02 DB 구축 보고서

진행 상태	읽기 전

Lockument DB 구축 통합 보고서 (메인+부록, 캡처 반영판)

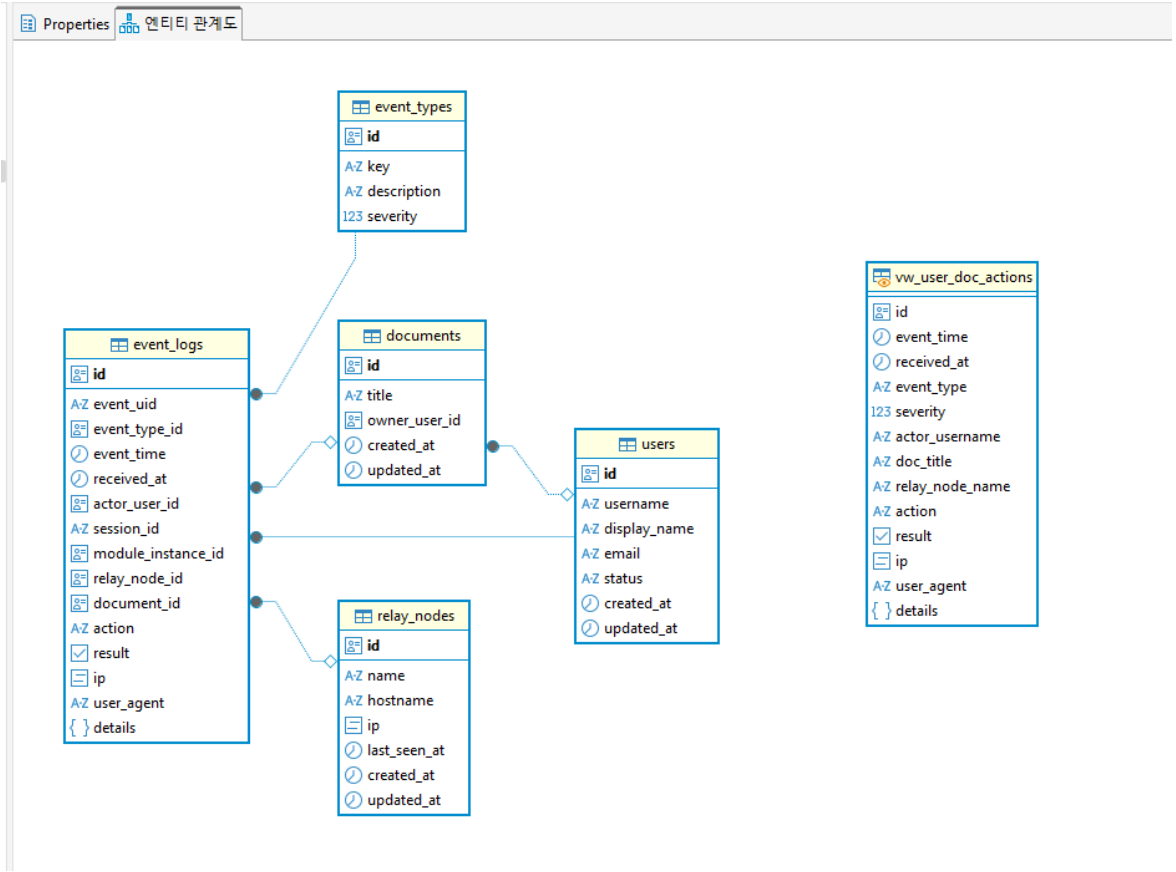
- **작성일**: 2025-09-01
- **대상**: 한국폴리텍대학 / 프로젝트 담당 교수 / 팀원
- **시스템명**: 문서 통합 플랫폼 – 감사(Audit) DB 모듈

---

0) 핵심 요약

- **무엇**: PostgreSQL `lockument_db`, 스키마 `pii_audit` 구축함. 이벤트를 **JSON 한 건**으로 받아 **하나의 함수**로 저장하는 구조 확립함
- **왜**: 운영 단순화, 기록 일관성, 보안(최소 데이터 보관) 확보 목적
- **어떻게**: 중계 API → `ingest_event_json(jsonb)` → `event_logs` 삽입, 운영 조회는 **뷰** `vw_user_doc_actions` 활용함
- **정책**: 시간대 **Asia/Seoul(UTC+9)**, **severity=1/2/3** 고정(CHECK 제약), **단일 테이블** 기본, 파티션은 옵션으로 유지함

*(캡처: **pii_audit 스키마 전체 ERD**) – DBeaver >* `lockument_db → Schemas → pii_audit` *우클릭 → View Diagram → ER Diagram → Export PNG*



1) 보고서 목적

- 감사 로그(DB)의 설계·구현·운영 기준을 **간단·명확**하게 공유함
- 교수/동료가 바로 확인 가능한 **쿼리·그림(ERD/결과 캡처)** 포함해 이해도 향상함

2) 배경·범위

- **배경**: 업로드·암/복호화·다운로드 등 사용자/시스템 행동의 **시간순 추적** 필요성 증가함
- **범위**: DB 설계, 적재 함수, 운영 뷰, 권한, 기본 데이터(severity 포함) 정의 및 검증함
- **비범위**: 문서 원본·가공물 저장 제외(보안/운영 단순화), 대용량 파이프라인·비정형 저장소 제외함

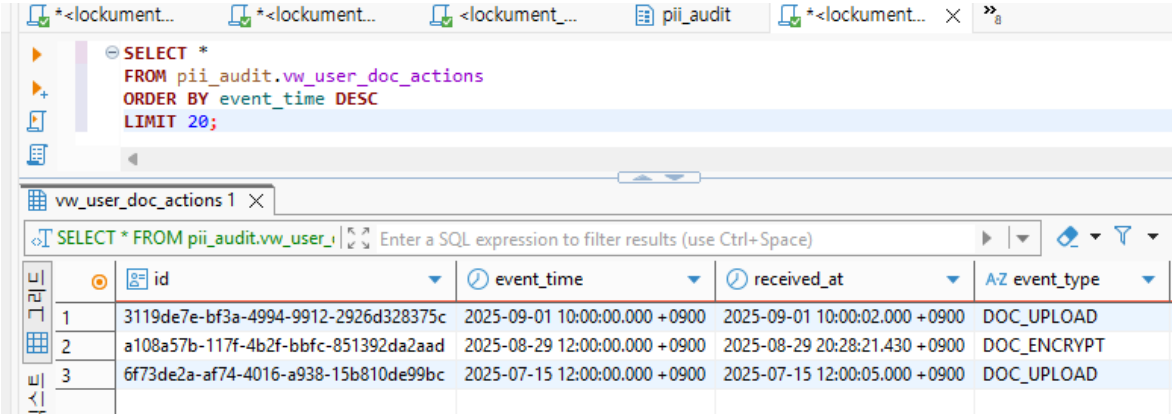
3) 시스템 구조(텍스트 다이어그램)

사용자/시스템 → **중계 API** → `ingest_event_json(jsonb)` → **PostgreSQL: pii_audit.event_logs**

↘ 운영/리포트 → `vw_user_doc_actions` **조회**

*(캡처: **운영 뷰 최근 20건**) —*

```
SELECT *
FROM pii_audit.vw_user_doc_actions
ORDER BY event_time DESC
LIMIT 20;
```



4) 우리가 만든 것(객체 요약)

구분	이름	하는 일
테이블	`event_logs`	이벤트 본표(언제/무엇/누가/무엇에/결과/부가정보) 저장
테이블	`event_types`	이벤트 종류와 **severity(1~3)** 관리
테이블	`documents`	문서 메타(제목·소유자). **원본/가공물 비저장**
테이블	`users`	사용자 기본 정보
테이블	`relay_nodes`	이벤트 유입 중계 노드
뷰	`vw_user_doc_actions`	최근 활동 타임라인(운영 조회용)
함수	`ingest_event_json(jsonb)`	JSON 한 건 → 내부 ID 매핑 → `event_logs` 삽입

*(캡처: **event_logs 중심 미니 ERD**) – `event_logs` 더블클릭 → References → Diagram*

5) 설계 선택 이유(쉬운 설명)

- **단일 테이블 우선**: 개발·운영 단순화, 대부분 요구 충분 충족함
- **월별 파티션 옵션**: 데이터 폭증 시 선택 적용 가능함
- **문서 원본 비저장**: 보안·운영 비용 절감, 메타만 보관함

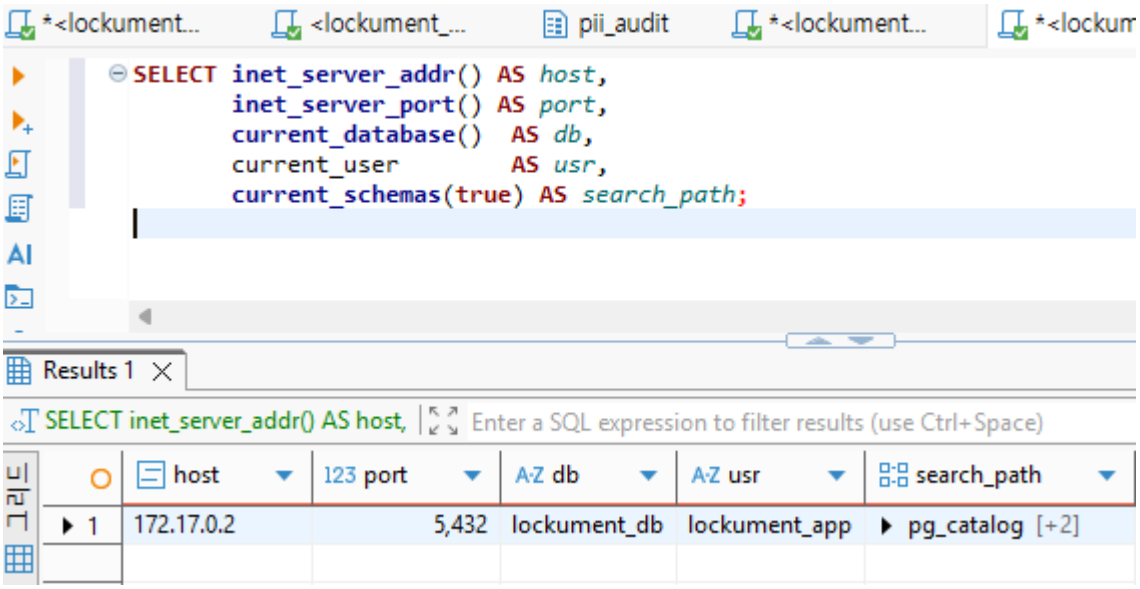
- **severity 3등급**: 알림/리포트 기준 단순화(1=보통, 2=주의, 3=중대) 함

6) 운영 방법(쓰기/읽기)

6-1) 세션 기본값 확인

*(캡처: **세션/검색경로 확인**)*

```
SELECT inet_server_addr() AS host,
       inet_server_port() AS port,
       current_database()  AS db,
       current_user        AS usr,
       current_schemas(true) AS search_path;
```

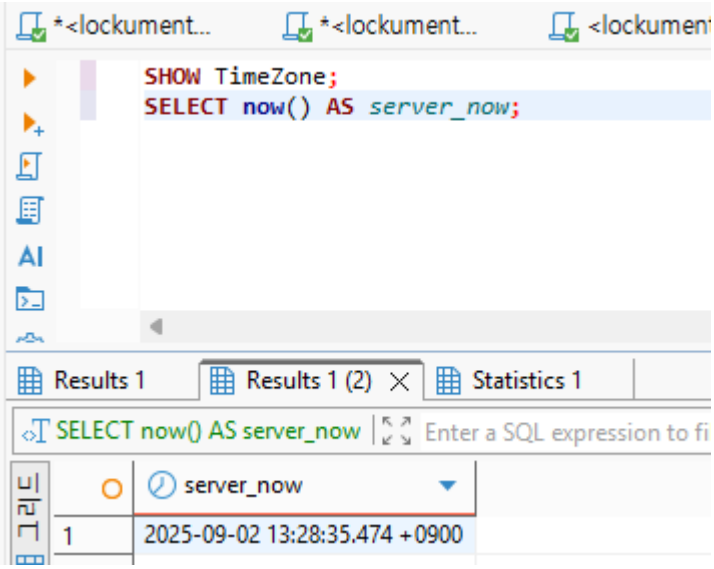


> 정상 기준: db=lockument_db, search_path={pii_audit, public}

6-2) 타임존 확인

*(캡처: **DB 시간대/현재시각**)*

```
SHOW TimeZone;
SELECT now() AS server_now;
```



> 정상 기준: TimeZone = Asia/Seoul

6-3) JSON 쓰기(ingest) 스모크

*(캡처: **ingest 호출 결과 – inserted_id**)*

```
SELECT pii_audit.ingest_event_json('{
  "event_uid":"report-evt-0001",
  "event_type":"DOC_UPLOAD",
  "event_time":"2025-09-01T10:00:00+09:00",
  "received_at":"2025-09-01T10:00:03+09:00",
  "actor_username":"tester",
  "document_title":"보고서_샘플.xlsx",
  "action":"upload",
  "result":true,
  "ip":"203.0.113.10",
  "user_agent":"Mozilla/5.0",
  "relay_node_name":"relay-api-01",
  "details":{"note":"capture"}
}'::jsonb) AS inserted_id;
```

*(캡처: **방금 건 원표 확인**)*

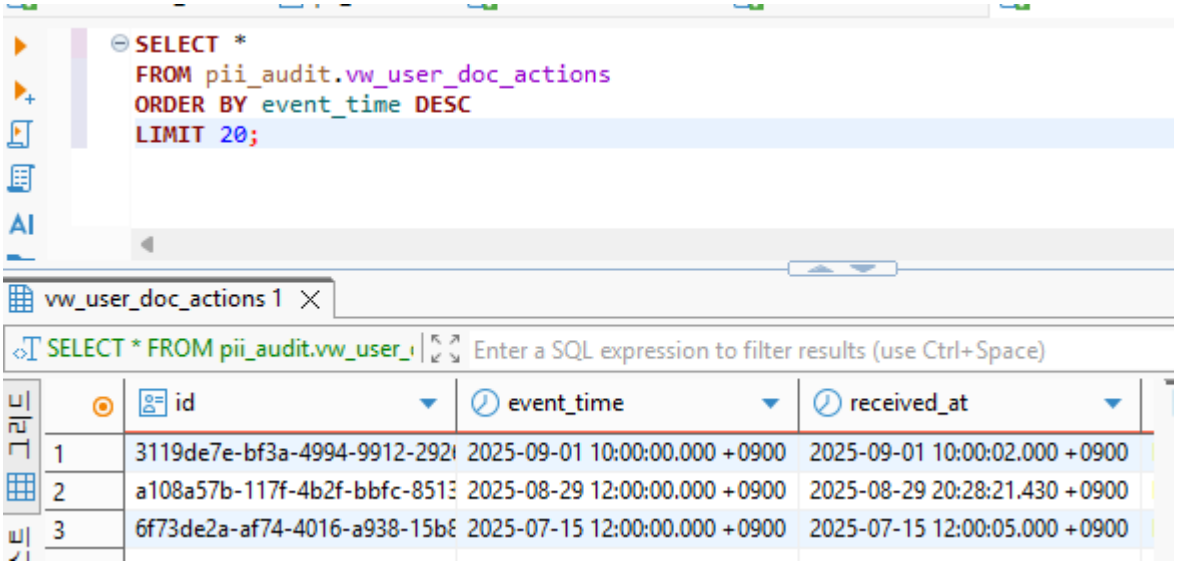
```
SELECT id, event_uid, event_time, action, result
FROM pii_audit.event_logs
```

```
WHERE event_uid='report-evt-0001';
```

6-4) 운영 뷰 읽기(최근)

*(캡처: **뷰 최근 20건**)*

```
SELECT *
FROM pii_audit.vw_user_doc_actions
ORDER BY event_time DESC
LIMIT 20;
```

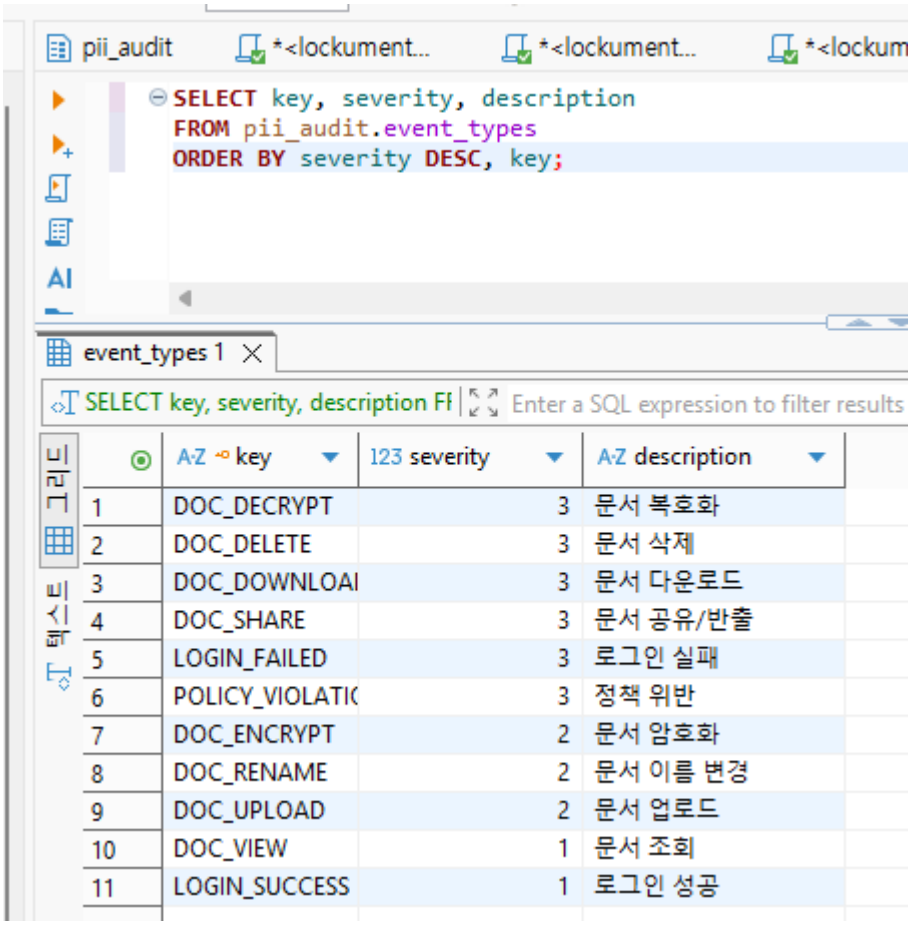


7) 정책(간단 표)

항목	값/설명
시간대	Asia/Seoul(UTC+9)
severity	1=보통(관찰), 2=주의(보안 민감/실패), 3=중대(권한 실패/PII 대량/정책 위반) – CHECK 제약
데이터 최소화	문서 원본·가공물 **미보관**
저장 구조	기본: 단일 테이블 / 선택: 월별 파티션
권한	앱 계정 최소 권한(스키마 USAGE, 테이블 SELECT, 함수 EXECUTE)
검색 경로	`search_path = pii_audit, public` 고정

*(캡처: **event_types 표준키/심각도**)*

```
SELECT key, severity, description
FROM pii_audit.event_types
ORDER BY severity DESC, key;
```



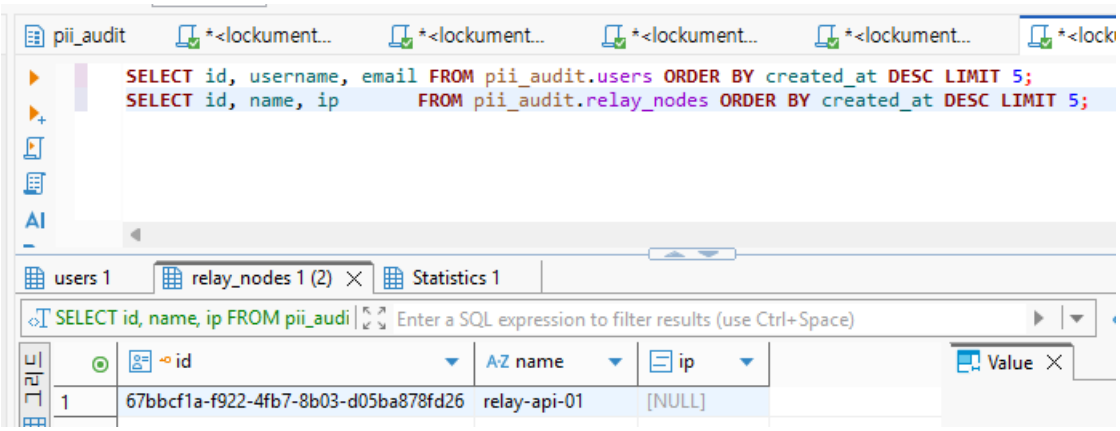
---

8) 점검 체크리스트(빠른 확인)

- **기준데이터**

  *(캡처: **users/relay_nodes 샘플**)*

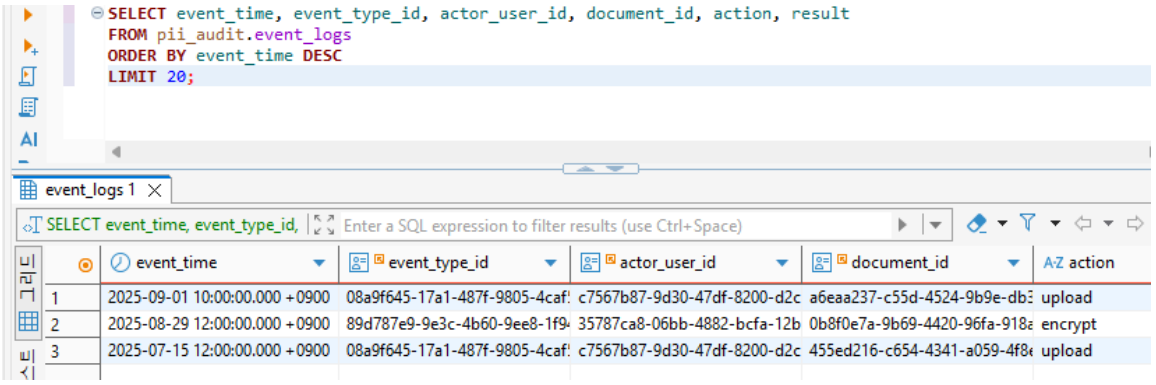
```
SELECT id, username, email FROM pii_audit.users ORDER BY created_a
t DESC LIMIT 5;
SELECT id, name, ip      FROM pii_audit.relay_nodes ORDER BY created
_at DESC LIMIT 5;
```



- **조회/정렬**

  *(캡처: **타임라인 정렬**)*

  ```
  SELECT event_time, event_type_id, actor_user_id, document_id, action, result
  FROM pii_audit.event_logs
  ORDER BY event_time DESC
  LIMIT 20;
  ```



---

9) Docker 연동(요약)

- Ubuntu 22.04 LTS + Docker/Compose 설치 완료 상태 가정함
- API 컨테이너에서 DSN 예시:
  `postgresql://lockument_app:비번@10.10.30.169:5432/lockument_db?search_path=pii_audit,public`
- `docker compose up -d` 기동, `/api/ingest`, `/api/actions/recent` 엔드포인트로 스모크 수행함

---

10) 자주 묻는 질문(FAQ)

- **왜 severity는 3단계인가** → 알림/리포트 기준 단순화, 현장 적용 용이성 확보 목적
- **문서 버전 저장 안 하나** → 보안·운영 비용 고려, 메타만 보관 정책 선택함(필요 시 앱 레벨 관리)
- **파티션 꼭 필요한가** → 아님. 데이터 폭증 시에만 전환함

---

11) 향후 과제

- severity=2/3 조건 알림(메일/Slack) 연동함
- 운영 대시보드(뷰 기반 테이블·필터) 제작함
- 백업/보존 주기 문서화 및 자동화 스크립트 구성함
- 대량 데이터 시 월별 파티션 전환 스크립트 적용함

---

12) 부록 — 테이블·컬럼 설명

> 표기: PK 기본키, FK 외래키, [옵션] 선택 입력, 🔗 연계

12.1 `pii_audit.event_logs` — 이벤트 본표

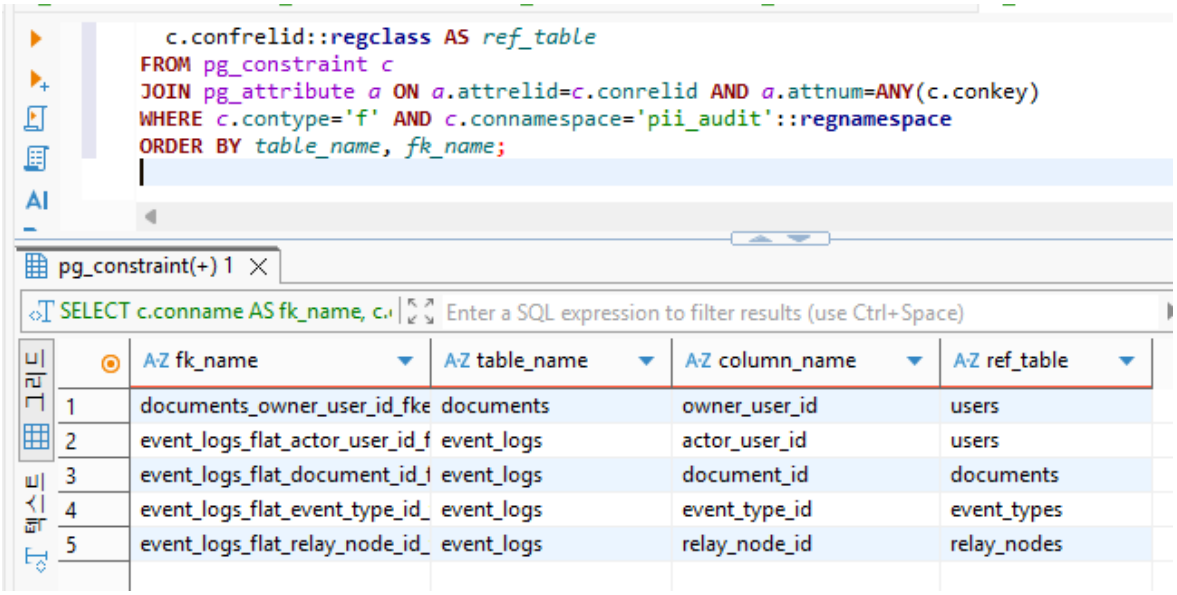
요지: “언제, 누가, 무엇을, 어떤 문서에, 결과는?” 한 줄 저장

컬럼	타입	설명	비고
`id` **(PK)**	uuid	로그 식별자	내부 생성
`event_uid`	text	외부 이벤트 고유키(중복 방지)	필요 시 엄격 dedup
`event_type_id` **(FK)**	uuid	이벤트 분류	🔗 `event_types.id`
`event_time`	timestamptz	사건 실제 시각	앱 기준
`received_at`	timestamptz	서버 수신 시각	now 기본
`actor_user_id` **(FK)**	uuid	행위 주체	🔗 `users.id`
`session_id`	uuid	세션 식별자	[옵션]
`module_instance_id`	uuid	모듈 인스턴스	[옵션]
`relay_node_id` **(FK)**	uuid	중계 노드	🔗 `relay_nodes.id`
`document_id` **(FK)**	uuid	대상 문서	🔗 `documents.id`
`action`	text	세부 동작 키	[옵션]
`result`	bool	성공/실패	[옵션]
`reason`	text	사유	[옵션]

컬럼	타입	설명	비고
ip	inet	클라이언트 IP	[옵션]
user_agent	text	UA 문자열	[옵션]
details	jsonb	확장 메타	[옵션]
skew_ms	int8	received_at - event_time ms	[옵션]

*(캡처: **FK 목록**)*

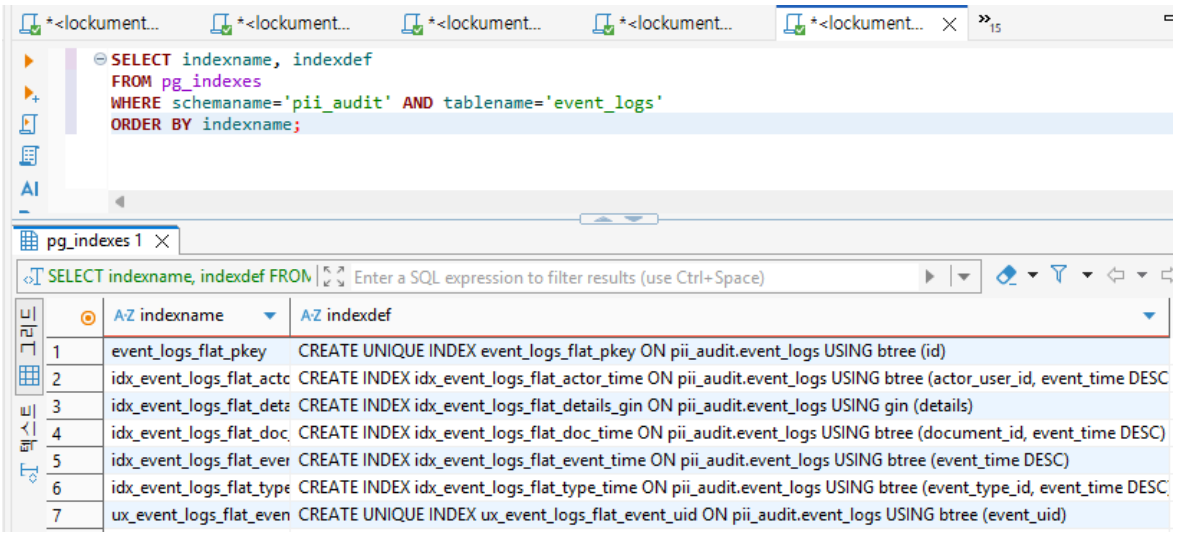
```
SELECT
  c.conname AS fk_name,
  c.conrelid::regclass AS table_name,
  a.attname AS column_name,
  c.confrelid::regclass AS ref_table
FROM pg_constraint c
JOIN pg_attribute a ON a.attrelid=c.conrelid AND a.attnum=ANY(c.conkey)
WHERE c.contype='f' AND c.connamespace='pii_audit'::regnamespace
ORDER BY table_name, fk_name;
```



*(캡처: **event_logs 인덱스**)*

```
SELECT indexname, indexdef
FROM pg_indexes
WHERE schemaname='pii_audit' AND tablename='event_logs'
```

```
ORDER BY indexname;
```

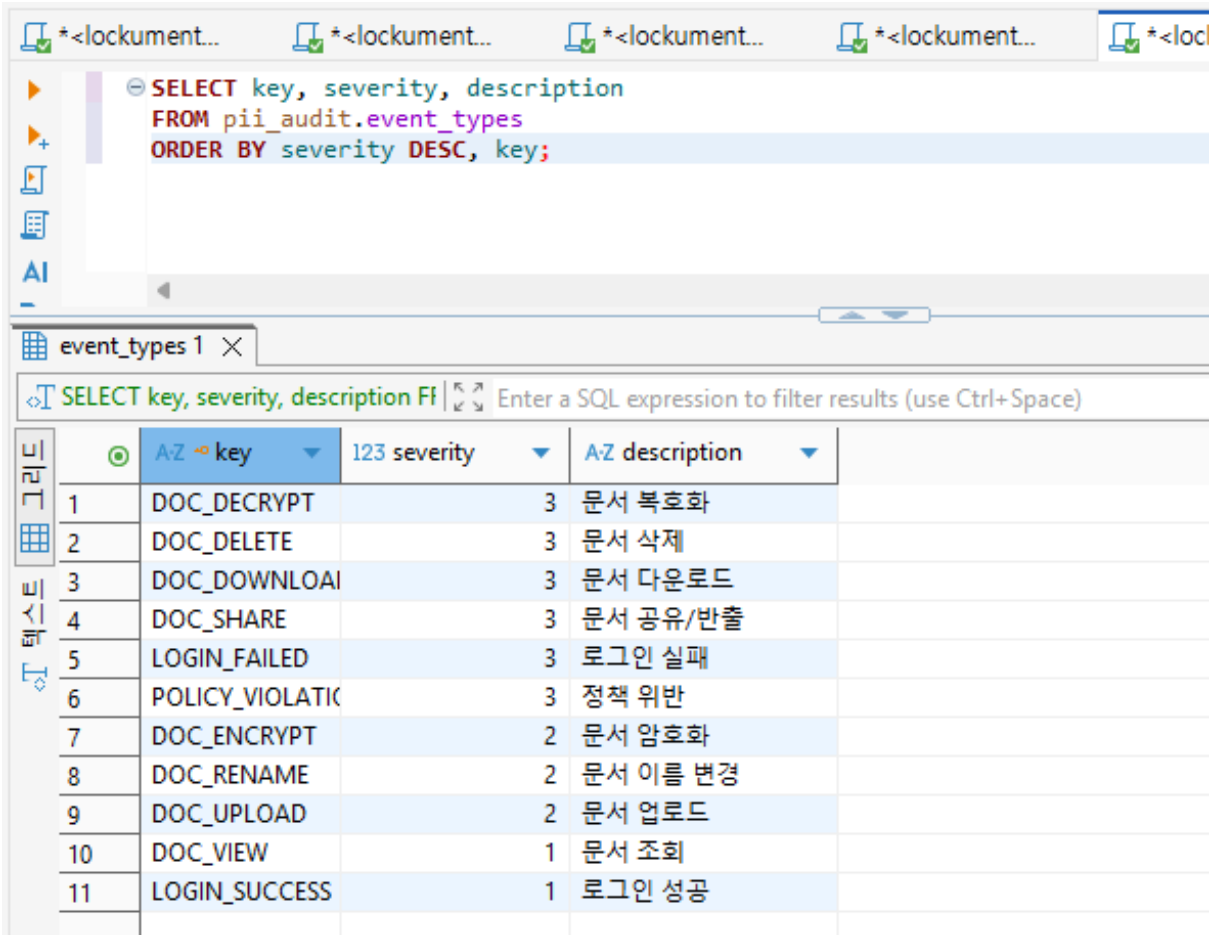


12.2 `pii_audit.event_types` — 이벤트 종류표

컬럼	타입	설명
`id` **(PK)**	uuid	종류 식별자
`key` (unique)	text	예: DOC_UPLOAD, DOC_ENCRYPT
`description`	text	설명
`severity` (1~3)	int4	1=보통, 2=주의, 3=중대(CHECK)

*(캡처: **event_types 행**)*

```
SELECT key, severity, description
FROM pii_audit.event_types
ORDER BY severity DESC, key;
```

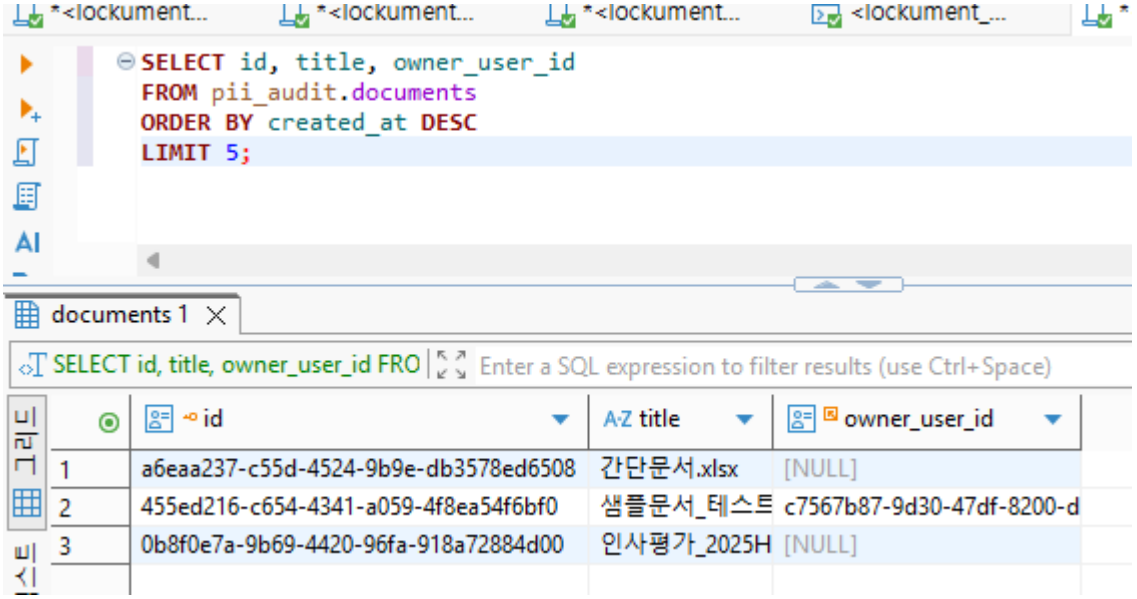


12.3 pii_audit.documents — 문서 메타

컬럼	타입	설명
id **(PK)**	uuid	문서 식별자
title	text	제목
owner_user_id **(FK)**	uuid	소유자(🔗 users.id)
created_at / updated_at	timestamptz	생성/갱신

*(캡처: **documents 샘플**)*

```
SELECT id, title, owner_user_id
FROM pii_audit.documents
ORDER BY created_at DESC
LIMIT 5;
```

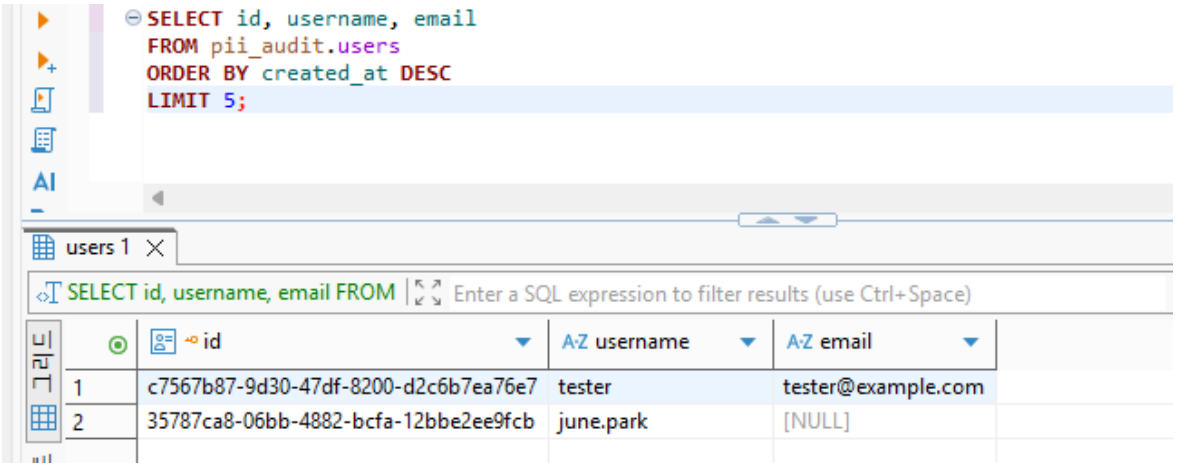


12.4 pii_audit.users — 사용자

컬럼	타입	설명
id (**PK**)	uuid	사용자 식별자
username (unique)	text	표준 계정명
display_name	text	표시 이름
email (unique)	text	이메일
status	text	상태
created_at / updated_at	timestamptz	생성/갱신

*(캡처: **users 샘플**)*

```
SELECT id, username, email
FROM pii_audit.users
ORDER BY created_at DESC
LIMIT 5;
```

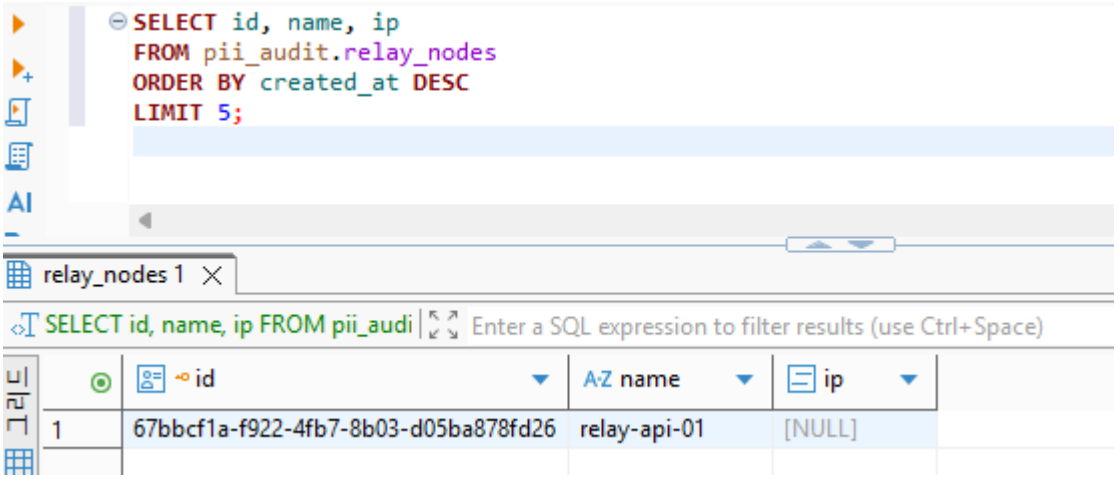


12.5 pii_audit.relay_nodes — 중계 노드

컬럼	타입	설명
id **(PK)**	uuid	노드 식별자
name (unique)	text	예: relay-api-01
hostname	text	호스트명
ip	inet	노드 IP
last_seen_at	timestamptz	최근 통신
created_at / updated_at	timestamptz	생성/갱신

*(캡처: **relay_nodes 샘플**)*

```
SELECT id, name, ip
FROM pii_audit.relay_nodes
ORDER BY created_at DESC
LIMIT 5;
```



12.6 뷰 `pii_audit.vw_user_doc_actions`

- 열: `id, event_time, received_at, event_type, severity, actor_username, doc_title, relay_node_name, action, result, ip, user_agent, details`

12.7 함수 `pii_audit.ingest_event_json(jsonb)`

- JSON 키 → 내부 ID 매핑 → `event_logs` 삽입
- `event_uid` 중복 시 보조 테이블로 엄격 차단 가능(옵션)

---

13) 참고 캡처 파일명(권장 규칙)

- `erd_pii_audit_full.png`
- `session_search_path.png` / `timezone_now.png`
- `event_types_severity.png` / `users_sample.png` / `relay_nodes_sample.png` / `documents_sample.png`
- `ingest_inserted_id.png` / `event_logs_verify_uid.png` / `vw_actions_recent20.png`
- `fk_list.png` / `idx_event_logs.png`

결론

- 단순·명확·보안 친화 스키마로 **즉시 서비스 연동 가능** 상태 확립함