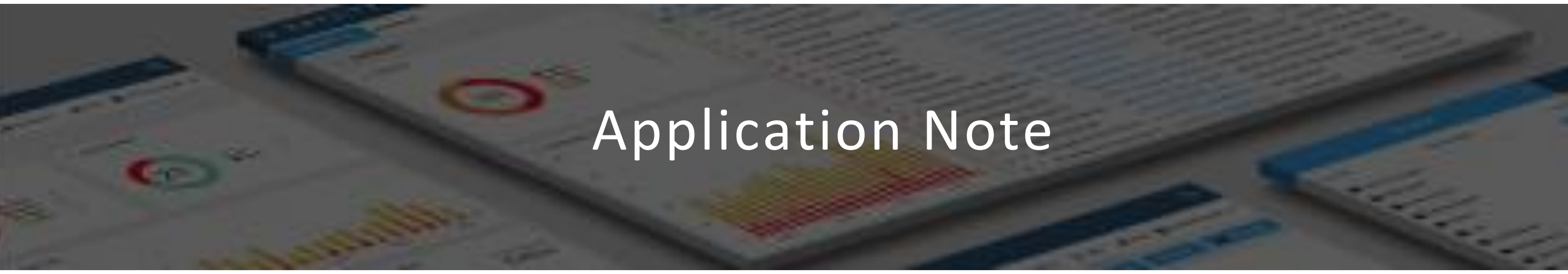
ENTUITY
NETWORK ANALYTICS


Application Note

# Application Note

# Creating a Multiple View Device Type Count Tabular Report using the Report Builder

27th March 2019

John Diamond

## Table of Contents

1 Overview ... 3

2 Challenges to be overcome ... 4

3 Creating tables where multiple types of iteration are required ... 4

4 Iterating over multiple views ... 4

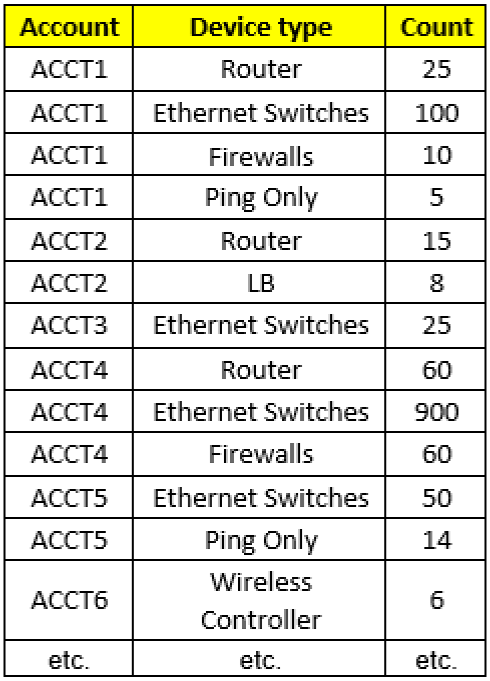
5 Displaying unique device types in a table ... 6

6 Report output ... 8

# 1 Overview

This Application Note describes an approach to creating a report using the Report Builder that will iterate through multiple views displaying tables populated with a list of the different device types, such as Router and Ethernet Switch, along with the number of each type in each view. The requirement is that the resulting tables have three columns; view name, device type and count.

This is a sketch of how an MSP had requested the output from such a report should look:



| Account | Device type | Count |
|---|---|---|
| ACCT1 | Router | 25 |
| ACCT1 | Ethernet Switches | 100 |
| ACCT1 | Firewalls | 10 |
| ACCT1 | Ping Only | 5 |
| ACCT2 | Router | 15 |
| ACCT2 | LB | 8 |
| ACCT3 | Ethernet Switches | 25 |
| ACCT4 | Router | 60 |
| ACCT4 | Ethernet Switches | 900 |
| ACCT4 | Firewalls | 60 |
| ACCT5 | Ethernet Switches | 50 |
| ACCT5 | Ping Only | 14 |
| ACCT6 | Wireless Controller | 6 |
| etc. | etc. | etc. |

Each customer of the MSP is assumed to have their own view in which the devices are contained.

There was no requirement to aggregate view contents across multiple remote servers.

It should be noted that the finished report was created using Entuity Network Analytics 17.0 and that it would not have been possible using version 16.5 or earlier as the iteration over multiple views would not have been possible using the Report Builder.

## 2 Challenges to be overcome

i) A single table cannot iterate over more than one component type so, in this case, it cannot iterate over different views and different device types at the same time.

ii) The Report Builder assumes that a single view will be selected and that all displayed metrics will be derived from components within that view. Iterating over multiple views is not a standard mode of operation.

iii) Each row of a table is assumed to correspond to an instance of a selected component such as a device, port, module, etc. Different device types don’t qualify as examples of instances of the same component so this concept can’t be leveraged in its ordinary manner.

iv) If a sub-report of a composite report is to be used, then there is no standard way for the expressions used to build custom attributes to obtain the name of the view being used to constrain that report.

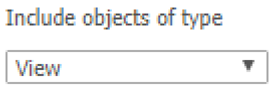
## 3 Creating tables where multiple types of iteration are required

The chosen approach to separating the iteration through the views and the unique device types within each view in such a way that all the relevant device types for each view are guaranteed to be included was to use a composite report. The top-level composite report would iterate through the views and, for each view, display a sub-report that would display a table with one row per unique device type within that view. This means that each table only performs one type of iteration.

The downside of this approach is that it results in one table per view rather than one table containing all the unique device type counts for all views. This limitation was considered acceptable.

## 4 Iterating over multiple views

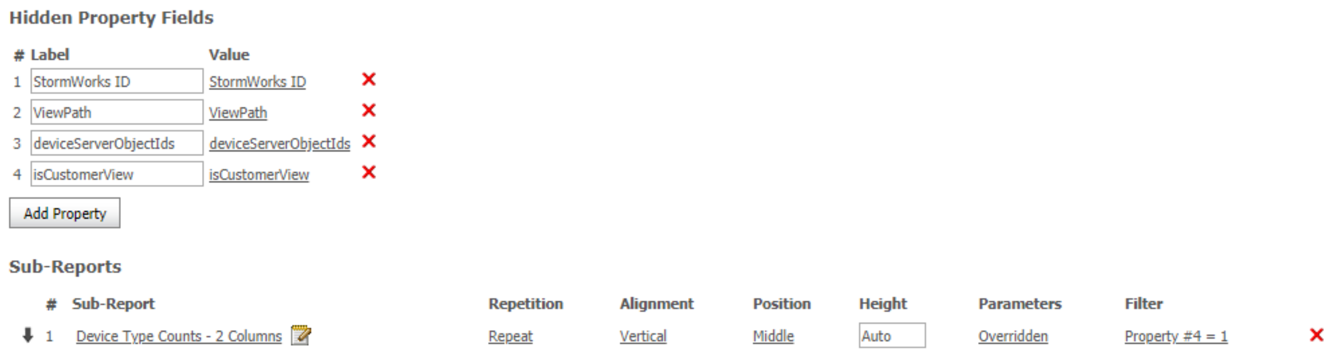
The way in which each report iterates is specified by the “Include objects of type” menu setting:



Include objects of type

View

Where a report is required to iterated over multiple views the object type should be set to “View” which has to be selected using the “Other Types…” menu option as it isn’t one of the handful of popular types listing in the menu by default. The “My Network” view contains all the other views so, by selecting “My Network” the composite report will iterate over all the sub-views. The list of sub-views includes all the My Network views for all the users so user “fred” would have a My Network view called “fred::My Network”. These user-specific views will need to be eliminated from the list. Every sub-view of every other sub-view will also be listed and will need to be eliminated. Finally, the “All Objects” special view should be eliminated as it doesn’t represent a customer’s network. Row filtering within a composite report can be achieved using the Filter setting which is visible when the Advanced Options mode is enabled. This allows one of the Hidden Property Fields to be tested for equality to a single value. In this case, the logic to determine whether the current view is one that should be used by a sub-report is implemented in the Hidden Property Field called "isCustomerView" which returns 1 if the view is to be used an 0 otherwise:



**Hidden Property Fields**

| # | Label | Value |
|---|---|---|
| 1 | StormWorks ID | StormWorks ID |
| 2 | ViewPath | ViewPath |
| 3 | deviceServerObjectIds | deviceServerObjectIds |
| 4 | isCustomerView | isCustomerView |

**Sub-Reports**

| # | Sub-Report | Repetition | Alignment | Position | Height | Parameters | Filter |
|---|---|---|---|---|---|---|---|
| 1 | Device Type Counts - 2 Columns | Repeat | Vertical | Middle | Auto | Overridden | Property #4 = 1 |

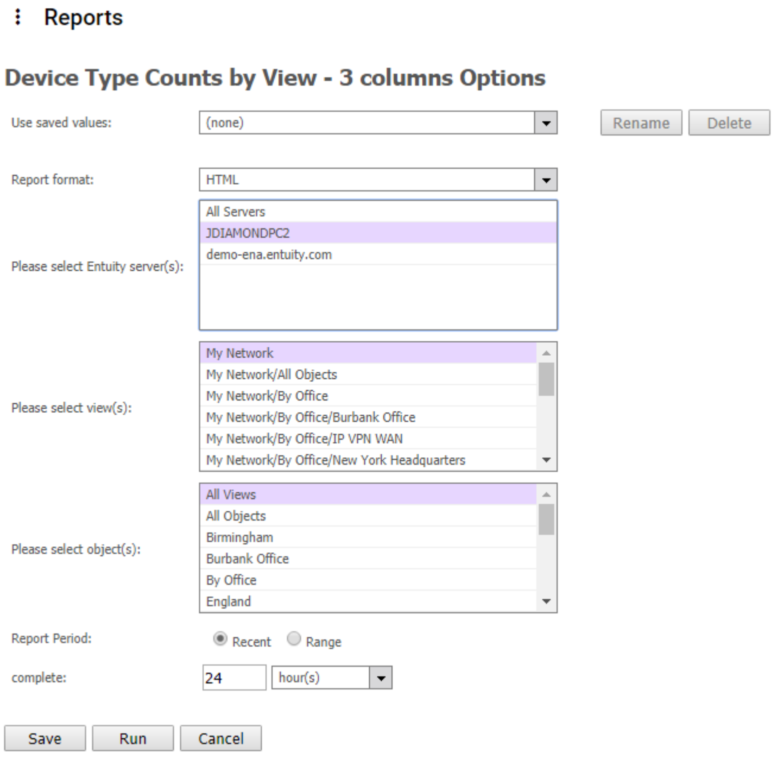
The statement language syntax for the isCustomerView Hidden Property Field is:

(regex("/",viewPath)[1] == -1) && (regex("::",viewPath)[1] == -1) && (viewPath != "All Objects")

This eliminates:

- Views with "/" in the path which covers sub-views
- views with "::" in the path which covers the "My Network" views individually owned by all the users
- the "All Objects" special view

When the report is launched, a specific server should be selected if running in multi-server mode and the default view selection (My Network and All Views) should be adopted:



**Device Type Counts by View - 3 columns Options**

| | |
|---|---|
| Use saved values: | (none) |
| Report format: | HTML |
| Please select Entuity server(s): | All Servers<br>JDIAMONDPC2 (selected)<br>demo-ena.entuity.com |
| Please select view(s): | My Network (selected)<br>My Network/All Objects<br>My Network/By Office<br>My Network/By Office/Burbank Office<br>My Network/By Office/IP VPN WAN<br>My Network/By Office/New York Headquarters |
| Please select object(s): | All Views (selected)<br>All Objects<br>Birmingham<br>Burbank Office<br>By Office<br>England |
| Report Period: | ● Recent ○ Range |
| complete: | 24 hour(s) |

# 5 Displaying unique device types in a table

The table displayed for a specific view needs to contain one instance of each unique device type. This can be achieved by designing the sub-report that will display the table to include every device in the view then add a custom field into the composite report to select only one instance of each device type to be included. This selection logic can be performed in a Hidden Property Field in the composite report. With the Advanced Options mode selected, create a new Hidden Property Field using the Add Property button. Then use the Define Attribute button in the Static Attributes Selection dialog enter “deviceServerObjectIds” as its Description. The Expression syntax is:

```
variable vid=viewId;variable deviceTypes=unique(foreach(objects_in_view(vid, OBJ_TYPE_DEVICE,
true),BasicDevice(this).devType));variable devices=foreach(deviceTypes,variable
dType=this;head(foreach(objects_in_view(vid, OBJ_TYPE_DEVICE,
true),this,BasicDevice(this).devType==dType)));server_object_id_string(devices)
```

The logic of this expression is better understood when shown as follows:

```
variable vid=viewId;

variable deviceTypes=unique(foreach(objects_in_view(vid, OBJ_TYPE_DEVICE,
true),BasicDevice(this).devType));

variable devices=foreach(deviceTypes,variable dType=this;head(foreach(objects_in_view(vid,
OBJ_TYPE_DEVICE, true),this,BasicDevice(this).devType==dType)));

server_object_id_string(devices)
```

The composite report is iterating through multiple views so the context of the sub-report is a view. The ID of the current view is obtained from the viewId attribute of the view and saved in a variable called “vid”. This is to allow it to be used later on where the current context is no longer the view.

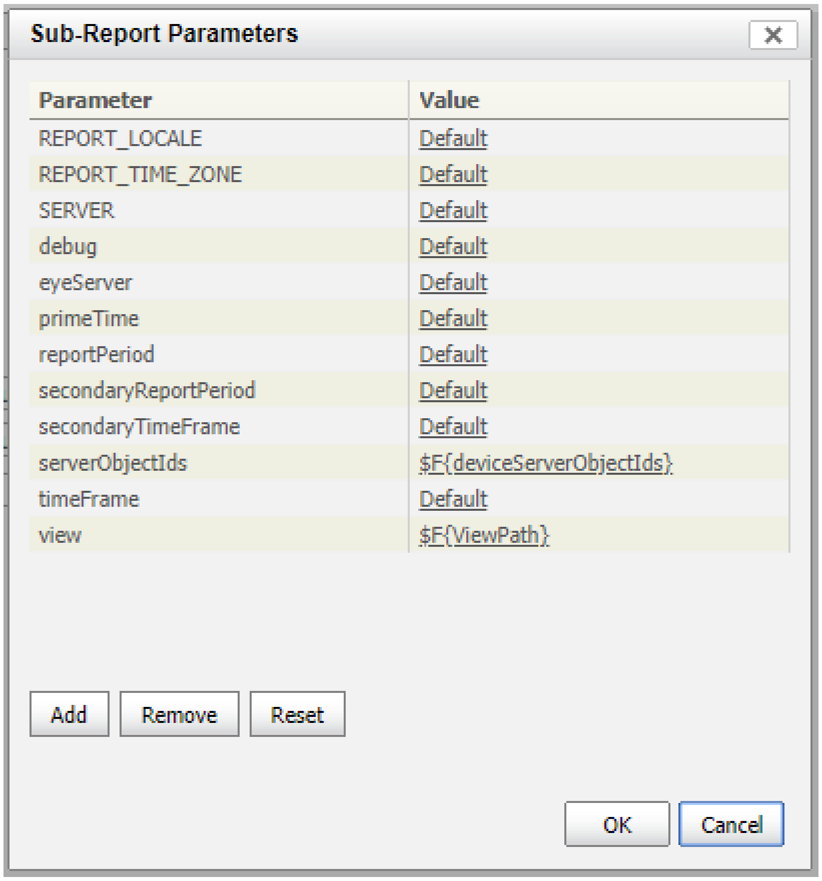
A list of unique device types in the view is then computed and saved in a variable called “deviceTypes”.

One example of each of the unique device types is then selected and placed into a list that is held in a variable called “devices”.

Finally, The list of devices is converted into a list of serverId:objectId pairs which is the format used to pass such a list to a sub-report.

The view to be used by the sub-report needs to be passed to it as a parameter to override the default behavior of simply using the same view that the composite report is running under which would be “My Network” in this case. A Hidden Property Field should be added and the “View Path” attribute located using the Find Attributes button. The label for this Hidden Property Field will default to being the description of the attribute “View Path” which contains a space. As spaces are not allowed in the label the description should be compressed to “ViewPath”.

The Parameters setting for the sub-report will initially be set to “Default” and should be clicked to allow the view and serverObjectIds parameters to be overridden:



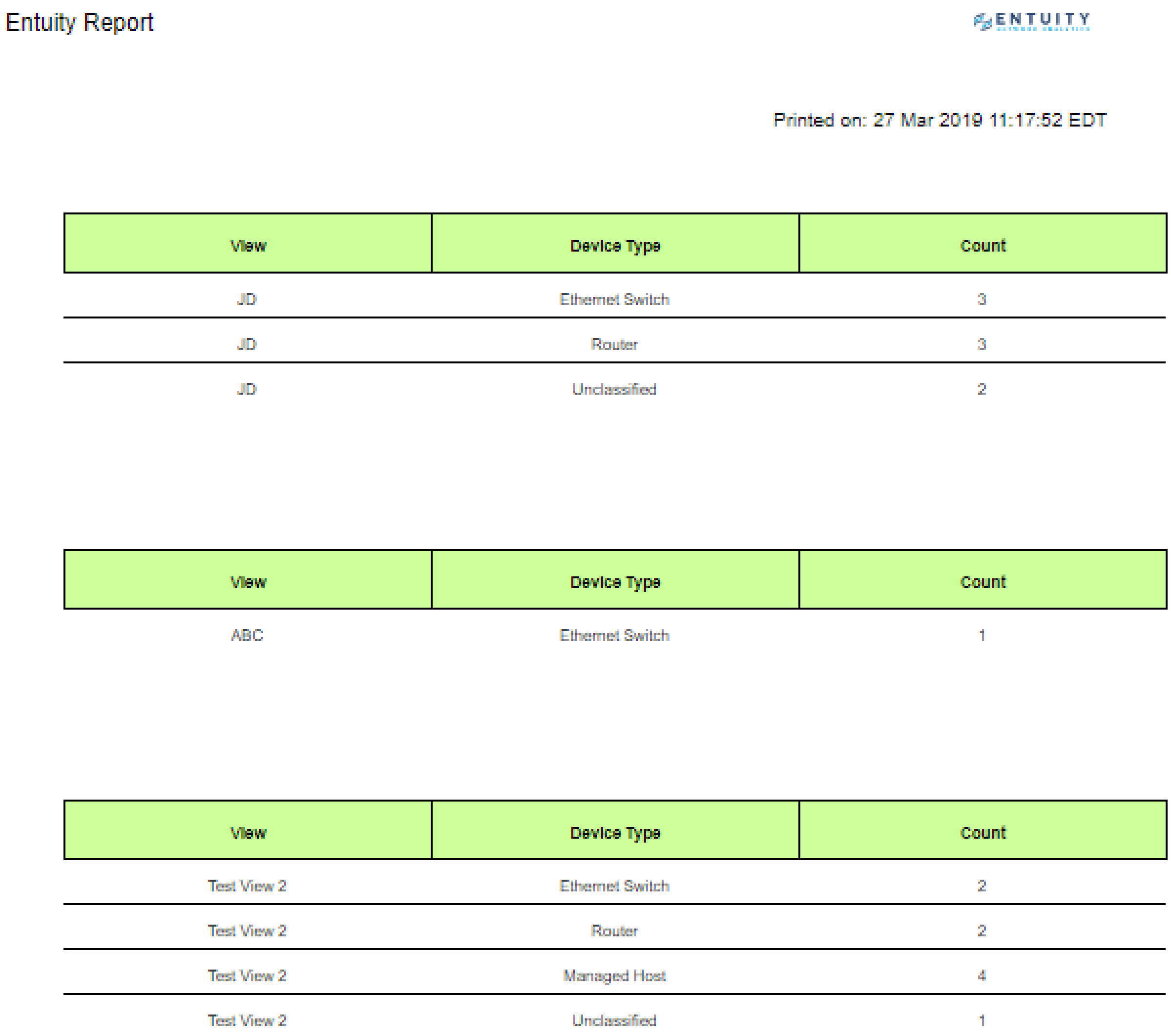
Sub-Report Parameters

| Parameter | Value |
|---|---|
| REPORT_LOCALE | Default |
| REPORT_TIME_ZONE | Default |
| SERVER | Default |
| debug | Default |
| eyeServer | Default |
| primeTime | Default |
| reportPeriod | Default |
| secondaryReportPeriod | Default |
| secondaryTimeFrame | Default |
| serverObjectIds | $F{deviceServerObjectIds} |
| timeFrame | Default |
| view | $F{ViewPath} |

The “deviceServerObjectIds” Hidden Property Field should be used to override the “serverObjectIds” Parameter using the Field setting. The “ViewPath” Hidden Property Field should be used to override the “view” Parameter using the Field setting.

## 6 Report output

This is how a report for a server with three custom views is displayed:



Entuity Report

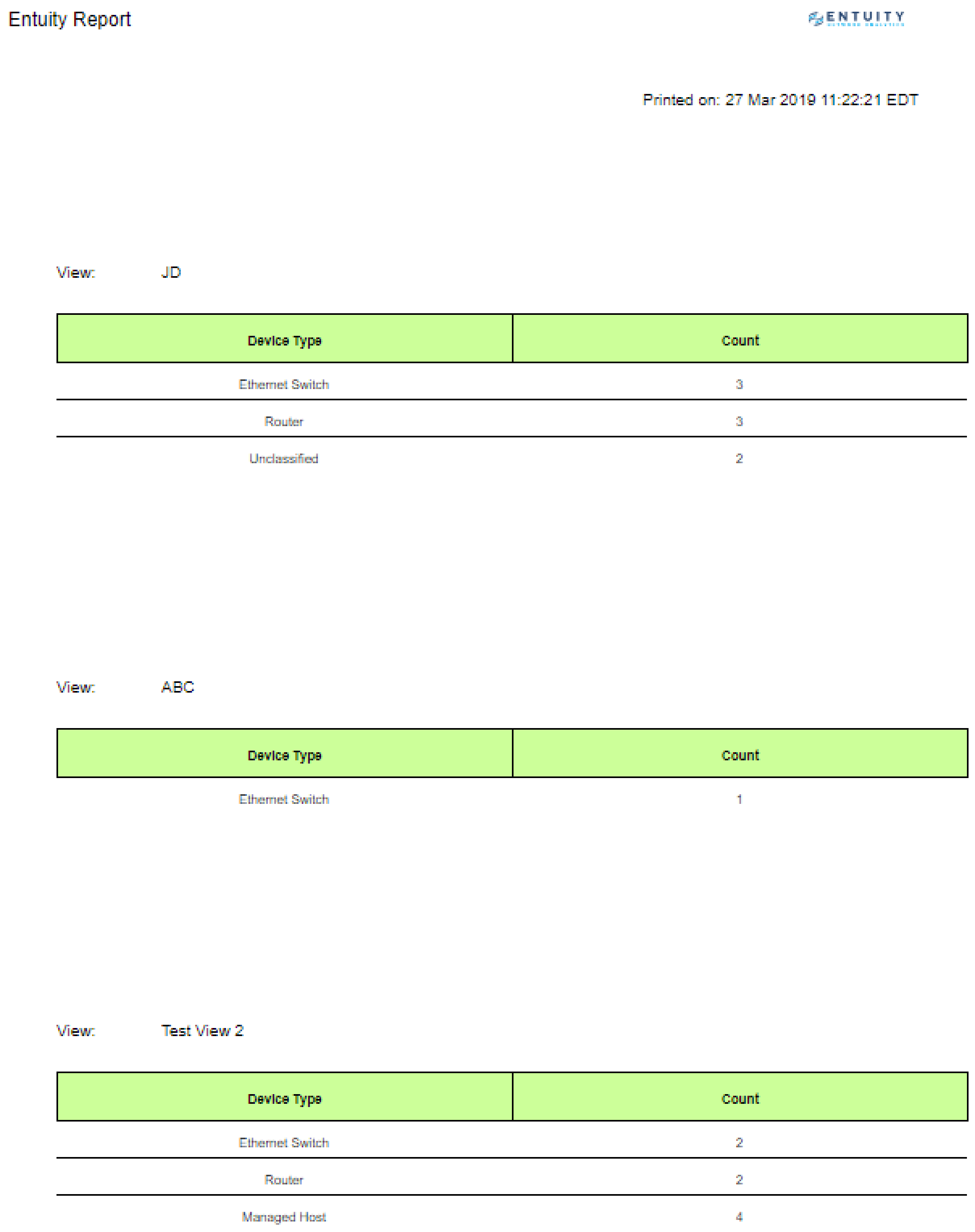
Printed on: 27 Mar 2019 11:17:52 EDT

| View | Device Type | Count |
|---|---|---|
| JD | Ethernet Switch | 3 |
| JD | Router | 3 |
| JD | Unclassified | 2 |

| View | Device Type | Count |
|---|---|---|
| ABC | Ethernet Switch | 1 |

| View | Device Type | Count |
|---|---|---|
| Test View 2 | Ethernet Switch | 2 |
| Test View 2 | Router | 2 |
| Test View 2 | Managed Host | 4 |
| Test View 2 | Unclassified | 1 |

An alternative approach to the report formatting would have been to remove the “View” column from the tables and identify the view in a heading at the top of each table:



Entuity Report

Printed on: 27 Mar 2019 11:22:21 EDT

View: JD

| Device Type | Count |
|---|---|
| Ethernet Switch | 3 |
| Router | 3 |
| Unclassified | 2 |

View: ABC

| Device Type | Count |
|---|---|
| Ethernet Switch | 1 |

View: Test View 2

| Device Type | Count |
|---|---|
| Ethernet Switch | 2 |
| Router | 2 |
| Managed Host | 4 |