**StatefulSets in Kubernetess**

* A StatefulSet is the Kubernetes controller used to run the stateful application as containers (Pods) in the Kubernetes cluster. StatefulSets assign a sticky identity—an ordinal number starting from zero—to each Pod instead of assigning random IDs for each replica Pod.
* A new Pod is created by cloning the previous Pod’s data. If the previous Pod is in the pending state, then the new Pod will not be created. If you delete a Pod, it will delete the Pod in reverse order, not in random order.

**When to Use StatefulSets?**

* Assume you deployed a MySQL database in the Kubernetes cluster and scaled this to three replicas, and a frontend application wants to access the MySQL cluster to read and write data. The read request will be forwarded to three Pods. However, the write request will only be forwarded to the first (primary) Pod, and the data will be synced with the other Pods. You can achieve this by using StatefulSets.
* Deleting or scaling down a StatefulSet will not delete the volumes associated with the stateful application. This gives you your data safety. If you delete the MySQL Pod or if the MySQL Pod restarts, you can have access to the data in the same volume.

**Create a Headless Service**

* StatefulSets require a headless service for stable network identities.

nginx-headless.yml

apiVersion: v1

kind: Service

metadata:

name: nginx

labels:

app: nginx

spec:

ports:

- port: 80

name: web

clusterIP: None

selector:

app: nginx

* Apply this configuration,
* Create the StatefulSet

nginx-statefulset.yml

apiVersion: apps/v1

kind: StatefulSet

metadata:

name: nginx

spec:

serviceName: "nginx"

replicas: 3

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:1.14.2

ports:

- containerPort: 80

volumeClaimTemplates:

- metadata:

name: www

spec:

accessModes: ["ReadWriteOnce"]

resources:

requests:

storage: 1Gi

* kubectl get statefulsets

**CronJobs in Kubernetes**

* CronJob is meant for performing regular scheduled actions such as backups, report generation, and so on. One CronJob object is like one line of a crontab (cron table) file on a Unix system. It runs a Job periodically on a given schedule, written in Cron format.
* Here’s an example of a CronJob that runs every minute and prints the current date and time,

apiVersion: batch/v1

kind: CronJob

metadata:

name: date-cronjob

spec:

schedule: "\*/1 \* \* \* \*"

jobTemplate:

spec:

template:

spec:

containers:

- name: date

image: busybox

command: ["sh", "-c", "date; echo Hello from the Kubernetes CronJob"]

restartPolicy: OnFailure

* kubectl get cronjobs
* kubectl get jobs --watch