

Pokonałem optymalizator MariaDB: z 94 sekund do 55 milisekund

Sylvain ARBAUDIE · July 15, 2025

MARIADB

OPTIMIZER

PERFORMANCE

SQL

BEATING THE MARIADB OPTIMIZER — 94s TO 55ms
LEFT JOIN anti-pattern → CTE + EXCEPT — 1,700x improvement

LEFT JOIN + IS NULL

```
SELECT ... FROM products p
LEFT JOIN discontinued dp ON ...
WHERE dp.product_id IS NULL
```

3.5 billion row-by-row comparisons

CTE + EXCEPT

```
WITH active AS (SELECT ...)
SELECT ... FROM active
EXCEPT SELECT ... FROM discontinued
```

Hash-based set difference — 217K ops

94 SECONDS

55 MILLISECONDS

1,700x FASTER — think sets, not loops

Beat the optimizer by thinking like a mathematician, not a programmer

Kontekst: migracja MySQL 8 na MariaDB 11.4

Projekt migracji z MySQL 8 na MariaDB 11.4 przebiega ogólnie dobrze. Testy funkcjonalne przechodzą pomyślnie, testy wydajnościowe również — z wyjątkiem jednego zapytania. Jednego zapytania, które na MySQL 8 wykonywało się w 2 sekundy, a teraz na MariaDB 11.4 zajmuje **94 sekundy**.

To klasyczna regresja migracyjna: optymalizatory MySQL i MariaDB znacząco się rozeszły od momentu forka. Plan wykonania, który działał dobrze na jednym silniku, może być katastrofalny na drugim.

Problematyczne zapytanie

Oryginalne zapytanie wykorzystuje klasyczny wzorzec LEFT JOIN do znajdowania rekordów, które NIE istnieją w innej tabeli:

```
SELECT p.product_id, p.product_name, p.category_id
FROM products p
LEFT JOIN discontinued_products dp
```

```
ON p.product_id = dp.product_id
AND p.category_id = dp.category_id
WHERE dp.product_id IS NULL
AND p.status = 'active'
AND p.created_at >= '2023-01-01';
```

Intencja jest jasna: znaleźć wszystkie aktywne produkty, które nie figurują w tabeli produktów wycofanych. To wzorzec "anti-join" implementowany przez LEFT JOIN + IS NULL.

Dlaczego 94 sekundy?

Analiza planu wykonania na MariaDB 11.4 ujawnia problem. Optymalizator wybiera:

1. Skanowanie tabeli `products` używając indeksu na `status` (200 000 wierszy)
2. Dla każdego wiersza wykonanie skanu tabeli `discontinued_products` w celu weryfikacji braku dopasowania

Przy tabeli `discontinued_products` mającej 17 500 wierszy daje to około **3,5 miliarda porównań**. Optymalizator MySQL 8 wybierał inny plan z hash join, znacznie wydajniejszy dla tego wzorca.

Fundamentalny problem nie leży w optymalizatorze MariaDB — to samo zapytanie jest wadliwe. LEFT JOIN + IS NULL do implementacji anti-join to historyczny anty-wzorzec z epoki, gdy SQL nie miał lepszych alternatyw.

Rozwiązanie: CTE + EXCEPT

MariaDB obsługuje Common Table Expressions (CTE) od wersji 10.2 i operator `EXCEPT` od wersji 10.3. Te dwie funkcjonalności pozwalają na przepisanie logiki w znacznie bardziej wyrazisty sposób:

```
WITH active_products AS (
  SELECT product_id, category_id
  FROM products
  WHERE status = 'active'
  AND created_at >= '2023-01-01'
),
still_active AS (
  SELECT product_id, category_id FROM active_products
  EXCEPT
```

```
SELECT product_id, category_id FROM discontinued_products
)
SELECT p.product_id, p.product_name, p.category_id
FROM products p
JOIN still_active sa
ON p.product_id = sa.product_id
AND p.category_id = sa.category_id;
```

Dlaczego to szybsze

1. **CTE** `active_products` materializuje 200 000 aktywnych produktów w pamięci. Jedno skanowanie tabeli `products`.
2. **Operator** `EXCEPT` wykonuje operację zbiorową: bierze zbiór aktywnych produktów i odejmuje od niego te, które pojawiają się w `discontinued_products`. To hash-based set difference, nie porównanie wiersz po wierszu.
3. **Końcowy JOIN** z CTE `still_active` to prosty lookup na już przefiltrowanym zbiorze.

Wynik: 55 milisekund

Metryka	LEFT JOIN	CTE + EXCEPT	Poprawa
Czas	94 sekundy	55 ms	1 700x
Porównania	~3,5 mld	~217 500	99,99%
Podejście	Wiersz po wierszu	Zbiorowe	—

Z 94 sekund do 55 milisekund. Czynniki **1 700**. Nie przez zmianę parametru konfiguracji. Nie przez dodanie indeksu. Przez **przemyślenie logiki zapytania**.

Anty-wzorzec LEFT JOIN: dlaczego przetrwał

Wzorzec `LEFT JOIN + IS NULL` dla anti-join jest wszędzie. Znajduje się w samouczkach, kursach online, odpowiedziach na Stack Overflow. Przetrwał z kilku powodów:

Historyczny: Przed SQL:1999 nie istniał `EXCEPT`, nie było zoptymalizowanego `NOT EXISTS`, nie było CTE. `LEFT JOIN + IS NULL` był jedyną przenośną opcją.

Nawyk: Programiści uczą się wzorca, który działa i go powtarzają. "Działa" jest wrogiem "jest optymalne".

Kompatybilność: LEFT JOIN działa na wszystkich wersjach wszystkich RDBMS. EXCEPT jest obsługiwany tylko przez nowsze wersje.

Alternatywy do poznania

Dla anti-joinów istnieją trzy alternatywy:

NOT EXISTS (często najlepszy wybór)

```
SELECT p.product_id, p.product_name, p.category_id
FROM products p
WHERE p.status = 'active'
      AND p.created_at >= '2023-01-01'
      AND NOT EXISTS (
  SELECT 1 FROM discontinued_products dp
  WHERE dp.product_id = p.product_id
        AND dp.category_id = p.category_id
);
```

NOT EXISTS jest generalnie dobrze optymalizowany przez oba silniki MariaDB i MySQL. Optymalizator może użyć odwróconego semi-joina, znacznie wydajniejszego niż LEFT JOIN.

NOT IN (uwaga na NULL)

```
SELECT product_id, product_name, category_id
FROM products
WHERE status = 'active'
      AND created_at >= '2023-01-01'
      AND (product_id, category_id) NOT IN (
  SELECT product_id, category_id
  FROM discontinued_products
);
```

Uwaga: NOT IN ma podstępne zachowanie z wartościami NULL. Jeśli discontinued_products.product_id może być NULL, semantyka NOT IN zwraca pusty wynik. Zawsze używaj NOT EXISTS, jeśli wartości NULL są możliwe.

EXCEPT (najbardziej czytelny)

```
SELECT product_id, category_id FROM products
WHERE status = 'active' AND created_at >= '2023-01-01'
EXCEPT
SELECT product_id, category_id FROM discontinued_products;
```

`EXCEPT` to najczystsze wyrażenie operacji zbiorowej "A minus B". Ale zwraca tylko kolumny operacji, nie dodatkowe kolumny — stąd użycie CTE do ponownego wprowadzenia brakujących kolumn przez JOIN.

Lekcja

Optymalizator to narzędzie, nie cud. Gdy zapytanie jest wolne, pierwsze pytanie nie powinno brzmieć "jaką wskazówkę mogę dodać?" czy "jaki parametr zmienić?". Pierwsze pytanie powinno brzmieć: **czy moje zapytanie prawidłowo wyraża moją intencję?**

LEFT JOIN + IS NULL wyraża "złącz, a potem odfiltruj brak dopasowań". `EXCEPT` wyraża bezpośrednio "odejmij ten zbiór od tego innego zbioru". Druga formuła jest bliższa intencji biznesowej i daje optymalizatorowi znacznie lepszą szansę na znalezienie wydajnego planu.

Pokonaj optymalizator, myśląc jak matematyk, nie jak programista.

Ten artykuł został pierwotnie opublikowany na [Medium](#).