Many-to-Many Relationships and All

Posted At : July 9, 2008 7:00 AM | Posted By : Steve
Related Categories: SQL

Most of us have deal with many-to-many relationships from time to time. This is generally easy. What happens, though, when the client wants some items to be available to all members of the related table?

For example, I might have some articles that are only available to users with certain permissions.

If that were the case, then I might have an articles2permissions table. Then I might pass in a comma-delimited list of permissions when retrieving articles:

```
SELECT   *
FROM   articles
WHERE   1 = 1
   AND   EXISTS (
         SELECT   1
         FROM   articles2permissions
         WHERE   articles2permissions.article_id = articles.article_id
            AND   permission_id IN (<cfqueryparam value="#arguments.permissions#" cfsqltype="CF_SQL_INTEGER"
list="yes">)
      )
```

I often see an INNER JOIN used here instead of EXISTS. I prefer EXISTS because it seems to read better as English and because it doesn't get in the way of any other joins that might be needed (and allows for more flexibility).

The subquery has "SELECT 1" because I want it to be clear in the code that I don't actually need any real data from the subquery.

This solves the problem of relating articles to permissions. Now, what if the client wants this article available to all permissions?

One solution that I have seen is to add a button (or some other UI element) in the administrative form for articles that uses JavaScript to select each permission on that form. This does work in the short run, but it isn't really a good long-term solution.

What if permission is added after this point? How would you know which articles are supposed to have all permissions and which ones just happen to have all of the existing permissions checked?

I prefer to have a field that indicates the "all" relationship. So, in this case I might add an "allPermissions" field to the articles table. This would allow me to change the previous query to the following:

```
SELECT   *
FROM   articles
WHERE   1 = 1
   AND   (
            allPermissions = 1
         OR   EXISTS (
               SELECT   1
               FROM   articles2permissions
               WHERE   articles2permissions.article_id = articles.article_id
                  AND   permission_id IN (<cfqueryparam value="#arguments.permissions#" cfsqltype="CF_SQL_INTEGER"
list="yes">)
            )
      )
```

This achieves the desired result. Any article that has allPermissions selected will appear regardless of the users permissions. Other articles will only appear if the user has the appropriate permissions.

I have the two SQL statements in one parenthetical "AND" statement so that they won't interfere with any other criteria that might exist (or need to exist in the future) in this query.

Now if an article is selected to be available to all permissions, then it will be even if other permissions are added after the article is entered. No fuss, no muss, no worries.