Hibernate EhCache 二级缓存配置

1、 简介：

EhCache 缓存插件是 Hibernate 的另一个项目，Hibernate 框架内置了对它的支持，对于单机应用推荐使用它做为 Hibernate 的二级缓存。

环境配置如下：

1. MySql 5.0 + Hibernate 3.3
2. 数据库建表 DDL：

```
Create Table Dept(
    id int auto_increment Primary Key ,          /* 编号 */
    dname varchar(30) not null,                  /* 部门名称 */
    version int                                  /* 对象版本字段，无实际意义 */
);
```

2、 添加 Hibernate 框架：

a) 在项目中添加 Hibernate 3.3 的 jar 包，并实现与 Dept 表的映射。能正常执行以下语句：

```
Configuration conf = new Configuration().configure();
SessionFactory sf = conf.buildSessionFactory();
Dept dept = new Dept();
dept.setDname("软件");
Session sess = sf.openSession();
Transaction tx = null;
tx = sess.beginTransaction();
session.save(dept);
tx.commit();
session.close();
```

3、 配置 EhCache 缓存插件，添加 Dept 数据到二级缓存：

a) 添加 ehcache 的 jar 包到项目中，在项止的 src 目录下，新建一个名称为：ehcache.xml 的文件，内容如下：

```
<ehcache>
    <diskStore path="java.io.tmpdir"/>
    <defaultCache
        maxElementsInMemory="10000"
        eternal="false"
        overflowToDisk="true"
```

```
        timeToIdleSeconds="120"
        timeToLiveSeconds="120"
        diskPersistent="false"
        diskExpiryThreadIntervalSeconds="120"
        />
        <cache name="com.entity.Dept"
            maxElementsInMemory="10000"
             eternal="false"
             overflowToDisk="true"
             timeToIdleSeconds="120"
            timeToLiveSeconds="120"
         />
</ehcache>
```

b) ehcache.xml 元素说明：

diskStore：设置缓存数据存放的路径。

defaultCache：设置数据的默认过期策略。

cache 设置具体的命名缓存的数据过期策略。

c) 在 ehcache.xml 文件中通过 cache 元素为每个需要二级缓存的类或集合设置缓存的数据过期策略，cache 元素的属性 ：

name：设置缓存的名称，它的值是类的完整名称或集合的名称。

maxElementsInMemory：设置内存中允许存放对象的最大数量。

eternal：如果为 true，表示对象永不过期（默认为 false）。

overflowToDisk：如果为 true，表示当对象数量超过了最大上限，则把溢出的对象写到基于硬盘的缓存中。

timeToIdleSeconds：设置允许对象空闲的最大时间。当超过这个时间，Ehcache 会把对象从缓存中清除。

timeToLiveSeconds：设置对象允许存在于缓冲中的最大时间。如果取值为 0：表示对象可以无限期的存在于缓存中，它的值只有在大于等于 timeToIdleSeconds 时，才有意义。

d) 在具体类的映射文件中，添加<cache>元素，完整的 Dept.hbm.xml：

```
<hibernate-mapping package="com.entity">
    <class name="Dept" table="dept" optimistic-lock="version">
        <cache usage="read-only" />
        <id name="id" type="integer" column="id" >
            <generator class="native"/>
        </id>
        <version name="version" column="version"
                      type="integer" />
        <property name="dname" column="dname"
                       type="string" />
    </class>
</hibernate-mapping>
```

在 <id>元素前添加<cache>元素，设置使用二级缓存时的策略。适合在单机环境中使用的取值有：

read-only：只读策略，缓存中的数据只能读取不能修改。

read-write：读写策略，缓存中的数据既能读也能写。

4、 启用项目的二级缓存

a) 在 hibernate.cfg.xml 中，添加以下属性：

```
<property name="hibernate.cache.provider_class">
       org.hibernate.cache.EhCacheProvider
</property>
<property name="hibernate.cache.use_query_cache">true</property>
```

5、 测试：

向数据库表中添加一条数据，测试下面的代码：

```
Configuration conf = Configuration().configure();
SessionFactory sf = conf.buildSessionFactory();
Session s1 = sf.openSession();

List d1 = s1.createQuery("from Dept").setCacheable(true).list();
s1.close();
System.out.println(" \n ================ \n");
Session s2 = sf.openSession();
List d3 = s2.createQuery("from Dept").setCacheable(true).list();
s2.close();
sf.close();
```

提示：

a) 在测试过程中，可以通过配置 启用 / 禁用 二级缓存查看更详细的效果。在 Hibernate 的配置文件中，添加如下配置：

```
<property name="hibernate.cache.use_second_level_cache">
    false
</property>
```

默认值为：true，表示启用二级缓存；改为 false：禁用二级缓存。