

pg_statviz

A minimalist **extension and utility pair**
for time series analysis and visualization
of **PostgreSQL** internal statistics

Jimmy Angelakos

Senior Principal Engineer
Deriv

Extension Ecosystem Summit

2024-10-22

PostgreSQL internal statistics

- The **Cumulative Statistics System** (FKA Statistics Collector)
 - Postgres subsystem that collects info about system activity
- Dynamic statistics (right now)
- Cumulative statistics, but can be reset
- Table/index information on row & disk block levels
- This info can be reported via views

Motivation

i

- Why?
 - Track PostgreSQL performance over time and potentially perform tuning or troubleshooting
- Yes, but why?
 - So that people can understand their system better at a glance 🙄🙄

Motivation

ii

- Working with customers
 - Who often have no idea how their database is performing
 - Or why it's not working well
- Their monitoring tools don't give them insights

How?

- Created for:
 - Snapshotting **cumulative and dynamic** statistics
 - Performing **time series analysis** on them
- Utility can produce visualizations for selected time ranges on the stored stats snapshots

Design Philosophy

i

- **K.I.S.S.** and **UNIX** philosophies
- Tool aims to be:
 - Modular
 - Minimal
 - Unobtrusive
- Does only what it's meant for: create snapshots of PostgreSQL statistics for visualization and analysis.

Design Philosophy

ii

- Not for live monitoring displays
 - But one could...
- Open schema, clearly defined
 - Data easily exportable
- No built-in scheduler
- No built-in data retention policy mechanism

Design

- Components
 - PostgreSQL extension
 - Python utility for retrieving stored snapshots & creating simple visualizations using **Matplotlib**
- Nothing to put in **shared_preload_libraries**
- No need to restart Postgres

Usage

i

- Extension can be used by superusers, or any user that has **pg_monitor** role privileges
- To take a snapshot, e.g. from **psql**:

```
SELECT  
pgstatviz.snapshot();
```

```
psql (15.1 (Ubuntu 15.1-1.pgdg22.04+1))  
Type "help" for help.  
  
faf=> SELECT pgstatviz.snapshot();  
NOTICE:  created pg_statviz snapshot  
        snapshot  
-----  
2023-04-20 14:15:14.5869+01  
(1 row)  
  
faf=> □
```



```
usage: pg_statviz [--help] [--version] [-d DBNAME] [-h HOSTNAME] [-p PORT] [-U USERNAME]
                [-W] [-D FROM TO] [-O OUTPUTDIR]
                {analyze,buf,cache,checkp,conn,io,lock,tuple,wait,wal,xact} ...
```

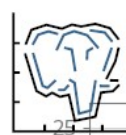
run all analysis modules

positional arguments:

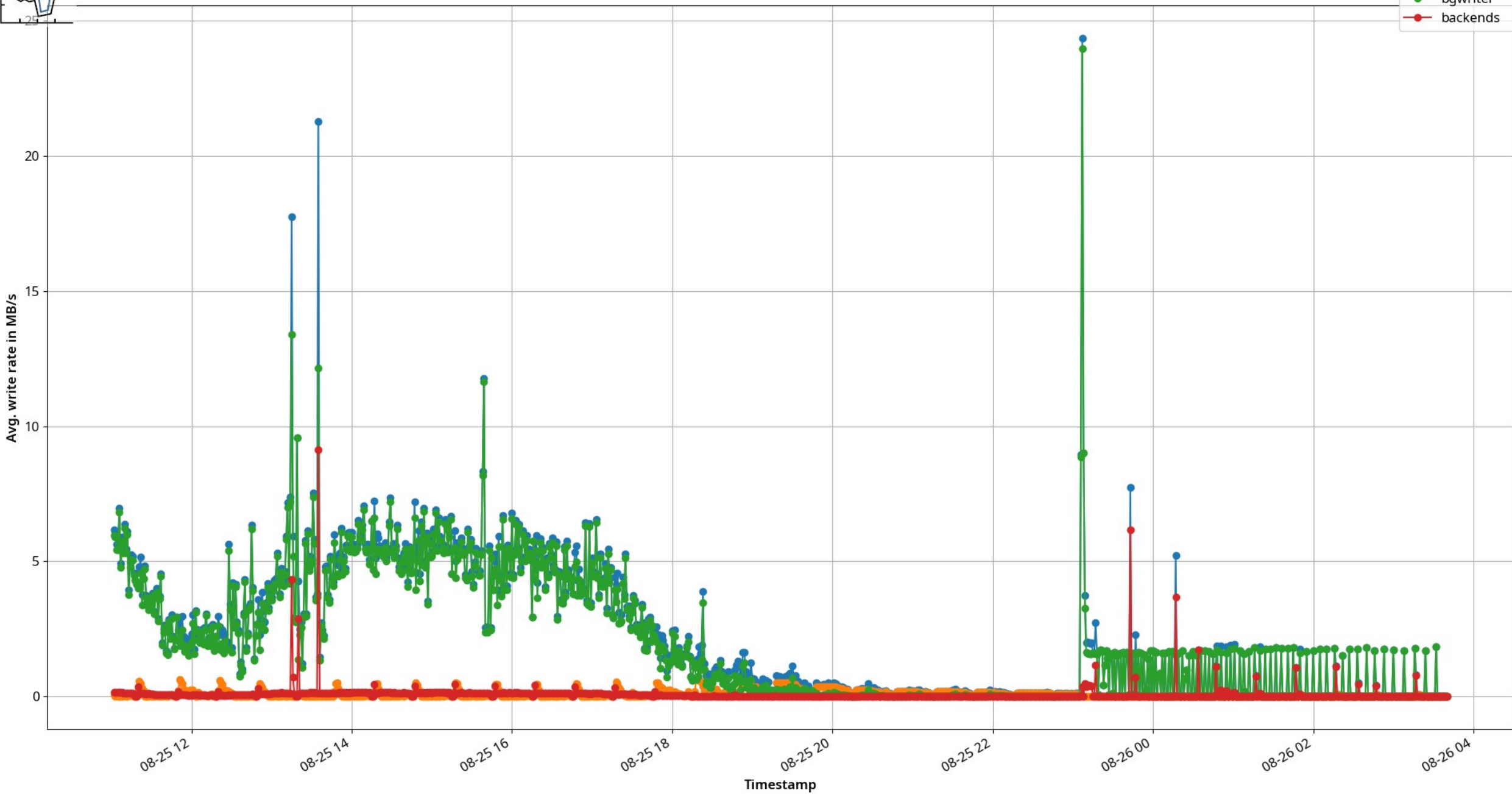
```
{analyze,buf,cache,checkp,conn,io,lock,tuple,wait,wal,xact}
analyze          run all analysis modules
buf              run buffers written analysis module
cache            run cache hit ratio analysis module
checkp           run checkpoint analysis module
conn             run connection count analysis module
io               run I/O analysis module
lock             run locks analysis module
tuple            run tuple count analysis module
wait             run wait events analysis module
wal              run WAL generation analysis module
xact             run transaction count analysis module
```

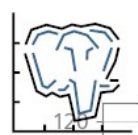
options:

```
--help
--version          show program's version number and exit
-d DBNAME, --dbname DBNAME
                  database name to analyze (default: 'vyruss')
-h HOSTNAME, --host HOSTNAME
                  database server host or socket directory (default: '/var/run/postgresql')
-p PORT, --port PORT database server port (default: '5432')
-U USERNAME, --username USERNAME
                  database user name (default: 'vyruss')
-W, --password    force password prompt (should happen automatically) (default: False)
-D FROM TO, --daterange FROM TO
                  date range to be analyzed in ISO 8601 format e.g. 2023-01-01T00:002023-01-01T23:59 (default:
-O OUTPUTDIR, --outputdir OUTPUTDIR
                  output directory (default: -)
```

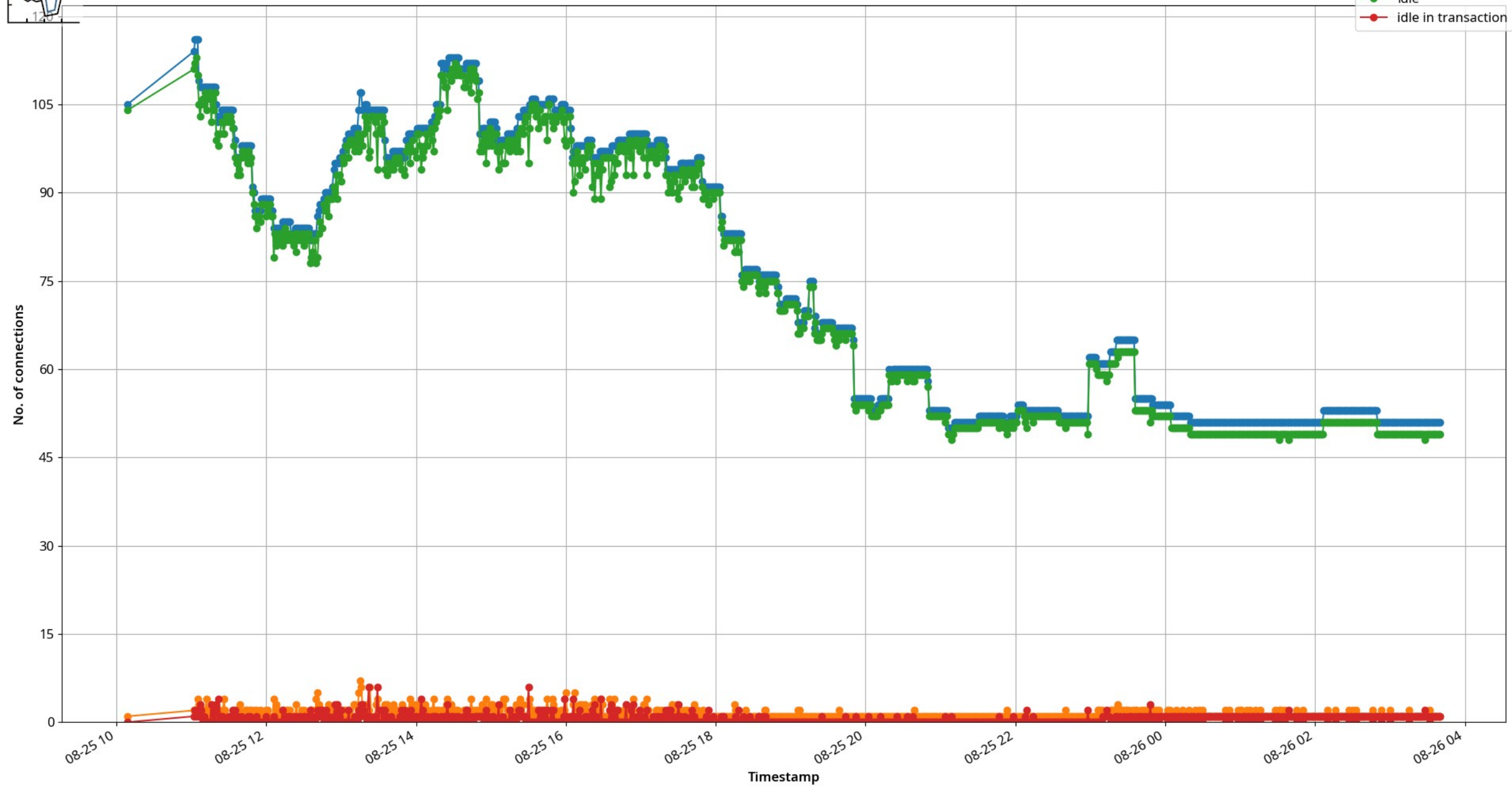
- total
- checkpoints
- bgwriter
- backends

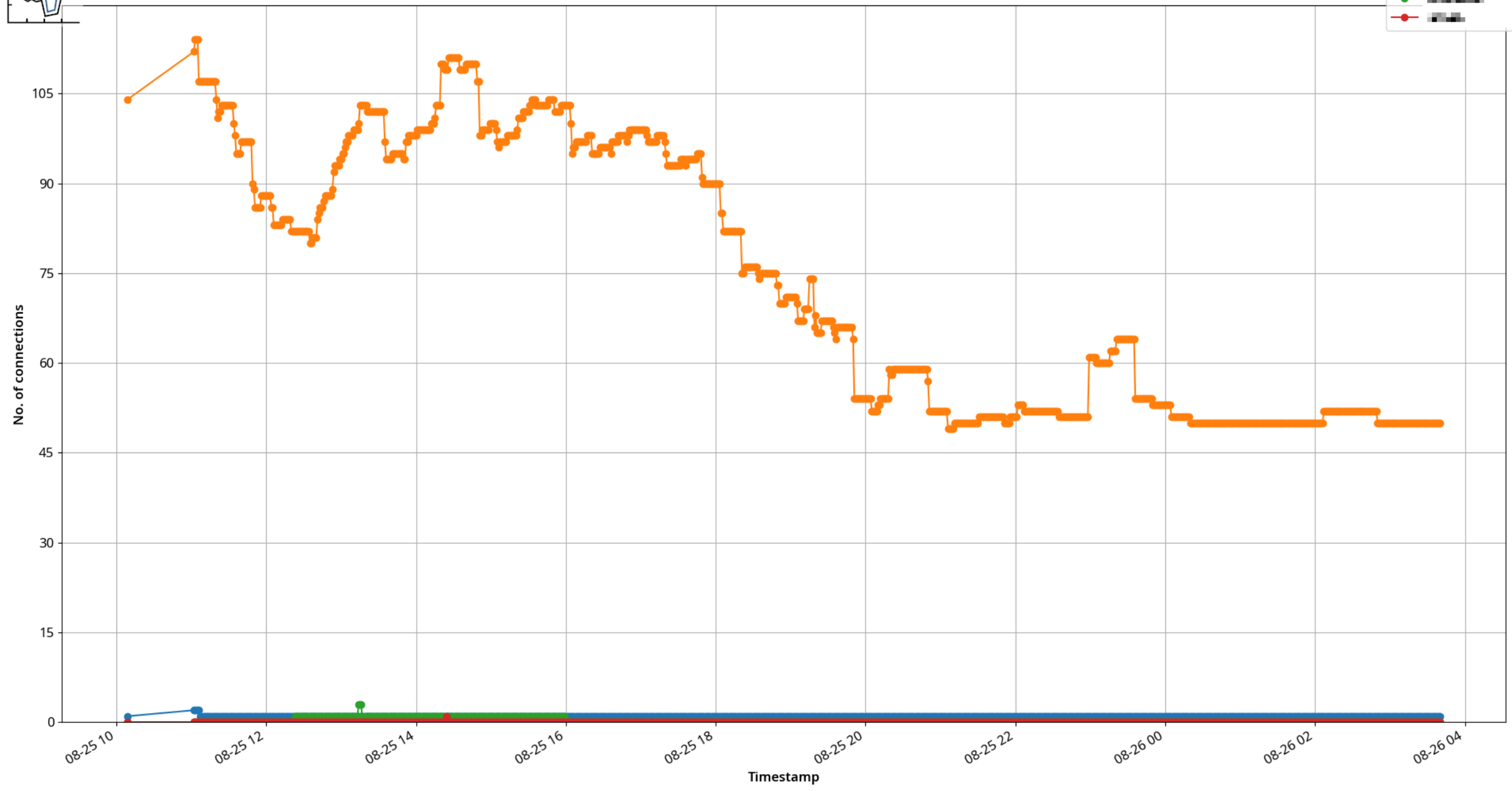
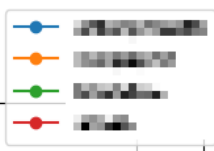
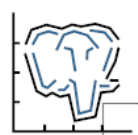


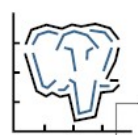


Connection/status count

- total
- active
- idle
- idle in transaction

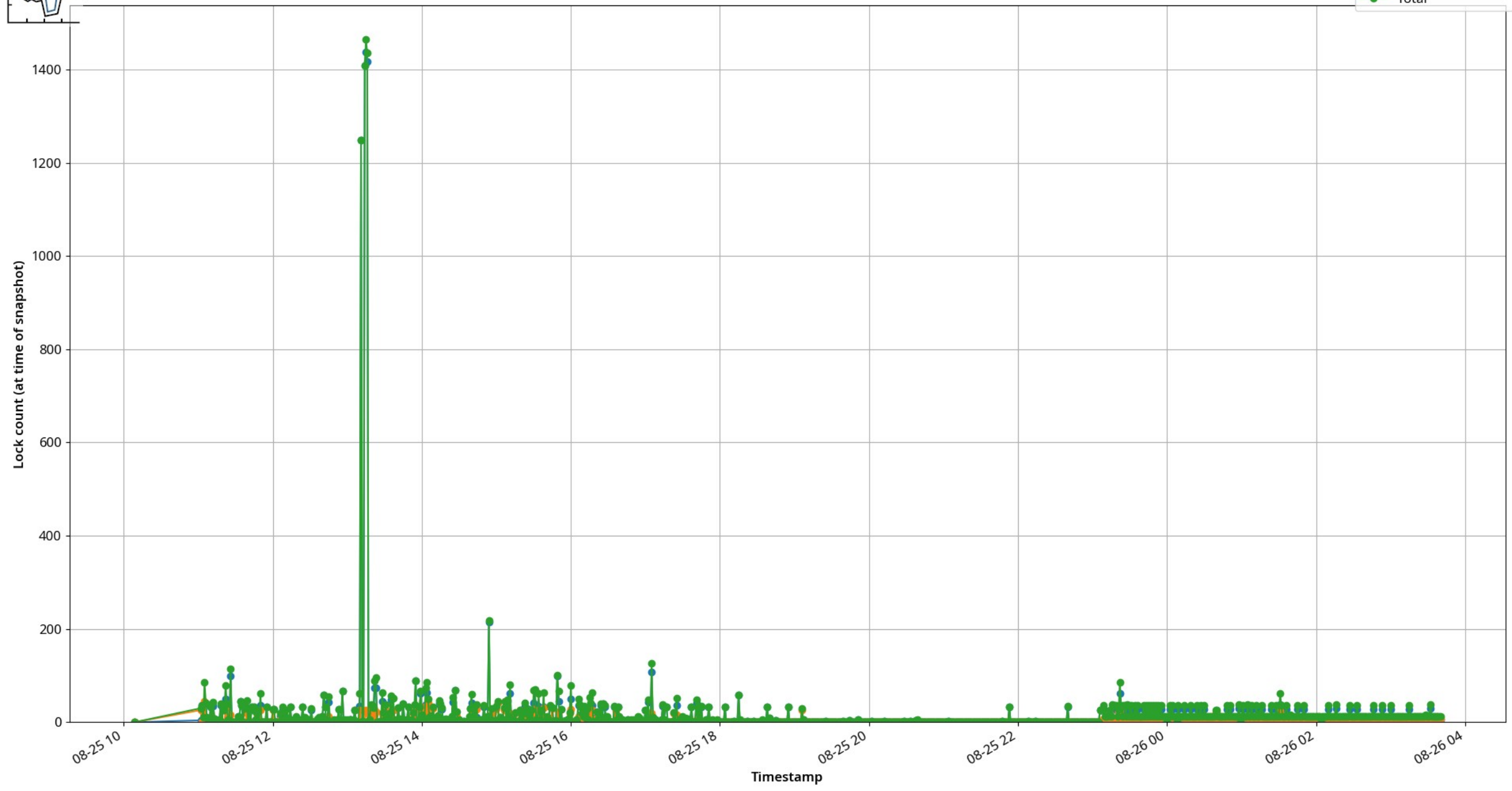


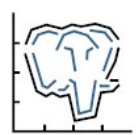




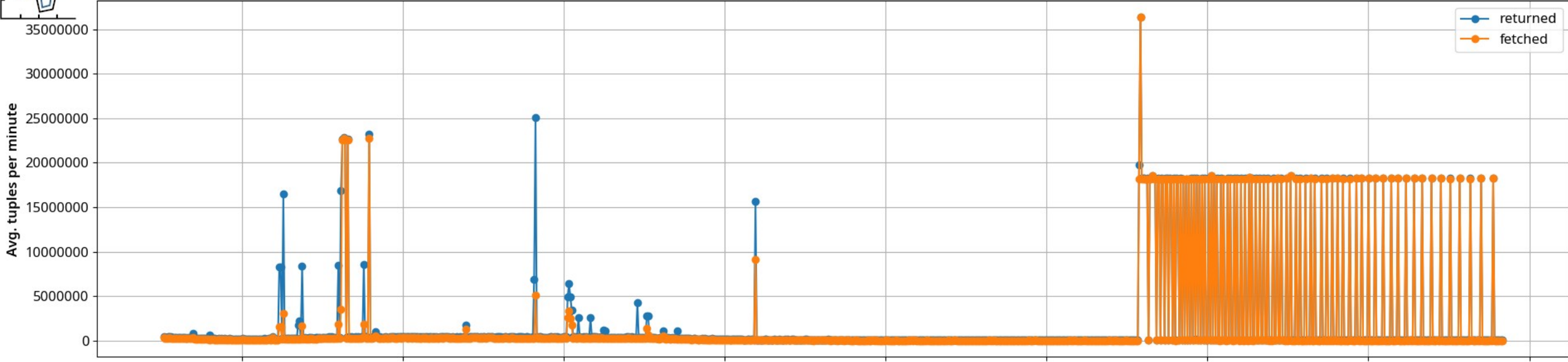
Locks

- AccessShareLock
- RowExclusiveLock
- Total

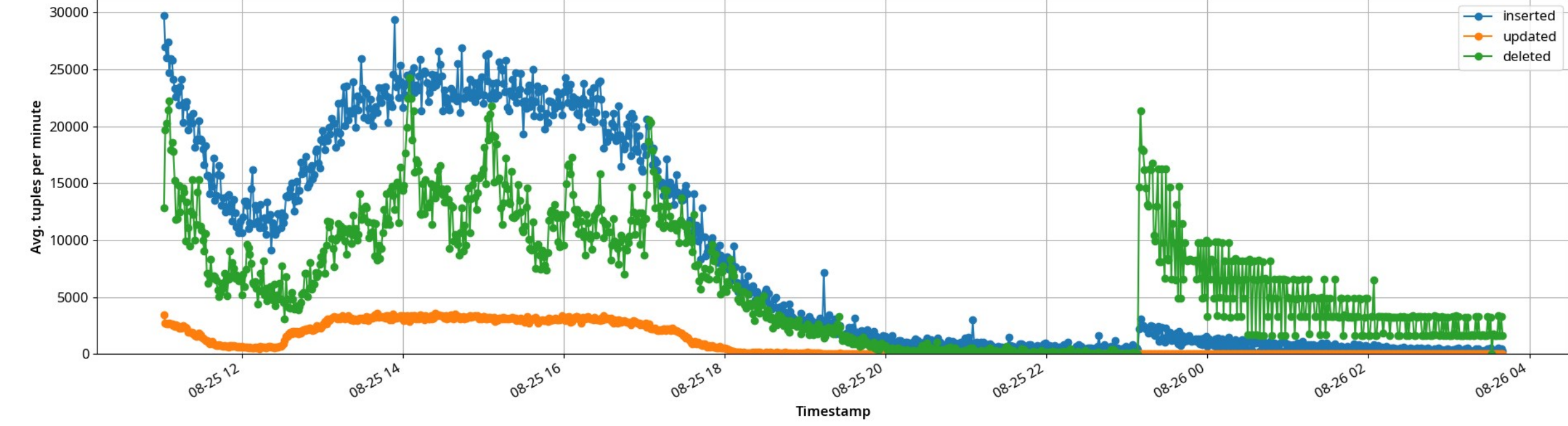


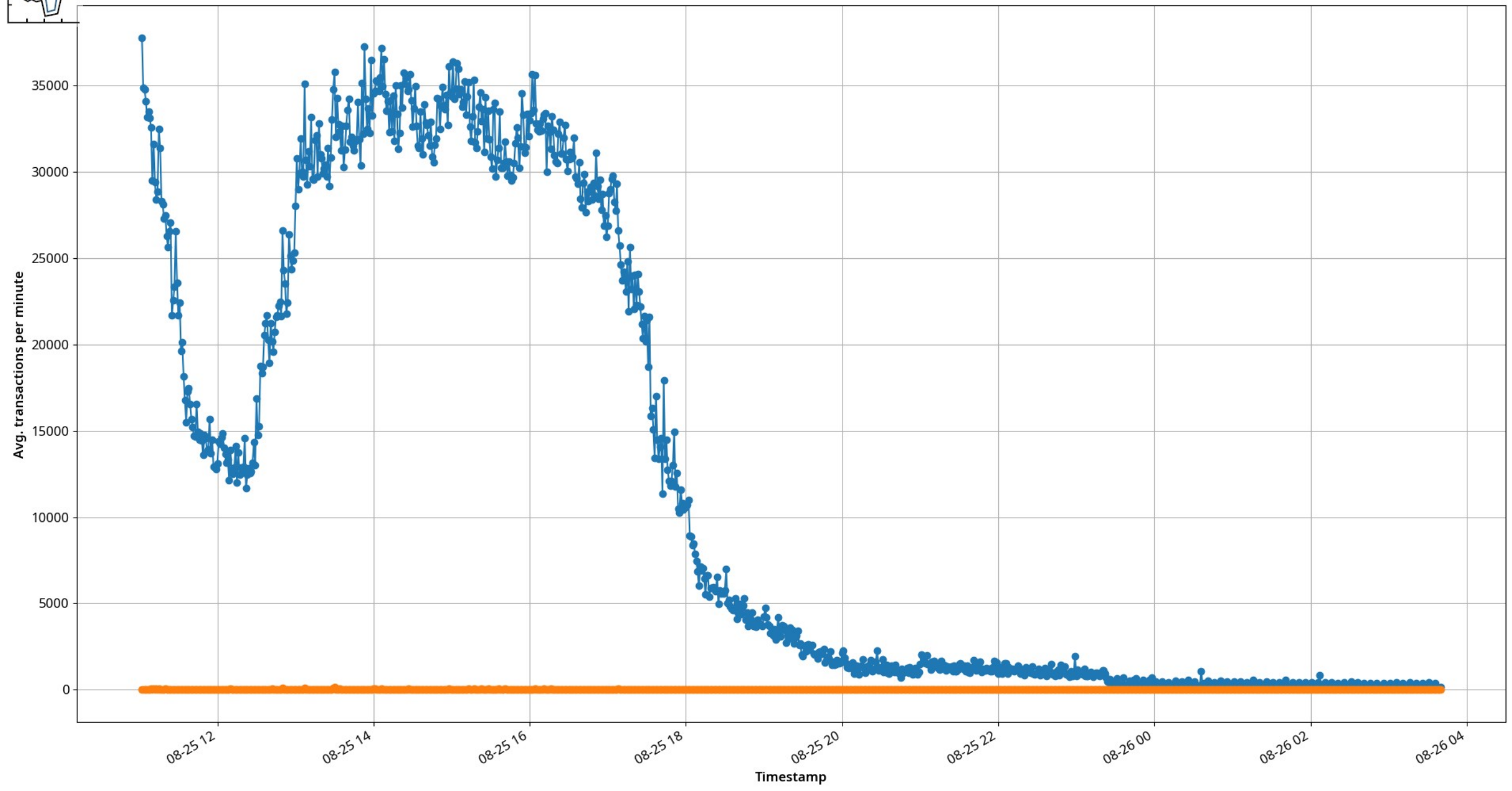


Tuple read rate



Tuple write rate





Use cases

- "Black box" database
 - Deploy and let the developers wreak havoc
 - Identify users/components
- Performance troubleshooting
- Observe and monitor DB behaviour over a long period
 - During a stress test run
 - 8 hours (working hours) / 24 hours (complete day cycle)
 - A month / years (?)

Extension implementation

i

```
faf=> \dt pgstatviz.*  
          List of relations  
 Schema | Name | Type | Owner  
-----+-----+-----+-----  
 pgstatviz | buf | table | postgres  
 pgstatviz | conf | table | postgres  
 pgstatviz | conn | table | postgres  
 pgstatviz | db | table | postgres  
 pgstatviz | io | table | postgres  
 pgstatviz | lock | table | postgres  
 pgstatviz | snapshots | table | postgres  
 pgstatviz | wait | table | postgres  
 pgstatviz | wal | table | postgres  
(9 rows)
```


Extension implementation

ii

```
faf=> \df pgstatviz.*
```

List of functions				
Schema	Name	Result data type	Argument data types	Type
pgstatviz	delete_snapshots	void		func
pgstatviz	snapshot	timestamp with time zone		func
pgstatviz	snapshot_buf	void	snapshot_tstamp timestamp with time zone	func
pgstatviz	snapshot_conf	void	snapshot_tstamp timestamp with time zone	func
pgstatviz	snapshot_conn	void	snapshot_tstamp timestamp with time zone	func
pgstatviz	snapshot_db	void	snapshot_tstamp timestamp with time zone	func
pgstatviz	snapshot_io	void	snapshot_tstamp timestamp with time zone	func
pgstatviz	snapshot_lock	void	snapshot_tstamp timestamp with time zone	func
pgstatviz	snapshot_wait	void	snapshot_tstamp timestamp with time zone	func
pgstatviz	snapshot_wal	void	snapshot_tstamp timestamp with time zone	func

(10 rows)

Utility implementation

i

- Modular code in Python

```
7  __license__ = "PostgreSQL License"
8
9  import getpass
10 from argh.decorators import arg
11 from pg_statviz.modules.buf import buf
12 from pg_statviz.modules.cache import cache
13 from pg_statviz.modules.checkp import checkp
14 from pg_statviz.modules.conn import conn
15 from pg_statviz.modules.io import io
16 from pg_statviz.modules.lock import lock
17 from pg_statviz.modules.tuple import tuple
18 from pg_statviz.modules.wait import wait
19 from pg_statviz.modules.wal import wal
20 from pg_statviz.modules.xact import xact
21 from pg_statviz.libs.dbconn import dbconn
22 from pg_statviz.libs.info import getinfo
23
24
```



Utility implementation

ii

- Plotting

```
142
143     # Plot buffer rates
144     plt, fig = plot.setup()
145     plt.suptitle(f"pg_statviz_{info['hostname']}:{port}",
146                fontweight='semibold')
147     plt.title("Buffer write rate")
148     plt.plot_date(tstamps, total, label="total", aa=True,
149                  linestyle='solid')
150     plt.plot_date(tstamps, checkpoints, label="checkpoints", aa=True,
151                  linestyle='solid')
152     plt.plot_date(tstamps, bgwriter, label="bgwriter", aa=True,
153                  linestyle='solid')
154     plt.plot_date(tstamps, backends, label="backends", aa=True,
155                  linestyle='solid')
156
157     plt.xlabel("Timestamp", fontweight='semibold')
158     plt.ylabel("Avg. write rate in MB/s", fontweight='semibold')
159     fig.legend()
160     fig.tight_layout()
161     outfile = f"{outputdir.rstrip("/") + "/" if outputdir
162               else ''}pg_statviz_{info['hostname']}
163               .replace("/", "-")}_{port}_buf_rate.png"
164     _logger.info(f"Saving {outfile}")
165     plt.savefig(outfile)
166
```


The Future

- Code is currently at "beta / testing" maturity
- Needs:
 - Additional modules for stats to record (such as replication) 
 - More data management/retention functions

Google Summer of Code 👍



Program: 2023

Timeline

Projects

Members

Contributors

statviz



Showing 1 results for the search term "statviz"

All times in Europe/London timezone.

CONTRIBUTOR	PROJECT	ORGANIZATION	ASSIGNED MENTORS	EVALUATIONS	TASK ↑
Rajiv Harlalka	pg_statviz: Time Serie...	PostgreSQL	Jimmy Angelakos, Pavlo (+1)	Passed / Passed	Final Evaluation Ready to view Deadline: Sep 04, 2023 7:00 P

Thank you!

- Project page:

https://github.com/vyruss/pg_statviz

- Download:

- **PostgreSQL YUM & APT repos**

(thanks Devrim, Christoph)

- **PGXN (extension)**

- **PyPI (utility)**