DT228/2, DT282/2 Databases I
Joins and Functions

Objectives

The objectives of this lab are to:

- To use group functions to form the output

- To become more familiar with the SELECT statement

- To become familiar with using joins in the SELECT statement to traverse a physical database

- To use functions to format the output and within the where and group clause.

Contents

1. Create your data structures

Download DT228-DT282-Week6Lab.sql from Webcourses and run it in SQL Developer.

This will create the tables for the data model shown in the ERD below. It is the model created in the labs in week 4 to store details on sporting events scheduled in a sports competition, the competitors competing in these competitions and the positions these competitors finished in that competition with some small changes to the model and a little bit of extra data.

So if Usain Bolt, email address UB@jam.com, is competitor number 1, finishes 1st in the 100m Mens final on 16th Aug 2016, event 1, finishing in position 1, in the sport Athletics, sport 1, there will be a row in the SPORT table with a sportCode of 1 and a sportName Athletics; a row in the EVENT table event ID 1, eventName 100m Mens Final, eventDate 16 AUG 2016, a sportCode of 1 plus the amount of prize money available in prizeFund; a row in the COMPETITOR table with compID 1, compName Usain Bolt, compEmail UB@jam.com; and then a row in the EVENTRESULTS table with the eventNo 1, compID 1 and position 1.

At the end of the script are a number of select statements which will show you the data included in each table.
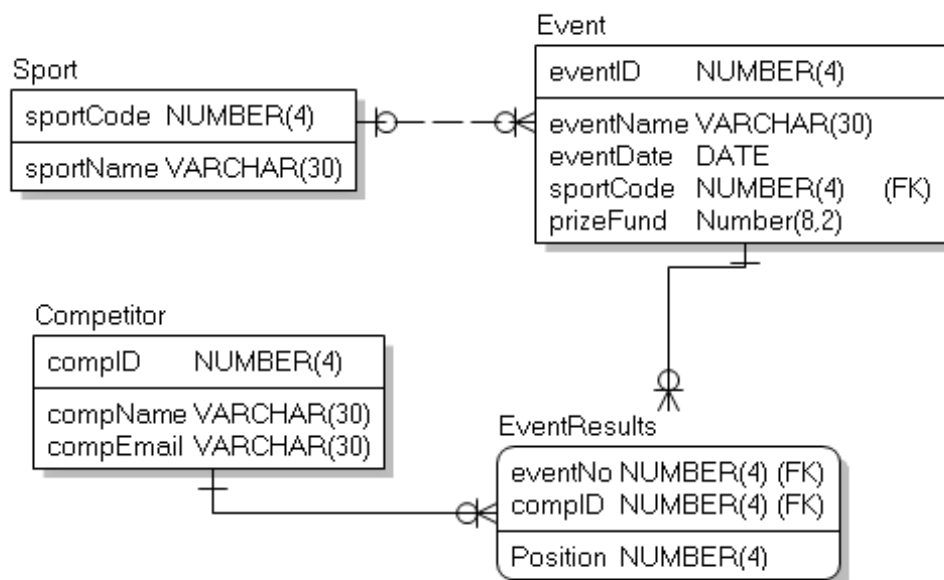
1. Grouping

1. Write the SQL to output the number of events in each sport. Include the sportcode in the output.
    What table has information about events and a sportcode column?
    Use COUNT as the function. Think about what is being counted (events so what column is unique for each event?). Think about what forms the groups – you want to get the number of events in each sport so which column do you use in the group by clause?
    You should have 3 rows in your output.

2. Change the SQL for 2 to include the sportname in the output
    How would be get the sportname – think about what table sportname is in?
    What has to change in the group by clause? Think about an error we talked about in class?

3. Write the SQL to output the average finishing position for each competitor. Include the competitor ID in the output.
    What is the function? What table has information about finishing position and a competitorID column?
    Think about what forms the groups?
    You should have 11 rows in your output.

4. Change the SQL for 3 to include the competitor name in the output.
    Think about what table competitor name is in.
    What has to change in the group by clause?

5. Change the SQL for 4 so that it will only include competitors with a lowercase letter i in their name.
    How to restrict what goes into a group by? What clause do you need? You need to use a function.
    You should get 9 rows in your output

6. Change the SQL for 5 to only show the output for competitors with an average finishing position higher than 2.
    What additional clause is needed?
    You should have 4 rows in your output

7. Write the SQL to find the average prizefund for each sport.
    Could you amend the SQL you wrote for 1?

8. Change the SQL you wrote for 7 so that instead of null values for average prizefund, 0 is output.
    What function could you use to handle nulls?

9. Change the SQL you wrote for 8 so that the average prize fund is output preceded by the euro symbol and as a 6 digits with 2 decimal places.

    Use the to_char function, U is the local currency symbol.

10. Write the SQL to output the average prizefund for events that happened on a Wednesday or on a Sunday.
    How would you figure out the day of the week a date happened on? To_char(datex,'D')
    Monday is considered day 1.

11. Change the SQL you wrote for 10 so that it will only include events for sportcode 2. Remember the rules of precedence.

12. Change the SQL you wrote for 11 so that it will only output averages that have no value. How do you check if a value is null?