# Assignment 8

Tasks:

1. Create a block that shows the first\_name and last\_name of all patients. **(5 points)**

Изображение выглядит как текст, компьютер, снимок экрана, монитор

Автоматически созданное описание

do $$  
 declare  
 pat record;  
 begin  
 for pat in  
 select ptn\_f\_name, ptn\_l\_name  
 from patient loop  
 raise notice 'Patient: % %',  
 pat.ptn\_f\_name, pat.ptn\_l\_name;  
 end loop;  
 end; $$;

1. Create a block that shows the average salary of all doctors. **(5 points)**

Изображение выглядит как текст, снимок экрана, монитор, компьютер

Автоматически созданное описание

do $$  
 declare  
 average float;  
 begin  
 select avg(dc\_salary)  
 into average  
 from doctor;  
 raise notice 'The average salary of all doctors: %', average;  
 end; $$;

1. Create a block that finds all doctors whose salary is less than 500$. **(5 points)**

Изображение выглядит как текст, монитор, компьютер, снимок экрана

Автоматически созданное описание

do $$  
 declare  
 few record;  
 begin  
 for few in  
 select dc\_f\_name, dc\_l\_name, dc\_salary  
 from doctor  
 where dc\_salary < 500 loop  
 raise notice 'Doctors whose salaries are less than 500$: % % %', few.dc\_f\_name, few.dc\_l\_name, few.dc\_salary;  
 end loop;  
 end; $$;

1. Create a block that finds all phone number of patients except the number that starts from number 3 (use continue statement with for loop). **(5 points)**

do  
$$  
declare  
 phone\_number record;  
begin  
 for phone\_number in  
 select ptnt\_phone, ptn\_f\_name, ptn\_l\_name  
 from patient  
 loop  
 continue when phone\_number.ptnt\_phone like '3%';  
 raise notice 'Patient numbers that do not start with 3: % % %', phone\_number.ptnt\_phone, phone\_number.ptn\_f\_name, phone\_number.ptn\_l\_name;  
 end loop;  
end;  
$$;

1. Create a block that finds all doctors whose salary is less than the average salary. **(5 points)**

Изображение выглядит как текст, снимок экрана, монитор, электроника

Автоматически созданное описание

do $$  
 declare  
 average record;  
 begin  
 for average in  
 select dc\_f\_name, dc\_l\_name, dc\_salary  
 from doctor  
 where dc\_salary < (select avg(dc\_salary) from doctor) loop  
 raise notice 'Doctors whose salaries are less than the average salary: % % %', average.dc\_f\_name, average.dc\_l\_name, average.dc\_salary;  
 end loop;  
 end; $$;

1. Create a function that finds ten highly paid doctors. **(5 points)**

Изображение выглядит как текст, снимок экрана, монитор, компьютер

Автоматически созданное описание

create function *highly\_paid* ()  
returns varchar  
language plpgsql  
as $$  
 declare  
 doc\_count record;  
 begin  
 for doc\_count in  
 select dc\_f\_name, dc\_l\_name, dc\_salary  
 from doctor  
 order by dc\_salary desc  
 limit 10 loop  
 raise notice '10 highly paid doctors: % % %', doc\_count.dc\_f\_name, doc\_count.dc\_l\_name, doc\_count.dc\_salary;  
 end loop;  
 return doc\_count;  
 end;$$;

1. Create a function that takes one parameter - a pattern of the name of the disease and show who is sick with this disease. **(5 points)**

Изображение выглядит как текст, снимок экрана, монитор, компьютер

Автоматически созданное описание

create function *disease\_patient* (disease varchar(50))  
returns setof disease  
language plpgsql  
as $$  
 declare  
 dp record;  
 begin  
 for dp in  
 select ptn\_f\_name, ptn\_l\_name, disease\_name  
from patient  
inner join appointment a on patient.patient\_id = a.patient\_id  
inner join disease d on d.disease\_id = a.disease\_id  
 where disease\_name = disease  
 loop  
 raise notice '% % sick with %', dp.ptn\_f\_name, dp.ptn\_l\_name, dp.disease\_name ;  
 end loop;  
 return;  
 end;$$;

1. Create a function that contains three parameters: the first and second parameter is the age range of the doctor, and the third is the name of the specialization. Call the function with the appropriate argument values and display the results. **(5 points)**

Изображение выглядит как текст, снимок экрана, компьютер

Автоматически созданное описание

create function *by\_spec* (age\_from int, age\_to int, spec\_name varchar)  
returns setof varchar  
language plpgsql  
as $$  
 declare  
 by\_specialization record;  
 begin  
 for by\_specialization in  
 select dc\_f\_name, dc\_l\_name,  
 date\_part('year',age(current\_date, dc\_bith\_of\_date)) "ages", spzltn\_name  
 from doctor  
 inner join specialization s on doctor.doctor\_id = s.specialization\_id  
 where date\_part('year',age(current\_date, dc\_bith\_of\_date)) between age\_from and age\_to and s.spzltn\_name=spec\_name  
 loop  
 raise notice '% %, %, works for a %', by\_specialization.dc\_f\_name, by\_specialization.dc\_l\_name, by\_specialization.ages, by\_specialization.spzltn\_name ;  
 end loop;  
 return;  
 end;$$;

1. Create a function that contains two parameters, the first parameter is the start point of the drivers’ id, and the second parameter is the endpoint. Loop through and show the ids of drivers with first\_name and last\_name. **(10 points)**

Изображение выглядит как текст, снимок экрана, компьютер

Автоматически созданное описание

create function *drivers*(id\_from int, id\_to int)  
returns setof varchar  
language plpgsql  
as $$  
 declare  
 by\_id record;  
 begin  
 for by\_id in  
 select driver\_id, driver\_f\_name, driver\_l\_name  
 from driver  
 where driver\_id between id\_from and id\_to  
 loop  
 raise notice 'Driver % %, whose id = %', by\_id.driver\_f\_name, by\_id.driver\_l\_name, by\_id.driver\_id;  
 end loop;  
 return;  
 end;$$;

1. Create and call a procedure that takes one argument as the name of a disease and then displays the average age for that disease. **(10 points)**

Изображение выглядит как текст, снимок экрана, компьютер, монитор

Автоматически созданное описание

create procedure *avg\_age\_for\_disease*(  
name varchar  
)  
language plpgsql  
as $$  
 declare aod record;  
 begin  
 for aod in  
 select disease\_name, round(avg(date\_part('year',age(current\_date, ptnt\_birth\_date)))) as ages  
from patient  
inner join appointment a on patient.patient\_id = a.patient\_id  
inner join disease d on d.disease\_id = a.disease\_id  
where disease\_name = name  
group by disease\_name  
 loop  
 raise notice 'Average age for % = %', aod.disease\_name, aod.ages;  
 end loop;  
 end;$$;

1. Create and call a procedure that raises the salary of doctors by 10% whose salary is less than 500. **(10 points)**

Изображение выглядит как текст

Автоматически созданное описание

create procedure *promotion*()  
language plpgsql  
as $$  
 declare increase record;  
 begin  
 for increase in  
 select dc\_f\_name, dc\_l\_name, dc\_salary before, dc\_salary\*1.1 after  
from doctor  
where dc\_salary<500  
 loop  
 raise notice 'Salary of doctor % % before promotion = %, after = %', increase.dc\_f\_name, increase.dc\_l\_name, increase.before, increase.after;  
 end loop;  
 end;$$;

1. Create and call a procedure that finds the top three highly paid doctors for each year of appointment table. **(10 points)**

Изображение выглядит как текст, снимок экрана, монитор, компьютер

Автоматически созданное описание

create procedure *top\_three*()  
language plpgsql  
as $$  
declare doc record;  
begin  
for doc in  
select \* from  
(select extract(year from a.appointment\_date) as year,  
 dc\_f\_name, dc\_l\_name , dc\_salary,  
 dense\_rank() over(partition by extract(year from appointment\_date)  
 order by d.dc\_salary desc) as top  
 from doctor d  
 join appointment a  
 on d.doctor\_id = a.doctor\_id) as doctors\_salary  
 where top<=3  
 loop  
 raise notice '% year % % got %', doc.year, doc.dc\_f\_name, doc.dc\_l\_name, doc.dc\_salary;  
 end loop;  
end;$$;

1. Create and call a procedure that finds most popular specialization name by maximum bill price. **(10 points)**

Изображение выглядит как текст, снимок экрана, монитор, компьютер

Автоматически созданное описание

create procedure *most\_popular\_spec*()  
language plpgsql  
as $$  
 declare by\_bill\_price record;  
 begin  
 for by\_bill\_price in  
 select max(bill\_price) max, spzltn\_name  
from doctor  
 inner join specialization s on doctor.doctor\_id = s.specialization\_id  
inner join appointment a on doctor.doctor\_id = a.doctor\_id  
inner join bill b on a.bill\_id = b.bill\_id  
group by spzltn\_name, bill\_price  
order by bill\_price desc  
limit 1  
 loop  
 raise notice 'Most popular specialization name by maximum bill price: (%,%)', by\_bill\_price.spzltn\_name, by\_bill\_price.max;  
 end loop;  
 end;$$;

1. Create and call a procedure that divides the age of patients into the following classification age range: **(10 points)**

* child : 5-8
* teenage: 9-17
* young: 18-30
* adult: 31-60
* senior: greater than 60

Изображение выглядит как текст, снимок экрана, электроника, компьютер

Автоматически созданное описание

create procedure *age\_range*()  
language plpgsql  
as $$  
 declare age record;  
 begin  
 for age in  
 select  
 date\_part('year',age(current\_date, ptnt\_birth\_date)) ages,  
 Case  
 When date\_part('year',age(current\_date, ptnt\_birth\_date)) between 5 and 8 Then 'child'  
 When date\_part('year',age(current\_date, ptnt\_birth\_date)) between 9 and 17 Then 'teenage'  
 When date\_part('year',age(current\_date, ptnt\_birth\_date)) between 18 and 30 Then 'young'  
 When date\_part('year',age(current\_date, ptnt\_birth\_date)) between 31 and 60 Then 'adult'  
 When date\_part('year',age(current\_date, ptnt\_birth\_date)) > 60 Then 'senior'  
END duration  
from patient  
 loop  
 raise notice '%', age;  
 end loop;  
 end;$$;

1. Drop any 2 functions and procedures.

Изображение выглядит как текст, снимок экрана, монитор, компьютер

Автоматически созданное описание

drop procedure *age\_range*();  
drop procedure *most\_popular\_spec*();  
drop function *by\_spec*();  
drop function *drivers*(id\_from int, id\_to int);