Email Distribution With Scopes

nVision Scopes are useful for running a single nVision Report Request to several individual nVision Excel files (instances). These instances can each represent a department, for example. It's like have an income statement for the entire company, but with an extra "where" clause per department, for each file/instance.

These instances can be output to the Web (Report Manager), File (file server) or to email.

With very little setup, you can easily choose a single email address for "all" instances for a given Report Request to send to. But with additional setup, you can have each "instance" go to its own "set of email addresses". There are two levels of complexity for this. You can have each "department instance" to have a single email address, or you can have a tree where different levels of the tree can contain email addresses.

In this chapter, we will show you how you can obtain this widest level of configuration, using a tree. This will allow both multiple levels of emails in the tree, as well as having two or more people on the email distribution. We also allow email setup to be by User Profile name, Security Role, or the actual email address (useful for external emails.)

This document will show you how to set this up in your PeopleSoft environment. There are only two "custom" objects that are created; two views that will be new to your system. The remaining steps can all be done from the online PIA pages (configuration).

We will be using Tree Manager to take full advantage of nVision Scopes and Email Distributions. (This set up also allows emails to be set at a higher Tree Node level.) This solution also allows you to use a Deptid Range when setting up the Leafs on your Tree.

We'll add special notes or additional information in these information boxes.

For the example in this Chapter, we are using two new views. If you create a solution where all of the email addresses are in one field (separated by semi-colons), stored in a table/view keyed by Department ID, you can use that table/view instead.

The PeopleBooks section titled "Creating Scope Definitions" can also be referenced alongside this Chapter. It shows how to utilize Scopes with the Manager ID from the Department Table.

Google "nvision peoplebooks Creating Scope Definitions" to find more.

Tools 8.54
https://docs.oracle.com/cd/E80738_01/pt854pbh2/eng/pt/tnvs/task_CreatingScopeDefinitions-074ec7.html

Full nVision PeopleBooks 8.57 (Search for "Creating Scope Definitions")
https://docs.oracle.com/cd/E99484_01/psft/pdf/pt857tnvs-b092018.pdf

Pre-Requisites

The following is necessary to perform this set up.

- Access to add two new "views" in the App Designer tool.
- Access to create a new Tree Structure, then a new Tree.
- Access to add a new Scope.
- Existing nVision report, that can easily be tied to a scope.
    - In our set up, we will use Deptid as the type of scope.
- Access to the following menu/component/page: TREEMANAGER/DEPT_NODE/DEPT_NODE_TBL.
    - This is a delivered page we utilize, to store multiple email distributions for each tree node.

One additional requirement remains. You can only set up one Tree with this Tree Structure. If you need two trees, it is best that the main view we will be creating soon (Z_DEPT_DISTVW) be hardcoded for the first tree, and another view be created for the second tree. Otherwise, the nVision Report Request won't know which Tree to use.

Two New Views

These two views begin with "Z". You are welcome to name them using your own custom naming convention. These are only referenced in the Tree Structure and the nVision Scope.

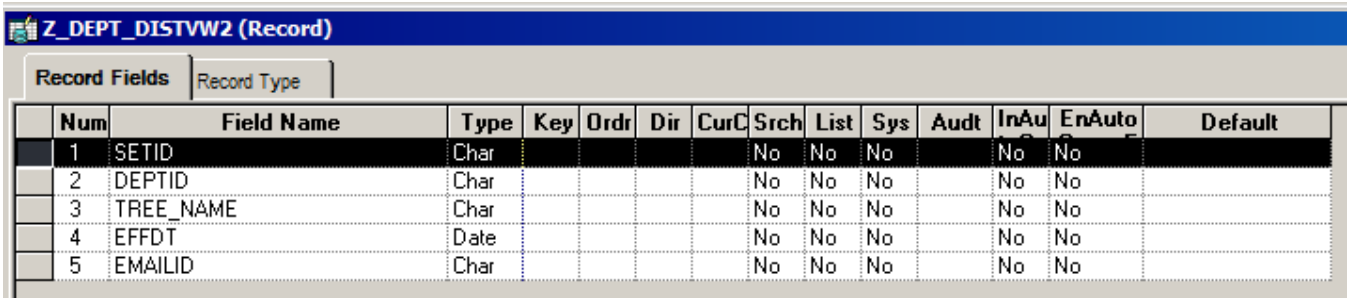
The first view (Z_DEPT_DISTVW2) pulls data from the DEPT_NODE_TBL page, tied in with the tree.

The second view (Z_DEPT_DISTVW) does an aggregate collection per deptid, from the first view.

View 1 (Z_DEPT_DISTVW2)

Fields in Record:

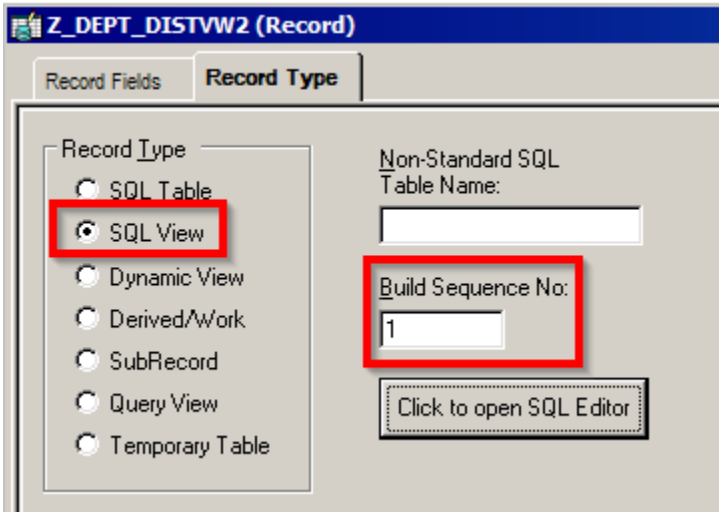
(No special Key, Alt Search Keys, List Box Items required.)



Num	Field Name	Type	Key	Ordr	Dir	CurC	Srch	List	Sys	Audt	InAu	EnAuto	Default
1	SETID	Char					No	No	No		No	No	
2	DEPTID	Char					No	No	No		No	No	
3	TREE_NAME	Char					No	No	No		No	No	
4	EFFDT	Date					No	No	No		No	No	
5	EMAILID	Char					No	No	No		No	No	

Record Type: View

(Use Build Sequence No 1)



SQL View Text:

```
SELECT DEPT.SETID
 , DEPT.DEPTID
 , TD.TREE_NAME
 , TD.EFFDT
 , CASE WHEN DAT.EMAILID <> ' ' THEN DAT.EMAILID WHEN DAT.DISTIDTYPE = '2' THEN 'U:' ||
DAT.DISTID WHEN DAT.DISTIDTYPE = '3' THEN 'R:' || DAT.DISTID ELSE 'ERROR' END
  FROM PS_DEPT_ACCESS_TBL DAT
  , PSTREENODE TN
  , PSTREELEAF TL
  , PSTREEDEFN TD
  , PSTREESTRCT TS
  , PS_DEPT_TBL DEPT
 WHERE TN.SETID = DAT.SETID
   AND TN.TREE_NODE = DAT.TREE_NODE
   AND TN.EFFDT = DAT.EFFDT

   AND TL.SETID = TN.SETID
   AND TL.TREE_NAME = TN.TREE_NAME
   AND TL.EFFDT = TN.EFFDT
   AND TL.TREE_NODE_NUM BETWEEN TN.TREE_NODE_NUM AND TN.TREE_NODE_NUM_END

   AND DEPT.DEPTID BETWEEN TL.RANGE_FROM AND TL.RANGE_TO
   AND DEPT.EFFDT = (SELECT MAX(DEPT_EFF.EFFDT) FROM PS_DEPT_TBL DEPT_EFF
                     WHERE DEPT_EFF.SETID = DEPT.SETID AND DEPT_EFF.DEPTID = DEPT.DEPTID
                       AND DEPT_EFF.EFFDT <= TD.EFFDT)
   AND DEPT.EFF_STATUS = 'A'

   AND TD.SETID = TN.SETID
   AND TD.TREE_NAME = TN.TREE_NAME
   AND TD.EFFDT = TN.EFFDT

   AND TD.EFF_STATUS = 'A'

   AND TS.TREE_STRCT_ID = TD.TREE_STRCT_ID
   AND TS.NODE_RECNAME = 'DEPT_NODE_TBL'

   AND ((DAT.EMAILID <> ' ' )
    OR (DAT.DISTIDTYPE IN ('2','3')
   AND DAT.DISTID <> ' ' ))
```

The first view is pulling in from any tree, where the tree structure has the Tree Node Record of DEPT_NODE_TBL.

We use a CASE statement to pull in the Email ID if it was entered manually on the page. If the Email ID is blank, but the User ID logic was used, we put a “U:” in front. Then we check if the Role logic was used, and we put a “R:” in front of that.

The goal of this view is to find all Trees using a special Tree structure, then tying that to the DEPT_ACCESS_TBL found on the DEPT_NODE_TBL page to get the distribution information. Then finding all active departments (based on the tree effective date) where the departments fall into the Tree Leaf range. The end result should be a listing of all active departments for a tree, with each distribution having its own line. (We’ll use the second View to aggregate these into a single row for each Tree Name, Tree Effdt, and Department combination.)

View 2 (Z_DEPT_DISTVW)

Fields in Record:

Make sure that DEPTID_DESCR is marked as a List Box Item.

Z_DEPT_DISTVW (Record)

Record Fields | Record Type

Num	Field Name	Type	Key	Ordr	Dir	CurC	Srch	List	Sys	Audt	InAu	EnAuto	Default
1	SETID	Char	Key	1	Asc		Yes	Yes	No		No	No	
2	TREE_NAME	Char	Key	2	Asc		Yes	Yes	No		No	No	
3	EFFDT	Date	Key	3	Asc		Yes	Yes	No		No	No	
4	DEPTID	Char	Key	4	Asc		Yes	Yes	No		No	No	
5	DEPTID_DESCR	Char	Alt		Asc		No	Yes	No		No	No	
6	EMAIL_LIST	Long					No	No	No		No	No	

Record Type: View

(Use Build Sequence No 2)

Z_DEPT_DISTVW (Record)

Record Fields | Record Type

Record Type
- SQL Table
- SQL View
- Dynamic View
- Derived/Work
- SubRecord
- Query View
- Temporary Table

Non-Standard SQL Table Name:

Build Sequence No: 2

Click to open SQL Editor

SQL View Test:

Oracle and DB2 Databases

```
SELECT DAT.SETID
 , DAT.TREE_NAME
 , DAT.EFFDT
 , DAT.DEPTID
 , DEPT.DESCR
 , LISTAGG(DAT.EMAILID, '; ') WITHIN GROUP (ORDER BY DAT.EMAILID)
FROM PS_Z_DEPT_DISTVW2 DAT
   , PS_DEPT_ALL_VW DEPT
WHERE DEPT.DEPTID = DAT.DEPTID
GROUP BY DAT.SETID, DAT.TREE_NAME, DAT.EFFDT, DAT.DEPTID, DEPT.DESCR

UNION

SELECT TD.SETID
 , TD.TREE_NAME
 , TD.EFFDT
 , DEPT.DEPTID
 , DEPT.DESCR
 , 'Trash'
FROM PS_DEPT_ALL_VW DEPT
   , PSTREEDEFN TD
   , PSTREESTRCT TS
WHERE TS.TREE_STRCT_ID = TD.TREE_STRCT_ID
  AND TS.NODE_RECNAME = 'DEPT_NODE_TBL'

  AND TD.EFF_STATUS = 'A'

  AND NOT EXISTS (SELECT 'X' FROM PS_Z_DEPT_DISTVW2 DAT
                  WHERE DAT.SETID = TD.SETID
                    AND DAT.TREE_NAME = TD.TREE_NAME
                    AND DAT.EFFDT = TD.EFFDT
                    AND DAT.DEPTID = DEPT.DEPTID)
```

MS SQL Server Databases (Starting with SQL Server 2017)

Change the portion in Yellow to be:

```
, STRING_AGG(DAT.EMAILID, '; ') WITHIN GROUP (ORDER BY DAT.EMAILID)
```

MS SQL Server Databases (SQL Server 2016 and below)

Change the portion in Yellow to be:

```
, STUFF((SELECT distinct ';' + DAT2.EMAILID
         from PS_Z_DEPT_DISTVW2 DAT2
         where DAT2.SETID = DAT.SETID
           AND DAT2.TREE_NAME = DAT.TREE_NAME
           AND DAT2.EFFDT = DAT.EFFDT
           AND DAT2.DEPTID = DAT.DEPTID
  FOR XML PATH('')), 1, LEN(';'), '')
```

We use a specific SQL statement to “aggregate” the email addresses into a single field when the SQL results are returned. For Oracle and DB2, we use LISTAGG. For MS SQL Server, we use STRING_AGG or STUFF, depending on your MS SQL Server version.

You will notice we do a Union, bringing in **any** department that isn’t already on our first view (Z_DEPT_DISTVW2). This is necessary so that when this view (Z_DEPT_DISTVW) is defined on the Tree Structure, users can still pick departments not already on the tree. It is necessary, in order to solve a classic “which came first, the chicken or the egg”, when adding Tree Leafs (Departments) to the Tree in an upcoming step. We need the department to be “selectable” for a Tree Leaf, but at the same time, the nVision Scope uses our Tree Structure to know “where to find the details” so it can do the email distribution. We plan to use this view (Z_DEPT_DISTVW) to be the Leaf information table, so that each Department found has the Email distribution list ready to be used by the nVision Scopes.

Stated in another way: we are using the Z_DEPT_DISTVW for both “finding which departments” can be added to a tree, as well as the “email distribution” found within the same tree, at different tree node levels.

New Tree Structure

We need to use a new Tree Structure, which leverages:

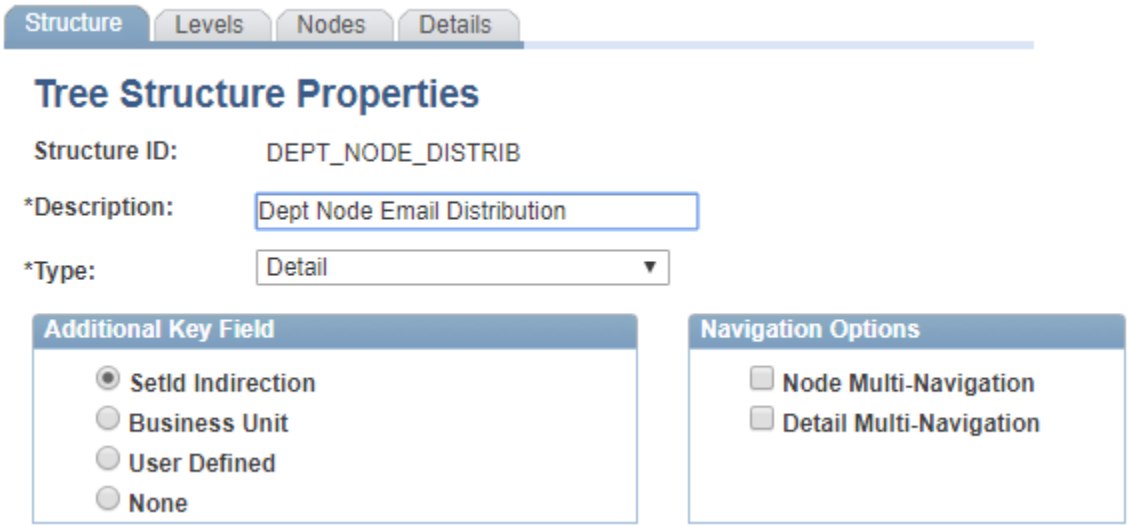
- a delivered page (DEPT_NODE_TBL) for linking Tree nodes to distributions (email ids, etc.)
- the delivered page for adding new departments, but with a twist on the record name used. (We'll use our new view Z_DEPT_DISTVW.)

Tree Structures can be added at the following location:

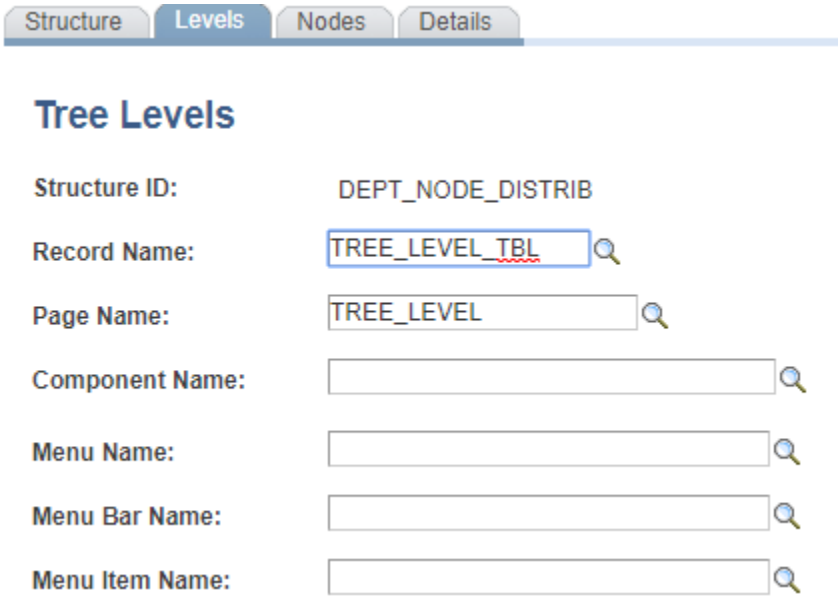
Tree Manager → Tree Structure

Add: DEPT_NODE_DISTRIB

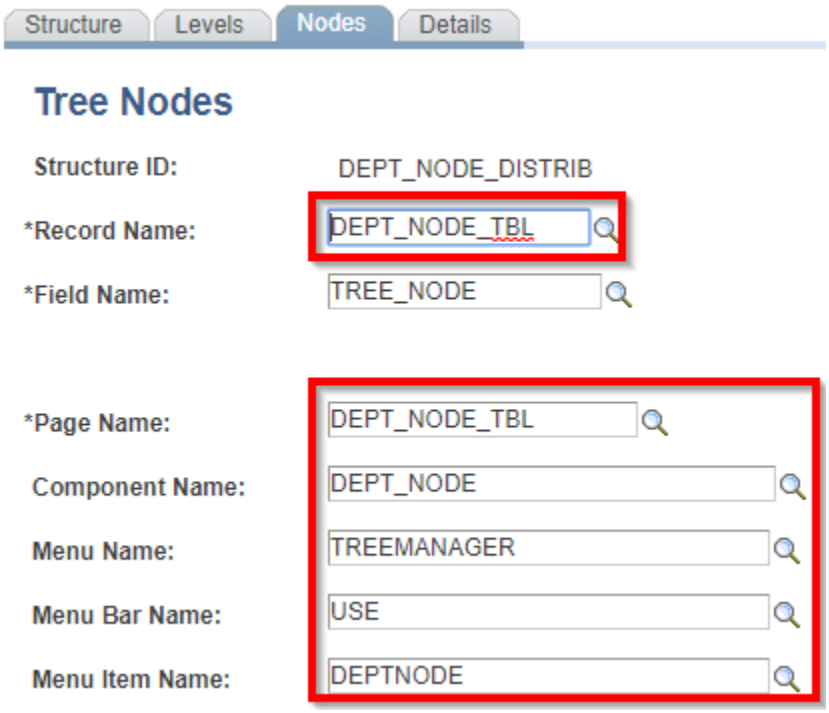
First Tab: Structure



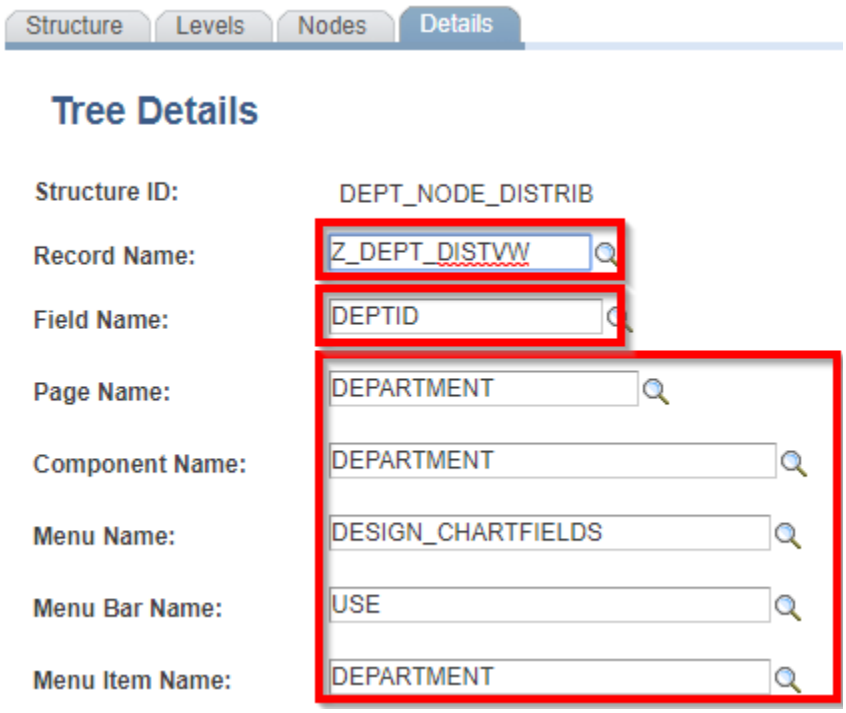
Second Tab: Levels



Third Tab: Nodes



Fourth Tab: Details



Your system may already have a tree structure called TREE_NODE_DISTRIB, which is similar to what we are building here. The difference is the example we build has both Nodes and Details/Leafs, where as the delivered structure TREE_NODE_DISTRIB only has Nodes.

Two important aspects to this Tree Structure:

- We use DEPT_NODE_TBL as the Node Record Name.
  - This allows us to tie a Tree Node to the Dept Node Distribution page delivered by PeopleSoft.
- We use Z_DEPT_DISTVW as the Detail Record Name.
  - This allows us to do two things:
    - Locate existing Departments (regardless if they have been set up in the tree, thanks to our special union in view Z_DEPT_DISTVW)
    - nVision Scope (we'll be creating soon) to have a list of Departments and which emails should be used for that Department.

New Tree

You need to create a new tree that corresponds to the new Tree Structure.

For our tree, we will use the Tree Name of DISTRIB_BY_DEPTID. (The Effdt Date is also very important.)

Tree Definition and Properties

*Tree Name: DISTRIB_BY_DEPTID

*Structure ID: DEPT_NODE_DISTRIB

*Effective Date: 01/01/1901 *Status: Active

*Description: Email Distrib By Deptid

*Category: DEFAULT

*Use of Levels: Level Not Used Performance Options

*SetID: SHARE

Audits

- All Detail Values in this Tree
- Allow Duplicate Detail Values

Perform Audits

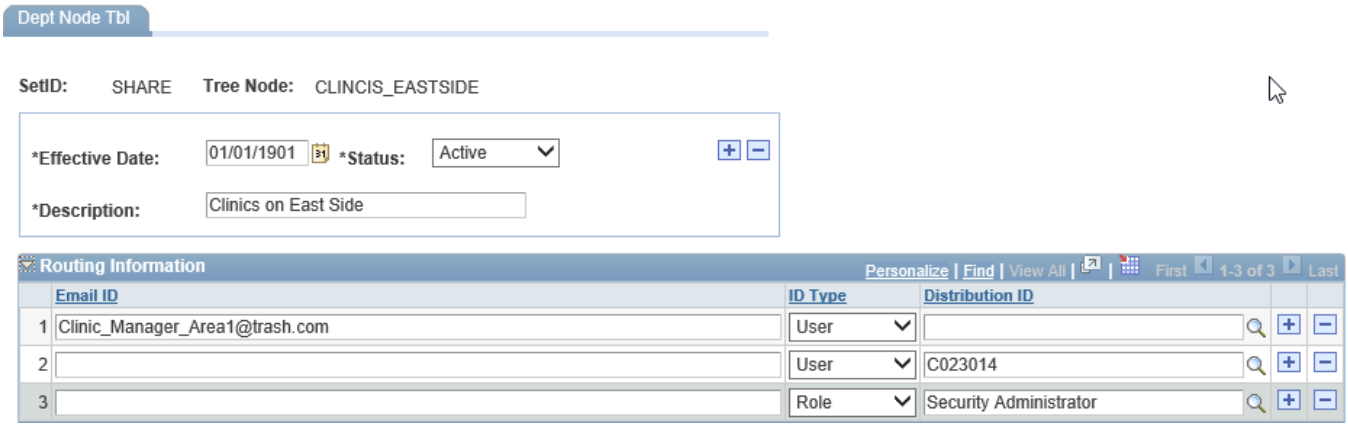
Item Counts

Node Count:	4
Leaf Count:	4
Level Count:	0
Branch Count:	0

You can now add several layer of nodes, to best define your distribution needs.

- ROOT - Root Node
  - CLINIC_LEADERSHIP - Clinic Leadership
    - CLINCIS_EASTSIDE - Clinics on East Side
      - [9473] - Neurology Physicians
      - [9474] - General Surgeons
    - CLINCS_WESTSIDE - Clinics on the West Side
      - [9475] - Hospitalists
      - [9477] - Orthopedic Surgeons

Here is an example of the data created for Tree Node "CLINICS_EASTSIDE". Notice we have a combination of external email address, User Ids, and Roles.



Here is a summary of which email addresses were placed for each Node. Notice we made a higher node configured for the email address Clinic_Leader@trash.com. This should allow all departments under this higher tree node, to be distributed to this leader.

Tree Node	Distribution Info
CLINIC_LEADERSHIP	Clinic_Leader@trash.com
CLINCIS_EASTSIDE	Clinic_Manager_Area1@trash.com User: C023014 Role: Security Administrator
CLINCS_WESTSIDE	User: C023016

The Tree Nodes are meant to organize the tree, as well as provide hooks for the email distributions. The Tree Leafs (Details) are to signify which Departments the Scopes will be created for. We can use both individual and ranges.

We'll be tying distributions per Tree Node Names, so use Node Names that are unique if you plan to have two reporting trees.

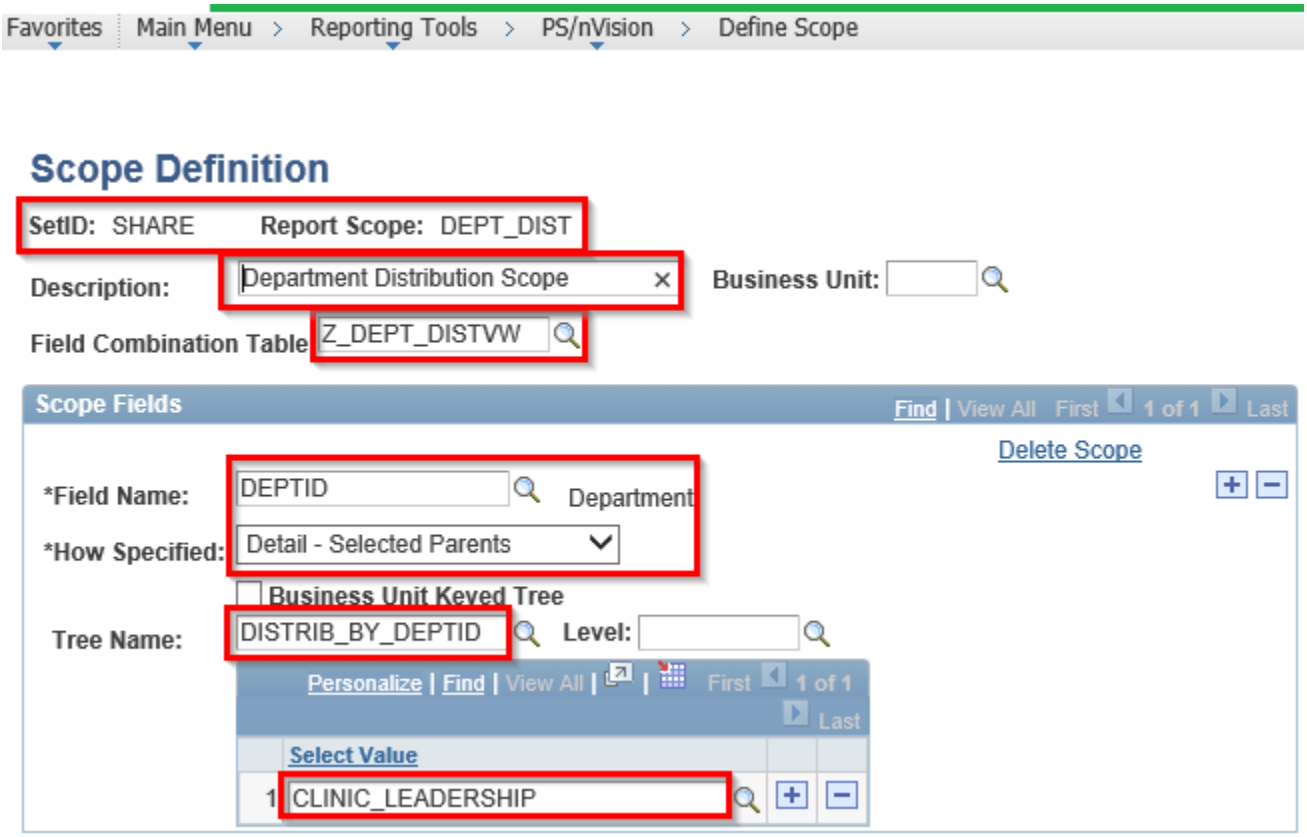
If you need a single department to have its own email distribution, you will need to have a Tree Node specifically for the single Department. It might look redundant to have a Tree Node with a single Department under the Tree Node, but the Tree Node's purpose is to hold the distribution email information, and the Tree Leaf's purpose is to point to the Department.

New Scope

Create a new nVision Scope.

This will allow the nVision Report Request to know which fieldname (DEPTID) to use as the filter, and which tree to look for the details.

With the following setup, it instructs the nVision Report Request to go to the Tree DISTRIB_BY_DEPTID, for Node CLINIC_LEADERSHIP. It should then look for any Tree Leafs (known as the Detail) that fall under this Selected Parent Tree Node.



We don't worry that our tree has several layers. We are using the Scope to find all departments that should be included.

The intent of this scope is two-fold:

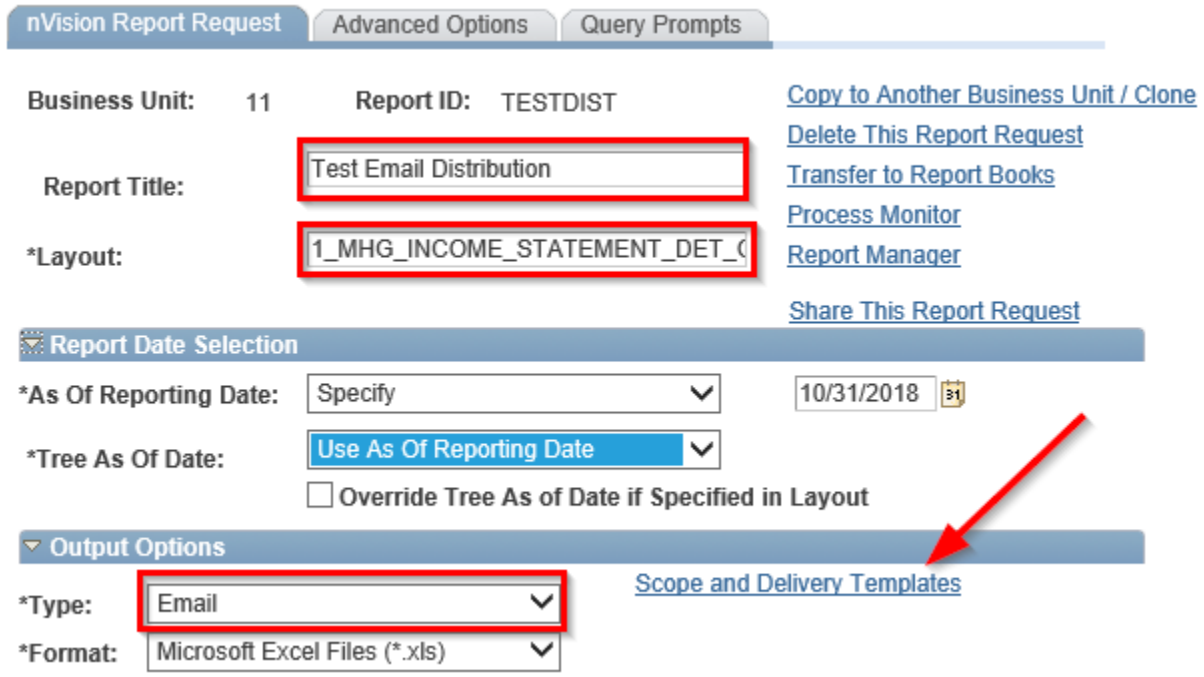
- Find all Departments (Tree Leafs) that fall under the CLINIC_LEADERSHIP node.
- For each department found, build an instance/nVision report.
  - Use the fieldname DEPTID in the Where clause, for that instance/nVision report.

Report Request Setup

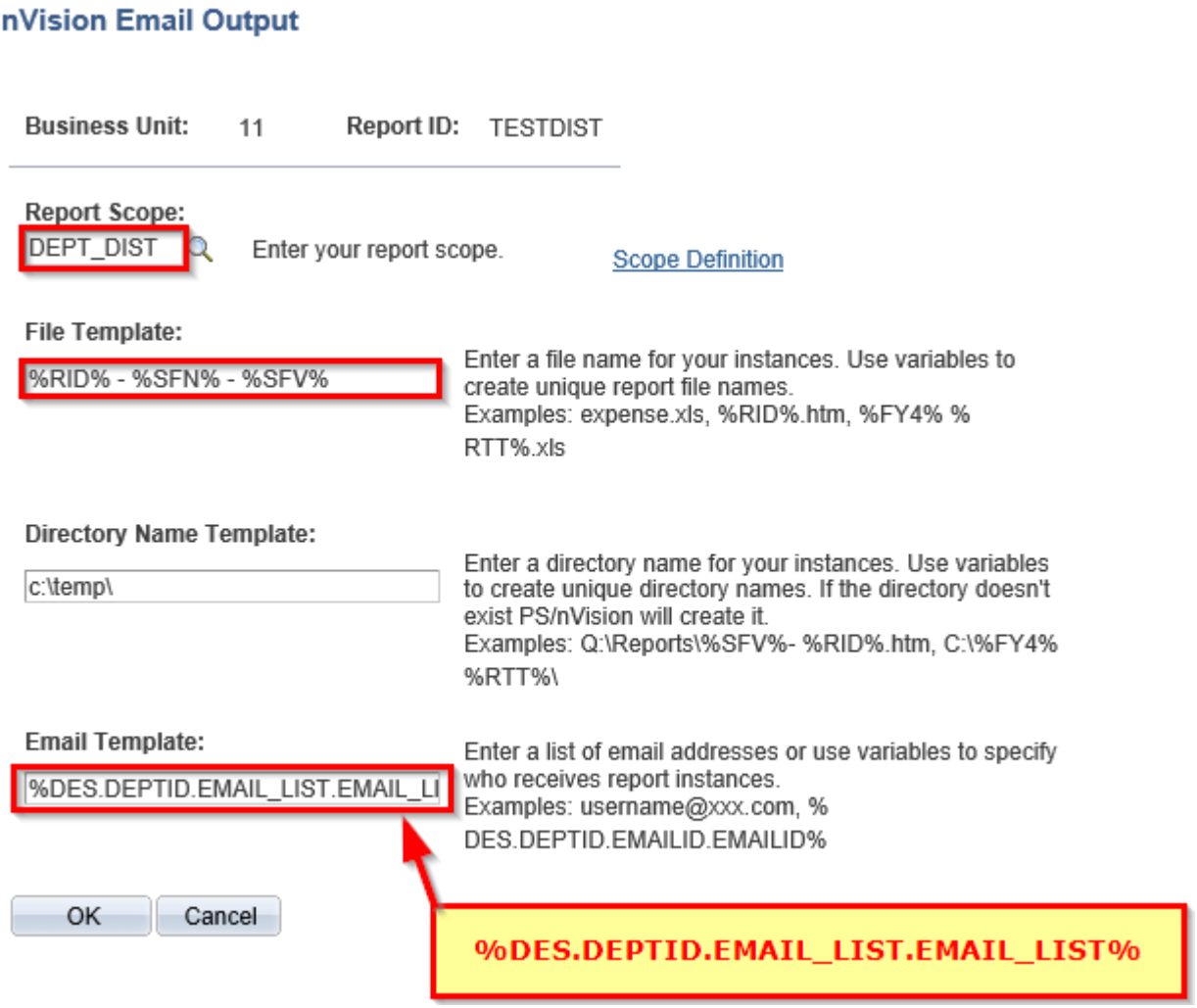
We are now ready to run our nVision Report Request.

You can use an existing nVision Report Request, for this test. You can also clone the Request.

Change the output type to Email.



Under Scope and Delivery Templates, use the following:



For the email template, use: %DES.DEPTID.EMAIL_LIST.EMAIL_LIST%

Before you run the report, be sure to click Save. The “Run Report” pushbutton does not force a Save on the page.

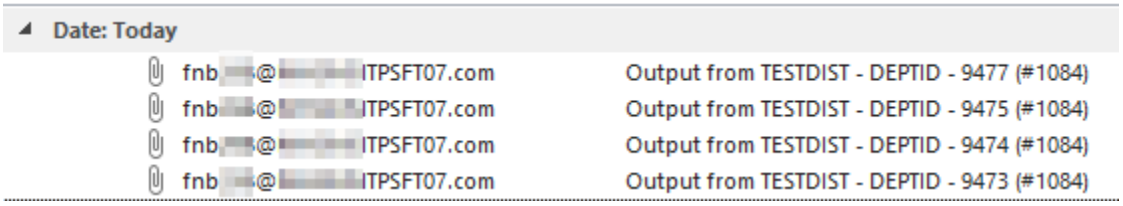
By using “%DES.DEPTID.EMAIL_LIST.EMAIL_LIST%” for the Email Template, this instructs the nVision Report to use the EMAIL_LIST field, found on the Scope’s Detail record.

In this Scope, it is using a Tree. That Tree uses a Tree Structure, which leads us to the Details tab on the Tree Structure, where we defined the Record Z_DEPT_DISTVW. On that table, we have a field called EMAILID_LIST, which holds a list of the emails to send to.

Lastly, run the report.

Results

If everything was configured correctly, you should have emails being distributed by Deptid, for the email distributions you defined.



Final Thoughts

Please review what is described above, test it thoroughly, before using in Production. David Vandiver assumes no responsibility on how this code is used in your environment. This information is provided to assist in whatever way you deem necessary.

If you have updates or additional ways to perform these tasks, feel free to share at David@VandiverHouse.com.

For free tips and code, check out David's website at: www.PeopleSoftTricks.com