Kubernetes User's Guide for Altibase

목차

개요

쿠버네티스(Kubernetes)는 컨테이너화된 애플리케이션의 배포, 스케일링 및 관리를 자동화하는 오픈소스 시스템이다. 쿠버네티스에서 이러한 작업을 처리하는 최소 단위는 파드(Pod)이다. 이 문서는 쿠버네티스 v1.20.4 환경에서 도커 허브(Docker Hub)에 등록된 Altibase 컨테이너 이미지를 이용하여 Altibase 파드를 생성하고 사용하는 가이드를 제시한다. Altibase 도커 이미지 생성은 Altibase 도커 가이드를, 기타 쿠버네티스의 세부 기능은 Kubernetes 홈페이지를 참고한다.

파드 생성 및 사용하기

Altibase 파드 생성

디플로이먼트(Deployment) 워크로드 리소스를 사용하여 생성한 예시와 파드를 직접 생성하는 예시를 소개한다. 일반적으로 파드는 직접 생성하지는 않으며, 워크로드 리소스를 사용하여 생성한다.

디플로이먼트 워크로드 리소스를 사용하여 파드 생성

디플로이먼트를 사용하여 Altibase 서버를 시작하는 컨테이너로 구성된 파드를 생성하는 YAML 파일 예시와 파드 생성 상태를 확인하는 방법이다.

1. YAML 파일 작성

```yaml
# 파일명 : altibase-deploy-pod.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  name: altibase-deploy-pod          # 디플로이먼트 이름 설정
spec:
  replicas: 1
  selector:
    matchLabels:
      app: altibase-deploy-pod       # 디플로이먼트가 관리할 파드.
spec.template.metadata.labels와 일치해야 한다.
  template:
    # 여기서부터 파드 템플릿
    metadata:
      labels:
        app: altibase-deploy-pod     # 파드 이름 설정
    spec:
      containers:
      - image: altibase/altibase     # 컨테이너에서 실행할 도커 허브의 Altibase 이미지
        name: altibase               # 컨테이너 이름
        ports:
        - containerPort: 20300       # 컨테이너 외부로 노출할 Altibase 서버 서비스 포트
          protocol: TCP
        env:
        - name: MODE                 # Altibase 서버 시작 모드 설정
          value: daemon
     # 여기까지 파드 템플릿
```

> template.spec.containers.env 필드에 Altibase 서버 구동 시 필요한 환경 변수를 추가할 수 있다. 관련 환경 변수는 Docker Hub 페이지를 참고한다.

2. 파드 생성

```
$ kubectl create -f altibase-deploy-pod.yaml
deployment.apps/altibase-deploy-pod created
```

3. 파드 생성 확인

```
$ kubectl get pod -o wide
NAME                  READY   STATUS    RESTARTS   AGE   IP              NODE
NOMINATED NODE   READINESS GATES
altibase-deploy-pod   1/1     Running   0          24s   172.16.135.38   node3   <none>
        <none>
```

4. Altibase 서버 접속

```
$ kubectl exec -it altibase-deploy-pod -- /bin/bash
altibase@altibase-pod:~$ . set_altibase.env
altibase@altibase-pod:~$ is
--------------------------------------------------------------------
     Altibase Client Query utility.
     Release Version 7.1.0.5.1
     Copyright 2000, ALTIBASE Corporation or its subsidiaries.
     All Rights Reserved.
--------------------------------------------------------------------
ISQL_CONNECTION = TCP, SERVER = localhost, PORT_NO = 20300
iSQL> SELECT * FROM TAB;
```

파드 직접 생성

Altibase 서버를 시작하는 컨테이너로 구성된 파드를 직접 생성하는 YAML 파일 예시와 파드 상태를 확인하는 방법이다.

1. YAML 파일 작성

```
# 파일명 : altibase-pod.yaml

apiVersion: v1
kind: Pod
metadata:
  labels:
    app: altibase
  name: altibase-pod                  # 파드 이름 설정
spec:
  containers:
  - image: altibase/altibase          # 컨테이너에서 실행할 도커 허브 이미지
    name: altibase
    ports:
    - containerPort: 20300            # 컨테이너 외부로 노출할 Altibase 서비스 포트
      protocol: TCP
    env:
```

```
    - name: MODE                          # altibase/altibase 이미지 시작 모드 설정
      value: daemon
```

> spec.containers.env 필드에 Altibase 서버 구동 시 필요한 환경 변수를 추가할 수 있다. 관련 환경 변수는 Docker Hub 페이지를 참고한다.

2. 파드 생성

```
$ kubectl create -f altibase-pod.yaml
pod/altibase-pod created
```

3. 파드 생성 확인

```
$ kubectl get pod -o wide
NAME           READY   STATUS    RESTARTS   AGE   IP              NODE    NOMINATED NODE
  READINESS GATES
altibase-pod   1/1     Running   0          11m   172.16.135.37   node3   <none>
  <none>
```

4. Altibase 서버 접속

```
$ kubectl exec -it altibase-pod -- /bin/bash

altibase@altibase-pod:~$ . set_altibase.env
altibase@altibase-pod:~$ is
--------------------------------------------------------------------
     Altibase Client Query utility.
     Release Version 7.1.0.5.1
     Copyright 2000, ALTIBASE Corporation or its subsidiaries.
     All Rights Reserved.
--------------------------------------------------------------------
ISQL_CONNECTION = TCP, SERVER = localhost, PORT_NO = 20300
iSQL> SELECT * FROM TAB;
```

볼륨(Volume) 사용

파드의 특징 중 하나는 반영속성이다. 파드가 임의의 이유로 종료되어 새로 생성될 경우 기존 파드에 저장되어 있는 모든 데이터는 휘발된다. 파드의 종료 여부와 상관없이 데이터를 보존하기 위해 쿠버네티스는 외장 디스크를 추상화 한 다양한 종류의 볼륨(Volume)을 제공한다. 볼륨에 대한 설명과 자세한 사용법에 대해서는 Kubernetes 홈페이지를 참고한다.

다음은 NFS (Network File System) 볼륨에 DB 데이터 파일과 온라인 로그 파일의 경로를 설정한 YAML 파일 예시이다. 이 예시는 NFS 서버 설정 후에 진행해야 한다.

1. YAML 파일 작성

```
# 파일명 : altibase-deploy-vol-node1.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  name: altibase-deploy-vol-node1                          # 디플로이먼트 이름 설정
spec:
  replicas: 1
  selector:
    matchLabels:
      app: altibase-node1                                  # 디플로이먼트가 관리할 파드.
spec.template.metadata.labels와 일치해야 한다.
  template:
    # 여기서부터 파드 템플릿
    metadata:
      labels:
        app: altibase-node1                                # 파드 이름 설정
    spec:
      containers:
      - name: altibase                                     # 컨테이너 이름
        image: altibase/altibase                           # 컨테이너에서 실행할 도커 허브의
Altibase 이미지
        volumeMounts:
        # spec.volumes.name에서 지정한 볼륨을 컨테이너에 마운트할 위치 선언
        - name: altibase-nfs-dbs                           # spec.volumes.name에서 지정한 볼
륨(DB 데이터 파일 용도)
          mountPath: /home/altibase/altibase_home/dbs      # altibase-nfs-dbs 볼륨에서 사용할
디렉토리 지정
        - name: altibase-nfs-logs                          # spec.volumes.name에서 지정한 볼
륨(온라인 로그 파일 용도)
          mountPath: /home/altibase/altibase_home/logs     # altibase-nfs-logs 볼륨에서 사용
할 디렉토리 지정
        ports:
        - containerPort: 20300                             # 컨테이너 외부로 노출할 Altibase
서비스 포트
          protocol: TCP
        env:
        - name: MODE
          value: daemon
      volumes:
      - name: altibase-nfs-dbs                             # 파드에 제공할 볼륨 지정(DB 데이터
파일 용도)
        nfs:                                               # altibase-nfs-dbs 볼륨의 유형 지
정
          server: 192.168.204.139                          # NFS 서버 IP
          path: /home/altibase-nfs/node1/dbs               # NFS 서버의 디렉토리 지정
      - name: altibase-nfs-logs                            # 파드에 제공할 볼륨 지정(온라인 로그
파일 용도)
        nfs:                                               # altibase-nfs-logs 볼륨의 유형 지
정
          server: 192.168.204.139                          # NFS 서버 IP
```

```
          path: /home/altibase-nfs/node1/logs            # NFS 서버의 디렉토리 지정
    # 여기까지 파드 템플릿
```

2. 파드 생성

```
$ kubectl create -f altibase-deploy-vol-node1.yaml
deployment.apps/altibase-deploy-vol-node1 created
```

3. 파드 생성 확인

```
$ kubectl get pod -o wide
NAME                                          READY   STATUS    RESTARTS   AGE   IP
        NODE        NOMINATED NODE   READINESS GATES
altibase-deploy-vol-node1-7599dcb85b-75jwp    1/1     Running   0          9s
 172.16.107.207   k8s-node3   <none>           <none>
```

서비스(Service) 사용하기

파드는 파드 고유의 IP를 가지긴 하지만 클러스터 내부용 IP이며 디플로이먼트가 파드를 재생성할 때마다 동적 IP를 할당받는다. 쿠버네티스는 이렇게 동적으로 변하는 파드에 고정된 방법으로 접근하기 위해서 서비스(Service) 리소스를 제공한다. 서비스를 생성하면 정적 IP가 할당되고 고유한 DNS 이름을 사용할 수 있으며 서비스가 존재하는 동안 변경되지 않는다.

다음은 한 개의 서비스와 Altibase 파드 2개를 생성하고 고정 IP를 이용해 한 노드에서 다른 노드의 Altibase 서버에 접속한 예이다.

1. YAML 파일 생성

```
# 파일명 : altibase_pod_svc.yaml

# altibase-deploy-pod1 디플로이먼트
apiVersion: apps/v1
kind: Deployment
metadata:
  name: altibase-deploy-pod1          # 디플로이먼트 이름
spec:
  replicas: 1
  selector:
    matchLabels:
      app: altibase-deploy-pod1       # 디플로이먼트가 관리할 파드.
spec.template.metadata.labels와 일치해야 한다.
  template:
    # 여기서부터 파드 템플릿
    metadata:
      labels:
        app: altibase-deploy-pod1      # 파드 이름
    spec:
      containers:
      - image: altibase/altibase
        name: altibase                 # 컨테이너 이름
```

```
        ports:
        - containerPort: 20300          # 컨테이너 외부로 노출할 Altibase 서비스 포트
          protocol: TCP
        env:
        - name: MODE
          value: daemon
    # 여기까지 파드 템플릿
---

# altibase-deploy-pod2 디플로이먼트
apiVersion: apps/v1
kind: Deployment
metadata:
  name: altibase-deploy-pod2          # 디플로이먼트 이름
spec:
  replicas: 1
  selector:
    matchLabels:
      app: altibase-deploy-pod2       # 디플로이먼트가 관리할 파드.
spec.template.metadata.labels와 일치해야 한다.
  template:
    # 여기서부터 파드 템플릿
    metadata:
      labels:
        app: altibase-deploy-pod2     # 파드 이름
    spec:
      containers:
      - image: altibase/altibase
        name: altibase                # 컨테이너 이름
        ports:
        - containerPort: 20300        # 컨테이너 외부로 노출할 Altibase 서비스 포트
          protocol: TCP
        env:
        - name: MODE
          value: daemon
     # 여기까지 파드 템플릿
---

# 고정 IP를 할당받을 서비스
apiVersion: v1
kind: Service
metadata:
  name: altibase-service       # 서비스 이름
spec:
  ports:
  - port: 20300                # 보통 port와 targetport는 같은 값으로 설정
    targetPort: 20300          # 서비스로 들어온 요청을 전달할, 파드가 LISTEN하고 있는 포트
  selector:
    app: altibase-deploy-pod1  # 서비스가 요청을 전달할 파드
```

2. 파드와 서비스 생성

파드와 서비스를 생성한다.

```
$ kubectl create -f altibase_pod_svc.yaml
```

3. 파드와 서비스 상태 확인

altibase-service 서비스에 고정 IP 10.110.31.65이 할당되었다.

```
$ kubectl get pod,service -o wide
NAME                                        READY   STATUS    RESTARTS   AGE     IP
        NODE        NOMINATED NODE   READINESS GATES
pod/altibase-deploy-pod1-68d566b68c-sm6kw   1/1     Running   0          5m43s
172.16.169.145   k8s-node2   <none>           <none>
pod/altibase-deploy-pod2-579498ccdb-zv2mg   1/1     Running   0          5m29s
172.16.107.202   k8s-node3   <none>           <none>


NAME                       TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)     AGE
SELECTOR
service/altibase-service   ClusterIP   10.110.31.65   <none>        20300/TCP   4m49s
app=altibase-deploy-pod1
service/kubernetes         ClusterIP   10.96.0.1      <none>        443/TCP     15m
<none>
```

3. 고정IP 로 Altibase 서버에 접속

altibase-deploy-pod2에서 iSQL을 이용해서 서비스에 할당된 고정IP로 altibase-deploy-pod1에 구동한 Altibase 서버에 접속하는 예이다.

```
$ k exec -it altibase-deploy-pod2-579498ccdb-zv2mg -- /bin/bash

[altibase@altibase-pod2 ~] $ . set_altibase.env
[altibase@altibase-pod2 ~] $ is -s 10.110.31.65 -port 20300 -u sys -p manager   # 서비스에 할당된 고정IP 사용
--------------------------------------------------------------------
     Altibase Client Query utility.
     Release Version 7.1.0.1.6
     Copyright 2000, ALTIBASE Corporation or its subsidiaries.
     All Rights Reserved.
--------------------------------------------------------------------
ISQL_CONNECTION = TCP, SERVER = 10.98.64.213, PORT_NO = 20300
iSQL>
```

파드를 사용하여 Altibase 이중화하기

디플로이먼트, 서비스를 이용하여 Altibase 이중화 노드 구성 방법을 설명한다. 파드는 생성할 때마다 동적 IP를 할당받기 때문에 Altibase 이중화 생성 구문에 원격 서버의 IP 대신 호스트 네임을 사용한다.

아래는 디플로이먼트로 고정된 DNS 이름을 갖는 파드 생성 및 파드를 가리키는 서비스를 생성하고 Altibase 서버에 접속하여 이중화 객체 생성과 수행 여부를 확인하는 예시이다.

1. 디플로이먼트를 생성하는 YMAL 파일 작성

Altibase 이중화 노드를 구성할 2개의 파드를 생성한다.

```
# 파일명 : altibase-deploy-pod.yaml

# altibase-deploy-vol-node1 디플로이먼트 생성
apiVersion: apps/v1
kind: Deployment
metadata:
  name: altibase-deploy-vol-node1  # 디플로이먼트 이름
spec:
  replicas: 1
  selector:
    matchLabels:
      app: altibase-node1          # 디플로이먼트가 관리할 파드.
spec.template.metadata.labels와 일치해야 한다.
  template:
    # 여기서부터 파드 템플릿
    metadata:
      labels:
        app: altibase-node1        # 파드 이름 설정
    spec:
      containers:
      - name: altibase
        image: altibase/altibase
        volumeMounts:
        - name: altibase-nfs-dbs
          mountPath: /home/altibase/altibase_home/dbs
        - name: altibase-nfs-logs
          mountPath: /home/altibase/altibase_home/logs
        ports:
        - containerPort: 20300      # 컨테이너 외부로 노출할 Altibase 서비스 포트
          protocol: TCP
        - containerPort: 20301      # 컨테이너 외부로 노출할 Altibase 이중화 포트
          protocol: TCP
        env:
        - name: MODE
          value: replication
        - name: SLAVE_REP_PORT      # Altibase 이중화 활성화
          value: "20301"            # Altibase 이중화 포트
      volumes:
      - name: altibase-nfs-dbs
        nfs:
```

```
          server: 192.168.204.139
          path: /home/altibase-nfs/node1/dbs
      - name: altibase-nfs-logs
        nfs:
          server: 192.168.204.139
          path: /home/altibase-nfs/node1/logs

---

# altibase-deploy-vol-node2 디플로이먼트 생성
apiVersion: apps/v1
kind: Deployment
metadata:
  name: altibase-deploy-vol-node2     # 디플로이먼트 이름
spec:
  replicas: 1
  selector:
    matchLabels:
      app: altibase-node2             # 디플로이먼트가 관리할 파드.
spec.template.metadata.labels와 일치해야 한다.
  template:
   # 여기서부터 파드 템플릿
    metadata:
      labels:
        app: altibase-node2           # 파드 이름 설정
    spec:
      containers:
      - name: altibase
        image: altibase/altibase
        volumeMounts:
        - name: altibase-nfs-dbs
          mountPath: /home/altibase/altibase_home/dbs
        - name: altibase-nfs-logs
          mountPath: /home/altibase/altibase_home/logs
        ports:
        - containerPort: 20300      # 컨테이너 외부로 노출할 Altibase 서비스 포트
          protocol: TCP
        - containerPort: 20301      # 컨테이너 외부로 노출할 Altibase 이중화 포트
          protocol: TCP
        env:
        - name: MODE
          value: replication
        - name: SLAVE_REP_PORT      # Altibase 이중화 활성화
          value: "20301"            # Altibase 이중화 포트
      volumes:
      - name: altibase-nfs-dbs
        nfs:
          server: 192.168.204.139
          path: /home/altibase-nfs/node2/dbs
      - name: altibase-nfs-logs
        nfs:
          server: 192.168.204.139
          path: /home/altibase-nfs/node2/logs
```

2. 서비스를 생성하는 YAML 파일 작성

2개 파드와 대응하는 2개의 서비스를 생성하는 예시이다.

```
# 파일명 : altibase-service.yaml

apiVersion: v1
kind: Service
metadata:
  labels:
    app: altibase
  name: altibase-svc-node1     # 서비스 이름
spec:
  ports:
  - name: service-port
    port: 20300                # 보통 port와 targetport는 같은 값으로 설정
    targetPort: 20300
  - name: replication-port
    port: 20301                # 보통 port와 targetport는 같은 값으로 설정
    targetPort: 20301
  selector:
    app: altibase-node1        # 서비스가 요청을 전달할 파드

---

apiVersion: v1
kind: Service
metadata:
  labels:
    app: altibase
  name: altibase-svc-node2     # 서비스 이름
spec:
  ports:
  - name: service-port
    port: 20300                # 보통 port와 targetport는 같은 값으로 설정
    targetPort: 20300
  - name: replication-port
    port: 20301                # 보통 port와 targetport는 같은 값으로 설정
    targetPort: 20301
  selector:
    app: altibase-node2        # 서비스가 요청을 전달할 파드
```

3. 파드와 서비스 생성

```
$ kubectl create -f deploy.yaml
deployment.apps/altibase-deploy-vol-node1 created
deployment.apps/altibase-deploy-vol-node2 created

$ kubectl create -f service.yaml
service/altibase-svc-node1 created
service/altibase-svc-node2 created
```

4. 파드와 서비스 확인

```
$ kubectl get pod,service -o wide
NAME                                                READY   STATUS    RESTARTS   AGE     IP
              NODE        NOMINATED NODE   READINESS GATES
pod/altibase-deploy-vol-node1-74d75695c4-q7459   1/1     Running   0          3m12s
172.16.169.150   k8s-node2   <none>           <none>
pod/altibase-deploy-vol-node2-677b65857-x4hxm    1/1     Running   0          3m12s
172.16.107.208   k8s-node3   <none>           <none>

NAME                          TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)
  AGE     SELECTOR
service/altibase-svc-node1   ClusterIP   10.98.138.97   <none>        20300/TCP,20301/TCP
  73s     app=altibase-node1
service/altibase-svc-node2   ClusterIP   10.101.72.3    <none>        20300/TCP,20301/TCP
  73s     app=altibase-node2
service/kubernetes           ClusterIP   10.96.0.1      <none>        443/TCP
  4m32s   <none>
```

5. 이중화 객체 생성 및 구동

각각의 파드의 Altibase 컨테이너에서 서비스와 동일한 이름으로 이중화 객체 생성하고 구동한다.

	파드 altibase-node1의 Altibase 컨테이너	**파드 altibase-node2의 Altibase 컨테이너**
이중화 테이블 생성	CREATE TABLE t1 (c1 INTEGER PRIMARY KEY, c2 INTEGER);	CREATE TABLE t1 (c1 INTEGER PRIMARY KEY, c2 INTEGER);
이중화 객체 생성	CREATE REPLICATION rep1 WITH **'altibase-svc-node2'**, 20301 FROM sys.t1 TO sys.t1;	CREATE REPLICATION rep1 WITH **'altibase-svc-node1'**, 20301 FROM sys.t1 TO sys.t1;
이중화 시작	ALTER REPLICATION rep1 START;	ALTER REPLICATION rep1 START;

6. 이중화 동작 확인

양 서버에서 데이터를 입력 후 원격 서버에 반영된 데이터를 확인한다.

	파드 altibase-node1의 Altibase 컨테이너	파드 altibase-node2의 Altibase 컨테이너
altibase-node1에서 데이터 입력	iSQL> INSERT INTO t1 VALUES(1, 1); 1 row inserted.	
양 서버에서 데이터 확인	iSQL> SELECT * FROM t1; C1 C2 ---------- 1 1 1 row selected.	iSQL> SELECT * FROM t1; C1 C2 ---------- 1 1 1 row selected.
altibase-node2에서 데이터 입력		iSQL> INSERT INTO t1 VALUES (2, 2); 1 row inserted.
양 서버에서 데이터 확인	iSQL> SELECT * FROM t1; C1 C2 ---------- 1 1 2 2 2 rows selected.	iSQL> SELECT * FROM t1; C1 C2 ---------- 1 1 2 2 2 rows selected.