Step Into OneProxy

平民软件

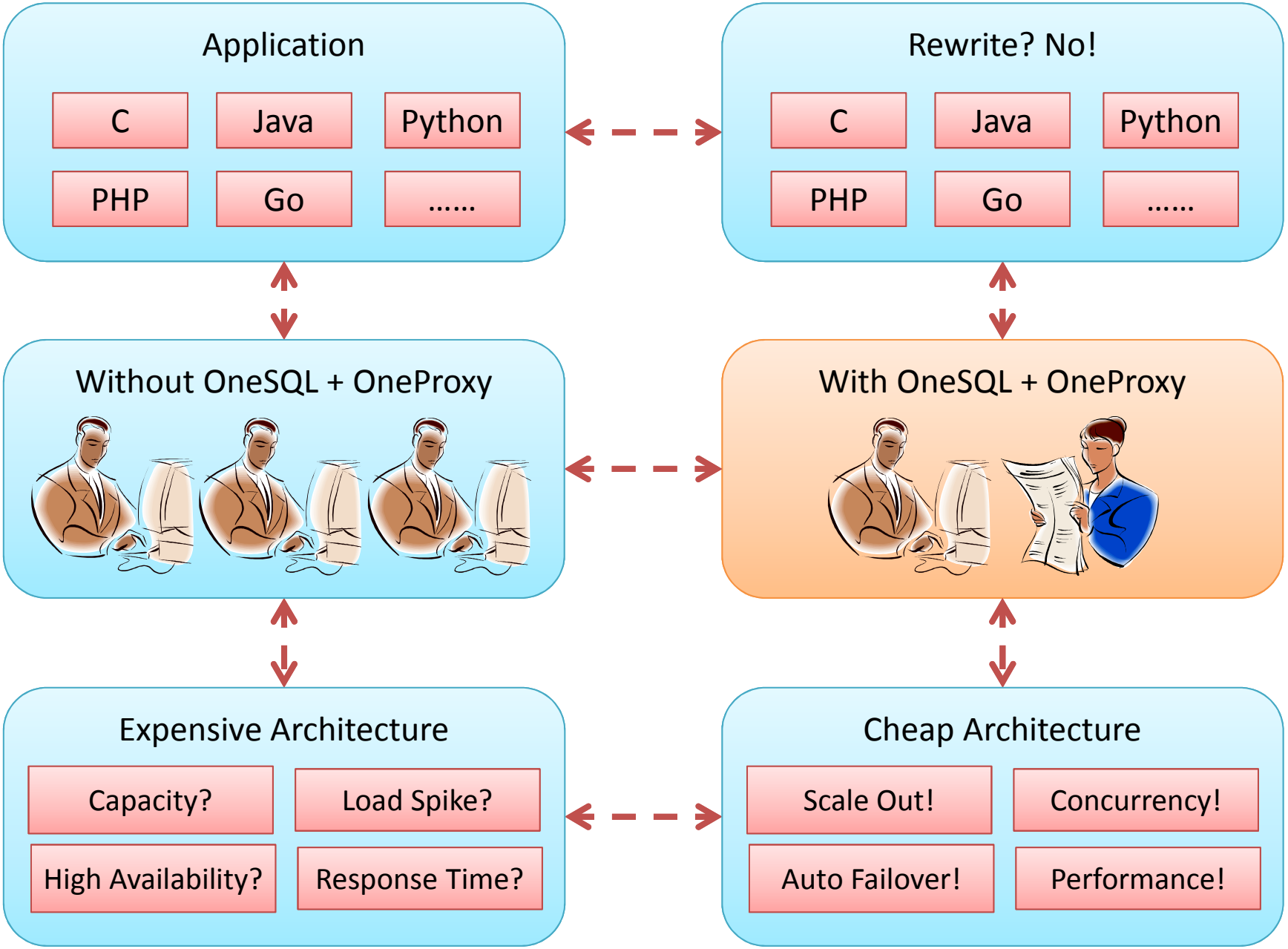
http://www.onexsoft.com

About Me

- 个人经历
  - 超过15年数据库领域经验
  - 支付宝工作5年，天猫1年
  - 设计并主导支付宝数据架构
  - 支付宝DB团队主管
  - 了解电商业务对技术的需求
  - 精通Oracle等传统架构
  - MySQL Contributor
  - 平民软件有限公司创始人
- 项目经验
  - 支付宝数据库架构规划
  - 支付宝数据库的垂直拆分
  - 支付宝数据库的水平拆分
  - 支付宝数据库去小型机
  - 支付宝数据库去高端存贮
  - 支付宝高可用(Failover)方案
  - 天猫库存秒杀场景优化
  - 天猫热点库存隔离方案

One good proxy is better than many proxies!
We need one good proxy for many business!
We are building the OneProxy for you!

http://www.onexsoft.com


Application
C
Java
Python
PHP
Go
……
Rewrite? No!
C
Java
Python
PHP
Go
……
Without OneSQL + OneProxy
With OneSQL + OneProxy
Expensive Architecture
Capacity?
Load Spike?
High Availability?
Response Time?
Cheap Architecture
Scale Out!
Concurrency!
Auto Failover!
Performance!

环境准备

- MySQL服务器
  - 创建test用户，密码test
  - 存在test数据库
  - 确认远程能登录
- OneProxy服务器
  - 可以和MySQL共同一台机器

简单实例

- 请注意安装的目录，修改ONEPROXY_HOME所指位置

```
#/bin/bash
#
export ONEPROXY_HOME=/usr/local/oneproxy

${ONEPROXY_HOME}/oneproxy --keepalive --proxy-address=:3307 \
  --proxy-master-addresses=<host>:<port>@default \
  --proxy-user-list=test/1378F6CC3A8E8A43CA388193FBED5405982FBBD3@test \
  --proxy-charset=gbk_chinese_ci
  --log-file=${ONEPROXY_HOME}/oneproxy.log \
  --pid-file=${ONEPROXY_HOME}/oneproxy.pid
```

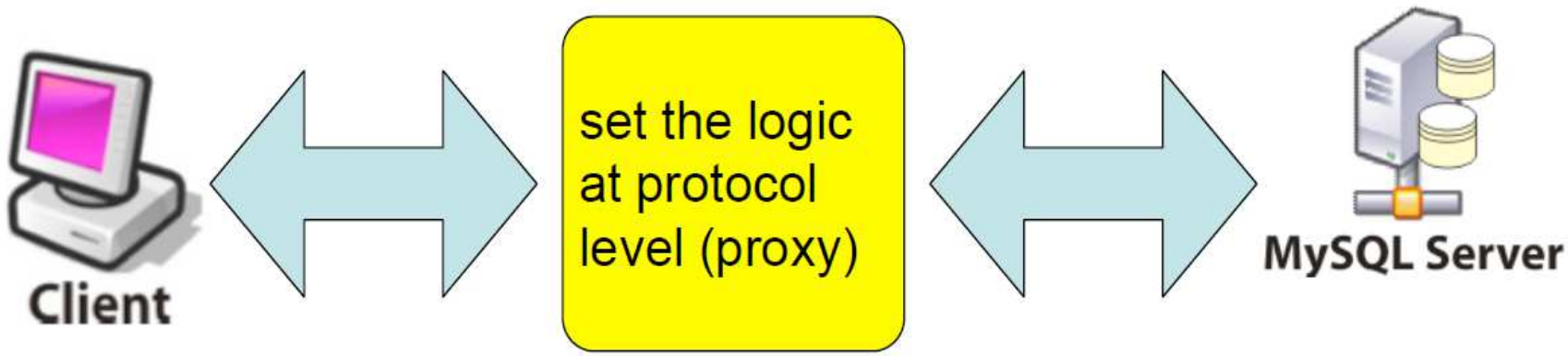
登录OneProxy

- 工作端口
  - 与后端MySQL交互的端口，操作数据
  - 命令：mysql –utest –ptest –h127.0.0.1 –P4041
- 管理端口
  - 管理OneProxy的端口，不操作数据
  - 命令：mysql –uadmin –pOneProxy –h127.0.0.1 –P4041
- 单机多实例
  - 单OneProxy占用两个端口
  - 端口不重复，可以单机启动多个实例

关闭OneProxy

- 后台运行
  - --keepalive，起动守护进程
  - 遇内存Bug Crash时，可以自动重起
  - 应用自动重新连接
- 杀死进程
  - Ps –ef | grep oneproxy
  - Kill -9 所有oneproxy进程号
- 管理端口
  - Shutdown force

协议转发



透明中间件

- 功能特色
  - 连接池
  - 单点切换
  - 读写分离
  - 分库分表
  - 跨库查询
  - 并行查询
  - 安全防火墙
  - 管理端口
- 受益部门
  - 运维
  - 架构
  - 开发
  - 安全
  - 大数据

透明兼容


phpwind


社区动力
DISCUZ!


php


Drupal

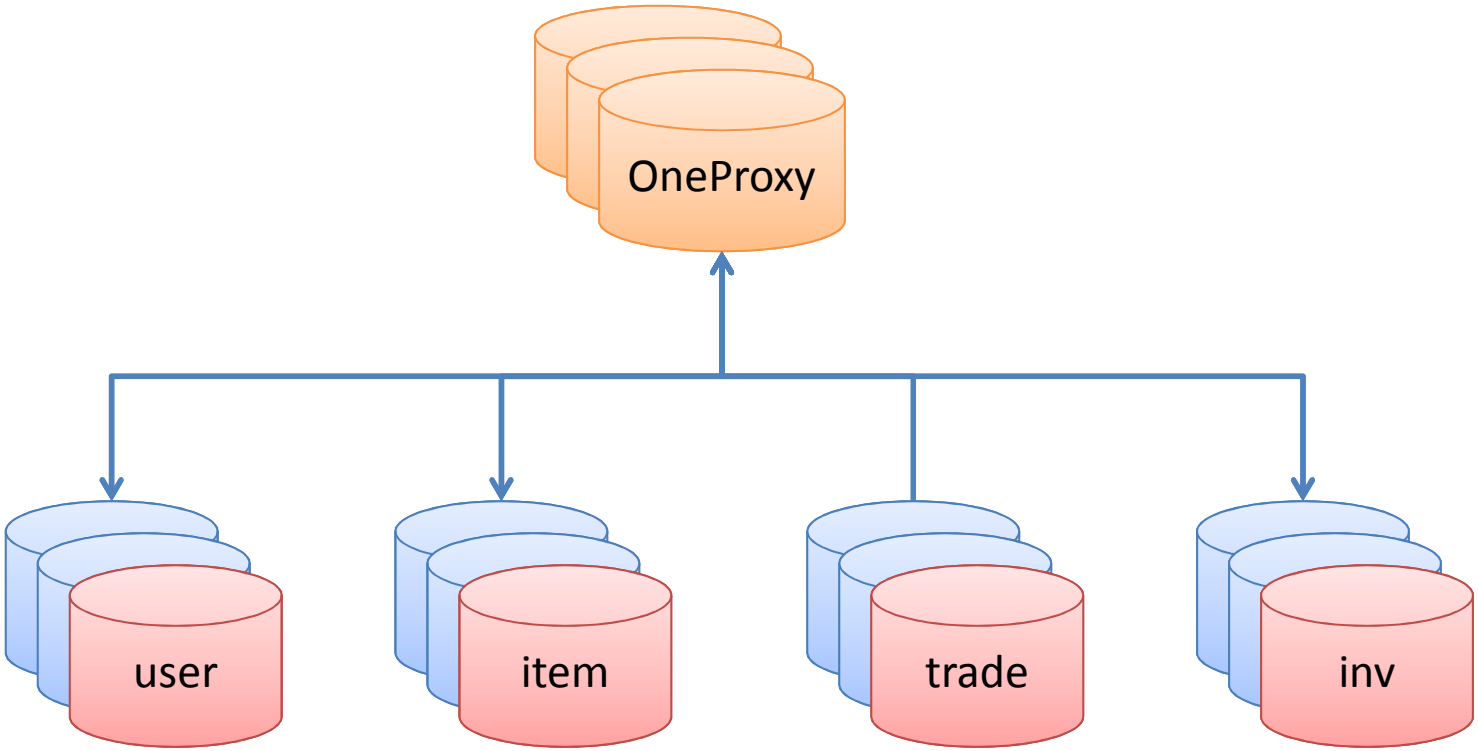

WordPress.org


C++


ecshop
ShopEx旗下
最大开源网店系统


DEDECMS

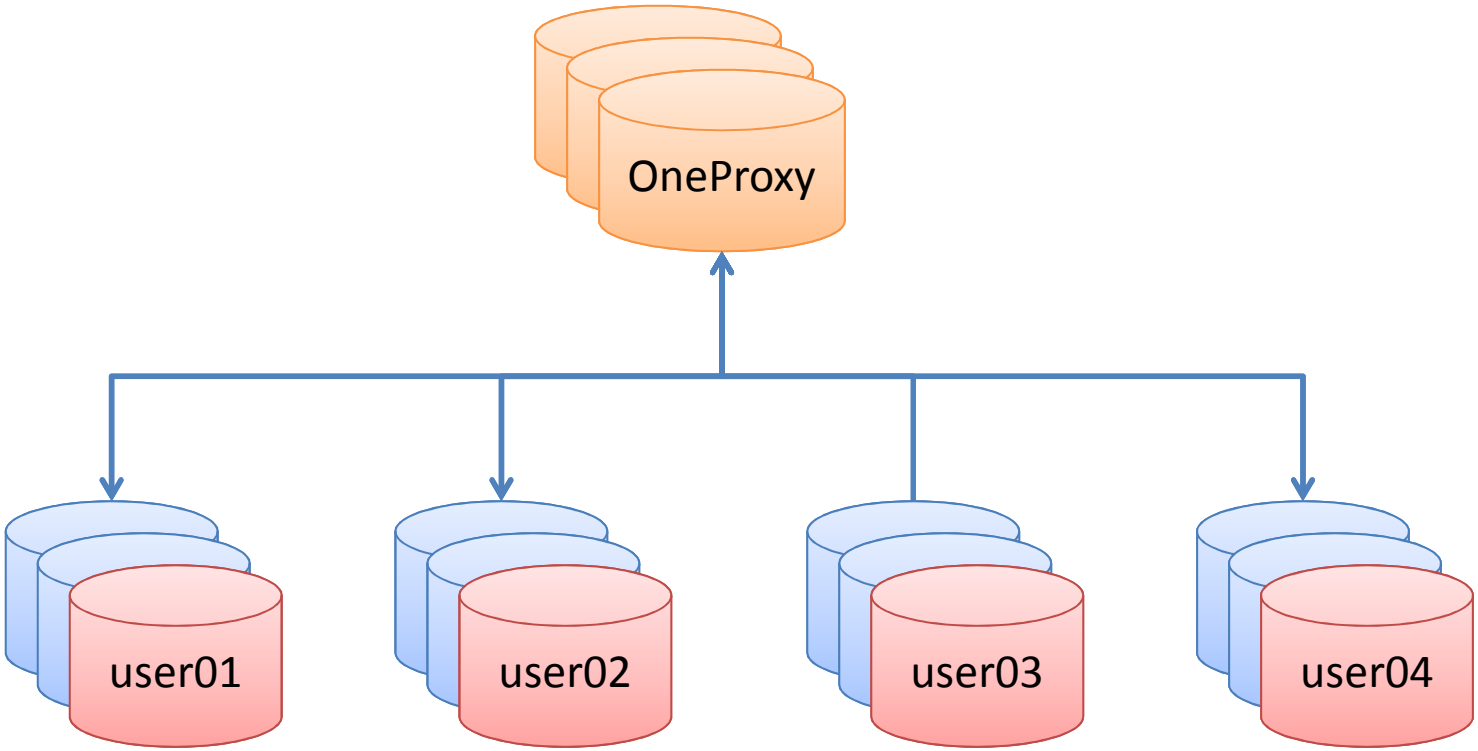
多集群管理



主机列表

集群	主机名	集群	主机名
user	userdb1	item	itemdb1
	userdb2		itemdb2
	userdb3		itemdb3
trade	tradedb1	inv	invdb1
	tradedb2		invdb2
	tradedb3		invdb3

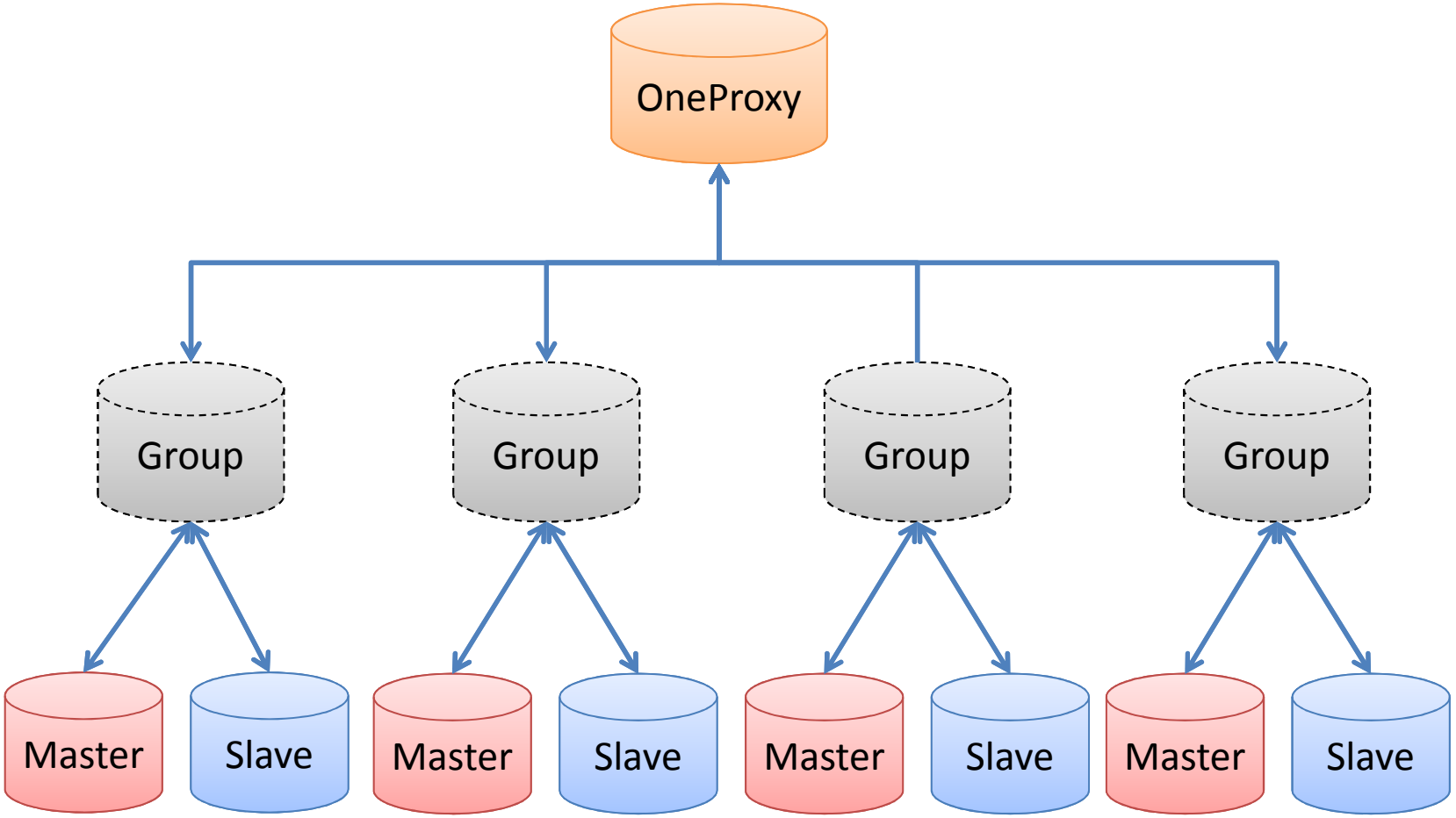
水平分片管理



机器列表

集群	主机名	集群	主机名
user01	userdb01a	user02	userdb02a
	userdb01b		userdb02b
	userdb01c		userdb02c
user03	userdb03a	user04	userdb04a
	userdb03b		userdb04b
	userdb03c		userdb04c

管理结构


OneProxy
Group
Group
Group
Group
Master
Slave
Master
Slave
Master
Slave
Master
Slave

配置主机

多集群管理

--proxy-master-addresses=userdb1:3306@user
--proxy-slave-addresses=userdb2:3306@user
--proxy-slave-addresses=userdb3:3306@user
--proxy-master-addresses=itemdb1:3306@item
--proxy-slave-addresses=itemdb2:3306@item
--proxy-slave-addresses=itemdb3:3306@item
--proxy-master-addresses=tradedb1:3306@trade
--proxy-slave-addresses=tradedb2:3306@trade
--proxy-slave-addresses=tradedb3:3306@trade
--proxy-master-addresses=invdb1:3306@invdb
--proxy-slave-addresses=invdb2:3306@invdb
--proxy-slave-addresses=invdb3:3306@invdb

水平分片

--proxy-master-addresses=userdb01a:3306@user01
--proxy-slave-addresses=userdb01b:3306@user01
--proxy-slave-addresses=userdb01c:3306@user01
--proxy-master-addresses=userdb02a:3306@user02
--proxy-slave-addresses=userdb02b:3306@user02
--proxy-slave-addresses=userdb02c:3306@user02
--proxy-master-addresses=userdb03a:3306@user03
--proxy-slave-addresses=userdb03b:3306@user03
--proxy-slave-addresses=userdb03c:3306@user03
--proxy-master-addresses=userdb04a:3306@user04
--proxy-slave-addresses=userdb04b:3306@user04
--proxy-slave-addresses=userdb04c:3306@user04

集群名字

- OneProxy用名字来管理集群
- 一台MySQL服务器属于一个集群
- 有复制关系的MySQL属于同一集群
- 集群名字由如下参数决定
  - --proxy-master-addresses=host:port@集群名
  - --proxy-slave-addresses=host:port@集群名
- 如果不指定集群名，则属于”default”集群

复杂脚本

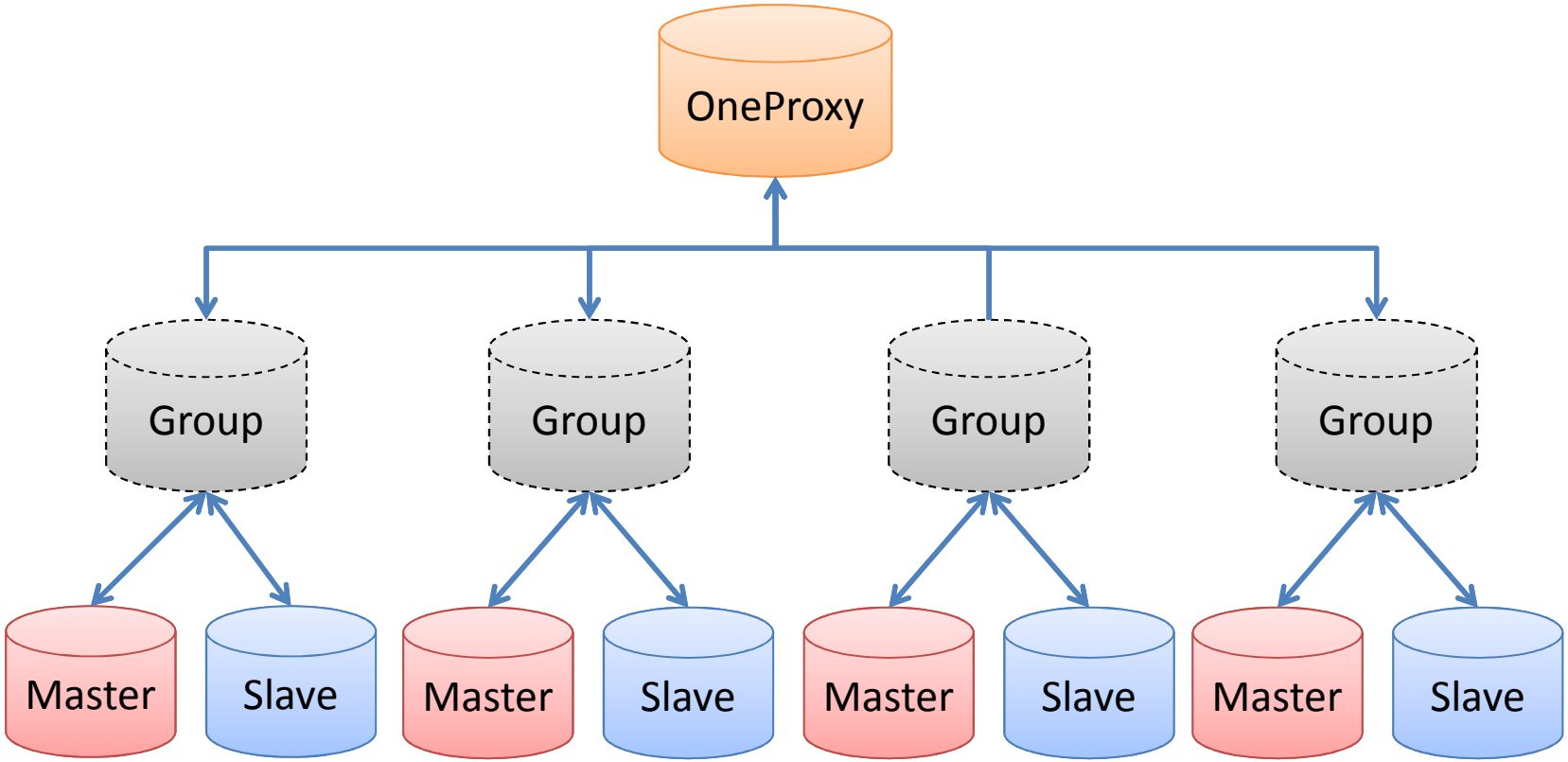
```
#/bin/bash
#
export ONEPROXY_HOME=/data/oneproxy

# valgrind --leak-check=full --show-reachable=yes \
${ONEPROXY_HOME}/oneproxy --keepalive --proxy-address=:3307 \
  --proxy-master-addresses=192.168.1.119:3306@default \
  --proxy-slave-addresses=192.168.1.20:3306@default \
  --proxy-user-list=test/1378F6CC3A8E8A43CA388193FBED5405982FBBD3@test \
  --proxy-part-tables=${ONEPROXY_HOME}/part.txt \
  --proxy-charset=gbk_chinese_ci --proxy-found-rows \
  --event-threads=6 --proxy-group-security=default:0 \
  --repadmin-username=repadmin \
  --repadmin-password=803764B6CAA31A76ABDD249910195655E1A3296B \
  --log-file=${ONEPROXY_HOME}/oneproxy.log \
  --pid-file=${ONEPROXY_HOME}/oneproxy.pid
```

查看帮助

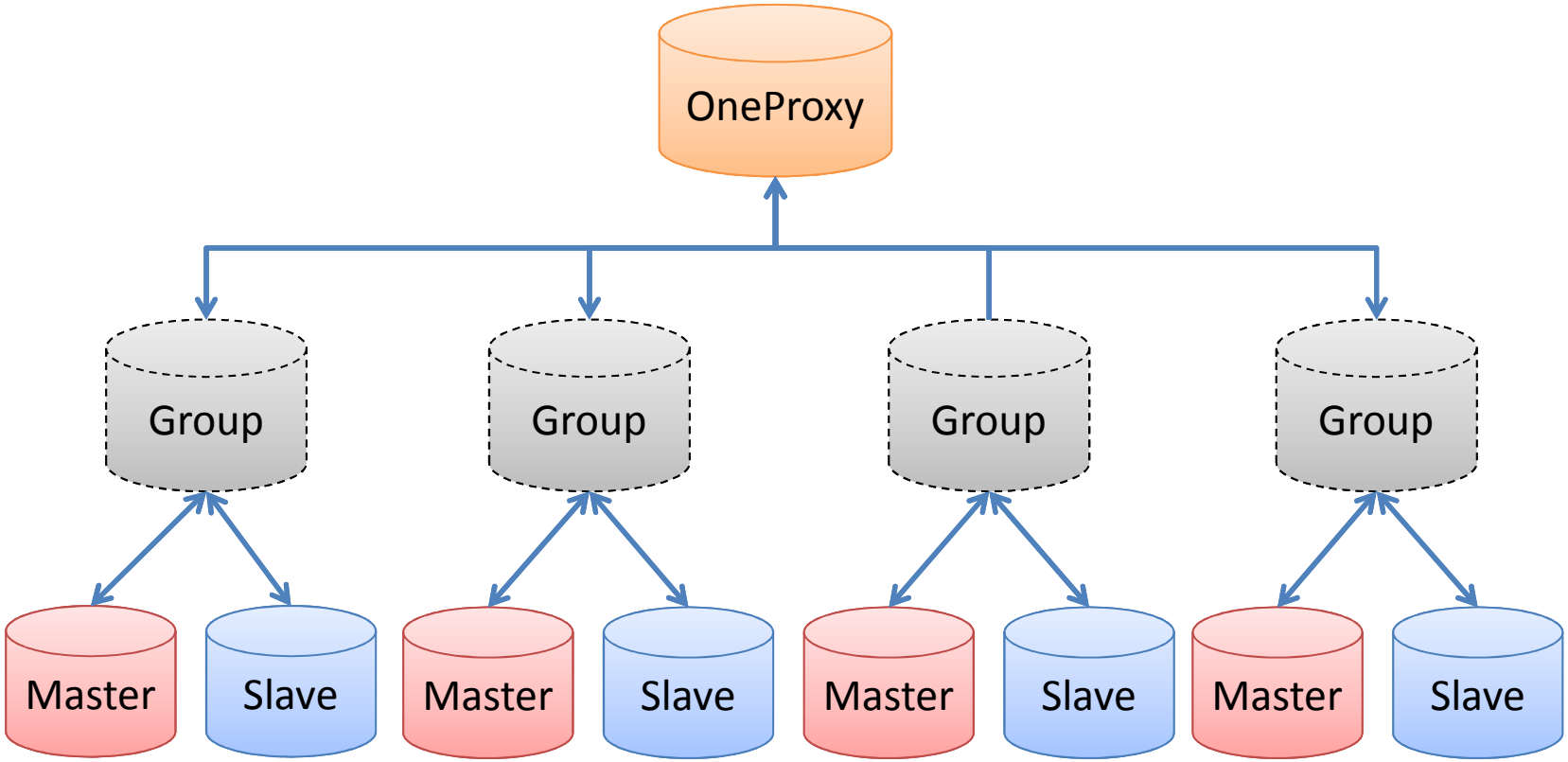
- ./oneproxy –help-all
- 了解参数
  - 基本参数
  - 读写分离相关
  - 分库分表相关
  - 安全设置相关

应用登录

Proxy host : port : group



数据库登录



user / password @ dbname

登录设置

- 设置参数
  - --proxy-user-list=username/password@dbname
  - 密码必须加密，使用管理端口passwd命令生成
  - --proxy-database: 指定默认数据库
  - --proxy-charset: 指定数据库字符集
- 注意事项
  - 所有MySQL节点有相同的用户名和口令
  - 所有MySQL节点有相同的数据库名
  - 所有MySQL节点使用相同的字符集设置
  - 前端应用程序可以切换操作集群
    - 命令：use groupname
  - 前端应用程序不能切换MySQL数据库

健康检查

- 每秒钟Proxy会和后端进行通信，检测后端数据库状态。
- 后端状态
  - Markup: 正常状态，可以提供服务
  - Markdown：异常状态，不可以提供服务
- 查看状态
  - 登录管理端口：list backend
  - 查看日志文件：tail -100f oneproxy.log

多个数据库

- OneProxy的use命令变成切换集群
- OneProxy不支持数据库切换
  - 程序里不可以有use命令（不良习惯）
- 可以设定多个用户（不同的默认库）
  - --proxy-user-list=user/pass@数据库名
  - --proxy-user-list=user/pass@数据库名
- 如果不指定，则默认的数据库名为
  - --proxy-database (默认为test)

密码加密

- --proxy-user-list中的密码部份必须加密
- 加密方法：
  - 起动OneProxy
  - 登录到OneProxy管理端口，默认4041
  - 默认用户名：admin，口令：OneProxy
  - 用MySQL客户端登录，执行passwd命令
- 加密方法：
  - OneProxy特有的passwd命令
  - ~~不同于MySQL的passwd函数~~

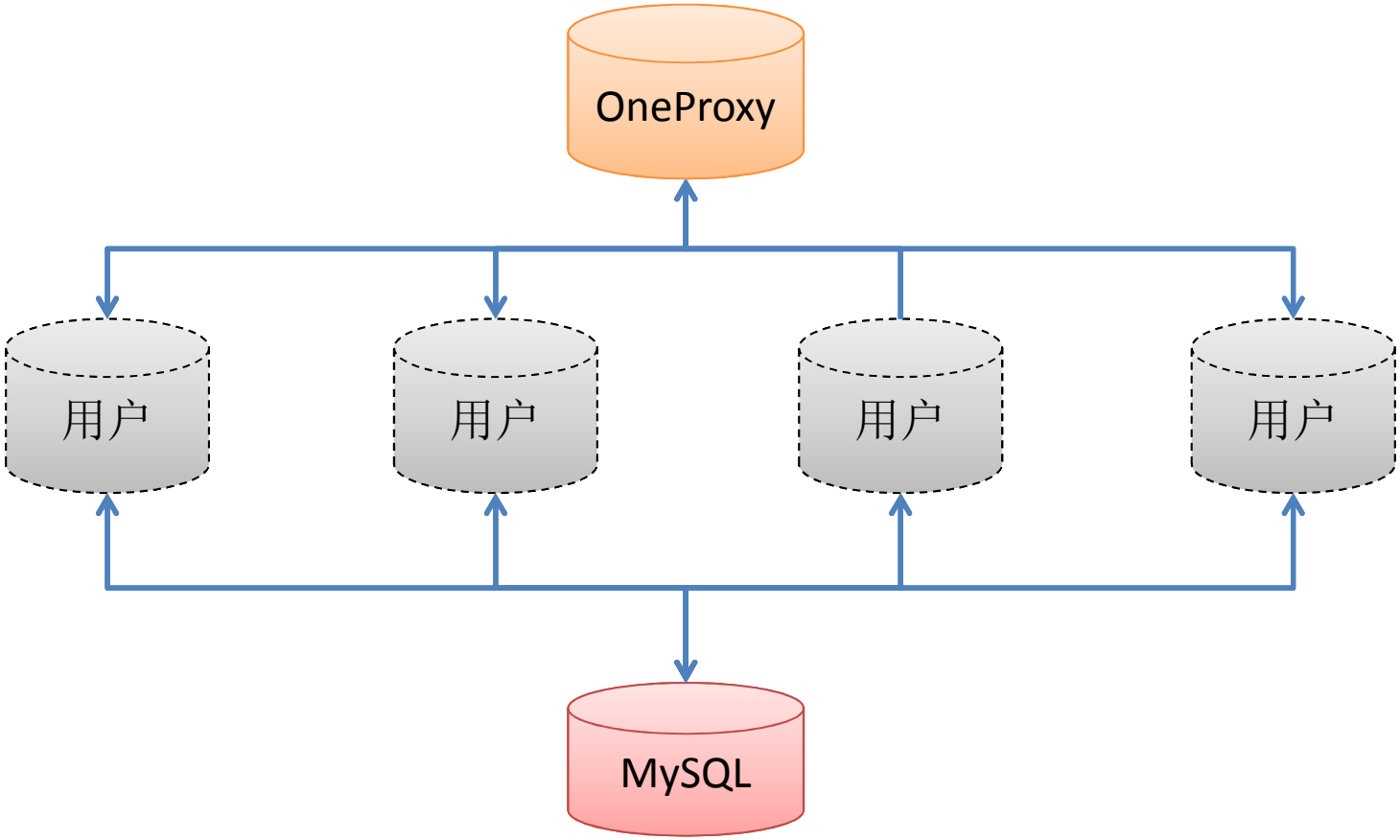
创建密码

```
[root@localhost data]# mysql -u admin -h127.0.0.1 -P4041 -pOneProxy -c
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.0.99-OneProxy-admin

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> passwd 'test';
+------+-------------------------------------------+
| TEXT | PASSWORD                                  |
+------+-------------------------------------------+
| test | 1378F6CC3A8E8A43CA388193FBED5405982FBBD3 |
+------+-------------------------------------------+
1 row in set (0.00 sec)
```

两个名字

- 所属集群名字
  - --proxy-master-addresses=db_host:port@集群名
  - --proxy-slave-addresses=db_host:port@集群名
- 数据库名字
  - --proxy-user-list=用户名/口令@默认数据库
- OneProxy可切换集群
  - 修改了use 命令，程序中不应当用use命令
- OneProxy不可切换数据库
  - 连接池里每个连接必须保持一致。

连接池



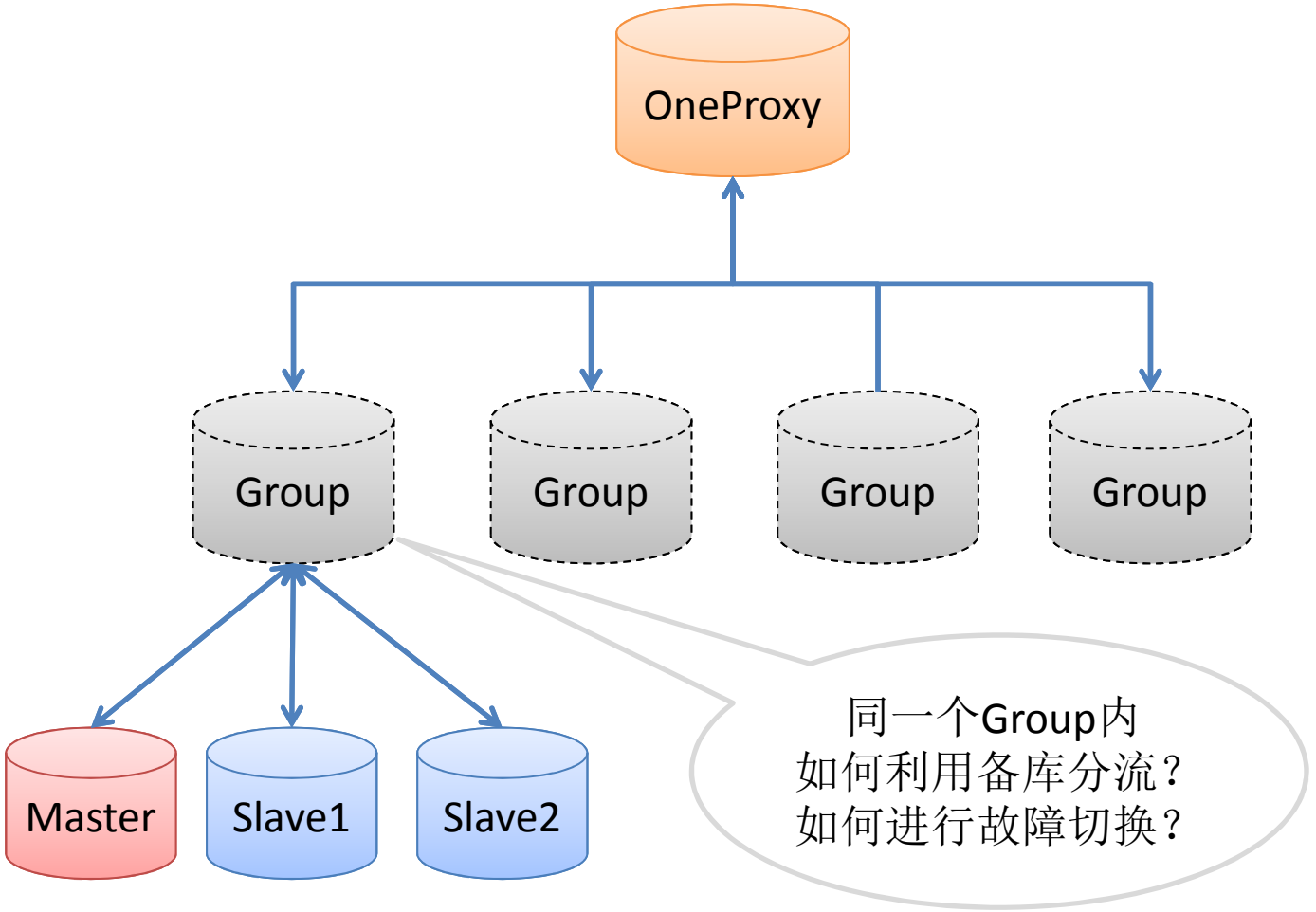
连接保护

- 优点：
  - 接管登录验证，降底MySQL负载
  - 减少建立连接时间，提升响应时间
  - 阻止连接风暴对MySQL的伤害
  - 对应用透明，对开发语言无要求
  - 无法禁用连接池，必备功能
- 限制：
  - 不支持会话级设置，直接禁用SET命令
  - 不支持会话级切换后端数据库
  - 不支持MySQL C 客户端的Prepare接口
  - 不支持服务器端的Prepare命令

连接回收

- OneProxy每10秒钟检一次连接池。
- 自动回收5分钟不使用的连接。
- 要求数据库的超时设置大于300秒 + 10秒
  - 确认inactive_timeout设置
  - 确认net_read_timeout设置
  - 确认net_write_timeout设置

服务分流


OneProxy
Group
Group
Group
Group
Master
Slave1
Slave2
同一个Group内
如何利用备库分流？
如何进行故障切换？

分流策略

分流策略	读操作	写操作
Master-only	master	master
Read-failover	master, slave if no master	master
Read-slave	slave, master if no slave	master
Read-balance	any	master
Big-slave	slave, master if no slave	master
Big-balance	any	master
Write-failover	any	master (auto failover)
Write-balance	any	any

复杂查询

- 无Where条件
- 有Order By字段
- 有Group By字段
- 有汇总函数
- 涉及两个以上表
  - 关联
  - 子查询

分流设置

- 设置参数
  - --proxy-group-policy=<group name>:<policy name>
  - 管理端口：set gpolicy group_name policy_name
  - 查看设置：list group
- 注意事项
  - 需要结合业务场景考虑。
  - 默认为master only，即读写都通过master节点。
  - 除非经过认真考虑，同group请保证只有一个master节点。
  - OneProxy不能自动补齐多台Slave的Binlog位置。

切换优势

- 应用无感知
  - 后端切换，应用到OneProxy连接不断
  - 秒级后端状态检测
  - 需要注意密码准确性（max_connect_errors）
- 不依赖DNS
  - 应用程序有DNS缓存时间，切换时间长
- 无VIP限制
  - 过多依赖外部程序
  - 有同网段的限制

多主设置

- 自动选主
  - 默认自动选主，保障写服务可用
  - 需要根据业务接受度来设置
- 设置参数
  - --proxy-manual-master=<group name>
  - 管理端口：set gmanual group_name {0|1}
  - 查看设置：list group
- 注意事项
  - 同group有多个RW类型时，一个master失败，会自动选择另一个来作为master。
  - 自动选主时还不能保证数据同步。
  - 根据业务要求决定能否自动选主。
  - 在双主强同步或XtraDB Cluster下可考虑。
  - OneProxy不能自动补齐多台Slave的Binlog位置

主备时延

- 时延检测
  - 默认不进行备库时延检测
- 设置参数
  - --repadmin-username 复制管理用户名
  - --repadmin-password 复制管理用户口令
  - --proxy-group-slavedelay=<group name>:秒数
  - 需要注意密码准确性（max_connect_errors）
- 注意事项
  - 所有后端MySQL有同样的用户名和口令
  - 确保可执行show master status命令
  - 确保可执行show slave status命令
  - 建议可执行start slave {io_thread | sql_thread}命令
  - IO/SQL Thread其中之一停止时，会偿试重起
  - 默认使用Seconds_Behind_Master的值

时延算法

- 更加精确
  - 创建oneproxy_replication_check表。
  - 定时更新master节点上的表
  - 支持级联的情况
- 参数设置
  - --proxy-replication-check
  - 时延查看：list backend
- 主意事项
  - 默认库（--proxy-database）建表权限
  - 只对一个Group中的所选master节点进行更新

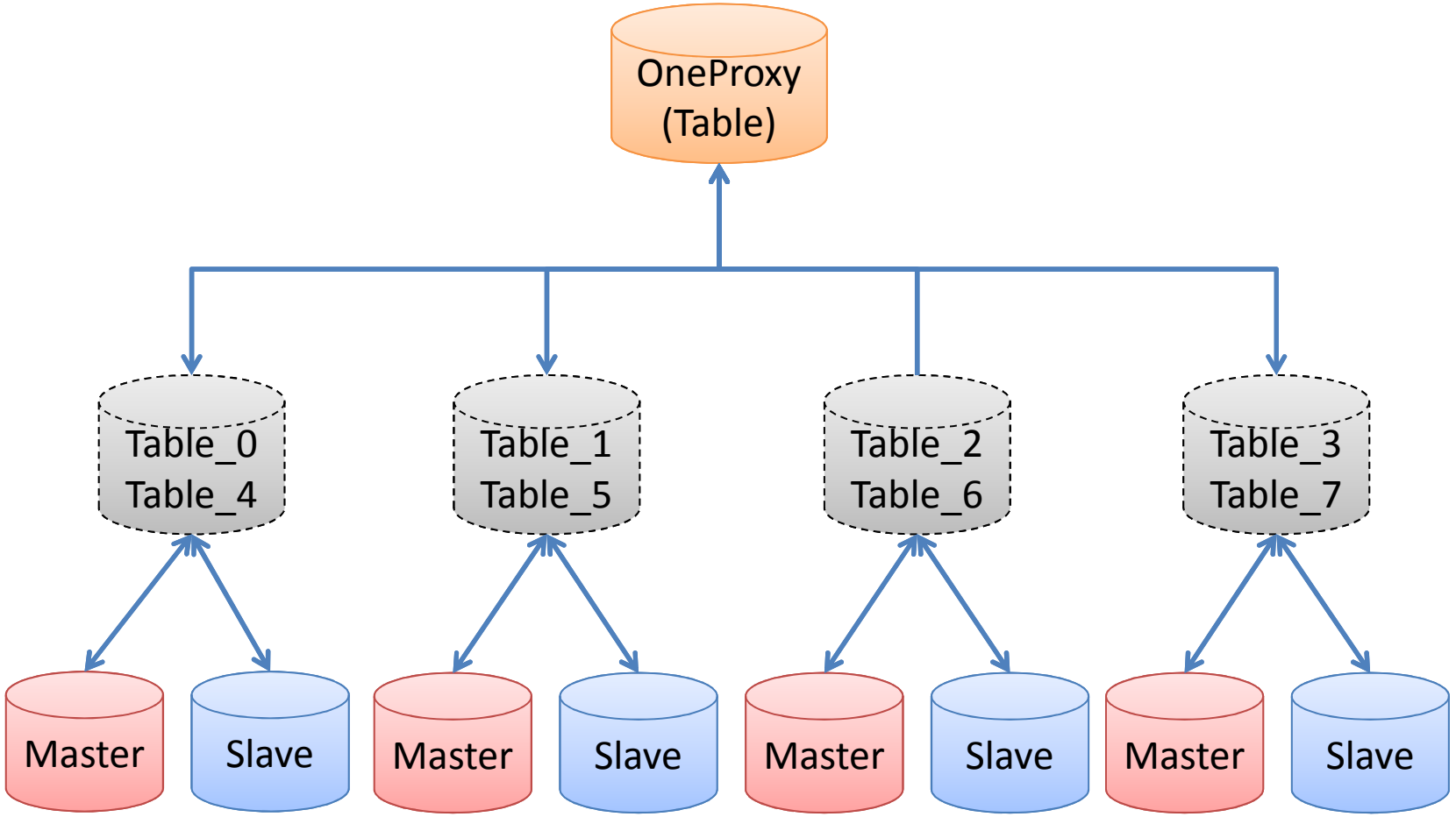
权限设置

- DDL权限
  - 默认禁止DDL （Create/Alter/Drop命令）
- 设置参数
  - --proxy-group-security=<group name>:{0|1|2|4|8}
  - 管理端口：set gaccess group_name {0|1|2|4|8}
  - 查看设置：list group
- 注意事项
  - 级别0：没有限制
  - 级别1：禁止DDL
  - 级别2：必须要有where条件
  - 级别4：禁止Delete语句
  - 级别8：禁止DML，只读状态

表级设置

- 针对个别表设置
- 设置参数
  - --proxy-table-security=<table name>:{0|1|2|4|8}
  - 可以使用多次，指定多个表
- 注意事项
  - 级别0：没有限制
  - 级别1：禁止DDL
  - 级别2：必须要有where条件
  - 级别4：禁止Delete语句
  - 级别8：禁止DML，只读状态

分库分表


OneProxy
(Table)
Table_0
Table_4
Table_1
Table_5
Table_2
Table_6
Table_3
Table_7
Master
Slave
Master
Slave
Master
Slave
Master
Slave

分表算法

- Global
  - 全局表，每个节点留一份
- Range
  - 按某个字面的范围进行分表
- List
  - 按某个这段的值进行分表
- Hash
  - 按某个字段的值的Hash值进行分表（OneProxy算法）
  - 数字类型，hash值等于数值
  - 分区数取模
- Crc32
  - 同Hash分区，Hash值算法为标准的CRC32
  - 数字类型，hash值等于数值
  - 分区数取模

Global类型

- 特征
  - DDL/DML所有的节点处理
  - SELECT操作只查当前节点
- 作用
  - 数据不分区，镜象到所有节点
  - 解决分区表与非分区表的关联问题
- 注意
  - 未用分布式事务保证
  - 用Replace替换Insert
  - 必须要有主键

Global分表

```
{
    "table"   : "my_global",
    "pkey"    : "id",
    "type"    : "int",
    "method"  : “global",
    "partitions":
      [
        {"group": “server1"},
        {"group": “server2"},
        {"group": “server3"},
        {"group": “server4"}
      ]
 }
```

Range分表

```
{
    "table"   : "my_range",
    "pkey"    : "id",
    "type"    : "int",
    "method"  : "range",
    "partitions":
      [
        { "suffix" : "_0", "group": “server1", "value" : 100000 },
        { "suffix" : "_1", "group": “server2", "value" : 200000 },
        { "suffix" : "_2", "group": “server3", "value" : 300000 },
        { "suffix" : "_3", "group": “server4", "value" : null    }
      ]
}
```

List分表

```
{
    "table"   : "my_list",
    "pkey"    : "id",
    "type"    : "int",
    "method"  : "list",
    "partitions":
      [
        { "suffix" : "_0", "group": “server1", "value" : [1,2,3] },
        { "suffix" : "_1", "group": “server2", "value" : [4,5,6] },
        { "suffix" : "_2", "group": “server3", "value" : [7,8,9] },
        { "suffix" : "_3", "group": “server4", "value" : [10,11,12] },
        { "suffix" : "_4", "group": “server5", "value" : [] }
      ]
}
```

Hash分表

```
{
    “table”   : “my_hash”,
    "pkey"    : "id",
    "type"    : "int",
    "method"  : “hash",
    "partitions":
      [
        { "suffix" : "_0", "group": “server1"},
        { "suffix" : "_1", "group": “server2"},
        { "suffix" : "_2", "group": “server3"},
        { "suffix" : "_3", "group": “server4"}
      ]
}
```

Crc32分表

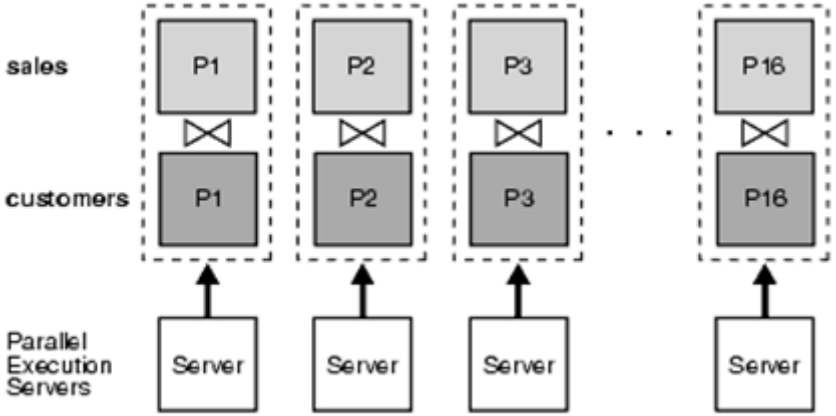
```
{
    “table”   : “my_crc32”,
    "pkey"    : "id",
    "type"    : "int",
    "method"  : “crc32",
    "partitions":
      [
        { "suffix" : "_0", "group": “server1"},
        { "suffix" : "_1", "group": “server2"},
        { "suffix" : "_2", "group": “server3"},
        { "suffix" : "_3", "group": “server4"}
      ]
}
```

分表设计

- 分区字段可以是CHAR或INT或DATE
- 关联查询的表，用相同的维度进行分表
- 可对分表信息进行冗余
  - 用户号最后两位做分表
  - 交易号前两位冗余用户号最后两位

跨库处理

- DML操作不能跨机器
  - 受制于分布式事务
- JOIN操作不能跨机器
  - 受制于SQL优化器层功能
  - 建议大表用相同唯度拆分
  - 建议小表使用Global分表
- 受限制的结果集合并
  - 在OneProxy内存中处理
  - 支持简单结集合并
  - 支持排序、分页、分组汇总
  - 对avg/count(distinct)/having有限制，无法查询
  - std/variance非分区键分组统计，取各分片平均值



查询合并

- 支持
  - select * from my_range order by id;
  - select * from my_range order by id limit 2, 3;
  - select col2, sum(id) from my_range group by col2;
  - select distinct col2 from my_range;
- 不支持（仅在查询涉及多分片时）
  - select count(distinct …) from …
  - select avg(…) from …
  - select … from … group by … having count(*) > …

预演模式

- 参数设置
  - --proxy-dryrun-mode
- 运行机制
  - 装载分库分表的规则
  - 对所有的SQL进行规则检测
  - 转发原有的SQL到数据库端，不影响结果
- 方案测试
  - 用于快速测试分库分表方案
- 查看结果
  - 管理端口：list sqlfail

分析模式

- 参数设置
  - --proxy-trans-debug
- 运行机制
  - 分析事务和复杂SQL中的表
  - 记录表与表之间的使用关系和频率
- 架构分析
  - 选择最佳的分库分表方案
- 查看结果
  - 管理端口：list trans_debug

并行查询

- 可以按分区进行并行查询
- 除跨库查询限制外，无限制
- 增强实时查询能力
  - 查询涉及多个分表才能并行
- 减少大数据实时同步需求
- 功能测试：
  - 单机10亿条记录count，20多秒
  - 增加机器可获得线性扩展

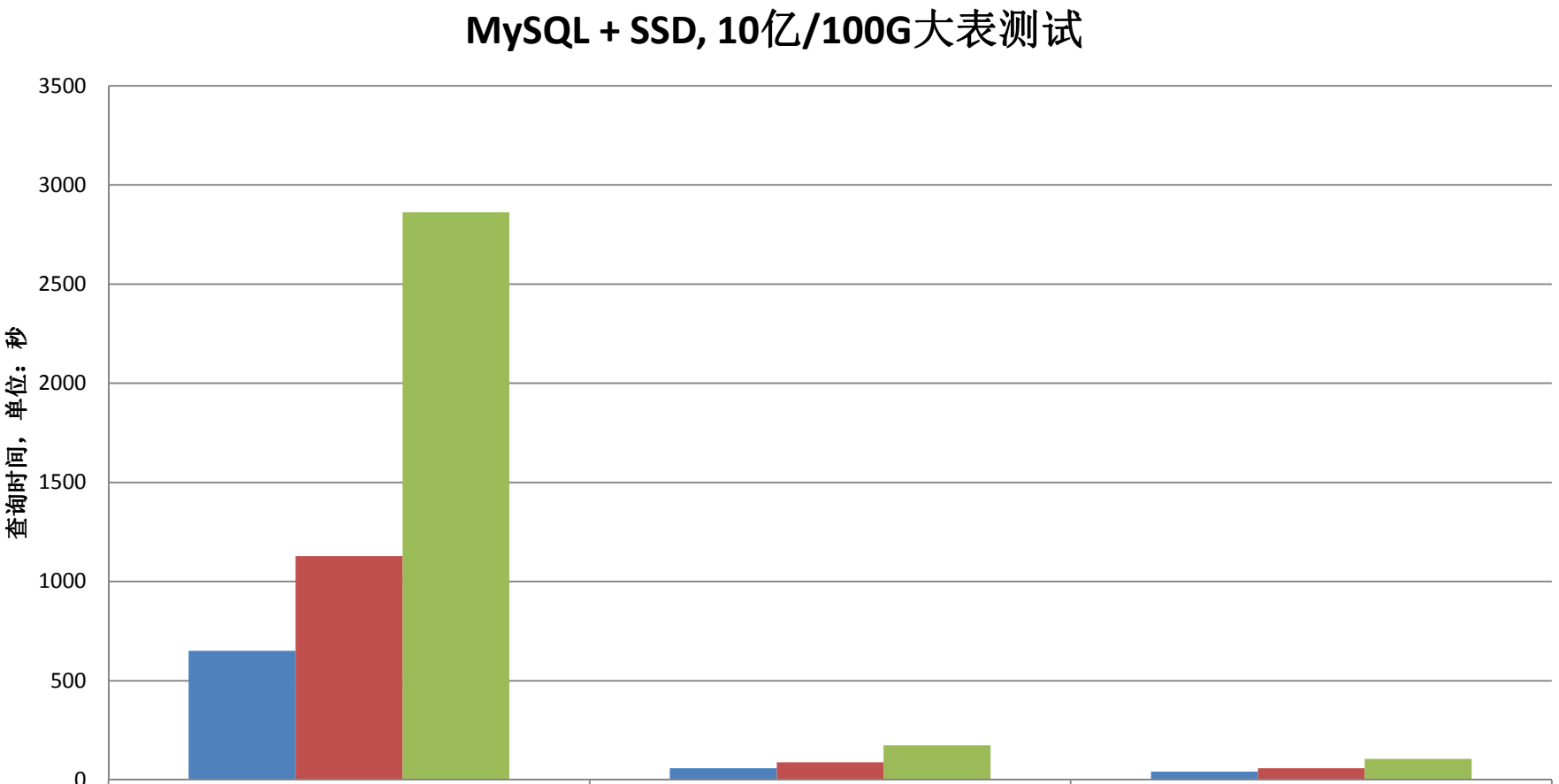
并行测试

- 测试用例
  - 工具：OneProxy, mydbtest
  - 透明分成256个子表（10个字段）
  - 生成10亿条数据
- 测试环境
  - 单机 12Core (24 Thread)
  - PCIE SSD
- 参数设置
  - MySQL Data Buffer 12GB
  - InnoDB数据文件100GB
- 测试目的
  - 考验SSD的IO能力
  - 测试OneProxy/MySQL的并行查询能力

测试语句

- Select /* parallel */ count(*) from bigtable
- Select /* parallel */ col2, count(*) from bigtable group by col2
- Select /* parallel */ col2, count(*), sum(col3), sum(col4), max(id) from bigtable group by col2

测试结果



	无并发	16并发	24并发
计数	651	58	42
汇总1	1129	89	59
汇总2	2861	174	106

安全功能

- 非法命令拦截和检测
  - 非SQL关键开头的SQL指令
  - 禁止call/prepare/execute/deallocate命令
- 二次认证(开发中)
  - 除数据库密码之外的保护
  - 基于客户端/服务器上一致算法的动态口令
- IP白名单
  - 只有指定的客户端才能连接
- SQL白名单
  - 只有指定的SQL才能被执行
- 敏感信息的SQL签名
  - 重要表的SQL在需要发送签名

黑白名单

- 黑名单
  - 设置层层规则，SQL响应时间增加
  - 偿试几次后，很容易绕过规则
  - 传统防火墙安全机制做法
- 白名单
  - OneProxy强烈推荐
  - 只有法律规定的才可以执行
  - 对SQL响应时间无影响，超高性能
  - 需要收集SQL语句，可开启观察模式

签名机制

```
mysql> desc t_safe;
+-------+---------+------+-----+---------+-------+
| Field | Type    | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| col1  | int(11) | NO   | PRI | NULL    |       |
| col2  | int(11) | NO   |     | NULL    |       |
+-------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc t_unsafe;
+-------+---------+------+-----+---------+-------+
| Field | Type    | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| col1  | int(11) | NO   | PRI | NULL    |       |
| col2  | int(11) | NO   |     | NULL    |       |
+-------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> select * from t_unsafe;
+------+------+
| col1 | col2 |
+------+------+
|    1 |   10 |
+------+------+
1 row in set (0.00 sec)

mysql> select * from t_safe;
ERROR 1045 (28000): Statement was blocked by OneProxy SQL Firewall
mysql>
mysql>
```

性能数据

```
mysql> list sqlstats;
+------------+-------+-----------------------+------+---------+--------+--------+------+--------+
| HASHCODE   | EXECS | DIST                  | FAIL | ELAPSE  | AVGELA | MAXELA | ROWS | AVGROW |
+------------+-------+-----------------------+------+---------+--------+--------+------+--------+
| 4091520238 |     1 | 99,0,0,0,0,0,0,0,0,0  |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
| 1751540541 |     1 | 0,99,0,0,0,0,0,0,0,0  |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
|  403169710 |     1 | 99,0,0,0,0,0,0,0,0,0  |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
| 1280354203 |     1 | 99,0,0,0,0,0,0,0,0,0  |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
| 4272914078 |     1 | 99,0,0,0,0,0,0,0,0,0  |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
| 1677079106 |     1 | 99,0,0,0,0,0,0,0,0,0  |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
| 1929037930 |     1 | 99,0,0,0,0,0,0,0,0,0  |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
| 2191039147 |     5 | 99,0,0,0,0,0,0,0,0,0  |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
| 2678581325 |     5 | 80,0,0,0,0,0,0,0,0,20 |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
| 1463889950 |     9 | 77,0,0,0,0,0,0,0,11,11|    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
|  832761468 |     9 | 99,0,0,0,0,0,0,0,0,0  |    0 |  0.0000 | 0.0000 | 0.0000 |    0 |      0 |
+------------+-------+-----------------------+------+---------+--------+--------+------+--------+
11 rows in set (0.00 sec)

mysql> list sqltext;
+------------+-----------------------------------------------------------+
| HASHCODE   | SQLTEXT                                                   |
+------------+-----------------------------------------------------------+
| 4091520238 | select * from my_list where id in (? , ? , ?)             |
| 1751540541 | select * from my_hash where id in (? , ? , ?)             |
|  403169710 | select col2 , sum(id) from my_list group by col2          |
| 1280354203 | select * from my_list where id = ?                        |
| 1463889950 | insert into my_hash(id , col2 , col3) values(? , ? , ?)   |
| 2191039147 | update my_list set col2 = col2 + ? where id = ?           |
| 2678581325 | select * from my_list where id in (? , ?)                 |
|  832761468 | insert into my_list(id , col2 , col3) values(? , ? , ?)   |
| 4272914078 | select * from my_hash where id = ?                        |
| 1677079106 | select @@version_comment limit ?                          |
| 1929037930 | select col2 , sum(id) from my_hash group by col2          |
+------------+-----------------------------------------------------------+

mysql> list ipstats;
+---------------+------+------+------+-------+-------+--------+--------+------+--------+
| ADDRESS       | CONN | AUTH | FAIL | QUERY | ERROR | DBTIME | DBROWS | DENY | FWFAIL |
+---------------+------+------+------+-------+-------+--------+--------+------+--------+
| 192.168.1.119 |   27 |   23 |    0 | 42746 |     0 |  49675 |  42714 |    0 |      0 |
| 127.0.0.1     |  259 |  259 |    0 |     0 |     0 |      0 |      0 |    0 |      0 |
+---------------+------+------+------+-------+-------+--------+--------+------+--------+
2 rows in set (0.00 sec)

mysql> list tabstats;
+----------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
| TABLE    | INSERT | INSROW | INSTIM | UPDATE | UPDROW | UPDTIM | DELETE | DELROW | DELTIM | SELECT | SELROW | SELTIM |
+----------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
| my_range |      0 |      0 |      0 |      0 |      0 |      0 |      0 |      0 |      0 |  42682 |  42682 | 495841 |
+----------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
1 row in set (0.00 sec)

mysql> list userstats;
+-------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
| TABLE | INSERT | INSROW | INSTIM | UPDATE | UPDROW | UPDTIM | DELETE | DELROW | DELTIM | SELECT | SELROW | SELTIM |
+-------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
| test  |      0 |      0 |      0 |      0 |      0 |      0 |      0 |      0 |      0 |  42682 |  42682 | 495841 |
+-------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
1 row in set (0.00 sec)
```

高可用

- 启用多机热备
- 指定VIP地址
  - 选项：--vip-address=ip addr/netdev:0
  - 服务器1：--vip-address=192.168.1.120/eth0:0
  - 服务器2：--vip-address=192.168.1.120/em0:0
  - 服务器3：--vip-address=192.168.1.120/en:0

客户案例

- 马展金融（读写分离）
- 新浪手机微博（读写分离）
- 某国家保密单位（分库分表、并发查询）
- 贝贝网（读写分离）
- 金微蓝/金证股份（分库分表）
- 河狸家（读写分离）
- ……