

专注java开发

一个拿着底层薪资操着架构师的心码农

博客园

首页

新随笔

联系

订阅

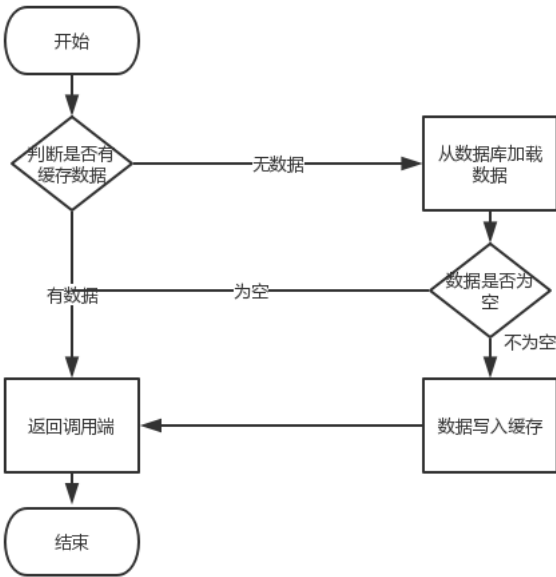
管理

【原创】分布式之数据库和缓存双写一致性方案解析

引言

为什么写这篇文章？

首先，缓存由于其高并发和高性能的特性，已经在项目中被广泛使用。在读取缓存方面，大家没啥疑问，都是按照下图的流程来进行业务操作。



但是在更新缓存方面，对于更新完数据库，是更新缓存呢，还是删除缓存。又或者是先删除缓存，再更新数据库，其实大家存在很大的争议。目前没有一篇全面的博客，对这几种方案进行解析。于是博主战战兢兢，顶着被大家喷的风险，写了这篇文章。

文章结构

本文由以下三个部分组成

- 1、讲解缓存更新策略
- 2、对每种策略进行缺点分析
- 3、针对缺点给出改进方案

正文

先做一个说明，从理论上来说，给缓存设置过期时间，是保证最终一致性的解决方案。这种方案下，我们可以对存入缓存的数据设置过期时间，所有的写操作以数据库为准，对缓存操作只是尽最大努力即可。也就是说如果数据库写成功，缓存更新失败，那么只要到达过期时间，则后面的读请求自然会从数据库中读取新值然后回填缓存。因此，接下来讨论的思路不依赖于给缓存设置

公告



我的公众号(孤独烟)，大夏正还是博客园和公众

昵称：孤独烟
园龄：4年1个月
粉丝：1179
关注：2

+加关注

搜索

随笔分类

JAVA基础(5)

版权声明(1)

分布式架构(22)

面试解析(1)

手把手系列(7)

过期时间这个方案。
在这里，我们讨论**三种**更新策略：

- 1. 先更新数据库，再更新缓存
- 2. 先删除缓存，再更新数据库
- 3. 先更新数据库，再删除缓存

应该没人问我，为什么没有先更新缓存，再更新数据库这种策略。

(1)先更新数据库，再更新缓存

这套方案，大家是普遍反对的。为什么呢？有如下两点原因。

原因一（线程安全角度）

同时有请求A和请求B进行更新操作，那么会出现

- (1) 线程A更新了数据库
- (2) 线程B更新了数据库
- (3) 线程B更新了缓存
- (4) 线程A更新了缓存

这就出现请求A更新缓存应该比请求B更新缓存早才对，但是因为网络等原因，B却比A更早更新了缓存。这就导致了脏数据，因此不考虑。

原因二（业务场景角度）

有如下两点：

- (1) 如果你是一个写数据库场景比较多，而读数据场景比较少的业务需求，采用这种方案就会导致，数据压根还没读到，缓存就被频繁的更新，浪费性能。
- (2) 如果你写入数据库的值，并不是直接写入缓存的，而是要经过一系列复杂的计算再写入缓存。那么，每次写入数据库后，都再次计算写入缓存的值，无疑是浪费性能的。显然，删除缓存更为适合。

接下来讨论的就是争议最大的，先删缓存，再更新数据库。还是先更新数据库，再删缓存的问题。

(2)先删缓存，再更新数据库

该方案会导致不一致的原因是。同时有一个请求A进行更新操作，另一个请求B进行查询操作。那么会出现如下情形：

- (1) 请求A进行写操作，删除缓存
- (2) 请求B查询发现缓存不存在
- (3) 请求B去数据库查询得到旧值
- (4) 请求B将旧值写入缓存
- (5) 请求A将新值写入数据库

上述情况就会导致不一致的情形出现。而且，如果不采用给缓存设置过期时间策略，该数据永远都是脏数据。

那么，**如何解决呢？采用延时双删策略**

伪代码如下

```
public void write(String key,Object data){
    redis.delKey(key);
    db.updateData(data);
    Thread.sleep(1000);
    redis.delKey(key);
}
```

转化为中文描述就是

- (1) 先淘汰缓存
- (2) 再写数据库（这两步和原来一样）
- (3) 休眠1秒，再次淘汰缓存

这么做，可以将1秒内所造成的缓存脏数据，再次删除。

那么，这个1秒怎么确定的，具体该休眠多久呢？

针对上面的情形，读者应该自行评估自己的项目的读数据业务逻辑的耗时。然后写数据的休眠时间则在读数据业务逻辑的耗时基础上，加几百ms即可。这么做的目的，就是确保读请求结束，写请求可以删除读请求造成的缓存脏数据。

如果你用了mysql的读写分离架构怎么办？

ok，在这种情况下，造成数据不一致的原因如下，还是两个请求，一个请求A进行更新操作，另一个请求B进行查询操作。

- (1) 请求A进行写操作，删除缓存
- (2) 请求A将数据写入数据库了，
- (3) 请求B查询缓存发现，缓存没有值
- (4) 请求B去从库查询，这时，还没有完成主从同步，因此查询到的是旧值
- (5) 请求B将旧值写入缓存
- (6) 数据库完成主从同步，从库变为新值

上述情形，就是数据不一致的原因。还是使用双删延时策略。只是，睡眠时间修改为在主从同步的延时时间基础上，加几百ms。

积分与排名

积分 - 117977

排名 - 3725

最新评论

1. Re:【原创】小说:我拜读。

阅读排行榜

1. 【原创】分布式之数致性方案解析(23465)
2. 【原创】分布式之re70)
3. 【原创】分布式之消(19921)

推荐排行榜

1. 【原创】分布式之消(235)
2. 【原创】分布式之re0)
3. 【原创】分布式之数致性方案解析(155)

采用这种同步淘汰策略，吞吐量降低怎么办？

ok，那就将第二次删除作为异步的。自己起一个线程，异步删除。这样，写的请求就不用沉睡一段时间了，再返回。这么做，加大吞吐量。

第二次删除,如果删除失败怎么办？

这是个非常好的问题，因为第二次删除失败，就会出现如下情形。还是有两个请求，一个请求A进行更新操作，另一个请求B进行查询操作，为了方便，假设是单库：

- (1) 请求A进行写操作，删除缓存
- (2) 请求B查询发现缓存不存在
- (3) 请求B去数据库查询得到旧值
- (4) 请求B将旧值写入缓存
- (5) 请求A将新值写入数据库
- (6) 请求A试图去删除请求B写入对缓存值，结果失败了。

ok,这也就是说。如果第二次删除缓存失败，会再次出现缓存和数据库不一致的问题。

如何解决呢？

具体解决方案，且看博主对第(3)种更新策略的解析。

(3)先更新数据库，再删缓存

首先，先说一下。老外提出了一个缓存更新套路，名为《[Cache-Aside pattern](#)》。其中就指出

- **失效**：应用程序先从cache取数据，没有得到，则从数据库中取数据，成功后，放到缓存中。
- **命中**：应用程序从cache中取数据，取到后返回。
- **更新**：先把数据存到数据库中，成功后，再让缓存失效。

另外，知名社交网站facebook也在论文《[Scaling Memcache at Facebook](#)》中提出，他们用的也是先更新数据库，再删缓存的策略。

这种情况不存在并发问题么？

不是的。假设这会有两个请求，一个请求A做查询操作，一个请求B做更新操作，那么会有如下情形产生

- (1) 缓存刚好失效
- (2) 请求A查询数据库，得一个旧值
- (3) 请求B将新值写入数据库
- (4) 请求B删除缓存
- (5) 请求A将查到的旧值写入缓存

ok，如果发生上述情况，确实是会发生脏数据。

然而，发生这种情况的概率又有多少呢？

发生上述情况有一个先天性条件，就是步骤（3）的写数据库操作比步骤（2）的读数据库操作耗时更短，才有可能使得步骤

（4）先于步骤（5）。可是，大家想想，数据库的读操作的速度远快于写操作的（不然做读写分离干嘛，做读写分离的意义就是因为读操作比较快，耗资源少），因此步骤（3）耗时比步骤（2）更短，这一情形很难出现。

假设，有人非要抬杠，有强迫症，一定要解决怎么办？

如何解决上述并发问题？

首先，给缓存设有效时间是一种方案。其次，采用策略（2）里给出的异步延时删除策略，保证读请求完成以后，再进行删除操作。

还有其他造成不一致的原因么？

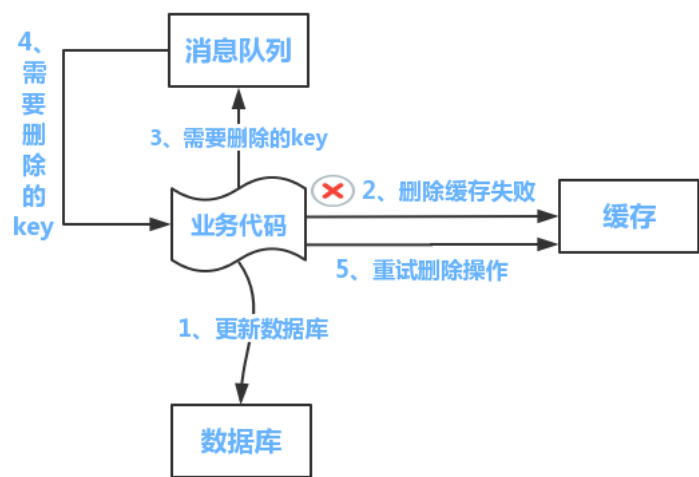
有的，这也是缓存更新策略（2）和缓存更新策略（3）都存在的一个问题，如果删缓存失败了怎么办，那不是会有不一致的情况出现么。比如一个写数据请求，然后写入数据库了，删缓存失败了，这会就出现不一致的情况了。这也是缓存更新策略（2）里留下的最后一个疑问。

如何解决？

提供一个保障的重试机制即可，这里给出两套方案。

方案一：

如下图所示

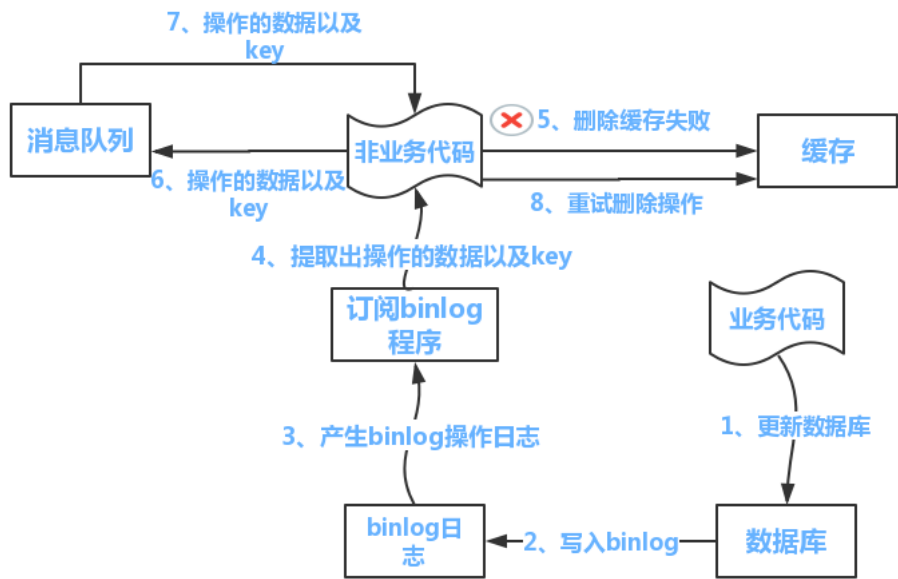


流程如下所示

- (1) 更新数据库数据；
- (2) 缓存因为种种问题删除失败
- (3) 将需要删除的key发送至消息队列
- (4) 自己消费消息，获得需要删除的key
- (5) 继续重试删除操作，直到成功

然而，该方案有一个缺点，对业务线代码造成大量的侵入。于是有了方案二，在方案二中，启动一个订阅程序去订阅数据库的binlog，获得需要操作的数据。在应用程序中，另起一段程序，获得这个订阅程序传来的信息，进行删除缓存操作。

方案二：



流程如下图所示：

- (1) 更新数据库数据
- (2) 数据库会将操作信息写入binlog日志当中
- (3) 订阅程序提取出所需要的数据以及key
- (4) 另起一段非业务代码，获得该信息
- (5) 尝试删除缓存操作，发现删除失败
- (6) 将这些信息发送至消息队列
- (7) 重新从消息队列中获得该数据，重试操作。

备注说明：上述的**订阅binlog程序**在mysql中有现成的中间件叫canal，可以完成订阅binlog日志的功能。至于oracle中，博主目前不知道有没有现成中间件可以使用。另外，重试机制，博主采用的是消息队列的方式。如果对一致性要求不是很高，直接在程序中另起一个线程，每隔一段时间去重试即可，这些大家可以灵活自由发挥，只是提供一个思路。

总结

本文其实是对目前互联网中已有的一致性方案，进行了一个总结。对于先删缓存，再更新数据库的更新策略，还有方案提出维护一个内存队列的方式，博主看了一下，觉得实现异常复杂，没有必要，因此没有必要在文中给出。最后，希望大家有所收获。

参考文献

- 1、主从DB与cache一致性
- 2、缓存更新的套路

作者：孤独烟 出处：<http://rjzheng.cnblogs.com/>
本文版权归作者和博客园共有，欢迎转载，但未经作者同意必须保留此段声明，且在文章页面明显位置给出原文连接，否则保留追究法律责
任的权利。如果觉得还有帮助的话，可以点一下右下角的【推荐】。

分类： 分布式架构

好文要顶

关注我

收藏该文

孤独烟

关注 - 2

粉丝 - 1179

+加关注

« 上一篇：【原创】分布式之消息队列复习精讲
» 下一篇：【原创】分布式之redis复习精讲

posted @ 2018-05-15 17:15 孤独烟 阅读(23465) 评论(117) 编辑 收藏

评论列表

#101楼[楼主] 2018-07-02 15:37 孤独烟

@ 张桦林的小号
还有小号。。666

支持(0) 反对(0)

#102楼 2018-07-29 20:23 man_or_superman

你好，如果方案1中的第3步失败了怎么办？还是又其他办法保证？请教

支持(0) 反对(0)

#103楼[楼主] 2018-07-29 20:25 孤独烟

@ man_or_superman
你问的应该是删更删这个策略吧

支持(0) 反对(0)

#104楼 2018-07-29 20:26 man_or_superman

@ 孤独烟
就是你提供的两个解决方案中的第一个

支持(0) 反对(0)

#105楼 2018-07-29 20:32 man_or_superman

方案一
https://images.cnblogs.com/cnblogs_com/rjzheng/1202350/o_update1.png
流程如下所示
(1) 更新数据库数据；
(2) 缓存因为种种问题删除失败
(3) 将需要删除的key发送至消息队列
(4) 自己消费消息，获得需要删除的key

- (5) 继续重试删除操作，直到成功

就是这个，第3步没有将删除的key发送到消息队列里面 该怎么处理？

支持(0) 反对(0)
- #106楼[楼主] 2018-07-29 20:58 孤独烟

@ man_or_superman

看我的另外一篇，分布式之消息队列复习精讲，有一个如何保证可靠性的

支持(0) 反对(0)
- #107楼 2018-07-29 21:19 man_or_superman

@ 孤独烟

虽然保证mq的高可靠性，但依然不能保证百分百的发送数据到mq成功，其实我更想问的是(1)和(3)的这两步的原子性。如何一定保证(3)成功。因为在（1）中 数据库已经更新。所以感觉（1）（3）必须保证在一个事务中才能做到。

支持(0) 反对(0)
- #108楼[楼主] 2018-07-29 21:22 孤独烟

@ man_or_superman

并不能保证一定成功。。只能降低发生几率。。

支持(0) 反对(0)
- #109楼 2018-07-29 21:35 man_or_superman

@ 孤独烟

感觉方案一，就像两军问题一样，cache操作失败的话，用mq来保证，mq失败的话，就没有东西保证了。有点像两军问题。
感谢作者提供方案二,谢谢

支持(0) 反对(0)
- #110楼 2018-08-13 14:04 wintersoft

@ man_or_superman

引用

@孤独烟

感觉方案一，就像两军问题一样，cache操作失败的话，用mq来保证，mq失败的话，就没有东西保证了。有点像两军问题。
感谢作者提供方案二,谢谢

这种属于最终一致性解决方案，重试到一定次数还是失败需要报警去人工处理。LZ的这套方案基于mq来实现，mq肯定不能出问题，mq做集群出问题的几率并不大，一般企业没那么大的业务量。

支持(0) 反对(0)

#111楼[楼主] 2018-08-13 14:06 孤独烟

@ geek_power

明白人。。。话说，你看下。。分布式之数据库和缓存双写一致性方案解析（三）
。。。我的观点一致和沈剑不一样。

支持(0) 反对(0)

#112楼 2018-09-18 10:11 关河冷断

博主所说的3方案，也存在mysql读写分离，读操作读取滞后值，更新缓存错误的情况吧？

支持(0) 反对(0)

#113楼 2018-09-18 10:13 关河冷断

是否考虑下这种方案，后台进程订阅binlog日志，业务线程只做缓存读取操作，缓存删除和更新交给订阅binlog日志的线程去做？

支持(0) 反对(0)

#114楼[楼主] 2018-09-18 10:25 孤独烟
- 155

6
- https://www.cnblogs.com/rjzheng/p/9041659.html

6/7

@ 关河冷断

(1)A线程写入主库,从库未同步，A线程删缓存。
(2)B线程发现缓存没东西，去从库读，然后将从库的数据set进缓存
(3)主从同步完成。

步骤二。发现缓存没东西，强制读主库。。或者代码逻辑判断，发现缓存没东西，延时读就行。

支持(0) 反对(0)

#115楼 2018-09-18 10:30 关河冷断

@ 孤独烟

明白了

支持(0) 反对(0)

#116楼 2018-11-05 16:56 尘世中一只迷途的小羔羊

针对第三种解决方案可以这样吗，先更新数据库，然后删除缓存，如果缓存删除失败则回滚数据库的数据，这样可行吗

支持(0) 反对(0)

#117楼 2019-03-28 23:15 正先生

一直采用的场景一，笑哭。。。

支持(0) 反对(0)

< Prev

1

2

3

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

- 【推荐】超50万C++/C#源码: 大型实时仿真组态图形源码
- 【培训】IT职业生涯指南，Java程序员薪资翻3倍的秘密
- 【推荐】工作996，生病ICU，程序员不加班就没前途吗？
- 【推荐】专业便捷的企业级代码托管服务 - Gitee 码云

