Task1. Write a procedure that will raise doctors salaries. Ask user to enter doctor\_id. If his country is Italy and specialization is Neurologist, then raise salary by 10%. Otherwise rollback. (10 points)

Изображение выглядит как текст, монитор, снимок экрана, электроника

Автоматически созданное описание

create or replace procedure *increase\_salary*(id int)  
language plpgsql  
as $$  
 declare  
 my\_country varchar; my\_spec varchar;  
 needed\_country varchar := 'Italy';  
 needed\_spec varchar := 'Neurologist';  
 begin  
 select dc\_country, spzltn\_name  
 into my\_country, my\_spec  
 from doctor d join specialization s on s.specialization\_id = d.splztn\_id  
 where doctor\_id = id;  
 if my\_country = needed\_country and my\_spec = needed\_spec then  
 update doctor  
 set dc\_salary = dc\_salary\*1.1;  
 commit;  
 else rollback ;  
 end if;  
 end;$$;

Task2. Create a procedure to find a doctor with minimum salary and delete this doctor from Hospital db. Commit if succeeded. Rollback otherwise. (10 points)

Изображение выглядит как текст, монитор, компьютер, снимок экрана

Автоматически созданное описание

create or replace procedure *delete\_doctor*()  
language plpgsql  
as $$  
 declare  
 my\_doctor doctor.doctor\_id%type;  
 begin  
 select doctor\_id  
 into my\_doctor  
 from doctor  
 where dc\_salary = (select min(dc\_salary) from doctor);  
 if not found then  
 rollback;  
 end if;  
 update appointment  
 set doctor\_id = null  
 where doctor\_id = my\_doctor;  
 commit;  
 delete from doctor  
 where doctor\_id = my\_doctor;  
 commit;  
 end;$$;

Task 3. Create a procedure for giving 20% discount for services of doctor with id 150. (10 points)

Изображение выглядит как текст, монитор, внутренний, компьютер

Автоматически созданное описание

create or replace procedure *discount*()  
language plpgsql  
as $$  
 declare  
 disc record;  
 begin  
 for disc in  
 select appointment\_id,bill\_id  
 from appointment  
 where doctor\_id = 150  
 loop  
 update bill  
 set bill\_price = bill\_price\*0.8  
 where bill\_id = (select disc.bill\_id from appointment where doctor\_id = 150  
 and appointment\_id = disc.appointment\_id);  
 end loop;  
 end;$$;

Task4. Create a procedure for changing ambulance id to 1 for those drivers who drive ambulance with id 110. Commit if succeeded. Rollback otherwise. (10 points)

Изображение выглядит как текст, монитор, компьютер, снимок экрана

Автоматически созданное описание

create or replace procedure *change\_id*()  
language plpgsql  
as $$  
 declare  
 change driver.driver\_id%type;  
 begin  
 select driver\_id  
 into change  
 from driver  
 where ambulance\_id = 110;  
 if not found then  
 rollback;  
 end if;  
 update driver  
 set ambulance\_id = 1  
 where driver\_id = (select driver\_id from driver where ambulance\_id = 110);  
 commit;  
 end;$$;

Task 5. Create a procedure to insert a new medicament into Medicaments table. The procedure should take medicament\_name as input parameter and check if the medicament exists in the table. You should add a the medicament if not. Rollback otherwise. (10 points)

Изображение выглядит как текст, монитор, снимок экрана, компьютер

Автоматически созданное описание

create or replace procedure *insert\_medicament*(new\_medicament medicaments.medicament\_name%type)  
language plpgsql  
as $$  
 declare  
 count\_medicament int;  
 counter int;  
 begin  
 select max(medicament\_id)  
 into counter  
 from medicaments;  
 select count(\*)  
 into count\_medicament  
 from medicaments  
 where lower(medicament\_name) = lower(new\_medicament);  
 if count\_medicament < 1 then  
 insert into medicaments(medicament\_id, medicament\_name)  
 values (counter+1,new\_medicament);  
 else rollback ;  
 end if;  
 end; $$;

Task6. Create a procedure for checking hospital status. It should take a parameter for inserting hospital country. If the country is 'China', change hospital status to green. Rollback otherwise. (10 points)

Изображение выглядит как текст, монитор, снимок экрана, компьютер

Автоматически созданное описание

create or replace procedure *check\_status*(id int)  
language plpgsql  
as $$  
 declare  
 insert\_country varchar;  
 needed\_country varchar := 'China';  
 begin  
 select hsptl\_country  
 into insert\_country  
 from hospitals  
 where hospital\_id=id;  
 if insert\_country = needed\_country then  
 update hospitals  
 set hospital\_status = 'green'  
 where hsptl\_country='China' and hospital\_id=id;  
 commit;  
 else rollback ;  
 end if;  
 end;$$;

Task7. Create a procedure for deleting first 50 medicaments from Medicaments table. (10 points)

Изображение выглядит как текст, монитор, снимок экрана, внутренний

Автоматически созданное описание

create or replace procedure *delete\_medicament*()  
language plpgsql  
as $$  
 begin  
 delete from medicaments  
 where medicament\_id=(select medicament\_id from medicaments where medicament\_id>0 limit 50);  
 commit;  
 end;$$;

Task8. Create a procedure for raising salary by 5% for female doctors from Iran. (10 points)

Изображение выглядит как текст, монитор, снимок экрана, компьютер

Автоматически созданное описание

create or replace procedure *raise\_salary*()  
language plpgsql  
as $$  
 declare  
 my\_country varchar; my\_gender varchar;  
 needed\_country varchar := 'Iran';  
 needed\_gender varchar := 'Female';  
 begin  
 select dc\_country, dc\_gender  
 into my\_country, my\_gender  
 from doctor;  
 if my\_country = needed\_country and my\_gender = needed\_gender then  
 update doctor  
 set dc\_salary = dc\_salary\*1.05;  
 commit;  
 end if;  
 end;$$;

Task9. Create a procedure for adding an appointment to Appointment table. It should check whether inserted values are valid or not (valid means they exist in relevant tables : doctor\_id in doctors table, patient\_id in patient table, etc.). If yes - insert. Rollback otherwise. (20 points)

Изображение выглядит как текст, снимок экрана, монитор, компьютер

Автоматически созданное описание

create or replace procedure *add\_new\_appointment*(  
insert\_doc\_id int,  
insert\_pat\_id int,  
insert\_disease\_id int,  
insert\_bill\_id int  
) as $$  
begin  
if exists(select doctor\_id from doctor where doctor\_id=insert\_doc\_id) and  
 exists(select patient\_id from patient where patient\_id=insert\_pat\_id) and  
 exists(select disease\_id from disease where disease\_id=insert\_disease\_id) and  
 exists(select bill\_id from bill where bill\_id=insert\_bill\_id) then  
 insert into appointment(appointment\_id, doctor\_id, patient\_id, disease\_id, appointment\_date, bill\_id) values  
 ((select max(appointment\_id)+1 from appointment), insert\_doc\_id, insert\_pat\_id, insert\_disease\_id, current\_date, insert\_bill\_id);  
 else rollback ;  
 end if;  
end; $$  
language plpgsql;  
  
call *add\_new\_appointment*(1001,1001,1001,1001);