**Григорьев Ю.А.**

**Методические указания к лабораторной работе 2.**

**Оптимизация запросов. Основы** **EXPLAIN в PostgreSQL. Индексация**

[1. Цель лабораторной работы 1](#_Toc115809063)

[2. Подключение к виртуальной машине и к базе данных 1](#_Toc115809064)

[3. Основы EXPLAIN 2](#_Toc115809065)

[4. Эффективные методы индексации в PostgreSQL 4](#_Toc115809066)

[5. Требования к отчёту 6](#_Toc115809067)

[6. Литература 6](#_Toc115809068)

Внимание! Перед выполнением лабораторной работы необходимо полностью прочитать методические указания.

# Цель лабораторной работы

Целью лабораторной работы является формирование следующей компетенции: студент должен получить навыки работы с командой EXPLAIN. Он также должен познакомиться с эффективными методами индексации в PostgreSQL.

# Подключение к виртуальной машине и к базе данных

1. Запустите виртуальную машину. Откройте терминал (пункт Open Terminal контекстного меню).

2. Подключитесь к БД iu6 как пользователь admin:

sudo -u admin psql iu6

password: iu6-magisters

# Основы EXPLAIN

1. Создать тестовую таблицу.

CREATE TABLE foo (c1 integer, c2 text);

2. Включить в таблицу 1 млн записей.

INSERT INTO foo

SELECT i, md5(random()::text)

FROM generate\_series(1, 1000000) AS i;

Примечание. md5 – создание строки длиной 32 символа

3. Прочитать данные с помощью EXPLAIN. Какие данные работы оптимизатора отображаются?

EXPLAIN SELECT \* FROM foo;

4. Добавьте 10 строк (по аналогии с п. 2). Повторите команду EXPLAIN. Что-то изменилось? Почему?

5. Обновить статистику для таблицы foo (команда ANALYZE foo;). Повторите команду EXPLAIN. Что изменилось по сравнению с предыдущим вариантом? Какие действия выполняет команда ANALYZE?

6. Сравните результаты работы оптимизатора с реальным выполнением запроса (используйте EXPLAIN (ANALYZE) - вместе с ANALYZE). Что отображается?

7. Добавьте в запрос условие

EXPLAIN SELECT \* FROM foo WHERE c1 > 500;

8. Создайте индекс для атрибута c1 таблицы foo. Повторите предыдущую команду EXPLAIN. Выполните команду EXPLAIN вместе с ANALYZE. Используется ли индекс? Почему?

9. Измените запрос. Используется ли индекс? Почему?

EXPLAIN SELECT \* FROM foo WHERE c1 < 500;

10. Усложните условие. Используйте ещё текстовое поле. Выполните команду EXPLAIN. Как оптимизатор планирует выполнение запроса?

EXPLAIN SELECT \* FROM foo

WHERE c1 < 500 AND c2 LIKE 'abcd%';

11. Оставьте условие только для текстового поля (SELECT \* FROM foo WHERE c2 LIKE 'abcd%'). Выполните команду EXPLAIN вместе с ANALYZE. Как оптимизатор планирует выполнение запроса? Почему?

12. Создайте индекс для атрибута c2 таблицы foo, используя класс оператора text\_pattern\_ops. Выполните команду EXPLAIN. Как оптимизатор планирует выполнение запроса?

13. Удалите индекс, созданный для c1 таблицы foo. Выполните команду EXPLAIN вместе с ANALYZE для запроса SELECT \* FROM foo ORDER BY c1. Как выполняется запрос?

14. Увеличьте объём используемой оперативной памяти:

SET work\_mem TO '200MB';

15. Выполните команду EXPLAIN вместе с ANALYZE для предыдущего запроса. Как изменилось выполнение запроса?

16. Создайте индекс для c1 таблицы foo. Выполните команду EXPLAIN вместе с ANALYZE для предыдущего запроса (см. 13). Как изменилось выполнение запроса?

17. Создайте новую таблицу, заполните её.

CREATE TABLE bar (c1 integer, c2 boolean);

INSERT INTO bar

SELECT i, i%2=1

FROM generate\_series(1, 500000) AS i;

18. Соберите для таблицы bar статистику с помощью команды ANALYZE.

19. Выполните команду EXPLAIN вместе с ANALYZE для запроса SELECT \* FROM foo JOIN bar ON foo.c1=bar.c1. Поясните порядок выполнения запроса.

20. Как изменится порядок выполнения предыдущего запроса, если создать индекс для атрибута c1 таблицы bar? Как изменилось время выполнения запроса?

21. Удалите индекс, созданный для c1 таблицы foo, выйдите из консоли PostgreSQL в консоль 10.192.6.XX (\q). Остановите PostgreSQL, принудительно зафиксируйте изменения в файловой системе, очистите кэш, запустите PostgreSQL.

sudo service postgresql stop

sudo sync

sudo su root

echo 3 > /proc/sys/vm/drop\_caches

exit

sudo service postgresql start

22. Войдите в консоль PostgreSQL (sudo -u admin psql iu6). Для запроса SELECT \* FROM foo выполните команду EXPLAIN вместе с (ANALYZE,BUFFERS) 3 раза. Как меняется количество блоков, считанных из кэша PostgreSQL.

# Эффективные методы индексации в PostgreSQL

1. Выйдите из консоли PostgreSQL (\q) и подключитесь к базе postgres как пользователь postgres:

sudo -u postgres psql postgres

2. Создайте базу данных:

create table users  
(  
 id serial not null  
 constraint users\_pk  
 primary key,  
 email text not null,  
 type text not null,  
 extra jsonb not null,  
 created\_at timestamp not null,  
 updated\_at timestamp not null  
);  
  
alter table users  
 owner to postgres;  
  
create unique index users\_email\_uindex  
 on users (email);  
  
create unique index users\_id\_uindex  
 on users (id);

3. Заполните базу данных

insert into users (email, created\_at, updated\_at, type, extra)  
select 'user\_' || seq || '@' || (  
 CASE (*RANDOM*() \* 2)::INT  
 WHEN 0 THEN 'yandex'  
 WHEN 1 THEN 'main'  
 WHEN 2 THEN 'custom'  
 END  
 ) || '.ru' AS email,  
 timestamp '2014-01-10 20:00:00' +  
 *random*() \* (timestamp '2016-01-10 20:00:00' -  
 timestamp '2014-01-10 10:00:00'),  
 timestamp '2016-01-10 20:00:00' +  
 *random*() \* (timestamp '2018-01-20 20:00:00' -  
 timestamp '2016-01-10 10:00:00'),  
 (  
 CASE (*RANDOM*() \* 9)::INT  
 WHEN 0 THEN 'admin'  
 ELSE 'user'  
 END  
 ) as type,  
 (  
 CASE (*RANDOM*() \* 9)::INT  
 WHEN 0 THEN '{"extra\_data": true}'::jsonb  
 ELSE '{}'  
 END  
 )  
from *generate\_series*(1, 1000000) seq;

Необходимо выполнить следующие задачи предметной области:

**Основная идея задания - покрыть каждый запрос нужным индексом, чтобы в рамках выполнения запроса не было seq scan (полного прохода по всей таблице)**

Пример запроса с seq scan (медленно):

explain select \* from users where updated\_at='2016-03-22 01:51:57.261738'::timestamp and type='user';

QUERY PLAN

----------------------------------------------------------------------------------------------------------------

Seq Scan on users (cost=0.00..25442.00 rows=1 width=51)

Filter: ((updated\_at = '2016-03-22 01:51:57.261738'::timestamp without time zone) AND (type = 'user'::text))

(2 rows)

Пример запроса с индексом:

explain select \* from users where email='user\_1@yandex.ru' and type = 'admin';

QUERY PLAN

---------------------------------------------------------------------------------

Index Scan using users\_email\_uindex on users (cost=0.42..8.45 rows=1 width=51)

Index Cond: (email = 'user\_1@yandex.ru'::text)

Filter: (type = 'admin'::text)

**Задачи для выполнения:**

1. **Создать индекс для поиска по контексту значения поля email.**

Примечание. Используйте модуль pg\_trgm и оператор индекса gin:

<https://niallburkley.com/blog/index-columns-for-like-in-postgres/>

Выполните explain для следующего оператора select до и после создания индекса.

SELECT COUNT(\*) FROM users WHERE email ILIKE '%yandex.ru';

Сравните результаты.

1. **Создать индекс по update\_at пользователей, но только для админов (type = ‘admin’)**

Примечание. Используйте частичные индексы:

<https://www.postgresql.org/docs/8/indexes-partial.html>

Выполните explain для следующего оператора select до и после создания индекса.

select \* from users where updated\_at = '2016-03-22 01:51:57.261738'::timestamp and type = 'admin';

Сравните результаты.

1. **Создать индекс для поля** **extra.**

Примечание. Используйте эффективное индексирование данных jsonb:

<https://postgrespro.ru/docs/postgrespro/9.5/functions-json>

Выполните explain для следующего оператора select до и после создания индекса.

select count(\*) from users where extra @> '{"extra\_data": true}'::jsonb;

Сравните результаты.

# Требования к отчёту

Отчет должен содержать:

1. Исходные данные, результаты выполнения операторов explain, ответы на вопросы (или пояснения) пунктов 1-22 раздела 3.

2. Решение задач, сформулированных в разделе 4: постановка задачи, оператор создания индекса, результаты выполнения оператора explain до и после создания индекса, сравнение результатов.

# Литература

1. Документация по PostgreSQL 9.6

<https://postgrespro.ru/media/docs/postgresql/9.6/ru/postgres-A4-fop.pdf>