MyBatis查询缓存的作用域是根据映射文件mapper的namespace划分的，相同namespace的mapper查询数据存放在同一个缓存区域。不同的namespace下的数据互不干扰。无论是一级缓存还是二级缓存，都是按照namespace进行分别存放的。

但一、二级缓存的不同之处在于，SqlSession一旦关闭，则SqlSession中的数据将不存在，即一级缓存就不复存在。而二级缓存的生命周期会与整个应用同步，与SqlSession是否关闭无关。

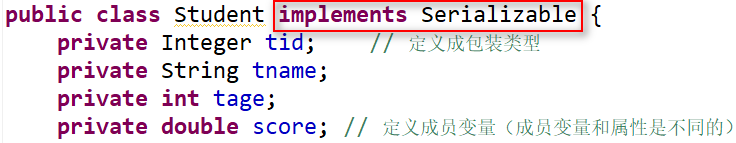
使用二级缓存的目的，不是共享数据，因为MyBatis从缓存中读取数据的依据是SQL的id，而非查询出的对象。所以，二级缓存中的数据不是为了在多个查询之间共享（所有查询中只要查询结果中存在该对象，就直接从缓存中读取，这是对数据的共享，Hibernate中的缓存就是为了共享，但MyBatis的不是），而是为了延长该查询结果的保存时间，提高系统性能。

MyBatis内置的二级缓存为org.apache.ibatis.cache.impl.PerpetualCache。

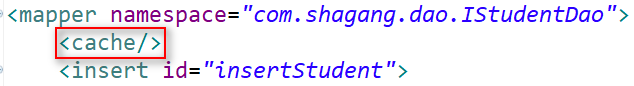
###### 二级缓存的用法

二级查询缓存的使用分为两步：

1. 实体序列化



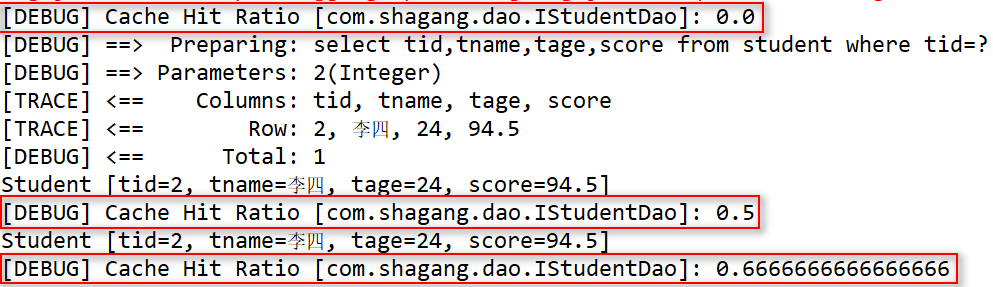
1. mapper映射中添加<cache/>标签



###### 二级查询缓存的存在性证明



Cache Hit Ratio表示缓存命中率。开启二级缓存后，每执行一次查询，系统都会计算一次二级缓存的命中率。第一次查询也是先从缓存中查询，只不过缓存中一定是没有的。，所以会再从DB中查询。



###### 二级缓存的配置

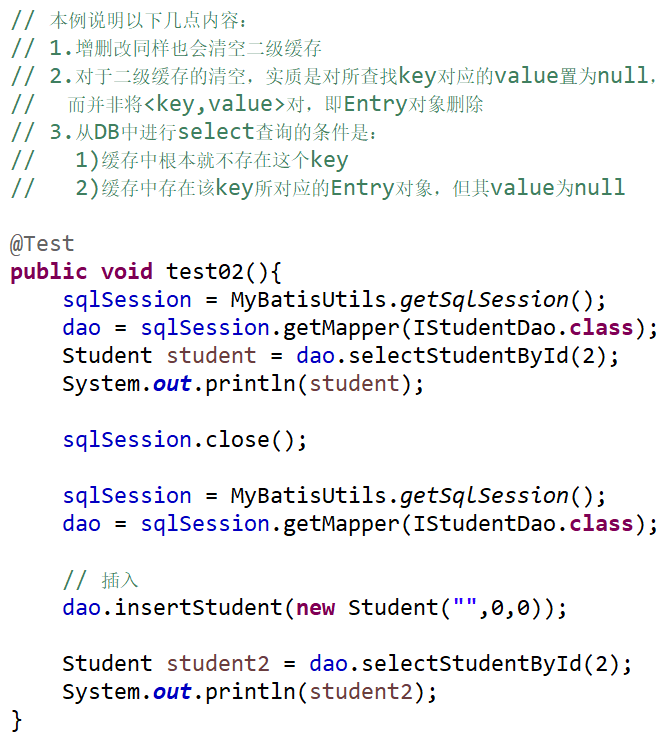
<cache   
eviction="FIFO"    
flushInterval="10800000"  size="512"   
readOnly="true"  />   
 这个配置创建了一个FIFO缓存，并每隔3个小时刷新缓存，储存结果对象或列表的512个引用，而且返回的对象被认为是只读的，因此在不同线程中的调用者之间修改他们会导致冲突。

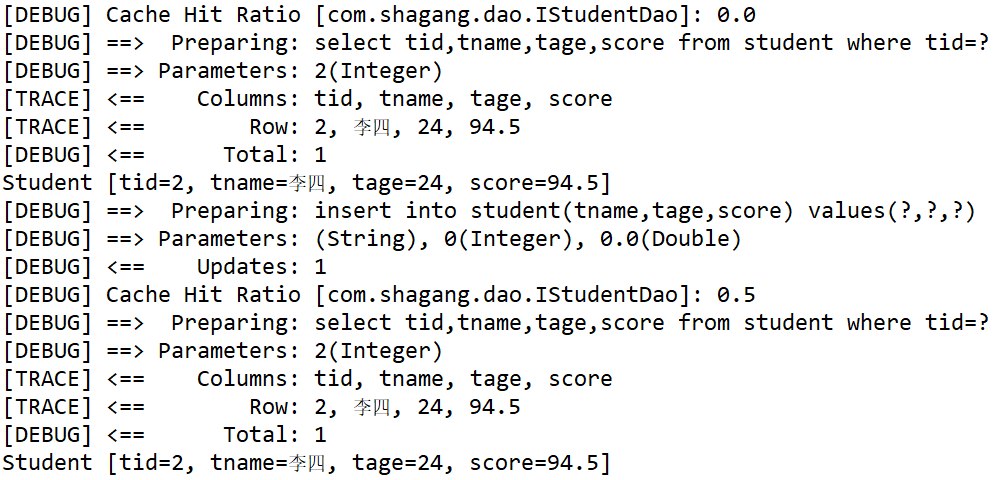
可用的收回策略有：

* 【默认】LRU——最近最少使用的：移除最长时间不被使用的对象
* FIFO——先进先出的：按对象进入缓存的顺序来移除他们
* SOFT——软引用：移除基于垃圾回收器状态和软引用规则的对象
* 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（60\*60\*1000这种形式是不允许的），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（通过发序列化）。这会慢一些，但是安全，因此默认是false。

###### 增删改对二级查询缓存的影响

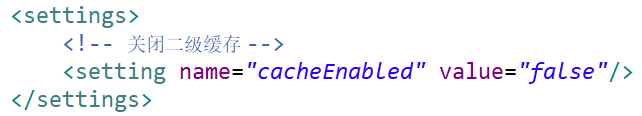




###### 二级缓存的关闭

1. 全局关闭

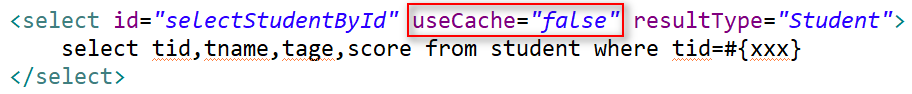
所谓全局关闭是指，整个应用的二级缓存全部关闭，所有查询均不使用二级缓存。全局开关设置在主配置文件的全局设置<settings/>中，该属性为cacheEnabled，设置为false，则关闭，默认值为true开启。



1. 局部关闭

所谓局部关闭是指，整个应用的二级缓存是开启的，但只是针对于某个<select/>查询，不适用二级缓存。此时可以单独只关闭该<select/>标签的二级缓存。

在该要关闭二级缓存的<select/>标签中，将其属性useCache设置为false，即可关闭该查询的二级缓存。该属性默认为true，即每个<select/>查询的二级缓存默认是开启的。



###### 二级缓存的使用原则

1. **多个namespace不要操作同一张表**

由于二级缓存中的数据是基于namespace的，即不同namespace中的数据互不干扰。在多个namespace中若均存在对同一个表的操作，那么这多个namespace中的数据可能就会出现不一致现象。

1. **在单表上使用二级缓存**

如果一个表与其他表有关联关系，那么就非常有可能存在多个namespace对同一数据的操作。而不同namespace中的数据互补干扰，所以有可能出现这多个namespace中的数据不一致的现象。

1. **查询多于修改时使用二级缓存**

在查询操作远远多于增删改操作的情况下可以使用二级缓存。因为任何增删改操作都将刷新二级缓存，对二级缓存的频繁刷新将降低系统性能。