图文 129 案例实战：一线电商公司的订单系统是如何进行数据库设计的?

手机观看

210 人次阅读 2020-11-13 07:00:00

详情 目录 评论

案例实战：一线电商公司的订单系统是如何进行数据库设计的?

欢迎大家加入我们的**儒猿技术交流群**，一个纯粹的**交流技术、分享面经**的地方。

群里有一线大厂助教答疑、专栏优秀作业交流、互联网大厂面经分享、知名互联网公司内推，**点击下方链接了解**：

- 儒猿技术交流群学员面经分享
- 儒猿技术窝优秀学员作业展示
- 儒猿技术交流群日常技术交流及助教答疑

如何加群？点击了解

儒猿学院官网上线，内有石杉老师架构课最新大纲，儒猿云平台详细介绍，敬请浏览

官网：www.ruyuan2020.com（**建议PC端访问**）

今天我们来给大家讲讲第二个案例拓展，也就是一般互联网公司的订单系统是如何做分库分表的，既然要聊订单系统的分库分表，那么就得先说说为什么订单需要分库分表，其实最关键的一点就是要分析一下订单系统的数据量，那么订单系统的数据量有多大？这个就得看具体公司的情况了。

儒猿技术窝

进店逛逛

相关频道

从零开始带你成为MySQL实战优化高手

已更新139期

比如说一个小型互联网公司，如果是涉及到电商交易的，那么肯定每天都会有一些订单进来的，那么比如小型互联网公司假设有500万的注册用户，每天日活的用户会有多少人？意思就是说，你500万的注册用户，并不是每个人每天都来光顾你这里的！

我们往多了说，即使按照28法则，你500万的注册用户，每天最多是20%的用户会过来光顾你这里，也就是会来访问你的APP/小程序/网站，也就是100万的日活用户，但是这个日活比例恐怕很多公司都达不到，所以一般靠谱点就算他是10%的用户每天会来光顾你，算下来就是平均每个注册用户10天会来光顾你一次，这就是50万的日活用户。

但是这50万的日活用户仅仅是来看看而已，那么有多少人会来买你的东西呢？这个购买比例可就更低了，基本上很可能这种小型互联网公司每天就做个1w订单，或者几万订单，这就已经相当的不错了，咱们就以保守点按1w订单来算吧。

那么也就是说，这个互联网公司的订单表每天新增数据大概是1w左右，每个月是新增30w数据，每年是新增360w数据。大家对这个数据量感觉如何？看着不大是吧，但是按照我们上次说的，一般建议单表控制在千万以内，尽量是100w到500w之间，如果控制在几十万是最好了！

所以说，分析下来，大家会发现，哪怕是个小互联网公司，居然订单数据量也不少！因为订单这种数据和用户数据是不同的，你用户数据一般不会增长过快，而且很快会达到一个天花板，就不会怎么再涨了，但是订单数据是每天都有增量的，他们的特点是不同的。

所以说这个订单表，即使你按一年360w数据增长来计算，最多3年就到千万级大表了，这个就绝对会导致你涉及订单的操作，速度挺慢的。我这里可以给大家分享两个我亲身体验过的订单这块的案例。

一个是我使用过的某社保类的APP，这个APP可以让你在上面下单自助缴纳五险一金，你每次自助缴纳，说白了就是下一个订单，把钱给他，他帮你缴纳五险一金，这个东西对于很多自由职业者是很有用的。

这个APP，很明显就是订单日积月累很多，而且一定是没有做任何的分表，导致每次对自己的订单进行查询的时候，基本都是秒级，每次打开订单页面都很慢，有时候甚至会达到两三秒的样子，这个体验就很差。

另外一个是我使用过的一个企业银行的APP，大家都知道，企业银行是可以允许财务提交打款申请，然后有人可以去审批的，但是有一个银行APP，很明显也是对这类申请和审批的数据表，没有做分库分表的处理，导致数据日积月累的增加，每次在申请和审批的查询界面都很慢，起码要卡1s以上的时间，这个体验也很不好。

所以说，基本上个这类订单表，哪怕是个小互联网公司，按分库分表几乎是必须得做的，那么怎么做呢？订单表，一般在拆分的时候，往往要考虑到三个维度，一个是必然要按照订单id为粒度去分库分表，也就是把订单id进行hash后，对表数量进行取模然后把订单数据均匀分散到100~1000个表里去，再把这些表分散在多台服务器上。

但是这里有个问题，另外两个维度是用户端和运营端，用户端，就是用户可能要查自己的订单，运营端就是公司可能要查所有

订单，那么怎么解决这类问题呢？其实就跟上次的差不多，基本上针对用户端，你就需要按照（userid, orderid）这个表结构，去做一个索引映射表。

userid和orderid的一一对应映射关系要放在这个表里，然后针对userid为粒度去进行分库分表，也就是对userid进行hash后取模，然后把数据均匀分散在很多索引映射表里，再把表放在很多数据库里。

然后每次用户端拿出APP查询自己的订单，直接根据userid去hash然后取模路由到一个索引映射表，找到这个用户的orderid，这里当然可以做一个分页了，因为一般订单都是支持分页的，此时可以允许用于户分页查询orderid，然后拿到一堆orderid了，再根据orderid去按照orderid粒度分库分表的表里提取订单完整数据。

至于运营端，一般都是要根据N多条件对订单进行搜索的，此时跟上次讲的一样，可以把订单数据的搜索条件都同步到ES里，然后用ES来进行复杂搜索，找出来一波orderid，再根据orderid去分库分表里找订单完整数据。

其实大家到最后会发现，分库分表的玩法基本都是这套思路，按业务id分库分表，建立索引映射表同时进行分库分表，数据同步到ES做复杂搜索，基本这套玩法就可以保证你的分库分表场景下，各种业务功能都可以支撑了。

End

儒猿技术窝精品专栏及课程推荐：

- 《从零开始带你成为消息中间件实战高手》
- 《互联网Java工程师面试突击》（第2季）
- 《互联网Java工程师面试突击》（第1季）
- 《互联网Java工程师面试突击》（第3季）
- 《从零开始带你成为JVM实战高手》
- 《C2C电商系统微服务架构120天实战训练营》