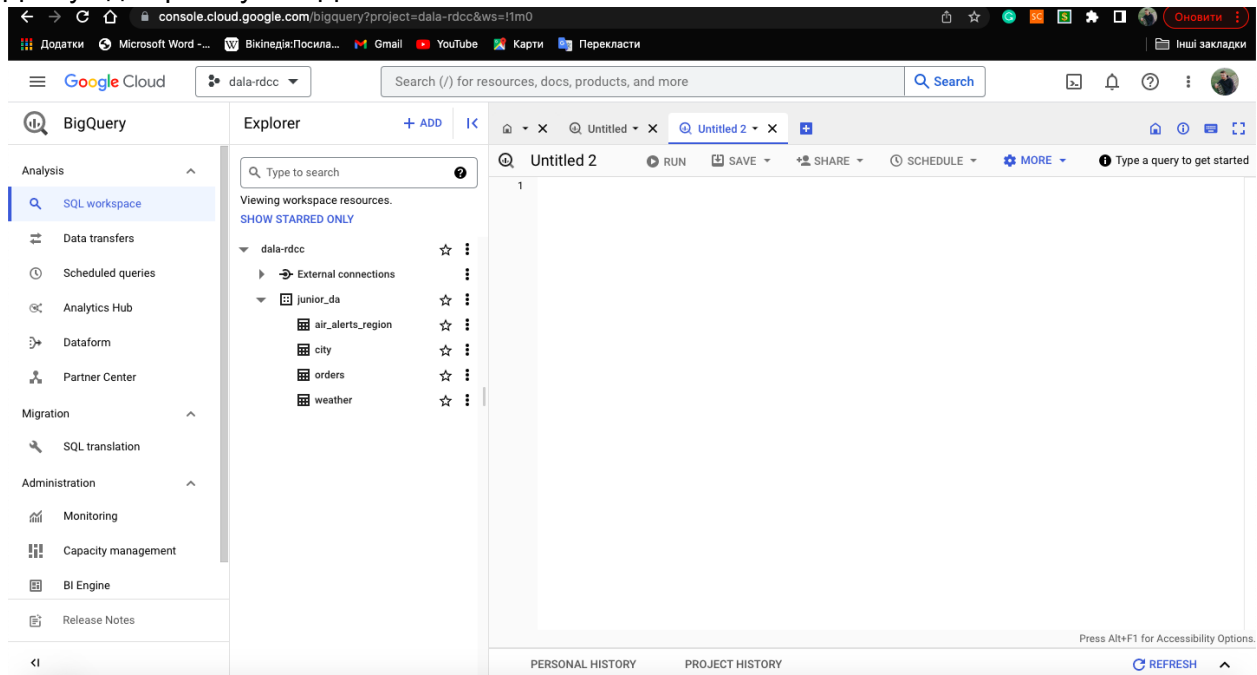


1. Синтаксис SQL: Оператори

Завдання 1.1

Доступ до проекту та БД:



Завдання 1.2

1.2.1. Написати запит, який виведе з таблиці замовлень(**orders**) наступні колонки:

- ID замовлення,
- ID водія,
- ID пасажирів,
- ID міста,
- час розміщення замовлення пасажиром,
- повна сума замовлення,
- тип класу таксі,
- тип оплати. Обмежити кількість рядків у результуючій таблиці — 20 записів.

```
SELECT order_id
,driver_id
,rider_id
,city_id
,placed_at
,order_amount
,product
,payment_type
FROM junior_da.orders
```

`LIMIT 20;`

Search (/) for resources, docs, products, and more

Search

Untitled 2

orders

Query completed.

```
1 SELECT order_id
2     ,driver_id
3     ,rider_id
4     ,city_id
5     ,placed_at
6     ,order_amount
7     ,product
8     ,payment_type
9 FROM junior_da.orders
10 LIMIT 20;
```

Query results

SAVE RESULTSEXPLORE DATA

JOB INFORMATIONRESULTSJSONEXECUTION DETAILSEXECUTION GRAPHPREVIEW

Row	rider_id	city_id	placed_at	order_amount	product	payment_type	
1	ic...	0140b3a5-01df-5639-bd22-17a...	1006	2022-01-01 00:27:07 UTC	133	Standard	PAYMENT_TYPE_CARD
2	7...	08f90dbd-2ff0-5add-b956-050...	1006	2022-01-01 01:21:33 UTC	308	Driver	PAYMENT_TYPE_CARD
3	78...	187cf2b6-7830-5703-a5d1-97c...	1006	2022-01-01 01:25:21 UTC	86	Standard	PAYMENT_TYPE_CARD
4	2...	70d897a6-f36b-5502-9762-d3f...	1006	2022-01-01 01:39:25 UTC	117	Standard	PAYMENT_TYPE_CARD
5	id...	dfdc1983-384d-568a-a6af-ae7...	1006	2022-01-01 01:41:57 UTC	117	Standard	PAYMENT_TYPE_CARD

Results per page: 501 - 20 of 20

PERSONAL HISTORYPROJECT HISTORYREFRESH

1.2.2. Вивести унікальні типи оплат з таблиці замовлень.

Результуюча таблиця має містити одну колонку — `payment_type`.

```
SELECT DISTINCT payment_type
FROM junior_da.orders;
```

Search (/) for resources, docs, products, and more

Search

Untitled 2

```
4 ,city_id
5 ,placed_at
6 ,order_amount
7 ,product
8 ,payment_type
9 FROM junior_da.orders
10 LIMIT 20;
11 --
12 SELECT DISTINCT payment_type
13 FROM junior_da.orders;
14
15
```

Query completed.

Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	payment_type
1	PAYMENT_TYPE_CARD
2	PAYMENT_TYPE_GOOGLE_PAY
3	PAYMENT_TYPE_APPLE_PAY
4	PAYMENT_TYPE_CASH
5	PAYMENT_TYPE_CORPORATE...

PERSONAL HISTORY PROJECT HISTORY

REFRESH

1.2.3. Вивести унікальні типи класу таксі з таблиці замовлень.

Результуюча таблиця має містити одну колонку — `product`.

`SELECT DISTINCT product`

`FROM junior_da.orders;`

Search (/) for resources, docs, products, and more

Search

Untitled 2

```
9 FROM junior_da.orders
10 LIMIT 20;
11 --
12 SELECT DISTINCT payment_type
13 FROM junior_da.orders;
14 --
15 SELECT DISTINCT product
16 FROM junior_da.orders;
17 --
```

Query completed.

Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	product
1	Standard
2	Driver
3	Premium
4	MiniVan
5	Wagon
6	Evacuation
7	Kids
8	Econom
9	Business
10	CovidProtected

PERSONAL HISTORY PROJECT HISTORY

REFRESH

1.2.4. Написати ОДИН запит до таблиці замовлень, який виведе рядки, у яких:

повна сума замовлення більше 500 грн.
тип класу таксі - Стандарт.
тип оплати - Apple Pay та Google Pay.

Умови мають виконуватись одночасно!

Результуюча таблиця має містити наступні колонки:

- ID замовлення,
- ID водія,
- ID пасажирів,
- ID міста,
- час розміщення замовлення пасажиром,
- повна сума замовлення,
- тип класу таксі,
- тип оплати.

```
SELECT order_id
       ,driver_id
       ,rider_id
       ,city_id
       ,placed_at
       ,order_amount
       ,product
       ,payment_type
FROM junior_da.orders
WHERE order_amount > 500
AND product = 'Standard'
AND payment_type IN ('PAYMENT_TYPE_GOOGLE_PAY', 'PAYMENT_TYPE_APPLE_PAY');
```

Search (/) for resources, docs, products, and more Search

Untitled 2 RUN SAVE SHARE SCHEDULE MORE

```

16 FROM junior_da.orders;
17 --
18 SELECT order_id
19        ,driver_id
20        ,rider_id
21        ,city_id
22        ,placed_at
23        ,order_amount
24        ,product
25        ,payment_type
26 FROM junior_da.orders
27 WHERE order_amount > 500
28       AND product = 'Standard'
29       AND payment_type IN ('PAYMENT_TYPE_GOOGLE_PAY', 'PAYMENT_TYPE_APPLE_PAY');

```

Press Alt+F1 for Accessibility Options.

Query results SAVE RESULTS EXPLORE DATA ↕ ×

JOB INFORMATION **RESULTS** JSON EXECUTION DETAILS EXECUTION GRAPH **PREVIEW**

Row	order_id	driver_id	rider_id	city_id	placed_at	order_amount
1	02a8971c-2f95-597b-8d71-c45...	3ae4031f-d8f4-535b-a0f4-58c6...	0198166f-2ff0-5e16-8334-2722...	1010	2022-03-01 14:59:46 UTC	945
2	8421078d-3bcb-5109-a03e-c5c...	37b0b3f2-cfec-507a-b55b-be2...	b928ce0d-34a2-53bc-89dc-a1d...	1011	2022-05-17 15:10:09 UTC	616
3	af086e9a-3dad-5994-b8be-ed9...	null	bf68bc0f-09fd-5745-ae00-f04d...	1007	2022-02-24 07:16:41 UTC	534
4	abc198d9-8cfd-59d2-b7c3-c0e...	null	4c1158a2-74f2-5a6e-9863-ccf...	1006	2022-01-03 17:03:08 UTC	828

Results per page: 50 1 - 14 of 14 |< < > |>

PERSONAL HISTORY PROJECT HISTORY REFRESH ^

1.2.5. Виведіть усі значення з таблиці замовлень, де тип оплати містить слово “PAY” у кінці строки.

Результуюча таблиця має містити наступні колонки:

- ID замовлення,
- ID пасажирів,
- час розміщення замовлення пасажиром,
- повна сума замовлення,
- повна дистанція поїздки,
- тип оплати.

```

SELECT order_id
       ,rider_id
       ,placed_at
       ,order_amount
       ,route_distance
       ,product
       ,payment_type
FROM junior_da.orders
WHERE payment_type LIKE "%PAY";

```

Search (/) for resources, docs, products, and more

Search

Untitled 2

RUN

SAVE

SHARE

SCHEDULE

MORE

```

37 order_amount > 500
38 AND product = 'Standard'
39 AND payment_type IN ('PAYMENT_TYPE_GOOGLE_PAY',
40 'PAYMENT_TYPE_APPLE_PAY');
41 --
42 SELECT order_id
43 , rider_id
44 , placed_at
45 , order_amount
46 , route_distance
47 , product
48 , payment_type
49 FROM junior_da.orders
50 WHERE payment_type LIKE "%PAY";

```

Press Alt+F1 for Accessibility Options.

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	rider_id	placed_at	order_amount	route_distance	product	payment_type
1	d478f0ba-f6fb-592e-94cd-f514...	2022-01-01 03:47:27 UTC	102	6574	Standard	PAYMENT_TYPE_GOOGLE_PAY
2	8f0cb9fb-2737-5adc-bcb1-7da...	2022-01-01 05:16:12 UTC	170	12215	Standard	PAYMENT_TYPE_APPLE_PAY
3	43eaa9d2-c43e-5b0e-9778-56d...	2022-01-01 11:39:17 UTC	100	7588	Standard	PAYMENT_TYPE_APPLE_PAY
4	a0c7d86a-fbeb-5986-a46a-37c...	2022-01-01 13:33:11 UTC	102	7969	Standard	PAYMENT_TYPE_APPLE_PAY

Results per page: 50 1 - 50 of 235009

PERSONAL HISTORY PROJECT HISTORY

REFRESH

6. Напишіть ОДИН запит, який виведе 10 рядків з таблиці замовлень у місті з ідентифікатором — 1011 та типом класу таксі — Преміальний, відсортованих по повній сумі замовлень (від найбільшої суми до найменшої).

Тобто, топ-10 замовлень преміального таксі, у місті 1011.

Результуюча таблиця має містити наступні колонки:

- ID замовлення,
- ID водія,
- ID пасажирів,
- ID міста,
- час розміщення замовлення пасажиром,
- повна сума замовлення,
- тип класу таксі,
- тип оплати.

```

SELECT order_id
, driver_id
, rider_id
, city_id
, placed_at
, order_amount
, product
, payment_type

```

```

FROM junior_da.orders
WHERE city_id = 1011 AND product = 'Premium'
ORDER BY order_amount DESC
LIMIT 10;

```

Search (/) for resources, docs, products, and more

Search

Untitled 2

RUN SAVE SHARE SCHEDULE MORE

```

56 ,placed_at
57 ,order_amount
58 ,product
59 ,payment_type
60 FROM junior_da.orders
61 WHERE city_id = 1011 AND product = 'Premium'
62 ORDER BY order_amount DESC
63 LIMIT 10;

```

Press Alt+F1 for Accessibility Options.

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	driver_id	rider_id	city_id	placed_at	order_amount	product
1	1737-524a-8697-e84...	05f951e8-2fa9-5a2e-afa7-f8ed...	1011	2022-02-27 09:57:24 UTC	3019	Premium
2	607d-5690-a95d-8e9...	05f951e8-2fa9-5a2e-afa7-f8ed...	1011	2022-02-25 20:00:16 UTC	2520	Premium
3	-0e25-5197-a177-4a2...	05f951e8-2fa9-5a2e-afa7-f8ed...	1011	2022-02-27 08:44:07 UTC	2364	Premium
4	b3cf-5ad4-9183-9b7...	c55da984-1c21-5912-aa2b-d1e...	1011	2022-03-10 11:43:20 UTC	2164	Premium
5	5624-5263-ba1c-f7c...	05f951e8-2fa9-5a2e-afa7-f8ed...	1011	2022-02-27 09:00:21 UTC	2070	Premium
6	-a1af-5bc8-87f3-fb39...	495f2b41-1f9d-5417-bf5b-088...	1011	2022-03-04 17:29:29 UTC	1845	Premium
7	-89e7-5122-8a45-f98...	2fae592d-820b-538a-8d23-28b...	1011	2022-03-03 12:57:24 UTC	1399	Premium
8	b0bf-582d-8c14-782...	200f7640-e472-5db2-bc41-905...	1011	2022-02-26 22:08:35 UTC	1172	Premium
9	be8c-57bf-810d-740...	d16c5e22-c811-51fe-a450-20b...	1011	2022-03-07 16:09:34 UTC	1076	Premium
10	65fe-5282-86e1-a1e...	789b53e0-7f87-52b2-86de-b4e...	1011	2022-01-09 12:21:02 UTC	945	Premium

PERSONAL HISTORY PROJECT HISTORY REFRESH

2. Функції агрегації даних, аналітичні функції

Завдання 2.1

Доступ до проекту та БД

Google Cloud My First Project

Search (/) for resources, docs, products, and more

Search

Explorer

Viewing workspace resources.

SHOW STARRED ONLY

still-entity-383321

junior_da_personal

air_alerts_region

city

orders

weather

SHOW MORE

SHOW MORE

Untitled 2

RUN SAVE SHARE SCHEDULE MORE

```

1 create table junior_da_personal.city AS
2 (select * from dala-rdcc.junior_da.city);
3 create table junior_da_personal.orders AS
4 (select * from dala-rdcc.junior_da.orders);
5 create table junior_da_personal.weather AS
6 (select * from dala-rdcc.junior_da.weather);
7 create table junior_da_personal.air_alerts_region AS
8 (select * from dala-rdcc.junior_da.air_alerts_region);

```

elapsed time 18 sec

statements processed 4

JOB STATUS SUCCESS

Press Alt+F1 for Accessibility Options.

Завдання 2.2

2.2.1. Напишіть ОДИН запит, який виведе наступні показники:

- кількість замовлень всього
- кількість унікальних замовлень таксі
- кількість замовлень, які були скасовані (перевіряти по колонці cancelled_at)
- кількість замовлень, які були здійснені, та не мали знижки (перевіряти по колонці promo_amount)

Результуюча таблиця має містити 4 колонки, яким надати наступні назви:

- orders_cnt
- unique_orders_cnt
- cancelled_orders_cnt
- full_price_orders_cnt

Чи відрізняється кількість замовлень всього від кількості унікальних замовлень?

```
select count(order_id) as orders_cnt
      ,count(distinct order_id) as unique_orders_cnt
      ,countif(cancelled_at is not null) as cancelled_orders_cnt
      ,countif(promo_amount is null AND completed_at is not null) as full_price_orders_cnt
from junior_da_personal.orders;
```

Query results

Row	orders_cnt	unique_orders_cnt	cancelled_orders_cnt	full_price_orders_cnt
1	1365382	1365382	409591	765015

2.2.2. Напишіть ОДИН запит, який для кожного типу класу таксі, порахує наступні показники:

- * кількість унікальних замовлень
- * загальна сума усіх замовлень (total)
- * мінімальна сума замовлення
- * максимальна сума замовлення
- * середня вартість замовлень (округліть значення до 2х знаків після коми)

Врахуйте, що нас цікавлять показники тільки для здійснених замовлень та без знижок.

Відсортуйте дані за сумою максимального замовлення, за зменшенням.

Результуюча таблиця має містити 5 колонок, яким надати наступні назви:

- * тип класу таксі (поле product, без зміни назви)
- * unique_orders_cnt
- * total_orders_amount
- * min_order_amount
- * max_order_amount
- * average_order_amount

```
select product
      ,count(distinct order_id) as unique_orders_cnt
      ,sum(order_amount) as total_orders_amount
      ,min(order_amount) as min_order_amount
      ,max(order_amount) as max_order_amount
```



```

, round(avg(order_amount), 2) as average_order_amount
from junior_da_personal.orders
where completed_at is not null
and promo_amount is null
group by product;

```

ла... Gmail YouTube Карты Перекласти

Search (/) for resources, docs, products, and more Search

Untitled 2 RUN SAVE SHARE SCHEDULE MORE T

```

7 select product
8   , count(distinct order_id) as unique_orders_cnt
9   , sum(order_amount) as total_orders_amount
10  , min(order_amount) as min_order_amount
11  , max(order_amount) as max_order_amount
12  , round(avg(order_amount), 2) as average_order_amount
13 from junior_da_personal.orders
14 where completed_at is not null and promo_amount is null
15 group by product;

```

Query results SAVE RESULTS

Row	product	unique_orders_cnt	total_orders_amount	min_order_amount	max_order_amount	average_order_amount
1	Standard	611020	47653609	59	3214	77.99
2	Premium	50860	4316916	63	2164	84.88
3	Wagon	4333	391086	63	792	90.26
4	Driver	289	46287	106	415	160.16
5	MiniVan	2485	271392	72	1982	109.21
6	Evacuation	199	104053	60	13576	522.88
7	Kids	58	6031	68	499	103.98
8	CovidProtected	24	1741	60	158	72.54
9	Econom	94779	6947030	58	3069	73.3
10	Business	968	93852	72	270	96.95

PERSONAL HISTORY PROJECT HISTORY

2.2.3. Для покращення якості обслуговування пасажирів таксі, необхідно знайти ID водіїв, які часто скасовували замовлення:

- * Кількість замовлень, яку отримав водій — більше 10;
- * Відсоток скасованих замовлень — більше 70% (або більше 0.7).

Врахуйте, що існують замовлення таксі, для яких водій не був знайдений (їх треба відфільтрувати).

Напишіть ОДИН запит, який врахує попередні умови та порахує наступні показники, для кожного водія:

- * кількість отриманих замовлень
- * кількість скасованих замовлень
- * частку (або відсоток) скасованих замовлень

Відсортуйте дані за кількістю отриманих замовлень, за зменшенням.

Результуюча таблиця має містити 4 колонки, яким надати наступні назви:

- * ідентифікатор водія
- * orders_cnt
- * cancelled_orders_cnt
- * cancelled_orders_part

```

select driver_id
      ,count(order_id) as orders_cnt
      ,countif(cancelled_at is not null) as cancelled_orders_cnt
      ,round(countif(cancelled_at is not null) / count(order_id), 3) as cancelled_orders_part
from junior_da_personal.orders
where driver_id is not null
group by driver_id
having orders_cnt > 10 AND cancelled_orders_part > 0.7
order by orders_cnt DESC;

```

Query results

Row	driver_id	orders_cnt	cancelled_order	cancelled_order
1	ace338ab-e4cf-5097-8d5f-e2ef...	33	24	0.727
2	e55a3120-d9a7-570c-8811-f7f...	30	24	0.8
3	dc019404-e012-500a-9929-04...	16	14	0.875
4	b32014bf-86bc-5e34-a2fb-197...	13	11	0.846

2.2.4. Для проведення промо-акції по заохоченню вже існуючих пасажирів, необхідно вивантажити ID лояльних клієнтів:

- * Пасажири, у яких не було скасованих замовлень та ніколи не було знижки.
- * Зробили більше 100 замовлень АБО більше 10 замовлень з середньою сумою замовлення в 150грн та більше.

Результуюча таблиця має містити 3 колонки, яким надати наступні назви:

- * ідентифікатор пасажирів
- * кількість замовлень — orders_cnt
- * середня вартість замовлень (округліть значення до 2х знаків після коми) — average_order_amount

```

select rider_id
      ,count(order_id) as orders_cnt
      ,round(avg(order_amount), 2) as average_order_amount
from junior_da_personal.orders
where cancelled_at is null and promo_amount is null
group by rider_id
having orders_cnt > 100 or (orders_cnt > 10 and average_order_amount >= 150);

```


Завдання 3.1

3.1.1. Використовуючи SET оператор, напишіть ОДИН запит, який виведе з таблиць замовлень у Івано-Франківську (orders_if) та у Рівному (orders_rivne), наступні колонки:

- * ID замовлення,
- * повна сума замовлення,
- * тип класу таксі,
- * ID міста

? Питання: Скільки рядків має результуюча таблиця? Обґрунтуйте, чому саме так спрацював ваш SQL-запит.

```
select order_id
       ,order_amount
       ,product
       ,city_id
from junior_da_small.orders_if
UNION ALL
select order_id
       ,order_amount
       ,product
       ,city_id
from junior_da_small.orders_rivne;
```

- Результуюча таблиця має 6 рядків, адже ми об'єднали два запити, що мали по 3 рядки кожен з унікальними значеннями. В даному випадку, аналогічний результат ми отримаємо і при об'єднанні двох запитів за допомогою UNION DISTINCT, адже дублікатів не буде, бо всі записи є унікальними.

```
select order_id
       ,order_amount
       ,product
       ,city_id
from junior_da_small.orders_if
UNION DISTINCT
select order_id
       ,order_amount
       ,product
       ,city_id
from junior_da_small.orders_rivne;
```

2023-04-25 15:33:31 RUN SAVE SHARE SCHEDULE

```
21  --
22  select order_id
23  |       ,order_amount
24  |       ,product
25  |       ,city_id
26  | from junior_da_small.orders_if
27  | UNION DISTINCT
28  | select order_id
29  | |       ,order_amount
30  | |       ,product
31  | |       ,city_id
32  | from junior_da_small.orders_rivne;
33  --
```

Query results

SAVE RESULTS

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS		EXECUTION GRAPH	PRE
Row	order_id	order_amount	product	city_id			
1	e6d89d4a-519f-5d79-b3a4-939...	84	Standard	1009			
2	a6bc5a16-8d50-5629-a26c-6d4...	82	Standard	1009			
3	c65ffd74-36b6-58b2-806e-1b7...	85	Standard	1009			
4	e68b8689-df65-5fc8-9041-93f9...	107	Standard	1007			
5	4756e2de-047a-59b6-8d92-16...	101	Standard	1007			
6	0d1f3662-04d2-5112-ae1f-e12...	135	Standard	1007			

3.1.2. Використовуючи SET оператор, напишіть ОДИН запит, який виведе замовлення з міста Рівного, які присутні в таблиці orders, але відсутні в таблиці orders_rivne.

Ідентифікатор міста Рівне: 1009.


Результуюча таблиця має містити 2 колонки:


- * ID замовлення,
- * повна сума замовлення


? Питання: Скільки рядків має результуюча таблиця?


```
select order_id, order_amount
from junior_da_small.orders
where city_id = 1009
EXCEPT DISTINCT
select order_id, order_amount
from junior_da_small.orders_rivne;
```


- Результуюча таблиця має два рядки, оскільки ми фільтруємо першим запитом до 5 записів по Рівному з загальної таблиці orders, а потім віднімаємо три записи по Рівному у другому селекті з таблиці orders_rivne, тому в результаті в нас віднімаються дублікати і на виході 2 рядки результуючі, .


 2023-04-25 15:33:31

 RUN

 SAVE ▾

 SHARE ▾

 SCHEDULE ▾



```
31 |      ,city_id
32 | from junior_da_small.orders_rivne;
33 | --
34 | select order_id, order_amount
35 | from junior_da_small.orders
36 | where city_id = 1009
37 | EXCEPT DISTINCT
38 | select order_id, order_amount
39 | from junior_da_small.orders_rivne;|
40 | --
41 | select o.order_id
42 |      ,o.order_amount
43 |      .o.driver id
```

P

Query results

 SAVE RESULTS ▾

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREV
Row	order_id	order_amount				
1	c701ced9-8cc2-58a9-9653-76d...	92				
2	d536c0d5-639a-53ff-a45e-5fde...	94				

Завдання 3.2

3.2.1. Напишіть 2 запити, які виведуть з таблиці orders_if та city наступні колонки:

- * ID замовлення,
- * повна сума замовлення,
- * ID водія,
- * ID пасажирів,
- * ID міста (з таблиці orders_if),
- * назва міста (з таблиці city),

Запит №1 має поєднувати таблиці за допомогою внутрішнього об'єднання.

```
select o.order_id
      ,o.order_amount
      ,o.driver_id
      ,o.rider_id
      ,o.city_id
      ,c.city_name
from junior_da_small.orders_if as o
INNER JOIN junior_da_small.city as c ON o.city_id = c.city_id;
```

2023-04-25 15:33:31 RUN SAVE SHARE SCHEDULE MORE

```
38 select order_id, order_amount
39 from junior_da_small.orders_rivne;
40 --
41 select o.order_id
42       ,o.order_amount
43       ,o.driver_id
44       ,o.rider_id
45       ,o.city_id
46       ,c.city_name
47 from junior_da_small.orders_if as o
48 INNER JOIN junior_da_small.city as c ON o.city_id = c.city_id;
49 --
50 select o.order_id
```

Press Alt+F1 for Accessibility

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row		order_amount	driver_id	rider_id	city_id	city_name
1	...	107	c87bf631-4119-5861-8532-36b...	f7ff8710-59cc-5fe2-90f7-9e94e...	1007	Ivano-Frankivsk
2	...	101	c251b67f-fcfc-5a8f-bac9-8603...	974fb1e6-8612-561e-9023-1db...	1007	Ivano-Frankivsk
3	...	135	8bb81e53-5c64-58e0-8863-f46...	5e157c82-f1bf-57cb-a28a-cea...	1007	Ivano-Frankivsk

Запит №2 має поєднувати таблиці за допомогою зовнішнього об'єднання.

```
select o.order_id
      ,o.order_amount
      ,o.driver_id
      ,o.rider_id
      ,o.city_id
      ,c.city_name
from junior_da_small.orders_if as o
LEFT JOIN junior_da_small.city as c ON o.city_id = c.city_id;
```

2023-04-25 15:33:31
RUN
SAVE
SHARE
SCHEDULE
MORE

```

46      ,c.city_name
47 from junior_da_small.orders_if as o
48 INNER JOIN junior_da_small.city as c ON o.city_id = c.city_id;
49 --
50 select o.order_id
51      ,o.order_amount
52      ,o.driver_id
53      ,o.rider_id
54      ,o.city_id
55      ,c.city_name
56 from junior_da_small.orders_if as o
57 LEFT JOIN junior_da_small.city as c ON o.city_id = c.city_id;
58 --

```

Press Alt+F1 for Accessibility

Query results

[SAVE RESULTS](#)
[EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS		EXECUTION GRAPH	PREVIEW
Row	order_id	order_amount	driver_id	rider_id	city_id	city_name	
1	e68b8689-df65-...	107	c87bf631-4119-...	f7ff8710-59cc-5fe2-90-...	1007	Ivano-Frankivsk	
2	4756e2de-047a-...	101	c251b67f-fcfc-...	974fb1e6-8612-561e-9-...	1007	Ivano-Frankivsk	
3	0d1f3662-04d2-...	135	8bb81e53-5c6-...	5e157c82-f1bf-57cb-a-...	1007	Ivano-Frankivsk	

? Питання: Який тип зовнішнього об'єднання необхідно використати, щоб запит №1 та запит №2 повернули однакову результуючу таблицю?

- Для того, щоб запит №1 та запит №2 повернули однакову результуючу таблицю, необхідно використати такий тип зовнішнього об'єднання, як **LEFT JOIN**, адже ми об'єднаємо значення першої таблиці з назвами міст з другої таблиці завдяки колонці **city_id**.

3.2.2. Напишіть 3 запити, які виведуть з таблиці `orders_zp` та `city` наступні колонки:

- ID замовлення,
- повна сума замовлення,
- тип класу таксі,
- ID міста, з таблиці `orders_zp` (надайте назву `orders_city_id`)
- ID міста (з таблиці `city`) (надайте назву `city_city_id`)
- назва міста (з таблиці `city`)

☀ Запити мають поєднувати таблиці за допомогою зовнішнього об'єднання. Використайте різні типи зовнішнього об'єднання для кожного запиту.

```

1) select o.order_id
      ,o.order_amount
      ,o.product
      ,o.city_id as orders_city_id
      ,c.city_id as city_city_id
      ,c.city_name
from junior_da_small.orders_zp as o
LEFT JOIN junior_da_small.city as c ON o.city_id = c.city_id;

```


2023-04-25 15:33:31

RUN

SAVE

SHARE

SCHEDULE

MORE

```

58 --
59 select o.order_id
60       ,o.order_amount
61       ,o.product
62       ,o.city_id as orders_city_id
63       ,c.city_id as city_city_id
64       ,c.city_name
65 from junior_da_small.orders_zp as o
66 LEFT JOIN junior_da_small.city as c ON o.city_id = c.city_id;
67 --
68 select o.order_id
69       ,o.order_amount

```

Press Alt+F1 for Accessibility

Query results

SAVE RESULTS

EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS		EXECUTION GRAPH	PREVIEW
Row	order_id	order_amount	product	orders_city_id	city_city_id	city_name	
1	74fbe73f...	86	Standard	1006	1006	Zaporizhia	
2	50ccda63...	117	Standard	1006	1006	Zaporizhia	
3	3c55b1fc...	133	Undefined	1111	null	null	

```

2) select o.order_id
      ,o.order_amount
      ,o.product
      ,o.city_id as orders_city_id
      ,c.city_id as city_city_id
      ,c.city_name
from junior_da_small.orders_zp as o
RIGHT JOIN junior_da_small.city as c ON o.city_id = c.city_id;

```

2023-04-25 15:33:31
 ▶ RUN
📄 SAVE ▾
👤 SHARE ▾
🕒 SCHEDULE ▾
⚙️ MORE ▾

```

67  --
68  select o.order_id
69         ,o.order_amount
70         ,o.product
71         ,o.city_id as orders_city_id
72         ,c.city_id as city_city_id
73         ,c.city_name
74  from junior_da_small.orders_zp as o
75  RIGHT JOIN junior_da_small.city as c ON o.city_id = c.city_id;
76  --
77  select o.order_id
78         ,o.order_amount
    
```

Press Alt+F1 for Accessibility

Query results📄 SAVE RESULTS ▾📊 EXPLORE DATA ▾

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS		EXECUTION GRAPH	PREVIEW
Row	order_id	order_amount	product	orders_city_id	city_city_id	city_name	
1	null	null	null	null	1010	Khmelnyskyi	
2	null	null	null	null	1009	Rivne	
3	null	null	null	null	1007	Ivano-Frankivsk	
4	null	null	null	null	1008	Zhytomyr	
5	null	null	null	null	1011	Vinnysia	
6	74fbe73f-19eb-53e6-ae...	86	Standard	1006	1006	Zaporizhia	
7	50ccda63-25b2-5f66-a6...	117	Standard	1006	1006	Zaporizhia	

```

3) select o.order_id
        ,o.order_amount
        ,o.product
        ,o.city_id as orders_city_id
        ,c.city_id as city_city_id
        ,c.city_name
from junior_da_small.orders_zp as o
FULL JOIN junior_da_small.city as c ON o.city_id = c.city_id;
    
```

2023-04-25 15:33:31
 RUN
 SAVE
 SHARE
 SCHEDULE
 MORE

```

77 select o.order_id
78     ,o.order_amount
79     ,o.product
80     ,o.city_id as orders_city_id
81     ,c.city_id as city_city_id
82     ,c.city_name
83 from junior_da_small.orders_zp as o
84 FULL JOIN junior_da_small.city as c ON o.city_id = c.city_id;
85 --
86 select c.city_name
    
```

Press Alt+F1 for Accessibility

Query results

SAVE RESULTS

EXPLORE DATA

JOB INFORMATION

RESULTS

JSON

EXECUTION DETAILS

EXECUTION GRAPH

PREVIEW

Row	order_id	order_amount	product	orders_city_id	city_city_id	city_name
1	74fbe73f-19eb-53e6...	86	Standard	1006	1006	Zaporizhia
2	50ccda63-25b2-5f6...	117	Standard	1006	1006	Zaporizhia
3	null	null	null	null	1009	Rivne
4	3c55b1fc-68fd-593a...	133	Undefined	1111	null	null
5	null	null	null	null	1011	Vinnytsia
6	null	null	null	null	1010	Khmelnyskyi
7	null	null	null	null	1008	Zhytomyr
8	null	null	null	null	1007	Ivano-Frankivsk

? Питання: Який тип зовнішнього об'єднання поверне найбільшу кількість рядків у результуючій таблиці? Чому?

- Найбільшу кількість рядків у результуючій таблиці поверне такий тип зовнішнього об'єднання, як **FULL JOIN**. Адже **FULL JOIN** поєднує в собі як **LEFT JOIN**, так і **RIGHT JOIN**, адже **FULL JOIN** повертає записи обох таблиць, а де немає співпадиння по колонці **city_id**, то повертає нам **NULL**.

Завдання 3.3

3.3.1. Напишіть ОДИН запит, який для кожного міста, порахує наступні показники:

- кількість унікальних замовлень
- кількість здійснених замовлень
- загальна сума усіх замовлень (total)
- загальна сума здійснених замовлень

Відсортуйте дані за кількістю здійснених замовлень, за зменшенням.

Результуюча таблиця має містити 5 колонок, яким надати наступні назви:

- назва міста (з таблиці **city**)
- **unique_orders_cnt**
- **completed_orders_cnt**
- **total_order_amount**
- **completed_order_amount**

● Важливо: необхідно вивести саме **назву** міста, а НЕ його ідентифікатор.

```

select c.city_name
     ,count(distinct o.order_id) as unique_orders_cnt
     ,countif(o.completed_at is not null) as completed_orders_cnt
     ,sum(order_amount) as total_order_amount
    
```

```

, sum(if(o.completed_at is not null, o.order_amount, null)) as completed_order_amount
from junior_da.city as c
LEFT JOIN junior_da.orders as o ON c.city_id = o.city_id
group by c.city_name
order by completed_orders_cnt DESC;

```

2023-04-25 15:33:31	RUN	SAVE	SHARE	SCHEDULE	MORE
<pre> 85 -- 86 select c.city_name 87 , count(distinct o.order_id) as unique_orders_cnt 88 , countif(o.completed_at is not null) as completed_orders_cnt 89 , sum(order_amount) as total_order_amount 90 , sum(if(o.completed_at is not null, o.order_amount, null)) as completed_order_amount 91 from junior_da.city as c 92 LEFT JOIN junior_da.orders as o ON c.city_id = o.city_id 93 group by c.city_name 94 order by completed_orders_cnt DESC; </pre>					
Query results					
<div>SAVE RESULTS</div> <div>EXPLORE DATA</div>					
JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH
					PREVIEW
Row	city_name	unique_orders_cnt	completed_orders_cnt	total_order_amount	completed_order_amount
1	Zaporizhia	358860	262091	31725853	22370955
2	Vinnytsia	291241	222040	23811157	17395911
3	Ivano-Frankiv...	255742	198896	20257780	15102056
4	Rivne	184969	149134	13977729	10874805
5	Zhytomyr	142322	101246	11221112	7518415
6	Khmelnyskyi	132248	94177	10595776	7236602

? Питання: В якому місті було здійснено найбільше замовлень, найбільша сума здійснених замовлень?

- Найбільше замовлень було здійснено у м. Запоріжжя, в цьому ж місті найбільша сума здійснених замовлень.

3.3.2. Для кожного міста порахуйте кількість поїздок під час повітряних тривог та яку частку вони становлять від усіх поїздок.

Поїздка під час повітряної тривоги — це замовлення, яке було розміщене, коли повітряна тривога була ВЖЕ оголошена.

Результуюча таблиця має містити 3 колонки:

- * назва міста (з таблиці city)
- * кількість здійснених замовлень, які були розміщені під час повітряної тривоги (надайте назву alert_trips)
- * частка здійснених замовлень під час тривоги (alert_trips) від усіх здійснених поїздок (надайте назву alert_trips_part)

Врахуйте, що нас цікавлять тільки здійсненні замовлення.

SQL запит може виконуватись більше 1 хвилини — це нормально.

```

select c.city_name
      ,count(DISTINCT if(o.completed_at is not null and (o.placed_at >= a.start_date and
o.placed_at < a.end_date), order_id, null)) as alert_trips
      ,count(DISTINCT if(o.completed_at is not null and (o.placed_at >= a.start_date and
o.placed_at < a.end_date), order_id, null)) / count(DISTINCT if(o.completed_at is not null,
order_id, null)) as alert_trips_part
from junior_da.orders as o
LEFT JOIN junior_da.city as c ON o.city_id = c.city_id
LEFT JOIN junior_da.air_alerts_region as a ON o.city_id = a.city_id AND (o.placed_at >=
a.start_date AND o.placed_at < a.end_date)
group by c.city_name;

```

139	select c.city_name
140	,count(DISTINCT if(o.completed_at is not null and (o.placed_at >= a.start_date and o.placed_at < a.end_date), order_id, null)) as alert_trips
141	,count(DISTINCT if(o.completed_at is not null and (o.placed_at >= a.start_date and o.placed_at < a.end_date), order_id, null)) / count
	(DISTINCT if(o.completed_at is not null, order_id, null)) as alert_trips_part
142	from junior_da.orders as o
143	LEFT JOIN junior_da.city as c ON o.city_id = c.city_id
144	LEFT JOIN junior_da.air_alerts_region as a ON o.city_id = a.city_id AND (o.placed_at >= a.start_date AND o.placed_at < a.end_date)
145	group by c.city_name;
146	

Query results		SAVE RESULTS	EXPLORE DATA	
JOB INFORMATION	RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH PREVIEW
Row	city_name	alert_trips	alert_trips_part	
1	Zaporizhia	16430	0.062688150298941966	
2	Ivano-Frankivsk	6286	0.0316044566004344	
3	Zhytomyr	5250	0.051853900401003496	
4	Rivne	4535	0.030408894014778656	
5	Khmelnyskyi	3184	0.0338086794015524	
6	Vinnytsia	9807	0.044167717528373264	

4. Типи даних в SQL та їх перетворення

Завдання 4.1

4.1.1. Напишіть ОДИН запит, який виведе з таблиці замовлень(`orders`) наступні колонки:

- ID замовлення,
- повна сума замовлення,
- дата розміщення замовлення,
- час розміщення замовлення,
- номер тижня, коли було розміщене замовлення,
- порахуйте час очікування, поки для замовлення буде знайдено водія, в секундах (по колонці `accepted_at`),
- порахуйте тривалість поїздки в хвилинах (по колонці `completed_at`).

☀ *Врахуйте, що нас цікавлять тільки здійснені замовлення.*

Результуюча таблиця має містити 7 колонок:

- ID замовлення,
- повна сума замовлення,
- `placed_at_date`,
- `placed_at_time`,
- `week_number`,
- `waiting_time`,
- `total_trip_time`.

```

SELECT order_id
,order_amount
,date(placed_at) as placed_at_date
,time(placed_at) as placed_at_time
,EXTRACT(ISO WEEK FROM placed_at) as week_number
,DATETIME_DIFF(accepted_at, placed_at, SECOND) as waiting_time
,DATETIME_DIFF(completed_at, accepted_at, MINUTE) as total_trip_time
FROM junior_da.orders
WHERE completed_at IS NOT NULL;

```

```

157 SELECT order_id
158 ,order_amount
159 ,date(placed_at) as placed_at_date
160 ,time(placed_at) as placed_at_time
161 ,EXTRACT(ISO WEEK FROM placed_at) as week_number
162 ,DATETIME_DIFF(accepted_at, placed_at, SECOND) as waiting_time
163 ,DATETIME_DIFF(completed_at, accepted_at, MINUTE) as total_trip_time
164 FROM junior_da.orders
165 WHERE completed_at IS NOT NULL;

```

Press Alt

Query results

[SAVE RESULTS](#)

[EXI](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS		EXECUTION GRAPH	PREVIEW
Row	order_id	order_amount	placed_at_date	placed_at_time	week_number	waiting_time	total_trip_time
1	454d9c4f-9834-54f8-a665-b4d...	107	2022-05-31	13:09:24	22	1180	22
2	5ec2e9b7-1d41-5973-a735-ad9...	101	2022-05-31	13:14:02	22	38	23
3	dbb758fa-9f5e-510a-a326-db8...	76	2022-05-31	13:56:19	22	225	9
4	73d17fa1-990a-5bd4-8bd1-4bc...	106	2022-05-31	14:00:36	22	43	27
5	97486d2e-27db-555d-bea0-25...	89	2022-05-31	14:16:53	22	32	12
6	f37a7e55-12ac-5834-9595-9e2...	117	2022-05-31	14:17:30	22	177	21
7	11bca4b3-cfaf-548e-aa28-4ed...	67	2022-05-31	14:19:46	22	68	487
8	3b7dd22b-80b2-55a3-81e3-3c...	119	2022-05-31	14:37:36	22	466	33

Results per page: 50 1 – 50 of 1027584

4.1.2. Напишіть ОДИН запит, який для кожного місяця та кожного міста виведе кількість замовлень.

Відсортуйте дані за назвою міста та місяцем, за збільшенням.

Результуюча таблиця має містити 3 колонки:

- о назва міста,
- о місяць розміщення замовлень — дата у форматі 2022-05-01 (надайте назву orders_month),
- о кількість замовлень (надайте назву orders_cnt).

● Врахуйте, що нас цікавлять тільки здійснені замовлення

```

SELECT c.city_name
,DATE(DATE_TRUNC(placed_at, month)) as orders_month
,COUNT(o.order_id) as orders_cnt
FROM junior_da.orders as o
LEFT JOIN junior_da.city as c ON o.city_id = c.city_id
WHERE completed_at IS NOT NULL
GROUP BY c.city_name, orders_month
ORDER BY c.city_name ASC, orders_month ASC;

```

1
SELECT c.city_name
2
,DATE(Date_Trunc(placed_at, month)) as orders_month
3
,COUNT(o.order_id) as orders_cnt
4
FROM junior_da.orders as o
5
LEFT JOIN junior_da.city as c ON o.city_id = c.city_id
6
WHERE completed_at IS NOT NULL
7
GROUP BY c.city_name, orders_month
8
ORDER BY c.city_name ASC, orders_month ASC;

Press

Query results

SAVE RESULTS

Job Information	RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	city_name	orders_month	orders_cnt	
1	Ivano-Frankivsk	2022-01-01	41815	
2	Ivano-Frankivsk	2022-02-01	34178	
3	Ivano-Frankivsk	2022-03-01	24359	
4	Ivano-Frankivsk	2022-04-01	32539	
5	Ivano-Frankivsk	2022-05-01	33369	

Results per page: 50 1 – 36 of 36

Завдання 4.2

4.2.1. Перевірте як працюють функції CAST () та SAFE_CAST ().

Для цього, напишіть 2 запити до таблиці orders, які спробують перетворити колонку product на тип даних INT64.

```
SELECT CAST(product AS INT64) as cast_result
FROM junior_da.orders;
```

176
SELECT CAST(product AS INT64) as cast_result
177
FROM junior_da.orders;
178
--
179
SELECT SAFE_CAST(product AS INT64) as safe_cast_result
180
FROM junior_da.orders;

Press All

Query results

SAVE RESULTS

EXI

Job Information	RESULTS	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
<div> <div></div> <div>Bad int64 value: Standard</div> </div>				

```
SELECT SAFE_CAST(product AS INT64) as safe_cast_result
FROM junior_da.orders;
```

```

171 SELECT COUNT(junior_da.city_id) AS c ON c.city_id = c.city_id
172 WHERE completed_at IS NOT NULL
173 GROUP BY c.city_name, orders_month
174 ORDER BY c.city_name ASC, orders_month ASC;
175 --
176 SELECT CAST(product AS INT64) AS cast_result
177 FROM junior_da.orders;
178 --
179 SELECT SAFE_CAST(product AS INT64) AS safe_cast_result
180 FROM junior_da.orders;

```

Query results [SAVE RESULTS](#) [EXI](#)

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH **PREVIEW**

Row	safe_cast_result
1	null
2	null
3	null
4	null
5	null
6	null
7	null
8	null

Results per page: 50 1 – 50 of 1365382

? Питання: Які результати ви отримали? На які типи даних можна перетворювати стовпець, який містить `STRING` дані? Коли це може бути корисно?

- Перший запит з використанням функції `CAST()` для перетворення колонки `product` на тип даних `INT64` повернув помилку, а використання функції `SAFE_CAST()` для аналогічної мети повернув `NULL` значення для кожного рядка у колонці `product`.
- Згідно документації, колонку, яка містить `STRING` дані можна перетворити лише на `STRING` формат, але можна перетворити на інший тип даних, якщо в `STRING` даних вміщені такі "лантентні" типи даних, які потрібно витягти із `STRING`:
 - `BOOL`
 - `INT64`
 - `NUMERIC`
 - `BIGNUMERIC`
 - `FLOAT64`
 - `STRING`
 - `BYTES`
 - `DATE`
 - `DATETIME`
 - `TIME`
 - `TIMESTAMP`
- Функції `CAST()` та `SAFE_CAST()` можуть бути корисними, наприклад, коли в нас є числові значення, але вони збережені у форматі `STRING`, але ми хочемо перетворити їх на числовий формат (`INT64`: `SELECT CAST("35" AS INT64)`), або, наприклад, коли у нас дата або час записані у форматі `STRING`, але ми хочемо перетворити їх на формат дати або часу (`DATE/TIME`).

Завдання 4.3

4.3.1. Напишіть ОДИН запит, який розділить інформацію зі стовпця `DROPOFF_POINT` з таблиці `orders` по окремим стовпцям, згідно умов:

- значення `type` (без лапок)
- значення координати у вигляді масиву
- значення першої координати, у вигляді числа
- значення другої координати, у вигляді числа

Результуюча таблиця має містити 5 колонок, яким надати наступні назви:

- географічна точка місця висадки пасажирів (стовпець `DROPOFF_POINT` у початковому вигляді)
- `point_type`

- coordinates_array
- coordinate_a
- coordinate_b

```
SELECT dropoff_point
,JSON_VALUE(dropoff_point, '$.type') as point_type
,JSON_QUERY_ARRAY(dropoff_point, '$.coordinates') as coordinates_array
,JSON_QUERY(dropoff_point, '$.coordinates[0]') as coordinate_a
,JSON_QUERY(dropoff_point, '$.coordinates[1]') as coordinate_b
FROM junior_da_small.orders;
```

```
193 SELECT dropoff_point
194 ,JSON_VALUE(dropoff_point, '$.type') as point_type
195 ,JSON_QUERY_ARRAY(dropoff_point, '$.coordinates') as coordinates_array
196 ,JSON_QUERY(dropoff_point, '$.coordinates[0]') as coordinate_a
197 ,JSON_QUERY(dropoff_point, '$.coordinates[1]') as coordinate_b
198 FROM junior_da_small.orders;
199
200
```

Press Alt+F1 for Accessibility

Query results

[SAVE RESULTS](#)

[EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	dropoff_point	point_type	coordinates_array	coordinate_a	coordinate_b	
1	{ "coordinates": [2.4688830000000000e+01, 4.8926610000000000e+01	Point	2.4688830000000000e+01	2.4688830000000000e+01	4.8926610000000000e+01	
2	{ "coordinates": [2.4699830000000000e+01, 4.8945500000000000e+01	Point	2.4699830000000000e+01	2.4699830000000000e+01	4.8945500000000000e+01	

Results per page: 50 1 - 30 of 30

5. Підзапити, CTE, View

Завдання 5.1

5.1.1. Напишіть ОДИН запит, який виведе з таблиці замовлень(`orders`) усі замовлення, вартість яких вище середньої вартості замовлення.

🟡 Врахуйте, що нас цікавлять тільки здійснені замовлення у місті Запоріжжя, за березень місяць.

Перевірте себе

- Використовуйте підзапити.
- Умова для фільтрації має бути застосована як до основного запиту, так і до підзапиту.

Результуюча таблиця має містити 4 колонки:

- ID замовлення,
- повна сума замовлення,
- дату розміщення замовлення (надайте назву `placed_at_date`),
- ID міста.

```
SELECT order_id
,order_amount
,DATE(placed_at) AS placed_at_date
,city_id
FROM junior_da.orders
WHERE order_amount > (SELECT AVG(order_amount)
FROM junior_da.orders
WHERE completed_at IS NOT NULL
AND EXTRACT(MONTH FROM DATE(placed_at)) = 3
AND city_id = 1006)
AND completed_at IS NOT NULL
AND EXTRACT(MONTH FROM DATE(placed_at)) = 3
AND city_id = 1006
ORDER BY order_amount;
```

2023-04-25 15:33:31 RUN SAVE SHARE SCHEDULE MORE

```

201 --
202 SELECT order_id
203 ,order_amount
204 ,DATE(placed_at) AS placed_at_date
205 ,city_id
206 FROM junior_da.orders
207 WHERE order_amount > (SELECT AVG(order_amount)
208                        FROM junior_da.orders
209                        WHERE completed_at IS NOT NULL
210                        AND EXTRACT(MONTH FROM DATE(placed_at)) = 3
211                        AND city_id = 1006)
212 AND completed_at IS NOT NULL
213 AND EXTRACT(MONTH FROM DATE(placed_at)) = 3
214 AND city_id = 1006
215 ORDER BY order_amount;
216 --

```

Press Alt+F1 for Accessibility Options.

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	order_id	order_amount	placed_at_date	city_id		
1	b0945c67-09a9-59ec-9ce0-5e5...	87	2022-03-01	1006		
2	f73fde39-3775-5a8c-9905-2de...	87	2022-03-01	1006		
3	eb3aedbe-248c-5aab-a746-821...	87	2022-03-01	1006		

Results per page: 50 1 - 50 of 8870

5.1.2. Напишіть ОДИН запит, який порахує кількість водіїв які використовували 1, 2, 3, тощо автомобілів, за увесь період. Тобто, 1 автомобіль - 10 водіїв - 50%, 2 авто - 7 водіїв - 35%, 3 авто - 3 водії - 15%.

Відсортуйте дані за кількістю автомобілів, за збільшенням.

Результуюча таблиця має містити 3 колонки:

- кількість автомобілів (надайте назву cars_amount),
- кількість водіїв (надайте назву drivers_cnt),
- відсоток водіїв від загальної кількості (drivers_percent).

Врахуйте, що ID водіїв (driver_id) та ID автомобілів (vehicle_id) в таблиці orders неунікальні.

```

SELECT cars_amount
      ,COUNT(driver_id) AS drivers_cnt
      ,COUNT(driver_id) / (SELECT COUNT(DISTINCT driver_id) FROM junior_da.orders) AS
drivers_percent
FROM (SELECT driver_id
      ,COUNT(DISTINCT vehicle_id) AS cars_amount
      FROM junior_da.orders
      WHERE driver_id IS NOT NULL
      GROUP BY driver_id)
GROUP BY cars_amount
ORDER BY cars_amount;

```

```

274 --загальна кількість водіїв за групами кількості машин
275 SELECT cars_amount
276      ,COUNT(driver_id) AS drivers_cnt
277      ,COUNT(driver_id) / (SELECT COUNT(DISTINCT driver_id) FROM junior_da.orders) AS drivers_percent
278 FROM (SELECT driver_id
279      ,COUNT(DISTINCT vehicle_id) AS cars_amount
280      FROM junior_da.orders
281      WHERE driver_id IS NOT NULL
282      GROUP BY driver_id)
283 GROUP BY cars_amount
284 ORDER BY cars_amount;
285 --

```

Press Alt+F1 for Accessibility Options.

Query results SAVE RESULTS EXPLORE DATA

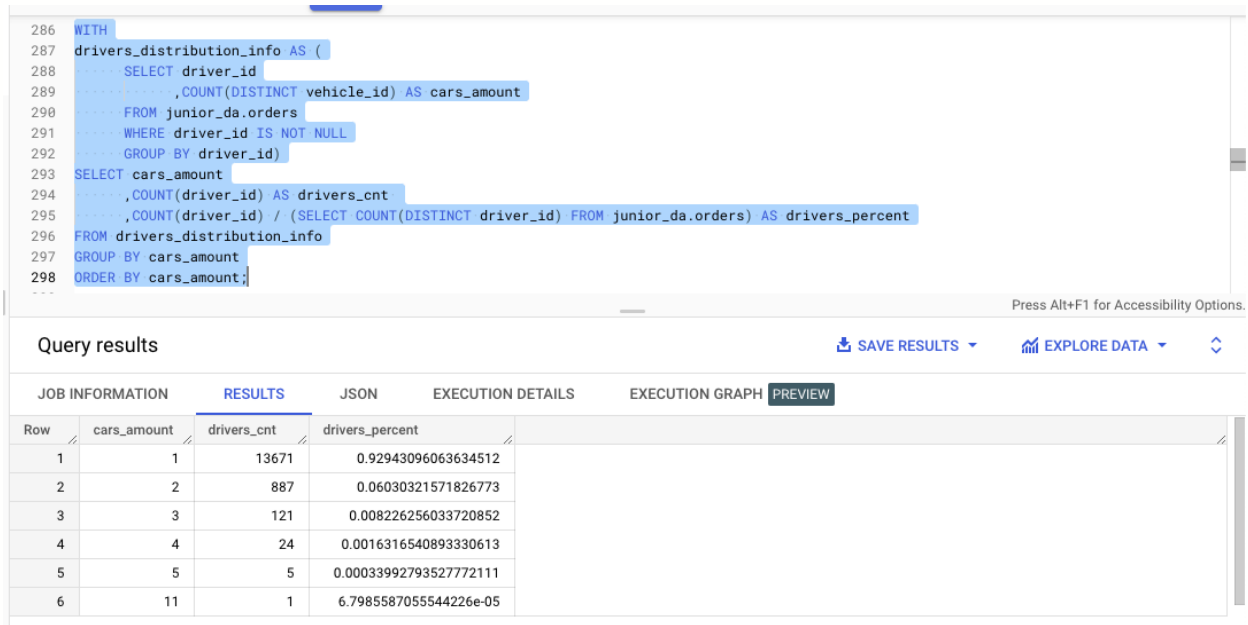
JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	cars_amount	drivers_cnt	drivers_percent			
1	1	13671	0.92943096063634512			
2	2	887	0.06030321571826773			
3	3	121	0.008226256033720852			
4	4	24	0.0016316540893330613			
5	5	5	0.0003399279352772111			
6	11	1	6.7985587055544226e-05			

Завдання 5.2

5.2.1. Перепишіть запит із минулого завдання (1.2) на SQL-запит, який використовує CTE.

WITH

```
drivers_distribution_info AS (  
    SELECT driver_id  
           ,COUNT(DISTINCT vehicle_id) AS cars_amount  
    FROM junior_da.orders  
    WHERE driver_id IS NOT NULL  
    GROUP BY driver_id)  
SELECT cars_amount  
       ,COUNT(driver_id) AS drivers_cnt  
       ,COUNT(driver_id) / (SELECT COUNT(DISTINCT driver_id) FROM junior_da.orders) AS  
drivers_percent  
FROM drivers_distribution_info  
GROUP BY cars_amount  
ORDER BY cars_amount;
```



```
286 WITH  
287 drivers_distribution_info AS (  
288     SELECT driver_id  
289           ,COUNT(DISTINCT vehicle_id) AS cars_amount  
290     FROM junior_da.orders  
291     WHERE driver_id IS NOT NULL  
292     GROUP BY driver_id)  
293 SELECT cars_amount  
294       ,COUNT(driver_id) AS drivers_cnt  
295       ,COUNT(driver_id) / (SELECT COUNT(DISTINCT driver_id) FROM junior_da.orders) AS drivers_percent  
296 FROM drivers_distribution_info  
297 GROUP BY cars_amount  
298 ORDER BY cars_amount;
```

Query results

[SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	cars_amount	drivers_cnt	drivers_percent			
1	1	13671	0.92943096063634512			
2	2	887	0.06030321571826773			
3	3	121	0.008226256033720852			
4	4	24	0.0016316540893330613			
5	5	5	0.00033992793527772111			
6	11	1	6.7985587055544226e-05			

Завдання 5.3

5.3.1. Необхідно поділити пасажирів на поведінкові групи, в залежності від:
кількості здійснених замовлень

- 1 замовлення
- (1;5] замовлень
- (5;10] замовлень
- (10;20] замовлень
- (20;50] замовлень
- більше 50 замовлень

реальної суми витраченої на замовлення таксі

- до 100 грн
- (100;500] грн
- (500;1000] грн
- (1000;3000] грн
- (3000;5000] грн
- більше 5000 грн

кількості днів використання сервісу таксі

- 1 день
- (1;7] днів
- (7;14] днів
- (14;30] днів
- (30;60] днів
- більше 60 днів

Кожній групі надайте відповідні назви, у форматі:

- (5;10] замовлень — '5-10 orders'
- (500;1000] грн — '500-1000 hrn'
- (1;7] днів — '1-7days'

● Врахуйте:

- для всіх метрик нас цікавлять тільки здійснені замовлення;
- витрачену суму необхідно оцінювати по реальній сумі замовлення, тобто від повної суми замовлення відняти суму знижки, якщо вона була;
- якщо замовлення було зроблене 2 рази в один і той самий день, то рахуємо це як 1 день.

● Використайте CTE при написанні запиту.

Результуюча таблиця має містити 4 колонки:

- ID пасажирів
- група по кількості здійснених замовлень (надайте назву orders_group)
- група по реальній витраченій сумі грошей (надайте назву payment_group)
- група по кількості днів (надайте назву days_group)

WITH

```
diff_groups AS (  
    SELECT rider_id  
        ,COUNT(DISTINCT order_id) AS orders_cnt  
        ,SUM(order_amount) AS sum_amount  
        ,COUNT(DISTINCT (DATE(completed_at))) AS days_cnt  
    FROM junior_da.orders  
    WHERE completed_at IS NOT NULL  
    GROUP BY rider_id)  
SELECT rider_id  
    ,CASE  
        WHEN orders_cnt = 1 THEN "1 order"  
        WHEN orders_cnt <= 5 THEN "1-5 orders"  
        WHEN orders_cnt <= 10 THEN "5-10 orders"  
        WHEN orders_cnt <= 20 THEN "10-20 orders"  
        WHEN orders_cnt <= 50 THEN "20-50 orders"  
        WHEN orders_cnt > 50 THEN "50+ orders"  
    END AS orders_group  
    ,CASE  
        WHEN sum_amount <= 100 THEN "less or equal 100 hrn"  
        WHEN sum_amount <= 500 THEN "100-500 hrn"  
        WHEN sum_amount <= 1000 THEN "500-1000 hrn"  
        WHEN sum_amount <= 3000 THEN "1000-3000 hrn"  
        WHEN sum_amount <= 5000 THEN "3000-5000 hrn"  
        WHEN sum_amount > 5000 THEN "5000+ hrn"  
    END AS payment_group  
    ,CASE  
        WHEN days_cnt = 1 THEN "1 day"  
        WHEN days_cnt <= 7 THEN "1-7 days"  
        WHEN days_cnt <= 14 THEN "7-14 days"  
        WHEN days_cnt <= 30 THEN "14-30 days"  
        WHEN days_cnt <= 60 THEN "30-60 days"  
        WHEN days_cnt > 60 THEN "60+ days"  
    END AS days_group  
FROM diff_groups;
```

```

445 WITH
446 diff_groups AS (
447     SELECT rider_id
448           ,COUNT(DISTINCT order_id) AS orders_cnt
449           ,SUM(order_amount) AS sum_amount
450           ,COUNT(DISTINCT (DATE(completed_at))) AS days_cnt
451     FROM junior_da.orders
452    WHERE completed_at IS NOT NULL
453    GROUP BY rider_id)
454 SELECT rider_id
455        ,CASE
456            WHEN orders_cnt = 1 THEN "1 order"
457            WHEN orders_cnt <= 5 THEN "1-5 orders"
458            WHEN orders_cnt <= 10 THEN "5-10 orders"
459            WHEN orders_cnt <= 20 THEN "10-20 orders"

```

Press Alt+F1 for Accessibility Options.

Query results [SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	rider_id	orders_group	payment_group	days_group		
1	f6b4d4e9-8391-5c69-b878-927...	50+ orders	3000-5000 hrn	30-60 days		
2	1a6f70ba-e354-549c-889a-853...	50+ orders	3000-5000 hrn	30-60 days		
3	2d34c0d3-c201-59b3-8cc4-8c6...	50+ orders	3000-5000 hrn	30-60 days		
4	39e6c99a-a614-5dca-9663-b93...	50+ orders	3000-5000 hrn	30-60 days		

Results per page: 50 1 - 50 of 245150

5.3.2. Створіть подання `rider_behavioural_groups`, яке буде складатись з відповідного SQL-запиту з попереднього завдання 5.3.1).

```

CREATE VIEW junior_da_personal.rider_behavioural_groups AS (
WITH
riders_info AS (
    SELECT rider_id
           ,COUNT(DISTINCT order_id) AS orders_cnt
           ,SUM(order_amount) AS sum_amount
           ,COUNT(DISTINCT (DATE(completed_at))) AS days_cnt
    FROM junior_da_personal.orders
   WHERE completed_at IS NOT NULL
   GROUP BY rider_id)
SELECT rider_id
       ,CASE
           WHEN orders_cnt = 1 THEN "1 order"
           WHEN orders_cnt <= 5 THEN "1-5 orders"
           WHEN orders_cnt <= 10 THEN "5-10 orders"
           WHEN orders_cnt <= 20 THEN "10-20 orders"
           WHEN orders_cnt <= 50 THEN "20-50 orders"
           WHEN orders_cnt > 50 THEN "50+ orders"
       END AS orders_group
       ,CASE
           WHEN sum_amount <= 100 THEN "less or equal 100 hrn"
           WHEN sum_amount <= 500 THEN "100-500 hrn"
           WHEN sum_amount <= 1000 THEN "500-1000 hrn"
           WHEN sum_amount <= 3000 THEN "1000-3000 hrn"
           WHEN sum_amount <= 5000 THEN "3000-5000 hrn"
           WHEN sum_amount > 5000 THEN "5000+ hrn"
       END AS payment_group
       ,CASE
           WHEN days_cnt = 1 THEN "1 day"
           WHEN days_cnt <= 7 THEN "1-7 days"
           WHEN days_cnt <= 14 THEN "7-14 days"
           WHEN days_cnt <= 30 THEN "14-30 days"
           WHEN days_cnt <= 60 THEN "30-60 days"
           WHEN days_cnt > 60 THEN "60+ days"
       END AS days_group
FROM riders_info);

```

```
1 CREATE VIEW junior_da_personal.rider_behavioural_groups AS (  
2 WITH  
3   riders_info AS (  
4     SELECT rider_id  
5           ,COUNT(DISTINCT order_id) AS orders_cnt  
6           ,SUM(order_amount) AS sum_amount  
7           ,COUNT(DISTINCT (DATE(completed_at))) AS days_cnt  
8     FROM junior_da_personal.orders  
9     WHERE completed_at IS NOT NULL  
10    GROUP BY rider_id)  
11 SELECT rider_id  
12    ,CASE  
13      WHEN orders_cnt = 1 THEN "1 order"
```

Query results

JOB INFORMATION RESULTS EXECUTION DETAILS EXECUTION GRAPH PREVIEW

This statement created a new view named rider_behavioural_groups. [GO TO VIEW](#)

```
SELECT *  
FROM junior_da_personal.rider_behavioural_groups;
```

```
28    ,CASE  
29      WHEN days_cnt = 1 THEN "1 day"  
30      WHEN days_cnt <= 7 THEN "1-7 days"  
31      WHEN days_cnt <= 14 THEN "7-14 days"  
32      WHEN days_cnt <= 30 THEN "14-30 days"  
33      WHEN days_cnt <= 60 THEN "30-60 days"  
34      WHEN days_cnt > 60 THEN "60+ days"  
35    END AS days_group  
36 FROM riders_info);  
37 --  
38 SELECT *  
39 FROM junior_da_personal.rider_behavioural_groups;  
40
```

Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	rider_id	orders_group	payment_group	days_group
1	b86b4b41-ce44-5980-adda-b3...	50+ orders	3000-5000 hrn	30-60 days
2	fa010a0c-c234-55e5-9568-b01...	50+ orders	3000-5000 hrn	30-60 days
3	7f2d44f4-4f76-5924-bc4b-d791...	50+ orders	5000+ hrn	60+ days
4	ebca762f-f625-567a-a1a7-d03...	50+ orders	5000+ hrn	60+ days
5	c94c5a10-41b2-568c-a0c9-b9a...	50+ orders	3000-5000 hrn	30-60 days
6	eeec696-1bbe-5f38-81f4-85b...	50+ orders	3000-5000 hrn	30-60 days

Results per page: 50 1 - 50 of 245150

6. Віконні функції

Завдання 6.1

6.1.1. Таблиця про погоду містить інформацію про початковий час прогнозу погоди – `REFERENCE_TIME_UTC`, але час, коли актуальність даного прогнозу закінчується – невідомий. Напишіть ОДИН запит, який виведе з таблиці погоди (`weather`) період актуальності прогнозу погоди – початок та кінець, а також ідентифікатор міста.

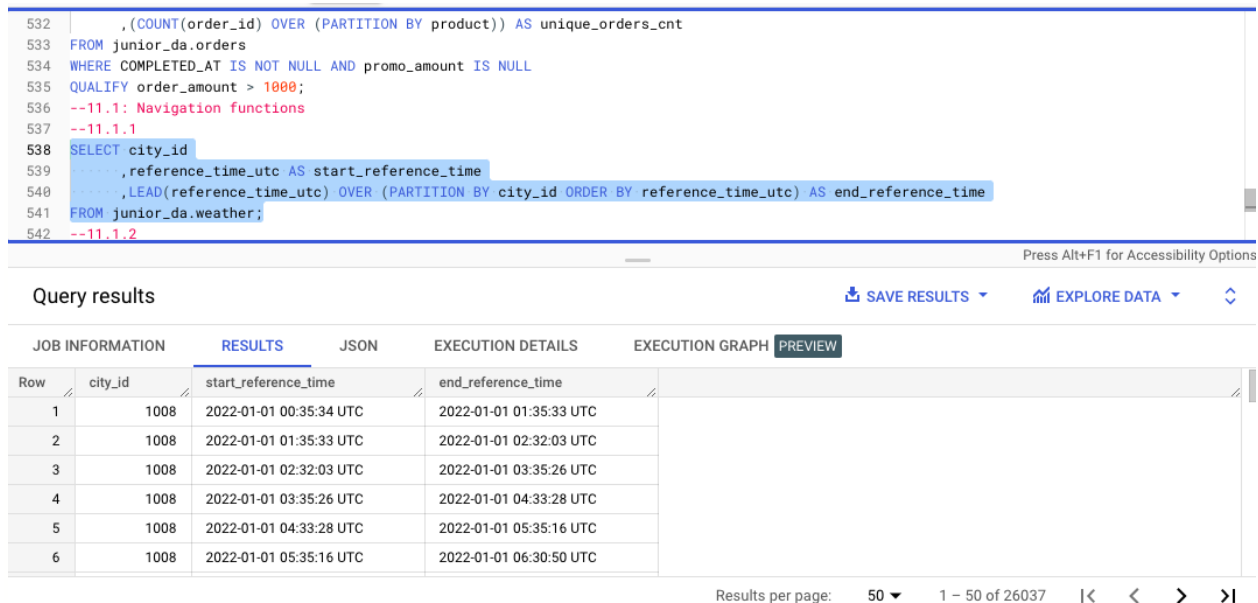
Пояснення

- Для того, щоб знайти кінець актуальності прогнозу погоди, треба знайти наступний час старту прогнозу (`REFERENCE_TIME_UTC`) за допомогою віконних функцій.

Результуюча таблиця має містити 3 колонки:

- ID міста,
- початковий час прогнозу погоди (надайте назву `start_reference_time`),
- кінцевий час прогнозу погоди (надайте назву `end_reference_time`).

```
SELECT city_id
       ,reference_time_utc AS start_reference_time
       ,LEAD(reference_time_utc) OVER (PARTITION BY city_id ORDER BY reference_time_utc) AS
end_reference_time
FROM junior_da.weather;
```



```
532      ,(COUNT(order_id) OVER (PARTITION BY product)) AS unique_orders_cnt
533 FROM junior_da.orders
534 WHERE COMPLETED_AT IS NOT NULL AND promo_amount IS NULL
535 QUALIFY order_amount > 1000;
536 --11.1: Navigation functions
537 --11.1.1
538 SELECT city_id
539        ,reference_time_utc AS start_reference_time
540        ,LEAD(reference_time_utc) OVER (PARTITION BY city_id ORDER BY reference_time_utc) AS end_reference_time
541 FROM junior_da.weather;
542 --11.1.2
```

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	city_id	start_reference_time	end_reference_time			
1	1008	2022-01-01 00:35:34 UTC	2022-01-01 01:35:33 UTC			
2	1008	2022-01-01 01:35:33 UTC	2022-01-01 02:32:03 UTC			
3	1008	2022-01-01 02:32:03 UTC	2022-01-01 03:35:26 UTC			
4	1008	2022-01-01 03:35:26 UTC	2022-01-01 04:33:28 UTC			
5	1008	2022-01-01 04:33:28 UTC	2022-01-01 05:35:16 UTC			
6	1008	2022-01-01 05:35:16 UTC	2022-01-01 06:30:50 UTC			

Results per page: 50 1 – 50 of 26037

6.1.2. Напишіть ОДИН запит, який для кожного замовлення у пасажирів порахує кількість пройденого часу між поточним та наступним замовленнями.

Результуюча таблиця має містити 6 колонок:

- ID замовлення,
- ID пасажирів,
- час розміщення замовлення,
- час розміщення наступного замовлення (надайте назву `next_placed_at`),
- різниця між наступним та поточним замовленням, в хвилинах (надайте назву `diff_minutes`),
- різниця між наступним та поточним замовленням, в днях (надайте назву `diff_days`).

```

SELECT order_id
      ,rider_id
      ,placed_at
      ,next_placed_at
      ,TIMESTAMP_DIFF(next_placed_at, placed_at, MINUTE) AS diff_minutes
      ,DATE_DIFF(DATE(next_placed_at), DATE(placed_at), DAY) AS diff_days
FROM (SELECT order_id
      ,rider_id
      ,placed_at
      ,LEAD(placed_at) OVER (PARTITION BY rider_id ORDER BY placed_at) AS
next_placed_at
      FROM junior_da.orders)
ORDER BY rider_id;

```

549SELECT order_id

550 ,rider_id

551 ,placed_at

552 ,next_placed_at

553 ,TIMESTAMP_DIFF(next_placed_at, placed_at, MINUTE) AS diff_minutes

554 ,DATE_DIFF(DATE(next_placed_at), DATE(placed_at), DAY) AS diff_days

555FROM (SELECT order_id

556 ,rider_id

557 ,placed_at

558 ,LEAD(placed_at) OVER (PARTITION BY rider_id ORDER BY placed_at) AS next_placed_at

559 FROM junior_da.orders)

560ORDER BY rider_id;

Press Alt+F1 for Accessibility Options

Query results

SAVE RESULTS

EXPLORE DATA

JOB INFORMATION	RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW	
Row	order_id	rider_id	placed_at	next_placed_at	diff_minutes	diff_days
1	fd4ee4cd-cc16-5479-9c88-f75...	00000c6b-7227-5b8f-a5ac-073...	2022-04-03 16:25:40 UTC	2022-04-13 16:39:45 UTC	14414	10
2	e11e2722-b037-5e7a-9f17-a90...	00000c6b-7227-5b8f-a5ac-073...	2022-04-13 16:39:45 UTC	2022-04-24 18:33:06 UTC	15953	11
3	5a756024-c067-5dfd-a767-664...	00000c6b-7227-5b8f-a5ac-073...	2022-04-24 18:33:06 UTC	2022-06-24 18:57:45 UTC	87864	61
4	1e5c3d7a-bad5-5fcd-ac63-715...	00000c6b-7227-5b8f-a5ac-073...	2022-06-24 18:57:45 UTC	null	null	null
5	5fec4c77-30ce-5f90-9d2a-37ae...	000022b4-1ae0-5c53-a98e-7a9...	2022-05-20 14:31:32 UTC	2022-06-03 09:54:29 UTC	19882	14
6	c92af69d-c0fc-50bc-9fc8-ea93...	000022b4-1ae0-5c53-a98e-7a9...	2022-06-03 09:54:29 UTC	2022-06-14 15:23:05 UTC	16168	11

Results per page:

50

1 – 50 of 1365382

|<

<

>

|>

6.1.3. Напишіть ОДИН запит, який виведе найпопулярніший тип оплати для кожного класу таксі. Популярність визначаємо по кількості замовлень.

☀ Врахуйте, що нас цікавлять тільки здійснені замовлення.

Результуюча таблиця має містити 4 колонки:

- тип класу таксі,
- тип оплати,
- кількість замовлень для данного типу оплати та класу таксі (надайте назву quantity)
- найпопулярніший спосіб оплати для поточного типу класу таксі (надайте назву most_popular_payment)


```

SELECT product
      ,payment_type
      ,quantity
      ,FIRST_VALUE(payment_type) OVER (PARTITION BY product ORDER BY quantity DESC) AS
most_popular_payment
FROM (SELECT product
      ,payment_type
      ,COUNT(order_id) AS quantity
FROM junior_da.orders
WHERE completed_at IS NOT NULL
GROUP BY product, payment_type
ORDER BY product, quantity DESC);

```

570 SELECT product
571 ,payment_type
572 ,quantity
573 ,FIRST_VALUE(payment_type) OVER (PARTITION BY product ORDER BY quantity DESC) AS most_popular_payment
574 FROM (SELECT product
575 ,payment_type
576 ,COUNT(order_id) AS quantity
577 FROM junior_da.orders
578 WHERE completed_at IS NOT NULL
579 GROUP BY product, payment_type
580 ORDER BY product, quantity DESC);
581 --11.2: Numbering functions

Press Alt+F1 for Accessibility Options

Query results [SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	product	payment_type	quantity	most_popular_payment		
1	Business	PAYMENT_TYPE_CASH	522	PAYMENT_TYPE_CASH		
2	Business	PAYMENT_TYPE_APPLE_PAY	333	PAYMENT_TYPE_CASH		
3	Business	PAYMENT_TYPE_CARD	307	PAYMENT_TYPE_CASH		
4	Business	PAYMENT_TYPE_GOOGLE_PAY	30	PAYMENT_TYPE_CASH		
5	Business	PAYMENT_TYPE_CORPORATE...	8	PAYMENT_TYPE_CASH		
6	CovidProtected	PAYMENT_TYPE_CASH	35	PAYMENT_TYPE_CASH		

Results per page: 50 1 - 47 of 47

Інший варіант скрипта з тотожним результатом:

```

SELECT product
      ,payment_type
      ,quantity
      ,FIRST_VALUE(payment_type) OVER (PARTITION BY product ORDER BY quantity DESC ROWS BETWEEN
UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS most_popular_payment
FROM (SELECT product
      ,payment_type
      ,COUNT(order_id) AS quantity
FROM junior_da.orders
WHERE completed_at IS NOT NULL
GROUP BY product, payment_type
ORDER BY product, quantity DESC);

```

```

582 SELECT product
583        ,payment_type
584        ,quantity
585        ,FIRST_VALUE(payment_type) OVER (PARTITION BY product ORDER BY quantity DESC ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED
FOLLOWING) AS most_popular_payment
586 FROM (SELECT product
587        ,payment_type
588        ,COUNT(order_id) AS quantity
589        FROM junior_da.orders
590        WHERE completed_at IS NOT NULL
591        GROUP BY product, payment_type
592        ORDER BY product, quantity DESC);

```

Query results

[SAVE RESULTS](#)
[EXPLORE DATA](#)

JOB INFORMATION					
RESULTS					
JSON					
EXECUTION DETAILS					
EXECUTION GRAPH					
PREVIEW					
Row	product	payment_type	quantity	most_popular_payment	
1	CovidProtected	PAYMENT_TYPE_CASH	35	PAYMENT_TYPE_CASH	
2	CovidProtected	PAYMENT_TYPE_CARD	14	PAYMENT_TYPE_CASH	
3	CovidProtected	PAYMENT_TYPE_APPLE_PAY	6	PAYMENT_TYPE_CASH	
4	CovidProtected	PAYMENT_TYPE_GOOGLE_PAY	2	PAYMENT_TYPE_CASH	
5	MiniVan	PAYMENT_TYPE_CASH	2673	PAYMENT_TYPE_CASH	
6	MiniVan	PAYMENT_TYPE_CARD	564	PAYMENT_TYPE_CASH	

Results per page: 50 1 – 47 of 47

Завдання 6.2

6.2.1. Напишіть ОДИН запит, який пронумерує замовлення для кожного водія по часу прийняття замовлення.

Врахуйте, що існують замовлення для яких водій не був знайдений – їх треба відфільтрувати. Результуюча таблиця має містити 4 колонки:

- ID водія,
- ID замовлення,
- час прийняття замовлення,
- номер замовлення (надайте назву order_num).

```

SELECT driver_id
       ,order_id
       ,accepted_at
       ,ROW_NUMBER() OVER (PARTITION BY driver_id ORDER BY accepted_at) AS order_num
FROM junior_da.orders
WHERE driver_id IS NOT NULL AND accepted_at IS NOT NULL
ORDER BY driver_id, accepted_at ASC;

```

```

578 WHERE completed_at IS NOT NULL
579 GROUP BY product, payment_type
580 ORDER BY product, quantity DESC);
581 --11.2: Numbering functions
582 SELECT driver_id
583        ,order_id
584        ,accepted_at
585        ,ROW_NUMBER() OVER (PARTITION BY driver_id ORDER BY accepted_at) AS order_num
586 FROM junior_da.orders
587 WHERE driver_id IS NOT NULL AND accepted_at IS NOT NULL
588 ORDER BY driver_id, accepted_at ASC;
589 --11.3: Aggregation functions

```

Query results

[SAVE RESULTS](#)
[EXPLORE DATA](#)

JOB INFORMATION					
RESULTS					
JSON					
EXECUTION DETAILS					
EXECUTION GRAPH					
PREVIEW					
Row	driver_id	order_id	accepted_at	order_num	
113	0000548b-16a0-515a-b82a-7ec...	08f45226-4b7f-5b35-8cfa-b581...	2022-06-30 09:40:18 UTC	113	
114	0000548b-16a0-515a-b82a-7ec...	49624aea-0930-5372-a928-36f...	2022-06-30 11:00:29 UTC	114	
115	0000548b-16a0-515a-b82a-7ec...	9cb2e50d-9a14-5735-8bdc-9e7...	2022-06-30 11:29:35 UTC	115	
116	000697fd-e7fb-5ec0-ac3e-928...	6befd6ca-eeeb-58f2-82a9-cf2e...	2022-01-10 14:04:53 UTC	1	
117	000697fd-e7fb-5ec0-ac3e-928...	0d588a46-5f63-57ac-a5d9-cb1...	2022-01-10 14:32:58 UTC	2	
118	000697fd-e7fb-5ec0-ac3e-928...	8807ahdc-d657-5f87-b8a3-3a4...	2022-01-10 14:49:17 UTC	3	

Results per page: 200 1 – 200 of 1121691

Завдання 6.3

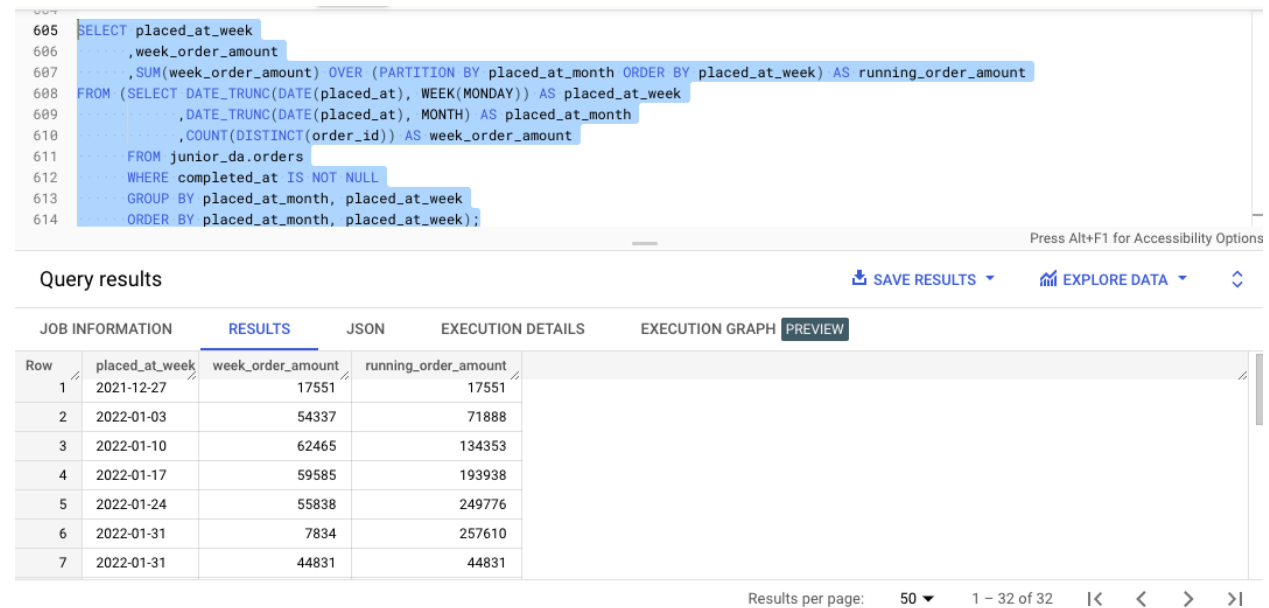
6.3.1. Напишіть ОДИН запит, який порахує накопичувальний підсумок суми замовлень для кожного тижня в місяці. З початком нового місяця накопичувальний підсумок має обнулятися. Тобто, накопичувальний підсумок рахуємо по тижням в рамках одного місяця.

● Врахуйте, що нас цікавлять тільки здійснені замовлення.

Результуюча таблиця має містити 3 колонки:

- тиждень розміщення замовлення, у форматі '2022-01-02' (надайте назву placed_at_week) ,
- загальна сума замовлень за тиждень (надайте назву week_order_amount),
- накопичувальна сума замовлень (надайте назву running_order_amount).

```
SELECT placed_at_week
       ,week_order_amount
       ,SUM(week_order_amount) OVER (PARTITION BY placed_at_month ORDER BY placed_at_week) AS
running_order_amount
FROM (SELECT DATE_TRUNC DATE(placed_at), WEEK(MONDAY)) AS placed_at_week
     ,DATE_TRUNC DATE(placed_at), MONTH) AS placed_at_month
     ,COUNT(DISTINCT(order_id)) AS week_order_amount
FROM junior_da.orders
WHERE completed_at IS NOT NULL
GROUP BY placed_at_month, placed_at_week
ORDER BY placed_at_month, placed_at_week);
```



Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	placed_at_week	week_order_amount	running_order_amount			
1	2021-12-27	17551	17551			
2	2022-01-03	54337	71888			
3	2022-01-10	62465	134353			
4	2022-01-17	59585	193938			
5	2022-01-24	55838	249776			
6	2022-01-31	7834	257610			
7	2022-01-31	44831	44831			

Results per page: 50 1 - 32 of 32

7. Bonus: PIVOT Function

Завдання 7.1

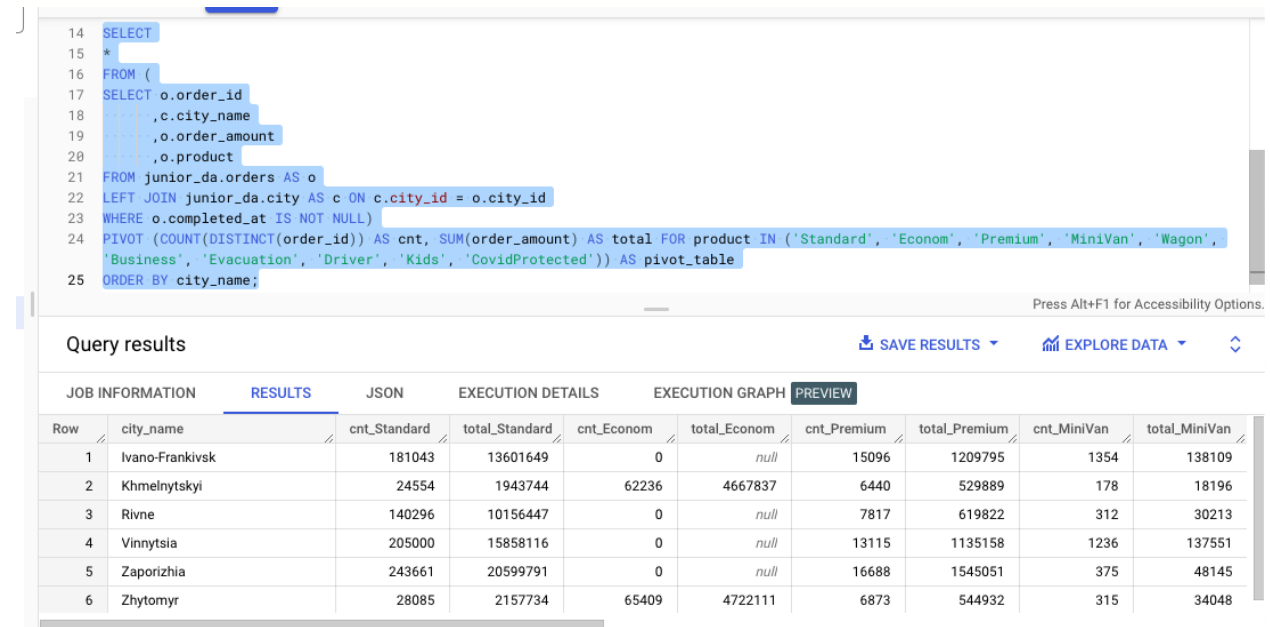
7.1.1. Напишіть ОДИН запит, який для кожного міста та типу класу таксі, порахує наступні показники, у вигляді зведеної таблиці:

- кількість унікальних замовлень (надайте назву cnt)
- загальна сума усіх замовлень (надайте назву total)

Врахуйте:

- нас цікавлять показники тільки для здійснених замовлень.
- назва міста має бути у рядках.
- тип класу таксі має бути у стовпчиках.

```
SELECT
*
FROM (
SELECT o.order_id
      ,c.city_name
      ,o.order_amount
      ,o.product
FROM junior_da.orders AS o
LEFT JOIN junior_da.city AS c ON c.city_id = o.city_id
WHERE o.completed_at IS NOT NULL)
PIVOT (COUNT(DISTINCT(order_id)) AS cnt, SUM(order_amount) AS total FOR product IN ('Standard',
'Econom', 'Premium', 'MiniVan', 'Wagon', 'Business', 'Evacuation', 'Driver', 'Kids',
'CovidProtected')) AS pivot_table
ORDER BY city_name;
```



The screenshot shows a database query editor with the SQL query pasted into the editor area. Below the editor, the 'Query results' section is displayed, showing a table with 10 columns and 6 rows of data. The table is titled 'Query results' and has tabs for 'JOB INFORMATION', 'RESULTS', 'JSON', 'EXECUTION DETAILS', 'EXECUTION GRAPH', and 'PREVIEW'. The 'RESULTS' tab is selected, showing the query results in a table format. The table has 10 columns: Row, city_name, cnt_Standard, total_Standard, cnt_Econom, total_Econom, cnt_Premium, total_Premium, cnt_MiniVan, and total_MiniVan. The data is as follows:

Row	city_name	cnt_Standard	total_Standard	cnt_Econom	total_Econom	cnt_Premium	total_Premium	cnt_MiniVan	total_MiniVan
1	Ivano-Frankivsk	181043	13601649	0	null	15096	1209795	1354	138109
2	Khmelnyskyi	24554	1943744	62236	4667837	6440	529889	178	18196
3	Rivne	140296	10156447	0	null	7817	619822	312	30213
4	Vinnytsia	205000	15858116	0	null	13115	1135158	1236	137551
5	Zaporizhia	243661	20599791	0	null	16688	1545051	375	48145
6	Zhytomyr	28085	2157734	65409	4722111	6873	544932	315	34048