“Hot Standby Streaming Replication” Sunucu Yapılandırması

Bahadır Demircioğlu

Nisan, 2016

Öncelikli olarak tabii ki postgreSQl kurulumunun yapılması gerekiyor. Ben depolardan kurmak yerine son kararlı sürümü EnterpriseDB sitesinden indirip kuruyorum. Bunun için;

http://www.enterprisedb.com/products-services-training/pgdownload

adresinden uyumlu sürümü indirip kurabilirsiniz.
İkinci olarak makinelerimiz;

```
Master: 192.168.222.100
Slave(Standby):192.168.222.101
```

Kurulum tamamlandıktan sonra adım adım ilerlemeye başlayabiliriz.

1) /opt/PostgreSQL/9.2 dizininin sahibinin postgres olması gerekmekte.

```
chown -R postgres /opt/PostgreSQL/9.2
```

2) Postgres kullanıcısı için şifresiz haberleşmeyi sağlamak için master node'da ssh key oluşturacağız.

```
su -postgrtes
ssh-keygen -t rsa
```

3) Key dosyasını standby sunucu için postgres kullanıcısına atalım.

```
ssh-copy-id -i /opt/PostgreSQL/9.2/.ssh/id_rsa.pub 192.168.222.101
```

4) Slave node üzerindeki postgres kullanıcısınında master node'a şifresiz erişimi için slave node'da ssh key oluşturacağız ve master node'da göndereceğiz.

```
ssh-keygen -t rsa
ssh-copy-id -i ~/.ssh/id_rsa.pub 192.168.222.100
```

Herhangi bir sorun yok ise artık yapılandırmaya başlayabiliriz.

5) Master üzerindeki postgresql.conf dosyası üzerinde;

```
listen_addresses = "*"
wal_level = hot_standby
checkpoint_segments=30
archive_mode=on
archive_command='cd .'
max_wal_senders=2
wal_keep_segments=5000
hot_standby=on
```

olarak yapılandıralım.

6) Master node üzerinde pg_hba.conf dosyası;

```
host all all 192.168.222.1/24 trust
host replication all 192.168.222.1/24 trust
```

7) Şu an için postgresql yapılandırmamız bitti. Test database'i oluşturalım:

```
su -postgres
createdb pgbench
```

8) Database üzerinde biraz data oluşturalım.

```
pgbench -i -s 10 pgbench
```

9) Slave node üzerindeki bütün datayı silelim bakalım ne olacak! Öncelikle tabii ki postgrs sunucusunu kapatalım.

```
/etc/init.d/postgresql-9.2 stop
rm -rf /opt/PostgreSQL/9.2/data/*
```

10) Master node üzerinde database'e bağlanabilirliği test edelim. Bu komutu slave node üzerinden vereceğiz.

```
psql -h 192.168.222.100 -d pgbench
```

Bağlantıda sorun yok ise artık repgmr uygulamasını kurup yapılandırmaya başlayabiliriz.

11) repmgr kuralım. Buynun için source code'dan derleme yapacağız. Bu adımları hem master hem de slave için yapıyoruz.

Yüklememiz gerek paketler:

```
apt-get install make gcc libpam0g-dev openssl libkrb5-dev libssl-dev libpam-dev libxslt-dev
    libedit-dev
```

Dikkat edilmesi gereken şey postgresql'in path de tanımlı olması gerektiği.

```
wget http://www.repmgr.org/download/repmgr-1.2.0.tar.gz
tar xvfz repmgr-1.2.0
cd repmgr-1.2.0
make USE_PGXS=1
make USE_PGXS=1 install
```

12) Master node'un klonunu alacağız. Bunu slave node üzerinde yapıyoruz.

```
su - postgres
repmgr -D /opt/PostgreSQL/9.2/data/ -d pgbench -p 5432 -R postgres --verbose standby clone
    192.168.222.100
```

13) Artık slave node'u çalıştırabiliriz.

```
/etc/init.d/postgresql start
```

14) repmgr.conf dosyası süzenleyeceğiz. Master için;

```
cluster=test
node=1
conninfo='host=192.168.222.100 user=postgres dbname=pgbench'
```

Slave için;

```
cluster=test
node=2
conninfo='host=192.168.222.101 user=postgres dbname=pgbench'
```

15) Register etmek için;

```
master ------------------>> repmgr -f /path/to/repmgr.conf --verbose master register
slave  ------------------>> repmgr -f /path/to/repmgr.conf --verbose standby register
```

16) İşlem bitti. Atık test etmek için master node üzerinde işlemler yapıp slave node üzerinde var mı bakalım.

```
psql pgbench -c "create table test ( test varchar(30));"
psql pgbench -c "insert into test values ( 'test123');"
```

```
psql -h 192.168.222.101 pgbench -c "select * from test"
```

```
postgres@slave:~$ psql pgbench -c "select * from test"
Password:
  test
-------
 test123
(1 row)
```