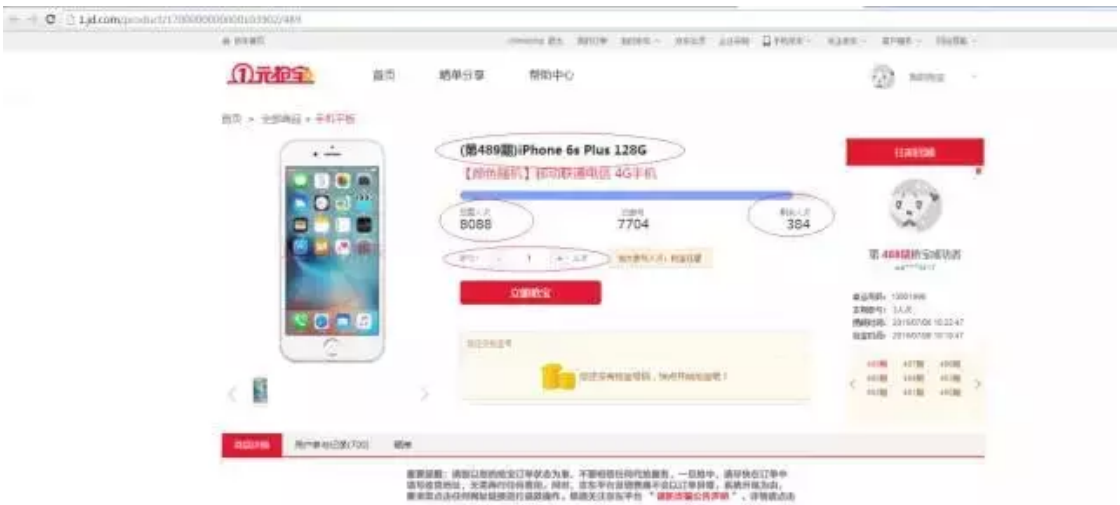
京东618实践：一元抢宝系统的数据库架构优化

DBAplus社群 2016-07-18

一元抢宝系统是京东虚拟新兴的一个业务系统，上线以来订单量一直持续增长。在距离618前两个月时，京东商城商品虚拟研发部对系统做了整体预估，订单量快速增长及618大促的到来都将带来单量剧增，届时势必会对数据库容量和负载造成压力。

分析结果表明数据库很可能成为影响性能的瓶颈，并决定对数据库底层做分库分表改造，确保数据水平动态扩展能力，满足数据容量持续增长的需求，并提高下单效率。

一、业务介绍



上图是一元抢宝商品详情页，从图中可以看出，一元抢宝的商品即商品项，其不同于其他京东商品的地方在于：有期次、总人次和剩余人次的概念；假设一个商品项有100个库存，则会分100期次售卖，每期次一个售卖的是一个库存；总人次即设置的每一期抢宝商品价格，假设1000人次，则商品总价是1000元（每人1元）；当剩余人次为0时，本期抢宝结束，然后按照相应算法产生抢宝者；然后进行下一期抢宝。

通过技术改造，从整体上来说实现三个目标：

1. 底层路由策略实现；
2. 历史数据迁移；

3. 业务改造。下面详细介绍本次改造的过程。

二、数据库容器预估

分库分表最重要的是要先做容器预估，依据数据量和业务特性估算出容器/库/表的数量及分库分表规则。

假设一天100万订单，一年则产生3.6亿订单量；假设数据结构是这样的：订单表10个字段，一个字段50个字符；一条订单则需要500字节存储，那么3.6亿订单则需要大约170GB存储空间；假设每台机器存储空间为200GB，则每年增加一台机器即可满足容量需求。而实际需求要根据压测结果来决定；如压测其他一些指标是否满足需求，如QPS、响应时间等。

三、底层路由策略选择及实现

分库分表路由策略是基础，影响整个系统架构，后期业务需求是否满足和支持，使用是否方便都与此有关。路由策略设计合理，上层业务使用会很方便。一元抢宝项目的路由策略适配和实现是在DAO层实现，对上层业务层透明，可不用关心具体实现，并且路由策略不涉及结构上的改动，对上层不会产生影响。

我们知道常见的分表策略有两种：

- **hash路由**

优点：可实现数据分散，热点分散；
不足：增加数据库节点时，会影响路由策略，需做数据迁移；

- **分区路由（增量区间路由）**

优点：策略支持动态扩容，理论上可无限扩展；
不足：存在数据热点问题，新产生的表，读写频率较高；每次查询需要经过路由策略表。

当然每种策略都不是完美的，只有最适合业务场景的策略才是好的。该项目采用的是两种方式的结合。

首先按抢宝项hash分库，然后按抢宝期区间段分表，如下图所示：

期的路由策略表规则如下：

为什么使用这种策略?

抢宝项是业务上层维度，可以理解为商品，大部分表中都有这个字段；此id生成时是连续的，长期来看，hash分库后数据是均衡的。抢宝期是抢宝项下的一个维度，如一个项库存是100，不停售前提下，会生成100期，在售的期次只有一个。

为什么选择期id区间作为分表路由策略呢，有朋友会认为也可以选择订单id，从路由策略上来说，没有问题，但一元抢宝项目的业务场景，有根据项id和期id查询订单参与纪录的场景，所以要考虑通过这两个维度能查到订单。另外，使用区间作为分表策略，可以动态扩展，即使每次查询经过路由表，这点开销可以忽略，而且都是通过缓存加载。

那以上策略，可以路由的维度有哪些呢?

1. 通过订单id路由:订单号按照一定规则生成，其存储了库和表的信息，可以根据订单号直接定位到相应的库和表；
2. 通过抢宝项id和抢宝期id路由：抢宝项hash定位到库，抢宝期查询路由策略表定位到表，具体图示如下：

四、聚合查询及聚合数据同步的实现

有分就涉及到聚合查询，我们如何实现呢？先看如下架构图：

上图是数据层改造后的架构图，之前是单表主从模式，改造后为多个分库、基础库。聚合采用了elastic search （以下简称ES）。

为什么使用它呢，首先，简单便捷，容易接入；其次，支持动态扩容分片，对业务层透明等。系统中的聚合查询主要使用了ES，当然我们有很多降级方案，后面会讲到。ES不能当作库来使用，它并不能百分之百保证数据完整性，所以一定要有数据备份，我们使用了

聚合表，保存一段时间内的数据，用于降级使用，一旦ES有延迟或集群不可用，就会降级查询聚合表。

同步ES我们是怎么做的呢？我们使用了canal。有的朋友可能说了，为什么不在直接在代码中插入时去同步，可以这样做，但有两个问题，一是同步失败如何处理，如何保证事务，二是与业务代码强耦合，借用术语，不beautify。使用canal，代码解耦，不侵入与代码。

它其实是模拟了数据库主从复制机制，伪装为一个从库，当数据库（为不影响主库生产，我们监听的是从库）binlog有变化时，canal监听到，通过解析服务解析过滤binlog，把需要的日志过滤出来。解析后，我们通过发送MQ消息，消息体是表名和主键id，不是整条数据，消费端接到变化的表名和id，实时从库中查询最新数据，同步到ES、聚合表。

为什么通过MQ消息呢？还可以用以上两点来解释，一是消息支持失败重试，存储失败后抛异常，等待下次处理，二是系统间解耦。细心的朋友可以看到，一个消息队列，通过多个消费订阅（可以理解为每个消费者的队列都是镜像复制的）。这样做为了在存储时不相互影响；如果使用一个订阅者处理，存储ES失败，其他两个聚合存储成功，那也要抛异常或其他处理方式，下次消费时，另两个聚合还要存储一次。

以上就是我们聚合和同步聚合的设计。查询时，一部分业务会先查询缓存，不存在再查询ES，如果降级，才会查库，正常的聚合查询都不会查到库。

五、历史数据迁移

由于我们系统上线时是单库，分库是上线几个月后做的技改，所以数据需要迁移，主要迁移步骤如下：

前半部分，从扫描到同步到分库是新代码，后面canal到同步ES、聚合表都是复用上面逻辑，这样设计，降低我们整体工作量，并且保证数据迁移完整。
具体迁移细节如下：

可以看出，主要分为两部分，停机前和停机后。停机前是迁移历史数据，支持重复迁移；停机后，只迁移增量部分，这样，大大缩短我们的上线时间。停机后只需要迁移很少的数据量。

迁移就涉及到数据校验，校验逻辑整体来说比较简单：

三个维度分别和基础库做对比，如果不同，重新迁移某一天数据。

六、系统关键节点降级

这一部分也很重要，我们的降级主要有两点，一是canal同步延迟降级，一是ES不可用降级。第一种如下：

如果canal同步延迟，或者从库挂掉，开启开关，扫描主库数据（最近几小时）直接同步到ES、聚合表；这样，即使从库挂掉，也不影响业务数据，这一点很重要，实际业务场景中我们也遇到过。

ES降级，ES不可用时，关闭ES开关，直接查询聚合表。

七、总结

一个系统从设计到最终完成，依赖于整个团队，每个人的想法、不同思路的碰撞和付出；再有前期合理细致的设计尤为重要，每个时间点和具体上线步骤和回滚方案做好详细计划；另外，就是细致深入测试，测试环境和线上多轮测试和回归，也是正常上线的重要保证。

以上就是京东一元抢宝项目分库分表的主要思想，希望有同样想法的朋友可以深入交流，互相提升系统架构。

作者介绍 匙凯明

- 京东高级开发工程师，在京东负责一元抢宝系统架构和开发工作；多年互联网经验，对于系统架构和设计有自己的见解和经验。

> 经平台同意授权转载
>
> 来源：开涛的博客 订阅号（id: kaitao-1234567）

精选专题（点击蓝色标题可阅读全文）

- **技术分享：**[线上1-50期] [北京站] [上海站] [广州站] [杭州站] [济南站] [Gdevops杭州站] [Gdevops北京站] [DAMS 2016]
- **专家专栏：**[杨志洪] [杨建荣] [陈能技] [丁俊] [卢钧轶] [李海翔] [魏兴华] [邹德裕] [周正中] [高强] [白鳝] [卢飞] [王佩]
- **热门话题：**[Oracle] [MySQL] [DB2] [大数据] [PostgreSQL] [云计算] [DevOps] [职场心路] [其他]