DBMS Assignment 2

Sarah Groark

Sep 22, 2024

Part 2:

1) Average price of foods at each restaurant

The query used to solve this problem utilizes a cross join to select rows that correspond to foods being served at each restaurant. Then the system calculates the average price of the foods being served at each restaurant, rounding to the hundredths place.

```
select restaurants.name, round(avg(foods.price),2) as average_price
from serves, restaurants, foods
where serves.foodID = foods.foodID and serves.restID = restaurants.restID
group by restaurants.name
order by average_price desc;
```

The output of the query is as follows:

name	average_price
La Trattoria	13.50
Bistro Paris	13.50
Indian Spice	13.50
Sushi Haven	12.00
Thai Delight	12.00
Taco Town	9.50

2) Maximum food price at each restaurant

This query uses inner join on to join the serves table with matching rows from the restaurants table (using restID) and the foods table (using foodID). The query calculates the maximum price of all the foods served at each restaurant and sorts the values in descending order, grouped by restaurant name.

```
select restaurants.name, round(max(foods.price),2) as max_food_price
from serves
inner join foods
        on serves.foodID = foods.foodID
inner join restaurants
        on serves.restID = restaurants.restID
group by restaurants.name
order by max_food_price desc;
```

Output:

name	max_food_price
Bistro Paris	18
La Trattoria	15
Indian Spice	15
Sushi Haven	14
Thai Delight	13
Taco Town	11

3) Count of different food types served at each restaurant

This query calculates the distinct number of food types served at each restaurant. Utilizing inner join, the query joins the serves table to both the restaurants and foods tables. The corresponding matching rows that are returned are based on the restID and foodID conditions. The query uses the distinct keyword to calculate the unique food types served at each restaurant. The restaurants served food types are grouped by the restaurant and sorted in ascending order.

```
select restaurants.name, count(distinct foods.type) as food_type_count
from serves
inner join foods using(foodID)
inner join restaurants using (restID)
group by restaurants.name
order by food_type_count;
```

Output:

name	food_type_count
Bistro Paris	1
Indian Spice	1
La Trattoria	1
Taco Town	1
Thai Delight	1
Sushi Haven	2

4) Average price of foods served by each chef

This query utilizes inner join to join the works table with three other tables (foods, chefs, and serves). The returning elements are those that match based on the following conditions: foodID, chefID, and restID. As a result, the query will calculate the price of foods served by each chef by examining which restaurant they work at and which foods are served by those restaurants. The resulting output shows individual chefs grouped by their name in addition to the average price of food they serve.

```
select chefs.name, avg(foods.price) as food_sold_avg
from works
inner join serves using (restID)
inner join foods using (foodID)
inner join chefs using (chefID)
group by chefs.name
order by food_sold_avg desc;
```

Output:

name	food_sold_avg
Jane Smith	12.75
Robert Brown	12.75
Emily Davis	12.75
Michael Wilson	12.75
John Doe	11.50
Alice Johnson	11.50

5) Find the restaurant with the highest average food price

This query uses a cross join to pull together all possible rows from serves, restaurants, and foods. The cross join uses foodID and restID to return matching rows and then calculates the average price of foods served at those restaurants. Finally, after being sorted in descending order, the first row from the results is outputted, which displays the restaurant name(s) that have the highest average food price and the corresponding price.

```
select restaurants.name, round(avg(foods.price),2) as average_price
from serves
inner join restaurants using (restID)
inner join foods using (foodID)
group by restaurants.name
having (average_price) >= all
   (select avg(foods.price)
   from serves
   inner join restaurants using (restID)
         inner join foods using (foodID)
   group by restaurants.name);
```

Output: In this case, there are three restaurants with the same average food price, which happens to be the highest in the data, so all three restaurants are displayed.

name	average_price
La Trattoria	13.50
Bistro Paris	13.50
Indian Spice	13.50

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.

This query uses inner join to join the works table with the following tables: serves, chefs, foods, restaurants. The resulting rows have matching values in columns of restID (from serves), chefID, foodID, and restID (from restaurants). The query then calculates the average price of food served by each chef. The "having" clause allows for sorting through the results to find the highest averages. The output will return the name(s) of the chefs with the highest average(s) found in the data, as well as the list of restaurants they've worked at.

```
select chefs.name, round(avg(foods.price),2) as avg_food_price, group_concat(distinct
restaurants.name) as restaurants_worked_at
from works
inner join serves using (restID)
inner join chefs using (chefID)
inner join foods using (foodID)
inner join restaurants using (restID)
group by chefs.name
having (avg_food_price) >= all
        (select avg(foods.price)
   from works
   inner join serves using (restID)
        inner join chefs using (chefID)
        inner join foods using (foodID)
        inner join restaurants using (restID)
   group by chefs.name);
```

Output:

name	avg_food_price	restaurants_worked_at
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