스프링 트랜잭션 관리 (Spring Transaction Management)

1. 트랜잭션(Transaction)

- 트랜잭션이란 데이터베이스의 상태를 변화시키기 위해 수행하는 하나의 작업 단위를 말한다.

- 트랜잭션은 데이터의 일관성과 무결성을 유지하는 데 필수적이다.

- 데이터베이스와 상호작용하는 어플리케이션에서 트랜잭션 관리는 매우 중요하다. 트랜잭션을 통해 데이터의 일관성과 무결성을 보장할 수 있으며, 여러 작업을 하나의 작업으로 묶어 모두 성공하거나 모두 실패하도록 할 수 있다.

- 스프링에서 트랜잭션 관리를 사용하는 방법에는 크게 선언적 방식과 프로그래매틱 방식이 있다.

2. 트랜잭션의 속성

원자성(Atomicity): 트랜잭션 내의 모든 작업은 모두 성공하거나 모두 실패해야 한다. 예를 들어, 결제 단계에서 문제가 발생한다면, 책 선택 단계로 되돌려야 한다.

일관성(Consistency): 트랜잭션은 데이터베이스를 하나의 일관된 상태에서 다른 일관된 상태로 변화시켜야 한다.

독립성(Isolation): 동시에 여러 트랜잭션이 처리될 때, 각 트랜잭션은 서로의 연산에 영향을 받지 않아야 한다.

지속성(Durability): 트랜잭션이 성공적으로 완료되면, 그 결과는 영구적으로 데이터베이스에 반영되어야 한다.

3. 선언적 트랜잭션 (Declarative Transaction)

- 스프링의 선언적 트랜잭션 관리는 트랜잭션 구현의 복잡성을 추상화하고, 개발자가 비즈니스 로직에 더 집중할 수 있도록 도와준다.

- 스프링 프레임워크는 선언적 트랜잭션 관리를 제공하여, 개발자가 트랜잭션 관리 코드를 직접 작성하지 않아도 되도록 해준다. 이를 통해 개발자는 비즈니스 로직에 더 집중할 수 있으며, 트랜잭션 관리는 스프링이 알아서 처리해준다.

- 선언적 트랜잭션 관리는 주로 @Transactional 어노테이션을 사용하여 구현된다.
이 방식은 비즈니스 로직을 담고 있는 메소드나 클래스에 @Transactional을 선언함으로써, 해당 메소드 또는 클래스의 메소드들이 트랜잭션 범위 안에서 실행되도록 한다.

- @Transactional 어노테이션은 다양한 속성을 제공한다.
예를 들어, readOnly, timeout, isolation, propagation 등의 속성을 통해 트랜잭션의 성질을 세밀하게 제어할 수 있다.

readOnly: 트랜잭션이 데이터를 읽기만 하는 경우 최적화하기 위해 사용된다.
timeout: 트랜잭션이 너무 오래 실행되는 것을 방지하기 위한 타임아웃을 설정할 수 있다.
isolation: 트랜잭션의 격리 수준을 설정할 수 있으며, 이는 다른 트랜잭션과의 격리 정도를 결정한다.
propagation: 트랜잭션의 전파 행위를 정의할 수 있으며, 이는 현재 트랜잭션 내에서 다른 트랜잭션이 어떻게 동작할지를 결정한다.

4. 프로그래매틱 트랜잭션 (Programmatic Transaction)

- 프로그래매틱 트랜잭션 관리에서는 PlatformTransactionManager를 사용하여 직접 트랜잭션을 시작, 커밋, 롤백하는 등의 제어를 코드 내에서 수행한다. 이 방식은 선언적 트랜잭션 관리보다 더 세밀한 제어가 필요한 상황에 적합하다.

- 하지만 코드가 복잡해지고, 트랜잭션 관리 코드가 비즈니스 로직과 섞이게 되어, 가독성과 유지보수성이 저하될 수 있다.