Atitit 数据库分片 分库后问题总结

目录

[1、分布式事务问题 1](#_Toc4799)

[2、跨库join的问题（业务表做分区解决） 1](#_Toc8815)

[3、横向扩容的问题 使用userid解决 1](#_Toc30388)

[4、结果集合并、排序的问题 分区+es 1](#_Toc7701)

### **将此业务表切换到分布式数据库产品中**

此方案最简单直接。但我们的业务中，只有此表数据量较大且需要查询详细单据。仅为了一张表就引入一种存储机制，考虑到运维和经济成本，总觉得不划算。另外，此表还与其他表有一定的联合查询操作，分离出去后会增加应用层的复杂度；

三、分库分表后引入的问题

## 1、分布式事务问题

如果我们做了垂直分库或者水平分库以后，就必然会涉及到跨库执行SQL的问题，这样就引发了互联网界的老大难问题-"分布式事务"。那要如何解决这个问题呢？

1.使用分布式事务中间件 2.使用MySQL自带的针对跨库的事务一致性方案(XA)，不过性能要比单库的慢10倍左右。3.能否避免掉跨库操作(比如将用户和商品放在同一个库中)

## 2、跨库join的问题（业务表做分区解决）

分库分表后表之间的关联操作将受到限制，我们无法join位于不同分库的表，也无法join分表粒度不同的表， 结果原本一次查询能够完成的业务，可能需要多次查询才能完成。粗略的解决方法： 全局表：基础数据，所有库都拷贝一份。 字段冗余：这样有些字段就不用join去查询了。 系统层组装：分别查询出所有，然后组装起来，较复杂。

用分区解决。。。

## 3、横向扩容的问题 使用userid解决

当我们使用HASH取模做分表的时候，针对数据量的递增，可能需要动态的增加表，此时就需要考虑因为reHash导致数据迁移的问题。

## 4、结果集合并、排序的问题 分区+es

因为我们是将数据分散存储到不同的库、表里的，当我们查询指定数据列表时，数据来源于不同的子库或者子表，就必然会引发结果集合并、排序的问题。如果每次查询都需要排序、合并等操作，性能肯定会受非常大的影响。走缓存可能一条路!

少量分区解决，大量使用使用mongodb es解决