

INDEKS NUMER 1

PRZED:

Jak widzimy zapytanie przed optymalizacją wykonuje Clustered Index Scan na TrainingSessions, czyli pełne przeszukiwanie tabeli, gdzie sort na StartTime (koszt aż 77%). Jednak SQL Server musi przeszukać całą tabelę, co chcemy poprawić przy częstych przeszukiwaniach.

The screenshot shows the SQL Server Enterprise Manager interface. At the top, a tab is labeled "SQLQuery1.sql - lo...AWELEK\pault (71))*". The main window displays a SQL query:

```
SELECT
    t.Id,
    t.StartTime,
    t.DurationInMinutes,
    t.MemberId,
    t.IsGroupSession
FROM dbo.TrainingSessions AS t
WHERE t.TrainerId = 2
ORDER BY t.StartTime;
```

Below the query, the "Execution plan" tab is selected. It shows three queries with their respective costs:

- Query 1: Query cost (relative to the batch): 0%
SET STATISTICS IO, TIME ON;
- Query 2: Query cost (relative to the batch): 0%
DECLARE @TrainerId INT = 2; -- any existing trainer id
- Query 3: Query cost (relative to the batch): 100%
SELECT t.Id, t.StartTime, t.DurationInMinutes, t.MemberId, t.IsG...

The execution plan for Query 3 is shown at the bottom. It consists of three steps:

- SELECT**: Cost: 0 %
- Sort**: Cost: 77 %
- Clustered Index Scan (Cluste... [TrainingSessions].[PK_Train...]**: Cost: 23 %

Arrows indicate the flow of data from the Clustered Index Scan to the Sort step, and then to the SELECT step.

WHERE t.TrainerId = 2
ORDER BY t.StartTime;

100 %

Messages Execution plan

Query 1: Query cost (relative to the b
SET STATISTICS IO, TIME ON;

T-SQL
SET STAT
Cost: 0 %

Query 2: Query cost (relative to the b
DECLARE @TrainerId INT = 2; -- any exi

T-SQL
ASSIGN
Cost: 0 %

Query 3: Query cost (relative to the b
SELECT t.Id, t.StartTime, t.DurationIn

SELECT Sort Clustered Index
Cost: 0 % Cost: 77 % Cost:

Query executed successfully.

Clustered Index Scan (Clustered)
Scanning a clustered index, entirely or only a range.

Physical Operation	Clustered Index Scan
Logical Operation	Clustered Index Scan
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.0034019 (23%)
Estimated I/O Cost	0.003125
Estimated Subtree Cost	0.0034019
Estimated CPU Cost	0.0002769
Estimated Number of Executions	1
Estimated Number of Rows to be Read	109
Estimated Number of Rows for All Executions	27.25
Estimated Number of Rows Per Execution	27.25
Estimated Row Size	32 B
Ordered	False
Node ID	1

Predicate
[GymManagerDb].[dbo].[TrainingSessions].[TrainerId] as [t].[TrainerId] =
[@TrainerId]

Object
[GymManagerDb].[dbo].[TrainingSessions].[PK_TrainingSessions] [t]

Output List
[GymManagerDb].[dbo].[TrainingSessions].Id, [GymManagerDb].[dbo].
[TrainingSessions].StartTime, [GymManagerDb].[dbo].
[TrainingSessions].IsGroupSession, [GymManagerDb].[dbo].
[TrainingSessions].MemberId, [GymManagerDb].[dbo].
[TrainingSessions].DurationInMinutes

Desktop 3 localhost\MS

SQLQuery1.sql - lo...AWELEK\pault (71)*

```
CREATE NONCLUSTERED INDEX TrainingSessions_Trainer_StartTime_INDEX
ON dbo.TrainingSessions (TrainerId, StartTime)
INCLUDE (MemberId, DurationInMinutes, IsGroupSession);
```

100 %

Messages

SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.

Table 'TrainingSessions'. Scan count 1, logical reads 3, physical reads 1, page server read:

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 30 ms.

Completion time: 2025-06-09T18:24:54.7122958+02:00

SQLQuery1.sql - lo...AWELEK\pault (71))*

```

SET STATISTICS IO, TIME ON;

SELECT
    t.Id,
    t.StartTime,
    t.DurationInMinutes,
    t.MemberId,
    t.IsGroupSession
FROM dbo.TrainingSessions t
WHERE t.TrainerId = 2
ORDER BY t.StartTime;

```

100 %

Messages Execution plan

Query 1: Query cost (rel)

SET STATISTICS IO, TIME

T-SQL

SET STATS

Cost: 0

Query 2: Query cost (rel)

SELECT t.Id, t.StartTime

Index Seek (NonClustered)

Cost: 0

Index Seek (NonClustered)

Scan a particular range of rows from a nonclustered index.

Index Seek (NonClustered)	
Physical Operation	Index Seek
Logical Operation	Index Seek
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.0032853 (100%)
Estimated I/O Cost	0.003125
Estimated Subtree Cost	0.0032853
Estimated CPU Cost	0.0001603
Estimated Number of Executions	1
Estimated Number of Rows to be Read	3
Estimated Number of Rows for All Executions	3
Estimated Number of Rows Per Execution	3
Estimated Row Size	28 B
Ordered	True
Node ID	0
Object	[GymManagerDb].[dbo].[TrainingSessions].
	[TrainingSessions_Trainer_StartTime_INDEX] [t]
Output List	[GymManagerDb].[dbo].[TrainingSessions].Id, [GymManagerDb].[dbo].
	[TrainingSessions].StartTime, [GymManagerDb].[dbo].
	[TrainingSessions].IsGroupSession, [GymManagerDb].[dbo].
	[TrainingSessions].MemberId, [GymManagerDb].[dbo].
	[TrainingSessions].DurationInMinutes
Seek Predicates	
Seek Keys[1]: Prefix:	[GymManagerDb].[dbo].
	[TrainingSessions].TrainerId = Scalar Operator(CONVERT_IMPLICIT(int,

Query executed successfully.

localhost\M

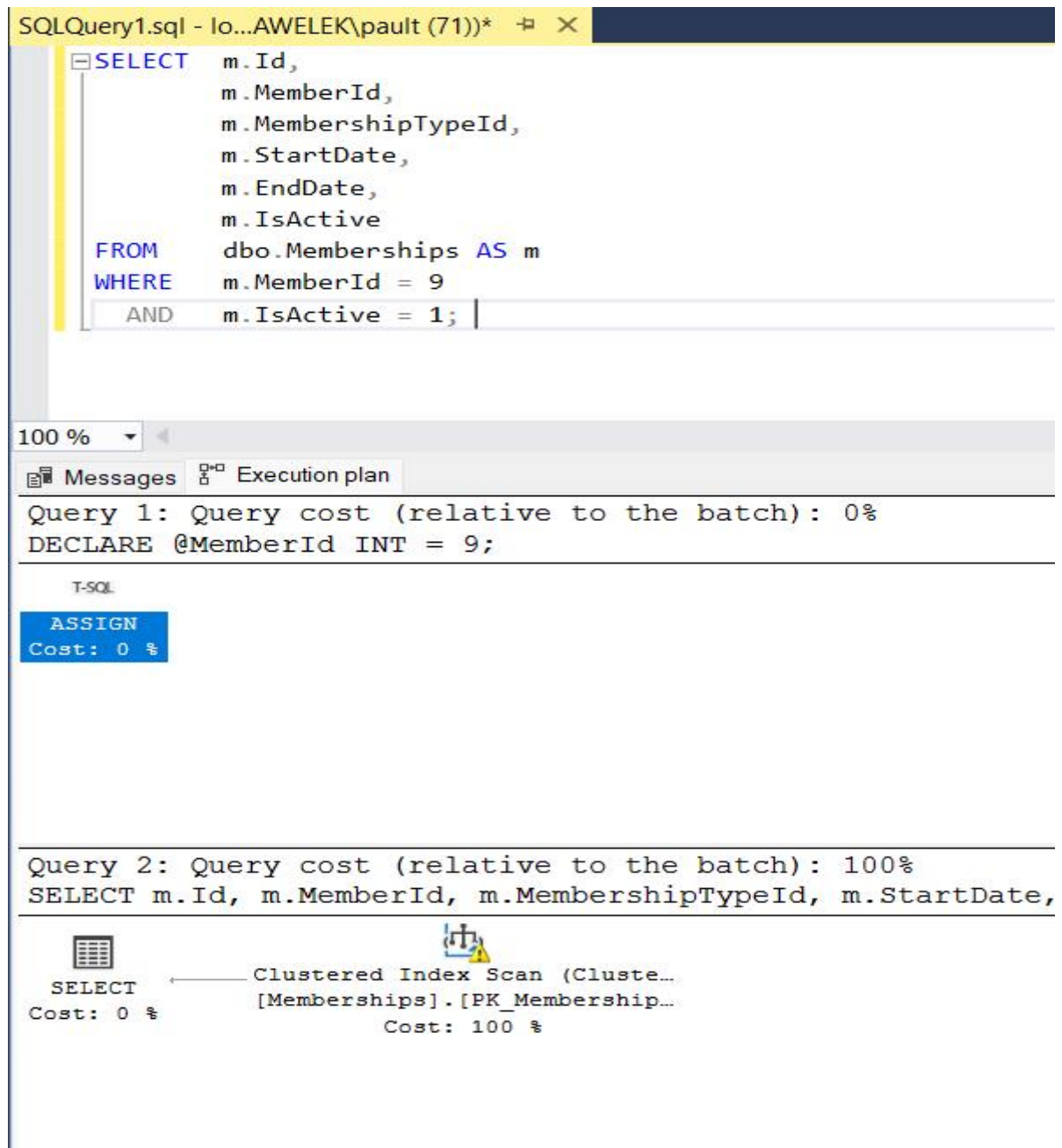
PO:

SQL Server korzysta teraz z Index Seek (NonClustered), dzięki czemu sort zniknął, ponieważ indeks już przechowuje dane w wymaganej kolejności. Widzimy mamy tylko przejście po 3 recordach – większa wydajność, spadł koszt.

INDEKS NUMER 2

PRZED:

Zapytanie wykonuje Clustered Index Scan na tabeli Memberships, co oznacza pełne przeszukiwanie tabeli, mimo że filtrujemy po MemberId oraz IsActive. W efekcie koszt operatora wynosi 100%, a liczba odczytów logicznych jest większa niż potrzebna. To zapytanie może być wykonywane często (np. sprawdzanie, czy dany użytkownik ma aktywną subskrypcję), więc wymaga optymalizacji.



The screenshot displays the SQL Server Enterprise Manager interface. At the top, a tab for 'SQLQuery1.sql' is open. The query editor shows the following SQL statement:

```
SELECT m.Id,
       m.MemberId,
       m.MembershipTypeId,
       m.StartDate,
       m.EndDate,
       m.IsActive
FROM   dbo.Memberships AS m
WHERE  m.MemberId = 9
AND    m.IsActive = 1;
```

Below the query editor, the 'Execution plan' tab is selected. It shows the execution plan for the query. The plan consists of two queries:

- Query 1:** Query cost (relative to the batch): 0%. It is a T-SQL statement: `DECLARE @MemberId INT = 9;`. The execution plan for this query is a simple 'ASSIGN' operator with a cost of 0%.
- Query 2:** Query cost (relative to the batch): 100%. It is a SELECT statement: `SELECT m.Id, m.MemberId, m.MembershipTypeId, m.StartDate,`. The execution plan for this query is a 'Clustered Index Scan (Clustered Index on [Memberships].[PK_Membership...])' with a cost of 100%.

The execution plan for Query 2 is visualized with a diagram showing a table icon connected to the 'Clustered Index Scan' operator. The cost of the scan is 100%.

SQL Query (T-SQL) - 10...AWLEEK\paul (7/17)

SELECT

m.Id,

m.MemberId,

m.MembershipTypeId,

m.StartDate,

m.EndDate,

m.IsActive

FROM

dbo.Memberships AS

WHERE

m.MemberId = 9

AND

m.IsActive = 1;

100 %

Messages

Execution plan

Query 1: Query cost (relative)

DECLARE @MemberId INT = 9;

T-SQL

ASSIGN

Cost: 0 %

Query 2: Query cost (relative)

SELECT m.Id, m.MemberId, m

SELECT

Cost: 0 %

Clustered Index S

[Memberships].[PK

Cost: 10

Clustered Index Scan (Clustered)

Scanning a clustered index, entirely or only a range.

Physical Operation	Clustered Index Scan
Logical Operation	Clustered Index Scan
Estimated Execution Mode	Row
Storage	RowStore
Estimated I/O Cost	0.003125
Estimated Operator Cost	0.0032952 (100%)
Estimated CPU Cost	0.0001702
Estimated Subtree Cost	0.0032952
Estimated Number of Executions	1
Estimated Number of Rows to be Read	12
Estimated Number of Rows for All Executions	2
Estimated Number of Rows Per Execution	2
Estimated Row Size	36 B
Ordered	False
Node ID	0

Predicate

[GymManagerDb].[dbo].[Memberships].[MemberId] as [m].[MemberId]=

[@MemberId] AND [GymManagerDb].[dbo].[Memberships].[IsActive] as

[m].[IsActive]=(1)

Object

[GymManagerDb].[dbo].[Memberships].[PK_Memberships] [m]

Output List

[GymManagerDb].[dbo].[Memberships].Id, [GymManagerDb].[dbo].

[Memberships].MemberId, [GymManagerDb].[dbo].

[Memberships].MembershipTypeId, [GymManagerDb].[dbo].

[Memberships].StartDate, [GymManagerDb].[dbo].

[Memberships].EndDate, [GymManagerDb].[dbo].[Memberships].IsActive

Warnings

Columns With No Statistics: [GymManagerDb].[dbo].

[Memberships].IsActive

Query executed successfully.

SQLQuery1.sql - lo...AWELEK\pault (71))* -# X

```
CREATE NONCLUSTERED INDEX Memberships_MemberId_IsActive_Index
ON dbo.Memberships (MemberId, IsActive)
INCLUDE (StartDate, EndDate, MembershipTypeId);
```

100 %

Messages

SQL Server parse and compile time:

CPU time = 0 ms, elapsed time = 0 ms.

Table 'Memberships'. Scan count 1, logical reads 2, physical reads 1, page server read

SQL Server Execution Times:

CPU time = 0 ms, elapsed time = 16 ms.

Completion time: 2025-06-09T18:48:39.2331808+02:00

SQLQuery1.sql - lo...AWELEK\pault (71))* -# X

```
SELECT m.Id,
       m.MemberId,
       m.MembershipTypeId,
       m.StartDate,
       m.EndDate,
       m.IsActive
FROM   dbo.Memberships AS
WHERE  m.MemberId = 9
AND    m.IsActive = 1;
```

Index Seek (NonClustered)

Scan a particular range of rows from a nonclustered index.

Physical Operation	Index Seek
Logical Operation	Index Seek
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.0032831 (100%)
Estimated I/O Cost	0.003125
Estimated Subtree Cost	0.0032831
Estimated CPU Cost	0.0001581
Estimated Number of Executions	1
Estimated Number of Rows Per Execution	1
Estimated Number of Rows to be Read	1
Estimated Number of Rows for All Executions	1
Estimated Row Size	36 B
Ordered	True
Node ID	0

Object

[GymManagerDb].[dbo].[Memberships].
[Memberships_MemberId_IsActive_Index] [m]

Output List

[GymManagerDb].[dbo].[Memberships].Id, [GymManagerDb].[dbo].
[Memberships].MemberId, [GymManagerDb].[dbo].
[Memberships].MembershipTypeId, [GymManagerDb].[dbo].
[Memberships].StartDate, [GymManagerDb].[dbo].
[Memberships].EndDate, [GymManagerDb].[dbo].
[Memberships].IsActive

Warnings

Columns With No Statistics: [GymManagerDb].[dbo].
[Memberships].IsActive

Seek Predicates

Seek Keys[1]: Prefix: [GymManagerDb].[dbo].
[Memberships].MemberId, [GymManagerDb].[dbo].
[Memberships].IsActive = Scalar Operator(CONVERT_IMPLICIT(int

100 %

Messages

Query 1: Query cost (relative)
SELECT m.Id, m.MemberId, m.

SELECT
Cost: 0 %

Index Seek (NonC
[Memberships].[Mem
Cost: 100

Query executed successfully.

PO:

SQL Server korzysta teraz z Index Seek (NonClustered) na Memberships, co znacząco poprawia wydajność zapytania. Dzięki nowemu indeksowi (MemberId, IsActive), system od razu trafia do odpowiednich rekordów bez konieczności skanowania całej tabeli.

Zniknęła operacja Clustered Index Scan, a odczyty zostały ograniczone tylko do niezbędnych danych (widzimy przejście po zaledwie 1 rekordzie). Całkowity koszt operatora spadł, a czas wykonania zapytania uległ skróceniu.