programmers

실리콘밸리에서 날아온 데이터베이스

1. 데이터베이스 시스템 소개

keeyonghan@hotmail.com

한기용

Harmonize, Inc

programmers

Contents

◆ 강사 소개 - 한기용

- ❖ 2020-현재 Harmonize Health 데이터 팀 디렉터
- ❖ 2018-2020년 데이터 관련 컨설턴트/어드바이저/엔젤투자자
- ❖ 2014-2018년 Udemy 데이터 팀 시니어 디렉터
- ❖ 2012-2014년 Polyvore 데이터 팀 시니어 매니저
- ❖ 2004-2011년 Yahoo Search 엔지니어링 디렉터
- ❖ 2000년에 미국 실리콘밸리로 이주
- ❖ 1995-2000년 삼성전자 소프트웨어 개발자
- ❖ 서울대학교 컴퓨터 공학과 학사/석사

programmers

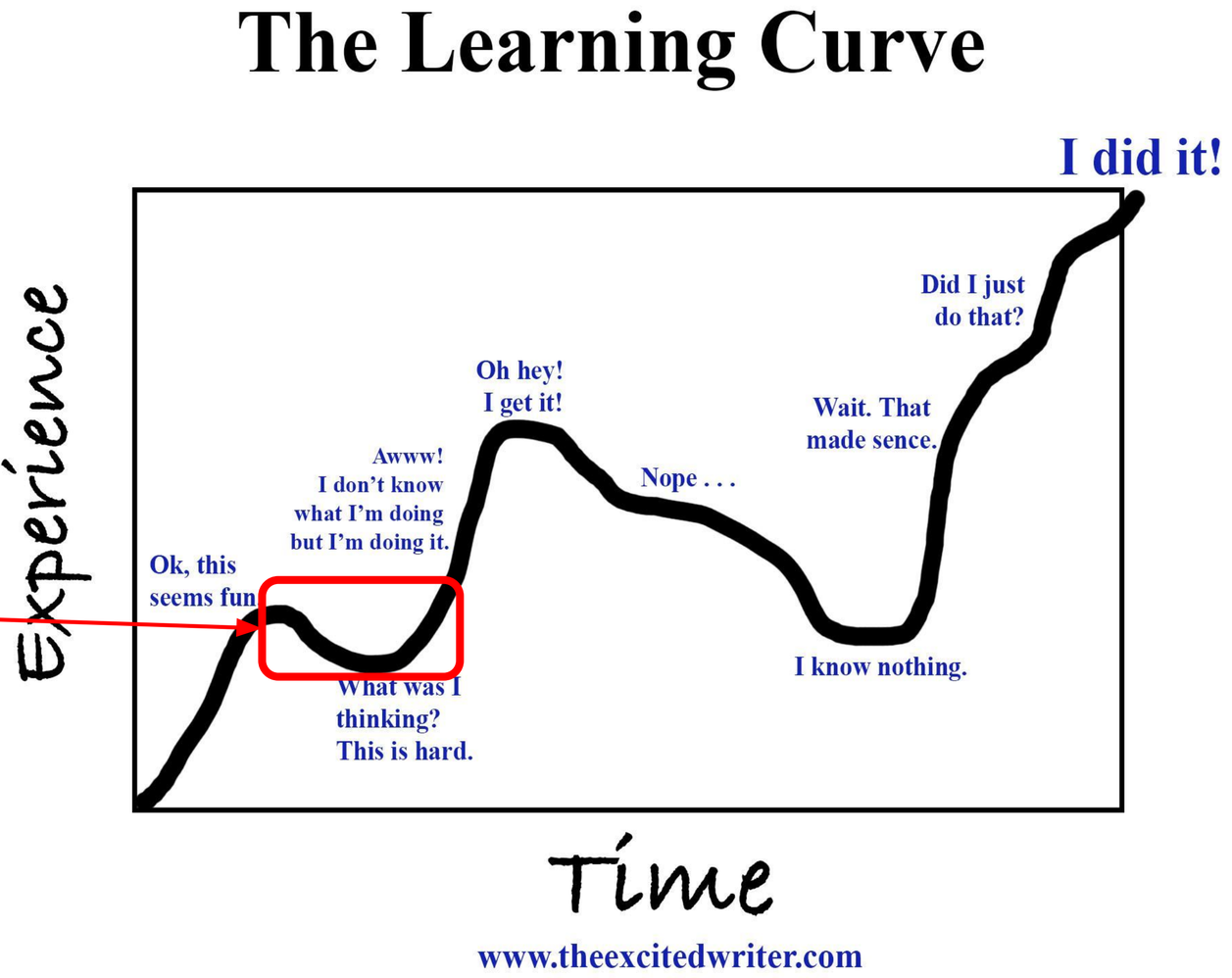
요즘 세상의 배움이란?

배움에는 시간과 노력이 걸림

◆ 배움의 전형적인 패턴

여기서 어떻게 하느냐가 아주 중요!

1. 가장 중요한 것은 버티는 힘
   a. 이걸 즐겨야함 :)
2. 내가 뭘 모르지는지 생각해봐야함
   a. 내가 어디서 막혔는지 구체적으로 질문할 수 있나?
3. 잘 하는 사람 보고 기죽지 않기



◆새로운 것을 처음 배울 때의 좋은 자세 (1)

❖ 자신이 아는 것과 모르는 것을 분명히 이해하는지?

- 멍청한 질문이란 없다는 점 명심. 대충 알거나 모르면서 안 물어보는 것이 더 큰 문제
- 이는 피드백을 잘 받아들일 수 있는지와도 연계됨

❖ 마음을 편하게 먹기

- 내가 이해하기 힘들다면 남들도 이해하기 힘듬
- 나보다 잘 하는 사람들은 그만큼 시간을 쏟았기 때문

◆ 새로운 것을 처음 배울 때의 좋은 자세 (2)

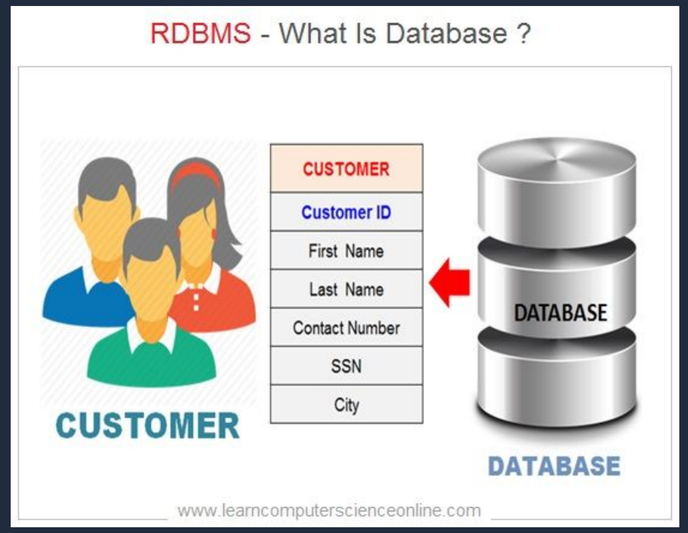
❖ 배움의 발전은 tipping point를 거치면서 폭발하는 형태임

- "The secret is to build the resolve and spirit to enjoy the plateaus the times when it doesn't feel like you're improving and you question why you are doing this. If you're patient, the plateaus will become springboards" (Quotes from Steve Nash)
- 발전이 더딘 기간을 **즐기는** 자세가 필요

programmers

데이터베이스가 왜 필요한가?

IT 서비스를 만드는데 왜 데이터베이스가 필요한가?



Source

◆ 모든 서비스는 데이터를 만들어내고 이는 기록되어야 함

❖ 모든 서비스는 데이터를 만들어내고 그 데이터의 저장을 필요로 함

- 예) 카카오톡
  - 사용자 등록시 사용자 정보 (ID, 암호, 전화번호 등등)의 저장이 필요
  - 사용자 친구 리스트 관리 필요 (친구들의 ID를 저장)
  - 특정 사용자 혹은 단톡방에서의 채팅 기록을 저장해야함
- 예) 쿠팡
  - 회원 가입시 회원 정보 (ID, 암호, 전화번호, 주소 등등) 저장이 필요
  - 구매 관련 정보의 저장이 필요
  - 추천을 잘 하려면 사용자별로 검색기록과 보거나 클릭했던 상품정보의 저장과 가공이 필요
  - ...

◆ 어디에 이런 데이터를 저장해야할까?

❖ 프로덕션 관계형 데이터베이스 (RDBMS)

- 어떤 서비스의 운영에 필요 데이터를 저장하는 곳 (MySQL, PostgreSQL, …)
- 빠른 처리속도가 중요함
  - vs. 데이터 웨어하우스 관계형 데이터베이스
- 데이터를 구조화된 테이블들의 집합으로 구성하여 저장하고 관리
- 백엔드 개발자이건 프런트엔드 개발자이건 잘 알아야하는 기본 기술
  - 관계형 데이터베이스의 프로그래밍 언어가 SQL

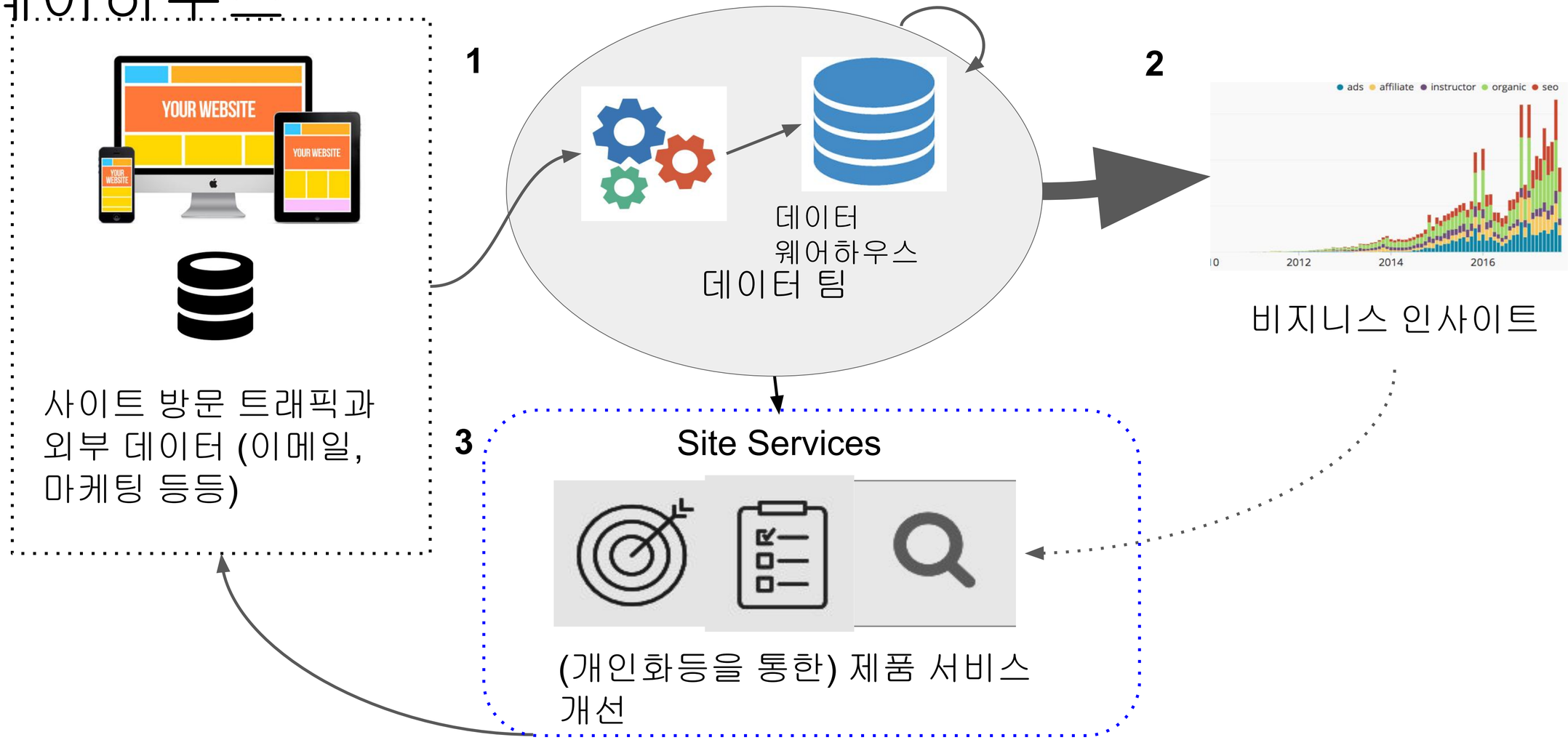
Name	FName	City	Age	Salary
Smith	John	3	35	$280
Doe	Jane	1	28	$325
Brown	Scott	3	41	$265
Howard	Shemp	4	48	$359
Taylor	Tom	2	22	$250

◆ 데이터 분석을 위한 데이터베이스

❖ 데이터 웨어하우스

- 회사 관련 데이터를 저장하고 분석함으로써 의사결정과 서비스 최적화에 사용
  - BigQuery, Snowflake, Redshift, MySQL, ...
- 처리속도 보다는 구조화된 큰 데이터를 처리하는 것이 중요
  - vs. 프로덕션 관계형 데이터베이스
- 데이터 직군이라면 반드시 알아야함 (SQL)
  - 데이터 엔지니어, 데이터 분석가, 데이터 과학자

◆데이터 순환 구조: 프로덕션용 데이터베이스와 데이터 웨어하우스



◆ 다양한 데이터베이스의 종류

❖ 관계형 데이터베이스: 구조화된 데이터

- **프로덕션용 관계형 데이터베이스**
- 데이터 웨어하우스용 관계형 데이터베이스

❖ 비관계형 데이터베이스: 비구조화 데이터도 다룸

- 흔히 NoSQL 데이터베이스라고 부르기도 함
- 보통은 프로덕션용 관계형 데이터베이스를 보완하기위한 용도로 많이 사용됨
- 크게 4종류가 존재
  - Key/Value Storage: Redis, Memcache, ...
  - Document Store: MongoDB,
  - Wide Column Storage: Cassandra, HBase, DynamoDB
  - Search Engine: ElasticSearch

programmers

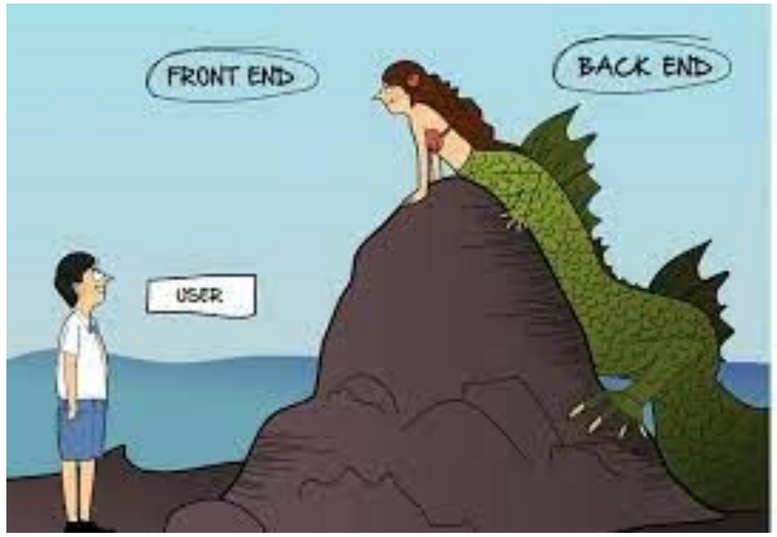
백엔드 시스템 구성도 예제 보기

어떤 개발자 직군들이 있는지 알아보자
관계형 데이터베이스가 전체 시스템에서 어떻게 사용되는지
몇가지 구성도를 살펴보자

◆ 프런트엔드와 백엔드

❖ 웹/앱 서비스를 간단하게 보면 크게 프런트 엔드와 백엔드로 구성

- 프런트엔드: 사용자와 인터랙션을 하는 부분으로 보통 웹 브라우저 혹은 모바일 폰에 사용자에게 노출되는 서비스를 말함
- 백엔드: 프런트엔드 뒤에 숨어서 사용자에게 보이지는 않지만 실제 데이터를 저장/추가하고 사용자가 요구한 일을 수행하는 부분. 여기에 다양한 데이터베이스들이 사용됨
- 초기에는 이렇게 크게 두 직군이 존재



https://www.geeksforgeeks.org/

◆ 다른 직군의 등장

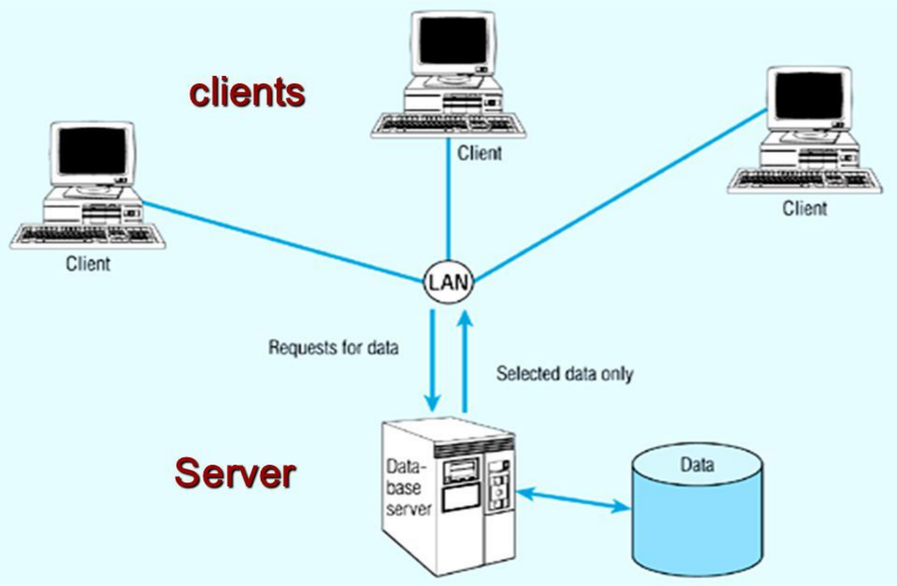
- ❖ 데브옵스 (DevOps): 주로 백엔드에 집중을 두고 서비스의 운영을 책임지는 팀으로 회사가 작을 때는 보통 백엔드 팀이 이 일을 담당
- ❖ 풀스택 (Fullstack): 개발속도를 내기위해 프런트엔드/백엔드를 모두 할 수 있는 개발자로 보통 작은 회사에서 선호하는 형태의 직군
- ❖ 데이터 직군: 데이터의 중요성이 증대되면서 3가지 데이터 직군이 등장
  - 데이터 엔지니어: 사실상 소프트웨어 개발자로 데이터 웨어하우스와 관련일을 담당
    - MLOps라는 직군이 나타나기 시작
  - 데이터 분석가: 데이터 웨어하우스를 기반으로 다양한 지표설정과 분석 수행
  - 데이터 과학자: 수집된 과거 데이터를 기반으로 미래를 예측하는 모델링 혹은 개인화 작업으로 서비스의 만족도를 높이고 프로세스의 최적화를 수행

◆ 시스템 구성의 변화: 2 tier

❖ 보통 데스크탑 응용프로그램에서 사용되는 아키덱쳐

❖ 클라이언트와 서버 두 개의 티어로 구성

- 클라이언트는 사용자가 사용하는 UI가 됨 (front-end)
  - 비지니스 로직은 보통 클라이언트에 위치
- 서버단이 데이터베이스가 됨 (back-end)

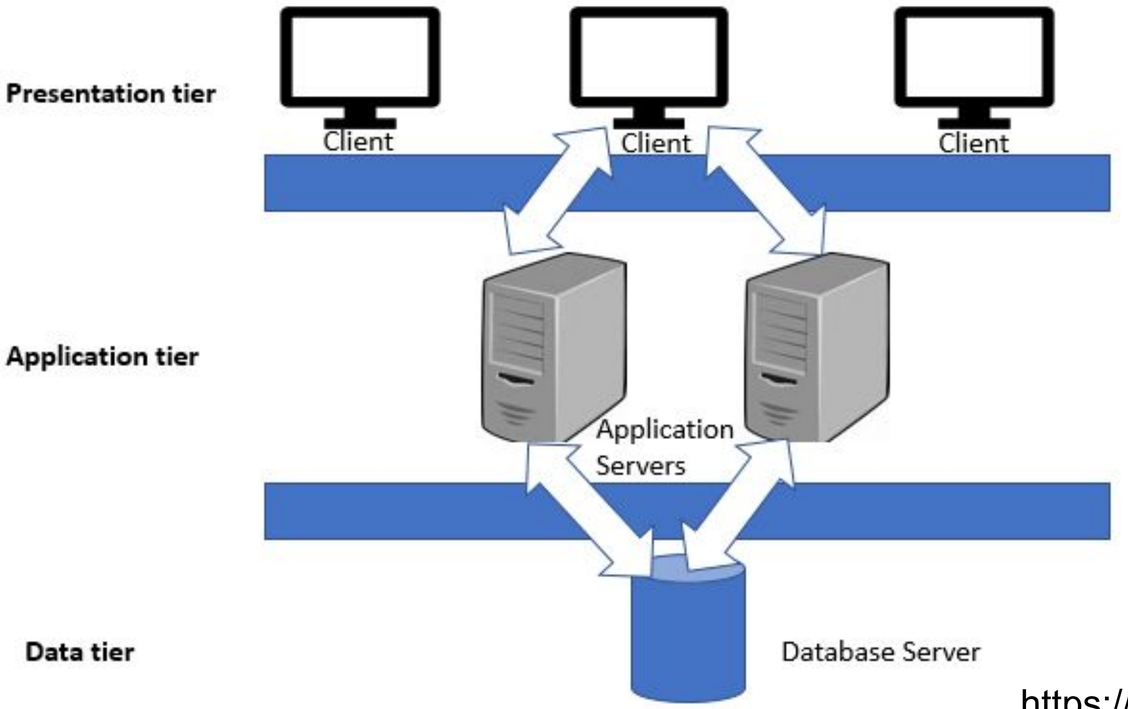


source: https://slideplayer.com/slide/5057689/

◆ 시스템 구성의 변화: 3 tier

❖ 웹 서비스에서 많이 사용되는 아키텍쳐

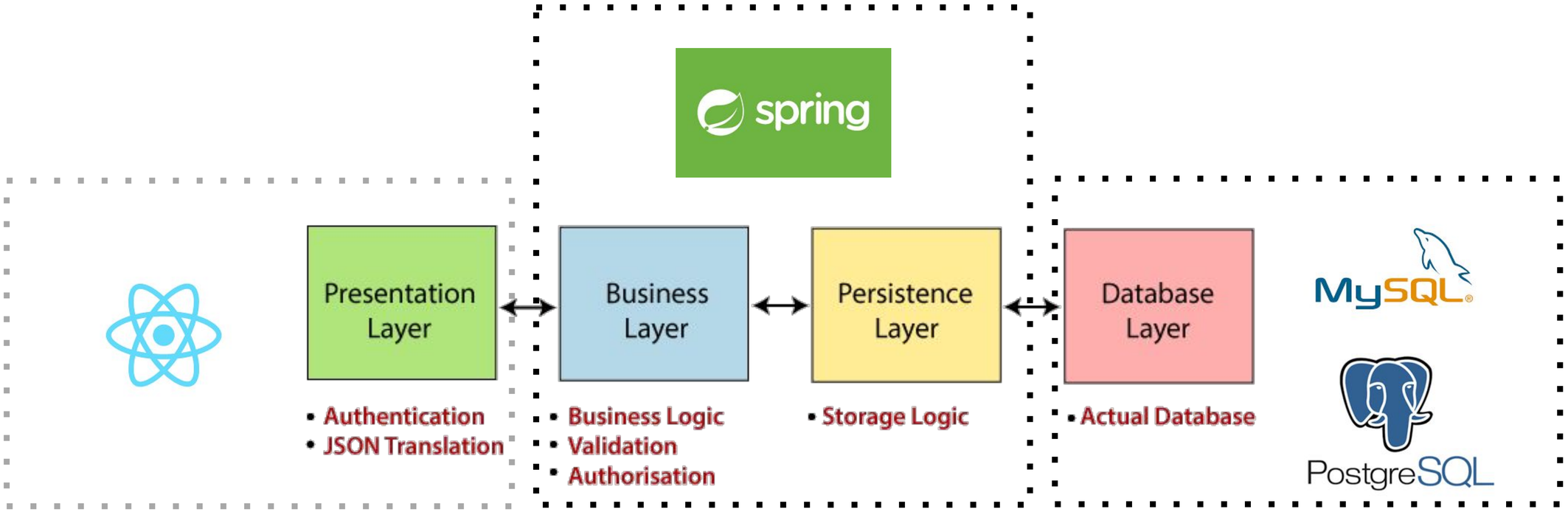
- 프리젠테이션 티어 (Presentation Tier): 프런트엔드
- 애플리케이션 티어 (Application Tier): 백엔드
- 데이터 티어 (Data Tier): 백엔드



https://subscription.packtpub.com/book/application_development/

◆ 시스템 구성 예: Spring Boot

- ❖ 기본적으로 3 tier 구조라고 볼 수 있음
- ❖ Spring Boot는 백엔드 프레임웍 중의 하나 (자바기반)
  - 파이썬의 경우 Django/Flask, 자바스크립트의 경우 Node.js
- ❖ 프런트엔드로 가장 많이 쓰이는 옵션은 React



◆ 관계형 데이터베이스의 중요성

- ❖ 어떤 구조이건 데이터베이스는 꼭 필요한 컴포넌트
- ❖ 이 데이터베이스를 잘 다룬 것이 좋은 개발자가 되기 위해 필요
  - 기본은 SQL을 잘 아는 것
- ❖ 백엔드 개발자로써 중요한 부분
  - 데이터 모델을 잘 만들고 그걸 프런트 개발자와 공유/협업
  - 속도 개선을 위한 쿼리 성능 모니터링하고, 필요시 성능 개선 수행
    - 어떤 경우에는 이를 전담하는 사람이 존재 (DBA - DataBase Administrator)

programmers

관계형 데이터베이스 소개

구조화된 데이터를 저장하는데 사용되는 관계형
데이터베이스가 무엇인지 알아보자

◆ 관계형 데이터베이스 (1)

❖ 구조화된 데이터를 저장하고 질의할 수 있도록 해주는 스토리지

- 엑셀 스프레드시트 형태의 테이블로 데이터를 정의하고 저장
  - 테이블에는 컬럼(열)과 레코드(행)이 존재

id	first_name	last_name	email	phone_number
1	John	Doe	john.doe@gmail.com	1-408-111-1111
2	Jane	Doe	jane.doe@gmail.com	1-408-111-2222
3	Keeyong	Han	keeyong.han@gmail.c	1-408-111-3333

Customers
테이블

id	purchased_date	customer_id	product_id	amount
1	2021-01-01	1	100	19.99
2	2021-01-02	1	101	29.99
3	2021-01-03	3	102	11.99

Sales 테이블

◆ 관계형 데이터베이스 (2)

❖ 관계형 데이터베이스를 조작하는 프로그래밍 언어가 SQL

- 테이블 정의를 위한 DDL (Data Definition Language)
  - 앞서 보여준 테이블의 포맷을 정의해주는 언어
- 테이블 데이터 조작/질의를 위한 DML (Data Manipulation Language)
  - DDL로 정의된 테이블에 레코드를 추가, 수정, 삭제 혹은 읽어들이기 위해 사용하는 언어

◆ 대표적 관계형 데이터베이스

❖ 프로덕션 데이터베이스: MySQL, PostgreSQL, Oracle, …

- OLTP (OnLine Transaction Processing)
- 빠른 속도에 집중. 서비스에 필요한 정보 저장
- MySQL이 바로 우리의 집중 탐구 대상

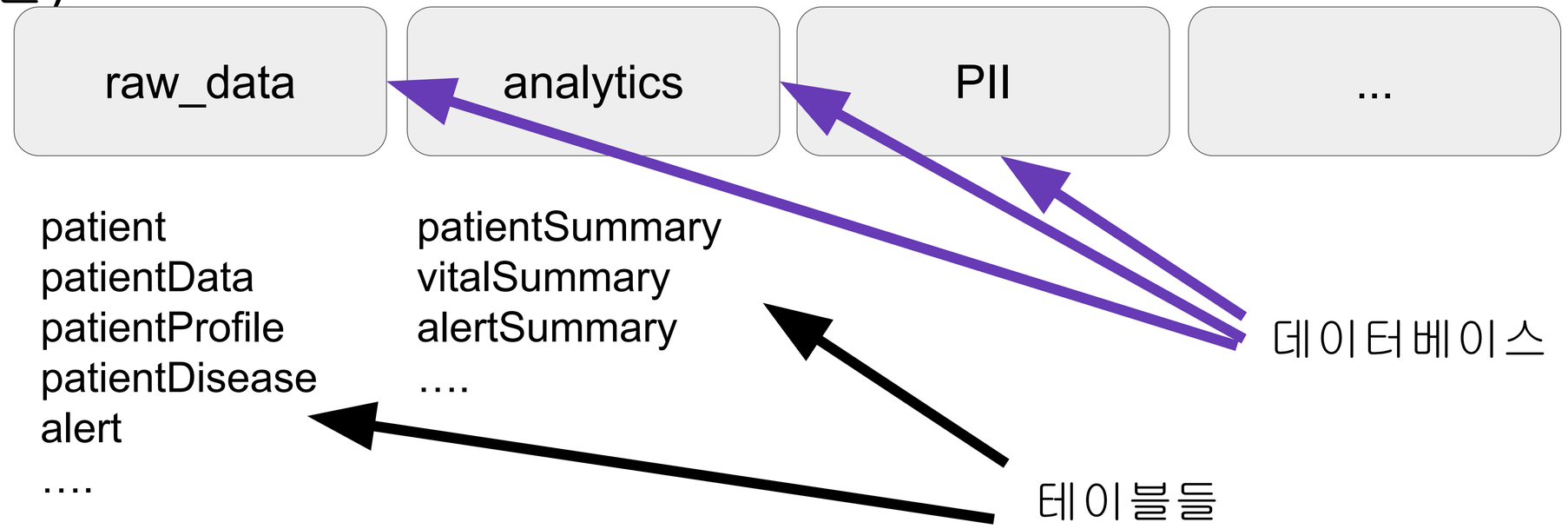
❖ 데이터 웨어하우스: Redshift, Snowflake, BigQuery, Hive, …

- OLAP (OnLine Analytical Processing)
- 처리 데이터 크기에 집중. 데이터 분석 혹은 모델 빌딩등을 위한 데이터 저장
  - 보통 프로덕션 데이터베이스를 복사해서 데이터 웨어하우스에 저장

◆ 관계형 데이터베이스의 구조

❖ 관계형 데이터베이스는 2 단계로 구성됨

- 가장 밑단에는 테이블들이 존재 (테이블은 엑셀의 시트에 해당)
- 테이블들은 데이터베이스(혹은 스키마)라는 폴더 밑으로 구성 (엑셀에서는 파일)



◆ 관계형 데이터베이스의 구조

❖ 테이블의 구조 (테이블 스키마라고 부르기도 함)

- 테이블은 레코드들로 구성 (행)
- 레코드는 하나 이상의 필드(컬럼)로 구성 (열)
- 필드(컬럼)는 이름과 타입과 속성(primary key)으로 구성됨

컬럼	타입
user_id	int
session_id	varchar(32)
channel	varchar(32)

테이블 스키마

user_id	session_id	channel
779	7cdace91c487558e27ce54df7cdb299c	Instagram
230	94f192dee566b018e0acf31e1f99a2d9	Naver
369	7ed2d3454c5eea71148b11d0c25104ff	Youtube
248	f1daf122cde863010844459363cd31db	Naver

테이블 레코드 예

programmers

SQL 소개

관계형 데이터베이스의 프로그래밍 언어 **SQL**에 대해 배우자!

◆ SQL 소개

❖ SQL: Structured Query Language

- 관계형 데이터베이스에 있는 데이터(테이블)를 질의하거나 조작해주는 언어

❖ SQL은 1970년대 초반에 IBM이 개발한 구조화된 데이터 질의 언어

❖ 두 종류의 언어로 구성됨

- DDL (Data Definition Language):
  - 테이블의 구조를 정의하는 언어
- DML (Data Manipulation Language):
  - 테이블에서 원하는 레코드들을 읽어오는 질의 언어
  - 테이블에 레코드를 추가/삭제/갱신해주는데 사용하는 언어

◆ SQL은 빅데이터 세상에서도 중요!

- ❖ 구조화된 데이터를 다루는한 SQL은 데이터 규모와 상관없이 쓰임
- ❖ 모든 대용량 데이터 웨어하우스는 SQL 기반
  - Redshift, Snowflake, BigQuery, Hive
- ❖ Spark이나 Hadoop도 예외는 아님
  - SparkSQL과 Hive라는 SQL 언어가 지원됨
- ❖ 백엔드/프런트엔드/데이터 분야에서 반드시 필요한 기본 기술

◆ SQL의 단점

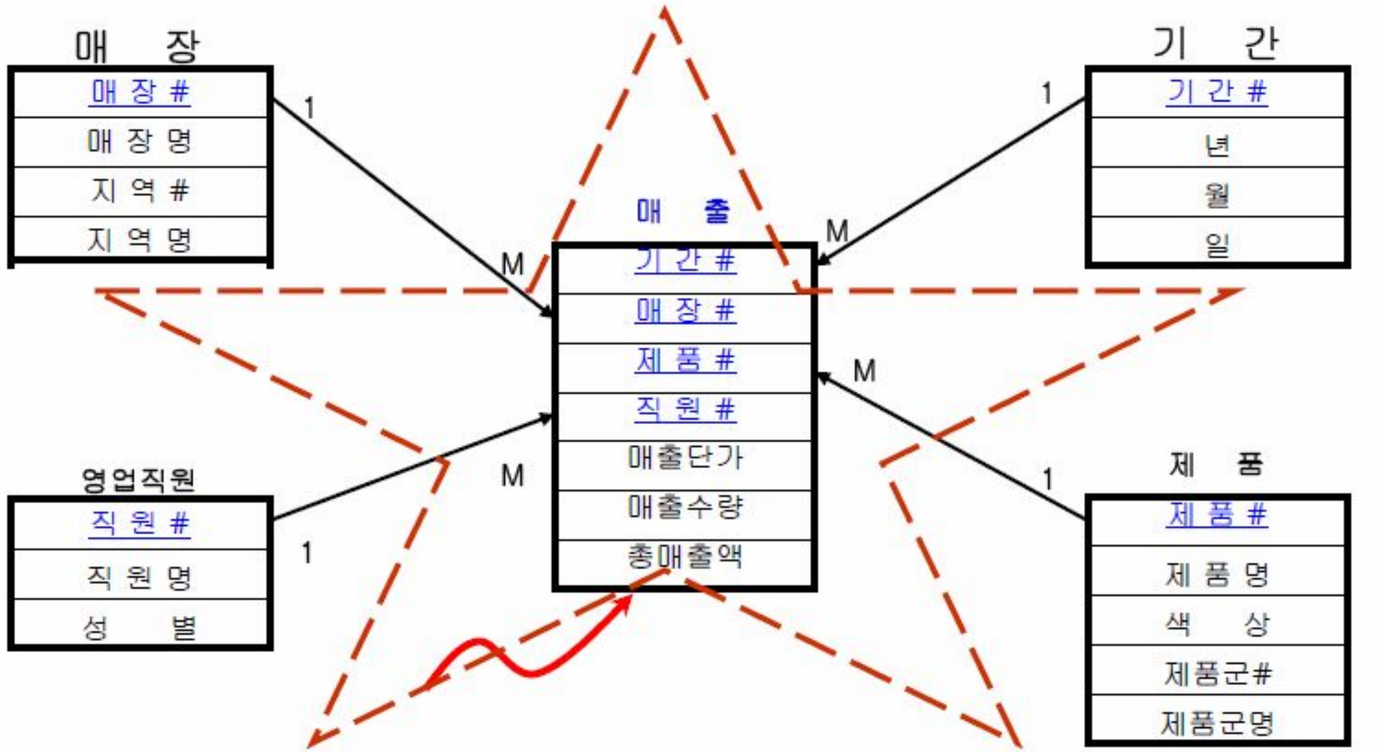
❖ 구조화된 데이터를 다루는데 최적화가 되어있음

- 정규표현식을 통해 비구조화된 데이터를 어느 정도 다루는 것은 가능하나 제약이 심함
- 많은 관계형 데이터베이스들이 플랫한 구조만 지원함 (no nested like JSON)
  - 구글 빅쿼리는 nested structure를 지원함
- 비구조화된 데이터를 다루는데 **Spark, Hadoop**과 같은 분산 컴퓨팅 환경이 필요해짐
  - 즉 SQL만으로는 비구조화 데이터를 처리하지 못함

❖ 관계형 데이터베이스마다 SQL 문법이 조금씩 상이

◆ Star schema

- ❖ Production DB용 관계형 데이터베이스에서는 보통 스타 스키마를 사용해 데이터를 저장
- ❖ 데이터를 논리적 단위로 나눠 저장하고 필요시 조인
- ❖ 스토리지의 나비기 덮힌구 업데이트기 싱은



◆ Denormalized schema

❖ NoSQL이나 데이터 웨어하우스에서 사용하는 방식

- 단위 테이블로 나눠 저장하지 않음으로 별도의 조인이 필요 없는 형태를 말함

❖ 이는 스토리지를 더 사용하지만 조인이 필요 없기에 빠른 계산이 가능

년 월 일
매장명
지역명
직원명
성별
매출단가
매출수량
총매출액
제품명
색상
제품군명

Denormalize된
매출 테이블

◆ SQL 기본

- ❖ 먼저 다수의 SQL 문을 실행한다면 세미콜론으로 분리 필요
  - SQL문1; SQL문2; SQL문3;
- ❖ SQL 주석
  - -- : 인라인 한줄짜리 주석. 자바에서 //에 해당
  - /* -- */: 여러 줄에 걸쳐 사용 가능한 주석
- ❖ SQL 키워드는 대문자를 사용한다던지 하는 나름대로의 포맷팅이 필요
  - 팀 프로젝트라면 팀에서 사용하는 공통 포맷이 필요
- ❖ 테이블/필드이름의 명명규칙을 정하는 것이 중요
  - 단수형 vs. 복수형
    - User vs. Users
  - _ vs. CamelCasing
    - user_session_channel vs. UserSessionChannel

◆ SQL DDL - 테이블 구조 정의 언어 (1)

❖ CREATE TABLE

❖ Primary key 속성을 지정할 수 있음

- Primary key uniqueness: 유일키 보장

❖ 성능향상을 위해 인덱스를 지정할 수 있음

```
CREATE TABLE raw_data.user_session_channel (
    userid int,
    sessionid varchar(32) primary key,
    channel varchar(32)
);
```

◆ SQL DDL - 테이블 구조 정의 언어 (2)

❖ DROP TABLE

- DROP TABLE table_name;
  - 없는 테이블을 지우려고 하는 경우 에러를 냄
- DROP TABLE **IF EXISTS** table_name;
- vs. DELETE FROM
  - DELETE FROM은 조건에 맞는 레코드들을 지움 (테이블 자체는 존재)

◆ SQL DDL - 테이블 구조 정의 언어 (3)

❖ ALTER TABLE

- 새로운 컬럼 추가:
  - ALTER TABLE 테이블이름 ADD COLUMN 필드이름 필드타입;
- 기존 컬럼 이름 변경:
  - ALTER TABLE 테이블이름 RENAME 현재필드이름 to 새필드이름
- 기존 컬럼 제거:
  - ALTER TABLE 테이블이름 DROP COLUMN 필드이름;
- 테이블 이름 변경:
  - ALTER TABLE 현재테이블이름 RENAME to 새테이블이름;

◆ SQL DML - 테이블 데이터 조작 언어 (1)

❖ 레코드 질의 언어: SELECT

- 뒤에서 더 자세히 설명
- SELECT FROM: 테이블에서 레코드와 필드를 읽어오는데 사용
- WHERE를 사용해서 레코드 선택 조건을 지정
- GROUP BY를 통해 정보를 그룹 레벨에서 뽑는데 사용하기도 함
  - DAU, WAU, MAU 계산은 GROUP BY를 필요로 함
- ORDER BY를 사용해서 레코드 순서를 결정하기도 함
- 보통 다수의 테이블의 조인해서 사용하기도 함. 이에 대해 4일차에 더 배움

◆ SQL DML - 테이블 데이터 조작 언어 (2)

❖ 레코드 추가/삭제/수정 언어:

- INSERT INTO: 테이블에 레코드를 추가하는데 사용
- UDATE FROM: 테이블 레코드의 필드 값 수정
- DELETE FROM: 테이블에서 레코드를 삭제
  - vs. TRUNCATE

강의 요약

요즘 세상의 배움
데이터베이스는 IT 시스템의 필수 조건
관계형 데이터베이스와 SQL