[Hibernate与 MyBatis的比较](http://blog.csdn.net/firejuly/article/details/8190229)

## 1.1 Hibernate 简介

Hibernate对[**数据库**](http://lib.csdn.net/base/mysql)结构提供了较为完整的封装，Hibernate的O/R Mapping实现了POJO 和数据库表之间的映射，以及SQL 的自动生成和执行。程序员往往只需定义好了POJO 到数据库表的映射关系，即可通过Hibernate 提供的方法完成持久层操作。程序员甚至不需要对SQL 的熟练掌握， Hibernate/OJB 会根据制定的存储逻辑，自动生成对应的SQL 并调用JDBC 接口加以执行。

## 1.2 MyBatis简介

iBATIS 的着力点，则在于POJO 与SQL之间的映射关系。然后通过映射配置文件，将SQL所需的参数，以及返回的结果字段映射到指定POJO。 相对Hibernate“O/R”而言，iBATIS 是一种“Sql Mapping”的ORM实现。

# 第二章 开发对比

## 开发速度

Hibernate的真正掌握要比Mybatis来得难些。Mybatis框架相对简单很容易上手，但也相对简陋些。个人觉得要用好Mybatis还是首先要先理解好Hibernate。

## 开发社区

Hibernate 与Mybatis都是流行的持久层开发框架，但Hibernate开发社区相对多热闹些，支持的工具也多，更新也快，当前最高版本4.1.8。而Mybatis相对平静，工具较少，当前最高版本3.2。

## 开发工作量

Hibernate和MyBatis都有相应的代码生成工具。可以生成简单基本的DAO层方法。

针对高级查询，Mybatis需要手动编写SQL语句，以及ResultMap。而Hibernate有良好的映射机制，开发者无需关心SQL的生成与结果映射，可以更专注于业务流程。

# 第三章 系统调优对比

## Hibernate的调优方案

1. 制定合理的缓存策略；
2. 尽量使用延迟加载特性；
3. 采用合理的Session管理机制；
4. 使用批量抓取，设定合理的批处理参数（batch\_size）;
5. 进行合理的O/R映射设计

## Mybatis调优方案

MyBatis在Session方面和Hibernate的Session生命周期是一致的，同样需要合理的Session管理机制。MyBatis同样具有二级缓存机制。 MyBatis可以进行详细的SQL优化设计。

## SQL优化方面

Hibernate的查询会将表中的所有字段查询出来，这一点会有性能消耗。Hibernate也可以自己写SQL来指定需要查询的字段，但这样就破坏了Hibernate开发的简洁性。而Mybatis的SQL是手动编写的，所以可以按需求指定查询的字段。

Hibernate HQL语句的调优需要将SQL打印出来，而Hibernate的SQL被很多人嫌弃因为太丑了。MyBatis的SQL是自己手动写的所以调整方便。但Hibernate具有自己的日志统计。Mybatis本身不带日志统计，使用Log4j进行日志记录。

## 扩展性方面

Hibernate与具体数据库的关联只需在XML文件中配置即可，所有的HQL语句与具体使用的数据库无关，移植性很好。MyBatis项目中所有的SQL语句都是依赖所用的数据库的，所以不同数据库类型的支持不好。

# 第四章 对象管理与抓取策略

## 对象管理

Hibernate 是完整的对象/关系映射解决方案，它提供了对象状态管理（state management）的功能，使开发者不再需要理会底层数据库系统的细节。也就是说，相对于常见的 JDBC/SQL 持久层方案中需要管理 SQL 语句，Hibernate采用了更自然的面向对象的视角来持久化 [**Java**](http://lib.csdn.net/base/javase) 应用中的数据。

换句话说，使用 Hibernate 的开发者应该总是关注对象的状态（state），不必考虑 SQL 语句的执行。这部分细节已经由 Hibernate 掌管妥当，只有开发者在进行系统性能调优的时候才需要进行了解。

而MyBatis在这一块没有文档说明，用户需要对对象自己进行详细的管理。

## 抓取策略

Hibernate对实体关联对象的抓取有着良好的机制。对于每一个关联关系都可以详细地设置是否延迟加载，并且提供关联抓取、查询抓取、子查询抓取、批量抓取四种模式。 它是详细配置和处理的。

而Mybatis的延迟加载是全局配置的。

# 第五章 缓存机制对比

## Hibernate缓存

Hibernate一级缓存是Session缓存，利用好一级缓存就需要对Session的生命周期进行管理好。建议在一个Action操作中使用一个Session。一级缓存需要对Session进行严格管理。

Hibernate二级缓存是SessionFactory级的缓存。 SessionFactory的缓存分为内置缓存和外置缓存。内置缓存中存放的是SessionFactory对象的一些集合属性包含的数据(映射元素据及预定SQL语句等),对于应用程序来说,它是只读的。外置缓存中存放的是数据库数据的副本,其作用和一级缓存类似.二级缓存除了以内存作为存储介质外,还可以选用硬盘等外部存储设备。二级缓存称为进程级缓存或SessionFactory级缓存，它可以被所有session共享，它的生命周期伴随着SessionFactory的生命周期存在和消亡。

## MyBatis缓存

MyBatis 包含一个非常强大的查询缓存特性,它可以非常方便地配置和定制。MyBatis 3 中的缓存实现的很多改进都已经实现了,使得它更加强大而且易于配置。

默认情况下是没有开启缓存的,除了局部的 session 缓存,可以增强变现而且处理循环 依赖也是必须的。要开启二级缓存,你需要在你的 SQL 映射文件中添加一行:  <cache/>

字面上看就是这样。这个简单语句的效果如下:

1. 映射语句文件中的所有 select 语句将会被缓存。
2. 映射语句文件中的所有 insert,update 和 delete 语句会刷新缓存。
3. 缓存会使用 Least Recently Used(LRU,最近最少使用的)算法来收回。
4. 根据时间表(比如 no Flush Interval,没有刷新间隔), 缓存不会以任何时间顺序 来刷新。
5. 缓存会存储列表集合或对象(无论查询方法返回什么)的 1024 个引用。
6. 缓存会被视为是 read/write(可读/可写)的缓存,意味着对象检索不是共享的,而 且可以安全地被调用者修改,而不干扰其他调用者或线程所做的潜在修改。

所有的这些属性都可以通过缓存元素的属性来修改。

比如: <cache  eviction="FIFO"  flushInterval="60000"  size="512"  readOnly="true"/>

这个更高级的配置创建了一个 FIFO 缓存,并每隔 60 秒刷新,存数结果对象或列表的 512 个引用,而且返回的对象被认为是只读的,因此在不同线程中的调用者之间修改它们会 导致冲突。可用的收回策略有, 默认的是 LRU:

1. LRU – 最近最少使用的:移除最长时间不被使用的对象。
2. FIFO – 先进先出:按对象进入缓存的顺序来移除它们。
3. SOFT – 软引用:移除基于垃圾回收器状态和软引用规则的对象。
4. WEAK – 弱引用:更积极地移除基于垃圾收集器状态和弱引用规则的对象。

flushInterval(刷新间隔)可以被设置为任意的正整数,而且它们代表一个合理的毫秒 形式的时间段。默认情况是不设置,也就是没有刷新间隔,缓存仅仅调用语句时刷新。

size(引用数目)可以被设置为任意正整数,要记住你缓存的对象数目和你运行环境的 可用内存资源数目。默认值是1024。

readOnly(只读)属性可以被设置为 true 或 false。只读的缓存会给所有调用者返回缓 存对象的相同实例。因此这些对象不能被修改。这提供了很重要的性能优势。可读写的缓存 会返回缓存对象的拷贝(通过序列化) 。这会慢一些,但是安全,因此默认是 false。

## 相同点

Hibernate和Mybatis的二级缓存除了采用系统默认的缓存机制外，都可以通过实现你自己的缓存或为其他第三方缓存方案，创建适配器来完全覆盖缓存行为。

## 不同点

Hibernate的二级缓存配置在SessionFactory生成的配置文件中进行详细配置，然后再在具体的表-对象映射中配置是那种缓存。

MyBatis的二级缓存配置都是在每个具体的表-对象映射中进行详细配置，这样针对不同的表可以自定义不同的缓存机制。并且Mybatis可以在命名空间中共享相同的缓存配置和实例，通过Cache-ref来实现。

## 两者比较

因为Hibernate对查询对象有着良好的管理机制，用户无需关心SQL。所以在使用二级缓存时如果出现脏数据，系统会报出错误并提示。

而MyBatis在这一方面，使用二级缓存时需要特别小心。如果不能完全确定数据更新操作的波及范围，避免Cache的盲目使用。否则，脏数据的出现会给系统的正常运行带来很大的隐患。

# 第六章 Hibernate与Mybatis对比总结

## 两者相同点

* Hibernate与MyBatis都可以是通过SessionFactoryBuider由XML配置文件生成SessionFactory，然后由SessionFactory 生成Session，最后由Session来开启执行事务和SQL语句。其中SessionFactoryBuider，SessionFactory，Session的生命周期都是差不多的。
* Hibernate和MyBatis都支持JDBC和JTA事务处理。

## Mybatis优势

* MyBatis可以进行更为细致的SQL优化，可以减少查询字段。
* MyBatis容易掌握，而Hibernate门槛较高。

## Hibernate优势

* Hibernate的DAO层开发比MyBatis简单，Mybatis需要维护SQL和结果映射。
* Hibernate对对象的维护和缓存要比MyBatis好，对增删改查的对象的维护要方便。
* Hibernate数据库移植性很好，MyBatis的数据库移植性不好，不同的数据库需要写不同SQL。
* Hibernate有更好的二级缓存机制，可以使用第三方缓存。MyBatis本身提供的缓存机制不佳。

## 他人总结

* Hibernate功能强大，数据库无关性好，O/R映射能力强，如果你对Hibernate相当精通，而且对Hibernate进行了适当的封装，那么你的项目整个持久层代码会相当简单，需要写的代码很少，开发速度很快，非常爽。
* Hibernate的缺点就是学习门槛不低，要精通门槛更高，而且怎么设计O/R映射，在性能和对象模型之间如何权衡取得平衡，以及怎样用好Hibernate方面需要你的经验和能力都很强才行。
* iBATIS入门简单，即学即用，提供了数据库查询的自动对象绑定功能，而且延续了很好的SQL使用经验，对于没有那么高的对象模型要求的项目来说，相当完美。
* iBATIS的缺点就是框架还是比较简陋，功能尚有缺失，虽然简化了数据绑定代码，但是整个底层数据库查询实际还是要自己写的，工作量也比较大，而且不太容易适应快速数据库修改。