National University of Singapore
School of Computing

CS2102: Database Systems
Semester 2, 2017/18

Using PostgreSQL's psql

1 Introduction

PostgreSQL is an open-source object-relational database management system that started out as a research project called POSTGRES at the University of California at Berkeley. This guide introduces how to use `PostgreSQL`'s command-line client program called `psql`.

2 Accessing PostgreSQL Servers

To use `psql` requires a connection to a `PostgreSQL` server. This section covers how to connect to the `PostgreSQL` servers running on SoC's Unix servers. Alternatively, you could also install `PostgreSQL` server to run on your own computer[1].

1. You will need a SoC account to login to the Unix server sunfire.comp.nus.edu.sg. If you're a non-SoC or an exchange student without a SoC account, proceed to `https://mysoc.nus.edu.sg/~newacct` to create your SoC account. If you have just created your SoC account, you might have to wait a while before your `PostgreSQL` account becomes active and available for use.

2. With your SoC account, login to `sunfire.comp.nus.edu.sg`.

   ```
   ssh SOC_ACCOUNT_ID@sunfire.comp.nus.edu.sg
   ```

3. Download the file psql.zip as follows.

   ```
   $ cd ~
   $ wget http://www.comp.nus.edu.sg/~cs2102/psql.zip
   $ unzip psql.zip
   ```

   The unzipped directory psql/ contains 6 files: `check.sh`, `path.sh`, `psql0.sh`, `psql1.sh`, `resale-flat-prices.csv`, and `test.sql`.

4. In order to run `psql`, you need to add the directory `/usr/local/postgres/10-pgdg/bin/64` to the *PATH* environment variable. To do this, execute the following commands.

   ```
   $ cd psql
   $ bash path.sh
   $ source ~/.bash_profile
   $ echo $PATH
   ```

[1] https://www.postgresql.org/download/

You should see `/usr/local/postgres/10-pgdg/bin/64` in the output produced by the echo command. Note that you do not need to repeat this step for subsequent logins to sunfire.

5. Each student has been assigned a `PostgreSQL` account on one of two `PostgreSQL` servers (named `psql0` and `psql1`). If your NUSNET User ID ends with an even digit, your `PostgreSQL` account is on server psql0; otherwise, your `PostgreSQL` account is on server psql1.

6. You can now use the `psql` command from sunfire to connect to a target database on your assigned `PostgreSQL` server with a `PostgreSQL` user account and password. The target database that you will be using is named cs2102. Your `PostgreSQL` user account is your SoC account ID[2], and your default `PostgreSQL` account password is your NUS student number (with letters in uppercase). Assuming that your assigned `PostgreSQL` server is psql0, you can start `psql` as follows.

```
$ psql -U POSTGRESQL_ACCOUNT_NAME -d cs2102 -h psql0
Password for user POSTGRESQL_ACCOUNT_NAME: POSTGRESQL_ACCOUNT_PASSWORD
psql (10.1)
Type "help" for help.

cs2102=>
```

To exit `psql`, enter \q at `psql`'s "cs2102=>" prompt.

3 Using `psql`

`psql` provides an interactive terminal interface to edit/execute SQL commands/queries and view query results.

At psql's prompt "cs2102=>", you can type in SQL commands. Input lines that terminate with a semicolon will be sent to the database server for execution.

```
cs2102=> create table students (
cs2102(> sid integer,
cs2102(> name varchar(80)
cs2102(> );
CREATE TABLE
cs2102=>
```

To change your `PostgreSQL` account password to "NEW_PASSWORD", use the `alter` command.

```
cs2102=> alter user POSTGRESQL_ACCOUNT_NAME password 'NEW_PASSWORD';
ALTER ROLE
cs2102=>
```

Besides SQL commands, you can also type in meta-commands that will be processed by `psql`. Each meta-command begins with an unquoted backslash. Any SQL command that has been typed but has

[2] As PostgreSQL does not allow hypen characters in account names, omit any hyphen character in your account name. E.g., if your SoC account ID is "nosmo-king", then your `PostgreSQL` account name is "nosmoking".

not yet been sent for execution is stored in a memory buffer called *query buffer*. The contents of this buffer can be edited by invoking a configurable text editor within `psql`. The default editor is `vi` and you'll find out how this could be reconfigured in the next section.

The following table shows some of the basic meta-commands.

Meta-command	**Meaning**
\q	Quit psql.
\h	Display all SQL commands with available syntax help.
\h *COMMAND*	Display syntax of *COMMAND*. E.g., \h *create table*
\d	List all created tables.
\d TABLE	List information on relation named TABLE.
\p	Display the contents of the query buffer; if the current query buffer is empty, display the most recently executed query.
\w FILE	Output the contents of the query buffer to the file named *FILE*; if the current query buffer is empty, ouput the most recently executed query to *FILE*.
\r	Clear the query buffer.
\e	Invoke the text editor to edit the contents of the query buffer; if the query buffer is empty, edit the mostly recently executed query.
\e FILE	Invoke the text editor to edit the contents of a file named *FILE*. The contents of the edited file will be copied to the query buffer at the end of the edit session.
\o FILE	Enable future query results to be saved to the file named *FILE*.
\g	Send the contents of the current query buffer to the server for execution; if the current query buffer is empty, the most recently sent query is re-executed.
\i FILE	Reads the contents from the file named *FILE* and sent their contents to the server for execution.
\!	Escapes from the `psql` session to a sub-shell. The `psql` session resumes when the sub-shell is exited.

The following example illustrates the execution of some meta-commands.

```
cs2102=> \d
```

List of relations

Schema	Name	Type	Owner
alice	students	table	alice

(1 row)

```
cs2102=> \d students
```

Table "alice.students"

Column	Type	Collation	Nullable	Default
sid	integer			
name	character varying(80)			

```
cs2102=> \q
$
```

4 Optional Configurations

This section presents some optional configurations to make it more convenient to use `psql`.

To avoid having to enter the -U, -d, and -h parameter values when starting `psql` and to configure the default editor to `nano` (which could be easier to use if you are not familiar with vi), run either `psql0.sh` (if your `PostgreSQL` server is psql0) or `psql1.sh` (if your `PostgreSQL` server is psql1). If you wish to configure psql's editor to be vim or emacs, edit psql0.sh/psql1.sh to uncomment the appropriate `export` command (by deleting the beginning "#" character).

The following example assumes that your assigned `PostgreSQL` server is psql0 and your account name is "alice".

```
$ bash psql0.sh
$ source ~/.bash_profile
$ bash check.sh
PGUSER=alice
PGHOST=psql0
PGDATABASE=cs2102
PGSQL_EDITOR=/usr/local/bin/nano
$
```

The configuration using psql0.sh/psql1.sh needs to be performed only once. You can now start `psql` with simply "psql".

Finally, to avoid having to enter your account password when starting `psql`, create a password file named .pgpass in your home directory as follows (assuming that your assigned `PostgreSQL` server is psql0, your account name is "alice" and your password is "wonderland").

```
$ echo "psql0:*:cs2102:alice:wonderland" > ~/.pgpass
$ chmod 600 ~/.pgpass
$
```

The `.pgpass` file will need to be edited whenever you change your password using the `alter` command.

5 Trying out `psql`

The following example illustrates how to execute a SQL script named `test.sql` which creates and loads data into a table, and shows the number of records in it.

```
$ psql
cs2102=> \i test.sql
CREATE TABLE
COPY 759895
  count
--------
 759895
(1 row)
cs2102=> \q
$
```

You can also execute the script from the command line using "`psql < test.sql`" or "`psql -f test.sql`".

6 Getting help

- To reset your SoC account password, visit https://mysoc.nus.edu.sg/~myacct/iforgot.cgi.
- For any administrative problems with your SoC account, contact techsvc@comp.nus.edu.sg.
- If you forgot your `PostgreSQL` account password or run into any PostgreSQL administrative problems (e.g., can't execute queries because of corrupted database system files) that you're unable to resolve, contact psql@comp.nus.edu.sg for help.

The following are some useful online documentation.

- **psql**
  - https://www.postgresql.org/docs/current/static/app-psql.html
  - https://www.postgresql.org/docs/curent/static/tutorial-accessdb.html
  - https://www.citusdata.com/blog/2017/07/16/customizing-my-postgres-shell-using-psqlrc
- **PostgreSQL**
  - http://www.postgresql.org/docs/current/static/index.html
  - http://www.comp.nus.edu.sg/~cs2102/postgresql/doc/html
- **Nano editor**
  - https://www.howtogeek.com/howto/42980/the-beginners-guide-to-nano-the-linux-command-line-text-editor
- **pgAdmin**
  If you would like use `pgAdmin`'s graphical interface to connect to the `PostgreSQL` servers, you can download a basic configuration guide at http://www.comp.nus.edu.sg/~cs2102/using-pgadmin.pdf. Note that if you opt to use `pgAdmin`, you will have to learn how to use it on your own.