관계형 Database, NoSQL(빅데이터) 사용법

Version	V0.2
Updated Date	2024.11.28 MongoDB added
Developer	Don Lee

v0.1: 2024.11.27 1st ver. MariaDB

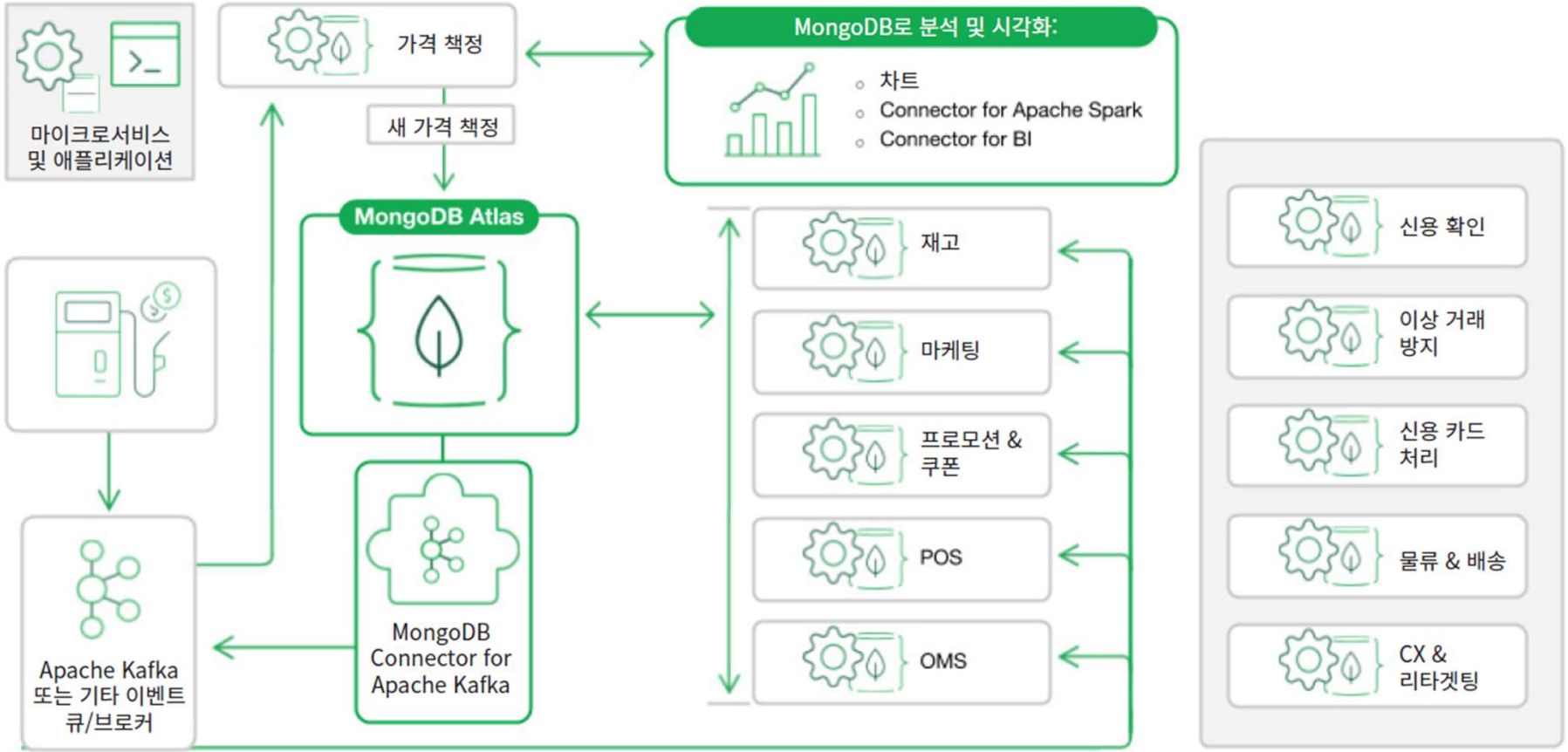
CONTENTS

Prolog: 빅데이터는 인공지능 리얼타임으로 간다
빅데이터 Scale-Out

빅데이터는 인공지능 리얼타임으로 간다

예) 전자 상거래 이벤트 기반 NoSQL Database

시계열과 IoT: 로봇도 실시간으로 수집되고 처리되어야 할 많은 데이터가 있다 !!!

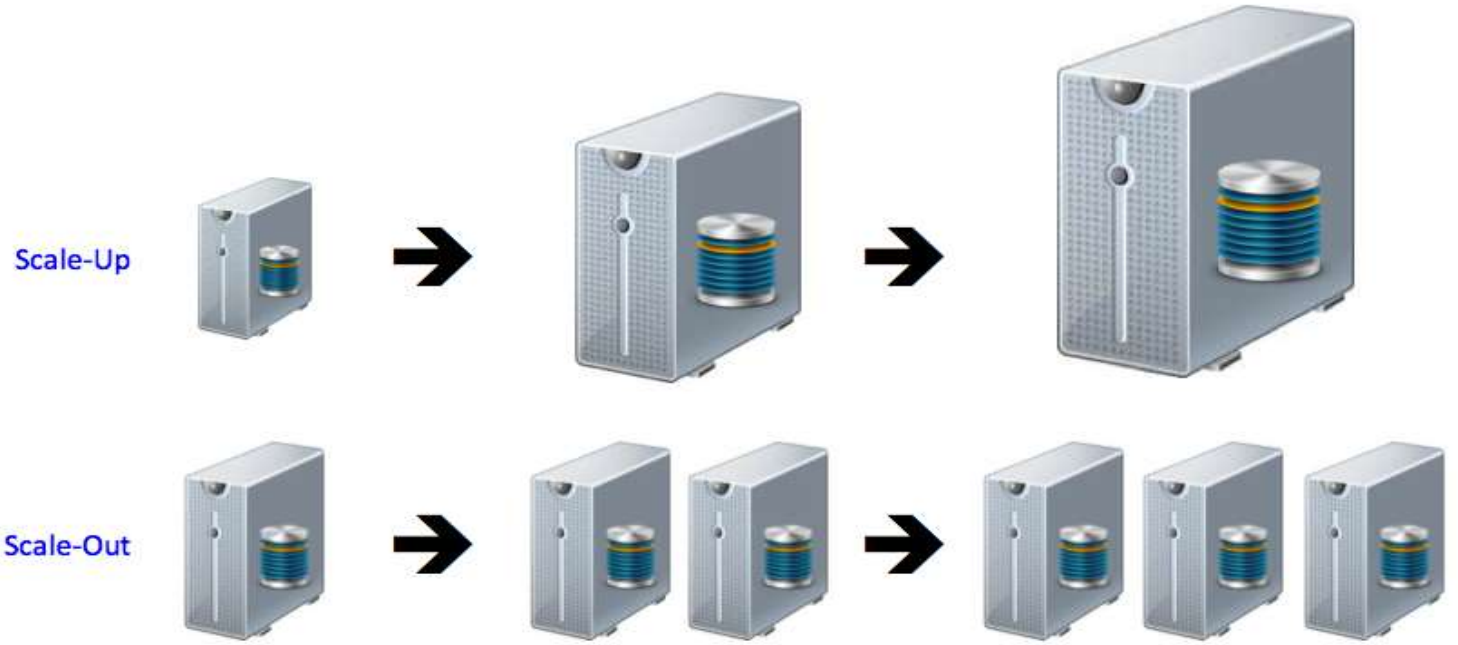


* Ref.: MongoDB 백서: MongoDB가 이벤트 기반 아키텍처로 실시간 데이터를 지원하는 방법, 2019년 10월

빅데이터 Scale-Out

빅데이터는 용량 증가에 유연하게 대응 가능

갑작스러운 데이터 증가에 준비되어 있는가?



Scale-Out이 대세

- Continuous Availability
- Continuous Redundancy
- Cost/Performance Flexibility
- Continuous Upgrade
- Geographical Distribution

Scale-Up도 상황에 따라 적용

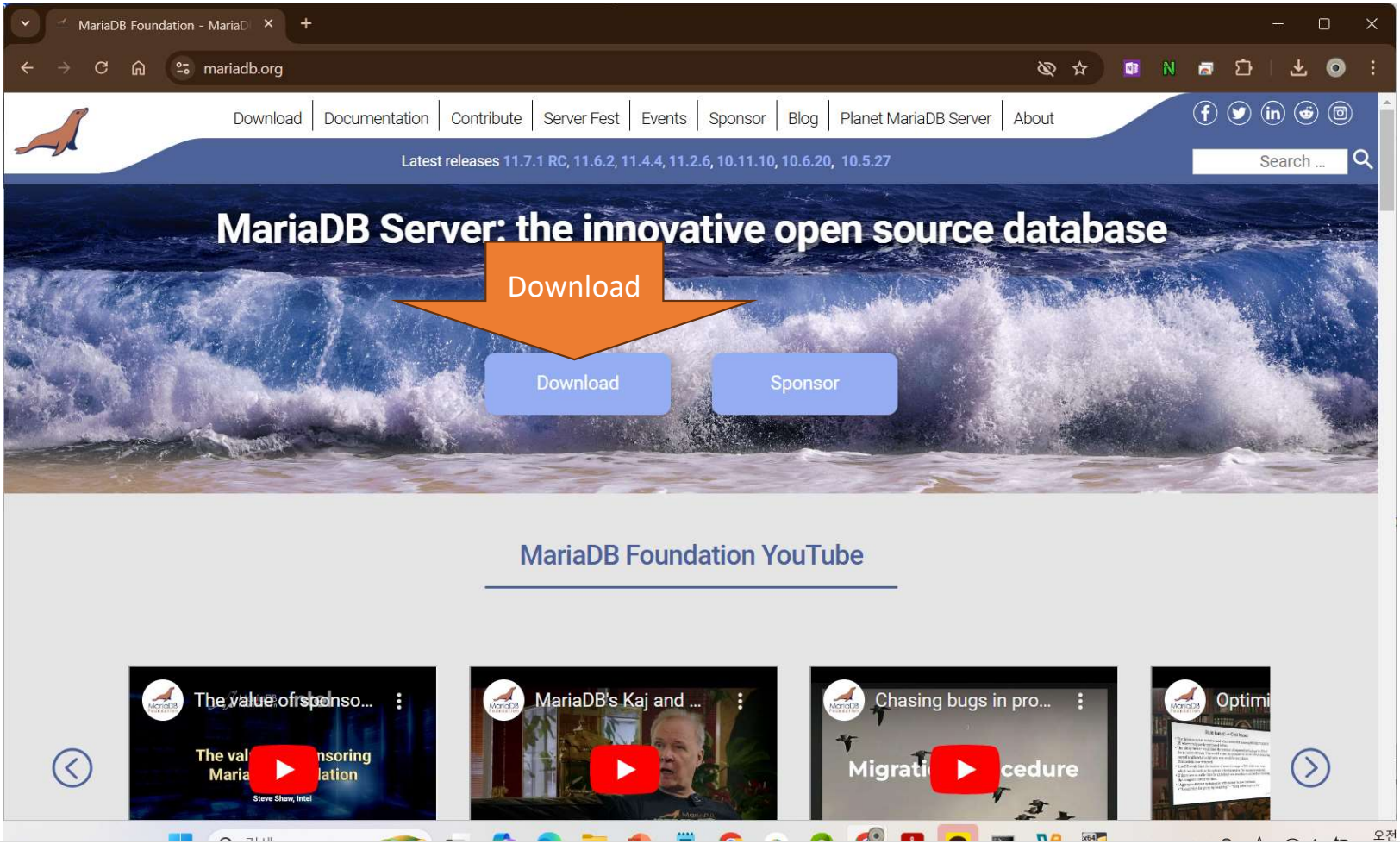
Chapter. 01

MariaDB 설치

- 무료로 사용 가능

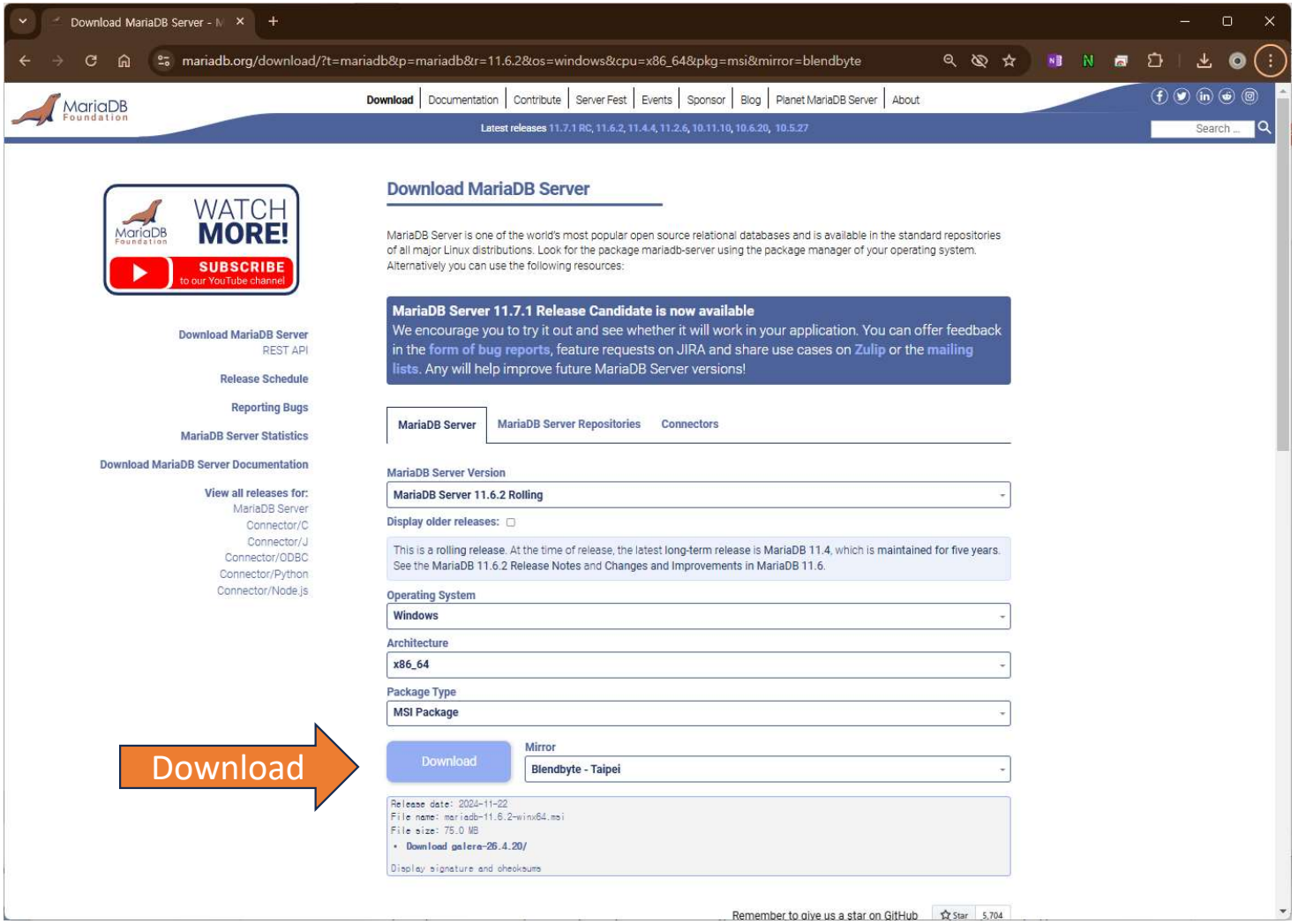
홈 페이지에서 Download

https://mariadb.org/



MariaDB 설치

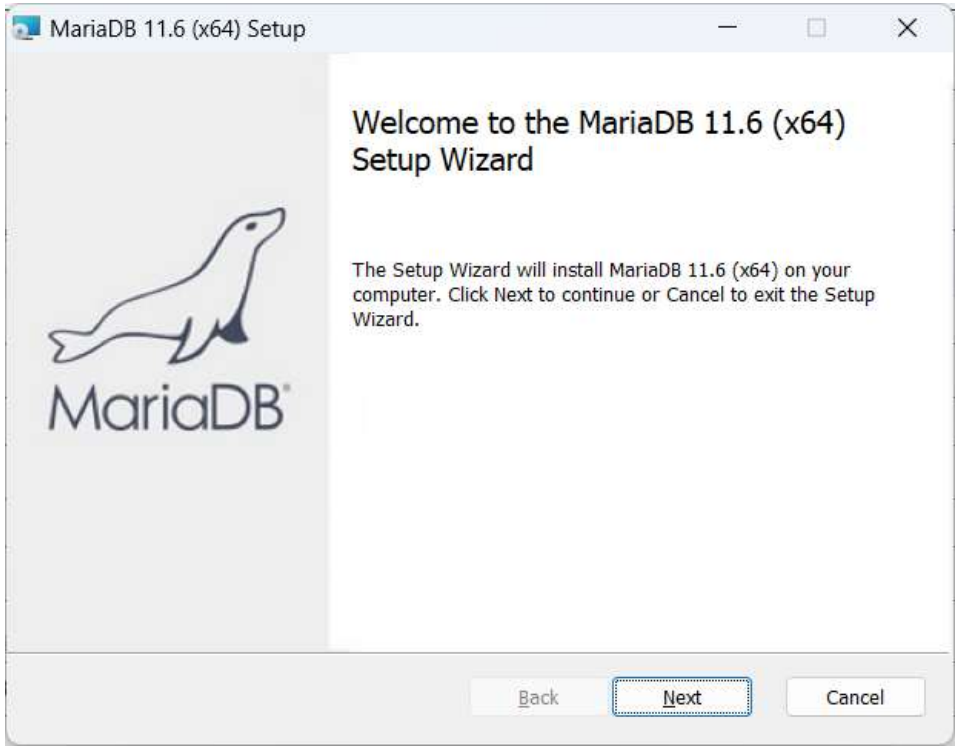
OS, 버전 선택

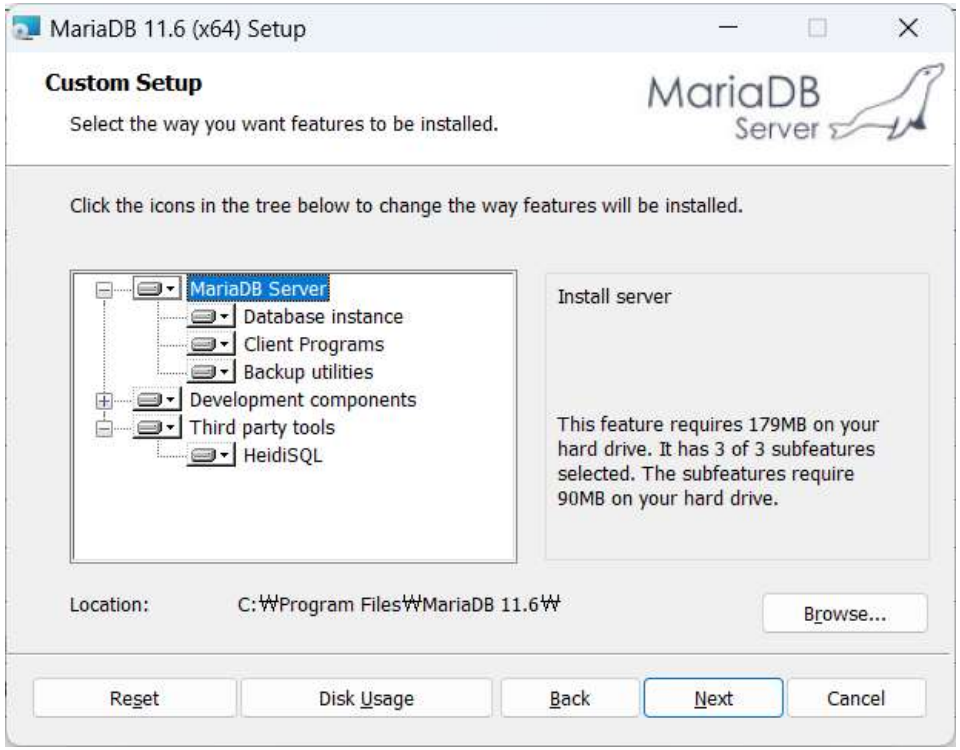


* 위키피디아, ARM architecture family, https://en.wikipedia.org/wiki/ARM_architecture_family

MariaDB 설치

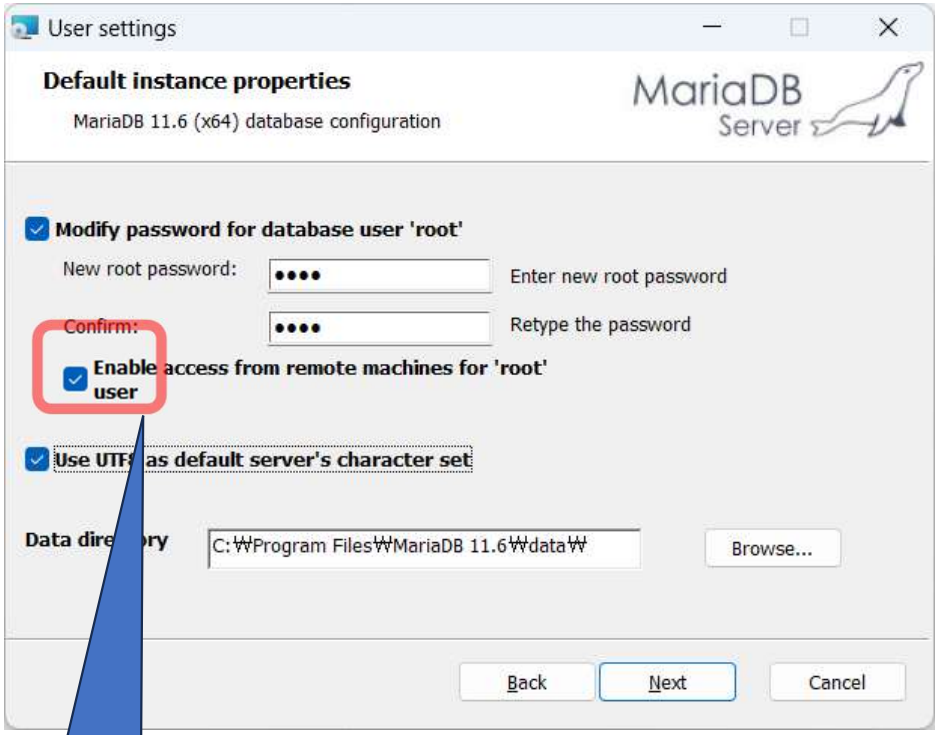
대부분 디폴트 선택



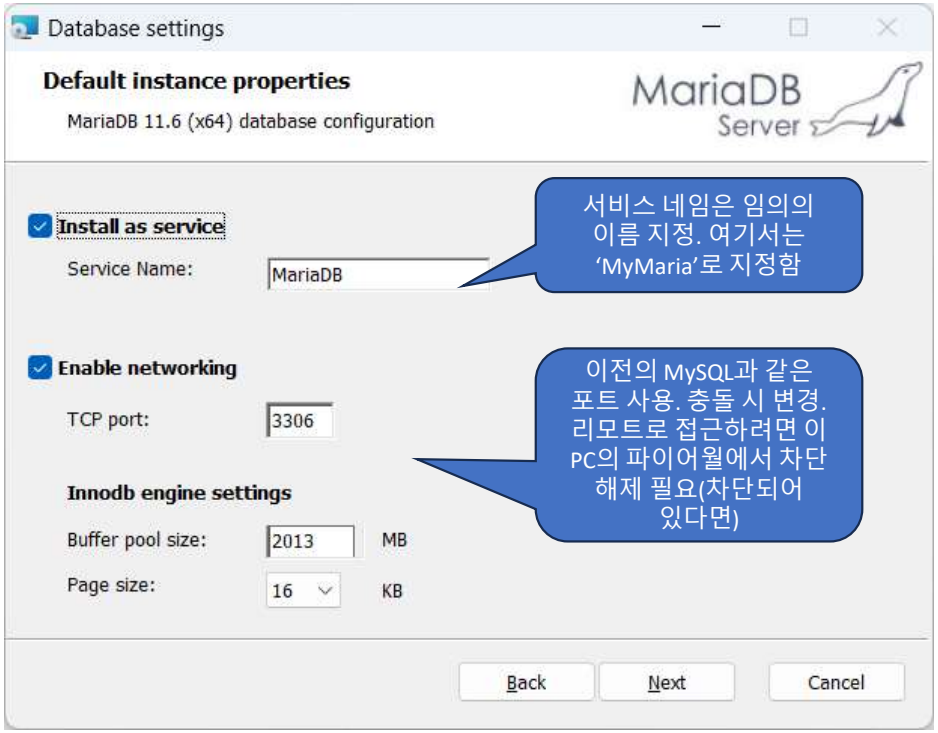


MariaDB 설치

리모트로 root user 사용 가능 체크(보안 상으로는 비추)



체크로 인스톨 진행. 루트 패스워드 보안에 주의

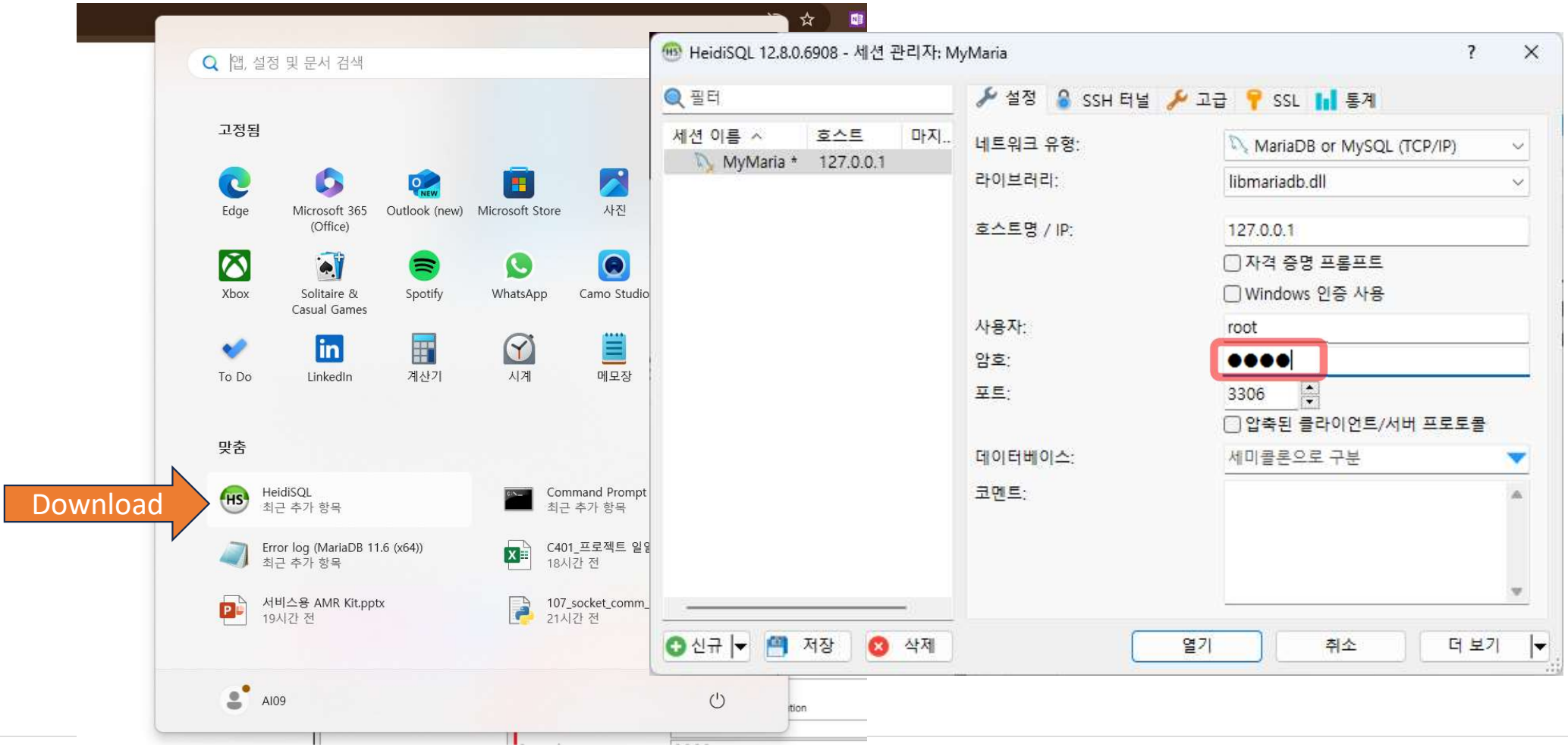


서비스 네임은 임의의 이름 지정. 여기서는 'MyMaria'로 지정함

이전의 MySQL과 같은 포트 사용. 충돌 시 변경. 리모트로 접근하려면 이 PC의 파이어월에서 차단 해제 필요(차단되어 있다면)

MariaDB 설치

포함된 HeidiSQL로 local/remote 데이터 관리



Download

Chapter. 02

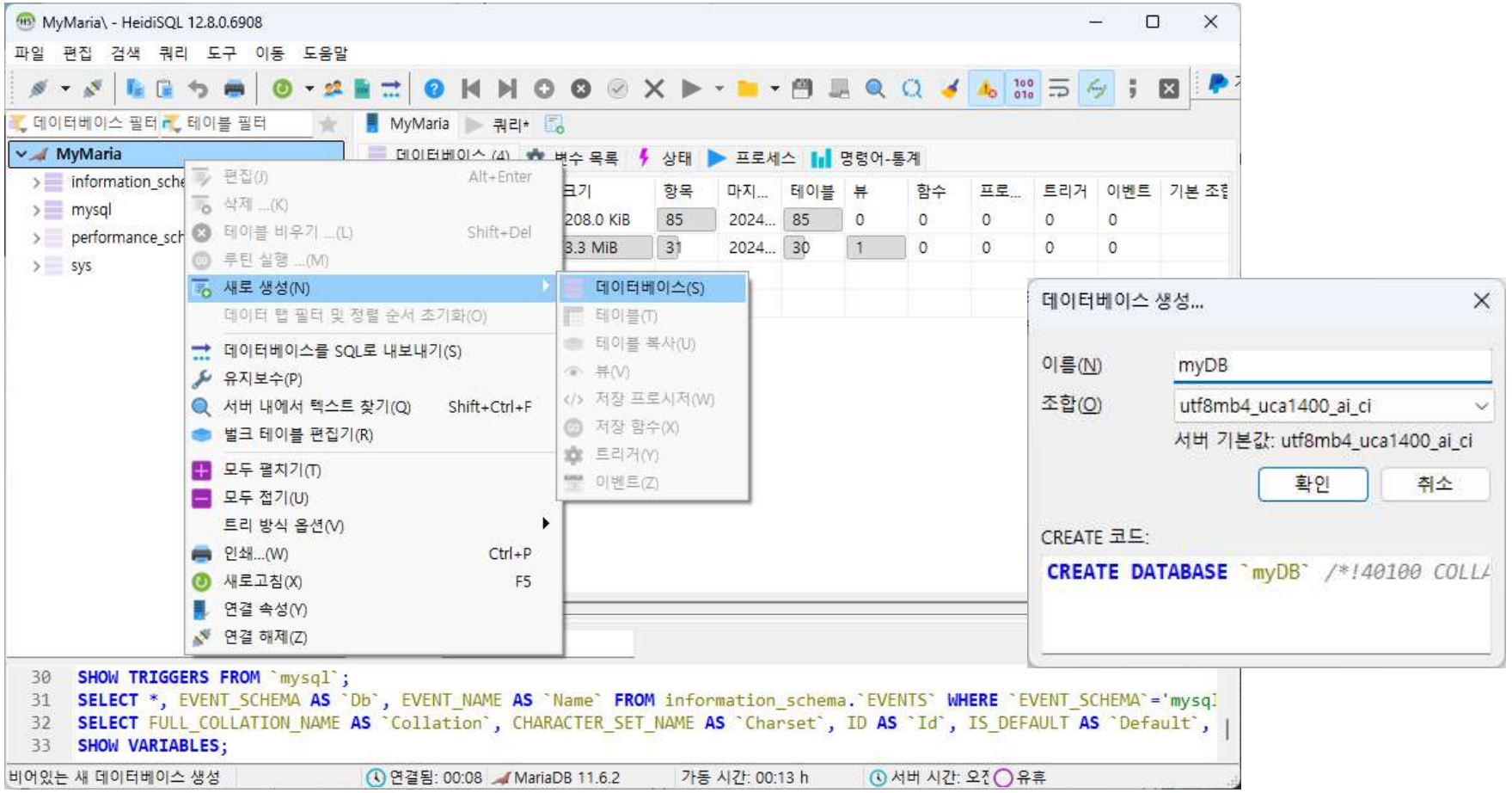
database 생성 (HeidiSQL)

참조: https://dololak.tistory.com/770

database 생성(HeidiSQL)

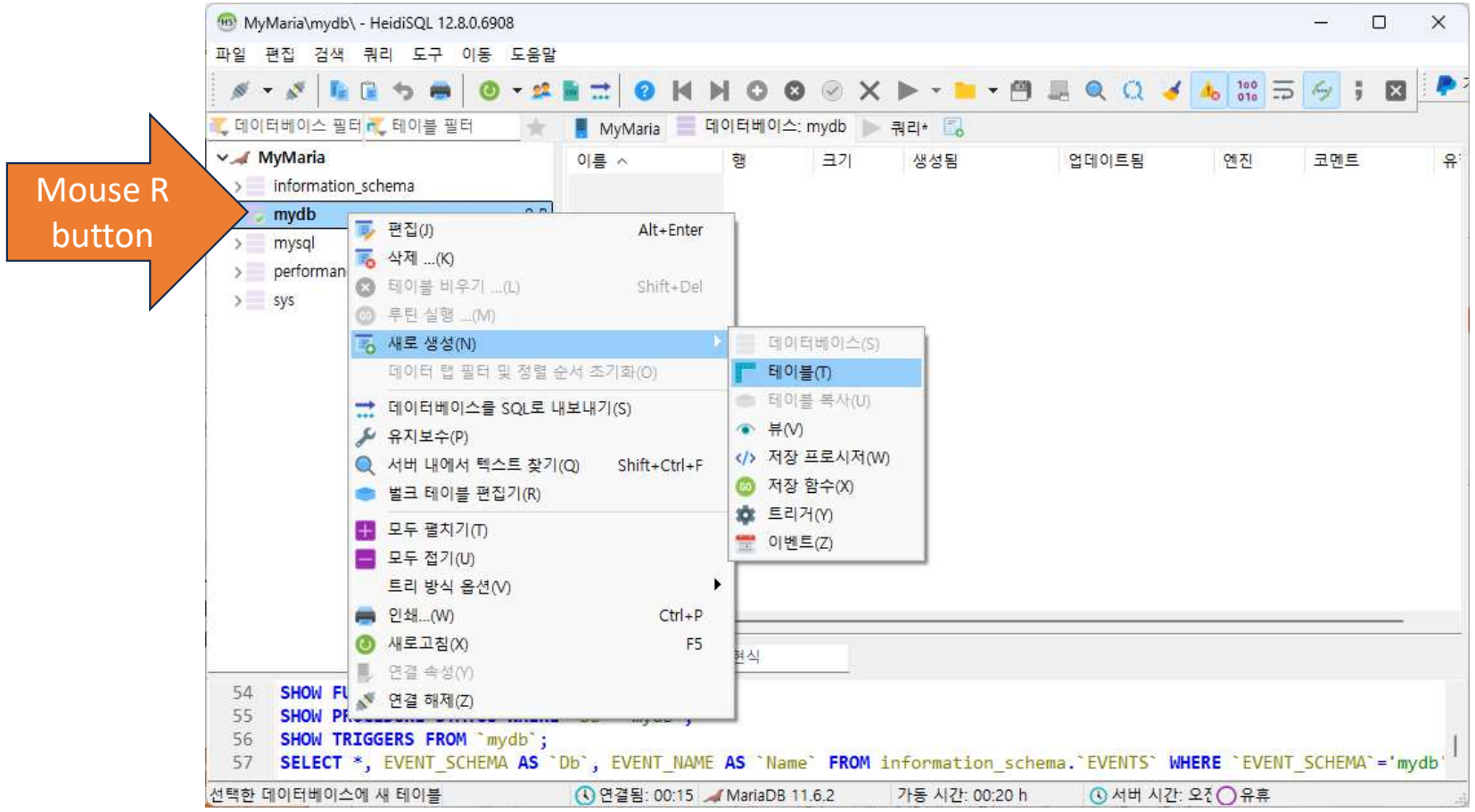
데이터베이스 생성

Mouse R button

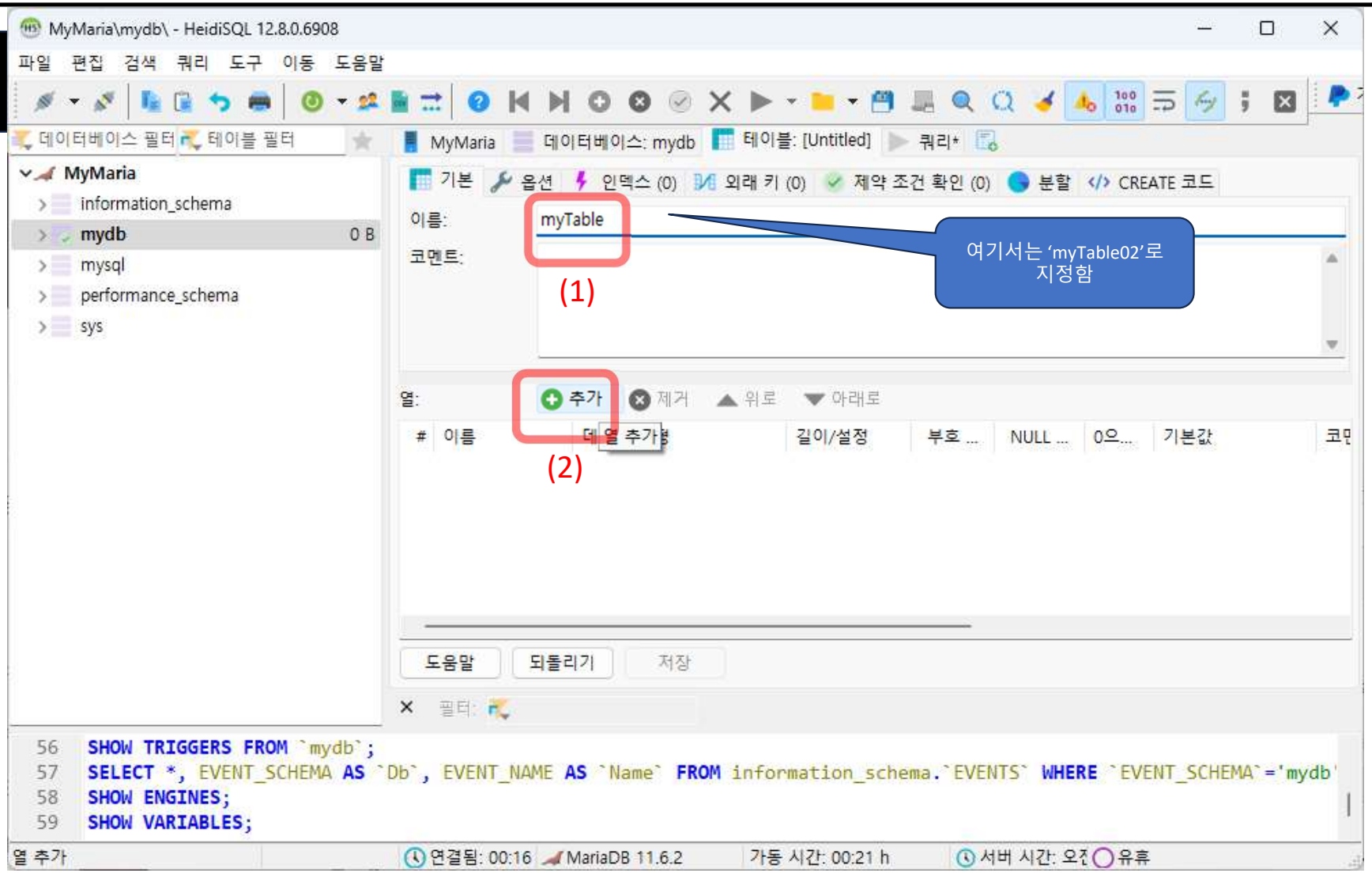


database 생성(HeidiSQL)

데이터베이스 내에서 필요한 여러 테이블 생성

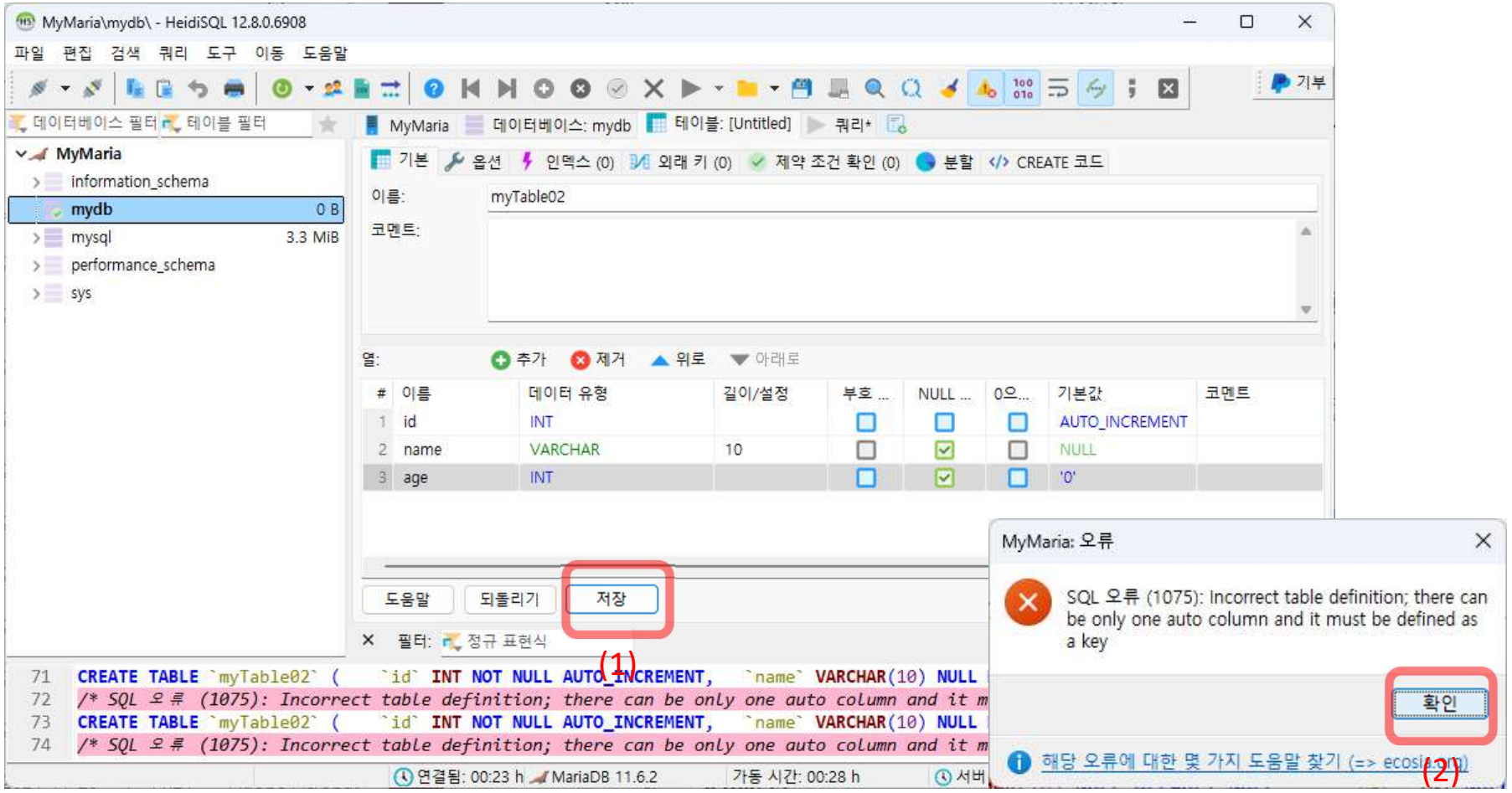


테이블 이름 지정과 필드 추가

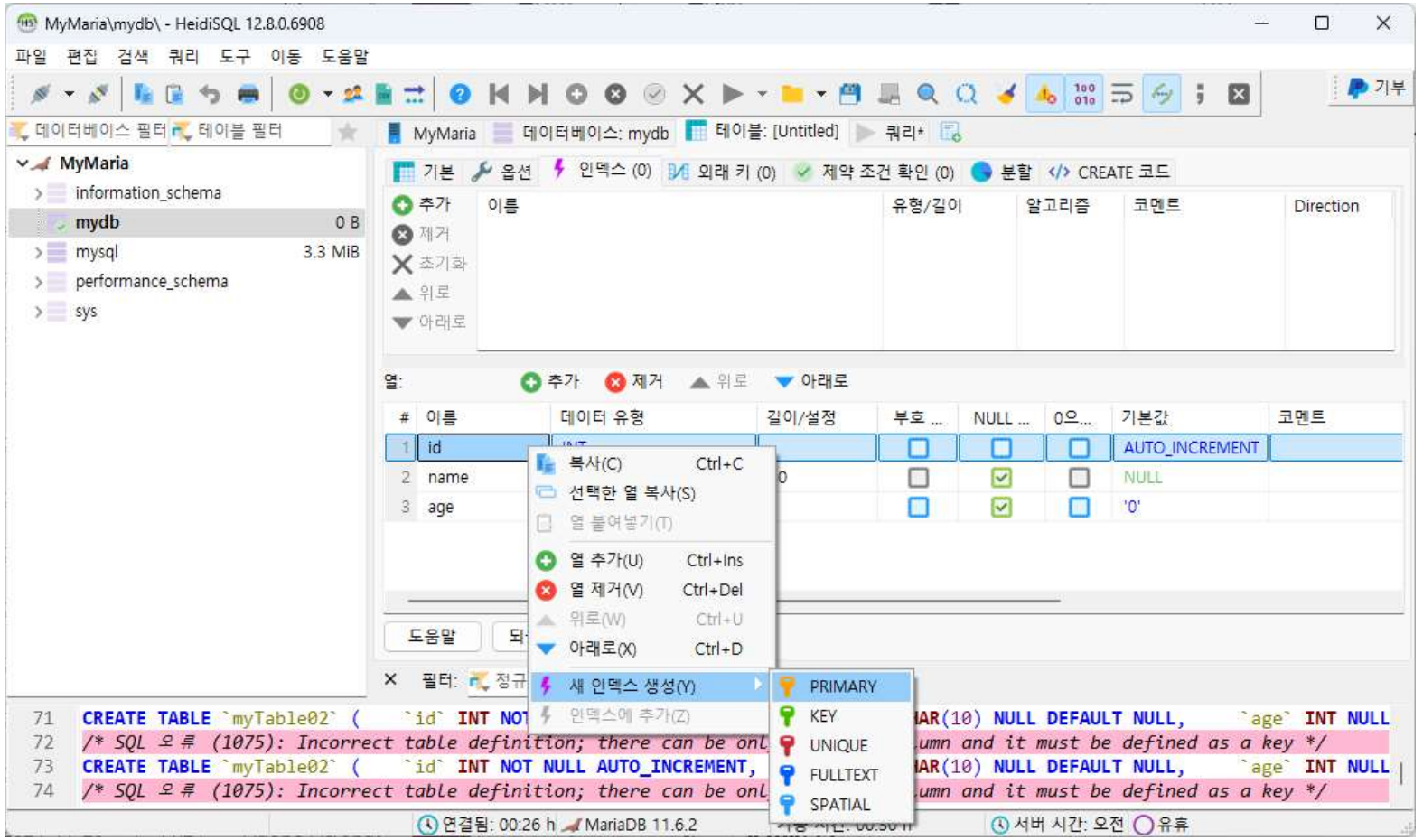


database 생성(HeidiSQL)

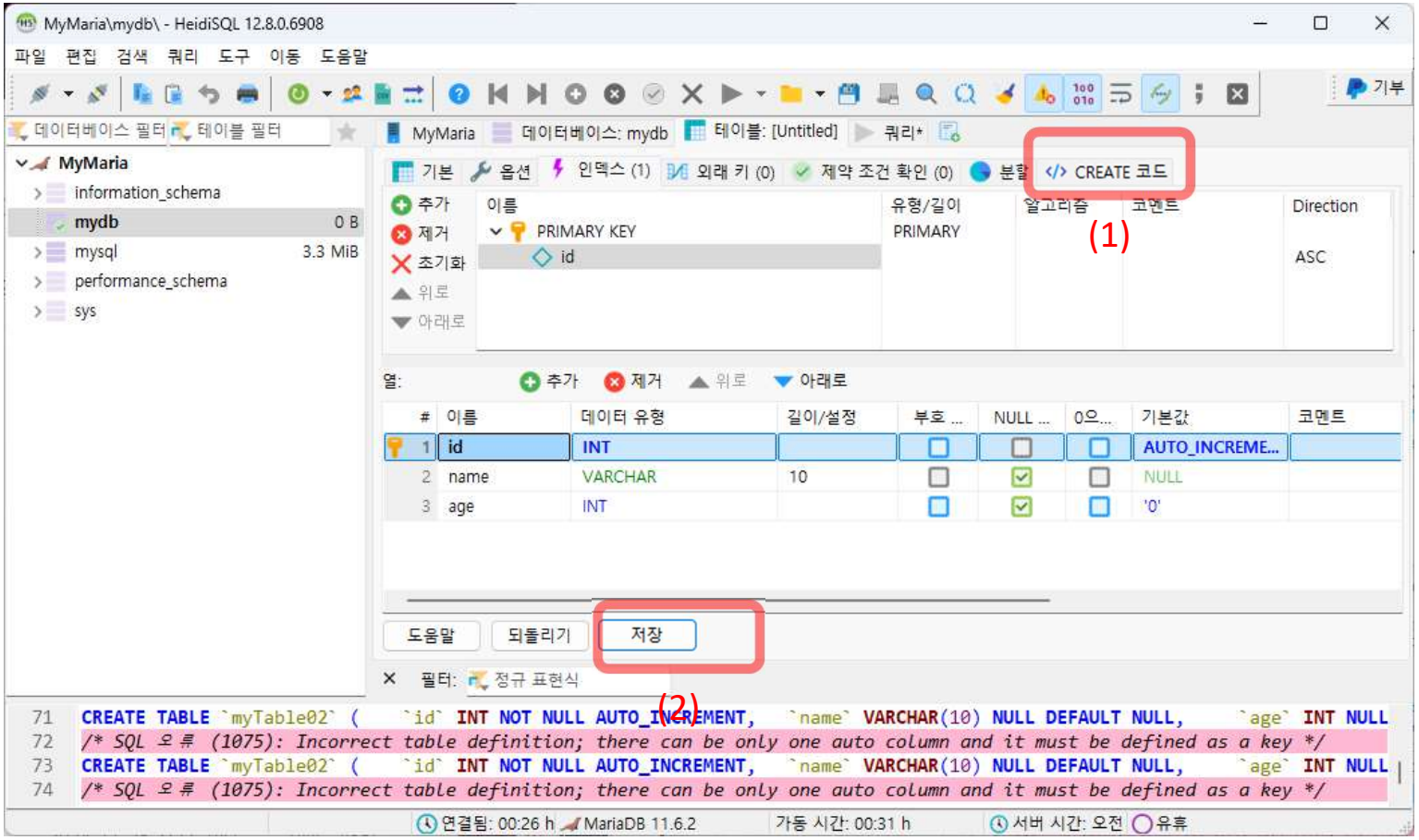
저장 -> 에러: 기본 키가 반드시 한 개 필요



id 필드를 기본(PRIMARY) 키로 지정

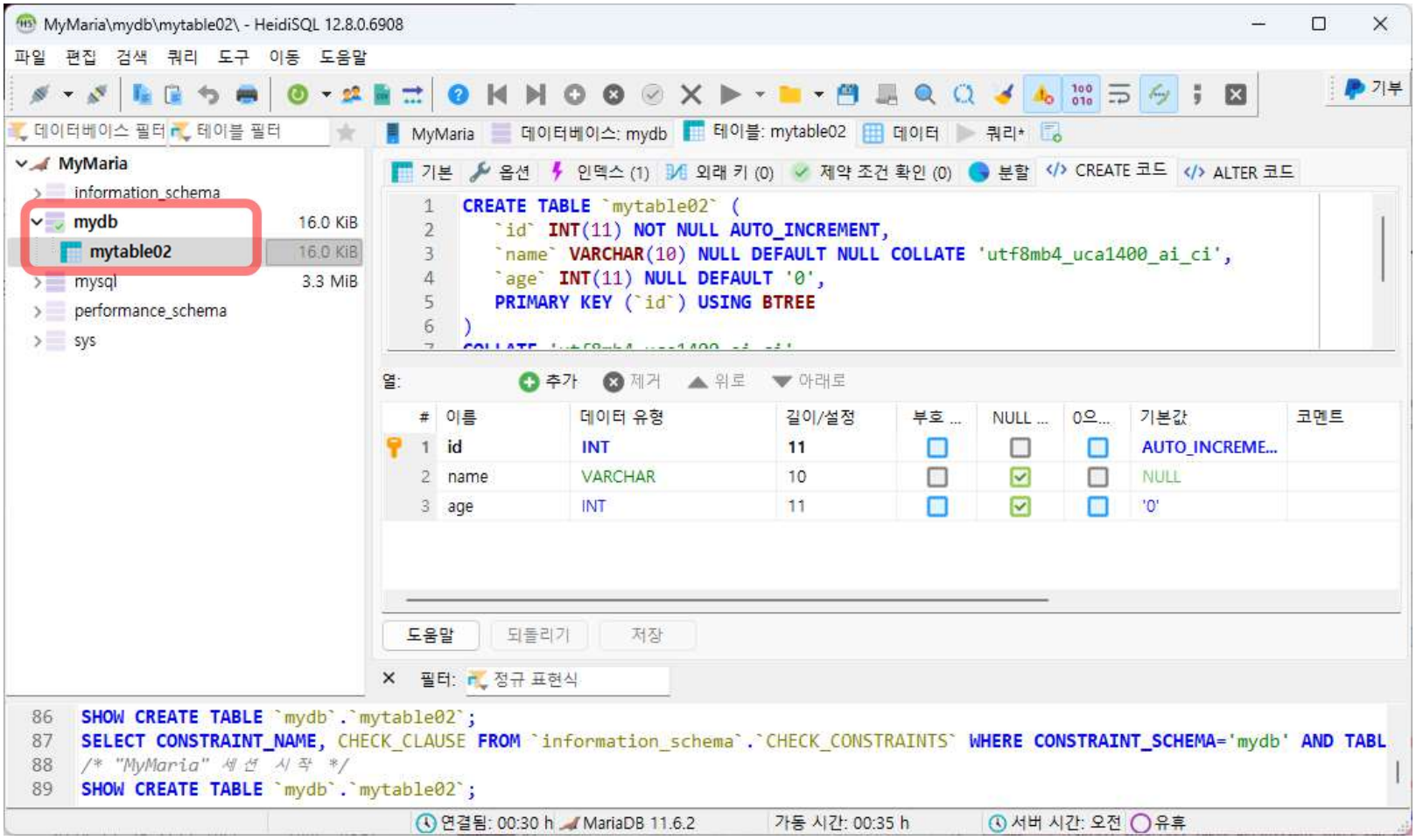


테이블 생성 후 저장


MyMaria\mydb\ - HeidiSQL 12.8.0.6908
파일 편집 검색 쿼리 도구 이동 도움말
MyMaria
information_schema
mydb 0 B
mysql 3.3 MiB
performance_schema
sys
CREATE 코드
(1)
추가
제거
초기화
위로
아래로
PRIMARY KEY
id
열:
이름 데이터 유형 길이/설정 부호 ... NULL ... 0으... 기본값 코멘트
1 id INT AUTO_INCREME...
2 name VARCHAR 10 NULL
3 age INT '0'
도움말
되돌리기
저장
(2)
71 CREATE TABLE `myTable02` (`id` INT NOT NULL AUTO_INCREMENT, `name` VARCHAR(10) NULL DEFAULT NULL, `age` INT NULL
72 /* SQL 오류 (1075): Incorrect table definition; there can be only one auto column and it must be defined as a key */
73 CREATE TABLE `myTable02` (`id` INT NOT NULL AUTO_INCREMENT, `name` VARCHAR(10) NULL DEFAULT NULL, `age` INT NULL
74 /* SQL 오류 (1075): Incorrect table definition; there can be only one auto column and it must be defined as a key */
연결됨: 00:26 h MariaDB 11.6.2
가동 시간: 00:31 h
서버 시간: 오전
유휴

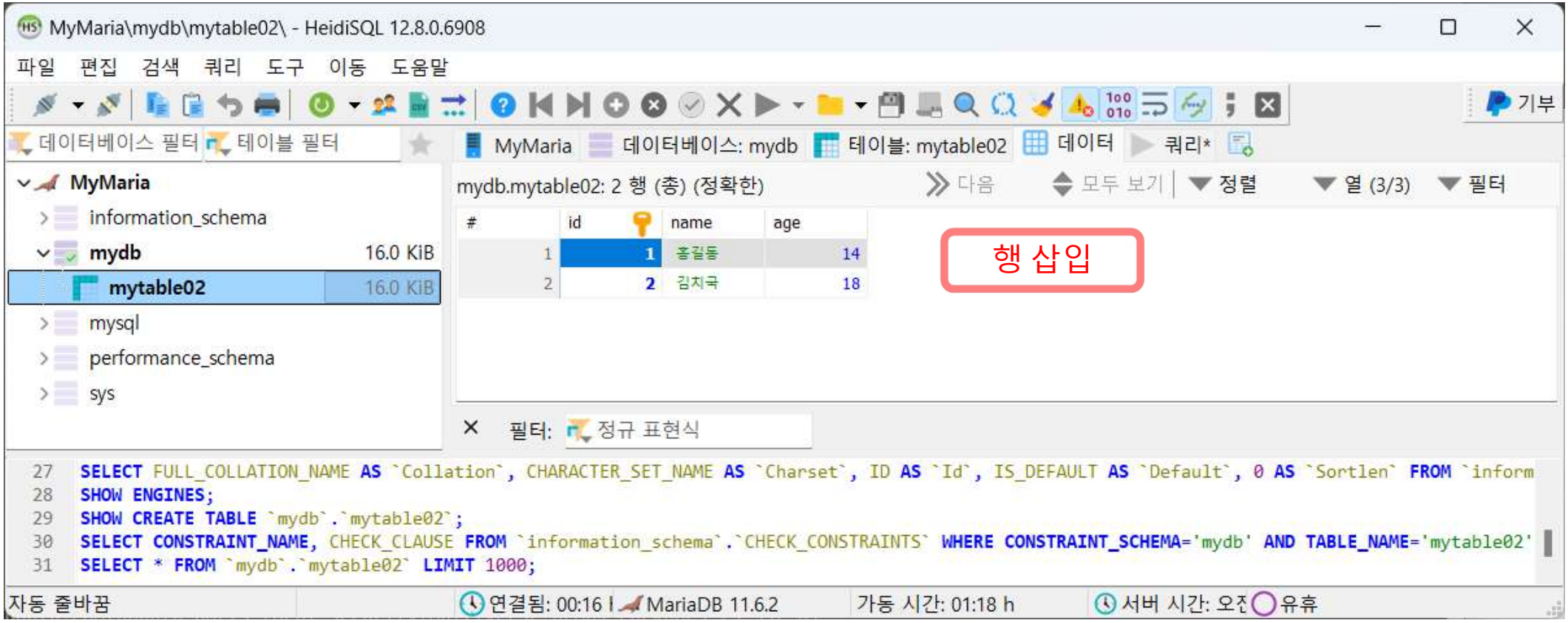
database 생성(HeidiSQL)

만들어진 테이블



database 생성(HeidiSQL)

필요한 행(레코드)를 삽입



Chapter. 03

Database (local/Remote) access

- 파이썬에서의 접근 방법

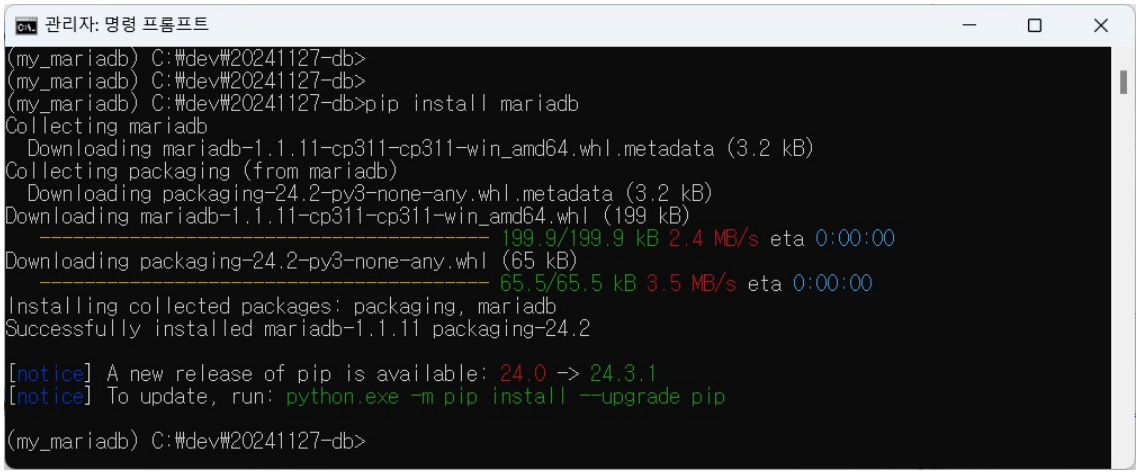
참조:
https://logdeveloper.github.io/python/python-mariadb-example/

Python 사용 시 mariadb 인스톨

pip install mariadb

안되면...

ChatGPT
등도 활용



sudo apt-get update

sudo apt-get install libmariadb-dev

sudo apt install mariadb-plugin-connect

sudo pip install mariadb --force-reinstall

...

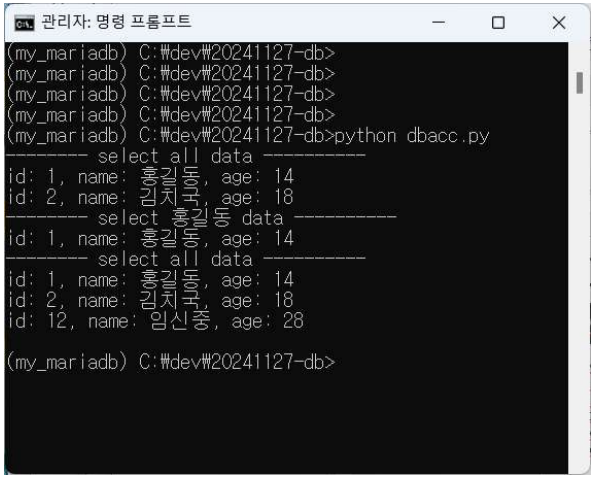
파이썬에서 동작 예

```
# Module Imports
import mariadb
import sys
# Connect to MariaDB Platform
try:
    conn = mariadb.connect(
        user="root",
        password="0000",
        host="localhost",
        port=3306,
        database="mydb"
    )
except mariadb.Error as e:
    print(f"Error connecting to MariaDB Platform: {e}")
    sys.exit(1)

conn.autocommit = True
#-----------------------------------------

# Get Cursor
cur = conn.cursor()
# selectall = "SELECT * from mytable02"
select_all_query = "SELECT id, name, age from mytable02"
cur.execute( select_all_query )

# query 결과를 list 형으로 가져옴.
resultset = cur.fetchall()

print('-------- select all data ----------')
for id, name, age in resultset:
    print(f"id: {id}, name: {name}, age: {age}")
```

NariaDB 서버 연결

이것이 True가 아니면 데이터 수정 반영이 안됨

모든 데이터 가져오기




```
#-----------------------------------------

some_name = "홍길동"
select_where_query = "SELECT id, name, age from mytable02 WHERE name=?"

cur.execute( select_where_query,(some_name,))
resultset = cur.fetchall()

print('-------- select 홍길동 data ----------')
for id, name, age in resultset:
    print(f"id: {id}, name: {name}, age: {age}")

#-----------------------------------------

another_name = "임신중"
another_age = 28

insert_query = "INSERT INTO mytable02 (name, age) VALUES (?, ?)"

try:
    cur.execute( insert_query, (another_name, another_age))
except mariadb.Error as e:
    print(f"Error: {e}")

#-----------------------------------------

cur.execute( select_all_query )

# query 결과를 list 형으로 가져옴.
resultset = cur.fetchall()

print('-------- select all data ----------')
for id, name, age in resultset:
    print(f"id: {id}, name: {name}, age: {age}")
```

특정 조건의 데이터 가져오기

데이터 추가

모든 데이터 가져오기

Chapter. 04

Bigdata NoSQL MongoDB 설치

- 무료로 사용 가능. 오픈소스

MongoDB 설치

MongoDB 란?

- **MongoDB ∈ NoSQL ??**
  - **MongoDB**는 **NoSQL**로 분류되며 이것이 MySQL, MariaDB와 구별되는 큰 특징
  - **NoSQL (= Not only SQL ≈ Non-Relational SQL** = 관계형 테이블과 다른 형식으로 데이터를 저장)
- **관계형 데이터베이스를 사용하지 않는다**는 뜻은 아니고,
  - **여러 유형의 데이터베이스를 사용**한다는 뜻
- NoSQL은 Instagram, Facebook, X(Twitter) 등 다양한 서비스에서 사용

MongoDB 특징

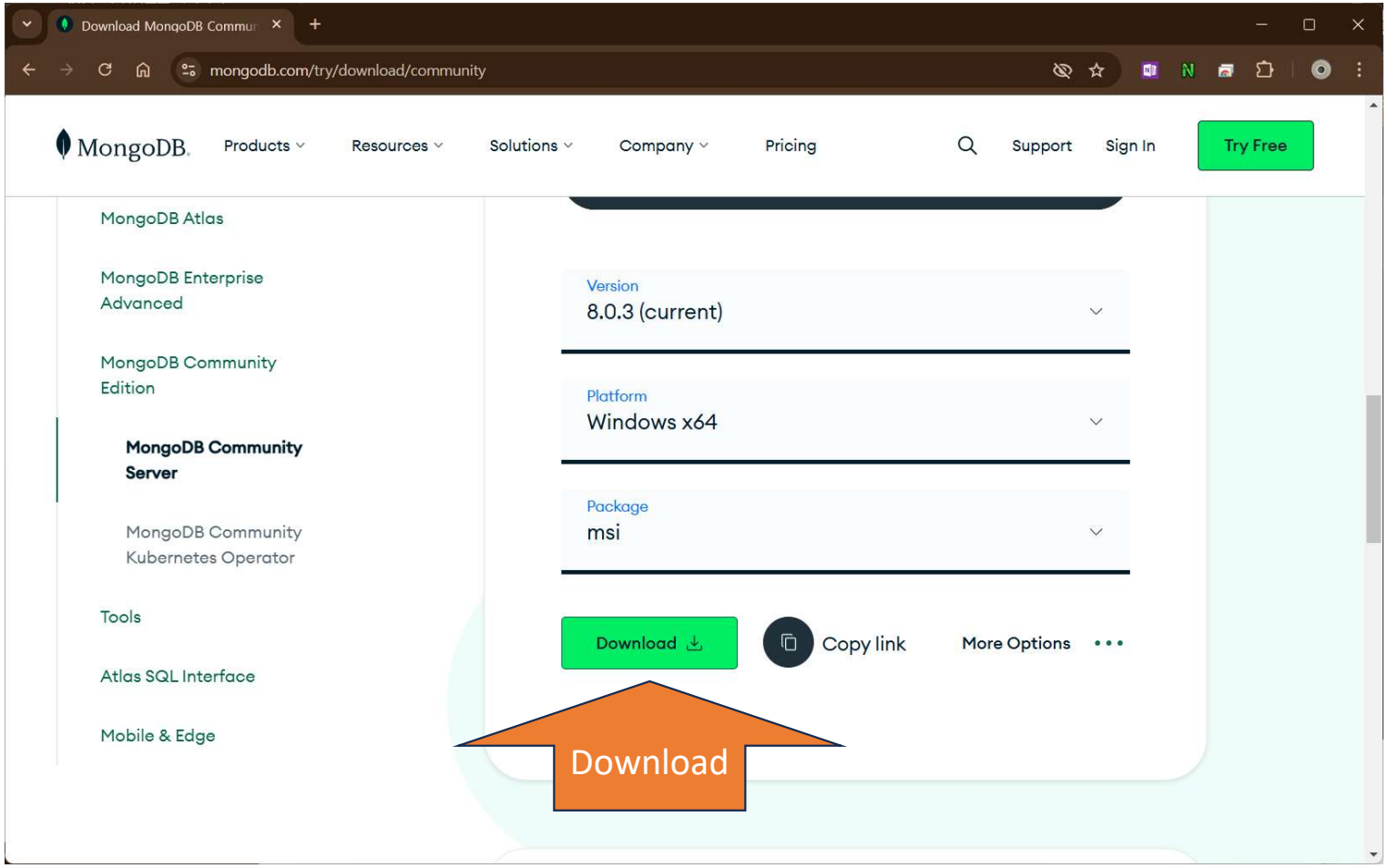
- **문서 지향 데이터베이스** : MongoDB는 JSON과 유사한 **BSON(Binary JSON)** 형식으로 문서 저장
- **유연한 스키마 (Schema-less)** : 스키마(Schema)가 고정이 아니어서 데이터 구조를 자유롭게 변경 가능
  * 스키마(Schema) : 데이터가 DB 내 어떻게 저장되는가를 나타내는 데이터베이스 구조
- **확장성** : MongoDB의 샤딩(Sharding)을 통해 데이터를 여러 서버에 분산 저장하여 성능과 용량을 확장하기 편리
  * 샤딩(Sharding) : 데이터베이스를 분산하여 저장하는 방법
- **Replica Set 구축** : Replica Set를 구축하여 데이터베이스의 데이터를 여러 서버에 동기화하여 **데이터의 안정성**, **서버 다운 방지**, **접근 가능한 데이터 상태 유지**
- **다양한 쿼리와 인덱스 지원** : MongoDB는 많은 쿼리와 인덱스를 지원하여 복잡한 쿼리를 효율적으로 처리 가능

MongoDB vs. MySQL

DBMS	MongoDB	MySQL
데이터 모델	문서형 (NoSQL)	관계형 (RDMBS)
데이터 저장 형식	BSON(Binary JSON) 형식	테이블(Table) 형식
데이터 관계	비정형 데이터 및 복잡한 관계에서 유리	정형화된 데이터 관계
트랜잭션	제한적인 트랜잭션 지원	완전한 트랜잭션 지원
사용 사례	실시간 분석, 빅데이터	금융, ERP 시스템
상대적 특징	빠른 개발 주기, 대규모 데이터 처리	데이터 간 안정성, 데이터 무결성

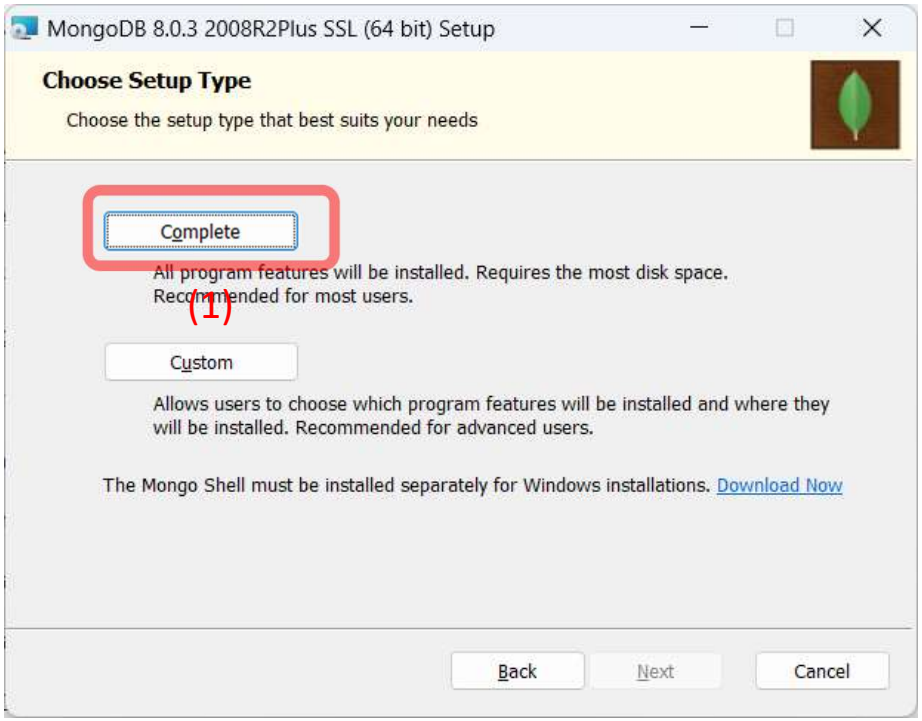
홈 페이지에서 Download

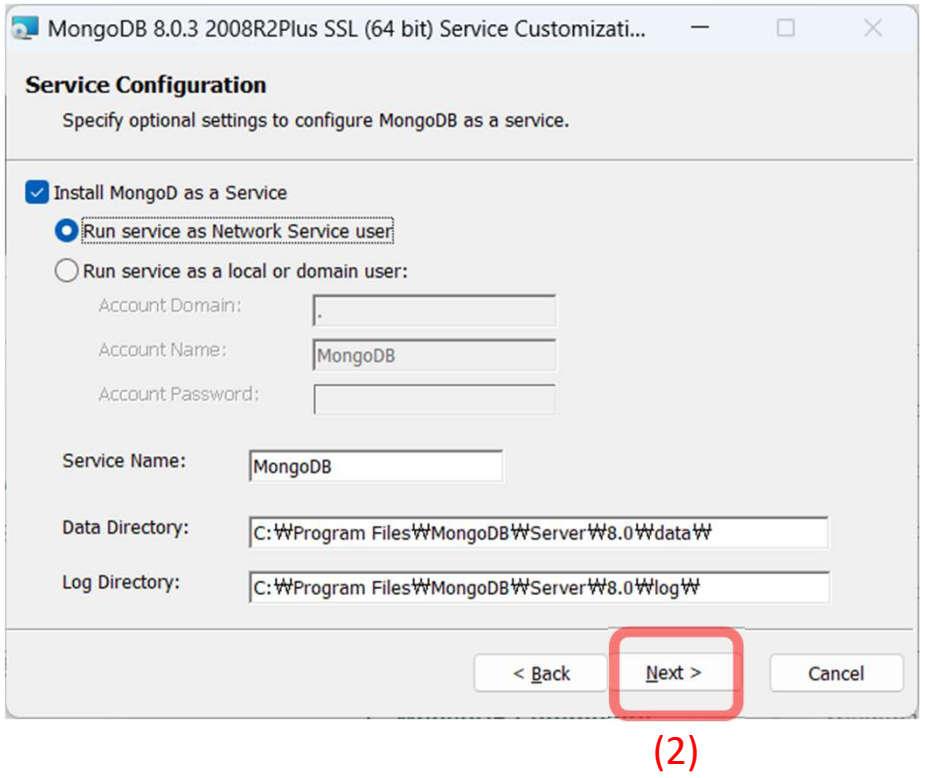
https://www.mongodb.com/try/download/community



MongoDB 설치

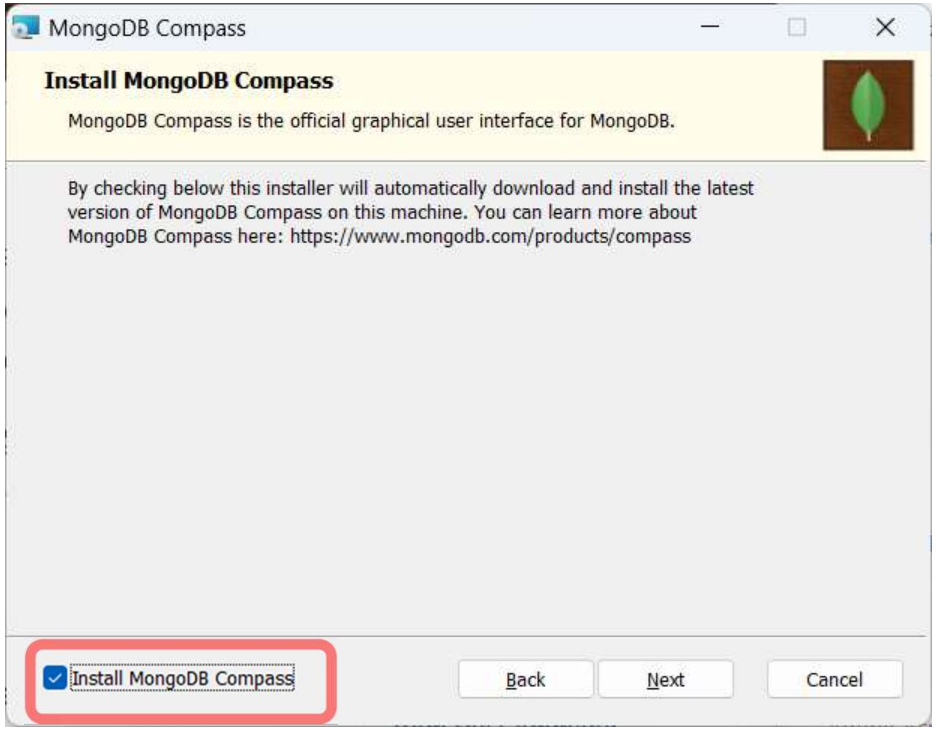
대부분 디폴트 선택

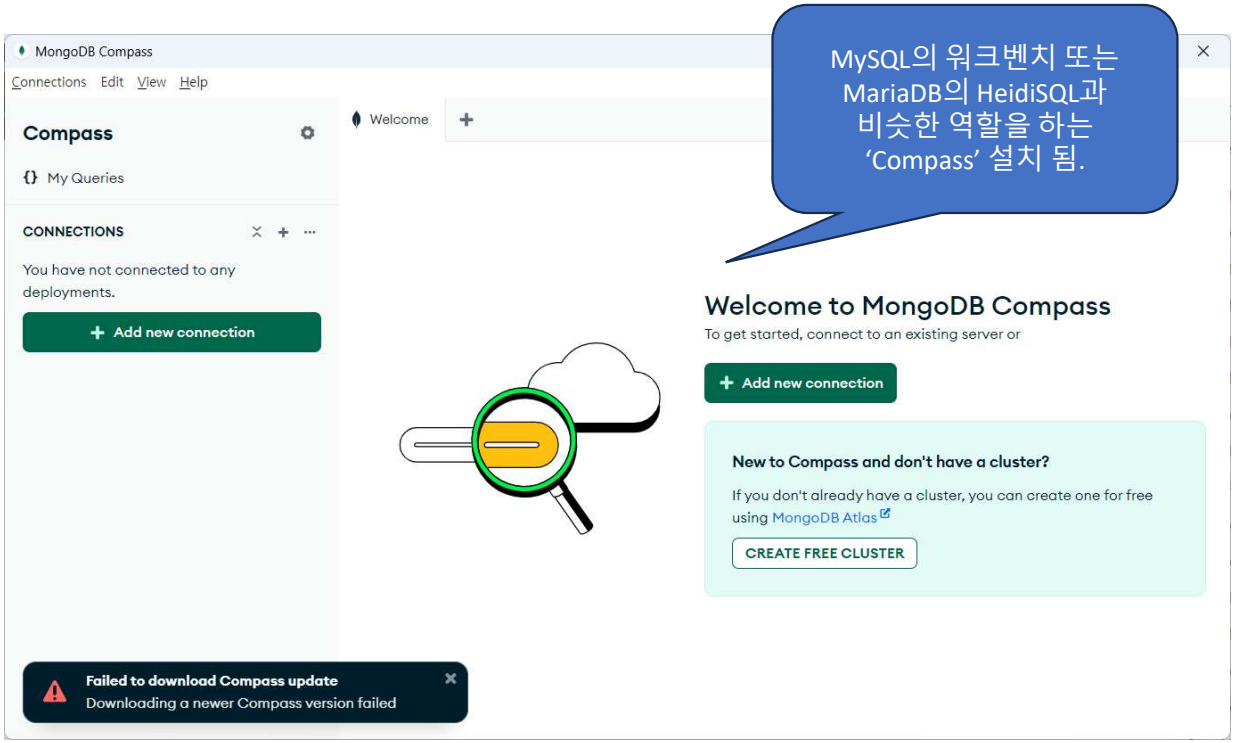




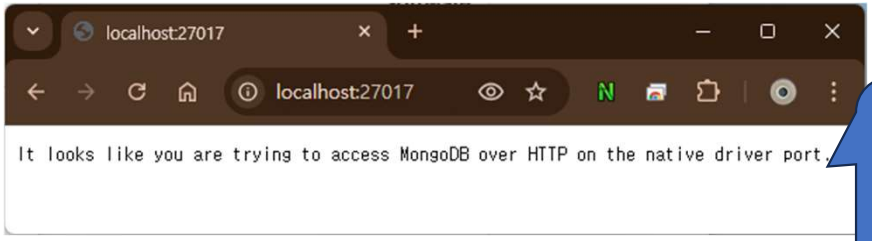
MongoDB 설치

디폴트 선택으로 인스톨 진행 시 'Compass' 자동 설치





MySQL의 워크벤치 또는 MariaDB의 HeidiSQL과 비슷한 역할을 하는 'Compass' 설치 됨.



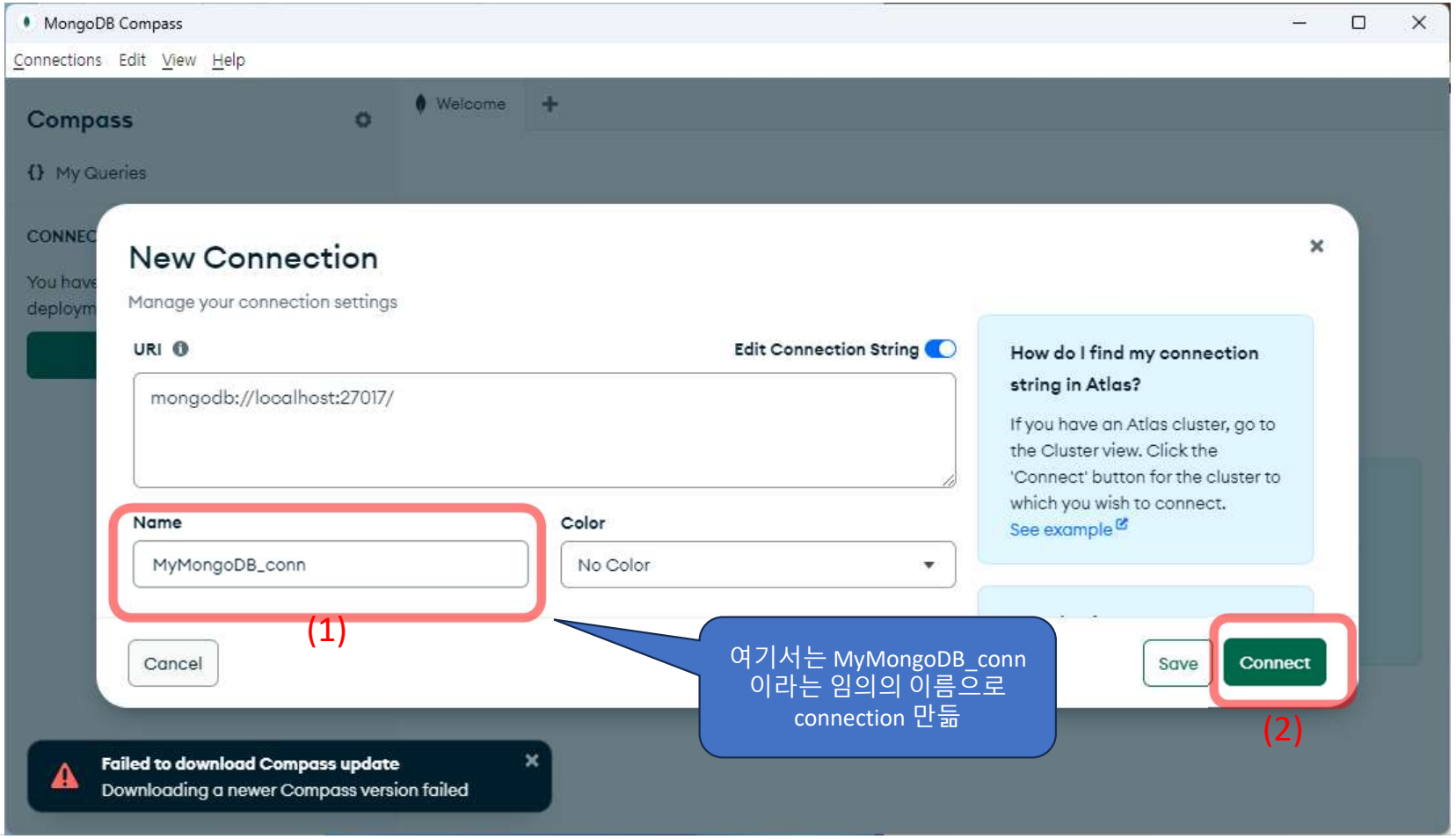
http://localhost:27017/ 라고 브라우저에서 입력 했을 때 이런 메시지가 나오면 MongoDB가 제대로 실행하고 있는 것임.

Chapter. 05

NoSQL 생성 (MongoDB)

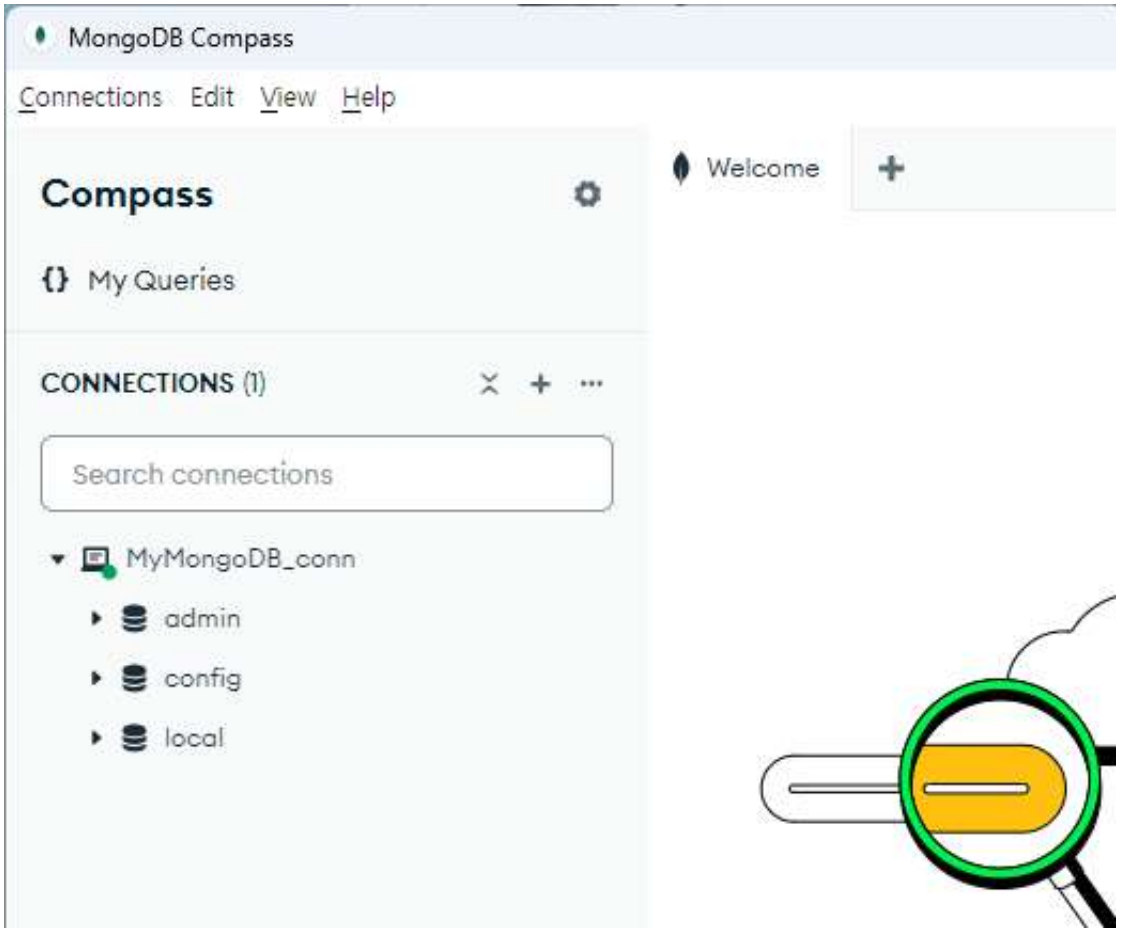
MongoDB 생성

새로운 connection 만들기

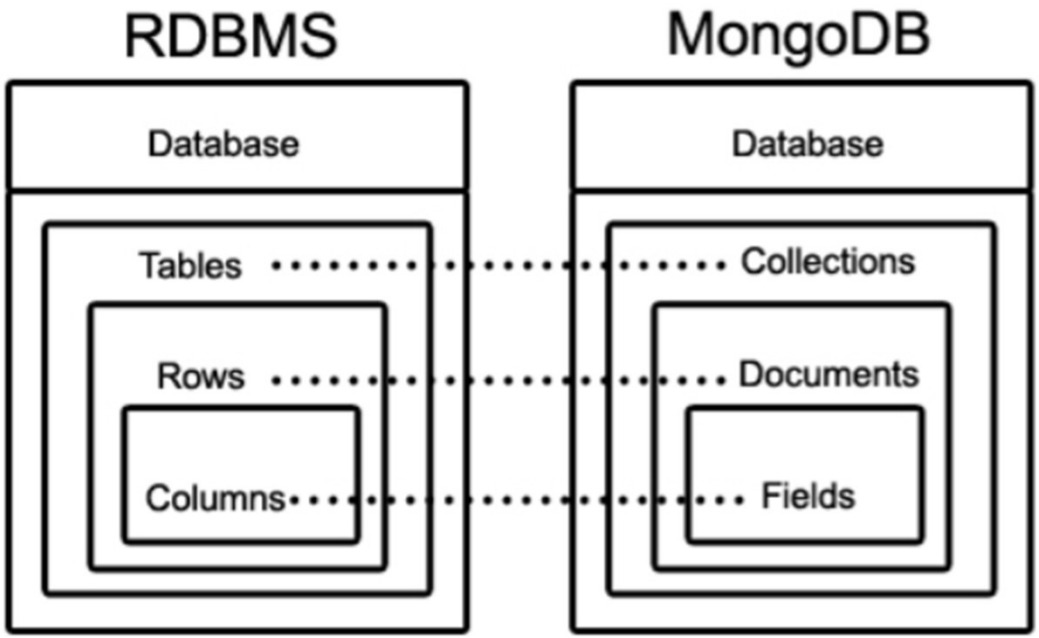


MongoDB 생성

디폴트 선택으로 인스톨 진행 시 'Compass' 자동 설치



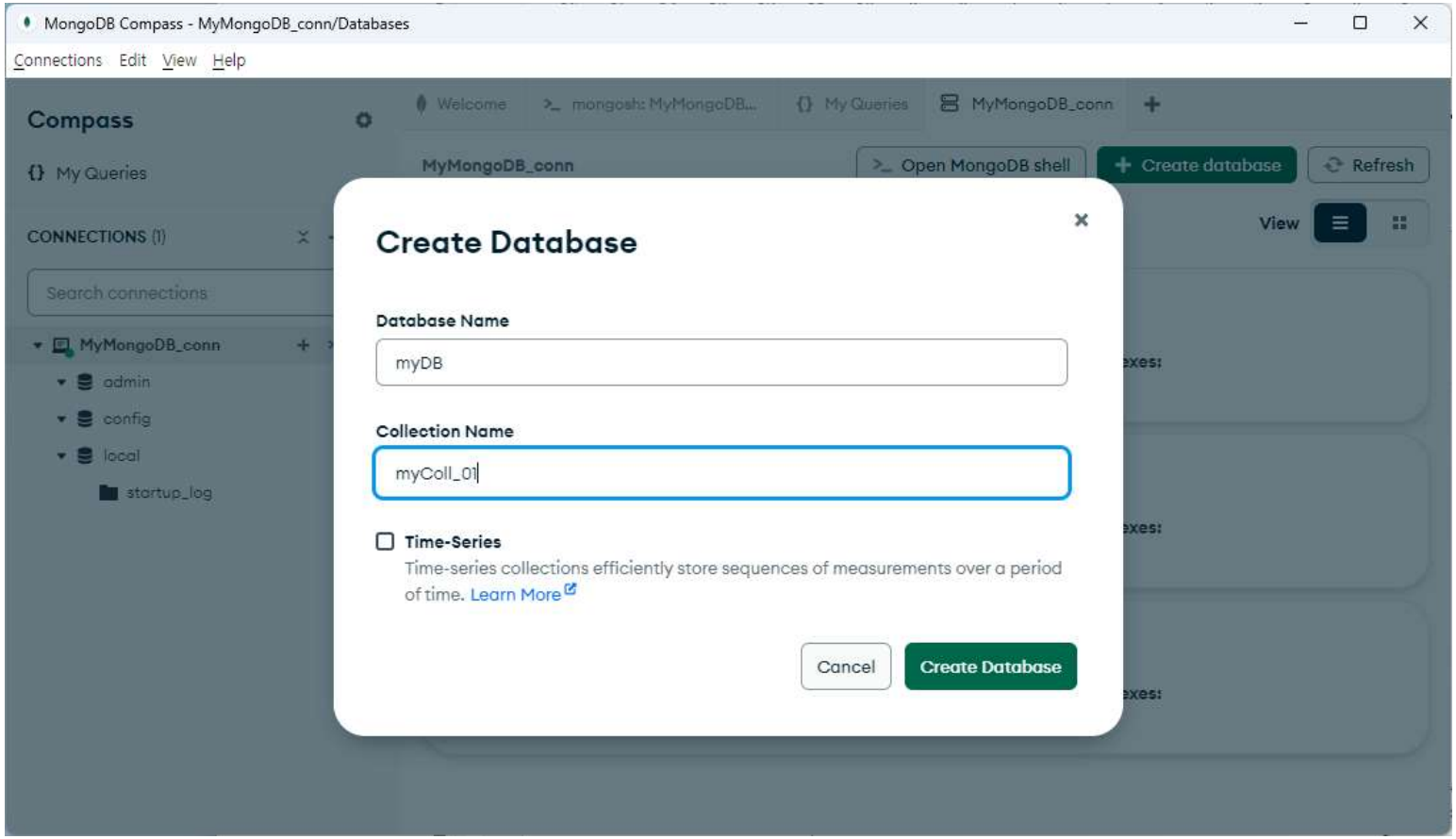
MongoDB 용어 – 관계형 DB와 비교



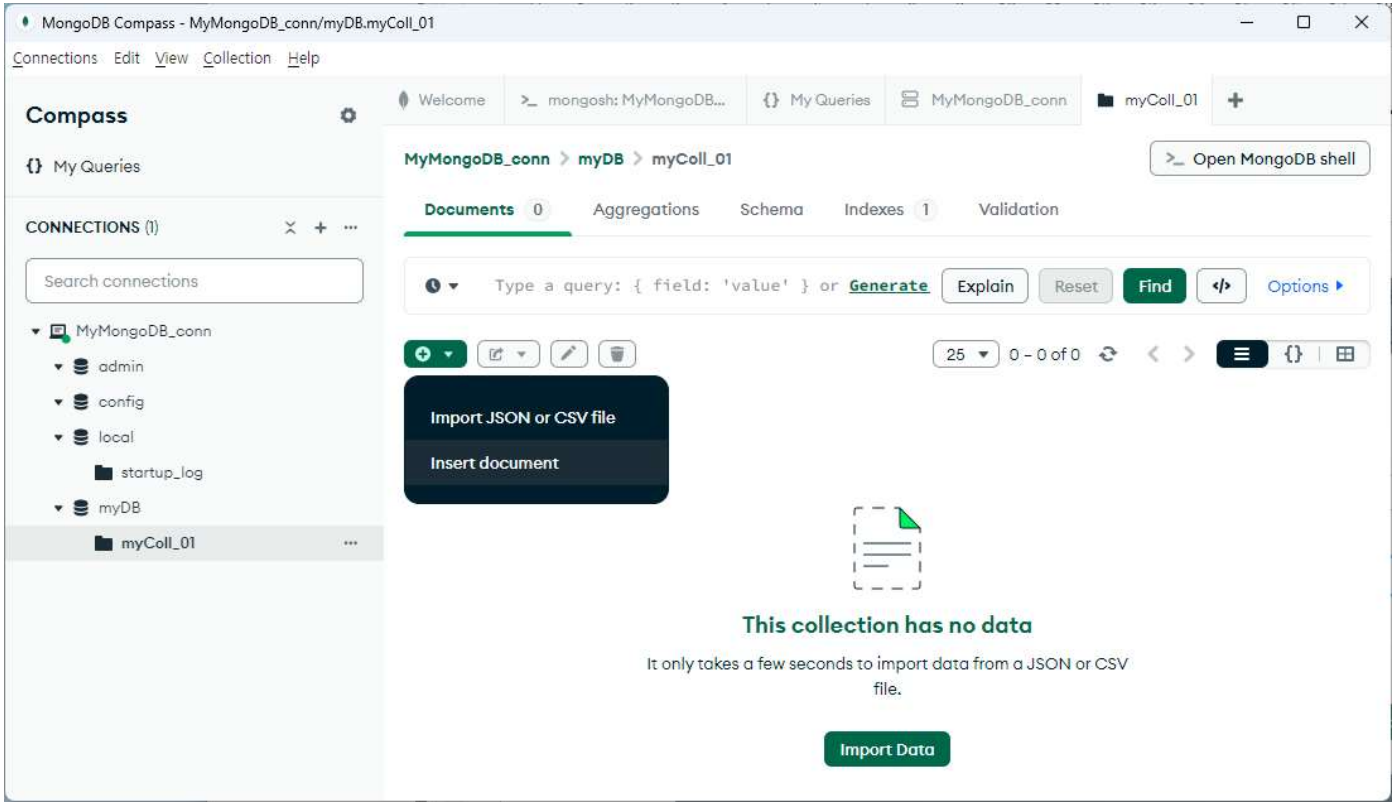
* ref.: MongoDB의 database와 Collection 사용하기, https://www.bearpooh.com/166#google_vignette

MongoDB 생성

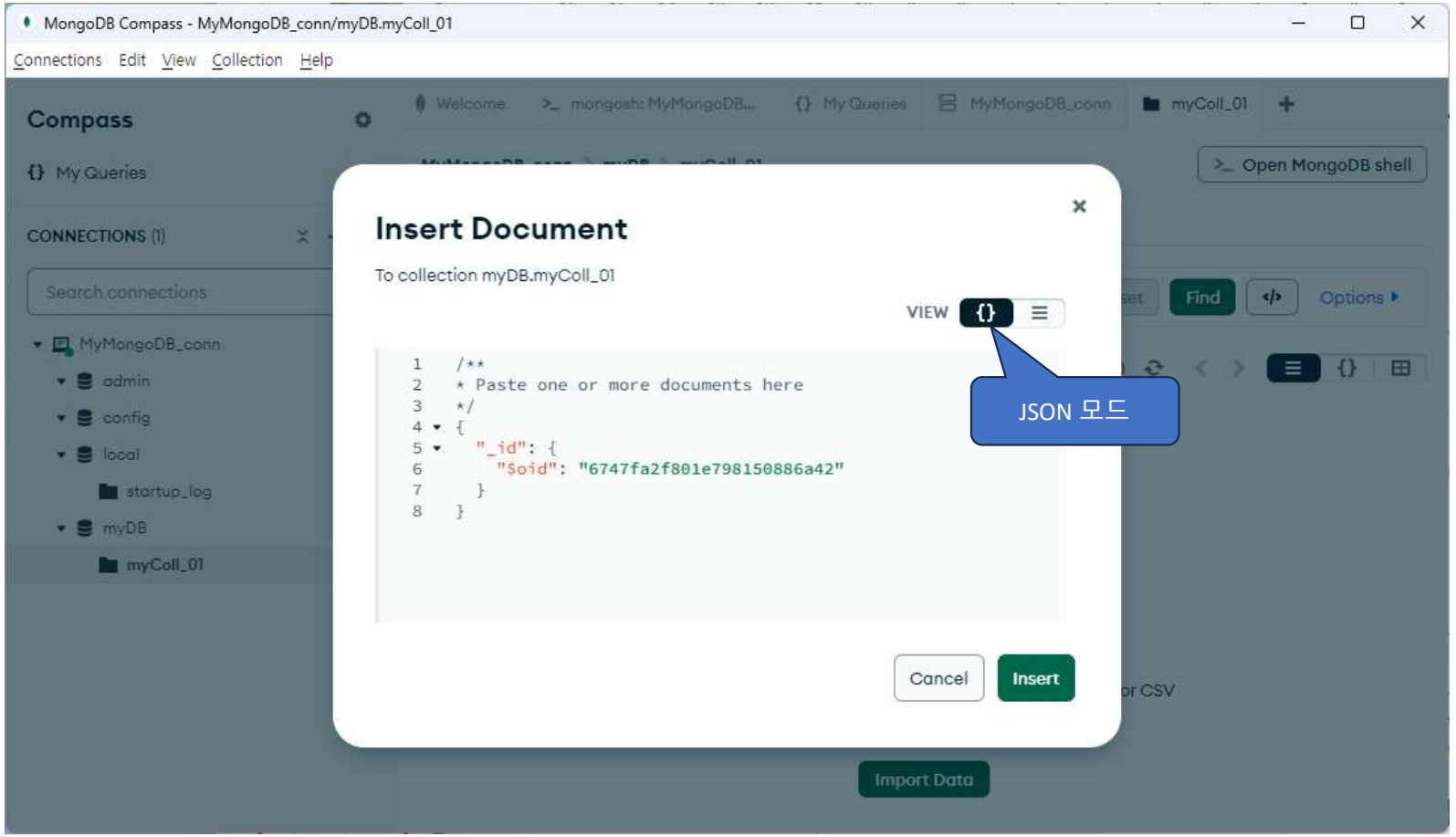
데이터베이스 생성 후 Collection 생성



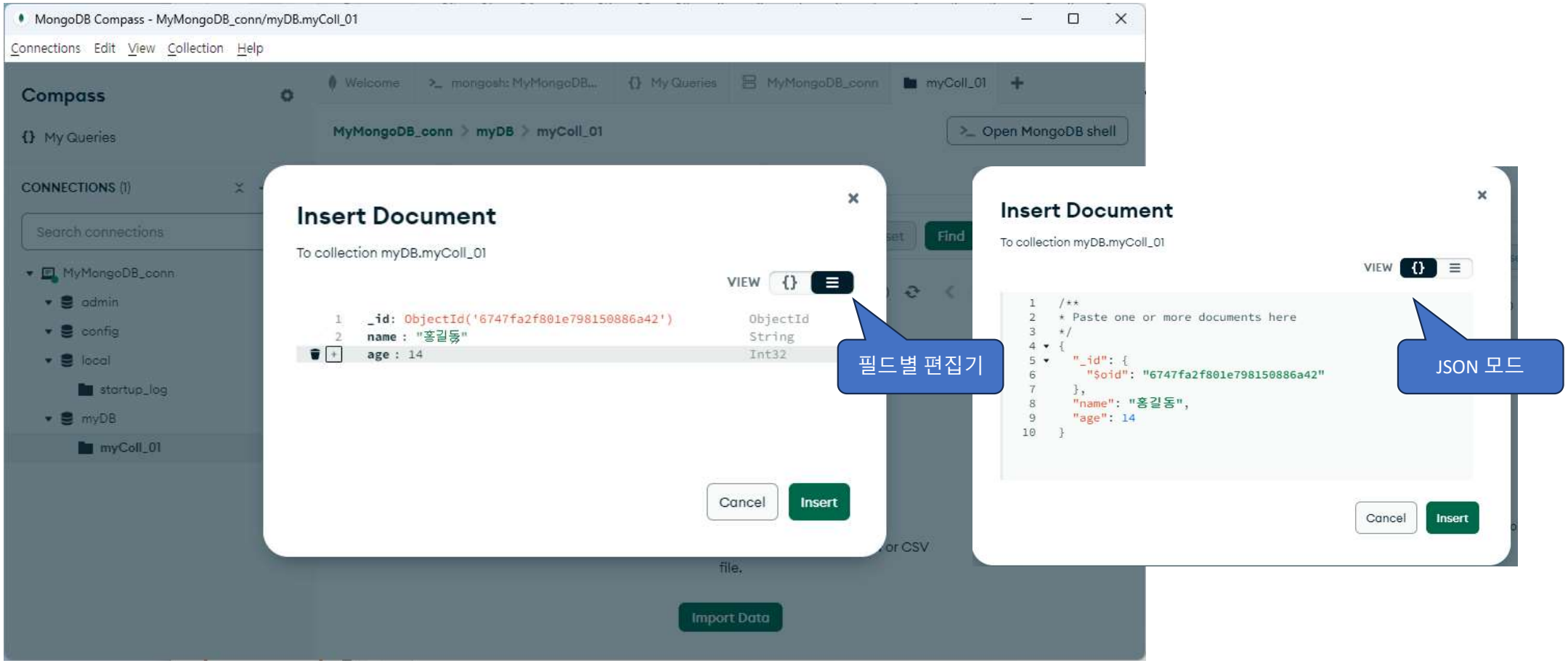
수동으로 Document 추가



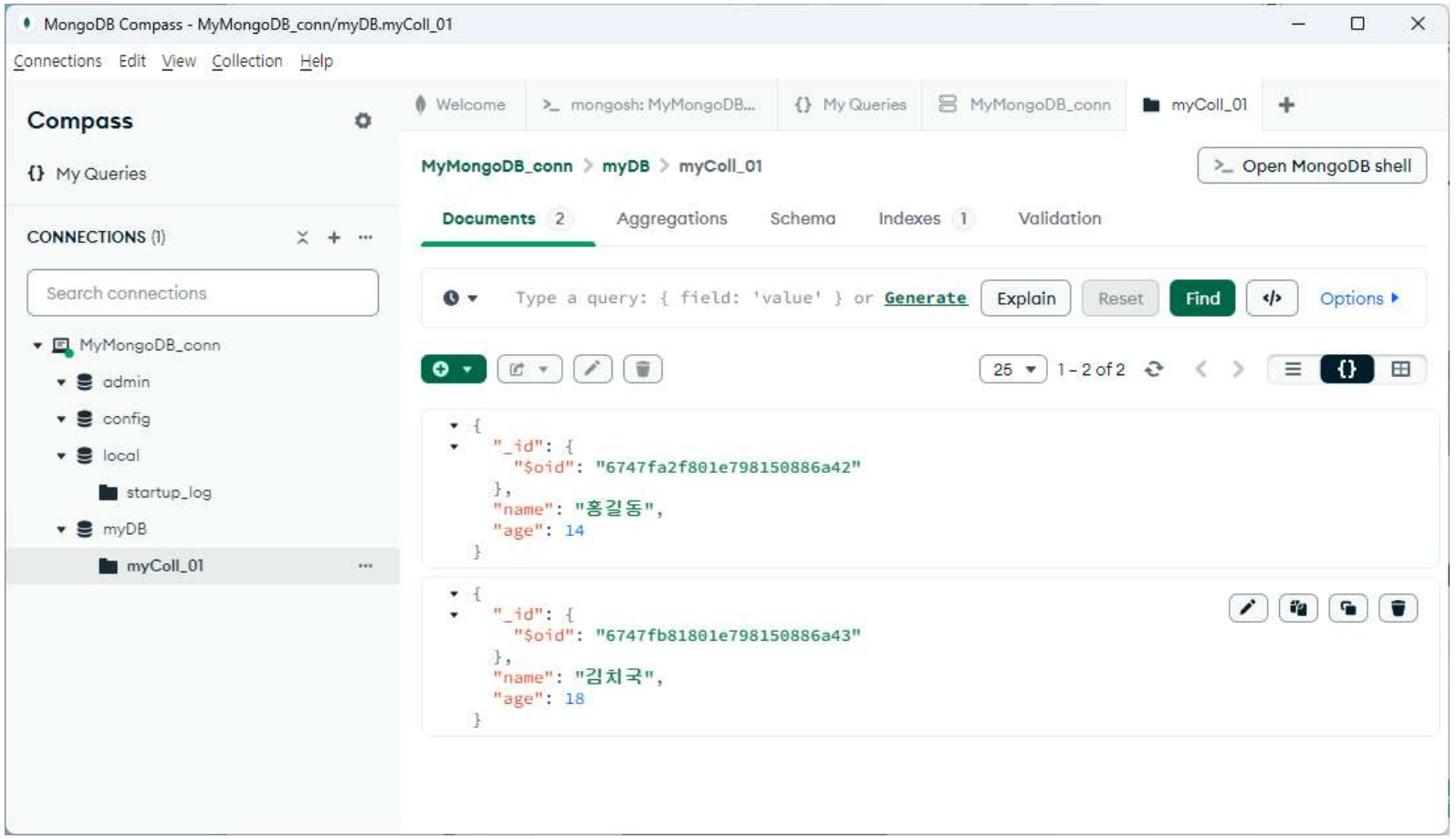
JSON 모드에서 Document 추가 방법: 이것보다 다음의 '필드별 편집기' 추천



필드별 편집기 형태로 Document 추가 방법



추가된 Document 보기



Chapter. 06

MongoDB (local/Remote) access

- 파이썬에서의 접근 방법

Python 사용 시 pymongo 인스톨

```
PS C:\dev\20241127-db> pip install pymongo
Collecting pymongo
  Downloading pymongo-4.10.1-cp311-cp311-win_amd64.whl.metadata (22 kB)
Collecting dnspython<3.0.0,>=1.16.0 (from pymongo)
  Downloading dnspython-2.7.0-py3-none-any.whl.metadata (5.8 kB)
Downloading pymongo-4.10.1-cp311-cp311-win_amd64.whl (876 kB)
   ---------------------------------------- 876.5/876.5 kB 14.0 MB/s eta 0:00:00
Downloading dnspython-2.7.0-py3-none-any.whl (313 kB)
   ---------------------------------------- 313.6/313.6 kB 20.2 MB/s eta 0:00:00
Installing collected packages: dnspython, pymongo
Successfully installed dnspython-2.7.0 pymongo-4.10.1

[notice] A new release of pip is available: 24.0 -> 24.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
PS C:\dev\20241127-db>
```

파이썬에서 동작 예

```python
from pymongo import MongoClient

client = MongoClient(host='localhost', port=27017)

print('---------- database names')
print(client.list_database_names())

# DB access
db = client['myDB']

print('---------- collection names')
for item in db.list_collection_names():
    print(item)

# collection access
coll = db['myColl_01']

# print all documents
print('---------- all documents')
for doc in coll.find():
    print(doc)

# document query
print('---------- name query')
for doc in coll.find({"name": "홍길동"}):
    print(doc)
```

```
PS C:\dev\20241127-db> python no_sql_acc.py
---------- database names
['admin', 'config', 'local', 'myDB']
---------- collection names
myColl_01
---------- all documents
{'_id': ObjectId('6747fa2f801e798150886a42'), 'name': '홍길동', 'age': 14}
{'_id': ObjectId('6747fb81801e798150886a43'), 'name': '김치국', 'age': 18}
---------- name query
{'_id': ObjectId('6747fa2f801e798150886a42'), 'name': '홍길동', 'age': 14}
---------- 2 documents add
---------- all documents
{'_id': ObjectId('6747fa2f801e798150886a42'), 'name': '홍길동', 'age': 14}
{'_id': ObjectId('6747fb81801e798150886a43'), 'name': '김치국', 'age': 18}
{'_id': ObjectId('67480fe6dedccf316ba1d11d'), 'name': '임신중', 'age': 28}
{'_id': ObjectId('67480fe6dedccf316ba1d11e'), 'name': '박찬호', 'age': 45}
---------- 1 document add with a additional field
---------- all documents
{'_id': ObjectId('6747fa2f801e798150886a42'), 'name': '홍길동', 'age': 14}
{'_id': ObjectId('6747fb81801e798150886a43'), 'name': '김치국', 'age': 18}
{'_id': ObjectId('67480fe6dedccf316ba1d11d'), 'name': '임신중', 'age': 28}
{'_id': ObjectId('67480fe6dedccf316ba1d11e'), 'name': '박찬호', 'age': 45}
{'_id': ObjectId('67480fe6dedccf316ba1d11f'), 'name': '수호자', 'age': 25, 'MBTI': 'ISFJ'}
{'_id': ObjectId('67480fe6dedccf316ba1d120'), 'name': '전략가', 'age': 26, 'MBTI': 'INTJ'}
PS C:\dev\20241127-db>
```

```python
print('---------- 2 documents add')
user_profiles = [
    {"name": "임신중", "age": 28},
    {"name": "박찬호", "age": 45}
]
result = coll.insert_many(user_profiles)
# print all documents
print('---------- all documents')
for doc in coll.find():
    print(doc)

print('---------- 2 documents add with an additional field')
user_profiles = [
    {"name": "수호자", "age": 25, "MBTI": "ISFJ"},
    {"name": "전략가", "age": 26, "MBTI": "INTJ"}
]
result = coll.insert_many(user_profiles)
# print all documents
print('---------- all documents')
for doc in coll.find():
    print(doc)

client.close()
```

Field 추가 등의
변경이 자유로움