c#编写的高并发数据库控制访问代码

投稿：hebedich 字体：[增加 减小] 类型：转载 时间：2015-03-17 我要评论

> 往往大数据量,高并发时, 瓶颈都在数据库上, 好多人都说用数据库的复制,发布, 读写分离等技术, 但主从数据库之间同步时间有延迟.

代码的作用在于保证在上端缓存服务失效（一般来说概率比较低）时，形成倒瓶颈，从而能够保护数据库，数据库宕了，才是大问题（比如影响其他应用）。

假设（非完全正确数据，仅做示例）：
每秒支持10,000,000次查询(千万);
一次读库需要耗时：1ms;
修改内存变量需要耗时：0.001ms;
那么：
每秒最终访问的数据库的请求数量 < 1000
其他的9,900,000个请求会返回到其他页面。这就是为啥很多抢单网站有人可以访问，而有人得到繁忙中页面的原因。

微观到1ms来看，在currentValidSessionID == -1的时间是 1ms，从而平均会有10000条记录涌入。
currentValidSessionID从-1变为其他值的时间为0.001ms，这个时间内，

代码如下: 复制代码

```
lock (databaseDoor)
{
  // now there is only one request can reach below codes.
  if (currentValidSessionID == -1)
  {
    currentValidSessionID = currentRequest.SessionID;
  }
}
```

平均会有 10000×0.001=10条记录会执行到上述这段代码，操作系统会为锁形成等待序列。
那么我们的目标是，每毫秒只允许一次读库（因为其他应用也会使用）,所以我们只希望这进入的10条，最终只有一条能够继续前进。
那么这就是

代码如下: 复制代码

```
if (currentValidSessionID == -1)
{
}
```

的作用了。再次进行一次判断，进入原子保护队列的请求，也只有一个能够继续。

一点思考：
其实对于一个主频能上N GHz的服务器来说，一个内存数赋值给另一个内存数据就是1～4条指令(平均2条，两次MOV操作)，也就是2/N ns时间，而不是我们上述假设的 1000ns（0.001ms）。其实不用原子，我们已经可以把千亿级请求的访问数控制在个位数。

大家感兴趣的内容

1 C#几种截取字符串的方法小结
2 c#实现16进制和字符串之间转换的
3 C# 一个WCF简单实例
4 C# Stream 和 byte[] 之间的转换
5 C# DataGridView添加新行的2个方
6 C#中List（string）和string[]数
7 C# 16进制与字符串、字节数组之间
8 C# WORD操作实现代码
9 C#连接MySql数据库的方法
10 使用VS2010 C#开发ActiveX控件（

最近更新的内容

C#中Equality和Identity浅析
DataGridView控件显示行号的正确代码及
c#读取excel内容内容示例分享
c#中XML解析文件出错解决方法
C#类的多态性详解
C#实现让ListBox适应最大Item宽度的方法
C#使用foreach遍历哈希表（hashtable）
C#的path.GetFullPath 获取上级目录实现
C#实现农历日历的方法
详解C#编程中一维数组与多维数组的使用

常用在线小工具

不过一个架构师，如果可以用一个99.99%安全的方案，就绝对不用99.9%。 SO。

代码如下: 复制代码

```
public static long currentValidSessionID = -1;
public static object databaseDoor = new object();
void readDatabase(Request currentRequest)
{
   // use currentValidSessionID to filter out other requests came in during the execute time gap
   if (currentValidSessionID == -1)
   {
      // use object-lock to filter out other requests came in during the variable change time gap.
      lock (databaseDoor)
      {
         // now there is only very little number of requests can reach below codes.
         if (currentValidSessionID == -1)
         {   // now there will be only one request can access the database
            currentValidSessionID = currentRequest.SessionID;
         }
      }
   }
   if (currentValidSessionID == currentRequest.SessionID)
   {   // here is the one !
      try
      {
         // use transaction to guarantee the execute time to void block
         // access database codes go here
      }
      catch()
      {
         // exception codes go here
      }
      finally
      {
         currentValidSessionID = -1;  // recover to original state
      }
   }
}
```

以上就是本文所述的全部内容了，希望对大家学习C#的高并发编程能够有所帮助。

您可能感兴趣的文章:

让Win2008+IIS7+ASP.NET支持10万并发请求

c#实现服务器性能监控并发送邮件保存日志

C#线程执行超时处理与并发线程数控制实例

C#使用队列(Queue)解决简单的并发问题

在ASP.NET 2.0中操作数据之二十一：实现开放式并发

在ASP.NET 2.0中操作数据之四十四：DataList和Repeater数据排序（三）

在ASP.NET 2.0中操作数据之四十五：DataList和Repeater里的自定义Button

在ASP.NET 2.0中操作数据之四十六：使用SqlDataSource控件检索数据

在ASP.NET 2.0中操作数据之四十七：用SqlDataSource控件插入、更新、删除数据

在ASP.NET 2.0中操作数据之四十八：对SqlDataSource控件使用开放式并发

Tags: C# 高并发 数据库

相关文章

介绍C# 泛型类在使用中约束 2013-09-09

c#匹配整数和小数的正则表达式 2010-12-12
C#中foreach语句使用break暂停遍历的方法 2015-04-04
winfrom 打印表格 字符串的封装实现代码 附源码下载 2013-02-02
C# 常用日期时间函数（老用不熟） 2009-09-09
WinForm实现基于BindingSource的方法扩展 2014-08-08
C#二进制序列化实例分析 2015-05-05
C#中常量和只读变量的区别小结 2014-01-01
LZW压缩算法 C#源码 2016-06-06

最新评论