对 791 测试服 Mysql 性能优化方案

1 交流,不足,情况
2 测试服,调研
3 优化方案
4 拓展方案
*5 性能对比测试

张志远

物联网业务瓶颈:存储和数据库,向前推

CDN----MYBATIS—REDIS---CACHE---MYSQL

一　查看公司硬件

1 查看系统内核---内核比较新 linux-2.6.32

```
[jiuyi@jiuyi-test mysql]$ uname -a
Linux jiuyi-test 2.6.32-431.el6.x86_64 #1 SMP Fri Nov 22 03:15:09 U
TC 2013 x86_64 x86_64 x86_64 GNU/Linux
```

2 查看系统信息—系统稳定 centos6.5.x_86_64

```
[jiuyi@jiuyi-test mysql]$ cat /proc/version
Linux version 2.6.32-431.el6.x86_64 (mockbuild@c6b8.bsys.dev.centos
.org) (gcc version 4.4.7 20120313 (Red Hat 4.4.7-4) (GCC) ) #1 SMP
Fri Nov 22 03:15:09 UTC 2013
```

3 查看 cpu 详情—性能够用 Interl-E3

```
bogomips        : 6186.08
clflush size    : 64
cache_alignment : 64
address sizes   : 36 bits physical, 48 bits virtual
power management:

processor       : 3
vendor_id       : GenuineIntel
cpu family      : 6
model           : 58
model name      : Intel(R) Xeon(R) CPU E3-1220 V2 @ 3.10GHz
stepping        : 9
cpu MHz         : 1600.000
cache size      : 8192 KB
physical id     : 0
siblings        : 4
core id         : 3
cpu cores       : 4
apicid          : 6
initial apicid  : 6
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
```

3.1 查看物理 cpu 个数—偏少 1 颗

```
[jiuyi@jiuyi-test mysql]$ cat /proc/cpuinfo| grep "physical id"| sort
| uniq| wc -l
1
```

3.2 查看每个物理 cpu 的核数—正常 4 核

```
[jiuyi@jiuyi-test mysql]$ cat /proc/cpuinfo| grep "cpu cores"| uniq
cpu cores       : 4
```

3.3 查看逻辑 cpu 个数—偏少 4 个

```
[jiuyi@jiuyi-test mysql]$ cat /proc/cpuinfo| grep "processor"| wc -l
4
```

3.4 cpu 利用率 3% CPU 上下文切换时并发的瓶颈,（案例:一个程序多线程和单线程，在一定程度上多线程的速度才比单线程快）

```
top - 16:34:36 up 7 days,  7:00,  3 users,  load average: 0.14, 0.07, 0.05
Tasks: 185 total,   1 running, 184 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.4%us,  0.1%sy,  0.0%ni, 98.8%id,  0.7%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   8030000k total,  6639624k used,  1390376k free,   304544k buffers
```

4 查看内存信息 –内存使用较多 总内存不足 8G，使用了 6.5G

```
[jiuyi@jiuyi-test mysql]$ free -m
             total       used       free     shared    buffers     cached
Mem:          7841       6309       1531          0        279       1181
-/+ buffers/cache:       4848       2992
Swap:         7983          0       7983
```

5 查看所有监听端口 netstate –lntp ------监听服务较多 一去就医服务所以服务

```
tcp        0      0 ::ffff:127.0.0.1:50105      :::*                        LISTEN
/java
tcp        0      0 :::18009                    :::*                        LISTEN
tcp        0      0 ::1:25                      :::*                        LISTEN
tcp        0      0 :::41465                    :::*                        LISTEN
tcp        0      0 ::ffff:127.0.0.1:50011      :::*                        LISTEN
tcp        0      0 :::50012                    :::*                        LISTEN
tcp        0      0 ::ffff:127.0.0.1:51005      :::*                        LISTEN
/java
tcp        0      0 :::51102                    :::*                        LISTEN
tcp        0      0 :::51103                    :::*                        LISTEN
/java
tcp        0      0 :::51104                    :::*                        LISTEN
/java
tcp        0      0 :::51106                    :::*                        LISTEN
tcp        0      0 :::51107                    :::*                        LISTEN
tcp        0      0 :::51046                    :::*                        LISTEN
/java
tcp        0      0 :::51110                    :::*                        LISTEN
```

6 查看磁盘使用情况 –情况良好 一块磁盘 还剩余%70

```
[jiuyi@jiuyi-test ~]$ df
Filesystem                      1K-blocks     Used Available Use% Mounted on
/dev/mapper/vg_jiuyitest-lv_root  51606140  7216476  41768224  15% /
tmpfs                              4015000       72   4014928   1% /dev/shm
/dev/sda2                           495844    39386    430858   9% /boot
/dev/sda1                           204580      272    204308   1% /boot/efi
/dev/mapper/vg_jiuyitest-lv_home 420358800 10627192 388378548   3% /home
```

7 查看磁盘的类型

二 查看公司数据库详情

1 查看数据库版本 –版本较低—5.1.73

```
[jiuyi@jiuyi-test ~]$ mysql --version
mysql  Ver 14.14 Distrib 5.1.73, for redhat-linux-gnu (x86_64) using readline 5.1
[jiuyi@jiuyi-test ~]$
```

2 查看数据库配置文件位置

```
[jiuyi@jiuyi-test ~]$ mysql --help | grep my.cnf
                      order of preference, my.cnf, $MYSQL_TCP_PORT,
/etc/mysql/my.cnf /etc/my.cnf ~/.my.cnf
[jiuyi@jiuyi-test ~]$
```

2.1 查看配置文件内容 有待优化，配置文件信息太少，未加缓存

```
port=51101
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
# Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links=0
default-character-set=utf8

[mysqld_safe]
skip-grant-tables
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
~
~
~
~
~
```

3 查看数据库支持引擎和默认引擎—默认引擎不支持事务

```
*************************** 3. row ***************************
      Engine: MyISAM
     Support: DEFAULT
     Comment: Default engine as of MySQL 3.23 with great performance
Transactions: NO
          XA: NO
  Savepoints: NO
*************************** 4. row ***************************
      Engine: InnoDB
     Support: YES
     Comment: Supports transactions, row-level locking, and foreign keys
Transactions: YES
          XA: YES
  Savepoints: YES
*************************** 5. row ***************************
      Engine: MEMORY
     Support: YES
     Comment: Hash based, stored in memory, useful for temporary tables
Transactions: NO
          XA: NO
  Savepoints: NO
5 rows in set (0.00 sec)
```

4 查看引擎 MyISAM

```
mysql>  show variables like '%storage_engine%';
+----------------+--------+
| Variable_name  | Value  |
+----------------+--------+
| storage_engine | MyISAM |
+----------------+--------+
1 row in set (0.00 sec)

mysql>
```

/**/

数据库引擎简介 MyIsam 和 InnoDB

	myisam	InnoDB
构成	每个MyISAM在磁盘上存储成三个文件。第一个文件的名字以表的名字开始，扩展名指出文件类型。 .frm文件存储表定义。 数据文件的扩展名为.MYD (MYData)。 索引文件的扩展名是.MYI (MYIndex)。	基于磁盘的资源是InnoDB表空间数据文件和它的日志文件，InnoDB 表的大小只受限于操作系统文件的大小，一般为 2GB
事务处理	MyISAM类型的表强调的是性能，其执行数度比InnoDB类型更快，但是不提供事务支持	InnoDB提供事务支持事务，外部键等高级数据库功能
select update insert delete	如果执行大量的SELECT，MyISAM是更好的选择	如果你的数据执行大量的INSERT或UPDATE，出于性能方面的考虑，应该使用InnoDB表
表的具体行数	select count(*) from table,MyISAM只要简单的读出保存好的行数，注意的是，当count(*)语句包含 where条件时，两种表的操作是一样的	InnoDB 中不保存表的具体行数，也就是说，执行select count(*) from table时，InnoDB要扫描一遍整个表来计算有多少行
锁	表锁	提供行锁(locking on row level)，提供与 Oracle 类型一致的不加锁读取(non-locking read in SELECTs)，另外，InnoDB表的行锁也不是绝对的，如果在执行一个SQL语句时MySQL不能确定要扫描的范围，InnoDB表同样会锁全表，例如update table set num=1 where name like “%aaa%”

MyIsam 缓存 key InnoDB 缓存 data

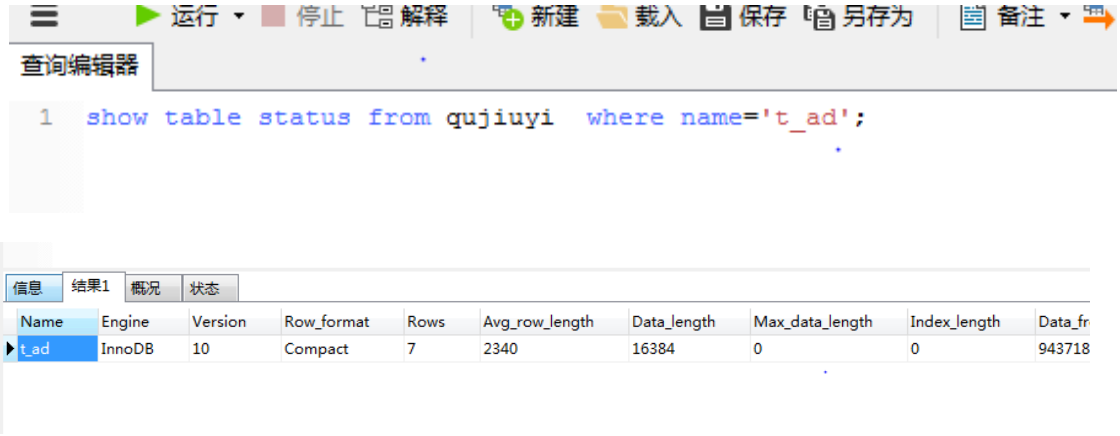
/**/

5 查看 powerdesiner 中创建数据库文件 sql –创建指定引擎

```
)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_bin;

alter table t_doctor_consult comment '医生会诊记录表';
```

6 查看数据库表的引擎



7 查看数据库的状态—未做缓存设置，命中率为 0，没有日志

show global status\G

```
       Value: 0
*************************** 189. row ***************************
Variable_name: Innodb_data_read
       Value: 5246976
*************************** 190. row ***************************
Variable_name: Innodb_data_reads
       Value: 188
*************************** 191. row ***************************
Variable_name: Innodb_data_writes
       Value: 11529
*************************** 192. row ***************************
Variable_name: Innodb_data_written
       Value: 94405632
*************************** 193. row ***************************
Variable_name: Innodb_dblwr_pages_written
       Value: 2684
*************************** 194. row ***************************
Variable_name: Innodb_dblwr_writes
```

Variable_name	Value
Innodb_buffer_pool_pages_data	203
Innodb_buffer_pool_pages_dirty	0
Innodb_buffer_pool_pages_flushed	6932
Innodb_buffer_pool_pages_free	304
Innodb_buffer_pool_pages_misc	5
Innodb_buffer_pool_pages_total	512
Innodb_buffer_pool_read_ahead_rnd	2
Innodb_buffer_pool_read_ahead_seq	0
Innodb_buffer_pool_read_requests	14049795
Innodb_buffer_pool_reads	181
Innodb_buffer_pool_wait_free	0
Innodb_buffer_pool_write_requests	121081

innodb_buffer_pool_size 缓冲池字节大小，InnoDB 缓存表和索引数据的内存区域

5.7 以前需要重启设置,大于 10G 启动在现代化服务器上需要 6S

有数据的 page 为 203，free page 为 304

buffer_pool 默认合理 使用率 39.648%

Buffer_pool 读取次数为 14049795 从磁盘中读取数据为 181

命中率为 (14049795-181)/ 14049795=99.999987%

```
mysql> SELECT @@innodb_buffer_pool_size
    -> ;
+---------------------------+
| @@innodb_buffer_pool_size |
+---------------------------+
|                   8388608 |
+---------------------------+
1 row in set (0.00 sec)
```

带下为 (8388608/1024)/1024=8M //该参数可以为内存的 1/2(不考虑其他服务)

```
mysql> show status like 'innodb_log%';
+----------------------------+-------+
| Variable_name              | Value |
+----------------------------+-------+
| Innodb_log_waits           | 0     |
| Innodb_log_write_requests  | 10858 |
| Innodb_log_writes          | 22111 |
+----------------------------+-------+
3 rows in set (0.00 sec)
```

```
mysql> select @@query_cache_type
    -> ;
+--------------------+
| @@query_cache_type |
+--------------------+
| ON                 |
+--------------------+
1 row in set (0.00 sec)

mysql> show variables like 'have_query_cache';
+------------------+-------+
| Variable_name    | Value |
+------------------+-------+
| have_query_cache | YES   |
+------------------+-------+
1 row in set (0.00 sec)

mysql> select @@global.query_cache_size;
+---------------------------+
| @@global.query_cache_size |
+---------------------------+
|                         0 |
+---------------------------+
1 row in set (0.00 sec)
```

```
mysql>  show global status like 'Qcache%'
    -> ;
+-------------------------+-------+
| Variable_name           | Value |
+-------------------------+-------+
| Qcache_free_blocks      | 0     |
| Qcache_free_memory      | 0     |
| Qcache_hits             | 0     |
| Qcache_inserts          | 0     |
| Qcache_lowmem_prunes    | 0     |
| Qcache_not_cached       | 0     |
| Qcache_queries_in_cache | 0     |
| Qcache_total_blocks     | 0     |
+-------------------------+-------+
8 rows in set (0.00 sec)

mysql>
```

查询缓存开启了可是没有设定值

```
*************************** 234. row ***************************
Variable_name: Qcache_free_memory
        Value: 0
*************************** 235. row ***************************
Variable_name: Qcache_hits
        Value: 0
*************************** 236. row ***************************
Variable_name: Qcache_inserts
        Value: 0
*************************** 237. row ***************************
```

8 查看数据文件路径

```
mysql> show variables like '%dir%';
+----------------------------------------+-----------------------------+
| Variable_name                          | Value                       |
+----------------------------------------+-----------------------------+
| basedir                                | /usr/                       |
| binlog_direct_non_transactional_updates | OFF                         |
| character_sets_dir                     | /usr/share/mysql/charsets/  |
| datadir                                | /var/lib/mysql/             |
| innodb_data_home_dir                   |                             |
| innodb_log_group_home_dir              | ./                          |
| innodb_max_dirty_pages_pct             | 90                          |
| plugin_dir                             | /usr/lib64/mysql/plugin     |
| slave_load_tmpdir                      | /tmp                        |
| tmpdir                                 | /tmp                        |
+----------------------------------------+-----------------------------+
10 rows in set (0.00 sec)
```

9 查看数据文件目录

```
[jiuyi@jiuyi-test mysql]$ ll
æ€»ç”¨é‡  45144
-rw-rw----. 1 mysql mysql 35651584 8æœˆ   4 09:13 ibdata1
-rw-rw----. 1 mysql mysql  5242880 8æœˆ   4 09:13 ib_logfile0
-rw-rw----. 1 mysql mysql  5242880 8æœˆ   3 07:49 ib_logfile1
drwx------. 2 mysql mysql     4096 2æœˆ  19 15:52 jeeshop
drwx------. 2 mysql mysql     4096 12æœˆ 29 2015 JiuYishop
drwx------. 2 mysql mysql     4096 11æœˆ 23 2015 mysql
srwxrwxrwx  1 mysql mysql        0 8æœˆ   3 13:59 mysql.sock
drwx------. 2 mysql mysql     4096 4æœˆ  20 14:37 nutzbook
drwx------. 2 mysql mysql     4096 4æœˆ  25 20:50 poker
drwx------. 2 mysql mysql    53248 8æœˆ   2 17:04 qujiuyi
drwx------. 2 mysql mysql     4096 11æœˆ 23 2015 test
drwx------  2 mysql mysql     4096 7æœˆ  27 14:03 tszy
drwx------. 2 mysql mysql     4096 4æœˆ  22 16:23 yao
```

10 查看日志开启情况—未开

```
mysql> show variables like '%log';
+--------------------------------+-------+
| Variable_name                  | Value |
+--------------------------------+-------+
| back_log                       | 50    |
| general_log                    | OFF   |
| innodb_locks_unsafe_for_binlog | OFF   |
| log                            | OFF   |
| relay_log                      |       |
| slow_query_log                 | OFF   |
| sync_binlog                    | 0     |
+--------------------------------+-------+
7 rows in set (0.00 sec)
```

慢查询日志未开

11 查看索引情况——索引情况良好

查询编辑器

```
1  show index from t_doctor
2
```

信息 | 结果1 | 概况 | 状态

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Pack
t_doctor	0	PRIMARY	1	id	A	19	(Null)	(Null
t_doctor	0	index_doctor_phone	1	phone	A	19	(Null)	(Null
t_doctor	1	index_doctor_hospital	1	hospitalId	A	19	(Null)	(Null
t_doctor	1	index_doctor_token	1	token	A	19	(Null)	(Null
t_doctor	1	hospitalId	1	hospitalId	A	19	(Null)	(Null
t_doctor	1	hospitalId	2	departmentId	A	19	(Null)	(Null
t_doctor	1	hospitalId_2	1	hospitalId	A	19	(Null)	(Null
t_doctor	1	hospitalId_2	2	departmentId	A	19	(Null)	(Null
t_doctor	1	hospitalId_2	3	titleId	A	19	(Null)	(Null

三 分析

1 a 系统 64 位，内核较新
b 内存使用情况较多 8g 用了 7G
c 监听服务较多

2 a 磁盘使用情况良好,raid 情况未知
b cpu 处理性能尚可，增加更好,只有一颗 cpu

3 a MySQL 默认版本 5.1

b 默认引擎为 myisam 建议更改为 innodb,
c innodb cache 未设置具体值，
e 慢查询处理未设置
f 配置文件信息太少(信息获取太少)

4 存在 root 用户，建议删除（后台代码中用的 root 用户连接数据库）

5 测试服性能瓶颈在内存和 CPU,测试服数据量很少，完全够用

三 优化方案 -→拓展到正式服

1 硬件(现阶段测试服可以不用)

a 磁盘只有一块，建议冗余 然后做 raid5

/**********raid 介绍**********/

raid 独立磁盘冗余阵列,把多个相对便宜的硬盘组合起来，成为一个硬盘阵列

类型	优点	缺点	试用场景
raid0	读写性能很快	无冗余	Slave,作节点
raid1	100%冗余，镜像	占用硬件，速度一般	数据重要,不能宕机的业务。系统盘
raid5	具备一定冗余,可以坏磁盘，读性能不错	写入性能不高	一般业务都可以
raid10	读写很快,100%冗余	成本高	性能和冗余都需要的业务,存储的主节点,MySQL主库

/**********/

b 高并发可以适当 ssd 替换 sata

c 内存建议增大(公司为 8G，可以 16-32G) 物理 CPU 核数建议增加，增强处理性能

2 软件优化

a 系统 x64

b mysql 建议不要 yum 安装大于 5.5 稳定版本,源码安装或者二进制安装都可以 (公司为 Yum 安装)

c 系统监听业务太多，建议专门搭建数据库服务器

3 配置优化 my.cnf

a 添加 my.cnf 一些调优参数(针对 InnoDb)

```
max_connections = 512
max_connect_errors = 100
thread_concurrency = 8     //cpu 逻辑线程数 2-4 倍
default-storage-engine=INNODB
innodb_buffer_pool_size = 1G   //物理内存 1/2  根据其他情况
log-bin= /mysql_binlog/mysql-bin   //数据恢复方案 2   二进制备份
//mysql 主从的时候需要开启
thread_cache_size=64   //联合查询缓存区(后端经常会用到联合查询)
query_cache_size=64M   //后端不是所有的数据都缓存在 redis 中
//日志文件
Innodb_flush_log_at_trx_commit=0 //1 代表执行事务马上刷新  0 为每妙刷新
Innodb_log_buffer_size=8M
long_query_time=2 //  慢查询
slow_query_log_file = /tmp/mysqlslow.log1
```

b 监控 1 show global status\G 2 软件 myreport

3 SQL 优化

a 抓出慢查询,每天反馈给后端开发人员

b 少用避免多表之间的连接 join union

c 大的 sql 子查询，连接查询，尽量拆分

d 索引的优化，索引优化提升的查询性能

e 数据库是存储的地方，数据库的计算函数少用，存储过程少

用，对数据处理应该拿到后端的 java 程序中

4 架构优化

a 业务拆分,一部分频繁使用业务使用 nosql ,redis 前端缓存 session,token,用户信息 (现阶段医生信息,token 和医生欢迎语放在其中)

b 一部分业务直接用 redis 代替

c 查询尽量不用数据库 %like% (效率低，不能走索引),后期可以用 solr，从我的使用角度来讲，性能很好.

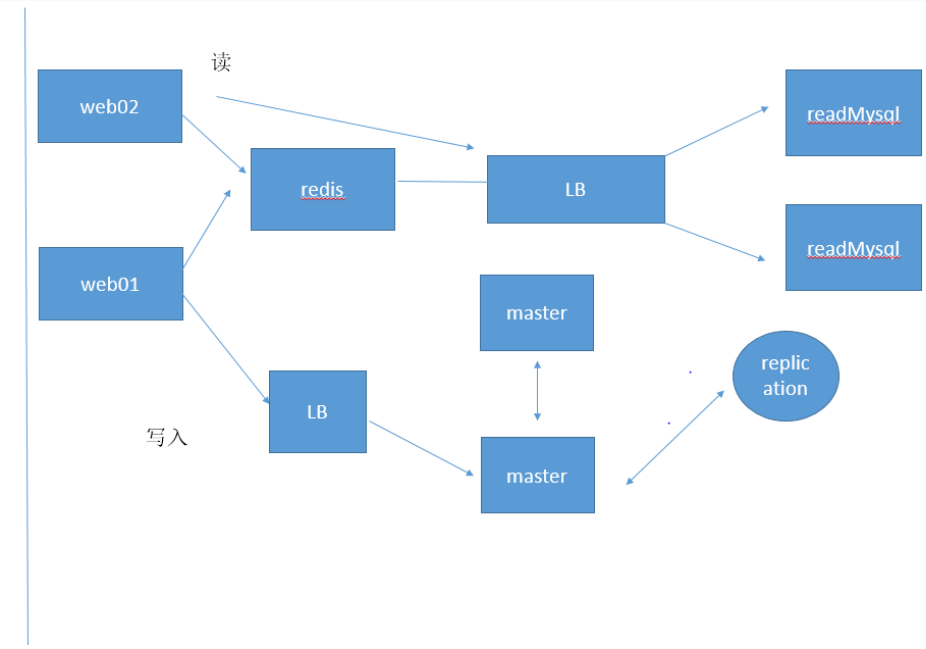
d 可以的首页，做成 html 静态页面，保存在 redis 中，需要更改的时候再添加数据----->静态化

/**********架构拓展 mysql**********/

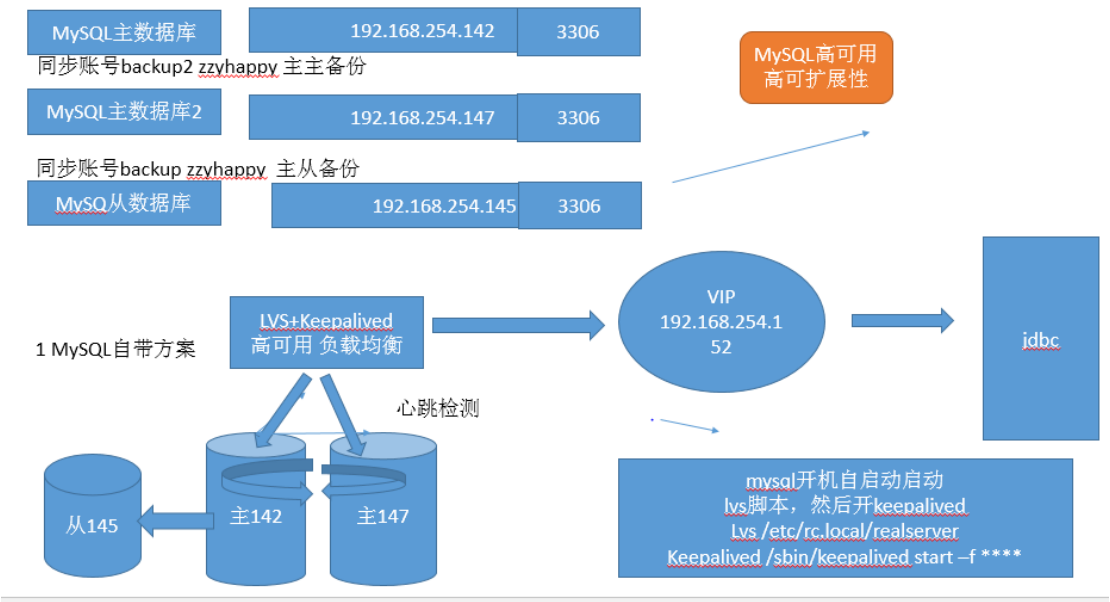
1 读写分离

a 根据情况 把读库的数据库引擎改为 myisam，查询性能优于 Innodb

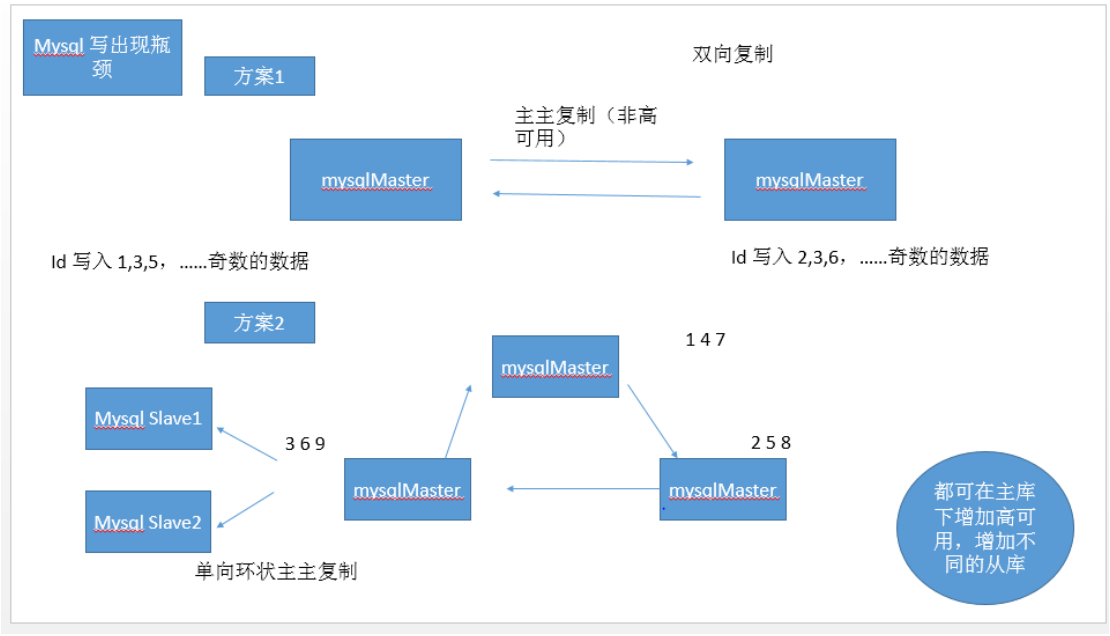
b 读库做负载均衡



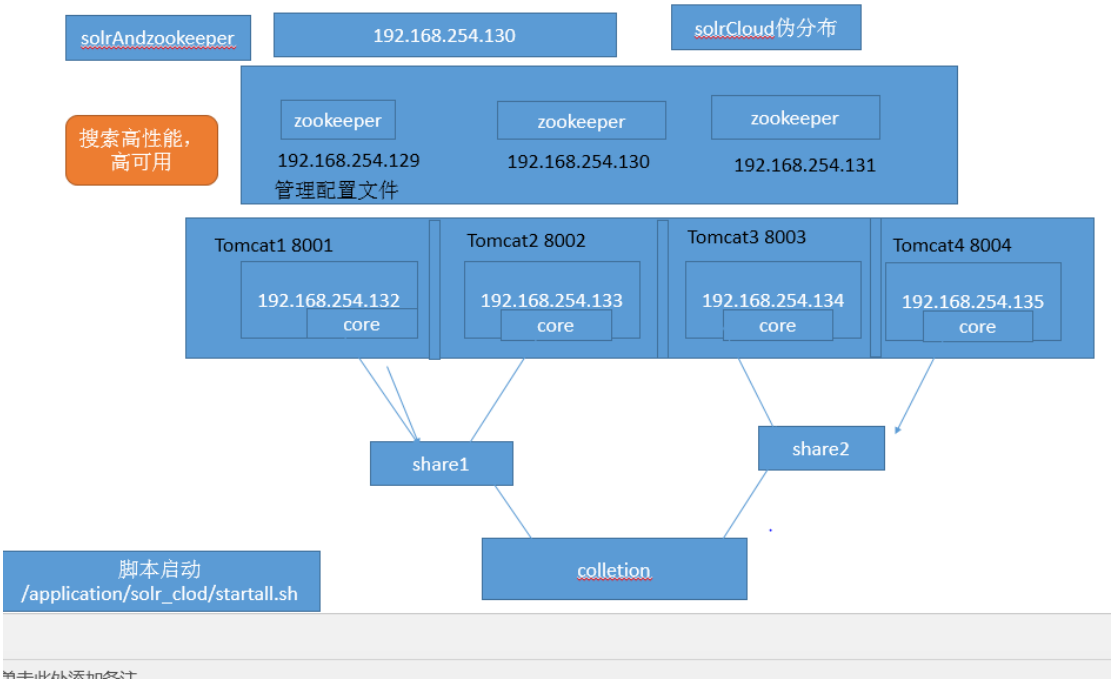
2 主从高可用



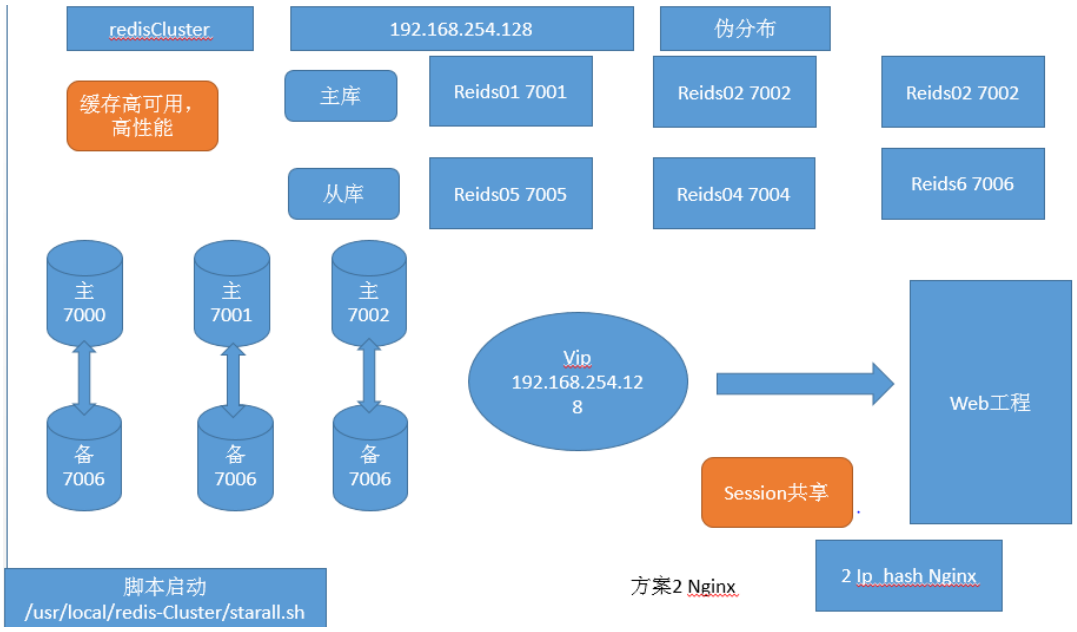
3 主从高性能(有写性能瓶颈)



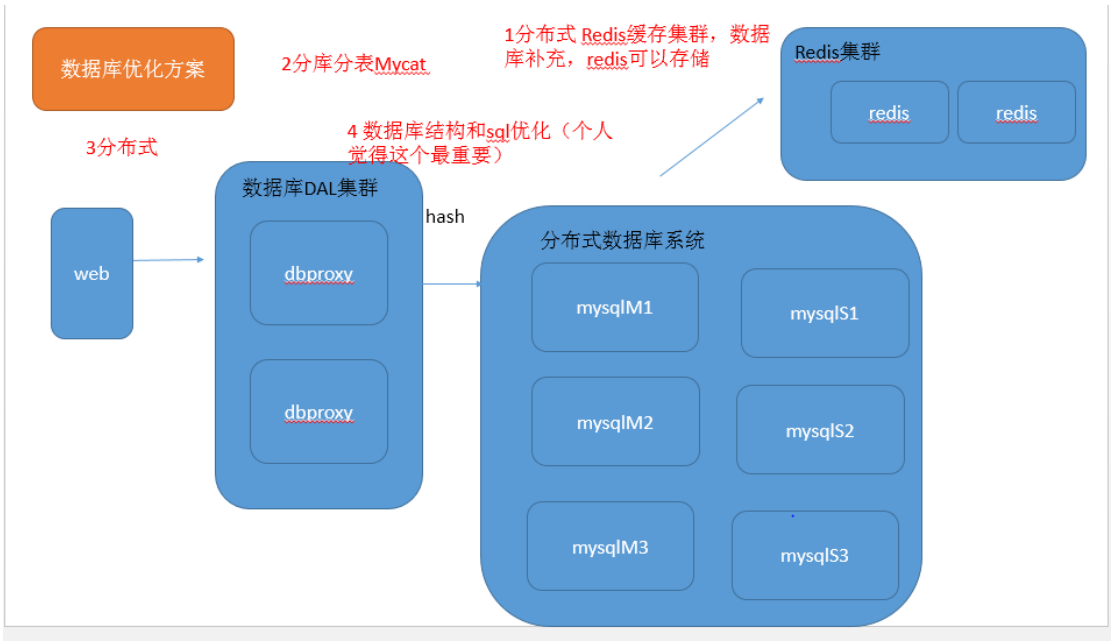
4 分离搜索集群[solrCload] 我测试过一台单实例和 6 个伪实例的 MySQL 数据 3000 条导入速度,6 个慢得多



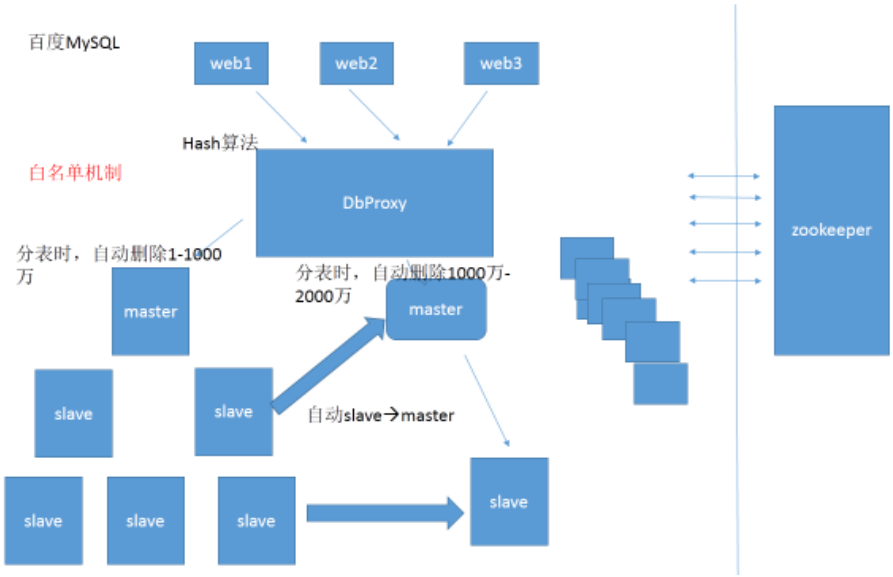
5 缓存 redis 集群



6 单表过大[人工—程序—软件--]



7 百度 mysql 架构


百度MySQL
web1
web2
web3
Hash算法
白名单机制
DbProxy
zookeeper
分表时，自动删除1-1000万
分表时，自动删除1000万-2000万
master
master
slave
slave
自动slave→master
slave
slave
slave
slave