PG BootCamp Russia 2025 Ekaterinburg

PGBootCamp.ru

Upgrade каталогов PostgreSQL/Greenplum

Cloudberry and greenplum development

Kirill Reshke
Developer, Yandex

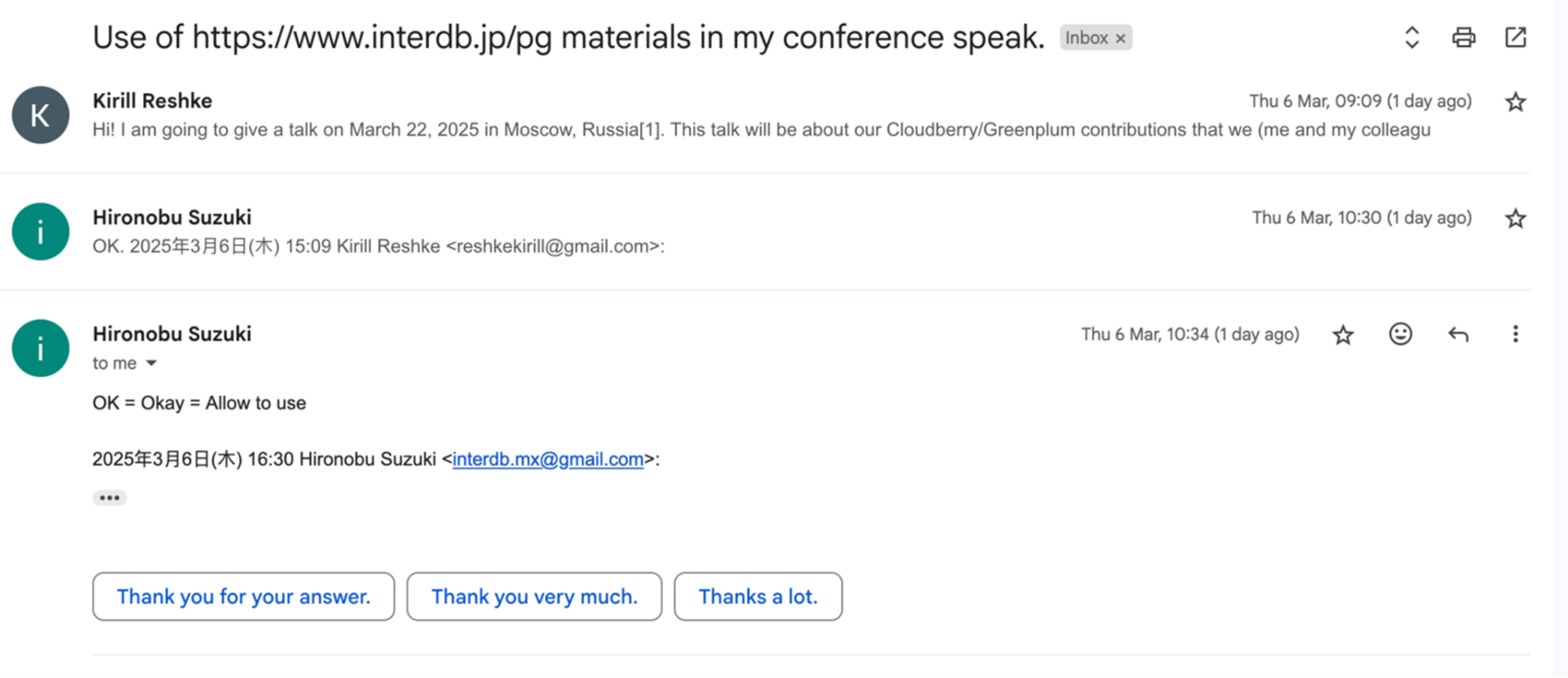
Это я.

Я иногда удачно попадаю по клавиатуре

Это Postgres.

Postgres – это СУБД.

Я использую картинки interdb.jp и мне – можно.



Use of https://www.interdb.jp/pg materials in my conference speak.

Kirill Reshke Thu 6 Mar, 09:09 (1 day ago)

Hi! I am going to give a talk on March 22, 2025 in Moscow, Russia[1]. This talk will be about our Cloudberry/Greenplum contributions that we (me and my colleagu

Hironobu Suzuki Thu 6 Mar, 10:30 (1 day ago)

OK. 2025年3月6日(木) 15:09 Kirill Reshke <reshkekirill@gmail.com>:

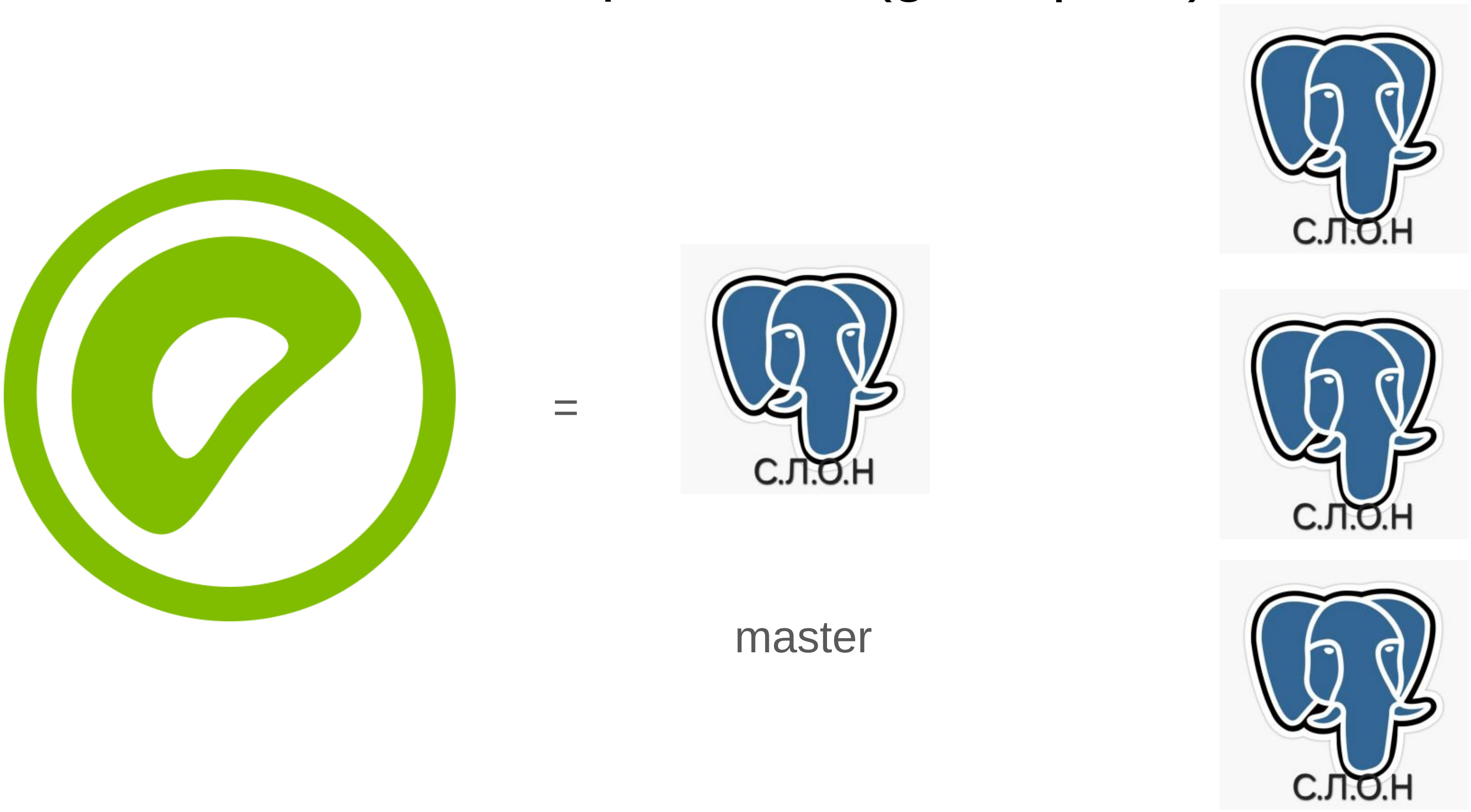
Hironobu Suzuki Thu 6 Mar, 10:34 (1 day ago)

to me

OK = Okay = Allow to use

2025年3月6日(木) 16:30 Hironobu Suzuki <interdb.mx@gmail.com>:

Это гринплам (greenplum)



master

seg1

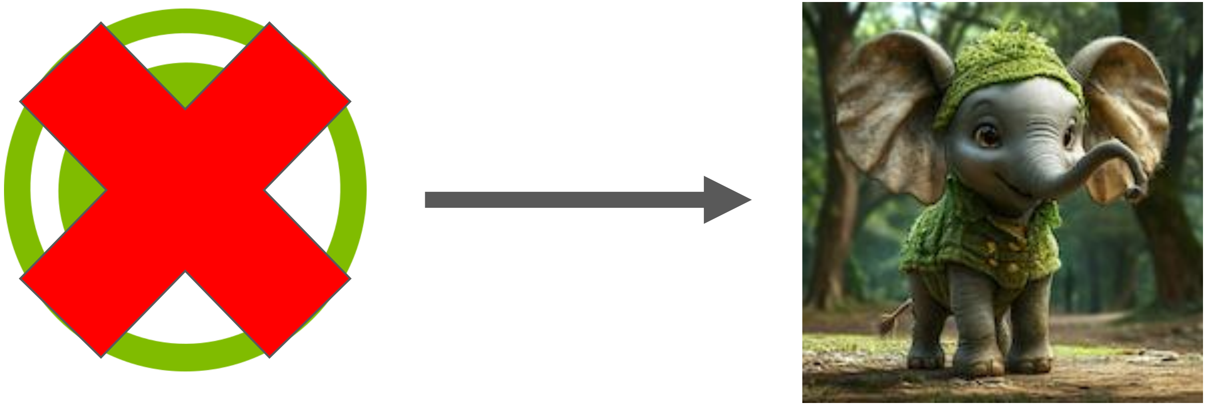
seg2

seg3

Что случилось в мае 24?

=

25.05.2024 закрыли opensource Greenplum



open-gpdb

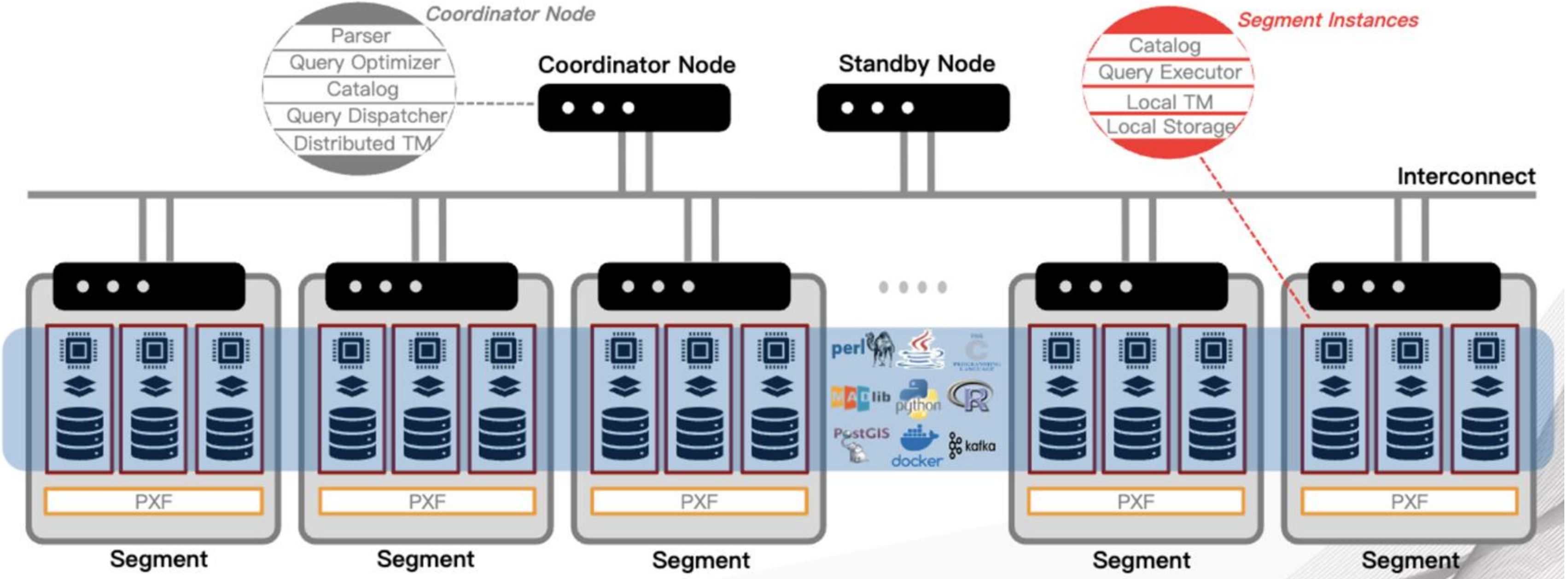
Но ноутбуки у нас не отобрали

open-gpdb

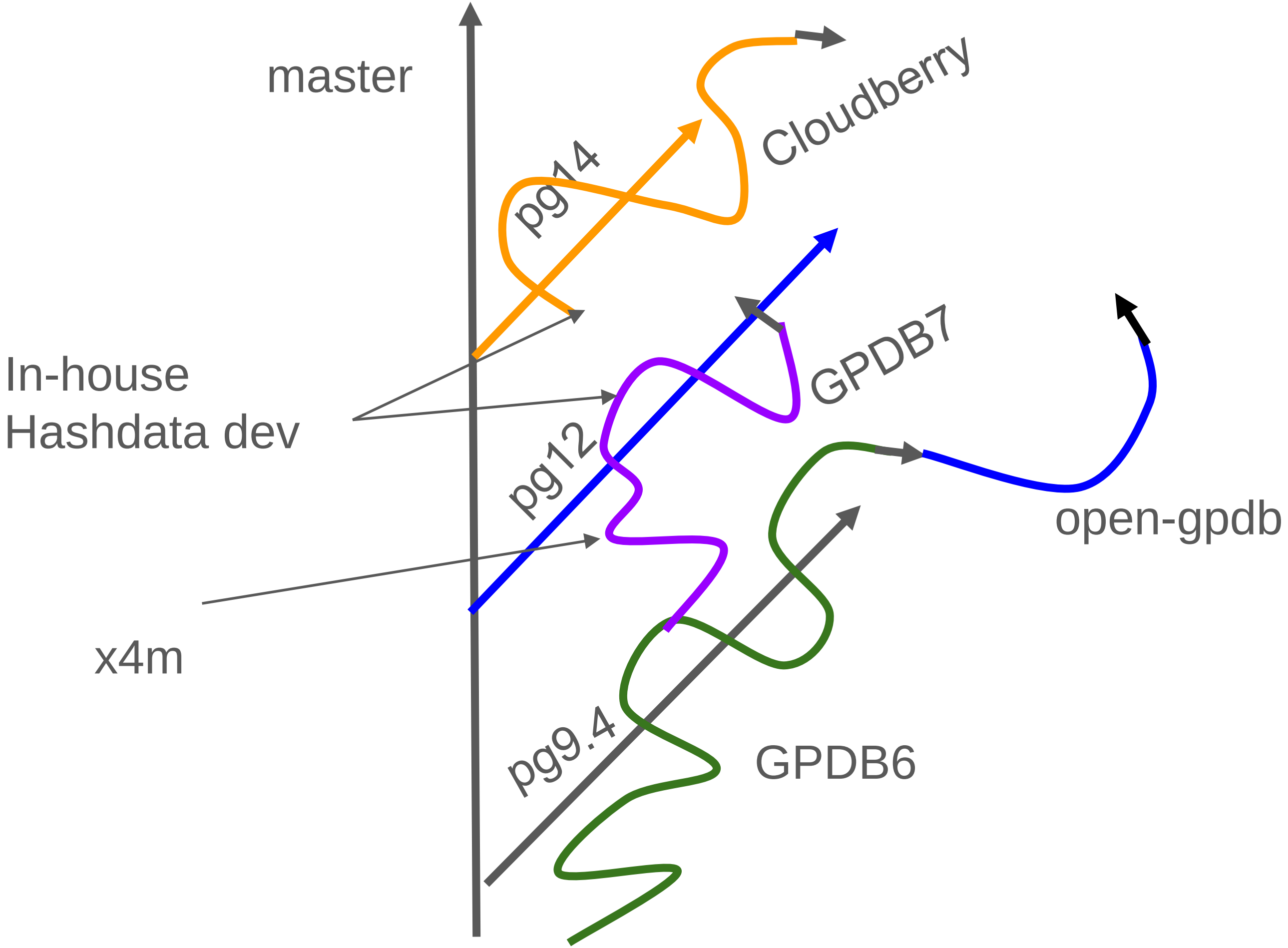
CLOUDBERRY
DATABASE

Что это?

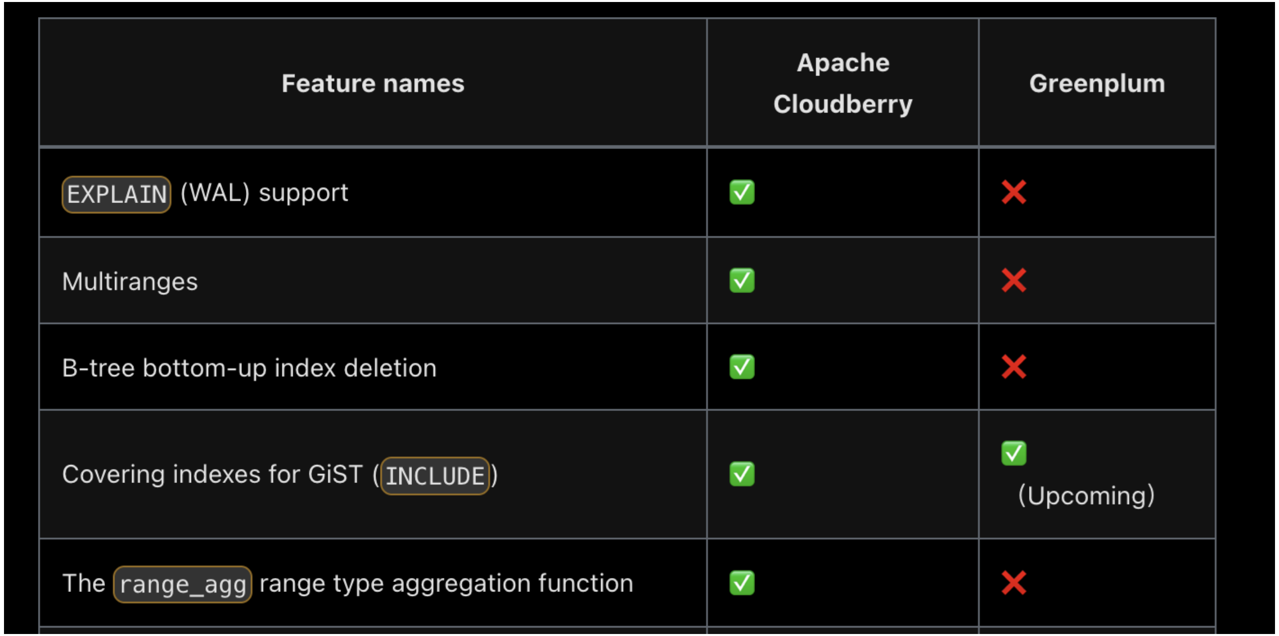
Cloudberry?



Cloudberry Database: MPP Shared–Nothing Architecture, fully integrated with PostgreSQL 14.4


master
pg14
Cloudberry
In-house
Hashdata dev
GPDB7
pg12
open-gpdb
x4m
pg9.4
GPDB6

В чем разница?



Feature names	Apache Cloudberry	Greenplum
`EXPLAIN` (WAL) support	✅	❌
Multiranges	✅	❌
B-tree bottom-up index deletion	✅	❌
Covering indexes for GiST (`INCLUDE`)	✅	✅ (Upcoming)
The `range_agg` range type aggregation function	✅	❌

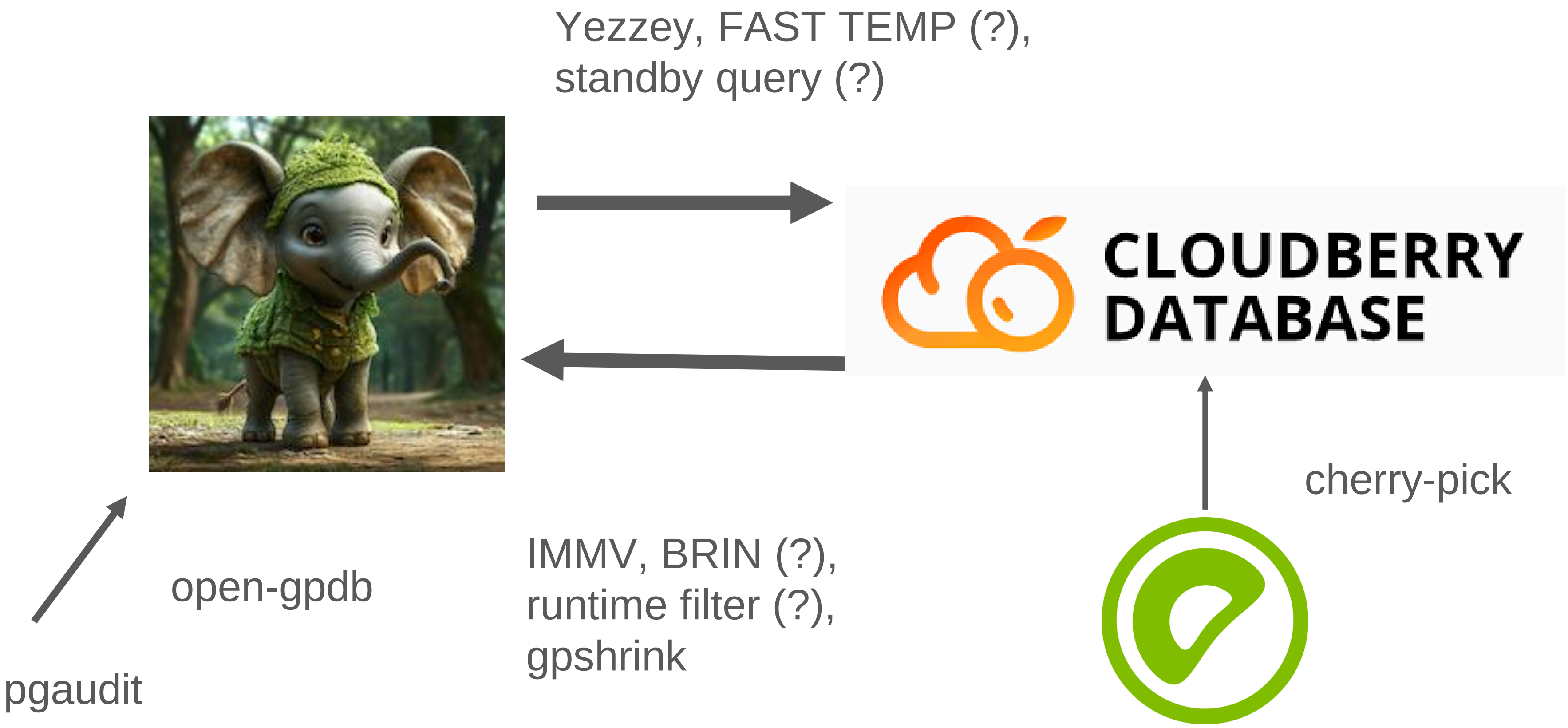
https://cloudberry.apache.org/docs/cbdb-vs-gp-features/

Что из этого нам интересно

- Runtime filter (!ABI)
- Indexes for AO (!ABI)
- Query parallelism (!ABI)
- FAST ANALYZE
- Create index progress (!ABI)

https://cloudberry.apache.org/docs/cbdb-vs-gp-features/

Абаюдная разработка



Yezzey, FAST TEMP (?),
standby query (?)

open-gpdb

pgaudit

IMMV, BRIN (?),
runtime filter (?),
gpshrink

cherry-pick

Как из гп сделать клаудберри?
Как из одного postgres перетащить фичу в другой?

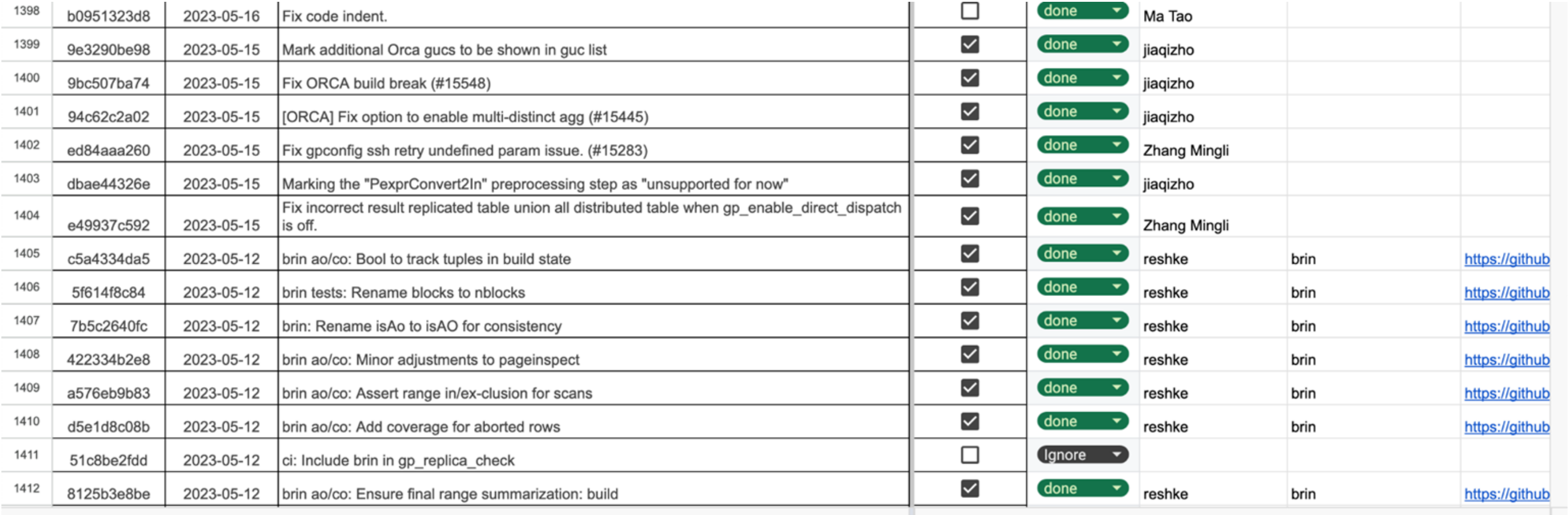
I. Cherry-pick

II. Просто написать код под 6 gp (отличается от п.1?)

III. pg_upgrade/gp_upgrade

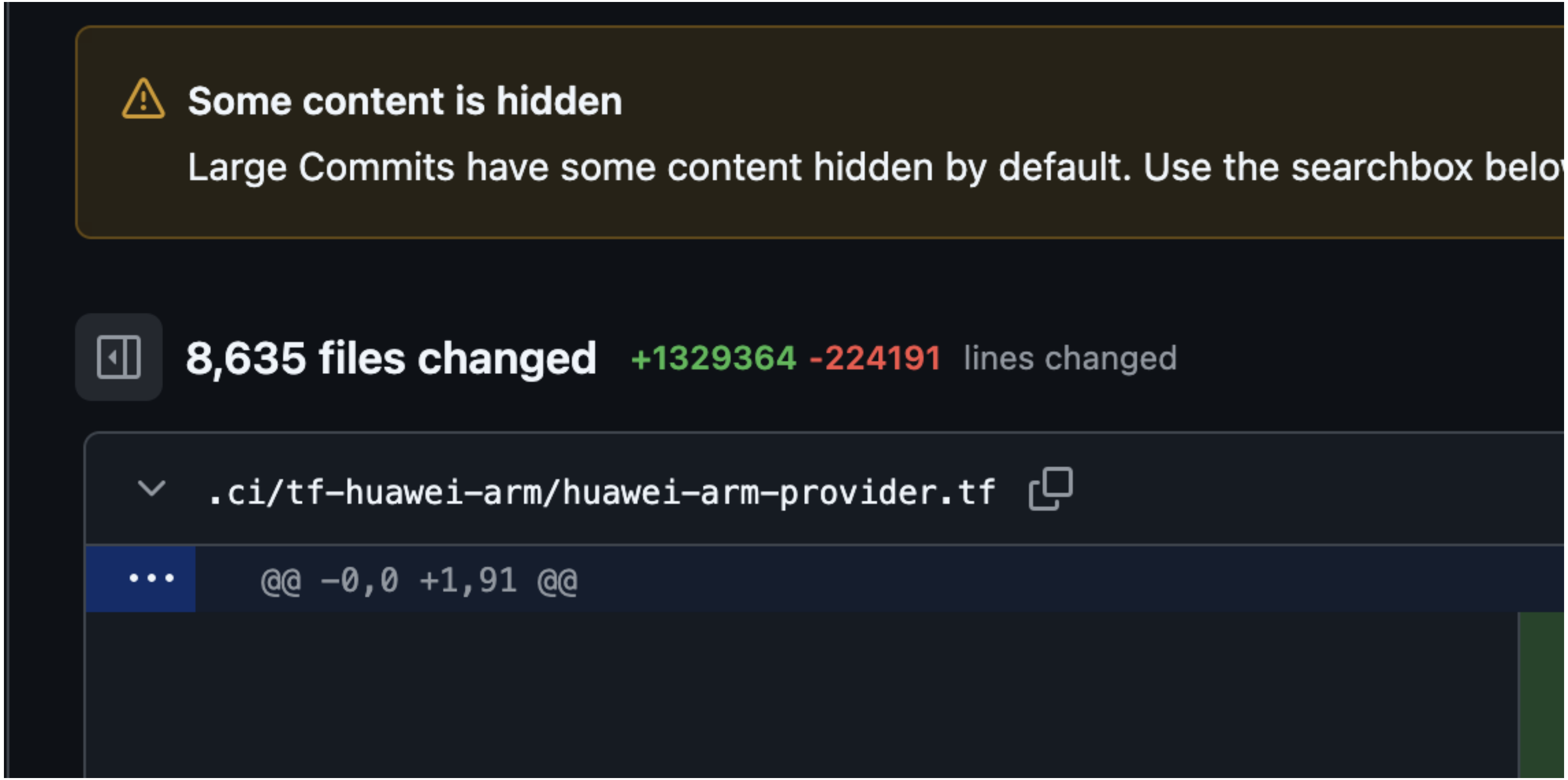
Надо ли все это делать?

Как выглядит работа с cherry-pick



1398	b0951323d8	2023-05-16	Fix code indent.	☐	done	Ma Tao		
1399	9e3290be98	2023-05-15	Mark additional Orca gucs to be shown in guc list	☑	done	jiaqizho		
1400	9bc507ba74	2023-05-15	Fix ORCA build break (#15548)	☑	done	jiaqizho		
1401	94c62c2a02	2023-05-15	[ORCA] Fix option to enable multi-distinct agg (#15445)	☑	done	jiaqizho		
1402	ed84aaa260	2023-05-15	Fix gpconfig ssh retry undefined param issue. (#15283)	☑	done	Zhang Mingli		
1403	dbae44326e	2023-05-15	Marking the "PexprConvert2In" preprocessing step as "unsupported for now"	☑	done	jiaqizho		
1404	e49937c592	2023-05-15	Fix incorrect result replicated table union all distributed table when gp_enable_direct_dispatch is off.	☑	done	Zhang Mingli		
1405	c5a4334da5	2023-05-12	brin ao/co: Bool to track tuples in build state	☑	done	reshke	brin	https://github
1406	5f614f8c84	2023-05-12	brin tests: Rename blocks to nblocks	☑	done	reshke	brin	https://github
1407	7b5c2640fc	2023-05-12	brin: Rename isAo to isAO for consistency	☑	done	reshke	brin	https://github
1408	422334b2e8	2023-05-12	brin ao/co: Minor adjustments to pageinspect	☑	done	reshke	brin	https://github
1409	a576eb9b83	2023-05-12	brin ao/co: Assert range in/ex-clusion for scans	☑	done	reshke	brin	https://github
1410	d5e1d8c08b	2023-05-12	brin ao/co: Add coverage for aborted rows	☑	done	reshke	brin	https://github
1411	51c8be2fdd	2023-05-12	ci: Include brin in gp_replica_check	☐	Ignore			
1412	8125b3e8be	2023-05-12	brin ao/co: Ensure final range summarization: build	☑	done	reshke	brin	https://github

Как выглядит работа с cherry-pick



⚠ **Some content is hidden**

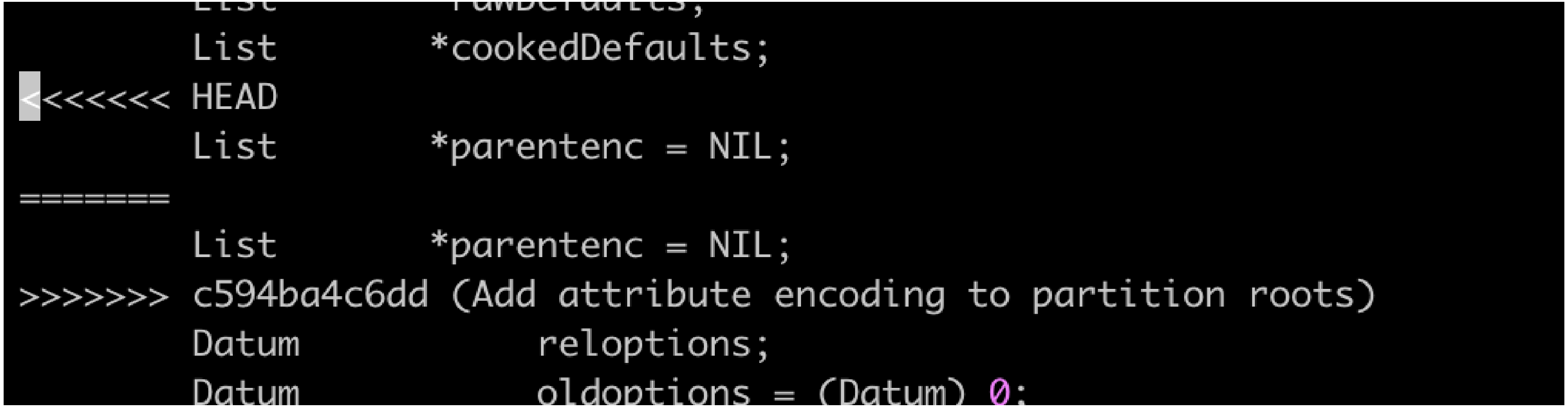
Large Commits have some content hidden by default. Use the searchbox belo

8,635 files changed +1329364 -224191 lines changed

```
.ci/tf-huawei-arm/huawei-arm-provider.tf
```

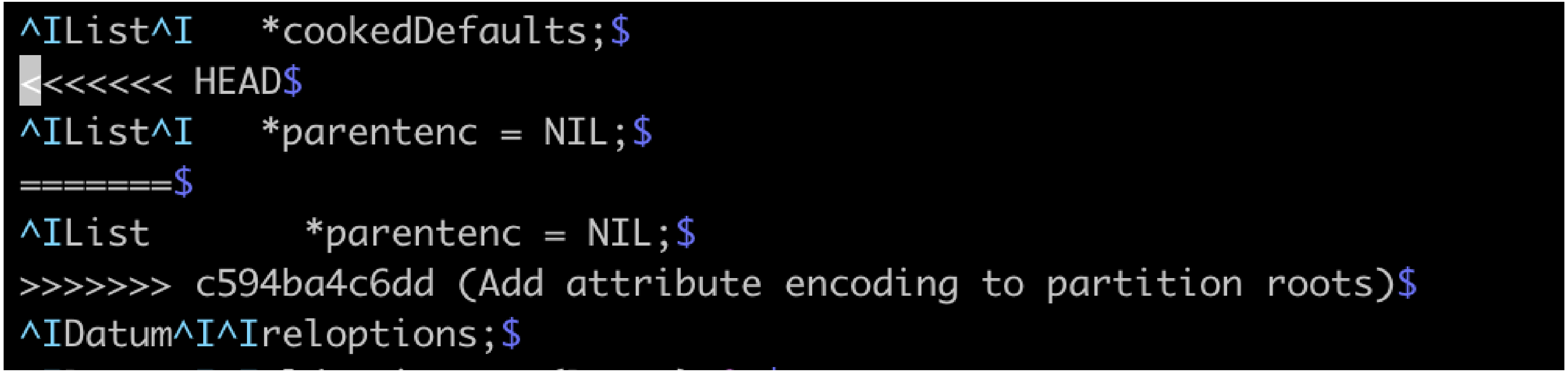
```
@@ -0,0 +1,91 @@
```

Tabs vs spaces

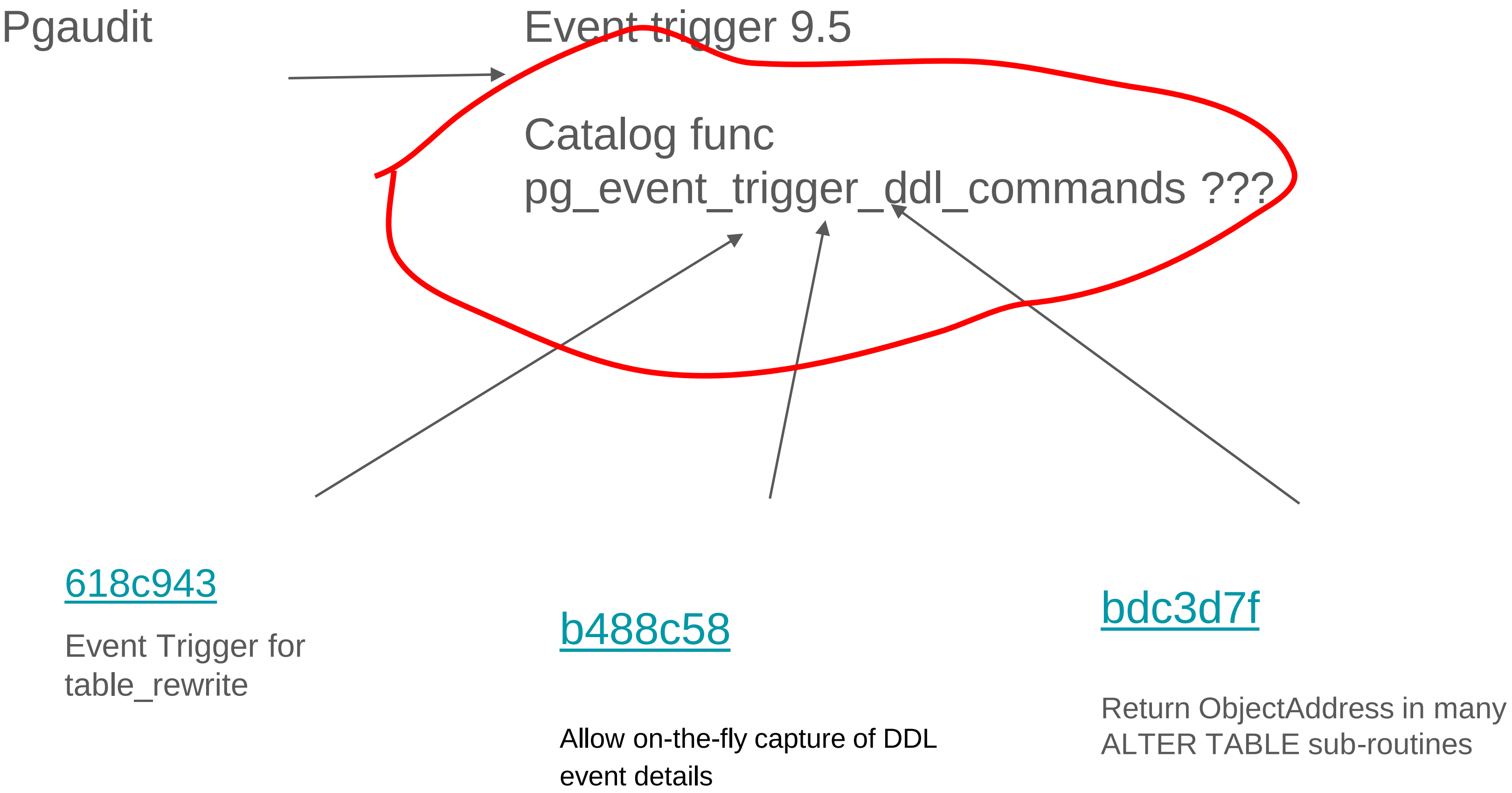


```
        List       *rawDefaults;
        List       *cookedDefaults;
<<<<<<< HEAD
        List       *parentenc = NIL;
=======
        List       *parentenc = NIL;
>>>>>>> c594ba4c6dd (Add attribute encoding to partition roots)
        Datum           reloptions;
        Datum           oldoptions = (Datum) 0;
```

Tabs vs spaces



```
^IList^I   *cookedDefaults;$
<<<<<<< HEAD$
^IList^I   *parentenc = NIL;$
=======$
^IList       *parentenc = NIL;$
>>>>>>> c594ba4c6dd (Add attribute encoding to partition roots)$
^IDatum^I^Ireloptions;$
```



Pgaudit

Event trigger 9.5

Catalog func pg_event_trigger_ddl_commands ???

618c943

Event Trigger for table_rewrite

b488c58

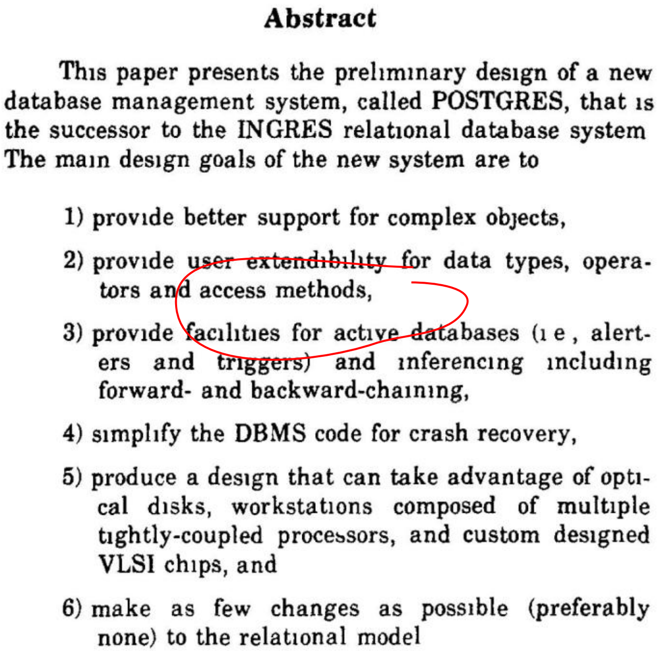
Allow on-the-fly capture of DDL event details

bdc3d7f

Return ObjectAddress in many ALTER TABLE sub-routines

Насколько PostgreSQL расширяемый?

Какие вещи нужно коммитить в ядро, а какие писать сбоку?



Abstract

Thıs paper presents the prelımınary desıgn of a new database management system, called POSTGRES, that ıs the successor to the INGRES relatıonal database system The maın desıgn goals of the new system are to

1) provıde better support for complex objects,

2) provıde user extendıbılıty for data types, operators and access methods,

3) provıde facılıtıes for actıve databases (ı e , alerters and triggers) and ınferencıng ıncludıng forward- and backward-chaınıng,

4) sımplıfy the DBMS code for crash recovery,

5) produce a desıgn that can take advantage of optıcal dısks, workstatıons composed of multıple tıghtly-coupled processors, and custom desıgned VLSI chıps, and

6) make as few changes as possıble (preferably none) to the relatıonal model

Насколько PostgreSQL расширяемый?

PostgeSQL 9.6 – index access method

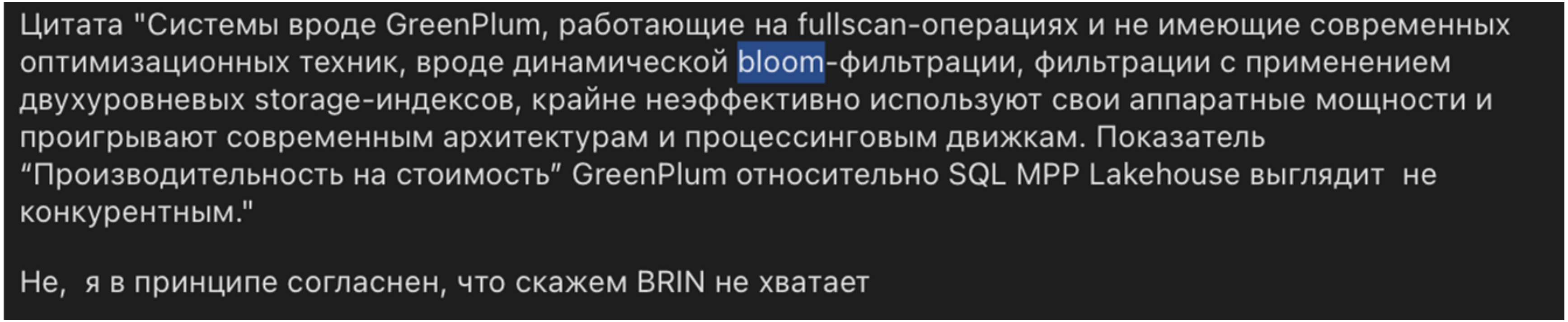
PostgreSQL 12 – table access method

PostgreSQL 15 – custom rmgr

CSN - WIP in pgsql-hackers

Greenplum as extension? Но в open-gpdb postgresql 9.4

Давайте попробуем сделать свой индекс

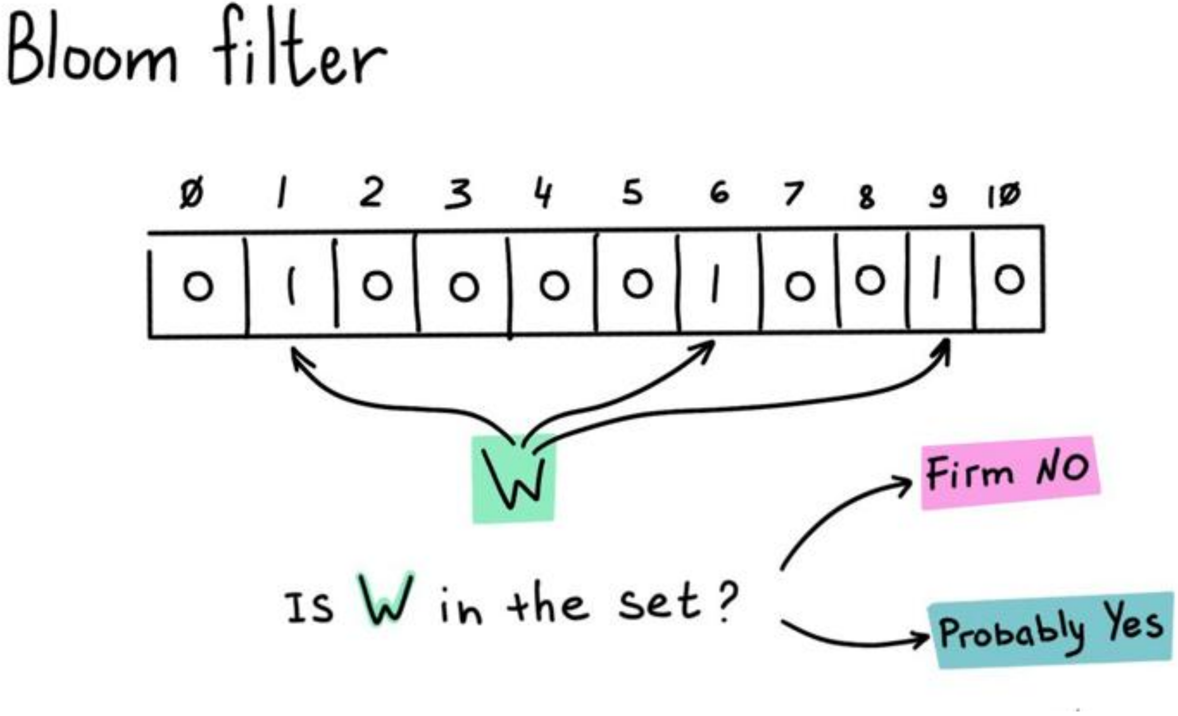


Цитата "Системы вроде GreenPlum, работающие на fullscan-операциях и не имеющие современных оптимизационных техник, вроде динамической bloom-фильтрации, фильтрации с применением двухуровневых storage-индексов, крайне неэффективно используют свои аппаратные мощности и проигрывают современным архитектурам и процессинговым движкам. Показатель “Производительность на стоимость” GreenPlum относительно SQL MPP Lakehouse выглядит не конкурентным."

Не, я в принципе согласнен, что скажем BRIN не хватает

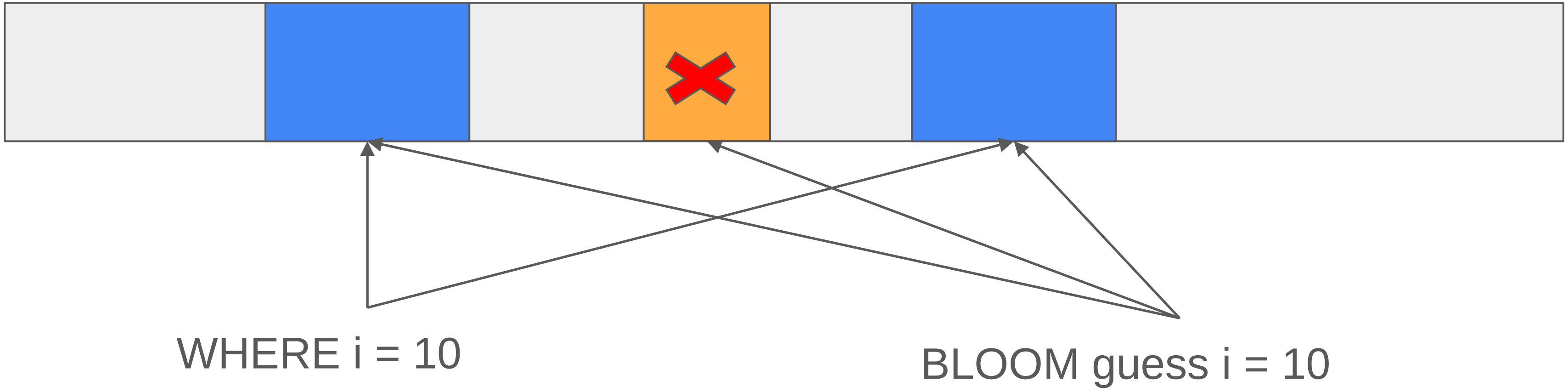
Greenplum Russia

Кто такой bloom filter

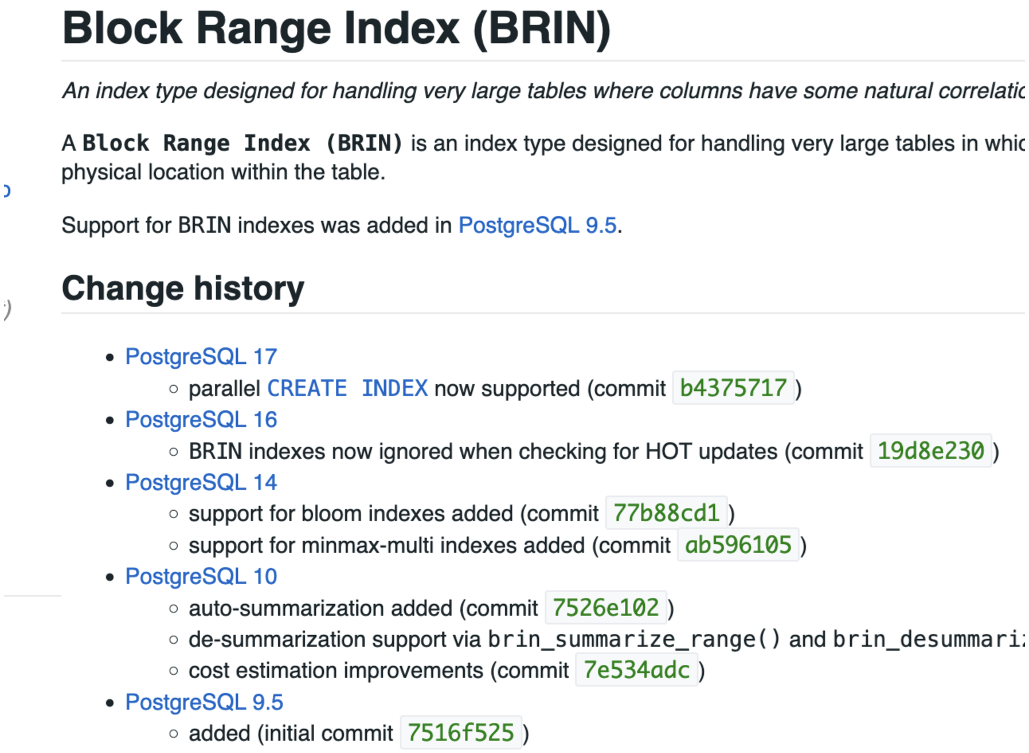


Bloom filter ускоряет запросы по данным?

Точно? Бенчи через ~30 слайдов



Bloom filter есть в PostgreSQL/Greenplum?



Block Range Index (BRIN)

An index type designed for handling very large tables where columns have some natural correlatic

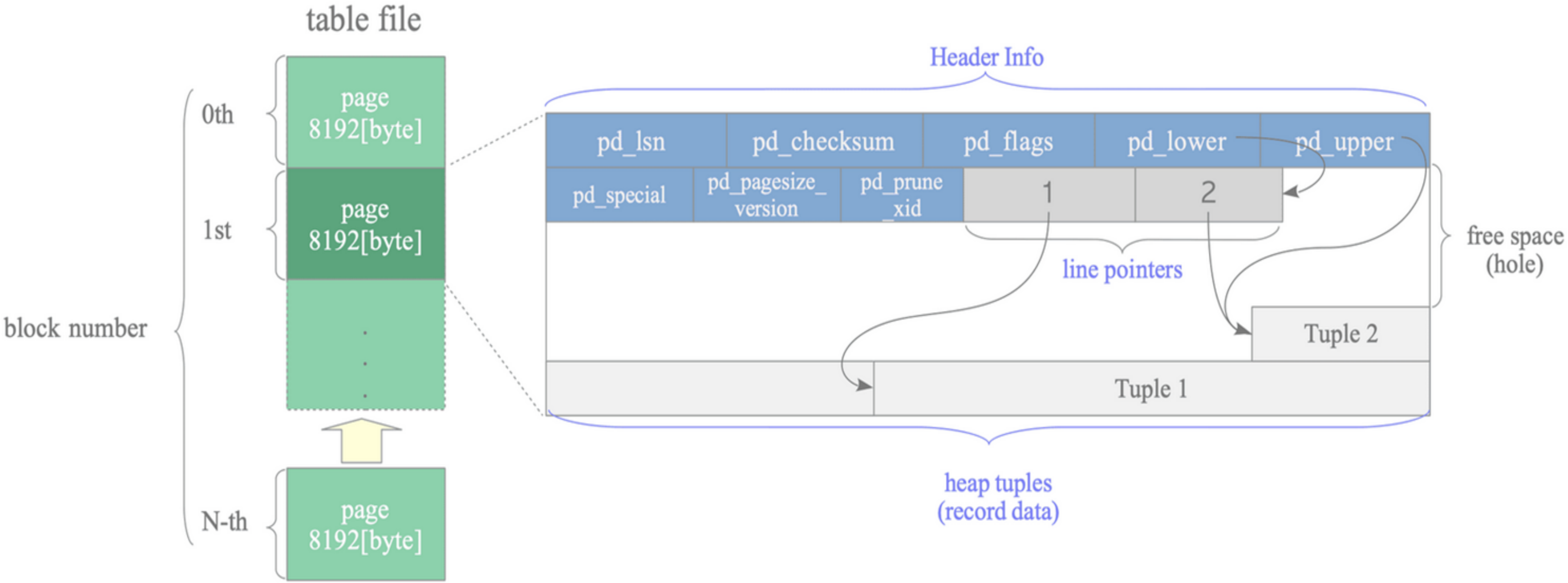
A **`Block Range Index (BRIN)`** is an index type designed for handling very large tables in whic
physical location within the table.

Support for BRIN indexes was added in PostgreSQL 9.5.

Change history

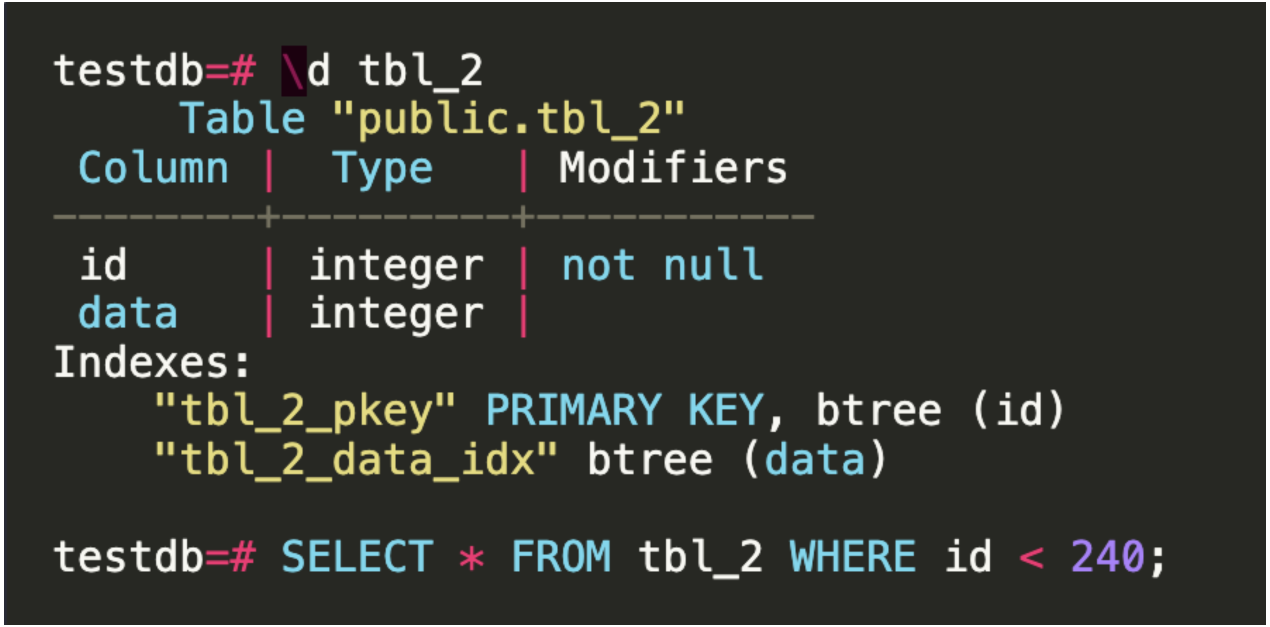
- PostgreSQL 17
  - parallel `CREATE INDEX` now supported (commit `b4375717`)
- PostgreSQL 16
  - BRIN indexes now ignored when checking for HOT updates (commit `19d8e230`)
- PostgreSQL 14
  - support for bloom indexes added (commit `77b88cd1`)
  - support for minmax-multi indexes added (commit `ab596105`)
- PostgreSQL 10
  - auto-summarization added (commit `7526e102`)
  - de-summarization support via `brin_summarize_range()` and `brin_desummari`
  - cost estimation improvements (commit `7e534adc`)
- PostgreSQL 9.5
  - added (initial commit `7516f525`)

HEAP в postgresql



Жахнем индекс сбоку?

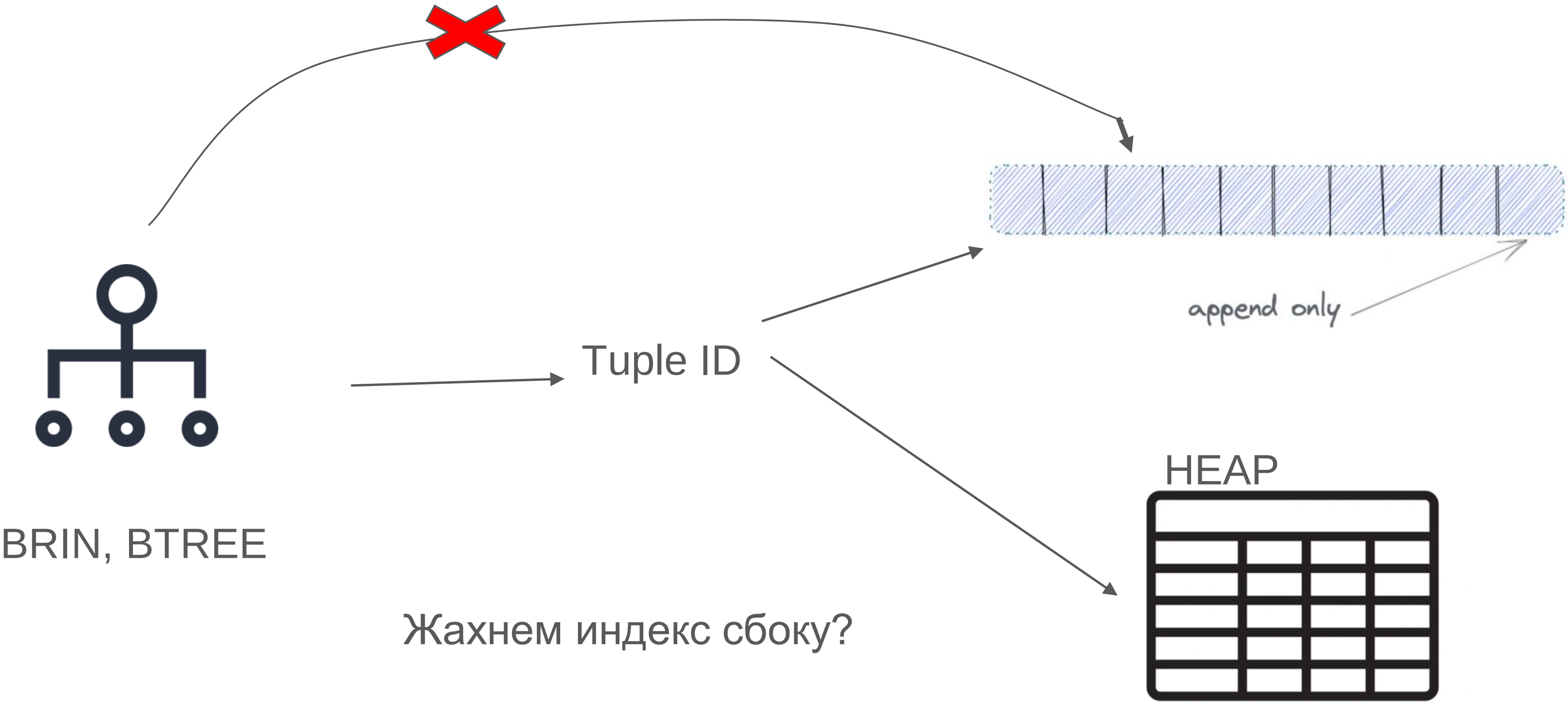
Index = access method for table data



```
testdb=# \d tbl_2
     Table "public.tbl_2"
 Column |  Type   | Modifiers
--------+---------+-----------
 id     | integer | not null
 data   | integer |
Indexes:
    "tbl_2_pkey" PRIMARY KEY, btree (id)
    "tbl_2_data_idx" btree (data)

testdb=# SELECT * FROM tbl_2 WHERE id < 240;
```

Index (predicate) -> list of tids

Index = access method for table data



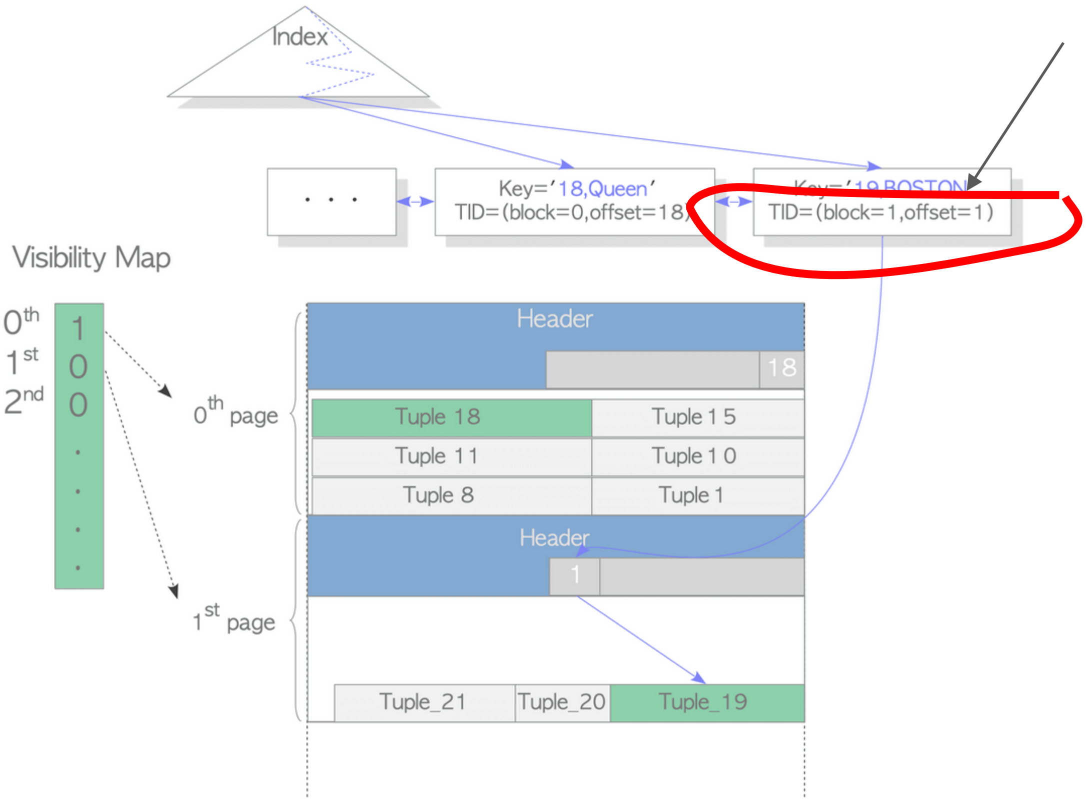
Tuple ID

HEAP

BRIN, BTREE

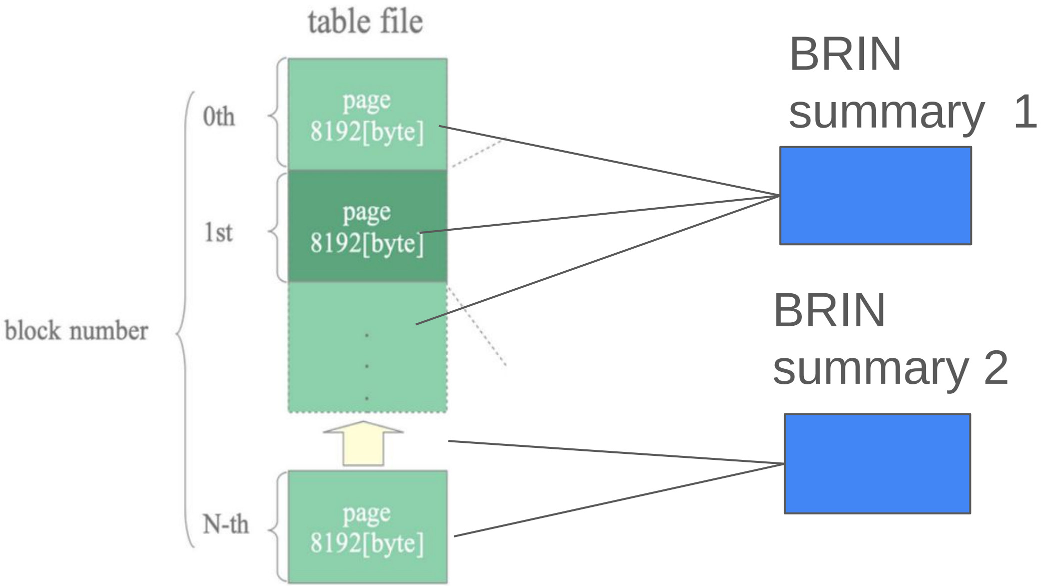
Жахнем индекс сбоку?

Index в обычном heap



Работает математика

Brin index в PostgreSQL

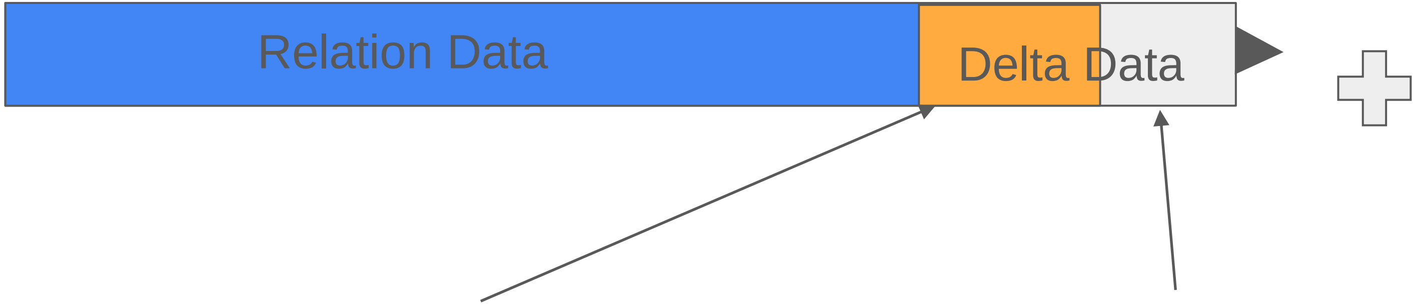


BRIN
summary 1

BRIN
summary 2

Размер много
меньше данных

Устройство Append-Only. Сжатие



3 heap relations

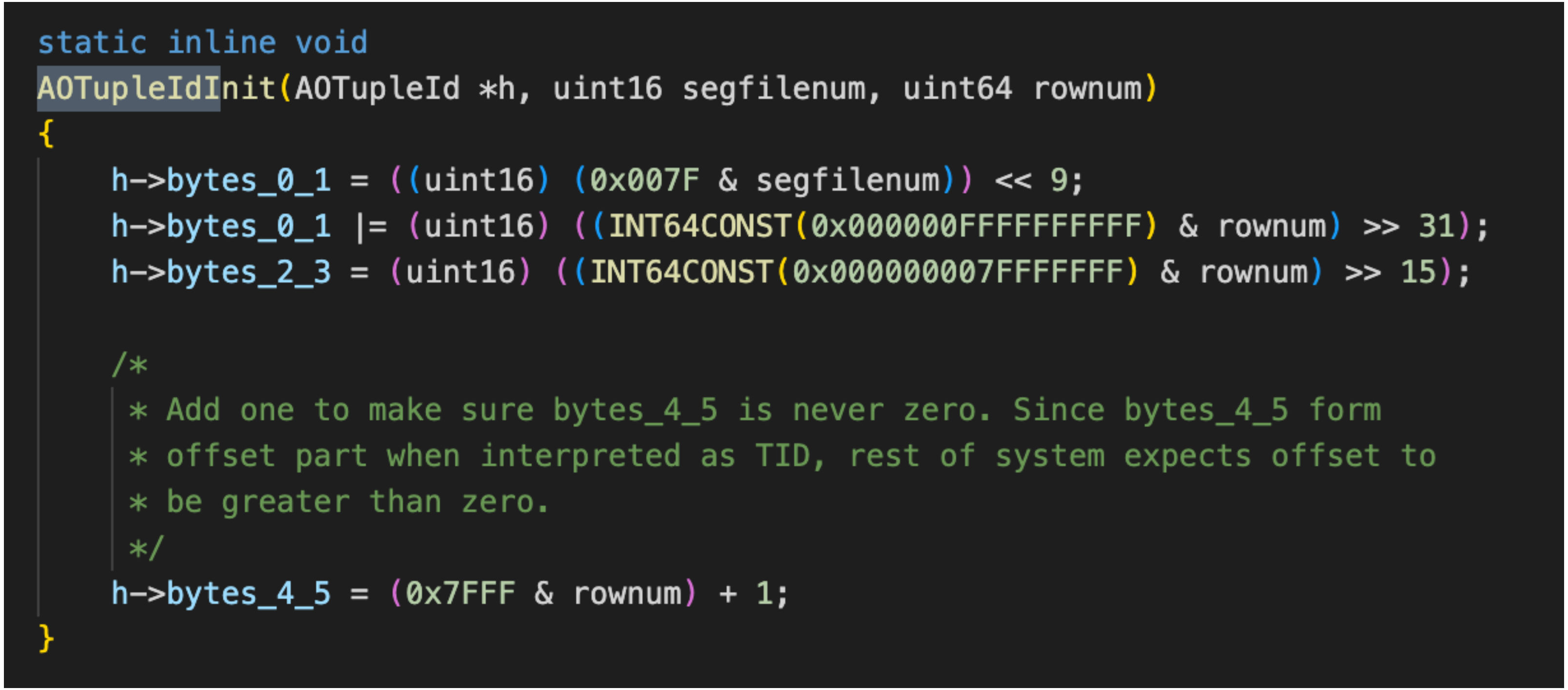
Varblock. Возможно сжатый

Varblock-in-progress

AO tuple id != (heap)TID

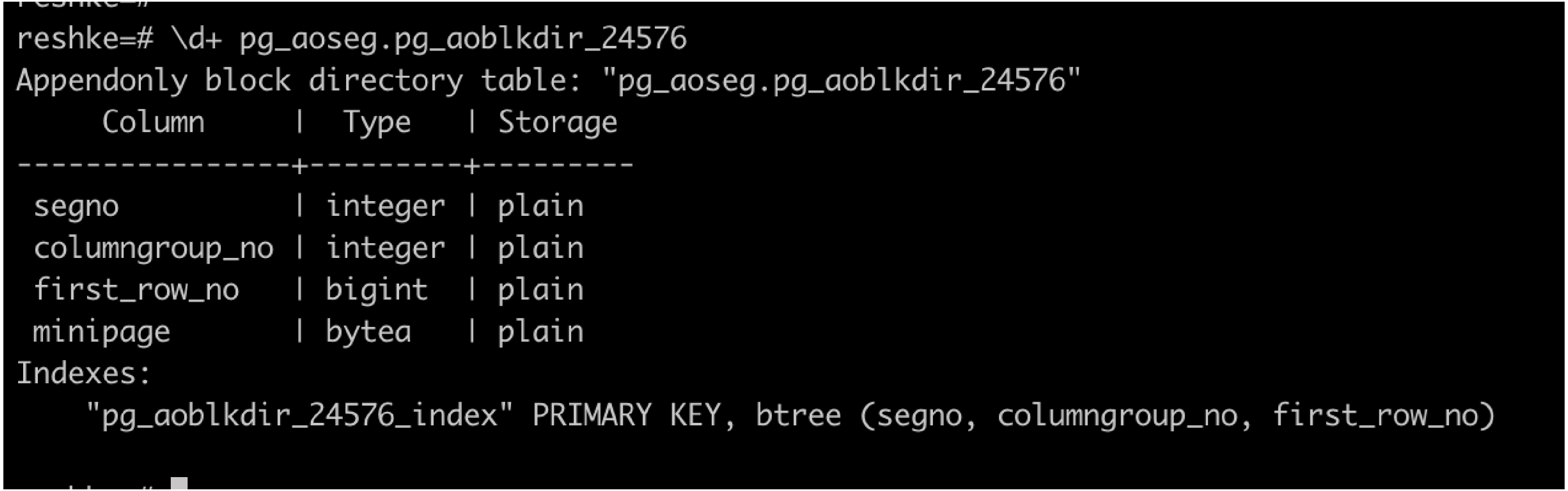
Жахнем индекс сбоку?

Appendonly tuple id



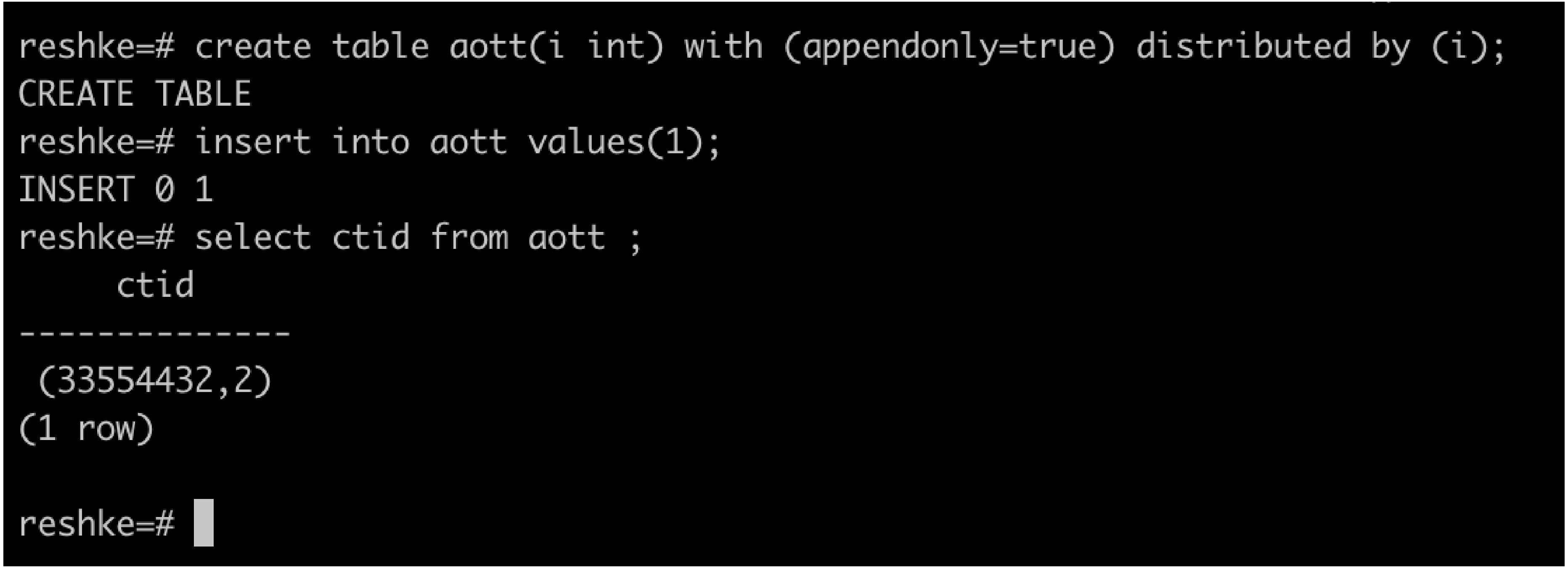
```
static inline void
AOTupleIdInit(AOTupleId *h, uint16 segfilenum, uint64 rownum)
{
    h->bytes_0_1 = ((uint16) (0x007F & segfilenum)) << 9;
    h->bytes_0_1 |= (uint16) ((INT64CONST(0x000000FFFFFFFFFF) & rownum) >> 31);
    h->bytes_2_3 = (uint16) ((INT64CONST(0x000000007FFFFFFF) & rownum) >> 15);

    /*
     * Add one to make sure bytes_4_5 is never zero. Since bytes_4_5 form
     * offset part when interpreted as TID, rest of system expects offset to
     * be greater than zero.
     */
    h->bytes_4_5 = (0x7FFF & rownum) + 1;
}
```

Где начинается нужный TID page?



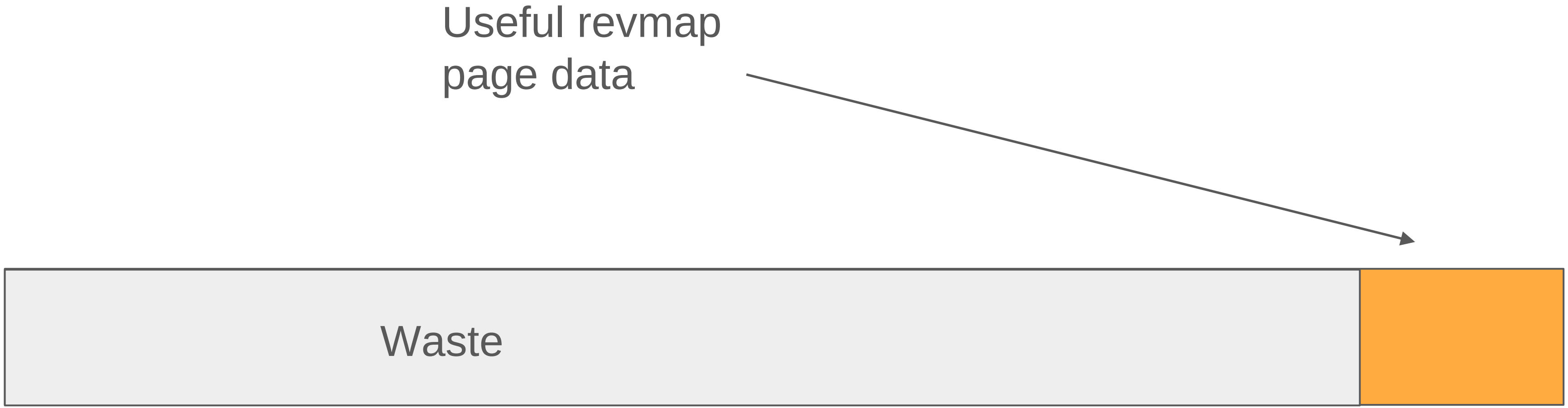
```
reshke=#
reshke=# \d+ pg_aoseg.pg_aoblkdir_24576
Appendonly block directory table: "pg_aoseg.pg_aoblkdir_24576"
     Column      |  Type   | Storage
-----------------+---------+---------
 segno           | integer | plain
 columngroup_no  | integer | plain
 first_row_no    | bigint  | plain
 minipage        | bytea   | plain
Indexes:
    "pg_aoblkdir_24576_index" PRIMARY KEY, btree (segno, columngroup_no, first_row_no)

reshke=#
```

Brin index в greenplum

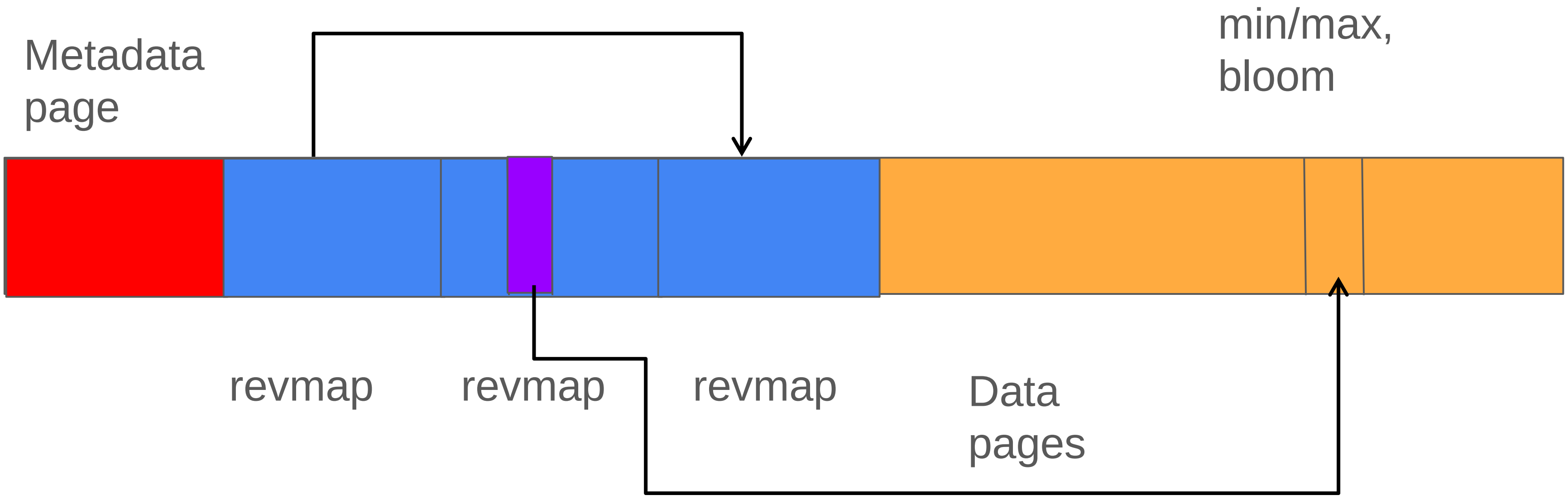


```
reshke=# create table aott(i int) with (appendonly=true) distributed by (i);
CREATE TABLE
reshke=# insert into aott values(1);
INSERT 0 1
reshke=# select ctid from aott ;
     ctid
--------------
 (33554432,2)
(1 row)

reshke=# 
```

Brin index в greenplum



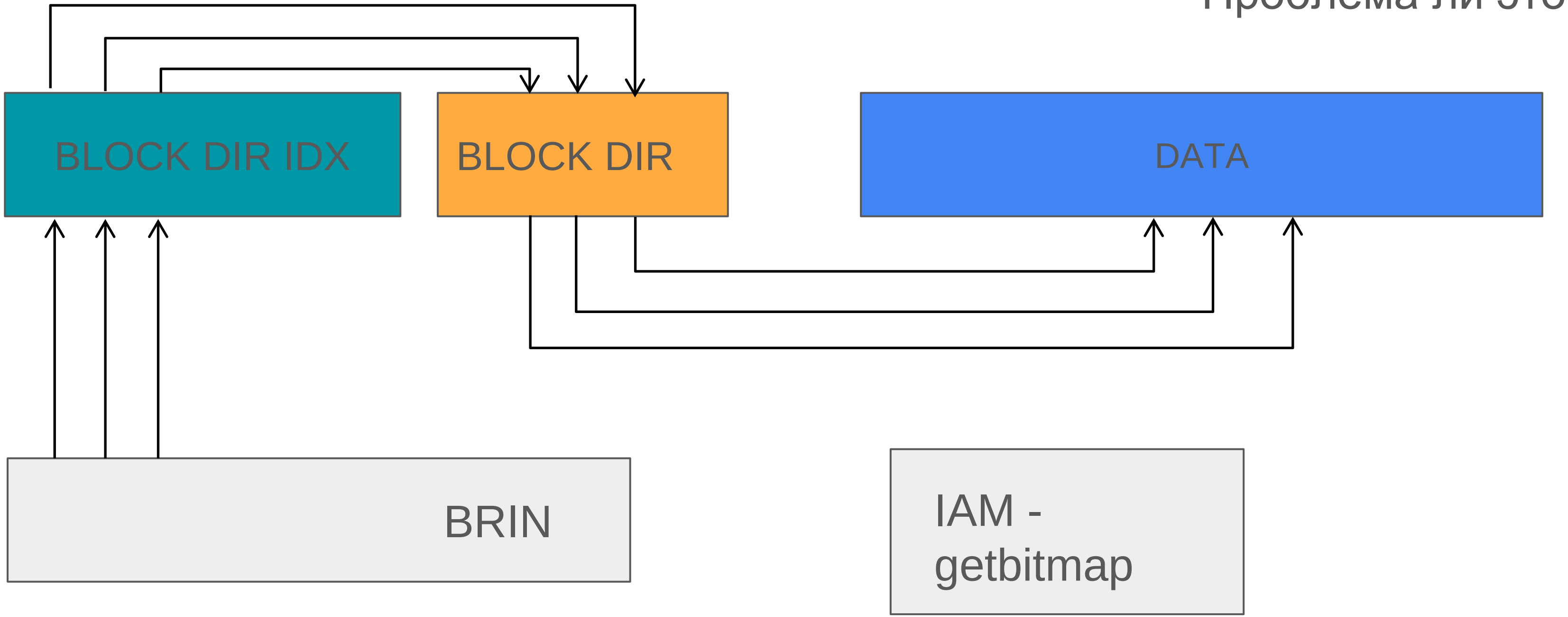
Brin index в greenplum. Revmap struct

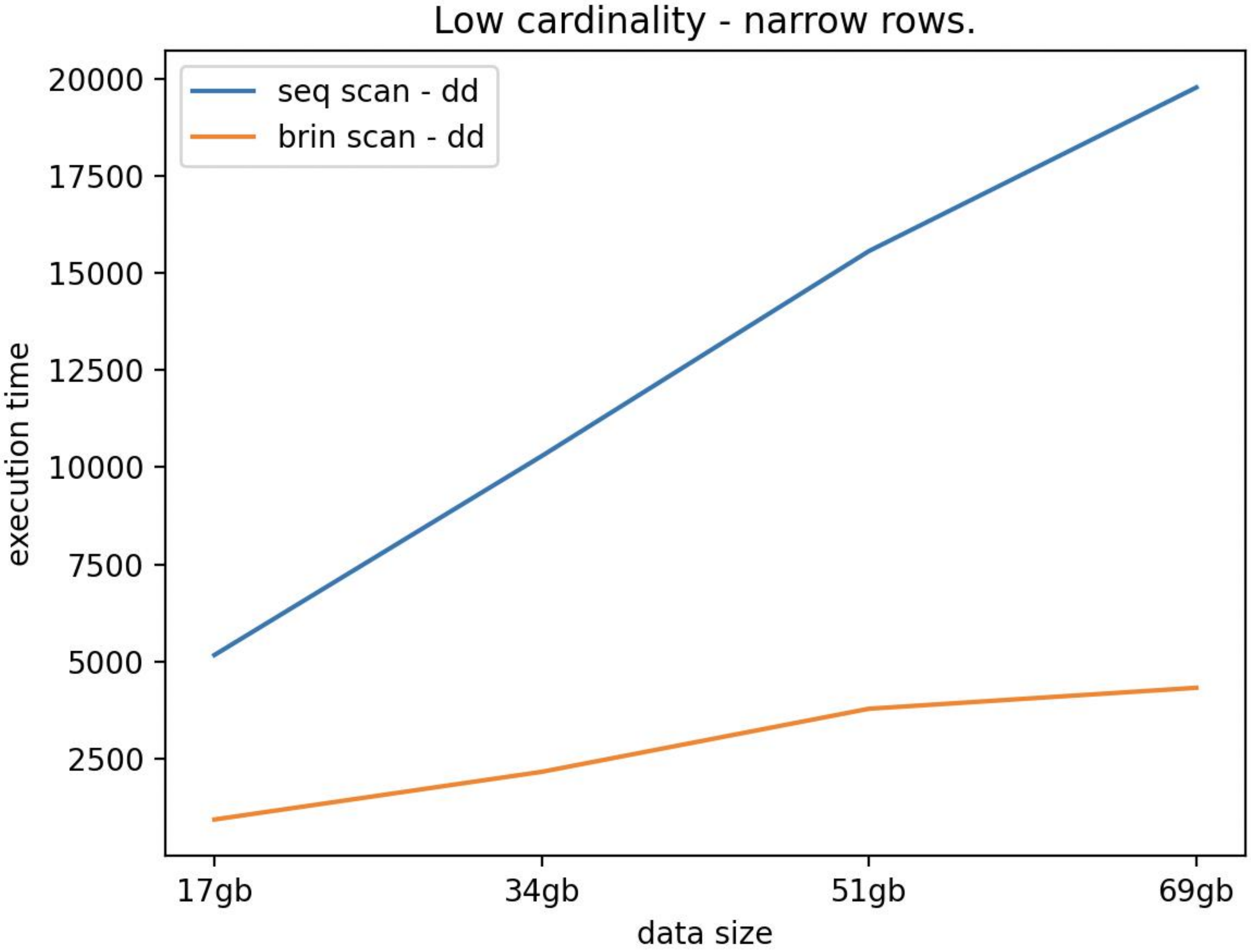


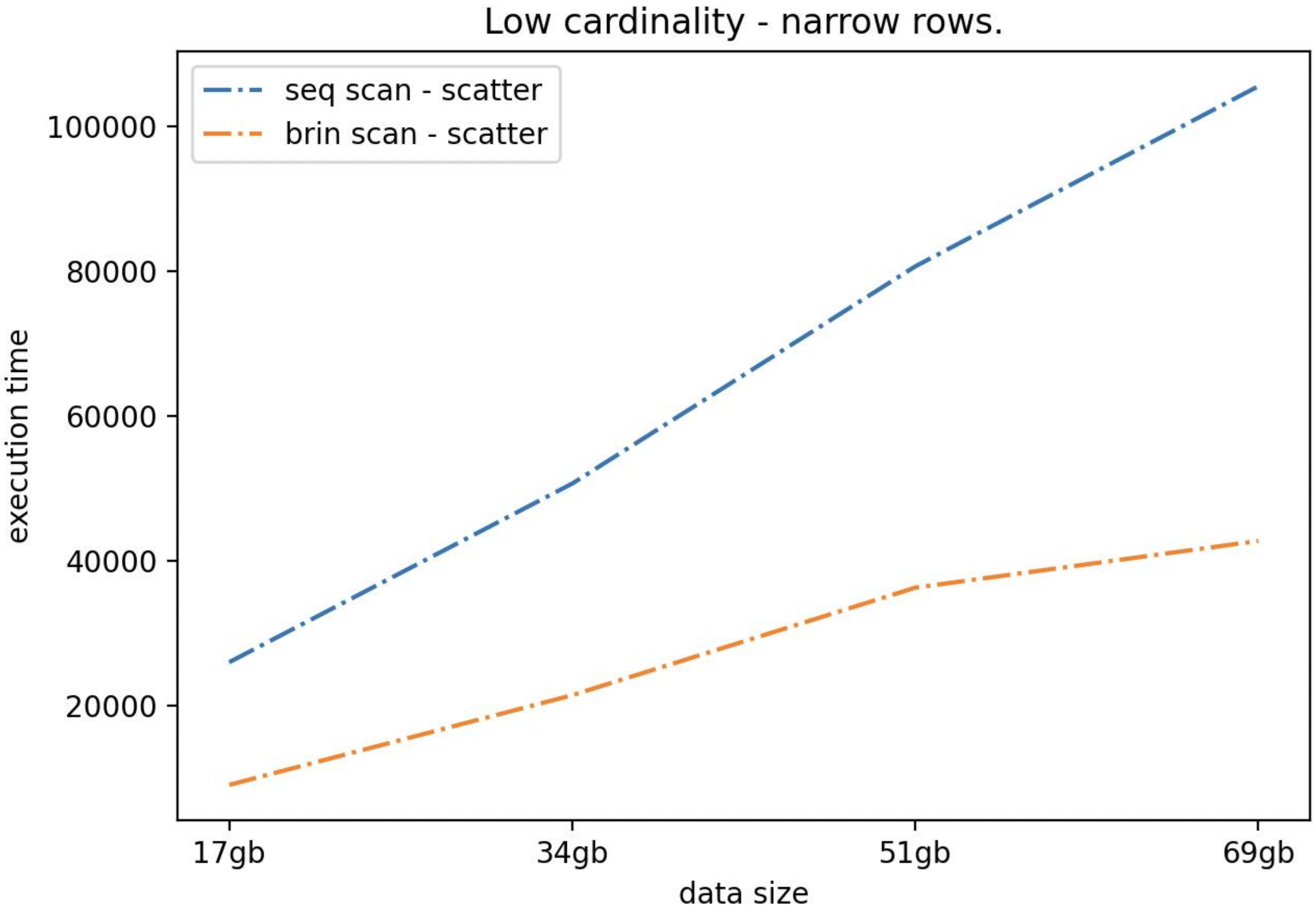
Индексы в Appendonly

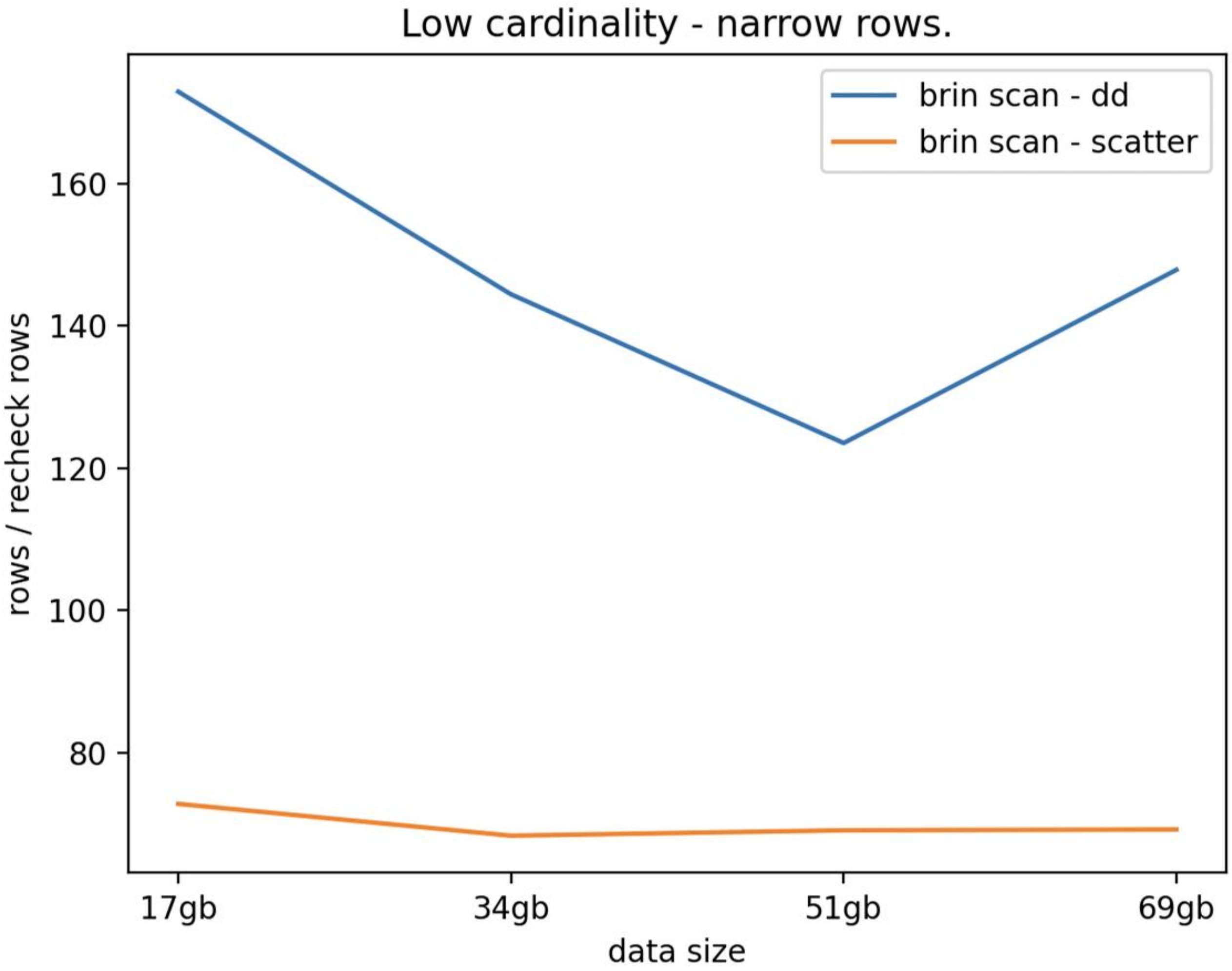
Почему BRIN вообще такой сложный? (потому что PostgreSQL – расширяемый)

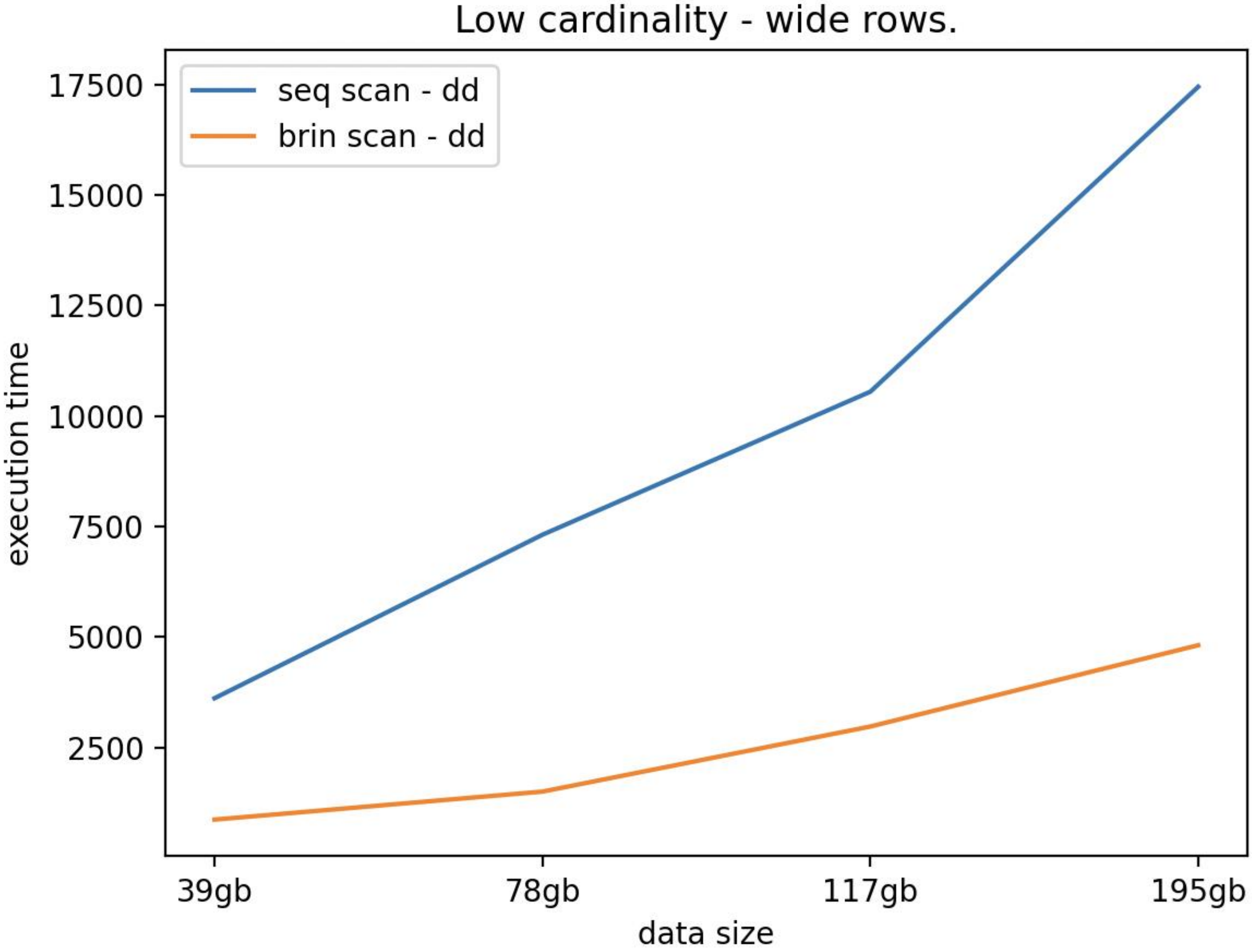
Проблема ли это?

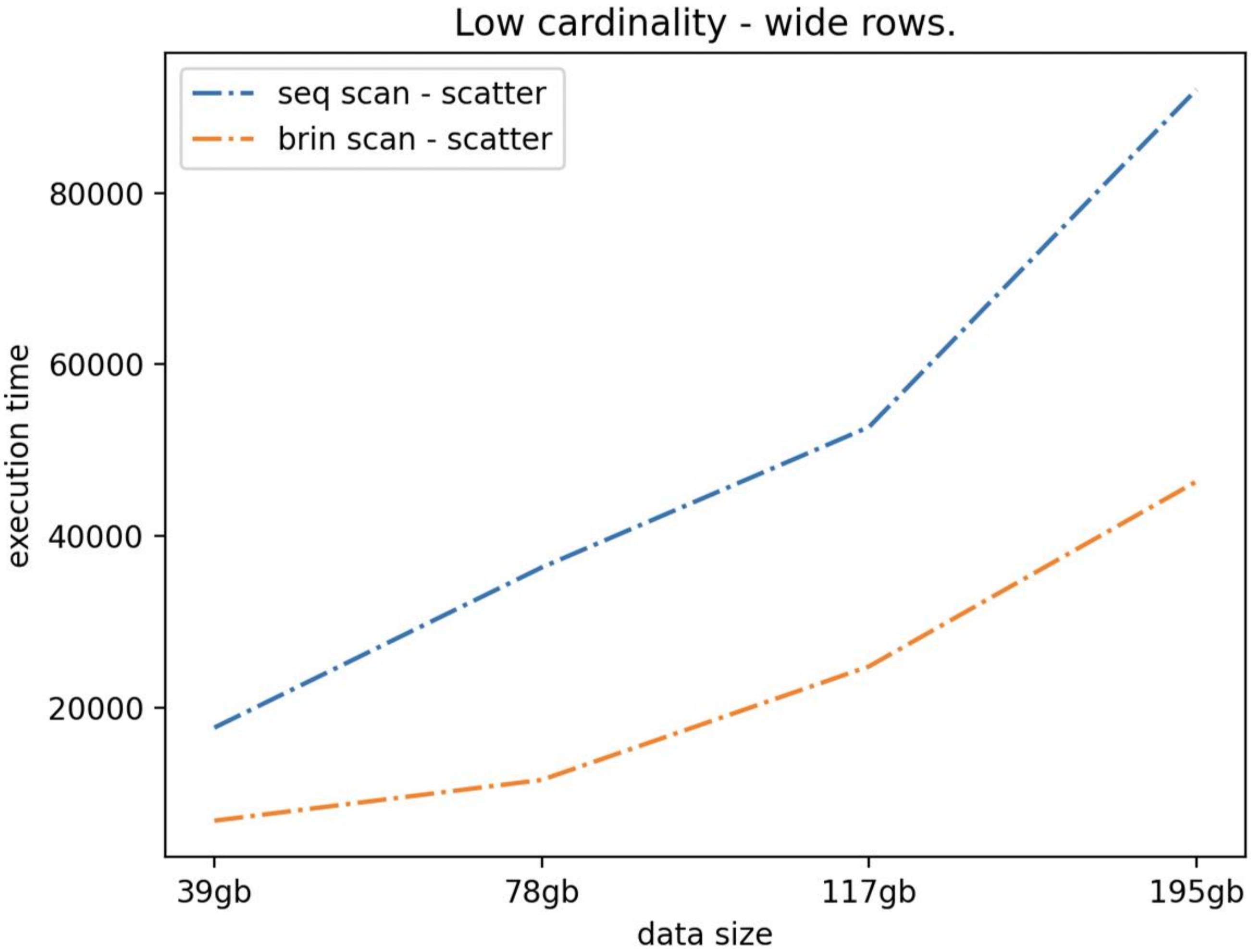


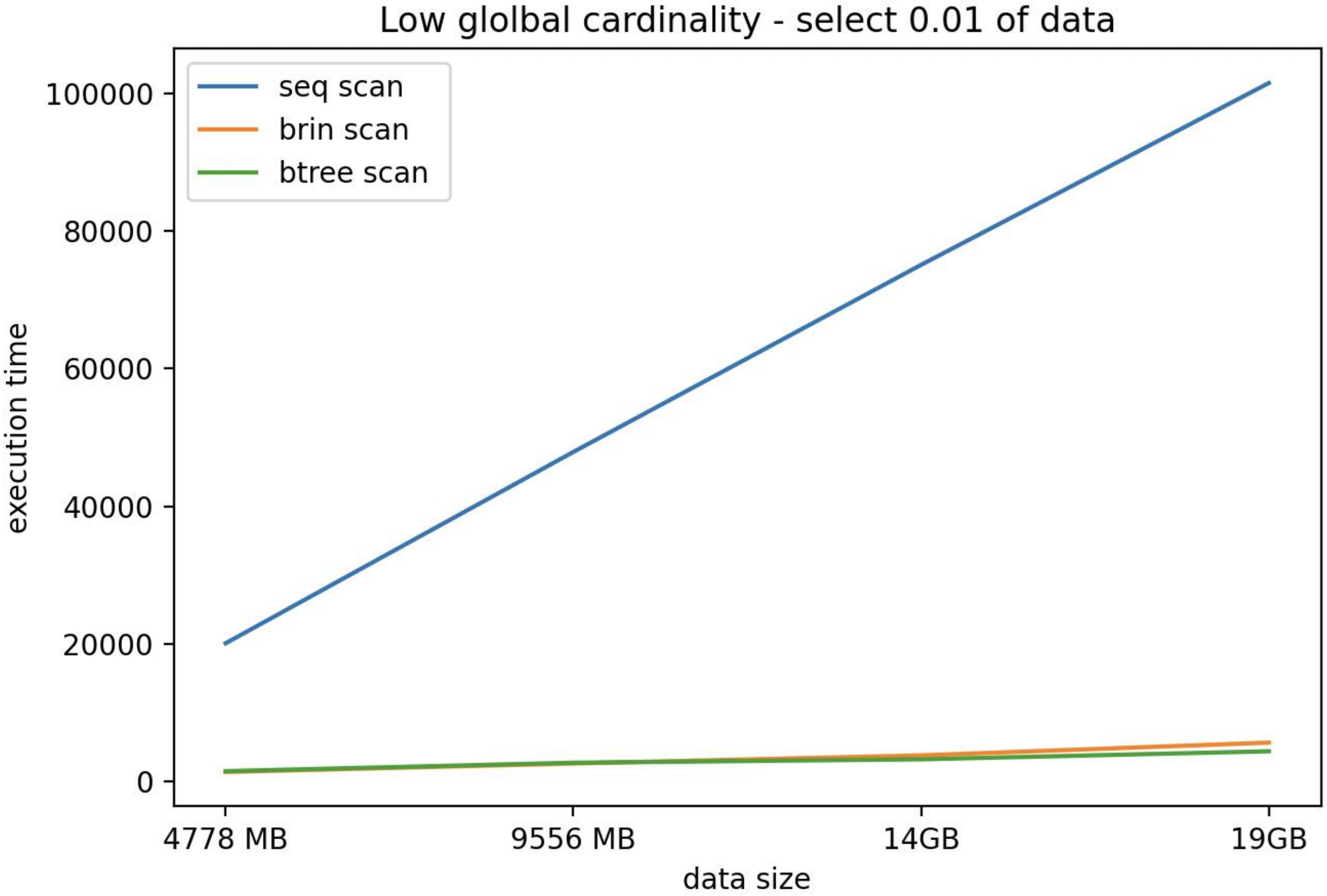

Low cardinality - narrow rows.
seq scan - dd
brin scan - dd
execution time
20000
17500
15000
12500
10000
7500
5000
2500
17gb
34gb
51gb
69gb
data size


Low cardinality - narrow rows.
seq scan - scatter
brin scan - scatter
execution time
100000
80000
60000
40000
20000
17gb
34gb
51gb
69gb
data size

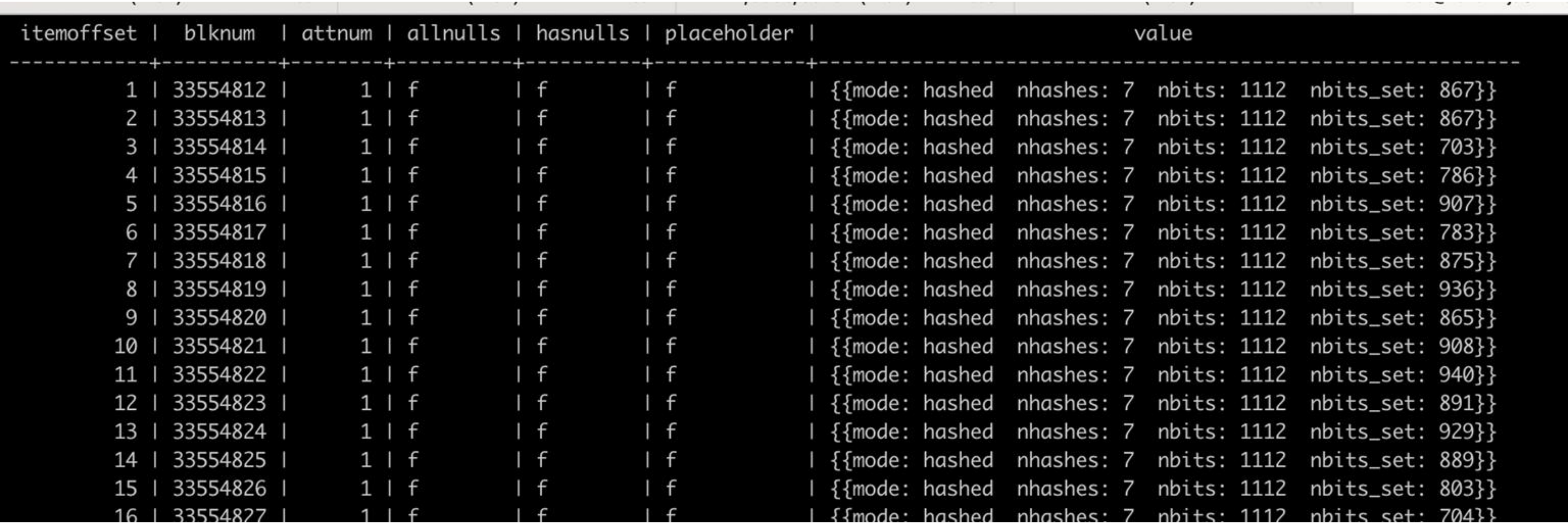

Low cardinality - narrow rows.
brin scan - dd
brin scan - scatter
rows / recheck rows
160
140
120
100
80
17gb
34gb
51gb
69gb
data size


Low cardinality - wide rows.
seq scan - dd
brin scan - dd
execution time
17500
15000
12500
10000
7500
5000
2500
data size
39gb
78gb
117gb
195gb


Low cardinality - wide rows.
seq scan - scatter
brin scan - scatter
execution time
80000
60000
40000
20000
data size
39gb
78gb
117gb
195gb

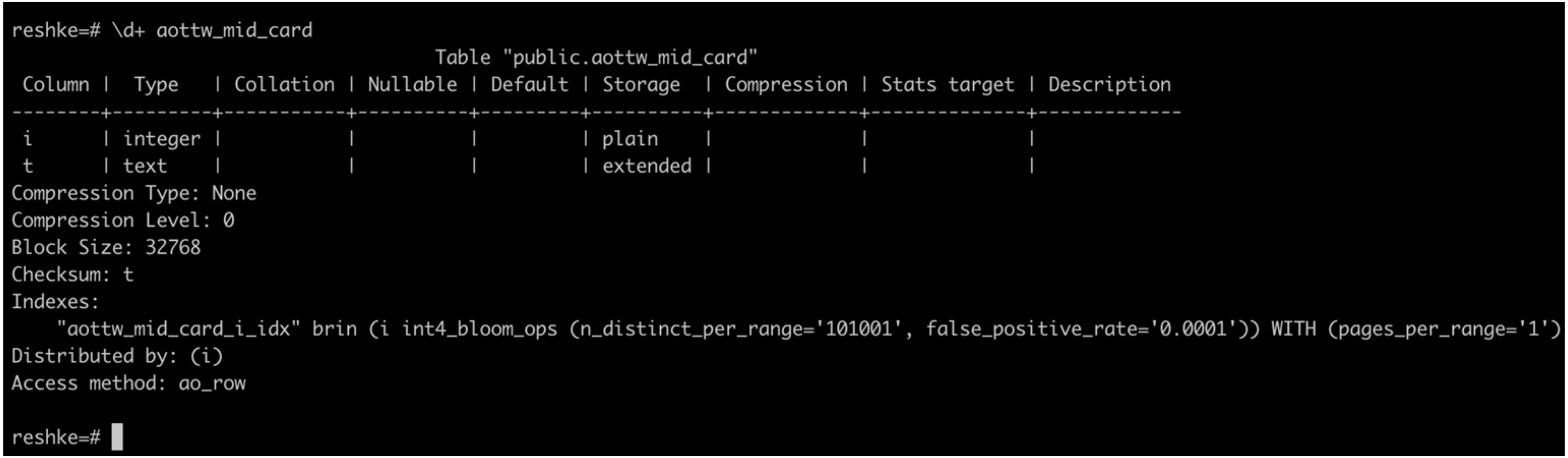

Low glolbal cardinality - select 0.01 of data
seq scan
brin scan
btree scan
execution time
100000
80000
60000
40000
20000
0
4778 MB
9556 MB
14GB
19GB
data size

Посмотрим в brin_page_items



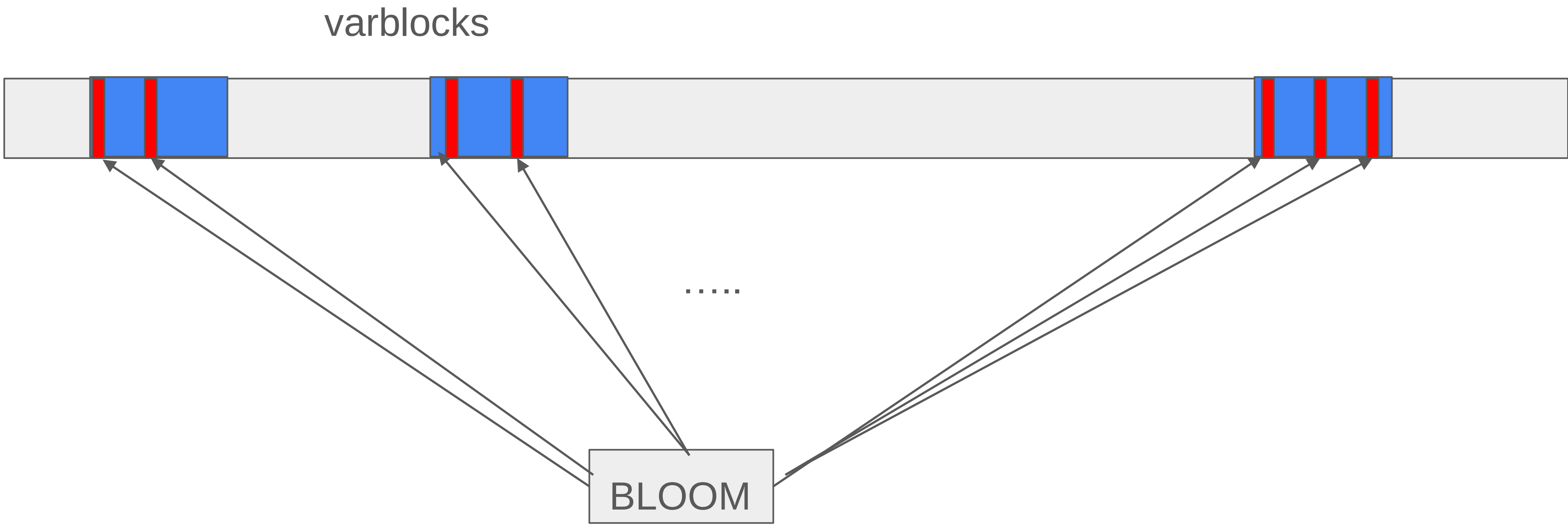
```
 itemoffset |  blknum  | attnum | allnulls | hasnulls | placeholder |                           value
------------+----------+--------+----------+----------+-------------+-----------------------------------------------------------------
          1 | 33554812 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 867}}
          2 | 33554813 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 867}}
          3 | 33554814 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 703}}
          4 | 33554815 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 786}}
          5 | 33554816 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 907}}
          6 | 33554817 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 783}}
          7 | 33554818 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 875}}
          8 | 33554819 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 936}}
          9 | 33554820 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 865}}
         10 | 33554821 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 908}}
         11 | 33554822 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 940}}
         12 | 33554823 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 891}}
         13 | 33554824 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 929}}
         14 | 33554825 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 889}}
         15 | 33554826 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 803}}
         16 | 33554827 |      1 | f        | f        | f           | {{mode: hashed  nhashes: 7  nbits: 1112  nbits_set: 704}}
```

n_distinct_per_range / false_positive_rate



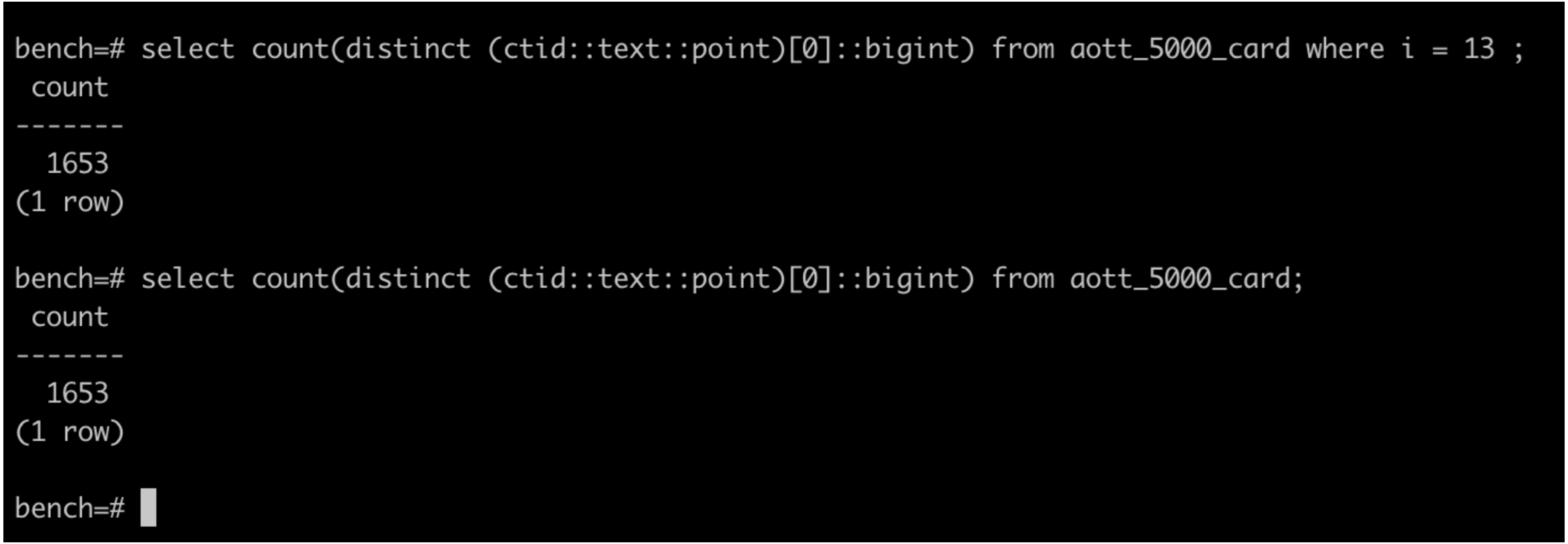
```
reshke=# \d+ aottw_mid_card
                                      Table "public.aottw_mid_card"
 Column |  Type   | Collation | Nullable | Default | Storage  | Compression | Stats target | Description
--------+---------+-----------+----------+---------+----------+-------------+--------------+-------------
 i      | integer |           |          |         | plain    |             |              |
 t      | text    |           |          |         | extended |             |              |
Compression Type: None
Compression Level: 0
Block Size: 32768
Checksum: t
Indexes:
    "aottw_mid_card_i_idx" brin (i int4_bloom_ops (n_distinct_per_range='101001', false_positive_rate='0.0001')) WITH (pages_per_range='1')
Distributed by: (i)
Access method: ao_row

reshke=# 
```

AOTID не зависит от varblock



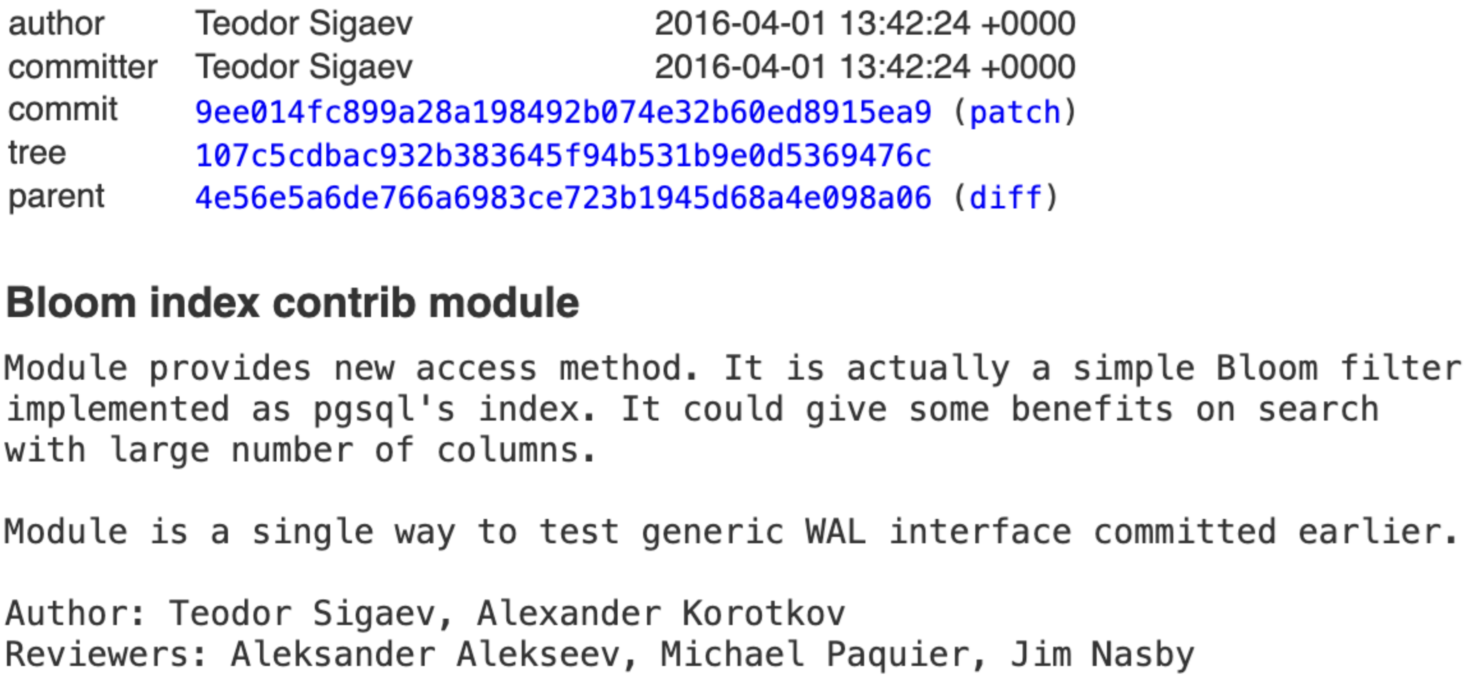
Всегда 32768 таплов

Иногда фильтр не нужен вообще



```
bench=# select count(distinct (ctid::text::point)[0]::bigint) from aott_5000_card where i = 13 ;
 count
-------
  1653
(1 row)

bench=# select count(distinct (ctid::text::point)[0]::bigint) from aott_5000_card;
 count
-------
  1653
(1 row)

bench=# 
```

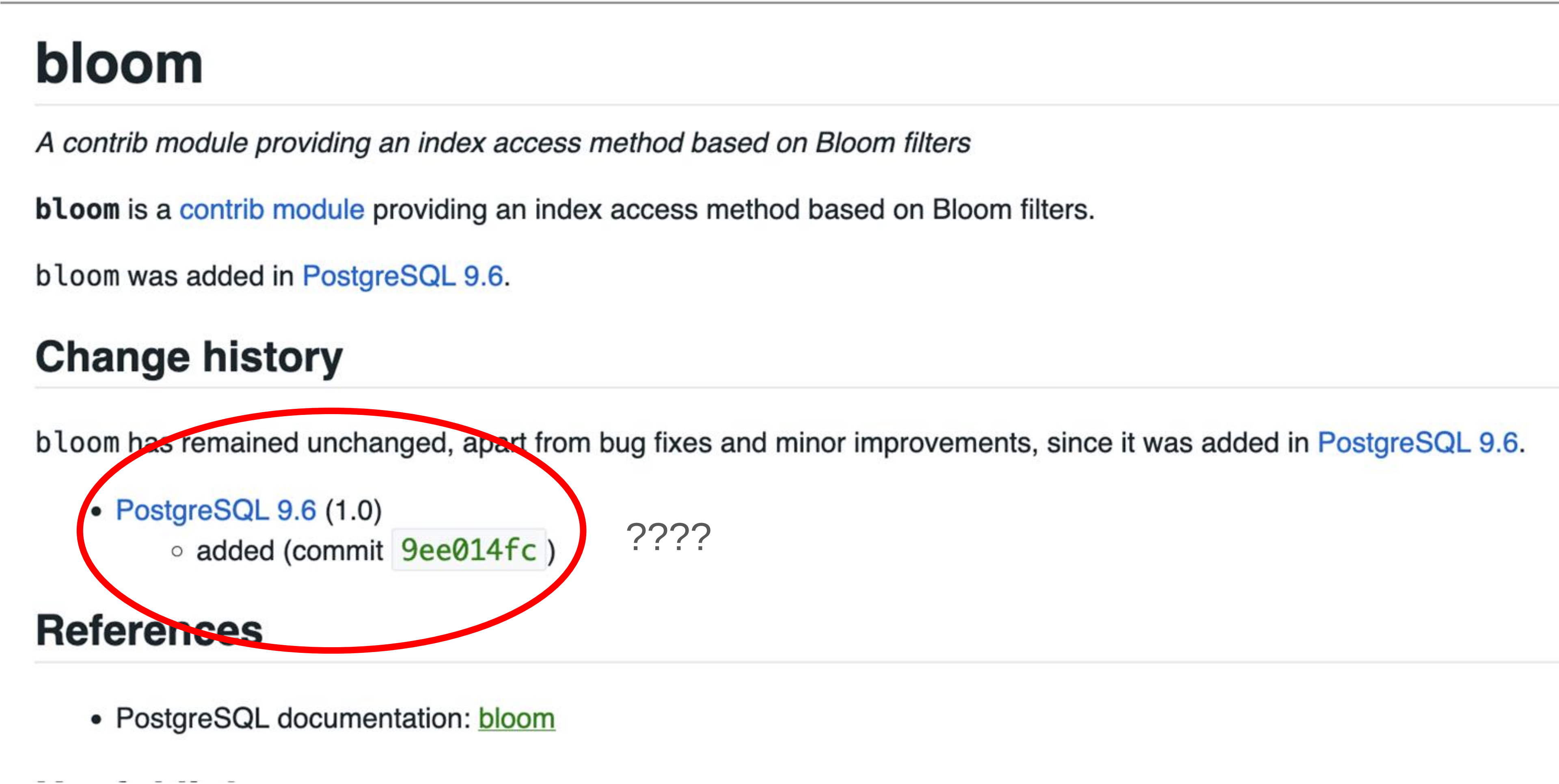
Сузим задачу – получим более простое решение



author	Teodor Sigaev	2016-04-01 13:42:24 +0000
committer	Teodor Sigaev	2016-04-01 13:42:24 +0000
commit	9ee014fc899a28a198492b074e32b60ed8915ea9 (patch)	
tree	107c5cdbac932b383645f94b531b9e0d5369476c	
parent	4e56e5a6de766a6983ce723b1945d68a4e098a06 (diff)	

Bloom index contrib module

```
Module provides new access method. It is actually a simple Bloom filter
implemented as pgsql's index. It could give some benefits on search
with large number of columns.

Module is a single way to test generic WAL interface committed earlier.

Author: Teodor Sigaev, Alexander Korotkov
Reviewers: Aleksander Alekseev, Michael Paquier, Jim Nasby
```



bloom

A contrib module providing an index access method based on Bloom filters

`bloom` is a contrib module providing an index access method based on Bloom filters.

`bloom` was added in PostgreSQL 9.6.

Change history

`bloom` has remained unchanged, apart from bug fixes and minor improvements, since it was added in PostgreSQL 9.6.

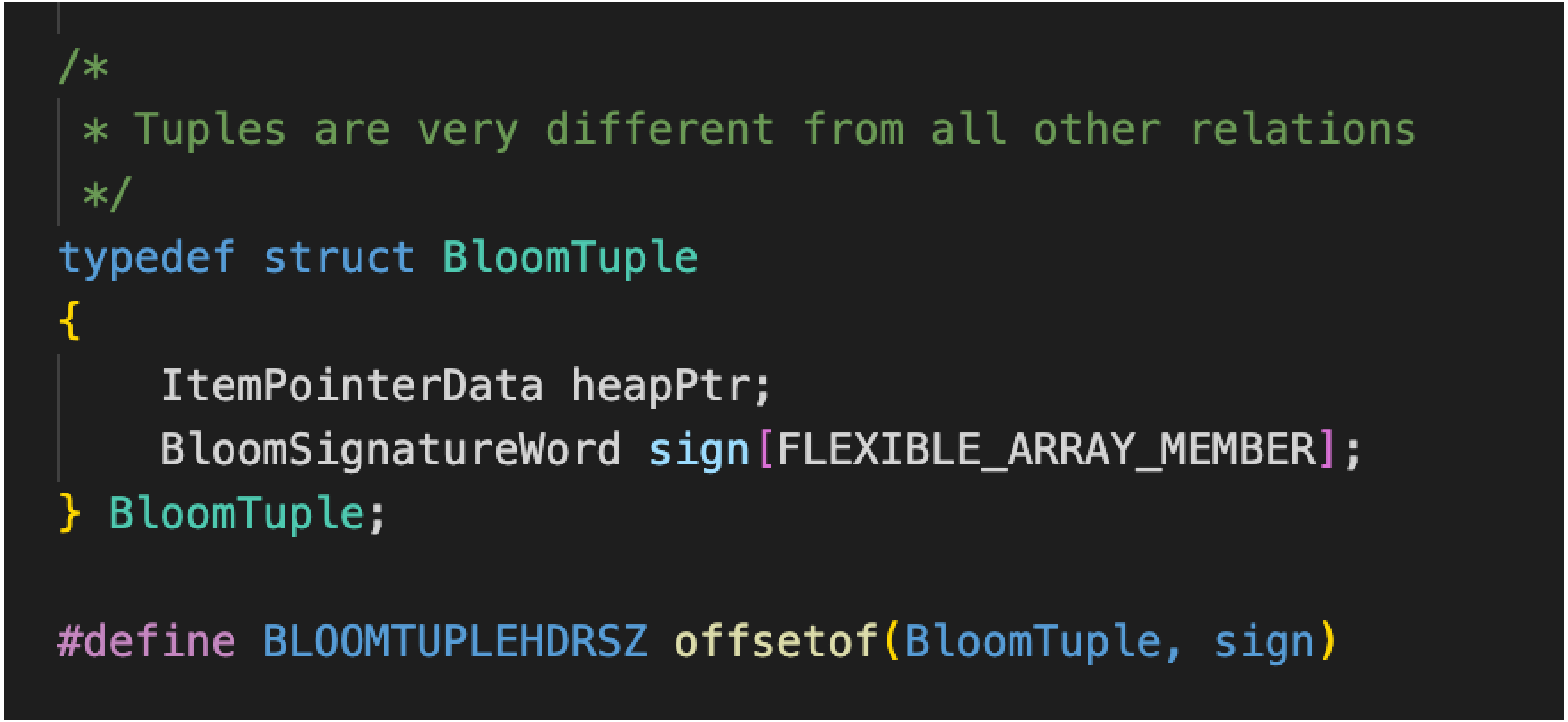
- PostgreSQL 9.6 (1.0)
  - added (commit `9ee014fc`)

????

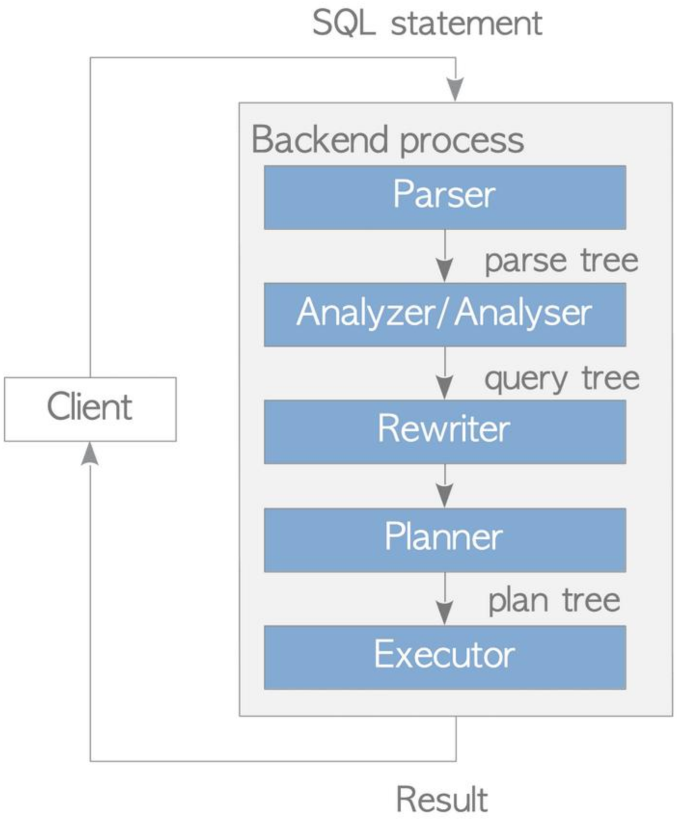
References

- PostgreSQL documentation: bloom

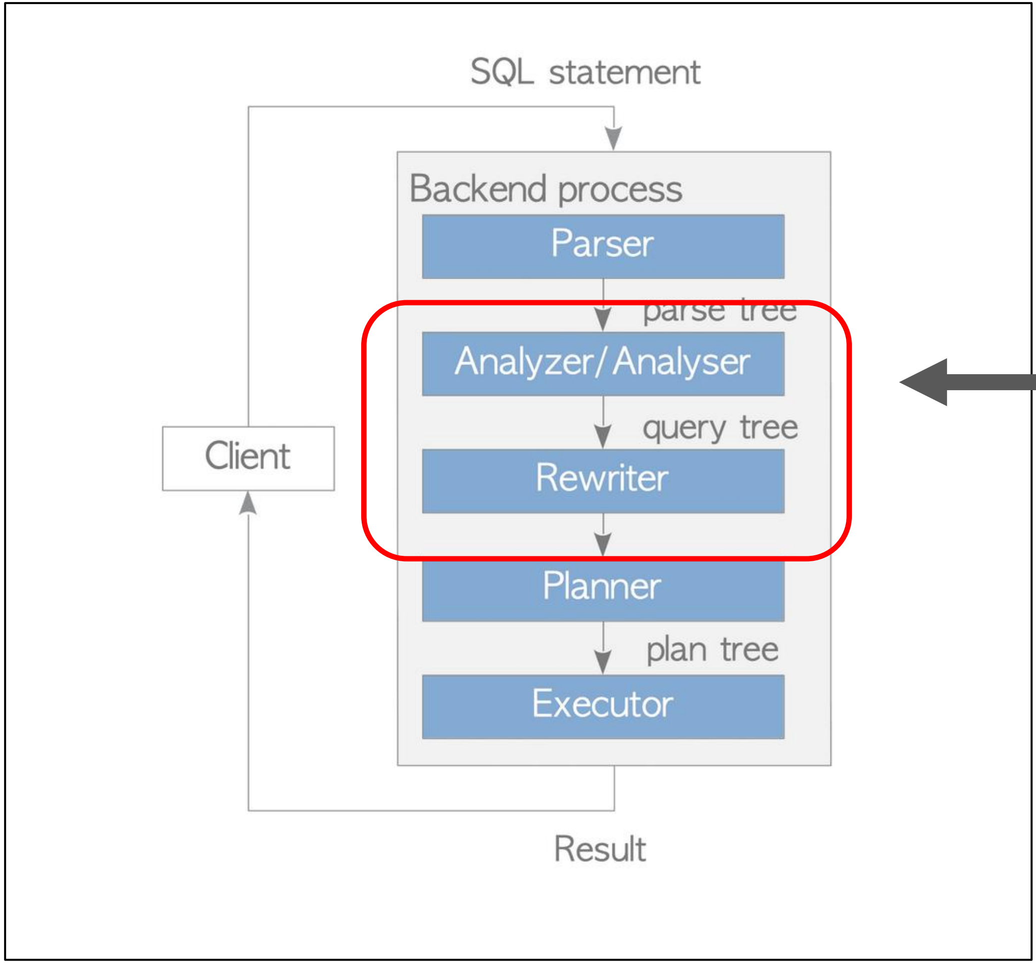
1 index tuple = 1 relation tuple



```
/*
 * Tuples are very different from all other relations
 */
typedef struct BloomTuple
{
    ItemPointerData heapPtr;
    BloomSignatureWord sign[FLEXIBLE_ARRAY_MEMBER];
} BloomTuple;

#define BLOOMTUPLEHDRSZ offsetof(BloomTuple, sign)
```

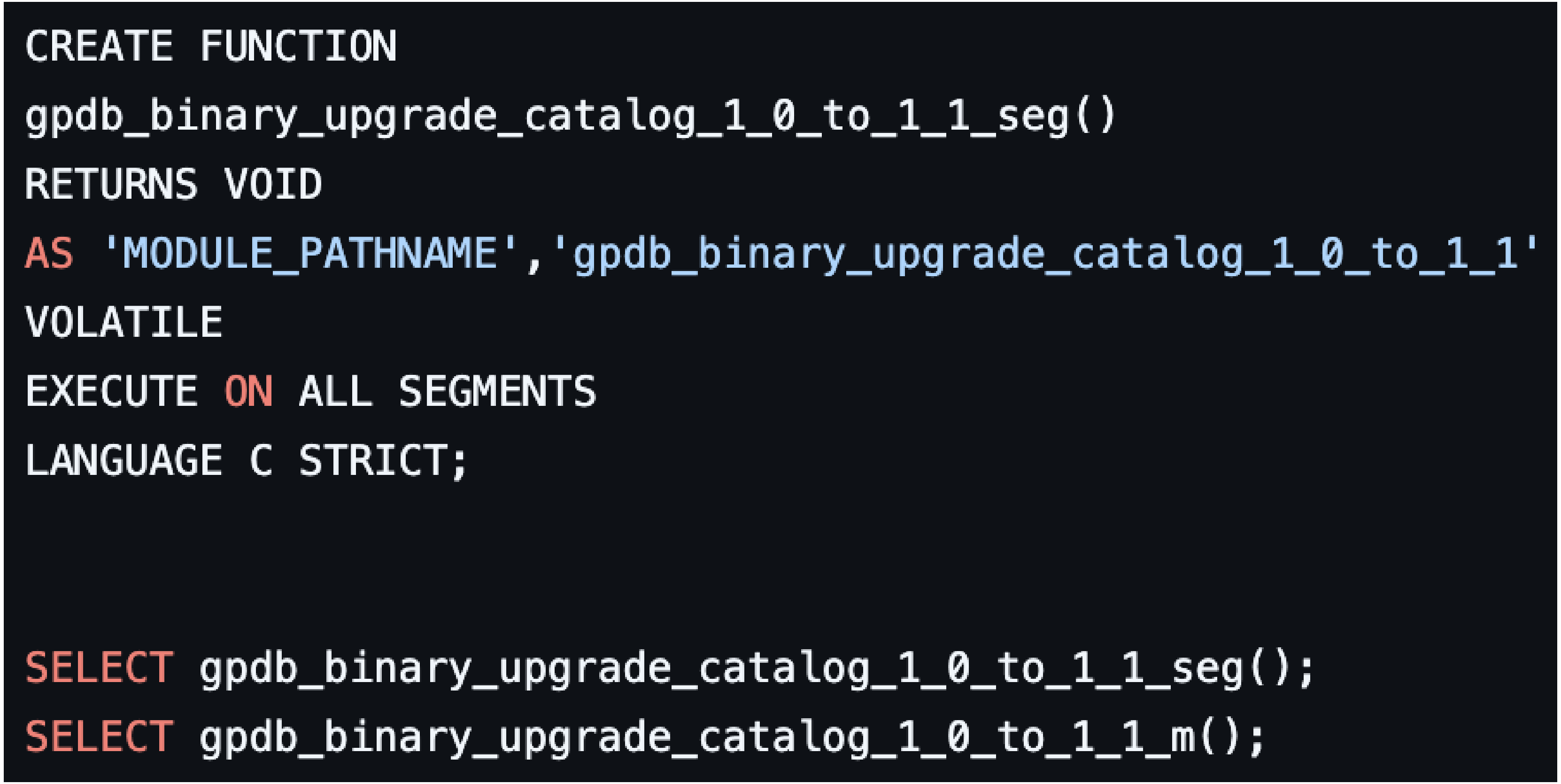


База данных должна выполнять запросы



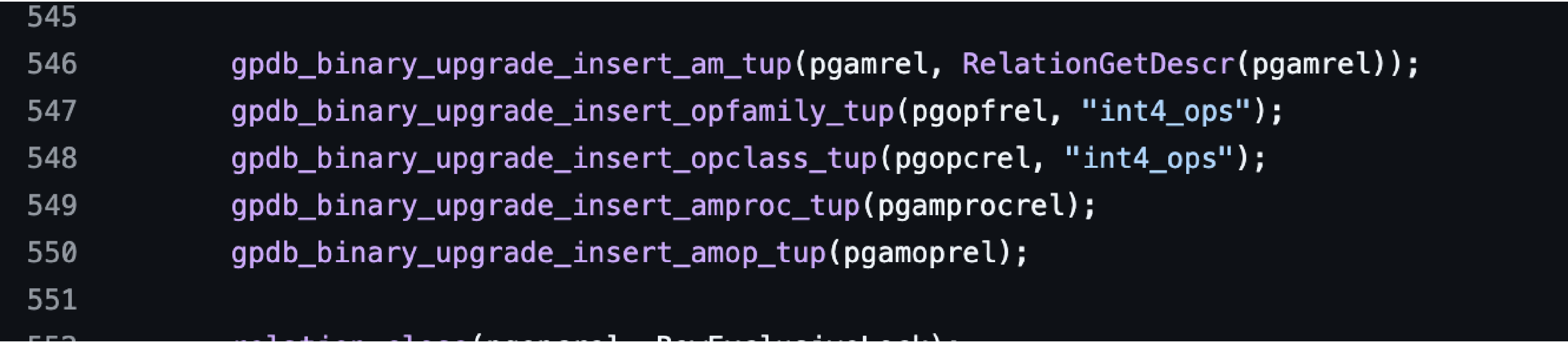
Нужен каталог

gp_aux_catalog!



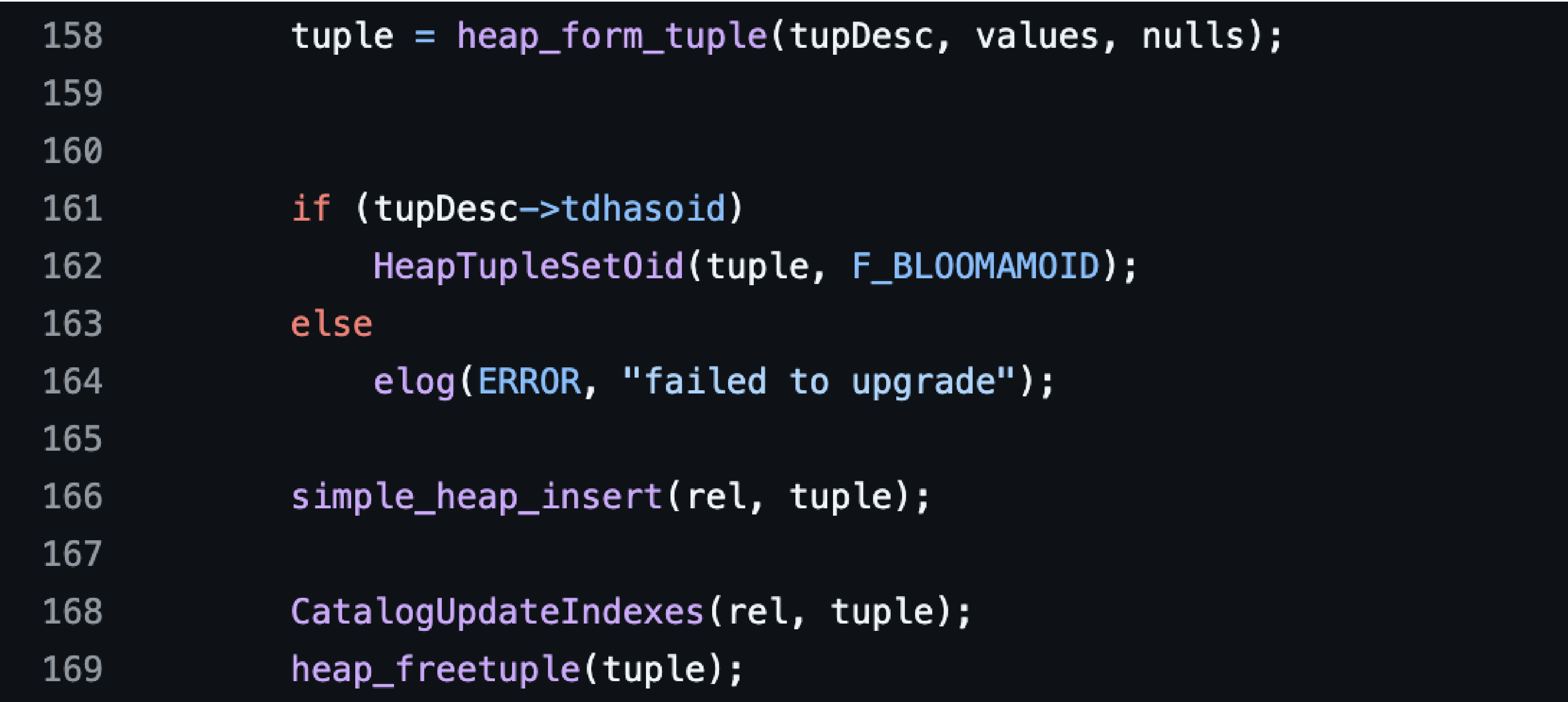
```
CREATE FUNCTION
gpdb_binary_upgrade_catalog_1_0_to_1_1_seg()
RETURNS VOID
AS 'MODULE_PATHNAME','gpdb_binary_upgrade_catalog_1_0_to_1_1'
VOLATILE
EXECUTE ON ALL SEGMENTS
LANGUAGE C STRICT;



SELECT gpdb_binary_upgrade_catalog_1_0_to_1_1_seg();
SELECT gpdb_binary_upgrade_catalog_1_0_to_1_1_m();
```

Что оно делает?



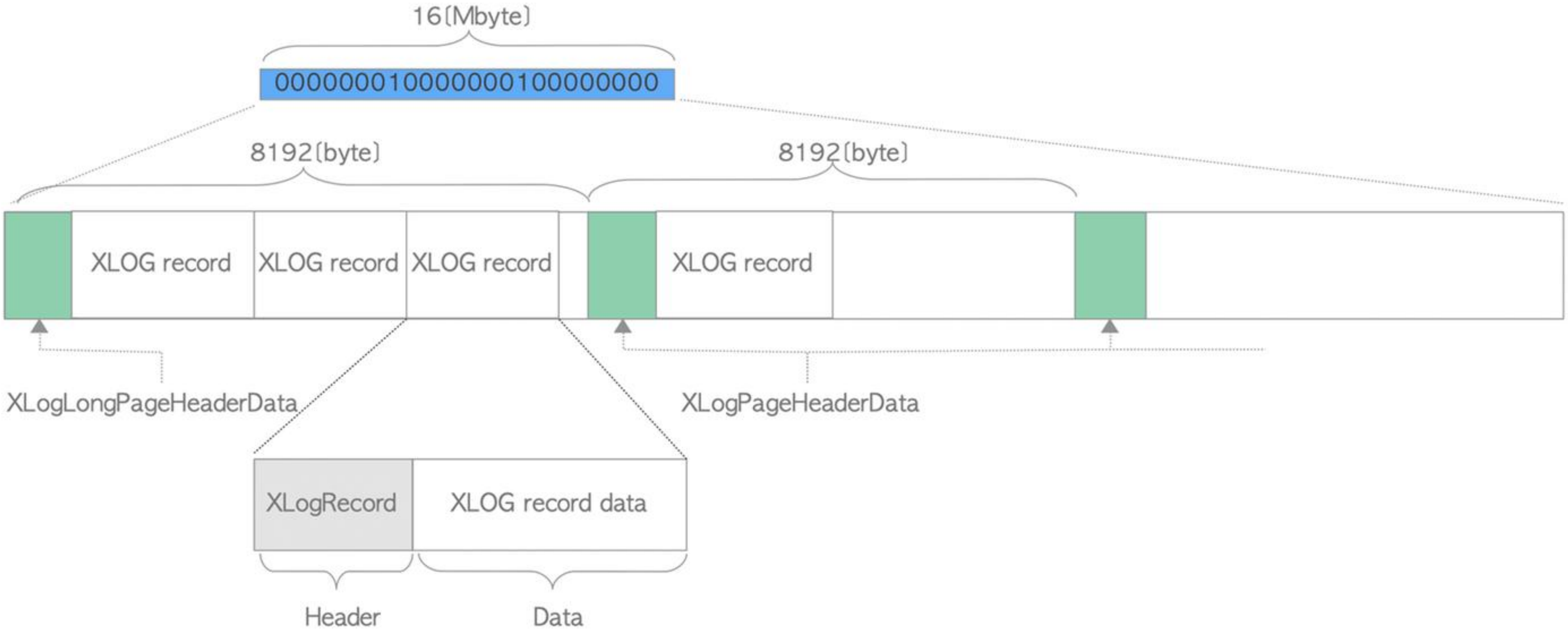
```
        gpdb_binary_upgrade_insert_am_tup(pgamrel, RelationGetDescr(pgamrel));
        gpdb_binary_upgrade_insert_opfamily_tup(pgopfrel, "int4_ops");
        gpdb_binary_upgrade_insert_opclass_tup(pgopcrel, "int4_ops");
        gpdb_binary_upgrade_insert_amproc_tup(pgamprocrel);
        gpdb_binary_upgrade_insert_amop_tup(pgamoprel);
```

Что оно делает?

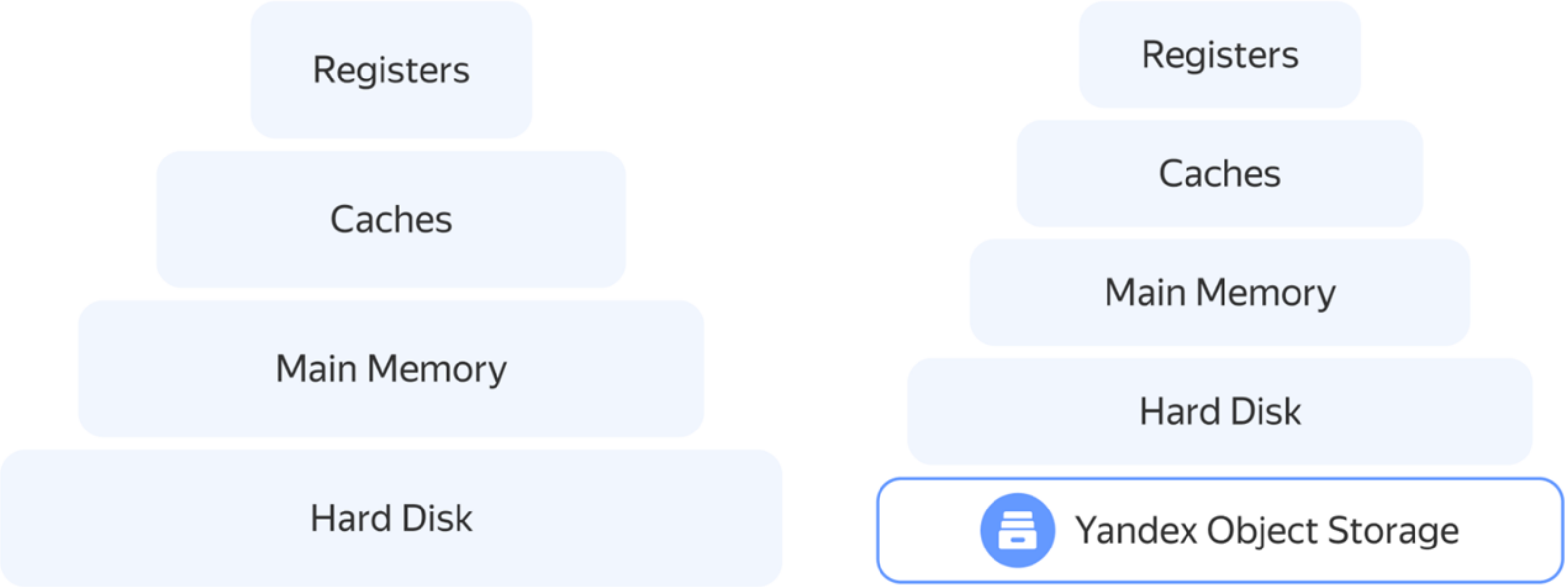


```
tuple = heap_form_tuple(tupDesc, values, nulls);


if (tupDesc->tdhasoid)
    HeapTupleSetOid(tuple, F_BLOOMAMOID);
else
    elog(ERROR, "failed to upgrade");

simple_heap_insert(rel, tuple);

CatalogUpdateIndexes(rel, tuple);
heap_freetuple(tuple);
```

Убрать WAL в BRIN/hint fork. Generic xlog



Yezzey ABI in gpdb6



Registers

Caches

Main Memory

Hard Disk

Registers

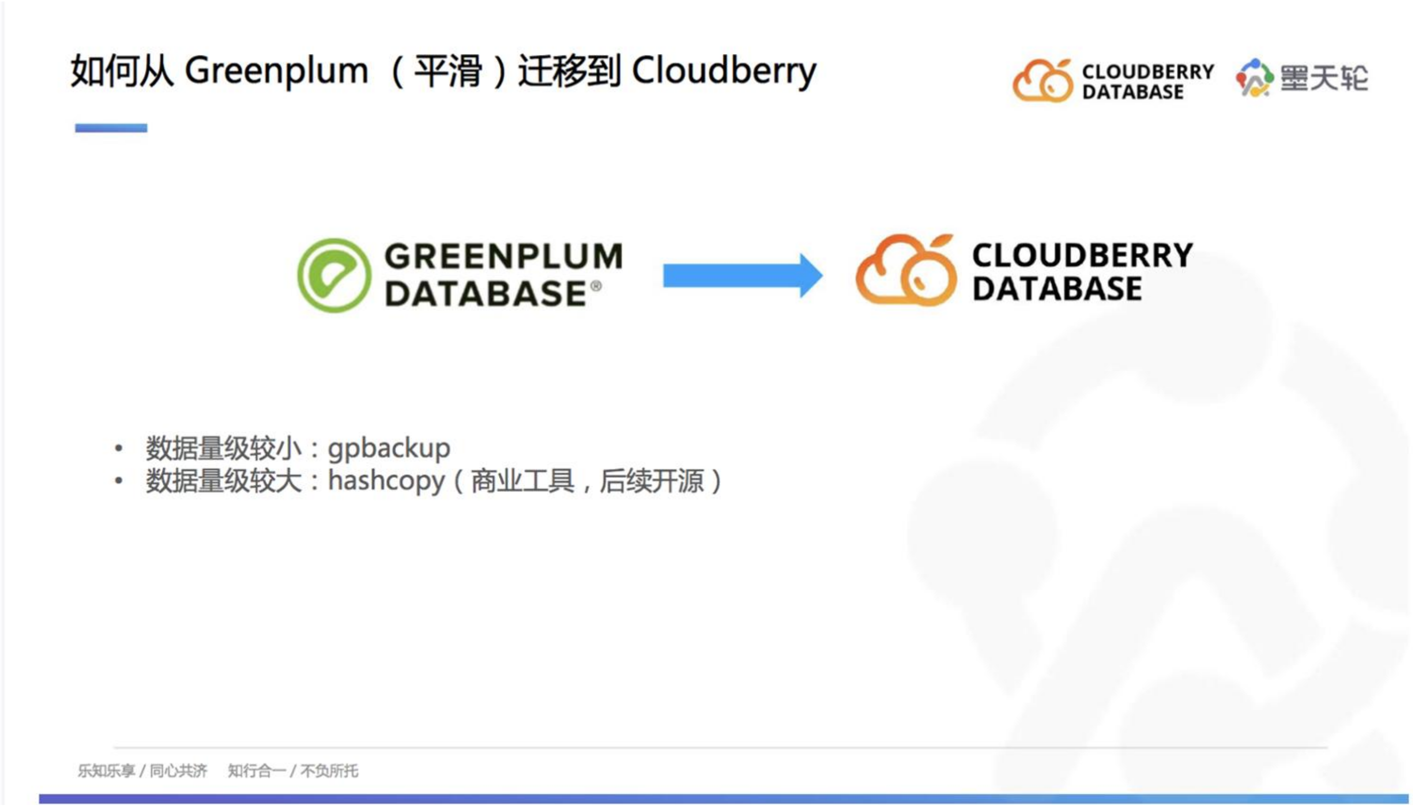
Caches

Main Memory

Hard Disk

Yandex Object Storage

Как из гп сделать клаудберри?

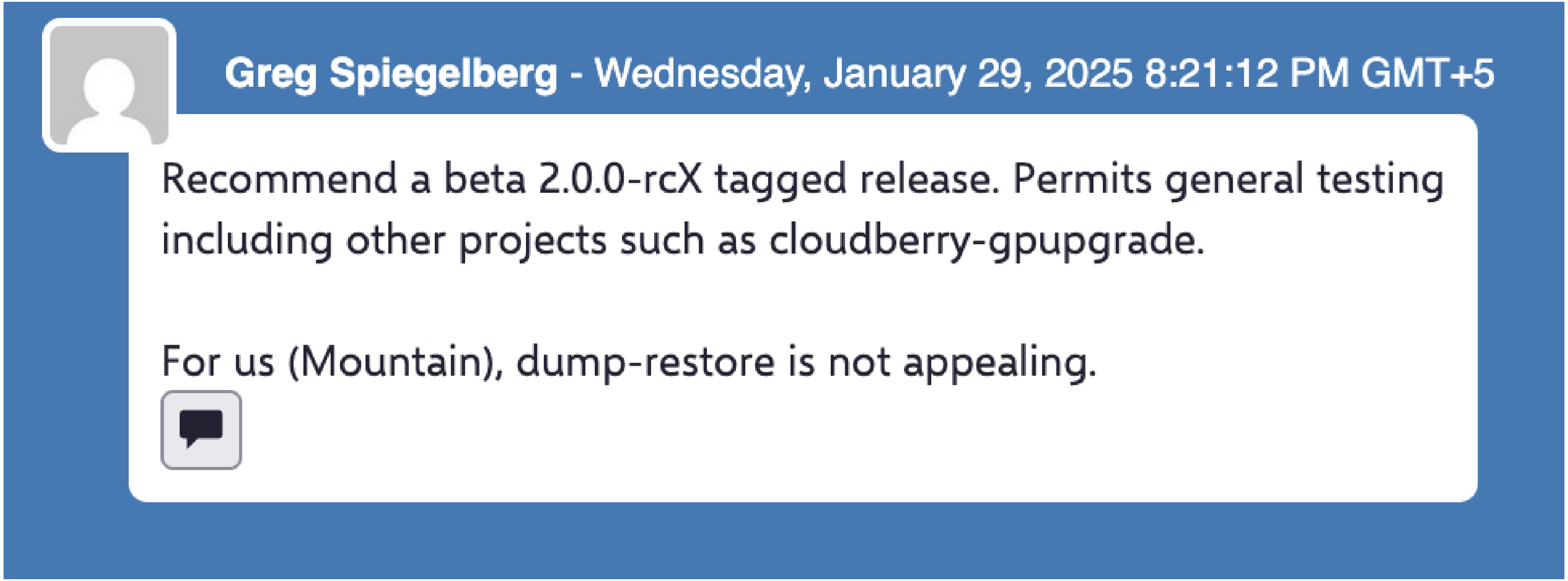


如何从 Greenplum （平滑）迁移到 Cloudberry

GREENPLUM DATABASE® → CLOUDBERRY DATABASE

- 数据量级较小：gpbackup
- 数据量级较大：hashcopy（商业工具，后续开源）

Как из гп сделать клаудберри?



Greg Spiegelberg - Wednesday, January 29, 2025 8:21:12 PM GMT+5

Recommend a beta 2.0.0-rcX tagged release. Permits general testing including other projects such as cloudberry-gpupgrade.

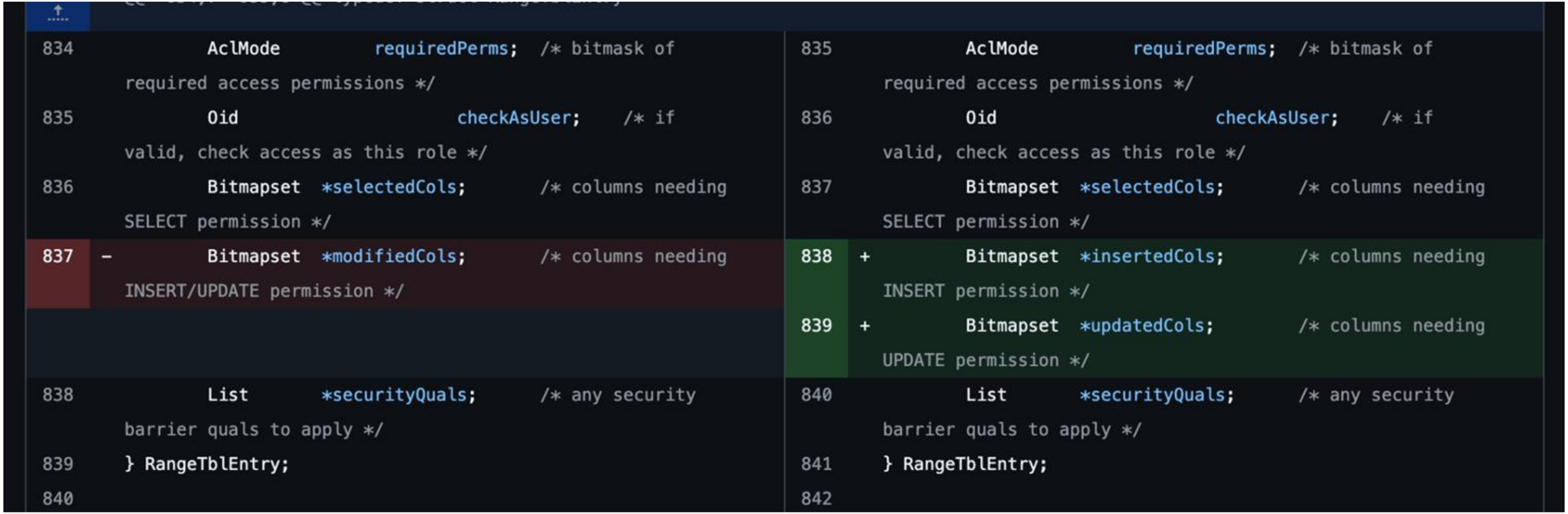
For us (Mountain), dump-restore is not appealing.

Quick intro into PostgreSQL ABI

Что вообще меняется при смене мажорной версии PostgreSQL?

- Появляются новые catalogue таблички и удаляются старые
- Появляются новые колонки и удаляются старые
- Меняются OID почти произвольных образом
- Возможно перестанет работать статистика

ABI конфликты на сладкое

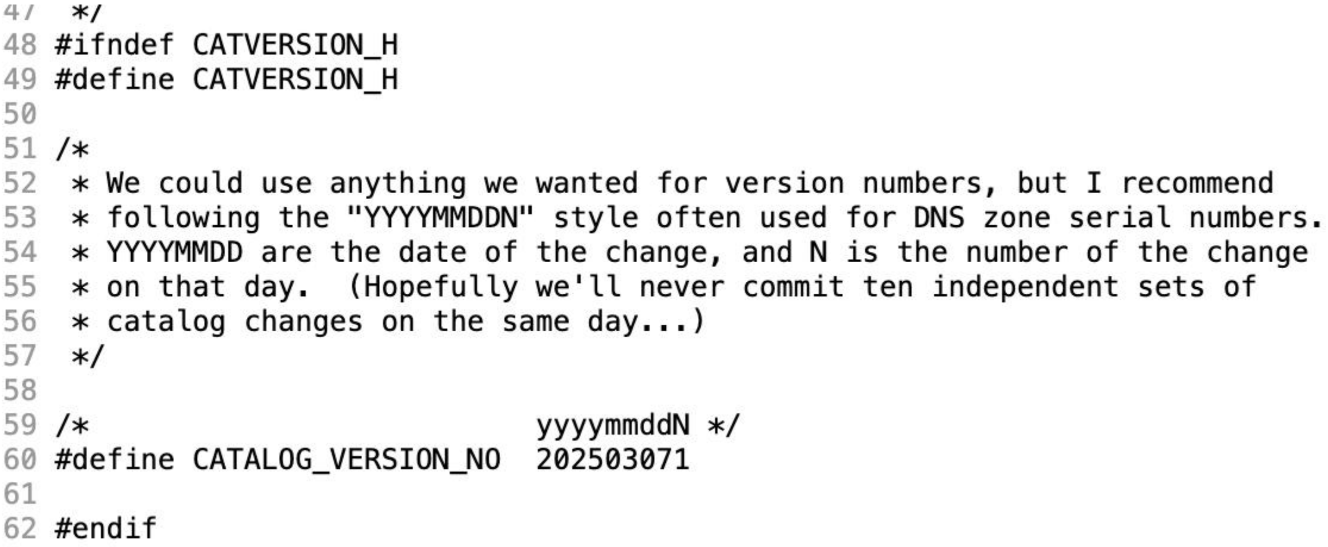


```
        AclMode         requiredPerms;  /* bitmask of required access permissions */
        Oid                     checkAsUser;    /* if valid, check access as this role */
        Bitmapset  *selectedCols;       /* columns needing SELECT permission */
-       Bitmapset  *modifiedCols;       /* columns needing INSERT/UPDATE permission */
        List       *securityQuals;      /* any security barrier quals to apply */
} RangeTblEntry;
```

```
        AclMode         requiredPerms;  /* bitmask of required access permissions */
        Oid                     checkAsUser;    /* if valid, check access as this role */
        Bitmapset  *selectedCols;       /* columns needing SELECT permission */
+       Bitmapset  *insertedCols;       /* columns needing INSERT permission */
+       Bitmapset  *updatedCols;        /* columns needing UPDATE permission */
        List       *securityQuals;      /* any security barrier quals to apply */
} RangeTblEntry;
```

https://git.postgresql.org/gitweb/?p=postgresql.git;a=commitdiff;h=2c8f4836db058d0715bc30a30655d646287ba509

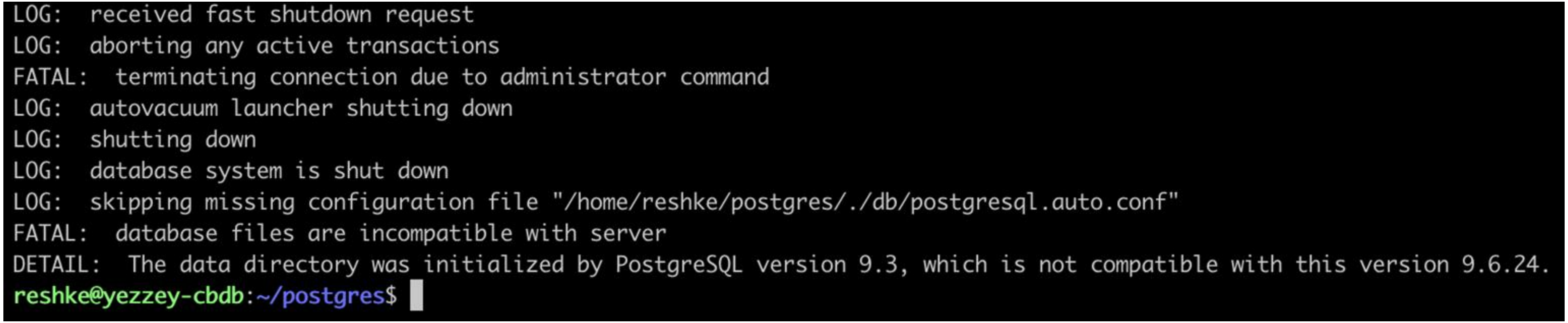
Санитары позаботились



```
 */
#ifndef CATVERSION_H
#define CATVERSION_H

/*
 * We could use anything we wanted for version numbers, but I recommend
 * following the "YYYYMMDDN" style often used for DNS zone serial numbers.
 * YYYYMMDD are the date of the change, and N is the number of the change
 * on that day.  (Hopefully we'll never commit ten independent sets of
 * catalog changes on the same day...)
 */

/*                          yyyymmddN */
#define CATALOG_VERSION_NO  202503071

#endif
```

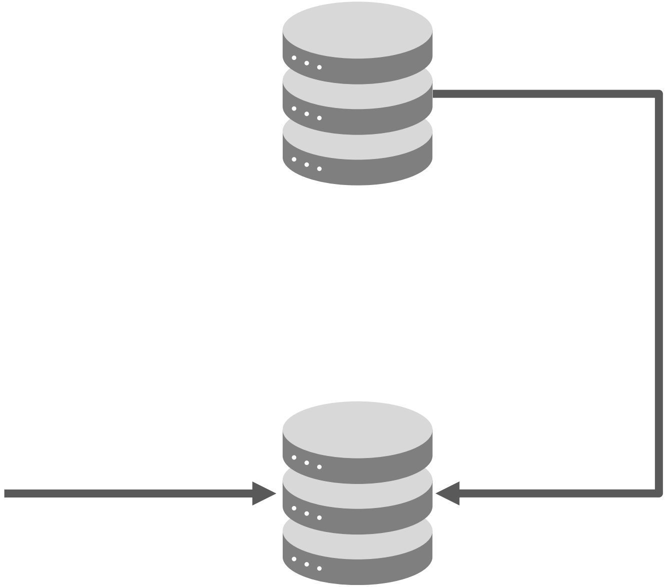
https://git.postgresql.org/gitweb/?p=postgresql.git;a=blob;f=src/include/catalog/catversion.h;h=f427a89618b9a17f6afcc5879d6c6339e085e240;hb=d3fc7a51208b3f4f2be2476d44aa2542f52879de#l59

Санитары позаботились



```
LOG:  received fast shutdown request
LOG:  aborting any active transactions
FATAL:  terminating connection due to administrator command
LOG:  autovacuum launcher shutting down
LOG:  shutting down
LOG:  database system is shut down
LOG:  skipping missing configuration file "/home/reshke/postgres/./db/postgresql.auto.conf"
FATAL:  database files are incompatible with server
DETAIL:  The data directory was initialized by PostgreSQL version 9.3, which is not compatible with this version 9.6.24.
reshke@yezzey-cbdb:~/postgres$ 
```

Gpupgrade!

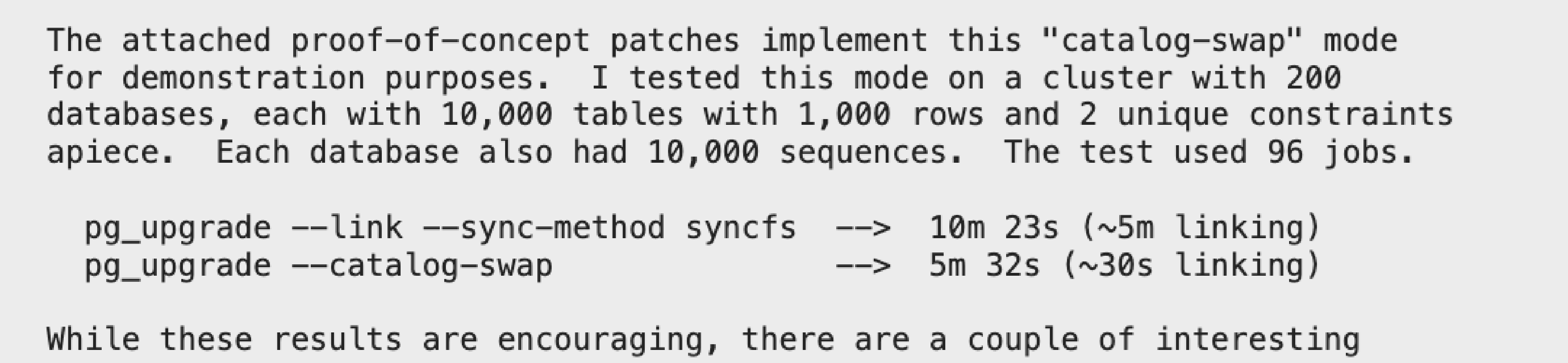


initdb

pg_dump/pg_restore +
relfilenode transfer

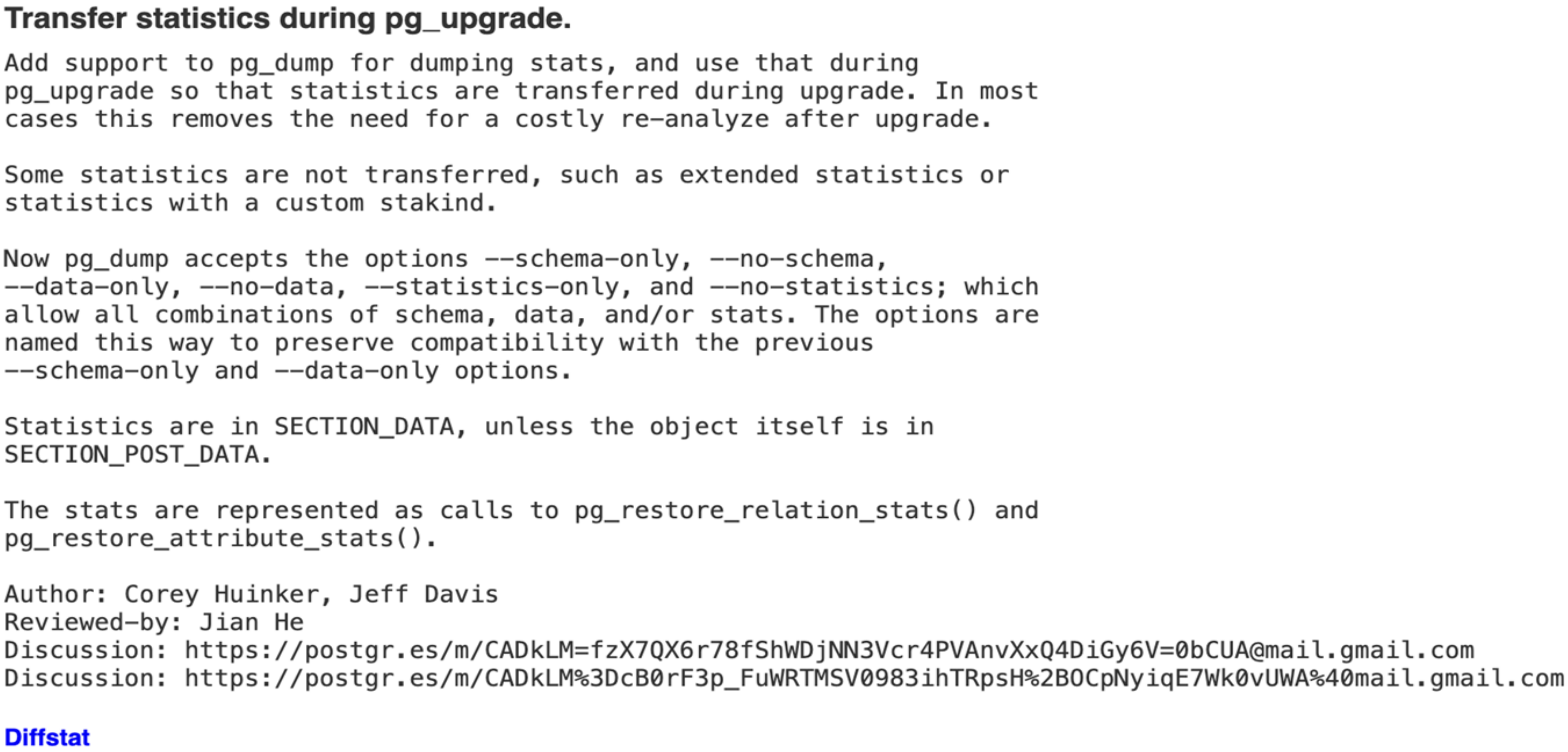
Relfilenode (relfilelocalor) transfer

https://www.postgresql.org/message-id/Zyvop-LxLXBLrZiI@nathan



```
The attached proof-of-concept patches implement this "catalog-swap" mode
for demonstration purposes.  I tested this mode on a cluster with 200
databases, each with 10,000 tables with 1,000 rows and 2 unique constraints
apiece.  Each database also had 10,000 sequences.  The test used 96 jobs.

  pg_upgrade --link --sync-method syncfs  -->  10m 23s (~5m linking)
  pg_upgrade --catalog-swap               -->  5m 32s (~30s linking)

While these results are encouraging, there are a couple of interesting
```

Import/Export statistics



Transfer statistics during pg_upgrade.

```
Add support to pg_dump for dumping stats, and use that during
pg_upgrade so that statistics are transferred during upgrade. In most
cases this removes the need for a costly re-analyze after upgrade.

Some statistics are not transferred, such as extended statistics or
statistics with a custom stakind.

Now pg_dump accepts the options --schema-only, --no-schema,
--data-only, --no-data, --statistics-only, and --no-statistics; which
allow all combinations of schema, data, and/or stats. The options are
named this way to preserve compatibility with the previous
--schema-only and --data-only options.

Statistics are in SECTION_DATA, unless the object itself is in
SECTION_POST_DATA.

The stats are represented as calls to pg_restore_relation_stats() and
pg_restore_attribute_stats().

Author: Corey Huinker, Jeff Davis
Reviewed-by: Jian He
Discussion: https://postgr.es/m/CADkLM=fzX7QX6r78fShWDjNN3Vcr4PVAnvXxQ4DiGy6V=0bCUA@mail.gmail.com
Discussion: https://postgr.es/m/CADkLM%3DcB0rF3p_FuWRTMSV0983ihTRpsH%2BOCpNyiqE7Wk0vUWA%40mail.gmail.com
```

Diffstat

Ставьте лайки