

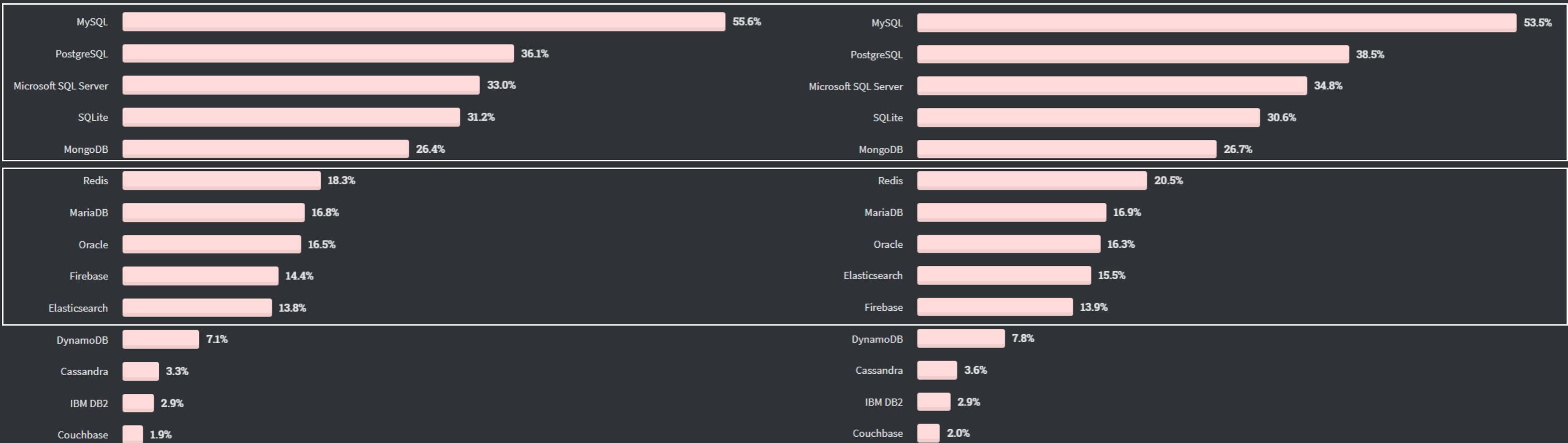
SQL vs NoSQL

향해99 8기 f반

발표자 : 이민석 / 깃 : github.com/uncharteder

Stackoverflow, 2020

Most Popular Database



전체 응답자

전문 개발자

Stackoverflow, 2020

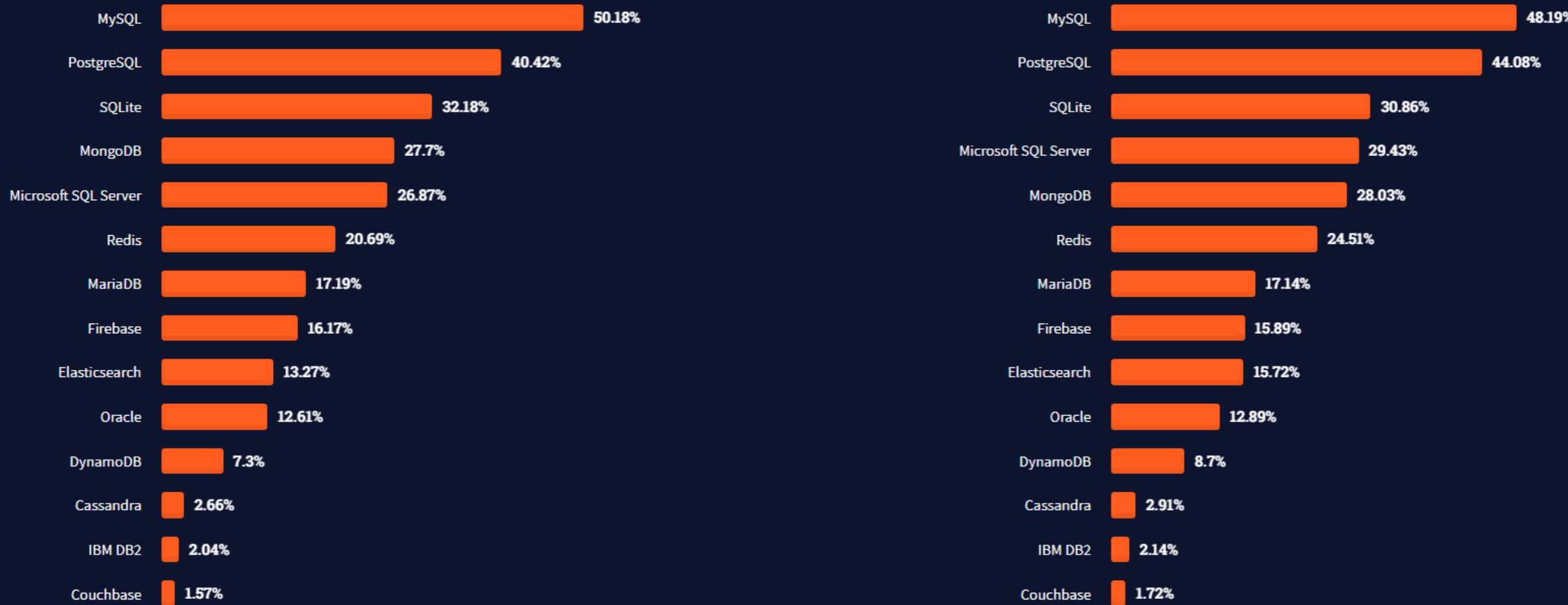
Most Popular Database



전체 응답자에서는 Firebase 가 0.6 p 높지만,
전문 개발자에서는 Elasticsearch 가 1.6 p가 높다

Stackoverflow, 2021

Most Popular Database



Stackoverflow, 2020 and 2021

Most Popular Database

		MySQL	PostgreSQL	Gap
2020	전체	55.6%	36.1%	19.50%
	전문	53.5%	38.5%	15.40%
2021	전체	50.18%	40.42%	9.76%
	전문	48.19%	44.08%	4.10%

SQL vs NoSQL / 정의



SQL

행과 열로 이루어진 , 엑셀?

NoSQL

다양한 형태, 다양한 목적?

SQL vs NoSQL / 특징



SQL

다양한 테이블 간의 '**관계**'를 통해서,
높은 확장성을 가지게 됨

NoSQL

특정한 요구사항에 '**최적화**' 된 성능과
높은 유연함을 가지게 됨



SQL

다양한 서 비스 공급 업체의 존재
두터운 사용자 층 만큼, 다양한 **참고자료**
보편적인 상황에서 **높은 성능**

NoSQL

복잡한 기술 과제를 손쉽게 **구현** 가능
제한된 상황에서 **높은 성능** 보장
몇몇 서비스는 매우 **쉽게** 적용 가능



SQL

SQL 문법 를 어느 정도 배워야 함
행과 열 에 대한 감을 잡기가 어려움

NoSQL

장점의 대다수가 동시에 **단점** 이 되기도 함

SQL vs NoSQL / 사용은?



SQL

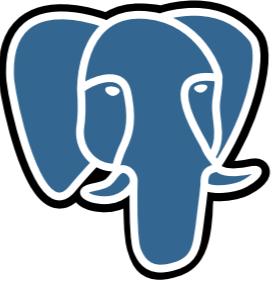
많은 업체에서 사용하고 있음

NoSQL

역시나 많은 업체에서 사용하고 있음

SQL vs NoSQL / 예시들 . . .

SQL



PostgreSQL



MySQL

MySQL

ORACLE

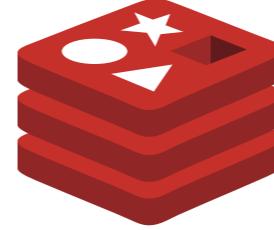
Oracle

NoSQL



MongoDB

문서 기반,
MongoDB



메모리 기반,
Redis



문서 기반,
DynamoDB



그래프 기반,
Neptune

SQL / 언어로서의 SQL 과 표준화 기관

Java 나 JavaScript 와 같은 언어의 일종으로,
관계형 데이터 베이스 엑세스 및 조작을 위한 표준 언어

1970년대 IBM에서 개발

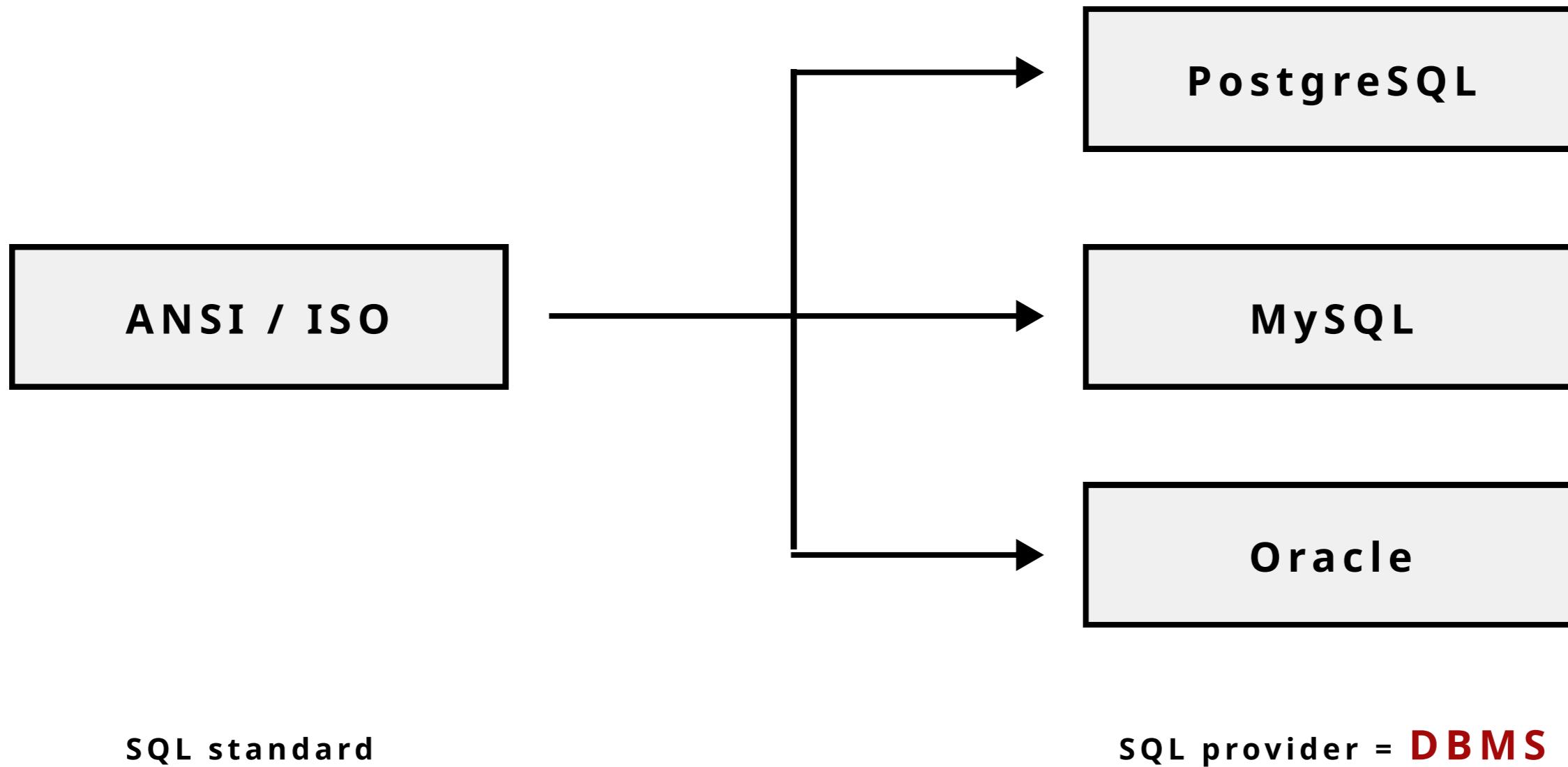
1986년도 ANSI American National Standards Institute에서 표준으로 채택

1987년도 ISO International Organization for Standardization에서 표준으로 채택

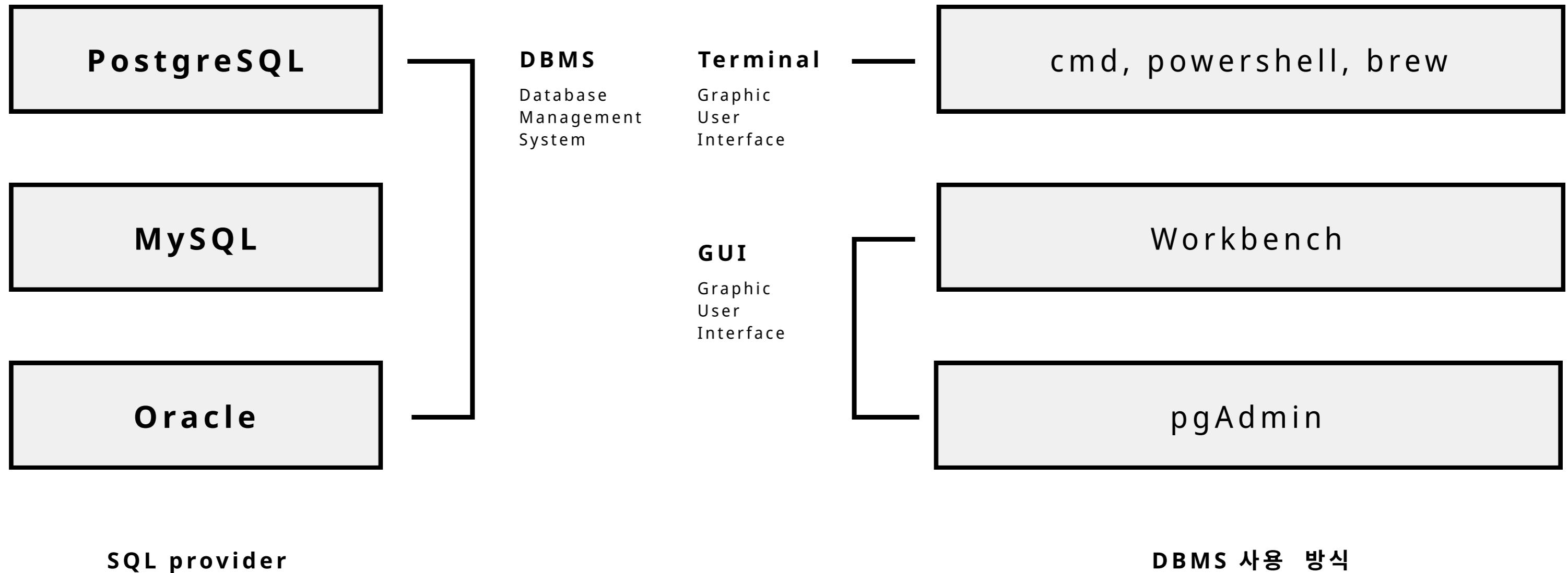
ANSI & ISO 기관에서 SQL 언어의 표준화

ECMAScript에서 JavaScript를 개선하는 것 처럼...?

SQL / ANSI 와 여러 SQL 공급 업체



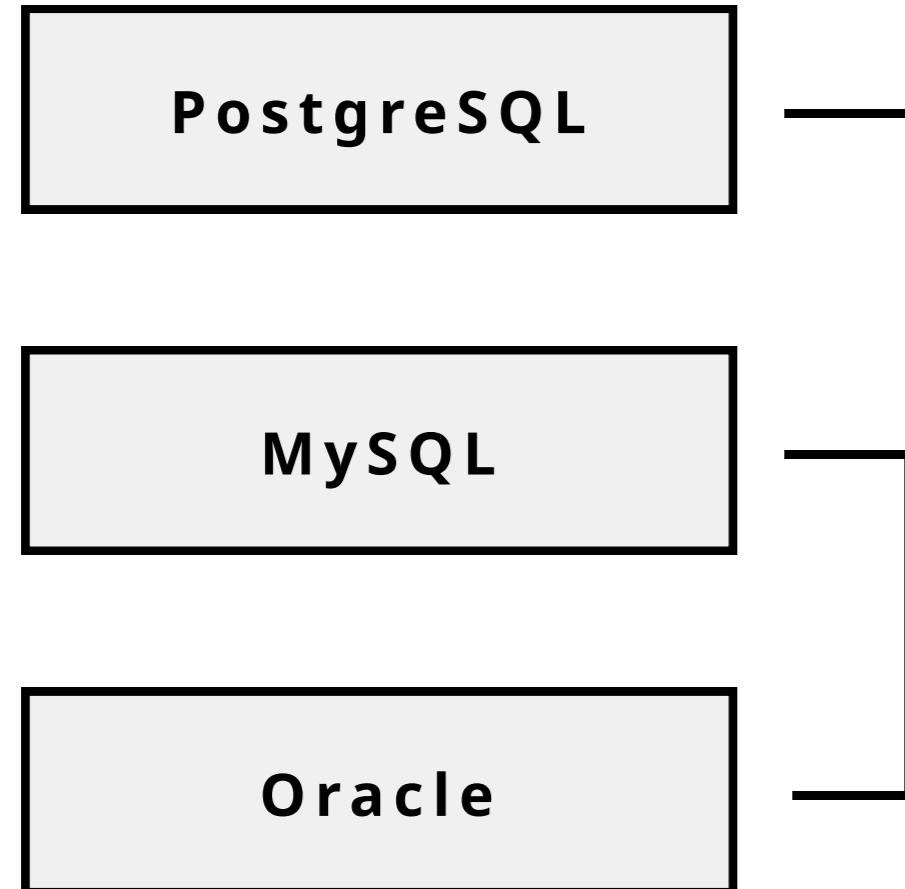
SQL / DBMS 의 사용 방식



DBMS 란 무엇일까요?

<https://hongong.hanbit.co.kr/%EB%8D%B0%EC%9D%B4%ED%84%B0%EB%B2%A0%EC%9D%B4%EC%8A%A4-%EC%9D%B4%ED%95%B4%ED%95%98%EA%B8%B0-database db-dbms-sql%EC%9D%98-%EA%B0%9C%EB%85%90>

SQL / RDBMS vs ORDBMS



SQL provider

ORDBMS
Object
Relationship

RDBMS는 **객체 - 관계형 데이터 모델**을 기반으로 하는 관계형 데이터베이스 관리 시스템입니다.
SQL 언어를 사용하여 접근할 수 있으며, **테이블**, 행과 열로 이루어진, 을 기반으로 설계되었습니다.

객체, 배열 과 이에서 파생된 수많은 복잡한 자료구조를 처리할 수 있습니다.

RDBMS
Relationship

RDBMS는 **관계형 데이터 모델**을 기반으로 하는 관계형 데이터베이스 관리 시스템입니다.
SQL 언어를 사용하여 접근할 수 있으며, **테이블**, 행과 열로 이루어진, 을 기반으로 설계되었습니다.

문자, 숫자, 날짜, 금액 과 같은 단순한 정보만 처리할 수 있습니다.

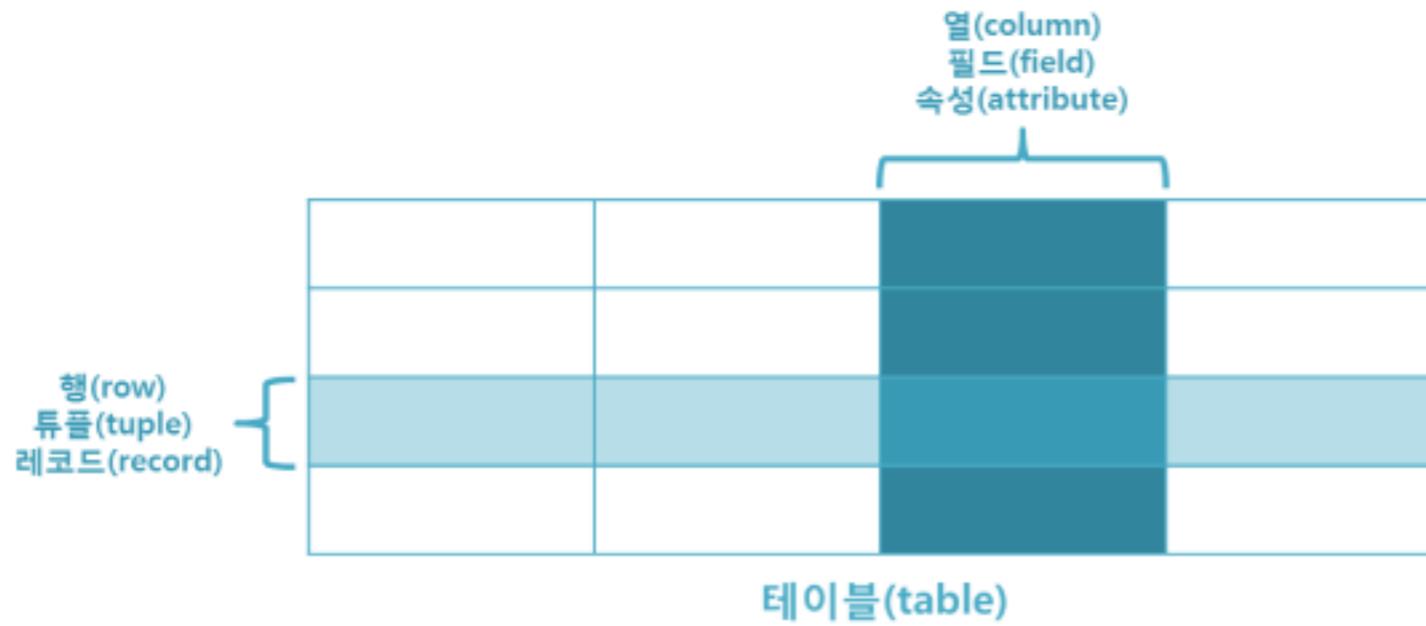


MySQL / PostgreSQL

PostgreSQL / 테이블이란 ?

행과 열로 이루어진 **테이블(표)**로 구성된 데이터베이스

행과 행, 열과 열, 테이블과 테이블 등 다양한 **관계**들을 이용하여 요구사항을 달성



PostgreSQL / RMDBS 의 특징

1. 데이터의 분류, 정렬, 탐색 속도가 빠릅니다.
2. 오랫동안 사용된 만큼 신뢰성이 높고, 어떤 상황에서도 데이터의 무결성을 보장해 줍니다.
3. 기존에 작성된 스키마를 수정하기가 어렵습니다.
4. 데이터베이스의 부하를 분석하는 것이 어렵습니다.

PostgreSQL / RMDBS 의 특징

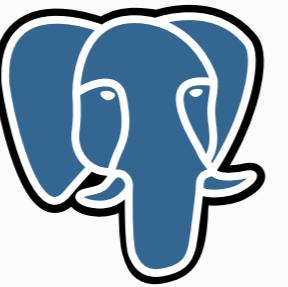
무결성 ?

- 정확성, 일관성, 유효성이 유지되는 것, 단순하게 생각해보면 그냥 결점 없는 데이터, 을 의미합니다.

스키마 ?

- 데이터베이스를 구성하는 단위로 SQL 에서는 테이블의 구성요소들을 포함합니다.
- 테이블에 어떤 행 이 존재하고 각 행의 자료형과 범위가 어디까지인지 등을 포함합니다.
- 자료형 : 숫자 문자 날짜 화폐 등
- 범위(제약조건) : NOT NULL, UNIQUE 등

PostgreSQL / 사례로 살펴보는 (O)RDBMS



쇼핑몰

PostgreSQL / 요구사항 1 - 유저 정보 담기!

아래 정보를 담아주세요!

유저이름	username
비밀번호	password
자기소개	desription
핸드폰 번호	phone_number
주소	address
상세 주소	address_detail
우편 번호	address_post_number

PostgreSQL / 요구사항에 맞게 테이블 설계 해보기

아래 정보를 담아주세요!

기본 정보	유저이름	username
	비밀번호	password
부가 정보	자기소개	desription
	핸드폰 번호	phone_number
	주소	address
	상세 주소	address_detail
	우편 번호	address_post_number

PostgreSQL / 요구사항에 맞게 테이블 세분화 해보기

아래 정보를 담아주세요!

기본 정보

유저이름	username
비밀번호	password

개인 정보

자기소개	desription
핸드폰 번호	phone_number

주소 정보

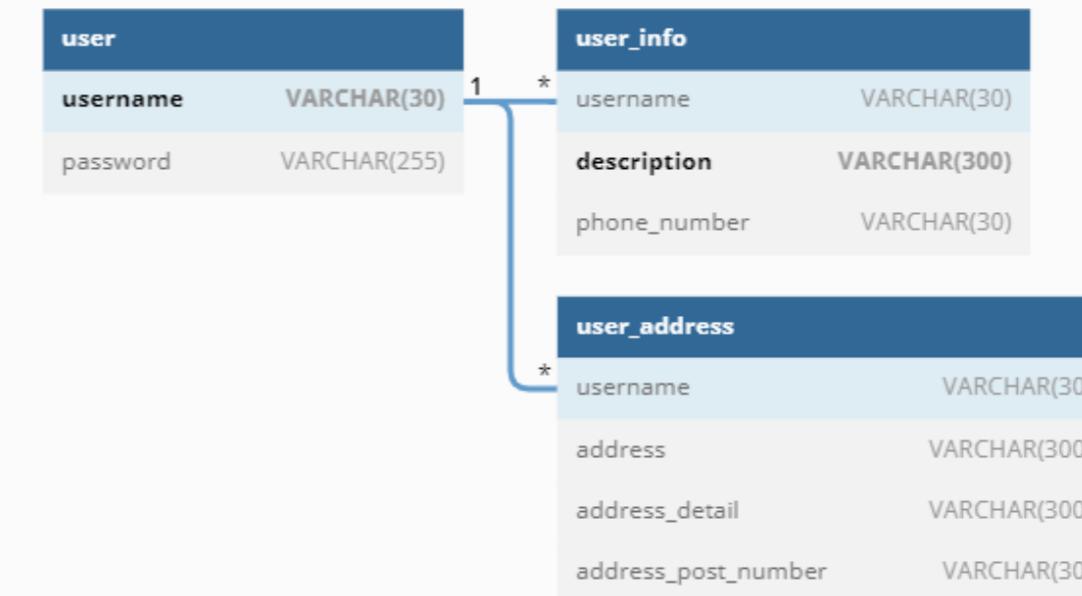
주소	address
상세 주소	address_detail
우편 번호	address_post_number

PostgreSQL / 다이어그램으로 본 테이블 구조

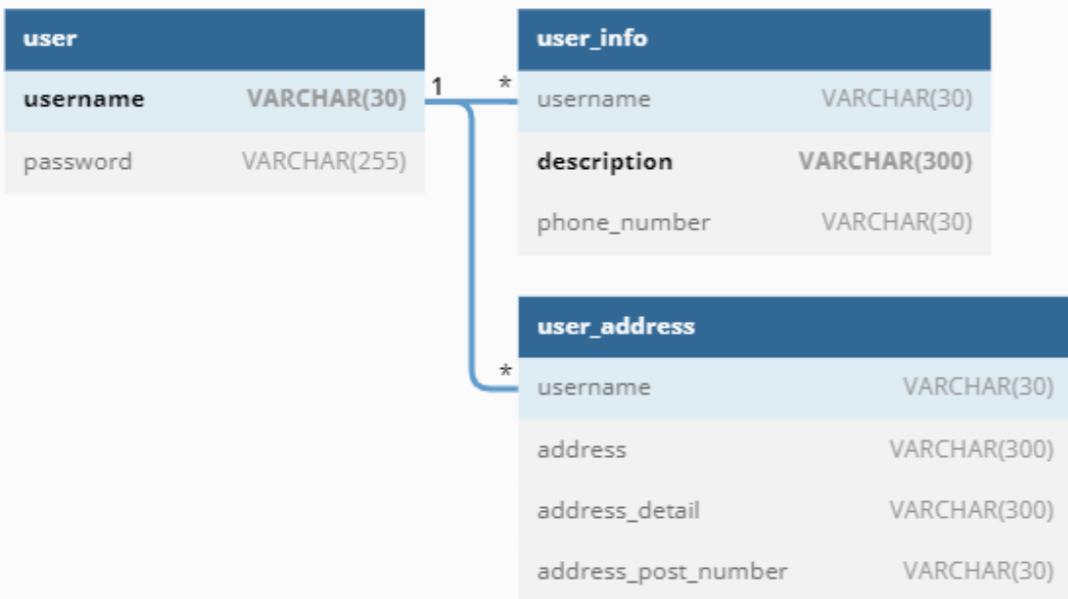
하나의 테이블

user	
username	VARCHAR(30)
password	VARCHAR(255)
description	VARCHAR(300)
phone_number	VARCHAR(30)
address	VARCHAR(300)
address_detail	VARCHAR(300)
address_post_number	VARCHAR(30)

N 개의 테이블, 그리고 서로의 관계



PostgreSQL / N 개의 테이블로 인한 이슈



N 개의 테이블, 그리고 서로의 관계



N 개의 테이블, 그리고 N 번 꺼내기?

PostgreSQL / N 개의 테이블로 인한 이슈

이름, 비밀번호, 주소, 상세 주소, 우편 번호 꺼내기!

N 개의 테이블로 인한 이슈

Join

2 개의 테이블에 2 번 접속해서 꺼내기 ?

무엇을	SELECT username, password
어디서	FROM user
어떤 친구를	WHERE username = 'unchapteree'd';
무엇을	SELECT address, address_detail, address_post_number
어디서	FROM user_address
어떤 친구를	WHERE username = 'unchapter ed';

2 개의 테이블에 1 번 접속해서 꺼내기 ?

PostgreSQL / N 개의 테이블로 인한 이슈

SELECT * FROM user WHERE username = 'A';

username	password
A	password_string
(1개 행)	

SELECT * FROM user_address WHERE username = 'A';

username	address	address_detail	address_post_number
A	address	address_detail	address_post_number
(1개 행)			

SELECT * FROM user, user_address WHERE user.username = 'A';

username	password	username	address	address_detail	address_post_number
A	password_string	A	address	address_detail	address_post_number
A	password_string	B	address	address_detail	address_post_number
A	password_string	C	address	address_detail	address_post_number
(3개 행)					

PostgreSQL / N 개의 테이블로 인한 이슈

```
SELECT * FROM user WHERE username = 'A';
```

username	password
A	password_string
(1개 행)	

```
SELECT * FROM user_address WHERE username = 'A';
```

username	address	address_detail	address_post_number
A	address	address_detail	address_post_number
(1개 행)			

```
SELECT * FROM user, user_address  
WHERE user.username = user_address.username  
AND user.username = 'A';
```

username	password	username	address	address_detail	address_post_number
A	password_string	A	address	address_detail	address_post_number
(1개 행)					

PostgreSQL / 연속된 작업이 실패했을 경우

회원 탈퇴 후 등록된 상품 전부 제거

연속된 작업이 실패했을 경우

Transaction

BEGIN;

변경 사항 반영을 대기 처리해 둠

성공

DELETE FROM user WHERE username = 'uncahptered';

성공

DELETE FROM user_info WHERE username = 'uncahptered';

실패

DELETE FROM user_address WHERE username = 'uncahptered';

이미 지워진 정보는 어떻게 ?

BEGIN;

변경 사항 반영을 대기 처리해 둠

성공

DELETE FROM user WHERE username = 'uncahptered';

성공

DELETE FROM user_info WHERE username = 'uncahptered';

?

DELETE FROM user_address WHERE username = 'uncahptered';

성공

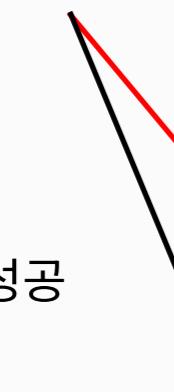
ROLLBACK;

모든 변경 내역 취소

COMMIT;

모든 변경 내역 반영

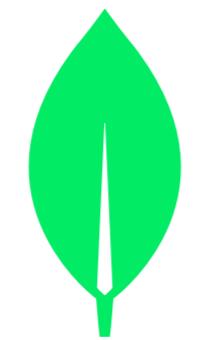
실패



회원 탈퇴 후 등록된 상품 전부 제거

DB 안에 담긴 정보가 너무 많을 때 ?

Index / Table Split



MongoDB

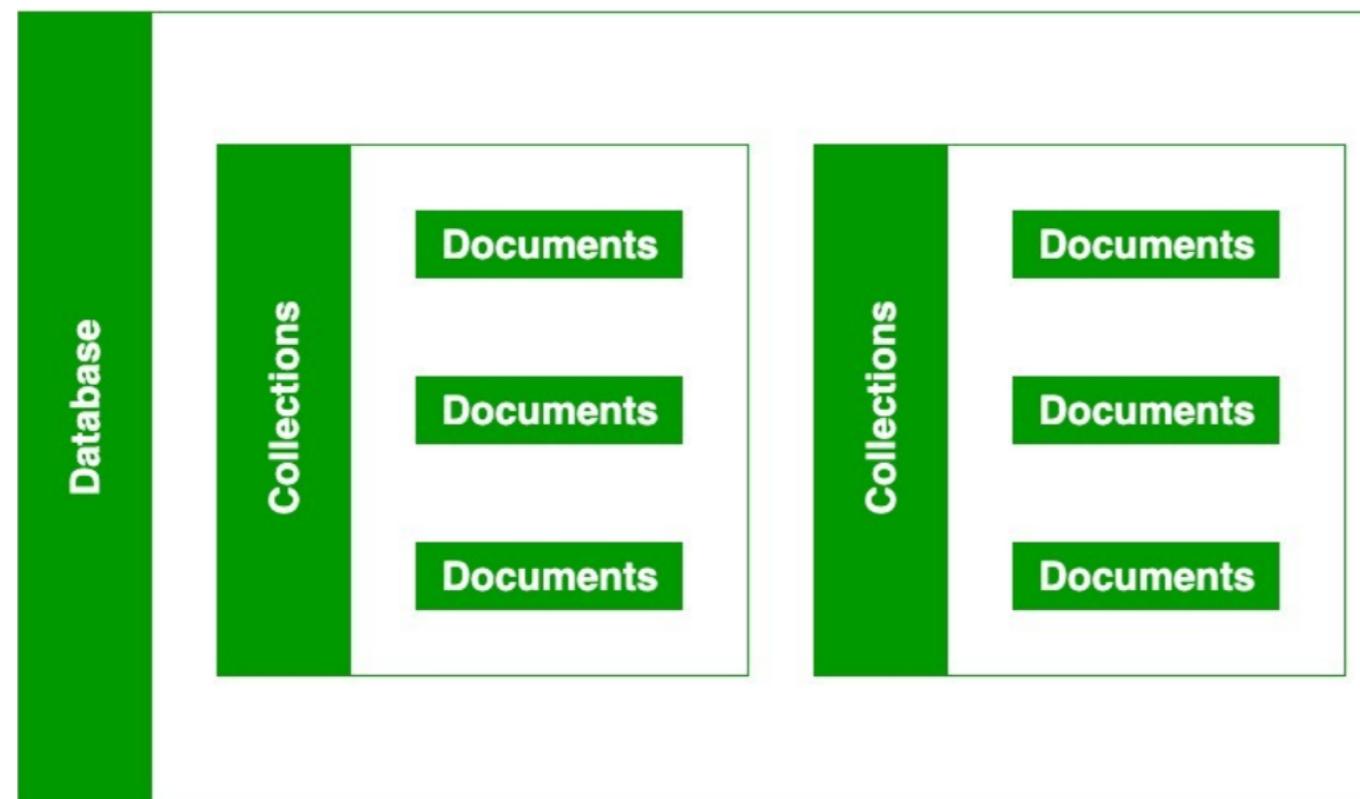
문서 기반,
MongoDB

MongoDB / 문서 기반 데이터베이스 란?

문서 를 기반으로 기반으로 하여, 특정한 문서를 저장 및 제공하는데 높은 성능을 보이는 데이터베이스입니다.
XML 을 대체하면서 일종의 표준으로 떠오르게 된 **JSON** 과 매우 유사한 **BSON** 을 사용하고 있습니다.

기본 설정으로 문서를 생성하게 되면, 검색에 최적화 가 되어있는 `_id` (`ObjectId`) 를 제공해줍니다.
따라서, 별도의 이유가 없다면 해당의 값을 탐색에 사용하는 것이 효율적입니다.

비슷한 문서의 묶음을 **컬렉션**, **collection** 이라고 부릅니다.



SQL vs NoSQL (MySQL vs. MongoDB)
<https://siyoon210.tistory.com/130>

MongoDB – Database, Collection, and Document
<https://www.geeksforgeeks.org/mongodb-database-collection-and-document/>

MongoDB / 사례로 살펴보는 문서 기반 데이터베이스



MongoDB 블로그

MongoDB / 요구사항에 맞게 문서 설계 해보기

아래 정보를 담아주세요!

게시글_제목 post_title

게시글_내용 post_context

게시글_작성자 post_author

댓글_내용 comment_context

댓글_작성자 comment_author

두 개의 Schema 를 만들고, 하나의 Schema 에 다른 친구의 _id 를 기록?

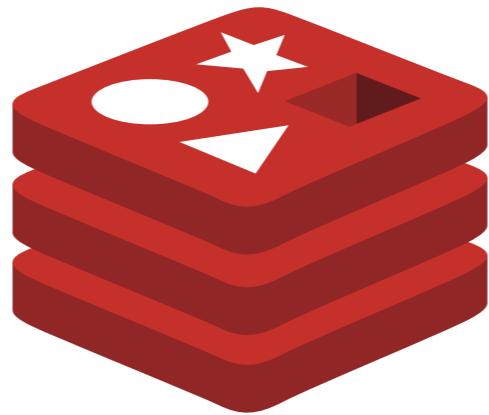
```
const postSchema = new mongoose.Schema({
    post_title: { type: String },
    post_content: { type: String },
    post_author: { type: String },
    post_comment_list: [{ type: ObjectId, ref: 'Comment' }]
});

const commentSchema = new mongoose.Schema({
    comment_context: { type: String },
    comment_author: { type: String }
});
```

하나의 Schema 를 만들고, 댓글은 해당 Schema 안에 내장

```
const postSchema = new mongoose.Schema({  
    post_title: { type: String },  
    post_content: { type: String },  
    post_author: { type: String },  
    post_comment_list: [  
        comment_context: { type: String },  
        comment_author: { type: String }  
    ]  
});
```

**Transaction
Index, ...**



메모리 기반,
Redis

Redis / 메모리 기반 데이터베이스 란?

메모리 영역에서 **쓰기/읽기** 작업이 진행되며, redis 가 종료되면 내부의 정보가 사라지는 특성을 가집니다.
또한, **Key - Value** 의 구조로 정보를 저장하기 때문에, 정보의 수와 무관하게 폭발적인 속도를 보장해줍니다.

휘발성을 가지고 있기 때문에, 주저장소 보다는 **캐싱** 을 통한 성능 개선이 주된 목적입니다.

일반적인 사용 사례는 **읽기 전략**, **쓰기 전략** 등이 있을 수 있으며 사용 간에 몇 가지 주의사항이 필요합니다.

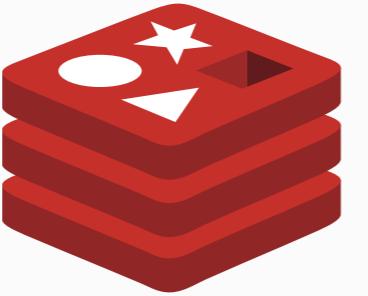
특이한 사용 사례로는 **사용자 일일 방문자 수** 를 비트 연산을 통해서 매우 효율적으로 기록할 수 있는 경우가 있습니다.

<https://any-ting.tistory.com/91>

<https://www.geeksforgeeks.org/difference-between-redis-and-mongodb/>

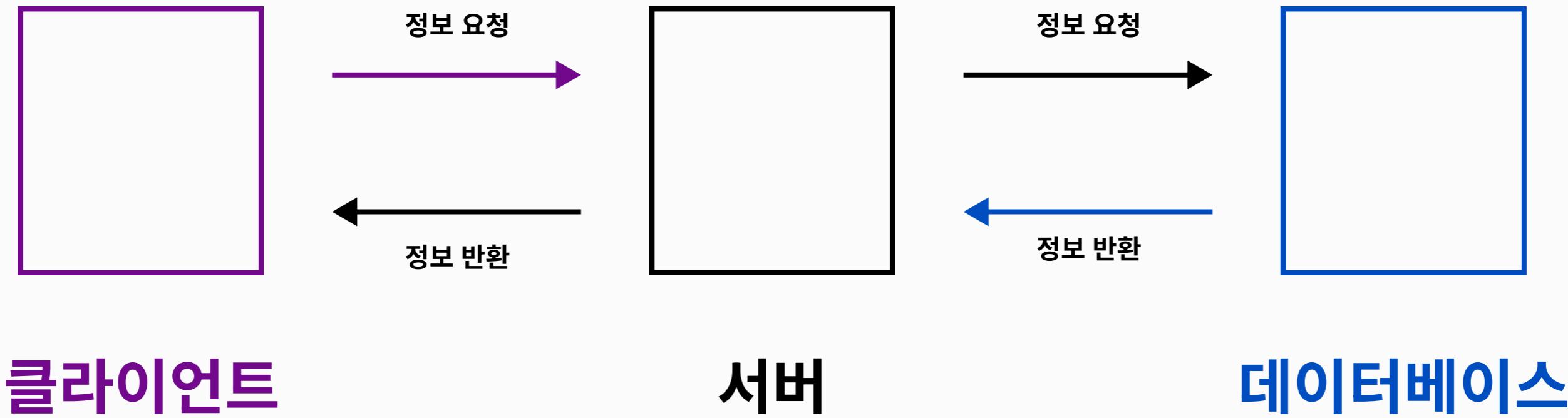
#:
#:~:text=Redis%20stands%20for%20Remote%20Dictionary,in%20ANSI%20and%20C%20languages.

Redis / 사례로 살펴 보는 메모리 기반 데이터베이스



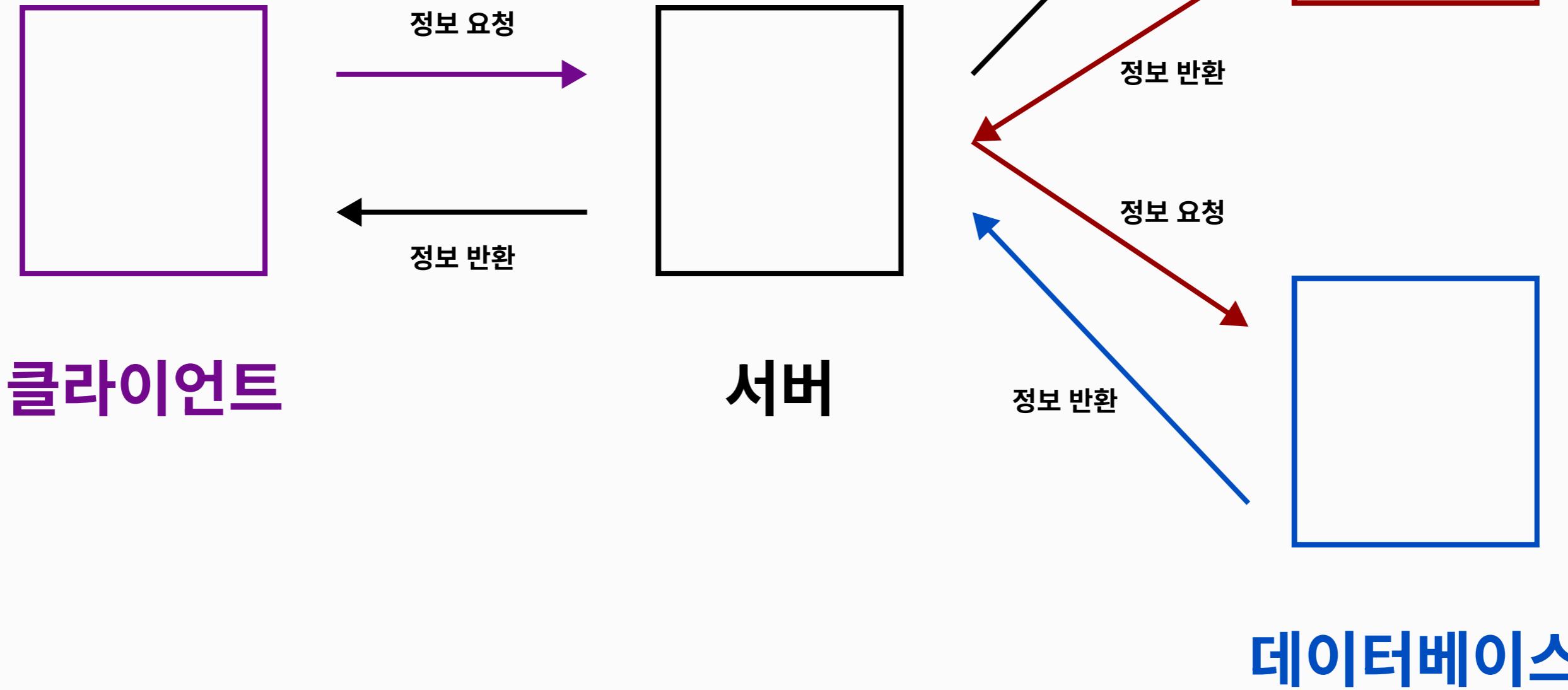
메모리 캐시

Redis / 기본적인 서버 통신



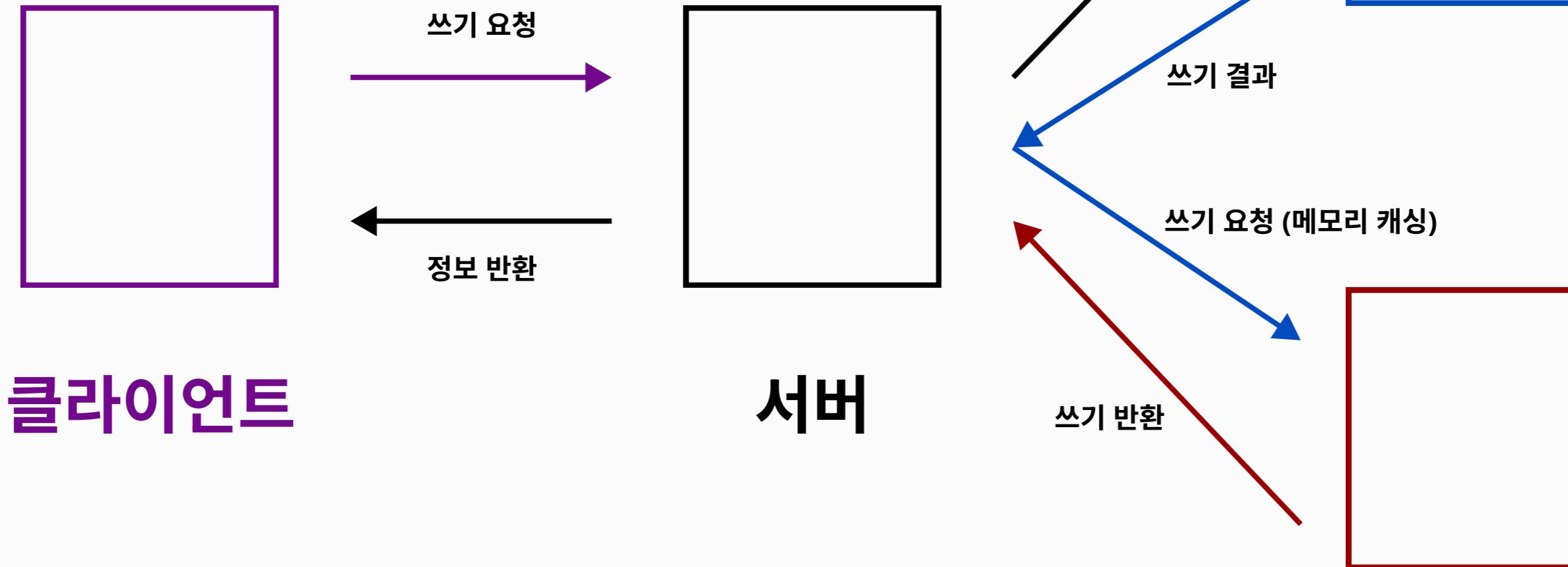
Redis / 읽기 작업을 개선한 서버 통신

읽기 전략



Redis / 읽기 작업을 개선한 서버 통신

쓰기 전략



Redis

The New York Times



HALFBRICK

duolingo



lyft

venmo

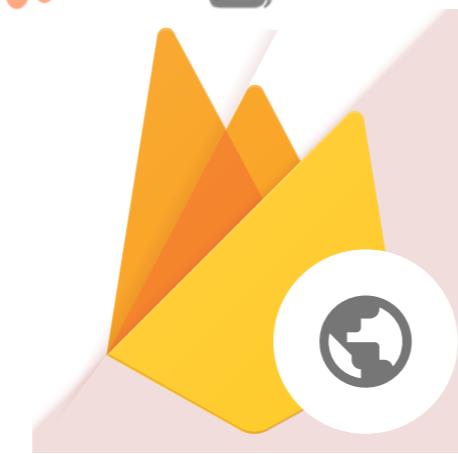
The Economist

trivago

Ctrip

wattpad

GAMELOFT



Google,
Firebase

https://firebase.google.com/?gclid=Cj0KCQjw852XBhC6ARIsAJsFPN0fjkEA5ebw0107Qwc6-GPd6QRWmn4HT-M59n14ZbquOFCVcTJz09UaAvyIEALw_wcB&gclsrc=aw.ds

NETFLIX

Uber

slack

twilio

Microsoft



복합 서비스,
Elastic Search

https://www.elastic.co/kr/?ultron=B-Stack-Trials-APJ-Exact&gambit=Stack-Core&blade=adwords-s&hulk=paid&Device=c&thor=elastic%20search&gclid=Cj0KCQjw852XBhC6ARIsAJsFPN1GunItUGuoG1koNCcSABFnW87p9Fk_5D-24Aq0ZNIDM8D0-XyqyPsaAsGJEALw_wcB

Disney

 **Dropbox**

Disney+, 매일 수십억 개의 고객
작업을 수집하여 시청자 경험을
개선 »



Snap Inc., Amazon DynamoDB로 대
기 시간 중간값을 20% 단축 »

zoom

Zoom Video Communications, Inc.,
1천만에서 3억 명으로 급증한 모임 참
가자 관리 »



문서 기반,
DynamoDB

<https://aws.amazon.com/ko/dynamodb/>

**Cox
AUTOMOTIVE™**

COX Automotive, 보다 뛰어난 360도
고객 데이터 가시성 구축 »

 **GAMES|24
SEVEN**

Games24x7, 토너먼트에서 플레이어
사기 및 승부 조작 탐지 가속화 »



그래프 기반,
Neptune

<https://aws.amazon.com/ko/neptune/>

ADP®

ADP, 인적 자본 관리(HCM) 솔루션 비
용 절감»

 **Careem**

Careem, Amazon Neptune를 사용하
여 커밋 전에 사기 차단 »