聊天系统的数据优化方案

考虑到聊天系统产生大量的聊天数据，存到数据库，如果用户量多的话大量的查询会存在io问题，造成系统的性能下降。针对这个问题，我提出了以下优化方案。

数据库方面优化

1. 采用主从数据库实现读写分离。（读库可以考虑MyIsam引擎）

由于消息被频繁查询，MyISAM管理非事务表。它提供高速存储和检索，以及全文搜索能力。对消息所在的库选用MyISAM存储引擎。

1. 分表。分表分两个方向，一个是水平切分，一个是垂直切分。垂直切分，把不同的业务分在不同的表中，水平切分把大量数据放在多个表中。

考虑到聊天信息数据量大的时候，可采用水平切分技术，实现数据库的分表操作，将聊天信息存到多个表上。水平切分需要考虑跨库join，有关联的表需要进行配置，使其和关联的字段进行对应。

对于公用的表，可以设置成全局表，本次聊天系统中，可以把用户表设置成全局表。

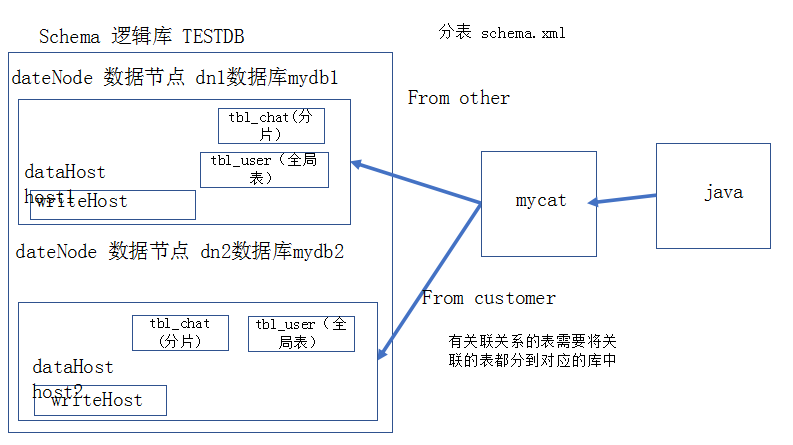
水平切分又放哪多中情况，单库多表，多库多表。

对于咱们这个项目，使用人员不是特别多，可能存在的就是聊天数据量多一些，当然数据量太大单台也扛不住。

如果考虑对数据量的处理，不考虑DB性能的话可采用单库多表，如果考虑到人群多的话，对数据库压力比较大，影响数据库性能，可以采用多库多表，将表拆分到不同的数据库。

具体的分表方案可使用mycat中间件进行操作。

分表 需要配置scheme.xml和rule.xml(分表规则表)。



分表整体布局如上图

分表规则的选取：有十几种分表规则，默认即可满足要求。

补充：

分片的原则：

1.能不分就不分，1000万以内的表，不建议分片，通过合适的索引，读写分离等方式，可以很好的解决性能问题。

2.分片数量尽量少，分片尽量均匀分布在多个DataHost上，因为一个查询SQL跨分片越多，则总体性能越差，虽然要好于所有数据在一个分片的结果，只在必要的时候进行扩容，增加分片数量。

3.分片规则需要慎重选择，分片规则的选择，需要考虑数据的增长模式，数据的访问模式，分片关联性问题，以及分片扩容问题，最近的分片策略为范围分片，枚举分片，一致性Hash分片，这几种分片都有利于扩容   
 4.尽量不要在一个事务中的SQL跨越多个分片，分布式事务一直是个不好处理的问题   
 5.查询条件尽量优化，尽量避免Select \* 的方式，大量数据结果集下，会消耗大量带宽和CPU资源，查询尽量避免返回大量结果集，并且尽量为频繁使用的查询语句建立索引。

分表会对分页查询的性能造成影响，分页查询单表查询性能最佳。

Sql优化：

分页查询，如果从百万级别的数据中查询

select \* from news order by id desc limit 0,10 耗时0.003秒

select \* from news order by id desc limit 10000,10 耗0.058秒

select \* from news order by id desc limit 100000,10  耗0.57秒

select \* from news order by id desc limit 1000000,10 耗时7.2秒

优化

select \* from news  where id >  (select id from news order by id desc  limit 1000000, 1) order by id desc limit 0,10 查询时间 0.365秒。

使用条件对id进行了筛选，在子查询 (select id from news order by id desc limit 1000000, 1) 中我们只查询了id这一个字段比起select \* 或 select 多个字段 节省了大量的查询开销

对scheme.xml的配置

<?xml version="1.0"?>

<!DOCTYPE mycat:schema SYSTEM "schema.dtd">

<mycat:schema xmlns:mycat="http://io.mycat/">

<schema name="TESTDB" checkSQLschema="false" sqlMaxLimit="100" dataNode="dn1">

<table name="tb\_chat" dataNode="dn1,dn2" rule="mod\_rule" ></table>

<table name="tbl\_user" dataNode="dn1,dn2" type="**global**" ></table>

</schema>

<dataNode name="dn1" dataHost="host1" database="gy\_im" />

<dataNode name="dn2" dataHost="host2" database="gy\_im" />

<dataHost name="host1" maxCon="1000" minCon="10" balance="2"

writeType="0" dbType="mysql" dbDriver="native" switchType="1" slaveThreshold="100">

<heartbeat>select user()</heartbeat>

<writeHost host="hostm1" url="192.168.67.1:3306" user="root"

password="123123">

<!-- 可以配置多个读库

<readHost host="hosts1" url="192.168.67.131:3306" user="root"

password="123123">

</readHost>

-->

</writeHost>

</dataHost>

<dataHost name="host2" maxCon="1000" minCon="10" balance="0"

writeType="0" dbType="mysql" dbDriver="native" switchType="1" slaveThreshold="100">

<heartbeat>select user()</heartbeat>

<writeHost host="hostm2" url="192.168.67.1:3306" user="root"

password="123123">

</writeHost>

</dataHost>

对rule.xml的配置

<tableRule name="mod\_rule">

<rule>

<columns>id</columns>

<algorithm>mod-long</algorithm>

</rule>

</tableRule>

<function name="mod-long" class="io.mycat.route.function.PartitionByMod">

<!-- how many data nodes -->

<property name="count">2</property>

</function>