

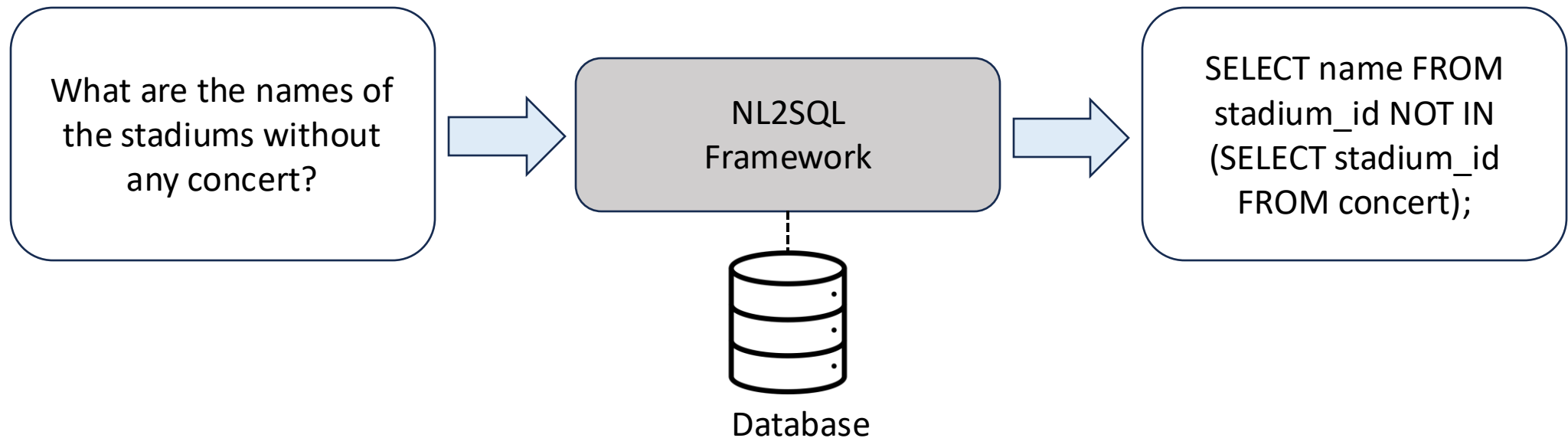
NL2SQL

VLDB Lab.

Professor Sang-Won Lee

What is NL2SQL?

- 자연어 질문을 SQL 쿼리로 변환하는 작업 (Text-to-SQL이라고도 함)
- SQL문법이 익숙하지 않아도 데이터에 쉽게 접근할 수 있도록 하는 기능.



Challenges of NL2SQL

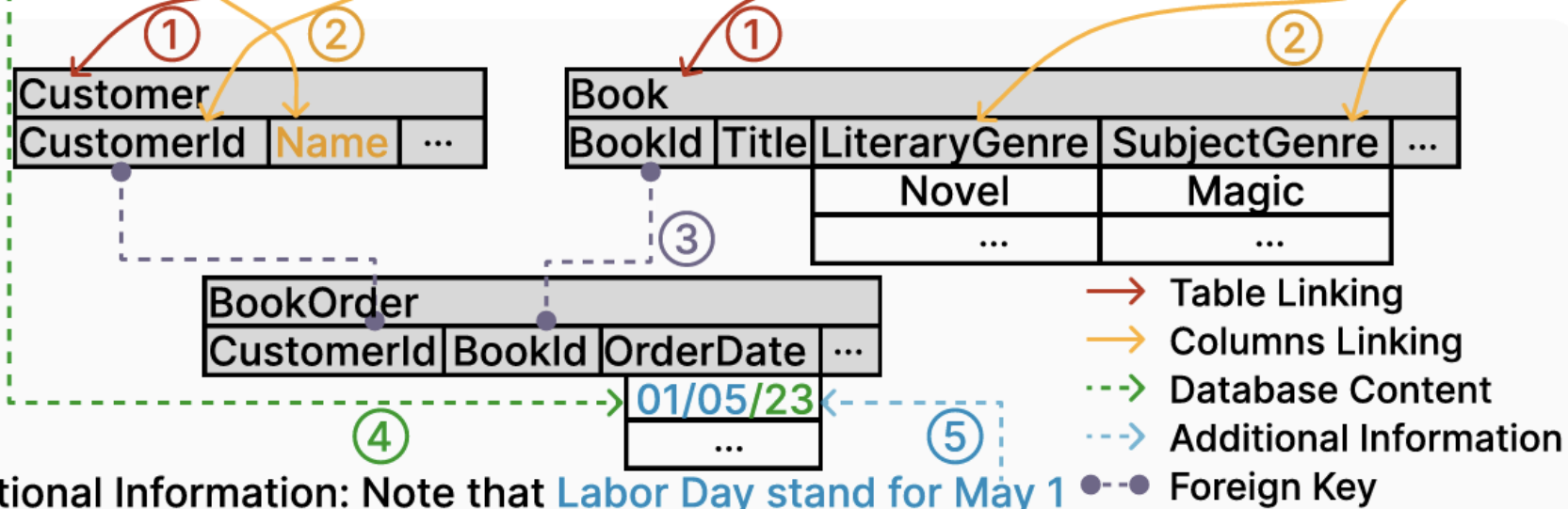
1. **불확실한 자연어 질의:** 사용자가 입력하는 자연어 질의는 종종 모호하고 불완전하여, 시스템이 의도한 의미를 정확히 파악하기 어려움.
2. **복잡한 데이터베이스와 Dirty content:** 실제 데이터베이스는 테이블간 관계가 복잡하며, 컬럼명이나 값이 일관되지 않거나 오류가 있는 경우도 많아 질의 생성에 어려움을 줌.
3. **자연어에서 SQL로의 변환:** 자유로운 자연어 vs. 제약된 형식의 SQL
4. **NL2SQL 시스템 개발의 기술적인 이슈:** 비용 효율적인 솔루션, 모델 효율성 등 고려 필요

Challenges of NL2SQL

NL Query:

Find the **names** of all **customers** who checked out **books** on exactly 3 different **genres** on **Labor Day in 2023**.

Database:



Additional Information: Note that **Labor Day** stand for **May 1**

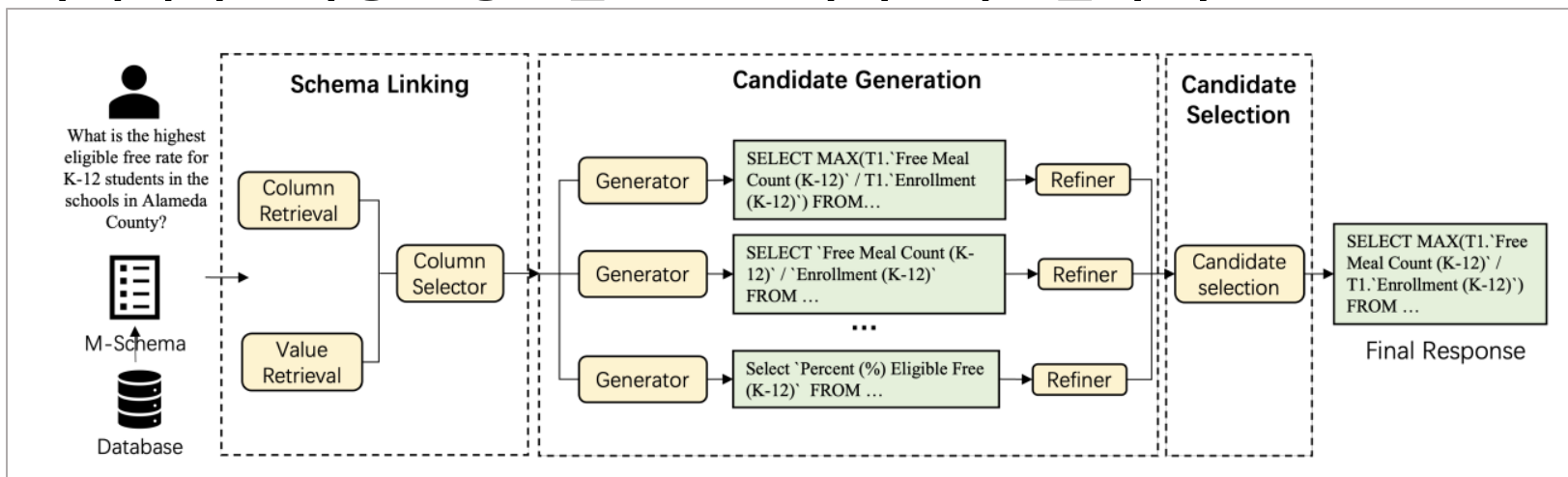
SQL:

```
SELECT Name
FROM Customer
NATURAL JOIN BookOrder
NATURAL JOIN Book
WHERE OrderDate='01/05/23'
GROUP BY CustomerId, Name
HAVING COUNT(DISTINCT SubjectGenre)=3
```

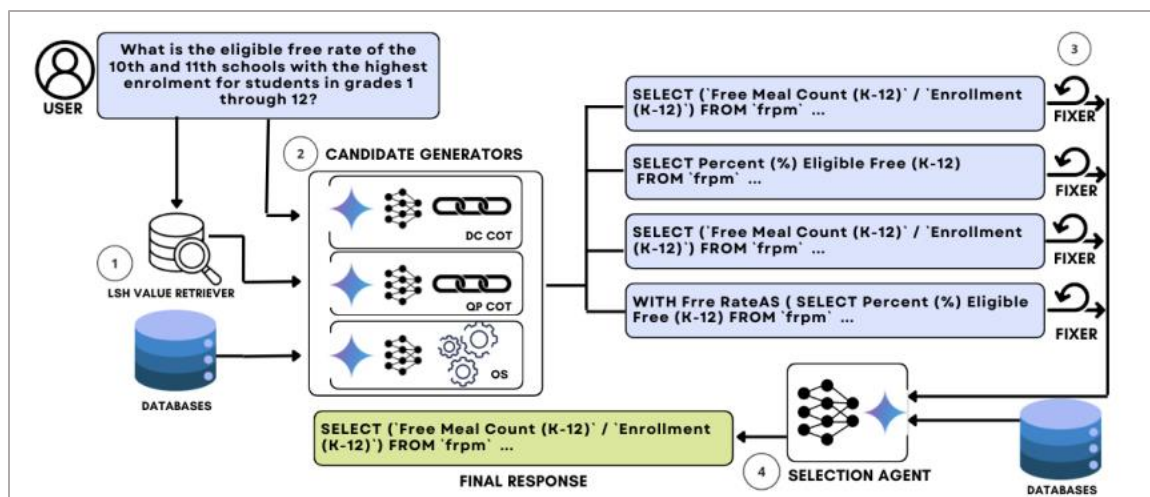
```
SELECT Name
FROM Customer
WHERE CustomerId = (SELECT CustomerId
FROM BookOrder NATURAL JOIN Book
WHERE OrderDate='01/05/23'
GROUP BY CustomerId
HAVING COUNT(DISTINCT SubjectGenre)=3)
```

Current NL2SQL Frameworks

- 학계에서는 다양한 방법을 도입한 파이프라인을 구축



XiYan-SQL: A Multi-Generator Ensemble Framework for Text-to-SQL



CHASE-SQL: Multi-Path Reasoning and Preference Optimized Candidate Selection in Text-to-SQL

NL2SQL Platform

- 최근에는 NL2SQL을 지원하는 많은 플랫폼들이 생겨남.
 - <https://www.eversql.com/text-to-sql/>
 - <https://www.text2sql.ai>
 - <https://www.sqlai.ai/text-to-sql-ai>
- Oracle같은 DBMS에서도 NL2SQL을 개발중임.
 - <https://blogs.oracle.com/machinelearning/post/introducing-natural-language-to-sql-generation-on-autonomous-database>
- Snowflake는 스키마를 사람이 읽고 해석하기 쉬운 비즈니스 단위로 스키마 객체를 추상화
 - <https://docs.snowflake.com/en/user-guide/views-semantic/overview>

NL2SQL Practice

- 목적

- 자연어를 SQL 쿼리로 자동 변환하는 NL2SQL 시스템 체험
- ChatGPT를 활용해 자연어 질의 → SQL 생성 → 실행결과 확인까지 수행

- 실습 도구

- ChatGPT
- Google Colab 환경

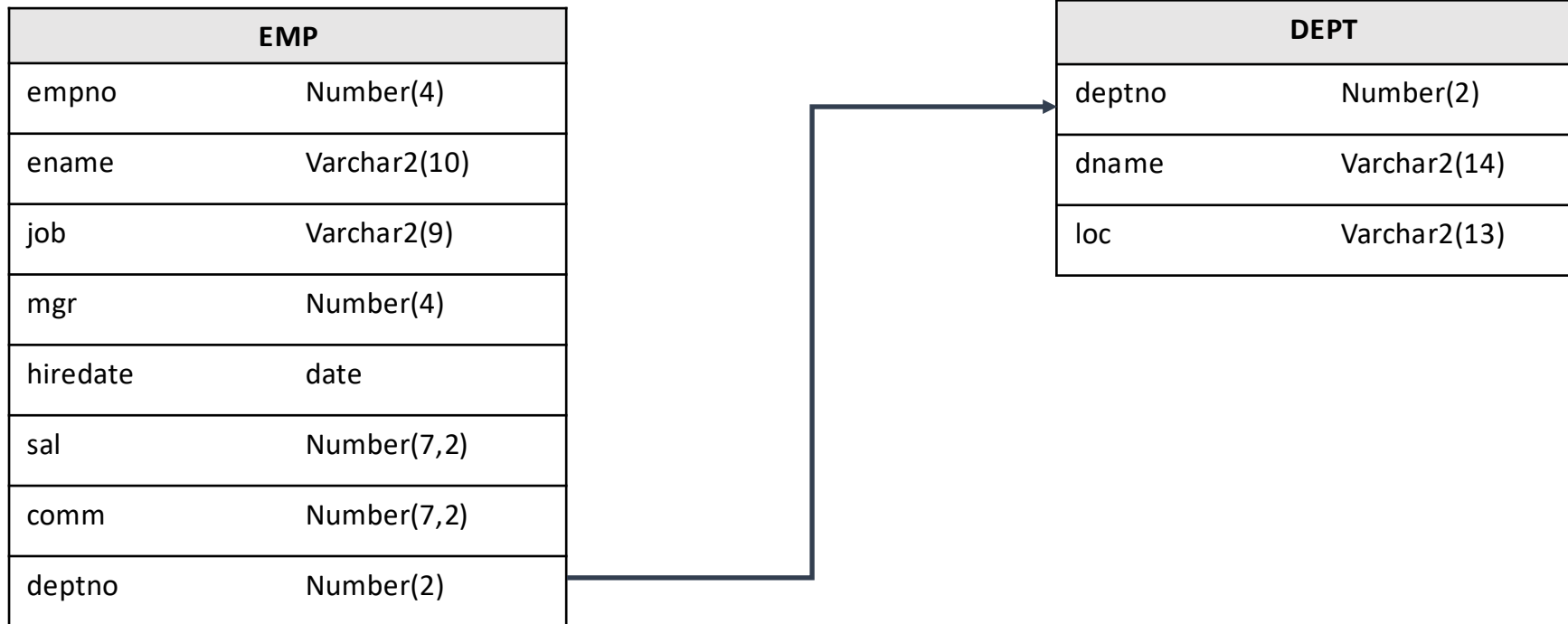
NL2SQL Prompt

- ChatGPT 사용시, ChatGPT가 대답을 잘 생성할 수 있도록 적당한 프롬프트를 넣어주어야 함.
- **프롬프트란?** 누군가(생성형 AI)의 특정한 작업 수행을 도우려 전달하는 메시지.



- 프롬프트를 어떻게 구성하느냐에 따라 다른 결과가 출력될 수 있음.
- NL2SQL에서는 **데이터베이스 스키마와 사용자 질문**을 사용해서 프롬프트를 구성해야함.

NL2SQL Practice (Scott Schema)



NL2SQL Practice (Schema Information)

- 다음과 같은 스키마가 존재한다고 가정할 때, 스키마 정보에 대한 프롬프팅 방법이 여러가지 있을 수 있음.

1.

```
CREATE TABLE DEPT (DEPTNO NUMBER(2) CONSTRAINT PK_DEPT PRIMARY  
KEY, DNAME VARCHAR2(14) , LOC VARCHAR2(13) ) ;
```

```
CREATE TABLE EMP (EMPNO NUMBER(4) CONSTRAINT PK_EMP PRIMARY KEY,  
ENAME VARCHAR2(10), JOB VARCHAR2(9), MGR NUMBER(4), HIREDATE DATE,  
SAL NUMBER(7,2), COMM NUMBER(7,2), DEPTNO NUMBER(2) CONSTRAINT  
FK_DEPTNO REFERENCES DEPT);
```

2.

```
DEPT (DEPTNO, DNAME, LOC)  
EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO)
```

NL2SQL Practice (NLQ)

- 자연어 질문 예시

1. Display all employee whose location is DALLAS?
2. Display all the departments where department has 3 employees?
3. Delete all employees those who are reporting to BLAKE?
4. Display average salary for job SALESMAN
5. Display all ename, empno, dname, loc from emp, dept table without joining two tables?

NL2SQL Practice (Final Prompt)

- 최종 프롬프트는 다양하게 생성 가능.

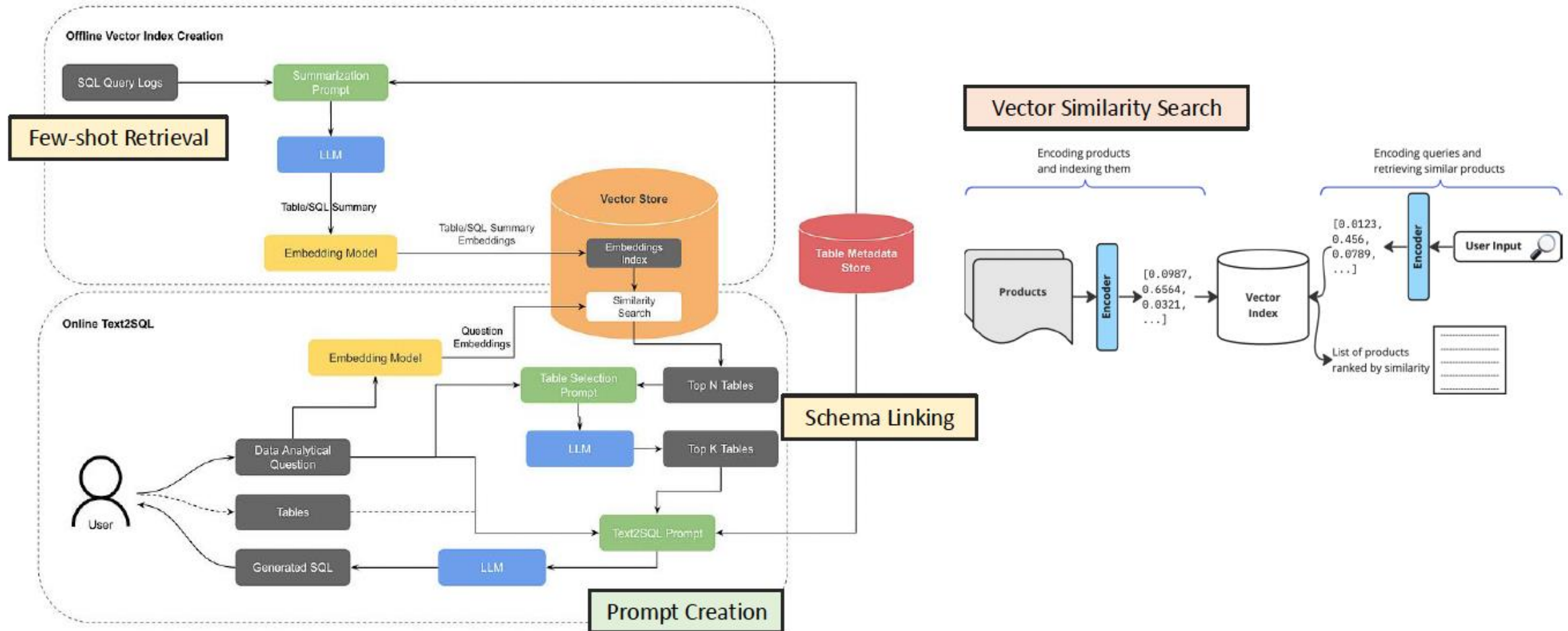
1.

```
### Create the Postgres SQL Query using the following schema:  
# CREATE TABLE DEPT (DEPTNO NUMBER(2) CONSTRAINT PK_DEPT PRIMARY  
KEY, DNAME VARCHAR2(14) , LOC VARCHAR2(13) ) ;  
# CREATE TABLE EMP (EMPNO NUMBER(4) CONSTRAINT PK_EMP PRIMARY  
KEY, ENAME VARCHAR2(10), JOB VARCHAR2(9), MGR NUMBER(4), HIREDATE  
DATE, SAL NUMBER(7,2), COMM NUMBER(7,2), DEPTNO NUMBER(2)  
CONSTRAINT FK_DEPTNO REFERENCES DEPT);  
### Display all employee whose location is DALLAS?
```

2.

```
### Create the Postgres SQL Query using the following schema:  
### [Schema]  
# DEPT (DEPTNO, DNAME, LOC)  
# EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO)  
# EMP.DEPTNO references DEPT.DEPTNO  
### [NLQ]  
### Display all employee whose location is DALLAS.
```

Providing Additional Contexts for NL2SQL



<https://medium.com/pinterest-engineering/how-we-built-text-to-sql-at-pinterest-30bad30dabff>

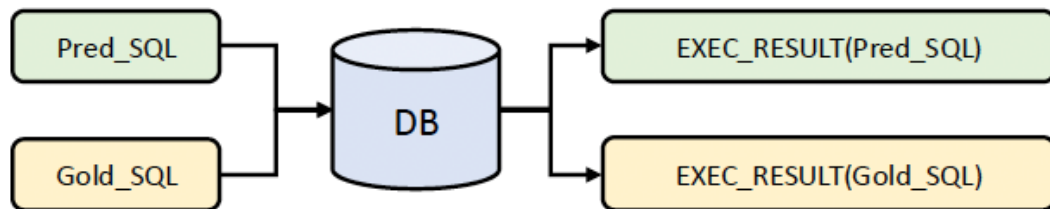
Bird-SQL

- Bird Benchmark

- 대규모 데이터베이스 기반 NL2SQL 평가용 벤치마크
- 총 12,751개의 고유한 질의-SQL쌍으로 구성됨
- 총 95개의 대형 데이터베이스, 37개 이상의 도메인 포함

- 성능 평가 지표: 실행 정확도 (Execution Accuracy, EX, %)

- 예측된 SQL문을 실행한 결과와 정답 SQL 실행 결과가 일치하는지 평가



Leaderboard - Execution Accuracy (EX)

	Model	Code	Size	Oracle Knowledge	Dev (%)	Test (%)
Human Performance <i>Data Engineers + DB Students</i>				✓		92.96
1 Mar 11, 2025	AskData + GPT-4o <i>AT&T CDO - DSAIR</i>		UNK	✓	75.36	77.14
2 Feb 27, 2025	Contextual-SQL <i>Contextual AI</i>		UNK	✓	73.50	75.63
3 Dec 17, 2024	XiYan-SQL <i>Alibaba Cloud</i> [Yifu Liu et al. '24]	[link]	UNK	✓	73.34	75.63
4 Nov 24, 2024	CHASE-SQL + Gemini <i>Google Cloud</i> [Pourreza et al. '24]		UNK	✓	74.46	74.79
5 Oct 27, 2024	ExSL + granite-34b-code <i>IBM Research AI</i>		34B	✓	72.43	73.17
6 Aug 21, 2024	OpenSearch-SQL _{v2} + GPT-4o <i>Alibaba Cloud</i> [Xiangjin Xie et al. '25]	[link]	UNK	✓	69.30	72.28
7 Jul 22, 2024	Distillery + GPT-4o <i>Distyl AI Research</i> [Maamari et al. '24]		UNK	✓	67.21	71.83
8 May 21, 2024	CHESS _{IR} + CG + UT <i>Stanford</i> [Taleai et al. '24]	[link]	UNK	✓	68.31	71.10

Superhero Database (BIRD-benchmark)

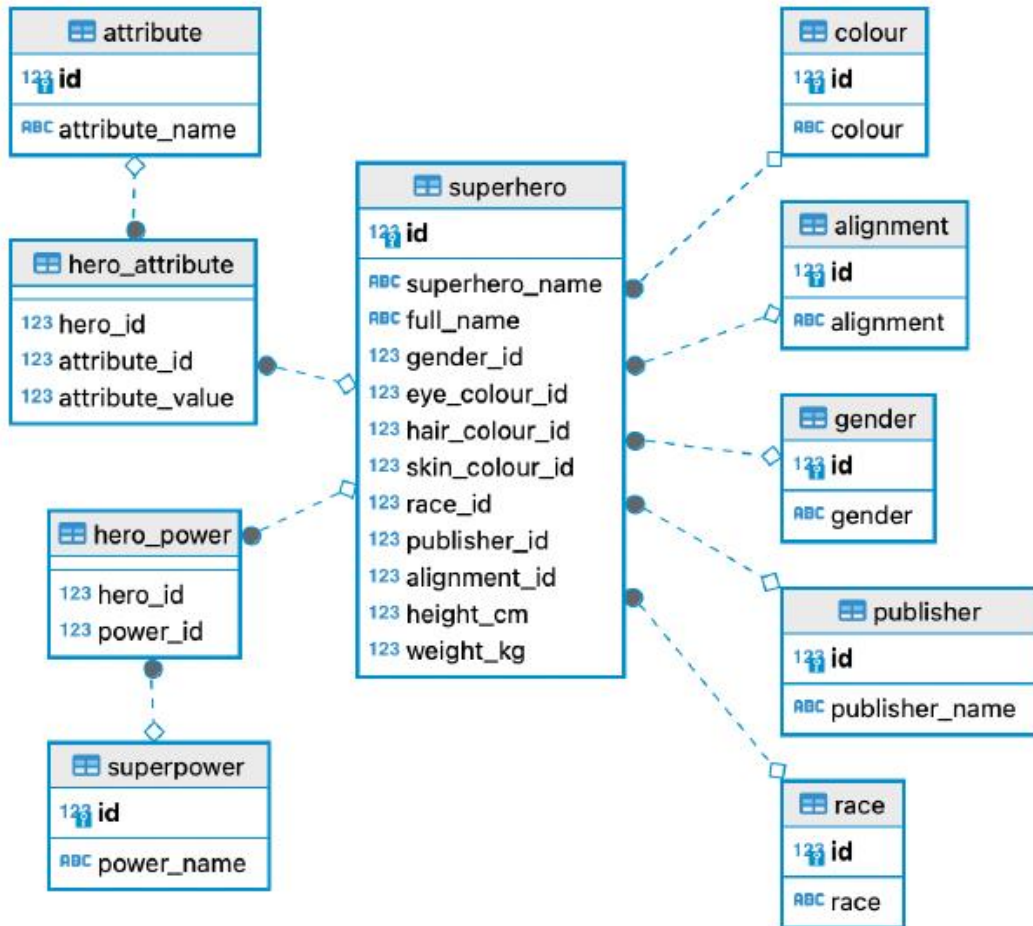
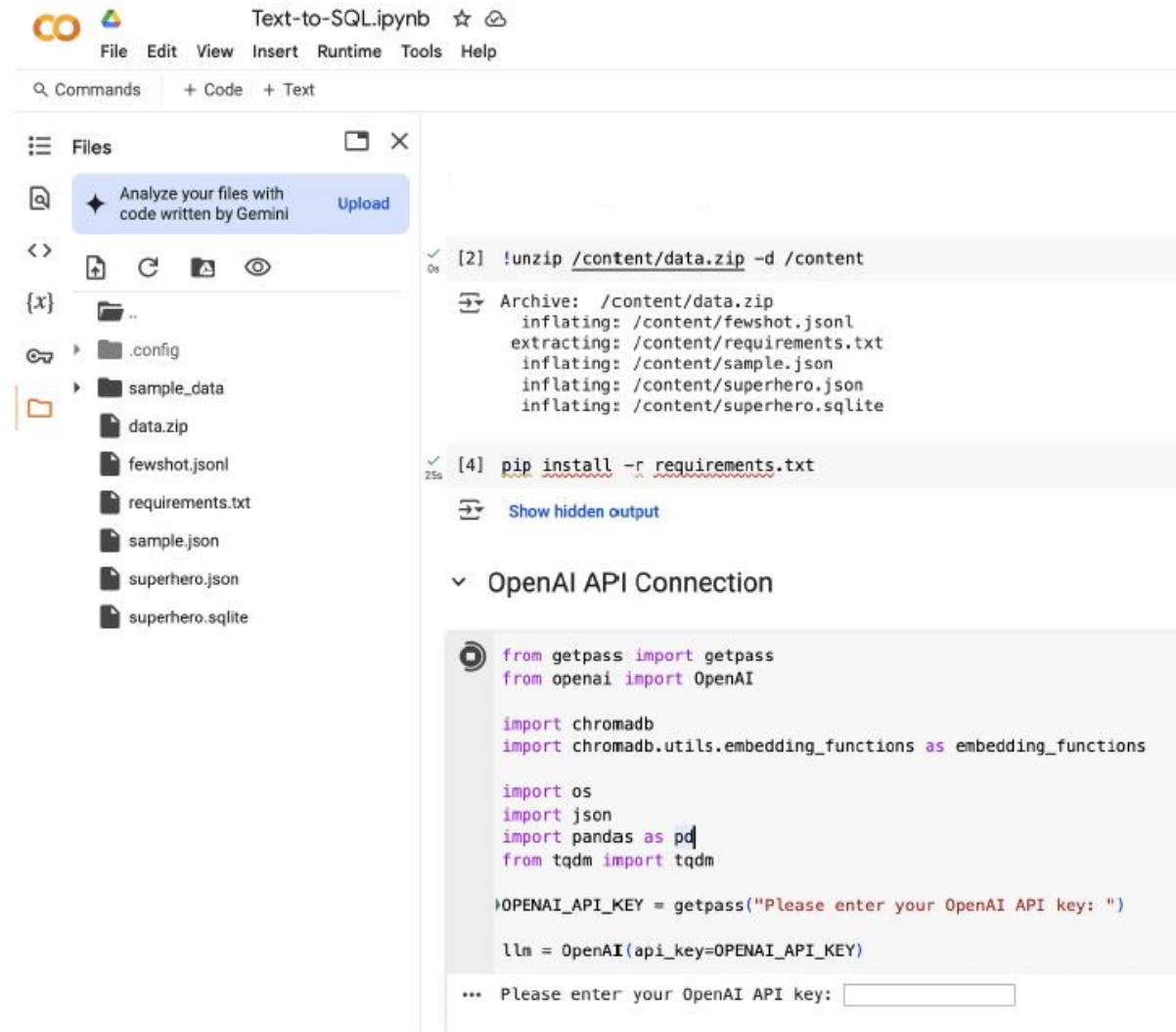


Table: superhero

	123 id	ABC superhero_na	ABC full_name	123 gender_id	123 eye_colour_id
1	1	3-D Man	Charles Chandler	1	9
2	2	A-Bomb	Richard Milhouse Jones	1	33
3	3	Abe Sapien	Abraham Sapien	1	7
4	4	Abin Sur	-	1	7
5	5	Abomination	Emil Blonsky	1	14
6	6	Abraxas	Abraxas	1	7
7	7	Absorbing Man	[NULL]	1	7
8	8	Adam Monroe	-	1	7
9	9	Adam Strange	Adam Strange	1	7
10	10	Agent 13	Sharon Carter	2	7
11	11	Agent Bob	Bob	1	9
12	12	Agent Zero	Christoph Nord	1	1
13	13	Air-Walker	Gabriel Lan	1	7
14	14	Ajax	[NULL]	1	9
15	15	Alan Scott	[NULL]	1	7
16	16	Alex Mercer	Alexander J. Mercer	1	1

Preparation

- Google Colab
- Data.zip
 - requirements.txt
 - superhero.sqlite
 - superhero.json
 - fewshot.jsonl



The screenshot shows a Google Colab notebook interface. The top bar includes the Colab logo, the notebook title 'Text-to-SQL.ipynb', and a menu with options: File, Edit, View, Insert, Runtime, Tools, and Help. Below the menu is a search bar and tabs for 'Commands', '+ Code', and '+ Text'. The left sidebar displays a file explorer with a folder named 'sample_data' containing files: 'data.zip', 'fewshot.jsonl', 'requirements.txt', 'sample.json', 'superhero.json', and 'superhero.sqlite'. The main area shows two code cells. The first cell, labeled '[2]', contains the command `!unzip /content/data.zip -d /content` and its output, which lists the files being inflated: `/content/fewshot.jsonl`, `/content/requirements.txt`, `/content/sample.json`, `/content/superhero.json`, and `/content/superhero.sqlite`. The second cell, labeled '[4]', contains the command `pip install -r requirements.txt` and a 'Show hidden output' button. Below the code cells is a section titled 'OpenAI API Connection' with a code block for importing libraries and setting the API key. The code includes imports for `getpass`, `openai`, `chromadb`, `os`, `json`, `pandas`, and `tqdm`. It then prompts the user to enter their OpenAI API key using `getpass` and initializes the `OpenAI` client. At the bottom of this section is a text input field with the placeholder text 'Please enter your OpenAI API key:'.

```
from getpass import getpass
from openai import OpenAI

import chromadb
import chromadb.utils.embedding_functions as embedding_functions

import os
import json
import pandas as pd
from tqdm import tqdm

OPENAI_API_KEY = getpass("Please enter your OpenAI API key: ")

llm = OpenAI(api_key=OPENAI_API_KEY)
```

... Please enter your OpenAI API key: