트랜잭션

Transaction이란 무엇이며, 왜 중요한가요?

트랜잭션이란, 데이터베이스에서 하나의 논리적인 작업 단위를 의미합니다. 일련의 작업이 모두 성공하거나, 하나라도 실패하면 전체가 롤백되는 원자성을 보장하는 개념입니다.

트랜잭션은 데이터의 일관성과 무결성을 유지하는데 중요한 역할을 합니다. 예를 들어, 은행 계좌 이체 시, 송금 계좌에서 돈이 출금되고 수신 계좌에 입금되는 과정이 하나의 트랜잭션으로 처리되지 않으면, 일부만 반영되어 데이터 불일치가 발생할 수 있습니다.

ACID

ACID는 트랜잭션이 안전하게 수행되기 위한 4가지 핵심 속성을 의미합니다.

- 원자성(Atomicity): 트랜잭션 내 작업이 하나의 단위로 수행되며 하나라도 실패하면 전체가 롤백된다.
- 일관성(Consistency): 트랜잭션 실행 전 후에 데이터가 항상 일관된, 무결한 상태가 유지되어야 한다. (예를 들면, 100만원 송금하면 100만원이 입금되어 전체적으로 총량이 변화가 없어야 한다.)
- 격리성(Isolation): 동시에 실행되는 트랜잭션들이 서로 영향을 주지 않도록 보장해야한다.
- 지속성(Durability): 트랜잭션이 성공적으로 완료되면, 시스템에 장애가 발생하더라도 결과가 유지되어야 한다.

트랜잭션의 기본 연산인 커밋(Commit)과 롤백(Rollback)에 대해 설명해주세요

Commit

트랜잭션이 성공적으로 완료되었음을 데이터베이스에 확정하는 연산. commit 이우헤는 변경사항이 영구적으로 반영됩니다.

Rollback

트랜잭션 수행 중 오류가 발생하거나 문제가 생겼을 때, 이전 상태로 되돌리는 연산입니다.

세이브포인트(Savepoint)란 무엇이며 어떤 상황에서 사용하나요?

세이프 포인트는 트랜잭션 내에서 특정 시점에 저장점을 설정하여, 필요 시 해당 점까지 롤백할 수 있도록 하는 기능입니다.

ex) 여러개의 계좌로 송금하는 트랜잭션을 수행할 때, 일부 계좌의 송금이 실패했을 때 전체 송금을 롤백하는 것이 아니라 실패한 부분만 롤백하고 나머지는 유지할 수 있도록 세이브 포인트를 설정할 수 있습니다.

트랜잭션 격리 수준이란 무엇인가요?

트랜잭션 격리 수준은 여러 트랜잭션이 동시에 실행될 때, 각 트랜지션이 서로에게 미치는 영향을 조정하는 설정입니다. SQL에서는 4가지 격리 수준을 정의하고 있으며, 낮은 수준일수록 성능이 향상되지만, 데이터 일관성 문제가 발생할 수 있습니다.

- Read Uncommitted: 다른 트랜잭션의 미확정 데이터를 읽을 수 있음. => 더티 리드 발생 가능
- Read Committed: 다른 트랜잭션이 커밋한 데이터만 읽을 수 있음. => 더티 리드 방지
- Repeatable Read: 하나의 트랜잭션이 수행되는 동안 동일한 데이터를 여러 번 조회하면 항상 같은 결과가 보장됨. => 반복 불가능한 읽기 방지
- Serializable: 가장 엄격한 격리 수준으로 트랜잭션이 순차적으로 수행되는 것처럼 보장됨. => 팬텀 리드 방지, 성능 저하 가능성 존재.

더티 리드(Dirty Read)란 무엇인지 설명하고 예시를 들어주세요.

더티 리드는 하나의 트랜잭션에서 아직 커밋되지 않은 다른 트랜잭션의 데이터를 읽는 현상이다.

이후에 해당 트랜잭션이 롤백되면 읽은 데이터는 유효하지 않은 값이 될 수 있다.

예를들면,

1. A 트랜잭션이 계좌 잔액을 100만원에서 50만원으로 변경했지만 아직 커밋하지 않은 상황에서
2. B 트랜잭션이 계좌 잔액을 조회했을 때 50만원으로 조회를 하고,
3. 이후 A 트랜잭션이 롤백되면 잔액은 다시 100만원이 되지만, B는 잘못된 정보를 사용했을 수 있음.

=> 해결방법: Read Committed 이상의 격리 수준을 적용하여 커밋된 데이터만 읽도록 설정한다.

반복 불가능한 읽기(Non-repeatable Read)란 무엇인가요?

반복 불가능한 읽기는 하나의 트랜잭션에서 **같은 데이터를 여러 번 읽을 때**, 중간에 다른 트랜잭션이 값을 변경하여 서로 다른 결과가 나오는 문제이다. => 기존 데이터의 값이 변경되거나 삭제되어 **다른 조회 결과**가 나오는 것(DELETE, UPDATE)

예를 들면,

1. A 트랜잭션이 고객의 계좌 잔액을 조회했을 때 100만원이었는데
2. B 트랜잭션이 같은 계좌의 잔액을 50만원으로 변경하고 커밋했다면,
3. A 트랜잭션이 다시 같은 계좌를 조회했을 때 50만원으로 바뀌어 조회되게 된다.

=> 해결방법: Repeatable Read 이상의 격리 수준을 적용하여 동일 트랜잭션 내에서는 데이터가 변경되지 않도록 보장한다.

팬텀 리드(Phantom Read)란 무엇인가요?

팬텀리드는 한 트랜잭션이 **특정 조건으로 데이터를 조회**한 후, 다른 트랜잭션이 새로운 데이터를 추가하거나 삭제하여 같은 조회 시 결과가 달라지는 현상이다. => 기존 데이터의 값이 추가되거나 삭제되어 조회 결과 행 **개수**가 달라지는 것 (DELETE,

INSERT)

=> 해결 방법: Serializable 수준의 격리 수준을 적용한다.
