TSQL Basics Part 16: EXCEPT INTERSEPT UNION

This is part 16 of a 19 part series on TSQL Basics. You will first gain an understanding of the differences between each of these types of JOINS, and when they should be used. Then we'll explore some common uses for JOINS, such as replacing a not in clause with JOIN to get the same results with better performance. You will take home the right foundations to get at the data that you want. Make sure to check of the other parts in this series in my JOIN Types playlist.

Transcription:

0:07
that are on the poster, but they're technically not in the presentation. So if we go back and look at table one and table two, again, there's the concept of accept and intersect, which I've represented in the Venn diagrams on the poster, but they're technically not join types. What it's saying is give me all the IDS from table one, but throw out all the IDs that are in table two. And there that shows up as null. And if we run it the other way around, give me all the IDS from table two that don't show up or in throw out all of those that are actually in table one, we can see that IDs of four, six and seven are missing. And because of that, that maps to what was at orange, mauve and something else that were not chosen. So it's sort of an another way to get it similar type of data that we were seeing on the left and right outer joins with exclusions. Now this is one of those things when we do corruption repair, I like to compare results of tables before and after. And you can use the Accept statement like this to say compare everything from one table to everything from another table and see what the results look like. intersect is similar. But instead of saying throw out specific rows, it says only show me those rows that match up. This is more like an inner join. And then union is sort of like the full OUTER JOIN, which says give me everything from both tables. And including the Nol. And it gives all the IDS there that are that are being used. And if you want to include duplicates, you can use the union all to bring those together and see that some of them are accessed more than one time. Or the union all is way more efficient than the union itself because when you just use straight union, it has to sort the entire result set to throw out the duplicates, whereas the union all doesn't. Anyway, just a little bit different way of looking at some data there. That's also included included in the join types poster even though they're truly not going types. Also, just a reminder, you can download the join types poster at SteveStedman.com and this is a poster I created a few years ago that just sort of shows how all the different joins that we're going to be covering through this presentation are done. It's handy to print out and keep on your cube wall or near your workspace.