Complex SQL

BY,

BHARATH VENKATESH SRINIVASAN

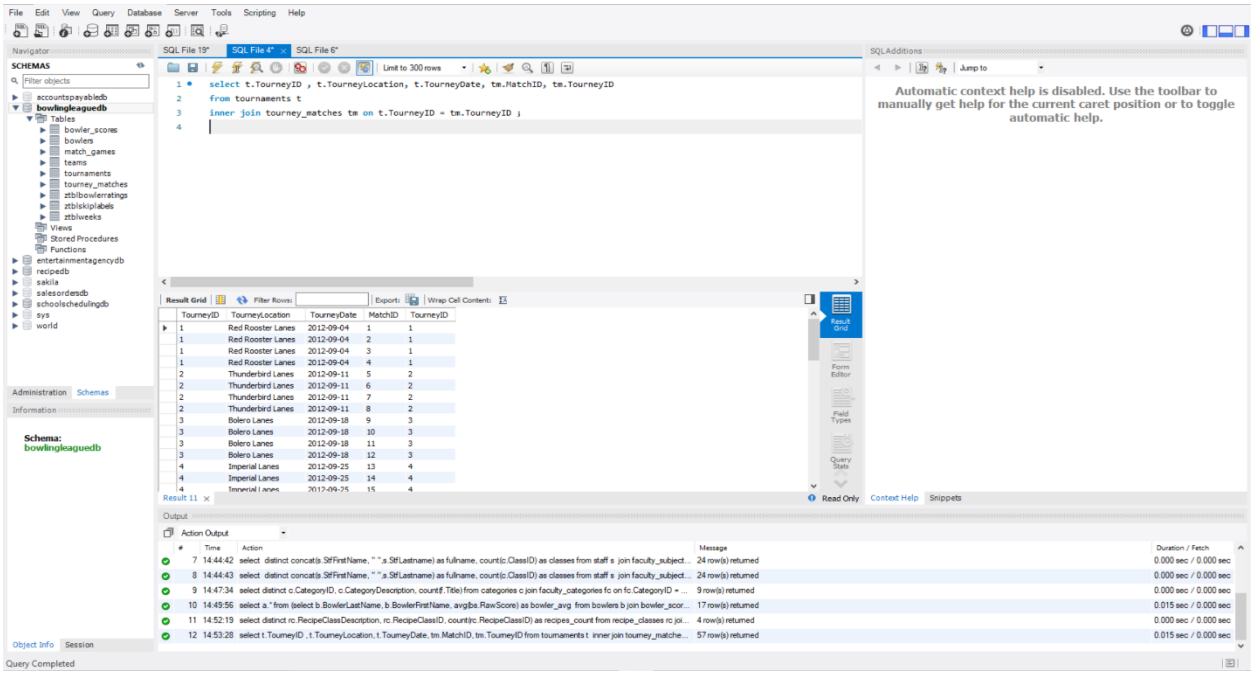
Question: Display all tournaments and any matches that have been played.

Translation: Select tournaments and list of matches that are played in tournament.

Cleanup: Select tourneyID, tourney location, tourneydate, matchID, tourneyID from tournaments and match ID from tournaments table by using inner join between tournaments and tourney_matches.

Query:

```
select t.TourneyID , t.TourneyLocation, t.TourneyDate, tm.MatchID, tm.TourneyID

from tournaments t

inner join tourney_matches tm on t.TourneyID = tm.TourneyID ;
```



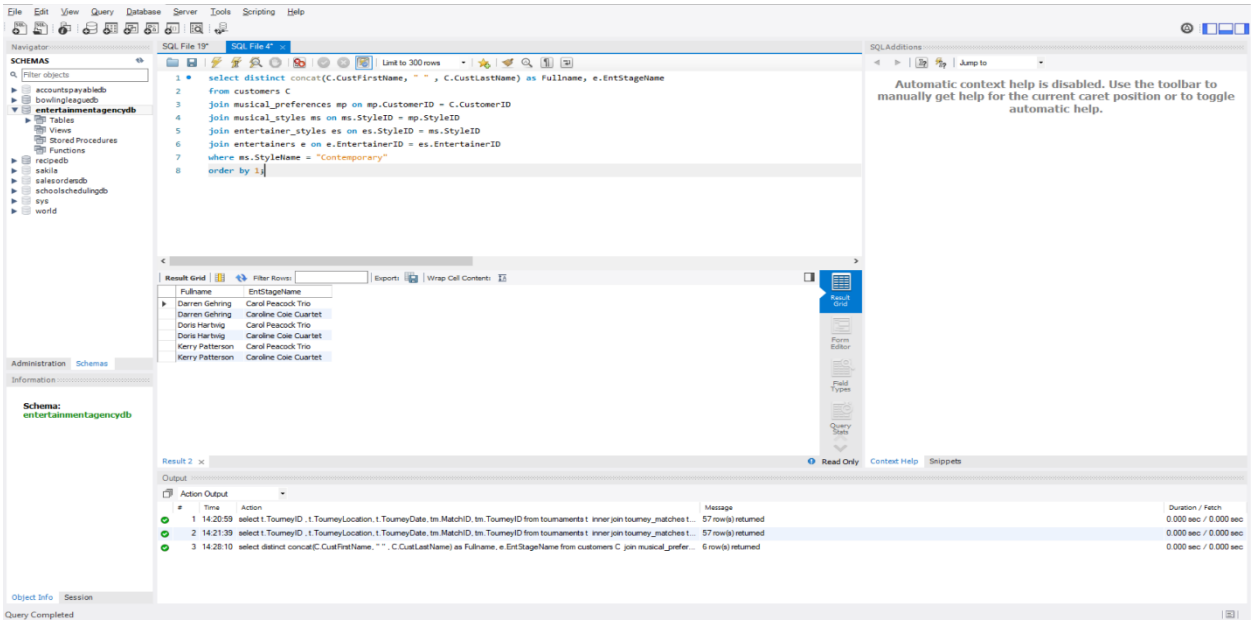
Question: Produce a list of customers who like contemporary music together with a list of entertainers who play contemporary music.

Translation: Select customers who like contemporary music and the entertainers who play contemporary music.

Cleanup: Select customers whose stylename is contemporary music by inner joining musical_preferences to customers table. Musical_styles is then joined with musical_preferences and entertainer_styles is joined with musical_styles/ Entertainers is joined with musical_styles to get EntStagename. At the end, where statements is used to filter contemporary music.

Query:

```
select distinct concat(C.CustFirstName, " " , C.CustLastName) as Fullname, e.EntStageName

from customers C

join musical_preferences mp on mp.CustomerID = C.CustomerID

join musical_styles ms on ms.StyleID = mp.StyleID

join entertainer_styles es on es.StyleID = ms.StyleID

join entertainers e on e.EntertainerID = es.EntertainerID

where ms.StyleName = "Contemporary"

order by 1;
```



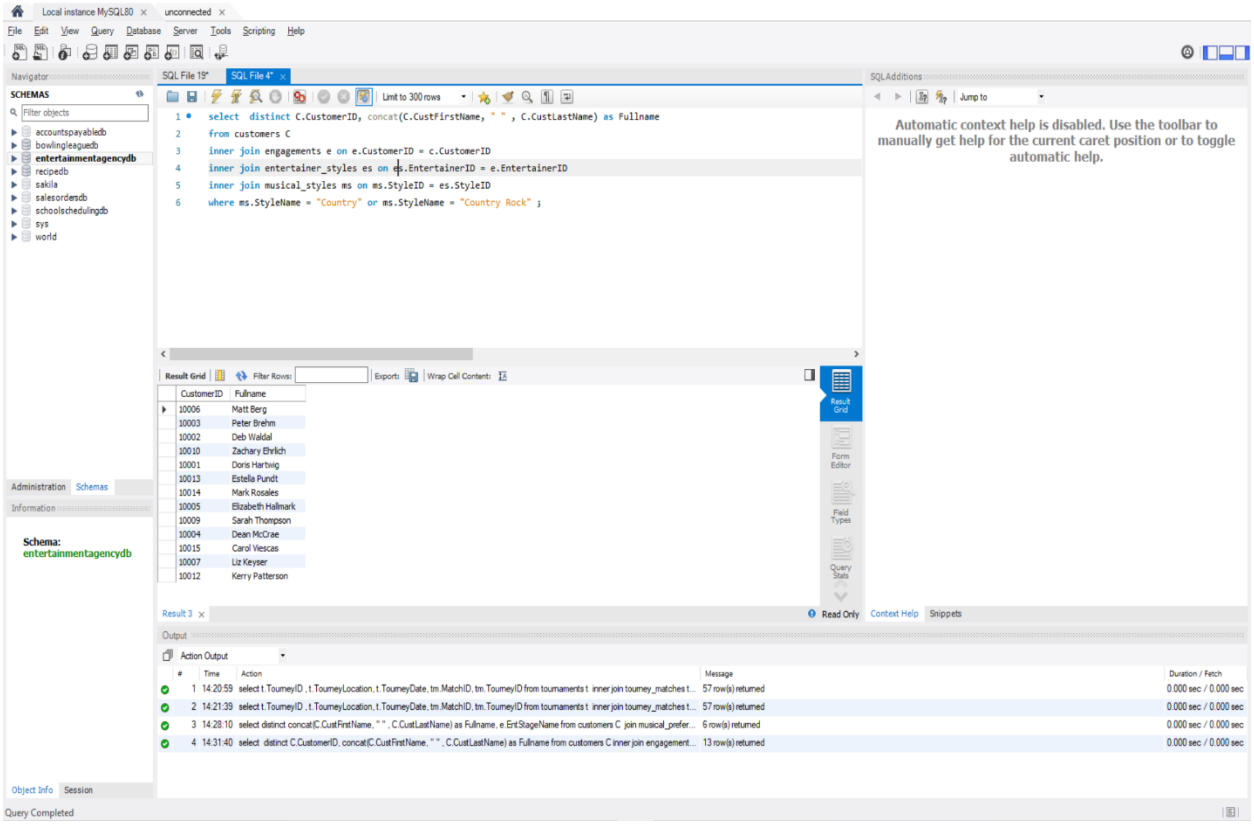
Question: List customers who have booked entertainers who play country or country rock.

Translation: Select Customer details who booked entertainers that play conutry or contryrock music.

Cleanup: Select CustomerID, Customerfirstname, Custlastname from customers table and inner join it to engagements table to join it to entertainer_styles table. Then, join musical_styles to entertainer_styles to identify entertainers who played country or country rock.

Query:

```
select  distinct C.CustomerID, concat(C.CustFirstName, " " , C.CustLastName) as Fullname

from customers C

inner join engagements e on e.CustomerID = c.CustomerID

inner join entertainer_styles es on es.EntertainerID = e.EntertainerID

inner join musical_styles ms on ms.StyleID = es.StyleID

where ms.StyleName = "Country" or ms.StyleName = "Country Rock" ;
```



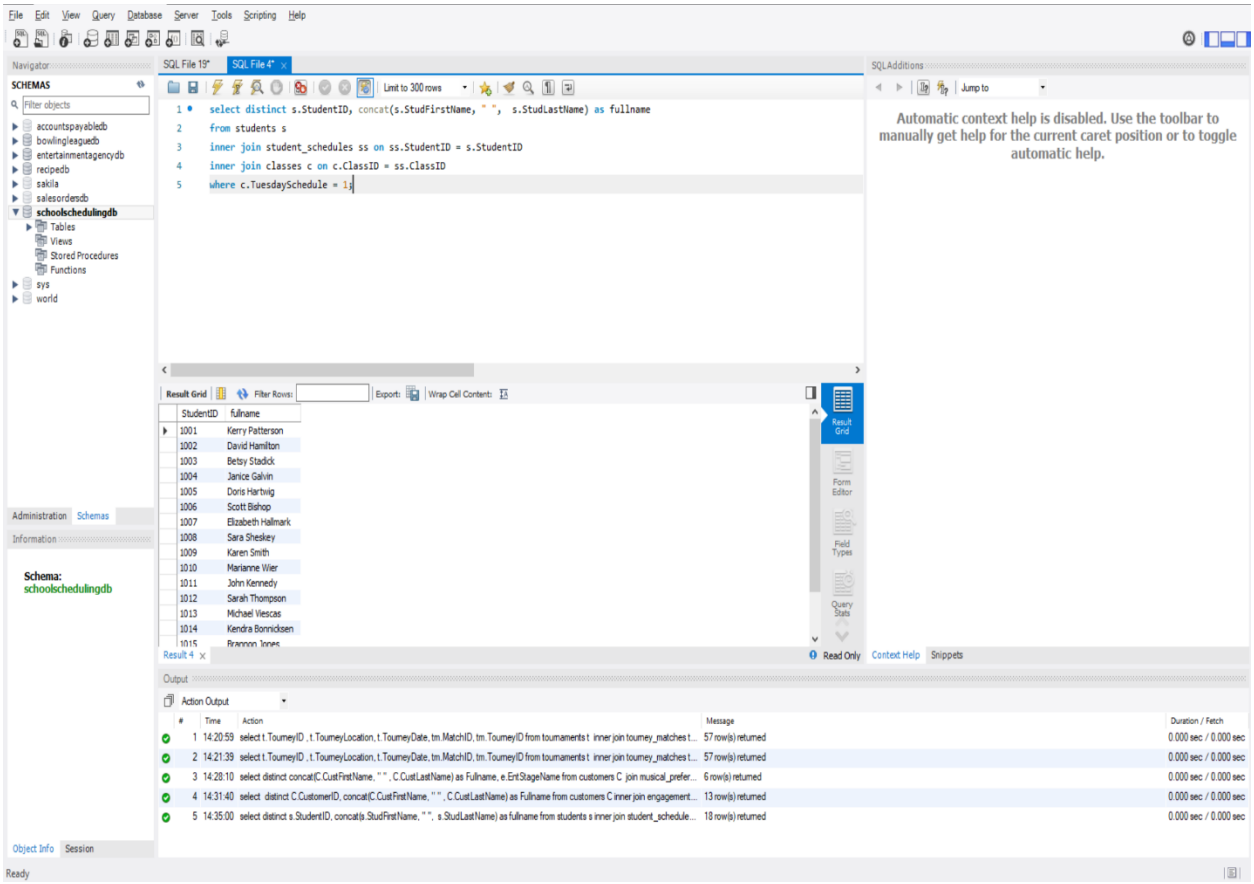
Question: Display students enrolled in a class on Tuesday.

Translation: Select all the students who are enrolled in a class on Tuesday.

Cleanup: Select StudentID, StudFirstName, StudLastName from students table by joining student_schedules and students table and also joining student_schedules and classes table where Tuesday schedule = 1.

Query:

```
select distinct s.StudentID, concat(s.StudFirstName, " ",  s.StudLastName) as fullname

from students s

inner join student_schedules ss on ss.StudentID = s.StudentID

inner join classes c on c.ClassID = ss.ClassID

where c.TuesdaySchedule = 1;
```



Question: List the ingredients that are used in some recipe where the measurement amount in the recipe is not the default measurement amount

Translation: Select all ingredients which are used in some recipes where measurement amount in the recipe is not equal to default measurement amount

Cleanup: Select IngredientName, RecipeTitle, measurementDescription, Preparation from ingredients by joining measurements on ingredients table. Next, we need to join recipe_ingredients to ingredients table and join recipes to recipe_ingredients and filter by using a condition of measurementamountID in measurements is not equal to measurementAmountID in recipe_ingredients.

Query:

SELECT distinct IngredientName,r.RecipeTitle, m.MeasurementDescription, r.Preparation FROM ingredients i

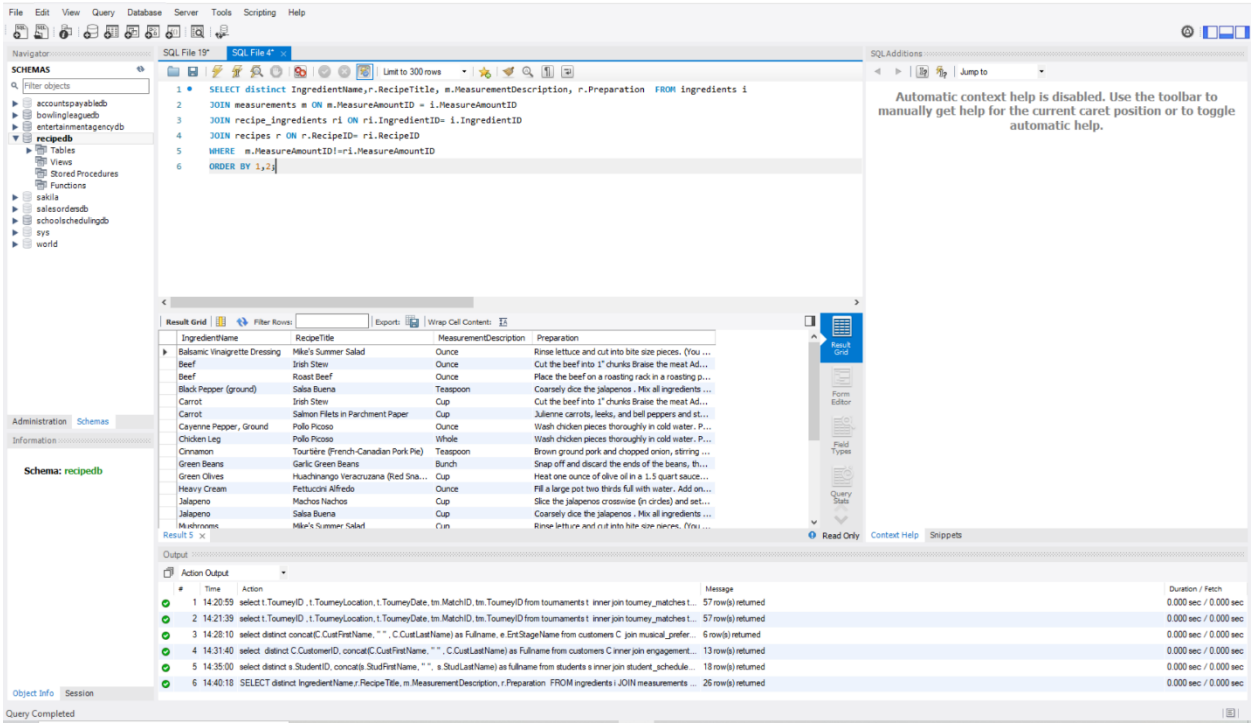
JOIN measurements m ON m.MeasureAmountID = i.MeasureAmountID

JOIN recipe_ingredients ri ON ri.IngredientID= i.IngredientID

JOIN recipes r ON r.RecipeID= ri.RecipeID

WHERE m.MeasureAmountID!=ri.MeasureAmountID

ORDER BY 1,2;



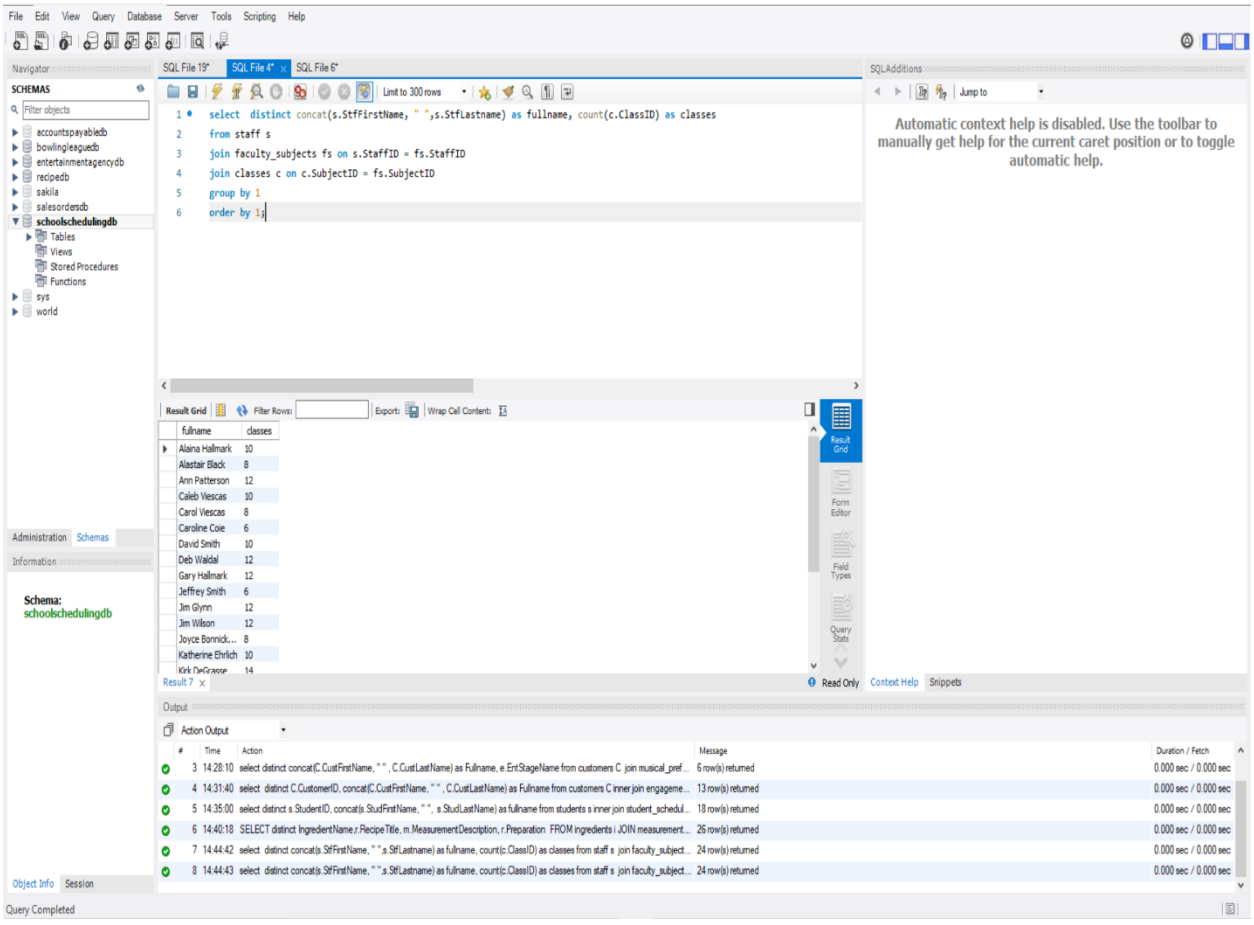
Question: List each staff member and the count of classes each is scheduled to teach.

Translation: Select all staff members and the count of classes they are scheduled to teach.

Cleanup: Select StaffID, StfFirstName, StfLastName from staff table and join faculty_subjects with staff and then join classes with faculty_subjects to get count of classID.

Query:

```
select  distinct concat(s.StfFirstName, " ",s.StfLastname) as fullname, count(c.ClassID) as classes

from staff s

join faculty_subjects fs on s.StaffID = fs.StaffID

join classes c on c.SubjectID = fs.SubjectID

group by 1

order by 1;
```



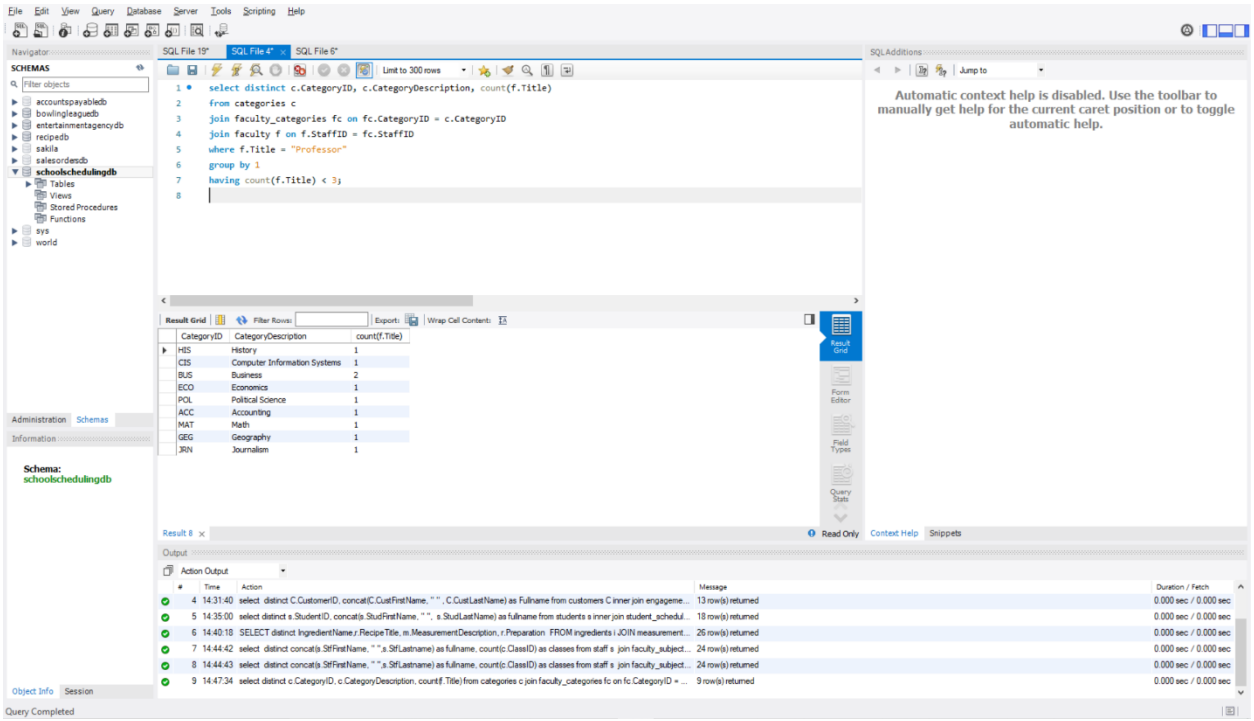
Question: Show me the subject categories that have fewer than three full professors teaching that subject.

Translation: Select the subject categories which contain fewer than three full professors teaching that subject.

Cleanup: Select CategoryID, CategoryDescription from categories table by joining it with faculty_categories table and further joining faculty_categories with faculty table and use a filter of Professor title and use having statement to filter count function.

Query:

```
select distinct c.CategoryID, c.CategoryDescription, count(f.Title)

from categories c

join faculty_categories fc on fc.CategoryID = c.CategoryID

join faculty f on f.StaffID = fc.StaffID

where f.Title = "Professor"

group by 1

having count(f.Title) < 3;
```



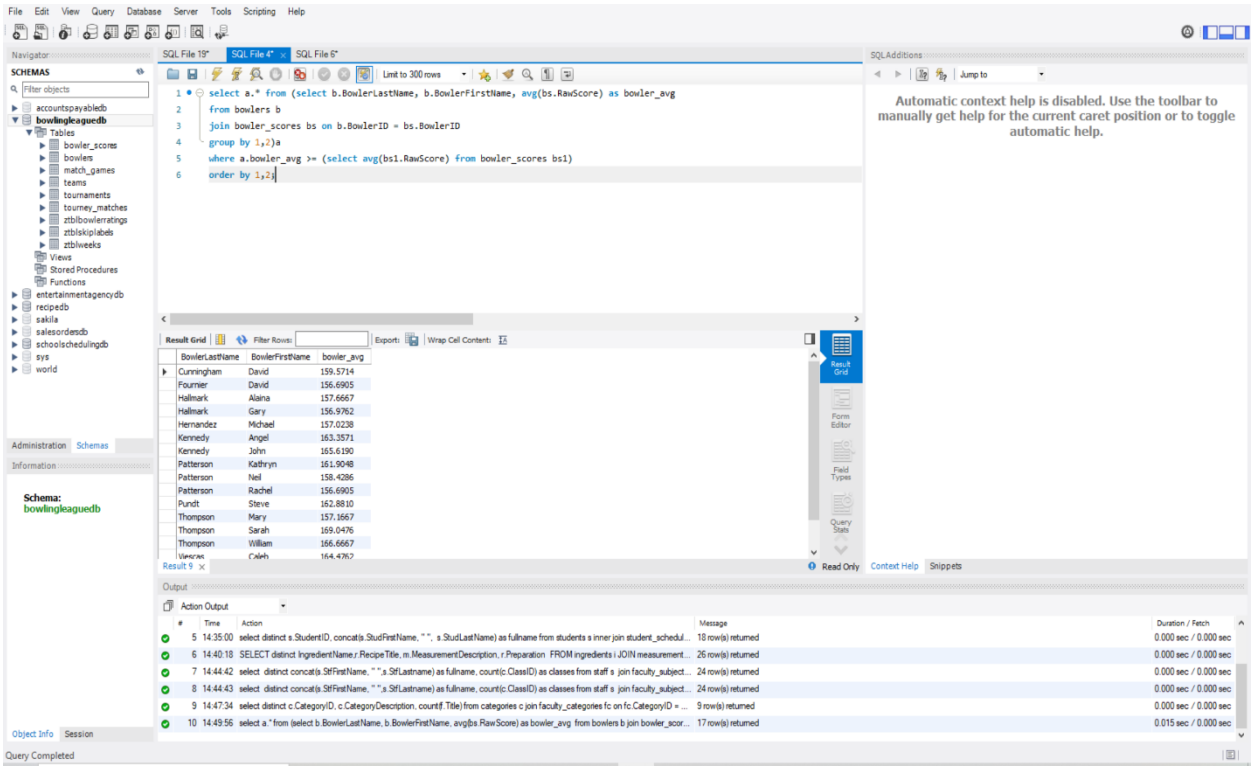
Question: List the last name and first name of every bowler whose average raw score is greater than or equal to the overall average score.

Translation: Select bowlers whose average raw score is greater than or equal to overall average score.

Cleanup: Initially write a subquery of Select BowlerFirstName, BowlerLastName and bowlervise average score from bowlers table and join it with bowler_scores table. Then write a query which will compare the result of subquery to overall average of bowler score.

Query:

```
select a.* from (select b.BowlerLastName, b.BowlerFirstName, avg(bs.RawScore) as bowler_avg

from bowlers b

join bowler_scores bs on b.BowlerID = bs.BowlerID

group by 1,2)a

where a.bowler_avg >= (select avg(bs1.RawScore) from bowler_scores bs1)

order by 1,2;
```



Question: For what class of recipe do I have two or more recipes.

Translation: Select classes of recipe which are having two or more recipes.

Cleanup: Select RecipeclassDescription, recipeclassID and count of recipeclassID from recipe_classes and join with recipes and filter the count fucntion using having after group by, by using count >=2.

Query:

```
select distinct rc.RecipeClassDescription, rc.RecipeClassID, count(rc.RecipeClassID) as recipes_count
from recipe_classes rc
join recipes r on rc.RecipeClassID = r.RecipeClassID
group by 2
having recipes_count >=2;
```