r/learnprogramming

Posts New? READ ME FIRST! Posting guidelines FAQ

20

Posted by u/SwiimGood 4 years ago

[SQL] Referring to values in the outer SELECT within the inner SELECT

Solved

I'm doing this tutorial - http://sqlzoo.net/wiki/SELECT_within_SELECT_Tutorial - and question 6 is

> Find the largest country (by area) in each continent, show the continent, the name and the area:

I've got the answer correct, but I really don't understand what I'm doing or why. The answer is

```
SELECT continent, name, area
FROM world x
Where area >= ALL
(SELECT area FROM world y
WHERE y.continent=x.continent
AND area>0)
```

I'm trying to figure out what exactly the x and y are doing and how I use them.

Would someone be willing to break this down line by line and tell me what's going on?

93% Upvoted

16 Comments Share Save

COMMUNITY DETAILS

r/learnprogramming

948k Members | 2.0k Online

A subreddit for all questions related to programming in any language.

JOIN

CREATE POST

This thread is archived
New comments cannot be posted and votes cannot be cast

SORT BY **BEST**

[deleted] 2 points · 4 years ago

Both the x and the y are aliases for the world table.

It's like

SELECT continent, name, area
FROM world [referred to from now on as x]
Where area >= ALL
(SELECT area FROM world [referred to from now on as y]
WHERE y.continent=x.continent
AND area>0)

It's just a kind of workaround to select twice from the same table.

You can do this using the "AS" keyword (*SELECT continent, name, area FROM world AS x*) which makes things clearer, but it's optional.

Share **Report** **Save**

SwiimGood 1 point · 4 years ago

Okay, so why do I need **WHERE y.continent=x.continent**?

And why do I need to give them aliases at all?

Sorry, I'm incredibly new at this.

Share **Report** **Save**

WELCOME TO LEARNPROGRAMMING!

New? READ ME FIRST!

Posting guidelines

Frequently asked questions

Subreddit rules

Message the moderators

R/LEARNPROGRAMMING RULES

lightcloud5 2 points · 4 years ago

In general, aliases are used for two things:

- For convenience. Aliases are usually shorter and easier to type. (e.g. you can shorten the table `users` to the alias `u`)
- Required when the same table participates multiple times in one query. This is common in several queries; notably, hierarchical queries. For instance, aliases are necessary to identify the correct table in the following query:

query:

```
SELECT
    e.employeeName,
    manager.employeeName as '
name of boss/manager'
FROM employees e
JOIN employees manager
    ON e.managerEmployeeId =
manager.employeeId
```

Share **Report** **Save**

1. No unprofessional/derogatory speech
2. No spam or tasteless self-promotion
3. No off-topic posts
4. Do not ask exact duplicates of FAQ questions
5. Do not delete posts
6. No app/website review requests/app showcases
7. No rewards
8. No indirect links
9. Do not promote illegal or unethical practices
10. No complete solutions

ASKING DEBUGGING QUESTIONS

When posting a question about code, you must include the following:

1. A **concise but descriptive title**.
2. A **good description** of the problem.
3. A **minimal, easily runnable**, and **well-formatted** program that illustrates your problem.
4. The output you expected, and what you got instead. If you got an error, include the **full** error message.

See debugging question guidelines for more info.

ASKING CONCEPTUAL QUESTIONS

Many conceptual questions have already been asked and answered. Read our FAQ page and search old posts before asking your question. If your question is similar to one in the FAQ, explain how it's different.

See conceptual questions guidelines for more info.

SwiimGood 1 point · 4 years ago

> Required when the same table participates multiple time in one query.

So that's why I have to give **world** an alias, then? Because I am referencing it twice? I think I understand.

What about the line **WHERE y.continent=x.continent**?

I'm trying to figure out why this part is necessary and what exactly it does.

I assume **y.continent** is referring to the continent column in (world)y, and it will be compared with the same in x, but I'm not sure why they are necessary or what exactly they're doing in this particular query.

I understand the last line (**AND area>0**) is basically **IS NOT NULL**, correct?

Share **Report** **Save**

[deleted] 2 points · 4 years ago · *edited 4 ye*

> The real difficulty in this question is not finding the country with the largest population, or the country with the largest population for a given continent. Those are both simple.
>
> It's finding the single largest country in *each* continent.
>
> So you need to have a way to compare each country with all the other countries in the same continent.
>
> So it's like "find the country whose area is greater than or equal to all the countries in its continent". In order to do that

OTHER GUIDELINES AND LINKS

1. Frequently asked questions
2. Asking homework questions
3. Answering questions
4. Learning resources
5. Other communities

MODERATORS

u/trpcicm

u/zzyzzyxx

u/desrtfx

u/michael0x2a

u/BotBust

u/lp-bot botbustproof

u/AutoModerator

VIEW ALL MODERATORS

you can't just do a single select, because you need two lists of all countries.

EDIT:

To look at it another way, if you do this (taking out the aliases):

```
SELECT continent, name,
 population FROM world
  WHERE population >= A
LL
    (SELECT population
FROM world
    WHERE continent = c
ontinent
      AND population>0)
```

you get only one result, China. Because `WHERE continent = continent` becomes meaningless. You get the single biggest country in *any* continent.

But if we "pretend" that we have two different tables, SQL matches them up and we get one result per continent.

Share **Report** **Save**

SwiimGood 1 point · 4 years ago

Okay, so **WHERE y.continent=x.continent** basically allows it to compare the area of countries where the continents match and ignore the others, correct? It basically goes down the list, continent by continent, and pops out the ones with the largest area?

Share **Report** **Save**

BACK TO TOP

[deleted] 2 points · 4 years ago

Exactly. It's like it builds two tables of "countries in Africa" and compares all of them with all of them to make sure weve found the biggest, repeat for other continents.

It's probably quite inefficient/memory hungry.

Share **Report** **Save**

SwiimGood 1 point · 4 years ago

I understand now! Thank you!

Share **Report** **Save**

C0rinthian 1 point · 4 years ago

> It's probably quite inefficient/memory hungry.

Not necessarily true. It really depends on what columns are indexed. This is a pretty standard relational problem, which is exactly what relational databases are good at.

One trap people fall into with SQL is they approach it with a procedural (or OO) mindset. That's wrong, and leads to people making complex or inefficient queries because they're trying to "optimize". Or even worse, they

pull large datasets into the app and manipulate it there because they think they know better than the database engine.

Writing good SQL is all about understanding how to communicate what you want relationally. That's all a SQL query is: a declaration of what you want the database to spit out. You don't tell it HOW to give you the answer. It figures that out itself. (Which is pretty amazing when you think about it)

Share **Report** **Save**

[deleted] 1 point · 4 years ago

I'm thinking of it as inefficient because I'm seeing a list of 200 countries in the first select and 200 in the second so there will be 40,000 comparisons. But maybe that's not how it works.

I'm also comparing it with a loop in your programming language of choice which goes through a list of continents and uses sql to find the largest country.

Share Report Save

C0rinthian 1 point · 4 years ago

That's exactly the trap I said people fall into. Don't assume it's working the way you would implement it, because you are almost definitely wrong.

Databases use some advanced data structures and algorithms for

storage and retrieval. (B-Trees, for example, are used extensively) What they do under the hood is usually very complex, but also very efficient.

In this example, the only thing that makes this query potentially slow is that it's looking at all records. But that's unavoidable for the problem being solved, and it's most likely only doing it once. (the subselects would only be accessing the records they need to based on the WHERE condition)

Share Report Save

Continue this thread →

More posts from the learnprogramming community

2.1k
I'm a self-taught developer with nearly a decade of industry experience. I went from years spent trying to make game engines with no luck to having > $100k annual salary. Here's my advice.
Posted by u/TheAdventMaster 5 days ago
211 Comments Share Save

1.4k
A brief intro to git - for absolute beginners
Posted by u/imnos 4 days ago
68 Comments Share Save

942
Learn to code by watching other developers work.
Posted by u/mmmbc 6 days ago
66 Comments Share Save

817
Tutorial A detailed tutorial on scraping information from the Web and tweeting it programmatically using a bot!
Posted by u/schedutron 6 days ago
25 Comments Share Save

630
Master Basics of Data Science this Saturday- Live Streamed Intro to NumPy and Pandas Course
Posted by u/g_pal 2 days ago
40 Comments Share Save

476
Are there any people or places that review your code and suggest improvements?
Posted by u/Jagg3rs 1 day ago
73 Comments Share Save