트랜잭션의 격리수준(isolation Level)이란?

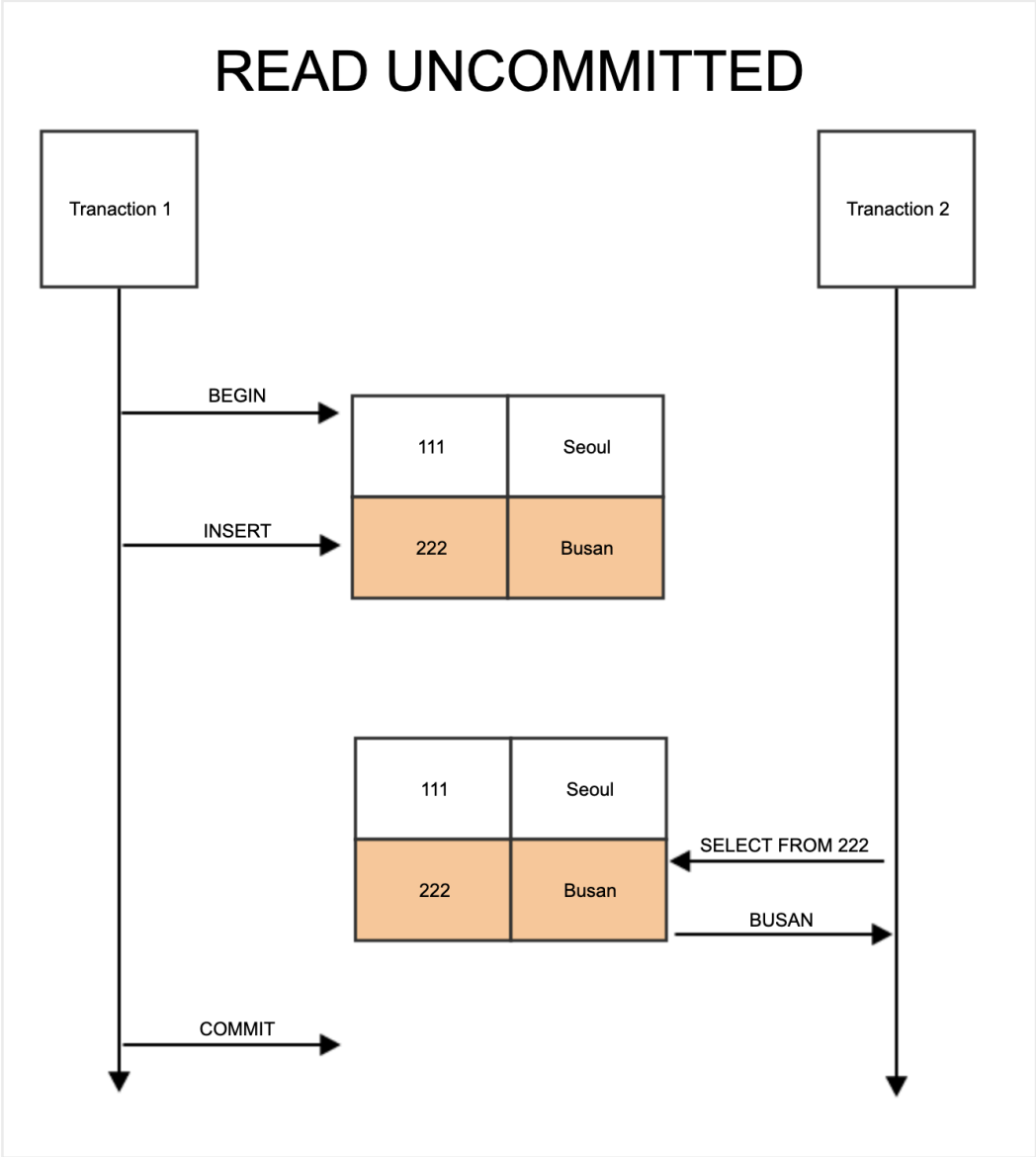
트랜잭션의 격리수준이란 동시에 여러 트랜잭션이 처리될 때, 특정 트랜잭션이 다른 트랜잭션에서 변경하거나 조회하는 데이터를 볼 수 있도록 허용할지 말지를 결정하는 것입니다.

트랜잭션의 격리수준의 종류

- READ UNCOMMITTED
- READ COMMITTED
- REPEATABLE READ
- SERIALIZABLE

READ UNCOMMITTED

- 각 트랜잭션에서의 변경 내용이 Commit이나 Rollback 여부에 상관 없이 다른 트랜잭션에서 값을 읽을 수 있습니다.
- 정합성에 문제가 많은 격리 수준이기 때문에 사용하지 않는 것을 권장합니다.
- 아래의 그림과 같이 Commit이 되지 않는 상태지만 Update된 값을 다른 트랜잭션에서 읽을 수 있습니다.



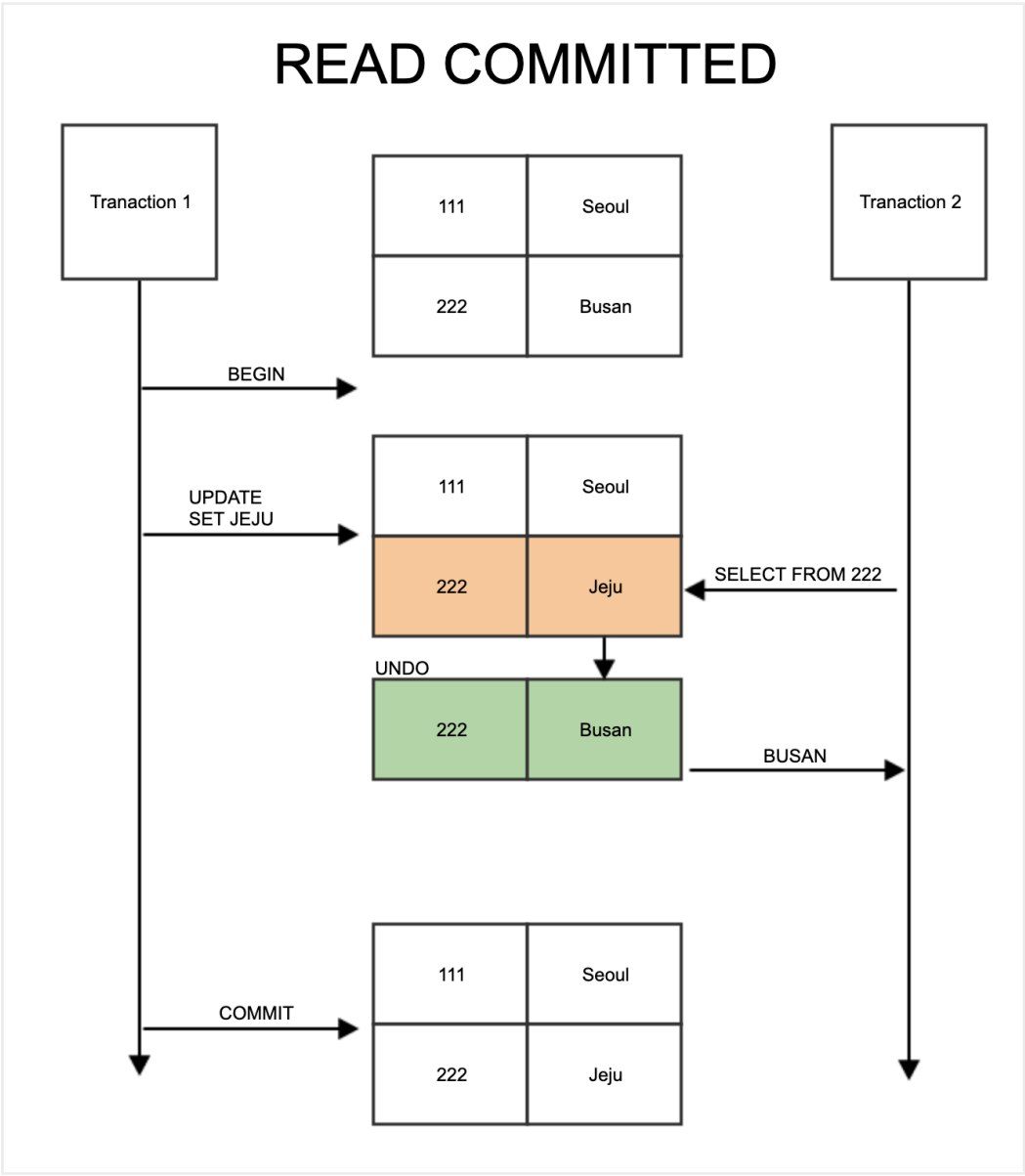
READ UNCOMMITTED의 문제점?

Dirty Read 현상 발생이 문제점입니다. 트랜잭션이 작업이 완료되지 않았는데도 다른 트랜잭션에서 볼 수 있게 되는 현상입니다.

READ COMMITEED

RDB에서 대부분 기본적으로 사용되고 있는 격리 수준입니다.
Dirty Read와 같은 현상은 발생하지 않습니다.

실제 테이블 값을 가져오는 것이 아니라 Undo 영역에 백업된 레코드에서 값을 가져옵니다.



READ COMMITTED의 문제점?

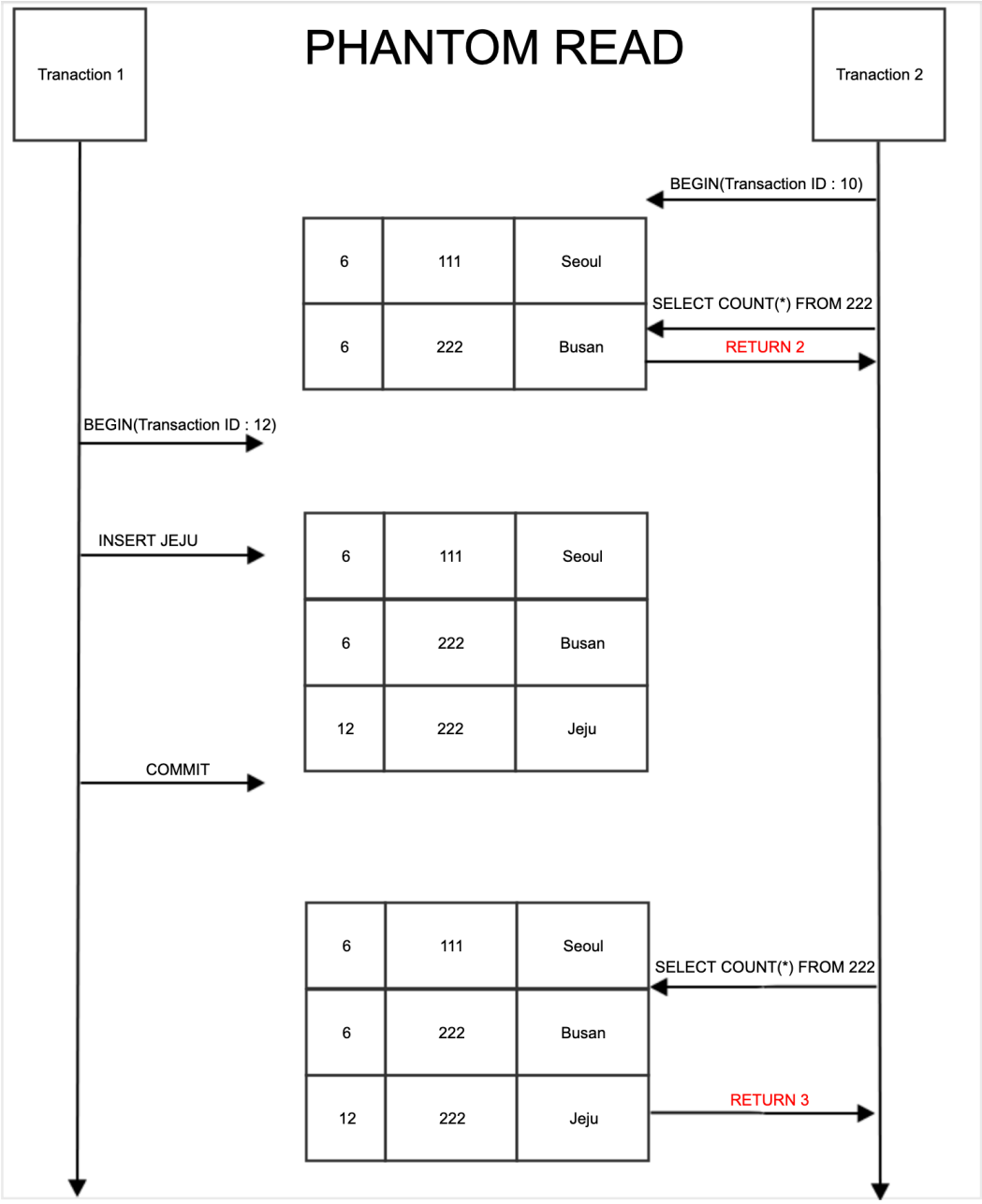
[트랜잭션 -1] 이 Commit한 이후 아직 끝나지 않은 [트랜잭션 -2] 가 다시 테이블 값을 읽으면 값이 변경됨을 알 수 있습니다.
하나의 트랜잭션내에서 똑같은 SELECT 쿼리를 실행했을 때는 항상 같은 결과를 가져와야 하는 REPEATABLE READ 의 정합성에 어긋납니다.
이러한 문제는 주로 입금, 출금 처리가 진행되는 금융쪽의 처리에서 주로 발생합니다. (데이터의 정합성은 깨지고, 버그는 찾기 어려워집니다.)

REPEATABLE READ

MySQL에서는 트랜잭션마다 트랜잭션 ID를 부여하여 트랜잭션 ID보다 작은 트랜잭션 번호에서 변경한 것만 읽게 됩니다.
Undo 공간에 백업해두고 실제 레코드 값을 변경합니다.

REPEATABLE READ 문제점?

PHANTOM READ라는 현상이 발생합니다.
다른 트랜잭션에서 수행한 변경 작업에 레코드가 보였다가 안 보였다가 안 보였다가 하는 현상입니다.
이를 방지하기 위해서는 쓰기 잠금을 걸어야 합니다.



SERIALIZABLE

가장 단순한 격리 수준이지만 가장 엄격한 격리 수준입니다.
성능 측면에서는 동시 처리 성능이 가장 낮습니다.
SERIALIZABLE 에서는 PHANTOM READ가 발생하지 않습니다. (하지만,

데이터베이스에서 거의 사용되지 않습니다.)