[https://mp.weixin.qq.com/s?\_\_biz=MzA5MTkxMDQ4MQ==&mid=2648933452&idx=1&sn=48b3b1cbd27c50186122fef8943eca5f&chksm=88621c72bf159564e629ee77d180424274ae9effd8a7c2997f853135b28f3401970793d8098d&token=1919005508&lang=zh\_CN&scene=21#wechat\_redirect](https://mp.weixin.qq.com/s?__biz=MzA5MTkxMDQ4MQ==&mid=2648933452&idx=1&sn=48b3b1cbd27c50186122fef8943eca5f&chksm=88621c72bf159564e629ee77d180424274ae9effd8a7c2997f853135b28f3401970793d8098d&token=1919005508&lang=zh_CN&scene=21" \l "wechat_redirect)

# 聊聊db和缓存一致性常见的实现方式

数据存储在数据库中，为了加快业务访问的速度，我们将数据库中的一些数据放在缓存中，那么问题来了，如何确保db和缓存中数据的一致性呢？我们列出了5种方法，大家都了解一下，然后根据业务自己选择。

### 方案1

#### **获取缓存逻辑**

使用过定时器，定时刷新redis中的缓存。

#### **db更新数据逻辑**

更新数据不用考虑缓存中的数据，直接更新数据就可以了

#### **存在的问题**

缓存中数据和db中数据一致性可能没有那么及时，不过最终在某个时间点，数据是一致的。

### 方案2

#### **获取缓存逻辑**

c1：根据key在redis中获取对应的value  
c2：如果value存在，直接返回value；若value不存在，继续下面步骤  
c3：从数据库获取值，赋值给value，然后将key->value放入redis，返回value

#### **更新db逻辑**

u1：开始db事务  
u2：更新数据  
u3：提交db事务  
u4：删除redis中当前数据的缓存

#### **存在的问题**

1、上面u3成功，u4失败，会导致删除缓存失败，导致缓存中数据和db数据会不一致。

2、如果同时有很多线程到达c2发现缓存不存在，同时请求c3访问db，会对db造成很大的压力

### 方案3

#### **获取缓存逻辑**

c1：根据key在redis中获取对应的value  
c2：如果value存在，直接返回value；若value不存在，继续下面步骤  
c3：从数据库获取值，赋值给value，然后将key->value放入redis，返回value

#### **更新db逻辑**

u1：删除redis中当前数据的缓存  
u2：开始db事务  
u3：更新数据  
u4：提交db事务

#### **存在的问题**

1、更新数据的线程执行u1成功之后，u2还未执行时，此时获取缓存的线程刚好执行了c1到c3的逻辑，此时会将旧的数据放入redis，导致redis和db数据不一致

2、同样存在方案2中说到的问题：如果同时有很多线程到达c2发现缓存不存在，同时请求c3访问db，会对db造成很大的压力

#### **方案4**

对方案2做改进，确保db更新成功之后，删除缓存操作一定会执行，我们可以通过可靠消息来实现，可靠消息可以确保更新db操作和删除redis中缓存最终要么都成功要么都失败，依靠的是最终一致性来实现的。

改进之后过程如下。

#### **获取缓存逻辑**

c1：根据key在redis中获取对应的value  
c2：如果value存在，直接返回value；若value不存在，继续下面步骤  
c3：从数据库获取值，赋值给value，然后将key->value放入redis，返回value

#### **更新db逻辑**

u1：开始db事务  
u2：更新数据  
u3：投递删除redis缓存的消息  
u4：提交db事务

#### **消息消费者-清理redis缓存的消费者**

接受到清理redis缓存的消息之后，将redis中对应的缓存清除。

#### **存在的问题**

1、更新db和清理redis中的缓存之间存在一定的时间延迟，这段时间内，redis缓存的数据是旧的，也就是说这段时间内db和缓存数据是不一致的，但是最终会一致，这个不一致的时间可能比较小（这个需要看消息消费的效率了）

2、同样存在方案2中说到的问题：如果同时有很多线程到达c2发现缓存不存在，同时请求c3访问db，会对db造成很大的压力

### 方式5

我们先了解一些知识。

#### **redis中几个方法**

##### **get(key)**

获取key的值，如果存在，则返回；如果不存在，则返回nil

##### **setnx(key,value)**

setnx的含义就是SET if Not Exists，该方法是原子的，如果key不存在，则设置当前key成功，返回1；如果当前key已经存在，则设置当前key失败，返回0

##### **del(key)**

将key对应的值从redis中删除

#### **数据库相关知识**

select v from t where t.key = #key# for update;

update  t set v = #v# where t.key = #key#;

上面两个sql会相互阻塞，直到其中一个提交之后，另外一个才可以继续执行。

下面我们就通过上面的知识来实现db和缓存强一致性。

#### **更新数据逻辑**

1. 打开db事务  
   **2**.update t set v = #v# where t.key = #key#;  
   **3**.根据key删除redis中的缓存：RedisUtil.del(key);  
   **4**.提交db事务

#### **获取缓存逻辑**



