Datenbankadministration - Handreichung zur Präsentation

am Beispiel von mySQL und postgreSQL

Ahmad Nessar Nazar, Michael Dienert

28. November 2011

Inhaltsverzeichnis

1 Quelloffene Datenbanken **1**
1.1 mySQL 1
1.2 postgreSQL 2

2 MySQL und PostgreSQL installieren **2**
2.1 Warum selbst compilieren 2
2.2 MySQL compilieren, installieren, starten 3
2.3 postgreSQL compilieren, installieren, starten 4

3 Konfiguration von mySQL **5**
3.1 Die Benutzerdatenbank mysql 5
3.2 Eintrag in die Tabelle `user` 6

4 Konfigurationsdateien von postgreSQL **6**
4.1 Serverkonfiguration 6
4.2 Benutzerverwaltung 7

1 Quelloffene Datenbanken

1.1 mySQL

mySQL bekanntestes, am weitesten verbreitetes RDBMS

Dual License GPL (Community Version) und kommerzielle Lizenz

MySQL AB gegründet von Michael Widenius (Finne), David Axmark, Alan Larsson (Schweden) im Jahr 1994

Speichersubsysteme **MyISAM**: sehr schnell **Innodb**: transaktionssicher

2005 Oracle kauft Innobase Oy, Hersteller des mySQL-Speichersystems *InnoDB*

2008 SUN kauft MySQL AB für 1 Mrd (!) USD

2009 Oracle übernimmt SUN

1.2 postgreSQL

postgreSQL wird an der University of California, Berkeley seit Anfang der 80er (!) entwickelt. Leiter der Entwicklung: Michael Stonebreaker (inzwischen am MIT)

eigene Lizenz PostgreSQL-Lizenz ist ähnlich der BSD-Lizenz, nicht so streng wie GPL

SQL-Standards PostgreSQL erfüllt SQL92 und grosse Teile von SQL2008

Datenintegrität transaktionssicher, Fremdschlüssel und referentielle Integrität

Stored Procedures eigene Funktionen in PL/pgSQL, C, Perl, Java uvm. möglich

Betriebssysteme alle Unix-Derivate. Seit 2008 nativ unter Windows;

2 MySQL und PostgreSQL installieren

2.1 Warum selbst compilieren

- Traue keiner Software, die Du nicht selbst compiliert hast
- Quellen sind immer aktueller als Binärpakete
- Quellen sind grösstenteils **plattformunabhängig**
- beim Linken werden immer die vorliegenden Bibliotheken verwendet, so dass keine Versionsprobleme auftreten können
- Installationspfade und Eigenschaften sind **konfigurierbar**, unabhängig von der Distribution
- Die Konfiguration von MySQL ist in den Distributionen nicht einheitlich (openSuse, Ubuntu, Debian, RH).
- Ein selbst compiliertes und installiertes MySQL oder postgres liegt normalerweise (default-Einstellung) **komplett** (einschliesslich Datenbanken, logs, Dokus, etc.) in

```
/usr/local/mysql bzw. /usr/local/pgsql
```

2.2 MySQL compilieren, installieren, starten

- Quellen herunterladen. Z.B. so:

```
mysql-5.5.18.tar.gz
```

- Paket im Download-verzeichnis auspacken:

```
tar -xzf mysql-5.5.18.tar.gz
```

- bei Bedarf diverse Pakete nachinstallieren (cmake ist zwingend):

```
aptitude install cmake
aptitude install libncurses5-dev
aptitude install build-essential
```

- Benutzer `mysql` und Gruppe `mysql` anlegen:

```
groupadd mysql
useradd -g mysql mysql
```

- MySQL bauen und installieren:

```
cd mysql-5.5.18
cmake .
make
make install
```

- Eigentümer anpassen:

```
cd /usr/local/mysql
chown -R mysql:mysql .
```

- Datenbank 'mysql' installieren

```
./scripts/mysql_install_db --user=mysql
  \ --basedir=/usr/local/mysql
```

- Suchpfad erweitern:

```
export PATH=$PATH:/usr/local/mysql/bin
```

- MySQL-Server (als Superuser) starten:

```
mysqld_safe --datadir=/usr/local/mysql/data &
```

- Mit der Systemdatenbank 'mysql' verbinden. Diese Datenbank enthält die komplette Rechteverwaltung von MySQL:

```
mysql mysql
```

Das erste Wort 'mysql' ist der Name des MySQL-Client-Kommandos, das zweite 'mysql' ist der Name der Datenbank.

- Für später: mysqld wieder stoppen:

```
mysqladmin shutdown
```

2.3 postgreSQL compilieren, installieren, starten

- Quellen herunterladen. Z.B.

```
postgresql-9.1.1.tar.gz
```

- Paket im Downloadverzeichnis auspacken:

```
tar -xzf postgresql-9.1.1.tar.gz
```

- bei Bedarf diverse Pakete nachinstallieren (ubuntu/debian):

```
aptitude install libreadline6-dev
aptitude install lib32z1-dev
```

- Konfigurieren, Compilieren und Installieren. Das Compilieren dauert einige Minuten, da sollte man einen Kaffee trinken gehen:

```
cd postgresql-9.1.1
./configure
make
make install
```

- Benutzer `postgres` anlegen. Nur das Passwort angeben, die restlichen interaktiven Angaben leer lassen. Der Benutzer muss nicht *postgres* heissen, man kann auch einen anderen Namen wählen:

```
adduser postgres --no-create-home
```

- Das Verzeichnis `/usr/local/pgsql/data` anlegen. Das soll der Speicherort für die eigentlichen Datenbanken werden:

```
cd /usr/local/pgsql
mkdir data
chown -R postgres:postgres .
```

- Die Identität von User *postgres* annehmen und das Datenbankcluster initialisieren. Das Cluster soll das Verzeichnis `/usr/local/pgsql/data` sein.

```
su postgres
export PATH=$PATH:/usr/local/pgsql/bin
initdb -D data
```

- Den Server starten. Der Server lässt sich nicht unter der ID `root` starten, er muss also z.B. unter der Identität von `postgres` gestartet werden. Da wir aber schon `postgres` sind, funktioniert das Starten so:

```
postgres -D data &
```

- Wenn man den Server wieder stoppen möchte geht das so:

```
pg_ctl -D data stop
```

3 Konfiguration von mySQL

mySQL besitzt eine Konfigurationsdatei, die die Optionen für den Serverstart enthält.

Die Datenbankbenutzer dagegen werden nicht mit einer Datei, sondern mit der speziellen Datenbank `mysql` verwaltet. Hier ein **stark gekürztes** Beispiel der Datei `my.cnf`:

```
[client]
#password        = your_password
port               = 3306
socket               = /tmp/mysql.sock

[mysqld]
port               = 3306
socket               = /tmp/mysql.sock
```

3.1 Die Benutzerdatenbank mysql

Die Benutzerdatenbank `mysql` enthält 24 Tabellen. Die drei wichtigsten davon sind:

user Diese Tabelle gilt für den gesamten Datenbankserver. Hat ein Benutzer hier ein Recht vergeben bekommen, gilt dies für **alle** Datenbanken auf dem Server.

db Hiermit kann man Benutzern Rechte geben, die nur für eine **bestimmte** Datenbank gelten.

host Ist das Host-Attribut in der `db`-Tabelle **leer**, kommt diese Tabelle zum Einsatz: mySQL gewährt ein Recht dann nur, wenn in `db` und `host` ein 'Y' steht.

3.2 Eintrag in die Tabelle `user`

Eintragen eines Benutzers mit Rechten auf alle Datenbanken in die Tabelle `user`:

```
insert into user (host,user,password, select_priv,
            insert_priv, update_priv, delete_priv,
                         create_priv, drop_priv)
   values ('%', 'alfred', password('neu'),
                'Y','Y','Y','Y','Y','Y');
```

Ganz wichtig ist die Verwendung der Funktion password('xxyyzz'): Damit wird das Passwort verschlüsselt in die Tabelle eingetragen.

Genauso wichtig ist das Neueinlesen der Tabellen:

```
bin/mysqladmin reload
```

Das Ergebnis des INSERT-Befehls sieht so aus (es sind nur die ersten 3 Spalten der Tabelle gezeigt):

```
+-------------+--------+-------------------------------------------+
| host        | user   | password                                  |
+-------------+--------+-------------------------------------------+
| localhost   | root   |                                           |
| eddie.local | root   |                                           |
| 127.0.0.1   | root   |                                           |
| ::1         | root   |                                           |
| localhost   |        |                                           |
| eddie.local |        |                                           |
| %           | alfred | *3542A0A01117DE95A8C99813EA5EEECBC6601B28 |
+-------------+--------+-------------------------------------------+
```

Das %-Zeichen als Eintrag bei 'host' bedeutet soviel wie 'any host'.

Anmeldung am Server mit dem Client-Programm von einem entfernten Host aus:

```
mysql -h 192.168.2.109 -u alfred -pneu
```

4 Konfigurationsdateien von postgreSQL

postgreSQL besitzt zwei Konfigurationsdateien. Bei einer Quellinstallation stehen diese im Verzeichnis, das mit `initdb` angelegt wurde:

- `postgresql.conf` enthält allgemeine Parameter für den Server
- `pg_hba.conf` ist für die Client-Authentifizierung zuständig (**h**ost **b**ased **a**uthentication).

4.1 Serverkonfiguration

In `postgresql.conf` erlauben wir nur, dass sich Clients über das Internet anmelden können. Dazu müssen folgende Parameter angepasst werden:

- `listen_addresses = '*'`
- `port = 5432`

4.2 Benutzerverwaltung

In `pg_hba.conf` kann fein reguliert werden, welche Datenbankbenutzer sich von wo anmelden können und welche Passwort-Verfahren angewendet werden sollen. Exemplarisch wird nur ein Beispiel gezeigt und auf die hervorragende psql-Doku

```
http://www.postgresql.org/docs/9.1/static/auth-pg-hba-conf.html
```

verwiesen: Die Methode trust wurde durch md5 ersetzt und es wurden zwei Netze, aus denen heraus sich Clients verbinden dürfen, hinzugefügt:

```
# TYPE  DATABASE     USER     ADDRESS         METHOD

# "local" is for Unix domain socket connections only
local   all          postgres                trust

# IPv4 local connections:
host    all          all      127.0.0.1/32    md5
# IPv6 local connections:
host    all          all      ::1/128         md5

host    emaille      alfred   192.168.2.0/24  md5
host    all          all      10.16.0.0/12    md5
```

Um die Konfiguration zu testen, wird ein Benutzer und eine Datenbank angelegt:

```
createuser -Pse alfred
createdb -eO alfred emaille
```

verbinden kann man sich (auch von entfernten Rechnern) mit:

```
psql -U alfred -h 192.168.2.109 emaille
```