1.前言

1.1 概念介绍:

1.1.1、分区

对业务透明，分区只不过把存放数据的文件分成了许多小块，例如mysql中的一张表对应三个文件.MYD,MYI,frm。

根据一定的规则把数据文件(MYD)和索引文件（MYI）进行了分割，分区后的表呢，还是一张表。分区可以把表分到不同的硬盘上，但不能分配到不同服务器上。

- 优点：数据不存在多个副本，不必进行数据复制，性能更高。
- 缺点：分区策略必须经过充分考虑，避免多个分区之间的数据存在关联关系，每个分区都是单点，如果某个分区宕机，就会影响到系统的使用。

1.1.2、分片

对业务透明，在物理实现上分成多个服务器，不同的分片在不同服务器上。如HDFS。

1.1.3、分表

同库分表：所有的分表都在一个数据库中，由于数据库中表名不能重复，因此需要把数据表名起成不同的名字。

- 优点：由于都在一个数据库中，公共表，不必进行复制，处理更简单。
- 缺点：由于还在一个数据库中，CPU、内存、文件IO、网络IO等瓶颈还是无法解决，只能降低单表中的数据记录数。表名不一致，会导后续的处理复杂（参照mysql meage存储引擎来处理）

不同库分表：由于分表在不同的数据库中，这个时候就可以使用同样的表名。

- 优点：CPU、内存、文件IO、网络IO等瓶颈可以得到有效解决，表名相同，处理起来相对简单。
- 缺点：公共表由于在所有的分表都要使用，因此要进行复制、同步。一些聚合的操作，join,group by,order等难以顺利进行。

1.1.4、分库

分表和分区都是基于同一个数据库里的数据分离技巧，对数据库性能有一定提升，但是随着业务数据量的增加，原来所有的数据都是在一个数据库上的，网络IO及文件IO都集中在一个数据库上的，因此CPU、内存、文件IO、网络IO都可能会成为系统瓶颈。

当业务系统的数据容量接近或超过单台服务器的容量、QPS/TPS接近或超过单个数据库实例的处理极限等。此时，往往是采用垂直和水平结合的数据拆分方法，把数据服务和数据存储分布到多台数据库服务器上。

分库只是一个通俗说法，更标准名称是数据分片，采用类似分布式数据库理论指导的方法实现，对应用程序达到数据服务的全透明和数据存储的全透明

来自：https://www.cnblogs.com/ijavanese/p/9512369.html

2.介绍

2.1 什么是MyCat?

简单的说，MyCAT就是：

- 一个新颖的数据库中间件产品；
- 一个彻底开源的、面向企业应用开发的“大数据库集群”；
- 支持事务、ACID、可以替代MySQL的加强版数据库；
- 一个可以视为“MySQL”集群的企业级数据库，用来替代昂贵的Oracle集群；
- 一个融合内存缓存技术、Nosql技术、HDFS大数据的新型SQL Server；

- 结合传统数据库和新型分布式数据仓库的新一代企业级数据库产品。

2.2 MyCat的目标

MyCAT的目标是：低成本的将现有的单机数据库和应用平滑迁移到“云”端，解决数据存储和业务规模迅速增长情况下的数据瓶颈问题。

2.3 MyCat的关键特性

- 支持 SQL 92标准

支持Mysql集群，可以作为Proxy使用

支持JDBC连接ORACLE、DB2、SQL Server，将其模拟为MySQL Server使用

支持NoSQL数据库

支持galera for mysql集群，percona-cluster或者mariadb cluster，提供高可用性数据分片集群

自动故障切换，高可用性

支持读写分离，支持Mysql双主多从，以及一主多从的模式

支持全局表，数据自动分片到多个节点，用于高效表关联查询

支持独有的基于E-R 关系的分片策略，实现了高效的表关联查询

支持一致性Hash分片，有效解决分片扩容难题

多平台支持，部署和实施简单

支持Catelet开发，类似数据库存储过程，用于跨分片复杂SQL的人工智能编码实现，143行Demo完成跨分片的两个表的JION查询。

支持NIO与AIO两种网络通信机制，Windows下建议AIO，Linux下目前建议NIO

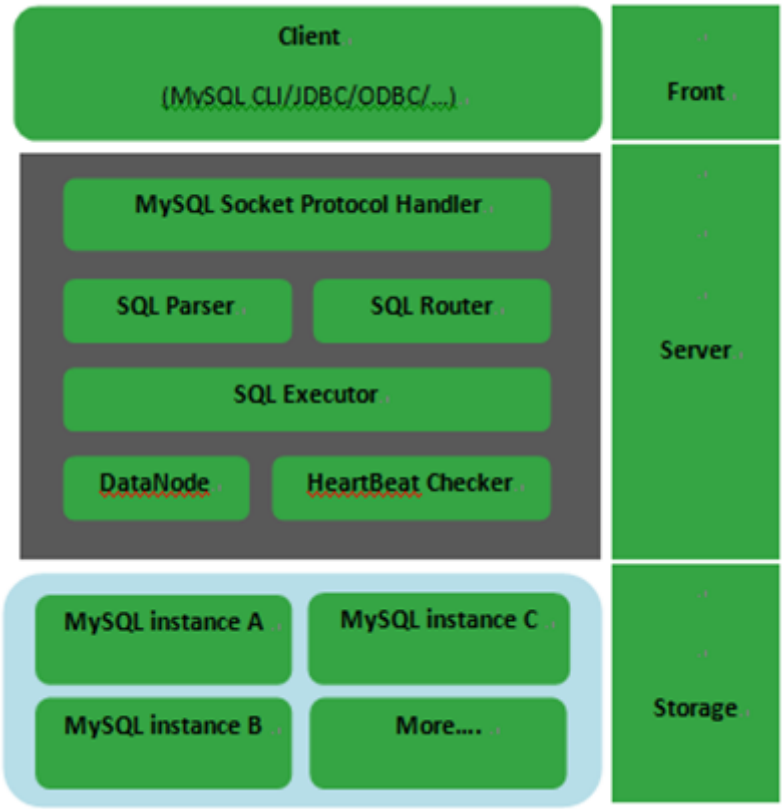
支持Mysql存储过程调用

以插件方式支持SQL拦截和改写

支持自增长主键、支持Oracle的Sequence机制

2.4 总体架构

MyCAT的架构如下图所示：



MyCAT使用MySQL的通讯协议模拟成一个MySQL服务器，并建立了完整的Schema（数据库）、Table （数据表）、User（用户）的逻辑模型，并将这套逻辑模型映射到后端的存储节点DataNode（MySQL Instance）上的真实物理库中，这样一来，所有能使用MySQL的客户端以及编程语言都能将MyCAT当成是MySQLServer来使用，不必开发新的客户端协议。

当MyCAT收到一个客户端发送的SQL请求时，会先对SQL进行语法分析和检查，分析的结果用于SQL路由，SQL路由策略支持传统的基于表格的分片字段方式进行分片，也支持独有的基于数据库E-R关系的分片策略，对于路由到多个数据节点（DataNode）的SQL，则会对收到的数据集进行“归并”然后输出到客户端。

SQL执行的过程，简单的说，就是把SQL通过网络协议发送给后端的真正的数据库上进行执行，对于MySQL Server来说，是通过MySQL网络协议发送报文，并解析返回的结果，若SQL不涉及到多个分片节点，则直接返回结果，写入客户端的SOCKET流中，这个过程是非阻塞模式（NIO）。

DataNode是**MyCAT**的逻辑数据节点，映射到后端的某一个物理数据库的一个Database，为了做到系统高可用，每个DataNode可以配置多个引用地址（DataSource），当主DataSource被检测为不可用时，系统会自动切换到下一个可用的DataSource上，这里的DataSource即可认为是Mysql的主从服务器的地址。

2.5 术语

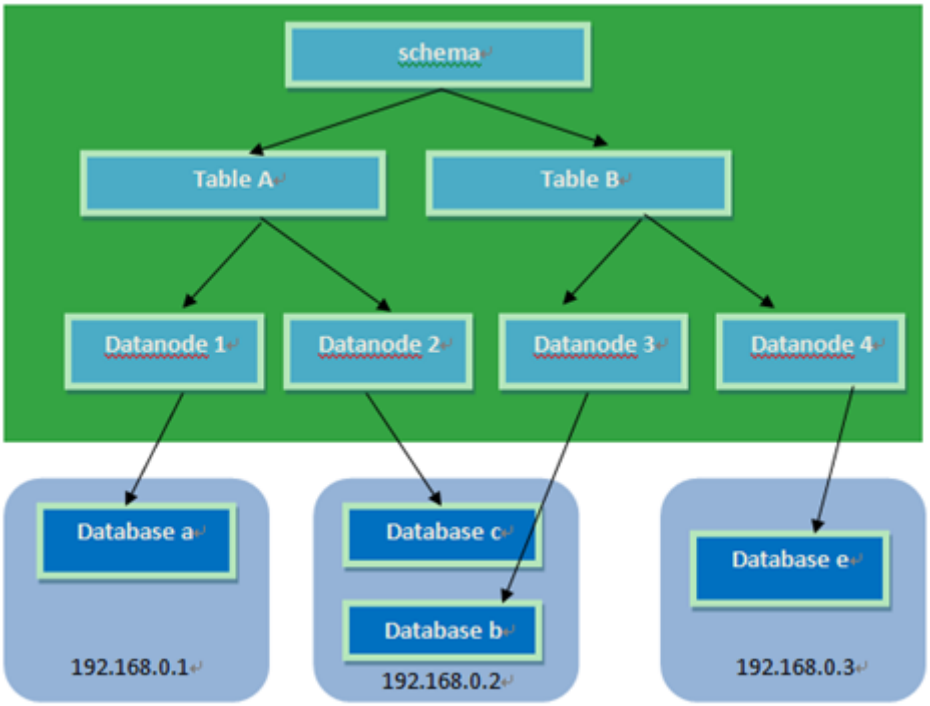
与任何一个传统的关系型数据库一样，**MyCAT**也提供了"数据库"的定义，并有用户授权的功能，下面是**MyCAT**逻辑库相关的一些概念：

- **schema:逻辑库，与MySQL中的Database（数据库）对应，一个逻辑库中定义了所包括的Table。**
- **table：表，即物理数据库中存储的某一张表，与传统数据库不同，这里的表格需要声明其所存储的逻辑数据节点DataNode，这是通过表格的分片规则定义来实现的，table可以定义其所属的"子表(childTable)"，子表的分片依赖于与"父表"的具体分片地址，简单的说，就是属于父表里某一条记录A的子表的所有记录都与A存储在同一个分片上。**
- **分片规则：是一个字段与函数的捆绑定义，根据这个字段的取值来返回所在存储的分片（DataNode）的序号，每个表格可以定义一个分片规则，分片规则可以灵活扩展，默认提供了基于数字的分片规则，字符串的分片规则等。**
- **dataNode: MyCAT的逻辑数据节点，是存放table的具体物理节点，也称之为分片节点，通过DataSource来关联到后端某个具体数据库上，一般来说，为了高可用性，每个DataNode都设置两个DataSource，一主一从，当主节点宕机，系统自动切换到从节点。**
- **dataHost：定义某个物理库的访问地址，用于捆绑到dataNode上。**

MyCAT目前通过配置文件的方式来定义逻辑库和相关配置：

- MYCAT_HOME/conf/schema.xml中定义逻辑库，表、分片节点等内容;
- MYCAT_HOME/conf/rule.xml中定义分片规则;
- MYCAT_HOME/conf/server.xml中定义用户以及系统相关变量，如端口等。

下图给出了MyCAT 一个可能的逻辑库到物理库（MySQL的完整映射关系），可以看出其强大的分片能力以及灵活的Mysql集群整合能力。


schema
Table A
Table B
Datanode 1
Datanode 2
Datanode 3
Datanode 4
Database a
192.168.0.1
Database c
Database b
192.168.0.2
Database e
192.168.0.3