Node Selector and Node Affinity

Node labels: - You can constrain a pod to run only on a particular node. To do this use label selectors to make the selection.

i. List of nodes:
   "kubectl get nodes"
ii. To add a label to a node:
   "kubectl label nodes node1 size=big"
iii. You can verify that it worked by running
   "kubectl get nodes –show-labels"

Node Selector: - nodeSelector is the simplest form of node selection constraint. nodeSelector is a field of Pod specification. It specifies a map of key-value pairs.

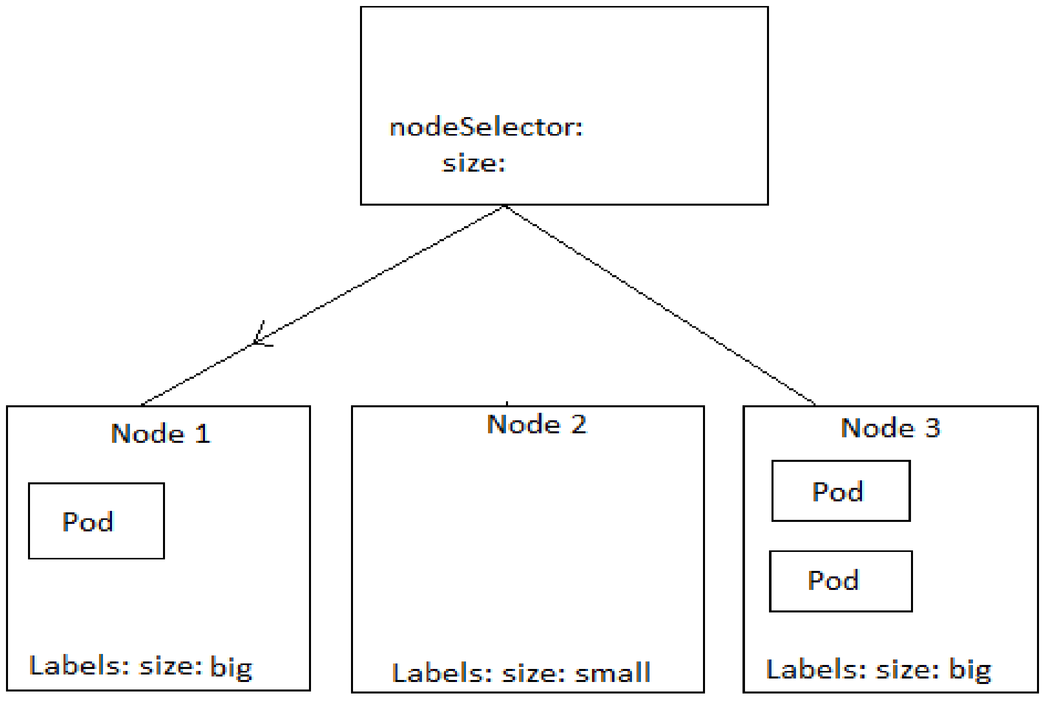
Pod-definition.yml

```
apiVersion: v1
kind: Pod
metadata:
    name: myapp-pod
spec:
   containers:
   - name: data-processor
     image: data-processor
   nodeSelector:
      size: big
```

If you are using workload controller for your application you have to specify nodeSelector in pod template.

Pod-definition.yml

```
apiVersion: v1
kind: Pod
metadata:
    name: myapp-pod
spec:
   replicas: 3
   selector:
     matchLabels:
        app: webapp
        tier: frontend
template:
   metadata:
      labels:
        app: webapp
        tier: frontend
   spec:
     containers:
     - name: data-processor
      image: data-processor
      ports:
      - containerPort: 80
     nodeSelector:
        size: big
```



Node Affinity: - Using node we cannot provide advanced expression like “OR” or “NOT” with node selectors.

The Node Affinity feature provide us with these advanced capabilities to limit pod placement on specific nodes.

Example:

Suppose we have 3 nodes,

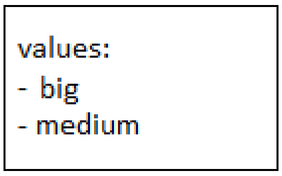
I) Add a label to 2 nodes

i. Kubectl label nodes node1 size=big
ii. Kubectl label nodes node2 size=medium

II) pod-definition.yml

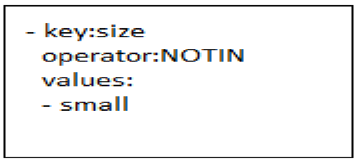
```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
   containers:
   - name: nginx
     image: nginx
   affinity:
      nodeAffinity:
         requiredDuringSchedulingIgnoredDuringExecution:
           nodeSelectorTerms:
           - matchExpressions:
             - key: disktype
               operator: In
               values:
               - big
               - medium
```

➢ If you want your pod could be placed on “big or medium (node1 or node2)” you could simply add the values to the list of values.



```
values:
- big
- medium
```

➢ You could use NOTIN operator to say something like size not in small. Where node affinity will match the node with a size not set to small.



```
 - key:size
   operator:NOTIN
   values:
   - small
```

- We have only set the labels to node 1 and node 2 (bigr and medium). The third node don’t even have the label set. So, we don’t really have to even check the value of the label as long as we sure we don’t set a label to third node. Using the Exists operator will give us the same result.

```
- key:size
  operator:Exists
```

  “Exists” operator will simply check if the label exists on the nodes. You don’t need to values section for that as it does not compare the values.

  Simply, when you not set labels on some nodes and you want to deploy pods on those nodes then use Exists operator. The Exists operator check if the label exists on the nodes if labels is not present the deploy the pods on that nodes.

Node Affinity Types:

- Available Types:
  i. requiredDuringSchedulingIgnoredDuringExecution
  ii. preferredDuringSchedulingIgnoredDuringExecution

- Planned (Coming in Future update)

  requiredDuringSchedulingRequiredExecution

- DuringScheduling:

  DuringScheduling is the state where a pod does not exist and is created for the first time. When a pod is first created the affinity, rules specified are considered to place the pod on the right node.

- required:

  what if the nodes with matching labels are not available?

Suppose we forgot to add label to the node. This time the type of node affinity used comes into play. If you select required type which is the first one the schedular will mandate that the pod be placed on a node with given affinity rules. If it cannot find one the pod will not be scheduled.

- preferred:

  Pod placement is less important than running the workload itself. In this case you

Could set it to preferred and in case where a matching node is not found. Then the schedular will simply ignore node affinity rules and place the pod on any of the available node (Simply this the way of telling the schedular hey try your best to place the pod on matching node but if you really could not find one node just placed it anywhere(node)).

- DuringExecution:

DuringExecution is the state where the pod has been running and a change is made in the environment that affects node affinity. Such as a change in the label of a node.

eg.

Any administrator removed the label from the node. These two affinity types have this value set to ignored which means pod will continue to run and any changes in node affinity will not impact them once they are scheduled.

New type of affinity(requiredDuringSchedulingRequiredExecution) expected in future only have difference in the during execution phase a new option called "requiredduringexecution" is introduced which will kill any pods that are running on nodes that do not meet affinity rules.