目录

[一、mysqlsla安装： 1](#_Toc368223638)

[1.下载 mysqlsla 1](#_Toc368223639)

[2、解压： 2](#_Toc368223640)

[3、执行perl脚本检查包依赖关系： 2](#_Toc368223641)

[4、安装： 2](#_Toc368223642)

[5、使用：（出现错误） 2](#_Toc368223643)

[6、解决方法： 3](#_Toc368223644)

[7、重新使用成功 3](#_Toc368223645)

[二、mysqlsla使用： 3](#_Toc368223646)

[1、查看帮助： 3](#_Toc368223647)

[2、简单使用： 4](#_Toc368223648)

[3、慢查询使用实例： 4](#_Toc368223649)

[4、常用使用参数说明： 5](#_Toc368223650)

[5、使用实例： 6](#_Toc368223651)

# 一、mysqlsla安装：

## 1.下载 mysqlsla

[root@localhost tmp]# wget http://hackmysql.com/scripts/mysqlsla-2.03.tar.gz

--19:45:45-- http://hackmysql.com/scripts/mysqlsla-2.03.tar.gz

Resolving hackmysql.com... 64.13.232.157

Connecting to hackmysql.com|64.13.232.157|:80... connected.

HTTP request sent, awaiting response... 200 OK

Length: 33674 (33K) [application/x-tar]

Saving to: `mysqlsla-2.03.tar.gz.2'

100%[===========================================================================================>] 33,674 50.2K/s in 0.7s

19:45:47 (50.2 KB/s) - `mysqlsla-2.03.tar.gz.2' saved [33674/33674]

## 2、解压：

[root@phicomm-crm2 zwj]# tar zxvf mysqlsla-2.03.tar.tar

mysqlsla-2.03/

mysqlsla-2.03/Changes

mysqlsla-2.03/INSTALL

mysqlsla-2.03/README

mysqlsla-2.03/Makefile.PL

mysqlsla-2.03/bin/

mysqlsla-2.03/bin/mysqlsla

mysqlsla-2.03/META.yml

mysqlsla-2.03/lib/

mysqlsla-2.03/lib/mysqlsla.pm

mysqlsla-2.03/MANIFEST

## 3、执行perl脚本检查包依赖关系：

[root@phicomm-crm2 mysqlsla-2.03]# perl Makefile.PL

Checking if your kit is complete...

Looks good

Writing Makefile for mysqlsla

yum install perl-ExtUtils-Embed -y

## 4、安装：

[root@phicomm-crm2 mysqlsla-2.03]# make && make install

cp lib/mysqlsla.pm blib/lib/mysqlsla.pm

cp bin/mysqlsla blib/script/mysqlsla

/usr/bin/perl -MExtUtils::MY -e 'MY->fixin(shift)' -- blib/script/mysqlsla

Manifying blib/man3/mysqlsla.3pm

Installing /usr/local/share/perl5/mysqlsla.pm

Installing /usr/local/share/man/man3/mysqlsla.3pm

Installing /usr/local/bin/mysqlsla

## 5、使用：（出现错误）

[root@phicomm-crm2 bin]# mysqlsla -lt slow /usr/local/mysql/mysql\_slow.log

Can't locate Time/HiRes.pm in @INC (@INC contains: /usr/local/lib64/perl5 /usr/local/share/perl5 /usr/lib64/perl5/vendor\_perl /usr/share/perl5/vendor\_perl /usr/lib64/perl5 /usr/share/perl5 .) at /usr/local/bin/mysqlsla line 2095.

BEGIN failed--compilation aborted at /usr/local/bin/mysqlsla line 2095.

## 6、解决方法：

[root@phicomm-crm2 zwj]# yum install perl-Time-HiRes

7、重新使用成功：

[root@phicomm-crm2 zwj]# mysqlsla -lt slow /usr/local/mysql/mysql\_slow.log

Report for slow logs: /usr/local/mysql/mysql\_slow.log

0 queries total, 0 unique

Sorted by 't\_sum'

Grand Totals: Time 0 s, Lock 0 s, Rows sent 0, Rows Examined 0

# 二、mysqlsla使用：

## 1、查看帮助：

mysqlsla慢查询分析工具与其他软件不同，看他的帮助只能使用man mysqlsla,不能使用mysqlsla –help或者mysqlsla –h。

[root@phicomm-crm2 mysql]# man mysqlsla

mysqlsla(3) User Contributed Perl Documentation mysqlsla(3)

NAME

mysqlsla - Parse, filter, analyze and sort MySQL slow, general and binary logs

SYNOPSIS

# Basic operation: parse a MySQL slow or general log

mysqlsla --log-type slow LOG

mysqlsla --log-type general LOG

# Parse output from mysqlbinlog

# mysqlsla cannot directly parse binary logs

mysqlbinlog LOG | mysqlsla --log-type binary -

# Parse a microslow patched slow log

mysqlsla --log-type msl LOG

# Replay a replay file

mysqlsla --replay FILE

# Parse a user-defined log specify its format

mysqlsla --log-type udl --udl-format FILE

# Let mysqlsla automatically determine the log type

mysqlsla LOG

## 2、简单使用：

二进制日志：

[root@phicomm-crm2 mysql]# mysqlbinlog mysql-bin.000008 | mysqlsla -lt binary -

通用日志：

mysqlsla -lt general general.log或

mysqlsla –log-type general general.log

慢查询日志：

[root@phicomm-crm2 mysql]# mysqlsla -lt slow /usr/local/mysql/mysql\_slow.log

## 3、慢查询使用实例：

[root@localhost mysqlsla-2.03]# mysqlsla -lt slow /tmp/127\_slow.log | more

Report for slow logs: /tmp/127\_slow.log

24 queries total, 6 unique

Sorted by 't\_sum'

Grand Totals: Time 16 s, Lock 1 s, Rows sent 18, Rows Examined 2.10M

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 001 \_\_\_

Count : 18 (75.00%)

Time : 15 s total, 833.333 ms avg, 0 to 8 s max (93.75%)

95% of Time : 7 s total, 411.765 ms avg, 0 to 4 s max

Lock Time (s) : 0 total, 0 avg, 0 to 0 max (0.00%)

95% of Lock : 0 total, 0 avg, 0 to 0 max

Rows sent : 0 avg, 0 to 0 max (0.00%)

Rows examined : 116.51k avg, 8 to 1.05M max (99.99%)

Database :

Users :

root@localhost : 100.00% (18) of query, 100.00% (24) of all users

Query abstract:

INSERT INTO t2 SELECT \* FROM t2;

Query sample:

insert into t2 select \* from t2;

........

选项说明：

总查询次数 (queries total)， 去重后的sql数量 (unique)

输出报表的内容排序(sorted by)

最重大的慢sql统计信息, 包括 平均执行时间, 等待锁时间, 结果行的总数, 扫描的行总数.

Count, sql的执行次数及占总的slow log数量的百分比.

Time, 执行时间, 包括总时间, 平均时间, 最小, 最大时间, 时间占到总慢sql时间的百分比.

95% of Time, 去除最快和最慢的sql, 覆盖率占95%的sql的执行时间.

Lock Time, 等待锁的时间.

95% of Lock , 95%的慢sql等待锁时间.

Rows sent, 结果行统计数量, 包括平均, 最小, 最大数量.

Rows examined, 扫描的行数量.

Database, 属于哪个数据库

Users, 哪个用户,IP, 占到所有用户执行的sql百分比

Query abstract, 抽象后的sql语句

Query sample, sql语句

## 4、常用使用参数说明：

1) -log-type (-lt) type logs:

通过这个参数来制定log的类型，主要有slow, general, binary, msl, udl,分析slow log时通过制定为slow.

2) -sort:

制定使用什么参数来对分析结果进行排序，默认是按照t\_sum来进行排序。

t\_sum:按总时间排序

c\_sum:按总次数排序

c\_sum\_p: sql语句执行次数占总执行次数的百分比。

3) -top:

显示sql的数量，默认是10,表示按规则取排序的前多少条

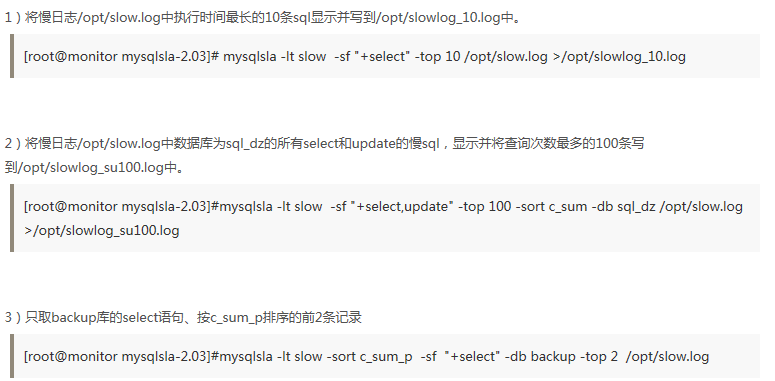
4) –statement-filter (-sf) [+-][TYPE]:

过滤sql语句的类型，比如select、update、drop.

[TYPE]有SELECT, CREATE, DROP, UPDATE, INSERT，例如"+SELECT,INSERT"，不出现的默认是-，即不包括。

5) –db --databases：要处理哪个库的日志：

## 5、使用实例：



**1）普通查看慢查询日志内容：**

[root@phicomm-crm2 mysql]# cat mysql\_slow.log

/usr/local/mysql/bin/mysqld, Version: 5.6.10-log (Source distribution). started with:

Tcp port: 3306 Unix socket: (null)

Time Id Command Argument

# Time: 130929 13:42:46

# User@Host: root[root] @ localhost [] Id: 2383

# Query\_time: 54.155006 Lock\_time: 0.000100 Rows\_sent: 3387835 Rows\_examined: 6791718

use test;

SET timestamp=1380433366;

select distinct serialNumber from pro\_warranty;

# Time: 130929 13:50:01

# User@Host: root[root] @ localhost [] Id: 2495

# Query\_time: 7.663700 Lock\_time: 0.019585 Rows\_sent: 0 Rows\_examined: 1000000

SET timestamp=1380433801;

create table zz as select \* from pro\_warranty limit 1000000;

# Time: 130929 13:50:59

# User@Host: root[root] @ localhost [] Id: 2495

# Query\_time: 15.212835 Lock\_time: 0.008970 Rows\_sent: 0 Rows\_examined: 2000000

SET timestamp=1380433859;

create table xx as select \* from pro\_warranty limit 2000000;

# Time: 130929 13:51:37

# User@Host: root[root] @ localhost [] Id: 2495

# Query\_time: 18.659476 Lock\_time: 0.008885 Rows\_sent: 0 Rows\_examined: 2500000

SET timestamp=1380433897;

create table yy as select \* from pro\_warranty limit 2500000;

**2）以上显示结果不太好看。**

**我们使用mysqlsla查看慢查询日志，比较友好的界面：**

**显示慢查询日志中执行时间最长的前3条语句：**

[root@phicomm-crm2 mysql]# mysqlsla -lt slow --top 3 /usr/local/mysql/mysql\_slow.log

Report for slow logs: /usr/local/mysql/mysql\_slow.log

4 queries total, 4 unique

Sorted by 't\_sum'

Grand Totals: Time 96 s, Lock 0 s, Rows sent 3.39M, Rows Examined 12.29M

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 001 \_\_\_

Count : 1 (25.00%)

Time : 54.155006 s total, 54.155006 s avg, 54.155006 s to 54.155006 s max (56.59%)

Lock Time (s) : 100 祍 total, 100 祍 avg, 100 祍 to 100 祍 max (0.27%)

Rows sent : 3.39M avg, 3.39M to 3.39M max (100.00%)

Rows examined : 6.79M avg, 6.79M to 6.79M max (55.25%)

Database : test

Users :

root@localhost : 100.00% (1) of query, 100.00% (4) of all users

Query abstract:

SET timestamp=N; SELECT DISTINCT serialnumber FROM pro\_warranty;

Query sample:

SET timestamp=1380433366;

select distinct serialNumber from pro\_warranty;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 002 \_\_\_

Count : 1 (25.00%)

Time : 18.659476 s total, 18.659476 s avg, 18.659476 s to 18.659476 s max (19.50%)

Lock Time (s) : 8.885 ms total, 8.885 ms avg, 8.885 ms to 8.885 ms max (23.67%)

Rows sent : 0 avg, 0 to 0 max (0.00%)

Rows examined : 2.50M avg, 2.50M to 2.50M max (20.34%)

Database :

Users :

root@localhost : 100.00% (1) of query, 100.00% (4) of all users

Query abstract:

SET timestamp=N; CREATE TABLE yy AS SELECT \* FROM pro\_warranty LIMIT N;

Query sample:

SET timestamp=1380433897;

create table yy as select \* from pro\_warranty limit 2500000;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 003 \_\_\_

Count : 1 (25.00%)

Time : 15.212835 s total, 15.212835 s avg, 15.212835 s to 15.212835 s max (15.90%)

Lock Time (s) : 8.97 ms total, 8.97 ms avg, 8.97 ms to 8.97 ms max (23.89%)

Rows sent : 0 avg, 0 to 0 max (0.00%)

Rows examined : 2.00M avg, 2.00M to 2.00M max (16.27%)

Database :

Users :

root@localhost : 100.00% (1) of query, 100.00% (4) of all users

Query abstract:

SET timestamp=N; CREATE TABLE xx AS SELECT \* FROM pro\_warranty LIMIT N;

Query sample:

SET timestamp=1380433859;

create table xx as select \* from pro\_warranty limit 2000000;

**3）按照执行总时间t\_sum进行排序：**

**[root@phicomm-crm2 mysql]# mysqlsla -lt slow --top 3 --sort t\_sum /usr/local/mysql/mysql\_slow.log**

**4）按照执行总次数c\_sum进行排序**

**[root@phicomm-crm2 mysql]# mysqlsla -lt slow --top 3 --sort c\_sum /usr/local/mysql/mysql\_slow.log**

**5）为了方便查看，也可以将输出结果重定向到某个文件：**

**[root@phicomm-crm2 mysql]# mysqlsla -lt slow --top 3 --sort c\_sum /usr/local/mysql/mysql\_slow.log > /tmp/zwj/mysql\_slwo.log**

**6）只取出create语句的慢查询结果：**

**[root@phicomm-crm2 mysql]# mysqlsla -lt slow -sf "+create"--top 3 --sort c\_sum /usr/local/mysql/mysql\_slow.log > /tmp/zwj/mysql\_slwo.log**

**7）只取出数据库test的create语句的慢查询（前3条）**

**[root@phicomm-crm2 mysql]# mysqlsla -lt slow -sf "+create"--top 3 --sort c\_sum -db test /usr/local/mysql/mysql\_slow.log > /tmp/zwj/mysql\_slwo.log**