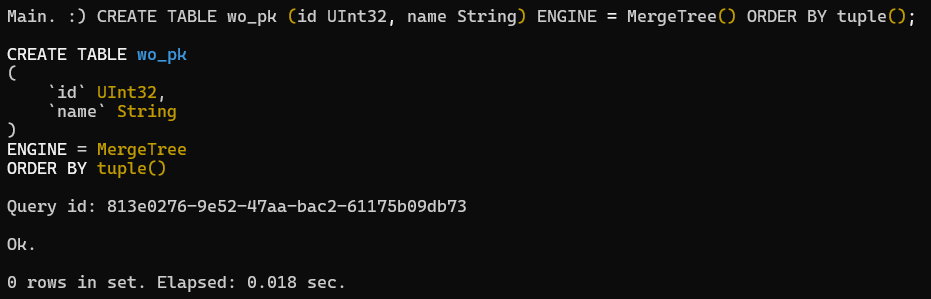
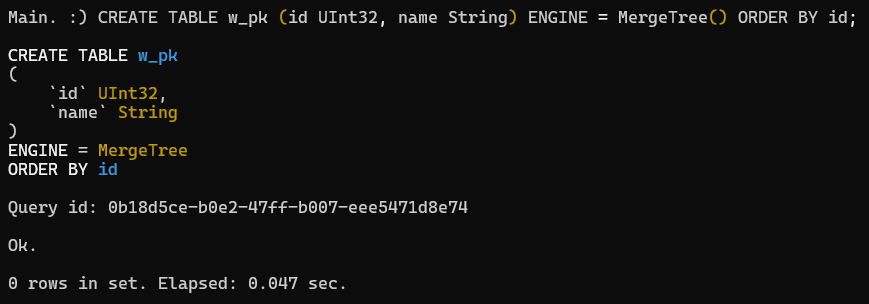
* Создание запроса без первичного ключа

**CREATE TABLE wo\_pk (id UInt32, name String) ENGINE = MergeTree() ORDER BY tuple();**



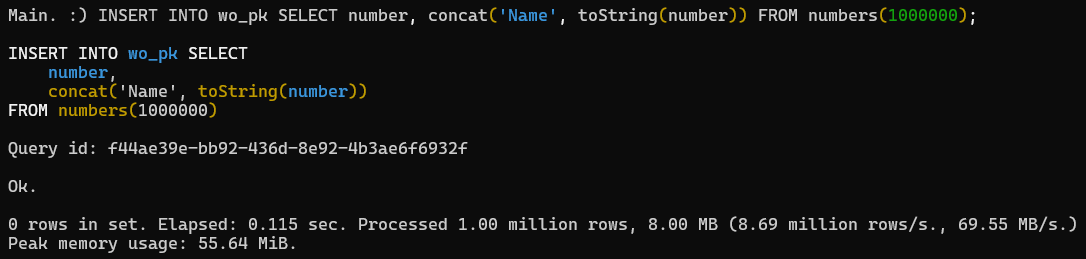
* Создание запроса с первичным ключом

**CREATE TABLE w\_pk (id UInt32, name String) ENGINE = MergeTree() ORDER BY id;**



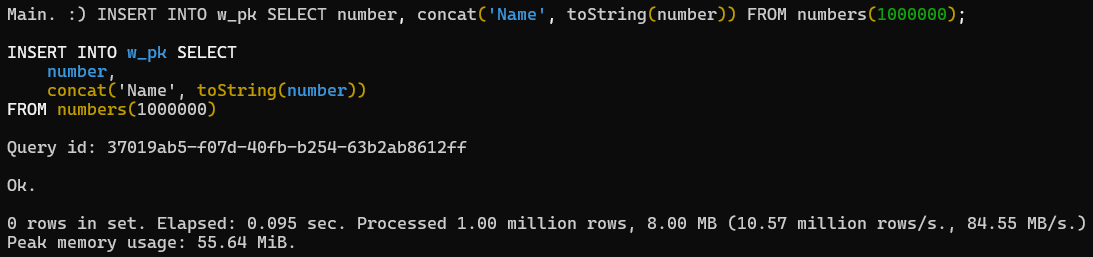
* Заполнение данными таблицы без первичного ключа

**INSERT INTO wo\_pk SELECT number, concat('Name', toString(number)) FROM numbers(1000000);**



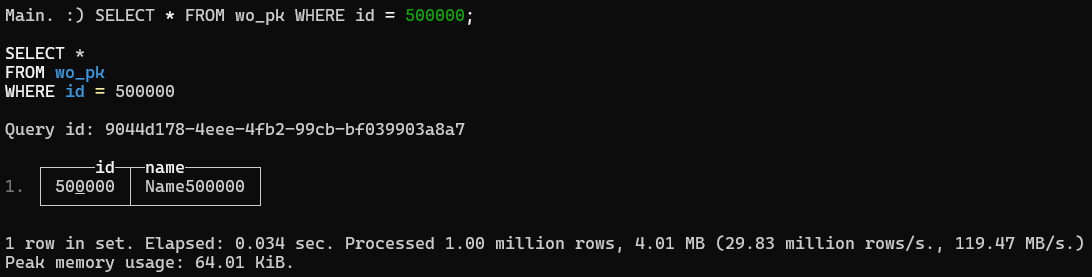
* Заполнение данными таблицы с первичным ключом

**INSERT INTO w\_pk SELECT number, concat('Name', toString(number)) FROM numbers(1000000);**

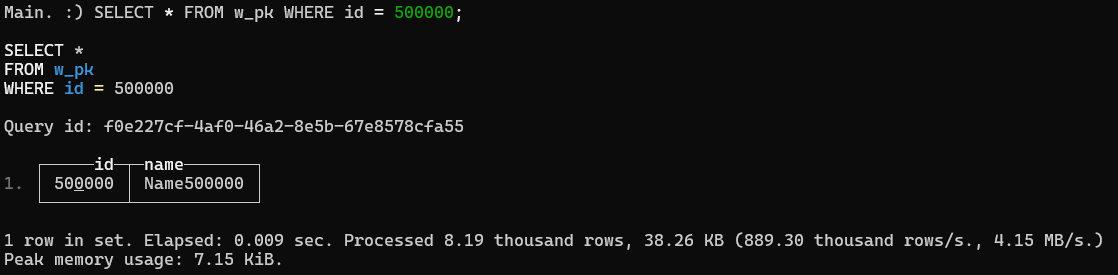


* Выполнение запросов с условием по полю «id»

**SELECT \* FROM wo\_pk WHERE id = 500000;**



**SELECT \* FROM w\_pk WHERE id = 500000;**



* Анализ информации из логов для таблицы без первичного ключа

**SELECT**

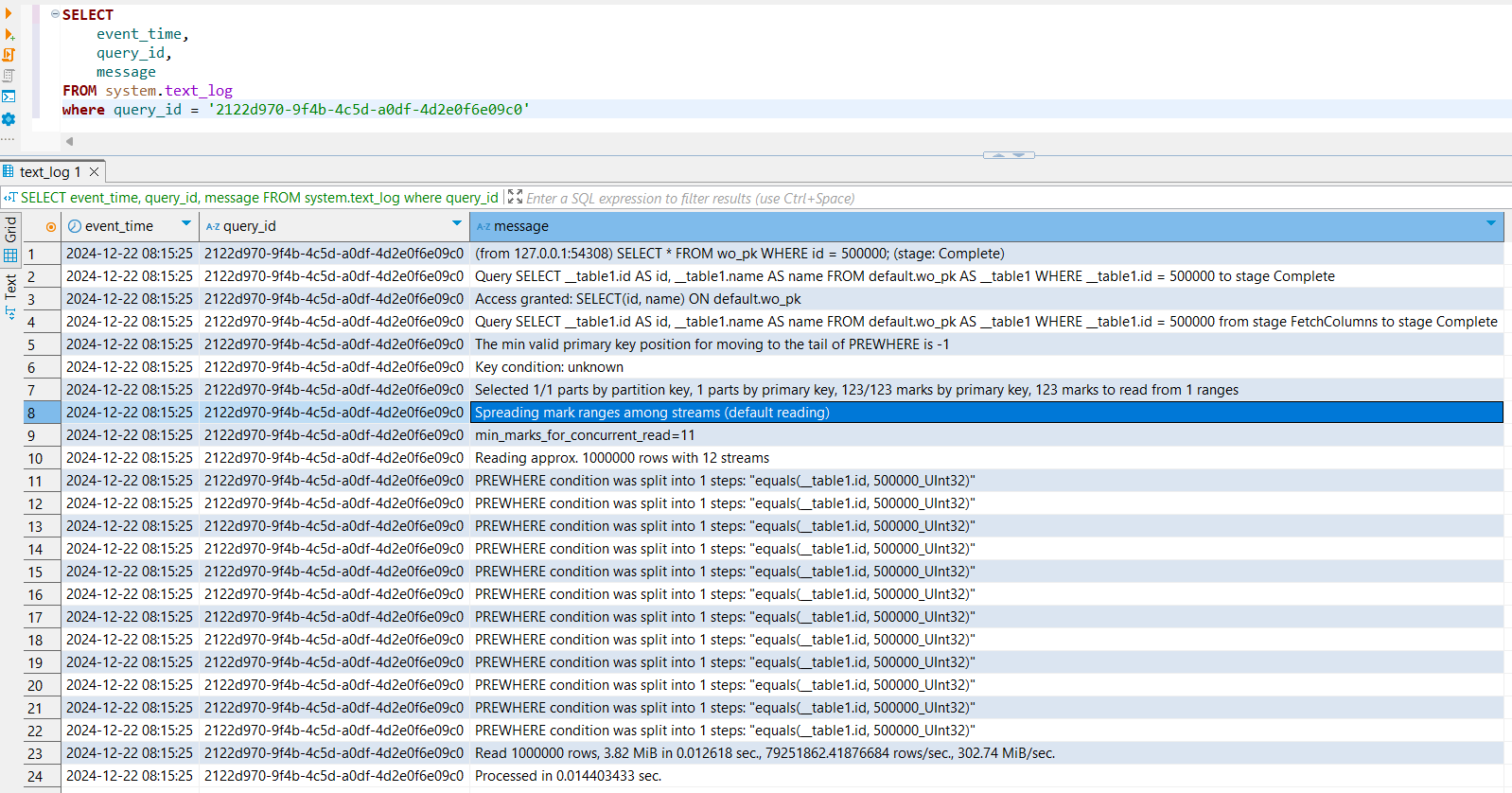
**event\_time,**

**query\_id,**

**message**

**FROM system.text\_log**

**where query\_id = '2122d970-9f4b-4c5d-a0df-4d2e0f6e09c0'**

****

От сюда видно, что идет сканирование всей таблицы:

**Selected 1/1 parts by partition key, 1 parts by primary key, 123/123 marks by primary key, 123 marks to read from 1 ranges**

В 12 потоков:

**Reading approx. 1000000 rows with 12 streams**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**PREWHERE condition was split into 1 steps: "equals(\_\_table1.id, 500000\_UInt32)"**

**Read 1000000 rows, 3.82 MiB in 0.012618 sec., 79251862.41876684 rows/sec., 302.74 MiB/sec.**

* Анализ информации из логов для таблицы с первичным ключом

**SELECT**

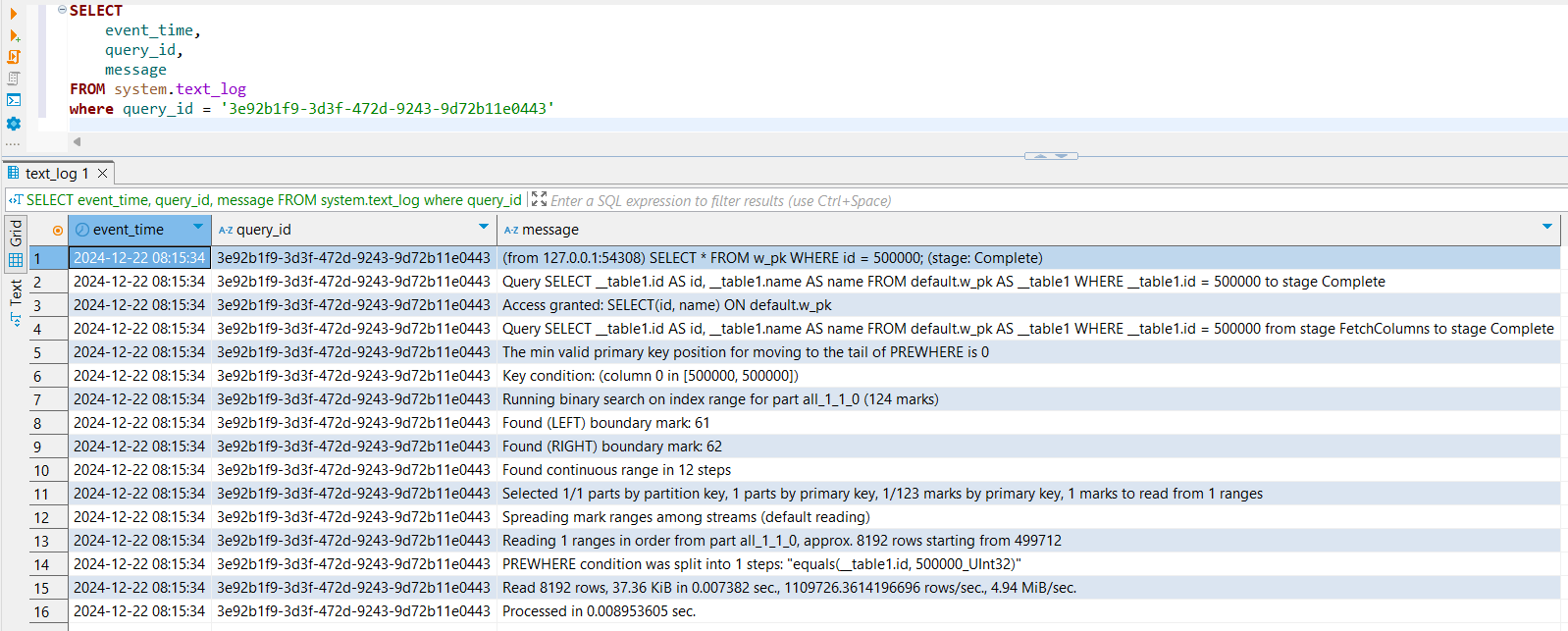
event\_time,

query\_id,

message

**FROM** system.text\_log

**where** query\_id = '3e92b1f9-3d3f-472d-9243-9d72b11e0443'



Вместо сканирования всей таблицы видно, что идет чтение только одной парты:

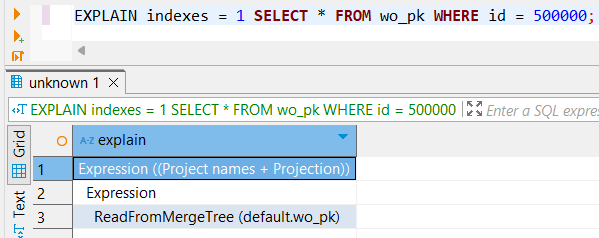
**Selected 1/1 parts by partition key, 1 parts by primary key, 1/123 marks by primary key, 1 marks to read from 1 ranges**

**Reading 1 ranges in order from part all\_1\_1\_0, approx. 8192 rows starting from 499712**

**Read 8192 rows, 37.36 KiB in 0.007382 sec., 1109726.3614196696 rows/sec., 4.94 MiB/sec.**

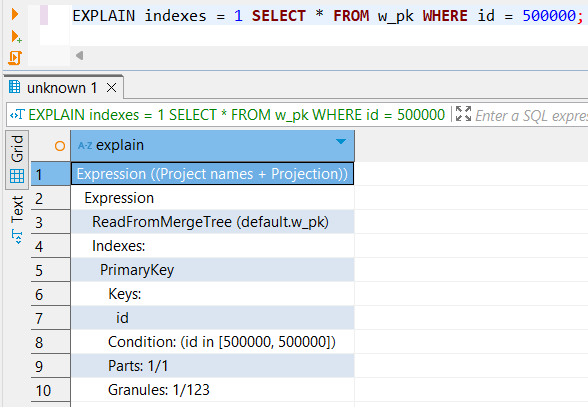
* Анализ плана запроса к таблице без первичного ключа

EXPLAIN indexes = 1 **SELECT** \* **FROM** wo\_pk **WHERE** id = 500000;



Видно, что в плане запроса отсутствует использование первичного ключа и происходит сканирование всей таблицы

* Анализ плана запроса к таблице с первичным ключом



Видно, что в плане запроса используется первичный ключ для быстрого поиска партов по предикату с условием в запросе. При этом из всех гранул и партов читается только одна гранула и одна парта