HOW TO TERMINATE POSTGRESQL SESSIONS

I have encountered an interesting issue, as I could not perform specific database operations due to unwanted and active sessions using the database. Thus, I will briefly note the solution for further reference.

Prerequisites

This blog post is based on a *Debian Wheezy* and *PostgreSQL 9.1 version*.

```
$ lsb_release -d
Description:	Debian GNU/Linux 7.5 (wheezy)
postgres=# select * from version();
PostgreSQL 9.1.13 on x86_64-unknown-linux-gnu, compiled by gcc (Debian 4.7.2-5) 4.7.2, 64-bit
```

I have deliberately written down this information here, as there are some minor differences between PostgreSQL versions, so please be aware of potential differences.

The problem and the solution

Sometimes you need to terminate connections initiated by badly behaving client application, or just make sure nobody is querying database during a major update.

The solution is to use `pg_stat_activity` view to identify and filter active database sessions and then use `pg_terminate_backend` function to terminate them.

To prevent access during an update process or any other important activity you can simply `revoke connect` permission for selected database users or alter `pg_database` system table.

Who is permitted terminate connections

Every database role with `superuser` rights is permitted to terminate database connections.

How to display database sessions

`pg_stat_activity` system view provides detailed information about server processes.

```
SELECT datname as database,
      procpid as pid,
      usename as username,
      application_name as application,
      client_addr as client_address,
      current_query
  FROM pg_stat_activity
```

Sample output that will be used in the following examples.

```
 database | pid  | username | application | client_address |
current_query
```

```
----------+------+----------+------------+---------------+-----------------------------------------------
-----------------------------------------------------------------------------------------------------
 blog     | 8603 | blog     | blog_app    | 192.168.3.11   | select * from posts order by pub_date
 postgres | 8979 | postgres | psql        |                | select datname as database, procpid as pid,
usename as username, application_name as application, client_addr as client_address,
current_query from pg_stat_activity
 wiki     | 8706 | wiki     | wiki_app    | 192.168.3.8    |
(3 rows)
```

How to terminate all connections to the specified database

Use the following query to terminate all connections to the specified database.

```
SELECT pg_terminate_backend(procpid)
  FROM pg_stat_activity
 WHERE datname = 'wiki'
```

How to terminate all connections tied to specific user

Use the following query to terminate connections initiated by a specific user.

```
SELECT pg_terminate_backend(procpid)
  FROM pg_stat_activity
 WHERE usename = 'blog'
```

How to terminate all connections but not my own

To terminate every other database connection you can use process ID attached to the current session.

```
SELECT pg_terminate_backend(procpid)
  FROM pg_stat_activity
 WHERE procpid <> pg_backend_pid()
```

Alternatively, you can simply use `username` to filter out permitted connections.

```
SELECT pg_terminate_backend(procpid)
  FROM pg_stat_activity
 WHERE username <> current_username
```

Every example mentioned above can be extended to include more conditions like database name, client name, query, or even client address.

How to cancel running query

It is not always desired to abruptly terminate existing database connection, as you can just cancel running query using function shown in the following query.

```
SELECT pg_cancel_backend(procpid)
  FROM pg_stat_activity
 WHERE usename = 'postgres'
```

How to prevent users from connecting to the database

Database `connect` privilege

To prevent connections from specific database user `revoke` the `connect` privilege for selected database.

```
REVOKE CONNECT
    ON DATABASE wiki
  FROM wiki
```

To reverse this process use the `GRANT` statement.

```
GRANT CONNECT
   ON DATABASE wiki
   TO wiki
```

Use the `public` keyword to specify every database user.

```
REVOKE CONNECT
    ON DATABASE wiki
  FROM public
```

Database user `login` privilege

I did not mentioned it earlier but you can also use database user `login` privilege to disallow new connections.

```
ALTER ROLE wiki NOLOGIN;
```

To reverse this modification use the following query.

```
ALTER ROLE wiki LOGIN;
```

`pg_database` system table

Alternatively, you can alter `pg_database` system table to disallow new connections to specific database.

```
UPDATE pg_database
   SET datallowconn = FALSE
 WHERE datname = 'blog'
```

To reverse this process use the following query.

```
UPDATE pg_database
   SET datallowconn = TRUE
 WHERE datname = 'blog'
```

How to use the above-mentioned queries inside shell script

Use the `postgres` user to terminate connections..

```
#!/bin/sh
su postgres  -l -c "psql  -c 'select pg_terminate_backend(procpid) \
```

```
                from pg_stat_activity \
                where datname = '\"'wiki'\"'"
```

Use role with `superuser` rights to terminate connections.

```
#!/bin/sh
PGHOST=localhost PGUSER=admin PGPASSWORD=adminpass psql postgres -c "select
pg_terminate_backend(procpid) \
                                                from pg_stat_activity \
                                                where datname = 'wiki'"
```

Source: https://blog.sleeplessbeastie.eu/2014/07/23/how-to-terminate-postgresql-sessions/