**Домашнее задание № 9**

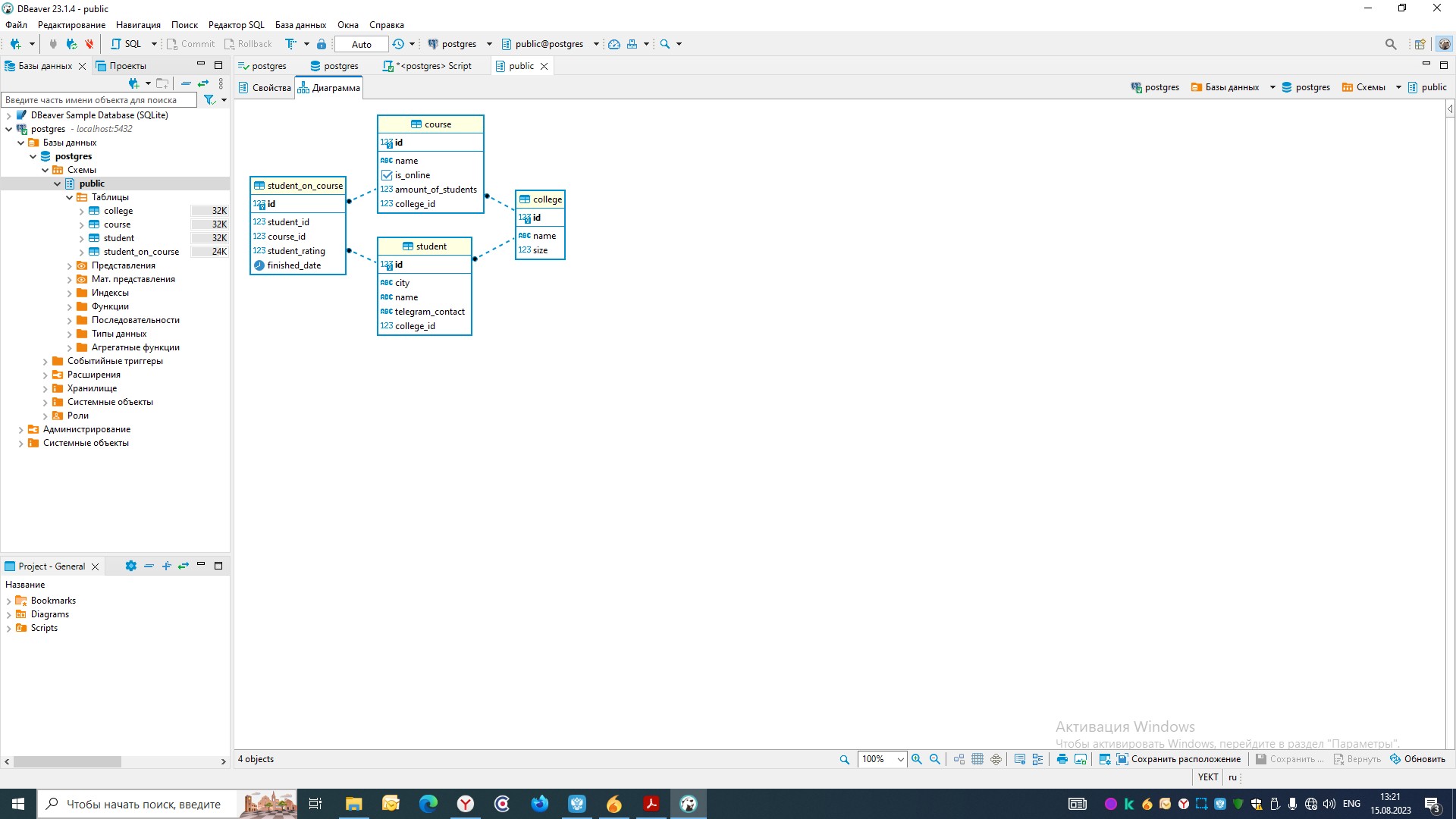
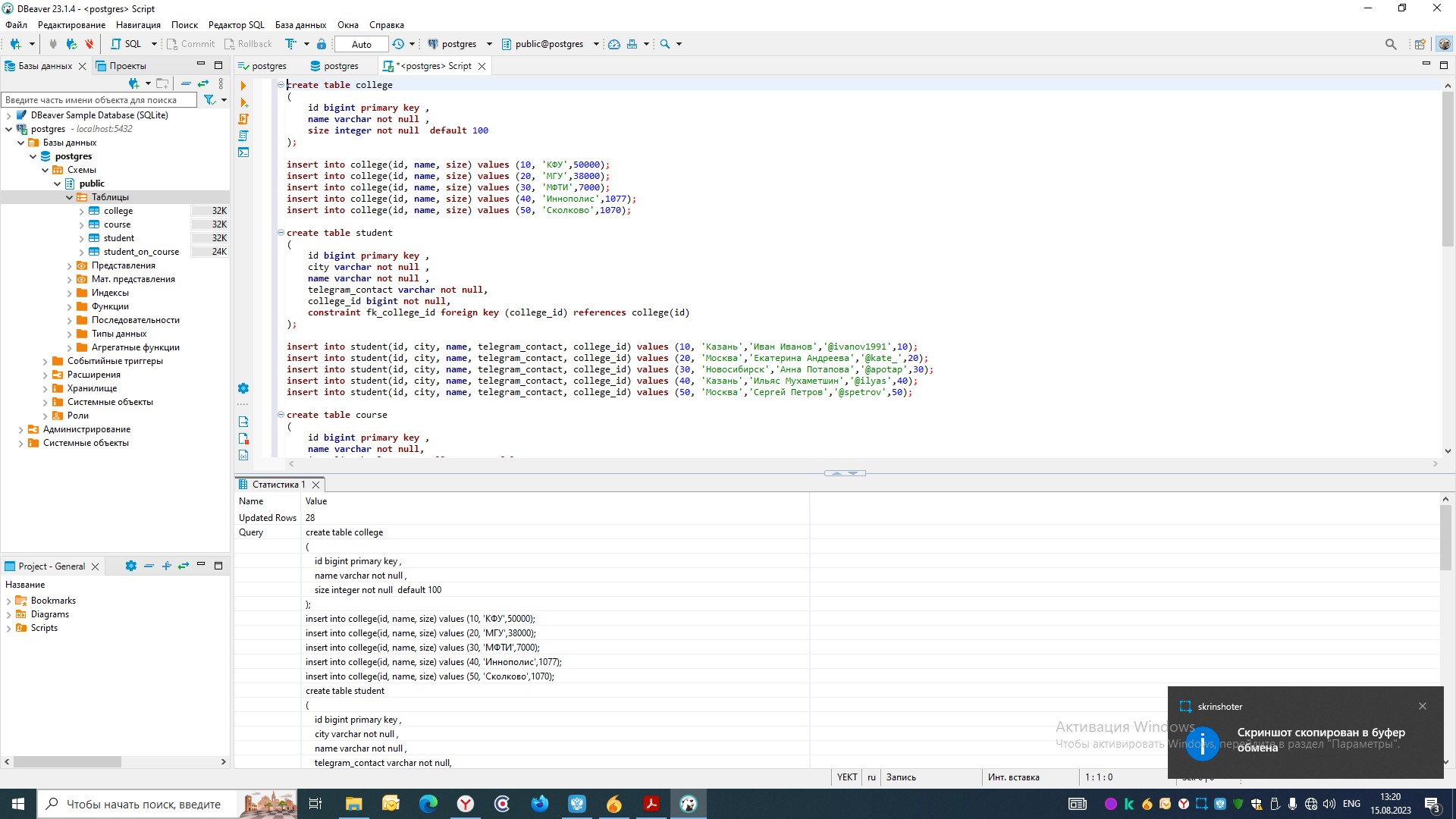
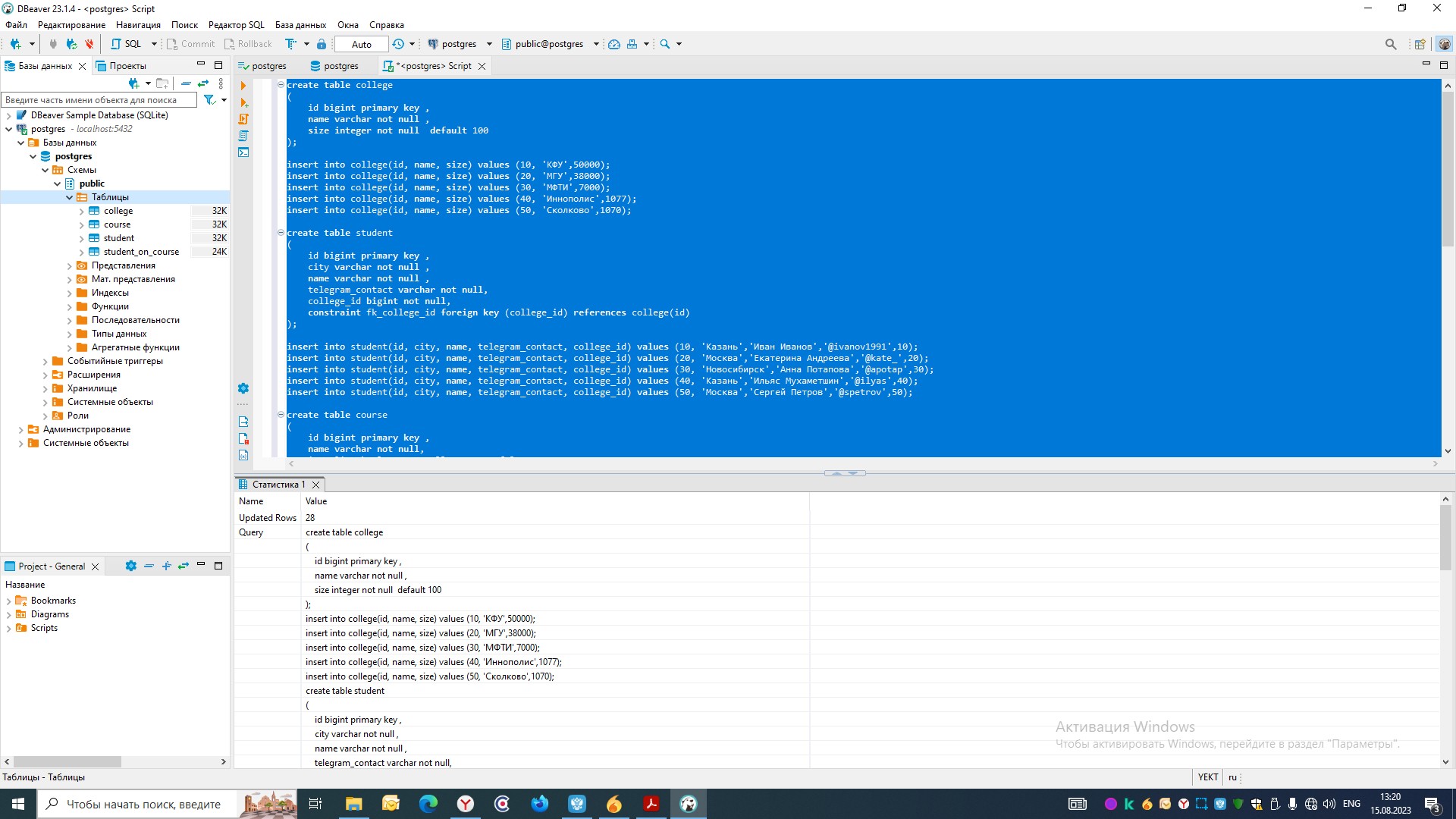
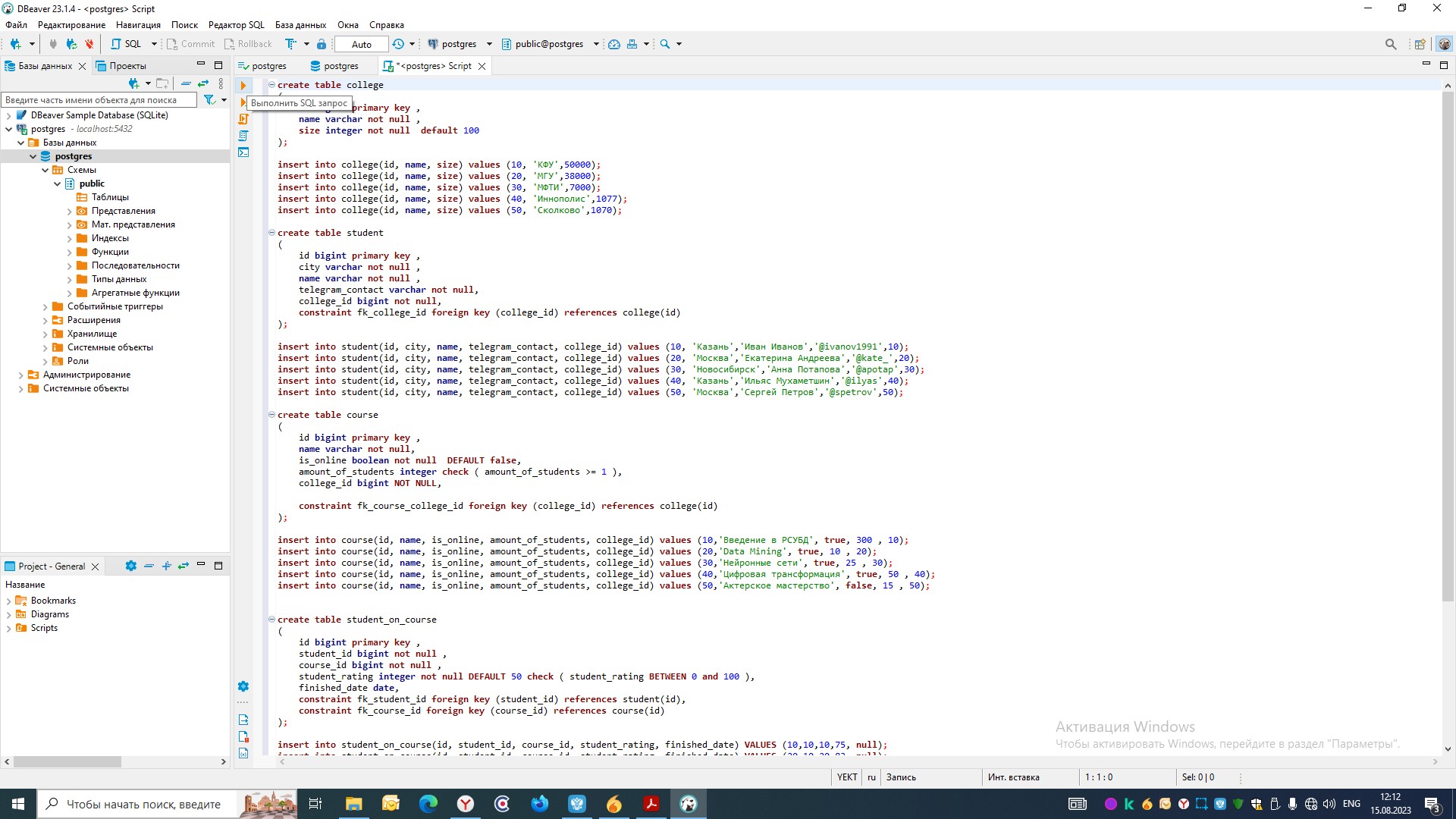
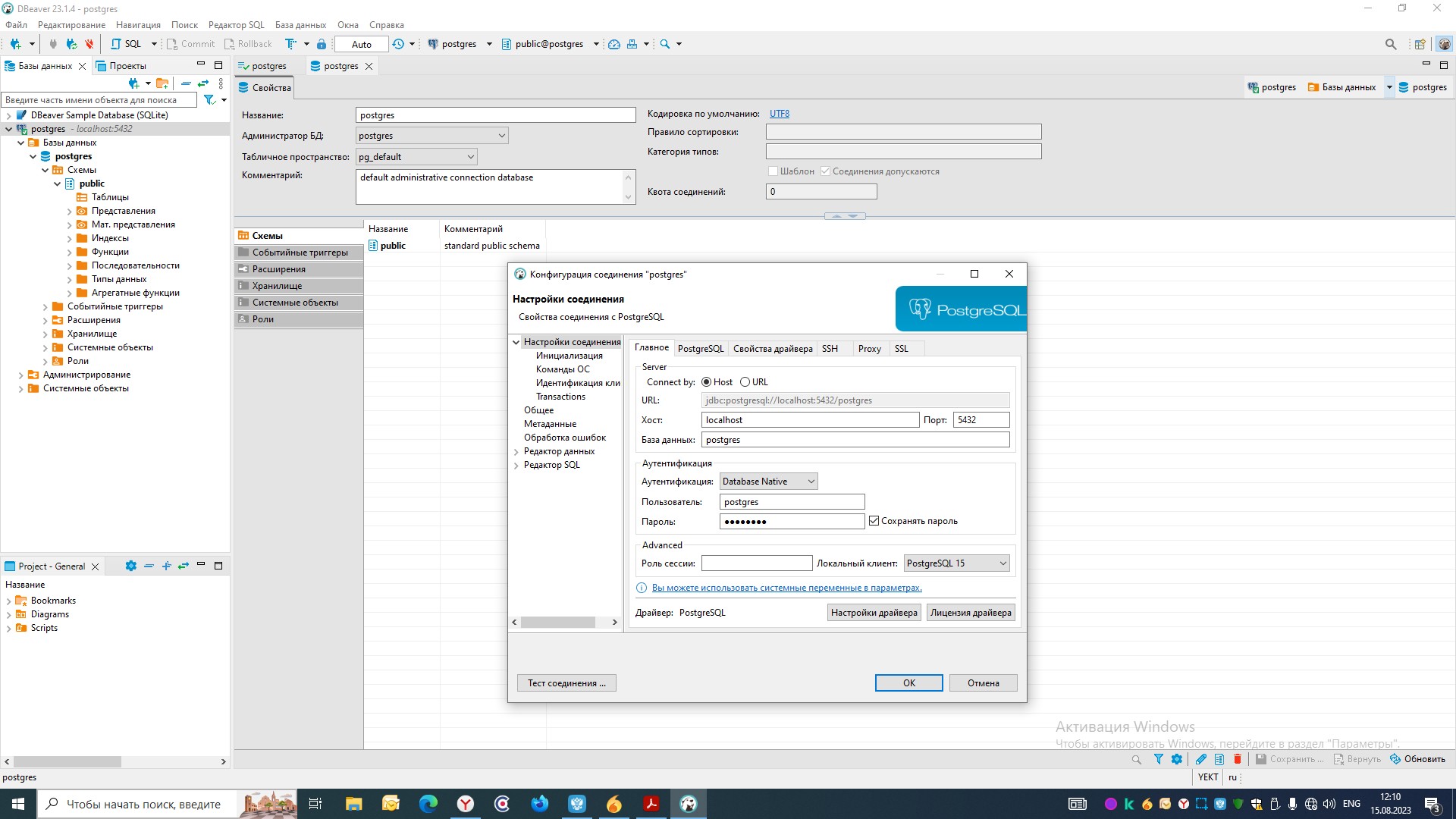
**Лабораторная работа по теме «Язык запросов SQL»**

**Автор: Сорокин Максим Евгеньевич**

**Формулировка задания:**

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.

**Решение:**



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

**select**

**name**, telegram\_contact

**from**

student

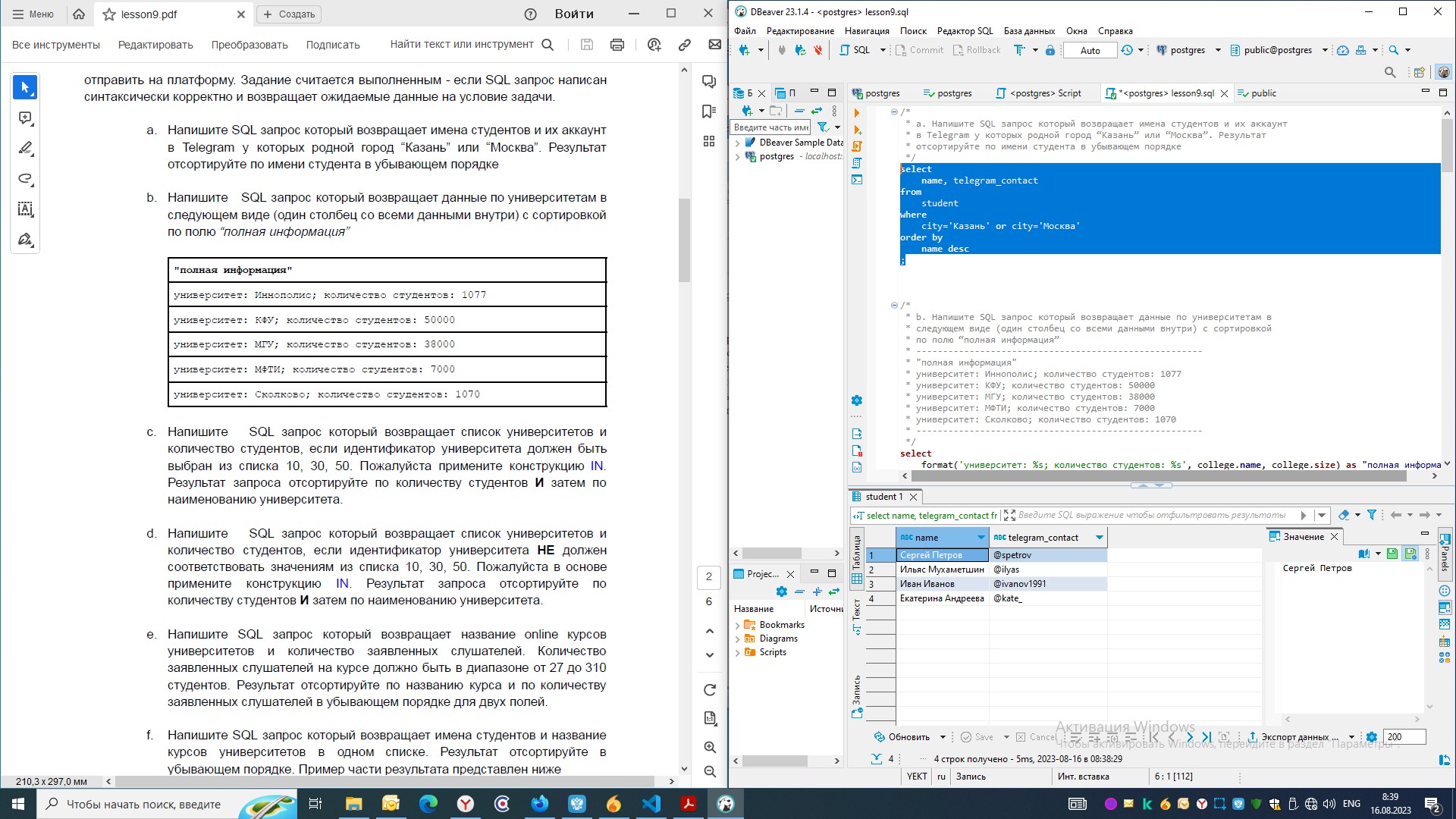
**where**

city='Казань' **or** city='Москва'

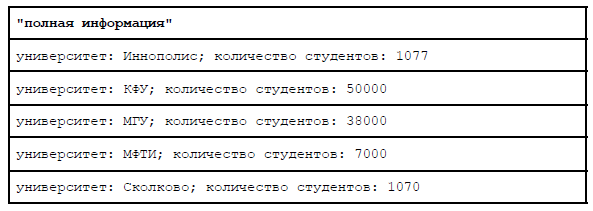
**order** **by**

**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”



**select**

format('университет: %s; количество студентов: %s', college.**name**, college.**size**) **as** "полная информация"

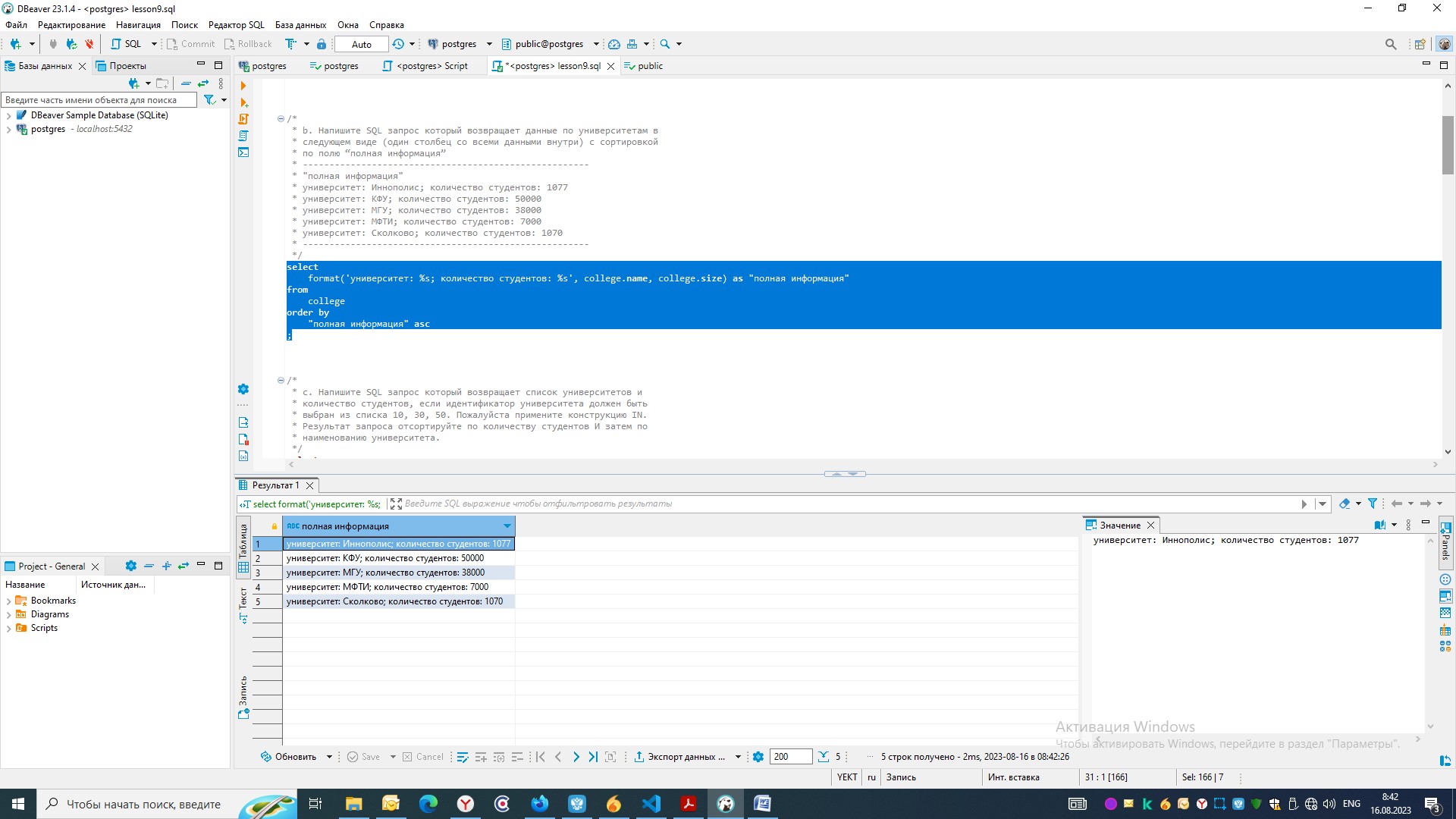
**from**

college

**order** **by**

"полная информация" **asc**

;



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.

**select**

**name**, **size**

**from**

college

**where**

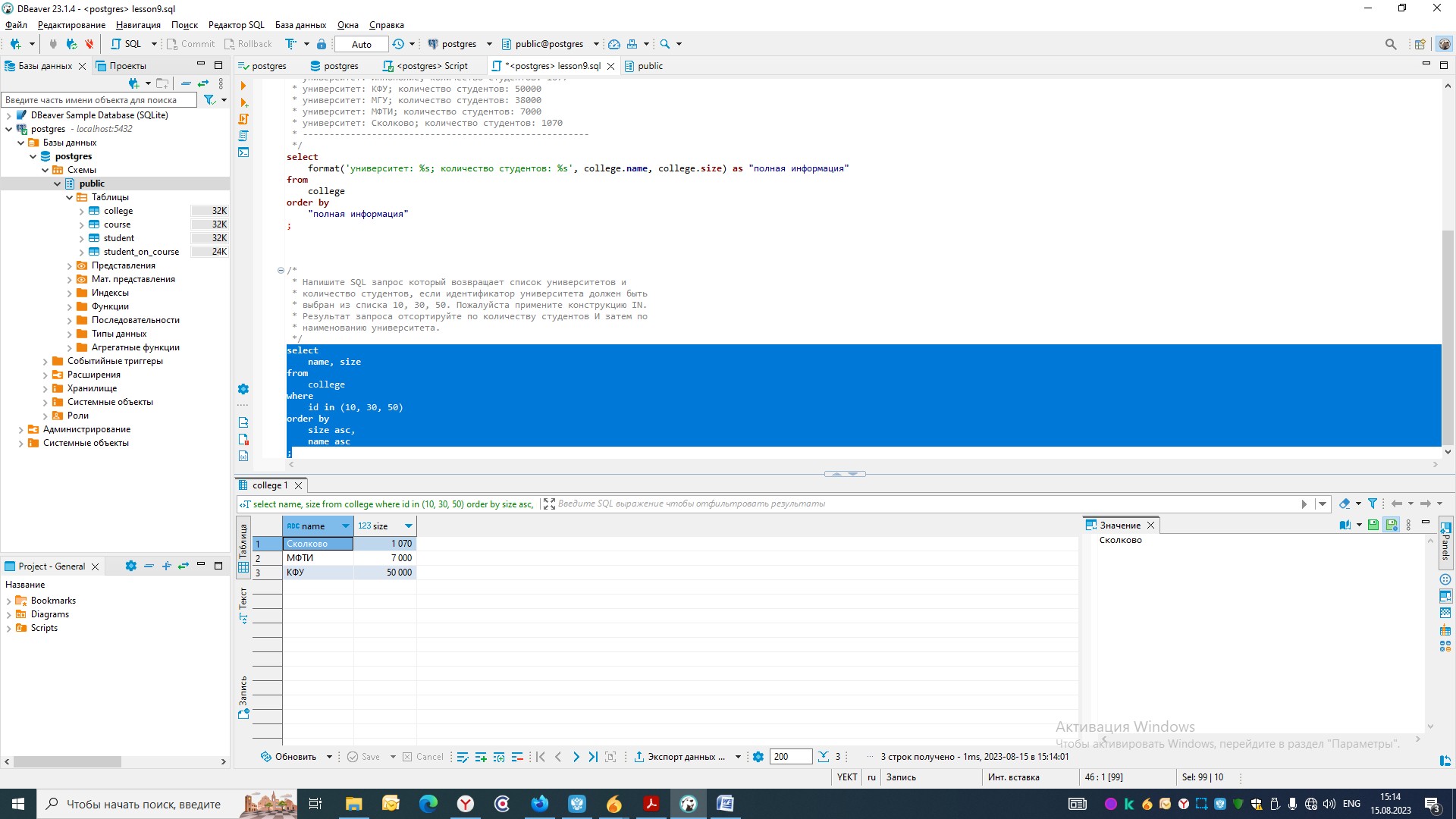
id **in** (10, 30, 50)

**order** **by**

**size** **asc**,

**name** **asc**

;



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.

**select**

**name**, **size**

**from**

college

**where**

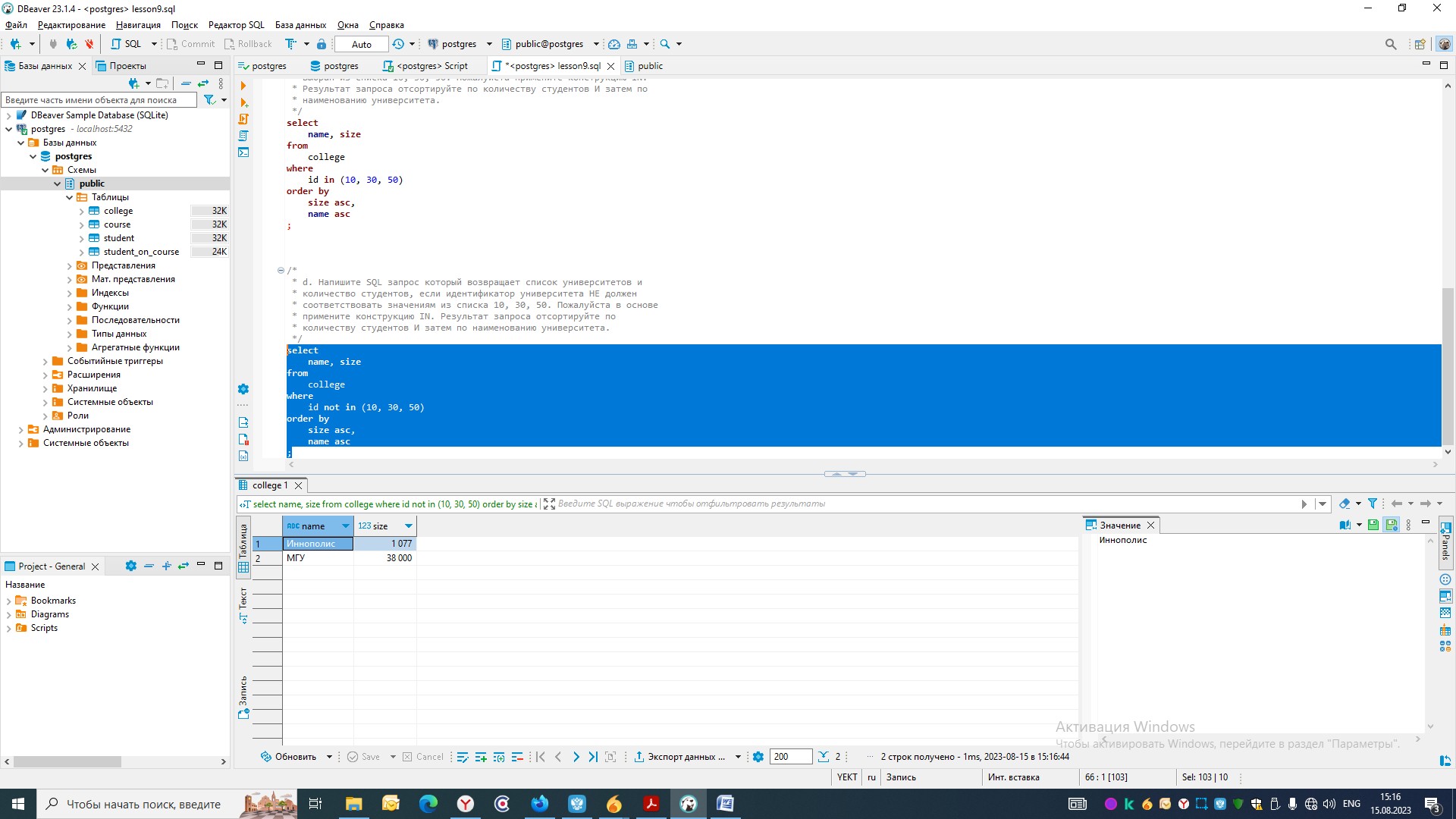
id **not** **in** (10, 30, 50)

**order** **by**

**size** **asc**,

**name** **asc**

;



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.

**select**

**name** **as** "название online курсов университетов",

amount\_of\_students **as** "количество заявленных слушателей"

**from**

course

**where**

is\_online=**true**

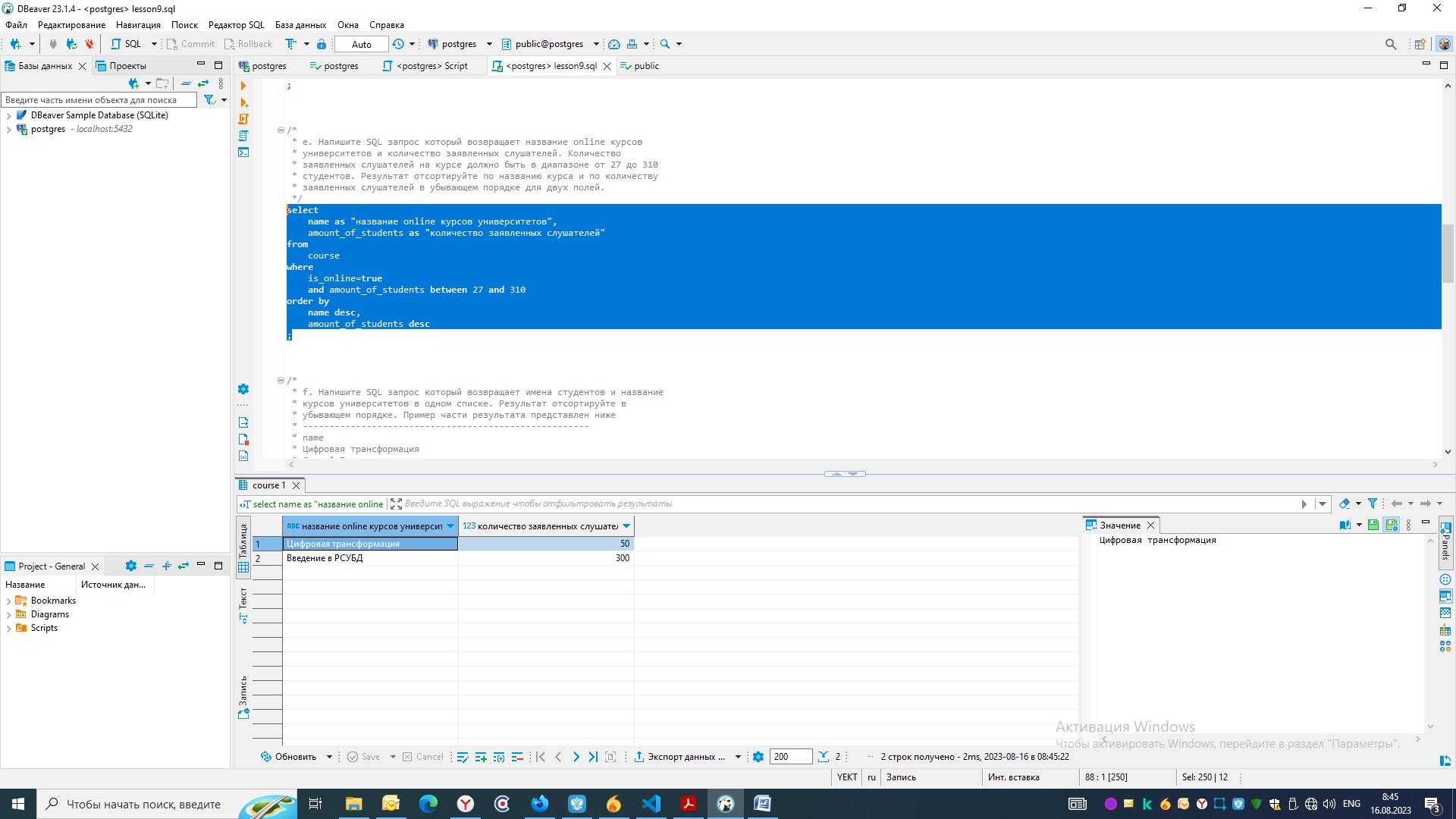
**and** amount\_of\_students **between** 27 **and** 310

**order** **by**

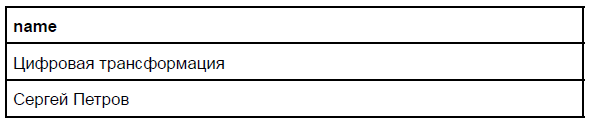
**name** **desc**,

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



**select**

**name**

**from**

student

**union**

**select**

**name**

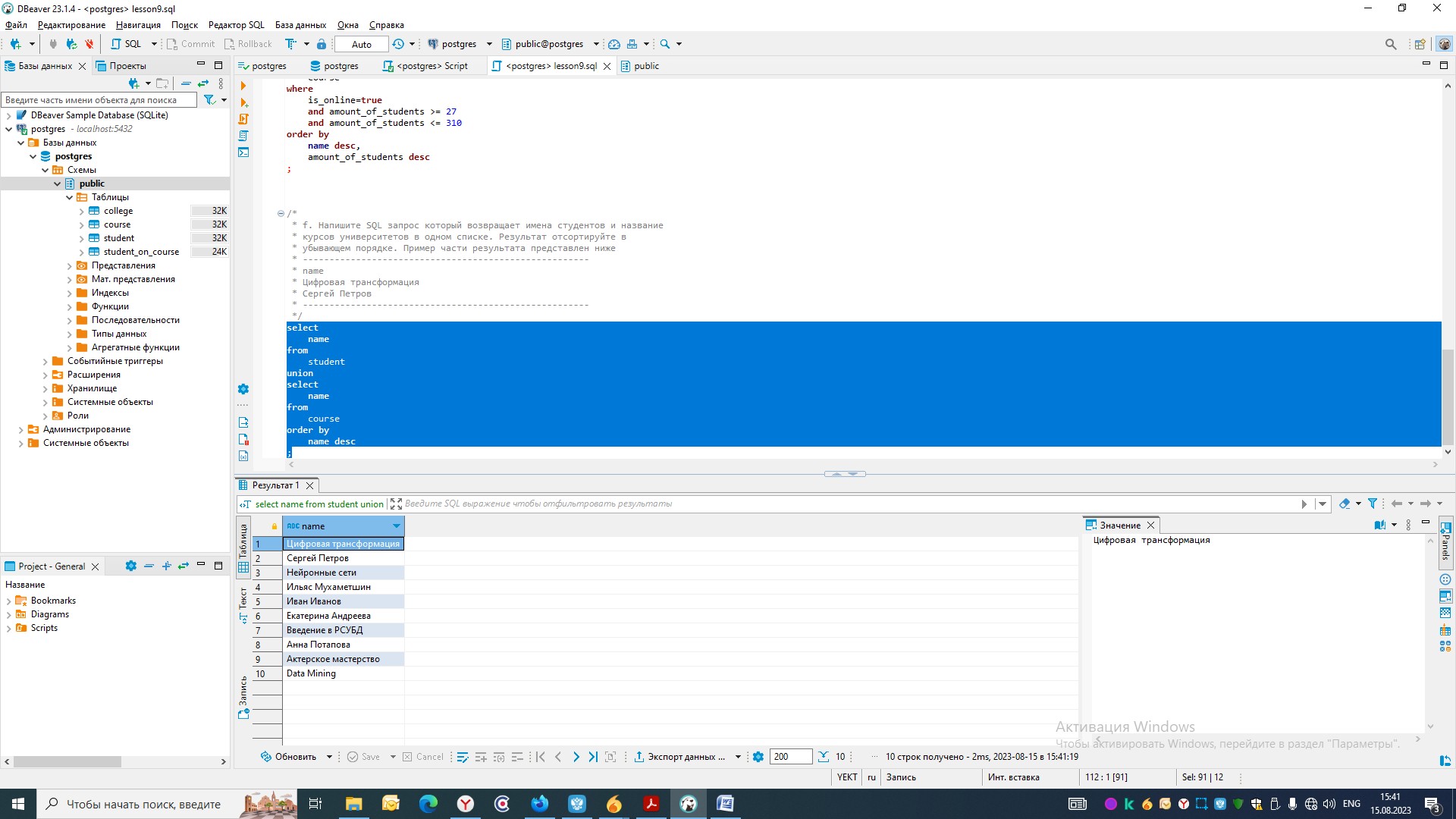
**from**

course

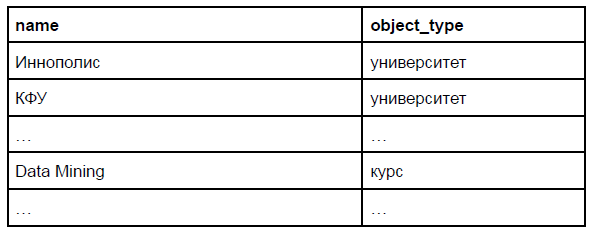
**order** **by**

**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



**select**

**name**,

'университет' **as** object\_type

**from**

college

**union**

**select**

**name**,

'курс' **as** object\_type

**from**

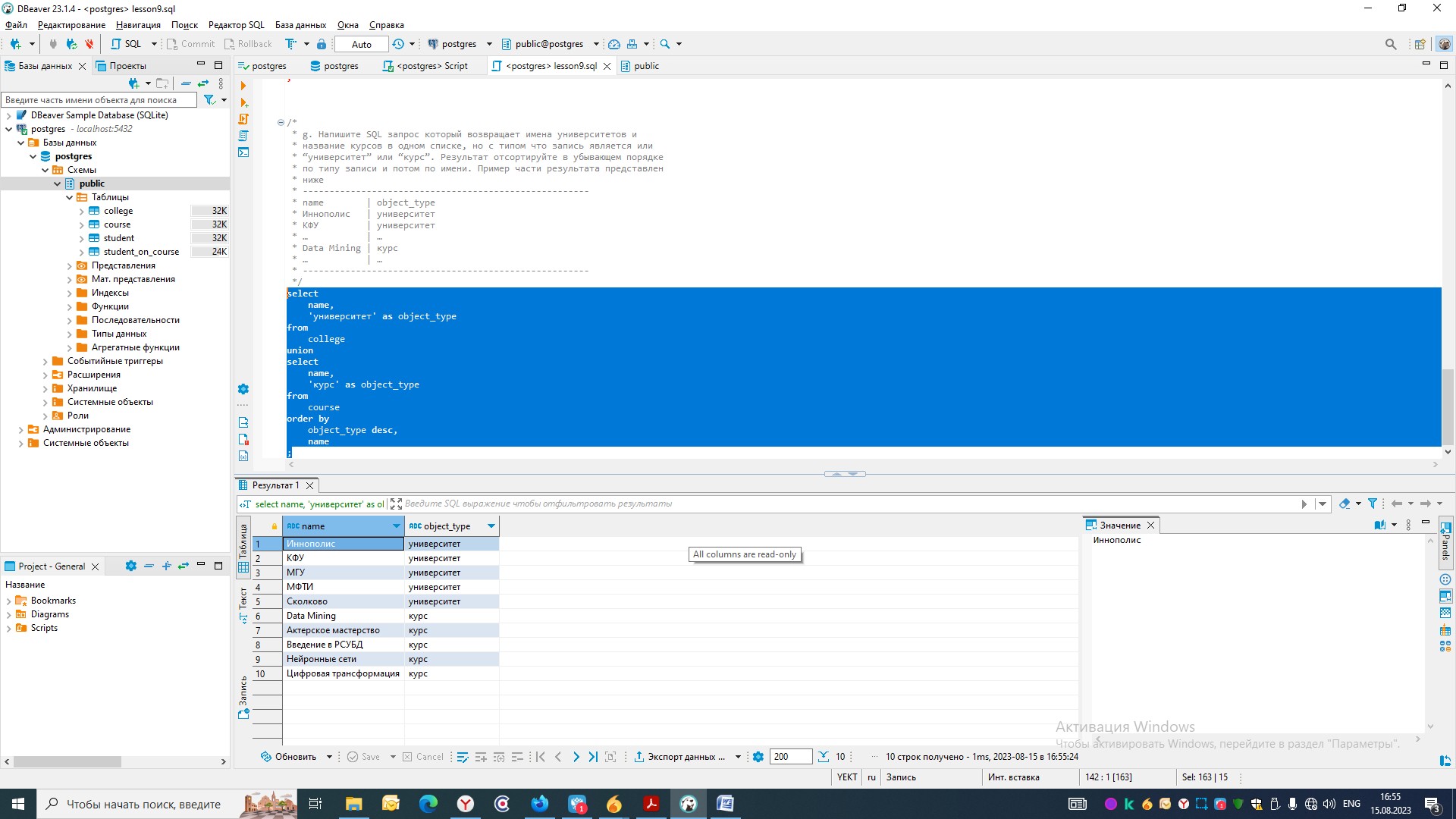
course

**order** **by**

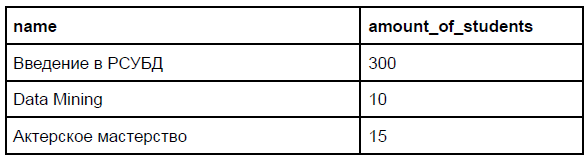
object\_type **desc**,

**name**

;



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.

**select**

**name**,

amount\_of\_students

**from**

course

**order** **by**

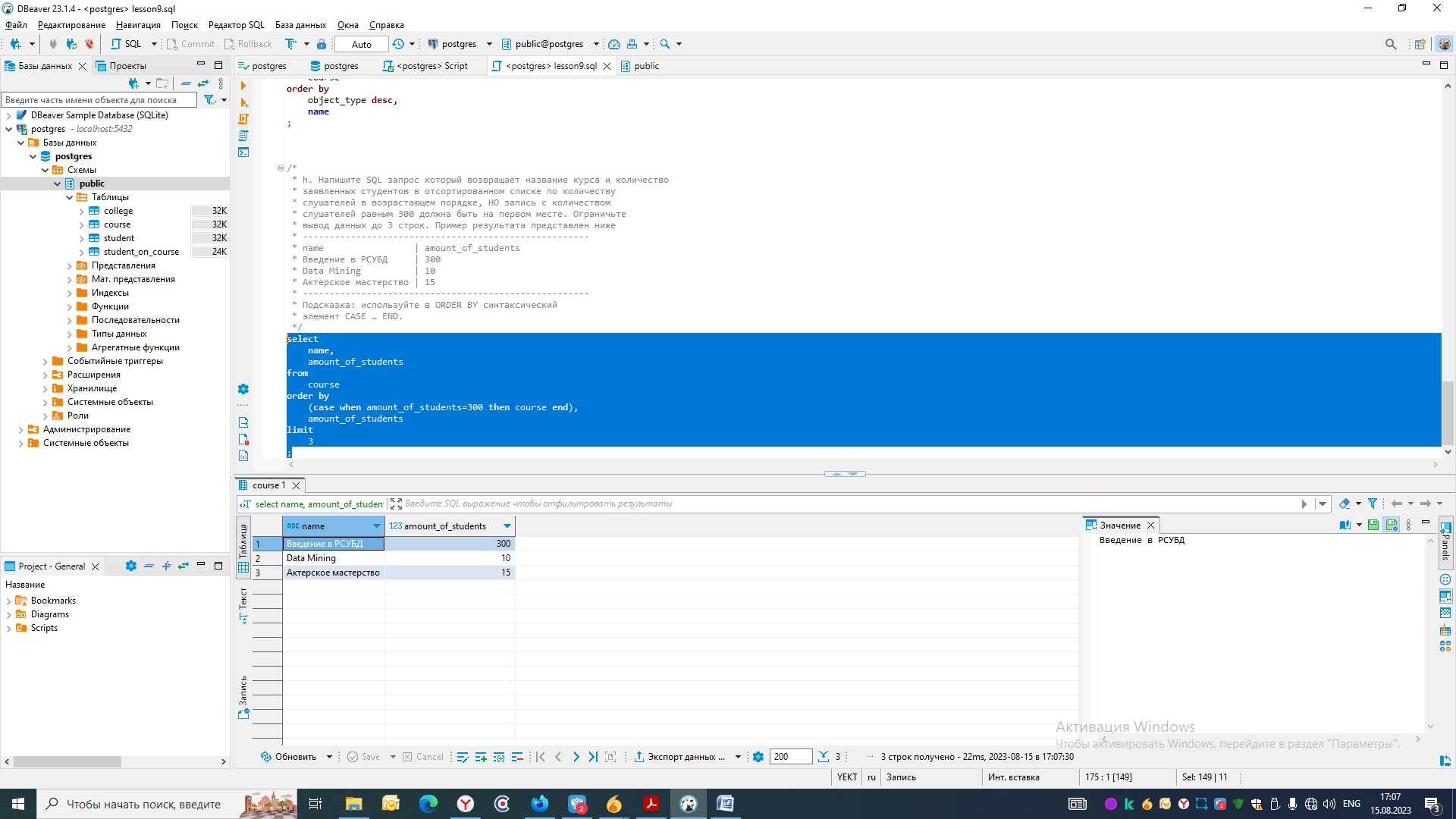
(**case** **when** amount\_of\_students=300 **then** course **end**),

amount\_of\_students

**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.

**select**

\*

**from**

course

;

------------------------------------------------------

**with**

search\_course

**as** (

**select**

id **as** search\_course\_id

**from**

course

**where**

**name**='Data Mining'

)

**insert** **into** course

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

**select**

60, 'Machine Learning', **false**, 17, search\_course\_id

**from**

search\_course

;

------------------------------------------------------

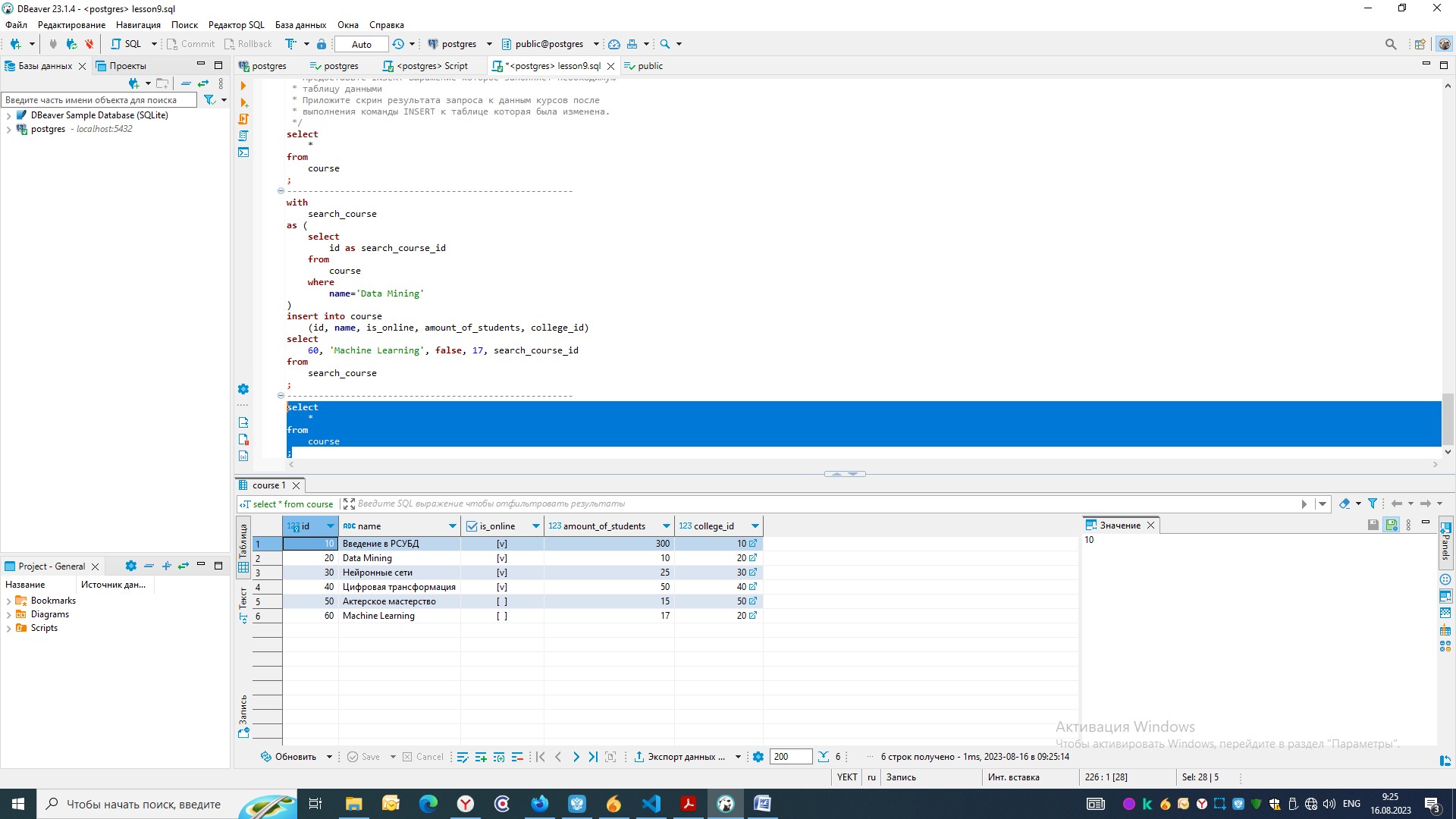
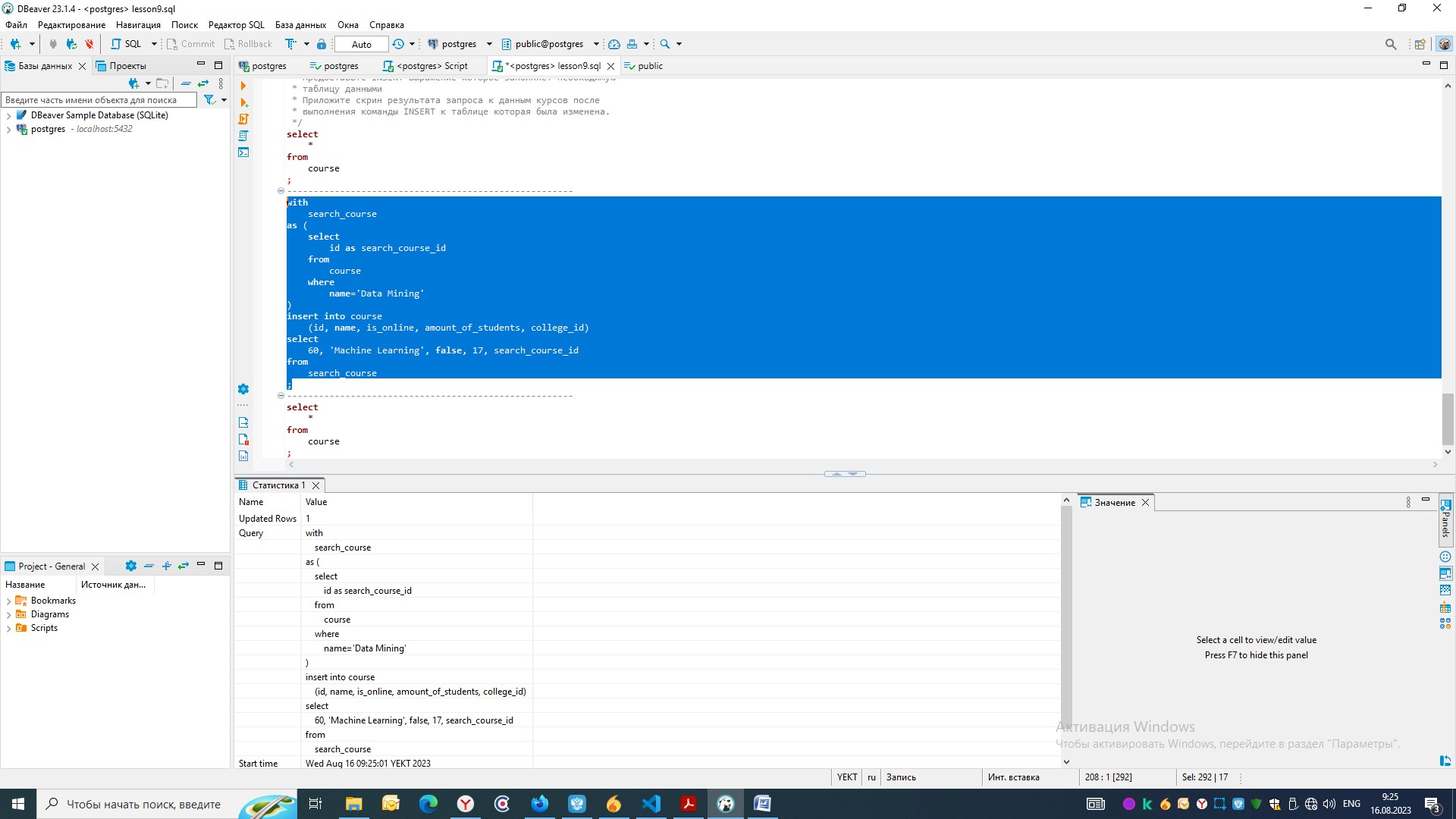
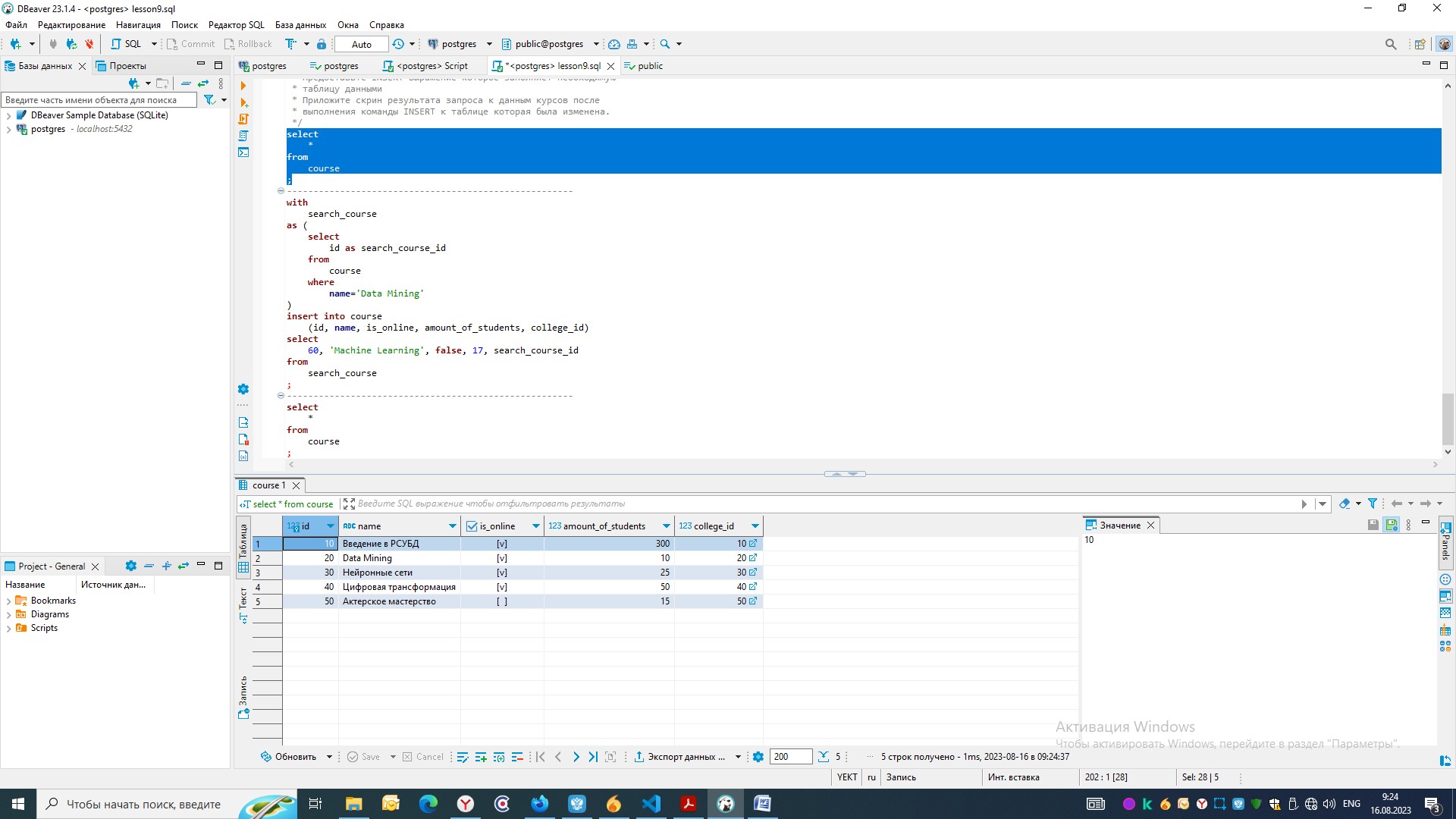
**select**

\*

**from**

course

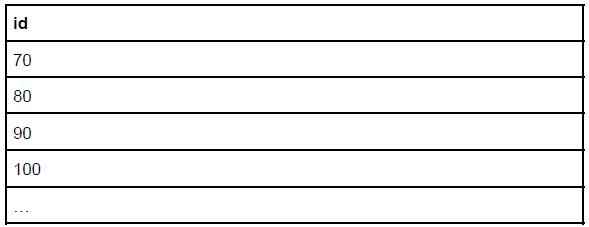
;



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.



(

(

**select**

id

**from**

course

)

**except**

(

**select**

id

**from**

student\_on\_course

)

)

**union**

(

(

**select**

id

**from**

student\_on\_course

)

**except**

(

**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



**select**

student.**name** **as** "student\_name",

course.**name** **as** "course\_name",

college.**name** **as** "student\_college",

student\_on\_course.student\_rating **as** "student\_rating"

**from**

student

**left** **join**

student\_on\_course

**on**

student\_on\_course.student\_id=student.id

**left** **join**

course

**on**

course.id=student\_on\_course.course\_id

**left** **join**

college

**on**

college.id=student.college\_id

**where**

student\_on\_course.student\_rating > 50

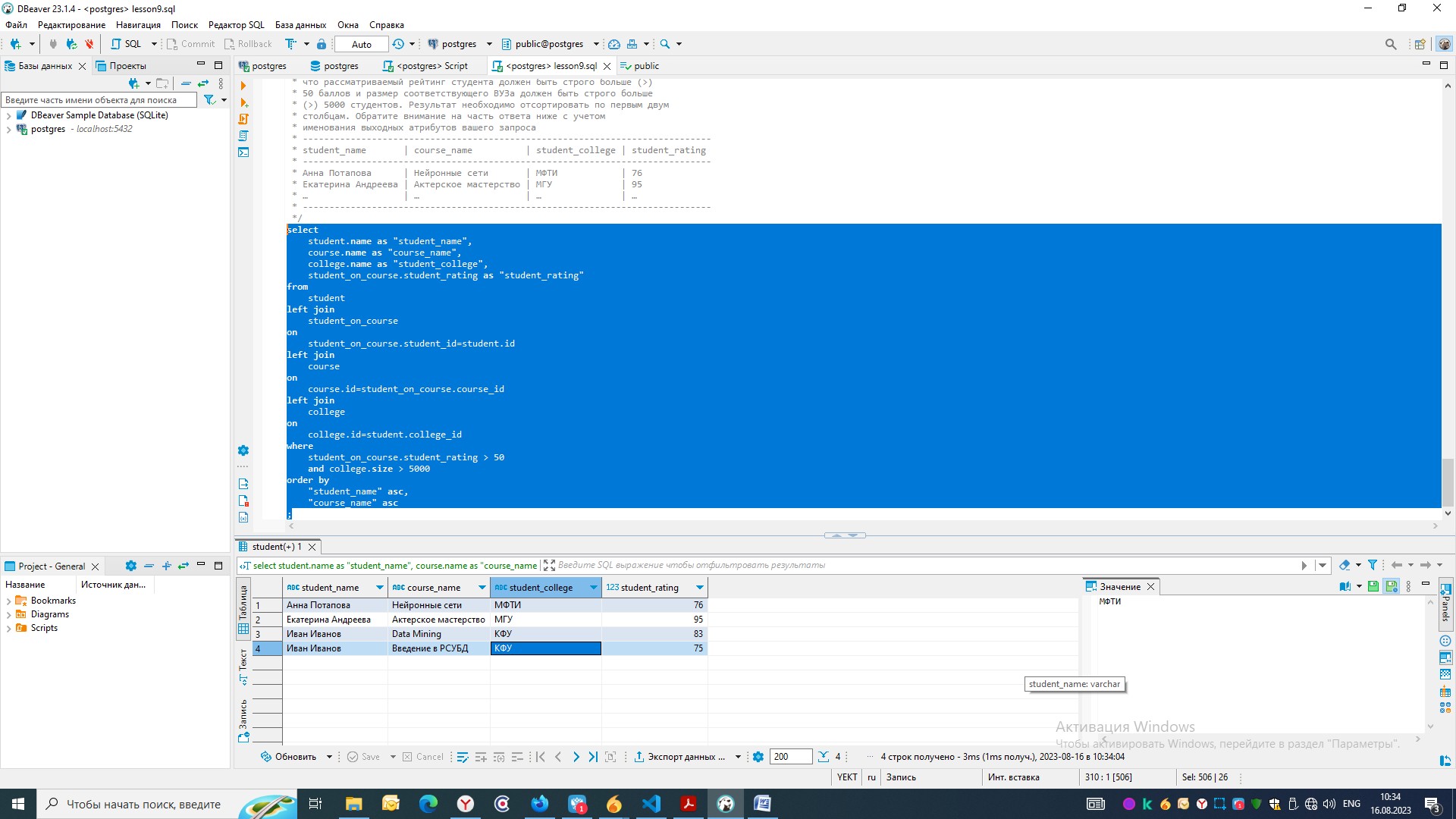
**and** college.**size** > 5000

**order** **by**

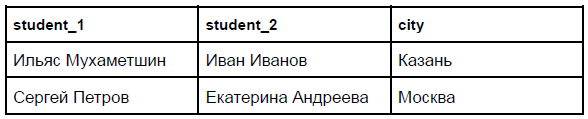
"student\_name" **asc**,

"course\_name" **asc**

;



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



**select** **distinct** **on** ("city")

\*

**from**

(

**select**

student\_1.**name** **as** "student\_1"

, student\_2.**name** **as** "student\_2"

, student\_2.city **as** "city"

**from**

student **as** student\_1

**left** **join**

student **as** student\_2

**on**

student\_2.city=student\_1.city

**and** **not** student\_2.**name**=student\_1.**name**

) **as** list

**where**

**not** "city"='NULL'

**group** **by**

"city",

"student\_1",

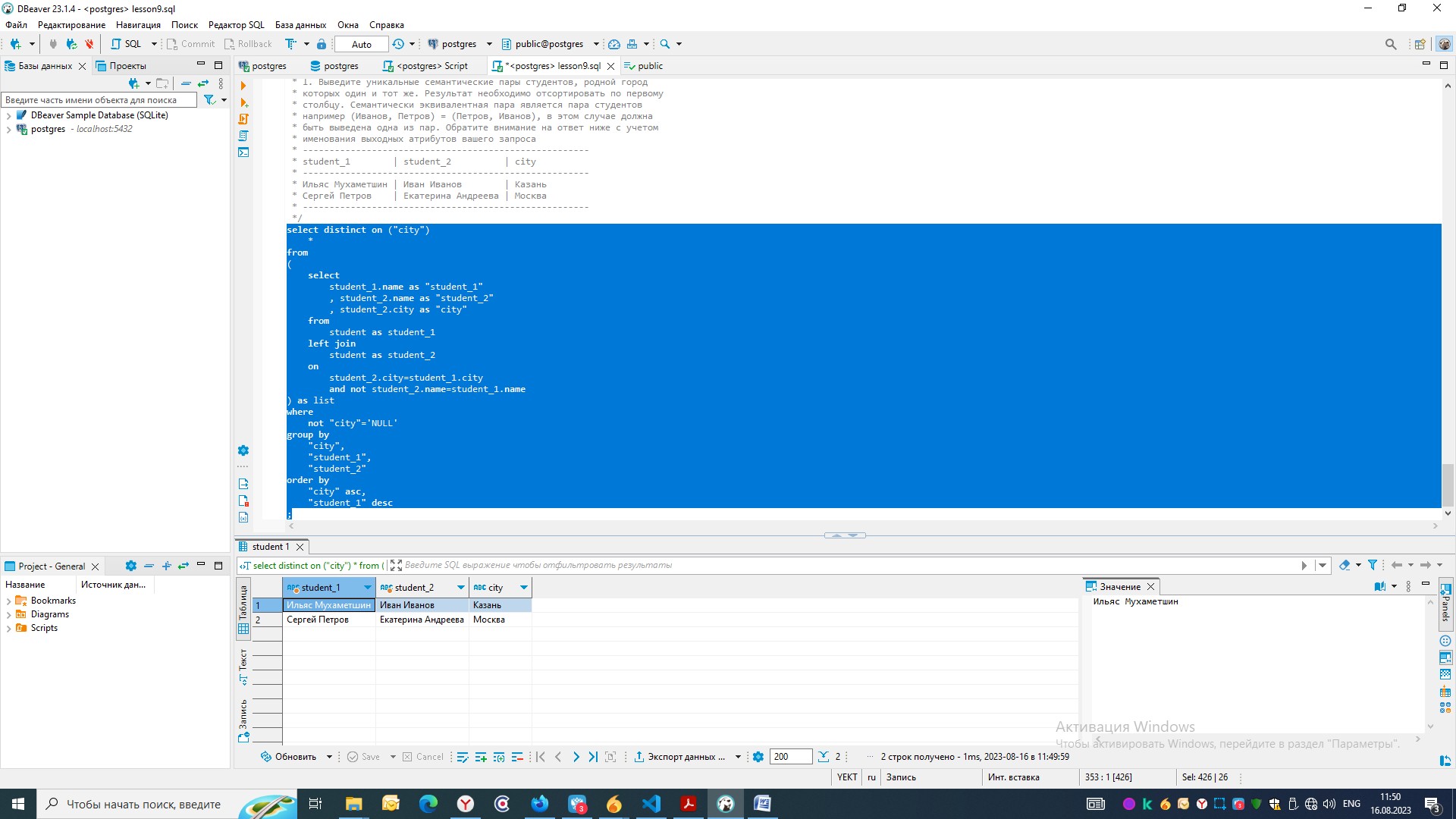
"student\_2"

**order** **by**

"city" **asc**,

"student\_1" **desc**

;



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.

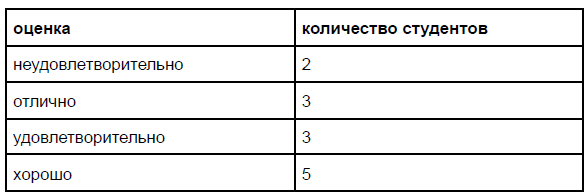
ЕСЛИ оценка < 30 ТОГДА неудовлетворительно

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.



**select**

list."student\_rating" **as** "оценка"

, **count**(student.**name**) **as** "количество студентов"

**from**

(

**select**

**case**

**when**

student\_on\_course.student\_rating < 30

**then**

'неудовлетворительно'

**when**

student\_on\_course.student\_rating >= 30

**and** student\_on\_course.student\_rating < 60

**then**

'удовлетворительно'

**when**

student\_on\_course.student\_rating >= 60

**and** student\_on\_course.student\_rating < 85

**then**

'хорошо'

**else**

'отлично'

**end** **as** "student\_rating"

, student\_on\_course.student\_id **as** "student\_id"

**from**

student\_on\_course

) **as** list

**left** **join**

student

**on**

student.id="student\_id"

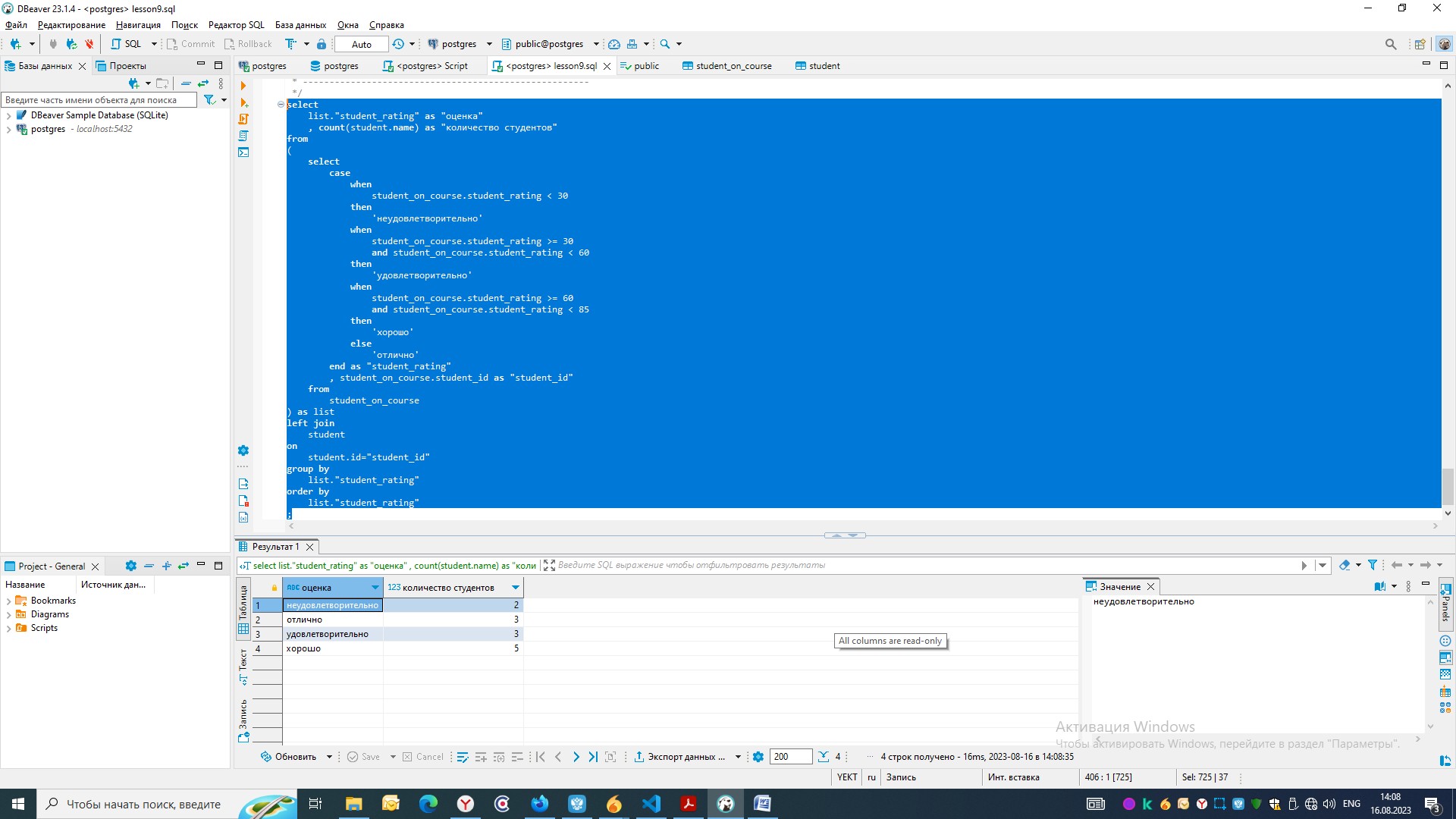
**group** **by**

list."student\_rating"

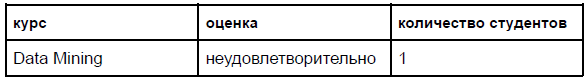
**order** **by**

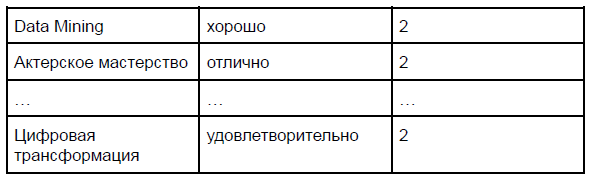
list."student\_rating"

;



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.





**select**

course.**name** **as** "курс"

, list."student\_rating" **as** "оценка"

, **count**(student.**name**) **as** "количество студентов"

**from**

(

**select**

**case**

**when**

student\_on\_course.student\_rating < 30

**then**

'неудовлетворительно'

**when**

student\_on\_course.student\_rating >= 30

**and** student\_on\_course.student\_rating < 60

**then**

'удовлетворительно'

**when**

student\_on\_course.student\_rating >= 60

**and** student\_on\_course.student\_rating < 85

**then**

'хорошо'

**else**

'отлично'

**end** **as** "student\_rating"

, student\_on\_course.student\_id **as** "student\_id"

, student\_on\_course.student\_rating **as** "student\_rating\_number"

**from**

student\_on\_course

) **as** list

**left** **join**

student

**on**

student.id=list."student\_id"

**left** **join**

student\_on\_course

**on**

student\_on\_course.student\_id=student.id

**left** **join**

course

**on**

course.id=student\_on\_course.course\_id

**group** **by**

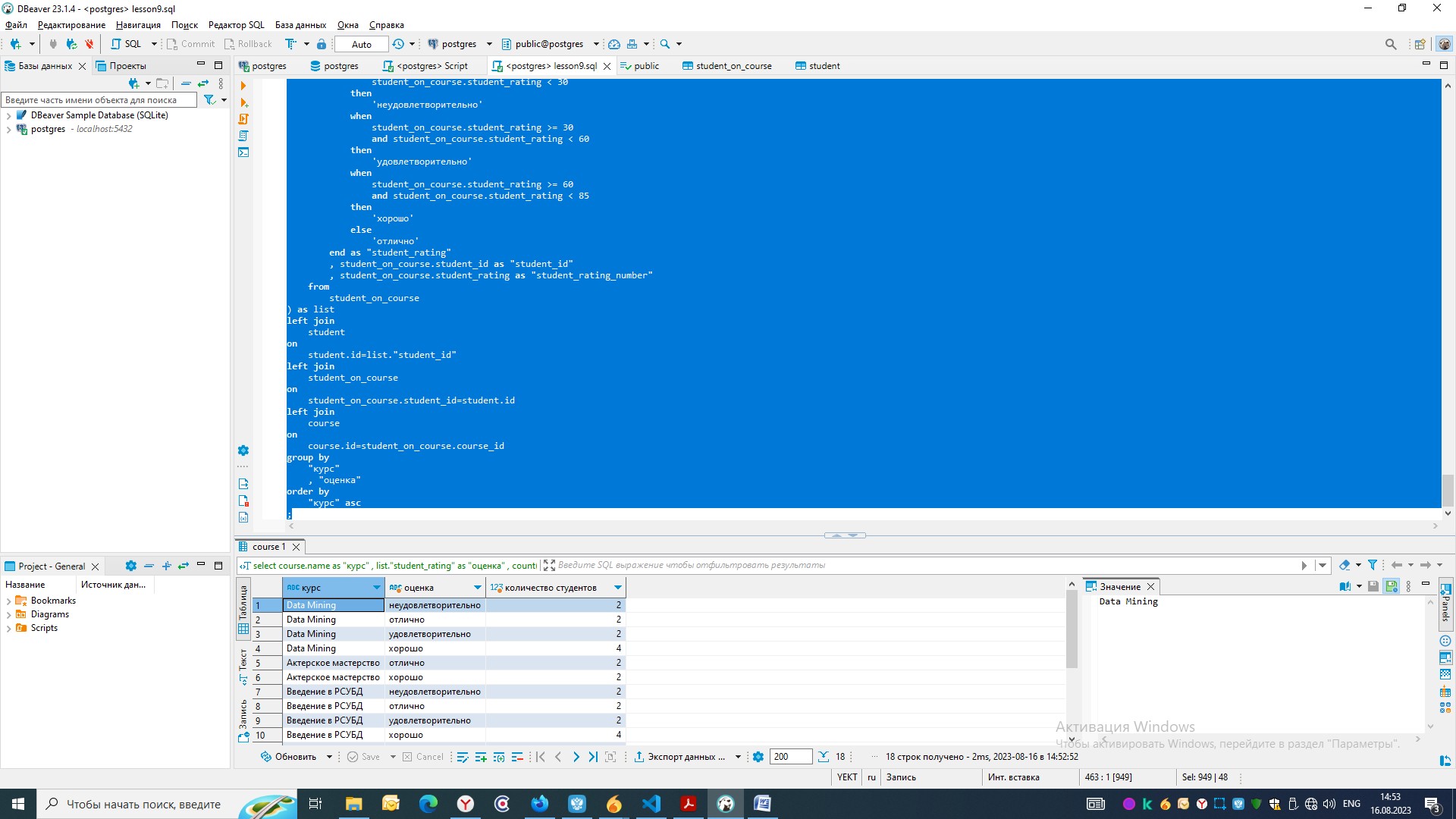
"курс"

, "оценка"

**order** **by**

"курс" **asc**

;



К решению приложены файлы:

«lesson9.sql» для PostgreSQL

Ссылка на GitHub:

<https://github.com/SotGE/innopolis2023/tree/main/lesson9>