order
CustomerID	varchar	10	FK	Start with letter Cu-0000 and followed by sequential number	Unique ID of Customer
DeliveryID	varchar	10	FK	Start with letter De-0000 and followed by sequential number	Unique ID of Deliveries

Propagation Constraint

Propagation Constraint for DeliveryID in Orders table

Foreign key (DeliveryID) references O (OrderID)

On Delete Cascade

On Update Cascade

Customers

Entity : Customers Primary Key: CustomerID Foreign Key:					
Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
CustomerID	varchar	10	PK	Start with letter Cu-0000 and followed by sequential number	Unique ID for Customers
CustomerName	varchar	30			Name of Customer
Email	varchar	20			Email of Customer
Gender	varchar	10			Gender of Customer
PhoneNumber	varchar	20			Phone number of Customers
CustomerAddress	varchar	200			Address of Customers

Entity : Order_Details

Primary Key: OrderDetail_ID

Foreign Key: ShopProductID, OrderID

Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
OrderDetail_ID	varchar	10	PK	Start with letter Od-0000 and followed by sequential number	Unique ID for Order Details
ShopProductID	varchar	10	FK	Start with letter Sp-0000 and followed by sequential number	Unique ID for Shop Product
OrderID	varchar	10	FK	Start with letter Or-0000 and followed by sequential number	Unique ID for Orders
Amount	integer				Amount of Products in Orders

Quantity	integer				Quantity of Products in Orders
Order_Date	date				Date for Order

Order_Details

Propagation Constraint

Propagation Constraint for ShopProductID in Order Details table

Foreign key (ShopProductID) references Shop_Product (ShopProductID)

On Delete No Action

Propagation Constraint for OrderID in Order Details table

Foreign key (OrderID) references Orders (OrderID)

On Delete Cascade

On Update Cascade

Shop_Product

Entity : Shop_Product

Primary Key: ShopProductID

Foreign Key: ProductID, ShopID

Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
ShopProductID	varchar	10	PK	Start with letter Od-0000 and followed by sequential number	Unique ID for Shop Product
ProductID	varchar	10	FK	Start with letter Od-0000 and followed by sequential number	Unique ID for Product
ShopID	varchar	10	FK	Start with letter Od-0000 and followed by sequential number	Unique ID for Seller

Payment

Entity : Payment

Primary Key: OrderID

Foreign Key: PaymentMethodID

Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
OrderID	varchar	10	PK	Start with letter Od-0000 and followed by sequential number	Unique ID of Order
PaymentMethodID	varchar	10	FK	Start with letter Pm-0000 and followed by sequential number	Unique ID of Payment Method
TotalAmount	integer				Amount for Payment
PaymentDate	date				Date of Payment
Promotion Price	integer				Promotion Price of Payment

Payment_Methods

Entity : Payment_Methods Primary Key: PaymentMethodsID Foreign Key:					
Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
PaymentMethodID	varchar	10	PK	Start with letter Pm-0000 and followed by sequential number	Unique ID for PaymentMethod
PaymentMethodName	varchar	30			Name of PaymentMethod

Deliveries

Entity : Deliveries Primary Key: DeliveryID Foreign Key: OrderID, ShippingMethods_ID, DeliveryStaff_ID,

Name	Type	Size	Integrity Constraint	Domain Constraint	Description
DeliveryID	varchar	10	PK	Start with letter De-0000 and followed by sequential number	Unique ID for Delivery
ShippingMethod_ID	varchar	10	FK	Start with letter Sm-0000 and followed by sequential number	Unique ID for Shipping Methods
DeliveryStaff_ID	varchar	10	FK	Start with letter Ds-0000 and followed by sequential number	Unique ID for Delivery Staff
DeliveryDate	date				Date of Delivery

Delivery_Staff

Entity : Delivery_Staff Primary Key: DeliveryStaffID Foreign Key:

Name	Type	Size	Integrity Constraint	Domain Constraint	Description
DeliveryStaffID	varchar	10	PK	Start with letter Ds-0000 and followed by sequential number	Unique ID of Delivery Staffs
DeliveryStaffName	varchar	30			Name of Delivery Staff

Shipping_Methods

Entity : Shipping_Methods

Primary Key: ShippingMethodID

Foreign Key:

Name	Type	Size	Integrity Constraint	Domain Constraint	Description
ShippingMethodID	varchar	10	PK	Start with letter Sm-0000 and followed by sequential number	Unique ID for ShippingMethods
ShippingMethodName	varchar	30			Name of ShippingMethod

Rating

Entity : Rating

Primary Key: RatingID

Foreign Key: ShopProductID, CustomerID, OrderID

Name	**Type**	**Size**	**Integrity Constraint**	**Domain Constraint**	**Description**
RatingID	varchar	10	PK	Start with letter Ra-0000 and followed by sequential number	Unique ID for Rating
ShopProductID	varchar	10	FK	Start with letter Sp-0000 and followed by sequential number	Unique ID for Shop Product
CustomerID	varchar	10	FK	Start with letter Su-0000 and followed by sequential number	Unique ID of Customer
OrderID	varchar	10	FK	Start with letter Or-0000 and followed by sequential number	Unique ID of OrderID
Score	integer				Score of products from customers
Remark	varchar	100			Remark for product
DateRecorded	date				Date recorded rating

Task 3

Normalization

Normalization, Purpose of Normalization and Steps of Normalization

Normalization is the process of structuring and manipulating relationships between data to minimize redundancy in a relational table and avoid unnecessary anomaly properties from the database, such as insert, update, and delete. This helps to split large database tables into smaller ones and establish relationships between them. It can remove redundant data and make it easier to add, change, or delete table fields.

Normalization defines the rules for a relational table as to whether or not it conforms to normal form. Normal form is a process that evaluates each relationship against specific criteria and removes multivalued, unified, functional, and trivial dependencies from the relationship. If any

data is updated, deleted, or inserted, it does not cause problems for the database tables and helps to improve the integrity and efficiency of the relational table. (Anon., 2021)

Firstly, list the UNF according to the entities. Then make the level for UNF. Normalization level for UNF can be thought as 1 for single and 2 for many. There are three steps of normalization.

In 1st normal form, looking for the repeating groups. Then remove the repeating groups of information. Two tables are created. They are connected by foreign keys

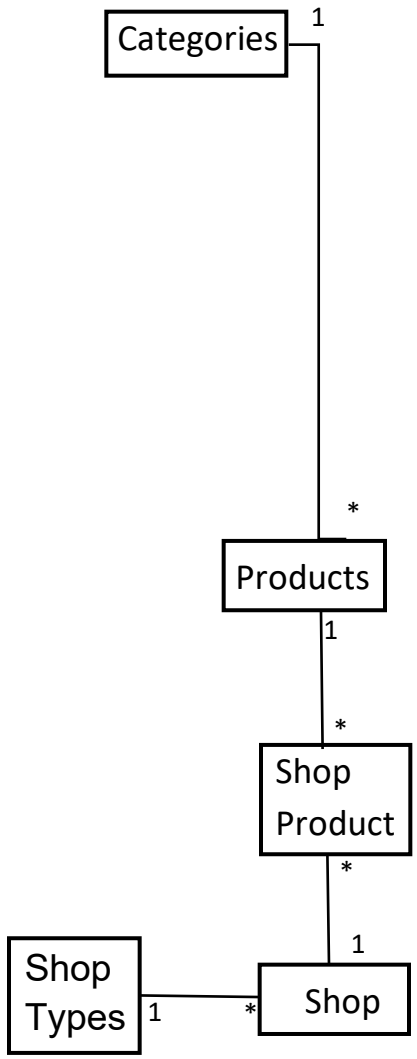
In 2nd normal form, depending on the attributes 1NF, partial key dependencies are removed. Sometimes, dummy tables are created. Tables have both primary keys and foreign keys.

In 3rd normal form, after defining primary keys and foreign keys, non-key dependencies which are not dependent on the other attributes are removed. Non-dependent attributes are separated as the new table. The foreign keys become composite primary keys. After removing, the complete entities and attributes will be gotten.

Table 1. Products are sold by Shops

UNF	Level	1NF	2NF	3NF	Entity Name
CategoryID CategoryName	1 1	ProductID(PK) ProductName	ProductID(PK) ProductName	CategoryID(PK) CategoryName	Categories

ProductID	1	CategoryID	CategoryID		
ProductName	1	CategoryName	CategoryName	ProductID(PK)	Products
ShopID	1			CategoryID(FK)	
ShopName	1	ProductID(FK)	ProductID(PK)(FK)	ProductName	
ContactNumber	2	ShopID	ShopID(PK)(FK)		
Address	2	ShopName		ShopProductID(PK)	Shop_Product
ShopTypeID	1	ContactNumber	ShopID(PK)	ProductID(FK)	
ShopType	2	Address	ShopName	ShopID(FK)	
		ShopID	ContactNumber		Shops
		ShopType	Address	ShopID(PK)	
			ShopID	ShopTypeID(PK)	
			ShopType	ShopName	
				ContactNumber	
				Address	
					Shop_Type
				ShopTypeID(PK)	
				ShopType	

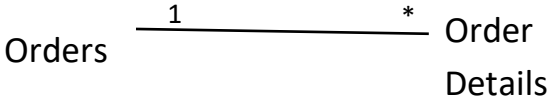


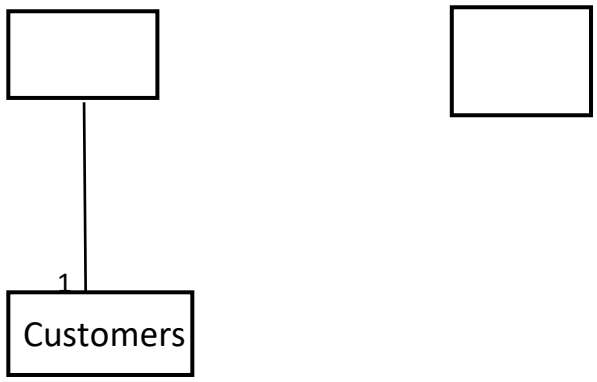
List attributes from Categories, Products, Shops and Shop Type in UNF. Remove the repeating groups and get 1NF. One primary key and foreign key is founded. Then remove the partial key dependencies and get 2NF. Then remove any non-key dependencies to get 3NF. Entities are formed.

Table 2. Customers Sale Order Voucher

UNF	Level	1NF	2NF	3NF	Entity Name

CustomerID	1	CustomerID(PK)	OrderID(PK)	OrderID(PK)	Orders
CustomerName	1	CustomerName	CustomerID(FK)	CustomerID(FK)	
Email	1	Email			
Gender	2	Gender	OrderDetail_ID(PK)	OrderDetail_ID(PK)	
PhoneNumber	1	PhoneNumber	ShopProductID(FK)	ShopProductID(FK)	Order_Details
Address	2	Address	Amount	OrderID(FK)	
OrderDetail_ID	1		Quantity	Amount	
ShopProductID	1	CustomerID(FK)	TotalAmount	Quantity	
Amount	2	OrderDetail_ID		TotalAmount	
Quantity	2	ShopProductID	CustomerID(PK)		
TotalAmount	2	Amount	CustomerName	CustomerID(PK)	Customers
OrderID	1	Quantity	Email	CustomerName	
OrderDate	1	TotalAmount	Gender	Email	
	2	OrderID	PhoneNumber	Gender	
		PaymentID	Address	PhoneNumber	
		OrderDate		Address	





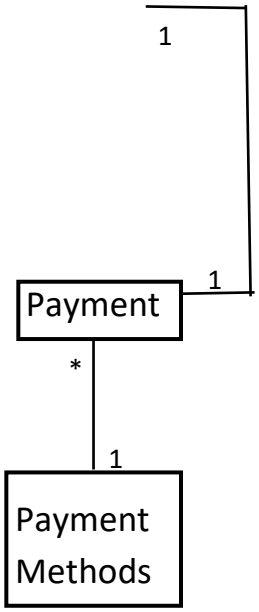
List attributes from Orders, Order Details and Customers in UNF. Remove the repeating groups and get 1NF. One primary key and foreign key is founded. Then remove the partial key dependencies and get 2NF. Then remove any non-key dependencies to get 3NF. Entities are formed.

Table 3. Payment Receipt

UNF	Level	1NF	2NF	3NF	Entity Name

PaymentID OrderID CustomerID Amount PaymentDate Promotion PaymentMethodID PaymentName	**1** **1** **1** **2** **2** **2** **1** **1**	PaymentMethodID(FK) OrderID CustomerID Amount PaymentDate Promotion PaymentMethodID(PK) PaymentName	OrderID(PK) CustomerID(FK) OrderID (PK) PaymentMethodID(FK) Amount PaymentDate Promotion PaymentMethodID(PK) PaymentName	OrderID(PK) CustomerID(FK) OrderID(PK) PaymentMethodID(FK) Amount PaymentDate Promotion PaymentMethodID(PK) PaymentName	Orders Payment Payment Methods

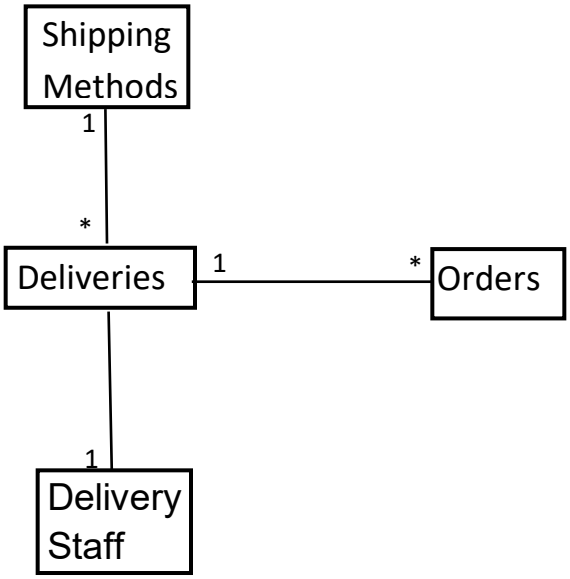
Orders



List attributes from Orders, Payment and Payment Methods in UNF. Remove the repeating groups and get 1NF. One primary key and foreign key is founded. Then remove the partial key dependencies and get 2NF. Then remove any non-key dependencies to get 3NF. Entities are formed.

Table 4. Order and Delivery List

UNF	Level	1NF	2NF	3NF	Entity Name
OrderID CustomerID PaymentID DeliveryID DeliveredDate DeliveryStaffID DeliveryStaffName ShippingMethodID ShippingMethodName	1 1 1 1 2 1 1 1 1 2	OrderID(PK) CustomerID(FK) PaymentID(FK) OrderID(FK) DeliveryID DeliveredDate DeliveryStaffID DeliveryStaffName ShippingMethodID ShippingMethodName	DeliveryID(PK) OrderID(FK) ShippingMethodID(FK) DeliveredDate DeliveryStaffID DeliveryStaffName ShippingMethodID(PK) ShippingMethodName OrderID(PK) CustomerID(FK) PaymentID(FK)	DeliveryID(PK) OrderID(FK) ShippingMethodID(FK) DeliveryStaffID(FK) DeliveredDate DeliveryStaffID(PK) DeliveryStaffName ShippingMethodID(PK) ShippingMethodName OrderID(PK) CustomerID(FK) PaymentID(FK)	Deliveries Delivery Staff Shipping Methods Orders



List attributes from Orders, Deliveries, Delivery Staff, Shipping Methods in UNF. Remove the repeating groups and get 1NF. One primary key and foreign key is founded. Then remove the partial key dependencies and get 2NF. Then remove any non-key dependencies to get 3NF. Entities are formed.

Anomalies

An anomaly means a database is not normalized properly. (Anon., 2021)

Insert Anomalies

Update Anomalies

Delete Anomalies

How Normalization solves the problem of update anomalies

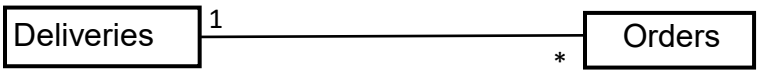
Customer ID	Order ID	Order	Order Date	Quantity	Delivery ID	Delivery Date	Description
Cu0002	Or0002	Shwe Bo Paw San rice	18/02/2021	20	De0002	22/2/2020	The ordered product was delayed to Feb 22 due to the Delivery staffs are busy

Update Anomalies

If we make one to one relation between orders and deliveries, the delivery will delay because of delivery staff and budget for delivery. If the delivery will delay, the company will get bad review. Then the company will get products and shops problem and payments problem. So the database about orders and deliveries must be update.

Solution

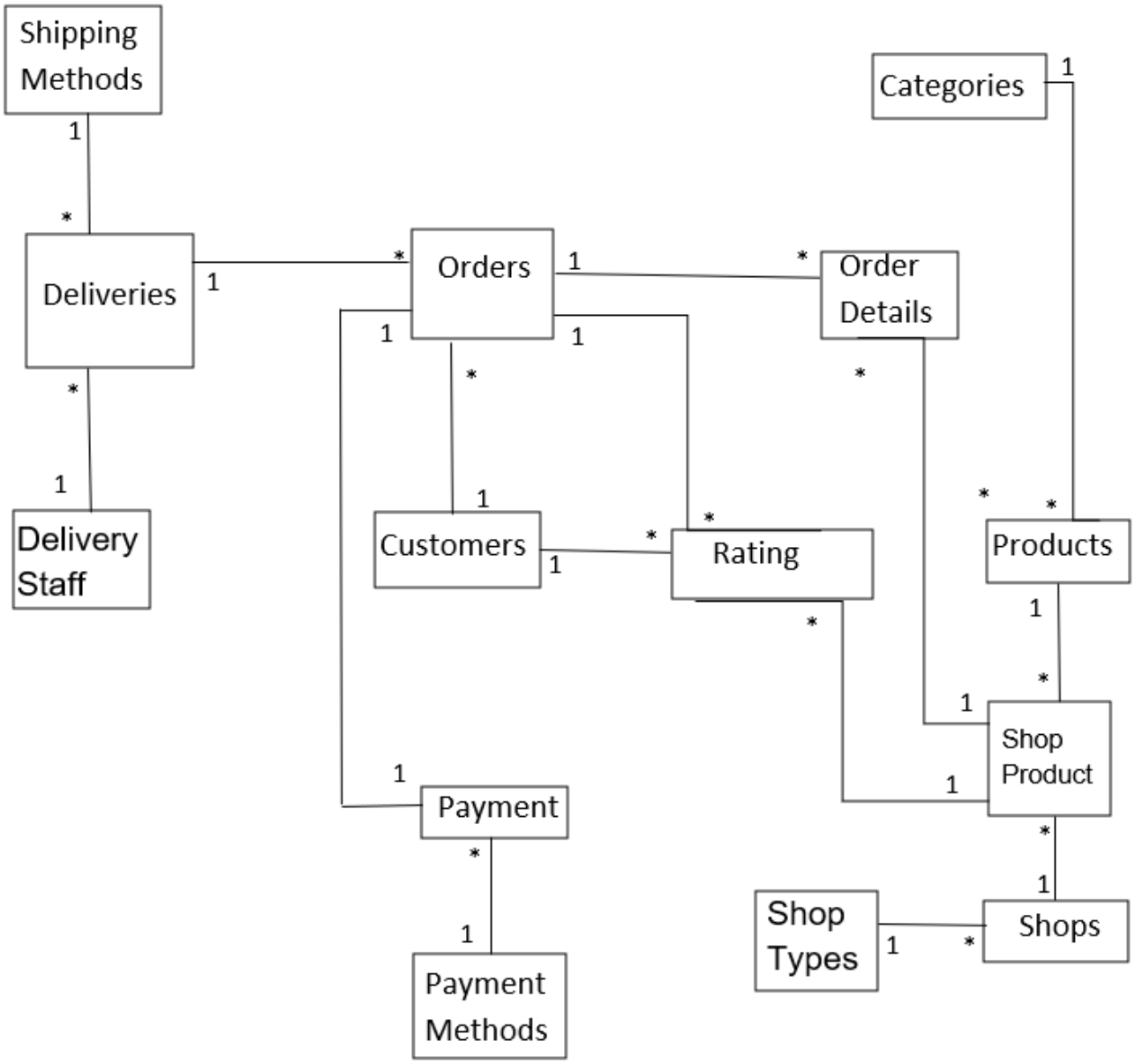
To prevent the problems, the table of deliveries and orders should be one to many relationships. Because the delivery staffs will deliver one or many orders in one delivery way. So the delivery systems will not delay. By splitting two table with one to many relationships, the company will not also get other problems. Also it would avoid the update anomalies problem.



Task 4

Assessment of Design

Mapped logical database design to physical database design.



In changing from Logical Database Design (ERD) into Physical Database Design (Table), we should know about DBMS and SQL query codes.

In logical design Shop Product and Order Details are connected into one to many relationship. By changing to physical design, Shop Product and Order Details tables are created.

According to ERD Shop Product is also connected with Products and Shops with one to many relationships in logical design. By changing them into physical design, tables are created to Shops and Products table that are connected with shop products in one to many relations.

Shops and Products have Shop Types and Categories in logical design and connected in one to many relationships. In order to change physical design, Shop Types and Categories are connected in one to many with shops and products as shown in figure.

Rating is a one to many relationship between Orders, Shop Product and Customer. Rating table is created and connected into one to many relationship with Shop Product and Orders and one to many with customers in physical design.

In logical design Orders and Order Details are connected into one to many relationship. By changing to physical design, Orders and Order Details tables are created.

Orders table is connected in one to one relationships with Payment. Orders and payments tables are created in one to one in physical design.

In ERD, Payment Methods is connected in one to many with Payment. In order to change physical design, Payment Methods and Payment are created in one to many relationships.

Customers table is a one to many relationship between Orders. Customers table is created and connected into one to many relationship with Orders in physical design.

Deliveries and Orders are connected in one to many relationships in logical design. After changing to physical, Deliveries and Orders are created with one to many relationships.

In ERD, Delivery Staffs and Shipping Methods are connected in one to many with Deliveries. In changing to physical design, Deliveries Staffs and Shipping Methods are created in one to many with Deliveries.

Designed tables for your target DBMS

To design for target DBMS, we should have knowledge about SQL. There are four types of SQL languages and they are DDL, DML, DIL and DCL. DDL is data definition language that used to create and modify tables. DDL includes create, alter drop. DML or Data Manipulation Language is used to add, delete and modify data. DML includes insert, delete and update statement. DIL is Data Integrity Language and DCL is Data Control Language.

The Create Table statement is used to create new table and there are column names behind it.

The Insert Into statement is used to new records into tables and put values in values parameter.

The Select statement is used to select data from tables.

The Alter table statement is used to add, delete, or modify columns and add and drop various constraints on existing tables in database.

The Update state is to modify records in a table.

The Drop statement is used to drop tables.

The Delete statement is used to delete existing records.

Derived

Derived data of total amount column in payment table.

Logical design

Amount	Quantity	Promotion	Total Amount
3000	5	100	14900
800	10	50	7950
4000	20	50	14900
3000	5	100	14900
500	12	50	5950
700	100	100	69900

Changing Logical to Physical design

Before

```
Select * from Payment
```

121 %

Results | Messages

	OrderID	PaymentMetho...	Payment_D...	Promoti...
1	Or0001	Pm0002	2021-02-21	100
2	Or0002	Pm0001	2021-02-22	50
3	Or0003	Pm0002	2021-02-18	50
4	Or0004	Pm0004	2021-02-18	100
5	Or0005	Pm0003	2021-02-19	50
6	Or0006	Pm0005	2021-02-19	100

After adding the total amount column in payment table.

```
Alter table Payment
Add TotalAmount integer
```

1 %

Messages

```
Command(s) completed successfully.
```

```
Select * from Payment
```

121 %

Results | Messages

	OrderID	PaymentMetho...	Payment_D...	Promoti...	TotalAmo...
1	Or0001	Pm0002	2021-02-21	100	NULL
2	Or0002	Pm0001	2021-02-22	50	NULL
3	Or0003	Pm0002	2021-02-18	50	NULL
4	Or0004	Pm0004	2021-02-18	100	NULL
5	Or0005	Pm0003	2021-02-19	50	NULL
6	Or0006	Pm0005	2021-02-19	100	NULL

After derived table to total amount column in payment table.

```
Update Payment
Set TotalAmount =(od.Amount * od.Quantity)- p.Promotion
From Order_Details od, Payment p, Orders o
where od.OrderID=o.OrderID And o.OrderID=p.OrderID;
```

0 %

Messages

```
(6 row(s) affected)
```

```
Select * from Payment
```

121 %

Results | Messages

	OrderID	PaymentMetho...	Payment_D...	Promoti...	TotalAmo...
1	Or0001	Pm0002	2021-02-21	100	14900
2	Or0002	Pm0001	2021-02-22	50	7950
3	Or0003	Pm0002	2021-02-18	50	79950
4	Or0004	Pm0004	2021-02-18	100	14900
5	Or0005	Pm0003	2021-02-19	50	5950
6	Or0006	Pm0005	2021-02-19	100	69900

Task 5

Scripts to Creates to table Structure

In order to change physical, we have to create table by using “create” statement from SQL Data Definition Language (DDL). Tables will be created by the situation of primary keys and foreign keys. I have reminded that the problem of creating foreign key tables before creating primary key table.

Coding and View of “Categories” Table

```
Create table Categories
(
    CategoryID varchar(10) not null,
    CategoryName varchar(30),
    Primary key(CategoryID)
);
```

```
Messages
Command(s) completed successfully.
```

Categories table is created by create statement. There are two columns in a Categories table.

```
Select * from Categories
```

```
Messages
CategoryID  CategoryNa...
```

Result of creating “Categories” table.

Coding and View of “Products” Table

```
Create table Products
(
    ProductID varchar(10) not null,
    ProductName varchar(30),
    CategoryID varchar(10),
    Primary key(ProductID),
    Foreign key (CategoryID) references Categories(CategoryID)
);
```

```
%
Messages
Command(s) completed successfully.
```

Products table is created by create statement. There are three columns in a Products table. CategoryID is a foreign key from Categories table.

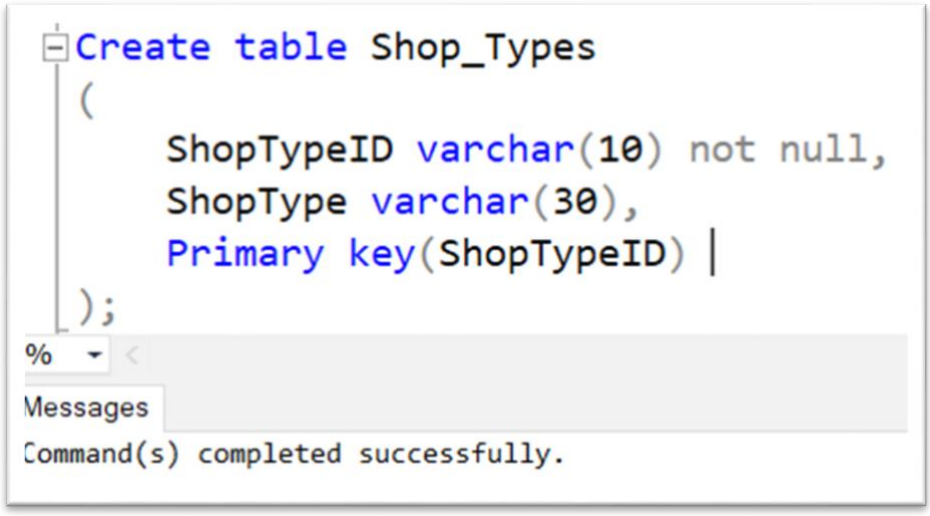
```
Select * from Products
```

esults | Messages

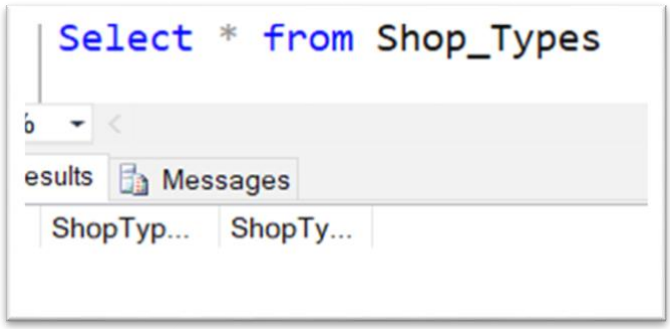
Product...	ProductNa...	CategoryID

Result of Products table.

Coding and View of “Shop Types” Table

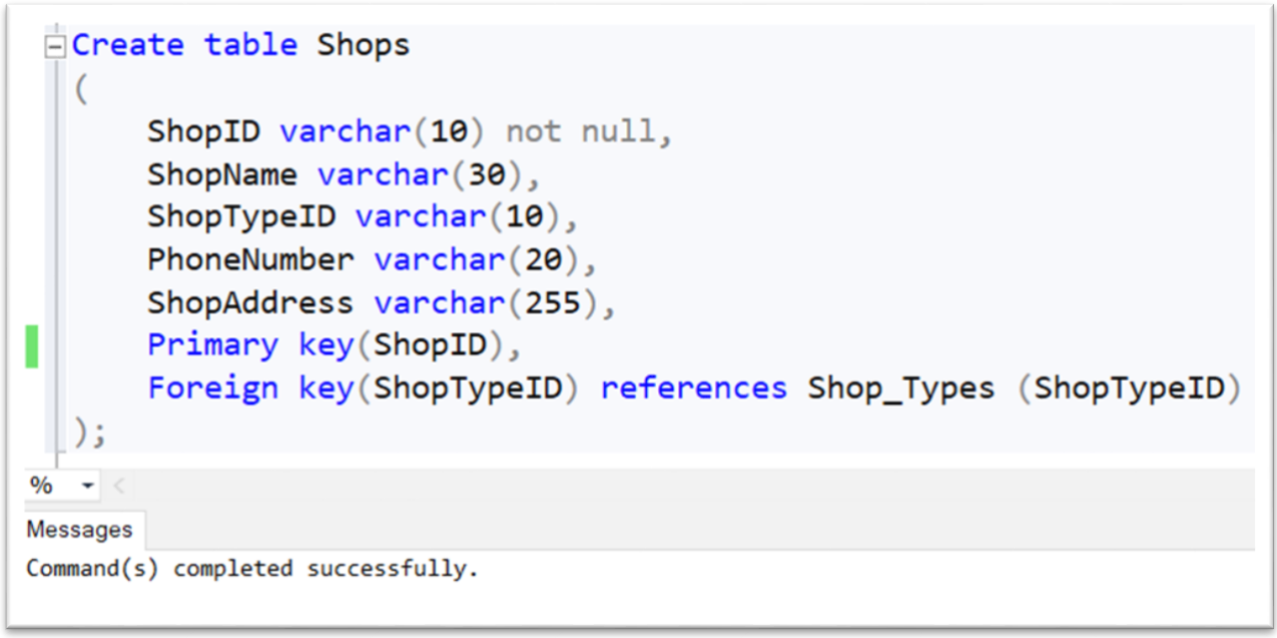


Shop Types table is created by create statement. There are two columns in a Shop Types table.

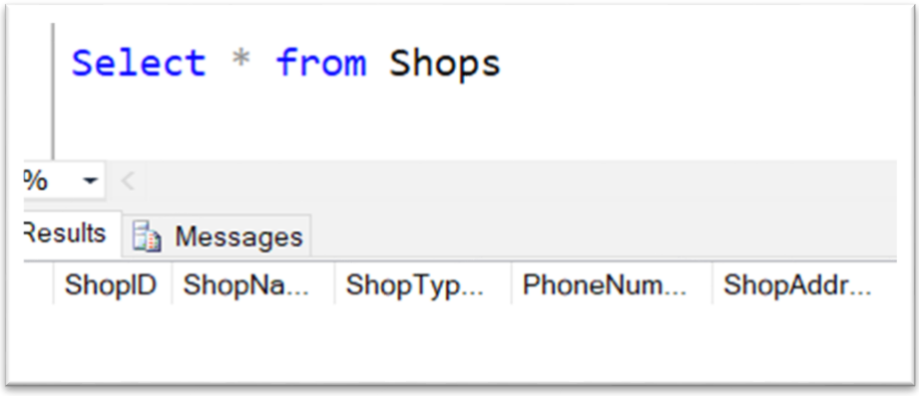


Results of Shop Types table.

Coding and View of "Shops" Table

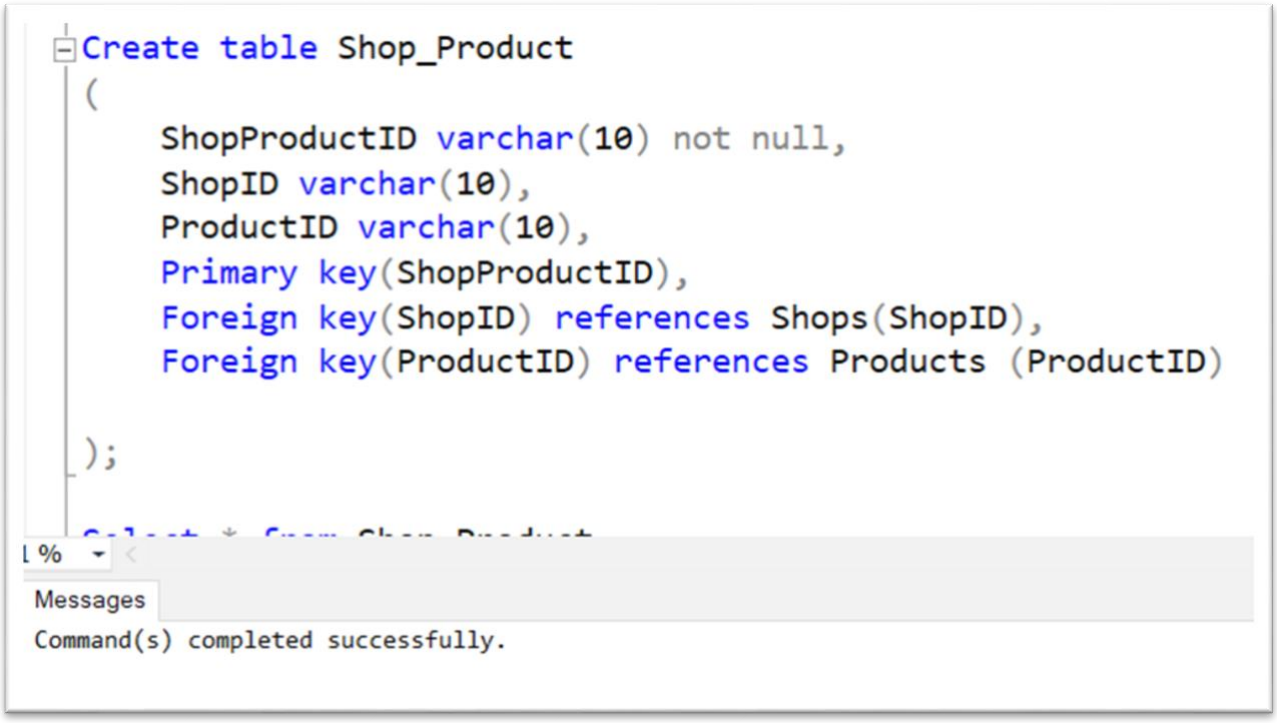


Shops table is created by create statement. There are three columns in a Shops table. ShopTypeID is a foreign key from ShopType table.



Result of Shops table.

Coding and View of “Shop Product” Table



Shop Product table is created by create statement. There are three columns in a Shops table. ShopID and ProductID are foreign key from Shops table and Product table.

```
Select * from Shop_Product
```

esults | Messages

ShopProduc...	Shop...	Product...

Result of Shop Product table.

Coding and View of “Payment Methods” Table

```
Create table Payment_Methods
(
    PaymentMethodID varchar(10) not null,
    PaymentMethodName varchar(30),
    Primary key(PaymentMethodID)
);
```

Messages

Command(s) completed successfully.

Payment Methods table is created by create statement. There are two columns in a Payment Methods table.

```
Select * from Payment_Methods
```

ults | Messages

PaymentMetho...	PaymentMethodNa...

Results of Payment Methods table.

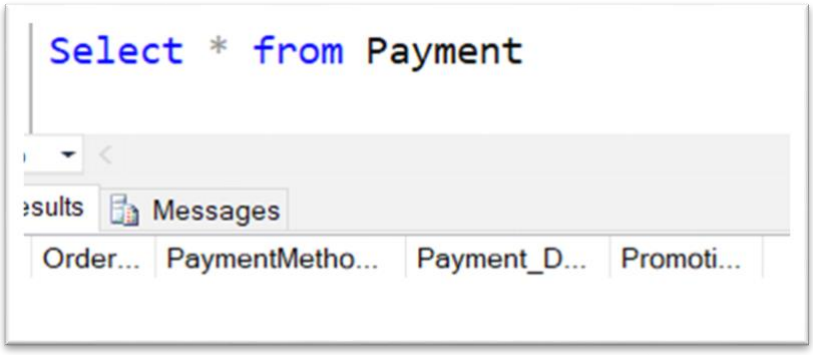
Coding and View of “Payment” Table

```
Create table Payment
(
    OrderID varchar(10) not null,
    PaymentMethodID varchar(10),
    Payment_Date date,
    Promotion integer,
    Primary key(OrderID),
    Foreign key(PaymentMethodID) references Payment_Methods (PaymentMethodID),

);
```

Messages

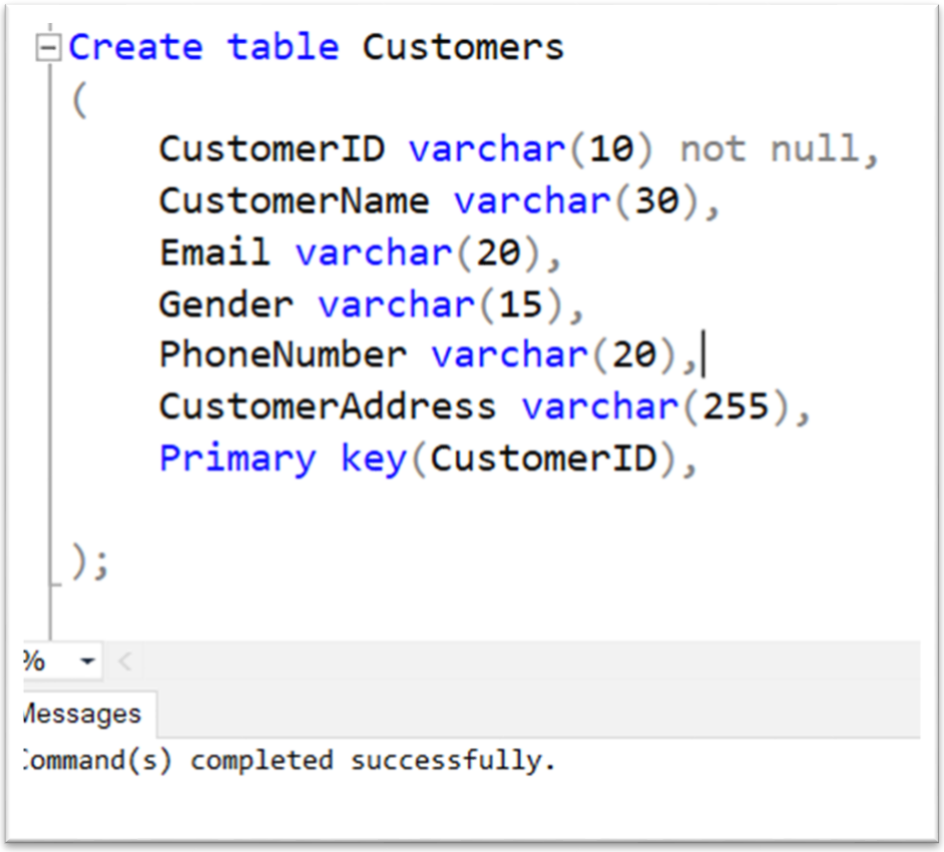
```
Command(s) completed successfully.
```

Payment table is created by create statement. There are four columns in a Payment table. PaymentMethodID is a foreign key from Payment Methods table.

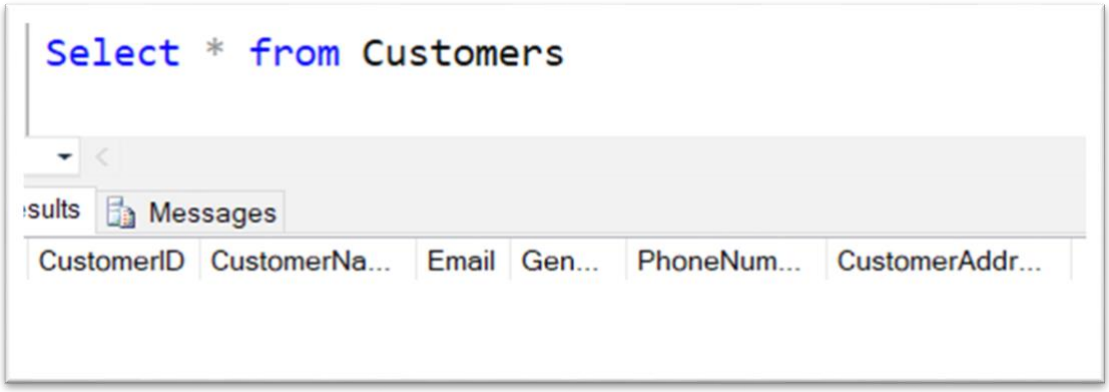


Result of Payment table.

Coding and View of “Customers” Table

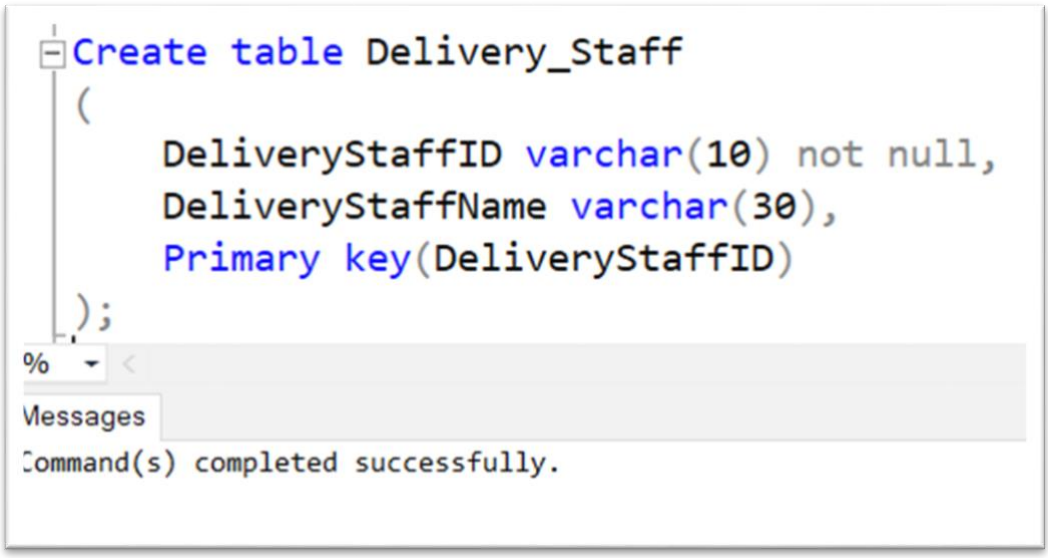


Customers table is created by create statement. There are six columns in a Customers table.

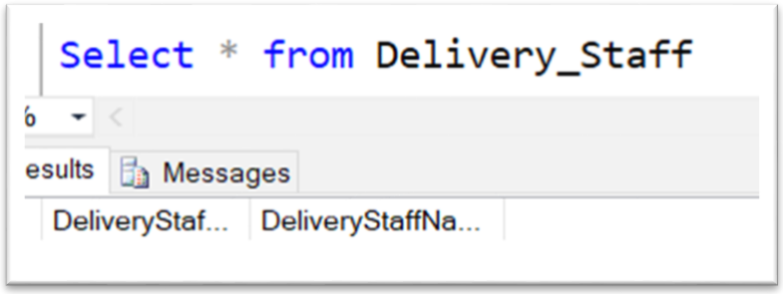


Result of Customers table.

Coding and View of “Delivery Staff” Table

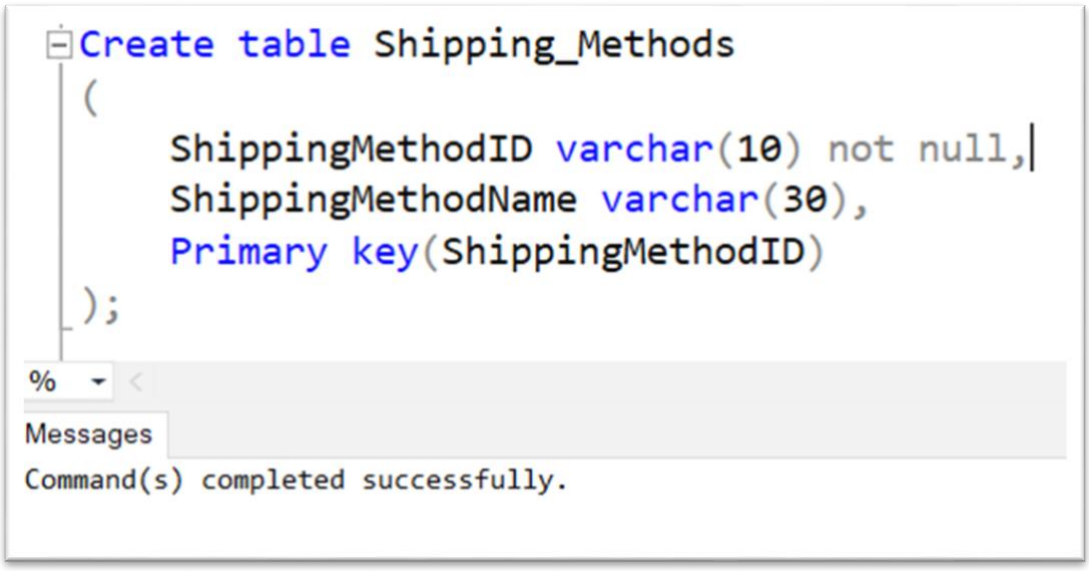


Delivery Staff table is created by create statement. There are two columns in a Delivery Staff table.



Result of Delivery Staff table.

Coding and View of "Shipping Methods" Table



Shipping Methods table is created by create statement. There are two columns in a shipping methods table.

```
Select * from Shipping_Methods
```

ılts Messages

ShippingMetho... ShippingMethodNa...

Result of Shipping Methods table.

Coding and View of “Deliveries” Table

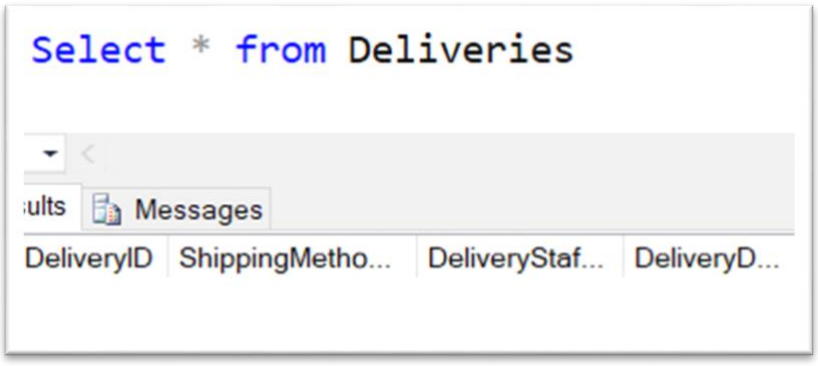
```
Create table Deliveries
(
DeliveryID varchar(10) not null,
ShippingMethodID varchar(10),
DeliveryStaffID varchar(10),
DeliveryDate date,
Primary key(DeliveryID),
Foreign key(ShippingMethodID) references Shipping_Methods (ShippingMethodID),
Foreign key(DeliveryStaffID) references Delivery_Staff (DeliveryStaffID)

);
```

%

Messages

Command(s) completed successfully.

Deliveries table is created by create statement. There are four columns in a Shops table. ShippingMethodID and DeliveryStaffID are foreign key from Shipping Methods table and Delivery Staff table.



Result of Deliveries table.

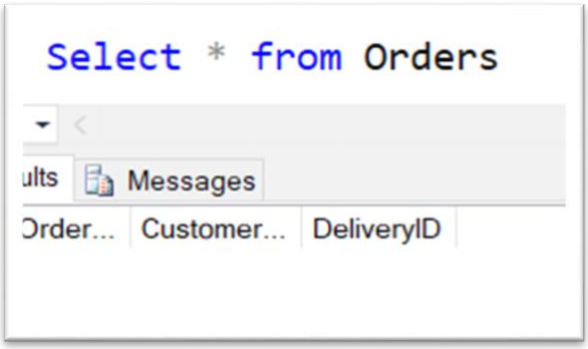
Coding and View of “ Orders ” Table

```
Create table Orders
(
OrderID varchar(10) not null,
CustomerID varchar(10),
DeliveryID varchar(10),
Primary key(OrderID),
Foreign key(CustomerID) references Customers (CustomerID),
Foreign key(DeliveryID) references Deliveries (DeliveryID)
 On delete cascade
 On update cascade

);
```

```
%
Messages
Command(s) completed successfully.
```

Orders table is created by create statement. There are three columns in an Orders table. CustomerID and DeliveryID are foreign key from Customers table and Deliveries table. There are propagation constraints included.

```
Select * from Orders
```



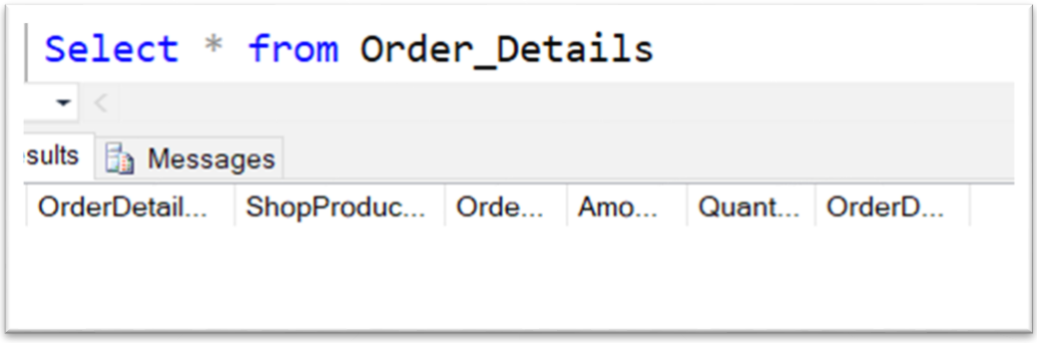
Result of Orders table.

Coding and View of “ Order_Details ” Table

```
Create table Order_Details
(
    OrderDetail_ID varchar(10) not null,
    ShopProductID varchar(10),
    OrderID varchar(10),
    Amount integer,
    Quantity integer,
    OrderDate date,
    Primary key(OrderDetail_ID),
    Foreign key(ShopProductID) references Shop_Product(ShopProductID)
    On delete no action,
    Foreign key(OrderID) references Orders(OrderID)
    On update cascade
    On delete cascade
);
```

```
%
Messages
Command(s) completed successfully.
```

Order Detail table is created by create statement. There are six columns in an Order Detail table. ShopProductID and OrderID are foreign key from Shop_Product table and Orders table. There are propagation constraints included.



Result of Order Detail table.

Coding and View of "Rating" Table

```
Create table Rating
(
RatingID varchar(10) not null,
ShopProductID varchar(10),
CustomerID varchar(10),
OrderID varchar(10),
Score integer,
Remark varchar(100),
DateRecorded date
Primary key(RatingID),
Foreign key(ShopProductID) references Shop_Product (ShopProductID),
Foreign key(CustomerID) references Customers (CustomerID),
Foreign key(OrderID) references Orders (OrderID)

);
```

```
%
Messages
Command(s) completed successfully.
```

Rating table is created by create statement. There are seven columns in a Rating table. CustomerID and OrderID are foreign key from Customerstable and Orders table.

```
Select * from Rating
```

```
%
Results  Messages
```

RatingID	ShopProduc...	Customer...	Orde...	Sco...	Rem...	DateRecord...

Result of Rating table.

Task 6

Data Population

Insert script and Result for “Caregories” table

```
Insert into Categories  values ('Ca0001','Tea');
Insert into Categories  values ('Ca0002','Milk' );
Insert into Categories  values ('Ca0003','Oil' );
Insert into Categories  values ('Ca0004','Drink' );
Insert into Categories  values ('Ca0005','Rice' );
Insert into Categories  values ('Ca0006','Snack' );
```

```
:1 %
Messages

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Categories
```

121 %

Results | Messages

	CategoryID	CategoryNa...
1	Ca0001	Tea
2	Ca0002	Milk
3	Ca0003	Oil
4	Ca0004	Drink
5	Ca0005	Rice
6	Ca0006	Snack

Insert script and Result for “Products” table

```
Insert into Products  values ('Pr0001','Yee Mon pickled tea leaves','Ca0001' );
Insert into Products  values ('Pr0002','Organic Soybean Oil','Ca0003'  );
Insert into Products  values ('Pr0003','Shark energy drink','Ca0004'  );
Insert into Products  values ('Pr0004','Swel sunflower seeds','Ca0006'  );
Insert into Products  values ('Pr0005','Milk Tea powder','Ca0001'  );
Insert into Products  values ('Pr0006','Shwe Bo Paw San rice','Ca0005'  );
```

Messages

```
(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Products
```

Results | Messages

	Product...	ProductName	CategoryID
1	Pr0001	Yee Mon pickled tea leaves	Ca0001
2	Pr0002	Organic Soybean Oil	Ca0003
3	Pr0003	Shark energy drink	Ca0004
4	Pr0004	Swel sunflower seeds	Ca0006
5	Pr0005	Milk Tea powder	Ca0001
6	Pr0006	Shwe Bo Paw San rice	Ca0005

Insert script and Result for “Shop Types” table

```
Insert into Shop_Types  values ('St0001','Retail' );
Insert into Shop_Types  values ('St0002','Wholesale' );
Insert into Shop_Types  values ('St0003','Hybrid' );
```

```
) %
Messages

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Shop_Types
```

100 %

Results | Messages

	ShopTyp...	ShopType
1	St0001	Retail
2	St0002	Wholesale
3	St0003	Hybrid

Insert script and Result for “Shops” table

```
Insert into Shops  values ('Sh0001','LoTaYa','St0001','+954758374','NaungDone St, Sancahung Tsp,Yangon' );
Insert into Shops  values ('Sh0002','MyanTea','St0002','+953847597','Main Rd, Kalaw Tsp, Shan State'  );
Insert into Shops  values ('Sh0003','Linn','St0001','+956847532','34th St, Latha Tsp, Yangon'  );
Insert into Shops  values ('Sh0004','Loi Hein','St0002','+954758438','WarTan Rd, Alone Tsp, Yangon'  );
Insert into Shops  values ('Sh0005','Platinum','St0001','+952847563','Dangon Tsp, Yangon'  );
Insert into Shops  values ('Sh0006','Swel','St0002','+958736453','HaKha District, Chin State'  );
```

%

Messages

```
(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Shops
```

100 %

Results | Messages

	ShopID	ShopNa...	ShopTyp...	PhoneNum...	ShopAddress
1	Sh0001	LoTaYa	St0001	+954758374	NaungDone St, Sancahung Tsp,Yangon
2	Sh0002	MyanTea	St0002	+953847597	Main Rd, Kalaw Tsp, Shan State
3	Sh0003	Linn	St0001	+956847532	34th St, Latha Tsp, Yangon
4	Sh0004	Loi Hein	St0002	+954758438	WarTan Rd, Alone Tsp, Yangon
5	Sh0005	Platinum	St0001	+952847563	Dangon Tsp, Yangon
6	Sh0006	Swel	St0002	+958736453	HaKha District, Chin State

Insert script and Result for “Customers” table

```
Insert into Customers  values ('Cu0001','Alexender','alex99@yahoo.com','Male','+9594658693','YikeThar St,YanKin Tsp,Yangon' );
Insert into Customers  values ('Cu0002','Andrew Myo','myoan@hotmail.com','Male','+9595869475','NweAye St,Tharkayta Tsp, Yangon'  );
Insert into Customers  values ('Cu0003','Nan Khin','nankhin111@gmail.com','Female','+9599747583','ShwePyi Rd, Taunggyi City, Shan'  );
Insert into Customers  values ('Cu0004','Pyone Cho','pyone34@gmail.com','Male','+9593008574','ShweThar Village, ShweBo Tsp, Sagaing'  );
Insert into Customers  values ('Cu0005','Khin Mya Mya','khinlay8@gmail.com','Female','+9596840385','34th Rd, ChanMyaTharSi Tsp, Mandalay'  );
Insert into Customers  values ('Cu0006','Daniel Saw','dsaw67@yahoo.com','Male','+9597694759','Main Rd, Hlaing Tsp,Yangon'  );
```

%

Messages

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

```
Select * from Customers
```

100 %

Results | Messages

	CustomerID	CustomerNa...	Email	Gender	PhoneNumber	CustomerAddress
1	Cu0001	Alexender	alex99@yahoo.com	Male	+9594658693	YikeThar St,YanKin Tsp,Yangon
2	Cu0002	Andrew Myo	myoan@hotmail.com	Male	+9595869475	NweAye St,Tharkayta Tsp, Yangon
3	Cu0003	Nan Khin	nankhin111@gmail.com	Female	+9599747583	ShwePyi Rd, Taunggyi City, Shan
4	Cu0004	Pyone Cho	pyone34@gmail.com	Male	+9593008574	ShweThar Village, ShweBo Tsp, Sagaing
5	Cu0005	Khin Mya Mya	khinlay8@gmail.com	Female	+9596840385	34th Rd, ChanMyaTharSi Tsp, Mandalay
6	Cu0006	Daniel Saw	dsaw67@yahoo.com	Male	+9597694759	Main Rd, Hlaing Tsp,Yangon

Insert script and Result for “Delivery Staff” table

```
Insert into Delivery_Staff  values ('Ds0001','Zaw Khun' );
Insert into Delivery_Staff  values ('Ds0002','Nan Khin' );
Insert into Delivery_Staff  values ('Ds0003','Aung Nay Myo' );
Insert into Delivery_Staff  values ('Ds0004','Zaw Myo Aung' );
Insert into Delivery_Staff  values ('Ds0005','James Aung' );
Insert into Delivery_Staff  values ('Ds0006','Wai Yan Lin' );
```

%

Messages

```
(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Delivery_Staff
```

100 %

Results | Messages

	DeliveryStaf...	DeliveryStaffNa...
1	Ds0001	Zaw Khun
2	Ds0002	Nan Khin
3	Ds0003	Aung Nay Myo
4	Ds0004	Zaw Myo Aung
5	Ds0005	James Aung
6	Ds0006	Wai Yan Lin

Insert script and Result for “Shipping Methods” table

```
Insert into Shipping_Methods  values ('Sm0001','Brought from warehouse' );
Insert into Shipping_Methods  values ('Sm0002','DHL' );
Insert into Shipping_Methods  values ('Sm0003','Shop Express' );
Insert into Shipping_Methods  values ('Sm0004','Royal Express' );
Insert into Shipping_Methods  values ('Sm0005','FedEx' );
```

```
%
Messages

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Shipping_Methods
```

100 %

Results | Messages

	ShippingMetho...	ShippingMethodName
1	Sm0001	Brought from warehouse
2	Sm0002	DHL
3	Sm0003	Shop Express
4	Sm0004	Royal Express
5	Sm0005	FedEx

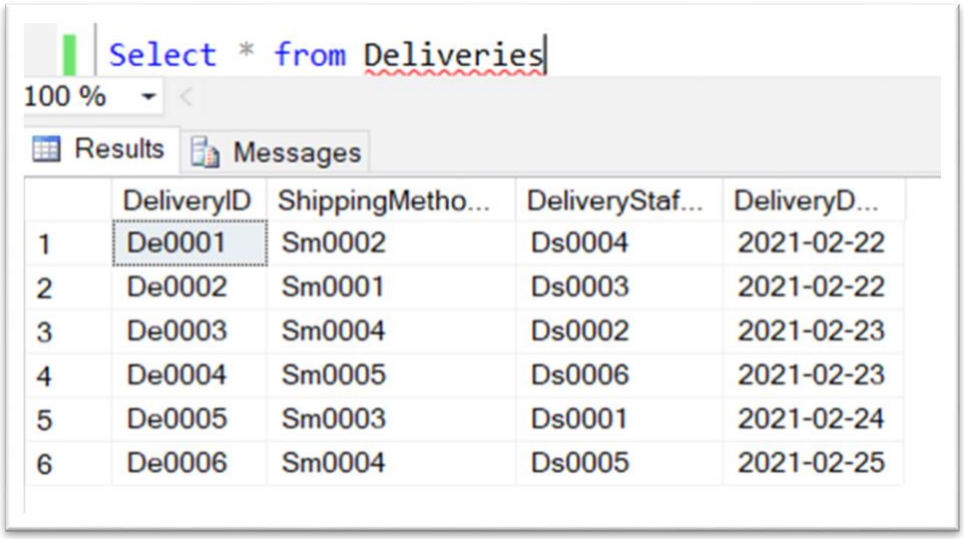
Insert script and Result for “Deliveries” table

```
Insert into Deliveries  values ('De0001','Sm0002','Ds0004','2021-02-22' );
Insert into Deliveries  values ('De0002','Sm0001','Ds0003','2021-02-22' );
Insert into Deliveries  values ('De0003','Sm0004','Ds0002','2021-02-23' );
Insert into Deliveries  values ('De0004','Sm0005','Ds0006','2021-02-23' );
Insert into Deliveries  values ('De0005','Sm0003','Ds0001','2021-02-24' );
Insert into Deliveries  values ('De0006','Sm0004','Ds0005','2021-02-25' );
```

%

Messages

```
(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```



	DeliveryID	ShippingMetho...	DeliveryStaf...	DeliveryD...
1	De0001	Sm0002	Ds0004	2021-02-22
2	De0002	Sm0001	Ds0003	2021-02-22
3	De0003	Sm0004	Ds0002	2021-02-23
4	De0004	Sm0005	Ds0006	2021-02-23
5	De0005	Sm0003	Ds0001	2021-02-24
6	De0006	Sm0004	Ds0005	2021-02-25

Insert script and Result for “Orders” table

```
Insert into Orders  values ('Or0001','Cu0001','De0001' );
Insert into Orders  values ('Or0002','Cu0002','De0002' );
Insert into Orders  values ('Or0003','Cu0003','De0003' );
Insert into Orders  values ('Or0004','Cu0001','De0003' );
Insert into Orders  values ('Or0005','Cu0004','De0004' );
Insert into Orders  values ('Or0006','Cu0002','De0004' );
```

) %

Messages

```
(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Orders
```

100 %

Results | Messages

	OrderID	Customer...	DeliveryID
1	Or0001	Cu0001	De0001
2	Or0002	Cu0002	De0002
3	Or0003	Cu0003	De0003
4	Or0004	Cu0001	De0003
5	Or0005	Cu0004	De0004
6	Or0006	Cu0002	De0004

Insert script and Result for “Shop Product” table

```
Insert into Shop_Product values ('Sp0001','Sh0002','Pr0001' );
Insert into Shop_Product values ('Sp0002','Sh0004','Pr0003'  );
Insert into Shop_Product values ('Sp0003','Sh0005','Pr0005'  );
Insert into Shop_Product values ('Sp0004','Sh0006','Pr0004'  );
Insert into Shop_Product values ('Sp0005','Sh0005','Pr0003'  );
Insert into Shop_Product values ('Sp0006','Sh0001', 'Pr0006' );
```

```
Messages

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Shop_Product
```

Results | Messages

	ShopProduc...	ShopID	Product...
1	Sp0001	Sh0002	Pr0001
2	Sp0002	Sh0004	Pr0003
3	Sp0003	Sh0005	Pr0005
4	Sp0004	Sh0006	Pr0004
5	Sp0005	Sh0005	Pr0003
6	Sp0006	Sh0001	Pr0006

Insert script and Result for “Order Details” table

```
Insert into Order_Details  values ('Od0001','Sp0003','Or0001','3000','5','2021-02-18' );
Insert into Order_Details  values ('Od0002','Sp0005','Or0002','800','10','2021-02-18' );
Insert into Order_Details  values ('Od0003','Sp0006','Or0003','4000','20','2021-02-19' );
Insert into Order_Details  values ('Od0004','Sp0001','Or0004','3000','5','2021-02-18' );
Insert into Order_Details  values ('Od0005','Sp0004','Or0005','500','12','2021-02-18' );
Insert into Order_Details  values ('Od0006','Sp0002','Or0006','700','100','2021-02-18' );
```

%

Messages

```
(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Order_Details
```

121 %

Results | Messages

	OrderDetail...	ShopProduc...	OrderID	Amo...	Quant...	OrderDate
1	Od0001	Sp0003	Or0001	3000	5	2021-02-18
2	Od0002	Sp0005	Or0002	800	10	2021-02-18
3	Od0003	Sp0006	Or0003	4000	20	2021-02-19
4	Od0004	Sp0001	Or0004	3000	5	2021-02-18
5	Od0005	Sp0004	Or0005	500	12	2021-02-18
6	Od0006	Sp0002	Or0006	700	100	2021-02-18

Insert script and Result for “Payment Methods” table

```
Insert into Payment_Methods  values ('Pm0001','Cash On Delivery');
Insert into Payment_Methods  values ('Pm0002','KBZ Pay' );
Insert into Payment_Methods  values ('Pm0003','Wave Pay' );
Insert into Payment_Methods  values ('Pm0004','Bank Transfer' );
Insert into Payment_Methods  values ('Pm0005','Credit/Debit Card' );
Insert into Payment_Methods  values ('Pm0006','Pay Pal' );
```

%

Messages

```
(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Payment_Methods
```

121 %

Results | Messages

	PaymentMetho...	PaymentMethodNa...
1	Pm0001	Cash On Delivery
2	Pm0002	KBZ Pay
3	Pm0003	Wave Pay
4	Pm0004	Bank Transfer
5	Pm0005	Credit/Debit Card
6	Pm0006	Pay Pal

Insert script and Result for “Payment” table

```
Insert into Payment  values ('Or0001','Pm0002','2021-02-21','100' );
Insert into Payment  values ('Or0002','Pm0001','2021-02-22','50' );
Insert into Payment  values ('Or0003','Pm0002','2021-02-18','50' );
Insert into Payment  values ('Or0004','Pm0004','2021-02-18','100' );
Insert into Payment  values ('Or0005','Pm0003','2021-02-19','50' );
Insert into Payment  values ('Or0006','Pm0005','2021-02-19','100' );
```

0 %

Messages

```
(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Payment
```

121 %

Results | Messages

	OrderID	PaymentMetho...	Payment_D...	Promoti...
1	Or0001	Pm0002	2021-02-21	100
2	Or0002	Pm0001	2021-02-22	50
3	Or0003	Pm0002	2021-02-18	50
4	Or0004	Pm0004	2021-02-18	100
5	Or0005	Pm0003	2021-02-19	50
6	Or0006	Pm0005	2021-02-19	100

Insert script and Result for “Rating” table

```
Insert into Rating  values ('Ra0001','Sp0003','Cu0002','Or0001','8','good taste','2021-02-28' );
Insert into Rating  values ('Ra0002','Sp0006','Cu0002','Or0002','7','null','2021-02-28' );
Insert into Rating  values ('Ra0003','Sp0001','Cu0003','Or0003','8','good price','2021-03-03' );
Insert into Rating  values ('Ra0004','Sp0003','Cu0001','Or0004','5','not bad','2021-02-28' );
Insert into Rating  values ('Ra0005','Sp0002','Cu0004','Or0005','6','need something','2021-03-21' );
Insert into Rating  values ('Ra0006','Sp0005','Cu0005','Or0006','4','I do not like','2021-03-01' );
```

1 %

Messages

```
(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)

(1 row(s) affected)
```

```
Select * from Rating
```

121 %

Results | Messages

	RatingID	ShopProduc...	Customer...	OrderID	Sco...	Remark	DateRecord...
1	Ra0001	Sp0003	Cu0002	Or0001	8	good taste	2021-02-28
2	Ra0002	Sp0006	Cu0002	Or0002	7	null	2021-02-28
3	Ra0003	Sp0001	Cu0003	Or0003	8	good price	2021-03-03
4	Ra0004	Sp0003	Cu0001	Or0004	5	not bad	2021-02-28
5	Ra0005	Sp0002	Cu0004	Or0005	6	need something	2021-03-21
6	Ra0006	Sp0005	Cu0005	Or0006	4	I do not like	2021-03-01

Task 7

SQL Report

Select Shop Name, Shop Address, Shop Type, Product Name, Category Name

Selecting Shop Name, Shop Address, Product Name and Category Name from Shops, Products, Shop Product and Categories tables.

```
Select s.ShopName, s.ShopAddress, st.ShopType, p.ProductName, c.CategoryName
from Shops s, Products p , Shop_Product sp , Categories c, Shop_Types st
Where s.ShopID=sp.ShopID And sp.ProductID=p.ProductID And p.CategoryID=c.CategoryID
and s.ShopTypeID=st.ShopTypeID
```

Result

Results | Messages

	ShopNa...	ShopAddress	ShopType	ProductName	CategoryNa...
1	MyanTea	Main Rd, Kalaw Tsp, Shan State	Wholesale	Yee Mon pickled tea leaves	Tea
2	Loi Hein	WarTan Rd, Alone Tsp, Yangon	Wholesale	Shark energy drink	Drink
3	Platinum	Dangon Tsp, Yangon	Retail	Milk Tea powder	Tea
4	Swel	HaKha District, Chin State	Wholesale	Swel sunflower seeds	Snack
5	Platinum	Dangon Tsp, Yangon	Retail	Shark energy drink	Drink
6	LoTaYa	NaungDone St, Sancahung Tsp,Yangon	Retail	Shwe Bo Paw San rice	Rice

Select Order ID, Product Name, Order Date, Customer Name

Selecting Order ID, Product Name, Order Date and Customer Name from Products, Order Details, Orders, Customers and Shop Product

```
Select o.OrderID, p.ProductName, od.OrderDate, c.CustomerName
from Products p, Order_Details od, Orders o, Customers c, Shop_Product sp
Where c.CustomerID=o.CustomerID And o.OrderID=od.OrderID And
od.ShopProductID=sp.ShopProductID And sp.ProductID=p.ProductID
```

Result

Results | Messages

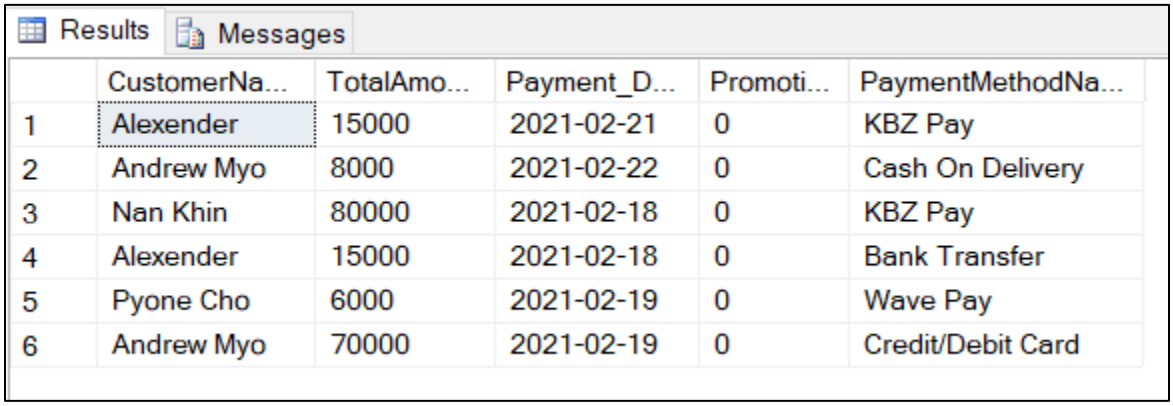
	OrderID	ProductName	OrderDate	CustomerNa...
1	Or0001	Milk Tea powder	2021-02-18	Alexender
2	Or0002	Shark energy drink	2021-02-18	Andrew Myo
3	Or0003	Shwe Bo Paw San rice	2021-02-19	Nan Khin
4	Or0004	Yee Mon pickled tea leaves	2021-02-18	Alexender
5	Or0005	Swel sunflower seeds	2021-02-18	Pyone Cho
6	Or0006	Shark energy drink	2021-02-18	Andrew Myo

Select Customer Name, TotalAmount, Payment Date, Promotion, Payment Method Name

Select Customer Name, TotalAmount, Payment Date, Promotion, Payment Method Name from Customers, Orders, Payment, Payment_Methods tables

```
Select cu.CustomerName, p.TotalAmount, p.Payment_Date, p.Promotion, pm.PaymentMethodName
from Customers cu, Orders o, Payment p, Payment_Methods pm
where cu.CustomerID=o.CustomerID And p.OrderID=o.OrderID And p.PaymentMethodID=pm.PaymentMethodID
```

Result



Results | Messages

	CustomerNa...	TotalAmo...	Payment_D...	Promoti...	PaymentMethodNa...
1	Alexender	15000	2021-02-21	0	KBZ Pay
2	Andrew Myo	8000	2021-02-22	0	Cash On Delivery
3	Nan Khin	80000	2021-02-18	0	KBZ Pay
4	Alexender	15000	2021-02-18	0	Bank Transfer
5	Pyone Cho	6000	2021-02-19	0	Wave Pay
6	Andrew Myo	70000	2021-02-19	0	Credit/Debit Card

Select Customer Name, Delivery Date DeliveyStaffName, ShippingMethodName

Selecting Customer Name, Delivery Date, Delivey Staff Name, Shipping Method Name from Customers, Orders, Deliveries, Delivery Staff, Shipping Methods.

```
Select cu.CustomerName, d.DeliveryDate, ds.DeliveryStaffName, sm.ShippingMethodName
from Customers cu , Orders o ,  Deliveries d ,  Delivery_Staff ds , Shipping_Methods sm
where cu.CustomerID=o.CustomerID and o.DeliveryID=d.DeliveryID and d.DeliveryStaffID=ds.DeliveryStaffID and
 d.ShippingMethodID= sm.ShippingMethodID
```

Result

Results | Messages

	CustomerNa...	DeliveryD...	DeliveryStaffNa...	ShippingMethodName
1	Alexender	2021-02-22	Zaw Myo Aung	DHL
2	Andrew Myo	2021-02-22	Aung Nay Myo	Brought from warehouse
3	Nan Khin	2021-02-23	Nan Khin	Royal Express
4	Alexender	2021-02-23	Nan Khin	Royal Express
5	Pyone Cho	2021-02-23	Wai Yan Lin	FedEx
6	Andrew Myo	2021-02-23	Wai Yan Lin	FedEx

Select Total Amount of Order

Selecting Total Amount of Order from Payment to know income.

```
Select SUM(p.TotalAmount) As Total_Amount_of_order
from Payment p
```

Result

Results | Messages

	Total_Amount_of_or...
1	194000

Select Total Score

Selecting total score form rating table to know company score.

```
Select SUM(r.Score) As Score_of_product
from Rating r
```

Result

Results Messages

	Score_of_prod...
1	38

Select Remark, Shop Name, Product Name

Selecting Remark, Shop Name, Product Name from rating, products and shops tables.

```
Select r.Remark , p.ProductName, s.ShopName
from Rating r ,Shop_Product sp , Products p, Shops s
where r.ShopProductID=sp.ShopProductID and sp.ProductID=p.ProductID and sp.ShopID=s.ShopID
```

Result

Results Messages

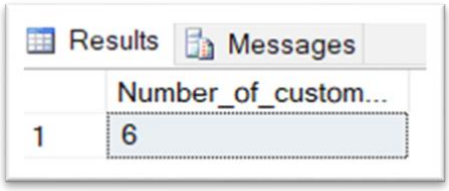
	Remark	ProductName	ShopNa...
1	good taste	Milk Tea powder	Platinum
2	null	Shwe Bo Paw San rice	LoTaYa
3	good price	Yee Mon pickled tea leaves	MyanTea
4	not bad	Milk Tea powder	Platinum
5	need something	Shark energy drink	Loi Hein
6	I do not like	Shark energy drink	Platinum

Select numbers of customers

Selecting numbers of customers from customers table.

```
Select COUNT(c.CustomerID) As Number_of_customers
From Customers c
```

Result

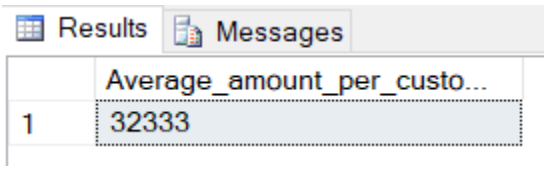


Select Average Amount per Customers

Selecting average amount per customers from payment table.

```
Select AVG(p.TotalAmount) As Average_amount_per_customers
From Payment p
```

Result

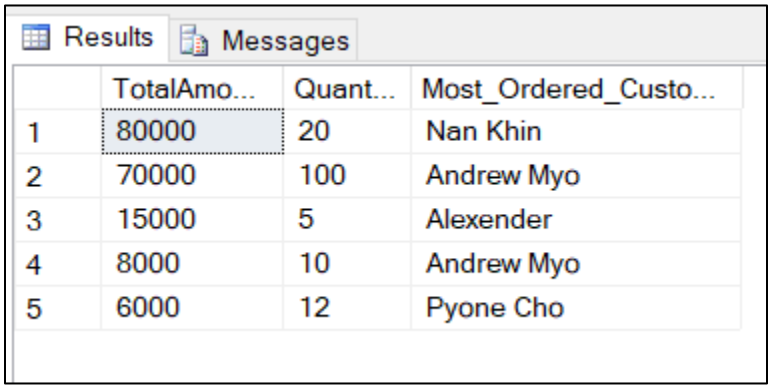


Select Total Amount, Quantity and Most Ordered Customers

Selecting Total Amount, Quantity and Most Ordered Customers from Payment, Orders, Customers and Order Details table

```
Select p.TotalAmount, od.Quantity, MAX(c.CustomerName) As Most_Ordered_Customer
from Payment p, Orders o, Customers c, Order_Details od
where p.OrderID=o.OrderID And o.OrderID=od.OrderID
And o.CustomerID=c.CustomerID
Group By p.TotalAmount, od.Quantity
Order By p.TotalAmount Desc
```

Result



	TotalAmo...	Quant...	Most_Ordered_Custo...
1	80000	20	Nan Khin
2	70000	100	Andrew Myo
3	15000	5	Alexender
4	8000	10	Andrew Myo
5	6000	12	Pyone Cho

Select Total Amount, Quantity and Second Most Ordered Customers

Selecting Total Amount, Quantity and Second Most Ordered Customers from Payment, Orders, Customers and Order Details table

```
Select p.TotalAmount, od.Quantity, MAX(c.CustomerName) As Most_Ordered_Customer
from Payment p, Orders o, Customers c, Order_Details od
where p.OrderID=o.OrderID And o.OrderID=od.OrderID
And o.CustomerID=c.CustomerID
Group By p.TotalAmount, od.Quantity
Order By p.TotalAmount Desc
```

Result

Results | Messages

	TotalAmo...	Quant...	Most_Ordered_Custo...
1	80000	20	Nan Khin
2	70000	100	Andrew Myo
3	15000	5	Alexender
4	8000	10	Andrew Myo
5	6000	12	Pyone Cho

Task 8

Future development of a distribute database

Distributed Database System

A distributed database (DDB) is an integrated collections of databases in which data is stored across different physical locations. A Distributed Database Management System (DDBMS) is a software system that manages a distributed database so that distribution aspects are transparent to users. (Anon., 2021)

Components of Distributed Database System

Hardware, Communications Media, Software, Data Processors, Transaction Processors are the components of distributed database management system.

Distributed database

Advantages

The advantages of distributed database are Modular Development, More Reliable, Better Response, Lower Communication Cost, Improved Performance, Improved share ability and local autonomy. (Anon., 2021)

Disadvantages

The disadvantages of distributed database are Complexity, Cost, Security, Integrity Control more difficult, Lack of Standards, Lack of experience, Database Design more complex. (Anon., 2021)

Organization and Geographic Structure

The organization will use both replication and fragmentation. In replication, the data will be copy from main database to side database so the data system will more effect in improving. Admins and users of organization can share and copy same data with consistency. In fragmentation, Full database or table will separating into site tables. Both will suitable for organization.

There are two types of distributed database. They are Homogeneous and Heterogeneous. Organization will use heterogeneous because different sites can use different operating system, DBMS and data models.

Replication

Replication is the process of copying data from a central database to one or more databases. (Anon., 2021)

Fragmentation

Fragmentation is a feature of the database server that allows you to control where data is stored at the table level. Fragmentation allows you to define groups of rows or index keys in a table according to some algorithm or schema. (Anon., 2021)

Homogeneous and Heterogeneous

All the sites use identical DBMS and operating systems in a homogeneous distributed database. Different sites have different operating systems, database products, and data models in a heterogeneous distributed database. (Anon., 2021)

How a distributed database might allow the organization’s business to adapt to potential future expansion.

Distributed database system is used in my organization. It will give some benefits for organization. The first one is Emulating organizational structure and this means that if different functions are distributed across different sites, then the data can be distributed accordingly. Then the control will grater because data is located where it is needed, so it is easier to control it. The availability will be improved. Access to locally stored data is less likely to be disrupted due to network problems or problems with the central database. Data stored locally is more likely to be under the control of those who know about it and therefore can keep it in a reliable state is to get greater reliability.

Task 9

Evaluate

Description

I developed grocery ordering system because traditional grocery ordering system are not systematics. There are not relationships between customers and shops by shop. Customers can look products in one place. There are thirteen tables and developed by 9 tasks.

Feeling

I am very happy to develop this system. There is Covid-19 pandemic cases so I can't interview outside. But I interview from online. I know many problems and I glad to solve the problems.

Evaluation

By doing this assignment, I researched more about POS system. Then I researched more about grocery shops. So, I got knowledge about traditional grocery ordering system and their problem. I already have some knowledge about traditional ordering system so I have easy ways to solve the problem. I had good prepared for SQL language so I didn't have much difficulty.

I have some confused about normalization. So, I watched You Tube and used other search engines to know about it.

Analysis

I analyzed the big problem of ordering and delivering in traditional system. In traditional system, shops are separated and delivery system are also separated. So that customers would not know many shops and their products and their relationships. In this system, customers can buy from different shops in one place. A delivery way for an order for retail shops is the problem and delivery system in delay in retail shop. For wholesale shops, the delivery system has only few problems. In this system, there are many orders in a delivery way. And customers can buy different products from different shops.

Conclusion

I have some knowledge about traditional grocery ordering system and then I researched the problem of traditional system. I got the problem about traditional system easily but I had to think about the solutions difficultly. Because there are no like this database system before in our environment. I learnt more problems from environment. At first, I don't really know about details

solutions and rash the assignment. Then I make corrections about the errors. Also, there were some SQL statement errors because of ERD. I should prepare for plan of assignment. I should not rash the assignment next times. Also, I learnt some challenges from this assignment.

Action Plan

If I will do this system update, I will not rash it and I should prepare for the right ways. Draw ERD diagram based upon detail problems and solutions. There are some challenges about SQL. I should practice SQL before making the system.

References

Anon., 2021. *BenchPartner.* [Online]
Available at: https://benchpartner.com/advantage-and-disadvantage-of-distributed-database-management-system-ddbms
[Accessed 29 10 2021].

Anon., 2021. *Data Dictionary.* [Online]
Available at: http://library.ucmerced.edu/data-dictionaries

Anon., 2021. *IBM.* [Online]
Available at: https://www.ibm.com/docs/en/i2-ibase/8.9.13?topic=replication-what-is-database
[Accessed 21 October 2021].

Anon., 2021. *IBM.* [Online]
Available at: https://www.ibm.com/docs/en/informix-servers/14.10?topic=strategies-what-is-fragmentation
[Accessed 21 October 2021].

Anon., 2021. *JavaTpoint.* [Online]
Available at: https://www.javatpoint.com/dbms-purpose-of-normalization
[Accessed 6 10 2021].

Anon., 2021. *NCC Database Design and Development.* s.l.:NCC.

Anon., 2021. *Ques10.* [Online]
Available at: https://www.ques10.com/p/17615/what-are-the-homogeneous-and-heterogeneous-distr-1/
[Accessed 21 October 2021].

Anon., 2021. *Wikipedia.* [Online]
Available at: https://en.wikipedia.org/wiki/Distributed_database
[Accessed 21 October 2021].