What use are the PostgreSQL activity statistics?

Who's Guillaume Lelarge?

- French translator of the PostgreSQL manual
- Member of pgAdmin's team
- Vice-treasurer of PostgreSQL Europe
- CTO of Dalibo

- Mail: guillaume@lelarge.info
- Twitter: g_lelarge
- Blog: http://blog.guillaume.lelarge.info

Agenda

- Statistics collector process
- How to read the statistics
- What we can do with them
- Tools

Don't be mistaken!

2 kinds of statistics with PostgreSQL

- Data statistics
  - Gathered with ANALYZE statement
  - Used to optimize query execution
  - **We won't talk about that!**
- Activity statistics
  - Gathered by a subprocess
  - Used by the autovacuum... and you!

Stats collector process

```
[guillaume@laptop ~]$ ps xj | grep postgres
/opt/postgresql-9.1/bin/postgres
 \_ postgres: writer process
 \_ postgres: wal writer process
 \_ postgres: autovacuum launcher process
 \_ postgres: stats collector process
```

When PostgreSQL is started

- Creates the UDP socket
- The stats collector is **forked**
- It loads $PGDATA/global/pgstat.stat file in memory
- ... and deletes the file

While PostgreSQL runs

- The stats collector waits for new incoming message
- Every 500ms, it writes the stats in the pgstat.stat file
  - stats_temp_directory: new GUC in 8.4
  - default value: pg_stat_tmp
- It also answers to two signals:
  - SIGHUP: to reload configuration
  - SIGKILL: to quit

When PostgreSQL is stopped

- The stats collector writes the stats in $PGDATA/global/pgstat.stat
- And it deletes the $stats_temp_directory/pgstat.stat file

How do we look at the stats?

- By reading the stats in memory
- ... through stored functions

```
SELECT pg_stat_get_db_xact_commit(oid)
  AS xact_commit
FROM pg_database
WHERE datname='postgres';
 xact_commit
-------------
           4
(1 row)
```

There is a simpler way

- By using the catalog stats views

```
SELECT datname, xact_commit, xact_rollback
FROM pg_stat_database
WHERE datname='postgres';
 datname  | xact_commit | xact_rollback
----------+-------------+---------------
 postgres |          21 |             1
(1 row)
```

- Get the list on psql with \dS

A quick overview

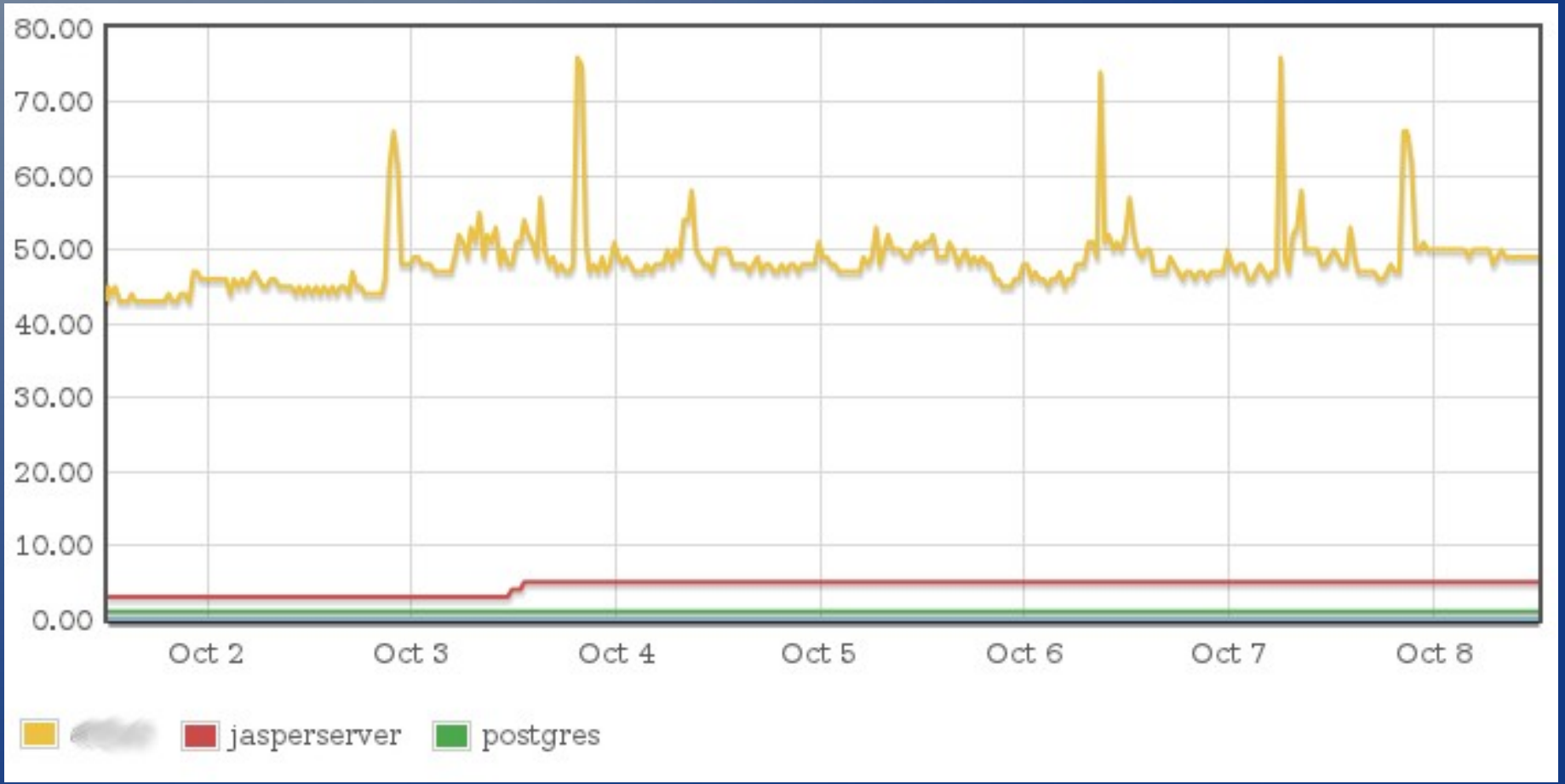
- 2 kinds of statistics views
- Cumulative views
  - Mostly number of scans, rows, blocks, calls, xact
  - On databases, tables, indexes, sequences, functions
- Live views
  - Sessions
  - Locks

Stats - pg_stat_activity

datid	Database ID		
datname	Database name		
procpid	Backend PID		
usesysid	Owner OID		
usename	Owner name		
application_name	Application name	9.0	
client_addr	Client IP address		
client_hostname	Client hostname	9.1	log_hostname
client_port	Client port		
backend_start	Backend start timestamp		
xact_start	Transaction start timestamp	8.3	track_activity
query_start	Query timestamp		track_activity
waiting	Waiting for locks?		
current_query	Current query		track_activity

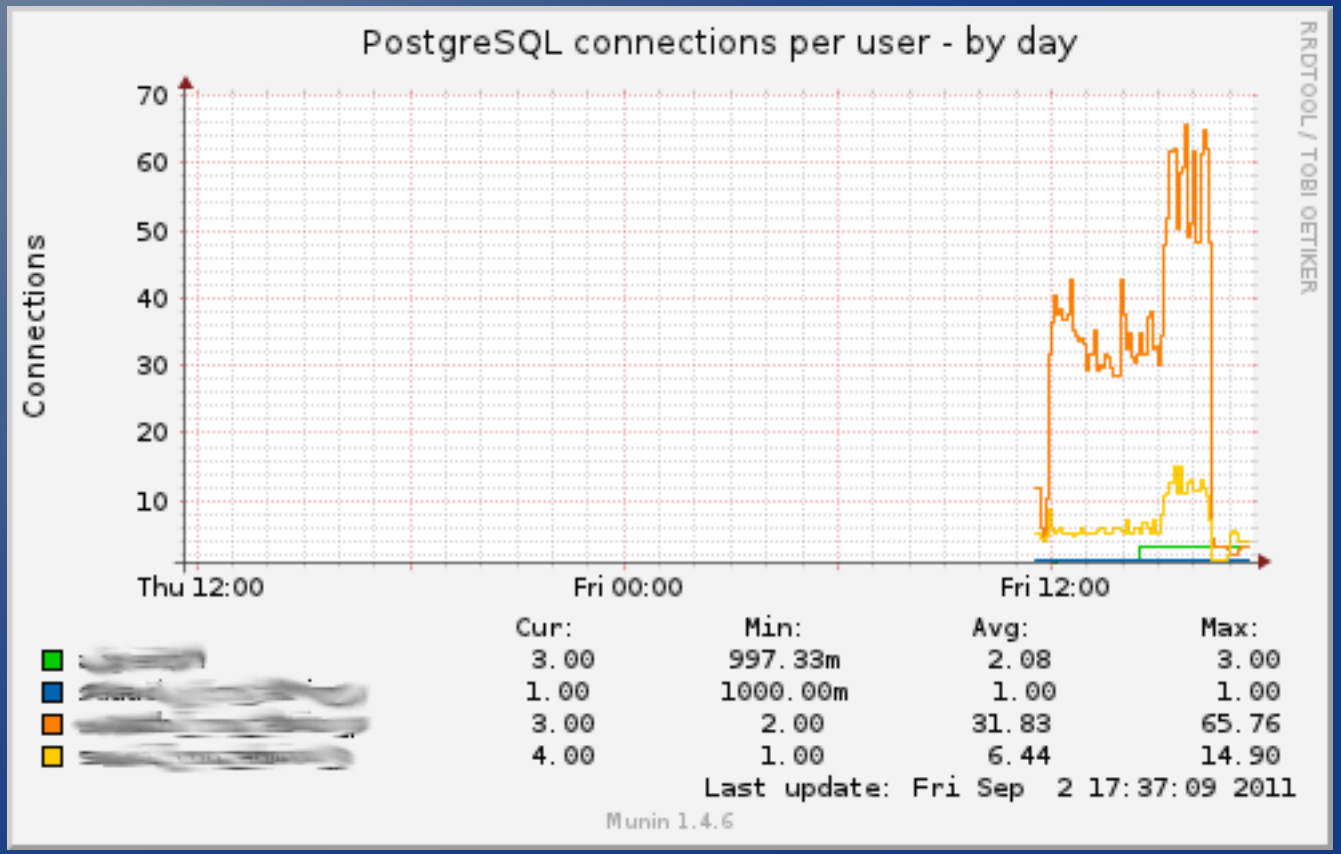
How many connections per database?

```
SELECT datname, numbackends
FROM pg_stat_database
```



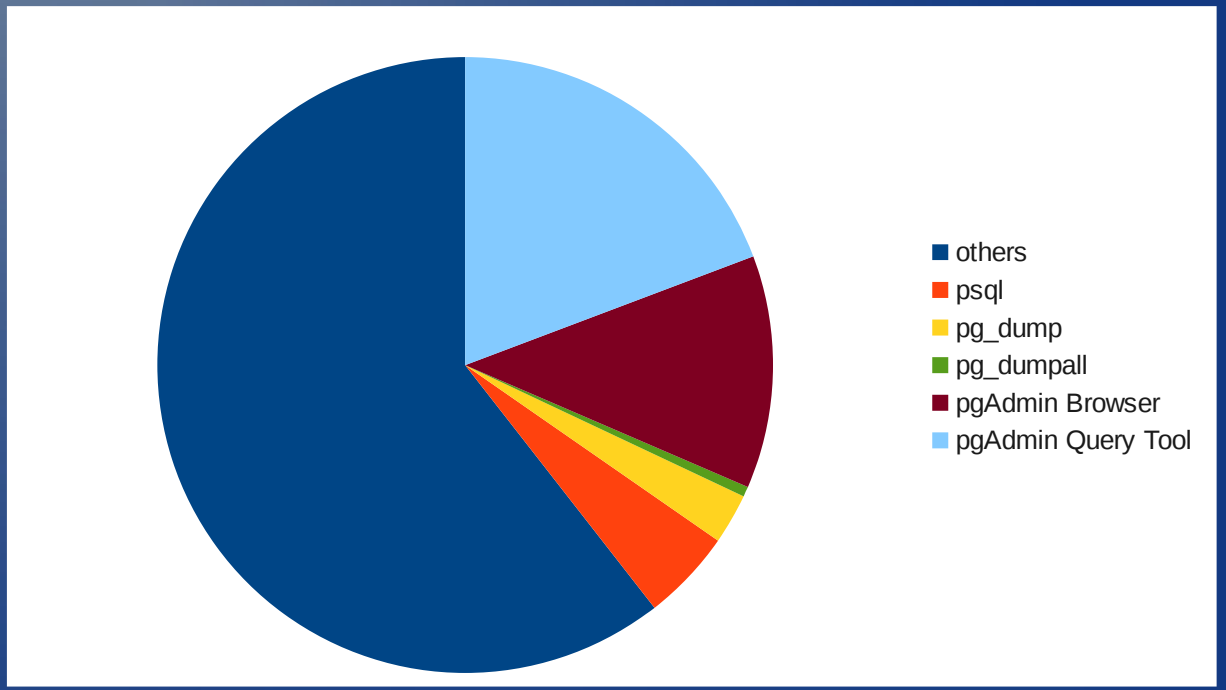
How many connections per user?

```
SELECT usename, count(*)
FROM pg_stat_activity
GROUP BY 1
```



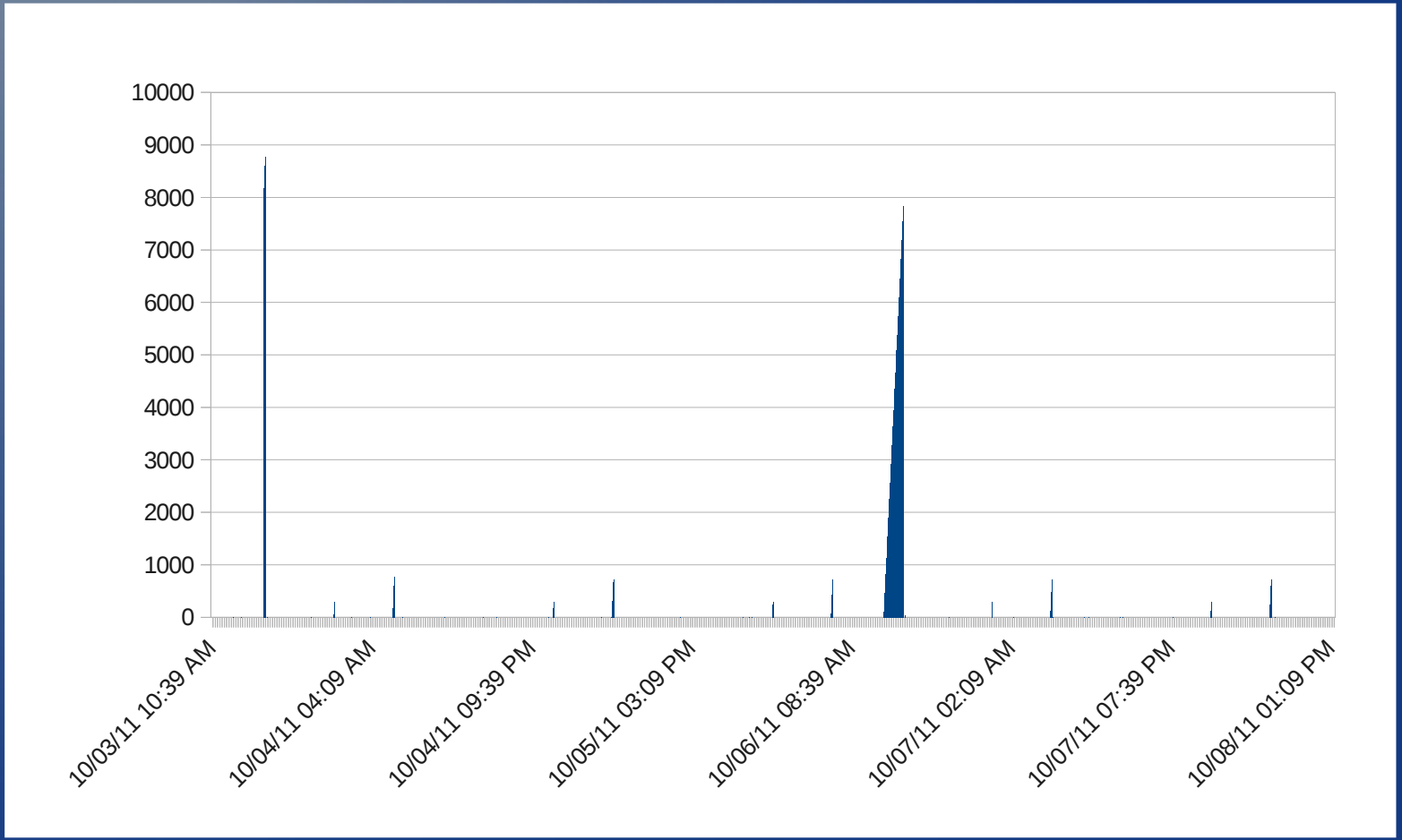
How many connections per apps?

```
SELECT coalesce(application_name, 'others'),
  count(*)
FROM pg_stat_activity
GROUP BY 1
```



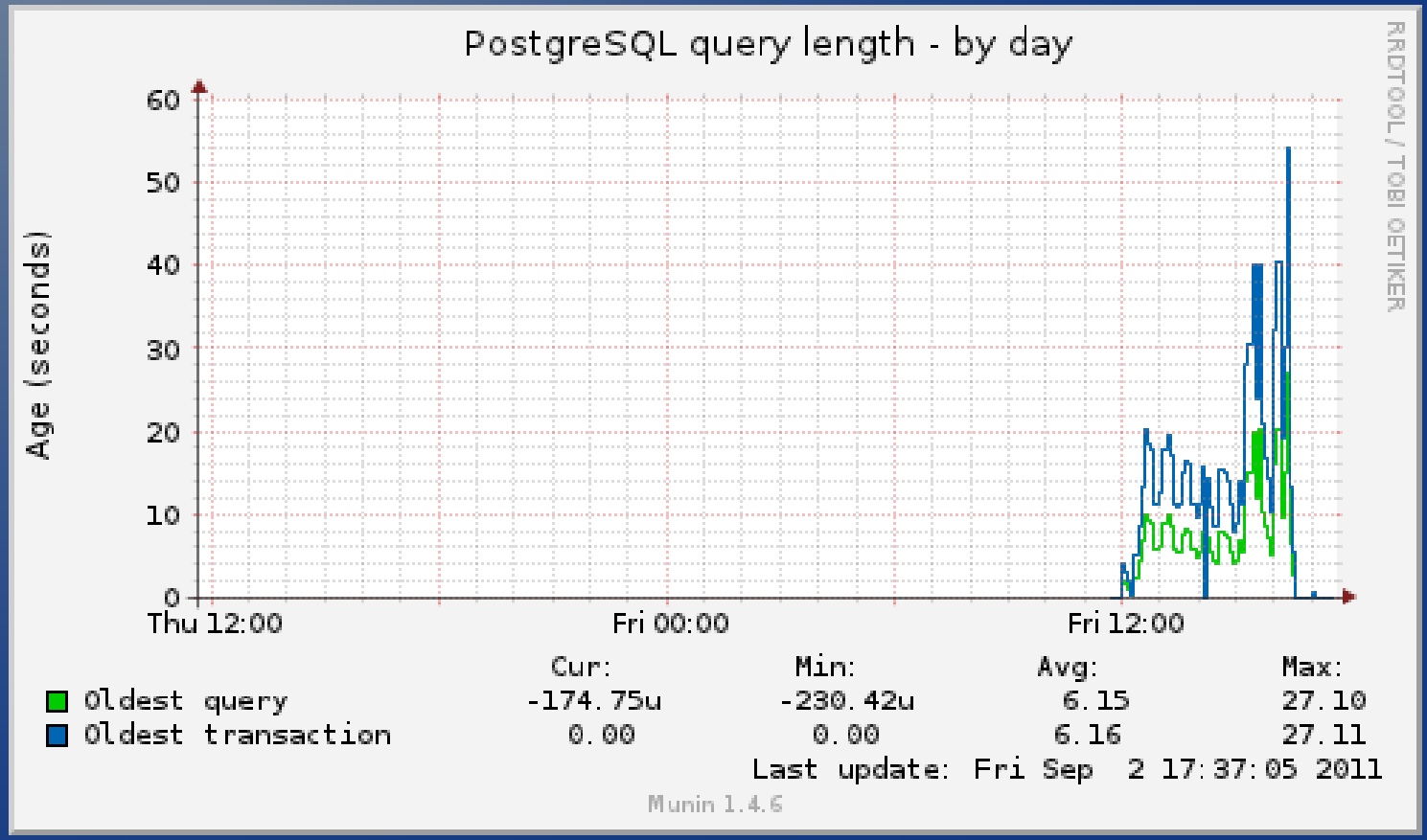
Session duration?

```
SELECT COALESCE(max(extract(
  epoch FROM CURRENT_TIMESTAMP-backend_start)),0)
FROM pg_stat_activity
```



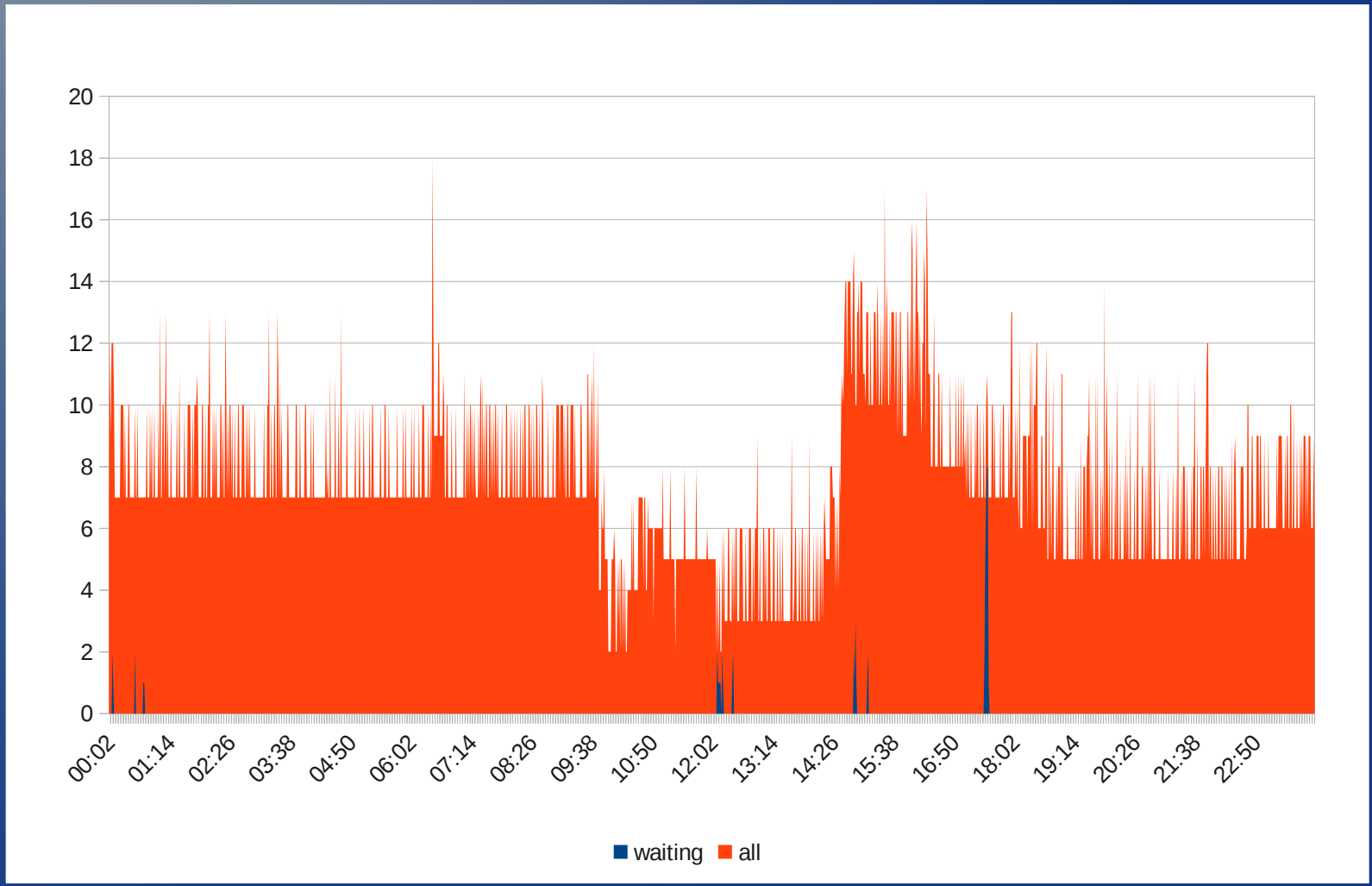
Oldest query/transaction?

```
SELECT COALESCE(max(extract(
  epoch FROM CURRENT_TIMESTAMP-query_start)),0)
FROM pg_stat_activity
WHERE current_query NOT LIKE '<IDLE%'
```



Waiting sessions?

```
SELECT count(*) FROM pg_stat_activity
WHERE waiting IS TRUE
```

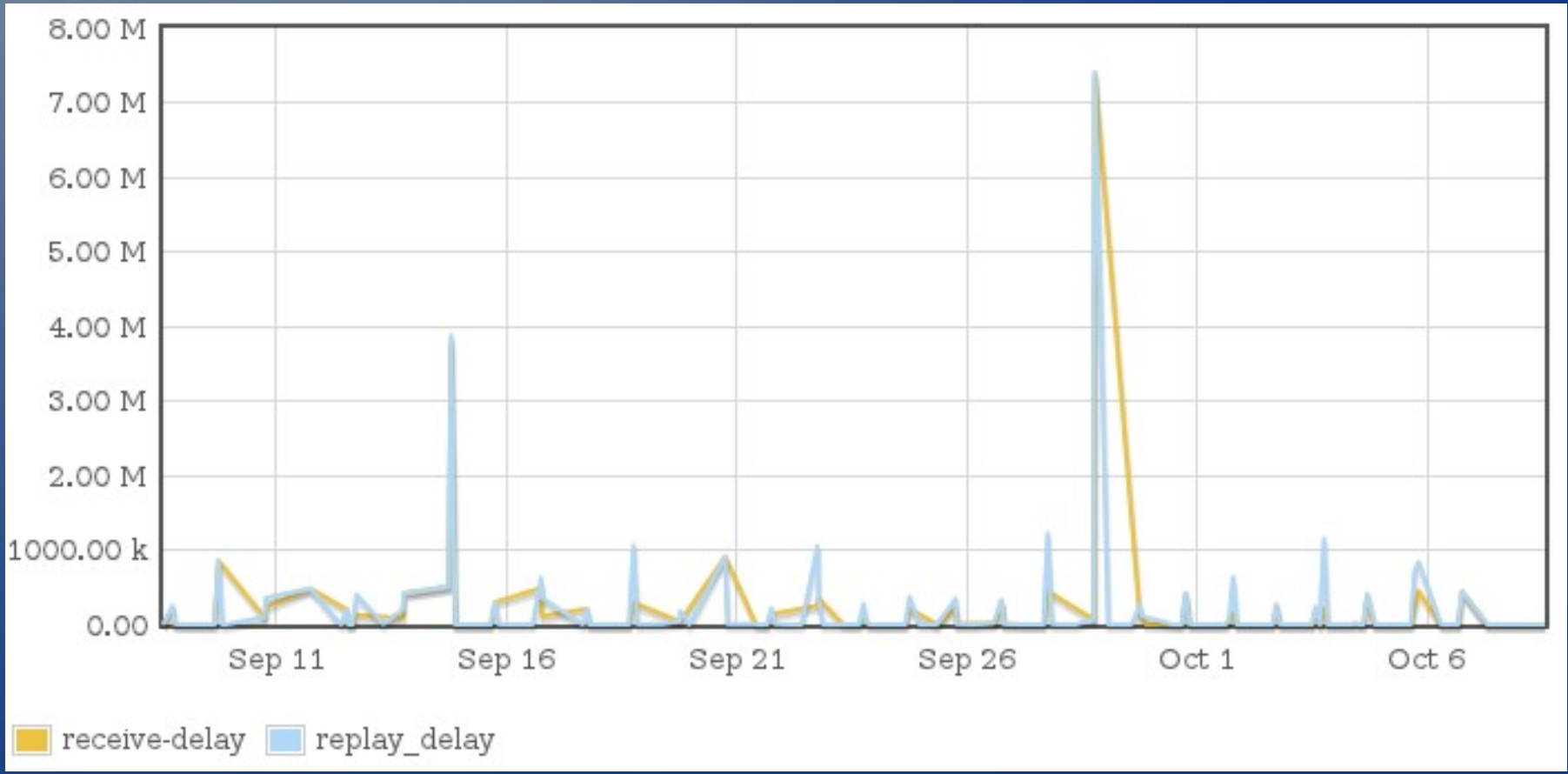


Stats - pg_stat_replication

procpid	Backend PID	
usesysid	Role OID	
usename	Role name	
application_name	Application name	
client_addr	Client IP address	
client_hostname	Client hostname	log_hostname
client_port	Client port	
backend_start	Backend start timestamp	
state	State of walsender	
sent_location	Last sent location	
write_location	Last written location	
flush_location	Last flushed location	
replay_location	Last replayed location	
sync_priority	Priority amongst sync slaves	
sync_state	Synchronous state	

Replication lag?

- lag on write: write_location – sent_location
- lag on replay : replay_sent – sent_location

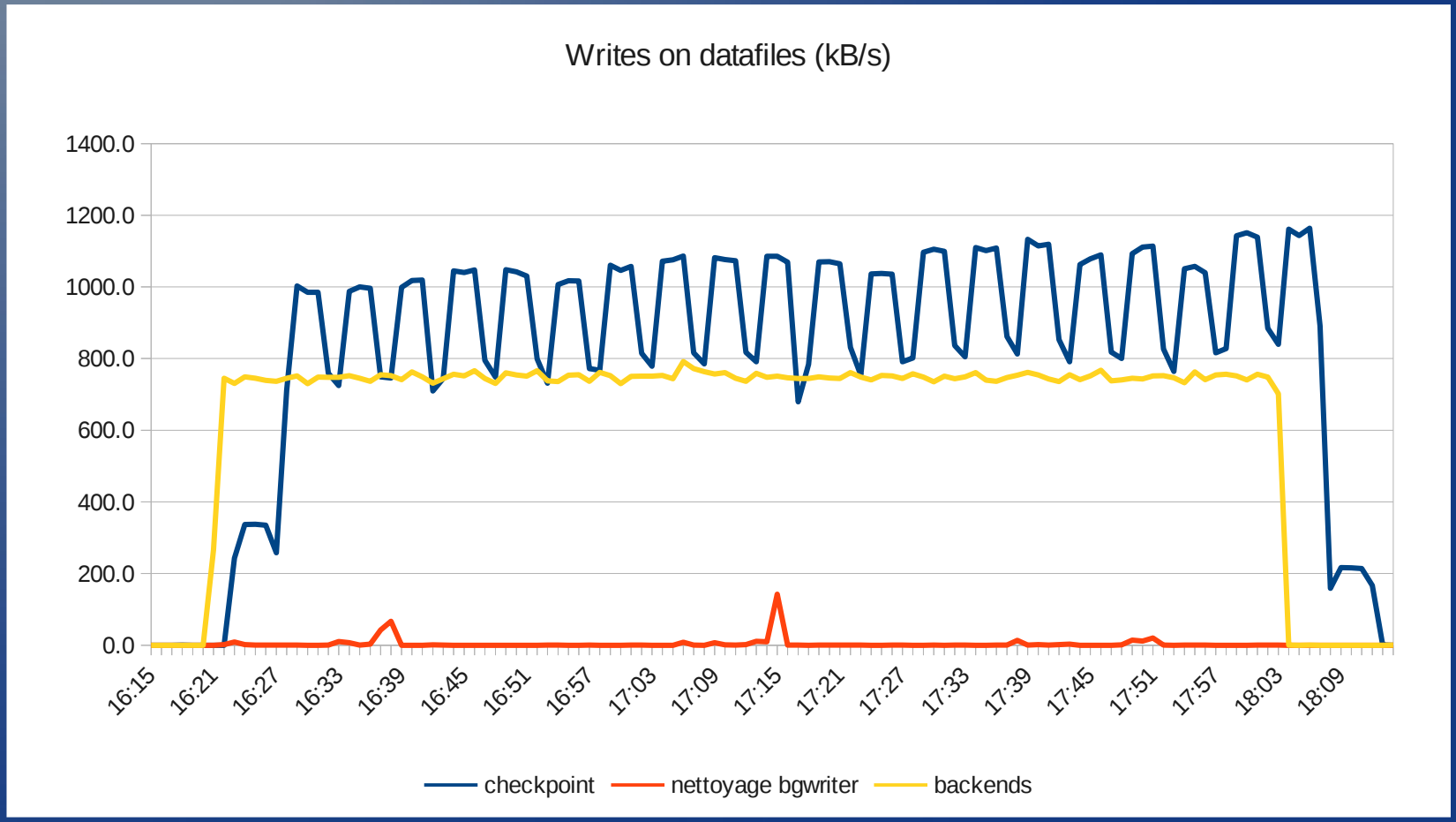


Stats - pg_stat_bgwriter

checkpoints_timed	Scheduled CHECKPOINTs	
checkpoints_req	Requested CHECKPOINTs	
buffers_checkpoint	Buffers written by CHECKPOINT	
buffers_clean	Buffers written by bgwriter	
maxwritten_clean	# of times bgwriter stopped cleaning because it already wrote too many buffers	
buffers_backend	Buffers written by backends	
buffers_backend_fsync	# of fsync done by backends	9.1
buffers_alloc	Buffers allocated	
stats_reset	Reset stats timestamp	9.1

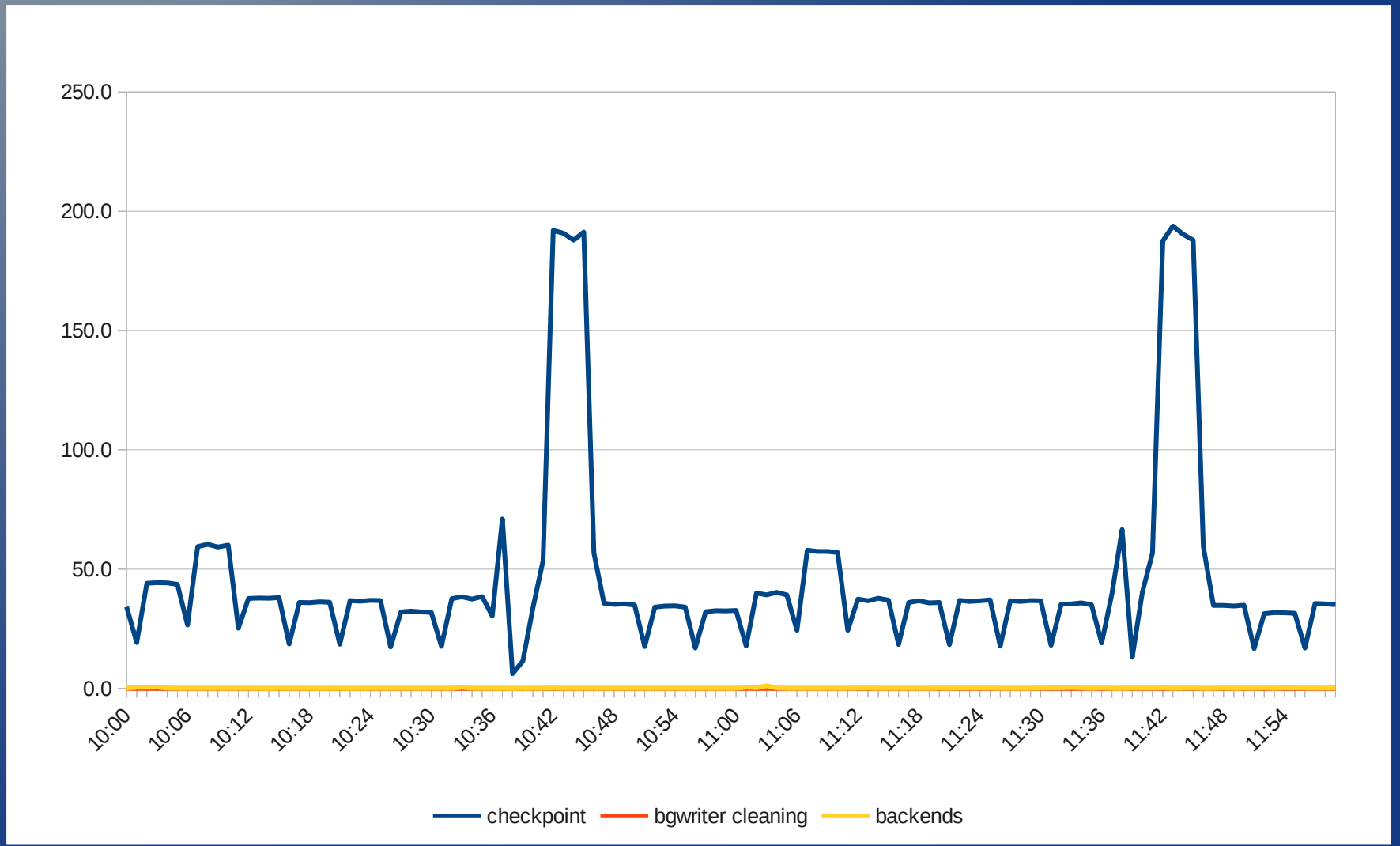
Who does the writes?

```
SELECT buffers_checkpoint, buffers_clean,
  buffers_backend
FROM pg_stat_bgwriter
```



Who does the writes?

- Better behaviour

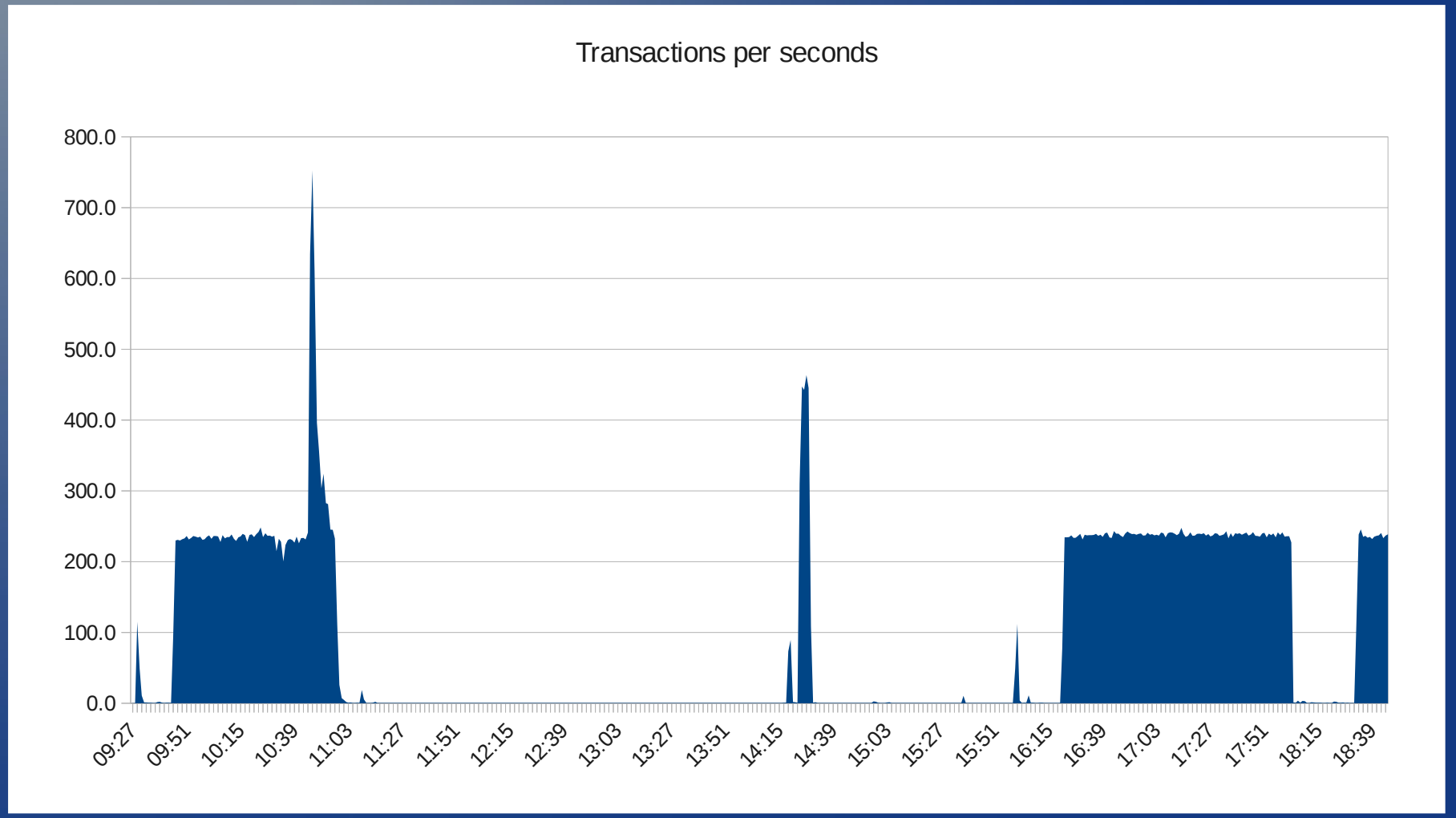


Stats - pg_stat_database

datid	Database OID		
datname	Database name		
numbackends	# of backends		
xact_commit	# of commited xact		track_counts
xact_rollback	# of rollbacked xact		track_counts
blks_read	# of blocks read outside PostgreSQL cache		track_counts
blks_hit	# of blocks read in PostgreSQL cache		track_counts
tup_returned	# of rows returned	8.3	track_counts
tup_fetched	# of rows fetched	8.3	track_counts
tup_inserted	# of rows inserted	8.3	track_counts
tup_updated	# of rows updated	8.3	track_counts
tup_deleted	# of rows deleted	8.3	track_counts
conflicts	# of conflicts	9.1	Standby server only
stats_reset	Reset stats timestamp	9.1	

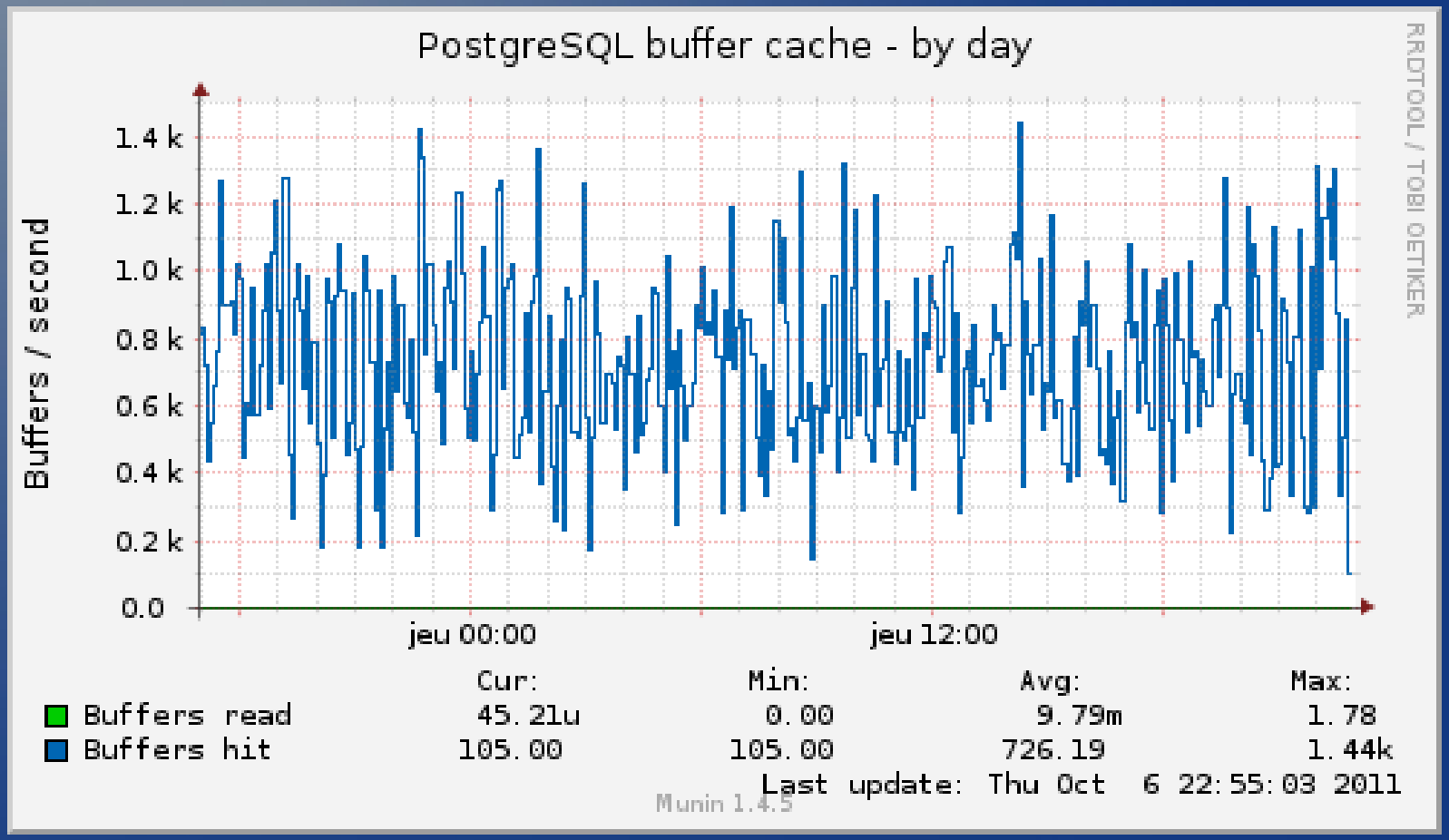
How many xact/sec?

```
SELECT sum(xact_commit+xact_rollback)
FROM pg_stat_database
```



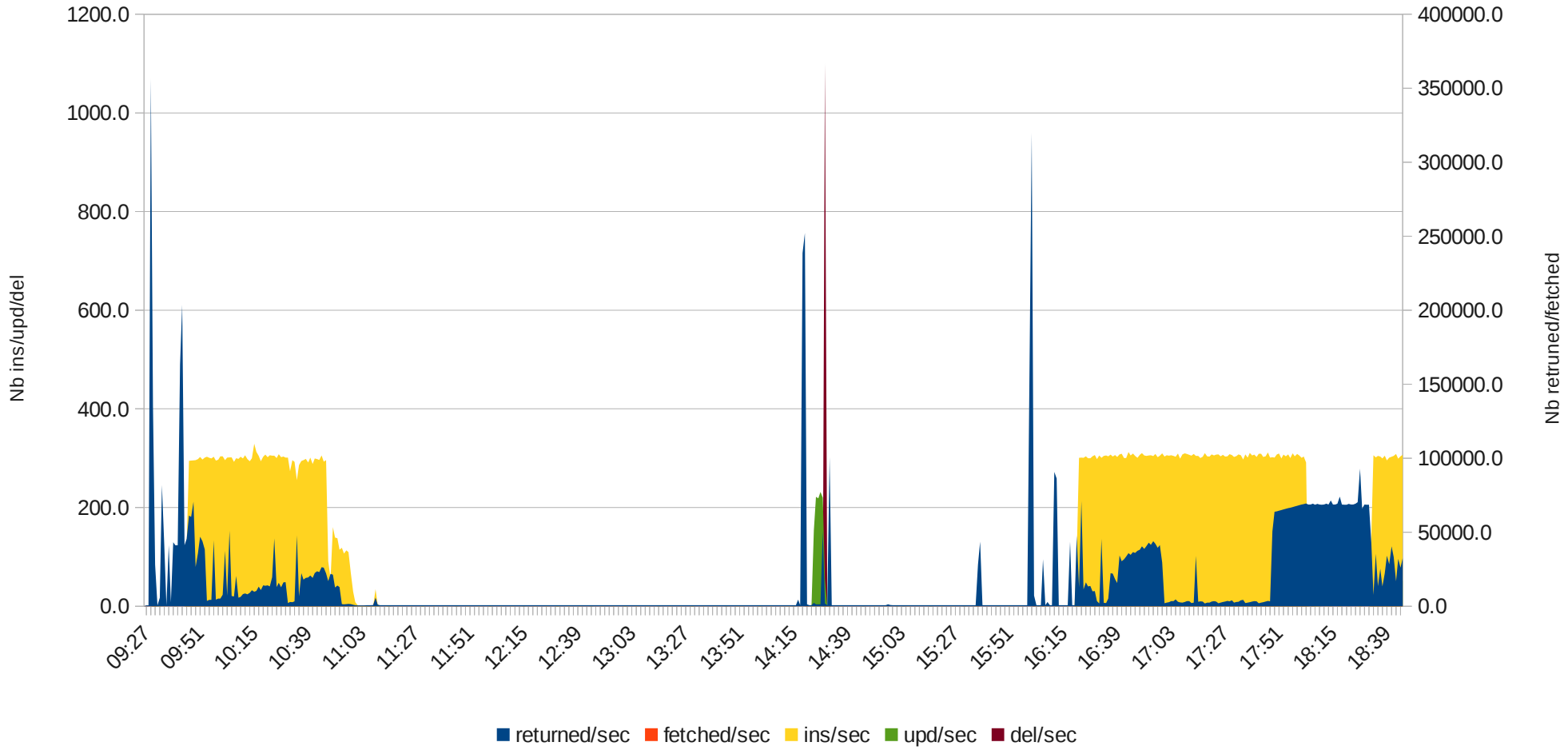
Hit ratio? Cache efficiency?

```
SELECT 100.0*blks_hit/(blks_hit+blks_read)
FROM pg_stat_database
```



Inserts/Updates/Deletes per minute

```
SELECT tup_inserted, tup_updated, tup_deleted
FROM pg_stat_database
```

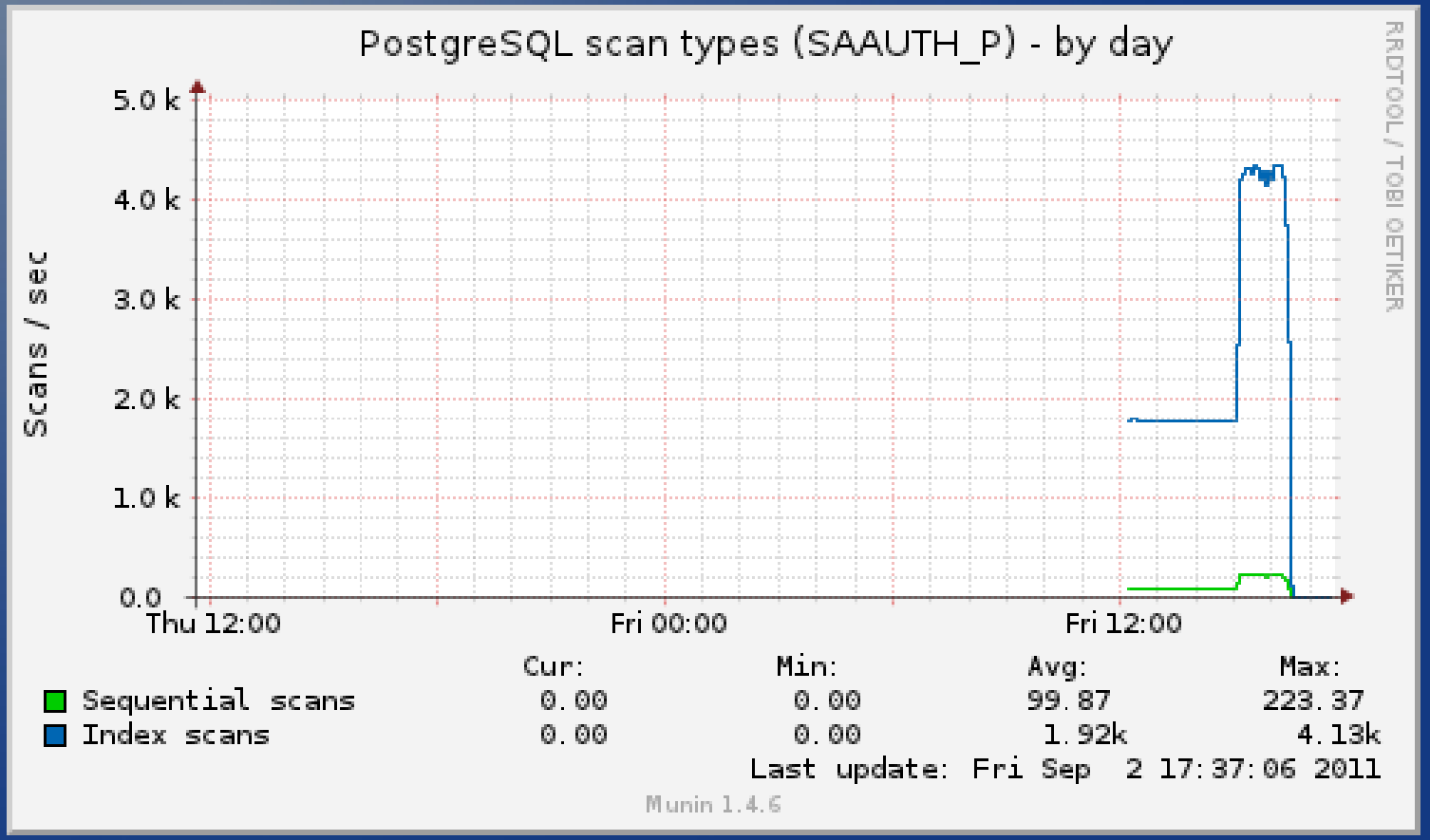


Stats - pg_stat_all_tables

relid	Table OID		
schemaname	Schema name		
relname	Table name		
seq_scan	# of sequential scans		track_counts
seq_tup_read	# of live rows fetched by sequential scans		track_counts
idx_scan	# of index scans		track_counts
idx_tup_fetch	# of rows fetched by index scan		track_counts
n_tup_ins	# of inserted rows		track_counts
n_tup_upd	# of updated rows		track_counts
n_tup_del	# of deleted rows		track_counts
n_tup_hot_upd	# of updated rows with HOT	8.3	track_counts
n_live_tup	# of live tuples	8.3	track_counts
n_dead_tup	# of dead tuples	8.3	track_counts

Seq scan or index scan?

```
SELECT seq_scan, idx_scan
FROM pg_stat_all_tables
```



Stats - pg_stat_all_tables

last_vacuum	Timestamp of last manual VACUUM		track_counts
last_autovacuum	Timestamp of last automatic VACUUM		track_counts
last_analyze	Timestamp of last manual ANALYZE		track_counts
last_autoanalyze	Timestamp of last automatic ANALYZE		track_counts
vacuum_count	# of manual VACUUMs	9.1	track_counts
autovacuum_count	# of automatic VACUUMs	9.1	track_counts
analyze_count	# of manual ANALYZEs	9.1	track_counts
autoanalyze_count	# of automatic ANALYZEs	9.1	track_counts

Stats - pg_statio_all_tables

relid	Table OID	
schemaname	Schema name	
relname	Table name	
heap_blks_read	# of blocks read in HEAP outside of PostgreSQL cache	track_counts
heap_blks_hit	# of blocks read in HEAP in PostgreSQL cache	track_counts
idx_blks_read	Same for index	track_counts
idx_blks_hit	Same for index	track_counts
toast_blks_read	# of blocks read in TOAST outside of PostgreSQL cache	track_counts
toast_blks_hit	# of blocks read in TOAST in PostgreSQL cache	track_counts
tidx_blks_read	Same for index	track_counts
tidx_blks_hit	Same for index	track_counts

Stats - pg_stat_all_indexes

relid	Table OID	
indexrelid	Index OID	
schemaname	Schema name	
relname	Table name	
indexrelname	Index name	
idx_scan	# of index scans	track_counts
idx_tup_read	# of entries returned by index scan	track_counts
idx_tup_fetch	# of live rows returned by index scan	track_counts

Stats - pg_statio_all_indexes

relid	Table OID	
indexrelid	Index OID	
schemaname	Schema name	
relname	Table name	
indexrelname	Index name	
idx_blks_read	# of blocks read outside of PostgreSQL cache	track_counts
idx_blks_hit	# of blocks read in PostgreSQL cache	track_counts

Stats - pg_statio_all_sequences

relid	Sequence OID	
schemaname	Schema name	
relname	Sequence name	
blks_read	# of blocks read outside of PostgreSQL cache	track_counts
blks_hit	# of blocks read in PostgreSQL cache	track_counts

Stats - pg_stat_user_functions

funcid	Function OID	
schemaname	Schema name	
funcname	Function name	
calls	# of calls	
total_time	Total duration time	
self_time	Self duration time (ie, without other functions total_time)	

Wait, there are more

- pg_stat_database_conflicts (new in 9.1)
- pg_statio_all_tables
- pg_stat_all_indexes
- pg_statio_all_indexes
- pg_statio_all_sequences
- pg_stat_user_functions

Other interesting math

- seq_tup_read/seq_scan
  - avg # of rows per seq scan
- idx_tup_fetch/idx_tup_read
  - efficiency of the index
- total_time/calls
  - avg execution time per functions

What should we do with these?

- Grab them from time to time
- It's as easy as executing every minute
  - SELECT * FROM <your_stats_view>
- And put the results in a file
- Afterwards, it'll help answer some questions

Tools

- pgstatpack
- pgstats
- Munin
- check_postgres.pl
- pgAdmin, phpPgAdmin

Tools - pgstatpack

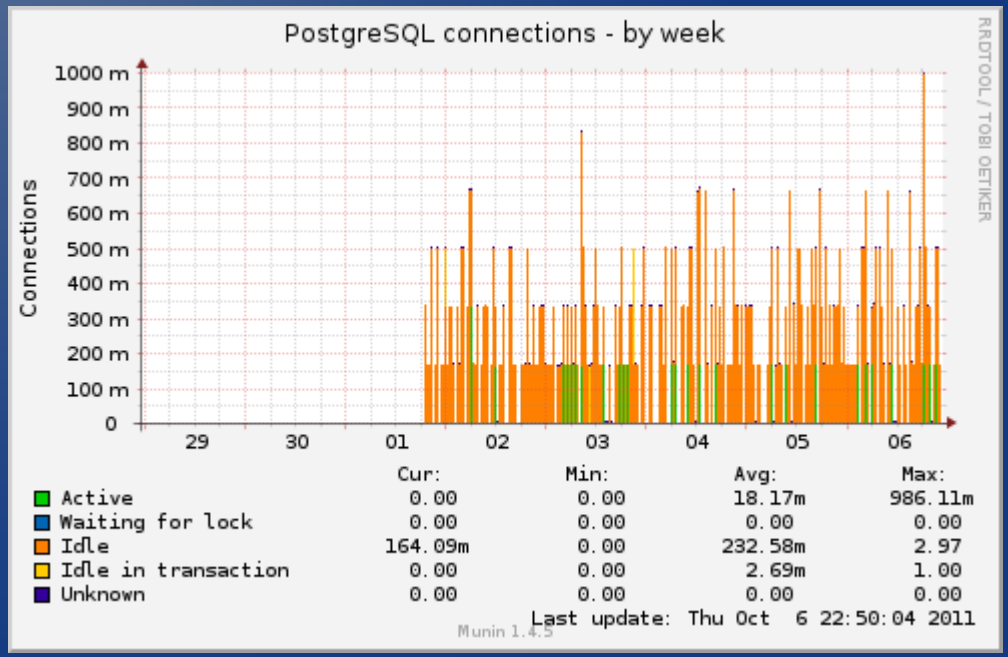
- Records statistics snapshots
- Build a nice report between two snapshots

Tools - pgstats

- Stores statistics in CSV files
- Use cron to do it from time to time
- But then, you're on your own to use the statistics
  - Open the CSV files with LibreOffice or Excel
  - Load the CSV files' contents in a PostgreSQL table

Munin

- Simple to install and configure
- Don't consume much CPU
- Builds HTML and PNG files
- Easy to add more checks

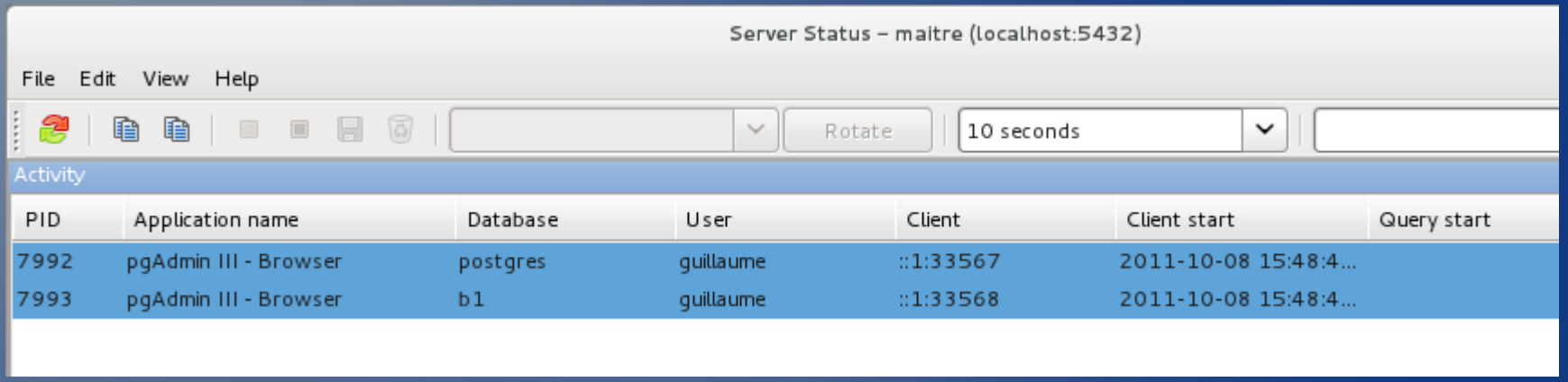


Tools - check_postgres.pl

- Targets Nagios and MRTG
- Mostly for alerts
- But can help with graph based on the perfdata
  - PNP4Nagios
  - Yang
- Perfdatas are not always as useful as they could be

Tools – pgAdmin, and ppa

- Server status in pgAdmin



- And its phpPgAdmin equivalent

Conclusion

- Main entry point for monitoring system
- Find the right tools for you
- Or code it!