Database Administrators

Why do wildcards in GROUP BY statements not work?

Asked 7 years, 2 months ago Active 6 years, 8 months ago Viewed 13k times

28

3

I am trying to make the following SQL statement work, but I get a syntax error:

```
SELECT A.*, COUNT(B.foo)
FROM TABLE1 A
LEFT JOIN TABLE2 B ON A.PKey = B.FKey
GROUP BY A.*
```

Here, A is a wide table with 40 columns and I would like to avoid listing each column name in the GROUP BY clause if possible. I have many such tables over which I have to run a similar query, so I will have to write a Stored Procedure. What's the best way to approach this?

I am using MS SQL Server 2008.

edited Jul 20 '12 at 13:38
Jon Seigel
15.8k 6 37 74

asked Jul 20 '12 at 13:12
Anonymous Maximus
419 2 7 15

2 Answers

31

`GROUP BY A.*` is not allowed in SQL.

You can bypass this by using a subquery where you group by, and then join:

```
SELECT A.*, COALESCE(B.cnt, 0) AS Count_B_Foo
FROM TABLE1 AS A
  LEFT JOIN
      ( SELECT FKey, COUNT(foo) AS cnt
        FROM TABLE2
        GROUP BY FKey
      ) AS B
    ON A.PKey = B.FKey ;
```

There is a feature in SQL-2003 standard to allow in the `SELECT` list, columns that are not in the `GROUP BY` list, as long as they are functionally dependent on them. If that feature had been implemented in SQL-Server, your query could have been written as:

```
SELECT A.*, COUNT(B.foo)
FROM TABLE1 A
LEFT JOIN TABLE2 B ON A.PKey = B.FKey
GROUP BY A.pk                          --- the Primary Key of table A
```

Unfortunately, this feature has not yet been implemented, not even in SQL-Server 2012 version - and not in any other DBMS as far as I know. Except for MySQL which has it but inadequately (inadequately as: the above query will work but the engine will do no checking for functional dependency and other ill-written queries will show wrong, semi-random results).

As @Mark Byers informed us in a comment, **PostgreSQL 9.1 added a new feature** designed for this purpose. It is more restrictive than MySQL's implementation.

edited Jan 7 '13 at 19:53 answered Jul 20 '12 at 13:19

Can you mention a couple of RDBMS's that implement that portion of the standard as written? I know, for example, that MySQL will allow you (given the proper settings) to include items not within the `GROUP BY` clause in the `SELECT` list, but it leaves it undefined as to which row that value will come from (so if the column or expression *isn't* functionally dependent upon the grouping expression, then it *could* come from any row within the group). – Adam Robinson Jul 20 '12 at 15:08

@Adam: No, I know no RDBMS that has it implemented. MySQL has it but inadequately, as your comment says. – ypercubeᵀᴹ Jul 20 '12 at 15:10

Gotcha. I was actually asking if there were, since I have experience with far fewer RDBMS's than I would imagine most of the people answering questions on this site would have ;) But that was my suspicion. – Adam Robinson Jul 20 '12 at 15:23

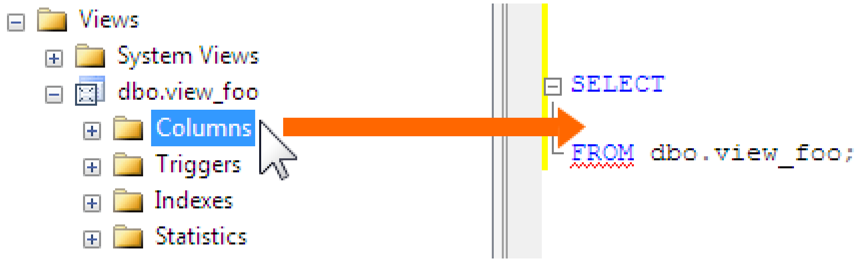
3 "and not in any other DBMS as far as I know." PostgreSQL 9.1 added a new feature designed for this purpose. It is more restrictive than MySQL's implementation. – Mark Byers Jul 21 '12 at 7:29

@MarkByers: thnx, I didn't know that. – ypercubeᵀᴹ Jul 21 '12 at 7:31

24

In addition to @ypercube's workaround, "typing" is never an excuse for using `SELECT *`. I've written about this here, and even with the workaround I think your `SELECT` list should still include the column names - even if there are a massive number like 40.

Long story short, you can avoid typing these big lists by clicking and dragging the Columns node for the object in Object Explorer onto your query window. The screen shot shows a view but the same thing can be done for a table.



```
SELECT

FROM dbo.view_foo;
```

But if you want to read about all the reasons *why* you should subject yourself to this huge level of effort of dragging an item a few inches, please read my post. :-)

edited Jul 20 '12 at 13:28

answered Jul 20 '12 at 13:22
Aaron Bertrand ♦
161k 19 323 528

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.

I certainly agree that `SELECT *` should not be used. I'm curious about the `GROUP BY` case though. @Aaron, are there efficiency issues with having 40 columns in the Group By list? – ypercubeᵀᴹ Jul 20 '12 at 13:33

1 @ypercube - As far as I've seen if you group by `A.PK, A.some, A.other, A.columns` it doesn't bother actually comparing `some, other, columns` this is just required by the syntax. – Martin Smith Jul 20 '12 at 14:09

1 @datagod sorry, no, any gaps could only be explained by the SSMS dev team. :-) – Aaron Bertrand ♦ Jul 20 '12 at 14:29

1 @Pacerier Sorry, I disagree completely, but maybe you could elaborate. – Aaron Bertrand ♦ Apr 9 '15 at 15:41