

- 데이터베이스 기초
 - DB, 테이블, 주요키
- MySQL 설치
- SQL 기초
 - select
 - insert
 - update
 - delete
 - create table

- DB(DataBase)
 - 논리적으로 연관된 데이터를 모아 일정한 형태로 저장해 놓은 것
 - 빠른 탐색과 검색을 위해 조직된 데이터의 집합체
- DBMS
 - Database Management System
 - 데이터베이스를 관리하기 위한 시스템
 - 주요 기능
 - 데이터의 추가/조회/변경/삭제
 - 데이터의 무결성(integrity) 유지
 - 트랜잭션 관리
 - 데이터의 백업 및 복원
 - 데이터 보안

RELATIONAL DATABASE

Example of SQL

ORACLE



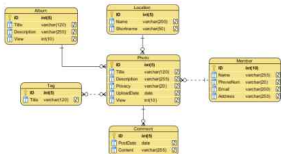
PostgreSQL

MySQL

Microsoft
SQL Server



SQLite



NON-RELATIONAL DATABASE

Example of NoSQL



redis



mongoDB



APACHE
HBASE



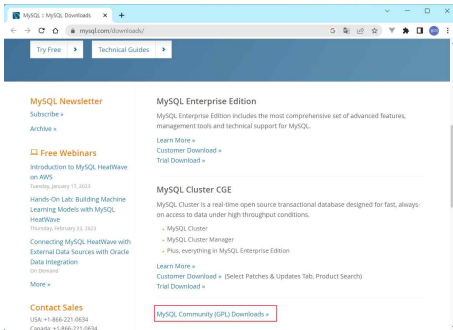
cassandra

```

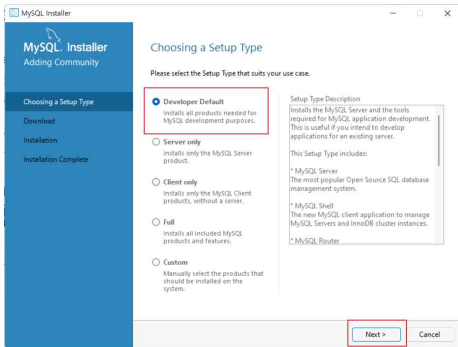
{
  "VehicleNo": "1317",
  "TripId": 7052622,
  "RouteNo": "C45",
  "Direction": "EAST",
  "Destination": "COTTONWOOD",
  "Pattern": "EB1",
  "Latitude": 49.196250,
  "Longitude": -122.556733,
  "RecordedTime": "11:16:28 am",
  "RouteMap": {
    "Href": "http://nb.translink.ca/geodata/C45.kmz"
  }
}
    
```

- 다운로드

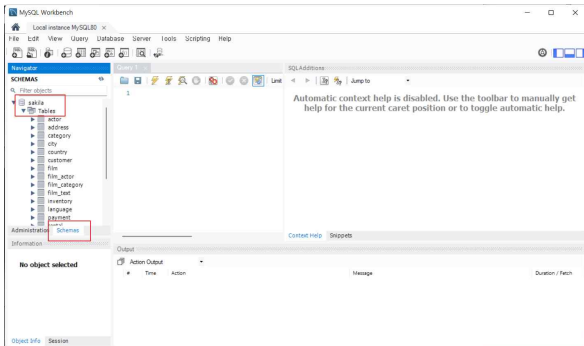
- <https://www.mysql.com/downloads/>



• DBMS 설치

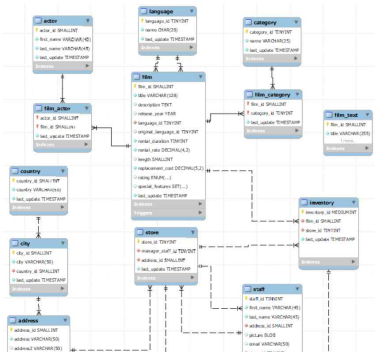


- Workbench 설치



- Sakila

- the Sakila sample database is designed to represent a DVD rental store.
- C:\Program Files (x86)\MySQL\Samples and Examples 8.0\Sample Databases\Sakila



- **테이블**

- 데이터가 저장되는 가상의 장소
- 테이블은 1개 이상의 칼럼으로 구성
- 이런 테이블의 구성을 스키마(schema)라고 함

- **칼럼**

- 칼럼은 타입을 가지며, 제약(값의 길이, 가질 수 있는 값 등)을 갖는다.

- **레코드**

- 칼럼의 모음을 레코드(record)라고 표현
- 하나의 테이블은 여러 개의 레코드로 구성

MEMBERID	PASSWORD	NAME	EMAIL
javaman	java	최범균	javaman@a.com
jspman	jsp	최모모	jspman@a.com

→ 레코드(Record) ←

- **주요키(Primary Key)**
 - 각각의 레코드를 구별하기 위해 사용되는 것
 - 각 레코드가 서로 다른 값을 갖는 칼럼
 - 주요키 값을 이용해서 빠른 검색 가능
- **인덱스**
 - 지정한 칼럼에 맞춰 데이터의 정렬 순서를 미리 계산
 - 주요키도 인덱스의 종류
 - 인덱스로 사용되는 칼럼은 중복된 값을 가질 수도 있음

•

SQL 타입	설명
CHAR	확정 길이의 문자열을 저장. 표준의 경우 255 글자까지만 저장가능
• VARCHAR	가변 길이의 문자열을 저장. 표준의 경우 255 글자까지만 저장가능
LONG VARCHAR	긴 가변 길이의 문자열을 저장
NUMERIC	숫자를 저장
DECIMAL	십진수를 저장
• INTEGER	정수를 저장
TIMESTAMP	날짜 및 시간을 저장
• TIME	시간을 저장
DATE	날짜를 저장
CLOB	대량의 문자열 데이터를 저장
BLOB	대량의 바이너리 데이터를 저장

- Structured Query Language

- 데이터 조회, 삭제 등의 데이터베이스 작업을 수행할 때 사용되는 언어
- SQL과 DATABASE는 다르나



- DDL(Data Description Language)

- 테이블 생성과 같이 데이터를 정의할 때 사용되는 SQL
- CREATE : 테이블을 생성 구조
- ALTER : 테이블을 수정
- DROP : 테이블 삭제
- TRUNCATE : 테이블을 초기화

- DML(Data Manipulation Language)

- 데이터 삽입, 조회, 삭제와 같이 데이터를 다루기 위해 사용되는 SQL
- ★ SELECT : 데이터 검색
- INSERT : 데이터를 입력
- UPDATE : 데이터를 수정
- DELETE : 데이터를 삭제

- create table 구문

```
create table TABLENAME (  
    COL_NAME1          COL_TYPE1(LEN1),  
    COL_NAME2          COL_TYPE2(LEN2),  
    ...,  
    COL_NAMEn          COL_TYPEn(LENn)  
)
```

- create table 예

```
create table MEMBER (  
    MEMBERID          VARCHAR(10) NOT NULL PRIMARY KEY,  
    PASSWORD          VARCHAR(10) NOT NULL,  
    NAME              VARCHAR(20) NOT NULL,  
    EMAIL             VARCHAR(80)  
)
```

- insert 구문

```
insert into [테이블이름] ([칼럼1], [칼럼2], .., [칼럼n])  
values ([값1], [값2], .., [값n])
```

- 새로운 레코드를 삽입
- 칼럼에 대해 값을 설정
- 칼럼 목록을 지정하지 않은 경우 values 에 모든 칼럼에 대한 값을 지정

- insert 예

```
insert into MEMBER (MEMBERID, PASSWORD, NAME)  
values ('madvirus', '1234', '최범균');
```

- select 구문

```
SELECT [칼럼1], ..., [칼럼n]  
FROM [테이블이름]  
WHERE <컬럼> = [value]  
ORDER BY [col]
```

```
SELECT MEMBERID, NAME  
FROM MEMBER  
WHERE addr = '대구'  
ORDER BY memberid
```

- select 절의 컬럼

- 테이블에서 조회하고 싶은 데이터
- 테이블 전체를 조회할 때는 select *

- where 절

- 조건에 맞는 레코드 검색

- `select * from MEMBER where NAME = '최범균'`

- and와 or로 다양한 조건 지정 가능

- `where NAME = '최범균' and EMAIL = 'madvirus@madvirus.net'`

- 주요 비교문

- `=, <>, >=, >, <=, <`

- `is null, is not null, like`

- **order by**를 이용한 조회 정렬 순서 지정
 - select .. from [테이블이름] where [조건절]
order by [칼럼1] asc, [칼럼2] desc, ...

Group by 는 세 그룹으로

- **집합 관련 함수**

- select max(SALARY), min(SALARY), sum(SALARY) from ...
 - max() - 최대값
 - min() - 최소값
 - sum() - 합

칼럼

- 수정 쿼리

- update [테이블이름] set [칼럼1]=[값1], [칼럼2]=[값2], .. where [조건절]
- where절을 사용하지 않을 경우 모든 레코드가 수정

- 삭제 쿼리

- delete from [테이블이름] where [조건절]
- where 절을 사용하지 않을 경우 모든 레코드가 삭제

- join
 - 두 개 이상의 테이블로부터 관련 있는 데이터를 읽어올 때 사용

- 기본 구문

```
select A. 칼럼1, A. 칼럼2, B. 칼럼3, B. 칼럼4
from [테이블1] as A, [테이블2] as B
where A.[ 칼럼x] = B.[ 칼럼y]
```

- 조인 사용에 따른 장단점
 - 다수의 테이블을 한번에 조회할 때 유용
 - 조인이 복잡해 질수록 조회 속도가 느려질 가능성 높음
 - 복잡한 인덱스 설계 등을 필요로 함