PostgreSQL Sunucu ve İstemci Komutları

Bu makale de PostgreSQL 8.0 ile gelen istemci ve sunucu komutları anlatılacaktır. Bu belgenin en son haline http://www.enderunix.org/docs/postgresql/postgresql_commands.pdf adresinden ulaşabilirsiniz.

Yazar: İsmail YENİGÜL
EnderUNIX Çekirdek Takım Üyesi
ismail at enderunix dot org
ismail.yenigul at endersys dot com
http://www.enderunix.org

Değişiklikler:
27 Mar 2006 Pts EEST 15:29:00 – İlk Yazım

İçindekiler

1. Sunucu Uygulama Komutları

1.1. initdb

initdb komutu yeni bir PostgreSQL veritabanı kümelemesi (database cluster) oluşturmak için kullanılır. Buradaki database cluster ifadesi tek bir PostgreSQL sunucusu tarafından yönetilen tüm veritabanları demektir.
veritabanı kümelemesi oluşturma işleminde aşağıdaki işler yapılır.

- Verilerin saklanacağı dizin oluşturulur. Bu dizin adı genelde /usr/local/pgsql/data'dır.
- Tüm kümelemeye ait katalog tabloları oluşturulur. Sistem kataloglarında RDMS (Relational Database Management System)'e ait tablo, sütun, kullanıcı vb bilgileri ve veritabanına ait iç bilgileri tutmaktadır.
- template0 ve template1 veritabanı oluşturulur. template1 veritabanında oluşturulan her şey yeni oluşturulan veritabanlarını otomatik olarak kopyalanacaktır

initdb komutunun PostgreSQL kullanıcısı ile çalıştırılması gerekir. Fakat bu komutu çalıştırmadan önce data dizini oluşturulmalıdır. Dizin root kullanıcısı ile oluşturulduktan sonra dizin sahibi PostgreSQL kullanıcısı yapılabilir.

```
root# mkdir /usr/local/pgsql/data
root# chown postgres /usr/local/pgsql/data
root# su  - pgsql  (Burada kullanıcı adı olarak pgsql seçilmiştir)
```

initdb aynı zamanda veritabanı için ön tanımlı locale ve karakter setini tanımlamaktadır. Kullanılacak locale değeri en başta doğru seçilmelidir. Çünkü LC_COLLATE (karakter sıralama düzenini belirler) ve LC_CTYPE (Character Classification – harfleri ve büyük harf karşılıklarını belirler) değerlerini daha sonra değiştirmenin imkânı yoktur.
Karakter seti ise yeni bir veritabanı oluştururken belirlenebilir. Veritabanı oluşturulurken karakter seti özel olarak belirtilmezse ön tanımlı olanı kullanılır.

Parametreleri

`-D dizin`
`--pgdata=dizin`
Veritabanı kümelemenin oluşturulacağı data dizinini belirtir. Bu parametre verilmeze Data dizininin adı kullanıcının PGDATA çevre değişkeninden alınır.

`-E encoding`
`--encoding=encoding`
Kullanılacak karakter setini belirler.

`--locale=`*`locale`*
Ön tanımlı locale değerini belirler.

1.2. pg_controldata

PostgreSQL veritabanı kümelemesinin kontrol bilgilerini görüntüler.

```
# su - pgsql
$ pg_controldata
pg_control version number:            74
Catalog version number:               200411041
Database system identifier:           4908221539622420991
Database cluster state:               in production
pg_control last modified:             Çar 22 Mar 14:45:33 2006
Current log file ID:                  0
Next log file segment:                1
Latest checkpoint location:           0/C47D9C
Prior checkpoint location:            0/C41C24
Latest checkpoint's REDO location:    0/C47D9C
Latest checkpoint's UNDO location:    0/0
Latest checkpoint's TimeLineID:       1
Latest checkpoint's NextXID:          2416
Latest checkpoint's NextOID:          25438
Time of latest checkpoint:            Çar 22 Mar 14:45:33 2006
Database block size:                  8192
Blocks per segment of large relation: 131072
Bytes per WAL segment:                16777216
Maximum length of identifiers:        64
Maximum number of function arguments: 32
Date/time type storage:               floating-point numbers
Maximum length of locale name:        128
LC_COLLATE:                           tr_TR.ISO8859-9
LC_CTYPE:                             tr_TR.ISO8859-9
$
```

1.3. pg_ctl

pg_ctl PostgreSQL'i başlatmak, durdurmak, yeniden başlatmak ve servisin durumu hakkında bilgi veren bir uygulamadır.
Kullanım şekilleri:

```
pg_ctl start [-w] [-s] [-D datadir] [-l filename] [-o options] [-p path]
pg_ctl stop [-W] [-s] [-D datadir] [-m s[mart] | f[ast] | i[mmediate] ]
pg_ctl restart [-w] [-s] [-D datadir] [-m s[mart] | f[ast] | i[mmediate] ] [-o options]
pg_ctl reload [-s] [-D datadir]
pg_ctl status [-D datadir]
pg_ctl kill [signal_name] [process_id]
```

-D: data dizini
-l: Kayıt (log) dosyasının yolunu belirtir.
-m: servisin hangi kipte kapatılacağını belitir. Üç farklı tip vardır.

- Smart: Tüm istemciler bağlantıyı koparana kadar bekler. (Ön tanımlı)
- Fast: İstemcilerin bağlantıyı koparmasını beklemez. Tüm aktif işlemler (transaction) geri alınır (roll back) ve servis kapatılır.
- Immediate: Tüm PostgreSQL prosesleri düzgün kapanmadan sonlandırılır. Bu işlem servisin tekrar açıldığında kurtarma (recovery) işleminin yapılmasına sebep olur.

-s Bilgi mesajlarını göstermez sadece hataları gösterir.
-w: prosesi açma veya kapama işlemi gerçekleşene kadar bekler.

Örnekler:

```
# su  - pgsql
$ pg_ctl status
pg_ctl: postmaster is running (PID: 17264)
/usr/local/bin/postgres -D /usr/local/pgsql/data
$ pg_ctl start -D /usr/local/pgsql/data –l logfile
postmaster starting
$ pg_ctl stop -D /usr/local/pgsql/data -m smart
waiting for postmaster to shut down.... done
postmaster stopped

$  pg_ctl start -w -l logfile
waiting for postmaster to start.... done
postmaster started
$
```

1.4. postmaster

postmaster çok kullanıcılı PostgreSQL sunucusudur. postmaster istemci bağlantıları için postgres isimli ayrı bir proses başlatır. postmaster ek olarak sunucu prosesleri arasındaki bağlantıyı yönetir.

Ön tanımlı olarak ön planda (foreground) çalışır ve hata mesajlarını ekrana basar. Ön tanımlı PostgreSQL data dizini yoktur. Bu yüzden data dizini mutlaka –D parametresi ile verilmelidir.

Parametreleri:

-D: data dizini
-d: debug seviyesi
-l: SSL bağlantısını etkinleştirir (sertifikaların oluşturulmuş olması gerekir)
-p: port numarası (ön tanımlı olarak 5432)
-S: Sessiz mod. Bu moda prosesler arka planda (background) çalıştırır, çıktı ve hatalar /dev/null dosyasına gönderilir.
-h: hostname/IP (hangi hostname veya IP adresini dinleyeceğini belirtir)

Örnek:
```
$ postmaster –D /usr/local/pgsql/data
```

1.5. postgres

postgres programı sorguları işleyen PostgreSQL sunucu prosesidir. Doğrudan çalıştırılmaz bunun yerine postmaster prosesi tarafından çalıştırılır. postgres PostgreSQL sunucusunu tek kullanıcı modda çalıştırmak için kullanılır. Bu modda genelde kurtarma veya hata ayıklama durumlarında çalıştırılır.

2. İstemci Uygulama Komutları

2.1. createdb

PostgreSQL sunucu üzerinde yeni bir veritabanı oluşturmak için kullanılır.

Kullanım şekli
`createdb` [*option*...] [*dbname*] [*description*]

`-E` *encoding*
`--encoding` *encoding* : Karakter setini belirler.

`-O` *owner*
`--owner` *owner*: Yeni veritabanının sahibini belirler.

`-h` *host*
`--host` *host:* Yeni veritabanın oluşturulacağı sunucu adını belirlir.

`-p` *`port`*
`--port` *`port:`* Veritabanı oluşturulacak sunucunun port numarasını belirtir.

Örnek:

```
$ createdb denemedb;
CREATE DATABASE
$ createdb -E LATIN1 denemetr;
CREATE DATABASE
devel~$ psql -l
      List of databases
   Name    | Owner | Encoding
-----------+-------+----------
 deneme    | pgsql | LATIN5
 denemedb  | pgsql | LATIN5
 denemetr  | pgsql | LATIN1
 enderunix | pgsql | LATIN5
 mydb      | pgsql | LATIN5
 template0 | pgsql | LATIN5
 template1 | pgsql | LATIN5
(7 rows)

$
```

2.2. dropdb

PostgreSQL sunucudaki veritabanını siler. –h parametresi ile sunucu adı –p ile de port belirtilebilir.

Örnek:

```
$dropdb deneme;
```

2.3. createuser

Yeni bir PostgreSQL kullanıcısı oluşturmak için kullanılır.

Kullanım Şekli:
`$ createuser [`*`option`*`...] [`*`username`*`]`

Seçenekler:

-a: Oluşturulan kullanıcı başka kullanıcılar açabilir.
-A : Yeni kullanıcı açamaz.
-d: Kullanıcı veritabanı oluşturabilir.
-D: Kullanıcı veritabanı oluşturamaz.

Aşağıdaki her iki komutta aynı manaya gelmektedir.

```
devel~$ createuser bsd
```

```
Shall the new user be allowed to create databases? (y/n) y
Shall the new user be allowed to create more new users? (y/n) n
CREATE USER
devel~$ createuser -A -d bsd
CREATE USER
devel~$
```

2.4. dropuser

Veritabanından kullanıcı silmek için kullanılır. Kullanıcı –i parametresi ile interaktif olarak silinebilir.

```
devel~$ dropuser -i bsd
User "bsd" will be permanently removed.
Are you sure? (y/n) y
DROP USER
devel$
```

2.5. pg_config

Kurulu PostgreSQL sunucunun yapılandırma bilgilerini gösterir. PostgreSQL'de verilerin nerelere kurulduğu görmek için kullanılabilir.

Kullanım Şekli:

```
$ pg_config {--bindir | --includedir | --includedir-server | --libdir
| --pkglibdir | --pgxs | --configure | --version}
```

--version: PostgreSQL versiyonunu gösterir.
--configure: Kurulum sırasındaki konfigürasyon parametrelerini gösterir.
--bindir: Binary dosyaların yerini gösterir.
--includedir: Başlık dosyalarının yerini gösterir.

Örnekler:

```
devel~$ pg_config --version
PostgreSQL 8.0.7
devel~$ pg_config --bindir
/usr/local/bin
devel~$ pg_config --includedir
/usr/local/include
devel~$ pg_config --libdir
/usr/local/lib
devel~$ pg_config --configure
'--with-libraries=/usr/local/lib' '--with-
includes=/usr/local/include' '--with-
docdir=/usr/local/share/doc/postgresql' '--with-openssl' '--disable-
```

```
nls' '--prefix=/usr/local' 'i386-portbld-freebsd4.11' 'LDFLAGS= -
rpath=/usr/lib:/usr/local/lib -L/usr/local/lib -lgnugetopt' 'CFLAGS=-
O2 -pipe -march=pentiumpro' 'CPPFLAGS=-I/usr/local/include'
'host_alias=i386-portbld-freebsd4.11' 'build_alias=i386-portbld-
freebsd4.11' 'target_alias=i386-portbld-freebsd4.11' 'CC=cc'
devel~$
```

2.6. psql

psql terminal tabanlı PostgreSQL sorgu arabimidir.

Bazı Parametreler:

-d veritabanı
-f dosya adı: SQL sorgularının dosyadan okunmasını sağlar.
-o dosya adı: Sorgu sonuçlarını belirtilen dosyaya yazar.
-h hostname: PostgreSQL sunucu adı / IP adresi
-p port: PostgreSQL sunucu portu.
-U kullanıcı adı: Veritabanına bağlanacak kullanıcı adını belirler.
-V: psql versiyonunu belirtir.

Örnek:

```
devel~$ psql deneme
Welcome to psql 8.0.7, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit

deneme=#
```

2.6.1. Kısa Yollar

Psql komutu ile veritabanına bağlandıktan sonra aşağıdaki kısa yollar kullanılarak veritabanı ve nesneler hakkında bilgiler alınabilir.

\? : Tüm kısa yolları ve parametrelerini listeler.
\c[onnect] [DBNAME|- [USER]] *:* Yeni bir veritabanına bağlantı kurar.
Örnek:

```
deneme=# \c template1
You are now connected to database "template1".
template1=# \connect deneme
You are now connected to database "deneme".
```

```
deneme=#
```

\q : psql'den çıkılır.
\l : Tüm veritabanlarını listeler. Sonuna + eklenerek daha detaylı bilgi alınabilir.

```
deneme=# \l
     List of databases
   Name    | Owner | Encoding
-----------+-------+----------
 deneme    | pgsql | LATIN5
 enderunix | pgsql | LATIN5
 template0 | pgsql | LATIN5
 template1 | pgsql | LATIN5
(4 rows)

deneme=# \l+
                 List of databases
   Name    | Owner | Encoding |        Description
-----------+-------+----------+---------------------------
 deneme    | pgsql | LATIN5   |
 enderunix | pgsql | LATIN5   |
 template0 | pgsql | LATIN5   |
 template1 | pgsql | LATIN5   | Default template database
(4 rows)

deneme=#
```

\dt: Tabloları listeler.
\d isim: Belirtilen isimdeki tablo, index vs hakkında bilgi verir.

```
deneme=# \dt
      List of relations
 Schema |  Name  | Type  | Owner
--------+--------+-------+-------
 public | deneme | table | pgsql
 public | test   | table | pgsql
(2 rows)

deneme=# \d test
                                  Table "public.test"
 Column |         Type          |                      Modifiers
--------+-----------------------+-----------------------------------
------------------
 id     | integer               | not null default
nextval('public.test_id_seq'::text)
 name   | character varying(30) |

deneme=#
```

2.7. pg_dump

PostgreSQL'de tanımlı tek bir veritabanının yedeğini düz-metin dosyaya veya arşiv dosyasına alır.

Kullanım şekli:
`pg_dump` [*option*...] [*dbname*]

Bazı Parametreler:

-a: Sadece verileri alır.
-f *dosya*: çıktıyı belirtilen dosyaya yazar.
-F *format*: Çıktının türünü belirler. Format türleri:
P: Düz metin çıktı (ön tanımlı değer)
t: tar arşivi oluşturur. Bu çıktı daha sonra pg_restore komutu ile yedekten dönmek için kullanılabilir.
-o: Tablolardaki verilere ait OID (Object IDentifier) değerlerinin yedeğini de alır. Bu parametre sadece sütunların OID değerlerine referans verildiği durumlarda kullanılır.
-t *tablo*: Sadece ilgili tablonun yedeği alınır.

Örnek:

```
devel~$ pg_dump  deneme >db.dump
devel~$ pg_dump –f db.dump deneme

devel~$ pg_dump -F t deneme >yedek.tar
devel~$ tar tvf yedek.tar
tar: Record size = 16 blocks
-rw------- 2048/1024      1965 27 Mar 17:11 2006 toc.dat
-rw------- 2048/1024        12 27 Mar 17:11 2006 1473.dat
-rw------- 2048/1024         5 27 Mar 17:11 2006 1474.dat
-rw------- 2048/1024      2060 27 Mar 17:11 2006 restore.sql
$
```

2.8. pg_dumpall

PostgreSQL'de tanımlı tüm bir veritabanlarının yedeğini düz-metin dosyaya alır.

Kullanım şekli:
`pg_dumpall` [*option*...]

Bazı Parametreler:

-a: Sadece verileri alır.
-o: Tablolardaki verilere ait OID (Object IDentifier) değerlerinin yedeğini de alır. Bu parametre sadece sütunların OID değerlerine referans verildiği durumlarda kullanılır.

```
devel~$ pg_dumpall >all.dump
```

Yeniden tüm veritabanını yüklemek için

```
devel~$ psql –f all.dump template1
```

2.9. pg_restore

pg_dump tarafından oluşturulan arşiv formatındaki yedekten geri dönülmesini sağlar. Arşiv dosyasının düz-metin dosyadan farkı veritabanının yedeğini saklandığı andaki haliyle geri dönmeyi sağlar. Ek olarak sadece belirli kısımların da yedekten dönmesi sağlanabilir.

Devam edecek...

3. Kaynaklar

http://www.postgresql.org/docs/8.0/interactive/index.html