

## 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 ;
```

The screenshot displays the SQL Developer interface. The 'Navigator' pane on the left shows the 'bowlingleaguepdb' schema with tables like 'bowling\_scores', 'bowlers', 'match\_games', 'teams', 'tournaments', 'tourney\_matches', 'ttblbowlerstats', 'ttblskiplabels', and 'ttblweeks'. The 'SQL' pane in the center contains the following query:

```
1 select t.TourneyID , t.TourneyLocation, t.TourneyDate, tm.MatchID, tm.TourneyID
2 from tournaments t
3 inner join tourney_matches tm on t.TourneyID = tm.TourneyID ;
4
```

The 'Result Grid' pane shows the query results with 15 rows. The columns are TourneyID, TourneyLocation, TourneyDate, MatchID, and TourneyID. The results are as follows:

TourneyID	TourneyLocation	TourneyDate	MatchID	TourneyID
1	Red Rooster Lanes	2012-09-04	1	1
1	Red Rooster Lanes	2012-09-04	2	1
1	Red Rooster Lanes	2012-09-04	3	1
1	Red Rooster Lanes	2012-09-04	4	1
2	Thunderbird Lanes	2012-09-11	5	2
2	Thunderbird Lanes	2012-09-11	6	2
2	Thunderbird Lanes	2012-09-11	7	2
2	Thunderbird Lanes	2012-09-11	8	2
3	Bolero Lanes	2012-09-18	9	3
3	Bolero Lanes	2012-09-18	10	3
3	Bolero Lanes	2012-09-18	11	3
3	Bolero Lanes	2012-09-18	12	3
4	Imperial Lanes	2012-09-25	13	4
4	Imperial Lanes	2012-09-25	14	4
4	Imperial Lanes	2012-09-25	15	4

The 'Output' pane at the bottom shows the execution log with 12 messages, including the query execution and the number of rows returned (57 rows).

**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;
```

The screenshot displays the SQL Enterprise Manager interface. The central pane shows a SQL query in a file named 'SQL File 19'. The query is as follows:

```
1 select distinct concat(C.CustFirstName, " ", C.CustLastName) as Fullname, e.EntStageName
2 from customers C
3 join musical_preferences mp on mp.CustomerID = C.CustomerID
4 join musical_styles ms on ms.StyleID = mp.StyleID
5 join entertainer_styles es on es.StyleID = ms.StyleID
6 join entertainers e on e.EntertainerID = es.EntertainerID
7 where ms.StyleName = "Contemporary"
8 order by 1;
```

The left pane shows the 'Schemas' tree with 'entertainmentagencydb' selected. The right pane shows a message: 'Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.'

Below the query, the 'Result Grid' shows the results of the query. The columns are 'Fullname' and 'EntStageName'. The results are:

Fullname	EntStageName
Darren Gehring	Carol Peacock Trio
Darren Gehring	Caroline Cole Quartet
Doris Harwig	Carol Peacock Trio
Doris Harwig	Caroline Cole Quartet
Kerry Patterson	Carol Peacock Trio
Kerry Patterson	Caroline Cole Quartet

At the bottom, the 'Output' pane shows the execution log. The log indicates that the query was executed successfully and returned 6 rows.

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

Translation: Select Customer details who booked entertainers that play country or country rock 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" ;
```

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
1 select distinct C.CustomerID, concat(C.CustFirstName, " ", C.CustLastName) as Fullname
2 from customers C
3 inner join engagements e on e.CustomerID = C.CustomerID
4 inner join entertainer_styles es on es.EntertainerID = e.EntertainerID
5 inner join musical_styles ms on ms.StyleID = es.StyleID
6 where ms.StyleName = "Country" or ms.StyleName = "Country Rock" ;
```

The Results window displays the following data:

CustomerID	Fullname
10006	Matt Berg
10003	Peter Brehm
10002	Deb Waldd
10010	Zachary Eivlich
10001	Dore Hartwig
10013	Estella Pundt
10014	Mark Rooles
10005	Elizabeth Hallmark
10009	Sarah Thompson
10004	Dean McCrae
10015	Carol Vescas
10007	Liz Keyser
10012	Kerry Patterson

The Output window shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	14:20:59	select t.TourneyID, t.TourneyLocation, t.TourneyDate, t.MatchID, t.TourneyID from tournaments t inner join touney_matches t	57 row(s) returned	0.000 sec / 0.000 sec
2	14:21:39	select t.TourneyID, t.TourneyLocation, t.TourneyDate, t.MatchID, t.TourneyID from tournaments t inner join touney_matches t	57 row(s) returned	0.000 sec / 0.000 sec
3	14:28:10	select distinct concat(C.CustFirstName, " ", C.CustLastName) as Fullname, e.EntStageName from customers C join musical_prefer...	6 row(s) returned	0.000 sec / 0.000 sec
4	14:31:40	select distinct C.CustomerID, concat(C.CustFirstName, " ", C.CustLastName) as Fullname from customers C inner join engagement...	13 row(s) returned	0.000 sec / 0.000 sec

## 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;
```

The screenshot displays the SQL Server Enterprise Manager interface. The left pane shows the 'SCHEMAS' tree with 'schoolschedulingdb' selected. The central pane shows a SQL query in 'SQL File 4':

```
1 select distinct s.StudentID, concat(s.StudFirstName, " ", s.StudLastName) as fullname
2 from students s
3 inner join student_schedules ss on ss.StudentID = s.StudentID
4 inner join classes c on c.ClassID = ss.ClassID
5 where c.TuesdaySchedule = 1;
```

The right pane shows a message: 'Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.'

Below the query, the 'Result Grid' shows the results of the query. The columns are 'StudentID' and 'fullname'. The results are:

StudentID	fullname
1001	Kerry Patterson
1002	David Hamilton
1003	Betsy Stadick
1004	Janice Galvin
1005	Doris Hartwig
1006	Scott Bishop
1007	Elizabeth Hallmark
1008	Sara Shesley
1009	Karen Smith
1010	Marianne Wier
1011	John Kennedy
1012	Sarah Thompson
1013	Michael Vescas
1014	Kendra Bonnickson
1015	Brennon Innes

The bottom pane shows the 'Output' window with 'Action Output' selected. It displays a list of actions and their durations:

#	Time	Action	Message	Duration / Fetch
1	14:20:59	select t.TourneyID, t.TourneyLocation, t.TourneyDate, tm.MatchID, tm.TourneyID from tournaments t inner join tourney_matches t...	57 row(s) returned	0.000 sec / 0.000 sec
2	14:21:39	select t.TourneyID, t.TourneyLocation, t.TourneyDate, tm.MatchID, tm.TourneyID from tournaments t inner join tourney_matches t...	57 row(s) returned	0.000 sec / 0.000 sec
3	14:28:10	select distinct concat(C.CustFirstName, " ", C.CustLastName) as Fullname, e.EntStageName from customers C join musical_prefer...	6 row(s) returned	0.000 sec / 0.000 sec
4	14:31:40	select distinct C.CustomerID, concat(C.CustFirstName, " ", C.CustLastName) as Fullname from customers C inner join engagement...	13 row(s) returned	0.000 sec / 0.000 sec
5	14:35:00	select distinct s.StudentID, concat(s.StudFirstName, " ", s.StudLastName) as fullname from students s inner join student_sch...	10 row(s) returned	0.000 sec / 0.000 sec

**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;
```

The screenshot displays the SQL Server Enterprise Manager interface. The left pane shows the 'Schemas' tree with 'recipepdb' selected. The central pane shows a SQL query in 'SQL File 1' with the following text:

```
1 SELECT distinct IngredientName,r.RecipeTitle, m.MeasurementDescription, r.Preparation FROM ingredients i
2 JOIN measurements m ON m.MeasureAmountID = i.MeasureAmountID
3 JOIN recipe_ingredients ri ON ri.IngredientID= i.IngredientID
4 JOIN recipes r ON r.RecipeID= ri.RecipeID
5 WHERE m.MeasureAmountID!=ri.MeasureAmountID
6 ORDER BY 1,2;
```

The right pane shows the 'SQL Additions' tab with a message: 'Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.'

Below the query, the 'Result Grid' shows the results of the query. The columns are 'IngredientName', 'RecipeTitle', 'MeasurementDescription', and 'Preparation'. The results are as follows:

IngredientName	RecipeTitle	MeasurementDescription	Preparation
Balsamic Vinaigrette Dressing	Mike's Summer Salad	Ounce	Rinse lettuce and cut into bite size pieces. (You ...
Beef	Irish Stew	Ounce	Cut the beef into 1" chunks Brase the meat Ad...
Beef	Roast Beef	Ounce	Place the beef on a roasting rack in a roasting p...
Black Pepper (ground)	Salita Buena	Teaspoon	Coarsely dice the jalapenos. Mix all ingredients ...
Carrot	Irish Stew	Cup	Cut the beef into 1" chunks Brase the meat Ad...
Carrot	Salmon Filets in Parchment Paper	Cup	Julienne carrots, leeks, and bell peppers and st...
Cayenne Pepper, Ground	Pollo Picoso	Ounce	Wash chicken pieces thoroughly in cold water. P...
Chicken Leg	Pollo Picoso	Whole	Wash chicken pieces thoroughly in cold water. P...
Cinnamon	Tourtiere (French-Canadian Pork Pie)	Teaspoon	Brown ground pork and chooped onion, stirring ...
Green Beans	Garlic Green Beans	Bunch	Snap off and discard the ends of the beans, th...
Green Olives	Huachinango Veracruzana (Red Sna...	Cup	Heat one ounce of olive oil in a 1.5 quart sauce...
Heavy Cream	Pettucchi Alfredo	Ounce	Fill a large pot two thirds full with water. Add on...
Jalapeno	Machos Nachos	Cup	Slice the jalapenos crosswise (in circles) and set...
Jalapeno	Salita Buena	Cup	Coarsely dice the jalapenos. Mix all ingredients ...
Mushrooms	Mike's Summer Salad	Cin	Rinse lettuce and cut into bite size pieces. (You ...

The bottom pane shows the 'Output' tab with a table of 'Action Output' and 'Message' columns. The messages indicate the execution progress of the query, such as '57 row(s) returned' and '25 row(s) returned'.

**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;
```

The screenshot displays the SQL Server Enterprise Manager interface. The left pane shows the 'Schemas' tree with 'schoolschedulngdb' selected. The central pane shows the following SQL query:

```
1 select distinct concat(s.StfFirstName, " ",s.StfLastname) as fullname, count(c.ClassID) as classes
2 from staff s
3 join faculty_subjects fs on s.StaffID = fs.StaffID
4 join classes c on c.SubjectID = fs.SubjectID
5 group by 1
6 order by 1;
```

The right pane shows a message: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

Below the query, the 'Result Grid' shows the following data:

fullname	classes
Alana Hallmark	10
Alastair Black	8
Ann Patterson	12
Caleb Vescas	10
Carol Vescas	8
Caroline Coe	6
David Smith	10
Deb Waldal	12
Gary Hallmark	12
Jeffrey Smith	6
Jim Glynn	12
Jim Wilson	12
Joyce Bonnick...	8
Katherine Ehrlich	10
Kirk DeGraep	14

The bottom pane shows the 'Output' window with the following message:

```
Result 7 x
# Time Action Message Duration / Fetch
3 14:28:10 select distinct concat(C.CustFirstName, "", C.CustLastName) as FullName, e.EngStageName from customers C join musical_pref... 6 row(s) returned 0.000 sec / 0.000 sec
4 14:31:40 select distinct C.CustomerID, concat(C.CustFirstName, "", C.CustLastName) as FullName from customers C inner join engageme... 13 row(s) returned 0.000 sec / 0.000 sec
5 14:35:00 select distinct s.StudentID, concat(s.StudFirstName, "", s.StudLastName) as FullName from students s inner join student_schedul... 10 row(s) returned 0.000 sec / 0.000 sec
6 14:40:18 SELECT distinct IngredientName, r.RecipeTitle, m.MeasurementDescription, r.Preparation FROM ingredients i JOIN measurement... 25 row(s) returned 0.000 sec / 0.000 sec
7 14:44:42 select distinct concat(s.StfFirstName, " ", s.StfLastName) as fullname, count(c.ClassID) as classes from staff s join faculty_subj... 24 row(s) returned 0.000 sec / 0.000 sec
8 14:44:43 select distinct concat(s.StfFirstName, " ", s.StfLastName) as fullname, count(c.ClassID) as classes from staff s join faculty_subj... 24 row(s) returned 0.000 sec / 0.000 sec
```

**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;
```

The screenshot displays the SQL Enterprise Manager interface. The central pane shows a query in the SQL File 6 editor:

```
1 select distinct c.CategoryID, c.CategoryDescription, count(f.Title)
2 from categories c
3 join faculty_categories fc on fc.CategoryID = c.CategoryID
4 join faculty f on f.StaffID = fc.StaffID
5 where f.Title = "Professor"
6 group by 1
7 having count(f.Title) < 3;
```

The Results pane below the query shows the output of the query:

CategoryID	CategoryDescription	count(f.Title)
HIS	History	1
CIS	Computer Information Systems	1
BUS	Business	2
ECO	Economics	1
POL	Political Science	1
ACC	Accounting	1
MAT	Math	1
GEO	Geography	1
JRN	Journalism	1

The bottom pane shows the Action Output, which includes a list of actions and their results:

Action	Message	Duration / Fetch
4 14:31:40 select distinct C.CustomerID, concat(C.CustFirstName, " ", C.CustLastName) as FullName from customers C inner join engagem...	13 row(s) returned	0.000 sec / 0.000 sec
5 14:35:00 select distinct s.StudentID, concat(s.StuFirstName, " ", s.StuLastName) as FullName from students s inner join student_sch...	10 row(s) returned	0.000 sec / 0.000 sec
6 14:40:18 SELECT distinct IngredientName / RecipeTitle, m.MeasurementDescription, i.Preparation FROM ingredients i JOIN measurement...	25 row(s) returned	0.000 sec / 0.000 sec
7 14:44:42 select distinct concat(s.StuFirstName, " ", s.StuLastName) as FullName, count(c.ClassID) as classes from staff s join faculty_sch...	24 row(s) returned	0.000 sec / 0.000 sec
8 14:44:43 select distinct concat(s.StuFirstName, " ", s.StuLastName) as FullName, count(c.ClassID) as classes from staff s join faculty_sch...	24 row(s) returned	0.000 sec / 0.000 sec
9 14:47:34 select distinct c.CategoryID, c.CategoryDescription, count(f.Title) from categories c join faculty_categories fc on fc.CategoryID = ...	9 row(s) returned	0.000 sec / 0.000 sec

**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 bowlerwise 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;
```

The screenshot displays the SQL Enterprise Manager interface. The central pane shows a query in the SQL File 4\* editor:

```
1 select a.* from (select b.BowlerLastName, b.BowlerFirstName, avg(bs.RawScore) as bowler_avg
2 from bowlers b
3 join bowler_scores bs on b.BowlerID = bs.BowlerID
4 group by 1,2)a
5 where a.bowler_avg >= (select avg(bs1.RawScore) from bowler_scores bs1)
6 order by 1,2;
```

The Results pane below the query shows the output of the query, displaying columns: BowlerLastName, BowlerFirstName, and bowler\_avg. The results are as follows:

BowlerLastName	BowlerFirstName	bowler_avg
Cunningham	David	159.5714
Fournier	David	156.6905
Hallmark	Alana	157.6667
Hallmark	Gary	156.5762
Hernandez	Michael	157.0238
Kennedy	Angel	163.3571
Kennedy	John	155.6190
Patterson	Kathryn	161.9048
Patterson	Ned	158.4286
Patterson	Rachel	156.6905
Pundt	Steve	162.8810
Thompson	Mary	157.1667
Thompson	Sarah	169.0476
Thompson	William	166.6667
Wierwille	Caleb	164.4762

The Output pane at the bottom shows the execution log, indicating that the query was executed successfully and returned 17 rows.



**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 function 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;
```

The screenshot displays the SQL Server Enterprise Manager interface. The left pane shows the 'Schemas' tree with 'recipe\_classes' selected. The central pane shows the following SQL query:

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

The 'Result Grid' shows the following data:

RecipeClassDescription	RecipeClassID	recipes_count
Main course	1	7
Vegetable	2	2
Hors d'oeuvres	5	2
Dessert	6	2

The bottom pane shows the 'Output' window with the following messages:

```
6 14:40:18 SELECT distinct IngredientName, RecipeTitle, m.MeasurementDescription, r.Preparation FROM ingredients i JOIN measurement... 25 row(s) returned
7 14:44:42 select distinct concat(s.SFirstName, " " + s.SFLastName) as fullname, count(c.ClassID) as classes from staff s join faculty_subject... 24 row(s) returned
8 14:44:43 select distinct concat(s.SFirstName, " " + s.SFLastName) as fullname, count(c.ClassID) as classes from staff s join faculty_subject... 24 row(s) returned
9 14:47:34 select distinct c.CategoryID, c.CategoryDescription, count(f.Title) from categories c join faculty_categories f on f.CategoryID = ... 9 row(s) returned
10 14:49:56 select a.* from (select b.BowlerLastName, b.BowlerFirstName, avg(b.RanScore) as bowler_avg from bowlers b join bowler_scor... 17 row(s) returned
11 14:52:19 select distinct rc.RecipeClassDescription, rc.RecipeClassID, count(rc.RecipeClassID) as recipes_count from recipe_classes rc jo... 4 row(s) returned
```