2022 미래차 충전인프라구축운영 전문인력 양성교육
전기차 충전기 빅데이터 심화

NoSQL

2022.11

한국기술교육대학교 KOREATECH
한국자동차환경협회 Korea Automobile Environmental Association
아주자동차대학교 AJOU MOTOR COLLEGE

차례

NoSQL ?

- "NoSQL 데이터베이스" 수평적 확장성
  - 비관계형 데이터베이스
  - "non SQL(비 SQL)", "not only SQL(SQL만을 사용하지 않는)" 약자로 생각
  - 관계형 데이터베이스 이외의 형식으로 데이터를 저장하는 데이터베이스
  - 대부분 클러스터에서 실행할 목적으로 만들어졌기 때문에 관계형 모델을 사용하지 않음. (NoSQL 데이터 모델 중 하나인 그래프 데이터베이스는 관계형 데이터베이스와 비슷한 분산 모델 사용)
  - 스키마 없이 동작. 구조에 대한 정의를 변경할 필요 없이 데이터베이스 레코드에 자유롭게 필드를 추가
  - RDBMS와 달리 테이블간 관계를 정의하지 않음. 데이터 테이블은 그냥 하나의 테이블이며 테이블 간의 관계를 정의하지 않아 일반적으로 테이블 간 Join 도 불가능

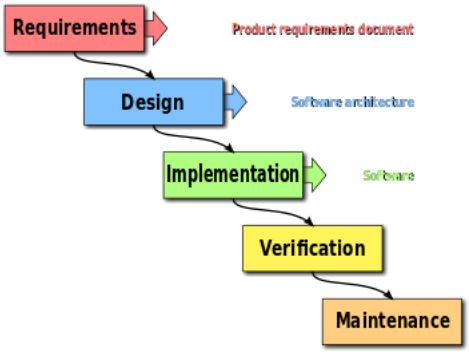
• No SQL - 수평적 확장성: 서버를 늘리기만 하면 스케일이 늘어남. 응답속도 보장
• RDBMS - 수직적 확장성: SSD, CPU 업그레이드 하거나, 확장하는데 돈과 작업 시간 소요

참고자료

2022 미래차 충전인프라구축운영 전문인력 양성교육

SQL(Structured Query Langauge) ?

- 데이터가 고정된 열과 행을 가지고 있는 테이블에 저장되는 관계형 데이터베이스와 상호 작용할 때 SQL를 사용
- 1970년 초반, 스토리지가 매우 비쌌기 때문에 소프트웨어 엔지니어링들은 데이터 중복을 줄이기 위해 데이터베이스를 정규화
- (1970년대) 소프트웨어 개발에 있어 폭포수(Waterfall) 모델을 따름



NoSQL 데이터베이스 사용해야 하는 이유?

- 유연성
  - 유연한 스키마를 제공, 빠르고 반복적인 개발 가능
- 확장성
  - 고가의 강력한 서버 → 분산형 하드웨어 클러스터를 이용해 확장
- 고성능
  - 특정 데이터 모델(문서, 키 값, 그래프 등) 및 엑세스 패턴에 대해 최적화되어 관계형 데이터베이스를 통해 유사한 기능을 충족하려 할 때 보다 뛰어난 성능을 보임
- 고기능성
  - 각 데이터 모델에 맞추어 특별히 구축된 뛰어난 기능의 API와 데이터 유형을 제공

2022 미래차 충전인프라구축운영 전문인력 양성교육

NoSQL 특징

- RDBMS와 달리 데이터 간의 관계를 정의하지 않음
  - RDBMS는 데이터 관계를 외래키(Foreign Key) 등으로 정의하고 JOIN 연산을 수행할 수 있지만, NoSQL는 JOIN연산이 불가능
- RDBMS에 비해 대용량의 데이터를 저장
  - 페타바이트(Petabyte, PB, 10^{15}) 급의 대용량 데이터 저장
- 분산형 구조
  - 여러 곳의 서버에 데이터를 분산, 복사 저장해 특정 서버에 장애가 발생했을 때도 데이터 유실 혹은 서비스 중지가 발생하지 않도록 함
- 고정되지 않은 테이블 스키마를 가짐
  - RDBMS와 달리 테이블의 스키마가 유동적
  - 데이터를 저장하는 칼럼이 각기 다른 이름과 다른 데이터 타입을 갖는 것이 허용

NoSQL 장점

- RDMBS에 비해 저렴한 비용으로 분산처리와 병렬 처리 가능
- 비정형 데이터 구조 설계로 설계 비용 감소
- 빅데이터 처리에 효과적
- 가변적인 구조로 데이터 저장이 가능
- 데이터 모델의 유연한 변화가 가능



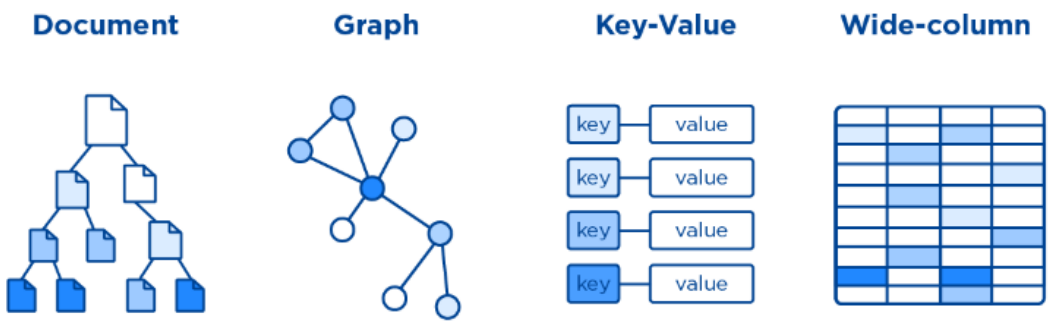
2022 미래차 충전인프라구축운영 전문인력 양성교육

NoSQL 단점

- 데이터 업데이트 중 장애가 발생하면 데이터 손실 발생 가능
- 많은 인덱스를 사용하려면 충분한 메모리가 필요. 인덱스 구조가 메모리에 저장
- 데이터 일관성이 항상 보장되지 않음



NoSQL 의 종류



NoSQL 종류

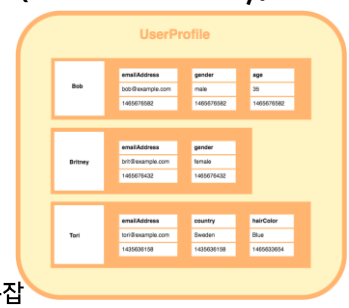
1. 키-값(Key/Value) Database

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

- KEY-VALUE 하나의 묶음
- 구조가 단순, 속도 빠름, 분산 저장 용이
- Key안에 (COLUMN, VALUE) 형태로 된 여러 개의 필드, 즉 COLUMN FAMILIES 가짐
- 주고 SERVER CONFIG, SESSION CLUSTERING등에 사용되고 액세스 속도 빠름
- SCAN에는 용이하지 않음

Ex) Redis, Oracle NoSQL Database, VoldeMorte

2. 컬럼패밀리(Wide-Column) Database/Big Table Database(=Ordered Key/Values)

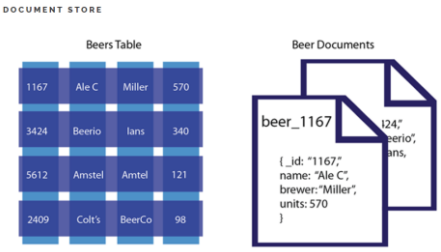


- 가장 복잡
- 행마다 키와 해당 값을 저장할 때마다 각각 다른값의 다른 수의 스키마를 가질 수 있음(조인할 필요 없이 원하는 컬럼을 모아서 하나의 로우에 담음)
- Wide Column Database는 대량의 데이터의 압축, 분산처리, 집계쿼리 및 쿼리 동작 속도 그리고 확장성이 뛰어남

Ex) Hbase, GoogleBigTable, Vertica

NoSQL 종류

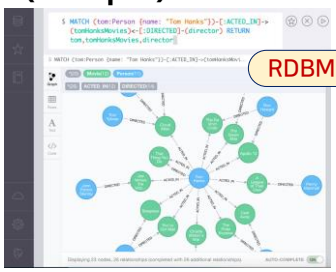
3. 문서(Document) Database



- 문서는 여러 속성을 저장 가능
- 테이블의 스키마가 유동적, 즉 레코드마다 각각 다른 스키마를 가짐
- 보통 XML, JSON, YAML과 같은 Document를 이용해 레코드를 저장
- 트리형 구조로 레코드를 저장하거나 검색하는데 효과적

Ex) MongoDB, CouchDB, Azure Cosmos DB

4. 그래프(Graph) Database



- 테이블 노드(Node)로 표현하며, 노드 사이의 관계(Relationship)를 엣지(direction, type, start node, end node)로 표현
- RDBMS 보다 성능이 좋고 유연하며, 유지보수에 용이한 것이 특징
- SNS, 교통망, 전력망 등 연결 관계가 복잡한 형태의 데이터를 다루는데 용이

Ex) Neo4j, BlazeGraph, OrientDB

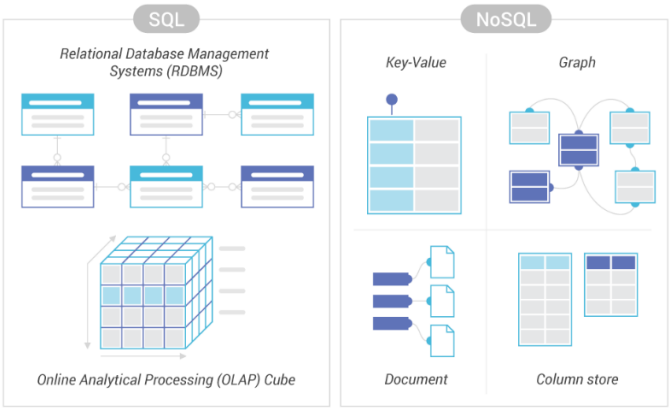
NoSQL의 종류 : 키-값 데이터베이스

- 키-값 DB
  - Key-Value 형태를 가짐
  - 데이터를 식별하기 위해 key값을 알면 연결된 데이터를 확인할 수 있는 구조
- 키-값 DB의 구성요소
  - 키
    - 값을 가져오기 위한 식별자, 이름공간 내에서 고유
    - 이름공간: 키-값을 묶어둔 것.
    - 키를 통해 값을 찾아와야 하는 명명 규칙을 세워서 의미있는 값으로 만듦
    - 키값을 처리하기 위해 해시(Hash) 함수를 사용
  - 값
    - 키-값 DB에서 값의 데이터 타입에는 큰 제약 없음
    - 정수, 실수, 문자열, BLOB, JSON객체, 이미지, 오디오, List 등 ...

NoSQL의 종류 : 키-값 데이터베이스

- 단점과 한계점
  - 값 검색에 대한 한계
    - 키를 통해 값을 검색하고 수정 및 삭제를 진행할 수 있으나, 값을 기준으로 한 검색은 어려움 ➔ 방법이 없는 것은 아니나 키를 기준으로 한 검색보다는 효율성이 떨어짐
    - 값에 대한 텍스트 검색을 하는 방법(Riak)
    - 키만 사용해서 검색하는 경우 보조 인덱스(Secondary Index)를 사용하여 값의 속성을 인덱스로 만들 수 있음
  - 범위 질의를 지원하지 않음
    - 시작일 ~ 종료일 사이의 데이터를 검색하거나 특정 범위 내의 데이터를 검색하는 기능 없음
    - 보조 인덱스를 하용해서 기능을 보완
  - RDBMS의 SQL처럼 활용성 있는 질의 언어가 없음
    - 일부 DB의 경우 XML, JSON등의 데이터 구조를 지원하므로 이를 이용해서 검색 가능
- 설계 시 고려 사항
  - 단어 사용 시 애매하지 않은 단어를 이용해서 명확하게 표현
  - 키를 만들기 위해 공통 구분자를 사용(일반적으로 :(콜론))
  - 구간 검색을 쉽게 하기 위해 범위를 기반으로 하는 용어 사용(날짜나 정수형)
  - 키는 가능한 짧게

SQL(관계형) vs. NoSQL 데이터베이스 비교



	NoSQL	RDBMS
장단점	데이터 무결성, 정합성 보장하지 않음 비정형, 반정형 데이터 처리	데이터 무결성 보장(CA) 정규화된(정형) 데이터 처리 확장성 이슈, 분산환경 부적합
특징	약한 consistency schema가 없거나 변경이 용이	JOIN ACID
use case	대량 데이터 처리 빠른 성능 요구	중요한 트렌젝션처리(금융) 요구되는 경우

MongoDB특징

- 도큐먼트 데이터베이스
  - 도큐먼트: HTML과 같은 특정 형식의 태그 구조 의미
  - JSON(JavaScript Object Notation) 형식으로 데이터를 관리
  - NoSQL 데이터베이스 중 도큐먼트 데이터베이스로 분류
  - 도큐먼트는 MongoDB가 데이터를 저장하는 최소 단위
  - 도큐먼트는 필드와 값의 쌍으로 구성, 관계를 갖는 데이터를 중첩 도큐먼트와 배열을 사용하여 1개의 도큐먼트로 표현
  - 데이터 입출력 시에는 JSON 형식의 도큐먼트를 사용하나 데이터베이스 저장시에는 이진 포맷으로 인코딩한 BSON(Binary JSON) 형식의 도큐먼트로 변환되어 저장
- 유연한 스키마
  - 스키마의 선언 없이 필드의 추가와 삭제가 자유로운 Schema-less 구조
  - 관계형 데이터베이스는 테이블 내 모든 로우(Row)의 칼럼 집합이 동일하고 같은 칼럼은 동일한 데이터 타입을 갖는 정형 스키마
  - 컬렉션 내 모든 도큐먼트들의 필드 집합이 동일하지 않고 같은 필드라도 데이터 타입이 다를 수 있는 비정형 스키마



2022 미래차 충전인프라구축운영 전문인력 양성교육

MongoDB특징

- 비 관계형 데이터베이스
  - 관계형 데이터베이스의 관계(Relationship) 개념이 없는 비 관계형 데이터베이스
  - 조인(Join)을 지원하지 않으며, 대신 임베디드 방식의 도큐먼트 구조를 사용하거나 레퍼런스 방식의 도큐먼트 구조를 사용한 후 애플리케이션에서 조인해야 함
- 비트랜잭션
  - 트랜잭션을 지원하지 않고 각각의 도큐먼트 단위로 처리
  - 트랜잭션을 지원하지 않으므로 Commit 또는 Rollback 개념이 없으며 모두 Auto Commit으로 처리



JSON 및 BSON

Basis of Comparison	JSON	BSON
Type	Standard file format	Binary file format
Speed	Comparatively less fast	Faster
Space	Consumes comparatively less space.	More space is consumed.
Usage	Transmission of data.	Storage of data.
Characteristics	Key-value pair only used for transmission of data.	Lightweight, fast and traversable.
Structure	Language independent format used for asynchronous server browser communication.	Binary JSON which consist of a list of ordered elements containing a field name, type, and value. Field name types are typically a string.

2022 미래차 충전인프라구축운영 전문인력 양성교육

MongoDB 장점

- Schema-less 구조
  - 다양한 형태의 데이터 저장 가능
  - 데이터 모델의 유연한 변화 가능(데이터 모델 변경, 필등 확장 용이)
- Read/Write 성능이 뛰어남
- Scale Out 구조
  - 많은 데이터 저장이 가능
  - 장비 확장이 간단함
- JSON 구조: 데이터를 직관적으로 이해 가능
- 사용방법이 쉽고, 개발이 편리함

MongoDB 단점

- 데이터 업데이트 중 장애 발생시, 데이터 손실 가능
- 많은 인덱스 사용시, 충분한 메모리 확보 필요
- 데이터 공간 소모가 RDBMS에 비해 많은(비효율적인 Key 중복 입력)
- 복잡한 JOIN사용시 성능 제약이 따름
- 트랜잭션 지원이 RDBMS 대비 미약함
- 제공되는 MapReduce 작업이 Hadoop에 비해 성능이 떨어짐

빅데이터 처리 특화

- Memory Mapped(데이터 쓰기 시에 OS의 가상 메모리에 데이터를 넣은 후 비동기로 디스크에 기록하는 방식)를 사용
- 방대한 데이터를 빠르게 처리 가능
- OS의 메모리를 활용하기 때문에 메모리가 차면 하드디스크로 처리하여 속도가 급격히 느려짐
- 하드웨어적인 측면에서 투자가 필요

MongoDB 불안정성

- 데이터 양이 많을 경우
  - 일부 데이터가 손실 가능성 존재
  - 샤딩의 비정상적인 동작 가능성
  - 레플리카 프로세스의 비정상 동작 가능성



참고자료

2022 미래차 충전인프라구축운영 전문인력 양성교육

SQL?

- Structured Query Language(구조적 질의 언어)의 줄임말
- 관계형 데이터베이스 시스템(RDBMS)에서 자료를 관리 및 처리하기 위해 설계된 언어
- (1970년대)IBM에서 최초 개발
- 현재 SQL표준: ANSI SQL이 정립



SQL 문법의 종류

- DDL(Data Definition Language, 데이터 정의 언어)
  - 각 릴레이션의 정의하기 위해 사용하는 언어(명령어)(CREATE, ALTER, DROP, ...)
- DML(Data Manipulation Language, 데이터 조작 언어)
  - 데이터를 추가/수정/삭제하기 위한, 즉 데이터 관리를 위한 언어 (SELECT, INSERT, UPDATE, ...)
- DCL(Data Control Language, 데이터 제어 언어)
  - 사용자 관리 및 사용자별로 릴레이션 또는 데이터를 관리하고 접근하는 권한을 다루기 위한 언어(GRANT, REVOKE,...)



SQL의 언어적 특성

- SQL은 대소문자 구분 없음
  - 단, 서버 환경이나 DBMS 종류에 따라 데이터베이스 또는 필드명에 대해 대소문자 구분도 하기도 함
- SQL 명령은 반드시 세미콜론(;)으로 끝남
- 고유의 값은 따옴표(")로 감쌈
  예) SELECT * FROM EMP WHERE NAME="James";
- SQL에서 객체를 나타낼 때는 백틱(` `)으로 감쌈
  예) SELECT `COST`, `TYPE` FROM `INVOCE`;
- 주석은 일종의 도움말, 주석 처리된 문장은 프로그램에서 동작하지 않음 '—'
  예) -SELECT * FROM EMP; 이 쿼리는 실행되지 않음
- 여러 줄 주석은 /* */로 감쌈
  예)

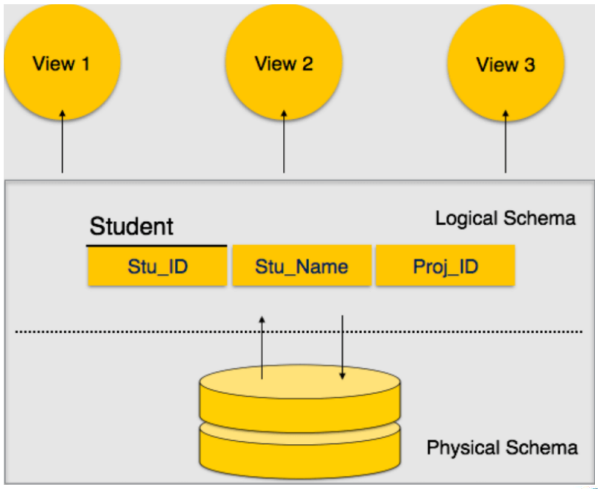
```
/*
  SELECT * FROM EMP WHERE EMPID = (SELECT * FROM EMP WHERE NAME='홍길동')
*/
```



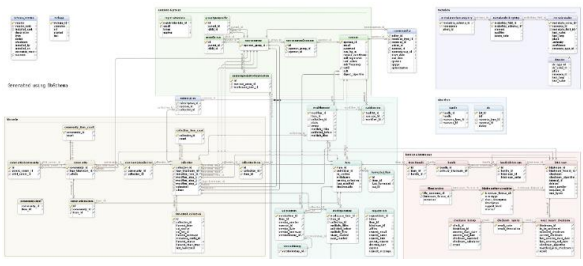
참고자료

스카마(Schema)란?

- "데이터의 구조 " "데이터베이스의 설계"
- 논리적 보기를 나타내는 일종의 골격 구조
- 데이터베이스의 구조와 제약조건에 관한 전반적인 명세를 기술한 것
- 데이터베이스를 구성하는 데이터 객체(Entity), 개체의 특성을 나타내는 속성(Attribute), 개체 사이에 존재하는 관계(Relationship) 및 데이터 조작 시 데이터 값들이 갖는 제약 조건 등에 관하여 기술



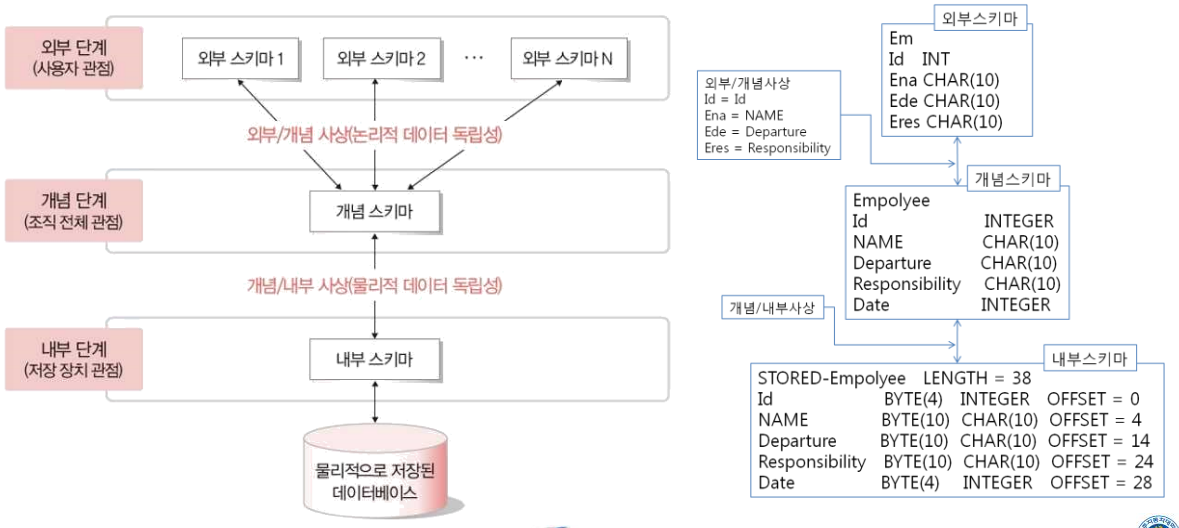
스카마 예시



출처:
 https://wiki.lyrasis.org/display/DSDOC5x/Storage+Layer?preview=/45548226/68355659/dspace5.png

출처:https://drawsql.app/templates

스키마 구조



2022 미래차 충전인프라구축운영 전문인력 양성교육

스키마의 3단계 구조

- 외부 스키마(서브 스키마)
  - 외부(사용자)에서 바라보는 스키마 의미
  - 사용자들이 사용할 데이터를 보여주는 것이기 때문에 추상화가 되어 있고, 여러 사용자가 바라보는 관점에 따라 여러 스키마가 존재→(예) 네이버 데이터베이스: 네이버 메일 스키마, 네이버 카페 스키마, 네이버 블로그 스키마 등…
  - 사용자는 데이터베이스에서 데이터를 사용하는 사람이므로 응용 프로그래머라 볼 수 있음
  - 사용자는 어떤 데이터가 필요한지를 결정하기 때문에 쿼리(Query)를 이용해서 데이터 조작 가능

스키마의 3단계 구조

- 개념스키마
  - 전체적인 개념 정의
  - 전체 데이터베이스가 어떤 구조로 되었는지 구체적으로 어떤 데이터가 있고, 그 데이터들은 어떤 테이블에 있고, 각 테이블마다 어떤 관계가 존재하는지를 정의
  - 데이터베이스 자체의 전체적인 구조를 확인하기 때문에 개념 스키마를 확인하는 사람은 데이터베이스 관리자



2022 미래차 충전인프라구축운영 전문인력 양성교육

스키마의 3단계 구조

- 내부스키마
  - 실제 데이터의 내부를 정의
  - 데이터 내부 정의 – 데이터의 필드 이름이 무엇이고, 해당 필드는 몇 Byte이며 인덱스가 있는지 등을 정의
  - 데이터를 물리적으로 어떻게 저장할지에 대해 정의한 것이므로 저장스키마라 부름
  - 물리적 저장장치의 입장으로 바라보기 때문에 내부 스키마를 확인하는 사람은 시스템 프로그래머



참조

- https://www.mongodb.com/ko-kr/nosql-explained
- https://code-lab1.tistory.com/53
- NoSQL 철저 입문 : 댄 설리번, 2015, 길벗
- 대용량 데이터 처리를 위한 Real MongoDB : 이성욱, 2018, 위키북스
- MongoDB in Action(몽고 디비 인 액션) 2nd Edition : 카일 뱅커, 2018, 제이펍
- https://docs.mongodb.com/manual/
- https://ko.wikipedia.org/wiki/NoSQL

2022 미래차 충전인프라구축운영 전문인력 양성교육

참조

- https://www.flaticon.com/free-icon/graphic-editor_3940046?term=monitor&page=1&position=69
- https://www.flaticon.com/free-icon/settings_646437?term=gear&page=4&position=21
- https://www.flaticon.com/free-icon/employee_554857?term=person&page=1&position=73
- https://www.flaticon.com/free-icon/man_2922510?term=person&page=1&position=18
- https://inpa.tistory.com/entry/DB-%F0%9F%93%9A-NoSQL-%EA%B0%9C%EB%85%90-%EC%A0%95%EB%A6%AC
- https://im-designloper.tistory.com/67
- https://edu.goorm.io/learn/lecture/15413/%ED%95%9C-%EB%88%88%EC%97%90-%EB%81%9D%EB%82%B4%EB%8A%94-sql/lesson/767683/sql%EC%9D%B4%EB%9E%80