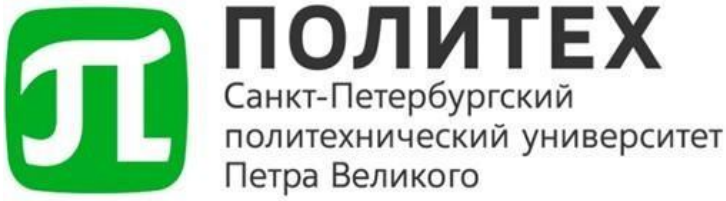
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»
ВШ программной инженерии



ОТЧЕТ ЛР3

по дисциплине “Системы управления базами данных”

Выполнила
студентка гр. 3530202/00201 Козлова Е. А.

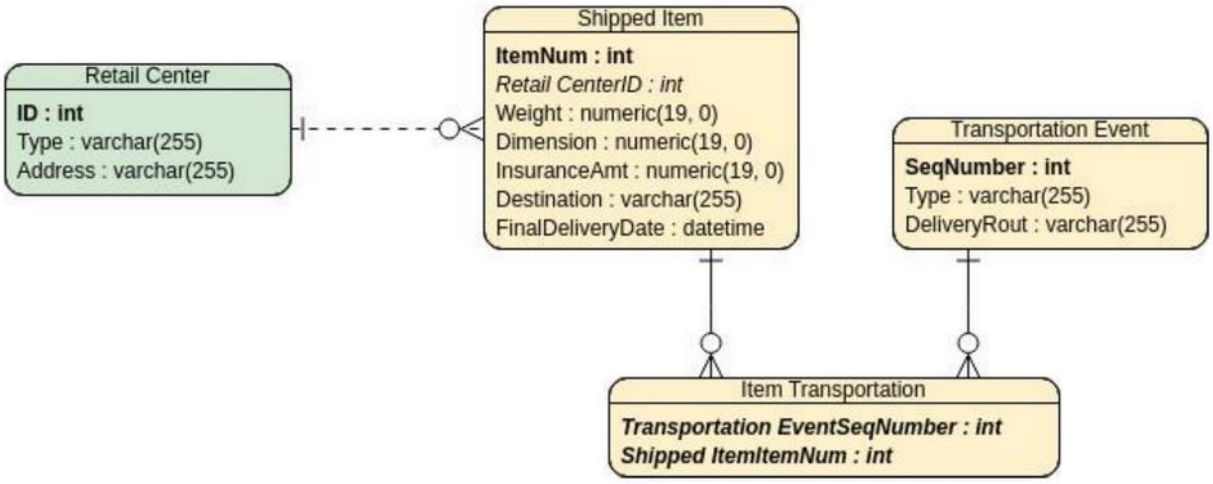
Руководитель Прокофьев О. В.

Санкт-Петербург
2023 г.

Оглавление

Практическое задание 3.1

ER диаграмма



Скрипт-генератор хранимой процедуры с использованием plpython3u для таблиц аналитической БД

```
CREATE TABLE retailCenter (
ID int PRIMARY KEY GENERATED ALWAYS AS IDENTITY,
TYPE varchar(255),
address varchar(255)
);

CREATE TABLE shippedItem (
itemNum int PRIMARY KEY GENERATED ALWAYS AS IDENTITY,
retailCenterId int REFERENCES retailCenter (id),
weight numeric(19, 0),
dimension numeric(19, 0),
insuranceAmt numeric(19, 0),
destination varchar(255),
final_delivery_date timestamp WITHOUT time zone NOT NULL
);
```

```
CREATE EXTENSION IF NOT EXISTS plpython3u;
CREATE OR REPLACE PROCEDURE insertDataRetail()
LANGUAGE plpython3u
AS $$
IMPORT random
typeDict = ["Corner Store", "Convenience center", "Neighborhood center", "Community center", "Lifestyle center"]
addressDict = ["694 Sugar Ave", "9954 Jockey Hollow Dr.", "66 Glenwood Avenue", "827 Kirkland St.", "97 S. Lawrence St.",
               "74 North Parker St.", "9180 Rockville Drive", "123 Greenview Dr.", "7296 Greenview Drive", "95 Lancaster St.",
               "90 South St.", "56 East Lower River Court", "361 Applegate Street", "684 W. Foster Ave.", "684 W. Foster Ave.",
               "75 Madison Street", "7869 Pawnee Avenue", "77 Longbranch Drive", "7183 W. West Ave.", "91 West Lane",
               "18 South Ketch Harbour Street", "1 Cherry Hill St.", "420 East Greenrose Drive", "22 Old Eagle Ave.",
               "197 Sycamore Street", "791 Dogwood Ave.", "72 East Highland St.", "62 Birch Hill Court", "400 Thorne Dr.",
               "73 East Cedar St.", "247 Chapel Ave.", "62 Edgewater Ave.", "453 Wellington Ave.", "7730 Bayport Drive",
               "657 Shadow Brook Lane", "7904 N. Prairie St.", "238 Willow Avenue", "347 Birchpond Street",
               "9574 Purple Finch Court", "462 South Linda St.", "970 W. Alderwood St.", "68 Kirkland Street", "93 6th St.",
               "362 Glenwood St.", "9308 Brickyard Lane", "739 Holly Drive", "165 Old Grant St.", "49 Birchpond Lane",
               "564 Rocky River Ave.", "256 Rockwell St.", "8390 Glendale Street", "36 W. Mulberry Ave.", "7670 Brook Dr.",
               "996 York Dr.", "607 Mulberry Road"]
cityDict = ["Groton", "Largo", "Cookeville", "Winter Garden", "Valley Stream", "Wenatchee", "Beverly", "Oswego", "Waterford",
            "Cherry Hill", "Scarsdale", "West Islip", "Dallas", "Creston", "Evanston", "Arroyo Grande", "San Simeon", "Avila Beach",
            "Chicago", "San Luis Obispo", "Derby", "Chelsea", "Merrimack", "Massapequa", "Billings", "Toledo", "Painesville",
            "Toms River", "Thibodaux", "Chester", "Brandon", "Lincolnton", "Tualatin", "Jackson Heights", "Southgate", "Orlando",
            "Uniondale", "Muskogee", "Bloomfield", "Commack", "Roanoke Rapids", "Miami", "Lewiston", "Milwaukee"]
FOR i IN RANGE(1000000) :
TYPE = typeDict[random.randint(0, len(typeDict) - 1)]
address = cityDict[random.randint(0, len(cityDict) - 1)] + ' ' + addressDict[random.randint(0, len(addressDict) - 1)]
plan = plpy.prepare("INSERT INTO retailCenter (type, address) VALUES($1, $2)", ["varchar(255)", "varchar(255)"])
plpy.execute(plan, [type, address])
$$;
CALL insertDataRetail();

CREATE OR REPLACE PROCEDURE insertDataShipped()
LANGUAGE plpython3u
AS $$
IMPORT random
IMPORT datetime
addressDict = ["44 Sunset Lane", "19 SW. Longbranch Dr.", "7004 Purple Finch Street", "1 Courtland St.",
               "14 E. Proctor Avenue", "60 Border Lane", "11 Union St.", "981 Brown Lane",
               "7818 William St.", "9604 Bishop Rd.", "95 Gulf Drive", "310 E. St Paul Ave.", "91 Fifth Ave.",
               "44 St Margarets Dr.", "9 Birch Hill Ave.", "120 Old Glenwood Road", "587 Whitemarsh Avenue",
               "41 Ryan Dr.", "200 East Dr.", "7700 Poor House Street", "43 Country Club Street",
               "9701 Wilson Ave.", "39 North Hilldale Ave.", "656 Rockland St.", "40 North Kirkland Court",
               "95 Jockey Hollow St.", "9897 Wood Drive", "8332 Race Street", "803 Marsh St.",
               "7198 Halifax Dr.", "54 Summer St.", "92 N. 6th St.", "290 Saxon St.", "7512 Sutor Dr.",
               "252 Hickory Ave.", "93 South Squaw Creek Rd.", "779 Cambridge Ave.", "110 Brickell Street",
               "8355 Poplar Street", "26 University St.", "38 Summerhouse Drive", "9212 Johnson Street",
               "113 South Nichols St.", "8877 State Dr.", "8727 Logan Street", "28 North Shipley Drive",
               "7258 Euclid St.", "916 Deerfield Road", "769 Homestead St.", "3 Woodsman Ave.",
               "502 Circle Ave.", "94 North Hill Ave.", "37 Vermont Avenue", "684 East Adams Street",
               "9082 Whitemarsh Dr.", "5 Taylor Ave.", "976 Illinois Rd.", "1 West Mill Pond Street",
               "435 Euclid St.", "48 Leatherwood Lane"]
cityDict = ["Derby", "Chelsea", "Merrimack", "Massapequa", "Billings", "Toledo", "Painesville",
            "Toms River", "Thibodaux", "Chester", "Brandon", "Lincolnton", "Tualatin", "Jackson Heights",
            "Southgate", "Orlando", "Uniondale", "Muskogee", "Bloomfield", "Commack", "Roanoke Rapids",
            "Miami", "Lewiston", "Milwaukee", "Groton", "Largo", "Cookeville", "Winter Garden",
            "Valley Stream", "Wenatchee", "Beverly", "Oswego", "Waterford", "Cherry Hill", "Scarsdale",
            "West Islip", "Dallas", "Creston", "Evanston", "Arroyo Grande", "San Simeon", "Avila Beach",
            "Chicago", "San Luis Obispo"]
FOR i IN RANGE(0, 1000000) :
centerId = random.randint(1, 1000000)
weight = random.getrandbits(63)
dimension = random.getrandbits(63)
insuranceAmt = random.getrandbits(63)
destination = cityDict[random.randint(0, len(cityDict) - 1)] + ' ' + addressDict[random.randint(0, len(addressDict) - 1)]
final_delivery_date = datetime.datetime.now() + datetime.timedelta(days=random.randint(-100, 100))
plan = plpy.prepare("INSERT INTO shippedItem (retailCenterId, weight, dimension, insuranceAmt, destination, final_delivery_date)
                     VALUES($1, $2, $3, $4, $5, $6)", ["int", "numeric(19, 0)", "numeric(19, 0)", "numeric(19, 0)",
                                                       "varchar(255)", "timestamp WITHOUT time zone"])
plpy.execute(plan, [centerId, weight, dimension, insuranceAmt, destination, final_delivery_date])
$$;

CALL insertDataShipped();
```

План запроса A с индексами/без индексов (однотабличный запрос с фильтрами)

```
EXPLAIN ANALYZE
SELECT * FROM retailCenter
WHERE address ILIKE '%' || 'north' || '%' AND TYPE = 'Corner Store';
CREATE INDEX retail_index ON retailCenter(type, address);
DROP INDEX retail_index
```

Без индекса

QUERY PLAN
Gather (cost=1000.00..18069.90 rows=7829 width=49) (actual time=0.476..163.687 rows=3688 loops=1)
Workers Planned: 2
Workers Launched: 2
-> Parallel Seq Scan on retailcenter (cost=0.00..16287.00 rows=3262 width=49) (actual time=0.161..155.533 rows=1229 loops=3)
Filter: (((address)::text ~~* '%north%'::text) AND ((type)::text = 'Corner Store'::text))
Rows Removed by Filter: 332104
Planning Time: 0.687 ms
Execution Time: 163.808 ms

С индексом

QUERY PLAN
Bitmap Heap Scan on retailcenter (cost=2221.38..15208.39 rows=7829 width=49) (actual time=17.864..143.307 rows=3688 loops=1)
Recheck Cond: ((type)::text = 'Corner Store'::text)
Filter: ((address)::text ~~* '%north%'::text)
Rows Removed by Filter: 196697
Heap Blocks: exact=10037
-> Bitmap Index Scan on retail_index (cost=0.00..2219.43 rows=196667 width=0) (actual time=15.975..15.975 rows=200385 loops=1)
Index Cond: ((type)::text = 'Corner Store'::text)
Planning Time: 3.529 ms
Execution Time: 143.560 ms

План запроса Б с индексами/без индексов (многотабличный запрос с фильтрами)

```
EXPLAIN ANALYZE
SELECT * FROM retailCenter
JOIN shippedItem ON retailCenter.id = shippedItem.retailCenterId
WHERE retailCenter.type = 'Convenience center' AND shippedItem.destination ILIKE '%' || 'miami' || '%';
CREATE INDEX item_index ON shippedItem(destination);
CREATE INDEX retail2_index ON retailCenter(type);
```

Без индексов

QUERY PLAN
Gather (cost=42109.53..54285.45 rows=7880 width=129) (actual time=566.558..592.100 rows=8893 loops=1)
Workers Planned: 2
Workers Launched: 2
-> Parallel Hash Join (cost=41109.53..52497.45 rows=3283 width=129) (actual time=560.077..583.540 rows=2964 loops=3)
Hash Cond: (retailcenter.id = shippeditem.retailcenterid)
-> Parallel Bitmap Heap Scan on retailcenter (cost=2275.05..13340.70 rows=82292 width=49) (actual time=6.993..22.270 rows=66610 loops=3)
Recheck Cond: ((type)::text = 'Convenience center'::text)
Heap Blocks: exact=3343
-> Bitmap Index Scan on retail_index (cost=0.00..2225.68 rows=197500 width=0) (actual time=6.102..6.102 rows=199830 loops=1)
Index Cond: ((type)::text = 'Convenience center'::text)
-> Parallel Hash (cost=38626.67..38626.67 rows=16625 width=80) (actual time=552.882..552.882 rows=15001 loops=3)
Buckets: 65536 Batches: 1 Memory Usage: 5632kB
-> Parallel Seq Scan on shippeditem (cost=0.00..38626.67 rows=16625 width=80) (actual time=0.381..549.697 rows=15001 loops=3)
Filter: ((destination)::text ~~* '%miami%'::text)
Rows Removed by Filter: 651666
Planning Time: 9.964 ms
Execution Time: 592.366 ms

С индексами

QUERY PLAN
Gather (cost=42053.53..54229.45 rows=7880 width=129) (actual time=481.817..499.150 rows=8893 loops=1)
Workers Planned: 2
Workers Launched: 2
-> Parallel Hash Join (cost=41053.53..52441.45 rows=3283 width=129) (actual time=475.937..491.525 rows=2964 loops=3)
Hash Cond: (retailcenter.id = shippeditem.retailcenterid)
-> Parallel Bitmap Heap Scan on retailcenter (cost=2219.05..13284.70 rows=82292 width=49) (actual time=5.255..13.776 rows=66610 loops=3)
Recheck Cond: ((type)::text = 'Convenience center'::text)
Heap Blocks: exact=3933
-> Bitmap Index Scan on retail2_index (cost=0.00..2169.68 rows=197500 width=0) (actual time=4.340..4.340 rows=199830 loops=1)
Index Cond: ((type)::text = 'Convenience center'::text)
-> Parallel Hash (cost=38626.67..38626.67 rows=16625 width=80) (actual time=470.453..470.453 rows=15001 loops=3)
Buckets: 65536 Batches: 1 Memory Usage: 5632kB
-> Parallel Seq Scan on shippeditem (cost=0.00..38626.67 rows=16625 width=80) (actual time=0.455..467.441 rows=15001 loops=3)
Filter: ((destination)::text ~~* '%miami%'::text)
Rows Removed by Filter: 651666
Planning Time: 1.682 ms
Execution Time: 499.358 ms

План запроса B с индексами/без индексов (полнотекстовый поиск)

```
EXPLAIN ANALYZE
SELECT address FROM retailCenter
WHERE to_tsvector('english', address) @@ to_tsquery('north');
CREATE INDEX gin_idx_retail on retailCenter using gin(to_tsvector('english', address));
DROP INDEX gin_idx_retail;
```

Без индексов

QUERY PLAN
Gather (cost=1000.42..223806.09 rows=5000 width=28) (actual time=37.885..946.754 rows=18388 loops=1)
Workers Planned: 2
Workers Launched: 2
-> Parallel Index Only Scan using retail_index on retailcenter (cost=0.42..222306.09 rows=2083 width=28) (actual time=17.255..941.473 rows=6129 loops=3)
Filter: (to_tsvector('english'::regconfig, (address)::text) @@ to_tsquery('north'::text))
Rows Removed by Filter: 327204
Heap Fetches: 0
Planning Time: 24.748 ms
Execution Time: 947.194 ms

С индексами

QUERY PLAN
Bitmap Heap Scan on retailcenter (cost=71.00..11061.51 rows=5000 width=28) (actual time=5.974..21.730 rows=18388 loops=1)
Recheck Cond: (to_tsvector('english'::regconfig, (address)::text) @@ to_tsquery('north'::text))
Heap Blocks: exact=8514
-> Bitmap Index Scan on gin_idx_retail (cost=0.00..69.75 rows=5000 width=0) (actual time=3.317..3.318 rows=18388 loops=1)
Index Cond: (to_tsvector('english'::regconfig, (address)::text) @@ to_tsquery('north'::text))
Planning Time: 0.870 ms
Execution Time: 22.907 ms

Оптимизация

```
-- Удаление старых данных - разделение на таблицы или индекс по дате
CREATE INDEX shippeditem_index ON shippedItem (final_delivery_date);
CREATE TABLE shippedItem_old PARTITION OF shippedItem
FOR VALUES FROM ('2023-01-01') to ('2023-05-09');
-- Вставка данных
ALTER TABLE shippedItem DISABLE TRIGGER ALL;
-- или ALTER INDEX ALL DISABLE для отключения только индексов, если триггеры нужны
-- Выполняем вставку данных с отключенными индексами
ALTER TABLE shippedItem ENABLE TRIGGER ALL;
-- Чтение данных - индексы на столбцы со связями
CREATE INDEX shipped_item_item_num_idx ON PUBLIC.shippedItem ("ItemNum");
CREATE INDEX shipped_item_item_num_idx ON PUBLIC.shippedItem ("RetailerCenterID");
```

Практическое задание 3.2

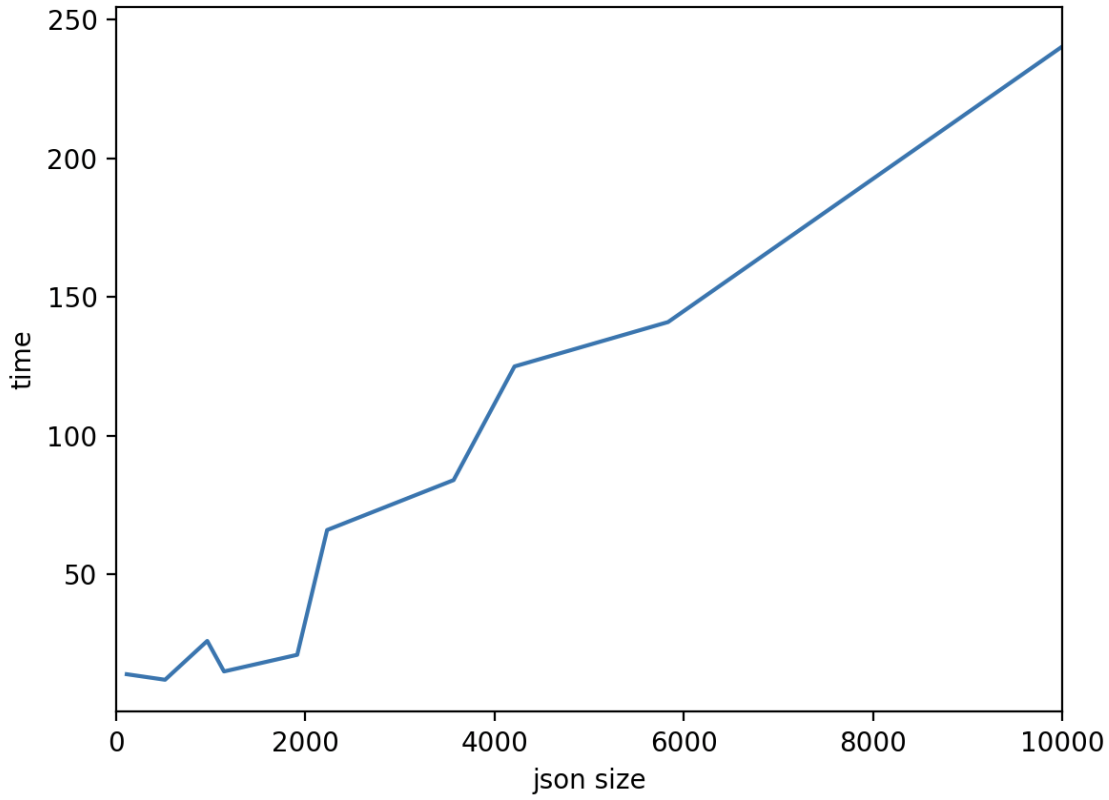
Запросы

```sql
-- Выбрать количество ролей для каждого актёра
SELECT "Name", jsonb_array_length("Roles" -> 'roles') AS RolesCount
FROM public."Actors";
-- Выбрать актеров из фильма Teleton
SELECT * FROM public."Actors"
WHERE "Roles" @> '{"roles": [{"Title": "Teleton"}]}';
-- Выбрать всех актёров у кого есть фильмы 2000 года и более
SELECT * FROM public."Actors"
WHERE jsonb_path_exists("Roles", '$.roles[*] ? (@.Year >= "2000")');
```

График зависимости времени (мс) от размера jsonb

ось X-длина jsonb (байт)

ось Y-время доступа к полю primaryName (мс)



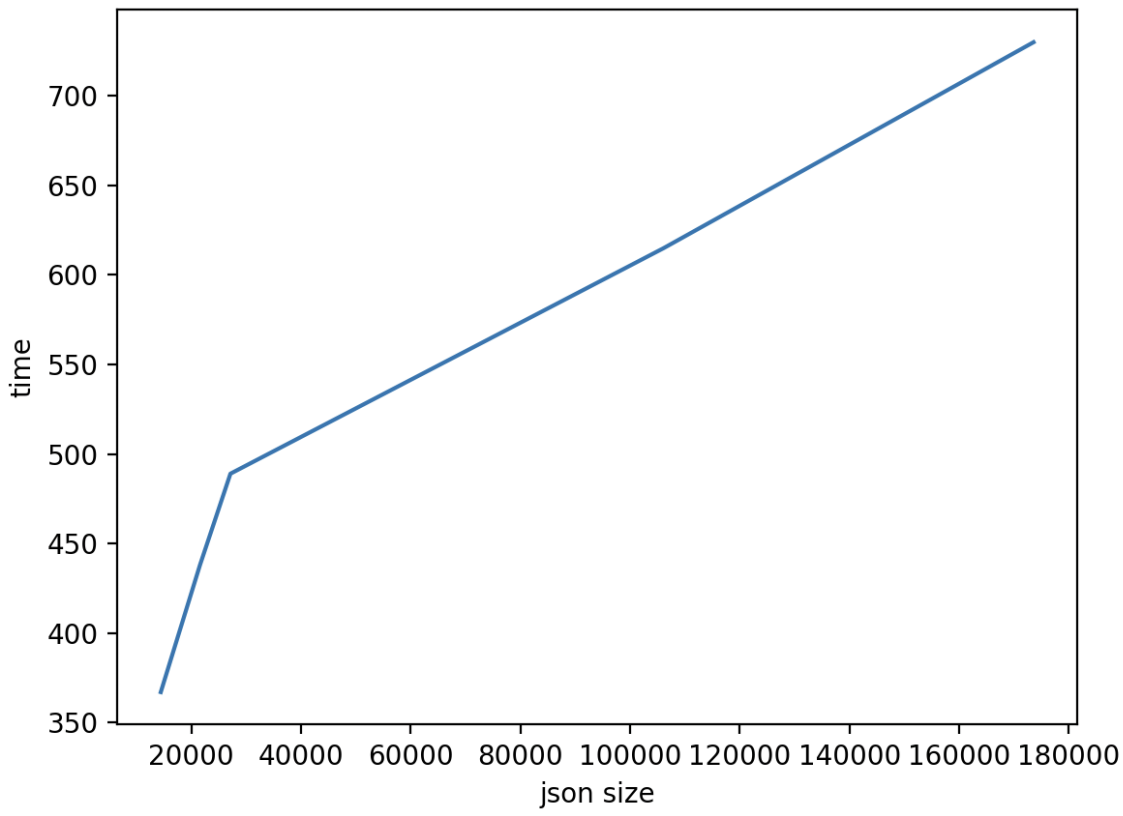


Таблица изменения размера БД для длинного и короткого jsonb (при изменении атрибута year по одной из ролей)

Размер json	112	173546
Размер таблицы до	907026432	907026432
Размер таблицы после	907026432	907026432
Прирост	0	0
WAL до	21444696936	22498369120
WAL после	21444697280	22498495976