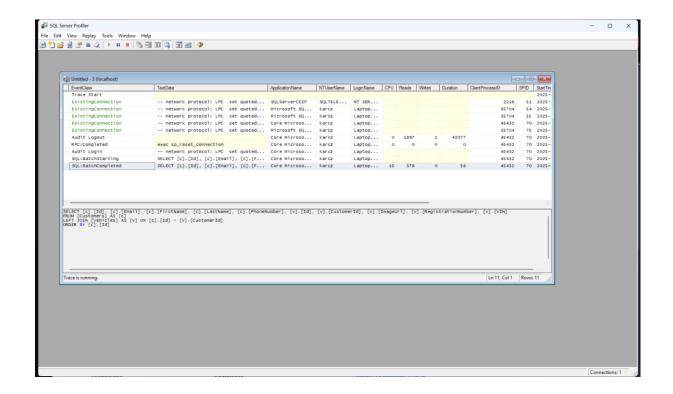
Analiza Endpointu Customer

Będziemy nasłuchiwać poniższy endpoint:

```
// GET: Customers
        public async Task<IActionResult> Index()
            _logger.LogInformation("Rozpoczęcie pobierania listy wszystkich
klientów"):
            try
            {
                var customers = await context.Customers
                    .Include(c => c.Vehicles)
                    .ToListAsync();
                _logger.LogInformation("Pobrano {Count} klientów wraz z
{VehicleCount} pojazdami",
                    customers.Count, customers.SelectMany(c =>
c.Vehicles).Count());
                return View(customers);
            catch (Exception ex)
                _logger.LogError(ex, "Błąd podczas pobierania listy
klientów");
                TempData["ErrorMessage"] = "Wystąpił błąd podczas
pobierania listy klientów.";
                return View(new List<Customer>());
            }
        }
```

Po skonfigurowaniu SQL Server Profiler oraz uruchomieniu naszej aplikacji i wywołaniu powyższego endpointu uzyskaliśmy następujące wyniki:



Zapytanie wykonuje LEFT JOIN między tabelą główną Customers (alias [c]) a tabelą powiązaną Vehicles (alias [v]). Warunek łączenia to [c].[Id] = [v].[CustomerId], co oznacza, że dla każdego klienta pobierane są wszystkie jego pojazdy. ORDER BY [c].[Id] sortuje wyniki według ID klienta dla zachowania spójnej kolejności

Czas wykonywania tej operacji to 56ms. Operacje wejścia/wyjścia podczas wykonania zapytania wyniosły 378 logical reads oraz 0 physical reads