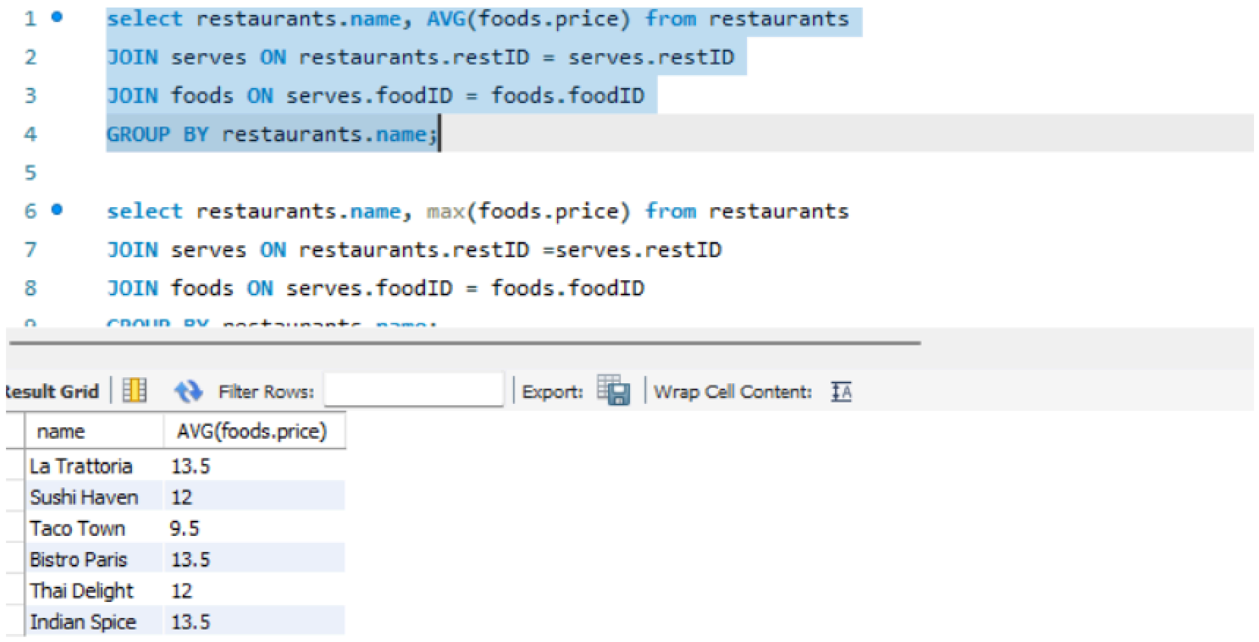
Problem 1: Find the average price of foods at each restaurant.
Query:
select restaurants.name, AVG(foods.price) from restaurants
JOIN serves ON restaurants.restID = serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY restaurants.name;
Explanation:
Query selects to display the restaurants' names, and the average of the food price from each restaurant by using the avg function. INNER JOINS will be used here since ID between tables match, specifically restID and foodID. The answers are grouped by restaurant name.



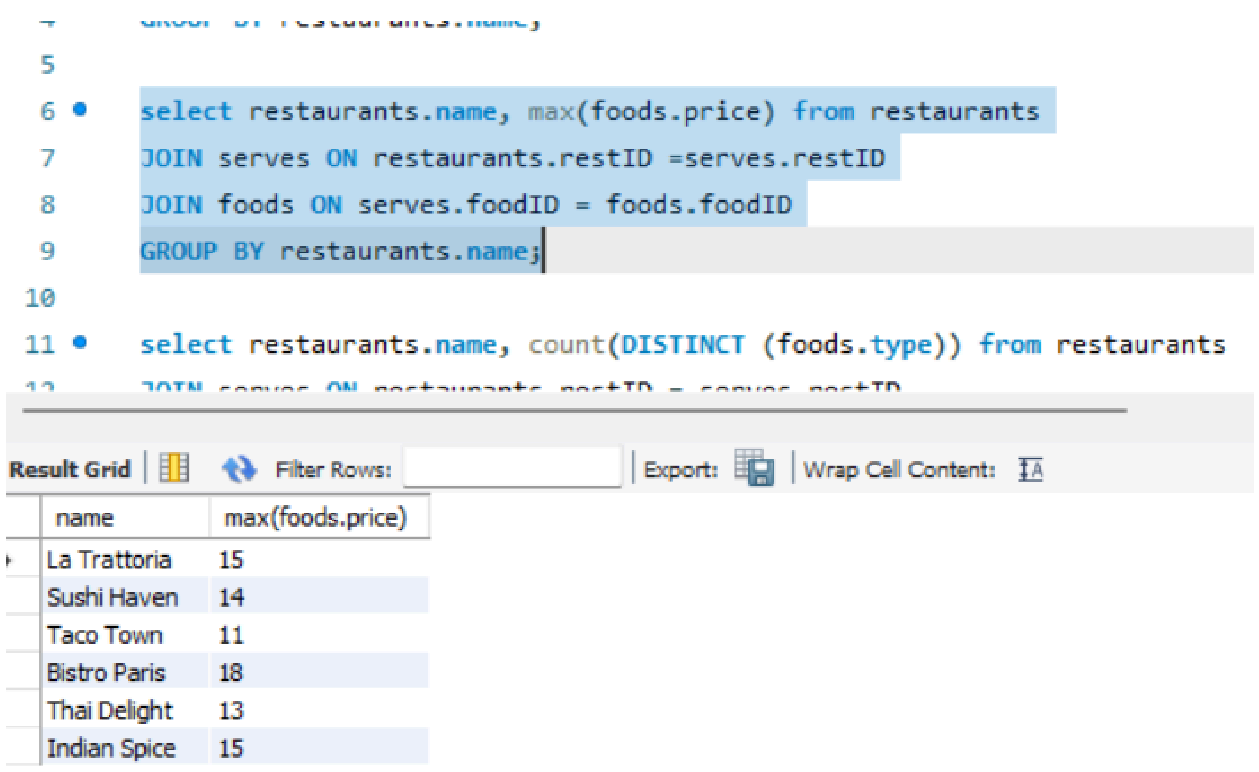
```
select restaurants.name, AVG(foods.price) from restaurants
JOIN serves ON restaurants.restID = serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY restaurants.name;

select restaurants.name, max(foods.price) from restaurants
JOIN serves ON restaurants.restID =serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY restaurants.name;
```

name	AVG(foods.price)
La Trattoria	13.5
Sushi Haven	12
Taco Town	9.5
Bistro Paris	13.5
Thai Delight	12
Indian Spice	13.5

Problem 2: Find the maximum food price at each restaurant
Query:

select restaurants.name, max(foods.price) from restaurants
JOIN serves ON restaurants.restID =serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY restaurants.name;

Explanation:
Query selects to display the restaurants' names, and the maximum food price from each restaurant by using the max function. INNER JOINS will be used here since ID between tables match, specifically restID and foodID. The answers are grouped by restaurant name.



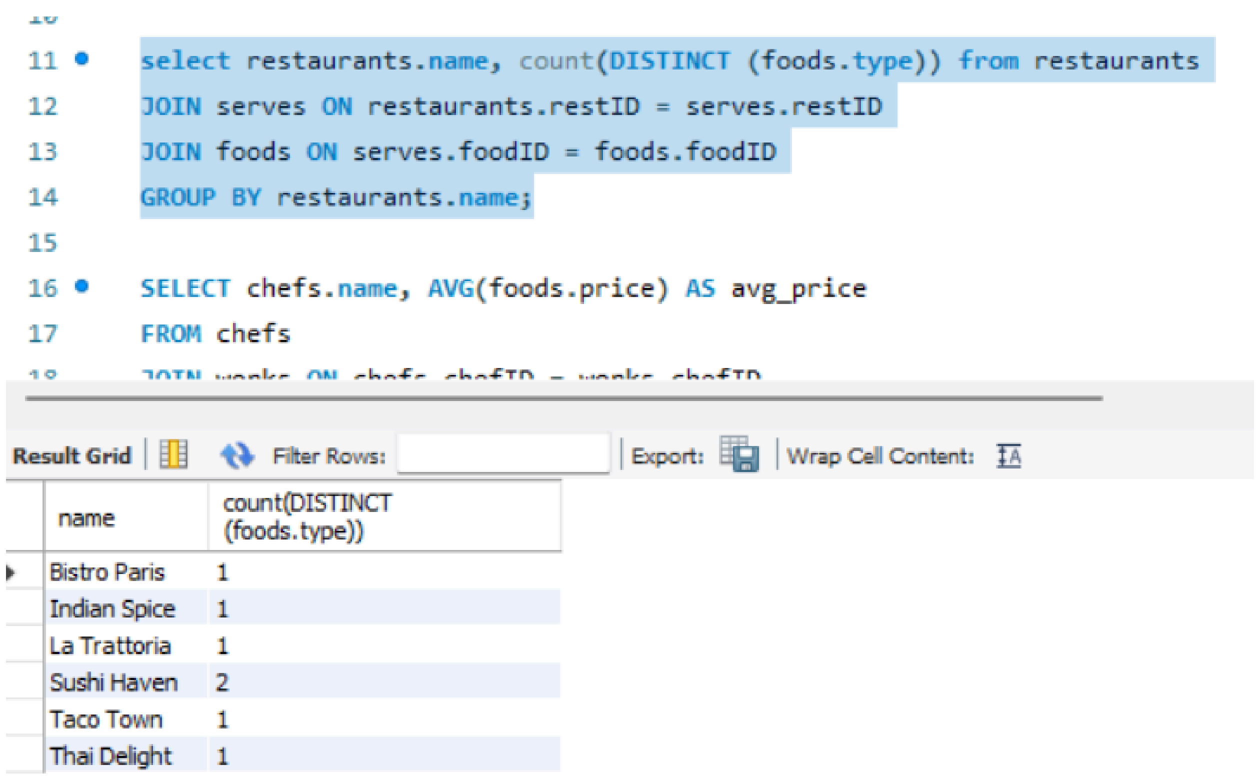
```
GROUP BY restaurants.name;

select restaurants.name, max(foods.price) from restaurants
JOIN serves ON restaurants.restID =serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY restaurants.name;

select restaurants.name, count(DISTINCT (foods.type)) from restaurants
JOIN serves ON restaurants.restID = serves.restID
```

name	max(foods.price)
La Trattoria	15
Sushi Haven	14
Taco Town	11
Bistro Paris	18
Thai Delight	13
Indian Spice	15

Problem 3: FInd the number of unique food types at each restaurant
Query:

select restaurants.name, count(DISTINCT (foods.type)) from restaurants
JOIN serves ON restaurants.restID = serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY restaurants.name;

Explanation:
Query selects to display the restaurants' names, and the number of unique foods from each restaurant by using the count function. The DISTINCT keyword is used to find the unique types from the list. INNER JOINS will be used here since ID between tables match, specifically restID and foodID. The answers are grouped by restaurant name.



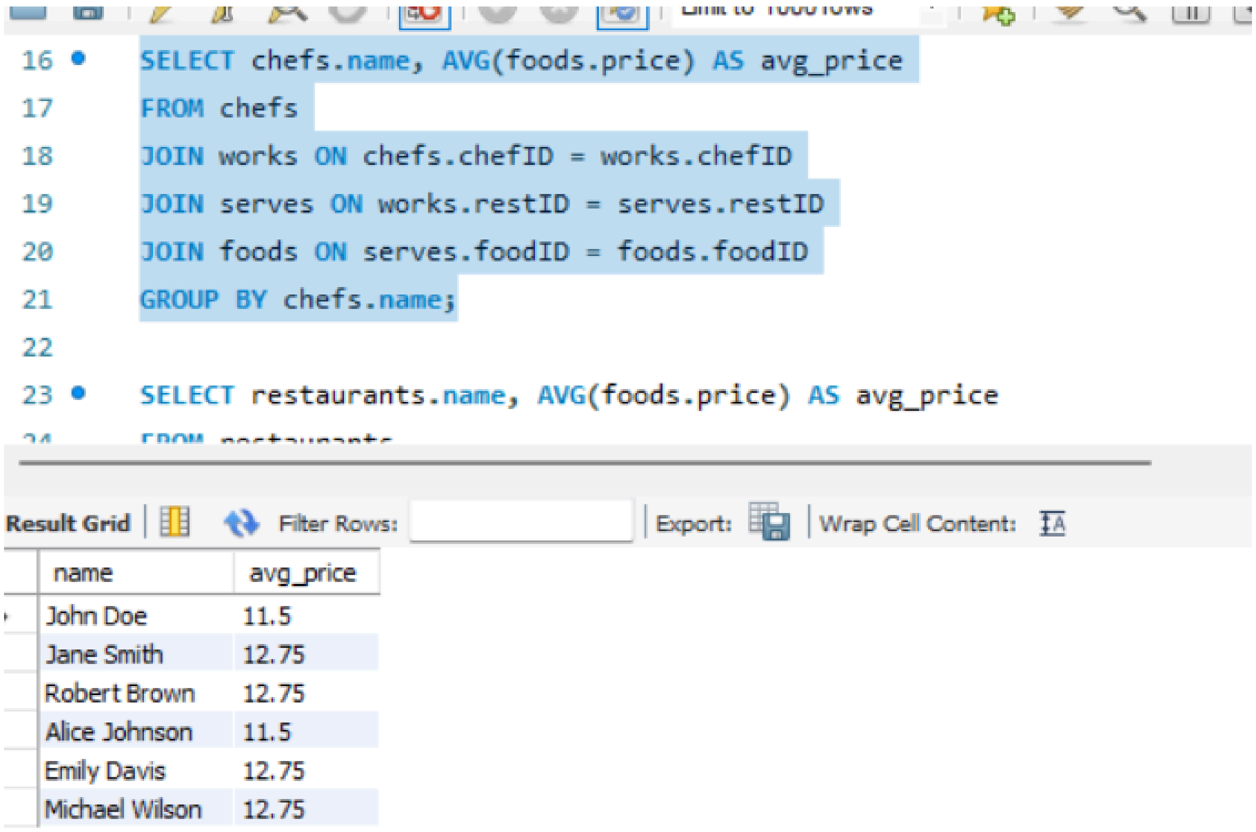
```
select restaurants.name, count(DISTINCT (foods.type)) from restaurants
JOIN serves ON restaurants.restID = serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY restaurants.name;

SELECT chefs.name, AVG(foods.price) AS avg_price
FROM chefs
```

name	count(DISTINCT (foods.type))
Bistro Paris	1
Indian Spice	1
La Trattoria	1
Sushi Haven	2
Taco Town	1
Thai Delight	1

Problem 4: Find the average price of foods served by each chef
Query:

SELECT chefs.name, AVG(foods.price) AS avg_price
FROM chefs
JOIN works ON chefs.chefID = works.chefID
JOIN serves ON works.restID = serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY chefs.name;

Explanation:
Query selects to display the chefs' names, and the average food price of the foods they serve. In this case the average food price is displayed as avg_price for readability. INNER JOINS will be used here since ID between tables match, specifically restID, chefID, and foodID. The answers are grouped by the chefs' names.



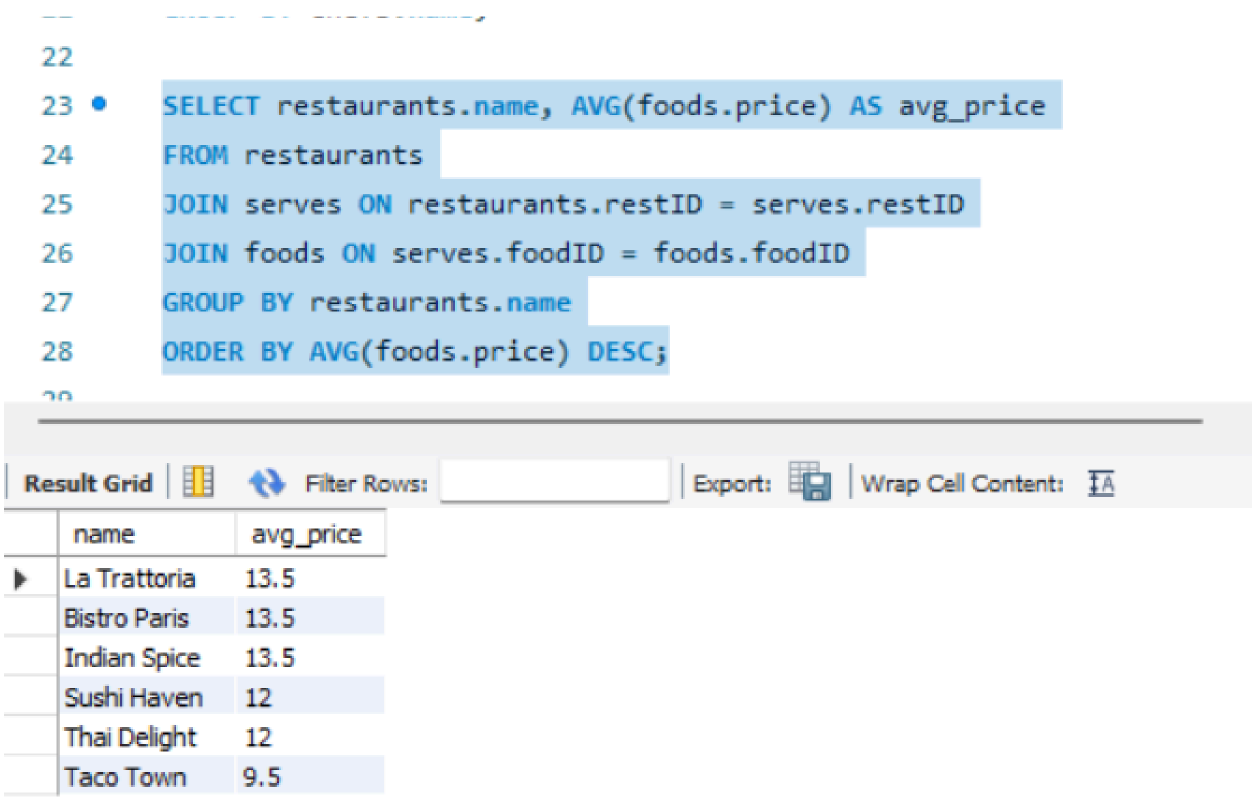
```
SELECT chefs.name, AVG(foods.price) AS avg_price
FROM chefs
JOIN works ON chefs.chefID = works.chefID
JOIN serves ON works.restID = serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY chefs.name;

SELECT restaurants.name, AVG(foods.price) AS avg_price
```

name	avg_price
John Doe	11.5
Jane Smith	12.75
Robert Brown	12.75
Alice Johnson	11.5
Emily Davis	12.75
Michael Wilson	12.75

Problem 5: Find the restaurant with the highest average food price.
Query:

SELECT restaurants.name, AVG(foods.price) AS avg_price
FROM restaurants
JOIN serves ON restaurants.restID = serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY restaurants.name
ORDER BY AVG(foods.price) DESC;

Explanation:
Query selects to display the restaurants' names, and the average food price for each restaurant, displayed as avg_price for readability. INNER JOINS will be used here since ID between tables match, specifically restID and foodID. The answers are grouped by the restaurants' names and are ordered by average food price and the DESC keyword is used to put them from highest to lowest values.



```
SELECT restaurants.name, AVG(foods.price) AS avg_price
FROM restaurants
JOIN serves ON restaurants.restID = serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY restaurants.name
ORDER BY AVG(foods.price) DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	avg_price
La Trattoria	13.5
Bistro Paris	13.5
Indian Spice	13.5
Sushi Haven	12
Thai Delight	12
Taco Town	9.5

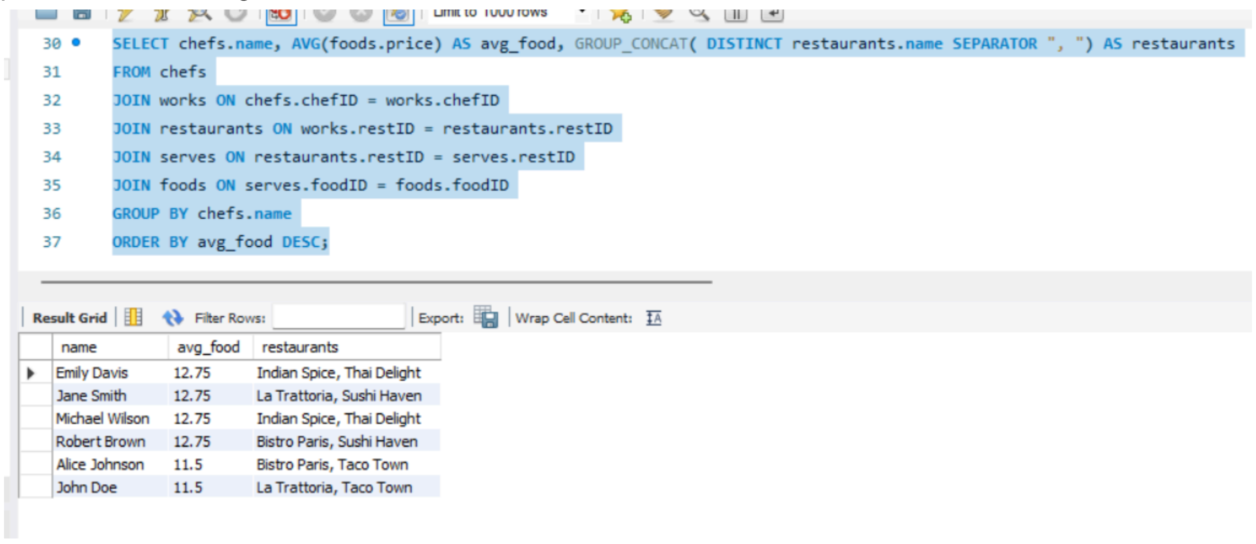
Problem 6: Extra credit. Determine which chef has the highest average price of the foods served at the restaurants where they work. Include the chef's name, the average food price, and the names of the restaurants where the chef works. Sort the results by the average food price in descending order.

Query:

```
SELECT chefs.name, AVG(foods.price) AS avg_food, GROUP_CONCAT( DISTINCT
restaurants.name SEPARATOR ", ") AS restaurants
FROM chefs
JOIN works ON chefs.chefID = works.chefID
JOIN restaurants ON works.restID = restaurants.restID
JOIN serves ON restaurants.restID = serves.restID
```

JOIN foods ON serves.foodID = foods.foodID
GROUP BY chefs.name
ORDER BY avg_food DESC;

Explanation:
The query selects to display chefs' names, the average food price of each restaurant, and the distinct restaurants are grouped here, since if they were not made distinct then they would be listed out multiple times in the same row of the output table. GROUP_CONCAT is used to allow the addition of a separator and to allow the querying of multiple rows for the resulting names of the restaurants, unlike normal CONCAT. The tables are joined using INNER JOINs on restID, chefID, and foodID. The resultant table is grouped by chefs' names and ordered by the food price in descending order.



```
SELECT chefs.name, AVG(foods.price) AS avg_food, GROUP_CONCAT( DISTINCT restaurants.name SEPARATOR ", ") AS restaurants
FROM chefs
JOIN works ON chefs.chefID = works.chefID
JOIN restaurants ON works.restID = restaurants.restID
JOIN serves ON restaurants.restID = serves.restID
JOIN foods ON serves.foodID = foods.foodID
GROUP BY chefs.name
ORDER BY avg_food DESC;
```

name	avg_food	restaurants
Emily Davis	12.75	Indian Spice, Thai Delight
Jane Smith	12.75	La Trattoria, Sushi Haven
Michael Wilson	12.75	Indian Spice, Thai Delight
Robert Brown	12.75	Bistro Paris, Sushi Haven
Alice Johnson	11.5	Bistro Paris, Taco Town
John Doe	11.5	La Trattoria, Taco Town