“Develop a passion for learning.”

2. The psql Client

Introduction to PostgreSQL

AGENDA

- Introduction to the psql client
- Configuring client access pg_hba.conf
- Command line parameters
- Meta-commands
- Input and Output

PSQL

- The *psql* client is a command-line interface (CLI) used to interact with the PostgreSQL database server.
- It is the primary tool for administration, querying, and managing PostgreSQL databases.
  - *Interactive Query:* query, manipulate, and manage database data by running SQL commands.
  - *Administration:* perform administrative tasks like managing users, and setting permissions.
  - *Script Execution:* can run SQL scripts to automate batch processing of database operations.

CONNECTION OPTIONS

- -h, --host
  - Specifies the host name or IP address of the PostgreSQL server to connect to. Defaults to a local connection if not specified.
  - *psql -h localhost or psql –host=192.168.1.100*
- -p, --port
  - Specifies the TCP port number the server is listening on. The default port is 5432.
  - *psql -p 5432 or psql –port=5433*
- -U, --username
  - Specifies the PostgreSQL username to connect as.
  - *psql -U postgres or psql --username=myuser*

CONNECTION OPTIONS

- -d, --dbname
  - Specifies the name of the database to connect to.
  - If omitted, psql connects to the database with the same name as the user.
  - *psql -d mydatabase or psql --dbname=mydatabase*
- -W, --password
  - Forces psql to prompt for a password before connecting. This ensures a password is always requested.

OTHER CONNECTION OPTIONS

- Often, we want to not open an interactive shell but just perform a specific operation.
- --l, --list
  - Lists all databases in the server
  - *psql -l*
- -c, --command
  - Executes the specified SQL command and then exits.
  - *psql -c "SELECT version();"*
- -f, --file
  - Executes commands from the specified file and then exits. This is useful for running SQL scripts.
  - *psql -f myscript.sql or psql --file=myscript.sql*

OTHER CONNECTION OPTIONS

- -F, --field-separator
  - Specifies the field separator for unaligned output format, overriding the default tab character.
  - *psql -F ',' to use a comma as the separator*.
- -A, --no-align
  - Switches to unaligned output mode, which prints columns without any padding, making it suitable for script processing.
- -t, --tuples-only
  - Outputs only the data rows, without headers or other formatting information. Useful for scripting or exporting data.
  - *psql -t -c "SELECT * FROM mytable;"*

OTHER CONNECTION OPTIONS

- -q, --quiet
  - Runs psql in quiet mode, suppressing some output such as welcome messages and command status.
- --echo-queries
  - Echoes SQL queries to standard output before executing them. Used for debugging.
- --no-password
  - Prevents psql from prompting for a password. It fails if a password is required

OTHER CONNECTION OPTIONS

- --single-transaction
- Executes all commands in a single transaction. If an error occurs, all changes are rolled back.
- *psql --single-transaction -f myscript.sql*
- -L, --log-file
- Logs all input to the specified file, useful for debugging or keeping a record of commands executed during a session.
- *psql -L mylogfile.log*

LAB 2-1

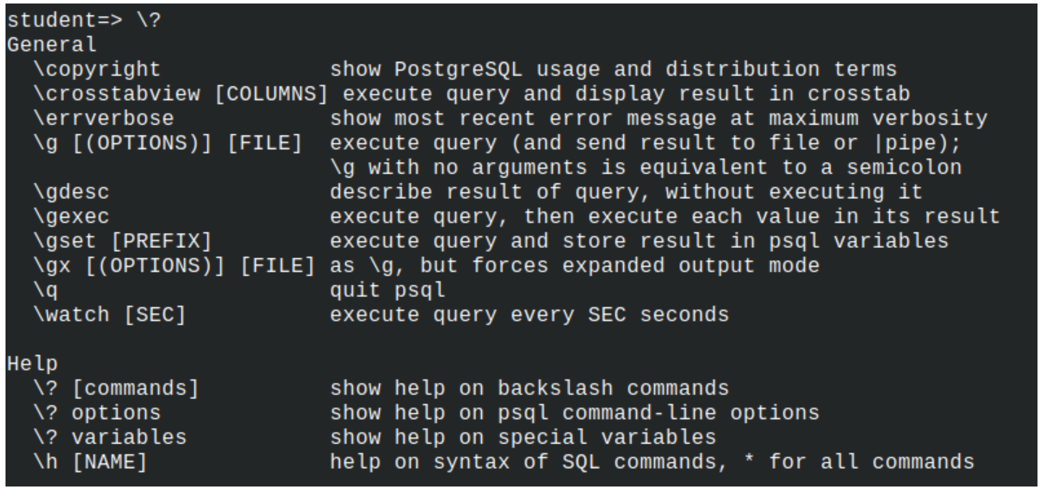
- The lab description and documentation is in the Lab directory in the class repository

PSQL META-COMMANDS

- Meta-commands begin with a backslash (\) and provide a ways to interact with PostgreSQL other than by SQL commands.
  - These commands are processed by `psql` itself, not the database server.
  - Many commands do not require any interaction with the serve, like *\q* and \c *database*
  - Those require information from the server, like *\dt*, are parsed into the appropriate SQL command and sent to the server.
  - The generated *\dt* query fetches information about the tables, from the *pg_class* and *pg_namespace* system catalogs.
  - psql formats the results of the query into human-readable tabular format which is displayed in psql console output

PSQL META-COMMANDS

- The \? command lists all the meta-commands available
  - Sample commands are listed for each group
- The main command groups we will be looking at are:
  - Database Listing Commands
  - Database Object Description Commands
  - Session Management Commands
  - Query Execution Control Commands
  - File and Script Execution Commands
  - Environment Variable Management Commands
  - Administrative Commands



```
student=> \?
General
  \copyright             show PostgreSQL usage and distribution terms
  \crosstabview [COLUMNS] execute query and display result in crosstab
  \errverbose            show most recent error message at maximum verbosity
  \g [(OPTIONS)] [FILE]  execute query (and send result to file or |pipe);
                         \g with no arguments is equivalent to a semicolon
  \gdesc                 describe result of query, without executing it
  \gexec                 execute query, then execute each value in its result
  \gset [PREFIX]         execute query and store result in psql variables
  \gx [(OPTIONS)] [FILE] as \g, but forces expanded output mode
  \q                     quit psql
  \watch [SEC]           execute query every SEC seconds

Help
  \? [commands]          show help on backslash commands
  \? options             show help on psql command-line options
  \? variables           show help on special variables
  \h [NAME]              help on syntax of SQL commands, * for all commands
```

DATABASE LISTING COMMANDS

- Displays information about the database and its objects, such as tables, indexes, schemas, and roles.
  - \l or \list: Lists all databases on the server.
  - \dt: Lists all tables in the current database.
  - \dv: Lists all views.
  - \di: Lists indexes.
  - \df: Lists functions.
  - \dn: Lists schemas.
  - \du: Lists roles/users

DATABASE OBJECT DESCRIPTION COMMANDS

- Provides detailed descriptions of specific database objects.
  - \d [object_name]: Describes a table, view, index, or sequence.
  - \d+ [object_name]: Provides a more detailed description, including additional metadata.
  - \df+ [function_name]: Shows detailed information about functions, including argument types and return type.

SESSION MANAGEMENT COMMANDS

- Manages database connections and the session environment, including connecting to different databases or switching users.
  - \c [database_name] or \connect [database_name]: Connects to a different database.
  - \conninfo: Displays information about the current connection.
  - \password [user]: Changes the password of a database user.

QUERY EXECUTION CONTROL COMMANDS

- Controls how SQL commands and queries are executed and displayed.
  - \timing: Toggles timing of commands, useful for performance analysis.
  - \watch [seconds]: Re-executes the last command every specified number of seconds.
  - \g [filename]: Sends query results to a file or program.

FILE AND SCRIPT EXECUTION COMMANDS

- Allows users to execute SQL commands from files or save query results to files.
  - \i [file_name]: Executes commands from a specified SQL file.
  - \o [file_name]: Redirects query output to a file.
  - \ir [file_name]: Executes SQL file, but reads it relative to the current working directory.
  - \copy: Used to copy data from a file to a table

FILE AND SCRIPT EXECUTION COMMANDS

- Allows users to execute SQL commands from files or save query results to files.
  - \i [file_name]: Executes commands from a specified SQL file.
  - \o [file_name]: Redirects query output to a file.
  - \ir [file_name]: Executes SQL file, but reads it relative to the current working directory.
  - \copy: Used to copy data from a file to a table

ENV VARIABLE MANAGEMENT COMMANDS

- Manages variables and the environment settings in the psql session.
  - \set [variable] [value]: Sets a psql variable.
  - \unset [variable]: Unsets a psql variable.
  - \echo :[variable]: Displays the psql variable contents
  - \prompt [text] [variable]: Prompts the user for input and stores it in a variable.

ADMINISTRATIVE COMMANDS

- Used for administrative tasks
  - \dconfig: Shows configuration parameters.
  - \dconfig+: Shows configuration parameters with comments and source.
  - \! [cmd]: Executes a shell command
  - \h [SQL command]: Provides help for SQL commands.

LAB 2-2

- The lab description and documentation is in the Lab directory in the class repository

SECURITY CONFIGURATION

- pg_hba.conf
  - PostgreSQL Host-Based Authentication
  - A file is used by PostgreSQL to control client authentication.
  - It specifies which users can connect to which databases from which hosts, and what authentication methods they use.
  - The location of the file being used by the running instance can displayed using:
  - *SHOW hba_file;*

SECURITY CONFIGURATION

- The file is a set of records, one per line.
- Each record specifies:
  - A connection type
  - A client IP address range if needed
  - A database name
  - A username,
  - The authentication method to be used for connections matching these parameters.

SECURITY CONFIGURATION

- The first record that matches all the connection parameters is used to perform authentication.
  - If authentication fails for one record, other records are not considered.
  - If no record matches, access is denied.

CONNECTION TYPES

- local
  - Connections using UNIX domain sockets.
  - Used for connections from the same machine where PostgreSQL is running.
  - Commonly used with authentication methods like peer that map PostgreSQL users with system users
- host
  - Connections attempts made using TCP/IP for either SSL or non-SSL connections
- hostssl
  - Connections that are secured using SSL
  - Ensures encrypted communication between the client and server.

CONNECTION TYPES

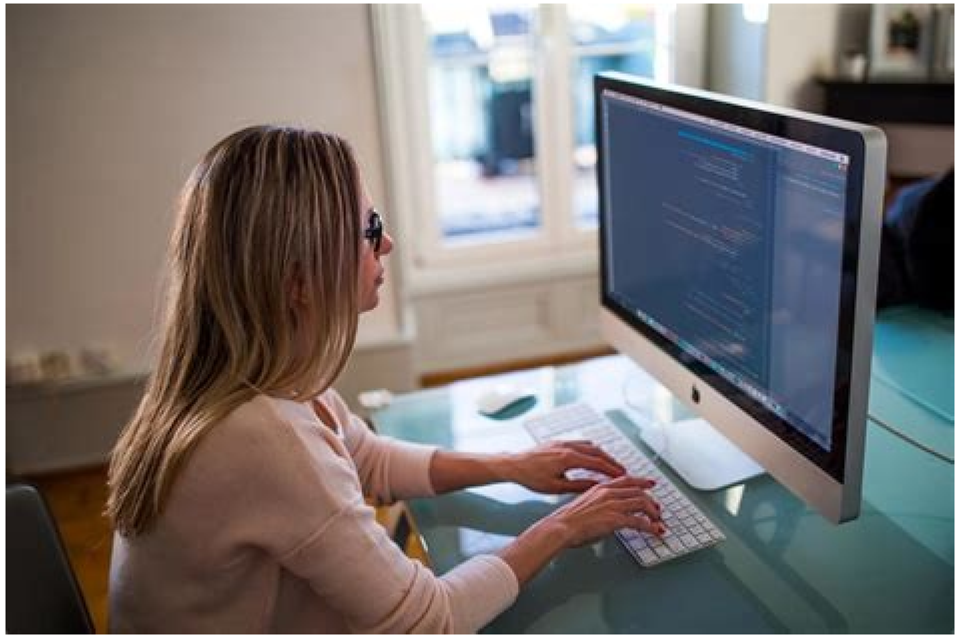
- hostnossl
  - Matches connection attempts made over TCP/IP that do not use SSL.
- hostgssenc
  - Connections made using TCP/IP with GSSAPI encryption
  - Used when Kerberos authentication is configured for secure connections
- hostnogssenc
  - Matches connection attempts made over TCP/IP that do not use GSSAPI.

DATABASE SPECIFICATION

- Specifies which database name(s) a record matches.
  - *all* matches all databases
  - *sameuser* matches if the database has the same name as the user
  - *replication* specifies that the record matches if a physical replication connection is requested
  - If the database name starts with a slash (/), the remainder of the name is treated as a regular expression
- Multiple database names and/or regular expressions can be supplied by separating them with commas

USER SPECIFICATION

- Specifies which database user name(s) this record matches with
  - *all* for all users
  - Specific PostgreSQL usernames
  - A regular expression starting with a /
  - OS level groups specified by +groupname
  - A comma delimited list of names and regular expressions

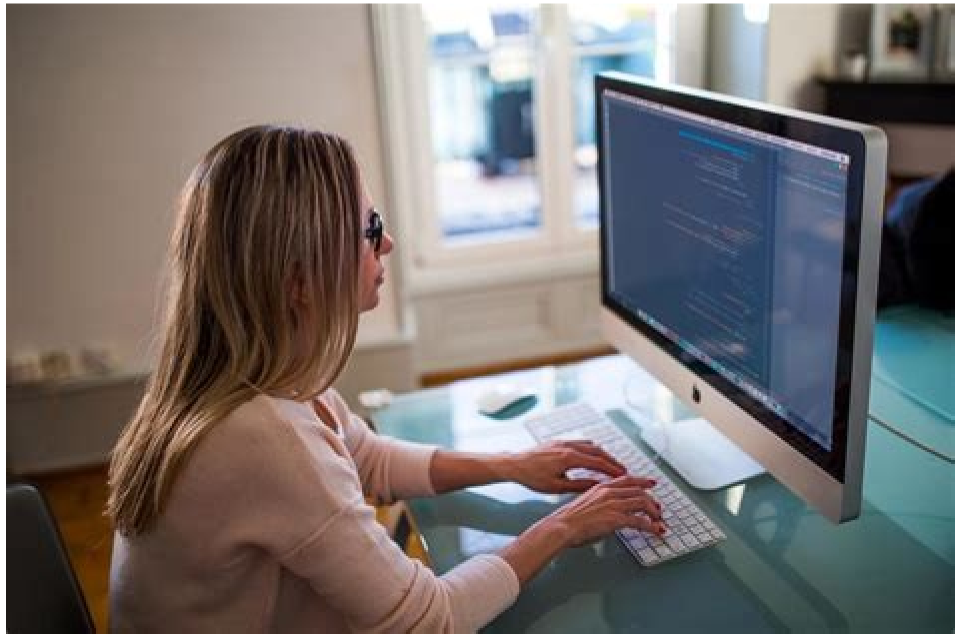


CONNECTION ADDRESSES

- Not required for non-network connections
  - Like local Unix sockets
- IP addresses that connections are allowed from
  - Can also be written as an IP address and netmask

AUTHENTICATION TYPES

- Some common authentication methods
  - *trust*: allows anyone that can connect to the server to login as any user they wish, without the need for authentication..
  - *reject*: reject the connection unconditionally - useful for “filtering out” certain hosts.
  - *scram-sha-256*: SCRAM-SHA-256 authentication to verify the password.
  - *md5*: SCRAM-SHA-256 or MD5 authentication to verify the password.
  - *peer*: uses client's operating system user name from the operating system, only available on local connection



SECURITY CONFIGURATION

- Postgresql.conf file
  - Specifies the connections that are allowed
- listen_addresses parameter
  - Specifies the TCP/IP address(es) on which the server is to listen for connections from client applications.
  - The special entry * corresponds to all available IP interfaces.
  - The entry 0.0.0.0 allows all IPv4 addresses and :: allows all IPv6 addresses.
  - If the list is empty, only Unix-domain sockets can be used to connect to it.
  - The default is *localhost*, which allows only local TCP/IP “loopback” connections to be made.

SECURITY CONFIGURATION

- unix_socket_directories parameter
  - Specifies the directory of the Unix-domain socket(s) on which the server is to listen for connections from client applications.
  - Multiple sockets can be specified
  - The default value is normally /tmp
  - On Windows, the default is empty

SECURITY CONFIGURATION

- unix_socket_permissions parameter
  - Sets the access permissions of the Unix-domain socket(s).
  - The default permissions are 0777, meaning anyone can connect.
  - This parameter is irrelevant on systems, that ignore socket permissions

WALK THROUGH

- Exploring the configuration files

End Module