





Lab_2_Nemkov_KMBO_05_23

Task 1.

Напишите запрос, который выводит статистику по количеству заказов (total_orders) и выручке (total_revenue) для всех возможных комбинаций города и региона, но только для тех пунктов выдачи, где выручка превышает среднюю выручку по их региону. Добавьте общие итоги по каждому городу и региону.

```
SELECT
    COALESCE(p.region, 'Все регионы') AS region,
    COALESCE(p.city, 'Все города') AS city,
    SUM(p.total_orders) AS total_orders,
    SUM(p.total_revenue) AS total_revenue
FROM pickup_points p
JOIN (
    SELECT
        region,
        AVG(total_revenue) AS avg_revenue
    FROM pickup_points
    GROUP BY region
) r ON p.region = r.region
WHERE p.total_revenue > r.avg_revenue
GROUP BY ROLLUP (p.region, p.city)
ORDER BY p.region, p.city;
```

Result:

	region character varying 	city character varying 	total_orders bigint 	total_revenue numeric 
1	Дальневосточный	Владивосток	370	185000.00
2	Дальневосточный	Хабаровск	190	95000.00
3	Дальневосточный	Южно-Сахалинск	170	85000.00
4	Дальневосточный	Все города	730	365000.00
5	Крымский	Севастополь	170	85000.00
6	Крымский	Симферополь	310	155000.00
7	Крымский	Ялта	150	75000.00
Total rows: 40 of 40 Query complete 00:00:00.125 Ln 16, Col 27				

Task 2.

Напишите запрос, который выводит иерархическую статистику по выручке (total_revenue) для каждого региона, города в регионе и общего итога по всем регионам. При этом исключите из результата строки, где выручка меньше 10% от общей выручки по региону.

```
SELECT
    COALESCE(p.region, 'Все регионы') AS region,
    COALESCE(p.city, 'Все города') AS city,
    SUM(p.total_revenue) AS total_revenue
FROM pickup_points p
JOIN (
    SELECT
        region,
        SUM(total_revenue) AS region_revenue
    FROM pickup_points
    GROUP BY region
) r ON p.region = r.region
GROUP BY GROUPING SETS ((p.region, p.city), (p.region), ())
HAVING SUM(p.total_revenue) >= 0.1 * MAX(r.region_revenue);
```

Result:

	region character varying	city character varying	total_revenue numeric
1	Все регионы	Все города	7595000.00
2	Центральный	Воронеж	210000.00
3	Южный	Ростов-на-Дону	645000.00
4	Приволжский	Самара	235000.00
5	Северный	Воркута	80000.00
6	Центральный	Тула	130000.00
7	Приволжский	Казань	480000.00
8	Уральский	Тюмень	145000.00
9	Северо-Западный	Мурманск	130000.00
10	Южный	Краснодар	210000.00
Total rows: 41 of 41 Query complete 00:00:00.113 Ln 9, Col 24			

image-122.png

Task 3.

Напишите запрос, который выводит все возможные комбинации группировок по городу и региону, но только для тех пунктов выдачи, где количество заказов (*total_orders*) превышает медианное значение по их региону. Добавьте общие итоги.

Указание: для вычисления медианы воспользуйтесь оконной функцией: percentile_cont(0.5) WITHIN GROUP (ORDER BY total_orders) AS median_order

```

SELECT
    COALESCE(p.region, 'Все регионы') AS region,
    COALESCE(p.city, 'Все города') AS city,
    SUM(p.total_orders) AS total_orders
FROM pickup_points p
JOIN (
    SELECT
        region,
        percentile_cont(0.5) WITHIN GROUP (ORDER BY total_orders) AS
median_order
    FROM pickup_points
    GROUP BY region
) r ON p.region = r.region

```

```
WHERE p.total_orders > r.median_order
GROUP BY CUBE (p.region, p.city)
ORDER BY p.region, p.city;
```

Result:

	region character varying	city character varying	total_orders bigint	
1	Дальневосточный	Владивосток	500	
2	Дальневосточный	Хабаровск	190	
3	Дальневосточный	Южно-Сахалинск	170	
4	Дальневосточный	Все города	860	
5	Крымский	Севастополь	170	
6	Крымский	Симферополь	310	
7	Крымский	Ялта	150	
8	Крымский	Все города	630	
9	Приволжский	Казань	590	
10	Приволжский	Нижний Новгород	190	
Total rows: 67 of 67 Query complete 00:00:00.103 Ln 1, Col 1				

image-123.png

Task 4.

Напишите запрос, который выводит для каждого региона три пункта выдачи с наибольшей выручкой (total~~revenue~~), но только если эти пункты были созданы в последние 6 месяцев и их выручка составляет не менее 20% от общей выручки региона.

Указание: для отбора последних шести месяцев: created_at >= NOW() - INTERVAL '6 months'

```
WITH recent_points AS (
  SELECT
    id,
    name,
    city,
    region,
    total_revenue,
```

```

        created_at
    FROM
        pickup_points
    WHERE
        created_at >= NOW() - INTERVAL '6 months'
),
region_revenue AS (
    SELECT
        region,
        SUM(total_revenue) as total_region_revenue
    FROM recent_points
    GROUP BY region
),
ranked_pickup_points AS (
    SELECT
        p.id,
        p.name,
        p.city,
        p.region,
        p.total_revenue,
        p.created_at,
        r.total_region_revenue,
        RANK() OVER (PARTITION BY p.region ORDER BY p.total_revenue DESC) AS
rank
    FROM recent_points AS p
    JOIN region_revenue AS r ON p.region = r.region
    WHERE p.total_revenue >= r.total_region_revenue * 0.2
)
SELECT
    *
FROM ranked_pickup_points
WHERE rank <= 3
ORDER BY region

```

Result:

	id [PK] integer	name character varying (255)	city character varying (100)	region character varying (100)	total_revenue numeric (10,2)	created_at timestamp without time zone	total_region_revenue numeric
1	31	Пункт выдачи 31	Владивосток	Дальневосточный	80000.00	2025-07-15 11:30:00	280000.00
2	32	Пункт выдачи 32	Владивосток	Дальневосточный	65000.00	2025-08-20 10:45:00	280000.00
3	41	Пункт выдачи 41	Симферополь	Крымский	60000.00	2026-05-05 01:10:00	200000.00
4	42	Пункт выдачи 42	Симферополь	Крымский	50000.00	2026-06-10 00:00:00	200000.00
5	43	Пункт выдачи 43	Севастополь	Крымский	40000.00	2026-07-15 23:50:00	200000.00
6	21	Пункт выдачи 21	Казань	Приволжский	100000.00	2024-09-20 13:00:00	360000.00
7	22	Пункт выдачи 22	Казань	Приволжский	80000.00	2024-10-25 17:00:00	360000.00
8	46	Пункт выдачи 46	Архангельск	Северный	50000.00	2026-10-30 20:20:00	150000.00
9	47	Пункт выдачи 47	Архангельск	Северный	40000.00	2026-11-05 19:10:00	150000.00
10	48	Пункт выдачи 48	Мурманск	Северный	30000.00	2026-12-10 18:00:00	150000.00

Total rows: 15 of 15 Query complete 00:00:00.177 Ln 43, Col 16

image-124.png

Task 5.

Напишите запрос, который выводит рейтинг пунктов выдачи внутри каждого региона на основе выручки (**total_revenue**), *но только для тех пунктов, которые работают более 2 лет и имеют количество заказов (total_orders) выше среднего по региону. Убедитесь, что рейтинг учитывает только актуальные данные.*

Указание: для отбора пунктов, которые работают более двух лет: `created_at <= NOW() - INTERVAL '2 years'`

```
SELECT
    p.region,
    p.name,
    p.total_revenue,
    RANK() OVER (PARTITION BY p.region ORDER BY p.total_revenue DESC) AS
rank
FROM pickup_points p
JOIN (
    SELECT
        region,
        AVG(total_orders) AS avg_orders
    FROM pickup_points
    GROUP BY region
) r ON p.region = r.region
WHERE
    p.created_at <= NOW() - INTERVAL '2 years'
    AND p.total_orders > r.avg_orders;
```

Result:

	region character varying (100) 🔒	name character varying (255) 🔒	total_revenue numeric (10,2) 🔒	rank bigint 🔒
1	Дальневосточный	Пункт выдачи 69	105000.00	1
2	Дальневосточный	Пункт выдачи 70	95000.00	2
3	Дальневосточный	Пункт выдачи 71	85000.00	3
4	Крымский	Пункт выдачи 75	95000.00	1
5	Крымский	Пункт выдачи 76	85000.00	2
6	Крымский	Пункт выдачи 77	75000.00	3
7	Приволжский	Пункт выдачи 63	115000.00	1
8	Приволжский	Пункт выдачи 64	105000.00	2
9	Приволжский	Пункт выдачи 65	95000.00	3
10	Северный	Пункт выдачи 78	90000.00	1
Total rows: 31 of 31 Query complete 00:00:00.167 Ln 1, Col 1				

image-125.png

Task 6.

Напишите запрос, который выводит для каждого пункта выдачи разницу в выручке (total_revenue) между текущим и предыдущим пунктом в том же городе, но только если разница превышает 10% от выручки предыдущего пункта. Убедитесь, что разница рассчитывается только для пунктов, созданных в одном году.

```
WITH revenue_lag AS (
    SELECT
        city,
        id,
        total_revenue,
        EXTRACT(YEAR FROM created_at) AS year,
        LAG(total_revenue) OVER (PARTITION BY city, EXTRACT(YEAR FROM
created_at) ORDER BY id) AS prev_revenue
    FROM pickup_points
)
SELECT
    city,
    id,
    total_revenue - prev_revenue AS revenue_diff
FROM revenue_lag
WHERE
```

```
prev_revenue IS NOT NULL
AND total_revenue > 1.1 * prev_revenue;
```

Result:




	city character varying (100) 	id [PK] integer 	revenue_diff numeric 
1	Владивосток	100	10000.00
2	Владивосток	101	10000.00
3	Екатеринбург	91	15000.00
4	Екатеринбург	92	10000.00
5	Казань	94	10000.00
6	Казань	95	15000.00
7	Москва	82	10000.00
8	Москва	83	15000.00
9	Новосибирск	88	10000.00
10	Новосибирск	89	10000.00
Total rows: 14 of 14 Query complete 00:00:00.178 Ln 1, Col 1			

image-126.png

Task 7.

Создайте пользовательский тип `working_schedule`, который включает поля `opening_time`, `closing_time` и `weekend`. Измените таблицу `pickup_points`, чтобы она использовала этот тип для хранения рабочего расписания. Напишите запрос, который выводит все пункты выдачи, где `opening_time` раньше 08:00, а `closing_time` позже 22:00, и при этом они работают в выходные дни.

Указание: для демонстрации работы запроса достаточно будет заполнить `working_schedule` для 10 пунктов выдачи

```
CREATE TYPE working_schedule AS (  
    opening_time TIME,  
    closing_time TIME,  
    weekend BOOLEAN  
);
```

```
ALTER TABLE pickup_points
```



```
ALTER COLUMN working_hours TYPE working_schedule USING
(
    ROW(
        (working_hours->>'opening_time')::TIME,
        (working_hours->>'closing_time')::TIME,
        (working_hours->>'weekend')::BOOLEAN
    )::working_schedule
);

SELECT * FROM pickup_points
WHERE (working_hours).opening_time < '08:00'::time AND
      (working_hours).closing_time > '22:00'::time AND
      (working_hours).weekend;
```

Result:

	id	name	city	region	total_orders	total_revenue	working_hours	created_at
	[PK] integer	character varying (255)	character varying (100)	character varying (100)	integer	numeric (10,2)	working_schedule	timestamp without time
1	1	Пункт выдачи 1	Москва	Центральный	200	100000.00	(07:00:00,23:00:00,t)	2023-01-15 10:00:00
2	5	Пункт выдачи 5	Тула	Центральный	80	40000.00	(07:30:00,23:30:00,t)	2023-05-12 16:20:00
3	11	Пункт выдачи 11	Новосибирск	Сибирский	180	90000.00	(07:00:00,23:00:00,t)	2023-11-11 09:00:00
4	15	Пункт выдачи 15	Красноярск	Сибирский	80	40000.00	(07:30:00,23:30:00,t)	2024-03-08 11:45:00
5	21	Пункт выдачи 21	Казань	Приволжский	200	100000.00	(07:00:00,23:00:00,t)	2024-09-20 13:00:00
6	25	Пункт выдачи 25	Нижний Новгород	Приволжский	100	50000.00	(07:30:00,23:30:00,t)	2025-01-15 10:10:00
7	31	Пункт выдачи 31	Владивосток	Дальневосточный	160	80000.00	(07:00:00,23:00:00,t)	2025-07-15 11:30:00
8	35	Пункт выдачи 35	Южно-Сахалинск	Дальневосточный	70	35000.00	(07:30:00,23:30:00,t)	2025-11-05 07:30:00
9	41	Пункт выдачи 41	Симферополь	Крымский	120	60000.00	(07:00:00,23:00:00,t)	2026-05-05 01:10:00
10	45	Пункт выдачи 45	Ялта	Крымский	40	20000.00	(07:30:00,23:30:00,t)	2026-09-25 21:30:00

Total rows: 10 of 10 Query complete 00:00:00.139 Ln 20, Col 31

image-127.png

Task 8.

Напишите запрос, который выводит название пункта выдачи и его рабочие часы (поле working_hours), но только для тех пунктов, где в рабочем расписании указано, что они работают в выходные дни (поле weekend равно true), а также выручка (total_revenue) превышает среднюю выручку по городу.

```
WITH city_avg_revenue AS (
    SELECT
        city,
        AVG(total_revenue) as city_total_revenue
    FROM pickup_points
    GROUP BY city
)
```

```

SELECT
  name,
  working_hours
FROM pickup_points AS P
JOIN city_avg_revenue AS cavg ON p.city = cavg.city
WHERE
  p.total_revenue > cavg.city_total_revenue
  AND (p.working_hours).weekend

```

Result:



	name character varying (255) 	working_hours working_schedule 
1	Пункт выдачи 1	(07:00:00,23:00:00,t)
2	Пункт выдачи 6	(08:00:00,20:00:00,t)
3	Пункт выдачи 11	(07:00:00,23:00:00,t)
4	Пункт выдачи 16	(08:00:00,20:00:00,t)
5	Пункт выдачи 21	(07:00:00,23:00:00,t)
6	Пункт выдачи 31	(07:00:00,23:00:00,t)
7	Пункт выдачи 52	(09:00:00,21:00:00,t)
8	Пункт выдачи 54	(08:00:00,20:00:00,t)
9	Пункт выдачи 56	(10:00:00,22:00:00,t)
10	Пункт выдачи 57	(08:00:00,20:00:00,t)
Total rows: 28 of 28 Query complete 00:00:00.147 Ln 15, Col 32		

image-128.png

Task 9.

Напишите запрос, который обновляет поле `working_hours` для всех пунктов выдачи, добавляя новое поле `holiday_hours` со значением `null`, если такое поле еще не существует. При этом обновление должно происходить только для пунктов, где количество заказов (`total_orders`) превышает 100.

Todo:

Task 10.

Напишите запрос, который выводит среднюю выручку (total_revenue) для пунктов выдачи, сгруппированных по наличию (поле weekend равно true) или отсутствию (поле weekend равно false) поля weekend в working_hours. Убедитесь, что результат включает только те группы, где количество пунктов больше 5, а также выводит общее количество пунктов в каждой группе.

```
SELECT
    (working_hours).weekend,
    AVG(total_revenue) AS avg_revenue,
    COUNT(*) AS points_count
FROM pickup_points
GROUP BY (working_hours).weekend
HAVING COUNT(*) > 5;
```

Result:




	weekend boolean 	avg_revenue numeric 	points_count bigint 	
1	false	73846.153846153846	39	
2	true	76048.387096774194	62	
Total rows: 2 of 2 Query complete 00:00:00.117 Ln 2, Col 1				

image-129.png