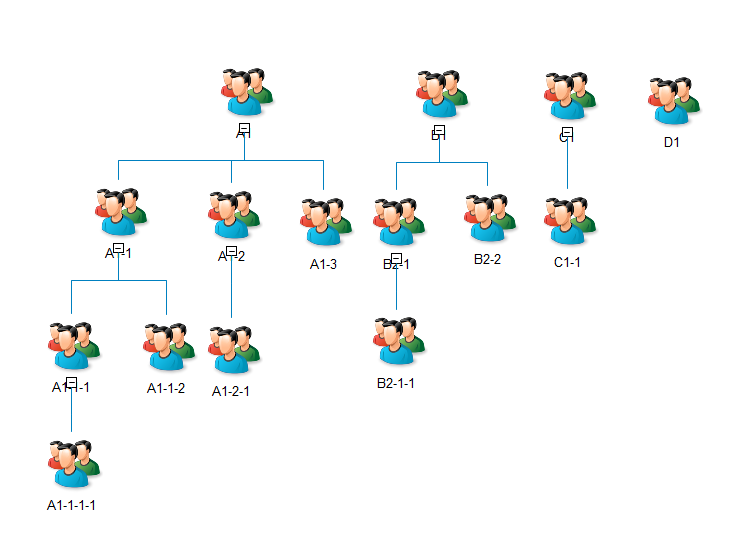
**SQL Hierarchical Queries without using Recursion and Cursors (The life before CTE in SQL Server)**

**…And Why We Need That Now! (A Modern Use Case)**

Remember those old times when we had to write scripts to find the ancestors or descendants of a particular node in a hierarchical tree structure stored within the RDBMS table. Recall that scripting circus we went through using cursors and/or recursive functions, which possibly triggered various side effects and performance issues.

As we were expecting heavy load on our application, we were totally against the use of cursors or recursions in our project back then. However, I couldn’t find any online or offline solution and also our DBA friends couldn’t suggest a better alternative. But I had an intuition that it could be possible and decided to give it a try myself. Eventually found a simple and elegant solution using a ‘while’ loop and a temporary table with identity column, which saved us a lot of time, effort and of course some extra CPU cycles.

After the introduction of Common Type Expression (CTE) which was released with Microsoft SQL Server 2005 version, a temporary named result set mechanism, I never had to use the above mentioned legacy solution. But suprisingly something happened recently in a project and I had to resort to the same classical solution once again. Actually, that inspired me to share the original script here after a decade and half. I guess it might be helpful for the people who are facing similar issues with corrupted hierarchical data.

Let’s start from the beginning. Check the below multiple organizations tree structure diagram

Read the full article here:

Medium: <https://medium.com/@sanish.abraham/sql-hierarchical-queries-without-recursion-and-cursors-the-life-before-cte-in-sql-server-bf7745e382de>

Blogspot: <https://saneecodes.blogspot.com/>

LinkedIn: <https://www.linkedin.com/in/sanishabraham/>

GitHub: <https://github.com/mailsanish/HierarchicalQueries>