

03 | 复杂而又重要的购物车系统，应该如何设计？

2020-03-03 李玥

后端存储实战课

[进入课程 >](#)



讲述：李玥

时长 15:52 大小 12.73M



你好，我是李玥。

今天这节课我们来说一下购物车系统的存储该如何设计。

首先，我们来看购物车系统的主要功能是什么。就是在用户选购商品时，下单之前，暂存用户想要购买的商品。购物车对数据可靠性要求不高，性能也没有特别的要求，在整个电商系统中，看起来是相对比较容易设计和实现的一个子系统。



购物车系统的功能，主要的就三个：把商品加入购物车（后文称“加购”）、购物车列表页、发起结算下单，再加上一个在所有界面都要显示的购物车小图标。

支撑购物车的这几个功能，对应的存储模型应该怎么设计？很简单，只要一个“购物车”实体就够了。它的主要属性有什么？你打开京东的购物车页面，对着抄就设计出来了：SKUID（商品 ID）、数量、加购时间和勾选状态。

购物车

自营

搜索

全部商品 3

配送至: 北京通州区马驹桥镇

<input type="checkbox"/> 全选	商品	单价	数量	小计	操作
<input checked="" type="checkbox"/>	<div><div>京东好店 晟发玩具专营店</div><div>满减 活动商品购满¥99.00, 即可享受满减 去凑单 ></div><div></div><div>合金飞机玩具A380空中客机战斗机直升飞机20仿真模型声光回力儿童玩具宝</div><div>选服务</div></div>	¥46.00	1	¥46.00	删除 移到我的关注
<input type="checkbox"/>	<div><div>京东好店 宝贝星玩具旗舰店</div><div>【可充电】立昕托马斯小火车轨道车儿童玩具多层超大电动过山</div><div>选服务</div></div>	¥119.00	1	¥119.00	删除 移到我的关注
<input checked="" type="checkbox"/>	<div><div>微软升昌专卖店</div><div>微软 (Microsoft) Surface Pro 6 平板电脑笔记本二合一 轻薄便携商务办</div><div>选服务</div></div>	¥7738.00	1	¥7738.00	删除 移到我的关注

☐ 全选 删除选中商品 移到关注 清理购物车

已选择 2 件商品 总价: ¥7784.00 促销: -¥0.00

去结算

备注：图片来源于网络，仅供本文介绍、评论及说明某问题，适当引用。

这个“勾选状态”属性，就是在购物车界面中，每件商品前面的那个小对号，表示在结算下单时，是不是要包含这件商品。至于商品价格和总价、商品介绍等等这些信息，都可以实时从其他系统中获取，不需要购物车系统来保存。

购物车的功能虽然很简单，但是在设计购物车系统的存储时，仍然有一些特殊的问题需要考虑。

设计购物车存储时需要把握什么原则？

比如下面这几个问题：

1. 用户没登录，在浏览器中加购，关闭浏览器再打开，刚才加购的商品还在不在？
2. 用户没登录，在浏览器中加购，然后登录，刚才加购的商品还在不在？
3. 关闭浏览器再打开，上一步加购的商品在不在？
4. 再打开手机，用相同的用户登录，第二步加购的商品还在不在呢？

上面这几个问题是不是有点儿绕？没关系，我们先简单解释一下这四个问题：

1. 如果用户没登录，加购的商品也会被保存在用户的电脑里，这样即使关闭浏览器再打开，购物车的商品仍然存在。
2. 如果用户先加购，再登录，登录前加购的商品就会被自动合并到用户名下，所以登录后购物车中仍然有登录前加购的商品。
3. 关闭浏览器再打开，这时又变为未登录状态，但是之前未登录时加购的商品已经被合并到刚刚登录的用户名下了，所以购物车是空的。
4. 使用手机登录相同的用户，看到的的就是该用户的购物车，这时无论你在手机 App、电脑还是微信中登录，只要是相同的用户，看到是同一个购物车，所以第二步加购的商品是存在的。

所以，上面这四个问题的答案依次是：存在、存在、不存在、存在。

如果你没有设计或者开发过购物车系统，你可能并不会想到购物车还有这么多弯弯绕。但是，作为一个开发者，如果你不仔细把这些问题考虑清楚，用户在使用购物车的时候，就会感觉你的购物车系统不好用，不是加购的商品莫名其妙地丢了，就是购物车莫名其妙地多出一些商品。

要解决上面这些问题，其实只要在存储设计时，把握这几个原则就可以了：

1. 如果未登录，需要临时暂存购物车的商品；
2. 用户登录时，把暂存购物车的商品合并到用户购物车中，并且清除暂存购物车；
3. 用户登陆后，购物车中的商品，需要在浏览器、手机 APP 和微信等等这些终端中都保持同步。

实际上，购物车系统需要保存两类购物车，一类是未登录情况下的“暂存购物车”，一类是登录后的“用户购物车”。

如何设计“暂存购物车”的存储？

我们先来看下暂存购物车的存储该怎么实现。暂存购物车应该存在客户端还是存在服务端？

如果保存在服务端，那每个暂存购物车都需要有一个全局唯一的标识，这个标识并不太容易设计，并且，存在服务端还要浪费服务端的资源。所以，肯定是保存在客户端好，既可以节约服务器的存储资源，也没有购物车标识的问题，因为每个客户端就保存它自己唯一一个购物车就可以了，不需要标识。

客户端的存储可以选择的不太多：Session、Cookie 和 LocalStorage，其中浏览器的 LocalStorage 和 App 的本地存储是类似的，我们都以 LocalStorage 来代表。

存在哪儿最合适？SESSION 是不太合适的，原因是，SESSION 的保留时间短，而且 SESSION 的数据实际上还是保存在服务端的。剩余的两种存储，Cookie 和 LocalStorage 都可以用来保存购物车数据，选择哪种方式更好呢？各有优劣。

在我们这个场景中，使用 Cookie 和 LocalStorage 最关键的区别是，客户端和服务端的每次交互，都会自动带着 Cookie 数据往返，这样服务端可以读写客户端 Cookie 中的数据，而 LocalStorage 里的数据，只能由客户端来访问。

使用 Cookie 存储，实现起来比较简单，加减购物车、合并购物车的过程中，由于服务端可以读写 Cookie，这样全部逻辑都可以在服务端实现，并且客户端和服务端请求的次数也相对少一些。

使用 LocalStorage 存储，实现相对就复杂一点儿，客户端和服务端都要实现一些业务逻辑，但 LocalStorage 的好处是，它的存储容量比 Cookie 的 4KB 上限要大得多，而且不用像 Cookie 那样，无论用不用，每次请求都要带着，可以节省带宽。

所以，选择 Cookie 或者是 LocalStorage 来存储暂存购物车都是没问题的，你可以根据它俩各自的优劣势来选择。比如你设计的是个小型电商，那用 Cookie 存储实现起来更简单。再比如，你的电商是面那种批发的行业用户，用户需要加购大量的商品，那 Cookie 可能容量不够用，选择 LocalStorage 就更合适。

不管选择哪种存储，暂存购物车保存的数据格式都是一样的，参照我们实体模型来设计就可以，我们可以直接用 JSON 表示：

复制代码

```
1 {
2   "cart": [
3     {
4       "SKUID": 8888,
5       "timestamp": 1578721136,
6       "count": 1,
7       "selected": true
8     },
9     {
10      "SKUID": 6666,
11      "timestamp": 1578721138,
12      "count": 2,
13      "selected": false
14    }
15  ]
16 }
```


如何设计“用户购物车”的存储？

接下来，我们再来看下用户购物车的存储该怎么实现。因为用户购物车必须要保证多端的数据同步，所以数据必须保存在服务端。常规的思路是，设计一张购物车表，把数据存在 MySQL 中。这个表的结构同样可以参照刚刚讲的实体模型来设计：

列名	数据类型	主键	非空	说明
id	BIGINT	是	是	自增主键
user_id	BIGINT		是	用户ID
sku_id	BIGINT		是	商品ID
count	INT		是	商品数量
timestamp	DATE		是	加购时间
selected	TINYINT (1)			购选状态

注意，需要在 user_id 上建一个索引，因为查询购物车表时，都是以 user_id 作为查询条件来查询的。

你也可以选择更快的 Redis 来保存购物车数据，以用户 ID 作为 Key，用一个 Redis 的 HASH 作为 Value 来保存购物车中的商品。比如：

 复制代码

```
1 {
2     "KEY": 6666,
3     "VALUE": [
4         {
5             "FIELD": 8888,
6             "FIELD_VALUE": {
7                 "timestamp": 1578721136,
8                 "count": 1,
9                 "selected": true
10            }
11        },
12        {
13            "FIELD": 6666,
14            "FIELD_VALUE": {
15                "timestamp": 1578721138,
16                "count": 2,
17                "selected": false
18            }
19        }
20    ]
21 }
```

这里为了便于你理解，我们用 JSON 来表示 Redis 中 HASH 的数据结构，其中 KEY 中的值 6666 是一个用户 ID，FIELD 里存放的是商品 ID，FIELD_VALUE 是一个 JSON 字符串，保存加购时间、商品数量和勾选状态。

大家都知道，从读写性能上来说，Redis 是比 MySQL 快非常多的，那是不是用 Redis 就一定比用 MySQL 更好呢？我们来比较一下使用 MySQL 和 Redis 两种存储的优劣势：

1. 显然使用 Redis 性能要比 MySQL 高出至少一个量级，响应时间更短，可以支撑更多的并发请求，“天下武功，唯快不破”，这一点 Redis 完胜。
2. MySQL 的数据可靠性是要好于 Redis 的，因为 Redis 是异步刷盘，如果出现服务器掉电等异常情况，Redis 是有可能丢数据的。但考虑到购物车里的数据，对可靠性要求也没那么苛刻，丢少量数据的后果也就是，个别用户的购物车少了几件商品，问题也不大。所以，在购物车这个场景下，Redis 的数据可靠性不高这个缺点，并不是不能接受的。

3. MySQL 的另一个优势是，它支持丰富的查询方式和事务机制，这两个特性，对我们今天讨论的这几个购物车核心功能没什么用。但是，每一个电商系统都有它个性化的需求，如果需要以其他方式访问购物车的数据，比如说，统计一下今天加购的商品总数，这个时候，使用 MySQL 存储数据，就很容易实现，而使用 Redis 存储，查询起来就非常麻烦而且低效。

综合比较下来，考虑到需求总是不断变化，还是更推荐你使用 MySQL 来存储购物车数据。如果追求性能或者高并发，也可以选择使用 Redis。

你可以感受到，我们设计存储架构的过程就是一个不断做选择题的过程。很多情况下，可供选择的方案不止一套，选择的时候需要考虑实现复杂度、性能、系统可用性、数据可靠性、可扩展性等等非常多的条件。需要强调的是，**这些条件每一个都不是绝对不可以牺牲的，不要让一些“所谓的常识”禁锢了你的思维。**

比如，一般我们都认为数据是绝对不可以丢的，也就是说不能牺牲数据可靠性。但是，像刚刚讲到的用户购物车的存储，使用 Redis 替代 MySQL，就是牺牲了数据可靠性换取高性能。我们仔细分析后得出，很低概率的情况下丢失少量数据，是可以接受的。性能提升带来的收益远大于丢失少量数据而付出的代价，这个选择就是划算的。

如果说不考虑需求变化这个因素，牺牲一点点数据可靠性，换取大幅性能提升，选择 Redis 才是最优解。

小结

今天我们讲了购物车系统的存储该如何设计。

购物车系统的主要功能包括：加购、购物车列表页和结算下单。核心的实体就只有一个“购物车”实体，它至少应包括：SKUID、数量、加购时间和勾选状态这几个属性。

在给购物车设计存储时，为了确保购物车内的数据在多端保持一致，以及用户登录前后购物车内商品能无缝衔接，除了每个用户的“用户购物车”之外还要实现一个“暂存购物车”保存用户未登录时加购的商品，并在用户登录后自动合并“暂存购物车”和“用户购物车”。

暂存购物车存储在客户端浏览器或者 App 中，可以选择存放到 Cookie 或者 LocalStorage 中。用户购物车保存在服务端，可以选择使用 Redis 或者是 MySQL 存储，使用 Redis 存

储会有更高的性能，可以支撑更多的并发请求，使用 MySQL 是更常规通用的方式，便于应对变化，系统的扩展性更好。

思考题

课后请你思考一下，既然用户的购物车数据存放在 MySQL 或者是 Redis 中各有优劣势。那能不能把购物车数据存在 MySQL 中，并且用 Redis 来做缓存呢？这样不就可以兼顾两者的优势了么？这样做是不是可行？如果可行，如何来保证 Redis 中的数据和 MySQL 中的数据是一样的呢？

欢迎你在留言区与我讨论，如果你觉得今天学到的知识对你有帮助，也欢迎把它分享给你的朋友。

后端存储实战课

类电商平台存储技术应用指南

李玥

京东零售计算存储平台部资深架构师



新版升级：点击「👤请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 02 | 流量大、数据多的商品详情页系统该如何设计？

下一篇 04 | 事务：账户余额总是对不上账，怎么办？



李玥 置顶

2020-03-03

hi, 我是李玥。

上节课我给你留了一道思考题，是这样的。如果说，用户下单这个时刻，正好赶上商品调价，就有可能出现这样的情况：我明明在商详页看到的价格是10块钱，下单后，怎么变成15块了？你的系统是不是偷偷在坑我？给用户的体验非常不好。你不要以为这是一个小概...

展开 ▾

💬 7

👍 23



黄海峰

2020-03-03

感觉购物车是写多于读，也就是经常变，用cache aside的方式保持一致性的话就经常删缓存，db压力减轻不了多少，还要多写一次缓存，没什么必要

展开 ▾

💬

👍 6



公号-云原生程序员

2020-03-03

读多写少用缓存，写多读少用MQ。对于前者，前提是读场景频繁且能具备较高的命中率。用户购物车数据不符合该场景。

展开 ▾

💬

👍 5



传志

2020-03-03

购物车，同时使用redis+mysql觉得可行。以redis为主，增，查询，删除都走redis.添加加时使用mq保证最终一致性。统计等需求可以在mysql中做

展开 ▾

💬

👍 2



京京beaver

2020-03-03

购物车一般建议放到MySQL中。一般电商购物车是不占库存的，但是某些特卖电商购物车是占库存的。在这种情况下，数据是不允许丢失的，不然客户体验会非常差。Redis做缓存没啥用，因为每个用户只访问自己的购物车，每次访问网站也不会打开很多次购物车，缓存数据的命中率太低，没有意义。

展开 ▾

💬

👍 2



aoe

2020-03-03

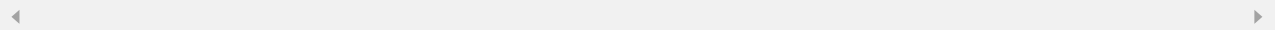
思考题是可行的，但是复杂，例如需要考虑：

1. Redis的容量可能远小于数据库容量，需要缓存策略缓存数据
2. 要处理老师提到的一致性问题
3. 性价比

...

展开 ∨

作者回复: 我个人的看法是，商品子系统存储商品快照更合理一些。



1

2



业余爱好者

2020-03-03

今日得到：以前对浏览器存储的认识只停留在cookie上。以前认为只有服务端才有session数据。以前虽然听过localStorage这个词，但是思想上没有重视。刚查了下资料，浏览器存储还有indexeddb。学习还是要系统。

存储的本质是把数据暂时或永久保存下来，单从持久化这个目标来看，什么方式存储都...

展开 ∨

1

1



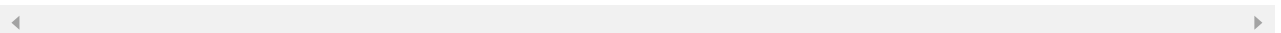
肥low

2020-03-03

我觉得完全可行 而且有时候比如MySQL主从架构下是有数据延迟更新问题的 用Redis我可以尽量避免这一点 不过有对用户加购的维护成本

展开 ∨

作者回复: 我会在下节课的评论区说一下我的理解，请关注。



1

1



墨雨

2020-03-03

不考虑复杂性和服务器成本的话，我认为是可行的。跟老师之前讲的方法一样，每次查询购物车先在 redis 里查，查不到再到 mysql 中查同时更新 redis 中数据。更新用户购物车数据时删除 redis 中数据。但我有一个问题是:用户本身购物车没数据的时候会导致 redis 和 mysql 查两遍.....

展开 ∨



大秦皇朝

2020-03-04

1、京东的浏览器端，为什么每次加完购物车都要跳转到一个中转界面上呢？这点我疑惑了很多年，从软件设计的逆向思维来考虑，我也想不出所以然来，还请李玥老师是否能给解答下呢？因为按照我们的日常使用习惯（如：阿里，京豆手机端等），都是点击加购物车，直接购物车数量加1提示就好了呀？为什么要多此一举影响用户体验呢？

2、李玥老师文稿中提到，用户的购物车偶发情况下丢失一些数据可以接受，但是站在消...
展开

作者回复: 关于你的第一个问题：“为什么每次加完购物车都要跳转到一个中转界面上呢？”，虽然我也在京东工作，我还真不知道为什么这样设计，不过不知道也好，我们还可以尝试去猜一下它为什么这样设计，如果我知道的话，可能会涉及商业秘密，反而不能回答了。

作为局外人，我的猜测是这样的，加购之后，一般用户就不会想继续看这个商详页了，接下来它可能的二个路径是：

- 1.去购物车结算；
- 2.去看其它商品；

所以，增加一个中间页，可以放好多推荐商品，引导用户继续购物，算是商家的小聪明吧。



2



知非

2020-03-04

用户访问购物车的时候一般都要操作它里面的商品了，这样看来加redis缓存避免读db意义不大。



Jxin

2020-03-04

- 1.能当然能。
- 2.能兼备mysql的可靠性和redis的读取性能。
- 3.这样做不可行，因为购物车写多读少，这样玩会频繁失效缓存，进而导致大部分读都要击穿到db并多做一步缓存的操作。实则弊大于利。
- 4.一旦修改购物车，redis的缓存直接失效。...

展开





bin

2020-03-04

存储mysql用redis做缓存这个方案不太合适，缓存的适用场景是“读多写少”的场景，因为存在两份数据经常写就会有数据不一致的风险。购物车并不适合这个场景。



博

2020-03-03

策略，更新数据时删缓存，数据一致性虽然保证了，可是如果更新购物车频繁一样会给数据库带来很大压力，所以前提假设都会成功，
使用mq更新redis，查找也是redis；至于数据库中的更新虽然会慢些，但是不影响最终一致性，扩展也不耽误。

展开 ∨



约书亚

2020-03-03

个人用购物车的感觉是，这是一个写跟读的比例差距不大的场景，不适合缓存模式，所以怀疑用redis+db这种模式的意义何在。

而且不管是cache aside，write back等等机制，理论上都有或大或小的不一致窗口时间。下单之后，后台会发现mysql购物车中不存在的商品或者数量不一致，导致下单失败。这问题出现的几率，个人觉得比redis down掉要高，给用户带来的不良体验也比购物车变...

展开 ∨



漏脚脖

2020-03-03

我觉得可以共用，而且一起用比较好

虽然写入的时候要先写mysql，再写redis，保证他们俩数据一致，这样多写一次redis

但是这样每次读就都读redis，效率会高，而且商品上有历史版本的话，加入购物车时候...

展开 ∨



刘楠

2020-03-03

可行，更新购物车的时候写mysql同时删除缓存，读的时候优先读redis，没有在打到db，同时cache一份到redis，这样应该可以保证一致性
同时，能不能每节课时把上节课的思考题解答下，谢谢



Cranliu

2020-03-03

当然是可以的。一致性的问题采用主动更新缓存解决。

展开 ∨



leslie

2020-03-03

这个问题的回答应当从两种数据库特性去说起吧：redis的特点是存储于内存，但是数据落地刷盘、、、mysql的特性是数据存储于硬盘。

由于存于内存故而查询速度非常可观：购物车环节其实商品变革的频率蛮高的，此时如果直接每次增删商品都访问硬盘数据库，这个代价就、、、尤其是在高并发场景下，真正与金额直接产生的交互的环节是结算环节，即付款；我记得老师曾经在消息队列的期中...

展开 ∨



Sephiroth

2020-03-03

应该可行，更新购物车的时候写mysql同时修改redis，读的时候优先读redis，没有在打到db，同时cache一份到redis，这样应该可以保证一致性

展开 ∨

