**Qutoas for Namespaces**

Configure Memory and CPU Quotas for a Namespace

## **Create a namespace**

Create a namespace so that the resources you create in this exercise are isolated from the rest of your cluster.

**kubectl create namespace quota-mem-cpu-example**

## **Create a ResourceQuota**

Here is the configuration file for a ResourceQuota object:

| [**quota-mem-cpu.yaml**](https://raw.githubusercontent.com/kubernetes/website/master/docs/tasks/administer-cluster/quota-mem-cpu.yaml) |
| --- |
| **apiVersion: v1**  **kind: ResourceQuota**  **metadata:**  **name: mem-cpu-demo**  **spec:**  **hard:**  **requests.cpu: "1"**  **requests.memory: 1Gi**  **limits.cpu: "2"**  **limits.memory: 2Gi** |

Create the ResourceQuota:

**kubectl create -f https://k8s.io/docs/tasks/administer-cluster/quota-mem-cpu.yaml --namespace=quota-mem-cpu-example**

View detailed information about the ResourceQuota:

**kubectl get resourcequota mem-cpu-demo --namespace=quota-mem-cpu-example --output=yaml**

The ResourceQuota places these requirements on the quota-mem-cpu-example namespace:

* Every Container must have a memory request, memory limit, cpu request, and cpu limit.
* The memory request total for all Containers must not exceed 1 GiB.
* The memory limit total for all Containers must not exceed 2 GiB.
* The CPU request total for all Containers must not exceed 1 cpu.
* The CPU limit total for all Containers must not exceed 2 cpu.

## **Create a Pod**

Here is the configuration file for a Pod:

| [**quota-mem-cpu-pod.yaml**](https://raw.githubusercontent.com/kubernetes/website/master/docs/tasks/administer-cluster/quota-mem-cpu-pod.yaml) |
| --- |
| **apiVersion: v1**  **kind: Pod**  **metadata:**  **name: quota-mem-cpu-demo**  **spec:**  **containers:**  **- name: quota-mem-cpu-demo-ctr**  **image: nginx**  **resources:**  **limits:**  **memory: "800Mi"**  **cpu: "800m"**  **requests:**  **memory: "600Mi"**  **cpu: "400m"** |

Create the Pod:

**kubectl create -f https://k8s.io/docs/tasks/administer-cluster/quota-mem-cpu-pod.yaml --namespace=quota-mem-cpu-example**

Verify that the Pod’s Container is running:

**kubectl get pod quota-mem-cpu-demo --namespace=quota-mem-cpu-example**

Once again, view detailed information about the ResourceQuota:

**kubectl get resourcequota mem-cpu-demo --namespace=quota-mem-cpu-example --output=yaml**

The output shows the quota along with how much of the quota has been used. You can see that the memory and CPU requests and limits for your Pod do not exceed the quota.

**status:**

**hard:**

**limits.cpu: "2"**

**limits.memory: 2Gi**

**requests.cpu: "1"**

**requests.memory: 1Gi**

**used:**

**limits.cpu: 800m**

**limits.memory: 800Mi**

**requests.cpu: 400m**

**requests.memory: 600Mi**

## **Attempt to create a second Pod**

Here is the configuration file for a second Pod:

| [**quota-mem-cpu-pod-2.yaml**](https://raw.githubusercontent.com/kubernetes/website/master/docs/tasks/administer-cluster/quota-mem-cpu-pod-2.yaml) |
| --- |
| **apiVersion: v1**  **kind: Pod**  **metadata:**  **name: quota-mem-cpu-demo-2**  **spec:**  **containers:**  **- name: quota-mem-cpu-demo-2-ctr**  **image: redis**  **resources:**  **limits:**  **memory: "1Gi"**  **cpu: "800m"**  **requests:**  **memory: "700Mi"**  **cpu: "400m"** |

In the configuration file, you can see that the Pod has a memory request of 700 MiB. Notice that the sum of the used memory request and this new memory request exceeds the memory request quota. 600 MiB + 700 MiB > 1 GiB.

Attempt to create the Pod:

**kubectl create -f https://k8s.io/docs/tasks/administer-cluster/quota-mem-cpu-pod-2.yaml --namespace=quota-mem-cpu-example**

The second Pod does not get created. The output shows that creating the second Pod would cause the memory request total to exceed the memory request quota.

**Error from server (Forbidden): error when creating "docs/tasks/administer-cluster/quota-mem-cpu-pod-2.yaml":**

**pods "quota-mem-cpu-demo-2" is forbidden: exceeded quota: mem-cpu-demo,**

**requested: requests.memory=700Mi,used: requests.memory=600Mi, limited: requests.memory=1Gi**

## **Discussion**

As you have seen in this exercise, you can use a ResourceQuota to restrict the memory request total for all Containers running in a namespace. You can also restrict the totals for memory limit, cpu request, and cpu limit.

If you want to restrict individual Containers, instead of totals for all Containers, use a [LimitRange](https://kubernetes.io/docs/tasks/administer-cluster/memory-constraint-namespace/).

## **Clean up**

Delete your namespace:

**kubectl delete namespace quota-mem-cpu-example**

# Configure a Pod Quota for a Namespace

## **Create a namespace**

Create a namespace so that the resources you create in this exercise are isolated from the rest of your cluster.

**kubectl create namespace quota-pod-example**

## **Create a ResourceQuota**

Here is the configuration file for a ResourceQuota object:

| [**quota-pod.yaml**](https://raw.githubusercontent.com/kubernetes/website/master/docs/tasks/administer-cluster/quota-pod.yaml) |
| --- |
| **apiVersion: v1**  **kind: ResourceQuota**  **metadata:**  **name: pod-demo**  **spec:**  **hard:**  **pods: "2"** |

Create the ResourceQuota:

**kubectl create -f https://k8s.io/docs/tasks/administer-cluster/quota-pod.yaml --namespace=quota-pod-example**

View detailed information about the ResourceQuota:

**kubectl get resourcequota pod-demo --namespace=quota-pod-example --output=yaml**

The output shows that the namespace has a quota of two Pods, and that currently there are no Pods; that is, none of the quota is used.

**spec:**

**hard:**

**pods: "2"**

**status:**

**hard:**

**pods: "2"**

**used:**

**pods: "0"**

Here is the configuration file for a Deployment:

| [**quota-pod-deployment.yaml**](https://raw.githubusercontent.com/kubernetes/website/master/docs/tasks/administer-cluster/quota-pod-deployment.yaml) |
| --- |
| **apiVersion: apps/v1beta1**  **kind: Deployment**  **metadata:**  **name: pod-quota-demo**  **spec:**  **replicas: 3**  **template:**  **metadata:**  **labels:**  **purpose: quota-demo**  **spec:**  **containers:**  **- name: pod-quota-demo**  **image: nginx** |

In the configuration file, **replicas: 3** tells Kubernetes to attempt to create three Pods, all running the same application.

Create the Deployment:

**kubectl create -f https://k8s.io/docs/tasks/administer-cluster/quota-pod-deployment.yaml --namespace=quota-pod-example**

View detailed information about the Deployment:

**kubectl get deployment pod-quota-demo --namespace=quota-pod-example --output=yaml**

The output shows that even though the Deployment specifies three replicas, only two Pods were created because of the quota.

**spec:**

**...**

**replicas: 3**

**...**

**status:**

**availableReplicas: 2**

**...**

**lastUpdateTime: 2017-07-07T20:57:05Z**

**message: 'unable to create pods: pods "pod-quota-demo-1650323038-" is forbidden:**

**exceeded quota: pod-demo, requested: pods=1, used: pods=2, limited: pods=2'**

## **Clean up**

Delete your namespace:

**kubectl delete namespace quota-pod-example**

# Configure Quotas for API Objects

## **Create a namespace**

Create a namespace so that the resources you create in this exercise are isolated from the rest of your cluster.

**kubectl create namespace quota-object-example**

## **Create a ResourceQuota**

Here is the configuration file for a ResourceQuota object:

| [**quota-objects.yaml**](https://raw.githubusercontent.com/kubernetes/website/master/docs/tasks/administer-cluster/quota-objects.yaml) |
| --- |
| **apiVersion: v1**  **kind: ResourceQuota**  **metadata:**  **name: object-quota-demo**  **spec:**  **hard:**  **persistentvolumeclaims: "1"**  **services.loadbalancers: "2"**  **services.nodeports: "0"** |

Create the ResourceQuota:

**kubectl create -f https://k8s.io/docs/tasks/administer-cluster/quota-objects.yaml --namespace=quota-object-example**

View detailed information about the ResourceQuota:

**kubectl get resourcequota object-quota-demo --namespace=quota-object-example --output=yaml**

The output shows that in the quota-object-example namespace, there can be at most one PersistentVolumeClaim, at most two Services of type LoadBalancer, and no Services of type NodePort.

**status:**

**hard:**

**persistentvolumeclaims: "1"**

**services.loadbalancers: "2"**

**services.nodeports: "0"**

**used:**

**persistentvolumeclaims: "0"**

**services.loadbalancers: "0"**

**services.nodeports: "0"**

## **Create a PersistentVolumeClaim:**

Here is the configuration file for a PersistentVolumeClaim object:

| [**quota-objects-pvc.yaml**](https://raw.githubusercontent.com/kubernetes/website/master/docs/tasks/administer-cluster/quota-objects-pvc.yaml) |
| --- |
| **kind: PersistentVolumeClaim**  **apiVersion: v1**  **metadata:**  **name: pvc-quota-demo**  **spec:**  **storageClassName: manual**  **accessModes:**  **- ReadWriteOnce**  **resources:**  **requests:**  **storage: 3Gi** |

Create the PersistentVolumeClaim:

**kubectl create -f https://k8s.io/docs/tasks/administer-cluster/quota-objects-pvc.yaml --namespace=quota-object-example**

Verify that the PersistentVolumeClaim was created:

**kubectl get persistentvolumeclaims --namespace=quota-object-example**

The output shows that the PersistentVolumeClaim exists and has status Pending:

**NAME STATUS**

**pvc-quota-demo Pending**

## **Attempt to create a second PersistentVolumeClaim:**

Here is the configuration file for a second PersistentVolumeClaim:

| [**quota-objects-pvc-2.yaml**](https://raw.githubusercontent.com/kubernetes/website/master/docs/tasks/administer-cluster/quota-objects-pvc-2.yaml) |
| --- |
| **kind: PersistentVolumeClaim**  **apiVersion: v1**  **metadata:**  **name: pvc-quota-demo-2**  **spec:**  **storageClassName: manual**  **accessModes:**  **- ReadWriteOnce**  **resources:**  **requests:**  **storage: 4Gi** |

Attempt to create the second PersistentVolumeClaim:

**kubectl create -f https://k8s.io/docs/tasks/administer-cluster/quota-objects-pvc-2.yaml --namespace=quota-object-example**

The output shows that the second PersistentVolumeClaim was not created, because it would have exceeded the quota for the namespace.

**persistentvolumeclaims "pvc-quota-demo-2" is forbidden:**

**exceeded quota: object-quota-demo, requested: persistentvolumeclaims=1,**

**used: persistentvolumeclaims=1, limited: persistentvolumeclaims=1**