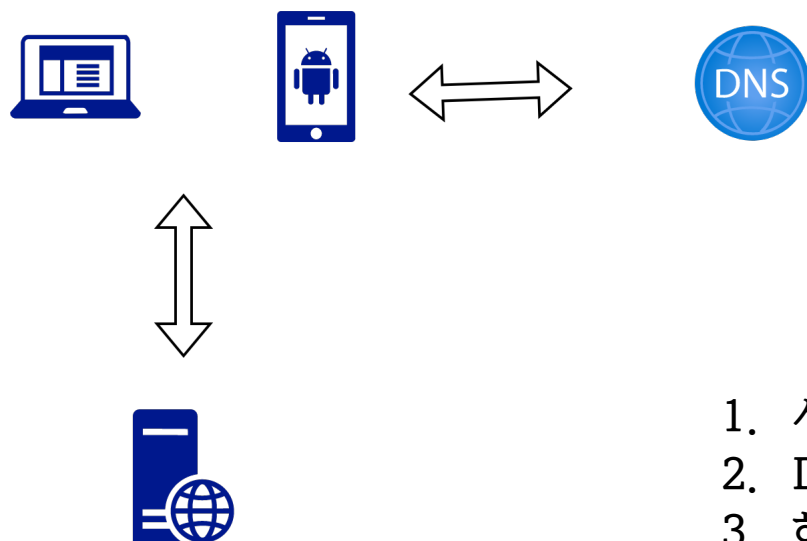


# 서버 시스템 설계하기

가상면접 사례로 배우는 대규모 시스템 설계 기초  
1장 내용 일부

# 단일 서버

Web App과 DB, Cache가 모두 하나의 서버에 있는 경우



1. 사용자는 도메인을 이용하여 사이트 접속
2. DNS(domain name service)를 이용해 IP 주소를 반환 받음
3. 해당 IP 주소로 웹서버에 접근

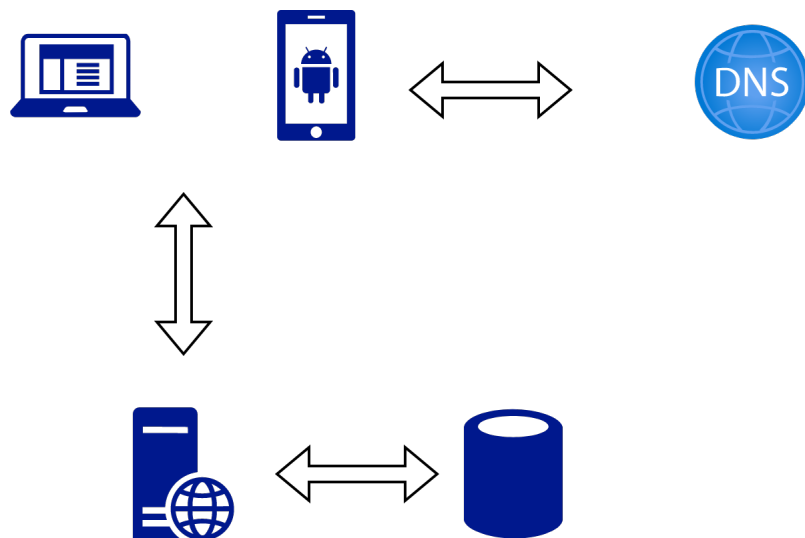


만약 서버에 사용자가 늘게 된다면...?

여러 서버를 구축 해야한다!

그중 DB 서버를 따로 분리하면

각각의 웹계층과 데이터계층을 독립적으로 확장이 가능하다!



DB의 종류는 현재 RDB(관계형)과 NoSQL(비관계형)이 존재한다.

그럼 어떤 DB를 사용해야 할까??

현재 내가 서비스하는 것과 잘 맞는 DB를 선택하는 것이다!

비관계형을 선택하는데 이점인 경우 :

- 낮은 응답 지연시간이 요구됨
- 다루는 데이터가 비정형이라 관계형이 아님
- 데이터를 직렬화하거나 역직렬화할 수 있기만하면됨
- 많은 양을 데이터 저장이 요구될 때



이렇게까지 했는데 서버에 무리가 오는 거 같을 때 어떡하지...?

서버 자체에 확장을 시도해보자!

수직적 규모 확장 vs 수평적 규모 확장

수직적 규모 확장

Scale Up

=>고사양 자원을 추가



수평적 규모 확장

Scale out

=> 더 많은 서버





이렇게까지 했는데 서버에 무리가 오는 거 같을 때 어떡하지...?

서버 자체에 확장을 시도해보자!

수직적 규모 확장 vs 수평적 규모 확장

수직적 규모 확장

Scale Up

=>고사양 자원을 추가



무한대로 증설 X  
자동복구, 다중화에 취약

수평적 규모 확장

Scale out

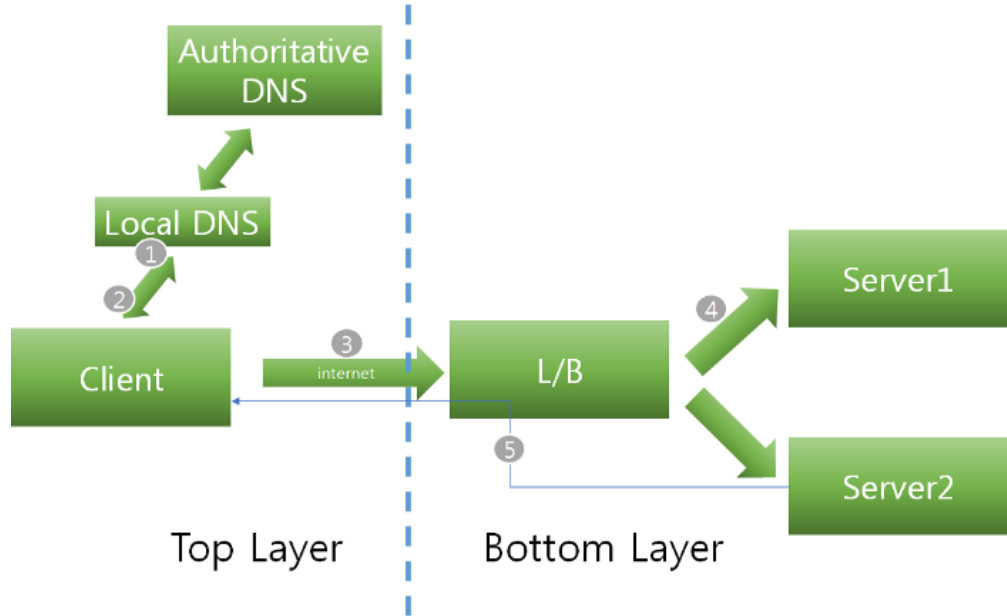
=> 더 많은 서버



대규모 어플리케이션에  
적합

# 로드밸런서

로드밸런서의 부하 집단에 속한 웹서버들에게 트래픽을 고르게 분산




1. Client 에서 DNS에 도메인의 ip주소를 요청
2. Ip 주소 반환
3. 반환받은 ip주소로 로드밸런서에 접근
4. 로드 밸런서에서 선별하여 웹서버에 접근
5. Client에 요청 반환

요기서 재미있는 점은  
로드밸런서에 접근하는 ip주소는  
공용 ip(Public Ip)주소이고  
로드밸런서에서 내부 서버에 접근하는 ip주소는  
내부ip(private ip)주소이다.

서버는 로드밸런서가 하고.. 그러면 DB는??  
많은 데이터 베이스 관리 시스템이 다중화를 지원

주   
main

쓰기 연산을 지원  
Insert / Update / Delete

부   
sub

읽기 연산을 지원

