Using The Postgresql Database on Linux

“”

Contents

```
  This book purports to provide
   ⇒ concise and practical
   ⇒ recipes for
  using the open-source relational
   ⇒  (sql) database systems
   ⇒ Postgresql on linux

  @@ www.postgresql.org
     the official site
  @@ http://www.cyberciti.biz/faq/
   ⇒ psql-fatal-ident-
   ⇒ authentication-failed-for-
   ⇒ user/
     a good article about
      ⇒ authentication problems (
      ⇒ i.e. you cant logon)

 * Watch postgresql calls from
  ⇒ your application on localhost
 >> sudo tcpdump -nnvvXSs 1514 -i
  ⇒ lo0 dst port 5432

INSTALLING POSTGRESQL

  * install the postgresql
   ⇒ database on a debian-style
   ⇒ distribution
  >> sudo apt-get install
   ⇒ postgresql

AUTHENTICATION

  The process of logging onto
   ⇒ something is generally
   ⇒ referred to
  as 'authentication'. Postgresql
   ⇒ uses reasonably complicated
   ⇒ authentication
  which may cause some head-
   ⇒ scratching.

  * switch user to the 'postgres'
   ⇒ user, in order to administer
   ⇒  the server
  >> sudo su postgres  ##(only
   ⇒ necessary on some linux
   ⇒ distributions, at first)
  >> psql -U postgres  ##(should
   ⇒ achieve the same thing)

  * find out where the
   ⇒ authentication configuration
   ⇒  file is
  >> find / -name 'pg_hba.conf'
  >> locate pg_hba.conf
   ⇒ ##(maybe the same)

STOPPING AND STARTING THE SERVER
 ⇒ ....

  * check that the postgresql
   ⇒ database server is running
  >> ps ax | grep postgres
  >> ps ax | grep postmaster
   ⇒ ##(an older form)
  >> pg_ctl status
   ⇒ ##(PGDATA needs to be set)
  >> sudo /etc/init.d/postgresql
   ⇒ -8.4 status  ##(maybe best
   ⇒ on debian-style linux)

  * start the postgresql database
   ⇒ server
  >> sudo /etc/init.d/postgresql
   ⇒ -8.4 start ##(best on debian
   ⇒  style linux?)
  >> pg_ctl start
  >> pg_ctl -w start     ##(
   ⇒ refuses connections until
   ⇒ the server is going)

  * the default 8.4 data folder on
   ⇒  ubuntu
  >> /var/lib/postgresql/8.4/main

  * start a postgresql server
   ⇒ listening on tcp port 5433
  >> pg_ctl -o "-p 5433" start

  * shutdown the postgre database
   ⇒ server
  >> pg_ctl stop
  >> sudo /etc/init.d/postgresql
   ⇒ -8.4 stop     ##(debian-
   ⇒ style)
```

```
  * restart the database server
  >> pg_ctl restart
  >> sudo /etc/init.d/postgresql
   ⇒ -8.4 restart  ##(debian-
   ⇒ style)

BATCH SQL STATEMENTS

 * Run gunzipped sql file in
  ⇒ PostGres, adding to the
  ⇒ library.
 >> gzip -dc /tmp/pavanlimo.gz |
  ⇒ psql -U user db

VIEWING DATA

 * view all data from the table '
  ⇒ trees'
 >> SELECT * FROM trees;

 * display table data with extra
  ⇒ ascii borders
 >> \pset border 2
 >> SELECT * FROM trees;

 * display table data with no
  ⇒ ascii borders
 >> \pset border 0

TABLES

 * view the definition (columns)
  ⇒ of the table 'offers'
 >> \d offers

CREATING TABLES ....

  * create a new table 'newtable'
   ⇒ with 2 fields (columns)
  \begin{lstlisting}

   testdb=> CREATE TABLE newtable
    ⇒ (
       testdb(>  first integer not
        ⇒  null default 0,
       testdb(>  second text)
       testdb-> ;
```

* postgresql SQL to show count of ALL tables (relations) including ¿¿ SELECT relname, reltuples, $pg_relation_size(relname) FROM pg_class r JOIN pg_namespace cON(relnamespace = n.oid) WHERE relkind = 'r' AND n.nspname = 'public' ORDER BY relname;$

* for x in 'psql -el— awk 'print 1'|$egrep - v$ >> $for x in `psql - el|awk 'print 1'` — egrep -v "(^{List}|^{Name}|-|)"; do pg_dump -C$ x — gzip ¿ /var/lib/pgsql/backups/$x -$ *nightly.dmp.gz*; *done*

DATABASES ....

* list databases on the database server ¿¿ psql -l

* List all PostgreSQL databases. Useful when doing backups ¿¿ psql -U postgres -lAt — gawk -F||'1 ! /^{template}/1 ! /^{postgres}/ NF > 1 print 1'

* create a postgresql database called 'company' ¿¿ createdb company (this is typed at the bash shell)

* delete the database called 'company' ¿¿ dropdb company

THE PSQL UTILITY ....

* connect to the 'test' database on the database server ¿¿ psql test (the name of a database is required, or defaults to the user) ¿¿ psql (may produce an error, if there is no database with username)

* list all databases on the database server ¿¿ psql -l

* connect to the 'test' database on the server using the role (user/group) 'bob' ¿¿ psql -U bob -d test

== psql commands .. - quit the psql interpreter .. - show database roles .. ł- list all databases ..

USER ACCOUNTS ....

* create a new postgre sql 'role' called 'bob' ¿¿ createuser bob (a role is like a user and group combined) pg:¿¿ CREATE ROLE bob; (within the psql interpreter)

* eliminate the postgresql role 'bob' ¿¿ dropuser bob pg:¿¿ DROP ROLE bob; pg:¿¿ drop role bob; (the same, its not case sensitive)

* show current roles within the database server pg:¿¿ pg:¿¿ SELECT rolname FROM pg_roles; (*onlydisplaysthenames*)

* create a new database role 'bob' which can create databases, users and login ¿¿ CREATE ROLE bob LOGIN CREATEDB CREATEROLE;

BACKUPS

* make pgsql backup and gzip it ¿¿ $pg_dumpotrs2|gzip > dump.gz$

* dump database from postgresql to a file ¿¿ $pg_dump - Ft - b - Uusername - hdb.host.comdb_name > db.tar$

* Export/Backup a PostgreSQL database ¿¿ $pg_dump - Upostgres[nomeDB] > db.dump$