МИНИСТЕРСТВО НАУКИ
И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

НГТУ НЭТИ | **Факультет прикладной математики и информатики**

Кафедра теоретической и прикладной информатики

Лабораторная работа № 7

по дисциплине «Администрирование информационных систем»

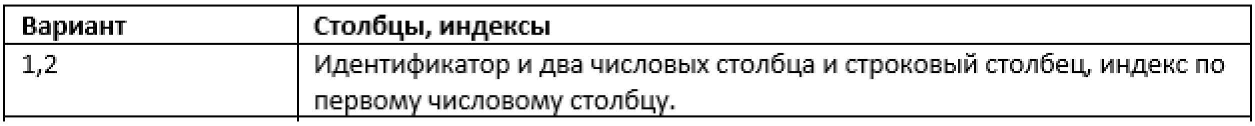
ВЫПОЛНЕНИЕ ЗАПРОСОВ

Бригада 2	ХАЙДАЕВ К.Е.
Группа ПМИ-82	ЗЯБЛИЦЕВА У.П.
Вариант 2	

Преподаватели	АВРУНЕВ О.Е.

Новосибирск, 2022

1 Создать таблицу со следующей структурой



Вариант	Столбцы, индексы
1,2	Идентификатор и два числовых столбца и строковый столбец, индекс по первому числовому столбцу.

Create table bookings.lab7

(id INT GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,

num1 INT,

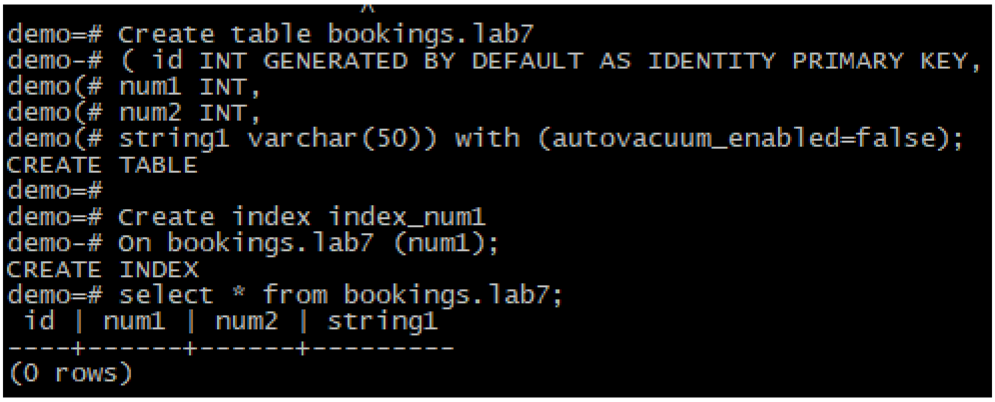
num2 INT,

string1 varchar(50)) with (autovacuum_enabled=false);

Create index index_num1

On bookings.lab7 (num1);

select * from bookings.lab7;



```
demo=# Create table bookings.lab7
demo-# ( id INT GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
demo(# num1 INT,
demo(# num2 INT,
demo(# string1 varchar(50)) with (autovacuum_enabled=false);
CREATE TABLE
demo=#
demo=# Create index index_num1
demo-# On bookings.lab7 (num1);
CREATE INDEX
demo=# select * from bookings.lab7;
 id | num1 | num2 | string1
----+------+------+---------
(0 rows)
```

2 Заполнить таблицу данными (150 000 строк)

do $$

begin

for i in 1..150000 loop

insert into bookings.lab7(num1,num2,string1) values (random() * 25, random() * 250,array_to_string(ARRAY(SELECT chr((ascii('A') + round(random() * 25)) :: integer) FROM generate_series(1,15)), ''));

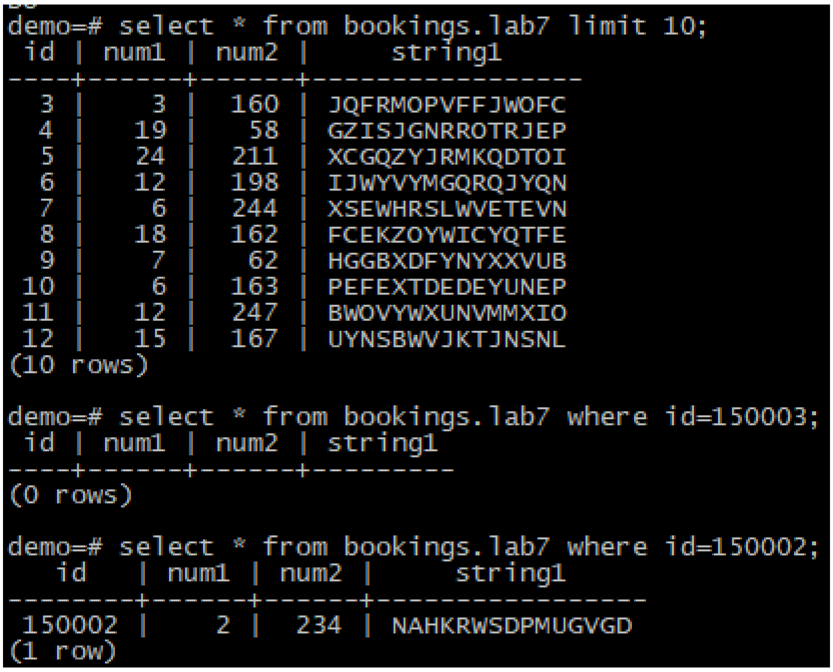
end loop;

end;

$$ language plpgsql;

Select * from bookings.lab7 limit 10;

Select * from bookings.lab7 where id=150002;

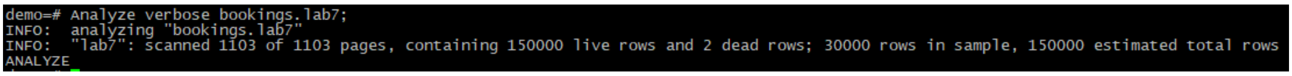


```
demo=# select * from bookings.lab7 limit 10;
 id | num1 | num2 |     string1
----+------+------+------------------
  3 |    3 |  160 | JQFRMOPVFFJWOFC
  4 |   19 |   58 | GZISJGNRROTRJEP
  5 |   24 |  211 | XCGQZYJRMKQDTOI
  6 |   12 |  198 | IJWYVYMGQRQJYQN
  7 |    6 |  244 | XSEWHRSLWVETEVN
  8 |   18 |  162 | FCEKZOYWICYQTFE
  9 |    7 |   62 | HGGBXDFYNYXXVUB
 10 |    6 |  163 | PEFEXTDEDEYUNEP
 11 |   12 |  247 | BWOVYWXUNVMMXIO
 12 |   15 |  167 | UYNSBWVJKTJNSNL
(10 rows)

demo=# select * from bookings.lab7 where id=150003;
 id | num1 | num2 | string1
----+------+------+---------
(0 rows)

demo=# select * from bookings.lab7 where id=150002;
   id    | num1 | num2 |     string1
---------+------+------+-----------------
 150002 |    2 |  234 | NAHKRWSDPMUGVGD
(1 row)
```

3 Привести значения статистики данных, для таблицы в целом (количество строк и страниц) и для каждого из столбцов (кол-во различных значений, наиболее часто встречающиеся значения, корреляция, ширина столбца).

Значения для таблицы в целом

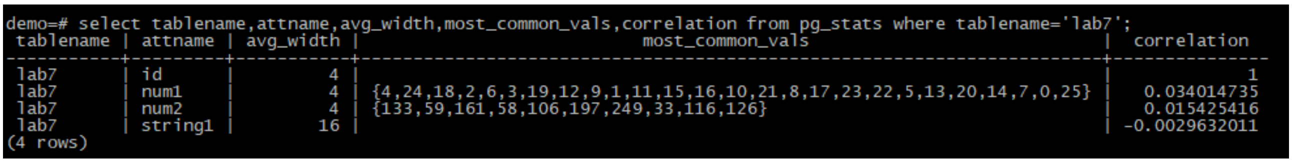
Analyze verbose bookings.lab7;



```
demo=# Analyze verbose bookings.lab7;
INFO:  analyzing "bookings.lab7"
INFO:  "lab7": scanned 1103 of 1103 pages, containing 150000 live rows and 2 dead rows; 30000 rows in sample, 150000 estimated total rows
ANALYZE
```

Для каждого из столбцов

select tablename,attname,avg_width,most_common_vals,correlation from pg_stats where tablename='lab7';



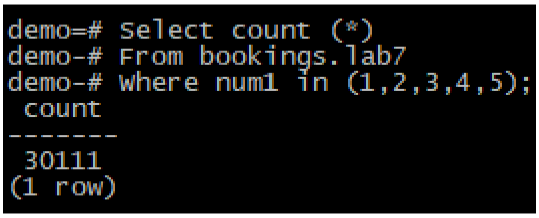
```
demo=# select tablename,attname,avg_width,most_common_vals,correlation from pg_stats where tablename='lab7';
 tablename | attname | avg_width |                              most_common_vals                               |  correlation
-----------+---------+-----------+-----------------------------------------------------------------------------+---------------
 lab7      | id      |         4 |                                                                             |             1
 lab7      | num1    |         4 | {4,24,18,2,6,3,19,12,9,1,11,15,16,10,21,8,17,23,22,5,13,20,14,7,0,25} |   0.034014735
 lab7      | num2    |         4 | {133,59,161,58,106,197,249,33,116,126}                                      |   0.015425416
 lab7      | string1 |        16 |                                                                             | -0.0029632011
(4 rows)
```

4 Выполнить запрос: получение строк из таблицы, условие вида “in ( )” на проиндексированный столбец, кол-во элементов в условии in – 5.

Select count (*)

From bookings.lab7

Where num1 in (1,2,3,4,5);



```
demo=# select count (*)
demo-# From bookings.lab7
demo-# where num1 in (1,2,3,4,5);
 count
-------
 30111
(1 row)
```

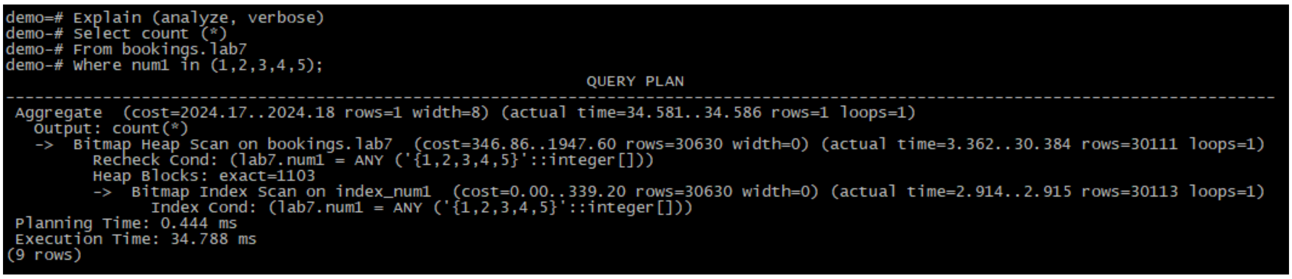
5 Привести план выполнения и время выполнения

Explain (analyze, verbose)

Select count (*)

From bookings.lab7

Where num1 in (1,2,3,4,5);



```
demo=# Explain (analyze, verbose)
demo-# Select count (*)
demo-# From bookings.lab7
demo-# where num1 in (1,2,3,4,5);
                                                            QUERY PLAN
---------------------------------------------------------------------------------------------------------------------------------------
 Aggregate  (cost=2024.17..2024.18 rows=1 width=8) (actual time=34.581..34.586 rows=1 loops=1)
   Output: count(*)
   ->  Bitmap Heap Scan on bookings.lab7  (cost=346.86..1947.60 rows=30630 width=0) (actual time=3.362..30.384 rows=30111 loops=1)
         Recheck Cond: (lab7.num1 = ANY ('{1,2,3,4,5}'::integer[]))
         Heap Blocks: exact=1103
         ->  Bitmap Index Scan on index_num1  (cost=0.00..339.20 rows=30630 width=0) (actual time=2.914..2.915 rows=30113 loops=1)
               Index Cond: (lab7.num1 = ANY ('{1,2,3,4,5}'::integer[]))
 Planning Time: 0.444 ms
 Execution Time: 34.788 ms
(9 rows)
```

6 Увеличить количество строк в таблице в два раза

do $$

begin

for i in 1..150000 loop

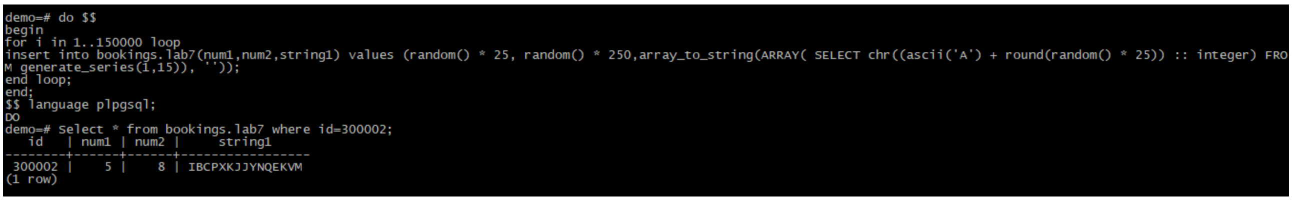
insert into bookings.lab7(num1,num2,string1) values (random() * 25, random() * 250,array_to_string(ARRAY(SELECT chr((ascii('A') + round(random() * 25)) :: integer) FROM generate_series(1,15)), ''));

end loop;

end;

$$ language plpgsql;

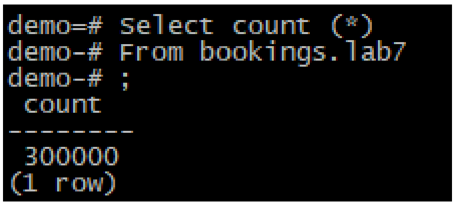
Select * from bookings.lab7 where id=300002;



```
demo=# do $$
begin
for i in 1..150000 loop
insert into bookings.lab7(num1,num2,string1) values (random() * 25, random() * 250,array_to_string(ARRAY( SELECT chr((ascii('A') + round(random() * 25)) :: integer) FRO
M generate_series(1,15)), ''));
end loop;
end;
$$ language plpgsql;
DO
demo=# Select * from bookings.lab7 where id=300002;
   id   | num1 | num2 |     string1
--------+------+------+-----------------
 300002 |    5 |    8 | IBCPXKJJYNQEKVM
(1 row)
```

Select count (*)

From bookings.lab7;



```
demo=# select count (*)
demo-# From bookings.lab7
demo-# ;
 count
--------
 300000
(1 row)
```

7 Повторно привести план и время выполнения запроса из п.4

Select count (*)

From bookings.lab7

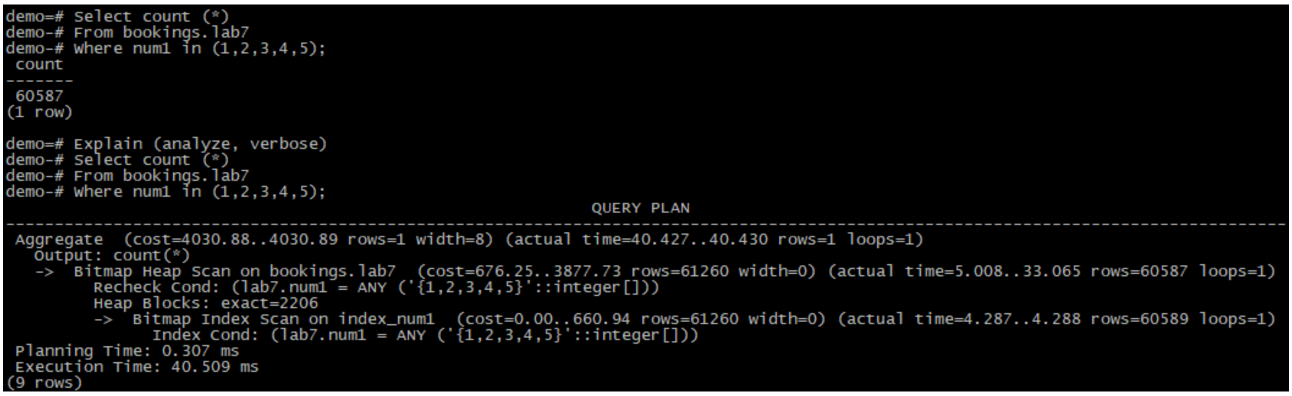
Where num1 in (1,2,3,4,5);

Explain (analyze, verbose)

Select count (*)

From bookings.lab7

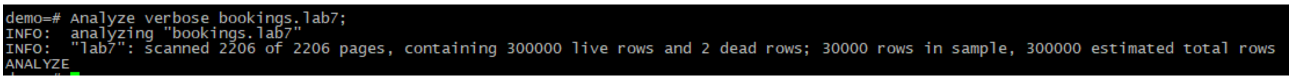
Where num1 in (1,2,3,4,5);



```
demo=# Select count (*)
demo-# From bookings.lab7
demo-# Where num1 in (1,2,3,4,5);
 count
-------
 60587
(1 row)

demo=# Explain (analyze, verbose)
demo-# Select count (*)
demo-# From bookings.lab7
demo-# Where num1 in (1,2,3,4,5);
                                                           QUERY PLAN
---------------------------------------------------------------------------------------------------------------------------------
 Aggregate  (cost=4030.88..4030.89 rows=1 width=8) (actual time=40.427..40.430 rows=1 loops=1)
   Output: count(*)
   ->  Bitmap Heap Scan on bookings.lab7  (cost=676.25..3877.73 rows=61260 width=0) (actual time=5.008..33.065 rows=60587 loops=1)
         Recheck Cond: (lab7.num1 = ANY ('{1,2,3,4,5}'::integer[]))
         Heap Blocks: exact=2206
         ->  Bitmap Index Scan on index_num1  (cost=0.00..660.94 rows=61260 width=0) (actual time=4.287..4.288 rows=60589 loops=1)
               Index Cond: (lab7.num1 = ANY ('{1,2,3,4,5}'::integer[]))
 Planning Time: 0.307 ms
 Execution Time: 40.509 ms
(9 rows)
```

8 Собрать статистику данных для таблицы

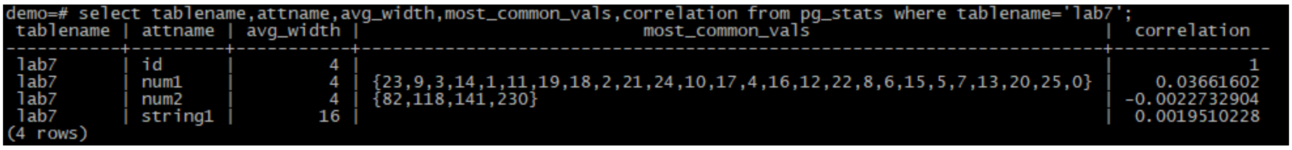
Analyze verbose bookings.lab7;



```
demo=# Analyze verbose bookings.lab7;
INFO:  analyzing "bookings.lab7"
INFO:  "lab7": scanned 2206 of 2206 pages, containing 300000 live rows and 2 dead rows; 30000 rows in sample, 300000 estimated total rows
ANALYZE
```

9 Привести значения статистики данных из п.3

select tablename,attname,avg_width,most_common_vals,correlation from pg_stats where tablename='lab7';

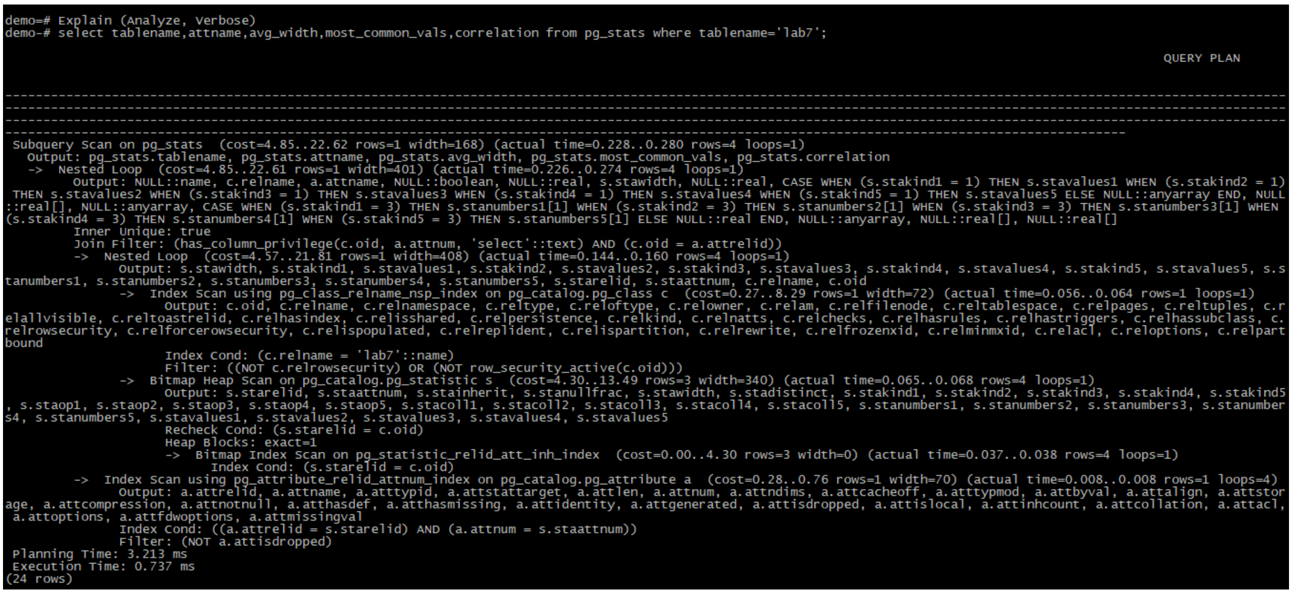


```
demo=# select tablename,attname,avg_width,most_common_vals,correlation from pg_stats where tablename='lab7';
 tablename | attname | avg_width |                          most_common_vals                          |  correlation
-----------+---------+-----------+--------------------------------------------------------------------+---------------
 lab7      | id      |         4 |                                                                    |             1
 lab7      | num1    |         4 | {23,9,3,14,1,11,19,18,2,21,24,10,17,4,16,12,22,8,6,15,5,7,13,20,25,0} |    0.03661602
 lab7      | num2    |         4 | {82,118,141,230}                                                   | -0.0022732904
 lab7      | string1 |        16 |                                                                    |  0.0019510228
(4 rows)
```

10 Привести план и время выполнения запроса. Сравнить планируемое и реальное время выполнения для пунктов 5,7,10

Explain (Analyze, Verbose)

select tablename,attname,avg_width,most_common_vals,correlation from pg_stats where tablename='lab7';



```
demo=# Explain (Analyze, Verbose)
demo-# select tablename,attname,avg_width,most_common_vals,correlation from pg_stats where tablename='lab7';
                                                                                                                                  QUERY PLAN

----------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------
 Subquery Scan on pg_stats  (cost=4.85..22.62 rows=1 width=168) (actual time=0.228..0.280 rows=4 loops=1)
   Output: pg_stats.tablename, pg_stats.attname, pg_stats.avg_width, pg_stats.most_common_vals, pg_stats.correlation
   ->  Nested Loop  (cost=4.85..22.61 rows=1 width=401) (actual time=0.226..0.274 rows=4 loops=1)
         Output: NULL::name, c.relname, a.attname, NULL::boolean, NULL::real, s.stawidth, NULL::real, CASE WHEN (s.stakind1 = 1) THEN s.stavalues1 WHEN (s.stakind2 = 1)
 THEN s.stavalues2 WHEN (s.stakind3 = 1) THEN s.stavalues3 WHEN (s.stakind4 = 1) THEN s.stavalues4 WHEN (s.stakind5 = 1) THEN s.stavalues5 ELSE NULL::anyarray END, NULL
::real[], NULL::anyarray, CASE WHEN (s.stakind1 = 3) THEN s.stanumbers1[1] WHEN (s.stakind2 = 3) THEN s.stanumbers2[1] WHEN (s.stakind3 = 3) THEN s.stanumbers3[1] WHEN
(s.stakind4 = 3) THEN s.stanumbers4[1] WHEN (s.stakind5 = 3) THEN s.stanumbers5[1] ELSE NULL::real END, NULL::anyarray, NULL::real[], NULL::real[]
         Inner Unique: true
         Join Filter: (has_column_privilege(c.oid, a.attnum, 'select'::text) AND (c.oid = a.attrelid))
         ->  Nested Loop  (cost=4.57..21.81 rows=1 width=408) (actual time=0.144..0.160 rows=4 loops=1)
               Output: s.stawidth, s.stakind1, s.stavalues1, s.stakind2, s.stavalues2, s.stakind3, s.stavalues3, s.stakind4, s.stavalues4, s.stakind5, s.stavalues5, s.s
tanumbers1, s.stanumbers2, s.stanumbers3, s.stanumbers4, s.stanumbers5, s.starelid, s.staattnum, c.relname, c.oid
               ->  Index Scan using pg_class_relname_nsp_index on pg_catalog.pg_class c  (cost=0.27..8.29 rows=1 width=72) (actual time=0.056..0.064 rows=1 loops=1)
                     Output: c.oid, c.relname, c.relnamespace, c.reltype, c.reloftype, c.relowner, c.relam, c.relfilenode, c.reltablespace, c.relpages, c.reltuples, c.r
elallvisible, c.reltoastrelid, c.relhasindex, c.relisshared, c.relpersistence, c.relkind, c.relnatts, c.relchecks, c.relhasrules, c.relhastriggers, c.relhassubclass, c.
relrowsecurity, c.relforcerowsecurity, c.relispopulated, c.relreplident, c.relispartition, c.relrewrite, c.relfrozenxid, c.relminmxid, c.relacl, c.reloptions, c.relpart
bound
                     Index Cond: (c.relname = 'lab7'::name)
                     Filter: ((NOT c.relrowsecurity) OR (NOT row_security_active(c.oid)))
               ->  Bitmap Heap Scan on pg_catalog.pg_statistic s  (cost=4.30..13.49 rows=3 width=340) (actual time=0.065..0.068 rows=4 loops=1)
                     Output: s.starelid, s.staattnum, s.stainherit, s.stanullfrac, s.stawidth, s.stadistinct, s.stakind1, s.stakind2, s.stakind3, s.stakind4, s.stakind5
, s.staop1, s.staop2, s.staop3, s.staop4, s.staop5, s.stacoll1, s.stacoll2, s.stacoll3, s.stacoll4, s.stacoll5, s.stanumbers1, s.stanumbers2, s.stanumbers3, s.stanumber
s4, s.stanumbers5, s.stavalues1, s.stavalues2, s.stavalues3, s.stavalues4, s.stavalues5
                     Recheck Cond: (s.starelid = c.oid)
                     Heap Blocks: exact=1
                     ->  Bitmap Index Scan on pg_statistic_relid_att_inh_index  (cost=0.00..4.30 rows=3 width=0) (actual time=0.037..0.038 rows=4 loops=1)
                           Index Cond: (s.starelid = c.oid)
         ->  Index Scan using pg_attribute_relid_attnum_index on pg_catalog.pg_attribute a  (cost=0.28..0.76 rows=1 width=70) (actual time=0.008..0.008 rows=1 loops=4)
               Output: a.attrelid, a.attname, a.atttypid, a.attstattarget, a.attlen, a.attnum, a.attndims, a.attcacheoff, a.atttypmod, a.attbyval, a.attalign, a.attstor
age, a.attcompression, a.attnotnull, a.atthasdef, a.atthasmissing, a.attidentity, a.attgenerated, a.attisdropped, a.attislocal, a.attinhcount, a.attcollation, a.attacl,
 a.attoptions, a.attfdwoptions, a.attmissingval
               Index Cond: ((a.attrelid = s.starelid) AND (a.attnum = s.staattnum))
               Filter: (NOT a.attisdropped)
 Planning Time: 3.213 ms
 Execution Time: 0.737 ms
(24 rows)
```

№	Планируемое время (ms)	Реальное время (ms)
5	0.444	34.788
7	0.307	40.509
10	3.213	0.737