```
---
--- # PGIO
#ls -l /tmp/pgio-0.9.tar
#tar -xvC ~ -f /tmp/pgio-0.9.tar
cd
git clone https://github.com/therealkevinc/pgio
tar -zxvf pgio/pgio-2019.09.21-v_1.0.tar.gz
--- # run pgio
cd pgio
---# create a database PGIO with default tablespace PGIO
sudo mkdir -p /u01/pgio && sudo chown -R $(whoami) /u01/pgio && df -Th /u01/pgio &&
du -hs /u01/pgio
psql postgres
 create tablespace pgio location '/u01/pgio';
 create database pgio tablespace pgio;
\q
du -h /u01/pgio
---
grep -vE "^[[:blank:]]*#|^$" pgio.conf
sed -ie '/^DBNAME=/s/=.*/=pgio/' pgio.conf
sed -ie '/^CONNECT_STRING=/s/=.*/=pgio/' pgio.conf
sed -ie '/^SCALE=/s/=.*/=100M/' pgio.conf
sed -ie '/^NUM_SCHEMAS=/s/=.*/=2/' pgio.conf
sed -ie '/^RUN_TIME=/s/=.*/=60/' pgio.conf
---
grep -vE "^[[:blank:]]*#|^$" pgio.conf
---
sh ./setup.sh
---
du -h /u01/pgio
pg_ctl -l $PGDATA/logfile restart
---
sh ./runit.sh       | grep -E "^|>[0-9]*<"
--- tmux send-keys -t :.1 top C-M
---# c m m m f u u s u s u...
---# you can compare: CPU shapes, mem settings, spectre/meltdown patches. This was
with Huge Pages | blks_read | blks_hit
--- tmux send-keys -t :.1 C-C C-M
--- tmux select-pane -t :.1
pmap $(head -1 $PGDATA/postmaster.pid) | sort -hk2 | tail -4 | grep -E "^|-s-"
grep -E "(shared_buffers|huge_pages).*=" $PGDATA/postgresql.conf
echo "huge_pages=off">> $PGDATA/postgresql.conf
grep -E "(shared_buffers|huge_pages).*=" $PGDATA/postgresql.conf
pg_ctl -l $PGDATA/logfile restart
pmap $(head -1 $PGDATA/postmaster.pid) | sort -hk2 | tail -4 | grep -E "^|-s-"
--- tmux select-pane -t :.0
sh ./runit.sh       | grep -E "^|>[0-9]*<"
---# Do you have a NUMA system? try that with numa=off
---
grep -E "(shared_buffers|huge_pages).*=" $PGDATA/postgresql.conf
```

```
sed -ie '/^huge_pages/d' $PGDATA/postgresql.conf
grep huge_pages $PGDATA/postgresql.conf
pg_ctl -l $PGDATA/logfile restart
pmap $(head -1 $PGDATA/postmaster.pid) | sort -hk2 | tail -4 | grep -E "^|-s-"
sh ./runit.sh       | grep -E "^|>[0-9]*<"
---
---
--- # Hints https://osdn.net/projects/pghintplan
---#sudo yum remove -y pg_hint_plan11
wget -O /tmp/pg_hint_plan11.rpm
https://osdn.net/projects/pghintplan/downloads/70540/pg_hint_plan11-1.3.4-1.el7.x86
_64.rpm/
sudo yum install -y /tmp/pg_hint_plan11.rpm
psql demo
load 'pg_hint_plan';
drop table if exists demo1;
create table demo1 as
select generate_series n , 1 a , lpad('x',1000,'x') x from
generate_series(1,10000);
create unique index demo1_n on demo1(n);
---
\d+ demo1
explain (analyze,verbose,costs,buffers)
select sum(n) from demo1 ;
---
/*+ IndexOnlyScan(demo1) */
explain (analyze,verbose,costs,buffers)
select sum(n) from demo1 ;
---
vacuum demo1;
---
explain (analyze,verbose,costs,buffers)
select sum(n) from demo1 ;
---
/*+ SeqScan(demo1) */
explain (analyze,verbose,costs,buffers)
select sum(n) from demo1 ;
---
--- # equivalent of profiles
drop table if exists people_country;
drop table if exists people_language;
create table people_country as
select generate_series id , 'CH' ctry from generate_series(1001,2000)
union all
select generate_series id , 'UK' ctry from generate_series(2001,3000)
;
create table people_language as
select generate_series id , 'DE' lang from generate_series(1001,1800)
union all
select generate_series id , 'FR' lang from generate_series(1801,2000)
```

```
union all
select generate_series id , 'EN' lang from generate_series(2001,3000)
;
analyze people_country;
analyze people_language;
\d+ people_country\d people_language
explain (analyze,verbose,costs,buffers)
select count(*) from people_country
join people_language using(id)
where ctry='UK' and lang='EN'
;
---
/*+ Rows(people_country people_language *2) */
explain (analyze,verbose,costs,buffers)
select count(*) from people_country
join people_language using(id)
where ctry='UK' and lang='EN'
;
---
\q
--- # pgSentinel
sudo su -
export PATH=$PATH:/usr/pgsql-11/bin
yum install -y postgresql11-devel postgresql11-contrib
cd ~ && git clone https://github.com/pgsentinel/pgsentinel.git
sed -ie '/nabstime.h/d' ~/pgsentinel/src/pgsentinel.c
cd pgsentinel/src && make CLANG=true && make install CLANG=true
whoami | grep root && exit
grep -vE "^[[:blank:]]*#|^$" $PGDATA/postgresql.conf
grep pgsentinel $PGDATA/postgresql.conf || cat >> $PGDATA/postgresql.conf <<CAT
shared_preload_libraries = 'pg_stat_statements,pgsentinel'
track_activity_query_size = 2048
pg_stat_statements.track = all
CAT
grep -vE "^[[:blank:]]*#|^$" $PGDATA/postgresql.conf
pg_ctl -l $PGDATA/logfile restart
psql demo
 CREATE EXTENSION pgsentinel;
insert into demo1(n) select generate_series n from
generate_series(1000000,2000000);
\x
select * from pg_active_session_history where pid=pg_backend_pid()
;


--- tmux select-pane -t :.1
--- tmux send-keys -t :.1 C-C
cd ~/pgio
sh ./runit.sh
--- tmux select-pane -t :.0
```

```
select * from pg_active_session_history order by ash_time desc fetch first 1 rows
only
;
--- tmux send-keys Up C-M
--- tmux send-keys Up C-M
--- tmux send-keys Up C-M
--- tmux send-keys Up C-M
--- tmux send-keys Up C-M
---
create extension pg_stat_statements;
\d pg_stat_statements;
select ash_time,ash.top_level_query,ash.query,stm.query "pg_stat_statement" from
pg_active_session_history ash join pg_stat_statements stm using(queryid) where
datname='pgio' order by ash_time desc fetch first 3 rows only
;
--- tmux select-pane -t :.1
--- tmux send-keys -t :.1 C-C
---
pgbench --initialize --scale 10 demo
--- tmux select-pane -t :.0
select * from pg_active_session_history order by ash_time desc fetch first 1 rows
only;
\x
select count(*),application_name,cmdtype,state,wait_event_type,wait_event
from pg_active_session_history where datname='demo' and ash_time>=current_timestamp
- interval '5 minutes'
group by application_name,cmdtype,state,wait_event_type,wait_event
order by 1;
--- tmux select-pane -t :.1
pgbench --select-only --no-vacuum --time=120 --client=10 --jobs=10 demo
--- tmux select-pane -t :.0
--- tmux send-keys Up C-M
vacuum full verbose pgbench_accounts;
select
count(*),application_name,cmdtype,state,wait_event_type,wait_event,substring(query,
1,12) from pg_active_session_history where datname='demo' and
ash_time>=current_timestamp - interval '5 minutes' group by
application_name,cmdtype,state,wait_event_type,wait_event,substring(query,1,12)
order by 1;
--- tmux send-keys -t :.1 C-C
select ash_time,pid,cmdtype,application_name,wait_event,blockers,blockerpid from
pg_active_session_history where wait_event_type='Lock' order by 1 desc,2 fetch
first 10 rows only
;
select ash_time,pid,cmdtype,application_name,wait_event,blockers,blockerpid from
pg_active_session_history where (ash_time,pid) in (select ash_time,blockerpid from
pg_active_session_history where wait_event_type='Lock') order by 1 desc,2 fetch
first 10 rows only
;
```

```
select
 ash_time,
 string_agg(
  case wait_event_type
   when 'IO' then 'd' when 'LWLock' then 'l' when 'Client' then 'n'
   else substr(wait_event_type,1,1) end
  ,'')
from pg_active_session_history
group by ash_time
order by 1
;

--- CPU
--- d IO
--- lw lock
--- CLIEnT
--- IPC
--- Timeout
--- Extension
--- Activity (server idle)
--- BufferPin
--- Lock


select distinct wait_event_type from pg_active_session_history;
```