Contents

[Demo Guide: Module 7 – Monitoring and Troubleshooting Containers 2](#_Toc491261531)

[Demo 1: ​ 2](#_Toc491261532)

[Monitor an Azure Container Service Kubernetes cluster with Microsoft Operations Management Suite (OMS) 2](#_Toc491261533)

Demo Guide: Module 7 – Monitoring and Troubleshooting Containers

Demo 1: ​ Troubleshoot the multi container application with kubectl

In this demo, you will use an existing Kubernetes cluster with the front-end and back-end application deployed (web app or web api). Then you are going to interact with the pods to understand if the application is running correctly.

In this scenario,

Tasks

1. Check the running web api pods
2. Type “kubectl get pods -o wide” to check if the pods are running and on which machine they are assigned
3. Type “kubectl logs <webapi\_pod>” to make sure that no error log is printed by the running pod
4. Type “kubectl describe pods <webapi\_pod>” and look at the relevant information regarding the health of the pod (e.g. last error code)
5. Check the connection between web app and web api pods
6. If you are in a **Linux cluster**
   1. Type “kubectl exec -it <webapp\_pod>” -- bash” to start a bash session inside the container running in the pod
   2. Type “curl http://demowebapi:9000/api/quotes -v” to check that we receive a valid response from the web api
   3. Type “curl http://demowebapi:9000/api/unknown -v” to show what we would receive in case of an invalid api call
7. If you are in a **Windows cluster**
   1. Type “kubectl exec -it <webapp\_pod>” -- Powershell” to start a Powershell session inside the container running in the pod
   2. Type “Test-NetConnection demowebapi -Port 9000” to show that we can access the port 9000 in the web api pod. If you have an error message, it is because the base OS is nanoserver and has limitation because of the Powershell version installed, but the result of this command is still meaningful
   3. Type “Invoke-WebRequest -Uri http://demowebapi:9000/api/quotes” to check that we receive a valid response from the web api
   4. Type “Invoke-WebRequest -Uri <http://demowebapi:9000/api/unknown>” to show what we would receive in case of an invalid api call

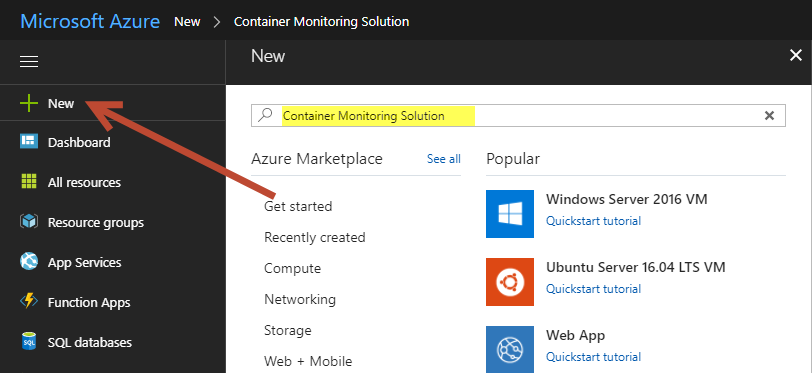
Demo 2: ​ Monitor an Azure Container Service Kubernetes cluster with Microsoft Operations Management Suite (OMS)

In this demo, you will deploy Microsoft Operations Management (OMS) agents to centrally monitor the performance of your containers in your Linux Kubernetes cluster.

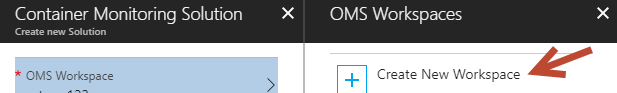
OMS is a Microsoft's cloud-based IT management solution that helps you manage and protect your on-premises and cloud infrastructure. Container Solution is a solution in OMS Log Analytics, which helps you view the container inventory, performance, and logs in a single location. You can audit, troubleshoot containers by viewing the logs in centralized location, and find noisy consuming excess container on a host.

Tasks

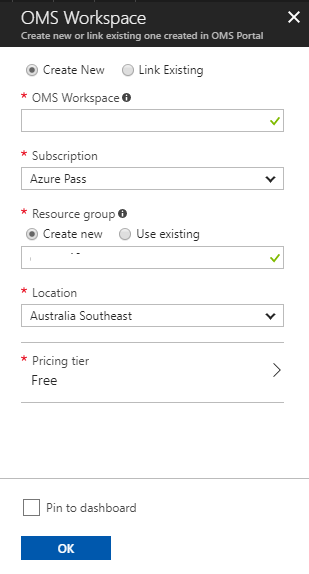
1. Create OMS Workspace
2. You need to complete steps #2-18 at least the night before so that OMS can populate with data, so you can do the demo. For the actual demo in the workshop, explain what you did ahead of time then show OMS starting with step #19.
3. Sign onto your LOD Windows VM and open Chrome on the VM. Navigate to portal.azure.com and sign in with your Azure Pass account.
4. Click + New. Search for Container Monitoring Solution



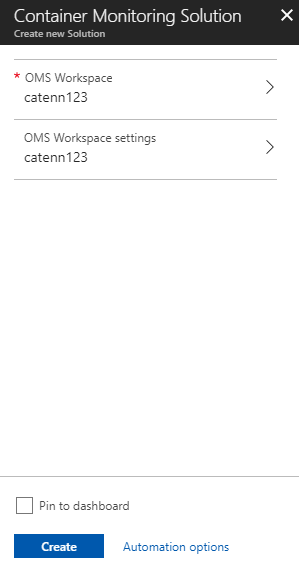
1. Click Create
2. Click OMS Workspace > Create a New Workspace



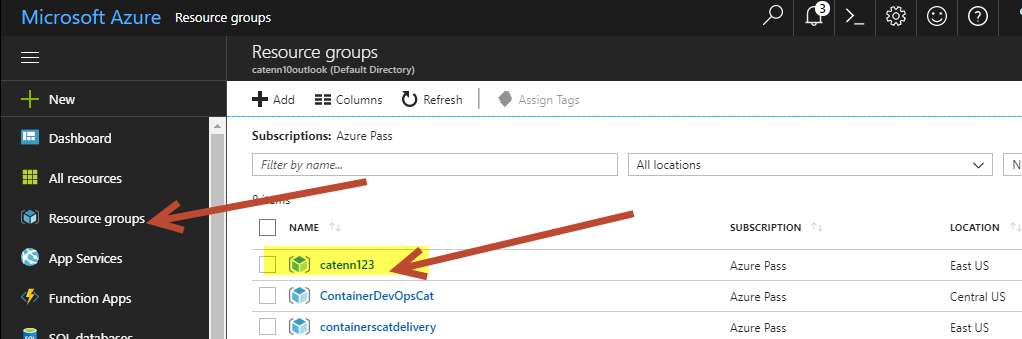
1. Fill in the boxes with any unique name for the OMS workspace and resource group you would like. Choose a location near to your physical location.



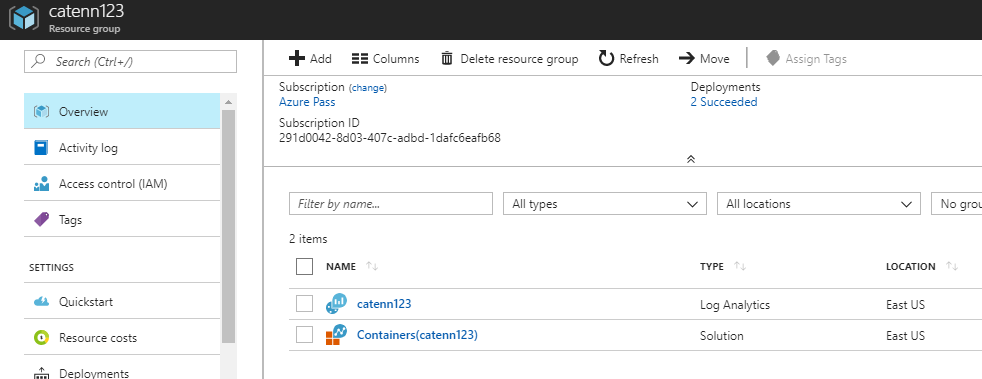
1. Hit OK.
2. Once you’ve chosen a workspace and the workspace settings, hit Create.



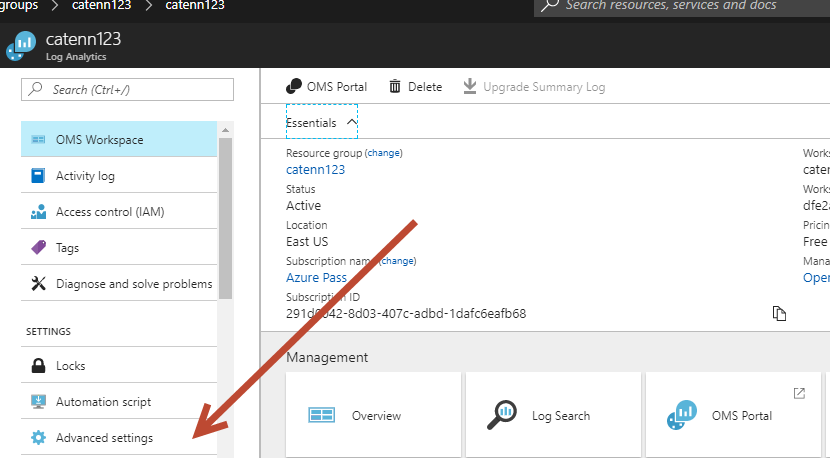
1. Wait a minute for it to deploy.
2. Click Resource Groups and click on your new OMS resource group.



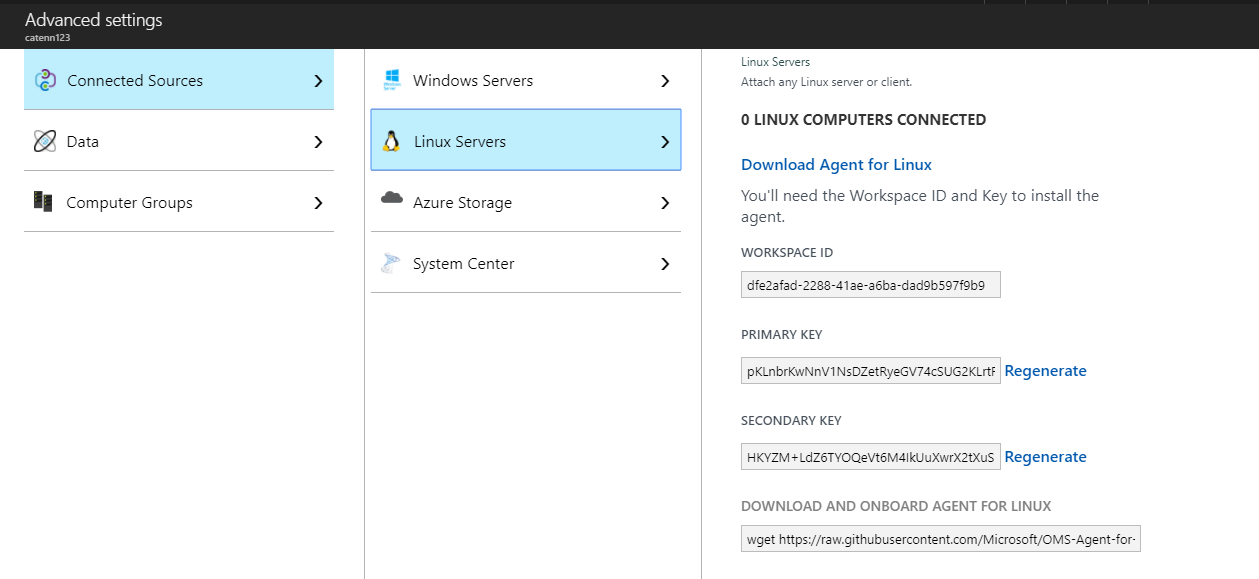
1. You should see two resources: Solution and Log Analytics:



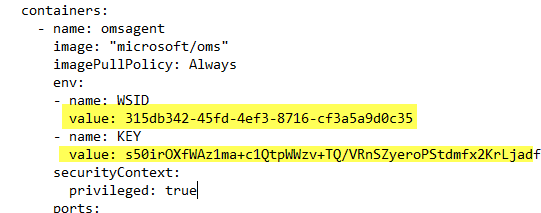
1. Click on the Log Analytics resource.
2. Click Advanced Settings:



1. Choose Connected Sources > Linux Servers > then you should see a Workspace ID and Primary key. Copy both of these into a Notepad.



1. Open **omsagent.yaml** file in the **C:\labs\module7** folder and replace **<WSID>** and **<KEY>** parameters with WORKSPACE ID and PRIMARY KEY values you have copied in the previous step, do not add quotes to these, just copy and paste directly as shown below:

.

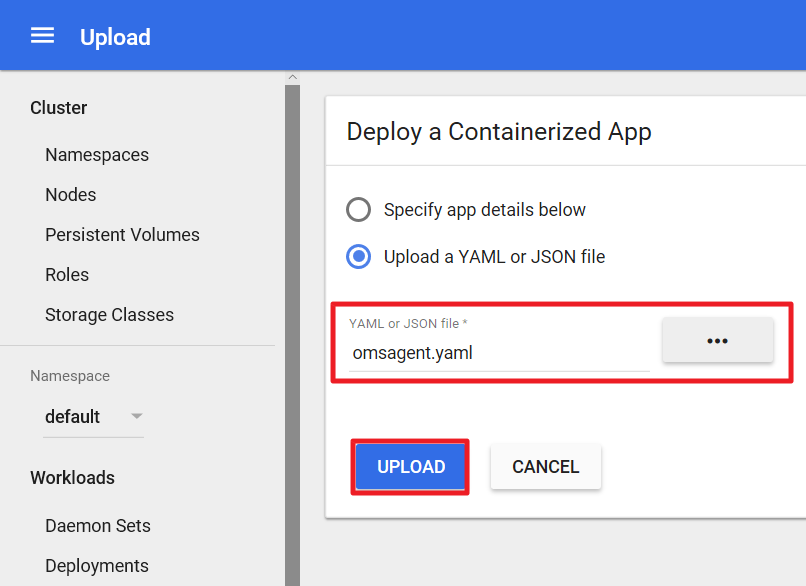
1. Now run the following command in PowerShell console to open your Linux Kubernetes cluster dashboard.

az aks browse -n=AKS-CLUSTER-NAME -g=RESOURCE-GROUP

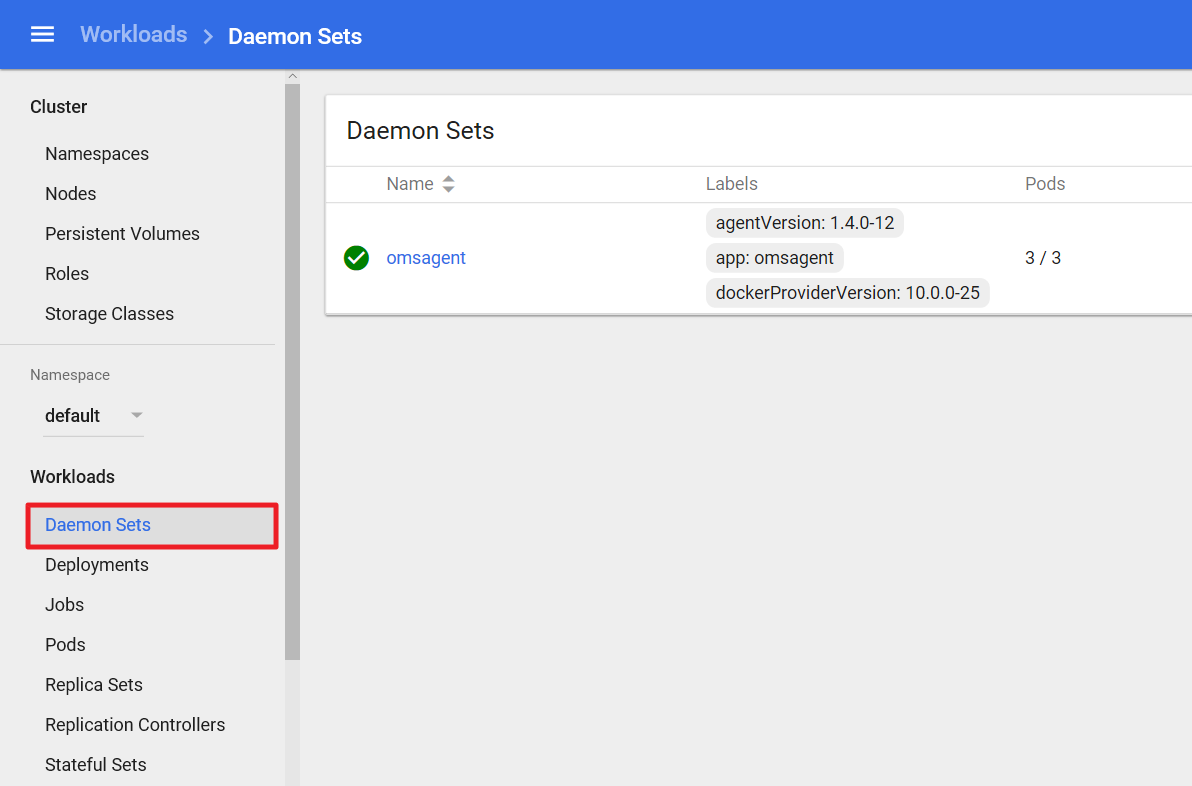
1. Go to the **Deployment** page and click **Create**.



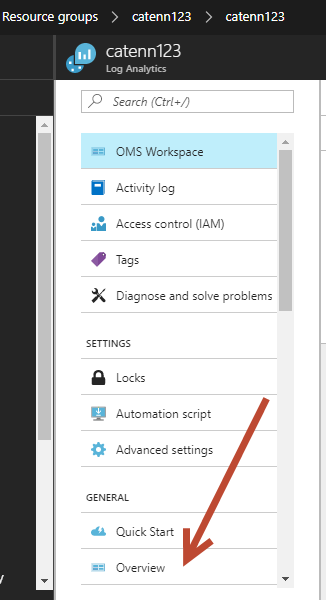
1. Now upload the **omsagent.yaml** file to deploy OMS agents into your cluster as DaemonSets. DaemonSets are used by Kubernetes to run a single instance of a container on each host in the cluster. They're perfect for running monitoring agents.



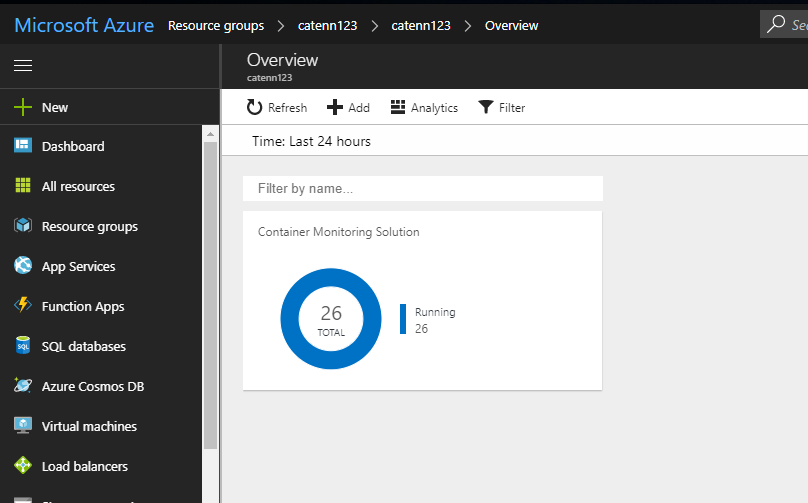
Check the progress of the deployment on **DaemonSets** page.



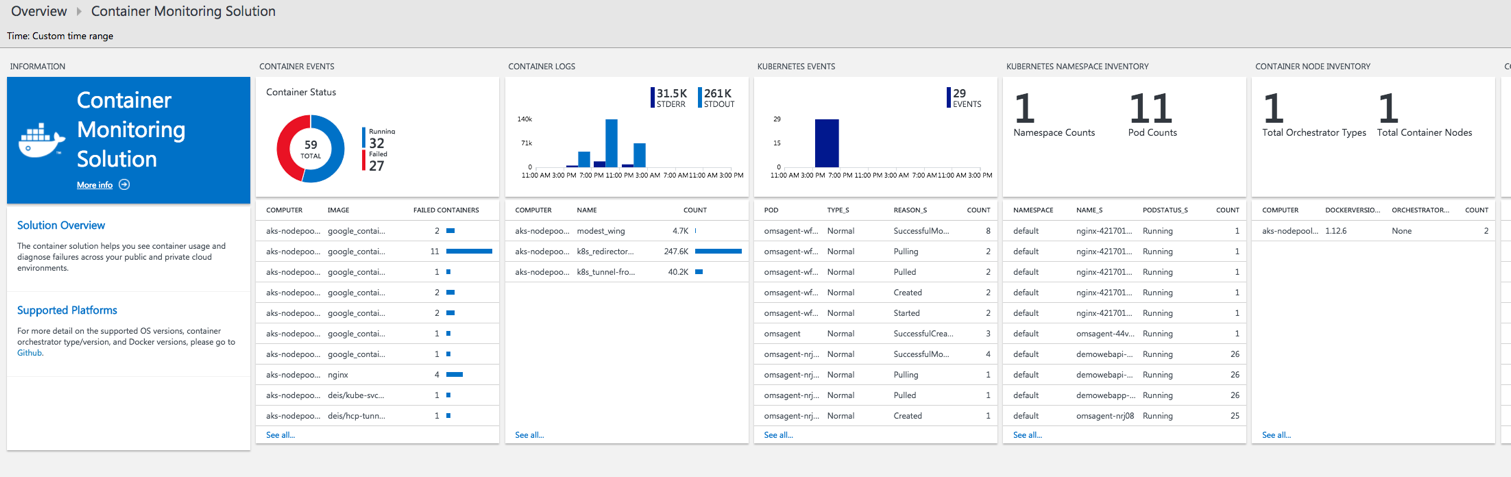
1. Go back to your Log Analytics resource and hit Overview.



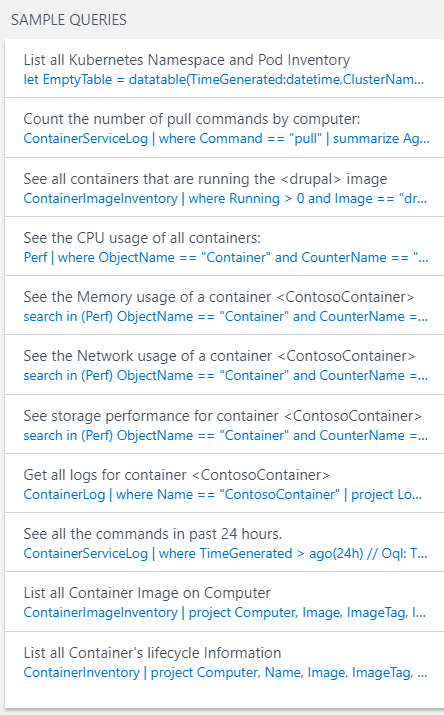
1. Although the OMS agent is up and running in your Kubernetes Dashboard it may take several minutes before OMS dashboard will start displaying the metrics collected from the Kubernetes cluster. Wait ~10 minutes for this to show up and display something other than 0.



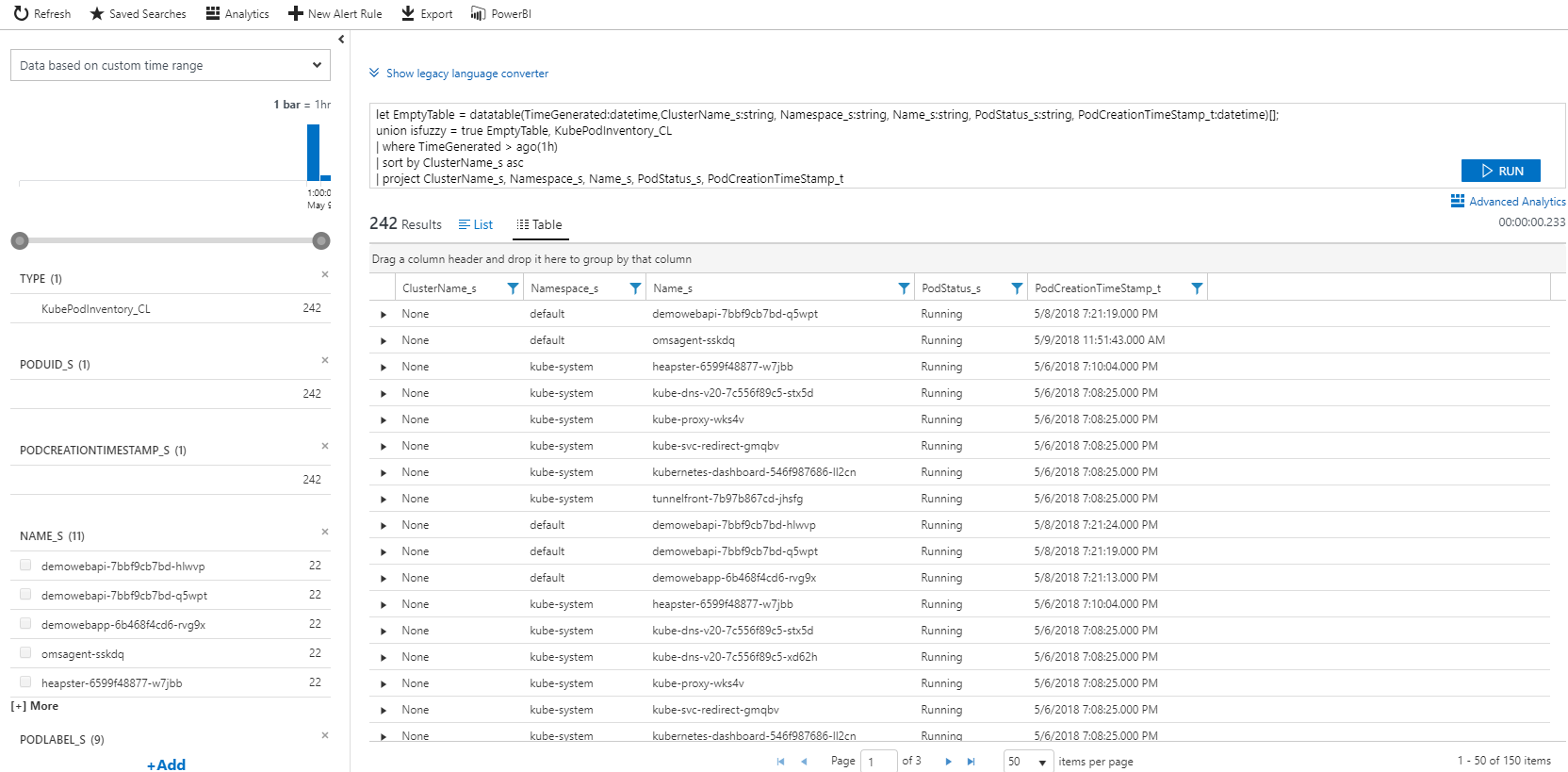
1. Click on the chart title **Container Monitoring Solution** and it will take you to the detail dashboard view. You will see charts about metrics like total running containers, total images, image type count, CPU and memory usage of the containers, and so on.



1. If you scroll all the way to the right, there is a section labeled Sample Queries:



1. Click on any of the blue text sample queries and you will end up in the Log Search. You will see a page like this for the Log Search:

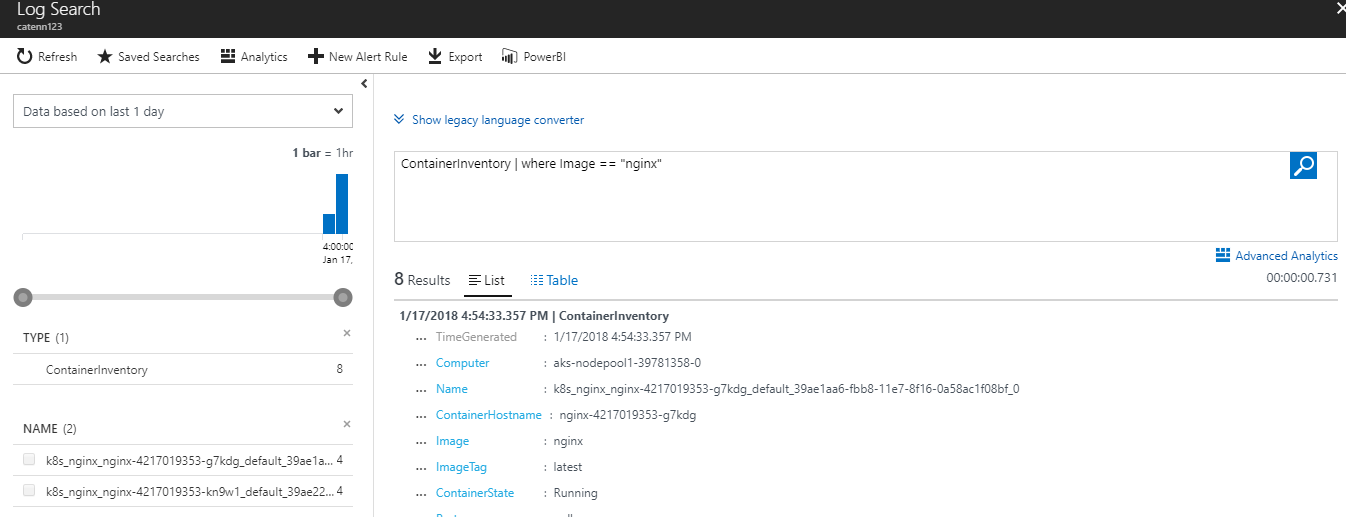


1. Log Search uses specific query language format. You will write a simple query for now but more details about the query language and its format are available at: <https://docs.loganalytics.io/docs/Language-Reference>

Also, more details about type of records collected related to containers is available at: <https://docs.microsoft.com/en-us/azure/log-analytics/log-analytics-containers>

1. Let’s search the logs for the containers based on “nginx” container image (you can replace nginx with any other container image that you have used during deployment). In the search box type the following command and press the search button.

ContainerInventory | where Image == "demo-webapp"

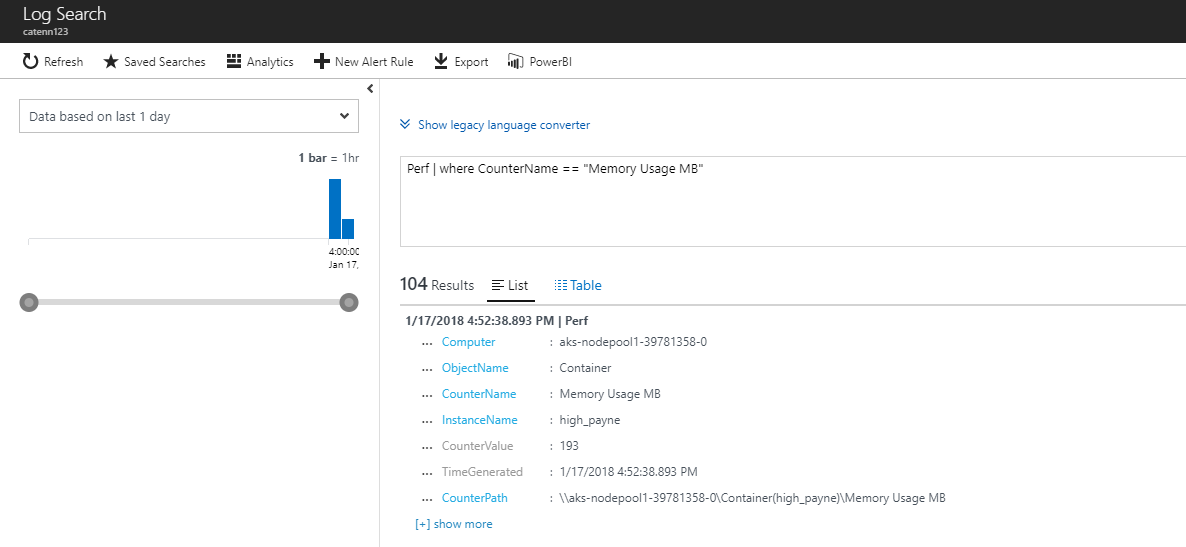


You can also filter based on other pieces of information available. For example: to find all the containers based on nginx that have failed you can use the query:

ContainerInventory | where Image == "demo-webapp" and ContainerState == "Running"

1. You can also view the performance related logs. For example, to look at memory consumption on all nodes you can run following query:

Perf | where CounterName == "Memory Usage MB"



1. Go back to the Sample Queries widget on the Dashboard and test out a few of the queries there (from step 20-21)!