Лабораторная работа по теме “Язык запросов SQL”

1. Необходимо установить базу данных PostgreSQL (***под администратором***

***системы***),

для Windows (инсталлятор и документация) : https://postgrespro.ru/windows

для Linux (инсталлятор и документация) : https://www.postgresql.org/download/

1. Скачать **одно** из приложений

- pgAdmin - https://www.pgadmin.org/

- dbeaver - https://dbeaver.io/

- DataGrip - <https://www.jetbrains.com/ru-ru/datagrip/>

1. Подключиться приложением из пункта №2 к базе данных из пункта №1



Необходимо указать

- **Host** - IP адрес компьютера (сервера) где находится установленная база

данных из пункта №1

- **User** - пользователь (оставить postgres)

- **Port** - порт (оставить 5432 если вы не меняли этот порт в настройках)

- **Password** - пароль (который вы указывали при инсталляции)

- **Database** - база данных (оставить postgres)

Примените пожалуйста скрипт 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)

Необходимо написать SQL запросы к следующим задачам ниже. SQL запрос

пишется в вашем IDE и прикладывается (как текст) вместе со скрином ответа в doc

файле - который в свою очередь необходимо приложить к домашнему заданию и

отправить на платформу. Задание считается выполненным - если SQL запрос написан

синтаксически корректно и возвращает ожидаемые данные на условие задачи.

1. Напишите SQL запрос который возвращает имена студентов и их аккаунт

в Telegram у которых родной город “Казань” или “Москва”. Результат

отсортируйте по имени студента в убывающем порядке

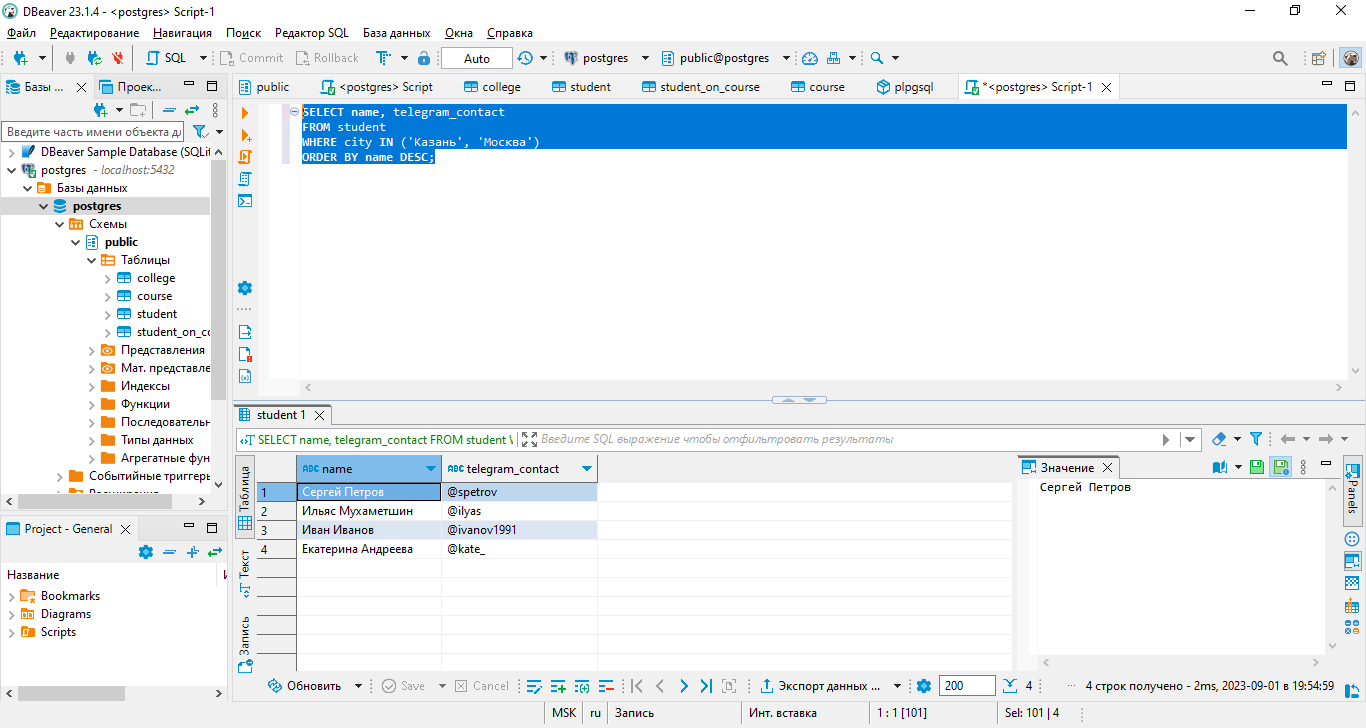
Ответ:

**SELECT** **name**, telegram\_contact

**FROM** student

**WHERE** city **IN** ('Казань', 'Москва')

**ORDER** **BY** **name** **DESC**;



1. Напишите SQL запрос который возвращает данные по университетам в

следующем виде (один столбец со всеми данными внутри) с сортировкой

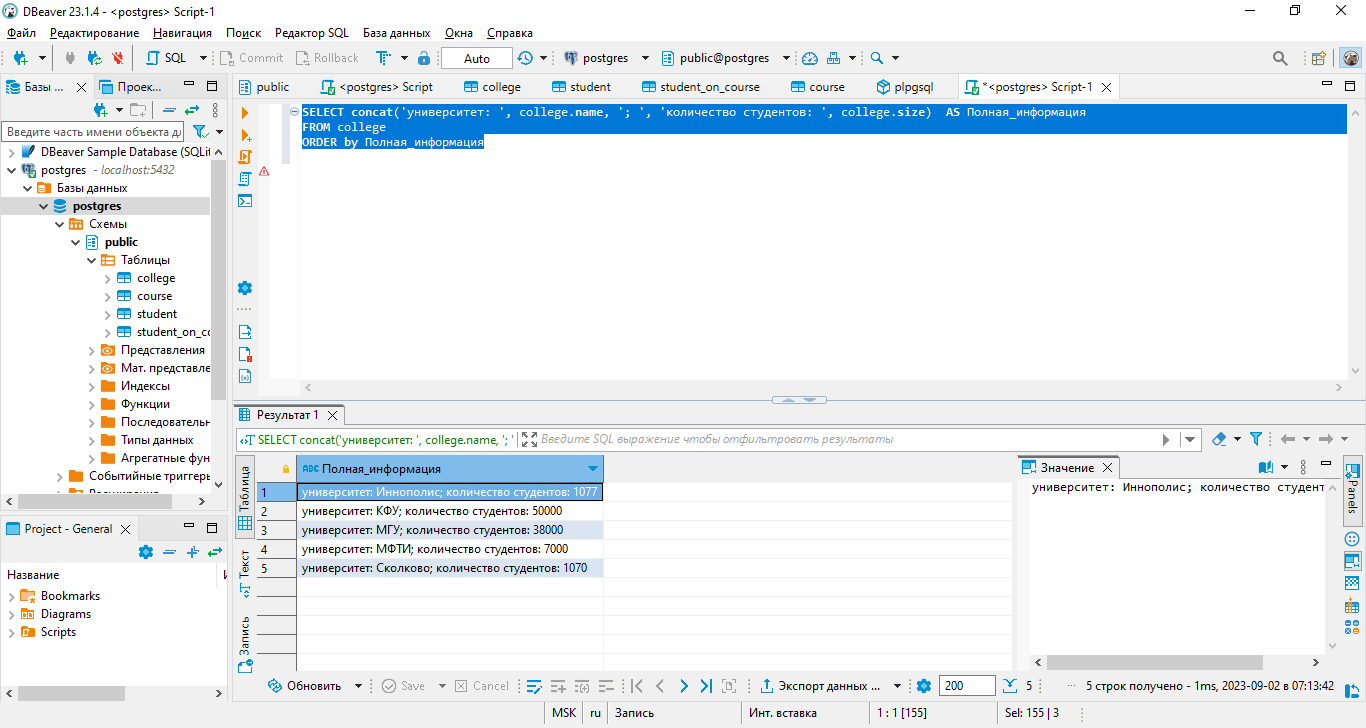
по полю *“полная информация”*

Ответ:

**SELECT** **concat**('университет: ', college.**name**, '; ', 'количество студентов: ', college.**size**) **AS** Полная\_информация

**FROM** college

**ORDER** **by** Полная\_информация



1. Напишите SQL запрос который возвращает список университетов и

количество студентов, если идентификатор университета должен быть

выбран из списка 10, 30, 50. Пожалуйста примените конструкцию IN.

Результат запроса отсортируйте по количеству студентов **И** затем по

наименованию университета.

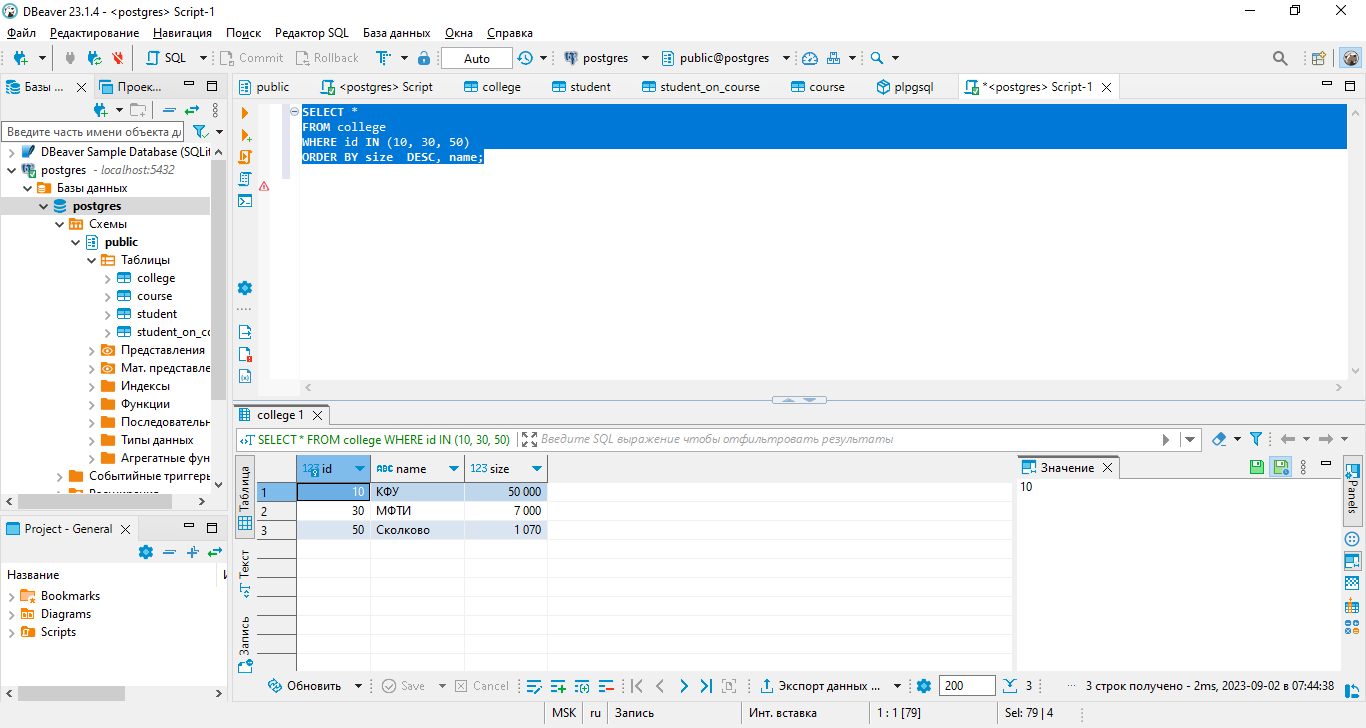
Ответ:

**SELECT** \*

**FROM** college

**WHERE** id **IN** (10, 30, 50)

**ORDER** **BY** **size** **DESC**, **name**;



1. Напишите SQL запрос который возвращает список университетов и

количество студентов, если идентификатор университета **НЕ** должен

соответствовать значениям из списка 10, 30, 50. Пожалуйста в основе

примените конструкцию IN. Результат запроса отсортируйте по

количеству студентов **И** затем по наименованию университета.

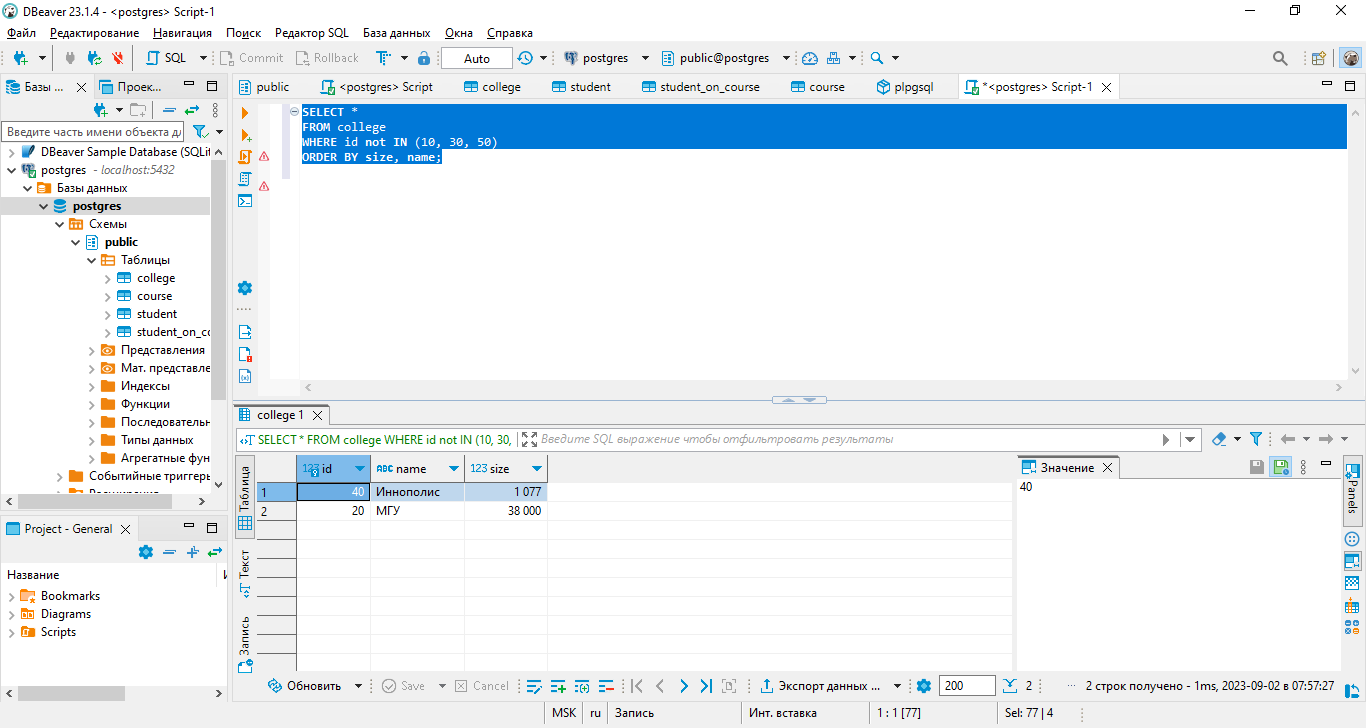
Ответ:

**SELECT** \*

**FROM** college

**WHERE** id **not** **IN** (10, 30, 50)

**ORDER** **BY** **size**, **name**;



1. Напишите SQL запрос который возвращает название online курсов

университетов и количество заявленных слушателей. Количество

заявленных слушателей на курсе должно быть в диапазоне от 27 до 310

студентов. Результат отсортируйте по названию курса и по количеству

заявленных слушателей в убывающем порядке для двух полей.

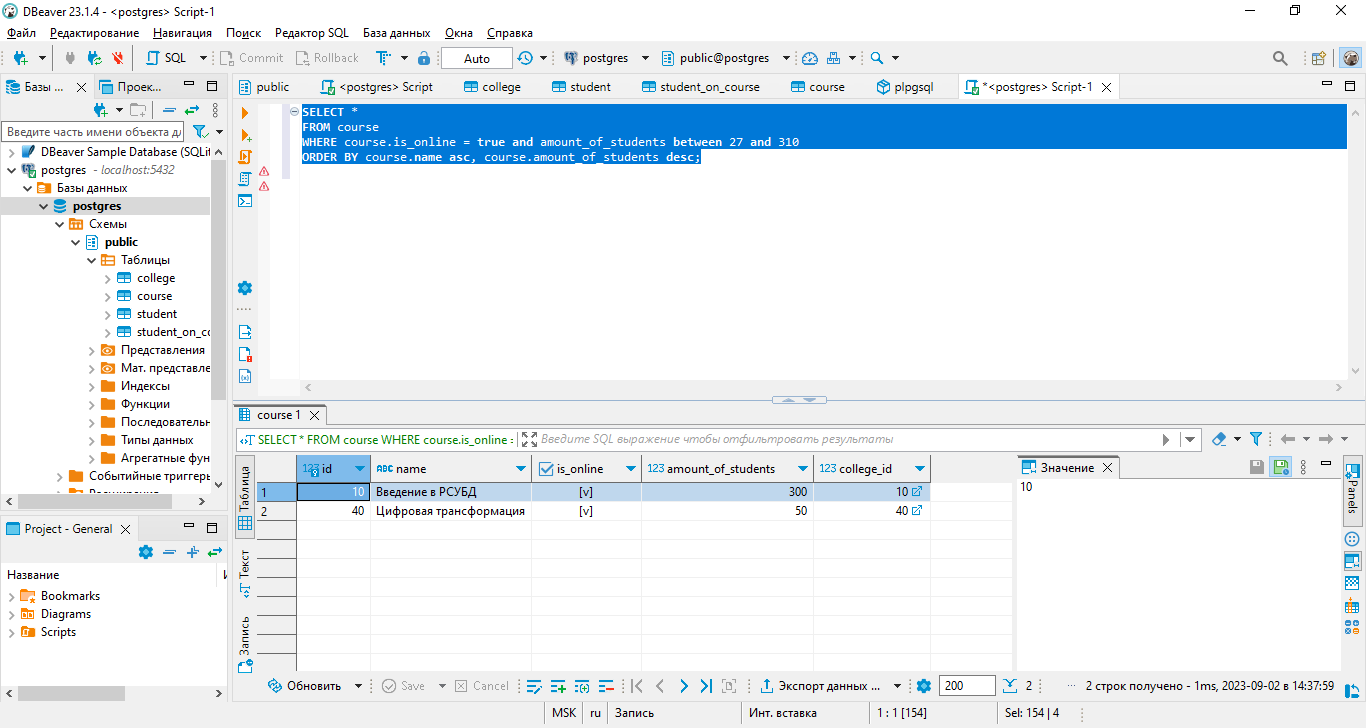
Ответ:

**SELECT** \*

**FROM** course

**WHERE** course.is\_online = **true** **and** amount\_of\_students **between** 27 **and** 310

**ORDER** **BY** course.**name** **asc**, course.amount\_of\_students **desc**;



1. Напишите SQL запрос который возвращает имена студентов и название

курсов университетов в одном списке. Результат отсортируйте в

убывающем порядке. Пример части результата представлен ниже

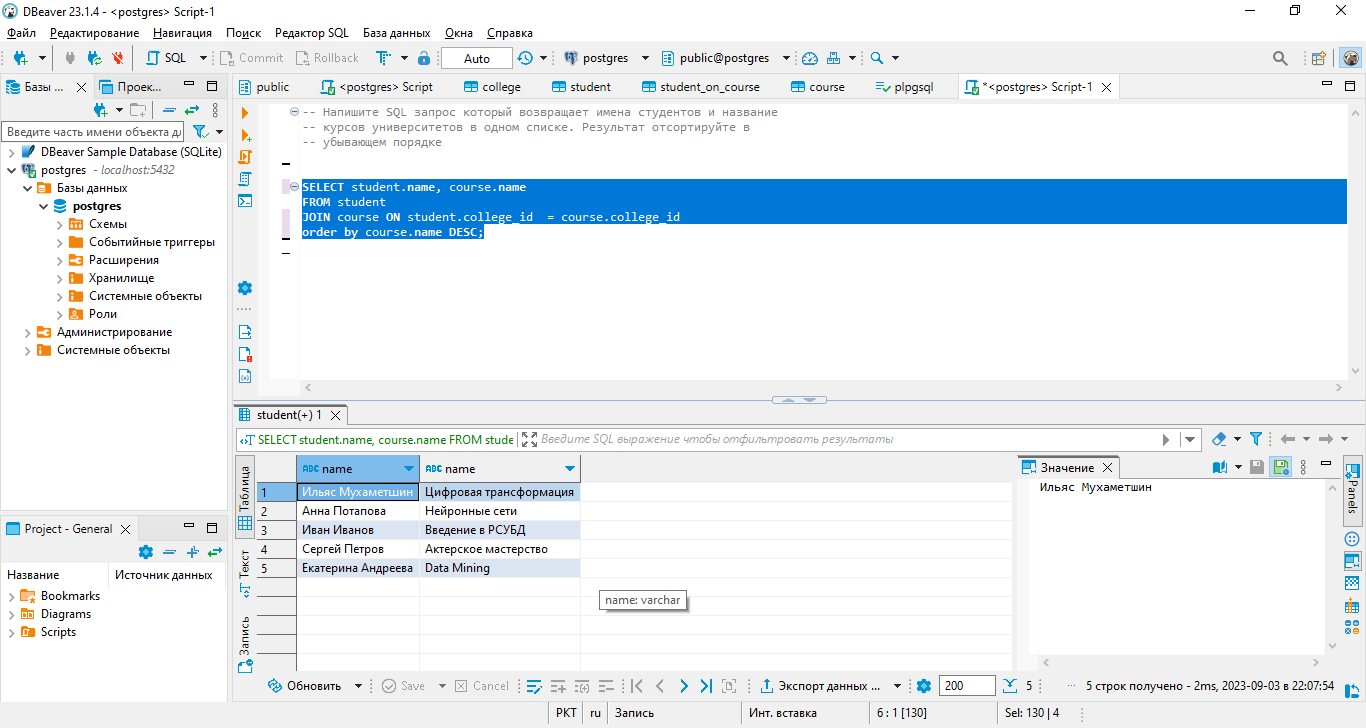
Ответ:

**SELECT** student.**name**, course.**name**

**FROM** student

**JOIN** course **ON** student.college\_id = course.college\_id

**ORDER** BY course.**name** **DESC**;



1. Напишите SQL запрос который возвращает имена университетов и

название курсов в одном списке, но с типом что запись является или

“университет” или “курс”. Результат отсортируйте в убывающем порядке

по типу записи и потом по имени. Пример части результата представлен

ниже

Ответ:

**SELECT** c.**name** , 'Курс' **as** object\_type

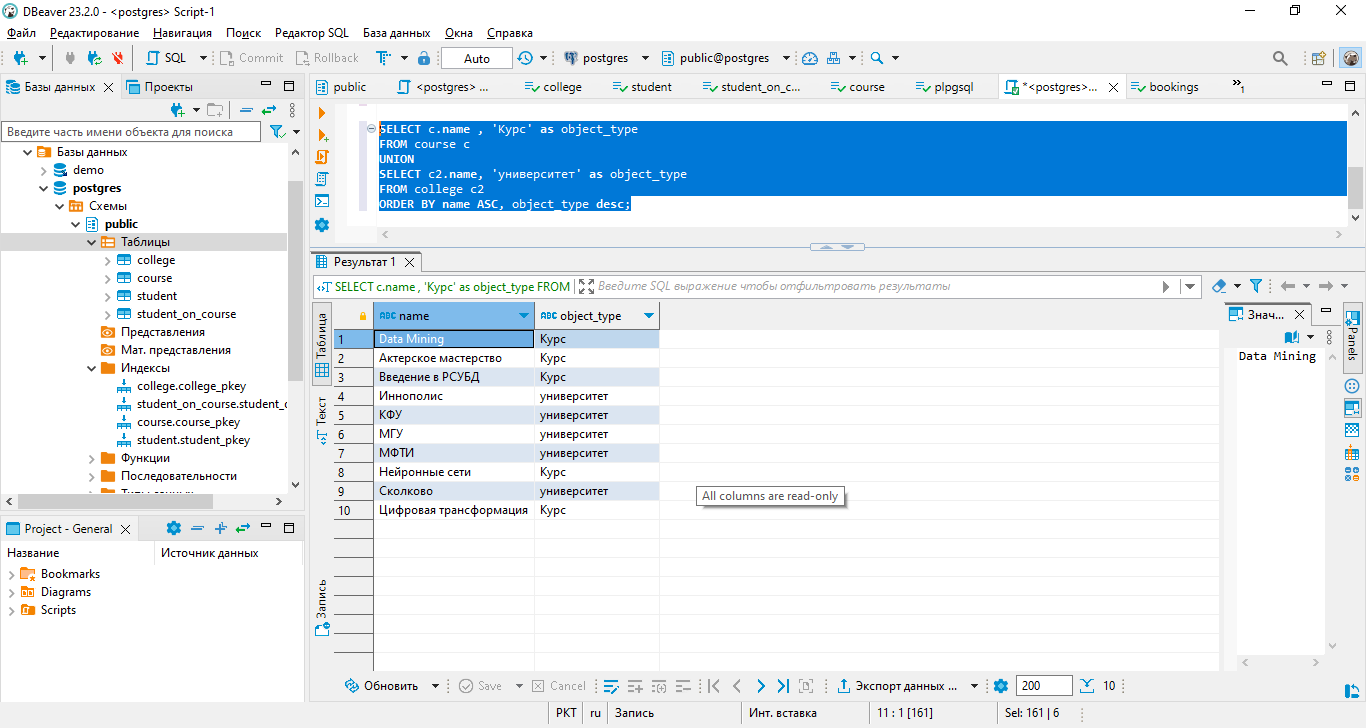
**FROM** course c

**UNION**

**SELECT** c2.**name**, 'университет' **as** object\_type

**FROM** college c2

**ORDER** **BY** **name** **ASC**, object\_type **desc**;



1. Напишите SQL запрос который возвращает название курса и количество

заявленных студентов в отсортированном списке по количеству

слушателей в возрастающем порядке, **НО** запись с количеством

слушателей равным 300 должна быть на первом месте. Ограничьте

вывод данных до 3 строк. Пример результата представлен ниже

**Подсказка**: используйте в ORDER BY синтаксический

элемент CASE … END. Саму конструкцию можно посмотреть вот здесь

Ответ:

**SELECT** **name**, amount\_of\_students

**from** course c

**order** **by**

**case**

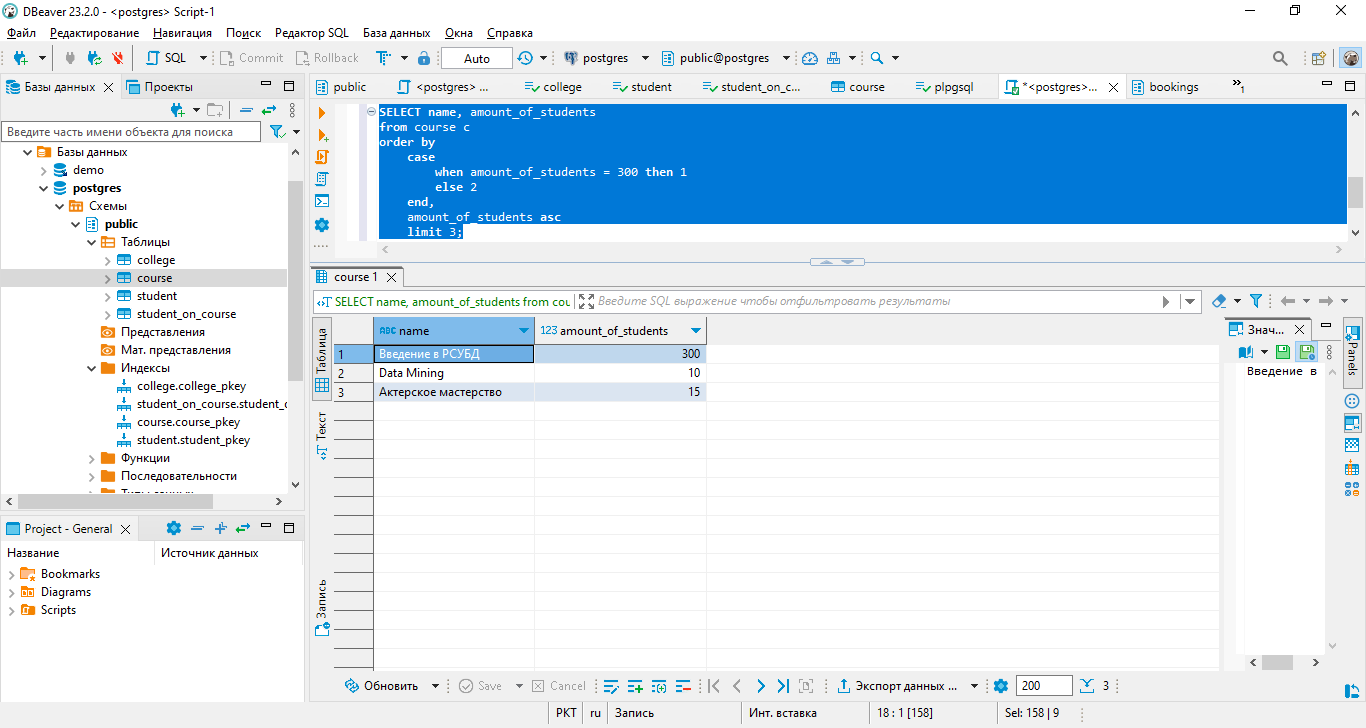
**when** amount\_of\_students = 300 **then** 1

**else** 2

**end**,

amount\_of\_students **asc**

**limit** 3;



1. Напишите DML запрос который создает новый **offline** курс со

следующими характеристиками:

- id = 60

- название курса = Machine Learning

- количество студентов = 17

- курс проводится в том же университете что и курс Data Mining

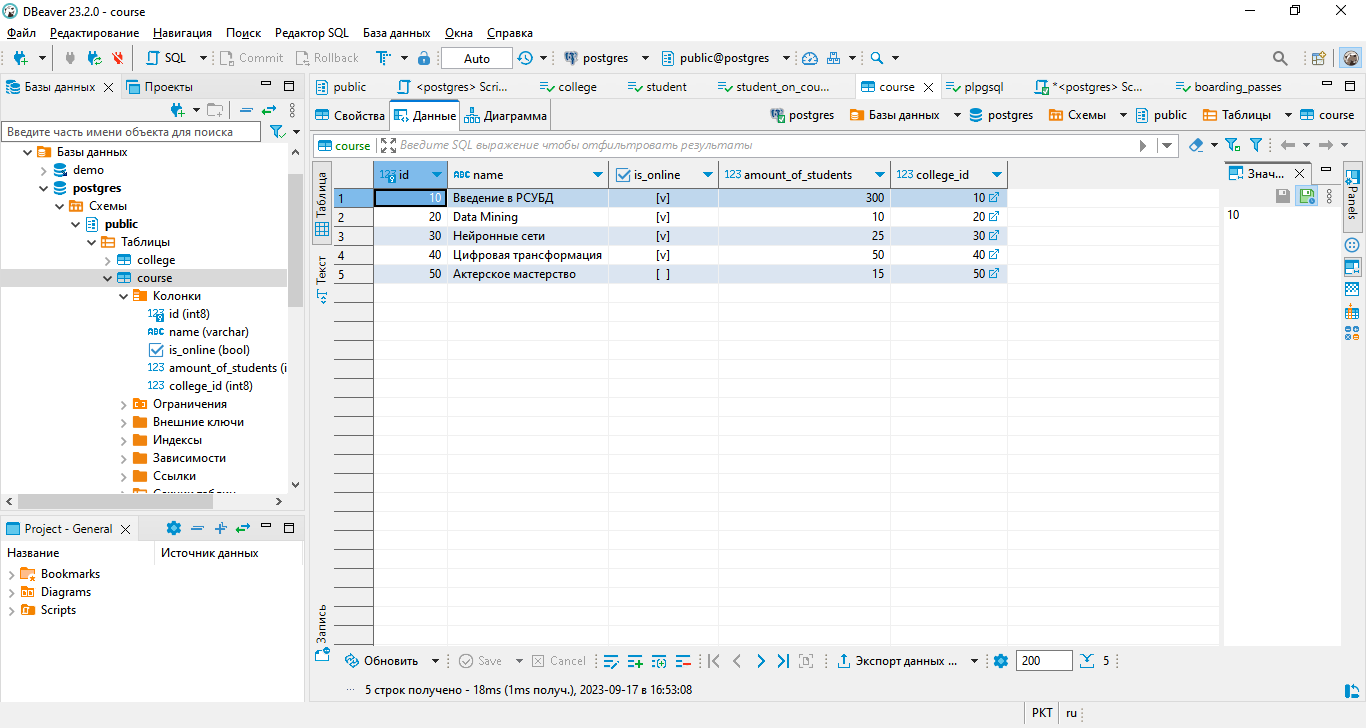
Предоставьте INSERT выражение которое заполняет необходимую

таблицу данными

Приложите скрин результата запроса к данным курсов после

выполнения команды INSERT к таблице которая была изменена.

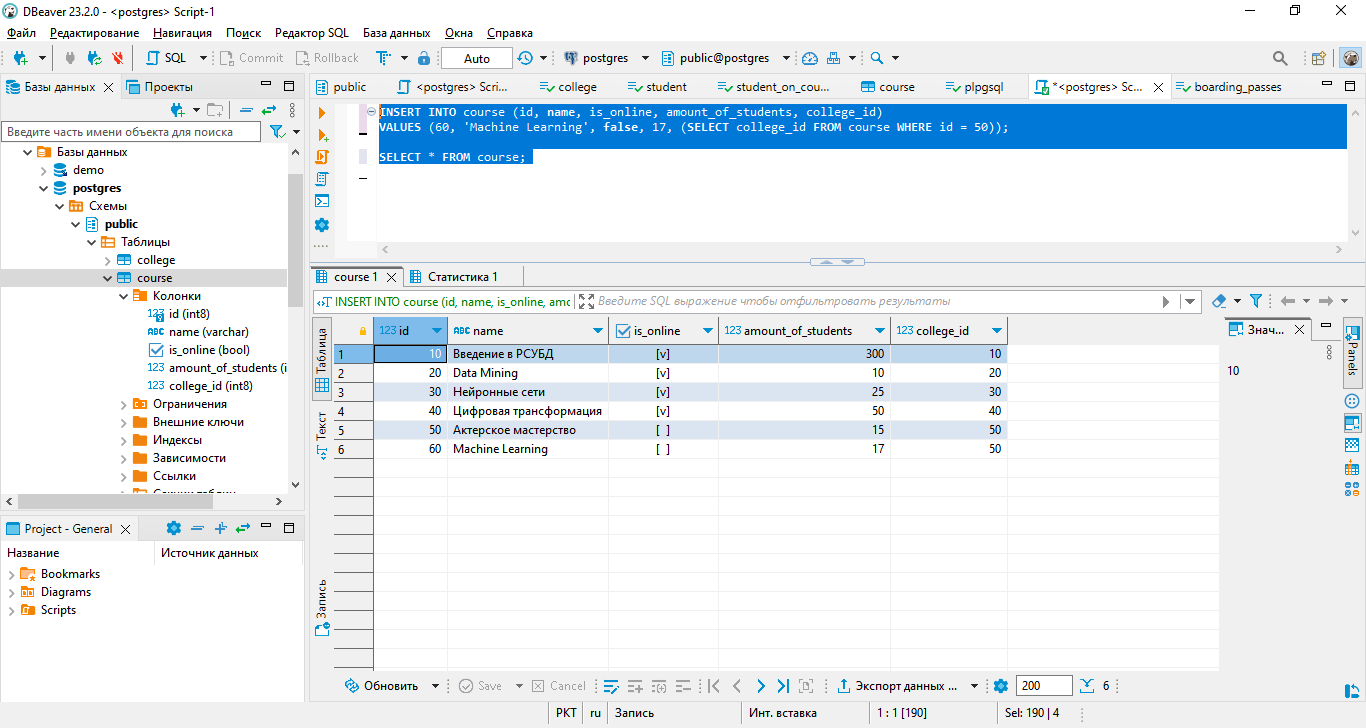
Ответ:



**INSERT** **INTO** course (id, **name**, is\_online, amount\_of\_students, college\_id)

**VALUES** (60, 'Machine Learning', **false**, 17, (**SELECT** college\_id **FROM** course **WHERE** id = 50));

**SELECT** \* **FROM** course;



1. Напишите SQL скрипт который подсчитывает симметрическую разницу

множеств A и B.

(A \ B) ⋃ (B \ A)

где A - таблица **course,** B - таблица **student\_on\_course,** “\” - это разница

множеств, “⋃” - объединение множеств. Необходимо подсчитать на

основании атрибута **id** из обеих таблиц. Результат отсортируйте по 1

столбцу. Пример результата представлен ниже.

Ответ:

SELECT id FROM course

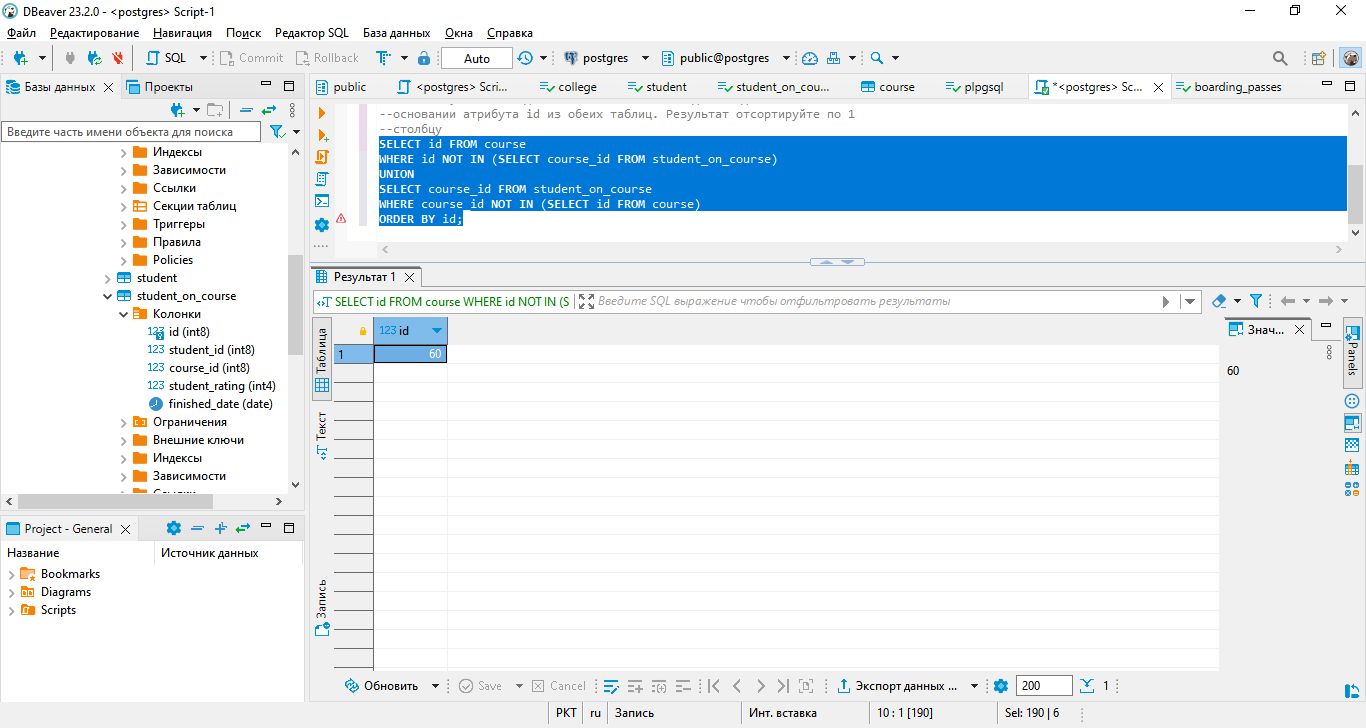
WHERE id NOT IN (SELECT course\_id FROM student\_on\_course)

UNION

SELECT course\_id FROM student\_on\_course

WHERE course\_id NOT IN (SELECT id FROM course)

ORDER BY id;



1. Напишите SQL запрос который вернет имена студентов, курс на котором

они учатся, названия их родных университетов (в которых они

официально учатся) и соответствующий рейтинг по курсу. С условием

что рассматриваемый рейтинг студента должен быть строго больше (>)

50 баллов и размер соответствующего ВУЗа должен быть строго больше

(>) 5000 студентов. Результат необходимо отсортировать по первым двум

столбцам. Обратите внимание на часть ответа ниже с учетом

**именования выходных атрибутов** вашего запроса

Ответ:

Select

Join (with)

Where фильтры

Сортировка ASC по нескольким полкам

1. Выведите уникальные семантические пары студентов, родной город

которых один и тот же. Результат необходимо отсортировать по первому

столбцу. Семантически эквивалентная пара является пара студентов

например (Иванов, Петров) = (Петров, Иванов), в этом случае должна

быть выведена одна из пар. Обратите внимание на ответ ниже с учетом

**именования выходных атрибутов** вашего запроса

Ответ:

Select

Join (with) union

Where фильтры

Сортировка ASC по 1 столбцу

1. Напишите SQL запрос который возвращает количество студентов,

сгруппированных по их **оценке**. Результат отсортируйте по названию

оценки студента. Формула выставления оценки представлена ниже как

псевдокод.

ЕСЛИ оценка < 30 ТОГДА неудовлетворительно

ЕСЛИ оценка >= 30 И оценка < 60 ТОГДА удовлетворительно

ЕСЛИ оценка >= 60 И оценка < 85 ТОГДА хорошо

В ОСТАЛЬНЫХ СЛУЧАЯХ отлично

Пример результата ниже. Обратите внимание на именование

результирующих столбцов в вашем решении. Курс “Machine Learning”, так

как у него нет студентов - проигнорируйте, используя соответствующий

тип JOIN.

Ответ:

Select

Case, then, when, and|or

Join

группировка

Сортировка ASC по 1 столбцу

1. Дополните SQL запрос из задания a), с указанием вывода имени курса и

количество оценок внутри курса. Результат отсортируйте по названию

курса и оценки студента. Пример части результата ниже.

Обратите внимание на именование результирующих столбцов в вашем

решении. Курс “Machine Learning”, так как у него нет студентов -

проигнорируйте, используя соответствующий тип JOIN.

Ответ:

Select

Case, then, when, and|or

Агрегатная функция

Join

группировка

Сортировка ASC по 2 столбцам