









































1 PXC集群 71 72 73

1pxc软件介绍

2pxc特点

3拓扑结构

4构建pxc集群

4.1 在所有服务器安装pxc软件（71/72/73）

]#cd pxc

]# rpm -ivh libev-4.15-1.el6.rf.x86\_64.rpm //安装依赖

]# yum -y install percona-xtrabackup-24-2.4.13-1.el7.x86\_64.rpm

]# rpm -ivh qpress-1.1-14.11.x86\_64.rpm //安装依赖

]# tar -xvf Percona-XtraDB-Cluster-5.7.25-31.35-r463-el7-x86\_64-bundle.tar

]# yum -y install Percona-XtraDB-Cluster-\*.rpm

]# ls /etc/percona-xtradb-cluster.conf.d/

mysqld.cnf mysqld\_safe.cnf wsrep.cnf

4.2 在3台数据库服务器分别修改配置文件

]# cd /etc/percona-xtradb-cluster.conf.d/

71】# vim mysqld.cnf

server-id=71

:wq

72】#vim mysqld.cnf

server-id=72

:wq

73】# vim mysqld.cnf

server-id=73

:wq

71】# vim wsrep.cnf

8 wsrep\_cluster\_address=gcomm://192.168.4.71,192.168.4.72,192.168.4.73

25 wsrep\_node\_address=192.168.4.71

27 wsrep\_cluster\_name=pxc-cluster

30 wsrep\_node\_name=pxc-cluster-node-71

39 wsrep\_sst\_auth="sstuser:123qqq...A"

:wq

72】#vim wsrep.cnf

8 wsrep\_cluster\_address=gcomm://192.168.4.71,192.168.4.72,192.168.4.73

25 wsrep\_node\_address=192.168.4.72

27 wsrep\_cluster\_name=pxc-cluster

30 wsrep\_node\_name=pxc-cluster-node-72

39 wsrep\_sst\_auth="sstuser:123qqq...A"

:wq

73】# vim wsrep.cnf

8 wsrep\_cluster\_address=gcomm://192.168.4.71,192.168.4.72,192.168.4.73

25 wsrep\_node\_address=192.168.4.73

27 wsrep\_cluster\_name=pxc-cluster

30 wsrep\_node\_name=pxc-cluster-node-73

39 wsrep\_sst\_auth="sstuser:123qqq...A"

:wq

4.3 在任意1台数据库服务器初始化集群

注意：关闭3台数据库服务器的 selinux 和 firewalld

[root@pxcnode71 ~]#systemctl start mysql@bootstrap.service

[root@pxcnode71 ~]# netstat -utnlp | grep 3306

tcp6 0 0 :::3306 :::\* LISTEN 8502/mysqld

[root@pxcnode71 ~]# netstat -utnlp | grep 4567

tcp 0 0 0.0.0.0:4567 0.0.0.0:\* LISTEN 8502/mysqld

[root@pxcnode71 ~]# ls /var/log/mysqld.log

/var/log/mysqld.log

[root@pxcnode71 ~]# grep password /var/log/mysqld.log

2020-08-26T14:22:55.211844Z 1 [Note] A temporary password is generated for root@localhost: qkJwa%qAb0N=

[root@pxcnode71 ~]# mysql -uroot -p'qkJwa%qAb0N='

mysql> alter user root@"localhost" identified by "123456";

Query OK, 0 rows affected (0.08 sec)

mysql> show databases;

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

| Database |

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

| information\_schema |

| mysql |

| performance\_schema |

| sys |

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

4 rows in set (0.00 sec)

mysql> grant reload,lock tables,replication client,process on \*.\* to sstuser@"localhost" identified by "123qqq...A";

mysql> exit

Bye

[root@pxcnode71 ~]# mysql -uroot -p123456

mysql> show databases;

mysql> exit

[root@pxcnode71 ~]# mysql -usstuser -p123qqq...A #测试授权用户sstuser

4.4 启动另外2台主机的数据库服务

[root@pxcnode72 ~]#systemctl start mysql

4.5 查看数据库服务器状态

[root@pxcnode72 ~]# yum -y install net-tools

[root@pxcnode72 ~]#netstat -utnlp | grep 3306

[root@pxcnode72 ~]#netstat -utnlp | grep 4567

[root@pxcnode72 ~]#mysql -uroot -p123456

mysql> select user from mysql.user where user="sstuser";

[root@pxcnode73 ~]# yum -y install net-tools

[root@pxcnode73 ~]#netstat -utnlp | grep 3306

[root@pxcnode73~]#netstat -utnlp | grep 4567

[root@pxcnode73 ~]#mysql -uroot -p123456

mysql> select user from mysql.user where user="sstuser";

4.6 查看主机的集群状态

> show status like "%wsrep%";

排错方法：

查看日志报错信息 grep -i "error" /var/log/mysqld.log

启动失败的原因：

检查修改的配置文件 mysqld.conf 和 wsrep.conf

没有关闭所有数据库服务器的 selinux 和 firewalld

重新初始化集群

]#rm -rf /var/lib/mysql/\*

]#systemctl start mysql@bootstrap.service

启动mysql服务

]#rm -rf /var/lib/mysql/\*

]#systemctl start mysql

71 初始化失败 解决 办法

71]#

rm -rf /var/lib/mysql/\*

rm -rf /etc/percona-xtradb-cluster.conf.d/mysqld\_safe.cnf

systemctl start mysql@bootstrap.service

或者

先把mysql的进程杀了，然后把/var/lib/mysql下面又清了一遍，删了一下mysqld\_safe.cnf 再执行 初始化

4.7 客户端连接集群中的任意主机存取数据

4.7.1 在任意1台数据库服务器添加用户给客户端连接使用，另外2台会自动同步添加的用户到本机

[root@pxcnode73 ~]# mysql -uroot -p123456 -e 'grant all on bbsdb.\* to yaya@"%" identified by "123456"'

[root@pxcnode71 ~]# mysql -uroot -p123456 -e 'show grants for yaya@"%"'

mysql: [Warning] Using a password on the command line interface can be insecure.

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

| Grants for yaya@% |

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

| GRANT USAGE ON \*.\* TO 'yaya'@'%' |

| GRANT ALL PRIVILEGES ON `bbsdb`.\* TO 'yaya'@'%' |

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

[root@pxcnode71 ~]#

[root@pxcnode72 ~]# mysql -uroot -p123456 -e 'show grants for yaya@"%"'

mysql: [Warning] Using a password on the command line interface can be insecure.

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

| Grants for yaya@% |

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

| GRANT USAGE ON \*.\* TO 'yaya'@'%' |

| GRANT ALL PRIVILEGES ON `bbsdb`.\* TO 'yaya'@'%' |

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

[root@pxcnode71 ~]#

[root@host50 ~]# mysql -h192.168.4.71 -uyaya -p123456 -e 'create database bbsdb'

[root@host50 ~]# mysql -h192.168.4.71 -uyaya -p123456 -e 'create table bbsdb.b(id int primary key auto\_increment , name char(10))'

[root@host50 ~]# mysql -h192.168.4.72 -uyaya -p123456 -e 'insert into bbsdb.b(name)values("bob")'

[root@host50 ~]# mysql -h192.168.4.73 -uyaya -p123456 -e 'insert into bbsdb.b(name)values("tom")'

[root@host50 ~]#

[root@host50 ~]# mysql -h192.168.4.71 -uyaya -p123456 -e 'select \* from bbsdb.b'

4.8 测试集群的高可用

停止集群中任意1台主机的数据库服务，都不会影响到另外2台主机

[root@pxcnode71 ~]# systemctl stop mysql@bootstrap.service

[root@pxcnode72 ~]# systemctl stop mysql

[root@host50 ~]# mysql -h192.168.4.73 -uyaya -p123456 -e 'select \* from bbsdb.b'

[root@host50 ~]# mysql -h192.168.4.73 -uyaya -p123456 -e 'insert into bbsdb.b(name)values("aaaa")'

4.9 测试服务器自动恢复

[root@pxcnode71 ~]# systemctl start mysql

[root@pxcnode72 ~]# systemctl start mysql

[root@host50 ~]# mysql -h192.168.4.71 -uyaya -p123456 -e 'select \* from bbsdb.b'

[root@host50 ~]# mysql -h192.168.4.72 -uyaya -p123456 -e 'select \* from bbsdb.b'

2 MySQL存储引擎（难点）理解

2.1 什么是存储引擎（存储引擎介绍） 功能组件之一，访问表是希望使用MySQL的什么功能对表做处理，就调用对应的存储引擎就可以了。

innodb

2 2 MySQL服务的体系结构

2 2 MySQL服务的体系结构（由8部分组成）

组件的名称和功能介绍

1 连接池 检查硬件资源 和 用户访问权限

2 管理工具 安装mysql服务软件后自带的安装命令 mysql mysqladmin mysqldump mysqlbinlog

3 sql接口 把执行的SQL命令 发送给mysql服务的进程处理

4 分析器 分析执行的SQL命令的语法格式和访问的数据是否有权限

5 优化器 对执行的SQL命令优化处理

6 缓存空间 查询缓存，存储曾经查找过的数据，存储空间从物理划分的

默认 MySQL服务查询缓存没有开启。

7 存储引擎，访问表中的数据时，会根据表使用的存储引擎，对数据做不同的处理，每种存储引擎都有各自的功能和数据存储方式。建表时可以根据需要指定表使用的存储引擎。（存储引擎，就是MySQL服务提供的各自功能）

8 文件系统 数据库服务器的硬盘（文件系统类型 ext2/3/4 、 xfs ）

2.3 MySQL服务处理查询访问的过程

1 客户端 ------> 数据库服务器

2 执行select 命令

3 MySQL服务 先访问 查询缓存 如果有数据 直接在查询缓存提前数据回复客户端，反之， 到硬盘里查找数据，把找到的数据 先 保存到查询缓存 再回复给客户端.

2.4 存储引擎管理

查看数据库服务默认使用的存储引擎

mysql> show engines;

查看已有表使用的存储引擎

mysql> show create table 库.表 \G

mysql> show create table db1.t10 \G

修改数据库服务默认使用的存储引擎

]# vim /etc/my.cnf

[mysqld]

default-storage-engine="myisam"

:wq

[root@host50 ~]# systemctl restart mysqld

[root@host50 ~]# mysql -uroot -p123qqq...A -e ' create table db1.a2(id int)'

[root@host50 ~]# mysql -uroot -p123qqq...A -e 'show create table db1.a2 \G'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Table: a2

Create Table: CREATE TABLE `a2` (

`id` int(11) DEFAULT NULL

) ENGINE=MyISAM DEFAULT CHARSET=latin1

[root@host50 ~]#

修改已有表使用的存储引擎

alter table table 库.表 engine=新存储引擎名;

建表时，设置表使用的存储引擎

create table 库.表（字段列表）engine=存储引擎名;

mysql> create table db1.c1 (id int )engine=innodb;

执行查询select数据和存储insert数据的命令时都会对表做访问，根据表使用的存储引擎做不同的处理，每种存储引擎有各自的功能和数据存储方式。

2.5 常用存储引擎特点介绍 (理论)

2.5.1 myisam特点

支持表级锁

不支持外键 事务 事务回滚

每张对应3个表文件

.frm .MYI .MYD

表结构 index 数据

desc 库.表 createt index select \* from 库.表

2.5.2 innodb特点

支持外键 支持事务 支持事务回滚 支持行级锁

每张对应2个表文件

.frm .ibd

表结构 index+数据

desc 库.表

专业术语？

锁 : 加锁的目的 是为了解决 数据并发访问的冲突问题。

锁粒度：表级锁 行级锁

表级锁：只要是对表做select 或insert 访问 会把整张表锁上，锁了之后不允许其他的任何访问，只有锁被释放后才会允许其他的访问

行级锁：对表做select 或insert 访问，仅仅对被访问的行分别加锁，没有被访问的行，不会加所有，允许其他的连接访问。

锁类型：读锁 和 写锁

读锁：对表执行select 访问 时加读锁，又叫共享锁，如果在加读锁期间有其他客户端也对表执行select 访问，是允许的，但insert 不允许。

写锁： 对表做insert 访问时加写锁。又叫排它锁 ，如果在加写锁期间不允许任何连接对表做select 或insert。

pc1 update db1.user set name="xxx" where id=1;

pc2 update db1.user set name="yyy" where id=1;

事务？ 连接数据库服务成功 各种执行各种操作命令 断开连接

库.表(innodb) 0 1 2 3 4

使用事务日志文件记录对innodb存储引擎的表执行的sql操作。

cd /var/lib/mysql/

ib\_logfile0 ib\_logfile1 ibdata1

事务回滚: 在执行的各种操作命令时，任意一步操作失败，把数据恢复到失败之前的状态。

事务的特点(ACID)

原子性 一致性 隔离性 持久性

事务特点演示？

连接1

create table db3.user2(id int)engine=innodb;

set autocommit=off;

insert into db3.user2 values(111);

select \* from db3.user2; 自己能看到

连接2

select \* from db3.user2; 看不到

连接3

select \* from db3.user2;看不到

连接1 commit; 提交

连接2

select \* from db3.user2; 看到

连接3

select \* from db3.user2;看到

连接1

delete from db3.user2;

连接2

select \* from db3.user2; 看到

连接3

select \* from db3.user2;看到

连接1

insert into db3.user2 values(8888);

连接2

select \* from db3.user2; 看不到

连接3

select \* from db3.user2;看不到

连接1 select \* from db3.user2; 看到

rollback；

select \* from db3.user2; 看到 插入和删除 之前的数据

连接1 delete from db3.user2;

commit;

连接2

select \* from db3.user2; 看不到

连接3

select \* from db3.user2;看不到

2.6 建表时，如何决定表使用哪种存储引擎？

接受查询select访问多的表，适合使用myisam存储引擎，节省系统的硬件资源

接受写(insert update delete)访问多的表，适合使用innodb存储引擎,并发访问量大。

**NSD RDBMS2 DAY05**

1. [案例1：安装软件](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/RDBMS2/DAY05/CASE/01/index.html#case1)
2. [案例2：配置服务](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/RDBMS2/DAY05/CASE/01/index.html#case2)
3. [案例3：测试配置](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/RDBMS2/DAY05/CASE/01/index.html#case3)
4. [案例4：MySQL存储引擎的配置](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/RDBMS2/DAY05/CASE/01/index.html#case4)
5. [案例5：事务特性](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN202001/RDBMS2/DAY05/CASE/01/index.html#case5)

**1 案例1：安装软件**

**1.1 问题**

* 环境准备
* 安装软件包

**1.2 方案**

准备3台虚拟主机，配置ip地址和主机名。具体如图-1所示：（不需要安装任何MySQL服务软件）



图-1

**1.3 步骤**

实现此案例需要按照如下步骤进行。

步骤一：环境准备

配置主机名与ip地址绑定

配置服务器192.168.4.71

1. ]# vim /etc/hosts
2. 192.168.4.71     pxcnode71
3. 192.168.4.72     pxcnode72
4. 192.168.4.73     pxcnode73
5. :wq
6. ]#hostname pxcnode71

配置服务器192.168.4.72

1. ]# vim /etc/hosts
2. 192.168.4.71     pxcnode71
3. 192.168.4.72     pxcnode72
4. 192.168.4.73     pxcnode73
5. :wq
6. ]#hostname pxcnode72

配置服务器192.168.4.73

1. ]# vim /etc/hosts
2. 192.168.4.71     pxcnode71
3. 192.168.4.72     pxcnode72
4. 192.168.4.73     pxcnode73
5. :wq
6. ]#hostname pxcnode73

在任意一台服务器上ping 对方的主机名，ping通为配置成功。

1. [root@host71 ~]# ping -c 2 pxcnode71 //成功
2. PING pxcnode71 (192.168.4.71) 56(84) bytes of data.
3. 64 bytes from pxcnode71 (192.168.4.71): icmp\_seq=1 ttl=255 time=0.011 ms
4. 64 bytes from pxcnode71 (192.168.4.71): icmp\_seq=2 ttl=255 time=0.020 ms
5. --- pxcnode71 ping statistics ---
6. 2 packets transmitted, 2 received, 0% packet loss, time 999ms
7. rtt min/avg/max/mdev = 0.011/0.015/0.020/0.006 ms
8. [root@host71 ~]#
9. [root@host71 ~]#
10. [root@host71 ~]# ping -c 2 pxcnode72 //成功
11. PING pxcnode72 (192.168.4.72) 56(84) bytes of data.
12. 64 bytes from pxcnode72 (192.168.4.72): icmp\_seq=1 ttl=255 time=0.113 ms
13. 64 bytes from pxcnode72 (192.168.4.72): icmp\_seq=2 ttl=255 time=0.170 ms
14. --- pxcnode72 ping statistics ---
15. 2 packets transmitted, 2 received, 0% packet loss, time 1000ms
16. rtt min/avg/max/mdev = 0.113/0.141/0.170/0.030 ms
17. [root@host71 ~]#
18. [root@host71 ~]#
19. [root@host71 ~]# ping -c 2 pxcnode73 //成功
20. PING pxcnode73 (192.168.4.73) 56(84) bytes of data.
21. 64 bytes from pxcnode73 (192.168.4.73): icmp\_seq=1 ttl=255 time=0.198 ms
22. 64 bytes from pxcnode73 (192.168.4.73): icmp\_seq=2 ttl=255 time=0.155 ms
23. --- pxcnode73 ping statistics ---
24. 2 packets transmitted, 2 received, 0% packet loss, time 1000ms
25. rtt min/avg/max/mdev = 0.155/0.176/0.198/0.025 ms
26. [root@host71 ~]#

步骤二：安装软件包

1）在192.168.4.71 服务器安装软件包

软件包之间有依赖注意软件包安装顺序

1. ]# rpm -ivh libev-4.15-1.el6.rf.x86\_64.rpm //安装依赖
2. ]# yum -y install percona-xtrabackup-24-2.4.13-1.el7.x86\_64.rpm
3. ]# rpm -ivh qpress-1.1-14.11.x86\_64.rpm //安装依赖
4. ]# tar -xvf Percona-XtraDB-Cluster-5.7.25-31.35-r463-el7-x86\_64-bundle.tar
5. ]# yum -y install Percona-XtraDB-Cluster-\*.rpm

2）在192.168.4.72 服务器安装软件包

1. ]# rpm -ivh libev-4.15-1.el6.rf.x86\_64.rpm //安装依赖
2. ]# yum -y install percona-xtrabackup-24-2.4.13-1.el7.x86\_64.rpm
3. ]# rpm -ivh qpress-1.1-14.11.x86\_64.rpm //安装依赖
4. ]# tar -xvf Percona-XtraDB-Cluster-5.7.25-31.35-r463-el7-x86\_64-bundle.tar
5. ]# yum -y install Percona-XtraDB-Cluster-\*.rpm

3）在192.168.4.73 服务器安装软件包

1. ]# rpm -ivh libev-4.15-1.el6.rf.x86\_64.rpm //安装依赖
2. ]# yum -y install percona-xtrabackup-24-2.4.13-1.el7.x86\_64.rpm
3. ]# rpm -ivh qpress-1.1-14.11.x86\_64.rpm //安装依赖
4. ]# tar -xvf Percona-XtraDB-Cluster-5.7.25-31.35-r463-el7-x86\_64-bundle.tar
5. ]# yum -y install Percona-XtraDB-Cluster-\*.rpm

**2 案例2：配置服务**

**2.1 问题**

* 修改mysqld.cnf文件
* 修改mysqld\_safe.cnf文件
* 修改wsrap.cnf文件
* 启动服务

**2.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：修改mysqld.cnf文件

1. [root@pxcnode71 ~]# vim /etc/percona-xtradb-cluster.conf.d/mysqld.cnf
2. [mysqld]
3. server-id=71                 //server-id 不允许重复
4. datadir=/var/lib/mysql             //数据库目录
5. socket=/var/lib/mysql/mysql.sock     //socket文件
6. log-error=/var/log/mysqld.log        //日志文件
7. pid-file=/var/run/mysqld/mysqld.pid    //pid文件
8. log-bin                    //启用binlog日志
9. log\_slave\_updates            //启用链式复制
10. expire\_logs\_days=7            //日志文件保留天数
11. :wq

修改服务器192.168.4.72

1. [root@pxcnode72 ~]# vim /etc/percona-xtradb-cluster.conf.d/mysqld.cnf
2. [mysqld]
3. server-id=72                 //server-id 不允许重复
4. datadir=/var/lib/mysql             //数据库目录
5. socket=/var/lib/mysql/mysql.sock     //socket文件
6. log-error=/var/log/mysqld.log        //日志文件
7. pid-file=/var/run/mysqld/mysqld.pid    //pid文件
8. log-bin                    //启用binlog日志
9. log\_slave\_updates            //启用链式复制
10. expire\_logs\_days=7            //日志文件保留天数
11. :wq

修改服务器192.168.4.73

1. [root@pxcnode73 ~]# vim /etc/percona-xtradb-cluster.conf.d/mysqld.cnf
2. [mysqld]
3. server-id=73                 //server-id 不允许重复
4. datadir=/var/lib/mysql             //数据库目录
5. socket=/var/lib/mysql/mysql.sock     //socket文件
6. log-error=/var/log/mysqld.log        //日志文件
7. pid-file=/var/run/mysqld/mysqld.pid    //pid文件
8. log-bin                    //启用binlog日志
9. log\_slave\_updates            //启用链式复制
10. expire\_logs\_days=7            //日志文件保留天数
11. :wq

步骤二：修改mysqld\_safe.cnf文件

1）分别修改3台服务器的mysqld\_safe.cnf （使用默认配置即可）

1. [root@pxcnode71 ~]# vim /etc/percona-xtradb-cluster.conf.d/mysqld\_safe.cnf
2. [mysqld\_safe]
3. pid-file = /var/run/mysqld/mysqld.pid
4. socket = /var/lib/mysql/mysql.sock
5. nice = 0
6. :wq

修改服务器192.168.4.72

1. [root@pxcnode72 ~]# vim /etc/percona-xtradb-cluster.conf.d/mysqld\_safe.cnf
2. [mysqld\_safe]
3. pid-file = /var/run/mysqld/mysqld.pid
4. socket = /var/lib/mysql/mysql.sock
5. nice = 0
6. :wq

修改服务器192.168.4.73

1. [root@pxcnode73 ~]# vim /etc/percona-xtradb-cluster.conf.d/mysqld\_safe.cnf
2. [mysqld\_safe]
3. pid-file = /var/run/mysqld/mysqld.pid
4. socket = /var/lib/mysql/mysql.sock
5. nice = 0
6. :wq

步骤三：修改wsrep.cnf文件

1）分别修改3台服务器的wsrep.cnf

1. [root@pxcnode71 ~]# vim /etc/percona-xtradb-cluster.conf.d/mysqld\_safe.cnf
2. wsrep\_cluster\_address=gcomm://192.168.4.71,192.168.4.72,192.168.4.73//成员列表
3. wsrep\_node\_address=192.168.4.71 //本机ip
4. wsrep\_cluster\_name=pxc-cluster //集群名
5. wsrep\_node\_name=pxcnode71 //本机主机名
6. wsrep\_sst\_auth="sstuser:123qqq...A" //SST数据同步授权用户及密码
7. :wq

修改服务器192.168.4.72

1. [root@pxcnode72 ~]# vim /etc/percona-xtradb-cluster.conf.d/mysqld\_safe.cnf
2. wsrep\_cluster\_address=gcomm://192.168.4.71,192.168.4.72,192.168.4.73//成员列表
3. wsrep\_node\_address=192.168.4.72 //本机ip
4. wsrep\_cluster\_name=pxc-cluster //集群名
5. wsrep\_node\_name=pxcnode72 //本机主机名
6. wsrep\_sst\_auth="sstuser:123qqq...A" //SST数据同步授权用户及密码
7. :wq

修改服务器192.168.4.73

1. [root@pxcnode73 ~]# vim /etc/percona-xtradb-cluster.conf.d/mysqld\_safe.cnf
2. wsrep\_cluster\_address=gcomm://192.168.4.71,192.168.4.72,192.168.4.73//成员列表
3. wsrep\_node\_address=192.168.4.73 //本机ip
4. wsrep\_cluster\_name=pxc-cluster //集群名
5. wsrep\_node\_name=pxcnode73 //本机主机名
6. wsrep\_sst\_auth="sstuser:123qqq...A" //SST数据同步授权用户及密码
7. :wq

步骤四：启动服务

1）启动集群服务

注意：在1台服务器上执行即可(192.168.4.71)，首次启动服务时间比较长

1. [root@pxcnode71 ~]# ]# systemctl start mysql@bootstrap.service //启动集群服务
2. [root@pxcnode71 ~]# grep pass /var/log/mysqld.log     //查看数据库管理员初始登录密码
3. 2019-06-20T12:29:42.489377Z 1 [Note] A temporary password is generated for root@localhost: W.HiOb8(ok)\_
4. [root@pxcnode71 ~]#mysql –uroot –p’ W.HiOb8(ok)\_’ //使用初始密码登录
5. Mysql> alter user root@”localhost” identified by “123456”;//修改登录密码
6. MySQL> exit;
7. [root@pxcnode71 ~]#mysql –uroot –p123456 //使用修改后的密码登录
8. Mysql> garnt reload, lock tables,replication client,process on \*.\* to
9. sstuser@"localhost” identified by “123qqq…A”; //添加授权用户

2）启动数据库服务

启动主机pxcnode72的数据库服务，会自动同步pxcnode71主机的root初始密码和授权用户sstuser

1. [root@pxcnode72 ~]# systemctl start mysql //启动数据库服务
2. [root@pxcnode72 ~]#
3. [root@pxcnode72 ~]# netstat -utnlp | grep :3306
4. tcp6 0 0 :::3306 :::\* LISTEN 12794/mysqld
5. [root@pxcnode72 ~]# netstat -utnlp | grep :4567
6. tcp 0 0 0.0.0.0:4567 0.0.0.0:\* LISTEN 12794/mysqld
7. [root@host72 ~]#

启动主机pxcnode73的数据库服务，会自动同步pxcnode71主机的root初始密码和授权用户sstuser

1. [root@pxcnode73 ~]# systemctl start mysql //启动数据库服务
2. [root@pxcnode73 ~]#
3. [root@pxcnode73 ~]# netstat -utnlp | grep :3306
4. tcp6 0 0 :::3306 :::\* LISTEN 12794/mysqld
5. [root@pxcnode73 ~]# netstat -utnlp | grep :4567
6. tcp 0 0 0.0.0.0:4567 0.0.0.0:\* LISTEN 12794/mysqld
7. [root@host73 ~]#

**3 案例3：测试配置**

**3.1 问题**

* 查看集群信息
* 访问集群，存取数据
* 测试故障自动恢复

**3.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：查看集群信息

1）启动数据库服务

在任意一台数据查看都可以。

1. [root@pxcnode71 ~]# mysql -uroot -p123456
2. wsrep\_incoming\_addresses 192.168.4.71:3306,192.168.4.72:3306,192.168.4.73:3306 //集群成员列表
3. wsrep\_cluster\_size 3 //集群服务器台数
4. wsrep\_cluster\_status Primary //集群状态
5. wsrep\_connected ON            //连接状态
6. wsrep\_ready ON //服务状态

步骤二：访问集群，存取数据

1）添加访问数据的连接用户 （在任意一台服务器上添加都可以，另外的2台服务器会自动同步授权用户）

1. [root@pxcnode72 ~]# mysql -uroot -p123456
2. mysql> grant all on gamedb.\* to yaya@"%" identified by "123456"; //添加授权用户
3. Query OK, 0 rows affected, 1 warning (0.18 sec)
4. [root@pxcnode71 ~]# mysql -uroot -p123456 -e 'show grants for yaya@"%" ' //查看
5. mysql: [Warning] Using a password on the command line interface can be insecure.
6. +--------------------------------------------------+
7. | Grants for yaya@% |
8. +--------------------------------------------------+
9. | GRANT USAGE ON \*.\* TO 'yaya'@'%' |
10. | GRANT ALL PRIVILEGES ON `gamedb`.\* TO 'yaya'@'%' |
11. +--------------------------------------------------+
12. [root@pxcnode71 ~]#
13. [root@pxcnode73 ~]# mysql -uroot -p123456 -e 'show grants for yaya@"%" ' //查看
14. mysql: [Warning] Using a password on the command line interface can be insecure.
15. +--------------------------------------------------+
16. | Grants for yaya@% |
17. +--------------------------------------------------+
18. | GRANT USAGE ON \*.\* TO 'yaya'@'%' |
19. | GRANT ALL PRIVILEGES ON `gamedb`.\* TO 'yaya'@'%' |
20. +--------------------------------------------------+
21. [root@pxcnode73 ~]#

2）客户端连接集群存取数据 （连接任意一台数据库服务器的ip地址都可以）

连接数据服务器主机73

1. client50 ~]# mysql -h192.168.4.73 -uyaya -p123456 //连接服务器73
2. mysql>
3. mysql> create database gamedb; //建库
4. Query OK, 1 row affected (0.19 sec)
5. mysql> create table gamedb.a(id int primary key auto\_increment,name char(10));//建表
6. Query OK, 0 rows affected (1.02 sec)
7. mysql> insert into gamedb.a(name)values("bob"),("tom"); //插入记录
8. Query OK, 2 rows affected (0.20 sec)
9. Records: 2 Duplicates: 0 Warnings: 0

3）在另外2台数据库服务器查看数据，客户端连接数据库服务器71主机查看数据。

1. client50 ~]# mysql -h192.168.4.71 -uyaya -p123456 //连接服务器71
2. mysql> select \* from gamedb.a; //查看记录
3. +----+-------+
4. | id | name |
5. +----+-------+
6. | 2 | bob |
7. | 5 | tom |

4）客户端连接数据库服务器73主机查看数据

1. client50 ~]# mysql -h192.168.4.73 -uyaya -p123456 //连接服务器73
2. mysql> select \* from gamedb.a; //查看记录
3. +----+-------+
4. | id | name |
5. +----+-------+
6. | 2 | bob |
7. | 5 | tom |

步骤三：测试故障自动恢复

1）停止数据库服务

停止3台服务器的任意一台主机的数据库服务都不会影响数据的存取。

1. [root@pxcnode71 ~]# systemctl stop mysql //停止71主机的数据库服务
2. Client50 ~]# client50 ~]# mysql -h192.168.4.72 -uyaya -p123456 //连接服务器72
3. mysql> insert into gamedb.a(name)values("bob2"),("tom2");
4. mysql> insert into gamedb.a(name)values("jerry"),("jack");
5. Query OK, 2 rows affected (0.20 sec)
6. Records: 2 Duplicates: 0 Warnings: 0

客户端50，连接数据库主机73，查看数据

1. client50 ~]# mysql -h192.168.4.73 -uyaya -p123456 //连接服务器73
2. mysql> select \* from gamedb.a;
3. +----+-------+
4. | id | name |
5. +----+-------+
6. | 2 | bob |
7. | 5 | tom |
8. | 7 | bob2 |
9. | 9 | tom2 |
10. | 11 | jerry |
11. | 13 | jack |
12. +----+-------+
13. 6 rows in set (0.00 sec)

3）启动71主机的数据库服务

数据库服务运行后，会自动同步宕机期间的数据。

1. client50 ~]# mysql -h192.168.4.71 -uyaya -p123456 //连接服务器71
2. mysql> select \* from gamedb.a;
3. +----+-------+
4. | id | name |
5. +----+-------+
6. | 2 | bob |
7. | 5 | tom |
8. | 7 | bob2 |
9. | 9 | tom2 |
10. | 11 | jerry |
11. | 13 | jack |
12. +----+-------+
13. rows in set (0.00 sec)

**4 案例4：MySQL存储引擎的配置**

**4.1 问题**

本案例要求MySQL数据存储引擎的使用，完成以下任务操作：

* 查看服务支持的存储引擎
* 修改服务默认使用的存储引擎
* 查看表使用的存储引擎
* 设置表使用的存储引擎
* 修改表存储引擎

**4.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：查看服务支持的存储引擎

登入MySQL服务器，查看当前支持哪些存储引擎。

使用mysql命令连接，以root用户登入：

1. [root@dbsvr1 ~]# mysql -u root –p
2. Enter password:
3. Welcome to the MySQL monitor. Commands end with ; or \g.
4. Your MySQL connection id is 9
5. Server version: 5.7.17 MySQL Community Server (GPL)
6. Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.
7. Oracle is a registered trademark of Oracle Corporation and/or its
8. affiliates. Other names may be trademarks of their respective
9. owners.
10. Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
11. mysql>

执行SHOW ENGINES\G指令可列表查看，MySQL 5.6可用的存储引擎有9种（除最后的FEDERATED以外，其他8种都支持），其中默认采用的存储引擎为InnoDB：

1. mysql> SHOW ENGINES\G
2. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
3. Engine: InnoDB
4. Support: DEFAULT //此存储引擎为默认
5. Comment: Supports transactions, row-level locking, and foreign keys
6. Transactions: YES
7. XA: YES
8. Savepoints: YES
9. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
10. Engine: MRG\_MYISAM
11. Support: YES
12. Comment: Collection of identical MyISAM tables
13. Transactions: NO
14. XA: NO
15. Savepoints: NO
16. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
17. Engine: MEMORY
18. Support: YES
19. Comment: Hash based, stored in memory, useful for temporary tables
20. Transactions: NO
21. XA: NO
22. Savepoints: NO
23. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 4. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
24. Engine: BLACKHOLE
25. Support: YES
26. Comment: /dev/null storage engine (anything you write to it disappears)
27. Transactions: NO
28. XA: NO
29. Savepoints: NO
30. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 5. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
31. Engine: MyISAM
32. Support: YES
33. Comment: MyISAM storage engine
34. Transactions: NO
35. XA: NO
36. Savepoints: NO
37. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 6. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
38. Engine: CSV
39. Support: YES
40. Comment: CSV storage engine
41. Transactions: NO
42. XA: NO
43. Savepoints: NO
44. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 7. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
45. Engine: ARCHIVE
46. Support: YES
47. Comment: Archive storage engine
48. Transactions: NO
49. XA: NO
50. Savepoints: NO
51. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 8. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
52. Engine: PERFORMANCE\_SCHEMA
53. Support: YES
54. Comment: Performance Schema
55. Transactions: NO
56. XA: NO
57. Savepoints: NO
58. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 9. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
59. Engine: FEDERATED
60. Support: NO //此引擎不被支持
61. Comment: Federated MySQL storage engine
62. Transactions: NULL
63. XA: NULL
64. Savepoints: NULL
65. 9 rows in set (0.01 sec)

步骤二：修改服务默认使用的存储引擎

在 mysql> 环境中，可以直接通过SET指令更改默认的存储引擎（只在本次连接会话过程中有效，退出重进即失效） 。比如临时修改为MyISAM，可执行下列操作：

1. mysql> SET default\_storage\_engine=MyISAM;             //改用MyISAM引擎
2. Query OK, 0 rows affected (0.00 sec)
3. mysql> SHOW VARIABLES LIKE 'default\_storage\_engine';         //确认结果
4. +------------------------+--------+
5. | Variable\_name | Value |
6. +------------------------+--------+
7. | default\_storage\_engine | MyISAM |
8. +------------------------+--------+
9. 1 row in set (0.00 sec)

若希望直接修改MySQL服务程序所采用的默认存储引擎，应将相关设置写入配置文件/etc/my.cnf，并重启服务后生效。比如：

1. [root@dbsvr1 ~]# vim /etc/my.cnf
2. [mysqld]
3. .. ..
4. default\_storage\_engine=myisam                             //改用myisam引擎
5. [root@dbsvr1 ~]# systemctl restart mysqld.service         //重启服务

重新登入 mysql> 确认修改结果：

1. [root@dbsvr1 ~]# mysql -u root -p
2. Enter password:
3. Welcome to the MySQL monitor. Commands end with ; or \g.
4. Your MySQL connection id is 3
5. Server version: 5.7.17 MySQL Community Server (GPL)
6. Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.
7. Oracle is a registered trademark of Oracle Corporation and/or its
8. affiliates. Other names may be trademarks of their respective
9. owners.
10. Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
11. mysql> SHOW VARIABLES LIKE 'default\_storage\_engine';
12. +------------------------+--------+
13. | Variable\_name | Value |
14. +------------------------+--------+
15. | default\_storage\_engine | MYISAM |                 //默认引擎已修改
16. +------------------------+--------+
17. 1 row in set (0.00 sec)
18. mysql> exit
19. Bye

步骤三：查看表使用的存储引擎

登入MySQL服务器查看。

1. mysql> show create table user \G; //查看建表命令
2. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
3. Table: user
4. Create Table: CREATE TABLE `user` (
5. `id` int(11) NOT NULL AUTO\_INCREMENT,
6. `name` char(50) DEFAULT NULL,
7. `age` tinyint(3) unsigned DEFAULT '19',
8. `password` char(1) DEFAULT NULL,
9. `uid` int(11) DEFAULT NULL,
10. `gid` int(11) DEFAULT NULL,
11. `comment` char(150) DEFAULT NULL,
12. `homedir` char(50) DEFAULT NULL,
13. `shell` char(50) DEFAULT NULL,
14. PRIMARY KEY (`id`)
15. ) ENGINE=InnoDB AUTO\_INCREMENT=46 DEFAULT CHARSET=latin1 //存储引擎是InnoDB
16. 1 row in set (0.00 sec)
17. ERROR:
18. No query specified
19. mysql>

步骤四：设置表使用的存储引擎

登入MySQL服务器设置。

1. mysql> create table stuinfo( name char(10), age int )engine=memory;//设置
2. Query OK, 0 rows affected (0.12 sec)
3. mysql>
4. mysql> show create table stuinfo\G; //查看
5. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
6. Table: stuinfo
7. Create Table: CREATE TABLE `stuinfo` (
8. `name` char(10) DEFAULT NULL,
9. `age` int(11) DEFAULT NULL
10. ) ENGINE=MEMORY DEFAULT CHARSET=latin1 //存储引擎名
11. 1 row in set (0.00 sec)
12. ERROR:
13. No query specified
14. mysql>

步骤五：修改表存储引擎

登入MySQL服务器修改。

1. mysql> alter table stuinfo engine=innodb; //修改
2. Query OK, 0 rows affected (0.54 sec)
3. Records: 0 Duplicates: 0 Warnings: 0
4. mysql>
5. mysql> show create table stuinfo\G; //查看
6. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
7. Table: stuinfo
8. Create Table: CREATE TABLE `stuinfo` (
9. `name` char(10) DEFAULT NULL,
10. `age` int(11) DEFAULT NULL
11. ) ENGINE=InnoDB DEFAULT CHARSET=latin1 //当前存储引擎名
12. 1 row in set (0.00 sec)
13. ERROR:
14. No query specified
15. mysql>

**5 案例5：事务特性**

**5.1 问题**

具体操作如下：

* 创建innodb存储引擎的表
* 关闭服务的自动提交功能
* 测试事务特性

**5.2 步骤**

实现此案例需要按照如下步骤进行。

步骤一：创建innodb存储引擎的表

1）数据库管理员root登录，创建新库、新表。

1. [root@host50 ~]# mysql -uroot -p123456 //访问服务
2. mysql> create database db10; //建库
3. Query OK, 1 row affected (0.05 sec)
4. mysql> use db10; //切换库
5. Database changed
6. mysql>
7. mysql> create table a(id int)engine=innodb; //建表并指定存储引擎
8. Query OK, 0 rows affected (0.52 sec)

步骤二：关闭服务的自动提交功能

1）数据库管理员root登录，关闭服务的自动提交功能。

1. mysql> show variables like "%commit%"; //查看所有包涵commit 字样的变量
2. +-----------------------------------------+-------+
3. | Variable\_name | Value |
4. +-----------------------------------------+-------+
5. | autocommit | ON |     //自动提交功能开启
6. | binlog\_group\_commit\_sync\_delay | 0 |
7. | binlog\_group\_commit\_sync\_no\_delay\_count | 0 |
8. | binlog\_order\_commits | ON |
9. | innodb\_api\_bk\_commit\_interval | 5 |
10. | innodb\_commit\_concurrency | 0 |
11. | innodb\_flush\_log\_at\_trx\_commit | 1 |
12. | slave\_preserve\_commit\_order | OFF |
13. +-----------------------------------------+-------+
14. 8 rows in set (0.01 sec)
15. mysql> set autocommit=off ; 关闭自动提交
16. Query OK, 0 rows affected (0.00 sec)
17. mysql> show variables like "autocommit"; //查看
18. +---------------+-------+
19. | Variable\_name | Value |
20. +---------------+-------+
21. | autocommit | OFF |            //已处于关闭状态
22. +---------------+-------+
23. 1 row in set (0.00 sec)
24. mysql>

步骤三：测试事务特性

1）插入新记录，不执行提交命令commit

1. mysql> insert into db10.a values(101); //插入记录
2. Query OK, 1 row affected (0.00 sec)
3. mysql> select \* from db10.a ;//查看记录
4. +------+
5. | id |
6. +------+
7. | 101 |
8. +------+
9. 1 row in set (0.00 sec)
10. mysql>

2）打开新终端访问数据服务，查看不到插入的数据

注：此处打开的终端称为终端2 ， 此处之前终端被称为终端1

1. [root@host50 ~]# mysql -uroot -p123456
2. mysql> select \* from db10.a; //没有记录
3. Empty set (0.00 sec)
4. mysql>

3）终端1 执行提交命令commit

1. mysql> select \* from db10.a ;
2. +------+
3. | id |
4. +------+
5. | 101 |
6. +------+
7. 1 row in set (0.00 sec)
8. mysql> commit ; 执行提交命令
9. Query OK, 0 rows affected (0.08 sec)

4）第终端2执行查看命令

1. mysql> select \* from db10.a;
2. Empty set (0.00 sec)
3. mysql> select \* from db10.a; //查看到数据
4. +------+
5. | id |
6. +------+
7. | 101 |
8. +------+
9. 1 row in set (0.00 sec)
10. mysql>

5）在终端1删除记录 ，不执行提交命令commit

将/etc/passwd文件复制到/var/lib/mysql-files/目录下，

1. mysql> select \* from db10.a ; //删除前查看
2. +------+
3. | id |
4. +------+
5. | 101 |
6. +------+
7. 1 row in set (0.00 sec)
8. mysql> delete from db10.a ;//删除表记录
9. Query OK, 1 row affected (0.00 sec)
10. mysql>
11. mysql> select \* from db10.a ; //查看不到记录
12. Empty set (0.00 sec)

6）在终端2 依然可以查看到记录

1. mysql> select \* from db10.a;
2. +------+
3. | id |
4. +------+
5. | 101 |
6. +------+
7. 1 row in set (0.00 sec)
8. mysql>

7）在终端1，执行回滚命令

1. mysql> select \* from db10.a ; //回滚前查看
2. Empty set (0.00 sec)
3. mysql>
4. mysql> rollback ; //数据回滚
5. Query OK, 0 rows affected (0.03 sec)
6. mysql>
7. mysql> select \* from db10.a ; //回滚后查看
8. +------+
9. | id |
10. +------+
11. | 101 |
12. +------+
13. 1 row in set (0.00 sec)
14. mysql>
15. mysql> delete from db10.a ; //删除记录
16. Query OK, 1 row affected (0.00 sec)
17. mysql> commit ; //提交
18. Query OK, 0 rows affected (0.08 sec)
19. mysql>
20. mysql> rollback ; //数据回滚
21. Query OK, 0 rows affected (0.00 sec)
22. mysql> select \* from db10.a ; //查看不到记录
23. Empty set (0.00 sec)
24. mysql>

8）在终端2 也查看不到记录

1. mysql> select \* from db10.a;
2. Empty set (0.00 sec)