

8) Создать индексы для таблиц (для которых это целесообразно), аргументировав выбор поля, по которому будет создан индекс.

```
1. CREATE index "nessname_goods"
ON pm.goods_information(name_goods)
```

Индекс "nessname_goods" будет использоваться для быстрого нахождения товара с определённым названием, то есть это позволяет сразу находить, кто из поставщиков предоставил данный вид товара и читать информацию о нём. В конкретном примере удобно пользоваться индексами, так как это не постоянно меняющиеся данные.

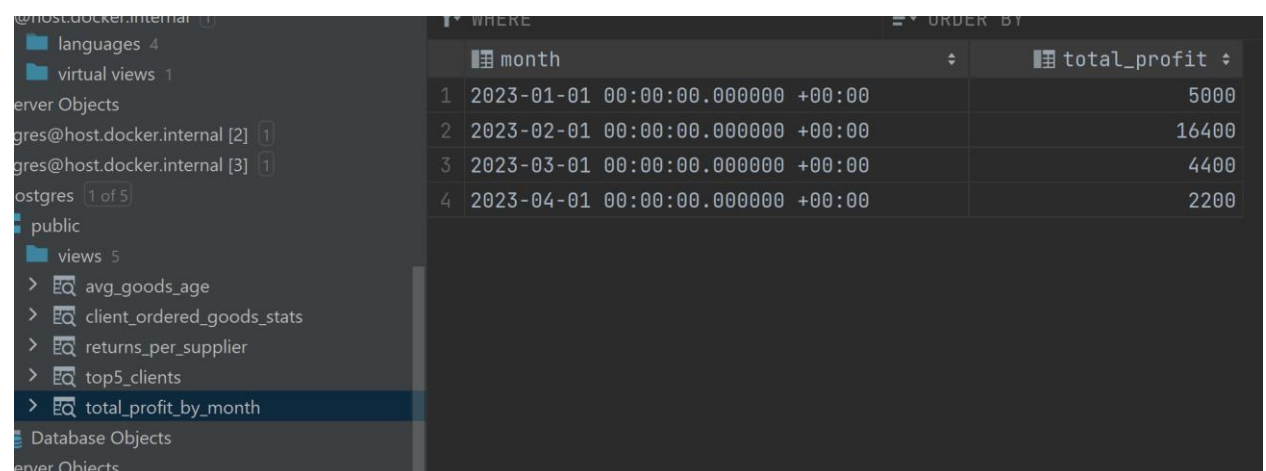
```
2. CREATE index "main_reason_return"
ON pm.order_return(reason_return)
```

Индекс "main_reason_return" создан для быстрого нахождения причин возвратов товаров. Это позволяет оперативно оценивать из-за чего именно произошел возврат, то есть это связано с браком товара или непосредственно с плохой доставкой.

9) Подготовить не менее 6 представлений

1. Суммарная прибыль с учетом возвратов за каждый месяц:

```
CREATE VIEW total_profit_by_month AS
SELECT DATE_TRUNC('month', time_order) AS month, SUM(cost_order)
- SUM(cost_return) AS total_profit
FROM pm.Orders_Archive
LEFT JOIN pm.Order_Return ON Orders_Archive.id_order =
Order_Return.id_order
GROUP BY month
ORDER BY month ASC
```

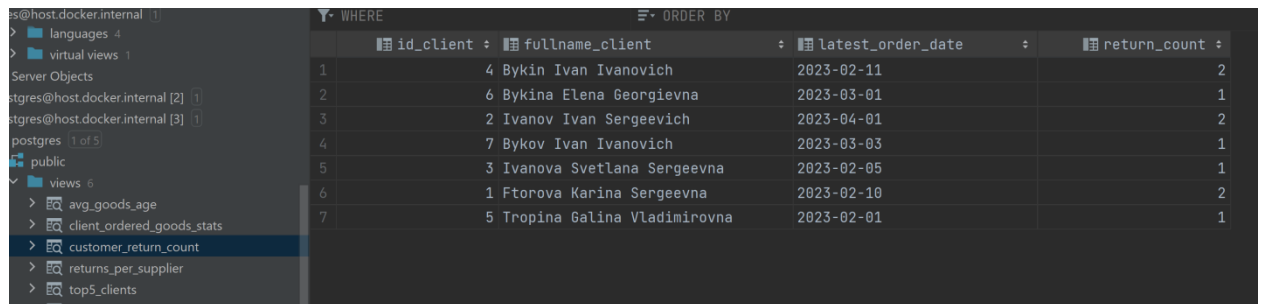


The screenshot shows a database management tool interface. On the left, a tree view displays the database structure, with the view 'total_profit_by_month' selected under the 'views' folder. The main pane on the right displays the data for this view, showing a table with two columns: 'month' and 'total_profit'. The data is sorted by month in ascending order.

	month	total_profit
1	2023-01-01 00:00:00.000000 +00:00	5000
2	2023-02-01 00:00:00.000000 +00:00	16400
3	2023-03-01 00:00:00.000000 +00:00	4400
4	2023-04-01 00:00:00.000000 +00:00	2200

2. Представление списка клиентов с датой последнего заказа и количеством возвращенных товаров :

```
CREATE VIEW customer_return_count AS
SELECT
    c.id_client,
    c.fullname_client,
    MAX(o.time_order) AS latest_order_date,
    COUNT(r.id_order) AS return_count
FROM
    pm.client AS c
    INNER JOIN pm.orders_archive AS o ON c.id_client =
o.id_client
    LEFT JOIN pm.order_return AS r ON o.id_order = r.id_order
GROUP BY
    c.id_client,
    c.fullname_client;
```



The screenshot shows a database client interface. On the left, a tree view of database objects is visible, with 'views' expanded and 'customer_return_count' selected. The main pane displays the data from this view, sorted by 'id_client'.

	id_client	fullname_client	latest_order_date	return_count
1	4	Bykin Ivan Ivanovich	2023-02-11	2
2	6	Bykina Elena Georgievna	2023-03-01	1
3	2	Ivanov Ivan Sergeevich	2023-04-01	2
4	7	Bykov Ivan Ivanovich	2023-03-03	1
5	3	Ivanova Svetlana Sergeevna	2023-02-05	1
6	1	Ftorova Karina Sergeevna	2023-02-10	2
7	5	Tropina Galina Vladimirovna	2023-02-01	1

3. Представление для отображения количества товара, заказанного каждым клиентом, и процентного соотношения количества этого товара к общему количеству заказов клиента:

```
CREATE VIEW Client_Ordered_Goods_Stats AS
SELECT id_client, name_goods, SUM(amount_goods) AS total_amount,
(SUM(amount_goods)*100)/(SELECT SUM(amount_goods) FROM
pm.goods_order) AS percent_of_total
FROM pm.goods_order
INNER JOIN pm.order ON goods_order.id_order = pm.order.id_order
GROUP BY id_client, name_goods;
```

The screenshot shows a PostgreSQL database interface with a query result for a view named 'client_ordered_goods_stats'. The view is located under the 'public' schema in the 'views' folder. The query result is displayed in a table with the following columns: 'id_client', 'name_goods', 'total_amount', and 'percent_of_total'. The data is ordered by 'total_amount' in descending order.

id_client	name_goods	total_amount	percent_of_total
1	7 Puzzles version 2	2	9
2	6 Puzzles	3	14
3	3 Kids sunglasses	2	9
4	2 Toy car	3	14
5	4 Markers	1	4
6	2 Puzzles	2	9
7	5 Puzzles version 2	3	14
8	4 Toy car	1	4
9	1 Puzzles	4	19

4. Топ-5 клиентов по общей стоимости заказов:

```
CREATE VIEW top5_clients AS
SELECT fullname_client, SUM(cost_order) AS total_cost
FROM pm.orders_archive oa
JOIN pm.client c ON oa.id_client = c.id_client
GROUP BY fullname_client
ORDER BY total_cost DESC
LIMIT 5;
```

The screenshot shows a PostgreSQL database interface with a query result for a view named 'top5_clients'. The view is located under the 'public' schema in the 'views' folder. The query result is displayed in a table with the following columns: 'fullname_client' and 'total_cost'. The data is ordered by 'total_cost' in descending order.

fullname_client	total_cost
Ftorova Karina Sergeevna	10500
Tropina Galina Vladimirovna	5000
Ivanov Ivan Sergeevich	4900
Bykin Ivan Ivanovich	4700
Ivanova Svetlana Sergeevna	4000

5. Среднее количество приобретенных товаров в зависимости от возраста клиента:

```
CREATE VIEW avg_goods_age AS
SELECT EXTRACT(YEAR FROM AGE(datage_client)) AS age,
AVG(amount_gclient) AS avg_goods
FROM pm.client
GROUP BY age
ORDER BY age;
```

The screenshot shows a PostgreSQL database interface. On the left, the 'Server Objects' tree is expanded to show the 'public' schema, where the view 'avg_goods_age' is selected. On the right, the view's data is displayed in a table with columns 'age' and 'avg_goods'.

	age	avg_goods
1	19	25
2	21	72.5
3	22	150
4	24	506.5
5	34	32
6	43	86.5
7	48	77

6. "Количество возвратов товаров для каждого поставщика":

```
CREATE VIEW returns_per_supplier AS
SELECT fullname_supplier, COUNT(*) AS num_returns
FROM pm.Supplier
JOIN pm.Order ON id_supplier = supplier.id_supplier
JOIN pm.Order_Return r ON r.id_order = pm.order.id_order
GROUP BY fullname_supplier
ORDER BY num_returns DESC;
```

The screenshot shows a PostgreSQL database interface. On the left, the 'Server Objects' tree is expanded to show the 'public' schema, where the view 'returns_per_supplier' is selected. On the right, the view's data is displayed in a table with columns 'fullname_supplier' and 'num_returns'.

	fullname_supplier	num_returns
1	Torova Anastasia Sergeevna	10
2	Grozd Nikolay Ivanovich	10
3	Grozdov Nikolay Sergeevich	10
4	Ivanov Ivan Dmitrievich	10
5	Frolova Karina Viktorovna	10
6	Robov Konstantin Denisovich	10
7	Ivanova Karina Vladimirovna	10
8	Kolov Sergey Sergeevich	10
9	Rybov Anna Georgievna	10
10	Grozdin Nikolay Sergeevich	10
11	Smirov Egor Alexandrovich	10

10) Создать не менее 2 хранимых процедур/функций.

1. Процедура для обновления количества товара, которое предоставил поставщик:

```
CREATE PROCEDURE update_client_gpa
(id_currclient INT, new_gpa DOUBLE) AS $$
UPDATE pm.client
SET gpa_allclient = new_gpa
WHERE id_client = id_currclient
$$ LANGUAGE sql;
```

2. Функция, предназначенная для добавления нового поставщика:

```
CREATE OR REPLACE FUNCTION add_supplier(
    IN name_goods VARCHAR(64),
    IN costsup_goods INTEGER,
    IN grasup_definite DOUBLE,
    IN amount_allgoods INTEGER)
    RETURN VOID AS $$
    BEGIN
        INSERT INTO pm.goods_Information (name_goods, costsup_goods,
        grasup_definite, amount_allgoods)
        VALUES (name_goods, costsup_goods, grasup_definite,
        amount_allgoods);
        COMMIT;
    END;
    $$ LANGUAGE sql;
```