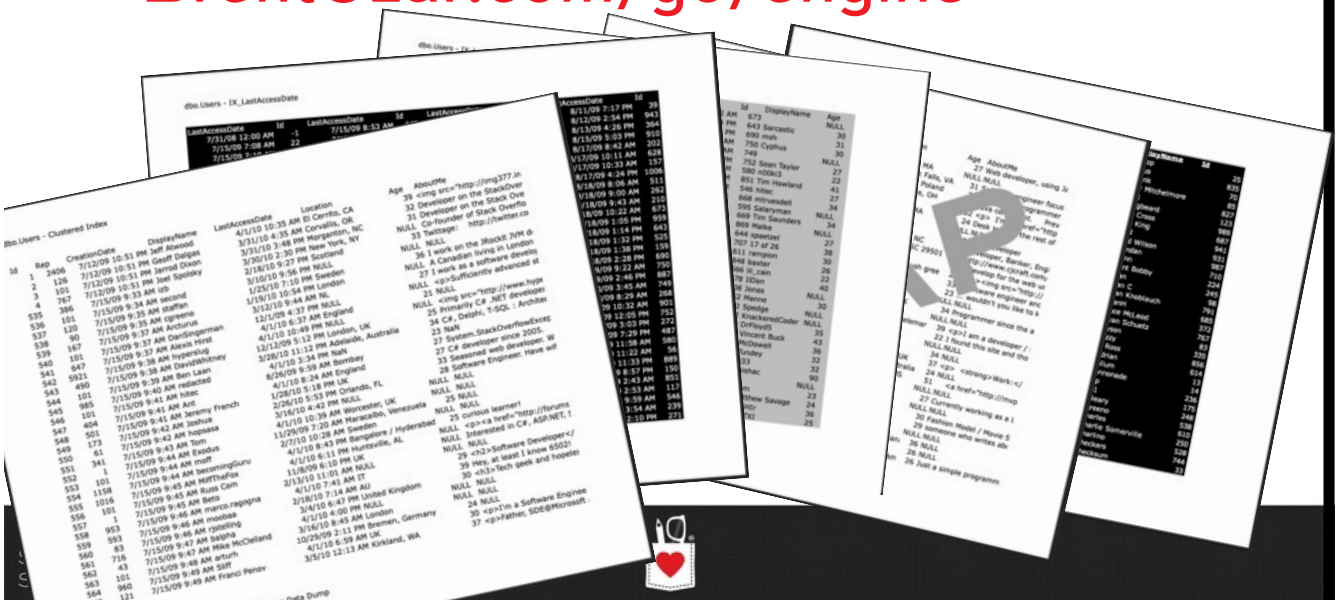
# How to Think Like the SQL Server Engine

## Part 1: Clustered Indexes, Filtering, Sorting

Slide 1
Slides, scripts, db: BrentOzar.com/go/engine

## Get the 5-page PDF: BrentOzar.com/go/engine



© Brent Ozar Unlimited®. All rights reserved.

# Source: StackOverflow2010 db

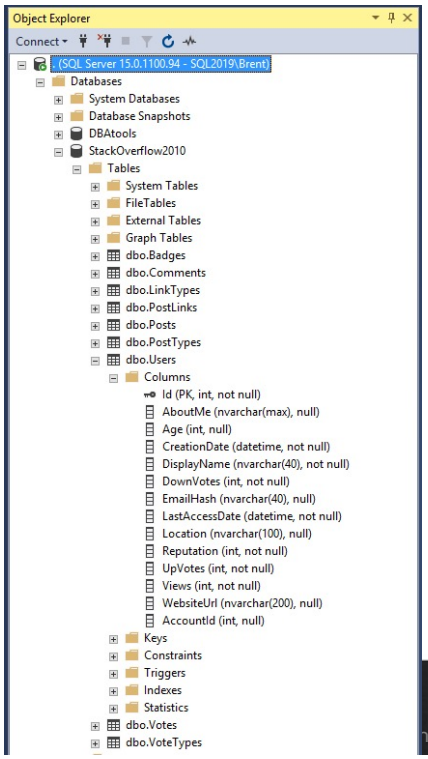
Open source, licensed with Creative Commons

SQL Server: https://BrentOzar.com/go/querystack

XML: https://archive.org/details/stackexchange

I'm using SQL Server 2019,
compatibility level 150/2019.
My cost threshold for parallelism is 5.






© Brent Ozar Unlimited®. All rights reserved.

dbo.PostTypes

dbo.Users

- Columns
  - Id (PK, int, not null)
  - AboutMe (nvarchar(max), null)
  - Age (int, null)
  - CreationDate (datetime, not null)
  - DisplayName (nvarchar(40), not null)
  - DownVotes (int, not null)
  - EmailHash (nvarchar(40), null)
  - LastAccessDate (datetime, not null)
  - Location (nvarchar(100), null)
  - Reputation (int, not null)
  - UpVotes (int, not null)
  - Views (int, not null)
  - WebsiteUrl (nvarchar(200), null)
  - AccountId (int, null)
- Keys

Slide 5
Slides, scripts, db: BrentOzar.com/go/engine

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.in |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/31/10 3:48 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 3/30/10 2:30 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 2/18/10 9:27 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 3/10/10 9:56 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/25/10 7:10 PM | Sweden | 36 | I work on the JRockit JVM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 1/19/10 10:54 PM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 3/12/10 9:44 AM | NL | 27 | I work as a software develo |
| 540 | 101 | 7/15/09 9:37 AM | DanSingerman | 12/1/09 4:37 PM | NULL | NULL | <p>Sufficiently advanced st |
| 541 | 647 | 7/15/09 9:37 AM | Alexis Hirst | 4/1/10 6:37 AM | England | 21 | NULL |
| 542 | 5921 | 7/15/09 9:38 AM | hyperslug | 4/1/10 10:49 PM | NULL | NULL | <img src="http://www.hype |
| 543 | 490 | 7/15/09 9:38 AM | DavidWhitney | 12/12/09 5:12 PM | London, UK | 25 | Primarily C# .NET developer |
| 544 | 101 | 7/15/09 9:39 AM | Ben Laan | 3/28/10 11:12 PM | Adelaide, Australia | 34 | C#, Delphi, T-SQL : Architec |
| 545 | 985 | 7/15/09 9:40 AM | redacted | 4/1/10 3:34 PM | NaN | 23 | NaN |
| 546 | 101 | 7/15/09 9:41 AM | hitec | 8/26/09 9:59 AM | Bombay | 27 | System.StackOverflowExcep |
| 547 | 404 | 7/15/09 9:41 AM | Ant | 4/1/10 8:24 AM | England | 27 | C# developer since 2005. |
| 548 | 501 | 7/15/09 9:41 AM | Jeremy French | 1/28/10 5:18 PM | UK | 33 | Seasoned web developer. W |
| 549 | 173 | 7/15/09 9:42 AM | Joshua | 2/26/10 5:53 PM | Orlando, FL | 28 | Software Engineer. Have wif |
| 550 | 61 | 7/15/09 9:42 AM | hopsasa | 3/16/10 4:42 PM | NULL | NULL | NULL |
| 551 | 341 | 7/15/09 9:43 AM | Tom | 4/1/10 10:39 AM | Worcester, UK | NULL | NULL |
| 552 | 1 | 7/15/09 9:44 AM | Exodus | 11/29/09 7:20 AM | Maracaibo, Venezuela | 25 | NULL |
| 553 | 101 | 7/15/09 9:44 AM | moff | 2/7/10 10:28 AM | Sweden | NULL | NULL |
| 554 | 1158 | 7/15/09 9:44 AM | becomingGuru | 4/1/10 8:43 PM | Bangalore / Hyderabad | 25 | curious learner! |
| 555 | 1016 | 7/15/09 9:45 AM | MiffTheFox | 4/1/10 6:11 PM | Huntsville, AL | NULL | <p><a href="http://forums |
| 556 | 101 | 7/15/09 9:45 AM | Russ Cam | 11/8/09 6:10 PM | UK | NULL | Interested in C#, ASP.NET, S |

© Brent Ozar Unlimited®. All rights reserved.

# StackOverflow2010.mdf: a bunch of 8KB pages

Slide 7
Slides, scripts, db: BrentOzar.com/go/engine

# A page is the smallest unit of data.

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.ir |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/30/10 2:30 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 2/18/10 9:27 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 3/10/10 9:56 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 1/25/10 7:10 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/19/10 10:54 PM | Sweden | 36 | I work on the Blockit .NM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 3/12/10 9:44 AM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 12/1/09 4:37 PM | NL | 27 | I work as a software develo |
| 540 | 101 | 7/15/09 9:37 AM | DanSingerman | 4/1/10 6:37 AM | NULL | NULL | <p>Sufficiently advanced st |
| 541 | 647 | 7/15/09 9:37 AM | Alexis Hirst | 4/1/10 10:49 PM | England | 21 | NULL |
| 542 | 5921 | 7/15/09 9:38 AM | hyperslug | 12/12/09 5:12 PM | NULL | NULL | <img src="http://www.hyp |
| 543 | 490 | 7/15/09 9:38 AM | DavidWhitney | 3/28/10 11:12 PM | London, UK | 25 | Primarily C# .NET developer |
| 544 | 101 | 7/15/09 9:39 AM | Ben Laan | 4/1/10 3:34 PM | Adelaide, Australia | 34 | C#, Delphi, T-SQL : Archite |
| 545 | 985 | 7/15/09 9:40 AM | redacted | 8/26/09 9:59 AM | NaN | 23 | NaN |
| 546 | 101 | 7/15/09 9:41 AM | hbec | 4/1/10 8:24 AM | Bombay | 27 | System.StackOverflowExce |
| 547 | 404 | 7/15/09 9:41 AM | Ant | 1/28/10 5:18 PM | England | 27 | C# developer since 2005. |
| 548 | 501 | 7/15/09 9:41 AM | Jeremy French | 2/26/10 5:53 PM | UK | 33 | Seasoned web developer. W |
| 549 | 173 | 7/15/09 9:42 AM | Joshua | 3/16/10 4:42 PM | Orlando, FL | 28 | Software Engineer. Have wo |
| 550 | 61 | 7/15/09 9:42 AM | hojnasa | 4/1/10 10:39 AM | NULL | NULL | NULL |
| 551 | 341 | 7/15/09 9:43 AM | Tom | 11/29/09 7:20 AM | Worcester, UK | NULL | NULL |
| 552 | 1 | 7/15/09 9:44 AM | Cecelia | 2/7/10 10:28 AM | Maracaibo, Venezuela | 25 | NULL |
| 553 | 101 | 7/15/09 9:44 AM | mcff | 4/1/10 8:43 PM | Sweden | NULL | NULL |
| 554 | 1158 | 7/15/09 9:44 AM | becomingGuru | 4/1/10 6:11 PM | Bangalore / Hyderabad | 25 | curious learner! |
| 555 | 1016 | 7/15/09 9:45 AM | MiffTheFox | 11/8/09 6:10 PM | Huntsville, AL | NULL | <p><a href="http://forums |
| 556 | 101 | 7/15/09 9:45 AM | Russ Cam | | UK | NULL | Interested in C#, ASP.NET, r |

If we need to read a row:

- Figure out which page has it
- Get that page up in memory

To write a row:

- Get that page up in memory
- Make our change
- Write the page to disk

Slide 8
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## In a perfect world

Your query is easy to understand

The data you want is only on one page

SQL Server knows exactly which page it is

SQL Server has that page cached,
and can jump to exactly that page

SQL Server can read the data out as-is,
without doing any additional manipulation

Slide 9
Slides, scripts, db: BrentOzar.com/go/engine

## In a perfect world...

Slide 10
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## But often in our world

Your query is **not** easy to understand

The data you want is **spread across many pages**

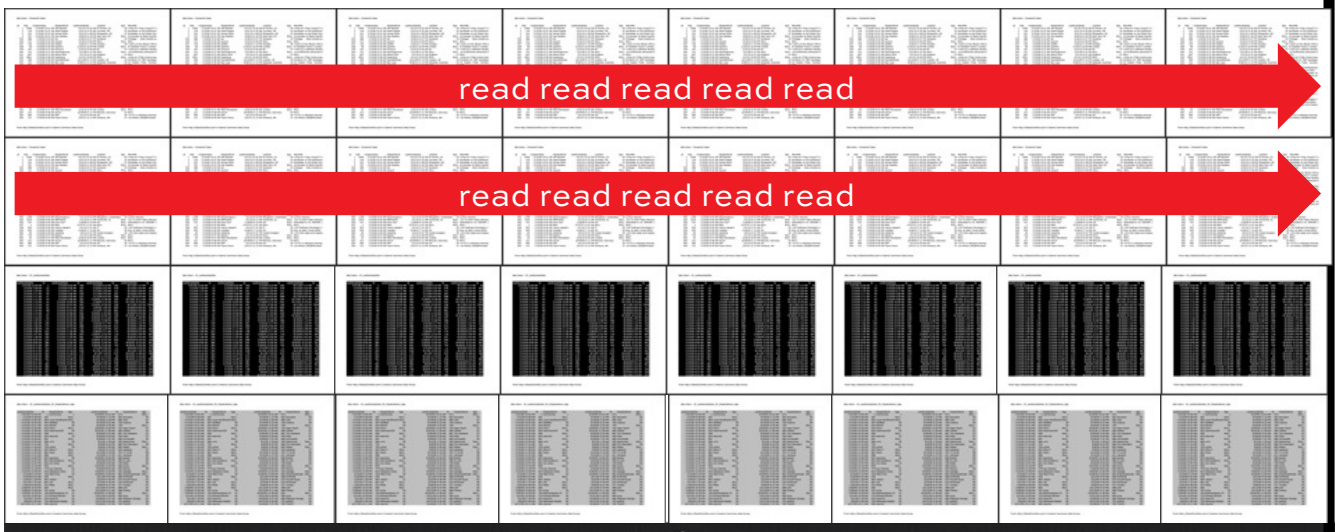
SQL Server **doesn't know exactly which** pages

SQL Server **doesn't have** that page cached,
and has to **read the data from storage**

SQL Server **has to do a bunch of work** on the data,
like summing, grouping, joining, and ordering

Slide 11
Slides, scripts, db: BrentOzar.com/go/engine

## In that world, we read 'em all




Slide 12
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## How to Think Like the Engine

Guess which row(s) the query wants

Guess which page(s) have those rows

Guess what will be the most efficient way to get 'em: often there are multiple copies of rows, aka indexes

Guess which tables should be processed first

Guess how much resources we should allocate

In this class, we'll keep it simple, but this goes deep.

Slide 13
Slides, scripts, db: BrentOzar.com/go/engine

It's a fun role playing game for the bedroom, too

# You: SQL Server.
# Me: end user.

Slide 14
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## First query:

```
SELECT Id
FROM dbo.Users
```

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.in |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/31/10 3:48 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 3/30/10 2:30 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 2/18/10 9:27 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 3/10/10 9:56 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/25/10 7:10 PM | Sweden | 36 | I work on the JRockit JVM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 1/19/10 10:54 PM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 3/12/10 9:44 AM | NL | 27 | I work as a software develo |
| 540 | 101 | 7/15/09 9:37 AM | DanSingerman | 12/1/09 4:37 PM | NULL | NULL | <p>Sufficiently advanced st |

## Your execution plan:

1. Shuffle through all of the pages, saying the Id of each record out loud.

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.in |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/31/10 3:48 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 3/30/10 2:30 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 2/18/10 9:27 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 3/10/10 9:56 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/25/10 7:10 PM | Sweden | 36 | I work on the JRockit JVM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 1/19/10 10:54 PM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 3/12/10 9:44 AM | NL | 27 | I work as a software develo |
| 540 | 101 | 7/15/09 9:37 AM | DanSingerman | 12/1/09 4:37 PM | NULL | NULL | <p>Sufficiently advanced st |

© Brent Ozar Unlimited®. All rights reserved.

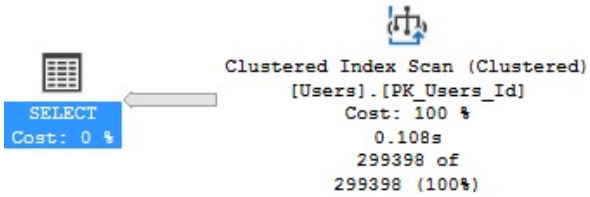
# SQL Server's execution plan

```
SELECT Id
FROM dbo.Users;
GO
```

150 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT Id FROM dbo.Users
```




Slide 17
Slides, scripts, db: BrentOzar.com/go/engine

# SET STATISTICS IO ON

Logical reads: the number of 8K pages we read.
(7,405 x 8KB = 59MB)

```
SET STATISTICS IO ON;
GO
SELECT Id
FROM dbo.Users;
GO
```

150 %

Results | Messages | Execution plan

```
(299398 rows affected)
Table 'Users'. Scan count 1, logical reads 7405,
physical reads 0, read-ahead reads 0,
lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.
```

Slide 18
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# That’s 15 reams.

Slide 19
Slides, scripts, db: BrentOzar.com/go/engine

## Let’s add a filter.

```
SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
```

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.in |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/31/10 3:48 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 3/30/10 2:30 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 2/18/10 9:27 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 3/10/10 9:56 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/25/10 7:10 PM | Sweden | 36 | I work on the JRockit JVM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 1/19/10 10:54 PM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 3/12/10 9:44 AM | NL | 27 | I work as a software develo |
| 540 | 101 | 7/15/09 9:37 AM | DanSingerman | 12/1/09 4:37 PM | NULL | NULL | <p>Sufficiently advanced et |

© Brent Ozar Unlimited®. All rights reserved.

# Your execution plan:

1. Shuffle through all of the pages, saying the Id of each record out loud, if their LastAccessDate > '2014/07/01'.

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.in |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/31/10 3:48 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 3/30/10 2:30 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 2/18/10 9:27 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 3/10/10 9:56 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/25/10 7:10 PM | Sweden | 36 | I work on the JRockit JVM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 1/19/10 10:54 PM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 3/12/10 9:44 AM | NL | 27 | I work as a software develo |

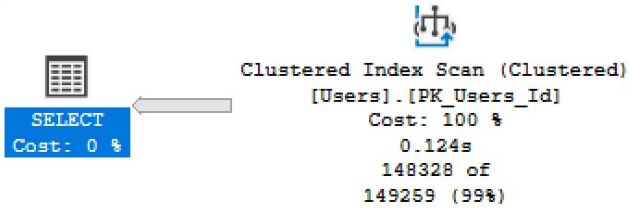
# SQL Server's execution plan

```
SELECT Id FROM dbo.Users WHERE LastAccessDate > '2014/07/01';
```

150 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1
```






© Brent Ozar Unlimited®. All rights reserved.

```
SET STATISTICS IO ON;

SELECT Id
FROM dbo.Users;

SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01';
```

100 %

Results | Messages | Execution plan

```
(299398 rows affected)
Table 'Users'. Scan count 1, logical reads 7405, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

(1 row affected)

(148328 rows affected)
Table 'Users'. Scan count 5, logical reads 7778, physical reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob read-ahead reads 0.

(1 row affected)
```

Slide 23
Slides, scripts, db: BrentOzar.com/go/engine

# Lesson:
# Using WHERE without a matching index means scanning all the data.

(And there are some extra reads when queries go parallel – but more on that in my more advanced classes.)

Slide 24
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

```
SET STATISTICS IO ON;

SELECT Id
FROM dbo.Users;

SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01';
```

100 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 50%
SELECT Id FROM dbo.Users
```

SELECT
Cost: 0 %

Clustered Index Scan (Clustered)
[Users].[PK_Users_Id]
Cost: 100 %
0.315s
299398 of
299398 (100%)

```
Query 2: Query cost (relative to the batch): 50%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1
Missing Index (Impact 94.5896): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] ON [dbo].[Users] ([LastAccessDate])
```

SELECT
Cost: 0 %

Parallelism
(Gather Streams)
Cost: 2 %
0.009s
148328 of
149259 (99%)

Clustered Index Scan (Clustered)
[Users].[PK_Users_Id]
Cost: 98 %
0.016s
148328 of
149259 (99%)

```
Query 1: Query cost (relative to the batch): 50%
SELECT Id FROM dbo.Users
```

SELECT
Cost:

Clustered Index Scan (Clustered)
[Users].[PK_Users_Id]
Cost: 100 %

**SELECT**

| | |
|---|---|
| Cached plan size | 16 KB |
| Estimated Operator Cost | 0 (0%) |
| Degree of Parallelism | 1 |
| Estimated Subtree Cost | 5.79558 |
| Estimated Number of Rows | 299398 |

**Statement**
SELECT Id
FROM dbo.Users

```
Query 2: Query cost (relative to the batch): 50%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1
Missing Index (Impact 94.5896): CREATE NONCLUSTERED INDEX [<Name
```

SELECT
Cost: 0 %

Parallelism
(Gather Streams)
Cost: 2 %
0.009s

Clustered Index Scan (Clustered)
[Users].[PK_Users_Id]
Cost: 98 %
0.016s
148328 of
149259 (99%)

**SELECT**

| | |
|---|---|
| Cached plan size | 24 KB |
| Estimated Operator Cost | 0 (0%) |
| Degree of Parallelism | 4 |
| Estimated Subtree Cost | 5.6828 |
| Memory Grant | 72 |
| Estimated Number of Rows | 149259 |

**Statement**
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1

Slide 26
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

Lesson:
Estimated Subtree Cost is a rough measure of CPU and IO work required for a query.



https://BrentOzar.com/go/querybucks

© Brent Ozar Unlimited®. All rights reserved.

## Let's add a sort.

```
SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate
```

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.in |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/31/10 3:48 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 3/30/10 2:30 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 2/18/10 9:27 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 3/10/10 9:56 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/25/10 7:10 PM | Sweden | 36 | I work on the JRockit JVM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 1/19/10 10:54 PM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 3/12/10 9:44 AM | NL | 27 | I work as a software develo |

## Your execution plan

1. Shuffle through all of the pages,
   **writing down fields** _____ for each record,
   if their LastAccessDate > '2014/07/01'.
2. **Sort the matching records by LastAccessDate.**

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.in |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/31/10 3:48 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 3/30/10 2:30 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 2/18/10 9:27 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 3/10/10 9:56 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/25/10 7:10 PM | Sweden | 36 | I work on the JRockit JVM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 1/19/10 10:54 PM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 3/12/10 9:44 AM | NL | 27 | I work as a software develo |

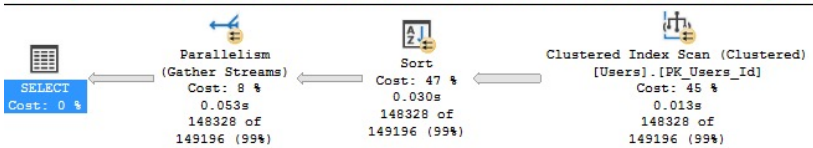
© Brent Ozar Unlimited®. All rights reserved.

# SQL Server's execution plan

```
SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO
```

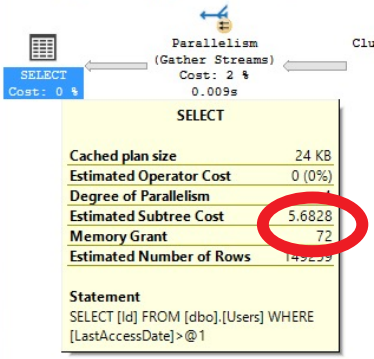
Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1 ORDER BY [LastAccessDate] ASC
Missing Index (Impact 43.6533): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] ON [dbo].[Users] ([LastAccessDate])
```

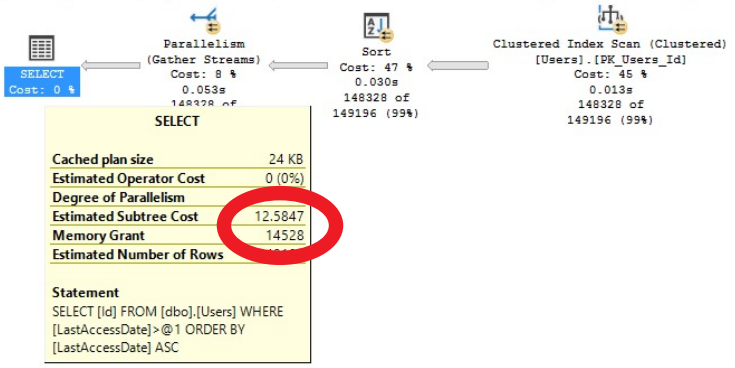



Slide 31
Slides, scripts, db: BrentOzar.com/go/engine

```
Query 2: Query cost (relative to the batch): 50%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1
Missing Index (Impact 94.5896): CREATE NONCLUSTERED INDEX [<Name
```




```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1 ORDER BY [LastAccessDa
Missing Index (Impact 43.6533): CREATE NONCLUSTERED INDEX [<Name of Missing Inc
```



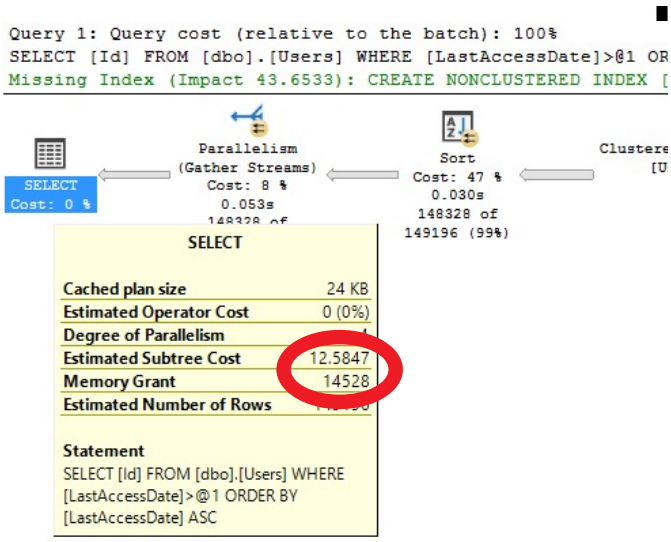

Slide 32
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Order By:

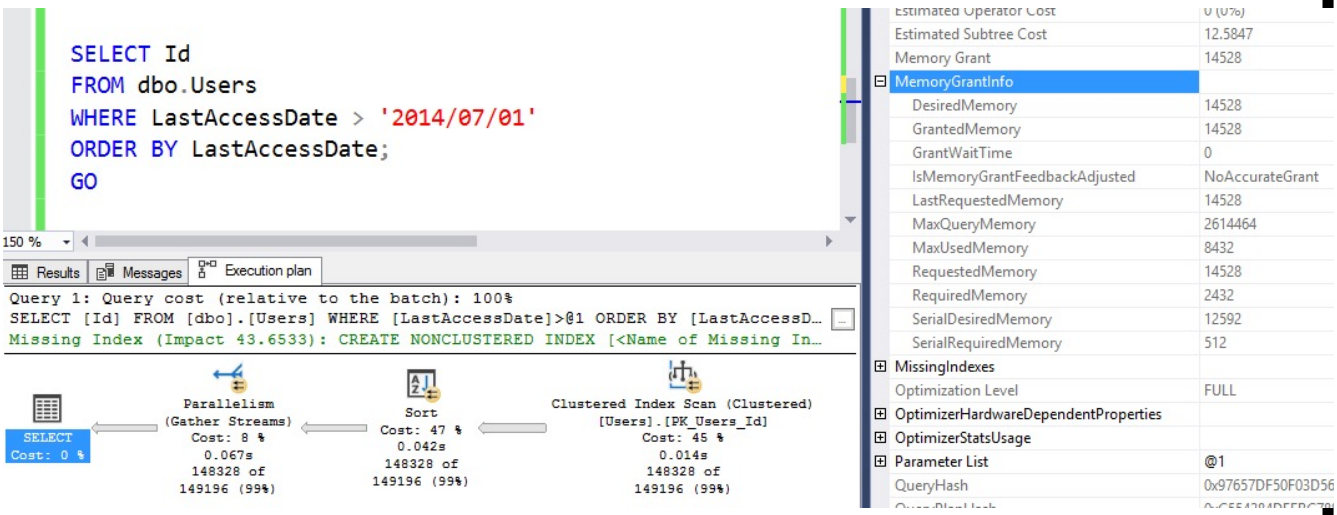
Cost is up about 2x

We needed space to write down our results, so we got a memory grant




Slide 33
Slides, scripts, db: BrentOzar.com/go/engine

# You can see more in Properties




Slide 34
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# You can't always get what you want.

| | |
|---|---|
| Memory Grant | 913768 |
| ◿ MemoryGrantInfo | |
| DesiredMemory | 5843424 |
| GrantedMemory | 913768 |
| GrantWaitTime | 0 |
| MaxUsedMemory | 913760 |
| RequestedMemory | 913768 |
| RequiredMemory | 1264 |
| SerialDesiredMemory | 5842672 |
| SerialRequiredMemory | 512 |

* - This screenshot is from a different query to show variances.

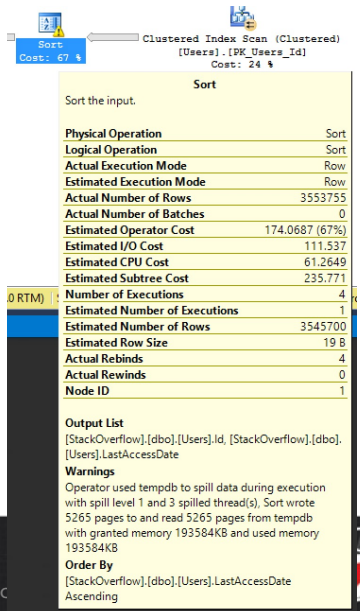
Memory is set when the query starts, and not revised.

SQL Server has to assume other people will run queries at the same time as you.

Your memory grant can change with each time that you run a query.

Slide 35
Slides, scripts, db: BrentOzar.com/go/engine

# And if you run out of memory...




**Warnings**
Operator used tempdb to spill data during execution with spill level 1 and 3 spilled thread(s), Sort wrote 5265 pages to and read 5265 pages from tempdb with granted memory 193584KB and used memory 193584KB
**Order By**
[StackOverflow].[dbo].[Users].LastAccessDate Ascending

Slide 36
Slides, scripts, db: BrentOzar.c

© Brent Ozar Unlimited®. All rights reserved.

## Let's get all the fields.

```
SELECT *
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate
```

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.in |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/31/10 3:48 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 3/30/10 2:30 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 2/18/10 9:27 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 3/10/10 9:56 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/25/10 7:10 PM | Sweden | 36 | I work on the JRockit JVM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 1/19/10 10:54 PM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 3/12/10 9:44 AM | NL | 27 | I work as a software develo |

## Your execution plan

1. Shuffle through all of the pages,
   **writing down fields** _____ for each record,
   if their LastAccessDate > '2014/07/01'.
2. **Sort the matching records by LastAccessDate.**

dbo.Users - Clustered Index

| Id | Rep | CreationDate | DisplayName | LastAccessDate | Location | Age | AboutMe |
|---|---|---|---|---|---|---|---|
| 1 | 2406 | 7/12/09 10:51 PM | Jeff Atwood | 4/1/10 10:35 AM | El Cerrito, CA | 39 | <img src="http://img377.in |
| 2 | 126 | 7/12/09 10:51 PM | Geoff Dalgas | 3/31/10 4:35 AM | Corvallis, OR | 32 | Developer on the StackOver |
| 3 | 101 | 7/12/09 10:51 PM | Jarrod Dixon | 3/31/10 3:48 PM | Morganton, NC | 31 | Developer on the Stack Ove |
| 4 | 767 | 7/12/09 10:51 PM | Joel Spolsky | 3/30/10 2:30 PM | New York, NY | NULL | Co-founder of Stack Overflo |
| 535 | 386 | 7/15/09 9:33 AM | izb | 2/18/10 9:27 PM | Scotland | 33 | Twittage: http://twitter.co |
| 536 | 101 | 7/15/09 9:34 AM | second | 3/10/10 9:56 PM | NULL | NULL | NULL |
| 537 | 120 | 7/15/09 9:35 AM | staffan | 1/25/10 7:10 PM | Sweden | 36 | I work on the JRockit JVM d |
| 538 | 90 | 7/15/09 9:35 AM | cgreeno | 1/19/10 10:54 PM | London | NULL | A Canadian living in London |
| 539 | 167 | 7/15/09 9:37 AM | Arcturus | 3/12/10 9:44 AM | NL | 27 | I work as a software develo |

© Brent Ozar Unlimited®. All rights reserved.

# That query sucks.

Slide 39
Slides, scripts, db: BrentOzar.com/go/engine

## But why does it suck?

Do we work harder to read the data?

Do we work harder to write the data?

Do we work harder to sort the data?

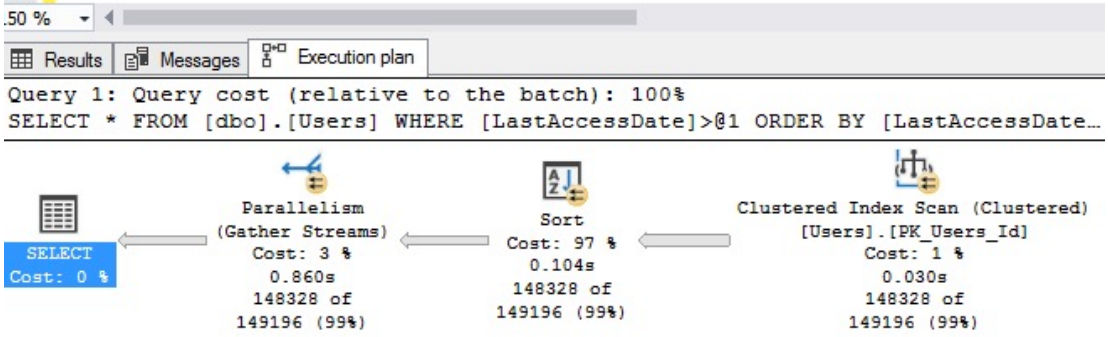
Do we work harder to output the data?

Slide 40
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

```sql
SELECT *
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO
```

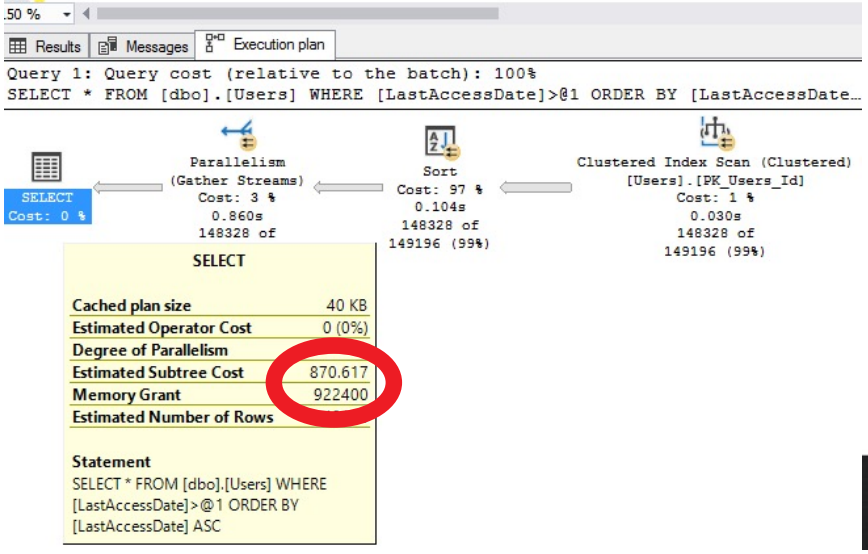
# The sort cost is now 97%...




Slide 41
Slides, scripts, db: BrentOzar.com/go/engine

```sql
SELECT *
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO
```

# Of a MUCH larger overall cost.




© Brent Ozar Unlimited®. All rights reserved.

| | SELECT ID | SELECT * |
|---|---|---|
| No order | 6 | 6 |
| ORDER BY | 13 | 871 |

Slide 43
Slides, scripts, db: BrentOzar.com/go/engine

Lesson:
Sorting data is expensive, and more fields make it worse.

Slide 44
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## Let's run it a few times.

```
SELECT *
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO 100
```

Slide 45
Slides, scripts, db: BrentOzar.com/go/engine

## Your execution plan

1. Shuffle through all of the pages, **writing down all the fields** for each record, if their LastAccessDate > '2014/07/01'.
2. Sort the matching records by LastAccessDate.
3. **Keep the output so you could reuse it the next time you saw this same query?**

Slide 46
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

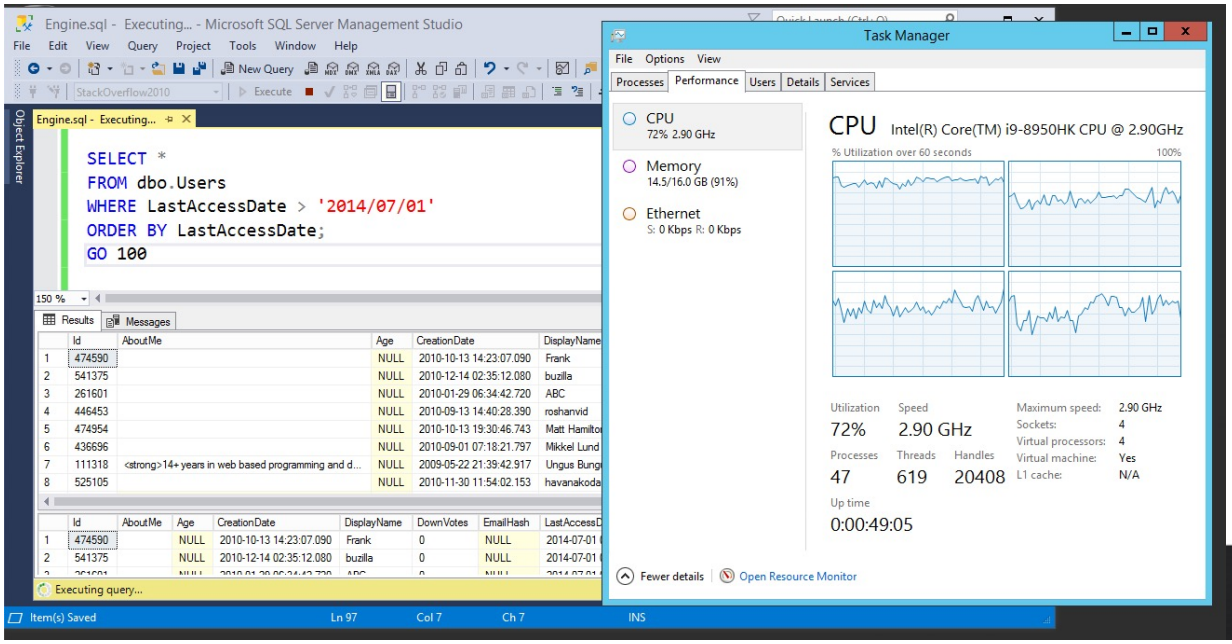
# Oracle can.

(One of the reasons it costs $47,000 per core.)

Slide 47
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# SQL Server reads & sorts 100 times.




Lesson:
SQL Server caches data pages,
not query output.

So how do we
make queries faster?



© Brent Ozar Unlimited®. All rights reserved.

# Recap (so far)

Slide 51
Slides, scripts, db: BrentOzar.com/go/engine

## What you learned so far

SET STATISTICS IO ON: shows # of 8KB pages read

SET STATISTICS TIME ON: shows CPU work done

WHERE without a supporting index: table scan

ORDER BY without a supporting index: CPU work

SQL Server caches pages, not query results

Slide 52
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## What you'll learn next

Nonclustered indexes, the good & bad

The difference between seeks & scans

Slide 53
Slides, scripts, db: BrentOzar.com/go/engine

BRENT OZAR
UNLIMITED®

# How to Think Like the SQL Server Engine

## Part 2: Nonclustered Indexes

Slide 54
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Nonclustered indexes: copies.

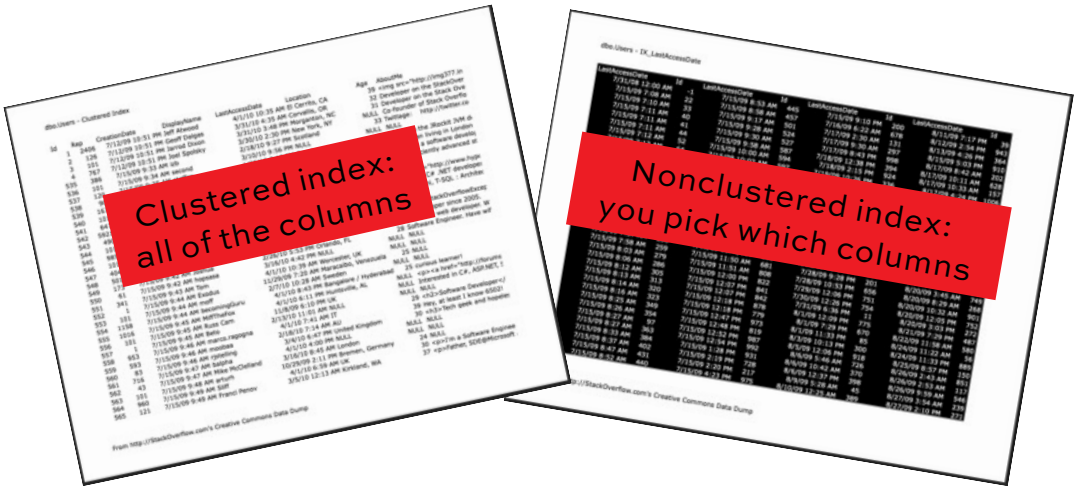
Stored in order we want, include the fields we want

```
CREATE INDEX
IX_LastAccessDate_Id
ON dbo.Users(LastAccessDate, Id)
```

dbo.Users - IX_LastAccessDate

| LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | 7/15/09 8:53 AM | 445 | 7/15/09 9:10 PM | 200 | 8/11/09 7:17 PM | 39 |
| 7/15/09 7:08 AM | 22 | 7/15/09 8:58 AM | 457 | 7/16/09 6:22 AM | 678 | 8/12/09 2:54 PM | 943 |
| 7/15/09 7:10 AM | 33 | 7/15/09 9:17 AM | 501 | 7/17/09 2:30 AM | 131 | 8/13/09 4:26 PM | 364 |
| 7/15/09 7:11 AM | 40 | 7/15/09 9:28 AM | 524 | 7/17/09 9:30 AM | 297 | 8/15/09 5:03 PM | 910 |
| 7/15/09 7:11 AM | 41 | 7/15/09 9:30 AM | 527 | 7/17/09 8:43 PM | 998 | 8/17/09 8:42 AM | 202 |
| 7/15/09 7:11 AM | 44 | 7/15/09 9:58 AM | 587 | 7/18/09 12:38 PM | 394 | 8/17/09 10:11 AM | 628 |
| 7/15/09 7:12 AM | 52 | 7/15/09 10:00 AM | 594 | 7/18/09 2:15 PM | 924 | 8/17/09 10:33 AM | 157 |
| 7/15/09 7:13 AM | 64 | 7/15/09 10:02 AM | 597 | 7/19/09 10:26 PM | 336 | 8/17/09 4:24 PM | 1006 |

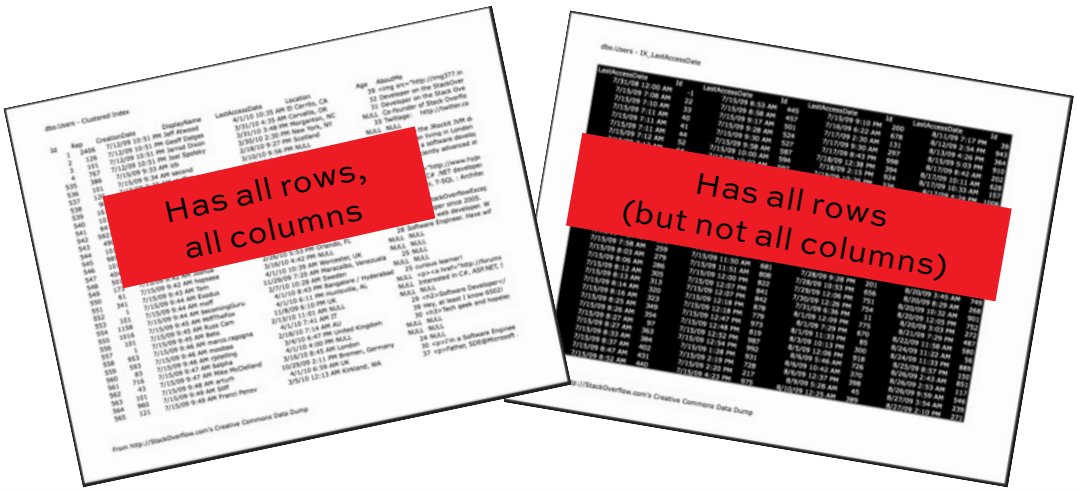
# Literally a copy of the table.





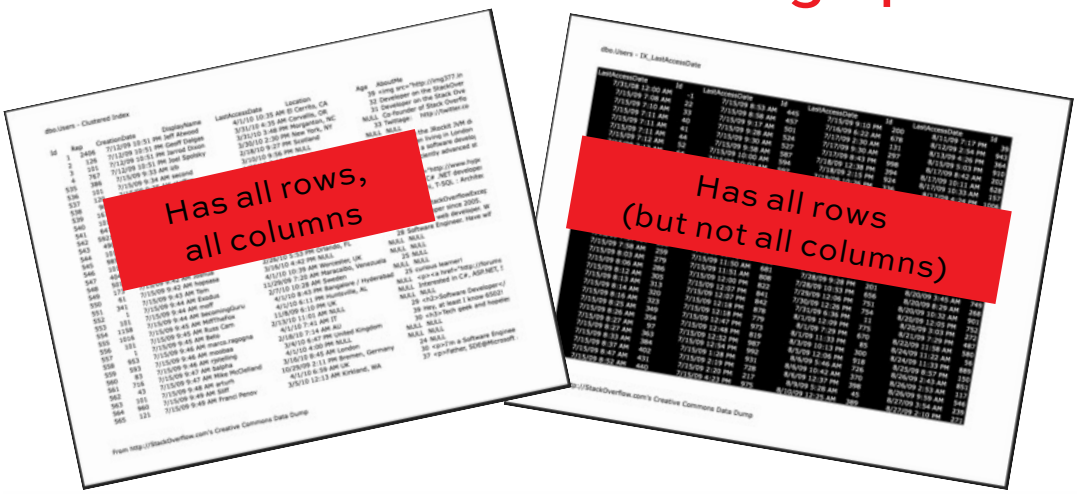

© Brent Ozar Unlimited®. All rights reserved.

# Deletes and inserts now have to do twice the writes.




Slide 57
Slides, scripts, db: BrentOzar.com/go/engine

# Updates only affect copies that have the column being updated.




Slide 58
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## Say we run this update query:

```
UPDATE dbo.Users
SET Age = Age + 1
WHERE DisplayName = 'Brent Ozar’
```

Age isn’t on this index, so we don’t have to touch it:

dbo.Users - IX_LastAccessDate

| LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | 7/15/09 8:53 AM | 445 | 7/15/09 9:10 PM | 200 | 8/11/09 7:17 PM | 39 |
| 7/15/09 7:08 AM | 22 | 7/15/09 8:58 AM | 457 | 7/16/09 6:22 AM | 678 | 8/12/09 2:54 PM | 943 |
| 7/15/09 7:10 AM | 33 | 7/15/09 9:17 AM | 501 | 7/17/09 2:30 AM | 131 | 8/13/09 4:26 PM | 364 |
| 7/15/09 7:11 AM | 40 | 7/15/09 9:28 AM | 524 | 7/17/09 9:30 AM | 297 | 8/15/09 5:03 PM | 910 |
| 7/15/09 7:11 AM | 41 | 7/15/09 9:30 AM | 527 | 7/17/09 8:43 PM | 998 | 8/17/09 8:42 AM | 202 |
| 7/15/09 7:11 AM | 44 | 7/15/09 9:58 AM | 587 | 7/18/09 12:38 PM | 394 | 8/17/09 10:11 AM | 628 |
| 7/15/09 7:12 AM | 52 | 7/15/09 10:00 AM | 594 | 7/18/09 2:15 PM | 924 | 8/17/09 10:33 AM | 157 |
| 7/15/09 7:13 AM | 64 | 7/15/09 10:02 AM | 597 | 7/19/09 10:26 PM | 336 | 8/17/09 4:24 PM | 1006 |

## In theory, don’t index hot columns.

The more frequently a column changes,
the more we’ll have to write to storage.

In theory, we make people go to the clustered
index if they want a hot column.

In practice, we may have to make compromises.

The faster your storage is,
and the less writes you’re doing,
the easier it is to make those compromises.

Slide 60
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## In theory, less indexes are good.

You want as few indexes as possible
in order to support your workload.

No indexes:
every SELECT scans tables.

Too many indexes:
D/U/I queries do too many writes.

Learn more: Fundamentals of Index Tuning

Slide 61
Slides, scripts, db: BrentOzar.com/go/engine

## Back to our SELECT query.

```
SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
```

dbo.Users - IX_LastAccessDate

| LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | 7/15/09 8:53 AM | 445 | 7/15/09 9:10 PM | 200 | 8/11/09 7:17 PM | 39 |
| 7/15/09 7:08 AM | 22 | 7/15/09 8:58 AM | 457 | 7/16/09 6:22 AM | 678 | 8/12/09 2:54 PM | 943 |
| 7/15/09 7:10 AM | 33 | 7/15/09 9:17 AM | 501 | 7/17/09 2:30 AM | 131 | 8/13/09 4:26 PM | 364 |
| 7/15/09 7:11 AM | 40 | 7/15/09 9:28 AM | 524 | 7/17/09 9:30 AM | 297 | 8/15/09 5:03 PM | 910 |
| 7/15/09 7:11 AM | 41 | 7/15/09 9:30 AM | 527 | 7/17/09 8:43 PM | 998 | 8/17/09 8:42 AM | 202 |
| 7/15/09 7:11 AM | 44 | 7/15/09 9:58 AM | 587 | 7/18/09 12:38 PM | 394 | 8/17/09 10:11 AM | 628 |
| 7/15/09 7:12 AM | 52 | 7/15/09 10:00 AM | 594 | 7/18/09 2:15 PM | 924 | 8/17/09 10:33 AM | 157 |
| 7/15/09 7:13 AM | 64 | 7/15/09 10:02 AM | 597 | 7/19/09 10:26 PM | 336 | 8/17/09 4:24 PM | 1006 |

© Brent Ozar Unlimited®. All rights reserved.

# Your execution plan

1. Grab IX_LastAccessDate and seek to 2014/07/01.
2. Read the Id's out in order.

dbo.Users - IX_LastAccessDate

| LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | 7/15/09 8:53 AM | 445 | 7/15/09 9:10 PM | 200 | 8/11/09 7:17 PM | 39 |
| 7/15/09 7:08 AM | 22 | 7/15/09 8:58 AM | 457 | 7/16/09 6:22 AM | 678 | 8/12/09 2:54 PM | 943 |
| 7/15/09 7:10 AM | 33 | 7/15/09 9:17 AM | 501 | 7/17/09 2:30 AM | 131 | 8/13/09 4:26 PM | 364 |
| 7/15/09 7:11 AM | 40 | 7/15/09 9:28 AM | 524 | 7/17/09 9:30 AM | 297 | 8/15/09 5:03 PM | 910 |
| 7/15/09 7:11 AM | 41 | 7/15/09 9:30 AM | 527 | 7/17/09 8:43 PM | 998 | 8/17/09 8:42 AM | 202 |
| 7/15/09 7:11 AM | 44 | 7/15/09 9:58 AM | 587 | 7/18/09 12:38 PM | 394 | 8/17/09 10:11 AM | 628 |
| 7/15/09 7:12 AM | 52 | 7/15/09 10:00 AM | 594 | 7/18/09 2:15 PM | 924 | 8/17/09 10:33 AM | 157 |
| 7/15/09 7:13 AM | 64 | 7/15/09 10:02 AM | 597 | 7/19/09 10:26 PM | 336 | 8/17/09 4:24 PM | 1006 |

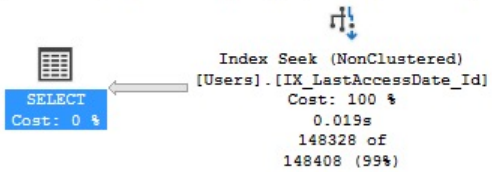
# The new plan uses the index

```
SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO
```

150 %

Results | Messages | Execution plan

Query 1: Query cost (relative to the batch): 100%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1 ORDER BY [LastAccessDate] ASC




Slide 64
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

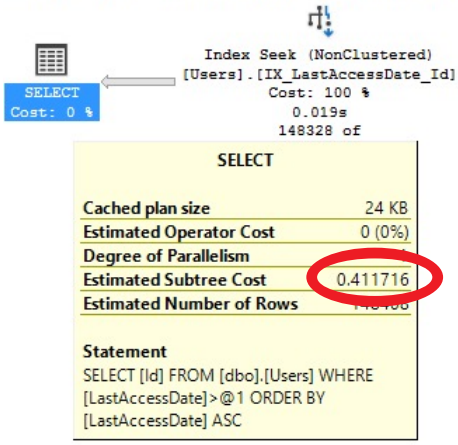
```sql
SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO
```

# And it's CHEAP.

150 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1 ORDER BY [LastAccessDate] ASC
```




| | SELECT ID | SELECT * |
|---|---|---|
| No order | 6 | 6 |
| ORDER BY | 13 | 871 |
| ORDER BY, with index | <1 | 48 |



© Brent Ozar Unlimited®. All rights reserved.

```sql
/* Back to back comparison of indexes */
SELECT Id
FROM dbo.Users WITH (INDEX = 1)
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;

SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO
```

150 %

Results | Messages | Execution plan

```
(148328 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, p
Table 'Users'. Scan count 5, logical reads 7778, p

(1 row affected)

(148328 rows affected)
Table 'Users'. Scan count 1, logical reads 335, ph

(1 row affected)
```

# Why cheaper?

For starters, it does less logical reads...

```sql
SET STATISTICS TIME, IO ON;
/* Back to back comparison of indexes */
SELECT Id
FROM dbo.Users WITH (INDEX = 1)
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;

SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO
```

50 %

Results | Messages

```
SQL Server parse and compile time:
   CPU time = 0 ms, elapsed time = 1 ms.

(148328 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physi
Table 'Users'. Scan count 5, logical reads 7778, physic

 SQL Server Execution Times:
   CPU time = 189 ms, elapsed time = 367 ms.
SQL Server parse and compile time:
   CPU time = 0 ms, elapsed time = 0 ms.

(148328 rows affected)
Table 'Users'. Scan count 1, logical reads 335, physica

 SQL Server Execution Times:
   CPU time = 63 ms, elapsed time = 429 ms.
```

# And less CPU, too.

SET STATISTICS TIME shows you how much CPU time each query burned up.

The index eliminates the sort, which burned up our CPUs.

© Brent Ozar Unlimited®. All rights reserved.

## The index covers the fields needed by the query, so we call it a covering index.*

*But covering isn't really a special kind of index – it's only covering when we're talking about a query.

Slide 69
Slides, scripts, db: BrentOzar.com/go/engine

## So index seeks are great, right?

Slide 70
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

```
SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO
```

150 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1 ORDER BY [LastAccessDate] ASC
```

SELECT
Cost: 0 %

Index Seek (NonClustered)
[Users].[IX_LastAccessDate_Id]
Cost: 100 %

"Seek" sounds small, right?

| | |
|---|---|
| Actual Number of Rows | 148328 |
| Number of Rows Read | 148328 |
| Estimated Number of Rows | 148408 |
| Estimated Row Size | 19 B |
| Estimated Data Size | 2754 KB |

But that's a lot of data.

**We think "seek" means,**
"I'm going to jump to a row and read that one row."

**We think "scan" means,**
"I'm going to read the whole thing."



© Brent Ozar Unlimited®. All rights reserved.

```
SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '1800/01/01'
ORDER BY LastAccessDate;
GO
```

Note that date

50 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id] FROM [dbo].[Users] WHERE [LastAccessDate]>@1 ORDER BY [LastAccessDate] ASC
```

SELECT
Cost: 0 %

Index Seek (NonClustered)
[Users].[IX_LastAccessDate_Id]
Cost: 100 %

"Seek" = read all rows

| | |
|---|---|
| Actual Number of Rows | 299398 |
| Number of Rows Read | 299398 |
| Estimated Number of Rows | 148408 |
| Estimated Row Size | 19 B |
| Estimated Data Size | 2754 KB |

# SQL Server doesn't know.

You and I know this means the whole table:

```
WHERE LastAccessDate > '1800/01/01'
```

But SQL Server doesn't, and can't guarantee it unless you tell it more about the data in the table, like add a constraint.



© Brent Ozar Unlimited®. All rights reserved.

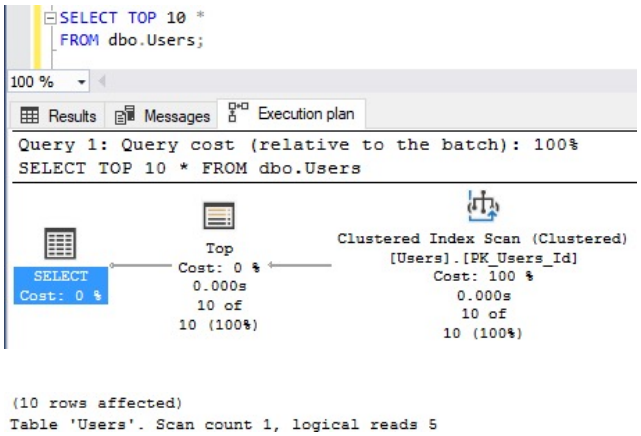
**Seek means,**
"I'm going to jump to a row and start reading."

**Scan means,**
"I'm going to start at either end of the object
(might be either the start, or the end)
and start reading."

Neither term defines
how many rows will be read.



# A scan that reads a few rows:






© Brent Ozar Unlimited®. All rights reserved.

## Seeks vs scans

A seek can start at the first row,
and read the entire table.

A scan can start at one end of the table,
and only read a few pages.

We can't just say, "All index seeks! We're done."

Slide 77
Slides, scripts, db: BrentOzar.com/go/engine

# Recap (so far)

Slide 78
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## What you learned so far

Indexes: literally a copy of parts of the table

- Good: indexes reduce page reads, CPU for sorts
- Bad: indexes slow down D/U/I work

Seek means jump to one area and start reading

Scan means start at either end of the index

Neither seek nor scan refers to how much we'll read

Slide 79
Slides, scripts, db: BrentOzar.com/go/engine

## What you'll learn next

Key lookups

The tipping point

How the engine chooses between plans

Cost-based query optimization

How statistics come into play

How your query syntax affects all this

Slide 80
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

BRENT OZAR
UNLIMITED®

# How to Think Like the SQL Server Engine

## Part 3: How the Engine Chooses Between Plans

Slide 81
Slides, scripts, db: BrentOzar.com/go/engine

## Where we left off

```
SELECT Id
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
```

dbo.Users - IX_LastAccessDate

| LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | 7/15/09 8:53 AM | 445 | 7/15/09 9:10 PM | 200 | 8/11/09 7:17 PM | 39 |
| 7/15/09 7:08 AM | 22 | 7/15/09 8:58 AM | 457 | 7/16/09 6:22 AM | 678 | 8/12/09 2:54 PM | 943 |
| 7/15/09 7:10 AM | 33 | 7/15/09 9:17 AM | 501 | 7/17/09 2:30 AM | 131 | 8/13/09 4:26 PM | 364 |
| 7/15/09 7:11 AM | 40 | 7/15/09 9:28 AM | 524 | 7/17/09 9:30 AM | 297 | 8/15/09 5:03 PM | 910 |
| 7/15/09 7:11 AM | 41 | 7/15/09 9:30 AM | 527 | 7/17/09 8:43 PM | 998 | 8/17/09 8:42 AM | 202 |
| 7/15/09 7:11 AM | 44 | 7/15/09 9:58 AM | 587 | 7/18/09 12:38 PM | 394 | 8/17/09 10:11 AM | 628 |
| 7/15/09 7:12 AM | 52 | 7/15/09 10:00 AM | 594 | 7/18/09 2:15 PM | 924 | 8/17/09 10:33 AM | 157 |
| 7/15/09 7:13 AM | 64 | 7/15/09 10:02 AM | 597 | 7/19/09 10:26 PM | 336 | 8/17/09 4:24 PM | 1006 |

© Brent Ozar Unlimited®. All rights reserved.

## Let's add a couple of columns.

```
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
```

dbo.Users - IX_LastAccessDate

| LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | 7/15/09 8:53 AM | 445 | 7/15/09 9:10 PM | 200 | 8/11/09 7:17 PM | 39 |
| 7/15/09 7:08 AM | 22 | 7/15/09 8:58 AM | 457 | 7/16/09 6:22 AM | 678 | 8/12/09 2:54 PM | 943 |
| 7/15/09 7:10 AM | 33 | 7/15/09 9:17 AM | 501 | 7/17/09 2:30 AM | 131 | 8/13/09 4:26 PM | 364 |
| 7/15/09 7:11 AM | 40 | 7/15/09 9:28 AM | 524 | 7/17/09 9:30 AM | 297 | 8/15/09 5:03 PM | 910 |
| 7/15/09 7:11 AM | 41 | 7/15/09 9:30 AM | 527 | 7/17/09 8:43 PM | 998 | 8/17/09 8:42 AM | 202 |
| 7/15/09 7:11 AM | 44 | 7/15/09 9:58 AM | 587 | 7/18/09 12:38 PM | 394 | 8/17/09 10:11 AM | 628 |
| 7/15/09 7:12 AM | 52 | 7/15/09 10:00 AM | 594 | 7/18/09 2:15 PM | 924 | 8/17/09 10:33 AM | 157 |
| 7/15/09 7:13 AM | 64 | 7/15/09 10:02 AM | 597 | 7/19/09 10:26 PM | 336 | 8/17/09 4:24 PM | 1006 |

## One execution plan

1. Grab IX_LastAccessDate_Id, seek to 2014/07/01.
2. Write down the Id and LastAccessDate of matching records.
3. Grab the clustered index (white pages), and look up each matching row by their Id to get DisplayName and Age.



© Brent Ozar Unlimited®. All rights reserved.

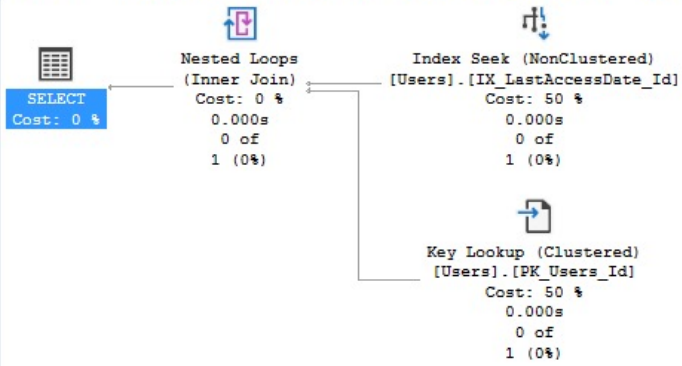
```sql
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate > '3014/07/01'
ORDER BY LastAccessDate;
GO
```

# Sometimes that happens, yes.

150 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id],[DisplayName],[Age] FROM [dbo].[Users] WHERE [LastAccessDate]>@1 ORDER BY [LastAccessDate] ASC
```




# That's why SQL includes the key

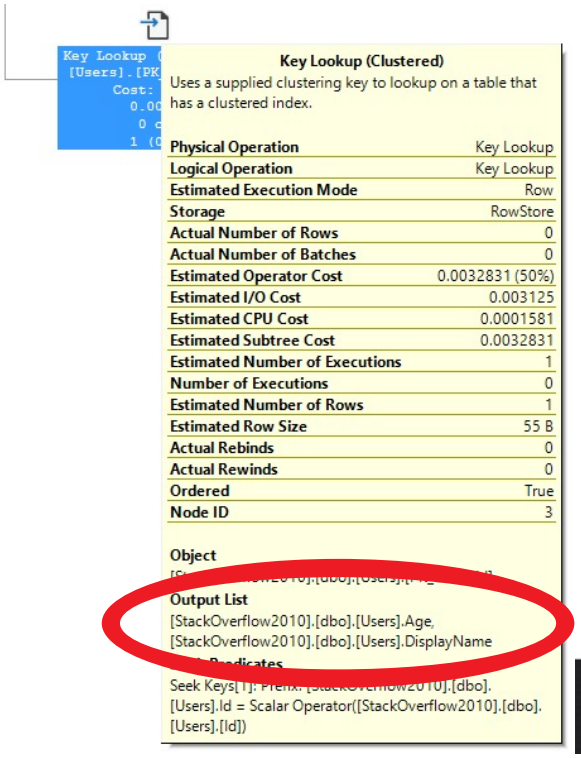
For simplicity, I said I created this index with the Id.

SQL Server always includes your clustering keys whether you ask for 'em or not because it has to join indexes together.

dbo.Users - IX_LastAccessDate

| LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id | LastAccessDate | Id |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | 7/15/09 8:53 AM | 445 | 7/15/09 9:10 PM | 200 | 8/11/09 7:17 PM | 39 |
| 7/15/09 7:08 AM | 22 | 7/15/09 8:58 AM | 457 | 7/16/09 6:22 AM | 678 | 8/12/09 2:54 PM | 943 |
| 7/15/09 7:10 AM | 33 | 7/15/09 9:17 AM | 501 | 7/17/09 2:30 AM | 131 | 8/13/09 4:26 PM | 364 |
| 7/15/09 7:11 AM | 40 | 7/15/09 9:28 AM | 524 | 7/17/09 9:30 AM | 297 | 8/15/09 5:03 PM | 910 |
| 7/15/09 7:11 AM | 41 | 7/15/09 9:30 AM | 527 | 7/17/09 8:43 PM | 998 | 8/17/09 8:42 AM | 202 |
| 7/15/09 7:11 AM | 44 | 7/15/09 9:58 AM | 587 | 7/18/09 12:38 PM | 394 | 8/17/09 10:11 AM | 628 |
| 7/15/09 7:12 AM | 52 | 7/15/09 10:00 AM | 594 | 7/18/09 2:15 PM | 924 | 8/17/09 10:33 AM | 157 |
| 7/15/09 7:13 AM | 64 | 7/15/09 10:02 AM | 597 | 7/19/09 10:26 PM | 336 | 8/17/09 4:24 PM | 1006 |

© Brent Ozar Unlimited®. All rights reserved.

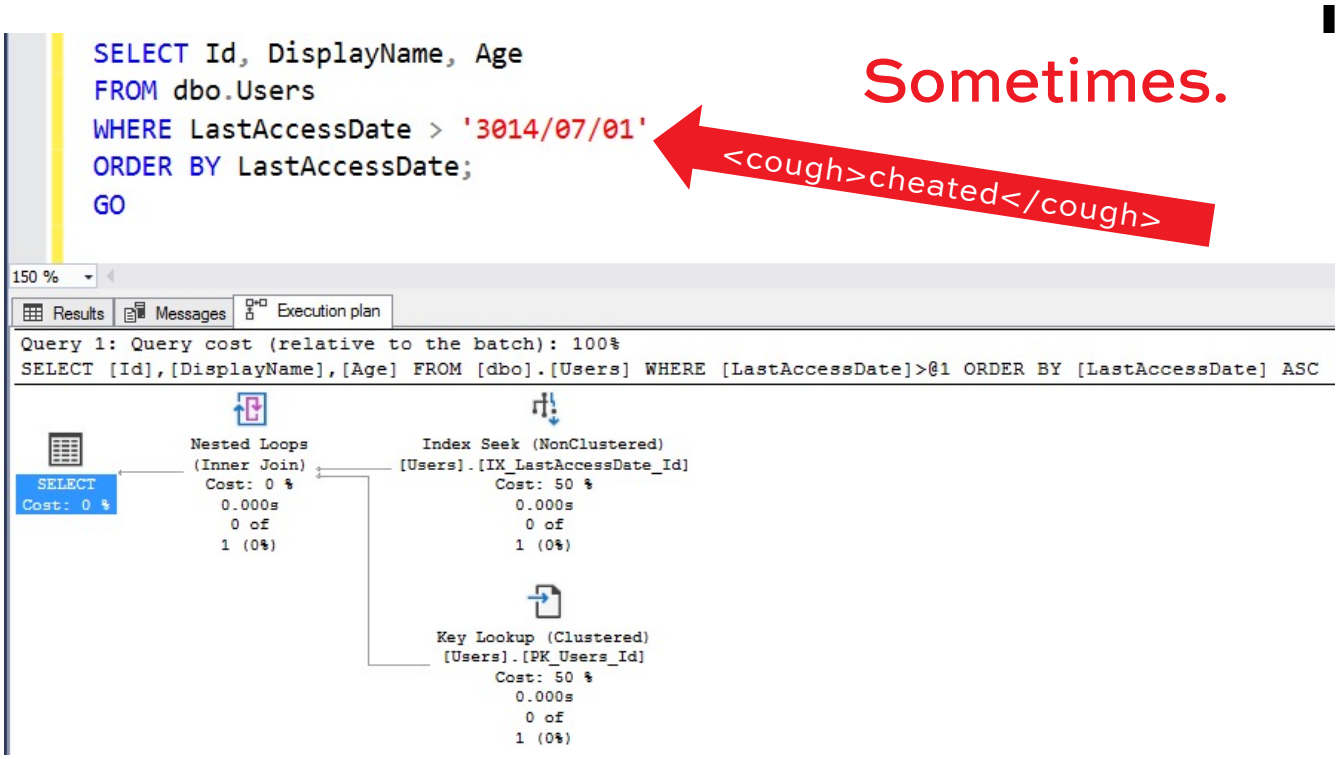



# Classic index tuning sign

Key lookup is required when the index doesn't have all the fields we need.

Hover your mouse over the key lookup, look for the OUTPUT.

Small fields? Frequently used? Add 'em to the index.




© Brent Ozar Unlimited®. All rights reserved.

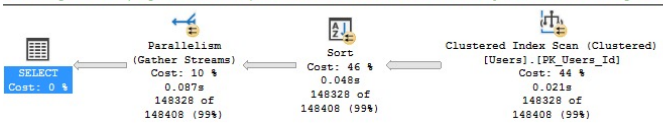
## But sometimes this happens.

```sql
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
GO
```

150 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id],[DisplayName],[Age] FROM [dbo].[Users] WHERE [LastAccessDate]>@1 ORDER BY [LastAccessDate] ASC
Missing Index (Impact 38.6651): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] ON [dbo].[Users] ([LastAccessDate]) INCLUDE ([Age],[DisplayName])
```




Slide 89
Slides, scripts, db: BrentOzar.com/go/engine

## Lesson:
## Even with an index, there's a tipping point where it's more efficient to scan the table.

Learn more about the tipping point:
https://BrentOzar.com/go/tipping

Slide 90
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# To find out why, add a hint

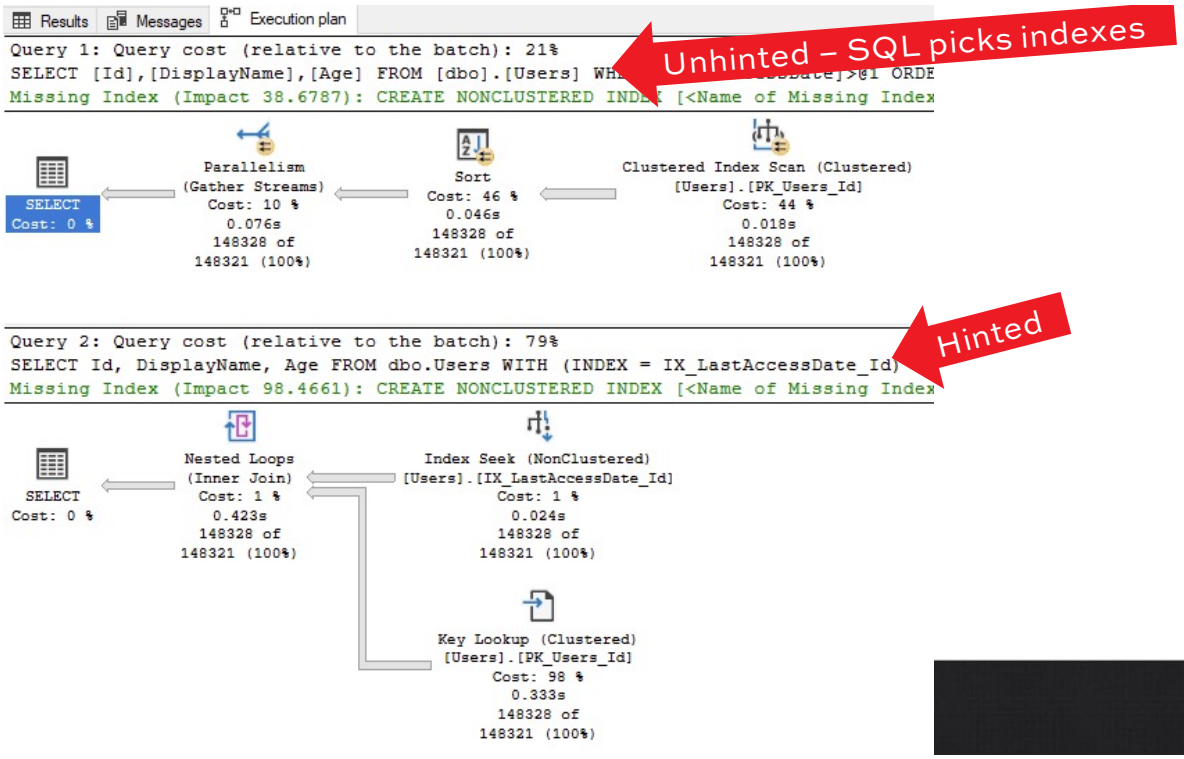
```
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;

SELECT Id, DisplayName, Age
FROM dbo.Users WITH (INDEX = IX_LastAccessDate_Id)
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
```

Unhinted – SQL picks indexes

Hinted

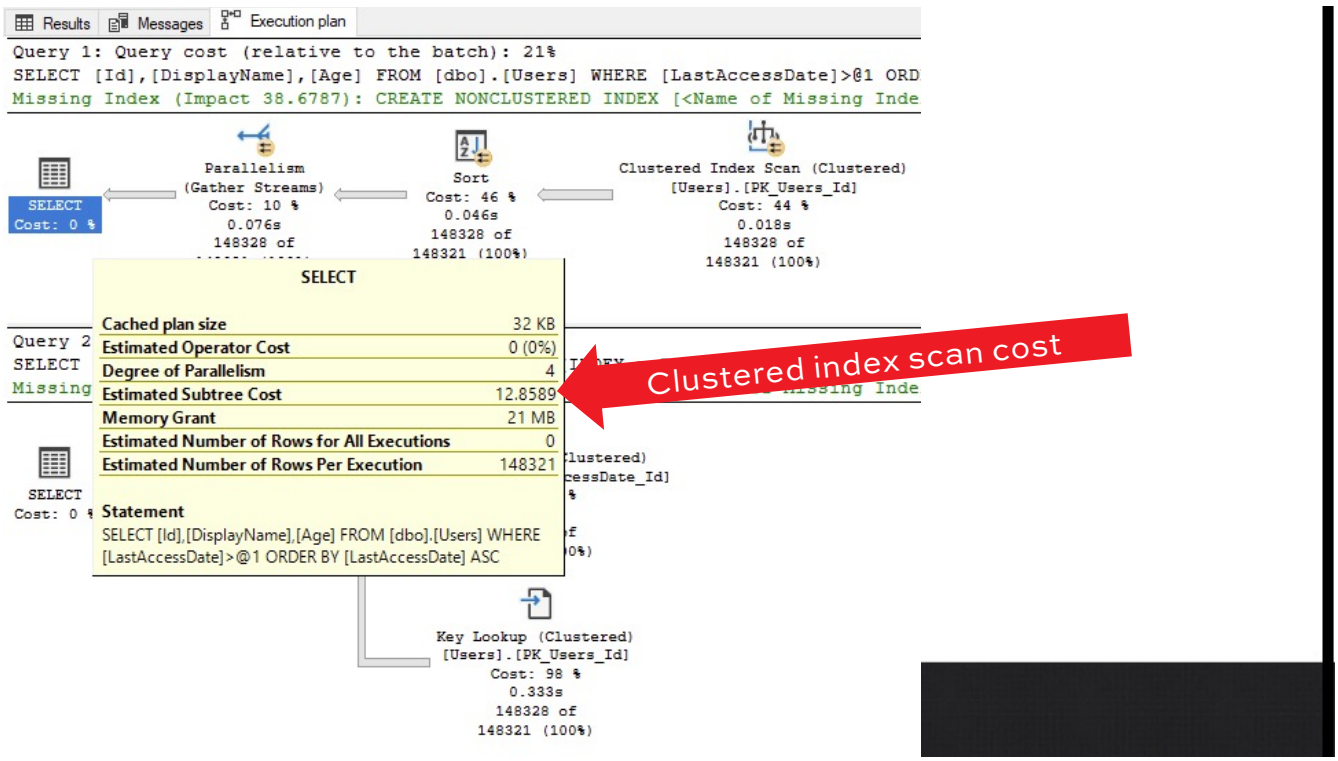
Slide 91
Slides, scripts, db: BrentOzar.com/go/engine

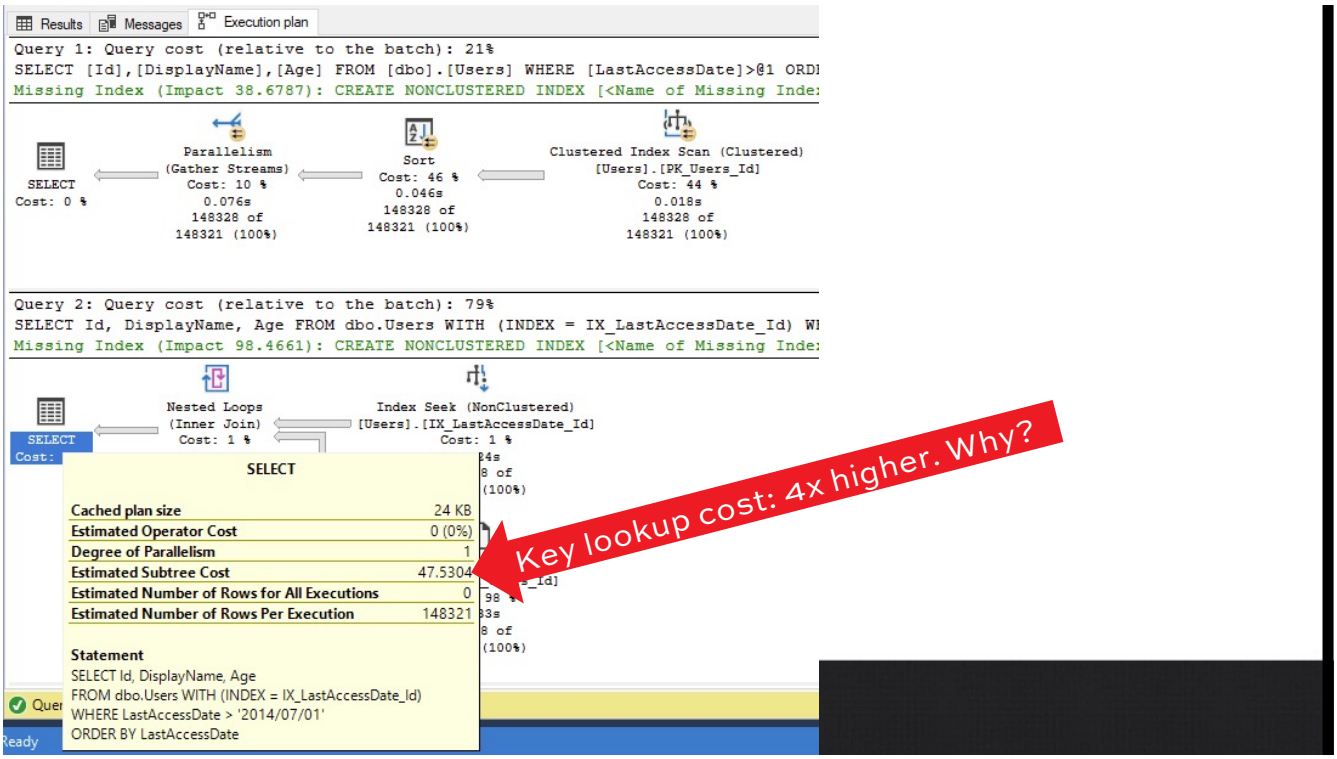



Unhinted – SQL picks indexes

Hinted

© Brent Ozar Unlimited®. All rights reserved.




© Brent Ozar Unlimited®. All rights reserved.

# Decisions, decisions

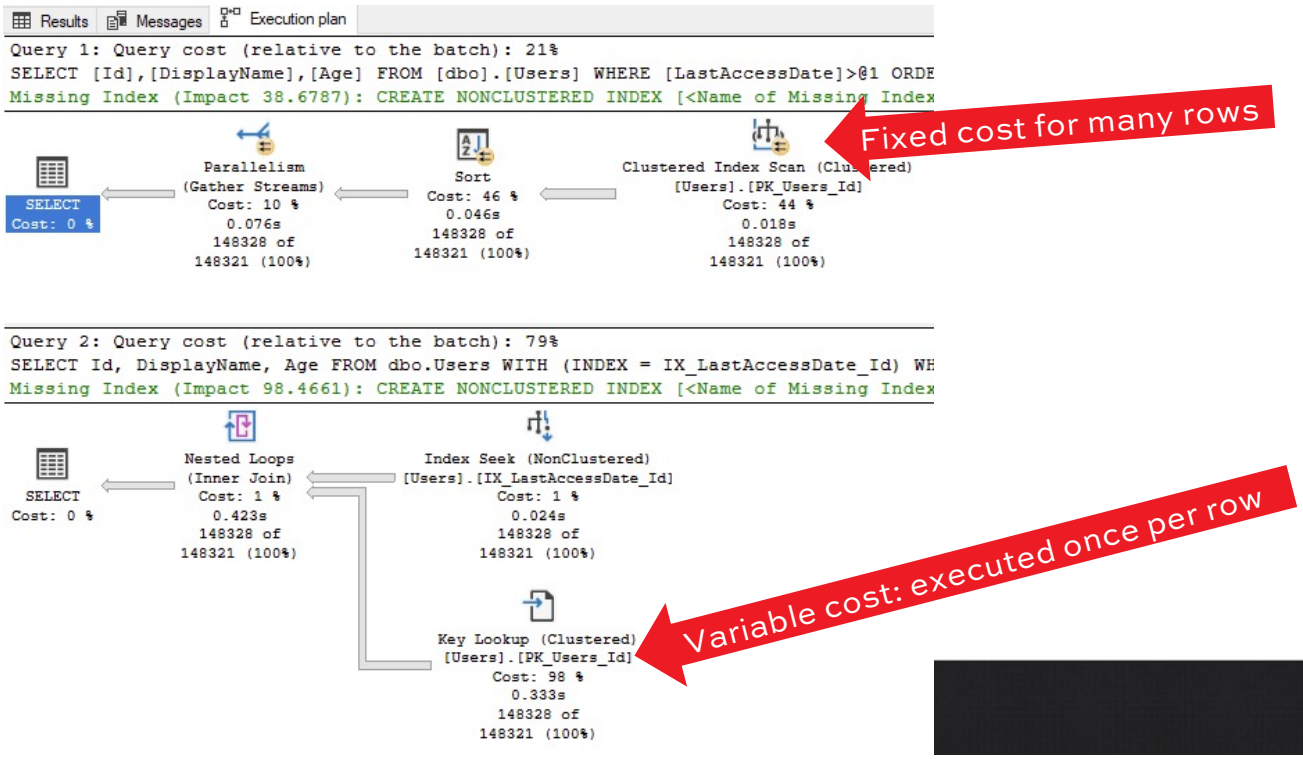
Table scan:

- Fixed, known cost to read all the pages
- Doesn't matter how many rows match our filter

Index seek + key lookups:

- Variable, unknown cost
- The initial seek is cheap, but…
- The key lookup is done once for every row we find
- For a lot of rows, this is worse than a table scan

Slide 95
Slides, scripts, db: BrentOzar.com/go/engine




© Brent Ozar Unlimited®. All rights reserved.

## SQL Server has to decide FIRST.

Phase 1: build the execution plan for a statement

Phase 2: execute it

There is no interplay between these:
we can't go back mid-execution
of one statement.

(We can for multiple statements.)

Slide 97
Slides, scripts, db: BrentOzar.com/go/engine

## How SQL Server decides: Cost-Based Optimizer

1. Build a plan
2. Sum up the cost of all its operators

   Estimated Subtree Cost 12.8589

3. Is there another possible plan?
   If so, go back to step 1 and build that too
4. Out of everything we built,
   go with the lowest-cost plan.

Slide 98
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# I'm simplifying a lot.

Sometimes a query is trivially simple

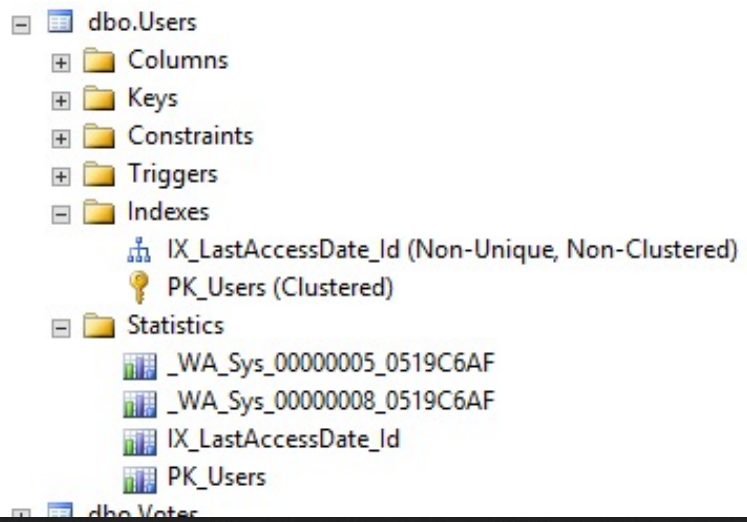
Sometimes the costs are so cheap,
SQL Server stops building more plans,
and goes with a good-enough plan

Sometimes building a plan takes so long that
SQL Server calls timeout , runs a plan as-is

Learn more:
https://BrentOzar.com/go/costbased

Slide 99
Slides, scripts, db: BrentOzar.com/go/engine

# The engine uses statistics.




Slide 100
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Statistics help the engine:

Decide which index to use

What order to process tables/indexes in

Whether to do seeks or scans

Guess how many rows will match your query

How much memory to allocate for the query

Slide 101
Slides, scripts, db: BrentOzar.com/go/engine

```
DBCC SHOW_STATISTICS('dbo.Users', 'IX_LastAccessDate_Id');
GO
```

150 %

Results | Messages

| | Name | Updated | Rows | Rows Sampled | Steps | Density | Average key length | String Index | Filter Expression | Unfiltered Rows | Persisted Sample Percent |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | IX_LastAccessDate_Id | Jan 23 2019 6:40AM | 299398 | 299398 | 134 | 0.9999164 | 12 | NO | NULL | 299398 | 0 |

| | All density | Average Length | Columns |
|---|---|---|---|
| 1 | 3.340515E-06 | 8 | LastAccessDate |
| 2 | 3.340036E-06 | 12 | LastAccessDate, Id |

| | RANGE_HI_KEY | RANGE_ROWS | EQ_ROWS | DISTINCT_RANGE_ROWS | AVG_RANGE_ROWS |
|---|---|---|---|---|---|
| 1 | 2008-08-01 00:59:11.147 | 0 | 1 | 0 | 1 |
| 2 | 2008-11-27 09:02:52.947 | 1222 | 1 | 1222 | 1 |
| 3 | 2009-04-30 11:47:04.320 | 2047 | 1 | 2047 | 1 |
| 4 | 2009-09-01 09:48:52.773 | 2047 | 1 | 2047 | 1 |
| 5 | 2009-10-25 17:47:51.473 | 2047 | 1 | 2047 | 1 |
| 6 | 2009-11-03 13:22:45.970 | 1023 | 1 | 1023 | 1 |
| 7 | 2009-12-10 01:29:00.200 | 5119 | 1 | 5119 | 1 |
| 8 | 2009-12-23 10:21:31.327 | 2047 | 1 | 2047 | 1 |
| 9 | 2010-01-06 04:43:41.433 | 1535 | 1 | 1535 | 1 |
| 10 | 2010-02-02 12:35:59.493 | 4607 | 1 | 4607 | 1 |
| 11 | 2010-03-03 03:29:14.040 | 5119 | 1 | 5119 | 1 |
| 12 | 2010-03-24 09:05:53.317 | 4095 | 1 | 4095 | 1 |

© Brent Ozar Unlimited®. All rights reserved.

# Other data types get stats, too

```sql
CREATE INDEX IX_Reputation ON dbo.Users(Reputation);
GO
DBCC SHOW_STATISTICS('dbo.Users', 'IX_Reputation');
GO
```

150 %

Results | Messages

| | Name | Updated | Rows | Rows Sampled | Steps | Density | Average key length | String Index |
|---|---|---|---|---|---|---|---|---|
| 1 | IX_Reputation | Jan 23 2019 11:11AM | 299398 | 299398 | 172 | 0.1184246 | 8 | NO |

| | All density | Average Length | Columns |
|---|---|---|---|
| 1 | 6.142506E-05 | 4 | Reputation |
| 2 | 3.340036E-06 | 8 | Reputation, Id |

INT

| | RANGE_HI_KEY | RANGE_ROWS | EQ_ROWS | DISTINCT_RANGE_ROWS | AVG_RANGE_ROWS |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 40713 | 0 | 1 |
| 2 | 3 | 111 | 4313 | 1 | 111 |
| 3 | 4 | 0 | 658 | 0 | 1 |
| 4 | 6 | 411 | 13551 | 1 | 411 |
| 5 | 8 | 261 | 3935 | 1 | 261 |
| 6 | 9 | 0 | 1213 | 0 | 1 |
| 7 | 11 | 536 | 18251 | 1 | 536 |
| 8 | 13 | 257 | 3189 | 1 | 257 |
| 9 | 15 | 507 | 692 | 1 | 507 |
| 10 | 16 | 0 | 6140 | 0 | 1 |
| 11 | 18 | 382 | 2292 | 1 | 382 |
| 12 | 20 | 917 | 637 | 1 | 917 |
| 13 | 21 | 0 | 8173 | 0 | 1 |

```sql
CREATE INDEX IX_DisplayName ON dbo.Users(DisplayName);
GO
DBCC SHOW_STATISTICS('dbo.Users', 'IX_DisplayName');
```

150 %

Results | Messages

| | Name | Updated | Rows | Rows Sampled | Steps | Density | Average key length | String Index |
|---|---|---|---|---|---|---|---|---|
| 1 | IX_DisplayName | Jan 23 2019 11:10AM | 299398 | 299398 | 198 | 0.7731416 | 20.35281 | YES |

NVARCHAR

| | | | Columns |
|---|---|---|---|
| 1 | [illegible] | | DisplayName |
| 2 | 3.340036E-06 | 20.3528 | DisplayName, Id |

| | RANGE_HI_KEY | RANGE_ROWS | EQ_ROWS | DISTINCT_RANGE_ROWS | AVG_RANGE_ROWS |
|---|---|---|---|---|---|
| 1 | guipo | 0 | 1 | 0 | 1 |
| 2 | Aaron | 1100 | 218 | 960 | 1.145833 |
| 3 | Adam | 1788 | 359 | 1271 | 1.406766 |
| 4 | Adrian | 1180 | 106 | 904 | 1.30531 |
| 5 | Ajay | 1498 | 57 | 1192 | 1.256711 |
| 6 | Alan | 950 | 145 | 658 | 1.443769 |
| 7 | Alex | 982 | 831 | 735 | 1.336054 |
| 8 | Alexander | 698 | 98 | 594 | 1.175084 |
| 9 | allan | 1478 | 47 | 1081 | 1.367253 |
| 10 | Amit | 1430 | 137 | 1051 | 1.360609 |
| 11 | Anders | 963 | 43 | 626 | 1.538339 |
| 12 | Andrew | 1087 | 463 | 705 | 1.541844 |

Slide 103
Slides, scripts, db: BrentOzar.com/go/engine

# How the engine uses statistics

It needs to figure out how many rows are going to match our WHERE clause:

```sql
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate > '2014/07/01'
ORDER BY LastAccessDate;
```

Slide 104
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# It reads stats on LastAccessDate

```
DBCC SHOW_STATISTICS('dbo.Users', 'IX_LastAccessDate_Id');
GO
```

150 %

Results | Messages

| | Name | Updated | Rows | Rows Sampled | Steps | Density | Average key length | String Index |
|---|---|---|---|---|---|---|---|---|
| 1 | IX_LastAccessDate_Id | Jan 23 2019 6:40AM | 299398 | 299398 | 134 | 0.9999164 | 12 | NO |

| | All density | Average Length | Columns |
|---|---|---|---|
| 1 | 3.340515E-06 | 8 | LastAccessDate |
| 2 | 3.340036E-06 | 12 | LastAccessDate, Id |

| | RANGE_HI_KEY | RANGE_ROWS | EQ_R | | AVG_RANGE_ROWS |
|---|---|---|---|---|---|
| 53 | 2013-09-23 18:59:13.903 | 1023 | 1 | | 1 |
| 54 | 2013-12-18 20:17:27.117 | 2047 | 1 | | 1 |
| 55 | 2014-03-03 15:30:38.193 | 1535 | | | 1 |
| 56 | 2015-03-20 14:26:11.510 | 8573 | 1 | 8573 | 1 |
| 57 | 2015-06-01 13:30:07.940 | 2047 | 1 | 2047 | 1 |
| 58 | 2015-07-27 06:50:21.030 | 2047 | 1 | 2047 | 1 |

Add it up
Add it up

```
WHERE LastAccessDate
> '2014/07/01'
```

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id],[DisplayName],[Age] FROM [dbo].[Users] WHERE [LastAccessDate]>@1 ORDER BY [LastAccessDate] ASC
Missing Index (Impact 38.6651): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] ON [dbo].[Users] ([LastAccessI
```

SELECT
Cost: 0 %

Parallelism
(Gather Streams)
Cost: 10 %
0.079s
148328 of
148408 (99%)

Sort
Cost: 46 %
0.044s
148328 of
148408 (99%)

Clustered Index Scan (Clustered)
[Users].[PK_Users_Id]
Cost: 44 %
0.018s
148328 of
148408 (99%)

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

| | |
|---|---|
| **Physical Operation** | Clustered Index Scan |
| **Logical Operation** | Clustered Index Scan |
| **Actual Execution Mode** | Row |
| **Estimated Execution Mode** | Row |
| **Storage** | RowStore |
| **Number of Rows Read** | 299398 |
| **Actual Number of Rows** | 148328 |
| **Actual Number of Batches** | 0 |
| **Estimated I/O Cost** | 5.46609 |
| **Estimated Operator Cost** | 5.63084 (44%) |
| **Estimated CPU Cost** | 0.164747 |
| **Estimated Subtree Cost** | 5.63084 |
| **Number of Executions** | 4 |
| **Estimated Number of Executions** | 1 |
| **Estimated Number of Rows** | 148408 |
| **Estimated Number of Rows to Be Read** | 299398 |
| **Estimated Row Size** | 67 B |
| **Actual Rebinds** | 0 |
| **Actual Rewinds** | 0 |
| **Ordered** | False |
| **Node ID** | 2 |

**Predicate**
[StackOverflow2010].[dbo].[Users].[LastAccessDate]>'2014-07-01 00:00:00.000'
**Object**
[StackOverflow2010].[dbo].[Users].[PK_Users_Id]
**Output List**
[StackOverflow2010].[dbo].[Users].Id, [StackOverflow2010].[dbo].[Users].Age, [StackOverflow2010].[dbo].[Users].DisplayName, [StackOverflow2010].[dbo].[Users].LastAccessDate

© Brent Ozar Unlimited®. All rights reserved.

## In this case, stats worked great!

We had statistics on the LastAccessDate column

The statistics were up to date and accurately sampled

Our query was easy to understand:

```
WHERE LastAccessDate > '2014/07/01'
```

We were using the date column contents as-is, not running calculations on it

Slide 107
Slides, scripts, db: BrentOzar.com/go/engine

## Ways this can go wrong

The statistics on LastAccessDate might be outdated

The statistics may be inaccurately sampled

Learn more about statistics contents & maintenance: https://BrentOzar.com/go/statsclass

Our query may not be as easy to understand as this:

```
WHERE LastAccessDate > '2014/07/01'
```

Slide 108
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

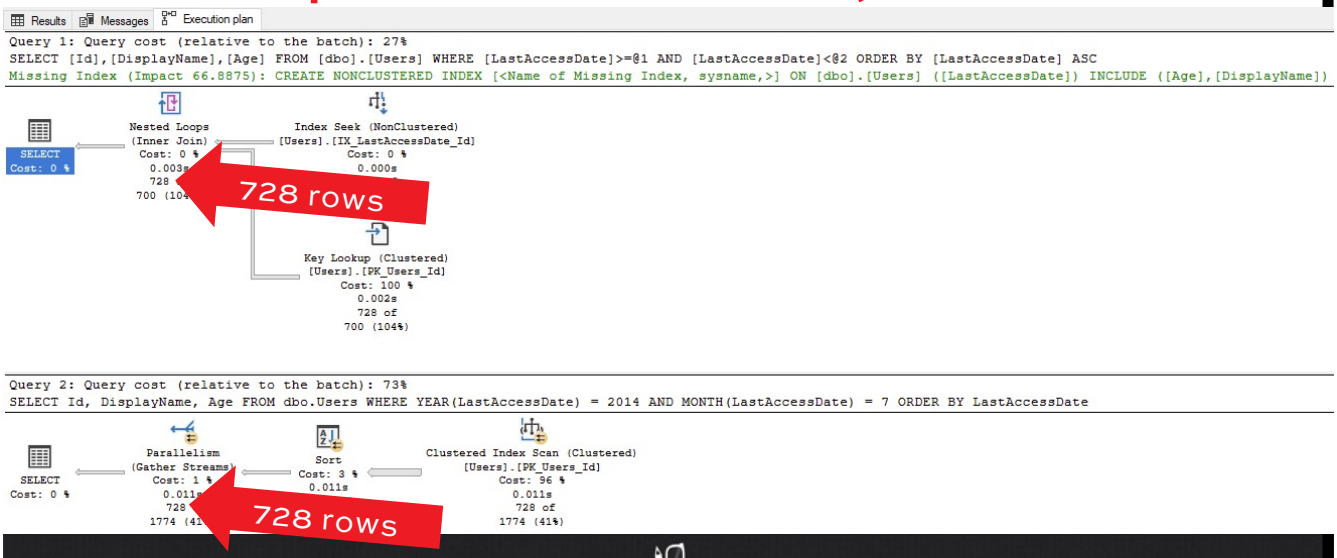
# Two ways to write the same query

```sql
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate >= '2014/07/01'
  AND LastAccessDate < '2014/08/01'
ORDER BY LastAccessDate;

SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE YEAR(LastAccessDate) = 2014
  AND MONTH(LastAccessDate) = 7
ORDER BY LastAccessDate;
```

Slide 109
Slides, scripts, db: BrentOzar.com/go/engine

# Both produce 728 rows, but...




Slide 110
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

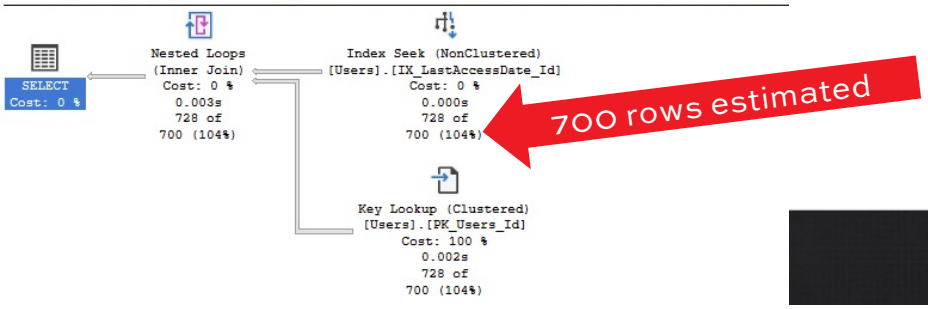
# This one was easy to interpret

```sql
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate >= '2014/07/01'
  AND LastAccessDate < '2014/08/01'
ORDER BY LastAccessDate;
```

200 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT [Id],[DisplayName],[Age] FROM [dbo].[Users] WHERE [LastAccessDate]>=@1 A
Missing Index (Impact 66.8875): CREATE NONCLUSTERED INDEX [<Name of Missing Ind
```



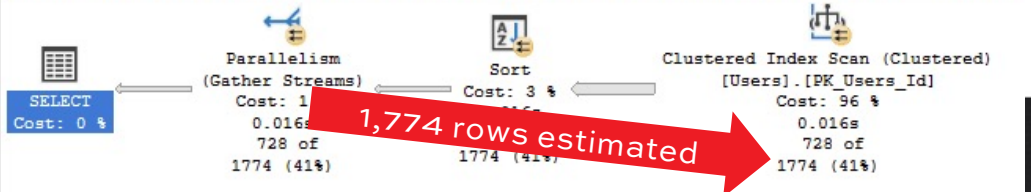

# This one was harder to understand

```sql
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE YEAR(LastAccessDate) = 2014
  AND MONTH(LastAccessDate) = 7
ORDER BY LastAccessDate;
```

200 %

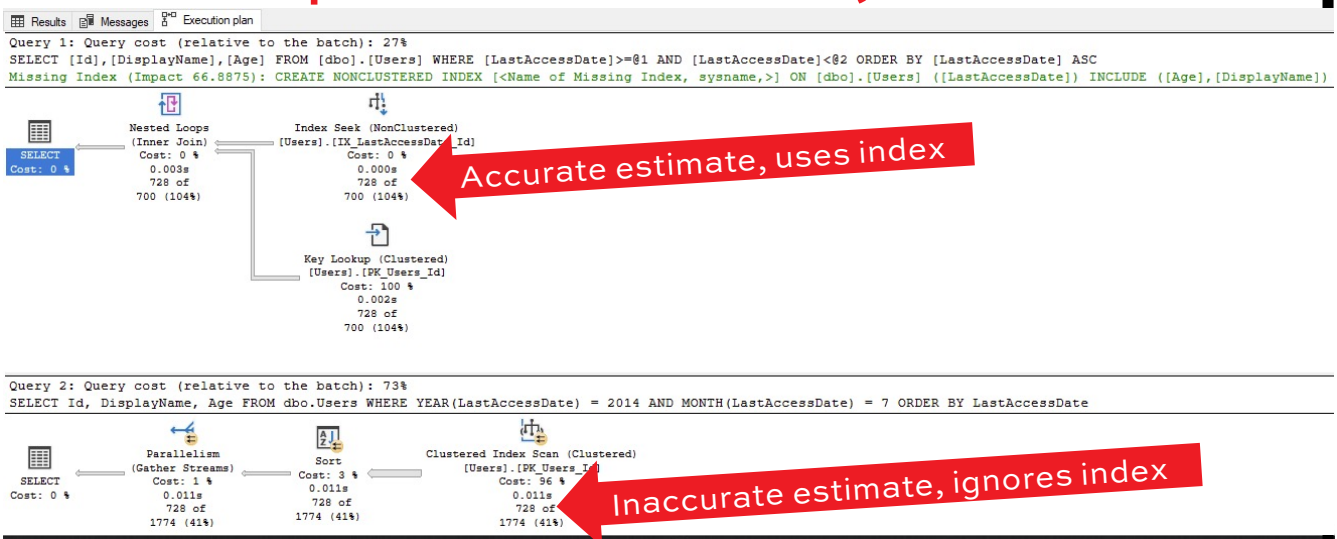
Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT Id, DisplayName, Age FROM dbo.Users WHERE YEAR(LastAccessDate) = 2014 A
```



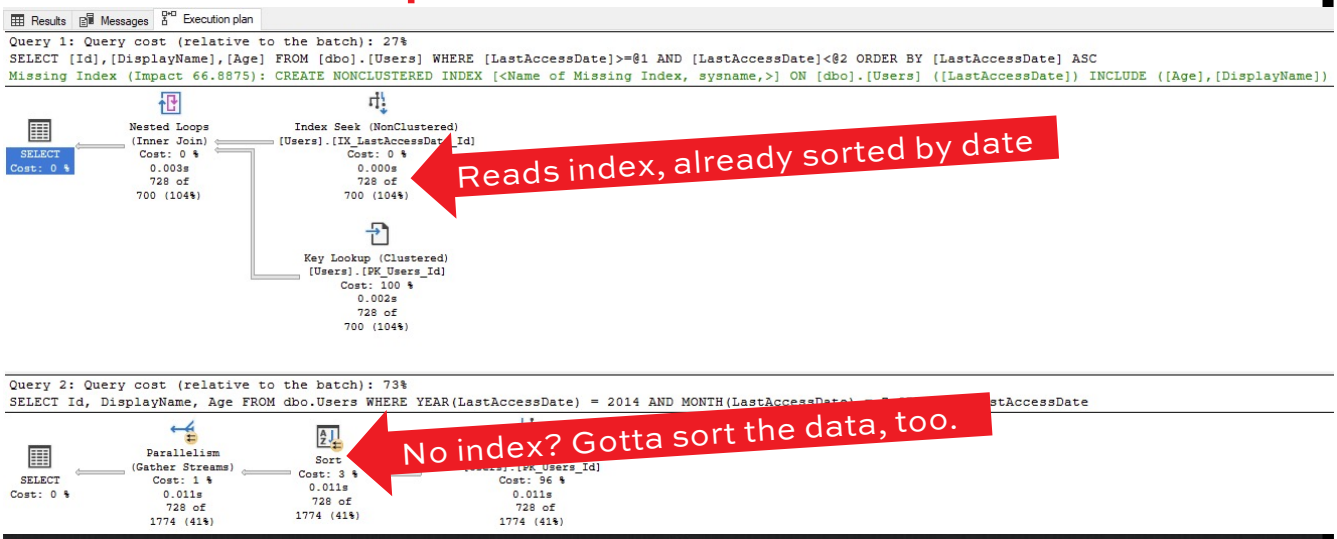

© Brent Ozar Unlimited®. All rights reserved.

# Both produce 728 rows, but...




Slide 113
Slides, scripts, db: BrentOzar.com/go/engine

# Another plan difference




Slide 114
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## To see the overhead of the sort

`SET STATISTICS IO ON`
shows logical reads

`SET STATISTICS TIME ON`
shows CPU and elapsed time

`SET STATISTICS IO, TIME ON`
shows both

Slide 115
Slides, scripts, db: BrentOzar.com/go/engine

## Comparing 'em

| | Passing in specific dates | YEAR() and MONTH() |
|---|---|---|
| Which | Index seek + key lookups | Clustered index scan |
| Logical reads | 2,244 | 7,778 |
| CPU time, milliseconds | 0 | 15 |
| Query duration, milliseconds | 78 | 116 |

The YEAR/MONTH solution is worse in every way.

Slide 116
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## The simpler your syntax, the better your queries run.

```sql
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate >= '2014/07/01'
  AND LastAccessDate < '2014/08/01'
ORDER BY LastAccessDate;

SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE YEAR(LastAccessDate) = 2014
  AND MONTH(LastAccessDate) = 7
ORDER BY LastAccessDate;
```

Slide 117
Slides, scripts, db: BrentOzar.com/go/engine

## But COULD it use the index?

```sql
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE YEAR(LastAccessDate) = 2014
  AND MONTH(LastAccessDate) = 7
ORDER BY LastAccessDate;

SELECT Id, DisplayName, Age
FROM dbo.Users WITH (INDEX = IX_LastAccessDate_Id)
WHERE YEAR(LastAccessDate) = 2014
  AND MONTH(LastAccessDate) = 7
ORDER BY LastAccessDate;
```

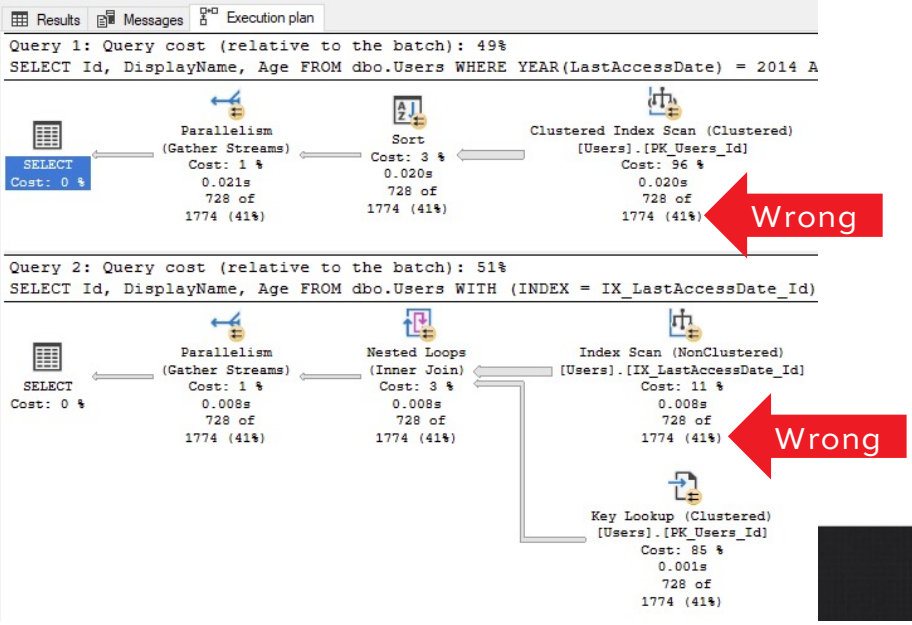
Let SQL pick, versus...

Tell it

Slide 118
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# The estimates are still wrong...




# But did the index go faster

| | SQL picks a table scan | We force the index |
|---|---|---|
| Logical reads | 7,778 | 2,942 |
| CPU time, milliseconds | 31 | 0 |
| Query duration, milliseconds | 101 | 98 |

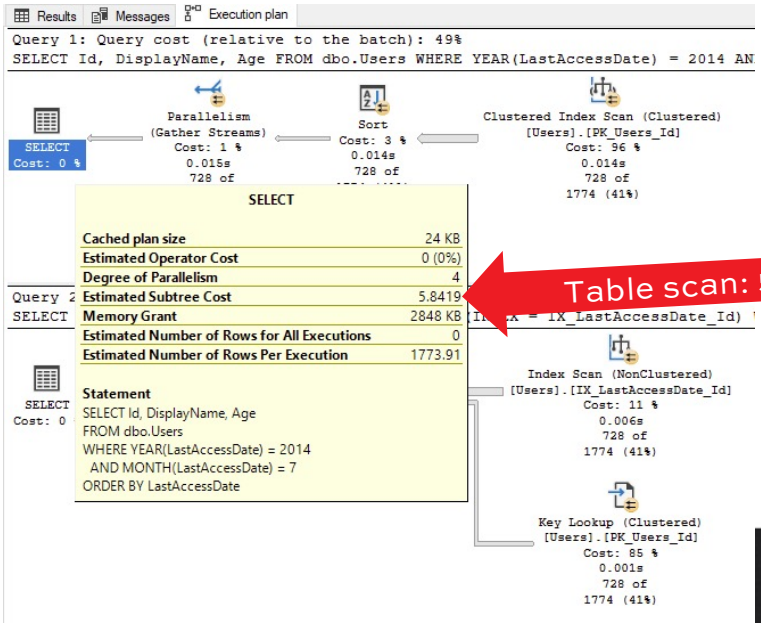
The index is better in every way.

So why didn't SQL pick the plan with the index?

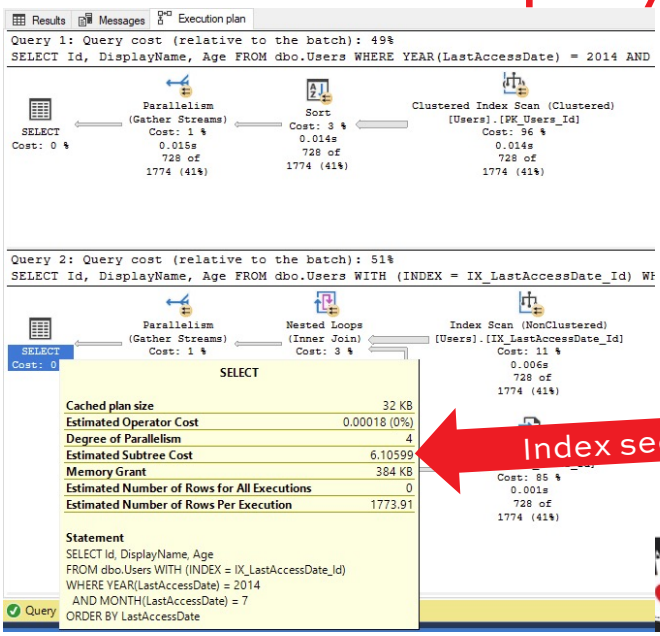


© Brent Ozar Unlimited®. All rights reserved.

# Look at the query costs.

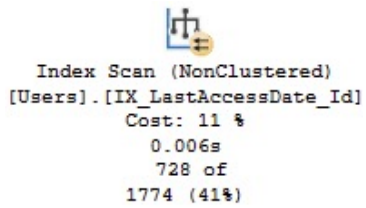



# Look at the query costs.




© Brent Ozar Unlimited®. All rights reserved.

# How is a faster plan more expensive?!?




Because the plan is built based on the estimated number of rows.

If we actually found more rows, the plan would have done more logical reads.

SQL Server picked the wrong plan because his estimated rows was wrong.

And that's your fault, really.*

Slide 123
Slides, scripts, db: BrentOzar.com/go/engine

# The simpler your syntax, the better your queries run.

```
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate >= '2014/07/01'
  AND LastAccessDate < '2014/08/01'
ORDER BY LastAccessDate;

SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE YEAR(LastAccessDate) = 2014
  AND MONTH(LastAccessDate) = 7
ORDER BY LastAccessDate;
```

Slide 124
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Recap (so far)

Slide 125
Slides, scripts, db: BrentOzar.com/go/engine

## What you learned so far

Key lookup: the query wanted more columns than we had in the nonclustered index

The tipping point: when SQL thinks it's more efficient to scan a table instead of all those key lookups

Cost-based optimization: how plans are chosen

The simpler your syntax, the more accurate the costs, and the better the plan

Slide 126
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## What you'll learn next

How included columns mitigate these problems

Why included columns aren't magically delicious

When to consider adding columns as includes

Why key column order is so important

Slide 127
Slides, scripts, db: BrentOzar.com/go/engine

BRENT OZAR
UNLIMITED®

# How to Think Like the SQL Server Engine

## Part 4: Reducing Problems by Adding Columns

Slide 128
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Where we left off

SQL had challenges choosing between:

- Index seek + key lookup, or
- Table scan

And our query syntax wasn't helping.

```sql
SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE LastAccessDate >= '2014/07/01'
  AND LastAccessDate < '2014/08/01'
ORDER BY LastAccessDate;

SELECT Id, DisplayName, Age
FROM dbo.Users
WHERE YEAR(LastAccessDate) = 2014
  AND MONTH(LastAccessDate) = 7
ORDER BY LastAccessDate;
```

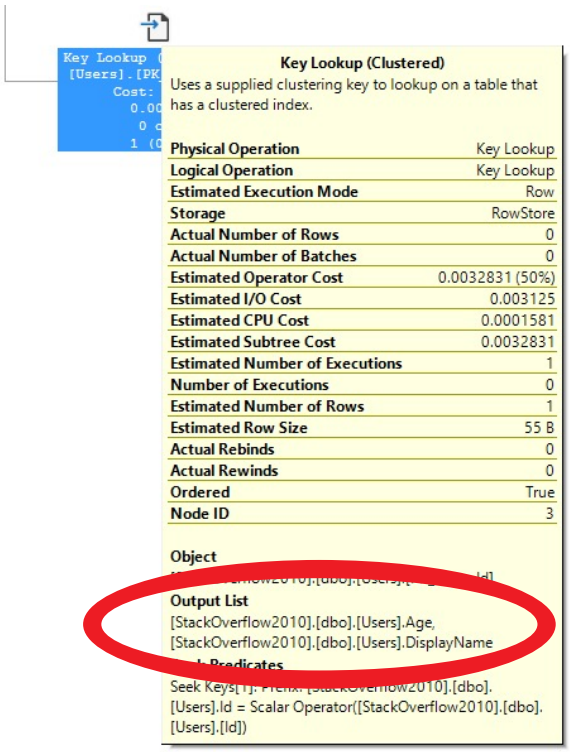
Slide 129
Slides, scripts, db: BrentOzar.com/go/engine

# Classic index tuning sign

A key lookup is required when the index doesn't have all the columns our query needs.

Hover mouse over the key lookup, look for the OUTPUT.

Small fields? Frequently used? Add 'em to the index.




Slide 130
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Add a new, wider index

```
CREATE INDEX IX_LastAccessDate_Id_DisplayName_Age
ON dbo.Users
(LastAccessDate, Id, DisplayName, Age)
```

dbo.Users - IX_LastAccessDate_ID_DisplayName_Age

| LastAccessDate | Id | DisplayName | Age | LastAccessDate | Id | DisplayName | Age |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | Community | 1 | 8/18/09 10:22 AM | 673 | | NULL |
| 7/15/09 9:58 AM | 587 | | NULL | 8/18/09 1:14 PM | 643 | Sarcastic | 30 |
| 7/15/09 10:00 AM | 594 | Louis Haußknecht | NULL | 8/18/09 2:28 PM | 690 | msh | 31 |
| 7/15/09 10:02 AM | 597 | elo80ka | 29 | 8/19/09 9:22 AM | 750 | Cyphus | 30 |
| 7/15/09 10:21 AM | 618 | Marien | 30 | 8/20/09 3:45 AM | 749 | | NULL |
| 7/15/09 10:26 AM | 623 | Fionn | NULL | 8/20/09 12:05 PM | 752 | Sean Taylor | 27 |
| 7/15/09 10:28 AM | 629 | Zoomzoom83 | 26 | 8/22/09 11:58 AM | 580 | n00ki3 | 22 |
| 7/15/09 10:32 AM | 638 | | NULL | 8/26/09 2:43 AM | 851 | Tim Howland | 41 |
| 7/15/09 10:36 AM | 642 | aherrick | 23 | 8/26/09 9:59 AM | 546 | hitec | 27 |
| 7/15/09 10:46 AM | 661 | | NULL | 8/28/09 7:51 PM | 668 | mtruesdell | 34 |

# The good news: even the “bad” query uses it.

```
9   SELECT Id, DisplayName, Age
10  FROM dbo.Users
11  WHERE YEAR(LastAccessDate) = 2014
12    AND MONTH(LastAccessDate) = 7
13  ORDER BY LastAccessDate;
```

200 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT Id, DisplayName, Age FROM dbo.Users WHERE YEAR(LastAccessDate) = 201
```

SELECT
Cost: 2 %

Index Scan (NonClustered)
[Users].[IX_LastAccessDate_Id_Displ...
Cost: 98 %
0.027s
728 of
1774 (41%)

Estimates are wrong, but doesn’t matter

© Brent Ozar Unlimited®. All rights reserved.

# Bad news: it's physically larger.

Because this index has more data in it,
it's physically larger in memory and on disk.

It takes more 8KB pages to store the same # of rows.

# Measuring index sizes

sp_BlitzIndex: free script from BrentOzar.com

```
25 | sp_BlitzIndex @TableName = 'Users'
```

200 %

Results | Messages

| | Details: db_schema.table.index(indexid) | Definition: [Property] ColumnName {datatype maxbytes} | Size |
|---|---|---|---|
| 1 | Database [StackOverflow2010] as of 2021-09-18 09:00... | http://FirstResponderKit.org | NULL |
| 2 | dbo.Users.PK_Users_Id (1) | [CX] [PK] [1 KEY] Id {int 4} | 299,398 rows; 58.1MB; |
| 3 | dbo.Users.IX_LastAccessDate_Id (2) | [2 KEYS] LastAccessDate {datetime 8}, Id {int 4} | 299,398 rows; 5.3MB |
| 4 | dbo.Users.IX_LastAccessDate_Id_DisplayName_Age (4) | [4 KEYS] LastAccessDate {datetime 8}, Id {int 4}, DisplayName {nvarchar (40)}, Age {int 4} | 299,398 rows; 12.7MB |

Learn more: https://BrentOzar.com/blitzindex/



© Brent Ozar Unlimited®. All rights reserved.

## The more indexes you have, and the more columns they have...

The more space they take up in memory
(the less other objects you can cache)

The more space they take up on disk
(on your primary and all of the replicas, too)

The longer your backups, restores, CHECKDB take

The longer DUI operations take

The longer index maintenance takes

Slide 135
Slides, scripts, db: BrentOzar.com/go/engine

## Warning signs

You might have too many indexes if:

- People complain about slow inserts & deletes
- Monitoring tools warn about blocking, deadlocks
- Big database size is a problem

You might not have enough if:

- People complain about slow selects
  (but not inserts & deletes)
- You can cache a lot of the database in memory

Slide 136
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Possible workaround: INCLUDES

Some folks will say to INCLUDE columns in an index.

INCLUDES aren't part of the key, they say.

It might help you cover more columns without increasing the index size.

```
CREATE INDEX IX_LastAccessDate_Id_Includes
ON dbo.Users (LastAccessDate, Id)
INCLUDE (DisplayName, Age)
```

Slide 137
Slides, scripts, db: BrentOzar.com/go/engine

# Except...hmm.

Both indexes are almost exactly the same size:

```
25  sp_BlitzIndex @TableName = 'Users'
```

200 %

Results | Messages

| | Details: db_schema.table.index(indexid) | Definition: [Property] ColumnName {datatype maxbytes} | Size |
|---|---|---|---|
| 1 | Database [StackOverflow2010] as of 2021-09-18 09:0... | http://FirstResponderKit.org | NULL |
| 2 | dbo.Users.PK_Users_Id (1) | [CX] [PK] [1 KEY] Id {int 4} | 299,398 rows; 58.1MB; |
| 3 | dbo.Users.IX_LastAccessDate_Id (2) | [2 KEYS] LastAccessDate {datetime 8}, Id {int 4} | 299,398 rows; 5.3MB |
| 4 | dbo.Users.IX_LastAccessDate_Id_Includes (3) | [2 KEYS] LastAccessDate {datetime 8}, Id {int 4} [2 INCLUDES] Age {int 4}, DisplayName {nvarchar (40)} | 299,398 rows; 12.7MB |
| 5 | dbo.Users.IX_LastAccessDate_Id_DisplayName_Age (4) | [4 KEYS] LastAccessDate {datetime 8}, Id {int 4}, DisplayName {nvarchar (40)}, Age {int 4} | 299,398 rows; 12.7MB |

Slide 138
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Their pages have the same stuff.

```
CREATE INDEX IX_LastAccessDate_Id_DisplayName_Age
ON dbo.Users (LastAccessDate, Id, DisplayName, Age)

CREATE INDEX IX_LastAccessDate_Id_Includes
ON dbo.Users (LastAccessDate, Id)
INCLUDE (DisplayName, Age)
```

dbo.Users - IX_LastAccessDate_ID_DisplayName_Age

| LastAccessDate | Id | DisplayName | Age |
|---|---|---|---|
| 7/31/08 12:00 AM | -1 | Community | 1 |
| 7/15/09 9:58 AM | 587 | | NULL |
| 7/15/09 10:00 AM | 594 | Louis Haußknecht | NULL |
| 7/15/09 10:02 AM | 597 | elo80ka | 29 |
| 7/15/09 10:21 AM | 618 | Marien | 30 |
| 7/15/09 10:26 AM | 623 | Fionn | NULL |
| 7/15/09 10:28 AM | 629 | Zoomzoom83 | 26 |
| 7/15/09 10:32 AM | 638 | | NULL |
| 7/15/09 10:36 AM | 642 | aherrick | 23 |
| 7/15/09 10:46 AM | 661 | | NULL |

| LastAccessDate | Id | DisplayName | Age |
|---|---|---|---|
| 8/18/09 10:22 AM | 673 | | NULL |
| 8/18/09 1:14 PM | 643 | Sarcastic | 30 |
| 8/18/09 2:28 PM | 690 | msh | 31 |
| 8/19/09 9:22 AM | 750 | Cyphus | 30 |
| 8/20/09 3:45 AM | 749 | | NULL |
| 8/20/09 12:05 PM | 752 | Sean Taylor | 27 |
| 8/22/09 11:58 AM | 580 | n00ki3 | 22 |
| 8/26/09 2:43 AM | 851 | Tim Howland | 41 |
| 8/26/09 9:59 AM | 546 | hitec | 27 |
| 8/28/09 7:51 PM | 668 | mtruesdell | 34 |

# In theory...

Included columns should be lighter-weight because when their contents change, they don't cause the data to be re-sorted.

But is that really true?

Let's find out.

Slide 140
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Say Sarcastic got older.

If Age is only INCLUDED in the index, we don't have to move Sarcastic around on the page. Okay. So far, so good.

```
CREATE INDEX IX_LastAccessDate_Id_Includes
ON dbo.Users (LastAccessDate, Id)
INCLUDE (DisplayName, Age)
```

dbo.Users - IX_LastAccessDate_ID_DisplayName_Age

| LastAccessDate | Id | DisplayName | Age | LastAccessDate | Id | DisplayName | Age |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | Community | 1 | 8/18/09 10:22 AM | 673 | | NULL |
| 7/15/09 9:58 AM | 587 | | NULL | 8/18/09 1:14 PM | 643 | Sarcastic | 30 |
| 7/15/09 10:00 AM | 594 | Louis Haußknecht | NULL | 8/18/09 2:28 PM | 690 | msh | 31 |
| 7/15/09 10:02 AM | 597 | elo80ka | 29 | 8/19/09 9:22 AM | 75[illegible] | | 30 |
| 7/15/09 10:21 AM | 618 | Marien | 30 | 8/20/09 3:45 AM | [illegible] | | NULL |
| 7/15/09 10:26 AM | 623 | Fionn | NULL | 8/20/09 12:05 PM | 752 | Sean Taylor | 27 |
| 7/15/09 10:28 AM | 629 | Zoomzoom83 | 26 | 8/22/09 11:58 AM | 580 | n00ki3 | 22 |
| 7/15/09 10:32 AM | 638 | | NULL | 8/26/09 2:43 AM | 851 | Tim Howland | 41 |
| 7/15/09 10:36 AM | 642 | aherrick | 23 | 8/26/09 9:59 AM | 546 | hitec | 27 |
| 7/15/09 10:46 AM | 661 | | NULL | 8/28/09 7:51 PM | 668 | mtruesdell | 34 |

Set Age = 31

# But if Age is in the index keys...

Now what happens?

```
CREATE INDEX IX_LastAccessDate_Id_DisplayName_Age
ON dbo.Users (LastAccessDate, Id, DisplayName, Age)
```

Do we have to move Sarcastic's row? If so, where on this page?

dbo.Users - IX_LastAccessDate_ID_DisplayName_Age

| LastAccessDate | Id | DisplayName | Age | LastAccessDate | Id | DisplayName | Age |
|---|---|---|---|---|---|---|---|
| 7/31/08 12:00 AM | -1 | Community | 1 | 8/18/09 10:22 AM | 673 | | NULL |
| 7/15/09 9:58 AM | 587 | | NULL | 8/18/09 1:14 PM | 643 | Sarcastic | 30 |
| 7/15/09 10:00 AM | 594 | Louis Haußknecht | NULL | 8/18/09 2:28 PM | 690 | msh | 31 |
| 7/15/09 10:02 AM | 597 | elo80ka | 29 | 8/19/09 9:22 AM | 75[illegible] | | 30 |
| 7/15/09 10:21 AM | 618 | Marien | 30 | 8/20/09 3:45 AM | [illegible] | | NULL |
| 7/15/09 10:26 AM | 623 | Fionn | NULL | 8/20/09 12:05 PM | 752 | Sean Taylor | 27 |
| 7/15/09 10:28 AM | 629 | Zoomzoom83 | 26 | 8/22/09 11:58 AM | 580 | n00ki3 | 22 |
| 7/15/09 10:32 AM | 638 | | NULL | 8/26/09 2:43 AM | 851 | Tim Howland | 41 |
| 7/15/09 10:36 AM | 642 | aherrick | 23 | 8/26/09 9:59 AM | 546 | hitec | 27 |
| 7/15/09 10:46 AM | 661 | | NULL | 8/28/09 7:51 PM | 668 | mtruesdell | 34 |

Set Age = 31

© Brent Ozar Unlimited®. All rights reserved.

## Key vs include isn't that important.

As long as your first column or two are kinda unique, then the rest of the keys don't matter as much.

There are indeed edge cases where they matter.

But just focus on these, and you can go a long way:

- The order of the first 1-2 columns
- The columns that you need on the index (whether they're in the key or the includes doesn't matter as much as just having them)

Slide 143
Slides, scripts, db: BrentOzar.com/go/engine

## Generally speaking

Columns in WHERE/GROUP/JOIN/ORDER:
often good candidates for the index keys.

Columns in the SELECT:
good candidates to INCLUDE in an index.
Just don't think that INCLUDES are free: they're not.

Slide 144
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

# Recap (so far)

Slide 145
Slides, scripts, db: BrentOzar.com/go/engine

## Lessons we learned

Less columns, less benefit:
queries usually need more than just 1-2 columns.

More columns, more problems:
bigger indexes, longer maintenance jobs, blocking.

Included columns aren't magically delicious and free.

Slide 146
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

## Where to go next

https://BrentOzar.com/go/engine

Resource links throughout the slides

Fundamentals of Index Tuning: 1-day online class

- Designing indexes for queries
- How SQL Server suggests indexes
- Hands-on homework assignments
- Then, you're ready for…

Mastering Index Tuning: 3-day online class

Slide 147
Slides, scripts, db: BrentOzar.com/go/engine

## Where to go next

Fundamentals of Query Tuning: 1-day online class

- How SQL Server builds query plans
- How to find the right queries to tune
- How parameters influence cached plans
- Common T-SQL anti-patterns
- Hands-on homework assignments
- Then, you're ready for…

Mastering Query Tuning: 3-day online class

Slide 148
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.

Hope you had fun learning,
and thanks for hanging out today!

Slide 149
Slides, scripts, db: BrentOzar.com/go/engine

© Brent Ozar Unlimited®. All rights reserved.