**FoodDB Codes**

CREATE DATABASE FoodserviceDB;

SELECT TOP 5 \* FROM Restaurant;

SELECT TOP 5 \* FROM Ratings;

SELECT TOP 5 \* FROM Consumer;

SELECT TOP 5 \* FROM Restaurant\_Cuisines;

SELECT MIN(Age)

FROM Consumer

JOIN Restaurant

-- Adding primary and foreign key constraints --

ALTER TABLE Ratings

ADD CONSTRAINT FK\_Ratings\_Consumer

FOREIGN KEY (Consumer\_id) REFERENCES Consumer(Consumer\_id);

ALTER TABLE Ratings

ADD CONSTRAINT FK\_Ratings\_Restaurant

FOREIGN KEY (Restaurant\_id) REFERENCES Restaurant(Restaurant\_id);

ALTER TABLE Restaurant\_Cuisines

ADD CONSTRAINT FK\_Cuisines\_Restaurant

FOREIGN KEY (Restaurant\_id) REFERENCES Restaurant(Restaurant\_id);

-- Part2:

-- 1. Write a query that lists all restaurants with a Medium range price with open area,serving Mexican food --

SELECT R.Restaurant\_ID,RC.Cuisine,R.Price,R.Area

FROM Restaurant R

JOIN Restaurant\_Cuisines RC

ON R.Restaurant\_ID = RC.Restaurant\_ID

WHERE Price = 'Medium' AND Area = 'Open' AND Cuisine = 'Mexican' ;

/\*2. Write a query that returns the total number of restaurants who have the overall rating

as 1 and are serving Mexican food. Compare the results with the total number of

restaurants who have the overall rating as 1 serving Italian food (please give

explanations on their comparison) \*/

-- for Mexican food --

SELECT COUNT (\*) AS Total, Cuisine

FROM Restaurant\_Cuisines RC

JOIN Ratings RS

ON RC.Restaurant\_ID = RS.Restaurant\_ID

WHERE Overall\_Rating = '1' AND Cuisine = 'Mexican'

GROUP BY Cuisine

UNION ALL

-- For Italian food --

SELECT COUNT (\*) AS Total, Cuisine

FROM Restaurant\_Cuisines RC

JOIN Ratings RS

ON RC.Restaurant\_ID = RS.Restaurant\_ID

WHERE Overall\_Rating = '1' AND Cuisine = 'Italian'

GROUP BY Cuisine;

--This results shows Mexican Cuisine have alot more 1 ratings than Italian , this also indicates lots of areas mexican cuisine can mprove on --

/\*3 Calculate the average age of consumers who have given a 0 rating to the 'Service\_rating'

column. (NB: round off the value if it is a decimal)\*/

SELECT AVG(Age) AS AvgAge

FROM Consumer C

JOIN Ratings RS

ON C.Consumer\_ID = RS.Consumer\_ID

WHERE Service\_Rating = '0';

/\* 4. Write a query that returns the restaurants ranked by the youngest consumer. You

should include the restaurant name and food rating that is given by that customer to

the restaurant in your result. Sort the results based on food rating from high to low.\*/

SELECT R.Name AS RestaurantName, RS.Food\_Rating, RS.Food\_Rating AS ConsumerFoodRating

FROM Restaurant R

JOIN Ratings RS ON R.Restaurant\_ID = RS.Restaurant\_ID

JOIN Consumer C ON RS.Consumer\_ID = C.Consumer\_ID

WHERE C.Age = (

SELECT MIN(Age)

FROM Consumer

)

ORDER BY RS.Food\_Rating DESC;

--

WITH YoungestConsumers AS (

SELECT Ra.Restaurant\_id,

MIN(C.Age) AS YoungestAge

FROM Ratings Ra

JOIN Consumer C ON Ra.Consumer\_id = C.Consumer\_id

GROUP BY Ra.Restaurant\_id

),

YoungestConsumerRatings AS (

SELECT R.Restaurant\_id,

R.Name AS RestaurantName,

Y.YoungestAge,

MAX(Ra.Food\_Rating) AS FoodRating

FROM Ratings Ra

JOIN Consumer C ON Ra.Consumer\_id = C.Consumer\_id

JOIN YoungestConsumers Y ON Ra.Restaurant\_id = Y.Restaurant\_id AND C.Age = Y.YoungestAge

JOIN Restaurant R ON R.Restaurant\_id = Ra.Restaurant\_id

GROUP BY R.Restaurant\_id, R.Name, Y.YoungestAge

)

SELECT RestaurantName, Restaurant\_id, YoungestAge, FoodRating

FROM YoungestConsumerRatings

ORDER BY FoodRating DESC;

/\*

i was trying to use case but had this error ' Msg 245, Level 16, State 1, Line 2

Conversion failed when converting the varchar value 'Youngest' to data type tinyint'

SELECT

R.Name AS RestaurantName,

RS.Food\_Rating,

CASE

WHEN C.Age BETWEEN 1 AND 18 THEN 'Youngest'

WHEN C.Age BETWEEN 19 AND 40 THEN 'Adult'

WHEN C.Age BETWEEN 41 AND 100 THEN 'Old'

END AS AgeGroup

FROM

Ratings RS

JOIN

Restaurant R ON RS.Restaurant\_ID = R.Restaurant\_ID

JOIN

Consumer C ON RS.Consumer\_ID = C.Consumer\_ID

WHERE Age = 'Youngest'

ORDER BY

Age,

RS.Food\_Rating DESC; \*/

/\* 5. Write a stored procedure for the query given as:

Update the Service\_rating of all restaurants to '2' if they have parking available, either

as 'yes' or 'public'\*/

CREATE PROCEDURE UpdateServiceRatingSForParking

AS

BEGIN

-- Update the Service\_rating to '2' for restaurants with parking available as 'yes' or 'public'

UPDATE Ratings

SET Service\_Rating = 2

WHERE Restaurant\_ID IN (

SELECT Restaurant\_ID

FROM Restaurant

WHERE Parking IN ('yes', 'public')

);

END;

EXEC UpdateServiceRatingSForParking;‹

/\* You should also write four queries of your own and provide a brief explanation of the

results which each query returns. You should make use of all of the following at least

once:

Nested queries-EXISTS

Nested queries-IN

System functions

Use of GROUP BY, HAVING and ORDER BY clauses \*/

/\*

Nested queries-EXISTS ; The query returns the consumer ID, city, and marital status of consumers who use a car for transportation and have rated a restaurant that serves Mexican cuisine.

The EXISTS clause ensures that only those consumers who meet these conditions are included in the result.\*/

Nested queries-EXISTS ; The query returns the consumer ID, city, and marital status of consumers who use a car for transportation and have rated a restaurant that serves Mexican cuisine.

The EXISTS clause ensures that only those consumers who meet these conditions are included in the result.

SELECT C.Consumer\_ID AS ConsumerID, C.City, C.Marital\_Status

FROM Consumer C

WHERE EXISTS (

SELECT 1

FROM Ratings R

JOIN Restaurant\_Cuisines RC ON R.Restaurant\_ID = RC.Restaurant\_ID

WHERE R.Consumer\_ID = C.Consumer\_ID

AND C.Transportation\_Method = 'Car'

AND RC.Cuisine = 'Mexican'

);

/\* Nested queries-IN; The query returns the consumer ID, city, and marital status of consumers who use a car for transportation and have rated a restaurant with a price categorized as 'yes' or 'Low'.

The IN clause ensures that only consumers meeting these conditions are included in the result.\*/

SELECT C.Consumer\_ID AS ConsumerID, C.City, C.Marital\_Status

FROM Consumer C

WHERE C.Consumer\_ID IN (

SELECT DISTINCT RS.Consumer\_ID

FROM Ratings RS

JOIN Restaurant R ON RS.Restaurant\_ID = R.Restaurant\_ID

WHERE R.Price IN ('yes', 'Low')

AND C.Transportation\_Method = 'Car'

);

/\* System Function \* Use count to find the total number of people with marital status married.\*/

System Function \* Use count to find the total number of people with marital status married

SELECT COUNT(\*) AS MarriedConsumersCount

FROM Consumer

WHERE Marital\_Status = 'Married';

/\* Use of GROUP BY, HAVING and ORDER BY clauses ;these are cities where the average age of consumers who have a

'Drink\_Level' of 'Abstemious' is less than 30, along with the corresponding average age, ordered by the average age in descending order.

This can provide insights into which cities have relatively younger consumers who are abstemious.\*/

Use of GROUP BY, HAVING and ORDER BY clauses ;these are cities where the average age of consumers who have a

'Drink\_Level' of 'Abstemious' is less than 30, along with the corresponding average age, ordered by the average age in descending order.

This can provide insights into which cities have relatively younger consumers who are abstemious

SELECT

C.City,

AVG(C.Age) AS AverageAge

FROM

Consumer C

WHERE

C.Drink\_Level = 'Abstemious'

GROUP BY

C.City

HAVING

AVG(C.Age) < 30

ORDER BY

AverageAge DESC;