# 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/

2. Скачать **одно** из приложений

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

- dbeaver - https://dbeaver.io/

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

3. 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

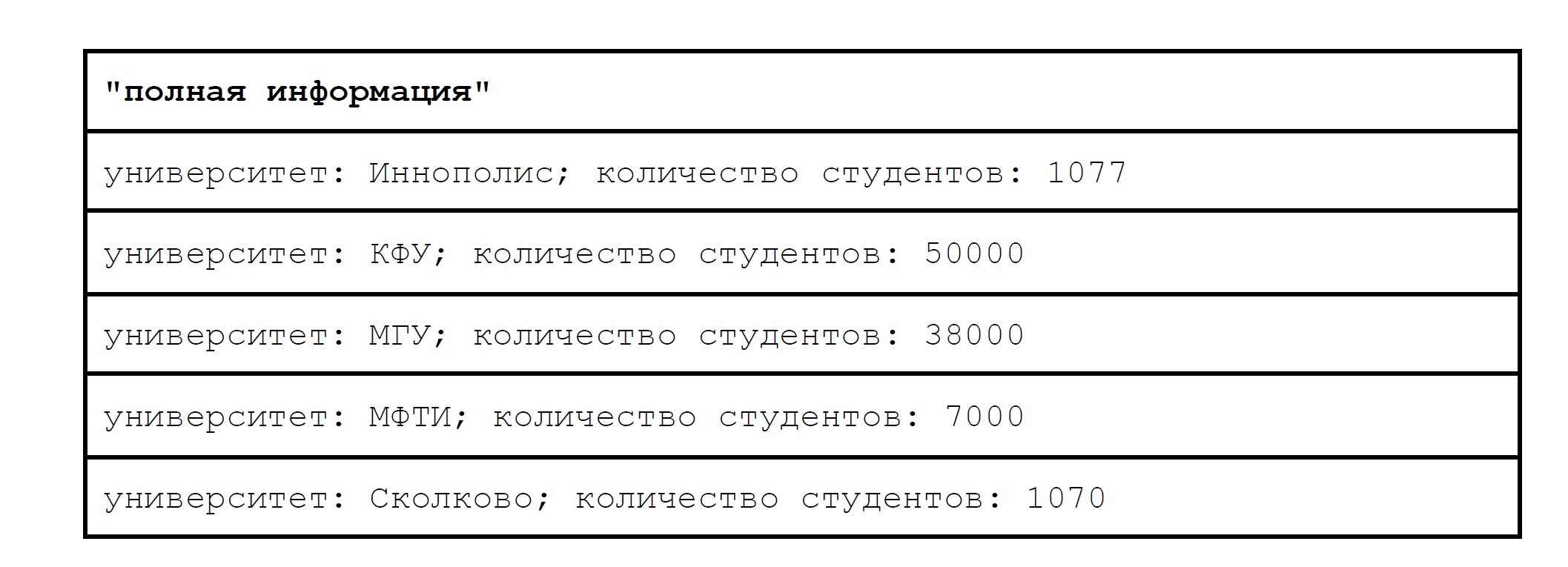
DROP TABLE *имя\_таблицы;*

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.

a. 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

b. Напишите SQL запрос который возвращает данные по университетам в следующем виде (один столбец со всеми данными внутри) с сортировкой по полю *“полная информация”*

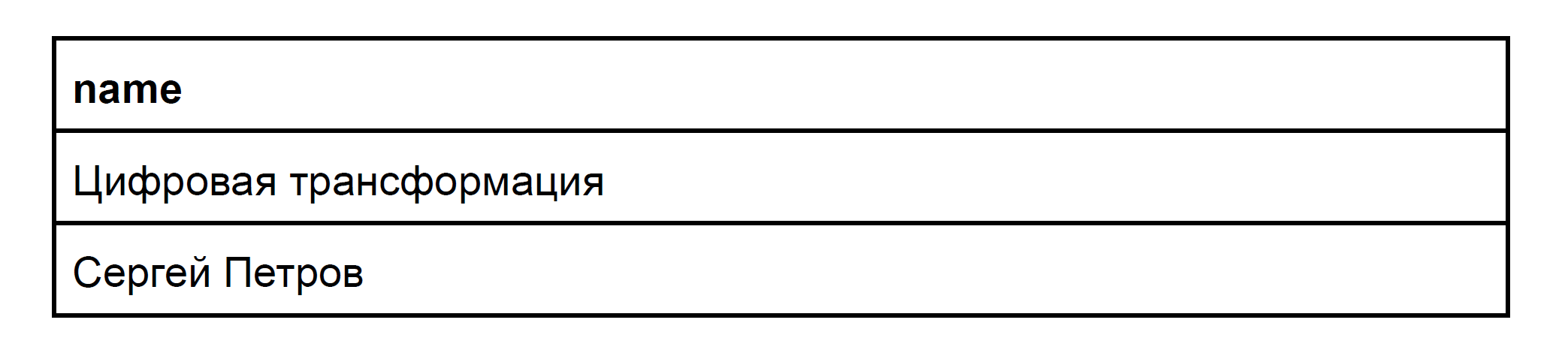


c. Напишите SQL запрос который возвращает список университетов и количество студентов, если идентификатор университета должен быть выбран из списка 10, 30, 50. Пожалуйста примените конструкцию IN. Результат запроса отсортируйте по количеству студентов **И** затем по наименованию университета.

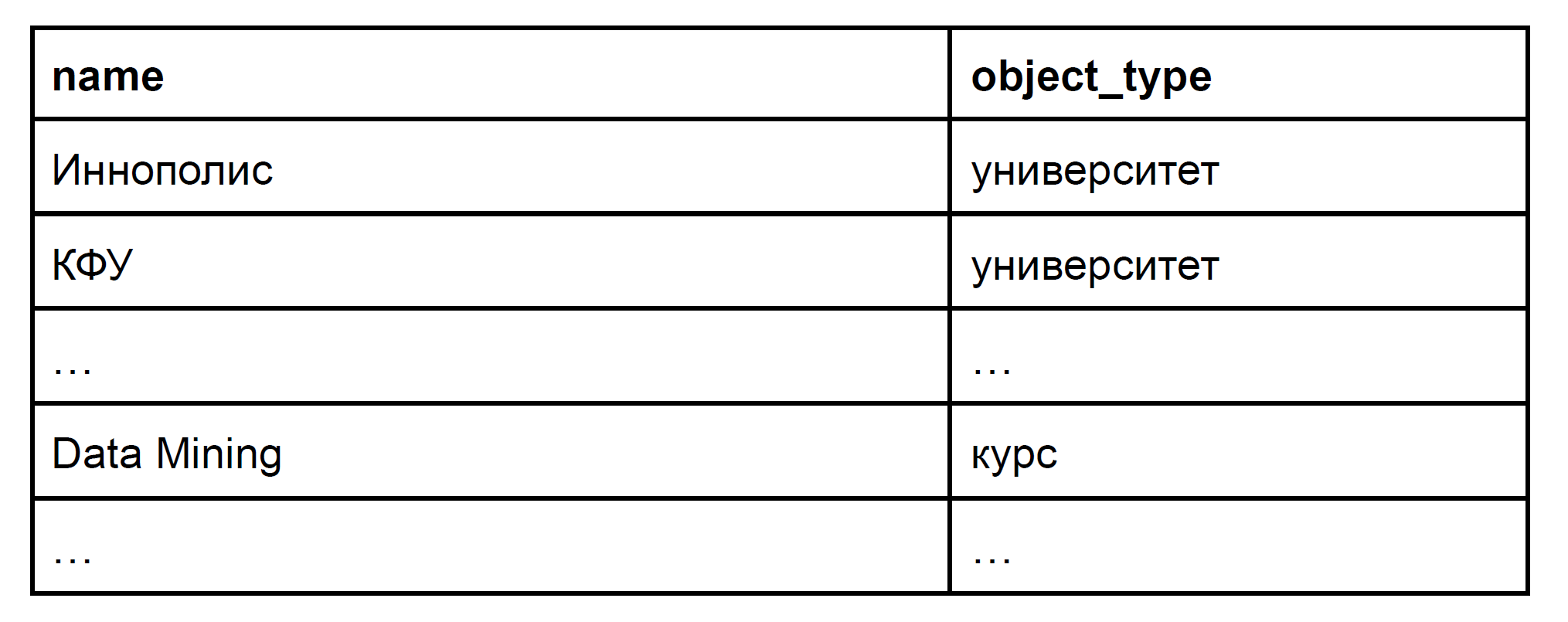
d. Напишите SQL запрос который возвращает список университетов и количество студентов, если идентификатор университета **НЕ** должен соответствовать значениям из списка 10, 30, 50. Пожалуйста в основе примените конструкцию IN. Результат запроса отсортируйте по количеству студентов **И** затем по наименованию университета.

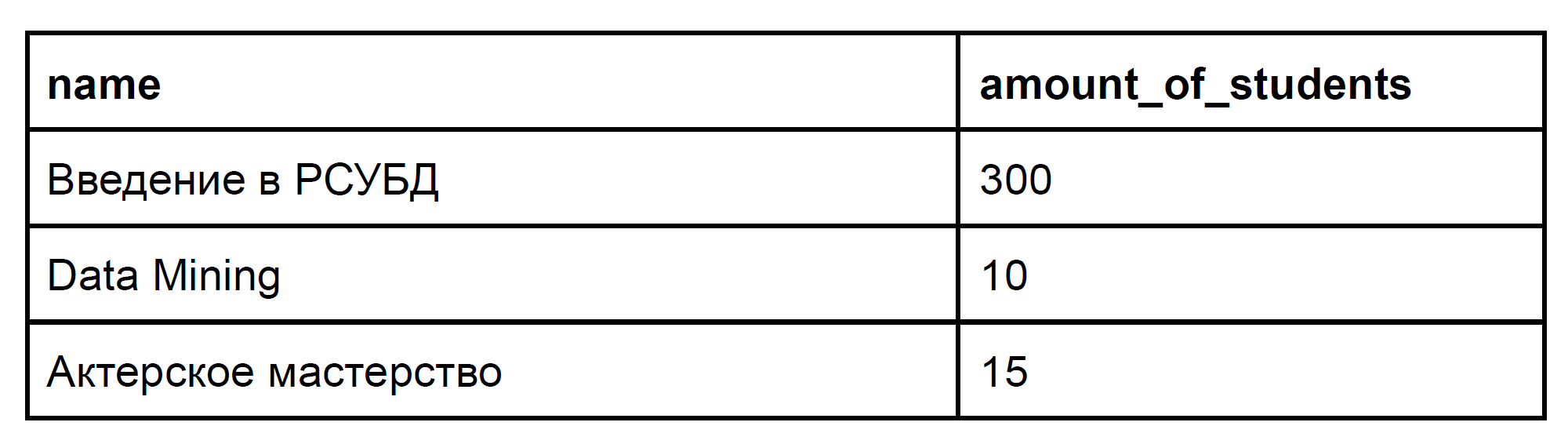
e. Напишите SQL запрос который возвращает название online курсов университетов и количество заявленных слушателей. Количество заявленных слушателей на курсе должно быть в диапазоне от 27 до 310 студентов. Результат отсортируйте по названию курса и по количеству заявленных слушателей в убывающем порядке для двух полей.

f. Напишите SQL запрос который возвращает имена студентов и название курсов университетов в одном списке. Результат отсортируйте в убывающем порядке. Пример части результата представлен ниже



g. 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

h. 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

i. 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.

j. 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

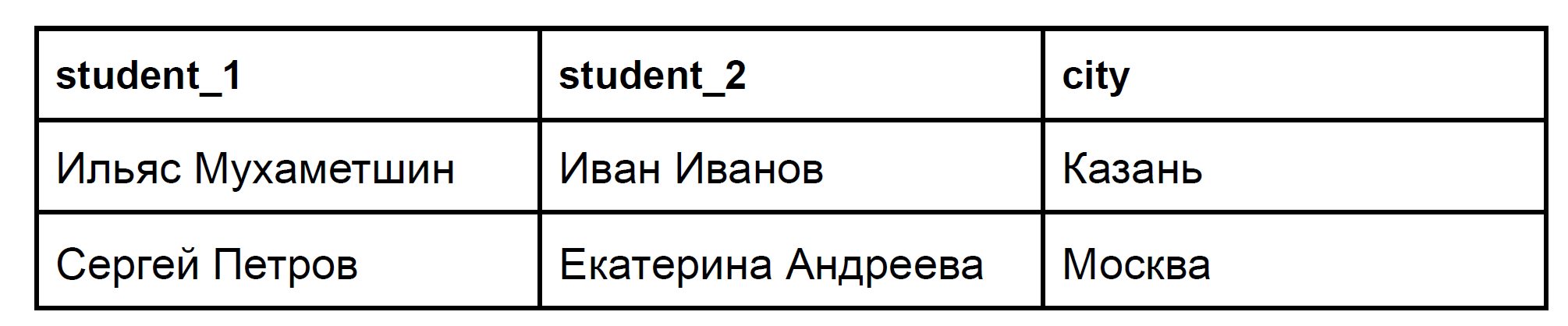
основании атрибута **id** из обеих таблиц. Результат отсортируйте по 1

столбцу. Пример результата представлен ниже.

k. 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



l. 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



m. Напишите SQL запрос который возвращает количество студентов, сгруппированных по их **оценке**. Результат отсортируйте по названию оценки студента. Формула выставления оценки представлена ниже как псевдокод.

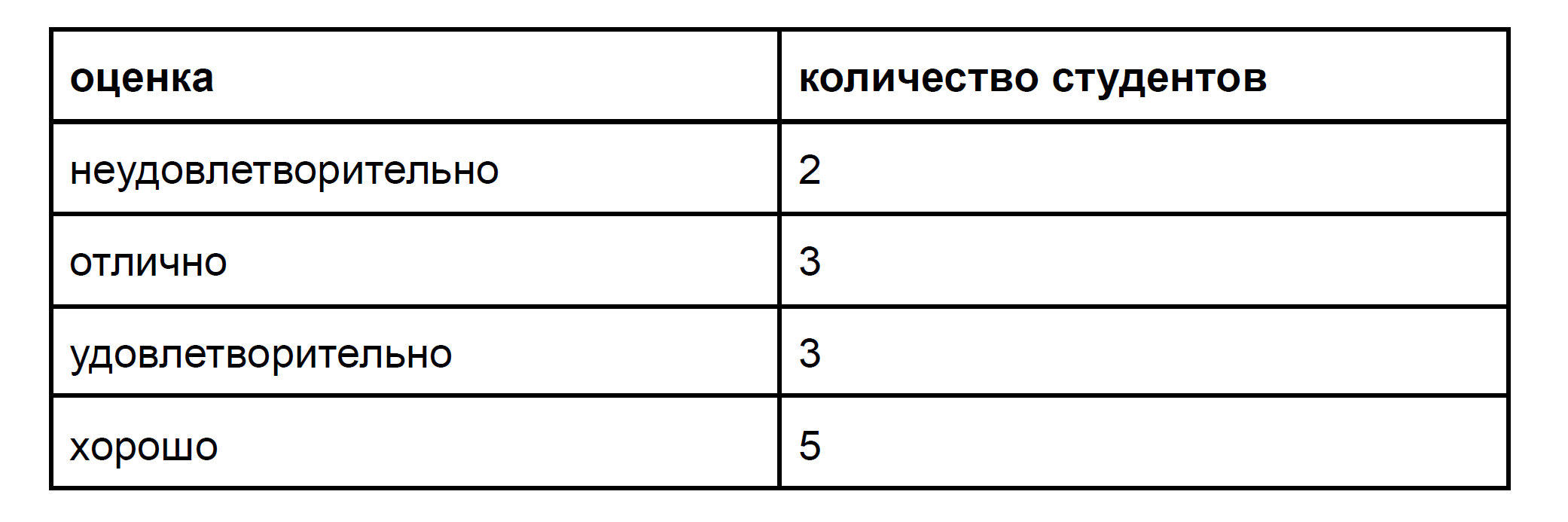
ЕСЛИ оценка < 30 ТОГДА неудовлетворительно

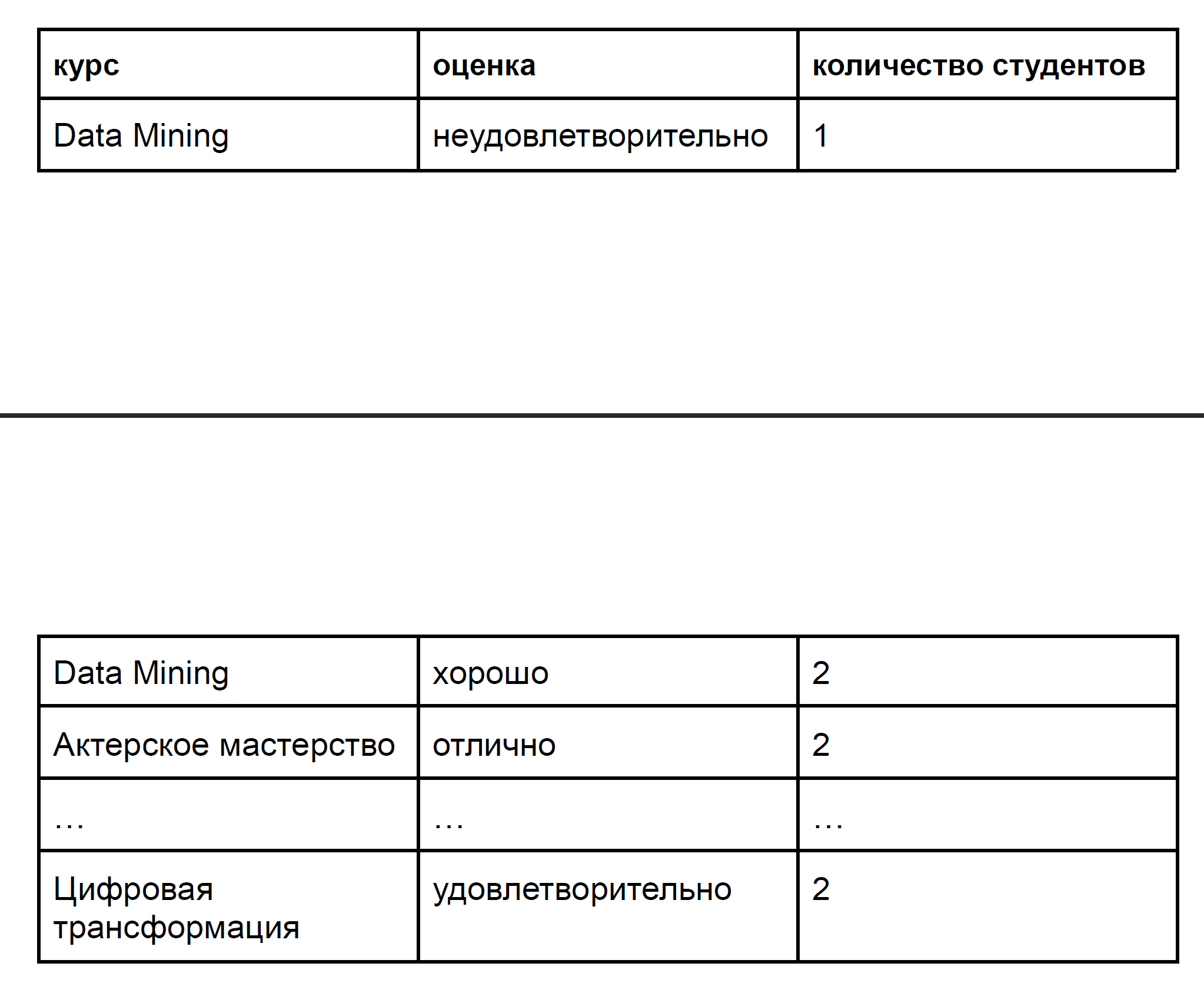
ЕСЛИ оценка >= 30 И оценка < 60 ТОГДА удовлетворительно

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.

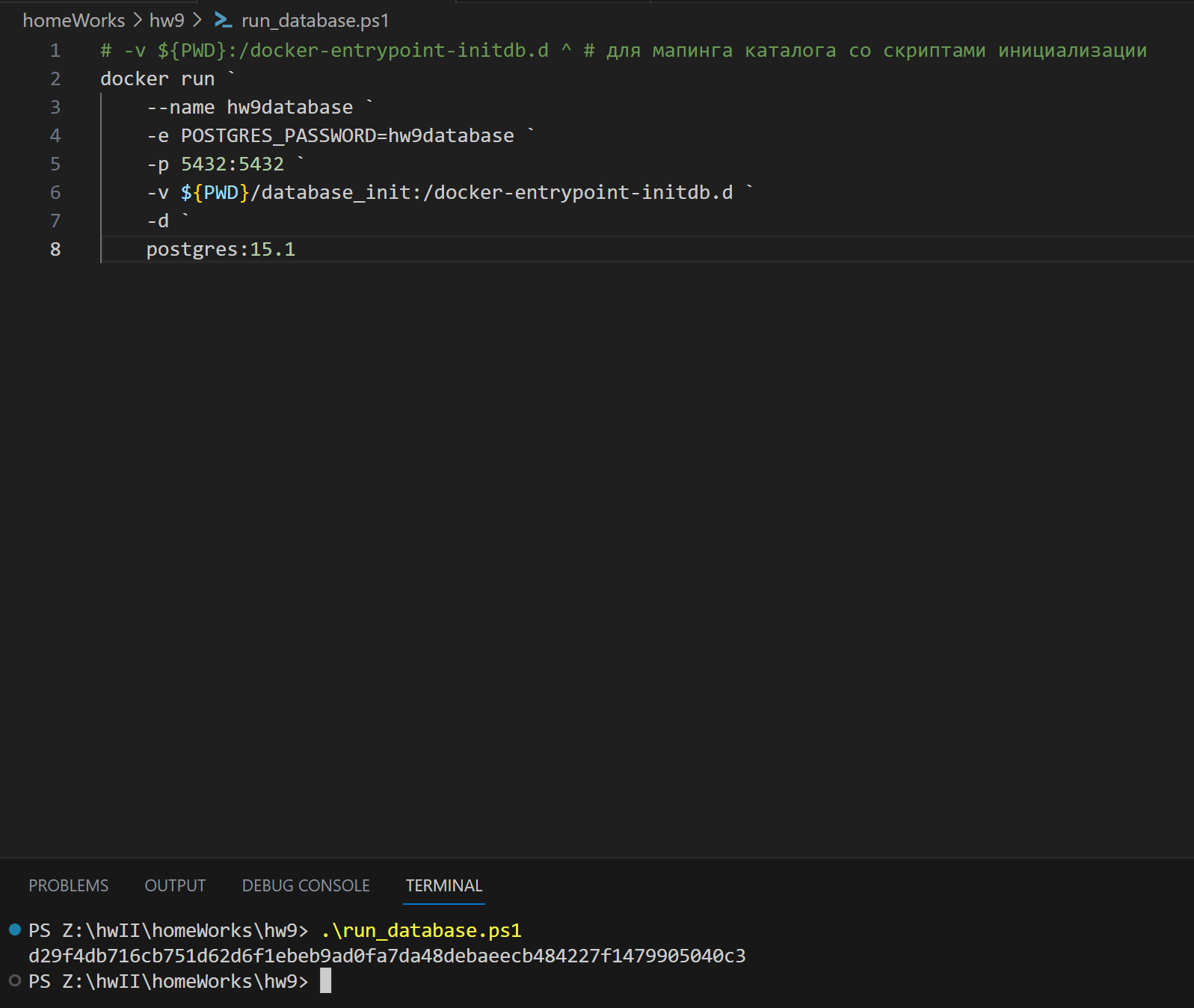
n. Дополните SQL запрос из задания a), с указанием вывода имени курса и количество оценок внутри курса. Результат отсортируйте по названию курса и оценки студента. Пример части результата ниже. Обратите внимание на именование результирующих столбцов в вашем решении. Курс “Machine Learning”, так как у него нет студентов - проигнорируйте, используя соответствующий тип JOIN.



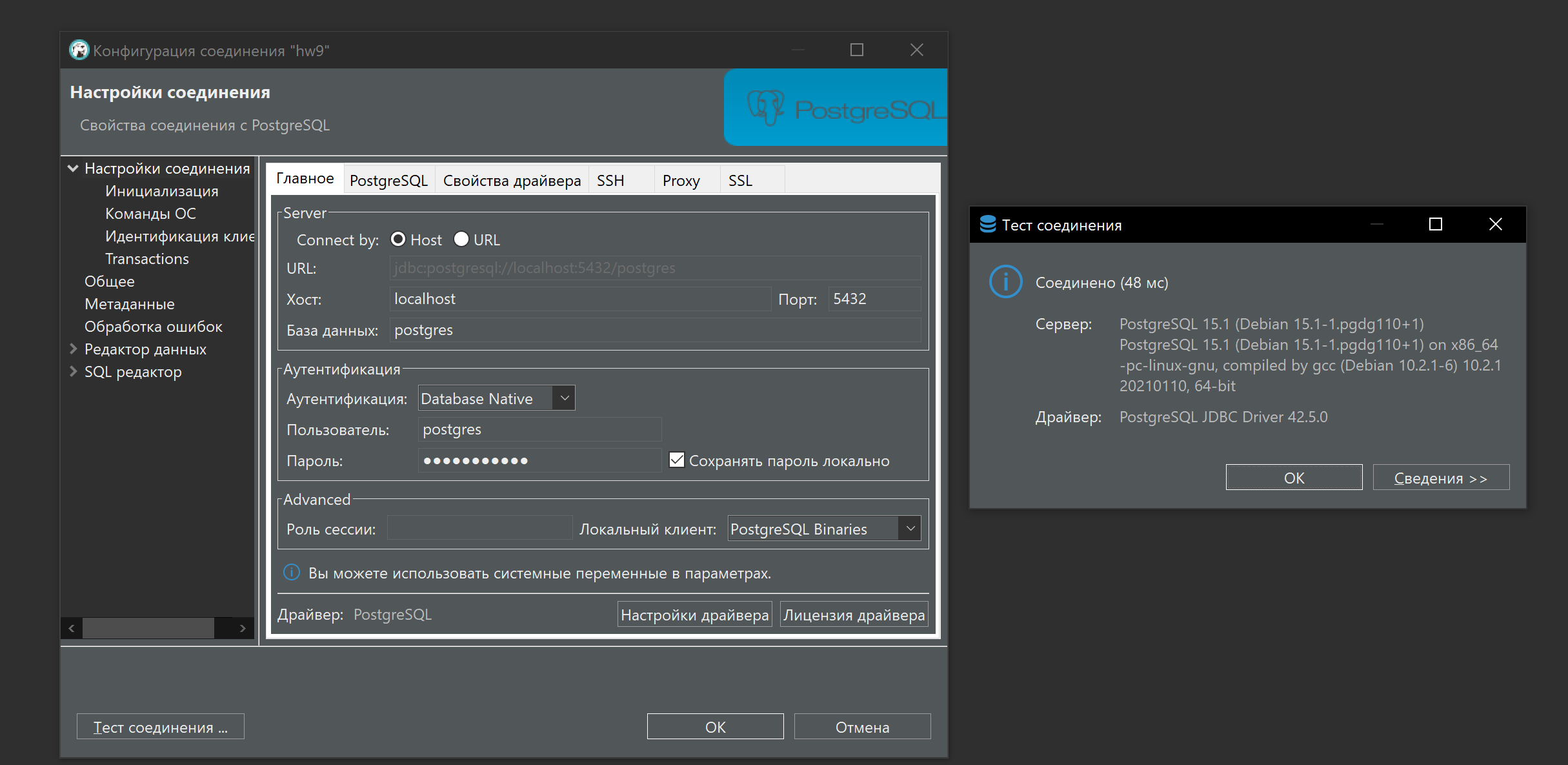
# Решение:

[Ссылка на репозиторий](https://github.com/keeper0null/hwII/tree/main/homeWorks/hw9)

Поднятие базы в docker



Проверка подключения в dbeaver



## Запросы:

/\* a. 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** s.**name**

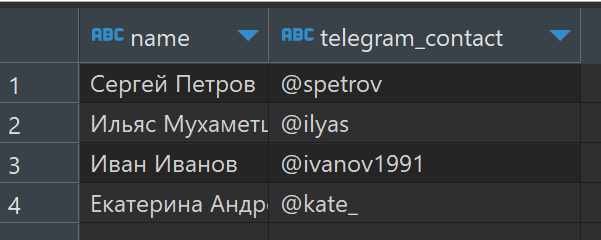
,s.telegram\_contact

**from** student s

**where** s.city **in** ('Казань', 'Москва')

**order** **by**

s.**name** **desc**;

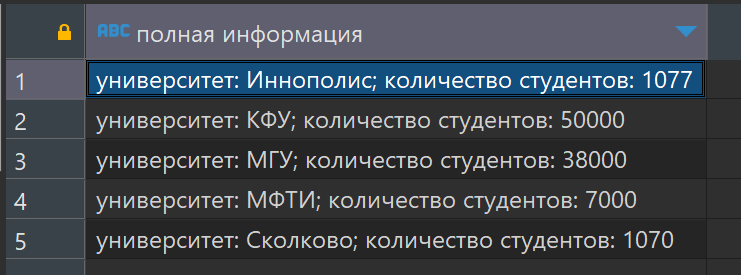


/\* b. Напишите SQL запрос который возвращает данные по университетам в следующем виде (один столбец со всеми данными внутри) с сортировкой по полю "полная информация" \*/

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

**from** college c

**order** **by** "полная информация";



/\* c. 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** c.**name**

,c.**size**

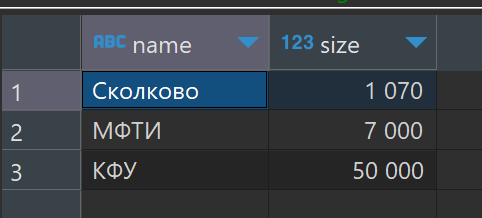
**from** college c

**where** c.id **in** (10, 30, 50)

**order** **by**

c.**size**

,c.**name**;



/\*d. 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** c.**name**

,c.**size**

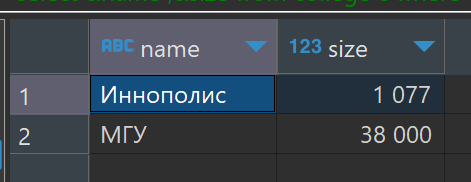
**from** college c

**where** c.id **not** **in** (10, 30, 50)

**order** **by**

c.**size**

,c.**name**;



/\*e. 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** c.**name**

,c.amount\_of\_students

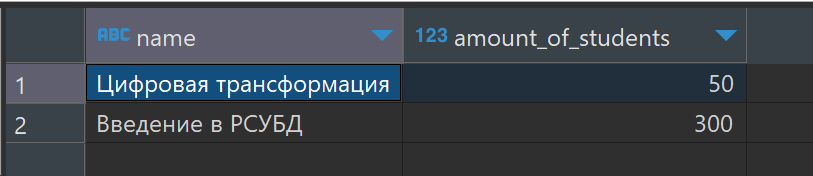
**from** course c

**where** c.amount\_of\_students **between** 27 **and** 310

**order** **by**

c.**name** **desc**

,c.amount\_of\_students **desc**;



/\*f. Напишите SQL запрос который возвращает имена студентов и название курсов университетов в одном списке.

Результат отсортируйте в убывающем порядке.\*/

**select** x.**name**

**from** (

**select** s.**name**

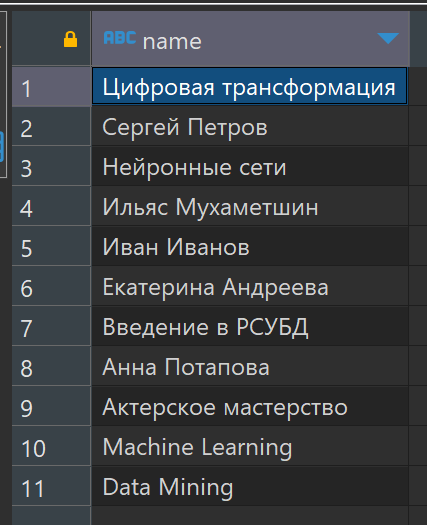
**from** student s

**union** **all**

**select** c.**name**

**from** course c) x

**order** **by** x.**name** **desc**;



/\*g. 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.\*/

**select** x.**name**

,x.object\_type

**from** (

**select** c.**name**, 'университет' object\_type

**from** college c

**union** **all**

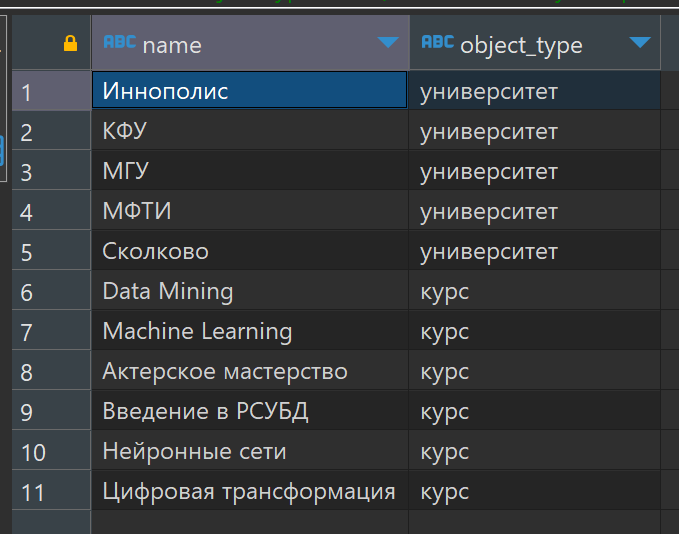
**select** c.**name**, 'курс' object\_type

**from** course c) x

**order** **by**

x.object\_type **desc**

,x.**name**; --из постатновки не понятно по имени по возрастанию или по убыванию, сделал по возрастанию т.к. на примере по возрастанию похоже



/\*h. 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.\*/

**select** c.**name**

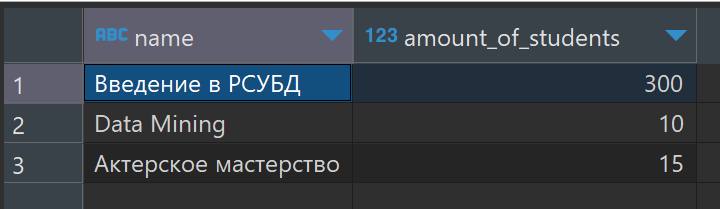
,c.amount\_of\_students

**from** course c

**order** **by**

**case** **when** c.amount\_of\_students = 300 **then** -1 **else** c.amount\_of\_students **end**

**limit** 3;



/\*i. 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 \*/

**insert** **into** course (id, **name**, amount\_of\_students, college\_id)

**select** 60 id

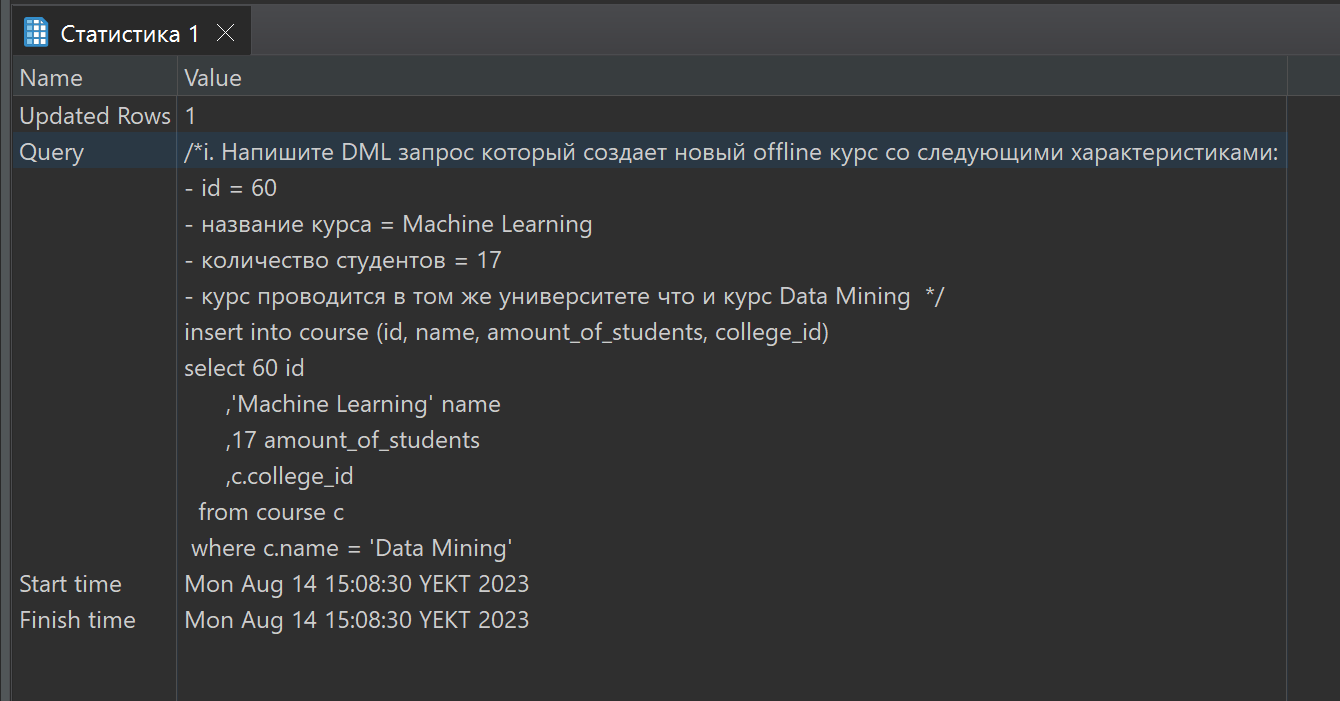
,'Machine Learning' **name**

,17 amount\_of\_students

,c.college\_id

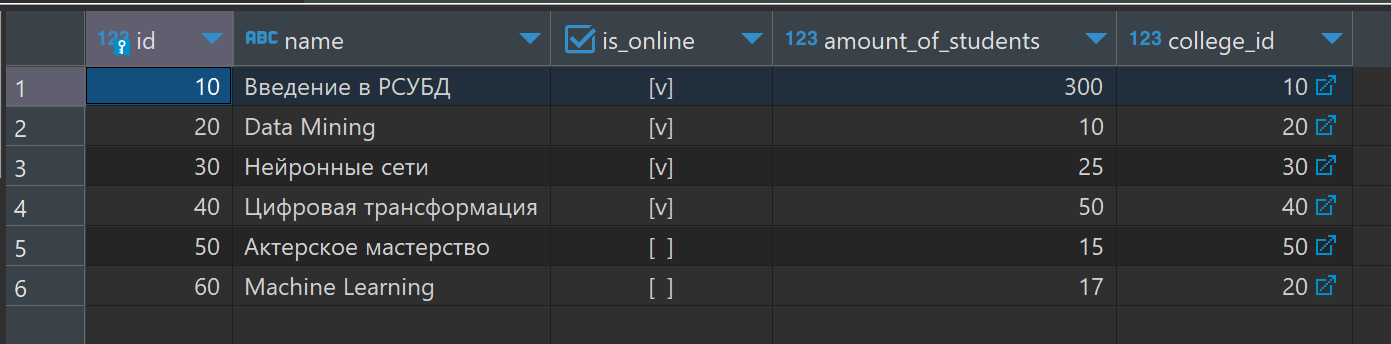
**from** course c

**where** c.**name** = 'Data Mining';



**select** \*

**from** course c ;



/\*j. 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.\*/

**with**

a **as** (**select** c.id **from** course c )

,b **as** (**select** soc.id **from** student\_on\_course soc)

**select** id

**from** (

(**select** id **from** a

**except**

**select** id **from** b)

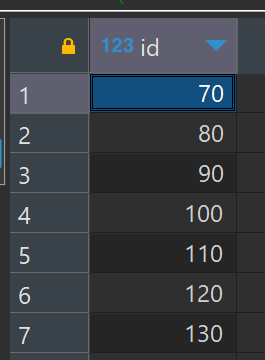
**union**

(**select** id **from** b

**except**

**select** id **from** a)) x

**order** **by** 1;



/\*k. 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.\*/

**select** s.**name** student\_name

,cr.**name** course\_na

,cl.**name** student\_college

,soc.student\_rating

**from** student s

**join** student\_on\_course soc **on**

soc.student\_id = s.id

**join** course cr **on**

cr.id = soc.course\_id

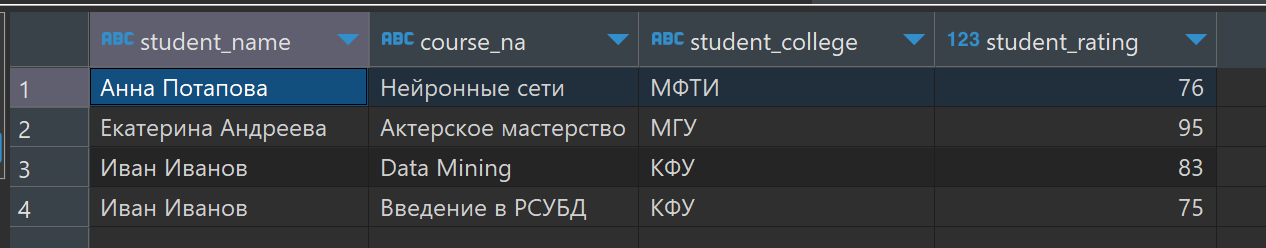
**join** college cl **on**

cl.id = s.college\_id

**where** soc.student\_rating > 50

**and** cl.**size** > 5000

**order** **by** 1, 2;



/\*l. 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.\*/

**select** x.student\_1

,x.student\_2

,x.city

**from** (

**select** s.**name** student\_1

,**first\_value**(s.**name**) **over** (**partition** **by** s.city) student\_2

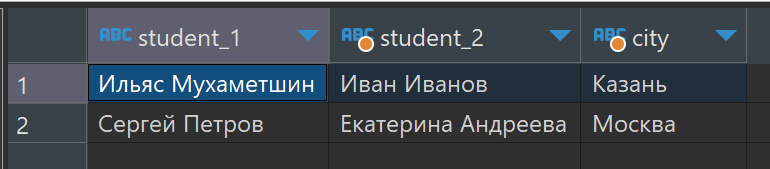
,**row\_number**() **over** (**partition** **by** s.city **order** **by** s.**name**) rn

,s.city

**from** student s) x

**where** x.rn > 1

**order** **by** 1;



/\*m. 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\*/

**select** **case**

**when** soc.student\_rating < 30 **then** 'неудовлетворительно'

**when** soc.student\_rating >= 30 **and** soc.student\_rating < 60 **then** 'удовлетворительно'

**when** soc.student\_rating >= 60 **and** soc.student\_rating < 85 **then** 'хорошо'

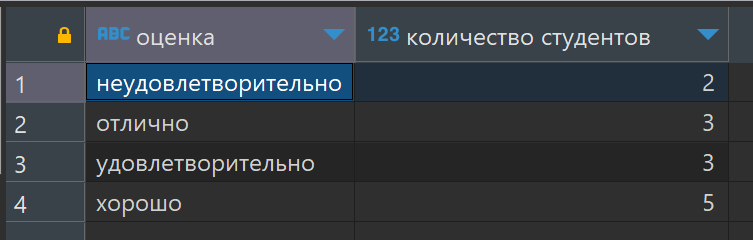
**else** 'отлично' **end** "оценка"

,**count**(1) "количество студентов"

**from** student\_on\_course soc

**group** **by** 1

**order** **by** "оценка";



/\*n. Дополните SQL запрос из задания a), с указанием вывода имени курса и количество оценок внутри курса.

Результат отсортируйте по названию курса и оценки студента.\*/

**select** c.**name** "курс"

,**case**

**when** soc.student\_rating < 30 **then** 'неудовлетворительно'

**when** soc.student\_rating >= 30 **and** soc.student\_rating < 60 **then** 'удовлетворительно'

**when** soc.student\_rating >= 60 **and** soc.student\_rating < 85 **then** 'хорошо'

**else** 'отлично' **end** "оценка"

,**count**(1) "количество студентов"

**from** student\_on\_course soc

**join** course c **on**

c.id = soc.course\_id

**group** **by** 1, 2

**order** **by** "курс", "оценка";

