

Re: GroupBy Error

Source:

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.programming/2004-11/3726.html>

From: Joe Celko (jcelko212_at_earthlink.net)

Date: 11/16/04

Date: Tue, 16 Nov 2004 14:30:31 -0800

1) All of your data element names are in violation of ISO-11179 naming rules. The "tbl-" prefix, names like "type_id", "category_id", etc. are textbook examples of how not to write SQL.

2) You do not understand how a SELECT statement works. Here is how a SELECT works in SQL ... at least in theory. Real products will optimize things when they can.

a) Start in the FROM clause and build a working table from all of the joins, unions, intersections, and whatever other table constructors are there. The table expression > AS <correlation name> option allows you give a name to this working table which you then have to use for the rest of the containing query.

b) Go to the WHERE clause and remove rows that do not pass criteria; that is, that do not test to TRUE (reject UNKNOWN and FALSE). The WHERE clause is applied to the working set in the FROM clause.

c) Go to the optional GROUP BY clause, make groups and reduce each group to a single row, replacing the original working table with the new grouped table. The rows of a grouped table must be group characteristics: (1) a grouping column (2) a statistic about the group (i.e. aggregate functions) (3) a function or (4) an expression made up those three items.

d) Go to the optional HAVING clause and apply it against the grouped working table; if there was no GROUP BY clause, treat the entire table as one group.

e) Go to the SELECT clause and construct the expressions in the list. This means that the scalar subqueries, function calls and expressions in the SELECT are done after all the other clauses are done. The "AS" operator can also give names to expressions in the SELECT list. These new names come into existence all at once, but after the WHERE clause, GROUP BY clause and HAVING clause has been executed; you cannot use them in the SELECT list or the WHERE clause for that reason.

If there is a SELECT DISTINCT, then redundant duplicate rows are removed. For purposes of defining a duplicate row, NULLs are treated as matching (just like in the GROUP BY).

f) Nested query expressions follow the usual scoping rules you would expect from a block structured language like C, Pascal, Algol, etc. Namely, the innermost queries can reference columns and tables in the queries in which they are contained.

g) The ORDER BY clause is part of a cursor, not a query. The result set is passed to the cursor, which can only see the names in the SELECT clause list, and the sorting is done there. The ORDER BY clause cannot have expression in it, or references to other columns because the result set has been converted into a sequential file structure and that is what is being sorted.

As you can see, things happen "all at once" in SQL, not from left to right as they would in a sequential file/procedural language model. In those languages, these two statements produce different results:

```
READ (a, b, c) FROM File_X;  
READ (c, a, b) FROM File_X;
```

while these two statements return the same data:

```
SELECT a, b, c FROM Table_X;  
SELECT c, a, b FROM Table_X;
```

Think about what a confused mess this statement is in the SQL model.

```
SELECT f(c2) AS c1, f(c1) AS c2 FROM Foobar;
```

That is why such nonsense is illegal syntax.

--CELKO--

Please post DDL, so that people do not have to guess what the keys, constraints, Declarative Referential Integrity, datatypes, etc. in your schema are. Sample data is also a good idea, along with clear specifications.

*** Sent via Developersdex <http://www.developersdex.com> ***
Don't just participate in USENET...get rewarded for it!