**Error Contest DB**

* **The error that occurred in my program**

The ORDER BY clause is invalid in views, inline functions, derived tables, subqueries, and common table expressions, unless TOP, OFFSET or FOR XML is also specified.

* **Command that generated it**

Occured when trying to get the companies from the opposition that have the highest position on the chart of the "best" companies.

**SELECT DISTINCT o.oppositon\_name, o.position\_in\_antivirus\_chart**

**FROM**

**(**

**SELECT O.oppositon\_name, O.position\_in\_antivirus\_chart**

**FROM Opposition O INNER JOIN Antivirus\_soft A ON A.opposition\_id = O.opposition\_id**

**ORDER BY position\_in\_antivirus\_chart DESC**

**)o**

* **Explanation of the error (my approach)**

Reading the error, I identified the keyword ‘subquery‘ , and then I realised that my ORDER BY is in a subquery which is invalid according to the error message. However, I did not exactly understand why, since I could see that working. Consequently, I did what any student would do and searched the error on Google and entered the first StackOverflow link that popped into my browser. Oddly, neither from here, nor from the other Computer-Science related forums I did not get a clear explanation, but rather what I deduced from the error too. The answer came from a more formal resource, namely learn.Microsoft and related that this issue is due to the way SQL Server optimizes queries. Moreover (quite logically), the database engine must know how many rows to return or how to structure the result before applying ordering(**ORDER BY**).

* **Fixing the error**

On the same page where I learn in depth about the problem, a solution was given to solve it, which went like **‘ASC OFFSET 0 ROWS’,** but honestly I did not unterstand properly how this would fix it. However, I observed the use of OFFSET (keyword from the error message), and I deduced that whatever I will use from the mentioned words will work, since it would prevent the ambiguity propagated to the SQL Server. Thus, I creatively thought about using **TOP(**which I also knew from usage at Lab2 homework :D ), but in a way that it will not affect the ordering or the subquery but only provide the necessary conditions for the DB engine to complete the query. Concretely I used ‘**TOP 100 PERCENT’**, such that with this the ordering will work without error and the conditions imposed by the Server will be satisfied. So, the final query was:

**SELECT DISTINCT o.oppositon\_name, o.position\_in\_antivirus\_chart**

**FROM**

**(**

**SELECT TOP 100 PERCENT O.oppositon\_name, O.position\_in\_antivirus\_chart**

**FROM Opposition O INNER JOIN Antivirus\_soft A ON A.opposition\_id = O.opposition\_id**

**ORDER BY position\_in\_antivirus\_chart DESC**

**)o**

* **Conclusion**

Even though it will surely not be the most interesting or bulky error sent, I consider it quite an interesting way to ‘**trick**’ the Database Engine into doing what you want by imposing a condition that is not necessarily needed. Ending the documentation with a self-invented phrase, which reflects my error talk, that might be considered untrue in the uprising **AI trend:**

*“Power lies not in machines, but in the coders who wield mastery."*

**Bugnaru Tudor Eduard**

**Group 921**