RETAIL STORE SALES MANAGEMENT SYSTEM

Shashidhar Reddy Daida, Asritha Mary Kondaveeti, Shresta Sankapally, Mithra Gatla

Data Science, Department of Information Science, University of North Texas

INFO 5707- Data Modelling for Information Professionals

Instructor: Xiaoyu Zhang

I. DATABASE OVERVIEW

1. OBJECTIVE OF THE PROJECT

The main objective of this database is to provide effective and efficient retail store sales management system (RSMS). This database maintains detailed information about its customers. Information about each sale and return transactions are stored in the database along with the products, quantity, and payment mode. Other information includes Employee information who makes the transaction and about workstation at which transaction is made. This database is managed by admin who has access to maintain all the activities of a store such as stock updates, employee updates etc.

2. SCOPE OF THE PROJECT:

Retail store sales management system analyses past sales, sales returns and ability to view the existing sales. As we know manual systems are quite tedious, time consuming, less efficient and less accurate in comparison to the computerized system. This eliminates manual process of maintaining records and provides quick way of operation by automating them. This project helps to

a) Reduce operating costs and eliminate complexity
b) Better identify, interact with, and retain customers
c) Gain real-time view on sales and returns of products and their respective transactions and payment modes
d) Instantly respond according to the availability of stocks
e) Make faster decisions on pricing, promotions, and reorders
f) Track records so that past records can be verified
g) Maintain high level of security for data leaking as only admin can access the database for availability of stocks, employee details etc.

3. USER REQUIREMENTS:

The purpose of this database is to provide an ease to manage the product sales and customer information of a retail store. When a customer purchases a product, the transaction will be recorded such as mode of payment, customer information, product information, total price (sub-total and tax). The Return items information will be logged on to the system when a customer places a return for an item purchased. Each transaction is stored in database so that they can be retrieved and verified. The workstations hold the information of the POS systems

where the sales are done. The system records the information of the workstations when a transaction is done. The system also holds the information of the availability of the products in the store. The detailed information of every employee in the store and the operations performed by them are maintained in the system.

II. TABLES AND RELATIONSHIPS

This section contains Data Dictionary, Business Rules, and Entity Relationship Diagram.

1. DATA DICTIONARY:

Table Name	Attribute Name	Contents	Data Type	Format	Required	PK or FK	FK Referenced
Workstation	Wstation_Id	Id of workstation	Int	10	Y	PK	
	Wstation_Name	Name of the workstation	nvarchar(100)	XXX-XXXX	Y		
Employee	Emp_Id	Id for Employee	nvarchar(10)	XXXX	Y	PK	
	Emp_FName	First name of employee	varchar(100)	XXXXXX	Y		
	Emp_LName	Last Name of employee	varchar(10)	XXXXX	Y		
	Emp_Phone	Phone number of Employee	nvarchar(50)	2565647475			
Operator	Operator_Id	Id of operator	nvarchar(10)	XXXXX	Y	PK, FK	Employee
	Operator-Type	Role of the operator (Cashier/ Manager etc.)	varchar(50)	XXXXX	Y	PK	

Customer	Cust_Id	Id of the customer who made transaction	Int	XXXXXX	Y	PK	
	Cust_FName	First name of the customer	varchar(100)	XXXXXX	Y		
	Cust_LName	Last name of the customer	varchar(100)	XXXXXX	Y		
	Cust_Phone	Phone number of customers	nvarchar(50)	1445641132	Y		
Product	Product_Id	Id of the product	nvarchar(100)	XXXXXXX	Y	PK	
	Product_Name	Name of the product	varchar(100)	XXXXXX	Y		
	Price	Price of the product	float	$ 9999.99	Y		
	Status	Sale status of the product(In stock/Out of stock)	varchar(100)	XXXXXX	Y		
Transaction	Trans_Number	Transaction number	nvarchar(100)	SA999/RT9999	Y	PK	
	Trans_Type	Type of the transaction sale/return	varchar(10)	XXXXXXX	Y		
	Trans_Date	Date of transaction made	date	yymmdd	Y		
	Cust_Id	Id of the customer who made transaction	Int	XXXXXXX	Y	FK	Customer
	Operator_Id	Id of operator who made transaction	nvarchar(10)	XXXXXXX	Y	FK	Operator

	Wstation_Id	Id of the workstation where the transaction made	Int	XXXXXX	Y	FK	Workstation
	Product_Qty	Total number of products in the transaction	Int	9999999	Y		
	Sub_Total	Total price of the products (positive for sale transaction and negative for return transaction)	float	$ 9999.99	Y		
	Tax	Tax included in the transaction	float	$ 9.99	Y		
	Total_Amt	Total amount of the transaction(Sub total + tax) (positive for sale transaction and negative for return transaction)	float	$ 9999.99	Y		
Transaction_ Sale_Item	Trans_Number	Sale transaction number (This is sale transaction only)	nvarchar(100)	SA99999	Y	PK, FK	Transaction
	Product_Id	Id of the product sold	nvarchar(100)	XXXXXXX	Y	PK, FK	Product
	Product_Qty	Quantity of that particular product sold	Int	999999	Y		

Transaction_ Return_Item	Trans_Number	Return transaction number (This is return transaction only)	nvarchar(100)	RT99999	Y	PK	
	Product_Id	Id of the product returned	nvarchar(100)	XXXXXXX	Y	FK	Product
	Product_Qty	Quantity of that particular product returned	Int	999999	Y		
	Ref_Trans_ Number	Transaction number of original sale transaction (sale)	nvarchar(100)	SA99999	Y	FK	Transaction
Payment_ Mode	Trans_Number	Transaction number (sale/return)	nvarchar(100)	SA999/RT999	Y	FK	Transaction
	Payment_ Medium	Mode of payment used Cash/Card	varchar(100)		Y		
	Amt	Amount paid using the particular payment mode	float	$ 9999.99	Y		
Transaction_ Payment (Bridge table)	Trans_Number	Transaction number	nvarchar(100)	SA9999/RT999	Y	PK	
	Paymet_ Medium	Mode of payment used Cash/Card	varchar(100)	XXXXXXX	Y		
Sale_ Transaction	Trans_Number	Transaction number (Sale only)	nvarchar(100)	SA9999	Y		

(Bridge table)	Product_Id	Id of the product	nvarchar(100)	XXXXXXX	Y	PK	
Return_ Transaction (Bridge table)	Trans_Number	Transaction number (Return only)	nvarchar(100)	RT999	Y	PK	
	Product_Id	Id of the product	nvarchar(100)	XXXXXXX	Y		

2. BUSINESS RULES:

a) Each workstation handles zero or many transactions.

b) Each transaction must be performed at one workstation.

c) Each operator can place one or more transactions.

d) Each transaction must be placed by one operator.

e) Each operator role can be assigned to one or more employees.

f) Each employee must be assigned to perform one operator role.

g) Each customer can place one or more transactions.

h) Each transaction is linked to one customer.

i) Each transaction can be paid by multiple payment modes.

j) Each payment mode can be used to pay for multiple transactions.

k) Each transaction contains one or more sale items.

l) Each sale item must be included in one transaction.

m) Each transaction may involve zero or many return items.

n) Each return item must be included in one transaction.

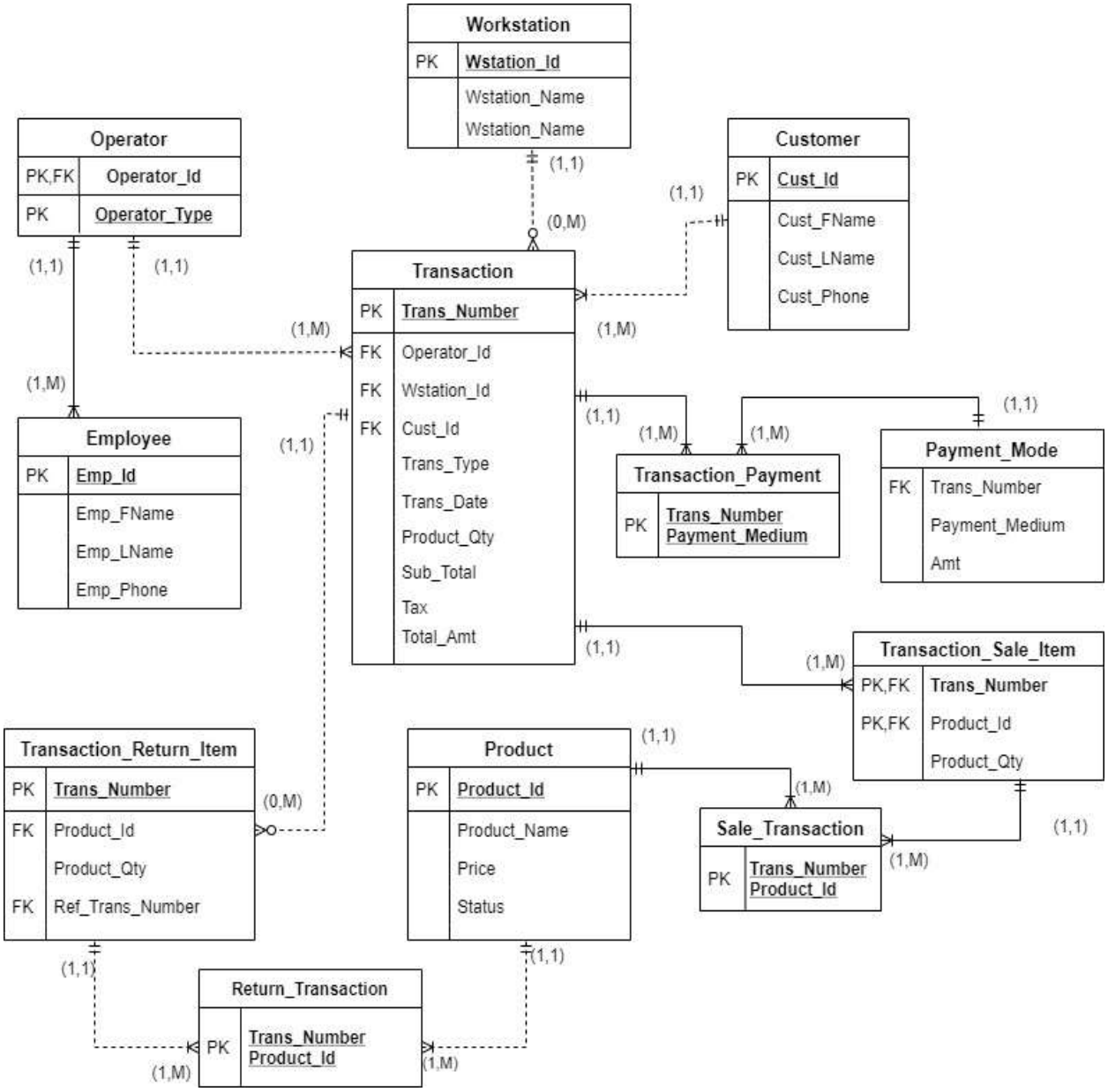
o) Each product may have one or more returns.

p) Each return item can have one or many products.

q) Each sale item contains one or more products.

r) Each product may have one or more sales.

3. ENTITY RELATIONSHIP DIAGRAM:



III. MySQL WORKBENCH DATABASE SYSTEM

This section contains queries to create tables, to insert data into created data tables and to retrieve data from the database.

1. QUERIES TO CREATE TABLES:

```
-- Custom Database SQL Script
-- Version 0.1.1
-- Designed by Group_4
--
-- Select database to configure
--
USE `salesmanagementsystem`;
--
-- Turn off foreign key constraints until DB is created
--
SET FOREIGN_KEY_CHECKS=0;
```

❖ <u>WORKSTATION TABLE:</u>

```
--
-- Table structure for table `WorkStation`
--
-- LOCK TABLES `media_consortiums` WRITE;
-- UNLOCK TABLES;
DROP TABLE IF EXISTS `WorkStation`;
CREATE TABLE `WorkStation` (
  `Wstation_Id` int NOT NULL AUTO_INCREMENT,
  `Wstation_Name` nvarchar(100) NOT NULL,
  -- changes made here
```

```
  PRIMARY KEY (`Wstation_Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- EMPLOYEE TABLE:

```
--
-- Table structure for table `Employee`
--
DROP TABLE IF EXISTS `Employee`;
CREATE TABLE `Employee` (
  `Emp_Id` nvarchar(10) NOT NULL,
  `Emp_FName` varchar(100) NOT NULL,
  `Emp_LName` varchar(100) NOT NULL,
  `Emp_Phone` nvarchar(50),
  PRIMARY KEY (`Emp_Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

- OPERATOR TABLE:

```
--
-- Table structure for table `Operator`
--
DROP TABLE IF EXISTS `Operator`;
CREATE TABLE `Operator` (
  `Operator_Id` nvarchar(10) NOT NULL,
  `Operator_Type` varchar(50) NOT NULL,
  PRIMARY KEY (`Operator_Id`),
  CONSTRAINT `Operator_Emp_FK` FOREIGN KEY (`Operator_Id`) REFERENCES
`Employee` (`Emp_Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

❖ CUSTOMER TABLE:

```
--
-- Table structure for table `Customer`
--
DROP TABLE IF EXISTS `Customer`;
CREATE TABLE `Customer` (
  `Cust_Id` Int Not Null AUTO_INCREMENT,
  `Cust_FName` varchar(100) Not Null,
  `Cust_LName` varchar(100),
  `Cust_Phone` nvarchar(50) Not Null,
  PRIMARY KEY (`Cust_Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

❖ PRODUCTS TABLE:

```
--
-- Table structure for table `Products`
--
DROP TABLE IF EXISTS `Product`;
CREATE TABLE `Product` (
  `Product_Id` nvarchar(100) NOT NULL,
  `Product_Name` Varchar(100) Not Null,
  `Price` float,
  `Status` Varchar(100) Not Null,
  PRIMARY KEY (`Product_Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

❖ TRANSACTION TABLE:

```
--
-- Table structure for table `Transaction`
--
DROP TABLE IF EXISTS `Transaction`;
CREATE TABLE `Transaction` (
  `Trans_Number` nvarchar(100)  NOT NULL,
  `Trans_Type` varchar(10)  NOT NULL,
  `Trans_Date` date  NOT NULL,
  `Cust_Id` Int Not Null,
  `Operator_Id` nvarchar(10)  NOT NULL,
  `Wstation_Id` int NOT NULL,
  `Product_Qty` int NOT NULL,
  `Sub_Total` float NOT NULL,
  `Tax` float NOT NULL,
  `Total_Amt` float NOT NULL,
  -- changes made here
  --
  PRIMARY KEY (`Trans_Number`),
  CONSTRAINT `Trans_Cust_FK` FOREIGN KEY (`Cust_Id`) REFERENCES `Customer` (`Cust_Id`),
  CONSTRAINT `Trans_Operator_FK` FOREIGN KEY (`Operator_Id`) REFERENCES `Operator` (`Operator_Id`),
  CONSTRAINT `Trans_Wstation_FK` FOREIGN KEY (`Wstation_Id`) REFERENCES `WorkStation` (`Wstation_Id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8;
```

❖ TRANSACTION_SALE_ITEM:

```
--
-- Table structure for table `Transaction_Sale_Item`
--
DROP TABLE IF EXISTS `Transaction_Sale_Item`;
CREATE TABLE `Transaction_Sale_Item` (
  `Trans_Number` nvarchar(100)  NOT NULL,
  `Product_Id` nvarchar(100) NOT NULL,
  `Product_Qty` int NOT NULL,
  PRIMARY KEY (`Trans_Number`,`Product_Id`),
  CONSTRAINT `Sale_Trans_FK` FOREIGN KEY (`Trans_Number`) REFERENCES
`Transaction` (`Trans_Number`),
  CONSTRAINT `Sale_Product_FK` FOREIGN KEY (`Product_Id`) REFERENCES
`Product` (`Product_Id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8;
```

❖ TRANSACTION_RETURN_ITEM:

```
--
-- Table structure for table `Transaction_Return_Item`
--
DROP TABLE IF EXISTS `Transaction_Return_Item`;
CREATE TABLE `Transaction_Return_Item` (
  `Trans_Number` nvarchar(100)  NOT NULL,
  `Product_Id` nvarchar(100) NOT NULL,
  `Product_Qty` int NOT NULL,
  `Ref_Trans_Number` nvarchar(100)  NOT NULL,
  PRIMARY KEY (`Trans_Number`),
  --
```

```
  CONSTRAINT `Return_Trans_FK` FOREIGN KEY (`Ref_Trans_Number`)
REFERENCES `Transaction` (`Trans_Number`),

  CONSTRAINT `Return_Product_FK` FOREIGN KEY (`Product_Id`) REFERENCES
`Product` (`Product_Id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- PAYMENT_MODE:

```
--

-- Table structure for table `Payment_Mode`

--

DROP TABLE IF EXISTS `Payment_Mode`;

CREATE TABLE `Payment_Mode` (

   `Trans_Number` nvarchar(100)  NOT NULL,

   `Payment_Medium` varchar(100)  NOT NULL,

   `Amt` float  NOT NULL,

   CONSTRAINT `Payment_Trans_FK` FOREIGN KEY (`Trans_Number`)
REFERENCES `Transaction` (`Trans_Number`),

  -- changes made here

  PRIMARY KEY (`Trans_Number`,`Payment_Medium`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- TRANSACTION_PAYMENT:

```
--

-- Table structure for table `Transaction_Payment

--

DROP TABLE IF EXISTS `Transaction_Payment`;

CREATE TABLE `Transaction_Payment` (

   `Trans_Number` nvarchar(100)  NOT NULL,

   `Payment_Medium` varchar(100)  NOT NULL,
```

```
  PRIMARY KEY (`Trans_Number`,`Payment_Medium`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- SALE_TRANSACTION:

```
--
-- Table structure for table `Sale_Transaction
--
DROP TABLE IF EXISTS `Sale_Transaction`;
CREATE TABLE `Sale_Transaction` (
  `Trans_Number` nvarchar(100)  NOT NULL,
  `Product_Id` nvarchar(100)  NOT NULL,
  PRIMARY KEY (`Trans_Number`,`Product_Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

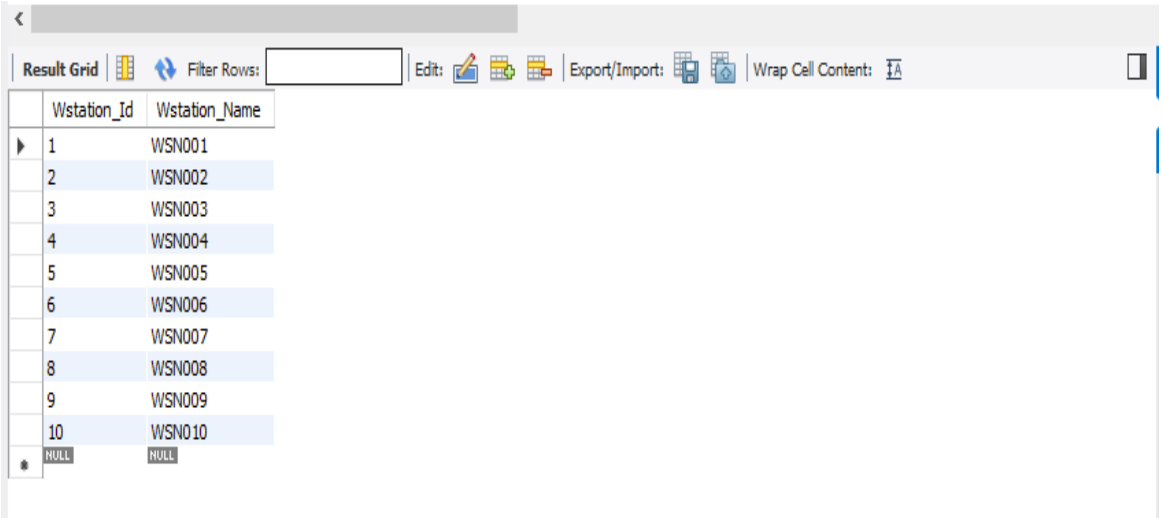
- RETURN_TRANSACTION:

```
--
-- Table structure for table `Return_Transaction
--
DROP TABLE IF EXISTS `Return_Transaction`;
CREATE TABLE `Return_Transaction` (
  `Trans_Number` nvarchar(100)  NOT NULL,
  `Product_Id` nvarchar(100)  NOT NULL,
  PRIMARY KEY (`Trans_Number`,`Product_Id`)
)  ENGINE=InnoDB DEFAULT CHARSET=utf8;
--
-- Turn foreign key constraints on
--
SET FOREIGN_KEY_CHECKS=1;
```

2. INSERTING DATA INTO TABLES:

❖ <u>WORKSTATION TABLE:</u>

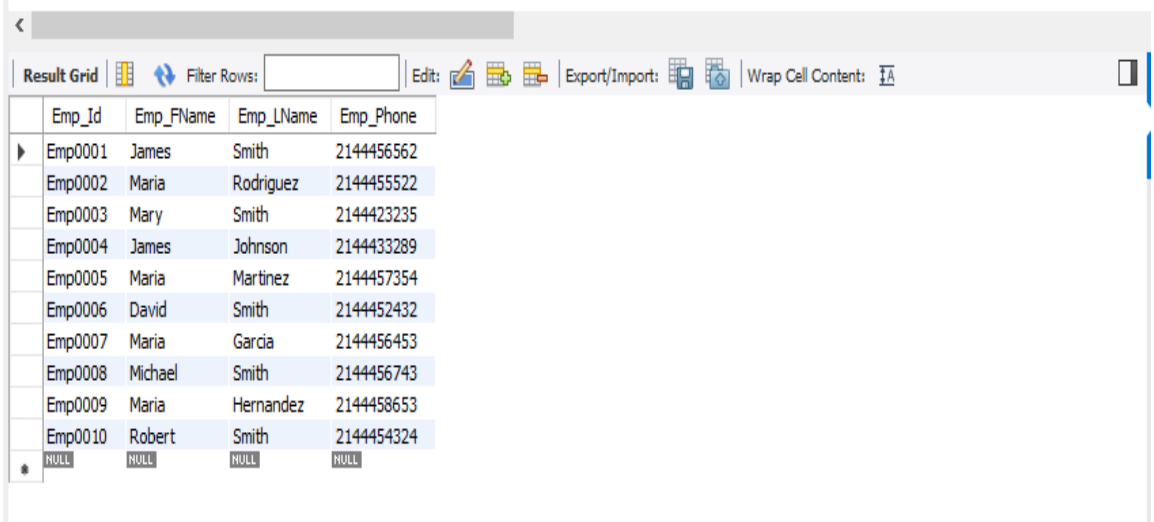
```
--
-- Select database to configure
--
USE `salesmanagementsystem`;
--
-- Inserting data for table `Workstation`
--
LOCK TABLES `WorkStation` WRITE;
INSERT INTO `WorkStation` (Wstation_Name) VALUES
  ('WSN001'),
  ('WSN002'),
  ('WSN003'),
  ('WSN004'),
  ('WSN005'),
  ('WSN006'),
  ('WSN007'),
  ('WSN008'),
  ('WSN009'),
  ('WSN010');
UNLOCK TABLES;
```



Wstation_Id	Wstation_Name
1	WSN001
2	WSN002
3	WSN003
4	WSN004
5	WSN005
6	WSN006
7	WSN007
8	WSN008
9	WSN009
10	WSN010
NULL	NULL

❖ <u>EMPLOYEE TABLE:</u>

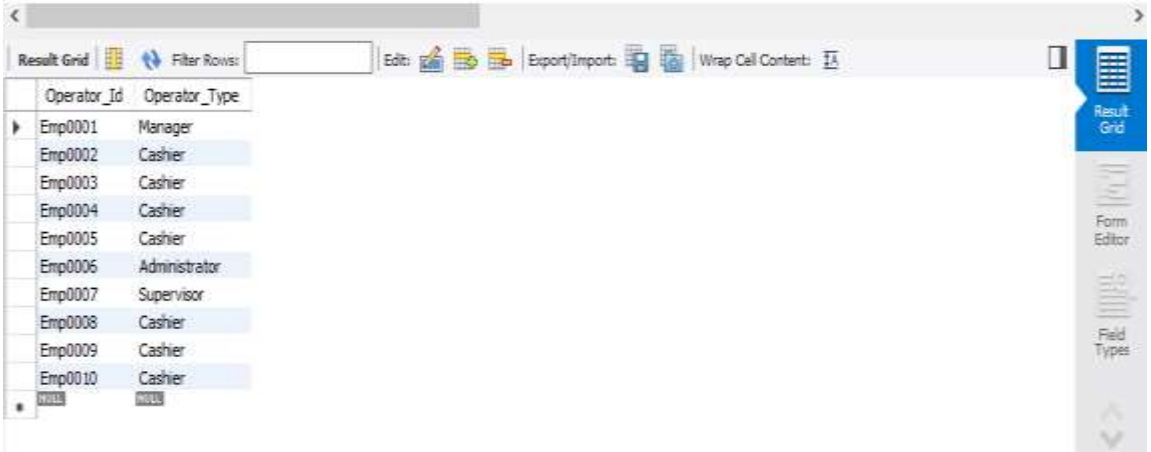
```
-- Inserting data for table `Employee`
--
LOCK TABLES `Employee` WRITE;
INSERT INTO `Employee` (Emp_Id,Emp_FName,Emp_LName,Emp_Phone)
VALUES
  ('Emp0001','James','Smith','2144456562'),
  ('Emp0002','Maria','Rodriguez','2144455522'),
  ('Emp0003','Mary','Smith','2144423235'),
  ('Emp0004','James','Johnson','2144433289'),
  ('Emp0005','Maria','Martinez','2144457354'),
  ('Emp0006','David','Smith','2144452432'),
  ('Emp0007','Maria','Garcia','2144456453'),
  ('Emp0008','Michael','Smith','2144456743'),
  ('Emp0009','Maria','Hernandez','2144458653'),
  ('Emp0010','Robert','Smith','2144454324');
UNLOCK TABLES;
```



Emp_Id	Emp_FName	Emp_LName	Emp_Phone
Emp0001	James	Smith	2144456562
Emp0002	Maria	Rodriguez	2144455522
Emp0003	Mary	Smith	2144423235
Emp0004	James	Johnson	2144433289
Emp0005	Maria	Martinez	2144457354
Emp0006	David	Smith	2144452432
Emp0007	Maria	Garcia	2144456453
Emp0008	Michael	Smith	2144456743
Emp0009	Maria	Hernandez	2144458653
Emp0010	Robert	Smith	2144454324
NULL	NULL	NULL	NULL

❖ <u>OPERATOR TABLE:</u>

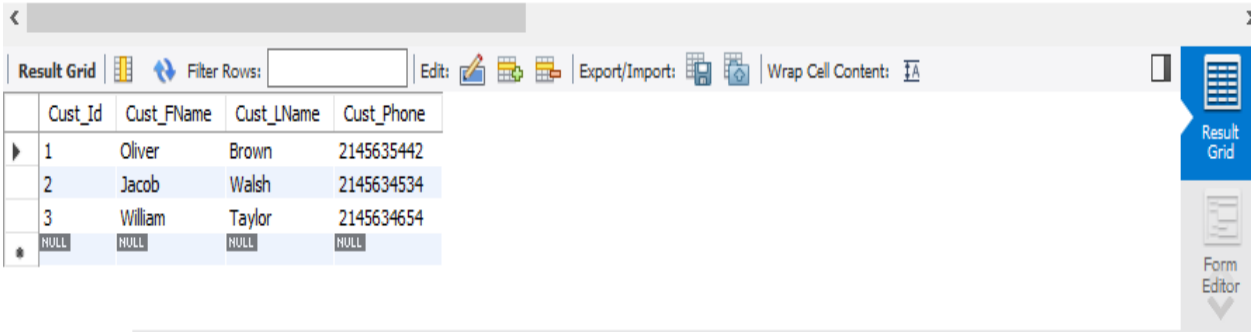
```
-- Inserting data for table `Operator`
--
LOCK TABLES `Operator` WRITE;
INSERT INTO `Operator` (Operator_Id,Operator_Type) VALUES
  ('Emp0008','Cashier'),
  ('Emp0006','Administrator'),
  ('Emp0007','Supervisor'),
  ('Emp0001','Manager'),
  ('Emp0002','Cashier'),
  ('Emp0005','Cashier'),
  ('Emp0004','Cashier'),
  ('Emp0003','Cashier'),
  ('Emp0010','Cashier'),
  ('Emp0009','Cashier');
UNLOCK TABLES;
```



Operator_Id	Operator_Type
Emp0001	Manager
Emp0002	Cashier
Emp0003	Cashier
Emp0004	Cashier
Emp0005	Cashier
Emp0006	Administrator
Emp0007	Supervisor
Emp0008	Cashier
Emp0009	Cashier
Emp0010	Cashier
NULL	NULL

❖ <u>CUSTOMER TABLE:</u>

```
LOCK TABLES `Customer` WRITE;
INSERT INTO `Customer` (Cust_FName,Cust_LName,Cust_Phone) VALUES
  ('Oliver','Brown','2145635442'),
  ('Jacob','Walsh','2145634534'),
  ('William','Taylor','2145634654');
UNLOCK TABLES;
```

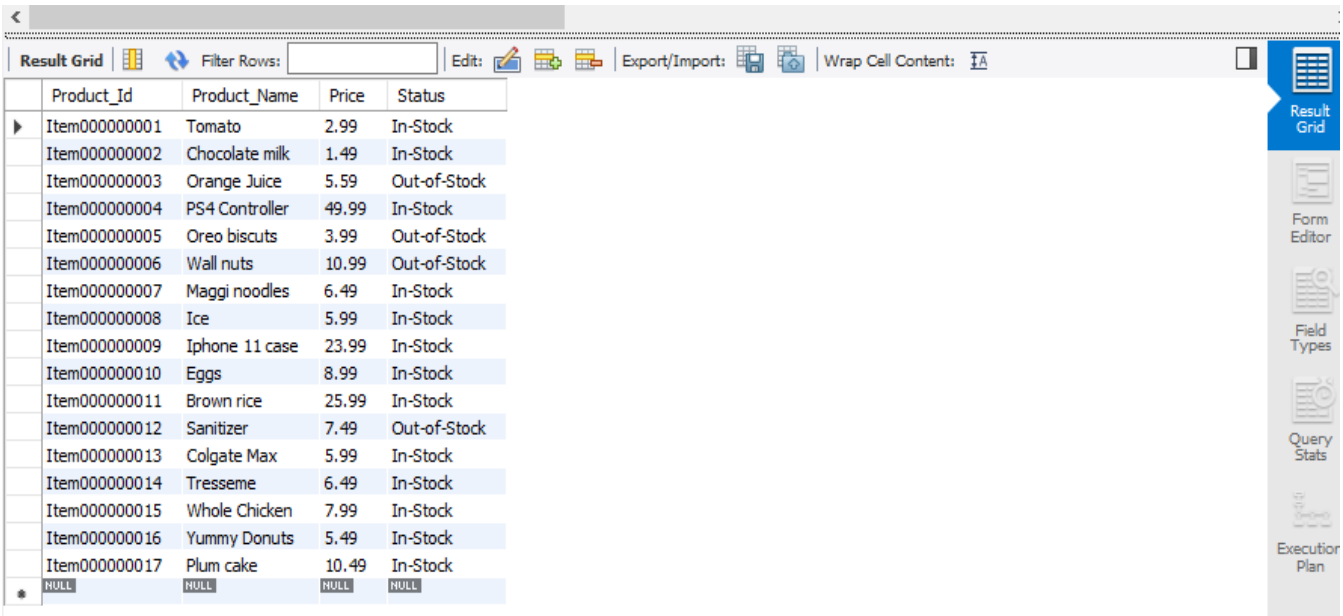


Cust_Id	Cust_FName	Cust_LName	Cust_Phone
1	Oliver	Brown	2145635442
2	Jacob	Walsh	2145634534
3	William	Taylor	2145634654
NULL	NULL	NULL	NULL

❖ <u>PRODUCTS TABLE:</u>

```
--
-- Inserting data for table `Product`
--
LOCK TABLES `Product` WRITE;
INSERT INTO `Product` (Product_Id, Product_Name, Price, Status) VALUES
  ('Item000000001','Tomato','2.99','In-Stock'),
```

```
('Item000000002','Chocolate milk','1.49','In-Stock'),
('Item000000003','Orange Juice','5.59','Out-of-Stock'),
('Item000000004','PS4 Controller','49.99','In-Stock'),
('Item000000005','Oreo biscuts','3.99','Out-of-Stock'),
('Item000000006','Wall nuts','10.99','Out-of-Stock'),
('Item000000007','Maggi noodles','6.49','In-Stock'),
('Item000000008','Ice','5.99','In-Stock'),
('Item000000009','Iphone 11 case','23.99','In-Stock'),
('Item000000010','Eggs','8.99','In-Stock'),
('Item000000011','Brown rice','25.99','In-Stock'),
('Item000000012','Sanitizer','7.49','Out-of-Stock'),
('Item000000013','Colgate Max','5.99','In-Stock'),
('Item000000014','Tresseme','6.49','In-Stock'),
('Item000000015','Whole Chicken','7.99','In-Stock'),
('Item000000016','Yummy Donuts','5.49','In-Stock'),
('Item000000017','Plum cake','10.49','In-Stock');
UNLOCK TABLES;
```



Product_Id	Product_Name	Price	Status
Item000000001	Tomato	2.99	In-Stock
Item000000002	Chocolate milk	1.49	In-Stock
Item000000003	Orange Juice	5.59	Out-of-Stock
Item000000004	PS4 Controller	49.99	In-Stock
Item000000005	Oreo biscuts	3.99	Out-of-Stock
Item000000006	Wall nuts	10.99	Out-of-Stock
Item000000007	Maggi noodles	6.49	In-Stock
Item000000008	Ice	5.99	In-Stock
Item000000009	Iphone 11 case	23.99	In-Stock
Item000000010	Eggs	8.99	In-Stock
Item000000011	Brown rice	25.99	In-Stock
Item000000012	Sanitizer	7.49	Out-of-Stock
Item000000013	Colgate Max	5.99	In-Stock
Item000000014	Tresseme	6.49	In-Stock
Item000000015	Whole Chicken	7.99	In-Stock
Item000000016	Yummy Donuts	5.49	In-Stock
Item000000017	Plum cake	10.49	In-Stock
NULL	NULL	NULL	NULL

❖ TRANSACTION TABLE:

```
LOCK TABLES `Transaction` WRITE;
INSERT INTO `Transaction` (Trans_Number, Trans_Type, Trans_Date, Cust_Id, Operator_Id,
Wstation_Id, Product_Qty, Sub_Total, Tax, Total_Amt) VALUES
 ('SA000000001','SALE','20100612',1, 'Emp0004', 5, 5, 65.95, 4.62, 69.94),
 ('SA000000002','SALE','20100612',2, 'Emp0003', 2, 2, 57.48, 4.02, 61.50),
 ('SA000000003','SALE','20100715',3, 'Emp0004', 5, 10, 60.90, 4.26, 65.16),
 ('SA000000004','SALE','20100801',1, 'Emp0010', 1, 20, 249.30, 17.45, 266.75),
 ('RT000000001','RETURN','20100705',2, 'Emp0009', 4, 1, 49.99, 0.00, 49.99),
 ('RT000000002','RETURN','20100823',1, 'Emp0003', 10 , 2, 99.98, 0.00, 99.98);
 UNLOCK TABLES;  ('RT000000002','RETURN','20100823',1, 'Emp0003', 10 , 2, 99.98, 0.00, 99.98);
 UNLOCK TABLES;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Trans_Number	Trans_Type	Trans_Date	Cust_Id	Operator_Id	Wstation_Id	Product_Qty	Sub_Total	Tax	Total_Amt
SA000000004	SALE	2010-08-01	1	Emp0010	1	20	249.3	17.45	266.75
SA000000003	SALE	2010-07-15	3	Emp0004	5	10	60.9	4.26	65.16
SA000000002	SALE	2010-06-12	2	Emp0003	2	2	57.48	4.02	61.5
SA000000001	SALE	2010-06-12	1	Emp0004	5	5	65.95	4.62	69.94
RT000000002	RETURN	2010-08-23	1	Emp0003	10	2	99.98	0	99.98
RT000000001	RETURN	2010-07-05	2	Emp0009	4	1	49.99	0	49.94
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Result Grid | Form Editor | Field Types

❖ TRANSACTION_SALE_ITEM:

```
--
-- Inserting data for table `Transaction_Sale_Item`
--
LOCK TABLES `Transaction_Sale_Item` WRITE;
```

```
INSERT INTO `Transaction_Sale_Item` (Trans_Number, Product_Id, Product_Qty) VALUES
('SA000000001','Item000000002', 2),
('SA000000001','Item000000006', 1),
('SA000000001','Item000000011', 2),
('SA000000002','Item000000004', 1),
('SA000000002','Item000000002', 1),
('SA000000003','Item000000001', 4),
('SA000000003','Item000000010', 3),
('SA000000003','Item000000013', 1),
('SA000000003','Item000000015', 2),
('SA000000004','Item000000005', 5),
('SA000000004','Item000000009', 1),
('SA000000004','Item000000012', 3),
('SA000000004','Item000000004', 2),
('SA000000004','Item000000011', 2),
('SA000000004','Item000000015', 2),
('SA000000004','Item000000001', 5);
UNLOCK TABLES;
```

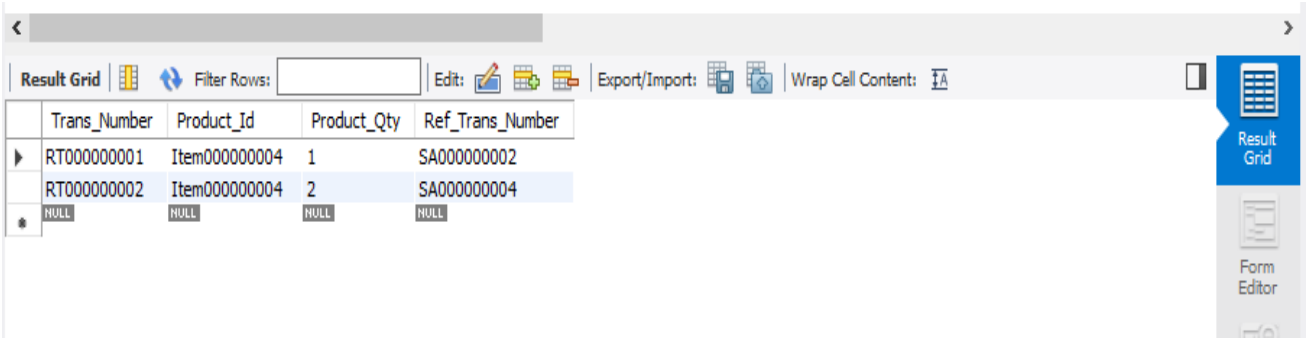
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Trans_Number	Product_Id	Product_Qty
SA000000001	Item000000002	2
SA000000001	Item000000006	1
SA000000001	Item000000011	2
SA000000002	Item000000002	1
SA000000002	Item000000004	1
SA000000003	Item000000001	4
SA000000003	Item000000010	3
SA000000003	Item000000013	1
SA000000003	Item000000015	2
SA000000004	Item000000001	5
SA000000004	Item000000004	2
SA000000004	Item000000005	5
SA000000004	Item000000009	1
SA000000004	Item000000011	2
SA000000004	Item000000012	3
SA000000004	Item000000015	2
NULL	NULL	NULL

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

❖ TRANSACTION_RETURN_ITEM:

```
--
-- Inserting data for table `Transaction_Return_Item`
--
LOCK TABLES `Transaction_Return_Item` WRITE;
INSERT INTO `Transaction_Return_Item` (Trans_Number, Product_Id, Product_Qty,Ref_Trans_Number) VALUES
  ('RT000000001','Item000000004', 1,'SA000000002'),
  ('RT000000002','Item000000004', 2,'SA000000004');
  UNLOCK TABLES;
```



Trans_Number	Product_Id	Product_Qty	Ref_Trans_Number
RT000000001	Item000000004	1	SA000000002
RT000000002	Item000000004	2	SA000000004
NULL	NULL	NULL	NULL

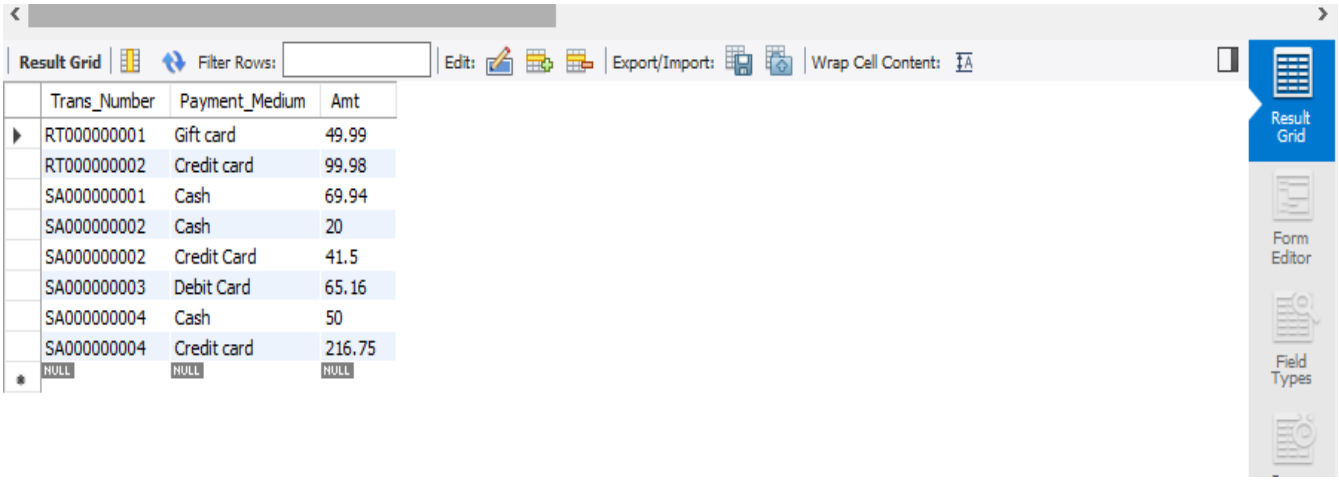
❖ PAYMENT_MODE:

```
--
-- Inserting data for table `Payment_Mode`
--
LOCK TABLES `Payment_Mode` WRITE;
INSERT INTO `Payment_Mode` (Trans_Number, Payment_Medium, Amt) VALUES
  ('SA000000001','Cash',69.94),
  ('SA000000002','Cash', 20.00),
  ('SA000000002','Credit Card', 41.50),
  ('SA000000003','Debit Card', 65.16),
  ('SA000000004','Cash', 50.00),
  ('SA000000004','Credit card', 216.75),
```

('RT000000001','Gift card', 49.99),

('RT000000002','Credit card', 99.98);

UNLOCK TABLES;



Trans_Number	Payment_Medium	Amt
RT000000001	Gift card	49.99
RT000000002	Credit card	99.98
SA000000001	Cash	69.94
SA000000002	Cash	20
SA000000002	Credit Card	41.5
SA000000003	Debit Card	65.16
SA000000004	Cash	50
SA000000004	Credit card	216.75
NULL	NULL	NULL

❖ TRANSACTION_PAYMENT:

--

-- Inserting data for table `Transaction_Payment`

--

LOCK TABLES `Transaction_Payment` WRITE;

INSERT INTO `Transaction_Payment` (Trans_Number, Payment_Medium) VALUES

('SA000000001','Cash'),

('SA000000002','Cash'),

('SA000000002','Credit Card'),

('SA000000003','Debit Card'),

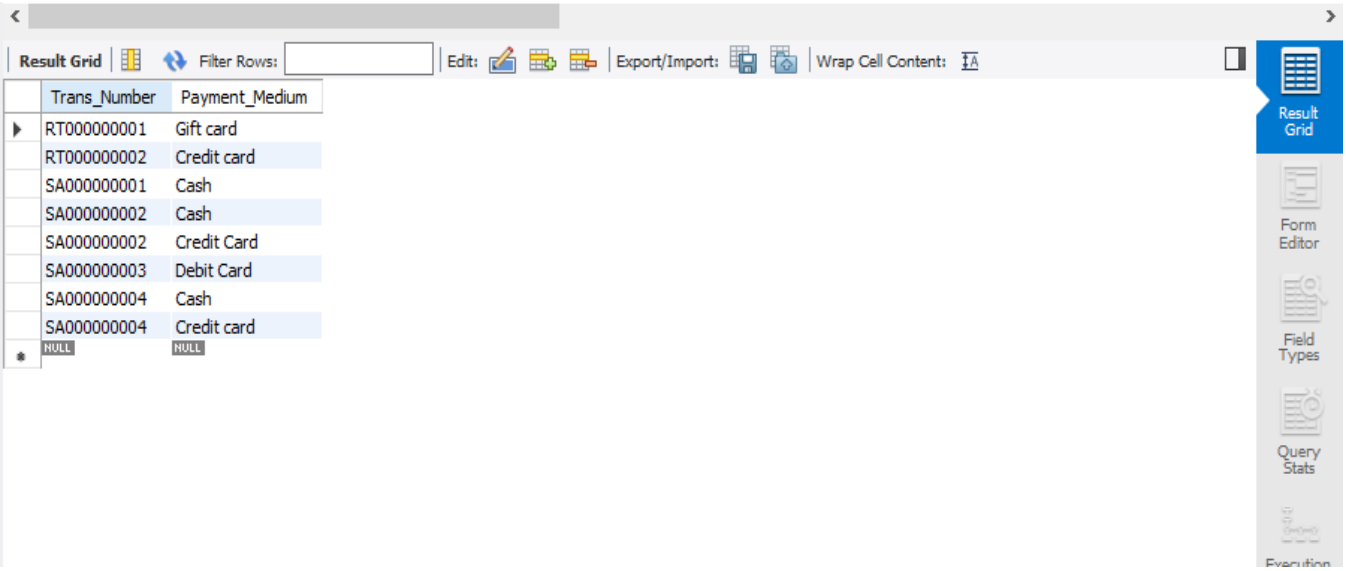
('SA000000004','Cash'),

('SA000000004','Credit card'),

('RT000000001','Gift card'),

('RT000000002','Credit card');

UNLOCK TABLES;

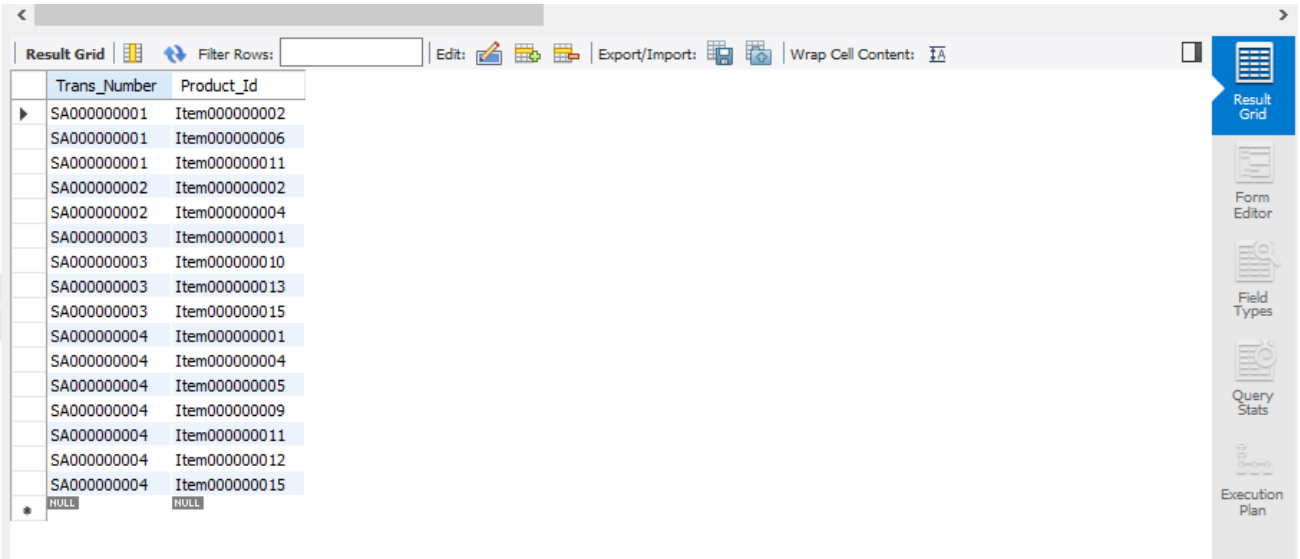


Trans_Number	Payment_Medium
RT000000001	Gift card
RT000000002	Credit card
SA000000001	Cash
SA000000002	Cash
SA000000002	Credit Card
SA000000003	Debit Card
SA000000004	Cash
SA000000004	Credit card
NULL	NULL

❖ <u>SALE_TRANSACTION:</u>

```
--
-- Inserting data for table `Sale_Transaction`
--
LOCK TABLES `Sale_Transaction` WRITE;
INSERT INTO `Sale_Transaction` (Trans_Number, Product_Id) VALUES
  ('SA000000001','Item000000002'),
  ('SA000000001','Item000000006'),
  ('SA000000001','Item000000011'),
  ('SA000000002','Item000000004'),
  ('SA000000002','Item000000002'),
  ('SA000000003','Item000000001'),
  ('SA000000003','Item000000010'),
  ('SA000000003','Item000000013'),
  ('SA000000003','Item000000015'),
  ('SA000000004','Item000000005'),
  ('SA000000004','Item000000009'),
  ('SA000000004','Item000000012'),
  ('SA000000004','Item000000004'),
```

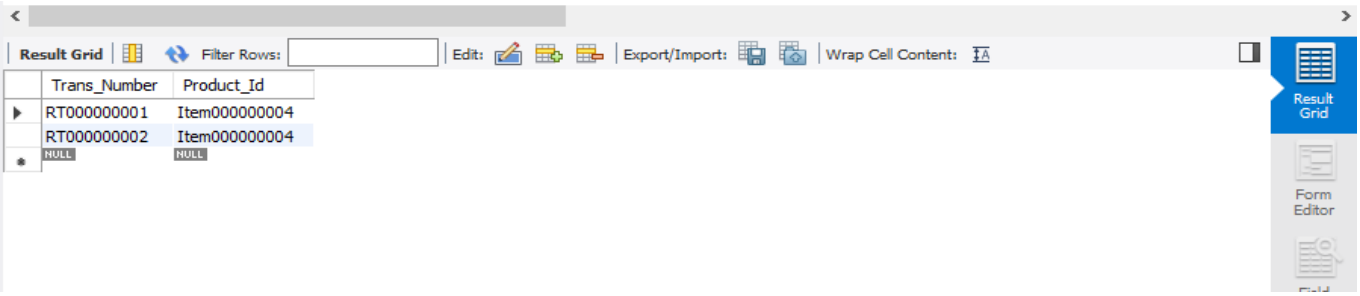
```
('SA000000004','Item000000011'),
('SA000000004','Item000000015'),
('SA000000004','Item000000001');
UNLOCK TABLES;
```



Trans_Number	Product_Id
SA000000001	Item000000002
SA000000001	Item000000006
SA000000001	Item000000011
SA000000002	Item000000002
SA000000002	Item000000004
SA000000003	Item000000001
SA000000003	Item000000010
SA000000003	Item000000013
SA000000003	Item000000015
SA000000004	Item000000001
SA000000004	Item000000004
SA000000004	Item000000005
SA000000004	Item000000009
SA000000004	Item000000011
SA000000004	Item000000012
SA000000004	Item000000015
NULL	NULL

❖ <u>RETURN_TRANSACTION:</u>

```
--
-- Inserting data for table `Return_Transaction`
--
LOCK TABLES `Return_Transaction` WRITE;
INSERT INTO `Return_Transaction` (Trans_Number, Product_Id) VALUES
  ('RT000000001','Item000000004'),
  ('RT000000002','Item000000004');
  UNLOCK TABLES;
```

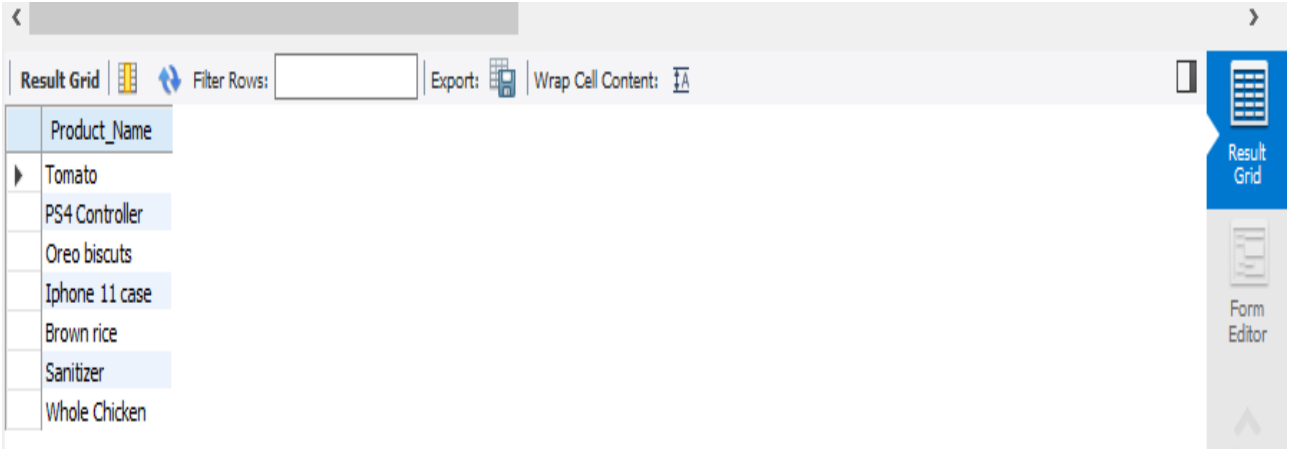


Trans_Number	Product_Id
RT000000001	Item000000004
RT000000002	Item000000004
NULL	NULL

3. RETRIEVING DATA FROM DATABASE:

- Get names of the products purchased by a customer whose name is Oliver Brown.

```
SELECT distinct P.Product_Name from
Product P inner join Transaction_Sale_Item  TSI
        on P.Product_Id = TSI.Product_Id
        inner join Transaction T
        on T.Trans_Number = TSI.Trans_Number
        inner join Customer C
        on C.Cust_Id = T.Cust_Id
        where C.Cust_FName = 'Oliver' and Cust_LName = 'Brown';
```



- Get total turnover of the store in the month of June 2010.

```
SELECT Sum(Total_Amt) TurnOver from Transaction
where month(Trans_Date) = 06 and year(Trans_Date) = 2010;
```

- Get the total sale amount paid through different payment medium separately

```
SELECT Payment_Medium, Sum(Amt) Amout_paid from Payment_Mode
where Trans_Number like 'SA%'
group by Payment_Medium
```

Payment_Medium	Amout_paid
Cash	139.94000244140625
Credit Card	258.25
Debit Card	65.16000366210938

- Get the Item details and number of quantities sold as of now in the store.

```
Select P.Product_Id, P.Product_Name, IFNUll(sum(TSI.Product_Qty),0)-
IFNull(sum(TRI.Product_Qty),0) Quantity_Sold
from Product P left join  Transaction_Sale_Item TSI on
P.Product_Id=TSI.Product_Id
left join Transaction_Return_Item TRI on TSI.Product_Id=TRI.Product_Id
group by P.Product_Id
```

Product_Id	Product_Name	Quantity_Sold
Item000000001	Tomato	9
Item000000002	Chocolate milk	3
Item000000003	Orange Juice	0
Item000000004	PS4 Controller	0
Item000000005	Oreo biscuts	5
Item000000006	Wall nuts	1
Item000000007	Maggi noodles	0
Item000000008	Ice	0
Item000000009	Iphone 11 case	1
Item000000010	Eggs	3
Item000000011	Brown rice	4
Item000000012	Sanitizer	3
Item000000013	Colgate Max	1
Item000000014	Tresseme	0
Item000000015	Whole Chicken	4
Item000000016	Yummy Donuts	0
Item000000017	Plum cake	0

- Get Customer details who is most frequently visited the store.

```
Select C.Cust_Id, C.Cust_FName, C.Cust_LName, C.Cust_Phone from
Customer C inner join Transaction T on C.Cust_Id=T.Cust_Id
where T.Cust_Id = (SELECT Cust_Id
FROM Transaction
GROUP BY Cust_Id
ORDER BY COUNT(Cust_Id) DESC
LIMIT 1 ) Group by C.Cust_Id
```

IV. CONCLUSION

In this project we have developed a database system for a Retail store which will store the data related to Sales, Employees, and Inventories of the store. This database system is an adequate solution for tedious tasks performed in a retail store which are simplified and secured with this database.

Though we have not developed a GUI to do data manipulation operations, we have designed and developed a database schema by using which basic operations of a Retail store are performed. We have created necessary tables and sample data through backend in MySQL workbench.

We hope this project can be used as a prototype for a Retail Store database management system.

V. REFERENCES

1. https://www.sqlservertutorial.net/sql-server-sample-database/
2. https://docs.microsoft.com/en-us/dynamicsax-2012/appuser-itpro/create-databases-for-retail-stores-retail-essentials
3. https://stackoverflow.com/