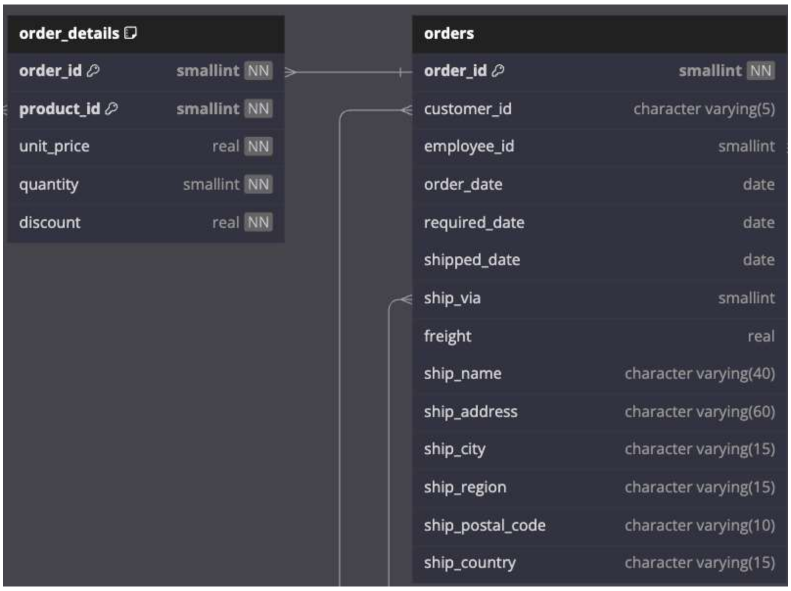
Питання 1

Правильно

Балів 1,00 з 1,00

Your supervisor came to you with a request. She thinks that some of sales managers accidentally double-entered a line item on an order, with a different **product_id**, but the same **quantity**. She know that the **quantity** was *60* or more. Show all the **order_id** that match this request, order by **order_id**.

Use the model:



order_details		
order_id	smallint	NN
product_id	smallint	NN
unit_price	real	NN
quantity	smallint	NN
discount	real	NN

orders		
order_id	smallint	NN
customer_id	character varying(5)	
employee_id	smallint	
order_date	date	
required_date	date	
shipped_date	date	
ship_via	smallint	
freight	real	
ship_name	character varying(40)	
ship_address	character varying(60)	
ship_city	character varying(15)	
ship_region	character varying(15)	
ship_postal_code	character varying(10)	
ship_country	character varying(15)	

Note. Use the subquery-derived table.

For example:

Тест	Result
-- Testing with original db	order_id -------- 10263 10658 10990 11030

Відповідь: (penalty regime: 0 %)

```
SELECT order_id
FROM (
    SELECT order_id, quantity
    FROM order_details
    WHERE quantity >= 60
    GROUP BY order_id, quantity
    HAVING COUNT(DISTINCT product_id) > 1
) AS subquery
ORDER BY order_id;

```

	Тест	Expected	Got	
✓	-- Testing with original db	order_id -------- 10263 10658 10990 11030	order_id -------- 10263 10658 10990 11030	✓
✓	-- Testing with extra row	order_id -------- 10658 10990 11030	order_id -------- 10658 10990 11030	✓

Пройшов усі тести! ✓

Правильно

Бали за цю відповідь: 1,00/1,00.

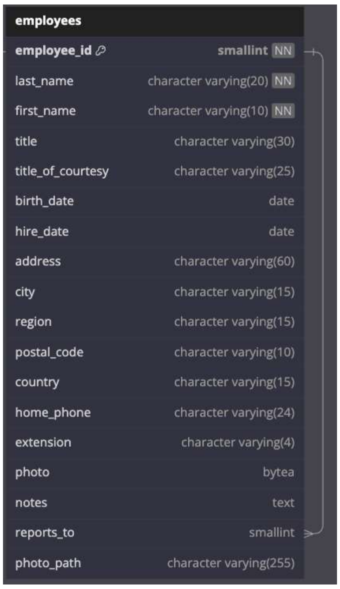
Питання 2

Правильно

Балів 1,00 з 1,00

We know that Andrew Fuller is the Vice President of Northwind Company. Create the report that shows the list of those employees (**last_name** and **first_name**) who were hired earlier than Fuller.

Use the table:



employees		
employee_id	smallint	NN
last_name	character varying(20)	NN
first_name	character varying(10)	NN
title	character varying(30)	
title_of_courtesy	character varying(25)	
birth_date	date	
hire_date	date	
address	character varying(60)	
city	character varying(15)	
region	character varying(15)	
postal_code	character varying(10)	
country	character varying(15)	
home_phone	character varying(24)	
extension	character varying(4)	
photo	bytea	
notes	text	
reports_to	smallint	
photo_path	character varying(255)	

Note. Use the subquery.

For example:

Тест	Result
-- Testing with original db	last_name first_name --------- ---------- Davolio Nancy Leverling Janet Ann Miller

Відповідь: (penalty regime: 0 %)

```
SELECT last_name, first_name
FROM employees
WHERE hire_date < (
    SELECT hire_date
    FROM employees
    WHERE first_name = 'Andrew'
    AND last_name = 'Fuller'
);
```

	Тест	**Expected**	**Got**	

✓	`-- Testing with original db`	`last_name  first_name`
		`---------  ----------`
		`Davolio    Nancy`
		`Leverling  Janet`
		`Ann        Miller`
			`last_name  first_name`
			`---------  ----------`
			`Davolio    Nancy`
			`Leverling  Janet`
			`Ann        Miller`
				✓

✓	`-- Testing with extra row`	`last_name  first_name`
		`---------  ----------`
		`Leverling  Janet`
		`Ann        Miller`
			`last_name  first_name`
			`---------  ----------`
			`Leverling  Janet`
			`Ann        Miller`
				✓

Пройшов усі тести! ✓

Правильно

Бали за цю відповідь: 1,00/1,00.

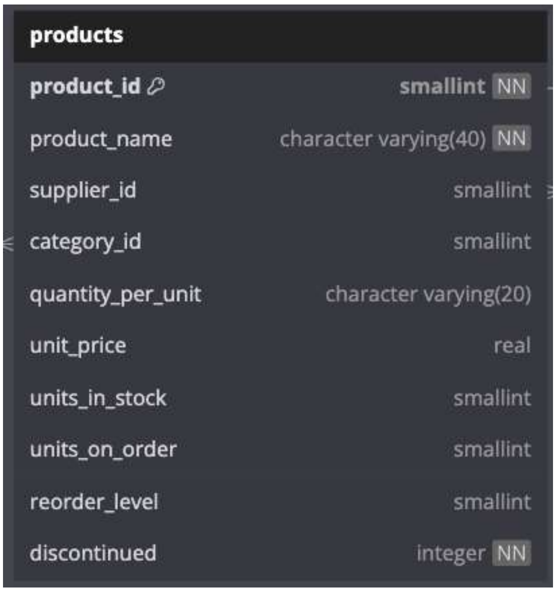
Питання 3

Правильно

Балів 1,00 з 1,00

Write the query that should create the list of products and their **unit_price** for **products** with a price greater than the *average* products' **unit_price**.
The result set should be ordered by **unit_price** ascending.

Use the table:



products	
product_id	smallint NN
product_name	character varying(40) NN
supplier_id	smallint
category_id	smallint
quantity_per_unit	character varying(20)
unit_price	real
units_in_stock	smallint
units_on_order	smallint
reorder_level	smallint
discontinued	integer NN

Note. Use the subquery to get the *average* **unit_price** from the **products** table.

For example:

<table>
<tr><th>Тест</th><th>Result</th></tr>
<tr><td><pre>-- Testing with original db</pre></td><td><pre>
product_name   unit_price
-------------------------- ----------
Perth Pasties   32.8
Wimmers gute Semmelknudel   33.25
Camembert Pierrot   34
Mozzarella di Giovanni   34.8
Gudbrandsdalsost   36
Queso Manchego La Pastora   38
Gnocchi di nonna Alice   38
Alice Mutton   39
Northwoods Cranberry Sauce 40
Schoggi Schokolade   43.9
Vegie-spread   43.9
Tomatoes - Diced, Canned   45.21
Russle Sauerkraut   45.6
Beans - Black Bean, Dry   45.82
Ipoh Coffee   46
Tarte au sucre   49.3
Manjimup Dried Apples   53
Raclette Courdavault   55
Carnarvon Tigers   62.5
Beef - Tenderloin Tails   62.76
Wine - Chardonnay Mondavi   68.16
Wine - Merlot Vina Carmen   69.59
Onions - White   75.86
Ecolab - Ster Bac   77.41
Sir Rodney's Marmalade   81
Sponge Cake Mix - Chocolate 95.53
Mishi Kobe Niku   97
Thuringer Rostbratwurst   123.79
Cute de Blaye   263.5
</pre></td></tr>
</table>

Відповідь: (penalty regime: 0 %)

```sql
SELECT product_name, unit_price
FROM products
WHERE unit_price > (
    SELECT AVG(unit_price)
    FROM products
)
ORDER BY unit_price ASC;

```

	Тест	Expected		Got		
✓	-- Testing with original db	product_name	unit_price	product_name	unit_price	✓
		--------------------------	----------	--------------------------	----------	
		Perth Pasties	32.8	Perth Pasties	32.8	
		Wimmers gute Semmelknudel	33.25	Wimmers gute Semmelknudel	33.25	
		Camembert Pierrot	34	Camembert Pierrot	34	
		Mozzarella di Giovanni	34.8	Mozzarella di Giovanni	34.8	
		Gudbrandsdalsost	36	Gudbrandsdalsost	36	
		Queso Manchego La Pastora	38	Queso Manchego La Pastora	38	
		Gnocchi di nonna Alice	38	Gnocchi di nonna Alice	38	
		Alice Mutton	39	Alice Mutton	39	
		Northwoods Cranberry Sauce	40	Northwoods Cranberry Sauce	40	
		Schoggi Schokolade	43.9	Schoggi Schokolade	43.9	
		Vegie-spread	43.9	Vegie-spread	43.9	
		Tomatoes - Diced, Canned	45.21	Tomatoes - Diced, Canned	45.21	
		Russle Sauerkraut	45.6	Russle Sauerkraut	45.6	
		Beans - Black Bean, Dry	45.82	Beans - Black Bean, Dry	45.82	
		Ipoh Coffee	46	Ipoh Coffee	46	
		Tarte au sucre	49.3	Tarte au sucre	49.3	
		Manjimup Dried Apples	53	Manjimup Dried Apples	53	
		Raclette Courdavault	55	Raclette Courdavault	55	
		Carnarvon Tigers	62.5	Carnarvon Tigers	62.5	
		Beef - Tenderloin Tails	62.76	Beef - Tenderloin Tails	62.76	
		Wine - Chardonnay Mondavi	68.16	Wine - Chardonnay Mondavi	68.16	
		Wine - Merlot Vina Carmen	69.59	Wine - Merlot Vina Carmen	69.59	
		Onions - White	75.86	Onions - White	75.86	
		Ecolab - Ster Bac	77.41	Ecolab - Ster Bac	77.41	
		Sir Rodney's Marmalade	81	Sir Rodney's Marmalade	81	
		Sponge Cake Mix - Chocolate	95.53	Sponge Cake Mix - Chocolate	95.53	
		Mishi Kobe Niku	97	Mishi Kobe Niku	97	
		Thuringer Rostbratwurst	123.79	Thuringer Rostbratwurst	123.79	
		Cute de Blaye	263.5	Cute de Blaye	263.5	

	Тест	Expected	Got	
✓	-- Testing with extra row	product_name unit_price --------------------------- ---------- Flotemysost 21.5 Putu chinois 24 Gravad lax 26 Mushroom - Chanterelle, Dry 28.23 Sirop d'urable 28.5 Mascarpone Fabioli 32 Perth Pasties 32.8 Wimmers gute Semmelknudel 33.25 Camembert Pierrot 34 Mozzarella di Giovanni 34.8 Gudbrandsdalsost 36 Gnocchi di nonna Alice 38 Vegie-spread 43.9 Tomatoes - Diced, Canned 45.21 Beans - Black Bean, Dry 45.82 Ipoh Coffee 46 Tarte au sucre 49.3 Manjimup Dried Apples 53 Raclette Courdavault 55 Beef - Tenderloin Tails 62.76 Wine - Chardonnay Mondavi 68.16 Wine - Merlot Vina Carmen 69.59 Onions - White 75.86 Ecolab - Ster Bac 77.41 Sponge Cake Mix - Chocolate 95.53 Cute de Blaye 263.5	product_name unit_price --------------------------- ---------- Flotemysost 21.5 Putu chinois 24 Gravad lax 26 Mushroom - Chanterelle, Dry 28.23 Sirop d'urable 28.5 Mascarpone Fabioli 32 Perth Pasties 32.8 Wimmers gute Semmelknudel 33.25 Camembert Pierrot 34 Mozzarella di Giovanni 34.8 Gudbrandsdalsost 36 Gnocchi di nonna Alice 38 Vegie-spread 43.9 Tomatoes - Diced, Canned 45.21 Beans - Black Bean, Dry 45.82 Ipoh Coffee 46 Tarte au sucre 49.3 Manjimup Dried Apples 53 Raclette Courdavault 55 Beef - Tenderloin Tails 62.76 Wine - Chardonnay Mondavi 68.16 Wine - Merlot Vina Carmen 69.59 Onions - White 75.86 Ecolab - Ster Bac 77.41 Sponge Cake Mix - Chocolate 95.53 Cute de Blaye 263.5	✓

Пройшов усі тести! ✓

Правильно

Бали за цю відповідь: 1,00/1,00.

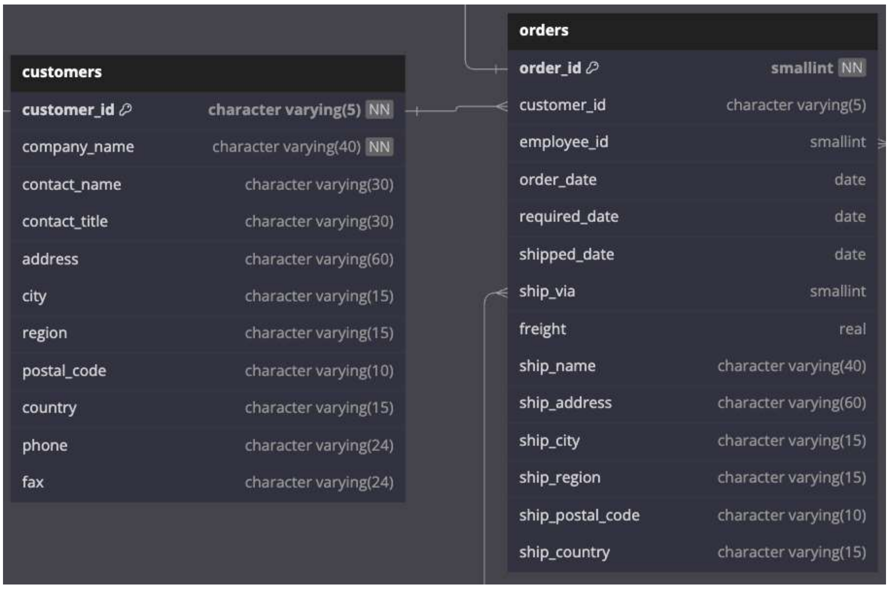
Питання 4

Правильно

Балів 1,00 з 1,00

Create the report that should show the **company_name** from Germany that placed orders in 1996 (*for test use 2016, see note*).

Use the model:



customers

customer_id	character varying(5)	NN
company_name	character varying(40)	NN
contact_name	character varying(30)	
contact_title	character varying(30)	
address	character varying(60)	
city	character varying(15)	
region	character varying(15)	
postal_code	character varying(10)	
country	character varying(15)	
phone	character varying(24)	
fax	character varying(24)	

orders

order_id	smallint	NN
customer_id	character varying(5)	
employee_id	smallint	
order_date	date	
required_date	date	
shipped_date	date	
ship_via	smallint	
freight	real	
ship_name	character varying(40)	
ship_address	character varying(60)	
ship_city	character varying(15)	
ship_region	character varying(15)	
ship_postal_code	character varying(10)	
ship_country	character varying(15)	

*Note. The test is running using SQLite Database that has the same structure, but slightly different information! You need to change the year from **1996** to **2016** when you copy your code from **supabase**. Also, a reminder that you need to use function **strftime('%Y', order_date)** to extract year from date instead of PostgreSQL **EXTRACT** when you copy your code from **supabase**.*

Use subquery to create this report.

For example:

Тест	Result
`-- Testing with original db`	company_name ------------------------ Alfreds Futterkiste Blauer See Delikatessen Drachenblut Delikatessen Frankenversand Kuniglich Essen Lehmanns Marktstand Morgenstern Gesundkost Ottilies Kuseladen QUICK-Stop Toms Spezialituten Die Wandernde Kuh

Відповідь: (penalty regime: 0 %)

```sql
SELECT DISTINCT c.company_name
FROM customers c
WHERE c.country = 'Germany'
AND c.customer_id IN (
    SELECT o.customer_id
    FROM orders o
    WHERE strftime('%Y', o.order_date) = '2016'
);

```

	Тест	Expected	Got	
✓	-- Testing with original db	company_name ------------------------ Alfreds Futterkiste Blauer See Delikatessen Drachenblut Delikatessen Frankenversand Kuniglich Essen Lehmanns Marktstand Morgenstern Gesundkost Ottilies Kuseladen QUICK-Stop Toms Spezialituten Die Wandernde Kuh	company_name ------------------------ Alfreds Futterkiste Blauer See Delikatessen Drachenblut Delikatessen Frankenversand Kuniglich Essen Lehmanns Marktstand Morgenstern Gesundkost Ottilies Kuseladen QUICK-Stop Toms Spezialituten Die Wandernde Kuh	✓
✓	-- Testing with extra row	company_name ------------------------ Blauer See Delikatessen Drachenblut Delikatessen Frankenversand Kuniglich Essen Lehmanns Marktstand Morgenstern Gesundkost Ottilies Kuseladen QUICK-Stop Toms Spezialituten Die Wandernde Kuh	company_name ------------------------ Blauer See Delikatessen Drachenblut Delikatessen Frankenversand Kuniglich Essen Lehmanns Marktstand Morgenstern Gesundkost Ottilies Kuseladen QUICK-Stop Toms Spezialituten Die Wandernde Kuh	✓

Пройшов усі тести! ✓

Правильно

Бали за цю відповідь: 1,00/1,00.

Питання 5

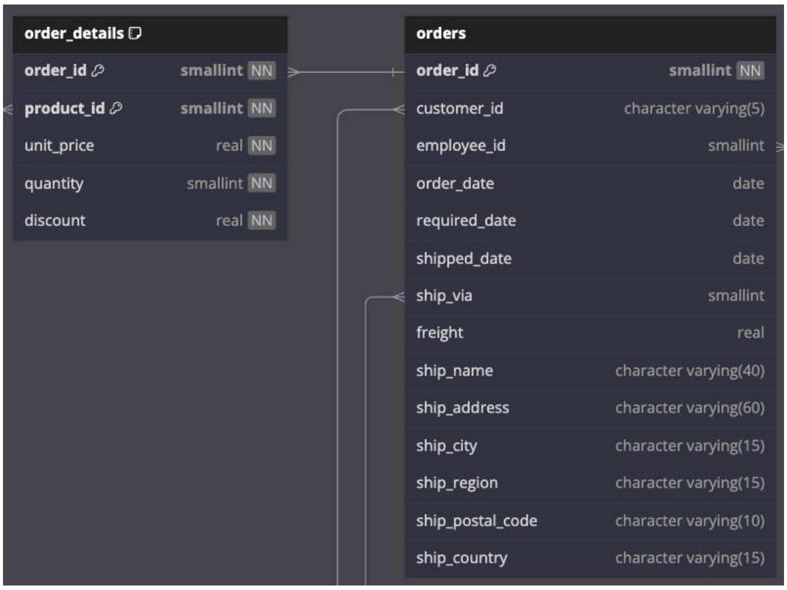
Правильно

Балів 1,00 з 1,00

Create the query that should show the date when the orders were shipped (**shipped_date**), the number of orders (**number_of_orders**) and total sum (including **discount**) of the orders (**total**) shipped at this date. The report includes only the 1st quarter of 1997 (*for test use 2016, see note*) with the number of orders greater than 3.

The result should be sorted by **shipped_date**

Use the model:



order_details	
order_id	smallint NN
product_id	smallint NN
unit_price	real NN
quantity	smallint NN
discount	real NN

orders	
order_id	smallint NN
customer_id	character varying(5)
employee_id	smallint
order_date	date
required_date	date
shipped_date	date
ship_via	smallint
freight	real
ship_name	character varying(40)
ship_address	character varying(60)
ship_city	character varying(15)
ship_region	character varying(15)
ship_postal_code	character varying(10)
ship_country	character varying(15)

Note. A subtotal is calculated by a sub-query for each order. The sub-query forms a table and then joined with the **orders** table.

Note. *The test is running using SQLite Database that has the same structure, but slightly different information! You need to change the year from* **1997** *to* **2016** *when you copy your code from* **supabase**.

Also, a reminder that you need to use function ***strftime('%Y', order_date)*** *to extract year from date instead of PostgreSQL* ***EXTRACT*** *when you copy your code from* ***supabase***.

In ***supabase*** *you might need to cast the* ***<value>*** *of the ROUND function to* ***numeric*** *type which is not needed in the test.*

Use ROUND(<*value*>, *2*) function for calculated sum of each order in subquery.

For example:

Тест	Result
-- Testing with original db	shipped_date number_of_orders total ------------ ---------------- -------- 2016-01-05 5 4328.64 2016-01-14 5 6535.57 2016-01-21 4 7749.3 2016-01-23 6 9470.21 2016-01-30 6 8662.72 2016-02-04 4 12125.34 2016-02-09 4 4307.26 2016-02-12 5 20551.0 2016-02-18 4 3719.1 2016-02-20 4 9183.85 2016-02-23 4 20285.0 2016-03-04 5 3310.25 2016-03-13 5 3531.58 2016-03-18 7 10474.05 2016-03-19 4 5279.95 2016-03-20 4 6422.69 2016-03-23 4 4185.75 2016-03-27 4 3552.75

Відповідь: (penalty regime: 0 %)

```
SELECT
    o.shipped_date,
    COUNT(o.order_id) AS number_of_orders,
    ROUND(SUM(od.total), 2) AS total
FROM orders o
JOIN (
    SELECT order_id, ROUND(SUM(unit_price * quantity * (1 - discount)), 2) AS total
    FROM order_details
    GROUP BY order_id
) AS od ON o.order_id = od.order_id
WHERE strftime('%Y', o.shipped_date) = '2016'
AND strftime('%m', o.shipped_date) IN ('01', '02', '03')
GROUP BY o.shipped_date
HAVING COUNT(o.order_id) > 3
ORDER BY o.shipped_date;

```

	Тест	Expected	Got	
✓	-- Testing with original db	shipped_date number_of_orders total ------------ ---------------- -------- 2016-01-05 5 4328.64 2016-01-14 5 6535.57 2016-01-21 4 7749.3 2016-01-23 6 9470.21 2016-01-30 6 8662.72 2016-02-04 4 12125.34 2016-02-09 4 4307.26 2016-02-12 5 20551.0 2016-02-18 4 3719.1 2016-02-20 4 9183.85 2016-02-23 4 20285.0 2016-03-04 5 3310.25 2016-03-13 5 3531.58 2016-03-18 7 10474.05 2016-03-19 4 5279.95 2016-03-20 4 6422.69 2016-03-23 4 4185.75 2016-03-27 4 3552.75	shipped_date number_of_orders total ------------ ---------------- -------- 2016-01-05 5 4328.64 2016-01-14 5 6535.57 2016-01-21 4 7749.3 2016-01-23 6 9470.21 2016-01-30 6 8662.72 2016-02-04 4 12125.34 2016-02-09 4 4307.26 2016-02-12 5 20551.0 2016-02-18 4 3719.1 2016-02-20 4 9183.85 2016-02-23 4 20285.0 2016-03-04 5 3310.25 2016-03-13 5 3531.58 2016-03-18 7 10474.05 2016-03-19 4 5279.95 2016-03-20 4 6422.69 2016-03-23 4 4185.75 2016-03-27 4 3552.75	✓
✓	-- Testing with extra row	shipped_date number_of_orders total ------------ ---------------- -------- 2016-02-04 4 12125.34 2016-02-09 4 4307.26 2016-02-12 5 20551.0 2016-02-18 4 3719.1 2016-02-20 4 9183.85 2016-02-23 4 20285.0 2016-03-04 5 3310.25 2016-03-13 5 3531.58 2016-03-18 7 10474.05 2016-03-19 4 5279.95 2016-03-20 4 6422.69 2016-03-23 4 4185.75 2016-03-27 4 3552.75	shipped_date number_of_orders total ------------ ---------------- -------- 2016-02-04 4 12125.34 2016-02-09 4 4307.26 2016-02-12 5 20551.0 2016-02-18 4 3719.1 2016-02-20 4 9183.85 2016-02-23 4 20285.0 2016-03-04 5 3310.25 2016-03-13 5 3531.58 2016-03-18 7 10474.05 2016-03-19 4 5279.95 2016-03-20 4 6422.69 2016-03-23 4 4185.75 2016-03-27 4 3552.75	✓

Пройшов усі тести! ✓

Правильно

Бали за цю відповідь: 1,00/1,00.

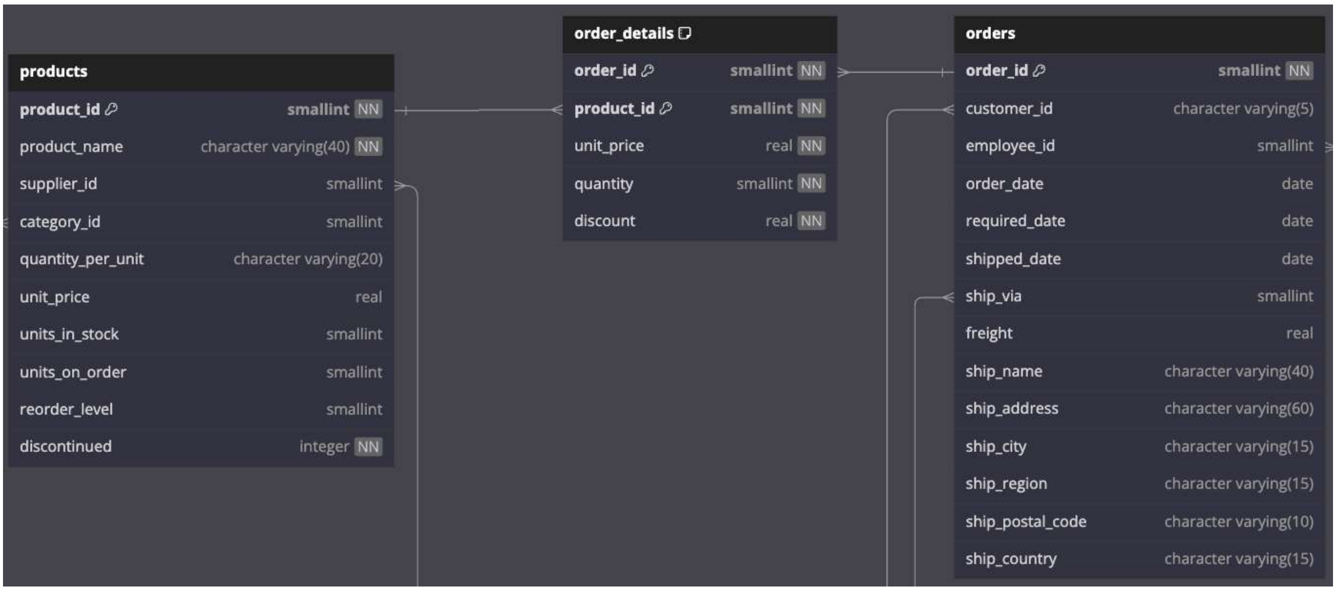
Питання 6

Правильно

Балів 1,00 з 1,00

For the category starts with 'Dairy' get the list of products sold and the total sales amount including discount (alias **product_sales**) during the 1st quarter of 1997 (*for test use 2016, see note*).

Use the model:



products

product_id	smallint	NN
product_name	character varying(40)	NN
supplier_id	smallint	
category_id	smallint	
quantity_per_unit	character varying(20)	
unit_price	real	
units_in_stock	smallint	
units_on_order	smallint	
reorder_level	smallint	
discontinued	integer	NN

order_details

order_id	smallint	NN
product_id	smallint	NN
unit_price	real	NN
quantity	smallint	NN
discount	real	NN

orders

order_id	smallint	NN
customer_id	character varying(5)	
employee_id	smallint	
order_date	date	
required_date	date	
shipped_date	date	
ship_via	smallint	
freight	real	
ship_name	character varying(40)	
ship_address	character varying(60)	
ship_city	character varying(15)	
ship_region	character varying(15)	
ship_postal_code	character varying(10)	
ship_country	character varying(15)	

Note. Use the subquery to get sales for each product on each order. Join the table from the subquery with an outer query on **product_id**.
Note. The test is running using SQLite Database that has the same structure, but slightly different information! You need to change the year from ***1996*** *to* ***2016*** *when you copy your code from* ***supabase****.*
Use ROUND(..., **1**) function for a calculated total for each product in the subquery.

For example:

Тест	Result
-- Testing with original db	category_name product_name product_sales ------------- ------------------------ ------------- Dairy products Camembert Pierrot 11845.6 Dairy products Flotemysost 3918.4 Dairy products Geitost 303.5 Dairy products Gorgonzola Telino 2762.6 Dairy products Gudbrandsdalsost 2394.0 Dairy products Mascarpone Fabioli 832.0 Dairy products Mozzarella di Giovanni 4541.4 Dairy products Queso Cabrales 3018.8 Dairy products Queso Manchego La Pastora 1900.0 Dairy products Raclette Courdavault 13158.8

Відповідь: (penalty regime: 0 %)

```sql
select a.category_name, b.product_name, round(sum(c.extended_price), 1) as product_sales
from categories AS a
 join products AS b on a.category_id = b.category_id
 join
(
    select  y.order_id,
        y.product_id,
        x.product_name,
        round(y.unit_price * y.quantity * (1 - y.discount), 1) as extended_price
    from products x
    inner join order_details y on x.product_id = y.product_id
    order by y.order_id
) AS c on c.product_id = b.product_id
 join orders AS d on d.order_id = c.order_id
where a.category_name like 'Dairy%'
and d.order_date between '2016-01-01' and '2016-03-31'
group by a.category_id, a.category_name, b.product_name
order by b.product_name;
```

	Тест	Expected	Got	
✓	-- Testing with original db	category_name product_name product_sales -------------- ------------------------- ------------- Dairy products Camembert Pierrot 11845.6 Dairy products Flotemysost 3918.4 Dairy products Geitost 303.5 Dairy products Gorgonzola Telino 2762.6 Dairy products Gudbrandsdalsost 2394.0 Dairy products Mascarpone Fabioli 832.0 Dairy products Mozzarella di Giovanni 4541.4 Dairy products Queso Cabrales 3018.8 Dairy products Queso Manchego La Pastora 1900.0 Dairy products Raclette Courdavault 13158.8	category_name product_name product_sales -------------- ------------------------- ------------- Dairy products Camembert Pierrot 11845.6 Dairy products Flotemysost 3918.4 Dairy products Geitost 303.5 Dairy products Gorgonzola Telino 2762.6 Dairy products Gudbrandsdalsost 2394.0 Dairy products Mascarpone Fabioli 832.0 Dairy products Mozzarella di Giovanni 4541.4 Dairy products Queso Cabrales 3018.8 Dairy products Queso Manchego La Pastora 1900.0 Dairy products Raclette Courdavault 13158.8	✓
✓	-- Testing with extra row	category_name product_name product_sales -------------- ------------------------- ------------- Dairy products Camembert Pierrot 11845.6 Dairy products Flotemysost 3918.4 Dairy products Geitost 42.0 Dairy products Gorgonzola Telino 318.8 Dairy products Gudbrandsdalsost 2394.0 Dairy products Mascarpone Fabioli 192.0 Dairy products Mozzarella di Giovanni 4541.4 Dairy products Queso Cabrales 3018.8 Dairy products Queso Manchego La Pastora 1900.0 Dairy products Raclette Courdavault 13158.8	category_name product_name product_sales -------------- ------------------------- ------------- Dairy products Camembert Pierrot 11845.6 Dairy products Flotemysost 3918.4 Dairy products Geitost 42.0 Dairy products Gorgonzola Telino 318.8 Dairy products Gudbrandsdalsost 2394.0 Dairy products Mascarpone Fabioli 192.0 Dairy products Mozzarella di Giovanni 4541.4 Dairy products Queso Cabrales 3018.8 Dairy products Queso Manchego La Pastora 1900.0 Dairy products Raclette Courdavault 13158.8	✓

Пройшов усі тести! ✓

Правильно

Бали за цю відповідь: 1,00/1,00.

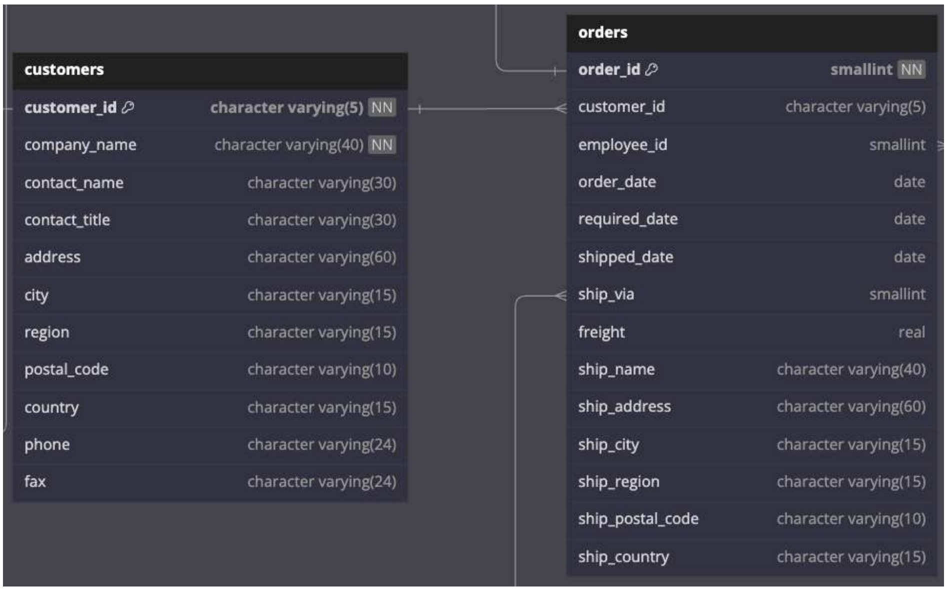
Питання 7

Правильно

Балів 1,00 з 1,00

Your boss wants to know the name of the company that placed order 10290.

Use the model:



customers	
customer_id	character varying(5) NN
company_name	character varying(40) NN
contact_name	character varying(30)
contact_title	character varying(30)
address	character varying(60)
city	character varying(15)
region	character varying(15)
postal_code	character varying(10)
country	character varying(15)
phone	character varying(24)
fax	character varying(24)

orders	
order_id	smallint NN
customer_id	character varying(5)
employee_id	smallint
order_date	date
required_date	date
shipped_date	date
ship_via	smallint
freight	real
ship_name	character varying(40)
ship_address	character varying(60)
ship_city	character varying(15)
ship_region	character varying(15)
ship_postal_code	character varying(10)
ship_country	character varying(15)

Note. Use subquery.

For example:

Тест	Result

`-- Testing with original db`	`company_name`
	`----------------`
	`Comurcio Mineiro`

Відповідь: (penalty regime: 0 %)

```
SELECT company_name
FROM customers
WHERE customer_id = (
    SELECT customer_id
    FROM orders
    WHERE order_id = 10290
);

```

	Тест	Expected	Got	

✓	`-- Testing with original db`	`company_name`
		`----------------`
		`Comurcio Mineiro`
			`company_name`
			`----------------`
			`Comurcio Mineiro`
				✓

	Тест	Expected	Got	

✓	`-- Testing with extra row`	`company_name`
		`-------------------`
		`Alfreds Futterkiste`
			`company_name`
			`-------------------`
			`Alfreds Futterkiste`
				✓

Пройшов усі тести! ✓

Правильно

Бали за цю відповідь: 1,00/1,00.

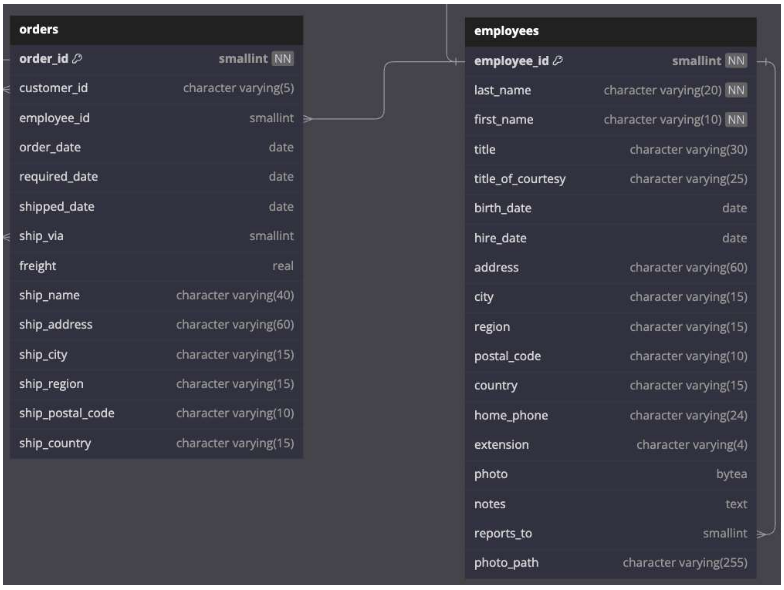
Питання **8**

Правильно

Балів 1,00 з 1,00

Some sales managers have more orders arriving late than others. Maybe they're not following up on the order process, and need more training. Your boss has been doing some more thinking about the problem of late orders (when order was delivered later than was required). He realized that just looking at the number of orders arriving late for each sales managers isn't a good idea. It needs to be compared against the total number of orders per sales managers.

Use the model:



orders	
order_id	smallint NN
customer_id	character varying(5)
employee_id	smallint
order_date	date
required_date	date
shipped_date	date
ship_via	smallint
freight	real
ship_name	character varying(40)
ship_address	character varying(60)
ship_city	character varying(15)
ship_region	character varying(15)
ship_postal_code	character varying(10)
ship_country	character varying(15)

employees	
employee_id	smallint NN
last_name	character varying(20) NN
first_name	character varying(10) NN
title	character varying(30)
title_of_courtesy	character varying(25)
birth_date	date
hire_date	date
address	character varying(60)
city	character varying(15)
region	character varying(15)
postal_code	character varying(10)
country	character varying(15)
home_phone	character varying(24)
extension	character varying(4)
photo	bytea
notes	text
reports_to	smallint
photo_path	character varying(255)

Note. To determine which orders are late, you can use a comparison of the **required_date** and **shipped_date**.
Use the aliases **all_orders** and **late_orders** for the calculated columns.
You'll need to join the **employees** table to get the **last_name**, and also add COUNT to show the total late orders.

For example:

Тест	Result
-- Testing with original db	employee_id last_name all_orders late_orders ----------- --------- ---------- ----------- 1 Davolio 123 2 2 Fuller 96 4 3 Leverling 127 5 4 Peacock 156 10 5 Buchanan 42 1 6 Suyama 67 3 7 King 72 4 8 Callahan 104 4 9 Dodsworth 43 4

```
employee_id  last_name  all_orders  late_orders
-----------  ---------  ----------  -----------
1            Davolio    123         2
2            Fuller     96          4
3            Leverling  127         5
4            Peacock    156         10
5            Buchanan   42          1
6            Suyama     67          3
7            King       72          4
8            Callahan   104         4
9            Dodsworth  43          4
```

Відповідь: (penalty regime: 0 %)

```sql
SELECT
    e.employee_id,
    e.last_name,
    COUNT(o.order_id) AS all_orders,
    SUM(CASE
        WHEN o.shipped_date > o.required_date THEN 1
        ELSE 0
    END) AS late_orders
FROM employees e
JOIN orders o ON e.employee_id = o.employee_id
GROUP BY e.employee_id, e.last_name
HAVING SUM(CASE
            WHEN o.shipped_date > o.required_date THEN 1
            ELSE 0
        END) > 0
ORDER BY e.employee_id;

```

<table>
<tr><th></th><th>Тест</th><th>Expected</th><th>Got</th><th></th></tr>
<tr><td>✓</td><td><pre>-- Testing with original db</pre></td><td><pre>employee_id last_name all_orders late_orders
----------- --------- ---------- -----------
1   Davolio   123   2
2   Fuller   96   4
3   Leverling 127   5
4   Peacock   156   10
5   Buchanan   42   1
6   Suyama   67   3
7   King   72   4
8   Callahan   104   4
9   Dodsworth 43   4</pre></td><td><pre>employee_id last_name all_orders late_orders
----------- --------- ---------- -----------
1   Davolio   123   2
2   Fuller   96   4
3   Leverling 127   5
4   Peacock   156   10
5   Buchanan   42   1
6   Suyama   67   3
7   King   72   4
8   Callahan   104   4
9   Dodsworth 43   4</pre></td><td>✓</td></tr>
<tr><td>✓</td><td><pre>-- Testing with extra row</pre></td><td><pre>employee_id last_name all_orders late_orders
----------- --------- ---------- -----------
1   Davolio   123   2
2   Fuller   96   4
3   Leverling 127   5
5   Buchanan   42   1
6   Suyama   67   3
7   King   72   4
8   Callahan   104   4
9   Dodsworth 43   4</pre></td><td><pre>employee_id last_name all_orders late_orders
----------- --------- ---------- -----------
1   Davolio   123   2
2   Fuller   96   4
3   Leverling 127   5
5   Buchanan   42   1
6   Suyama   67   3
7   King   72   4
8   Callahan   104   4
9   Dodsworth 43   4</pre></td><td>✓</td></tr>
</table>

Пройшов усі тести! ✓

Правильно

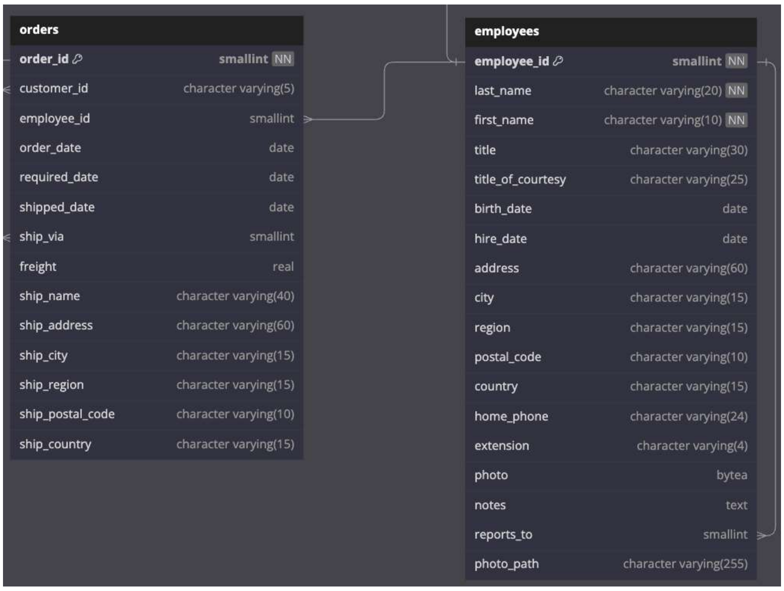
Бали за цю відповідь: 1,00/1,00.

Питання 9

Правильно

Балів 1,00 з 1,00

We know that Andrew Fuller is the Vice President of Northwind Company. Create the report that shows the list of those employees (**last_name** and **first_name**) who processed more orders than Fuller did.
Use the model:



orders	
order_id	smallint NN
customer_id	character varying(5)
employee_id	smallint
order_date	date
required_date	date
shipped_date	date
ship_via	smallint
freight	real
ship_name	character varying(40)
ship_address	character varying(60)
ship_city	character varying(15)
ship_region	character varying(15)
ship_postal_code	character varying(10)
ship_country	character varying(15)

employees	
employee_id	smallint NN
last_name	character varying(20) NN
first_name	character varying(10) NN
title	character varying(30)
title_of_courtesy	character varying(25)
birth_date	date
hire_date	date
address	character varying(60)
city	character varying(15)
region	character varying(15)
postal_code	character varying(10)
country	character varying(15)
home_phone	character varying(24)
extension	character varying(4)
photo	bytea
notes	text
reports_to	smallint
photo_path	character varying(255)

Note. Use the subqueries.

For example:

Тест	Result
-- Testing with original db	last_name first_name --------- ---------- Davolio Nancy Leverling Janet Peacock Margaret Callahan Laura

Відповідь: (penalty regime: 0 %)

```sql
SELECT e.last_name, e.first_name
FROM employees e
WHERE (
    SELECT COUNT(*)
    FROM orders o
    WHERE o.employee_id = e.employee_id
) > (
    SELECT COUNT(*)
    FROM orders o
    JOIN employees e2 ON o.employee_id = e2.employee_id
    WHERE e2.first_name = 'Andrew' AND e2.last_name = 'Fuller'
);

```

	Тест	Expected	Got	
✓	-- Testing with original db	last_name first_name --------- ---------- Davolio Nancy Leverling Janet Peacock Margaret Callahan Laura	last_name first_name --------- ---------- Davolio Nancy Leverling Janet Peacock Margaret Callahan Laura	✓
✓	-- Testing with extra row	last_name first_name --------- ---------- Huge Tomas Leverling Janet Peacock Margaret Callahan Laura	last_name first_name --------- ---------- Huge Tomas Leverling Janet Peacock Margaret Callahan Laura	✓

Пройшов усі тести! ✓

Правильно

Бали за цю відповідь: 1,00/1,00.

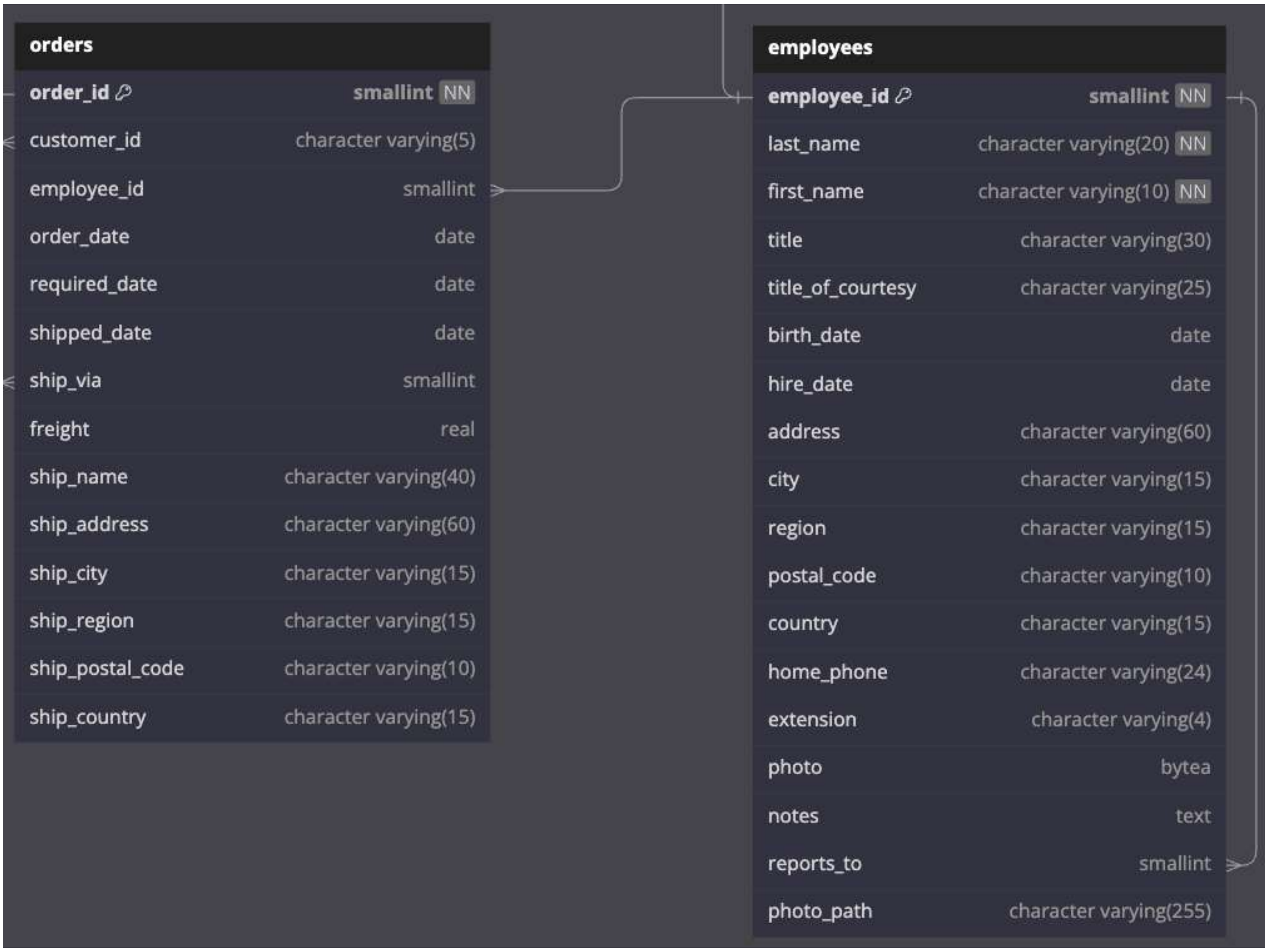
Питання 10

Правильно

Балів 1,00 з 1,00

Write the query that should return the **employee_id**, **order_id** and **order_date**. The criteria for the report is that the order must be the last for each employee (maximum **order_date**)

Use the model:



orders	
order_id	smallint NN
customer_id	character varying(5)
employee_id	smallint
order_date	date
required_date	date
shipped_date	date
ship_via	smallint
freight	real
ship_name	character varying(40)
ship_address	character varying(60)
ship_city	character varying(15)
ship_region	character varying(15)
ship_postal_code	character varying(10)
ship_country	character varying(15)

employees	
employee_id	smallint NN
last_name	character varying(20) NN
first_name	character varying(10) NN
title	character varying(30)
title_of_courtesy	character varying(25)
birth_date	date
hire_date	date
address	character varying(60)
city	character varying(15)
region	character varying(15)
postal_code	character varying(10)
country	character varying(15)
home_phone	character varying(24)
extension	character varying(4)
photo	bytea
notes	text
reports_to	smallint
photo_path	character varying(255)

Note. Use the correlated subquery.

For example:

Тест	Result
-- Testing with original db	employee_id order_id order_date ----------- -------- ---------- 1 11077 2016-05-11 2 11059 2016-05-18 3 11063 2016-05-05 4 11061 2016-05-19 5 11043 2016-04-27 6 11045 2016-04-28 7 11074 2016-05-11 8 11075 2016-05-11 9 11058 2016-05-04

Відповідь: (penalty regime: 0 %)

```
SELECT
    employee_id,
    order_id,
    order_date
FROM orders o1
WHERE order_date = (
    SELECT MAX(order_date)
    FROM orders o2
    FROM orders o2
    WHERE o2.employee_id = o1.employee_id
)
ORDER BY employee_id;

```

	Тест	Expected	Got	
✓	-- Testing with original db	employee_id order_id order_date ----------- -------- ---------- 1 11077 2016-05-11 2 11059 2016-05-18 3 11063 2016-05-05 4 11061 2016-05-19 5 11043 2016-04-27 6 11045 2016-04-28 7 11074 2016-05-11 8 11075 2016-05-11 9 11058 2016-05-04	employee_id order_id order_date ----------- -------- ---------- 1 11077 2016-05-11 2 11059 2016-05-18 3 11063 2016-05-05 4 11061 2016-05-19 5 11043 2016-04-27 6 11045 2016-04-28 7 11074 2016-05-11 8 11075 2016-05-11 9 11058 2016-05-04	✓
✓	-- Testing with extra row	employee_id order_id order_date ----------- -------- ---------- 1 11077 2016-05-11 2 11059 2016-05-18 3 11063 2016-05-05 4 11061 2016-05-19 5 11043 2016-04-27 6 11045 2016-04-28 7 11074 2016-05-11 8 11075 2016-05-11 9 11022 2016-04-19	employee_id order_id order_date ----------- -------- ---------- 1 11077 2016-05-11 2 11059 2016-05-18 3 11063 2016-05-05 4 11061 2016-05-19 5 11043 2016-04-27 6 11045 2016-04-28 7 11074 2016-05-11 8 11075 2016-05-11 9 11022 2016-04-19	✓

Пройшов усі тести! ✓

Правильно

Бали за цю відповідь: 1,00/1,00.