**Задание 1.** Примените пожалуйста [скрипт](https://drive.google.com/file/d/15CZE97IH7N9mgAvNSseu_61vxMf-wGqD/view?usp=sharing) для базы данных PostgreSQL. Если есть необходимость, то вы можете применить его заново, предварительно удалив таблицы командой

DROP TABLE *имя\_таблицы;*

Применить скрипт - это значит выполнить его через ваш используемый IDE (DataGrip, DBeaver , pgAdmin)

**Результат работы:**

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Автоматически созданное описание

**Задание 2.**

1. Напишите SQL запрос который создаст таблицу с именем “*student\_on\_course\_kn*” ,содержащую информацию по успеваемости студентов из Казани и Новосибирска (как часть данных таблицы “*student\_on\_course*” со всеми атрибутами). Используйте паттерн “CREATE … AS SELECT …”.

Пожалуйста приложите команду создания таблицы и результат запроса

SELECT \* FROM student\_on\_course\_kn ORDER BY id;

в виде скриншота к заданию.

**Код к запросу:**

CREATE TABLE student\_on\_course\_kn AS

SELECT soc.\*

FROM student\_on\_course soc

JOIN student s ON soc.student\_id = s.id

WHERE s.city IN ('Казань', 'Новосибирск');

**Результат работы:**

Изображение выглядит как текст, линия, Шрифт, снимок экрана

Автоматически созданное описание

Рис. 1. Результат выполнения запроса на создание таблицы

Изображение выглядит как текст, число, Шрифт, кроссворд

Автоматически созданное описание

Рис. 2. Таблица student\_course\_kn

1. Обновите баллы всех студентов в новой созданной таблице “*student\_on\_course\_kn*” **одной** UPDATE командой с учетом следующих правил

* уменьшаем на 5 баллов *student\_rating*, если значение *student\_rating* лежит между 0 и 60 баллами включительно
* уменьшаем на 10 баллов *student\_rating*, если значение *student\_rating* лежит между 61 и 80 баллами включительно
* уменьшаем на 15 баллов *student\_rating*, если значение *student\_rating* лежит между 81 и 90 баллами включительно
* во всех остальных случаях вычитаем 20 баллов из значения *student\_rating*

Пожалуйста приложите команду обновления таблицы и результат

запроса

SELECT \* FROM student\_on\_course\_kn ORDER BY id;

в виде скриншота к заданию.

**Код запроса:**

--SELECT \* FROM student\_on\_course\_kn

UPDATE student\_on\_course\_kn

SET student\_rating = CASE

WHEN student\_rating >= 0 and student\_rating <= 60 THEN student\_rating -5

WHEN student\_rating >= 61 and student\_rating <= 80 THEN student\_rating -10

WHEN student\_rating >= 81 and student\_rating <= 90 THEN student\_rating -15

ELSE student\_rating -20

END;

**Результат работы:**

Изображение выглядит как текст, линия, Шрифт, снимок экрана

Автоматически созданное описание

Рис. 1. Результат выполнения запроса на обновление

Изображение выглядит как текст, число, Шрифт, кроссворд

Автоматически созданное описание

Рис. 2. Измененная таблица student\_course\_kn

1. Удалите запись о студентах из таблицы “*student\_on\_course\_kn*”, родной город которых Новосибирск, посещающие курс “Цифровая трансформация” Пожалуйста приложите команду удаления строк таблицы и результат запроса

SELECT \* FROM student\_on\_course\_kn ORDER BY id;

в виде скриншота к заданию.

**Код запроса:**

--SELECT \* FROM student\_on\_course\_kn ORDER BY id;

DELETE FROM student\_on\_course\_kn

WHERE student\_id IN (

SELECT student.id

FROM student

JOIN student\_on\_course\_kn ON student.id = student\_on\_course\_kn.student\_id

JOIN course ON student\_on\_course\_kn.course\_id = course.id

WHERE student.city = 'Новосибирск'

AND course.name = 'Цифровая трансформация'

);

**Результат выполнения:**

Изображение выглядит как текст, линия, Шрифт, снимок экрана

Автоматически созданное описание

Рис. 1. Результат выполнения запроса на удаление

Изображение выглядит как текст, число, Шрифт, снимок экрана

Автоматически созданное описание

Рис. 2. Измененная таблица student\_course\_kn

1. Напишите SQL запрос который к существующим записям таблицы “*student\_on\_course\_kn*” добавит всех студентов из города Москва, **НО** с учетом вычитания баллов из задания b). Те баллы студентов должны соответствовать формуле. Пожалуйста приложите команду добавления строк и результат запроса

SELECT \* FROM student\_on\_course\_kn ORDER BY id;

в виде скриншота к заданию.

**Код запроса:**

INSERT TABLE student\_on\_course\_kn AS

SELECT soc.\*

FROM student\_on\_course soc

JOIN student s ON soc.student\_id = s.id

WHERE s.city = ‘Москва’;

**Результат работы:**

Рис. 1. Результат выполнения запроса на добавление.

Изображение выглядит как текст, число, Шрифт, кроссворд

Автоматически созданное описание

Рис. 2. Обновленная таблица student\_on\_course\_kn

1. Выведите имена студентов, рейтинг по курсу, учебное заведение студента, соответствующее имя курса, учебное заведение, которое проводит курс одним SQL запросом, используя новую таблицу “*student\_on\_course\_kn*”. Результат отсортируйте по имени студента и названию курса. Обратите внимание на имена столбцов ниже, которые должны быть указаны как атрибуты вывода вашего SQL запроса.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **student\_name** | **student\_rating** | **student\_college\_name** | **course\_name** | **course\_college\_name** |
| Анна Потапова | 66 | МФТИ | Нейронные сети | МФТИ |
| Екатерина Андреева | 75 | МГУ | Актерское мастерство | Сколково |
| … | … | … | … | … |

**Код запроса:**

SELECT

student.name AS student\_name,

student\_on\_course\_kn.student\_rating AS student\_rating,

college.name AS student\_college\_name,

course.name AS course\_name,

college.name AS student\_college\_name

FROM

student\_on\_course\_kn

JOIN student ON student.id = student\_on\_course\_kn.student\_id

JOIN college ON college.id = student.college\_id

JOIN course ON course.id = student\_on\_course\_kn.course\_id

ORDER BY

student\_name, course\_name;

**Результат запроса:**

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание