

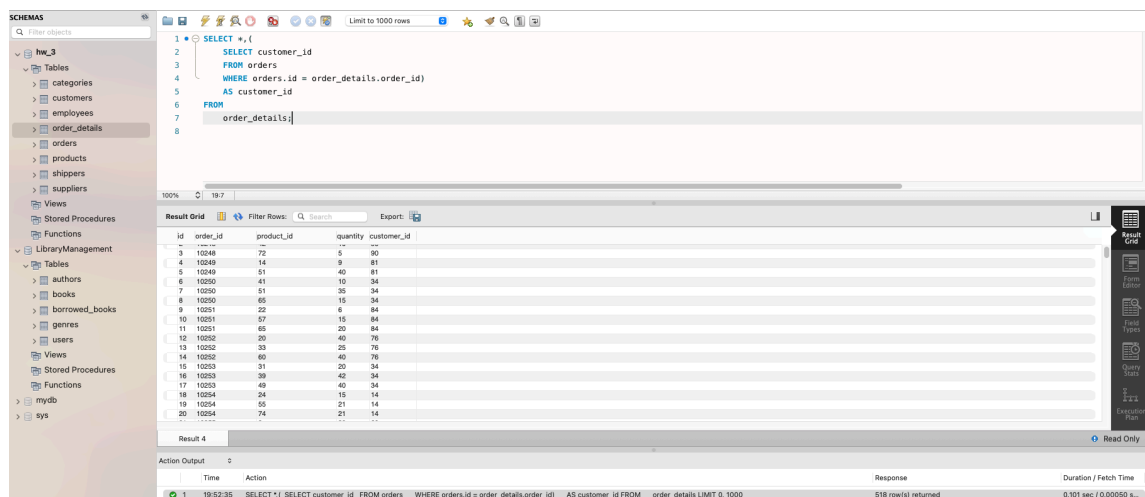
Опис домашнього завдання

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 вкладки запиту в операторі `SELECT`.

Відповідь:

```
SELECT *,(SELECT customer_id FROM orders
WHERE orders.id = order_details.order_id) AS customer_id
FROM order_details;
```



The screenshot shows a SQL IDE interface. On the left, a tree view displays the database schema, including tables like `categories`, `customers`, `employees`, `order_details`, `orders`, `products`, `shippers`, `suppliers`, `users`, `mydb`, and `sys`. The main editor displays the following SQL query:

```
1 SELECT *,(
2   SELECT customer_id
3   FROM orders
4   WHERE orders.id = order_details.order_id)
5   AS customer_id
6 FROM
7   order_details;
```

Below the query editor, the 'Result Grid' shows the results of the query. The grid has columns: `id`, `order_id`, `product_id`, `quantity`, and `customer_id`. The results are as follows:

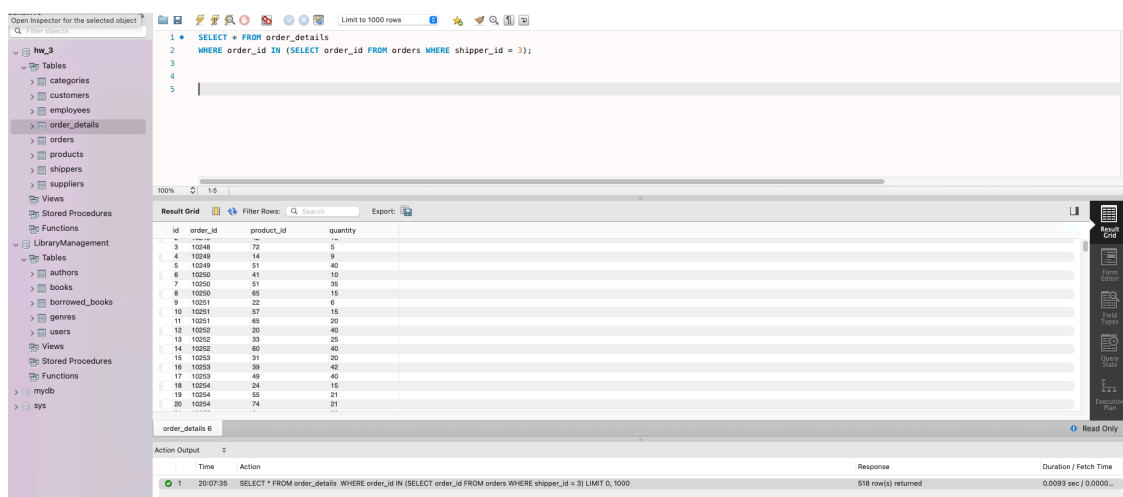
id	order_id	product_id	quantity	customer_id
3	10248	72	5	90
4	10249	14	9	81
5	10249	51	40	81
6	10250	41	10	34
7	10250	51	35	34
8	10250	65	15	34
9	10251	22	6	84
10	10251	67	15	84
11	10251	65	20	84
12	10252	20	40	76
13	10252	33	25	76
14	10252	60	40	76
15	10253	31	20	34
16	10253	39	42	34
17	10253	49	40	34
18	10254	24	15	14
19	10254	55	21	14
20	10254	74	21	14

The 'Action Output' section at the bottom shows the execution details: 19:52:35, SELECT *,(SELECT customer_id FROM orders WHERE orders.id = order_details.order_id) AS customer_id FROM order_details LIMIT 0, 1000, 518 row(s) returned, 0.101 sec / 0.00050 s.

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 вкладки запиту в операторі `WHERE`.

Відповідь:

```
SELECT * FROM order_details
WHERE order_id IN (SELECT order_id FROM orders WHERE shipper_id = 3);
```



The screenshot shows the same SQL IDE interface. The query editor now displays the following SQL query:

```
1 SELECT * FROM order_details
2 WHERE order_id IN (SELECT order_id FROM orders WHERE shipper_id = 3);
```

The 'Result Grid' shows the results of the query. The grid has columns: `id`, `order_id`, `product_id`, and `quantity`. The results are as follows:

id	order_id	product_id	quantity
3	10248	72	5
4	10249	14	9
5	10249	51	40
6	10250	41	10
7	10250	51	35
8	10250	65	15
9	10251	22	6
10	10251	67	15
11	10251	65	20
12	10252	20	40
13	10252	33	25
14	10252	60	40
15	10253	31	20
16	10253	39	42
17	10253	49	40
18	10254	24	15
19	10254	55	21
20	10254	74	21

The 'Action Output' section at the bottom shows the execution details: 20:07:35, SELECT * FROM order_details WHERE order_id IN (SELECT order_id FROM orders WHERE shipper_id = 3) LIMIT 0, 1000, 518 row(s) returned, 0.0093 sec / 0.00000 s.

3. Напишіть SQL запит, вкладений в оператор `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`.

Відповідь:

```
SELECT AVG(result.quantity) AS avg_quantity
FROM (SELECT order_id, quantity FROM order_details WHERE quantity > 10) AS result
GROUP BY order_id
ORDER BY avg_quantity ASC;
```



The screenshot shows a SQL IDE with a query editor at the top and a results grid below. The query is:

```
1 SELECT AVG(result.quantity) AS avg_quantity
2 FROM(
3     SELECT order_id, quantity
4     FROM order_details
5     WHERE quantity > 10) AS result
6 GROUP BY order_id
7 ORDER BY avg_quantity ASC;
```

The results grid displays a single column named 'avg_quantity' with 17 rows of data. The values range from 12.0000 to 15.0000. The bottom status bar indicates '175 row(s) returned' and '0.0011 sec / 0.00003'.

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.

Відповідь:

```
WITH TemporalType AS (
SELECT order_id, quantity FROM order_details WHERE quantity > 10)
SELECT AVG(quantity) AS avg_quantity FROM TemporalType
GROUP BY order_id
ORDER BY avg_quantity ASC;
```



The screenshot shows a SQL IDE with a query editor at the top and a results grid below. The query is:

```
1 WITH TemporalType AS (
2     SELECT order_id, quantity
3     FROM order_details
4     WHERE quantity > 10)
5 SELECT AVG(quantity) AS avg_quantity
6 FROM TemporalType
7 GROUP BY order_id
8 ORDER BY avg_quantity ASC;
```

The results grid displays a single column named 'avg_quantity' with 17 rows of data, identical to the previous screenshot. The bottom status bar shows two queries: the first query returned 175 rows in 0.0011 sec, and the second query (the CTE query) returned 175 rows in 0.0059 sec.

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`.

Відповідь:

```
DROP FUNCTION IF EXISTS Divide;
DELIMITER //
CREATE FUNCTION Divide(param1 FLOAT, param2 FLOAT)
RETURNS FLOAT
DETERMINISTIC
NO SQL
BEGIN
    DECLARE result FLOAT;
    SET result =param1/param2;
    RETURN result;
END //
DELIMITER ;
```

```
SELECT Divide (
(SELECT quantity FROM order_details LIMIT 1),
(SELECT quantity FROM order_details LIMIT 1 OFFSET 1)
) AS result
```

The screenshot displays a database IDE interface. The top section shows a query grid with columns: id, order_id, product_id, quantity. The bottom section shows the execution log with three entries:

Time	Action	Response	Duration / Fetch Time
21:46:07	DROP FUNCTION IF EXISTS Divide	0 row(s) affected	0.0015 sec
21:46:07	CREATE FUNCTION Divide(param1 FLOAT, param2 FLOAT) RETURNS FLOAT DETERMINISTIC NO SQL BEGIN DECLARE result FLOAT; SET result =param1/param2; RETURN result; END //	0 row(s) affected	0.0043 sec
21:46:07	SELECT Divide ((SELECT quantity FROM order_details LIMIT 1), (SELECT quantity FROM order_details LIMIT 1 OFFSET 1)) AS result	1 row(s) returned	0.0042 sec / 0.00001...

The result of the third query is 1.2.