ДЗ 7

Содержание

1 Формулировка задания

Написать 8 различных запросов на SQL. Будем считать, что у студента долг по предмету, если он изучает этот предмет и имеет по нему менее 60 баллов. В своей таблице я также буду считать, что если в поле оценка стоит `null`, то это значит, что студенту ее не поставили и это долг.

Использованная база данных – `postgres-9.5.4`.

2 Схема базы данных и заполнение

Таблицы:

1. GroupInfo (GroupId, GroupName)
2. CourseInfo (CourseId, CourseName)
3. LecturerInfo (LecturerId, LecturerName)
4. StudentInfo (StudentId, StudentName, GroupId)
5. Marks (StudentId, CourseId, Mark)
6. WorkingPlan (CourseId, GroupId, LecturerId)

```
CREATE TABLE GroupInfo (
  GroupId int NOT NULL,
  GroupName varchar(20) NOT NULL,
  PRIMARY KEY (GroupId));

CREATE TABLE CourseInfo (
  CourseId int NOT NULL,
  CourseName varchar(20) NOT NULL,
  PRIMARY KEY (CourseId));

CREATE TABLE LecturerInfo (
  LecturerId int NOT NULL,
  LecturerName varchar(20) NOT NULL,
  PRIMARY KEY (LecturerId));

CREATE TABLE StudentInfo (
  StudentId int NOT NULL,
  StudentName varchar(20) NOT NULL,
  GroupId int NOT NULL,
  PRIMARY KEY (StudentId),
  FOREIGN KEY (GroupId) REFERENCES GroupInfo(GroupId)
    ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE Marks (
  StudentId int NOT NULL,
  CourseId int NOT NULL,
  Mark int NOT NULL,
  PRIMARY KEY (StudentId,CourseId),
  FOREIGN KEY (StudentId) REFERENCES StudentInfo(StudentId)
```

```
    ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (CourseId) REFERENCES CourseInfo(CourseId)
    ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE WorkingPlan (
  CourseId int NOT NULL,
  GroupId int NOT NULL,
  LecturerId int NOT NULL,
  PRIMARY KEY (CourseId,GroupId,LecturerId),
  FOREIGN KEY (CourseId) REFERENCES CourseInfo(CourseId)
    ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (GroupId) REFERENCES GroupInfo(GroupId)
    ON DELETE CASCADE ON UPDATE CASCADE,
  FOREIGN KEY (LecturerId) REFERENCES LecturerInfo(LecturerId)
    ON DELETE CASCADE ON UPDATE CASCADE);
```

```
INSERT INTO LecturerInfo VALUES (1, 'Аркадий Шагал'),
  (2, 'Георгий Корнеев'), (3, 'Кудряшов Борис'), (4, 'Александр Сегаль');
INSERT INTO GroupInfo VALUES (1, 'M3437'), (2, 'M3438'), (3, 'M3439'), (4, 'M6666');
INSERT INTO StudentInfo VALUES (1, 'Отличный Студент', 3),
  (2, 'Хороший Студент', 3), (3, 'Нормальный Студент', 1),
  (4, 'Странный Студент', 2), (5, 'Плохой Студент', 2);
INSERT INTO CourseInfo VALUES (1, 'Базы данных'), (2, 'Численные методы'), (3, 'Java');
INSERT INTO WorkingPlan VALUES
  (1,2,2), (1,3,2),          -- Databases for m3338-39
  (3,1,2), (3,2,2), (3,3,2), -- Java for everybody
  (2,1,4);                   -- Segal for chosen ones
INSERT INTO Marks VALUES
  (1,1,94), (1,3,87),
  (2,1,75), (2,3,78),
  (3,3,70), (3,1,77),
  (4,1,50), (4,3,97),
  (5,1,40), (5,3,60);
```

3 Запросы

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

   Я хотел написать адекватное решение, которое не пользуется cascading на удалении в foreign ключах, но за час я так и не нашел способа адекватно удалить данные из двух баз одновременно, не повторяя внутренний подзапрос.

   В этот момент я переписал в объявлении базы все foreign key на cascading, и тогда запрос выглядит просто так:

   ```
   delete from studentinfo
   where studentid not in (
     select distinct studentid
     from studentinfo left join marks using (studentid)
     where mark is null or mark <= 60);
   ```

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

   ```
   delete from studentinfo
   where studentid in (
     select studentid
     from studentinfo left join marks using (studentid)
     where mark is null or mark <= 60
   ```

```
  group by studentid
  having count(*) >= 2);
```

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

```
delete from groupinfo
where groupid not in (select groupid from studentinfo where groupid is not null);
```

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

```
create view Losers as
select s.studentid, s.studentname, count(m.mark)
from studentinfo as s left join marks as m using (studentid)
where mark is null or mark <= 60
group by studentid;
```

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 as select * from Losers;

create function after_marks_update() returns trigger as $bump_losert$
begin drop table LoserT; -- insert into, on conflict update?
      create table LoserT as select * from Losers;
      return null;
end;

$bump_losert$ LANGUAGE plpgsql;

create trigger bump_losers
after insert or update or delete on Marks
for each statement execute procedure after_marks_update();
```

6. Отключите автоматическое обновление таблицы LoserT.

```
drop trigger bump_losert on Marks;
```

7. Node has run out of gas, script execution stopped.

Автор: Михаил Волхов M3438
Created: 2016-11-22 Tue 13:22
Validate