02-流量大、数据多的商品详情页系统该如何设计？

你好，我是李玥。

今天这节课我们看一下，如何设计一个快速、可靠的存储架构支撑商品系统。

相对于上节课提到的订单系统，电商的商品系统主要功能就是增删改查商品信息，没有很复杂的业务逻辑，支撑的主要页面就是商品详情页（下文简称：商详）。不过，设计这个系统的存储，你仍然需要着重考虑两个方面的问题。

第一，要考虑高并发的问题。不管是什么电商系统，商详页一定是整个系统中DAU（日均访问次数）最高的页面之一。这个也不难理解，用户购物么，看商详了不一定买，买之前一定会看好多商详货比三家，所以商详的浏览次数要远比系统的其他页面高。如果说，在设计存储的时候，没有考虑到高并发的问题，大促的时候，支撑商详页的商品系统必然是第一个被流量冲垮的系统。

第二，要考虑的是商品数据规模的问题。商详页的数据规模，我总结了六个字，叫：**数量多，重量大**。

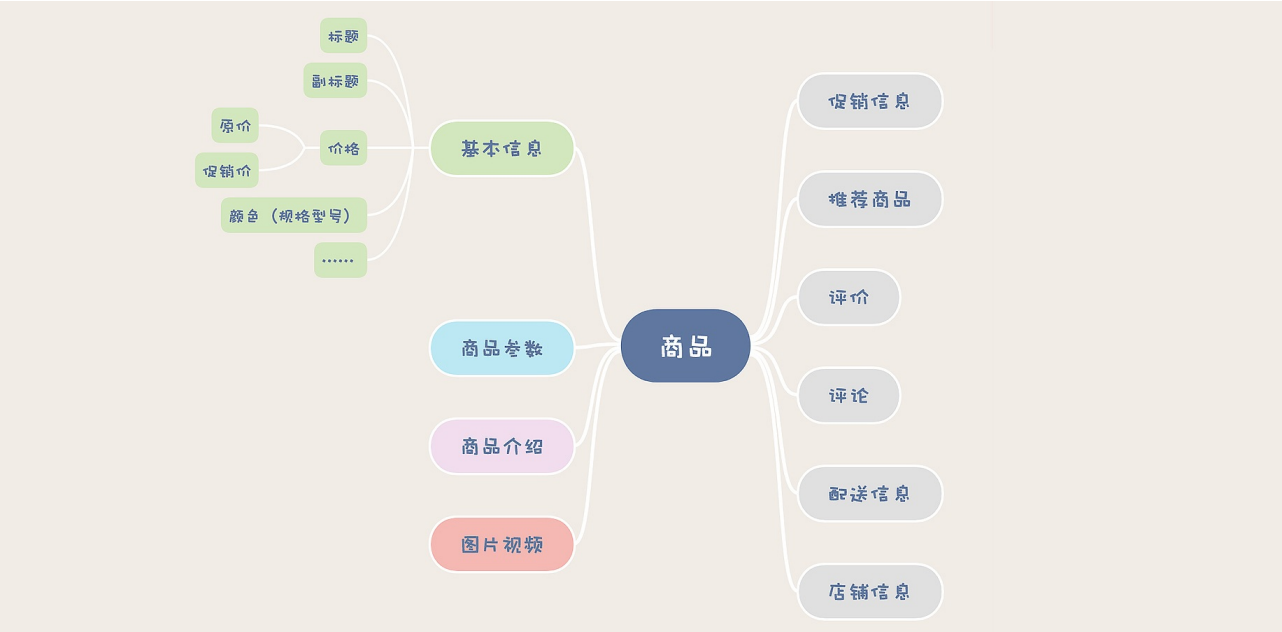
先说为什么数量多，国内一线的电商，SKU（直译为：库存单元，在电商行业，你可以直接理解为“商品”）的数量大约在几亿到几十亿这个量级。当然实际上并没有这么多种商品，这里面有很多原因，比如同一个商品它有不同版本型号，再比如，商家为了促销需要，可能会反复上下架同一个商品或者给同一个商品配不同的马甲，这都导致了SKU数量爆炸。

再说这个“重量大”，你可以打开一个电商商详页看一下，从上一直拉到底，你看看有多长？十屏以内的商详页那都叫短的，并且这里面不光有大量的文字，还有大量的图片和视频，甚至还有AR/VR的玩法在里面，所以说，每个商详页都是个“大胖子”。

支持商品系统的存储，要保存这么多的“大胖子”，还要支撑高并发，任务艰巨。

商品系统需要保存哪些数据？

先来看一下，一个商详页都有哪些信息需要保存。我把一个商详页里面的所有信息总结了一下，放在下面这张思维导图里面。



www.ixuexi.cf 分享站 用户标记 group_share

这里面，右边灰色的部分，来自于电商的其他系统，我们暂且不去管这些，左边彩色部分，都是商品系统需要存储的内容。

这么多内容怎么存？能不能像保存订单数据那样，设计一张商品表，把这些数据一股脑儿都放进去？一张表存不下就再加几张子表，这样行不行？你还真别说不行，现在这些电商大厂，在它们发展的早期就是这么干的。现在那么复杂的分布式存储架构，都是一点儿一点儿逐步演进过来的。

这么做的好处，就是糙快猛，简单可靠而且容易实现，但是，撑不了多少数据量，也撑不了多少并发。如果说，你要低成本快速构建一个小规模电商，这么做还真就是一个挺合理的选择。

当然，规模再大一点儿就不能这么干了。不能用数据库，那应该选择哪种存储系统来保存这么复杂的商品数据呢？任何一种存储都是没办法满足的，解决的思路是**分而治之，**我们可以把商品系统需要存储的数据按照特点，分成商品基本信息、商品参数、图片视频和商品介绍几个部分来分别存储。

商品基本信息该如何存储?

我们先来分析商品的基本信息，它包括商品的主副标题、价格、颜色等一些商品最基本、主要的属性。这些属性都是固定的，不太可能会因为需求或者不同的商品而变化，而且，这部分数据也不会太大。所以，还是建议你在数据库中建一张表来保存商品的基本信息。

然后，还需要在数据库前面，加一个缓存，帮助数据抵挡绝大部分的读请求。这个缓存，你可以使用Redis，也可以用Memcached，这两种存储系统都是基于内存的KV存储，都能解决问题。

接下来我和你简单看一下，如何来使用前置缓存来缓存商品数据。

处理商品信息的读请求时，先去缓存查找，如果找到就直接返回缓存中的数据。如果在缓存中没找到，再去查数据库，把从数据库中查到的商品信息返回给页面，顺便把数据在缓存里也放一份。

更新商品信息的时候，在更新数据库的同时，也要把缓存中的数据给删除掉。不然就有可能出现这种情况：数据库中的数据变了，而缓存中的数据没变，商详页上看到的还是旧数据。

这种缓存更新的策略，称为**Cache Aside**，是最简单实用的一种缓存更新策略，适用范围也最广泛。如果你要缓存数据，没有什么特殊的情况，首先就应该考虑使用这个策略。

除了Cache Aside以外，还有Read/Write Through、Write Behind等几种策略，分别适用于不同的情况，后面的课程中我会专门来讲。

设计商品基本信息表的时候，有一点需要提醒你的是，**一定要记得保留商品数据的每一个历史版本**。因为商品数据是随时变化的，但是订单中关联的商品数据，必须是下单那个时刻的商品数据，这一点很重要。你可以为每一个历史版本的商品数据保存一个快照，可以创建一个历史表保存到MySQL中，也可以保存到一些KV存储中。

使用MongoDB保存商品参数

我们再来分析商品参数，参数就是商品的特征。比如说，电脑的内存大小、手机的屏幕尺寸、酒的度数、口红的色号等等。和商品的基本属性一样，都是结构化的数据。但麻烦的是，不同类型的商品，它的参数是完全不一样的。

如果我们设计一个商品参数表，那这个表的字段就会太多了，并且每增加一个品类的商品，这个表就要加字段，这个方案行不通。

既然一个表不能解决问题，那就每个类别分别建一张表。比如说，建一个电脑参数表，里面的字段有CPU型号、内存大小、显卡型号、硬盘大小等等；再建一个酒类参数表，里面的字段有酒精度数、香型、产地等等。如果说，品类比较少，在100个以内，用几十张表分别保存不同品类的商品参数，这样做也是可以的。但是，有没有更好的方法呢？

大多数数据库，都要求数据表要有一个固定的结构。但有一种数据库，没有这个要求。特别适合保存像“商品参数”这种，属性不固定的数据，这个数据库就是MongoDB。

MongoDB是一个面向文档存储的NoSQL数据库，在MongoDB中，表、行、列对应的概念分别是：collection、document、field，其实都是一回事儿，为了便于你理解，在这里我们不咬文嚼字，还是用“表、行、列”来说明。

MongoDB最大的特点就是，它的“表结构”是不需要事先定义的，其实，在MongoDB中根本没有表结构。由于没有表结构，它支持你把任意数据都放在同一张表里，你甚至可以在一张表里保存商品数据、订单数据、物流信息等这些结构完全不同的数据。并且，还能支持按照数据的某个字段进行查询。

它是怎么做到的呢？MongoDB中的每一行数据，在存储层就是简单地被转化成BSON格式后存起来，这个BSON就是一种更紧凑的JSON。所以，即使在同一张表里面，它每一行数据的结构都可以是不一样的。当然，这样的灵活性也是有代价的，MongoDB不支持SQL，多表联查和复杂事务比较孱弱，不太适合存储一般的数据。

但是，对于商品参数信息，数据量大、数据结构不统一，这些MongoDB都可以很好的满足。我们也不需要事务和多表联查，MongoDB简直就是为了保存商品参数量身定制的一样。

使用对象存储保存图片和视频

图片和视频由于占用存储空间比较大，一般的存储方式都是，在数据库中只保存图片视频的ID或者URL，实际的图片视频以文件的方式单独存储。

现在图片和视频存储技术已经非常成熟了，首选的方式就是保存在对象存储（Object Storage）中。各大云厂商都提供对象存储服务，比如国内的七牛云、AWS的S3等等，也有开源的对象存储产品，比如MinIO，可以私有化部署。虽然每个产品的API都不一样，但功能大同小异。

对象存储可以简单理解为一个无限容量的大文件KV存储，它的存储单位是对象，其实就是文件，可以是一张图片，一个视频，也可以是其他任何文件。每个对象都有一个唯一的key，利用这个key就可以随时访问对应的对象。基本的功能就是写入、访问和删除对象。

云服务厂商的对象存储大多都提供了客户端API，可以在Web页面或者App中直接访问而不用通过后端服务来中转。这样，App和页面在上传图片视频的时候，直接保存到对象存储中，然后把对应key保存在商品系统中就可以了。

访问图片视频的时候，真正的图片和视频文件也不需要经过商品系统的后端服务，页面直接通过对象存储提供的URL来访问，又省事儿又节约带宽。而且，几乎所有的对象存储云服务都自带CDN（Content Delivery Network）加速服务，响应时间比直接请求业务的服务器更短。

www.ixuexi.cf 分享站 用户标记 group_share

国内的很多云厂商的对象存储对图片和视频，都做了非常多的针对性优化。最有用的是，缩放图片和视频转码，你只要把图片和视频丢到对象存储中，就可以随时获得任意尺寸大小的图片，视频也会自动转码成各种格式和码率的版本，适配各种App和场景。我只能说，谁用谁知道，真香！

将商品介绍静态化

商品介绍在商详页中占得比重是最大的，包含了大量的带格式文字、图片和视频。其中图片和视频自然要存放在对象存储里面，商品介绍的文本，一般都是随着商详页一起静态化，保存在HTML文件中。

什么是静态化呢？静态化是相对于动态页面来说的。一般我们部署到Tomcat中的Web系统，返回的都是动态页面，也就是在Web请求时，动态生成的。比如说商详页，一个Web请求过来，带着SKUID，Tomcat中的商详页模块，再去访问各种数据库、调用后端服务，动态把这个商详页拼出来，返回给浏览器。

不过，现在基本上没有系统会这么干了，你想，对于每个SKU的商详页，你每次动态生成的页面内容不是完全一样的么？生成这么多次，不仅浪费服务器资源，速度还慢，关键问题是，Tomcat能能抗的并发量和Nginx完全不是一个数量级的。

商详页的绝大部分内容都是商品介绍，它是不怎么变的。那不如就把这个页面事先生成好，保存成一个静态的HTML，访问商详页的时候，直接返回这个HTML。这就是静态化。

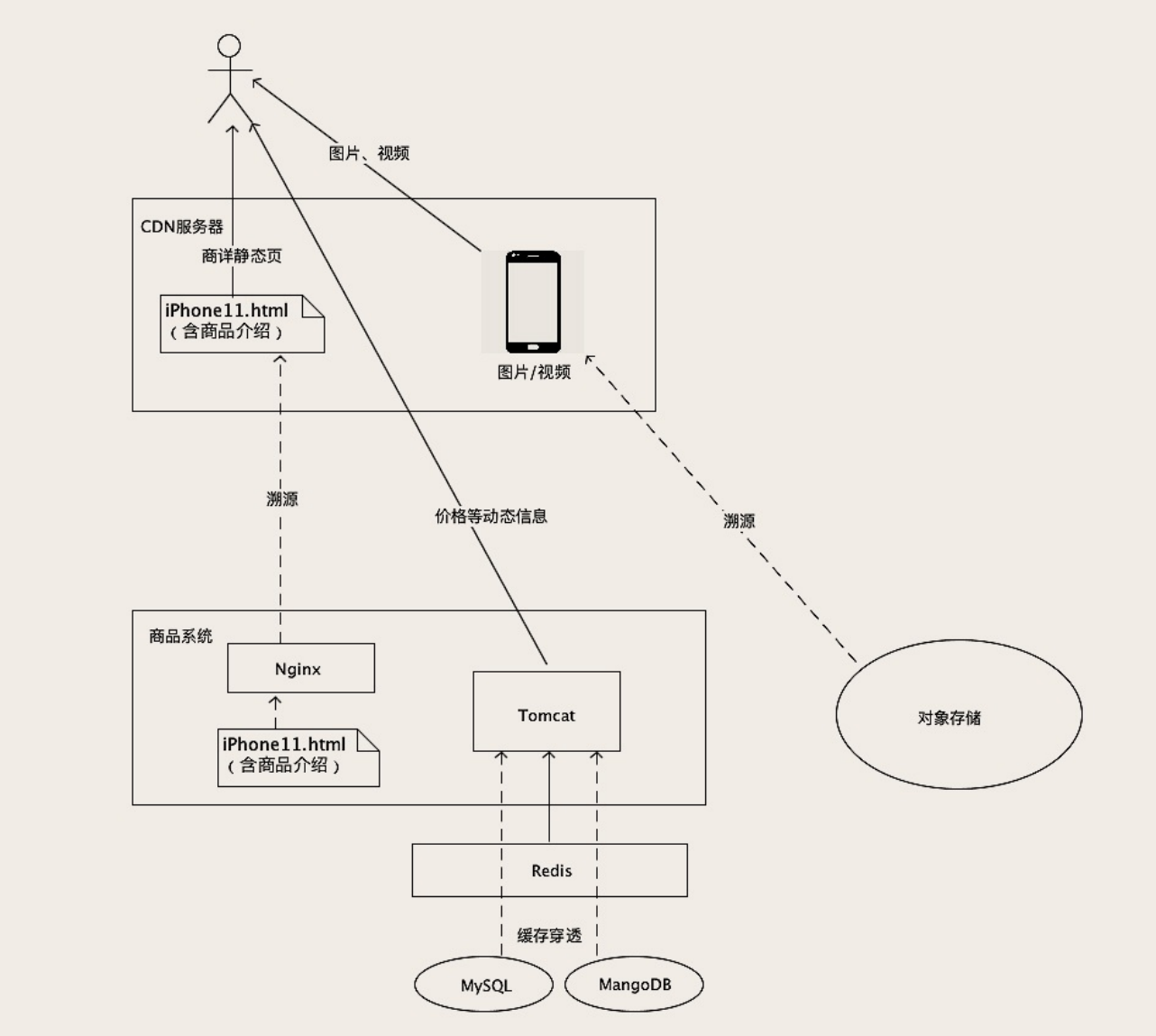
商详页静态化之后，不仅仅是可以节省服务器资源，还可以利用CDN加速，把商详页放到离用户最近的CDN服务器上，让商详页访问更快。

至于商品价格、促销信息等这些需要频繁变动的信息，不能静态化到页面中，可以在前端页面使用AJAX请求商品系统动态获取。这样就兼顾了静态化带来的优势，也能解决商品价格等信息需要实时更新的问题。

小结

最后，我们再来对今天的内容复个盘。商品系统的存储需要提供商品的基本信息、商品参数、图片和视频以及商品介绍等等这些数据。商品的基本信息和商品参数分别保存在MySQL和MongoDB中，用Redis作为前置缓存，图片和视频存放在对象存储中，商品介绍随着商详页一起静态化到商详静态页中。

我把商品系统的存储绘制成下面这张图：



一起来看一下图，这样一个商品系统的存储最终的效果是什么样的？图中实线表示每访问一次商详页，需要真正传输的数据，虚线表示当商详页数据发生变化的时候才需要进行一次数据传输。用户打开一个SKU的商详页时，首先去CDN获取商详页的HTML，然后访问商品系统获取价格等频繁变化的信息，这些信息从Redis缓存中获取。图片和视频信息，也是从对象存储的CDN中获取。

分析一下效果，数据量最大的图片、视频和商品介绍都是从离用户最近的CDN服务商获取的，速度快，节约带宽。真正打到商品系统的请求，就是价格这些需要动态获取的商品信息，一般做一次Redis查询就可以了，基本不会有流量打到MySQL中。

这样一个商品系统的存储的架构，把大部分请求都转移到了又便宜速度又快的CDN服务器上，可以用很少量的服务器和带宽资源，抗住大量的并发请求。

思考题

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？

这样给用户的体验非常不好。你不要以为这是一个小概率事件，当你的系统用户足够多的时候，每时每刻都有人在下单，这几乎是个必然出现的事件。

课后请你想一下，该怎么来解决这个问题？欢迎你在留言区与我交流互动。

www.ixuexi.cf 分享站 用户标记 group_share

感谢你的阅读，如果你觉得今天的内容对你有所帮助，也欢迎把它分享给你的朋友。

精选留言：

- 李玥 2020-02-26 18:02:50
  hi，我是李玥。跟上节课一样，我还是在留言板上同步一下上节课的思考题，大家一起来学习探讨。

  上节课我们讲了两种实现幂等的方法，课后呢，我也让你思考了下，在你负责开发的业务系统中，能不能用这节课中讲到的方法来实现幂等？除了这两种方法以外，还有哪些实现服务幂等的方法？

  关于这个问题，我是这么看的。

  其实总结下来这些实现幂等的方法，无非是两大类，一类是通过一些精巧的设计让更新本身就是幂等的，这种需要点儿运气，不是所有业务都适用的。另外，就是利用外部的、具备一致性的存储（比如说MySQL）来做冲突检测，你在设计幂等方法的时候一般都可以顺着这两个思路来开展。 [3赞]

- 谭伟 2020-02-26 21:09:03
  前提是需要明确希望以用户当时看到的价格为准，还是以最新的价格为准。
  1）以用户看到的价格为准.MVCC方式就行
  2）最新的话，也是带上版本信息，不一致则提升。

- Din 2020-02-26 18:27:39
  下单前先调用校验价格的接口，如果价格已经发生了变化，提示用户刷新页面。

- 公号-云原生程序员 2020-02-26 18:14:02
  完全感受到了老师的理论与实践结合的功力。

- 我叫徐小晋 2020-02-26 17:41:02
  老师，您好。
  思考题:进入下单页面，带上详情页面这个价格数据的版本。这样发起下单请求，把这个版本一起发送给后端