Lesson 24 - Group by и объединение запросов

Повторение

1. Теория из прошлого урока
2. Разбор домашнего задания

Legend

```sql
-- Группировка данных (GROUP BY)
-- Количество сотрудников в каждом отделе:
SELECT department, COUNT(*)
FROM employees
GROUP BY department;

SELECT avg(salary)
FROM employees;
-- 5754.5454545454545455

-- Средняя зарплата по отделам:
SELECT department, AVG(salary)
FROM employees
GROUP BY department;

SELECT first_name, last_name, COUNT(*)
FROM employees
GROUP BY first_name, last_name;

SELECT salary, COUNT(*)
FROM employees
GROUP BY salary
ORDER BY COUNT(*);

-- Фильтрация групп (HAVING)
-- Отделы, где более 3 сотрудников:
SELECT department, COUNT(*)
FROM employees
GROUP BY department
HAVING COUNT(*) >= 6;
```

```sql
-- Подзапрос в SELECT
-- Общий доход сотрудников по отделу:
SELECT department
FROM employees;
SELECT SUM(salary)
FROM employees;

-- SELECT department, SUM(salary)
-- FROM employees;

SELECT department,
       (SELECT SUM(salary)
        FROM employees e2
        WHERE e2.department = e1.department) AS total_salary
FROM employees e1
GROUP BY department;

-- Подзапрос в WHERE
-- Сотрудники с зарплатой выше средней:
SELECT last_name, first_name, salary
FROM employees
WHERE salary > 5000
ORDER BY salary;

SELECT AVG(salary)
FROM employees;

SELECT last_name, first_name, salary
FROM employees
WHERE salary > (SELECT AVG(salary) FROM employees)
ORDER BY salary;

SELECT AVG(salary)
FROM employees;
-- Сотрудники с максимальной зарплатой в каждом отделе:
SELECT e1.first_name, e1.salary, e1.department
FROM employees e1
WHERE salary = (SELECT MAX(e2.salary)
                FROM employees e2
                WHERE e2.department = e1.department);

-- Другие полезные операции
-- DISTINCT (удаление дублирующихся значений):
SELECT DISTINCT department
FROM employees;
```

```sql
SELECT DISTINCT department
FROM employees;


-- CASE (условные выражения):
-- Категоризация зарплат:
SELECT first_name,
       salary,
       CASE
           WHEN salary < 5000 THEN 'Low' -- Когда з/п < 5000, тогда Low
           WHEN salary = 6000 AND first_name = 'Bob' THEN 'Сын директора' --
Когда з/п < 5000, тогда Low
           WHEN salary BETWEEN 5000 AND 6999 THEN 'Medium'
           ELSE 'High'
           END AS salary_category
FROM employees
ORDER BY salary DESC;

-- Использование выражения COALESCE
-- Замена NULL значений на значение по умолчанию:
SELECT first_name, deleted_at
FROM employees;


SELECT first_name, COALESCE(deleted_at, 'Не удалено') AS status
FROM employees;

SELECT first_name,
       COALESCE(deleted_at, CURRENT_TIMESTAMP) AS status
FROM employees;

SELECT first_name,
       COALESCE(TO_CHAR(deleted_at, 'YYYY-MM-DD HH24:MI:SS'), 'Не удалено')
AS status
FROM employees;

---- status ----------
-- 2025-01-22 19:11:33 - timestamp
-- Не удалено - string (varchar)
-- Не удалено - string (varchar)
-- Не удалено - string (varchar)
-- 2025-01-22 19:11:33 - timestamp


-- Альтернатива
SELECT first_name,
```

```
       CASE
           WHEN deleted_at IS NULL THEN 'Не удалено'
           ELSE TO_CHAR(deleted_at, 'YYYY-MM-DD HH24:MI:SS')
           END AS status
FROM employees;

-- === **********************************************************************
=== ---

-- UNION и UNION ALL
-- Используются для объединения результатов нескольких запросов.

SELECT first_name, last_name, email FROM customers
UNION ALL
SELECT first_name, last_name, email FROM employees;

-- UNION:
-- Убирает дубликаты в результирующем наборе.
SELECT first_name, department
FROM employees
WHERE department = 'HR'
UNION
SELECT first_name, department
FROM employees
WHERE department = 'Marketing'
ORDER BY first_name;

-- UNION ALL:
-- Сохраняет дубликаты.
SELECT first_name, department
FROM employees
WHERE department = 'HR'
UNION ALL
SELECT first_name, department
FROM employees
WHERE department = 'Marketing'
ORDER BY first_name;

SELECT first_name, last_name, department
FROM employees
WHERE department IN ('HR', 'Marketing')
ORDER BY first_name, department;

-- INTERSECT:
-- Возвращает только те строки, которые присутствуют во всех запросах.
SELECT first_name, last_name, email FROM customers
```

```sql
INTERSECT
SELECT first_name, last_name, email FROM employees;

SELECT first_name, department, salary
FROM employees
WHERE department = 'HR'
INTERSECT
SELECT first_name, department, salary
FROM employees
WHERE salary > 5500;

SELECT first_name, department, salary
FROM employees
WHERE department = 'HR';
SELECT first_name, department, salary
FROM employees
WHERE salary > 5500;



-- EXCEPT:
-- Возвращает строки из первого запроса, которых нет во втором.

-- A - B
SELECT first_name, last_name FROM customers
EXCEPT ALL
SELECT first_name, last_name FROM employees;

SELECT first_name, department, salary
FROM employees
WHERE department = 'HR'
EXCEPT
SELECT first_name, department, salary
FROM employees
WHERE salary < 5000;

-- EXCEPT ALL:
-- Возвращает все строки из первого запроса, которых нет во втором.
SELECT first_name, department, salary
FROM employees
WHERE department = 'HR'
EXCEPT ALL
SELECT first_name, department, salary
FROM employees
WHERE salary < 5000;
```

```sql
-- INTERSECT ALL:
-- Возвращает все строки, которые есть в обоих запросах.
SELECT first_name
FROM employees
WHERE department = 'HR'
INTERSECT ALL
SELECT first_name
FROM employees
WHERE salary > 5000;
```

ДЗ-24

```sql
-- Часть 6: Подзапросы
-- Найдите сотрудников, у которых зарплата выше средней.
-- Для каждого отдела выведите общую сумму зарплат сотрудников.
-- Найдите сотрудников с максимальной зарплатой в каждом отделе.

-- Часть 7: CASE и группировка
-- Разделите всех сотрудников на три категории зарплат:
-- Low (меньше 5000),
-- Medium (от 5000 до 6999),
-- High (7000 и выше). Выведите их имена, зарплату и категорию.
-- Найдите отделы, где работает более 4 сотрудников.
-- Выведите среднюю зарплату для каждого отдела, но только для отделов, где
работает более 3 сотрудников.

-- Часть 9: Работа с объединением запросов (UNION, INTERSECT, EXCEPT)
-- Используя UNION, объедините списки сотрудников из отделов 'HR' и
'Finance', исключая дубликаты. Отсортируйте результат по фамилии.
-- Используя UNION ALL, объедините списки сотрудников из отделов 'HR' и
'Marketing', сохранив дубликаты.
-- Используя INTERSECT, найдите сотрудников из отдела 'HR', чья зарплата
превышает 5500.
-- Используя EXCEPT, выберите сотрудников из отдела 'Engineering', которые
не работают в отделе 'HR'.
-- Используя INTERSECT ALL, найдите сотрудников, работающих в отделе
'Finance', чьи зарплаты превышают 6000.
```