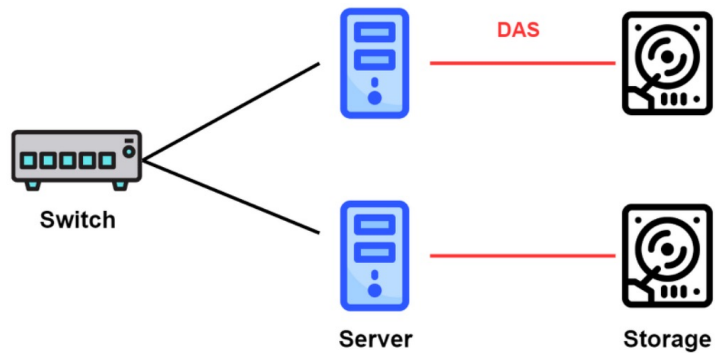


## **[Storage] Volumes and data**

# Storage 개요

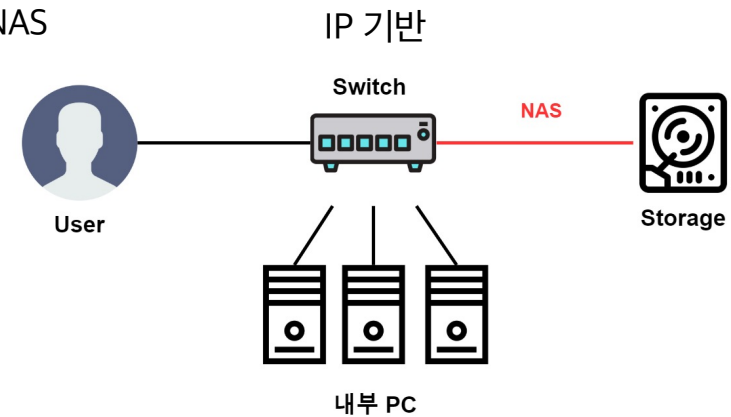
- 스토리지 종류

- DAS (직접 연결)

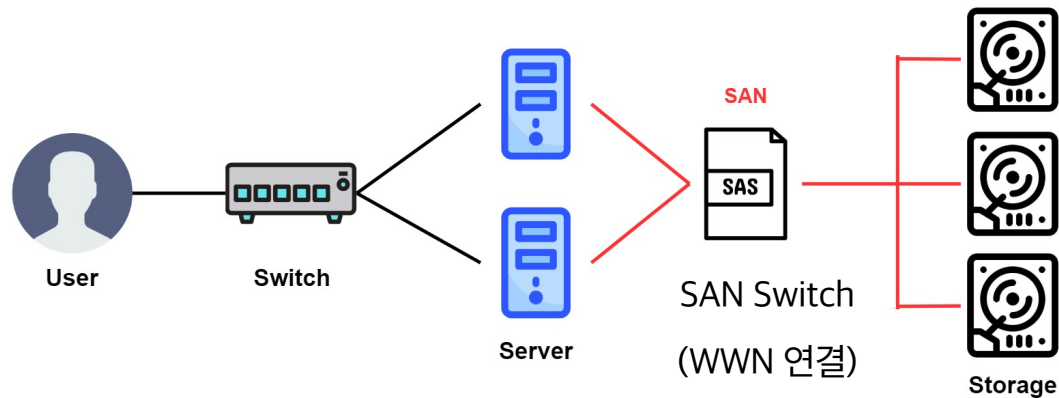


저렴 : Ceph, GlusterFS로 발전 꾀하는 중

- NAS



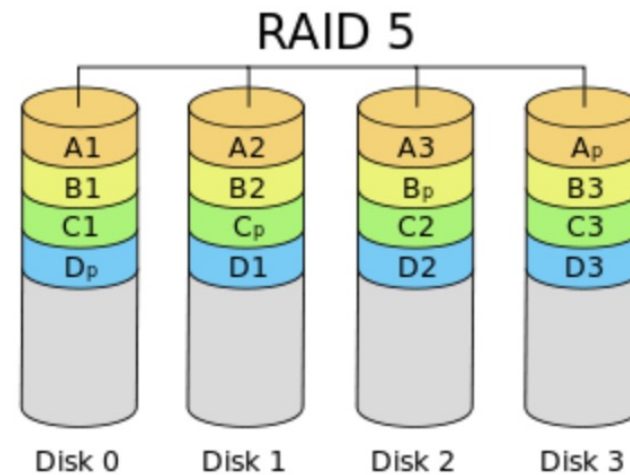
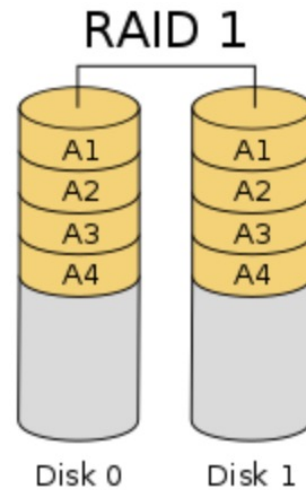
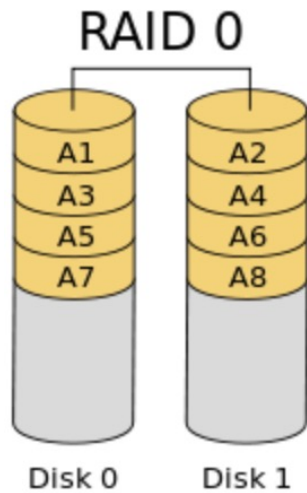
- SAN



- 가격 : DAS < NAS < SAN (비쌈)
- 성능 : NAS < DAS=SAN ( 좋음)
- 확장성 : DAS < SAN(보통 광) < NAS

# Storage 개요

- RAID 0 / 1 / 5 ( 그 외 RAID 2~4, 6도 있음 )
  - RAID 0 : 구매한 그대로 사용
  - RAID 1 : 구매한 절반 사용
  - RAID 5 : 패리티 비트 이용하여 데이터 복구 / 2개 이상 고장 시 복구 불가



# Volumes 소개


- 볼륨(volume) 은 하나의 파일 시스템을 갖춘 하나의 접근 가능한 스토리지 영역

물리 디스크	파티션	파일 시스템	드라이브 문자
하드 디스크 1	파티션 1	NTFS	C:
	파티션 2	FAT32	D:
하드 디스크 2	파티션 1	FAT32	E:

이 예에서

- "C:", "D:", "E:"는 볼륨이다.
- 하드 디스크 1과 하드 디스크 2는 물리 디스크이다.
- 이들 중 어떠한 것도 "드라이브"라 부를 수 있다.

```
[centos@osk-master-01 ~]$ sudo fdisk -l  
[centos@osk-master-01 ~]$ sudo cat /etc/fstab
```



파일 시스템은 컴퓨터에서 [파일](#)이나 [자료](#)를 쉽게 발견 및 접근할 수 있도록 보관 또는 조직하는 체계

파일 시스템은 통상 [하드 디스크](#)나 [CD-ROM](#) 같은 실제 자료 보관 장치를 사용하여 파일의 물리적 소재를 관리하는 것을 가리키나 네트워크 프로토콜([NFS](#), [SMB](#), [9P](#) 등)을 수행하는 클라이언트를 통하여 파일 서버 상의 자료로의 접근을 제공하는 방식과 가상의 형태로서 접근 수단만이 존재하는 방식([procfs](#) 등)도 파일 시스템의 범위에 포함

# Persistent Volume 소개(앞 설명 동일)

- Persistent Volume(PV) : 관리자가 프로비저닝하거나 스토리지 클래스를 사용하여 동적으로 프로비저닝한 클러스터의 스토리지
- Persistent Volume Claim(PVC) : 사용자의 스토리지에 대한 요청. 파드와 비슷.
  - ✓ 파드 : 노드 리소스를 사용 / POD 명세서 내 특정 수준의 리소스(CPU 및 메모리)를 요청
  - ✓ PVC : PV 리소스를 사용 / 클레임 명세서 내 특정 크기 및 접근 모드를 요청(예: ReadWriteOnce, ReadOnlyMany, ReadWriteMany)

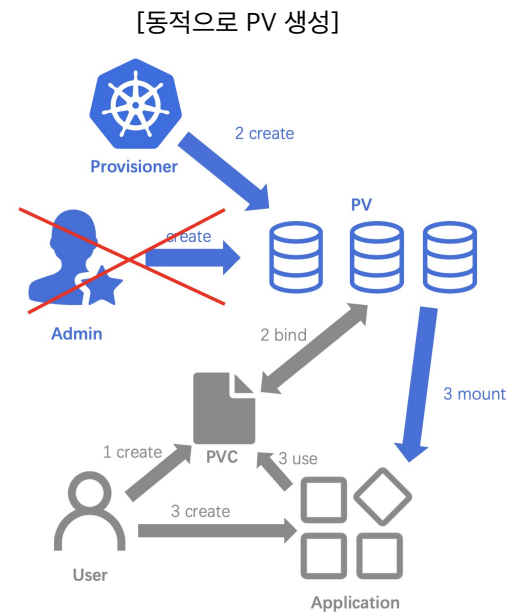
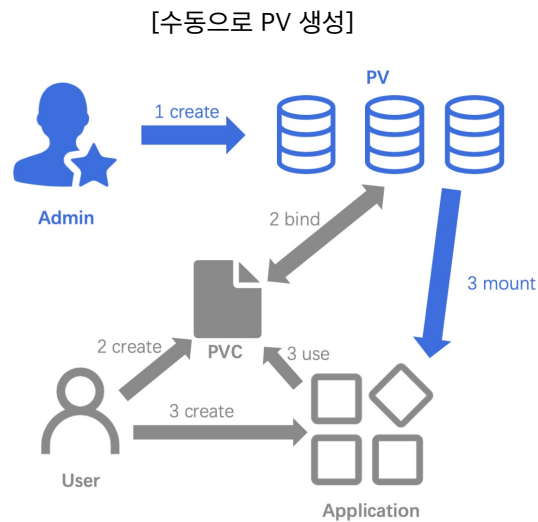


그림 출처 : <https://www.alibabacloud.com/blog/kubernetes-persistent-storage-process-1596505>

# Persistent Volume 소개

- PV 생성 : <https://kubernetes.io/ko/docs/concepts/storage/volumes/#hostpath>

```
[centos@osk-master-01 ~]$ cat hostpath.yaml
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
  - image: k8s.gcr.io/test-webserver
    name: test-container
    volumeMounts:
    - mountPath: /test-pd
      name: test-volume
  volumes:
  - name: test-volume
    hostPath:
      # 호스트의 디렉터리 위치
      path: /data
      # 이 필드는 선택 사항이다
      type: Directory
[centos@osk-master-01 ~]$ kubectl apply -f hostpath.yaml
pod/test-pd created
[centos@osk-master-01 ~]$ kubectl describe pod test-pd
Name:          test-pd (이후 생략)
Volumes:
  test-volume:
    Type:        HostPath (bare host directory volume)
    Path:        /data
    HostPathType: Directory
```

## Pod으로 Data(Volumes) 연결

```
[centos@osk-master-01 ~]$ kubectl get pod -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
pod	1/1	Running	0	64m	192.168.224.3	osk-worker-01.kr-central-1.c.internal	<none>	<none>
test-pd	1/1	Running	0	6s	192.168.183.131	osk-worker-02.kr-central-1.c.internal	<none>	<none>

```
[centos@osk-worker-02 ~]$ cd /data
```

```
[centos@osk-worker-02 data]$ sudo touch likelion
```

```
[centos@osk-worker-02 data]$ sudo ls  
likelion
```

```
[centos@osk-master-01 ~]$ kubectl delete pod test-pd  
pod "test-pd" delete
```

```
[centos@osk-worker-02 data]$ sudo ls  
likelion
```

```
[centos@osk-master-01 ~]$ kubectl apply -f hostpath.yaml  
pod/test-pd created
```

(결과 확인시 원활히 접속토록 공식홈페이지 yaml에서 이미지 nginx 등 수정바람)

```
[centos@osk-master-01 ~]$ kubectl get pod -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
pod	1/1	Running	0	17h	192.168.224.3	osk-worker-01.kr-central-1.c.internal	<none>	<none>
test-pd	0/1	ContainerCreating	0	9s	<none>	osk-worker-02.kr-central-1.c.internal	<none>	<none>

```
[centos@osk-master-01 ~]$
```

```
[centos@osk-master-01 ~]$ kubectl exec -it test-pd -- /bin/sh  
# cd /test-pd  
# ls  
likelion  
#
```

# ConfigMap

- 컨피그맵은 키-값 쌍으로 기밀이 아닌 데이터를 저장하는 데 사용하는 API 오브젝트
- 컨테이너에 필요한 환경 설정 내용을 컨테이너 내부가 아닌 외부에 분리하는데 용이
- 클러스터가 구성된 Config 방식을 이해하는 데 사용할 수도 있고, 클러스터를 업그레이드 할 때 활용할 수도 있음
- <https://kubernetes.io/docs/tasks/configure-pod-container/configure-pod-configmap/#configure-all-key-value-pairs-in-a-configmap-as-container-environment-variables>  
# kubectl create -f <https://kubernetes.io/examples/configmap/configmap-multikeys.yaml>  
# kubectl create -f <https://kubernetes.io/examples/pods/pod-configmap-envFrom.yaml>

```
[centos@osk-master-01 ~]$ kubectl create -f https://kubernetes.io/examples/configmap/configmap-multikeys.yaml
configmap/special-config created
[centos@osk-master-01 ~]$ kubectl create -f https://kubernetes.io/examples/pods/pod-configmap-envFrom.yaml
pod/dapi-test-pod created
[centos@osk-master-01 ~]$ kubectl logs dapi-test-pod
생략
SPECIAL_LEVEL=very
생략
SPECIAL_TYPE=charm
```