<https://docs.microsoft.com/en-us/sql/relational-databases/tables/lesson-2-2-populating-a-hierarchical-table-using-hierarchical-methods>

# Lesson 2-2 - Populating a Hierarchical Table Using Hierarchical Methods

2017-3-6 3 min to read Contributors

* [BYHAM](https://github.com/BYHAM)

* [Craig Guyer](https://github.com/craigg-msft)

**AdventureWorks2012** has 8 employees working in the Marketing department. The employee hierarchy looks like this:

**David**, **EmployeeID** 6, is the Marketing Manager. Three Marketing Specialists report to **David**:

* **Sariya**, **EmployeeID** 46
* **John**, **EmployeeID** 271
* **Jill**, **EmployeeID** 119

Marketing Assistant **Wanida** (**EmployeeID** 269), reports to **Sariya**, and Marketing Assistant **Mary** (**EmployeeID** 272), reports to **John**.

### To insert the root of the hierarchy tree

1. The following example inserts **David** the Marketing Manager into the table at the root of the hierarchy. The **OrdLevel** column is a computed column. Therefore, it is not part of the INSERT statement. This first record uses the [GetRoot()](https://docs.microsoft.com/en-us/sql/t-sql/data-types/getroot-database-engine) method to populate this first record as the root of the hierarchy.

Copy

INSERT HumanResources.EmployeeOrg (OrgNode, EmployeeID, EmpName, Title)

VALUES (hierarchyid::GetRoot(), 6, 'David', 'Marketing Manager') ;

GO

1. Execute the following code to examine initial row in the table:

Copy

SELECT OrgNode.ToString() AS Text\_OrgNode,

OrgNode, OrgLevel, EmployeeID, EmpName, Title

FROM HumanResources.EmployeeOrg ;

Here is the result set.

Copy

Text\_OrgNode OrgNode OrgLevel EmployeeID EmpName Title

------------ ------- -------- ---------- ------- -----------------

/ Ox 0 6 David Marketing Manager

As in the previous lesson, we use the ToString() method to convert the **hierarchyid** data type to a format that is more easily understood.

### To insert a subordinate employee

1. **Sariya** reports to **David**. To insert **Sariya's** node, you must create an appropriate **OrgNode** value of data type **hierarchyid**. The following code creates a variable of data type **hierarchyid** and populates it with the root OrgNode value of the table. Then uses that variable with the [GetDescendant()](https://docs.microsoft.com/en-us/sql/t-sql/data-types/getdescendant-database-engine) method to insert row that is a subordinate node. GetDescendant takes two arguments. Review the following options for the argument values:
   * If parent is NULL, GetDescendant returns NULL.
   * If parent is not NULL, and both child1 and child2 are NULL, GetDescendant returns a child of parent.
   * If parent and child1 are not NULL, and child2 is NULL, GetDescendant returns a child of parent greater than child1.
   * If parent and child2 are not NULL and child1 is NULL, GetDescendant returns a child of parent less than child2.
   * If parent, child1, and child2 are all not NULL, GetDescendant returns a child of parent greater than child1 and less than child2.

The following code uses the (NULL, NULL) arguments of the root parent because there are not yet any rows in the table except the root. Execute the following code to insert **Sariya**:

Copy

DECLARE @Manager hierarchyid

SELECT @Manager = hierarchyid::GetRoot()

FROM HumanResources.EmployeeOrg ;

INSERT HumanResources.EmployeeOrg (OrgNode, EmployeeID, EmpName, Title)

VALUES

(@Manager.GetDescendant(NULL, NULL), 46, 'Sariya', 'Marketing Specialist') ;

1. Repeat the query from the first procedure to query the table and see how the entries appear:

Copy

SELECT OrgNode.ToString() AS Text\_OrgNode,

OrgNode, OrgLevel, EmployeeID, EmpName, Title

FROM HumanResources.EmployeeOrg ;

Here is the result set.

Copy

Text\_OrgNode OrgNode OrgLevel EmployeeID EmpName Title

------------ ------- -------- ---------- ------- -----------------

/ Ox 0 6 David Marketing Manager

/1/ 0x58 1 46 Sariya Marketing Specialist

### To create a procedure for entering new nodes

1. To simplify entering data, create the following stored procedure to add employees to the **EmployeeOrg** table. The procedure accepts input values about the employee being added. This includes the **EmployeeID** of the new employee's manager, the new employee's **EmployeeID** number, and their first name and title. The procedure uses GetDescendant() and also the [GetAncestor()](https://docs.microsoft.com/en-us/sql/t-sql/data-types/getancestor-database-engine) method. Execute the following code to create the procedure:

Copy

CREATE PROC AddEmp(@mgrid int, @empid int, @e\_name varchar(20), @title varchar(20))

AS

BEGIN

DECLARE @mOrgNode hierarchyid, @lc hierarchyid

SELECT @mOrgNode = OrgNode

FROM HumanResources.EmployeeOrg

WHERE EmployeeID = @mgrid

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

BEGIN TRANSACTION

SELECT @lc = max(OrgNode)

FROM HumanResources.EmployeeOrg

WHERE OrgNode.GetAncestor(1) =@mOrgNode ;

INSERT HumanResources.EmployeeOrg (OrgNode, EmployeeID, EmpName, Title)

VALUES(@mOrgNode.GetDescendant(@lc, NULL), @empid, @e\_name, @title)

COMMIT

END ;

GO

1. The following example adds the remaining 4 employees that report directly or indirectly to **David**.

Copy

EXEC AddEmp 6, 271, 'John', 'Marketing Specialist' ;

EXEC AddEmp 6, 119, 'Jill', 'Marketing Specialist' ;

EXEC AddEmp 46, 269, 'Wanida', 'Marketing Assistant' ;

EXEC AddEmp 271, 272, 'Mary', 'Marketing Assistant' ;

1. Again, execute the following query examine the rows in the **EmployeeOrg** table:

Copy

SELECT OrgNode.ToString() AS Text\_OrgNode,

OrgNode, OrgLevel, EmployeeID, EmpName, Title

FROM HumanResources.EmployeeOrg ;

GO

Here is the result set.

Copy

Text\_OrgNode OrgNode OrgLevel EmployeeID EmpName Title

------------ ------- -------- ---------- ------- -----------------

/ Ox 0 6 David Marketing Manager

/1/ 0x58 1 46 Sariya Marketing Specialist

/1/1/ 0x5AC0 2 269 Wanida Marketing Assistant

/2/ 0x68 1 271 John Marketing Specialist

/2/1/ 0x6AC0 2 272 Mary Marketing Assistant

/3/ 0x78 1 119 Jill Marketing Specialist

The table is now fully populated with the Marketing organization.

## Next Task in Lesson

[Querying a Hierarchical Table Using Hierarchy Methods](https://docs.microsoft.com/en-us/sql/relational-databases/tables/lesson-2-3-querying-a-hierarchical-table-using-hierarchy-methods)