Database Management System

18CSC303J

Project Title:
Restaurant Management System

Group Members:

Devahuti Gogoi -	RA1811031010004
Shubham Mehta -	RA1811031010009
Sanskriti Shrivastava -	RA1811031010048

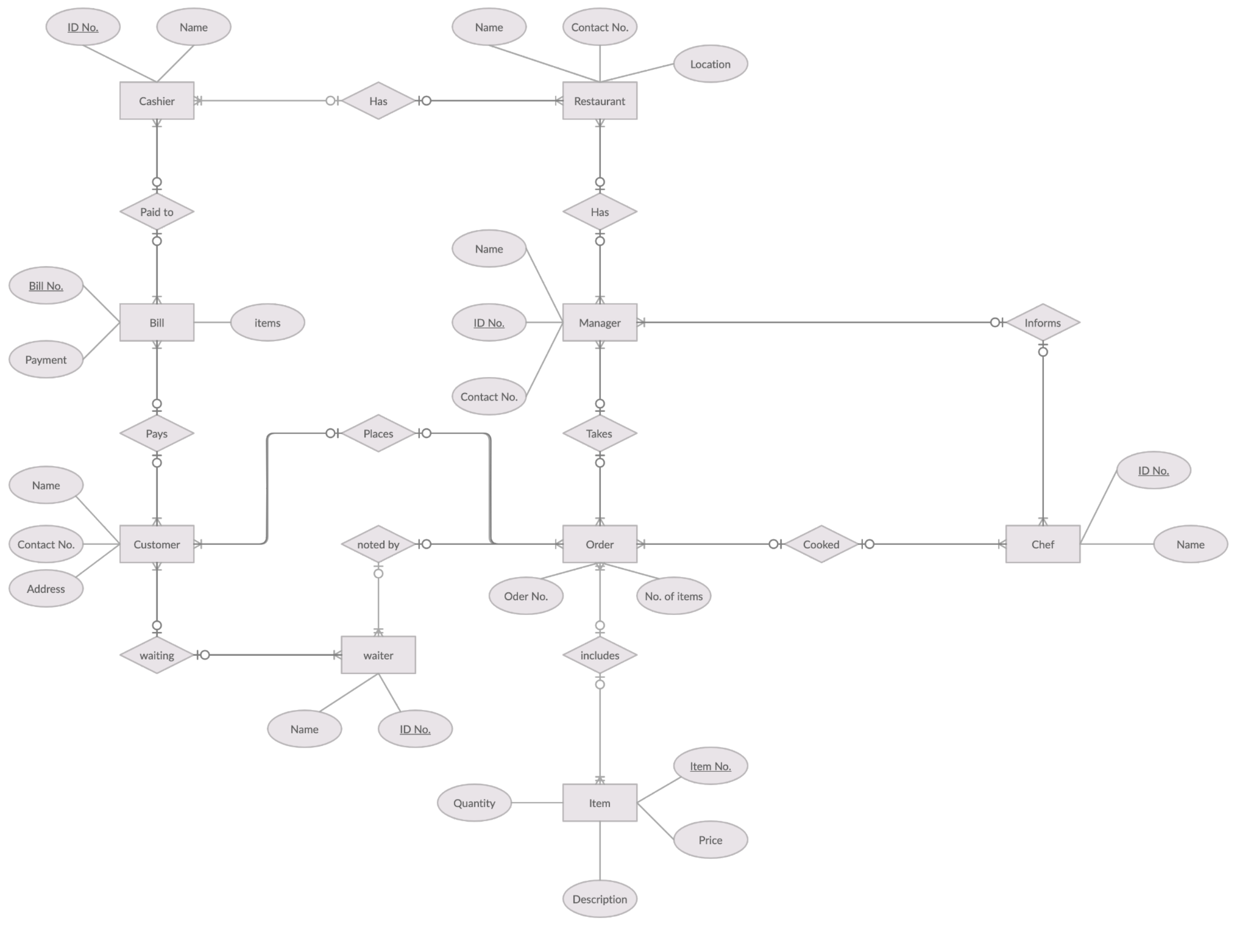
ABSTRACT

Our project topic is ‘Restaurant management system’. In this, we have created tables to manage the chef, the customer, the bill, the menu items, the restaurant, etc for an efficient system in a restaurant. We can update, delete and insert into these tables as and when required. In this project, we have used our knowledge in SQL and PL/SQL to execute various queries to show the working of this system. This system is to automate day to day activity of a restaurant. A restaurant is a kind of business that serves people all over the world ready-made food. This system is to provide service facilities to the restaurant and also to the customer. This restaurant management system can be used by employees in a restaurant to handle the clients and their orders.The main point of this system is to help restaurant administrators manage the restaurant business and help customers to gain their satisfaction. As the owner, he/she must be able to update the tables according to the restaurant’s needs depending on the demand and employment. Salaries can be changed, menu items can be altered, etc. For the customer, he/she can decide what to order according to their likings and affordability and can contact the manager or waiter if more assistance is needed.

REQUIREMENT SPECIFICATION

Basic knowledge of SQL, PL/SQL, SQL plus application.
Schemas: Attributes
Restaurant : Name, Location, Contact no.
Bill: Bill_No, Items, Total_payment
Menu_Items: Item_No, Name, Description, Quantity, Price
Customer: Cust_name, contact, Address
Manager: Manager_Id, M_Name, Contact, Address, Salary
Waiter: Waiter_Id, Waiter_name, Contact, Salary
Cashier: Cashier_Id, Cash_name, contact,Salary
Chef: Chef_Id,Chef_name, Contact, Address, Salary
Order: Order_No, Name

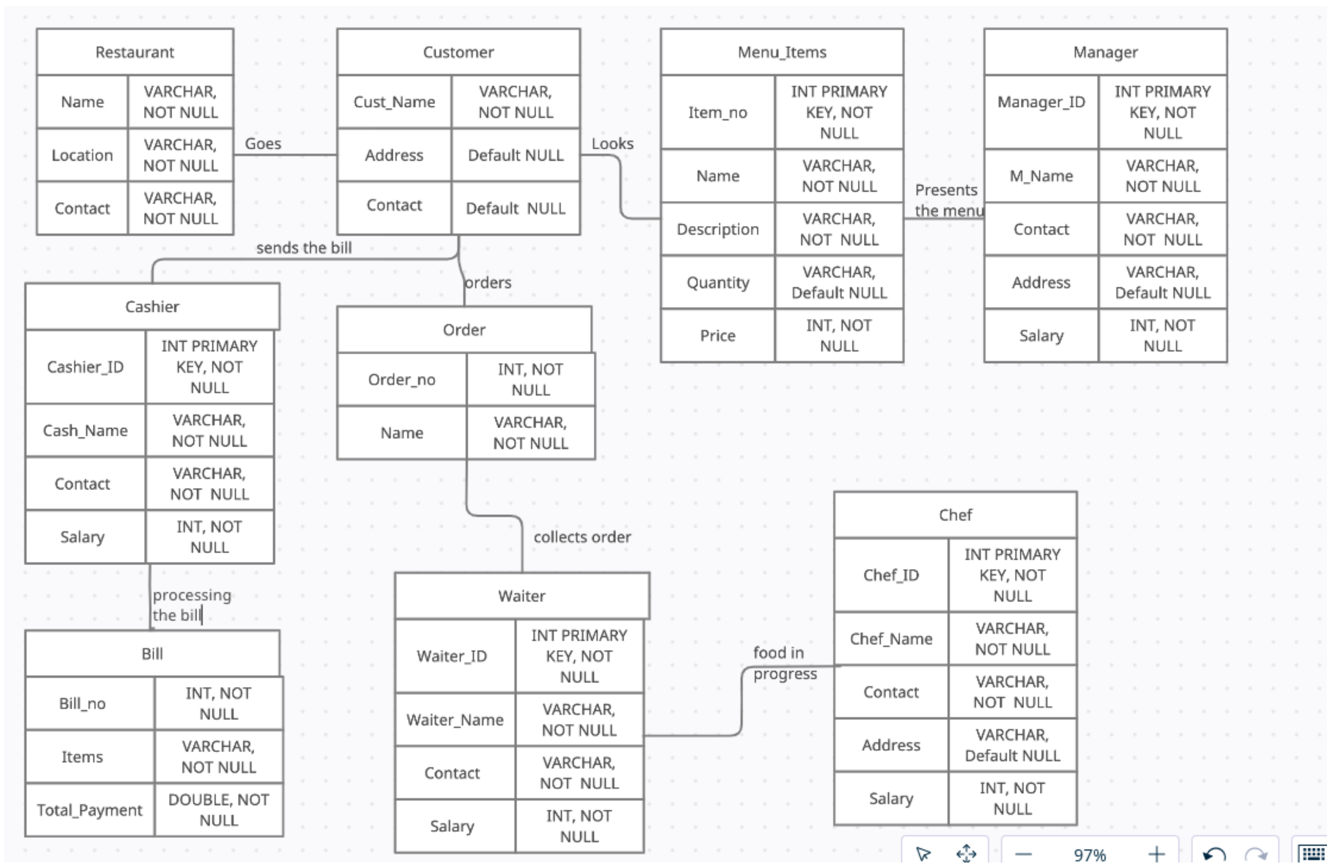
ER DIAGRAM



Effective restaurant management balances many different targets and processes to create a seamless operation. Food and labour costs, inventory tracking, staff training, food production, customer service, and marketing are part of daily restaurant management.
The above ER Diagram depicts around 9 main components to manage a restaurant namely- Cashier, Customer, Manager, Waiter, Chef, Bill, Items, Order & Restaurant.

The Restaurant has the attributes Name, Location and Contact no. ‘Bill’ consists of the columns Bill_no, items ordered and the total payment that the customer is due with. Menu_Items is the table name that consists of item no, the Name of the item, its description, quantity and its price. The table ‘Customer’ has Cust_name, contact and address as its attributes in the table. ‘Order’ is similar to Menu Items but consists of an ID which is the order no. and name. The rest of the tables Manager, Waiter, Cashier and Chef contain the columns name, contact, address and salary.

TABLE DESIGN WITH INTEGRITY CONSTRAINTS



Restaurant	
Name	VARCHAR, NOT NULL
Location	VARCHAR, NOT NULL
Contact	VARCHAR, NOT NULL

Customer	
Cust_Name	VARCHAR, NOT NULL
Address	Default NULL
Contact	Default NULL

Menu_Items	
Item_no	INT PRIMARY KEY, NOT NULL
Name	VARCHAR, NOT NULL
Description	VARCHAR, NOT NULL
Quantity	VARCHAR, Default NULL
Price	INT, NOT NULL

Manager	
Manager_ID	INT PRIMARY KEY, NOT NULL
M_Name	VARCHAR, NOT NULL
Contact	VARCHAR, NOT NULL
Address	VARCHAR, Default NULL
Salary	INT, NOT NULL

Cashier	
Cashier_ID	INT PRIMARY KEY, NOT NULL
Cash_Name	VARCHAR, NOT NULL
Contact	VARCHAR, NOT NULL
Salary	INT, NOT NULL

Order	
Order_no	INT, NOT NULL
Name	VARCHAR, NOT NULL

Chef	
Chef_ID	INT PRIMARY KEY, NOT NULL
Chef_Name	VARCHAR, NOT NULL
Contact	VARCHAR, NOT NULL
Address	VARCHAR, Default NULL
Salary	INT, NOT NULL

Bill	
Bill_no	INT, NOT NULL
Items	VARCHAR, NOT NULL
Total_Payment	DOUBLE, NOT NULL

Waiter	
Waiter_ID	INT PRIMARY KEY, NOT NULL
Waiter_Name	VARCHAR, NOT NULL
Contact	VARCHAR, NOT NULL
Salary	INT, NOT NULL

DDL QUERIES(Data Definition Language)

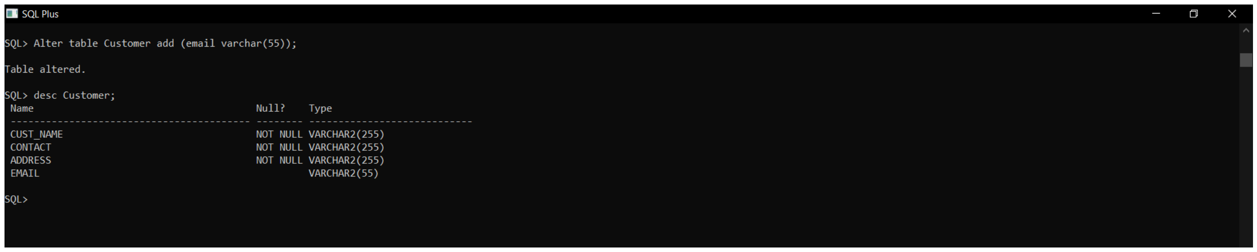
1) **Creation of tables-**

```
create table `RESTAURANT`
(
	`Name` varchar(100) NOT NULL,
	`Location` varchar(100) NOT NULL,
	`Contact No.` varchar(100) NOT NULL,
	PRIMARY KEY (`Name`)
);


create table `BILL`
(
	`Bill_No` int NOT NULL,
	`Items` varchar (200) NOT NULL,
	`Total_Payment` double NOT NULL,
	PRIMARY KEY (`Bill_No`)
);

create table Menu
(
	`Item_No` int NOT NULL,
	`Name` varchar(20) NOT NULL,
	`Description` varchar(100) NOT NULL,
	`Quantity` varchar(20) NOT NULL,
	`Price` int NOT NULL,

);


create table `CUSTOMER`
(
	`Cust_name` varchar(15) NOT NULL,
	`Contact` varchar(20) DEFAULT NULL,
	`Address` varchar(50) DEFAULT NULL,
);
```

```
create table `MANAGER`
(
	`Manager_Id` int NOT NULL,
	`M_Name` varchar(15) NOT NULL,
	`Contact` varchar(20) NOT NULL,
	`Address` varchar(30) DEFAULT NULL,
	`Salary` int NOT NULL,
	PRIMARY KEY (`Manager_Id`)
);


create table `WAITER`
(
	`Waiter_Id` int NOT NULL,
	`Waiter_name` varchar(15) NOT NULL,
	`Contact` varchar(20) NOT NULL,
	`Salary` int NOT NULL,
	PRIMARY KEY (`Waiter_Id`)
);


create table `CASHIER`
(
	`Cashier_Id` int NOT NULL,
	`Cash_name` varchar(15) NOT NULL,
	`Contact` varchar(20) NOT NULL,
	`Salary` int NOT NULL,
	PRIMARY KEY (`Cashier_Id`)
);


create table `CHEF`
(
	`CHEF_Id` int NOT NULL ,
	`Chef_name` varchar(15) NOT NULL,
	`Contact` varchar(20) NOT NULL,
	`Address` varchar(30) DEFAULT NULL,
	`Salary` int NOT NULL,
        PRIMARY KEY (`CHEF_Id`)
);



create table `ORDER`
(
        `Order_No` int NOT NULL ,
        `Name` varchar(100) NOT NULL,
        PRIMARY KEY(`Order_No')
);
```

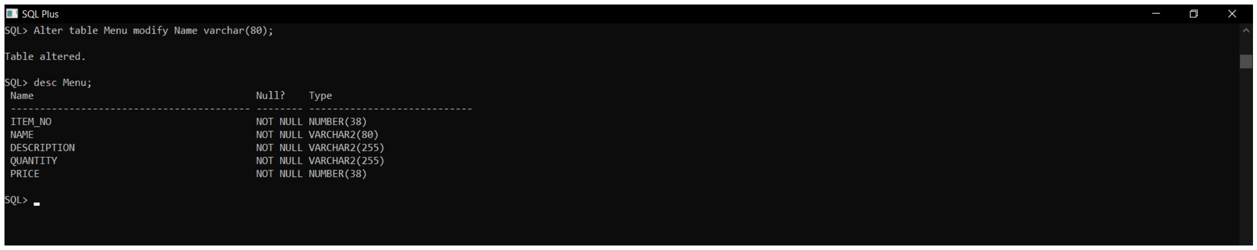
2) Alter table:

a) Add Column: Alter table CUSTOMER add (email varchar(55));



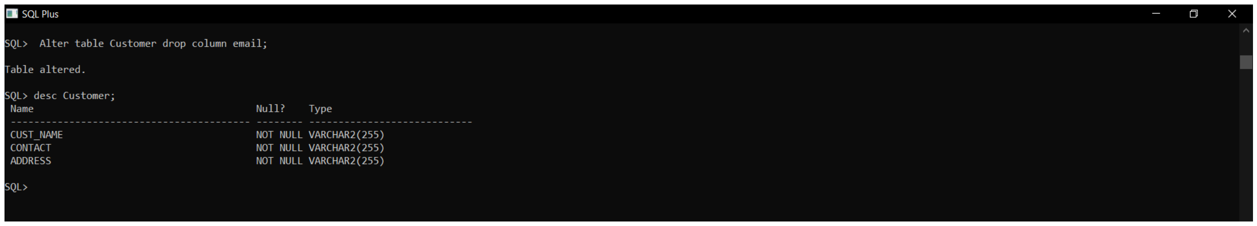
```
SQL> Alter table Customer add (email varchar(55));

Table altered.

SQL> desc Customer;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CUST_NAME                                 NOT NULL VARCHAR2(255)
 CONTACT                                   NOT NULL VARCHAR2(255)
 ADDRESS                                   NOT NULL VARCHAR2(255)
 EMAIL                                              VARCHAR2(55)

SQL>
```

b) Modify Column: Alter table Menu modify Name varchar(80);



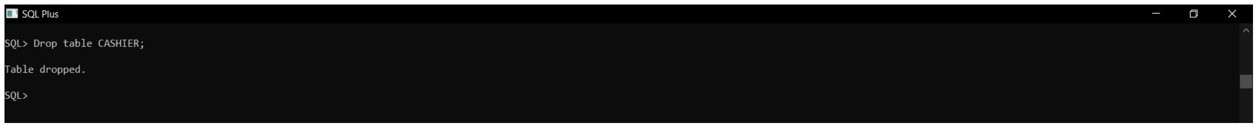
```
SQL> Alter table Menu modify Name varchar(80);

Table altered.

SQL> desc Menu;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ITEM_NO                                   NOT NULL NUMBER(38)
 NAME                                      NOT NULL VARCHAR2(80)
 DESCRIPTION                               NOT NULL VARCHAR2(255)
 QUANTITY                                  NOT NULL VARCHAR2(255)
 PRICE                                     NOT NULL NUMBER(38)

SQL>
```

c) Drop Column: Alter table Customer drop column email;



```
SQL>  Alter table Customer drop column email;

Table altered.

SQL> desc Customer;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CUST_NAME                                 NOT NULL VARCHAR2(255)
 CONTACT                                   NOT NULL VARCHAR2(255)
 ADDRESS                                   NOT NULL VARCHAR2(255)

SQL>
```

3) Drop table: Drop table CASHIER;



```
SQL> Drop table CASHIER;

Table dropped.

SQL>
```

DML QUERIES(Data Manipulation Language)

DML commands are used to modify the database. It is responsible for all form of changes in the database. The command of DML is not auto-committed that means it can't permanently save all the changes in the database. They can be rollback.

Here are some commands that come under DML:

1. INSERT
2. UPDATE
3. DELETE

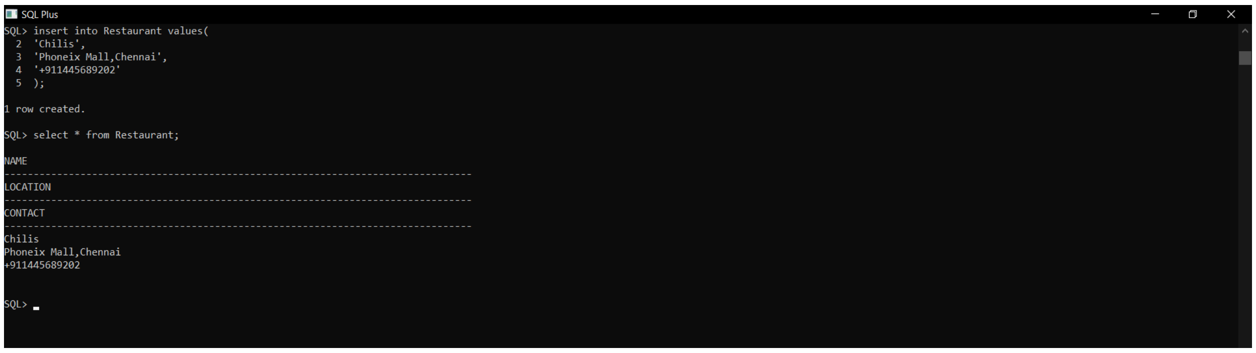
1) Insertion into the tables-

The INSERT Statement is used to insert values in a table.

Syntax:

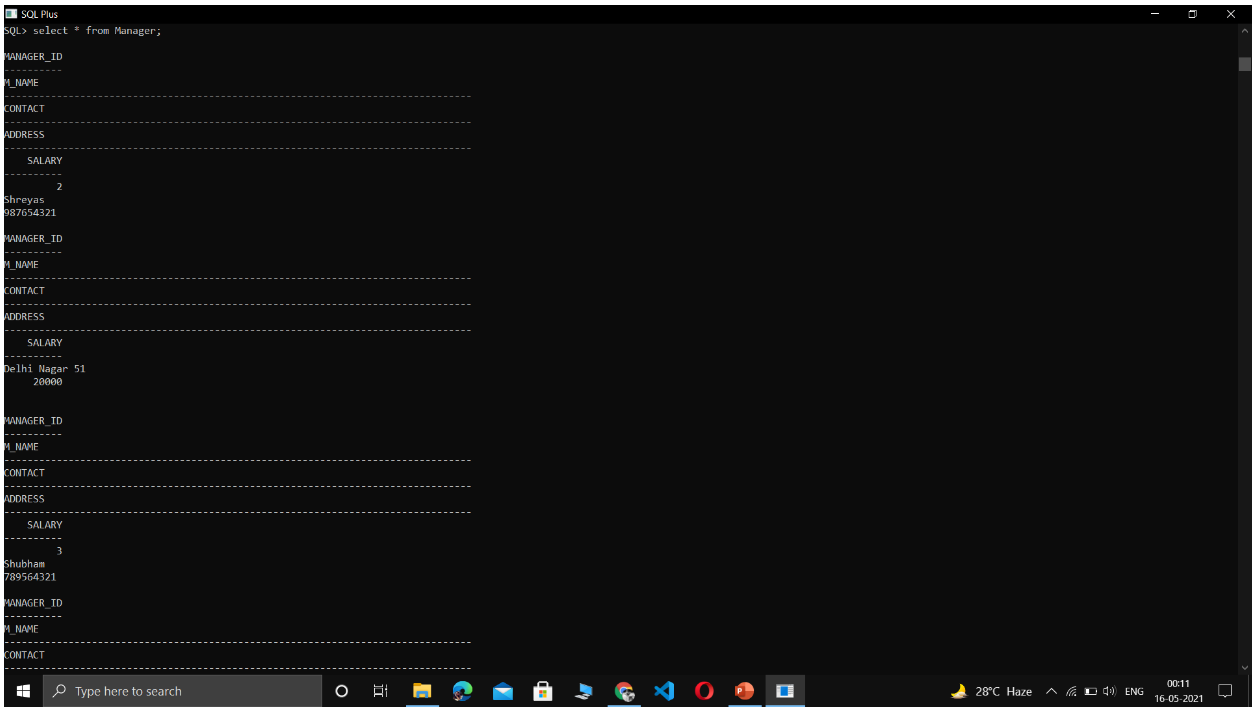
INSERT INTO table_name(“column1”, ”Column2”.....)
VALUES(value1, value2…….);

Insert into `RESTAURANT` values
("Chili's","Phoenix Mall, Chennai","+91 1449027569");



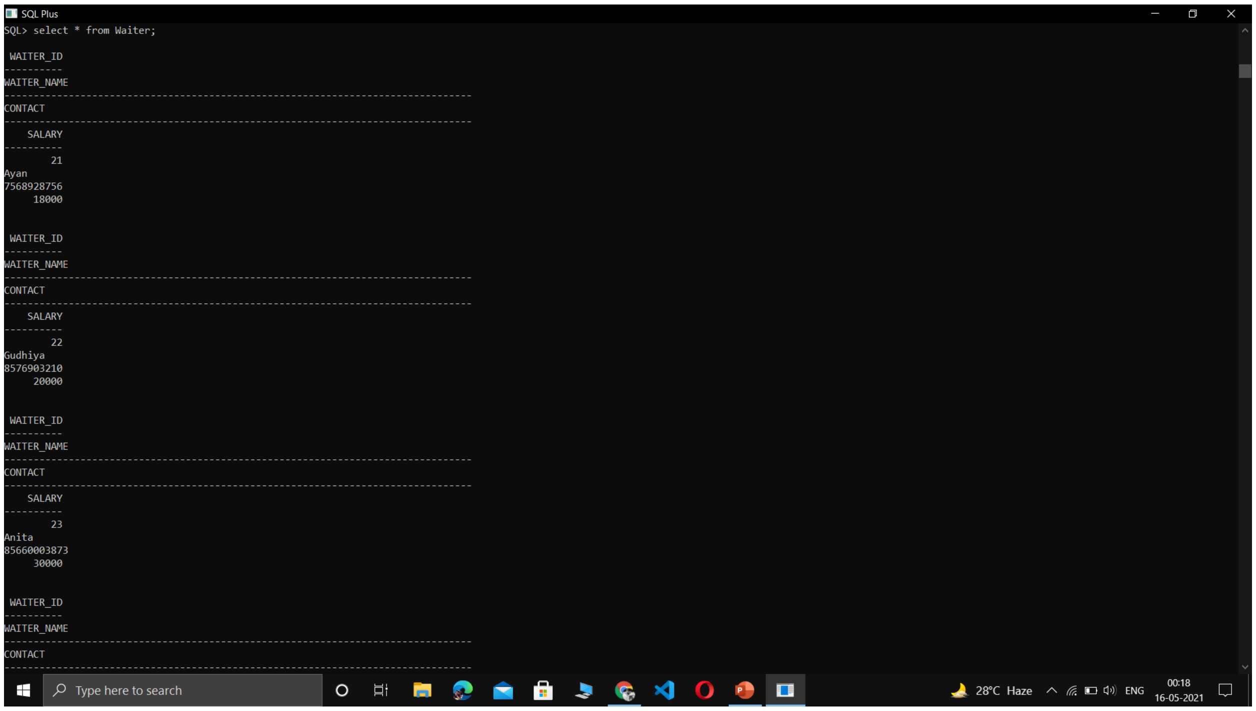
```
SQL> insert into Restaurant values(
  2  'Chilis',
  3  'Phoneix Mall,Chennai',
  4  '+911445689202'
  5  );

1 row created.

SQL> select * from Restaurant;

NAME
--------------------------------------------------------------------------------
LOCATION
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
Chilis
Phoneix Mall,Chennai
+911445689202


SQL> 
```

```
insert into `MANAGER` values
("Shreyans", "9827469182", "Delhi Nagar 51", “20000”),
("Megha", "9273648102", "Mysore Road, Nagaland", “25000”);
```



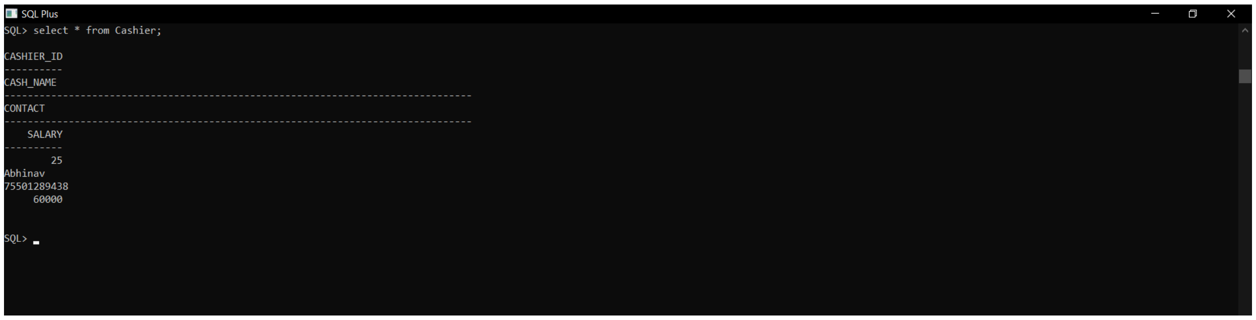
```
SQL> select * from Manager;

MANAGER_ID
----------
M_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
         2
Shreyas
987654321

MANAGER_ID
----------
M_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
Delhi Nagar 51
     20000


MANAGER_ID
----------
M_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
         3
Shubham
789564321

MANAGER_ID
----------
M_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
```

```
insert into `WAITER` values
("Ayan", "8892789078", “18000”),
("Gudiya", "7789092345", “15000”),
("Anita", "6678944566", “17000”),
("Homie", "9944558765", “12000”);
```



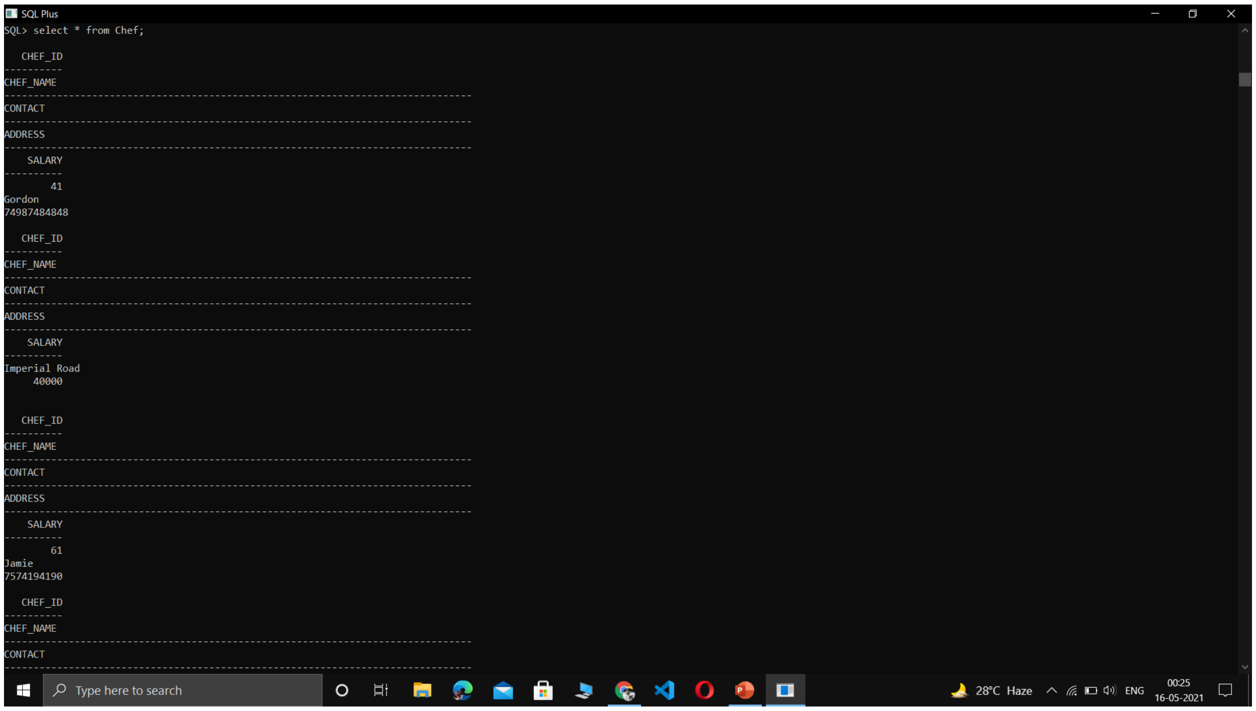
```
SQL> select * from Waiter;

 WAITER_ID
----------
WAITER_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
    SALARY
----------
        21
Ayan
7568928756
     18000


 WAITER_ID
----------
WAITER_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
    SALARY
----------
        22
Gudhiya
8576903210
     20000


 WAITER_ID
----------
WAITER_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
    SALARY
----------
        23
Anita
85660003873
     30000


 WAITER_ID
----------
WAITER_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
```

insert into `CASHIER` values
("Abhinav", "9435778823", “19000”);



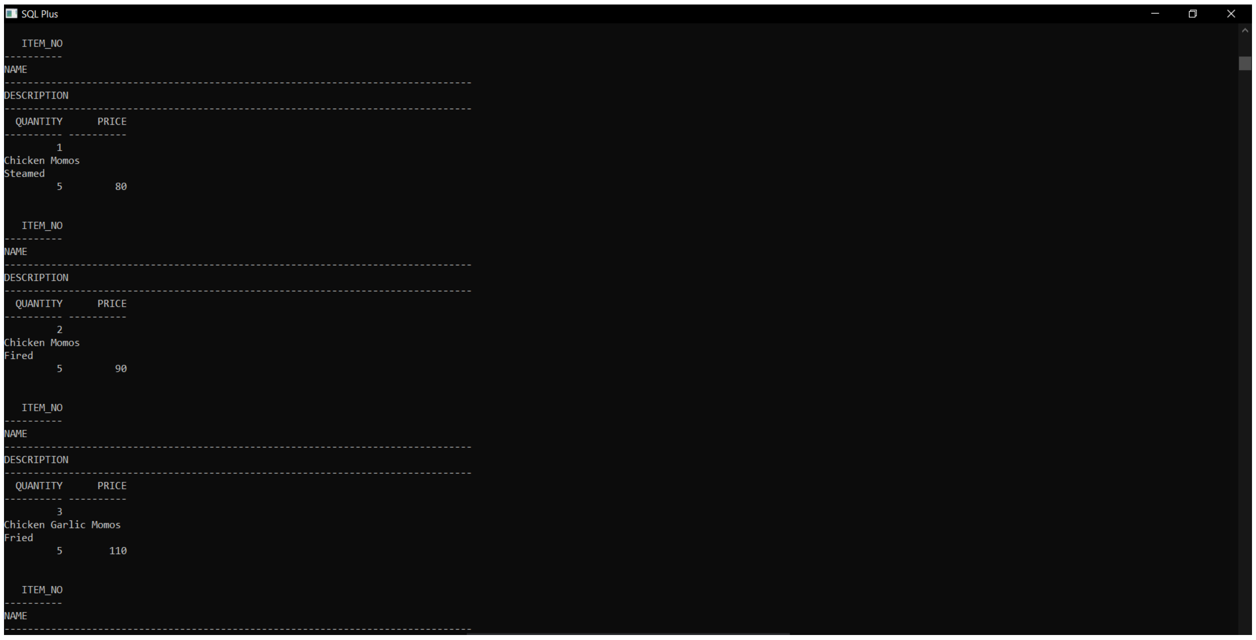
```
SQL> select * from Cashier;

CASHIER_ID
----------
CASH_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
    SALARY
----------
        25
Abhinav
75501289438
     60000


SQL> _
```

insert into `CHEF' values
("Gordon", "1190982736", "Imperial Road", “27000”),
("Jamie", "9946372837", "Burma Nagar", “25000”),
("Mulan", "9989172837", "Stremer Colony", “20000”);

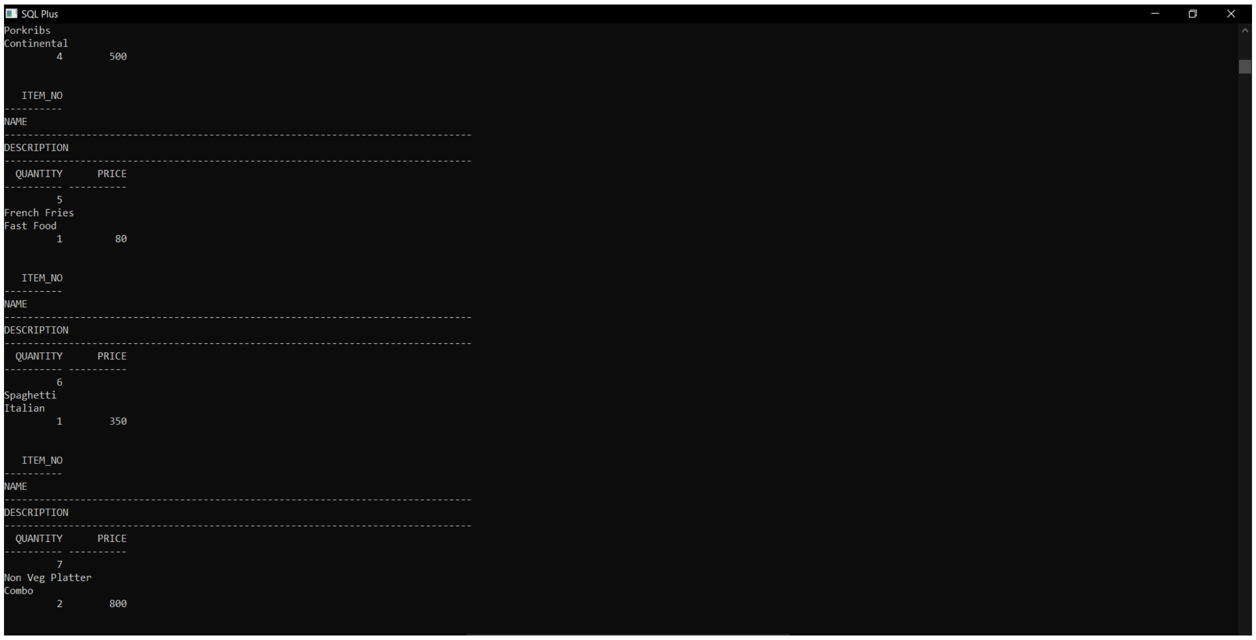


```
SQL> select * from Chef;

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
        41
Gordon
74987484848

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
Imperial Road
     40000


   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
        61
Jamie
7574194190

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
```

```
insert into `Menu` values
(1, "Chicken Momos", "Steamed", "5 Nos", "80"),
(2, "Chicken Momos", "Fried", "5 Nos", "90"),
(3, "Chicken Garlic Momos", "Chinese/Fried", "5 Nos", "110"),
(4, "Pork Ribs", "Continental", "4 Nos", "500"),
(5, "French Fries", "Fast Food", "1 Plate", "80"),
(6, "Spaghetti", "Italian", "1 Plate", "350"),
(7, "Non-Veg Platter", "Combo", "1 Huge Plate", "800"),
```

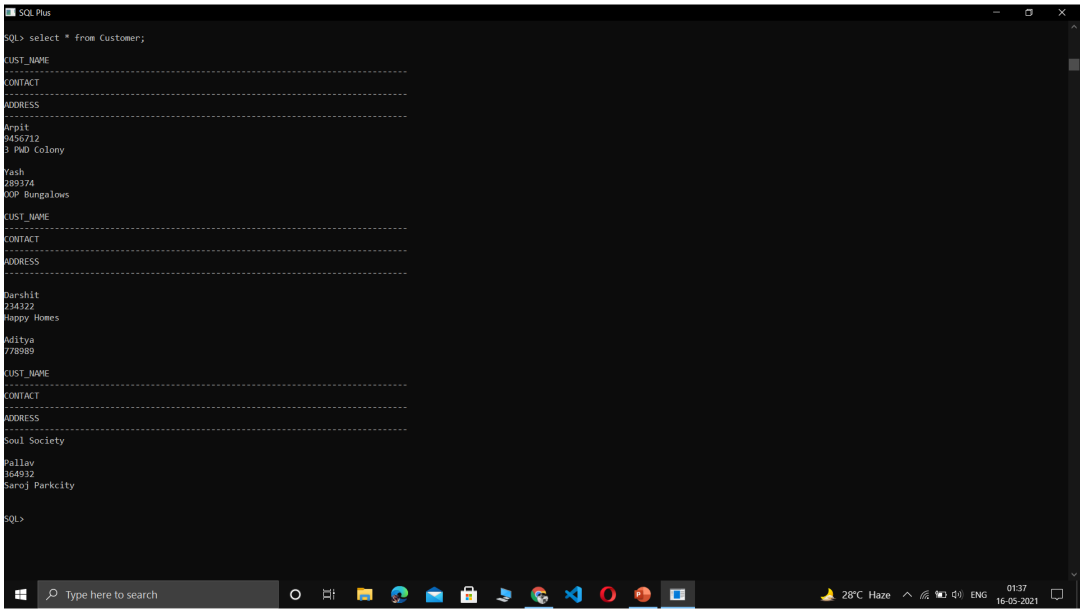


```
SQL Plus
   ITEM_NO
----------
NAME
--------------------------------------------------------------------------------
DESCRIPTION
--------------------------------------------------------------------------------
  QUANTITY      PRICE
---------- ----------
         1
Chicken Momos
Steamed
         5         80

   ITEM_NO
----------
NAME
--------------------------------------------------------------------------------
DESCRIPTION
--------------------------------------------------------------------------------
  QUANTITY      PRICE
---------- ----------
         2
Chicken Momos
Fired
         5         90

   ITEM_NO
----------
NAME
--------------------------------------------------------------------------------
DESCRIPTION
--------------------------------------------------------------------------------
  QUANTITY      PRICE
---------- ----------
         3
Chicken Garlic Momos
Fried
         5        110

   ITEM_NO
----------
NAME
--------------------------------------------------------------------------------
```



```
SQL Plus
Porkribs
Continental
         4        500

   ITEM_NO
----------
NAME
--------------------------------------------------------------------------------
DESCRIPTION
--------------------------------------------------------------------------------
  QUANTITY      PRICE
---------- ----------
         5
French Fries
Fast Food
         1         80

   ITEM_NO
----------
NAME
--------------------------------------------------------------------------------
DESCRIPTION
--------------------------------------------------------------------------------
  QUANTITY      PRICE
---------- ----------
         6
Spaghetti
Italian
         1        350

   ITEM_NO
----------
NAME
--------------------------------------------------------------------------------
DESCRIPTION
--------------------------------------------------------------------------------
  QUANTITY      PRICE
---------- ----------
         7
Non Veg Platter
Combo
         2        800
```

```
insert into `CUSTOMER` values
("Arpit", "938912","3 PWD Colony"),
("Yash", "289374","OOP Bungalows"),
("Darshit","234322","Happy Homes"),
("Aditya", "778989","Soul Society"),
("Pallav","364932","Saroj Parkcity");
```



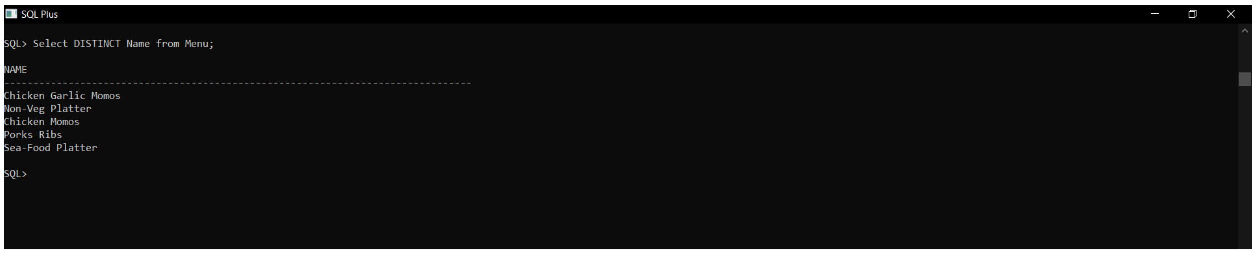
```
SQL> select * from Customer;

CUST_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
Arpit
9456712
3 PWD Colony

Yash
289374
OOP Bungalows

CUST_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------

Darshit
234322
Happy Homes

Aditya
778989

CUST_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
Soul Society

Pallav
364932
Saroj Parkcity


SQL>
```

2) Select:

a) Distinct: The SELECT DISTINCT statement is used to print distinct values .

Syntax:
SELECT DISTINCT Column1,Column2…. From table_name;

Select distinct **Name** from Menu;

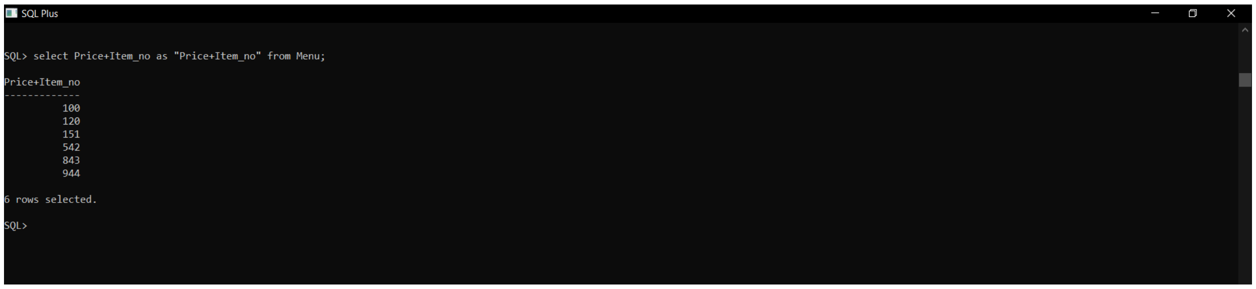


```
SQL> Select DISTINCT Name from Menu;

NAME
--------------------------------------------------------------------------------
Chicken Garlic Momos
Non-Veg Platter
Chicken Momos
Porks Ribs
Sea-Food Platter

SQL>
```

b) Arithmetic in select:

1) Select **price+10** from Menu;

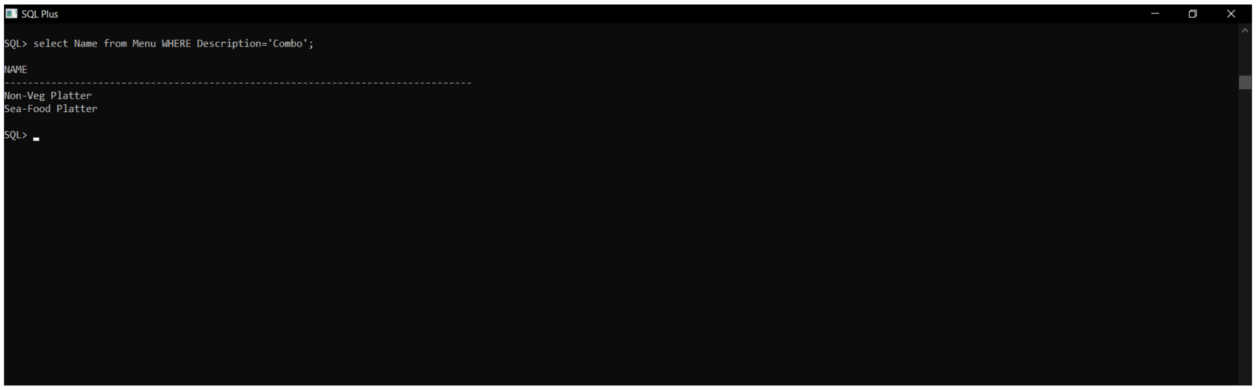
2) Select **price+item_no** as "price+item_no" from Menu;



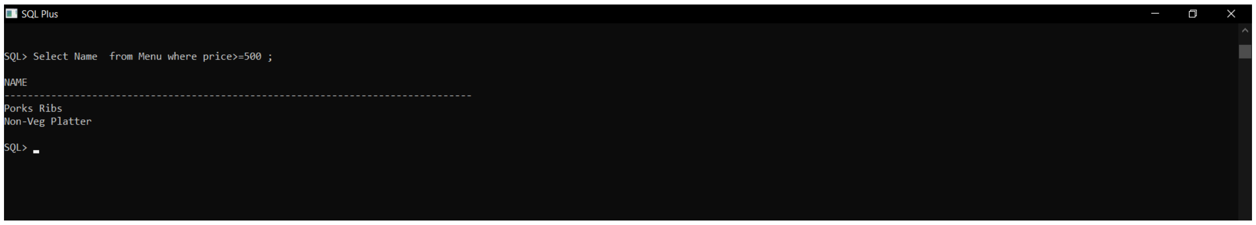
```
SQL> select Price+Item_no as "Price+Item_no" from Menu;

Price+Item_no
-------------
          100
          120
          151
          542
          843
          944

6 rows selected.

SQL>
```

c) Conditional Clause:

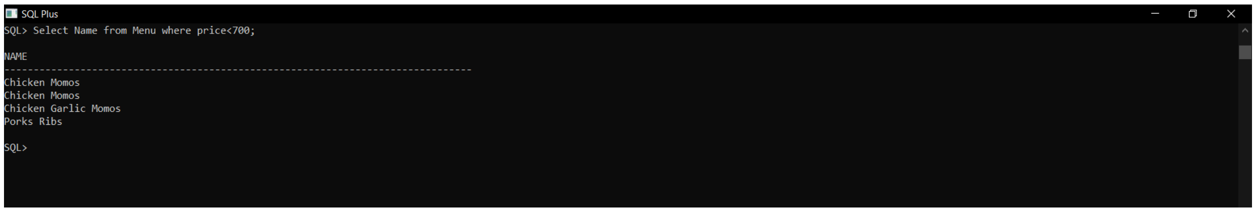
1) Select Name from Menu where price>=500 ;
2) Select Name from Menu where Description = “combo”;
3) Select Name from Menu where price<700;
4) Select * from chef where Chef_name=’Gordon’ and Salary= 27000;
5) Select * from waiter where Waiter_name=’Ayan’ or salary=15000;



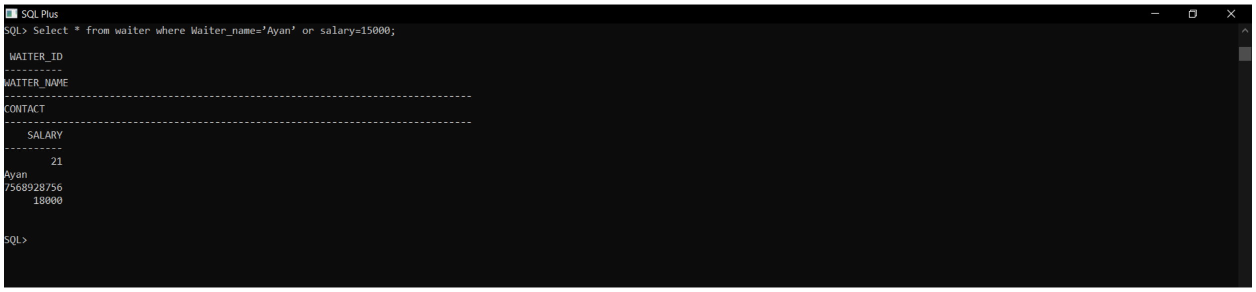
```
SQL> select Name from Menu WHERE Description='Combo';

NAME
--------------------------------------------------------------------------------
Non-Veg Platter
Sea-Food Platter

SQL>
```



```
SQL> Select Name  from Menu where price>=500 ;

NAME
--------------------------------------------------------------------------------
Porks Ribs
Non-Veg Platter

SQL>
```



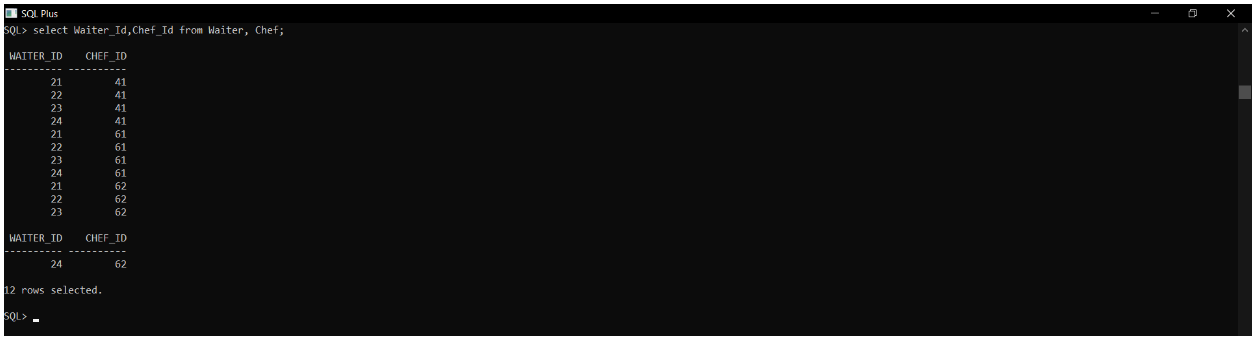
```
SQL> Select Name from Menu where price<700;

NAME
--------------------------------------------------------------------------------
Chicken Momos
Chicken Momos
Chicken Garlic Momos
Porks Ribs

SQL>
```



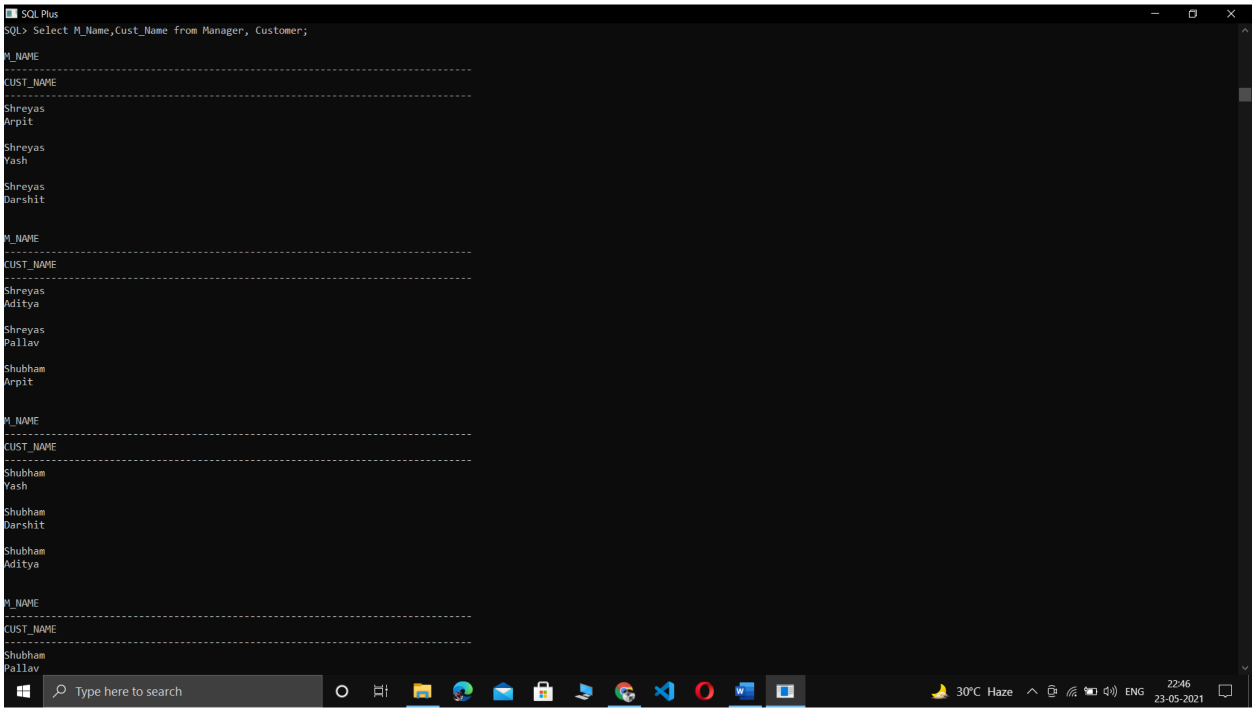
```
SQL> Select * from waiter where Waiter_name='Ayan' or salary=15000;

 WAITER_ID
----------
WAITER_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
    SALARY
----------
        21
Ayan
7568928756
     18000


SQL>
```

d) Select from multiple relations: (Cartesian Product)

1) Select Waiter_Id,Chef_Id from Waiter, Chef;
2) Select M_Name,Cust_Name from Manager, Customer;



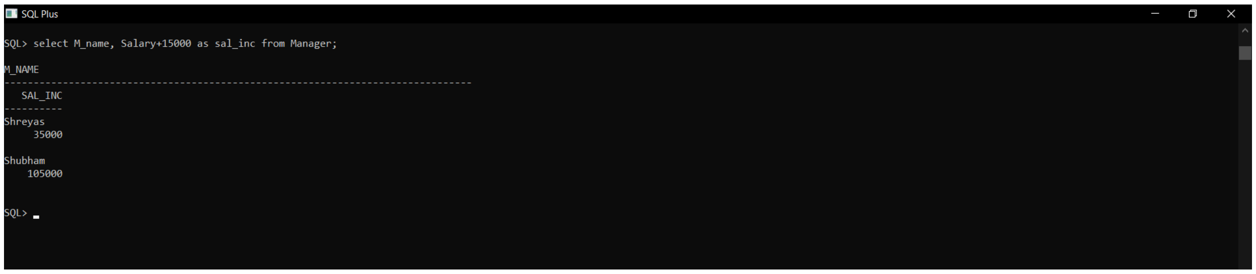
```
SQL> select Waiter_Id,Chef_Id from Waiter, Chef;

 WAITER_ID    CHEF_ID
---------- ----------
        21         41
        22         41
        23         41
        24         41
        21         61
        22         61
        23         61
        24         61
        21         62
        22         62
        23         62

 WAITER_ID    CHEF_ID
---------- ----------
        24         62

12 rows selected.

SQL> _
```



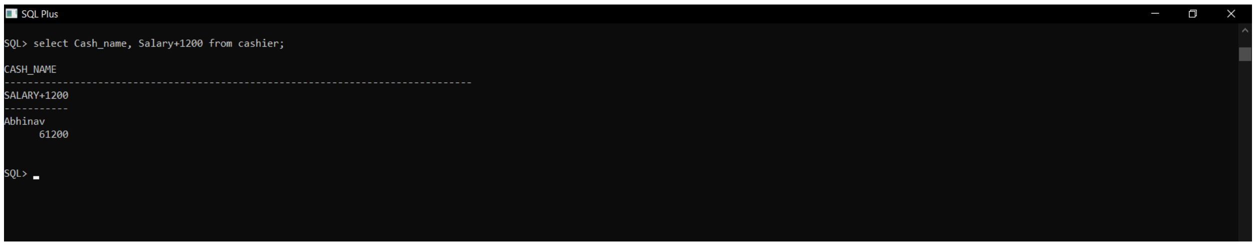
```
SQL> Select M_Name,Cust_Name from Manager, Customer;

M_NAME
--------------------------------------------------------------------------------
CUST_NAME
--------------------------------------------------------------------------------
Shreyas
Arpit

Shreyas
Yash

Shreyas
Darshit


M_NAME
--------------------------------------------------------------------------------
CUST_NAME
--------------------------------------------------------------------------------
Shreyas
Aditya

Shreyas
Pallav

Shubham
Arpit


M_NAME
--------------------------------------------------------------------------------
CUST_NAME
--------------------------------------------------------------------------------
Shubham
Yash

Shubham
Darshit

Shubham
Aditya


M_NAME
--------------------------------------------------------------------------------
CUST_NAME
--------------------------------------------------------------------------------
Shubham
Pallav
```

e) Rename:

1) Select Cash_name, Salary+1200 from cashier;
2) Select M_name, Salary+15000 as sal_inc from Manager;



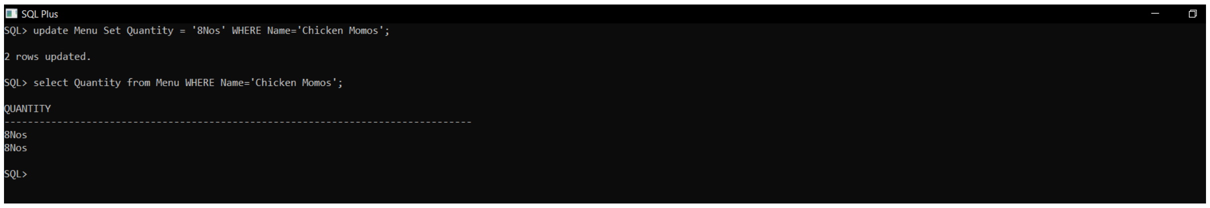
```
SQL> select M_name, Salary+15000 as sal_inc from Manager;

M_NAME
--------------------------------------------------------------------------------
   SAL_INC
----------
Shreyas
     35000

Shubham
    105000


SQL>
```



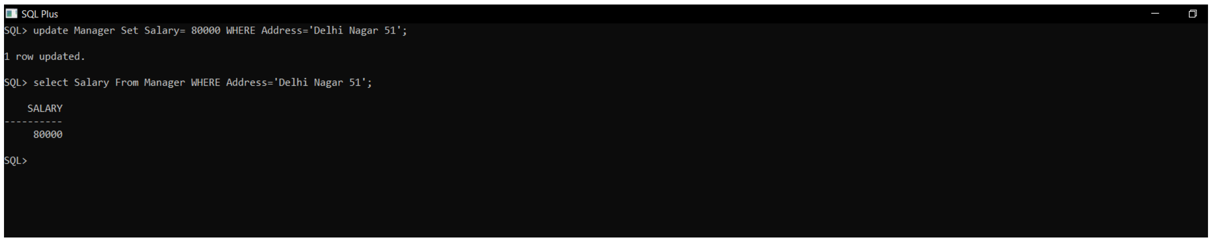
```
SQL> select Cash_name, Salary+1200 from cashier;

CASH_NAME
--------------------------------------------------------------------------------
SALARY+1200
-----------
Abhinav
      61200

SQL>
```

3) Update:

1) Update Menu Set Quantity = “8 Nos” where Name=”Chicken Momos”;
2) Update Manager Set Salary= 40000 where Address=”Mysore Road, Nagaland”;
3) Update Restaurant Set Location=”Velachery” where Name=”Chili’s”;



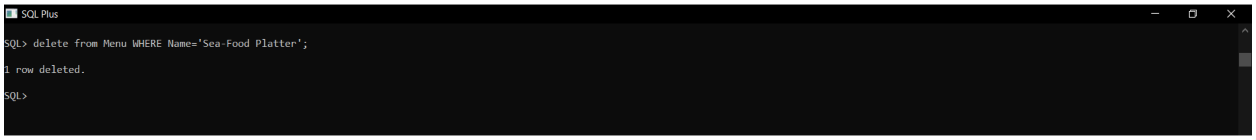
```
SQL> update Menu Set Quantity = '8Nos' WHERE Name='Chicken Momos';

2 rows updated.

SQL> select Quantity from Menu WHERE Name='Chicken Momos';

QUANTITY
--------------------------------------------------------------------------------
8Nos
8Nos

SQL>
```



```
SQL> update Manager Set Salary= 80000 WHERE Address='Delhi Nagar 51';

1 row updated.

SQL> select Salary From Manager WHERE Address='Delhi Nagar 51';

    SALARY
----------
     80000

SQL>
```

4) Delete:

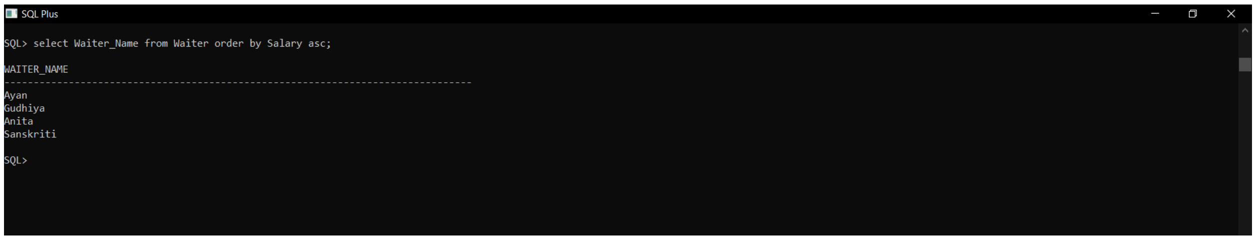
1) Delete from Menu where Name=”Sea-Food Platter”;



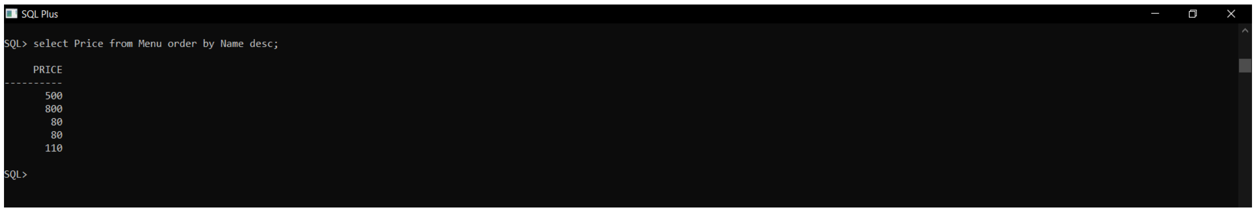
```
SQL> delete from Menu WHERE Name='Sea-Food Platter';

1 row deleted.

SQL>
```

5) Order-By Clause:

1) Select Waiter_Name from Waiter order by Salary asc;
2) Select Price from Menu order by Name desc;



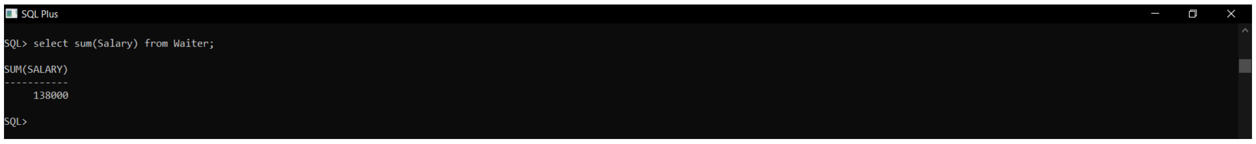
```
SQL> select Waiter_Name from Waiter order by Salary asc;

WAITER_NAME
--------------------------------------------------------------------------------
Ayan
Gudhiya
Anita
Sanskriti

SQL>
```



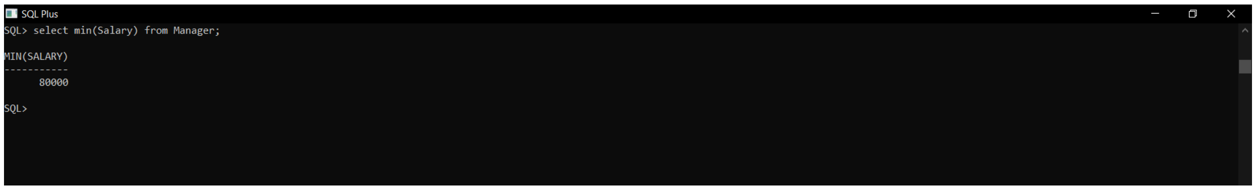
```
SQL> select Price from Menu order by Name desc;

     PRICE
----------
       500
       800
        80
        80
       110

SQL>
```

6) Aggregate Functions:

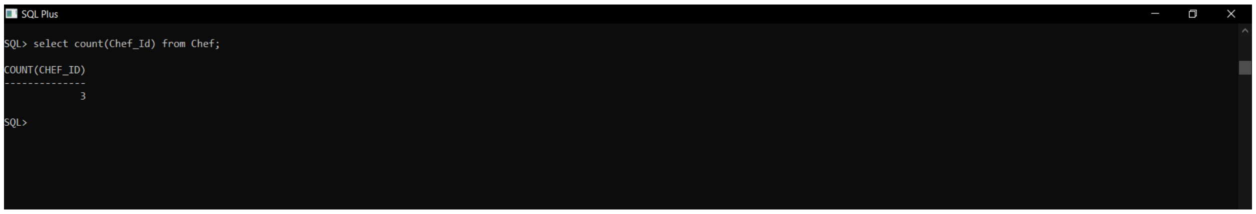
1) Sum: Select sum(Salary) from Waiter;
2) Min : Select min(Salary) from Manager;
3) Max: Select max(Salary) from Waiter;
4) Count: Select count(Chef_Id) from Chef;



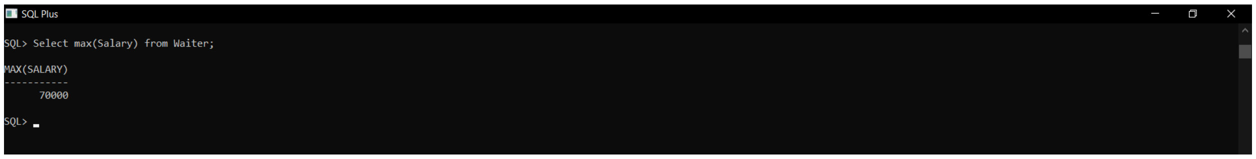
```
SQL> select sum(Salary) from Waiter;

SUM(SALARY)
-----------
     138000

SQL>
```



```
SQL> select min(Salary) from Manager;

MIN(SALARY)
-----------
      80000

SQL>
```



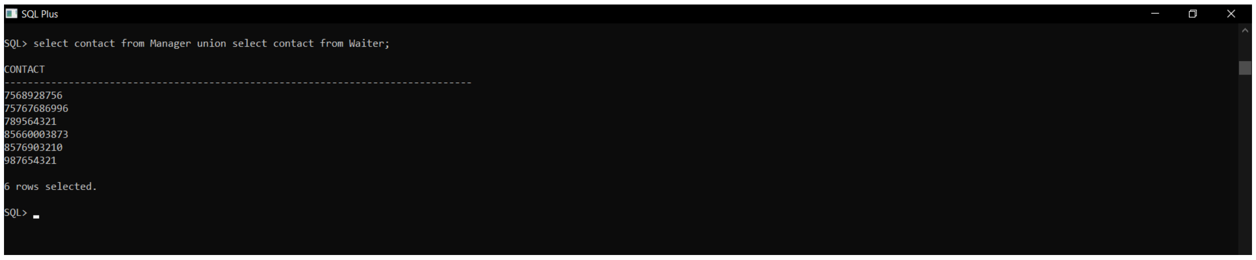
```
SQL> select count(Chef_Id) from Chef;

COUNT(CHEF_ID)
--------------
             3

SQL>
```



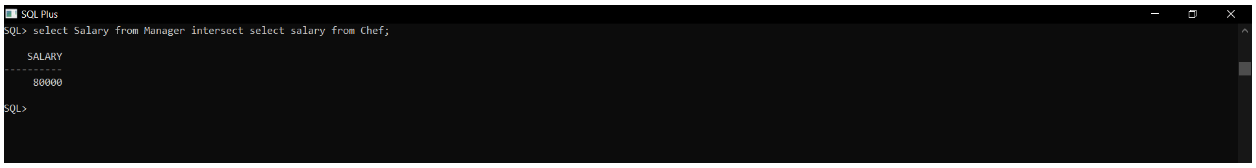
```
SQL> Select max(Salary) from Waiter;

MAX(SALARY)
-----------
      70000

SQL> 
```

7) Set Operations:

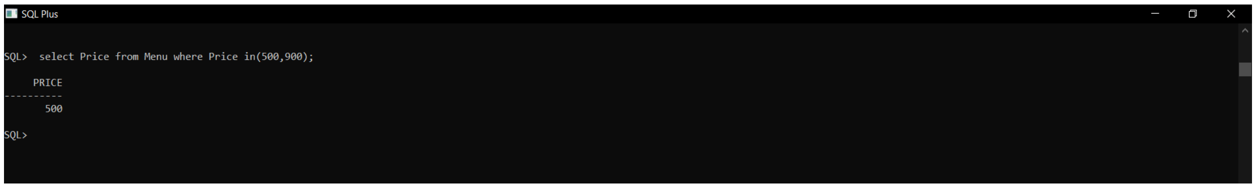
1) Union: Select contact from Manager union select contact from Waiter;
2) Intersect: Select Salary from Manager intersect select salary from Chef;
3) In: Select Price from Menu-Items where Price in(500,900);



```
SQL> select contact from Manager union select contact from Waiter;

CONTACT
--------------------------------------------------------------------------------
7568928756
75767686996
789564321
85660003873
8576903210
987654321

6 rows selected.

SQL> 
```

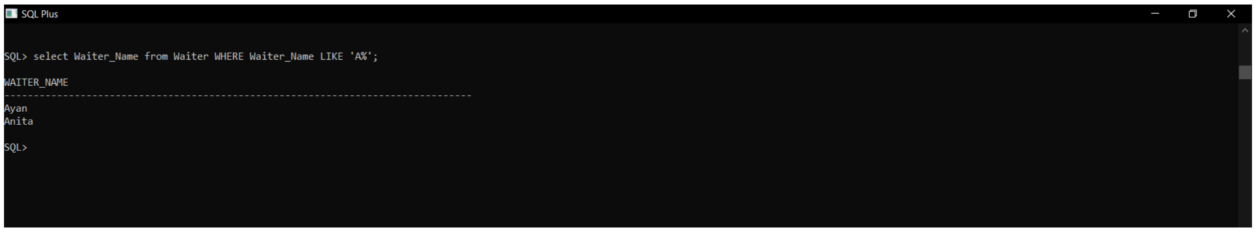


```
SQL> select Salary from Manager intersect select salary from Chef;

    SALARY
----------
     80000

SQL>
```



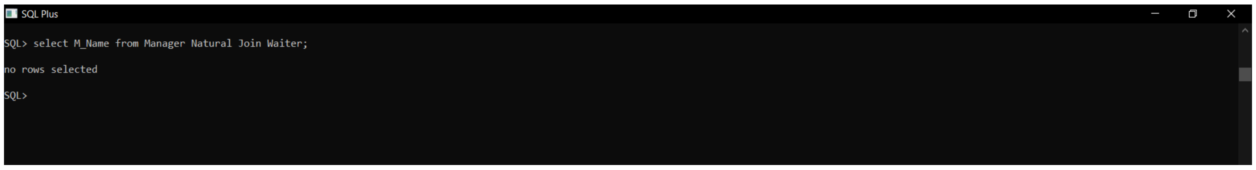
```
SQL>  select Price from Menu where Price in(500,900);

     PRICE
----------
       500

SQL>
```

8) String Operations: Select Waiter_Name from Waiter where Waiter_Name like ‘A%’;



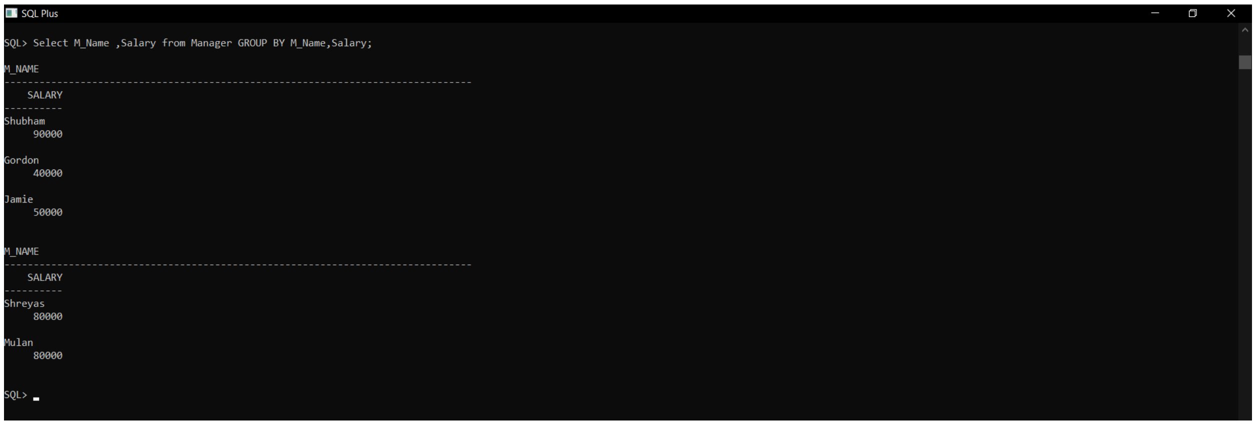
```
SQL> select Waiter_Name from Waiter WHERE Waiter_Name LIKE 'A%';

WAITER_NAME
--------------------------------------------------------------------------------
Ayan
Anita

SQL>
```

9)Natural Join: Select * from Manager Natural Join Waiter;



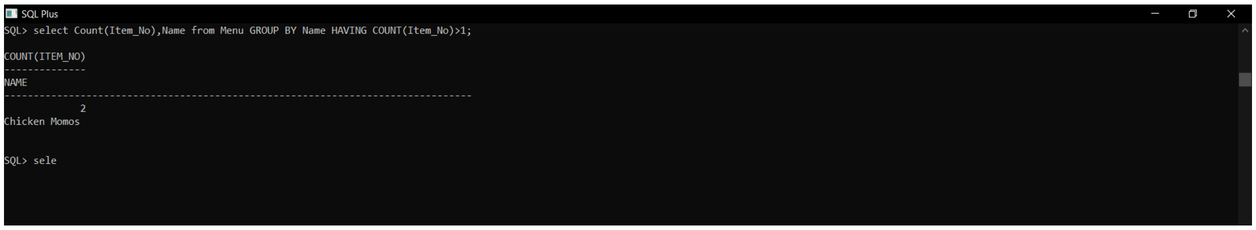
```
SQL> select M_Name from Manager Natural Join Waiter;

no rows selected

SQL>
```

10)Group By: Select M_Name ,Salary from Manager GROUP BY M_Name,Salary;



```
SQL> Select M_Name ,Salary from Manager GROUP BY M_Name,Salary;

M_NAME
--------------------------------------------------------------------------------
    SALARY
----------
Shubham
     90000

Gordon
     40000

Jamie
     50000


M_NAME
--------------------------------------------------------------------------------
    SALARY
----------
Shreyas
     80000

Mulan
     80000


SQL>
```

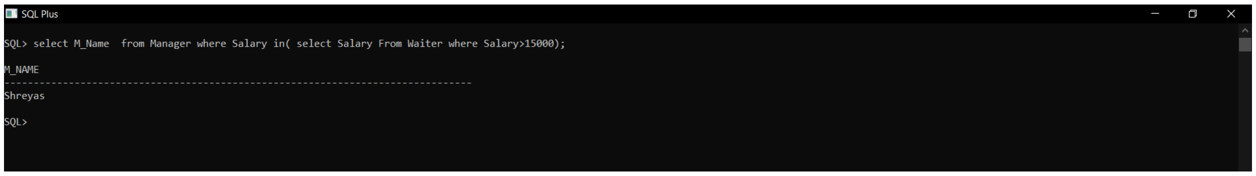
11) Having Clause: Select count(Item_No), Name from Menu group by Name having count(Item_No) > 1;



```
SQL> select Count(Item_No),Name from Menu GROUP BY Name HAVING COUNT(Item_No)>1;

COUNT(ITEM_NO)
--------------
NAME
--------------------------------------------------------------------------------
             2
Chicken Momos


SQL> sele
```

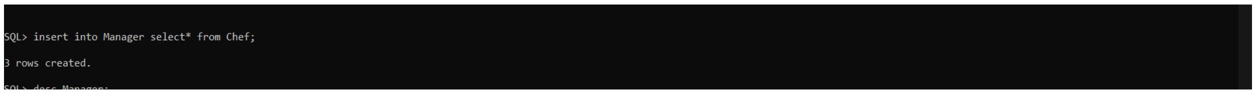
SUBQUERIES

1. Select M_Name from Manager where Salary in(select Salary From Waiter where Salary>15000);



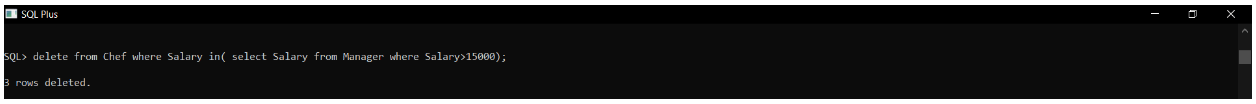
```
SQL> select M_Name  from Manager where Salary in( select Salary From Waiter where Salary>15000);

M_NAME
--------------------------------------------------------------------------------
Shreyas

SQL>
```

2. Insert into Manager select* from Chef;



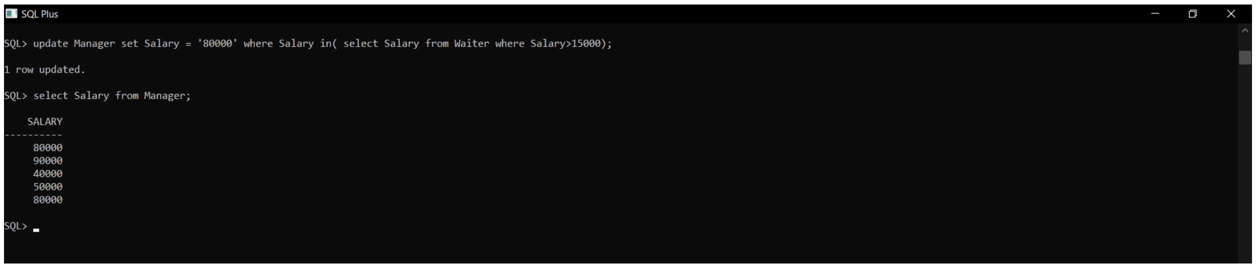
```
SQL> insert into Manager select* from Chef;

3 rows created.
```

3. Delete from Chef where Salary in(select Salary from Manager where Salary>15000);



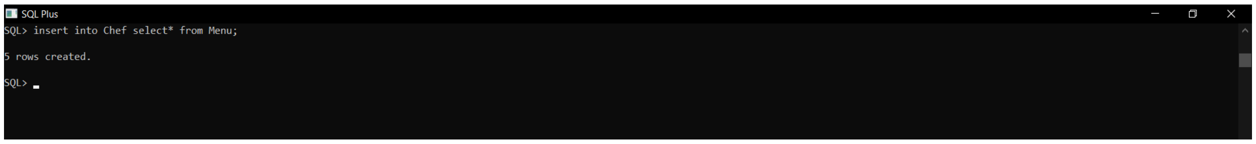
```
SQL> delete from Chef where Salary in( select Salary from Manager where Salary>15000);

3 rows deleted.
```

4. Update Manager set Salary = '80000' where Salary in(select Salary from Waiter where Salary>15000);



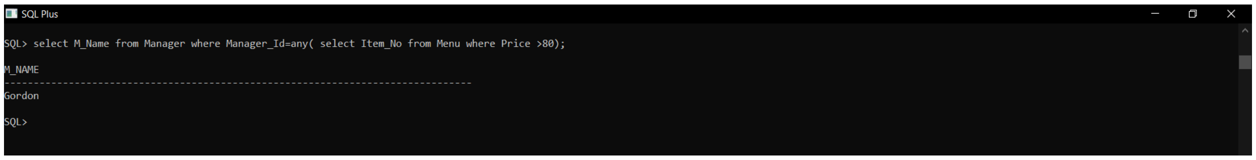
```
SQL> update Manager set Salary = '80000' where Salary in( select Salary from Waiter where Salary>15000);

1 row updated.

SQL> select Salary from Manager;

    SALARY
----------
     80000
     90000
     40000
     50000
     80000

SQL>
```

5. Insert into Chef select* from Menu;



```
SQL> insert into Chef select* from Menu;

5 rows created.

SQL>
```

6. Select M_Name from Manager where Manager_Id=any(select Item_No from Menu where Price >80);



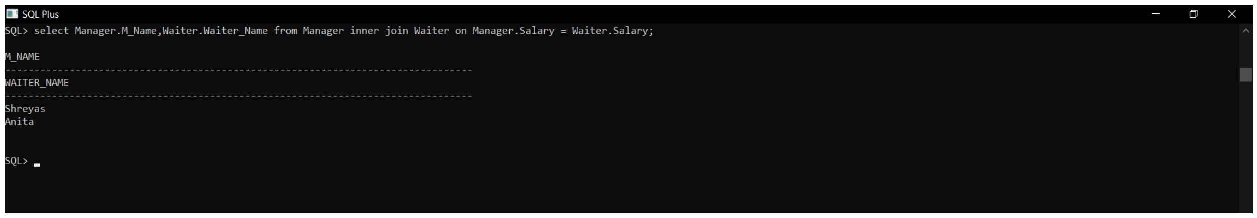
```
SQL> select M_Name from Manager where Manager_Id=any( select Item_No from Menu where Price >80);

M_NAME
--------------------------------------------------------------------------------
Gordon

SQL>
```

JOINS

1. **Inner Join:**

   The INNER JOIN keyword selects records that have matching values in both tables.

   select Manager.M_Name,Waiter.Waiter_Name from Manager inner join Waiter on Manager.Salary = Waiter.Salary;

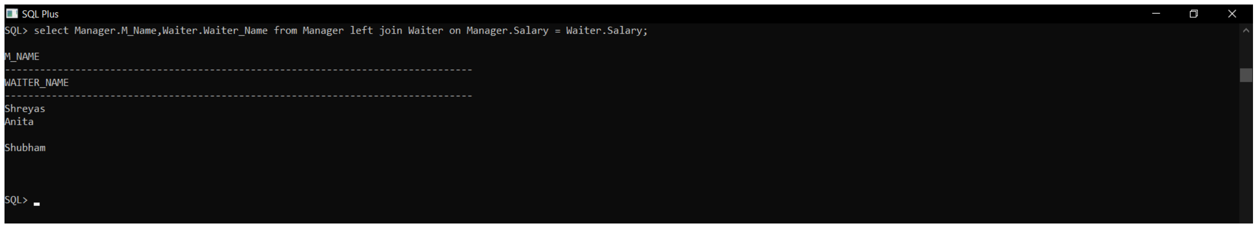


```
SQL> select Manager.M_Name,Waiter.Waiter_Name from Manager inner join Waiter on Manager.Salary = Waiter.Salary;

M_NAME
--------------------------------------------------------------------------------
WAITER_NAME
--------------------------------------------------------------------------------
Shreyas
Anita


SQL>
```

2. **Left Join:**

   The LEFT JOIN keyword returns all records from the left table (table1), and the matching records from the right table (table2). The result is 0 records from the right side, if there is no match.

   select Manager.M_Name,Waiter.Waiter_Name from Manager left join Waiter on Manager.Salary = Waiter.Salary;

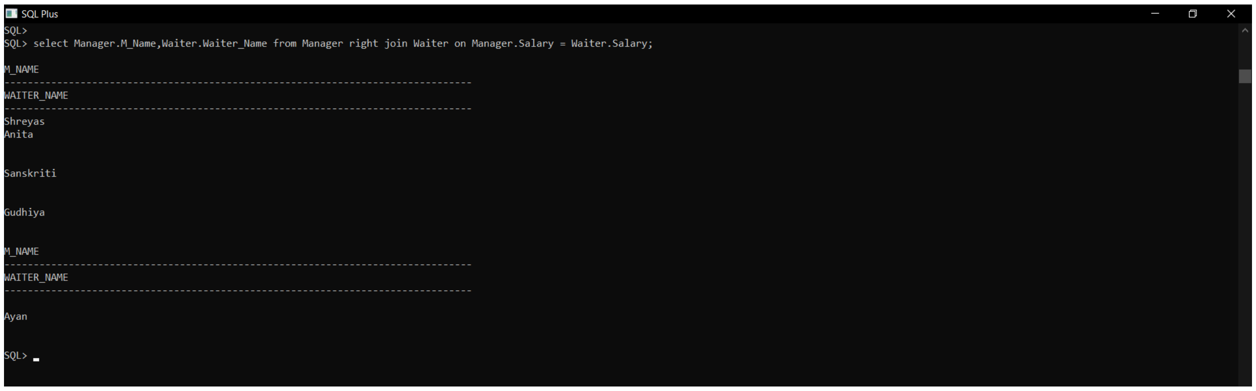


```
SQL> select Manager.M_Name,Waiter.Waiter_Name from Manager left join Waiter on Manager.Salary = Waiter.Salary;

M_NAME
--------------------------------------------------------------------------------
WAITER_NAME
--------------------------------------------------------------------------------
Shreyas
Anita

Shubham


SQL>
```

3. **Right Join:**

   The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side, if there is no match.

   select Manager.M_Name,Waiter.Waiter_Name from Manager right join Waiter on Manager.Salary = Waiter.Salary;

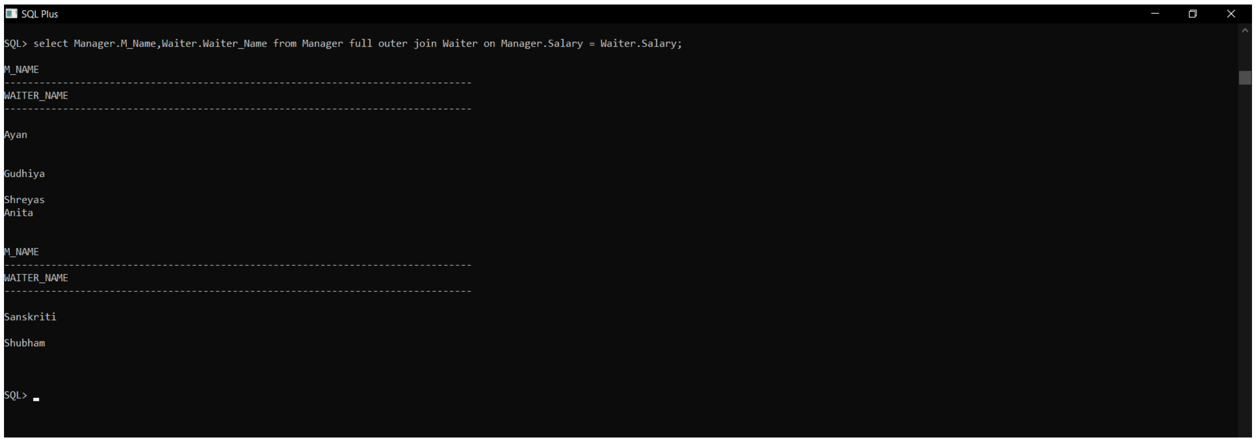


```
SQL>
SQL> select Manager.M_Name,Waiter.Waiter_Name from Manager right join Waiter on Manager.Salary = Waiter.Salary;

M_NAME
--------------------------------------------------------------------------------
WAITER_NAME
--------------------------------------------------------------------------------
Shreyas
Anita


Sanskriti


Gudhiya


M_NAME
--------------------------------------------------------------------------------
WAITER_NAME
--------------------------------------------------------------------------------

Ayan


SQL> _
```

4. **Full Join:**

The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.

select Manager.M_Name,Waiter.Waiter_Name from Manager full outer join Waiter on Manager.Salary = Waiter.Salary;

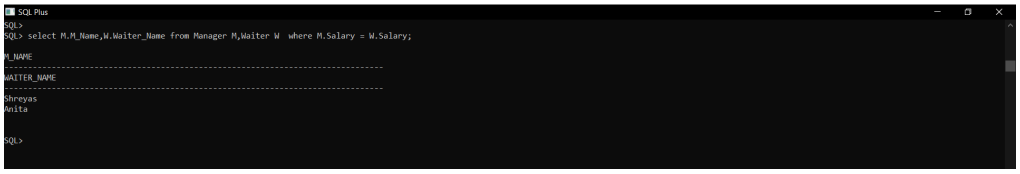


```
SQL> select Manager.M_Name,Waiter.Waiter_Name from Manager full outer join Waiter on Manager.Salary = Waiter.Salary;

M_NAME
--------------------------------------------------------------------------------
WAITER_NAME
--------------------------------------------------------------------------------

Ayan


Gudhiya

Shreyas
Anita


M_NAME
--------------------------------------------------------------------------------
WAITER_NAME
--------------------------------------------------------------------------------

Sanskriti

Shubham



SQL> _
```

5. **Self Join:**

A self join is a regular join, but the table is joined with itself.

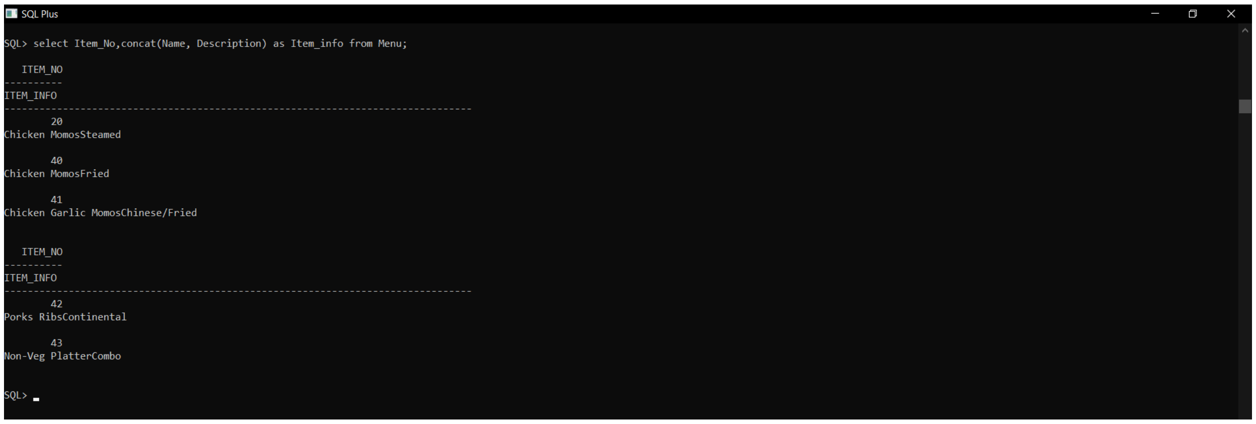
select M.M_Name,W.Waiter_Name from Manager M,Waiter W where M.Salary = W.Salary;



```
SQL>
SQL> select M.M_Name,W.Waiter_Name from Manager M,Waiter W  where M.Salary = W.Salary;

M_NAME
--------------------------------------------------------------------------------
WAITER_NAME
--------------------------------------------------------------------------------
Shreyas
Anita


SQL>
```

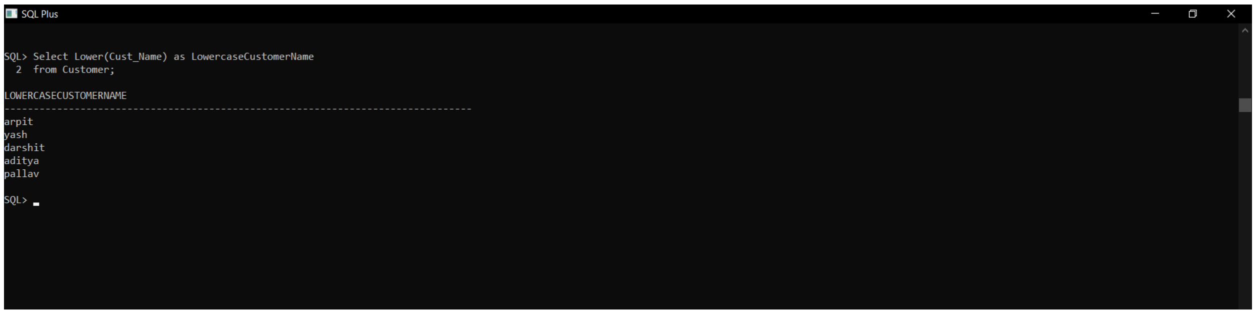
SQL FUNCTIONS

1. **Concat**: Select Item_No concat(Name, Description) Item_Info from Menu;



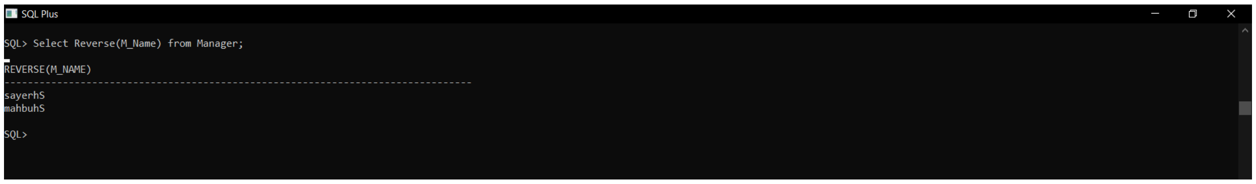
```
SQL> select Item_No,concat(Name, Description) as Item_info from Menu;

   ITEM_NO
----------
ITEM_INFO
--------------------------------------------------------------------------------
        20
Chicken MomosSteamed

        40
Chicken MomosFried

        41
Chicken Garlic MomosChinese/Fried


   ITEM_NO
----------
ITEM_INFO
--------------------------------------------------------------------------------
        42
Porks RibsContinental

        43
Non-Veg PlatterCombo


SQL>
```

2. **Lower:** Select Lower(Cust_Name) as LowercaseCustomerName from Customer;



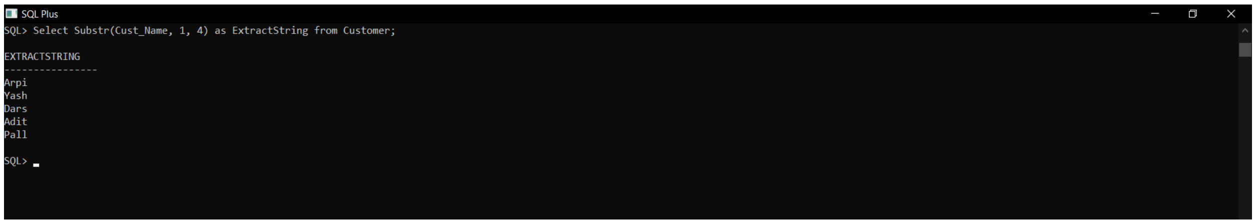
```
SQL> Select Lower(Cust_Name) as LowercaseCustomerName
  2  from Customer;

LOWERCASECUSTOMERNAME
--------------------------------------------------------------------------------
arpit
yash
darshit
aditya
pallav

SQL>
```

3. **Reverse:** Select Reverse(M_Name) from Manager;



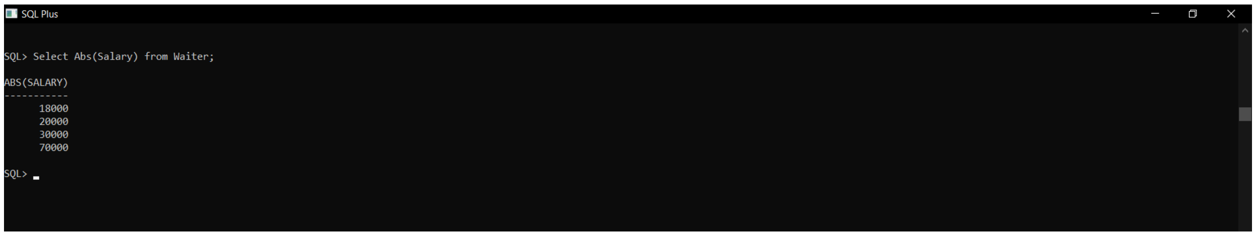
```
SQL> Select Reverse(M_Name) from Manager;

REVERSE(M_NAME)
--------------------------------------------------------------------------------
sayerhS
mahbuhS

SQL>
```

4. **Substring:** Select Substr(Cust_Name, 1, 4) as ExtractString from Customer;



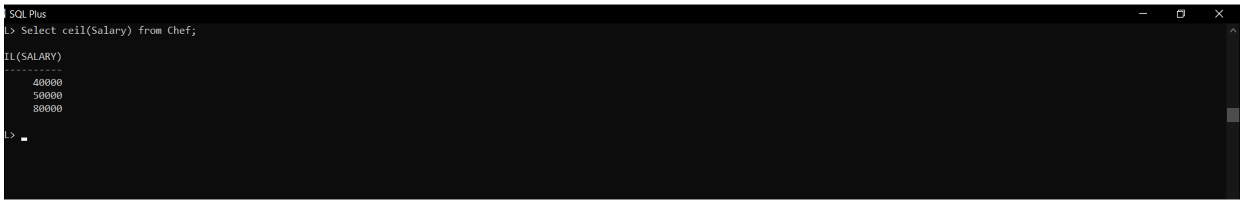
```
SQL> Select Substr(Cust_Name, 1, 4) as ExtractString from Customer;

EXTRACTSTRING
----------------
Arpi
Yash
Dars
Adit
Pall

SQL>
```

5. **Abs:** Select Abs(Salary) from Waiter;



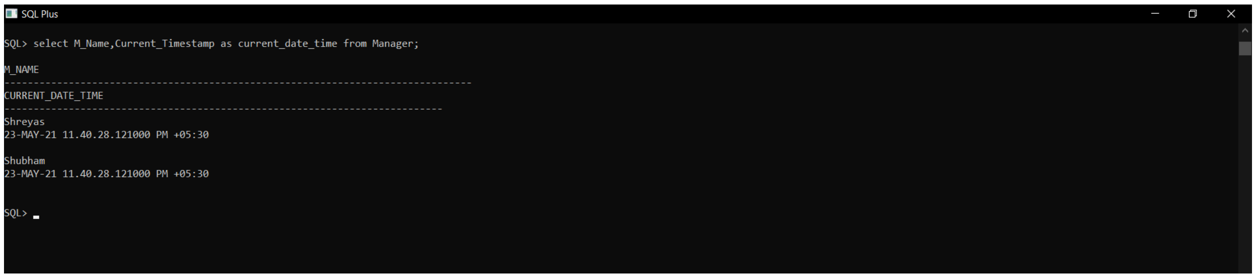
```
SQL> Select Abs(Salary) from Waiter;

ABS(SALARY)
-----------
      18000
      20000
      30000
      70000

SQL>
```

6. **Ceiling/floor:** Select ceil(Salary) from Chef;



```
L> Select ceil(Salary) from Chef;

IL(SALARY)
----------
     40000
     50000
     80000

L>
```

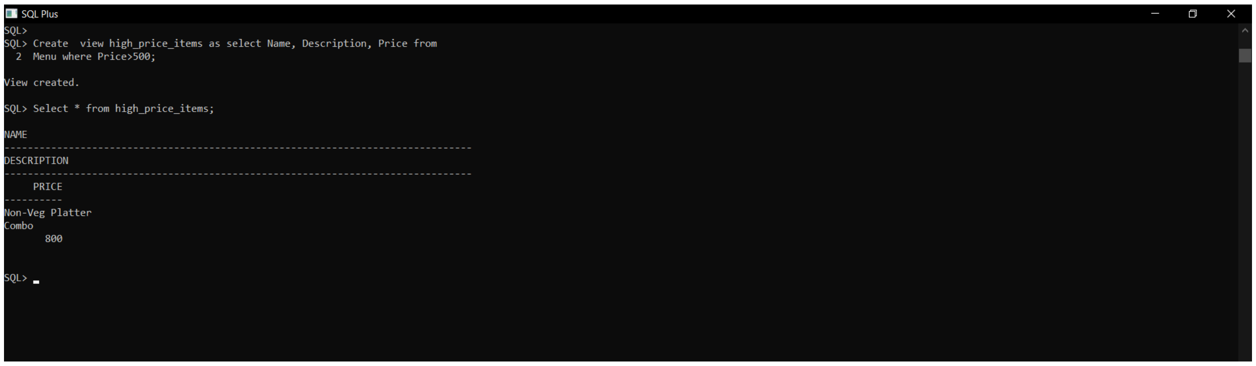
7. **Current_timestamp:** Select M_Name,Current_Timestamp As current_date_time from Manager;



```
SQL> select M_Name,Current_Timestamp as current_date_time from Manager;

M_NAME
--------------------------------------------------------------------------------
CURRENT_DATE_TIME
---------------------------------------------------------------------------
Shreyas
23-MAY-21 11.40.28.121000 PM +05:30

Shubham
23-MAY-21 11.40.28.121000 PM +05:30


SQL>
```

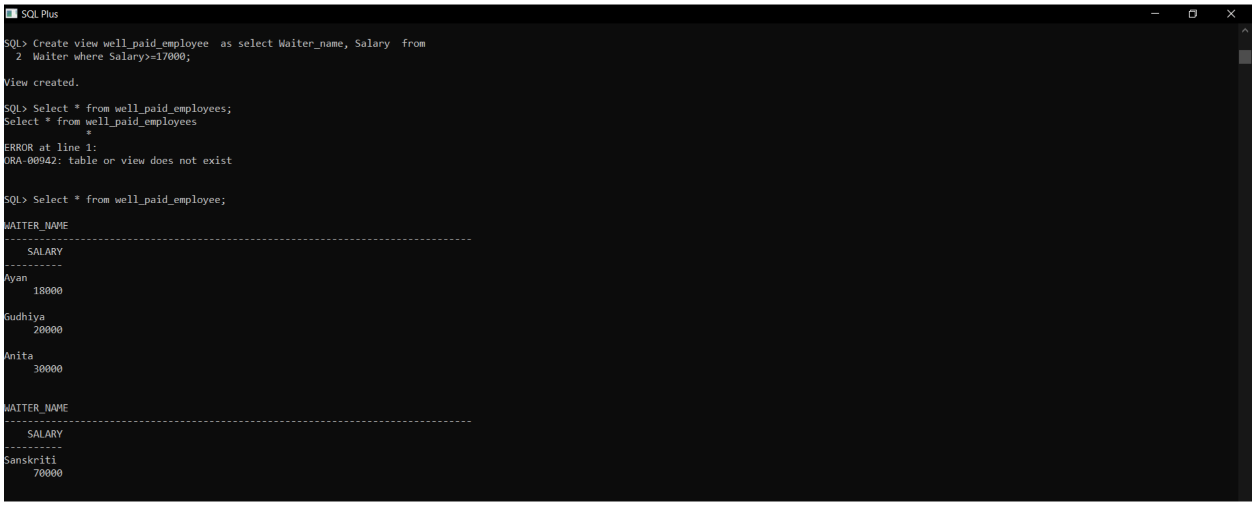
VIEWS

1) Create view high_price_items as select Name, Description, Price from
   Menu where Price>500;
   Select * from high_price_items;



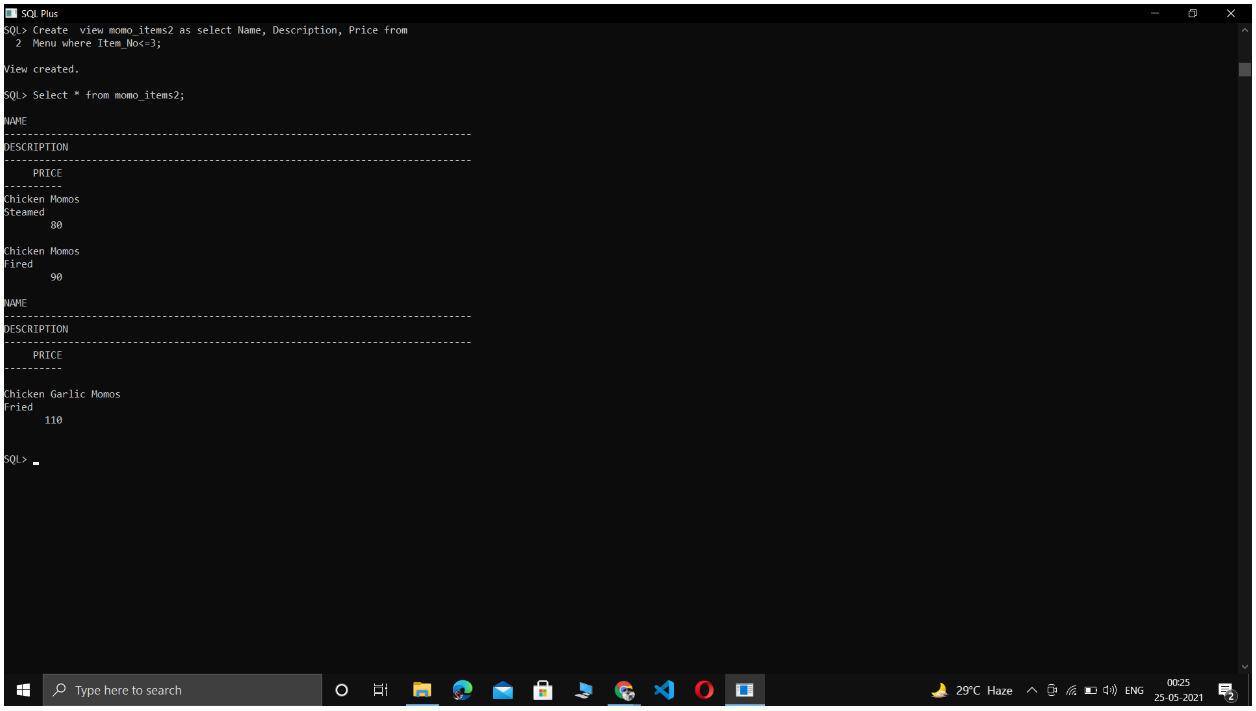
```
SQL>
SQL> Create  view high_price_items as select Name, Description, Price from
  2  Menu where Price>500;

View created.

SQL> Select * from high_price_items;

NAME
--------------------------------------------------------------------------------
DESCRIPTION
--------------------------------------------------------------------------------
     PRICE
----------
Non-Veg Platter
Combo
       800


SQL> _
```

2) Create view well_paid_employee as select Waiter_name, Salary from
   Waiter where Salary>=17000;
   Select * from well_paid_employees;



```
SQL> Create view well_paid_employee  as select Waiter_name, Salary  from
  2  Waiter where Salary>=17000;

View created.

SQL> Select * from well_paid_employees;
Select * from well_paid_employees
              *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> Select * from well_paid_employee;

WAITER_NAME
--------------------------------------------------------------------------------
    SALARY
----------
Ayan
     18000

Gudhiya
     20000

Anita
     30000


WAITER_NAME
--------------------------------------------------------------------------------
    SALARY
----------
Sanskriti
     70000
```

3) Create view momo_items2 as select Name, Description, Price from
   Menu where Item_No<=3;
   Select * from momo_items2;



```
SQL> Create  view momo_items2 as select Name, Description, Price from
  2  Menu where Item_No<=3;

View created.

SQL> Select * from momo_items2;

NAME
--------------------------------------------------------------------------------
DESCRIPTION
--------------------------------------------------------------------------------
     PRICE
----------
Chicken Momos
Steamed
        80

Chicken Momos
Fired
        90

NAME
--------------------------------------------------------------------------------
DESCRIPTION
--------------------------------------------------------------------------------
     PRICE
----------

Chicken Garlic Momos
Fried
       110


SQL> _
```

PL/SQL SUBPROGRAMS

1. IMPLICIT CURSOR

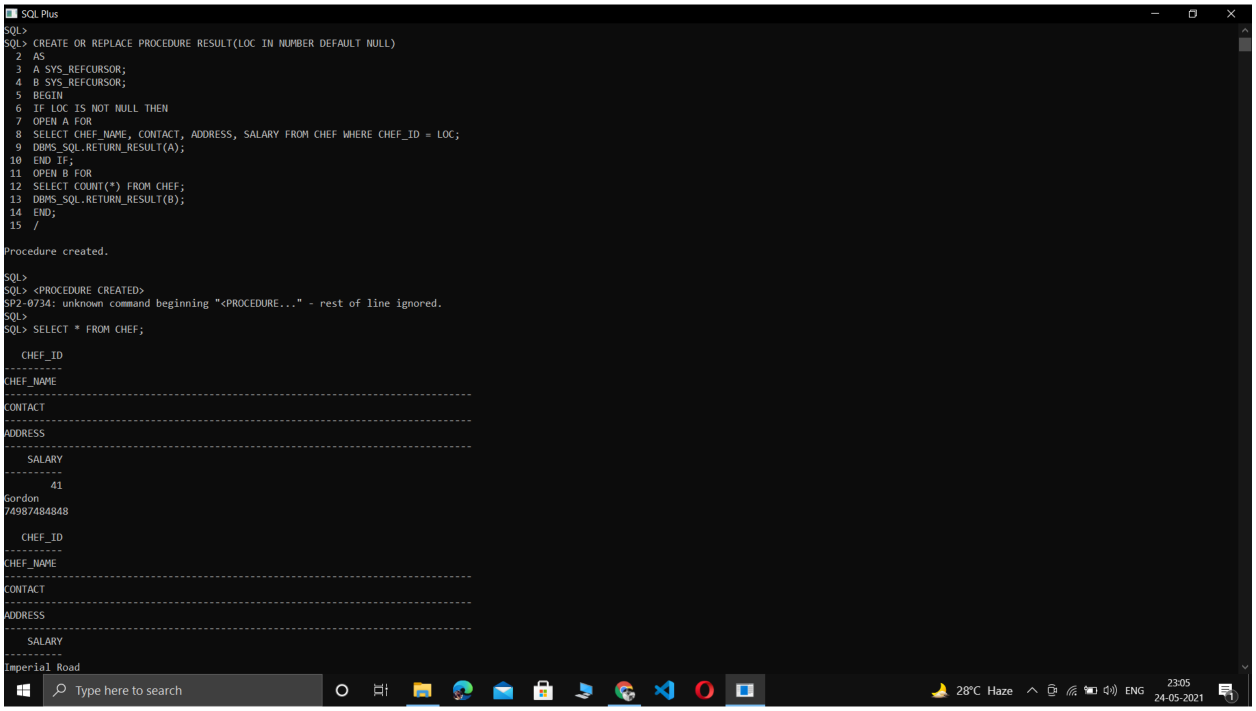
```
CREATE OR REPLACE PROCEDURE RESULT(LOC IN NUMBER DEFAULT NULL)
AS
A SYS_REFCURSOR;
B SYS_REFCURSOR;
BEGIN
IF LOC IS NOT NULL THEN
OPEN A FOR
SELECT CHEF_NAME, CONTACT, ADDRESS, SALARY FROM CHEF  WHERE CHEF_ID = LOC;
DBMS_SQL.RETURN_RESULT(A);
END IF;
OPEN B FOR
SELECT COUNT(*) FROM CHEF;
DBMS_SQL.RETURN_RESULT(B);
END;
/
```

<PROCEDURE CREATED>

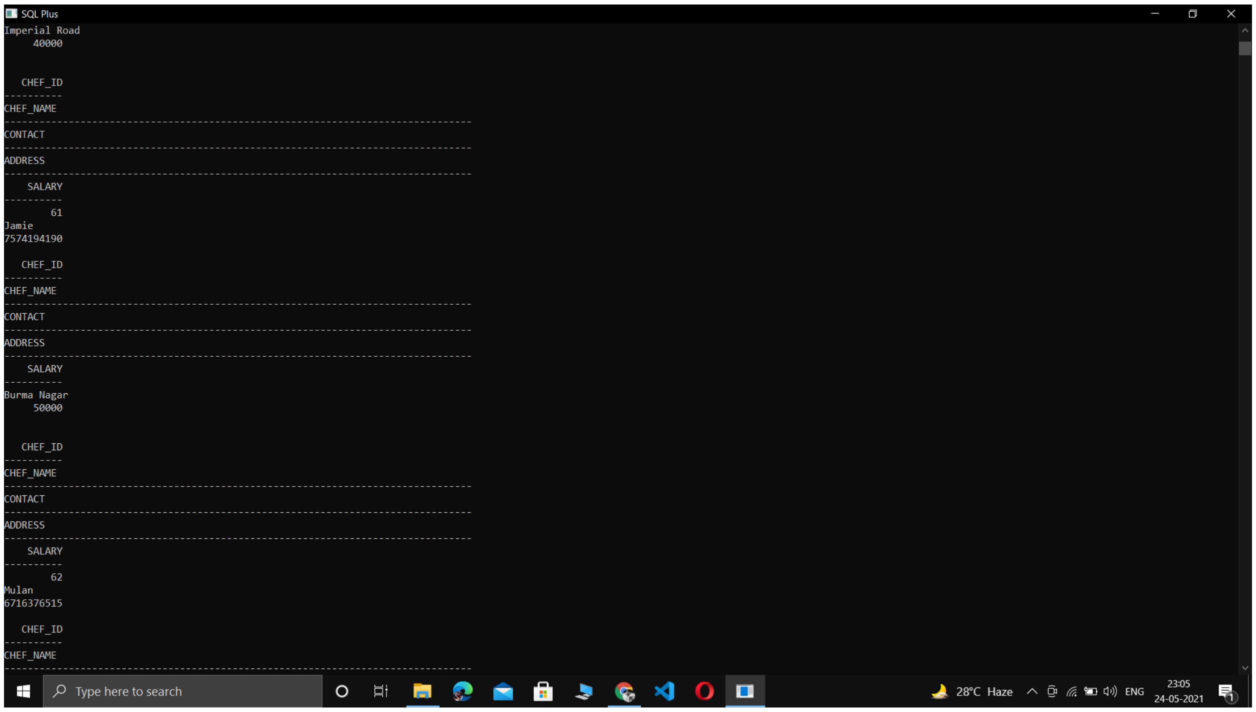
```
SELECT * FROM CHEF;
```

<TABLE DISPLAYED>

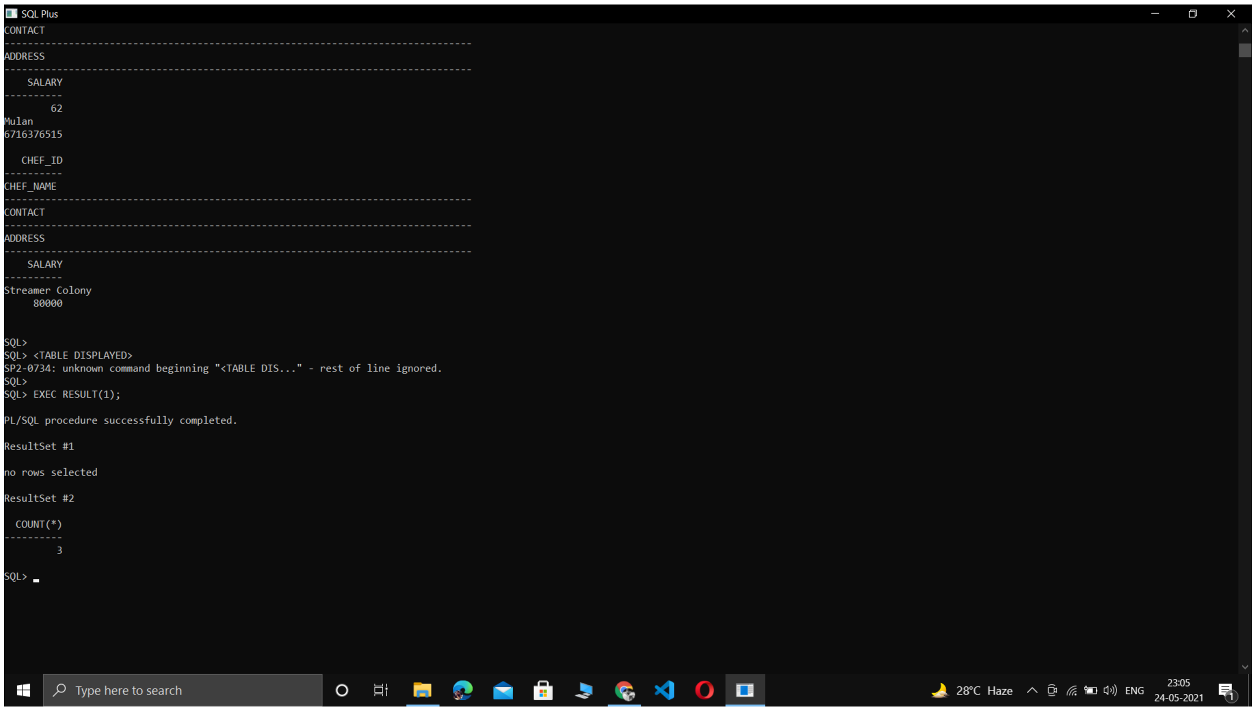
```
EXEC RESULT(1);
```



```
SQL>
SQL> CREATE OR REPLACE PROCEDURE RESULT(LOC IN NUMBER DEFAULT NULL)
AS
A SYS_REFCURSOR;
B SYS_REFCURSOR;
BEGIN
IF LOC IS NOT NULL THEN
OPEN A FOR
SELECT CHEF_NAME, CONTACT, ADDRESS, SALARY FROM CHEF WHERE CHEF_ID = LOC;
DBMS_SQL.RETURN_RESULT(A);
END IF;
OPEN B FOR
SELECT COUNT(*) FROM CHEF;
DBMS_SQL.RETURN_RESULT(B);
END;
/

Procedure created.

SQL>
SQL> <PROCEDURE CREATED>
SP2-0734: unknown command beginning "<PROCEDURE..." - rest of line ignored.
SQL>
SQL> SELECT * FROM CHEF;

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
        41
Gordon
74987484848

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
Imperial Road
```



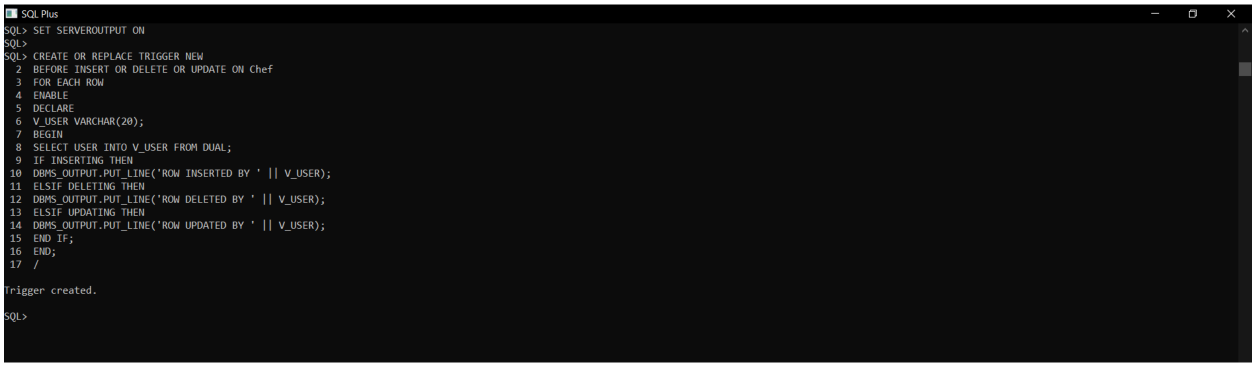
```
Imperial Road
     40000


   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
        61
Jamie
7574194190

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
Burma Nagar
     50000


   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
        62
Mulan
6716376515

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
```



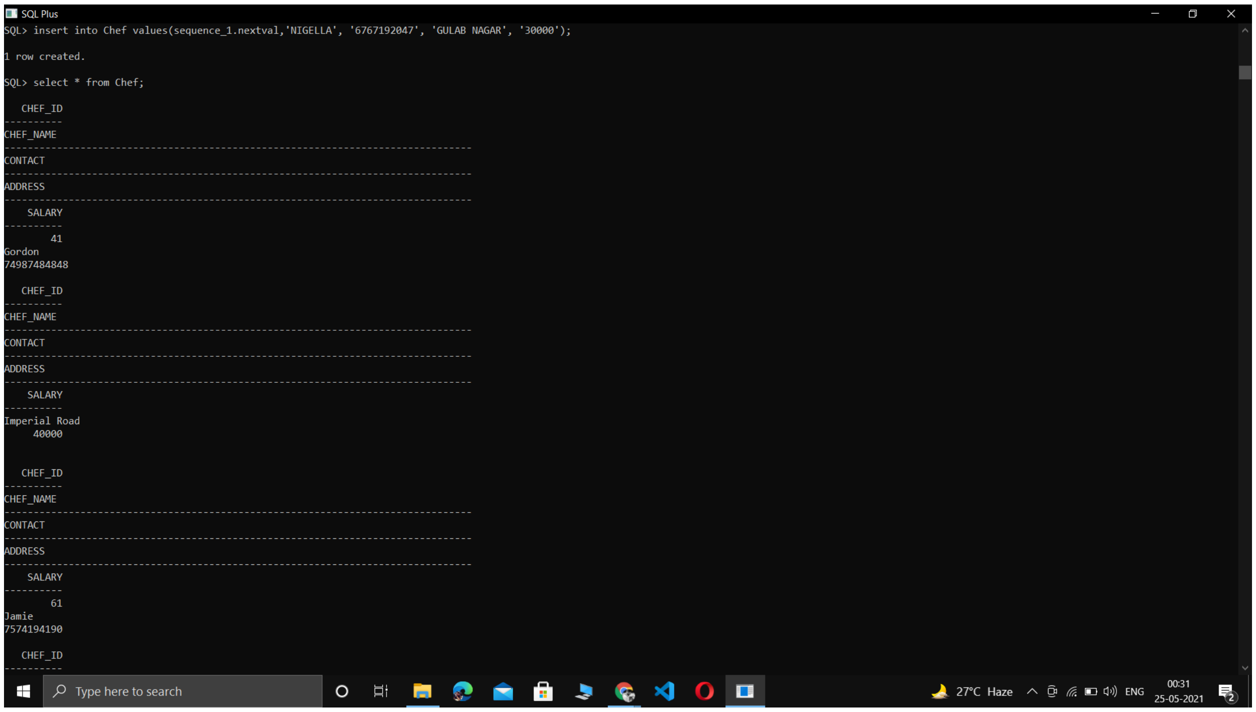
```
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
        62
Mulan
6716376515

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
Streamer Colony
     80000


SQL>
SQL> <TABLE DISPLAYED>
SP2-0734: unknown command beginning "<TABLE DIS..." - rest of line ignored.
SQL>
SQL> EXEC RESULT(1);

PL/SQL procedure successfully completed.

ResultSet #1

no rows selected

ResultSet #2

  COUNT(*)
----------
         3

SQL> _
```

2. TRIGGER: TO ENTER A ROW

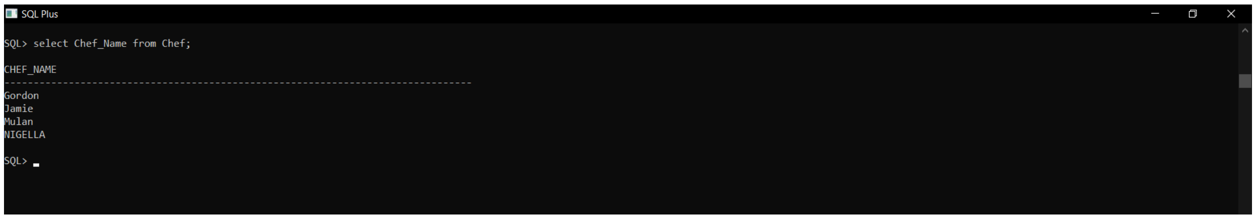
```
SET SERVEROUTPUT ON

CREATE OR REPLACE TRIGGER NEW
BEFORE INSERT OR DELETE OR UPDATE ON Chef
FOR EACH ROW
ENABLE
DECLARE
V_USER VARCHAR(20);
BEGIN
SELECT USER INTO V_USER FROM DUAL;
IF INSERTING THEN
DBMS_OUTPUT.PUT_LINE('ROW INSERTED BY ' || V_USER);
ELSIF DELETING THEN
DBMS_OUTPUT.PUT_LINE('ROW DELETED BY ' || V_USER);
ELSIF UPDATING THEN
DBMS_OUTPUT.PUT_LINE('ROW UPDATED BY ' || V_USER);
END IF;
END;
/
INSERT INTO CHEF VALUES('NIGELLA', '6767192047', 'GULAB NAGAR', '30000');
```



```
SQL> SET SERVEROUTPUT ON
SQL>
SQL> CREATE OR REPLACE TRIGGER NEW
BEFORE INSERT OR DELETE OR UPDATE ON Chef
FOR EACH ROW
ENABLE
DECLARE
V_USER VARCHAR(20);
BEGIN
SELECT USER INTO V_USER FROM DUAL;
IF INSERTING THEN
DBMS_OUTPUT.PUT_LINE('ROW INSERTED BY ' || V_USER);
ELSIF DELETING THEN
DBMS_OUTPUT.PUT_LINE('ROW DELETED BY ' || V_USER);
ELSIF UPDATING THEN
DBMS_OUTPUT.PUT_LINE('ROW UPDATED BY ' || V_USER);
END IF;
END;
/

Trigger created.

SQL>
```



```
SQL> insert into Chef values(sequence_1.nextval,'NIGELLA', '6767192047', 'GULAB NAGAR', '30000');

1 row created.

SQL> select * from Chef;

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
        41
Gordon
74987484848

   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
Imperial Road
     40000


   CHEF_ID
----------
CHEF_NAME
--------------------------------------------------------------------------------
CONTACT
--------------------------------------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
    SALARY
----------
        61
Jamie
7574194190

   CHEF_ID
----------
```



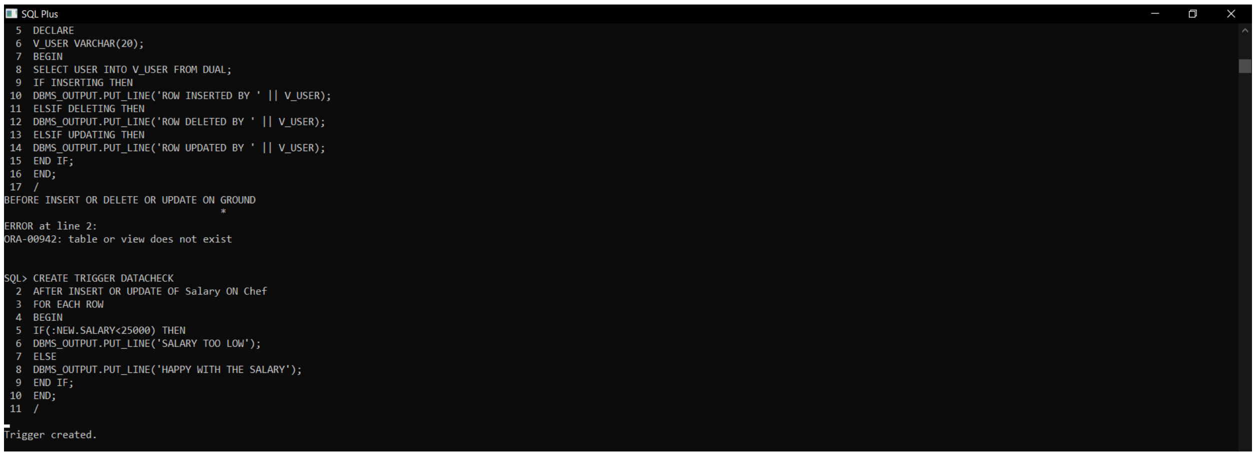
```
SQL> select Chef_Name from Chef;

CHEF_NAME
--------------------------------------------------------------------------------
Gordon
Jamie
Mulan
NIGELLA

SQL> _
```

3. TRIGGER 2: TO CHECK WHETHER SALARY IS TOO LOW

```
CREATE TRIGGER DATACHECK
AFTER INSERT OR UPDATE OF Salary ON Chef
FOR EACH ROW
BEGIN
IF(:NEW.SALARY<25000) THEN
DBMS_OUTPUT.PUT_LINE('SALARY TOO LOW');
```

ELSE
DBMS_OUTPUT.PUT_LINE('HAPPY WITH THE SALARY');
END IF;
END;
/

INSERT INTO Chef VALUES(“VICKY”, “9912038475”, “HIMALAYA”, '29000’);



```
  5  DECLARE
  6  V_USER VARCHAR(20);
  7  BEGIN
  8  SELECT USER INTO V_USER FROM DUAL;
  9  IF INSERTING THEN
 10  DBMS_OUTPUT.PUT_LINE('ROW INSERTED BY ' || V_USER);
 11  ELSIF DELETING THEN
 12  DBMS_OUTPUT.PUT_LINE('ROW DELETED BY ' || V_USER);
 13  ELSIF UPDATING THEN
 14  DBMS_OUTPUT.PUT_LINE('ROW UPDATED BY ' || V_USER);
 15  END IF;
 16  END;
 17  /
BEFORE INSERT OR DELETE OR UPDATE ON GROUND
                                     *
ERROR at line 2:
ORA-00942: table or view does not exist


SQL> CREATE TRIGGER DATACHECK
  2  AFTER INSERT OR UPDATE OF Salary ON Chef
  3  FOR EACH ROW
  4  BEGIN
  5  IF(:NEW.SALARY<25000) THEN
  6  DBMS_OUTPUT.PUT_LINE('SALARY TOO LOW');
  7  ELSE
  8  DBMS_OUTPUT.PUT_LINE('HAPPY WITH THE SALARY');
  9  END IF;
 10  END;
 11  /

Trigger created.
```

CONCLUSION

In Conclusion, the aim of this project was to apply what we learned in our Database Management Systems class, practically, to a basic Restaurant management system. It was under the guidance of our Teacher, Ms. Sindhu who always made sure we understood our concepts well. The functionality of our system caters to the administration side predominantly, keeping the documentation of their employees, orders, bills, etc in place.We hope that with our system, it will be easier to manage a restaurant and keep customers happy. To summarise, we gained a lot of knowledge while working on this project and are thankful for this opportunity to understand a real-life scenario better.