Text-to-SQL을 활용한 관계형 데이터베이스 연동 LLM 기반 질의응답 시스템

An LLM-based Question Answering System over Relational Databases via Text-to-SQL

2025.11.29. (토)
차세대융합기술연구원

정솔[1], 김민식[1], 강명구[2], 정재윤[1,2,3,*]

산업AI연구실
[1]경희대학교 빅데이터응용학과, [2]경희대학교 인공지능학과,
[3]경희대학교 산업경영공학과

목차

1. 서론

연구 배경

- **제조실행시스템 (Manufacturing Execution System: MES)에서는 실시간으로 생산 공정을 모니터링하며 방대한 양의 데이터 축적**
- **하지만 현장의 작업자들은 구조적 질의어(Structured Query Language: SQL)에 지식 부족으로 데이터에 접근이 어려움**
- **사람의 자연어 질의를 SQL로 바꿔주는 Text-to-SQL (YU et al., 2018; ZHONG et al., 2017) 기술이 꾸준히 연구되고 있으나 다음과 같은 상황에서의 질의 생성 능력 저하**
  - 복잡한 데이터베이스 스키마 (Mitsopoulou & Koutrika, 2025)
  - 불명확한 테이블이나 컬럼 이름 (BHASKAR et al., 2023)
- **이러한 한계를 해결하기 위해 데이터베이스(Database: DB) 구조 단순화와 자연어 답변 생성을 결합한 접근 활용**

연구 목표

- **데이터 마트(Data Mart: DM) 설계(Kimball & Ross, 2013)와 테이블 증강 생성(Table-Augmented Generation: TAG) (BISWAL et al., 2024)을 통한 자연어 질의 시스템 구축**
  - 관계형 데이터베이스(Relational Database: RDB)를 스타 스키마(Star Schema) 기반의 DM 변환 → **언어 모델의 DB 스키마 이해도 개선**
  - TAG 기반 자연어 질의 파이프라인(Pipeline) 구축 → **SQL 실행 결과에 대한 사용자의 이해 개선**

기대 효과

- **이력 데이터 분석을 위한 DM 설계를 통해 사용자 친화적 TAG 결과 생성**
- **현장 실무자의 자연어를 통해 이력 데이터에 대한 접근성 향상 및 신속한 의사 결정 지원**

2. 관련 연구

Text-to-SQL

- **Text-to-SQL: 사람의 자연어 질의를 SQL로 바꿔주는 기술**
  예: "우리 회사에서 가장 많이 팔린 제품 3개를 보여줘."

```sql
SELECT
    product_name,
    total_quantity
FROM
    product_sales
ORDER BY
    total_quantity DESC
LIMIT 3;
```

예시 질문에 대해 생성된 SQL

- **대규모 언어 모델(Large Language Model: LLM)의 등장으로 활발히 연구**
  - **현대 벤치마크:** "Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task" (YU, Tao, et al., 2018)
    → 200개 이상의 다양한 도메인과 JOIN, 서브쿼리(Subquery)가 포함된 복잡한 SQL에 대한 데이터셋을 발표한 논문
  - **State-of-the-art:** "MAC-SQL: Multi-Agent Collaboration for Text-to-SQL" (Wang, Chen et al., 2024)
    → 다중 에이전트 시스템(Selector-Decomposer-Refiner)을 활용하여 복잡한 질문을 분해하고, 생성된 SQL을 **실제 실행 환경에서 검증 및 자체 수정(Self-Correction)**

- **연구의 한계 (SHI et al., 2025)**
  - **복잡하고 비효율적인 쿼리 생성:** 다중 조인 및 중첩 쿼리 등 복잡한 SQL 구문 생성 시 논리적 오류 발생 위험
  - **비즈니스 맥락 부족 및 모호성:** 스키마 이름과 실제 비즈니스 의미 간의 차이로 인한 정확도 저하 (예: 'product_id2' 해석의 어려움)
  - **컨텍스트 창 한계:** 대규모 엔터프라이즈 데이터베이스의 방대한 스키마를 LLM의 입력 프롬프트에 모두 담기 어려운 기술적 제약

2. 관련 연구

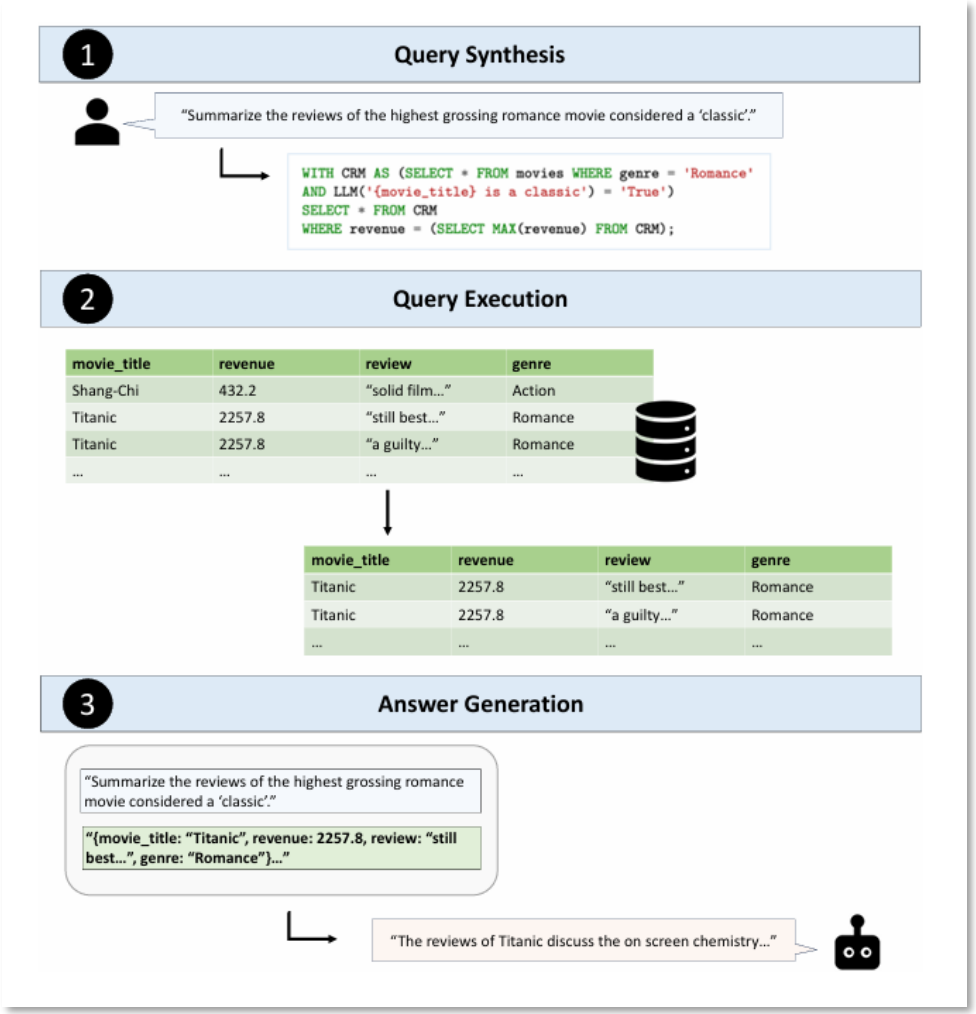
테이블 증강 생성(Table-Augmented Generation: TAG)

- **기존 Text-to-SQL 연구가 테이블 조작(관계 연산, Relation Algebra: RA) 수준에 머문 한계를 지적하며, 실제 사용자 질문은 의미 추론·범용 지식까지 요구함을 강조**

- **해결책으로 "Table-Augmented Generation (TAG)"이라는 패러다임을 제시**

- **사용자가 자연어 질의에 대해 "(1)질의 생성 → (2)질의 실행 → (3)답변 생성" 이라는 3단계 파이프라인을 통해 언어 모델(Language Model: LM)의 지식·추론 능력과 DB 시스템의 계산·집계 능력을 결합을 제시**

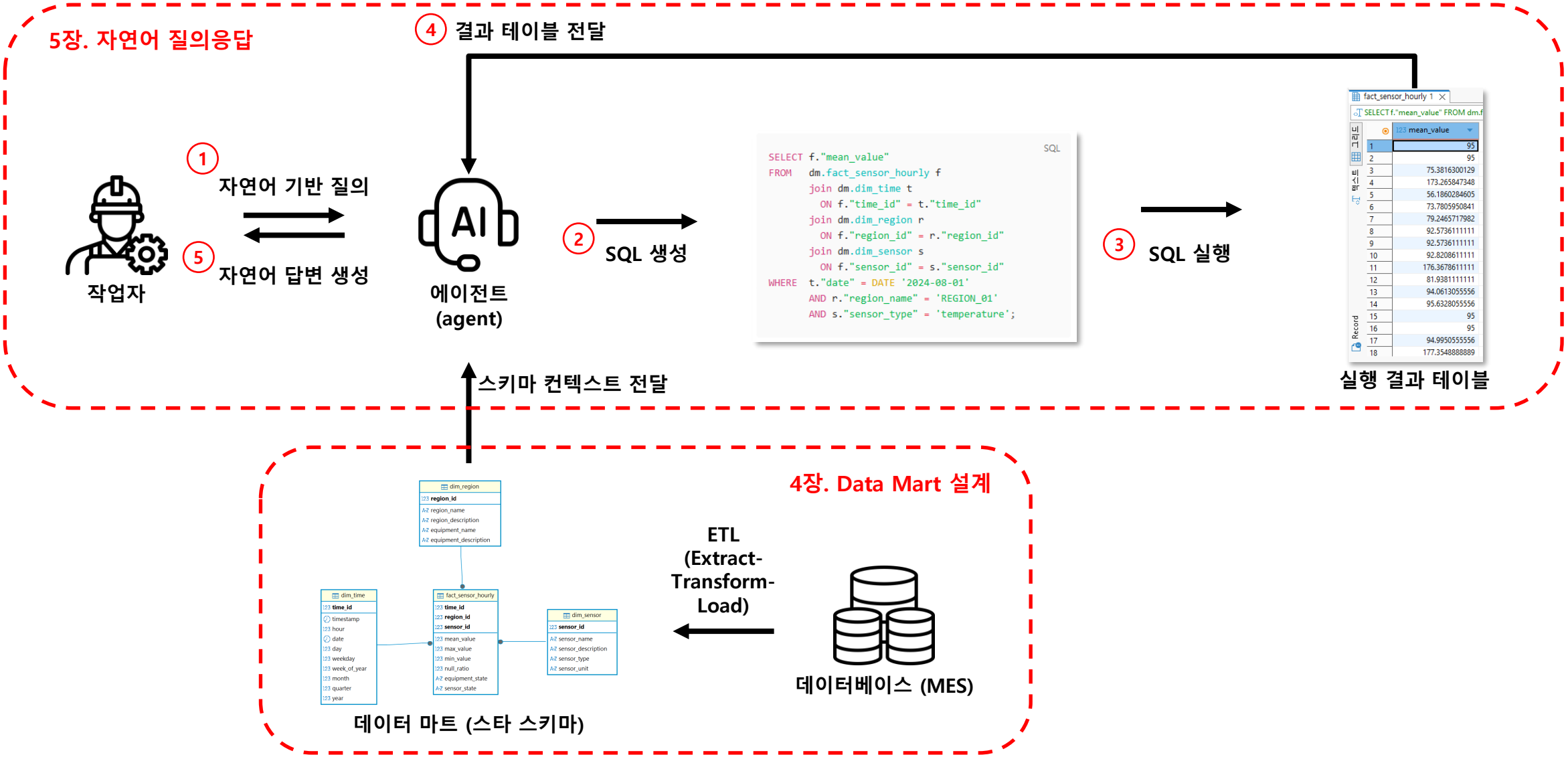
- **세부 진행 단계**
  - **질의 생성(Query Synthesis):** 사용자의 자연어 요청(R)을 실행 가능한 질의(Q)로 변환하는 단계
    예: "지난 달 A라인 불량률은?" → SELECT 문 생성
  - **질의 실행(Query Execution):** 생성된 질의(Q)를 DB 시스템에 실행해 관련 데이터(T)를 얻는 단계. DB 엔진의 효율적 연산 활용 (집계, 필터링, 조인 등).
  - **답변 생성(Answer Generation):** 자연어 요청(R) + 실행 결과(T)를 이용해 언어 모델이 최종 자연어 답변(A)를 생성 (요약, 분석, 인사이트 제공)



TAG (Biswal et al., 2024) 실행 예시

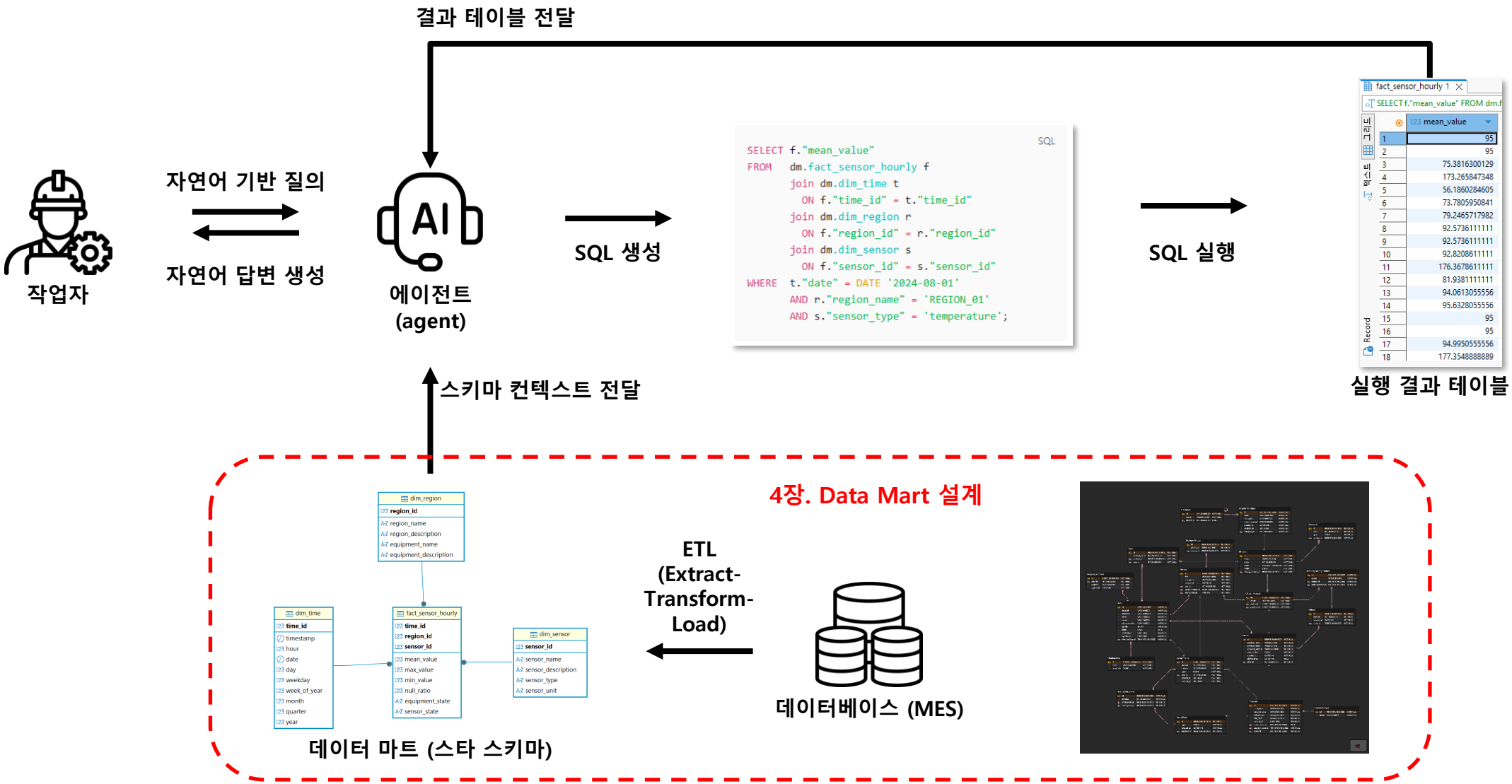
3. 프레임워크(Framework)

- 언어 모델을 이용한 TAG 기반 자연어 질의 응답 시스템



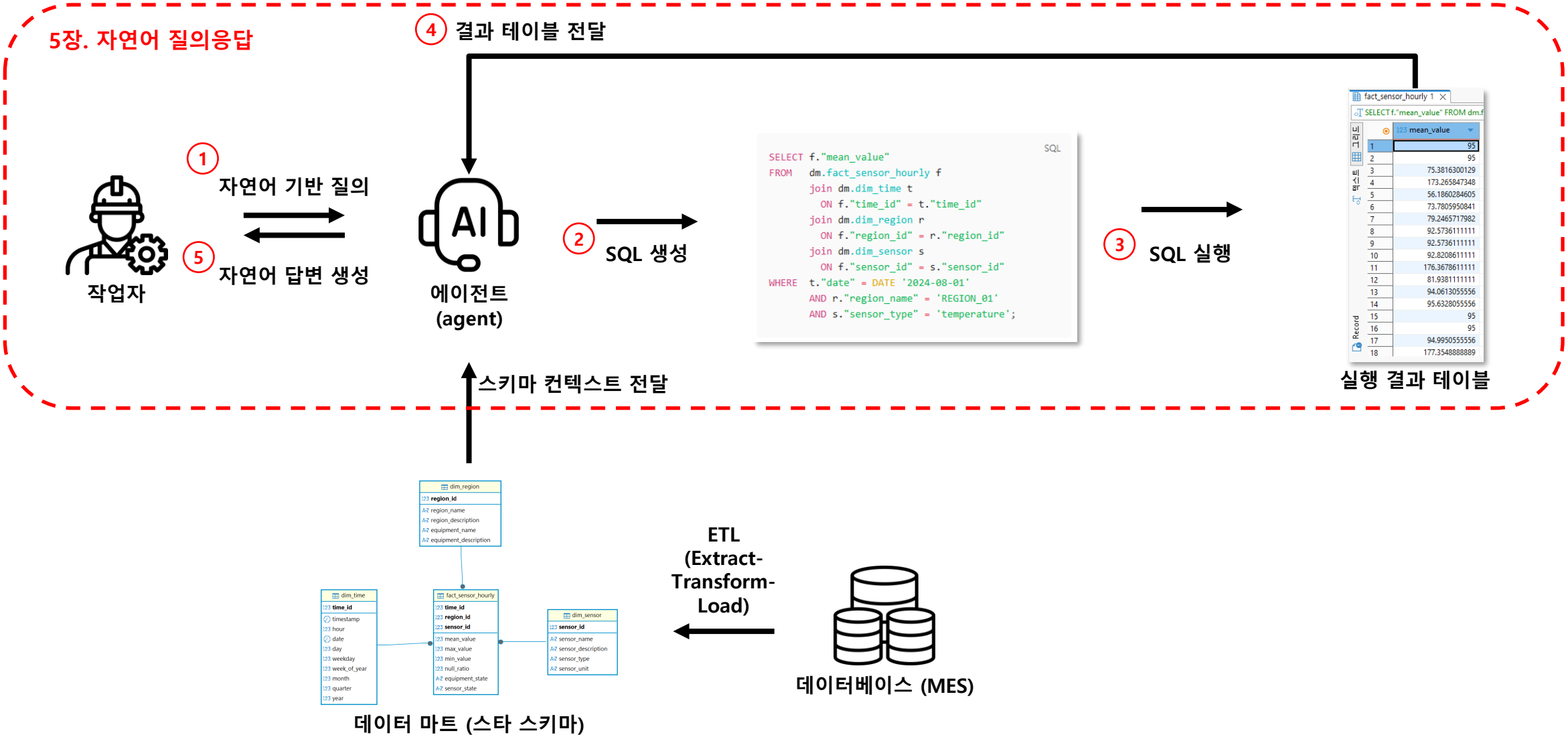
3. 프레임워크(Framework)

- 언어 모델을 이용한 TAG 기반 자연어 질의 응답 시스템



3. 프레임워크(Framework)

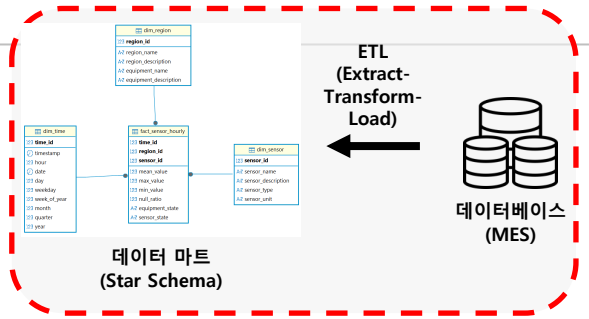
- 언어 모델을 이용한 TAG 기반 자연어 질의 응답 시스템



4. TAG를 위한 데이터 마트 설계 및 구현

데이터셋(Dataset)

- **협업 기업의 MES에서 초(second) 단위로 수집된, 설비의 센서 데이터**
- **데이터 수집 기간 : 2024. 08. 01. – 2024. 08. 09. (주말을 제외한 7일)**
- **수집된 csv에 존재하는 약 2,100개의 컬럼 중, 분석 및 사용자 질의에 주로 사용될 컬럼 77개 선정**
  - 시간 timestamp 1개 + 설비에 위치한 센서가 수집한 온도, 습도, 압력 등에 대한 컬럼 76개



날짜	요일	시작 시간	종료 시간	수집 시간	레코드 수	컬럼 수
08월 01일	목	8:46:45	17:20:08	8시간 33분	30,783	2,181
08월 02일	금	9:21:01	17:42:05	8시간 21분	30,066	2,181
08월 05일	월	11:13:12	16:27:07	5시간 14분	4,598	2,188
08월 06일	화	14:22:18	17:49:59	3시간 28분	12,464	2,188
08월 07일	수	14:24:38	18:01:18	3시간 37분	13,001	2,188
08월 08일	목	10:18:36	18:14:11	7시간 56분	28,490	2,188
08월 09일	금	8:20:30	16:20:01	7시간 59분	28,773	2,188
				합계 레코드 수	**148,175**	

실험에 사용된 데이터셋의 통계

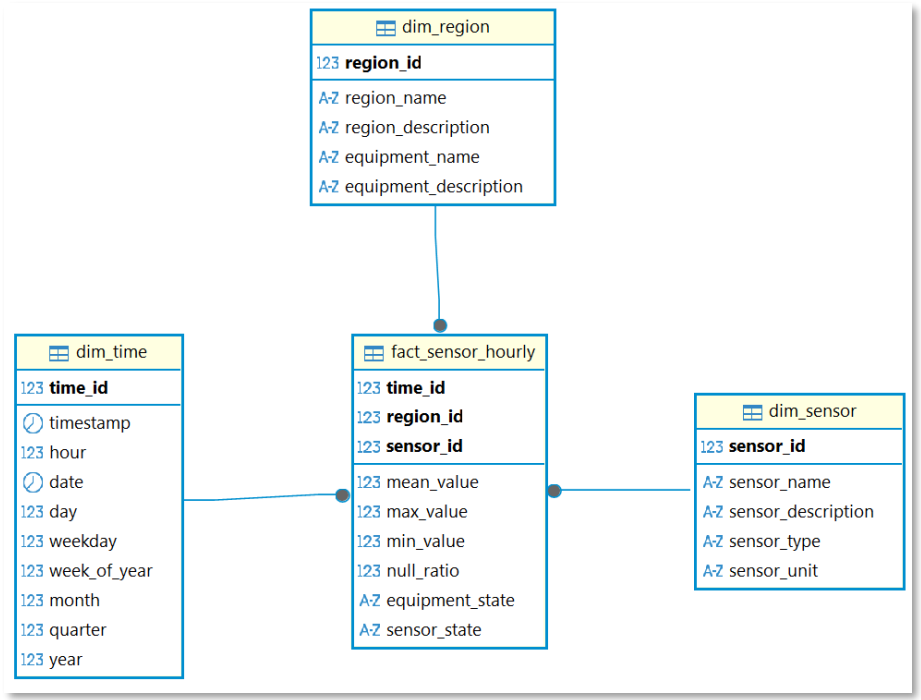
4. TAG를 위한 데이터 마트 설계 및 구현

ETL(Extract-Transform-Load)

- 원본 MES 데이터는 초 단위로 수집
- 데이터 마트는 운영이 아닌 분석 목적의 저장소이기 때문에 시간 해상도를 1시간 단위 집계로 정의
  - OLTP (OnLine **Transaction** Processing) **X**
    OLAP (OnLine **Analytical** Processing) **O**

데이터 마트 설계

- 데이터베이스에 대한 LLM의 이해 부담을 줄이기 위해 스타 스키마 기반의 DM 설계
  - **Fact 테이블:** 측정값과 같은 분석 대상의 수치 데이터를 저장하는 테이블
  - **Dimension 테이블:** Fact를 설명하기 위한 맥락 정보를 저장하는 테이블
- 하나의 fact 테이블과 세 개의 dimension(time, region, sensor)으로 구성
- fact 테이블은 시간(time), 공정 구역(region), 센서(sensor)를 외래 키(foreign key: FK)로 설정하여 dimension과 연결
- 1시간 단위로 집계된 fact 테이블은 (특정 시간의) 해당 구역(region)의 센서(sensor) 값의 평균, 최대, 최소값 등을 표시



스타 스키마 기반으로 설계된 데이터 마트

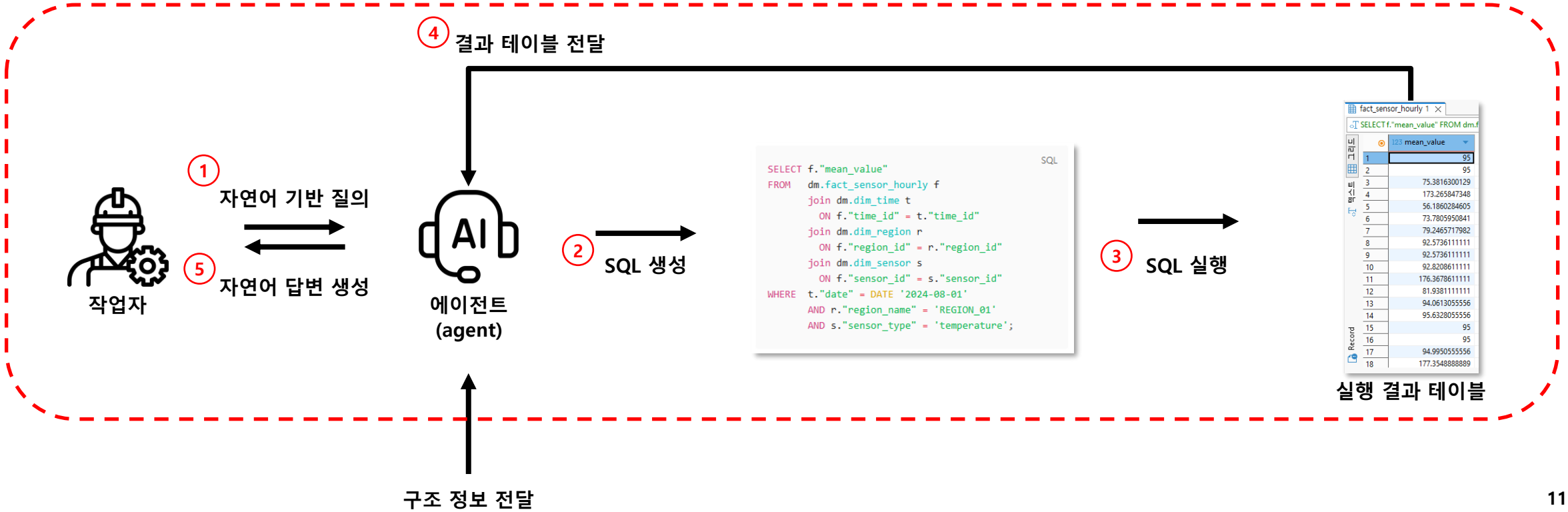
기대 효과

- DM을 통해 테이블을 역할 별로 명확히 분리하여 LLM이 SQL을 안정적으로 생성
- 자연어 질의가 시간, 지역, 센서 조건으로 명확하게 연결되며, TAG의 질의 생성 및 안정성 실행

5. LLM 기반 질의응답 시스템 구현

실행 단계

1. **자연어 기반 질의:** 사용자 질문과 스키마 컨텍스트를 LLM에게 전달
2. **SQL 생성:** 스키마와 사용자의 질문을 분석해 SQL 생성
3. **SQL 실행:** DB에서 대해 SQL 실행 후 결과 반환
4. **결과 전달:** 결과 테이블을 다시 에이전트에게 전달
5. **답변 생성:** 사용자 질의와 결과 테이블 활용해 사용자에게 자연어 답변 제공



6. 실험

실험 설계

- **비교 대상**
  - 스타 스키마 기반으로 설계된 데이터 마트 (제안 방법론, Proposed Method)
  - 77개 컬럼으로 구성된 wide 형태의 테이블 (비교 대상, baseline)

- **평가 모델**
  - GPT-4.1-mini (대형 모델) (OpenAI API)
  - Gemma3-12B (소형 모델) (llama.cpp, 원격 GPU)
  - Qwen3-4B (소형 모델) (vLLM, 원격 GPU)

  - MES 현장의 온프레미스 배포 환경을 가정하여 로컬 실행 가능한 소형 모델(SLM) 포함
  - Max context 8,192로 설정

- **테스트 질문**
  - 4개의 난이도로 구성된 20개의 질문 *(기본 조회, 임계값 필터링, 시간 기반 집계, 고급 분석 (WITH, WINDOW 함수))*
  - 3개 모델 × 2개 스키마 = 총 120회 실행

- **평가 지표**
  - SQL 실행 성공률
  - 실행 시간
  - 토큰 사용량

- **실험 환경:** GPU: NVIDIA TITAN RTX (24GB VRAM × 2)
  관계형 데이터베이스(Relational Database: RDB): PostgreSQL 16.3 (오픈 소스, Open Source)

6. 실험

실험 결과

- **SQL 실행 성공률:** 데이터 마트 설계에서 원본 스키마 대비 31.7%p 더 높은 SQL 실행 성공률
- **실행 시간:** 4.33초 감소
- **토큰 사용량:** 42.6% 절감

평가 지표	DM	원본 스키마	비교
SQL 실행 성공률(↑)	**76.7% (46/60)**	45.0% (27/60)	+31.7%p
평균 실행 시간(↓)	**6.01s**	10.34s	41.9% 감소
평균 토큰 사용량(↓)	**2,419**	4,211	42.6% 절감

오류 유형

- 데이터 마트 설계에서 원본 스키마 대비 약 60% 적은 오류

SQL 실행 오류

에러 유형	DM	원본 스키마	비교
따옴표 누락 (PostgreSQL 대소문자)	**1건 (7.1%)**	23건 (67.6%)	22개 감소
스키마 불일치 (컬럼/테이블 이름 오류)	3건 (21.4%)	**2건 (5.9%)**	1개 증가
SQL 문법 오류	8건 (57.1%)	8건 (23.5%)	동일
최대 토큰 수 한계 (8,192)	2건 (14.3%)	**1건 (2.9%)**	1개 증가
총 에러	**14건**	34건	20개 감소

오류 유형별 분석

- **스키마 불일치**
  - **질문:** "Q03. 2024년 8월 5일 / 지역별로 / 각 센서의 측정된 / 최고 온도 값을 알려줘."
  - **오류 발생 모델 및 스키마:** gemma3-12B / DM
  - **생성된 SQL:**

```sql
SELECT
  T1.region_name,
  MAX(T2.mean_value) AS max_temperature
FROM dm.dim_region AS T1
JOIN dm.fact_sensor_hourly AS T2 ON T1.region_id = T2.region_id
JOIN dm.dim_time AS T3 ON T2.time_id = T3.time_id
WHERE
  T3.date = '2024-08-05' AND T2.sensor_type = 'temperature'  -- ❌ 없는 컬럼!
GROUP BY T1.region_name;
```

  - **에러 메시지:** (psycopg2.errors.UndefinedColumn) 오류: t2.sensor_type 칼럼 없음
    HINT: 아마 "t2.sensor_state" 칼럼을 참조하는 것 같습니다.
  - **원인:** DM 스키마의 fact 테이블(fact_sensor_hourly)에 존재하지 않는 컬럼(sensor_type)을 참조 (LLM 환각)

오류 분석

- **문법 오류**
  - **질문:** "Q11. 최근 한 달 동안 / 각 지역별로 / 센서의 평균 값을 계산하고, / 평균 값이 0.8 이상인 값만 반환해."
  - **오류 발생 모델 및 스키마:** gemma3-12B / DM
  - **생성된 SQL:**

```sql
SELECT
    dr.region_name,
    ds.sensor_name,
    AVG(fsh.mean_value) AS average_value
FROM dm.fact_sensor_hourly fsh
JOIN dm.dim_region dr ON fsh.region_id = dr.region_id
JOIN dm.dim_sensor ds ON fsh.sensor_id = ds.sensor_id
JOIN dm.dim_time dt ON fsh.time_id = dt.time_id
WHERE
    dt.date >= DATE('now', '-1 month')  -- ❌ SQLite 문법!
GROUP BY dr.region_name, ds.sensor_name
HAVING AVG(fsh.mean_value) >= 0.8;
```

  - **에러 메시지:** (psycopg2.errors.UndefinedFunction) 오류: date(unknown, unknown) 이름의 함수가 없음
    HINT: 명시적 형변환자를 추가해야 할 수도 있습니다.
  - **원인:** PostgreSQL 문법이 아닌 표현 사용(SQLite 문법, (DATE('now', '-1 month')). *(PostgreSQL에서는 CURRENT_DATE – INTERVAL '1 month' 사용)*

오류 분석

- **잘못된 SQL 생성 + 최대 토큰 수 한계**
  - **질문:** "Q08. REGION_01에서 / RPM 센서 값이 2450에서 2490 사이였던 기록만 조회해줘."
  - **오류 발생 모델 및 스키마:** qwen3-4B / Raw Region
  - **생성된 SQL:**

```sql
SELECT *
FROM raw_region.raw_region_wide
WHERE "REGION_01_SUPPLY_FAN_RPM_PV" BETWEEN 2450 AND 2490
   OR "REGION_01_SUPPLY_FAN_RPM_SV" BETWEEN 2450 AND 2490
   OR "REGION_02_SUPPLY_FAN_RPM_PV" BETWEEN 2450 AND 2490
   OR "REGION_02_SUPPLY_FAN_RPM_SV" BETWEEN 2450 AND 2490
   OR "REGION_03_SUPPLY_FAN_RPM_PV" BETWEEN 2450 AND 2490
   OR "REGION_03_SUPPLY_FAN_RPM_SV" BETWEEN 2450 AND 2490
   OR "REGION_04_SUPPLY_FAN_RPM_PV" BETWEEN 2450 AND 2490
   OR "REGION_04_SUPPLY_FAN_RPM_SV" BETWEEN 2450 AND 2490;
```

  - **SQL 실행:** 정상적인 SQL이 만들어져 실행은 성공했으나, 의도와 다른 SQL이 만들어짐 (region 1을 물었는데, 모든 region 탐색).
    그 결과 너무 많은 레코드 (4,626개)를 반환해 소규모 언어 모델(Small Language Model: SML)에서의 설정했던 최대 토큰 수 (8,192) 초과
  - **에러 메시지:** SQL 실행 후 자연어 답변 생성 시
    Stage 3 (Text Generation): LLM 호출 실패 (qwen3-4b):
    Error code: 400 - "This model's maximum context length is 8192 tokens.
    However, you requested 41959 tokens (39911 in the messages, 2048 in the completion)."
  - **원인:** 결과 데이터가 너무 커서 (39,911 tokens) SLM context window(8,192) 초과

6. 실험

결과 요약

- **실행 오류:** 120건 대비 전체 48건의 실패 중 50%(24건)가 따옴표 누락 문제. DM은 24건 중 1건 (4.2%)
- **실행 시간 지연:** LLM에 전달되는 스키마 프롬프트는 DM이 더 컸으나, 원본 스키마는 더 넓은 범위를 탐색하게 되어(77개 컬럼) 필요한 컬럼을 선택하는 과정에서 실행 시간이 2.2배 증가 (*DM: 1,638 토큰 / 원본 스키마: 1,269 토큰)*
- **토큰 수 감소:** 원본 스키마 대비 42.6% 감소 *(DM: 2,419 토큰 / 원본 스키마: 4,211 토큰)*

주요 발견

- **오류 원인의 세부 분석**
  - **원본 스키마:** 대문자 컬럼명 사용 → PostgreSQL이 따옴표 필수 요구 → LLM이 따옴표 일관되게 생략해 SQL 실행 실패
    (프롬프트에 따옴표 규칙을 명시했으나 LLM이 지시를 일관되게 따르지 못함)
  - **DM 스키마:** ETL 집계 과정에서 mean, max 등 컬럼이 자연스럽게 소문자화 → PostgreSQL 호환성 확보
    → DM 설계 시 컬럼명 정규화가 부수적이지만 실질적인 성능 향상 요인

- **구조화된 DM의 효과**
  - 원본 스키마의 주요 실패 원인인 따옴표 누락 감소
    → 명확한(스타 스키마) 테이블 분리로 JOIN 경로 단순화 및 LLM의 스키마 이해 부담 감소

7. 논의

자연어 모호성과 스키마 구조의 상호작용

- DM은 1시간 단위의 집계 테이블 (mean_value라는 컬럼 존재)
- 5개 케이스에서 DM과 원본 스키마가 동일 질문에 다른 결과 반환
- SQL 문법·논리 모두 정상, 그러나 자연어 해석 차이로 결과 상이

질문 예시	DM에서의 해석	생성 SQL	원본 스키마에서의 해석	생성 SQL
"(특정 날짜의) 평균 온도를 조회해줘"	시간별 평균값 여러 개 반환	mean_value	전체 평균 1개 계산	AVG(value)

시사점

- Text-to-SQL 성능 평가 시 자연어 모호성 통제 필요
- 명확한 질문 단계(clarification) 또는 템플릿 도입 권장
  → 명확한 질문 제공 시 동일한 SQL 및 답변 생성 *(ex – "평균 온도를 하나의 값으로 조회해줘")*

8. 결론 및 추후 연구

연구 요약

- 본 연구는 MES 데이터에 대한 자연어 질의응답 시스템 구축을 위해 Star Schema 기반 데이터 마트 설계와 TAG 파이프라인을 결합한 프레임워크를 제안
- 실험 결과, DM 기반 시스템은 원본 스키마 대비:
  - SQL 실행 성공률 31.7%p 향상 (76.7% vs 45.0%)
  - 실행 시간 41.9% 단축 (6.01초 vs 10.34초)
  - 토큰 사용량 42.6% 절감 (2,419 vs 4,211)
- 이는 구조화된 스키마가 LLM의 SQL 생성 능력을 향상시키며, TAG 기반 시스템에서 데이터베이스 설계의 중요성 확인

연구의 기여

- **실무 적용 가능성 검증:** 실제 제조 현장 MES 데이터로 TAG 시스템의 실효성 입증
- **스키마 설계의 역할 규명:** Text-to-SQL 성능 향상에 DM 설계가 미치는 정량적 효과 제시
- **오류 원인의 정량적 분석:** 스키마 불일치, SQL 문법, LLM 한계 등 체계적 오류 분류 및 해결 방향 제시. DM 설계 시 ETL 과정의 컬럼명 정규화가 부수적이지만 실질적인 성능 향상 요인임을 확인

한계점

- 단일 도메인: MES 센서 데이터에 특화 → 다양한 산업 도메인에 대한 추가 검증 필요
- PostgreSQL 환경에서만 실험 수행 → MySQL, SQLite 등 타 DB 관리 시스템(Database Management System: DBMS)에 대한 일반화 검증 필요

8. 결론 및 추후 연구

추후 연구 방향

- **온톨로지 기반 도메인 지식 통합**
  - **현재 시도:**
    - OWL 온톨로지를 통해 센서 타입, 정상 범위, 유지보수 규칙 등을 정의
  - **한계:**
    - 실제 사용률 및 성능 개선 효과 미측정
    - 도메인 지식이 SQL 생성에 미치는 영향 정량화 필요

- **실시간 모니터링 적용**
  - **목표:** 현장 작업자를 위한 실용적 인터페이스 구축
  - **세부 방향:**
    - 음성 인식 통합: STT(Speech-to-Text) → TAG 파이프라인
    - 대화형 UI: 챗봇 형태의 연속 질의 지원

참고 문헌

- Biswal, A., Patel, L., Jha, S., Kamsetty, A., Liu, S., Gonzalez, J. E., ... & Zaharia, M. (2024). Text2sql is not enough: Unifying ai and databases with tag. *arXiv preprint arXiv:2408.14717*.
- Gao, D., Wang, H., Li, Y., Sun, X., Qian, Y., Ding, B., & Zhou, J. (2023). Text-to-sql empowered by large language models: A benchmark evaluation. *arXiv preprint arXiv:2308.15363*.
- Kimball, R., & Ross, M. (2013). *The data warehouse toolkit: The definitive guide to dimensional modeling*. John Wiley & Sons.
- Mitsopoulou, A., & Koutrika, G. (2025). Analysis of text-to-SQL benchmarks: limitations, challenges and opportunities. In *Proceedings 28th International Conference on Extending Database Technology, EDBT 2025* (pp. 199-212). OpenProceedings. org.
- Shi, L., Tang, Z., Zhang, N., Zhang, X., & Yang, Z. (2025). A survey on employing large language models for text-to-sql tasks. *ACM Computing Surveys*, *58*(2), 1-37.
- Wang, B., Ren, C., Yang, J., Liang, X., Bai, J., Chai, L., ... & Li, Z. (2025, January). Mac-sql: A multi-agent collaborative framework for text-to-sql. In *Proceedings of the 31st International Conference on Computational Linguistics* (pp. 540-557).
- Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., ... & Radev, D. (2018). Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*.
- Zhong, V., Xiong, C., & Socher, R. (2017). Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

Thank you very much.