

# Project DataBase Management

## **The Idea ( Pharmacy Inventory Management System ) :**

Preserving all information related to the pharmacy so that it is not lost or causing errors, saving customer, supplier and employee data for easy access to it whenever we need it, and recording all medications entering or leaving the pharmacy to know the quantities available and those that are about to run out.

### **Problems :**

- Medical mistake in medications due to randomness
- Delayed arrival of medications to the patient
- Lack of patient information record
- Placing medications in the wrong departments
- Lack of knowledge of available medications

### **Solution :**

Helping pharmacies access any data related to medical and nutritional medicines, as well as accessing patient data in a smooth and easy way. This helps pharmacy employees reduce the rate of medical mistake and prevent any patient from being given the wrong medication

### **Member Groub :**

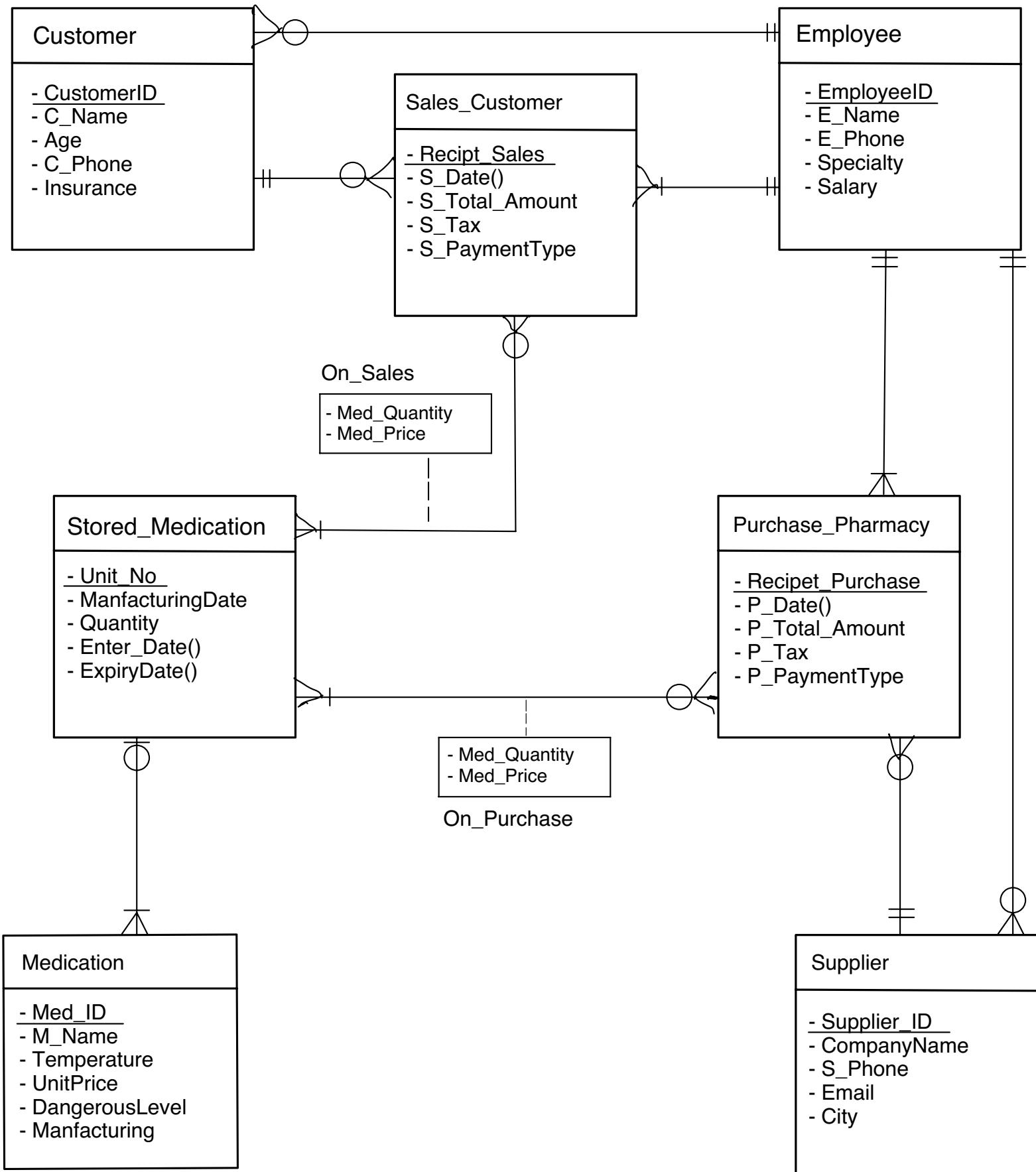
**The Leader : Yousef Mohammed Raad 2240032**

**Abdullah Ahmed Bazuhair 2240109**

**Abdullah Fayez Alshehri 2240044**

**Osama Ali Alghamdi 2240206**

# ER-Model



# Relation-Model



## Normalization

### UNF :

((EmployeeID# , E\_Name , E\_Phone , Specialty , Salary ( ( receipt\_sales # , S\_month , S\_day , S\_year ,  
S\_total\_amount , S\_tax , S\_paymentType , med\_quantity , med\_price)( ( CustomerID# , C\_Name , Age ,  
C\_Phone , Insurance)( SupplierID# , CompanyName , S\_Phone , Email ,City)( Receipt\_purchase#,  
P\_month, P\_day, year , P\_total\_amount , P\_tax ,P\_paymentType , med\_quantity , med\_price unit\_No#,  
manufacturingDate , quantity , enter\_Date , expire\_Date , med\_quantity , med\_price)( med\_ID #, name ,  
temperature , unitPrice , dangerLevel , manufacturing))

### 1NF :

Employee ( EmployeeID# , E\_Name , E\_Phone , Specialty , Salary )

Sales\_Customer ( receipt\_sales # , S\_month , S\_day , S\_year , S\_total\_amount , S\_tax , S\_paymentType ,  
med\_quantity , med\_price )

Customer ( CustomerID# , C\_Name , Age , C\_Phone , Insurance )

Supplier ( SupplierID# , CompanyName , S\_Phone , Email ,City )

Purchase\_pharmacy ( Receipt\_purchase# , P\_month, P\_day, year , P\_total\_amount , P\_tax  
,P\_paymentType , med\_quantity , med\_price)

Stored\_medication ( unit\_No# , manufacturingDate , quantity , enter\_Date , expire\_Date , med\_quantity  
, med\_price)

Medication ( med\_ID # , name , temperature , unitPrice , dangerLevel , manufacturing)

## 2NF :

Employee ( EmployeeID# , E\_Name , E\_Phone , Specialty , Salary )

Sales\_Customer ( receipt\_sales# , S\_month , S\_day , S\_year , S\_total\_amount , S\_tax , S\_paymentType , med\_quantity , med\_price )

Customer ( CustomerID# , C\_Name , Age , C\_Phone , Insurance )

Supplier ( SupplierID# , CompanyName , S\_Phone , Email , City )

Purchase\_pharmacy ( Receipt\_purchase# , P\_month , P\_day , year , P\_total\_amount , P\_tax , P\_paymentType , med\_quantity , med\_price )

Stored\_medication ( unit\_No# , manufacturingDate , quantity , enter\_Date , expire\_Date , med\_quantity , med\_price )

Medication ( med\_ID # , name , temperature , unitPrice , dangerLevel , manufacturing )

## 3NF :

Employee ( EmployeeID# , E\_Name , E\_Phone , Specialty , Salary )

Sales\_Customer ( receipt\_sales# , S\_month , S\_day , S\_year , S\_total\_amount , S\_tax , S\_paymentType , med\_quantity , med\_price )

Customer ( CustomerID# , C\_Name , Age , C\_Phone , Insurance )

Supplier ( SupplierID# , CompanyName , S\_Phone , Email , City )

Purchase\_pharmacy ( Receipt\_purchase# , P\_month , P\_day , year , P\_total\_amount , P\_tax , P\_paymentType , med\_quantity , med\_price )

Medication ( med\_ID # , name , temperature , unitPrice , dangerLevel , manufacturing )

Stored\_medication ( unit\_No# , manufacturingDate , enter\_Date , expire\_Date )

Stored\_medication\_inventory ( unit\_No# , med\_quantity , med\_price )

## Funcation Dependencies

EmployeeID → E\_Name , E\_Phone , Specialty , Salary

Receipt\_Sales → S\_month , S\_day , S\_year , S\_total\_amount ,  
S\_paymentType , S\_tax

CustomerID → C\_Name , Age , C\_Phone , Insurance

SupplierID → CompanyName , S\_Phone , Email , City

Receipt\_Purchase → P\_month , P\_day , P\_year , P\_total\_amount ,  
P\_paymentType , P\_tax

med\_ID → M\_Name , temperature , unitPrice , dangerLevel ,  
manufacturing

unit\_No → manufacturingDate , enter\_Date , expire\_Date

## Create The Tables :

Our Program Contains 7 Tables (Before Normalized ) :

- Employee
- Customer
- Sales\_Customer
- Supplier
- Purchase\_Pharmacy
- Medication
- Stored\_Medication

Our Program Contains 8 Tables (After Normalized ) :

- Employee
- Customer
- Sales\_Customer
- Supplier
- Purchase\_Pharmacy
- Medication
- Stored\_Medication
- Stored\_Medication Inventory

## Create Table : Employee

The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes Clear, Find, Actions (dropdown), Save, and Run buttons.
- Code Area:** Displays the SQL code for creating the Employee table. The code includes column definitions (ID, Name, Phone, Specialty, Salary) and primary key constraints. It also contains four comment lines providing employee names and IDs.
- Status Bar:** Shows the message "Table created." indicating the successful execution of the query.

```
SQL Worksheet

1 v CREATE TABLE Employee (
2     ID INT PRIMARY KEY,
3     Name VARCHAR(100),
4     Phone VARCHAR(20),
5     Specialty VARCHAR(50),
6     Salary DECIMAL(10, 2)
7 );
8
9 -- Abdullah Ahmed Bazuhair 2240109
10 -- Osama ali alghamdi 2240206
11 -- Abdullah Fayed Alshehri 2240044
12 -- Yousef Mohammed Raad 2240032
13

Table created.
```

## Create Table : Customer

The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes Clear, Find, Actions (dropdown), Save, and Run buttons.
- Code Area:** Displays the SQL code for creating the Customer table. The code includes column definitions (CustomerID, Name, Age, Phone, Insurance) and primary key constraints. It also contains four comment lines providing customer names and IDs.
- Status Bar:** Shows the message "Table created." indicating the successful execution of the query.

```
SQL Worksheet

1 v CREATE TABLE Customer (
2     CustomerID INT PRIMARY KEY,
3     Name VARCHAR(100),
4     Age INT,
5     Phone VARCHAR(20),
6     Insurance VARCHAR(100)
7 );
8
9 -- Abdullah Ahmed Bazuhair 2240109
10 -- Osama ali alghamdi 2240206
11 -- Abdullah Fayed Alshehri 2240044
12 -- Yousef Mohammed Raad 2240032

Table created.
```

## Create Table : Sales\_Customer

The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes buttons for Clear, Find, Actions (dropdown), Save, and Run.
- Code Area:** Displays the SQL code for creating the Sales\_Customer table. The code includes primary key constraints and several comments at the bottom.

```
1 v CREATE TABLE Sales_Customer (
2     receipt_sales INT PRIMARY KEY,
3     month INT,
4     day INT,
5     year INT,
6     total_amount DECIMAL(10, 2),
7     tax DECIMAL(6, 2),
8     paymentType VARCHAR(50),
9     med_quantity INT,
10    med_price DECIMAL(8, 2)
11 );
12 -- Abdullah Ahmed Bazuhair 2240109
13 -- Osama ali alghamdi 2240206
14 -- Abdullah Fayed Alshehri 2240044
15 -- Yousef Mohammed Raad 2240032
```
- Output Area:** Shows the message "Table created." indicating the successful execution of the query.

## Create Table : Supplier

The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes buttons for Clear, Find, Actions (dropdown), Save, and Run.
- Code Area:** Displays the SQL code for creating the Supplier table. Similar to the Sales\_Customer table, it includes a primary key constraint and comments at the bottom.

```
1 v CREATE TABLE Supplier (
2     SupplierID INT PRIMARY KEY,
3     CompanyName VARCHAR(100),
4     Phone VARCHAR(20),
5     Email VARCHAR(100),
6     City VARCHAR(50)
7 );
8 -- Abdullah Ahmed Bazuhair 2240109
9 -- Osama ali alghamdi 2240206
10 -- Abdullah Fayed Alshehri 2240044
11 -- Yousef Mohammed Raad 2240032
```
- Output Area:** Shows the message "Table created." indicating the successful execution of the query.

## Create Table : Purchase\_Pharmacy

The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes Clear, Find, Actions (dropdown), Save, and Run buttons.
- Code Area:** Contains the SQL code for creating the Purchase\_Pharmacy table. The code includes comments for four users: Abdullah Ahmed Bazuhair (2240109), Osama ali alghamdi (2240206), Abdullah Fayed Alshehri (2240044), and Yousef Mohammed Raad (2240032).
- Output Area:** Displays the message "Table created."

```
1 v CREATE TABLE Purchase_pharmacy (
2     Receipt_purchase INT PRIMARY KEY,
3     month INT,
4     day INT,
5     year INT,
6     total_amount DECIMAL(10, 2),
7     tax DECIMAL(6, 2),
8     paymentType VARCHAR(50),
9     med_quantity INT,
10    med_price DECIMAL(8, 2)
11 );
12 -- Abdullah Ahmed Bazuhair 2240109
13 -- Osama ali alghamdi 2240206
14 -- Abdullah Fayed Alshehri 2240044
15 -- Yousef Mohammed Raad 2240032

Table created.
```

## Create Table : Medication

The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes Clear, Find, Actions (dropdown), Save, and Run buttons.
- Code Area:** Contains the SQL code for creating the Medication table. The code includes comments for four users: Abdullah Ahmed Bazuhair (2240109), Osama ali alghamdi (2240206), Abdullah Fayed Alshehri (2240044), and Yousef Mohammed Raad (2240032).
- Output Area:** Displays the message "Table created."

```
1 v CREATE TABLE Medication (
2     med_ID INT PRIMARY KEY,
3     name VARCHAR(100),
4     temperature VARCHAR(20),
5     unitPrice DECIMAL(10, 2),
6     dangerLevel VARCHAR(50),
7     manufacturing DATE
8 );
9 -- Abdullah Ahmed Bazuhair 2240109
10 -- Osama ali alghamdi 2240206
11 -- Abdullah Fayed Alshehri 2240044
12 -- Yousef Mohammed Raad 2240032

Table created.
```

## Create Table : Stored\_Medication

The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes buttons for Clear, Find, Actions, Save, and Run.
- Code Area:** Displays the SQL code for creating the 'Stored\_medication' table. The code includes comments for users and dates.

```
1 v CREATE TABLE Stored_medication (
2     unit_No INT PRIMARY KEY,
3     manufacturingDate DATE,
4     enter_Date DATE,
5     expire_Date DATE
6 );
7 -- Abdullah Ahmed Bazuhair 2240109
8 -- Osama ali alghamdi 2240206
9 -- Abdullah Fayez Alshehri 2240044
10 -- Yousef Mohammed Raad 2240032
```

- Output Area:** Shows the message "Table created."

## Create Table : Stored\_Medication\_Inventory

The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes buttons for Clear, Find, Actions, Save, and Run.
- Code Area:** Displays the SQL code for creating the 'Stored\_medication\_inventory' table. The code includes comments for users and dates.

```
1 v CREATE TABLE Stored_medication_inventory (
2     unit_No INT PRIMARY KEY,
3     med_quantity INT,
4     med_price DECIMAL(8, 2)
5 );
6 -- Abdullah Ahmed Bazuhair 2240109
7 -- Osama ali alghamdi 2240206
8 -- Abdullah Fayez Alshehri 2240044
9 -- Yousef Mohammed Raad 2240032
```

- Output Area:** Shows the message "Table created."

## Our Schema ( Before Normalization ) :

The screenshot shows a schema interface with the following details:

- CUSTOMER**: Table, Status: Valid, Created 3 minutes ago.
- EMPLOYEE**: Table, Status: Valid, Created 3 minutes ago.
- MEDICATION**: Table, Status: Valid, Created 59 seconds ago.
- PURCHASE\_PHARMACY**: Table, Status: Valid, Created 21 seconds ago.
- SALES\_CUSTOMER**: Table, Status: Valid, Created 2 minutes ago.
- STORED\_MEDICATION**: Table, Status: Valid, Created 71 seconds ago.
- SUPPLIER**: Table, Status: Valid, Created 47 seconds ago.

Left sidebar options: Search Objects, Schema (My Schema), Sort By (Name), Options (Primary Objects selected), and a Reset Search button.

## Our Schema ( After Normalization ) :

The screenshot shows a schema interface with the following details:

- CUSTOMER**: Table, Status: Valid, Created 7 minutes ago.
- EMPLOYEE**: Table, Status: Valid, Created 10 minutes ago.
- MEDICATION**: Table, Status: Valid, Created 3 minutes ago.
- PURCHASE\_PHARMACY**: Table, Status: Valid, Created 4 minutes ago.
- SALES\_CUSTOMER**: Table, Status: Valid, Created 6 minutes ago.
- STORED\_MEDICATION**: Table, Status: Valid, Created 105 seconds ago.
- STORED\_MEDICATION\_INVENTORY**: Table, Status: Valid, Created 72 seconds ago.
- SUPPLIER**: Table, Status: Valid, Created 5 minutes ago.

Left sidebar options: Search Objects, Schema (My Schema), Sort By (Name), Options (Primary Objects selected), and a Reset Search button.

# Insert Values Inside The Tables :

Table One : Employee

The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes Clear, Find, Actions (dropdown), Save, and Run buttons.
- Code Area:** Displays 14 SQL INSERT statements into the Employee table, each followed by a comment indicating the user who ran the statement.

```
1 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (1, 'John Doe', '123-456-7890', 'Manager', 60000.00);
2 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (2, 'Jane Smith', '987-654-3210', 'Sales Representative', 45000.00);
3 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (3, 'David Johnson', '555-123-4567', 'IT Specialist', 55000.00);
4 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (4, 'Emily Davis', '111-222-3333', 'HR Coordinator', 50000.00);
5 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (5, 'Michael Wilson', '777-888-9999', 'Financial Analyst', 58000.00);
6 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (6, 'Sarah Brown', '333-444-5555', 'Marketing Manager', 62000.00);
7 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (7, 'Olivia Miller', '666-777-8888', 'Pharmacist', 63000.00);
8 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (8, 'James Garcia', '222-333-4444', 'Pharmacist', 59000.00);
9 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (9, 'Sophia Martinez', '999-888-7777', 'Accountant', 52000.00);
10 INSERT INTO Employee (ID, Name, Phone, Specialty, Salary) VALUES (10, 'William Taylor', '444-555-6666', 'Casher', 65000.00);
11 -- Abdullah Ahmed Bazuhair 2240109
12 -- Osama ali alghandi 2240206
13 -- Abdullah Fayed Alshehri 2240044
14 -- Yousef Mohammed Raad 2240032
```
- Output Area:** Shows the confirmation message "1 row(s) inserted." repeated 14 times, corresponding to each INSERT statement.

## Table Two : Customer

The screenshot shows a SQL Worksheet interface with the following content:

```
SQL Worksheet
Clear Find Actions Save Run

1 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (1, 'Alice Johnson', 35, '123-456-7890', 'ABC Insurance');
2 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (2, 'Bob Smith', 28, '987-654-3210', 'DEF Insurance');
3 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (3, 'Eva Martinez', 40, '555-123-4567', 'LMN Insurance');
4 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (4, 'David Lee', 45, '111-222-3333', 'LMN Insurance');
5 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (5, 'Sophie Brown', 30, '777-888-9999', 'DEF Insurance');
6 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (6, 'Chris Evans', 50, '333-444-5555', 'GHI Insurance');
7 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (7, 'Linda Garcia', 29, '666-777-8888', 'STU Insurance');
8 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (8, 'Michael Clark', 38, '222-333-4444', 'JKL Insurance');
9 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (9, 'Olivia Taylor', 42, '999-888-7777', 'JKL Insurance');
10 INSERT INTO Customer (CustomerID, Name, Age, Phone, Insurance) VALUES (10, 'Peter Harris', 33, '444-555-6666', 'ABC Insurance');

11 Abdullah Ahmed Bazuhair 2240109
12 Osama ali alghamdi 2240206
13 Abdullah Fayed Alshehri 2240044
14 Yousef Mohammed Raad 2240032
```

Output:

```
1 row(s) inserted.
```

## Table Three : Sales\_Customer

The screenshot shows a SQL Worksheet interface with the following content:

```
SQL Worksheet
Clear Find Actions Save Run

1 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1001, 11, 5, 2023, 150.00,
2 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1002, 11, 10, 2023, 200.00,
3 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1003, 11, 15, 2023, 180.00,
4 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1004, 11, 20, 2023, 300.00,
5 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1005, 11, 25, 2023, 250.00,
6 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1006, 12, 1, 2023, 270.00,
7 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1007, 12, 5, 2023, 320.00,
8 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1008, 12, 10, 2023, 190.00,
9 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1009, 12, 15, 2023, 280.00,
10 INSERT INTO Sales_Customer (receipt_sales, month, day, year, total_amount, tax, paymentType) VALUES (1010, 12, 20, 2023, 330.00,
11 -- Abdullah Ahmed Bazuhair 2240109
12 -- Osama ali alghamdi 2240206
13 -- Abdullah Fayed Alshehri 2240044
14 -- Yousef Mohammed Raad 2240032
```

Output:

```
1 row(s) inserted.
```

**Table Four : Supplier**

SQL Worksheet

Clear Find Actions Save Run

```
1 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (1, 'ABC Supplier Co.', '123-456-7890', 'abc@supplier.com')
2 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (2, 'XYZ Distributors', '987-654-3210', 'info@xyzdist.com')
3 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (3, 'PQR Enterprises', '555-123-4567', 'contact@pqrenter.com')
4 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (4, 'LMN Suppliers Inc.', '111-222-3333', 'support@lmnsuppli.com')
5 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (5, 'DEF Manufacturing', '777-888-9999', 'info@defmanufac.com')
6 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (6, 'GHI Exporters Ltd', '333-444-5555', 'sales@ghiexporter.com')
7 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (7, 'STU Import Co.', '666-777-8888', 'contact@stuimport.com')
8 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (8, 'VWX Industries', '222-333-4444', 'info@vwxindustries.com')
9 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (9, 'JKL Enterprises', '999-888-7777', 'sales@jklenter.com')
10 INSERT INTO Supplier (SupplierID, CompanyName, Phone, Email, City) VALUES (10, 'MNO Distributors', '444-555-6666', 'support@mnode.com')

11 -- Abdullah Ahmed Bazuhair 2240109
12 -- Osama ali alghamdi 2240206
13 -- Abdullah Fayed Alshehri 2240044
14 -- Yousef Mohammed Raad 2240032

1 row(s) inserted.

1 row(s) inserted.
```

**Table Five : Purchase Pharmacy**

## Table Six : Medication

The screenshot shows an SQL Worksheet interface with the following details:

- Toolbar:** Includes Clear, Find, Actions, Save, and Run buttons.
- SQL Editor:** Contains the following SQL code:

```
1 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(1, 'MedA', 25, 15.00, 'Low', TO
2 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(2, 'MedB', 28, 20.50, 'Moderate
3 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(3, 'MedC', 30, 18.75, 'Low', TO
4 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(4, 'MedD', 27, 22.00, 'High', T
5 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(5, 'MedE', 26, 17.80, 'Low', TO
6 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(6, 'MedF', 29, 19.25, 'Moderate
7 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(7, 'MedG', 24, 16.50, 'Low', TO
8 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(8, 'MedH', 28, 21.30, 'High', T
9 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(9, 'MedI', 25, 18.00, 'Moderate
10 INSERT INTO Medication (Med_ID, Name, Temperature, UnitPrice, DangerLevel, Manufacturing) VALUES(10, 'MedJ', 27, 23.00, 'High',
11 -- Abdullah Ahmed Bazuhair 2240109
12 -- Osama ali alghamdi 2240206
13 -- Abdullah Fayed Alshehri 2240044
14 -- Yousef Mohammed Raad 2240032
```
- Output Area:** Displays the results of the insertions:

```
1 row(s) inserted.
```

## Table Seven : Stored\_Medication

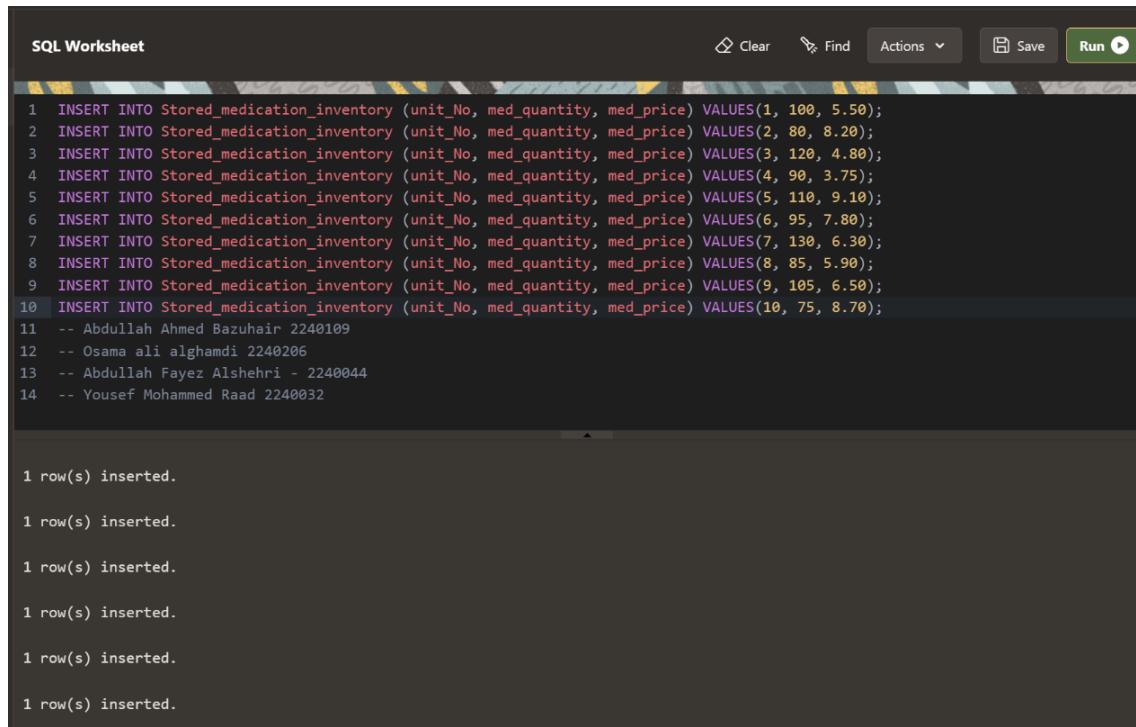
The screenshot shows an SQL Worksheet interface with the following details:

- Toolbar:** Includes Clear, Find, Actions, Save, and Run buttons.
- SQL Editor:** Contains the following SQL code:

```
1 INSERT INTO Stored_medication (Unit_no, manufacturingDate , Enter_date, Expire_date) VALUES (1, TO_DATE('2022-12-01' , 'YYYY-MM-
2 INSERT INTO Stored_medication (Unit_no, manufacturingDate , Enter_date, Expire_date) VALUES (2, TO_DATE('2023-01-05' , 'YYYY-MM-
3 INSERT INTO Stored_medication (Unit_no, manufacturingDate, Enter_date, Expire_date) VALUES (3, TO_DATE('2023-02-10' , 'YYYY-MM-
4 INSERT INTO Stored_medication (Unit_no, manufacturingDate, Enter_date, Expire_date) VALUES (4, TO_DATE('2023-03-15' , 'YYYY-MM-
5 INSERT INTO Stored_medication (Unit_no, manufacturingDate, Enter_date, Expire_date) VALUES (5, TO_DATE('2023-04-20' , 'YYYY-MM-
6 INSERT INTO Stored_medication (Unit_no, manufacturingDate, Enter_date, Expire_date) VALUES (6, TO_DATE('2023-05-25' , 'YYYY-MM-
7 INSERT INTO Stored_medication (Unit_no, manufacturingDate, Enter_date, Expire_date) VALUES (7, TO_DATE('2022-11-30' , 'YYYY-MM-
8 INSERT INTO Stored_medication (Unit_no, manufacturingDate, Enter_date, Expire_date) VALUES (8, TO_DATE('2023-06-10' , 'YYYY-MM-
9 INSERT INTO Stored_medication (Unit_no, manufacturingDate, Enter_date, Expire_date) VALUES (9, TO_DATE('2023-07-15' , 'YYYY-MM-
10 INSERT INTO Stored_medication (Unit_no, manufacturingDate, Enter_date, Expire_date) VALUES (10,TO_DATE('2023-08-20' , 'YYYY-MM-
```
- Output Area:** Displays the results of the insertions:

```
1 row(s) inserted.
```

## Table Eight : Stored\_Medication\_Inventory



The screenshot shows a SQL Worksheet interface with the following details:

- Toolbar:** Includes "Clear", "Find", "Actions", "Save", and a "Run" button.
- SQL Script:** The code inserts 10 rows into the "Stored\_medications\_inventory" table. The rows have unit numbers from 1 to 10, med quantities from 75 to 130, and med prices from 3.75 to 9.10. The last four rows are comments about users and their IDs.

```
1 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(1, 100, 5.50);
2 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(2, 80, 8.20);
3 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(3, 120, 4.80);
4 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(4, 90, 3.75);
5 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(5, 110, 9.10);
6 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(6, 95, 7.80);
7 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(7, 130, 6.30);
8 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(8, 85, 5.90);
9 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(9, 105, 6.50);
10 INSERT INTO Stored_medications_inventory (unit_No, med_quantity, med_price) VALUES(10, 75, 8.70);
11 -- Abdullah Ahmed Bazuhair 2240109
12 -- Osama ali alghamdi 2240206
13 -- Abdullah Fayez Alsbehri - 2240044
14 -- Yousef Mohammed Raad 2240032
```

- Output:** Shows 10 rows inserted successfully, each followed by a confirmation message: "1 row(s) inserted."

# Query

## 1) ORDER BY

```
1 v SELECT (Salary) ,Specialty
2 FROM Employee
3 ORDER BY Salary
4
5
6 -- Yousef Mohammed Raad - 2240032
7 -- osama ali alghamdi - 2240206
8 -- Abdullah Fayed Alshehri - 2240044
9 -- Abdullah Ahmed Bazuhair - 2240109
```

SALARY	SPECIALTY
45000	Sales Representative
50000	HR Coordinator
52000	Accountant
55000	IT Specialist
58000	Financial Analyst
59000	Pharmacist
60000	Manager
62000	Marketing Manager
63000	Pharmacist
65000	Casher

## 2) GROUP BY , HAVING ,COUNT

```
1 v SELECT COUNT(CustomerID) ,  Insurance
2 FROM Customer
3 GROUP BY Insurance
4 HAVING COUNT(CustomerID) > 1 ;
5
6
7
8 -- Yousef Mohammed Raad - 2240032
9 -- osama ali alghamdi - 2240206
10 -- Abdullah Fayed Alshehri - 2240044
11 -- Abdullah Ahmed Bazuhair - 2240109
```

COUNT(CUSTOMERID)	INSURANCE
2	DEF Insurance
2	JKL Insurance
2	LMN Insurance
2	ABC Insurance

[Download CSV](#)

4 rows selected.

### 3) WHERE

```
1 v SELECT (Receipt_purchase) , total_amount , med_quantity , med_price , paymentType
2 FROM Purchase_Pharmacy
3 WHERE paymentType = 'credit Card'
4
5
6
7
8
9
10
11 -- Yousef Mohammed Raad - 2240032
12 -- osama ali alghamdi - 2240206
13 -- Abdullah Fayed Alshehri - 2240044
14 -- Abdullah Ahmed Bazuhair - 2240109
```

RECEIPT_PURCHASE	TOTAL_AMOUNT	MED_QUANTITY	MED_PRICE	PAYMENTTYPE
2001	350.25	8	42.5	Credit Card
2005	180.5	4	28.5	Credit Card
2010	330	9	36.75	Credit Card
2007	310	8	38.75	Credit Card

[Download CSV](#)

4 rows selected.

### 4,5) Aggregate (COUNT , MAX , MIN , AVG ,SUM)

```
1 w SELECT COUNT(Receipt_sales) AS Receipt , MAX(total_amount) AS High_Earnings , MIN(total_amount) AS Least_Earnings , AVG(total_amount) AS Average_Earnings , SUM(total_amount) AS Total_Earnings
2 FROM Sales_Customer;
3
4
5
6 -- Yousef Mohammed Raad - 2240032
7 -- osama ali alghamdi - 2240206
8 -- Abdullah Fayed Alshehri - 2240044
9 -- Abdullah Ahmed Bazuhair - 2240109
```

[Download CSV](#)

#### SQL Worksheet

```
1 v SELECT AVG(Salary) AS Average_Salary ,
2      SUM(Salary) AS SUM_Salary
3 FROM Employee
4
5
6
7 -- Yousef Mohammed Raad - 2240032
8 -- osama ali alghamdi - 2240206
9 -- Abdullah Fayed Alshehri - 2240044
10 -- Abdullah Ahmed Bazuhair - 2240109
```

AVERAGE_SALARY	SUM_SALARY
56900	569000

[Download CSV](#)

## 6) Join query

```
1 v SELECT MANUFACTURINGDATE , Expire_date , MED_QUANTITY
2 FROM Stored_Medication
3 JOIN Stored_medication_inventory
4 on Stored_Medication.Unit_no = Stored_medication_inventory.unit_no ;
5 |
6
7
8 -- Yousef Mohammed Raad - 2240032
9 -- osama ali alghamdi - 2240206
10 -- Abdullah Fayed Alshehri - 2240044
11 -- Abdullah Ahmed Bazuhair - 2240109
```

MANUFACTURINGDATE	EXPIRE_DATE	MED_QUANTITY
01-DEC-22	01-JAN-24	100
05-JAN-23	01-FEB-24	80
10-FEB-23	01-MAR-24	120
15-MAR-23	01-APR-24	90
20-APR-23	01-MAY-24	110
25-MAY-23	01-JUN-24	95

[Download CSV](#)

6 rows selected.

## 7)

Main Query: which employees have salary greater than "John Doe"

Subquery: what is "John Doe" salary.

```
1 v SELECT Name, Salary
2 FROM Employee
3 WHERE Salary >
4 (SELECT Salary
5 FROM Employee
6 WHERE Name = 'John Doe')
7
8
9 -- Yousef Mohammed Raad - 2240032
10 -- osama ali alghamdi - 2240206
11 -- Abdullah Fayed Alshehri - 2240044
12 -- Abdullah Ahmed Bazuhair - 2240109
```

NAME	SALARY
Sarah Brown	62000
Olivia Miller	63000
William Taylor	65000

[Download CSV](#)

3 rows selected.

## Procedure :

A select procedure to see if the customer age achieve the condition by the id :

The screenshot shows an SQL Worksheet interface with the following code:

```
1 v CREATE OR REPLACE PROCEDURE Customer_inf(CID number, CAge number)
2 AS
3 CURSOR excute_pro IS
4 Select CustomerID,Name ,Age ,Phone,Insurance
5 from Customer where Age>= CAge and CustomerID= CID;
6 v BEGIN
7 FOR c_change IN excute_pro LOOP
8 dbms_output.put_line(c_change.CustomerID||' '|| c_change.Name||' '|| c_change.Age ||' '|| c_change.Phone );
9 END LOOP;
10 END Customer_inf;
```

Below the code, a message indicates the procedure was created:

Procedure created.

Execute the procedure

The screenshot shows an SQL Worksheet interface with the following code:

```
1 v execute Customer_inf(3,18)
2 execute Customer_inf(1,60)
3
```

Below the code, the output shows the statements processed and the results of the executions:

Statement processed.  
3 Eva Martinez 40 555-123-4567  
Statement processed.

## An update procedure to change the phone number

The screenshot shows an SQL Worksheet interface with a toolbar at the top featuring 'Clear', 'Find', 'Actions', 'Save', and a prominent green 'Run' button. The code area contains the following T-SQL script:

```
1 v Create or replace procedure updateCustomerphone
2 (
3     c_id Customer.CustomerID%TYPE,
4     c_Phone Customer.Phone%TYPE
5 )
6 AS
7 BEGIN
8     UPDATE Customer
9     SET Phone = c_Phone
10    WHERE CustomerID = c_id;
11    COMMIT;
12 END updateCustomerphone;
```

Below the code, a message 'Procedure created.' is displayed.

## Before

The screenshot shows an SQL Worksheet interface with a toolbar at the top featuring 'Clear', 'Find', 'Actions', 'Save', and a green 'Run' button. The code area contains the following T-SQL script:

```
1 SELECT * FROM Customer ;
```

Below the code, a table is displayed with the following data:

CUSTOMERID	NAME	AGE	PHONE	INSURANCE
1	Alice Johnson	35	123-456-7890	ABC Insurance

## Execute the procedure

The screenshot shows an SQL Worksheet interface with a toolbar at the top featuring 'Clear', 'Find', 'Actions', 'Save', and a green 'Run' button. The code area contains the following T-SQL script:

```
1 exec updateCustomerphone(1,'111-111-1111');
```

Below the code, a message 'Statement processed.' is displayed.

## After

The screenshot shows an SQL Worksheet interface with a toolbar at the top featuring 'Clear', 'Find', 'Actions', 'Save', and a green 'Run' button. The code area contains the following T-SQL script:

```
1 select *from Customer ;
```

Below the code, a table is displayed with the following data:

CUSTOMERID	NAME	AGE	PHONE	INSURANCE
1	Alice Johnson	35	111-111-1111	ABC Insurance