**AMERICAN INTERNATIONAL UNIVERSITY BANGLADESH (AIUB)**

**FACULTY OF SCIENCE & TECHNOLOGY**

A picture containing calendar

Description automatically generated

Course Title

**INTRODUCTION TO DATABASE (CSC2108)**

**Semester:   
Section: [X]**

**TITLE**

**Restaurant Management System**

**Supervised By**

DR. Rajarshri Roy Chowdhury

**Submitted By: Group no: 05**

|  |  |
| --- | --- |
| **Name** | **ID** |
| MD Sadman Al Sakib | 23-53467-3 |
| Ankon Debnath | 23-53468-3 |
| Tanvirul Islam | 23-53474-3 |
| Sania Ahmed | 23-54825-3 |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **TOPICS** | | **Page no.** |
| **Title Page** | | **1** |
| **Table of Content** | | **2** |
| **1.** | **Introduction** | **3** |
| **2.** | **Case Study** | **3** |
| **3.** | **ER Diagram** | **4** |
| **4.** | **Normalization** | **5-6** |
| **5.** | **Finalization** | **7** |
| **6.** | **Table Creation** | **8-10** |
| **7.** | **Data Insertion** | **11-19** |
| **8.** | **Query Test** | **19-30** |
| **9.** | **DB connection** | **31-33** |
|  |  |  |

# **Introduction**

This database management project based on oracle SQL is about the Restaurant management system where user can order food and all the operations from order to delivery can be tracked. We have used the structured query language to perform all the operations. Our contribution to tech sector by this project is to create and manage database of a restaurant management where all the data can be placed and retrieved by using SQL. The restaurant management side can be benefited by this project to store and track data about their management system.

**Case Study**

A restaurant management system needs a robust database to streamline its operations. The system involves various entities: ingredients, food items, chefs, sellers, delivery personnel, customers, and billing. Each food item is made using multiple ingredients and is crafted by skilled chefs. Food items are sent to sellers, who collaborate with delivery personnel to supply orders to customers. Delivery personnel are responsible for ensuring orders reach customers efficiently. The customers reside at specific addresses and place orders that generate bills. The database will maintain relationships such as many-to-many and one-to-many.This system ensures proper tracking of resources, transactions, and interactions within the restaurant's workflow.

# **ER Diagram**

**A diagram of a company

Description automatically generated**

**NORMALIZATION**

**Table: (Food – Seller)**

**UNF**: F\_ID,F\_name,F\_rating,F\_price,S\_ID,S\_name, S\_Phone,S\_startingdate

S\_Phone = multivalued attribute

**1NF**: F\_ID,F\_name,F\_rating,F\_price,S\_ID,S\_name,S\_startingdate

**2NF:** F\_ID,F\_name,F\_rating,F\_price

S\_ID,S\_Phone

S\_ID,S\_name,S\_startingdate

**3NF**: F\_ID,F\_name,F\_rating,F\_price

S\_ID,S\_Phone

S\_ID,S\_name,S\_startingdate

S\_ID,F\_ID

Foreign key: S\_ID, primary key : F\_ID

**Table: (Seller – Deliveryman)**

**UNF**: S\_ID,S\_name,S\_phone,S\_startingdate,DM\_ID,DM\_name,Transportnum

S\_Phone = Multivalued Attribute

**1NF**: S\_ID,S\_name,S\_statringdate

DM\_ID,DM\_name,Transportnum

**2NF**: S\_ID,S\_phone

S\_ID,S\_name,S\_startingdate

DM\_ID,DM\_name,Transportnum

**3NF**: S\_ID,S\_phone

S\_ID,S\_name,S\_startingdate

DM\_ID,DM\_name,Transportnum

Foreign key: S\_ID Primary Key: D\_ID

**Table: (Deliveryman – Customer)**

**UNF**: DM\_ID,DM\_name,Transportnum,C\_ID,C\_name,C\_address

**1NF**: DM\_ID,DM\_name,Transportnum,C\_ID,C\_name

**2NF**: C\_ID,C\_name,Area,Street,House

DM\_ID,DM\_name,Transportnum

DM\_ID,C\_ID

**3NF**: C\_ID,C\_name,Area,Street,House

DM\_ID,DM\_name,Transportnum

DM\_ID,C\_ID

Foreign key: C\_ID Primary Key: DM\_ID

**Table: (Food-Chef)**

**UNF**: F\_ID,F\_name,F\_rating,F\_price,CH\_ID,CH\_name,CH\_email

CH\_email = Multivalued Attribute

**1NF**: F\_ID,F\_name,F\_rating,F\_price,CH\_ID,CH\_name

**2NF**: F\_ID,F\_name,F\_rating,F\_price,CH\_ID,CH\_name

CH\_ID,CH\_email

**3NF**: F\_ID,F\_name,F\_rating,F\_price

CH\_ID,CH\_name,CH\_email

F\_ID,CH\_ID

Foreign key: F\_ID Primary Key : CH\_ID

**Table: (Customer-Bill)**

**UNF**: C\_ID,C\_name,C\_Address,B\_ID,Amount

**1NF**: C\_ID,C\_name,Area,street,house,B\_ID,Amount

**2NF**: C\_ID,C\_name,Area,street,House

B\_ID,C\_ID,Amount

**3NF**: C\_ID,C\_name,Area,street,House

B\_ID,C\_ID,Amount

Foreign key: B\_ID Primary Key: C\_ID

Table: (Food-Ingredient)

UNF: F\_ID,F\_name,F\_rating,F\_price,I\_ID,I\_name

1NF: F\_ID,F\_name,F\_rating,F\_price

I\_ID,I\_name

2NF: F\_ID,F\_name,F\_rating,F\_price

F\_ID,I\_ID

Foreign key: F\_ID Primary Key: I\_ID

**FINALIZATION**

**Red: Primary Key**

**Green: Foreign key**

F\_ID,F\_name,F\_rating,F\_price

S\_ID,S\_Phone

S\_ID,S\_name,S\_startingdate

S\_ID,F\_ID

DM\_ID,DM\_name,Transportnum

C\_ID,C\_name,Area,Street,House

DM\_ID,C\_ID

CH\_ID,CH\_name,CH\_email

F\_ID,CH\_ID

I\_ID,I\_name

F\_ID,I\_ID

B\_ID,C\_ID,Amount

# **Table Creation (DDL Operations)**

|  |  |  |
| --- | --- | --- |
| StudentID1: 23-53468-3 Name: Ankon Debnath | StudentID3: Name: | |
| StudentID2: Name: | StudentID4: Name: | |
| **CO4**: Creating DML, DDL using Oracle and connection with ODBC/JDBC for existing JAVA application | | |
| **PO-e-2:** Use modern engineering and IT tools for prediction and modeling of complex computer science and engineering problem | | Marks |

**Table: Food**

CREATE TABLE Food (

F\_ID NUMBER PRIMARY KEY,

F\_name VARCHAR2(50) NOT NULL,

F\_rating NUMBER(3, 2) NOT NULL,

F\_price NUMBER(10, 2) NOT NULL

);

**Table: Seller**

CREATE TABLE Seller (

S\_ID NUMBER(10) PRIMARY KEY,

S\_name VARCHAR2(100) NOT NULL,

S\_startingdate DATE NOT NULL

);

**Table: Seller\_Contact**

CREATE TABLE Seller\_Contact (

S\_ID INT PRIMARY KEY,

S\_Phone VARCHAR(15) NOT NULL

);

**Table: Seller\_Food**

CREATE TABLE Seller\_Food (

S\_ID NUMBER,

F\_ID NUMBER,

FOREIGN KEY (F\_ID) REFERENCES Food(F\_ID)

);

**Table: Delivery\_Man**

CREATE TABLE Delivery\_Man (

DM\_ID NUMBER PRIMARY KEY,

DM\_name VARCHAR2(50),

Transportnum VARCHAR2(20)

);

**Table: Customer**

CREATE TABLE Customer (

C\_ID NUMBER PRIMARY KEY,

C\_name VARCHAR2(50),

Area VARCHAR2(50),

Street VARCHAR2(50),

House VARCHAR2(50)

);

**Table: Delivery\_Customer**

CREATE TABLE Delivery\_Customer (

DM\_ID NUMBER PRIMARY KEY,

C\_ID NUMBER,

CONSTRAINT fk\_c FOREIGN KEY (C\_ID) REFERENCES Customer1(C\_ID)

);

**Table: Chef**

CREATE TABLE Chef (

CH\_ID NUMBER PRIMARY KEY,

CH\_name VARCHAR2(50),

CH\_email VARCHAR2(100)

);

**Table Chef\_Food:**

CREATE TABLE Chef\_Food (

F\_ID NUMBER PRIMARY KEY,

CH\_ID NUMBER,

CONSTRAINT fk\_ch\_id FOREIGN KEY (CH\_ID) REFERENCES Chef(CH\_ID)

);

**Table: Ingredient**

CREATE TABLE Ingredient (

I\_ID NUMBER PRIMARY KEY,

I\_name VARCHAR2(50)

);

**Table: Food\_Ingredient**

CREATE TABLE Food\_Ingredient (

I\_ID INT PRIMARY KEY,

F\_ID INT,

FOREIGN KEY (F\_ID) REFERENCES Food (F\_ID)

);

**Table: Bill**

CREATE TABLE Bill (

B\_ID NUMBER PRIMARY KEY,

C\_ID NUMBER,

Amount NUMBER(10, 2),

CONSTRAINT fk\_c\_id FOREIGN KEY (C\_ID) REFERENCES Customer1(C\_ID)

);

# **Inserted Values in the tables**

**Table: Food**

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (1, 'Special Breakfast', 4.2, 250.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (2, 'Eggy Breakfast', 4.1, 200.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (3, 'Bready Breakfast', 4.1, 220.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (4, 'Boiled Egg', 4.0, 50.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (5, 'Salad With Egg', 4.1, 180.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (6, 'Lemonade', 4.0, 120.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (7, 'Milk Shake', 4.5, 180.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (8, 'Orange Juice', 4.4, 150.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (9, 'Mango Juice', 4.5, 170.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (10, 'Watermelon Juice', 4.6, 160.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (11, 'North Indian Platter', 4.3, 300.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (12, 'Chicken With Plain Rice', 4.1, 250.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (13, 'Non Veg Special Plate', 4.4, 350.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (14, 'Kacchi Biriyani', 4.5, 400.00);

INSERT INTO Food (F\_ID, F\_name, F\_rating, F\_price) VALUES (15, 'Chicken Tehari', 4.5, 380.00);

A table with a list of food

Description automatically generated

**Table: Seller**

INSERT INTO Seller (S\_ID, S\_name, S\_startingdate) VALUES (1, 'Sadman', TO\_DATE('2020-01-15', 'YYYY-MM-DD'));

INSERT INTO Seller (S\_ID, S\_name, S\_startingdate) VALUES (2, 'Ankon', TO\_DATE('2018-07-10', 'YYYY-MM-DD'));

INSERT INTO Seller (S\_ID, S\_name, S\_startingdate) VALUES (3, 'Niloy', TO\_DATE('2021-03-20', 'YYYY-MM-DD'));

INSERT INTO Seller (S\_ID, S\_name, S\_startingdate) VALUES (4, 'Rajob', TO\_DATE('2019-11-05', 'YYYY-MM-DD'));

INSERT INTO Seller (S\_ID, S\_name, S\_startingdate) VALUES (5, 'Sakib', TO\_DATE('2022-05-25', 'YYYY-MM-DD'));

A screenshot of a computer

Description automatically generated

**Table: Seller\_Contact**

INSERT ALL

INTO Seller\_Contact (S\_ID, S\_Phone) VALUES (1, '1234567890')

INTO Seller\_Contact (S\_ID, S\_Phone) VALUES (2, '9876543210')

INTO Seller\_Contact (S\_ID, S\_Phone) VALUES (3, '5678901234')

INTO Seller\_Contact (S\_ID, S\_Phone) VALUES (4, '4561237890')

INTO Seller\_Contact (S\_ID, S\_Phone) VALUES (5, '7890123456')

SELECT \* FROM DUAL;

A screenshot of a phone number

Description automatically generated

**Table: Seller\_Food**

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (1, 1);

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (1, 2);

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (1, 3);

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (2, 4);

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (2, 5);

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (3, 11);

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (3, 12);

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (4, 13);

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (5, 14);

INSERT INTO Seller\_Food1 (S\_ID, F\_ID) VALUES (5, 15);

A screenshot of a number

Description automatically generated

**Table: Delivery\_Man**

INSERT INTO Delivery\_Man (DM\_ID, DM\_name, Transportnum) VALUES (1, 'Rajob', 'TN001');

INSERT INTO Delivery\_Man (DM\_ID, DM\_name, Transportnum) VALUES (2, 'Shaj', 'TN002');

INSERT INTO Delivery\_Man (DM\_ID, DM\_name, Transportnum) VALUES (3, 'Avi', 'TN003');

INSERT INTO Delivery\_Man (DM\_ID, DM\_name, Transportnum) VALUES (4, 'Sokal', 'TN004');

INSERT INTO Delivery\_Man (DM\_ID, DM\_name, Transportnum) VALUES (5, 'Reyan', 'TN005');

A screenshot of a computer

Description automatically generated

**Table: Customer**

INSERT INTO Customer1 (C\_ID, C\_name, Area, Street, House) VALUES (1, 'Dhrubo', 'Banani', 'Road 12', 'House 34');

INSERT INTO Customer1 (C\_ID, C\_name, Area, Street, House) VALUES (2, 'Bishal', 'Gulshan', 'Street 5', 'House 78');

INSERT INTO Customer1 (C\_ID, C\_name, Area, Street, House) VALUES (3, 'Utsha', 'Dhanmondi', 'Avenue 4', 'Flat 16');

INSERT INTO Customer1 (C\_ID, C\_name, Area, Street, House) VALUES (4, 'Rittik', 'Uttara', 'Sector 7', 'House 23');

INSERT INTO Customer1 (C\_ID, C\_name, Area, Street, House) VALUES (5, 'Labib', 'Mirpur', 'Block C', 'House 45');

A table with text on it

Description automatically generated

**Table: Delivery\_Customer**

INSERT INTO Delivery\_Customer (DM\_ID, C\_ID) VALUES (1, 1);

INSERT INTO Delivery\_Customer (DM\_ID, C\_ID) VALUES (2, 2);

INSERT INTO Delivery\_Customer (DM\_ID, C\_ID) VALUES (3, 3);

INSERT INTO Delivery\_Customer (DM\_ID, C\_ID) VALUES (4, 4);

INSERT INTO Delivery\_Customer (DM\_ID, C\_ID) VALUES (5, 5);

A screenshot of a computer

Description automatically generated

**Table: Chef**

INSERT INTO Chef (CH\_ID,CH\_name,CH\_email) VALUES (1,'Siam','siam@gmail.com');

INSERT INTO Chef (CH\_ID,CH\_name,CH\_email) VALUES (2,'Shishir','shishir@gmail.com');

INSERT INTO Chef (CH\_ID,CH\_name,CH\_email) VALUES (3,'Shaj','shaj@gmail.com');

INSERT INTO Chef (CH\_ID,CH\_name,CH\_email) VALUES (4,'Mahir','mahir@gmail.com');

INSERT INTO Chef (CH\_ID,CH\_name,CH\_email) VALUES (5,'Rehab','rehab@gmail.com');

A screenshot of a computer

Description automatically generated

**Table Chef\_Food:**

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (1, 1);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (2, 1);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (3, 1);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (4, 2);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (5, 2);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (6, 2);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (7, 3);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (8, 3);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (9, 3);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (10, 4);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (11, 4);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (12, 4);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (13, 5);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (14, 5);

INSERT INTO Chef\_Food (F\_ID, CH\_ID) VALUES (15, 5);

A screenshot of a number

Description automatically generated

**Table: Ingredient**

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (1, 'Basil');

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (2, 'Olive Oil');

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (3, 'Parmesan Cheese');

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (4, 'Chili Powder');

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (5, 'Cumin');

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (6, 'Honey');

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (7, 'Ginger');

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (8, 'Coriander');

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (9, 'Yeast');

INSERT INTO Ingredient (I\_ID, I\_name) VALUES (10, 'Vanilla Extract');

A screenshot of a menu

Description automatically generated

**Table: Food\_Ingredient**

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (1, 1);

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (2, 2);

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (3, 3);

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (4, 4);

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (5, 5);

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (6, 11);

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (7, 12);

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (8, 13);

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (9, 14);

INSERT INTO Food\_Ingredient (I\_ID, F\_ID) VALUES (10, 15);

A screenshot of a number

Description automatically generated

**Table: Bill**

INSERT INTO Bill (B\_ID, C\_ID, Amount) VALUES (1, 1, 320.00);

INSERT INTO Bill (B\_ID, C\_ID, Amount) VALUES (2, 2, 280.00);

INSERT INTO Bill (B\_ID, C\_ID, Amount) VALUES (3, 3, 300.00);

INSERT INTO Bill (B\_ID, C\_ID, Amount) VALUES (4, 4, 250.00);

INSERT INTO Bill (B\_ID, C\_ID, Amount) VALUES (5, 5, 290.00);

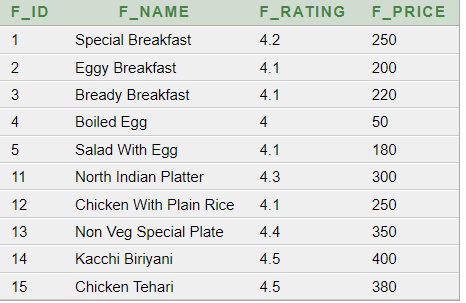
A screenshot of a table

Description automatically generated

# **Query Test in DB**

1.To find out the food table elements using view

create view foodview as select F\_ID,F\_name,F\_rating,F\_price from Food select \* from foodview



2.To find out the highest priced food using multiple row function MAX():

SELECT MAX(F\_price) as MAX FROM Food

A close-up of a number

Description automatically generated

3. To find out the food with rating more than equal to 4.5

SELECT F\_name FROM Food WHERE F\_rating >= 4.5

A screenshot of a computer

Description automatically generated

4. To select the seller who have joined after 1st January 2020:

SELECT S\_ID,S\_name FROM Seller WHERE S\_startingdate > TO\_DATE('01-01-2020', 'DD-MM-YYYY');

A screenshot of a calendar

Description automatically generated  
5. To update the seller name sadman (from short name) to Sadman Al sakib (full name):

UPDATE Seller SET S\_name = 'Sadman Al sakib' WHERE S\_name = 'Sadman'

A screenshot of a computer

Description automatically generated

6. showing the customers name by using single row function UPPER():

SELECT C\_ID,UPPER(C\_name) FROM Customer1

A screenshot of a computer

Description automatically generated

7. To Find the name of the food item with the highest rating: (single row sub query)

SELECT F\_ID,F\_name,F\_rating FROM Food WHERE F\_rating = (SELECT MAX(F\_rating)FROM Food)

A close up of a name

Description automatically generated

8.To Find the price of the food item with the lowest rating: (single row sub query)

SELECT F\_ID,F\_name,F\_rating FROM Food WHERE F\_rating = (SELECT MIN(F\_rating)FROM Food)

A close-up of a name

Description automatically generated

9.To Find the names of food items with a price greater than the average price of all food (multiple row sub query)

SELECT F\_NAME FROM Food WHERE F\_PRICE>(SELECT AVG(F\_PRICE) FROM Food);

A screenshot of a menu

Description automatically generated

10. To Find the emails of chefs whose CH\_ID is less than 4: (multiple row sub query):

SELECT CH\_EMAIL FROM Chef WHERE CH\_ID IN(SELECT CH\_ID FROM Chef WHERE CH\_ID < 4);

A screenshot of a email

Description automatically generated

11.To Update the phone number of the seller with ID 3:

UPDATE Seller\_Contact SET S\_phone = 1615828990 WHERE S\_ID = 3

A screenshot of a phone number

Description automatically generated

12.To update the table Seller to add the new column named “Salary”:

ALTER TABLE Seller ADD (SALARY INT)

**inserting values:**

UPDATE Seller SET SALARY = 25000 WHERE S\_ID = 1

UPDATE Seller SET SALARY = 28000 WHERE S\_ID = 2

UPDATE Seller SET SALARY = 22000 WHERE S\_ID = 3

UPDATE Seller SET SALARY = 21000 WHERE S\_ID = 4

UPDATE Seller SET SALARY = 28000 WHERE S\_ID = 5

A screenshot of a calendar

Description automatically generated

13. To find out the seller who earns the maximum salary:

SELECT MAX(SALARY) FROM Seller

A close up of a sign

Description automatically generated

14. To update the table chef by adding new column named location:

ALTER TABLE Chef ADD(Location Varchar(10))

**Inserting values:**

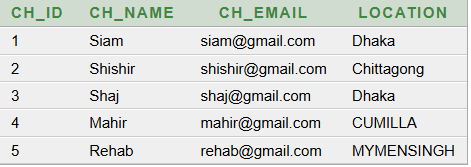
UPDATE Chef SET Location = 'Dhaka' WHERE CH\_ID = 1

UPDATE Chef SET Location = 'Chittagong' WHERE CH\_ID = 2

UPDATE Chef SET Location = 'Dhaka' WHERE CH\_ID = 3

UPDATE Chef SET Location = 'CUMILLA' WHERE CH\_ID = 4

UPDATE Chef SET Location = 'MYMENSINGH' WHERE CH\_ID = 5



15. To find out the number of chef who live out side dhaka:

SELECT COUNT(\*) AS OUSIDE\_DHAKA FROM Chef WHERE LOCATION <> 'Dhaka'

A close-up of a sign

Description automatically generated

16. To update the table Delivery\_Man to add column salary:

ALTER TABLE Delivery\_Man ADD (Salary INT)

**Inserting values:**

UPDATE Delivery\_Man SET Salary = 15000 WHERE DM\_ID = 1

UPDATE Delivery\_Man SET Salary = 12000 WHERE DM\_ID = 2

UPDATE Delivery\_Man SET Salary = 16000 WHERE DM\_ID = 3

UPDATE Delivery\_Man SET Salary = 14000 WHERE DM\_ID = 4

UPDATE Delivery\_Man SET Salary = 12000 WHERE DM\_ID = 5

A screenshot of a number

Description automatically generated

17. To find out the average salary of the delivery men:

SELECT AVG(Salary) FROM Delivery\_Man

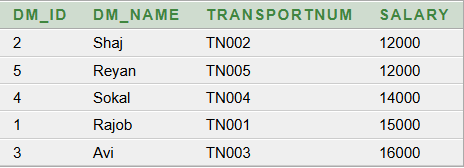
A close-up of a sign

Description automatically generated

18. Setting the table order by salary in ascending order:

SELECT \* FROM Delivery\_Man

ORDER BY Salary ASC



19. To find out the maximum and minimum amount of bill:

SELECT MAX(Amount) AS MAXIMUM\_AMOUNT , MIN(Amount) AS MINIMUM\_AMOUNT FROM Bill

A close-up of a number

Description automatically generated

20. Renaming the column Salary in Seller table to S\_Salary:

ALTER TABLE Seller RENAME COLUMN SALARY TO S\_Salary

A screenshot of a calendar

Description automatically generated

21. Renaming the column salary in Deliveryman table to DM\_Salary:

ALTER TABLE Delivery\_Man RENAME COLUMN Salary TO DM\_Salary

A screenshot of a computer

Description automatically generated

22. Adding new column in chef column as position and salary:

ALTER TABLE ChefADD (Position varchar(10));

UPDATE Chef SET position = 'senior' WHERE CH\_ID = 1

UPDATE Chef SET position = 'senior' WHERE CH\_ID = 2

UPDATE Chef SET position = 'junior' WHERE CH\_ID = 3

UPDATE Chef SET position = 'senior' WHERE CH\_ID = 4

UPDATE Chef SET position = 'junior' WHERE CH\_ID = 5

ALTER TABLE ChefADD (CH\_Salary INT);

UPDATE Chef SET CH\_Salary = 35000 WHERE CH\_ID = 1

UPDATE Chef SET CH\_Salary= 30000 WHERE CH\_ID = 2

UPDATE Chef SET CH\_Salary= 32000 WHERE CH\_ID = 3

UPDATE Chef SET CH\_Salary= 35000 WHERE CH\_ID = 4

UPDATE Chef SET CH\_Salary= 30000 WHERE CH\_ID = 5

A screenshot of a computer

Description automatically generated

23.To find out the average salary group by position of chef table:

SELECT UPPER(Position),AVG(CH\_Salary) AS "AVERAGE SALARY BY POSITION"

FROM Chef

GROUP BY Position

A screenshot of a number

Description automatically generated

24.To find out the name of the deliverymen whose name start with ‘s’:

SELECT DM\_Name FROM Delivery\_Man WHERE DM\_Name Like 'S%'

A screenshot of a computer

Description automatically generated

25. To find out which delivery man delivers to which customer and adding “delivers to” between two columns:

SELECT Delivery\_Man.DM\_Name || ' Delivers To '|| Customer1.C\_Name AS Delivery\_Process

FROM Delivery\_Man CROSS JOIN Customer1

A screenshot of a list of rows

Description automatically generated

26.To show the distinct location of the chef table:

SELECT DISTINCT Location FROM Chef

A screenshot of a phone

Description automatically generated

27. To add bonus in salary for sellers:  
SELECT S\_Name,S\_Salary+(0.25\*S\_Salary) AS BONUS FROM Seller

A screenshot of a computer screen

Description automatically generated

28. Using AND:

SELECT \* FROM Chef

WHERE LOCATION = 'Dhaka' AND CH\_Salary >= 32000

A screenshot of a phone

Description automatically generated

29. Using Replace:

SELECT S\_Name , REPLACE(S\_Name,'S','A') AS "REPLACED NAMES" FROM SELLER

A screenshot of a computer

Description automatically generated

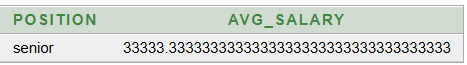
30. To Find the average salary of chefs grouped by their position but only include positions where the average salary is greater than 30000:

SELECT Position, AVG(CH\_SALARY) AS AVG\_SALARY

FROM Chef

GROUP BY Position

HAVING AVG(CH\_SALARY) > 32000;



**Table Joining:**

**INNER JOIN: To find out the food made by each chef**

1.SELECT Chef.CH\_name,Food.F\_name

FROM Chef

INNER JOIN Food ON Chef.CH\_ID = Food.F\_ID

A menu of a breakfast

Description automatically generated

**2.LEFT JOIN : To find out the seller name who sold foods**

SELECT Seller.S\_name,Food.F\_name

FROM Seller

LEFT JOIN Food ON Seller.S\_ID = Food.F\_ID

A screenshot of a menu

Description automatically generated

3.**To find out the name of deliveryman who delivers to customer**

SELECT DM\_name,C\_name

FROM Delivery\_Man

CROSS JOIN Customer1

A screenshot of a phone

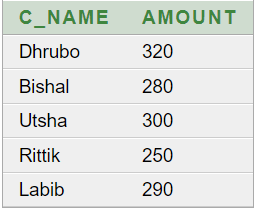
Description automatically generated

**4. Right Join: To Find out the amount of bill by customer name**

SELECT Customer1.C\_name,Bill.Amount

FROM Customer1

RIGHT JOIN Bill ON Bill.B\_ID = Customer1.C\_ID



**Description of Tables:**

1. desc Food

A screenshot of a computer

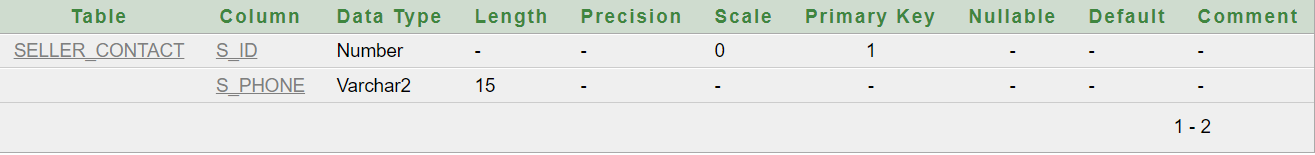
Description automatically generated

1. desc Seller

A screenshot of a computer

Description automatically generated

1. DESC Seller\_Contact



1. DESC Seller\_Food1

A screenshot of a computer

Description automatically generated

1. desc Delivery\_Man

A screenshot of a computer

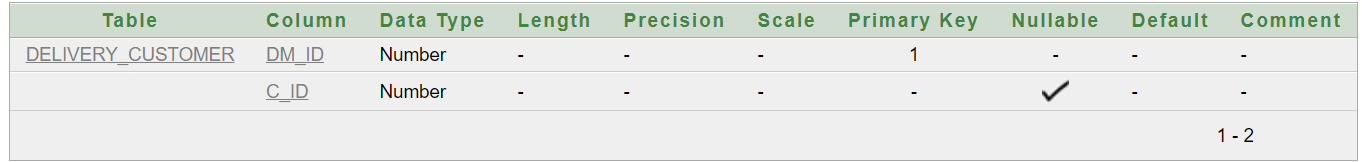
Description automatically generated

1. desc Customer1

A screenshot of a computer

Description automatically generated

1. desc Delivery\_Customer



1. desc Chef

A screenshot of a computer

Description automatically generated

1. desc Chef\_Food

A screenshot of a computer

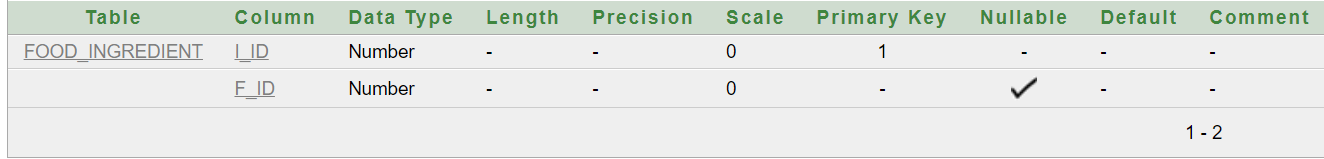
Description automatically generated

1. desc Ingredient

A screenshot of a computer

Description automatically generated

1. desc Food\_Ingredient



12.desc Bill

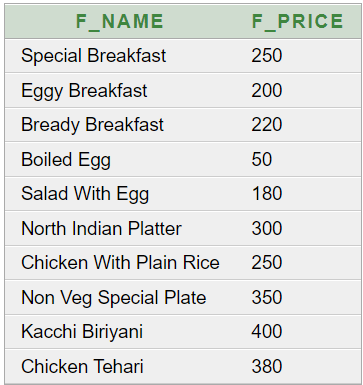
A screenshot of a computer

Description automatically generated

**Creating View from tables:**

**view of food table:**

create view food\_view as select F\_name,F\_price from Food select \* from food\_view



**view of Delivery\_Man table:**

create view Deliveryman\_Details(Deliveryman\_name,Delivery\_man\_transport) as

select DM\_name,Transportnum from Delivery\_Man select \* from Deliveryman\_Details

A screenshot of a delivery list

Description automatically generated

**Compelx View:**

CREATE VIEW Chef\_Food\_Seller\_View AS

SELECT

Chef.CH\_name AS Chef\_Name,

COUNT(Food.F\_ID) AS Number\_of\_Foods\_Prepared,

Seller.S\_name AS Seller\_Name

FROM

Chef

INNER JOIN Food ON Chef.CH\_ID = Food.F\_ID

INNER JOIN Seller\_Food1 ON Food.F\_ID = Seller\_Food1.F\_ID

INNER JOIN Seller ON Seller\_Food1.S\_ID = Seller.S\_ID

GROUP BY Chef.CH\_name, Seller.S\_name;

A screenshot of a computer

Description automatically generated

**CONNECTING WITH JAVA**

**Process we followed to add:**

1.Downloaded the mysql java connector

2. Downloaded the XAMPP server

3. Opened the xampp server and started the apache and mysql admin

4. Then we created a database according to our project name

5. Created the the table and inserted values in table we wanted to add “Food”

The Table in mysql:

A screenshot of a menu

Description automatically generated

6. Then We used visual studio code for java. In vs code we added the library we downloaded “my sql connector”

7. After writing the code we run the code and got the desired output.

**JAVA CONNECTION CODE:**

import java.sql.\*;

public class Customer {

    public static void main(String[] *args*) {

        String username = "root";

        String password = "";

        String url = "jdbc:mysql://localhost:3306/restaurant\_management\_system";

        Connection conn = null;

        Statement stmnt = null;

        ResultSet rs = null;

        try {

            Class.forName("com.mysql.cj.jdbc.Driver");

            conn = DriverManager.getConnection(url, username, password);

            System.out.println("Connected!!!");

            stmnt = conn.createStatement();

            rs = stmnt.executeQuery("SELECT \* FROM Food");

            while (rs.next()) {

                System.out.println("\tF\_ID: " + rs.getInt(1));

                System.out.print("\tF\_Name: " + rs.getString(2));

                System.out.print("\tF\_Rating: " + rs.getDouble(3));

                System.out.print("\tF\_Price: " + rs.getDouble(4));

                System.out.println();

            }

        }

        catch (ClassNotFoundException *e*) {

            System.err.println("JDBC Driver not found: " + e.getMessage());

        }

        catch (SQLException *e*) {

            System.err.println("SQL Exception: " + e.getMessage());

        }

        finally {

            try {

                if (rs != null) rs.close();

                if (stmnt != null) stmnt.close();

                if (conn != null) conn.close();

            }

            catch (SQLException *e*) {

                System.err.println("Error closing resources: " + e.getMessage());

            }

        }

    }

}

As I have entered the query to show all the elements of Food table, the output is given below:

A screenshot of a computer program

Description automatically generated

**CONCLUSION**

The **Restaurant Management System** project showcases the design and implementation of a relational database to efficiently manage food items, sellers, customers, delivery personnel, chefs, ingredients, and billing processes. By adhering to 3NF normalization, the system ensures data integrity, reduces redundancy, and supports seamless data operations. The integration of structured relationships between tables highlights the practical application of database principles in real-world scenarios.

Future enhancements include implementing online ordering, reservation management, personalized menu suggestions, and employee scheduling. Incorporating analytics for revenue tracking, cloud deployment for scalability, and secure payment gateways can further improve functionality. This project lays a solid foundation for automating and optimizing restaurant operations while preparing for future technological advancements.