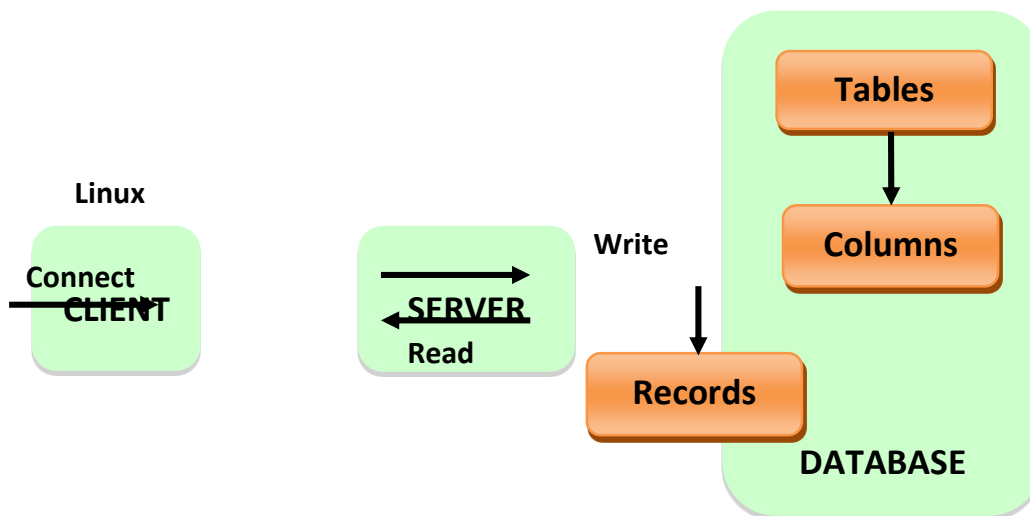


C++ to PostgreSQL Tutorial

Authors: Shuli Li and Jeff Terrell

Introduction

PostgreSQL is a powerful, open source object-relational database system. It has more than 15 years of active development and a proven architecture that has earned it a strong reputation for reliability, data integrity, and correctness. It runs on all major operating systems, including Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), and Windows. It is fully ACID compliant, has full support for foreign keys, joins, views, triggers, and stored procedures (in multiple languages). It includes most SQL:2008 data types, including INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, and TIMESTAMP. It also supports storage of binary large objects, including pictures, sounds, or video. It has native programming interfaces for C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC.



Client

PostgreSQL is available integrated with the package management on most Linux platforms. It provides proper integration with the operating system, including automatic patching and other management functionality, which makes it the recommended way to install PostgreSQL.

Used Linux based on future project requirements (NUC)

Library

libpq is Chosen to be used, it's already included in postgresql installation. Convenient.

libpq is the C application programmer's interface to PostgreSQL. libpq is a set of library functions that allow client programs to pass queries to the PostgreSQL backend server and to receive the results of these queries.

To build (i.e., compile and link) a program using libpq you need to do all of the following things: include the libpq-fe.h header file:

```
#include <libpq-fe.h>
```

How C++ interfaces to PostgreSQL:

Connect:

- Connection to database:

Execute:

- (write) create a table
- (write) create columns in the table
- (write) create records into the columns
- (read) fetch records from columns
- (write) erase records from table

Disconnect:

- Drop table from connection

Prerequisites:

- PostgreSQL 9.1 & related packages
- pgAdmin III (used to verify operations performed in test programs)

Setup:

1. Enter the following in Terminal:

```
sudo apt-get install postgresql-9.1 postgresql-client postgresql-client-9.1 postgresql-client-common postgresql-common postgresql-contrib postgresql-contrib-9.1
```

2. Add connection to server in pgAdmin III:

File → Add Server...

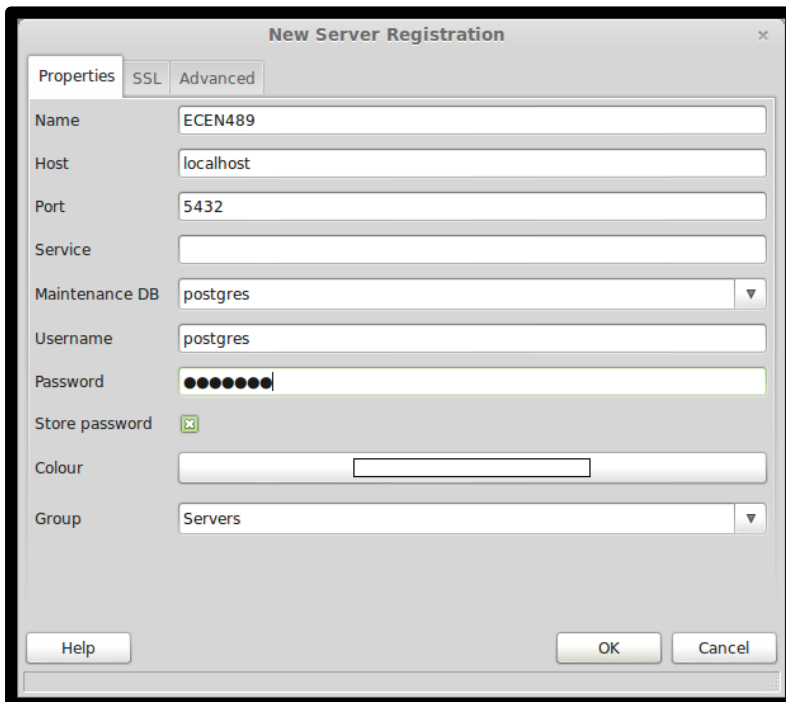
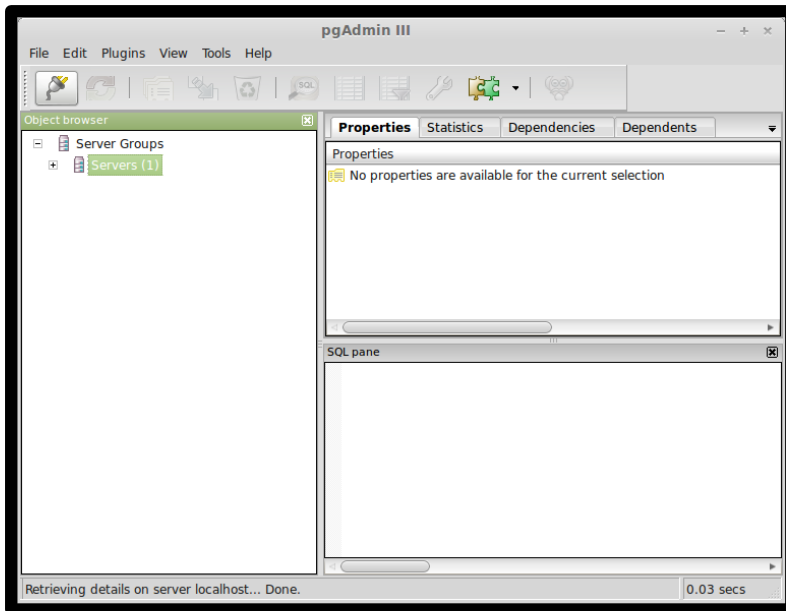
Name: ECEN489

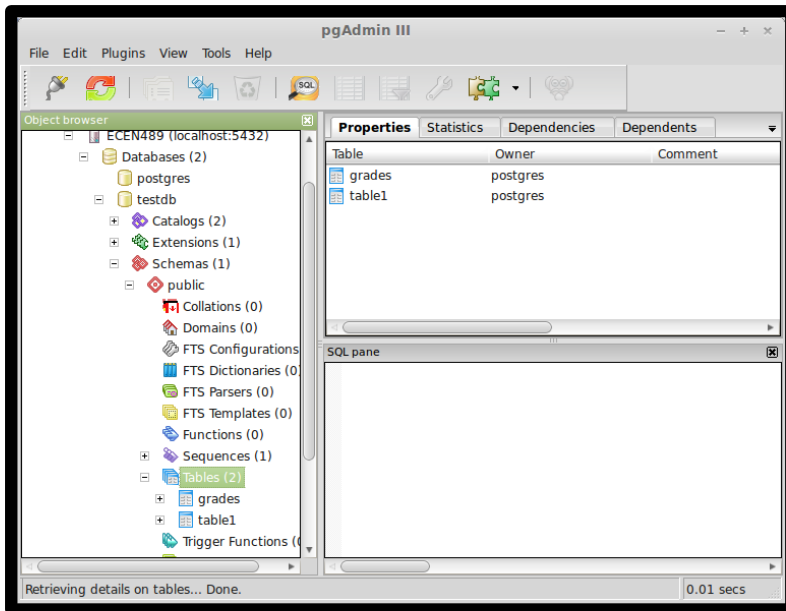
Host: localhost (running off of own IP)

Port: 5432

Username: postgres

Password: test123





Sample Program #1:

- Create new table and store some data

Source file: [writetest.cpp](#)

Sample output:

```

Terminal
File Edit View Search Terminal Help
jdterrell@AggieEE-NB ~ $ g++ writetest.cpp -o writetest -I/usr/local/include/ -l
pqxx -lpq
jdterrell@AggieEE-NB ~ $ ./writetest
Connection to database - OK
Create grade table - OK
Insert grade record - OK
Insert grade record - OK

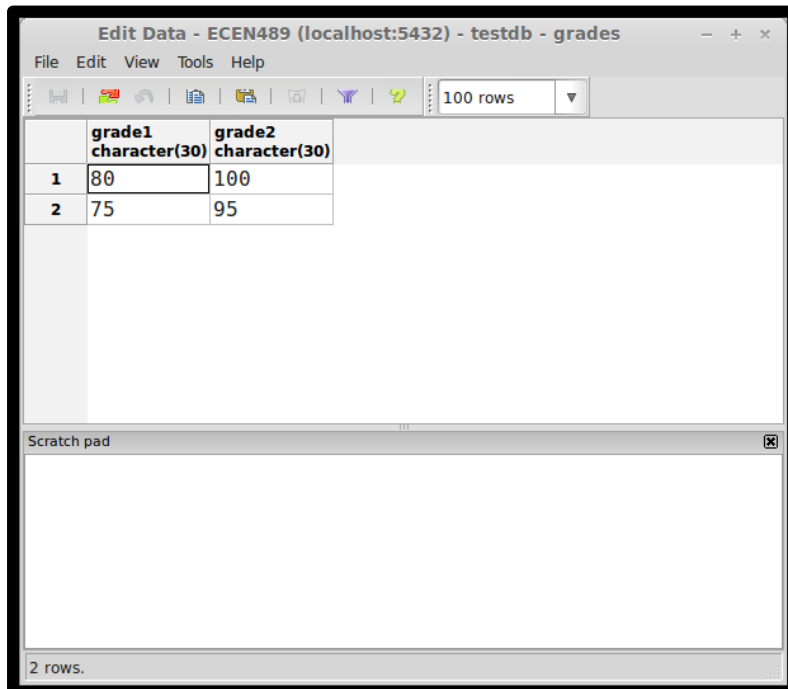
Fetch grade record:
*****
grade1      grade2
*****
80           100
75           95

Press ENTER to remove all records & table....

Delete grades record - OK
Drop grade table - OK
jdterrell@AggieEE-NB ~ $

```

Before the table is deleted...I used pgAdmin to verify the data:



The screenshot shows the pgAdmin interface for editing data in the 'grades' table. The table has two columns: 'grade1' (character(30)) and 'grade2' (character(30)). There are two rows of data: Row 1 with values '80' and '100', and Row 2 with values '75' and '95'. The interface includes a menu bar (File, Edit, View, Tools, Help), a toolbar, and a status bar at the bottom indicating '2 rows'.

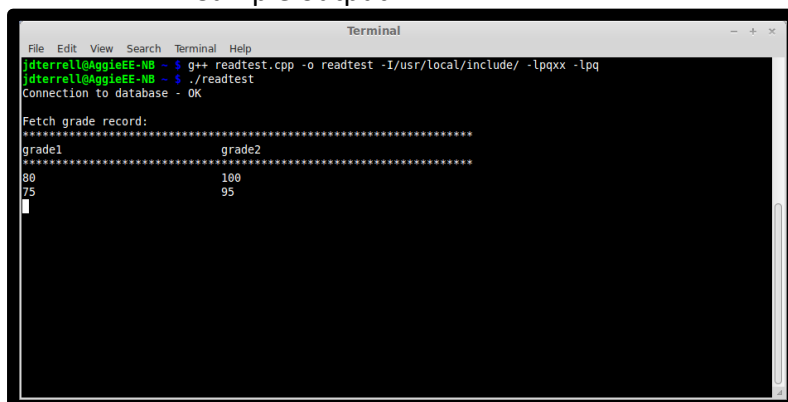
| | grade1 character(30) | grade2 character(30) |
|---|-------------------------|-------------------------|
| 1 | 80 | 100 |
| 2 | 75 | 95 |

Sample Program #2:

- Read from existing table

Source file: [readtest.cpp](#)

Sample output:



The screenshot shows a terminal window with the following output:

```

jdtterrell@AggieEE-NB ~$ g++ readtest.cpp -o readtest -I/usr/local/include/ -lpqxx -lpq
jdtterrell@AggieEE-NB ~$ ./readtest
Connection to database - OK

Fetch grade record:
*****
grade1          grade2
*****
80              100
75              95

```

FAQ:

Where can I find more information on PostgreSQL and libpq?

PostgreSQL: <http://www.postgresql.org/docs/9.1/interactive/index.html>

libpq: <http://www.postgresql.org/docs/9.1/interactive/libpq.html>

What if I want to create a database using the library?

Creating a database is a simple CREATE DATABASE SQL statement, same as any other libpq operation. You must connect to a temporary database (usually template1) to issue the CREATE DATABASE, then disconnect and make a new connection to the database you just created.

Sources/Additional Resources:

<http://www.linux.com/community/blogs/133-general-linux/562240-postgresql-c-tutorial>

<http://www.askyb.com/cpp/c-postgresql-example/>

<http://www.linuxjournal.com/content/accessing-postgresql-cc>