

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет инфокоммуникационных технологий

Дисциплина:

«Проектирование и реализация баз данных»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3
процедуры, функции, триггеры в PostgreSQL

Выполнила:

Арнаутова Елизавета
группа К32422

Проверила:

Говорова Марина Михайловна

Санкт-Петербург
2023

Цель работы: овладеть практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД PostgreSQL, pgadmin 4.

Практическое задание:

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).
2. Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.
3. Изучить графическое представление запросов и просмотреть историю запросов.
4. Создать простой и составной индексы для двух произвольных запросов и сравнить время выполнения запросов без индексов и с индексами. Для получения плана запроса использовать команду EXPLAIN.

СОЗДАНИЕ ХРАНИМЫХ ПРОЦЕДУР И ФУНКЦИЙ

Задание 1: Создать процедуры/функции согласно индивидуальному заданию и (согласно индивидуальному заданию, часть 4).

1. Для повышения оклада сотрудников, выполнивших задания с трехдневным опережением графика на заданный процент.

Сам запрос:

```
create or replace procedure raise_salary(in
percentage integer)
language plpgsql as $$
begin
update employee
set salary = salary * (1 + percentage) / 100
where id in (select distinct schedule.id_employee
from schedule
join task_steps on task_steps.id_schedule =
schedule.id_schedule
where task_steps.step_deadline >=
task_steps.step_check_date + 3
and task.status = 'завершено');
end;
$$;
```

Проведем проверку на вот таком запросе:

```
select distinct employee.name, employee.surname,
schedule.id_employee, employee.salary
from schedule
join employee on employee.id_employee =
schedule.id_employee
join task_steps on task_steps.id_schedule =
schedule.id_employee
where task_steps.step_deadline >=
task_steps.check_date + 3
and task_steps.step_status = 'завершено';
```

СОЗДАНИЕ ХРАНИМЫХ ПРОЦЕДУР И ФУНКЦИЙ





	name character varying (32) 	surname character varying (32) 	id_employee [PK] integer 	salary integer 
1	Баженов	Савелий	11	261276
2	Соколова	Ксения	15	273230
3	Ильин	Денис	31	710214
4	Казаков	Максим	62	152510
5	Ершова	Любовь	67	314862
6	Соловьев	Степан	128	585896
7	Шапошников	Марк	138	145828
8	Александров	Матвей	155	533092
9	Матвеева	Майя	161	642433
10	Нечаева	Аиша	166	636771
11	Константинов	Демид	188	28359
12	Зуев	Андрей	205	760245
13	Климова	Александра	218	58725
14	Савельева	Сабина	242	339408
15	Тимофеев	Андрей	254	743389
16	Лавров	Егор	258	129026
17	Лебедева	Милана	279	453436
18	Попова	Мария	291	365121
19	Ефимов	Николай	346	347966
20	Щукина	Евдокия	354	661455
21	Трофимов	Егор	381	450235
22	Руднева	Наталья	388	792361
23	Григорьева	Виктория	404	82384
24	Высоцкая	Евангелина	422	319345
25	Руднева	Наталья	454	655144
26	Щербакова	Елизавета	457	190971

Рис. 1 – До применения процедуры

СОЗДАНИЕ ХРАНИМЫХ ПРОЦЕДУР И ФУНКЦИЙ

После чего щедро повысим им зарплату в два раза:

```
call raise_salary(100);
```

	name character varying (32)	surname character varying (32)	id_employee [PK] integer	salary integer
1	Баженов	Савелий	11	522552
2	Соколова	Ксения	15	546461
3	Ильин	Денис	31	1420428
4	Казаков	Максим	62	305020
5	Ершова	Любовь	67	629725
6	Соловьев	Степан	128	1171792
7	Шапошников	Марк	138	291656
8	Александров	Матвей	155	1066184
9	Матвеева	Майя	161	1284867
10	Нечаева	Аиша	166	1273543
11	Константинов	Демид	188	56719
12	Зуев	Андрей	205	1520490
13	Климова	Александра	218	117451
14	Савельева	Сабина	242	678816
15	Тимофеев	Андрей	254	1486779
16	Лавров	Егор	258	258053
17	Лебедева	Милана	279	906872
18	Попова	Мария	291	730242
19	Ефимов	Николай	346	695933
20	Щукина	Евдокия	354	1322911
21	Трофимов	Егор	381	900471
22	Руднева	Наталья	388	1584722
23	Григорьева	Виктория	404	164768
24	Высоцкая	Евангелина	422	638690
25	Руднева	Наталья	454	1310288
26	Щербакова	Елизавета	457	381942

Рис. 2 – После применения процедуры

СОЗДАНИЕ ХРАНИМЫХ ПРОЦЕДУР И ФУНКЦИЙ

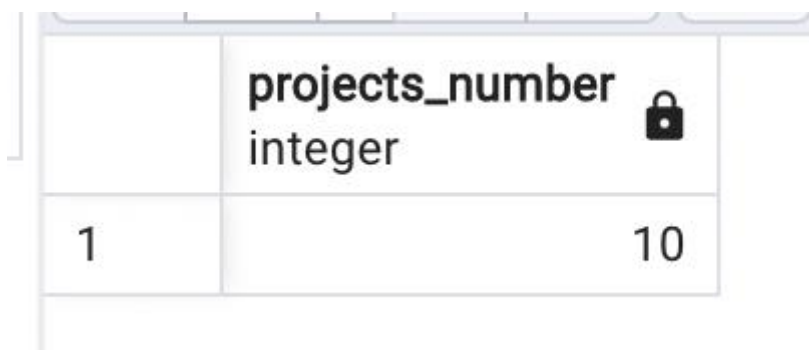
2. Для вычисления количества проектов, в выполнении которых участвует сотрудник.

Сам запрос:

```
create or replace function
projects_number(employee_id integer)
returns integer as $$
declare
project_counter integer;
begin
select count(distinct task.id_project) into
project_counter
from employee
join schedule on employee.id_employee =
schedule.id_employee
join task_steps on task_steps.id_schedule =
schedule.id_schedule
join task on task_steps.id_task = task.id_task
where employee.id_employee = employee_id;
return project_counter;
end;
$$ language plpgsql;
```

Проверим вот таким запросом:

```
select projects_number(3);
```



	projects_number integer
1	10

Рис. 3 – Результат применения функции

СОЗДАНИЕ ХРАНИМЫХ ПРОЦЕДУР И ФУНКЦИЙ

3. Для поиска номера телефона сотрудника (телефон установлен в каждом отделе).

Сам запрос:

```
create or replace function find_phone_number(empl_id
integer)
returns table(department_name varchar,
department_phone varchar) as $$
select department.department_name,
department.department_phone
from employee
join schedule on schedule.id_employee =
employee.id_employee
join department on department.id_department =
schedule.id_department
where employee.id_employee = empl_id;
$$ language sql;
```

Проверим вот таким запросом:

```
select (find_phone_number(1)).department_name,
(find_phone_number(1)).department_phone;
```

И тут наглядно — наш сотрудник числится в двух отделах, и потому для него обнаружено два телефона



	department_name character varying 	department_phone character varying 
1	Юридический отдел	798683973423
2	Администрация	298540309548

Рис. 4 – Результат применения функции

СОЗДАНИЕ ТРИГГЕРОВ

Задание 2. Создать необходимые триггеры.

Для начала — создадим табличку для логирования всего. Будем в нее записывать id операции, название таблицы, схему, какую строку таблицы меняем, что, собственно, делаем (обновляем, добавляем или удаляем) и время.

После чего:

```
create or replace function logger() returns trigger
as $$
    declare
        logging_operation varchar(6);
        row_id integer;
    begin
        if TG_OP = 'INSERT' then
            row_id = NEW.id_client;
            logging_operation := 'insert';
            INSERT INTO log_tab(tab_name, tab_schema,
row_id, operation, operation_time) values
(TG_TABLE_NAME, TG_TABLE_SCHEMA, row_id,
logging_operation, now());
            return new;
        elsif TG_OP = 'UPDATE' then
            row_id = NEW.id_client;
            logging_operation := 'update';
            INSERT INTO log_tab(tab_name, tab_schema,
row_id, operation, operation_time) values
(TG_TABLE_NAME, TG_TABLE_SCHEMA, row_id,
logging_operation, now());
            return new;
        elsif TG_OP = 'DELETE' then
            row_id = OLD.id_client;
            logging_operation := 'delete';
            INSERT INTO log_tab(tab_name, tab_schema,
row_id, operation, operation_time) values
(TG_TABLE_NAME, TG_TABLE_SCHEMA, row_id,
logging_operation, now());
            return old;
        end if;
    end;
$$ language plpgsql;
```


СОЗДАНИЕ ТРИГГЕРОВ

	trigger_catalog name	trigger_schema name	trigger_name name	event_manipulation character varying	event_object_catalog name	event_object_schema name
1	Job Accounting	public	client	INSERT	Job Accounting	public
2	Job Accounting	public	client	DELETE	Job Accounting	public
3	Job Accounting	public	client	UPDATE	Job Accounting	public
4	Job Accounting	public	department	INSERT	Job Accounting	public
5	Job Accounting	public	department	DELETE	Job Accounting	public
6	Job Accounting	public	department	UPDATE	Job Accounting	public
7	Job Accounting	public	employee	INSERT	Job Accounting	public
8	Job Accounting	public	employee	DELETE	Job Accounting	public
9	Job Accounting	public	employee	UPDATE	Job Accounting	public
10	Job Accounting	public	execution_control	INSERT	Job Accounting	public
11	Job Accounting	public	execution_control	DELETE	Job Accounting	public
12	Job Accounting	public	execution_control	UPDATE	Job Accounting	public
13	Job Accounting	public	job_title	INSERT	Job Accounting	public
14	Job Accounting	public	job_title	DELETE	Job Accounting	public
15	Job Accounting	public	job_title	UPDATE	Job Accounting	public
16	Job Accounting	public	project	INSERT	Job Accounting	public
17	Job Accounting	public	project	DELETE	Job Accounting	public
18	Job Accounting	public	project	UPDATE	Job Accounting	public
19	Job Accounting	public	schedule	INSERT	Job Accounting	public
20	Job Accounting	public	schedule	DELETE	Job Accounting	public
21	Job Accounting	public	schedule	UPDATE	Job Accounting	public
22	Job Accounting	public	task	INSERT	Job Accounting	public
23	Job Accounting	public	task	DELETE	Job Accounting	public
24	Job Accounting	public	task	UPDATE	Job Accounting	public
25	Job Accounting	public	task_steps	INSERT	Job Accounting	public
26	Job Accounting	public	task_steps	DELETE	Job Accounting	public
27	Job Accounting	public	task_steps	UPDATE	Job Accounting	public

Рис. 5 – Все имеющиеся в системе триггеры

После чего попробуем что-нибудь добавить в таблицу

СОЗДАНИЕ ТРИГГЕРОВ

```
INSERT INTO task_steps (id_task, id_schedule, step_price, step_start_date, step_deadline, step_status)
VALUES ( '1361', '251', 436107, '2023-1-15', '2024-8-18', 'не начат');

INSERT INTO execution_control (id_schedule, id_supervisor, control_date, comment, task_status)
VALUES ('197', 338, '2077-3-23', 'прозвучать', 'в процессе');

INSERT INTO task (id_project, task_price, task_deadline, task_start_date)
VALUES ('2372', 1506658176, '2023-7-22', '2031-5-23');

INSERT INTO project (id_client, id_manager, id_supervisor, project_status, project_name, project_deadlines, price, payment
VALUES ('381','68','380', 'выполнен', 'экономический правило', '2023-4-24', '253902688', 'полностью оплачен', '2023-4-12')

INSERT INTO schedule (id_employee, id_job_title, id_department, wage_rate)
VALUES ('1', '137', '84', 'полная ставка');

INSERT INTO job_title (id_department, job_title, salary)
VALUES ('351', 'Директор (генеральный директор, управляющий) предприятия', '355737548');

INSERT INTO department (department_name, department_phone)
VALUES ('Отдел продаж', '45064363670');

INSERT INTO employee (id_job_title, id_department, id_schedule, name, surname, patronymic, phone_number, salary)
VALUES ('60', '112', '495', 'Родин', 'Илья', 'Константинович', '792066714777', '461846558' );

INSERT INTO client (client_name, client_address, "Contact_person", "Contact")
VALUES ('должный вес', '6610 Foggy Close', 'Старостин Андрей Леонидович', '+28243684832' );
```

Рис. 5 – Ввод различных данных в различные таблицы

Или удалить

delete from client where client_name = 'должный вес'

Data OutputMessagesNotifications

	log_id [PK] integer	tab_name character varying	tab_schema character varying	row_id bigint	operation character varying (6)	operation_time timestamp without time zone
1	7	task_steps	public	3008	insert	2023-05-25 01:00:05.587784
2	8	execution_control	public	501	insert	2023-05-25 01:00:23.115072
3	9	task	public	3937	insert	2023-05-25 01:00:43.563242
4	10	project	public	2986	insert	2023-05-25 01:00:55.563792
5	11	schedule	public	1017	insert	2023-05-25 01:01:20.657772
6	12	job_title	public	501	insert	2023-05-25 01:01:41.144101
7	13	department	public	501	insert	2023-05-25 01:01:52.532348
8	14	employee	public	521	insert	2023-05-25 01:02:02.629892
9	15	client	public	603	insert	2023-05-25 01:02:15.700435
10	16	client	public	603	delete	2023-05-25 01:25:15.01027

Рис. 5 – Пример работы триггера для команды delete

СОЗДАНИЕ ТРИГГЕРОВ

Или обновить

```
1 update client
2 set client_name = 'полноценное направление'
3 where id_client = 601;
4
5 select * from log_tab
```

Data Output Messages Notifications

	log_id [PK] integer	tab_name character varying	tab_schema character varying	row_id bigint	operation character varying (6)	operation_time timestamp without time zone
1	7	task_steps	public	3008	insert	2023-05-25 01:00:05.587784
2	8	execution_control	public	501	insert	2023-05-25 01:00:23.115072
3	9	task	public	3937	insert	2023-05-25 01:00:43.563242
4	10	project	public	2986	insert	2023-05-25 01:00:55.563792
5	11	schedule	public	1017	insert	2023-05-25 01:01:20.657772
6	12	job_title	public	501	insert	2023-05-25 01:01:41.144101
7	13	department	public	501	insert	2023-05-25 01:01:52.532348
8	14	employee	public	521	insert	2023-05-25 01:02:02.629892
9	15	client	public	603	insert	2023-05-25 01:02:15.700435
10	16	client	public	603	delete	2023-05-25 01:25:15.01027
11	17	client	public	601	update	2023-05-25 01:29:00.664875
12	18	client	public	601	update	2023-05-25 01:29:42.243382
13	19	client	public	601	update	2023-05-25 01:30:01.177284

Рис. 5 – Пример работы триггера для команды update

Выводы:

В процессе выполнения данной лабораторной работы я овладела практическими навыками создания и использования процедур, функций и триггеров в базе данных PostgreSQL.