Where is the space, Postgres?

PGDAY RUSSIA '16

Alexey Ermakov
alexey.ermakov@postgresql-consulting.com

PostgreSQL-Consulting.com

PGDAY RUSSIA '16

О чем сегодня будем говорить

- Куда уходит свободное место?
- Инструменты и запросы
- Каким образом данные в действительности хранятся?
- Действия в аварийной и предаварийной ситуациях

PGDAY RUSSIA '16

Database File Layout

Проблемные места

```
base
global
pg_clog
pg_dynshmem
pg_log
pg_logical
pg_multixact
pg_notify
pg_replslot
pg_serial
pg_snapshots
pg_stat
pg_stat_tmp
pg_subtrans
pg_tblspc
          |-- ~16407 ->
pg_twophase
pg_xlog
```

PGDAY RUSSIA '16

Database File Layout

Хранение таблиц и индексов

```
base
    |-- 1
    |-- 13051
    |-- 13056
          |-- 79593      481080K
          |-- 90694      1024M
          |-- 90694.1    1024M
          |-- 90694.2    903136K
          |-- 90694_fsm  1564672
          |-- 90694_vm   0
          |-- ...
    |-- ...
    |-- pgsql_tmp
```

PGDAY RUSSIA '16

pgsql_tmp

Временные файлы

- пишутся при нехватке work_mem, maintenance_work_mem
- **temp_file_limit** (9.2+)
- log_temp_files

PGDAY RUSSIA '16

pg_log

Логи

- **log_min_duration_statement**
- log_directory
- log_filename
- no limit for message length

PGDAY RUSSIA '16

pg_xlog expected size

Write Ahead Log (журнал транзакций)

- 1 WAL segment 16MB
- used for recovery and replication
- removed/recycled after checkpoint

$$MAX \begin{cases} checkpoint\_segments + wal\_keep\_segments \\ (2 + checkpoint\_completion\_target) * checkpoint\_segments \end{cases}$$

PGDAY RUSSIA '16

pg_xlog more than expected size

- broken archive_command
- replication slots
- slow replay
- CPU or I/O overload (renice/ionice startup process)
- long running transactions on standby

PGDAY RUSSIA '16

TOP 20 largest databases

pg-utils/sql/top_databases.sql

```
   name    |  owner   |  size
-----------+----------+---------
 the       | postgres | 74 GB
 quick     | postgres | 65 GB
 brown     | postgres | 41 GB
 fox       | postgres | 36 GB
 jumps     | postgres | 30 GB
 over      | postgres | 28 GB
 lazy      | postgres | 25 GB
 dog       | postgres | 13 GB
```

PGDAY RUSSIA '16

TOP 20 largest tables/indexes in selected DB

pg-utils/sql/top_tables.sql

```
 nspname |          relname           | type |  size   | idxsize |  total
---------+----------------------------+------+---------+---------+---------
 public  | posts                      | r    | 20 GB   | 24 GB   | 43 GB
 public  | users                      | r    | 4152 MB | 3215 MB | 7367 MB
 public  | posts_type_id_created_at   | i    | 4861 MB | 0 bytes | 4861 MB
 public  | posts_user_id              | i    | 2612 MB | 0 bytes | 2612 MB
```

PGDAY RUSSIA '16

Unused indexes

github.com/pgexperts/pgx_scripts/indexes/unused_indexes.sql

reason	table/index name	index scan %	scans per write	index size	table size
Never Used Indexes	...	0.00	0.00	18 GB	78 GB
Never Used Indexes	...	0.00	0.00	6371 MB	64 GB
Never Used Indexes	...	0.00	0.00	4875 MB	64 GB
Never Used Indexes	...	0.00	0.00	4412 MB	64 GB
Low Scans, High Writes	...	0.00	0.00	6358 MB	64 GB
Low Scans, High Writes	...	0.00	0.00	6290 MB	64 GB
Seldom Used Large Indexes	...	0.24	29811.00	1171 MB	8726 MB
Seldom Used Large Indexes	...	0.00	3.00	1121 MB	8726 MB

PGDAY RUSSIA '16

Duplicate indexes

github.com/pgexperts/pgx_scripts/indexes/duplicate_indexes_fuzzy.sql

```
 table   index       index
 name    | name      | cols   | indexdef                                | index_scans
---------+-----------+--------+-----------------------------------------+-------------
 users   | users_a   | a      | CREATE INDEX ... USING btree (a)        |       10140
 users   | users_a_b | a,b    | CREATE INDEX ... USING btree (a, b)     |     2194900
```

Compact indexes

Иногда можно создать более компактный индекс

- Partial indexes
- gin indexes (btree_gin) for duplicates
- BRIN indexes

PGDAY RUSSIA '16

Bloat

- multiversion concurrency control (MVCC)
- При каждом update строки создается ее копия
- Ненужные копии подчищаются процессом autovacuum
- autovacuum не может уменьшить размер таблицы, если только нет пустых страниц в конце файла

PGDAY RUSSIA '16

Bloat

```
create table test(id integer primary key, flag boolean not null);
insert into test values(1, false);

select to_hex(xmin::text::bigint) as xmin, * from test;
 xmin | id | flag
------+----+------
 3ec  |  1 | f
```

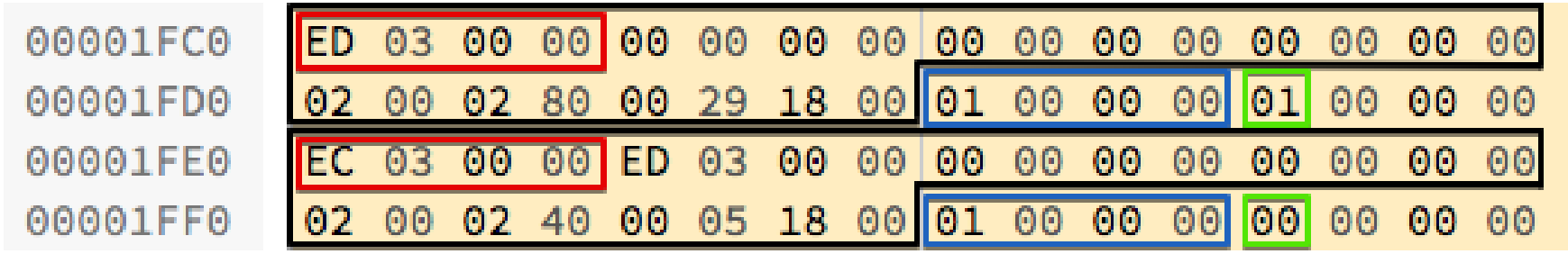
```
00001FD0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00001FE0  EC 03 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00001FF0  01 00 02 00 00 08 18 00  01 00 00 00 00 00 00 00
00002000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
```

PGDAY RUSSIA '16

Bloat

```
update test set flag = true where id = 1;

select to_hex(xmin::text::bigint) as xmin, * from test;
 xmin | id | flag
------+----+------
 3ed  |  1 | t
```



PGDAY RUSSIA '16

Where did all you zombies come from?

- Long running transactions
- Not tuned autovacuum
- Update large number of rows
- Delete large number of rows
- Drop columns

Long running transactions

Как бороться?

- Мониторинг длины самой долгой транзакции (на репликах с hot_standby_feedback = on тоже!)
- Автоматически прибивать по крону (см. pg_terminate_backend(), pg_stat_activity)
- Модифицировать приложение
- pg_dump ⇒ pg_basebackup

autovacuum

- autovacuum_vacuum_scale_factor по-умолчанию 0.2 (20% таблицы)
- autovacuum_max_workers
- autovacuum_vacuum_cost_delay

Bloat estimation

Оценка bloat на основе данных статистики с разной степенью достоверности

- heroku pg:bloat github.com/heroku/heroku-pg-extras
- check_postgres.pl bucardo.org/check_postgres/check_postgres.pl.html
- github.com/ioguix/pgsql-bloat-estimation
- pgstattuple extension

PGDAY RUSSIA '16

Table bloat

pg-utils/sql/table_bloat.sql

nspname	relname	total size	toast size	table waste percent	table waste	total waste percent	total waste
public	obscure	22 GB	0 bytes	72.9	16 GB	72.9	16 GB
public	problems	4814 MB	0 bytes	11.6	558 MB	11.6	558 MB
public	may	3428 MB	8192 by	2.0	68 MB	2.0	68 MB
public	require	103 MB	0 bytes	49.1	51 MB	49.1	51 MB
public	unusual	279 MB	11 MB	8.8	24 MB	10.4	29 MB
public	solutions	59 MB	0 bytes	41.3	24 MB	41.3	24 MB

PGDAY RUSSIA '16

Index bloat

pg-utils/sql/index_bloat.sql (btree indexes only)

schema	table name	table size	index name	index size	index scans	waste percent	waste
public	alfa	7046 MB	alfa_unifo	6226 MB	273454	80.0	5041 MB
public	alfa	7046 MB	alfa_romeo	4630 MB	324259	69.0	3226 MB
public	bravo	11 GB	bravo_hote	6197 MB	3562344	13.0	808 MB
public	alfa	7046 MB	alfa_lima	2154 MB	6140	35.0	754 MB
public	delta	13 GB	delta_echo	1500 MB	3045468	40.0	613 MB
public	alfa	7046 MB	alfa_charl	1850 MB	2699598	24.0	454 MB
public	delta	5421 MB	delta_foxt	1780 MB	12774	18.0	329 MB

VACUUM FULL/CLUSTER

- exclusive lock
- requires additional space
- could be optimal for small tables

create new empty table and swap with existing

- подходит для insert-only таблиц (логи)
- быстро

PGDAY RUSSIA '16

pg_repack

- позволяет сделать cluster таблицы
- требует дополнительное место под копию таблицы
- модифицирует системные каталоги

PGDAY RUSSIA '16

pgcompact/pgcompacttable

- не требует места под копию таблицы
- можно регулировать нагрузку на диски
- работает на уровне приложения

PGDAY RUSSIA '16

pgcompact/pgcompacttable как работает?

- initial vacuum table
- определение bloat таблиц/индексов
- fake updates: update only xxx set a=a where ctid = ANY(?)
- vacuum table
- reindex concurrently

PGDAY RUSSIA '16

pgcompacttable проблемы

- возможное неиспользование индексов на реплике с pgbouncer [1]
- не делается analyze после пересоздания функциональных индексов
- нет таймаута на alter index xxx rename to yyy
- можно сгенерировать значительное количество WAL
- не сжимает toast'ы

[1] http://bit.ly/28YR0F0 Relation cache invalidation on replica

TOAST

What is your function?

PGDAY RUSSIA '16

TOAST

- хранение длинных (2kb+) значений
- разбивает на chunk'и по 2кб и хранит в специальной таблице в схеме pg_toast
- может сжимать данные алгоритмом LZ
- toast таблицы нельзя модифицировать

TOAST

```
#create table test_toast (id serial primary key, payload text);
#insert into test_toast (payload)
      select string_agg(md5(j::text),'') from generate_series(1,1000) gs(j);
INSERT 0 1

# select reltoastrelid::regclass from pg_class where relname = 'test_toast';
  pg_toast.pg_toast_216572
# \d+ pg_toast.pg_toast_216572
TOAST table "pg_toast.pg_toast_216572"
   Column    |  Type   | Storage
-------------+---------+---------
 chunk_id    | oid     | plain
 chunk_seq   | integer | plain
 chunk_data  | bytea   | plain
```

PGDAY RUSSIA '16

TOAST cleanup ?

```
update products set description = description || ''
where id >= ... and id < ...;
vacuum products;
```

PGDAY RUSSIA '16

Tuples storage

- 23 bytes tuple header
- 1 byte padding or null bitmask
- total tuple size is multiple of MAXALIGN (8 bytes on x64)

Paddings

Какая таблица меньше?

```
test_paddings (
id integer,
is_active bool,
created_at timestamp,
is_removed bool,
updated_at timestamp
);
```

```
test_paddings2 (
id integer,
created_at timestamp,
updated_at timestamp,
is_active bool,
is_removed bool
);
```

```
test_paddings3 (
id integer,
is_active bool,
is_removed bool,
created_at timestamp,
updated_at timestamp
);
```

```
insert into test_paddings* ...
INSERT 0 100000
```

PGDAY RUSSIA '16

Paddings

Какая таблица меньше?

test_paddings (
id integer,
is_active bool,
created_at timestamp,
is_removed bool,
updated_at timestamp
);

5912 kB

test_paddings2 (
id integer,
created_at timestamp,
updated_at timestamp,
is_active bool,
is_removed bool
);

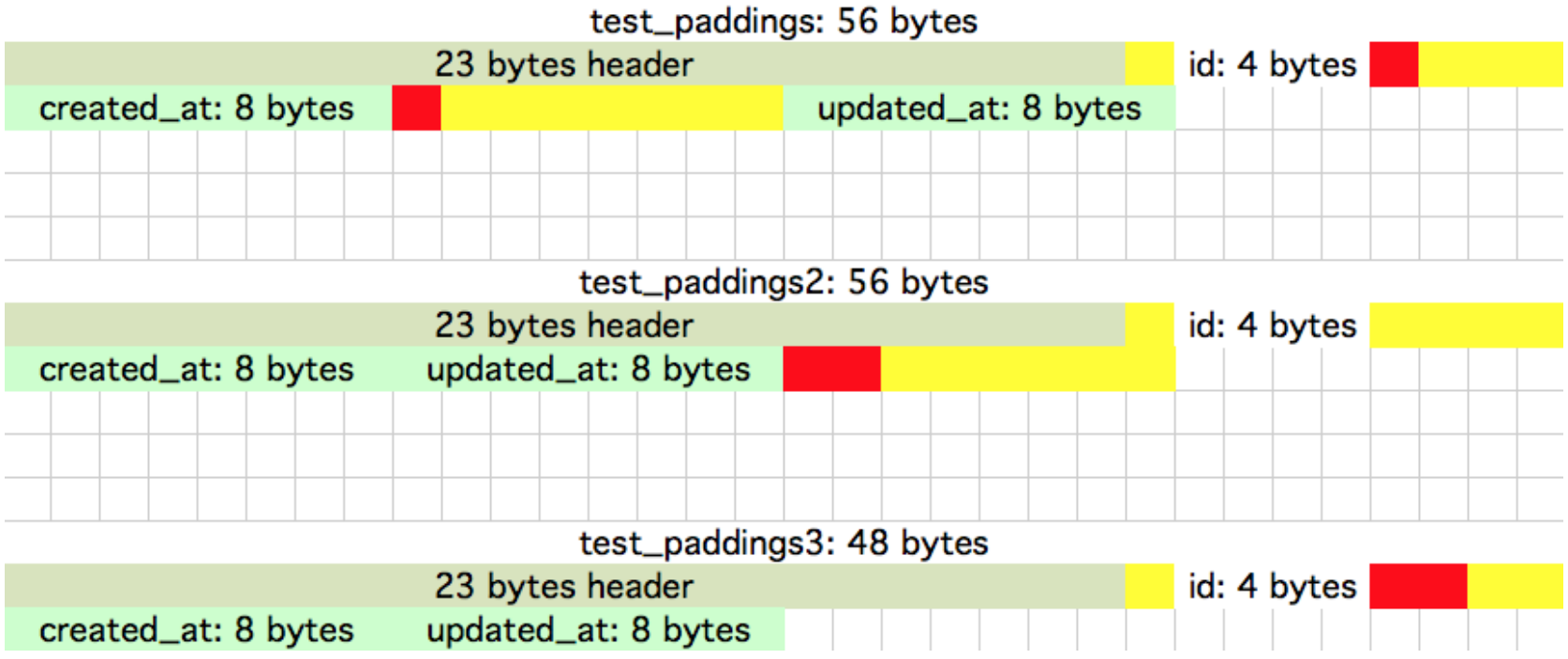
5912 kB

test_paddings3 (
id integer,
is_active bool,
is_removed bool,
created_at timestamp,
updated_at timestamp
);

5120 kB

PGDAY RUSSIA '16

Paddings



PGDAY RUSSIA '16

Paddings

```
select ... from pg_type where typname in (...) order by typlen;
  typname  | typlen | typalign
-----------+--------+----------
 json      |     -1 | 4
 jsonb     |     -1 | 4
 numeric   |     -1 | 4
 varchar   |     -1 | 4
 text      |     -1 | 4
 bool      |      1 | 1
 int2      |      2 | 2
 int4      |      4 | 4
 date      |      4 | 4
 float8    |      8 | 8
 timestamp |      8 | 8
 int8      |      8 | 8
```

PGDAY RUSSIA '16

Empty values

```
name         | pg_column_size
-------------+----------------
null         | 0[2]
''::text     | 4
'{}'::json   | 6
'{}'::jsonb  | 8
'{}'::int[]  | 16
```

[2] when less than 8 nullable fields in a row

PGDAY RUSSIA '16

Emergency? Monitoring!

- check long running transactions
- wal_keep_segments, checkpoint_segments (max_wal_size)
- reserved space
- remove/compress logs
- check biggest tables/indexes
- move to another tablespace

PGDAY RUSSIA '16

Not an emergency?

- Знаем размеры типов, tuple header, paddings
- Удаляем/архивируем ненужное (таблицы, индексы)
- Боремся с bloat
- Следим за местом
- Планируем обновление дисков заранее

PGDAY RUSSIA '16

Useful links

- PostgreSQL Manual 63.1. Database File Layout
- Different Approaches for MVCC used in well known Databases
- PostgreSQL Manual 63.2. TOAST
- github.com/pgexperts/pgx_scripts
- github.com/reorg/pg_repack
- github.com/grayhemp/pgtoolkit
- github.com/PostgreSQL-Consulting/pgcompacttable
- github.com/PostgreSQL-Consulting/pg-utils
- www.slideshare.net/alexius2/

PGDAY RUSSIA '16

Questions?

alexey.ermakov@postgresql-consulting.com