생성형 AI 기반 RAG 개발자 과정

문서 삭제 기능 (Delete)

Session 21 / 33 — Day 3

파일 삭제 시 관련 chunk 전부 제거

박수현 · (사)한국정보공학기술사회 · 2026

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

이번 시간을 마치면

🗑️ 삭제 구현

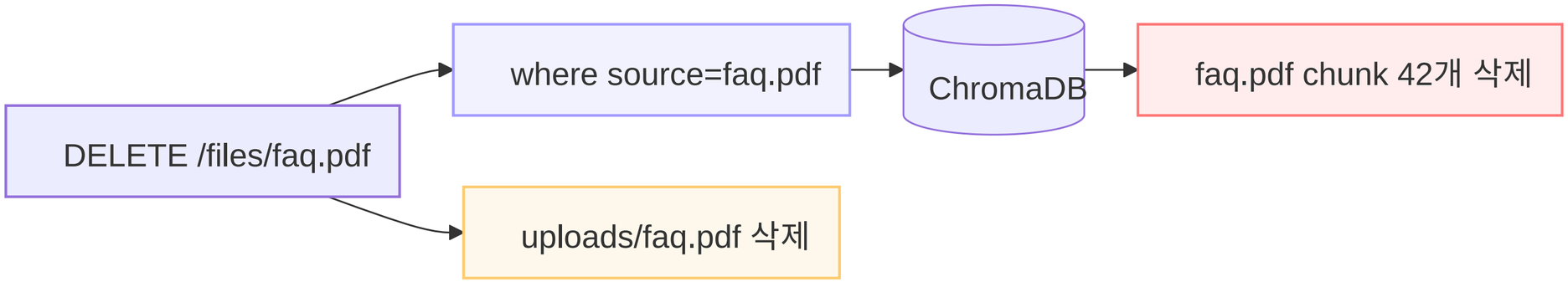
- **source 기준** 일괄 삭제
- `DELETE /files/<source>`
- `where` 필터로 대상 지정

♻️ 함께 처리할 항목

- 원본 파일 동시 삭제
- **재인덱싱**이 필요한 경우
- 멱등성(idempotent) 처리

> 🎯 목표: 파일 1개 삭제 시 관련 chunk 전부와 원본 파일까지 일괄 정리한다.

분산된 chunk의 일괄 삭제



💡 색인 시 부여한 **source 메타데이터**가 삭제 기준이다. 메타데이터가 없으면 삭제 자체가 불가능하다.

where 필터로 일괄 삭제 (스니펫)

```
def delete_document(source: str) → bool:
    store._collection.delete(             # source 일치 chunk 전부
        where={"source": source}
    )
    # 원본 파일도 제거
    path = os.path.join("uploads", source)
    if os.path.exists(path):
        os.remove(path)
    return True
```

- `_collection.delete(where=...)` — 조건에 맞는 chunk **일괄 삭제**
- 벡터(chroma_db) + 원본(uploads) **양쪽** 정리 → **파일↔벡터 동기화**
- 한쪽만 삭제하면 **고아 벡터**(원본 없는 chunk) 또는 유령 파일 발생

> 📁 참고: `vectorstore.py` 의 `delete_document`

리소스 삭제 (스니펫)

```
@app.delete("/files/<path:source>")
def remove_file(source):
    ok = delete_document(source)
    if not ok:
        return jsonify({"error": "삭제 실패"}), 404
    return jsonify({"deleted": source})
```

- `<path:source>` — 파일명에 `/` · `.`가 포함돼도 안전하게 수신
- 성공 시 삭제된 source 반환, 대상이 없으면 404

> 🔑 REST 원칙대로 **리소스 경로 + DELETE 메서드**. 동일 API를 curl로도 호출 가능.

재인덱싱 판단 기준

단순 삭제로 충분

- 문서 1개 제거는 `where` 삭제로 완료
- 나머지 chunk는 그대로 유지

재인덱싱이 필요

- **임베딩 모델 교체** 시
- **chunk 전략 변경** 시
- → 전체 컬렉션 비우고 다시 색인
- 문서 수정: `where` 삭제 → **다시 add** (일관성 유지)

멱등성 (idempotent)

- 동일 파일 재업로드 시 중복 색인 방지
- **결정적 ID** 또는 기존 source 삭제 후 추가
- 동일 요청을 반복해도 결과 동일

> 🔑 삭제 후 재추가 패턴이 **부분 수정**의 기본 방식.

핵심 5가지

1. ✅ 한 파일 = **여러 chunk** → `source` 로 일괄 삭제
2. ✅ `_collection.delete(where={"source": ...})`
3. ✅ 벡터 + 원본 파일 **둘 다** 정리
4. ✅ `DELETE /files/<source>` — REST 리소스 삭제
5. ✅ 모델·청크 변경 시 **재인덱싱**, 중복은 **멱등 처리**