Kunal Mukherjee (kxm180046)
Suraj Kothawade (snk170001)
Chia-Kai Pan (cxp170010)
5/1/20
CS 6360
Dr. Weili Wu
Dr. Smita Ghosh

Project phase 4 Submission:

a. **Problem Description:**

Food Festive Supermarket, a grocery store in Richardson, would like one relational database to store the information about their management system to be able to carry out their work in an organized way. They have some major modules such as Person, Products and Billing.
A Person must be an Employee or a Silver Customer. Details of a person such as ID, Name (First, Middle, Last), Address, Gender, Date of Birth (Must be 16 years or older), and Phone number (one person can have more than one phone number) are recorded. The Person ID should have the format "PXXX" where X is a number from 0 to 9.
A Silver Customer is classified as online or non-online customer. An online customer can also be a non-online customer. The email address of the online customer is stored. Only online customers can order products online. The order details such as order number, date, amount, product details and customer details are stored. One online customer can order multiple products and a product can be ordered by multiple online customers.
Employee is classified as Cashier, Floor Staff or Managers. A floor staff can be promoted to cashier and later become a manager. The start date for each designation is recorded. Each floor staff is assigned the duty of arranging products into aisles. Aisle information such as section and aisle number is recorded. The date of assignment along with aisle number is stored for each employee. One employee maybe be assigned to arrange different aisles. The information about product-aisle arrangement and the date of arrangement is also stored. Products do not have a fixed aisle and can be arranged in different aisles throughout the year.
Each employee works at a store. One employee can work in multiple stores but on a given day, can work only at one store. The date and working hours of the employee are stored. Store information such as name, address and contact are stored. Each store offers 'Sale' from time to time. Details such as sale ID, description and duration is recorded. The sale IDs are not unique and cannot be used to identify a sale in the system.
A bill transaction is made by a cashier who records the list of products that are purchased by a person along with the date of purchase, bill amount, store ID and payment method. The cashier details, person details, store details and product details are stored together.
A Gold Customer is someone who has some extra privileges than a Silver Customer. A Gold customer can be an Employee or a Non-online Customer or both. Different vouchers are issued by the store. A non-online customer needs to buy these vouchers but vouchers are given to a Gold Customers each month free of cost. Sometimes promotional discounts are offered on the vouchers and details such promotion ID and promotion description are recorded. The Promotional IDs are not unique and cannot be used to identify a promotion in the system. Each Gold Customer is issued a membership card. A unique membership ID is generated for each Gold Customer. This number, date of issue and other information are stored.
Product details such as product ID, Quantity (0,if out of stock), description and other information are stored. Products are further classified as either perishable or non-perishable items. Date of expiry is stored for the perishable items. Various suppliers, whose information are also stored in the system supply products. One Supplier may supply more than one product. But one product is supplied by only one supplier.

b. Project questions

1. Is the ability to model superclass/subclass relationships likely to be important in a grocery system environment such as Food Festive? Why or why not?

Yes, modeling the superclass/subclass relationship is imperative in such a grocery system environment. It is evident that there ought to be natural subclasses semantically. For instance, a cashier, manager, and floor staff are all employees. Hence, if these classes were to be implemented in an Object-Oriented fashion, an employee would be the superclass, and each from the cashier, manager, and floor staff would be a subclass. Moreover, this is ideal because we would want cashier, manager, and floor staff to inherit all the properties (attributes and relations) of the superclass employee. Similarly, requirements like, both perishable and non-perishable items are products, silver customers can be online and non-online; can be modeled naturally using a superclass/subclass relationships. Hence, these relationships are more intuitive in terms of design and implementation perspectives of a grocery system environment like scenarios.

2. Can you think of 5 more business rules (other than the one explicitly described above) that are likely to be used in a supermarket environment? Add your rules to the above requirement to be implemented.

1.An employee can only be one of Floor Staff, Manager, and cashier. Means that an employee cannot play two or three roles at the same times.
2.A Manager should have relevant working experience at least 2 years.
3. Only one entry can be made per day for how many hours in a day did the employee work for.
4.One Gold Customer can assign two more people to become silver customer for free. If Gold Customer want to assign more people, it needs to cost some money.
5.No more than two vouchers can be used at the same time.

3. Justify using a Relational DBMS like Oracle for this project.

We need a relational DBMS like oracle because it allows the creation of tables which can be represented by a mathematical structure, relation. Each relation or a table contains a column or columns that other tables can key on together, to get the information from that table. By storing this information in another table, the database can create a single table with the locations that can then be used for a variety of purposes by other tables in the database. Since, our conceptual schema was created with this aspect in mind, we do not have schema structures that are suited for non-relational database models. Relational database also keeps our information safe, thanks to ACID property. ACID is Atomicity, Consistency, Isolation, and Durability. ACID is a set of properties that are used when modifying a database. They guarantee that transactions are valid even in the event that you may encounter an error, power failure, crash, etc. Therefore, we use a Relational DBMS like Oracle for this project because it guarantees that our database is valid all the time, and in case of error we can re-store the state back and we can extract the power of relational database structure. This is imperative for a big departmental store, because any loss of data or crash of DBMS would result in loss of money, because the time we will spend on fixing the DBMS, we will lose out on customers and also have a high chance of losing monetary information within the inside.

c. Assumptions and EER diagram:

> design decision was made to separate store order from online order, because the TA made us aware that SQL does linear search. For high volume of data, our design will be faster. But we do understand that because of our design we have data redundancy

> Also, in this light level schema we are ignoring the constraint that a employee can work on only one store in a given day

> Multiple contact information can be there for store
> All store has bill option

> All employee will have an associated store
> Person can have multiple address
> Person can be only one type of gender
> All store must have one employee working

> All cashier bills
> Only one person for each store order
> Only one store for each store order
> Only one cashier for each for each bill

> Every sale will have a unique Sale number
> Silver Customer must be one of online and non-online
> A person may be both silver customer and Employee
> All vouchers can be bought by non-online customer
> A voucher has at least one promotional code
> All non-online customer may not buy vouchers
> A store can issue multiple voucher
> A same voucher cannot be issued by multiple store
> A voucher can have multiple promotion offer
> Promotion Number is unique

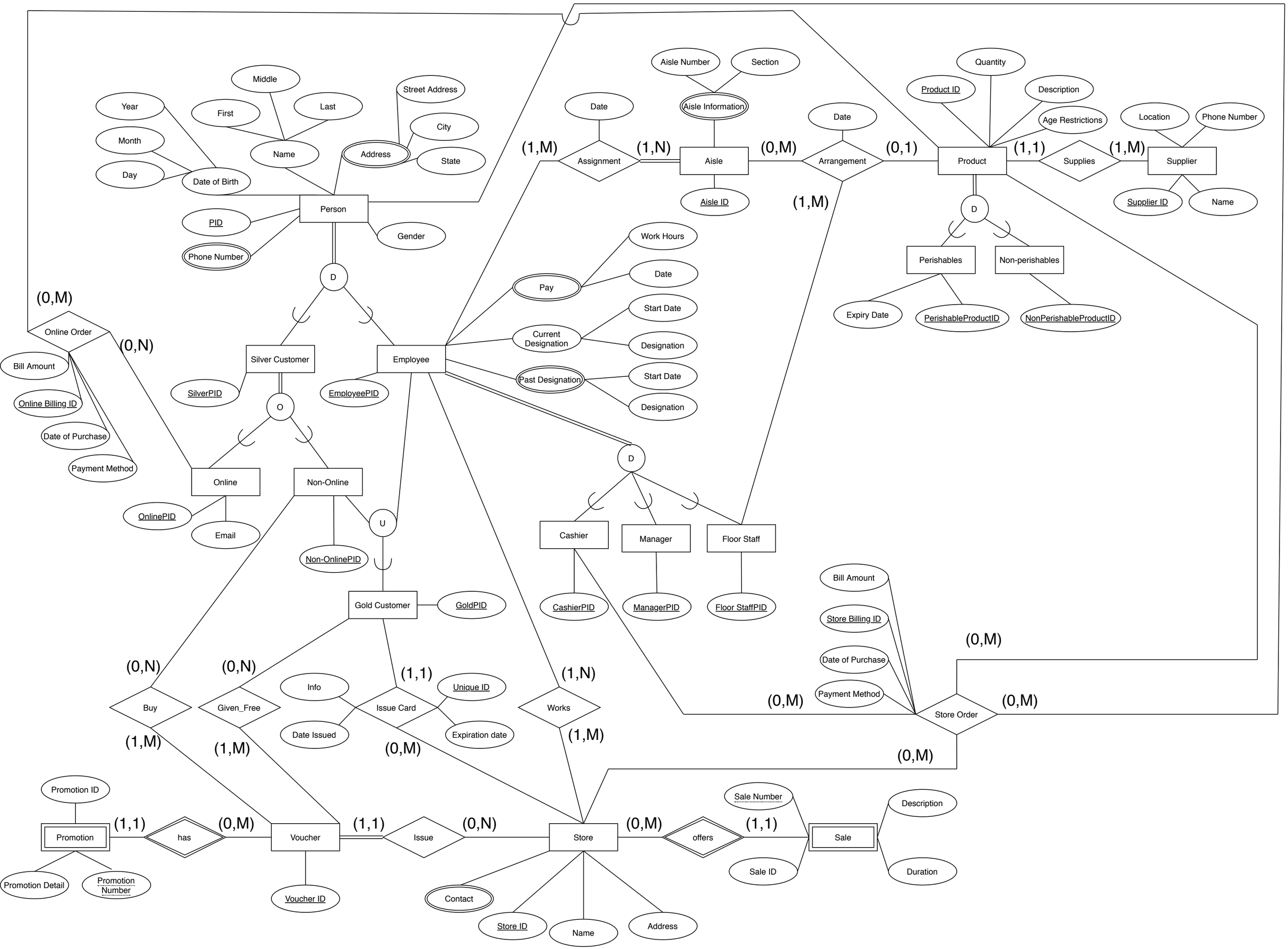
> All Gold members may not get a voucher
> All Gold customer must get a card from store
> Other information for Card for Gold customer is the expiration date
> All stores may not issue card to Gold customer
> All voucher is given to gold members for free
> Gold customer is issued a card from one store
> Each Gold customers are given multiple vouchers
> Each Voucher is given to multiple gold customer
> Each Non-online customer can buy multiple vouchers
> Each Voucher can be bought by multiple Non-online customer

> All product always has a supplier
> All supplier supplies at least one product
> Each Product can be in only one aisle
> Aisle can have multiple products
> A product may not be in an aisle

> All product may not be in aisle
> All products are billable

> Other information for Product is age restriction

> All aisle may not have product
> All employee must have an assignment
> All aisle may not be used for assignment


Middle
Year
First
Last
Street Address
City
Month
Name
Address
State
Day
Date of Birth
Person
PID
Gender
Phone Number
D
(0,M)
Online Order
(0,N)
Bill Amount
Online Billing ID
Date of Purchase
Payment Method
Silver Customer
Employee
SilverPID
EmployeePID
O
Online
Non-Online
OnlinePID
Email
U
Non-OnlinePID
Gold Customer
GoldPID
Aisle Number
Section
Aisle Information
Date
(1,M)
Assignment
(1,N)
Aisle
(0,M)
Arrangement
(0,1)
Aisle ID
(1,M)
Quantity
Product ID
Description
Age Restrictions
Location
Phone Number
Product
(1,1)
Supplies
(1,M)
Supplier
Supplier ID
Name
D
Perishables
Non-perishables
Expiry Date
PerishableProductID
NonPerishableProductID
Work Hours
Pay
Date
Start Date
Current Designation
Designation
Past Designation
Start Date
Designation
D
Cashier
Manager
Floor Staff
CashierPID
ManagerPID
Floor StaffPID
(0,N)
Buy
(0,N)
Given_Free
Info
(1,1)
Issue Card
Unique ID
Date Issued
Expiration date
(1,N)
Works
(1,M)
(0,M)
(1,M)
(1,M)
(0,M)
Bill Amount
Store Billing ID
Date of Purchase
Payment Method
(0,M)
Store Order
(0,M)
(0,M)
Promotion ID
Promotion
(1,1)
has
(0,M)
Voucher
(1,1)
Issue
(0,N)
Store
(0,M)
offers
(1,1)
Sale Number
Sale
Description
Sale ID
Duration
Promotion Detail
Promotion Number
Voucher ID
Contact
Store ID
Name
Address

d. Relational Schema after Normalization

SUPPLY_DATE

ProductID	SupplierID	SupplyDate

PRSN_ADDRESS

PID	AdState	AdCity	AdStreetAddr

SUPPLIER

SupplierID	SupplierName	SupplierLocation	SupplierPhoneNo

PHONE_NUMBER

PID	PhoneNo

ONLINE_ORDER

OnlinePID	ProductID	OnlineBillID	BillAmount	DatePurchase	PaymentMethod

PRODUCT

ProductID	ProductQuantity	ProductDescription	ProductAgeRestriction	ProductSupplierID

PERSON

PID	DOB	Fname	Mname	Lname	Gender

PERI_PRODUCT

PerishableProductID	ExpiryDate

NON_PERI_PRODUCT

NonPerishableProductID

ARRANGEMENT

ProductID	FloorStaffPID	AisleID	ArrangementDate

SILVER_CUST

SilverPID	DateOfJoin

AISLE_INFO

AisleID	AisleNumber	Section

ONLINE_CUST

OnlinePID	Email

AISLE

AisleID

BUY

NonOnlinePID	VoucherID	VoucherIssueStoreID	DatePurchase

NON_ONLINE_CUST

NonOnlinePID	GoldPID	CardStoreID	CardUniqID

GOLD_CUST

GoldPID	CardStoreID	CardUniqID	CardInfo	CardIssueDate	CardExpiraryDate

ASSIGNMENT

EmployeePID	AisleID	StoreID	AssignmentDate

VOUCHER

VoucherID	IssueStoreID

EMPLOYEE

EmployeePID	CurrStartDate	CurrDesignation	GoldPID	CardStoreID	CardUniqID

PROMOTION

PromotionNum	VoucherID	IssueStoreID	PromotionID	PromotionDetail

GIVEN_FREE

GoldPID	CardStoreID	CardUniqID	VoucherID	VoucherIssueStoreID

PAY

EmployeePID	WorkHour	PayDate

SALE

SaleNum	StoreID	SaleID	SaleDescription	Duration

PAST_DESIGNATION

EmployeePID	StartDate	Designation

STORE

StoreID	StoreName	StoreState	StoreCity	StoreStreet

WORKS

EmployeePID	StoreID

CASHIER

CashierPID

STORE_CONTACT

StoreID	StorePhoneNo

MANAGER

ManagerPID

FLOOR_STAFF

FloorStaffPID

STORE_ORDER

CashierPID	StoreID	ProductID	PID	StoreBillID	BillAmount	DatePurchase	PaymentMethod

e. All requested SQL statements

1. **List the details of all the managers of the store in the past two months.**

```
SELECT DISTINCT P.PID, P.Fname, P.Mname, P.Lname, P.Gender, PH.PhoneNo, PA.AdStreet, PA.AdCity,
PA.AdState, A.StoreID, S.StoreName
FROM PERSON AS P, PHONE_NUMBER AS PH, PRSN_ADDRESS AS PA, MANAGER AS M,
ASSIGNMENT AS A, STORE AS S
WHERE PH.PID=P.PID AND PA.PID=P.PID AND P.PID = M.ManagerPID AND M.ManagerPID =
A.EmployeePID AND S.StoreID = A.StoreID
AND A.AssignmentDate >= DATEADD(MONTH, DATEDIFF(MONTH, 0, GETDATE()) - 2, 0)
```

	PID	Fname	Mname	Lname	Gender	PhoneNo	AdStreet	AdCity	AdState	StoreID	StoreName
1	P011	Abdul	K	Kalam	Male	8947789092	LAMB 500	KOLKATA	IL	123	BIGBAZAR

2. **List customers who have bought all perishable items available in the store.**

```
SELECT DISTINCT P.PID, P.Fname, P.Mname, P.Lname, P.Gender, PH.PhoneNo, PA.AdStreet, PA.AdCity,
PA.AdState
FROM PERSON AS P, PHONE_NUMBER AS PH, PRSN_ADDRESS AS PA
WHERE PH.PID=P.PID AND PA.PID=P.PID
AND NOT EXISTS ((
        SELECT DISTINCT AR.ProductID
        FROM ASSIGNMENT AS ASS, FLOOR_STAFF AS F,
        ARRANGEMENT AS AR, PERI_PRODUCT AS P
        WHERE ASS.EmployeePID = F.FloorStaffPID AND
        F.FloorStaffPID = AR.FloorStaffPID AND
        AR.ProductID = P.PerishableProductID
        GROUP BY ASS.StoreID, AR.ProductID
        )
        EXCEPT
        (
        SELECT DISTINCT SO.ProductID
        FROM STORE_ORDER AS SO, PERI_PRODUCT AS PE
        WHERE SO.ProductID = PE.PerishableProductID AND SO.PID = P.PID
        GROUP BY SO.StoreID, SO.ProductID
        ))
GO
```

	PID	Fname	Mname	Lname	Gender	PhoneNo	AdStreet	AdCity	AdState
1	P004	Erin	S	Leinenbach	Female	2345046004	HWY 500	CHICAGO	IL

3. **Find the average number of purchases made by the top five Gold Customers.**

```
SELECT AVG(NumOfPurchase) AS AvgPurchase
FROM
(
   SELECT P1.PID, COUNT(*) AS NumOfPurchase
   FROM PERSON AS P1, STORE_ORDER as SO
```

```
   WHERE P1.PID IN
      (
         SELECT TOP 5 P.PID AS PID1
         FROM PERSON AS P, GOLD_CUST AS G
         WHERE P.PID = G.GoldPID AND
         (SELECT COUNT(*)
         FROM STORE_ORDER as SO
         WHERE G.GoldPID = SO.PID AND
         SO.DatePurchase >= DATEADD(YEAR, DATEDIFF(YEAR, 0, GETDATE()) - 1, 0)
         ) >= 12
         GROUP BY P.PID
      )
   AND P1.PID = SO.PID
   GROUP BY P1.PID
) AS Counts
GO
```

	AvgPurchase
1	20

4. **Find the expiry date of the perishable item that is purchased the most.**

```
SELECT TOP 1 PE.ExpiraryDate, COUNT(*)
FROM STORE_ORDER AS SO, PERI_PRODUCT AS PE
WHERE SO.ProductID = PE.PerishableProductID
GROUP BY PE.PerishableProductID, PE.ExpiraryDate
ORDER BY COUNT(*) DESC
```

	ExpiraryDate	PeriItemCount	PerishableProductID
1	2020-03-17	11	452

5. **Find the supplier details of products that are out of stock.**

```
SELECT S.SupplierID, S.SupplierName, S.SupplierLocation, S.SupplierPhoneNo
FROM PRODUCT AS P, SUPPLIER AS S
WHERE P.ProductQuantity = 0 AND P.ProductSupplierID = S.SupplierID
```

	SupplierID	SupplierName	SupplierLocation	SupplierPhoneNo
1	41	VIMAL	KOLKATA	5647821287

6. **Find the total number transactions made at each store.**

```
SELECT COUNT(*) AS TotalTrans, S.StoreID, S.StoreName
FROM STORE AS S, STORE_ORDER AS SO
WHERE S.StoreID = SO.StoreID
GROUP BY S.StoreID, S.StoreName
ORDER BY COUNT(*) DESC
```

	TotalTrans	StoreID	StoreName
1	13	123	BIGBAZAR
2	12	126	KROGER
3	5	125	WALMART

7**. Find the employee details who has worked every day of the past week.**

```
SELECT DISTINCT PAY.EmployeePID, P.Fname, P.Mname, P.Lname, P.Gender, PH.PhoneNo, PA.AdStreet,
PA.AdCity, PA.AdState
FROM PERSON AS P, PHONE_NUMBER AS PH,
PRSN_ADDRESS AS PA, EMPLOYEE AS E, PAY AS PAY
WHERE PH.PID=P.PID AND
PA.PID=P.PID AND
E.EmployeePID=P.PID AND
PAY.EmployeePID = E.EmployeePID AND
E.EmployeePID=P.PID AND
PAY.PayDate >= DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()) - 1, 0)
GROUP BY PAY.EmployeePID, P.Fname, P.Mname, P.Lname, P.Gender, PH.PhoneNo, PA.AdStreet,
PA.AdCity, PA.AdState
HAVING COUNT(*) = 7
```

	EmployeePID	Fname	Mname	Lname	Gender	PhoneNo	AdStreet	AdCity	AdState
1	P010	Minal	w	Bonde	Female	2345043504	MARY ROAD	CHICAGO	NM

8. **Find the count of customers who have bought the most popular product.**

```
SELECT COUNT(NumCust) AS CustBuyPopProd
FROM
(
   SELECT SO.PID, COUNT(*) AS NumCust
   FROM STORE_ORDER AS SO, TOP_POPULAR_PRODUCT AS TP
   WHERE SO.ProductID = TP.ProductID
   GROUP BY SO.PID
) CustList
```

	CustBuyPopProd
1	3

9. **List all transaction details issued after the most current employee was hired.**

```
SELECT SO.CashierPID, SO.StoreID, SO.ProductID, SO.PID, SO.StoreBillId, SO.BillAmount,
SO.DatePurchase, SO.PaymentMethod
FROM STORE_ORDER AS SO
WHERE SO.DatePurchase >= (
                    SELECT TOP 1 E.CurrStartDate
```

```
    FROM EMPLOYEE AS E
    ORDER BY E.CurrStartDate DESC
)
```

	CashierPID	StoreID	ProductID	PID	StoreBillId	BillAmount	DatePurchase	PaymentMethod
2	P010	123	454	P001	12	345	2019-04-10	Master
3	P010	123	454	P001	16	345	2019-04-10	Master
4	P010	123	454	P001	19	345	2019-04-10	Master
5	P010	123	454	P001	21	345	2019-04-10	Master
6	P010	123	454	P001	23	345	2019-04-10	Master
7	P010	123	454	P001	25	345	2019-04-10	Master
8	P010	123	455	P002	9	435	2019-03-11	Visa
9	P010	126	451	P004	27	12	2019-08-08	Visa
10	P010	126	452	P001	13	234	2019-05-08	Visa
11	P010	126	452	P001	17	234	2019-05-08	Visa

10. **List all the employees that have enrolled as Gold Customer within a month of being employed.**

```
SELECT DISTINCT P.Fname, P.Mname, P.Lname, P.Gender, PH.PhoneNo, PA.AdStreet, PA.AdCity,
PA.AdState
FROM PERSON AS P, PHONE_NUMBER AS PH, PRSN_ADDRESS AS PA, EMPLOYEE AS E,
GOLD_CUST AS G
WHERE PH.PID=P.PID AND PA.PID=P.PID AND E.EmployeePID=P.PID AND
E.GoldPID=G.GoldPID AND
G.CardIssueDate BETWEEN E.CurrStartDate AND DATEADD(MONTH, 1, E.CurrStartDate)
```

	Fname	Mname	Lname	Gender	PhoneNo	AdStreet	AdCity	AdSta…
1	Abdul	K	Kalam	Male	8947789092	LAMB 500	KOLKATA	IL

11. **Find the details of the voucher that are purchased the most.**

```
SELECT V1.VoucherID
FROM VOUCHER AS V1
WHERE V1.VoucherID IN (
            SELECT TOP 1 V.VoucherID
            FROM VOUCHER AS V, BUY
            WHERE V.VoucherID = BUY.VoucherID
            GROUP BY V.VoucherID
            ORDER BY COUNT(*) DESC
            )
GROUP BY V1.VoucherID
GO
```

	VoucherID
1	1018

12. **Find customers who have been Silver Customer for over 5 years.**

```
SELECT DISTINCT S.SilverPID, P.Fname, P.Mname, P.Lname, P.Gender, PH.PhoneNo, PA.AdStreet,
PA.AdCity, PA.AdState
FROM SILVER_CUST AS S, PERSON AS P, PHONE_NUMBER AS PH, PRSN_ADDRESS AS PA
WHERE PH.PID=P.PID AND PA.PID=P.PID and P.PID=S.SilverPID AND
S.DateOfJoin >= DATEADD(YEAR, DATEDIFF(YEAR, 0, GETDATE()) - 5, 0)
GO
```

	SilverPID	Fname	Mname	Lname	Gender	PhoneNo	AdStreet	AdCity	AdState
1	P001	Kunal		Mukherjee	Male	8125503890	COWBOYS PKWY	IRVING	TX
2	P002	Suraj	N	Kothawade	Male	8123546797	BAKER STREET	EVANSVILLE	IL

13. **Find the number of purchases made by the potential Gold Members in the last year.**

```
SELECT PG.PID, COUNT(*) AS NumOfPurchase
FROM POTENTIAL_GOLD_CUST AS PG, STORE_ORDER AS SO
WHERE PG.PID = SO.PID AND
SO.DatePurchase >= DATEADD(YEAR, DATEDIFF(YEAR, 0, GETDATE()) - 1, 0)
GROUP BY PG.PID
```

	PID	NumOfPurchase
1	P002	1

14. **Find the maximum bill amount and details of the store that has the maximum number of purchases in the last year.**

```
SELECT MAX(SO.BillAmount) AS MaxBillAmount, ST.StoreID, ST.StoreName, ST.StoreState,
ST.StoreCity, ST.StoreStreet
FROM STORE AS ST, STORE_ORDER AS SO
WHERE ST.StoreID IN (
                SELECT TOP 1 SO.StoreID
                FROM STORE AS S, STORE_ORDER AS SO
                WHERE S.StoreID = SO.StoreID
                GROUP BY SO.StoreID
                ORDER BY COUNT(*) DESC
            )
GROUP BY ST.StoreID, ST.StoreName, ST.StoreState, ST.StoreCity, ST.StoreStreet
GO
```

	MaxBillAmount	StoreID	StoreName	StoreState	StoreCity	StoreStreet
1	798	123	BIGBAZAR	IL	EVANVILLE	BAKER ST

15**. Find the date of the transaction that has a bill amount greater than the average bill amount of all transactions in the system.**

```
SELECT SO.DatePurchase, SO.BillAmount
FROM STORE_ORDER AS SO
WHERE SO.BillAmount >= (
                SELECT AVG(SO.BillAmount)
                FROM STORE_ORDER AS SO
                )
GO
```

	DatePurchase	BillAmou…
1	2019-04-15	798
2	2010-04-10	345
3	2019-04-10	345
4	2019-04-10	345
5	2019-04-10	345
6	2019-04-10	345
7	2019-04-10	345
8	2019-04-10	345
9	2010-03-11	435
…	2019-03-11	435
…	2019-03-08	443

VIEWS:

1. Top Gold Customer- This view returns the First Name, Last Name and Date of membership enrollment of those gold customers who have transactions more than 12 times in the past year.

```
CREATE VIEW TOP_GOLD_CUST(Fname, Lname, DOMemEnrol) AS
SELECT DISTINCT TOP 1  P.Fname, P.Lname, G.CardExpiraryDate
FROM PERSON AS P, GOLD_CUST AS G
WHERE P.PID = G.GoldPID AND
(SELECT COUNT(*)
FROM STORE_ORDER as SO
WHERE G.GoldPID = SO.PID AND
SO.DatePurchase >= DATEADD(YEAR, DATEDIFF(YEAR, 0, GETDATE()) - 1, 0)
) >= 12
GO
```

2. Popular Product- This view returns the details of the product that customers have purchased the most in the past 2 years.

```
CREATE VIEW TOP_POPULAR_PRODUCT (NumofItems, ProductID, ProductQuantity,
ProductDescription, ProductAgeRestriction, ProductSupplierID) AS
SELECT TOP 1 COUNT(*) AS NumofItems, P.ProductID, P.ProductQuantity, P.ProductDescription,
P.ProductAgeRestriction, P.ProductSupplierID
FROM PRODUCT AS P,  STORE_ORDER AS SO
WHERE P.ProductID = SO.ProductID AND SO.DatePurchase >= DATEADD(YEAR, DATEDIFF(YEAR, 0,
GETDATE()) - 2, 0)
GROUP BY P.ProductID, P.ProductQuantity, P.ProductDescription, P.ProductAgeRestriction,
P.ProductSupplierID
ORDER BY COUNT(*) DESC
GO
```

3. Top Store- This view returns the details of the store that has maximum number of purchases in the last year.

```
CREATE VIEW TOP_STORE (NumofItems, StoreID, StoreName, StoreStreet, StoreCity, StoreState) AS
SELECT TOP 1 COUNT(*) AS NumofItems, S.StoreID, S.StoreName, S.StoreStreet, S.StoreCity, S.StoreState
FROM STORE AS S,  STORE_ORDER AS SO
WHERE S.StoreID = SO.StoreID AND SO.DatePurchase >= DATEADD(YEAR, DATEDIFF(YEAR, 0,
GETDATE()) - 1, 0)
GROUP BY S.StoreID, S.StoreName, S.StoreStreet, S.StoreCity, S.StoreState
ORDER BY COUNT(*) DESC
GO
```

4. Potential Gold Customers- This view returns the name, phone number and ID of the Silver Customers who bought more than 10 vouchers in the last month.

```
CREATE VIEW POTENTIAL_GOLD_CUST (Fname, Lname, PhoneNo, PID) AS
SELECT DISTINCT P.Fname, P.Lname, PO.PhoneNo, P.PID
FROM PERSON AS P, PHONE_NUMBER AS PO, SILVER_CUST AS S, NON_ONLINE_CUST AS NC
WHERE PO.PID = P.PID AND P.PID = S.SilverPID AND S.SilverPID = NC.NonOnlinePID
AND
(SELECT COUNT(*)
FROM BUY as B
WHERE B.NonOnlinePID = NC.NonOnlinePID AND
B.DatePurchase >= DATEADD(MONTH, DATEDIFF(MONTH, 0, GETDATE()) - 1, 0)
) >= 10
GO
```

5. Top Supplier - This view returns the details of the supplier who has made the most number of supplies of perishable items in the past month.

```
CREATE VIEW TOP_SUPPLIER (NumofItems, SupplierID, SupplierName, SupplierLocation,
SupplierPhoneNo) AS
SELECT TOP 1 COUNT(*) AS NumofItems, S.SupplierID, S.SupplierName, S.SupplierLocation,
S.SupplierPhoneNo
FROM SUPPLIER AS S, SUPPLY_DATE AS SD
WHERE S.SupplierID = SD.SupplierID AND SD.SupplyDate >= DATEADD(MONTH,
DATEDIFF(MONTH, 0, GETDATE()) - 1, 0)
```

```
GROUP BY S.SupplierID, S.SupplierName, S.SupplierLocation, S.SupplierPhoneNo
ORDER BY COUNT(*) DESC
GO
```

TABLE CREATION

```
-- 1>Create the Person Table
CREATE TABLE PERSON
(
   PID CHAR(4) NOT NULL,
   DOB DATE NOT NULL,
   Fname VARCHAR(15) NOT NULL,
   Mname CHAR DEFAULT NULL,
   Lname VARCHAR(15) NOT NULL,
   Gender VARCHAR(15) NOT NULL DEFAULT 'Unknown',
   CONSTRAINT PER_PPK
      PRIMARY KEY(PID),
   CONSTRAINT CHK_Age
      CHECK (YEAR(GETDATE()) - YEAR(DOB) > 16),
   CONSTRAINT CHK_GNDR
      CHECK(Gender in ('Male', 'Female', 'Unknown')),
   CONSTRAINT CHK_PID
      CHECK (PID LIKE 'P[0-9][0-9][0-9]')
);
GO

-- 2>Create the Phone Number Table
CREATE TABLE PHONE_NUMBER
(
   PID CHAR(4) NOT NULL,
   PhoneNo CHAR(10) NOT NULL,
   CONSTRAINT PHO_PPK
      PRIMARY KEY(PID, PhoneNo),
   CONSTRAINT PHO_FRK
      FOREIGN KEY (PID) REFERENCES PERSON(PID)
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT CHK_PhoneNo
      CHECK (PhoneNo LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')
);
GO

-- 3>Create the Address Table
CREATE TABLE PRSN_ADDRESS
(
   PID CHAR(4) NOT NULL,
   AdState CHAR(2) NOT NULL,
   AdCity VARCHAR(15) NOT NULL,
   AdStreet VARCHAR(15) NOT NULL,
   CONSTRAINT ADDR_PPK
      PRIMARY KEY(PID, AdState, AdCity, AdStreet),
   CONSTRAINT ADDR_FRK
      FOREIGN KEY (PID) REFERENCES PERSON(PID)
```

```
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 4>Create the Silver Customer Table
CREATE TABLE SILVER_CUST
(
   SilverPID CHAR(4) NOT NULL,
   DateOfJoin DATE NOT NULL,
   CONSTRAINT SILVER_CUST_PPK
      PRIMARY KEY(SilverPID),
   CONSTRAINT SILVER_CUST_FRK
      FOREIGN KEY (SilverPID) REFERENCES PERSON(PID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 5>Create the Onile Customer Table
CREATE TABLE ONLINE_CUST
(
   OnlinePID CHAR(4) NOT NULL,
   Email VARCHAR(15) NOT NULL,
   CONSTRAINT ONLINE_CUST_PPK
      PRIMARY KEY(OnlinePID),
   CONSTRAINT ONLINE_CUST_FRK
      FOREIGN KEY (OnlinePID) REFERENCES SILVER_CUST(SilverPID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 6>Create the Store Table
CREATE TABLE STORE
(
   StoreID INT NOT NULL,
   StoreName VARCHAR(10) NOT NULL,
   StoreState CHAR(2) NOT NULL,
   StoreCity VARCHAR(15) NOT NULL,
   StoreStreet VARCHAR(15) NOT NULL,
   CONSTRAINT STORE_PPK
      PRIMARY KEY(StoreID)
);
GO

-- 7>Create the StoreContact Table
CREATE TABLE STORE_CONTACT
(
   StoreID INT NOT NULL,
   StorePhoneNo CHAR(10) NOT NULL,
   CONSTRAINT STORE_CONTACT_PPK
      PRIMARY KEY(StoreID),
   CONSTRAINT STORE_CONTACT_FRK
      FOREIGN KEY (StoreID) REFERENCES STORE(StoreID)
```

```
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT StorePhoneNo
      CHECK (StorePhoneNo LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')
);
GO

-- 8>Create the Sale Table
CREATE TABLE SALE
(
   SaleNum INT NOT NULL,
   StoreID INT NOT NULL,
   SaleID CHAR(10) NOT NULL,
   SaleDescription CHAR(15) NOT NULL,
   Duration DATETIME NOT NULL,
   CONSTRAINT SALE_PPK
      PRIMARY KEY(SaleNum, StoreID),
   CONSTRAINT SALE_FRK
      FOREIGN KEY (StoreID) REFERENCES STORE(StoreID)
      ON DELETE CASCADE ON UPDATE CASCADE,
);
GO

-- 9>Create the Voucher Table
CREATE TABLE VOUCHER
(
   VoucherID INT NOT NULL,
   IssueStoreID INT NOT NULL,
   CONSTRAINT VOUCHER_PPK
      PRIMARY KEY(VoucherID, IssueStoreID),
   CONSTRAINT VOCUHER_FRK
      FOREIGN KEY (IssueStoreID) REFERENCES STORE_CONTACT(StoreID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 10>Create the Promotion Table
CREATE TABLE PROMOTION
(
   PromotionNum INT NOT NULL,
   VoucherID INT NOT NULL,
   IssueStoreID INT NOT NULL,
   PromotionID INT NOT NULL,
   PromotionDetail VARCHAR(30) NOT NULL,
   CONSTRAINT PROMOTION_PPK
      PRIMARY KEY (PromotionNum, VoucherID, IssueStoreID),
   CONSTRAINT PROMOTION_FRK
      FOREIGN KEY (VoucherID, IssueStoreID) REFERENCES VOUCHER(VoucherID, IssueStoreID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 11>Create the Gold Customer Table
```

```
CREATE TABLE GOLD_CUST
(
   GoldPID CHAR(4) NOT NULL,
   CardStoreID INT NOT NULL,
   CardUniqID INT NOT NULL UNIQUE,
   CardInfo VARCHAR(15),
   CardIssueDate DATE NOT NULL,
   CardExpiraryDate DATE NOT NULL,
   CONSTRAINT GOLD_CUST_PPK
      PRIMARY KEY(GoldPID, CardStoreID, CardUniqID),
   CONSTRAINT SILVER_CUST_FRK1
      FOREIGN KEY (GoldPID) REFERENCES PERSON(PID)
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT GOLD_CUST_FRK2
      FOREIGN KEY (CardStoreID) REFERENCES STORE(StoreID)
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT CHK_DATE
      CHECK( CardIssueDate < CardExpiraryDate)
);
GO

-- 12>Create the Non-Online Customer Table
CREATE TABLE NON_ONLINE_CUST
(
   NonOnlinePID CHAR(4) NOT NULL,
   GoldPID CHAR(4),
   CardStoreID INT,
   CardUniqID INT,
   CONSTRAINT NON_ONLINE_CUST_PPK
      PRIMARY KEY(NonOnlinePID),
   CONSTRAINT NON_ONLINE_CUST_FRK1
      FOREIGN KEY (NonOnlinePID) REFERENCES SILVER_CUST(SilverPID)
     ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT NON_ONLINE_CUST_FRK2
      FOREIGN KEY (GoldPID, CardStoreID, CardUniqID) REFERENCES GOLD_CUST(GoldPID,
CardStoreID, CardUniqID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT CHK_NON_ONLINE_CUST_PID
      CHECK (NonOnlinePID = GoldPID)
);
GO

-- 13>Create the Employee Table
CREATE TABLE EMPLOYEE
(
   EmployeePID CHAR(4) NOT NULL,
   CurrStartDate DATE NOT NULL,
   CurrDesignation VARCHAR(15) NOT NULL,
   GoldPID CHAR(4),
   CardStoreID INT,
   CardUniqID INT,
   CONSTRAINT EMPLOYEE_PPK
```

```
      PRIMARY KEY(EmployeePID),
   CONSTRAINT EMPLOYEE_FRK1
      FOREIGN KEY (EmployeePID) REFERENCES PERSON (PID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT EMPLOYEE_FRK2
      FOREIGN KEY (GoldPID, CardStoreID, CardUniqID) REFERENCES GOLD_CUST(GoldPID,
CardStoreID, CardUniqID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT CHK_EMPLOYEE_PID
      CHECK (EmployeePID = GoldPID),
   CONSTRAINT CHK_EMPLOYEE_DESIGN
      CHECK (CurrDesignation In ('Cashier', 'Manager', 'FloorStaff'))
);
GO

-- 14>Create the Pay Table
CREATE TABLE PAY
(
   EmployeePID CHAR(4) NOT NULL,
   WorkHour INT NOT NULL,
   PayDate DATE NOT NULL,
   CONSTRAINT PAY_PPK
      PRIMARY KEY(EmployeePID, WorkHour, PayDate),
   CONSTRAINT PAY_FRK
      FOREIGN KEY (EmployeePID) REFERENCES EMPLOYEE (EmployeePID)
      ON DELETE CASCADE ON UPDATE CASCADE,
);
GO

-- 15>Create the Past Designation Table
CREATE TABLE PAST_DESIGNATION
(
   EmployeePID CHAR(4) NOT NULL,
   StartDate DATE NOT NULL,
   Designation VARCHAR(15) NOT NULL,
   CONSTRAINT PAST_DESIGNATION_PPK
      PRIMARY KEY(EmployeePID, StartDate, Designation),
   CONSTRAINT PAST_DESIGNATION_FRK
      FOREIGN KEY (EmployeePID) REFERENCES EMPLOYEE (EmployeePID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 16>Create the Cashier Table
CREATE TABLE CASHIER
(
   CashierPID CHAR(4) NOT NULL,
   CONSTRAINT CASHIER_PPK
      PRIMARY KEY(CashierPID),
   CONSTRAINT CASHIER_FRK
      FOREIGN KEY (CashierPID) REFERENCES EMPLOYEE (EmployeePID)
      ON DELETE CASCADE ON UPDATE CASCADE
```

```
);
GO
-- 17>Create the Manager Table
CREATE TABLE MANAGER
(
   ManagerPID CHAR(4) NOT NULL,
   CONSTRAINT MANAGER_PPK
      PRIMARY KEY(ManagerPID),
   CONSTRAINT MANAGER_FRK
      FOREIGN KEY (ManagerPID) REFERENCES EMPLOYEE (EmployeePID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO
-- 18>Create the Floor Staff Table
CREATE TABLE FLOOR_STAFF
(
   FloorStaffPID CHAR(4) NOT NULL,
   CONSTRAINT FLOOR_STAFF_PPK
      PRIMARY KEY(FloorStaffPID),
   CONSTRAINT FLOOR_STAFF_FRK
      FOREIGN KEY (FloorStaffPID) REFERENCES EMPLOYEE (EmployeePID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 19>Create the Aisle Table
CREATE TABLE AISLE
(
   AisleID INT NOT NULL,
   CONSTRAINT AISLE_PPK
      PRIMARY KEY(AisleID)
);
GO

-- 20>Create the Aisle Infor Table
CREATE TABLE AISLE_INFO
(
   AisleID INT NOT NULL,
   AisleNumber INT NOT NULL,
   Section INT NOT NULL,
   CONSTRAINT AISLE_INFO_PPK
      PRIMARY KEY(AisleID, AisleNumber, Section),
   CONSTRAINT AISLE_INFO_FRK
      FOREIGN KEY (AisleID) REFERENCES AISLE (AisleID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 21>Create the Supplier Table
CREATE TABLE SUPPLIER
(
   SupplierID INT NOT NULL,
```

```
   SupplierName VARCHAR(30) NOT NULL,
   SupplierLocation VARCHAR(30) NOT NULL,
   SupplierPhoneNo CHAR(10) NOT NULL,
   CONSTRAINT SUPPLIER_PPK
      PRIMARY KEY(SupplierID),
    CONSTRAINT CHK_SUPPLIER_PhoneNo
      CHECK (SupplierPhoneNo LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')
);
GO

-- 22>Create the Product Table
CREATE TABLE PRODUCT
(
   ProductID INT NOT NULL,
   ProductQuantity INT NOT NULL,
   ProductDescription VARCHAR(30) NOT NULL,
   ProductAgeRestriction VARCHAR(3),
   ProductSupplierID INT NOT NULL,
   CONSTRAINT PRODUCT_PPK
      PRIMARY KEY(ProductID),
   CONSTRAINT PRODUCT_FRK
      FOREIGN KEY (ProductSupplierID) REFERENCES SUPPLIER (SupplierID)
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT CHK_AGE_RES
      CHECK(ProductAgeRestriction in ('Yes', 'No', 'Unknown')),
);
GO

-- 23>Create the Perishables Table
CREATE TABLE PERI_PRODUCT
(
   PerishableProductID INT NOT NULL,
   ExpiraryDate DATE NOT NULL,
   CONSTRAINT PERI_PRODUCT_PPK
      PRIMARY KEY(PerishableProductID),
   CONSTRAINT PERI_PRODUCT_FRK
      FOREIGN KEY (PerishableProductID) REFERENCES PRODUCT (ProductID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 24>Create the Non-Perishables Table
CREATE TABLE NON_PERI_PRODUCT
(
   NonPerishableProductID INT NOT NULL,
   CONSTRAINT NON_PERI_PRODUCT_PPK
      PRIMARY KEY(NonPerishableProductID),
   CONSTRAINT NON_PERI_PRODUCT_FRK
      FOREIGN KEY (NonPerishableProductID) REFERENCES PRODUCT (ProductID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO
```

```
-- 25>Create the Suply Date
CREATE TABLE SUPPLY_DATE
(
   ProductID INT NOT NULL,
   SupplierID INT NOT NULL,
   SupplyDate DATE NOT NULL,
   CONSTRAINT SUPPLY_DATE_PPK
      PRIMARY KEY(ProductID, SupplierID, SupplyDate),
   CONSTRAINT SUPPLY_DATE_FRK1
      FOREIGN KEY (ProductID) REFERENCES PRODUCT (ProductID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT SUPPLY_DATE_FRK2
      FOREIGN KEY (SupplierID) REFERENCES SUPPLIER (SupplierID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
);
GO

-- 26>Create the Arrangement Table
CREATE TABLE ARRANGEMENT
(
   ProductID INT NOT NULL,
   FloorStaffPID CHAR(4) NOT NULL,
   AisleID INT NOT NULL,
   ArrangementDate DATE NOT NULL,
   CONSTRAINT ARRANGEMENT_PPK
      PRIMARY KEY(ProductID, FloorStaffPID, AisleID),
   CONSTRAINT ARRANGEMENT_FRK1
      FOREIGN KEY (ProductID) REFERENCES PRODUCT (ProductID)
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT ARRANGEMENT_FRK2
      FOREIGN KEY (FloorStaffPID) REFERENCES FLOOR_STAFF (FloorStaffPID)
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT ARRANGEMENT_FRK3
      FOREIGN KEY (AisleID) REFERENCES AISLE (AisleID)
      ON DELETE CASCADE ON UPDATE CASCADE,
);
GO

-- 27>Create the Buy Table
CREATE TABLE BUY
(
   NonOnlinePID CHAR(4) NOT NULL,
   VoucherID INT NOT NULL,
   VoucherIssueStoreID INT NOT NULL,
   DatePurchase DATE NOT NULL
   CONSTRAINT BUY_PPK
      PRIMARY KEY(NonOnlinePID, VoucherID, VoucherIssueStoreID),
   CONSTRAINT BUY_FRK1
      FOREIGN KEY (NonOnlinePID) REFERENCES NON_ONLINE_CUST (NonOnlinePID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT BUY_FRK2
```

```
      FOREIGN KEY (VoucherID, VoucherIssueStoreID) REFERENCES VOUCHER (VoucherID,
IssueStoreID)
      ON DELETE NO ACTION ON UPDATE NO ACTION
);
GO

-- 28>Create the Assignment Table
CREATE TABLE ASSIGNMENT
(
   EmployeePID CHAR(4) NOT NULL,
   AisleID INT NOT NULL,
   StoreID INT NOT NULL,
   AssignmentDate DATE NOT NULL,
   CONSTRAINT ASSIGNMENT_PPK
      PRIMARY KEY(EmployeePID, AisleID, StoreID),
   CONSTRAINT ASSIGNMENT_FRK1
      FOREIGN KEY (EmployeePID) REFERENCES EMPLOYEE (EmployeePID)
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT ASSIGNMENT_FRK2
      FOREIGN KEY (AisleID) REFERENCES AISLE (AisleID)
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT ASSIGNMENT_FRK3
      FOREIGN KEY (StoreID) REFERENCES STORE (StoreID)
      ON DELETE CASCADE ON UPDATE CASCADE
);
GO

-- 29>Create the Given Free Table
CREATE TABLE GIVEN_FREE
(
   GoldPID CHAR(4) NOT NULL,
   CardStoreID INT NOT NULL,
   CardUniqID INT NOT NULL,
   VoucherID INT NOT NULL,
   VoucherIssueStoreID INT NOT NULL,
   CONSTRAINT GIVEN_FREE_PPK
      PRIMARY KEY(GoldPID, CardStoreID, CardUniqID, VoucherID, VoucherIssueStoreID),
   CONSTRAINT GIVEN_FREE_FRK1
      FOREIGN KEY (GoldPID, CardStoreID, CardUniqID) REFERENCES GOLD_CUST(GoldPID,
CardStoreID, CardUniqID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT GIVEN_FREE_FRK2
      FOREIGN KEY (VoucherID, VoucherIssueStoreID) REFERENCES VOUCHER (VoucherID,
IssueStoreID)
      ON DELETE NO ACTION ON UPDATE NO ACTION
);
GO

-- 30>Create the Works Table
CREATE TABLE WORKS
(
   EmployeePID CHAR(4) NOT NULL,
```

```
   StoreID INT NOT NULL,
   CONSTRAINT WORKS_PPK
      PRIMARY KEY(EmployeePID, StoreID),
   CONSTRAINT WORKS_FRK1
      FOREIGN KEY (EmployeePID) REFERENCES EMPLOYEE (EmployeePID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT WORKS_FRK2
      FOREIGN KEY (StoreID) REFERENCES STORE (StoreID)
      ON DELETE NO ACTION ON UPDATE NO ACTION
);
GO

-- 31>Create the Online Order Table
CREATE TABLE ONLINE_ORDER
(
   OnlinePID CHAR(4) NOT NULL,
   ProductID INT NOT NULL,
   OnlineBillID INT NOT NULL UNIQUE,
   BillAmount INT NOT NULL,
   DatePurchase DATE NOT NULL,
   PaymentMethod VARCHAR(10) NOT NULL,
   CONSTRAINT ONLINE_ORDER_PPK
      PRIMARY KEY(OnlinePID, ProductID, OnlineBillID),
   CONSTRAINT ONLINE_ORDER_FRK1
      FOREIGN KEY (OnlinePID) REFERENCES ONLINE_CUST (OnlinePID)
      ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT ONLINE_ORDER_FRK2
      FOREIGN KEY (ProductID) REFERENCES PRODUCT (ProductID)
      ON DELETE CASCADE ON UPDATE CASCADE,

);
GO

-- 32>Create the Store Order Table
CREATE TABLE STORE_ORDER
(
   CashierPID CHAR(4) NOT NULL,
   StoreID INT NOT NULL,
   ProductID INT NOT NULL,
   PID CHAR(4) NOT NULL,
   StoreBillId INT NOT NULL UNIQUE,
   BillAmount INT NOT NULL,
   DatePurchase DATE NOT NULL,
   PaymentMethod VARCHAR(10) NOT NULL,
   CONSTRAINT STORE_ORDER_PPK
      PRIMARY KEY(CashierPID, StoreID, ProductID, PID, StoreBillId),
   CONSTRAINT STORE_ORDER_FRK1
      FOREIGN KEY (CashierPID) REFERENCES CASHIER (CashierPID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT STORE_ORDER_FRK2
      FOREIGN KEY (StoreID) REFERENCES STORE_CONTACT (StoreID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
```

```
   CONSTRAINT STORE_ORDER_FRK3
      FOREIGN KEY(ProductID) REFERENCES PRODUCT (ProductID)
      ON DELETE NO ACTION ON UPDATE NO ACTION,
   CONSTRAINT STORE_ORDER_FRK4
      FOREIGN KEY (PID) REFERENCES PERSON (PID)
      ON DELETE NO ACTION ON UPDATE NO ACTION
);
GO
```

DATA ENTRY:

```
-- Insert data in PERSON TABLE
INSERT INTO PERSON
VALUES
('P001','09-29-1997' ,'Kunal', '', 'Mukherjee', 'Male'),
('P002','06-21-1950' ,'Suraj', 'N', 'Kothawade', 'Male'),
('P003','01-26-1950' ,'Diksha', 'O', 'GodBole', 'Female'),
('P004','08-15-1947' ,'Erin', 'S', 'Leinenbach', 'Female'),
('P005','07-23-1990' ,'Smita', '', 'Ghosh', 'Female'),
('P007','01-01-1990' ,'James', 'F', 'Bond', 'Male'),
('P010','08-09-1997' ,'Minal', 'w', 'Bonde', 'Female'),
('P011','02-17-1950' ,'Abdul', 'K', 'Kalam', 'Male'),
('P012','04-16-1950' ,'Narendra', 'D', 'Modi', 'Male');
GO

-- Insert data in PHONE_NUMBER TABLE
INSERT INTO PHONE_NUMBER
VALUES
('P001', '8125503890'),
('P002', '8123546797'),
('P003', '8820893598'),
('P004', '2345046004'),
('P005', '7564990234'),
('P007', '9034758032'),
('P010', '2345043504'),
('P011', '8947789092'),
('P012', '3749823034');
GO

-- Insert data in PRSN_ADDRESS TABLE
INSERT INTO PRSN_ADDRESS
VALUES
('P001', 'TX', 'IRVING', 'COWBOYS PKWY'),
('P002', 'IL', 'EVANSVILLE', 'BAKER STREET'),
('P003', 'IN', 'PEORIA', 'STRIP RD'),
('P004', 'IL', 'CHICAGO', 'HWY 500'),
('P005', 'NM', 'DALLAS', 'JESSOR ROAD'),
('P007', 'MO', 'KOLKATA', 'SALT LAKE'),
('P010', 'NM', 'CHICAGO', 'MARY ROAD'),
('P011', 'IL', 'KOLKATA', 'LAMB 500'),
('P012', 'MO', 'DALLAS', 'SALT LAKE');
GO
```

```
-- Insert data in SILVER_CUST TABLE
INSERT INTO SILVER_CUST
VALUES
('P001', '10-25-2015'),
('P002', '10-22-2023'),
('P003', '06-29-2020');
GO
UPDATE SILVER_CUST
SET DateOfJoin = '09-04-2010'
WHERE SILVER_CUST.SilverPID='P003'
GO

-- Insert data in ONLINE_CUST TABLE
INSERT INTO ONLINE_CUST
VALUES
('P001', 'KXM6@UTDS.EDU'),
('P002', 'KU@ZAPAK.COM');
GO

-- Insert data in STORE TABLE
INSERT INTO STORE
VALUES
('123', 'BIGBAZAR', 'IL', 'EVANVILLE', 'BAKER ST'),
('124', 'KHOLS', 'TX', 'CHICAGO', 'GULL ST'),
('125', 'WALMART', 'AZ', 'DALLAS', 'FILL ST'),
('126', 'KROGER', 'MI', 'PEORIA', 'TABLE ST'),
('127', 'TARGET', 'FL', 'NOVI', 'MOSS ST');
GO

-- Insert data in STORE_CONTACT TABLE
INSERT INTO STORE_CONTACT
VALUES
('123', '8125503890'),
('124', '8123546797'),
('125', '8820893598'),
('126', '2345046004'),
('127', '7564990234');
GO

-- Insert data in SALE TABLE
INSERT INTO SALE
VALUES
('10', '123', '24', 'FOOD', '09-29-2020'),
('11', '124', '24', 'FOOD', '09-12-2020'),
('12', '125', '25', 'TV', '09-06-2020'),
('13', '125', '26', 'XBOX', '02-29-2020');
GO

-- Insert data in VOUCHER TABLE
INSERT INTO VOUCHER
VALUES
('1009', '125'),
```

```
('1010', '125'),
('1011', '126'),
('1012', '127'),
('1013', '125'),
('1014', '127'),
('1015', '125'),
('1016', '125'),
('1017', '127'),
('1018', '125'),
('1019', '125'),
('1020', '127'),
('1021', '127');
GO

-- Insert data in PROMOTION TABLE
INSERT INTO PROMOTION
VALUES
('510','1009', '125', 1, '10% OFF'),
('511','1010', '125', 1, '20% OFF'),
('512','1012', '127', 1, '30% OFF');
GO

-- Insert data in GOLD_CUST TABLE
INSERT INTO GOLD_CUST
VALUES
('P001','123', '9002', 'FUEL50OFF', '09-06-2010', '05-06-2020'),
('P003','125', '9003', 'PIZZA10OFF', '01-06-2010', '03-04-2020'),
('P007','123', '9000', 'FUEL50OFF', '09-06-2010', '05-06-2020'),
('P005','125', '9001', 'PIZZA10OFF', '01-06-2010', '03-04-2020'),
('P011','127', '9004', 'WATCH10OFF', '05-06-2015', '05-08-2020');
GO
UPDATE GOLD_CUST
SET CardIssueDate = '06-10-2010'
WHERE GoldPID = 'P011'
GO

-- Insert data in NON_ONLINE_CUST TABLE
INSERT INTO NON_ONLINE_CUST
VALUES
('P001', 'P001', '123', '9002')
GO
INSERT INTO NON_ONLINE_CUST ([NonOnlinePID])
VALUES
('P002')
GO

-- Insert data in EMPLOYEE TABLE
INSERT INTO EMPLOYEE
VALUES
('P011','06-07-2010' , 'Manager', 'P011','127', '9004')
GO
INSERT INTO EMPLOYEE ([EmployeePID],[CurrStartDate],[CurrDesignation])
```

```
VALUES
('P010','09-09-2015' , 'Cashier'),
('P012','08-16-2016' ,'FloorStaff')
GO
UPDATE EMPLOYEE
SET CurrStartDate = '03-11-2019'
WHERE EmployeePID = 'P010'
GO

-- Insert data in PAY TABLE
INSERT INTO PAY
VALUES
('P010',15,'06-07-2010'),
('P010',02,'06-08-2010'),
('P010',12,'06-09-2010'),
('P011',34,'07-10-2010'),
('P011',23,'07-09-2010'),
('P011',12,'07-08-2010'),
('P012',11,'06-12-2010'),
('P012',12,'06-13-2010'),
('P012',02,'06-14-2010'),
('P010',12,'04-11-2010'),
('P010',34,'04-12-2010'),
('P010',23,'04-13-2010'),
('P010',12,'04-14-2010'),
('P010',11,'04-15-2010'),
('P010',12,'04-16-2010'),
('P010',02,'04-17-2010'),
('P010',12,'04-11-2020'),
('P010',34,'04-12-2020'),
('P010',23,'04-13-2020'),
('P010',12,'04-14-2020'),
('P010',11,'04-15-2020'),
('P010',12,'04-16-2020'),
('P010',02,'04-17-2020');
GO

-- Insert data in PAST_DESIGNATION TABLE
INSERT INTO PAST_DESIGNATION
VALUES
('P010','07-07-2006','Cashier'),
('P010','01-07-2005','FloorStaff'),
('P011','02-03-2009','FloorStaff');
GO

-- Insert data in CASHIER TABLE
INSERT INTO CASHIER
VALUES
('P010')
GO

-- Insert data in MANAGER TABLE
```

```
INSERT INTO MANAGER
VALUES
('P011')
GO

-- Insert data in FLOOR_STAFF TABLE
INSERT INTO FLOOR_STAFF
VALUES
('P012')
GO

-- Insert data in AISLE TABLE
INSERT INTO AISLE
VALUES
('3800'),
('3801'),
('3802'),
('3803'),
('3804')
GO

-- Insert data in AISLE_INFO TABLE
INSERT INTO AISLE_INFO
VALUES
('3800',1,1),
('3800',1,2),
('3801',1,1),
('3801',1,2),
('3801',1,3)
GO

-- Insert data in SUPPLIER TABLE
INSERT INTO SUPPLIER
VALUES
('41','VIMAL','KOLKATA', '5647821287'),
('42','CAPAK','DERJELING','9038592480'),
('43','DELL','SEOUL', '0192783409'),
('44','HP','MUMBAI', '9082349865'),
('45','BP','DHARAVI', '2359875467')
GO

-- Insert data in PRODUCT TABLE
INSERT INTO PRODUCT
VALUES
('451','12','FOOD', 'YES',  '41'),
('452','34','GAS','NO','44'),
('453','86','FOOD', 'YES','43'),
('454','46','GAS', 'YES','44'),
('455','45','MOVIE','NO' ,'42'),
('456','0','FOOD', 'NO',  '41');
GO
```

```
-- Insert data in PERI_PRODUCT TABLE
INSERT INTO PERI_PRODUCT
VALUES
('451', '02-27-2020'),
('452', '03-17-2020'),
('453','05-27-2026')
GO

-- Insert data in NON_PERI_PRODUCT TABLE
INSERT INTO NON_PERI_PRODUCT
VALUES
('454'),
('455')
GO

-- Insert data in SUPPLY_DATE TABLE
INSERT INTO SUPPLY_DATE
VALUES
('452', '44', '02-2-2020'),
('452', '44', '04-15-2020'),
('452', '44', '08-15-2019'),
('455', '44', '03-24-2020'),
('455', '44', '03-25-2020'),
('455', '44', '03-26-2020'),
('453', '44', '03-27-2020'),
('453', '42', '03-28-2020'),
('453', '42', '03-29-2020'),
('451', '42', '03-30-2020'),
('451', '42', '04-01-2020'),
('454', '41', '04-01-2020');

-- Insert data in ARRANGEMENT TABLE
INSERT INTO ARRANGEMENT
VALUES
('451', 'P012', '3800','03-15-2020'),
('452', 'P012', '3802','04-17-2020'),
('453', 'P012', '3803', '03-07-2020'),
('455', 'P012', '3803', '03-15-2021')
GO

-- Insert data in BUY TABLE
INSERT INTO BUY
VALUES
('P002', '1009', '125', '02-2-2020'),
('P002', '1011', '126', '04-15-2020'),
('P002', '1012', '127', '08-15-2019'),
('P002', '1013', '125', '03-24-2020'),
('P002', '1014', '127', '03-25-2020'),
('P002', '1015', '125', '03-26-2020'),
('P002', '1016', '125', '03-27-2020'),
('P002', '1017', '127', '03-28-2020'),
('P002', '1018', '125', '03-29-2020'),
```

```
('P002', '1019', '125', '03-30-2020'),
('P002', '1020', '127', '04-01-2020'),
('P002', '1021', '127', '04-01-2020'),
('P001', '1018', '125', '03-29-2020');
GO

-- Insert data in ASSIGNMENT TABLE
INSERT INTO ASSIGNMENT
VALUES
('P010','3800','123' ,'04-07-2020'),
('P010','3802','123' ,'04-08-2020'),
('P011','3800','123' ,'04-10-2020'),
('P011','3802','123' ,'04-09-2020'),
('P012','3803','124' ,'04-12-2020'),
('P012','3804','124' ,'04-13-2020'),
('P010','3800','126' ,'04-08-2020'),
('P012','3800','126' ,'04-08-2020');
GO

-- Insert data in GIVEN_FREE TABLE
INSERT INTO GIVEN_FREE
VALUES
('P001','123', '9002', '1009', '125'),
('P003','125', '9003', '1012', '127'),
('P007','123', '9000', '1010', '125'),
('P005','125', '9001', '1011', '126'),
('P011','127', '9004', '1011', '126')
GO

-- Insert data in WORKS TABLE
INSERT INTO WORKS
VALUES
('P011', '123'),
('P010', '123'),
('P012', '125')
GO

-- Insert data in ONLINE_ORDER TABLE
INSERT INTO ONLINE_ORDER
VALUES
('P001', '451', 1, 12, '06-07-2010', 'Master'),
('P002', '452', 2, 435, '06-08-2010', 'Visa'),
('P001', '452', 3, 67, '06-08-2010', 'Visa'),
('P002', '453', 4, 234, '06-09-2010', 'Master'),
('P001', '453', 5, 345, '06-09-2010', 'Master'),
('P001', '454', 6, 234, '06-10-2010', 'Visa'),
('P001', '454', 7, 100, '06-11-2010', 'Visa')
GO

-- Insert data in STORE_ORDER TABLE
INSERT INTO STORE_ORDER
VALUES
```

```
('P010', '123', 451,'P001',1, 12, '02-07-2010', 'Master'),
('P010', '123', 455,'P002',2, 435, '03-11-2010', 'Visa'),
('P010', '125', 451,'P001',3, 67, '02-12-2010', 'Visa'),
('P010', '126', 454,'P003',4, 234, '03-09-2010', 'Master'),
('P010', '123', 454,'P001',5, 345, '04-10-2010', 'Master'),
('P010', '126', 452,'P001',6, 234, '05-08-2010', 'Visa'),
('P010', '125', 452,'P005',7, 100, '02-09-2010', 'Visa'),
('P010', '123', 451,'P001',8, 12, '02-07-2019', 'Master'),
('P010', '123', 455,'P002',9, 435, '03-11-2019', 'Visa'),
('P010', '125', 451,'P001',10, 67, '02-12-2019', 'Visa'),
('P010', '126', 454,'P003',11, 234, '03-09-2019', 'Master'),
('P010', '123', 454,'P001',12, 345, '04-10-2019', 'Master'),
('P010', '126', 452,'P001',13, 234, '05-08-2019', 'Visa'),
('P010', '125', 452,'P005',14, 100, '02-09-2019', 'Visa'),
('P010', '123', 451,'P001',15, 12, '02-07-2019', 'Master'),
('P010', '123', 454,'P001',16, 345, '04-10-2019', 'Master'),
('P010', '126', 452,'P001',17, 234, '05-08-2019', 'Visa'),
('P010', '125', 451,'P001',18, 67, '02-12-2019', 'Visa'),
('P010', '123', 454,'P001',19, 345, '04-10-2019', 'Master'),
('P010', '126', 452,'P001',20, 234, '05-08-2019', 'Visa'),
('P010', '123', 454,'P001',21, 345, '04-10-2019', 'Master'),
('P010', '126', 452,'P001',22, 234, '05-08-2019', 'Visa'),
('P010', '123', 454,'P001',23, 345, '04-10-2019', 'Master'),
('P010', '126', 452,'P001',24, 234, '05-08-2019', 'Visa'),
('P010', '123', 454,'P001',25, 345, '04-10-2019', 'Master'),
('P010', '126', 452,'P001',26, 234, '05-08-2019', 'Visa'),
('P010', '126', 451,'P004',27, 12, '08-08-2019', 'Visa'),
('P010', '123', 452,'P004',28, 798, '04-15-2019', 'Master'),
('P010', '126', 453,'P004',29, 443, '03-08-2019', 'Master'),
('P010', '126', 452,'P004',30, 43, '01-08-2019', 'Master')
GO
```

f. Dependency Diagram

Kunal Mukherjee (kxm180046)
Suraj Kothawade (snk170001)
Chia-Kai Pan (cxp170010)

4/25/20

CS 6360

Dr. Wili Wu

Dr. Smita Ghosh