Aller plus loin avec VACUUM

À propos de l'auteur

- Auteur: Julien Rouhaud
- julien.rouhaud@dalibo.com
- Société: Dalibo
- Date: Juin 2014 pour le PG Day France
- URL: ww.dalibo.org

Licence Create Commons BY-NC-SA

- Libre de redistribuer et/ou modifier cette création selon les conditions suivantes :
  - paternité
  - pas d'utilisation commerciale
  - partage des conditions initiales à l'identique

Introduction

- PostgreSQL est une base robuste
- Des tâches de maintenance sont nécessaires
  - VACUUM
  - ANALYZE
- But de cette conférence
  - comprendre les opérations de maintenance
  - comprendre leurs paramétrages

Rappels théoriques

- Moteur MVCC de PostgreSQL
  - Copy on Write
  - visibilité des lignes
  - fragmentation
  - freeze des lignes

Théorie: MVCC 1/2

- Mise à jour d'une ligne
  - duplique la ligne
- Suppression d'une ligne
  - marque la ligne comme invisible
- Deux colonnes système
  - xmin
  - xmax

Exemple : nouvelle table

```
conf=# CREATE TABLE t1(c1 integer, c2 text);
CREATE TABLE
conf=# INSERT INTO t1 VALUES (1, 'un'), (2, 'deux'),
       (3, 'trois');
INSERT 0 3
conf=# SELECT xmin, xmax, * FROM t1;
 xmin | xmax | c1 |  c2
------+------+----+-------
 2386 |    0 |  1 | un
 2386 |    0 |  2 | deux
 2386 |    0 |  3 | trois
(3 rows)
```

Exemple : modification

```
conf=# BEGIN;
BEGIN
conf=# UPDATE t1 SET c2=upper(c2) WHERE c1=2;
UPDATE 1
conf=# SELECT ctid, xmin, xmax, * FROM t1;
 ctid  | xmin | xmax | c1 |  c2
-------+------+------+----+-------
 (0,1) | 2386 |    0 |  1 | un
 (0,3) | 2386 |    0 |  3 | trois
 (0,4) | 2387 |    0 |  2 | DEUX
(3 rows)
```

Exemple : autre session

```
conf=# SELECT ctid, xmin, xmax, * FROM t1;
 ctid  | xmin | xmax | c1 |  c2
-------+------+------+----+-------
 (0,1) | 2386 |    0 |  1 | un
 (0,2) | 2386 | 2387 |  2 | deux
 (0,3) | 2386 |    0 |  3 | trois
(3 rows)
```

Exemple : après commit

```
conf=# SELECT ctid, xmin, xmax, * FROM t1;
 ctid  | xmin | xmax | c1 |  c2
-------+------+------+----+-------
 (0,1) | 2386 |    0 |  1 | un
 (0,3) | 2386 |    0 |  3 | trois
 (0,4) | 2387 |    0 |  2 | DEUX
(3 rows)
```

- La ligne (0,2) n'est plus visible, quelque soit la session

Mais la ligne est toujours présente

```
conf=# create extension pageinspect;
CREATE EXTENSION
conf=# SELECT lp, t_xmin, t_xmax, t_ctid
       FROM heap_page_items(get_raw_page('t1', 0));
 lp | t_xmin | t_xmax | t_ctid
----+--------+--------+--------
  1 |   2386 |      0 | (0,1)
  2 |   2386 |   2387 | (0,4)
  3 |   2386 |      0 | (0,3)
  4 |   2387 |      0 | (0,4)
(4 rows)
```

Théorie: MVCC 2/2

- Les lignes non visibles doivent être nettoyées
- Transaction wraparound
  - Une ligne contient l'identifiant de transaction qui l'a créée et supprimée
  - Passé 2 milliards de transactions, elle sera vue dans le future
  - FREEZE: l'identifiant de la ligne est passé à une valeur spéciale “toujours visible”

Le VACUUM et l'ANALYZE

- Commandes SQL:
  - VACUUM
  - ANALYZE
  - VACUUM ANALYZE
- Binaire :
  - vacuumdb
  - vacuumdb -Z (9.0+)
  - vacuumdb -z

Autovacuum 1/2

- Daemon PostgreSQL
  - lance des VACUUM et des ANALYZE
  - lance également des VACUUM FREEZE
  - apparu en 8.1
  - activé par défaut en 8.3
  - nécessite d'activer
    - autovacuum
    - track_counts

Autovacuum 2/2

- Désactiver l'autovacuum ?
  - mauvaise idée dans 99% des cas
  - oubli de cron pour VACUUM/ANALYZE
    - lancé sur la mauvaise base
    - ou oubli d'une base
- Cela n'empêche pas des appels manuels

VACUUM - Fonctionnement

- 1ère passe
  - mémorisation des enregistrements à supprimer
  - nombre de lignes conditionné par maintenance_work_mem
- 2ème passe
  - nettoyage des index pour ces enregistrements
- 3ème passe
  - nettoyage des lignes

Autovacuum - configuration

- Configuration globale
  - fichier postgresql.conf
- Configuration par table
  - `ALTER TABLE nom_table SET (parametre=valeur);`
- Paramétrage bon pour petite et moyenne table
  - nécessite une configuration au cas par cas pour des tables de plusieurs dizaines de millions de lignes

Autovacuum – configuration 1/3

- autovacuum = on
- track_counts = on
- maintenance_work_mem
- autovacuum_max_workers

Autovacuum – configuration 2/3

- autovacuum_naptime (défaut 1min)
- autovacuum_vacuum_cost_delay (défaut 20ms)
- autovacuum_vacuum_cost_limit (défaut 200)

Autovacuum – configuration 3/3

- autovacuum_(vacuum|analyze)_threshold
  - nombre de lignes
  - par défaut à 50
- autovacuum_(vacuum|analyze)_scale_factor
  - pourcentage de lignes
  - par défaut à
    - 20% pour l'autovacuum
    - 10% pour l'autoanalyze

Exemple – nouvelle table

```
conf=# CREATE TABLE t2(id integer);

CREATE TABLE

conf=# INSERT INTO t2 SELECT generate_series(1,
1000000);

INSERT 0 1000000

conf=# SELECT n_dead_tup, last_autovacuum

       FROM pg_stat_user_tables WHERE relname='t2';

 n_dead_tup | last_autovacuum

------------+-----------------

          0 |

(1 row)
```

Exemple - suppression

```
conf=# DELETE FROM t2 WHERE id<200000;
DELETE 199999
conf=# SELECT pg_sleep(60);
 pg_sleep
----------

(1 row)

conf=# SELECT n_dead_tup, last_autovacuum
       FROM pg_stat_user_tables WHERE relname='t2';
 n_dead_tup | last_autovacuum
------------+-----------------
     199999 |
(1 row)
```

Exemple – plus de suppression

```
conf=# DELETE FROM t2 WHERE id<201000;
DELETE 1000
conf=# SELECT pg_sleep(60);
 pg_sleep
----------

(1 row)

conf=# SELECT n_dead_tup, last_autovacuum
       FROM pg_stat_user_tables WHERE relname='t2';
 n_dead_tup |       last_autovacuum
------------+-------------------------------
          0 | 2014-05-20 11:47:43.94275+02
```

Supervision

- Deux catalogues systèmes
  - pg_database
  - pg_class
- Un catalogue statistique
  - pg_stat_all_tables

Supervision standard

- pg_stat_all_tables
  - n_dead_tup
  - n_mod_since_analyze (9.4)
  - last_vacuum
  - last_autovacuum
  - last_analyze
  - last_autoanalyze
  - (vacuum, autovacuum, etc)_count

Supervision FREEZE

- pg_database
  - datfrozenxid
- pg_class
  - relfrozenxid

Estimation de la fragmentation

- Estimations
  - check_postgres
  - pg_freespacemap
- Calcul exact
  - pgstattuple

Exemple – nouvelle table

```
conf=# CREATE TABLE t3(id integer);

CREATE TABLE

conf=# ALTER TABLE t3 SET (autovacuum_enabled=false);

ALTER TABLE

conf=# INSERT INTO t3 SELECT generate_series(1,
1000000);

INSERT 0 1000000

conf=# DELETE FROM t3 WHERE id%3=1;

DELETE 333334

conf=# CREATE EXTENSION pgstattuple;

CREATE EXTENSION

conf=# CREATE EXTENSION pg_freespacemap;

CREATE EXTENSION
```

Exemple – pgstattuple (avant)

```
conf=# SELECT * FROM pgstattuple('t3');
-[ RECORD 1 ]------+---------
table_len          | 36249600
tuple_count        | 666666
tuple_len          | 18666648
tuple_percent      | 51.49
dead_tuple_count   | 333334
dead_tuple_len     | 9333352
dead_tuple_percent | 25.75
free_space         | 125700
free_percent       | 0.35
```

Exemple – pgstattuple (après)

```
conf=# SELECT * FROM pgstattuple('t3');
-[ RECORD 1 ]------+---------
table_len          | 36249600
tuple_count        | 666666
tuple_len          | 18666648
tuple_percent      | 51.49
dead_tuple_count   | 76
dead_tuple_len     | 2128
dead_tuple_percent | 0.01
free_space         | 10789956
free_percent       | 29.77
```

Exemple – pg_freespacemap (avant)

```
conf=# SELECT * FROM pg_freespace('t3') LIMIT 2;
 blkno | avail
-------+-------
     0 |     0
     1 |     0
(2 rows)
conf=# SELECT sum(avail) FROM pg_freespace('t3');
 sum
-----
   0
(1 row)
```

Exemple – pg_freespacemap (après)

```
conf=# SELECT * FROM pg_freespace('t3') LIMIT 2;
 blkno | avail
-------+-------
     0 |  2432
     1 |  2400
(2 rows)
conf=# SELECT sum(avail) FROM pg_freespace('t3');
   sum
----------
 10668512
(1 row)
```

Problèmes fréquents

- Tables temporaires
- Streaming Replication
- Traitements longs

Problèmes fréquents 1/2

- Tables temporaires
  - fragmentation du catalogue système
  - non prise en compte par l'autovacuum
- Streaming Replication
  - conflit d'autovacuum avec la réplication
  - hot_standby_feedback
    - empêche ces conflits
    - augmente la fragmentation

Problèmes fréquents 2/2

- Traitements qui importent de nombreuses lignes
- Traitements sur une seule transaction
  - autovacuum/analyze ne “voit” pas ces lignes
  - nécessaire d'effecter VACUUM/ANALYZE manuellemment durant le traitement
- autovacuum FREEZE
  - peut se superviser
  - déclencher le FREEZE à un meilleur moment

Conclusion

- MVCC excellent mais avec défauts/limitations
- Deux opérations de maintenance essentielles
  - VACUUM
  - ANALYZE
- Autovacuum à conserver activé
  - mais optimisation à faire dans la configuration

Des questions ?

- Auteur: Julien Rouhaud
  - julien.rouhaud@dalibo.com
- Société: Dalibo
  - http://ww.dalibo.org