

# sem2\_Nemkov

## Task 1

```
-- task 1
-- Выведите список студентов с их статусом успеваемости
-- - "Отличник" (средний балл ≥ 4.5)
-- - "Хорошист" (средний балл ≥ 3.5, но < 4.5)
-- - "Троечник" (средний балл < 3.5)
SELECT
    id,
    name,
    surname,
    patronymic,
    avg_grade,
    CASE
        WHEN avg_grade >= 4.5 THEN 'Отличник'
        WHEN avg_grade >= 3.5 AND avg_grade < 4.5 THEN 'Хорошист'
        ELSE 'Троечник'
    END AS status
FROM
    Students;
```

## Результат:

	id [PK] integer	name text	surname text	patronymic text	avg_grade numeric (3,2)	status text
1	1	Иван	Петров	Александрович	4.80	Отличник
2	2	Мария	Сидорова	Павловна	3.60	Хорошист
3	3	Алексей	Иванов	Сергеевич	4.20	Хорошист
4	4	Ольга	Кузнецова	Игоревна	4.50	Отличник
5	5	Дмитрий	Смирнов	Владимирович	3.20	Троечник
6	6	Екатерина	Федорова	Дмитриевна	4.90	Отличник
7	7	Николай	Васильев	Аркадьевич	3.80	Хорошист
8	8	Светлана	Михайлова	Олеговна	4.10	Хорошист
9	9	Григорий	Алексеев	Леонидович	3.70	Хорошист
10	10	Анна	Зайцева	Ильинична	4.60	Отличник
11	11	Павел	Тимофеев	Петрович	3.90	Хорошист
12	12	Татьяна	Громова	Андреевна	4.00	Хорошист
13	13	Сергей	Орлов	Алексеевич	4.70	Отличник
14	14	Юлия	Костина	Романовна	3.50	Хорошист
15	15	Артем	Никитин	Михайлович	4.30	Хорошист
16	16	Ирина	Шубина	Викторовна	3.60	Хорошист
17	17	Владимир	Егоров	Константинович	4.40	Хорошист
18	18	Евгений	Морозов	Денисович	3.30	Троечник
19	19	Оксана	Филатова	Петровна	4.20	Хорошист
20	20	Роман	Данилов	Игоревич	4.00	Хорошист

## Task 2

```
-- task 2
-- Создайте представление, которое:
-- - Показывает только активных студентов, у которых средний балл выше
-- среднего по курсу.
-- - Для каждого студента считает количество курсов, на которые он
-- записан.
-- - Включает строковые функции для преобразования ФИО в формат "Фамилия
-- И.О."
```

```
CREATE VIEW ActiveStudents AS
SELECT
    s.id AS student_id,
    CONCAT(s.surname, ' ', LEFT(s.name, 1), '.'),
    COALESCE(LEFT(s.patronymic, 1), ''), '.' AS full_name,
```

```

        s.avg_grade,
        e.course_id,
        COUNT(e.course_id) OVER (PARTITION BY s.id) AS courses_count
FROM
    Students s
JOIN
    Enrollments e ON s.id = e.student_id
JOIN (
    SELECT
        course_id,
        AVG(s.avg_grade) AS avg_course_grade
    FROM
        Enrollments e
    JOIN
        Students s ON e.student_id = s.id
    WHERE
        e.status = 'active'
    GROUP BY
        course_id
) course_avg
ON e.course_id = course_avg.course_id
WHERE
    e.status = 'active'
    AND s.avg_grade > course_avg.avg_course_grade
GROUP BY
    s.id, s.surname, s.name, s.patronymic, s.avg_grade, e.course_id;

SELECT * FROM ActiveStudents;

```




## Результат:

	student_id integer	full_name text	avg_grade numeric (3,2)	course_id integer	courses_count bigint
1	1	Петров И.А.	4.80	2	2
2	1	Петров И.А.	4.80	3	2
3	7	Васильев Н.А.	3.80	7	1
4	8	Михайлова С.О.	4.10	9	1
5	10	Зайцева А.И.	4.60	1	2
6	10	Зайцева А.И.	4.60	4	2
7	13	Орлов С.А.	4.70	2	1

## Task 3

```
-- task 3
-- Найдите самый сложный курс, требующий прохождения
-- максимального количества предварительных курсов.
-- (Используйте рекурсивный cte)
WITH RECURSIVE RecQuery AS (
SELECT
    id,
    title,
    prerequisite_id,
    1 AS depth
FROM Courses
WHERE prerequisite_id IS NOT NULL
    UNION ALL
SELECT
    c.id,
    c.title,
    c.prerequisite_id,
    cp.depth + 1
FROM Courses c
JOIN RecQuery cp
    ON c.prerequisite_id = cp.id
)
SELECT
    id,
    title,
    MAX(depth) AS max_depth
FROM RecQuery
GROUP BY id, title
ORDER BY max_depth DESC
LIMIT 1;
```

## Результат

	id integer 	title text 	max_depth integer 
1	10	Big Data и ML	3

## Task 4

```
-- task 4
-- Найдите 10 самых востребованных языков программирования, которые
-- студенты знают.
SELECT
    UNNEST(skills) AS languages,
    COUNT(*) AS count
FROM
    Students
GROUP BY
    languages
ORDER BY
    count DESC
LIMIT 10;
```

## Результат

	languages text 	count bigint 
1	Python	9
2	Java	4
3	SQL	4
4	C++	4
5	Rust	3
6	JavaScript	3
7	Scala	2
8	R	2
9	Kotlin	2
10	Objective-C	2

---

## Task 5







```
-- task 5
-- Найдите самых успешных студентов, учитывая:
-- - Средний балл выше 4.0.
-- - Минимум два курса пройдено.
-- - Знание Python или SQL.
SELECT
    s.id,
    s.name,
    s.surname,
    s.patronymic,
    s.avg_grade,
    COUNT(e.course_id) AS completed_courses
```

```

FROM
    Students s
JOIN
    Enrollments e ON s.id = e.student_id
WHERE
    s.avg_grade > 4.0
    AND e.status = 'completed'
    AND (ARRAY['Python'] && s.skills OR ARRAY['SQL'] && s.skills)
GROUP BY
    s.id, s.name, s.surname, s.patronymic, s.avg_grade
HAVING
    COUNT(e.course_id) >= 2
ORDER BY
    s.avg_grade DESC;

```

## Результат

	id [PK] integer 	name text 	surname text 	patronymic text 	avg_grade numeric (3,2) 	completed_courses bigint 
1	6	Екатерина	Федорова	Дмитриевна	4.90	2
2	15	Артём	Никитин	Михайлович	4.30	2