POSTGRESQL CONFIGURATION FOR *HUMANS*

ONGRES

//ABOUT ONGRES

- **OnGres** (ON PostGRES) is an IT firm specialized on R&D on Databases, more specifically PostgreSQL, developing products that complement the PostgreSQL's ecosystem.
- **OnGres** is a global reference in **corporate level** services and products for **PostgreSQL** ecosystem.
- Active members of the PostgreSQL Community:
  - Founders of **PostgreSQL Spain**, one of the largest PUGs in the world, with about 900 members.
  - Trustees of Fundación PostgreSQL, a non-profit Foundation, and organizers of **PostgreSQL Ibiza** (pgibz.io).

//POSTGRESQL CONFIGURATION

- Mainly ***postgresql.conf*** (here's most of the meat).
- Authentication: ***pg_hba.conf*** (and ***pg_ident.conf***).
- Replicas: ***recovery.conf*** (may be merged soon).
- Some ***initdb*** parameters.
- Per-object settings (eg. ***fillfactor***).

Advanced stuff:

- Developer parameters.
- Compile-time #defines.

//WHY CONFIGURE POSTGRESQL?

- Otherwise, it only listens on localhost.
- You can only replicate by default in >= 10.
- To enable WAL archiving.
- Application developers say they get “connection refused!”
- Set defaults for client connections.
- Load extensions that required shared libraries.
- Enable checksums (*initdb*!)
- Compatibility with older versions.

ANY OTHER REASON? ;)

//PERFORMANCE, PERFORMANCE, PERFORMANCE (I)

Don't,
Don't,
Don't.

Usual performance optimization advice.

Do,
Do,
Do.

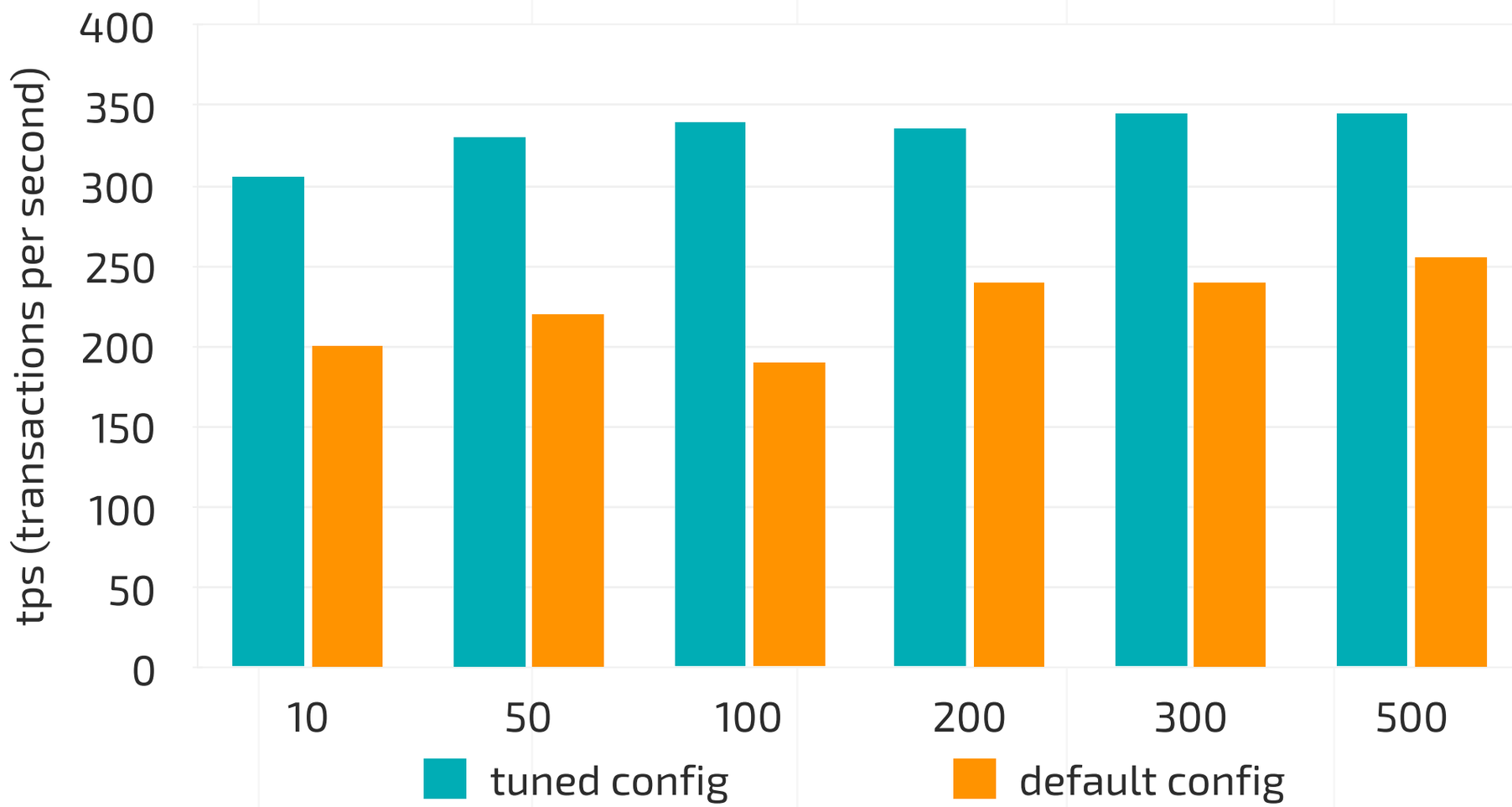
Usual **PostgreSQL** advice.
(*unless you run on a Raspberry PI 1st gen*).

- Run your own app-centric benchmarks.
- *pg_bench* is just one benchmark more.

ALL THE USUAL PRECAUTIONS ABOUT BENCHMARKS APPLY

//PERFORMANCE, PERFORMANCE, PERFORMANCE (II)

pg_bench, scale 2000, m4.large (2 vCPU, 8GB RAM, 1k IOPS).

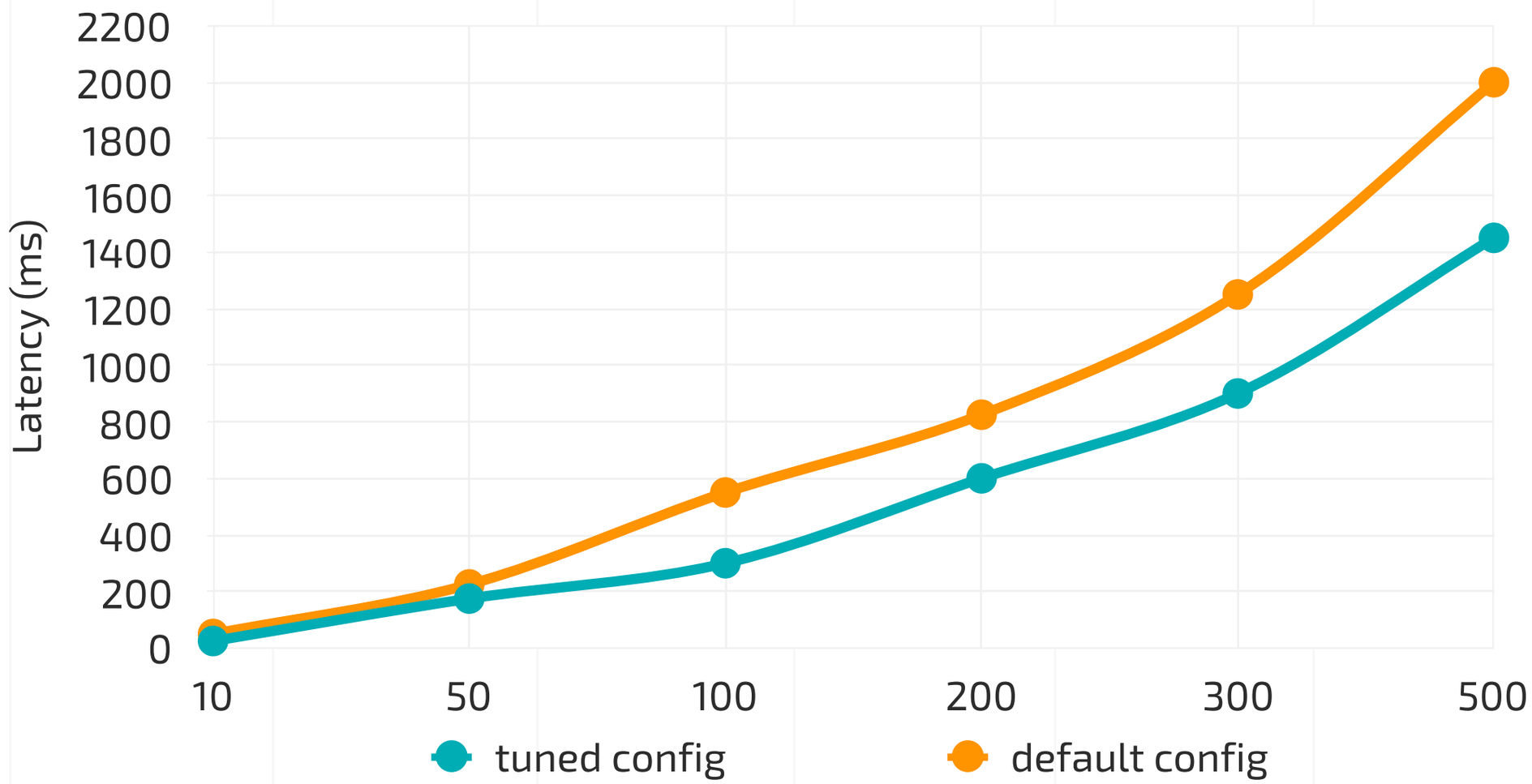


ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*

//PERFORMANCE, PERFORMANCE, PERFORMANCE (III)

pg_bench, scale 2000, m4.large (2 vCPU, 8GB RAM, 1k IOPS).



ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*

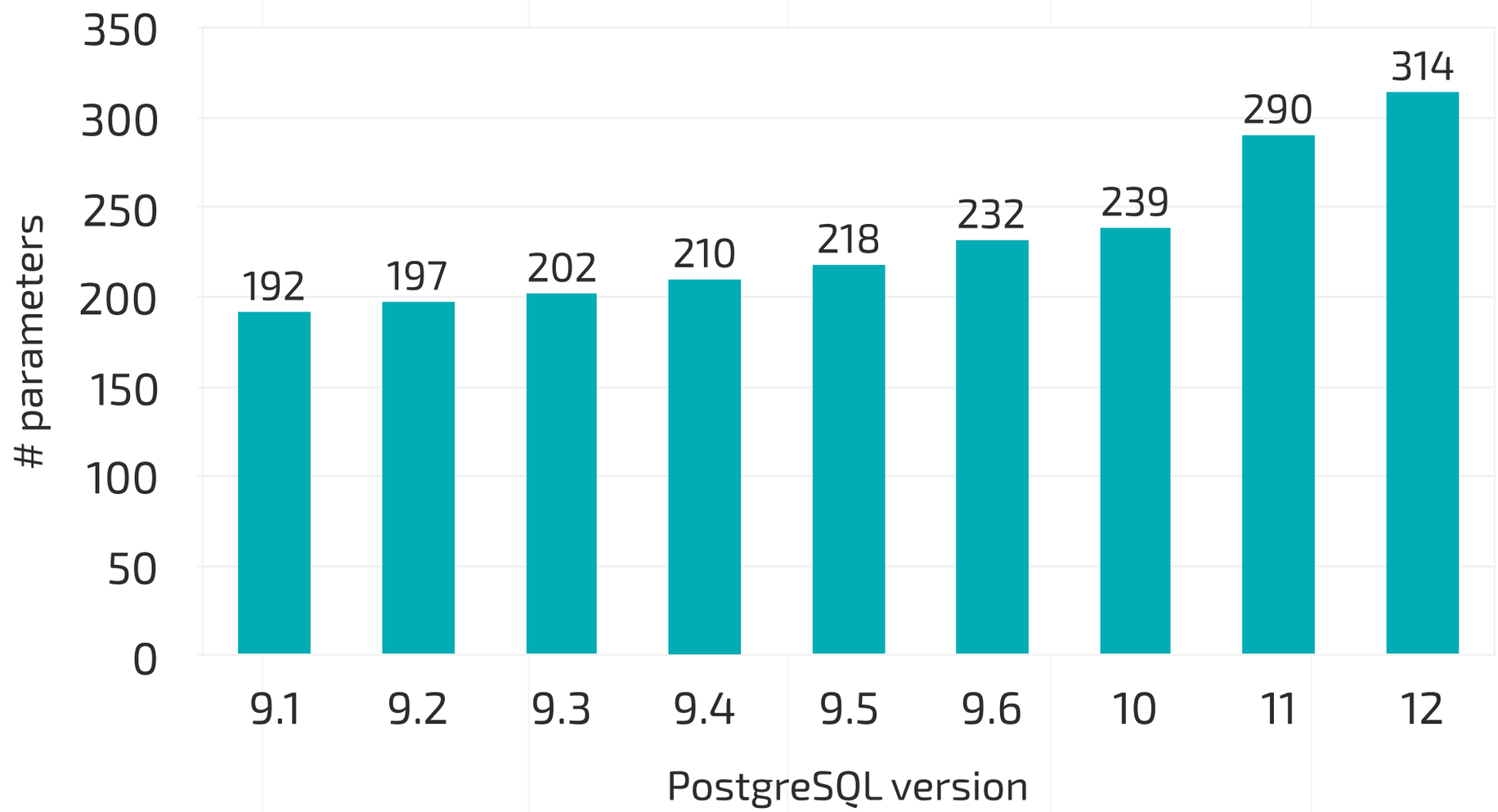
//POSTGRESQL.CONF PARAMETERS

- More than 200 parameters (*no kidding!*)
- Classified into 40 categories / subcategories.
- 650 lines, 23Kb *sample* config file.

How many to tune? **2? 5? 10? 20? 40? 100?**

- Parameters are *integer*, *real*, *string*, *enum*, *real* or *bool*. Numeric values may have units (or are unit-less).
- Some units are a bit uneasy (like “blocks of 8Kb”) or too synthetic (*cpu_tuple_cost*).

//TUNABLE POSTGRESQL.CONF PARAMETERS



ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*

SOME IDEAS ABOUT POSTGRESQL TUNING...

ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*

//DISCLAIMER

- No, you won't get here final numbers on how to tune your *postgresql.conf.*
- Only a few dozens parameters discussed here.
- Only hints provided: do your homework.
- My opinion may differ from other's.
- I am probably wrong.
- YMMV

(YOU GET THE POINT)

//INITDB

- Sometimes run on your behalf (Debian/Ubuntu), bad for selecting non defaults.
- ***-E (encoding)***. Use UTF-8 unless you know what you do.
- ***--locale***, ***--lc_collate***, ***--lc-ctype***.
- ***--username:*** If 'postgres' is not the superuser.
- ***--data-checksums:*** Enable them!

//DB CONNECTIONS 101 (I)

- ***max_connections*** is a hard limit.
- PostgreSQL will reject connections over this number.
- Users not happy.
- Default is 100.
- “My app has more 100 concurrent users!”

SOLUTION IS OBVIOUS: RAISE *MAX_CONNECTIONS*!

//DB CONNECTIONS 101 (II)

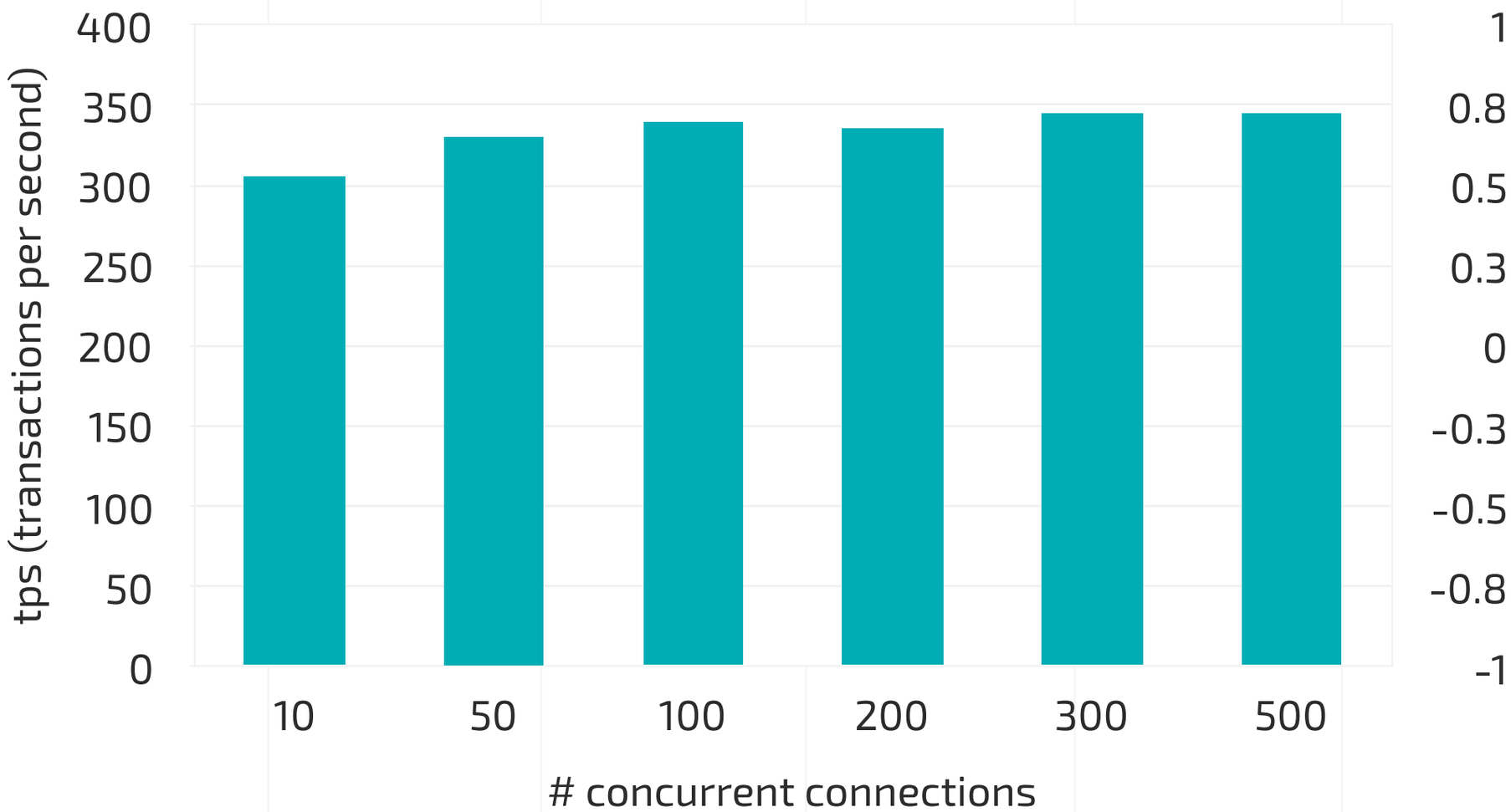
~~SOLUTION IS OBVIOUS: RAISE *MAX_CONNECTIONS*!~~

- One process per connection (excl. parallelism!)
- One process handled by one core
- How many cores do you have?
- Sure, you have a multi-process, time-sharing OS but what is the scheduling overhead with many processes?

EVEN WORSE: CACHE TRASHING!

//DB CONNECTIONS 101 (III)

pg_bench, scale 2000, m4.large (2 vCPU, 8GB RAM, 1k IOPS).

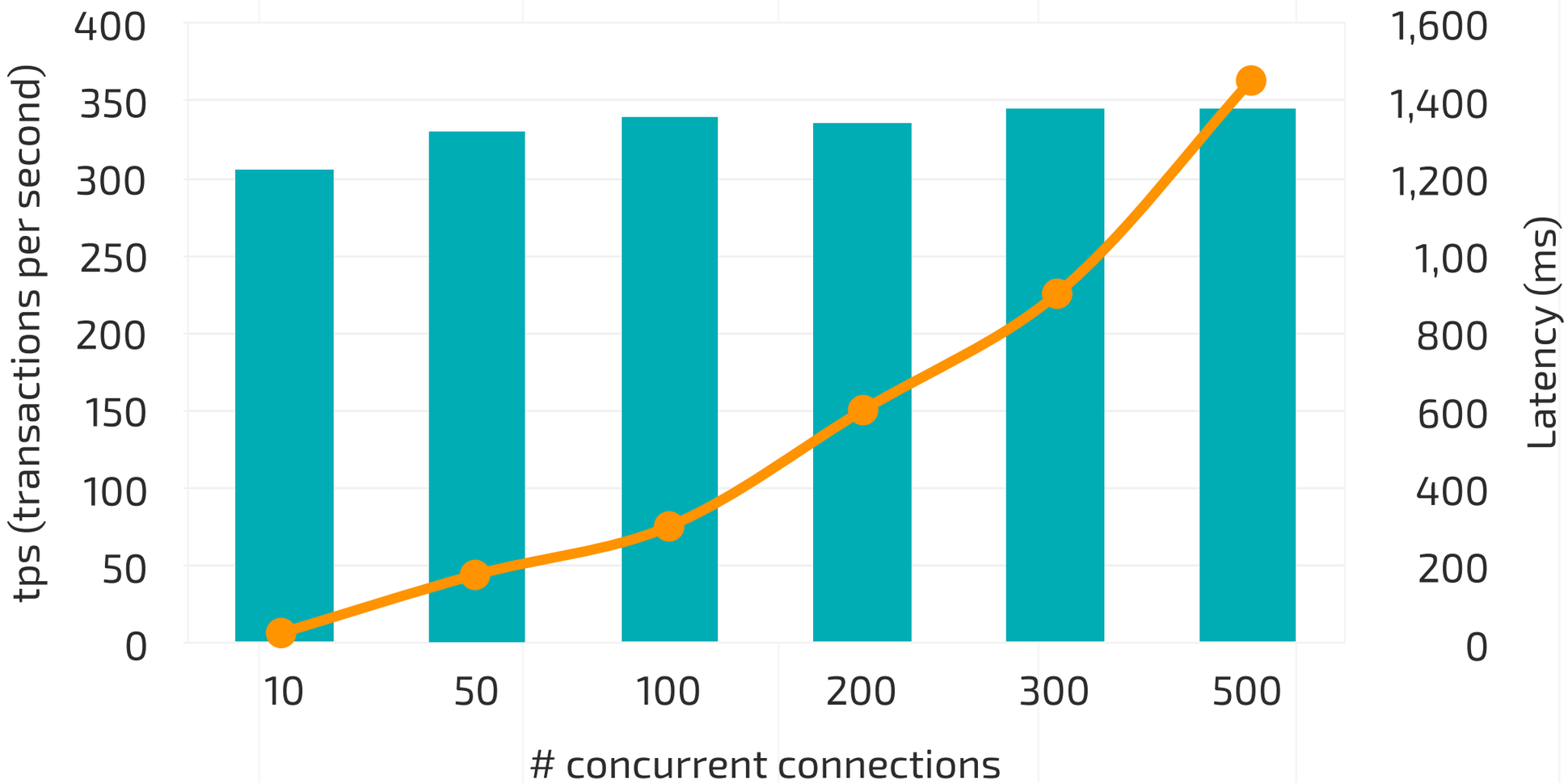


ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*

//DB CONNECTIONS 101 (III)

pg_bench, scale 2000, m4.large (2 vCPU, 8GB RAM, 1k IOPS).



ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*

//DB CONNECTIONS 101 (V)

- Solution is obvious: **lower *max_connections***!
- But how we solve the connection refused problem?
- ***PgBouncer***!
- Size your connections almost 1:1 *pgbouncer:max_conns.*
- Use this formula:

$$\text{CONNECTIONS} = \frac{\text{CORES}}{\text{\% EFFECTIVE UTILIZATION CONNECTION}} \text{ X SCALE FACTOR}$$

//SHARED_BUFFERS

- The first recommendation everybody tells you.
- Set it to 1/4th of RAM and *effective_cache_size* 3/4th.
- ~~Done!~~
- 1/4th too low on low memory, too high on high memory.
- **Benchmark, benchmark, benchmark.**
- Is the database dedicated in the host? OS memory?
- How much memory other PostgreSQL parts use, like *maintenance_work_mem* or all the memory used by query processes?

//WORK_MEM

- Max local process memory used for operations like sort and joins in queries.
- Not written in stone: users can SET it overriding its value.
- But if more memory is used, it spills to disk (and may use different algorithm) reducing performance.
- Not the same if you are OLTP, OLAP, DW (small to very large).
- Raise it from defaults, but don't forget it could be times (*max_connections* * max nodes query).

//OTHER MEMORY/DISK TUNABLES

- ***maintenance_work_mem***: vacuum, create index, check FKs… raise it.
- ***{min,max}_wal_size***: it's only disk space, but too low will cause excessive checkpoints. Make min at least 1GB, max several GB up to 50-100GB.
- ***stats_temp_directory***: run on a not very small RAMdisk.

//THIS REQUIRES RESTART, THINK CAREFULLY

- ***listen_addresses*** (take care with '*'), port.
- ***ssl:*** activate only if needed, use pooling!
- ***huge_pages:*** benchmark, benchmark, benchmark (typically off).
- ***shared_preload_libraries:*** add your extensions beforehand!
- ***logging_collector:*** *on*.
- ***wal_level:*** replica or *logical*.
- ***archive_mode***, ***archive_command*** = *'/bin/true'* if you don't use archiving.
- ***cluster_name***.

//THE TYRANNY OF *MAX_**

- Most of the *max_** also require restart.
- Sometimes hard to estimate, but restarting the database is pretty bad: raise limits and control usage.
- ***max_wal_senders:*** replicas, backups, make room.
- ***max_replication_slots***.
- ***max_worker_processes***, ***max_parallel_workers_per_gather***, ***max_parallel_workers***.
- ***autovacuum_max_workers*** (# cores for cleanup?)
- ***max_prepared_transactions*** (2PC, 0 by default).

//AUTOVACUUM / VACUUM (I)

- Almost always too conservative.
- Bloat is one of the most frequent operational burdens.
- Hard to get it right: analyze and re-tune periodically.
- Some parameters are set to "*-1*" which means "look at these numbers from the vacuum parameters".
- ***autovacuum_{vacuum,analyze}_scale_factor:*** you may override on a per-table level.

//*AUTOVACUUM / VACUUM* (II)

General advice:

- Raise *vacuum_cost_limit* significantly.
- Reduce *autovacuum_vacuum_cost_delay*.
- Use more *autovacuum_max_workers* if you have many cores.

//CHECKPOINTS AND *BGWRITER*

- You typically want to spread checkpoints farther apart (raise *checkpoint_timeout*).
- *min_wal_size* 1GB min.
- Log checkpoints and look for warnings.
- Raise *checkpoint_completion_target*, eg. 0.9, but study your I/O patterns, *shared_buffers*, *wal size*.
- Increase bgwriter activity, very conservative default:
  - Decrease *bgwriter_delay*.
  - Increase *bgwriter_lru_maxpages*.

//LOGGING

- “Only” 24 parameters (plus some others).
- Spend some time here, It pays off when analyzing your db.
- Turn on:
  - *logging_collector*.
  - *log_checkpoints*.
  - *log_connections*, *log_disconnections*.

//OTHER INTERESTING PARAMETERS

- *password_encryption* use SHA-256 if possible (>= PG 10).
- *effective_io_concurrency* (how many “spindles” your I/O has?)
- *max_standby_{archive,streaming}_delay* and *hot_standby_feedbak:* keep replication query conflicts burden on the primary or secondaries?
- *default_statistics_target:* if setting by table is not your job.
- Adjust the *random_page_cost* / *seq_page_cost* (4.0 / 1.0 by default), so that it is lower on SSD (start with 1.x) or indexes may not be used when it could be beneficial.

WAS THAT TOO MUCH? TOOLS TO HELP?

ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*

//WAS THAT TOO MUCH? TOOLS TO HELP? (I)

Parameters of your system

DB Version — what is this?

10

OS Type — what is this?

Linux/OS X

DB Type — what is this?

Web applications

Total Memory (RAM) — what is this?

16 GB

Number of Connections — what is this?

200

Generate

PostgreSQL settings (add/modify this settings in `postgresql.conf` and restart database):

```
# DB Version: 10
# OS Type: linux
# DB Type: web
# Total Memory (RAM): 16 GB
# Number of Connections: 200

max_connections = 200
shared_buffers = 4GB
effective_cache_size = 12GB
work_mem = 20971kB
maintenance_work_mem = 1GB
min_wal_size = 1GB
max_wal_size = 2GB
checkpoint_completion_target = 0.7
wal_buffers = 16MB
default_statistics_target = 100
```

//WAS THAT TOO MUCH? TOOLS TO HELP? (II)

PostgreSQL Configuration Tool 2.0 beta

TUNING ADVISOR ABOUT

SHARE! EXPORT

GNU/Linux Based | 64 Bits (x86-64) | 2

Application Profile: General web applications | Max connections *: 100 | PostgreSQL version: 9.6

RUN IT NOW!

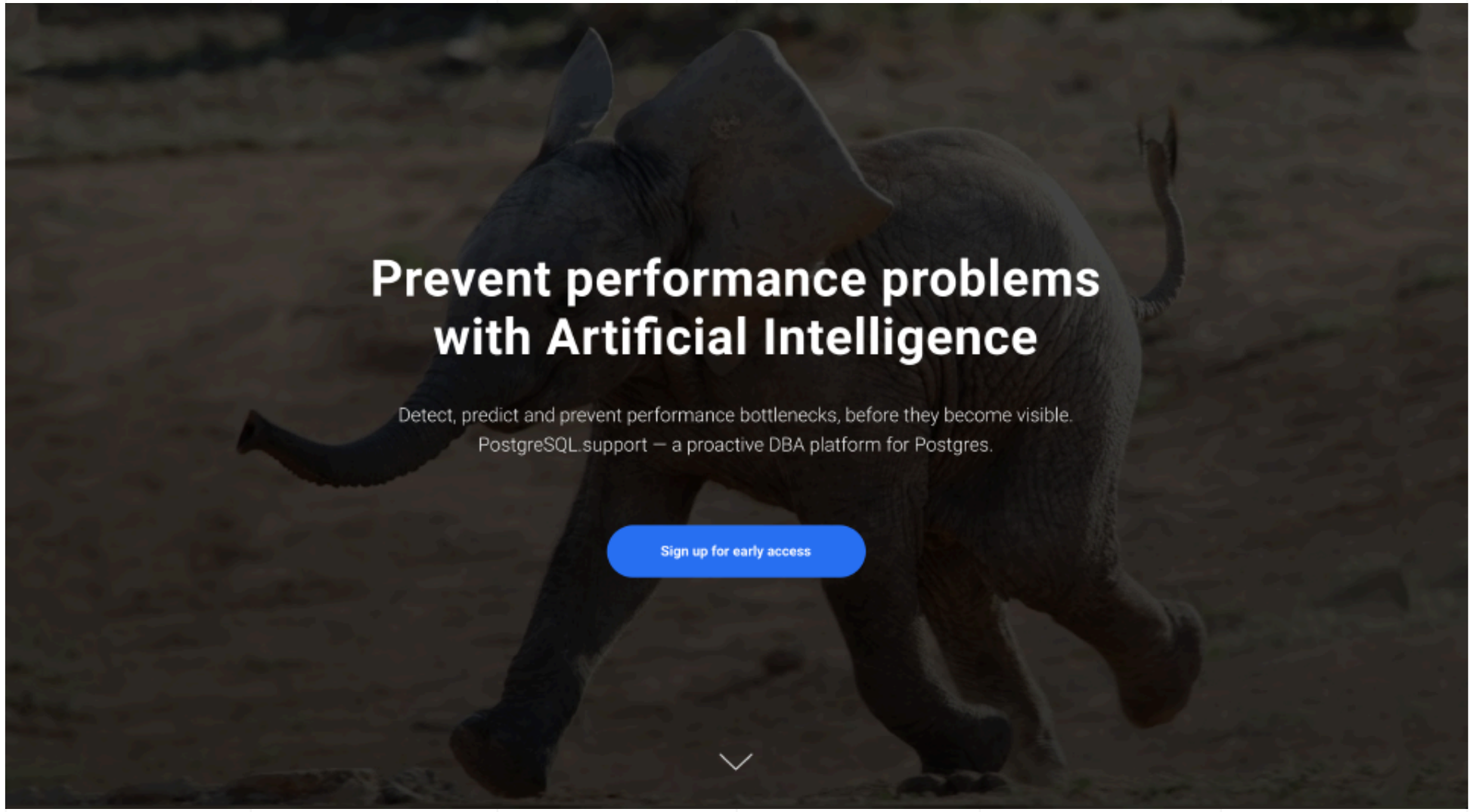
Memory Configuration

	Default Value	WEB	OLTP	DW	Mixed	Desktop
shared_buffers *(integer)*	128MB	512MB	512MB	512MB	512MB	128MB
effective_cache_size *(integer)*	4GB	2GB	2GB	2GB	2GB	512MB
work_mem *(integer)*	4MB	20MB	20MB	10MB	10MB	4MB
maintenance_work_mem *(integer)*	64MB	128MB	128MB	256MB	128MB	128MB

Checkpoint Related Configuration

	Default Value	WEB	OLTP	DW	Mixed	Desktop
min_wal_size *(integer)*	80MB	512MB	1GB	2GB	512MB	2GB
max_wal_size *(integer)*	1GB	2GB	3GB	6GB	2GB	1GB
checkpoint_completion_target *(floating)*	0.5	0.7	0.9	0.9	0.9	0.5

//WAS THAT TOO MUCH? TOOLS TO HELP? (III)



POSTGRESQL WANTS A NEW CONFIGURATION TOOL

(IMVHO)

ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*

//POSTGRESQLCO.NF (I)

POSTGRESQL CONFIGURATION FOR *HUMANS*

POSTGRESQL**CO.NF** DOCUMENTATION PUBLIC CONFIGURATIONS FAQ

CONFIGURED

CATEGORIES

NAMESPACE	NAME	VALUE	INFORMATION	STATUS	RESTORE	EDIT
	archive_timeout	50	Forces a switch to the next WAL file if a new file has not been started within N seconds			
namespace_01	checkpoint_warning	true	Enables warnings if checkpoint segments are filled more frequently than this			
	ssl_passphrase_command_supports_reload	on	Also use ssl_passphrase_command during server reload			
namespace_02	wal_buffers	GREATEST({DBInstanceClassMemory*1024/63963136},65536)	Sets the number of disk-page buffers in shared memory for WAL			

+ ADD PARAMETER

SEPARATE OFFICIAL/CUSTOM

DEFAULT VALUES

CATEGORIES

NAME	VALUE	INFORMATION	TUNED	EDIT
allow_system_table_mods	off	Allows modifications of the structure of system tables		
application_name		Sets the application name to be reported in statistics and logs		
archive_command		Sets the shell command that will be called to archive a WAL file		
archive_mode	off	Allows archiving of WAL files using archive_command		
archive_timeout	0	Forces a switch to the next WAL file if a new file has not been started within N seconds		
array_nulls	on	Enable input of NULL elements in arrays		
authentication_timeout	60	Sets the maximum allowed time to complete client authentication		
autovacuum	on	Starts the autovacuum subprocess		
autovacuum_analyze_scale_factor	0.1	Number of tuple inserts, updates, or deletes prior to analyze as a fraction of reltuples		
autovacuum_analyze_threshold	50	Minimum number of tuple inserts, updates, or deletes prior to analyze		
autovacuum_freeze_max_age	200000000	Age at which to autovacuum a table to prevent transaction ID wraparound		
autovacuum_max_workers	3	Sets the maximum number of simultaneously running autovacuum worker processes		
autovacuum_multixact_freeze_max_age	400000000	Multixact age at which to autovacuum a table to prevent multixact wraparound		

ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*

//POSTGRESQLCO.NF (II)

- Drag&drop your *postgresql.conf*.
- Automatic validation.
- Publish and share your config.
- Download as *postgresql.conf*, *JSON*, *yaml*, *SQL*.
- Rest API.

SUBSCRIBE ON POSTGESQLCO.NF
TO BECOME A BETA TESTER!

//CONTACT

USA

1177 Avenue of the Americas, Suite 500
10036 New York (New York)
United States of America
+1 6464527168

ESPAÑA

Carretera de Fuencarral, 44
Edificio 4B, Loft 33
28108 Alcobendas (Madrid),
España
+34 91 867 55 54

ARGENTINA

Manuel Ugarte 2110,
C1428 Buenos Aires, Argentina
+54 11 5365-6900

www.ongres.com
info@ongres.com

ONGRES

POSTGRESQL CONFIGURATION FOR *HUMANS*