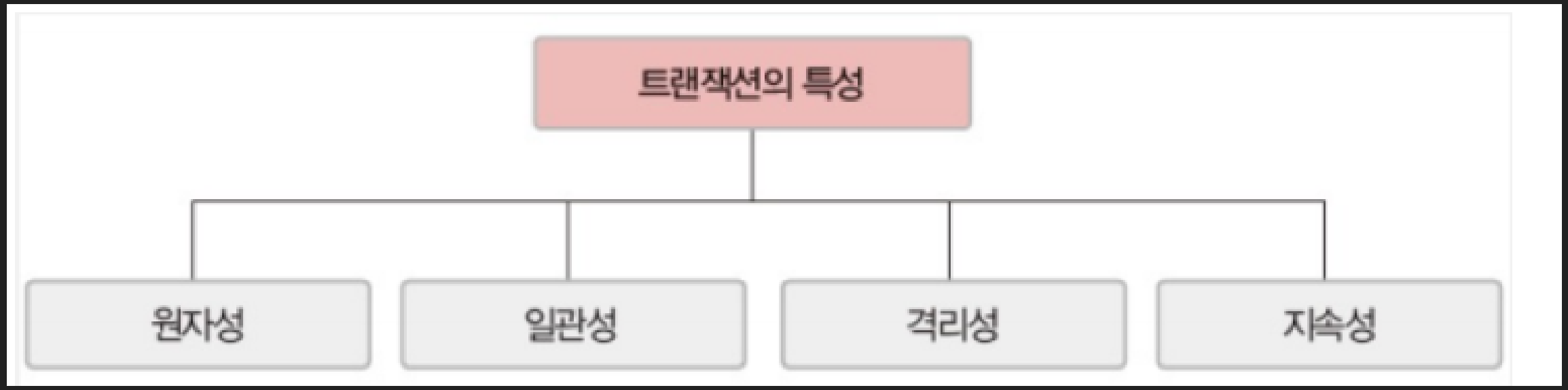
Transaction

Transaction?

- 하나의 작업을 수행하는 데 필요한 데이터베이스 연산들을 모아 놓은 것
- 작업 수행에 필요한 SQL문들의 모임
- 논리적인 작업의 단위
- 장애 발생 시 복구 작업이나 병행 제어 작업을 위한 중요한 단위
- 데이터베이스의 무결성과 일관성을 보장하기 위해 꼭 필요한 개념


계좌이체 트랜잭션
❶ 성호 계좌에서 5,000원 인출
UPDATE 계좌
SET 잔액 = 잔액 - 5000
WHERE 계좌번호 = 100;
❷ 은경 계좌에 5,000원 입금
UPDATE 계좌
SET 잔액 = 잔액 + 5000
WHERE 계좌번호 = 200;
성호 잔액 : 10,000원
은경 잔액 : 0원
계좌이체 전의
데이터베이스 상태
성호 잔액 : 5,000원
은경 잔액 : 5,000원
계좌이체 후의
데이터베이스 상태

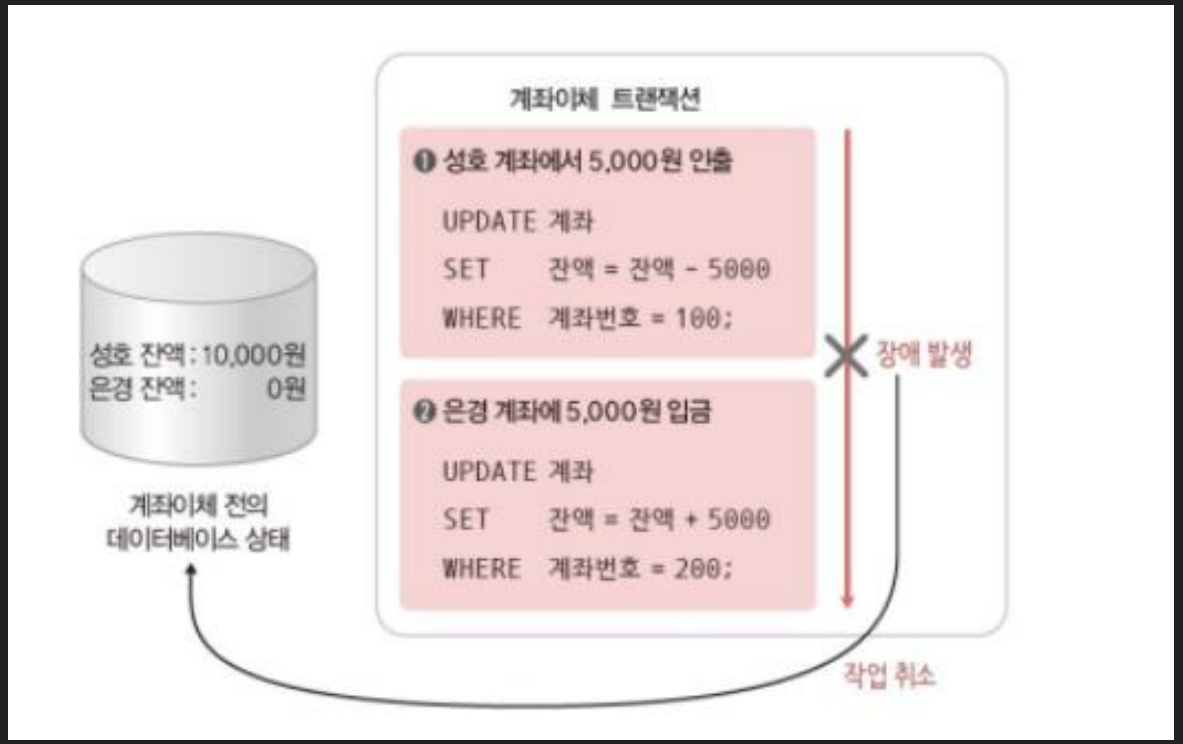
트랜잭션의 예


트랜잭션의 특성
원자성
일관성
격리성
지속성

Atomicity(원자성)

- 트랜잭션의 연산들이 모두 정상적으로 실행되거나 하나도 실행되지 않아야 하는 all-or-nothing 방식

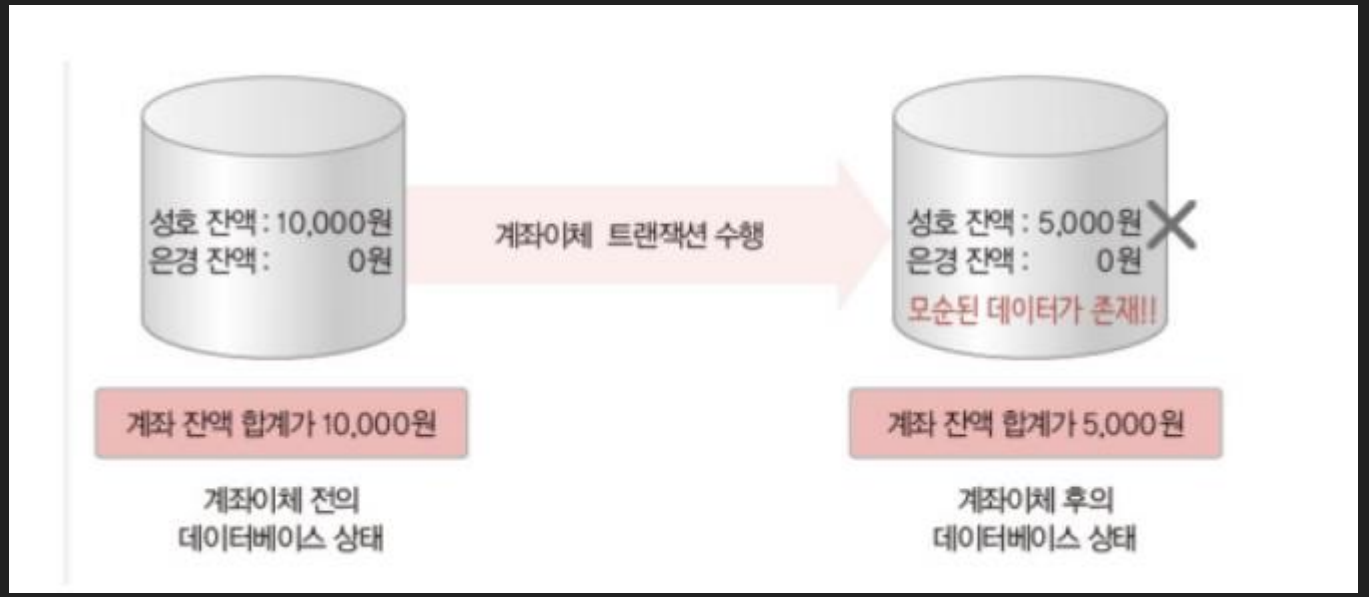
- 트랜잭션 도중에 장애가 발생한다면, 실행한 연산 처리를 모두 취소하고, 데이터베이스를 원래 상태(작업 전 상태)로 되돌려야 한다. rollback!!



* rollback
트랜잭션을 수행하는 데 실패했음을 선언(작업 취소)

Consistency(일관성)

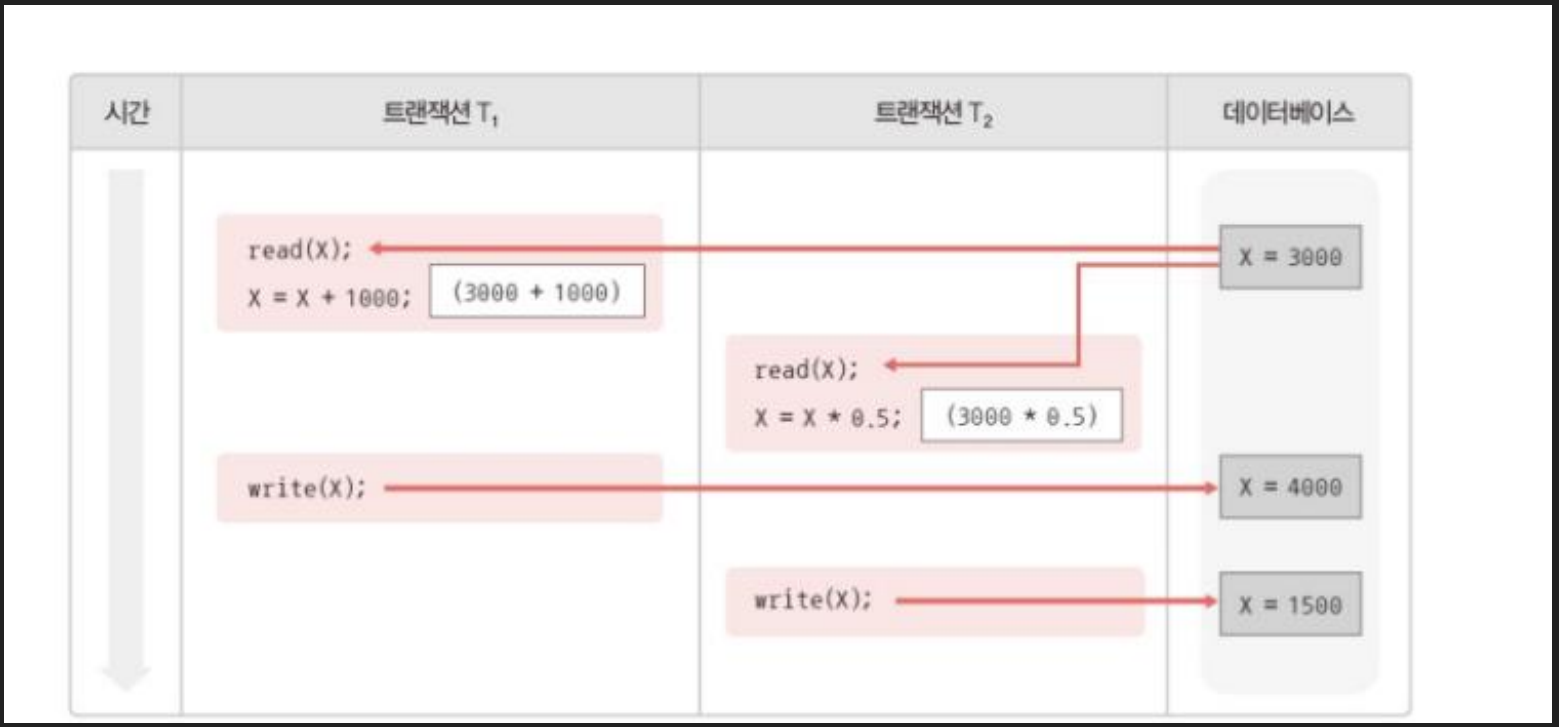
- 트랜잭션이 성공적으로 수행된 후에도 데이터베이스가 일관된 상태를 유지해야한다.
- 트랜잭션 수행 전, 후에 데이터 모델의 모든 제약 조건(기본키, 외래키, 도메인, 제약조건 등)을 만족하는 것을 통해 보장



Isolation(격리성)

- 수행 중인 트랜잭션이 완료될 때까지 다른 트랜잭션이 중간 연산 결과에 접근할 수 없다.
- 격리성의 보장을 위해 여러 트랜잭션이 동시에 수행되더라도 마치 순서대로 하나씩 수행되는 것처럼 일관된 결과를 얻을 수 있도록 제어해야 한다.

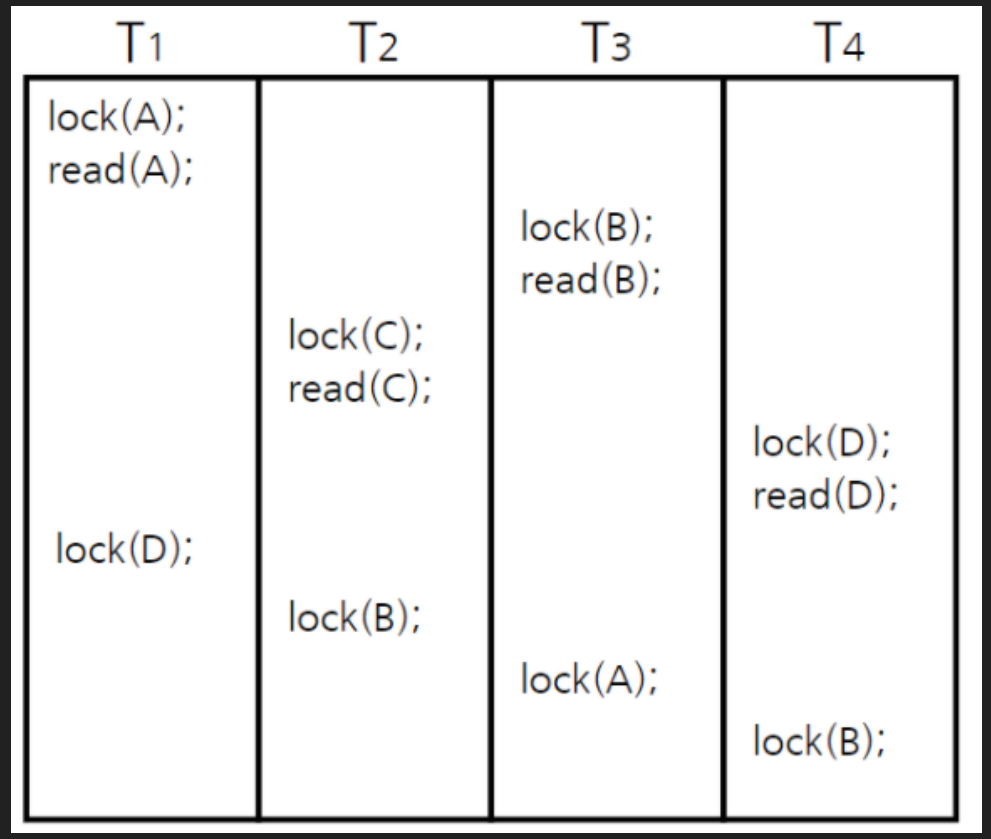
  제어를 하지 않는다면 어떤 문제가 발생할까?



https://codingram.tistory.com/88?category=926088

결국엔, 트랜잭션을 병행 제어하는 것이 중요!

로킹 기법 : 한 트랜잭션이 먼저 접근한 데이터에 대한 연산을 끝낼 때까지는 다른 트랜잭션이 그 데이터에 접근하지 못하도록 상호 배제한다.



lock : 트랜잭션이 데이터에 대한 독점권을 요청하는 연산

unlock : 트랜잭션이 데이터에 대한 독점권을 반환하는 연산

단점 : deadlock

다른 트랜잭션이 독점하고 있는 데이터에 unlock을 기다리면서 트랜잭션의 수행을 중단하고 있는 상태가 발생.

두 개 이상의 트랜잭션이 특정 자원의 lock을 획득한 채로 다른 트랜잭션이 소유하고 있는 lock을 요구하면 순환 상태가 발생.

2PL 프로토콜 (2 Phase Locking Protocol)

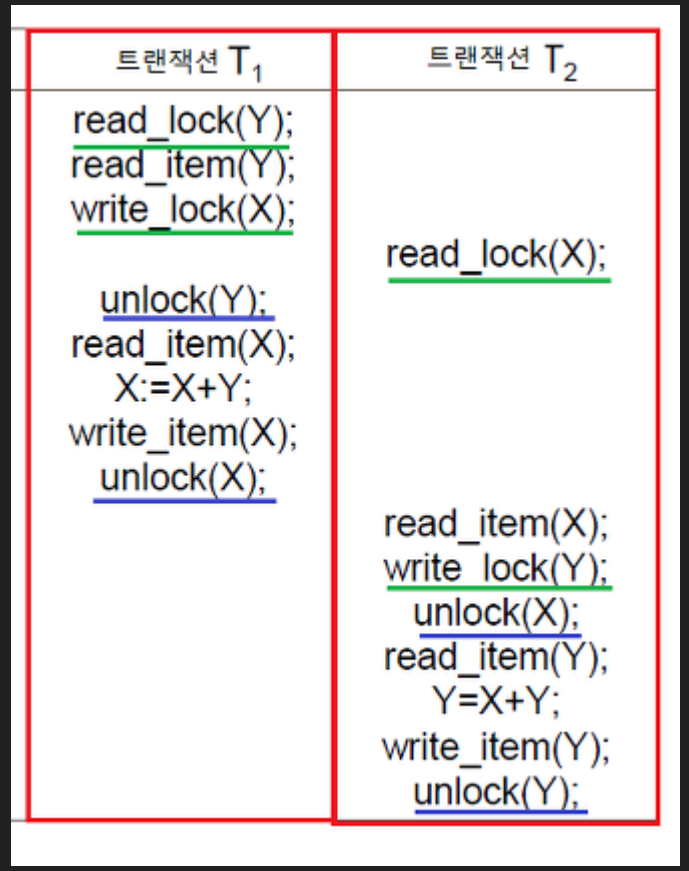
2가지 단계의 lockng이 존재한다.

- Growing phase(상승 단계)는 lock에 해당

- Shrinking phase(하강 단계)는 unlock에 해당

2PL 프로토콜은 상승 단계와 하강 단계가 섞이면 안된다는 것을 의미한다.

즉, lock이 쭉 수행된 후에 unlock이 수행되어야 한다는 것.



병행 수행을 제어하는 또 다른 방법

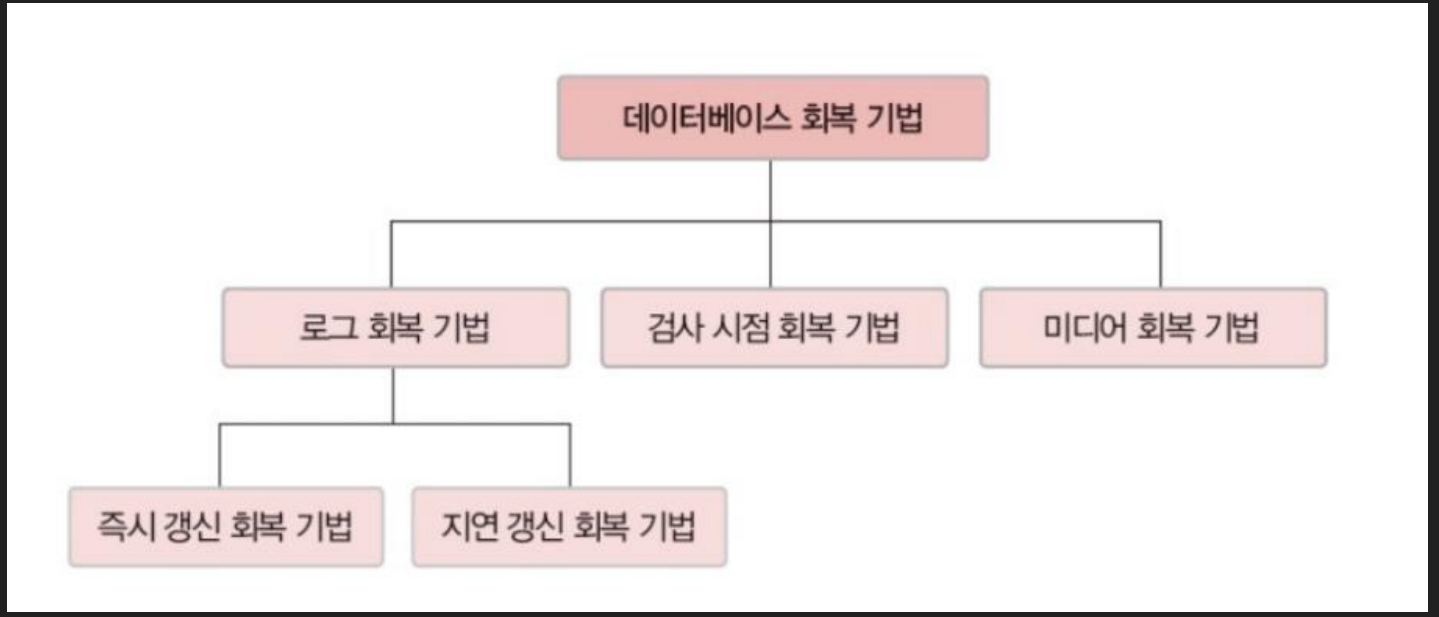
트랜잭션 스케쥴 : 트랜잭션에 포함되어 있는 연산들을 수행하는 순서

트랜잭션 스케줄	의미
직렬 스케줄	인터리빙 방식을 이용하지 않고 각 트랜잭션별로 연산들을 순차적으로 실행시키는 것
비직렬 스케줄	인터리빙 방식을 이용하여 트랜잭션들을 병행해서 수행시키는 것
직렬 가능 스케줄	직렬 스케줄과 같이 정확한 결과를 생성하는 비직렬 스케줄

인터리빙 : 인접한 메모리 위치를 서로 다른 메모리 뱅크에 둠으로써 동시에 여러 곳을 접근할 수 있게 하는 방법 , < https://codingram.tistory.com/88?category=926088 >

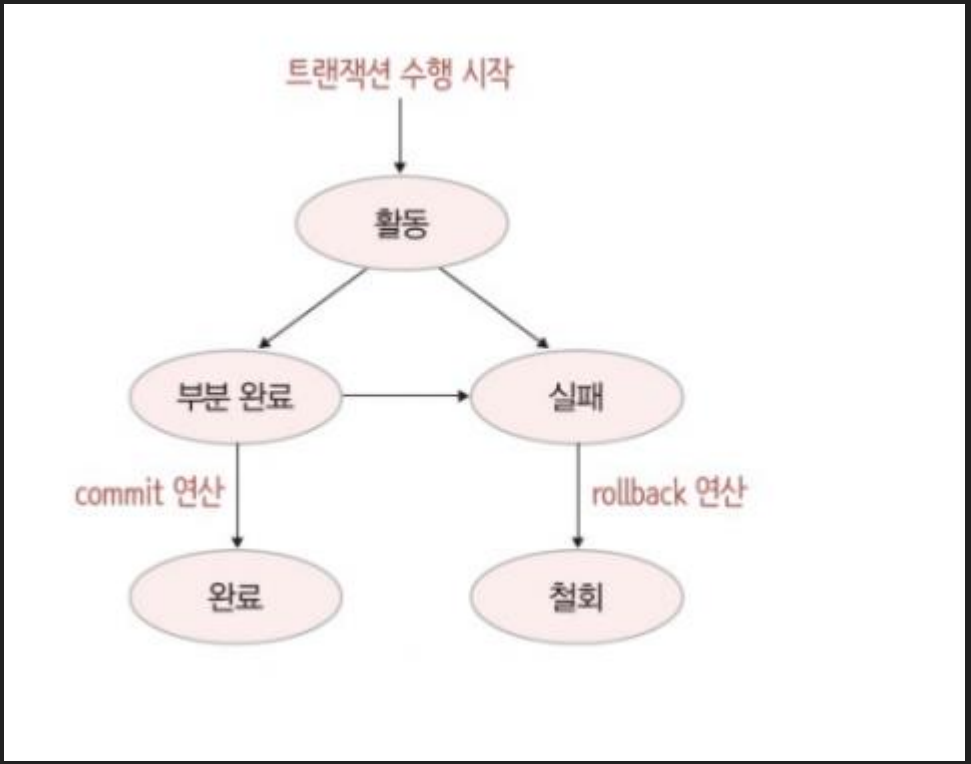
Durability(지속성)

- 트랜잭션이 성공적으로 완료된 후 데이터베이스에 반영한 수행 결과는 영구적이어야 한다.
- 지속성의 보장을 위해서는 장애 발생 시 회복 기능이 반드시 필요하다.



회복 기능 : 장애가 발생했을 때 데이터베이스를 장애가 발생하기 전의 일관된 상태로 복구 시키는 기능 (https://codingram.tistory.com/87?category=926088)

트랜잭션의 상태



- 활동 : 트랜잭션이 수행되기 시작하여 수행중인 상태
- 부분 완료 : 트랜잭션의 마지막 연산이 실행된 직후 상태, commit 연산 전
- 완료 : 트랜잭션이 성공적으로 완료되어 커밋 연산을 실행한 상태, 최종 결과를 데이터베이스에 반영함.
- 실패 : 장애가 발생하여 트랜잭션의 수행이 중단된 상태
- 철회 : 트랜잭션의 수행 실패로 롤백 연산을 실행한 상태, 데이터베이스 상태를 원래대로 되돌리고 트랜잭션 종료