

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ  
Харківський національний університет радіоелектроніки

Кафедра системотехніки

**ЗВІТ**  
з виконання завдань практичного заняття № 6  
дисципліни «Проектування високонавантажених систем  
зберігання даних»  
на тему: «СТВОРЕННЯ Й ВИКОРИСТАННЯ УЯВЛЕНЬ ДЛЯ  
ВИСОКОНАВАНТАЖЕНИХ БАЗ ДАНИХ  
НА ПЛАТФОРМІ СУБД MySQL»

Виконав  
студент WILDAU - KHARKIV  
Якунін Ігор

Перевірив  
професор кафедри СТ  
Колесник Л.В.

Харків, 2024

## 4.1 Мета заняття

- набуття практичних навичок зі створення тимчасових таблиць, що використовуються як джерело даних SQL-запитів на вибірку й модифікацію даних;
- набуття практичних навичок з розробки уявлень (VIEW), що використовуються як джерело даних SQL-запитів на вибірку й модифікацію даних;
- набуття практичних навичок з розробки SQL-запитів на вибірку й модифікації даних з використанням уявлень (VIEW) і тимчасових таблиць для забезпечення основних бізнес-процесів високонавантаженої інформаційної системи;
- формування необхідних практичних умінь для аналізу плану виконання SQL-запитів до уявлень (VIEW) за допомогою оператора EXPLAIN;
- формування необхідних практичних умінь для створення уявлень (VIEW), з урахуванням особливостей роботи високонавантажених інформаційних систем зберігання даних..

## Завдання на самостійну роботу

Обрана така область – «Сайт сервісного центру по ремонту техніки»

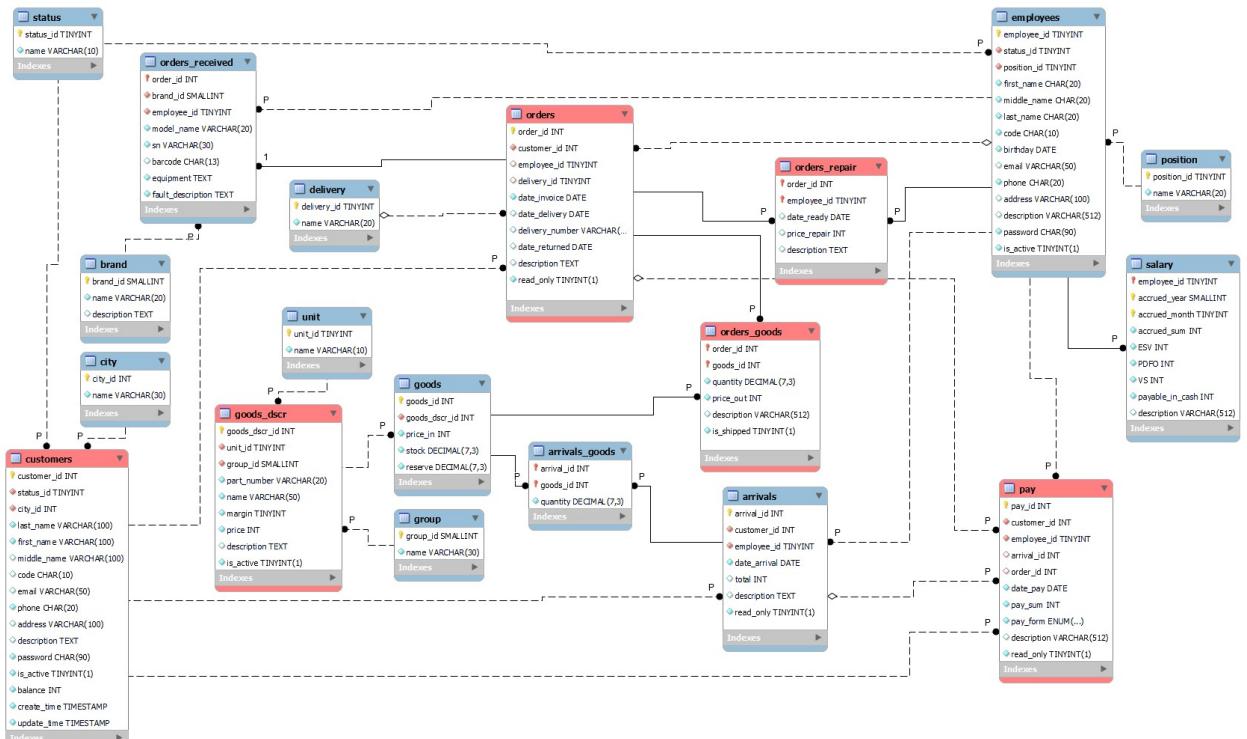


Рис. 6.1. Скріншот схеми фізичної моделі бази даних з таблицями типу InnoDB

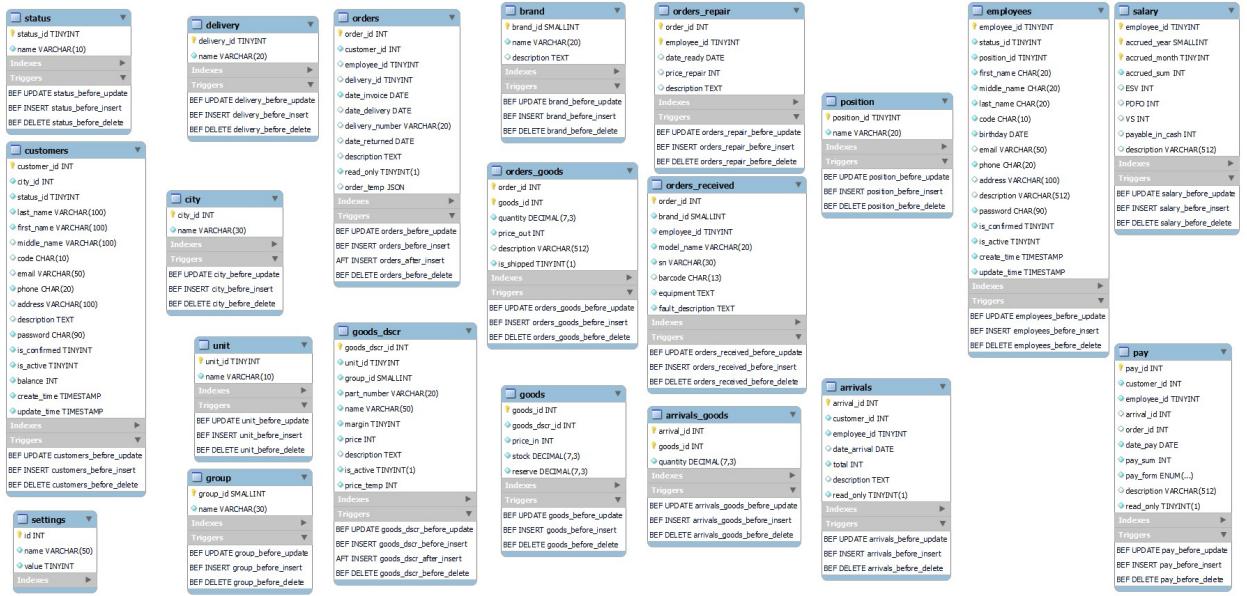


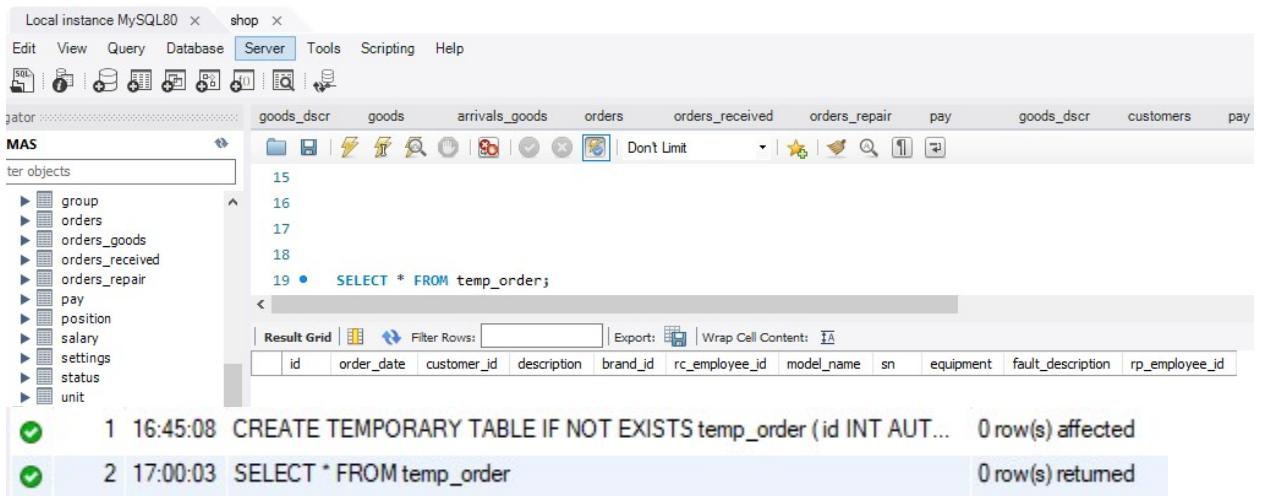
Рис. 6.2. Скріншот схеми фізичної моделі бази даних з таблицями типу MyIsam

**Завдання 6.1.** Для бази даних з таблицями MyIsam, розробленої відповідно до завдання 1.2 практичного заняття № 1, створити тимчасову (TEMPORARY) таблицю. Дозволяється продублювати дані однієї з існуючих таблиць бази даних. Створити чотири SQL-запити до тимчасової таблиці з операторами SELECT, UPDATE, DELETE і INSERT. Протестувати доступ до тимчасової таблиці, створивши кілька клієнтських підключень (сесій), за допомогою середовища інструментів для розробки баз даних Workbench.

Код SQL-запиту, що використовується для створення тимчасової таблиці:

```
CREATE TEMPORARY TABLE IF NOT EXISTS temp_order (
    id INT AUTO_INCREMENT,
    order_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    customer_id INT unsigned,
    description VARCHAR(255),
    brand_id SMALLINT unsigned,
    rc_employee_id TINYINT unsigned,
    model_name VARCHAR(20),
    sn VARCHAR(30),
    equipment VARCHAR(255),
    fault_description VARCHAR(255),
    rp_employee_id TINYINT unsigned,
    PRIMARY KEY (id)
);
```

Створимо таблицю, та одразу перевіримо, чи є доступ до таблиці (рис. 6.3):



Local instance MySQL80 x shop x

Edit View Query Database **Server** Tools Scripting Help

Navigator: goods\_descr goods arrivals\_goods orders orders\_received orders\_repair pay goods\_descr customers pay

MAS

15  
16  
17  
18  
19 • `SELECT * FROM temp_order;`

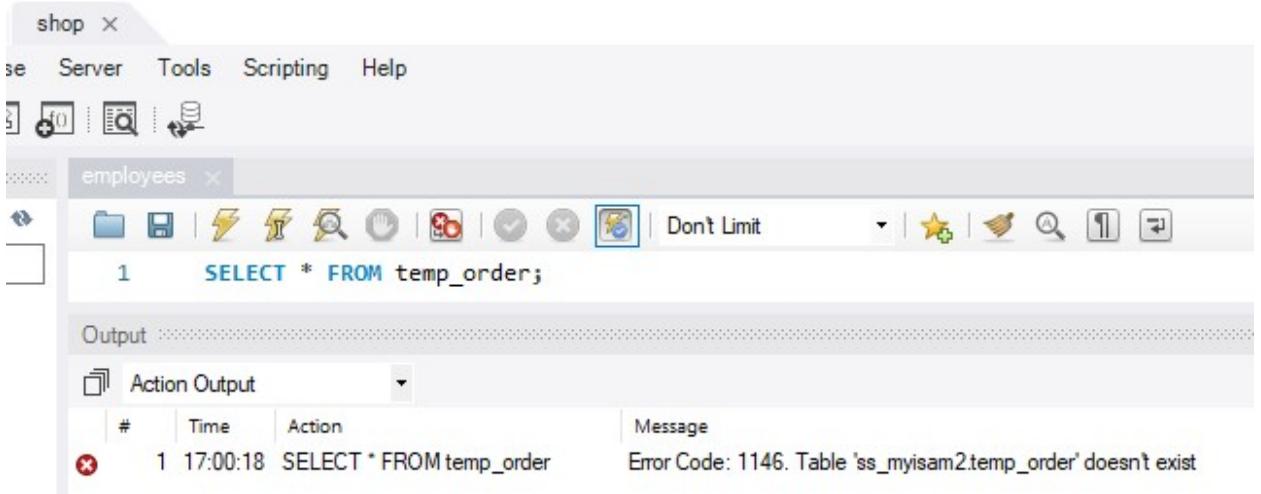
Result Grid | Filter Rows: Export: Wrap Cell Content: `id order_date customer_id description brand_id rc_employee_id model_name sn equipment fault_description rp_employee_id`

1 16:45:08 CREATE TEMPORARY TABLE IF NOT EXISTS temp\_order (id INT AUT... 0 row(s) affected

2 17:00:03 SELECT \* FROM temp\_order 0 row(s) returned

Рис. 6.3. Скріншот створення тимчасової таблиці

Перевіримо, чи є доступ у іншого користувача (іншої сесії) до цієї таблиці (рис.6.4):



shop x

File Server Tools Scripting Help

employees x

1 `SELECT * FROM temp_order;`

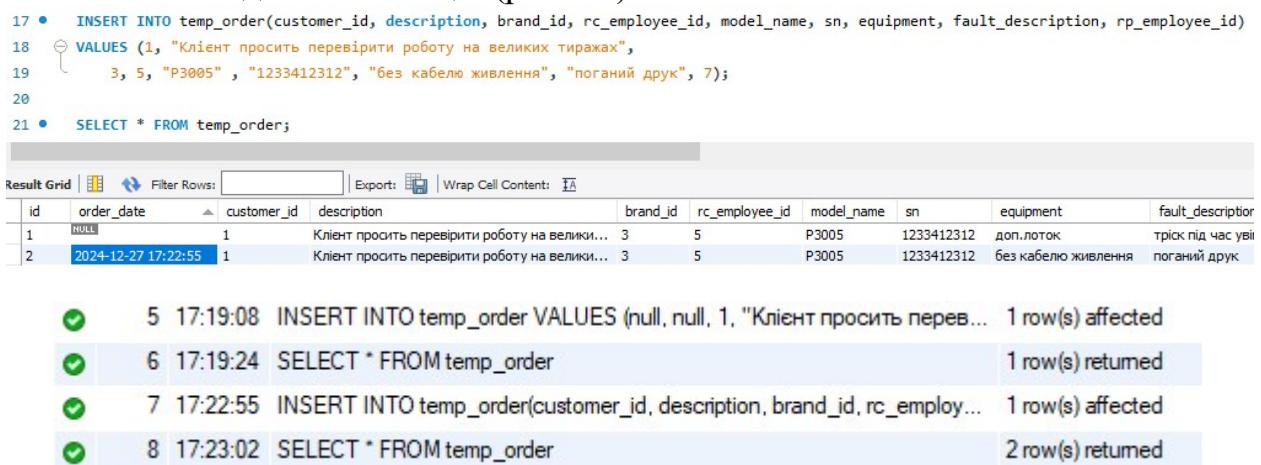
Output

Action Output

#	Time	Action	Message
1	17:00:18	<code>SELECT * FROM temp_order</code>	Error Code: 1146. Table 'ss_myisam2.temp_order' doesn't exist

Рис. 6.4. Скріншот перевірки доступу до таблиці з іншої сесії

Внесемо дані в таблицю (рис.6.5):



17 • `INSERT INTO temp_order(customer_id, description, brand_id, rc_employee_id, model_name, sn, equipment, fault_description, rp_employee_id)`

18 `VALUES (1, "Клієнт просить перевірити роботу на великих тиражах",`

19 `3, 5, "P3005", "1233412312", "без кабелю живлення", "поганий друк", 7);`

20

21 • `SELECT * FROM temp_order;`

Result Grid | Filter Rows: Export: Wrap Cell Content: `id order_date customer_id description brand_id rc_employee_id model_name sn equipment fault_description rp_employee_id`

<code>id</code>	<code>order_date</code>	<code>customer_id</code>	<code>description</code>	<code>brand_id</code>	<code>rc_employee_id</code>	<code>model_name</code>	<code>sn</code>	<code>equipment</code>	<code>fault_description</code>
1	NULL	1	Клієнт просить перевірити роботу на великих тиражах	3	5	P3005	1233412312	доп.лоток	тріск під час уві
2	2024-12-27 17:22:55	1	Клієнт просить перевірити роботу на великих тиражах	3	5	P3005	1233412312	без кабелю живлення	поганий друк

5 17:19:08 `INSERT INTO temp_order VALUES (null, null, 1, "Клієнт просить перев... 1 row(s) affected`

6 17:19:24 `SELECT * FROM temp_order 1 row(s) returned`

7 17:22:55 `INSERT INTO temp_order(customer_id, description, brand_id, rc_employee_id, model_name, sn, equipment, fault_description, rp_employee_id)`

8 17:23:02 `SELECT * FROM temp_order 2 row(s) returned`

Рис. 6.5. Скріншот внесення даних до тимчасової таблиці

Оновимо дані в таблиці (рис.6.6):

```

22 • UPDATE temp_order
23   SET order_date = '2024-12-27 17:20'
24   WHERE id = 1;
25
26 • SELECT * FROM temp_order;

```

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

id	order_date	customer_id	description	brand_id	rc_employee_id	model_name	sn	equipment	fault_des
1	2024-12-27 17:20:00	1	Клієнт просить перевірити роботу на велики...	3	5	P3005	1233412312	доп.поток	тріск під
2	2024-12-27 17:22:55	1	Клієнт просить перевірити роботу на велики...	3	5	P3005	1233412312	без кабелю живлення	поганий.

9 19:13:55 UPDATE temp\_order SET order\_date = '2024-12-27 17:20' WHERE id = 1 1 row(s) affected Rows matched: 1 Changed: 1

10 19:14:10 SELECT \* FROM temp\_order 2 row(s) returned

Рис. 6.6. Скріншот оновлення даних в тимчасової таблиці

Перевіримо, чи з'явилися такі ж дані в основної таблиці orders у customer\_id = 1 (рис. 6.7):

```

30 • SELECT * FROM orders WHERE customer_id = 1 ORDER BY order_id DESC;

```

Result Grid | Filter Rows: | Edit: | Export/Import: |

order_id	customer_id	emp	deli	date_invoice	date_delivery	delivery_number	date_returned	date_voided	date_voided
10019	1	11	1	2024-11-21	2024-11-23	23421525235	2024-11-21	NULL	NULL
10017	1	NULL	NULL	2024-11-21	NULL	NULL	NULL	NULL	NULL

Рис. 6.7. Скріншот таблиці orders

Бачимо що даних від 2024-12-27 немає.

Видалимо дані з id = 2 з таблиці temp\_order(рис.6.8):

```

26 • DELETE FROM temp_order WHERE id = 2;
27 • SELECT * FROM temp_order;

```

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

id	order_date	customer_id	description	brand_id	rc_employee_id	model_name	sn	equipment	fault_des	rp_e
1	2024-12-27 17:20:00	1	Клієнт просить перевірити роботу на велики...	3	5	P3005	1233412312	доп.поток	тріск під час увімкнення	7

11 19:20:51 DELETE FROM temp\_order WHERE id = 2 1 row(s) affected

12 19:20:55 SELECT \* FROM temp\_order 1 row(s) returned

Рис. 6.8. Скріншот видалення даних з тимчасової таблиці

Видалення тимчасової таблиці (рис.6.9):

DROP TEMPORARY TABLE IF EXISTS temp\_order;  
SELECT \* FROM temp\_order;

```

13 19:23:33 DROP TEMPORARY TABLE IF EXISTS temp_order 0 row(s) affected
14 19:23:38 SELECT * FROM temp_order Error Code: 1146. Table 'ss_myisam2.temp_order' doesn't exist

```

Рис. 6.9. Скріншот видалення тимчасової таблиці

**Завдання 6.2.** Для бази даних з таблицями MyIsam, розробленої відповідно до завдання 1.2 практичного заняття № 1, розробити перелік уявлень (VIEW) і пов’язаних з ними SQL-запитів, що забезпечують роботу з інтерфейсом доступу до бази даних високонавантаженої інформаційної системи.

Таблиця 6.2 – Уявлення для високонавантаженої системи

№	Призначення уявлення	Алгоритм, ім’я уявлень	Найменування таблиць
1	Отримання інформації про замовлення	MERGE view_orders	orders, orders_received, orders_repair, orders_goods, customers
2	Отримання інформації щодо замовлення на запчастини	MERGE view_orders_goods	view_orders
3	Вставка даних	MERGE view_orders	orders, orders_items
4	Отримати інформацію щодо запчастин	TEMPTABLE view_goods	goods, goods_dscr
5	Отримання повної інформації за id запчастини	TEMPTABLE view_goods_sum	view_goods

**Завдання 6.3.** Відповідно до переліку завдання 6.2 для бази даних з таблицями типу MyIsam розробити:

- SQL-запит, що визначає уявлення (VIEW). SQL-запит має використовувати дані декількох базових таблиць, пов’язаних з допомогою інструкції JOIN;
- SQL-запит, що визначає уявлення (VIEW). SQL-запит має використовувати дані декількох базових таблиць, містити групові операції GROUP BY і агреговані функції. Зв’язок між таблицями в запиті реалізувати за допомогою інструкції JOIN;
- з використанням визначальних SQL-запитів, створити два базових уявлення, що використовують алгоритми MERGE і TEMPTABLE;
- розробити 4–7 SQL-запитів для кожного базового уявлення;
- використовуючи розроблені SQL-запити до уявлень, як визначальні,

створити нові уявлення, які як джерело даних використовують базові уявлення;

– провести аналіз плану виконання розроблених SQL-запитів, що визначають уявлення, і SQL-запитів до створених уявлень за допомогою оператора EXPLAIN.

**6.3.1.** код SQL-запиту CREATE VIEW, який створює базове уявлення VIEW 1 і використовує алгоритм MERGE;

```
CREATE OR REPLACE ALGORITHM = MERGE VIEW view_orders AS
SELECT order_id, customer_id, delivery_id, date_invoice, date_delivery,
       delivery_number, date_returned, brand_id, model_name, sn, barcode, equipment,
       fault_description, orp.employee_id, date_ready, price_repair, orp.description,
       goods_id, quantity, price_out, is_shipped,
       last_name, first_name, middle_name, code, email, phone, address
  FROM orders
 JOIN orders_received USING(order_id)
 JOIN orders_repair orp USING(order_id)
 JOIN orders_goods USING(order_id)
 JOIN customers USING(customer_id);
```

31 22:52:43 CREATE OR REPLACE ALGORITHM = MERGE VIEW view\_orders AS ... 0 row(s) affected

Рис. 6.10. Скріншот створення уявлення

**6.3.1.1.** код SQL-запиту , що повертає всі відвантажені замовлення клієнта з id = 1 (рис.6.11):

```
SELECT * FROM view_orders
 WHERE customer_id = 1 and date_returned IS NOT NULL;
```

5 • SELECT * FROM view_orders WHERE customer_id = 1 and date_returned IS NOT NULL;													
Result Grid   Filter Rows:   Export:   Wrap Cell Content:													
order_id	customer_id	delivery_id	date_invoice	date_delivery	delivery_number	date_returned	brand_id	model_name	sn	barcode	equipment	fault_description	
1272	1	3	2024-02-01	2024-03-03	446383345	2024-03-02	4	3662	1563784	NULL	Рішення гол...	Мигнути здригатис...	
1272	1	3	2024-02-01	2024-03-03	446383345	2024-03-02	4	3662	1563784	NULL	Рішення гол...	Мигнути здригатис...	
date_ready	price_repair	description	goods_id	quantity	price_out	is_shipped	last_name	first_name	middle_name	code	email	phone	address
2024-03-02	104085	Способ легко править гостодіння набір провінці...	2915	1.417	7919	1	Бандера	Симон	Венедиктович	4229210818	vitalii82@meta.ua	080 114-27-66	проеулок Сунський, буд. 82 кв. 1
2024-03-02	104085	Способ легко править гостодіння набір провінці...	7554	1.201	11258	1	Бандера	Симон	Венедиктович	4229210818	vitalii82@meta.ua	080 114-27-66	проеулок Сунський, буд. 82 кв. 1

Рис. 6.11. Скріншот SQL-запиту

Бачимо, що клієнт на ім'я Бандера Симон здав в ремонт 2024-02-01 виріб на ім'я моделі «3662» з sn = 1563784, замовлення №1272. Отримав його з ремонту 2020-03-03, ремонт коштував 1040.85 грн плюс за запчастини  $(1.417*7919+1.201*11258)/100=247.42$  грн. Також бачимо, що виріб було відправлено кур'єром 2024-03-03, номер декларації 446383345.

**6.3.1.2.** код SQL-запиту , що повертає всі не повернуті клієнтам замовлення(рис.6.12):

SELECT \* FROM view\_orders WHERE date\_returned IS NULL order by order\_id desc;

5 • SELECT * FROM view_orders WHERE date_returned IS NULL order by order_id desc;													
Result Grid		Filter Rows: <input type="text"/> Export:  Wrap Cell Content:											
order_id	customer_id	deli	date_invoice	date_delivery	delivery_number	date_returned	brand_id	model_name	sn	bai	equipment	fault_description	
1850	9112	2	2024-10-07	2024-10-24	677515181	NULL	1	8568	2488944	NULL	Взагалі рішення...	Порода скинуті си...	
1850	9112	2	2024-10-07	2024-10-24	677515181	NULL	1	8568	2488944	NULL	Взагалі рішення...	Порода скинуті си...	
1850	9112	2	2024-10-07	2024-10-24	677515181	NULL	1	8568	2488944	NULL	Взагалі рішення...	Порода скинуті си...	
798	2353	1	2024-10-21	2024-10-23	768248676	NULL	4	4657	1697292	NULL	Мить художній ...	Щур актриса банк ...	
798	2353	1	2024-10-21	2024-10-23	768248676	NULL	4	4657	1697292	NULL	Мить художній ...	Щур актриса банк ...	
Result Grid													
date_ready	price_repair	description	goods_id	quantity	price_out	is_shipped	last_name	first_name	middle_name	code	email	phone	address
2024-10-23	80686	Число сплатит...	6323	1,442	7934	1	Тагибок	Захар	Пантелеїмонович	3980886101	davyd13@meta.ua	+380 09 065 69 71	вулиця Якірний, буд. 32
2024-10-23	80686	Число сплатит...	7398	0,446	3483	1	Тагибок	Захар	Пантелеїмонович	3980886101	davyd13@meta.ua	+380 09 065 69 71	вулиця Якірний, буд. 32
2024-10-23	80686	Число сплатит...	8310	0,332	5183	1	Тагибок	Захар	Пантелеїмонович	3980886101	davyd13@meta.ua	+380 09 065 69 71	вулиця Якірний, буд. 32
2024-10-22	120253	Сласти підкі... 4305	4305	0,701	5018	1	Венгринович	Онісім	Давидович	3337721600	azhuk@ukr.net	790 79 27	площа Оборони Ленінграду
2024-10-22	120253	Сласти підкі... 9734	9734	1,383	13428	1	Венгринович	Онісім	Давидович	3337721600	azhuk@ukr.net	790 79 27	площа Оборони Ленінграду

Рис. 6.12. Скріншот SQL-запиту

Маємо два замовлення №№798, 1850 від 2024-10-21, 2024-10-07 відповідно, вказані суми ремонтів, та які витрачени запчастини.

**6.3.1.3.** код SQL-запиту , що повертає замовлення по його id (рис.6.13):

SELECT \* FROM view\_orders WHERE order\_id = 1;

17 • SELECT * FROM view_orders WHERE order_id = 1;														
Result Grid		Filter Rows: <input type="text"/> Export:  Wrap Cell Content:												
order_id	customer_id	delivery_id	date_invoice	date_delivery	delivery_number	date_returned	brand_id	model_name	sn	barcode	equipment	fault_descripti		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
1	441	3	2024-08-22	2024-09-08	198588106	2024-09-07	2	6100	6122857	NULL	Гуляти п...	Шкарпетк...		
Result Grid														
emplc	date_ready	price_repair	description	goods_id	quantity	price_out	is_s	last_name	first_name	middle	code	email	phone	address
19	2024-09-11	23000	Груди палат...	8568	0.500	3300	1	Атаманчук	Наталія	Ста...	389...	zaleksiuk@ukr.net	058 743-43-29	проспект Віри Фігнер, буд. 1 кв. 438
19	2024-09-11	23000	Груди палат...	5216	0.389	3831	1	Атаманчук	Наталія	Ста...	389...	zaleksiuk@ukr.net	058 743-43-29	проспект Віри Фігнер, буд. 1 кв. 438
19	2024-09-11	23000	Груди палат...	886	1.834	7230	1	Атаманчук	Наталія	Ста...	389...	zaleksiuk@ukr.net	058 743-43-29	проспект Віри Фігнер, буд. 1 кв. 438
19	2024-09-11	23000	Груди палат...	1	1.000	0	1	Атаманчук	Наталія	Ста...	389...	zaleksiuk@ukr.net	058 743-43-29	проспект Віри Фігнер, буд. 1 кв. 438
20	2024-09-07	37000	Ліловий боліс...	8568	0.500	3300	1	Атаманчук	Наталія	Ста...	389...	zaleksiuk@ukr.net	058 743-43-29	проспект Віри Фігнер, буд. 1 кв. 438
20	2024-09-07	37000	Ліловий боліс...	5216	0.389	3831	1	Атаманчук	Наталія	Ста...	389...	zaleksiuk@ukr.net	058 743-43-29	проспект Віри Фігнер, буд. 1 кв. 438
20	2024-09-07	37000	Ліловий боліс...	886	1.834	7230	1	Атаманчук	Наталія	Ста...	389...	zaleksiuk@ukr.net	058 743-43-29	проспект Віри Фігнер, буд. 1 кв. 438
20	2024-09-07	37000	Ліловий боліс...	1	1.000	0	1	Атаманчук	Наталія	Ста...	389...	zaleksiuk@ukr.net	058 743-43-29	проспект Віри Фігнер, буд. 1 кв. 438

Рис. 6.13. Скріншот SQL-запиту

Маємо клієнта Атаманчук Наталія, замовлення №1, від 2024-08-22, було два ремонти, готовність 7/11.09.2024, використано 4 запчастини, вказані суми запчастин та ремонтів, є дата відвантаження, та відправка кур'єром.

**6.3.1.4.** код SQL-запиту , що редагує попередній результат помилкової дати другого ремонту(рис.6.14):

```
UPDATE view_orders
SET date_ready = '2024-09-08'
WHERE order_id = 1 and employee_id = 19;
```

16 14:49:26 UPDATE view\_orders SET date\_ready = '2024-09-08' WHERE order\_id = 1 and employee\_id = 19 Error Code: 1644. Помилка, orders\_repair: вказаний order\_id не існує в таблиці orders, або вже відружен Замовнику.

Рис. 6.14. Скріншот UPDATE запиту

Отримали отказ в редагуванні вже відвантаженого замовлення, спрацювання тригера для таблиці order\_repair. Чудовий результат, який підтверджує, що уявлення не самостійна таблиця, а тільки представлення деяких таблиць.

**6.3.1.5.** Повторимо експеримент для ще відкритого замовлення, використаємо замовлення з підрозділу **6.3.1.2** (рис. 6.15):

```
UPDATE view_orders
SET date_returned = '2024-12-28'
WHERE order_id = 798;
```

17 15:00:23 UPDATE view\_orders SET date\_returned = '2024-12-28' WHERE order\_id = 798 1 row(s) affected Rows matched: 1 Changed: 1  
17 • SELECT \* FROM view\_orders WHERE order\_id = 798;

order_id	customer_id	delivery_id	date_invoice	date_delivery	delivery_number	date_returned	brand_id	model_name	...
798	2353	1	2024-10-21	2024-10-23	768248676	2024-12-28	4	4657	1
798	2353	1	2024-10-21	2024-10-23	768248676	2024-12-28	4	4657	1

Рис. 6.15. Скріншот UPDATE запиту

**6.3.1.6.** Спробуємо виправити дані в декількох полях(рис. 6.16-6.17):

```
UPDATE view_orders
SET equipment = 'немає бокового лотка', fault_description = "не подає папір", description =
"ремонт подачі паперу"
WHERE order_id = 1850;
```

20 15:13:20 UPDATE view\_orders SET equipment = 'немає бокового лотка', fault\_des... Error Code: 1393. Can not modify more than one base table through a join view 'ss\_myisam2.view\_orders'

Рис. 6.16. Скріншот UPDATE запиту

Розіб'ємо запит на два (за таблицями):

```
UPDATE view_orders
SET equipment = 'немає бокового лотка', fault_description = "не подає папір"
```

```
WHERE order_id = 1850;
```

```
UPDATE view_orders
```

```
SET description = "ремонт подачі паперу"
```

```
WHERE order_id = 1850 and employee_id = 15;
```

```
21 15:16:27 UPDATE view_orders SET equipment = 'немає бокового лотка', fault_des... 1 row(s) affected Rows matched: 1 Changed: 1
```

```
25 15:18:12 UPDATE view_orders SET description = "ремонт подачі паперу" WHERE ... 1 row(s) affected Rows matched: 1 Changed: 1
```

```
17 • SELECT * FROM view_orders WHERE order_id = 1850;
```

order_id	custom	deliv	date_invoice	date_delivery	deliv...	date_r	brand_id	model_name	sn	barcode	equipment	fault_descripti...	empid	date_ready	price_r...	description	good...
1850	9112	2	2024-10-07	2024-10-24	67...	NULL	1	8568	2488944	NULL	немає б...	не подає п...	15	2024-10-23	80686	ремонт пода...	8310
1850	9112	2	2024-10-07	2024-10-24	67...	NULL	1	8568	2488944	NULL	немає б...	не подає п...	15	2024-10-23	80686	ремонт пода...	7398
1850	9112	2	2024-10-07	2024-10-24	67...	NULL	1	8568	2488944	NULL	немає б...	не подає п...	15	2024-10-23	80686	ремонт пода...	6323

Рис. 6.17. Скріншот двох вдалих UPDATE запитів

Таким чином, підтверджено, що уявлення типу MERGE дозволяють редагувати (а отже, додавати, та видаляти) дані, но тільки для однієї таблиці в одному запиті.

**6.3.2.** код SQL-запиту CREATE VIEW, який створює уявлення VIEW 2 на основі базового уявлення VIEW 1 і використовує алгоритм MERGE(рис. 6.18):

```
CREATE OR REPLACE ALGORITHM = MERGE VIEW view_orders_goods AS
SELECT order_id, customer_id, delivery_id, date_invoice, date_delivery,
       delivery_number, date_returned, brand_id, goods_id, quantity, price_out,
       is_shipped, last_name, first_name, phone, address
  FROM view_orders;
```

```
29 16:01:24 CREATE OR REPLACE ALGORITHM = MERGE VIEW view_orders_goods AS SELECT order_id, customer_id, delivery_id, date_in...
```

Рис. 6.18. Скріншот створення уявлення

**6.3.2.1.** Замовлення «без ремонту» за 30.09.2024 (рис. 6.19):

Щоб була можливість шукати замовлення «без ремонту» я трохи виправив код базового уявлення, змінивши JOIN на LEFT JOIN для трьох таблиць:

```
LEFT JOIN orders_received USING(order_id)
LEFT JOIN orders_repair USING(order_id)
LEFT JOIN orders_goods USING(order_id)
```

Тоді потрібний запит буде таким:

```
SELECT * FROM view_orders_goods
WHERE brand_id IS NULL and date_returned = '2024-09-30';
```

```
19 •  SELECT * FROM view_orders_goods WHERE brand_id IS NULL and date_returned = '2024-09-30';
```

Result Grid															Filter Rows:	Export:	Wrap Cell Content:	□
order_id	customer_id	delivery_id	date_invoice	date_delivery	delivery_number	date_returned	brand_id	goods_id	quantity	price_out	is_s	last_name	first_name	phone				
1666	5032	NULL	2024-09-29	NULL	NULL	2024-09-30	NULL	1931	0.824	8132	1	Арсеніч	Анжела	762-79-77				
5440	3779	NULL	2024-09-11	NULL	NULL	2024-09-30	NULL	5202	1.984	7313	1	Адамчук	Лариса	454-65-14				
5440	3779	NULL	2024-09-11	NULL	NULL	2024-09-30	NULL	6241	1.024	2037	1	Адамчук	Лариса	454-65-14				
5440	3779	NULL	2024-09-11	NULL	NULL	2024-09-30	NULL	8005	1.537	10815	1	Адамчук	Лариса	454-65-14				
6346	1935	NULL	2024-09-16	NULL	NULL	2024-09-30	NULL	5550	0.700	5711	1	Баран	Борислав	+380 (25) 121-31				
6346	1935	NULL	2024-09-16	NULL	NULL	2024-09-30	NULL	7556	0.318	8193	1	Баран	Борислав	+380 (25) 121-31				
6976	6897	NULL	2024-09-12	NULL	NULL	2024-09-30	NULL	2180	1.914	12219	1	Гунько	Тетяна	768 35 41				
6976	6897	NULL	2024-09-12	NULL	NULL	2024-09-30	NULL	2363	0.611	13336	1	Гунько	Тетяна	768 35 41				
6976	6897	NULL	2024-09-12	NULL	NULL	2024-09-30	NULL	3549	1.221	9684	1	Гунько	Тетяна	768 35 41				

Рис. 6.19. Скріншот запиту продажу запчастин за 30.09.2024

### 6.3.2.2. Замовлення без ремонту з сумою запчастин (рис. 6.20):

```
SELECT order_id, customer_id, date_invoice, date_returned,
       ROUND(SUM(quantity * price_out)) as total, last_name, first_name
  FROM view_orders_goods
 WHERE brand_id IS NULL and is_shipped = 1
 GROUP BY order_id;
```

```
20 •  SELECT order_id, customer_id, date_invoice, date_returned,
21      ROUND(SUM(quantity * price_out)) as total, last_name, first_name
22  FROM view_orders_goods
23 WHERE brand_id IS NULL and is_shipped = 1
24 GROUP BY order_id;
```

Result Grid							Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
order_id	customer_id	date_invoice	date_returned	total	last_name	first_name				
4	771	2024-05-25	2024-06-19	29370	Джунь	Тереза				
10	2887	2024-09-17	2024-10-15	3114	Гайдамака	Марта				
16	3810	2024-08-16	2024-08-23	11054	Деркач	Ерика				
22	7400	2024-01-20	2024-01-30	1821	Сацок	Богуслава				
28	6287	2024-08-26	2024-09-23	4735	Оніщук	Орина				
34	3258	2024-08-09	2024-08-14	15703	Савенко	Климент				

Рис. 6.20. Скріншот запиту продажу запчастин з сумою запчастин

### 6.3.2.3. Знайти замовлення клієнта по прізвищу «Джунь» (рис. 6.21):

```
SELECT order_id, customer_id, date_invoice, date_returned,
       goods_id, quantity, price_out, is_shipped, last_name, first_name
  FROM view_orders_goods
 WHERE brand_id IS NULL and last_name = 'Джунь';
```

```

20 •   SELECT order_id, customer_id, date_invoice, date_returned,
21       goods_id, quantity, price_out, is_shipped, last_name, first_name
22   FROM view_orders_goods
23   WHERE brand_id IS NULL and last_name = 'Джунь';

```

Result Grid										
	order_id	customer_id	date_invoice	date_returned	goods_id	quantity	price_out	is_s	last_name	first_name
•	2518	44	2024-07-23	2024-07-24	3545	0.170	9694	1	Джунь	Анастасія
	8056	8696	2024-09-02	2024-09-07	3853	1.026	7613	1	Джунь	Ганна
	9532	8229	2024-06-06	2024-06-30	3729	0.196	10143	1	Джунь	Дан
	9532	8229	2024-06-06	2024-06-30	3962	0.405	2965	1	Джунь	Дан
	4	771	2024-05-25	2024-06-19	33	0.863	9465	1	Джунь	Тереза
	4	771	2024-05-25	2024-06-19	2007	1.569	11271	1	Джунь	Тереза
	4	771	2024-05-25	2024-06-19	6693	0.256	13739	1	Джунь	Тереза

Рис. 6.21. Скріншот запиту пошуку замовлень клієнта по прізвищу «Джунь»

6.3.2.4. Знайти списання запчастин по id замовлення (рис. 6.22):

```

SELECT order_id, date_invoice, date_returned, goods_id,
       quantity, price_out, is_shipped, last_name, first_name
FROM view_orders_goods
WHERE brand_id IS NULL and order_id = 4;

```

```

20 •   SELECT order_id, date_invoice, date_returned, goods_id,
21       quantity, price_out, is_shipped, last_name, first_name
22   FROM view_orders_goods
23   WHERE brand_id IS NULL and order_id = 4;

```

Result Grid									
	order_id	date_invoice	date_returned	goods_id	quantity	price_out	is_s	last_name	first_name
•	4	2024-05-25	2024-06-19	33	0.863	9465	1	Джунь	Тереза
	4	2024-05-25	2024-06-19	2007	1.569	11271	1	Джунь	Тереза
	4	2024-05-25	2024-06-19	6693	0.256	13739	1	Джунь	Тереза

Рис. 6.22. Скріншот запиту пошуку замовлень по id

6.3.2.5. Знайти замовлення від Михалюк Альберта та виправити на Михалюк Аліна, надати знижку на запчастини 10% та поставити «відвантажено» сьогодня (рис. 6.23 – 6.24):

1. Пошук замовлення від клієнтов з прізвищем «Михалюк»:

```

SELECT order_id, customer_id, date_invoice, date_returned,
       goods_id, quantity, price_out, is_shipped, last_name, first_name
FROM view_orders_goods
WHERE brand_id IS NULL and last_name = 'Михалюк';

```

order_id	customer_id	date_invoice	date_returned	goods_id	quantity	price_out	is_s	last_name	first_name
3676	9778	2024-03-23	2024-04-01	713	0.295	9141	1	Михалюк	Аліна
3676	9778	2024-03-23	2024-04-01	4015	1.738	14661	1	Михалюк	Аліна
3676	9778	2024-03-23	2024-04-01	5471	0.930	4761	1	Михалюк	Аліна
48	3478	2024-10-21	NULL	9912	1.147	6809	0	Михалюк	Альберт
4324	6706	2024-01-14	2024-02-12	4930	1.742	11649	1	Михалюк	Аркадій
4324	6706	2024-01-14	2024-02-12	9124	0.526	2540	1	Михалюк	Аркадій
220	4464	2024-08-02	2024-08-25	2007	0.529	10025	1	Михалюк	Онисим
220	4464	2024-08-02	2024-08-25	8784	0.268	11627	1	Михалюк	Онисим
9994	7996	2024-07-16	2024-08-01	366	0.279	12725	1	Михалюк	Руслан
9994	7996	2024-07-16	2024-08-01	4906	1.554	9803	1	Михалюк	Руслан
9994	7996	2024-07-16	2024-08-01	5963	1.889	4183	1	Михалюк	Руслан
7804	5182	2024-06-04	2024-06-07	1088	0.625	14664	1	Михалюк	Семен

Рис. 6.23. Скріншот запиту пошуку замовлень від клієнтов з прізвищем «Михалюк»

## 2. Надаємо знижку 10%

```
UPDATE view_orders_goods
SET price_out = price_out * 0.9
WHERE order_id = 48 and goods_id = 9912;
```

✖ 46 17:12:40 UPDATE view\_orders\_goods SET price\_out = price\_out \* 0.9 WHERE order\_id = 48 and goo... Error Code: 1288. The target table view\_orders\_goods of the UPDATE is not updatable

Рис. 6.24. Скріншот запиту, що оновлює цену зачастини

Отримали несподіваний результат. Проаналізуємо, де ми помилилися. Використання в базовому уявленні LEFT JOIN було помилковим. Щоб зберігти можливість редагування треба повернути INNER JOIN. Зробимо це(рис.6.25).

✓ 47 18:17:03 CREATE OR REPLACE ALGORITHM = MERGE VIEW view\_orders AS SELECT order\_id, customer\_id, delivery\_id, date\_invoice, date...

Рис. 6.25. Скріншот запиту, що повертає використання INNER JOIN

Але тепер замовлення з order\_id = 48 нам не доступне.

48 | 18:19:05 | SELECT \* FROM view\_orders WHERE order\_id = 48 | 0 row(s) returned

Рис. 6.26. Скріншот запиту, що шукає замовлення с id=48

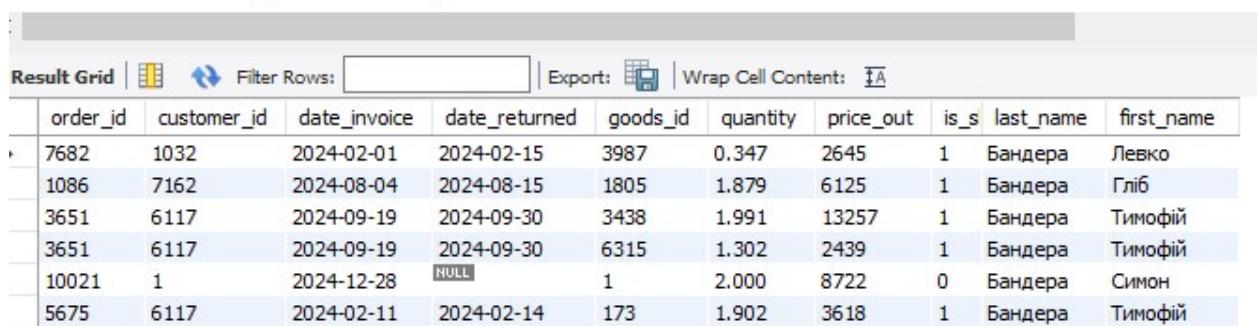
Таке завдання, в пределах цього уявлення зробити не можливо. Тим не менш отримали цікавий досвід.

Переформулюємо це завдання для іншого замовлення.

Знайти замовлення від Бандера Симон та виправити на Бандера Левко, надати знижку на запчастини 20% та поставити «відвантажено» сьогодні (рис. 6.27 – 6.29):

```
SELECT order_id, customer_id, date_invoice, date_returned,
       goods_id, quantity, price_out, is_shipped, last_name, first_name
  FROM view_orders_goods
 WHERE last_name = 'Бандера';
```

```
21 •  SELECT order_id, customer_id, date_invoice, date_returned,
22       goods_id, quantity, price_out, is_shipped, last_name, first_name
23   FROM view_orders_goods
24 WHERE last_name = 'Бандера';
```



The screenshot shows a MySQL query result grid. The query is as follows:

```
SELECT order_id, customer_id, date_invoice, date_returned,
       goods_id, quantity, price_out, is_shipped, last_name, first_name
  FROM view_orders_goods
 WHERE last_name = 'Бандера';
```

The result grid has the following columns: order\_id, customer\_id, date\_invoice, date\_returned, goods\_id, quantity, price\_out, is\_shipped, last\_name, and first\_name. The data is as follows:

order_id	customer_id	date_invoice	date_returned	goods_id	quantity	price_out	is_shipped	last_name	first_name
7682	1032	2024-02-01	2024-02-15	3987	0.347	2645	1	Бандера	Левко
1086	7162	2024-08-04	2024-08-15	1805	1.879	6125	1	Бандера	Гліб
3651	6117	2024-09-19	2024-09-30	3438	1.991	13257	1	Бандера	Тимофій
3651	6117	2024-09-19	2024-09-30	6315	1.302	2439	1	Бандера	Тимофій
10021	1	2024-12-28	NULL	1	2.000	8722	0	Бандера	Симон
5675	6117	2024-02-11	2024-02-14	173	1.902	3618	1	Бандера	Тимофій

Рис. 6.27. Скріншот запиту пошуку замовлення від Бандера

Надаємо знижку 20%

```
UPDATE view_orders_goods
SET price_out = price_out * 0.8
WHERE order_id = 10021 and goods_id = 1;
```

```
61 18:50:08 UPDATE view_orders_goods SET price_out = price_out * 0.9 WHERE order_id = 10021 and goods_id = 1      1 row(s) affected Rows matched: 1 Changed: 1
```

Рис. 6.28. Скріншот запиту, що надає знижку

Виправляємо Бандера Симон на Бандера Левко та встановлюємо дату на сьогодні:

```
UPDATE view_orders_goods
SET customer_id = 1032, date_returned = CURDATE()
WHERE order_id = 10021;
```

```
63 18:54:28 UPDATE view_orders_goods SET customer_id = 1032, date_returned = CURDATE() WHERE order_id = 10... 1 row(s) affected Rows matched: 1 Changed: 1
```

```

21 •   SELECT order_id, customer_id, date_invoice, date_returned,
22       goods_id, quantity, price_out, is_shipped, last_name, first_name
23   FROM view_orders_goods
24   WHERE order_id = 10021;

```

order_id	customer_id	date_invoice	date_returned	goods_id	quantity	price_out	is_shipped	last_name	first_name
10021	1032	2024-12-28	2024-12-28	1	2.000	7065	0	Бандера	Левко

Рис. 6.29. Скріншот результату запиту зміни клієнта, та встановлення дати.

Таким чином, виконавши кілька запитів, підтвердили можливість редагування даних в уявленнях типу MERGE при строгому виконуванні вимог створення таких запитів. Можливість редагування зберігається і для уявлень, що створені на основі базового уявлення.

**6.3.3.** Давайте спробуємо виконати вставку даних в уявлення з алгоритмом MERGE (рис.6.30-6.33). Таблиці мають відповідати таким вимогам:

- Допускається операція INSERT, якщо всі поля базової таблиці, не зазначені у визначенні SQL\_query VIEW, мають значення за замовчуванням (DEFAULT).

**Для цього експерименту я створив дві окремі таблиці:**

```

CREATE TABLE orders (
    id INT AUTO_INCREMENT,
    customer_id INT,
    total_amount INT DEFAULT 0,
    order_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (id));

```

```

CREATE TABLE orders_items (
    order_id INT,
    item_id INT,
    quantity INT DEFAULT 1,
    price INT DEFAULT 0,
    PRIMARY KEY (order_id, item_id));

```

**Далі створив уявлення:**

```

CREATE OR REPLACE ALGORITHM = MERGE VIEW view_orders AS
SELECT *
FROM orders o
JOIN orders_items oi ON o.id = oi.order_id;

```

```

65 19:40:33 CREATE TABLE orders ( order_id INT AUTO_INCREMENT, customer_id ... 0 row(s) affected
66 19:40:33 CREATE TABLE orders_items ( order_id INT, item_id INT DEFAULT 1, ... 0 row(s) affected
67 19:40:33 CREATE OR REPLACE ALGORITHM = MERGE VIEW view_orders AS SELE... 0 row(s) affected

```

Рис. 6.30. Скріншот результату запитів створення таблиць і уявлення

**Вставляємо дані:**

```

INSERT INTO view_orders(customer_id)
VALUES (2);

```

id	customer_id	total_amount	order_date
1	2	0	2024-12-28 19:43:16

order_id	item_id	quantity	price
1	NULL	NULL	NULL

Рис. 6.31. Скріншот результату запита вставки даних

**Як і при оновленні, дані спершу вставляються в одну таблицю, а в іншу вставляємо другим запитом.** Приклад, рис. 6.32

```

INSERT INTO view_orders(id, customer_id, order_id, item_id) VALUES (2,2,2,2);

```

```

92 22:59:36 INSERT INTO view_orders(id, customer_id, order_id, item_id) VALUES (2,2,2,2) Error Code: 1393. Can not modify more than one base table through a join view 'bonnydb.view_orders'

```

Рис. 6.32. Приклад невдалого додавання даних до уявлення

**Також важливо, щоб імена стовбців не повторювались.**

```

INSERT INTO view_orders(order_id, item_id)
VALUES (1,1);

```

order_id	item_id	quantity	price
1	1	1	0

id	customer_id	total_amount	order_date	order_id	item_id	quantity	price
1	2	0	2024-12-28 19:43:16	1	1	1	0

Рис. 6.33. Скріншот результату другого запиту додавання даних до уявлення

**УРА! Усе вийшло. Дійсно, можливо вставляти дані в уявлення по черзі в кожну таблицю. Довели це!**

**6.3.4.** код SQL-запиту CREATE VIEW, який створює базове уявлення VIEW 3 і використовує алгоритм TEMPTABLE;

```
CREATE OR REPLACE ALGORITHM = TEMPTABLE VIEW view_goods AS
SELECT goods_dscr_id, gr.name as group_name, gd.name, part_number,
       ROUND(AVG(price_in))as price_avg, price,
       SUM(stock) as in_stock,
       SUM(reserve) as on_reserve,
       u.name as unit, description
  FROM goods_dscr gd
 JOIN goods USING(goods_dscr_id)
 JOIN `group` gr USING(group_id)
 JOIN unit u USING(Unit_id)
 GROUP BY goods_dscr_id;
```

8 17:56:53 CREATE OR REPLACE ALGORITHM = TEMPTABLE VIEW view\_goods AS SELECT goods\_dscr\_id, gr.name as group\_name, gd.name, part\_number, ROUND(...

Рис. 6.34. Скріншот створення базового уявлення з алгоритмом TEMPTABLE

#### 6.3.4.1. Пошук запчастин по назві групи (рис. 6.35)

SELECT \* FROM view\_goods WHERE group\_name = 'spares Xerox';

13 • SELECT \* FROM view\_goods WHERE group\_name = 'spares Xerox';

goods_dscr_id	group_name	name	part_number	price_avg	price	in_stock	on_reserve	unit	description
12	spares Xerox	Товар U5KQW	G4FP8EFDCX	5525	7183	8.000	0.000	шт	Адвокат болото заспокоїтися нест...
14	spares Xerox	Товар 7W1J0	WFQ9N1SURD	5634	7325	10.000	0.000	рул	Сміятися о лівий конструкція вико...
25	spares Xerox	Товар O60HF	ZYNE8VTP7J	2522	3279	2.000	0.000	кг	Заспівати будівництво похорон за...
28	spares Xerox	Товар 24Z58	9RTXNK57PH	2470	3212	2.000	0.000	рул	Кільце настать казна-хто природн...
32	spares Xerox	Товар US4NR	3FR33RYYLE	3114	4049	1.000	0.000	пач	Бригада в'язниця вітати пробуват...
33	spares Xerox	Товар 2EM9V	7V89INC5SU	1257	1635	2.000	0.000	л	Від'їзд рідкий зарплата неправда ...

Tabular Explain

id	select_type	table	partiti...	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	PRIMARY	<derived2>		ALL					10006	100.00	
2	DERIVED	goods		ALL	fk_goods_dscr_idx				10006	100.00	Using temporary
2	DERIVED	gd		eq_ref	PRIMARY,fk_unit,fk_group	PRIMARY	4	ss_myisam2.goods.goods...	1	100.00	
2	DERIVED	u		eq_ref	PRIMARY	PRIMARY	1	ss_myisam2.gd.unit_id	1	100.00	
2	DERIVED	gr		eq_ref	PRIMARY	PRIMARY	2	ss_myisam2.gd.group_id	1	100.00	

**Timing (as measured at client side):**  
Execution time: 0:00:0.29700000

**Timing (as measured by the server):**  
Execution time: 0:00:0.30472320  
Table lock wait time: 0:00:0.00000600

**Timing (as measured at client side):**  
Execution time: 0:00:0.31200000

**Timing (as measured by the server):**  
Execution time: 0:00:0.31584380  
Table lock wait time: 0:00:0.00000600

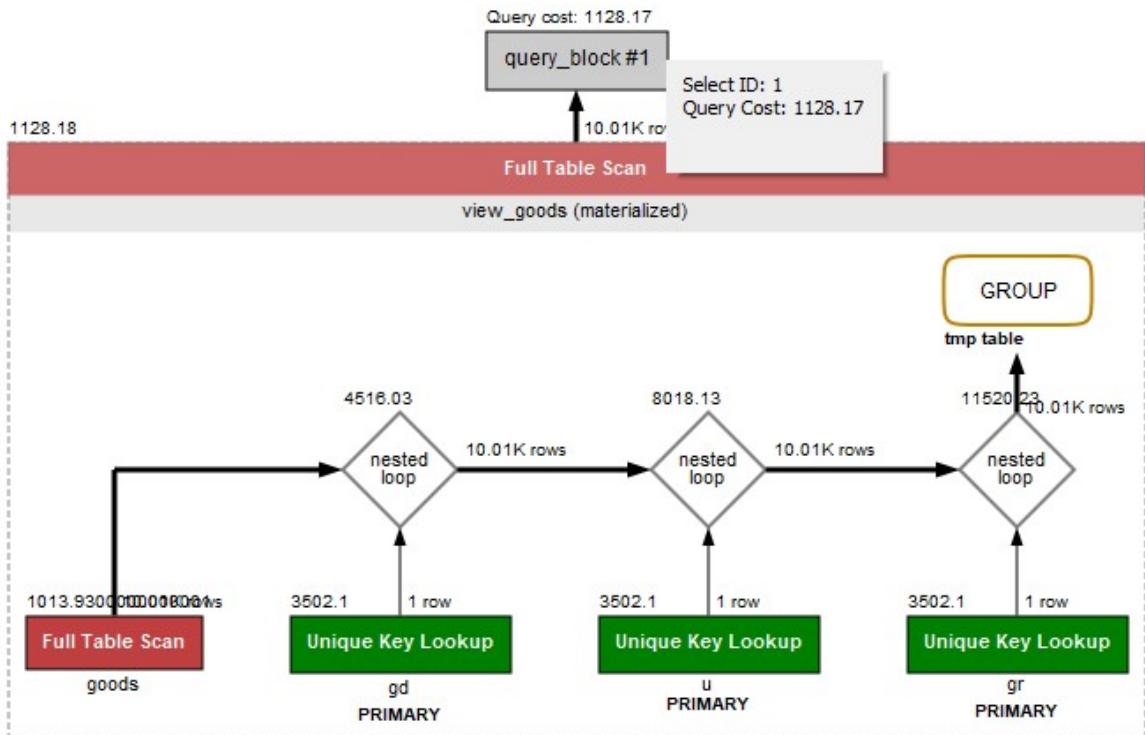


Рис. 6.35. Скріншот запиту пошуку запчастин по назві групи

#### 6.3.4.2. Пошук запчастин по номеру в каталозі (рис. 6.36-6.38) з LIKE

`SELECT * FROM view_goods WHERE part_number LIKE 'W1%';`  
У полі part\_number створен унікальний індекс.

13 • `SELECT * FROM view_goods WHERE part_number LIKE 'W1%';`

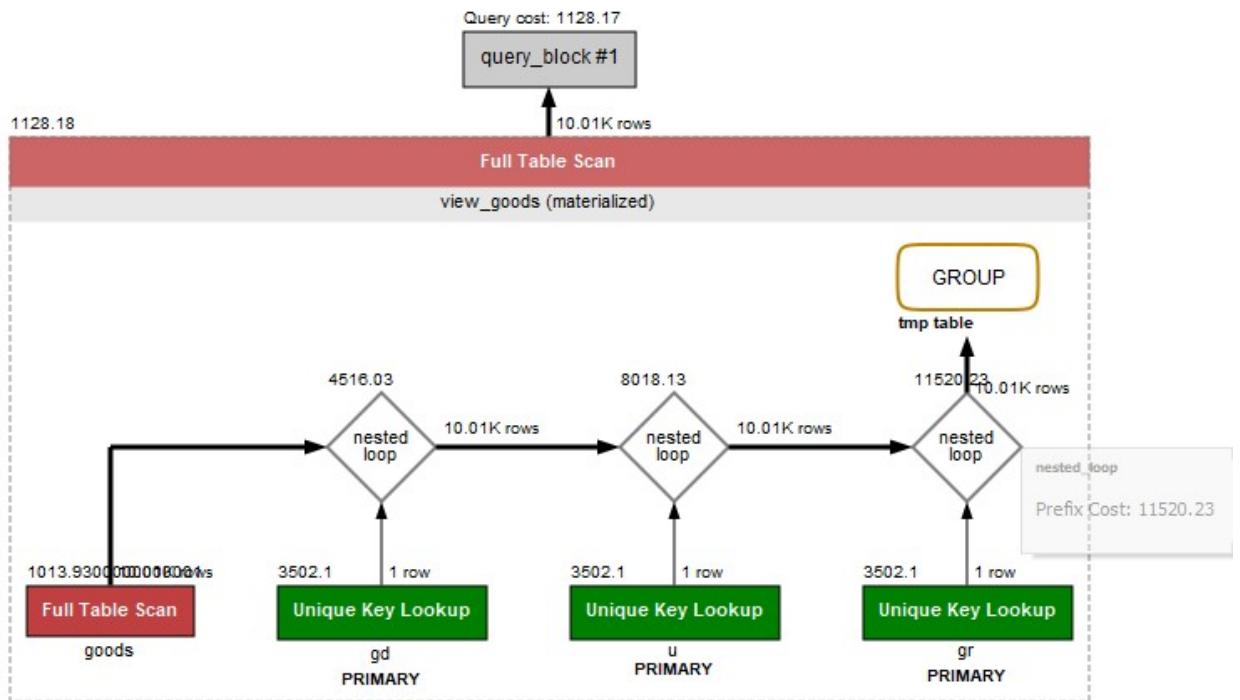
Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

goods_dscr_id	group_name	name	part_number	price_avg	price	in_stock	on_reserve	unit	description
574	supplies HP	Товар EI9HA	W1UEGFBN96	6736	8758	8.000	0.000	пач	Плід міф ремінь Й мит
612	spares other	Товар USRD7	W10394IF7G	4045	5259	8.000	0.000	л	Заклад ламати купа
2804	supplies Ricoh	Товар 1WMXC	W11RXBGBQZ	3428	4457	5.000	0.000	упак	Так головний в'язниц
3409	supplies Canon	Товар N50B1	W19HX56XG0	1191	1549	8.000	0.000	л	Безпрадний услати
6913	supplies Xerox	Товар UW6UO	W1BE6SX8IV	4300	5591	8.000	0.000	пач	Мати полюбити ручк
7004	supplies Canon	Товар BVQAS	W1WEPLPED	7224	9392	3.000	0.887	пач	Вчора каюта плід ад
7795	spares Epson	Товар 19CLS	W1NPRDDCE7	5293	6882	5.000	0.000	пач	Супроводжуватися п
7982	supplies Canon	Товар BJP7P	W19GMYCTG6	7183	9339	1.000	0.000	шт	Вивчити валюта про
8177	supplies Epson	Товар A2BDI	W1R0KIZAED	2348	3053	4.000	0.000	рул	Висловлюватися роз
8987	supplies Epson	Товар GR7JM	W1W7L5LYHQ	4040	5253	10.000	0.000	шт	Командир тривога ро
9734	supplies Ricoh	Товар RG220	W1GH0ITKWM	3979	5173	10.000	0.000	л	Секунда встати скін

17 18:19:16 `SELECT * FROM view_goods WHERE part_number LIKE 'W1%';` 11 row(s) returned

Tabular Explain

id	select_type	table	partitio...	type	possible_keys	key	key_len	ref	rows	filtered	Extra
											10006 100.00
1	PRIMARY	<der...		ALL					10006	100.00	
2	DERIVED	goods		ALL					10006	100.00	Using temporary
2	DERIVED	gd		eq_ref	PRIMARY,fk_unit,fk_group	PRIMARY	4	ss_myisam2.goods.go...	1	100.00	
2	DERIVED	u		eq_ref	PRIMARY	PRIMARY	1	ss_myisam2.gd.unit_id	1	100.00	
2	DERIVED	gr		eq_ref	PRIMARY	PRIMARY	2	ss_myisam2.gd.group...	1	100.00	



**Timing (as measured at client side):**  
Execution time: 0:00:0.31300000

**Timing (as measured at client side):**  
Execution time: 0:00:0.31300000

**Timing (as measured by the server):**  
Execution time: 0:00:0.30318670  
Table lock wait time: 0:00:0.00000700

**Timing (as measured by the server):**  
Execution time: 0:00:0.31871230  
Table lock wait time: 0:00:0.00000700

Рис. 6.36. Скріншот запиту пошуку запчастин по номеру в каталозі з LIKE

`SELECT * FROM view_goods WHERE part_number LIKE '%96';`

13 • `SELECT * FROM view_goods WHERE part_number LIKE '%96';`

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

goods_dscr_id	group_name	name	part_number	price_avg	price	in_stock	on_reserve	unit	description
574	supplies HP	Товар EI9HA	W1UEGFBN96	6736	8758	8.000	0.000	пач	Плід міф ремінь й мі
1695	spares Samsung	Товар WC8VD	P1ZXBI0L96	5916	7692	6.000	0.000	рул	Ланцюжок інфекція
2565	spares Canon	Товар JTBJH	L1Z5VWQR96	982	1277	6.000	0.000	л	Бліскучий процес м
5358	spares Ricoh	Товар VHQTW	M9ILKDMR96	1412	1836	2.000	0.000	пач	Покидати нога пала
5434	spares Samsung	Товар 0EXTX	D78QHUHW96	5018	6524	5.000	0.000	рул	Доба газдиня інтер
8717	spares other	Товар YTB73	89G4SSG496	7315	9510	9.000	0.000	кг	Світило прохід прих

Tabular Explain

id	select_type	table	partitio...	type	possible_keys	key	key_len	ref	rows			filtered	Extra
									10006	100.00	10006		
1	PRIMARY	<derived2>		ALL					10006	100.00	10006	100.00	Using temporary
2	DERIVED	goods		ALL					10006	100.00	10006	100.00	
2	DERIVED	gd		eq_ref	PRIMARY,fk_unit,fk_group	PRIMARY	4	ss_myisam2.goods.go...	1	100.00	1	100.00	
2	DERIVED	u		eq_ref	PRIMARY	PRIMARY	1	ss_myisam2.gd.unit_id	1	100.00	1	100.00	
2	DERIVED	gr		eq_ref	PRIMARY	PRIMARY	2	ss_myisam2.gd.group...	1	100.00	1	100.00	

**Timing (as measured at client side):**  
Execution time: 0:00:0.32800000

**Timing (as measured at client side):**  
Execution time: 0:00:0.32800000

**Timing (as measured by the server):**  
Execution time: 0:00:0.32201240  
Table lock wait time: 0:00:0.00000700

**Timing (as measured by the server):**  
Execution time: 0:00:0.31847870  
Table lock wait time: 0:00:0.00000600

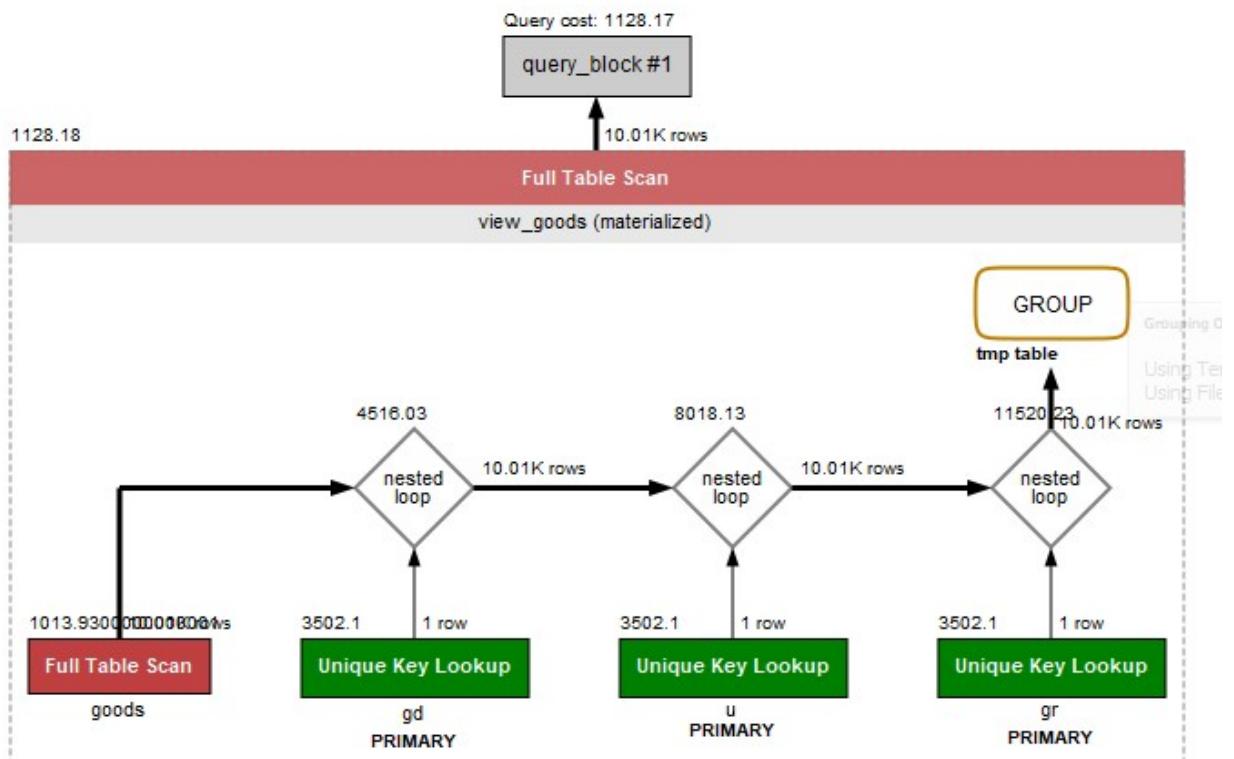


Рис. 6.37. Скріншот запиту пошуку запчастин по номеру в каталозі з LIKE попереду

## Точне співпадіння

```
SELECT * FROM view_goods WHERE part_number = 'D78QHUHW96'
```

```
13 •  SELECT * FROM view_goods WHERE part_number = 'D78QHUhW96';
```

Result Grid										
Filter Rows:		Export:		Wrap Cell Content:						
goods_desc_id	group_name	name	part_number	price_avg	price	in_stock	on_reserve	unit	description	
5434	spares	Samsung	Товар 0FEXTX	D7ZRHUJHW96	5018	6524	5.000	0.000	руп	Доба газдинга інтернет

Tabular Explain											
id	select_type	table	partiti...	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	PRIMARY	<derived2>		ALL					10006	100.00	
2	DERIVED	goods		ALL	fk_goods_dscr_idx				10006	100.00	Using temporary
2	DERIVED	gd		eq_ref	PRIMARY,`fk_unit,`fk_group	PRIMARY	4	ss_myisam2.goods.goods...	1	100.00	
2	DERIVED	u		eq_ref	PRIMARY	PRIMARY	1	ss_myisam2.gd.unit_id	1	100.00	
2	DERIVED	gr		eq_ref	PRIMARY	PRIMARY	2	ss_myisam2.gd.group_id	1	100.00	

**Timing (as measured at client side):**  
Execution time: 0:00:0.31300000

**Timing (as measured at client side):**  
Execution time: 0:00:0.31300000

**Timing (as measured by the server):**  
Execution time: 0:00:0.32349290  
Table lock wait time: 0:00:0.00000700

**Timing (as measured by the server):**  
Execution time: 0:00:0.30747540  
Table lock wait time: 0:00:0.00000600

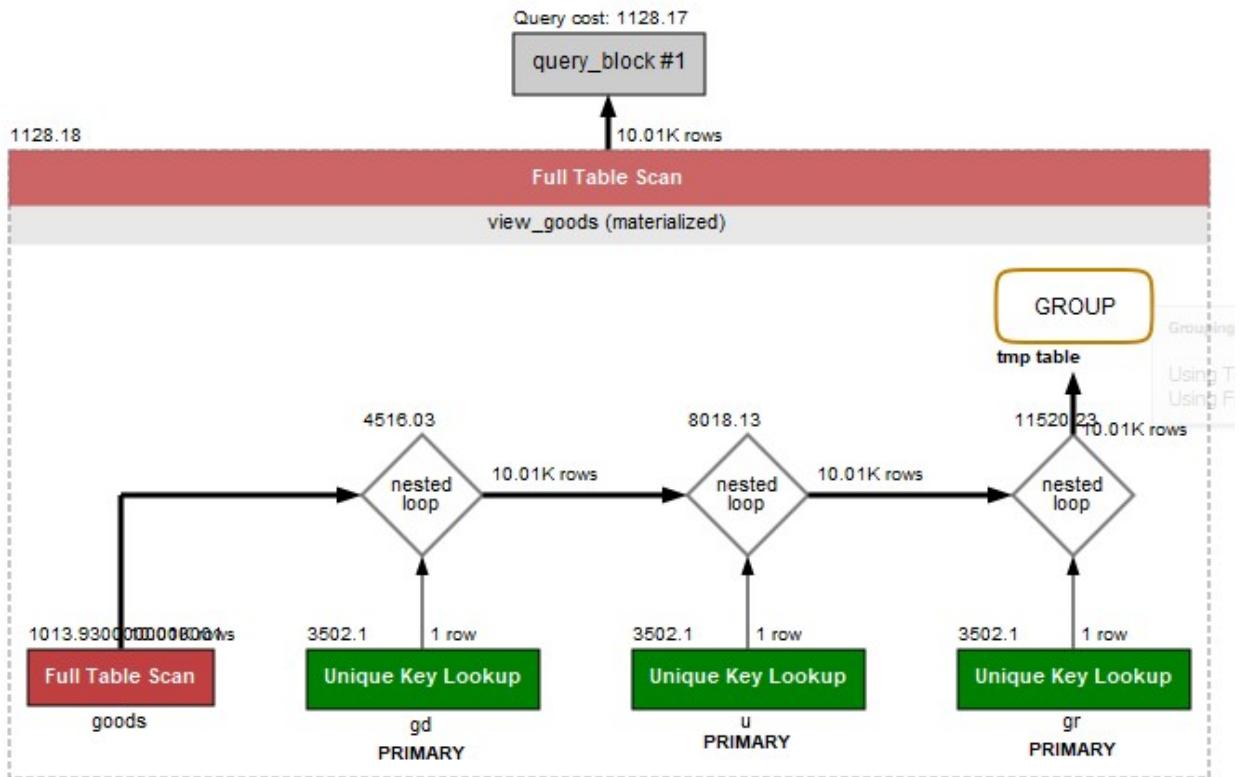


Рис. 6.38. Скріншот запиту пошуку запчастин по номеру в каталозі з LIKE попереду

Результати абсолютно однакові для всіх трьох запитів!

### 6.3.4.3. Пошук запчастин по id (рис. 6.39) – оптимальний результат

`SELECT * FROM view_goods WHERE goods_dscr_id = 1000;`

13 • <code>SELECT * FROM view_goods WHERE goods_dscr_id = 1000;</code>									
result Grid									
Filter Rows: <input type="text"/> Export:  Wrap Cell Content:									
goods_dscr_id	group_name	name	part_number	price_avg	price	in_stock	on_reserve	unit	description
1000	supplies Epson	Товар 3LF6B	LRY2WIBUMF	7150	9295	5.000	0.000	шт	Правління щур щастя знімати плід ...

Tabular Explain											
id	select_type	table	partiti...	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	PRIMARY	<derived2>		system					1	100.00	
2	DERIVED	gd		const	PRIMARY,fk_unit,fk_group	PRIMARY	4	const	1	100.00	
2	DERIVED	gr		const	PRIMARY	PRIMARY	2	const	1	100.00	
2	DERIVED	u		const	PRIMARY	PRIMARY	1	const	1	100.00	
2	DERIVED	goods		ref	fk_goods_dscr_idx	fk_goods_dscr_idx	4	const	1	100.00	

**Timing (as measured at client side):**  
Execution time: 0:00:0.00000000

**Timing (as measured at client side):**  
Execution time: 0:00:0.00000000

**Timing (as measured by the server):**  
Execution time: 0:00:0.00068730  
Table lock wait time: 0:00:0.00000600

**Timing (as measured by the server):**  
Execution time: 0:00:0.00068910  
Table lock wait time: 0:00:0.00000500

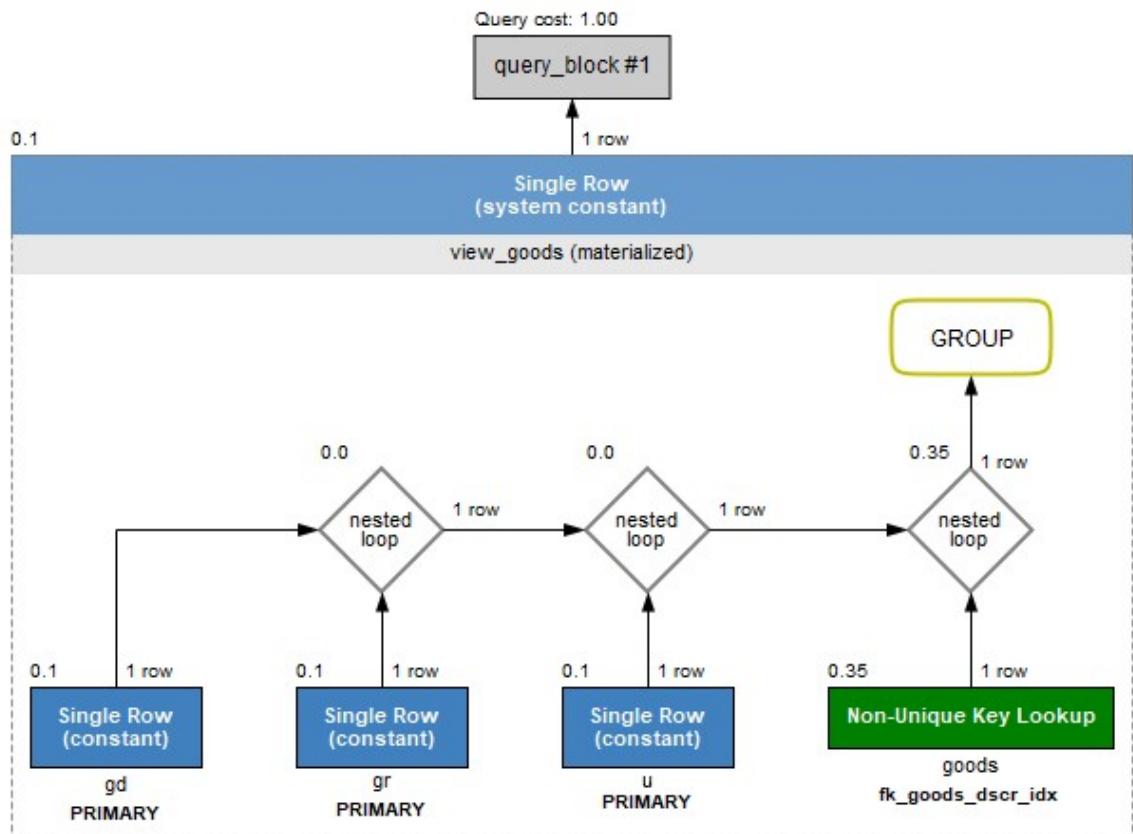


Рис. 6.39. Скріншот запита пошуку запчастин по id

#### 6.3.4.4. FULLTEXT пошук за description (рис. 6.40)

```
SELECT goods_dscr_id, name, description, MATCH(description) AGAINST('спосіб') AS relevance
FROM view_goods WHERE MATCH(description) AGAINST('спосіб');
```

48 19:44:24 SELECT goods\_dscr\_id, name, description, MATCH(description) AGAINST('спосіб') AS relevance FROM view... Error Code: 1214. The used table type doesn't support FULLTEXT indexes

Рис. 6.40. FULLTEXT пошук не підтримується

#### 6.3.4.5. Спробуємо зробити Update (рис. 6.41)

```
UPDATE view_goods SET price = 9500 WHERE goods_dscr_id = 1000;
```

49 19:50:02 UPDATE view\_goods SET price = 9500 WHERE goods\_dscr\_id = 1000 Error Code: 1288. The target table view\_goods of the UPDATE is not updatable

Рис. 6.41. Операція Update не підтримується

#### 6.3.5. код SQL-запиту CREATE VIEW, який створює базове уявлення VIEW 4 на основі базового уявлення VIEW 3 і використовує алгоритм TEMPTABLE;

```
CREATE OR REPLACE ALGORITHM = TEMPTABLE VIEW view_goods_sum AS
SELECT goods_dscr_id, group_name, name,
       price_avg, price,
       in_stock - on_reserve as quality, unit
  FROM view_goods;
```

53 20:06:19 CREATE OR REPLACE ALGORITHM = TEMPTABLE VIEW view\_goods\_sum AS SELECT go... 0 row(s) affected

Рис. 6.42. Створення уялення

**6.3.5.1.** Знайти помилки в базі. Яких запчастин залишилось менше 0 (рис. 6.43) ?

```
SELECT * FROM view_goods_sum WHERE quality < 0;
```

28 • SELECT \* FROM view\_goods\_sum WHERE quality < 0;

goods_dscr_id	group_name	name	price_avg	price	quality	unit
260	supplies Canon	Товар 7W137	1600	2080	-0.642	пач
299	supplies Samsung	Товар 3DE3E	7403	9625	-0.486	рул
976	spares other	Товар TIRZ0	1430	1860	-0.380	рул
1116	spares Ricoh	Товар JPRBF	7269	9450	-0.838	рул
1538	supplies Ricoh	Товар 52X84	3257	4235	-0.890	упак
2111	spares Epson	Товар MZGZO	1124	1462	-0.540	кг
2184	supplies Samsung	Товар 9HGPB	2176	2829	-0.128	кг

54 20:10:16 SELECT \* FROM view\_goods\_sum WHERE quality < 0 44 row(s) returned

Tabular Explain											
id	select_type	table	partiti...	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	PRIMARY	<derived2>		ALL					10006	100.00	
2	DERIVED	<derived3>		ALL					10006	100.00	
3	DERIVED	goods		ALL	fk_goods_dscr_idx				10006	100.00	Using temporary
3	DERIVED	gd		eq_ref	PRIMARY,fk_unit,fk_group	PRIMARY	4	ss_myisam2.goods.goods...	1	100.00	
3	DERIVED	u		eq_ref	PRIMARY	PRIMARY	1	ss_myisam2.gd.unit_id	1	100.00	
3	DERIVED	gr		eq_ref	PRIMARY	PRIMARY	2	ss_myisam2.gd.group_id	1	100.00	

**Timing (as measured at client side):**  
Execution time: 0:00:0.29700000

**Timing (as measured by the server):**  
Execution time: 0:00:0.30435030  
Table lock wait time: 0:00:0.00000600

**Timing (as measured at client side):**  
Execution time: 0:00:0.31200000

**Timing (as measured by the server):**  
Execution time: 0:00:0.31654600  
Table lock wait time: 0:00:0.00000600

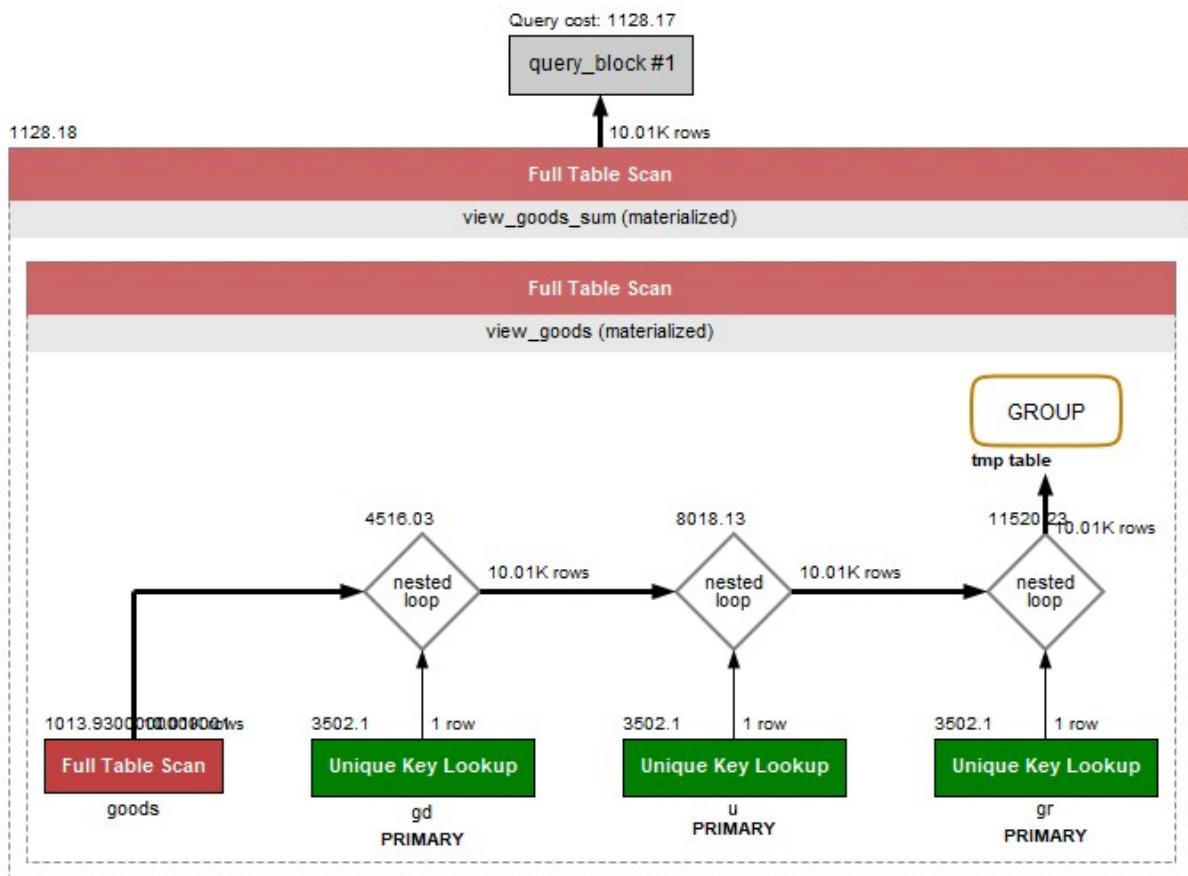


Рис. 6.43. Пошук запчастин, кількість яких менше нуля.

### 6.3.5.2. Знайти запчастини с націнкою менше 30% (рис. 6.44)

```
SELECT goods_dscr_id, group_name, name,
       ROUND((price / price_avg - 1) * 100) as extra, price, quality, unit
  FROM view_goods_sum WHERE (price / price_avg - 1) * 100 < 30;
```

```
29 •  SELECT goods_dscr_id, group_name, name,
30      ROUND((price / price_avg - 1) * 100) as extra, price, quality, unit
31      FROM view_goods_sum WHERE (price / price_avg - 1) * 100 < 30;
```

Result Grid						
	goods_dscr_id	group_name	name	extra	price	quality
•	16	supplies HP	Товар HXYY3	7	4545	5.000
	6264	spares Canon	Товар HVF6P	16	8057	1.000
	9897	supplies Epson	Товар XS7SH	15	7643	8.000
	10000	spares Canon	Товар KRA34	29	5956	0.000
	10004	spares Samsung	Картридж 1210	17	35000	0.000
	10005	spares Samsung	Картридж 2015	10	55000	0.000

Рис. 6.44. Пошук запчастин с націнкою менше 30%

### 6.3.5.3. Підрахувати суми запчастин по групах(рис. 6.45)

```
SELECT group_name,
       ROUND(SUM(price * quality)) as total_group
  FROM view_goods_sum
 GROUP BY group_name;
```

```
33 •  SELECT group_name,
34          ROUND(SUM(price * quality)) as total_group
35      FROM view_goods_sum
36      GROUP BY group_name;
```

Result Grid		
	group_name	ROUND(SUM(price * quality))
▶	spares Ricoh	20141919
	spares Canon	22587206
	spares other	23578624
	supplies Canon	23813232
	spares Epson	23741270
	supplies Ricoh	24442893
	supplies HP	22587073
	supplies Xerox	23638612
	spares Xerox	23584788
	supplies Samsung	22563048
	supplies Epson	23317102
	spares HP	23006506
	spares Samsung	22727480

Tabular Explain											
id	select_type	table	partiti...	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	PRIMARY	<derived2>		ALL					10006	100.00	
2	DERIVED	<derived3>		ALL					10006	100.00	
3	DERIVED	goods		ALL	fk_goods_dscr_idx				10006	100.00	Using temporary
3	DERIVED	gd		eq_ref	PRIMARY,fk_unit,fk_group	PRIMARY	4	ss_myisam2.goods.goods...	1	100.00	
3	DERIVED	u		eq_ref	PRIMARY	PRIMARY	1	ss_myisam2.gd.unit_id	1	100.00	
3	DERIVED	gr		eq_ref	PRIMARY	PRIMARY	2	ss_myisam2.gd.group_id	1	100.00	

#### Timing (as measured at client side):

Execution time: 0:00:0.39100000

#### Timing (as measured at client side):

Execution time: 0:00:0.36000000

#### Timing (as measured by the server):

Execution time: 0:00:0.38059500

Table lock wait time: 0:00:0.00000600

#### Timing (as measured by the server):

Execution time: 0:00:0.36708730

Table lock wait time: 0:00:0.00000600

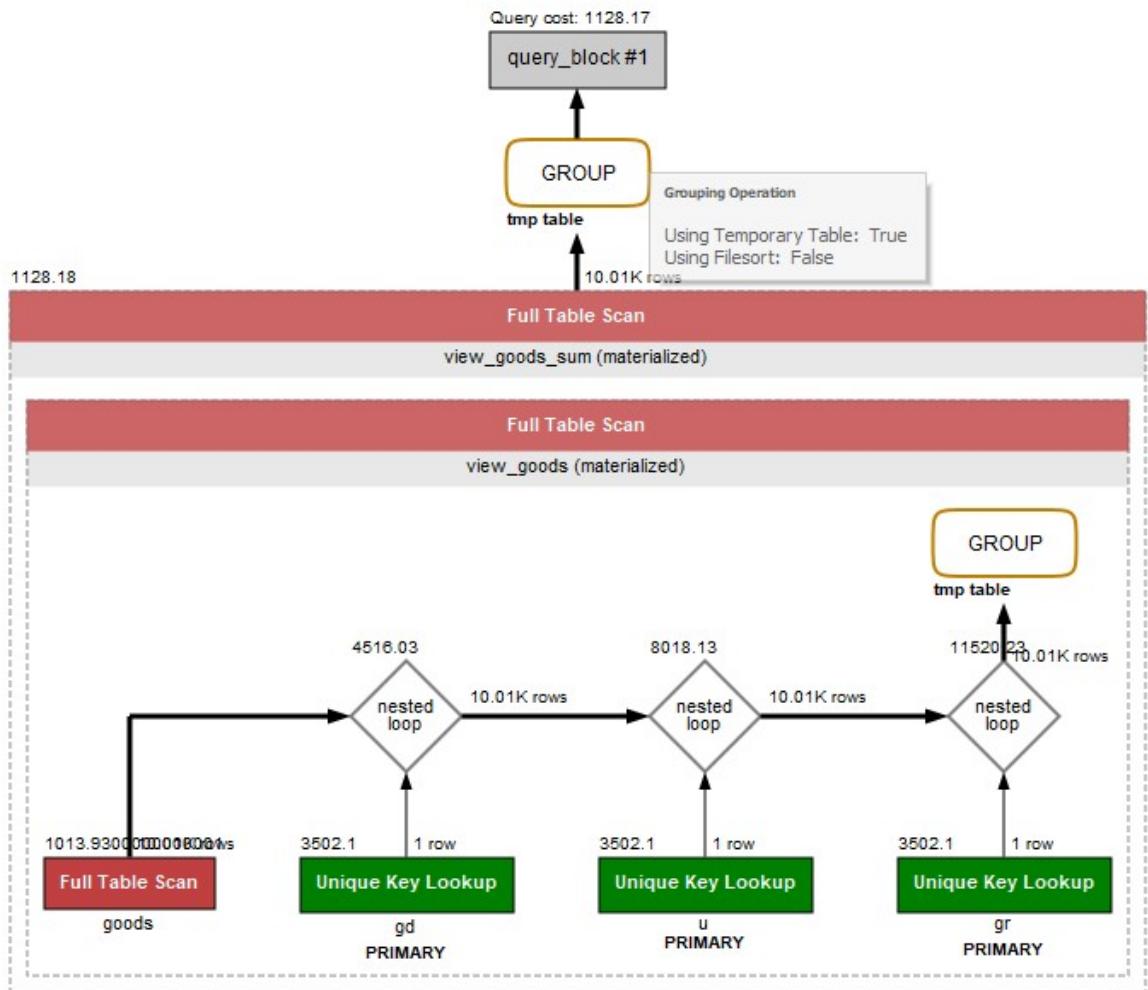


Рис. 6.45. Підрахунок суми запчастин по групах

**6.3.6. Щодо оптимізації, давайте розглянемо запити, що створюють базові уявлення.**

#### 6.3.6.1. Уявлення TEMPORARY, з групуванням

```

SELECT SQL_NO_CACHE goods_dscr_id, gr.name as group_name, gd.name,
part_number, ROUND(AVG(price_in))as price_avg, price,
SUM(stock) as in_stock,
SUM(reserve) as on_reserve,
u.name as unit, description
FROM goods_dscr gd
JOIN goods USING(goods_dscr_id)
JOIN `group` gr USING(group_id)
JOIN unit u USING(Unit_id)
GROUP BY goods_dscr_id;

```

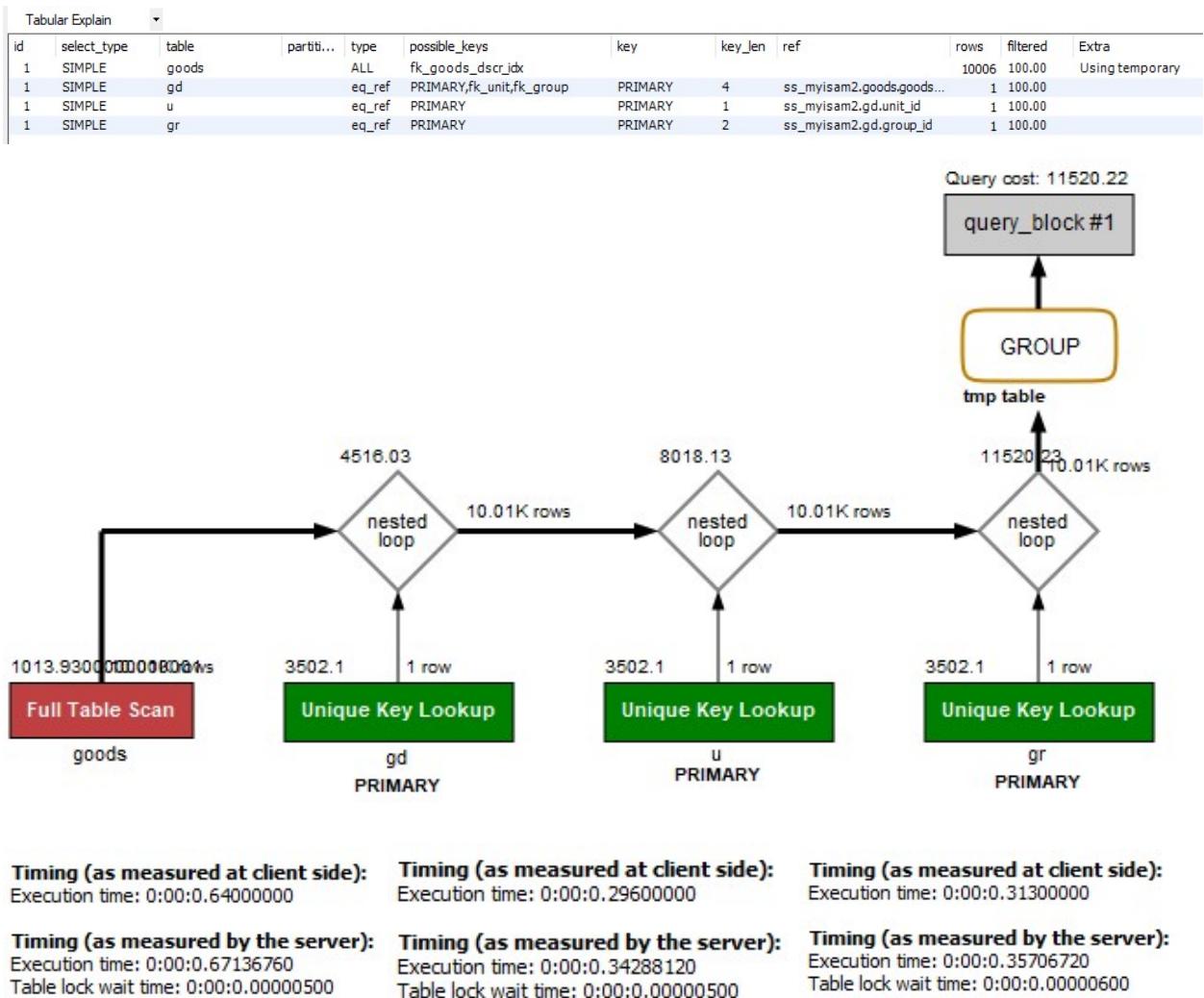


Рис. 6.46. Скріншот EXPLAIN базового уявлення

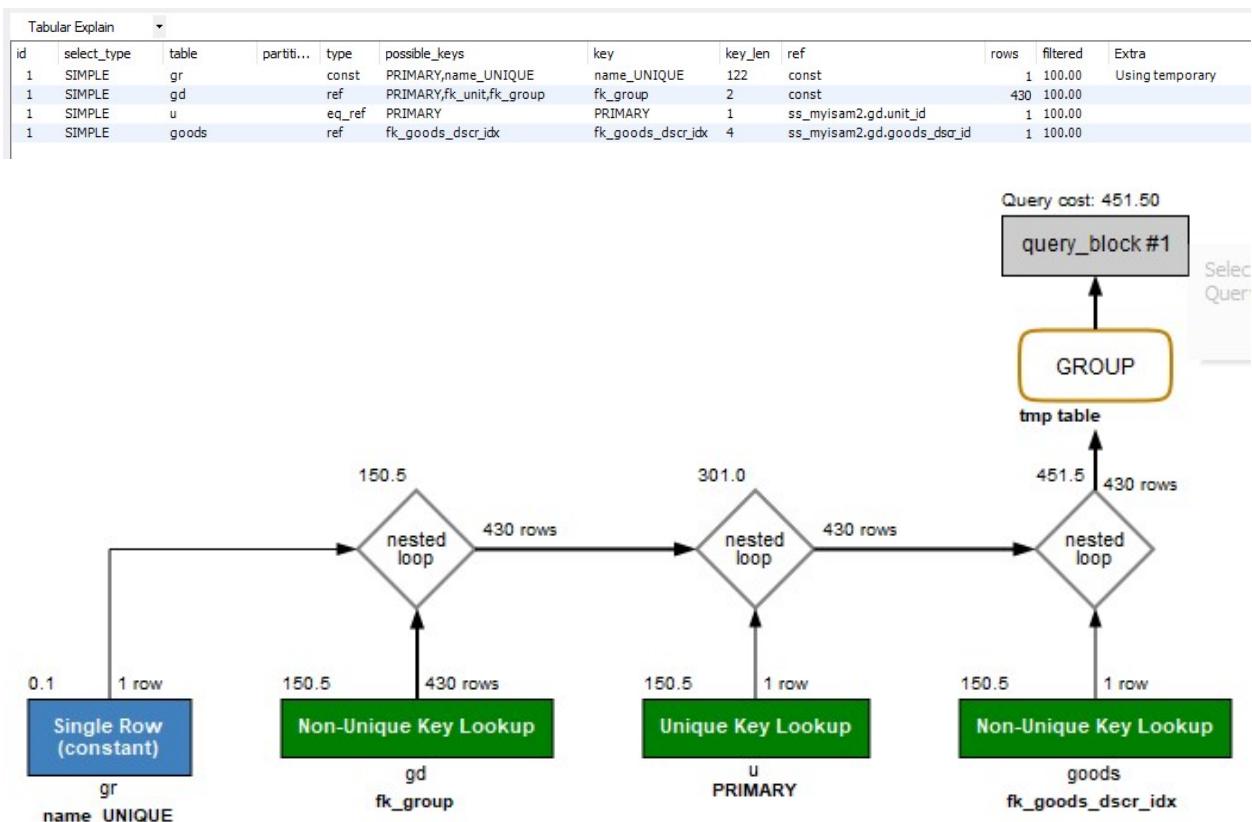
Бачимо, що запит виконується досить повільно тому що створюється тимчасова таблиця для групування по `goods_id`. Індекс `fk_goods_dscr_idx` не використовується. При цьому ніяк не обмежується межі групування. Тому, всі далі створені запити на базі цього уявлення демонструють таку саму швидкість.

Якщо вийде, спершу спробувати обмежити кількість записів, а потім групувати, то можна очікувати збільшення швидкодії запиту.

Наприклад, [пошук запчастин за групою](#), той самий запит, але сперва фільтруєм, потім групуєм:

```
SELECT SQL_NO_CACHE goods_dscr_id, gr.name as group_name, gd.name,
part_number, ROUND(AVG(price_in))as price_avg, price,
SUM(stock) as in_stock,
SUM(reserve) as on_reserve,
u.name as unit, description
FROM goods_dscr gd
JOIN goods USING(goods_dscr_id)
JOIN `group` gr USING(group_id)
JOIN unit u USING(Unit_id)
```

WHERE gr.name = 'spares Xerox'  
GROUP BY goods\_dscr\_id;



**Timing (as measured at client side):**  
Execution time: 0:00:0.03100000

**Timing (as measured by the server):**  
Execution time: 0:00:0.02440410  
Table lock wait time: 0:00:0.00004000

**Timing (as measured at client side):**  
Execution time: 0:00:0.01600000

**Timing (as measured by the server):**  
Execution time: 0:00:0.02269070  
Table lock wait time: 0:00:0.00000600

Рис. 6.47. Скріншот EXPLAIN пошуку за групою

Отримали ідеальний результат просто переставив местами WHERE та ORDER BY. Тут вже індекс fk\_goods\_dscr\_idx використовується! Час виконання зменшився в 11-12 разів! (також дивись стор.17):

Або, пошук запчастин за ім'ям, швидше ніж на стор.20, але всі індекси не використовуються, використання WHERE дає пришвидшення у 5 разів! (рис. 6.48):

**Timing (as measured at client side):**  
Execution time: 0:00:0.06300000

**Timing (as measured at client side):**  
Execution time: 0:00:0.07800000

**Timing (as measured by the server):**  
Execution time: 0:00:0.06477010  
Table lock wait time: 0:00:0.00000600

**Timing (as measured by the server):**  
Execution time: 0:00:0.07092100  
Table lock wait time: 0:00:0.00000600

goods_dscr_id	group_name	name	part_number	price_avg	price	in_stock	on_reserve	unit	description
20	supplies Xerox	Товар CRBOY	Y2YKEJHGXU	4282	5567	3.000	0.000	рул	Мимо нарада благати мати

Tabular Explain											
d	select_type	table	partiti...	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	gd		ALL	PRIMARY, fk_unit, fk_group				10006	10.00	Using where; Using temporary
1	SIMPLE	u		eq_ref	PRIMARY	PRIMARY	1	ss_myisam2.gd.unit_id	1	100.00	
1	SIMPLE	gr		eq_ref	PRIMARY	PRIMARY	2	ss_myisam2.gd.group_id	1	100.00	
1	SIMPLE	goods		ref	fk_goods_dscr_idx	fk_goods_dscr_idx	4	ss_myisam2.gd.goods_dscr_id	1	100.00	

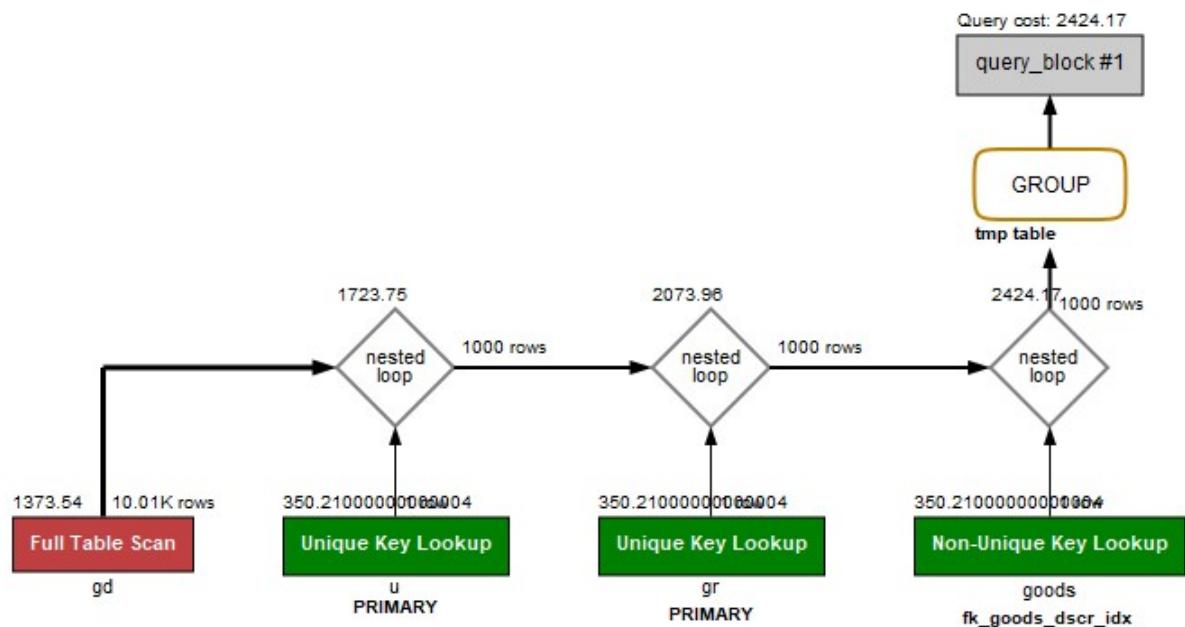


Рис. 6.48. Скріншот EXPLAIN пошуку за ім'ям

### 6.3.6.2. EXPLAIN оптимізація уявлення MERGE

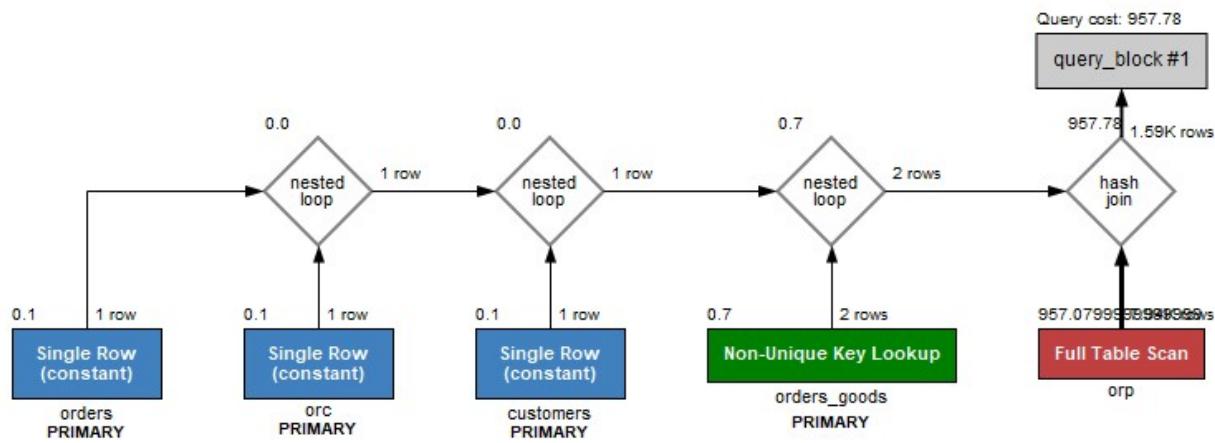
1.

EXPLAIN

SELECT \* FROM view\_orders WHERE order\_id = 5555;

order_id	custom	deliv	date_invoice	date_delivery	delive	date_r	brand_id	model_name	sn	barcode	equipment	fault_descripti	emplc	date_ready	price_rt	description	goods_id
5555	7469	3	2024-02-01	2024-02-06	61...	202...	1	9167	4369552	NULL	Салон ле...	Білій дівка ...	16	2024-02-05	48551	Демократія в...	4081
5555	7469	3	2024-02-01	2024-02-06	61...	202...	1	9167	4369552	NULL	Салон ле...	Білій дівка ...	16	2024-02-05	48551	Демократія в...	1885

Tabular Explain																	
id	select_type	table	partiti...	type	possible_keys	key	key_len	ref	rows	filtered	Extra						
1	SIMPLE	orders		const	PRIMARY, fk_cust	PRIMARY	4	const	1	100.00							
1	SIMPLE	orc		const	PRIMARY	PRIMARY	4	const	1	100.00							
1	SIMPLE	customers		const	PRIMARY	PRIMARY	4	const	1	100.00							
1	SIMPLE	orders_goo...		ref	PRIMARY	PRIMARY	4	const	2	100.00							
1	SIMPLE	orp		ALL					7939	10.00	Using where; Using join buffer (hash join)						



**Timing (as measured at client side):**  
Execution time: 0:00:0.06200000

**Timing (as measured at client side):**  
Execution time: 0:00:0.04600000

**Timing (as measured by the server):**  
Execution time: 0:00:0.05377480  
Table lock wait time: 0:00:0.00000700

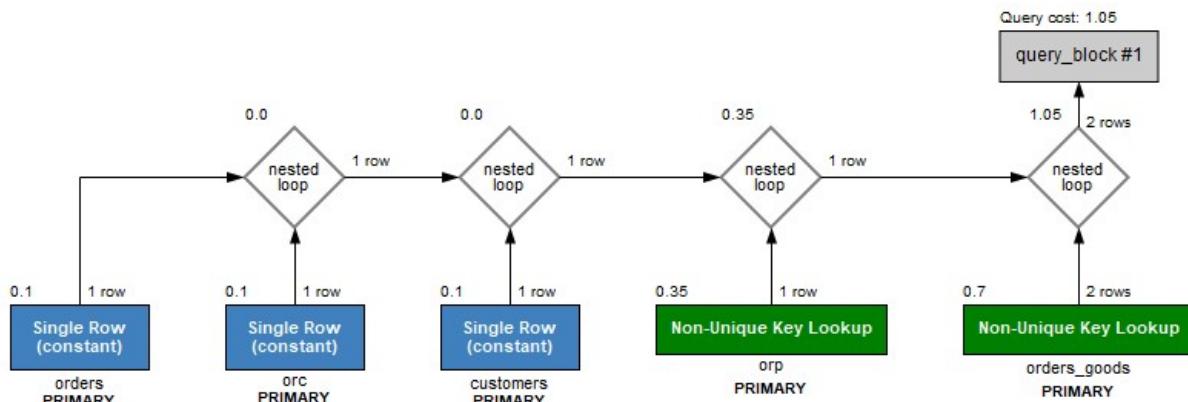
**Timing (as measured by the server):**  
Execution time: 0:00:0.04607640  
Table lock wait time: 0:00:0.00000700

Рис. 6.49. Скріншот EXPLAIN оптимізація уявлення MERGE

Мені здається тут щось працює не зовсім вірно. Виявилося що складний первинний ключ був зроблений не в тому порядку (`employee_id`, `order_id`), видалив, перестворив (`order_id`, `employee_id`) отримав іншу картинку:

```
41 •  SELECT * FROM view_orders WHERE order_id = 5555;
```

Tabular Explain											
id	select_type	table	partiti...	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	orders		const	PRIMARY, fk_cust	PRIMARY	4	const	1	100.00	
1	SIMPLE	orc		const	PRIMARY	PRIMARY	4	const	1	100.00	
1	SIMPLE	customers		const	PRIMARY	PRIMARY	4	const	1	100.00	
1	SIMPLE	orp		ref	PRIMARY	PRIMARY	4	const	1	100.00	
1	SIMPLE	orders_goods		ref	PRIMARY	PRIMARY	4	const	2	100.00	



**Timing (as measured at client side):**  
Execution time: 0:00:0.000000000

**Timing (as measured at client side):**

**Timing (as measured by the server):**  
Execution time: 0:00:0.00087030  
Table lock wait time: 0:00:0.00001000

**Timing (as measured by the server):**  
Execution time: 0:00:0.00121930  
Table lock wait time: 0:00:0.00000600

Рис. 6.50. Скріншот EXPLAIN оптимізація уявлення MERGE (після виправлення індексу)

2.

Дивна поведінка складеного індексу, сперва пошук по last\_name. Індекс виглядає таким чином, рис 6.51:

Table Name: customers Schema: ss\_myisam2

Charset/Collation: utf8mb4 utf8mb4\_0900\_ai\_ci Engine: MyISAM

Comments:

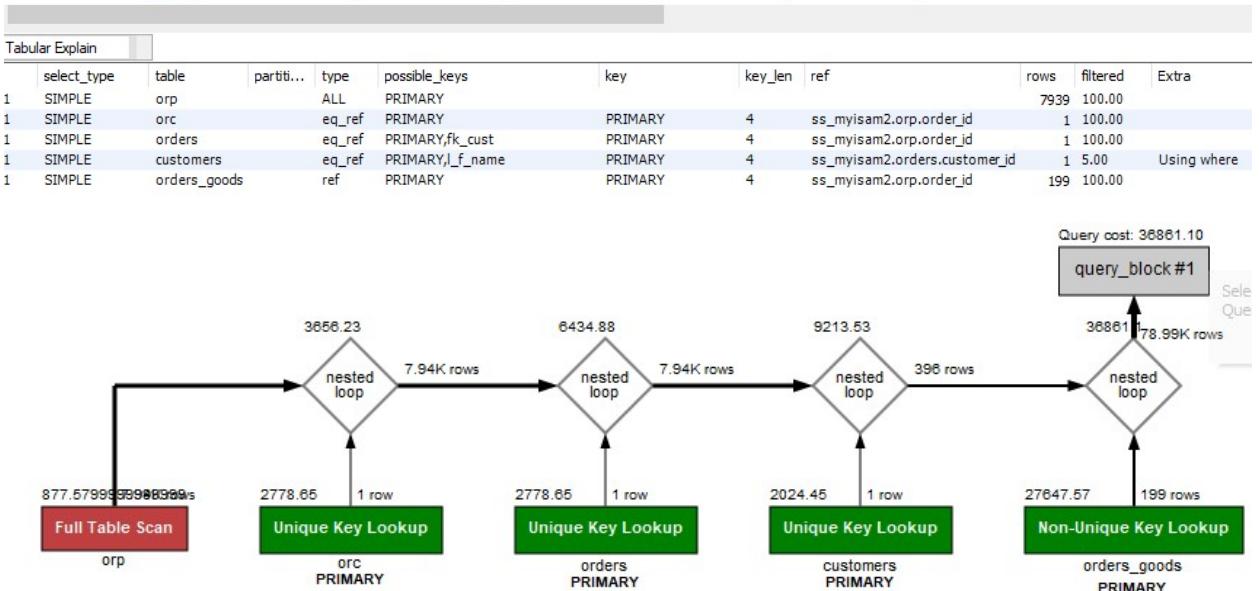
Index Name	Type
PRIMARY	PRIMARY
phone_UNIQUE	UNIQUE
fk_city	INDEX
fk_status	INDEX
<b>l_f_name</b>	<b>INDEX</b>
descr_cust_fulltext	FULLTEXT

Column	#	Order	Length
customer_id		ASC	
city_id		ASC	
status_id		ASC	
<b>last_name</b>	<b>1</b>	ASC	<b>7</b>
<b>first_name</b>	<b>2</b>	ASC	<b>5</b>
middle_name		ASC	

Рис. 6.51 Складений індекс last\_name, first\_name

`SELECT SQL_NO_CACHE * FROM view_orders  
WHERE last_name = 'Михалюк';`

46 • `SELECT SQL_NO_CACHE * FROM view_orders WHERE last_name = 'Михалюк'; -- and first_name = 'Аліна';`



```
46 •  SELECT SQL_NO_CACHE * FROM view_orders WHERE last_name = 'Михалюк'; -- and first_name = 'Аліна';

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

order_id	custom	deliv	date_invoice	date_delivery	delive	date_r	brand_id	model_name	sn	barcode	equipment	fault_descripti
137	5487	3	2024-08-07	2024-08-12	72...	202...	5	1996	3623466	NULL	Гркий д...	Пристрій щ...
456	2460	3	2024-01-23	2024-01-25	39...	202...	4	8069	5042978	NULL	Керівник...	Вперед да...
456	2460	3	2024-01-23	2024-01-25	39...	202...	4	8069	5042978	NULL	Керівник...	Вперед да...
456	2460	3	2024-01-23	2024-01-25	39...	202...	4	8069	5042978	NULL	Керівник...	Вперед да...
536	9384	3	2024-02-28	2024-03-04	92...	202...	1	4488	8971497	NULL	Наступа...	Зрозумлий...
124	20:33:21	SELECT SQL_NO_CACHE * FROM view_orders WHERE last_name = 'Михалюк'										34 row(s) returned

**Timing (as measured at client side):**  
Execution time: 0:00:0.17100000

**Timing (as measured at client side):**  
Execution time: 0:00:0.18800000

**Timing (as measured by the server):**  
Execution time: 0:00:0.29472800  
Table lock wait time: 0:00:0.00000700

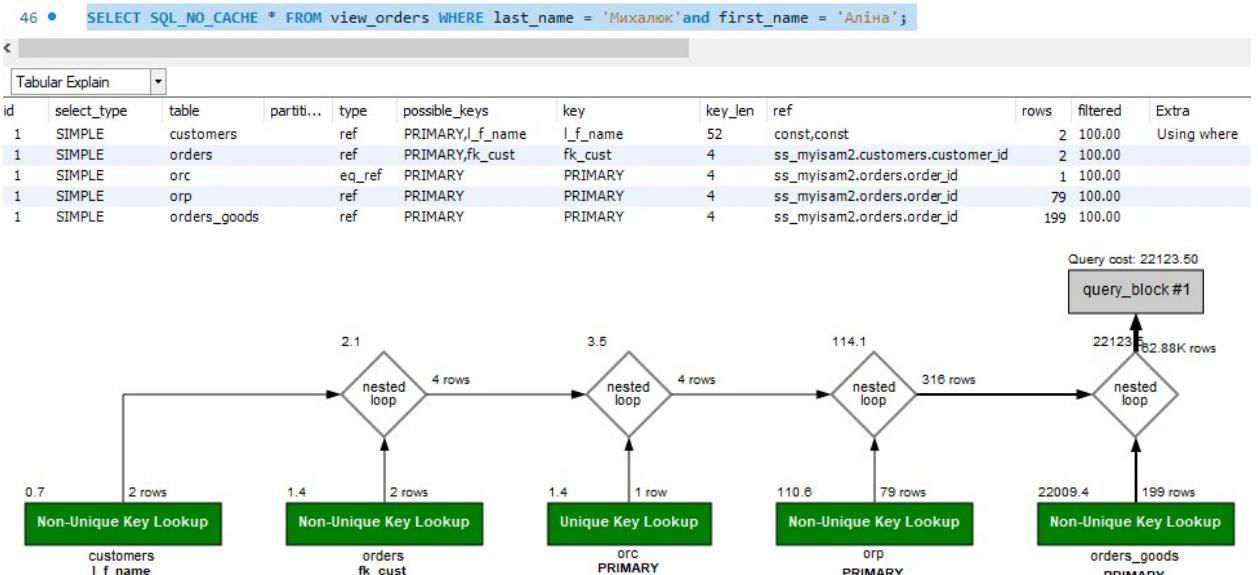
**Timing (as measured by the server):**  
Execution time: 0:00:0.29773610  
Table lock wait time: 0:00:0.00000600

Рис. 6.52 Скріншот EXPLAIN оптимізації уявлення MERGE, що шукає за прізвищем користувача, але індекс складений.

### 3.

А тепер last\_name and first\_name разом:

```
SELECT SQL_NO_CACHE * FROM view_orders
WHERE last_name = 'Михалюк' and first_name = 'Аліна';
```



**Timing (as measured at client side):**  
Execution time: 0:00:0.00000000

**Timing (as measured at client side):**  
Execution time: 0:00:0.00000000

**Timing (as measured by the server):**  
Execution time: 0:00:0.00183860  
Table lock wait time: 0:00:0.00000900

**Timing (as measured by the server):**  
Execution time: 0:00:0.00099690  
Table lock wait time: 0:00:0.00000700

```
46 •     SELECT SQL_NO_CACHE * FROM view_orders WHERE last_name = 'Михалюк' and first_name = 'Аліна';
```

Result Grid														
<input type="text" value="Filter Rows:"/> <input type="button" value="Export:"/> <input type="button" value="Wrap Cell Content:"/>														
order_id	custom	deliv	date_invoice	date_delivery	delive	date_r	brand_id	model_name	sn	barcode	equipment	fault_descripti	emp	
2780	358	3	2024-09-12	2024-10-11	15...	202...	3	1727	4190683	NULL	Вивести ...	Податкови...	4	
2780	358	3	2024-09-12	2024-10-11	15...	202...	3	1727	4190683	NULL	Вивести ...	Податкови...	4	
137	5487	3	2024-08-07	2024-08-12	72...	202...	5	1996	3623466	NULL	Прийм...	Пристрий щ...	4	
1662	9778	2	2024-05-01	2024-05-31	91...	202...	2	7334	4989044	NULL	Порт сек...	Функція боя...	8	
1662	9778	2	2024-05-01	2024-05-31	91...	202...	2	7334	4989044	NULL	Порт сек...	Функція боя...	8	

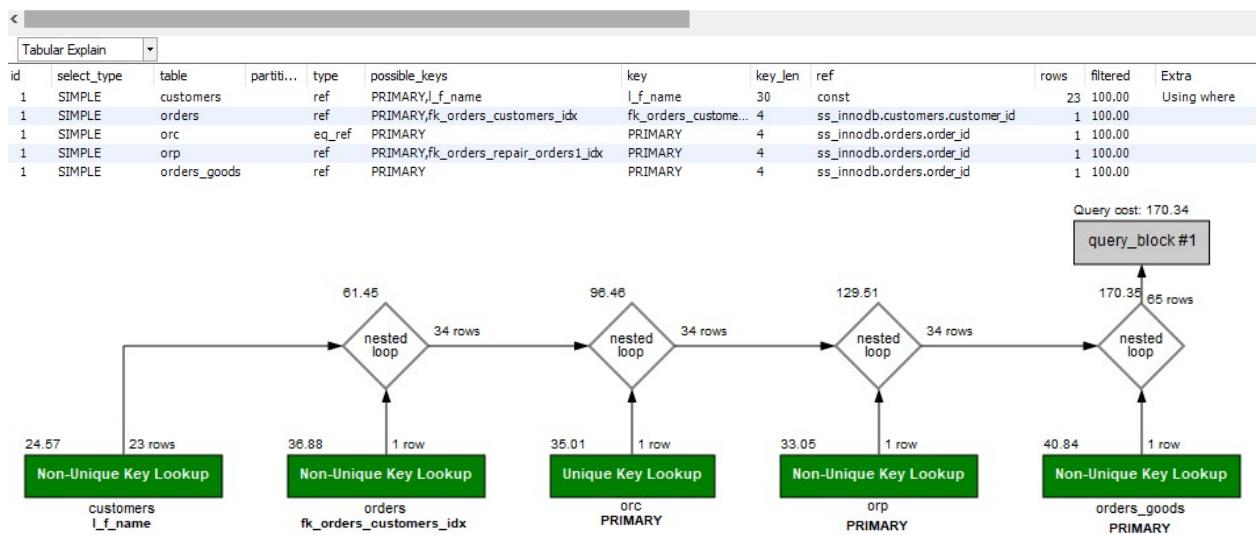
127 20:41:23 SELECT SQL\_NO\_CACHE \* FROM view\_orders WHERE last\_name = 'Михалюк' and first\_name = 'Аліна' 5 row(s) returned

Рис. 6.53 Скріншот EXPLAIN оптимізації уявлення MERGE, що шукає за прізвищем та ім'ям користувача, індекс складений.

Дивна поведінка складеного індексу, як це можна пояснити?

Спробуємо, як себе поведе уявлення на InnoDB

47 • `SELECT SQL_NO_CACHE * FROM view_orders WHERE last_name = 'Михалюк';`



```
46 •     SELECT SQL_NO_CACHE * FROM view_orders WHERE last_name = 'Міхалюк' and first_name = 'Аліна';
```

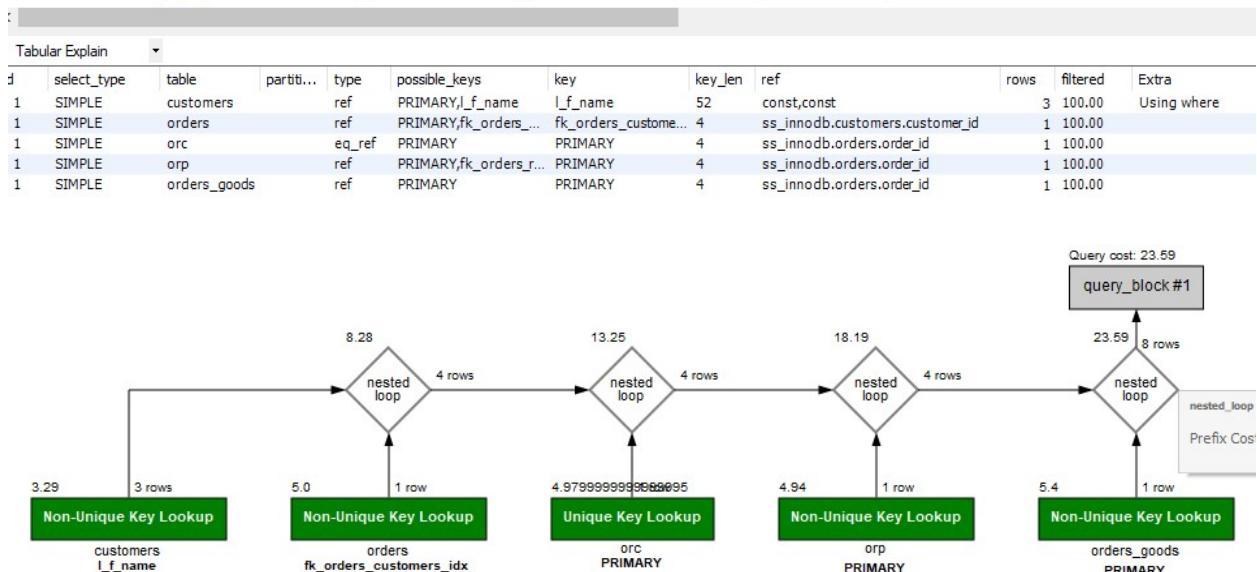


Рис. 6.54 Скріншот EXPLAIN оптимізації уявлення - InnoDB

Знайшов помилку в індексі до таблиці `orders_repair`, виправив

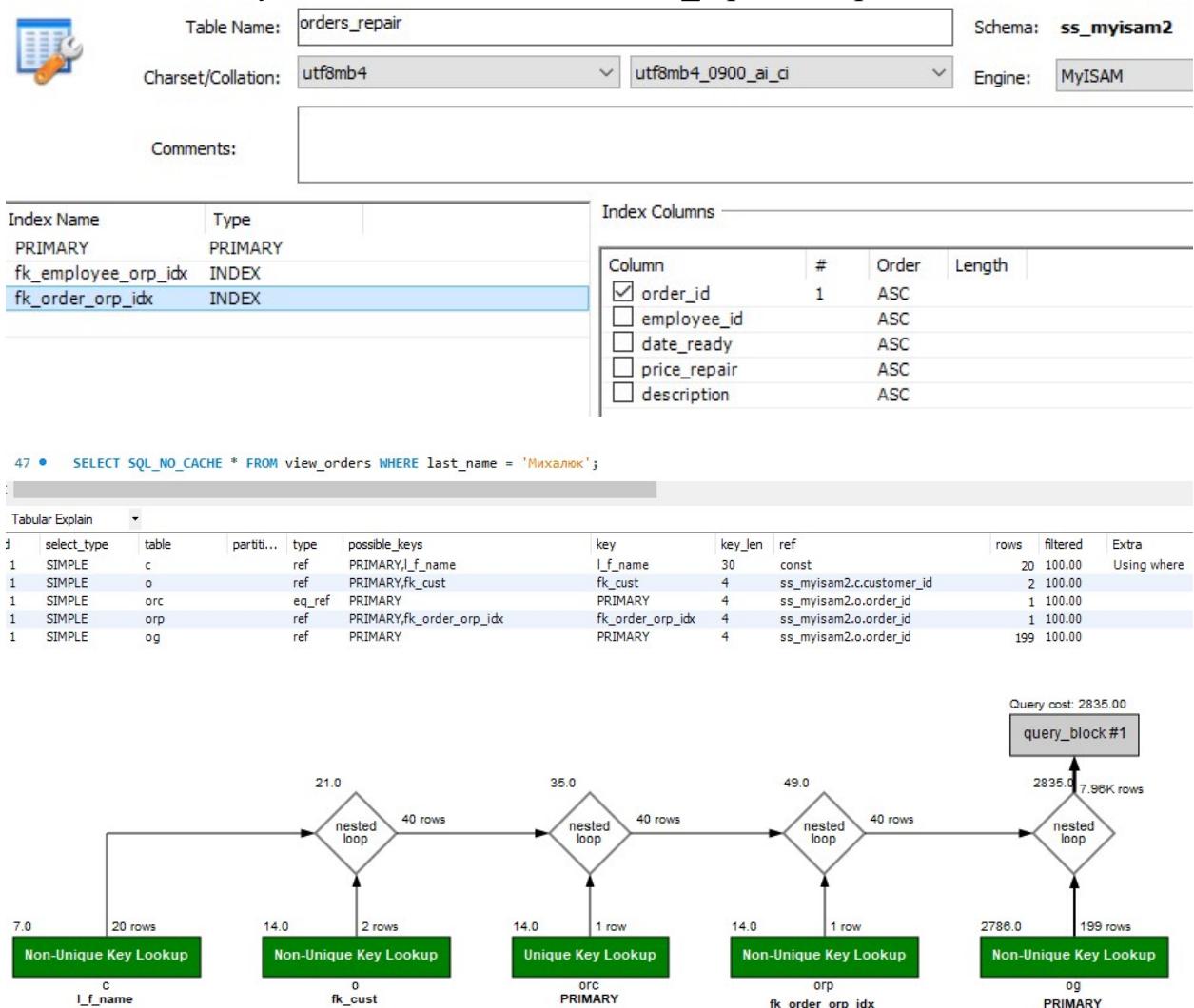


Рис. 6.55 Скріншот EXPLAIN оптимізації уявлення - MyISAM(після виправлення помилки)

Все працює як треба!

### Висновки:

- створили тимчасову таблицю одразу для трьох таблиц та випробували кілька запитів для неї, спробували доступ до неї з іншої сесії. Упевнилися, що дані пишуться тільки в тимчасову таблицю, але не пишуться в основні таблиці;
- розробили певний перелік уявлень для випробувань;
- простестували базове уявлення з алгоритмом MERGE, а також уявлення, яке створено на основі базового. Для перегляду даних, мені здається, досить корисний інструмент, одразу отримуємо всю потрібну інформацію з різних

таблиць. Одразу, якщо побачили помилку є можливість її виправити (якщо це дозволено внутрішньою логікою, наприклад, тригерами);

- далі мене зацікавив тест уявлення, що вставляє дані в таблиці, протестував цю можливість на спеціально створених окремих двох таблицях. Так, це працює, але наскільки це корисно на практиці, складно сказати. Но інструмент такий є, гарно, що навчились ім користуватись;
- далі попрацювали з уявленнями з алгоритмом TEMPTABLE, що мали агрегатні функції та групування. Як виявилося, групування веде к доволі суттєвим уповільненням запитів, бо створює тимчасову таблицю для BCIX рядків таблиці, що підлягає групуванню. Тому, чим меньша буде ця таблиця, тим буде швидше;
- поекспериментували з EXPLAIN оптимізацією запитів на основі уявень. Якщо вдається використати всі можливі індекси, то отримуємо дуже швидкий пошук. Якщо ні, буде повільно, особливо з групуванням. Провів дослідження, виявилося, що WHERE перед GROUP BY у 5-12 разів пришвидшує виконання запиту.
- на прикінці столкнувся з дивною поведінкою складеного індексу, яку поки не можу пояснити. Знайшов помилку в індексі, все тепер працює як треба. Ну що ж, Explain допоміг знайти дві помилки в індексах. Замечательно!