트랜잭션과 무결성

트랜잭션

데이터베이스에서 하나의 논리적 기능을 수행하기 위한 작업의 최소 단위

여러 개의 쿼리들을 하나로 묶는 단위

특징으로 ACID (원자성, 일관성, 독립성, 지속성)이 있음

원자성(Atomicity)

“All or Nothing”

트랜잭션과 관련된 일이 모두 수행되었거나 되지 않았음을 보장하는 특징

트랜잭션을 커밋 시 문제가 발생하여 롤백하는 경우, 그 이후에 모두 수행되지 않음을 보장하는 것

예시

1000만원을 가진 홍철이가 0원을 가진 규영이에게 500만원을 이체하는 경우

결과는 두 사람 모두 500만원을 가져야하며, 아래와 같은 operation 단위들로 이루어진 과정을 거침

1. 홍철의 잔고 조회
2. 홍철에게서 500만원을 뺌
3. 규영에게 500만원을 넣음

DB 사용자는 위 과정을 볼 수도, 참여할 수도 없음. 결과만 확인이 가능함

이 작업을 취소할 경우, 홍철이는 1000만원, 규영이는 0원을 가져야함

일부만 적용되어 홍철이의 500만원이 사라져서는 안됨

또한, 트랜잭션 단위로 여러 로직들을 묶을 경우, 외부 API를 호출하는 것이 있으면 안됨

롤백이 일어났을 때 어떻게 해야할 것인지에 대한 해결 방법이 있어야함

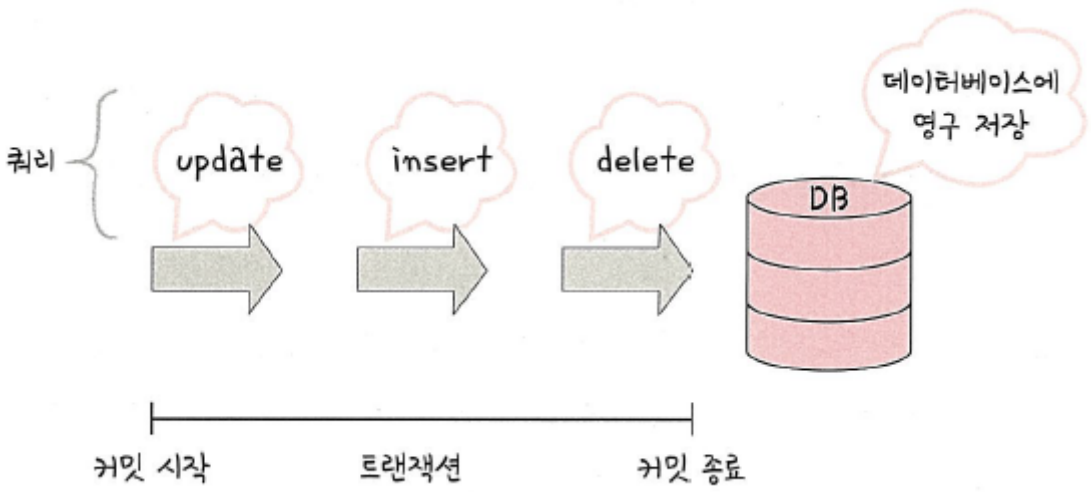
트랜잭션 전파를 신경 써서 관리해야함

커밋(commit)

여러 쿼리가 성공적으로 처리되었다고 확정하는 명령어

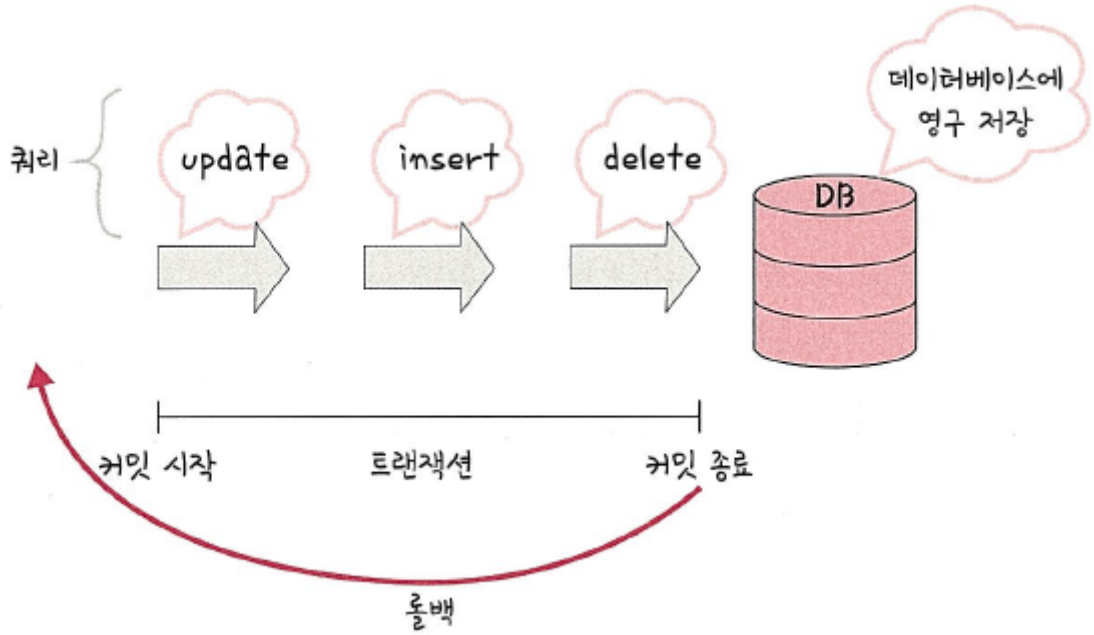
트랜잭션 단위로 수행되며 변경된 내용이 모두 영구적으로 저장되는 것

“커밋이 수행되었다.” 를 “하나의 트랜잭션이 성공적으로 수행되었다.” 라고도 말함



위 그림과 같이 update, insert, delete 쿼리가 하나의 트랜잭션 단위로 수행되고, 이후에 DB에 영구 저장됨

롤백(rollback)



에러나 여러 이슈 때문에 트랜잭션 전으로 돌려야 하는 경우 사용

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

커밋과 롤백으로 인해 데이터 무결성이 보장됨

데이터 변경전에 변경사항을 쉽게 확인 할 수 있고, 해당 작업을 그룹화 할 수 있음

트랜잭션 전파

트랜잭션 수행 시 커넥션 단위로 수행하므로 커넥션 객체를 넘겨서 수행해야함

이를 매번 넘겨주기가 어렵고 귀찮으므로, 이를 넘겨서 수행하지 않고 여러 트랜잭션 관련 메소드의 호출을 하나의 트랜잭션에 묶이도록 하는 것

ex) @Transactional 어노테이션을 통해 여러 쿼리 관련 코드들을 하나의 트랜잭션으로 처리

일관성 (Consistency)

허용된 방식으로만 데이터를 변경해야 하는 것

DB에 기록된 모든 데이터는 여러가지 조건, 규칙에 따라 유효함을 가져야함

예시

홍철이는 1000만원이 있고 범석이는 0원이 있는 경우

범석이가 홍철이에게 500만원을 입금하는 것은 불가능함

격리성 (Isolation)

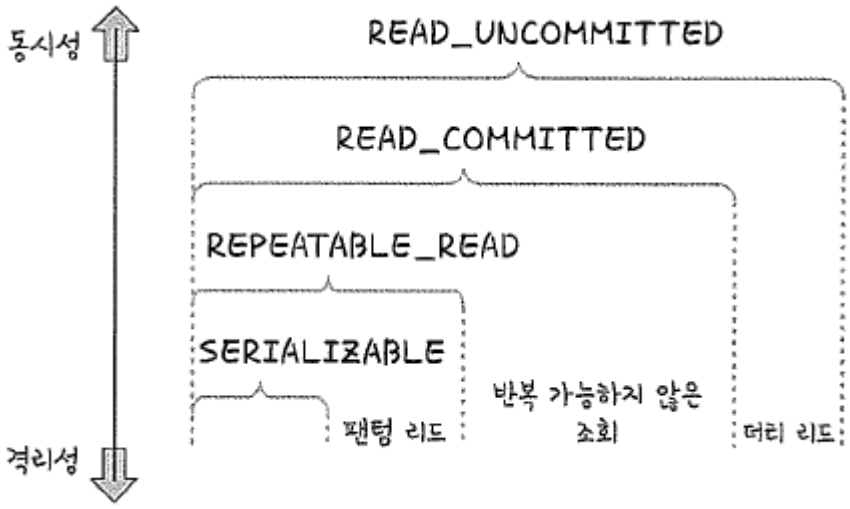
트랜잭션 수행 시 서로 끼어들지 못하는 것

복수의 병렬 트랜잭션은 서로 격리되어 순차적으로 실행되는 것처럼 작동되어야함

DB는 여러 사용자가 같은 데이터에 접근할 수 있어야함

순차적으로 하면 쉽게 할 수 있으나, 이는 성능이 안 좋음

격리성은 여러 개의 격리 수준으로 나뉘어 격리성을 보장함



격리 수준에 따라 발생하는 현상

팬텀 리드 (phantom read)

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

사용자 A가 회원 테이블에서 age 가 12 이상인 회원들을 조회하는 쿼리를 보내는 경우

이 결과로 세 개의 행이 조회된다고 가정

사용자 B가 age 가 15인 회원 레코드를 삽입한 경우, 세 개가 아닌 네 개의 행이 조회됨

반복 가능하지 않은 조회 (non-repeatable read)

한 트랜잭션 내의 같은 행에 두 번 이상 조회가 발생했는데, 그 값이 다른 경우

사용자가 A가 큰돌의 보석 개수가 100개라는 값을 가진 데이터였는데, 그 이후 사용자 B가 그 값을 1로 변경해서 커밋했다고 하면, 사용자는 100이 아닌 1을 읽게 됨

팬텀 리드와 차이점: 반복 가능하지 않은 조회는 행 값이 달라질 수도 있는데, 팬텀 리드는 다른 행이 선택될 수도 있다는 것을 의미함

더티 리드 (dirty read)

반복 가능하지 않은 조회와 유사

한 트랜잭션이 실행 중일 때 다른 트랜잭션에 의해 수정되었지만 아직 커밋되지 않은 행의 데이터를 읽을 수 있을 때 발생함

사용자 A가 큰돌의 보석 개수 100을 1로 변경한 내용이 커밋되지 않은 상태라도 그 이후 사용자 B가 조회한 결과가 1로 나오는 경우

격리 수준

SERIALIAZABLE

트랜잭션을 순차적으로 진행시키는 것

여러 트랜잭션이 동시에 같은 행에 접근할 수 없음

매우 엄격한 수준으로 해당 행에 대해 격리시키고, 이후 트랜잭션이 이 행에 대해 일어난다면 기다려야함 ⇒ 교착 상태가 일어날 확률도 많고 가장 성능이 떨어지는 격리 수준

REPEATABLE_READ

하나의 트랜잭션이 수정한 행을 다른 트랜잭션이 수정할 수 없도록 막아주나, 새로운 행을 추가하는 것은 막지 않음

따라서 이후에 추가된 행이 발견될 수도 있음

READ_COMMITED

가장 많이 사용되는 격리 수준. 대부분의 DBMS 에서 기본값으로 설정되어 있음

다른 트랜잭션이 커밋하지 않은 정보는 읽을 수 없음

즉, 커밋 완료된 데이터에 대해서만 조회를 허용

그러나 어떤 트랜잭션이 접근한 행을 다른 트랜잭션이 수정할 수도 있으므로, 같은 행을 다시 읽을 때 다른 내용이 발견될 수 있음

READ_UNCOMMITED

가장 낮은 격리 수준으로, 하나의 트랜잭션이 커밋되기 이전에 다른 트랜잭션에 노출되는 문제가 있으나, 갖아 빠름

데이터 무결성을 위해 되도록이면 사용하지 않는 것이 이상적

몇몇 행이 제대로 조회되지 않더라도 괜찮은 거대한 양의 데이터를 어림잡아 집계하는 데에는 사용하면 좋음

지속성 (Durability)

성공적으로 수행된 트랜잭션은 영구적으로 반영되어야하는 것을 의미함

DB에 시스템 장애가 발생해도 원래 상태로 복구하는 회복기능이 있어야 하며, DB는 이를 위해 체크섬, 저널링, 롤백 등의 기능을 제공함

체크섬 (checksum)

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

저널링 (journaling)

파일 시스템이나 데이터베이스 시스템에 변경 사항을 반영(commit)하기 전에 로깅하는 것

트랜잭션 등 변경사항에 대한 로그를 남기는 것

무결성

데이터의 정확성, 일관성, 유효성을 유지하는 것

무결성이 유지되어야 DB에 저장된 데이터 값과 그 값에 해당하는 현실 세계의 실제 값이 일치하는지에 대한 신뢰가 생김

무결성의 종류

- **개체 무결성**

  기본키로 선택된 필드는 빈 값을 허용하지 않음

- 참조 무결성

  서로 참조 관계에 있는 두 테이블의 데이터는 항상 일관된 값을 유지해야함

- 고유 무결성

  특정 속성에 대해 고유한 값을 가지도록 조건이 주어진 경우 그 속성 값은 모두 고유한 값을 가짐

- NULL 무결성

  특정 속성 값에 NULL이 올 수 없다는 조건이 주어진 경우, 그 속성 값은 NULL이 될 수 없음