초격차 패키지 Online.

JPA 기본기

PART1 | ORM, JPA, JPQL 개요

기본기 정리하고 넘어가기

PART2 | 기존의 기술들 - iBATIS, MyBatis, JdbcTemplate

JPA 를 사용하기 전에 널리 사용하던 기술들

PART3 | Hibernate vs. Spring Data JPA

둘이 무슨 차이에요?

PART4 | in memory 테스트 DB - H2

날아가도 안전한 테스트 DB를 손쉽게 만드는 방법

JPA 기본기

2 기존의 기술들
- iBATIS, MyBatis, JdbcTemplate

SQL Mapper

RDBMS 쿼리문의 실행 결과를 자바 코드에 매핑하는 프레임워크

- JDBC API 를 사용
- persistence framework
- 프로그램 코드와 SQL 을 분리

Apache iBATIS

SQL 데이터베이스와 객체 간 매핑을 지원해주는 persistence framework

- 지원 언어: Java, .NET, Ruby
- SQL 문을 별도의 XML 문서로 작성하여 프로그램 코드와 분리한 형식
- 2001년 Clinton Begin 이 개발
- 2004년 iBATIS 2.0 릴리즈 - 아파치 소프트웨어 재단에 기증, 아파치에서 6년간 운영됨
- 2010년 iBATIS 3.0 릴리즈 - MyBatis로 개발 프로젝트 이동, 아파치 애틱(Attic) 프로젝트로 분류됨
- DAO 패턴이 발전하던 시기
  - Data Access Object 패턴: 애플리케이션 비즈니스 레이어와 영속성 레이어를 추상화된 API를 이용하여 분리
  - DB 접근 구현 클래스를 ~~~Dao 라고 네이밍하는 관례가 많았던 시기

iBATIS: Clinton Begin

Clinton Begin · 3촌
Principal Software Engineer at Coinbase
캐나다 앨버타 에어드리 · 연락처
1촌 500+명

경력 사항

Principal Software Engineer
Coinbase · Permanent Full-time
2021년 9월 - 현재 · (1개월)
Canada

Principal Software Engineer
Riot Games
2015년 12월 ~ 2021년 9월 · (5년 10개월)
Calgary, Canada Area
The best description of what I work on at Riot can be found in the following blog series.

https://technology.riotgames.com/news/running-online-services-riot-part-i

Interim CTO [Startup]
FetchBot
2015년 2월 ~ 2015년 12월 · (11개월)
Calgary, Canada Area

Creator
MyBatis.org
2002년 7월 ~ 2012년 7월 · (10년 1개월)
Calgary, AB

Vice President - Apache iBATIS
Apache Software Foundation
2004년 7월 ~ 2010년 5월 · (5년 11개월)
Calgary, AB
This project was moved to MyBatis.org

iBATIS

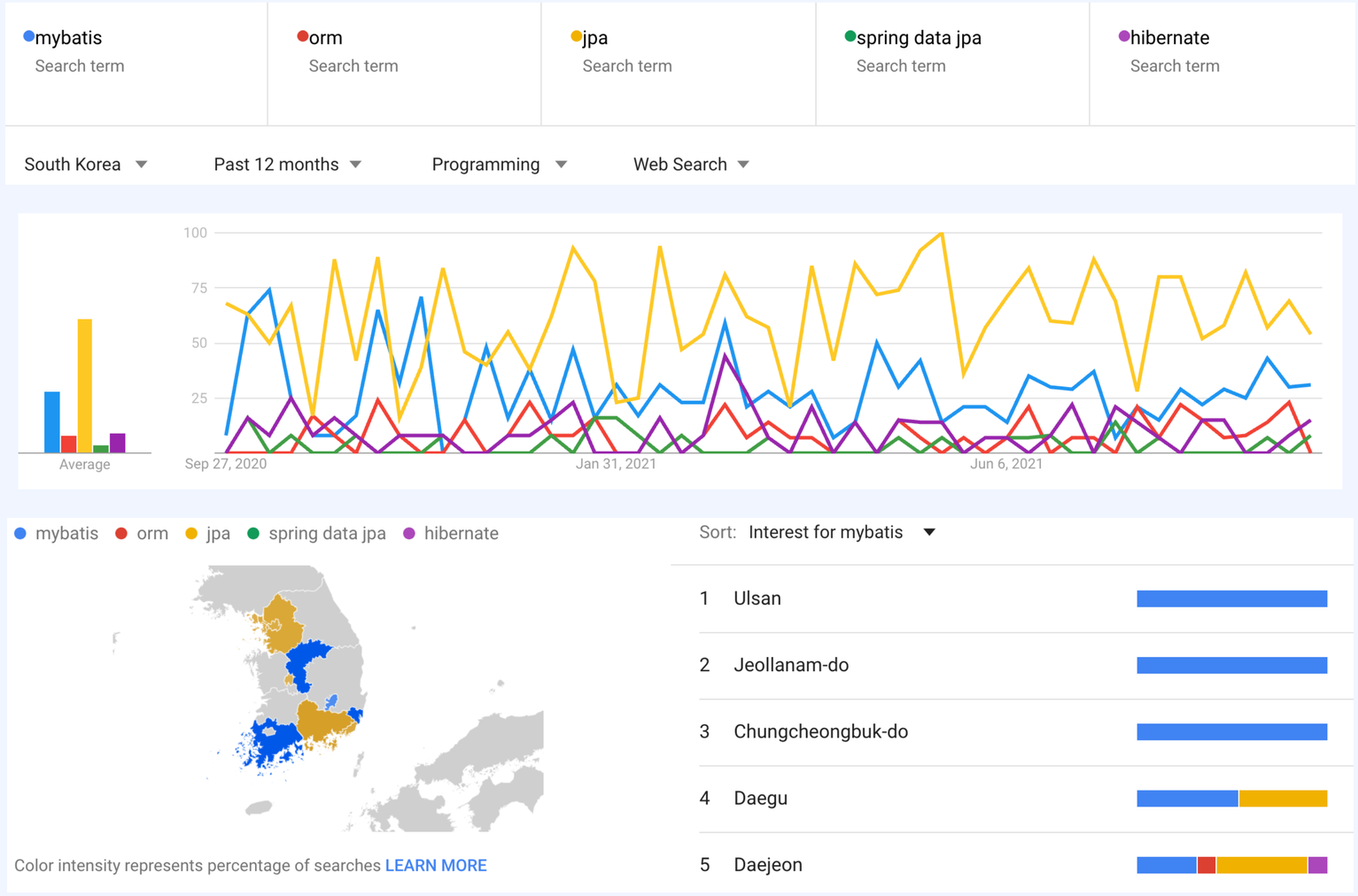
```
<select id="getProduct" parameterClass="java.lang.Long" resultClass="com.example.Product">
select PROD_ID as id,
           PROD_DESC as description
  from PRODUCT
where PROD_ID = #value#
</select>
```

```
Product resultProduct = (Product) sqlMapClient.queryForObject("getProduct", 123);
```

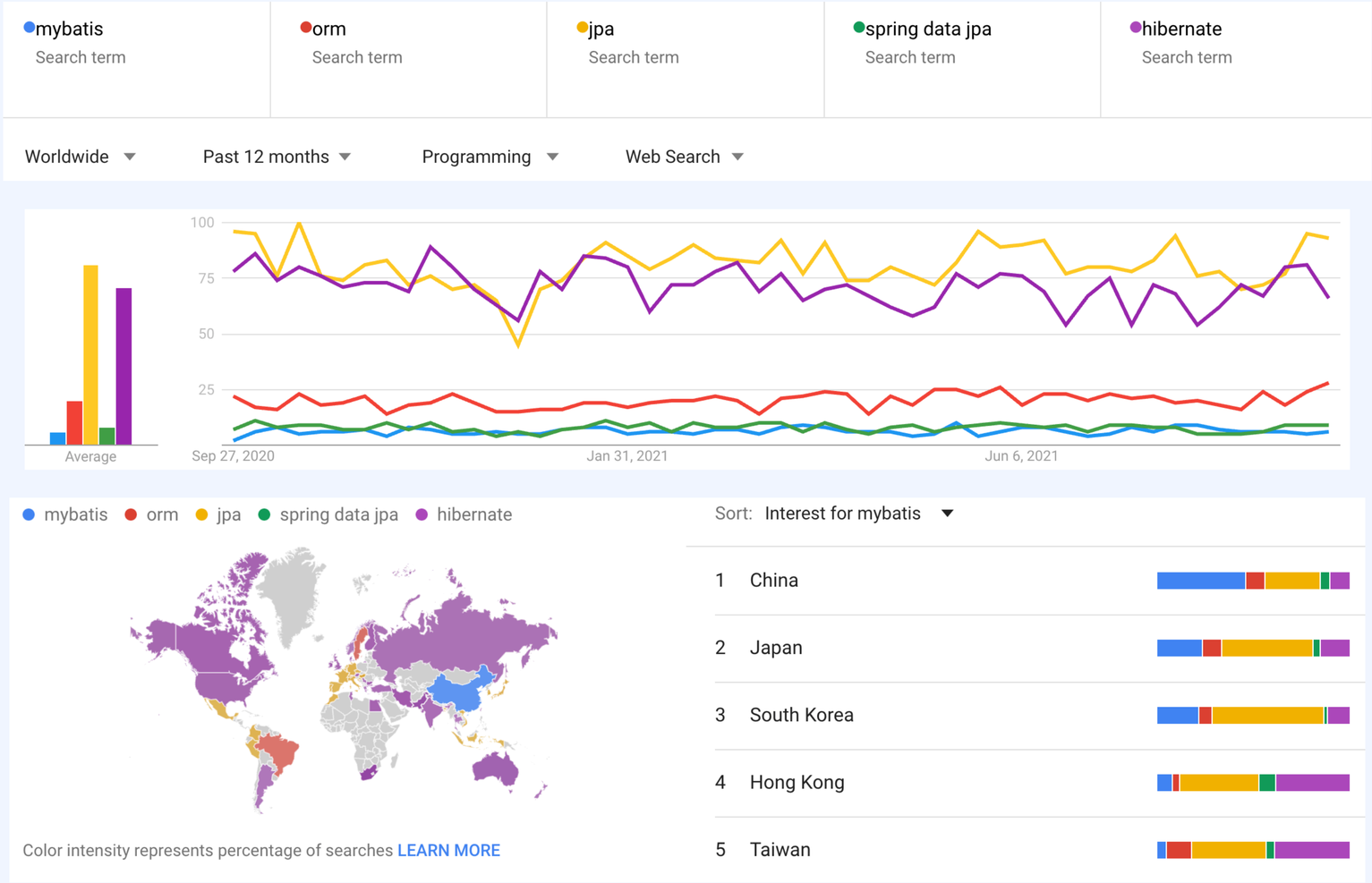
MyBatis

- iBATIS 3.0 에서 출발한 persistence framework (iBATIS 랑 비교할 필요 없이 이거 쓰면 됨)
- 아직 쌩쌩히 살아있는 프로젝트
- 스프링, 스프링 부트와 연동을 지원
  - 스프링: org.mybatis:mybatis-spring
  - 스프링 부트: org.mybatis.spring.boot:mybatis-spring-boot-starter
- 다양한 프레임워크와 연동을 지원
  - Freemarker, Velocity, Hazelcast, Memcached, Redis, Ignite, Guice
- ORM vs. MyBatis
  - ORM: 자바 객체를 DB 테이블과 매핑
  - MyBatis: 자바 메소드를 SQL 실행 결과와 매핑

MyBatis



MyBatis



MyBatis

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE mapper
 PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
 "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="org.mybatis.example.BlogMapper">
 <select id="selectBlog" resultType="Blog">
  select * from Blog where id = #{id}
 </select>
</mapper>
```

```java
package org.mybatis.example;

public interface BlogMapper {
  @Select("select * from Blog where id = #{id}")
  Blog selectBlog(int id);
}
```

```java
BlogMapper mapper = session.getMapper(BlogMapper.class);
Blog blog = mapper.selectBlog(101);
```

MyBatis + Spring Boot

```java
@SpringBootApplication
public class SampleMybatisApplication implements CommandLineRunner {

 private final CityMapper cityMapper;

 public SampleMybatisApplication(CityMapper cityMapper) {
  this.cityMapper = cityMapper;
 }

 public static void main(String[] args) {
  SpringApplication.run(SampleMybatisApplication.class, args);
 }

 @Override
 public void run(String... args) throws Exception {
  System.out.println(this.cityMapper.findByState("CA"));
 }

}
```

```java
@Mapper
public interface CityMapper {

 @Select("SELECT * FROM CITY WHERE state = #{state}")
 City findByState(@Param("state") String state);

}
```

MyBatis vs. iBATIS 사소한 팁

당연히 디테일에서 많은 변화와 차이가 있지만 특별히 알면 좋은 차이점 - 쿼리 실행 결과

	iBATIS	MyBatis
SELECT	SELECT 결과	SELECT 결과
INSERT	NULL	1
UPDATE	1	UPDATE 된 행의 개수
DELETE	DELETE 된 행의 개수	DELETE 된 행의 개수

JDBC API (Spring JDBC)

스프링에서 제공하는 jdbc 기반 persistence framework

- spring-boot-starter-jdbc (spring-boot-starter-data-jdbc 랑 다름)
- JdbcTemplate: Spring JDBC 에서 제공하는 템플릿 클래스. 쿼리 실행과 결과 전달 기능을 제공

JdbcTemplate

```
@SpringBootApplication
public class RelationalDataAccessApplication implements CommandLineRunner {

 public static void main(String args[]) { SpringApplication.run(RelationalDataAccessApplication.class, args); }

 @Autowired JdbcTemplate jdbcTemplate;

 @Override
 public void run(String... strings) throws Exception {
  jdbcTemplate.execute("DROP TABLE customers IF EXISTS");
  jdbcTemplate.execute("CREATE TABLE customers(id SERIAL, first_name VARCHAR(255), last_name VARCHAR(255))");

  // Split up the array of whole names into an array of first/last names
  List<Object[]> splitUpNames = Arrays.asList("John Woo", "Jeff Dean", "Josh Bloch", "Josh Long")
     .stream().map(name -> name.split(" ")).collect(Collectors.toList());

  // Uses JdbcTemplate's batchUpdate operation to bulk load data
  jdbcTemplate.batchUpdate("INSERT INTO customers(first_name, last_name) VALUES (?,?)", splitUpNames);

  jdbcTemplate.query(
     "SELECT id, first_name, last_name FROM customers WHERE first_name = ?", new Object[] { "Josh" },
     (rs, rowNum) -> new Customer(rs.getLong("id"), rs.getString("first_name"), rs.getString("last_name"))
  ).forEach(customer -> System.out.println(customer.toString()));
 }
}
```

SQL Mapper: 사용평

나쁘진 않지만 아쉬운 영속성 프레임워크

- 프로그램 코드에서 아직 SQL 을 완전히 분리하지 못함
  - 개발자가 여전히 SQL 을 알아야 한다 (개발자가 SQL을 진짜로 모르는 건 아니지만)
  - 프로그램이 (특정 DB에 종속된) SQL 을 알아야 한다 -> 전체 코드가 특정 DB 기술과 결합을 가짐
  - XML 관리: SQL 을 분리하는 목적으로 만들었지만.. 내가 결국 XML 을 알아야 해?
- type-safety 를 온전히 활용하지 못한다: 쿼리 실행 결과는 대체로 Map, ResultSet 구조로 넘어옴
  - 결국 매핑은 내가 구현해 줘야 함
  - Map 구조가 데이터 클래스와 비교해서 갖는 단점
    - 어떤 "필드"(맵에서는 key)가 있음을 보장하지 않는다
    - 각 데이터의 타입을 보장하지 않는다
- 결론: 객체 지향적이지 않음

Reference

- https://en.wikipedia.org/wiki/Java_Database_Connectivity
- https://en.wikipedia.org/wiki/Data_access_object
- https://ibatis.apache.org/
- https://blog.mybatis.org/
- https://en.wikipedia.org/wiki/Apache_iBATIS
- https://en.wikipedia.org/wiki/MyBatis
- https://spring.io/guides/gs/relational-data-access/
- https://docs.spring.io/spring-framework/docs/current/reference/html/data-access.html#jdbc