04 | 缓存设计：做好缓存设计的关键是什么？

2021-05-25 尉刚强

《性能优化高手课》 课程介绍 >

讲述：尉刚强

时长 14:43 大小 13.48M

你好，我是尉刚强，今天我们来聊聊基于性能的缓存设计。

缓存就是一个临时储存数据的地方。当用户查询数据时，首先会在缓存中查找，如果找到了就直接使用；如果找不到，就再到数据的原始位置去寻找。所以，缓存本质上是一种用空间换时间的技术，通过数据在空间上的重复，来提升数据的访问速度。

不过，随着分布式和云计算技术的发展，数据存储技术也发生了翻天覆地的变化，而且不同存储技术在价格和性能上都存在很大的差异，所以在针对性能进行软件设计时，如果我们没有做好多层级的缓存设计，不仅可能浪费钱，而且获取的性能收益可能也不够理想。

所以在今天的课程上，我会结合互联网应用与服务场景，给你讲解如何做好缓存设计，并会剖析典型的缓存使用案例，帮助你建立起缓存技术原理的系统认识，以此指导你在产品

的软件设计中，可以正确使用缓存来提升系统的性能。

缓存设计的通关之路

那么首先，我想从两个问题开始，来带你了解下缓存设计要在什么时候做，以及通过不同数据类型的特性对比分析，来跟你一起探讨如何才能做好缓存设计。

好，第一个问题：**在互联网应用服务中，使用缓存技术的目的就只是为了提升访问速度吗？**

其实我认为，并不是所有的缓存都只是为了提升速度，因为**在分布式系统中，缓存机制实际上是系统级性能设计的一个重要权衡手段。**比如当某个数据库的负载比较高，接近系统瓶颈时，我们就可以使用缓存技术，把负荷分担到其他数据库中，那么这里使用缓存的目的，主要就是负载均衡，而不是提升访问速度。

第二个问题：**一个大型系统中的数据种类会非常多，那么需要为每种数据都设计缓存机制吗？**

其实完全没有必要。在实际的业务场景下，系统包含的业务数据太多了，你不可能针对每种数据都设计和实现缓存机制，因为一方面是投入的软件成本太高，另一方面也很可能无法带来比较高的性能收益。所以，在进行缓存设计之前，**你首先需要识别出哪些数据访问对性能的影响比较大。**

那么我们该怎么去识别哪些数据是需要缓存机制呢？在第 1 讲中，我已经给你介绍了性能建模设计方法，也就是通过分析和评估手段，来识别出哪个数据访问操作对性能的影响比较关键，然后再进行缓存设计。

不过接下来，在识别出需要使用缓存机制的数据之后，你可能会发现，这些数据种类之间的特性差异非常大，如果使用同一种缓存设计的话，其实是很难发挥出软件性能的最佳状态的。

所以在这里，我给你总结了三种需要缓存机制的数据种类，分别是不变性数据、弱一致性数据、强一致性数据。了解这三种缓存数据种类的差异，以及对应的设计缓存机制的方法，你就掌握了缓存设计的精髓。

好，下面我们就来具体了解下吧。

不变性数据

首先是不变性数据，它代表数据永远不发生变化，或者是在较长一个时间段内不会发生变化，因此我们也可以认为这部分数据是不变的。

这类数据就是可以**优先考虑使用缓存技术**的一种数据类型，在实际的业务场景中也非常多。比如，Web 服务中的静态网页、静态资源，或者数据库表中列数据与 key 的映射关系、业务的启动配置，等等，这些都可以认为是不变性数据。

而且，**不变性数据也意味着实现分布式一致性会非常容易**，我们可以为这些数据选择任意的数据存储方式，也可以选择任意的存储节点位置。因此，我们实现缓存机制的方式就可以很灵活，也会比较简单，比如说在 Java 语言中，你可以直接使用内存 Caffeine，或者内置的结构体来作为缓存都可以。

另外这里你要注意，当你针对不变性数据进行缓存设计时，其中的缓存失效机制可以采用永久不失效，或者基于时间的失效方式。而在采用基于时间的失效方式的时候，你还需要根据具体的业务需求，在缓存容量和访问速度之间做好设计实现上的权衡。

弱一致性数据

第二种是弱一致性数据，它代表数据会经常发生变化，但是业务对数据的一致性要求不高，也就是说，不同用户在同一时间点上看到不完全一致的数据，都是可以接受的。

由于这类数据**对一致性的要求比较低**，所以在设计缓存机制时，你只需要实现最终一致性就可以了。这类数据在实际业务中也比较多，比如业务的历史分析数据、一些搜索查找返回数据等，即使最近的一些数据没有记录进去，关系也不大。

另外，快速识别这类数据还有一个方法，那就是使用数据库 Replica（复制）节点中读取的数据，大部分都是这种类型的数据（很多数据库 Replica 节点的数据因为数据同步时延，是不满足强一致性要求的）。

针对弱一致性的数据，我们通常使用的缓存失效机制是基于时间的失效方式，同时因为弱一致性的特性，你可以比较灵活地选择数据存储技术，比如内存 Cache，或者是分布式数

据库 Cache。你甚至可以基于负载均衡的调度，来设计多层级缓存机制。

强一致性数据

第三种缓存数据类型是强一致性数据，它是指代码数据会经常发生变化，而且业务对数据库的一致性要求非常高，也就是说当数据发生变更后，其他用户在系统中的任何地方，都应该看到的是更新后的数据。

那么，针对这种类型的数据，我一般是**不推荐你去使用缓存机制**，因为这类数据在使用缓存时会比较复杂，而且很容易会引入新的问题。比如说，用户可以直接提交和修改的各种数据内容，如果没有同步修改缓存中的数据，就会引发数据不一致性的问题，导致比较严重的业务故障。

不过在一些特殊的业务场景中，比如，在针对个别的数据访问频率非常高的情况下，我们还是需要通过设计缓存机制，来进一步提升性能。因此针对这类强一致性数据，在设计缓存机制时，你需要特别注意两点：

1. 这种数据的缓存一定要**采用修改同步的实现方式**。也就是说，所有的数据修改都必须确保可以同步修改缓存与数据库中的数据。
2. 准确识别特定业务流程中，可以**使用缓存获取数据的时间**有多长。因为有些缓存数据（比如一次 REST 请求中，多个流程都需要使用的数据）只可以在单次业务流程中使用，不能跨业务流程使用。

好了，以上就是三种典型的数据种类的缓存设计思路了。这里你需要注意的是，使用缓存一定是以性能优化为目的，因此，你还需要使用**评估模型**来分析缓存是否达到了性能优化的目标。

那么具体是什么评估模型呢？我们来看一下这个性能评估模型的公式：**AMAT = Thit + MR * MP**。其中：

AMAT（Average Memory Access Time），代表的是平均内存访问时间；

Thit，是指命中缓存之后的数据访问时间；

MR，是指访问缓存的失效率；

MP，是指缓存失效后，系统访问缓存的时间与访问原始数据请求的时间之和。

另外这里你可能会注意到，AMAT 与原始数据访问之间的差值，代表的就是使用缓存所带来的访问速度的提升。而在一些缓存使用不当的场景下，增加的缓存机制很可能会造成数据访问速度下降的情况。所以接下来，我就通过真实的缓存设计案例，来带你理解如何正确地使用缓存，以此帮助你更好地提升系统性能。

缓存设计的典型使用场景

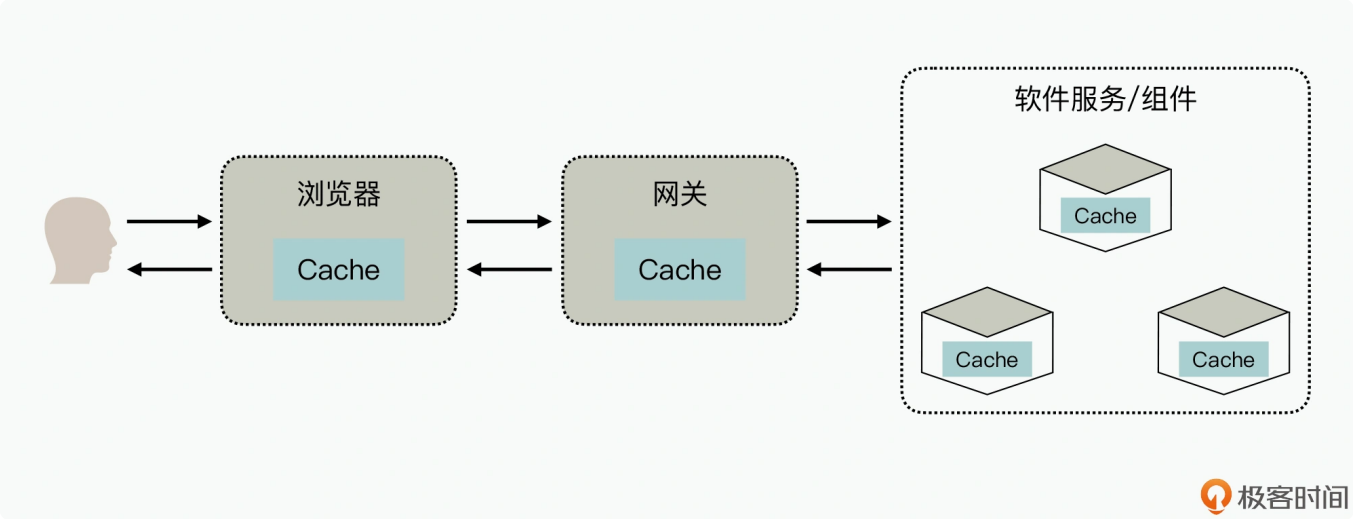
好，在开始介绍之前呢，我还想给你说明一下，在真实的业务中，缓存设计的场景其实有很多，这里我的目的主要是让你明确缓存设计的方法。因此，我会从两个比较典型的案例场景入手，来带你理解缓存的使用。

如何做好静态页面的缓存设计？

在 Web 应用服务中，一个重要的应用场景就是静态页面的缓存使用。这里的静态页面是指一个网站内，所有用户看到的都是一样的页面，除非重新部署否则一般不发生变更，比如大部分公司官网的首页封面等。

通常静态页面的访问并发量是比较大的，如果你不使用缓存技术，**不仅会造成用户响应时延比较长，而且会对后端服务造成很大的负载压力**。

那么针对静态页面，我们在使用缓存技术时，可以通过将静态缓存放到距离用户近的地方，来减少页面数据在网络上的传输时延。现在，我们来看一个针对静态页面使用缓存设计的示意图：



如图上所示，针对静态网页，首先你就可以在软件后端服务的实例中使用缓存技术，从而避免每次都要重新生成页面信息。然后，由于静态网页属于不变性数据，所以你可以使用

内存或文件级缓存。另外，针对访问量非常大的静态页面，为了进一步减少对后端服务的压力，你还可以将静态页面放在网关处，然后利用 OpenResty 等第三方框架增加缓存机制，来保存静态页面。

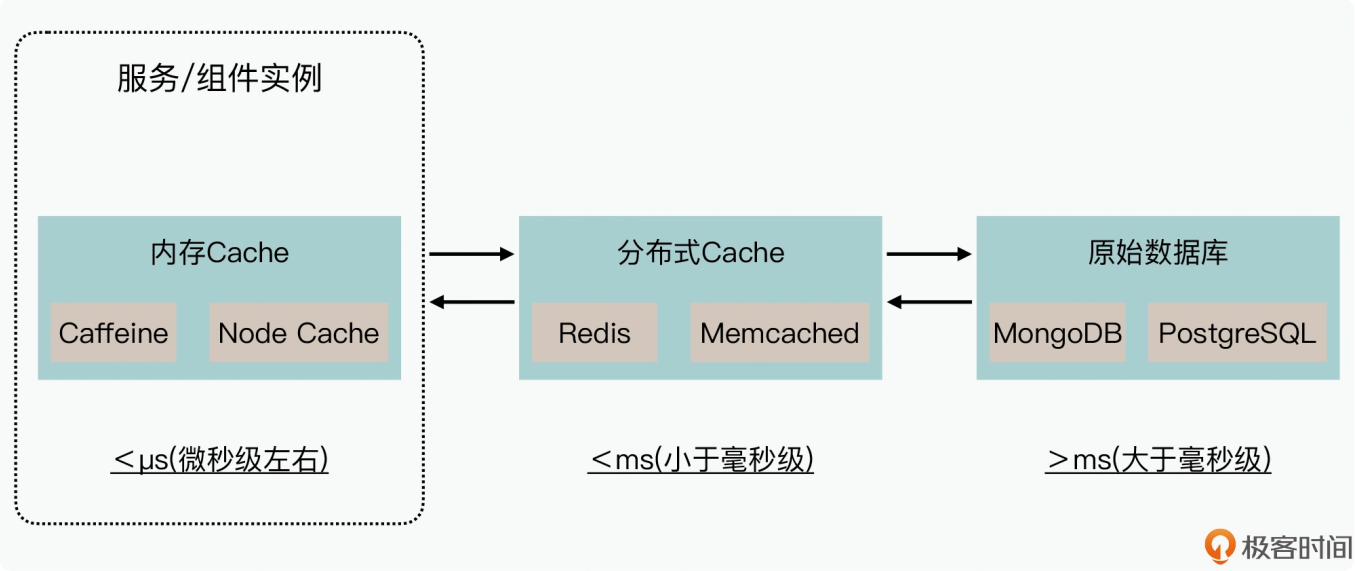
除此之外，在网页中很多的静态页面或静态资源文件，还需要使用浏览器的缓存，来进一步提升性能。

注意，这里我并不是建议你针对所有的静态页面，你都需要设计三层的缓存机制，而是你要知道，在软件设计阶段，一般就需要考虑如何做静态页面的缓存设计了。

后端服务如何设计数据库的多级缓存机制？

还有一个典型的缓存场景是针对数据库的缓存。现在的数据库通常都是分布式存储的，而且规模都比较大，在针对大规模数据进行查询与分析计算时，都需要花费一定的时间周期。

因此，我们可以先识别出这些计算结果中可以使用缓存机制的数据，然后就可以使用缓存来提升访问速度了。下面是一张针对数据库缓存机制的原理图：



从图上我们可以看到，内存级 Cache、分布式 Cache 都可以作为数据计算分析结果的缓存。而且，不同级的缓存访问速度是不一样的，内存级的 Cache 访问速度可以到微秒级别，甚至更好；分布式 Cache 访问速度通常可以小于毫秒级别；而针对原生数据库的查询与分析，通常是大于毫秒级别的。

因此，在具体设计缓存机制的时候，你就需要依据前面我介绍的缓存使用原理，识别出数据类型，然后选择并设计缓存实现机制。

另外，在使用缓存机制实现访问速度优化的过程中，我们的**主要关注点是不同层级缓存所带来的访问速度提升**，而在这里，不同层级缓存也是可以在一个数据库中的。比如，在我参与设计的一个性能优化项目中，其 Cache 策略就是，使用 MongoDB 中的另外一个 Collection（集合），来作为缓存查询分析，以此优化性能。

所以，你在做缓存设计时，关注点应该放到不同的数据种类，以及不同层级缓存的性能评估模型上，而不是只关注数据库。只有这样，你才能设计出更好、更优的性能缓存方案。

小结

今天，我重点介绍了缓存技术的使用原理和典型应用场景，当你在进行软件业务系统性能设计时，可以结合今天学习的内容，识别出系统中各种可缓存的数据类型，然后有针对性地设计缓存方案，并且还可以根据评估模型，来进行前期的性能验证分析。

另外，在具体的缓存技术实现中，比如缓存替换算法、缓存失效策略等，通常这部分能力是内置于缓存库的配置选项当中的，或者选用第三方库即可，需要自定义设计算法的实现场景很少。所以这里你需要重点做的，就是选择合适的配置和策略即可。

思考题

对于一致性要求比较高的数据信息，在微服务多实例架构中，我们是否可以选择使用内存 Cache 呢？

欢迎在留言区分享你的答案和思考。如果觉得有收获，也欢迎你把今天的内容分享给更多的朋友。

上一篇 03 | 并行设计（下）：如何高效解决同步互斥问题？

下一篇 05 | IO设计：如何设计IO交互来提升系统性能？

精选留言 (4)

写留言

Jem
2021-06-07

可以使用,比如作者所说的静态资源但是这就会引出例外一个问题:内存资源消耗 ,因为需要保证每个微服务中的内存cache必需都有,有点得不偿失.

展开

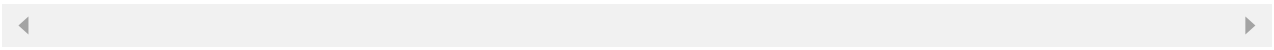
作者回复: 所以需要在内存和速度之间权衡，很多时候，当客户对响应速度更关注的时候，就可以使用这种权衡手段。

落叶之葉
2021-06-01

我们做一个商品价格展示，影响价格的维度有很多，地区，人员类型，批次，等等，目前数据写入到solr库来，最大的问题就是数据库数据提取到solr中

展开

作者回复: solr本来就擅长多维度存储数据，你的调整是识别哪些维度需要保存吧？




Ming☀

2021-05-27

内存cache只在当前进程下可使用，分布式多实例无法共享

展开

作者回复: 是的！不过有些数据是不需要共享的


raisecomer

2021-05-26

"在分布式系统中，缓存机制实际上是系统级性能设计的一个重要权衡手段.....可以使用缓存技术，把负荷分担到其他数据库中"，缓存技术是如何把负荷分担到其它数据库的呢？

作者回复: 缓存可以讲部分请求负载(吞吐量)从原始数据库中卸载下来，转移到缓存中。


1