[MySQL常见错误集锦](http://www.cnblogs.com/shiyangxt/articles/1260491.html)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Mysql 常见错误集锦\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Host ’...’ is blocked错误   
如果你得到象这样的一个错误：   
Host ’hostname’ is blocked because of many connection errors.   
Unblock with ’mysqladmin flush-hosts’

这意味着，mysqld已经得到了大量(max\_connect\_errors)的主机’hostname’的在中途被中断了的连接请求。在max\_connect\_errors次失败请求后，mysqld认定出错了(象来自一个黑客的攻击)，并且阻止该站点进一步的连接，直到某人执行命令mysqladmin flush-hosts。   
缺省地，mysqld在10个连接错误后阻塞一台主机。你可以通过象这样启动服务器很容易地调整它：   
shell> safe\_mysqld -O max\_connect\_errors=10000 &   
  
注意，对给定的主机，如果得到这条错误消息，你应该首先检查该主机的TCP/IP连接有没有问题。如果你的TCP/IP连接不在运行，增加max\_connect\_errors变量的值对你也不会有帮助！

2. Too many connections错误   
如果在你试图连接MySQL时，你得到错误Too many connections，这意味着已经有max\_connections个客户连接了mysqld服务器。   
如果你需要比缺省(100)更多的连接，那么你应该重启mysqld，用更大的 max\_connections 变量值。   
注意，mysqld实际上允许(max\_connections+1)个客户连接。最后一个连接是为一个用Process权限的用户保留的。通过不把这个权限给一般用户(他们不应该需要它)，有这个权限一个管理员可以登录并且使用SHOW PROCESSLIST找出什么可能出错。见7.21 SHOW句法(得到表，列的信息）。

3. Out of memory错误   
如果你发出查询并且得到类似于下面的错误：   
mysql: Out of memory at line 42, ’malloc.c’   
mysql: needed 8136 byte (8k), memory in use: 12481367 bytes (12189k)   
ERROR 2008: MySQL client ran out of memory   
  
注意，错误指向了MySQL客户mysql。这个错误的原因很简单，客户没有足够的内存存储全部结果。   
为了修正这个问题，首先检查你的查询是否正确。它应该返回这么多的行，这合理吗？如果是这样，你可以使用mysql --quick，它使用mysql\_use\_result()检索结果集合。这将较少的负担放在了客户端(只是服务器更多)。   
  
4.Packet too large错误   
当一个MySQL客户或mysqld服务器得到一个比max\_allowed\_packet个字节长的包，它发出一个Packet too large错误并终止连接。   
如果你正在使用mysql客户，你可以通过用mysql --set-variable=max\_allowed\_packet=8M指定一个更大的缓冲区来启动客户程序。   
如果你正在使用不允许你指定最大包大小的其他客户(例如 DBI)，你需要在你启动服务器时设置包大小。你可以使用mysqld的命令行选项设置max\_allowed\_packet为一个更大的尺寸。例如，如果你正期望将一个全长的BLOB存入一张表中，你将需要用--set-variable=max\_allowed\_packet=24M选项来启动服务器。

5. The table is full错误   
这个错误发生在内存临时表变得比tmp\_table\_size字节大时。为了避免这个问题，你可以使用mysqld的-O tmp\_table\_size=#选项来增加临时表的大小，或在你发出有疑问的查询之前使用SQL选项SQL\_BIG\_TABLES。见7.25 SET OPTION句法。   
你也可以使用--big-tables选项启动mysqld。这与为所有查询使用SQL\_BIG\_TABLES完全相同。

6. Commands out of sync in client错误   
如果你在你的客户代码中得到Commands out of sync; You can’t run this command now，你正在以错误的次序调用客户函数！   
这可能发生，例如，如果你正在使用mysql\_use\_result()并且在你已经调用了mysql\_free\_result()之前试图执行新查询。如果你在mysql\_use\_result()或mysql\_store\_result()之间试图执行返回数据的2个查询，它也可能发生。   
  
7. Ignoring user错误   
如果你得到下列错误：   
Found wrong password for user: ’some\_user@some\_host’; Ignoring user   
这意味着在mysqld启动时或在它再次装载权限表时，它在user表中找到了一个有一个无效口令的条目。结果，条目简单地被权限系统忽略。   
可能导致这个问题的原因和修正：   
你可能正在运行一个有一个老的user表的新版本mysqld。你可以通过执行mysqlshow mysql user看看口令字段是否少于 16个字符来检查它。如果是这样，你可以通过运行scripts/add\_long\_password脚本改正这种情况。   
用户有一个老式的口令(8个字符长)并且你没使用--old-protocol选项启动mysqld。用一个新口令更新在user表中的用户或用--old-protocol重启mysqld。   
你没有使用PASSWORD()函数在在user表中指定了一个口令。使用mysql以一个新口令更新在user表中的用户。确保使用PASSWORD()函数：   
mysql> update user set password=PASSWORD(’your password’)   
where user=’XXX’;

8. Table ’xxx’ doesn’t exist错误   
如果你得到错误Table ’xxx’ doesn’t exist或Can’t find file: ’xxx’ (errno: 2)，这意味着在当前数据库中没有名为xxx的表存在。   
注意，因为MySQL使用目录和文件存储数据库和表，数据库和表名件是区分大小写的！（在Win32上，数据库和表名不是区分大小写的，但是在查询中对所有表的引用必须使用相同的大小写！）   
你可以用SHOW TABLES检查你在当前数据库中有哪个表。见7.21 SHOW句法(得到表、列的信息）。

9. MySQL怎样处理一个溢出的磁盘   
当出现一个磁盘溢出的情况时，MySQL做下列事情：   
它每分钟检查一次看是否有足够空间写入当前行。如果有足够的空间，它继续好像发生什么事情。   
每6分钟它将有关磁盘溢出的警告写入日志文件。   
为了缓和这个问题，你可以采取下列行动：   
继续，你只需释放足够的空闲磁盘空间以便插入所有记录。   
放弃线程，你必须发一个mysqladmin kill到线程。在下一次检查磁盘时，线程将被放弃(在1分钟内)。   
注意，其他线程可能正在等待引起“磁盘溢出”条件的表。如果你有几个“锁定的”的线程，杀死正在等待磁盘溢出条件的那个线程将允许其他线程继续。   
  
10. 如何从一个文本文件运行SQL命令   
一般地，mysql客户被交互性地使用，象这样：   
shell> mysql database   
然而，也可以把你的SQL命令放在一个文件中并且告诉mysql从该文件读取其输入。要想这样做，创造一个文本文件“text\_file”，它包含你想要执行的命令。然后如下那样调用mysql：   
shell> mysql database < text\_file   
  
你也能启动有一个USE db\_name语句的文本文件。在这种情况下，在命令行上指定数据库名是不必要的：   
shell> mysql < text\_file    
  
11. MySQL在哪儿存储临时文件   
MySQL使用TMPDIR环境变量的值作为存储临时文件的目录的路径名。如果你没有设置TMPDIR，MySQL使用系统缺省值，它通常是“/tmp”或“/usr/tmp”。如果包含你的临时文件目录的文件系统太小，你应该编辑safe\_mysqld设定TMPDIR指向你有足够空间的一个文件系统！你也可以使用mysqld的--tmpdir选项目设置临时目录。   
MySQL以“隐含文件”创建所有临时文件。这保证了如果mysqld被终止，临时文件也将被删除。使用隐含文件的缺点是你将看不到一个大的临时文件填满了临时文件目录所在的文件系统。   
当排序(ORDER BY或GROUP BY)时，MySQL通常使用一个或两个临时文件。最大磁盘空间需求是：   
(存储东西的长度 + sizeof (数据库指针))   
\* 匹配的行数   
\* 2

sizeof(数据库指针)通常是4，但是在未来对确实很大的表可能增加。   
对一些SELECT查询，MySQL也创建临时SQL表。这些没被隐含且有“SQL\_\*”格式的名字。   
ALTER TABLE和OPTIMIZE TABLE在原数据库表的同一个目录中创建一张临时表。   
  
12. 怎样保护“/tmp/mysql.sock ”不被删除   
如果你有这个问题，事实上任何人可以删除MySQL通讯套接字“/tmp/mysql.sock”，在Unix的大多数版本上，你能通过为其设置sticky（t）位来保护你的“/tmp”文件系统。作为root登录并且做下列事情：   
shell> chmod +t /tmp   
  
这将保护你的“/tmp”文件系统使得文件仅能由他们的所有者或超级用户(root)删除。   
你能执行ls -ld /tmp检查sticky位是否被设置，如果最后一位许可位是t，该位被设置了。   
  
13. Access denied错误   
见6.6 权限系统如何工作。并且特别要看6.13 引起Access denied错误的原因。   
  
14. 怎样作为一个一般用户运行MySQL   
MySQL服务器mysqld能被任何用户启动并运行。为了将mysqld改由Unix用户user\_name来运行，你必须做下列事情：   
如果它正在运行，停止服务器(使用mysqladmin shutdown)。   
改变数据库目录和文件以便user\_name有权限读和写文件(你可能需要作为Unix的root用户才能做到)：   
shell> chown -R user\_name /path/to/mysql/datadir   
  
如果在MySQL数据目录中的目录或文件是符号链接，你也将需要顺着那些链接并改变他们指向的目录和文件。chown -R不能跟随符号链接。   
以user\_name用户启动服务器，或如果你正在使用MySQL 3.22或以后版本，以Unix root用户启动mysqld并使用--user=user\_name选项，mysqld将在接受任何连接之前切换到以Unix user\_name用户运行。   
如果在系统被重新启动时，你使用mysql.server脚本启动mysqld，你应该编辑mysql.server用su以用户user\_name运行mysqld，或使用--user选项调用mysqld。（不改变safe\_mysqld是必要的。）   
现在，你的mysqld进程应该正在作为Unix用户user\_name运行，并运行完好。尽管有一件事情没有变化：权限表的内容。缺省 地(就在运行了脚本mysql\_install\_db安装的权限表后)，MySQL用户root是唯一有存取mysql数据库或创建或抛弃数据库权限的用户。除非你改变了那些权限，否则他们仍然保持。当你作为一个Unix用户而不是root登录时，这不应该阻止你作为MySQL root用户来存取MySQL；只要为客户程序指定-u root的选项。   
注意通过在命令行上提供-u root，作为root存取MySQL，与作为Unix root用户或其他Unix用户运行MySQL没有关系。MySQL的存取权限和用户名与Unix用户名字是完全分开的。唯一与Unix用户名有关的是，如果当你调用一个客户程序时，你不提供一个-u选项，客户将试图使用你的Unix登录名作为你的MySQL用户名进行连接。   
如果你的Unix机器本身不安全，你可能应该至少在存取表中为MySQL root用户放上一个口令。否则，在那台机器上有一个帐号的任何用户能运行mysql -u root db\_name并且做他喜欢做的任何事情。

15. 怎样重新设置一个忘记的口令   
如果你忘记了MySQL的root用户的口令，你可以用下列过程恢复它。   
通过发送一个kill（不是kill -9)到mysqld服务器来关闭mysqld服务器。pid 被保存在一个.pid文件中，通常在MySQL数据库目录中：   
kill `cat /mysql-data-directory/hostname.pid`

你必须是一个UNIX root用户或运行服务器的相同用户做这个。   
使用--skip-grant-tables选项重启mysqld。   
用mysql -h hostname mysql连接mysqld服务器并且用一条GRANT命令改变口令。见7.26 GRANT和REVOKE句法。你也可以用mysqladmin -h hostname -u user password ’new password’ 进行。   
用mysqladmin -h hostname flush-privileges或用SQL命令FLUSH PRIVILEGES来装载权限表。   
  
16. 文件许可权限问题   
如果你有与文件许可有关的问题，例如，如果当你创建一张表时，mysql发出下列错误消息：   
ERROR: Can’t find file: ’path/with/filename.frm’ (Errcode: 13)   
  
那么可能是在mysqld启动时，环境变量UMASK可能设置不正确。缺省的umask值是0660。你可以如下启动safe\_mysqld改变其行为：   
shell> UMASK=384 # = 600 in octal   
shell> export UMASK   
shell> /path/to/safe\_mysqld &   
  
17. 文件没找到   
如果你从MySQL得到ERROR ’...’ not found (errno: 23), Can’t open file: ... (errno: 24)或任何其他有errno 23或errno 24的错误，它意味着，你没有为MySQL分配足够的文件描述符。你能使用perror实用程序得到错误号含义是什么的描述：   
shell> perror 23   
File table overflow   
shell> perror 24   
Too many open files   
  
这里的问题是mysqld正在试图同时保持打开太多的文件。你也可以告诉mysqld一次不打开那么多的文件，或增加mysqld可得到的文件描述符数量。为了告诉mysqld一次保持打开更少的文件，你可以通过使用safe\_mysqld的-O table\_cache=32选项（缺省值是64)使表缓冲更小。减小max\_connections值也将减少打开文件的数量(缺省值是90)。   
要想改变mysqld可用的文件描述符数量，修改safe\_mysqld脚本。脚本中有一条注释了的行ulimit -n 256。你可以删除’#’字符来去掉该行的注释，并且改变数字256改变为mysqld可用的文件描述符的数量。   
ulimit能增加文件描述符的数量，但是只能到操作系统强加的限制。如果你需要增加每个进程可用的文件描述符数量的OS限制，参见你的操作系统文档。注意，如果你运行tcsh外壳，ulimit将不工作！当你请求当前限制时，tcsh也将报告不正确的值！在这种情况下，你应该用sh启动safe\_mysqld！   
  
18. 使用DATE列的问题   
一个DATE值的格式是’YYYY-MM-DD’。根据ANSI SQL，不允许其他格式。你应该在UPDATE表达式和SELECT语句的WHERE子句中使用这个格式。例如：   
mysql> SELECT \* FROM tbl\_name WHERE date >= ’1997-05-05’;

为了方便，如果日期用在数字上下文，MySQL自动变换一个日期到一个数字(并且反过来也如此)。当更新时和将一个日期与TIMESTAMP、DATE或DATETIME列比较的一个WHERE子句中，也是足够灵活以允许一种“宽松”的字符串格式。（宽松格式意味着任何标点字符用作在部件之间的分割符。例如，’1998-08-15’和’1998#08#15’是等价的。）MySQL也能变换不包含分割符的一个字符串(例如 ’19980815’)，如果它作为一个日期说得通。特殊日期’0000-00-00’可以作为’0000-00-00’被存储和检索。当通过MyODBC使用一个’0000-00-00’日期时，在MyODBC 2.50.12和以上版本，它将自动被转换为NULL，因为ODBC不能处理这种日期。   
因为MySQL实行了上述的变换，下列语句可以工作：   
mysql> INSERT INTO tbl\_name (idate) VALUES (19970505);   
mysql> INSERT INTO tbl\_name (idate) VALUES (’19970505’);   
mysql> INSERT INTO tbl\_name (idate) VALUES (’97-05-05’);   
mysql> INSERT INTO tbl\_name (idate) VALUES (’1997.05.05’);   
mysql> INSERT INTO tbl\_name (idate) VALUES (’1997 05 05’);   
mysql> INSERT INTO tbl\_name (idate) VALUES (’0000-00-00’);   
  
mysql> SELECT idate FROM tbl\_name WHERE idate >= ’1997-05-05’;   
mysql> SELECT idate FROM tbl\_name WHERE idate >= 19970505;   
mysql> SELECT mod(idate,100) FROM tbl\_name WHERE idate >= 19970505;   
mysql> SELECT idate FROM tbl\_name WHERE idate >= ’19970505’;   
  
然而，下列将不工作：   
mysql> SELECT idate FROM tbl\_name WHERE STRCMP(idate,’19970505’)=0;   
  
STRCMP()是字符串函数，因此它将idate转换为一个字符串并且实施字符串比较。它不将’19970505’转换为一个日期并实施日期比较。   
注意，MySQL不检查日期是否正确。如果你存储一个不正确的日期，例如’1998-2-31’，错误的日期将被存储。如果日期不能被变换到任何合理的值，在DATE字段中存储一个0。这主要是一个速度问题并且我们认为检查日期是应用程序的责任，而不服务器。

19. 时区问题   
如果你有一个问题，SELECT NOW()以GMT时间返回值而不是你的本地时间，你必须设定TZ环境变量为你的当前时区。这应该在服务器运行的环境进行，例如在safe\_mysqld或mysql.server中。   
  
20. 在搜索中的大小写敏感性   
缺省地，MySQL搜索是大小写不敏感的(尽管有一些字符集从来不是忽略大小写的，例如捷克语)。这意味着，如果你用col\_name LIKE ’a%’搜寻，你将得到所有以A或a开始的列值。如果你想要使这个搜索大小写敏感，使用象INDEX(col\_name, "A")=0检查一个前缀。或如果列值必须确切是"A"，使用STRCMP(col\_name, "A") = 0。   
简单的比较操作(>=、>、= 、< 、<=、排序和聚合)是基于每个字符的“排序值”。有同样排序值的字符(象E，e和’e)被视为相同的字符！   
LIKE比较在每个字符的大写值上进行(E==e 但是E<>’e）。   
如果你想要一个列总是被当作大小写敏感的方式，声明它为BINARY。见7.7 CREATE TABLE句法。   
如果你使用以所谓的big5编码的中文数据，你要使所有的字符列是BINARY，它可行，是因为big5编码字符的排序顺序基于 ASCII代码的顺序。   
  
21. NULL值问题   
NULL值的概念是造成SQL的新手的混淆的普遍原因，他们经常认为NULL是和一个空字符串’’的一样的东西。不是这样的！例如，下列语句是完全不同的：   
mysql> INSERT INTO my\_table (phone) VALUES (NULL);   
mysql> INSERT INTO my\_table (phone) VALUES ("");   
  
两个语句把值插入到phone列，但是第一个插入一个NULL值而第二个插入一个空字符串。第一个的含义可以认为是“电话号码不知道”，而第二个则可意味着“她没有电话”。   
在SQL中，NULL值在于任何其他值甚至NULL值比较时总是假的（FALSE）。包含NULL的一个表达式总是产生一个NULL值，除非在包含在表达式中的运算符和函数的文档中指出。在下列例子，所有的列返回NULL：   
mysql> SELECT NULL,1+NULL,CONCAT(’Invisible’,NULL);   
  
如果你想要寻找值是NULL的列，你不能使用=NULL测试。下列语句不返回任何行，因为对任何表达式，expr = NULL是假的：   
mysql> SELECT \* FROM my\_table WHERE phone = NULL;   
  
要想寻找NULL值，你必须使用IS NULL测试。下例显示如何找出NULL电话号码和空的电话号码：   
mysql> SELECT \* FROM my\_table WHERE phone IS NULL;   
mysql> SELECT \* FROM my\_table WHERE phone = "";   
  
在MySQL中，就像很多其他的SQL服务器一样，你不能索引可以有NULL值的列。你必须声明这样的列为NOT NULL，而且，你不能插入NULL到索引的列中。当用LOAD DATA INFILE读取数据时，空列用’’更新。如果你想要在一个列中有NULL值，你应该在文本文件中使用N。字面词’NULL’也可以在某些情形下使用。见

22. LOAD DATA INFILE句法。当使用ORDER BY时，首先呈现NULL值。如果你用DESC以降序排序，NULL值最后显示。当使用GROUP BY时，所有的NULL值被认为是相等的。为了有助于NULL的处理，你能使用IS NULL和IS NOT NULL运算符和IFNULL()函数。   
对某些列类型，NULL值被特殊地处理。如果你将NULL插入表的第一个TIMESTAMP列，则插入当前的日期和时间。如果你将NULL插入一个AUTO\_INCREMENT列，则插入顺序中的下一个数字。

23. alias问题   
你可以在GROUP BY、ORDER BY或在HAVING部分中使用别名引用列。别名也可以用来为列取一个更好点的名字：   
SELECT SQRT(a\*b) as rt FROM table\_name GROUP BY rt HAVING rt > 0;   
SELECT id,COUNT(\*) AS cnt FROM table\_name GROUP BY id HAVING cnt > 0;   
SELECT id AS "Customer identity" FROM table\_name;   
  
注意，你的 ANSI SQL 不允许你在一个WHERE子句中引用一个别名。这是因为在WHERE代码被执行时，列值还可能没有终结。例如下列查询是不合法：SELECT id,COUNT(\*) AS cnt FROM table\_name WHERE cnt > 0 GROUP BY id;   
  
WHERE语句被执行以确定哪些行应该包括GROUP BY部分中，而HAVING用来决定应该只用结果集合中的哪些行。   
  
24. 从关联的表中删除行   
因为MySQL不支持子选择或在DELETE语句中使用多个表，你应该使用下列方法从2个关联的表中删除行：   
在主表中基于某个WHERE条件SELECT行。   
在主表中基于相同的条件DELETE行。   
DELETE FROM related\_table WHERE related\_column IN (selected\_rows)   
如果在related\_column查询中的字符的全部数量超过1,048,576(缺省值max\_allowed\_packet），你应该分成更小的部分并且执行多个DELETE语句。如果related\_column是一个索引，你每次只删除100-1000个related\_column id将可能使得DELETE最快。如果related\_column不是一个索引，速度与IN子句中参数的数量无关。   
  
25. 解决没有匹配行的问题   
如果你有一个复杂的查询，涉及多个表，但没有返回任何行，你应该使用下列过程查找你的询问有什么不对：   
EXPLAIN测试查询并且检查你是否能找出显然是错误的一些东西。见7.22 EXPLAIN句法(得到关于一个SELECT的信息)。   
仅选择那些在WHERE子句中使用的字段。   
一次从查询中删除一个表，直到它返回一些行。如果表很大，对查询使用LIMIT 10是一个好主意。   
对应该已经匹配一行的列做一个SELECT，针对从询问中做后被删除的表。   
如果你将FLOAT或DOUBLE列与有小数的数字进行比较，你不能使用=!。这个问题在大多数计算机语言是常见的，因为浮点值不是准确的值。   
mysql> SELECT \* FROM table\_name WHERE float\_column=3.5;   
->   
mysql> SELECT \* FROM table\_name WHERE float\_column between 3.45 and 3.55;

在大多数情况下，将FLOAT改成一个DOUBLE将修正它！   
如果你仍然不能发现错误是什么，创建一个最小的可运行mysql test < query.sql的测试来显示你的问题。你可以用mysqldump --quick database tables > query.sql创建一个测试文件，在一个编辑器编辑文件，删除一些插入行(如果有太多这些语句)并且在文件末尾加入你的选择语句。测试你仍然有问题，可以这样做：   
shell> mysqladmin create test2   
shell> mysql test2 < query.sql   
  
使用mysqlbug的邮寄测试文件到mysql@lists.mysql.com。   
  
26. 与ALTER TABLE有关的问题   
如果ALTER TABLE死于这样一个错误：   
Error on rename of ’./database/name.frm’ to ’./database/B-a.frm’ (Errcode: 17)   
  
问题可能是MySQL在前一个ALTER TABLE中已经崩溃并且留下了一个名为“A-xxx”或“B-xxx”的老的数据库表。在这种情况下，到MySQL数据目录中并删除所有名字以A-或B-开始的文件。（你可以把他们移到别的地方而不是删除他们)。   
ALTER TABLE工作方式是：   
以要求的改变创建一个名为“A-xxx”的新表。   
从老表把所有行拷贝到“A-xxx”。   
老表被改名为“B-xxx”。   
“A-xxx”被改名为你的老表的名字。   
“B-xxx”被删除。   
如果某些改名操作出错，MySQL试图还原改变。如果出错严重(当然，这不应该发生。)，MySQL可能留下了老表为“B-xxx”但是一个简单改名就应该恢复你的数据。

27. 怎样改变一张表中列的顺序   
SQL的要点是中抽象应用程序以避免数据存储格式。你应该总是以你想要检索数据的意愿指定顺序。例如：   
SELECT col\_name1, col\_name2, col\_name3 FROM tbl\_name;   
  
将以col\_name1、col\_name2、col\_name3的顺序返回列，而：   
SELECT col\_name1, col\_name3, col\_name2 FROM tbl\_name;   
  
将以col\_name1、col\_name3、col\_name2的顺序返回列。   
在一个应用程序中，你应该决不基于他们的位置使用SELECT \* 检索列，因为被返回的列的顺序永远不能保证；对你的数据库的一个简单改变可能导致你的应用程序相当有戏剧性地失败。   
不管怎样，如果你想要改变列的顺序，你可以这样做：   
以正确的列顺序创建一张新表。   
执行INSERT INTO new\_table SELECT fields-in-new\_table-order FROM old\_table.   
删除或改名old\_table。   
ALTER TABLE new\_table RENAME old\_table。

28. 错误提示：

InnoDB: Unable to lock ./ibdata1, error: 11  
解决：  
ps -ef|grep mysqld ，杀死僵死进程  
然后重新启动mysql

29. 错误提示：  
error while loading shared libraries: libssl.so.0.9.8:

解决：  
更新db,openssl

pacman -S db ,openssl

这样就不会出现找不到连接库文件了

30. mysqld启动不了

错误提示：  
[root@philix lib]# /usr/sbin/mysqld 060715 21:16:36 [ERROR] Fatal error: Please read "Security" section of the manual to find out how to run mysqld as root! 060715 21:16:36 [ERROR] Aborting 060715 21:16:36 [Note] /usr/sbin/mysqld: Shutdown complete

解决办法:在/etc/my.conf中的mysqld加入

user=mysql

说明：这个错误本人今天刚好也碰到了，除了加入user=mysql，也可以试试用/usr/bin/mysqld\_safe --user=mysql & 来启动，注意mysql目录的权限。

31. [ERROR] Fatal error: Can't open and lock privilege tables: Table

解决办法

#mysql\_install\_db

解决mysql“Access denied for user ['root'@'localhost'](mailto:'root'@'localhost')”    
我的系统是ubuntu6.06，最近新装好的mysql在进入mysql工具时，总是有错误提示:  
# mysql -uroot -p  
Enter password:  
ERROR 1045 (28000): Access denied for user ['root'@'localhost'](mailto:'root'@'localhost') (using password: NO)

使用网上介绍的方法修改root用户的密码：  
# mysqladmin -uroot -p password 'newpassword'  
Enter password:  
mysqladmin: connect to server at 'localhost' failed  
error: 'Access denied for user ['root'@'localhost'](mailto:'root'@'localhost') (using password: YES)'

现在终于被我找到了解决方法，如下（请先测试方法三，谢谢！）：  
方法一：  
# /etc/init.d/mysql stop  
# mysqld\_safe --user=mysql --skip-grant-tables --skip-networking &  
# mysql -u root mysql  
mysql> UPDATE user SET Password=PASSWORD('newpassword') where USER='root';  
mysql> FLUSH PRIVILEGES;  
mysql> quit

# /etc/init.d/mysql restart  
# mysql -uroot -p  
Enter password: <输入新设的密码newpassword>

mysql>

方法二：  
直接使用/etc/mysql/debian.cnf文件中[client]节提供的用户名和密码:  
# mysql -udebian-sys-maint -p  
Enter password: <输入[client]节的密码>  
mysql> UPDATE user SET Password=PASSWORD('newpassword') where USER='root';  
mysql> FLUSH PRIVILEGES;  
mysql> quit

# mysql -uroot -p  
Enter password: <输入新设的密码newpassword>

mysql>

方法三：  
这种方法我没有进行过测试，因为我的root用户默认密码已经被我修改过了，那位有空测试一下，把结果告诉我，谢谢！！  
# mysql -uroot -p  
Enter password: <输入/etc/mysql/debian.cnf文件中[client]节提供的密码>

32. 问题： mysql DNS反解：skip-name-resolve  
 错误日志有类似警告：  
120119 16:26:04 [Warning] IP address '192.168.1.10' could not be resolved: Name or service not known

120119 16:26:04 [Warning] IP address '192.168.1.14' could not be resolved: Name or service not known

120119 16:26:04 [Warning] IP address '192.168.1.17' could not be resolved: Name or service not known

通过show processlist发现大量类似如下的连接：

|592|unauthenticated user|192.168.1.10:35320|NULL|Connect| |login|NULL|

|593|unauthenticated user|192.168.1.14:35321|NULL|Connect| |login|NULL|

|594|unauthenticated user|192.168.1.17:35322|NULL|Connect| |login|NULL|

skip-name-resolve 参数的作用：不再进行反解析（ip不反解成域名），这样可以加快数据库的反应时间。

 修改配置文件添加并需要重启：

[mysqld]   
skip-name-resolve

33. 问题错误日志：Error: Can't create a new thread (errno 12)

数据库服务器问题，数据库操作无法创建新线程。一般是有以下3个方面的原因：  
1）、MySQL 线程开得太多。  
2）、服务器系统内存溢出。  
3）、环境软件损坏或系统损坏。

【问题解决】

1）进入 phpmyadmin 的 mysql 数据库中的 user 表，对数据库的用户进行编辑，修改 max\_connections 的值。适当的改小一点。

2）联系服务器管理员检查服务器的内存和系统是否正常，如果服务器内存紧张，请检查一下哪些进程消耗了服务器的内存，同时考虑是否增加服务器的内存来提高整个系统的负载能力。

3）mysql版本更改为稳定版本

4）优化网站程序的sql等等

34.  操作报错：ERROR 1010 (HY000): Error dropping database

mysql> drop database xjtrace;  
ERROR 1010 (HY000): Error dropping database (can't rmdir './xjtrace/

在做数据库删除时出现这种提示，其原因是在database下面含有自己放进去的文件，譬如\*.txt文件或\*.sql文件等，只要进去把这个文件删了在执行。

mysql>drop database xjtrace;

Query OK, 0 rows affected (0.00 sec)

果断删除即可！！

35.导出数据很快，导入到新库时却很慢：

 MySQL导出的SQL语句在导入时有可能会非常非常慢，经历过导入仅400万条记录，竟用了近2个小时。在导出时合理使用几个参数，可以大大加快导入的速度。

 -e 使用包括几个VALUES列表的多行INSERT语法;  
--max\_allowed\_packet=XXX 客户端/服务器之间通信的缓存区的最大大小;  
--net\_buffer\_length=XXX TCP/IP和套接字通信缓冲区大小,创建长度达net\_buffer\_length的行

注意：max\_allowed\_packet和net\_buffer\_length不能比目标数据库的配置数值大，否则可能出错。

首先确定目标库的参数值

mysql> show variables like 'max\_allowed\_packet';

+--------------------+---------+

| Variable\_name | Value |

+--------------------+---------+

| max\_allowed\_packet | 1048576 |

+--------------------+---------+

1 row in set (0.00 sec)

mysql> show variables like 'net\_buffer\_length';

+-------------------+-------+

| Variable\_name | Value |

+-------------------+-------+

| net\_buffer\_length | 16384 |

+-------------------+-------+

1 row in set (0.00 sec)

根据参数值书写mysqldump命令，如：

mysql>mysqldump -uroot -p   数据库名 -e --max\_allowed\_packet=1048576 -net\_buffer\_length=16384 > SQL文件

例如：

mysql> mysqldump -uroot -p xjtrace -e --max\_allowed\_packet=1048576 --net\_buffer\_length=16384 >xjtrace\_data\_`date +%F`.sql

之前2小时才能导入的sql现在几十秒就可以完成了。

----------------------------------------------------------------------------------

MySQL Replication的常见错误

每个案例都是通过Last\_IO\_Errno/Last\_IO\_Error或者Last\_SQL\_Errno/Last\_SQL\_Error给出错误关键信息.  
  
        Last\_SQL\_Errno: 1677  
        Last\_SQL\_Error: Column 1 of table 'test.t' cannot be converted from type 'int' to type 'bigint(20)'  
        解决方法：这个案例是从网上找到的，自己动手实验了一把。从错误信息来看表面上是由于在slave上无法执行一条转换字段类型的SQL语句。实际上并不是有这种语句直接引起的，而是间接引起的（之前某些操作导致主从表字段类型不一致，接下来对这个表进行DML时就会报错）。它的意思是在对这个表t进行DML操作时，发现主从上表结果不一致，比如这里是说在主上t的字段1是int类型，但是从上t的字段1是bigint类型，所以报错。那么为什么要报错呢？这是从安全角度考虑，因为如果字段类型不一致可能会导致数据截断之类的问题。那么解决方法呢？通过参数slave\_type\_conversions进行控制，它有三种取值：  
ALL\_LOSSY：仅支持有损转换，什么叫有损？比如一个值本来是bigint存储为9999999999999，现在转换为int类型势必会要截断从而导致数据不一致。  
ALL\_NON\_LOSSY：仅支持无损转换，只能在无损的情况下才能进行转换  
ALL\_LOSSY,ALL\_NON\_LOSSY：有损/无算转换都支持  
空，即不设置这个参数：必须主从的字段类型一模一样。  
注意： 前面说的这几中情况都只在binlog\_format=ROW的情况下才有效。

        Last\_SQL\_Errno: 1194  
        Last\_SQL\_Error: Error 'Table 'traincenter' is marked as crashed and should be repaired' on query. Default database: 'basketballman'. Query: 'update traincenter set points='4',pointstime='1361912066'  where uid = '1847482697' limit 1'  
        解决方法：myisam表traincenter损坏，直接repair table即可。至于为什么myisam类型表比innodb更容易损坏，我觉得有两个原因：1，innodb有double write机制，损坏或者half write的页可以用它恢复，第二innodb是事务引擎，都有操作都是事务的，而myisam是非事务的，存在写一半但是操作终止情况。

        Last\_IO\_Errno: 1236  
        Last\_IO\_Error: Got fatal error 1236 from master when reading data from binary log: 'Could not find first log file name in binary log index file'  
        解决方法：主库上的binlog文件已经不存在但是在index file中确有相应记录存在。我这里发生这个错误的原因在于由于复制中断时间很长，报警出来一直没人处理，这个中断时间超过master上binlog超期时间，等恢复复制时需要的binlog已经由于其超期而被删掉，没办法只好重建这个实例了。以大家都要引以为戒。  
  
  
        Last\_IO\_Errno: 1593  
        Last\_IO\_Error: Fatal error: The slave I/O thread stops because master and slave have equal MySQL server ids; these ids must be different for replication to work (or the --replicate-same-server-id option must be used on slave but this does not always make sense; please check the manual before using it).  
        解决方法：主从配置的server-id一样，而在主从复制环境中server-id一样的binlog events都会被过滤掉。具体server-id的含义可以了解一下复制原理。这个一般是因为拷贝配置文件时忘记修改server-id导致，遇到这类问题也比较容易，平时操作谨慎一点即可。  
  
  
        Last\_Errno: 1053  
        Last\_Error: Query partially completed on the master (error on master: 1053) and was aborted. There is a chance that your master is inconsistent at this point. If you are sure that your master is ok, run this query manually on the slave and then restart the slave with SET GLOBAL SQL\_SLAVE\_SKIP\_COUNTER=1; START SLAVE; . Query: 'insert into ...  
        解决方法：查询在master上部分完成，然后终止了。这马上又能想到是myisam表，结果也正是这样。由于myisam不支持事务所以可能存在一个查询完成一部分然后失败的情况。解决方法一般也就是提示信息给出的跳过一个binlog event。不过确认跳过之前最好还是查询一下master上是否真的存在相应的记录，因为错误信息同时还会给出它认为在master上执行一部分然后终止的查询语句。  
  
  
        Last\_SQL\_Errno: 1666  
        Last\_SQL\_Error: Error executing row event: 'Cannot execute statement: impossible to write to binary log since statement is in row format and BINLOG\_FORMAT = STATEMENT.'   
        解决方法：这个案例的背景是做一个ABC结构的复制，B、C中设定的binlog\_format=statement，A中的是MIXED，所以当B尝试重做A过来的relay log，然后记录binlog（传给C）时发现relay log的binlog\_format与自己设定的binlog\_format不一致。我当时就是直接先更改BC的binlog\_format=MIXED解决。  
  
  
        Last\_Errno: 1032  
        Last\_Error: Could not execute Update\_rows event on table db.table; Can't find record in 'table', Error\_code: 1032; handler error HA\_ERR\_KEY\_NOT\_FOUND; the event's master log mysql-bin.000064, end\_log\_pos 158847  
        解决方法：这个是在binlog\_format=row复制下发生的。原因是因为row格式复制是最严格的，所以在mysql看来如果在从库上找不到要更新的这条记录，那么就代表主从数据不一致，因此报错。另外顺便说一句，对于row格式binlog，如果某个更新操作实际上并没有更新行，这个操作是不会记binlog的，因为row格式的binlog宗旨就是只记录发生了改变的行。所以这个解决办法根据你自己实际应用来定，最好的方法还是重做slave吧，这样更放心。  
  
  
        Last\_Errno: 28  
        Last\_Error: Error in Append\_block event: write to '/tmp/SQL\_LOAD-32343798-72213798-1.data' failed  
        解决方法： 首先说错误原因：主库执行load data infile，同步到从库后load data infile存放的文件默认是放在/tmp(由参数slave\_load\_tmpdir控制)，而/tmp空间不够因此报错。因此只要将从库上slave\_load\_tmpdir设置到一个磁盘空间足够大的分区就行。