/\* Напишите SQL запрос который возвращает имена студентов и их аккаунтов в Telegram у которых родной город

\* “Казань” или “Москва”. Результат отсортируйте по имени студента в убывающем порядке

\*/

**select**

**name**,

telegram\_contact

**from**

student

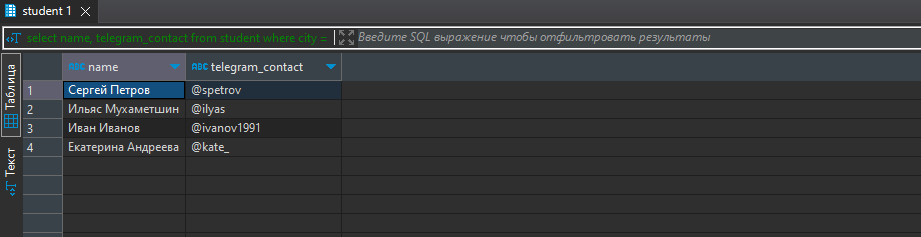
**where**

city = 'Казань'

**or** city = 'Москва'

**order** **by**

**name** **desc**;



/\*Напишите SQL запрос, который возвращает данные по университетам в следующем виде

\* (один столбец с семью данными внутри) с сортировкой по полю "полная информация".

\*/

**select**

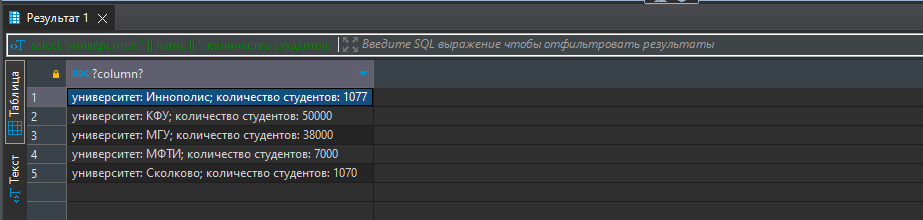
'университет: ' || **name** || '; количество студентов: ' || **size**

**from**

college

**order** **by**

**name**;



/\*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 имени университета.

\*/

**select**

**name** **as** университет,

**size** **as** количество\_студентов

**from**

college

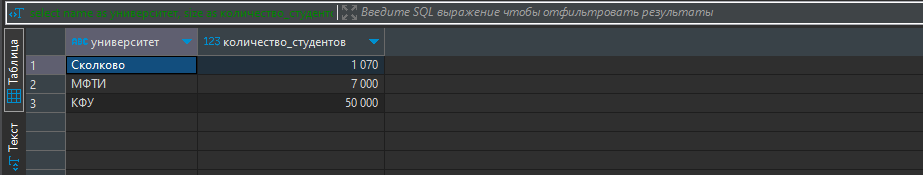
**where**

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

**order** **by**

**size**,

**name**;



/\* Напишите SQL запрос, который возвращает список университетов и количество студентов,

если идентификатор университета НЕ должен соответствовать значениям из списка 10, 30, 50.

Пожалуйста, воспользуйтесь конструкцией NOT IN. Результат запроса отсортируйте по количеству студентов и затем по имени университета.

\*/

**select**

**name** **as** университет,

**size** **as** количество\_студентов

**from**

college

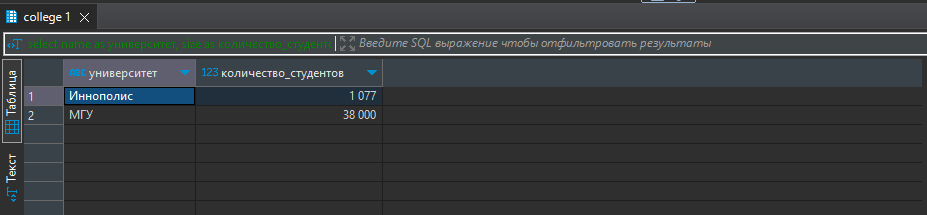
**where**

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

**order** **by**

**size**,

**name**;



/\*

Задача: 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\_курса,

amount\_of\_students **as** количество\_заявленных\_слушателей

**from**

course

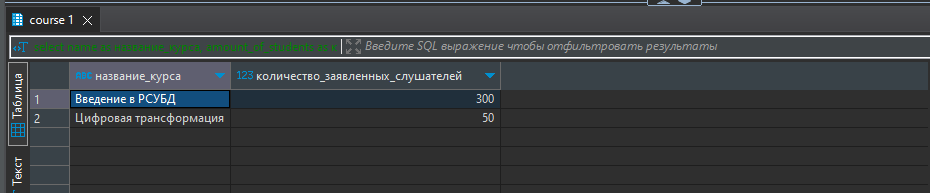
**where**

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

**order** **by**

course,

amount\_of\_students **desc**;



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

Результат отсортируйте в убывающем порядке.

\*/

**select**

**name** **as** студенты\_и\_курсы

**from**

student

**union**

**select**

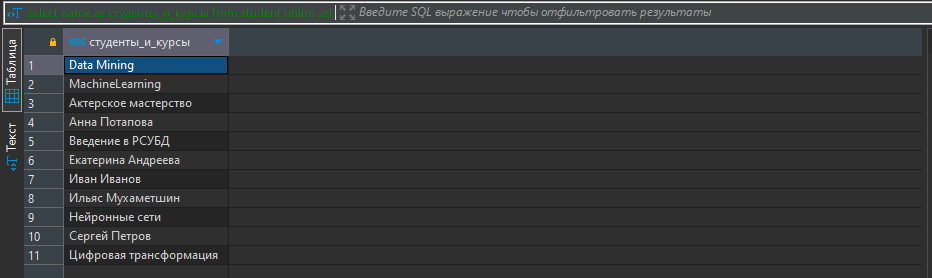
**name**

**from**

course

**order** **by**

студенты\_и\_курсы



/\*

Напишите SQL-запрос, который возвращает имена университетов и количество заявленных слушателей.

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

Результат запроса отсортируйте по названию курса и по количеству заявленных слушателей в убывающем порядке для двух полей.

\*/

**select**

**name**,

'университет' **as** object\_type

**from**

college

**union**

**select**

**name**,

'курс' **as** object\_type

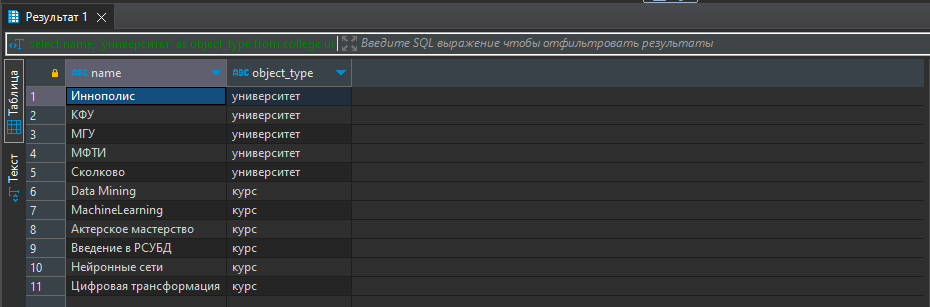
**from**

course

**order** **by**

object\_type **desc**,

**name**;



/\*

Задача: Напишите SQL-запрос, который возвращает названия курсов и количество заявленных студентов

в отсортированном списке по количеству слушателей в возрастающем порядке.

Однако запись с количеством студентов равным 300 должна быть на первом месте.

Ограничьте вывод данных до 3 строк.

\*/

**select**

**name**,

amount\_of\_students

**from**

course

**order** **by**

**case**

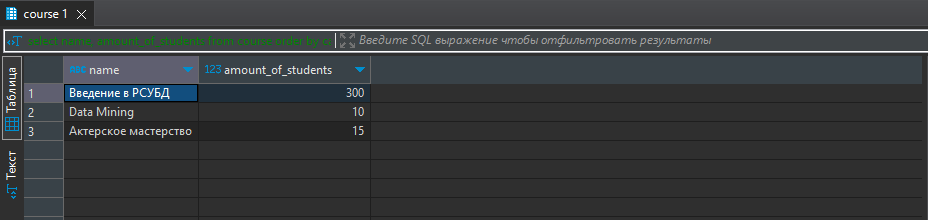
**when** amount\_of\_students = 300 **then** 0

**else** 1

**end**,

amount\_of\_students

**limit** 3;



/\*

Напишите DML запрос, который создает новый offline курс с следующими характеристиками:

- id = 60

- название курса = MachineLearning

- количество студентов = 17

- курс проводится в том же университете, что и курс DataMining

Предоставьте INSERT выражение, которое заполняет необходимую таблицу данными.

\*/

-- Находим, где преподается DataMining

**select**

college\_id

**from**

course

**where**

**name** = 'Data Mining'

-- Создаем новый offline курс

**insert**

**into**

course (id,

**name**,

is\_online,

amount\_of\_students,

college\_id)

**values** (60,

'MachineLearning',

'no',

17,

20)

-- Выводим таблицу для проверки сощдания курса

**select**

id,

**name**,

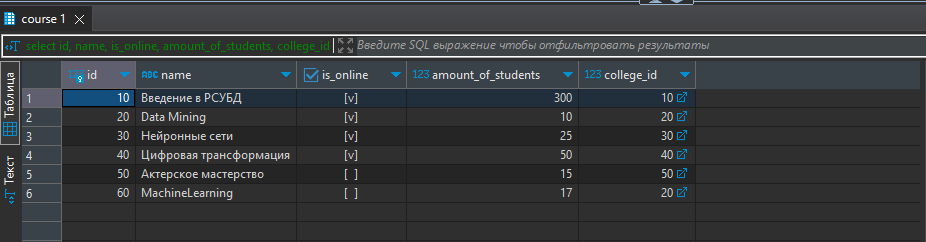
is\_online,

amount\_of\_students,

college\_id

**from**

course



/\*

Напишите SQL скрипт, который подсчитывает симметрическую разницу множеств A и B,

где A - таблица course, B - таблица student\_on\_course,

"\" - это операция разницы множеств, и "⋃" - это операция объединения множеств.

Необходимо подсчитать на основании атрибута id из обеих таблиц.

Результат отсортирован по 1 столбцу.

\*/

(

**select**

id

**from**

course

**except**

**select**

id

**from**

student\_on\_course

)

**union** **all**

(

**select**

id

**from**

student\_on\_course

**except**

**select**

id

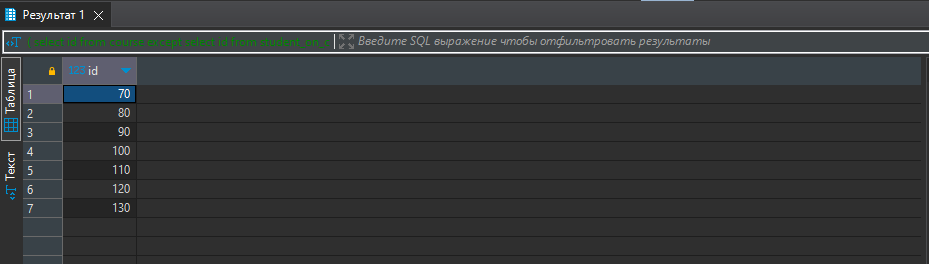
**from**

course

)

**order** **by**

id;



/\*

Напишите SQL-запрос, который возвращает имена студентов, названия курсов,

названия их родных университетов и соответствующий рейтинг курса.

С условием, что рейтинг студента должен быть строго больше (>) 50 баллов

и размер соответствующего университета должен быть строго больше (>) 5000 студентов.

Результат необходимо отсортировать по первым двум столбцам.

\*/

**select**

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

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

st.**name** **as** student\_college,

sco.student\_rating

**from**

student\_on\_course sco

**join** student s **on**

sco.student\_id = s.id

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

sco.course\_id = c.id

**join** college st **on**

s.college\_id = st.id

**where**

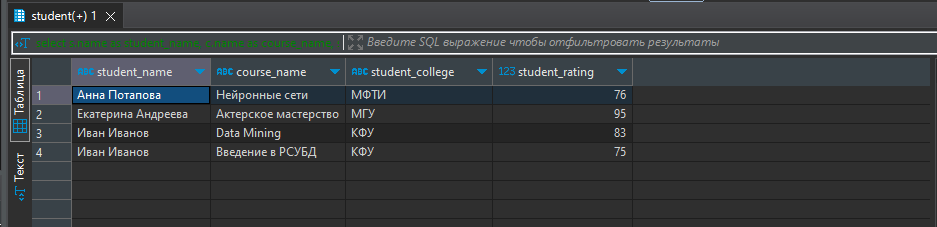
sco.student\_rating > 50

**and** st.**size** > 5000

**order** **by**

student\_name,

course\_name;



/\*Выведите уникальные семантические пары студентов,

\* родной город которых один и тот же. Результат необходимо

\* отсортировать по первому столбцу.

\*/

**with** StudentPairs **as** (

**select**

s1.**name** **as** student\_1,

s2.**name** **as** student\_2,

s1.city

**from**

student s1

**join** student s2 **on**

s1.city = s2.city

**and** s1.**name** < s2.**name**

)

**select**

**distinct**

student\_1,

student\_2,

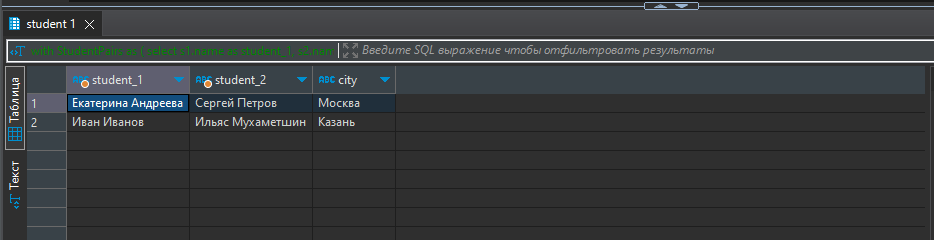
city

**from**

StudentPairs

**order** **by**

student\_1;



/\*

Напишите SQL-запрос, который возвращает количество студентов,

сгруппированных по их оценке. Результат отсортирован 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** student\_rating < 30 **then** 'неудовлетворительно'

**when** student\_rating >= 30

**and** student\_rating < 60 **then** 'удовлетворительно'

**when** student\_rating >= 60

**and** student\_rating < 85 **then** 'хорошо'

**else** 'отлично'

**end** **as** grade,

**COUNT**(\*) **as** student\_count

**from**

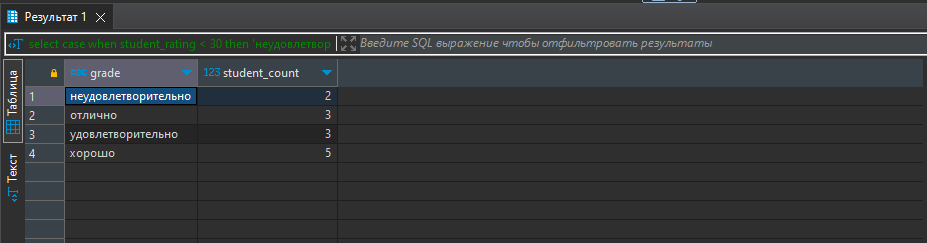
student\_on\_course

**group** **by**

grade

**order** **by**

grade;



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

\* Результат отсортируйте по названию курса и оценке студента.

\*/

**select**

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

s.telegram\_contact **as** telegram\_account,

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

**COUNT**(soc.student\_rating) **as** rating\_count

**from**

Student s

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

s.id = soc.student\_id

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

soc.course\_id = c.id

**where**

s.city **in** ('Казань', 'Москва')

**group** **by**

s.**name**,

s.telegram\_contact,

c.**name**

**order** **by**

c.**name**,

rating\_count;

