볼륨: 컨테이너에 디스크 스토리지 연결

볼륨 유형

1. emptyDir: 일시적인 데이터를 저장하는 데 사용되는 간단한 빈 디렉터리.
2. hostPath: 워커 노드의 파일시스템을 파드의 디렉터리로 마운트하는데 사용한다.
3. gitRepo: 깃 레포지터리의 콘텐츠를 체크아웃해 초기화한 볼륨.
4. nfs: NFS 공유를 파드에 마운트한다.
5. gcePersistentDisk, awsElasticBlockStor, azureDisk: 클라우드 제공자의 전용 스토리지를 마운트하는 데 사용.
6. configMap, secret, downwardAPI: 쿠버네티스 리소스나 클러스터 정보를 파드에 노출하는 데 사용되는 특별한 유형의 볼륨.
7. persistentVolumeClaim: 사전에 혹은 동적으로 프로비저닝된 퍼시스턴트 스토리지를 사용하는 방법.

---

emptyDir 볼륨

- 볼륨의 라이프사이클이 파드에 묶여 있으므로 파드가 삭제되면 볼륨의 콘텐츠는 사라진다.
- 동일 파드에서 실행 중인 컨테이너 간 파일을 공유할 때 유용하다.
- 단일 컨테이너에서도 임시 데이터를 디스크에 쓰는 목적인 경우 사용 가능하다.

```
# 동일한 볼륨을 공유하는 컨테이너 두 개가 있는 파드: fortune-pod.yaml

apiVersion: v1
kind: Pod
metadata:
  name: fortune
spec:

  containers:
```

```
  - image: luksa/fortune
    name: html-generator
    volumeMounts:
    # html이라는 볼륨을 컨테이너의 /var/htdocs에 마운트한다.
    - name: html
      mountPath: /var/htdocs
  - image: nginx:alpine
    name: web-server
    volumeMounts:
    # 위와 동일한 볼륨을 /usr/share/ngingx/html에 읽기 전용으로 마운트한다.
    - name: html
      mountPath: /usr/share/nginx/html
      readOnly: true
    ports:
    - containerPort: 80
      protocol: TCP

  volumes:
  # html이라는 단일 emptyDir 볼륨을 위의 컨테이너 두 개에 마운트한다.
  - name: html
    emptyDir: {}
```

- 컨테이너와 볼륨이 같은 파드에서 구성됐더라도, 컨테이너에서 접근하려면 파드에서 **VolumeMount**를 컨테이너 스펙에 정의해야 한다.
- emptyDir을 디스크가 아닌 메모리를 사용하는 tmpfs 파일시스템으로 생성하도록 요청할 수 있다.

  ⇒ emptyDir의 medium을 Memory로 지정한다.

gitRepo 볼륨

- ~~현재 gitRepo는 deprecated되었다.~~
- 기본적으로 emptyDir이며 파드가 시작되면 깃 레포지터리를 복제하고 특정 리비전을 체크아웃해 데이터로 채운다.
- gitRepo 볼륨이 생성된 후에는 참조하는 레포지터리와 동기화되지 않는다.
  gitRepo에 변경을 푸시할 때마다 파드를 삭제해줘야 하거나 추가 프로세스를 실행하여 동기화할 수 있다.
  - 깃 동기화 프로세스를 사이드카 컨테이너에서 실행한다. (도커 허븐의 gitSync)

- 프라이빗 깃 레포지터리를 커테이너에 복제하려면 깃 동기화 사이드카 아니면 gitRepo 볼륨을 대신하는 다른 방법을 사용해야 한다.

emptyDir와 gitRepo 볼륨은 파드가 삭제되면 볼륨과 컨텐츠는 삭제된다.

---

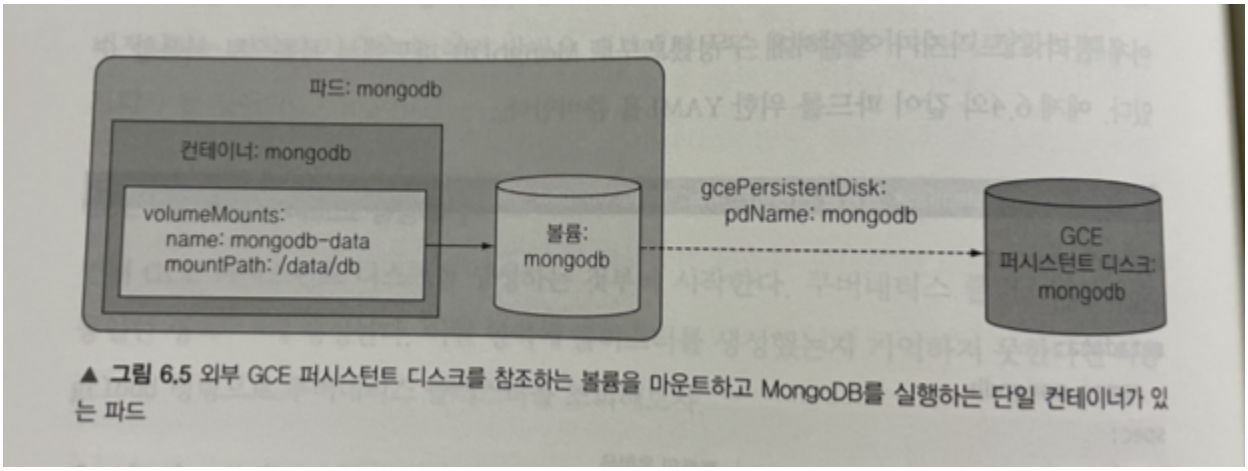
hostPath 볼륨

- 특정 시스템 레벨의 파드는 노드의 파일을 읽거나 파일시스템을 통해 노드 디바이스를 접근하기 위해 노드 파일시스템을 사용해야 한다.
- hostPath 볼륨은 노드 파일시스템의 특정 파일이나 디렉터리를 가리킨다.
- 퍼시스턴트 스토리지 유형 중 하나로, 파드가 삭제되는 경우 hostPath 볼륨의 컨텐츠는 삭제되지 않는다.
- 파드가 삭제되면 다음 파드가 호스트의 동일 경로를 가리키는 hostPath 볼륨을 사용하고, *이전 파드와 동일한 노드에 스케줄링된다는 조건*에서 새로운 파드는 이전 파드가 남긴 모든 항목을 볼 수 있다.
- **목적**
  일반적으로 hostPath 볼륨을 자체 데이터를 저장하기 위한 목적으로 사용하지 않는다.
  단지 노드 데이터에 접근하기 위해 사용하며, 대부분이 노드의 로그파일이나 kubeconfig(쿠버네티스 구성 파일), CA 인증서를 접근하기 위해 사용한다.
  여러 파드에 걸쳐 데이터를 유지하기 위해서는 절대 사용 금지.

퍼시스턴트 스토리지

- 파드에서 실행 중인 애플리케이션이 디스크에 데이터를 유지해야 하고
  파드가 다른 노드로 재스케줄링된 경우에도 동일한 데이터를 사용해야 한다면 emptyDir,

gitRepo, hostPath 볼륨을 사용할 수 없다.
이러한 데이터는 어떤 클러스터 노드에서도 접근이 필요하기 때문에 NAS(Network-Attatched Storage) 유형에 저장돼야 한다.

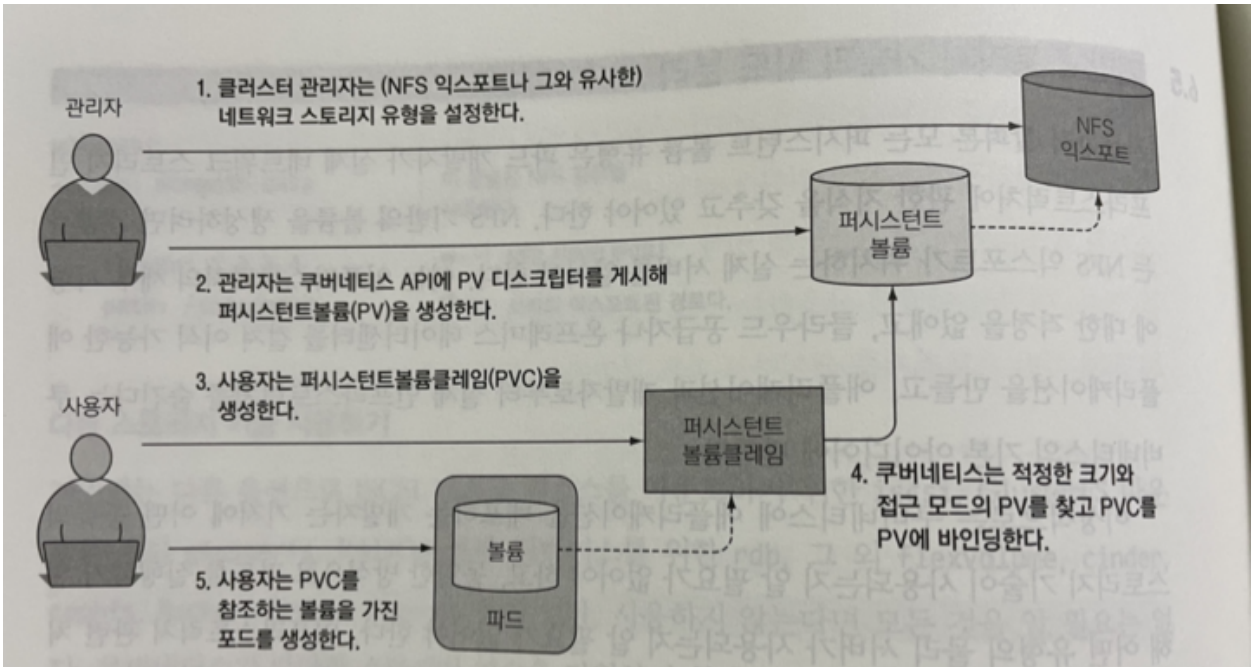


▲ 그림 6.5 외부 GCE 퍼시스턴트 디스크를 참조하는 볼륨을 마운트하고 MongoDB를 실행하는 단일 컨테이너가 있는 파드

- 위와 같은 구조에서, 파드를 삭제하고 재생성해도 데이터는 그대로 유지된다.
  이런 식으로 GCE 퍼시스턴트 디스크를 파드 인스턴스 여러 개에서 데이터를 유지하는 데 사용할 수 있다.

PersistentVolume & PersistentVolumeClaim

💡 이상적으로 쿠버네티스에 애플리케이션을 배포하는 개발자는
기저에 어떤 종류의 스토리지 기술이 사용되는지 알 필요가 없어야 하고,
동일한 방식으로 파드를 실행하기 위해 어떤 유형의 물리 서버가 사용되는지 알 필요가 없어야 한다.



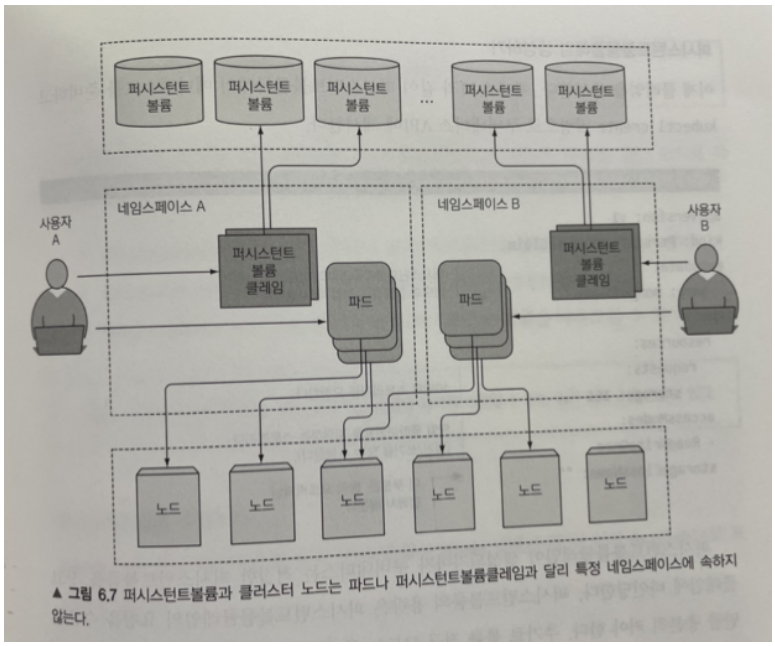
▲ **그림 6.6** 클러스터 관리자가 퍼시스턴트볼륨을 프로비저닝하면 파드는 퍼시스턴트볼륨클레임을 통해 이를 사용한다.

- 최소 크기와 필요한 접근 모드를 명시한 PersistentVolumeClaim 매니페스트를 생성한다.

  ▼ 접근모드

  - RWO(ReadWriteOnce): 단일 노드만이 읽기/쓰기용으로 볼륨을 마운트할 수 있다.
  - ROX(ReadOnlyMany): 다수 노드가 읽기용으로 볼륨을 마운트할 수 있다.
  - RWX(ReadWriteMany): 다수 노드가 읽기/쓰기용으로 볼륨을 마운트할 수 있다.
- PersistentVolumeClaim 매니페스트를 쿠버네티스 API 서버에 게시하면, 쿠버네티스는 적절한 PersistentVolume을 찾아 클레임을 볼륨에 바인딩한다.
- PersistentVolumeClaim의 바인딩을 삭제해 릴리즈될 때까지 다른 사용자는 동일한 퍼시스턴트 볼륨을 사용할 수 없다.
  - PersistentVolume의 STATUS가 Available일 때 바인딩할 수 있다.

- **PV와 네임스페이스**



퍼시스턴트 볼륨과 클러스터 노드는 파드나 퍼시스턴트볼륨클레임과 달리 특정 네임스페이스에 속하지 않는다.

▼ **persistentVolumeClaimPolicy**

- Retain: 클레임이 해제돼도 볼륨과 콘텐츠를 유지한다.
    - persistentVolumeClaimPolicy가 Retain인 경우, 파드와 PVC를 삭제한 후 기존 PV에 바인딩을 시도하면 실패한다.
      그 이유는, 이미 볼륨을 사용했기 때문에 데이터를 가지고 있으므로 클러스터 관리자가 볼륨을 완전히 비우지 않으면 새로운 클레임에 바인딩할 수 없기 때문이다.
- Delete: 기반 스토리지를 삭제한다.
- Recycle: 볼륨의 콘텐ㅌ츠를 삭제하고 볼륨이 다시 클레임될 수 있도록 볼륨을 사용 가능하게 만든다.
    - Recycle옵션은 현재 GCE 퍼시스턴트 디스크에서 사용할 수 없다.

PersistentVolume의 동적 프로비저닝

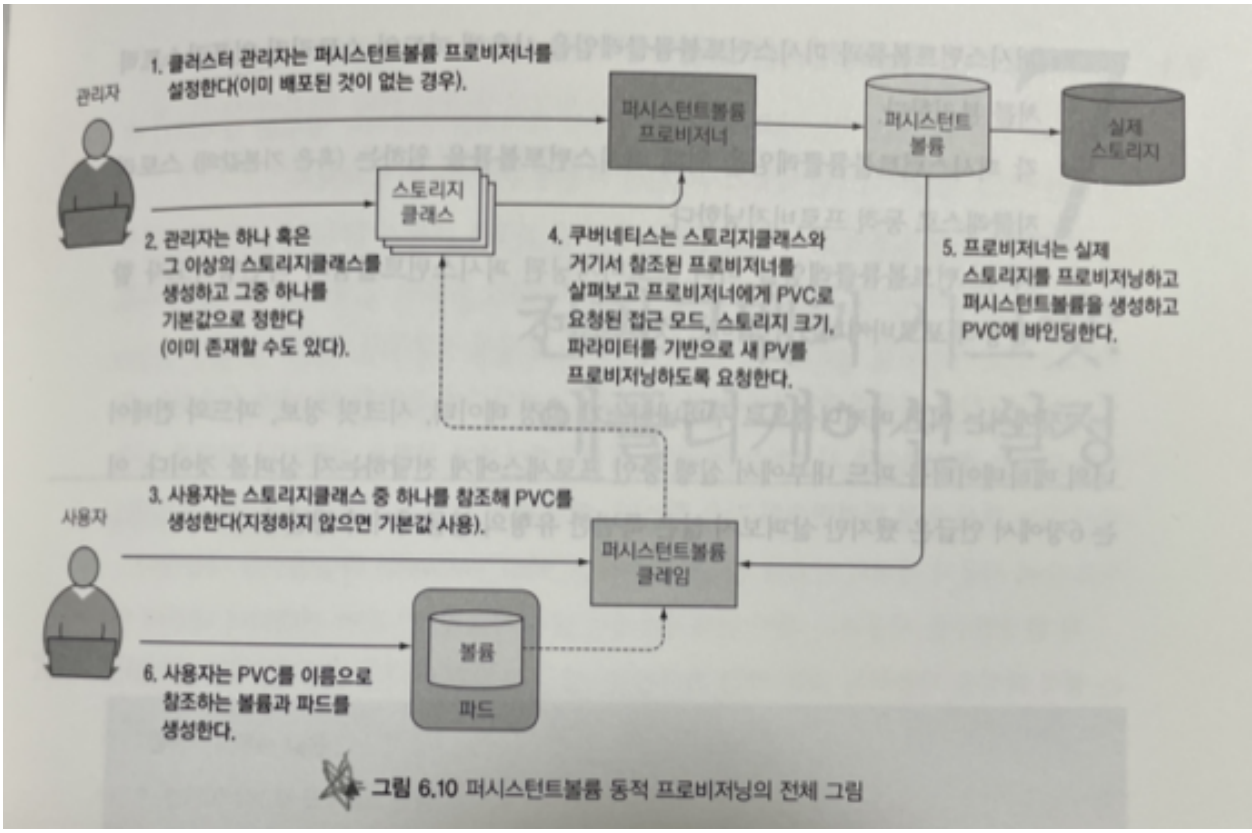


그림 6.10 퍼시스턴트볼륨 동적 프로비저닝의 전체 그림

- 관리자가 많은 PersistentVolume을 미리 프로비저닝하는 대신 하나 혹은 그 이상의 StroageClass를 정의하면 시스템은 누군가 PVC를 통해 요청 시 새로운 PV를 생성한다.

  ⇒ PersistentVolume가 부족할 일이 없다.

- PVC 정의에서 storageClassName 속성을 빈 문자열로 지정하면 PVC가 새로운 PV를 동적 프로비저닝하는 대신 미리 프로비저닝된 PV에 바인딩된다.