# Mysql性能提升方案对比

1. 目前项目在高并发环境中，需要我们提供持续稳定的数据库解决方案，因此解决方 案需要包括一下特点：
2. 高可用（热备）

***解决单节点失效，可以动态路由到其他可用节点问题。***

2. 读写分离，主写，从读，从节点做负载均衡，（主从复制）

***解决读写请求的分发，主写，从读，一定程度上负载了读写的压力。（用于多读的项目场景）***

1. 多主之间做负载均衡，多主复制

***解决所有sql请求可以负载路由到不同的mysql服务。***

1. 分库分表

***解决数据存储，避免所有数据存在一个库一张表中，解决单表单库数据过大。***

1. 分区

***解决数据存储，避免所有数据存在一个库一张表中，解决单表数据过大。***

1. 目前可用的解决方案，以及优缺点对比：
2. Mysql Router（innodb cluster）
3. Innerdb clustoer一部分，可以实现热备（read-write：首个可用，其他 备用）、 请求负载（read-only：轮询）
4. read-only:可以对后端服务进行故障转移
5. 2.1.x单个节点连接限制在500以内，8.0版本官网说在5000+
6. 部署在应用服务器，可以多级部署，暂时没有做router的高可用
7. 增删节点只能重启。
8. 不对sql进行拆包检查，不会过滤sql，做对请求进行路由。

**我们可以使用其read-only：轮询模式，对请求负载均衡。**

1. MySQL Proxy

官方不建议使用与生产环境，并且目前已经不提供下载插件的链接

**完全放弃**

1. Ndb cluster（mysql cluster）
2. 分布式存储引擎，可以由多个NDBCluster存储引擎组成集群分别存放整体 数据的一部分。
3. 和Innodb一样，支持事务，内存数据库。
4. 可以和mysqld分开存在于独立的主机上，然后通过网络和mysqld通 信交互。
5. 内存需求量巨大：新版本索引以及被索引的数据必须存放在内存中，老版 本所有数据和索引必须存在与内存中。
6. 不支持创建临时表（create temporary table）
7. 不支持基于行的二进制异步复制(row 和maxed)，只支持同步复制，效率 更快。
8. 不支持分布式事务XA，多join查询会变慢，单表查询会非常快。
9. 只支持读已提交事务隔离级别。默认是可重复读。
10. 只是对简单查询提升速度，写入也会有速度提升，因为放在内存中检索， 考虑事务隔离级别带来的问题。（幻读）默认隔离级别是有效 解决了幻 读问题。
11. 多个库数据一致，同时对多个库进行写入，然后放入内存中读取。
12. 传说中的高并发数据库环境，本身多个数据库节点服务，支持失效转移， 热备功能。

**原则上只是加快了查询性能，但是额外的内存和机器都必须加大。**

**可以权衡利弊选择，特别是以后大数据量优化考虑。**

1. Mycat
2. 支持多个分表模式，支持多租户，主从模式（读写分离、热备），多库之 间二进制复制数据。
3. 但是根据项目需要改一些业务代码，来适应Mycatl的分表规则，二次开 发工作比较多，并且以后都要按着mycat的规则走。
4. 热备、集群方面都有解决方案。

**如果能容忍修改业务代码，也可以考虑，由于是java项目使用多路复用线程模型，会受到硬件和jvm性能的限制。**

1. Mysql官方分表
2. 表存储引擎为merge，MyISAM表以逻辑方式组合在一起，并作为1个 对象引用它们。
3. 通常把多个MyISAM表的结果union起来组成merge存储引擎返回结 果。

**不适合**

1. Mysql 分区

支持4种分区模式和子分区，解决数据存储一个区的问题。

**项目中根据时间分区，可以使用。**

1. 多主模式，解决多写，多读（多主之间异步复制），负载均衡，热备问题

Haproxy

* + 1. 软件负载均衡，多种路由规则，故障转移，4层7层代理，支持4万+并发。
    2. 可以使用keepalived实现热备功能。

Lvs

* + 1. 硬件负载均衡，最稳定并发最高的负载方案，配置维护繁琐，支支持4层代理。
    2. 可以使用keepalived实现热备功能。

Nginx

* + 1. 新版本支持mysql负载均衡，新版本支持4层7层代理，支持上万并发连接请求。
    2. 可以使用keepalived实现热备功能

**可以使用任意一种方案，目前使用haproxy没有问题。**

1. Mysql group replication
2. 把一组服务，多个mysql服务配置在一个组中，一个事务的提交需要经过 组内大多数节点决议，来决定该事务是否可以提交。
3. 可以维护各个节点的数据最终一致性。
4. 如果两个请求对两个节点中同一行数据进行修改操作，只有第一个请求 的事务会成功，其他会失败，高并发中会有问题。
5. 集群个数是n+1个，必须满足大多数，否则事务会一致阻塞。
6. 节点数最多9个，配置为奇数个，否则就会出现选举对半现象，造成集群 不可用。
7. 两种模式，单主和多主，所有节点只能配置同一种模式。
8. 存储引擎必须是innodb
9. 每个表必须有主键，只支持ipv4，网络带宽要好
10. 没有负载均衡概念，只是维护一组中节点数据最终一致性

**只有热备功能，高并发会有问题**

9. Maxscale

1. mariadb公司开发一款负载均衡中间件，支持主从读写分离，同样主从之 间通过异步复制同步数据
2. 可以配置只读服务和读写服务节点，支持单节点故障转移和自动恢复
3. 需要解析sql语句然后去路由。
4. 效率上低于mysql router和负载均衡中间件。
5. 可以通过权重比来配置节点路由策略。(readConnRouter)
6. 不支持分库分表

**解析sql进行多主的负载一定程度上有并发性能的损耗**

10. tddl阿里的分布式数据库

**目前文档很少，并且没有完全开源，不建议使用**

1. OneProxy

数据库中间件，阿里工程师开发，具有一下特点：

1. 读写分离，自动故障切换
2. 高可用，可以结合keepalived这只浮动Ip
3. 主从服务器延时检测
4. 动态扩容
5. 性能统计
6. 连接池，禁止存储过程调用
7. 水平分区表，通过配置分表后缀，需要提前创建预设的分表
8. 不支持部分函数查询count distinct、avg、having、跨库join
9. 目前最高只支持mysql5.7版本
10. 主从复制使用的是同步复制，需要使用Mysql group replication进行组 内同步复制，这个高并发修改时，有问题。

**网上有一部分使用案例，如果时间允许可以尝试，但**

**不建议生产使用**。

1. Sharding-jdbc

当当网开发的分库分表方案，就一个jar包，通过mybatis配置分表索引号，然后集成重写分表实现类的方式进行分表。

**需要写额外代码，分表越多二次开发工作量也越大。**

1. Rds

**阿里的分布式数据库，需要收费**

1. TiDB

PingCAP 公司受 Google Spanner / F1 论文启发而设计的开源分布式数据库

**目前使用的不多，坑比较多，不建议使用**

1. 巨山数据库

Nosql文档性数据库，不能直接支持sql，需要其他中间件才能支持

对spark有比较好的支持

有社区版和企业版

**不直接支持sql语句，不适合**

1. 可使用且符合项目场景的方案

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中间件 | 高可用（热备） | 负载均衡（同异步复制） | 分表分区 | 稳定性 |
| Mysql Router |  |  | mysql自带分区 |  |
| haproxy、lvs、nginx | * e |  | mysql自带分区 |  |