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.

٢

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 <u>https://docs.oracle.com/cd/E80738_01/pt854pbh2/eng/pt/tnvs/task_CreatingScopeDefinitions-</u> <u>074ec7.html</u>

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.)

đ	at Z_DEPT_DISTVW2 (Record)														
ſ	Record Fields Record Type														
Г	Nur	n	Field Name	Туре	Кеу	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_N	IAME	Char				0	No	No	No		No	No	
	4	EFFDT		Date					No	No	No		No	No	
	5	EMAILID)	Char				0	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 TI
  , 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.

at Z_DEPT_DISTVW (Record)													
Record Fields Record Type													
Num	Field Name	Туре	Key	Ordr	Dir	CurC Src	1 List	Sys	Audt	InAu	EnAuto	Default	
1 SETI)	Char	Key	1	Asc	Yes	Yes	No		No	No		
2 TREE	_NAME	Char	Key	2	Asc	Yes	Yes	No		No	No		
3 EFFD	Т	Date	Key	3	Asc	Yes	Yes	No		No	No		
4 DEPT	'ID	Char	Key	4	Asc	Yes	Yes	No		No	No		
5 DEPT	ID_DESCR	Char	Alt		Asc	No	Yes	No	0	No	No		
6 EMAI	L_LIST	Long			ļ	No	No	No		No	No		
	Z_DEPT_DIS Record Fields 1 SETION 2 TREE 3 EFFD 4 DEPT 5 DEPT 6 EMAIN	Z_DEPT_DISTVW (Record) Record Type Num Field Name 1 SETID 2 TREE_NAME 3 EFFDT 4 DEPTID_DESCR 5 DEPTID_DESCR 6 EMAIL_LIST	Z_DEPT_DISTVW (Record) Record Type Num Field Name Type 1 SETID Char 2 TREE_NAME Char 3 EFFDT Date 4 DEPTID_DESCR Char 5 DEPTID_DESCR Char	Num Field Name Type Num Field Name Key 1 SETID Char Key 2 TREE_NAME Char Key 3 EFFDT Date Key 4 DEPTID_DESCR Char Alt 5 DEPTID_DESCR Char Alt 6 EMAIL_LIST Long Char	Secord Type Record Type Num Field Name Type Char Key 1 1 SETID Char Key 1 2 TREE_NAME Char Key 3 3 EFFDT Date Key 3 4 DEPTID_DESCR Char Alt 4 5 DEPTID_DESCR Char Alt 4 6 EMAIL_LIST Long 4 4	Secord Type Record Type Num Field Name Type Key Ord Dir 1 SETID Char Key 1 Asc 2 TREE_NAME Char Key 2 Asc 3 EFFDT Date Key 3 Asc 4 DEPTID_DESCR Char Alt Asc 5 DEPTID_DESCR Char Alt Asc 6 EMAIL_LIST Long	Second Type Record Type Num Field Name Type Key Ord Dir CurC SrcH 1 SETID Char Key 1 Asc Yes 2 TREE_NAME Char Key 2 Asc Yes 3 EFFDT Date Key 3 Asc Yes 4 DEPTID_DESCR Char Alt Asc No 5 DEPTID_DESCR Char Alt Asc No 6 EMAIL_LIST Long Vers No No	Second Type Num Field Name Type Char Char Ordr Dir Char Char <th col<="" th=""><th>Second Type Num Field Name Type Key Ordr Dir CurC Src List Sys 1 SETID Char Key 1 Asc Yes Yes No 2 TREE_NAME Char Key 2 Asc Yes Yes No 3 EFFDT Date Key 3 Asc Yes Yes No 4 DEPTID_DESCR Char Key 4 Asc Yes No 5 DEPTID_DESCR Char Alt Asc No Yes No 6 EMAIL_LIST Long Ver No No No No</th><th>Second Type Record Type Num Field Name Type Key Ordr Dir CurC Srch List Sys Audt 1 SETID Char Key 1 Asc Yes Yes No 2 TREE_NAME Char Key 2 Asc Yes Yes No 3 EFFDT Date Key 3 Asc Yes Yes No 4 DEPTID_DESCR Char Key 4 Asc Yes No No 5 DEPTID_DESCR Char Alt Asc No Yes No 6 EMAIL_LIST Long Ves No No No No</th><th>Z_DEPT_DISTVW (Record) Record Type Num Field Name Type Key Ordr Dir CurC Srch List Sys Audt InAu 1 SETID Char Key 1 Asc Yes Yes No No 2 TREE_NAME Char Key 2 Asc Yes Yes No No 3 EFFDT Date Key 3 Asc Yes Yes No No 4 DEPTID_DESCR Char Key 4 Asc Yes No No 5 DEPTID_DESCR Char Alt Asc No Yes No No 6 EMAIL_LIST Long Ves No No No No</th><th>Z_DEPT_DISTVW (Record) Record Type Num Field Name Type Key Ordr Dir CurC Set Sys Audt InAu EnAuto 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_DESCR Char Alt Asc Yes No No No No 5 DEPTID_DESCR Char Alt Asc No No No No 6 EMAIL_LIST Long Ves No No No No No No</th></th>	<th>Second Type Num Field Name Type Key Ordr Dir CurC Src List Sys 1 SETID Char Key 1 Asc Yes Yes No 2 TREE_NAME Char Key 2 Asc Yes Yes No 3 EFFDT Date Key 3 Asc Yes Yes No 4 DEPTID_DESCR Char Key 4 Asc Yes No 5 DEPTID_DESCR Char Alt Asc No Yes No 6 EMAIL_LIST Long Ver No No No No</th> <th>Second Type Record Type Num Field Name Type Key Ordr Dir CurC Srch List Sys Audt 1 SETID Char Key 1 Asc Yes Yes No 2 TREE_NAME Char Key 2 Asc Yes Yes No 3 EFFDT Date Key 3 Asc Yes Yes No 4 DEPTID_DESCR Char Key 4 Asc Yes No No 5 DEPTID_DESCR Char Alt Asc No Yes No 6 EMAIL_LIST Long Ves No No No No</th> <th>Z_DEPT_DISTVW (Record) Record Type Num Field Name Type Key Ordr Dir CurC Srch List Sys Audt InAu 1 SETID Char Key 1 Asc Yes Yes No No 2 TREE_NAME Char Key 2 Asc Yes Yes No No 3 EFFDT Date Key 3 Asc Yes Yes No No 4 DEPTID_DESCR Char Key 4 Asc Yes No No 5 DEPTID_DESCR Char Alt Asc No Yes No No 6 EMAIL_LIST Long Ves No No No No</th> <th>Z_DEPT_DISTVW (Record) Record Type Num Field Name Type Key Ordr Dir CurC Set Sys Audt InAu EnAuto 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_DESCR Char Alt Asc Yes No No No No 5 DEPTID_DESCR Char Alt Asc No No No No 6 EMAIL_LIST Long Ves No No No No No No</th>	Second Type Num Field Name Type Key Ordr Dir CurC Src List Sys 1 SETID Char Key 1 Asc Yes Yes No 2 TREE_NAME Char Key 2 Asc Yes Yes No 3 EFFDT Date Key 3 Asc Yes Yes No 4 DEPTID_DESCR Char Key 4 Asc Yes No 5 DEPTID_DESCR Char Alt Asc No Yes No 6 EMAIL_LIST Long Ver No No No No	Second Type Record Type Num Field Name Type Key Ordr Dir CurC Srch List Sys Audt 1 SETID Char Key 1 Asc Yes Yes No 2 TREE_NAME Char Key 2 Asc Yes Yes No 3 EFFDT Date Key 3 Asc Yes Yes No 4 DEPTID_DESCR Char Key 4 Asc Yes No No 5 DEPTID_DESCR Char Alt Asc No Yes No 6 EMAIL_LIST Long Ves No No No No	Z_DEPT_DISTVW (Record) Record Type Num Field Name Type Key Ordr Dir CurC Srch List Sys Audt InAu 1 SETID Char Key 1 Asc Yes Yes No No 2 TREE_NAME Char Key 2 Asc Yes Yes No No 3 EFFDT Date Key 3 Asc Yes Yes No No 4 DEPTID_DESCR Char Key 4 Asc Yes No No 5 DEPTID_DESCR Char Alt Asc No Yes No No 6 EMAIL_LIST Long Ves No No No No	Z_DEPT_DISTVW (Record) Record Type Num Field Name Type Key Ordr Dir CurC Set Sys Audt InAu EnAuto 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_DESCR Char Alt Asc Yes No No No No 5 DEPTID_DESCR Char Alt Asc No No No No 6 EMAIL_LIST Long Ves No No No No No No

Record Type: View

(Use Build Sequence No 2)

📸 Z_DEPT_DIST	VW (Record)	
Record Fields	Record Type]
Record Type C SQL Tab SQL Viet C Dynamic C Derived/ C SubReco C Query Vi C Tempora	le View Work ord ew ry 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 <u>any</u> 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 \rightarrow Tree Structure

Add: DEPT_NODE_DISTRIB

irst Tab: Structure							
Structure Levels Nodes Details							
Tree Structure Properties							
Structure ID:	DEPT_NODE_DISTRIB						
*Description:	Dept Node Email Distribution]				
*Type:	Detail	•					
Additional Key Fi	eld		Navigation Options				
SetId Indi	SetId Indirection		Node Multi-Navigation Detail Multi-Navigation				
User Defined							
None							

Second Tab: Levels

Structure Levels	Nodes Details						
Tree Levels							
Structure ID:	DEPT_NODE_DISTRIB						
Record Name:	TREE_LEVEL_TBL						
Page Name:	TREE_LEVEL						
Component Name:							
Menu Name:		Q					
Menu Bar Name:		Q					
Menu Item Name:		Q					
Third Tab: Nodes							
Structure Levels Nodes Details							

Tree Nodes

Structure ID:	DEPT_NODE_DISTRIB
*Record Name:	
*Field Name:	TREE_NODE
*Page Name:	DEPT_NODE_TBL
Component Name:	DEPT_NODE

Menu Name:

Menu Bar Name:

Menu Item Name:

DEPT_NODE_IBL	
DEPT_NODE	<u> </u>
TREEMANAGER	٩
USE	٩
DEPTNODE	Q

Fourth Tab: Details

Structure Levels Nodes Details						
Tree Details						
Structure ID:	DEPT_NODE_DISTRIB					
Record Name: Z_DEPT_DISTVW						
Field Name:						
Page Name:	DEPARTMENT					
Page Name: Component Name:	DEPARTMENT	2				
Page Name: Component Name: Menu Name:	DEPARTMENT	2				
Page Name: Component Name: Menu Name: Menu Bar Name:	DEPARTMENT	2				



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:	Tree Name: DISTRIB_BY_DEPTID						
*Structure ID:	DEPT_NODE_DIS	TRIB					
*Effective Date:	01/01/1901 *S	tatus: Active	\checkmark				
*Description:	Email Distrib By D	eptid					
*Category:	DEFAULT	Q					
*Use of Levels:	Level Not Used	✓ Pe	erformance Options				
*SetID:	SHARE						
Audits		Item Counts					
All Detail Va	lues in this Tree	Node Count:	4				
Allow Duplic	ate Detail Values	Leaf Count:	4				
		Level Count:	0				
Perform Audits		Branch Count:	0				

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



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.

Dept Node Tbl					
SetID: SHARE Tree Node: CLINCIS_EASTSIDE				6	
*Effective Date: 01/01/1901 🔅 *Status: Active 🗸					
*Description: Clinics on East Side					
Routing Information	Pe	rsonalize Find View All 🗖 🛗	First 🚺 1-3 of	3 🕨	Last
Email ID	ID Type	Distribution ID			
1 Clinic_Manager_Area1@trash.com	User	▼	Q	+	-
2	User	✓ C023014	Q	+	-
3	Role	Security Administrator	্	+	-

Here is a summary of which email addresses were placed for each Node. Notice we made a higher node configured for the email address <u>Clinic Leader@trash.com</u>. 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.

Favorites Main Mer	nu > Reporting Tools > PS/nVision >	Define Scope		
Scope Defi	nition			
SetID: SHARE	Report Scope: DEPT_DIST			
Description:	Department Distribution Scope ×	Business Unit:	Q	
Field Combination	n Table Z_DEPT_DISTVW Q			
Scope Fields			Find View All First 🚺 1 of 1	Last
			Delete Scope	
*Field Name:	DEPTID Q Department			+ -
*How Specified:	Detail - Selected Parents 🗸 🗸			
	Business Unit Keved Tree			
Tree Name:	DISTRIB_BY_DEPTID	Q		
	Personalize Find View All 🗖 🛗	First 1 of 1		
	Select Value			
	1 CLINIC_LEADERSHIP	Q 🛨 🖃		

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.

nVision Report Request	Advanced Options Query Prompts						
Business Unit: 11	Report ID: TESTDIST	Copy to Another Business Unit / Clone Delete This Report Request					
Report Title:	Test Email Distribution	Transfer to Report Books					
		Process Monitor					
*Layout:	1_MHG_INCOME_STATEMENT_DET_(Report Manager					
		Share This Report Request					
Report Date Selection							
*As Of Reporting Date:	Specify V	10/31/2018					
*Tree As Of Date:	Use As Of Reporting Date						
Override Tree As of Date if Specified in Layout							
Output Options							
*Type: Email	✓ Scope and D	elivery Templates					
Format: Microsoft Excel Files (.xls)							

Under Scope and Delivery Templates, use the following:

nVision Email Output

Business Unit: 11 Report ID	: TESTDIST	
Report Scope: DEPT_DIST Q Enter your report s	cope. <u>Scope Definition</u>	
File Template: %RID% - %SFN% - %SFV%	Enter a file name for your instances. Use variables to create unique report file names. Examples: expense.xls, %RID%.htm, %FY4% % RTT%.xls	
Directory Name Template:	Enter a directory name for your instances. Use variables to create unique directory names. If the directory doesn't exist PS/nVision will create it. Examples: Q:\Reports\%SFV%- %RID%.htm, C:\%FY4% %RTT%\	
Email Template: %DES.DEPTID.EMAIL_LIST.EMAIL_LI	Enter a list of email addresses or use variables to specify who receives report instances. Examples: username@xxx.com, % DES.DEPTID.EMAILID.EMAILID%	
OK Cancel	%DES.DEPTID.EMAIL_LIST.EMAIL_LIST%	

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.

4	Date: Today	/		
	U	fnb @	ITPSFT07.com	Output from TESTDIST - DEPTID - 9477 (#1084)
	U	fnb @	ITPSFT07.com	Output from TESTDIST - DEPTID - 9475 (#1084)
	U	fnb @	ITPSFT07.com	Output from TESTDIST - DEPTID - 9474 (#1084)
	U	fnb 🔤 🥘 🔤	ITPSFT07.com	Output from TESTDIST - DEPTID - 9473 (#1084)

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 <u>David@VandiverHouse.com</u>.

