

PostgreSQL-Consulting

data egret



My "default" postgresql.conf file,
step by step



- 269 settings in version 10
- 314 so far in version 12
- Settings in *postgresql.conf* are to be change manually
- *postgresql.auto.conf* can be changed only through **ALTER SYSTEM**
- **pg_settings** view combines everything together

```

postgres=# \x
Expanded display is on.
postgres=# select * from pg_settings where name ~ 'checkpoint_timeout';
-[ RECORD 1 ]-----+-----
name           | checkpoint_timeout
setting        | 3600
unit           | s
category       | Write-Ahead Log / Checkpoints
short_desc     | Sets the maximum time between automatic WAL checkpoints.
extra_desc     |
context        | sighup
vartype        | integer
source         | configuration file
min_val        | 30
max_val        | 86400
enumvals       |
boot_val       | 300
reset_val      | 3600
sourcefile     | /etc/postgresql/10/main/postgresql.conf
sourceline     | 208
pending_restart | f

```

```
postgres=# select distinct(context) from pg_settings ;
           context
-----
postmaster
superuser-backend
user
internal
backend
sighup
superuser
(7 rows)
```

- Please do not change the order of the settings when you edit them manually

- Please do not change the order of the settings when you edit them manually
- *postgresql.conf* supports includes

- Please do not change the order of the settings when you edit them manually
- *postgresql.conf* supports includes
- Always check **pg_settings** if you doubt...



- Please do not change the order of the settings when you edit them manually
- *postgresql.conf* supports includes
- Always check **pg_settings** if you doubt...
- And off we go

- * or 127.0.0.1

- * or 127.0.0.1
- 127.0.0.1 is OK, when pgbouncer is used

- * or 127.0.0.1
- 127.0.0.1 is OK, when pgbouncer is used
- Your database **must** be firewall protected

- Client connection cause Postgres to spawn a "heavy" Unix-Process

- Client connection cause Postgres to spawn a "heavy" Unix-Process
- Thats why things like *max_connections = 1000* will never work

- Client connection cause Postgres to spawn a "heavy" Unix-Process
- Thats why things like *max_connections = 1000* will never work
- A much better idea: *max_connections = 100 or 200* and really small pool sizes in pgbouncer or another connection pooler

- When all of max_connections are utilized, DBA needs to connect to a database server in order to troubleshoot such situation



- When all of max_connections are utilized, DBA needs to connect to a database server in order to troubleshoot such situation
- Should be at least 5, better 10

- If network is unstable, 5 seconds can really help



- If network is unstable, 5 seconds can really help
- *$tcp_keepalives_interval = 1$*
- *$tcp_keepalives_count = 5$*

- Rule of Thumb: 25% of RAM



- Rule of Thumb: 25% of RAM
- But to use 16/32/64Gb of **shared_buffers** efficiently, fast discs are required

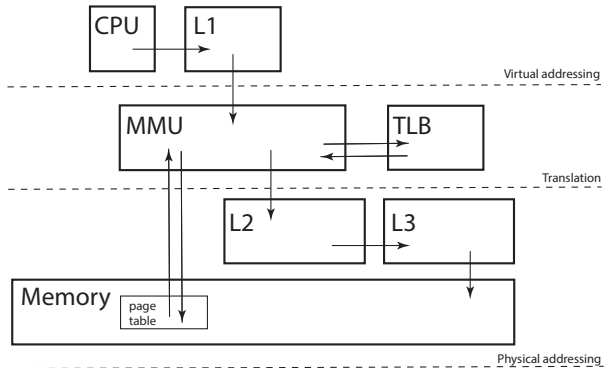
- Rule of Thumb: 25% of RAM
- But to use 16/32/64Gb of **shared_buffers** efficiently, fast discs are required
- If the database is definitely smaller than RAM, 75% of RAM for *shared_buffers* can also work

- Rule of thumb: when there are 8-16Gb *shared_buffers*, using of Huge Pages is recommended
- *huge_pages = on*

- Rule of thumb: when there are 8-16Gb *shared_buffers*, using of Huge Pages is recommended
- *huge_pages = on*
- Huge Pages should be first enabled in kernel

- Rule of thumb: when there are 8-16Gb *shared_buffers*, using of Huge Pages is recommended
- *huge_pages = on*
- Huge Pages should be first enabled in kernel
- *vm.nr_overcommit_hugepages* and *vm.nr_hugepages*

Linux Memory allocation



- RAM per process, Postgres workers use this RAM for sorting, hash joins etc.

- RAM per process, Postgres workers use this RAM for sorting, hash joins etc.
- 128Mb is a good starting point

- RAM per process, Postgres workers use this RAM for sorting, hash joins etc.
- 128Mb is a good starting point
- Too high setting could cause OOM

- RAM per process, Postgres workers use this RAM for sorting, hash joins etc.
- 128Mb is a good starting point
- Too high setting could cause OOM
- Could be individually configured for each session

- Same as *work_mem* but for superuser connections

- Same as *work_mem* but for superuser connections
- 256-512Mb, if there is enough RAM

- Same as *work_mem* but for superuser connections
- 256-512Mb, if there is enough RAM
- Could be quite helpful for **CREATE INDEX CONCURRENTLY**

- Same as *work_mem* but for superuser connections
- 256-512Mb, if there is enough RAM
- Could be quite helpful for **CREATE INDEX CONCURRENTLY**
- *autovacuum_work_mem* is a part of *maintenance_work_mem*, can be smaller

- *$\text{vacuum_cost_delay} = 0$*
- *$\text{vacuum_cost_page_hit} = 0$*
- *$\text{vacuum_cost_page_miss} = 10$*
- *$\text{vacuum_cost_page_dirty} = 10$*
- *$\text{vacuum_cost_limit} = 100$*

- *wal_level = replica*

- *wal_level = replica*
- *checkpoint_timeout = 60min*, if it is by given recovery target acceptable, could gain performance improvement
- *max_wal_size = 16GB*

- *wal_level = replica*
- *checkpoint_timeout = 60min*, if it is by given recovery target acceptable, could gain performance improvement
- *max_wal_size = 16GB*
- *checkpoint_completion_target = 0.9*

- Background Writer helps Checkpointer to send unused dirty pages to disk

- Background Writer helps Checkpointer to send unused dirty pages to disk
- Regret to say, it is not the best part of PostgreSQL codebase

- Background Writer helps Checkpointer to send unused dirty pages to disk
- Regret to say, it is not the best part of PostgreSQL codebase
- All settings to maximum:
 - ▶ *bgwriter_delay = 10ms*
 - ▶ *bgwriter_lru_maxpages = 1000*
 - ▶ *bgwriter_lru_multiplier = 10.0*

- 1 by default, which enables prefetch

- 1 by default, which enables prefetch
- Documentation recommends higher values for storages with high parallelism capabilities

- 1 by default, which enables prefetch
- Documentation recommends higher values for storages with high parallelism capabilities
- But there are evidences, that sometimes disabling *effective_io_concurrency* leads to better results
(<https://www.postgresql.org/message-id/flat/6c7a45df-f6ab-f2ce-6f84-9555864f6c86>)

Must have optimizer settings

- *effective_cache_size = 2 * shared_buffers* or less
- *default_statistics_target = 100*



- *autovacuum_vacuum_threshold* = 50
- *autovacuum_vacuum_scale_factor* = 0.05

- *autovacuum_vacuum_threshold = 50*
- *autovacuum_vacuum_scale_factor = 0.05*
- *autovacuum_naptime = 1s*

- *autovacuum_vacuum_threshold = 50*
- *autovacuum_vacuum_scale_factor = 0.05*
- *autovacuum_naptime = 1s*
- *autovacuum_max_workers = 10*

- *autovacuum_vacuum_threshold = 50*
- *autovacuum_vacuum_scale_factor = 0.05*
- *autovacuum_naptime = 1s*
- *autovacuum_max_workers = 10*
- *autovacuum_analyze_threshold = 50*
- *autovacuum_analyze_scale_factor = 0.05*

- *autovacuum_vacuum_threshold = 50*
- *autovacuum_vacuum_scale_factor = 0.05*
- *autovacuum_naptime = 1s*
- *autovacuum_max_workers = 10*
- *autovacuum_analyze_threshold = 50*
- *autovacuum_analyze_scale_factor = 0.05*

- *autovacuum_vacuum_threshold = 50*
- *autovacuum_vacuum_scale_factor = 0.05*
- *autovacuum_naptime = 1s*
- *autovacuum_max_workers = 10*
- *autovacuum_analyze_threshold = 50*
- *autovacuum_analyze_scale_factor = 0.05*

- *autovacuum_vacuum_threshold* = 50
- *autovacuum_vacuum_scale_factor* = 0.05
- *autovacuum_naptime* = 1s
- *autovacuum_max_workers* = 10
- *autovacuum_analyze_threshold* = 50
- *autovacuum_analyze_scale_factor* = 0.05
- *autovacuum_freeze_min_age* = 20000000 # 9.6 and older - default is most likely enough, older versions often require up to 1B
- *autovacuum_freeze_table_age* = 15000000

```
log_directory = ''/var/log/postgresql''  
log_filename = ''postgresql-%Y-%m-%d.log''  
log_rotation_age = 1d  
log_rotation_size = 0  
log_min_error_statement = error  
log_min_duration_statement = 1000  
log_checkpoints = on  
log_line_prefix = ''%m %p %u@%d from %h [vxid:%v txid:%x] [%i] ''  
log_lock_waits = on  
log_statement = ''none''  
log_replication_commands = on  
log_temp_files = 0  
log_timezone = ''Europe/Berlin''
```

Don't forget about one very useful extension

- *shared_preload_libraries = 'pg_stat_statements'*
- *pg_stat_statements.max = 10000*
- *pg_stat_statements.track = top*

Questions?

ik@dataegret.com

