

Re: bitwise and

Source:

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.programming/2004-12/1523.html>

From: Ben (*Ben_at_discussions.microsoft.com*)

Date: 12/06/04

Date: Mon, 6 Dec 2004 14:53:02 -0800

Just to add my two cents:

The bottom line here is that you need some method of determining what access a user has to your application. In addition, you don't want to present them with application functions to which they do not have permissions.

Using a binary method is cryptic, slow and inflexible. Already you are asking how big to make the field. Most likely you will not make it big enough. So use the concepts of normalization and create a structure that can expand on demand without changing your application or data structure.

Others have stated that you should not integrate with the granting structure built into SQL Server. I have real world experience with this. I did make an application that used the SQL Server groups for granting permissions adding my own table of permissions and a many to many relationship between those permissions and the SQL Server groups. This allowed me to create a user in one centralized interface and assign them to groups enabling both back end security and front end functionality.

THIS WILL NOT ALWAYS WORK. The application represented here was quite simple, and the security was not very granular. Still, if you are going to have application permissions you must persist this information somewhere. So what options are left:

1) If you are using a database system that actually creates an SQL Server login for each users (rare these days) then you can still use the method mentioned above. However, you will most likely not gain much over the assignment of a User ID because you will need a different interface for creating permissions.

2) Most applications today use connection pools and other mechanisms to reduce costs in database connections and therefore don't create an actual SQL Server account for every user. In this case you are going to have your own user table anyway. Just persist their permissions in a structure in your database. I would recommend the following entities:

Users
Permissions
Users_Permissions

Cheers,

Ben
"David Portas" wrote:

> *Bitmap columns are a very poor way to store data in SQL. A basic principle*
> *of good database design is that each column should represent a single*
> *attribute. You haven't stated any reason to compromise that principle.*
>
> *SQL Server supports user-defined roles for exactly the purpose you have*
> *described. See Books Online for details. Alternatively you could create a*
> *table something like the following to assign users to your own roles or*
> *groups.*
>
> *CREATE TABLE UserGroups (user_id INTEGER NOT NULL REFERENCES Users*
> *(user_id), group_code CHAR(4) NOT NULL REFERENCES Groups (group_code),*
> *PRIMARY KEY (user_id, group_code))*
>
> *Then validate permissions like this:*
>
> *EXISTS*
> *(SELECT **
> *FROM UserGroups*
> *WHERE group_code = 'XXXX'*
> *AND user_id = @current_user)*
>
> --
> *David Portas*
> *SQL Server MVP*
> --
>
>
>