**Useful Pg\_stat\_activity Queries for Monitoring Connections:**

1)Number of connected user:

SELECT usename AS username

FROM pg\_stat\_activity

where usename!=''

GROUP BY usename;

2) Which user and how many concurrent connections:

SELECT usename AS username, count(\*) AS concurrent\_statements

FROM pg\_stat\_activity

WHERE state = 'active'

GROUP BY usename;

3) If you need to figure out where the connections are going, you can break down

the connections by database.

SELECT datname, numbackends FROM pg\_stat\_database;

4) investigate connections to a specific database, query pg\_stat\_activity.

SELECT \* FROM pg\_stat\_activity WHERE datname='postgres';

5) All active connections but not the current query:

SELECT

age(clock\_timestamp(), query\_start),

usename,

datname,

query

FROM pg\_stat\_activity

WHERE

state != 'idle'

AND query NOT ILIKE '%pg\_stat\_activity%'

ORDER BY age desc;

6) All processes that are not idle but do have a wait event:

SELECT

usename,

datname,

query,

wait\_event\_type,

wait\_event

FROM pg\_stat\_activity

WHERE

state != 'idle'

AND wait\_event != '';

7) Query backend\_type equal to client\_backend.

SELECT \* FROM pg\_stat\_activity WHERE backend\_type = 'client backend';

8) Query to find start, state,state\_change,pid and duration

SELECT pid, now() - query\_start AS duration, query\_start, state\_change, state, query FROM pg\_stat\_activity WHERE backend\_type = 'client backend';

9) Session which are running for more 10 seconds and are not idle

select

now()-query\_start as runtime,

pid as process\_id,

datname as db\_name,client\_addr,client\_hostname,

query

from pg\_stat\_activity

where state!='idle'

and now() - query\_start > '10 seconds':: interval

order by 1 desc;

10) Kill connections based on time frame:

SELECT pg\_terminate\_backend(pid)

FROM pg\_stat\_activity

WHERE datname = 'Database\_Name'

AND pid <> pg\_backend\_pid()

AND state in ('idle', 'idle in transaction', 'idle in transaction (aborted)', 'disabled')

AND state\_change < current\_timestamp - INTERVAL '15' MINUTE;

11) Postgres kill all idle in transaction

SELECT pg\_terminate\_backend(pid)

FROM pg\_stat\_activity

WHERE datname='db'

AND state = 'idle in transaction';

12) To kill all active connections to a PostgreSQL database, execute the query below:

SELECT

pg\_terminate\_backend(pid)

FROM

pg\_stat\_activity

WHERE

datname ='postgres'

AND

leader\_pid

IS NULL;

13) query to kill all connections except for yours:

SELECT

pg\_terminate\_backend(pid)

FROM

pg\_stat\_activity

WHERE

datname =

'postgres'

AND

pid != pg\_backend\_pid()

AND

leader\_pid

IS NULL;

14) To terminate all connections to all databases in a Postgres server(except yours),

run the following query:

SELECT

pg\_terminate\_backend(pid)

FROM

pg\_stat\_activity

WHERE

pid != pg\_backend\_pid()

AND

datname

IS NOT NULL

AND

leader\_pid

IS NULL;

15) specific session:

select pid, query from pg\_stat\_activity where datname = current\_database();

select pg\_terminate\_backend(123);