Arkproxy -MySQL 中间件系统

公司简介

北京极数云舟科技有限公司创建于 2017 年，创始团队由数名数据库相关领域专家组成，凭借在数据库方面多年的运维管理与内核源码研究经验，提供 MySQL、Redis、HBase 等系列全套平台解决方案、技术产品、培训咨询和专业的技术支持服务。

极数云舟的创始人周彦伟是活跃在 MySQL 领域的知名专家，不仅是中国 MySQL 用户组（ACMUG）的联合创始人和现任主席，也是 Oracle 官方授予的 MySQL 方向的 ACE Director，这在全世界也仅仅只有六个人。

极数云舟的核心技术团队在百度、微博、小米、去哪儿网、人人网、陌陌、达梦数据库、华为等公司的工作经验以及他们浙江大学、北京大学、武汉大学、华中科技大学、北京航空航天大学、北京交通大学、东北大学的求学经历是整个公司核心竞争力和技术实力的体现。

极数云舟依托自身团队长期一线互联网公司的工作经验和开源社区的广泛积累，积极探索，勇敢挑战，持续创新，致力于企业级开源数据库解决方案和自动化运维体系建设。

Arkproxy 部署

1.环境说明

两台服务器:

1. 搭建一主两从复制环境。
2. 部署 Arkproxy,可以共用某台服务器进行部署。

2.相关资源包

3.MySQL 部署

搭建 MySQL 主从复制。

角色	主机名	IP:Port
Master	arkdb001	10.0.0.145
Slave1	arkdb002	10.0.0.146
Slave2	arkdb003	10.0.0.147
Arkproxy	arkdb001	10.0.0.145

3.1 搭建一主两从 MySQL 集群 。

4.Arkproxy 部署

4.1 从极数云舟获取 arkproxy 安装包。

4.2 规范化部署目录,统一安装在/data/ark/arkproxy 下,使用 10.0.0.145 进行 Arkproxy 部署。

```
mkdir -p /data/ark/arkproxy
cd /data/ark/arkproxy
```

```
mkdir -p bin    conf      log
```

上传安装包文件 arkproxy,路径信息如下:

```
/data/ark/arkproxy/bin
```

4.3 在目录/data/ark/arkproxy/conf 新建配置文件 arkproxy.cnf

基本配置文件 arkproxy.cnf 如下:

```
##代表 ArkProxy 的配置，具体每个参数的意义，请参考后面说明
[arkproxy]
proxy_backend_passwd            = arkproxy
proxy_backend_user              = arkproxy
proxy_port                      = 3336
proxy_check_slave_lag_period    = 10
proxy_config_host               =
proxy_config_passwd             =
proxy_config_port               = 0
proxy_config_user               =
proxy_digest_trace              = ON
proxy_flush_thread_num          = 5
proxy_format_fullpath           = OFF
proxy_format_num_per_time       = 10
proxy_format_parameterize       = ON
proxy_format_thread_num         = 5
proxy_general_log               = ON
proxy_general_log_file          = /data/ark/arkproxy/log/proxy.log
proxy_license_key_path          = /data/ark/arkproxy/key/arkproxy.key
proxy_log_error                 = /data/ark/arkproxy/log/error.log
proxy_multi_write_mode          = ON
proxy_namespace                 =
proxy_non_encrypted_ips         = 127.0.0.1
proxy_recv_queue_size           = 1000
proxy_send_queue_size           = 10
proxy_server_address            =    %
```

```
proxy_server_heartbeat_period     = 100
proxy_shell_listen_port           = 3335
proxy_shell_password              = password
proxy_shell_username              = proxyshell
proxy_sql_trace                   = ON
proxy_sql_trace_thread_num        = 5
proxy_trace_busy_strategy         = 0
proxy_trace_busy_wait_time        = 500
proxy_trace_client_queue_size     = 1000
proxy_trace_flush_thread_sleep_msec = 500
proxy_trace_format_thread_sleep_msec = 500
proxy_trace_hash_mem_length       = 134217728
proxy_trace_hash_size             = 200
proxy_trace_max_delay_time        = 3
proxy_trace_queue_array_length    = 500
proxy_trace_sql_bucket_length     = 100
proxy_username_encypted           = OFF

#配置数据库服务
[server1]
#proxy 类型，表示当前节点类型为 Server
proxy_type=server
#数据库地址
backend_host=10.0.0.145
#数据库端口
backend_port=3306
#路由权重
weight=40
#当前节点，如果延迟达到这个值，就会停止前其路由请求
max_slave_lag= 200
#节点默认状态，中间层启动之后就是这个状态。
server_status = ONLINE
```

```
#对这个节点的注释
config_comment = day day up_server1

#同上
[server2]
proxy_type=server
backend_host=10.0.0.146
backend_port=3306
weight=30
config_comment = day day up_server2

[server3]
proxy_type=server
backend_host=10.0.0.147
backend_port=3306
weight=30
config_comment = day day up_server3



#路由设置
[router1]
#proxy 类型，当前配置节点为路由节点
proxy_type=router
#路由类型：readwrite，读写节点
router_type=readwrite
#配置路由 db
router_servers=server1
#路由注释
config_comment = xxxxxxxxx

[router2]
proxy_type=router
```

```
#只读路由
router_type=readonly
#有多个路由目标时，以逗号分隔，写在一起。
router_servers=server1,server2,server3
```

4.4 数据库集群写节点进行用户授权,proxy_shell_username && proxy_shell_password 只需要在配置文件中填写用户和密码就好，配置完成后直接访问管理端口。

中间层 IP 授权:

```
GRANT ALL PRIVILEGES ON *.* TO 'arkproxy'@'127.0.0.1' IDENTIFIED BY
'arkproxy'  WITH GRANT OPTION ;
GRANT ALL PRIVILEGES ON *.* TO 'arkproxy'@'%' IDENTIFIED BY 'arkproxy'
WITH GRANT OPTION;
GRANT ALL PRIVILEGES ON *.* TO 'sentinel_username'@'%' IDENTIFIED BY
'sentinel_password';
```

4.5 启动 arkproxy

```
/data/ark/arkproxy/bin/arkproxy  --defaults-
file=/data/ark/arkproxy/conf/arkproxy.cnf &
```

4.6. 登录 arkproxy 业务端口进行业务 IP 授权，业务账号统一到中间层管理端口进行授权。

```
mysql -uarkproxy -parkproxy -P3336 -h10.0.0.145    -A
```

业务端口说明

ArkProxy 业务访问管理登录(DEV/应用程序):

```
  mysql -uarkproxy -parkproxy -P3336 -h10.0.0.145   -A
```

ArkProxy shell 管理登录(DBA/管理员):

```
  mysql -uproxyshell -ppassword -P3335 -h10.0.0.145 -A
```

Arkproxy 功能测试

为了方便理解功能测试的效果，人工模拟主从数据不一致的场景。

模拟步骤如下:

1. 登录 Master，创建库表和数据

```
create database arkproxy_test;
create table arkproxy_test.test(id int not null primary key
AUTO_INCREMENT,name varchar(100), comment varchar(100));
insert into arkproxy_test.test values(1,'Master','Master');
```

2.登录 Slave1,插入如下数据

```
SET sql_log_bin =OFF;
insert into arkproxy_test.test values(2,'Slave1','Slave1');
SET sql_log_bin =ON;
```

3.登录 Slave2,插入如下数据

```
SET sql_log_bin =OFF;
insert into arkproxy_test.test values(3,'Slave2','Slave2');
SET sql_log_bin =ON;
```

通过查询 select * from arkproxy_test.test; 可以快速辨别出是哪个数据节点。

Master 节点数据如下:

```
mysql> select * from arkproxy_test.test;
+----+--------+---------+
| id | name   | comment |
+----+--------+---------+
|  1 | Master | Master  |
+----+--------+---------+
1 row in set (0.00 sec)
```

Slave1 节点数据如下:

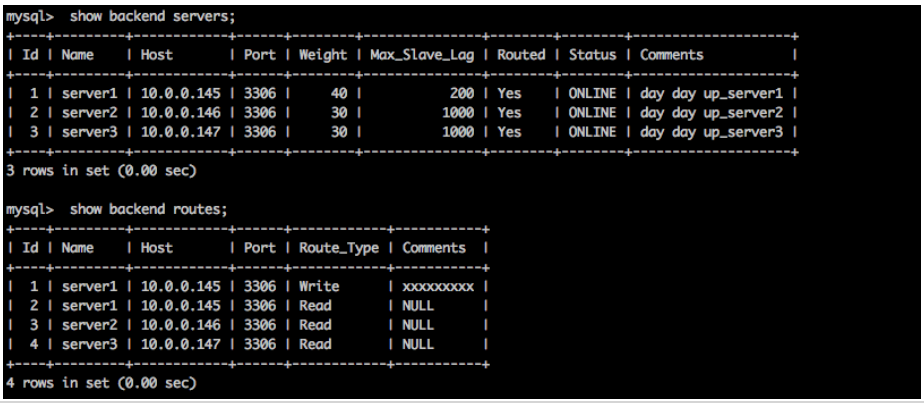
```
mysql>   select * from arkproxy_test.test;
+----+--------+---------+
| id | name   | comment |
+----+--------+---------+
|  1 | Master | Master  |
|  2 | Slave1 | Slave1  |
+----+--------+---------+
2 rows in set (0.00 sec)
```

Slave2 节点数据如下:

```
mysql> select * from arkproxy_test.test;
+----+--------+---------+
| id | name   | comment |
+----+--------+---------+
|  1 | Master | Master  |
|  3 | Slave2 | Slave2  |
+----+--------+---------+
2 rows in set (0.00 sec)
```

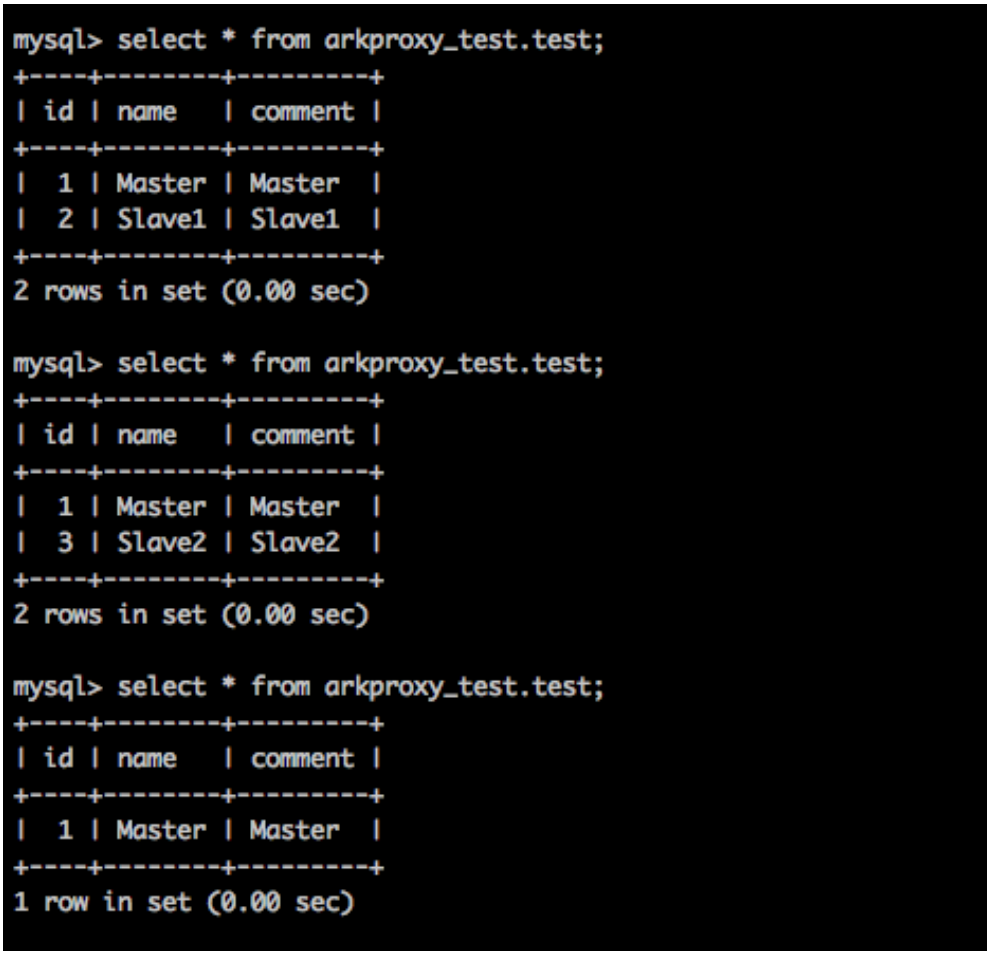
1.Arkproxy 后台管理

功能名称:	Arkproxy 后台管理和运维命令
预期结果	正常登录Arkproxy管理后台和运维命令基本使用

操作步骤	1. 登录Arkproxy管理端口，通常方便于DBA/运维人员使用。 mysql -uproxyshell -ppassword -P3335 -h127.0.0.1 -A 2. 查看支持的命令，执行 mysql> config help; 3. 查看后端配置数据库的状态,执行 mysql> show backend servers; 查看后台server对应route的状态,执行 mysql> show backend routes;  mysql> show backend servers; +----+---------+-------------+------+--------+---------------+--------+--------+--------------------+ | Id | Name | Host | Port | Weight | Max_Slave_Lag | Routed | Status | Comments | +----+---------+-------------+------+--------+---------------+--------+--------+--------------------+ | 1 | server1 | 10.0.0.145 | 3306 | 40 | 200 | Yes | ONLINE | day day up_server1 | | 2 | server2 | 10.0.0.146 | 3306 | 30 | 1000 | Yes | ONLINE | day day up_server2 | | 3 | server3 | 10.0.0.147 | 3306 | 30 | 1000 | Yes | ONLINE | day day up_server3 | +----+---------+-------------+------+--------+---------------+--------+--------+--------------------+ 3 rows in set (0.00 sec) mysql> show backend routes; +----+---------+-------------+------+------------+-----------+ | Id | Name | Host | Port | Route_Type | Comments | +----+---------+-------------+------+------------+-----------+ | 1 | server1 | 10.0.0.145 | 3306 | Write | xxxxxxxxx | | 2 | server1 | 10.0.0.145 | 3306 | Read | NULL | | 3 | server2 | 10.0.0.146 | 3306 | Read | NULL | | 4 | server3 | 10.0.0.147 | 3306 | Read | NULL | +----+---------+-------------+------+------------+-----------+ 4 rows in set (0.00 sec)
操作结果	符合预期

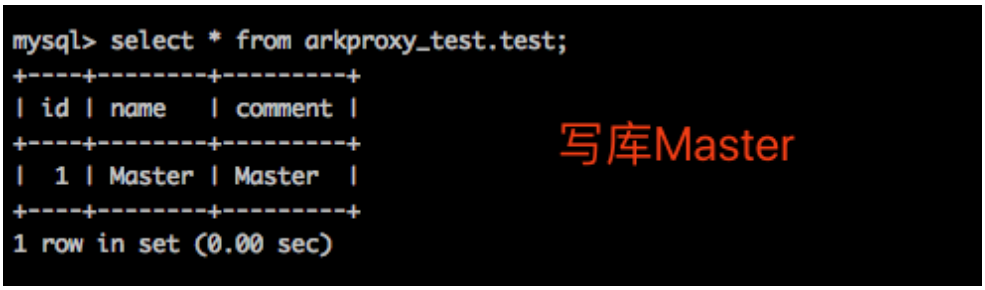
2.Arkproxy 读写分离

功能名称:	Arkproxy Hint 分发
预期结果	访问Arkproxy，SQL读请求会分发到读写节点。
操作步骤	1. 登录 Arkproxy 管理端口(mysql -h127.0.0.1 -uarkproxy -parkproxy -P3336) 2. 查询 select * from arkproxy_test.test; 会路由到Master、Slave1、Slave2。

	
操作结果	符合预期

3.Arkproxy 负载均衡

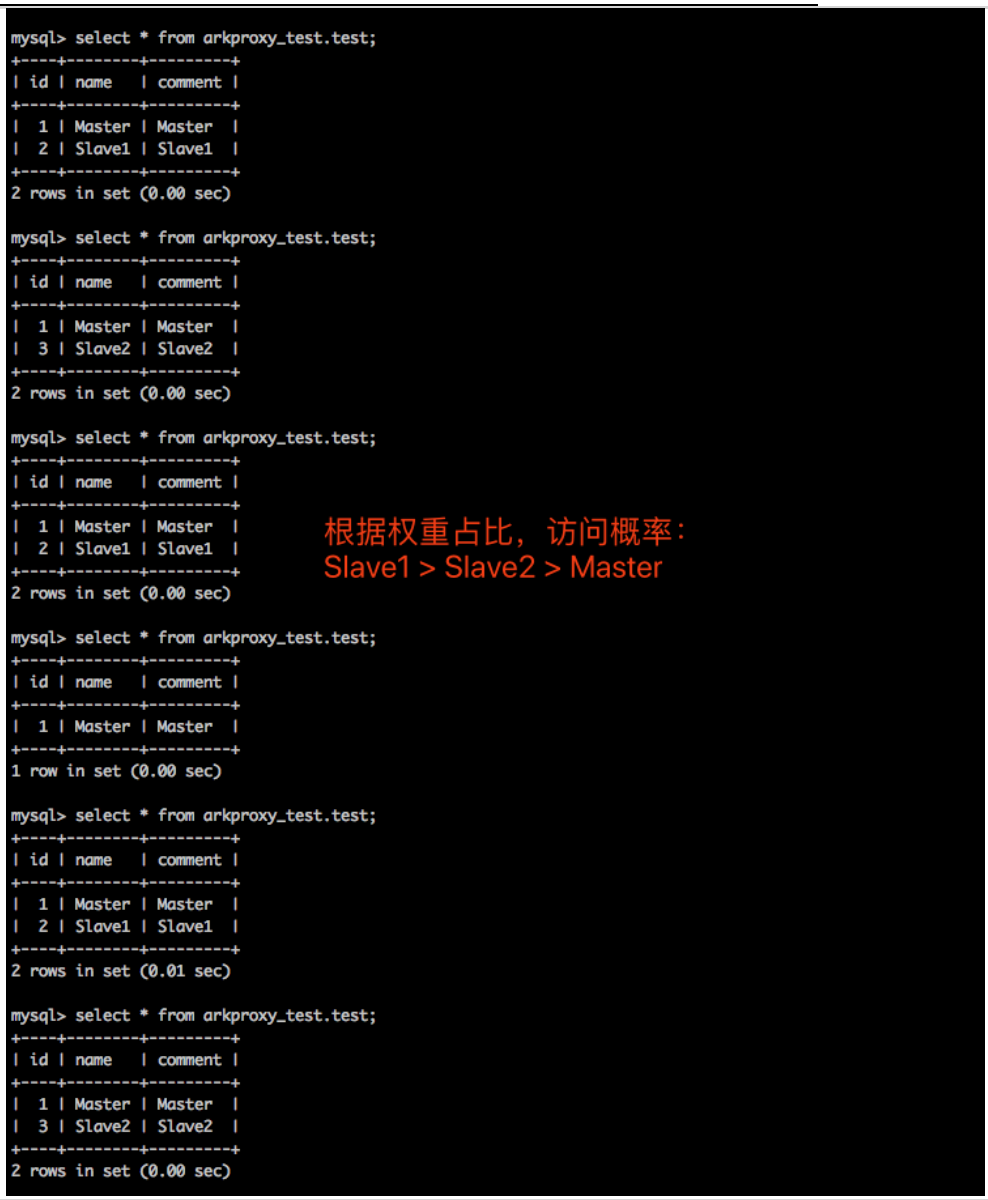
功能名称:	Arkproxy 负载均衡测试
预期结果	proxy会按照配置权重分配读SQL

操作步骤	1.准备数据，让读库，写库数据不一致，方便区分读写库    2.修改配置文件/data/ark/arkproxy/conf/arkproxy.cnf 设置节点权重

```
#配置数据库服务
[server1]
#proxy类型，表示当前节点类型为Server
proxy_type=server
#数据库地址
backend_host=10.0.0.145
#数据库端口
backend_port=3306
#路由权重
Weight=10
#当前节点，如果延迟达到这个值，就会停止前其路由请求
max_slave_lag= 200
#节点默认状态，中间层启动之后就是这个状态。
server_status = ONLINE
#对这个节点的注释
config_comment = day day up_server1

#同上
[server2]
proxy_type=server
backend_host=10.0.0.146
backend_port=3306
Weight=60
config_comment = day day up_server2

[server3]
proxy_type=server
backend_host=10.0.0.147
backend_port=3306
Weight=30
config_comment = day day up_server3


#路由设置
[router1]
#proxy类型，当前配置节点为路由节点
proxy_type=router
#路由类型：readwrite，读写节点
router_type=readwrite
#配置路由db
router_servers=server1
#路由注释
config_comment = xxxxxxxxx

[router2]
proxy_type=router
#只读路由
router_type=readonly
#有多个路由目标时，以逗号分隔，写在一起。
router_servers=server1,server2,server3
```

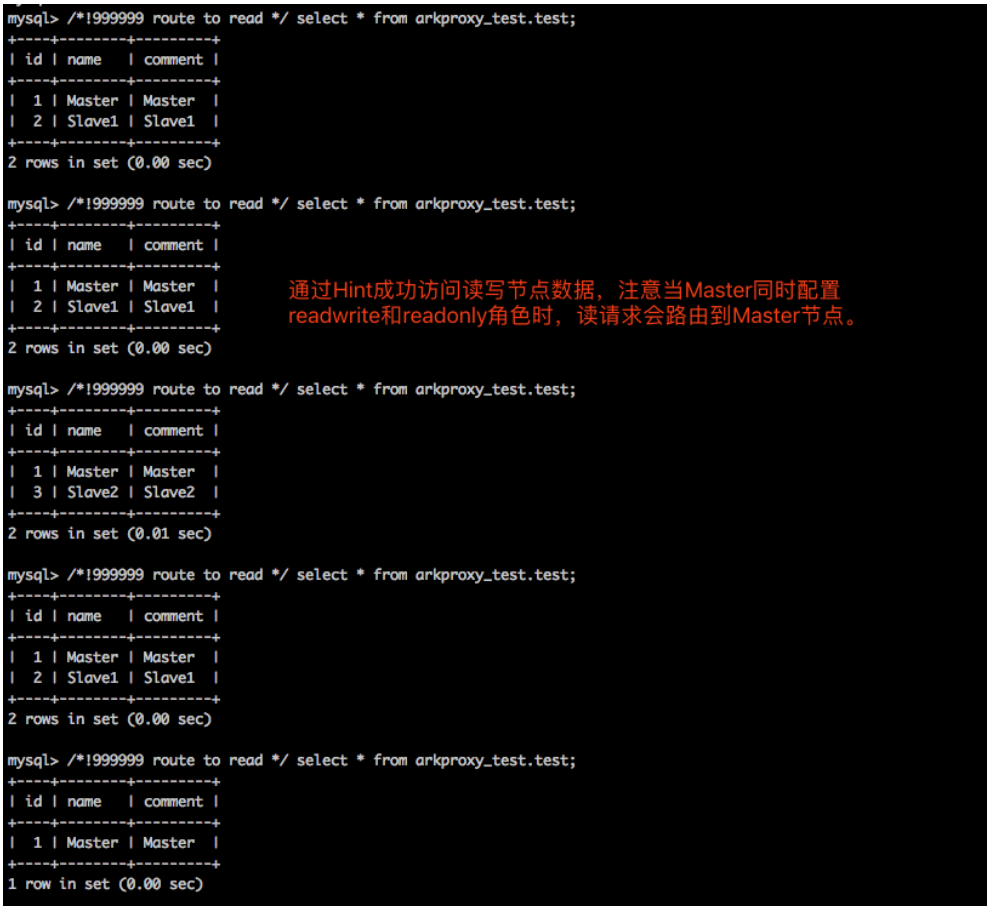
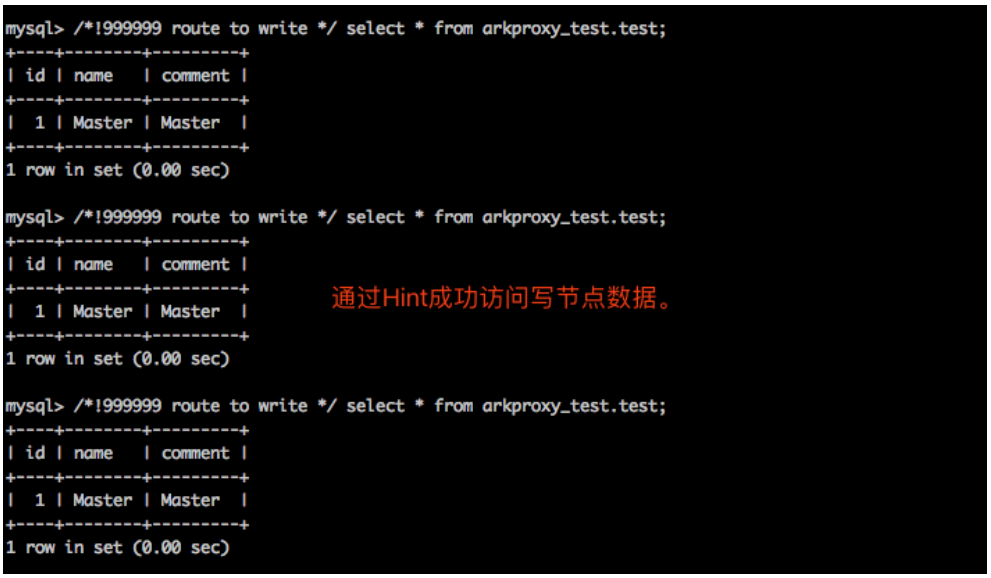
权重越大，分配到此节点概率越大

3. 重启arkproxy以生效配置文件，登陆Arkproxy (mysql -h127.0.0.1 -uarkproxy -parkproxy -P3336)，查看数据，确定是否按照配置会均衡分发

	
操作结果	符合预期

4.Arkproxy Hint 分发

功能名称:	Arkproxy Hint 分发
预期结果	按照hint分发到读写节点

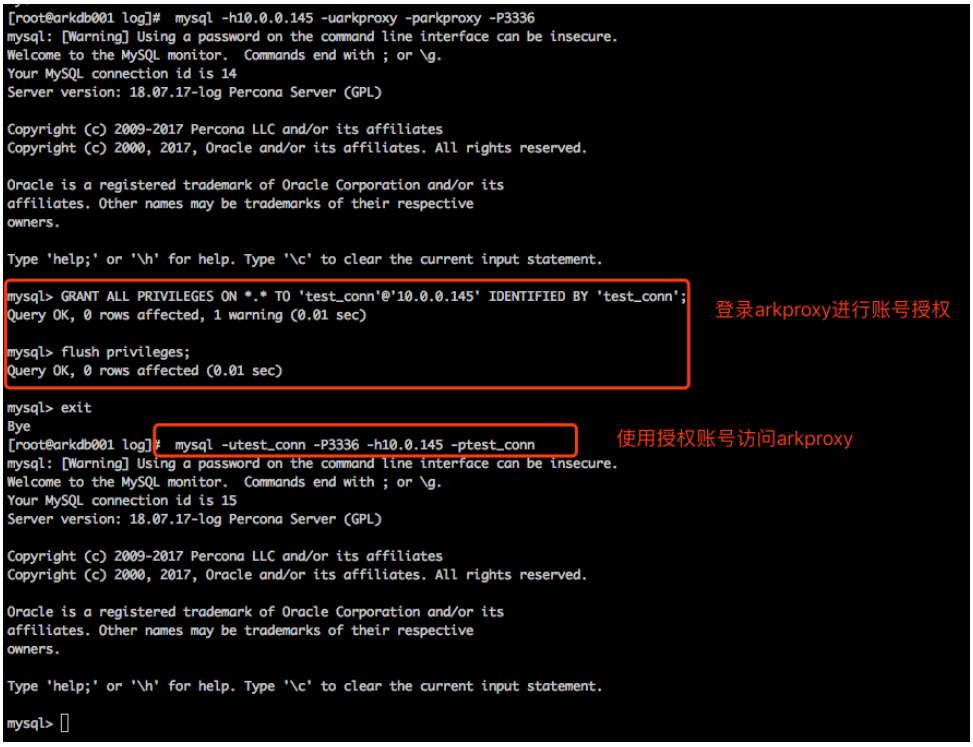
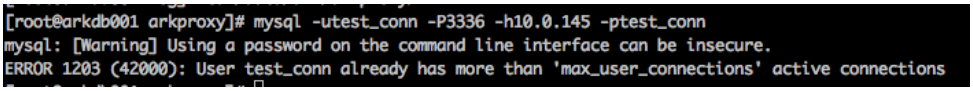
操作步骤	1. hint 分配到读节点，多次测试，是否一致  2. hint 到写节点，多次测试 
操作结果	符合预期

5.Arkproxy shell

功能名称:	Arkproxy shell
预期结果	在管理用户界面（arkdb1），可以获取详细的帮助信息
操作步骤	1. 登陆 Arkproxy shell，进入管理库(arkdb1) mysql -h127.0.0.1 -uproxyshell -ppassword -P3335 2. 输入 config help\G；查看帮助信息 
操作结果	符合预期

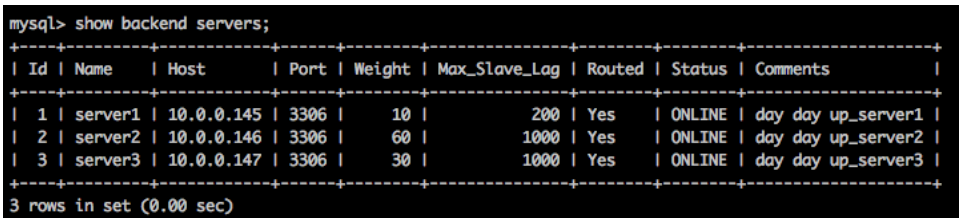
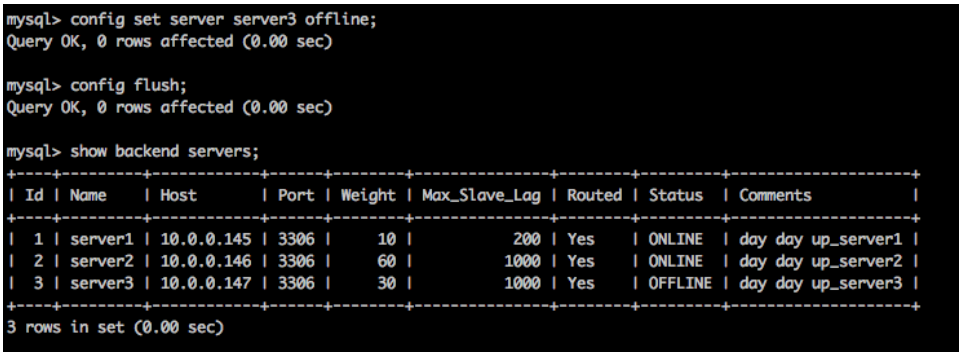
6.Arkproxy 用户连接数限制

功能名称:	Arkproxy 用户连接数限制
预期结果	超过连接数限制直接报错
操作步骤	1. 登录 arkproxy 进行账号授权: GRANT ALL PRIVILEGES ON *.* TO 'test_conn'@'10.0.0.145' IDENTIFIED BY 'test_conn'; flush privileges;

	 2. 在 arkproxy 部署机器上通过 Arkproxy shell 登陆管理配置窗口 mysql -h127.0.0.1 -uproxyshell -ppassword -P3335 3. 设置最大连接数为 0  4. 再次尝试登陆 Arkproxy 报错。 
操作结果	符合预期

7.Arkproxy 手动下线读

功能名称:	Arkproxy 手动下线读节点
预期结果	下线后不再分发SQL到下线实例

操作步骤	1.在arkdb01上通过Arkproxy shell登陆管理配置界面 mysql -h127.0.0.1 -uproxyshell -ppassword -P3335 查看机器情况 mysql> show backend servers;  2.. 下线server3  3.登录arkproxy查询数据(mysql -h127.0.0.1 -uarkproxy -parkproxy -P3336),查看是否会有server3的数据

```
mysql> show backend servers;
+----+---------+-----------+------+--------+--------------+--------+--------+-------------------+
| Id | Name    | Host      | Port | Weight | Max_Slave_Lag | Routed | Status | Comments          |
+----+---------+-----------+------+--------+--------------+--------+--------+-------------------+
|  1 | server1 | 10.0.0.145 | 3306 |     10 |          200 | Yes    | ONLINE | day day up_server1 |
|  2 | server2 | 10.0.0.146 | 3306 |     60 |         1000 | Yes    | ONLINE | day day up_server2 |
|  3 | server3 | 10.0.0.147 | 3306 |     30 |         1000 | Yes    | ONLINE | day day up_server3 |
+----+---------+-----------+------+--------+--------------+--------+--------+-------------------+
3 rows in set (0.00 sec)
```

```
mysql> config set server server3 offline;
Query OK, 0 rows affected (0.00 sec)

mysql> config flush;
Query OK, 0 rows affected (0.00 sec)

mysql> show backend servers;
+----+---------+-----------+------+--------+--------------+--------+---------+-------------------+
| Id | Name    | Host      | Port | Weight | Max_Slave_Lag | Routed | Status  | Comments          |
+----+---------+-----------+------+--------+--------------+--------+---------+-------------------+
|  1 | server1 | 10.0.0.145 | 3306 |     10 |          200 | Yes    | ONLINE  | day day up_server1 |
|  2 | server2 | 10.0.0.146 | 3306 |     60 |         1000 | Yes    | ONLINE  | day day up_server2 |
|  3 | server3 | 10.0.0.147 | 3306 |     30 |         1000 | Yes    | OFFLINE | day day up_server3 |
+----+---------+-----------+------+--------+--------------+--------+---------+-------------------+
3 rows in set (0.00 sec)
```

	
操作结果	符合预期

8.Arkproxy SQL 兼容

功能名称:	Arkproxy SQL兼容性
预期结果	Arkproxy 兼容日常使用的SQL
操作步骤	1. 登陆 Arkproxy，执行SQL测试 mysql -h127.0.0.1 -uarkproxy -parkproxy -P3336

建库,建表测试

```
mysql> create database arkproxy_test;
Query OK, 1 row affected (0.00 sec)

mysql> create database arkproxy_test2 charset=utf8;
Query OK, 1 row affected (0.00 sec)

mysql> use arkproxy_test
Database changed
mysql> create table demo(id int NOT NULL AUTO_INCREMENT primary key, content varchar(30));
Query OK, 0 rows affected (0.05 sec)

mysql> create table demo1(id int NOT NULL , content varchar(30) character set latin1) engine=myisam, charset=utf8
;
Query OK, 0 rows affected (0.01 sec)

mysql> create TEMPORARY table tempory_table2 like demo1;
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CREATE TABLE `partition_table` (
    ->   `id` int(11) NOT NULL AUTO_INCREMENT,
    ->   `content` varchar(30) NOT NULL DEFAULT '',
    ->   `val1` int(11) DEFAULT NULL,
    ->   `update_time` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    ->   `create_time` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    ->   `timecol` timestamp(3) NULL DEFAULT '2014-09-08 17:51:04.777',
    ->   `order_day` datetime NOT NULL,
    ->   `user_name` varchar(20) DEFAULT NULL,
    ->   `user_mobile` varchar(11) DEFAULT NULL,
    ->   PRIMARY KEY (`id`,`order_day`),
    ->   UNIQUE KEY `uniq_content_orderday` (`content`,`order_day`),
    ->   KEY `idx_val1` (`val1`),
    ->   KEY `idx_name_mobile` (`user_name`,`user_mobile`)
    -> ) ENGINE=InnoDB AUTO_INCREMENT=23 DEFAULT CHARSET=utf8mb4
    -> /*!50100 PARTITION BY RANGE (YEAR(order_day))
    -> (PARTITION p_2010 VALUES LESS THAN (2010) ENGINE = InnoDB,
    ->  PARTITION p_2011 VALUES LESS THAN (2011) ENGINE = InnoDB,
    ->  PARTITION p_2012 VALUES LESS THAN (2012) ENGINE = InnoDB,
    ->  PARTITION p_catchall VALUES LESS THAN MAXVALUE ENGINE = InnoDB) */;
Query OK, 0 rows affected (0.06 sec)
```

```
mysql> CREATE TABLE `ts` (
    ->   `id` int(11) DEFAULT NULL,
    ->   `purchased` date DEFAULT NULL
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
    -> /*!50100 PARTITION BY RANGE ( YEAR(purchased))
    -> SUBPARTITION BY HASH ( TO_DAYS(purchased))
    -> (PARTITION p0 VALUES LESS THAN (1990)
    ->  (SUBPARTITION s0 ENGINE = InnoDB,
    ->   SUBPARTITION s1 ENGINE = InnoDB),
    ->  PARTITION p1 VALUES LESS THAN (2000)
    ->  (SUBPARTITION s2 ENGINE = InnoDB,
    ->   SUBPARTITION s3 ENGINE = InnoDB),
    ->  PARTITION p2 VALUES LESS THAN MAXVALUE
    ->  (SUBPARTITION s4 ENGINE = InnoDB,
    ->   SUBPARTITION s5 ENGINE = InnoDB)) */;
Query OK, 0 rows affected (0.03 sec)
```

DML测试

```
mysql> insert into demo values(null,'insert test1');
Query OK, 1 row affected (0.00 sec)

mysql> insert into demo values(null,'insert test2');
Query OK, 1 row affected (0.01 sec)

mysql> insert into demo values(null,'insert test3');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> delete from demo where content = 'insert test1';
Query OK, 1 row affected (0.00 sec)

mysql> update demo set content='update test' where content = 'insert test2';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from demo;
+----+-------------+
| id | content     |
+----+-------------+
|  4 | update test |
|  7 | insert test3 |
+----+-------------+
2 rows in set (0.01 sec)
```

DDL测试

```
mysql> alter table demo add column val1 int not null;
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

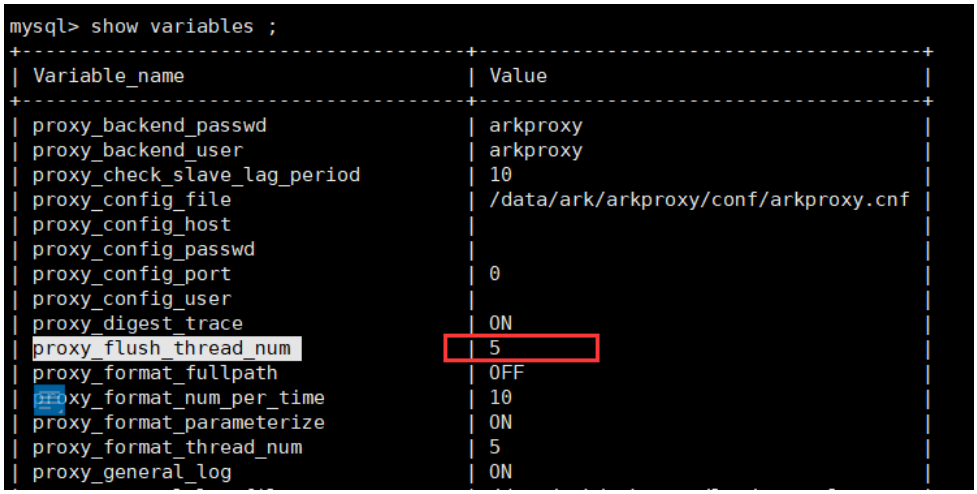
1. 直接登陆后端数据库3306端口查看数据(只截取部分)

[root@arkdb01 ~]# mysql -h127.0.0.1 -uroot -P3306

```
mysql> show tables;
+---------------------------+
| Tables_in_arkproxy_test |
+---------------------------+
| demo                      |
| demo1                     |
| partition_table           |
| ts                        |
+---------------------------+
4 rows in set (0.00 sec)

mysql> select * from demo;
+----+--------------+------+
| id | content      | val1 |
+----+--------------+------+
|  4 | update test  |    0 |
|  7 | insert test3 |    0 |
+----+--------------+------+
```

操作结果	符合预期

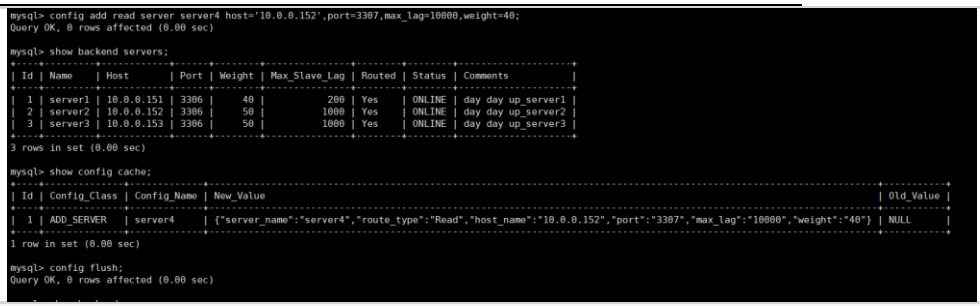
9.配置文件动态加载

功能名称:	Config reload
预期结果	修改配置文件后，可以热加载生效。
操作步骤	1. 登陆 Arkproxy shell，进入管理库(arkdb1) mysql -h127.0.0.1 -uproxyshell -ppassword -P3335 show variables 查看当前系统参数详情  2. 修改配置文件，将原proxy_flush_thread_num改为10 3. 登录Arkproxy shell，进入管理端口，config reload后查看参数变量，生效。
操作结果	符合预期

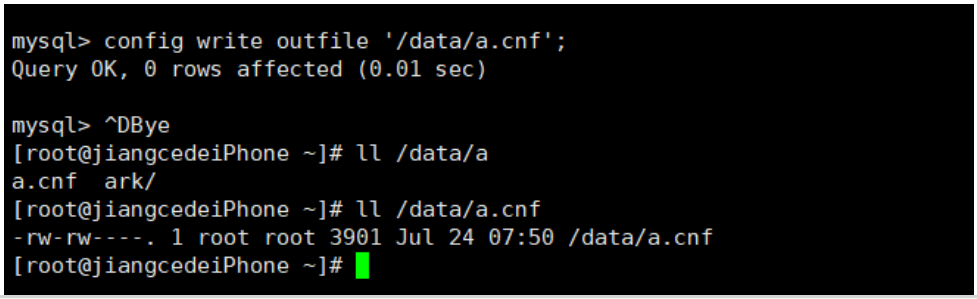
```
mysql> show variables ;
+-------------------------------------+-------------------------------------+
| Variable_name                       | Value                               |
+-------------------------------------+-------------------------------------+
| proxy_backend_passwd                | arkproxy                            |
| proxy_backend_user                  | arkproxy                            |
| proxy_check_slave_lag_period        | 10                                  |
| proxy_config_file                   | /data/ark/arkproxy/conf/arkproxy.cnf |
| proxy_config_host                   |                                     |
| proxy_config_passwd                 |                                     |
| proxy_config_port                   | 0                                   |
| proxy_config_user                   |                                     |
| proxy_digest_trace                  | ON                                  |
| proxy_flush_thread_num              | 5                                   |
| proxy_format_fullpath               | OFF                                 |
| proxy_format_num_per_time           | 10                                  |
| proxy_format_parameterize           | ON                                  |
| proxy_format_thread_num             | 5                                   |
| proxy_general_log                   | ON                                  |
```

10.配置项加载

功能名称:	Config flush
预期结果	在线修改某个配置项之后，动态加载生效。
操作步骤	1. 登陆 Arkproxy shell，进入管理库(arkdb1) mysql -h127.0.0.1 -uproxyshell -ppassword -P3335 2. 修改某个配置项，例如在线增加server，config add read server server4 host='10.0.0.152,port=3307,max_lag=10000,weight=40; 3. 增加之后查看server信息为生效，配置在cache中，需要config flush;

	
操作结果	符合预期

11.配置导出

功能名称:	当前配置导出到外部文件
预期结果	执行完成之后，当前配置导出到外部文件
操作步骤	1. 登陆 Arkproxy shell，进入管理库(arkdb1) mysql -h127.0.0.1 -uproxyshell -ppassword -P3335 2. 执行config write outfile '/data/a.txt';即可导出。如果不指定绝对路径导出到arkproxy的bin目录下 
操作结果	符合预期

12.配置删除

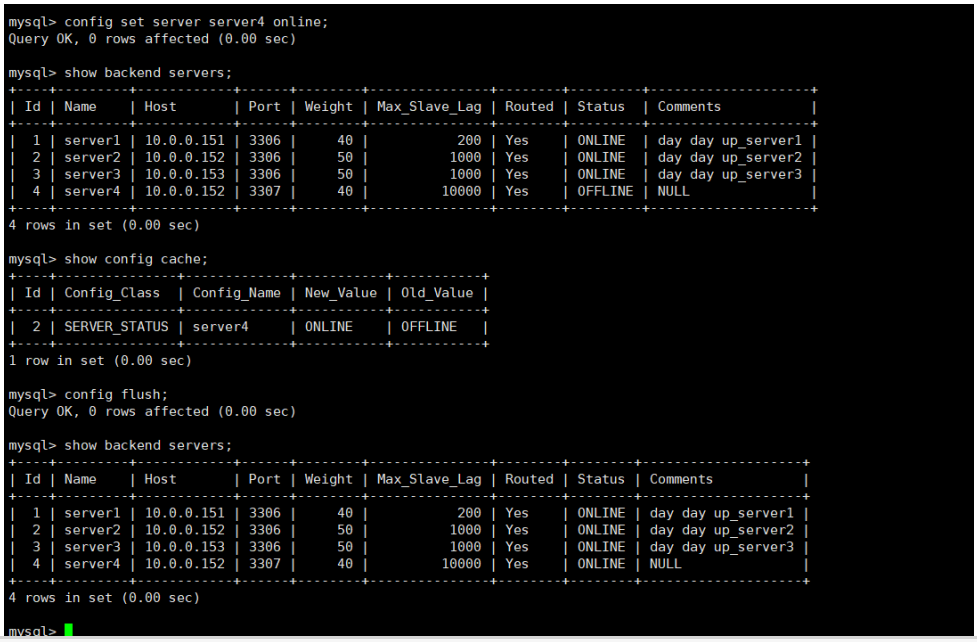
功能名称:	删除某个配置项
预期结果	修改配置之后，删除配置项。
操作步骤	1. 登陆 Arkproxy shell，进入管理库(arkdb1) mysql -h127.0.0.1 -uproxyshell -ppassword -P3335 2. 在线调整server上线，执行config set server server4 online后，show config cache 查看到如下信息。

	mysql> show config cache; +----+---------------+-------------+-----------+-----------+ | Id | Config_Class | Config_Name | New_Value | Old_Value | +----+---------------+-------------+-----------+-----------+ | 1 | SERVER_STATUS | server4 | ONLINE | OFFLINE | +----+---------------+-------------+-----------+-----------+ 1 row in set (0.00 sec) 3.config delete 1即可删除之前操作。 mysql> config delete 1; Query OK, 0 rows affected (0.00 sec) mysql> show config cache; Empty set (0.00 sec)
操作结果	符合预期

13.在线增加 server

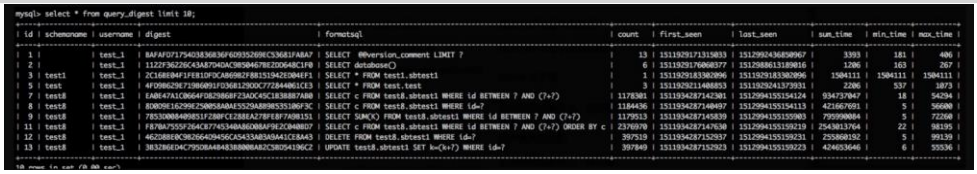
功能名称:	在线增加server
预期结果	操作完成后server可以查看正常
操作步骤	1. 登陆 Arkproxy shell，进入管理库(arkdb1) mysql -h127.0.0.1 -uproxyshell -ppassword -P3335 2. 执行增加新server，然后刷新缓存设置生效。 mysql> config add read server server4 host='10.0.0.152',port=3307,max_lag=10000,weight=40; Query OK, 0 rows affected (0.00 sec) mysql> show backend servers; +----+---------+-----------+------+--------+---------------+--------+---------+-------------------+ | Id | Name | Host | Port | Weight | Max_Slave_Lag | Routed | Status | Comments | +----+---------+-----------+------+--------+---------------+--------+---------+-------------------+ | 1 | server1 | 10.0.0.151 | 3306 | 40 | 200 | Yes | ONLINE | day day up_server1 | | 2 | server2 | 10.0.0.152 | 3306 | 50 | 1000 | Yes | ONLINE | day day up_server2 | | 3 | server3 | 10.0.0.153 | 3306 | 50 | 1000 | Yes | ONLINE | day day up_server3 | +----+---------+-----------+------+--------+---------------+--------+---------+-------------------+ 3 rows in set (0.00 sec) mysql> show config cache; +----+--------------+-------------+-----------+-----------+ | Id | Config_Class | Config_Name | New_Value | Old_Value | +----+--------------+-------------+-----------+-----------+ | 1 | ADD_SERVER | server4 | {"server_name":"server4","route_type":"Read","host_name":"10.0.0.152","port":"3307","max_lag":"10000","weight":"40"} | NULL | +----+--------------+-------------+-----------+-----------+ 1 row in set (0.00 sec) mysql> config flush; Query OK, 0 rows affected (0.00 sec) mysql> show backend servers; +----+---------+-----------+------+--------+---------------+--------+---------+-------------------+ | Id | Name | Host | Port | Weight | Max_Slave_Lag | Routed | Status | Comments | +----+---------+-----------+------+--------+---------------+--------+---------+-------------------+ | 1 | server1 | 10.0.0.151 | 3306 | 40 | 200 | Yes | ONLINE | day day up_server1 | | 2 | server2 | 10.0.0.152 | 3306 | 50 | 1000 | Yes | ONLINE | day day up_server2 | | 3 | server3 | 10.0.0.153 | 3306 | 50 | 1000 | Yes | ONLINE | day day up_server3 | | 4 | server4 | 10.0.0.152 | 3307 | 40 | 10000 | Yes | OFFLINE | NULL | +----+---------+-----------+------+--------+---------------+--------+---------+-------------------+ 4 rows in set (0.00 sec)
操作结果	符合预期

14.在线上线 server

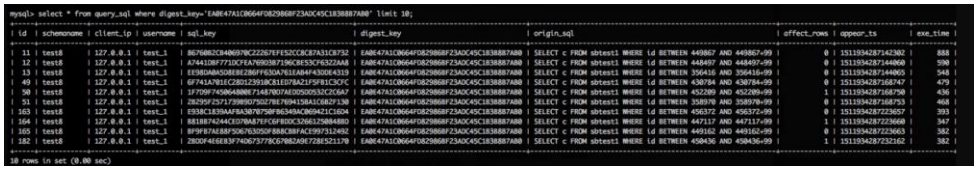
功能名称:	在线上线server
预期结果	将下线的server在线调整为上线
操作步骤	1. 登陆 Arkproxy shell，进入管理库(arkdb1) mysql -h127.0.0.1 -uproxyshell -ppassword -P3335 2. 执行增加新server，然后刷新缓存设置生效。查询数据后正常可以显示 
操作结果	符合预期

15. 智能统计

功能名称:	开启Trace进行智能统计
预期结果	统计SQL语句
操作步骤	1. 参数proxy_digest_trace & proxy_sql_trace 默认开启，相关配置表在本地mysql数据库中。 mysql> show tables; +---------------+ | Tables_in_tes | +---------------+ | proxy_user | | query_digest | | query_sql | +---------------+ digest的统计情况select * from query_digest;



如果要看具体的SQL，可以统一digest列，进行where过滤：



可以很清晰的看到，执行时的时间点，执行时间等。

2. 如果统计信息设置到远程数据库，可以配置参数到远程

```
proxy_config_host                    =
proxy_config_passwd                  =
proxy_config_port                    = 0
proxy_config_user                    =
```

统计之后的信息会被存储到proxy_config_host:proxy_config_port设置的目标数据库中，可能会对中间层的性能有点影响，但影响不大。

操作结果	符合预期