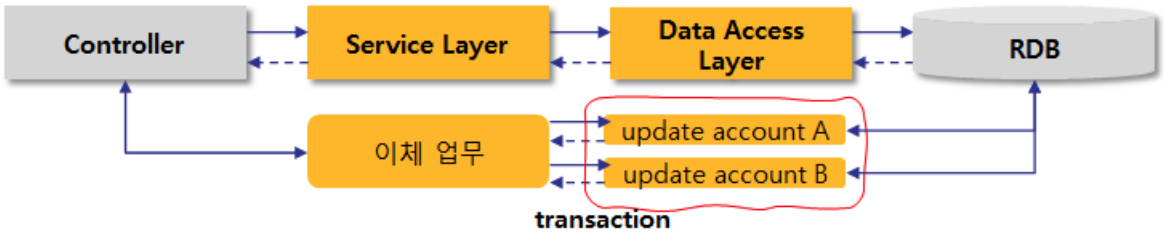
7/6 ~ 7/10 1주차 - 2

Spring과 Model

스프링에서 Model 단 프로그래밍을 위한 내용을 살펴본다.

일반적으로 Enterprise Application을 만들면서 모델은 비즈니스 로직을 다루고 Service Layer와 Data Access Layer로 나뉜다.



Data Access Layer

→ 애플리케이션에서 데이터 저장소(RDB 등)에 접근하기 위한 계층

→ 단일 쿼리 단위의 작업 처리를 담당하는 DAO(Data Access Object) 작성

→ 스프링에서는 DAO 대신 Repository 라는 용어를 즐겨 사용하낟.

Service Layer

→ high level의 기능을 컨트롤러에 노출시키며 Data Access Layer를 사용하는 계층

→ Use-Case 즉 비즈니스 로직이 구현되는 곳

→ 업무의 단위로 트랜젝션 처리의 기점이 되는 곳

스프링 이전에 기존 Data Access Layer의 문제점은?

JDBC를 이용하는 프로그래밍의 예를 보자. 쿼리를 작성하고 Stmt와 ResultSet을 생성하고 쿼리 처리, 리소스 반납의 절차가 쿼리별로 무수히 반복된다.

```java
@Override
public int insertRegion(Connection con, Regions regions) throws Exception {
  int result = -1;
  StringBuilder sql = new StringBuilder("insert into regions values(?,?)");

  PreparedStatement stmt = null;
  try {
    stmt = con.prepareStatement(sql.toString());
    stmt.setInt(1, regions.getRegionId());
    stmt.setString(2, regions.getRegionName());
    result = stmt.executeUpdate();

  }finally{
    stmt.close();
  }
  return result;
}
```

사실상 유일한 비즈니스 로직

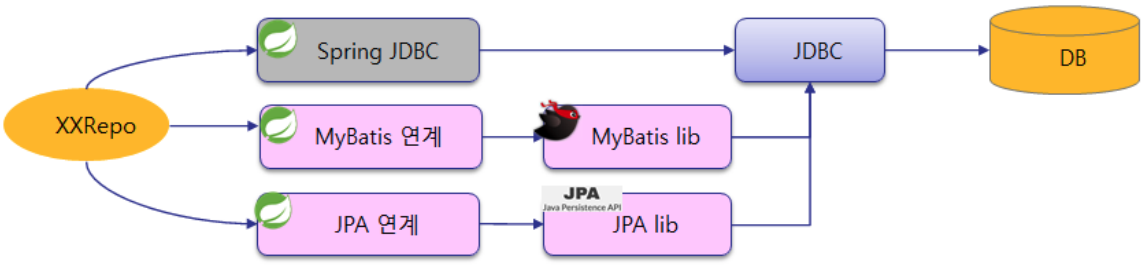
비즈니스 로직 처리를 위한 절차

사용한 리소스 반납

원하는 작업을 위해 군더더기 코드가 많음
→ 모든 로직에서 반복, 에러 발생 가능성 ↑

Cross Cutting Concern(횡단 관심사)와 AOP이다. 스프링에서는 @Repository 클래스에 대해 AOP를 적용해서 핵심 비즈니스 로직 개발에 전념할 수 있다.

> 데이터 액세스 레이어를 사용하기 위해서는 순수 JDBC가 사용되는 데 MyBatis 같은 SQL 매핑 프레임워크나 JPA같은 OR Mapping 프레임 워크로 JDBC 코드를 래핑해서 사용한다.



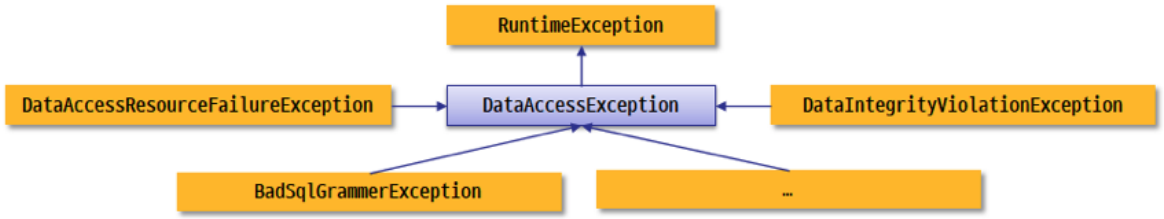
> 우리나라에서는 Mybatis가 추세이고 글로벌하게는 JPA가 점유율이 높지만 mybatis를 먼저 익힌다.

스프링은 범용적이면서 체계적인 예외처리를 지원한다.

기존의 JDBC코드는 SQLException을 발생시켰고 어떤 디비를 사용하는 지에 따라 제각각의 오류코드를 가지고 있었고 컨트롤이 어렵다.

스프링에서는 이런 예외들을 모두 매핑해서 체계적으로 예외를 관리하기 때문에 스프링에서 던져주는 예외만분석하면 된다.

Data Access Layer에 대한 **스프링의 예외**는 **RuntimeException을 상속 받은 DataAccessException**을 필두로 예외의 이름만 봐도 어떤 예외가 발생했는지 쉽게 알 수 있다.



자동화된 리소스 관리

Connection이나 Statement 같은 개체는 획득 후 반드시 반납 처리를 해줘야지 그렇지 않으면 잠재적으로 resource leak이 발생할 수 있다.

스프링에서는 이런 AutoClosable한 리소스들에 대해서 사용후 자동으로 닫아줘서 resource leak을 예방할 수 있다.

MyBatis

SQL을 편하게 작성할 수 있도록 SQL-Object 매핑을 지원하는 Persistence Framework이다.

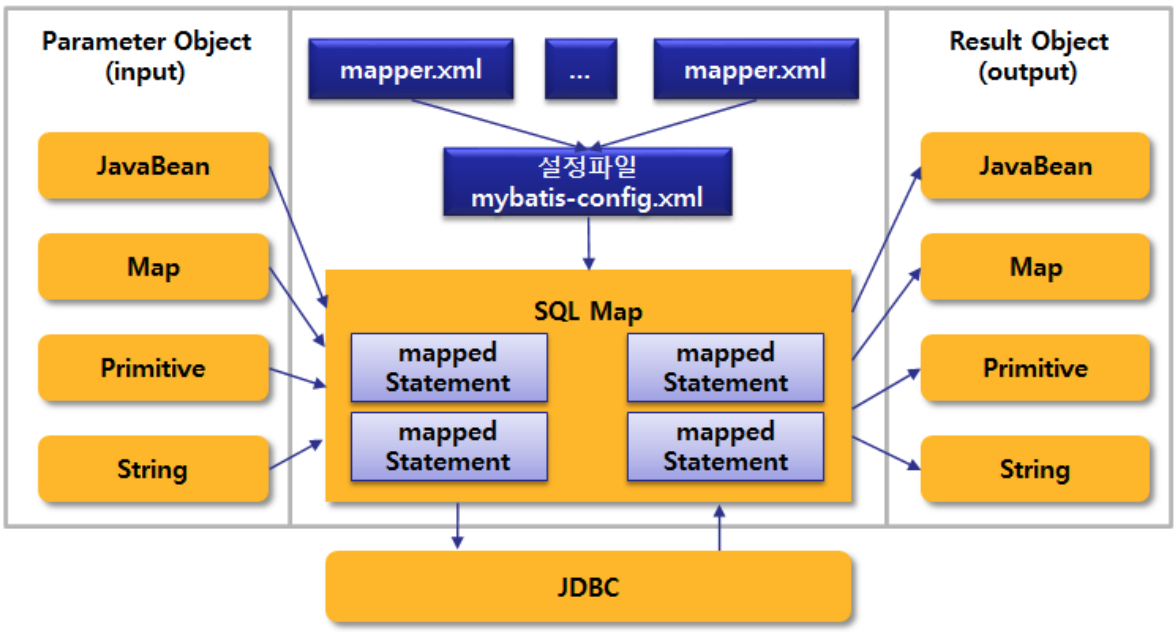
주요 특징

- 기본적으로 PreparedStatement를 사용하기 때문에 SQL 인젝션 공격에 대해 안전하다.
- ResultSet의 내용을 DTO나 컬렉션에 저장하기위한 코드가 필요 없다.

- 일반적으로 XML에서 SQL쿼리를 작성한다.
- 프로그램 코드와 SQL의 분리로 코드가 간결해지고 유지 보수성이 향산된다

MyBatis - 마이바티스 3 | 소개
마이바티스는 개발자가 지정한 SQL, 저장프로시저 그리고 몇가지 고급 매핑을 지원하는 퍼시스턴스 프레임워크이다. 마이바티스는 JDBC로 처리하는 상당부분의 코드와 파라미터 설정및 결과 매핑을 대신해준다. 마이바
https://mybatis.org/mybatis-3/ko/index.html

그럼 MyBatis의 기본 구조는 어떻게 될까?



mybatis-config.xml

→ MyBatis 동작에 대한 설정을 담은 파일

→ Spring Boot와 연동되면서 application.properties에서도 동일한 내용을 설정 가능

mapper.xml

→ xml로 SQL쿼리를 정의해서 등록하며 파라미터 및 결과에 대한 매핑을 처리한다.

→ 일반적으로 테이블당 하나의 mapper.xml을 작성한다.

mybatis-config에서는 여러 mapper들에 정의된 sql들을 로딩해서 SQL map이라는 것을 구성하는데 이를 mapped statement라고 한다. 이 문장을 호출 할때 파라미터로 JavaBeans, java.util.Map, primitive, String가 전달될 수 있고 호출의 결과로도 동일한 타입을 받을 수 있다.

실습은 다른 이론을 정리한 후에 하도록 한다.