Белорусский государственный технологический университет

факультет информационных технологий

кафедра информационных систем и технологий

**Методы сбора, хранения, обработки и анализа данных**

**Лабораторная работа №3**

Разработал Лопатнюк П.В.

ФИТ, 3 курс, группа 1/1

Проверил Нистюк О. А.

Минск 2024

1.Для базы данных в СУБД SQL Server добавить для одной из таблиц столбец данных иерархического типа.

|  |
| --- |
| ALTER TABLE Teacher  ADD PositionHierarchy HIERARCHYID; |

2.Создать процедуру, которая отобразит все подчиненные узлы с указанием уровня иерархии (параметр – значение узла).

|  |
| --- |
| GO  CREATE PROCEDURE GetTeacherHierarchy  @TeacherID INT  AS  BEGIN  DECLARE @HierarchyID HIERARCHYID;  SELECT @HierarchyID = PositionHierarchy FROM Teacher WHERE TeacherID = @TeacherID;  SELECT  T.TeacherID,  T.FullName,  T.Role,  T.DepartmentID,  T.PositionHierarchy.ToString() AS HierarchyPath,  T.PositionHierarchy.GetLevel() AS HierarchyLevel  FROM Teacher T  WHERE T.PositionHierarchy.IsDescendantOf(@HierarchyID) = 1  ORDER BY T.PositionHierarchy;  END; |

3.Создать процедуру, которая добавит подчиненный узел (параметр – значение узла).

|  |
| --- |
| GO  CREATE OR ALTER PROCEDURE AddTeacher  @FullName NVARCHAR(200),  @DepartmentID INT,  @Role NVARCHAR(200),  @ParentTeacherID INT = NULL  AS  BEGIN  DECLARE @NewTeacherID INT = NEXT VALUE FOR seq\_teacher\_id;  DECLARE @PositionHierarchy HIERARCHYID;  DECLARE @LastChild HIERARCHYID;  DECLARE @ParentHierarchy HIERARCHYID;  IF @ParentTeacherID IS NULL  BEGIN  SET @PositionHierarchy = HIERARCHYID::GetRoot();  END  ELSE  BEGIN  SELECT @ParentHierarchy = PositionHierarchy  FROM Teacher  WHERE TeacherID = @ParentTeacherID;  SELECT @LastChild = MAX(PositionHierarchy)  FROM Teacher  WHERE PositionHierarchy.GetAncestor(1) = @ParentHierarchy;  SET @PositionHierarchy = @ParentHierarchy.GetDescendant(@LastChild, NULL);  END  INSERT INTO Teacher (TeacherID, FullName, DepartmentID, Role, PositionHierarchy)  VALUES (@NewTeacherID, @FullName, @DepartmentID, @Role, @PositionHierarchy);  END; |

4.Создать процедуру, которая переместит всю подчиненную ветку (первый параметр – значение верхнего перемещаемого узла, второй параметр – значение узла, в который происходит перемещение).

|  |
| --- |
| GO  CREATE OR ALTER PROCEDURE MoveTeacherBranch (  @SourceTeacherID INT,  @TargetTeacherID INT  )  AS  BEGIN  DECLARE @SourceHierarchy HIERARCHYID;  DECLARE @TargetHierarchy HIERARCHYID;  DECLARE @NewHierarchy HIERARCHYID;  SELECT @SourceHierarchy = PositionHierarchy FROM Teacher WHERE TeacherID = @SourceTeacherID;  SELECT @TargetHierarchy = PositionHierarchy FROM Teacher WHERE TeacherID = @TargetTeacherID;  IF @TargetHierarchy.IsDescendantOf(@SourceHierarchy) = 1  BEGIN  RAISERROR('Невозможно переместить узел в его потомка', 16, 1);  RETURN;  END  DECLARE @LastChild HIERARCHYID;  SELECT @LastChild = MAX(PositionHierarchy) FROM Teacher WHERE PositionHierarchy.GetAncestor(1) = @TargetHierarchy;  SET @NewHierarchy = @TargetHierarchy.GetDescendant(@LastChild, NULL);  WITH CTE\_Teachers AS (  SELECT TeacherID, PositionHierarchy  FROM Teacher  WHERE PositionHierarchy.IsDescendantOf(@SourceHierarchy) = 1  )  UPDATE T  SET T.PositionHierarchy = CTE.PositionHierarchy.GetReparentedValue(@SourceHierarchy, @NewHierarchy)  FROM Teacher T  JOIN CTE\_Teachers CTE ON T.TeacherID = CTE.TeacherID;  END; |

1.Для базы данных в СУБД SQL Server добавить для одной из таблиц столбец данных иерархического типа.

|  |
| --- |
| ALTER TABLE Teacher  ADD ParentID NUMBER; |

2.Создать процедуру, которая отобразит все подчиненные узлы с указанием уровня иерархии (параметр – значение узла).

|  |
| --- |
| DECLARE  CURSOR c IS  SELECT t.TeacherID,  t.FullName,  LEVEL AS HierarchyLevel,  SYS\_CONNECT\_BY\_PATH(t.TeacherID, '/') AS HierarchyPath -- Изменено на TeacherID  FROM Teacher t  JOIN TeacherHierarchy th ON th.ChildTeacherID = t.TeacherID  START WITH t.TeacherID = 5  CONNECT BY PRIOR t.TeacherID = th.ParentTeacherID;  BEGIN  FOR rec IN c LOOP  DBMS\_OUTPUT.PUT\_LINE(rec.TeacherID || ' ' || rec.FullName || ' | Path: ' || rec.HierarchyPath || ' | Level: ' || rec.HierarchyLevel);  END LOOP;  END; |

3.Создать процедуру, которая добавит подчиненный узел (параметр – значение узла).

|  |
| --- |
| CREATE OR REPLACE PROCEDURE AddTeacher (  p\_FullName IN NVARCHAR2,  p\_DepartmentID IN NUMBER,  p\_Role IN NVARCHAR2,  p\_ParentTeacherID IN NUMBER DEFAULT NULL  ) AS  v\_NewTeacherID NUMBER;  BEGIN  SELECT seq\_teacher\_id.NEXTVAL INTO v\_NewTeacherID FROM dual;  INSERT INTO Teacher (TeacherID, FullName, DepartmentID, Role)  VALUES (v\_NewTeacherID, p\_FullName, p\_DepartmentID, p\_Role);  IF p\_ParentTeacherID IS NOT NULL THEN  INSERT INTO TeacherHierarchy (ParentTeacherID, ChildTeacherID)  VALUES (p\_ParentTeacherID, v\_NewTeacherID);  END IF;  END; |

4.Создать процедуру, которая переместит всю подчиненную ветку (первый параметр – значение верхнего перемещаемого узла, второй параметр – значение узла, в который происходит перемещение).

|  |
| --- |
| CREATE OR REPLACE PROCEDURE MoveTeacherBranch (  p\_SourceTeacherID IN NUMBER,  p\_TargetTeacherID IN NUMBER  ) IS  BEGIN  UPDATE TeacherHierarchy  SET ParentTeacherID = p\_TargetTeacherID  WHERE ChildTeacherID = p\_SourceTeacherID  OR ChildTeacherID IN (SELECT ChildTeacherID  FROM TeacherHierarchy  START WITH ParentTeacherID = p\_SourceTeacherID  CONNECT BY PRIOR ChildTeacherID = ParentTeacherID);  COMMIT;  END;  BEGIN  MoveTeacherBranch(3, 5);  END; |