Chap 04 데이터베이스_4.3 트랜잭션과 무결성

4.3 트랜잭션과 무결성

4.3.1 트랜잭션

▪트랜잭션이란?

데이터베이스에서 하나의 논리적 기능을 수행하기 위한 작업의 단위

여러 개의 쿼리들을 하나로 묶는 단위

특징 : ACID = 원자성, 일관성, 독립성, 지속성

▪쿼리란?

데이터베이스에 접근하는 방법

✅ 원자성 (atomicity)

- “all or noting”
- 트랜잭션과 관련된 일이 모두 수행되었거나 되지 않았거나를 보장
- 트랜잭션 단위로 여러 로직들을 묶을 때 외부 API 호출은 불가(있어도 롤백시 방안 있어야함)
- ex

  “홍철”이 “규영”에게 1000만원이 있는 통장에서 500만원을 이체하는 상황

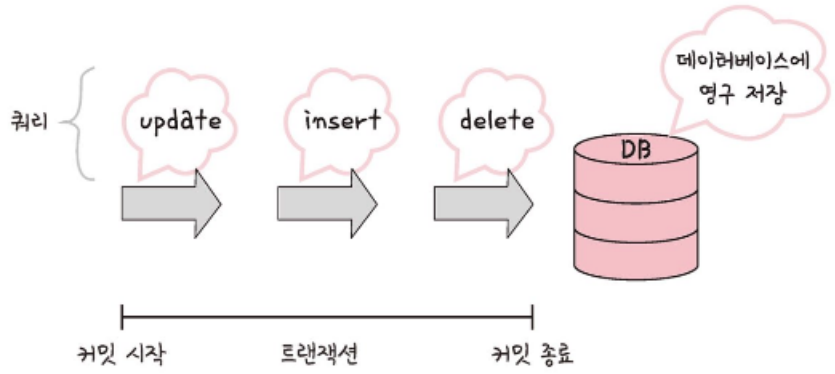
  1. 홍철의 잔고를 조회한다.
  2. 홍철에게서 500만원을 뺀다.
  3. 규영에게서 500만원을 넣는다.

  ⇒ 홍철 500, 규영 0 의 상황은 일어날 수 없다.

▪커밋

= 여러 쿼리가 성공적으로 처리되었다고 확정하는 명령어

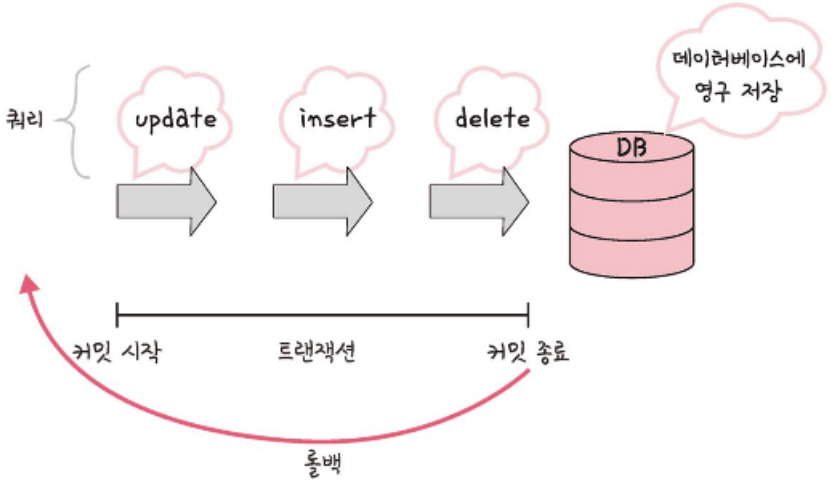
- 트랜잭션 단위로 수행되며 변경된 내용이 모두 영구적으로 저장
- “커밋이 수행 됨” = “하나의 트랜잭션이 성공적으로 수행되었다.”



▲ 그림 4-26 커밋

▪롤백

= 트랜잭션으로 처리한 하나의 묶음 과정을 일어나기 전으로 돌리는 일(취소)



▲ 그림 4-27 롤백

⇒ 커밋과 롤백으로 데이터의 무결성 보장

⇒ 데이터 변경 전 변경 사항을 쉽게 확인하고 해당 작업을 그룹화 할 수 있음

*데이터 무결성 : 데이터의 정확성과 일관성을 확인 (정확성, 일관성, 유효성 유지)

▪트랜잭션 전파

여러 트랜잭션 관련 메서드의 호출을 하나의 트랜잭션에 묶이도록 하는 것

@Transactional 애너테이션을 통해 여러 쿼리 관련 코드를 하나의 트랜잭션으로 처리

```
@Service
@Transaction(readOnly =true)
public class MemberService {
```

```
    private final MemberRepository memberRepository;
    public MemberService(MemberRepository memberRepository){
        this.memberRepository=memberRepository;
    }
}
```

✅ 일관성 (Consistency)

= "허용된 방식"으로만 데이터 변경

- 데이터베이스에 기록된 모든 데이터는 여러 가지 조건, 규칙에 따라 유효함을 가져야 함

✅ 격리성(isolation)

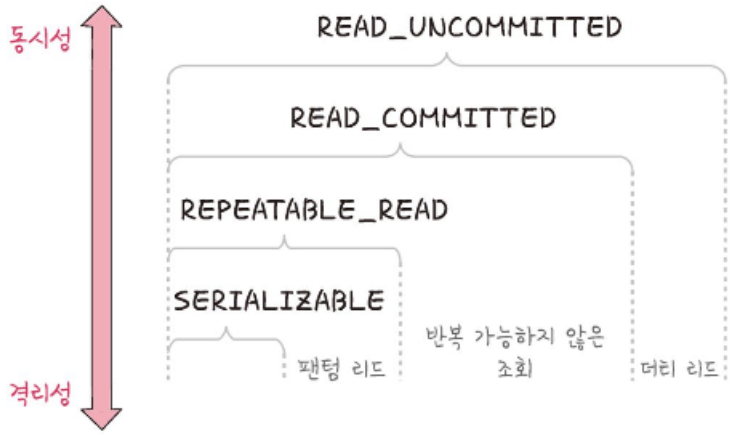
= 트랜잭션 수행 시 서로 끼어들지 못하는 것

- 복수의 병렬 트랜잭션은 서로 격리되어 마치 순차적으로 실행되는 것처럼 작동되어야 함
- DB는 여러 사용자가 같은 데이터에 접근할 수 있어야함
- 격리성은 여러 개의 격리 수준으로 나눠서 격리성 보장

  = 동시에 여러 트랜잭션이 처리될때, 트랜잭션끼리 얼마나 고립되어 있는가에 대한 수준

  = 준수한 처리 속도를 위해서는 트랜잭션의 완전한 격리가 아닌 완화된 수준의 격리 필요

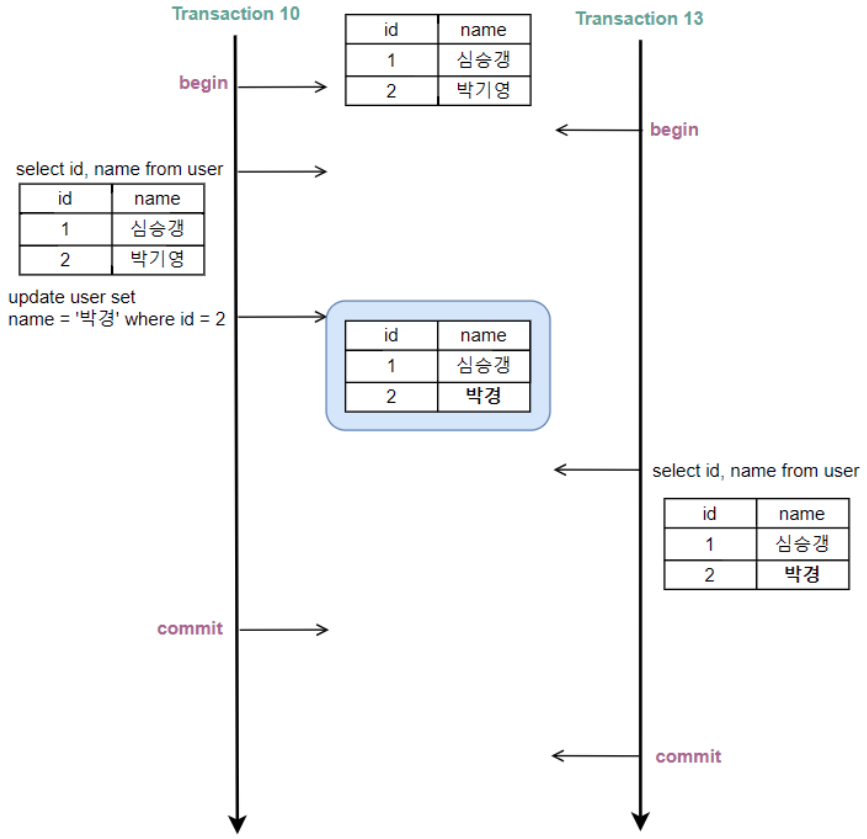
  = DBMS마다 격리 수준 내용 다름 (MySQL 기준)



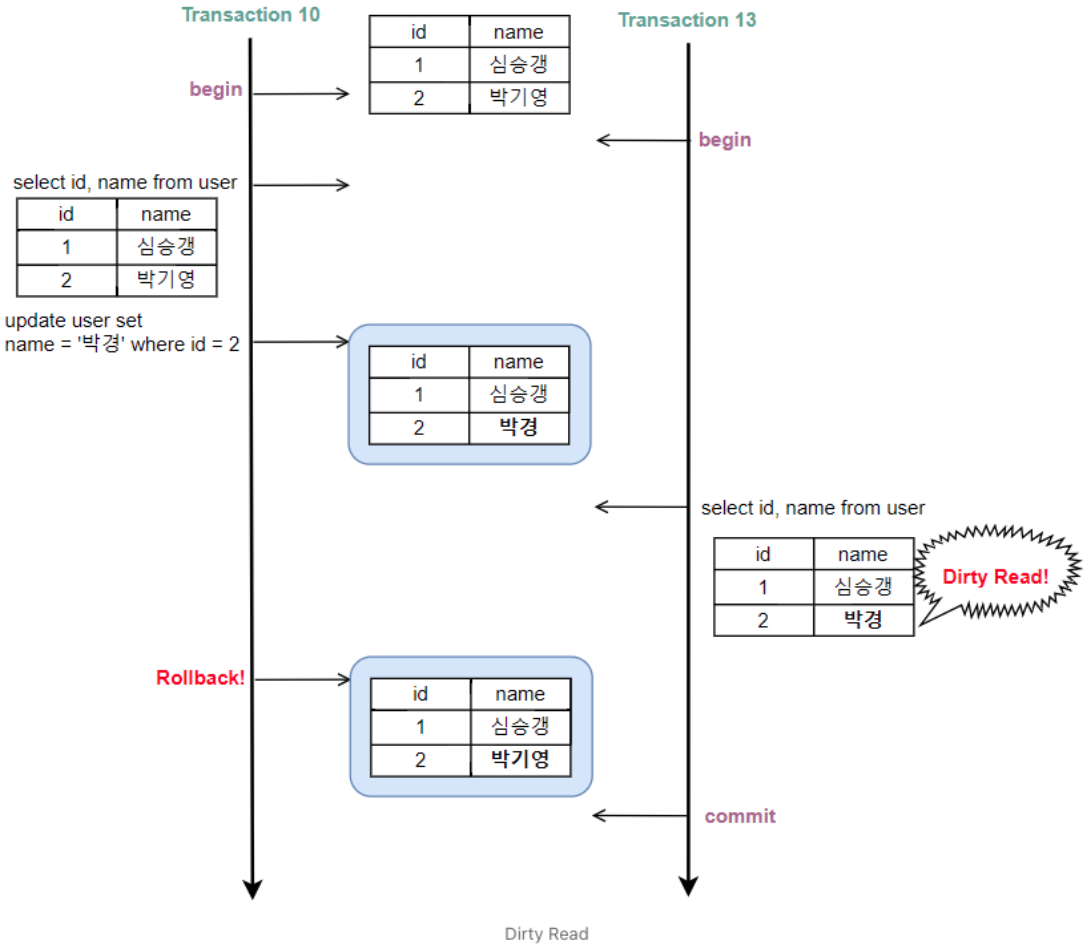
▲ 그림 4-28 여러 개의 격리 수준

1단계) Uncommitted Read

- 다른 트랜잭션에서 커밋되지 않은 데이터에 접근할 수 있게 하는 격리 수준
- 가장 저수준의 격리수준
- 일반적으로 사용x
- 데이터 부정합 발생 시킴 → oracle은 지원하지 않음
- = Dirty Read = 커밋되지 않는 트랜잭션에 접근하여 부정합을 유발할 수 있는 데이터를 읽는 것



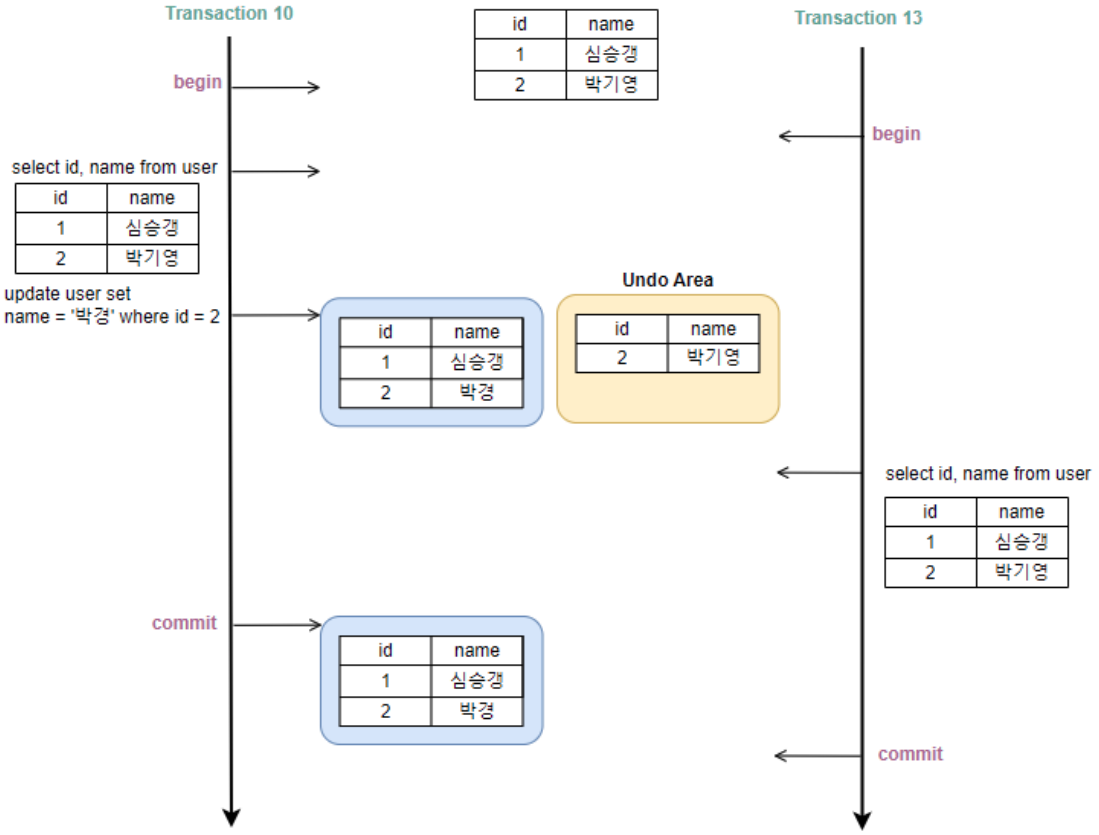
Uncommitted Read



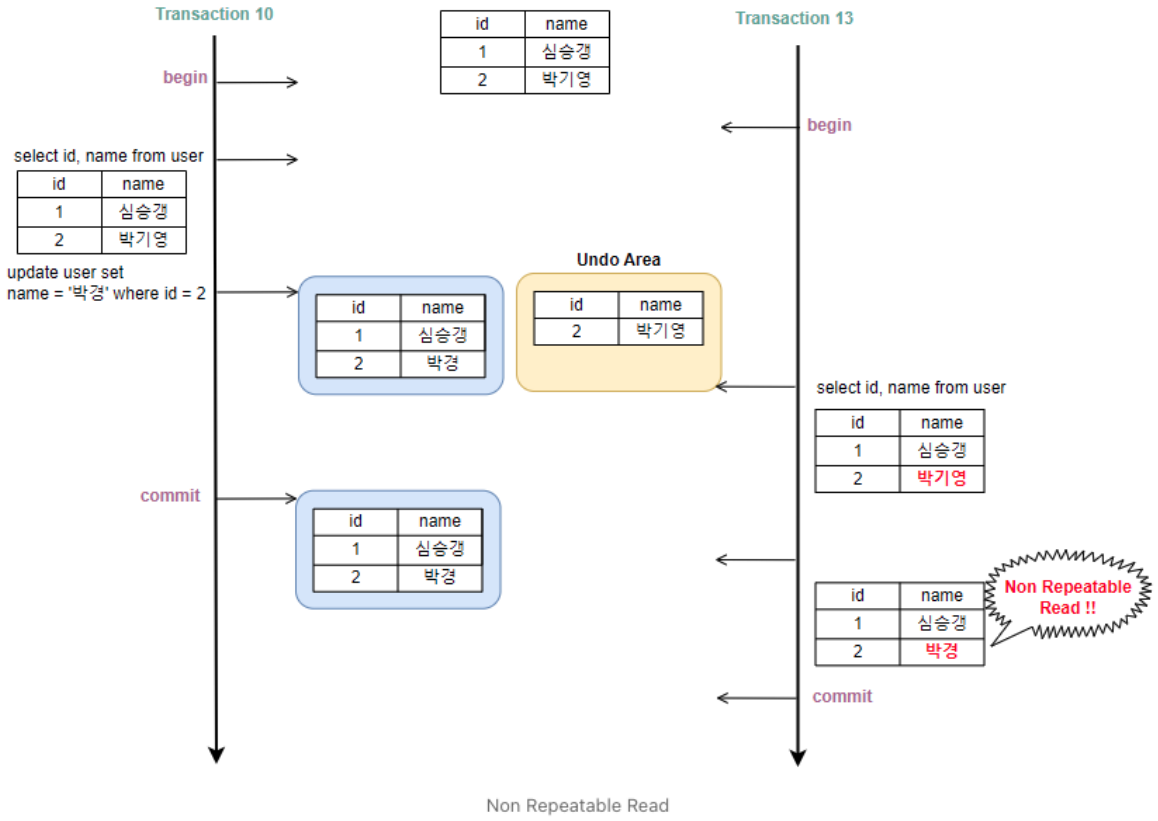
2단계) Committed Read

- 다른 트랜잭션에서 커밋된 데이터로만 접근할 수 있게 하는 격리 수준
- MySQL을 제외한 대부분 기본 격리수준으로 사용
- commit전 데이터는 undo Area에 저장
  - 트랜잭션에 대한 로그는 항상 남음

    → undo log가 undo log buffer 형태로 메모리에 저장



Committed Read



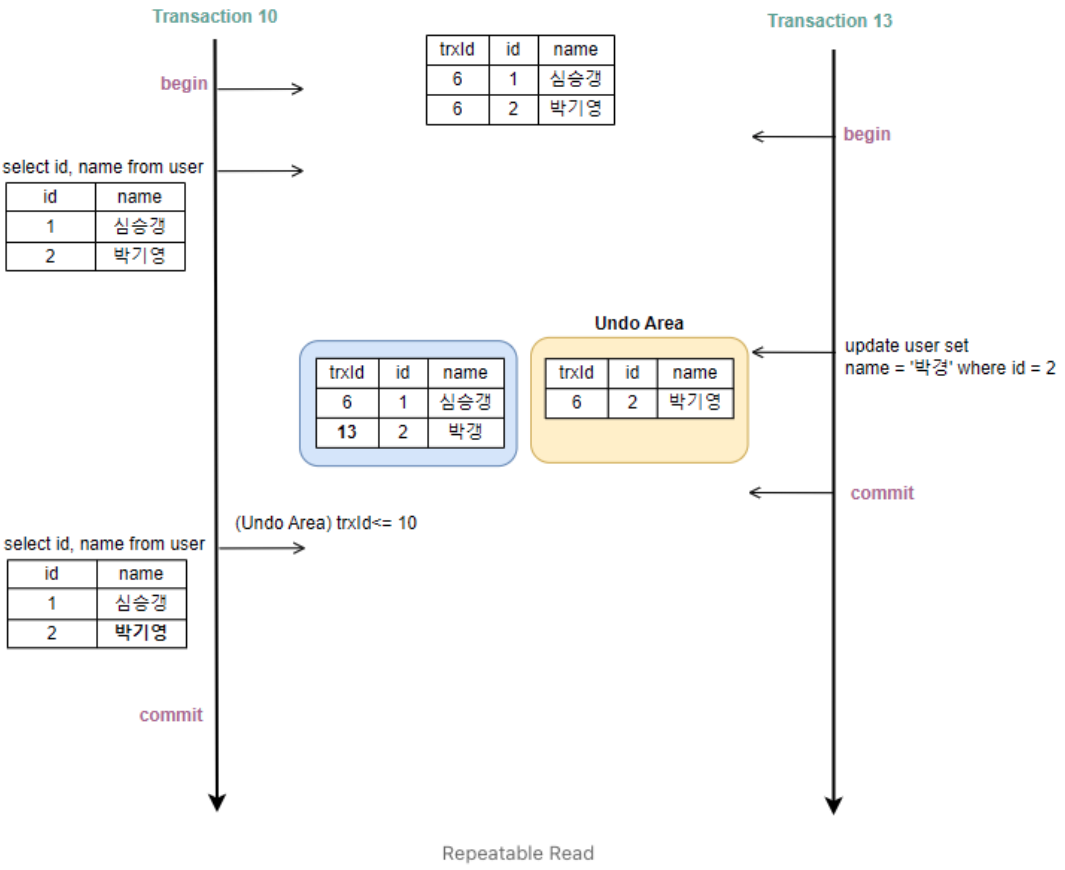
3단계) Repeatable Read

- Non repeatable read 문제를 해결하는 격리 수준
- 커밋된 데이터만 읽을 수 있되, 자신보다 낮은 트랜잭션 번호를 갖는 트랜잭션에서 커밋한 데이터만 읽을 수 있는 격리 수준
  - Undo 로그 활용
  - 트랜잭션 ID를 통해 undo 영역의 데이터를 스냅샷처럼 관리하여 동일한 데이터를 보장

    = MVCC(multi version concurrency control)

*10번 트랜잭션은 10번 보다 작은 트랜잭션에서 커밋한 데이터만 읽을 수 있음

→ 13번 트랜잭션에서 변경한 내용을 조회 불가능



4단계) Serializable

- 가장 고수준의 격리수준
- 트랜잭션을 무조건 순차적으로 진행

  → 데이터의 부정합 문제 발생 x

  → 동시 처리 불가능하므로 처리 속도 느려짐
- select 쿼리 실행 시 shared lock/ insert,update,delete 쿼리 실행시 exclusive lock 걸어버림

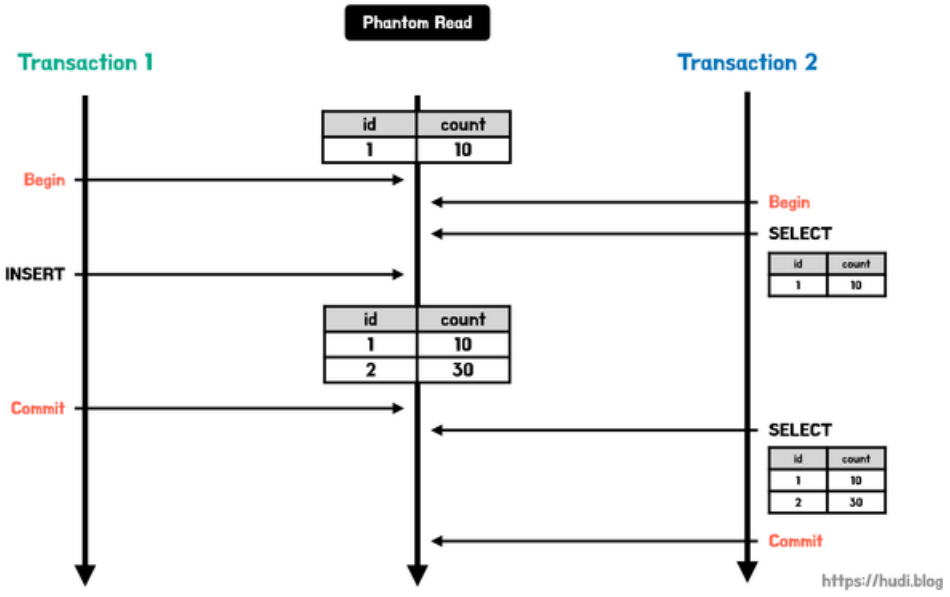
*shared lock = 읽기 작업 허용, 쓰기 작업 불가 (동시에 exclusive lock 허용x)

*exclusive lock = 특정 레코드나 테이블에 대해 다른 트랜잭션에서 읽기, 쓰기 작업을 할 수 없도록 하는 Lock

⇒ 팬텀리드 발생 가능

▪팬텀리드

한 트랜잭션 내에서 동일한 쿼리를 보냈을 때 해당 조회 결과가 다른 경우



✅ 지속성(durability)

성공적으로 수행된 트랜잭션은 영원히 반영되어야 한다.

= DB에 시스템 장애 발생해도 원래 상태로 복구하는 회복 기능이 있어야함

→ 체크섬, 저널링, 롤백 기능 제공

▪체크섬

- 중복 검사의 한 형태
- 오류 정정을 통해 송신된 자료의 무결성을 보호하는 단순한 방법

▪저널링

- 파일 시스템 또는 데이터베이스 시스템에 변경사항을 반영(commit)하기 전에 로깅하는 것
- 트랜잭션 등 변경사항에 대한 로그를 남기는 것

4.3.2 무결성

무결성이란?

데이터의 정확성, 일관성, 유효성을 유지하는 것을 말하며, 무결성이 유지되어야 데이터베이스에 저장된 데이터 값과 그 값에 해당하는 현실 세계의 실제 값이 일치하는지에 대한 신뢰가 생김

▼ 표 4-2 무결성 종류

이름	설명
개체 무결성	기본키로 선택된 필드는 빈 값을 허용하지 않습니다.
참조 무결성	서로 참조 관계에 있는 두 테이블의 데이터는 항상 일관된 값을 유지해야 합니다.
고유 무결성	특정 속성에 대해 고유한 값을 가지도록 조건이 주어진 경우 그 속성 값은 모두 고유한 값을 가집니다.
NULL 무결성	특정 속성 값에 NULL이 올 수 없다는 조건이 주어진 경우 그 속성 값은 NULL이 될 수 없다는 제약 조건입니다.