聊聊PostgreSQL表膨胀

作者	时间	QQ技术交流群
perrynzhou@gmail.com	2022/08/20	672152841

存储内核技术交流

微信扫描二维码，关注我的公众号

分布式存储技术研...

群号：672152841

扫一扫二维码，加入群聊。

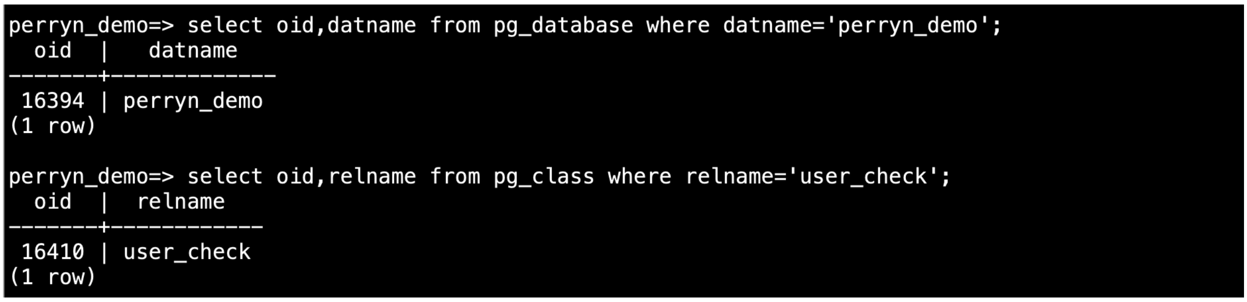
QQ

开源存储问题解答社区:https://github.com/perrynzhou/deep-dive-storage-in-china

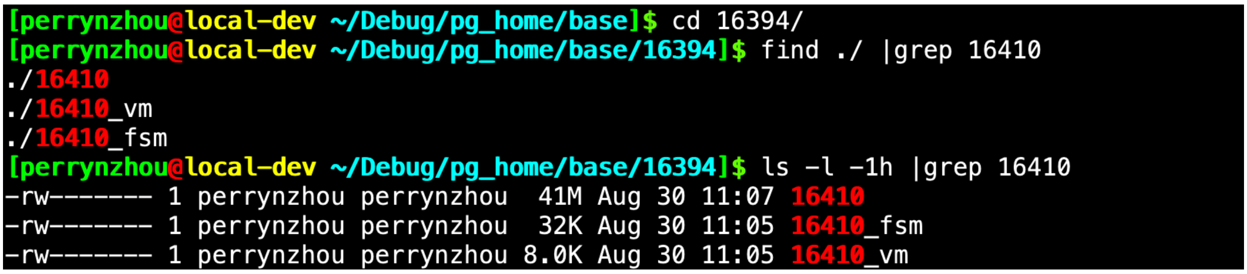
PostgreSQL Basic

- PG中的MVCC(多版本并发)设计目的是读不阻塞写。PG中的所有的insert和update操作都是创建新的一行数据；update和delete都不是立即删除旧版本无用的数据。tuple是否可见是由snapshot决定。

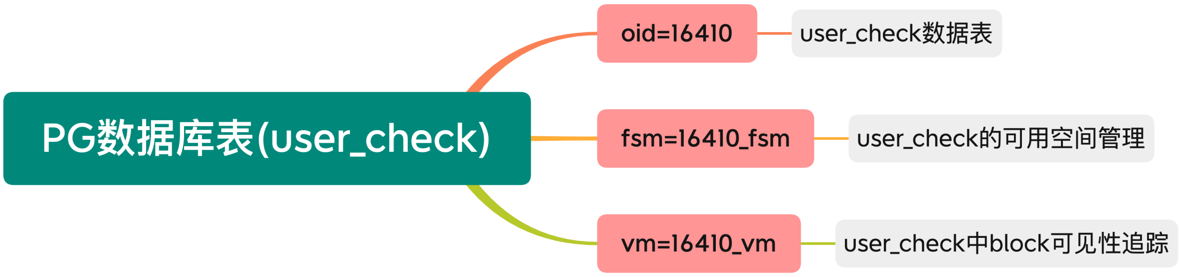
- PG中追踪每个表的Block可见性是通过表的vm文件。Table或者Index的可用空间管理是通过表或者索引的fsm文件管理，它是一个2级的binary tree，最底层存储了每个page可用空间，最上层聚合最低层的信息。



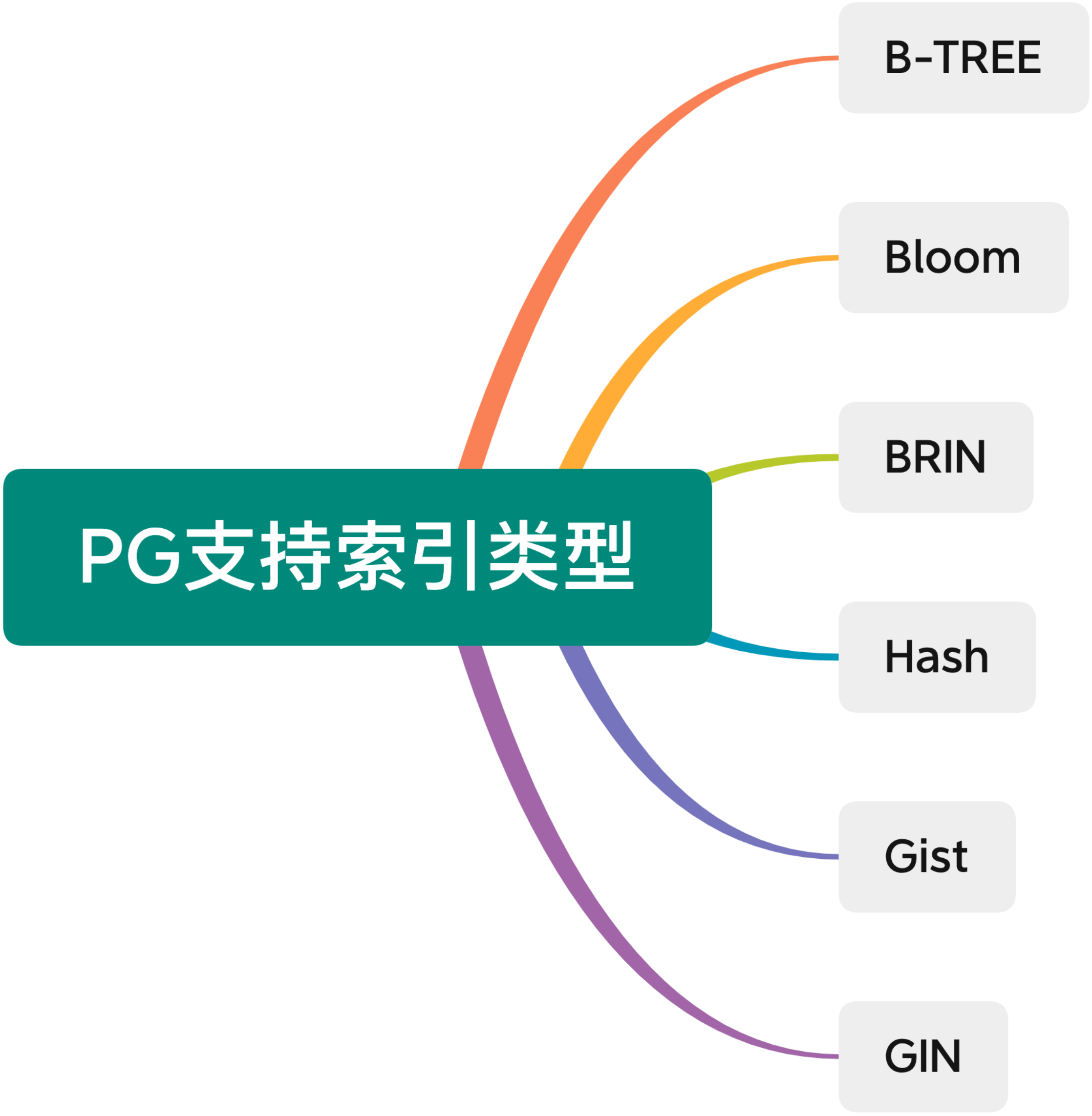
```
perryn_demo=> select oid,datname from pg_database where datname='perryn_demo';
  oid  |   datname
-------+-------------
 16394 | perryn_demo
(1 row)

perryn_demo=> select oid,relname from pg_class where relname='user_check';
  oid  |  relname
-------+------------
 16410 | user_check
(1 row)
```



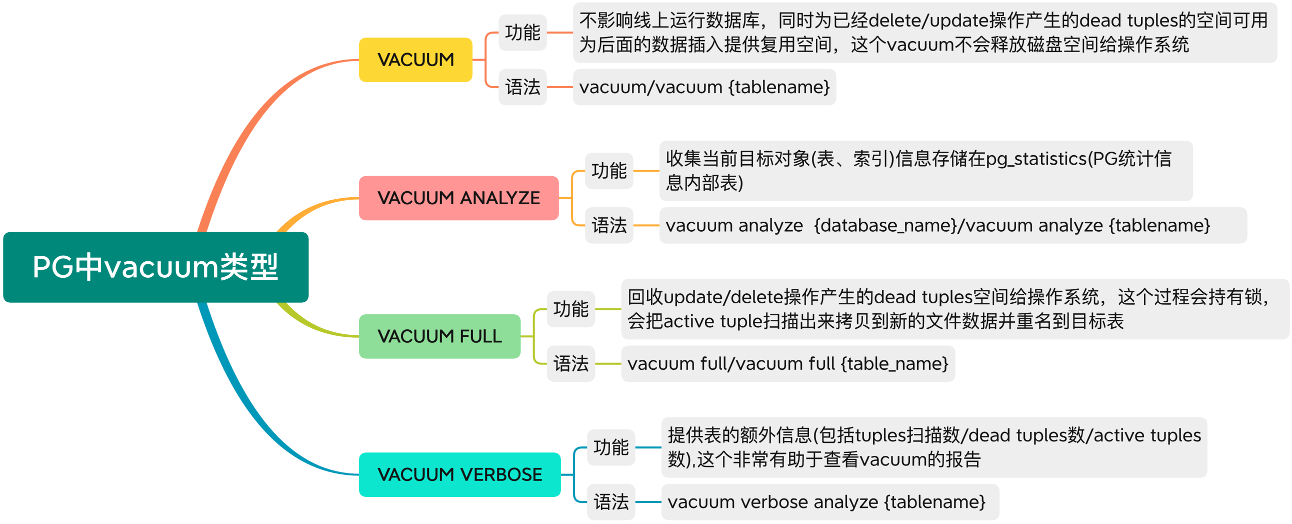
```
[perrynzhou@local-dev ~/Debug/pg_home/base]$ cd 16394/
[perrynzhou@local-dev ~/Debug/pg_home/base/16394]$ find ./ |grep 16410
./16410
./16410_vm
./16410_fsm
[perrynzhou@local-dev ~/Debug/pg_home/base/16394]$ ls -l -1h |grep 16410
-rw------- 1 perrynzhou perrynzhou  41M Aug 30 11:07 16410
-rw------- 1 perrynzhou perrynzhou  32K Aug 30 11:05 16410_fsm
-rw------- 1 perrynzhou perrynzhou 8.0K Aug 30 11:05 16410_vm
```



- `PG`目前支持多种索引类型，包括`B-Tree、Hash、Gin、Gist、Brin、Bloom`。



`PostgreSQL`膨胀



VACUUM

功能：不影响线上运行数据库，同时为已经delete/update操作产生的dead tuples的空间可用为后面的数据插入提供复用空间，这个vacuum不会释放磁盘空间给操作系统

语法：vacuum/vacuum {tablename}

VACUUM ANALYZE

功能：收集当前目标对象(表、索引)信息存储在pg_statistics(PG统计信息内部表)

语法：vacuum analyze {database_name}/vacuum analyze {tablename}

VACUUM FULL

功能：回收update/delete操作产生的dead tuples空间给操作系统，这个过程会持有锁，会把active tuple扫描出来拷贝到新的文件数据并重名到目标表

语法：vacuum full/vacuum full {table_name}

VACUUM VERBOSE

功能：提供表的额外信息(包括tuples扫描数/dead tuples数/active tuples数),这个非常有助于查看vacuum的报告

语法：vacuum verbose analyze {tablename}

- `膨胀`在`PG`中表示表或者索引的大小大于实际数据的大小，其次表中每个`block`或者`page`的空间利用率低。当一个事务`T1`读取表的`block B`中`A`行数据时候，第二个事务`T2`去更新这

个表中`Block B`中`A`行数据；为了确保`read`事务不阻塞`write`事务，`T2`的`write`事务把更新后的`A`这一行数据写到新的空闲空间，而`A`这行数据依然在`Block B`中，这个就是`dead tuple`.所以在`PG`中，如果有非常多的`update`和`delete`,会产生非常多的`dead tuples`,这些`dead tuples`的集合就是`PG`中的`膨胀`。

- 针对`PG`中的`膨胀`问题是通过`vacuum`来解决，`PG`中的`auto vacuum`会阻塞`read/write`操作，手动的`vacuum`则不会阻塞。`vacuum`有三种类型，分别是`普通的vacuum、vacuum analyze、vacuum full`.

验证`PostgreSQL膨胀`

- `OS`版本

```
[perrynzhou@local-dev ~/Debug/pg_home]$ uname -a
Linux local-dev 4.18.0-348.7.1.el8_5.x86_64 #1 SMP Wed Dec 22 13:25:12 UTC
2021 x86_64 x86_64 x86_64 GNU/Linux
```

- `PostgreSQL`版本

```
[perrynzhou@local-dev ~/Debug/pg_home]$ psql --version
psql (PostgreSQL) 14.3
```

- 测试`数据库`和`表`信息

```
/*****************设置测试数据库和登录用户***********/
[perrynzhou@local-dev ~/Debug]$ psql -d postgres
psql (14.3)
Type "help" for help.

// 创建测试数据库 perryn_demo
postgres=# create database perryn_demo;
CREATE DATABASE
// 创建perryn_demo数据库用户名称为perryn_demo
postgres=# CREATE USER perryn_demo WITH ENCRYPTED PASSWORD '123456';
CREATE ROLE

// 设置用户允许登录
postgres=# ALTER USER perryn_demo WITH login;
ALTER ROLE

// 授予perryn_demo数据库操作所有权限给用户perryn_demo
postgres=# grant all privileges on database perryn_demo to perryn_demo;
GRANT


/*****************创建测试表和数据***********/
```

```
[perrynzhou@local-dev ~/Debug]$ psql -d perryn_demo -U perryn_demo
psql (14.3)
Type "help" for help.
perryn_demo=> create table user_check as select generate_series (1,10000)
as id, substr(md5(random()::text), 0, 255) as uuid, to_char(random() *
1000000, '099999') as code, substring(random()::varchar,3,8) as md5;
```

表的隐藏列

- PG中的隐藏列设计是为了MVCC功能设计，一个事务中的查询如何找到这个事务开启时候应该读取数据的版本。PG包含了tableoid、xmax、xmin、cmax、cmin、ctid这些隐藏列。xmin、xmax是不同事务之间的数据版本判断的基础。cmin、cmax、ctid是判断同一个事务内的其他命令导致的行版本变更是否可见

```
// 查询user_check表这个所有列(包括隐藏列)

perryn_demo=> drop table user_check;
DROP TABLE
perryn_demo=> create table user_check as select generate_series (1,5) as
id, substr(md5(random()::text), 0, 255) as uuid, to_char(random() *
1000000, '099999') as code, substring(random()::varchar,3,8) as md5;
SELECT 5
perryn_demo=> SELECT attrelid::regclass::text, attname, format_type
(atttypid, atttypmod) FROM pg_attribute WHERE
attrelid::regclass::text='user_check' ORDER BY attnum;
  attrelid  | attname  | format_type
------------+----------+-------------
// tableoid是表的在PG内部唯一标识
 user_check | tableoid | oid
// 删除事务中的命令标识
 user_check | cmax     | cid
// 如果xmax 为0 , 表示数据没有被删除；如果不为0，则是删除这个数据的事务ID
 user_check | xmax     | xid
// 插入事务中的命令标识
 user_check | cmin     | cid
 // xmin 是每个事务中数据插入时候的事务ID
 user_check | xmin     | xid
 user_check | ctid     | tid
 user_check | id       | integer
 user_check | uuid     | text
 user_check | code     | text
 user_check | md5      | text
(10 rows)
```

- xmin隐藏列表示数据插入时候的事务ID,xmax隐藏列表示数据删除/更改时候的事务ID.这次模拟是在会话A中初始化插入数据->会话B中更新数据->在回到会话A中查询数据来观察数据表是如何膨胀的。

```
// 禁用数据表的vacuum
ALTER TABLE ucheck SET (
  autovacuum_enabled = false, toast.autovacuum_enabled = false
);
```

```
// 会话A:查询当前的事务ID,事务ID=811
perryn_demo=> begin;
BEGIN
perryn_demo=*> select txid_current();
 txid_current
--------------
          811
(1 row)

perryn_demo=*> create table ucheck as select generate_series (1,3) as id,
substr(md5(random()::text), 0, 255) as uuid, to_char(random() * 1000000,
'099999') as code, substring(random()::varchar,3,8) as md5;
SELECT 3
perryn_demo=*> select xmin,xmax,cmin,cmax,* from ucheck;
 xmin | xmax | cmin | cmax | id |               uuid               |  code
|   md5
------+------+------+------+----+----------------------------------+-------
--+----------
  811 |    0 |    5 |    5 |  1 | f12b88a762ec72f1885145b53148c79a |
692255 | 60326622
  811 |    0 |    5 |    5 |  2 | 2c7cd94aaa74ce04ed7325a93acdeb03 |
290345 | 59971147
  811 |    0 |    5 |    5 |  3 | d3e703cd56522833b5fbadd1459b9aa0 |
548640 | 96239513
(3 rows)

perryn_demo=*> commit;
COMMIT


// 会话B:更新ucheck中字段，事务ID=813
perryn_demo=*> select txid_current();
 txid_current
--------------
          813
(1 row)

// 这里会话B中更新时候插入了2条数据，会话A中原来旧版本数据依然存在
perryn_demo=*> update ucheck set md5=substring(random()::varchar,3,8) where
id>=2;
```

```
UPDATE 2
perryn_demo=*> select xmin,xmax,cmin,cmax,* from ucheck
;
 xmin | xmax | cmin | cmax | id |               uuid               |   code
|   md5
------+------+------+------+----+----------------------------------+-------
--+----------
  811 |    0 |    5 |    5 |  1 | f12b88a762ec72f1885145b53148c79a |
692255 | 60326622
  813 |    0 |    0 |    0 |  2 | 2c7cd94aaa74ce04ed7325a93acdeb03 |
290345 | 66262299
  813 |    0 |    0 |    0 |  3 | d3e703cd56522833b5fbadd1459b9aa0 |
548640 | 29500328
(3 rows)


// 会话A:再次查看ucheck表的数据，xmax事务ID是为更新的事务ID，这里就造成了表的膨胀
perryn_demo=> begin;
BEGIN
perryn_demo=*> select xmin,xmax,cmin,cmax,* from ucheck;
 xmin | xmax | cmin | cmax | id |               uuid               |   code
|   md5
------+------+------+------+----+----------------------------------+-------
--+----------
  811 |    0 |    5 |    5 |  1 | f12b88a762ec72f1885145b53148c79a |
692255 | 60326622
  811 |    0 |    5 |    5 |  2 | 2c7cd94aaa74ce04ed7325a93acdeb03 |
290345 | 59971147
  811 |    0 |    5 |    5 |  3 | d3e703cd56522833b5fbadd1459b9aa0 |
548640 | 96239513
(3 rows)

// 这里观察到xmax = 会话B中的事务ID
perryn_demo=*> select xmin,xmax,cmin,cmax,* from ucheck;
 xmin | xmax | cmin | cmax | id |               uuid               |   code
|   md5
------+------+------+------+----+----------------------------------+-------
--+----------
  811 |    0 |    5 |    5 |  1 | f12b88a762ec72f1885145b53148c79a |
692255 | 60326622
  811 |  813 |    0 |    0 |  2 | 2c7cd94aaa74ce04ed7325a93acdeb03 |
290345 | 59971147
  811 |  813 |    0 |    0 |  3 | d3e703cd56522833b5fbadd1459b9aa0 |
548640 | 96239513
(3 rows)
perryn_demo=*> commit;
COMMIT
```

分析膨胀表的空间

pageinspect查看表的dead tuples

- 授权perryn_demo为SUPERUSER

```
[perrynzhou@local-dev ~]$ psql -d postgres
psql (14.3)
Type "help" for help.

postgres=#
postgres=# ALTER ROLE perryn_demo SUPERUSER;
ALTER ROLE
```

- 查看表的dead tuples

```
[perrynzhou@local-dev ~]$ psql -U perryn_demo -d perryn_demo
psql (14.3)
Type "help" for help.

perryn_demo=# CREATE EXTENSION pageinspect;
CREATE EXTENSION

// t_xmax中的813都是dead tuples, 目前这个表已经被禁用auto vacuum
perryn_demo=# SELECT t_xmin, t_xmax, tuple_data_split('ucheck'::regclass,
t_data, t_infomask, t_infomask2, t_bits) FROM
heap_page_items(get_raw_page('ucheck', 0));
 t_xmin | t_xmax |
tuple_data_split
--------+--------+---------------------------------------------------------
---------------------------------------------------------------------------
---
    811 |      0 |
{"\\x01000000","\\x436631326238386137363265633732663138383531343562353331343
3863373961","\\x1120363932323535","\\x1336303332363632
32"}
```

```
    811 |      0 |
{"\\x01000000","\\x436631326238386137363265633732663138383531343562353331343
3863373961","\\x1120363932323535","\\x133630333236363232"}
    811 |    813 |
{"\\x02000000","\\x433263376364393461616137346365303465643733323561393361633
6465623033","\\x1120323930333435","\\x133539393731313437"}
    811 |    813 |
{"\\x03000000","\\x436433653730336364353635323238333362356662616464313435396
239616130","\\x1120353438363430","\\x133936323339353133"}
    813 |      0 |
{"\\x02000000","\\x433263376364393461616137346365303465643733323561393361633
6465623033","\\x1120323930333435","\\x133636323632323939"}
    813 |      0 |
{"\\x03000000","\\x436433653730336364353635323238333362356662616464313435396
239616130","\\x1120353438363430","\\x133239353030333238"}
(5 rows)
```

`vacuum`重用的`dead tuples`空间

```
//
perryn_demo=# vacuum verbose analyze ucheck;
INFO:  vacuuming "public.ucheck"
INFO:  table "ucheck": found 0 removable, 5 nonremovable row versions in 1
out of 1 pages
DETAIL:  0 dead row versions cannot be removed yet, oldest xmin: 4293968109
Skipped 0 pages due to buffer pins, 0 frozen pages.
CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.00 s.
INFO:  vacuuming "pg_toast.pg_toast_16516"
INFO:  table "pg_toast_16516": found 0 removable, 0 nonremovable row
versions in 0 out of 0 pages
DETAIL:  0 dead row versions cannot be removed yet, oldest xmin: 4293968109
Skipped 0 pages due to buffer pins, 0 frozen pages.
CPU: user: 0.00 s, system: 0.00 s, elapsed: 0.00 s.
INFO:  analyzing "public.ucheck"

INFO:  "ucheck": scanned 1 of 1 pages, containing 3 live rows and 0 dead
rows; 3 rows in sample, 3 estimated total rows
VACUUM
```

```
// 会话A插入数据
perryn_demo=# begin;
BEGIN
perryn_demo=*# select xmin,xmax,cmin,cmax,* from ucheck;
 xmin | xmax | cmin | cmax | id |               uuid               |  code
|   md5
------+------+------+------+----+----------------------------------+-------
--+----------
  821 |    0 |    5 |    5 |  1 | 6b65bf7e7080ef40110cdae28e145036 |
540085 | 48283533
  821 |    0 |    5 |    5 |  2 | e4f1b77d6b14f9b55f4607b812039074 |
516574 | 67780358
  821 |    0 |    5 |    5 |  3 | 2b81b3a381ec1f70c8f3ddc6af3976b5 |
541362 | 34386270
(3 rows)

perryn_demo=*# select xmin,xmax,cmin,cmax,* from ucheck;
 xmin | xmax | cmin | cmax | id |               uuid               |  code
|   md5
------+------+------+------+----+----------------------------------+-------
--+----------
  821 |    0 |    5 |    5 |  1 | 6b65bf7e7080ef40110cdae28e145036 |
540085 | 48283533
  821 |  822 |    0 |    0 |  2 | e4f1b77d6b14f9b55f4607b812039074 |
516574 | 67780358
```

```
  821 |  822 |    0 |    0 |  3 | 2b81b3a381ec1f70c8f3ddc6af3976b5 |
541362 | 34386270
(3 rows)

perryn_demo=*# commit;

// 会话B 更新数据
perryn_demo=# begin;
BEGIN
perryn_demo=*#
perryn_demo=*# select txid_current();
 txid_current
--------------
          822
(1 row)

perryn_demo=*# update ucheck set md5=substring(random()::varchar,3,8) where
id>=2;
UPDATE 2
perryn_demo=*#  select xmin,xmax,cmin,cmax,* from ucheck;
 xmin | xmax | cmin | cmax | id |               uuid               |  code
|   md5
------+------+------+------+----+----------------------------------+-------
--+----------
  821 |    0 |    5 |    5 |  1 | 6b65bf7e7080ef40110cdae28e145036 |
540085 | 48283533
  822 |    0 |    0 |    0 |  2 | e4f1b77d6b14f9b55f4607b812039074 |
516574 | 33335208
  822 |    0 |    0 |    0 |  3 | 2b81b3a381ec1f70c8f3ddc6af3976b5 |
541362 | 69940957
(3 rows)

perryn_demo=*# ALTER TABLE ucheck SET (autovacuum_enabled = false,
toast.autovacuum_enabled = false);
ALTER TABLE
perryn_demo=*# commit;


// 未执行vaccum之前的表信息，可以看到t_xmax=822的有2条记录，这个是会话A插入时候的产生
的数据，但是被会话B(事务ID=822)更新数据后,xmax被更新为822.同时会话B插入了2条新的记录，
从这里可以看出PG是采用cow策略进行数据的更新
perryn_demo=# SELECT t_xmin, t_xmax, tuple_data_split('ucheck'::regclass,
t_data, t_infomask, t_infomask2, t_bits) FROM
heap_page_items(get_raw_page('ucheck', 0));
 t_xmin | t_xmax |
tuple_data_split
--------+--------+------------------------------------------------------------
----------------------------------------------------------------------------
---
    821 |      0 |
{"\\x01000000","\\x4336623635626637653730383065663430313130636461653238653
3435303336","\\x1120353430303835","\\x1334383238333353333"}
    821 |    822 |
{"\\x02000000","\\x43653466316237376436623134663962353566343630376238313230
3339303734","\\x1120353136353734","\\x1336373738303333538"}
    821 |    822 |
{"\\x03000000","\\x43326238316233613338316563316637306338663364646336616633
3937366235","\\x1120353431333632","\\x1333343338363233730"}
    822 |      0 |
{"\\x02000000","\\x43653466316237376436623134663962353566343630376238313230
3339303734","\\x1120353136353734","\\x1333333333335323038"}
    822 |      0 |
{"\\x03000000","\\x43326238316233613338316563316637306338663364646336616633
3937366235","\\x1120353431333632","\\x1336393934303933537"}
(5 rows)

// 执行vaccum 空间数据被标记清空，但是占用的磁盘并没有归还给操作系统，其从821是会话A的
插入事务ID。822是会话B的更新事务的ID,这里有2条空的记录被标记为后面插入数据时候可以被复
用。
perryn_demo=# vacuum ucheck;
VACUUM
perryn_demo=# SELECT t_xmin, t_xmax, tuple_data_split('ucheck'::regclass,
t_data, t_infomask, t_infomask2, t_bits) FROM
heap_page_items(get_raw_page('ucheck', 0));
 t_xmin | t_xmax |
tuple_data_split
--------+--------+----------------------------------------------------------
----------------------------------------------------------------------------
---
    821 |      0 |
{"\\x01000000","\\x43366236356266376537303830656634303131306364616532386531
3435303336","\\x1120353430303835","\\x1334383238333353333"}
        |        |
        |        |
    822 |      0 |
{"\\x02000000","\\x43653466316237376436623134663962353566343630376238313230
3339303734","\\x1120353136353734","\\x1333333333335323038"}
    822 |      0 |
{"\\x03000000","\\x43326238316233613338316563316637306338663364646336616633
3937366235","\\x1120353431333632","\\x1336393934303933537"}
(5 rows)
```

`vacuum full`回收的`dead tuples`空间

```

// 普通vaccum仅仅标记
perryn_demo=# SELECT t_xmin, t_xmax, tuple_data_split('ucheck'::regclass,
t_data, t_infomask, t_infomask2, t_bits) FROM
heap_page_items(get_raw_page('ucheck', 0));
 t_xmin | t_xmax |
tuple_data_split
--------+--------+---------------------------------------------------------
----------------------------------------------------------------------------
---
    821 |       0 |
{"\\x01000000","\\x433662363562663765373038306566343031313063646165323865313
3435303336","\\x1120353430303835","\\x1334383238333353333"}
        |         |
        |         |
    822 |       0 |
{"\\x02000000","\\x436534663162373764366231346639623535663436303762383132303
3339303734","\\x1120353136353734","\\x1333333333335323038"}
    822 |       0 |
{"\\x03000000","\\x433262383162336133383165633166373063386633646463366166363
3937366235","\\x1120353431333632","\\x1336393934303933537"}
(5 rows)

// 这里执行vacuum full,可以看出被标记的复用空闲空间归还给操作系统了，但是这个操作会产生
表锁。
perryn_demo=# vacuum full ucheck;
VACUUM
perryn_demo=# SELECT t_xmin, t_xmax, tuple_data_split('ucheck'::regclass,
t_data, t_infomask, t_infomask2, t_bits) FROM
heap_page_items(get_raw_page('ucheck', 0));
 t_xmin | t_xmax |
tuple_data_split
--------+--------+---------------------------------------------------------
----------------------------------------------------------------------------
---
    821 |       0 |
{"\\x01000000","\\x433662363562663765373038306566343031313063646165323865313
3435303336","\\x1120353430303835","\\x1334383238333353333"}
    822 |       0 |
{"\\x02000000","\\x436534663162373764366231346639623535663436303762383132303
3339303734","\\x1120353136353734","\\x1333333333335323038"}
    822 |       0 |
{"\\x03000000","\\x433262383162336133383165633166373063386633646463366166363
3937366235","\\x1120353431333632","\\x1336393934303933537"}
(3 rows)
```