Set Operators

Section Recap

This section was about SET OPERATORS, which are the operators you use to combine the results of 2 queries into a single result set, and you learned that these queries that include SET operators are usually called COMPOUND QUERIES.

The first thing you learned about SET OPERATORS is that the queries you are going to combine must return the same number of columns, and each column must have the same or at least a compatible data type in both queries.

We also saw that a compound query can have only 1 order by clause, which must be specified at the end of the last component query.

So, the first 2 operators we looked at were UNION and UNION ALL, and they are pretty similar.

They combine the results of the 2 queries, into a single result set, so the final result set includes the rows returned by the first query and the rows returned by the second query.

Here is an example:

```
SELECT id, job_id
FROM employee
UNION ALL
SELECT id, name
FROM department
ORDER BY 1;
```

The difference between UNION and UNION ALL is that after combining the results of the queries UNION removes duplicate rows from the final result set, whereas UNION ALL does not remove duplicates.

This is an important difference from the performance perspective, so, if you don't have the intention of removing duplicates, or you know that no duplicates will result from the combined queries, you should use UNION ALL, to avoid the extra work that is required to verify if there are duplicates and to eliminate them.

Also, remember that the names of the columns in the final result set are those from the first component query.

After UNION and UNION ALL, we looked at the INTERSECT operator.

This operator returns the rows that are included in the results of the first query and in the results of the second query as well, so for a row to be returned, it must be returned by both component queries. And after doing the intersection, duplicate rows are removed in this case too.

And the other set operator we covered was MINUS, which performs a subtraction of result sets, so, both queries are executed, and then if there are any rows that were returned by the second query which were also included in the results of the first one, they are removed, then duplicate rows are eliminated, and finally only the remaining rows from the first query are returned.

And to conclude the section, we saw that SET operators are evaluated in the order in which they appear (unless parentheses force a different order), so, in this example:

```
SELECT id
FROM department
WHERE monthly_budget > 15000
INTERSECT
SELECT department_id
FROM employee
WHERE salary BETWEEN 2000 AND 2500
UNION
SELECT id
FROM department
WHERE id = 5
MINUS
SELECT id
FROM department
WHERE id = 4;
```

In the first step, the first 2 queries are combined using the INTERSECT operator. Then, the results from that intersection are combined with the next query using the UNION operator, and then the results of this UNION are combined with the final component query using the MINUS operator.

But you also learned that you can force a specific order of evaluation by using parentheses.
Okay, you are now ready for the VERY INTERESTING section that comes next, where you will learn about JOINS!

Yes, that is a very important topic, so I look forward to seeing you there.

Take care!

