ClusterIP는 클러스터 내부에 새로운 IP를 할당하고 여러 개의 Pod를 바라보는 Load Balancer 기능을 제공한다. 그리고 서비스 이름을 내부 도메인 서버에 등록해 pod 간에 서비스 이름으로 통신할 수 있다.

count-redis-svc.yml

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: redis  spec:  selector:  matchLabels:  app: counter  tier: db  template:  metadata:  labels:  app: counter  tier: db  spec:  containers:  - name: redis  image: redis  ports:  - containerPort: 6379  protocol: TCP  ---  apiVersion: v1  kind: Service  metadata:  name: redis  spec:  ports: *# 서비스가 생성할 Port*  - port: 6379  protocol: TCP  selector:  app: counter  tier: db |

적용 후 조회 해보자.

|  |
| --- |
| $ kubectl apply -f count-redis-svc.yml  $ kubectl get all |

그리고 이제 redis에 접근할 counter라는 앱을 deployment로 생성 해준다.

counter-app.yml

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: counter  spec:  selector:  matchLabels:  app: counter  tier: app  template:  metadata:  labels:  app: counter  tier: app  spec:  containers:  - name: counter  image: ghcr.io/subicura/counter:latest  env:  - name: REDIS\_HOST  value: "redis"  - name: REDIS\_PORT  value: "6379" |

그 다음 적용해준다.

|  |
| --- |
| $ kubectl apply -f counter-app.yml  $ kubectl get po |

counter pod에 접근 해주자

|  |
| --- |
| $ kubectl exec -it counter-7b76fd9bc6-vt4w6 -- sh  *# Unable to use a TTY - input is not a terminal or the right kind of file 해당 오류 뜨면 winty를 앞에 붙여준다.* |

|  |
| --- |
| # apk add curl busybox-extras # install telnet  # curl localhost:3000  # curl localhost:3000  # curl localhost:3000  # telnet redis 6379  > dbsize  > KEYS \*  > GET count  > quit |

그리고 아래의 명령어로 Endpoint를 찾을 수 있다

|  |
| --- |
| $ kubectl get ep    $ kubectl describe ep/redis *# redis의 endpoint* |