

01.06

UncheckedException vs checkedException

UncheckedException

- 확인되지 않은 실수 (개발자의 실수).
- 문법적으로 unchecked라 정해놓았기 때문에, 컴파일 시점에 예외를 catch 하는지 확인하지 않는다.
- spring에서는 default가 트랜잭션 Rollback 이 되는 것이다. → rollback여부 바꿀 수 있다.
- 관련 예외
 - RuntimeException - NullPointerException, IndexOutOfBoundsException

checkedException

- java문법에 의해서 check를 해야한다. 명시적으로 checked로 되어 있는 구문이다
- CheckedException 이 발생할 가능성이 있는 로직은 반드시 **복구, 회피, 처리**해줘야 한다.
- 컴파일 시점에서 예외에 대한 처리가 안되어있다면 with try/catch 컴파일 에러가 발생한다. 처리를 하지 않았다면 예외가 발생하는 메서드에서 throws 예약어를 활용해서 던져야 한다
- spring에서는 default가 트랜잭션 Rollback 이 안되게 되어있다. 즉,checked exception은 발생해도 트랜잭션이 commit 돼버리므로 주의해야한다 → rollback여부 바꿀 수 있다.
- 관련 예외
 - IOException, SQLException

Error vs Exception

1) **Error** : 프로세스 자체에 영향 → JVM에 영향(JVM은 한 프로세스), 주로 JVM에서 발생시키기 때문에 애플리케이션 코드에서 잡아서 안되며, 잡아서 대응할 수 있는 방법도 없다.

2) **Exception**: 한 스레드에만 영향

지연로딩이 적용 안되는 경우

프록시를 사용할 때 외래키를 직접 관리하지 않는 일대일 관계는 지연로딩으로 설정해도 즉시로딩이 된다.

@OneToOne에서 외래키가 A에 있으면 A를 조회하는 순간 B의 데이터가 있는지 확인할 수 있다. 그래서 null을 입력해야 할지, 아니면 프록시를 입력해야 할지 명확한 판단이 가능하다.

외래키가 없는 B를 조회할 때 A가 데이터가 있는지 없는지 판단이 불가능하다. 그래서 null을 입력할지 아니면 프록시를 입력해야 할지 판단이 불가능하다. 따라서 이 경우 강제로 즉시로딩을 해서 데이터가 있으면 해당 데이터를 넣고, 없으면 null을 입력하게 된다.

트랜잭션 격리레벨

1. READ UNCOMMITTED

- 버퍼풀을 읽음 (새로 업데이트된 값을 무조건 읽음)
- 더티 리드 발생 (트랜잭션 처리전에도 업데이트 값을 읽게 됨)

2. READ COMMITTED

- 언두 로그를 읽음 (수정 전 데이터 읽음, 백업용)
- 오라클 DBMS에서 default
- 같은 트랜잭션에도 select 쿼리 여러 번 실행 시 커밋 후 읽었지만 다른 결과가 나오는 경우 발생

생

- phantom read 발생

3. REPEATABLE READ

- 언두 로그를 읽음 (수정 전 데이터 읽음, 백업용)
- MySQL innoDB default

-- 트랜잭션마다 번호를 가져서(id), 자신의 번호보다 작은 번호의 트랜잭션에서 변경된 것만
언 두로그에서 볼 수 있음(현재 트랜잭션 이후의 트랜잭션의 변경사항을 볼 수 없음)

□ 읽는 쪽에서 쓰는 것을 잠금X, 쓰는 곳에서도 읽는 것 잠금X

-- phantom read 발생

4. SERJALIZABLE

-- 언두 로그를 읽음 (수정 전 데이터 읽음, 백업용)

-- 갭락과 넥스트 키락으로 PHANTOM READ 방지

-- 읽기 작업도 읽기 잠금을 획득해야함. -> 다른 트랜잭션은 절대 접근 불가