

이호영

백엔드 엔지니어

이메일 eric250@naver.com
전화 010-3421-6615
GitHub github.com/puppywimy

Java Kotlin Spring Boot Spring Data JPA PostgreSQL MySQL AWS Terraform

주요 트러블 슈팅

READTHEM.md 개발자의 도서 리뷰 플랫폼

소셜 로그인 직후 병렬 API 요청 6건 중 최대 5건이 실패하던 **세션 경쟁 조건 해결**

p.2

InvestUP 주식 초보자를 위한 교육형 모의 주식 트레이딩 서비스

데이터 보존이 필요해진 시점에 Flyway 도입, 8일간 마이그레이션 17건 자동 반영

p.12

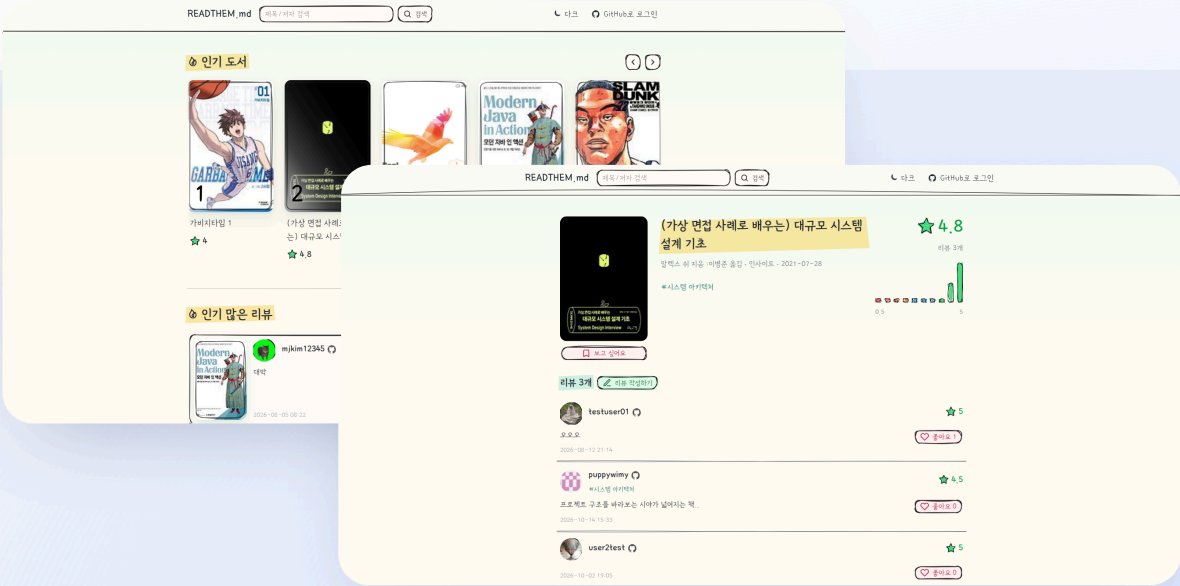
소셜 로그인 직후 병렬 API 요청 6건 중 최대 5건이 실패하던 세션 경쟁 조건 해결

- 01 문제 정의
- 02 해결 과정
- 03 성과 및 회고

프로젝트 소개

READTHEM.md 개발자의 도서 리뷰 플랫폼

진행 기간	2026.06.25 ~ 2026.07.10, 2026.07.23 ~ 2026.08.03 (약 4주)
역할	3인 팀 · 팀장 · 백엔드(도서 API 연동, 아키텍처 설계), 인프라 전담
기술 스택	Kotlin 2.3 · Spring Boot 4 · Spring Data JPA · Spring Security · Redis · MySQL · Docker · AWS · Terraform
링크	서비스 · GitHub



01 문제 정의



기존 구조

- Spring Security + JJWT: JWT 인증을 사용해 `SessionCreationPolicy.STATELESS` 설정
- Spring Security OAuth2 Client: GitHub 소셜 로그인만 지원
 - 로그인 과정에서만 CSRF 방지용 `state` 값을 **세션에 임시 저장** (2 + 4)

문제 상황

- 무중단 배포를 위해 다중 인스턴스로 구성하면서 운영 세션 저장소를 **톰캣 메모리에서 Redis로 변경**
- 이후 소셜 로그인 직후 메인 화면의 병렬 API 요청 6건 중 **매번 일부만 성공, 나머지는 500** (성공 건수와 성공 요청이 매번 다름)
- 로그에는 `Session was invalidated`, 가끔 `ERR no such key`

02 해결 과정 · 문제 상황 재현

쿠키를 모두 지운 상태에서 소셜 로그인 → 메인 화면 리다이렉트 직후 병렬 요청

Network 탭

Name	Status	Type	Waterfall
rank?type=views&page=0&size=10	500	fetch	
latest?page=0&size=10	500	fetch	
likeCount?page=0&size=10	200	fetch	
rank?type=rating&page=0&size=10	500	fetch	
rank?type=reviewCount&page=0&size=10	500	fetch	
me	500	fetch	

Name	Status	Type	Waterfall
rank?type=views&page=0&size=10	500	fetch	
latest?page=0&size=10	500	fetch	
likeCount?page=0&size=10	200	fetch	
rank?type=rating&page=0&size=10	500	fetch	
rank?type=reviewCount&page=0&size=10	500	fetch	
me	200	fetch	

Name	Status	Type	Waterfall
rank?type=views&page=0&size=10	200	fetch	
latest?page=0&size=10	500	fetch	
likeCount?page=0&size=10	500	fetch	
rank?type=rating&page=0&size=10	500	fetch	
rank?type=reviewCount&page=0&size=10	200	fetch	
me	200	fetch	

로그

```
java.lang.IllegalStateException (Create breakpoint) : Session was invalidated
    at org.springframework.session.data.redis.RedisSessionRepository.save(RedisSessionRepository.java:131)
    at org.springframework.session.data.redis.RedisSessionRepository.save(RedisSessionRepository.java:47)
    at org.springframework.session.web.http.SessionRepositoryFilter$SessionRepositoryRequestWrapper.commitSession(SessionRepositoryFilter.java:230)

org.springframework.data.redis.RedisSystemException (Create breakpoint) : Error in execution
    ... 16 frames omitted (Lettuce, RedisTemplate)
    at org.springframework.data.redis.core.RedisTemplate.rename(RedisTemplate.java:713)
    at org.springframework.session.data.redis.RedisSessionRepository$RedisSession.saveChangeSessionId(RedisSessionRepository.java:319)
    at org.springframework.session.data.redis.RedisSessionRepository$RedisSession.save(RedisSessionRepository.java:307)
    at org.springframework.session.data.redis.RedisSessionRepository.save(RedisSessionRepository.java:134)
    at org.springframework.session.data.redis.RedisSessionRepository.save(RedisSessionRepository.java:47)
    at org.springframework.session.web.http.SessionRepositoryFilter$SessionRepositoryRequestWrapper.commitSession(SessionRepositoryFilter.java:230)
    ...
Caused by: io.lettuce.core.RedisCommandExecutionException (Create breakpoint) : ERR no such key
```

두 에러(Session was invalidated , ERR no such key)의 스택 트레이스를 확인해보니

모두 RedisSessionRepository.save 에서 발생

02 해결 과정 · 원인 추적

SessionRepositoryFilter.commitSession

```
private void commitSession() {  
    HttpSessionWrapper wrappedSession = getCurrentSession();  
    if (wrappedSession == null) {  
        1  
        if (isInvalidateClientSession()) {  
            SessionRepositoryFilter.this.httpSessionIdResolver.expireSession(request, this, this.response);  
        }  
    }  
    else {  
        S session = wrappedSession.getSession();  
        String requestedSessionId = getRequestedSessionId();  
        clearRequestedSessionCache();  
        SessionRepositoryFilter.this.sessionRepository.save(session);  
        2  
        String sessionId = session.getId();  
    }  
}
```

병렬 요청 중 500 응답 1건의 Request Headers

Request URL	http://localhost:8080/api/v1/books/rank?type=views&page=0&size=10
Request Method	GET
Status Code	500 Internal Server Error
Cookie	refreshToken= ; accessToken= ; SESSION=ZDAyZWJmNTetYjAzNy00MjFhLWE2YjQtZjdlNWw1NWZhZjIi 3

스택 트레이스에서 프레임워크 코드로

- RedisSessionRepository.save 는 요청이 끝날 때마다 실행되는 commitSession 에서 호출(2)
- 단, 현재 세션이 있는 요청만 저장(1)

→ JWT 인증(STATELESS)을 쓰는 일반 API 요청이 세션을 가지고 있다는 뜻

브라우저에서 세션의 출처 확인

- 실패한 요청의 Cookie 에 토큰과 함께 SESSION 쿠키가 담겨 있음(3).
- 이 쿠키는 로그인 성공 응답(6)의 Set-Cookie 로 발급됨.

→ 소셜 로그인용 임시 세션이 로그인 이후에도 남아 모든 API 요청에 따라붙고 있었음.

* 6은 p.3 소셜 로그인 흐름도의 단계

02 해결 과정 · 임시 세션 만료

OAuthLoginSuccessHandler.onAuthenticationSuccess

```
// 2. '#' 앞은 redirectUrl, 뒤는 originState
redirectUrl = decodedStateParam.split(regex = "#".toRegex(), limit = 2).toTypedArray()[0]
}

// getSession이 null을 반환할 수 있어 세이프콜을 해야 합니다.
request.getSession(create = false)?.invalidate()

// ✅ 최종 리다이렉트
rq.sendRedirect(redirectUrl)
```

로그인 성공 시 세션 만료

- 임시 세션의 역할은 `state` 비교(④)에서 끝남 → 로그인 이후 요청에는 필요 없는 세션
- 로그인 성공 시점(⑥)에 **세션을 만료하는 코드 한 줄 추가**

결과

- 같은 재현 조건(쿠키 삭제 → 소셜 로그인 → 병렬 요청)에서 6건 모두 200

해결은 됐지만 의문이 남았음.

STATELESS 인데 왜 ⑥에서 세션 쿠키가 발급됐고, 같은 세션을 가진 요청들은 왜 실패했을까?

02 해결 과정 · 근본 원인 분석

STATELESS 인데 ⑥에서 SESSION 쿠키가 발급된 이유

- STATELESS 는 Spring Security가 세션에 인증 정보를 저장하지 않게 할 뿐, 다른 곳의 세션 생성은 막지 않음.
- 인증 성공 시 기본 전략(세션 고정 보호)이 `changeSessionId()` 로 세션 회전 → 응답에 새 SESSION 쿠키 발급

매번 다른 요청이, 두 종류의 에러로 실패한 이유

- ⑥ 이후 병렬 요청 6건은 모두 토큰과 SESSION 쿠키를 들고 감
- STATELESS 라 인증 정보가 세션에 없음 → `SessionManagementFilter` 가 매 요청을 "막 인증됨"으로 보고 `changeSessionId()` 실행
- → 6건이 **같은 세션을 각자 새 ID로 RENAME하며 경쟁**

02 해결 과정 · 근본 원인 분석

RedisSessionRepository.save

```
public void save(RedisSession session) {
    if (!session.isNew) {
        String key = getSessionKey(session.hasChangedSessionId() ? session.originalSessionId : session.getId());
        Boolean sessionExists = this.sessionRedisOperations.hasKey(key);
        if (sessionExists == null || !sessionExists) {
            throw new IllegalStateException("Session was invalidated");
        }
    }
}
```

RedisSession.saveChangeSessionId

```
private void saveChangeSessionId() {
    if (hasChangedSessionId()) {
        if (!this.isNew) {
            String originalSessionIdKey = getSessionKey(this.originalSessionId);
            String sessionIdKey = getSessionKey(getId());
            RedisSessionRepository.this.sessionRedisOperations.rename(originalSessionIdKey, sessionIdKey);
        }
    }
}
```

요청 상황	결과
가장 먼저 RENAME	200
다른 요청의 RENAME 이후에 세션 조회	세션 없음 → 저장 안 함 → 200
hasKey (129행) 전에 다른 요청이 RENAME	Session was invalidated
hasKey ~ RENAME (319행) 사이에 다른 요청이 RENAME	ERR no such key 구간이 짧아 가끔 발생

- 검증: 200 응답 중 Set-Cookie: SESSION 이 붙은 건 1건뿐 → 나머지 성공은 세션을 못 찾아 저장을 건너뛴 요청

02 해결 과정 · 근본 원인 분석

톰캣 메모리에서는 발생하지 않은 이유

- 톰캣: 모든 요청이 JVM 안의 **같은 세션 객체**를 공유 → ID 변경이 즉시 반영, 별도 저장 단계 없음.
- Redis: 요청마다 세션 **사본**을 읽고, 요청 끝에 `hasKey` → `rename` 으로 저장 → 원자적이지 않아 경쟁 발생

임시 세션 만료 재검토

- 원인을 알고 보니 세션 고정 보호를 끄는 선택지도 있었음.
- 그래도 임시 세션 만료를 유지
 - 문제의 원인은 보호 전략이 아니라 역할이 끝난 임시 세션이 남아 있던 것
 - 이후 세션을 쓰게 되더라도 세션 고정 보호를 유지할 수 있음.

03 성과 및 회고

성과

병렬 요청 6건 모두 성공

소셜 로그인 직후 메인 화면의 병렬 API 요청 6건이 모두 200

다중 인스턴스에서도 안정적인 세션 유지

다중 인스턴스 환경에서도 안정적으로 소셜 로그인의 임시 세션을 사용할 수 있음.

03 성과 및 회고

추후 개선 방향

인프라 변경 전 영향 범위 점검

세션 저장소를 바꾸기 전에 세션을 쓰는 곳(OAuth2 `state` 임시 세션)을 미리 정리했다면 배포 전에 발견할 수 있었음.

담당 영역 간 공유

일정상 각자 맡은 기능을 공유할 시간이 없었고, 그 결과 인증 흐름을 모르는 상태에서 장애를 맞아 소셜 로그인 구조를 공부하는 데 시간을 씀.
→ 각자 작업한 영역을 공유하는 자리가 필요

배운 점

스택 트레이스를 잘 확인하자

로그가 길게 나오면 피로부터 느꼈는데, 중요한 단서라는 것을 체감함.
아무리 난해한 오류도 단서를 잘 수집하면 설명 가능한 원인을 찾을 수 있음.

저장소를 바꾸면 동시성 조건도 바뀜

톰캣 메모리에서는 문제없던 코드가 Redis에서 경쟁 조건이 됨.
→ 외부 저장소로 옮길 때는 병렬 요청 시나리오를 함께 검증

InvestUP 주식 초보자를 위한 교육형 모의 주식 트레이딩 서비스

데이터 보존이 필요해진 시점에 Flyway 도입, 8일간 마이그레이션 17건 자동 반영

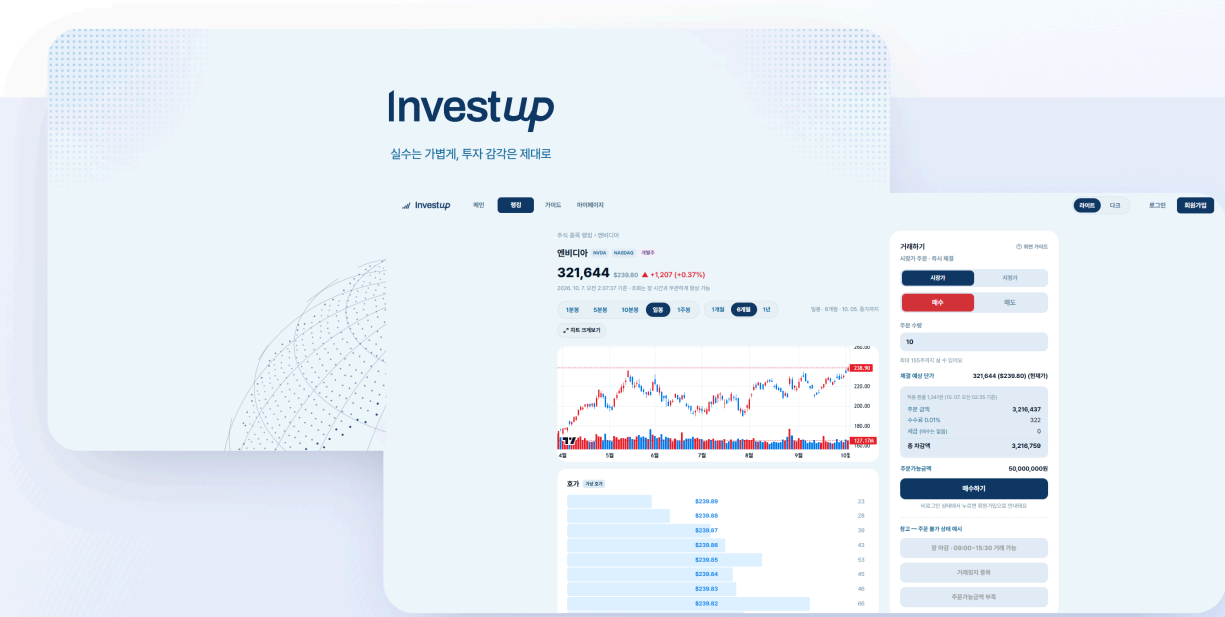
- 01 문제 정의
- 02 해결 과정
- 03 성과 및 회고

프로젝트 소개

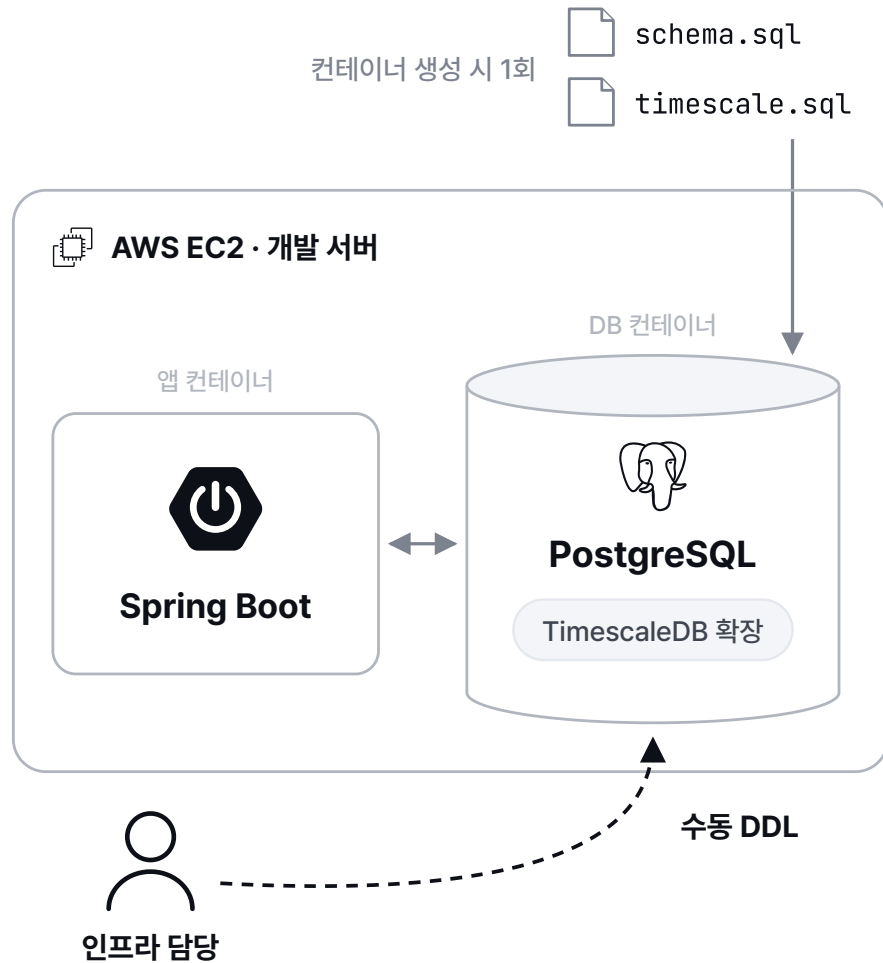
InvestUP 주식 초보자를 위한 교육형 모의 주식 트레이딩 서비스

 프로그래머스 데브코스 백엔드 10기 최종 프로젝트 최우수상(1위)

진행 기간	2026.08.21 ~ 2026.09.15 (약 4주)
역할	5인 팀 · 백엔드(외부 API 연동 · 데이터 적재), 인프라 전담
기술 스택	Java 21 · Spring Boot 3.5 · Spring Data JPA · PostgreSQL · TimescaleDB · Flyway · Docker · OCI(AWS에서 이전) · Terraform
링크	서비스 · GitHub



01 문제 정의



기존 구조

- PR 병합 시 앱은 개발 서버에 자동 배포, 스키마는 `schema.sql`, `timescale.sql` 로 관리
- 스키마 파일을 DB 컨테이너 init 스크립트로 삽입하기 때문에 변경 시 컨테이너 재생성 → 데이터 손실
- **환율 및 시세 데이터가 쌓이며 재생성 불가 → DDL로 수동 마이그레이션**

문제 상황

- 수동 DDL은 휴먼 에러로 개발 서버와 최신 브랜치의 **스키마가 어긋날 위험**이 있음.
- 변경 내용을 가장 잘 아는 작업자가 인프라 담당에게 DDL을 전달하고 **결과를 기다려야 함**.
- 서버에서 DDL을 직접 실행해 **사전 검증 없이** 바로 반영되고, 무엇을 실행했는지 **기록이 남지 않음**.
- 스키마 변경 3건이 추가로 예정 → **수동 반영이 반복될 상황**

02 해결 과정 · Flyway 도입

Flyway의 문제 해결 방식

데이터를 유지한 채 스키마 변경

컨테이너 재생성 불가 → DDL로 수동 마이그레이션

기동 시 적용되지 않은 마이그레이션만 실행 → 데이터 유지

스키마 드리프트 차단

수동 DDL 휴먼 에러로 인한 스키마 드리프트

모든 환경에 같은 마이그레이션 파일 적용

작업자가 직접 반영

인프라 담당에게 DDL 전달 후 대기

PR에 마이그레이션 파일을 넣으면 배포 시 자동 반영

사전 검증 및 기록

서버에서 직접 실행해 사전 검증·실행 기록 없음

로컬 기동으로 먼저 실행, PR 리뷰를 거쳐 반영, 실행 내역은 마이그레이션 파일 (git)과 `flyway_schema_history` 로 추적

02 해결 과정 · 도입을 위한 3단계 진행

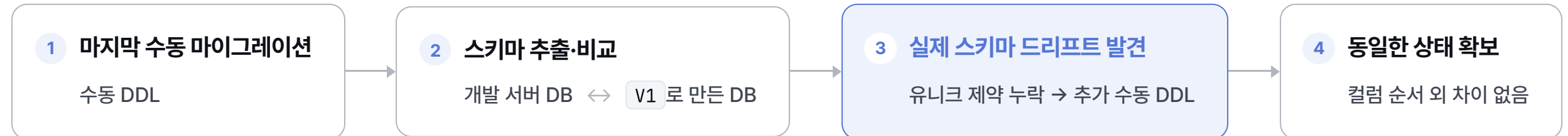


02 해결 과정 · 도입을 위한 3단계 진행

1단계

개발 서버 스키마·데이터를 최신으로 수동 마이그레이션

데이터를 보존하기 위해 기존 데이터를 최신 스키마(V1)로 맞춤.



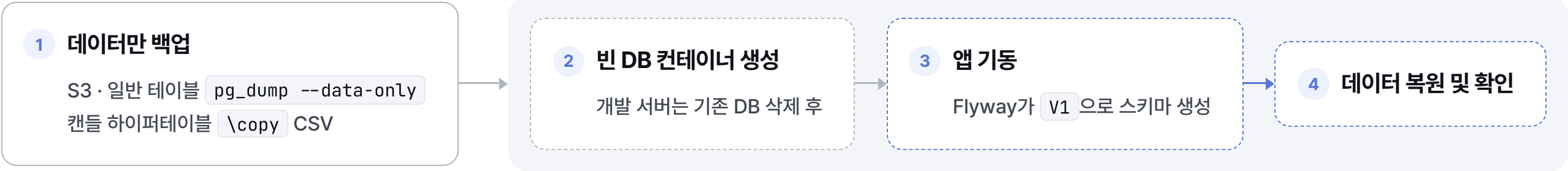
02 해결 과정 · 도입을 위한 3단계 진행

2단계

Flyway가 만든 DB에 데이터만 이관

수동 DDL로 맞춘 스키마 대신, Flyway가 처음부터 스키마를 관리하도록 V1 으로 DB 생성

별도 인스턴스 리허설 → 성공 후 개발 서버에 동일 적용



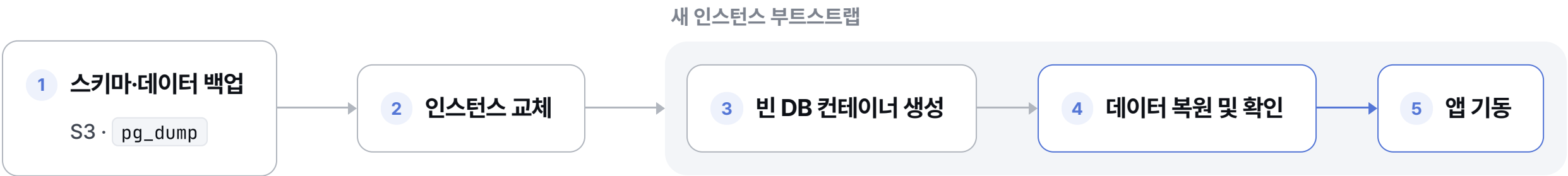
* 데이터를 먼저 넣으면 Flyway가 비어 있지 않은 DB로 판단해 실패하므로 앱 기동 후 복원

02 해결 과정 · 도입을 위한 3단계 진행

3단계

지속 가능한 백업·복원 구조

부트스트랩으로 작성해 인스턴스 교체 시 자동 복원함.



* 스키마와 함께 백업한 덤프에 `flyway_schema_history`가 포함되어, 데이터를 먼저 복원해도 Flyway가 이력을 이어받아 이후 마이그레이션만 실행

03 성과 및 회고

성과

데이터 손실 없이 이관

종목·일봉 등 약 6.5만 건의 데이터를 Flyway가 관리하는 DB로 옮기고, 이후 인스턴스 교체 시에도 자동 복원

PR만으로 스키마 반영

도입 후 8일간(9/8~9/15) 마이그레이션 17건이 수동 DDL 없이 PR 병합만으로 개발 서버에 반영

작업자가 직접 반영

팀원 5명 모두 마이그레이션을 직접 작성, 인프라 담당에게 DDL을 전달하지 않고 git으로 공유·리뷰

03 성과 및 회고

추후 개선 방향

정기 백업 자동화

지금은 인스턴스 교체 직전에만 백업해 예기치 않은 장애에 대비하지 못함.
→ cron 기반 정기 백업 (RDS는 비용 증가로 제외)

CI에서 버전 충돌 감지

병렬 작업이 늘면 같은 버전 번호나 순서 역전이 생길 수 있음
→ PR 단계에서 중복 버전·순서 역전을 검사

배운 점

도입이 늦을수록 이관 비용이 커짐

데이터가 쌓인 뒤 도입해 스키마 최신화부터 데이터 이관까지 3단계가 필요했지만, 데이터가 없을 때는 **V1** 파일 하나로 끝나는 작업
→ 다음 프로젝트는 첫 스키마부터 Flyway로 관리

수동 DDL은 기록이 남지 않음

무엇을 실행했는지 남지 않아 유니크 제약 누락 같은 드리프트를 비교해 보기 전까지 알 수 없었음.
→ 스키마 변경도 코드처럼 파일로 남기고 리뷰해야 함.

감사합니다

이메일	eric250@naver.com
전화	010-3421-6615
GitHub	github.com/puppywimy