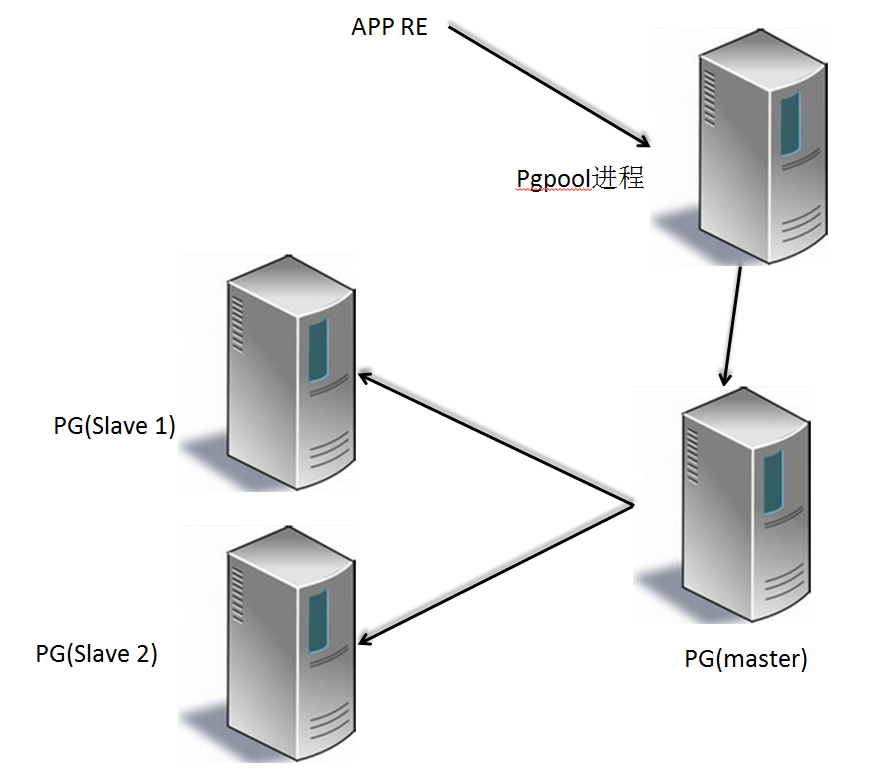
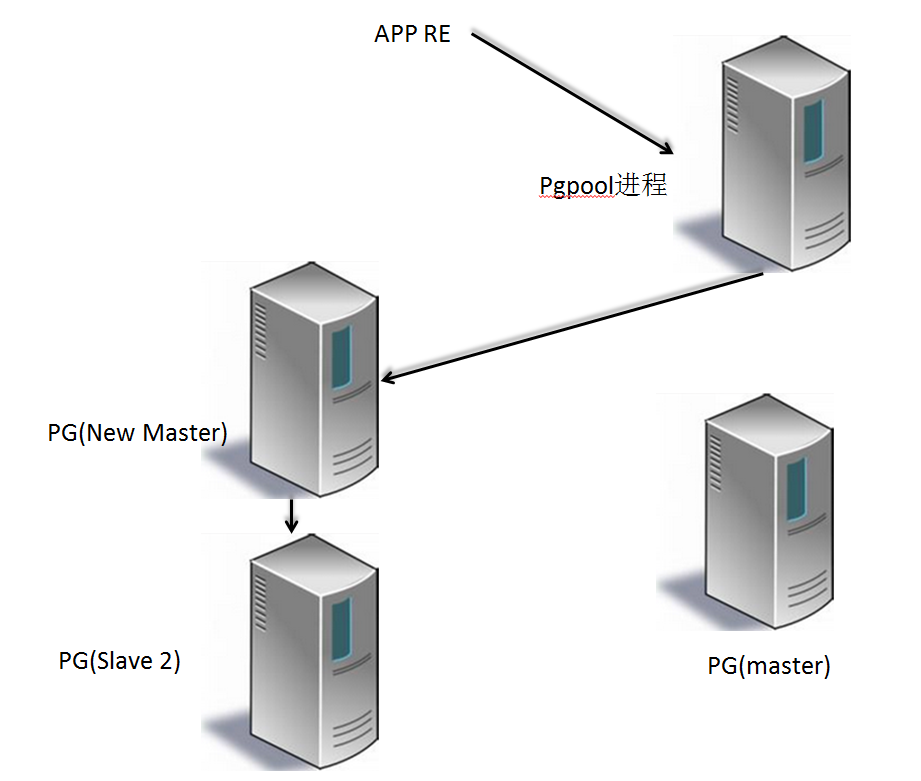
1. 构架图

本文档由 postgreSQL 专业群 深圳-GS 初稿, PostgreSQL 群 beta 2015-01-18 日修订

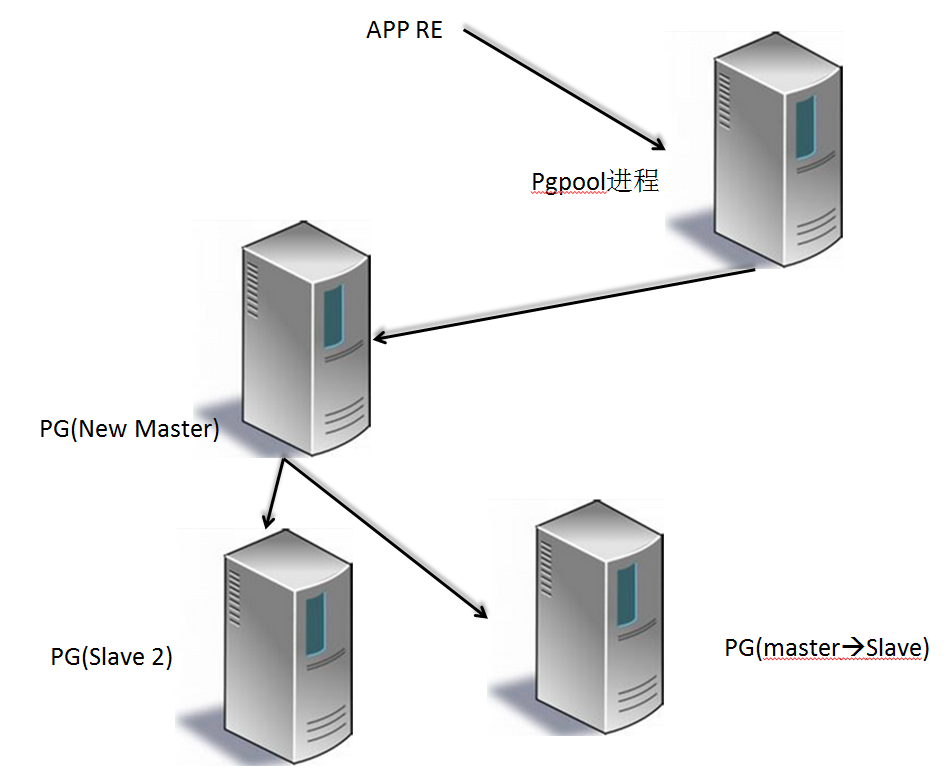
主要工作有 GS 完成



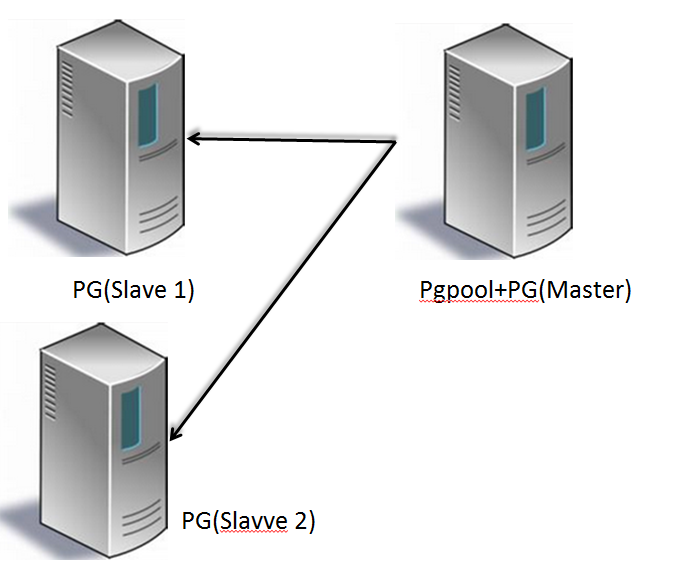
1. 主机宕机后构架图



1. 恢复完成后构架图



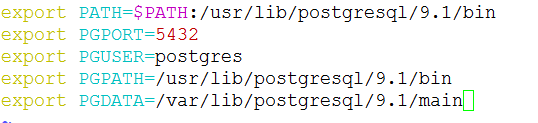
1. 实际部署图



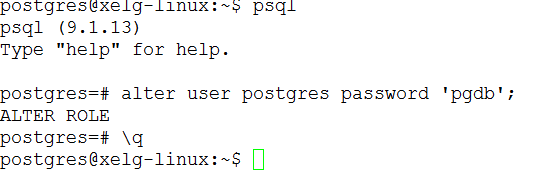
1. 操作步骤
2. 安装pg
   1. apt-get install postgresql-9.1
   2. ：apt-get install postgresql-contrib-9.1
   3. ：apt-get install libpq-dev
   4. ：apt-get install postgresql-server-dev-9.1
   5. ：配置系统用户postgres密码：pg
   6. ：使用root用户，创建目录用来做后面的归档目录



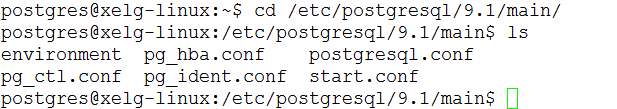
* 1. ：进入postgres的home目录，vim .bash\_profile添加环境变量，修改完成后使用source .bash\_profile激活环境变量



* 1. ：配置数据库超级用户postgres密码：pgdb，并使用\q推出SQL命令模式



* 1. 进入postgres配置目录（下面目录只针对pg在ubuntu上的apt-get默认安装）

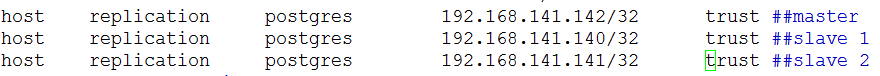


* 1. 修改：pg\_hba.conf: 添加行：

host replication postgres 192.168.141.142/32 trust

host replication postgres 192.168.141.140/32 trust

host replication postgres 192.168.141.141/32 trust



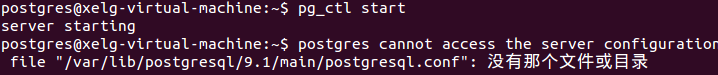
* 1. 修改postgresql.conf
     1. 启用参数：listen\_addresses
     2. 配置端口：port = 5432
     3. 启用参数：wal\_level = hot\_standby
     4. 启用参数：archive\_mode = on

archive\_command = 'cp %p /opt/archivedir/%f'

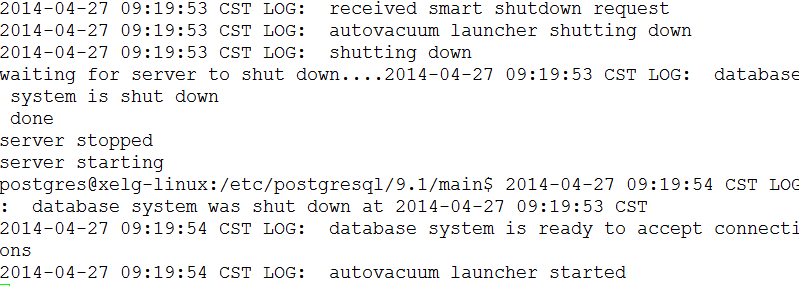
* + 1. 启用参数：hot\_standby = on
    2. 启用参数：max\_wal\_senders = 2
  1. 推出当前shell连接，重新登录系统用户postgres，确保环境变量生效
  2. 创建配置文件postgres.conf软连接文件到$PGDATA目录:

ln /etc/postgresql/9.1/main/postgresql.conf /var/lib/postgresql/9.1/main/

否则在使用pg\_ctl start启动数据库是出现如下错误：



* 1. 启动数据库（系统用户postgres下执行pg\_ctl restart）



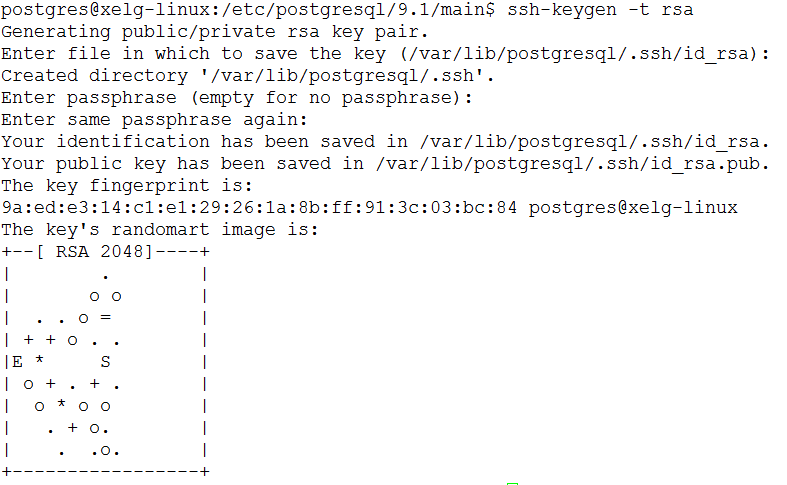
1. 配置ssh无密码连接（postgres）

把B主机的公钥文件拷贝到A主机的authorized\_keys ,这样B主机可以ssh免密码登录A

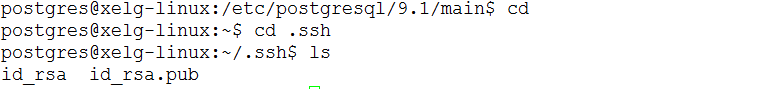
可以通过查看 /var/log/secure 来查看登录信息，这种方式为 publickey

a)安装ssh服务：apt-get install ssh

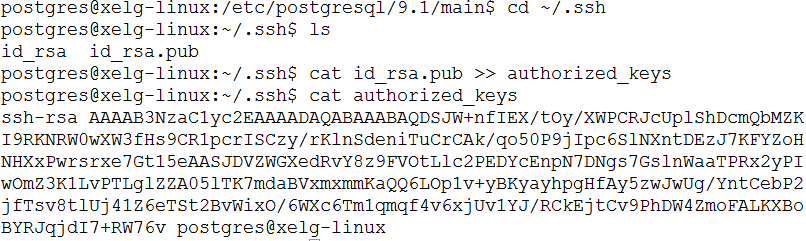
b)切换到系统用户postgres，执行ssh-keygen -t rsa命令，后续默认回车，并生成类似二维码的东西。



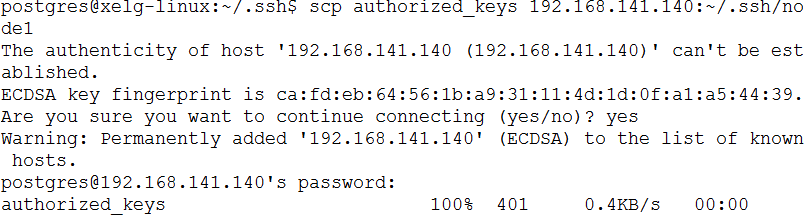
c)进入用户的home目录，用ls可以看到.ssh的文件夹，进入.ssh文件夹，查看文件：



* 1. 使用cat id\_rsa.pub >> authorized\_keys （####名称必须是authorized\_keys)



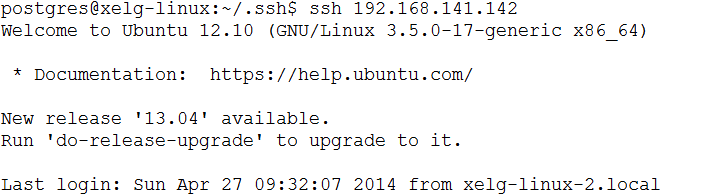
* 1. 使用 scp authorized\_keys node2\_ip:~/.ssh/node1 将当前的authorized\_keys发送到node2和node3的postgres主目录下面的.ssh文件夹下，并命名为node1



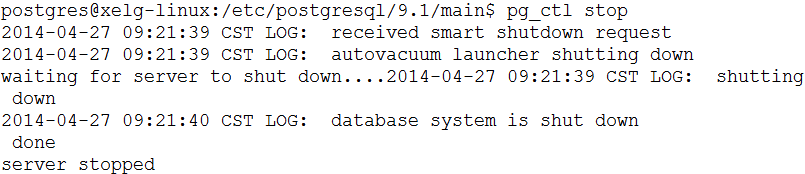
* 1. 使用同样的方法，在node2,node3上面将authorized\_keys传送给其他的2个节点上面，并使用cat node\* >> authorized\_keys追加到authorized\_keys里面

* 1. 测试无密码连接是否生效：在第一次连接输入密码后，以后的连接都不用输入密码



1. 配置pg流复制模式（@slave1,@slave2）
   1. 使用pg\_ctl命令，停止slave1,slave2的PG服务**（@slave）**



* 1. 返回master服务器，执行如下命令

postgres@xelg-linux:~/.ssh$ psql

psql (9.1.13)

Type "help" for help.

postgres=# select pg\_start\_backup('stream\_bak');

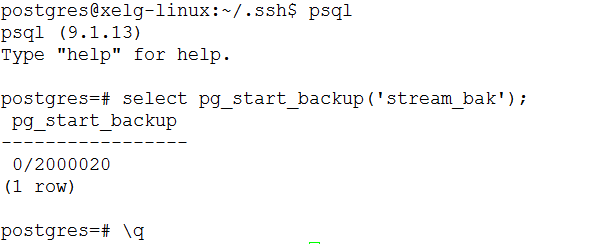
pg\_start\_backup

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

0/2000020

(1 row)

postgres=# \q



* 1. 使用命令scp -r 9.1/ 192.168.141.140: 和scp -r 9.1/ 192.168.141.141: 将备份文件分别传送给slave1 slave2**（@master）**
  2. 在master上面执行如下命令**（@master）**

postgres@xelg-linux:~$ psql

psql (9.1.13)

Type "help" for help.

postgres=# select pg\_stop\_backup();

NOTICE: pg\_stop\_backup complete, all required WAL segments have been archived

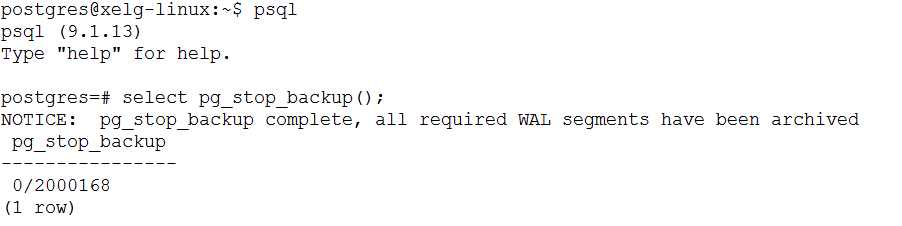
pg\_stop\_backup

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

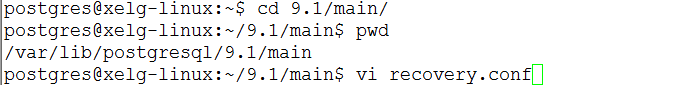
0/2000168

(1 row)

postgres=#



* 1. 切换终端到slave1和slave2，进入~/9.1/main目录，并执行vi recovery.conf**（@slave1 slave2）**



* 1. 添加行：

编辑文件 $PGDATA/recovery.conf

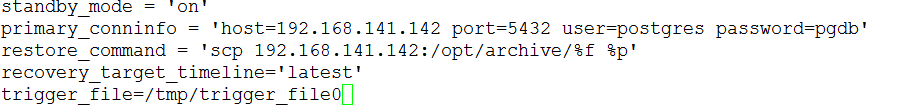
standby\_mode = 'on'

primary\_conninfo = 'host=192.168.141.142 port=5432 user=postgres password=pgdb'

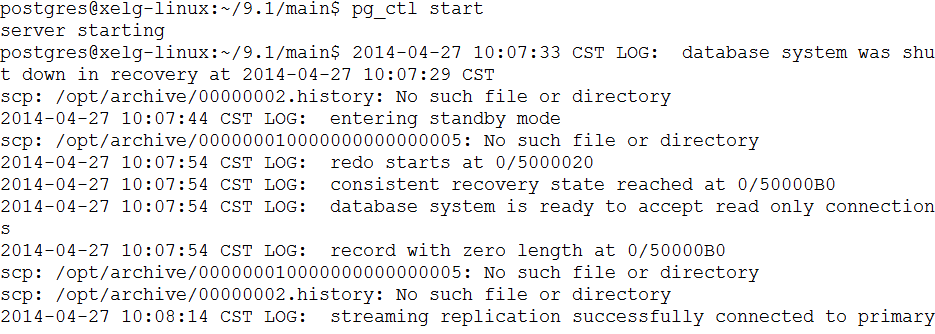
restore\_command = 'scp 192.168.141.142:/opt/archive/%f %p'

recovery\_target\_timeline='latest'

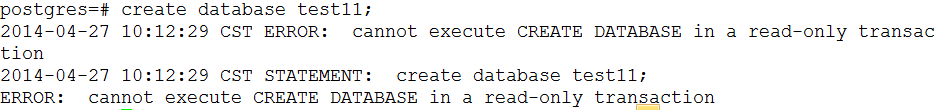
trigger\_file=/tmp/trigger\_file0



* 1. 使用pg\_ctl启动slave1，slave2数据库，并出现流复制成功连接到主节点，表示已经成功配置了pg的复制模式。



* 1. 测试：返回master节点，执行创建数据库命令，并在slave1,slave2上查看，是否已经同时创建该数据库
  2. 测试：返回slave1，slave，执行创建数据库命令，出现如下提示：到现在PG流复制已经成功配置



1. 安装pgpool(执行pgpool的系统用户必须有ssh无密码连接的权利，建议使用postgres用户安装pgpool)

在master执行命令

./configure --with-pgsql=/var/lib/postgresql --with-pgsql-libdir=/usr/lib/postgresql/9.1/lib --with-pgsql-includedir=/usr/include/postgresql

安装pgpool

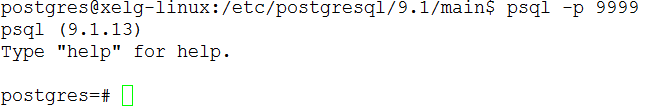
1. 配置pgpool，实现主备自动切换（原master宕机，以下统称M-1）
   1. 使用Root用户进入目录：/usr/local/etc
   2. 用户命令：pg\_md5获取数据库用户postgres密码pgdb的MD5值



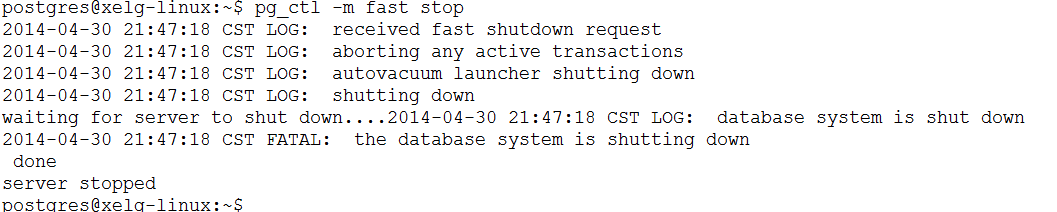
* 1. 执行修改pcp.conf
  2. 执行修改pgpool.conf
  3. 执行修改poo\_hba.conf
  4. 按照pgpool.conf里面配置的failover\_command参数，在对应的目录下创建脚本文件
  5. 参考下面内容：
  6. 配置完成后，使用root用户创建文件夹：mkdir /var/run/pgpool
  7. 使用pgpool命令启动pgpool



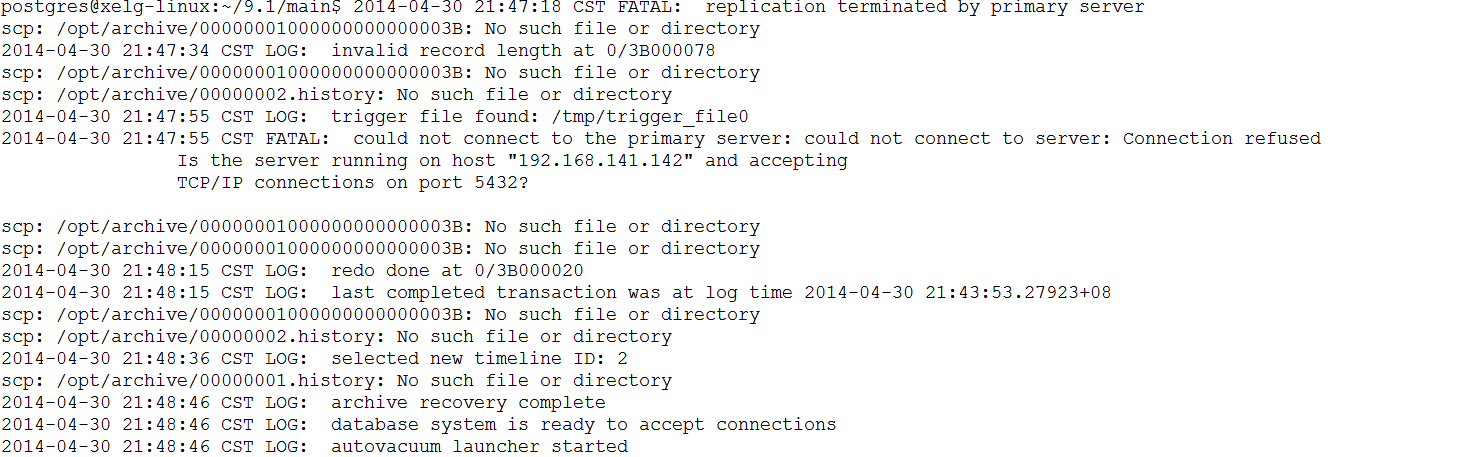
* 1. 切换系统用户到postgrs，执行命令psql –p 9999，看到如下界面是表示pgpool已经安装OK



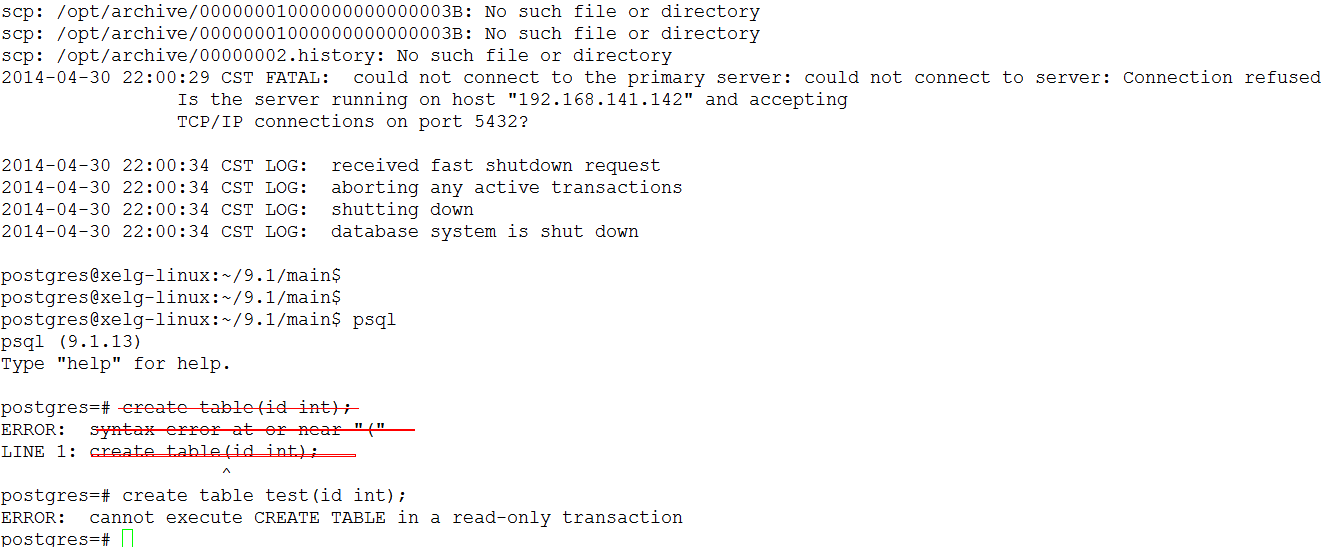
1. 恢复并挂载宕机服务器
   1. 返回master服务器，切换到系统用户postgres，执行pg\_ctl-m fast stop



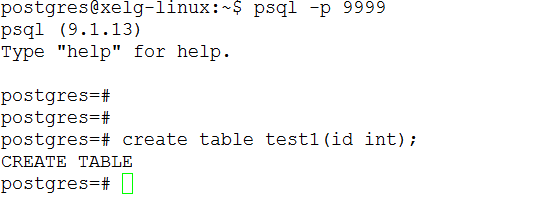
* 1. 切换到slave1，出现如下提示：首先提示主连接失去连接，然后开始启动本地编辑



* 1. 切换到slave2，出现如下提示



* 1. 回到master服务器，执行psql -p 9999,并创建table：test1，如下提示已经成功创建



* 1. **到目前位置，已经完成主备机切换，以下步骤开始挂载原master为slave21**
  2. 进入postgres的data目录：/var/lib/postgresql/9.1/main，vim recovery.conf，内容如下：

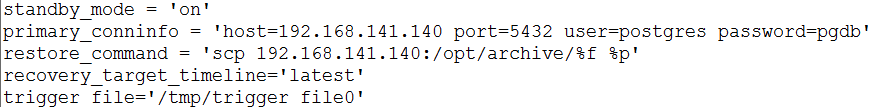
standby\_mode = 'on'

primary\_conninfo = 'host=192.168.141.140 port=5432 user=postgres password=pgdb'

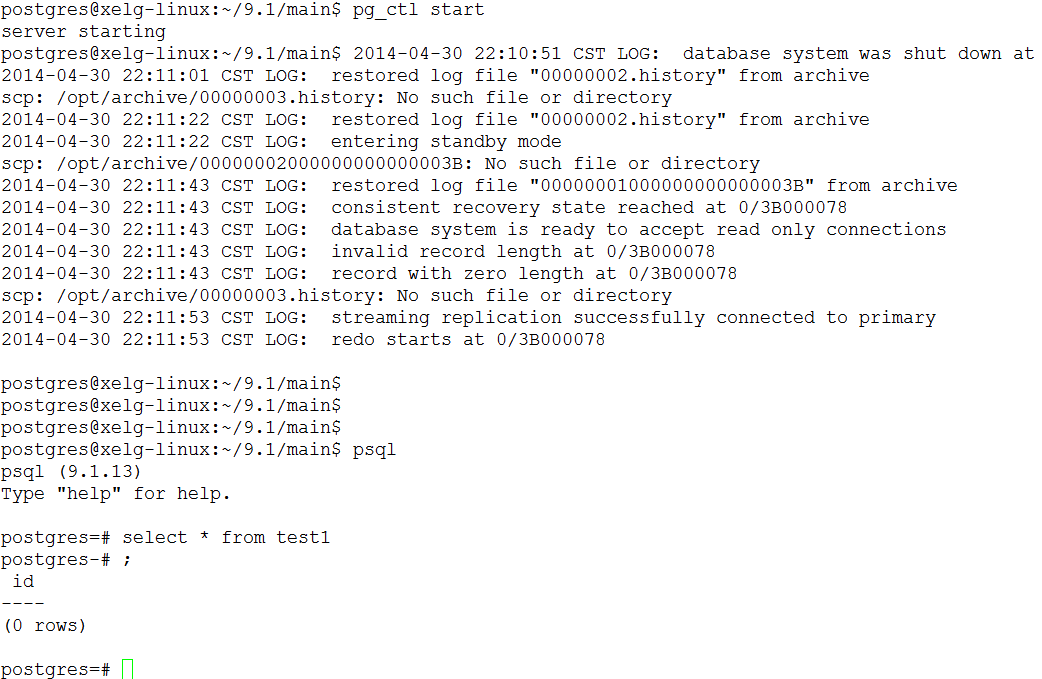
restore\_command = 'scp 192.168.141.140:/opt/archive/%f %p'

recovery\_target\_timeline='latest'

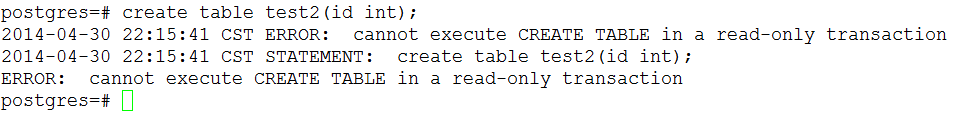
trigger\_file='/tmp/trigger\_file0'



* 1. 执行pg\_ctl start，启动数据库，并执行psql，select \* from test1，查看在原master停止后创建的表test1是否恢复到该数据库上面，如下图所示：test1已经成功出现在原master（现slave21）上面



* 1. 执行创建表语句，出现当期数据库为读模式的警告：



* 1. 在原主节点上执行命令：/usr/local/bin/pcp\_attach\_node 5 localhost 9898 postgres pgdb 0 重新将丢失的节点添加到pgpool上面。（必须步骤，否则以下步骤会出错）

5：超时时间

Localhost：安装pgpool的服务器ip或名称

9898：pcp端口号

Postgres：数据库超级管理员

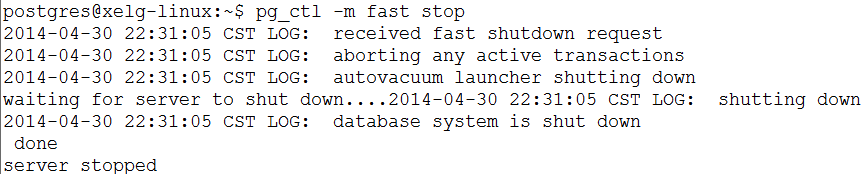
Pgdb：数据库超级管理员密码

0：需要重新加载服务器在pgpool.conf中对应的backend0的0（就是id）

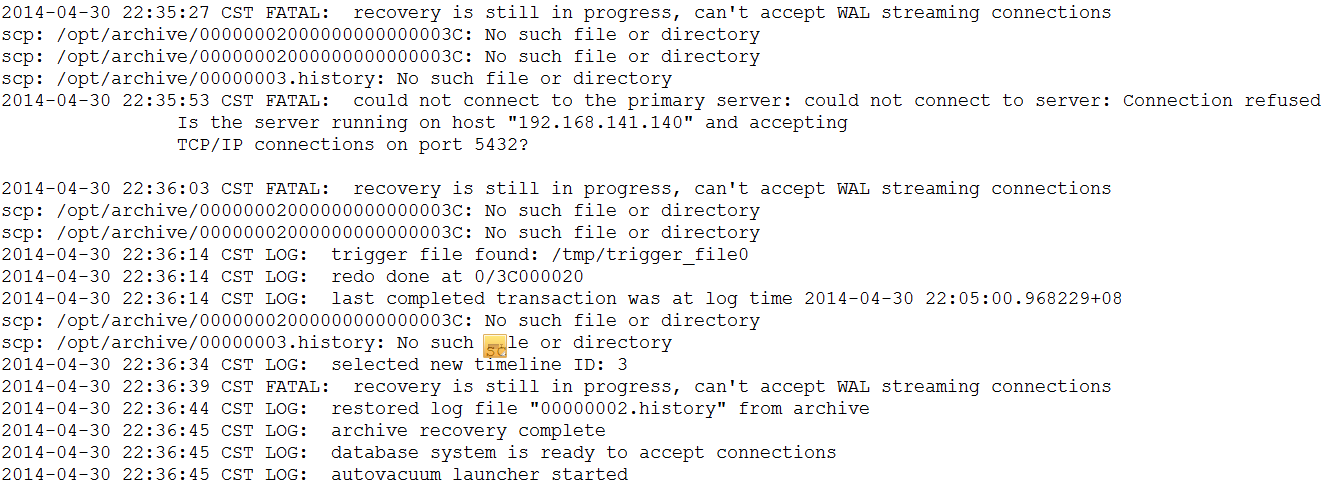
1. 新master宕机后，主备机切换（新master宕机，以下统称M-2，slave 21重新升级为master）
   1. 准备工作：切换到slave21的服务器（原master）上面，修改/var/lib/postgresql的failover.sh文件。修改如下：



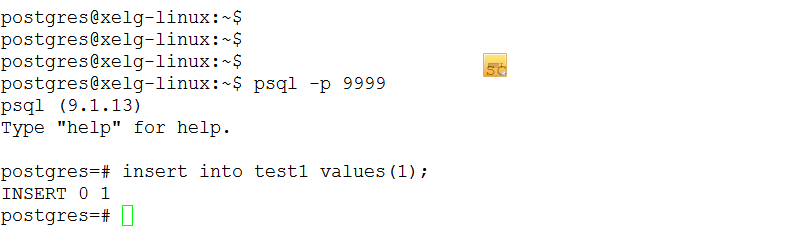
* 1. 切换到M-2服务器（原slave1），执行命令：pg\_ctl –m fast stop



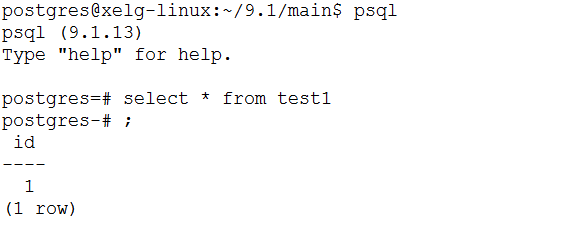
* 1. 切换到slave21服务器，出现如下提示



* 1. 在slave21上执行psql -p 9999，并执行：insert into test1 values(1);如下图，我们就胜利的完成了第二次的主备切换，这个步骤和第一次的主板切换步骤差不多，但是关键是将M-2重新挂载为slave2



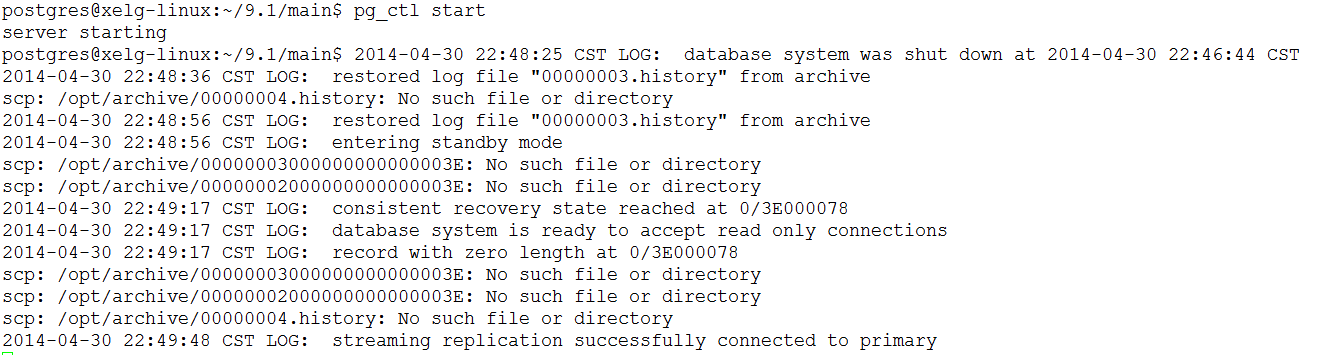
* 1. 检查slave2，中是否有数据插入：



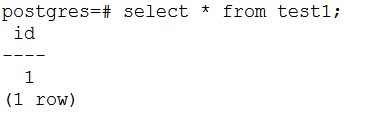
* 1. 移动data目录文件recovery.done为recovery.conf



* 1. **使用root用户执行命令：rm /tmp/trigger\_file0，或者重启服务器，重启后系统将自动删除该文件,**如果不执行该动作，我们在启动数据库的时候，就会出现错误（具体情况在J步骤中说明）
  2. 执行命令pg\_ctl start



* 1. 成功竟然流复制模式，查看slave2中是否有数据插入：



* 1. 忽略f步骤中，在M-2上不执行删除/tmp/trigger\_file0的动作，数据库启动，完成恢复后，直接进入可以读写模式，这不是我们想要的流模式，因为流模式只能进行读操作。具体的大家可以去试试，这里就不做了。因为这个问题，纠结了我2天。。。。囧

好了，到这里，我们已经完整的配置好了pgpool的流复制模式。