생성형 AI 기반 RAG 개발자 과정

ChromaDB 관리 조회 (Read)

Session 20 / 33 — Day 3

저장된 문서 목록 조회

박수현 · (사)한국정보공학기술사회 · 2026

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

이번 시간을 마치면

📋 목록 구성

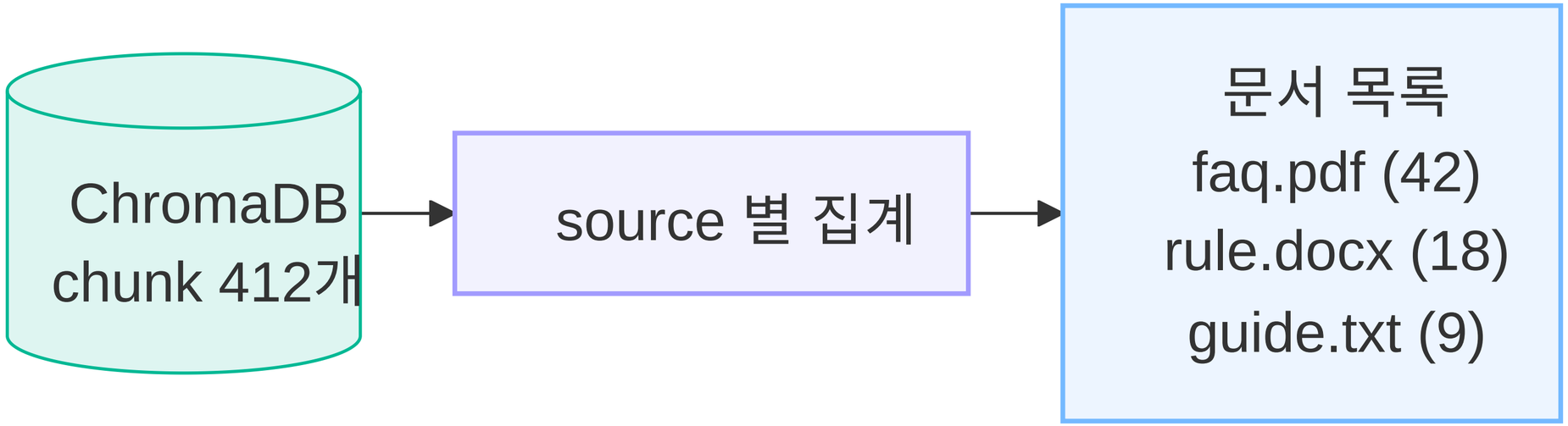
- `GET /files` 로 문서 목록
- chunk → **source 단위 집계**
- 파일별 chunk 수

🔎 메타데이터 활용

- Chroma 에서 메타데이터 **조회**
- `where` 필터 기본
- source 로 그룹핑

> 🎯 목표: 여러 chunk로 분할 저장된 문서를 **사용자용 문서 목록**으로 집계한다.

chunk의 파일 단위 그룹핑



💡 저장은 chunk 단위, 표시는 **파일(source) 단위**. 둘을 잇는 연결 고리가 메타데이터 `source`다.

list_documents 서비스 (스니펫)

```
def list_documents():
    data = store._collection.get(include=["metadatas"])
    counts = {}
    for m in data["metadatas"]:
        src = m.get("source", "unknown")
        counts[src] = counts.get(src, 0) + 1
    return [{"source": s, "chunks": n}
            for s, n in counts.items()]
```

- `_collection.get(include=["metadatas"])` — 모든 chunk 의 메타데이터
- `source` 로 카운트 → **파일별 chunk 수**
- `_collection.count()` — 전체 chunk 수로 **컬렉션 상태 점검**

> 📁 참고: `vectorstore.py` 의 `list_documents` / `_distinct_sources`

라우트는 서비스 호출만 (스니펫)

```
@app.get("/files")
def get_files():
    return jsonify(list_documents())
```

응답 예:

```
[
  {"source": "faq.pdf",   "chunks": 42},
  {"source": "rule.docx", "chunks": 18}
]
```

> 🔑 route는 서비스 결과를 JSON으로 반환할 뿐, 목록 UI 구성은 프런트가 담당한다.

특정 문서만 조회 (개념)

where 로 좁히기

```
store._collection.get(
  where={"source": "faq.pdf"},
  include=["metadatas"], limit=5
)
```

- `source`가 일치하는 chunk만 반환
- `include` / `limit`으로 **저장 내용 미리보기**

활용 시점

- **특정 문서 내부로 한정**한 검색
- 문서 **삭제 대상** 식별
- 부서·작성일 등으로 **필터링**

> 🔑 동일한 `where` 문법이 **문서 삭제·필터 검색**에 그대로 적용된다.

핵심 5가지

1. ✅ 저장은 **chunk 단위**, 표시는 **파일(source) 단위**
2. ✅ `GET /files` → `list_documents()` 서비스 호출
3. ✅ 메타데이터를 **source 로 집계**해 파일별 chunk 수
4. ✅ `get(include=["metadatas"])` 로 전체 조회 · `count()` 로 상태 점검
5. ✅ `where` **필터**로 특정 문서만 조회 (`limit` 으로 미리보기)