# mysql -uroot -p"<OypmLdyt8k)"

mysql> set global validate\_password\_policy=0;

mysql>set global validate\_password\_length=6;

mysql> alter user user() identified by "123456";

删除授权用户

1 MySQL>删除用户在mysql库下user表里对应的记录;flush privileges;

删除授权用户

MySQL>drop user 用户名；

授权用户登录后，修改登录密码

mysql> SET PASSWORD=PASSWORD("新密码");

管理员修改授权用户的登录密码

mysql>set password for 用户名@'客户端地址'=password("新密码"); 必须要有括号

执行SET PASSWORD命令修改密码

这个其实与validate\_password\_policy的值有关，默认为1，所以刚开始设置的密码必须符合长度，且必须含有数字，小写或大写字母，特殊字符。如果我们不希望密码设置的那么复杂，需要修改两个全局参数：validate\_password\_policy与validate\_password\_length。validate\_password\_length默认值为8,最小值为4，如果你显性指定validate\_password\_length的值小于6，尽管不会报错，但validate\_password\_length的值将设为6。

可参考下列指令

mysql> set global validate\_password\_policy=0;

myql> set global validate\_password\_length=6;

mysql> set password for 'root'@'localhost'=password('123456');

数据备份策略？

完全备份：备份所有数据（一数据库服务器、一个库 、一张

表）

差异备份：备份自完全备份后，所有新产生的数据。

增量备份：备份自上一次备份后，所有新产生的数据。

（完全、增量、差异）

4在生产环境下常用的备份策略？

单独完全备份

完全备份+差异备份

完全备份+增量备份

手动创建新的binlog日志文件

mysql> flush logs;

# systemctl start mysqld

#mysql -uroot -p123123 -e "flush logs"

#mysqldump -uroot -p123123 --flush-logs 库名 >

/opt/db2.sql

查看当前正在使用的日志文件

mysql> show master status;

删除已有的binlog日志文件

mysql> purge master logs to "binlog文件名";

mysql> purge master logs to "plj.000004";

mysql> reset master;

遇到的问题;

1. 安装完后mysql起不来
2. 先检查配置文件有没有细微的标点符号缺漏 是否写正确 路径是否正常
3. 检查有没有少安装些依赖性的包或者装错包
4. 权限是否有给，如果没有给相应的权限而去加载，会导致加载不进去，造成报错起不来，

如：chown **-**R mysql**:**mysql **/var**/lib/mysql

chown –R mysql:mysql /logdir/ 这个/logdir/是自己创建的，权限属于root 用mysql加载会加载权限不足

2.通过XtraBackup工具备份的数据库目录，若要恢复到另一个MySQL服务器，需要先做一个“--apply-log（准备还原（回滚日志）） --redo-only（日志回滚合并） ”的准备操作。

（最后一次日志增量回滚不需要--redo-only）

| **备份方法** | **备份速度** | **恢复速度** | **便捷性** | **功能** | **一般用于** |
| --- | --- | --- | --- | --- | --- |
| cp | 快 | 快 | 一般、灵活性低 | 很弱 | 少量数据备份 |
| mysqldump | 慢 | 慢 | 一般、可无视存储引擎的差异 | 一般 | 中小型数据量的备份 |
| lvm2快照 | 快 | 快 | 一般、支持热备、速度快 | 一般 | 中小型数据量的备份 |
| xtrabackup | 较快 | 较快 | 实现innodb热备、对存储引擎有要求 | 强大 | 较大规模的备份 |

其实我们还可以通过Master-Slave Replication 进行备份

* MyISAM

 热备 ×

 温备 √

 冷备 √

* InnoDB

 热备 √

 温备 √

 冷备 √

同步不了：

Slave\_IO\_Running: Connecting

Slave\_SQL\_Running: Yes

Last\_IO\_Error: error connecting to master 'replicater@192.168.4.1:3306' - retry-time: 60 retries: 2

原因：在从数据库中把主数据库的IP指错了, MASTER\_USER='replicater', 前后多了空格，密码要用replicater指定的对应密码pwd123，而不是混淆了数据库root密码123456。更改的时候先mysql> stop slave;

CHANGE MASTER TO  MASTER\_HOST='192.168.4.3',

MASTER\_USER='replicater',

MASTER\_PASSWORD='pwd123',

MASTER\_LOG\_FILE='ldc.000001',  //对应Master的日志文件

MASTER\_LOG\_POS=453; //对应Master的日志偏移位置

导致lave\_IO\_Running 为connecting 的原因主要有以下 3 个方面：

1、网络不通

2、密码不对

3、pos不对

[[MySQL生产环境复制故障修复]](http://blog.csdn.net/mchdba/article/details/12994151) Last\_IO\_Error: error connecting to master 'replicater @192.168.4.3:3306' - retry-time: 60 retries: 1

**1  修复集群，start slave 报错如下：**

1. mysql> stop slave;
2. Query OK, 0 **rows** affected (0.10 sec)
3. -- 重新设置复制参数
4. mysql> change master **to** master\_host='10.24.54.18',master\_port=3306,master\_user='replication',master\_password='xxxxxx';
5. Query OK, 0 **rows** affected, 0 warnings (0.40 sec)
7. mysql> start slave;
8. Query OK, 0 **rows** affected (0.01 sec)
10. mysql>  show slave status\G;
11. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
12. Slave\_IO\_State: Connecting **to** master
13. Master\_Host: 10.24.54.18
14. Master\_User: replication
15. Master\_Port: 3306
16. Connect\_Retry: 60
17. Master\_Log\_File: mysql-bin.000002
18. Read\_Master\_Log\_Pos: 1355028
19. Relay\_Log\_File: mysql-relay-bin.000001
20. Relay\_Log\_Pos: 4
21. Relay\_Master\_Log\_File: mysql-bin.000002
22. Slave\_IO\_Running: Connecting
23. Slave\_SQL\_Running: Yes
24. Replicate\_Do\_DB:
25. Replicate\_Ignore\_DB:
26. Replicate\_Do\_Table:
27. Replicate\_Ignore\_Table:
28. Replicate\_Wild\_Do\_Table:
29. Replicate\_Wild\_Ignore\_Table:
30. Last\_Errno: 0
31. Last\_Error:
32. Skip\_Counter: 0
33. Exec\_Master\_Log\_Pos: 1355028
34. Relay\_Log\_Space: 107
35. Until\_Condition: None
36. Until\_Log\_File:
37. Until\_Log\_Pos: 0
38. Master\_SSL\_Allowed: **No**
39. Master\_SSL\_CA\_File:
40. Master\_SSL\_CA\_Path:
41. Master\_SSL\_Cert:
42. Master\_SSL\_Cipher:
43. Master\_SSL\_Key:
44. Seconds\_Behind\_Master: NULL
45. Master\_SSL\_Verify\_Server\_Cert: **No**
46. Last\_IO\_Errno: 1045
47. Last\_IO\_Error: error connecting **to** master 'replication@10.24.54.18:3306' - retry-**time**: 60  retries: 86400
48. Last\_SQL\_Errno: 0
49. Last\_SQL\_Error:
50. Replicate\_Ignore\_Server\_Ids:
51. Master\_Server\_Id: 2
52. 1 row in **set** (0.00 sec)
54. <pre **name**="code" class="sql">-- 报错如下： Last\_IO\_Errno: 1045</pre>ERROR: No query specified

看问题，一开始怀疑是密码问题，但是密码都是复制上来的，没有变动，问题不大，还是看看check下别的情况吧。

**2 ，check问题所在，ping ip地址，是OK的。**

1. [xxx@xxx-c004db1 shared]$ ping 10.24.54.18
2. PING 10.24.54.18 (10.24.54.18) 56(84) bytes of data.
3. 64 bytes **from** 10.24.54.18: icmp\_seq=1 ttl=64 time=0.321 ms
4. 64 bytes **from** 10.24.54.18: icmp\_seq=2 ttl=64 time=0.370 ms
5. ^C
6. --- 10.24.54.18 ping statistics ---
7. 2 packets transmitted, 2 received, 0% packet loss, time 1417ms
8. rtt min/avg/max/mdev = 0.321/0.345/0.370/0.030 ms
9. [xxx@xxx-c004db1 shared]$

**3 ，access 数据库端口3306，也是OK的。**

1. [xxx@xxx-c004db1 shared]$ ssh -v -p3306 10.24.54.18
2. OpenSSH\_5.3p1, OpenSSL 1.0.0-fips 29 Mar 2010
3. debug1: Reading configuration data /etc/ssh/ssh\_config
4. debug1: Applying options **for** \*
5. debug1: Connecting to 10.24.54.18 [10.24.54.18] port 3306.
6. debug1: Connection established.
7. debug1: identity file /home/xxx/.ssh/xxx type -1
8. debug1: identity file /home/xxx/.ssh/id\_rsa type -1
9. debug1: identity file /home/xxx/.ssh/id\_dsa type -1
10. debug1: ssh\_exchange\_identification: O
11. ^C

**4，关闭防火墙service iptables stop，然后重启slave，还是报 Last\_IO\_Errno: 1045错误。**

**5，远程访问主库mysql 服务，报错如下：**

1. [xxx@xxx-c004db1 shared]$ mysql -h 10.24.54.18 -ureplication -pxxx
2. ERROR 1045 (28000): Access denied **for** user 'replication'@'xxx-xxx.abn-sjl.ea.com' (using password: YES)
3. [xxx@xxx-c004db1 shared]$

仔细检查帐户，发现原来是执行grant语句的时候，密码串里面多了一个空格导致，同事的文本编辑器里面空格显示不明显，所以没有发现。

**6，修复：去掉空格，重新执行grant语句，再重新启动slave，OK。**

1. mysql> show slave status\G
2. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
3. Slave\_IO\_State: Waiting **for** master **to** send event
4. Master\_Host: 10.24.54.18
5. Master\_User: replication
6. Master\_Port: 3306
7. Connect\_Retry: 60
8. Master\_Log\_File: mysql-bin.000660
9. Read\_Master\_Log\_Pos: 34886442
10. Relay\_Log\_File: mysql-relay-bin.001320
11. Relay\_Log\_Pos: 253
12. Relay\_Master\_Log\_File: mysql-bin.000660
13. Slave\_IO\_Running: Yes
14. Slave\_SQL\_Running: Yes
15. Replicate\_Do\_DB:
16. Replicate\_Ignore\_DB:
17. Replicate\_Do\_Table:
18. Replicate\_Ignore\_Table:
19. Replicate\_Wild\_Do\_Table:
20. Replicate\_Wild\_Ignore\_Table:
21. Last\_Errno: 0
22. Last\_Error:
23. Skip\_Counter: 0
24. Exec\_Master\_Log\_Pos: 34886442
25. Relay\_Log\_Space: 452
26. Until\_Condition: None
27. Until\_Log\_File:
28. Until\_Log\_Pos: 0
29. Master\_SSL\_Allowed: **No**
30. Master\_SSL\_CA\_File:
31. Master\_SSL\_CA\_Path:
32. Master\_SSL\_Cert:
33. Master\_SSL\_Cipher:
34. Master\_SSL\_Key:
35. Seconds\_Behind\_Master: 0
36. Master\_SSL\_Verify\_Server\_Cert: **No**
37. Last\_IO\_Errno: 0
38. Last\_IO\_Error:
39. Last\_SQL\_Errno: 0
40. Last\_SQL\_Error:
41. Replicate\_Ignore\_Server\_Ids:
42. Master\_Server\_Id: 2
43. 1 row in **set** (0.00 sec)

**总结：DBA最重要的一点：细心，与大家共勉**

读写分析遇到的问题：proxy分离解析监听不到主服务器数据库的端口3306，从服务器可以监听的到，

把主服务器防火墙 关了，setenforce 0 ，重新启动proxy分析器 就可以监听了

[root@node1 ~]# kill -9 5683 关掉proxy进程

[root@node1 ~]# maxscale -f /etc/a.cnf 启动proxy

[root@node1 ~]# netstat -antplu | grep maxscale

tcp 0 0 192.168.4.1:45486 192.168.4.2:3306 ESTABLISHED 2866/maxscale

tcp 0 0 192.168.4.1:43846 192.168.4.3:3306 ESTABLISHED 2866/maxscale

tcp6 0 0 :::6606 :::\* LISTEN 2866/maxscale

tcp6 0 0 :::4006 :::\* LISTEN 2866/maxscale

[Shell脚本语法-- if/then/elif/else/fi](http://blog.csdn.net/ljchlx/article/details/18263397)

**#! /bin/sh**

**if [ -f /bin/bash ]**

**then echo "/bin/bash is a file"**

**else echo "/bin/bash is NOT a file"**

**fi**

**if :; then echo "always true"; fi**

**:是一个特殊的命令，称为空命令，该命令不做任何事，但Exit Status总是真。此外，也可以执行/bin/true或/bin/false得到真或假的Exit Status。再看一个例子：**

**#! /bin/sh**

**echo "Is it morning? Please answer yes or no."**

**read YES\_OR\_NO**

**if [ "$YES\_OR\_NO" = "yes" ]; then**

**echo "Good morning!"**

**elif [ "$YES\_OR\_NO" = "no" ]; then**

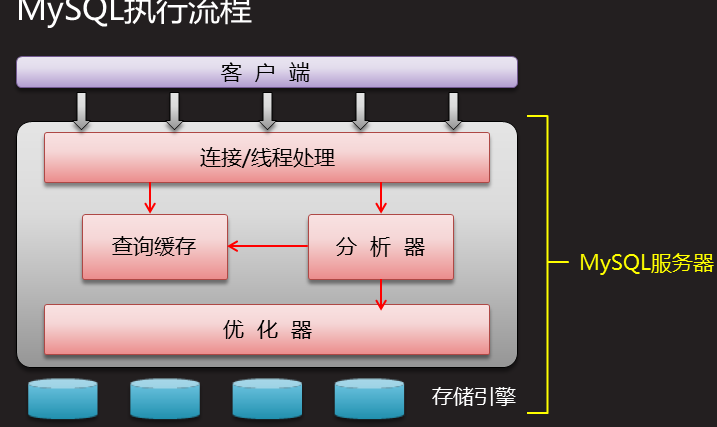
**echo "Good afternoon!"**

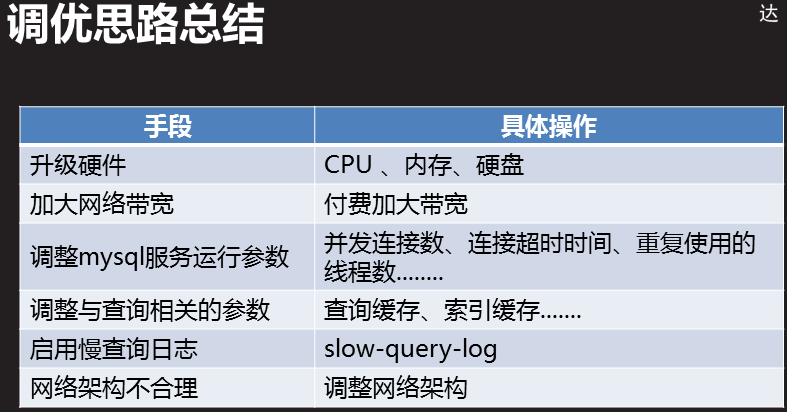
**else**

**echo "Sorry, $YES\_OR\_NO not recognized. Enter yes or no."**

**exit 1**

**fi**





81 88 23 99 100

(master81 master88 slave23 slave99 mmm-monitor )

