

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Псковский государственный университет»

Передовая инженерная школа гибридных технологий в станкостроении  
Союзного государства

Отделение информационно-коммуникационных технологий

## **Курсовая работа**

по дисциплине «Базы данных»

Выполнил студент:

Гр. 1433-05

Иванов Н.И.

Проверил преподаватель:

Вертешев А.С.

Псков

2024

<b>Задание.....</b>	<b>3</b>
<b>Схема базы данных.....</b>	<b>4</b>
<b>Создание базы данных.....</b>	<b>6</b>
<b>Подключение к БД и создание таблиц.....</b>	<b>7</b>
<b>Заполнение таблиц в БД.....</b>	<b>11</b>
<b>Запросы к БД.....</b>	<b>14</b>
<b>Подключение БД к C# .Net , WindowsForms через CMD.....</b>	<b>32</b>
<b>Ссылка на Github репозиторий.....</b>	<b>36</b>

## Задание

Вариант №6

Тема: учет успеваемости студентов в текущей сессии.

Объекты: факультеты, группы, студенты, экзамены, дисциплины, преподаватели.

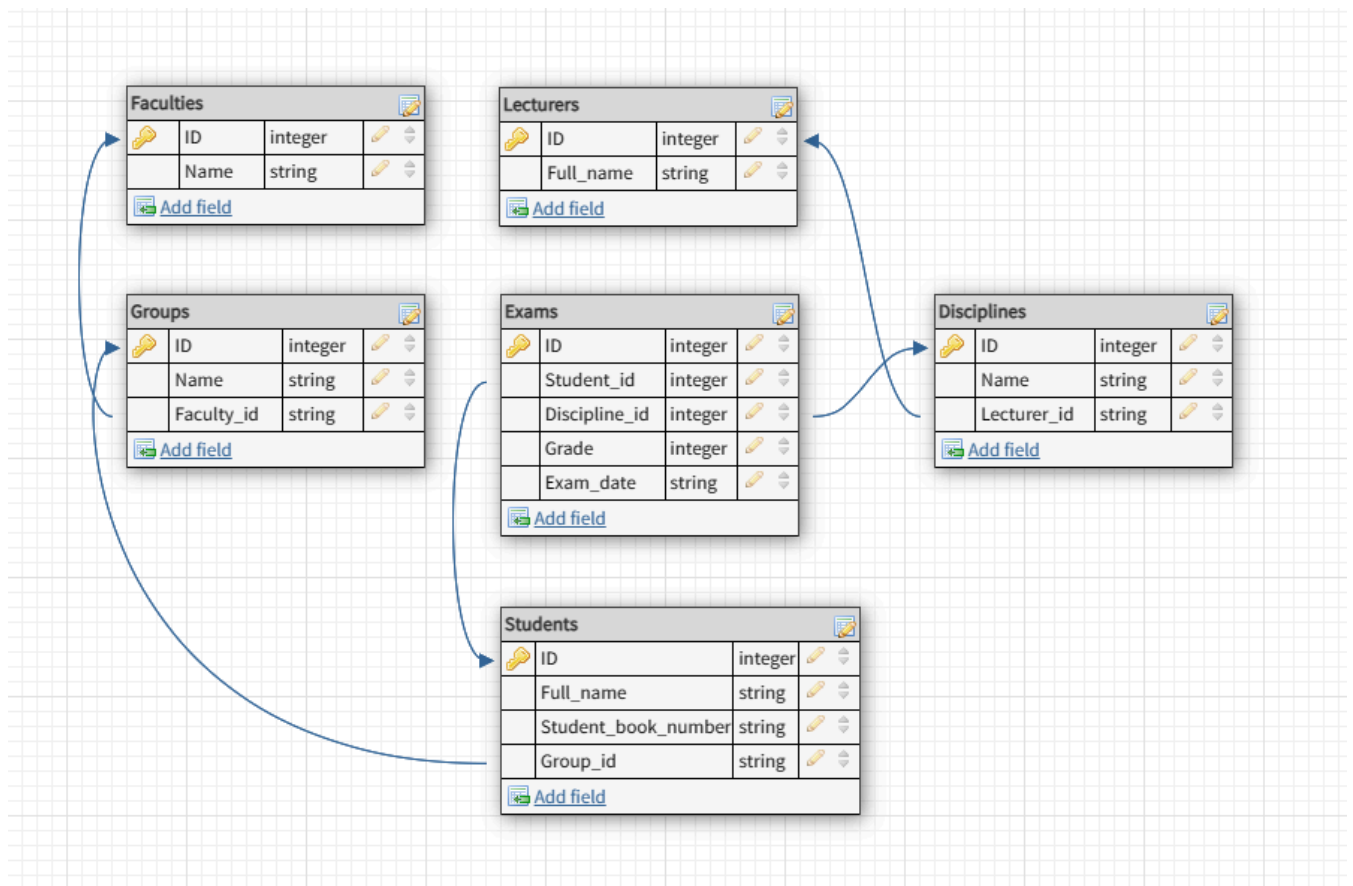
Для моделирования задачи необходимо хранить следующую информацию:

- \* Номер группы, в которой учится студент
- \* Факультет, к которому принадлежит группа
- \* ФИО студента
- \* Номер зачетной книжки студента
- \* Наименование дисциплины, которую изучает группа студентов
- \* ФИО лектора дисциплины
- \* Оценка, полученная студентом за экзамен по каждой дисциплине, изучаемой группой
- \* Дата сдачи экзамена студентом

Дополнительные условия:

- \* если студент пересдавал экзамен, то в таблице экзаменов может быть несколько записей;
- \* один лектор может читать несколько дисциплин.

## Схема базы данных



Описание схемы:

**faculties**: хранит информацию о факультетах.

Один факультет (faculties) может иметь много групп (groups).

Одна группа (groups) принадлежит одному факультету (faculties).

Name - имя факультета.

**groups**: хранит информацию о группах студентов и их принадлежности к факультетам.

Одна группа (groups) может иметь много студентов (students).

Один студент (students) принадлежит одной группе (groups).

Name - имя группы.

Faculty\_id - id факультета (FK).

**students:** хранит информацию о студентах, их ФИО, номере зачетной книжки и группе.

Один студент (students) может иметь много экзаменов (exams).

Один экзамен (exams) принадлежит одному студенту (students).

Full\_name - ФИО студента.

Student\_book\_number - номер зачетной книжки.

Group\_id - id группы (FK).

**lecturers:** хранит информацию о преподавателях.

Один лектор (lecturers) может преподавать много дисциплин (disciplines).

Одна дисциплина (disciplines) преподается одним лектором (lecturers).

Full\_name - ФИО преподавателя.

**disciplines:** хранит информацию о дисциплинах и преподавателях, которые их преподают.

Одна дисциплина (disciplines) может иметь много экзаменов (exams).

Один экзамен (exams) принадлежит одной дисциплине (disciplines).

Name - название дисциплины.

Lecturer\_id - id преподавателя (FK).

**exams:** хранит информацию об экзаменах, включая студента, дисциплину, оценку и дату экзамена.

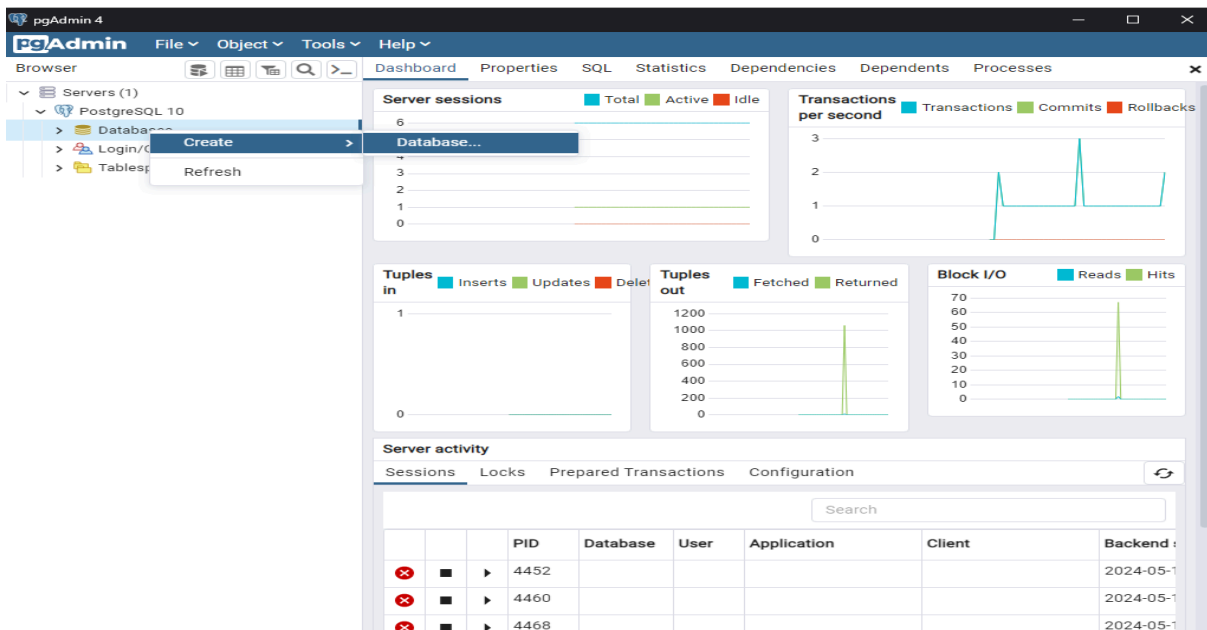
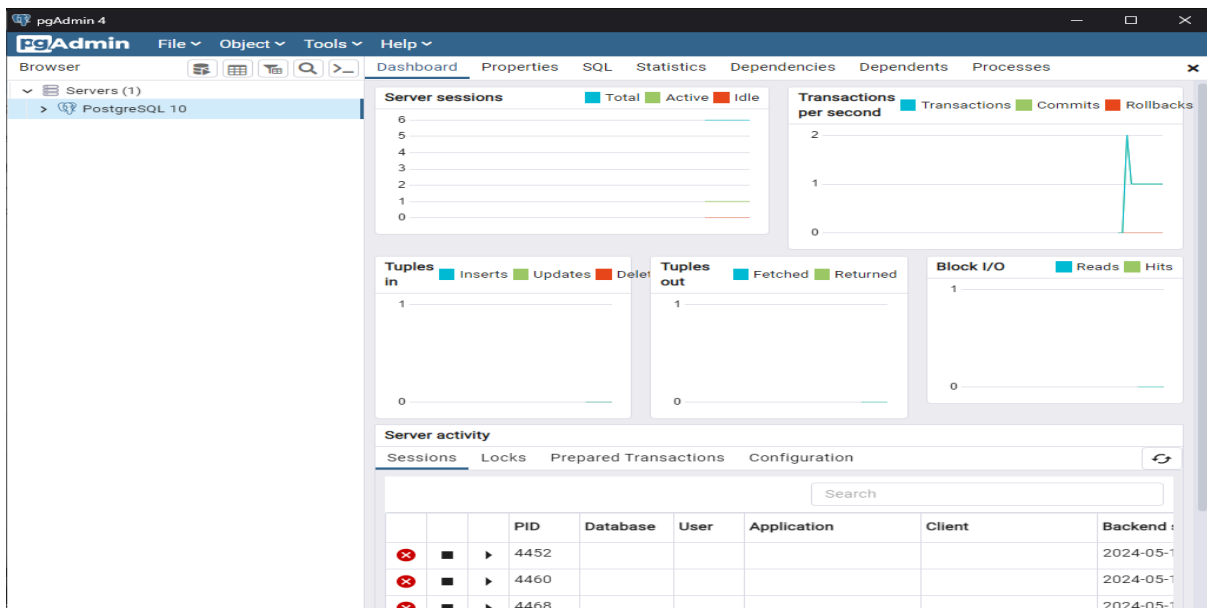
Student\_id - id студента (FK).

Discipline\_id - id дисциплины (FK).

Grade - оценка.

Exam\_date - дата экзамена.

## Создание базы данных



## Подключение к БД и создание таблиц

Подключаемся к базе данных.

```
SQL Shell (psql)
Server [localhost]:
Database [postgres]: DB
Port [5432]:
Username [postgres]:
Пароль пользователя postgres:
psql (10.23)
ПРЕДУПРЕЖДЕНИЕ: Кодовая страница консоли (866) отличается от основной
                  страницы Windows (1251).
                  8-битовые (русские) символы могут отображаться некорректно.
                  Подробнее об этом смотрите документацию psql, раздел
                  "Notes for Windows users".
Введите "help", чтобы получить справку.
DB=#
```

### 1. Создание таблицы faculties

```
CREATE TABLE faculties (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL);
```

```
Введите "help", чтобы получить справку.
DB=# CREATE TABLE faculties (
DB(#      id SERIAL PRIMARY KEY,
DB(#      name VARCHAR(100) NOT NULL
DB(# );
CREATE TABLE
```

```
DB=# SELECT * FROM faculties;
 id | name
----+-----
(0 строк)
```

### 2. Создание таблицы groups

```
CREATE TABLE groups (
    id SERIAL PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    faculty_id INTEGER REFERENCES faculties(id));
```

```

DB=# CREATE TABLE groups (
DB(#      id SERIAL PRIMARY KEY,
DB(#      name VARCHAR(50) NOT NULL,
DB(#      faculty_id INTEGER REFERENCES faculties(id)
DB(# );
CREATE TABLE
DB=# SELECT * FROM groups;
 id | name | faculty_id
----+-----+-----
(0 строк)

```

### 3. Создание таблицы students

```

CREATE TABLE students (
    id SERIAL PRIMARY KEY,
    full_name VARCHAR(150) NOT NULL,
    student_book_number VARCHAR(50) UNIQUE NOT NULL,
    group_id INTEGER REFERENCES groups(id));

```

```

DB=# CREATE TABLE students (
DB(#      id SERIAL PRIMARY KEY,
DB(#      full_name VARCHAR(150) NOT NULL,
DB(#      student_book_number VARCHAR(50) UNIQUE NOT NULL,
DB(#      group_id INTEGER REFERENCES groups(id)
DB(# );
CREATE TABLE
DB=# SELECT * FROM students;
 id | full_name | student_book_number | group_id
----+-----+-----+-----
(0 строк)

DB=#

```

### 4. Создание таблицы lecturers

```

CREATE TABLE lecturers (
    id SERIAL PRIMARY KEY,
    full_name VARCHAR(150) NOT NULL);

```



```
DB=# CREATE TABLE lecturers (
DB(#      id SERIAL PRIMARY KEY,
DB(#      full_name VARCHAR(150) NOT NULL
DB(# );
CREATE TABLE
DB=# SELECT * FROM lecturers;
 id | full_name
-----+-----
(0 строк)
```

## 5. Создание таблицы disciplines

```
CREATE TABLE disciplines (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    lecturer_id INTEGER REFERENCES lecturers(id));
```

```
DB=# CREATE TABLE disciplines (
DB(#      id SERIAL PRIMARY KEY,
DB(#      name VARCHAR(100) NOT NULL,
DB(#      lecturer_id INTEGER REFERENCES lecturers(id)
DB(# );
CREATE TABLE
DB=# SELECT * FROM disciplines;
 id | name | lecturer_id
----+-----+-----
(0 строк)
```

## 6. Создание таблицы exams

```
CREATE TABLE exams (
    id SERIAL PRIMARY KEY,
    student_id INTEGER REFERENCES students(id),
    discipline_id INTEGER REFERENCES disciplines(id),
    grade INTEGER CHECK (grade >= 0 AND grade <= 100),
    exam_date DATE NOT NULL);
```

```

DB=# CREATE TABLE exams (
DB(#      id SERIAL PRIMARY KEY,
DB(#      student_id INTEGER REFERENCES students(id),
DB(#      discipline_id INTEGER REFERENCES disciplines(id),
DB(#      grade INTEGER CHECK (grade >= 0 AND grade <= 100),
DB(#      exam_date DATE NOT NULL
DB(# );
CREATE TABLE
DB=# SELECT * FROM exams;
 id | student_id | discipline_id | grade | exam_date
-----+-----+-----+-----+-----
(0 строк)

```

## Заполнение таблиц в БД

INSERT INTO faculties (name) VALUES

('Факультет информатики'),

('Факультет математики'),

('Факультет физики');

```
DB=# INSERT INTO faculties (name) VALUES
DB-# ('Факультет информатики'),
DB-# ('Факультет математики'),
DB-# ('Факультет физики');
INSERT 0 3
DB=# SELECT * FROM faculties;
 id |      name
----+-----
  1 | Факультет информатики
  2 | Факультет математики
  3 | Факультет физики
(3 строки)
```

INSERT INTO groups (name, faculty\_id) VALUES

('Группа 101', 1),

('Группа 102', 1),

('Группа 201', 2);

```
DB=# INSERT INTO groups (name, faculty_id) VALUES
DB-# ('Группа 101', 1),
DB-# ('Группа 102', 1),
DB-# ('Группа 201', 2);
INSERT 0 3
DB=# SELECT * FROM groups;
 id |   name   | faculty_id
----+-----+-----
  1 | Группа 101 |         1
  2 | Группа 102 |         1
  3 | Группа 201 |         2
(3 строки)
```

INSERT INTO students (full\_name, student\_book\_number, group\_id) VALUES

('Иванов Иван Иванович', '123456', 1),

('Петров Петр Петрович', '654321', 1),

('Сидоров Сидор Сидорович', '112233', 2);

```
DB=# INSERT INTO students (full_name, student_book_number, group_id) VALUES
DB-# ('Иванов Иван Иванович', '123456', 1),
DB-# ('Петров Петр Петрович', '654321', 1),
DB-# ('Сидоров Сидор Сидорович', '112233', 2);
INSERT 0 3
DB=# SELECT * FROM students;
 id |      full_name      | student_book_number | group_id
-----+-----+-----+-----
  1 | Иванов Иван Иванович |         123456      |        1
  2 | Петров Петр Петрович |         654321      |        1
  3 | Сидоров Сидор Сидорович |       112233      |        2
(3 строки)
```

INSERT INTO lecturers (full\_name) VALUES

('Александров Александр Александрович'),

('Борисов Борис Борисович'),

('Викторов Виктор Викторович');

```
DB=# INSERT INTO lecturers (full_name) VALUES
DB-# ('Александров Александр Александрович'),
DB-# ('Борисов Борис Борисович'),
DB-# ('Викторов Виктор Викторович');
INSERT 0 3
DB=# SELECT * FROM lecturers;
 id |      full_name
-----+-----
  1 | Александров Александр Александрович
  2 | Борисов Борис Борисович
  3 | Викторов Виктор Викторович
(3 строки)

DB=#
```

INSERT INTO disciplines (name, lecturer\_id) VALUES

('Программирование', 1),

('Математика', 2),

('Физика', 3);

```
DB=# INSERT INTO disciplines (name, lecturer_id) VALUES
DB-# ('Программирование', 1),
DB-# ('Математика', 2),
DB-# ('Физика', 3);
INSERT 0 3
DB=# SELECT * FROM disciplines;
 id |      name      | lecturer_id
-----+-----+-----
  1 | Программирование |          1
  2 | Математика      |          2
  3 | Физика          |          3
(3 строки)
```

```
INSERT INTO exams (student_id, discipline_id, grade, exam_date) VALUES
(1, 1, 85, '2024-01-15'),
(1, 2, 90, '2024-01-20'),
(2, 1, 75, '2024-01-15'),
(2, 3, 80, '2024-01-25'),
(3, 2, 95, '2024-01-20');
```

```
DB=# INSERT INTO exams (student_id, discipline_id, grade, exam_date) VALUES
DB-# (1, 1, 85, '2024-01-15'),
DB-# (1, 2, 90, '2024-01-20'),
DB-# (2, 1, 75, '2024-01-15'),
DB-# (2, 3, 80, '2024-01-25'),
DB-# (3, 2, 95, '2024-01-20');
INSERT 0 5
DB=# SELECT * FROM exams;
 id | student_id | discipline_id | grade | exam_date
-----+-----+-----+-----+-----
  1 |          1 |             1 |     85 | 2024-01-15
  2 |          1 |             2 |     90 | 2024-01-20
  3 |          2 |             1 |     75 | 2024-01-15
  4 |          2 |             3 |     80 | 2024-01-25
  5 |          3 |             2 |     95 | 2024-01-20
(5 строк)
```

## Запросы к БД

### Три запроса на выборку с использованием join:

1. Выборка студентов с указанием их факультетов и групп

```
SELECT
    students.full_name AS student_name,
    groups.name AS group_name,
    faculties.name AS faculty_name
FROM
    students
JOIN
    groups ON students.group_id = groups.id
JOIN
    faculties ON groups.faculty_id = faculties.id;
```

```
DB=# SELECT
DB-#     students.full_name AS student_name,
DB-#     groups.name AS group_name,
DB-#     faculties.name AS faculty_name
DB-# FROM
DB-#     students
DB-# JOIN
DB-#     groups ON students.group_id = groups.id
DB-# JOIN
DB-#     faculties ON groups.faculty_id = faculties.id;
 student_name | group_name | faculty_name
-----+-----+-----
Петров Петр Петрович | Группа 101 | Факультет информатики
Иванов Иван Иванович | Группа 101 | Факультет информатики
Сидоров Сидор Сидорович | Группа 102 | Факультет информатики
(3 строки)
```

2. Выборка оценок студентов по дисциплинам и лекторам

```
SELECT
    students.full_name AS student_name,
```

```

disciplines.name AS discipline_name,
lecturers.full_name AS lecturer_name,
exams.grade,
exams.exam_date
FROM
    exams
JOIN
    students ON exams.student_id = students.id
JOIN
    disciplines ON exams.discipline_id = disciplines.id
JOIN
    lecturers ON disciplines.lecturer_id = lecturers.id;

```

```

C:\Администратор: C:\Windows\system32\cmd.exe - psql -h localhost -p 5432 -U postgres -d DB
DB-# disciplines.name AS discipline_name,
DB-# lecturers.full_name AS lecturer_name,
DB-# exams.grade,
DB-# exams.exam_date
DB-# FROM
DB-# exams
DB-# JOIN
DB-# students ON exams.student_id = students.id
DB-# JOIN
DB-# disciplines ON exams.discipline_id = disciplines.id
DB-# JOIN
DB-# lecturers ON disciplines.lecturer_id = lecturers.id;

```

student_name	discipline_name	lecturer_name	grade	exam_date
Иванов Иван Иванович	Программирование	Александров Александр Александрович	85	2024-01-15
Иванов Иван Иванович	Математика	Борисов Борис Борисович	90	2024-01-20
Петров Петр Петрович	Программирование	Александров Александр Александрович	75	2024-01-15
Петров Петр Петрович	Физика	Викторов Виктор Викторович	80	2024-01-25
Сидоров Сидор Сидорович	Математика	Борисов Борис Борисович	95	2024-01-20

(5 строк)

### 3. Выборка студентов, сдававших экзамены по конкретному факультету и дисциплине

```

SELECT
    students.full_name AS student_name,
    groups.name AS group_name,
    faculties.name AS faculty_name,
    disciplines.name AS discipline_name,

```

```

    exams.grade,
    exams.exam_date
FROM
    exams
JOIN
    students ON exams.student_id = students.id
JOIN
    groups ON students.group_id = groups.id
JOIN
    faculties ON groups.faculty_id = faculties.id
JOIN
    disciplines ON exams.discipline_id = disciplines.id
WHERE
    faculties.name = 'Факультет информатики'
    AND disciplines.name = 'Программирование';

```

```

Администратор: C:\Windows\system32\cmd.exe - psql -h localhost -p 5432 -U postgres -d DB
DB-# groups.name AS group_name,
DB-# faculties.name AS faculty_name,
DB-# disciplines.name AS discipline_name,
DB-# exams.grade,
DB-# exams.exam_date
DB-# FROM
DB-# exams
DB-# JOIN
DB-# students ON exams.student_id = students.id
DB-# JOIN
DB-# groups ON students.group_id = groups.id
DB-# JOIN
DB-# faculties ON groups.faculty_id = faculties.id
DB-# JOIN
DB-# disciplines ON exams.discipline_id = disciplines.id
DB-# WHERE
DB-# faculties.name = 'Факультет информатики'
DB-# AND disciplines.name = 'Программирование';
  student_name | group_name | faculty_name | discipline_name | grade | exam_date
-----+-----+-----+-----+-----+-----
Иванов Иван Иванович | Группа 101 | Факультет информатики | Программирование | 85 | 2024-01-15
Петров Петр Петрович | Группа 101 | Факультет информатики | Программирование | 75 | 2024-01-15
(2 строки)

```



**Запросы на объединение, пересечение, разность, произведение, проекцию, деление:**

### 1. Объединение

Запрос на объединение ФИО всех преподавателей и студентов

```
SELECT full_name
```

```
FROM students
```

```
UNION
```

```
SELECT full_name
```

```
FROM lecturers;
```

```
full_name
-----
Сидоров Сидор Сидорович
Иванов Иван Иванович
Петров Петр Петрович
Александров Александр Александрович
Викторов Виктор Викторович
Борисов Борис Борисович
(6 строк)
```

### 2. Пересечение

Запрос на поиск имен, которые есть среди студентов и преподавателей

```
SELECT full_name
```

```
FROM students
```

```
INTERSECT
```

```
SELECT full_name
```

```
FROM lecturers;
```

```
full_name
-----
(0 строк)
```

### 3. Разность

Запрос на поиск студентов, которые не являются преподавателями

```
SELECT full_name
```

```
FROM students
```

```
EXCEPT
```

```
SELECT full_name
FROM lecturers;
```

```

      full_name
-----
Сидоров Сидор Сидорович
Иванов Иван Иванович
Петров Петр Петрович
(3 строки)
```

#### 4. Произведение

Запрос на список всех возможных комбинаций студентов и дисциплин

```
SELECT  students.full_name  AS  student_name,  disciplines.name  AS
discipline_name
FROM students
CROSS JOIN disciplines;
```

```

      student_name      | discipline_name
-----+-----
Иванов Иван Иванович   | Программирование
Петров Петр Петрович    | Программирование
Сидоров Сидор Сидорович | Программирование
Иванов Иван Иванович   | Математика
Петров Петр Петрович    | Математика
Сидоров Сидор Сидорович | Математика
Иванов Иван Иванович   | Физика
Петров Петр Петрович    | Физика
Сидоров Сидор Сидорович | Физика
(9 строк)
```

#### 5. Проекцию

Запрос на список уникальных факультетов, на которых учатся студенты

```
SELECT DISTINCT faculties.name
FROM students
JOIN groups ON students.group_id = groups.id
JOIN faculties ON groups.faculty_id = faculties.id;
```

```

DB=# SELECT DISTINCT faculties.name
      name
-----
Факультет информатики
(1 строка)
```

#### 6. Деление

Запрос на поиск студентов, которые сдали все экзамены по всем дисциплинам

```
SELECT student_id
FROM exams
GROUP BY student_id
HAVING COUNT(DISTINCT discipline_id) = (SELECT COUNT(*) FROM disciplines);
```

```
student_id
-----
(0 строк)
```

### Запросы на соединение: left join, right join, full outer join:

#### 1. Left join

```
SELECT
    students.full_name AS student_name,
    exams.grade
FROM
    students
LEFT JOIN
    exams ON students.id = exams.student_id;
```

student_name	grade
Иванов Иван Иванович	85
Иванов Иван Иванович	90
Петров Петр Петрович	75
Петров Петр Петрович	80
Сидоров Сидор Сидорович	95

(5 строк)

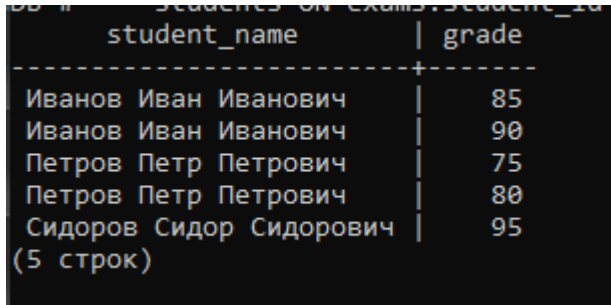
#### 2. Right join

```
SELECT
    students.full_name AS student_name,
    exams.grade
FROM
```

exams

RIGHT JOIN

students ON exams.student\_id = students.id;



student_name	grade
Иванов Иван Иванович	85
Иванов Иван Иванович	90
Петров Петр Петрович	75
Петров Петр Петрович	80
Сидоров Сидор Сидорович	95

(5 строк)

### 3. Full outer join

SELECT

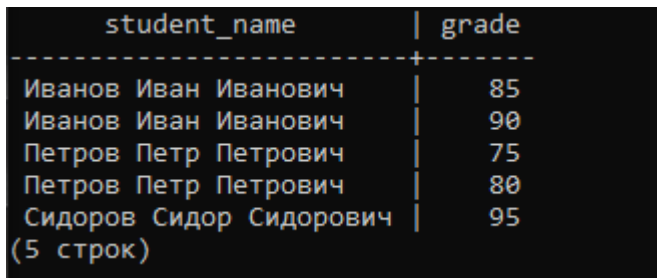
students.full\_name AS student\_name,  
exams.grade

FROM

students

FULL OUTER JOIN

exams ON students.id = exams.student\_id;



student_name	grade
Иванов Иван Иванович	85
Иванов Иван Иванович	90
Петров Петр Петрович	75
Петров Петр Петрович	80
Сидоров Сидор Сидорович	95

(5 строк)

## 2 подзапроса: простой, сложный:

### 1. Простой

SELECT

students.full\_name AS student\_name,

(

SELECT AVG(grade)

FROM exams

```

        WHERE exams.student_id = students.id
    ) AS average_grade
FROM
    students;

```

student_name	average_grade
Иванов Иван Иванович	87.5000000000000000
Петров Петр Петрович	77.5000000000000000
Сидоров Сидор Сидорович	95.0000000000000000
(3 строки)	

## 2. Сложный

```

SELECT
    students.full_name AS student_name,
    (
        SELECT AVG(grade)
        FROM exams
        WHERE exams.student_id = students.id
    ) AS average_grade
FROM
    students
WHERE
    (
        SELECT AVG(grade)
        FROM exams
        WHERE exams.student_id = students.id
    ) > (
        SELECT AVG(grade)
        FROM exams
    );

```

student_name	average_grade
Иванов Иван Иванович	87.50000000000000
Сидоров Сидор Сидорович	95.00000000000000

(2 строки)

**Запросы с различными уровнями вложенности подзапросов в частях select, From where, having:**

## 1. SELECT

SELECT

students.full\_name AS student\_name,

(

SELECT COUNT(\*)

FROM exams

WHERE exams.student\_id = students.id

) AS exam\_count

FROM

students;

student_name	exam_count
Иванов Иван Иванович	2
Петров Петр Петрович	2
Сидоров Сидор Сидорович	1

(3 строки)

## 2. FROM

SELECT

AVG(subquery.grade) AS average\_grade

FROM

(

SELECT grade

FROM exams

) AS subquery;

```

average_grade
-----
85.0000000000000000
(1 строка)

```

### 3. WHERE

```

SELECT
    students.full_name AS student_name
FROM
    students
WHERE
    students.id IN (
        SELECT student_id
        FROM exams
        WHERE grade >= 80
    );

```

```

student_name
-----
Иванов Иван Иванович
Петров Петр Петрович
Сидоров Сидор Сидорович
(3 строки)

```

### 4. HAVING

```

SELECT
    students.group_id,
    AVG(exams.grade) AS average_grade
FROM
    students
JOIN
    exams ON students.id = exams.student_id
GROUP BY
    students.group_id
HAVING

```

```

AVG(exams.grade) > (
    SELECT AVG(grade)
    FROM exams
);

```

group_id	average_grade
2	95.0000000000000000
(1 строка)	

### **Создание ролей пользователей user, guest:**

1. user

```
CREATE ROLE test_user LOGIN PASSWORD 'P@ssw0rd';
```

2. guest

```
CREATE ROLE test_guest LOGIN PASSWORD 'P@ssw0rd';
```

### **Создание двух транзакций с тремя точками восстановления на удаление, создание транзакции на вставку:**

1. Транзакции с тремя точками восстановления на удаление

```

BEGIN;
SAVEPOINT sp1;
DELETE FROM exams WHERE student_id = 1;
SAVEPOINT sp2;
DELETE FROM students WHERE id = 1;
SAVEPOINT sp3;
DELETE FROM exams WHERE student_id = 2;
ROLLBACK TO SAVEPOINT sp1;
COMMIT;

```



## SAVEPOINT 1

```
Администратор: C:\Windows\system32\cmd.exe - psql -U postgres DB
DB=# BEGIN;
BEGIN
DB=# SAVEPOINT sp1;
SAVEPOINT
DB=# SELECT * FROM exams;
 id | student_id | discipline_id | grade | exam_date
-----+-----+-----+-----+-----
  1 |          1 |             1 |    85 | 2024-01-15
  2 |          1 |             2 |    90 | 2024-01-20
  3 |          2 |             1 |    75 | 2024-01-15
  4 |          2 |             3 |    80 | 2024-01-25
  5 |          3 |             2 |    95 | 2024-01-20
(5 строк)

DB=# DELETE FROM exams WHERE student_id = 1;
DELETE 2
DB=# SELECT * FROM exams;
 id | student_id | discipline_id | grade | exam_date
-----+-----+-----+-----+-----
  3 |          2 |             1 |    75 | 2024-01-15
  4 |          2 |             3 |    80 | 2024-01-25
  5 |          3 |             2 |    95 | 2024-01-20
(3 строки)
```

## SAVEPOINT 2

```
DB=# SAVEPOINT sp2;
SAVEPOINT
DB=# SELECT * FROM students;
 id | full_name | student_book_number | group_id
-----+-----+-----+-----
  1 | Иванов Иван Иванович | 123456 | 1
  2 | Петров Петр Петрович | 654321 | 1
  3 | Сидоров Сидор Сидорович | 112233 | 2
(3 строки)

DB=# DELETE FROM students WHERE id = 1;
DELETE 1
DB=# SELECT * FROM students;
 id | full_name | student_book_number | group_id
-----+-----+-----+-----
  2 | Петров Петр Петрович | 654321 | 1
  3 | Сидоров Сидор Сидорович | 112233 | 2
(2 строки)
```

### SAVEPOINT 3

```
DB=# SAVEPOINT sp3;
SAVEPOINT
DB=# SELECT * FROM exams;
 id | student_id | discipline_id | grade | exam_date
-----+-----+-----+-----+-----
  3 |          2 |             1 |    75 | 2024-01-15
  4 |          2 |             3 |    80 | 2024-01-25
  5 |          3 |             2 |    95 | 2024-01-20
(3 строки)

DB=# DELETE FROM exams WHERE student_id = 2;
DELETE 2
DB=# SELECT * FROM exams;
 id | student_id | discipline_id | grade | exam_date
-----+-----+-----+-----+-----
  5 |          3 |             2 |    95 | 2024-01-20
(1 строка)
```

Возвращение к 1 точке сохранения

```
DB=# ROLLBACK TO SAVEPOINT sp1;
ROLLBACK
DB=#
```

BEGIN;

SAVEPOINT sp1;

DELETE FROM exams WHERE student\_id = 3;

SAVEPOINT sp2;

DELETE FROM students WHERE id = 3;

SAVEPOINT sp3;

DELETE FROM groups WHERE id = 3;

ROLLBACK TO SAVEPOINT sp1;

COMMIT;

## SAVEPOINT 1

```
DB=# BEGIN;
BEGIN
DB=# SAVEPOINT sp1;
SAVEPOINT
DB=# SELECT * FROM exams;
 id | student_id | discipline_id | grade | exam_date
-----+-----+-----+-----+-----
  1 |          1 |             1 |     85 | 2024-01-15
  2 |          1 |             2 |     90 | 2024-01-20
  3 |          2 |             1 |     75 | 2024-01-15
  4 |          2 |             3 |     80 | 2024-01-25
  5 |          3 |             2 |     95 | 2024-01-20
(5 строк)

DB=# DELETE FROM exams WHERE student_id = 3;
DELETE 1
DB=# SELECT * FROM exams;
 id | student_id | discipline_id | grade | exam_date
-----+-----+-----+-----+-----
  1 |          1 |             1 |     85 | 2024-01-15
  2 |          1 |             2 |     90 | 2024-01-20
  3 |          2 |             1 |     75 | 2024-01-15
  4 |          2 |             3 |     80 | 2024-01-25
(4 строки)
```

## SAVEPOINT 2

```
C:\ Администратор: C:\Windows\system32\cmd.exe - psql -U postgres DB

DB=# SAVEPOINT sp2;
SAVEPOINT
DB=# SELECT * FROM students;
 id | full_name | student_book_number | group_id
-----+-----+-----+-----
  1 | Иванов Иван Иванович | 123456 | 1
  2 | Петров Петр Петрович | 654321 | 1
  3 | Сидоров Сидор Сидорович | 112233 | 2
(3 строки)

DB=# DELETE FROM students WHERE id = 3;
DELETE 1
DB=# SELECT * FROM students;
 id | full_name | student_book_number | group_id
-----+-----+-----+-----
  1 | Иванов Иван Иванович | 123456 | 1
  2 | Петров Петр Петрович | 654321 | 1
(2 строки)

DB=#
```

### SAVEPOINT 3

```
DB=# SAVEPOINT sp3;
SAVEPOINT
DB=# SELECT * FROM groups;
 id |      name      | faculty_id
-----+-----+-----
  1 | Группа 101     |          1
  2 | Группа 102     |          1
  3 | Группа 201     |          2
(3 строки)

DB=# DELETE FROM groups WHERE id = 3;
DELETE 1
DB=# SELECT * FROM groups;
 id |      name      | faculty_id
-----+-----+-----
  1 | Группа 101     |          1
  2 | Группа 102     |          1
(2 строки)

DB=#
```

Возвращение к 1 точке сохранения

```
DB=# ROLLBACK TO SAVEPOINT sp1;
ROLLBACK
DB=#
```

### 2. Транзакция на вставку

INSERT INTO faculties (name) VALUES ('Факультет биологии');

```
Администратор: C:\windows\system32\cmd.exe - psql -d postgres DB
DB=# BEGIN;
BEGIN
DB=# SELECT * FROM faculties;
 id |      name
-----+-----
  1 | Факультет информатики
  2 | Факультет математики
  3 | Факультет физики
(3 строки)

DB=# INSERT INTO faculties (name) VALUES ('Факультет биологии');
INSERT 0 1
DB=# SELECT * FROM faculties;
 id |      name
-----+-----
  1 | Факультет информатики
  2 | Факультет математики
  3 | Факультет физики
  4 | Факультет биологии
(4 строки)
```

INSERT INTO groups (name, faculty\_id) VALUES ('Группа 301', (SELECT id FROM faculties WHERE name = 'Факультет биологии'));

```

Администратор: C:\Windows\system32\cmd.exe - psql -U postgres DB
DB=#
DB=#
DB=# SELECT * FROM groups;
 id |   name   | faculty_id
-----+-----+-----
  1 | Группа 101 |         1
  2 | Группа 102 |         1
  3 | Группа 201 |         2
(3 строки)

DB=# INSERT INTO groups (name, faculty_id) VALUES ('Группа 301', (SELECT id FROM faculties WHERE name = 'Факультет биологии'));
INSERT 0 1
DB=# SELECT * FROM groups;
 id |   name   | faculty_id
-----+-----+-----
  1 | Группа 101 |         1
  2 | Группа 102 |         1
  3 | Группа 201 |         2
  4 | Группа 301 |         4
(4 строки)

DB=#

```

INSERT INTO students (full\_name, student\_book\_number, group\_id) VALUES ('Иванова Мария Сергеевна', '789012', (SELECT id FROM groups WHERE name = 'Группа 301'));

```

(4 строки)

DB=# SELECT * FROM students;
 id |      full_name      | student_book_number | group_id
-----+-----+-----+-----
  1 | Иванов Иван Иванович |         123456      |         1
  2 | Петров Петр Петрович |         654321      |         1
  3 | Сидоров Сидор Сидорович |         112233      |         2
(3 строки)

DB=# INSERT INTO students (full_name, student_book_number, group_id) VALUES ('Иванова Мария Сергеевна', '789012', (SELECT id FROM groups WHERE name = 'Группа 301'));
INSERT 0 1
DB=# SELECT * FROM students;
 id |      full_name      | student_book_number | group_id
-----+-----+-----+-----
  1 | Иванов Иван Иванович |         123456      |         1
  2 | Петров Петр Петрович |         654321      |         1
  3 | Сидоров Сидор Сидорович |         112233      |         2
  4 | Иванова Мария Сергеевна |         789012      |         4
(4 строки)

DB=#

```

## Создание двух представлений: изменяемое, неизменяемое:

### 1. Изменяемое

```
CREATE VIEW student_groups AS
```

```
SELECT s.id AS student_id, s.full_name, s.student_book_number, g.name AS  
group_name
```

```
FROM students s
```

```
JOIN groups g ON s.group_id = g.id;
```

```
DB=# CREATE VIEW student_groups AS  
DB=# SELECT s.id AS student_id, s.full_name, s.student_book_number, g.name AS group_name  
DB=# FROM students s  
DB=# JOIN groups g ON s.group_id = g.id;  
CREATE VIEW  
DB=# SELECT * FROM student_groups;  
 student_id |      full_name      | student_book_number | group_name  
-----+-----+-----+-----  
          2 | Петров Петр Петрович |         654321      | Группа 101  
          1 | Иванов Иван Иванович |         123456      | Группа 101  
          3 | Сидоров Сидор Сидорович |        112233      | Группа 102  
          4 | Иванова Мария Сергеевна |        789012      | Группа 301  
(4 строки)
```

Создание безусловного правила

```
CREATE RULE update_student_groups AS
```

```
ON UPDATE TO student_groups
```

```
DO INSTEAD
```

```
UPDATE students
```

```
SET full_name = NEW.full_name,
```

```
    student_book_number = NEW.student_book_number
```

```
WHERE id = NEW.student_id;
```

```
DB=# SELECT * FROM student_groups;
```

student_id	full_name	student_book_number	group_name
1	Иванова Мария Ивановна	123456	Группа 101
2	Петров Петр Петрович	654321	Группа 101
3	Сидоров Сидор Сидорович	112233	Группа 102
4	Иванова Мария Сергеевна	789012	Группа 301

(4 строки)

## 2. Неизменяемое

CREATE VIEW student\_group\_faculties AS

SELECT s.id AS student\_id, s.full\_name, s.student\_book\_number, g.name AS group\_name, f.name AS faculty\_name

FROM students s

JOIN groups g ON s.group\_id = g.id

JOIN faculties f ON g.faculty\_id = f.id;

```
DB=# CREATE VIEW student_group_faculties AS
DB=# SELECT s.id AS student_id, s.full_name, s.student_book_number, g.name AS group_name, f.name AS faculty_name
DB=# FROM students s
DB=# JOIN groups g ON s.group_id = g.id
DB=# JOIN faculties f ON g.faculty_id = f.id;
CREATE VIEW
DB=# SELECT * FROM student_group_faculties;
ОШИБКА: отношение "student_group_faculties" не существует
СТРОКА 1: SELECT * FROM student_group_faculties;
```

```
DB=# SELECT * FROM student_group_faculties;
```

student_id	full_name	student_book_number	group_name	faculty_name
1	Иванова Мария Ивановна	123456	Группа 101	Факультет информатики
2	Петров Петр Петрович	654321	Группа 101	Факультет информатики
3	Сидоров Сидор Сидорович	112233	Группа 102	Факультет информатики
4	Иванова Мария Сергеевна	789012	Группа 301	Факультет биологии

(4 строки)

## Подключение БД к C# .Net , WindowsForms через CMD

Form1

ПОДКЛЮЧЕНИЕ NOT CONNECTED

факультеты

группы

студенты

лекторы

дисциплины

экзамены

Сделать запрос

Для подключения к БД необходимо нажать на кнопку “Подключение”.

Form1

ПОДКЛЮЧЕНИЕ ОК

факультеты

группы

студенты

лекторы

дисциплины

экзамены

Сделать запрос



Далее можем получить таблицы из БД или написать собственный запрос.

Form1

ПОДКЛЮЧЕНИЕ    ОК

факультеты  
группы  
студенты  
лекторы  
дисциплины  
экзамены

Сделать запрос

	id	name
▶	1	Факультет инф...
	2	Факультет мат...
	3	Факультет физ...
	4	Факультет биол...
*		

Form1

ПОДКЛЮЧЕНИЕ    ОК

факультеты  
группы  
студенты  
лекторы  
дисциплины  
экзамены

Сделать запрос

	id	name	faculty_id
▶	1	Группа 101	1
	2	Группа 102	1
	3	Группа 201	2
	4	Группа 301	4
*			

Form1

ПОДКЛЮЧЕНИЕ    ОК

факультеты

группы

**студенты**

лекторы

дисциплины

экзамены

	id	full_name	student_book...
▶	2	Петров Петр Пе...	654321
	3	Сидоров Сидор ...	112233
	4	Иванова Мария ...	789012
	1	Иванова Мария ...	123456
*			

Сделать запрос

Form1

ПОДКЛЮЧЕНИЕ    ОК

факультеты

группы

студенты

лекторы

дисциплины

экзамены

	average_grade
▶	85,000000000000...
*	

Сделать запрос

```

SELECT
  AVG(subquery.grade) AS average_grade
FROM
  (
    SELECT grade
    FROM exams
  ) AS subquery;

```

В случае, если мы не подключены в БД, статус выдаст ошибку при попытке получать какой-либо результат.

Form1

ПОДКЛЮЧЕНИЕ ERROR

факультеты

группы

студенты

лекторы

дисциплины

экзамены

Сделать запрос

## Ссылка на Github репозиторий

<https://github.com/nemajor1/cour>

