JPA 2.0 Cache Vs. Hibernate Cache: Differences in Approach

- July 9, 2010
- By Sangeetha S, Nitin KL
- Bio »
- Send Email »
- More Articles »

Pros, Cons and Best Uses of JPA Level 2 Cache

Here are the pros and cons of JPA Level 2 cache:

- Pros:
  - Avoids database access for already loaded entities
  - Faster for reading frequently accessed unmodified entities
- Cons:
  - Memory consumption for large amount of objects
  - Stale data for updated objects
  - Concurrency for write (optimistic lock exception or pessimistic lock)
  - Bad scalability for frequent or concurrently updated entities

Level 2 cache is best used for entities that are frequently read, infrequently modified, and not critical if stale. You can use the query cache technique to cache queries that are executed often with the same parameters for tables that are rarely modified.

Caching in Hibernate

Hibernate also maintains two levels of cache: first-level and second-level cache. The first-level cache is responsible for storing the results within a particular session instance, while the second-level cache is associated with the `SessionFactory` instance.

Hibernate uses first-level cache by default to store the objects for every transaction. Hibernate's second-level cache, which is supported by the `SessionFactory`, enables the objects to be accessed at the application level, thereby reducing the number of database transactions required for accessing an object. Hibernate achieves caching by storing the individual property values of the instance rather than storing the objects themselves.

Hibernate 3.0 supports the following four open source caching implementations for second-level cache:

- EHCache (`org.hibernate.cache.EhCacheProvider`) -- Default
- OSCache (`org.hibernate.cache.OSCacheProvider`)
- SwarmCache (`org.hibernate.cache.SwarmCacheProvider`)
- JBoss TreeCache (`org.hibernate.cache.TreeCacheProvider`)

The second-level cache can be enabled or disabled by setting the property `hibernate.cache.use_second_level_cache` to true (the default for classes that specify `<cache>` mapping) or false, respectively. Here is a true setting:

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

You can choose which implementation to use for an application by setting the `hibernate.cache.provider_class` property in the `hibernate.cfg.xml` file as follows.

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

You also can enable caching at the class level or the collection level by setting the `<cache>` element in the mapping file as follows:

```
<cache usage="read-only" region="regionName" include="all"/>
```

Here's a breakdown of the elements in the above code:

- `Usage` specifies the caching strategy and can take other values such as `transactional`, `read-write` and `nonstrict-read-write`.
- `Region` is an optional attribute that specifies the second-level cache region. By default it is the name of the class/collection.
- `Include` is an optional attribute that takes the `all` value by default. If the value `non-lazy` is set to include then entities having `lazy=true` cannot be cached.

You can also enable caching by setting the `<class-cache>` and `<collection-cache>` elements in the `hibernate.cfg.xml` file, after which you would configure the cache rule for classes for which you want cache to be enabled in a separate EhCache configuration file (`ehcache.xml`) and place it in the root directory of the project.

You can also cache queries that are executed very frequently with the same set of parameters by using query cache. Query cache is set to false by default and you can enable it by adding the following property in the `hibernate.cfg.xml` file:

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

This query adds `StandardQueryCache` and `UpdateTimestampsCache` regions, which hold the query cached results and the time stamps of the most recent update to the table, respectively. The query results can be cached for a particular query by calling `setCacheable(true)` on the query instance.

Hibernate 3.5 Caching

The Hibernate caching strategies remain the same in version 3.5, but certain additional cache providers such as JBoss Cache 2, JBoss Cache 1.x (part of Hibernate 3.2) and Hashtable (not for production) have been added.

The other major advancement in Hibernate 3.5 is the addition of Infinispan as another standard for second-level cache. Infinispan is an open source, scalable data grid platform that exposes a JCache (JSR-107)-compatible cache interface. Infinispan provides a much higher degree of concurrency because it uses a specialized data structure, provides a massive heap capability and is not tied just to Java. It also supports PHP, Python, Ruby etc.

Conclusion

In this article we compared caching in JPA 2.0 with the caching in Hibernate. By introducing new caching features and promoting standardization, JPA 2.0 has made many tasks easier for developers. However, Hibernate is far ahead in many aspects because all its features have been supported for much longer.

Acknowledgements

The authors would like to sincerely thank Mr. Subrahmanya SV (VP, ECOM Research Group, E&R) for his ideas, guidance, support and constant encouragement and Ms. Mahalakshmi for kindly reviewing this article and providing valuable comments.

About the Authors

Sangeetha S. *works as a Senior Technical Architect at the E-Commerce Research Labs at Infosys Technologies. She has over 10 years of experience in design and development of Java and Java EE applications. She has co-authored a book on 'J2EE Architecture' and also has written articles for online Java publications.*

Nitin KL *works at the E-Commerce Research Labs at Infosys Technologies. He is involved in design and development of Java EE applications using Hibernate, iBATIS, and JPA.*

Tags: Persistence

8 Comments (click to add your comment)

By Ramesh August 07 2010 17:42 PDT

If you are using PersistenceContextType.EXTENDED\n\n1) I guess you cannot use the application outside a container. For example a swing client \n\n2) What will be the behavior in a multi threaded environment.

Reply to this comment By Pgmr July 26 2010 05:54 PDT

Can someone tell me is it possible to execute PL/SQL in JPA.? if so can you share some links for that.

Reply to this comment By rozmar564 July 19 2010 11:43 PDT

Good article.

Reply to this comment By sam July 17 2010 10:37 PDT

it might be true that its only slow on console logging. But other than that the default hibernate cache logging is not that useful. thanks for sharing.

Reply to this comment By Lava Kafle July 12 2010 22:51 PDT

Perfect article!!

Reply to this comment By Santosh July 12 2010 14:44 PDT

\nHi Folks,\n\nGood article describing what has changed, but I for one remain unconvinced on the real world use of ORM and application caching.\n\nIn a real world, real time applications have to factor in the constant cost of translation from Java data structures to RDBMS database (ORM) and also the extra memory and processing introduced by Caching is not a luxury that can be called upon by real world applications.\n\nThese technologies remain good for white papers and POC/prototype applications, nothing more.\n\nThanks for the good article, anyways.\n\nRegards,\n\nSantosh

Reply to this comment By Champak October 15 2012 03:34 PDT

Agree with Santosh's comments above. I see following problems with ORM: 1. Instead of decreasing (as they claim), they increase the overall development time. 2. They generate even more SQL Queries and therefore load on the database. Just look at Hiberate Logs. Going crazy. 3. For complex queries you need to bypass ORM anyway. Tools like Hibernate/EclipseLink are good for White Papers and Resumes. I have come across very few large scale Prodcution Quality Systems with properly used ORM in it. The same stands for: Rules Engines like Drools, BPM Engines like Activiti and JBPM. These can be useful in limited controlled circumstances.

By Olivier Refalo October 11 2010 18:46 PDT

Using an ORM is typically slower than using a jdbc query, however as you application needs to be deployed on several nodes/instance to create a

cluster... you will quickly realize that it then makes complete sense and does optimize calls to the db. On top of it, it also provides an clean api to access the data. OR

Comment and Contribute

Your name/nickname

Your email

Subject

(Maximum characters: 1200). You have 1200 characters left.

Type the text

Privacy & Terms

Submit Your Comment

Enterprise Development Update

Don't miss an article. Subscribe to our newsletter below.

Enter Email Address SIGN UP

Most Popular Developer Stories

- Today
- This Week
- All-Time

- 1 Using JDBC with MySQL, Getting Started
- 2 An Introduction to Java Annotations
- 3 MIDP Programming with J2ME
- 4 An Introduction to JSP Standard Template Library (JSTL)
- 5 Debugging a Java Program with Eclipse

- 1 Using JDBC with MySQL, Getting Started
- 2 An Introduction to Java Annotations
- 3 An Introduction to JSP Standard Template Library (JSTL)
- 4 MIDP Programming with J2ME
- 5 Debugging a Java Program with Eclipse

- 1 Using JDBC with MySQL, Getting Started
- 2 An Introduction to Java Annotations
- 3 An Introduction to JSP Standard Template Library (JSTL)
- 4 MIDP Programming with J2ME
- 5 Debugging a Java Program with Eclipse

Most Commented On

- This Week
- This Month
- All-Time

- 1 10 Experimental PHP Projects Pushing the Envelope
- 2 Day 1: Learning the Basics of PL/SQL
- 3 C# Tip: Placing Your C# Application in the

System Tray
- 4 Logical Versus Physical Database Modeling
- 5 Is Ubuntu Contributing as Much as It Should to Free Software Projects?

- 1 Day 1: Learning the Basics of PL/SQL
- 2 The 5 Developer Certifications You'll Wish You Had in 2015
- 3 10 Experimental PHP Projects Pushing the Envelope
- 4 An Introduction to Struts
- 5 Inside Facebook's Open Source Infrastructure

- 1 Creating Use Case Diagrams
- 2 Day 1: Learning the Basics of PL/SQL
- 3 C# Tip: Placing Your C# Application in the System Tray
- 4 Using ASP.NET To Send Email
- 5 Using JDBC with MySQL, Getting Started

Recommended Partner Resources

- Cloud Computing Showcase for Developers
- Mobile Development Center
- HTML 5 Development Center

Sitemap | Contact Us

Thanks for your registration, follow us on our social networks to keep up-to-date