

Домашнє завдання до Теми 5. Вкладені запити. Повторне використання коду

1. Напишіть SQL запит, який буде відображати таблицю order_details та поле customer_id з таблиці orders відповідно для кожного поля запису з таблиці order_details.

Це має бути зроблено за допомогою введеного запиту в операторі SELECT.

```
3 • SELECT
4     od.*,
5     (SELECT customer_id FROM orders o WHERE o.id = od.order_id) AS customer_id
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	id	order_id	product_id	quantity	customer_id
▶	1	10248	11	12	90
	2	10248	42	10	90
	3	10248	72	5	90
	4	10249	14	9	81
	5	10249	51	40	81
	6	10250	41	10	34
	7	10250	51	35	34
	8	10250	65	15	34
	9	10251	22	6	84
	10	10251	57	15	84
	11	10251	65	20	84
	12	10252	20	40	76
	13	10252	33	25	76

Result 6 x

2. Напишіть SQL запит, який буде відображати таблицю order_details. Відфільтруйте результати так, щоб відповідний запис із таблиці orders виконував умову shipper_id=3.

Це має бути зроблено за допомогою введеного запиту в операторі WHERE.

```
18 FROM order_details od
19 JOIN orders o ON od.order_id = o.id
20 WHERE o.shipper_id IN (SELECT shipper_id FROM orders WHERE shipper_id = 3)
21 ;
??
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	id	order_id	product_id	quantity	shipper_id
▶	1	10248	11	12	3
	2	10248	42	10	3
	3	10248	72	5	3
	21	10255	2	20	3
	22	10255	16	35	3
	23	10255	36	25	3
	24	10255	59	30	3
	27	10257	27	25	3
	28	10257	39	6	3
	29	10257	77	15	3
	33	10259	71	10	3

Result 10 x

3. Напишіть SQL запит, вкладений в операторі FROM, який буде обирати рядки з умовою quantity>10 з таблиці order_details. Для отриманих даних знайдіть середнє значення поля quantity — групувати слід за order_id.

```
26 • SELECT
27     od.order_id,
28     AVG(od.quantity) AS average_quantity
29     FROM (SELECT * FROM order_details WHERE quantity>10) od
30     GROUP BY 1;
```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	order_id	average_quantity			
▶	10248	12.0000			
	10249	40.0000			
	10250	25.0000			
	10251	17.5000			
	10252	35.0000			
	10253	34.0000			
	10254	19.0000			
	10255	27.5000			
	10256	13.5000			
	10257	20.0000			
	10258	57.5000			
	10260	25.5000			

4. Розв'яжіть завдання 3, використовуючи оператор WITH для створення тимчасової таблиці temp. Якщо ваша версія MySQL більш рання, ніж 8.0, створіть цей запит за аналогією до того, як це зроблено в конспекті.

```
33 • WITH temp as (
34     SELECT id,
35             order_id,
36             product_id,
37             quantity
38     FROM order_details
39     WHERE quantity>10
40 )
41 SELECT
42     order_id,
43     AVG(quantity) as avg_quantity
44 FROM temp
45 GROUP BY 1
```




Result Grid			Filter Rows:	Export:
	order_id	avg_quantity		
▶	10248	12.0000		
	10249	40.0000		
	10250	25.0000		
	10251	17.5000		
	10252	35.0000		

Result 15 ×

5. Створіть функцію з двома параметрами, яка буде ділити перший параметр на другий. Обидва параметри та значення, що повертається, повинні мати тип FLOAT.

Використайте конструкцію DROP FUNCTION IF EXISTS. Застосуйте функцію до атрибута quantity таблиці order_details . Другим параметром може бути довільне число на ваш розсуд.

```
78     od.order_id,  
79     od.product_id,  
80     od.quantity,  
81     p.total_quantity,  
82     DivisionOperation(od.quantity, p.total_quantity) AS quantity_ratio_by_order  
83 FROM order_details od  
84 JOIN product_total_quantity p ON od.product_id = p.product_id  
85
```

Result Grid		 Filter Rows:			Export:	 Wrap Cell Content:	
	order_id	product_id	quantity	total_quantity	quantity_ratio_by_order		
▶	10248	11	12	182	0.0659341		
	10248	42	10	77	0.12987		
	10248	72	5	270	0.0185185		
	10249	14	9	152	0.0592105		
	10249	51	40	163	0.245399		
	10250	41	10	139	0.0719424		
	10250	51	35	163	0.214724		
	10250	65	15	175	0.0857143		
	10251	22	6	18	0.333333		