aiven

On the road to Oracle Diagnostic and Tuning Pack for PostgreSQL

Dirk Krautschick

Database Synergy Day 26.03.2025

#whoami

Dirk Krautschick
Senior Solution Architect

with Aiven since Nov 2023

18 years

DBA, Trainer, Consulting, Sales Engineering

PostgreSQL, Oracle, Kafka, Clickhouse, OpenSearch,...

Married, 2 Junior DBAs

Mountainbike, swimming, movies, music, hifi/home cinema, 8 bit computing

PostgreSQL User Group NRW

North Rhine-Westphalia, Germany

Founded Dec 2023

Feb 2024 Cologne, May 2024 Aachen, Oct 2024 Essen,

Feb 2025 Paderborn, ...

Upcoming events in the pipeline, stay tuned!

- Target is at least 3 meetups a year
- All around NRW

Disclaimer

Different audience, different perspectives

My experience, my honest opinion

Let‘s stay open minded

Always open for discussions

What happened so far…

There was a talk…

“Pro-Active Performance Analysis in PostgreSQL”

https://www.youtube.com/watch?v=ZRZ_hleWZag

About

- Performance problems overall
- Different analysis approaches
- Recommendations/usage of Extensions

What happened so far…

Massive feedback

Consensus in practical experiences

What about similarities to Oracle Database?

Many questions about “THAT LAST PART of the talk”

Motivation to do a Spin-Off talk :-)

The Oracle Approach

Collects almost everything per default (sometimes sampled)

Interpretation with

- Querying Views (obviously!)
- Statspack (basic, always available and “costless”)

The Oracle Approach

Collects almost everything per default (sometimes sampled)

Interpretation with

- Querying Views (obviously!)
- Statspack (basic, always available and “costless”)
- Diagnostic and Tuning Pack (expensive Option)
  - Only for Enterprise Edition
  - Several tools like ASH, AWR,...

The Oracle Diagnostic Pack

Automatic Database Diagnostic Monitor (ADDM)

Automatic Workload Repository (AWR)

Active Session History (ASH)

Data Dictionary-/v$ performance-Views

The Oracle Tuning Pack (requires Diagnostic Pack)

SQL Access Advisor

SQL Tuning Advisor

Real-Time SQL Monitoring

Data Dictionary-/v$ performance-Views

The Oracle Diagnostic/Tuning Pack

The PostgreSQL Approach

(in PostgreSQL core)

Everything is there...

- ...but only for **NOW!**

Obvious Sources

- Parameters, Sizing (at that time!)
- Information_schema, system catalogues

Main Challenge

- How to handle, keep and collect all that stuff!

What about logging...

PostgreSQL logging is awesome

Exhaustive possibilities

Straight and easy configuration

Be aware of storage and load

- High maintenance
- Evaluate Logging strategies

```
log_line_prefix = '%t [%p]:
user=%u,db=%d,app=%a,client=%h ,
…
log_parser_stats = off
log_planner_stats = off
log_executor_stats = off
log_statement_stats = on
…
log_checkpoints = on
log_connections = on
log_disconnections = on
log_lock_waits = on
log_temp_files = 0
log_autovacuum_min_duration = 0
log_error_verbosity = default
…
log_min_messages = debug5
log_min_error_statement = debug5
log_min_duration_statement = 0
log_min_duration_sample = 0
…
log_statement = 'all'
```

What about logging…

RECAP

Yes, pgbadger is awesome…but… :-)

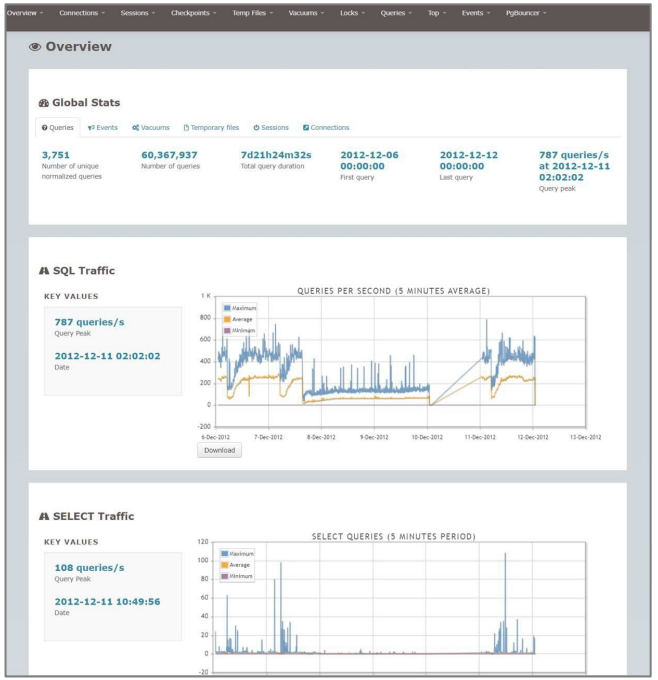
Advanced logging analysis reporting

https://pgbadger.darold.net/

Incremental daily/cumulative weekly reports

The right direction, but

- Massive logging still necessary
- Log file handling



What about monitoring...

For sure, monitoring is essential, but...

...it shows mostly

- that **something is slow**, sometimes maybe...
- **what is exactly slow**, but almost never...
- **why it is slow!**

PostgreSQL insights necessary

Deep dive or investigation as a next step anyway

What about monitoring...

For sure, monitoring is essential, but...

...it shows mostly

that **something is slow**, sometimes maybe...

what is exactly slow, but almost never...

why it is slow!

PostgreSQL insights necessary

Deep dive or investigation as a next step anyway

We are talking about Diagnostic/Tuning Pack...

...and NOT Enterprise Manager Cloud Control!!!

Philosophy question

Two database systems

- Two different mindsets
- Two way different levels of complexity

Out of the box comparison not fair and possible

Both solving the need of a relational database...

...but....

Under the Hood - Storage

Disk space usage (v17.4)

PGBIN

```
postgres@node0 /usr/pgsql-17 pgh170]# du -h
…
50M *     .
```

PGDATA

```
postgres@node1 /u00/postgres/17/pg17_cluster pgh170]# du -h
…
41M       .
```

* incl. pg_wait_sampling, pg_profile, …

Disk space usage (v21.5)

ORACLE_HOME

```
oracle@node0 /u00/oracle/product/21.0.0/dbhome_1]# du -h
…
9.0G      .
```

ORADATA

```
oracle@node0 /u00/oracle/oradata/CDB1]# du -h
1.2G    ./pdb1
1.2G    ./pdb2
1.0G    ./pdbseed
6.6G    .
```

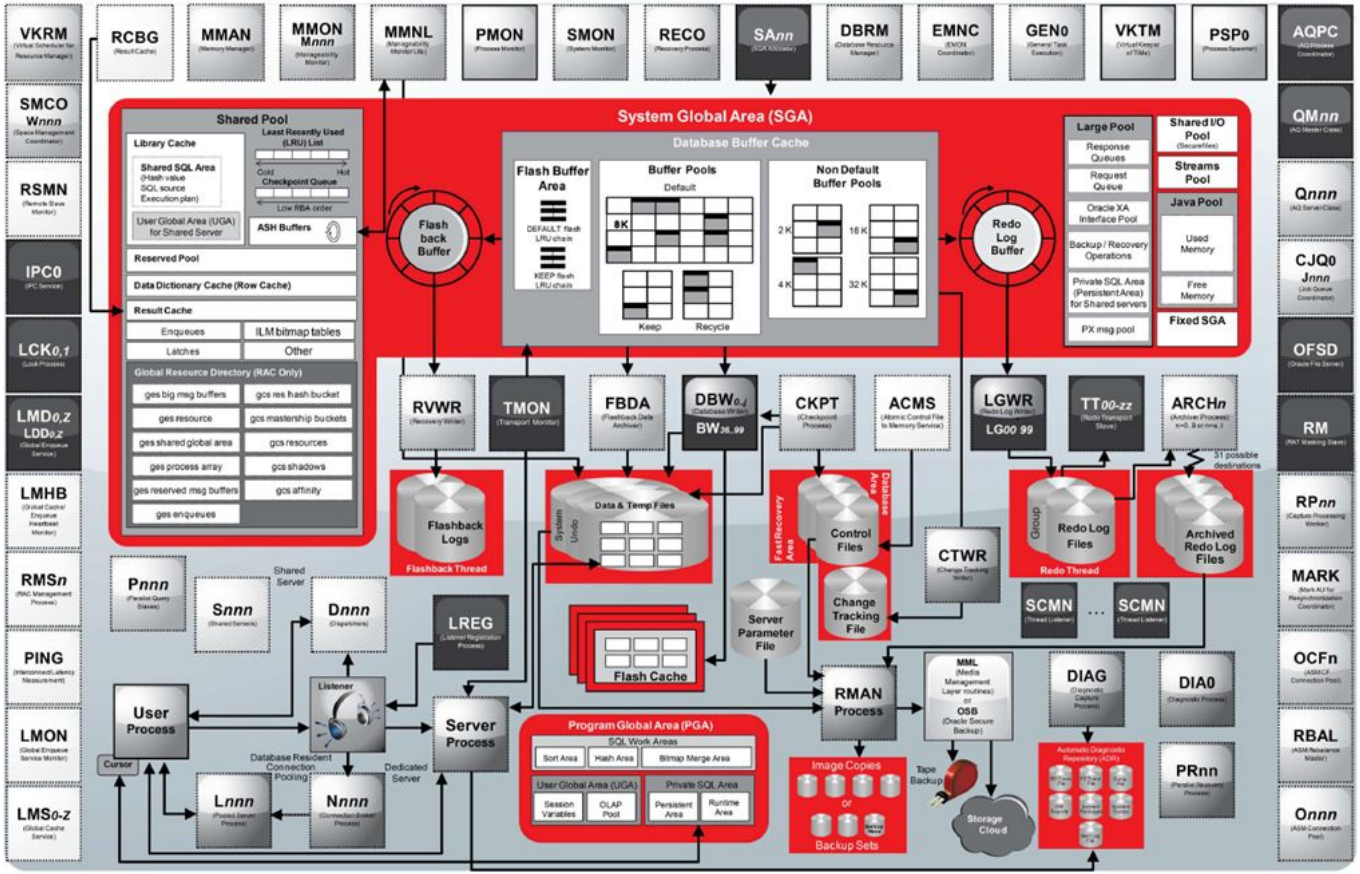
Under the Hood - Backend

~10 different backend process types

```
postgres@node0 ~]# systemctl status postgresql-17-core.service                    ...

postgresql-17-core.service - PostgreSQL 17 database server
…
   Memory: 23.5M
   CGroup: /system.slice/postgresql-17-core.service
        ├─ 1884 /usr/pgsql-17/bin/postmaster -D /u00/postgres/17/data
        ├─ 1938 postgres: core17: checkpointer
        ├─ 1939 postgres: core17: background writer
        ├─ 1940 postgres: core17: walwriter
        ├─ 1941 postgres: core17: autovacuum launcher
        ├─ 1943 postgres: core17: stats collector
        ├─ 1944 postgres: core17: pg_wait_sampling collector
        ├─ 1945 postgres: core17: logical replication launcher
        ├─ …
```

Under the Hood - Backend

ORACLE DATABASE



Under the Hood - Backend

~10 different backend process types

```
postgres@node0 ~]# systemctl status postgresql-17-core.service

postgresql-17-core.service - PostgreSQL 17 database server
…
   Memory: 23.5M
   CGroup: /system.slice/postgresql-17-core.service
        ├─ 1884 /usr/pgsql-17/bin/postmaster -D /u00/postgres/17/data
        ├─ 1938 postgres: core17: checkpointer
        ├─ 1939 postgres: core17: background writer
        ├─ 1940 postgres: core17: walwriter
        ├─ 1941 postgres: core17: autovacuum launcher
        ├─ 1943 postgres: core17: stats collector
        ├─ 1944 postgres: core17: pg_wait_sampling collector
        ├─ 1945 postgres: core17: logical replication launcher
        ├─ …
```

??? different backend processes (21.5)

```
SQL> select name, description from v$bgprocess;

…
```

Under the Hood - Backend

~10 different backend process types

```
postgres@node0 ~]# systemctl status postgresql-17-core.service

postgresql-17-core.service - PostgreSQL 17 database server
…
   Memory: 23.5M
   CGroup: /system.slice/postgresql-17-core.service
        ├─ 1884 /usr/pgsql-17/bin/postmaster -D /u00/postgres/17/data
        ├─ 1938 postgres: core17: checkpointer
        ├─ 1939 postgres: core17: background writer
        ├─ 1940 postgres: core17: walwriter
        ├─ 1941 postgres: core17: autovacuum launcher
        ├─ 1943 postgres: core17: stats collector
        ├─ 1944 postgres: core17: pg_wait_sampling collector
        ├─ 1945 postgres: core17: logical replication launcher
        ├─ …
```

524 different backend processes (21.5)

```
SQL> select name, description from v$bgprocess;

FMON  File Mapping Monitor Process
ACMS  Atomic Controlfile to Memory Server
BRDG  KSRPS Message Bus Bridge
LCK1  Lock Process 1
...
S000  Shared servers
TT03  Redo Transport
M003  MMON slave class 1

524 rows selected.
```

Under the Hood - Backend

~10 different backend process types

```
postgres@node0 ~]# systemctl status postgresql-17-core.service

postgresql-17-core.service - PostgreSQL 17 database server
…
   Memory: 23.5M
   CGroup: /system.slice/postgresql-17-core.service
        ├─ 1884 /usr/pgsql-17/bin/postmaster -D /u00/postgres/17/data
        ├─ 1938 postgres: core17: checkpointer
        ├─ 1939 postgres: core17: background writer
        ├─ 1940 postgres: core17: walwriter
        ├─ 1941 postgres: core17: autovacuum launcher
        ├─ 1943 postgres: core17: stats collector
        ├─ 1944 postgres: core17: pg_wait_sampling collector
        ├─ 1945 postgres: core17: logical replication launcher
        ├─ …
```

…but…to be honest…

“only” ~70-100 in usual environments

```
# ps auxw | grep -c "ora_"
73
```

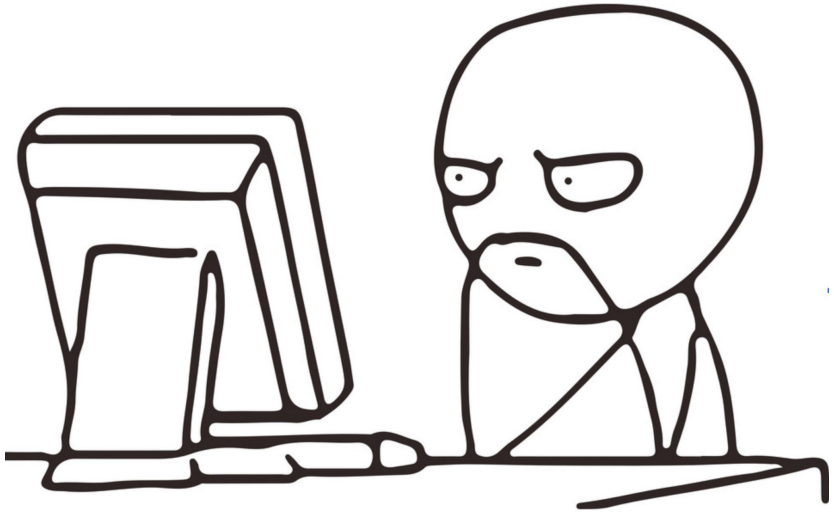
```
# ps auxwww | grep "ora_"

oracle    429876  0.0  1.6 1282444 61780 ?   Ss   00:06   0:00 ora_pmon_cdb2
oracle    429880  0.0  1.6 1282448 61900 ?   Ss   00:06   0:00 ora_clmn_cdb2
oracle    429884  0.0  1.6 1282196 62800 ?   Ss   00:06   0:00 ora_psp0_cdb2
...
oracle    501689  0.0  1.9 1283216 73588 ?   Ss   00:12   0:00 ora_m002_cdb2
oracle    548563  0.0  2.6 1284664 98448 ?   Ss   00:16   0:00 ora_m003_cdb2
```

All-In Monolith vs. Extensibility

- Common opinions of PostgreSQL
  - Missing things from the beginning
  - Challenges in selecting extensions
  - Compact, lightweight, learning curve
  - Tailored solution of choice
- Common opinions of Oracle Database
  - Happy place from the very beginning
  - Compatibility/Languages (if necessary, not that bad with PostgreSQL as well)
  - Ridiculously huge disk space usage, bloated software packages
  - Alternatives?

The Motivation

What is the **“comfort zone”** for (former?!) Oracle DBAs?



Oracle people want something like...

…mostly only those 2 solutions!

Active Session History (ASH)

Automatic Workload Repository (AWR)

The Idea

Active Session History (ASH)

Extensions

- PG_STAT_STATEMENTS
- PG_STAT_MONITOR (optional)
- PG_STAT_KCACHE
- PG_WAIT_SAMPLING

Automatic Workload Repository (AWR)

Extension

- PG_PROFILE

PG_STAT_STATEMENTS

Contrib, easy installation

Statement level statistics

Required by several monitoring tools

Statement based collection of e.g.

- Executions
- Execution times (min, max, average)
- Rows
- Blocks read/write
- ...

```
# \d pg_stat_statements
                    View "public.pg_stat_statements"
       Column         |        Type       | ...
----------------------+-------------------+-...
 userid               | oid               | ...
 dbid                 | oid               | ...
 queryid              | bigint            | ...
 query                | text              | ...
 total_plan_time      | double precision  | ...
…
 calls                | bigint            | ...
 total_exec_time      | double precision  | ...
 min_exec_time        | double precision  | ...
 max_exec_time        | double precision  | ...
 mean_exec_time       | double precision  | ...
 stddev_exec_time     | double precision  | ...
 rows                 | bigint            | ...
…
 blk_read_time        | double precision  | ...
 blk_write_time       | double precision  | ...
…
```

PG_STAT_STATEMENTS


```
# SELECT
    substring(query, 1, 50) as short_query,
    round(total_exec_time) as total_exec_time, calls,
    round(mean_exec_time) as mean_exec_time,
    round(100 * total_exec_time / (SELECT sum(total_exec_time) FROM stat_statements)) as percentage
  FROM
    pg_stat_statements
  ORDER BY
    percentage desc;
```

short_query	total_exec_time	calls	mean_exec_time	percentage
UPDATE pgbench_branches SET bbalance = bbalance +	7114	1500	5	63
UPDATE pgbench_tellers SET tbalance = tbalance + $	2506	1500	2	22
copy pgbench_accounts from stdin	664	1	664	6
UPDATE pgbench_accounts SET abalance = abalance +	194	1500	0	2
alter table pgbench_accounts add primary key (aid)	193	1	193	2
vacuum analyze pgbench_accounts	138	1	138	1

…

PG_STAT_KCACHE

Contrib, easy installation

Statistics about reads/writes on filesystem

Statistics about CPU usage

`pg_stat_statements` is required

```
postgres=# \d pg_stat_kcache_detail;

                    View "public.pg_stat_kcache"
      Column       |       Type       | Collation | Nullable | Default
-------------------+------------------+-----------+----------+---------
 datname           | name             |           |          |
 plan_user_time    | double precision |           |          |
 plan_system_time  | double precision |           |          |
 plan_minflts      | numeric          |           |          |
 plan_majflts      | numeric          |           |          |
 plan_nswaps       | numeric          |           |          |
 plan_reads        | numeric          |           |          |
 plan_reads_blks   | numeric          |           |          |
 plan_writes       | numeric          |           |          |
 plan_writes_blks  | numeric          |           |          |
…
 plan_nivcsws      | numeric          |           |          |
 exec_user_time    | double precision |           |          |
 exec_system_time  | double precision |           |          |
…
 exec_nsignals     | numeric          |           |          |
 exec_nvcsws       | numeric          |           |          |
 exec_nivcsws      | numeric          |           |          |
```

What’s still missing…?

…

What’s still missing… WAIT EVENTS!!!

PG_WAIT_SAMPLING

Not contrib, but in regular repositories

Wait events from `pg_stat_activity`

Sampled statistics of wait events

Combination with pg_stat_statements

github.com/postgrespro/pg_wait_sampling

Views

- `pg_wait_sampling_current`
- `pg_wait_sampling_history`
- `pg_wait_sampling_profile`

Functions

- `pg_wait_sampling_get_current(pid)`
- `pg_wait_sampling_reset_profile()`

PG_WAIT_SAMPLING


```
postgres=# select * from pg_wait_sampling_profile order by pid, count desc;

pid  | event_type |     event       |     queryid          |  count
-----+------------+-----------------------+----------------------+---------
1689 | IO         | DataFileWrite         |  2862011717192834034 | 4010499
1685 | IO         | DataFileRead    |  2862011717192834034 | 4010097
1686 | IO         | DataFileSync    | -4888004026240188267 | 4007477
1684 | Activity   | BgWriterHibernate     |  2862011717192834034 | 3991477
1683 | Activity   | CheckpointerMain|  1511417639870010300 | 3927957
1684 | Activity   | BgWriterMain    | -4888004026240188267 |   88494
3720 | Client     | ClientRead      | -4888004026240188267 |    2393
1685 | IO         | WALSync         |  6648255685428052402 |      65
3546 | Client     | ClientRead      | -4888004026240188267 |       1
3546 | IO         | DataFileRead    |  2862011717192834034 |       1

...
```

And now finally...collecting those data

Ring-Buffer-like settings in extensions are volatile

Several different retention

Several Views, Tables...but also volatile

How to handle the collection of all that information?

The Idea

- putting all in a repository/database
- while handling the retention of all information

PG_PROFILE - Introduction

Same Idea Statspack and AWR

Frequent snapshots of Performance Data in a Repository

Defined time periods and retention

Creation of reports based on those snapshots

Time frame between two or more snapshots

PG_PROFILE - Introduction

A good or the actual only(?) Example for AWR alternatives

Sample collection of

- System Catalogue, information_schema
- `PG_STAT_STATEMENTS`
- `PG_STAT_KCACHE`
- `PG_WAIT_SAMPLING`

PG_PROFILE - Development

Initiated by Andrei Zubkov

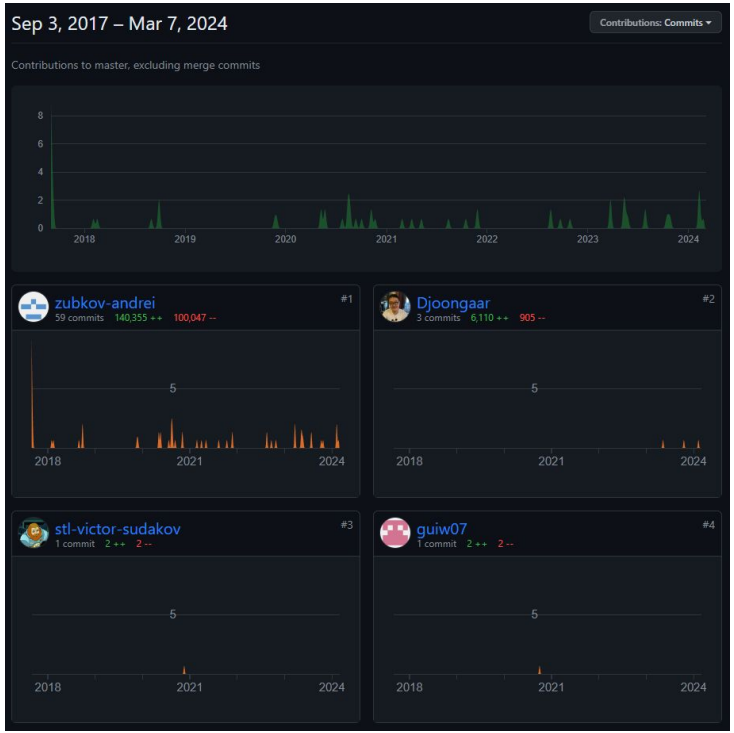
Individual Open Source license

https://github.com/zubkov-andrei/pg_profile

Starting Release v0.0.7 (Nov 2019)

Actual Release v4.8 (Jan 2025)

Pure PL/PGsql-based



PG_PROFILE - Example Report

Postgres profile report (58 -59)

pg_profile version 4.2

Server name: **node3_core15_postgres**

Report interval: **2023-06-25 19:00:01+00 - 2023-06-25 19:30:02+00**

Report sections

Server statistics

Database statistics

Database	Transactions			Block statistics			Tuples					Temp files		Si
	Commits	Rollbacks	Deadlocks	Hit(%)	Read	Hit	Ret	Fet	Ins	Upd	Del	Size	Files	
postgres	1424			100	35	4964931	5180119	3897083	3878	3347	2843			75.
Total	1424			100	35	4964931	5180119	3897083	3878	3347	2843			75.

Session statistics by database

Database	Timings			Sessions			
	Total	Active	Idle	Established	Abandoned	Fatal	Killed
postgres	1497.34	3.49	0.12	495			
Total	1497.34	3.49	0.12	495			

Tablespace statistics

Tablespace	Path	Size	Growth
pg_default		90 MB	312 kB
pg_global		531 kB	

Wait sampling

Wait events types

Wait event type	Statements Waited (s)	%Total	All Waited (s)	%Total
Activity			7279.99	81.62
Client			1556.66	17.45
IO			0.39	
LWLock			0.03	
Timeout			82.59	0.93
Total			8919.66	100

Top wait events (statements)

Top wait events detected in statements execution

Wait event type	Wait event	Waited (s)	%Total

Top wait events (All)

Top wait events detected in all backends

Wait event type	Wait event	Waited (s)	%Total
Client	ClientRead	1556.66	17.45
Activity	AutoVacuumMain	1472.61	16.51
Activity	LogicalLauncherMain	1472.61	16.51
Activity	WalWriterMain	1472.5	16.51
Activity	BgWriterHibernate	1441.63	16.16
Activity	CheckpointerMain	1389.65	15.58
Timeout	CheckpointWriteDelay	82.55	0.93
Activity	BgWriterMain	30.99	0.35
IO	DataFileWrite	0.25	
IO	WALSync	0.09	
Timeout	VacuumDelay	0.04	
LWLock	WALInsert	0.03	
IO	DataFileFlush	0.02	
IO	DataFileSync	0.02	
IO	ControlFileSyncUpdate	0.01	

Query ID	Database	User	Reads	%Total	Hits(%)	Elapsed(s)	Rows	Executions
[284d83ba74]								
b663e72bbf906c1a [30ba98ac8c]	postgres	postgres	28		100			

Top SQL by shared blocks read

Query ID	Database	User	Reads	%Total	Hits(%)	Elapsed(s)	Rows	Executions

Top SQL by shared blocks dirtied

Query ID	Database	User	Dirtied	%Total	Hits(%)	Dirtied	WAL	%Total	El
fb5ce79fc4bdf20f [0eeaf590a6]	postgres	postgres	832	832	100	90.63	6227 kB	84.33	
f8259241bf7cef37 [ab7afda8fa]	postgres	postgres	86	86	100	9.37	115 kB	1.56	

Top SQL by shared blocks written

Query ID	Database	User	Written	%Total	%BackendW	Hits(%)	Elapsed(s)	R
fb5ce79fc4bdf20f [0eeaf590a6]	postgres	postgres	35	3.44	100	100	2.53	

Top SQL by WAL size

Query ID	Database	User	WAL	%Total	Dirtied	WAL FPI	WAL records
fb5ce79fc4bdf20f [0eeaf590a6]	postgres	postgres	6227 kB	84.33	832	599	21596
f8259241bf7cef37 [ab7afda8fa]	postgres	postgres	115 kB	1.56	86	22	22

rusage statistics

Top SQL by system and user time

Query ID	Database	User	User Time		System Time	
			Exec (s)	%Total	Exec (s)	%Total
fb5ce79fc4bdf20f [0eeaf590a6]	postgres	postgres	1.42	76.83	0.06	42.4
4b097e9be01da20e [8093c244bf]	postgres	postgres	0.22	11.64	0.02	15.57
a75bdca7cac3636d [de1c8702bb]	postgres	postgres	0.07	4.05	0.01	10.75
cb3743bef713ee24 [12bdecef37]	postgres	postgres	0.04	1.91		2.96
f8259241bf7cef37 [ab7afda8fa]	postgres	postgres	0.02	1.28		

Top SQL by reads/writes done by filesystem layer

Query ID	Database	User	Reads	%Total	Writes	%Total

Complete list of SQL texts

Query ID	
8df9549c708fc694	SELECT DISTINCT ON (blocked_pid, locktype, blocking_pid) blocked_activity.pid AS blocked_pid, blocked_activi blocked_activity.query AS blocked_statement, blocking_activity.query AS current_statement_in_blocking_proces
8fbba6b1051e9899	SELECT ct.conname, ct.conkey, ct.confkey, nl.nspname AS fknsp, cl.relname AS fktab, nr.nspname AS refnsp, c
9c0db794731c91bc	SELECT nspname FROM pg_namespace nsp JOIN pg_proc pro ON pronamespace=nsp.oid WHERE proname = $1
9f4e11ca39a8b4c1	SELECT s.nspname AS schema_name, $1 AS package_name, f.proname AS function_name, $2::"char" AS function_type
a068feeeec1087c6	SET client_encoding TO 'utf8'
a106925175449189	SELECT nspname AS schema_name FROM pg_catalog.pg_namespace WHERE nspname = $1 OR nspname NOT LIKE $2
a75bdca7cac3636d	SELECT $2 AS package_name, f.proname AS function_name, $3::"char" AS function_type, f.prorettype::regtype AS s.nspname = $1
ae247e8d0d09785f	SELECT i.schemaname AS schema_name, i.indexrelname AS index_name, s.idx_scan, s.idx_tup_read, s.idx_tup_fet
b663e72bbf906c1a	SELECT n.nspname AS schema_name, c.relname AS view_name, pg_relation_size(c.oid) / $1 AS mview_size_mb, pg_
ca069d9d4625d451	WITH pg_stat_statistics AS (SELECT p.queryid, p.dbid, p.userid, substring(p.query, $1, $2) AS short_query, LEFT JOIN pg_database d ON (p.dbid = d.oid) order by p.queryid desc
cb3743bef713ee24	SELECT pg_catalog.version() AS version_string, ((select setting from pg_settings where name = $1)::decimal * AS effective_cache_size_mb, ((select setting from pg_settings where name = $13)::decimal * (select setting f wal_buffers_mb, (SELECT pg_postmaster_start_time())::timestamptz AS server_start_time
d38f8743c701e8e6	SELECT COALESCE(sum($1), $2) AS number_of_prepared_transactions FROM pg_catalog.pg_prepared_xacts
d4e4e1cfd8c55bdd	SELECT n.nspname AS schema_name, c.relname AS table_name, pg_relation_size(c.oid) / $1 AS table_size_mb, pg_
db11d8257cef5f93	SELECT s.schemaname as schema_name, s.relname as table_name, s.seq_scan, s.seq_tup_read, s.idx_scan, s.idx_t s.relid = i.relid JOIN pg_class c ON s.relid = c.oid WHERE c.relkind != $1
dc6cee39214b6118	SELECT version()
f8259241bf7cef37	SELECT schemaname AS schema_name, tablename AS table_name, sml.relpages AS estimated_pages, target_pages, R($6 ELSE ((block_size - page_header) / estimated_bytes_per_tuple) END) AS target_pages, estimated_bytes_per_t ELSE null_bitmap_size % max_align END)) + item_ptr::bigint AS estimated_bytes_per_tuple FROM (SELECT schem WHEN substring(v,$15,$16) IN ($17,$18,$19) THEN $20 ELSE $21 END AS tuple_header, CASE WHEN substring(v,$22
f915a933fb9cdb00	SELECT st.relid,st.schemaname,st.relname,st.seq_scan,st.seq_tup_read,st.idx_scan,st.idx_tup_fetch,st.n_tup_ins,st.r relsize,$2 relsize_diff,class.reltablespace AS tablespaceid,class.reltoastrelid,class.relkind,class.relpages
fb28b5e485efe49b	SELECT st.*,stio.idx_blks_read,stio.idx_blks_hit,$1 relsize,$2,pg_class.reltablespace as tablespaceid,(ix.ir st.indexrelid) LEFT OUTER JOIN pg_catalog.pg_constraint con ON (con.conindid = ix.indexrelid AND con.contype
fb5ce79fc4bdf20f	SELECT profile.take_sample()
45f640d85b0f9e4	WITH max_age AS (SELECT $1 as max_old_xid, setting AS autovacuum_freeze_max_age FROM pg_catalog.pg_settings (oldest_current_xid/autovacuum_freeze_max_age::float)) AS percent_towards_emergency_autovac FROM per_databas
e42d4769165b4eb	SELECT setting FROM pg_settings WHERE name=$1
fed2778f1d749ae	SELECT $1
22ee9e843be1d0bd	SELECT DISTINCT ON (name) name, setting, unit FROM pg_catalog.pg_settings
4b097e9be01da20e	SELECT c.relname AS view_name, c.relkind AS view_type, c.relispopulated AS ispopulated, spc.spcname AS table
4b5efda716747fde	SELECT datname AS database_name, datallowconn AS connections_allowed, pg_encoding_to_char(encoding) AS encoc
4d65a4b5cf1d099d	SELECT datname AS database_name, pid AS procpid, usename, client_addr, client_port, backend_start, xact_sta
5a1c6d5a186940e9	SELECT f.funcid,f.schemaname,f.funcname,pg_get_function_arguments(f.funcid) AS funcargs,f.calls,f.total_time
60ee2aded793ec13	SELECT c.relname AS index_name, r.relname AS table_name, i.indkey AS ind_keys FROM pg_catalog.pg_class c, pg
62eebb8d9a4cba8d	SELECT checkpoints_timed, checkpoints_req, buffers_clean, buffers_checkpoint, maxwritten_clean, buffers_back
6888a766116464 04	SELECT c.relname AS table_name, array_length(array_agg(i.indexrelid), $2) > $3 AS has_primary_key FROM (SELE

PG_PROFILE - Report content

Server Statistics

SQL query Statistics

Wait Event Statistics

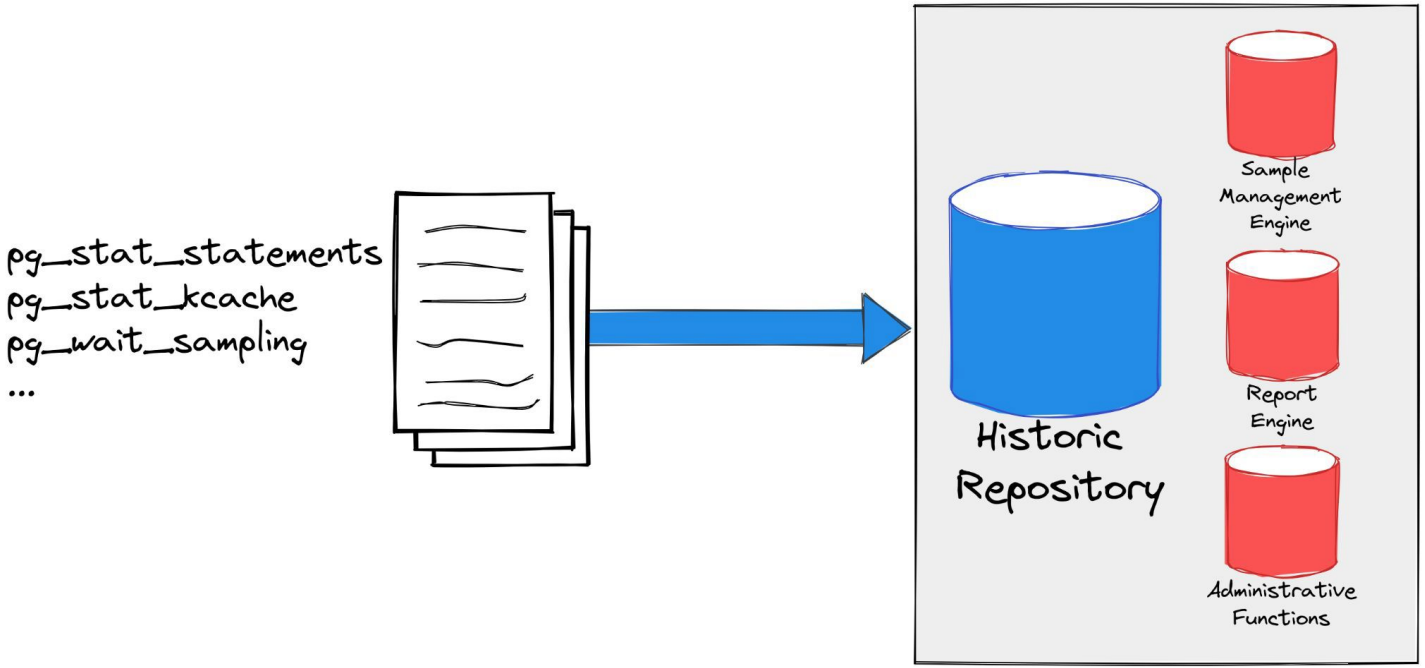
Schema Object Statistics

User Function Statistics

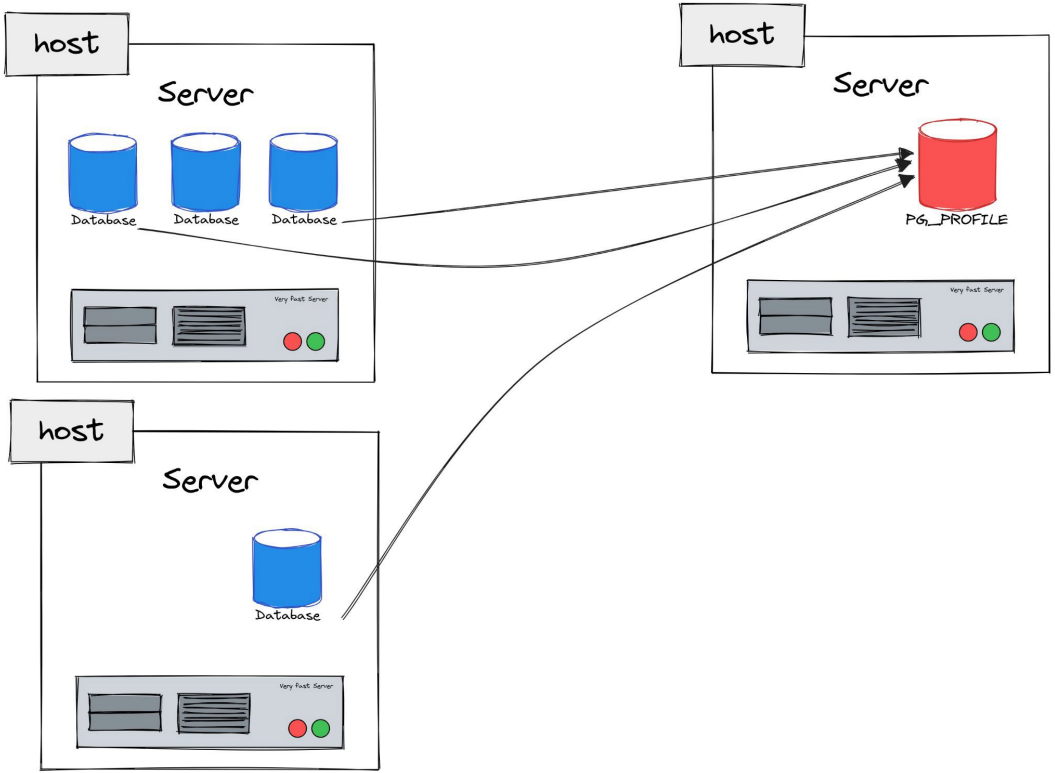
Vacuum-related stats

Cluster settings

PG_PROFILE - Architecture



PG_PROFILE - Architecture



PG_PROFILE - Prerequisite and Setup

Extension `dblink` (part of contrib)

Repositories, e.g.

```
# sudo dnf install pg_profile_17
```

Direct from github

```
# curl -LJO https://github.com/zubkov-andrei/pg_profile/releases/download/4.8/pg_profile--4.8.tar.gz
# sudo tar xzf pg_profile--4.8.tar.gz --directory $(pg_config --sharedir)/extension
```

PG_PROFILE - Prerequisites

Create Schema for Repository (optional)

```
CREATE SCHEMA profile;
```

Activate necessary Extensions

```
CREATE EXTENSION pg_profile SCHEMA profile;
CREATE EXTENSION dblink;
```

PG_PROFILE - Prerequisites on source DBs

Preload Extensions **(of your choice!)**

Set few recommended Parameters

```
# vi $PGDATA/postgresql.conf
…
shared_preload_libraries = 'pg_stat_statements, pg_wait_sampling, pg_stat_kcache'
…
track_activities = on
track_counts = on
track_io_timing = on
track_wal_io_timing = on
track_functions = all
```

PG_PROFILE - Configuration

Consider extension parameters

```
pg_profile.topn = 20
pg_profile.max_sample_age = 7
pg_profile.track_sample_timings = off
pg_profile.max_query_length = 20000
```

As well for the related extensions, like e.g.

```
pg_stat_statements.max = 10000
pg_stat_statements.track = 'all'
```

PG_PROFILE - Adding Clusters/Servers for Collection

Add Server/Database

```
SELECT profile.create_server('cluster17','host=node0 dbname=postgres port=5432');
```

Other functions

```
profile.drop_server(server name)
profile.enable_server(server name)
profile.disable_server(server name)
profile.show_servers()
```

…

PG_PROFILE - Collecting Data

Take a sample

```
select * from profile.take_sample();
select * from profile.take_sample('core16');
```

Check existing samples

```
select * from profile.show_samples();
select * from profile.show_samples('core16');
```

PG_PROFILE - Baselines

Tagged Group of Samples

Independent Retention

E.g. for bulk operations, load testings,...

Example handling

```
select * from profile.show_baselines();
select * from profile.create_baseline('cluster17', 'pgbench_run' , 70, 71);
```

PG_PROFILE - Collecting Data

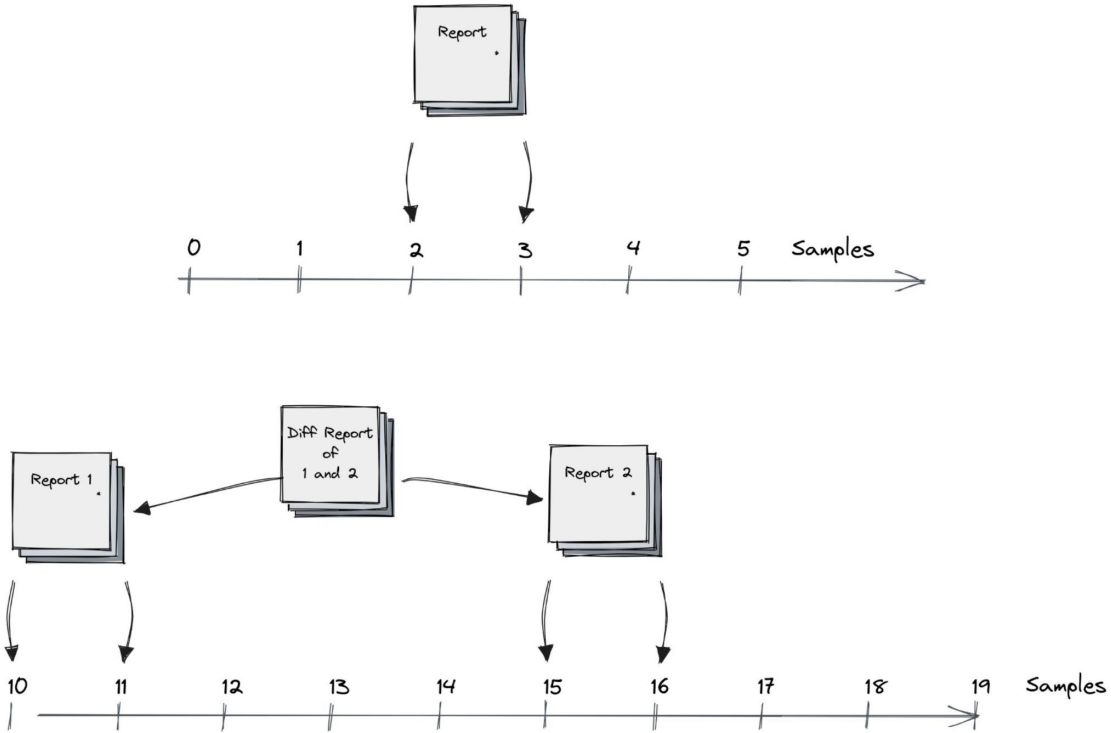
Best Practice Strategy

- Frequented 30 Min Samples, starting point
- Consider manual created Samples
- Baselines

Putting into cron

```
*/30 * * * * psql -c 'SELECT profile.take_sample()' > /dev/null 2>&1
```

PG_PROFILE - Creating Reports



PG_PROFILE - Creating Reports

Standard Report

```
psql -Aqtc \
"SELECT profile.get_report('core17',8, 9)" \
-o report_8_9.html
```

Diff Report Report

```
psql -Aqtc \
"SELECT profile.get_diffreport('core16', 8, 9, 11, 12)" \
-o diff_report_8_9-11_12.html
```

PG_PROFILE - A look into the repository schema

```
# \dt profile.*
List of relations
Schema | Name | Type | Owner
---------+-------------------------------+-------------------+----------
profile | baselines | table | postgres
profile | bl_samples | table | postgres
profile | funcs_list | table | postgres
profile | import_queries | table | postgres
profile | import_queries_version_order | table | postgres
profile | indexes_list | table | postgres
profile | last_stat_archiver | table | postgres
profile | last_stat_cluster | table | postgres
profile | last_stat_database | partitioned table | postgres
profile | last_stat_database_srv1 | table | postgres
profile | last_stat_database_srv2 | table | postgres
profile | last_stat_database_srv4 | table | postgres
profile | last_stat_indexes | partitioned table | postgres
profile | last_stat_indexes_srv1 | table | postgres
profile | last_stat_indexes_srv2 | table | postgres
profile | last_stat_indexes_srv4 | table | postgres
profile | last_stat_kcache | partitioned table | postgres
profile | last_stat_kcache_srv1 | table | postgres
profile | last_stat_kcache_srv2 | table | postgres
profile | last_stat_kcache_srv4 | table | postgres
profile | last_stat_statements | partitioned table | postgres
profile | last_stat_statements_srv1 | table | postgres
profile | last_stat_statements_srv2 | table | postgres
profile | last_stat_statements_srv4 | table | postgres
profile | last_stat_tables | partitioned table | postgres
profile | last_stat_tables_srv1 | table | postgres
profile | last_stat_tables_srv2 | table | postgres
profile | last_stat_tables_srv4 | table | postgres
profile | last_stat_tablespaces | partitioned table | postgres
profile | last_stat_tablespaces_srv1 | table | postgres
profile | last_stat_tablespaces_srv2 | table | postgres
profile | last_stat_tablespaces_srv4 | table | postgres
…
…
profile | last_stat_user_functions | partitioned table | postgres
profile | last_stat_user_functions_srv1 | table | postgres
profile | last_stat_user_functions_srv2 | table | postgres
profile | last_stat_user_functions_srv4 | table | postgres
profile | last_stat_wal | table | postgres
profile | report | table | postgres
profile | report_static | table | postgres
profile | report_struct | table | postgres
profile | roles_list | table | postgres
profile | sample_kcache | table | postgres
profile | sample_kcache_total | table | postgres
profile | sample_settings | table | postgres
profile | sample_stat_archiver | table | postgres
profile | sample_stat_cluster | table | postgres
profile | sample_stat_database | table | postgres
profile | sample_stat_indexes | table | postgres
profile | sample_stat_indexes_total | table | postgres
profile | sample_stat_tables | table | postgres
profile | sample_stat_tables_total | table | postgres
profile | sample_stat_tablespaces | table | postgres
profile | sample_stat_user_func_total | table | postgres
profile | sample_stat_user_functions | table | postgres
profile | sample_stat_wal | table | postgres
profile | sample_statements | table | postgres
profile | sample_statements_total | table | postgres
profile | sample_timings | table | postgres
profile | samples | table | postgres
profile | servers | table | postgres
profile | stmt_list | table | postgres
profile | tables_list | table | postgres
profile | tablespaces_list | table | postgres
profile | wait_sampling_total | table | postgres
(64 rows)
```

PG_PROFILE - A look into the repository schema

Global Retention Policy

```
pg_profile.max_sample_age
```

Server Retention Policy

```
pg_profile.set_server_max_sample_age()
```

PG_PROFILE - A look into the repository schema

Data Growth?

```
WITH schemas AS (
SELECT
        schemaname as name, sum(pg_relation_size(quote_ident(schemaname) || '.' || quote_ident(tablename)))::bigint as size
        FROM pg_tables GROUP BY schema name),db AS ( SELECT pg_database_size(current_database()) AS size
) SELECT schemas.name, pg_size_pretty(schemas.size) as absolute_size,
schemas.size::float / (SELECT size FROM db) * 100 as relative_size FROM schemas;

name                | absolute_size | relative_size
--------------------+---------------+---------------------
public              | 41.00 MB      | 51.55
pg_catalog          | 0.50 MB       | 6.02
information_schema  | 0.09 MB       | 0.10
profile             | 18.00 MB      | 22.04
(4 rows)
```

Conclusion – My 2 Cents

Extensibility is a advantage, not just a lack of features!

Very clean, pragmatic way to handle performance data

Sustainable repository approach

PG_PROFILE is **NOT** a plagiat of AWR, it fits a requested need of DBAs!!!

OF COURSE (!!!) ... not exactly the amount like the Oracle packs

- But still a big point for considering Oracle folks on migrations
- But mostly perfect for nearly all common problems

Conclusion – What is missing?

Dream of having such in contrib

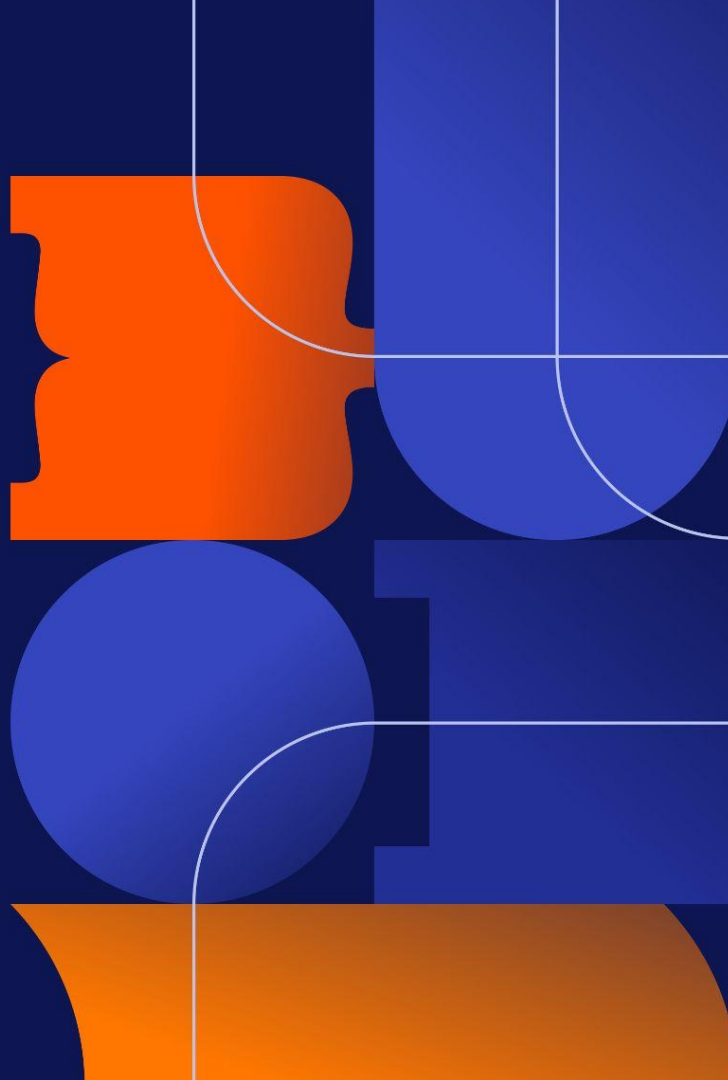
- ...or at least in all common Repos
- ...**or even better in PostgreSQL Core :-)**

Own Job handling would be nice, but cron is fine!

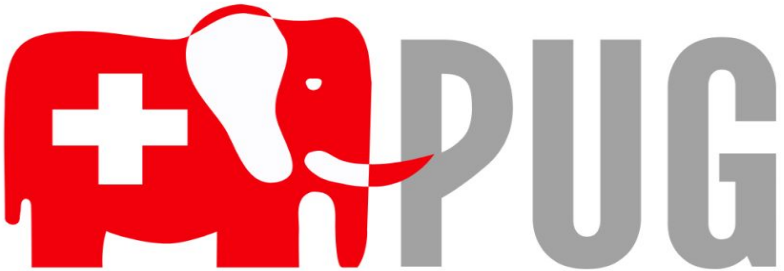
Much room for even more improvements and ideas

Questions?

Please do ask or
catch me outside!



THANK YOU!



www.pgday.ch

Swiss PGDay 2025

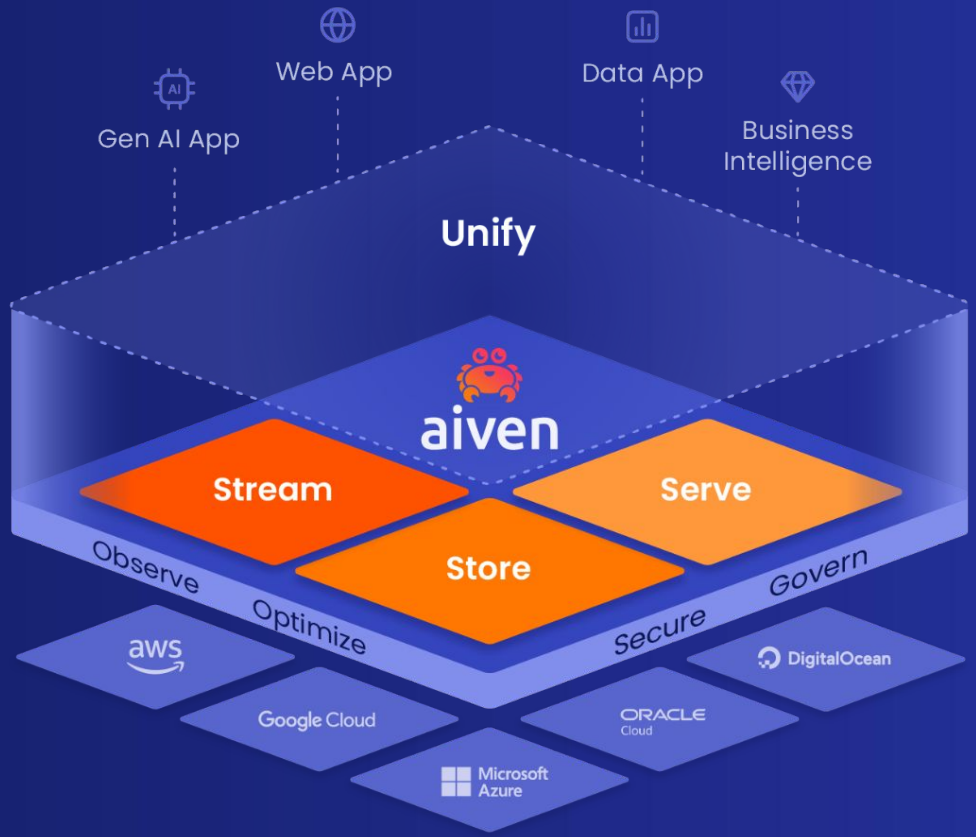
June 26 & 27
in Rapperswil (SG)

www.pgday.ch

aiven

Your Trusted Data & AI Platform

Streaming | Database Optimization | Analytics | Search | Data Warehousing | In-Memory Caching



One cloud data platform

STREAM

Event streaming

Aiven for **Apache Kafka®** and **Kafka® Connect**

Event stream processing

Aiven for **Apache Flink®**

STORE

Relational databases

Aiven for **AlloyDB Omni**

Aiven for **PostgreSQL®**

Aiven for **MySQL**

Key-value database

Aiven for **Valkey**

Aiven for **Dragonfly**

SERVE

Data warehouse

Aiven for **ClickHouse®**

Time series database

Aiven for **Metrics**

Search engine

Aiven for **OpenSearch®**

Data visualization

Aiven for **Grafana®**

UNIFIED PLATFORM

Host

aws

Google Cloud

Microsoft Azure

DigitalOcean

ORACLE Cloud

Bring your own cloud

Deploy

Terraform

Kubernetes

API REST API

CLI Aiven CLI

Aiven Console

Integrate

Datadog

Prometheus

AWS CloudWatch

GCP Monitoring

MongoDB

AWS S3

GCP BigQuery

Couchbase

Snowflake

splunk> Splunk

Sumologic

Debezium

GCP Pub/Sub

GCP Storage

Customers


okta


snyk


DOORDASH


priceline


fiverr.


Norauto


DECATHLON


GTL


ACTIVISION
BLIZZARD


MIRAKL


GOV.UK


goto financial


spare


Schibsted


TOYOTA


paf


CONRAD


adeo


ometria


WÄRTSILÄ