

PostgreSQL

Copyright © 2024 by
Robert M. Dondero, Ph.D.
Princeton University

Objectives

- The lecture will:
 - Provide *a taste of* the **PostgreSQL** DBMS
 - Give you enough information about **PostgreSQL** to:
 - Help you decide if you want to use it in your project
 - Help you get started with it

PostgreSQL

- **Who:** Michael Stonebreaker, UC Berkeley
- **When:** 1996
- **Why:**
 - Successor to *Ingres*
 - Add the fewest features needed to completely support data types



Motivation

- Why study PostgreSQL?
 - Popular
 - Preferred by Render & Heroku
 - Different
 - Contrasts with SQLite
 - More typical DBMS
 - Runs as a **process** that is distinct from the application process
 - Reasonable for COS 333 project
 - Used by many project teams

Options

- Options for using PostgreSQL
 - (1) On a cloud service
 - ElephantSQL, Render, or Heroku
 - Shown in lectures later in the course
 - (2) Locally
 - Described here

Installing PostgreSQL

- Linux (Debian, Ubuntu)
 - At a bash shell prompt:
 - `sudo apt-get install postgresql`

Installing PostgreSQL

- Mac
 - Install Homebrew
 - Browse to <https://brew.sh/> and follow instructions
 - At a bash shell prompt:
 - `brew install postgresql@16`

Installing PostgreSQL

- Mac (cont.)
 - Optionally, to enforce passwords
 - Edit file:
 - `$(brew --prefix) /var/postgresql@16/pg_hba.conf`

#	TYPE	DATABASE	USER	ADDRESS	METHOD
	local	all	rdonero		trust
	local	all	all		password
	host	all	all	127.0.0.1/32	password
	host	all	all	:::1/128	password
	local	replication	all		password
	host	replication	all	127.0.0.1/32	password
	host	replication	all	:::1/128	password

Substitute your Mac username for rdonero

Installing PostgreSQL

- MS Windows
 - Use a browser to download the latest version of PostgreSQL from `postgresql.org/download`
 - In File Manager double click on `postgresql-someversion-windows-x64.exe`
 - Use default settings
 - For “password for the database supervisor (postgres)” enter `xxx`.

Installing PostgreSQL

- MS Windows (cont.)
 - Add `C:\Program Files\PostgreSQL\someversion\bin` to your `PATH` environment variable

Starting the PostgreSQL Server

- Linux
 - At a bash shell prompt
 - `service postgresql start`
- Mac
 - At a bash shell prompt
 - `pg_ctl -D $(brew --prefix) /var/postgresql@16 start`
- MS Windows
 - PostgreSQL server is started automatically

Starting the PostgreSQL Server

- By default, DBMS listens for connections at default port 5432

Creating a User and a Database

- Linux

- At a bash shell prompt

- `sudo -u postgres psql postgres`

- At `psql` prompt

- `CREATE ROLE someusername LOGIN
PASSWORD 'somepassword';`
 - `CREATE DATABASE bookstore WITH OWNER =
someusername;`
 - `\q`

Creating a User and a Database

- Mac
 - At a bash shell prompt
 - `psql postgres`
 - At `psql` prompt
 - `CREATE ROLE someusername LOGIN
PASSWORD 'somepassword';`
 - `CREATE DATABASE bookstore WITH OWNER =
someusername;`
 - `\q`

Creating a User and a Database

- MS Windows

- In Command Prompt window

- `psql -U postgres`
 - Enter `xxx` when prompted for password

- At `psql` prompt

- `CREATE ROLE someusername LOGIN
PASSWORD 'somepassword';`
 - `CREATE DATABASE bookstore WITH OWNER =
someusername;`
 - `\q`

The psql Program

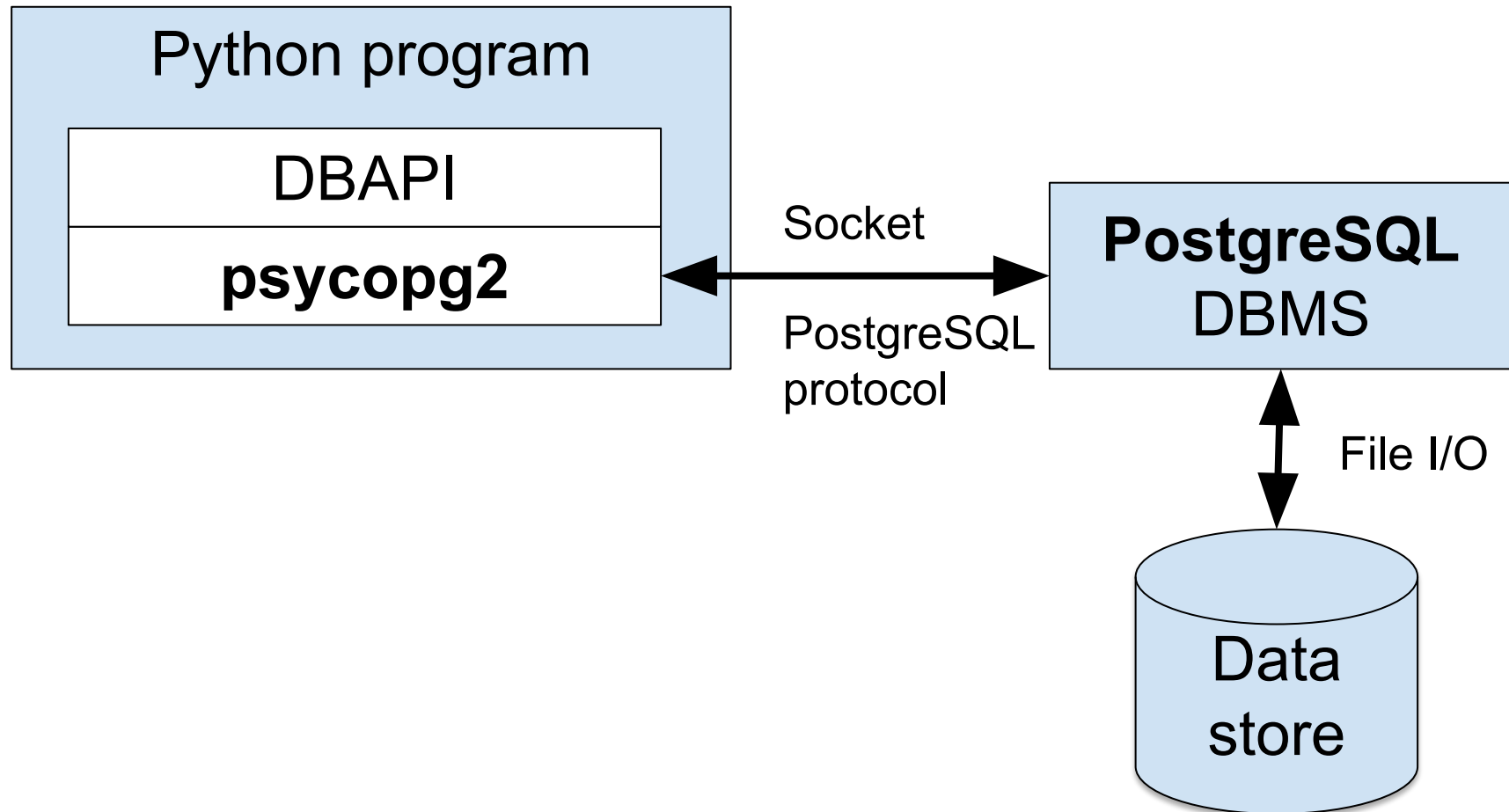
- At bash prompt or in Command Prompt window
 - psql
postgresql://*someusername:somepassword*@localhost/bookstore
 - Enter password if prompted
- At psql prompt
 - \i bookstore.sql
 - SELECT * FROM books;
 - ...
 - \q

The psql Program

Some `psql` commands

psql Command	sqlite3 Command
<code>\h</code>	<code>.help</code>
<code>\q</code>	<code>.quit</code>
<code>\list</code>	<code>(list databases)</code>
<code>\d</code>	<code>.tables</code>
<code>\d <i>table</i></code>	<code>.schema <i>table</i></code>
<code>\i <i>file</i></code>	<code>.read <i>file</i></code>

PostgreSQL via Python



Installing PostgreSQL Driver

- Linux, Mac, and MS Windows
 - At a bash shell prompt:
 - Activate your cos333 virtual env
 - `python -m pip install psycopg2`

PostgreSQL Pgmming in Python

- See **PostgreSQL/create.py**
 - No database file!
 - DBMS runs as a distinct process on a specified computer (localhost)
 - DBMS listens for connections at a known port (5432)

PostgreSQL Pgmming in Python

- See **PostgreSQL/display.py**
 - PostgreSQL uses standard SQL
- See **PostgreSQL/authorsearch.py**
 - PostgreSQL uses standard SQL
- See **PostgreSQL/order.py**
 - PostgreSQL uses standard SQL

PostgreSQL Pgmming in Python

- See **PostgreSQL/authorsearchprep.py**
 - PostgreSQL supports prepared statements
 - Note placeholder syntax
- See **PostgreSQL/orderprep.py**
 - (Much the same)

PostgreSQL Pgmming in Python

- See **PostgreSQL/purchase.py**
 - PostgreSQL supports transactions
- See **PostgreSQL/recovery.py**
 - Transactions work in PostgreSQL!

Stopping the PostgreSQL Server

- Linux
 - At a bash shell prompt
 - `service postgresql stop`
- Mac
 - At a bash shell prompt
 - `pg_ctl -D
"/opt/homebrew/var/postgresql@14" stop`
- MS Windows
 - PostgreSQL server is stopped when computer shuts down

PostgreSQL Assessment

- PostgreSQL assessment (vs. SQLite)
 - (con) Setup is much more complex
 - (con) Programmatic use is a little more complex
 - (pro) Production-quality
 - Distinct process
 - Authentication
 - Row-level (vs. database-level) locking
 - ...
 - (pro) Preferred by Render, Heroku

Summary

- The lecture has:
 - Provided *a taste of* the **PostgreSQL** DBMS
 - Given you enough information about **PostgreSQL** to:
 - Help you decide if you want to use it in your project
 - Help you get started with it