생성형 AI 기반 RAG 개발자 과정

Day1 미니프로젝트
텍스트 파일 QA

Session 8 / 33 — Day 1

오늘 배운 것만으로 — 첫 RAG 를 완성한다

박수현 · (사)한국정보공학기술사회 · 2026

LLM 이론 · LangChain · ChromaDB CRUD · Agent — 4일 핸즈온

만들 것 — 텍스트 QA 봇

입력

- 텍스트 파일 1개 (예: 사내 FAQ, 규정)
- 사용자 질문

출력

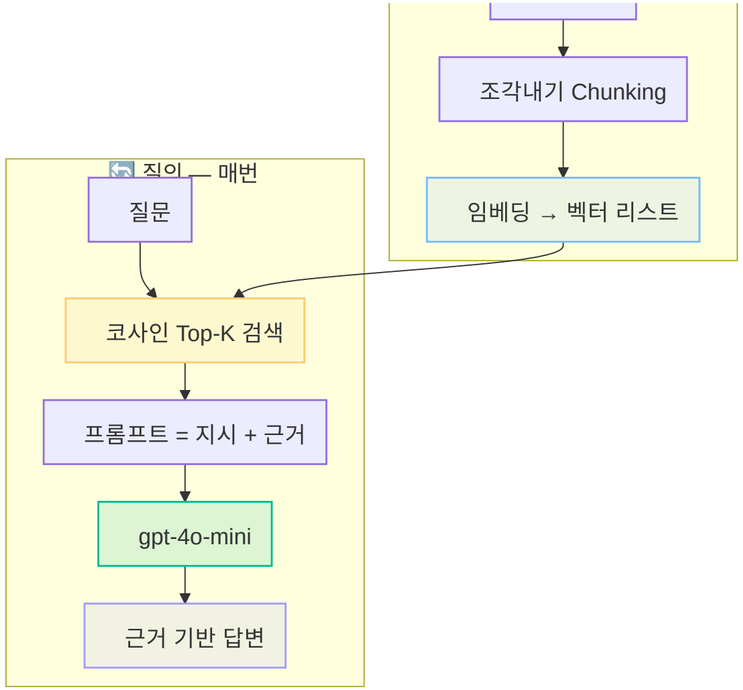
- 파일 **근거에 기반한** 답변
- (가능하면) 어느 조각에서 왔는지

재사용할 부품 (Session 7)

- `embed()` — 문장 → 벡터
- `cosine()` — 유사도
- `search()` — Top-K 검색

> 🎯 새 개념 없음. **조립**만 한다.

인덱싱(준비) → 질의(런타임)



💡 Session 6 의 RAG 3단계(R·A·G)가 그대로 코드가 됨.

준비: 파일 → 조각 (스니펫)

```
text = open("faq.txt", encoding="utf-8").read()

# 가장 단순한 분할 — 빈 줄(문단) 기준
chunks = [c.strip() for c in text.split("\n\n") if c.strip()]

# 각 조각을 미리 임베딩 (한 번만)
chunk_vecs = [embed(c) for c in chunks]
print(f"{len(chunks)}개 조각 준비 완료")
```

- 실무 분할은 Day2 의 **Text Splitter** 로 정교화 (overlap 등)
- 핵심: **미리 임베딩해 재사용** → 질의마다 다시 안 함

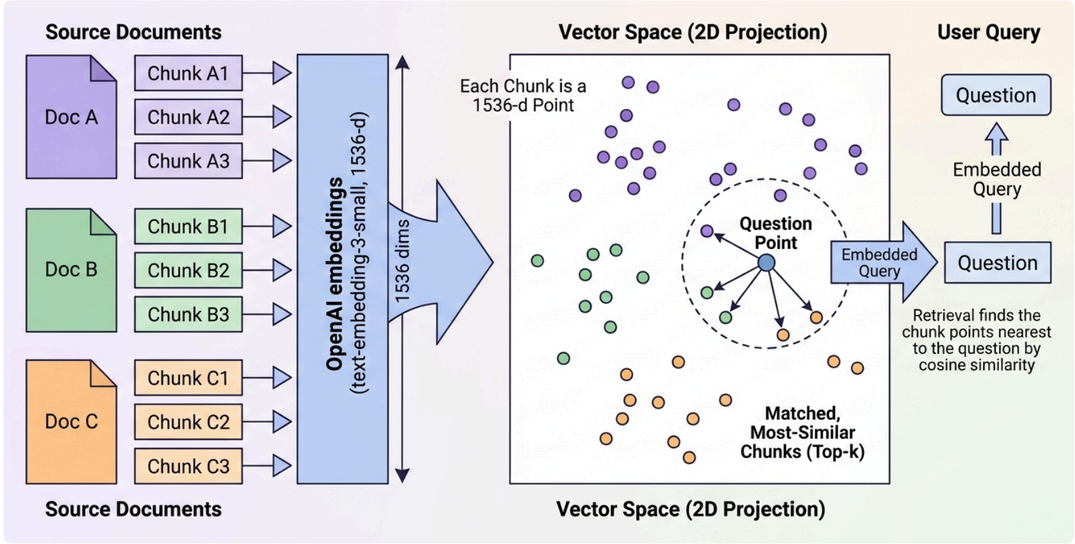
> 📁 참고: `2.langchain/7.RAG/2.loaders/`

질의: 찾고 → 근거 넣고 → 답하기 (스니펫)

```
def ask(question):
    hits = search(question, chunks, chunk_vecs, k=3)   # Top-3 근거
    context = "\n".join(c for _, c in hits)

    prompt = f"""아래 자료에 근거해서만 답하세요.
자료에 없으면 "자료에 없습니다"라고 답하세요.

[자료]
{context}

[질문] {question}"""

    return client.chat.completions.create(
        model="gpt-4o-mini",
        messages=[{"role": "user", "content": prompt}],
    ).choices[0].message.content
```

> 🔑 RAG 의 본질이 이 한 함수에 다 있음: **검색 → 근거 주입 → 생성**.

검색은 "같은 공간의 가장 가까운 점 찾기"



손코딩 RAG · 청크와 질문을 같은 차원으로 임베딩→코사인 Top-K로 근거 청크를 찾아 프롬프트에 주입

- 청크도 질문도 **같은 임베딩 모델 → 같은 차원**(예: 1536) 의 점
- 검색 = 질문 점과 **가장 가까운 청크 점** Top-K 골라내기
- 인덱싱(준비) → **검색** → 생성, 이 세 단계가 RAG 의 전부

> 💡 프레임워크의 한 줄 호출 뒤에서 일어나는 일이 바로 이 그림.

같은 질문, RAG 유무 비교

❌ RAG 없이 (순수 LLM)

> Q: "우리 회사 환불 기간은?"
> A: "보통 14일입니다" *(지어냄 — 모름)*

✅ RAG 로 (근거 주입)

> Q: "우리 회사 환불 기간은?"
> A: "**7일 이내** 가능합니다 (FAQ 3항 근거)"

직접 실험해 볼 것

- 자료에 **없는** 질문 → "자료에 없습니다" 나오는지
- `k` 값을 1 ↔ 5 로 바꿔 답 품질 변화 관찰
- chunk 분할 방식 바꿔보기 (문단 vs 문장)
- **유사도 점수**가 임계값 아래면 "관련 문서 없음" 경고 띄우기
  (거리 기반 `1/(1+거리)` **0~1 점수**면 0.2, **코사인**(−1~1)이면 0.7~0.8 등 — 척도에 맞춰 컷오프)

> 💡 이 세 가지 실험이 곧 Day2 의 **튜닝 포인트**(k, chunk, retriever).

이 미니 RAG 의 한계 → Day2·3 가 푼다

지금의 한계	다음에 해결
매번 전체 비교 (느림)	Day2 — **FAISS/Chroma** 인덱스
껐다 켜면 임베딩 사라짐	Day2 — **영속화**
TXT 만, PDF·DOCX 못 읽음	Day2 — **Document Loader**
프롬프트 직접 문자열 조립	Day2 — **LangChain LCEL**
출처·점수 표시 약함	Day2 — **citation**, Day3 — **웹 UI**
CLI 만	Day3 — **Flask 업로드 서비스**

오늘 만든 것 — 텍스트 QA 봇

1. ✅ 파일 → **조각내기(Chunking) → 임베딩** → 벡터 리스트
2. ✅ 질문 → **코사인 Top-K 검색** → 근거 조각 선택
3. ✅ 근거를 **프롬프트에 주입** → LLM 이 근거 기반 답변
4. ✅ RAG 3단계(**R·A·G**)를 라이브러리 없이 **손으로** 완성
5. ✅ RAG 유무 비교로 **환각 vs 근거** 차이 체감