Rundum glücklich Postgres Monitoring

PGConf.de 2013

Michael Renner

@terrorobe

Fragen?
Stellen!

Warum Monitoring?

Wer hatte schon mal ein Datenbankproblem?

Gründe zum Monitoren

- Wissen, dass
  - ...alles gut ist
  - ...alles gut war
  - ...alles gut sein wird

Von Zugsunglücken...

```
ERROR:  could not access status of transaction
500185903

DETAIL:  Could not open file "pg_clog/01DD": No
such file or directory.
```

```
Filesystem                    Size  Used Avail Use% Mounted on
/dev/sda3                      29G   29G   0G  100% /var/lib/postgresql
```

Drehender Rost und andere Speicherformen

Replikation

Von Queries

```
SELECT * FROM relativ_neues_feature
  WHERE userid = 42;
```

```
SELECT * FROM buchungen
  WHERE userid = 42
  ORDER BY buchungsdatum DESC
  LIMIT 20;
```

Entwickler mit Messy-Syndrom

Was monitoren?

Betriebssystem

- CPU-Zeiten
- RAM-Verwendung
- Disk IOs
- Freier Speicherplatz

Postgres Bordstatistiken

Queries

- pg_stat_statements (contrib, 9.2+)
- pg_stat_plans (extern)

Benutzer & Programme

- `pg_stat_activity`
  - Verbundene Clients
  - Laufende Queries
  - Offene Transaktionen

Tables & Indices

- `pg_stat_user_tables`
- `pg_statio_user_tables`
- `pg_stat_user_indexes`
- `pg_statio_user_indexes`

Unbenutzter Speicher

```
SELECT
  current_database() AS db, schemaname, tablename, reltuples::bigint AS tups, relpages::bigint AS pages, otta,
  ROUND(CASE WHEN otta=0 OR sml.relpages=0 OR sml.relpages=otta THEN 0.0 ELSE sml.relpages/otta::numeric END,1) AS tbloa
  CASE WHEN relpages < otta THEN 0 ELSE relpages::bigint - otta END AS wastedpages,
  CASE WHEN relpages < otta THEN 0 ELSE bs*(sml.relpages-otta)::bigint END AS wastedbytes,
  CASE WHEN relpages < otta THEN '0 bytes'::text ELSE (bs*(relpages-otta))::bigint || ' bytes' END AS wastedsize,
  iname, ituples::bigint AS itups, ipages::bigint AS ipages, iotta,
  ROUND(CASE WHEN iotta=0 OR ipages=0 OR ipages=iotta THEN 0.0 ELSE ipages/iotta::numeric END,1) AS ibloat,
  CASE WHEN ipages < iotta THEN 0 ELSE ipages::bigint - iotta END AS wastedipages,
  CASE WHEN ipages < iotta THEN 0 ELSE bs*(ipages-iotta) END AS wastedibytes,
  CASE WHEN ipages < iotta THEN '0 bytes' ELSE (bs*(ipages-iotta))::bigint || ' bytes' END AS wastedisize,
  CASE WHEN relpages < otta THEN
    CASE WHEN ipages < iotta THEN 0 ELSE bs*(ipages-iotta::bigint) END
    ELSE CASE WHEN ipages < iotta THEN bs*(relpages-otta::bigint)
      ELSE bs*(relpages-otta::bigint + ipages-iotta::bigint) END
  END AS totalwastedbytes
FROM (
  SELECT
    nn.nspname AS schemaname,
    cc.relname AS tablename,
    COALESCE(cc.reltuples,0) AS reltuples,
    COALESCE(cc.relpages,0) AS relpages,
    COALESCE(bs,0) AS bs,
    COALESCE(CEIL((cc.reltuples*((datahdr+ma-
      (CASE WHEN datahdr%ma=0 THEN ma ELSE datahdr%ma END))+nullhdr2+4))/(bs-20::float)),0) AS otta,
    COALESCE(c2.relname,'?') AS iname, COALESCE(c2.reltuples,0) AS ituples, COALESCE(c2.relpages,0) AS ipages,
    COALESCE(CEIL((c2.reltuples*(datahdr-12))/(bs-20::float)),0) AS iotta -- very rough approximation, assumes all cols
  FROM
     pg_class cc
  JOIN pg_namespace nn ON cc.relnamespace = nn.oid AND nn.nspname <> 'information_schema'
  LEFT JOIN
  (
    SELECT
      ma,bs,foo.nspname,foo.relname,
      (datawidth+(hdr+ma-(case when hdr%ma=0 THEN ma ELSE hdr%ma END)))::numeric AS datahdr,
      (maxfracsum*(nullhdr+ma-(case when nullhdr%ma=0 THEN ma ELSE nullhdr%ma END))) AS nullhdr2
    FROM (
      SELECT
```

Unbenutzter Speicher

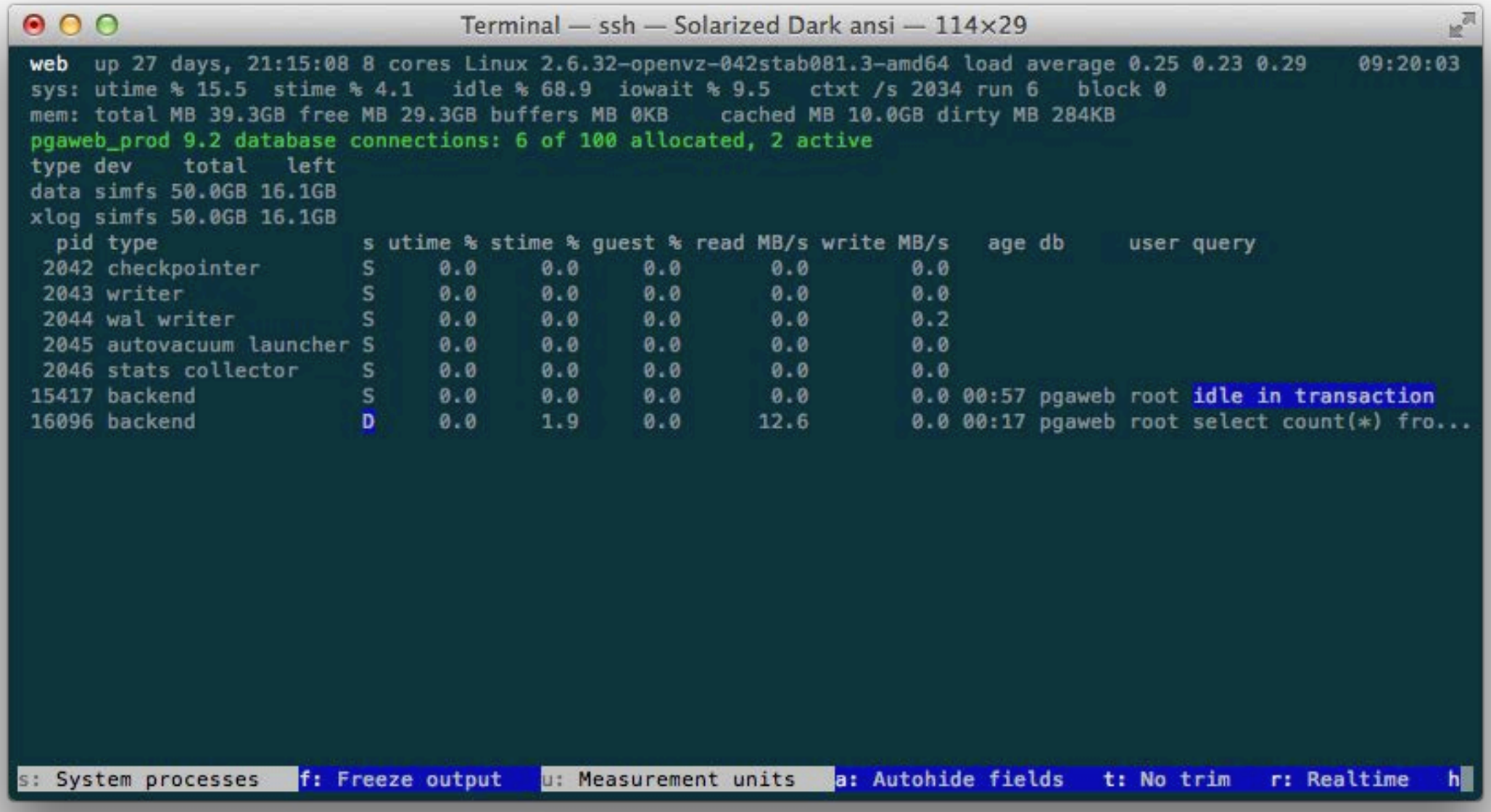
- oder `pgstattuple (contrib)`

Obskureres

- `pg_stat_database`
- `pg_stat_bgwriter`
- `pg_locks`
- `pg_settings (oder: SHOW ALL)`

Womit monitoren?

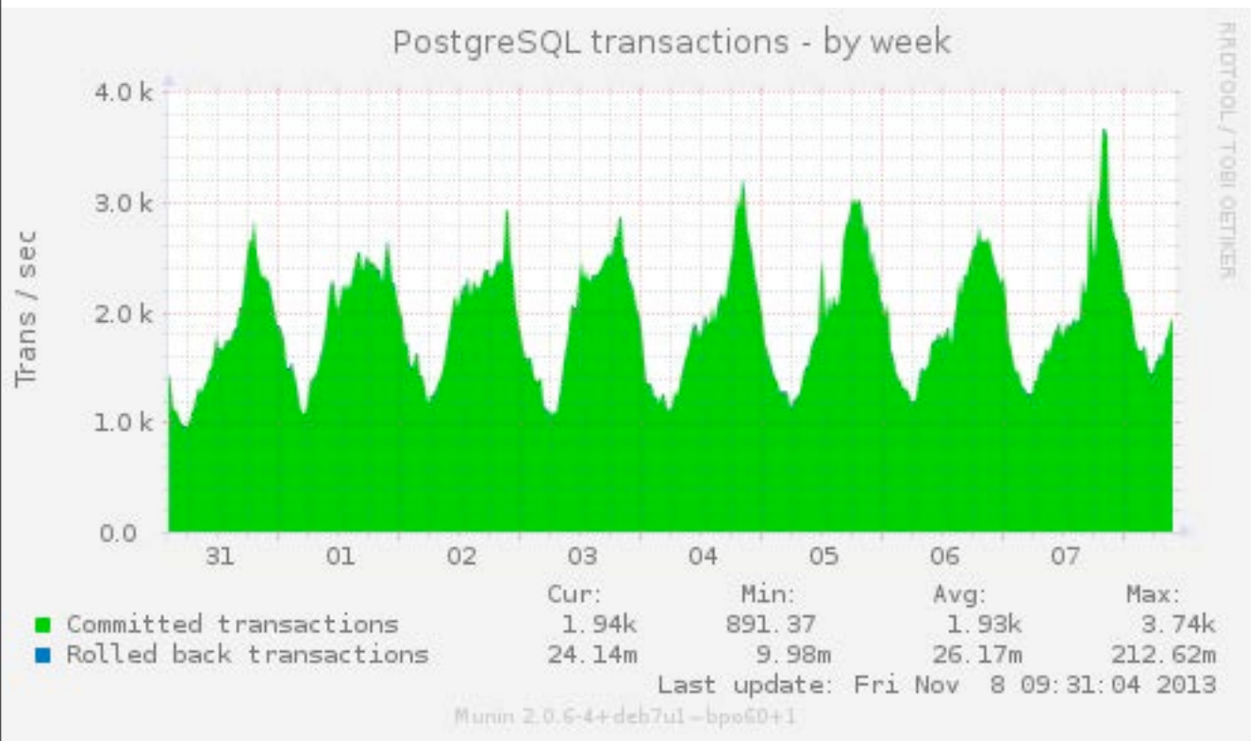
pg_view

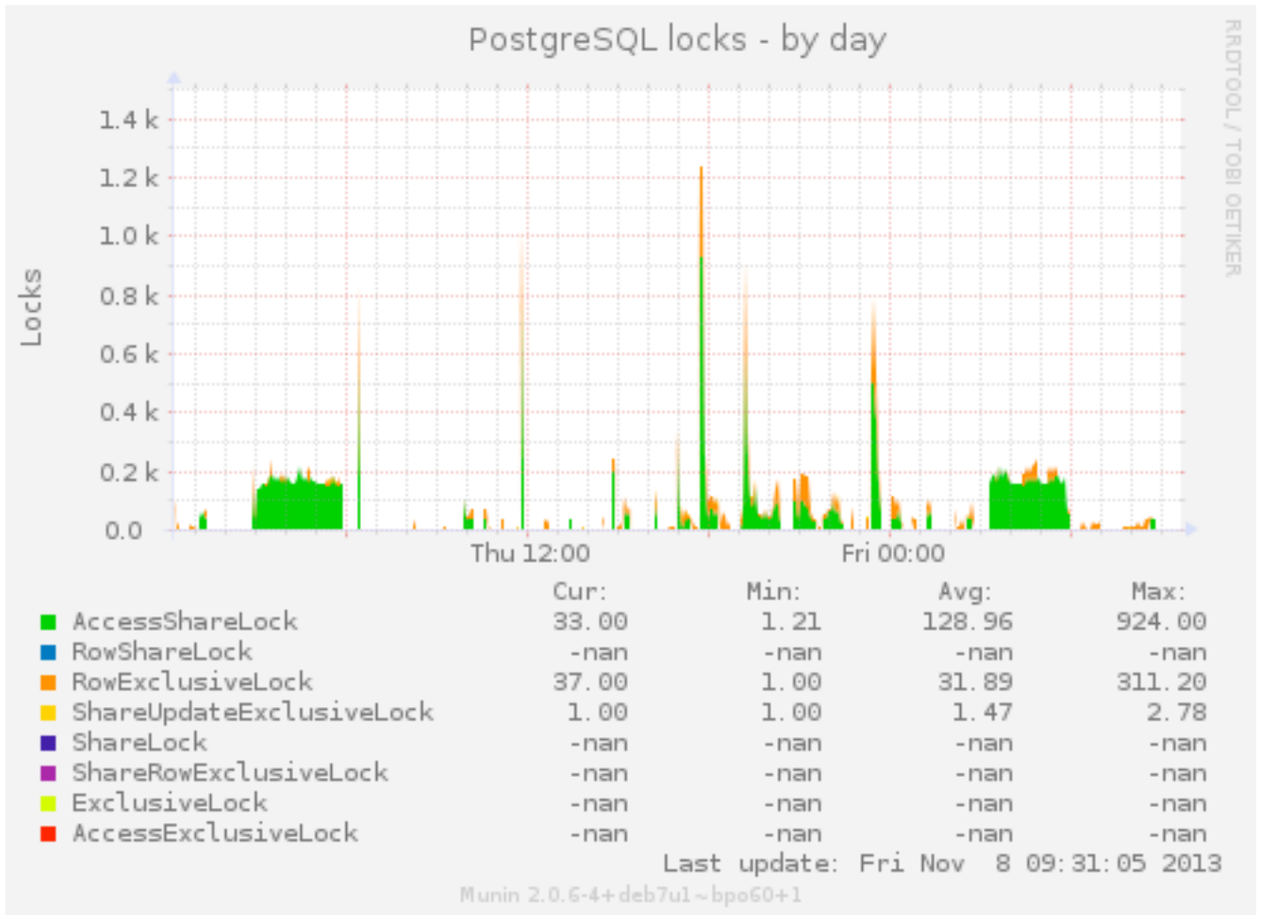

Terminal — ssh — Solarized Dark ansi — 114×29
web up 27 days, 21:15:08 8 cores Linux 2.6.32-openvz-042stab081.3-amd64 load average 0.25 0.23 0.29 09:20:03
sys: utime % 15.5 stime % 4.1 idle % 68.9 iowait % 9.5 ctxt /s 2034 run 6 block 0
mem: total MB 39.3GB free MB 29.3GB buffers MB 0KB cached MB 10.0GB dirty MB 284KB
pgaweb_prod 9.2 database connections: 6 of 100 allocated, 2 active
type dev total left
data simfs 50.0GB 16.1GB
xlog simfs 50.0GB 16.1GB
pid type s utime % stime % guest % read MB/s write MB/s age db user query
2042 checkpointer S 0.0 0.0 0.0 0.0 0.0
2043 writer S 0.0 0.0 0.0 0.0 0.0
2044 wal writer S 0.0 0.0 0.0 0.0 0.2
2045 autovacuum launcher S 0.0 0.0 0.0 0.0 0.0
2046 stats collector S 0.0 0.0 0.0 0.0 0.0
15417 backend S 0.0 0.0 0.0 0.0 0.0 00:57 pgaweb root idle in transaction
16096 backend D 0.0 1.9 0.0 12.6 0.0 00:17 pgaweb root select count(*) fro...
s: System processes f: Freeze output u: Measurement units a: Autohide fields t: No trim r: Realtime h

check_postgres

```
root@web:~/check_postgres-2.21.0# ./check_postgres.pl --help | grep -A100 Actions
Actions:
Which test is determined by the --action option, or by the name of the program
 archive_ready         - Check the number of WAL files ready in the pg_xlog/archive_status
 autovac_freeze        - Checks how close databases are to autovacuum_freeze_max_age.
 backends              - Number of connections, compared to max_connections.
 bloat                 - Check for table and index bloat.
 checkpoint            - Checks how long since the last checkpoint
 cluster_id            - Checks the Database System Identifier
 commitratio           - Report if the commit ratio of a database is too low.
 connection            - Simple connection check.
 custom_query          - Run a custom query.
 database_size         - Report if a database is too big.
 dbstats               - Returns stats from pg_stat_database: Cacti output only
 disabled_triggers     - Check if any triggers are disabled
 disk_space            - Checks space of local disks Postgres is using.
 fsm_pages             - Checks percentage of pages used in free space map.
 fsm_relations         - Checks percentage of relations used in free space map.
 hitratio              - Report if the hit ratio of a database is too low.
 hot_standby_delay     - Check the replication delay in hot standby setup
 index_size            - Checks the size of indexes only.
 last_analyze          - Check the maximum time in seconds since any one table has been analyzed.
 last_autoanalyze      - Check the maximum time in seconds since any one table has been autoanalyzed.
 last_autovacuum       - Check the maximum time in seconds since any one table has been autovacuumed.
 last_vacuum           - Check the maximum time in seconds since any one table has been vacuumed.
 listener              - Checks for specific listeners.
 locks                 - Checks the number of locks.
 logfile               - Checks that the logfile is being written to correctly.
[..]
```

munin & Plugins

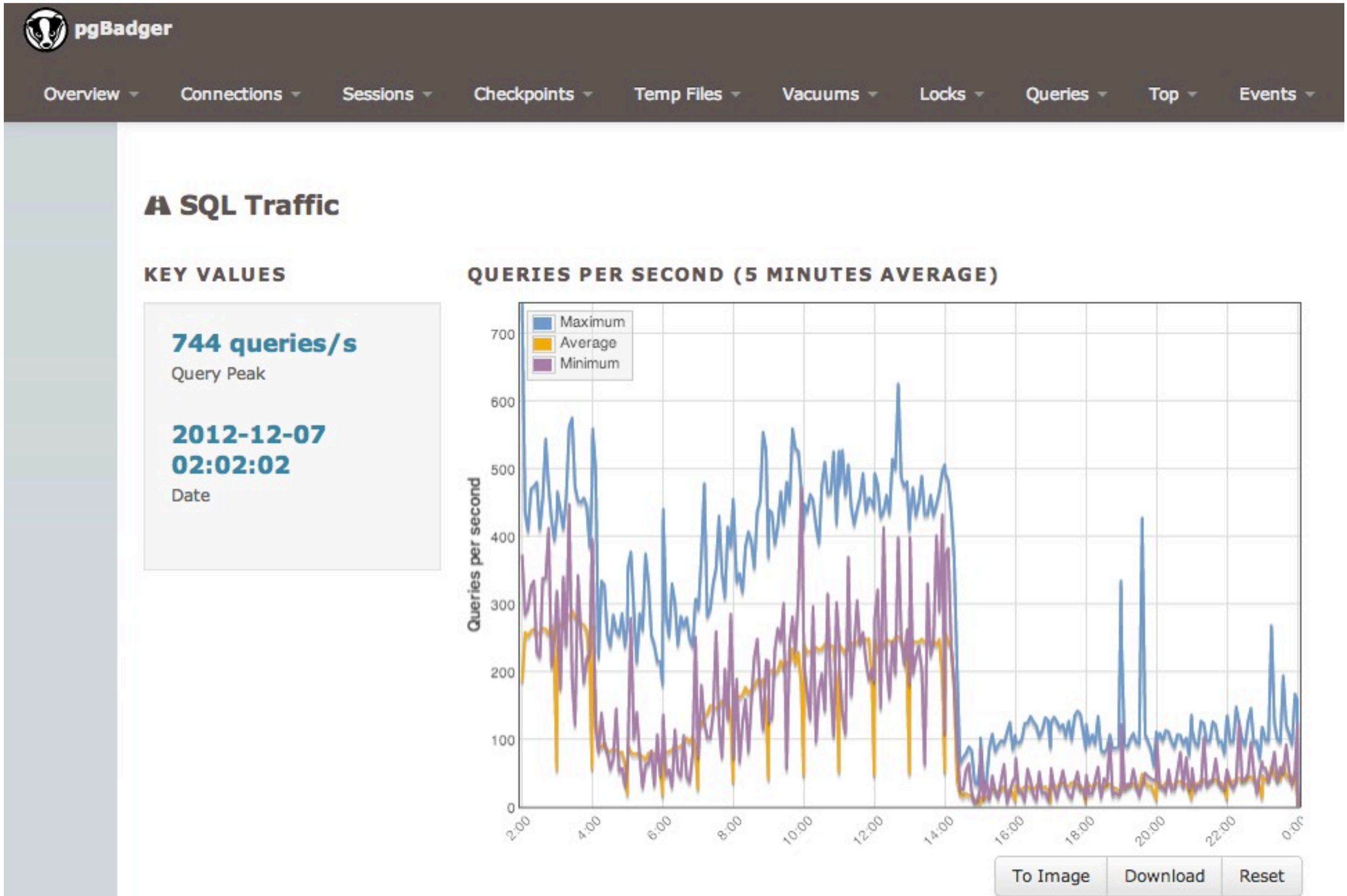




PGObserver


Load Average 15min Sprocs only
CPU Load 15min avg
Sproc Load 15 min
Top 10 Sprocs last 1 hour by total run time
Top 10 Sprocs last 1 hour by total calls
Top 10 Sprocs last 1 hour by avg. run time
Top 10 Sprocs last 3 hours by total run time
Top 10 Sprocs last 3 hours by total calls
Top 10 Sprocs last 3 hours by avg. run time

pgbadger


pgBadger
Overview
Connections
Sessions
Checkpoints
Temp Files
Vacuums
Locks
Queries
Top
Events
SQL Traffic
KEY VALUES
744 queries/s
Query Peak
2012-12-07 02:02:02
Date
QUERIES PER SECOND (5 MINUTES AVERAGE)
Maximum
Average
Minimum
Queries per second
0
100
200
300
400
500
600
700
2:00
4:00
6:00
8:00
10:00
12:00
14:00
16:00
18:00
20:00
22:00
To Image
Download
Reset

Der ganze Rest

http://wiki.postgresql.org/wiki/Monitoring

Alles ein bisschen

unzufriedenstellend

branch: **master** **pganalyze** / Commits

Nov 21, 2012

Add first collector draft
terrorobe authored a year ago

5a5c998248

Browse code

Nov 15, 2012

Initial commit.
lfittl authored a year ago

65f5d7818a

Browse code


pganalyze

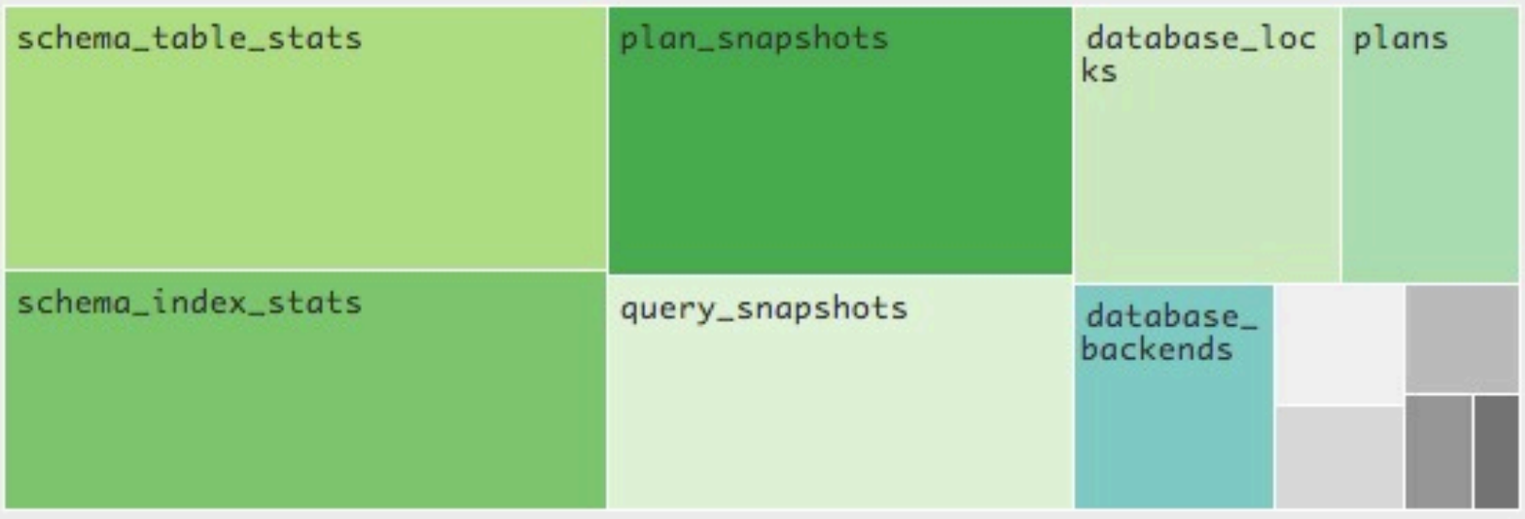
db.pganalyze.com > Dashboard

Average Query Runtime

Up & Running

- ! Some tables are overly wasting space
- ✔ All indices are in use
- ✔ All indices are valid
- ✔ No indices are overly wasting space
- ✔ Settings look alright
- ✔ Storage space looks good

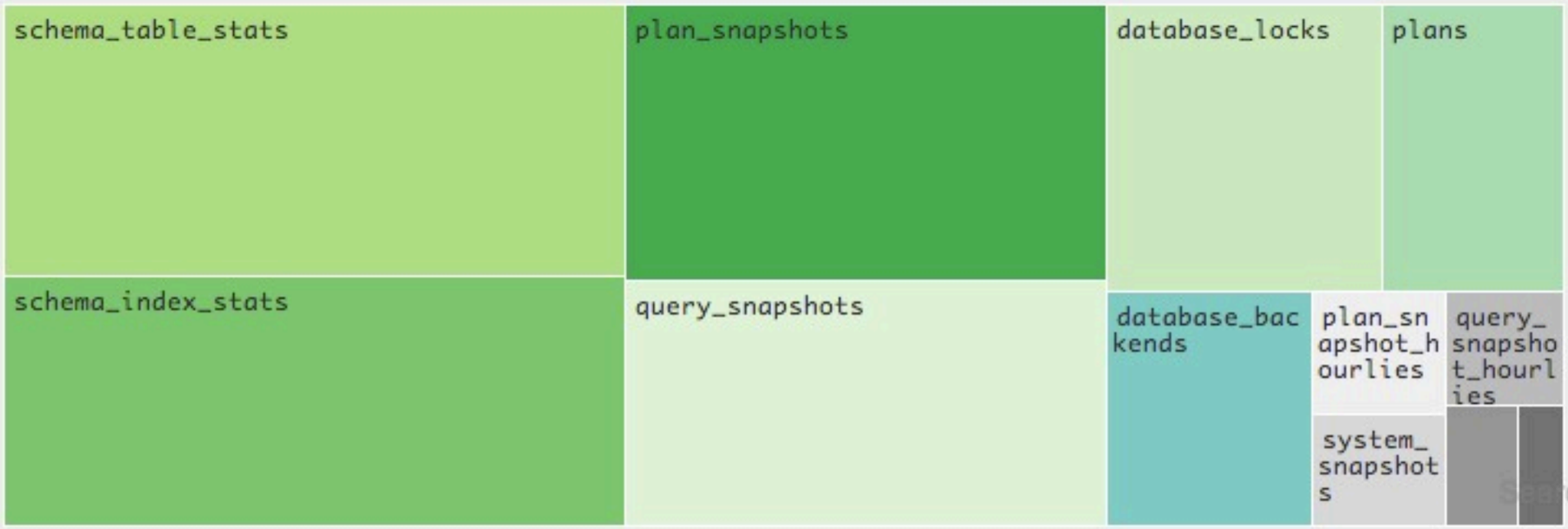
Tables



Hardware & OS

Debian 7.2 / Linux 2.6.32-openvz-042stab081.3-amd64

Load Avg: 0.62 0.28 0.27
Memory: 39.3 GB (74.74% free)
Disk Space: 50 GB (32.61% free)



Search:

Schema	Table	Size	Slack
public	schema_table_stats	1.38 GB	0
public	schema_index_stats	1.25 GB	370.01 MB
public	plan_snapshots	1.09 GB	0
public	query_snapshots	974.13 MB	245.40 MB
public	database_locks	658.77 MB	157.23 MB
public	plans	438.28 MB	127.46 MB
public	database_backends	401.02 MB	129.83 MB
public	plan_snapshot_hourlies	137.63 MB	39.58 MB
public	system_snapshots	121.35 MB	62.78 MB

db.pganalyze.com > Database > Queries

Search:

Query	Classified as	Avg Time (ms)	Calls	Total Time (ms)
SELECT query_id, total_time, calls, date_part(?, snapshots.collected_at) AS collected_at FROM query_snapshots JOIN snapshots ON (snapshot_id = snapshots.id) WHERE database_id = ? AND collected_at > NOW() - INTERVAL ?	OLTP	458.34	2781	1274650.00
SELECT "plan_snapshots".* FROM "plan_snapshots" WHERE "plan_snapshots"."plan_id" = $1 AND "plan_snapshots"."snapshot_id" = ? LIMIT ?	OLTP	1.68	78457	131867.00
SELECT "query_snapshots".* FROM "query_snapshots" WHERE "query_snapshots"."query_id" = $1 AND "query_snapshots"."snapshot_id" = ? LIMIT ?	OLTP	1.65	78454	129263.00
SELECT schema_index_id FROM schema_index_stats JOIN snapshots ON (snapshots.id =	OLTP	89.57	1366	122353.00

db.pganalyze.com > Database > Queries > Query #7724

```
SELECT query_id, total_time, calls, date_part(?, snapshots.collected_at) AS collected_at FROM query_snapshots JOIN sn
apshots ON (snapshot_id = snapshots.id) WHERE database_id = ? AND collected_at > NOW() - INTERVAL ?
```

Plan #53780

Query Plan

```
Nested Loop  (cost=0.01..2266395.74 rows=87375 width=24)
  ->  Index Scan on snapshots  (cost=0.01..8353.68 rows=1818 width=12)
        Index Cond: ((database_id = 3) AND (collected_at > (now() - '14 days 00:11:00'::interval)))
  ->  Index Scan on query_snapshots  (cost=0.00..1016.93 rows=225 width=20)
        Index Cond: (snapshot_id = snapshots.id)
```

Index Check

✓ snapshots.database_id ✓ query_snapshots.snapshot_id ✓ snapshots.id

ℹ snapshots.collected_at

Plan #59755

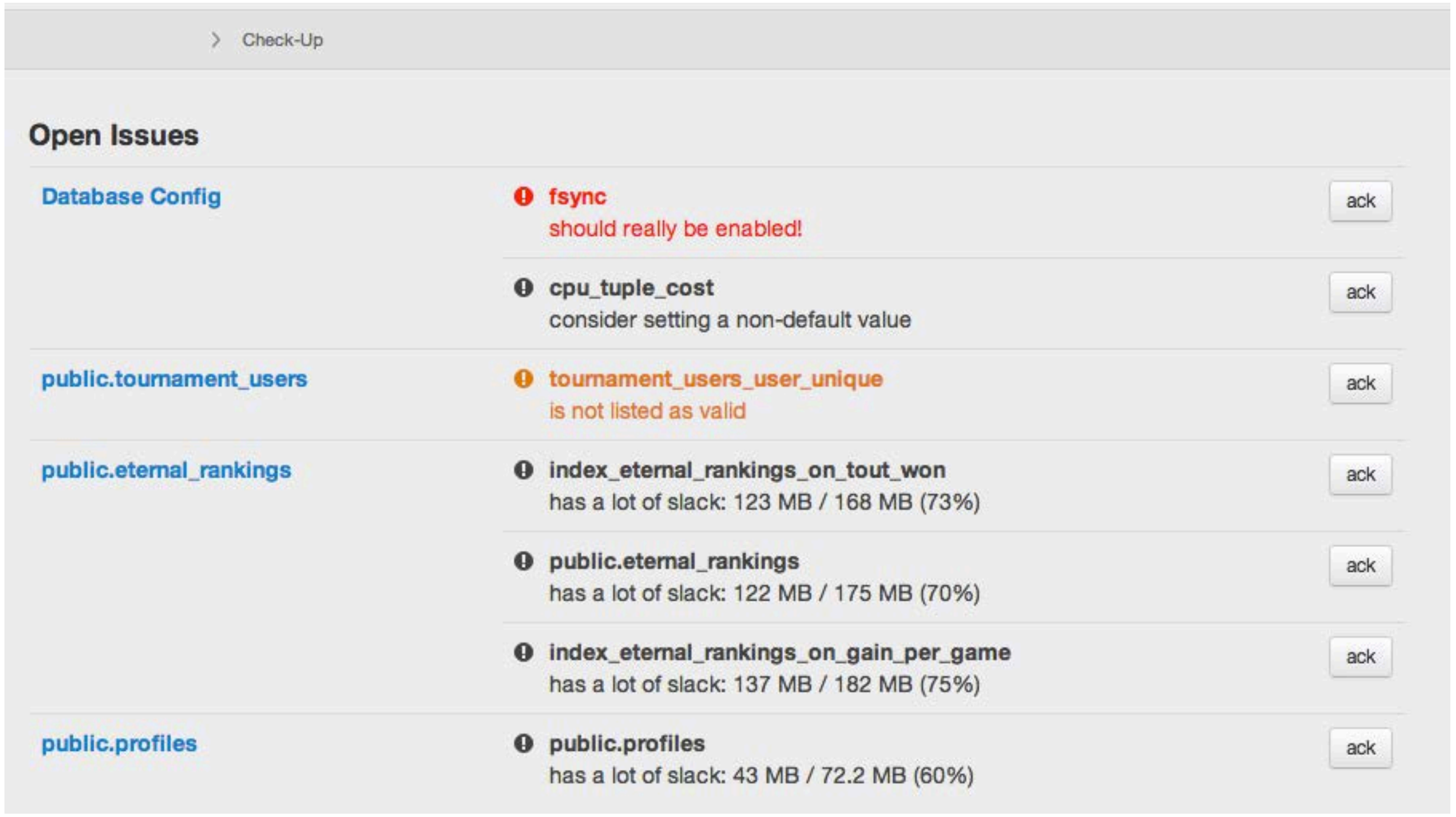
Query Plan

```
Nested Loop  (cost=6.59..1913714.18 rows=71857 width=24)
  ->  Index Scan on snapshots  (cost=0.01..7901.53 rows=1719 width=12)
        Index Cond: ((database_id = 29) AND (collected_at > (now() - '14 days 00:11:00'::interval)))
  ->  Bitmap Heap Scan on query_snapshots  (cost=6.59..907.57 rows=201 width=20)
        ->  Bitmap Index Scan  (cost=0.00..6.54 rows=201 width=0)
              Index Cond: (snapshot_id = snapshots.id)
```

Index Check

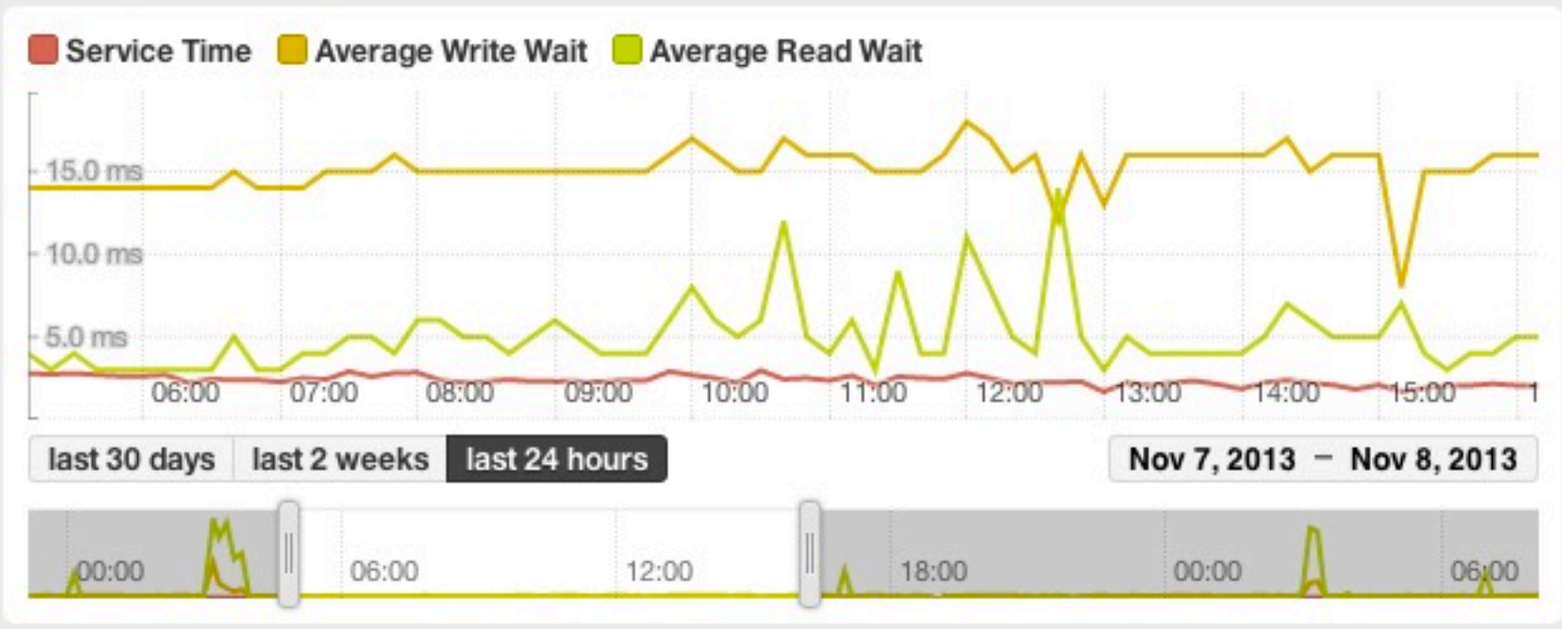
✓ snapshots.database_id ✓ query_snapshots.snapshot_id ✓ snapshots.id

ℹ snapshots.collected_at


Check-Up
Open Issues
Database Config
fsync
should really be enabled!
ack
cpu_tuple_cost
consider setting a non-default value
ack
public.tournament_users
tournament_users_user_unique
is not listed as valid
ack
public.eternal_rankings
index_eternal_rankings_on_tout_won
has a lot of slack: 123 MB / 168 MB (73%)
ack
public.eternal_rankings
has a lot of slack: 122 MB / 175 MB (70%)
ack
index_eternal_rankings_on_gain_per_game
has a lot of slack: 137 MB / 182 MB (75%)
ack
public.profiles
public.profiles
has a lot of slack: 43 MB / 72.2 MB (60%)
ack

Storage IOPS

Storage Latency



Community?

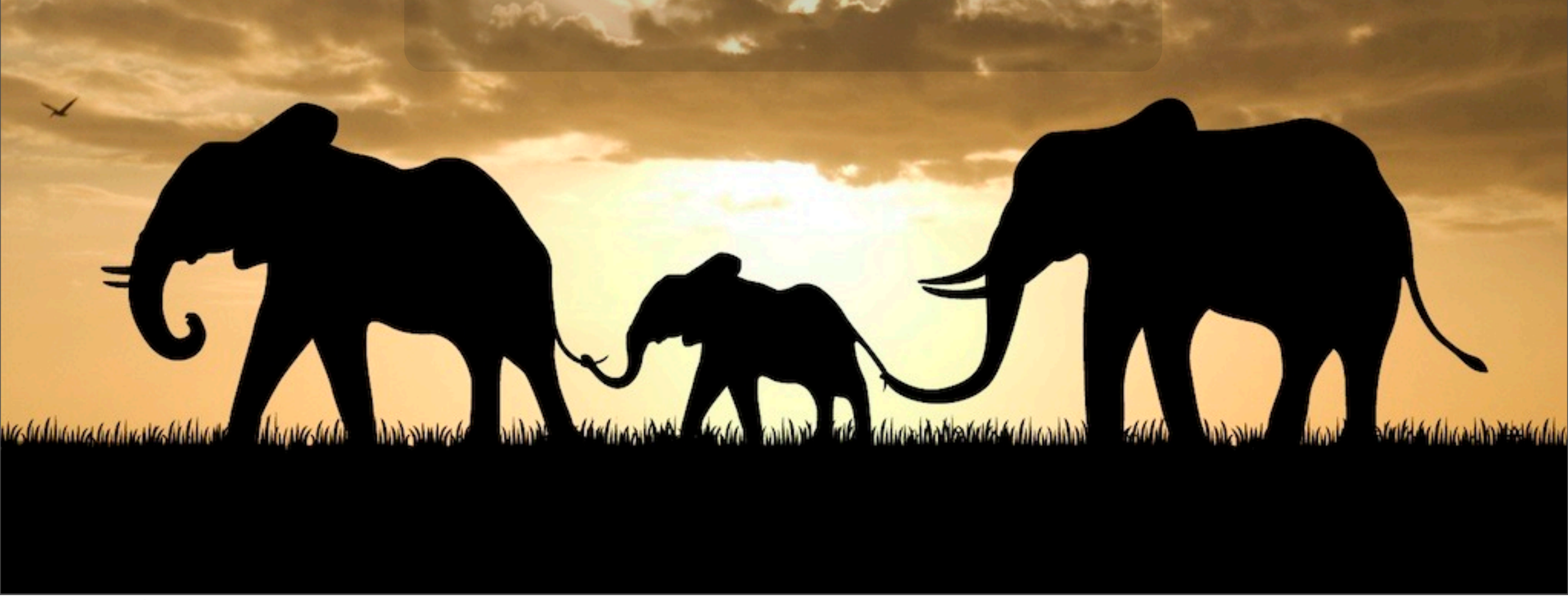
- Gratis für postgresql.org und nicht-kommerzielle Projekte
- Python Agent kann wiederverwendet werden, BSD Lizenz
- Javascript Graphing Library auch veröffentlicht

https://pganalyze.com/

https://github.com/pganalyze/

Abschliessend...

Fragen?



Danke!
Michael Renner
@terrorobe