DB详解

主数据库222详解：

1.关于crontab

```
* * * * * uptime >> /root/tasks/uptime.log                    #每次将uptime的内容写入当周的uptime.log里面
59 23 * * 0 sh /root/tasks/uptime-rotate.sh                   #每周日，将当周的uptime.log转存到指定目录
05 20 * * * cp -a `cat /data/confs-bkup/FileList` /data/confs-bkup   #每天拷贝使用的配置文件到指定目录，供rsync同步
10 23 * * * sh /root/tasks/backupfb.sh >> /root/tasks/backup.log 2>&1   #使用mysqldump进行数据库备份


*/1 * * * * /root/scripts/get_mysql_status.pl > /dev/null 2>&1   #根据iostat统计和快照获取数据库信息
0 0 * * * /root/scripts/switch_log.sh                          #切换日志文件并且对于超过七天的日志进行删除

09 00 * * * /root/scripts/mysql_flushlog.sh                    #重建日志文件，当日志进行轮询之后
```

2.关于各种日志存储

1）系统运行状态日志（uptime.log）

日志路径：/root/tasks/uptime.log

使用uptime命令查询系统的运行的参数。

```
[root@Pharm_DB_222 ~]# uptime
09:15:25 up 83 days, 17:07, 1 user, load average: 0.08, 0.23, 0.24
时间              已运行时间    一个用户          系统平均负载
```

在crontab中通过uptime >> /root/tasks/uptime.log

每分钟都记录到uptime.log中

```
[root@Pharm_DB_222 ~]# tail -20 /root/tasks/uptime.log
 09:02:01 up 83 days, 16:54,  0 users,  load average: 0.47, 0.28, 0.24
 09:03:01 up 83 days, 16:55,  0 users,  load average: 0.21, 0.24, 0.23
 09:04:01 up 83 days, 16:56,  0 users,  load average: 0.13, 0.21, 0.21
 09:05:01 up 83 days, 16:57,  0 users,  load average: 0.12, 0.19, 0.20
 09:06:01 up 83 days, 16:58,  0 users,  load average: 0.17, 0.19, 0.20
 09:07:01 up 83 days, 16:59,  0 users,  load average: 0.52, 0.27, 0.22
 09:08:02 up 83 days, 17:00,  1 user,  load average: 0.97, 0.45, 0.29
 09:09:01 up 83 days, 17:01,  1 user,  load average: 0.53, 0.42, 0.28
 09:10:01 up 83 days, 17:02,  1 user,  load average: 0.47, 0.40, 0.28
 09:11:01 up 83 days, 17:03,  1 user,  load average: 0.32, 0.38, 0.28
 09:12:01 up 83 days, 17:04,  1 user,  load average: 0.15, 0.32, 0.26
 09:13:01 up 83 days, 17:05,  1 user,  load average: 0.34, 0.34, 0.27
 09:14:01 up 83 days, 17:06,  1 user,  load average: 0.16, 0.29, 0.26
 09:15:01 up 83 days, 17:07,  1 user,  load average: 0.12, 0.25, 0.25
 09:16:01 up 83 days, 17:08,  1 user,  load average: 0.26, 0.26, 0.25
 09:17:01 up 83 days, 17:09,  1 user,  load average: 0.13, 0.23, 0.24
 09:18:01 up 83 days, 17:10,  1 user,  load average: 0.05, 0.18, 0.22
 09:19:01 up 83 days, 17:11,  1 user,  load average: 0.05, 0.16, 0.21
 09:20:01 up 83 days, 17:12,  1 user,  load average: 0.11, 0.16, 0.20
 09:21:01 up 83 days, 17:13,  1 user,  load average: 0.24, 0.19, 0.21
```

该日志的轮询并未采用系统自带的logrotate程序而是手写脚本轮询的

通过/root/tasks/uptime-rotate.sh

```
#!/bin/sh
cd /root/tasks
x=`date '+%V'`        #以星期一为一周的第一天，计算这是全年的第几周
mkdir -p uptime
mv uptime.log uptime/$x.log
```

```
[root@Pharm_DB_222 uptime]# ls
01.log  07.log  13.log  19.log  25.log  31.log  37.log  43.log  49.log
02.log  08.log  14.log  20.log  26.log  32.log  38.log  44.log  50.log
03.log  09.log  15.log  21.log  27.log  33.log  39.log  45.log  51.log
04.log  10.log  16.log  22.log  28.log  34.log  40.log  46.log  52.log
05.log  11.log  17.log  23.log  29.log  35.log  41.log  47.log  53.log
06.log  12.log  18.log  24.log  30.log  36.log  42.log  48.log
```

相当于以一周为转存周期，以一年为轮询周期的日志轮询。

2）本地数据库表备份日志

操作日志路径：/root/tasks/backup.log

用于记录数据备份过程中的情况，每日一次。

操作脚本：1./root/tasks/backupfb.sh　　#负责生成用来完成备份以及压缩操作的脚本

2./data/sqlback/bk.sh　　#第一个脚本调用的子脚本，负责执行备份工作

3./data/sqlback/gz.sh　　#第一个脚本调用的子脚本，负责对于备份完成之后的文件进行压缩

backupfb.sh内容：

```
#!/bin/sh
```

```
echo "begin"
echo `date`
BKDIR=/data/sqlback
if [ ! -d $BKDIR ]
then
  mkdir $BKDIR
  chown mysql.mysql $BKDIR
fi
cd $BKDIR
BKDBF=$BKDIR"/dblist.txt"
mysql -e 'show databases' |grep -v Database|grep -v information_schema > $BKDBF
cat $BKDBF |awk '{print "if [ ! -d "$1" ]; then mkdir "$1"; chown mysql.mysql "$1"; fi;rm -f "$1"/*;mysqldump --tab="$1" "$1"}' > $BKDIR"/bk.sh"          #将指定指令存入bk.sh  指令中表示先判断备份的目的目录是否存在，若存在则清空，备份指定的表到改目录中，一共会生成.txt和.sql两种文件，txt中存储所以数据，sql中存储表结构。
sh ./bk.sh
cat $BKDBF |awk '{print "gzip "$1"/*"}' > $BKDIR"/gz.sh"  #生成压缩脚本        压缩指定目录中的所以选项
sh ./gz.sh
echo "end"
echo `date`
```

两天完整日志截图：

```
begin
Sun Feb 1 23:10:01 CST 2015
mysqldump: Got error: 1194: Table 'message' is marked as crashed and should be repaired when exe
mysqldump: Got error: 1142: SELECT,LOCK TABL command denied to user 'root'@'localhost' for table
gzip: performance_schema/*: No such file or directory
end
Mon Feb 2 00:16:58 CST 2015
begin
Mon Feb 2 23:10:01 CST 2015
mysqldump: Got error: 1194: Table 'message' is marked as crashed and should be repaired when exe
mysqldump: Got error: 1142: SELECT,LOCK TABL command denied to user 'root'@'localhost' for table
gzip: performance_schema/*: No such file or directory
end
```

3）数据库错误日志mysqld

日志路径：/var/log/mysqld.log

轮询控制：/etc/logrotate.d/mysql 每日重置，保存包括当日在内的四天mysqld日志

关于轮询中特殊脚本控制：

```
postrotate
    if test -x /usr/bin/mysqladmin &&\
      /usr/bin/mysqladmin ping &>/dev/null     #若mysqladmin存在并且可执行同时数据库正处于运行状态。不将错误输出到
                                                屏幕上
    then
      /usr/bin/mysqladmin flush-logs           #重新生成有关mysql有关的日志
    fi
endscript
```

```
[root@Pharm_DB_222 ~]# ll /var/log/mysqld*
-rw-rw---- 1 mysql root 9643 Feb  4 10:10 /var/log/mysqld.log
-rw-rw---- 1 mysql root  887 Feb  4 04:02 /var/log/mysqld.log.1.gz
-rw-rw---- 1 mysql root  883 Feb  3 04:02 /var/log/mysqld.log.2.gz
-rw-rw---- 1 mysql root  884 Feb  2 04:02 /var/log/mysqld.log.3.gz
```

一般来说增删改查错误日志都可以在mysqld中查到了！

日志格式：

```
150204  9:25:18 [Warning] Unsafe statement written to the binary log using statement format since BI
use. This is unsafe because the set of rows included cannot be predicted. Statement: DELETE FROM `ge
150204  9:26:10 [Warning] Unsafe statement written to the binary log using statement format since BI
use. This is unsafe because the set of rows included cannot be predicted. Statement: DELETE FROM `ge
150204  9:27:33 [Warning] Unsafe statement written to the binary log using statement format since BI
use. This is unsafe because the set of rows included cannot be predicted. Statement: DELETE FROM `ge
150204  9:27:45 [Warning] Unsafe statement written to the binary log using statement format since BI
use. This is unsafe because the set of rows included cannot be predicted. Statement: DELETE FROM `ge
150204  9:27:59 [Warning] Unsafe statement written to the binary log using statement format since BI
use. This is unsafe because the set of rows included cannot be predicted. Statement: DELETE FROM `ge
150204 10:04:23 [Note] /usr/sbin/mysqld: Normal shutdown

150204 10:04:23 [Note] Event Scheduler: Purging the queue. 1 events
150204 10:04:24  InnoDB: Starting shutdown...
150204 10:04:28  InnoDB: Shutdown completed; log sequence number 1591090
150204 10:04:28 [Note] /usr/sbin/mysqld: Shutdown complete

150204 10:04:28 mysqld_safe mysqld from pid file /var/lib/mysql/Pharm_DB_222.pid ended
150204 10:04:29 mysqld_safe Starting mysqld daemon with databases from /var/lib/mysql
150204 10:04:29 [Warning] The syntax '--log-slow-queries' is deprecated and will be removed in a fut
150204 10:04:29 [Note] Flashcache bypass: disabled
150204 10:04:29 [Note] Flashcache setup error is : ioctl failed
```

年月日 系统时间 错误内容

4）慢查询日志

日志路径:/var/log/mysql_slow.log

轮询控制：/etc/logrotate.d/mysql-slow
每七天转储一次，保存当周在内的八周的信息。
my.cnf中关于慢查询的配置

```
long_query_time = 30                                  #慢查询的判定时间，选择超过30s则为慢查询
slow_query_log_file = /var/log/mysql_slow.log         #慢查询日志的存放位置
slow-query-log = on                                   #开启慢查询
slow_query_log_timestamp_precision = microsecond      #为慢查询的句子打上时间戳
```

慢查询格式：

```
# Time: 150206 11:56:33.0000519635
# User@Host: hub[hub] @  [10.103.33.154]
# Thread_id: 3312756  Schema: hub_hi2000  Last_errno: 0  Killed: 0
# Query_time: 55.580173  Lock_time: 0.000095  Rows_sent: 1  Rows_examined: 335113  Rows_affected: 0  Rows_read: 1
# Bytes_sent: 61  Tmp_tables: 0  Tmp_disk_tables: 0  Tmp_table_sizes: 0
use hub_hi2000;
SET timestamp=1423194993;
select Count(*) as num from chinapharmnet.product_en as p,chinapharmnet.company_en as c,hub_hi2000.minfo as m whe
re c.status=1 and p.status=1 and p.asid=c.id and c.poster = m.login and (m.mtype_en = 'Q' or m.mtype_en = 'P') li
mit 1;
```

一条完整的慢查询统计信息，包括时间，选择语句的源主机，查询持续时间，查询行数，具体语句等等。

5）系统日志/var/log/message

可以在系统日志中查看到，系统被请求连接的情况。这里主要的内容就是来自33.61的cacti的信息请求和来自33.70的rsync请求。

关于使用snmp的cacti连接请求情况：（每5分钟一次）

```
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40594
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Received SNMP packet(s) from UDP: [10.103.33.61]:40594
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40594
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40595
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Received SNMP packet(s) from UDP: [10.103.33.61]:40595
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40596
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Received SNMP packet(s) from UDP: [10.103.33.61]:40596
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40604
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Received SNMP packet(s) from UDP: [10.103.33.61]:40604
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40604
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40594
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40605
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Received SNMP packet(s) from UDP: [10.103.33.61]:40605
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40614
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Received SNMP packet(s) from UDP: [10.103.33.61]:40614
Feb  4 15:42:21 Pharm_DB_222 snmpd[1117]: Connection from UDP: [10.103.33.61]:40594
```

关于使用xinetd的rsync

```
[root@Pharm_DB_222 ~]# tail -10000 /var/log/messages |grep rsync
Feb  2 22:36:40 Pharm_DB_222 xinetd[1156]: START: rsync pid=21260 from=10.103.33.70
Feb  2 22:39:20 Pharm_DB_222 xinetd[1156]: EXIT: rsync status=0 pid=21260 duration=160(sec)
Feb  2 22:39:21 Pharm_DB_222 xinetd[1156]: START: rsync pid=23458 from=10.103.33.70
Feb  2 22:43:46 Pharm_DB_222 xinetd[1156]: EXIT: rsync status=0 pid=23458 duration=265(sec)
Feb  3 22:36:41 Pharm_DB_222 xinetd[1156]: START: rsync pid=4975 from=10.103.33.70
Feb  3 22:39:14 Pharm_DB_222 xinetd[1156]: EXIT: rsync status=0 pid=4975 duration=153(sec)
Feb  3 22:39:15 Pharm_DB_222 xinetd[1156]: START: rsync pid=7076 from=10.103.33.70
Feb  3 22:43:53 Pharm_DB_222 xinetd[1156]: EXIT: rsync status=0 pid=7076 duration=278(sec)
```

查看出现过的协议类型

```
[root@Pharm_DB_222 ~]# tail -10000 /var/log/messages |awk '{print $5}' |sort |uniq -c |sort -nr
   9963 snmpd[1117]:
     24 last
      8 xinetd[1156]:
      3 kernel:
      1 syslogd
      1 exiting
```

last是由于多次使用同一个IP进行snmp连接造成的。

6）系统运行状态日志（2）/root/script/log/watch.log

这个日志是由/root/scripts/get_mysql_status.pl该perl脚本生成的（perl脚本暂时看太懂）

由 /root/scripts/switch_log.sh进行日志轮询切换

```
[root@Pharm_DB_222 log]# ll
total 46964
-rw-r--r-- 1 root root 1623715 Feb  9 15:17 watch.log
-rw-r--r-- 1 root root 9171767 Feb  3 23:59 watch.log.02.03
-rw-r--r-- 1 root root 7739155 Feb  4 23:59 watch.log.02.04
-rw-r--r-- 1 root root 7033531 Feb  5 23:59 watch.log.02.05
-rw-r--r-- 1 root root 9242591 Feb  6 23:59 watch.log.02.06
-rw-r--r-- 1 root root 7683438 Feb  7 23:59 watch.log.02.07
-rw-r--r-- 1 root root 5492802 Feb  8 23:59 watch.log.02.08
```

```
15-02-09 15:11:02
slave lag :
thread number : 4
cpu : 0% and memory : 8.4%
Key block unused : 52630
disk util : 8.71%
temp disk file : 0

15-02-09 15:12:02
slave lag :
thread number : 11
cpu : 0% and memory : 8.4%
Key block unused : 52513
disk util : 8.71%
temp disk file : 4
```

第一行：时间 第二行：从数据库延迟 第三行：线程数 第四行：关键闲置酷块 第五行：磁盘使用率 第六行：临时文件数

7）系统登陆日志 /var/log/secure

在/etc/logrotate.d/syslog里面进行轮询