How to Analyze and Optimize Postgres

Analyze Postgres DB

Use the command below to access Postgres log stream: `HEROKU LOGS -T --PS HEROKU-POSTGRES -A DYNAMARTAPP` (replace <dynamartapp> with any other application name)

```
2022-09-02T20:59:56.000000+00:00 app[heroku-postgres]: source=HEROKU_POSTGRESQL_COBALT addon=postgresql-fitted-54557 sample#current_transaction=90492726 sample#db_size=423557984815bytes sample#tabl
es=1745 sample#active-connections=15 sample#waiting-connections=0 sample#index-cache-hit-rate=0.99806 sample#table-cache-hit-rate=0.53121 sample#load-avg-1m=0.029375 sample#load-avg-5m=0.05625 samp
le#load-avg-15m=0.0875 sample#read-iops=0 sample#write-iops=189.28 sample#tmp-disk-used=543698944 sample#tmp-disk-available=72435093504 sample#memory-total=130409756kB sample#memory-free=2104644kB
sample#memory-cached=126011284kB sample#memory-postgres=108888kB sample#follower-lag-commits=0 sample#wal-percentage-used=0.10665183461828531
```

This command provides useful information. Details about each metrics are available here. The key metrics are :

- index-cache-hit-rate: It is recommended to have a value of 0.99 or greater if possible. If the index rate hit consistently less than 0.99, review expensive query or upgrade database.
- table-cache-hit-rate: It is recommended to have a value of 0.99 or greater if possible. If the index rate hit consistently less than 0.99, you need to upgrade the database with additional memory.
- memory-postgres: Amount of memory used by postgres
- load-avg-1m, load-avg-5m, load-avg-15m: The average system load over a period of 1 minute, 5 minutes and 15 minutes, divided by the number of available CPUs (1.0 indicates that, on average, processes were requesting CPU resources for 100% of the timespan).

To access extra postgres analysis, install the 'pg-extras' CLI plugin: `HEROKU PLUGINS:INSTALL HEROKU-PG-EXTRA`

To access the list of commands with help use : `HEROKU HELP PG`

Optimize Query

EXPLAIN ANALYZE Visualized: https://explain.depesz.com/

Out of Temp Space

Access Papertrail log, search for 'temporary file', then, use the PID to search for the originating query.

Heroku reference document : My Heroku Postgres instance is running out of temporary space

PG_STAT_ACTIVITY

Identify the idling query run the query:

**SELECT age(clock_timestamp(), query_start), *
FROM pg_stat_activity
WHERE state != 'idle' AND query NOT ILIKE '%pg_stat_activity%'
ORDER BY query_start desc;**

PG-EXTRA

Install the CLI plugin : `HEROKU PLUGINS:INSTALL HEROKU-PG-EXTRA`

queries that have taken out an exclusive lock on a relation: `HEROKU PG:LOCKS -A DYNAMARTAPP` (replace <dynamartapp> with any other application name)

```
10:24 PM:~$ heroku pg:locks -a dynamartapp
   pid   | relname | transactionid | granted |                                query_snippet                                 |       age
---------+---------+---------------+---------+------------------------------------------------------------------------------+-----------------
 1026940 |         |      91132045 | t       | update int_zendesk.zendesk_tickets set assignee_id = replace(assignee_id ,'.0','') | 00:06:01.369219
 1026940 |         |               | t       | update int_zendesk.zendesk_tickets set assignee_id = replace(assignee_id ,'.0','') | 00:06:01.369219
 1029012 |         |               | t       | autovacuum: VACUUM ANALYZE int_zendesk.zendesk_tickets                       | 00:04:31.346725
 1026939 |         |               | t       | select * from int_totango.totango_users                                      | 00:00:59.821297
 1029378 |         |               | t       | autovacuum: VACUUM int_zendesk.zendesk_side_conversations                    | 00:00:31.307952
(5 rows)
```

Identify long running queries (over 5 min): `HEROKU PG:LONG-RUNNING-QUERIES -A DYNAMARTAPP` (replace <dynamartapp> with any other application name)

```
10:27 PM:~$ heroku pg:long-running-queries -a dynamartapp
   pid   |        duration        |                                   query
---------+------------------------+----------------------------------------------------------------------------
  348402 | 5 days 07:23:52.133706 | START_REPLICATION 588C/3C000000 TIMELINE 7
  349677 | 5 days 07:11:24.254419 | START_REPLICATION 588D/56000000 TIMELINE 7
 1026940 | 00:07:32.851601        | update int_zendesk.zendesk_tickets set assignee_id = replace(assignee_id ,'.0','')
 1029012 | 00:06:02.829107        | autovacuum: VACUUM ANALYZE int_zendesk.zendesk_tickets
(4 rows)
```

⚠️ START_REPLICATION lines see comment here.

kill a process: `HEROKU PG:KILL <PID> --FORCE -A DYNAMARTAPP` (replace <dynamartapp> with any other application name)

PG SETTING

List log setting: `HEROKU PG:SETTINGS DATABASE_URL - DYNAMARTAPP` (replace <dynamartapp> with any other application name)

control which SQL statement are logged: `HEROKU PG:SETTINGS:LOG-STATEMENT <OPTION> DATABASE_URL -A DYNAMARTAPP`
replace <option> with:
- none: stops logging normal queries
- ddl: data definition statement.
- mod: ddl + data modifying statement
- all : all functions

Reference document: Heroku PGSetting

Related articles

- How to Analyze and Optimize Postgres
- Encrypt and Decrypt using PGCRYPTO
- Heroku - Create Data Link
- Heroku - CLI Main Commands
- Heroku Prod DB Upgrade