Advanced Common Table Expressions

Presented by Steve Stedman

About the Speaker/Author

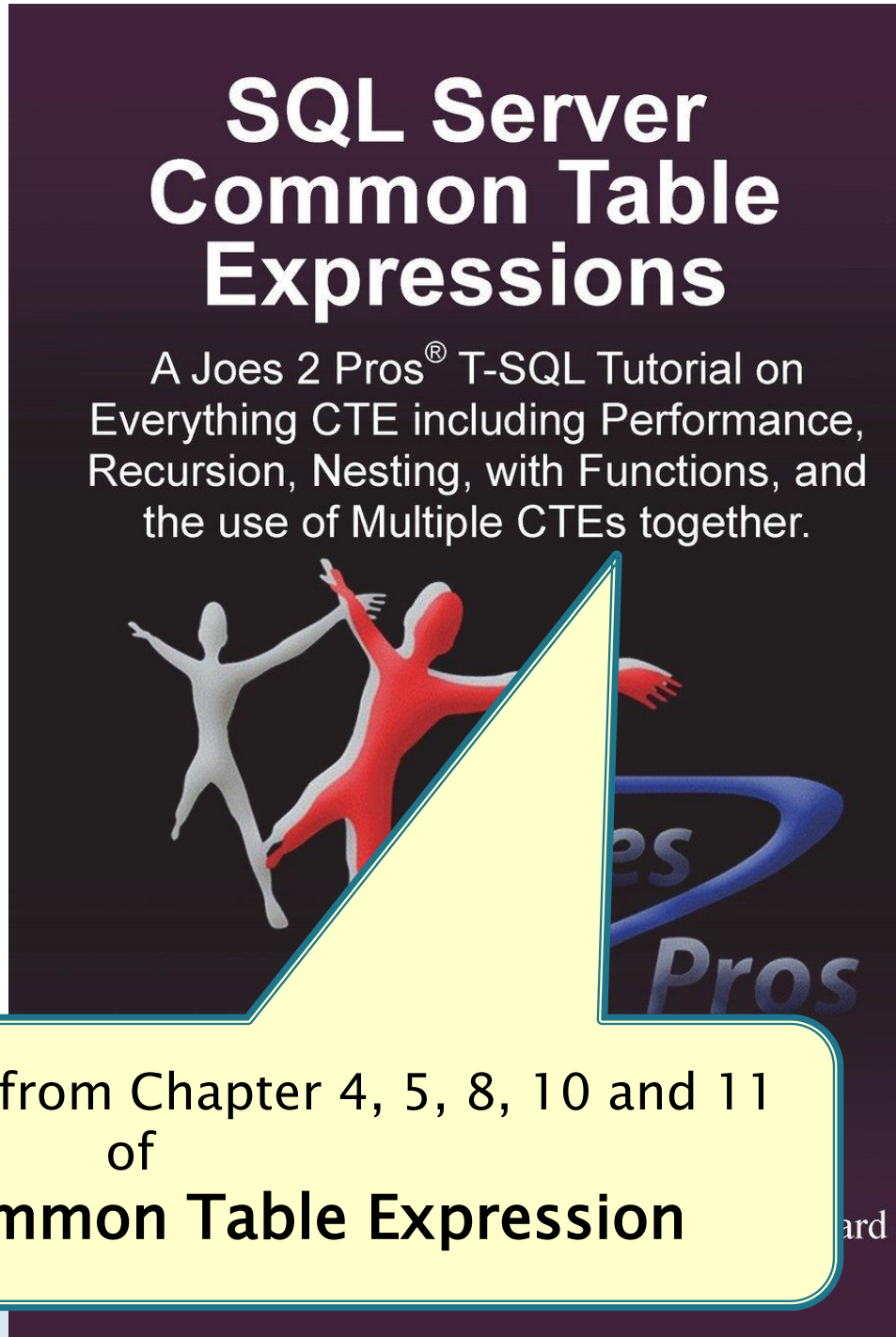
(Steve Stedman)

- Joes2Pros
  - Author of the Common Table Expression Book
  - Instructor at the Joes2Pros Academy
- 23 Years of database work (Microsoft 1990-1991)
- Developer of the Database Health Application
  - http://DatabaseHealth.com
- Working at Emergency Reporting as CTO
- Volunteer Firefighter and EMT

- Twitter: @SqlEmt
- Website: http://SteveStedman.com

Book Giveaway

- Signed copy of the Joes2Pros Common Table Expressions book.
- How to enter: Fill out and turn in the Speaker Evaluation and Raffle form.
- Drawing will be held at the end of the session.



Today's topic comes from Chapter 4, 5, 8, 10 and 11 of

SQL Server Common Table Expression

The Common Table Expression (CTE) is one of the more powerful and often overlooked features in Microsoft SQL Server. This Joes 2 Pros book will show some of the more interesting things you can do with a CTE. We will learn how CTEs are a great alternative to derived table queries, how to do data pasing with a CTE, and recursive
This bool
UPDATE
to use CT
discover
to perfor
The twel
introducti
usage scenarios where a CTE can change the way you look at writing T-SQL queries. Learn how to use a single CTE or multiple CTEs in a single statement as well as nesting them.

SQL Serve

SQL Server Common Table Expressions

Joes 2 Pros

ed
the

Expres

You can download the sample databases at Joes2Pros.com
(Find this link on the homepage)

DOWNLOAD YOUR LABS
(Setup scripts & Answer Keys)

DOWNLOAD YOUR INTERACTIVE LABS FOR EACH BOOK .

PRACTICE LABS

Joes Pros ®

joes2pros.com

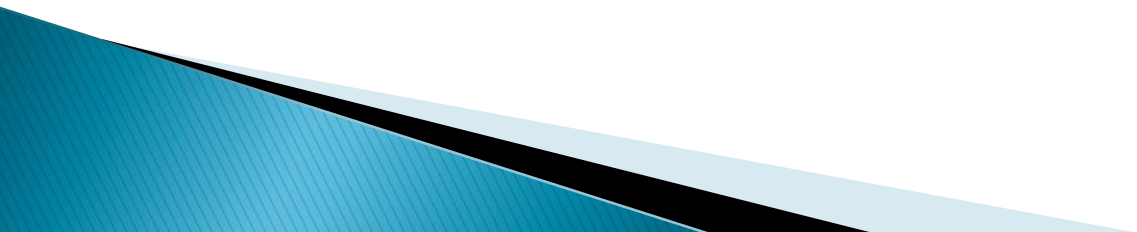
$19.99
ISBN 978-1-939666-18-5
51999>
9 781939 666185

Audience Survey

- How many people have heard of CTEs?
- How many people have used CTEs?
- How many people have used recursive CTEs?
- How many people have deleted data with a CTE?
- How many people use CTEs every day?
- How many people have used recursive CTEs with multiple anchor queries?

CTE – Benefits coming your way!

1. Recursive CTEs.
2. Hierarchical CTEs.
3. Manipulating Data.
4. Common Use Cases.
5. CTE Performance Considerations.



Quick CTE Review

In case you need it.

Simple CTE Syntax

```
;WITH expression_name [(column_name[,...n])]
AS
(
  CTE_query_definition
)

SELECT <column_list>
  FROM expression_name;
```

Simple CTE Syntax

```
;WITH departmentsCTE (id, department, parent)
AS
(
   SELECT id, department, parent
     FROM Departments
)

SELECT *
  FROM departmentsCTE;
```

Chapter 4

Recursion made easy.

4. Recursive CTE

- Considered recursive when the CTE references itself
- Recursion stops
  - When the recursive query produces no results
  - Or specify MAXRECURSION
- Uses
  - Hierarchical listing of categories
  - Recursive calculations
  - Much, much more…

Recursion Overview

- Sum the numbers from 1 to 10 without recursion

55 = 10 + 9 + 8 + 7 + 6 + 5 + 4 +3 + 2 + 1

- Sum the numbers from 1 to 10 recursively

55 = 10 + (sum of numbers 1 to 9)

55 = 10 + (9 + (sum of numbers 1 to 8))

55 = 10 + (9 + (8 + (sum of numbers 1 to 7)))

- Eventually we get to:

55 = 10 + (9 + (8 + (7 + (6 + (5 + (4 + (3 + (2 + 1)))))))

Recursive Terminology

- Anchor Query
  - Start the recursion
  - One or more anchor queries
- Recursive Query
  - The part that repeats
  - One or more recursive queries
- MAXRECURSION
  - The number of times to repeat the recursive query
  - Default is 100
  - MAXRECURSION of 0 implies no maximum

Example of How a Recursive CTE Works

1. Select some starting set of data from table A.
2. Join that starting set of data to table A.
3. For the results from step 2, join that to Table A.
4. Repeat until there are no more items produced by the join.

Demo: Recursive CTE

```
;WITH DepartmentCTE(DeptId, Department, Parent, Lvl)
AS
```

Step 1. Declare the CTE and Columns

Demo: Recursive CTE

```
;WITH DepartmentCTE(DeptId, Department, Parent, Lvl)
AS
( SELECT id AS DeptId, Department, parent, 0 AS Lvl
    FROM Departments
  WHERE parent IS NULL
```

Step 2 – Add the Anchor Query

Demo: Recursive CTE

```
;WITH DepartmentCTE(DeptId, Department, Parent, Lvl)
AS
( SELECT id AS DeptId, Department, parent, 0 AS Lvl
    FROM Departments
  WHERE parent IS NULL
   UNION ALL
```

Step 3 – Add the UNION ALL to connect to the recursive query

Demo: Recursive CTE

```
;WITH DepartmentCTE(DeptId, Department, Parent, Lvl)
AS
( SELECT id AS DeptId, Department, parent, 0 AS Lvl
    FROM Departments
  WHERE parent IS NULL
   UNION ALL -- and now for the recursive part
  SELECT d.id AS DeptId, d.Department, d.parent,
                DepartmentCTE.Lvl + 1 AS Lvl
    FROM Departments d
   INNER JOIN DepartmentCTE
           ON DepartmentCTE.DeptId = d.parent)
```

Step 4 – Add the recursive Query

Demo: Recursive CTE

```
;WITH DepartmentCTE(DeptId, Department, Parent, Lvl)
AS
( SELECT id AS DeptId, Department, parent, 0 AS Lvl
    FROM Departments
  WHERE parent IS NULL
   UNION ALL -- and now for the recursive part
  SELECT d.id AS DeptId, d.Department, d.parent,
                DepartmentCTE.Lvl + 1 AS Lvl
    FROM Departments d
   INNER JOIN DepartmentCTE
         ON DepartmentCTE.DeptId = d.parent)
SELECT *
  FROM DepartmentCTE
 ORDER BY parent;
```

Demo

Chapter 4

Recursive CTE Notes

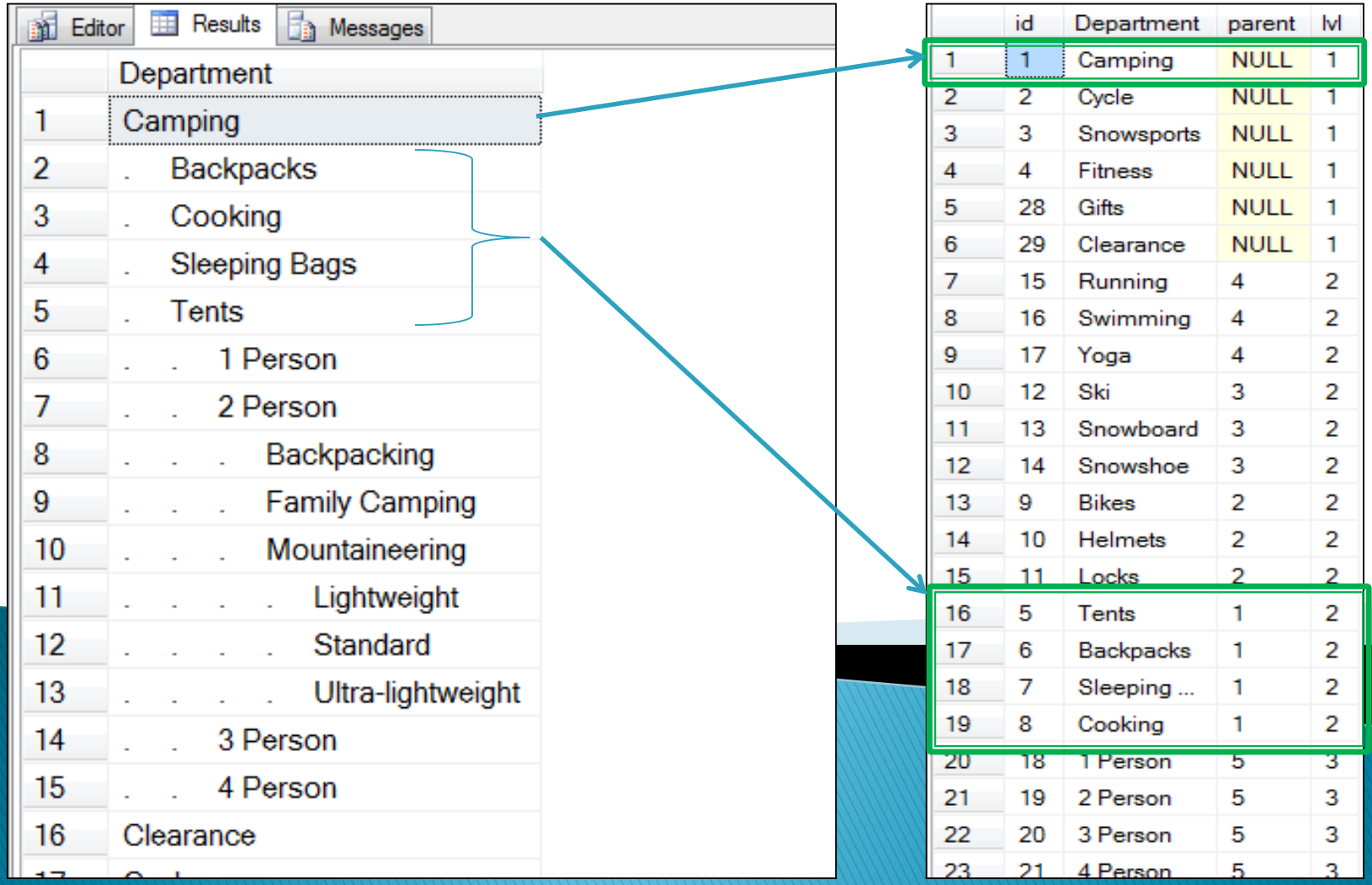
- Recursion stops
  - When the recursive query produces no results
  - Or specify MAXRECURSION
- Using TSQL functions for recursion allows for 32 levels of recursion
- Using CTE for recursion allows for 32767 levels of recursion in the MAXRECURSION option, but much more if you set MAXRECURSION to 0.
  - I have confirmed up to 100,000,000 levels of recursion.

Chapter 5

Recursion made to look good.

Hierarchical CTEs

Chapter 5 – Recursion made to look good – Hierarchical CTEs.

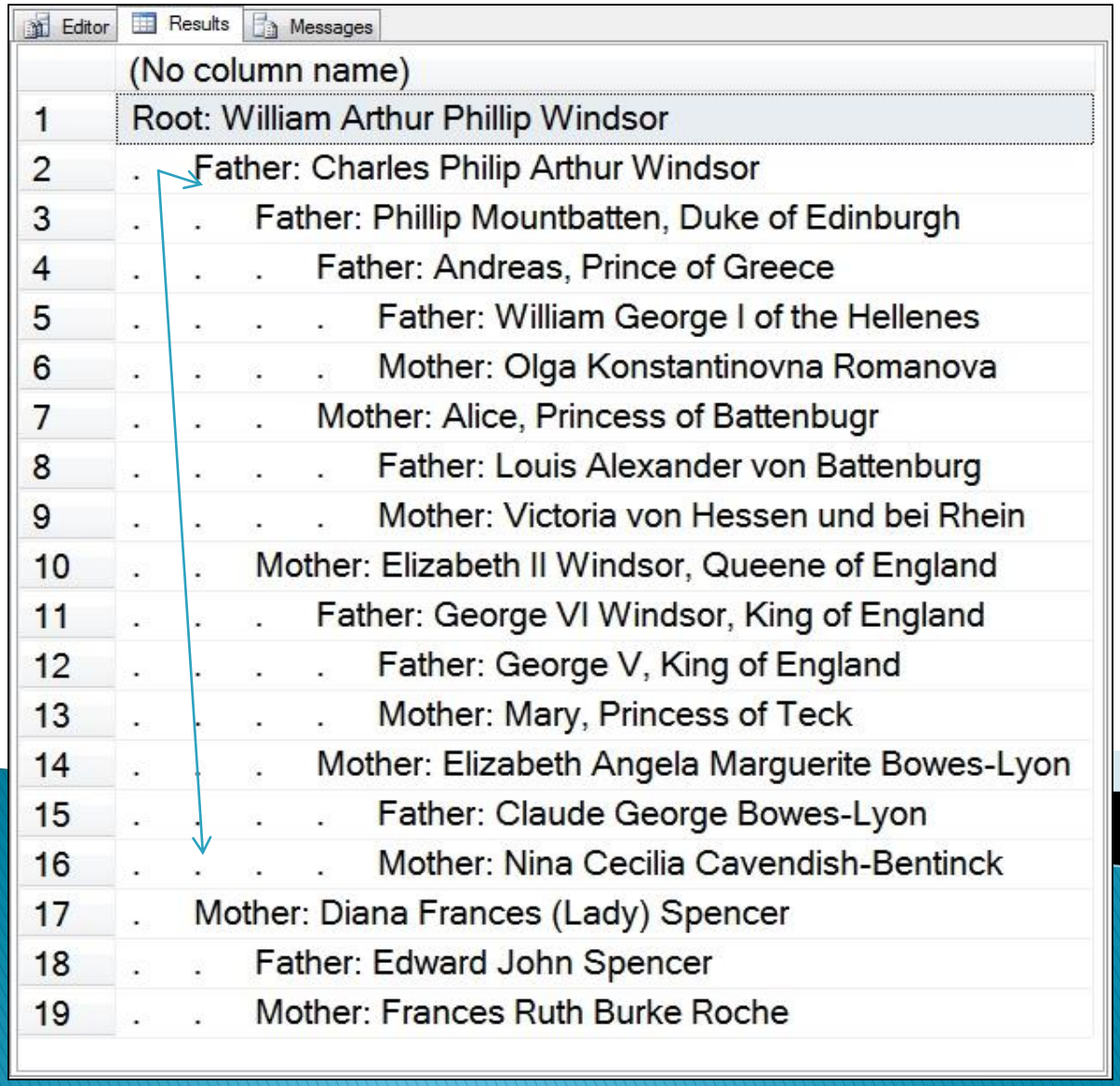


Editor | Results | Messages

	Department
1	Camping
2	. Backpacks
3	. Cooking
4	. Sleeping Bags
5	. Tents
6	. . 1 Person
7	. . 2 Person
8	. . . Backpacking
9	. . . Family Camping
10	. . . Mountaineering
11	. . . . Lightweight
12	. . . . Standard
13	. . . . Ultra-lightweight
14	. . 3 Person
15	. . 4 Person
16	Clearance

	id	Department	parent	lvl
1	1	Camping	NULL	1
2	2	Cycle	NULL	1
3	3	Snowsports	NULL	1
4	4	Fitness	NULL	1
5	28	Gifts	NULL	1
6	29	Clearance	NULL	1
7	15	Running	4	2
8	16	Swimming	4	2
9	17	Yoga	4	2
10	12	Ski	3	2
11	13	Snowboard	3	2
12	14	Snowshoe	3	2
13	9	Bikes	2	2
14	10	Helmets	2	2
15	11	Locks	2	2
16	5	Tents	1	2
17	6	Backpacks	1	2
18	7	Sleeping ...	1	2
19	8	Cooking	1	2
20	18	1 Person	5	3
21	19	2 Person	5	3
22	20	3 Person	5	3
23	21	4 Person	5	3

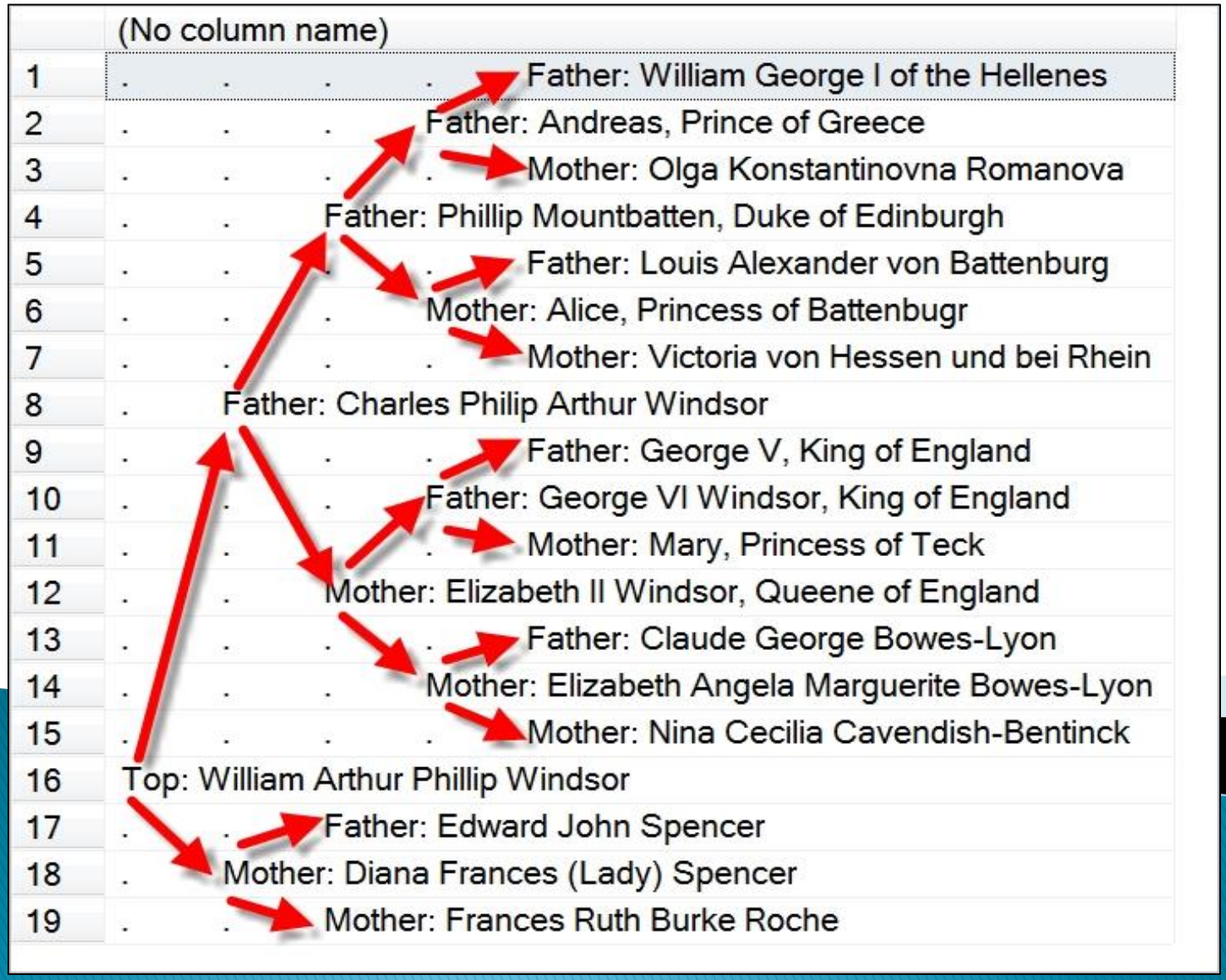
Hierarchy in Multiple Recursive Queries possible.



Editor | Results | Messages

	(No column name)
1	Root: William Arthur Phillip Windsor
2	. Father: Charles Philip Arthur Windsor
3	. . Father: Phillip Mountbatten, Duke of Edinburgh
4	. . . Father: Andreas, Prince of Greece
5	. . . . Father: William George I of the Hellenes
6	. . . . Mother: Olga Konstantinovna Romanova
7	. . . Mother: Alice, Princess of Battenbugr
8	. . . . Father: Louis Alexander von Battenburg
9	. . . . Mother: Victoria von Hessen und bei Rhein
10	. . Mother: Elizabeth II Windsor, Queene of England
11	. . . Father: George VI Windsor, King of England
12	. . . . Father: George V, King of England
13	. . . . Mother: Mary, Princess of Teck
14	. . . Mother: Elizabeth Angela Marguerite Bowes-Lyon
15	. . . . Father: Claude George Bowes-Lyon
16	. . . . Mother: Nina Cecilia Cavendish-Bentinck
17	. Mother: Diana Frances (Lady) Spencer
18	. . Father: Edward John Spencer
19	. . Mother: Frances Ruth Burke Roche

Hierarchy with advanced formatting.



	(No column name)
1	. . . . Father: William George I of the Hellenes
2	. . . Father: Andreas, Prince of Greece
3	. . . . Mother: Olga Konstantinovna Romanova
4	. . Father: Phillip Mountbatten, Duke of Edinburgh
5	. . . . Father: Louis Alexander von Battenburg
6	. . . Mother: Alice, Princess of Battenbugr
7	. . . . Mother: Victoria von Hessen und bei Rhein
8	. Father: Charles Philip Arthur Windsor
9	. . . . Father: George V, King of England
10	. . . Father: George VI Windsor, King of England
11	. . . . Mother: Mary, Princess of Teck
12	. . Mother: Elizabeth II Windsor, Queene of England
13	. . . . Father: Claude George Bowes-Lyon
14	. . . Mother: Elizabeth Angela Marguerite Bowes-Lyon
15	. . . . Mother: Nina Cecilia Cavendish-Bentinck
16	Top: William Arthur Phillip Windsor
17	. . Father: Edward John Spencer
18	. Mother: Diana Frances (Lady) Spencer
19	. . Mother: Frances Ruth Burke Roche

Demo

Chapter 5

Chapter 8

Manipulating Data:

Insert

Update

Delete

Chapter 8 – DELETE

```
DELETE FROM CTE;
```

1. Are the following SQL Statements valid?
2. Can you delete from a CTE?
3. What does that mean?

```
;WITH CustomerCTE AS
(
  SELECT *
    FROM Customer
   WHERE LastName like 'Williams'
)
DELETE FROM CustomerCTE;
```

```
;WITH CustomerCTE AS
(
 SELECT c.*
   FROM Customer AS c
  INNER JOIN SalesInvoice AS si
     ON si.CustomerID = c.CustomerID
  WHERE c.LastName like 'Williams'
)
DELETE FROM CustomerCTE;
```

Demo

Chapter 8 –

DELETE FROM CTE;

Chapter 8 – INSERT

INSERT INTO CTE;

1. Are the following SQL Statements valid?
2. Can you INSERT INTO a CTE?
3. What does that mean?

```
;WITH CustomerCTE AS
(
  SELECT *
    FROM Customer
   WHERE LastName like 'Stedman'
)
INSERT INTO CustomerCTE
   (CustomerID,FirstName,LastName)
VALUES (99999, 'Steve', 'Stedman');
```

```
;WITH CustomerCTE AS
(
 SELECT c.*
   FROM Customer AS c
  INNER JOIN SalesInvoice AS si
     ON si.CustomerID = c.CustomerID
  WHERE c.LastName like 'Stedman'
)
INSERT INTO CustomerCTE
   (CustomerID, FirstName, LastName)
VALUES (99999, 'Steve', 'Stedman');
```

Demo

Chapter 8 –

INSERT INTO CTE;

Chapter 8 – UPDATE

UPDATE CTE;

1. Are the following SQL Statements valid?
2. Can you UPDATE a CTE?
3. What does that mean?

```
;WITH CustomerCTE AS
(
  SELECT c.*
    FROM Customer AS c
   WHERE c.LastName like 'Williams'
)
UPDATE CustomerCTE
   SET LastName = 'Willie';
```

```
;WITH CustomerCTE AS
(
 SELECT c.*, si.Comment
   FROM Customer AS c
  INNER JOIN SalesInvoice AS si
     ON si.CustomerID = c.CustomerID
  WHERE c.LastName like 'Williams'
)
UPDATE CustomerCTE
   SET Comment = 'Some Comment',
       CompanyName = 'Willies Toys';
```

Demo

Chapter 8 – UPDATE CTE;

INSERT, UPDATE, DELETE Notes

- Insert, Update, and Delete all work against a CTE with only a single base table.
- Delete does not work for any CTE with multiple base tables referenced in the CTE query.
- Insert and Update works against a CTE with multiple base tables as long as only one base table is being udpated.

Chapter 10

Common Use Cases:

Alternative to a Numbers table

Finding Holes

Scrubbing duplicates

Alternative to a Numbers Table.

- **Demo**

```
;WITH Numbers (N) AS
(
  SELECT 1
   UNION ALL
  SELECT 1 + N FROM Numbers
   WHERE N < 1000
)
SELECT *
  FROM Numbers
OPTION (MAXRECURSION 1000);
```

Finding Holes.

- **Demo**

```
;WITH Numbers (N) AS
(  SELECT 0
   UNION ALL
  SELECT 1 + N FROM Numbers
   WHERE N < 23)
, OrderHours (HourOfDay, TheCount) AS
(  SELECT DATEPART(HOUR, OrderDate) as HourOfDay,
          COUNT(1) AS TheCount
    FROM SalesInvoice
   WHERE OrderDate < '02/01/2006'
   GROUP BY DATEPART(HOUR, OrderDate)   )
SELECT n.N AS HourOfDay, ISNULL(oh.TheCount, 0) OrderCount
  FROM OrderHours oh
 RIGHT JOIN Numbers n ON n.N = oh.HourOfDay
 ORDER BY TheCount ASC;
```

Scrubbing duplicates.

- **Demo**

```
WITH CustomerCTE AS
(
  SELECT *,
  ROW_NUMBER() OVER(PARTITION BY LastName,
                    FirstName ORDER BY CustomerID) AS DupNum
  FROM Customer
)
DELETE
  FROM CustomerCTE
 WHERE DupNum > 1;
```

Chapter 11

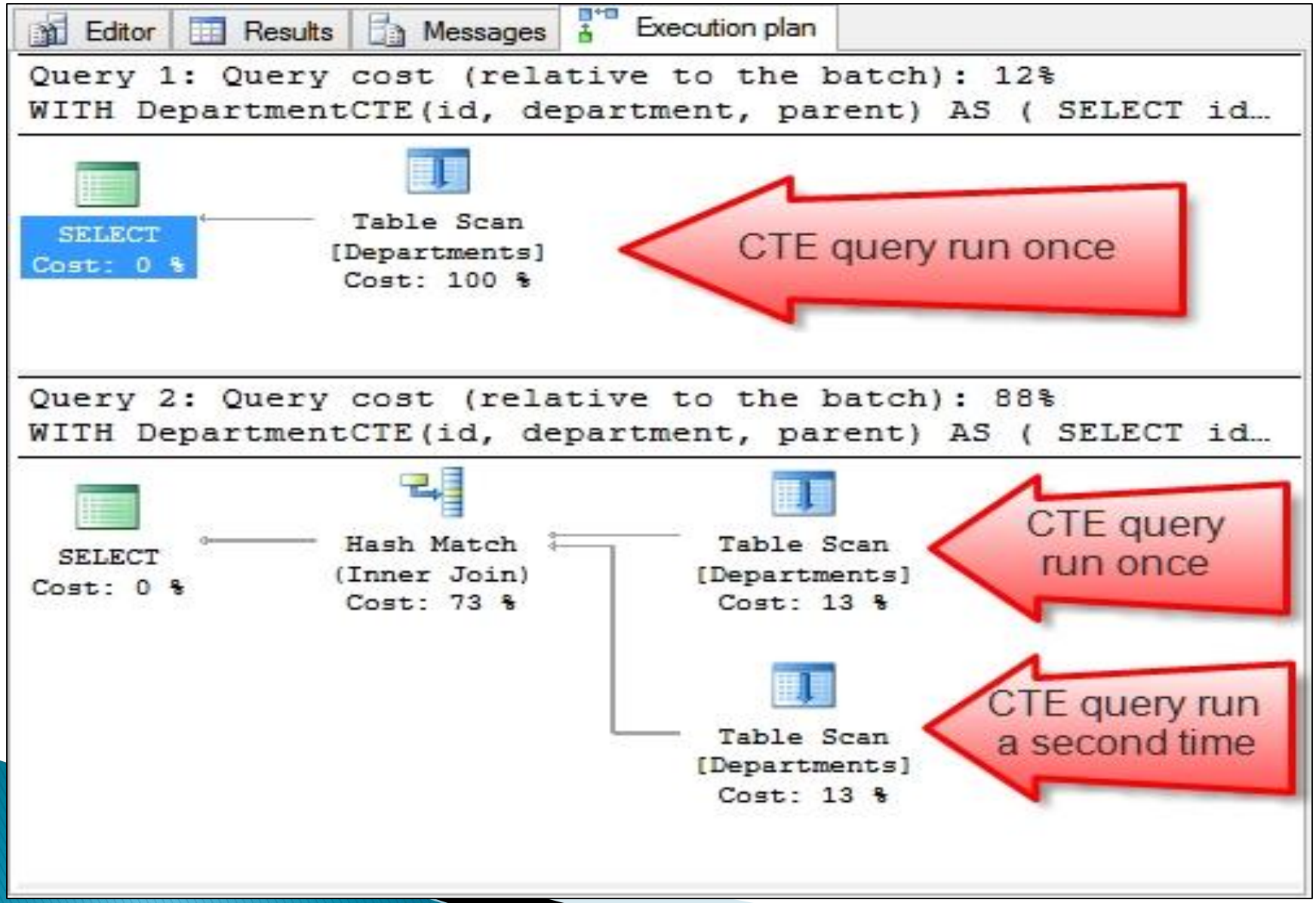
The need for speed

Understanding CTE performance

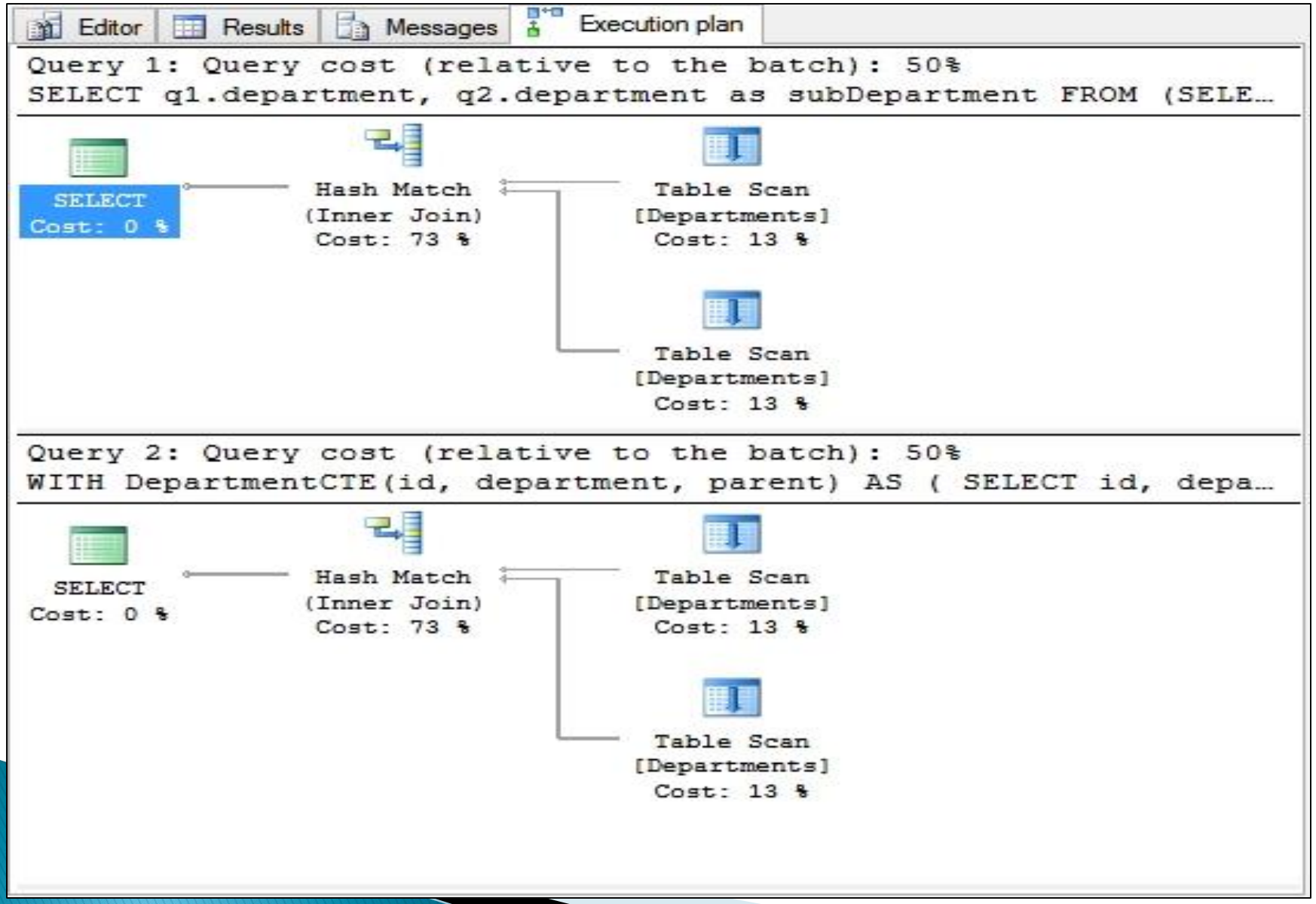
11. CTE Performance

- Non-Recursive Performance
  - Multiple references to a single CTE
  - CTEs vs. Derived Tables
  - CTEs vs. Views
  - Multiple CTEs in a query
- Recursive Performance
  - Deep Recursion

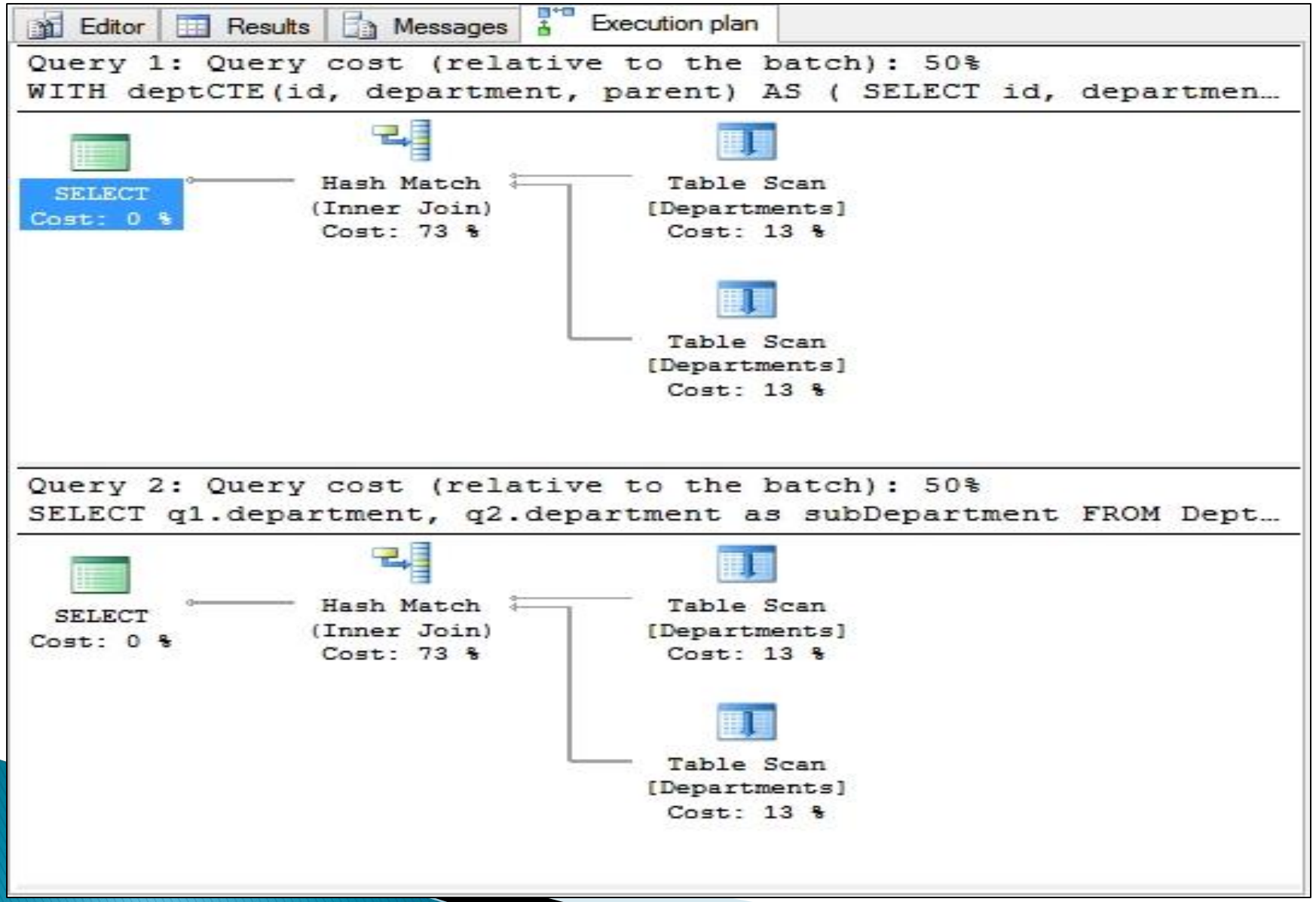
Multiple references to a single CTE



CTEs vs. Derived Tables



CTEs vs. Views



Multiple CTEs in a query

Nested CTEs

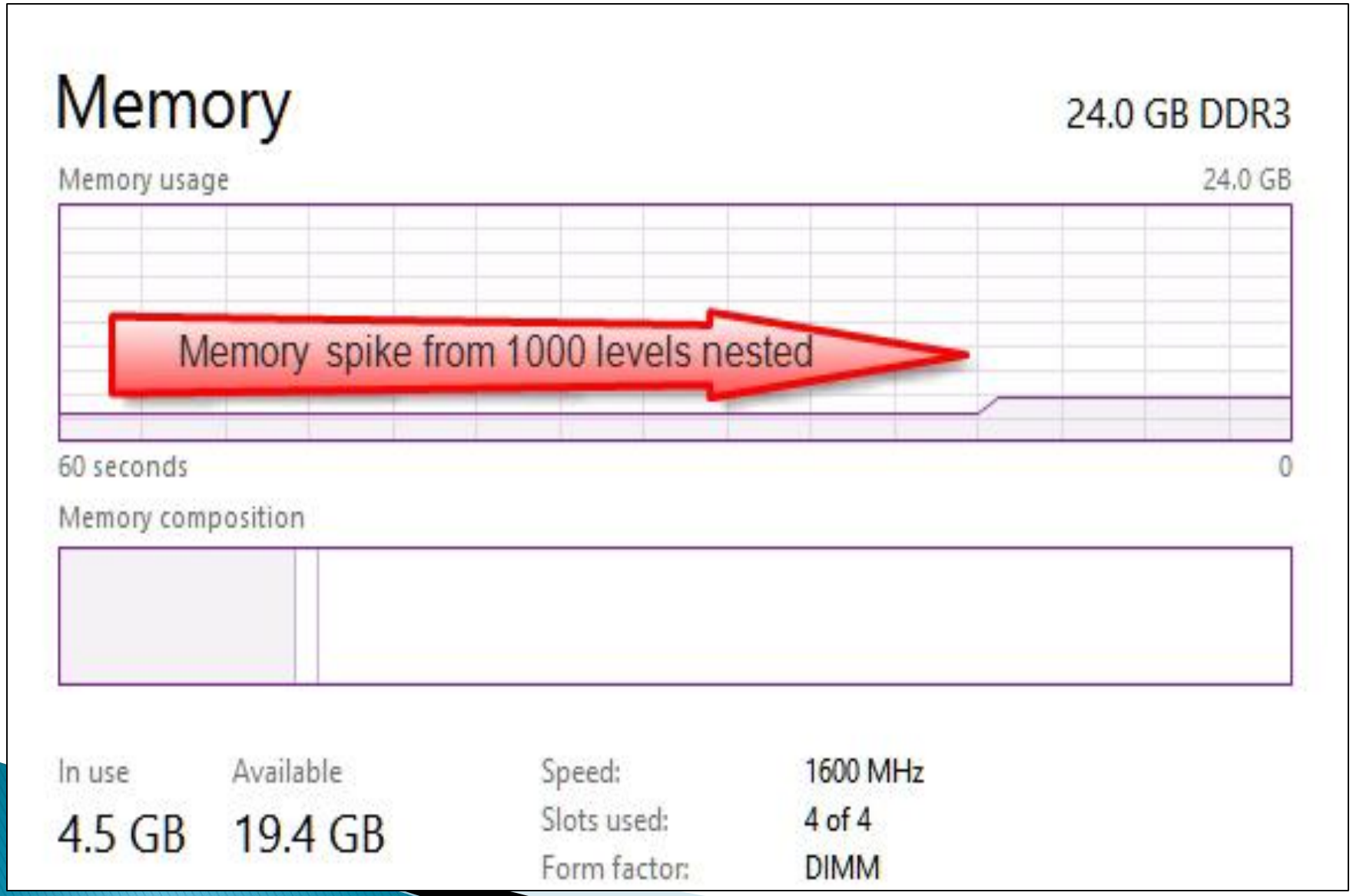
- Take it to the extreme

```
with cte0 as
(  select 1 as num )
, cte1 AS (SELECT * FROM cte0)
, cte2 AS (SELECT * FROM cte1)
```

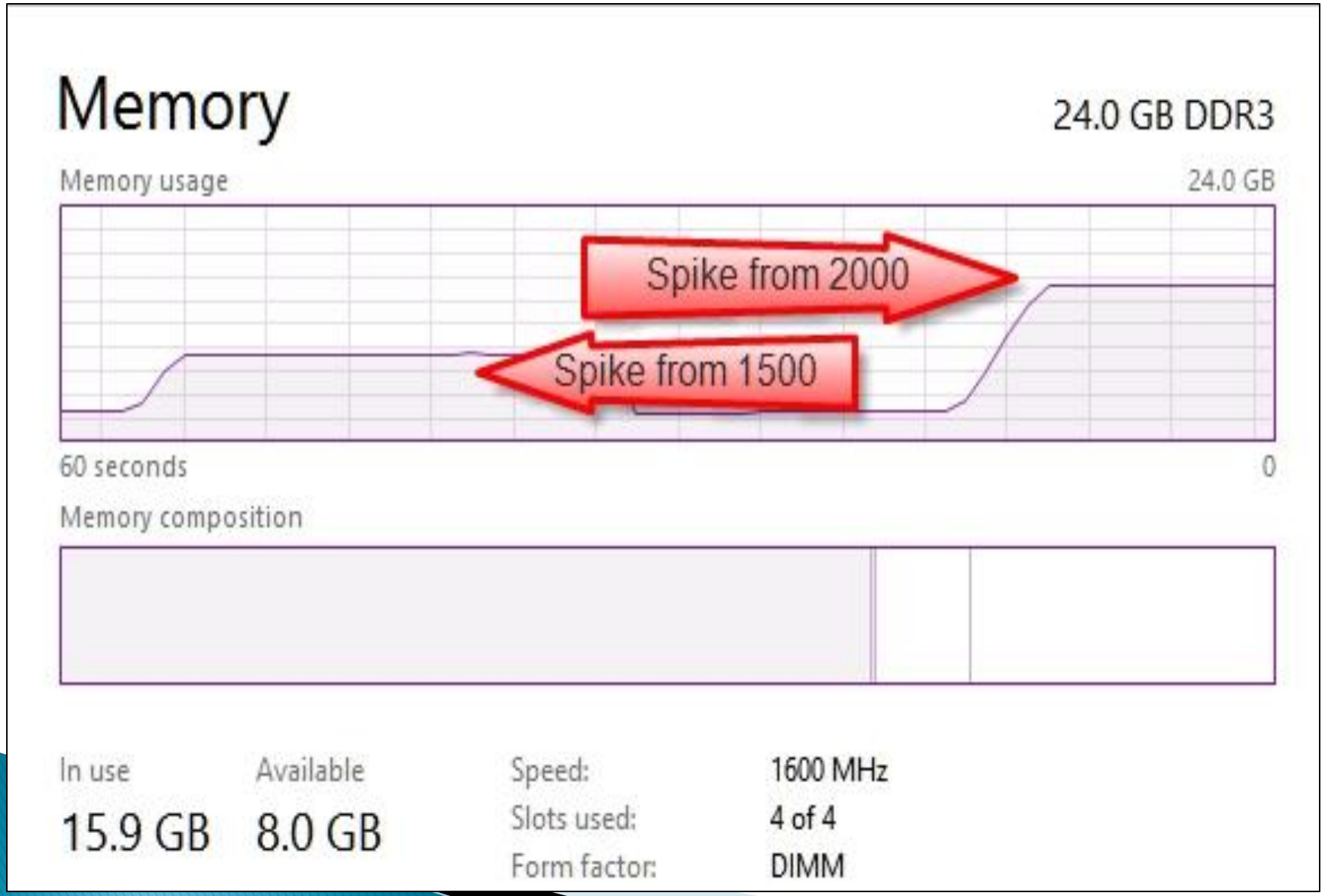
Repeated from 2 to 254.

```
, cte255 AS (SELECT * FROM cte254)
, cte256 AS (SELECT * FROM cte255)
, cte257 AS (SELECT * FROM cte256)
select * from cte257;
```

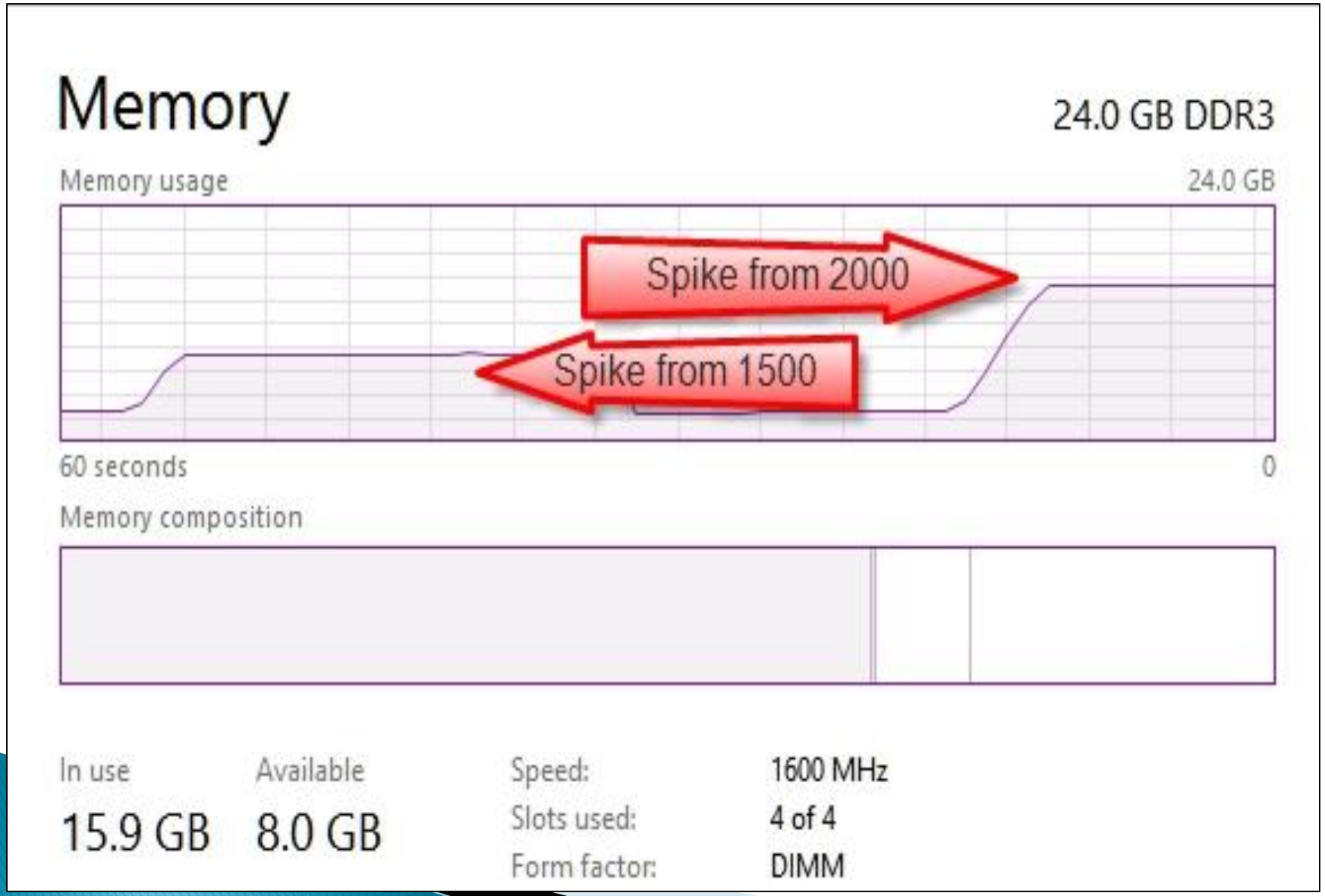
Nested CTEs


Memory
24.0 GB DDR3
Memory usage
24.0 GB
Memory spike from 1000 levels nested
60 seconds
0
Memory composition
In use
4.5 GB
Available
19.4 GB
Speed:
1600 MHz
Slots used:
4 of 4
Form factor:
DIMM

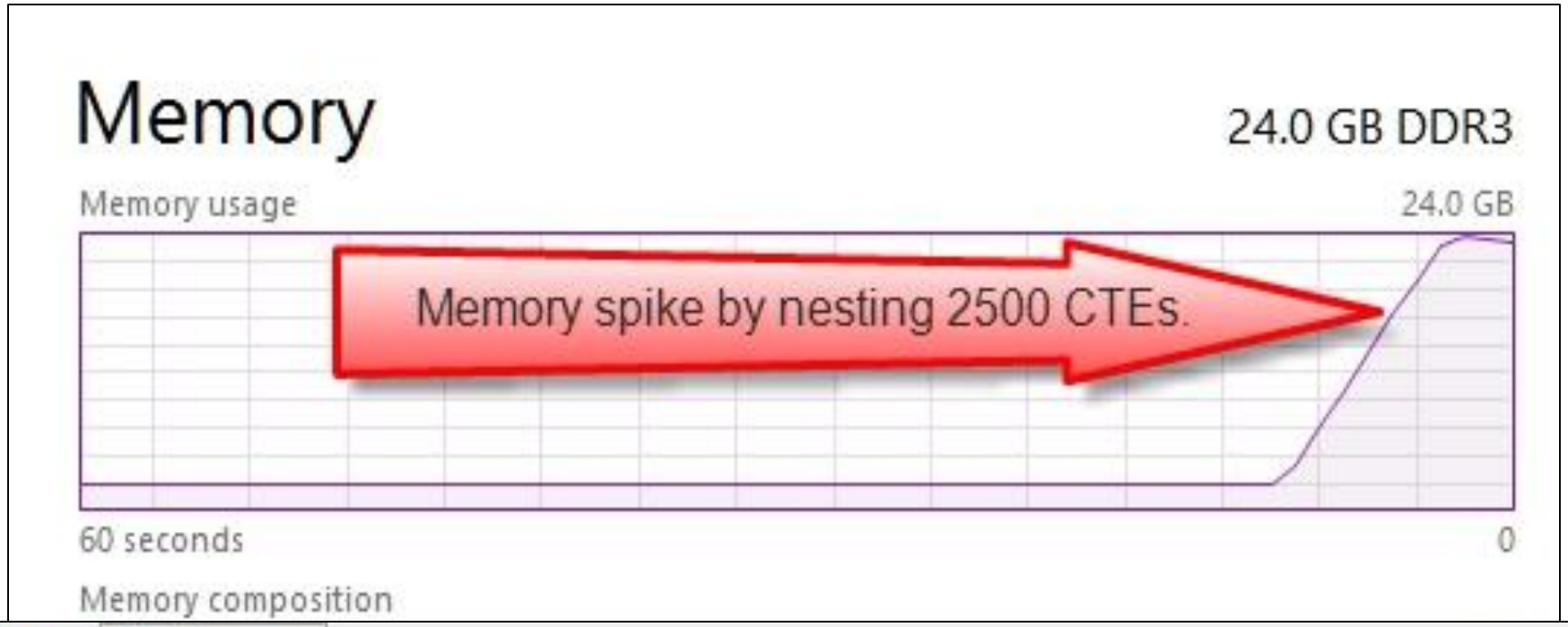
Nested CTEs


Memory
24.0 GB DDR3
Memory usage
24.0 GB
Spike from 2000
Spike from 1500
60 seconds
0
Memory composition
In use
15.9 GB
Available
8.0 GB
Speed:
1600 MHz
Slots used:
4 of 4
Form factor:
DIMM

Nested CTEs


Memory
24.0 GB DDR3
Memory usage
24.0 GB
Spike from 2000
Spike from 1500
60 seconds
0
Memory composition
In use
15.9 GB
Available
8.0 GB
Speed:
1600 MHz
Slots used:
4 of 4
Form factor:
DIMM

Nested CTEs

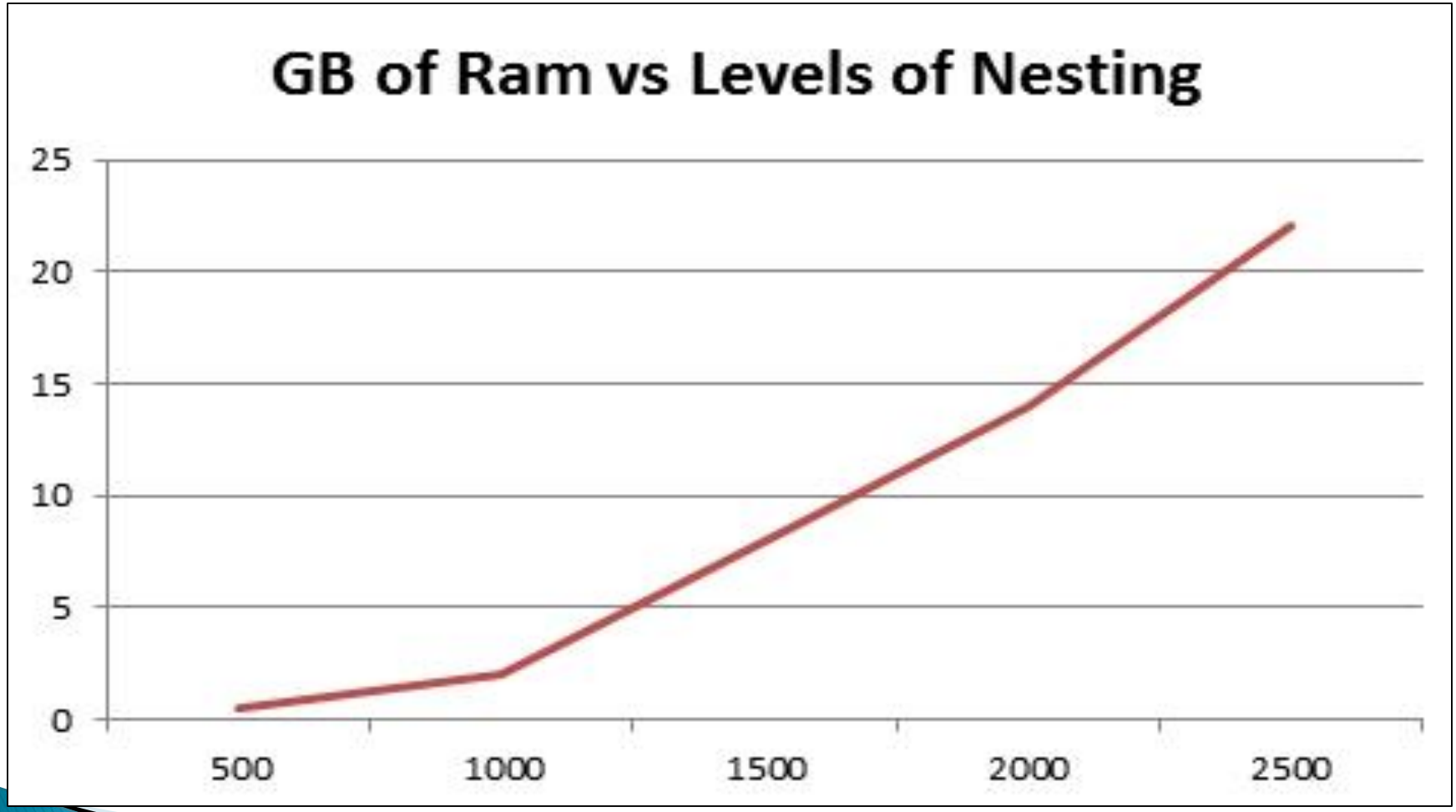


Editor | Messages

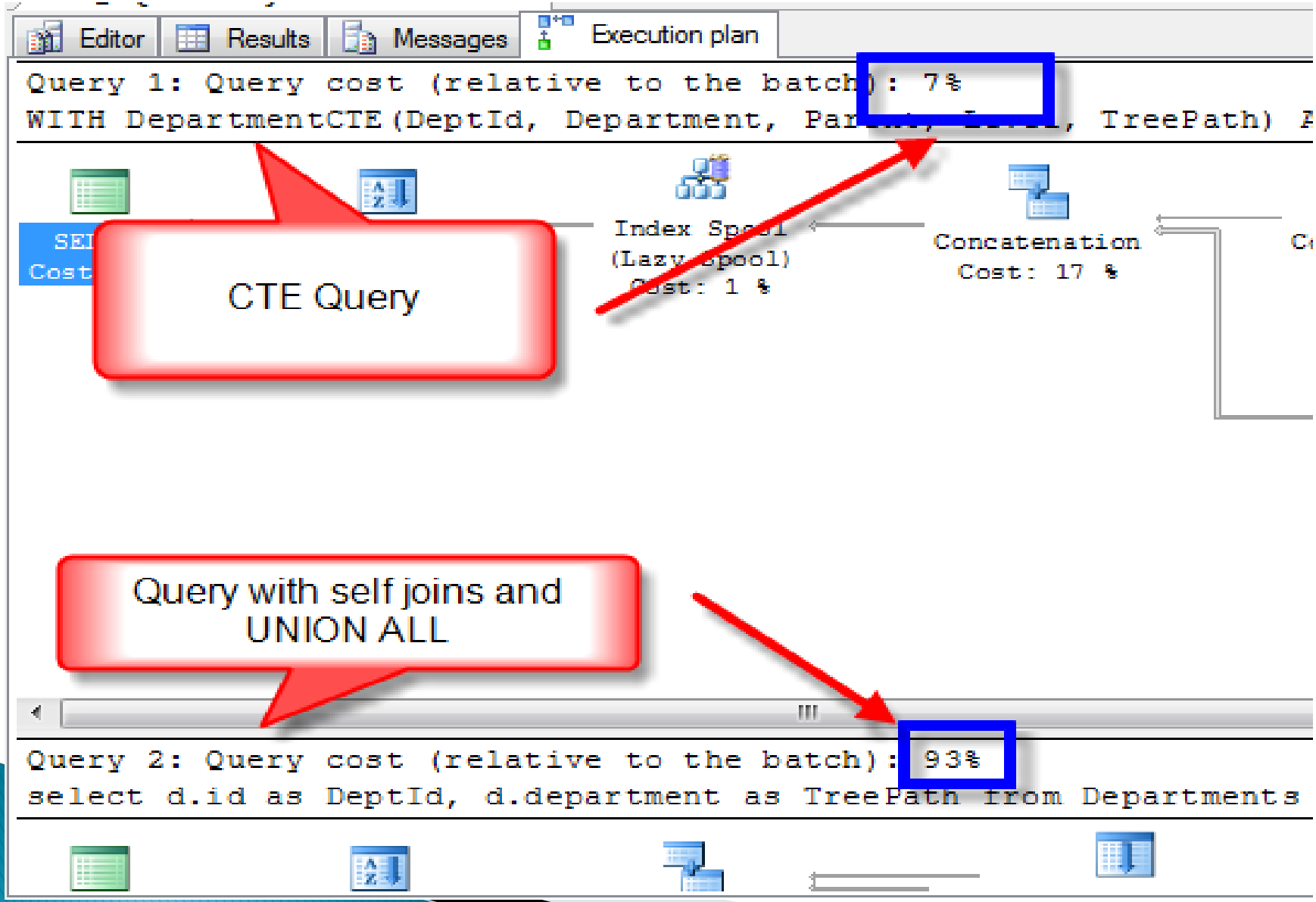
```
Msg 701, Level 17, State 123, Line 2
There is insufficient system memory in resource pool 'default' to run this query.
```

100 %

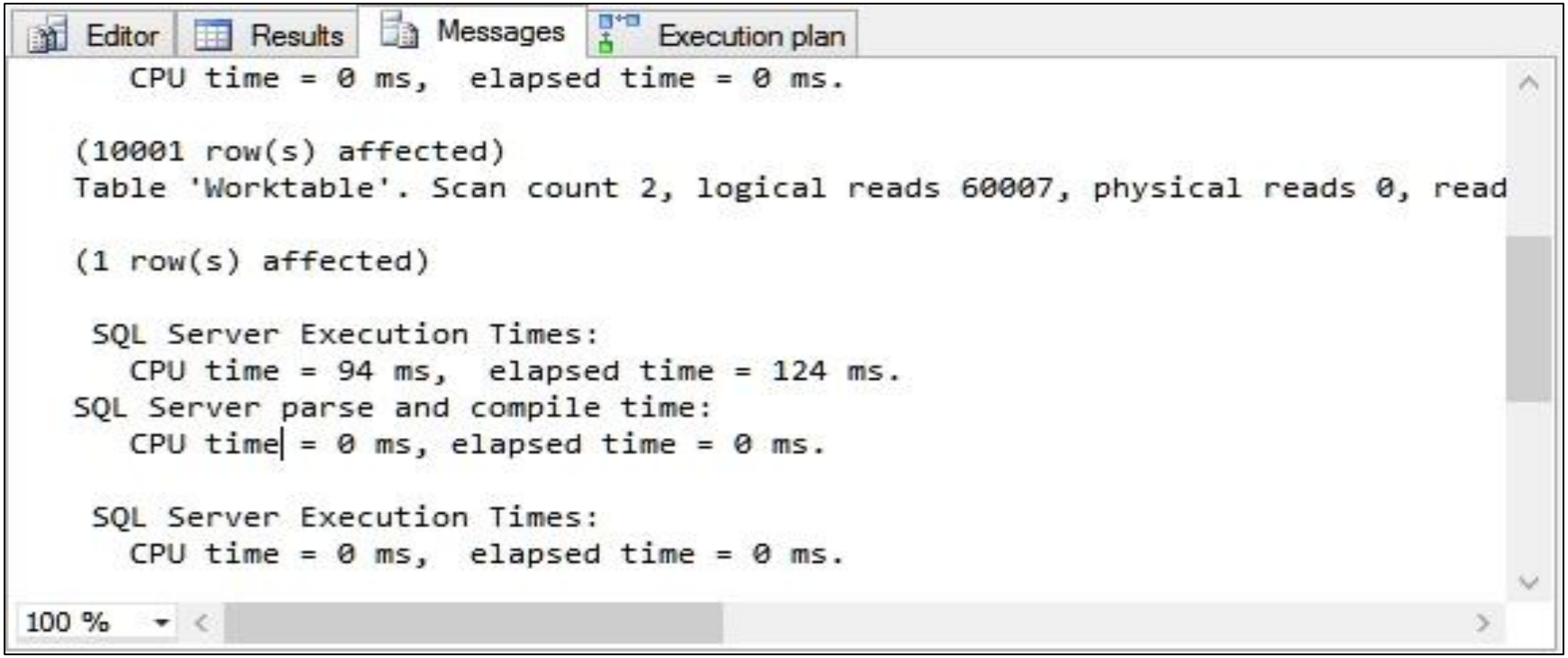
Recursive Performance



Recursive Performance



Deep Recursion



Demo

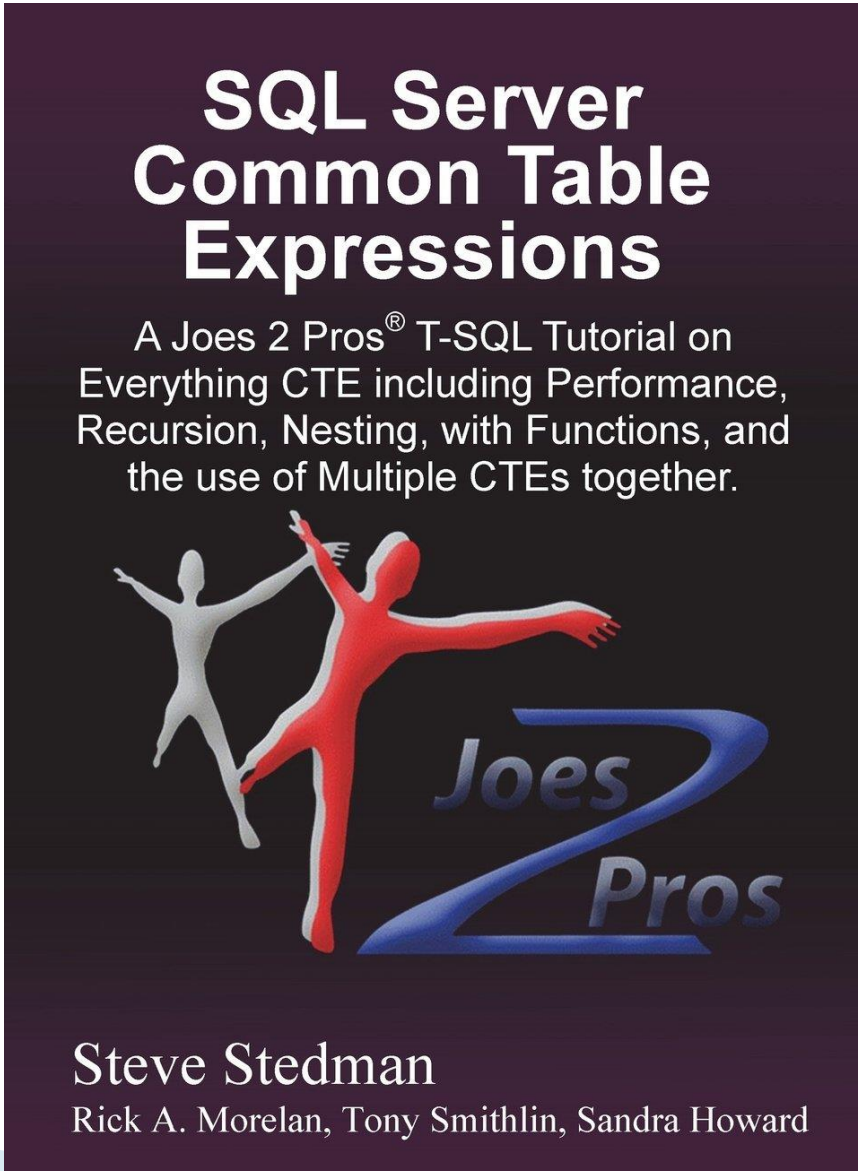
Chapter 11

More Information

- Follow me on Twitter
  - **@SqlEmt**
- Database Health Project
  - http://DatabaseHealth.com
- Visit my website
  - http://stevestedman.com
- Send me an email:
  - Steve@SteveStedman.com
- Download Slides and Sample TSQL
  - http://stevestedman.com/speaking/

Common Table Expressions Book Giveaway

- Published May 2013
- Available at Amazon.com and at Joes2Pros.com
- Print and Kindle versions both available.



CTE – Fact or Fiction

Steve Stedman

Debunking common myths about
Common Table Expressions

1. CTE Executions

As a named result set, the CTE is only run once even if it is referenced multiple times in a query.

True or False?

FALSE The CTE is executed once for EACH time that it is referenced in a query.

1. CTE Executions Explained

```
;WITH deptCTE(id, department, parent) AS
(SELECT id, department, parent
    FROM Departments)
SELECT q1.department, q2.department
  FROM deptCTE q1
 INNER JOIN deptCTE q2 on q1.id = q2.parent
 WHERE q1.parent is null;
```

- In this example the deptCTE is executed twice

2. CTEs are proprietary

CTEs are proprietary to Microsoft SQL Server.

True or False?

FALSE Common Table Expressions are supported by several major database platforms, among them PostgreSQL, DB2, Oracle and SQL Server, defined in SQL-99 spec

3. CTE and Hierarchical Queries

CTEs are a great way to create recursive hierarchical queries.

True or False?

TRUE Recursive hierarchical queries are easy to write with a CTE. CTE's save time, are easy to follow, and work great for hierarchical data.

5. Database Versions

SQL Server only supports CTE's on SQL Server Enterprise Edition 2008R2 and newer.

True or False?

FALSE Common Table Expressions have been supported since SQL Server 2005 and are available in all versions.

6. Stored Procedures and Functions

CTEs can be defined in user-defined routines, such as functions, stored procedures, triggers, or views.

True or False?

TRUE Common Table Expressions can be defined and used inside of stored procedures and functions.

7. CTEs and Nesting

CTEs can be nested and one CTE can reference an earlier CTE.

True or False?

TRUE Common Table Expressions can be nested. Just define multiple CTE's and reference an earlier CTE from a later one.

8. Indexing CTEs

Indexes can be added to CTEs to boost performance.

True or False?

FALSE A Common Table Expression is a temporary, "inline" view – you cannot add an index to a CTE.

9. VIEW vs CTE

Which performs better, a non-recursive CTE or a VIEW?

They are the same.

- The big gain is the recursive CTE, which you can't achieve with a view.

10. CTE's and Data Paging

CTE's are a great way to do Data Paging for a result grid.

True or False

It Depends……

SQL Server 2012 has the new OFFSET and FETCH clause on select statements, which is easier than CTE's. For 2005, 2008 and 2008R2 the CTE is the best option.

11. CTE's performance

Recursive CTE's perform the same as other pseudo recursive solutions?

True or False

FALSE……

12. CTE's and TempDB

CTE's are similar to Temp Tables or Table Variables in their use of TempDB?

True or False

FALSE Temp Tables and Table Variables both use TempDB, CTE's do not……

- See my blog posting for all the details on this one.
  - http://stevestedman.com/?p=2053
  - It is more than we have time to prove today.

13. Data Paging

An alternative to a CTE would be to use the ROW_NUMBER function in the WHERE clause to filter the results.

True or False?

FALSE ROW_NUMBER can be used to get the current row number in the result set, but it is a windowing function, and windowing functions are not allowed to be used in the WHERE clause.

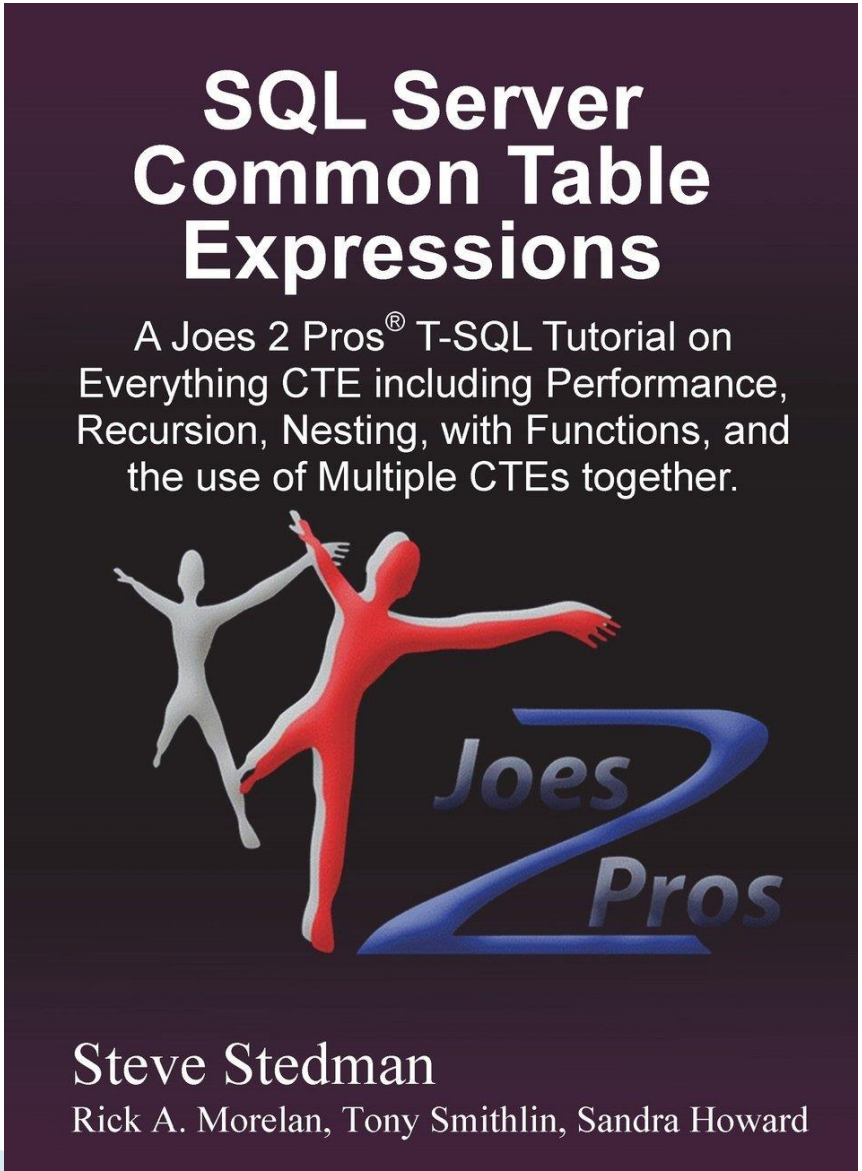
More Information

- Follow me on Twitter
  - **@SqlEmt**
- Database Health Project
  - http://DatabaseHealth.com
- Visit my website
  - http://stevestedman.com
- Send me an email:
  - Steve@SteveStedman.com
- Download Slides and Sample TSQL
  - http://stevestedman.com/speaking/

Common Table Expressions Book

- Published May 2013
- Available at Amazon.com and at Joes2Pros.com

- Print and Kindle versions both available.



Thank you to all of our Sponsors!

- Platinum Sponsors

- Gold Sponsors

- Marquee Sponsor

Thank you to all of our Sponsors!

- Silver Sponsors

- Bronze Sponsor

- Blog Sponsor

Thank you to all of our Sponsors!

- SWAG Sponsors