

# Belajar Cepat PostgreSQL

Apabila Anda memerlukan sistem penyimpanan data yang luar biasa di Linux, maka gunakanlah PostgreSQL. Walaupun merupakan sistem yang kompleks, kita tetap bisa menggunakannya dengan mudah.

**D**ata adalah hal terpenting dalam komputasi. Kehilangan data adalah musibah besar. Dan cara mengelola data secara berantakan adalah pemicu kehilangan data. Apabila data Anda cukup banyak dan merupakan data rahasia, maka Anda membutuhkan sebuah database server yang layak untuk menyimpannya. Menyimpan data ke dalam sebuah file text adalah tindakan konyol yang dapat memicu berbagai risiko hilangnya data.

Puluhan tahun sudah sistem database mengalami perubahan. Mungkin sudah terlalu banyak sistem database dan aplikasinya yang telah tumbuh dan kemudian berubah sesuai dengan perkembangan teknologi. Salah satu yang memegang peranan besar adalah sistem database yang dikenal dengan nama POSTGRES, yang dikembangkan di University of California at Berkeley pada tahun 1986. Waktu itu, POSTGRES mencetak perubahan besar yang segera membuat berbagai pihak tertarik. Setelah satu tahun dikembangkan, lahirlah versi demo-nya pada tahun 1987, yang segera dipertunjukkan di berbagai konferensi teknologi komputer saat itu.

Dan akhirnya, pada tahun 1989, lahirlah POSTGRES versi 1. Versi 2 yang datang dengan banyak perubahan desain pun diluncurkan satu tahun kemudian. Berkembang dan berkembang terus, pada tahun 1991, lahirlah POSTGRES 3, yang waktu itu menjadi begitu populer dan segera digunakan oleh banyak aplikasi bisnis. Pada tahun 1993, database ini makin populer lagi. Basis penggunaanya bertambah secara dramatis. Dan karena POSTGRES menjadi terlalu kompleks, dan jauh melebihi kebutuhan riset database, maka secara resmi proyek ini dihentikan setelah mencapai versi 4.2.

Pada tahun 1994, **Andrew Yu** dan **Jolly Chen** menambahkan kemampuan SQL ke dalam POSTGRES, yang kemudian

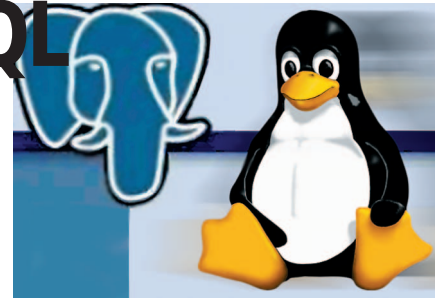
berubah menjadi proyek baru dengan nama Postgres95. Sistem database berbasis kode POSTGRES tersebut kemudian dilempar sebagai proyek *open source*, dan segera mendapatkan banyak tanggapan. Postgres95 memiliki banyak kelebihan dibandingkan dengan POSTGRES. Salah satunya adalah kecepatan dan kestabilan.

Pada tahun 1996, nama Postgres95 segera menjadi nama yang kadaluarsa. Kuno. Para pembuatnya kemudian mencari nama baru dan lahirlah PostgreSQL, yang mengacu kepada originasi dari POSTGRES yang memiliki kemampuan SQL. Dan uniknya, proyek ini dirilis langsung pada versi 6.0, mengacu kepada pengembangan POSTGRES yang seharusnya.

PostgreSQL adalah *object-relational database management system* yang datang dengan banyak fitur, di antaranya *multi-version concurrency control* (MVCC), mendukung hampir semua atura SQL, dan memiliki banyak dukungan bahasa pemrograman lain seperti C/C++, TCL, Java, Perl, dan Python. Dengan berbagai fitur tersebut, layaknya PostgreSQL mengklaim dirinya sebagai *most advanced database system*.

Proyek open source yang hebat umumnya relatif susah digunakan. Begitu pun dengan PostgreSQL. Dibandingkan dengan MySQL, PostgreSQL relatif lebih susah digunakan karena kompleksitasnya yang begitu besar. Namun, bagi Anda yang lebih peduli untuk merancang database yang bagus dan hanya sekedar menggunakannya, maka PostgreSQL tetap layak digunakan dan dapat dipelajari dalam waktu yang singkat, asalkan Anda telah mengerti konsep database dan SQL secara umum.

Apabila Anda menggunakan Debian (atau distro turunannya), maka berikanlah perintah berikut ini untuk menginstal postgresql:



```
apt-get install postgresql postgresql-client
postgresql-doc
```

PostgreSQL pun akan duduk dengan tenang di dalam sistem Anda. Untuk distro lain, gunakanlah manajemen paket dari distro Anda, atau carilah berbagai paket untuk postgresql server, *client*, dan dokumentasi. Umumnya, hampir setiap distro telah datang bersama database server ini. Anda juga dapat melakukan kompilasi sendiri dari source dengan *download source code*-nya terlebih dahulu dari [www.postgresql.org](http://www.postgresql.org).

Umumnya, server dari postgresql yang bernama *postmaster* akan dijalankan oleh *user postgres*. Menjalankan sebagai *user nobody* atau *root* sangat tidak disarankan. Sebagai contoh, kita menjalankan *postmaster* sebagai *user postgres*. Gantilah *user ini* dengan *user apapun* yang digunakan oleh distro Anda sebagai *user* yang menjalankan server postgresql.

Untuk berlatih, kita akan menggunakan program *client psql* terlebih dahulu, setelah itu, kita akan menggunakan *phpPgAdmin*. *Psql* lebih susah digunakan karena berbasis teks, sementara *phpPgAdmin* dijalankan lewat *web browser*.

PostgreSQL dapat menggunakan *identity-based authentication* dalam mengenali *user* yang akan menggunakan database server-nya. Untuk mudahnya, apabila Anda sedang login dengan *user nop* misalnya, maka kita juga akan membuat *user* di postgresql dengan nama *nop*. Gantilah *user nop* dengan *user lain* sesuai keinginan Anda. Sebagai catatan, *user authentication* pada postgresql sangatlah kompleks dan di luar cakupan artikel ini. Bacalah *administrator guide* yang datang bersamanya untuk otentikasi lanjutan.

Untuk membuat *user nop* di postgresql, jadilah *root*, kemudian, jadilah *user*

postgres. Berikut ini adalah perintahnya:

```
$ su
Password:
# su postgres
$
```

Setelah itu, gunakan program `createuser` untuk membuat user `nop` di `postgres`.

```
$ createuser nop
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
```

Apabila kita menjawab `y` pada pertanyaan pertama, maka user yang kita buat akan dapat membuat database sendiri. Kemudian, apabila kita menjawab `y` pada pertanyaan kedua, maka user yang dibuat akan dapat membuat user lain. Apabila program `createuser` berhasil, maka tulisan berikut ini akan tampil di layar:

**CREATE USER**

*Login*-lah kembali sebagai user `nop`. Kita akan membuat database baru dengan nama `nop`. Untuk itu, Anda bisa mempergunakan program `createdb`. Berikut ini adalah contohnya penggunaannya:

```
createdb nop
```

Apabila berhasil, maka tulisan berikut ini akan tampil di layar:

**CREATE DATABASE**

Pada saat pembuatan user `nop` oleh user `postgres`, kita mengizinkan `nop` untuk membuat database. Oleh karena itu, user `nop` bisa membuat banyak database. Untuk menghapus database, program `dropdb` dapat digunakan. Berikut ini adalah contoh penggunaan program `dropdb`:

```
dropdb nop2
```

Apabila berhasil, maka tulisan berikut ini akan tampil di layar:

**DROP DATABASE**

Kita telah memiliki database dengan nama `nop`. Sebentar lagi, kita akan masuk ke shell `postgres` sebagai user `nop`.

## Menggunakan `psql`

PostgreSQL datang dengan satu *client*

berbasis teks dengan nama `psql`. Apabila Anda *login* sistem dengan user `nop`, dan akan menggunakan database `nop` untuk bekerja, maka berikanlah perintah berikut ini untuk masuk ke server `postgres`:

```
$ psql nop
Welcome to psql 7.3.3, the PostgreSQL
interactive terminal.

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

nop=#
```

Selanjutnya, Anda dapat memberikan perintah khas `postgres` ataupun perintah SQL sesuai keinginan Anda. Berikut ini adalah beberapa perintah khas dan kegunaannya, yang diberikan pada program `psql`.

### Mencari bantuan

```
\?
```

### Melihat daftar database

```
\l
```

contoh keluaran:

List of databases		
Name	Owner	Encoding
----	-----	-----
nop	nop	SQL_ASCII
template0	postgres	SQL_ASCII
template1	postgres	SQL_ASCII
(3 rows)		

### Melihat daftar user

```
\du
```

contoh keluaran:

List of database users		
User name	User ID	Attributes
----	-----	-----
nop	100	create database
postgres	1	superuser, create database
test	101	create database
(3 rows)		

### Melihat daftar tabel

```
\dt
```

# IKLAN

Contoh keluaran:

List of relations			
Schema	Name	Type	Owner
public	test	table	nop
(1 row)			

## Keluar dari psql

```
\q
```

Berikut ini kita akan membuat tabel A yang memiliki dua *field*, yaitu nama dan hobi, yang masing-masing bertipe *varchar*.

```
CREATE TABLE A(
```

```
nama varchar(50),
```

```
hobi varchar(100)
```

```
);
```

Apabila perintah tersebut berhasil, maka tulisan CREATE TABLE akan ditampilkan ke layar.

Setelah itu, kita akan mengisi dua *record* ke dalam tabel A tersebut. Berikut ini adalah pemberian perintah dan keluarannya:

```
nop=# INSERT INTO A(nama, hobi)
VALUES ('nop','terbang');
```

```
INSERT 16995 1
```

```
nop=# INSERT INTO A(nama, hobi)
VALUES ('nop2','makan');
```

```
INSERT 16996 1
```

Perintah berikut ini berguna untuk menampilkan semua record di dalam tabel A:

```
nop=# SELECT * FROM A;
```

nama	hobi
nop	terbang
nop2	makan

```
(2 rows)
```

## Menggunakan phppgadmin

Apabila psql dirasa terlalu susah, kita dapat mempergunakan phppgadmin, yang jelas jauh lebih mudah untuk digunakan. Berikan perintah berikut ini untuk menginstal phppgadmin. Anda akan membutuhkan Apache dan PHP.

Berikan perintah berikut ini untuk menginstall phppgadmin:

```
apt-get install phppgadmin
```

Jalankan apache Anda. Kemudian bukalah URL <http://localhost/phppgadmin/>

```
File Edit View Terminal Go Help
nop=# psql nop
Welcome to psql 7.3.3, the PostgreSQL interactive terminal.
Type: \copyright for distribution terms
      \h for help with SQL commands
      \? for help on internal alias commands
      \q or terminate with semicolon to execute query
      \q to quit

nop=# \l
      List of databases
Name | Owner | Encoding
-----+-----+-----
nop | nop | SQL_ASCII
template0 | postgres | SQL_ASCII
template1 | postgres | SQL_ASCII
(3 rows)

nop=# \d
      List of relations
Schema | Name | Type | Owner
-----+-----+-----+-----
public | a | table | postgres
public | test | table | nop
(2 rows)

nop=# select * from test;
name | hobi
-----+-----
nop | terbang
nop2 | makan
(2 rows)

nop=#
```

### Program Psql, susah namun fleksibel

dengan *web browser* Anda. Pertama-tama, lakukan otentikasi terlebih dahulu. Secara *default*, Anda tidak perlu memasukkan *password*.

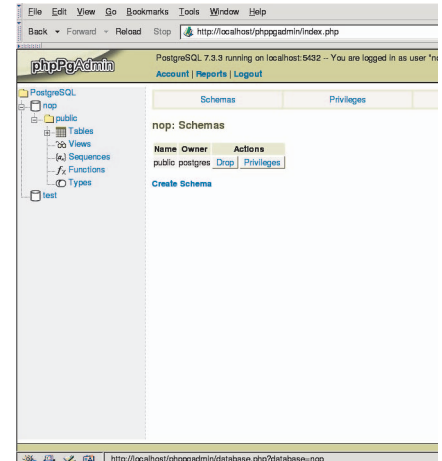
Setelah masuk, Anda dapat mempergunakan phppgadmin sebagai client untuk bekerja dengan server postgresql. Dengan phppgadmin, pengguna akan sangat banyak terbantu karena phppgadmin sangat menyederhanakan penggunaan postgresql. Klik sana, klik sini, isi beberapa nilai, dan Anda pun selesai. Bandingkan dengan program psql.

Sebagai contoh, kita akan membuat tabel baru, masih di dalam database nop. Kliklah tanda tambah pada database nop di frame kiri. Klik sekali lagi tanda tambah pada *public*. Seharusnya, Anda akan menjumpai tulisan *Tables* di sana. Kliklah *Tables* tersebut, dan lihatlah pada *frame* kanan.

Frame kanan akan mendaftar segala tabel Anda. Untuk membuat tabel baru, kliklah link *create table*. Anda akan diminta untuk mengisi nama tabel dan jumlah field-nya. Setelah semua langkah selesai, maka sebuah tabel baru telah Anda miliki.

Untuk menghapus tabel, kembalilah pada tampilan daftar tabel, kemudian kliklah tombol *Drop* untuk tabel yang bersangkutan.

Phppgadmin adalah salah satu client postgresql yang cukup mudah untuk digunakan. Apabila Anda malas menginstal Apache dan PHP, maka Anda dapat mempergunakan client lain yang berbasis GUI, seperti pgaccess.



### Lebih mudah dengan PHPPgAdmin

## Dump ke file teks

Apabila Anda hendak melakukan back-up, Anda tidak dapat meng-copy-kan begitu saja file-file database Anda. Sebagai gantinya, Anda dapat memasukkan semua isi database ke dalam sebuah file teks. File teks tersebut yang akan Anda bawa ke mana-mana.

PostgreSQL menyediakan program *pg\_dump* untuk memasukkan semua isi database ke file teks. Berikut ini adalah contoh penggunaannya:

```
pg_dump nop > nop.sql
```

Apabila Anda hanya ingin memasukkan isi tabel tertentu saja (misalnya tabel A pada database nop), maka berikanlah perintah berikut ini:

```
pg_dump nop --table=A > nop.A.sql
```

Selanjutnya, apabila Anda ingin memasukkan semua data yang telah didump ke dalam database server, gunakan kembali program psql, dengan cara sebagai berikut:

```
psql nop < nop.sql
```

PostgreSQL memang lebih susah dibandingkan dengan MySQL. Akan tetapi, fitur yang datang bersamanya juga sangat banyak, dibandingkan dengan MySQL. Namun, postgresql juga tidak cocok digunakan untuk segala situasi. Tidak semua data perlu disimpan di dalam postgresql. Gunakan postgresql apabila Anda benar-benar memerlukannya. Seperti kata pepatah, "gunakan alat yang sesuai untuk menyelesaikan masalah".  
Noprianto (noprianto@infolinux.co.id)