**Developer case -**

**Microsoft SQL**

# General

This exercise aims to see how to solve an SQL task.

# Specification

Develop a stored procedure that returns an organization tree with associated groups. Each group leader must also be included in his underlying group. One tip is to create fictitious groups to get to the levels.  
  
The basic data has a complete structure according to:

1 (CEO)  
 1.1 (team lead)  
 1.1.1 (team member)  
 1.2 (team lead)  
 1.2.1 (team member)  
 1.3 (team member)

The names are just to clarify the hierarchy. They can be anything really, people, products, etc.

Run Base data.sql against a database to access the data.

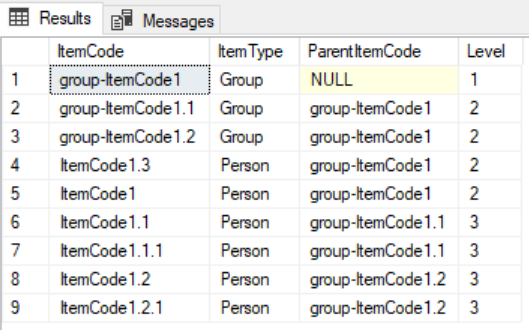
|  |  |
| --- | --- |
| **Field** | **Description** |
| ItemCode | Node key |
| ItemType | Type, ”Person” or ”Group”. |
| ParentItemCode | Parent of node. The root node has NULL as parent |
| Level | Tree level. 1, 2, 3… |

# Results

The tree should be returned with the same fields as the base data and hierarchy as follows:

1 (team1)  
 1 (CEO)  
 1.1 (team)  
 1.1 (team lead)  
 1.1.1 (team member)  
 1.2 (team)  
 1.2 (team lead)  
 1.2.1 (team member)  
 1.3 (team member)

Which gives the following results:



Note that no groups should be created on members without children (eg 1.3).

## The solution must be dynamic, so the tree should allow you to change the base data. If you add a group, or level, the SQL must be designed to adapt to the changes in the data and customize the tree after it.

## Input parameters

None.

# Sorting

Level.

# Technology and tools

Microsoft SQL Server