**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, Шрифт

Автоматически созданное описание

**Задание 2**

* 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

**Код запроса:**

SELECT s.name, c.name

FROM student s

INNER JOIN course c ON s.college\_id = c.college\_id

ORDER BY s.name DESC;

**Результат работы:**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

* 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

**Код запроса:**

SELECT name, 'университет' AS type

FROM college

UNION ALL

SELECT name, 'курс' AS type

FROM course

ORDER BY type DESC, name DESC;

**Результат работы:**

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

* 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

**Код запроса:**

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;

**Результат работы:**

Изображение выглядит как текст, снимок экрана, Шрифт, чек

Автоматически созданное описание

* 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 (Не используйте вложенные подзапросы!).

Приложите скрин результата запроса к данным курсов после

выполнения команды INSERT к таблице, которая была изменена.

**Код запроса:**

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

VALUES (60, 'Machine Learning', false, 17, 20)

**Результат работы:**

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Автоматически созданное описание

Рис. 1. Результат работы кода вставки

Изображение выглядит как текст, Шрифт, число, чек

Автоматически созданное описание

Рис. 2. Результат работы кода SELECT \*

* 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.

|  |
| --- |
| **id** |
| 70 |
| 80 |
| 90 |
| 100 |
| … |

**Код запроса:**

SELECT id

FROM (

-- (A \ B)

SELECT id FROM course

EXCEPT

SELECT id FROM student\_on\_course

UNION

-- (B \ A)

SELECT id FROM student\_on\_course

EXCEPT

SELECT id FROM course

) AS id

ORDER BY 1;

**Результат работы:**

Изображение выглядит как текст, число, снимок экрана, Шрифт

Автоматически созданное описание

* 1. (\*\* - **задача опциональная**) Напишите SQL запрос который подсчитывает коэффициент Жаккара (близости множеств) между двумя таблицами A (**course**) и B (**student\_on\_course**) по атрибуту id на основании формулы:

Изображение выглядит как Шрифт, белый, текст, символ

Автоматически созданное описание

где A - таблица **course,** B - таблица **student\_on\_course,** “⋃” - объединение множеств, “∩” - пересечение множеств, | A ∩ B | - мощность множества (количество записей) по пересечению, | A ⋃ B | - мощность множества (количество записей) по объединению

**Подсказка**: чтобы подсчитать количество элементов используйте

конструкцию SELECT count(\*) FROM …

Пример результата представлен ниже

|  |
| --- |
| **val** |
| 0.46153846153846153846 |

**Код программы:**

SELECT (CAST(COUNT(\*) AS FLOAT) / (

SELECT COUNT(\*)

FROM (

SELECT id

FROM course

UNION

SELECT id

FROM student\_on\_course

) AS combined\_ids

)) AS коэффициент\_Жаккара

FROM (

SELECT id

FROM course

INTERSECT

SELECT id

FROM student\_on\_course

) AS intersection\_ids;

**Результат работы программы:**

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание