PXC

Mysql服务高可用集群软件、可以弥补MHA的一些缺点

行情:

目前大多为mha

后出现的PXC,还不稳定,可能出bug(还不成熟,稳定)

MHA缺点

必须条件: 一主多从 一主多从

客户端访问必须连接vip vip ---->主库

且vip地址必须在主数据库服务器上

还原坏的服务器,需手动配置数据一致,配置为新主的从 还原手动

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

**部署在后端服务器上 自带mysql 不需安装mysq**l

PXC特点:

数据一致性强: 集群数据必须一致

没有主从切换操作

节点自动加入,无需手动拷贝数据(**自动同步宕机期间的数据**)

部署使用简单

支持innodb储存引擎

多线程复制

**两部分组成:**

**存储** **同步**

**涉及的端口:**

3306: 数据库服务端口(**实时开启**)

4567: 集群之间通信端口(**实时开启**)

4444: SST端口 SST:全量同步(完全同步)

4568: IST端口 IST:增量同步

**因为实时开启,开启后能看到3306,4567**

**配置文件**

• 配置文件路径

–/etc/percona-xtradb-cluster.conf.d

• 配置文件路径

–文件说明

mysqld.cnf //数据库运行参数配置文件(相似于SQL配置文件)

mysqld\_safe.cnf //Percona server5.7配置文件

wsrep.cnf //PXC集群配置文件(集群成员列表,等集群配置)

#################################################################################

**部署PXC**

1.配置主机名映射(3台主机都做)

[root@pxcnode71 ~]#vim /etc/hosts

192.168.4.71 pxcnode71

192.168.4.72 pxcnode72

192.168.4.73 pxcnode73

]# for i in 2 3

> do

>scp /etc/hosts [root@192.168.4.7$i:/etc/hosts](mailto:root@192.168.4.7$i:/etc/hosts)

>done

2.上传包至7{1..3}:

]$ scp -r /linux-soft/03/PXC [root@192.168.4.71:/root/](mailto:root@192.168.4.71:/root/)

........

]# ls PXC

• 前三个都为最后包的依赖包

热备依赖包:

-- libev-4.15-1.el6.rf.x86\_64.rpm

在线热备功能:

-- percona-xtrabackup-24-2.4.13-1.el7.x86\_64.rpm

递归压缩功能:

-- qpress-1.1-14.11.x86\_64.rpm #无依赖

提供集群服务: (主包)

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

解压后????????????server

1. 安装(三台)

–安装有先后顺序(依赖关系原因)

Libev ---> percona-xtrabackup ---> qpress ---> Percona-XtraDB-Cluster包里所有

依赖:

]#rpm -ivh libev-4.15-1.el6.rf.x86\_64.rpm

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

]#rpm -ivh qpress-1.1-14.11.x86\_64.rpm

主包:

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

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

1. 配置文件(三台)

以71主机为例:

Mysqld.cnf

[mysqld]

server-id=71

wsrep.cnf

8 wsrep\_cluster\_address=gcomm://192.168.4.71,....72,.....73 #集群成员列表

25 wsrep\_node\_address=192.168.4.71 #本机IP,去掉注释

27 wsrep\_cluster\_name=pxc-cluster #集群名称

30 wsrep\_node\_name=pxcnode71 #本机主机名

39 wsrep\_sst\_auth="sstuser:123qqq...A" #SST所用的用户&密码

1. 启动服务(**71主机启动,其他不操作**)

因为实时开启,开启后能看到3306,4567

启动:

]# systemctl start [mysql@bootstrap.service](mailto:mysql@bootstrap.service)

会做数据初始化(等同于之前的mysql首次启动)

启动服务后,mysql的root随机密码会记录在日志:/var/log/mysqld.log

修改root密码(没有密码策略)

授权用户(其他主机会自动同步)

> grant reload, lock tables,replication client,process on \*.\* to sst

其他机器启动mysql服务(不是此软件服务)

]# systemctl start **mysql** **#没有”d”**

#启动后会同步71的mysql操作(root改密,授权用户操作)

查看集群状态

> show status like "%wsrep%";

wsrep\_incoming\_addresses | 192.168.4.71:3306,192.168.4.72:3306

wsrep\_cluster\_size | 2

wsrep\_connected | ON

wsrep\_ready | ON

###########################################################################################

**测试pxc**

1. 客户端连接测试

• 连接:

]# mysql -uyaya10 -p123456 -h192.168.4.73

• 数据测试:

**创建的表必须要有主键,否则不能写入**

• 表自增字段规律:

–目的: 避免自增值重复(主键不允许重复)

首条记录:+ 2 (服务器数-1)

自加: +3 (服务器数)

中途1台宕机: 依然+3(原服务器数), 但就一次,之后变为+2,(宕机后的服务器数)

中途1台恢复: 直接+4(原服务器数+2)一次, 然后+3(恢复后的服务器数)

首次: 2 bob

之后: 5 tom

之后: 8 hry

2.模拟一台服务器宕机修复后,查看数据是否恢复

#####################################################################################

总结:

授权用户(其他主机会自动同步)

> grant reload, lock tables,replication client,process on \*.\* to sst

]#systemctl start [mysql@bootstrap.service](mailto:mysql@bootstrap.service)

show status like "%wsrep%"

################################################################################

**Mysql储存引擎**

本次理论大于操作

引擎: 自带的功能程序,处理表的处理器,不同的引擎数据存储的方式不一样

只有Innodb引擎有外键,事务日志文件

Mysql 5.0/5.1 ---> mylsam

Mysql ??????????????????

查看版本

“%version%”

查看支持的存储引擎

> show engines;

引擎名 DEFAULT(当前使用的) 描述

查看表的存储引擎

> show create table 表名

Myisam储存引擎

• 主要特点

–支持表级锁

–不支持事务、事务回滚、外键

• 表文件

–表名.frm 表结构

–表名.MYI 索引

–表名.MYD 数据

Innodb储存引擎

• 主要特点

–支持行级锁

–支持事务、事务回滚、外建

• 表文件

–表名.frm 表结构

–表名.ibd 表空间+索引

• 事务日志文件

–ibdata1

–ib\_logfile0

–ib\_logfile1

#############################################################################

**Mysql锁机制**

**锁**: 并发冲突问题, 多个用户同时对一个表修改时冲突问题,加锁后只允许一个用户访问,别的用户读,写等都不行

• 锁粒度(锁的范围)

–表级锁

–行级锁

• 锁类型

–读锁: (共享锁),支持并发读

–写锁: (互斥锁,独占锁)????????????

• 查看当锁所状态

–show status like “**table\_lock**”;

**事务特性 16.43**

• Atomic : 原子性

–事务的整个操作是一个整体,不可分割,必须全部成功,有一个操作错误,整个错误

• Consistency : 一致性

–事务操作的前后,表中的记录没有变化

• Isolation : 隔离性

–事务操作是相互隔离不受影响的

• Durability : 持久性

–数据一旦提交,不可改变,永久改变表数据

事务回滚:

概念: 操作中出错了,可以恢复初始的数据

原理: 读事务日志文件(二进制文件,记录命令)

自动提交: 在一回车就提交了

事务特性 • 相关命令

Innodb:

查看自动提交

> show variables like "autocommit";

关闭自动提交

> set autocommit=off

回滚|提交 数据

回滚: rollbake;

提交: commit;

########################################################################