

Королева Алиса, М3437

Структура базы данных «Деканат»:

- Students(StudentId, StudentName, GroupId)
- Groups(GroupId, GroupName)
- Courses(CourseId, CourseName)
- Lecturers(LecturerId, LecturerName)
- Plan(GroupId, CourseId, LecturerId)
- Marks(StudentId, CourseId, Mark)

В таблицах я считаю, что по всем предметам курса студента проставлены баллы, для этого была проделана операция.

```
insert into marks
(select StudentID, CourseID, 0 from Students natural join plan)
on conflict do nothing;
```

То есть в таблицу оценок записаны 0 тем студентам, у которых по курсу, который у них стоит по плану, не стоит оценка.

Я использую PostgreSQL.

1. Напишите запрос, удаляющий всех студентов, не имеющих долгов.

```
delete from Students where StudentID not in
(select StudentID from marks where
    MarkValue <= 60);
```

2. Напишите запрос, удаляющий всех студентов, имеющих 3 и более долгов.

```
delete from Students where StudentID in
(select StudentID from marks where
    MarkValue <= 60
group by (StudentID)
having count(StudentID) >= 3);
```

3. Напишите запрос, удаляющий все группы, в которых нет студентов.

```
delete from groups where GroupID not in
(select GroupID from Students);
```

4. Создайте view Losers в котором для каждого студента, имеющего долги указано их количество.

```
create view Losers as
select StudentID, count(MarkValue) from Marks where
    MarkValue <= 60
group by (StudentID);
```

Королева Алиса, M3437

5. Создайте таблицу LoserT, в которой содержится та же информация, что во view Losers. Эта таблица должна автоматически обновляться при изменении таблицы с баллами.

```
create table LoserT (  
    StudentID int primary key,  
    Count int  
);  
  
create or replace function updateLoserTProcedure() returns trigger as $$  
begin  
    truncate table LoserT;  
    insert into LoserT (select * from Losers);  
    return new;  
end;  
$$ LANGUAGE plpgsql;  
  
create trigger updateLoserT  
    after delete or insert or update on Marks  
    for each row  
    execute procedure updateLoserTProcedure();
```

6. Отключите автоматическое обновление таблицы LoserT.

```
drop trigger if exists updateLoserT on Marks;
```

8. Добавьте проверку того, что все студенты одной группы изучают один и тот же набор курсов.

Моя база данных не поддерживает проверку условий. В связи с чем я напишу триггер, который проверяет, что после вставки записи в таблицу оценок не нарушится условие, что студенты имеют оценки только по тем предметам, которые записаны в плане (что и будет означать, что студенты из одной группы изучают один и тот же набор курсов).

```
create or replace function checkMarks() returns trigger as $$  
begin  
    if exists (select * from Marks  
              where not exists  
                (select * from Plan where  
                  Plan.CourseID = Marks.CourseID and  
                  Plan.GroupID = (select GroupID from Students where  
Students.StudentID = Marks.StudentID)))  
    then  
        raise exception 'Студент не должен иметь оценок по курсам, которые он не  
проходил';  
    end if;  
    return new;  
end;  
$$ LANGUAGE plpgsql;
```

Королева Алиса, М3437

```
create trigger checkMarks
  after delete or insert or update on Marks
  for each row
  execute procedure checkMarks();
```

9. Создайте триггер, не позволяющий уменьшить баллы студента по предмету. При попытке такого изменения, баллы изменяться не должны.

```
create or replace function checkLoweringScore() returns trigger as $$
begin
  if new.MarkValue < old.MarkValue then
    return old;
  end if;
  return new;
end;
$$ LANGUAGE plpgsql;
```

```
create trigger checkLoweringScore
  before update on Marks
  for each row
  execute procedure checkLoweringScore();
```