yum -y install readline readline-devel bison bison-devel flex flex-devel zlib-devel

groupadd postgres

useradd -g postgres postgres

cd /usr/local/src

cd postgresql-9.2.2

./configure --prefix=/usr/local/pgsql

gmake world

gmake install -world

mkdir /home/pg\_data #数据目录 -->丛库也创建相应目录

mkdir /home/pg\_archive #归档目录 -->丛库也创建相应目录

mkdir /home/pg\_log #CSV日志目录 -->丛库也创建相应目录

chown -R postgres:postgres /home/pg\_\*

## pg\_stat\_statements 先验证，上面的步骤已经安装所有模块

ls /usr/local/pgsql/share/extension/

cd contrib/pg\_stat\_statements

make

make install

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

配置环境变量，在/home/postgres/.bash\_profile文件末尾添加如下内容

export PGPORT=5432 ---丛库加个export PGPORT2=5433

export PGDATA=/home/pg\_data ---丛库加个 export PGDATA2=/home/pg\_data\_n

export PGHOME=/usr/local/pgsql

export LD\_LIBRARY\_PATH=$PGHOME/lib:/lib64:/usr/lib64:/usr/local/lib64:/lib:/usr/lib:/usr/local/lib:$LD\_LIBRARY\_PATH

export DATE=`date +"%Y%m%d%H%M"`

export PATH=$PGHOME/bin:$PATH:.

export MANPATH=$PGHOME/share/man:$MANPATH

export PGUSER=postgres

export PGDATABASE=postgres

alias rm='rm -i'

alias ll='ls -lh'

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

加载环境变量

shell# su - postgres

shell# source .bash\_profile

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

初始化数据库集群,配置postgres用户密码

shell# initdb -D $PGDATA -E UTF-8 --locale=C -U postgres -W （初始化数据库） 丛库也初始化，注意路径

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

修改pg数据库配置文件，修改完成后如下：标红的更改，其他主从一样

shell# cat /home/pg\_data/postgresql.conf | egrep -v '^#|^$' | grep '^[a-zA-Z]'

listen\_addresses = '\*' # what IP address(es) to listen on;

port = 5432 # (change requires restart)

max\_connections = 1000 # (change requires restart)

shared\_buffers = 5GB # min 128kB

temp\_buffers = 128MB # min 800kB

work\_mem = 12MB # min 64kB

maintenance\_work\_mem = 64MB # min 1MB

shared\_preload\_libraries = 'pg\_stat\_statements' # (change requires restart)

pg\_stat\_statements.max = 1000

pg\_stat\_statements.track = all

wal\_level = hot\_standby # minimal, archive, or hot\_standby

checkpoint\_segments = 90 # in logfile segments, min 1, 16MB each

checkpoint\_timeout = 20min # range 30s-1h

checkpoint\_completion\_target = 0.8 # checkpoint target duration, 0.0 - 1.0

checkpoint\_warning = 0 # 0 disables

archive\_mode = on # allows archiving to be done

archive\_command = 'test ! -f /home/pg\_archive/%f && cp %p /home/pg\_archive/%f'

# command to use to archive a logfile segment

archive\_timeout = 0 # force a logfile segment switch after this

max\_wal\_senders = 6 # max number of walsender processes

wal\_keep\_segments = 500 # in logfile segments, 16MB each; 0 disables

synchronous\_standby\_names = '' # standby servers that provide sync rep

hot\_standby = on # "on" allows queries during recovery

max\_standby\_streaming\_delay = 30s # max delay before canceling queries

wal\_receiver\_status\_interval = 10s # send replies at least this often

effective\_cache\_size = 5GB

log\_destination = 'csvlog' # Valid values are combinations of

logging\_collector = on # Enable capturing of stderr and csvlog

log\_directory = '/home/pg\_log' # directory where log files are written,

log\_filename = 'postgresql-%Y-%m-%d\_%H%M%S.log' # log file name pattern,

log\_rotation\_age = 1d # Automatic rotation of logfiles will

log\_rotation\_size = 500MB # Automatic rotation of logfiles will

log\_statement = 'mod' # none, ddl, mod, all

log\_timezone = 'America/Manaus'

track\_activity\_query\_size = 3096 # (change requires restart)

datestyle = 'iso, mdy'

timezone = 'America/Manaus'

lc\_messages = 'C' # locale for system error message

lc\_monetary = 'C' # locale for monetary formatting

lc\_numeric = 'C' # locale for number formatting

lc\_time = 'C' # locale for time formatting

default\_text\_search\_config = 'pg\_catalog.english‘

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

开启数据库

shell# pg\_ctl -D $PGDATA -l /home/pg\_log/pgsql.log start

shell# netstat -anp | grep 5432

shell# ps -ef|grep postgresql

shellpg\_ctl -D /home/pg\_data restart -m fast

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

-------------------------postgres的从配置

备份数据库

shell# pg\_basebackup -D /home/pg\_data -Fp -Xs -v -P -h source\_ip -p 5432 -U postgres

这里使用了pg\_basebackup这个命令，/home/pg\_data这个目录是空的

成功之后，就可以看到这个目录中现有的文件都是一样的了。

进入到/data/pgsql/data2目录，复制recovery.conf，这个文件可以从pg的安装目录的share文件夹中获取，比如

cp /usr/local/pgsql/share/recovery.conf.sample /home/pg\_data/recovery.conf

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

修改recovery.conf，只要修改几个地方就行了

standby\_mode = on # 这个说明这台机器为从库

primary\_conninfo = 'host=127.0.0.1 port=5432 user=postgres password=^regex$' # 这个说明这台机器对应主库的信息

recovery\_target\_timeline = 'latest' # 这个说明这个流复制同步到最新的数据

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

确认主库和从库都配置好了

查看进程，主库所在的机器中会看到sender进程

8467 postgres 20 0 255m 2396 1492 S 0.0 0.1 0:00.66 postgres: wal sender process replica

从库所在的机器中会看到receiver进程

8466 postgres 20 0 298m 1968 1096 S 0.0 0.1 0:06.88 postgres: wal receiver process streaming 3/CF118C18

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

查看复制状态

主库中执行：

postgres=# select \* from pg\_stat\_replication;

-[ RECORD 1 ]----+------------------------------

pid | 8467 # sender的进程

usesysid | 44673 # 复制的用户id

usename | replica # 复制的用户用户名

application\_name | walreceiver

client\_addr | 10.12.12.12 # 复制的客户端地址

client\_hostname |

client\_port | 55804 # 复制的客户端端口

backend\_start | 2015-05-12 07:31:16.972157+08 # 这个主从搭建的时间

backend\_xmin |

state | streaming # 同步状态 startup: 连接中、catchup: 同步中、streaming: 同步

sent\_location | 3/CF123560 # Master传送WAL的位置

write\_location | 3/CF123560 # Slave接收WAL的位置

flush\_location | 3/CF123560 # Slave同步到磁盘的WAL位置

replay\_location | 3/CF123560 # Slave同步到数据库的WAL位置

sync\_priority | 0 #同步Replication的优先度

0: 异步、1～?: 同步(数字越小优先度越高)

sync\_state | async # 有三个值，async: 异步、sync: 同步、potential: 虽然现在是异步模式，但是有可能升级到同步模式

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

连接成功之后所有的命令都是使用”\“+ 字符或者word完成相应的功能。现将常用的几个列车

\l 列出所有数据库

\dt 列出连接数据库中所有表

\di 列出连接数据库中所有index

\dv 列出连接数据库中所有view

\h sql命令帮助

\? \ 所有命令帮助

\q 退出连接

\d tablename 列出指定tablename的表结构