Мамонов Антон 2ИСиП-19-1

Создание таблицы Sales

Query Editor

```
1  CREATE TABLE Sales
2  (s_customer_id serial primary key,
3   s_amt numeric(9,2),
4   s_date date)
```

Заполнение данными

Query Editor

```
1  insert into sales values (
2   000001, 1487.98, '12.02.2021'),
3   (000002, 1642.76, '14.05.2021'),
4   (000003, 47.01, '3.08.2021'),
5   (000004, 12387.99, '1.12.2021'),
6   (000005, 123, '19.02.2021'),
7   (000006, 1083.09, '22.04.2021'),
8   (000007, 356, '6.09.2021'),
9   (000008, 1342.9, '17.11.2021')
```

Создаем четыре дочерние таблицы для каждого квартала и ограничения проверки.

Query Editor

```
1  create table sales2021_q1 () inherits (sales);
2  create table sales2021_q2 () inherits (sales);
3  create table sales2021_q3 () inherits (sales);
4  create table sales2021_q4 () inherits (sales);
```

```
1  alter table sales2021_q1 add constraint check_date check (
2  s_date >= '2021-01-01 00:00:00'::timestamp and
3  s_date <  '2021-04-01 00:00:00'::timestamp
4  );
5
6  alter table sales2021_q2 add constraint check_date check (
7  s_date >= '2021-04-01 00:00:00'::timestamp and
8  s_date <  '2021-07-01 00:00:00'::timestamp
9  );
10
11 alter table sales2021_q3 add constraint check_date check (
12 s_date >= '2021-07-01 00:00:00'::timestamp and
13 s_date <  '2021-10-01 00:00:00'::timestamp
14 );
15
16 alter table sales2021_q4 add constraint check_date check (
17 s_date >= '2021-10-01 00:00:00'::timestamp and
18 s_date <  '2022-01-01 00:00:00'::timestamp
19 );
```

Создаем триггер на вставку записей в Sales

```
create function dates_ins () returns trigger as $$
 begin
    if new.s_date >= '2021-01-01 00:00:00'::timestamp and
       new.s_date <  '2021-04-01 00:00:00'::timestamp
    then
       insert into sales2021_q1 select new.*;
    elsif new.s_date >= '2021-04-01 00:00:00'::timestamp and
          new.s_date <  '2021-07-01 00:00:00'::timestamp
    then
       insert into sales2021_q2 select new.*;
    elsif new.s_date >= '2021-07-01 00:00:00'::timestamp and
          new.s_date <  '2021-10-01 00:00:00'::timestamp
    then
       insert into sales2021_q3 select new.*;
    elsif new.s_date >= '2021-10-01 00:00:00'::timestamp and
          new.s_date <  '2022-01-01 00:00:00'::timestamp
    then
       insert into sales2021_q4 select new.*;
    end if;
    return null;
end; $$ language plpgsql;
```

```
create trigger dates_ins_trg
    before insert on sales
    for each row
    execute procedure dates_ins();
```

Проверяем работу триггера на новые данные

```
insert into sales values (
9, 1238.8, '07-01-2021'),
(10, 2498.87, '21-04-2021'),
(11, 12345, '8-7-2021'),
(12, 457, '12-11-2021'),
(13, 789, '01-3-2021'),
(14, 6666.98, '28-6-2021')
```

	s_customer_id integer	s_amt numeric (9,2)	s_date date
1	9	1238.80	2021-01-07
2	13	789.00	2021-03-01

	s_customer_id integer	s_amt numeric (9,2)	s_date date
1	11	12345.00	2021-07-08

	s_customer_id integer	s_amt numeric (9,2)	s_date date
1	10	2498.87	2021-04-21
2	14	6666.98	2021-06-28

	s_customer_id integer	s_amt numeric (9,2)	s_date date
1	12	457.00	2021-11-12

Все работает

Перенести все строки из Sales в соответствующие секции.

```
with del as (
    delete from ONLY sales
     where s_date >= '2021-01-01 00:00:00'::timestamp and
           s_date <  '2021-04-01 00:00:00'::timestamp
    returning *
 )
insert into sales2021_q1 select * from del;

with del as (
    delete from ONLY sales
     where s_date >= '2021-04-01 00:00:00'::timestamp and
           s_date <  '2021-07-01 00:00:00'::timestamp
    returning *
 )
insert into sales2021_q2 select * from del;
```

```
with del as (
    delete from ONLY sales
     where s_date >= '2021-07-01 00:00:00'::timestamp and
           s_date <  '2021-10-01 00:00:00'::timestamp
    returning *
 )
insert into sales2021_q3 select * from del;

with del as (
    delete from ONLY sales
     where s_date >= '2021-10-01 00:00:00'::timestamp and
           s_date <  '2022-01-01 00:00:00'::timestamp
    returning *
)
insert into sales2021_q4 select * from del;
```

Данные, внесенные ранее, перенеслись, все работает

	s_customer_id integer	s_amt numeric (9,2)	s_date date
1	9	1238.80	2021-01-07
2	13	789.00	2021-03-01
3	1	1487.98	2021-02-12
4	5	123.00	2021-02-19

Решаем задачу с помощью партицирования

Создаем новую таблицу

```
CREATE TABLE sales_p
(s_customer_id serial,
 s_amt numeric(9,2),
 s_date date)
 PARTITION BY RANGE (s_date);
```

```
CREATE TABLE date_2021q1 PARTITION OF sales_p
    FOR VALUES FROM ('2021-01-01') TO ('2021-03-31');
CREATE TABLE date_2021q2 PARTITION OF sales_p
    FOR VALUES FROM ('2021-04-01') TO ('2021-06-30');
CREATE TABLE date_2021q3 PARTITION OF sales_p
    FOR VALUES FROM ('2021-07-01') TO ('2021-09-30');
CREATE TABLE date_2021q4 PARTITION OF sales_p
    FOR VALUES FROM ('2021-10-01') TO ('2021-12-31');
```

```
1  CREATE INDEX ON date_2021q1 (s_date);
2  CREATE INDEX ON date_2021q2 (s_date);
3  CREATE INDEX ON date_2021q3 (s_date);
4  CREATE INDEX ON date_2021q4 (s_date);
```

```
1  insert into sales_p values (
2      1, 432.03, '2021-01-02'),
3      (2, 3293.83, '2021-03-07'),
4      (3, 829.90, '2021-05-09'),
5      (4, 1839.33, '2021-06-17'),
6      (5, 282.39, '2021-08-02'),
7      (6, 293.29, '2021-10-25'),
8      (7, 1772.20, '2021-12-21');
```

```
1  select tableoid::regclass, count(*) from sales_p group by tableoid;
```

Результат | План выполнения | Notifications | Сообщения | История запросов

	tableoid regclass	count bigint
1	date_2021q3	1
2	date_2021q1	2
3	date_2021q2	2
4	date_2021q4	2

```
1  explain (costs off) select * from sales_p where s_date < '2021-05-09'::date;
```

Результат | План выполнения | Notifications | Сообщения | История запросов

	QUERY PLAN text
1	Append
2	-> Bitmap Heap Scan on date_2021q1 sales_p_1
3	Recheck Cond: (s_date < '2021-05-09'::date)
4	-> Bitmap Index Scan on date_2021q1_s_date_idx
5	Index Cond: (s_date < '2021-05-09'::date)
6	-> Bitmap Heap Scan on date_2021q2 sales_p_2
7	Recheck Cond: (s_date < '2021-05-09'::date)
8	-> Bitmap Index Scan on date_2021q2_s_date_idx
9	Index Cond: (s_date < '2021-05-09'::date)

Оцените план выполнения запроса при первом и втором варианте решения.

Первый вариант

QUERY PLAN

Append

-> Seq Scan on sales sales_1

Filter: (s_date < '2021-05-09'::date)

-> Seq Scan on sales_2021q1 sales_2

Filter: (s_date < '2021-05-09'::date)

-> Seq Scan on sales_2021q2 sales_3

Filter: (s_date < '2021-05-09'::date)

Второй вариант

QUERY PLAN

Append

-> Bitmap Heap Scan on sales_2021q1 sales_1

Recheck Cond: (s_date < '2021-05-09'::date)

-> Bitmap Index Scan on sales_2021q1_s_date_idx

Index Cond: (s_date < '2021-05-09'::date)

-> Bitmap Heap Scan on sales_2021q2 sales_2

Recheck Cond: (s_date < '2021-05-09'::date)

-> Bitmap Index Scan on sales_2021q2_s_date_idx

Index Cond: (s_date < '2021-05-09'::date)