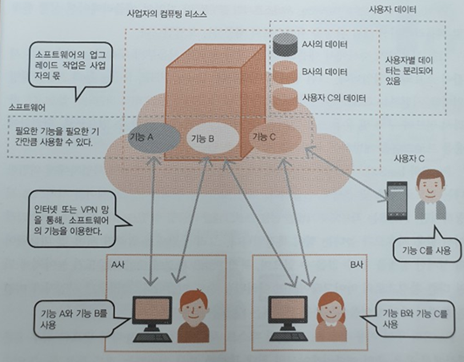
클라우드 개념 정리

**1. SaaS (Software as a Service)**  
- 서비스로서의 소프트웨어  
  
- 업무에서 사용하는 소프트웨어 기능을 인터넷 등의 네트워크를 통해 필요한 만큼 서비스로 이용할 수 있도록 제공하는 형태  
  
- 하나의 서버를 여러 기업에서 공유하는 것을 전제한 멀티 테넌트 방식. 그러나 데이터는 기업 사용자별로 분리되도록 설계하여 보안성 확보  
  
- 소프트웨어 업데이트 작업은 기업 사용자가 아니라 클라우드 사업자가 수행. 따라서 항상 최신 기능을 사용할 수 있으며 소프트웨어의 버그가 방치되지 않음  
  
- SaaS로 제공되는 대표적인 소프트웨어는 전자 메일, 그룹웨어, CRM(Customer Relationship Management, 고객 관리 시스템) 등이 있음  
  
- 대표적인 SaaS 서비스는 구글의 Google Apps



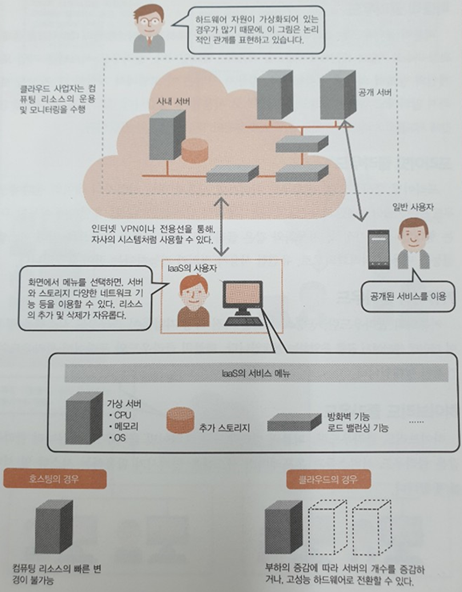
**2. PaaS (Platform as a Service)**  
- 서비스로서의 플랫폼  
  
- 기업의 애플리케이션 실행 환경 및 애플리케이션 개발 환경을 서비스로써 제공하는 모델  
  
- 기업 사용자가 자사에서 애플리케이션 개발 환경을 처음부터 구축하는 것은 많은 시간이 소요되는 일. 그런 점에서 PaaS에는 Java, PHP, Ruby 등의 프로그래밍 언어를 지원하는 애플리케이션 실행 환경이나 데이터베이스 등이 미리 마련되어 있음  
  
- IaaS와의 차이점은 서버, 네트워크, 보안 부분을 클라우드 사업자에게 위임한다는 점으로 구축 및 운영이 쉬움. 또한 SaaS는 제공되는 정해진 소프트웨어만을 쓰지만, PaaS는 자사에서 개발한 응용 프로그램을 가동할 수 있음  
  
- 그러나 반대로 말하면 서버 및 미들웨어의 상세한 설정을 할 수 없을 수 있고, 특정 PaaS 환경에 대한 의존도가 높아지게 되면 다른 환경으로의 마이그레이션이 어려워질 수도 있음  
  
- PaaS의 주된 용도는 개발 및 테스트 시행에 큰 처리 능력이 필요한 경우 또는 자사에서 운용 중인 애플리케이션의 최대 부하를 분산 처리하는 경우 등  
  
- 다양한 사물들이 인터넷을 통해 서로 통신하는 IoT에서 다양한 기기들이 생성하는 센서 데이터 같은 대용량 데이터를 효율적으로 수집하여 처리하는 플랫폼으로도 주목받고 있음  
  
- 대표적 서비스로는 세일즈 포스의 Force.com, 사이보우즈의 kintone, 오픈 소스 PaaS 기반 소프트웨어인 Cloud Foundry



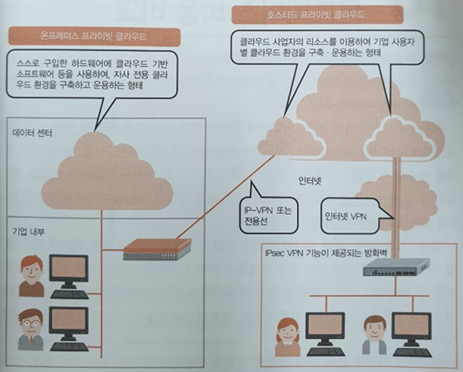
테이블이(가) 표시된 사진

자동 생성된 설명

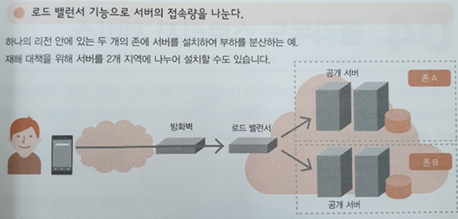
**3. IaaS (Infrastructure as a Service)**  
- 서비스로서의 인프라  
  
- CPU나 하드웨어 등의 컴퓨팅 리소스를 네트워크를 통해 서비스로 제공  
  
- 사용자는 하드웨어를 보유하지 않고 서버와 스토리지, 네트워크 등의 리소스와 기능을 사용할 수 있으며, 언제든지 신속하게 자원을 추가 및 제거할 수 있음  
  
- IaaS 서비스 예로는 가상 서버 및 온라인 스토리지 등  
  
- 가상 서버란 클라우드 사업자가 보유하고 있는 물리적 서버의 CPU, 메모리, 스토리지 등 하드웨어 자원을 소프트웨어적으로 나누어 사용자에게 제공하는 것. 가상 서버에서는 OS에 설치되는 데이터베이스와 미들웨어, 응용 프로그램 등의 소프트웨어를 자유롭게 운영할 수 있지만, 기업 사용자가 스스로 설치하고 관리해야 함  
  
- 대표적인 서비스로는 Amazon Web Services의 Amazon Elastic Compute Cloud(EC2)



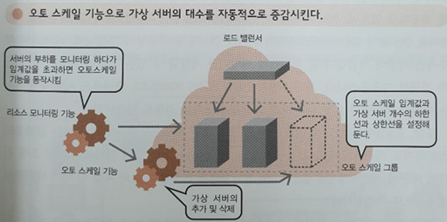
**1. 클라우드 이용 모델**  
(1) 퍼블릭 클라우드  
- 클라우드 사업자가 시스템을 구축하고, 인터넷망 등의 네트워크를 통해 불특정 다수의 기업과 개인에게 서비스를 제공하는 형태  
  
- 클라우드 시스템은 기업 또는 개인의 방화벽 외부에 구축됨  
  
- 사용자 기업은 이 모델에서 자사의 IT 자산을 보유하지 않더라도 컴퓨팅 리소스를 서비스로 사용할 수 있음  
   
  
(2) 프라이빗 클라우드  
- 사용자의 데이터 센터에 클라우드 관련 기술이 활용된 자사 전용 환경을 구축하여 컴퓨팅 리소스를 유연하게 이용할 수 있는 형태  
  
- 가상화, 자동화와 같은 클라우드 관련 기술의 활용으로 시스템의 성능과 비용이 최적화되고 유연한 사용자 정의가 가능  
   
  
(3) 커뮤니티 클라우드  
- 공통의 목적을 가진 특정 기업들이 클라우드 시스템을 형성하여 데이터 센터에서 공동 운영하는 형태  
  
- 퍼블릭 클라우드와 프라이빗 클라우드의 중간적 성격  
   
  
(4) 하이브리드 클라우드  
- 퍼블릭 클라우드와 프라이빗 클라우드, 커뮤니티 클라우드 같은 클라우드 서비스들과 온프레미스 시스템을 연계시켜 활용하는 시스템 및 서비스  
   
   
**2. 프라이빗 클라우드 종류**  
(1) 온프레미스 프라이빗 클라우드  
- 기업 사용자 스스로 클라우드 기반 소프트웨어 등을 이용하고, 자체적으로 구입한 서버 및 스토리지, 하드웨어 리소스를 기업 안에 설치하여 자사 전용 클라우드 환경을 구축 및 운용하는 형태  
  
- 기업 사용자 요구 사항에 맞춘 유연한 시스템 설계가 가능  
  
- 자체적인 보안 정책에 따른 강력한 보안 환경을 구축할 수 있음  
  
- 스스로 클라우드 환경을 구축해야 하기 때문에 부담이 있을 수 있음. 그러나 대규모로 클라우드 환경을 구축할 경우, 자체 프라이빗 클라우드가 더 저렴한 비용으로 유연한 환경을 제공할 수 있음  
   
  
(2) 호스티드 프라이빗 클라우드  
- 클라우드 사업자가 기업 사용자별로 클라우드 환경을 제공하여, 서버와 스토리지 같은 컴퓨팅 리소스를 서비스로 제공하는 형태  
  
- 기업 사용자는 장비를 구입하거나 사내 시스템을 구축할 필요가 없으므로 단기간에 전용 클라우드 환경을 구축하여 이용할 수 있음  
  
- 호스티드 사설 클라우드는 전용선과 VPN망을 통해 보안 수준이 높은 전용 클라우드 환경을 제공  
  
\*VPN(Virtual Private Network): 가상 사설망으로 인터넷망과 같은 공중망을 사설망처럼 이용할 수 있는 기업통신 서비스



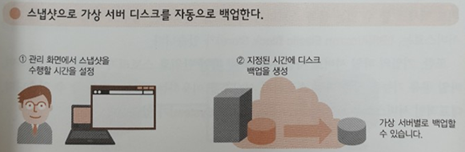
**3. 가상 서버**  
- 클라우드 서비스에 있어서 가장 기본적인 서비스로, 하나의 물리적 서버를 논리적으로 나누어 CPU나 메모리 등의 하드웨어 자원을 소프트웨어로 나누어 사용하는 것  
  
- 생성된 가상 서버는 실제 서버와 마찬가지로 OS 및 애플리케이션을 동작시킬 수 있음. 그러나 하드웨어를 에뮬레이션하기 때문에 가상화하지 않은 서버에 비해 성능이 떨어짐  
  
\*에뮬레이션(emulation): 어떤 계산기가 다른 종류의 계산기용으로 작성된 프로그램을 특별한 기구와 프로그래밍 기법을 이용하여 그대로 실행할 수 있도록 하는 것. 디지털 정보를 생산한 시점에서 사용된 하드웨어, 매체, 운영 체제, 소프트웨어의 운용을 그대로 흉내 내어(emulate) 그 내용을 읽어내는 프로그램을 통하여 재현하는 보존 전략  
  
- 가상 서버는 클라우드 서비스가 제공하는 로드 밸런스 기능을 사용하여 이중화 및 부하 분산 가능  
  
- 또한, 오토 스케일 기능을 통해 가상 서버에 접속이 집중될 때는 통신량에 따라 자동으로 가상 서버의 대수를 늘리거나 줄일 수 있음  
  
- 대표적으로 AWS(Amazon Web Services)의 EC2(Elastic Compute Cloud)  
   
   
**4. 가상 서버 옵션 기능**  
(1) 로드 밸런서  
- 대다수의 클라우드 서비스에서 가상 서버 사용 시 ‘리전(Region)’과 ‘존(Zone)’을 선택할 수 있음. 리전이란 서울, 미국 등 지리적으로 떨어진 독립적인 지역을 뜻함. 존은 같은 지역 내의 독립적인 로케이션(빌딩 등 물리적으로 격리된)을 의미  
  
- 리전에는 서로 다른 로케이션의 존이 설치되어 있으므로 여러 존을 선택하여 시스템의 이중화 및 부하의 분산처리가 가능  
  
- 로드 밸런서는 외부의 접속을 통신 데이터양 등의 조건에 따라 가상 서버에 전달. 이렇게 하면 통신의 부하를 분산하여 대량의 접속에도 대응할 수 있는 시스템을 손쉽게 구축



(2) 오토 스케일  
- 접속이 집중되는 때와 같이 외부의 통신 데이터양에 따른 부하에 따라 자동으로 가상 서버의 대수를 증감시키는 기능  
  
- 접속량이 대량으로 증가할 때 서버의 대수를 늘려서 대응. 접속량이 줄어들면 서버의 대수를 줄임으로써 불필요한 비용을 절감



(3) 스냅샷  
- 정기적인 스냅샷(백업 기능)은 수동 또는 이용자가 설정한 시간에 가상 서버 디스크의 백업을 자동으로 생성하여 복제할 수 있게 함



**5. 클라우드 스토리지 서비스**  
- 데이터 아카이브(보관) 및 백업(보호), 파일 서버, 재해 대책 등의 용도로 이용되는 서비스. 최근에는 빅데이터 분석 기반 구축에도 활용  
  
- 대표적인 서비스로는 AWS의 S3(Amazon Simple Storage Service). S3는 데이터를 파일 단위로 저장, 검색, 삭제하는 스토리지로 ‘객체 스토리지’라고 부름  
  
- 많은 클라우드 사업자들이 S3와 비슷한 규모의 스토리지 서비스를 제공하고 있으며, 여러 개의 데이터 센터에 데이터를 분산 저장함으로써 데이터 손실을 막음  
  
- 1년 동안 99.99% 이상의 뛰어난 견고성과 내구성이 보장되며, 데이터의 영구적인 저장에 적합. 또한 무제한의 데이터를 저장할 수 있는 용량이 제공되며 저장과 검색이 자유로움  
  
- AWS의 Amazon Glacier 클라우드 스토리지는 S3에 비해 비용을 크게 줄인 서비스로, 데이터의 읽기에 시간이 오래 걸리기 때문에 빈번하게 읽을 필요가 없는 장기 보존용 디지털 정보의 저장 및 자기 테이프의 대체품으로 이용  
  
- S3 등의 오브젝트(객체) 스토리지 서비스는 데이터의 읽기 및 쓰기 속도가 빠른 편이 아님. 따라서 엄격한 응답 시간이 요구되는 데이터베이스 용도로 이용할 경우, 블록 단위로 액세스 할 수 있는 블록 스토리지인 EBS(Amazon Elastic Block Store)가 있음  
  
- 기업의 파일 서버로 사용하는 경우, 파일 단위로 스토리지 액세스가 가능하며 파일 공유 기능을 갖춘 파일 스토리지 서비스인 EFS(Amazon Elastic File System)가 있음  
  
\*파일 서버(File Server): 구내 정보 통신망(LAN)에서 망의 모든 사용자가 접근할 수 있는 파일 기억 장치. LAN에서 원격 디스크 구동 장치로서의 기능을 하는 디스크 서버와는 달리, 파일 서버는 파일을 격납할 뿐만 아니라 파일을 관리하고 사용자들의 파일 요구와 파일 내용 변경의 질서를 유지하는 등 고도의 기능을 수행하는 장치

**1. 클라우드 네트워크 서비스**

**​**

- 클라우드 위에 네트워크를 구축. 대표적인 예로는 Amazon VPC(Virtual Private Cloud)

​

- Amazon VPC는 가상 프라이빗 클라우드라는 이름 그대로 AWS 위에 가상 네트워크를 만들어 개인 클라우드처럼 사용할 수 있는 서비스

​

- VPC 안에는 임의의 사설 IP 주소 범위를 설정하고, 서브넷에 가상 서버와 같은 리소스를 배치할 수 있음

​

- 클라우드 서비스는 인터넷으로 접속해야 하는데, 회사 내부용 서비스를 인터넷을 통해 이용한다는 점에서 보안상의 문제가 있음. Amazon VPC에서는 사설 IP 주소가 할당된 네트워크에 가상 VPN 게이트웨이를 마련하여 사내 네트워크 쪽 VPN 게이트웨이와 Ipsec을 통한 암호화 통신을 함. 이를 통해 거점 간 인터넷 VPN을 구축하여 클라우드 서비스를 안전하게 이용할 수 있음

​

\*VPN(Virtual Private Network): 가상 사설망으로 인터넷망과 같은 공중망을 사설망처럼 이용할 수 있는 기업통신 서비스

​

\*Ipsec(IP security protocol): 안전에 취약한 인터넷에서 안전한 통신을 실현하는 통신 규약

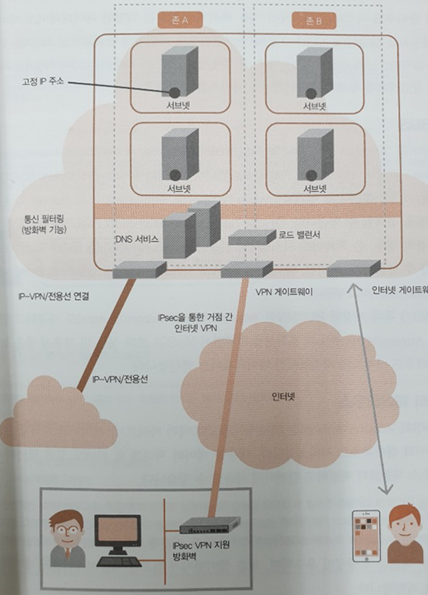
- 인터넷 VPN 외에도 NTT 커뮤니케이션즈 등의 통신 사업자가 제공하는 VPN망(IP-VPN)과 접속할 수 있는 서비스도 있음. 인터넷을 거치지 않고 통신 사업자의 VPN망을 통해 직접 연결하므로 더욱 안전한 네트워크 환경 이용 가능

​

- Amazon Route 53으로 대표되는 DNS 서비스는 자체 도메인 네임으로 웹이나 메일을 이용할 때 필요한 공개용 DNS 서버를 운영하고 관리. 존 정보와 레코드 정보를 설정할 수 있으므로 로드 밸런서 같은 서비스와 연계시켜 가용성이 높은 시스템 구축 가능

​

\*DNS 서버(Domain Name System Server): 도메인 이름에서 IP 주소를 추출하는 역할을 하는 서버. DNS 서버는 분산형 데이터 베이스로, 어떤 DNS 서버에서 IP 주소를 알지 못하면 상위의 DNS 서버를 검색. 인터넷을 경유하여 메일을 주고받기 위해서는 메일 서버 이름을 DNS 서버에 등록해야 함

[](https://m.blog.naver.com/shakey7/221398503566)

**​**

**​**

**​**

**​**

**2. 클라우드 데이터베이스 서비스**

**​**

(1) RDBMS(Relational Database Management System: 관계형 데이터베이스 관리 시스템)

​

- AWS의 Amazon RDS 서비스는 데이터베이스 엔진으로는 MySQL, PostgreSQL 같은 무료 엔진과 Oracle Database, Microsoft SQL Server 같은 유료 엔진을 선택할 수 있음

​

\*데이터베이스: 체계화된 데이터의 모임. 즉, 한 조직 내 여러 응용 시스템이 공유할 수 있도록 통합 및 저장된 운영 데이터의 집합

​

\*관계형 데이터베이스: 키(key)와 값(value)들의 간단한 관계를 테이블화 시킨 매우 간단한 원칙의 전산정보 데이터베이스. 이 모델은 데이터를 컬럼(column)과 로우(row)를 이루는 하나 이상의 테이블(또는 관계)로 정리하며, 고유 키(Primary key)가 각 로우를 식별. 로우는 레코드나 튜플로 부름. 일반적으로 각 테이블(관계)은 하나의 엔티티 타입(고객이나 제품과 같은)을 대표. 로우는 그 엔티티 종류의 인스턴스(예: 홍길동 or 3000원)를 대표하며 컬럼은 그 인스턴스의 속성이 되는 값들(예: 이름, 가격)을 대표

​

\*데이터베이스 엔진(Database Engine): 데이터베이스 관리 시스템을 구성하는 핵심 부분으로, 데이터가 저장되어 있는 복수의 데이터베이스 파일에 실제로 접근하여 검색, 삭제, 갱신 등의 기능을 담당하는 소프트웨어 모듈

​

※데이터베이스 엔진

화면 표시나 형식의 작성, 인쇄 등의 기능은 DBMS의 다른 부분이 담당. 통상적으로 일반 개인용 컴퓨터 사용자용 또는 단일 사용자용 DBMS에서는 이 두 부분이 일체화되어 있기 때문에 구별되는 일은 없음. 따라서 디베이스(dBASE), 패러독스, 액세스(Access)와 같은 PC용 DBMS를 데이터베이스 엔진과 같은 의미로 사용. 한편, 기업체나 단체 등의 조직 전체 규모의 대형 데이터베이스에 사용되는 클라이언트/서버형의 DBMS에서는 화면 표시나 인쇄를 행하는 부분은 클라이언트 측 PC에서 동작하고, 데이터베이스의 검색이나 관리를 행하는 부분은 서버 측에서 동작. 때문에 데이터베이스 파일을 직접 조작하는 기능의 소프트웨어를 데이터베이스 엔진이라는 용어로 부름

\*SQL(Structured Query Language 구조화 질의 언어): 관계형 데이터베이스 관리 시스템(RDBMS)의 데이터를 관리하기 위해 설계된 특수 목적의 프로그래밍 언어. 관계형 데이터베이스 관리 시스템에서 자료의 검색과 관리, 데이터베이스 스키마 생성과 수정, 데이터베이스 객체 접근 조정 관리를 위해 고안되었음. 많은 데이터베이스 관련 프로그램들이 SQL을 표준으로 채택

- 유료 제품의 경우, 온 프레미스 시스템에서 사용하고 있던 라이선스를 클라우드 서비스로 반입할 수 있음. 이처럼 라이선스를 클라우드 서비스에 반입하는 것을 BYOL(Bring Your Own License)이라고 부름

​

- AWS가 독자 개발한 RDBMS인 Amazon RDS for Aurora도 있음. Aurora는 MySQL과 호환성을 가지면서 MySQL보다 성능 및 가용성 등을 높이고 클라우드 서비스에 최적화시킨 데이터베이스 엔진

​

\*MySQL: SQL을 사용하는 오픈 소스의 관계형 데이터베이스 관리 시스템(RDBMS). 매우 빠르고 유연하며 사용하기 쉬운 특징. 다중 사용자, 다중 스레드(thread)를 지원하고, C, C++, 에펠(Eiffel), 자바, 펄, PHP, 파이선(Python) 스크립트 등을 위한 응용 프로그램 인터페이스(API)를 제공. 램프(LAMP), 즉 리눅스 운영 체계와 아파치(Apache) 서버 프로그램, MySQL, PHP 스크립트 언어 구성은 상호 연동이 잘 되면서도 오픈 소스로 개발되는 무료 프로그램이어서 홈 페이지나 쇼핑몰 등 일반적인 웹 개발에 널리 이용되고 있음

​

​

​

(2) 데이터 웨어하우스

- AWS의 Amazon Redshift는 페타바이트급 데이터를 처리할 수 있는 데이터 웨어하우스 서비스로, 대용량 데이터 처리를 할 때에도 일반 데이터 웨어하우스 제품보다 저렴하고 간편하게 사용할 수 있음

​

​

​

(3) NoSQL

- AWS의 Amazon DynamoDB는 AWS가 독자 개발한 NoSQL 데이터베이스 서비스로, 빠르고 확장성이 우수하며 낮은 레이턴시(latency)가 필요한 애플리케이션의 데이터베이스로 적합

​

\*NoSQL(Not only SQL): 단순히 기존 관계형 DBMS가 갖고 있는 특성뿐만 아니라, 다른 특성들을 부가적으로 지원하는 데이터베이스 관리 시스템. 기존의 관계형 데이터베이스보다 더 융통성 있는 데이터 모델을 사용하고, 데이터의 저장 및 검색을 위한 특화된 매커니즘을 제공. 이를 통해 NoSQL 데이터베이스는 단순 검색 및 추가 작업에 있어서 매우 최적화된 키, 값 저장 기법을 사용. 응답속도나 처리 효율 등에 매우 뛰어난 성능을 발휘

※ NoSQL

NoSQL이 등장한 이후에도 기업의 ERP나 MIS 시스템 등 데이터의 정확한 처리가 필수적인 시스템에서는 관계형 데이터베이스를 사용. SQL이라고 하는 데이터 처리 언어의 편의성 때문에 NoSQL 등 다른 데이터베이스 시스템들은 많이 활용되지 않았음. 그러나 2000년 후반 SNS 등이 등장하면서 관계형 데이터나 정형데이터가 아닌 데이터, 즉 비정형 데이터를 보다 쉽게 저장하고 처리하는데 관심이 쏠림. 따라서 현재 NoSQL 데이터베이스가 각광을 받게 됨

[텍스트, 영수증이(가) 표시된 사진

자동 생성된 설명](https://m.blog.naver.com/shakey7/221398503566)

**​**

**​**

**​**

**​**

**3. 클라우드 빅데이터 분석 서비스**

**​**

- 아래 그림을 참고하여, 전국에 여러 점포가 있는 기업에서 클라우드 빅데이터 서비스를 이용한다면, 각 매장과 자사의 데이터 센터 및 클라우드 서비스는 VPN망 등의 네트워크로 연결. 자사의 데이터 센터에서 BI(Business Intelligence) 시스템을 운영. 빅데이터 분석은 클라우드 서비스로 처리

​

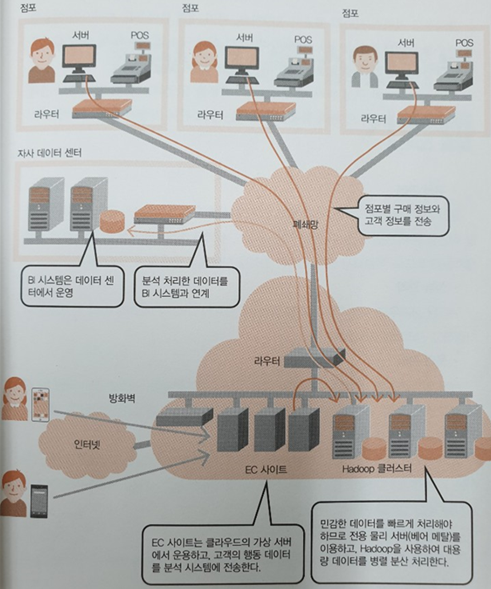
- EC 사이트는 가상 서버로 구축하고 운용. 한편, 매장에서 생성되는 방대한 데이터의 분석은 고속으로 처리해야 하고, 판매 데이터의 경우는 민감한 정보이므로 물리 서버(베어 메탈)를 이용. 물리 서버에는 하둡을 구현하여 하둡이 밤 동안 일괄적으로 데이터를 분석하고 처리

​

- 분석 처리한 데이터를 BI 시스템과 연계하여 판매 패턴을 분석. 그 결과를 바탕으로 제품의 수요를 예측하고 생산 계획에 반영

​

- 대표적인 소프트웨어로는 다수의 컴퓨팅 리소스를 활용하여 빅데이터의 효율적인 병렬 분산 처리를 전문으로 하는 Apache Hadoop과 실시간 분석을 전문으로 하는 Apache Spark가 있음

[](https://m.blog.naver.com/shakey7/221398503566)

​

**1. 클라우드를 실현하는 기술**

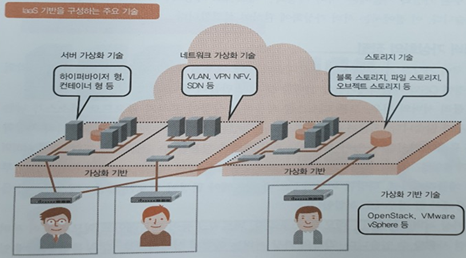
**​**

- 클라우드 서비스의 특징은 다양한 가상화 및 분산 처리 기술들로 이루어짐

​

- IaaS 구성 주요 기술: 가상화, 네트워크 가상화, 스토리지 기술 등

​

[](https://m.blog.naver.com/shakey7/221408055973)

​

​

​

**2. IaaS 기반을 구성하는 주요 기술**

**​**

(1) 가상화

- 가상화는 하나의 물리적 서버 리소스에 여러 개의 서버 환경을 할당하고, 각각의 환경에 OS와 애플리케이션을 실행할 수 있게 만들어 주는 기술

​

- 물리 서버의 경우, CPU나 메모리 등의 모든 리소스를 사용하는 상황이 흔치 않기 때문에 리소스가 남는 경우가 많음. 개별적으로 운영했던 많은 양의 물리 서버들을 서버 가상화를 통해 집약시키면 남는 리소스를 최대한 활용할 수 있음

​

- 물리 서버에서 실행 중인 시스템을 가상 서버로 옮기거나 가상 서버에서 실행 중인 시스템을 다른 가상 서버로 옮길 수 있고, 다양한 버전의 OS 및 미들웨어 환경이 필요한 개발 또는 테스트에도 적합

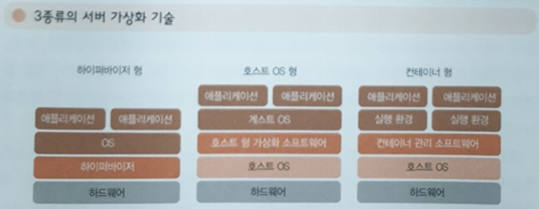
​

- 각각의 가상 서버는 독립된 상태이므로 같은 물리 서버에 있는 가상 서버 중 하나가 바이러스와 같은 위협에 노출되더라도 다른 가상 서버에는 영향을 미치지 않음

​

- 서버 가상화 기술은 주로 하이퍼바이저 형, 호스트 OS 형, 컨테이너 형 3종류로 분류됨

​

[](https://m.blog.naver.com/shakey7/221408055973)

​

​

1) 하이퍼바이저 형

- 하나의 물리 서버 하드웨어 위에 하이퍼바이저라는 가상화 소프트웨어를 동작시키고, 그 위에 리눅스나 윈도우 같은 여러 게스트 OS를 가동시키는 형태. 클라우드 사업자가 제공하는 클라우드 서비스는 하이퍼바이저 형 가상 서버에서 제공되는 것이 일반적

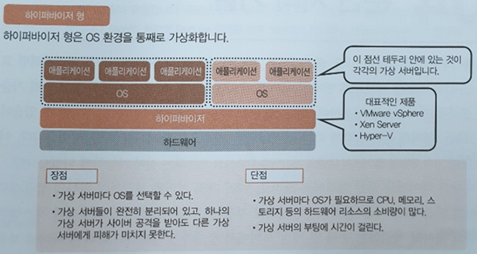
​

- 여러 개로 나누어진 각 가상 서버의 처리 능력은 물리 서버보다 떨어짐

​

- VM웨어의 VMware vSphere, MS의 Hyper-V, 시트릭스의 Xen, 리눅스 표준 기능인 KVM이 대표적

​

[](https://m.blog.naver.com/shakey7/221408055973)

​

​

2) 컨테이너 형

- 하나의 OS 환경에서 애플리케이션을 실행하기 위한 영역(사용자 공간)을 여러 개로 나누어 사용하는 것으로 최근 주목 받고 있는 기술

​

- 각각의 컨테이너는 다른 컨테이너에 영향을 미치지 않고 애플리케이션을 실행시킬 수 있음. 컨테이너는 호스트 OS 입장에서 볼 때, 하나의 프로세스(실행 중인 프로그램)

​

- 서버 가상화가 하드웨어 환경을 통째로 가상화한다면, 컨테이너는 애플리케이션 실행 환경을 가상화한다고 볼 수 있음. 다양한 OS를 지원하는 템플릿을 사용하여 컨테이너를 만들 수 있고, 이를 활용하여 하나의 호스트 OS 위에 멀티 OS 환경을 구현할 수 있음

​

- 가상 서버를 시작하는 데 수십 초에서 몇 분 정도가 걸리는 반면, 컨테이너는 가상화 오버헤드가 적기 때문에 빠르게 시작하고 정지할 수 있으며 성능 저하가 거의 없음

​

\*오버헤드(overhead): 특정한 목표를 달성하기 위해 간접적 혹은 추가적으로 요구되는 시간, 메모리, 대역폭 혹은 다른 컴퓨터 자원

​

- 게스트 OS를 마련할 필요가 없어 디스크 사용량을 줄일 수 있고, 개별 컨테이너가 요구하는 하드웨어 리소스(CPU, 메모리, 스토리지, 네트워크 등)가 적기 때문에 1대의 물리 서버에 매우 많은 수의 컨테이너를 탑재할 수 있음

​

\*게스트 OS: 가상머신(VM)에 설치된 OS

​

- 애플리케이션 실행 환경이 컨테이너 단위로 포장되어 있으므로 로컬 컴퓨터의 개발 환경을 클라우드로 옮기거나, 클라우드에서 다른 클라우드로 옮기는 경우 복제와 이식이 간편

​

- Docker의 오픈소스 애플리케이션 Docker가 대표적. 많은 장점이 기대되는 기술이나 아직 개척 단계

​

[텍스트이(가) 표시된 사진

자동 생성된 설명](https://m.blog.naver.com/shakey7/221408055973)

​

\*커널(kernel): 운영 체제를 규정짓는 매우 중요한 부분으로 하드웨어의 자원을 자원이 필요한 프로세스에 나눠주고, 프로세스 제어(태스크 매니저), 메모리 제어, 프로그램이 운영체제에 요구하는 시스템 콜 등을 수행. OS를 하나의 기업체로 비유한다면 커널은 인사 담당 부서

​

​

​

(2) 스토리지 기술

- 스토리지는 데이터와 프로그램을 저장하는 기록 장치

​

- 클라우드 서비스에서는 블록 스토리지, 파일 스토리지, 오브젝트 스토리지 3가지 액세스 방식을 제공

​

​

1) 블록 스토리지

- 일정한 크기의 블록으로 나뉜 스토리지의 논리 볼륨을 블록 단위로 액세스할 수 있는 스토리지

​

\*블록(block): 하나의 단위로서 다룰 수 있는 문자, 워드, 레코드의 집합으로, 데이터 전송과 기억 장치에서 블록은 기억 공간을 나누는 단위

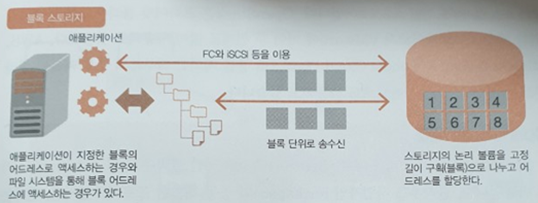
​

- 서버와 스토리지가 데이터를 교환할 때의 오버헤드가 적어 빠른 데이터 전송이 가능. 따라서 낮은 레이턴시가 요구되는 데이터베이스 용도에 적합

​

- 파이버 채널(FC)과 iSCSI 같은 전용 프로토콜 사용

​

[](https://m.blog.naver.com/shakey7/221408055973)

​

​

2) 파일 스토리지

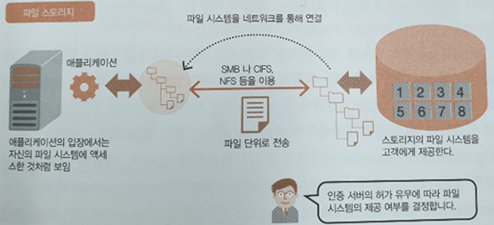
- 파일 단위로 저장 액세스가 가능하며, 파일 공유 기능을 갖춘 스토리지

​

- 주로 파일 서버로 이용되며 액세스 제어 및 속성 정보의 관리가 쉬움

​

- 윈도우 OS에서는 SMB(Server Message Block)와 CIFS(Common Internet File System), 유닉스 및 리눅스 OS에서는 NFS(Network File System) 같은 파일 공유 프로토콜 사용

[](https://m.blog.naver.com/shakey7/221408055973)

​

​

3) 오브젝트 스토리지

- 데이터를 객체 단위로 처리하는 스토리지

​

\*오브젝트(object, 객체): 데이터 및 메타데이터, 고유 ID를 하나로 묶은 단위

​

- 오브젝트 스토리지는 OS나 파일 시스템에 의존하지 않으면서도 데이터를 저장하고 오브젝트에 액세스 할 수 있음

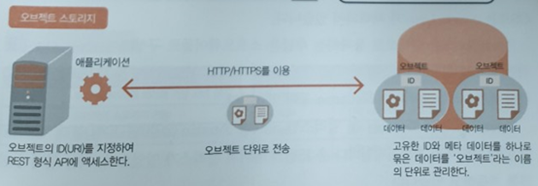
​

- 쉽게 용량을 늘릴 수 있으며, 데이터의 크기와 저장할 수 있는 데이터 수에 제한이 없음. 갱신 빈도가 낮은 데이터나 대량의 데이터를 저장하고 장기 보존하는 용도에 적합

​

- HTTP 프로토콜 기반의 REST(Representational State Transfer) 형식 API 사용

​

[](https://m.blog.naver.com/shakey7/221408055973)

​

​

​

(3) 네트워크 가상화

- 클라우드를 구현하기 위해서는 네트워크에도 물리적 구성에 얽매이지 않는 유연성이 필요. 네트워크 가상화 구현에는 VLAN, VPN, NFV, SDN 등이 있음

​

​

1) VLAN(Virtual LAN)

- 하나의 물리적인 네트워크를 여러 개의 논리적인 네트워크로 분할하는 기술

​

- 물리적인 배선을 변경하지 않고 네트워크 장비에 설정을 추가하여 네트워크를 나눌 수 있음

​

​

2) VPN(Virtual Private Network)

- 인터넷과 같은 불특정 다수가 이용하는 네트워크에 가상으로 전용선과 같은 사설망을 연결하는 기술

​

- 클라우드 서비스와 기업 사용자의 온프레미스 시스템이 인터넷을 통해 VPN에 연결할 때에는 IPsec라는 프로토콜을 사용. 이로 인해 통신 거점 인증, 통신 데이터 암호화가 이루어지므로 거점 간의 안전한 통신이 구현됨

​

​

3) NFV(Network Functions Virtualization)

- 네트워크 기능을 소프트웨어로 구현하여 가상 서버 위에 구축하는 기술

​

- 라우터, 게이트웨이, 방화벽, 로드 밸런서와 같은 네트워크 장비 기능을 가상 서버에서 애플리케이션 소프트웨어로 구현

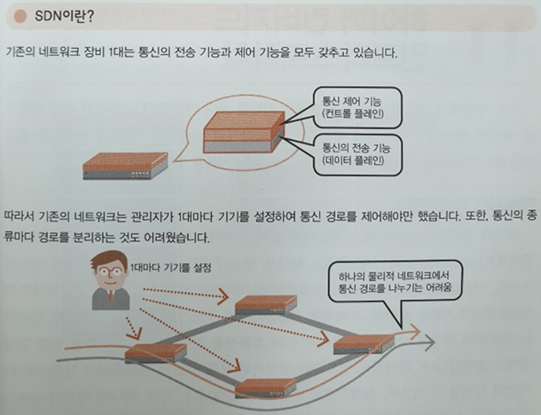
​

​

4) SDN(Software Defined Networking)

- 기존 네트워크 장비마다 가지고 있던 통신 전송 기능과 제어 기능을 분리하고, 제어 기능을 컨트롤러에 논리적으로 집중시켜서 데이터의 흐름을 소프트웨어로 정의하는 것

​

[](https://m.blog.naver.com/shakey7/221408055973)

​

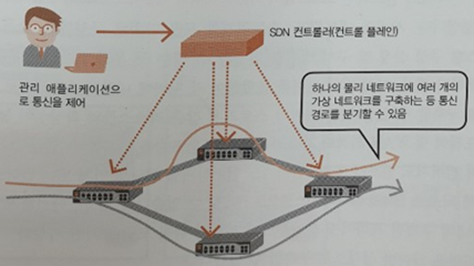
​

- 통신 제어 기능과 전송 기능을 분리하여 하드웨어는 전송 기능만을 갖게 하고, 컨트롤러와 같은 애플리케이션이 제어를 담당. 이를 통해 네트워크 가동 상황 및 운영에 맞추어 소프트웨어로 유연하게 데이터 전송 경로를 변경할 수 있음

​

- 서버 가상화와 마찬가지로 물리적 네트워크가 추상화되므로, 하나의 물리적 네트워크 위에 컨트롤러 별로 여러 가상 네트워크를 구축할 수 있음

​

[](https://m.blog.naver.com/shakey7/221408055973)

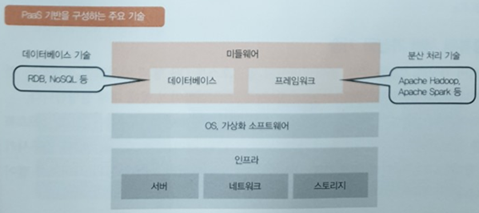
**1. 클라우드를 실현하는 기술**

**​**

- 클라우드 서비스의 특징은 다양한 가상화 및 분산 처리 기술들로 이루어짐

​

- PaaS 구성 주요 기술: 데이터베이스 기술, 분산 처리 기술 등

[](https://m.blog.naver.com/shakey7/221408505127)

**2. PaaS 기반을 구성하는 주요 기술**

​

(1) 분산 처리 기술

- 클라우드가 등장하기 전에는 테라 및 페타바이트 급 대용량 데이터를 처리하려면 고속 CPU와 대용량 메모리가 탑재된 서버가 필요했음. 하지만 지금은 분산 처리 기술과 클라우드 서비스를 이용하여 데이터를 여러 서버에 나누어 병렬로 처리할 수 있음

​

- 클러스터링은 여러 개의 서버를 결합하여 대량의 데이터를 분산 처리하는 장치로, 데이터 처리 중 몇 개의 서버에서 장애가 발생하더라도 다른 서버에 자동으로 작업을 할당하여 계속 이어나갈 수 있음

​

- Apache Hadoop(하둡)과 Apache Spark(스파크)가 분산 처리를 구현하는 대표적 소프트웨어

​

​

1) Apache Hadoop

- 1대의 마스터 서버와 그 아래에 있는 여러 대의 슬레이브 서버로 구성

​

- 마스터 서버가 데이터 처리 전체를 제어하고, 슬레이브 서버가 계산 처리. 처리 능력은 슬레이브 서버 대수에 비례

​

- 대용량 데이터의 일괄 처리에 적합

​

​

2) Apache Spark

- 메모리 안에서 대량 데이터를 병렬 분산 처리

​

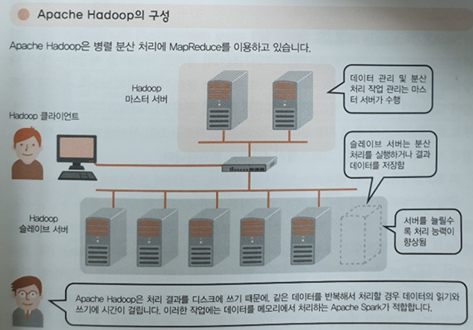
- Apache Hadoop이 반복 처리를 할 때 디스크에 데이터를 읽고 쓰는 것과 비교했을 때, Spark는 메모리 안에서만 읽고 쓰므로 속도가 매우 빠름

​

- 그러나 클러스터링으로 구성된 메모리에 모두 올릴 수 없는 테라바이트 이상의 데이터 처리에는 적합하지 않음

​

- 인공지능 기계학습 및 같은 데이터를 반복해서 처리하는 고급 데이터 분석을 빠르게 수행하는 데 적합

[](https://m.blog.naver.com/shakey7/221408505127)

​

​

​

(2) 데이터베이스 기술

- 대량의 데이터 분석 및 관리, 트랜잭션 처리(상거래와 같은 일련의 처리) 등에 쓰이는 기술로 RDB와 NoSQL 등이 있음. 최근에는 빅데이터 및 IoT로 인해 NoSQL의 이용도가 증가

​

\*SQL 및 NoSQL에 관한 설명 참고

**[클라우드 기본 개념 정리 3 (VPC / 네트워크 / 데이터베이스 / 웨어하우스 / 빅데이터)](https://blog.naver.com/shakey7/221398503566" \t "_blank)**

[1. 클라우드 네트워크 서비스 - 클라우드 위에 네트워크를 구축. 대표적인 예로는 Amazon VPC(Virtual...](https://blog.naver.com/shakey7/221398503566" \t "_blank)

[blog.naver.com](https://blog.naver.com/shakey7/221398503566" \t "_blank)

​

[테이블이(가) 표시된 사진

자동 생성된 설명](https://m.blog.naver.com/shakey7/221408505127)

​

​

1) RDB(Relational Database, 관계형 데이터베이스)

- RDB는 여러 개의 데이터를 행과 열이 있는 표 형식으로 표현하여 복잡한 데이터의 관계를 처리할 수 있도록 만든 데이터베이스

​

- RDBMS(Relational Database Management System, 관계형 데이터베이스 관리 시스템)라는 전용 소프트웨어로 관리

​

​

2) NoSQL(Not only SQL)

- RDB 같은 관계형 데이터베이스가 아닌 데이터베이스를 의미

​

- 같은 NoSQL 기반이더라도 제품별로 자료 구조(어떤 형태로 데이터를 저장하는가)가 다양. 하지만 대량의 데이터를 분산시켜 고속으로 처리하는 분산 데이터베이스라는 점은 공통적인 특징

​

- 분산 처리라는 특징으로 인해 클라우드 서비스 구현에 적합하며 주로 빅데이터 분석 등에 사용

​

- NoSQL은 자료 구조의 차이에 따라 키 밸류 형, 컬럼 지향형, 문서 지향형, 그래프형 4가지로 분류

​

① 키 밸류 형: 모든 데이터는 인덱싱된 값으로 구성되어 있음. 구성이 간단하고 확장성이 높으며 데이터 읽기 속도가 빠름

\*인덱싱(indexing): 데이터베이스에 색인을 남기는 것으로, 어떤 데이터가 어디에 있다는 위치 정보를 제공하여 검색 속도를 높이는 기술

② 컬럼 지향형: 컬럼(열) 단위로 데이터를 보관. 대량의 컬럼 단위 데이터의 집계 및 갱신 속도가 빠르고 데이터 쓰기 속도가 빠름

③ 문서 지향형: 복잡한 데이터를 문서에 저장하고 문서 단위로 데이터를 저장하여 검색하고 관리할 수 있음. 따라서 복잡한 데이터를 처리하는 응용 프로그램 개발에 적합

④ 그래프형: 데이터 간의 관계를 그래프로 형성하고 통합 검색을 빠르게 할 수 있음

[테이블이(가) 표시된 사진

자동 생성된 설명](https://m.blog.naver.com/shakey7/221408505127)