Using PostgreSQL on the lab workstations

We have a command line interface to PostgreSQL server, so you need to run it from a Unix prompt in a shell window. To enable the various applications required, first type

```
need postgresql
```

You may wish to add “need postgresql” command to your .cshrc file so that it is run automatically. Add this command after the command need SYSfirst, which has to be the first need command in your .cshrc file.

There are several commands you can type at the unix prompt:
createdb 〈db name〉

Create your database, named userid_nodejs by typing
createdb userid_nodejs

In order to access a database by using a password for authentication, you can set your password by connecting to some standard database (eg "postgres") and doing an alter user:

% psql postgres
psql (9.1.11)
Type "help" for help.

postgres=> alter user <username> with password 'XyZZy';

Note that this is the only database that is set up to allow access from an external program. Please use your ECS username as your userid.

To set up the password for your new database <userid>_nodejs, open an existing database (say with the name <userid>) by typing:
> psql userid_nodejs

and then proceed with

userid_nodejs

=>ALTER USER ECS username WITH PASSWORD 'newpassword';

This will set your password for the database
<userid>_nodejs

Note that this password should NOT be the same as your system password. It should protect your database from possible misuse, but need not to be as secure as your system password. Type

userid

=>\q
to close your old database.

Creates an empty database.

The database is stored in the same PostgreSQL cluster used by all the students in the class. You may freely name your database. But to ensure security, you must issue the following command as soon as you log-in into your database for the first time:

```
REVOKE CONNECT ON DATABASE <database_name> FROM PUBLIC;
```

You only need to do this once (unless you get rid of your database to start again).

Note, your markers may check whether you have issued this command and if they find you didn't, you may be penalized.

Something to know:

1. psql [–d 〈db name〉]

Starts an interactive SQL session with PostgreSQL to create, update, and query tables in the database. The db name is optional (unless you have multiple databases).

2. dropdb 〈db name〉

Gets rid of a database. (In order to start again, you will need to create a database again)

3. pg_dump -i 〈db name〉 **>** 〈file name〉

Dumps your database into a file in a form consisting of a set of SQL commands that would reconstruct the database if you loaded that file.

```
4. psql –d <database_name> -f <file_name>
```

Copies the file <file_name> into your database <database_name>

Other important functionalities:

Inside and interactive SQL session, you can type SQL commands. You can type the command on multiple lines (note how the prompt changes on a continuation line). End commands with a ';'.

There are also many single line PostgreSQL commands starting with '\'. No ';' is required. The most useful are

`\?` to list the commands,

\i 〈file name〉
loads the commands from a file (eg, a file of your table definitions or the file of data provided by others).

`\dt` to list your tables.

\d 〈table name〉 to describe a table.

`\q` to quit the interpreter

`\copy <table_name> to <file_name>`
Copy your table_name data into the file file_name.

`\copy <table_name> from <file_name>`
Copy data from the file file_name into your table table_name.

Note also that the PostgreSQL interpreter has some line editing facilities, including up and down arrow to repeat previous commands.

For longer commands, it is safer (and faster) to type your commands in an editor, then paste them into the interpreter!