

DB 연동하기

1. 데이터 베이스 소개

데이터 베이스 종류

챗봇 개발 시 대화 내용을 저장하기에 좋은 데이터베이스는 사용 사례와 요구 사항에 따라 다를 수 있다.

1. 관계형 데이터베이스 (RDBMS)

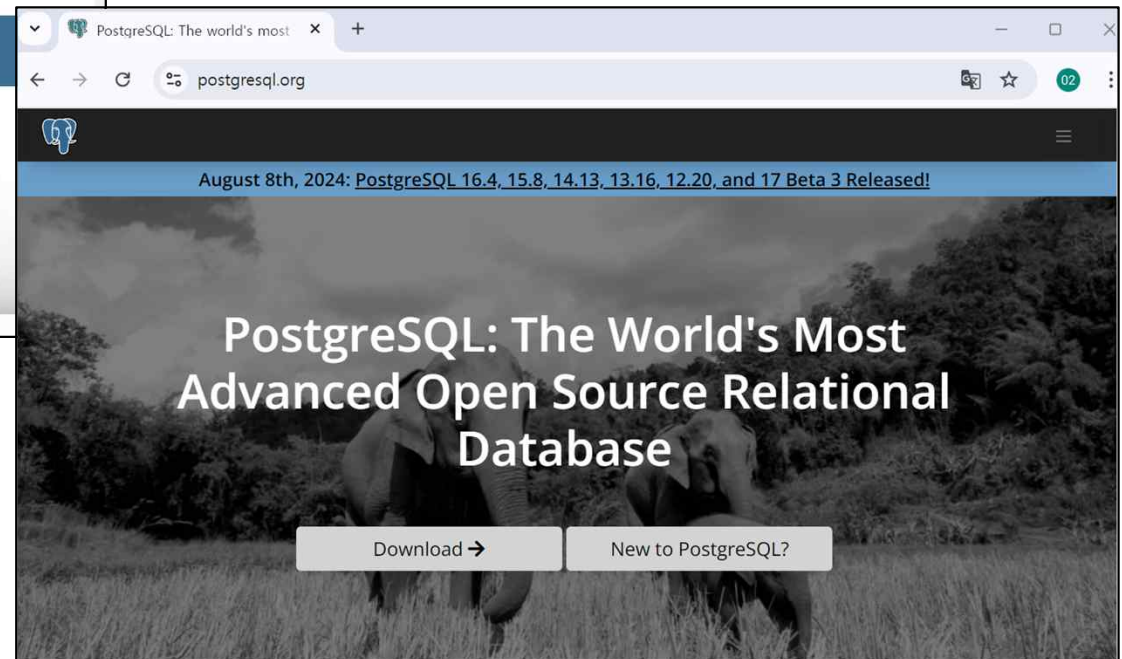
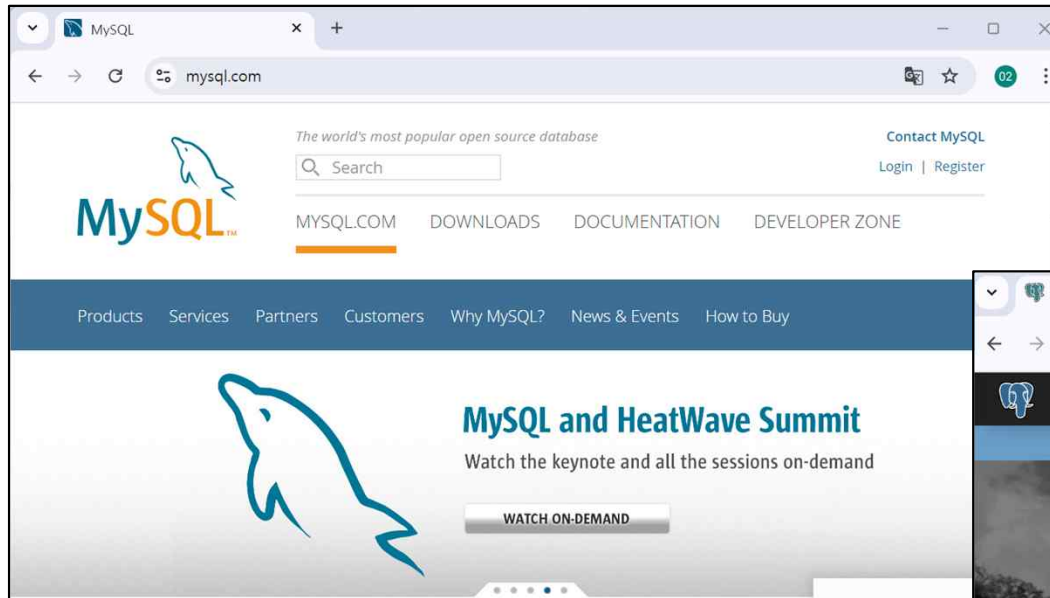
MySQL / PostgreSQL

장점 : SQL을 사용하여 복잡한 쿼리를 수행할 수 있고, 트랜잭션을 지원하여 데이터 무결성을 유지할 수 있다. 구조화된 데이터(예: 사용자의 프로필, 대화 로그)를 저장하는 데 적합하다.

단점 : 대규모 데이터 처리 시 성능이 저하될 수 있으며, 스키마 변경이 필요할 때 유연성이 떨어질 수 있다.

1. 데이터 베이스 소개

MySQL / PostgreSQL



2. NoSQL 데이터베이스

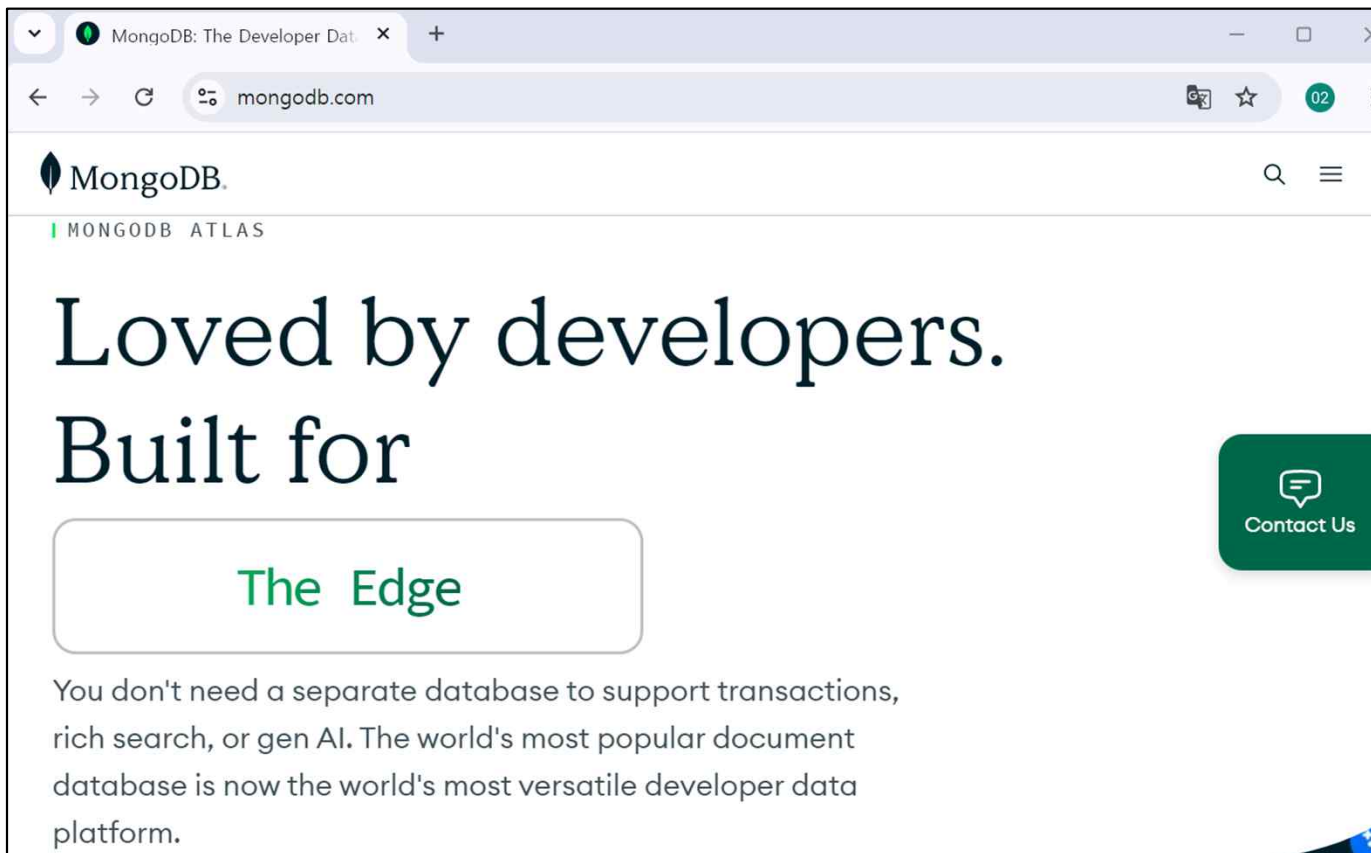
(1) MongoDB

장점: 스키마가 유연하여 다양한 형태의 데이터를 저장하기 좋습니다. 특히 대화 내용이 구조화되지 않은 경우 유용합니다. 확장성이 뛰어나 대규모 데이터 처리에도 적합합니다.

단점: 데이터 무결성 관리가 어렵고, 복잡한 쿼리 성능이 관계형 데이터베이스보다 떨어질 수 있습니다.

1. 데이터 베이스 소개

MongoDB



(2) Cassandra

장점: 대규모 분산 데이터를 처리하는 데 강점이 있으며,고가 용성과 확장성이 뛰어납니다. 챗봇의 대화 로그가 매우 큰 경우에 적합합니다.

단점: 쿼리 기능이 제한적이며, 초기 설정과 관리가 복잡할 수 있습니다.

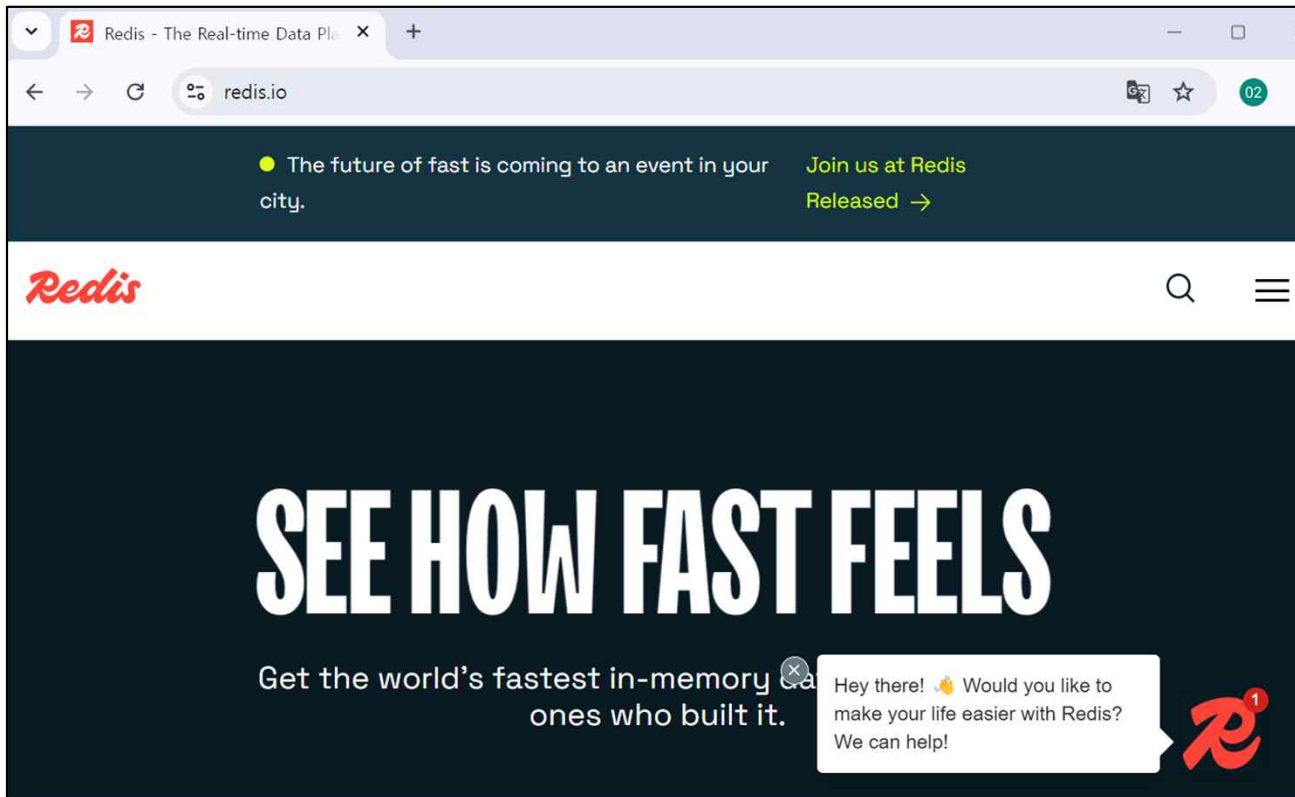
3. 키-값 저장소

- Redis

- **장점:** 빠른 속도와 높은 성능으로 실시간 처리에 유리.
대화 상태, 세션 정보 등을 저장하는 데 매우 유용하다.
- **단점:** 데이터가 **메모리에 저장**되므로, 대규모 데이터를 저장하기에는 비용이 많이 들 수 있다. 또한, 복잡한 데이터 구조를 관리하기 어렵다.

1. 데이터 베이스 소개

Redis



4. 시계열 데이터베이스

- InfluxDB

- **장점:** 시간 시계열 데이터(예: 대화 시간, 이벤트 로그)를 효과적으로 저장하고 쿼리할 수 있다. 대화 내용에 시간 흐름이 중요한 경우 유용하다.
- **단점:** 일반적인 데이터 저장용으로는 부적합하며, 특정 시계열 데이터에 특화되어 있다.

5. 문서 저장소

- ElasticSearch

- **장점:** 대용량 텍스트 데이터를 빠르게 검색할 수 있는 강력한 검색 기능을 제공. 대화 내용을 분석하거나 검색할 때 매우 유용하다.
- **단점:** 데이터 업데이트가 자주 일어나는 경우 성능이 저하될 수 있으며, 관계형 데이터 관리에 적합하지 않다.

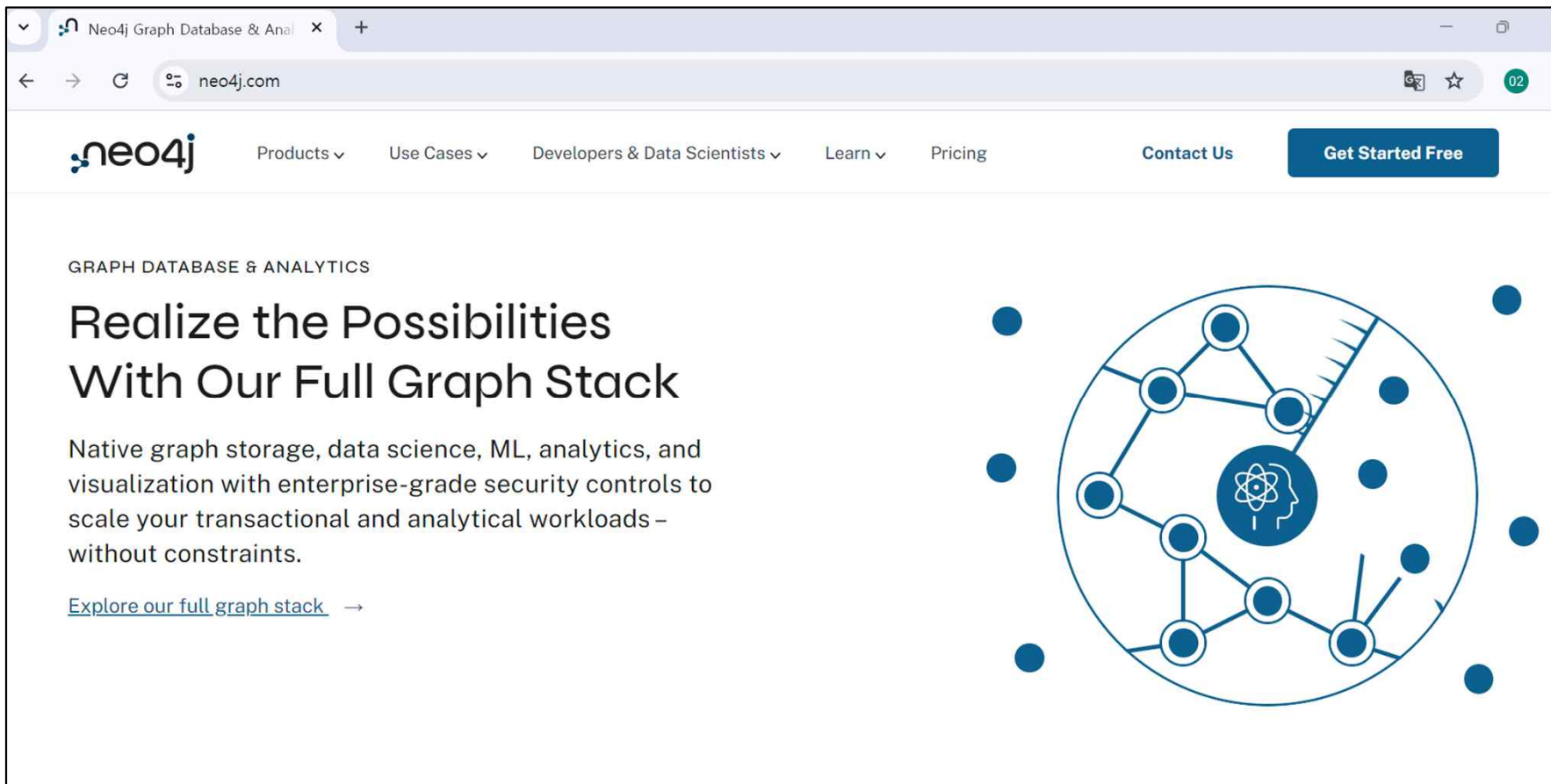
6. 그래프 데이터베이스

- **Neo4j**

- **장점:** 관계형 데이터베이스로는 다루기 힘든 복잡한 관계를 쉽게 표현하고 탐색할 수 있다. 대화 간의 관계, 대화 흐름 등을 시각화하고 분석하는 데 적합.
- **단점:** 그래프 쿼리가 익숙하지 않으면 사용이 어려울 수 있으며, 초기 학습 곡선이 가파를 수 있다.

1. 데이터 베이스 소개

Neo4j



DB 선택 가이드:

- 대화 내용이 구조화되어 있고, 데이터 무결성이 중요하다면
: **MySQL, PostgreSQL**
- 비구조화된 데이터를 유연하게 저장하고 싶다면
: **MongoDB**
- 대규모 데이터를 고속으로 처리하고 싶다면
: **Cassandra, Redis**
- 검색 기능이 중요하다면
: **ElasticSearch**
- 복잡한 데이터 관계를 관리하고 시각화하고 싶다면
: **Neo4j**

2. DB에 대화 내용 저장하기

SQLite3

2. DB에 대화 내용 저장하기

SQLite3 사용

SQLite는 MySQL나 PostgreSQL와 같은 데이터베이스 관리 시스템이지만, **서버가 아니라 응용 프로그램에 넣어 사용**하는 비교적 가벼운 데이터베이스이다. Anaconda에 파이썬 모듈이 기본적으로 내장되어 있다. 모든 데이터 베이스를 .db 확장자를 가지는 하나의 파일에 저장 한다. SQL 표준의 대부분을 준수한다

장점:

간단하고 빠름, 단일 파일 관리, 독립성과 휴대성

SQLite3는 자원이 제한된 환경에서도 효율적으로 작동하기 때문에, 내장형 시스템이나 로컬 애플리케이션에서 이상적입니다.

2. DB에 대화 내용 저장하기

단점

(1) 동시성 제한

SQLite3는 단일 쓰기 잠금 방식을 사용하기 때문에, 동시에 여러 클라이언트가 데이터를 쓰는 상황에서는 성능이 저하될 수 있다. 많은 사용자가 동시 접속하는 환경에서는 적합하지 않다.

(2) 확장성 문제

대규모 데이터셋을 처리하거나, 고성능이 요구되는 작업에는 한계가 있다.(소규모, 로컬 데이터베이스 애플리케이션에 최적화된 시스템)

(3) 네트워크 기반 애플리케이션에 부적합

SQLite3는 네트워크에서 동작하는 데이터베이스로 사용하기 적합하지 않다. 주로 로컬 환경에서 사용되며, 네트워크를 통한 동시 접근이 빈번한 애플리케이션에서는 성능이 떨어질 수 있다.

2. DB에 대화 내용 저장하기

SQLite3 DB 사용 파이썬 코드 구현 : SQLite3 DB 초기화

```
1 # SQLite3 DB 초기화 : sample.db
2 import sqlite3
3
4 # sample.db라는 파일 기반의 SQLite 데이터베이스에 연결
5 # 해당 파일이 없으면 새로 생성한다.
6 conn = sqlite3.connect('sample.db')
7
8 # SQL 명령을 실행하기 위해 커서 객체를 생성
9 c = conn.cursor()
10
11 # 테이블 생성: conversations라는 테이블을 생성. id는 기본 키이며 자동으로 증가합니다.
12 # question, answer 컬럼(스키마)을 정의
13 c.execute('''
14     CREATE TABLE IF NOT EXISTS conversations (
15         id INTEGER PRIMARY KEY AUTOINCREMENT,
16         question TEXT,
17         answer TEXT
18     )
19 ''')
20
21 conn.commit() # 변경 사항 저장 (커밋)
22 conn.close() # 데이터베이스 연결 종료
```

2. DB에 대화 내용 저장하기

SQLite3 DB 사용 파이썬 코드 구현 : DB에 데이터 저장하기

```
1 # DB에 데이터 저장하기
2
3 # 저장할 데이터
4 question = '하늘은 왜 파란가요?'
5 answer = '하늘이 파란 이유는 주로 대기 중의 산란 현상 때문입니다'
6
7 # sample.db라는 파일 기반의 SQLite 데이터베이스에 연결
8 # 해당 파일이 없으면 새로 생성한다.
9 conn = sqlite3.connect('sample.db')
10
11 # SQL 명령을 실행하기 위해 커서 객체를 생성
12 c = conn.cursor()
13
14 # 데이터 삽입 : conversations 테이블에 데이터를 삽입
15 # ?는 자리표시자(placeholder)로, 실제 값은 튜플로 전달된다.
16
17 c.execute('''
18     INSERT INTO conversations (question, answer) VALUES (?, ?)
19 ''', (question, answer))
20
21 conn.commit() # 변경 사항 저장 (커밋)
22 conn.close() # 데이터베이스 연결 종료
```

2. DB에 대화 내용 저장하기

SQLite3 DB 사용 파이썬 코드 구현 : 데이터 조회 및 출력

```
1 # 데이터 조회 및 출력
2 conn = sqlite3.connect('sample.db')
3 c = conn.cursor()
4
5 # 테이블의 모든 데이터를 조회
6 c.execute('SELECT * FROM conversations')
7
8 # SQL 쿼리의 결과로 반환된 모든 행을 한꺼번에 가져오고 리스트로 반환
9 conversations = c.fetchall()
10
11 # 결과 출력
12 print(conversations)
13
14 conn.close() # 데이터베이스 연결 종료
```

2. DB에 대화 내용 저장하기

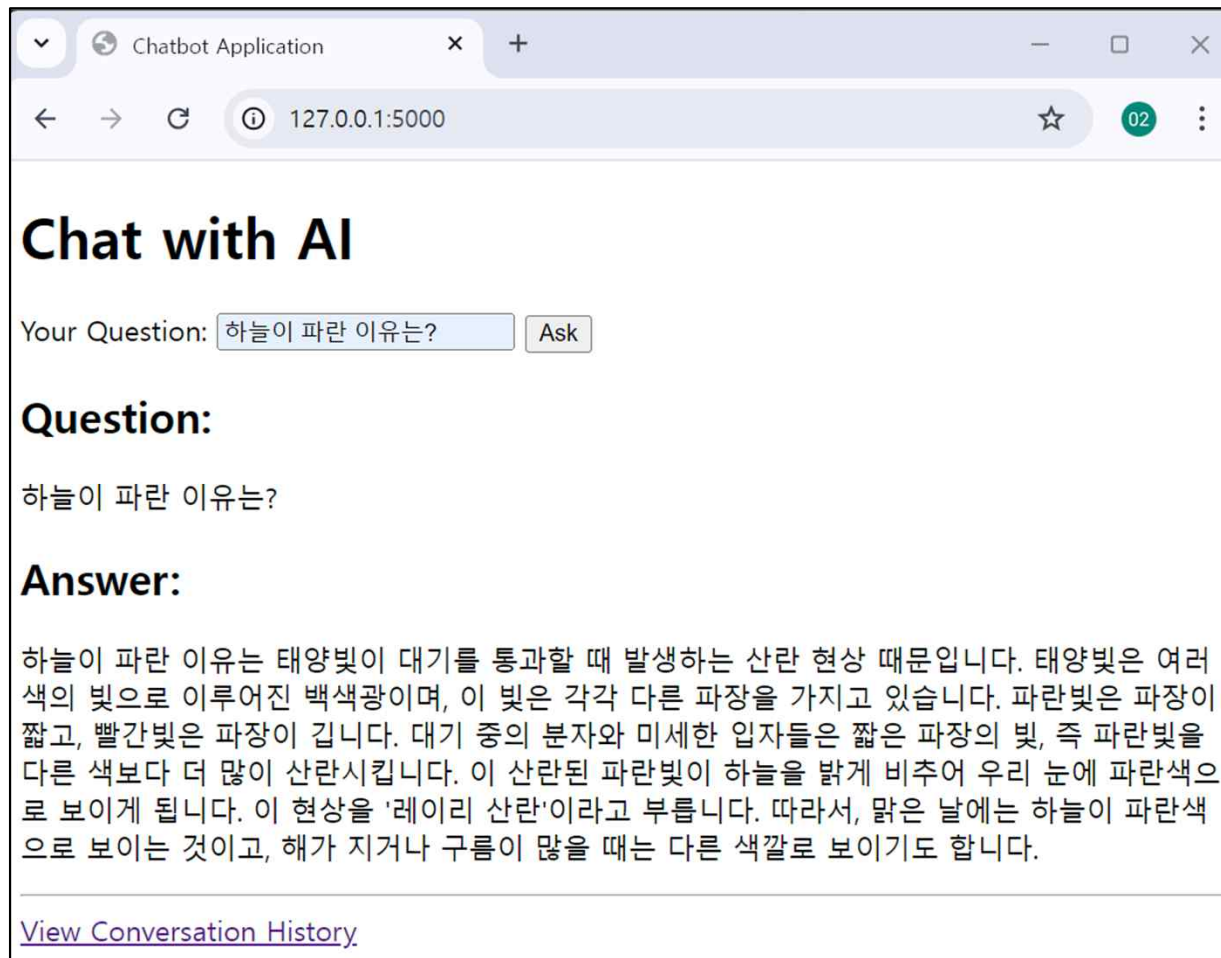
SQLite3 DB 사용 파이썬 코드 구현 : 함수로 구현

```
1 %%writefile database.py
2 import sqlite3
3
4 def init_db():
5     conn = sqlite3.connect('sample.db')
6     c = conn.cursor()
7     c.execute('''
8         CREATE TABLE IF NOT EXISTS conversations (
9             id INTEGER PRIMARY KEY AUTOINCREMENT,
10             question TEXT,
11             answer TEXT
12         )
13     ''')
14     conn.commit()
15     conn.close()
16
17 def save_conversation(question, answer):
18     conn = sqlite3.connect('sample.db')
19     c = conn.cursor()
20     c.execute('''
```

%%는 파이썬 매직 커맨드로
%%writefile database.py는 셀의
모든 내용을 database.py파일로
저장한다

2. DB에 대화 내용 저장하기

SQLite3 사용 챗봇 Flask 웹 페이지 실행 화면



2. DB에 대화 내용 저장하기

DB에 저장된 대화내용 불러오기 출력 화면

Conversation History

127.0.0.1:5000/history

Conversation History

ID	Question	Answer
1	Hello	Hello! How can I assist you today?
2	하늘은 왜 파란가?	하늘이 파란 이유는 주로 대기 중의 산란 현상 때문입니다. 태양빛은 여러 색의 빛이 혼합되어 있는 백색광인데, 이 중 파란색 빛은 파장이 짧고 에너지가 높습니다. 대기 중의 분자와 미세한 입자들이 태양빛을 산란시키는데, 파란색 빛은 다른 색상보다 더 많이 산란됩니다. 이 현상을 레일리 산란(Rayleigh scattering)이라고 하며, 이로 인해 하늘이 주로 파란색으로 보이게 됩니다. 해가 떠 있는 동안에는 하늘이 파란색으로 보이고, 해가 지거나 해가 낮은 위치에 있을 때는 주황색이나 빨간색으로 변하는 이유도 대기의 두께와 산란의 원리에 기인합니다.
3	오늘 날씨 는?	죄송하지만, 실시간 날씨 정보를 제공할 수는 없습니다. 현재 위치의 날씨를 확인하시려면 날씨 앱이나 웹사이트를 이용해 보시기 바랍니다. 다른 질문이 있으시면 도와드리겠습니다!
4	내 컴퓨터 가 켜지지 않아?	컴퓨터가 켜지지 않는 경우에는 여러 가지 원인이 있을 수 있습니다. 다음은 문제를 진단하고 해결하는 데 도움이 될 수 있는 몇 가지 단계입니다: 1. **전원 확인**: - 전원 케이블이 제대로 연결되어 있는지 확인하세요. - 멀티탭이나 전원 콘센트가 작동하는지 확인합니다. 다른 기기를 연결해보세요. 2. **전원 버튼**: - 전원 버튼을 길게 눌러서 껐다가 다시 켜보세요. 3. **LED 및 팬 소음 확인**: - 전원이 들어오면 LED가 켜지거나 팬 소음이 나야 합니다. 그렇지 않다면 전원 공급 장치에 문제가 있을 수 있습니다. 4. **하드웨어 점검**: - 내부 하드웨어 (RAM, 그래픽 카드 등)가 제대로 장착되어 있는지 확인합니다. - 최근에 하드웨어를 추가했거나 변경한 경우, 해당 부품이 문제를 일으킬 수 있습니다. 5. **모니터 확인**: - 모니터가 켜져 있는지, 케이블이 제대로 연결되어 있는지 확인하세요. - 다른 모니터를 연결해보는 것도 좋은 방법입니다. 6. **부팅 소음 및 비프음**: - 컴퓨터가 켜지지 않는 문제는 여러 가지 원인으로 발생할 수 있습니다. 다음은 몇 가지 점검 사항입니다: 1. **전

3. MongoDB에 대화 내용 저장하기

3. MongoDB에 대화 내용 저장하기

MongoDB

NoSQL 데이터베이스의 한 종류로, 데이터를 유연하게 저장하고 관리할 수 있는 **도큐먼트 DB** 시스템이다. 전통적인 관계형 데이터베이스 (RDBMS)와 달리 MongoDB는 데이터를 테이블과 행 대신 문서 (document) 형태로 저장한다. 이 문서들은 JSON과 유사한 **BSON(Binary JSON) 형식**으로 저장되며, 유연한 데이터 구조를 가질 수 있다.

MongoDB의 주요 특징:

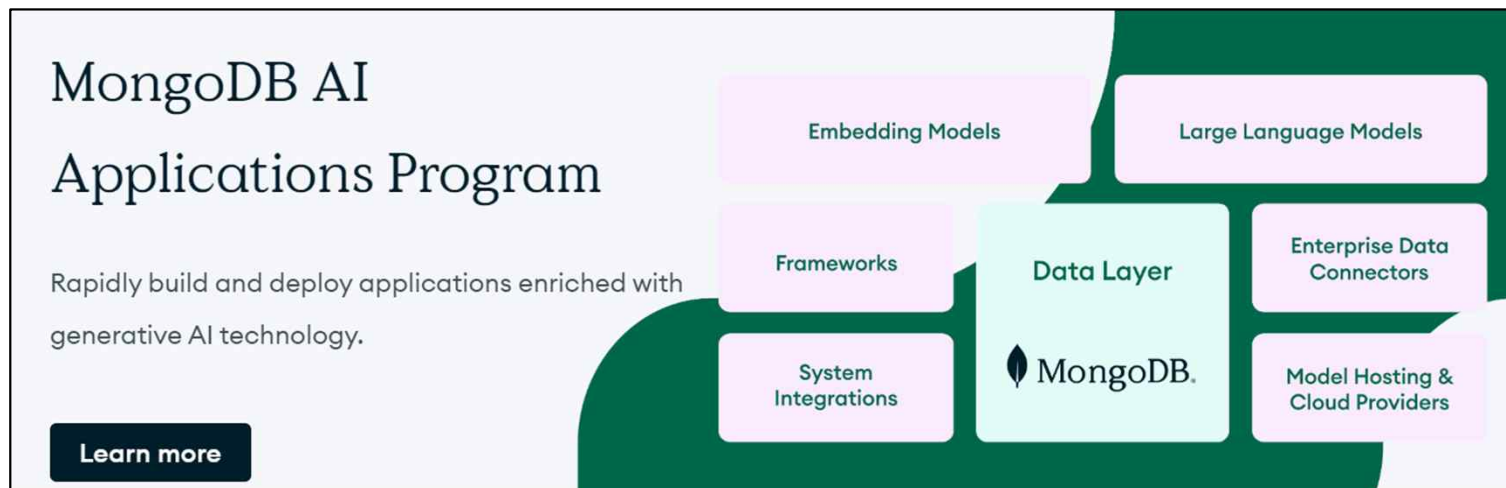
(1) 스키마가 없는 데이터베이스 : 미리 정해진 스키마 없이 데이터를 저장할 수 있어, 각 문서가 다른 구조를 가질 수 있다. 이는 다양한 형태의 데이터를 쉽게 처리할 수 있게 해준다.

(2) 유연한 확장성 : MongoDB는 데이터를 여러 서버에 분산시켜 저장할 수 있어, 대규모 데이터를 처리하는 데 적합하다.

3. MongoDB에 대화 내용 저장하기

(3) 고속 읽기/쓰기 : MongoDB는 데이터 읽기와 쓰기 작업이 빠르며, 특히 대용량 데이터 처리에 강점이 있습니다.

(4) 계층적 데이터 구조 : MongoDB의 문서는 중첩된 형태의 데이터를 가질 수 있어, 관계형 데이터베이스에서 테이블 조인을 사용해야 하는 복잡한 데이터를 더 직관적으로 관리할 수 있습니다.



3. MongoDB에 대화 내용 저장하기

MongoDB 구조 :

데이터베이스 (Database): 관련된 데이터의 집합. 예를 들어, myDatabase라는 이름의 데이터베이스를 가질 수 있다.

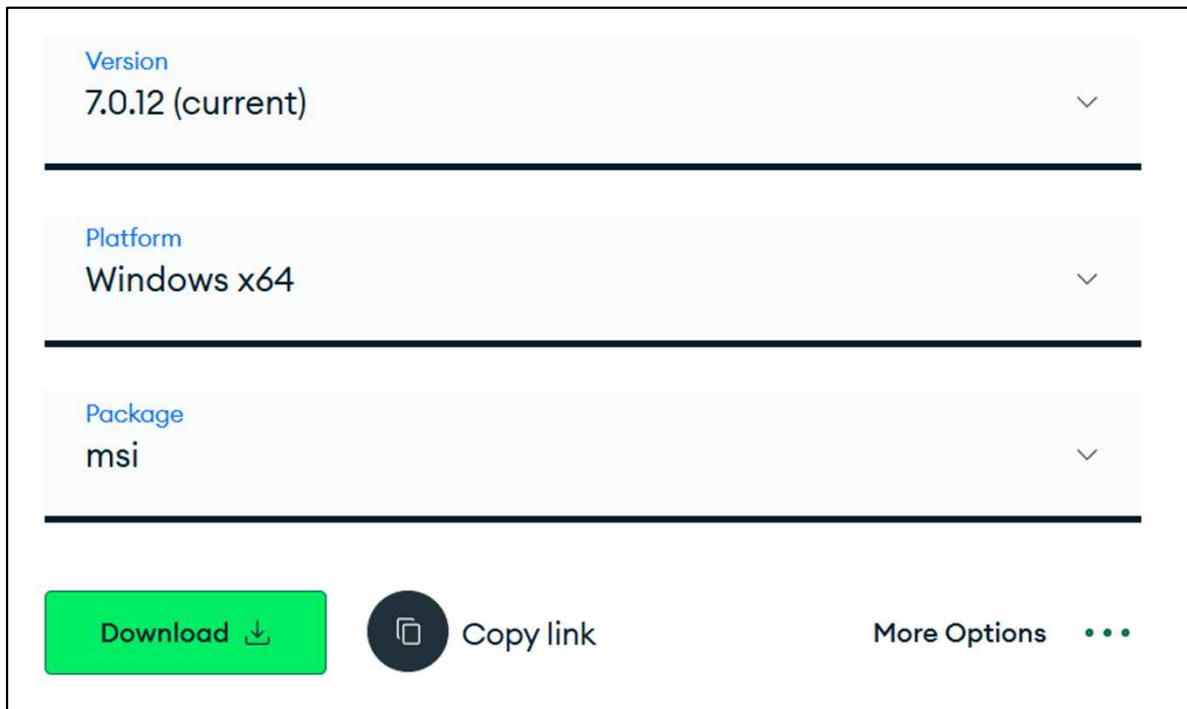
컬렉션 (Collection): RDBMS의 테이블과 비슷한 개념으로, 문서들의 집합이다. 예를 들어, users라는 컬렉션에 사용자 정보를 저장할 수 있다.

문서 (Document): 데이터의 기본 단위로, 각 문서는 고유한 구조를 가질 수 있다. 예를 들어, 사용자 정보(이름, 나이, 이메일 등)가 하나의 문서로 저장된다.

3. MongoDB에 대화 내용 저장하기

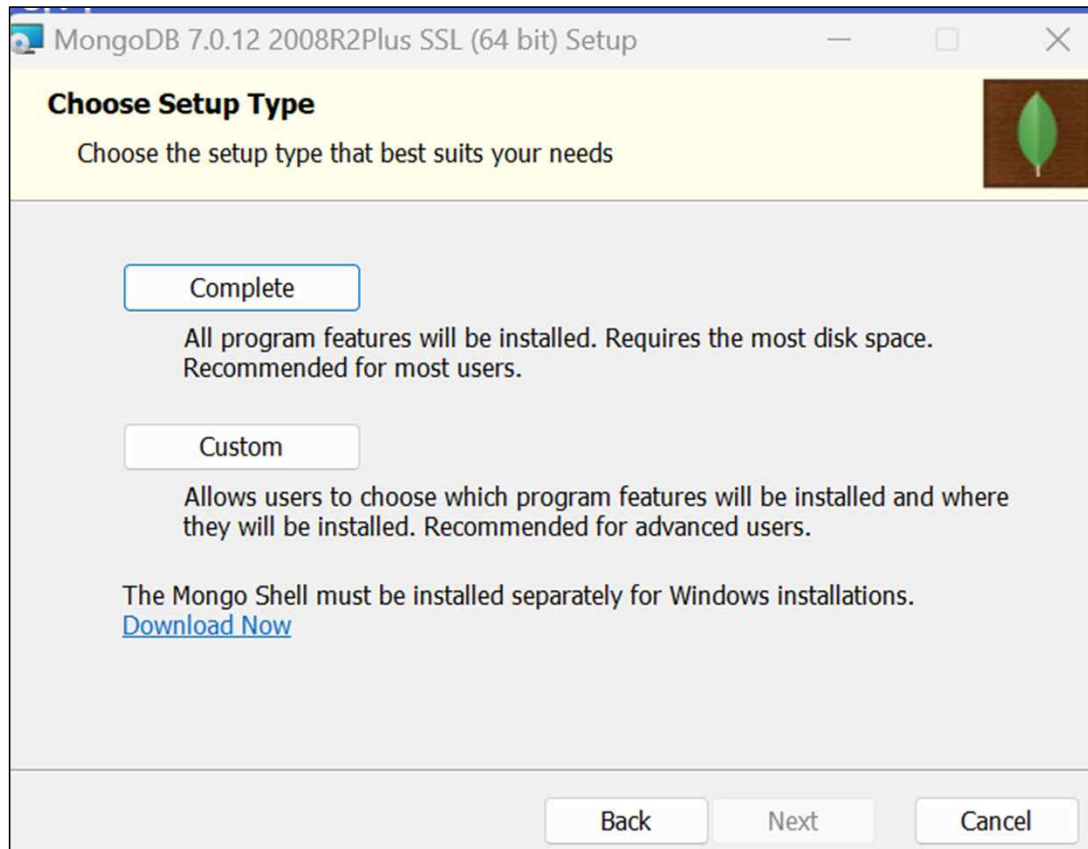
MongoDB 서버 설치(Windows x64)

<https://www.mongodb.com/try/download/community>에서 아래로 스크롤해서
mongodb-windows-x86_64-7.0.12-signed.msi 파일을 다운받아 설치한다



3. MongoDB에 대화 내용 저장하기

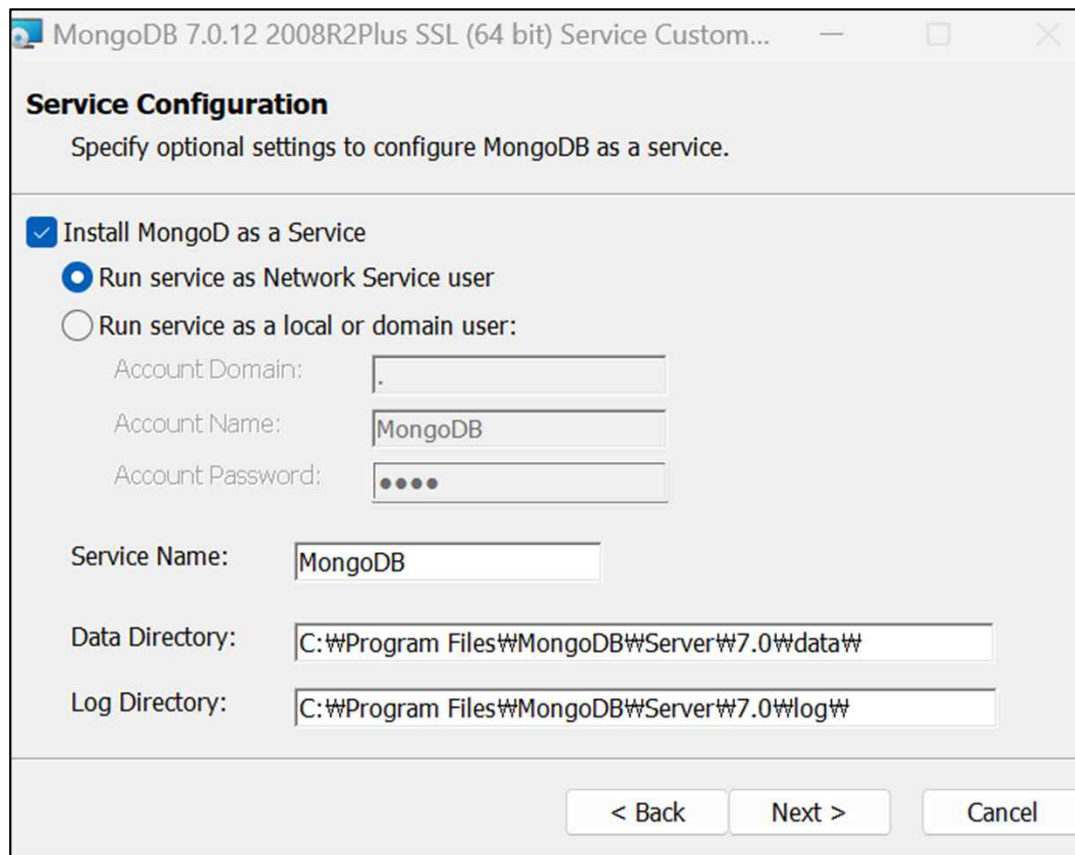
MongoDB 서버 설치(Windows x64) : Complete 선택



3. MongoDB에 대화 내용 저장하기

MongoDB 서버 설치(Windows x64)

: Run servicer as Network Server user 를 선택하고 Next 클릭



The screenshot shows the 'Service Configuration' window for MongoDB 7.0.12. The window title is 'MongoDB 7.0.12 2008R2Plus SSL (64 bit) Service Custom...'. The main heading is 'Service Configuration' with the subtitle 'Specify optional settings to configure MongoDB as a service.'.

The 'Install MongoDB as a Service' checkbox is checked. Below it, the 'Run service as Network Service user' radio button is selected. The 'Run service as a local or domain user:' option is unselected, and its associated fields (Account Domain, Account Name, Account Password) are empty.

The 'Service Name' field is filled with 'MongoDB'. The 'Data Directory' field is filled with 'C:\\Program Files\\MongoDB\\Server\\7.0\\data\\'. The 'Log Directory' field is filled with 'C:\\Program Files\\MongoDB\\Server\\7.0\\log\\'.

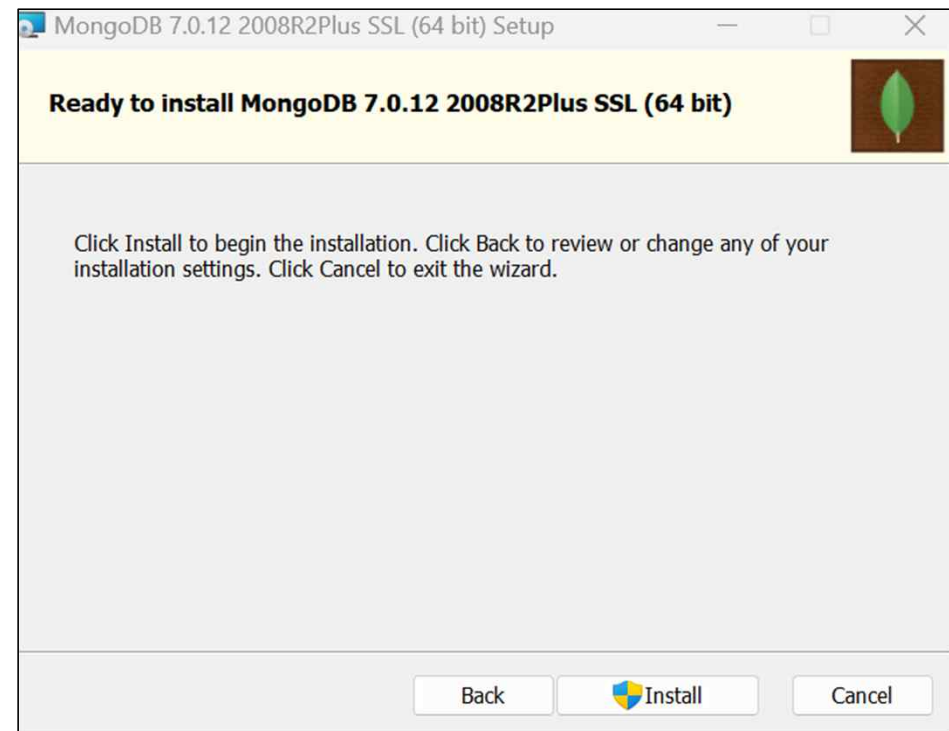
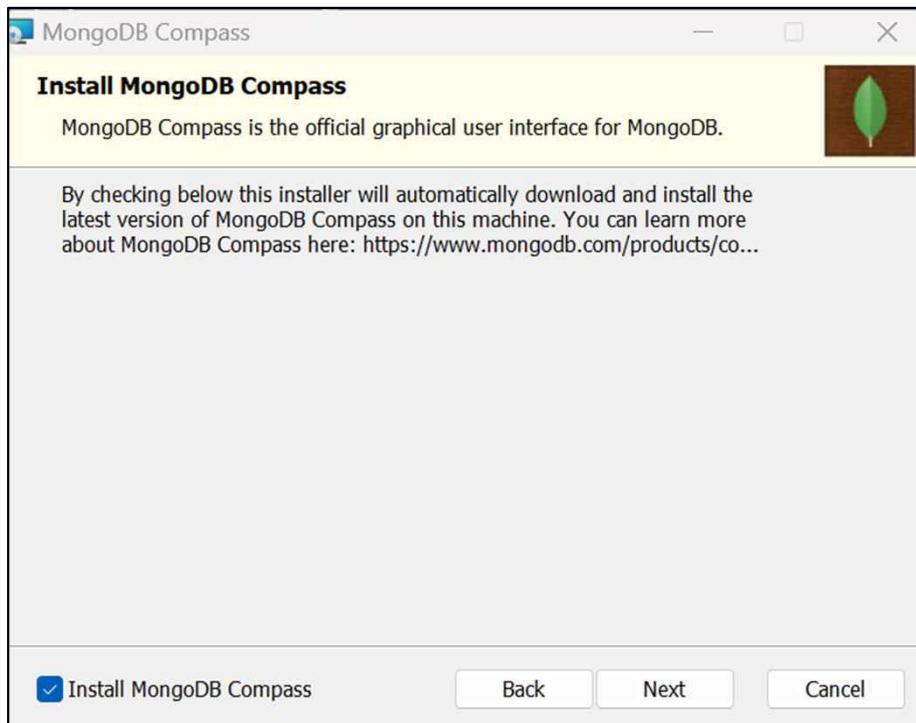
At the bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'.

3. MongoDB에 대화 내용 저장하기

OpenAI API 사용 Flask 웹 페이지 화면

: Next 클릭

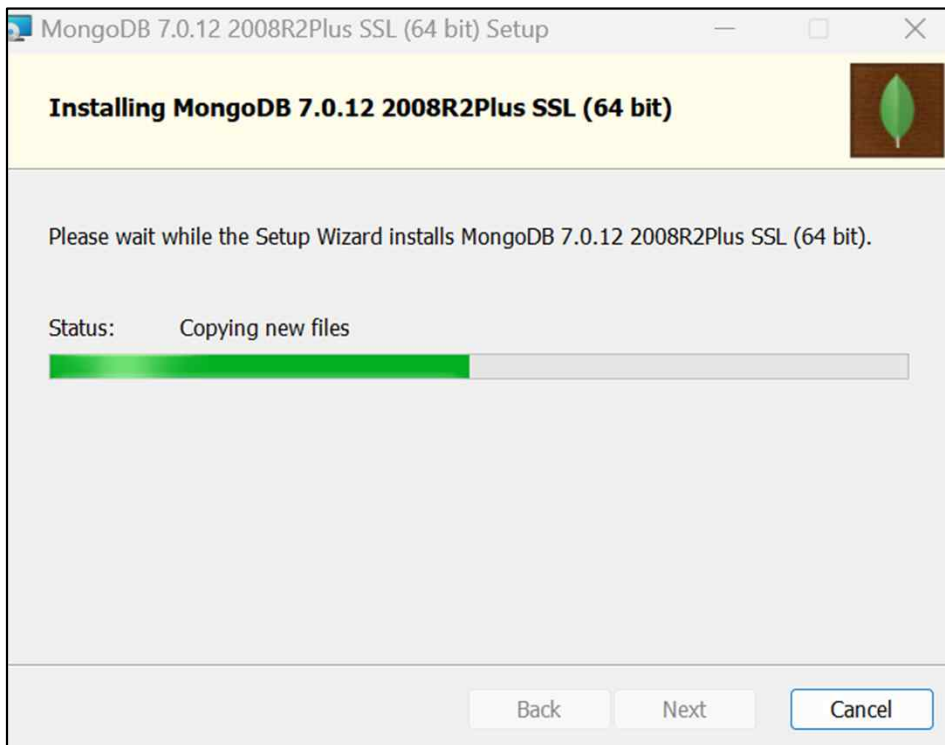
Install 클릭



3. MongoDB에 대화 내용 저장하기

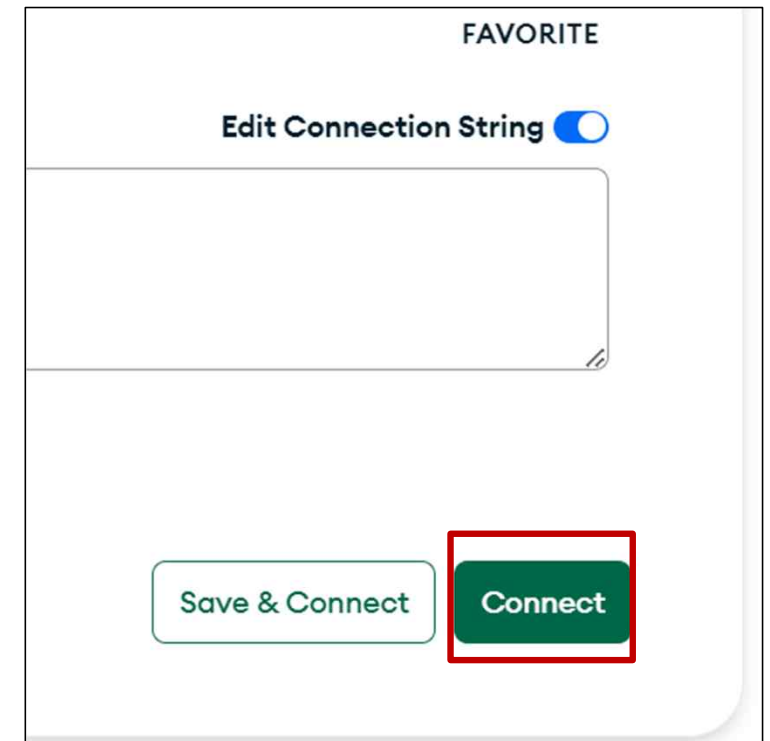
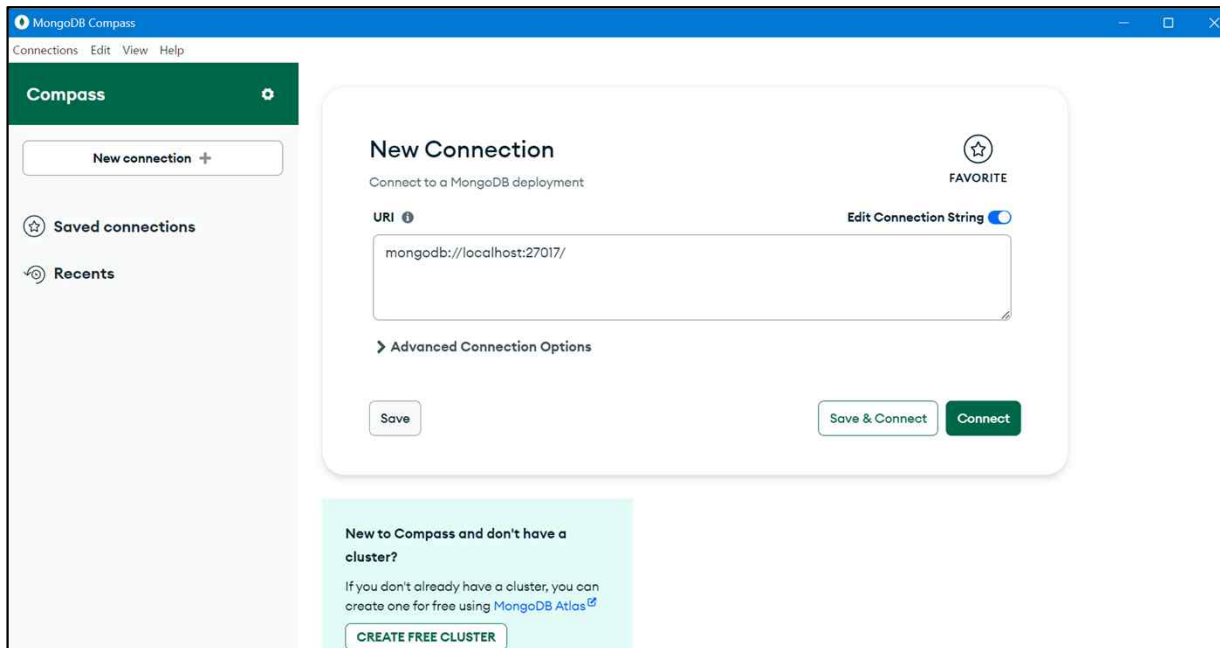
MongoDB 서버 설치(Windows x64)

: 수분 경과하여 설치 완료 된 후 Next 클릭한 후 Finish를 클릭하여 완료



3. MongoDB에 대화 내용 저장하기

MongoDB 모니터링: MongoDBCompass

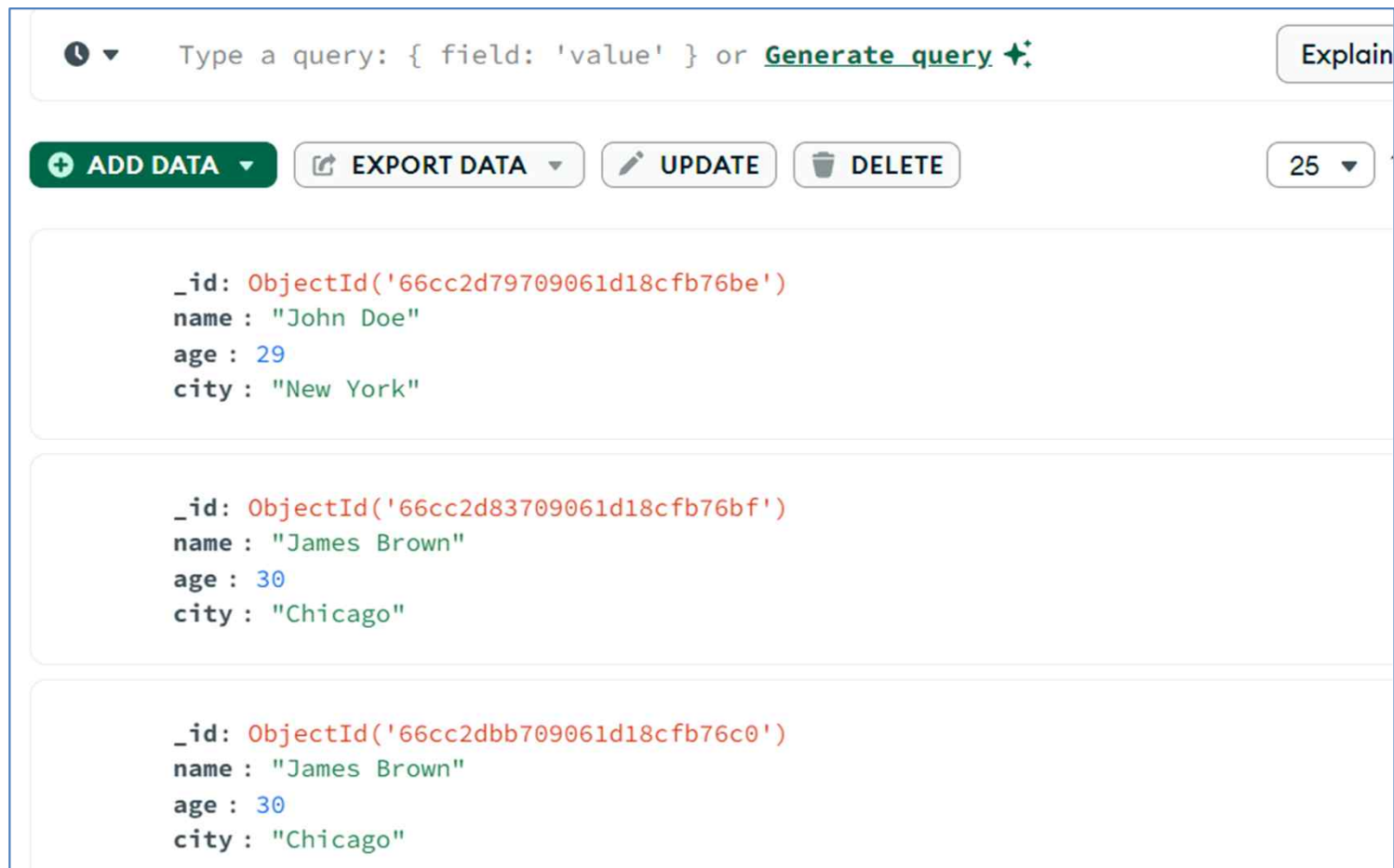


윈도 시작 메뉴에서 MongoDBCompass
실행

Connect 버튼 클릭

3. MongoDB에 대화 내용 저장하기

MongoDB 모니터링: MongoDBCompass

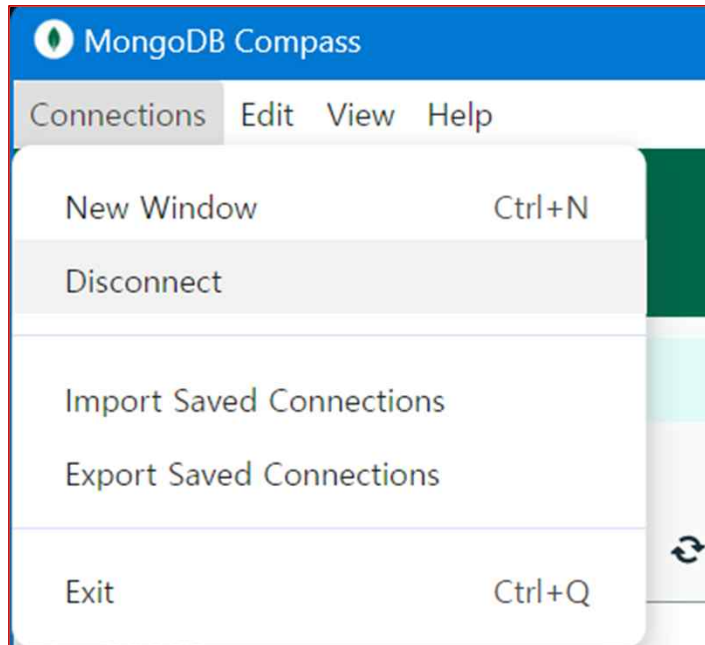


The screenshot displays the MongoDB Compass web interface. At the top, there is a search bar with the placeholder text "Type a query: { field: 'value' } or [Generate query](#)". To the right of the search bar is an "Explain" button. Below the search bar, there is a row of action buttons: "ADD DATA" (with a plus icon), "EXPORT DATA" (with a download icon), "UPDATE" (with a pencil icon), and "DELETE" (with a trash icon). To the right of these buttons is a dropdown menu showing "25". The main area of the interface displays a list of three documents. Each document is shown in a light gray box with a rounded border. The documents are as follows:

- Document 1:
`_id: ObjectId('66cc2d79709061d18cfb76be')`
`name: "John Doe"`
`age: 29`
`city: "New York"`
- Document 2:
`_id: ObjectId('66cc2d83709061d18cfb76bf')`
`name: "James Brown"`
`age: 30`
`city: "Chicago"`
- Document 3:
`_id: ObjectId('66cc2dbb709061d18cfb76c0')`
`name: "James Brown"`
`age: 30`
`city: "Chicago"`

3. MongoDB에 대화 내용 저장하기

MongoDBCompass 연결 중지



3. MongoDB에 대화 내용 저장하기

MongoDB 파이썬 라이브러리 설치(Jupyter Notebook)

```
! pip install pymongo
```

```
Collecting pymongo
```

```
  Downloading pymongo-4.8.0-cp310-cp310-win_amd64.whl.metadata (22 kB)
```

```
Collecting dnspython<3.0.0,>=1.16.0 (from pymongo)
```

```
  Using cached dnspython-2.6.1-py3-none-any.whl.metadata (5.8 kB)
```

```
Downloading pymongo-4.8.0-cp310-cp310-win_amd64.whl (582 kB)
```

```
----- 0.0/582.0 kB ? eta -:--:--
```

```
----- 122.9/582.0 kB 3.6 MB/s eta 0:00:01
```

```
----- 532.5/582.0 kB 6.7 MB/s eta 0:00:01
```

```
----- 582.0/582.0 kB 6.1 MB/s eta 0:00:00
```

```
Using cached dnspython-2.6.1-py3-none-any.whl (307 kB)
```

```
Installing collected packages: dnspython, pymongo
```

```
Successfully installed dnspython-2.6.1 pymongo-4.8.0
```

3. MongoDB에 대화 내용 저장하기

MongoDB 파이썬 클라이언트 실행

```
from pymongo import MongoClient

# MongoDB에 연결 (localhost:27017이 기본 설정)
client = MongoClient('localhost', 27017)

# 'testdb'라는 데이터베이스에 연결 (없으면 자동으로 생성됩니다)
db = client.testdb

# 'testcollection'이라는 컬렉션에 연결 (없으면 자동으로 생성됩니다)
collection = db.testcollection

# 삽입할 데이터 (JSON 형식)
data = {
    "name": "John Doe",
    "age": 29,
    "city": "New York"
}
```

3. MongoDB에 대화 내용 저장하기

MongoDB 파이썬 클라이언트 실행

```
# 데이터 삽입
inserted_id = collection.insert_one(data).inserted_id
print(f"Inserted document ID: {inserted_id}")

# 데이터 조회
result = collection.find_one({"name": "John Doe"})
print(f"Found document: {result}")
```

Inserted document ID: 66cc23ea73541bb315af224f

Found document: {'_id': ObjectId('66cc23ea73541bb315af224f'), 'name': 'John Doe', 'age': 29, 'city': 'New York'}

3. MongoDB에 대화 내용 저장하기

MongDB 사용 챗봇 대화 저장용 파이썬 함수 구현

```
# database.py
from pymongo import MongoClient

# MongoDB에 연결
client = MongoClient('localhost', 27017)
db = client['chatbot']
conversations_collection = db['conversations']

def init_db():
    # MongoDB에서는 컬렉션이 필요할 때 자동으로 생성되므로 초기화 과정이 필요 없다.
    pass

def save_conversation(question, answer):
    conversation = {
        'question': question,
        'answer': answer
    }
    conversations_collection.insert_one(conversation)
```

%%writefile database.py
를 수행하여 셀의 모든 내용을
database.py파일로 저장한다

2. DB에 대화 내용 저장하기

SQLite3 DB 사용 파이썬 코드 구현 : 함수로 구현

```
1 %%writefile database.py
2 import sqlite3
3
4 def init_db():
5     conn = sqlite3.connect('sample.db')
6     c = conn.cursor()
7     c.execute('''
8         CREATE TABLE IF NOT EXISTS conversations (
9             id INTEGER PRIMARY KEY AUTOINCREMENT,
10             question TEXT,
11             answer TEXT
12         )
13     ''')
14     conn.commit()
15     conn.close()
16
17 def save_conversation(question, answer):
18     conn = sqlite3.connect('sample.db')
19     c = conn.cursor()
20     c.execute('''
```

%%는 파이썬 매직 커맨드로
%%writefile database.py는 셀의
모든 내용을 database.py파일로
저장한다

3. MongoDB에 대화 내용 저장하기

MongoDB 사용 챗봇 대화 저장용 파이썬 함수 구현

```
def get_conversations():  
    conversations = conversations_collection.find({})  
    return list(conversations)  
  
# 데이터베이스 연결 종료 (원한다면, 하지만 일반적으로 MongoDB 연결을 유지해도 문제 없습니다.)  
def close_connection():  
    client.close()
```

3. MongoDB에 대화 내용 저장하기

MongDB사용 챗봇 웹페이지 실행 화면

▼ 새 탭 x | Home x | 05_MongoDB_Clie x | app_db_Mon

← → ↻ ⓘ 127.0.0.1:5000

Chat with AI

Your Question: Ask

Question:

내 컴퓨터가 켜지지 않아?

Answer:

컴퓨터가 켜지지 않는 이유는 여러 가지가 있을 수 있습니다. 다음은 문제를 해결하기 위해 확인하세요. 1. 전원 콘센트와 컴퓨터 본체 모두 확인해 보세요. 2. **전원 버튼 확인**: 전원 버튼을 눌러보세요. 3. **모니터 확인**: 컴퓨터가 켜져 있지만 모니터가 작동하지 않을 수 있습니다. 4. **하드웨어 문제**: 컴퓨터 내부의 하드웨어(예: RAM, 그래픽 카드 등)가 제대로 장착되어 있는지 확인하세요. 5. **온도 문제**: 컴퓨터가 너무 뜨거워져서 자동으로 꺼졌다면, 잠시 기다렸다가 다시 켜보세요. 6. **비프음 확인**: 컴퓨터가 켜질 때 비프음이 나오는지 확인하세요. 7. **배터리 확인**: 노트북의 경우 배터리를 충전하세요.

[View Conversation History](#)

3. MongoDB에 대화 내용 저장하기

MongDB사용 챗봇 웹페이지 실행 화면: 대화 내용 불러오기

← → ↺ ⓘ 127.0.0.1:5000/history

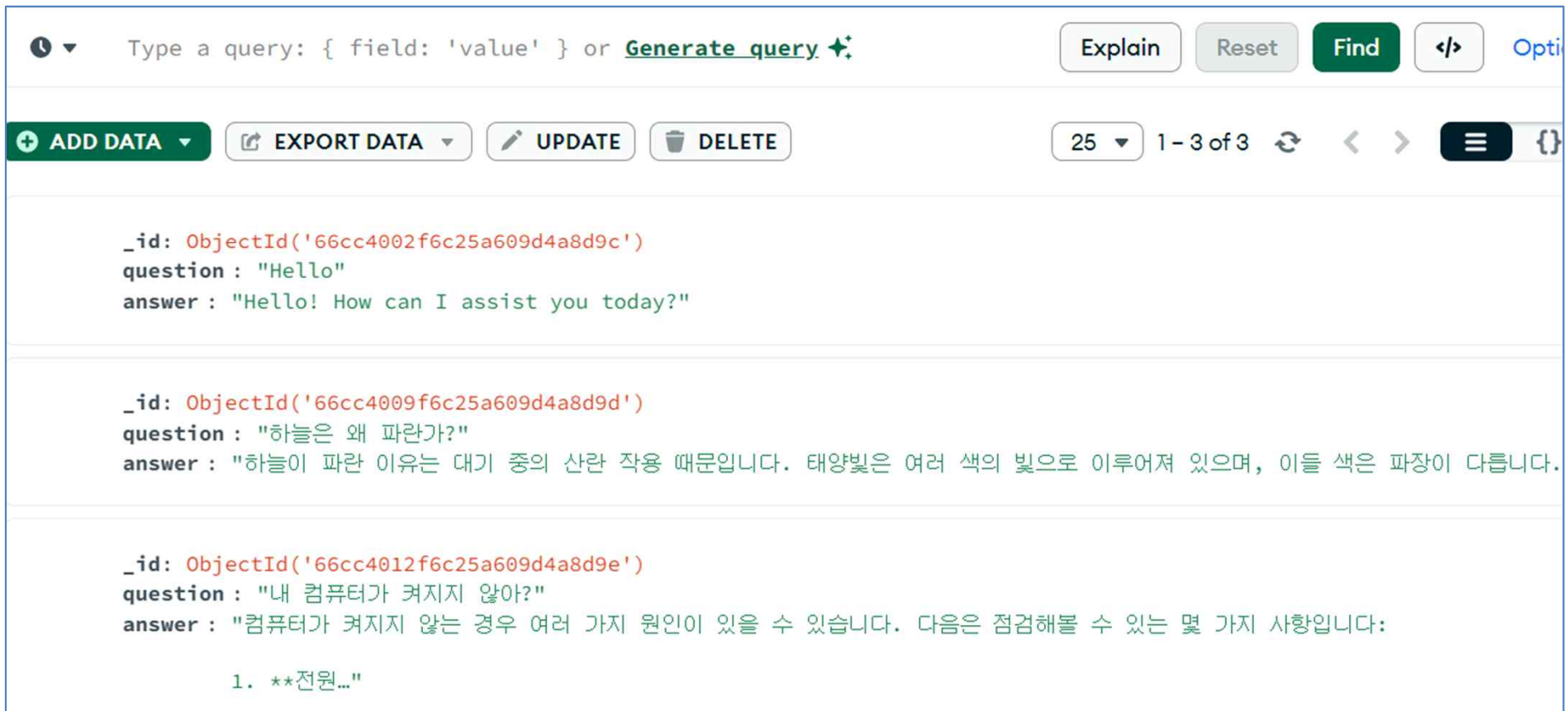
Conversation History

ID	Question	Answer
66cc4002f6c25a609d4a8d9c	Hello	Hello! How can I assist you today?
66cc4009f6c25a609d4a8d9d	하늘은 왜 파란가?	하늘이 파란 이유는 대기 중의 산란 작용 때문입니다. 태양빛은 여러 색의 빛으로 이루어져 있으며, 이들 빛은 파장이 짧고, 빨간색 빛은 파장이 깁니다. 태양빛이 대기에 들어오면, 빛은 대기 중의 분자와 입자에 의해 산란됩니다. 파란색 빛은 긴 파장의 빨간색 빛보다 더 많이 산란되기 때문에, 우리가 하늘을 볼 때 주로 파란색이 보거나 질 때는 태양빛이 대기를 더 많이 통과해야 하므로, 파란색 빛이 많이 산란되고, 빨간색과 주황색 빛도 나타납니다. 이러한 이유로 일출과 일몰 때 하늘이 붉거나 주황색으로 물드는 것입니다.
66cc4012f6c25a609d4a8d9e	내 컴퓨터가 켜지지 않아?	컴퓨터가 켜지지 않는 경우 여러 가지 원인이 있을 수 있습니다. 다음은 점검해볼 수 있는 몇 가지 사항입니다. 1. 전원 케이블이 제대로 연결되어 있는지, 전원 콘센트에 전기가 공급되고 있는지 확인하세요. 다른 전자제품으로 전원 케이블이 제대로 연결되어 있는지 확인하세요. 2. **전원 버튼**: 전원 버튼을 제대로 눌렀는지 확인하세요. 일회성 버튼을 길게 눌러야 켜지기도 합니다. 3. **모니터 확인**: 컴퓨터가 켜져 있지만 모니터가 켜지지 않는 경우 모니터의 전원 및 연결 상태를 확인하세요. 4. **하드웨어 문제**: 내부 하드웨어 문제(예: RAM, 그래픽 카드, 전원 공급 장치)가 부팅되지 않을 수 있습니다. 하드웨어를 점검하고, 필요시 재장착해 보세요. 5. **비프음 확인**: 부팅 시 비프음 소리에 따라 문제의 원인을 알 수 있습니다. 비프음의 패턴을 참고하여 문제를 진단해 보세요. 6. **배터리

[Back to Chat](#)

3. MongoDB에 대화 내용 저장하기

MongoDB Compass 모니터링 화면



The screenshot displays the MongoDB Compass web interface. At the top, there is a search bar with the text "Type a query: { field: 'value' } or [Generate query](#)". To the right of the search bar are buttons for "Explain", "Reset", "Find", and a code icon. Below the search bar, there is a row of action buttons: "ADD DATA", "EXPORT DATA", "UPDATE", and "DELETE". To the right of these buttons are pagination controls showing "25" items per page and "1 - 3 of 3" results. The main area of the interface displays a list of three chat messages, each with its own _id, question, and answer.

_id	question	answer
ObjectId('66cc4002f6c25a609d4a8d9c')	"Hello"	"Hello! How can I assist you today?"
ObjectId('66cc4009f6c25a609d4a8d9d')	"하늘은 왜 파란가?"	"하늘이 파란 이유는 대기 중의 산란 작용 때문입니다. 태양빛은 여러 색의 빛으로 이루어져 있으며, 이들 색은 파장이 다릅니다."
ObjectId('66cc4012f6c25a609d4a8d9e')	"내 컴퓨터가 켜지지 않아?"	"컴퓨터가 켜지지 않는 경우 여러 가지 원인이 있을 수 있습니다. 다음은 점검해볼 수 있는 몇 가지 사항입니다: 1. **전원..."

3. MongoDB에 대화 내용 저장하기

DB조회 시 반환 값이 리스트가 아닌 dict형이므로 history.html 파일을 아래와 같이 수정한다

```
<tbody>
  {% for conversation in conversations %}
  <tr>
    <td>{{ conversation[0] }}</td>
    <td>{{ conversation[1] }}</td>
    <td>{{ conversation[2] }}</td>
  </tr>
  {% endfor %}
```



```
{% for conversation in conversations %}
<tr>
  <td>{{ conversation['_id'] }}</td>
  <td>{{ conversation['question'] }}</td>
  <td>{{ conversation['answer'] }}</td>
</tr>
```

감사합니다