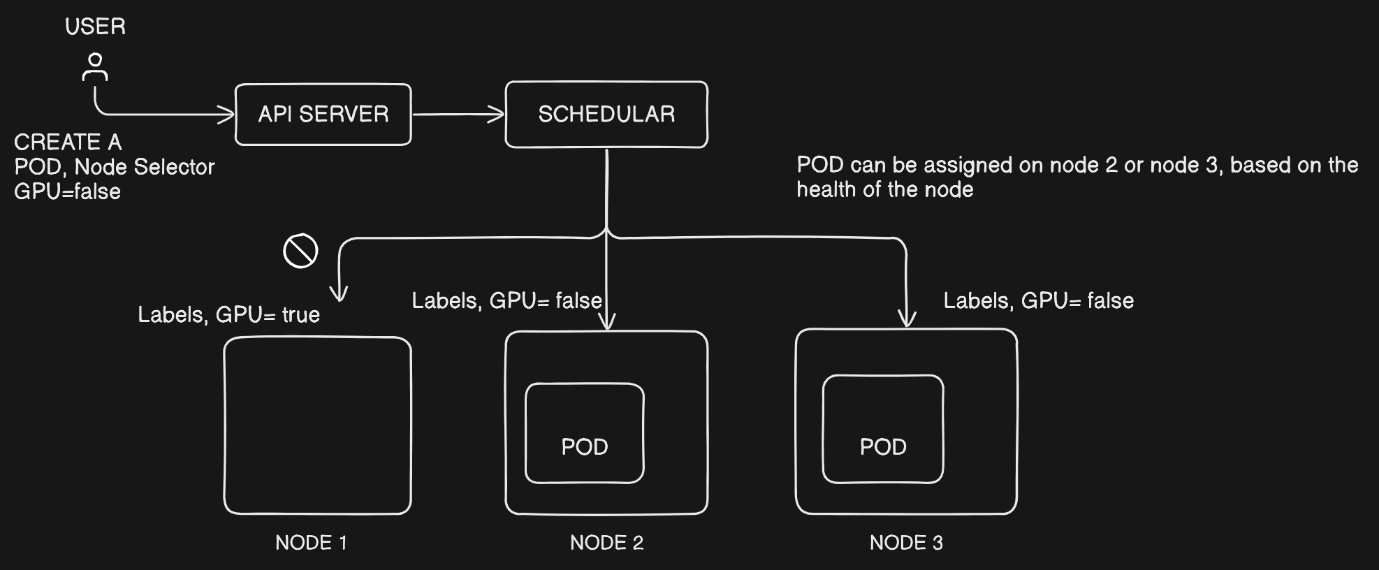
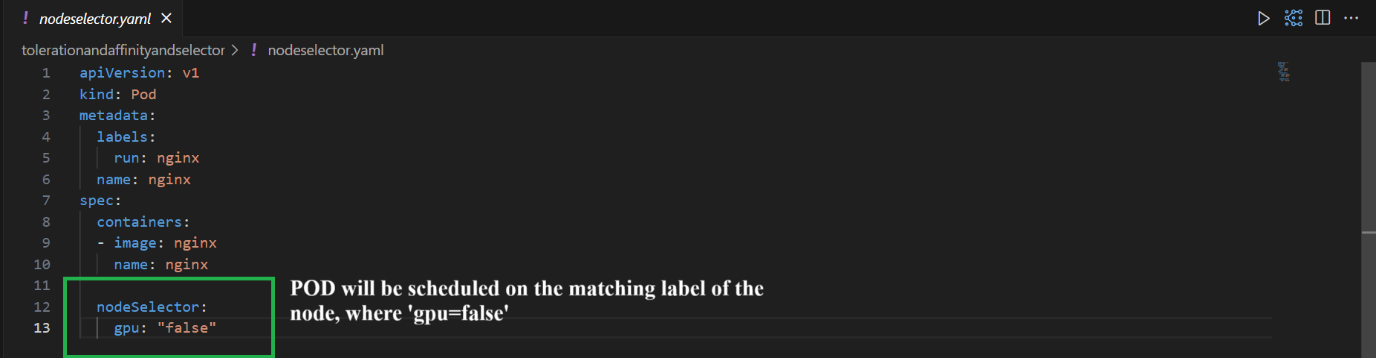
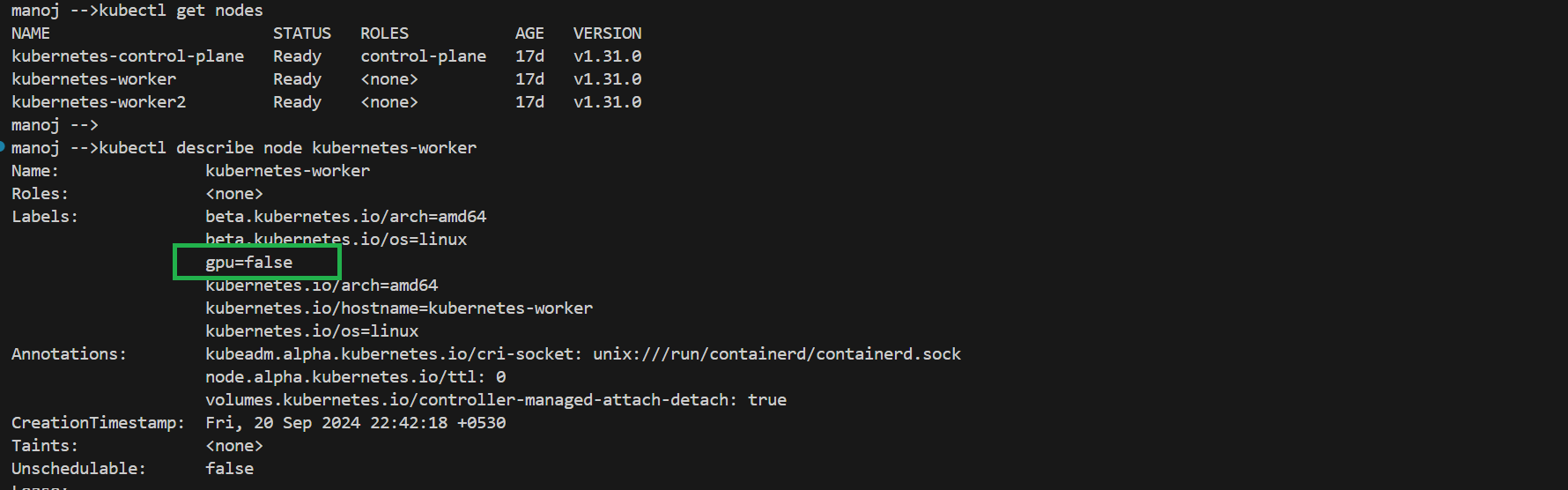
**Node Selector** is a simple way to constrain which nodes a pod can be scheduled on based on node labels. It allows you to ensure that a pod is only scheduled on nodes that match specific criteria.

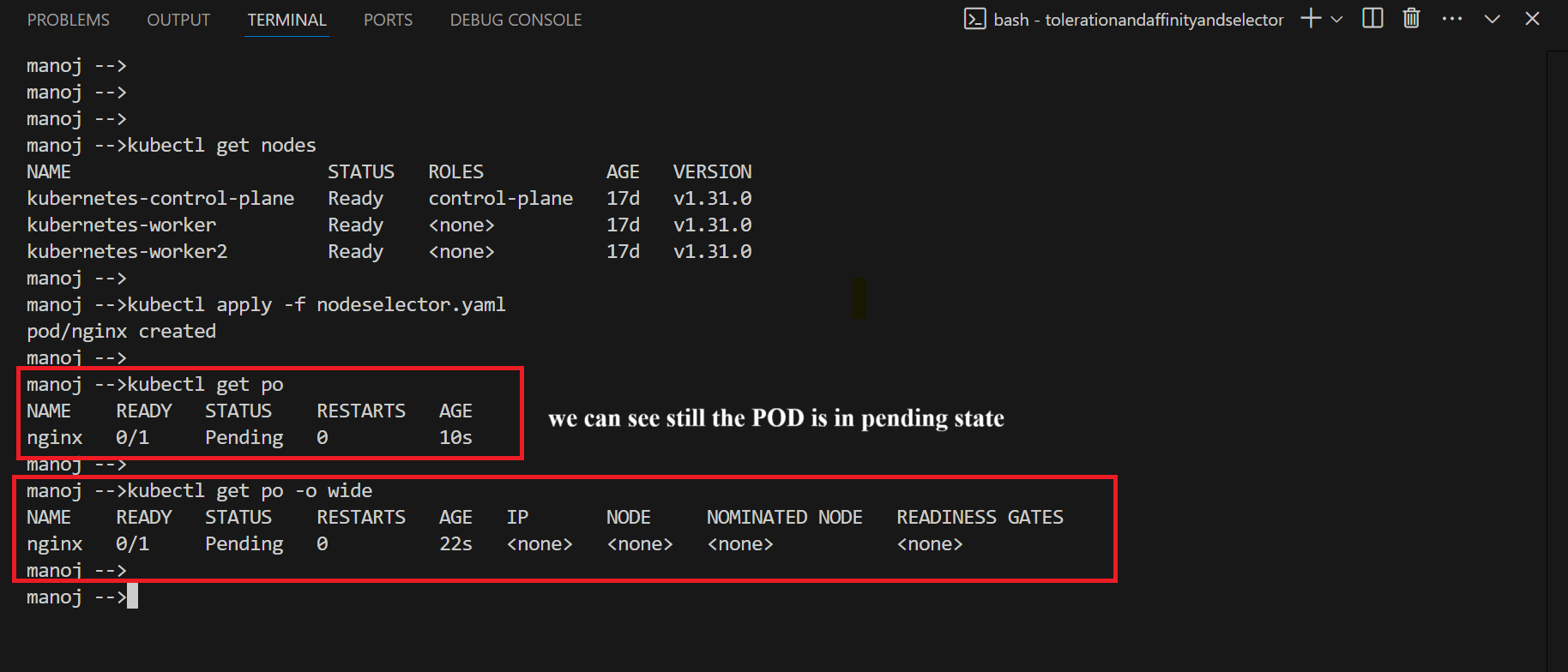
**How Node Selector Works**

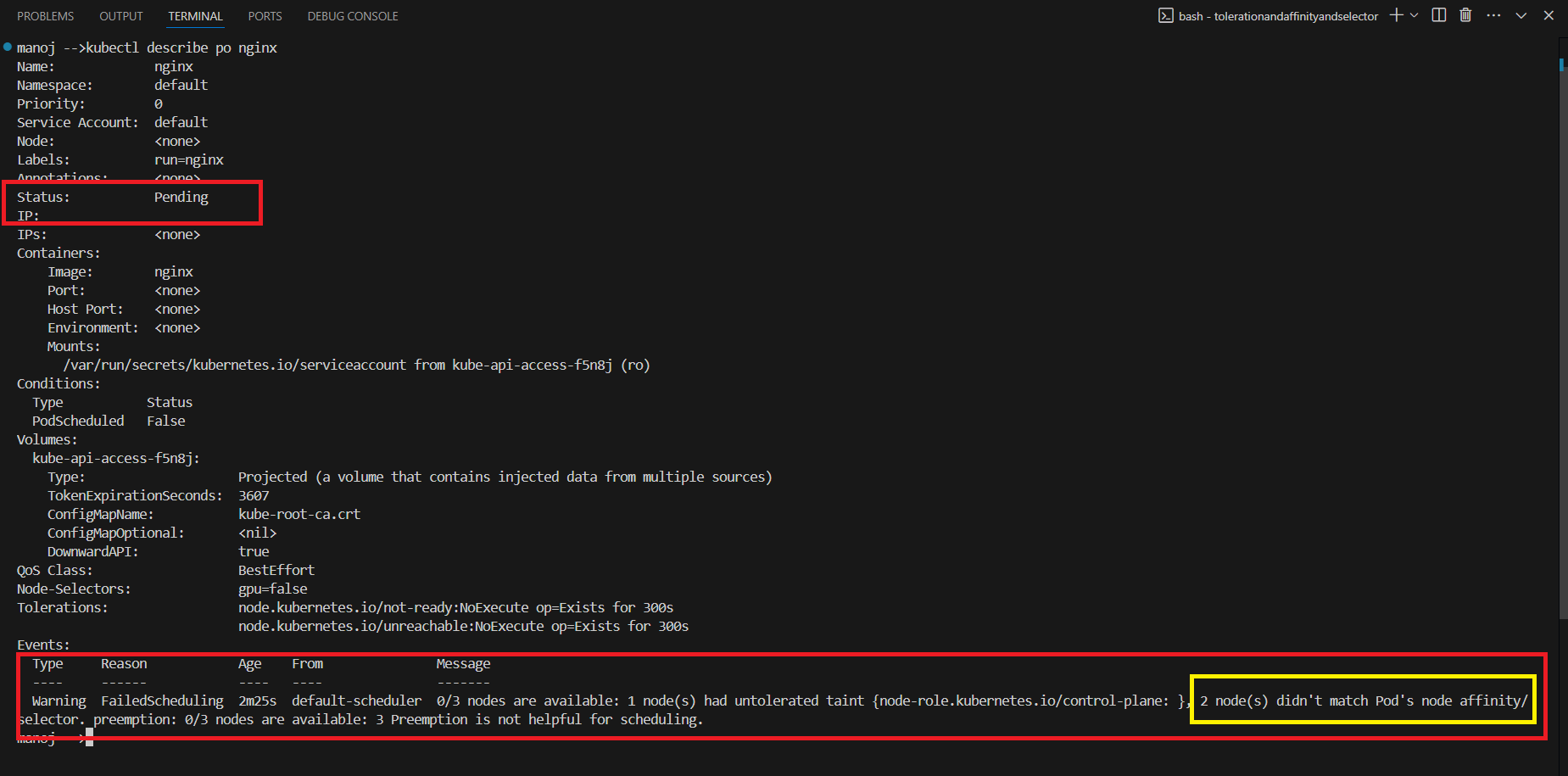
* **Node labels** are key-value pairs attached to nodes. You can add labels to nodes to categorize them (e.g., by hardware, software, or environment).
* A **Node Selector** in a pod's specification matches the node labels. The pod will only be scheduled on nodes that have the specified label(s).



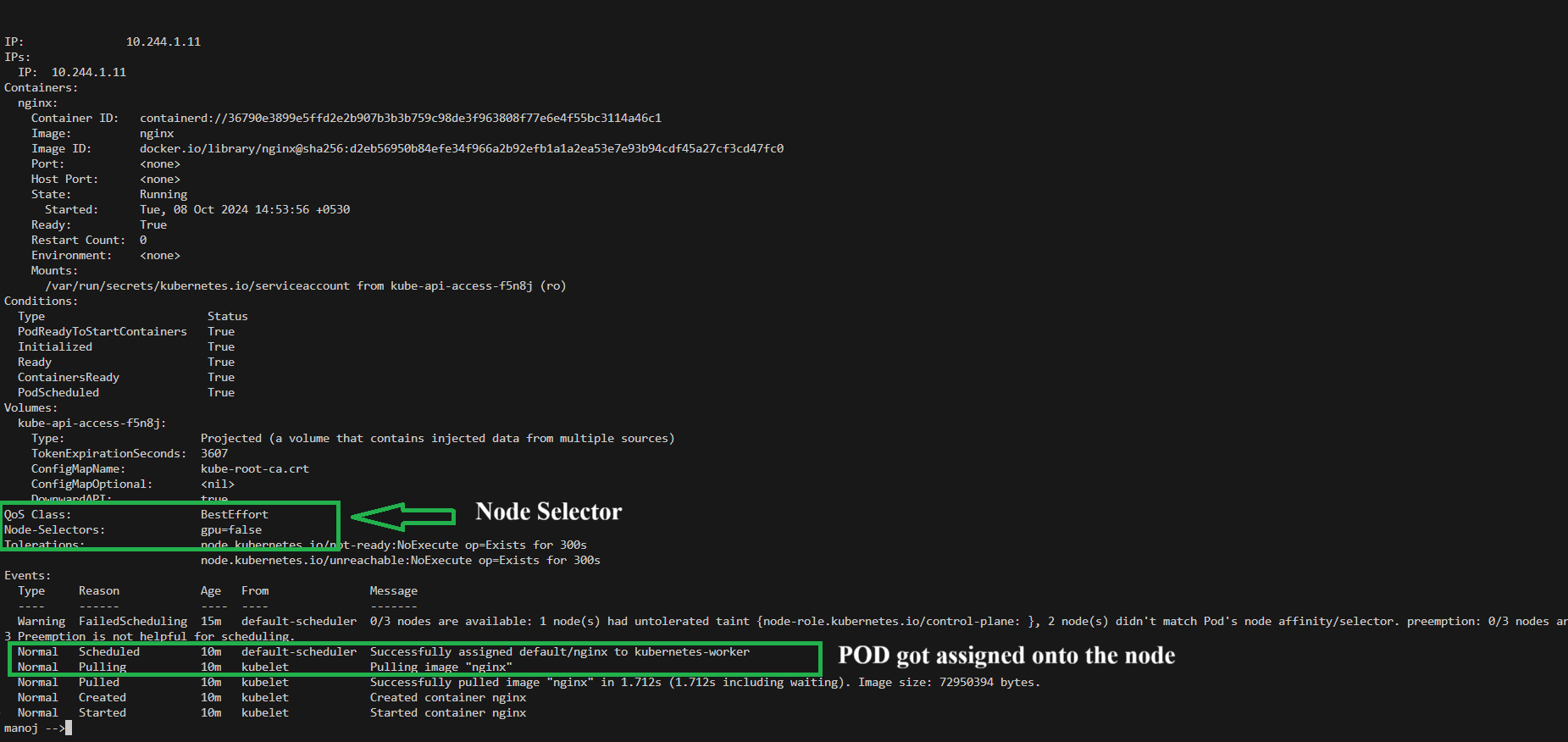
****

****

****

****

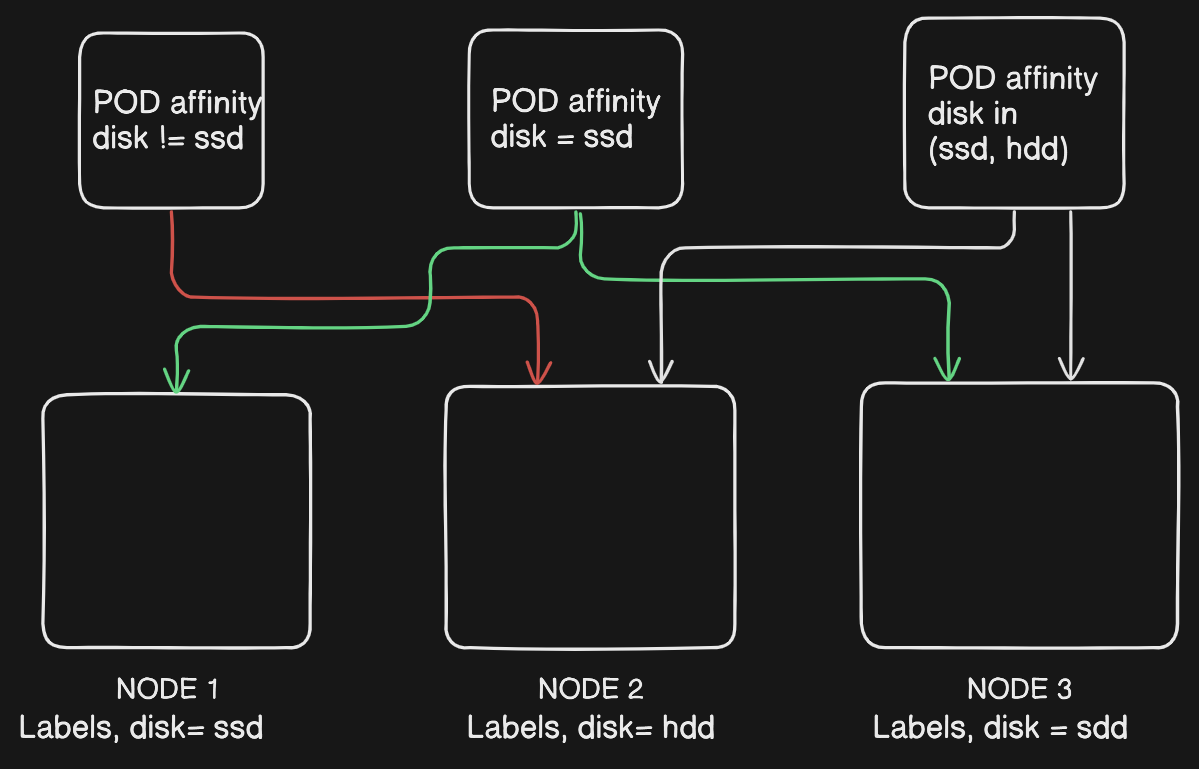
****

****

**Limitations**

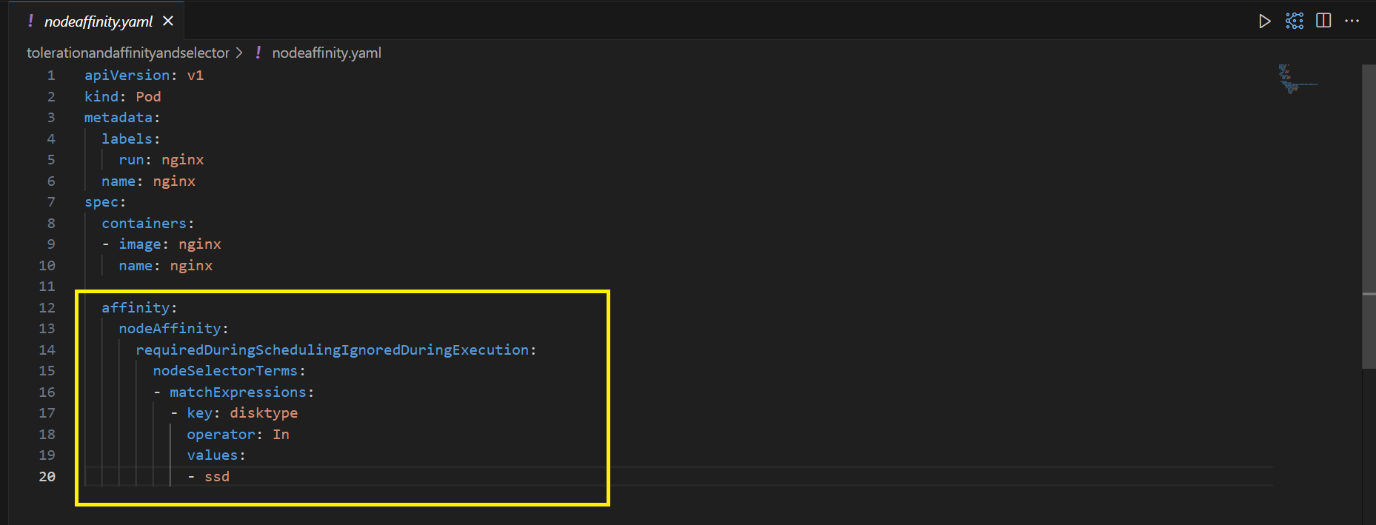
* **Node Selector** is a simple, exact match. If you need more complex placement logic (such as multiple conditions or soft constraints), you might want to use **Node Affinity**, which offers more flexibility for scheduling decisions.

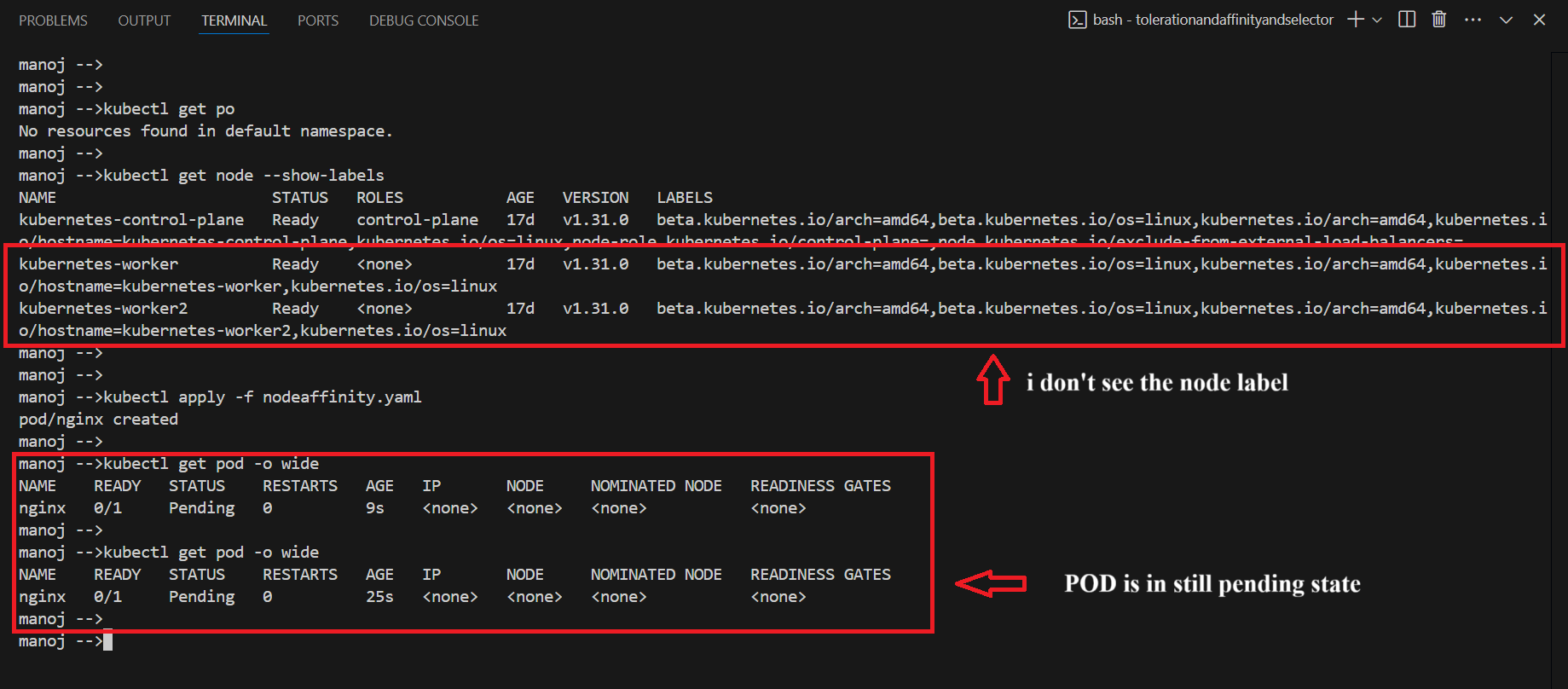
**Node Affinity** in Kubernetes is an advanced scheduling feature that provides more flexible rules for controlling which nodes a pod can be scheduled on, compared to the simpler **Node Selector**. It allows you to express both hard and soft constraints and supports more complex matching logic based on node labels.

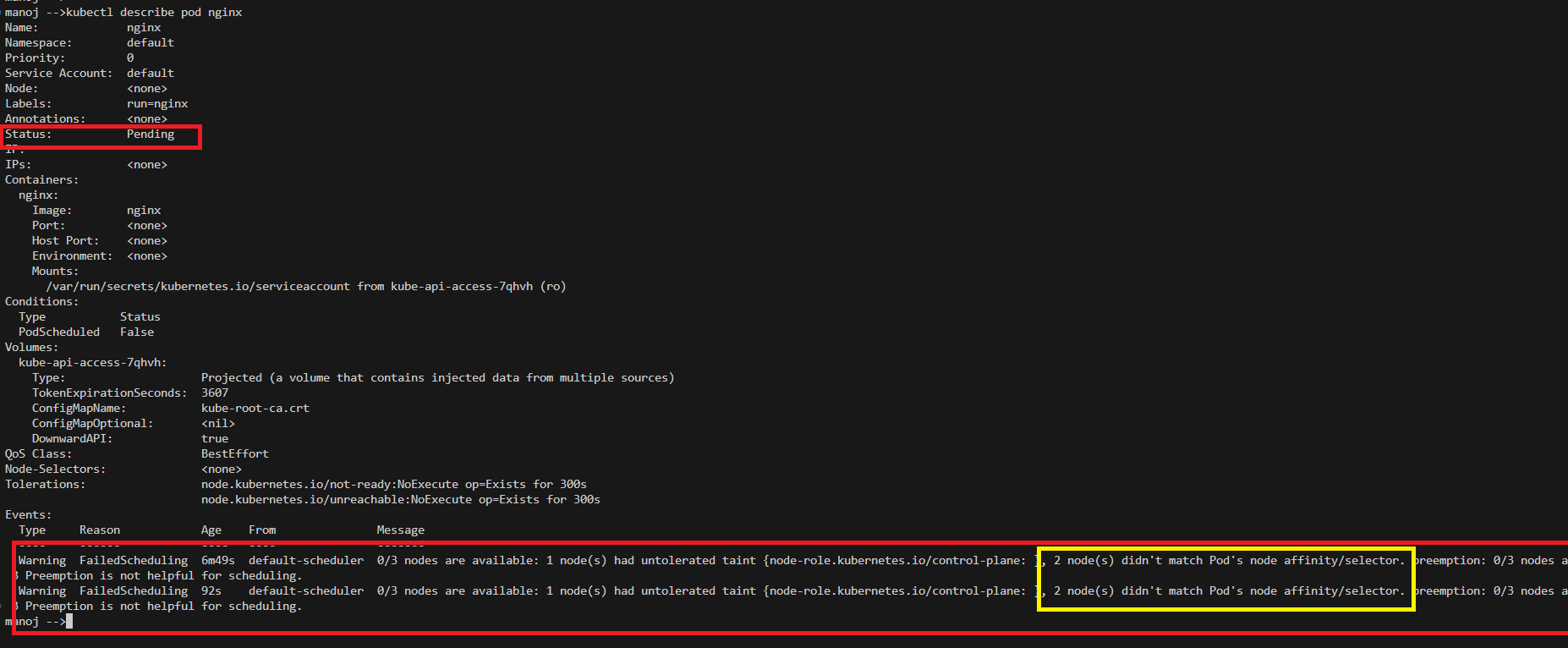


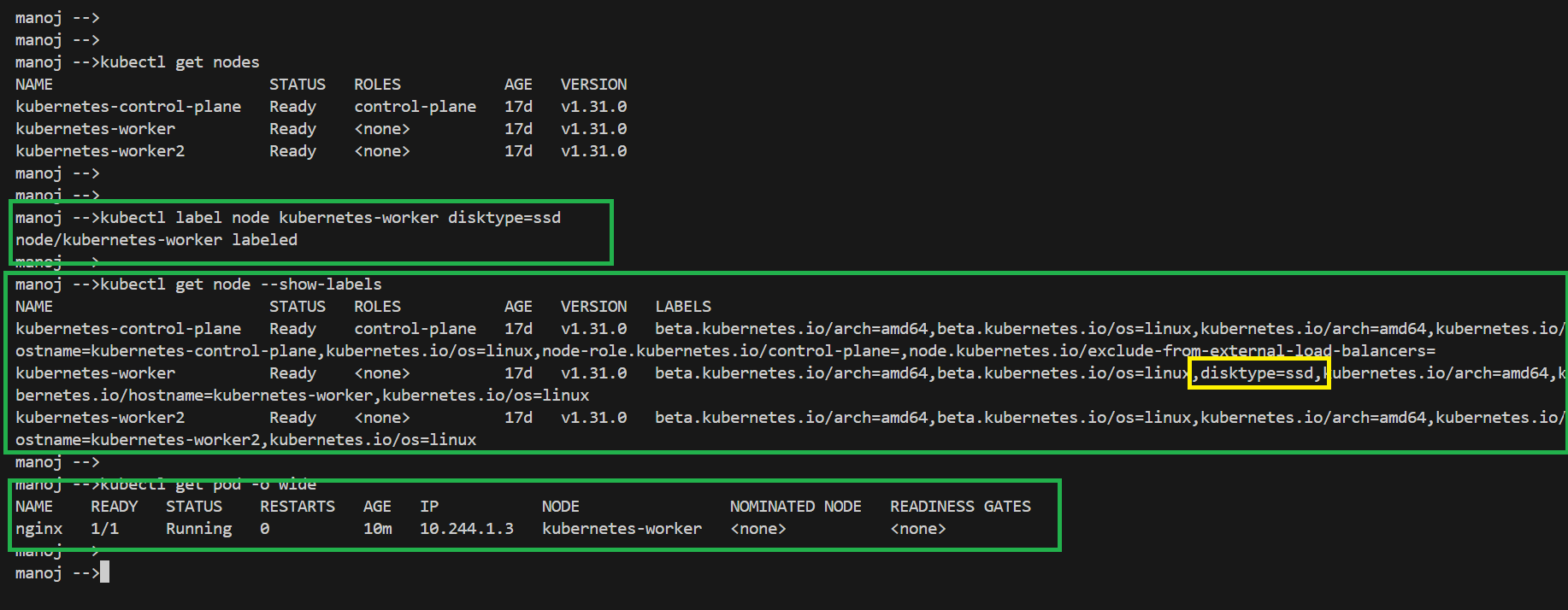
**Key Features of Node Affinity**

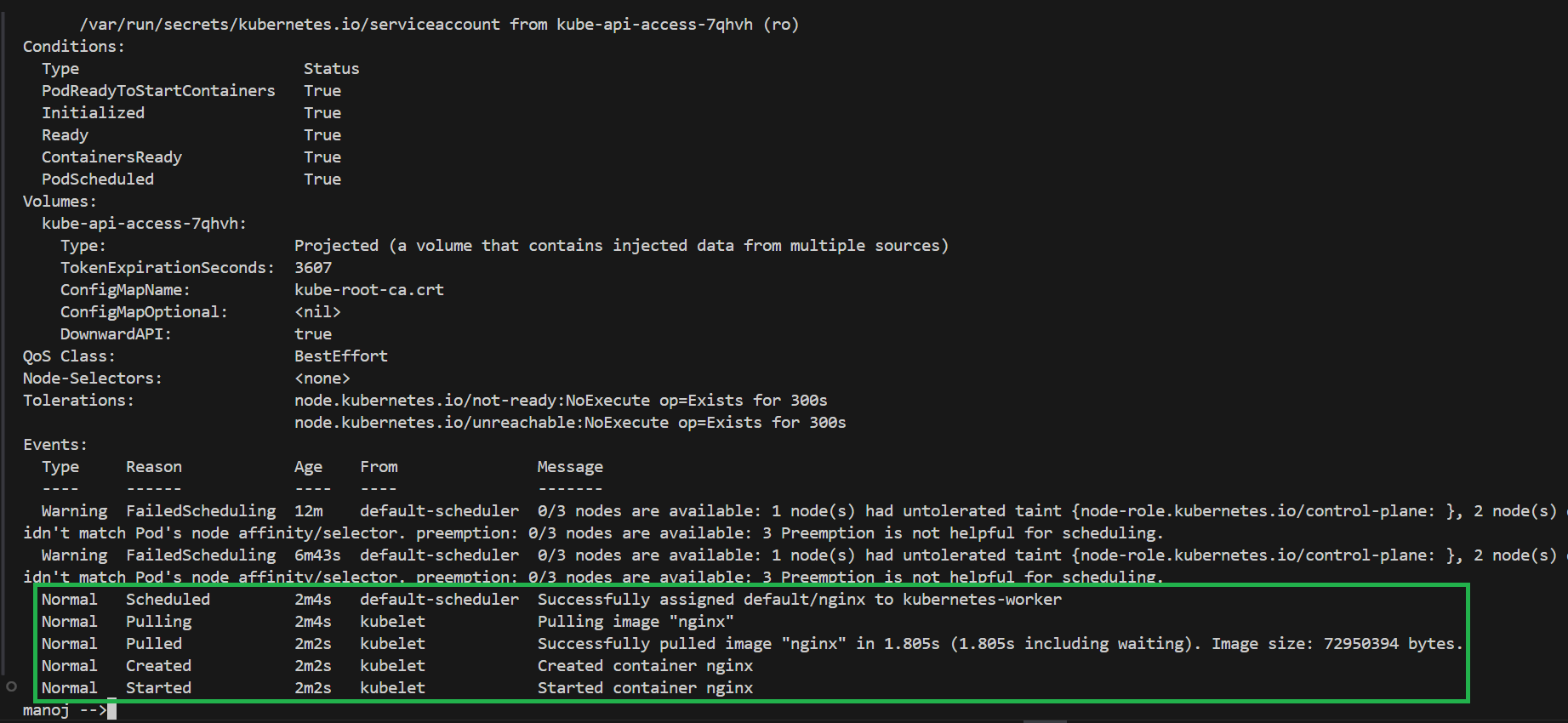
1. **Hard constraints (requiredDuringSchedulingIgnoredDuringExecution)**: These are mandatory rules. If a node does not meet these conditions, the pod will not be scheduled on that node. This behavior is similar to nodeSelector, but with more complex matching options.
2. **Soft constraints (preferredDuringSchedulingIgnoredDuringExecution)**: These are "preferences." Kubernetes will try to place the pod on a node that meets the soft constraint, but it's not a strict requirement. If no such nodes are available, the pod will still be scheduled on other nodes that don't meet the preference.
3. **Match Expressions**: Unlike nodeSelector, which only allows exact matches, node affinity allows you to use expressions such as:
   * **In**: Match nodes with any of the listed values for a key.
   * **NotIn**: Exclude nodes with any of the listed values for a key.
   * **Exists**: Match nodes that have the specified key, regardless of its value.
   * **DoesNotExist**: Exclude nodes that have the specified key.
   * **Gt/Lt**: Match nodes with values greater than or less than a specific number (for numerical label values).











**Note:** we use node affinity, taint and toleration together to make sure pod are accumulating in the nodes that are meant for it.

Eg: if we have large node and that node is run’s on particular type of workload like GPU specific workload or AIML specific workload or node with high performance that is only meant to run data warehousing workload etc, in those case we use this together.