GROUP BY

Let's look at a few different uses of GROUP BY.

Compare these two queries:

SELECT UserName FROM Users GROUP BY UserName;

SELECT UserID FROM Users GROUP BY UserID;

The first query will group together any users who happen to have the same name. The second query will not group together users who have the same name but are not the same user. UserID is a primary key, which means that it must be unique (and not null, but that's not relevant here). That uniqueness is a way to be sure not to group together different people.

You probably want the user's name, not the UserID. You'll be tempted to write a query like the following, which --spoiler alert-- won't work.

SELECT UserName FROM Users GROUP BY UserID;

Why doesn't this work? In order to select a column (or expression that includes a column value) with a GROUP BY, you need a column (or expression) that has one value that represents the entire group for each group. As humans, we know that if we GROUP BY UserID, then there is one name for the entirety of each group, because if there was a different person with a different name, there would be a different UserID, which means a different group if we GROUP BY UserID. However, SQL is not that smart. It does not realize what we know. In order to SELECT UserName, we need to include it in the GROUP BY. Since we don't want to return to the problem of our very first query (which would group together anyone with the same name), we do this:

SELECT UserName FROM Users GROUP BY UserID,UserName;

We know that UserID defines UserName, which means that it isn't possible to have two users with the same UserID but different UserNames. The result of this is that GROUP BY UserID,UserName will group rows the same as GROUP BY UserID. The only difference is that SQL has been clued in, and it will now allow us to SELECT UserName.

One more thing that is related: As stated earlier, if you user a GROUP BY, each expression you select must represent every row in each group. This means you can SELECT a column that was in the GROUP BY, and you can also select an aggregate function. By definition, an aggregate function gives a value that represents every row in a group, so it can be selected with a GROUP BY. Something like this:

```
SELECT Count(UserID),SUM(UserPoints),MIN(UserPoints),MAX(UserPoints) FROM Users GROUP BY UserID;
```