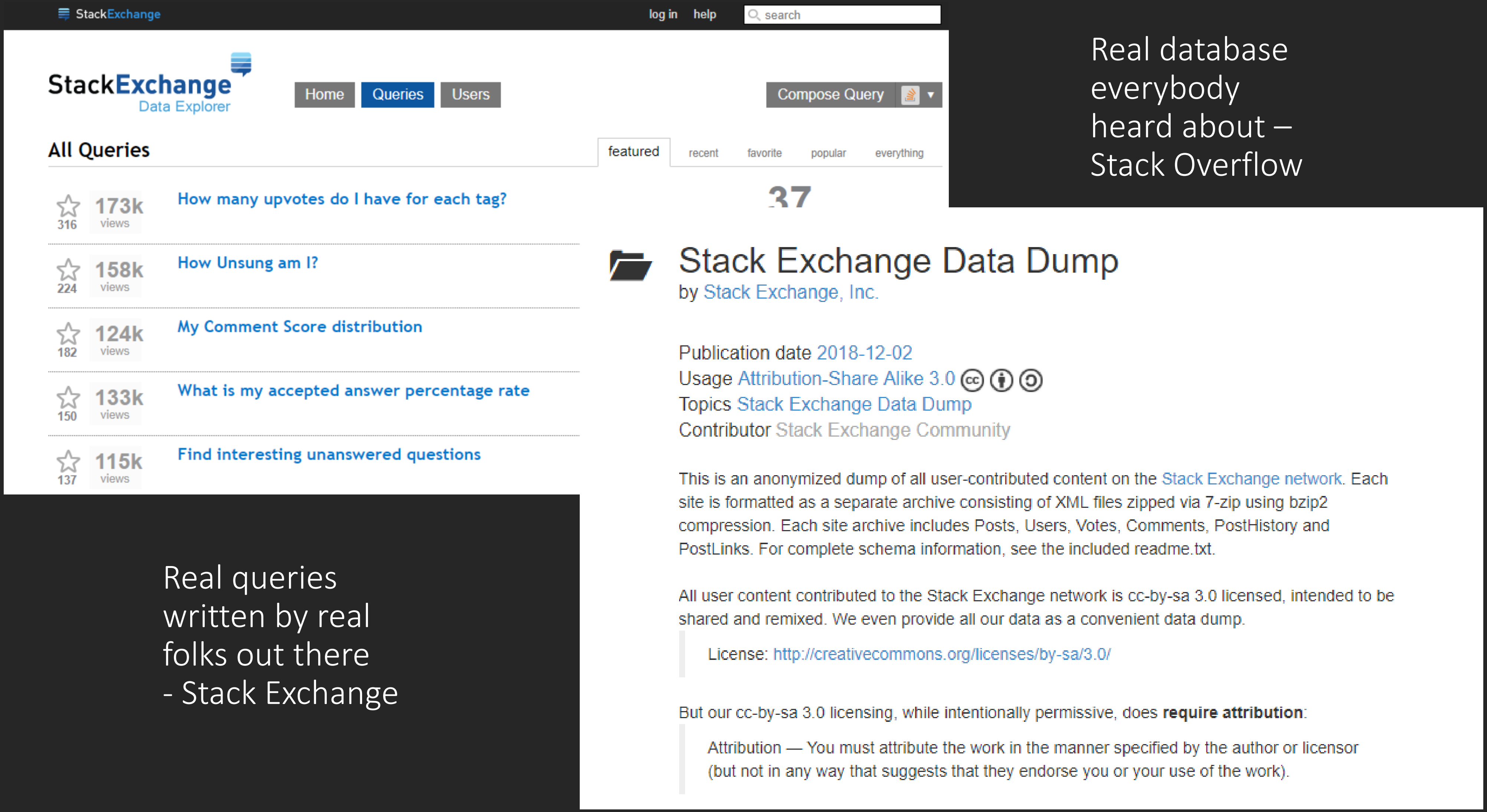
Noda Lóránd - 2019

Quick Start to SQL Server Performance Tuning

What to expect?

- demos a lot of demos
- with real data by using the Stack Overflow database
- ways of checking the execution of a query in SSMS
- types of query tunings
- frequently occurring issues in a query execution

Origin of the demo content:



All Queries

316	173k views	How many upvotes do I have for each tag?
224	158k views	How Unsung am I?
182	124k views	My Comment Score distribution
150	133k views	What is my accepted answer percentage rate
137	115k views	Find interesting unanswered questions

37

Stack Exchange Data Dump

by Stack Exchange, Inc.

Publication date 2018-12-02
Usage Attribution-Share Alike 3.0
Topics Stack Exchange Data Dump
Contributor Stack Exchange Community

This is an anonymized dump of all user-contributed content on the Stack Exchange network. Each site is formatted as a separate archive consisting of XML files zipped via 7-zip using bzip2 compression. Each site archive includes Posts, Users, Votes, Comments, PostHistory and PostLinks. For complete schema information, see the included readme.txt.

All user content contributed to the Stack Exchange network is cc-by-sa 3.0 licensed, intended to be shared and remixed. We even provide all our data as a convenient data dump.

> License: http://creativecommons.org/licenses/by-sa/3.0/

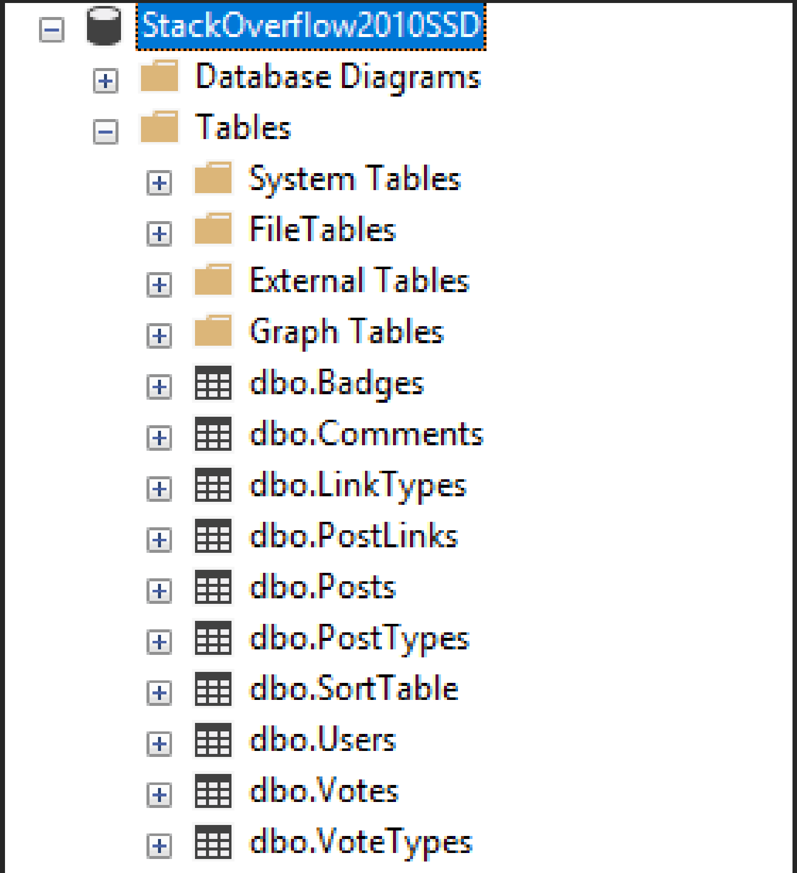
But our cc-by-sa 3.0 licensing, while intentionally permissive, does **require attribution**:

> Attribution — You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).

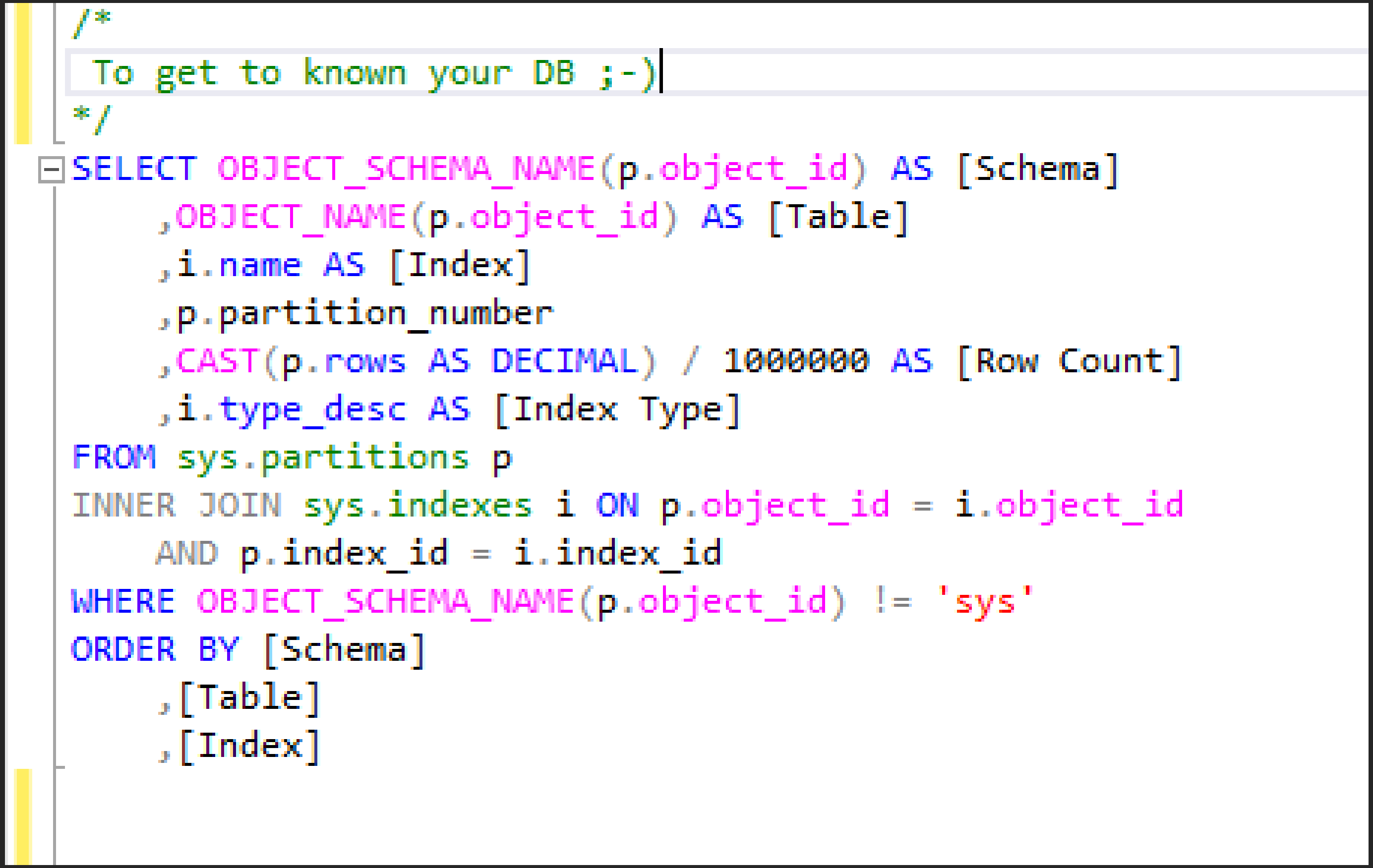
Real database
everybody
heard about –
Stack Overflow

Real queries
written by real
folks out there
- Stack Exchange

Demo: DB we’re going to use:



- StackOverflow2010SSD
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - Graph Tables
    - dbo.Badges
    - dbo.Comments
    - dbo.LinkTypes
    - dbo.PostLinks
    - dbo.Posts
    - dbo.PostTypes
    - dbo.SortTable
    - dbo.Users
    - dbo.Votes
    - dbo.VoteTypes



```sql
/*
 To get to known your DB ;-)
*/
SELECT OBJECT_SCHEMA_NAME(p.object_id) AS [Schema]
    ,OBJECT_NAME(p.object_id) AS [Table]
    ,i.name AS [Index]
    ,p.partition_number
    ,CAST(p.rows AS DECIMAL) / 1000000 AS [Row Count]
    ,i.type_desc AS [Index Type]
FROM sys.partitions p
INNER JOIN sys.indexes i ON p.object_id = i.object_id
    AND p.index_id = i.index_id
WHERE OBJECT_SCHEMA_NAME(p.object_id) != 'sys'
ORDER BY [Schema]
    ,[Table]
    ,[Index]
```

Query Tuning scenarios:

- Queries where you can't change the query
  - Index Tuning
  - Server Tuning

- Queries where you can't add/change indexes => rarely
  - in most cases this is the hardest
  - smart logic for writing something new

- No limitations ;-)

Okay, but how to check what is happening?

- SET STATISTICS IO, TIME ON;
- Estimated Execution plan (based on statistics)
- Actual Execution plan

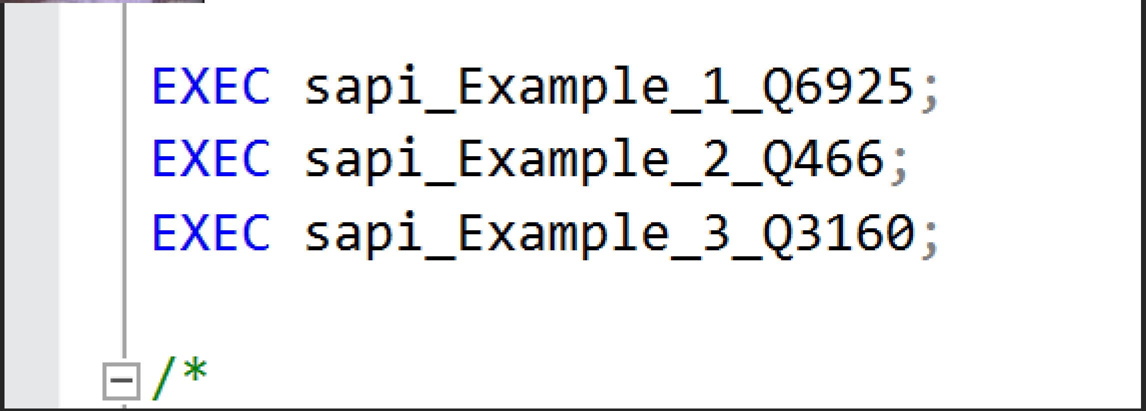
!! IMPORTANT: the estimated plan should relatively match with the actual one otherwise that is a sign of something bad happening in the background

- Live Query Statistics
- Profiler
- Extended events - not going to touch this

But first a quick statistics example :

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

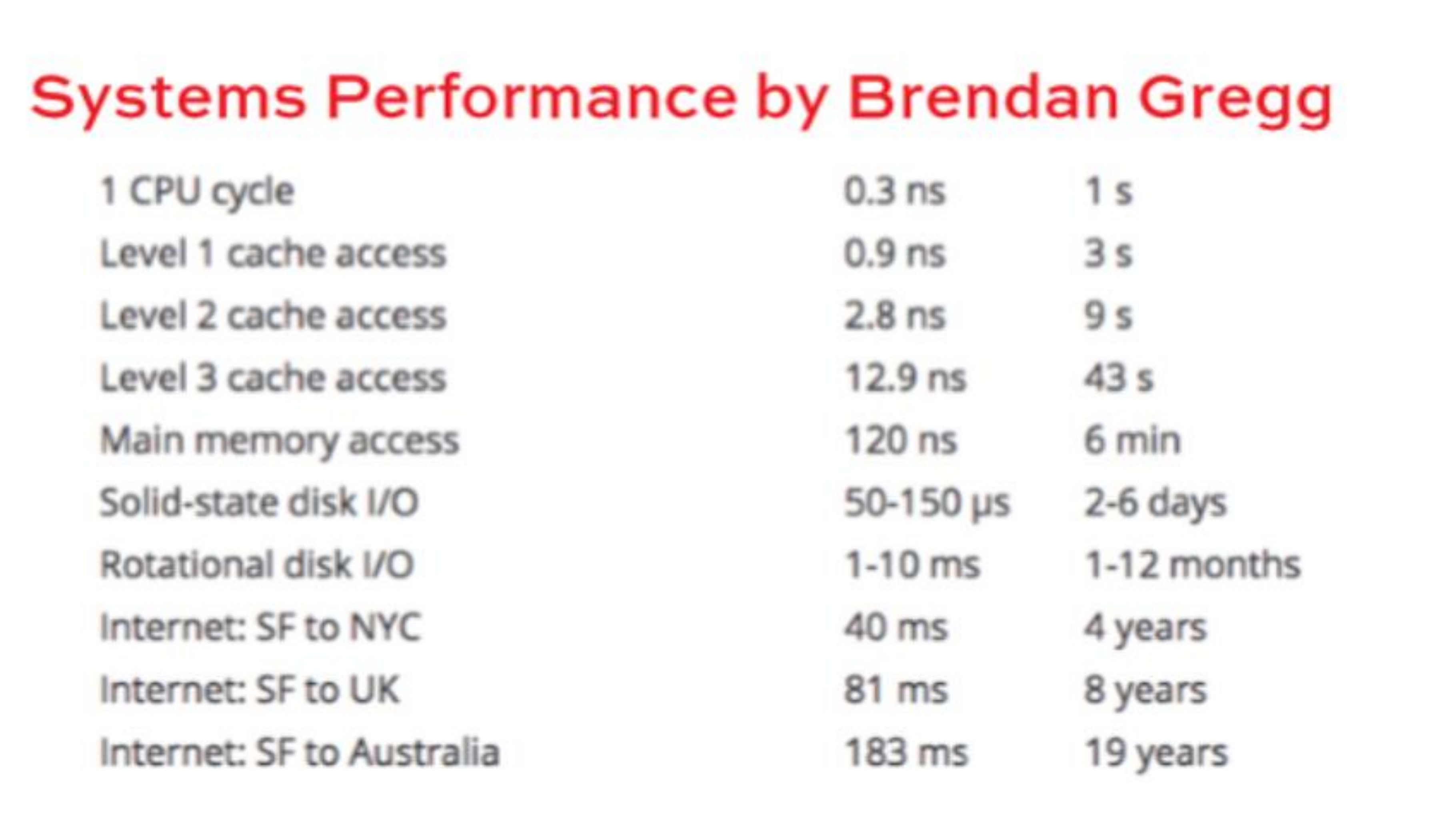
Going to demo the execution
with these procs:



```
EXEC sapi_Example_1_Q6925;
EXEC sapi_Example_2_Q466;
EXEC sapi_Example_3_Q3160;

/*
```

A quick server tuning demo + the following :



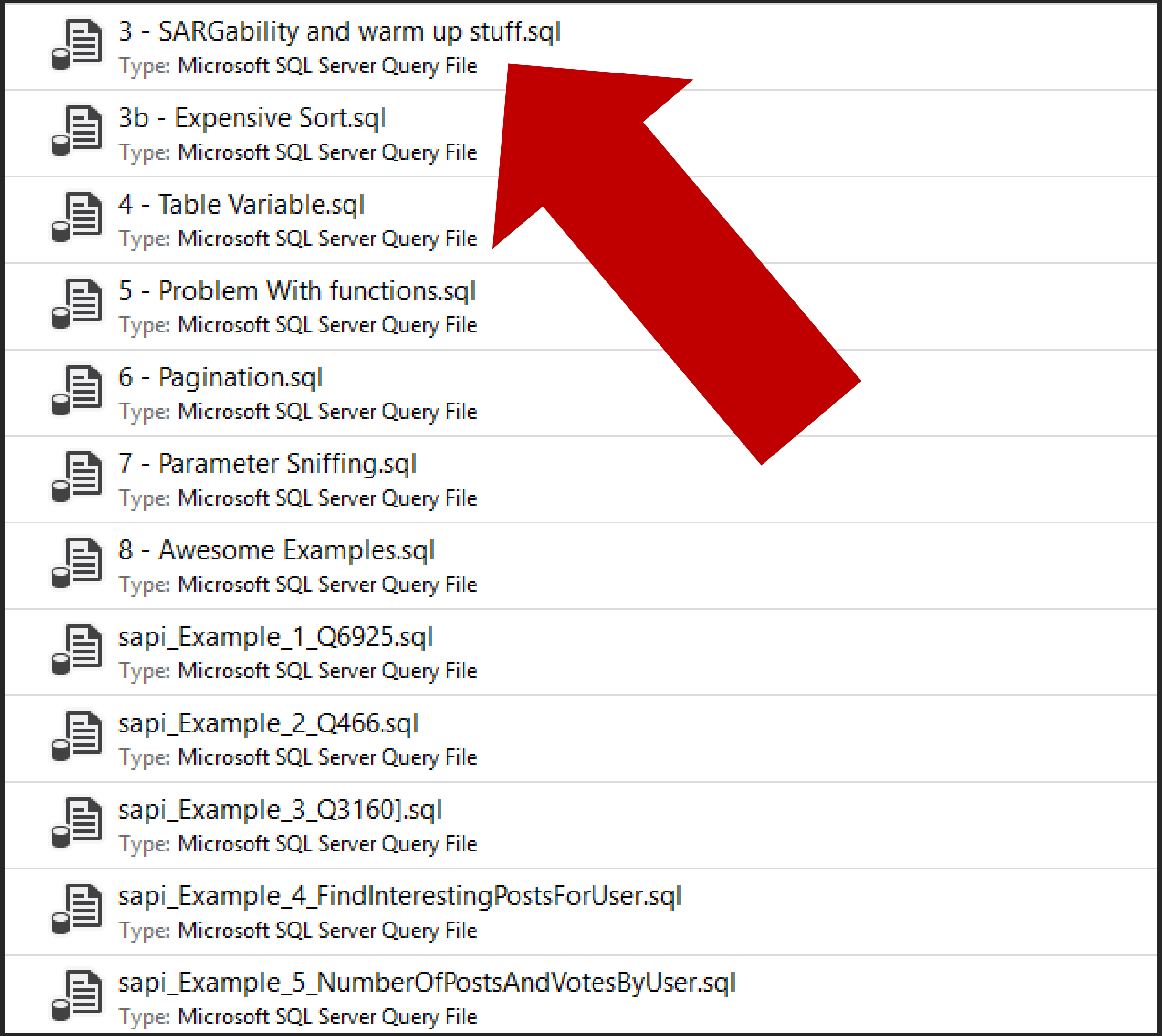
Systems Performance by Brendan Gregg

1 CPU cycle	0.3 ns	1 s
Level 1 cache access	0.9 ns	3 s
Level 2 cache access	2.8 ns	9 s
Level 3 cache access	12.9 ns	43 s
Main memory access	120 ns	6 min
Solid-state disk I/O	50-150 μs	2-6 days
Rotational disk I/O	1-10 ms	1-12 months
Internet: SF to NYC	40 ms	4 years
Internet: SF to UK	81 ms	8 years
Internet: SF to Australia	183 ms	19 years

Query Issues: The easy stuff

- Missing INDEX
- SARGability
- Implicit Conversion
- Expensive Sort
- Expensive Key Lookup

Demo round one:



3 - SARGability and warm up stuff.sql
Type: Microsoft SQL Server Query File

3b - Expensive Sort.sql
Type: Microsoft SQL Server Query File

4 - Table Variable.sql
Type: Microsoft SQL Server Query File

5 - Problem With functions.sql
Type: Microsoft SQL Server Query File

6 - Pagination.sql
Type: Microsoft SQL Server Query File

7 - Parameter Sniffing.sql
Type: Microsoft SQL Server Query File

8 - Awesome Examples.sql
Type: Microsoft SQL Server Query File

sapi_Example_1_Q6925.sql
Type: Microsoft SQL Server Query File

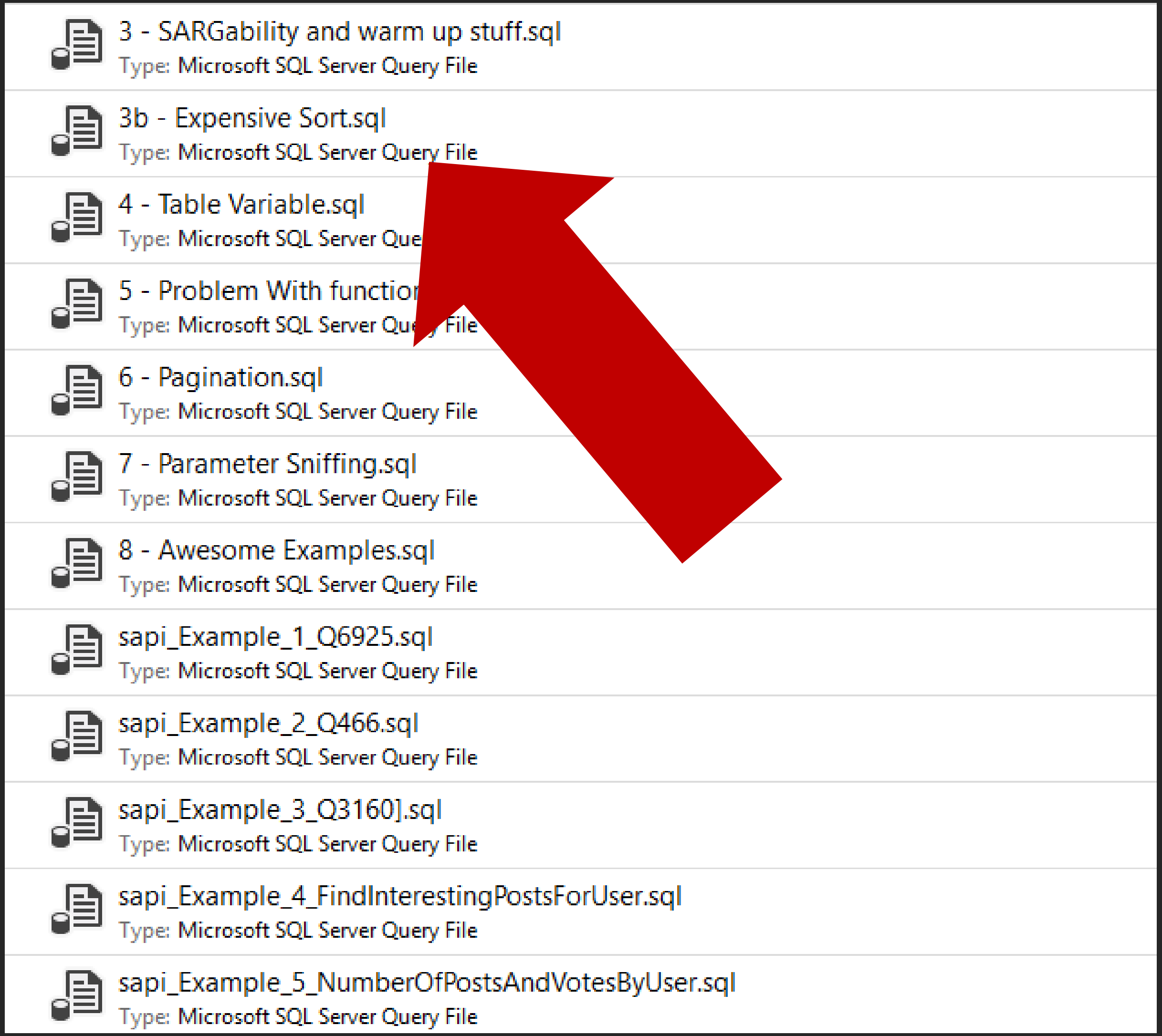
sapi_Example_2_Q466.sql
Type: Microsoft SQL Server Query File

sapi_Example_3_Q3160].sql
Type: Microsoft SQL Server Query File

sapi_Example_4_FindInterestingPostsForUser.sql
Type: Microsoft SQL Server Query File

sapi_Example_5_NumberOfPostsAndVotesByUser.sql
Type: Microsoft SQL Server Query File

Demo round one:



3 - SARGability and warm up stuff.sql
Type: Microsoft SQL Server Query File

3b - Expensive Sort.sql
Type: Microsoft SQL Server Query File

4 - Table Variable.sql
Type: Microsoft SQL Server Que

5 - Problem With functio
Type: Microsoft SQL Server Que File

6 - Pagination.sql
Type: Microsoft SQL Server Query File

7 - Parameter Sniffing.sql
Type: Microsoft SQL Server Query File

8 - Awesome Examples.sql
Type: Microsoft SQL Server Query File

sapi_Example_1_Q6925.sql
Type: Microsoft SQL Server Query File

sapi_Example_2_Q466.sql
Type: Microsoft SQL Server Query File

sapi_Example_3_Q3160].sql
Type: Microsoft SQL Server Query File

sapi_Example_4_FindInterestingPostsForUser.sql
Type: Microsoft SQL Server Query File

sapi_Example_5_NumberOfPostsAndVotesByUser.sql
Type: Microsoft SQL Server Query File

VERDICT round one:

- Limitations on index match with like, avoid using pattern matching characters
- Sargability stands for search argument ability – also avoid built in functions which changes the data type or doesn’t match with the column’s data type
- Issues with SELECT * , get the data you really need
- Syndrome of a SELECT * is usually an expensive key lookup

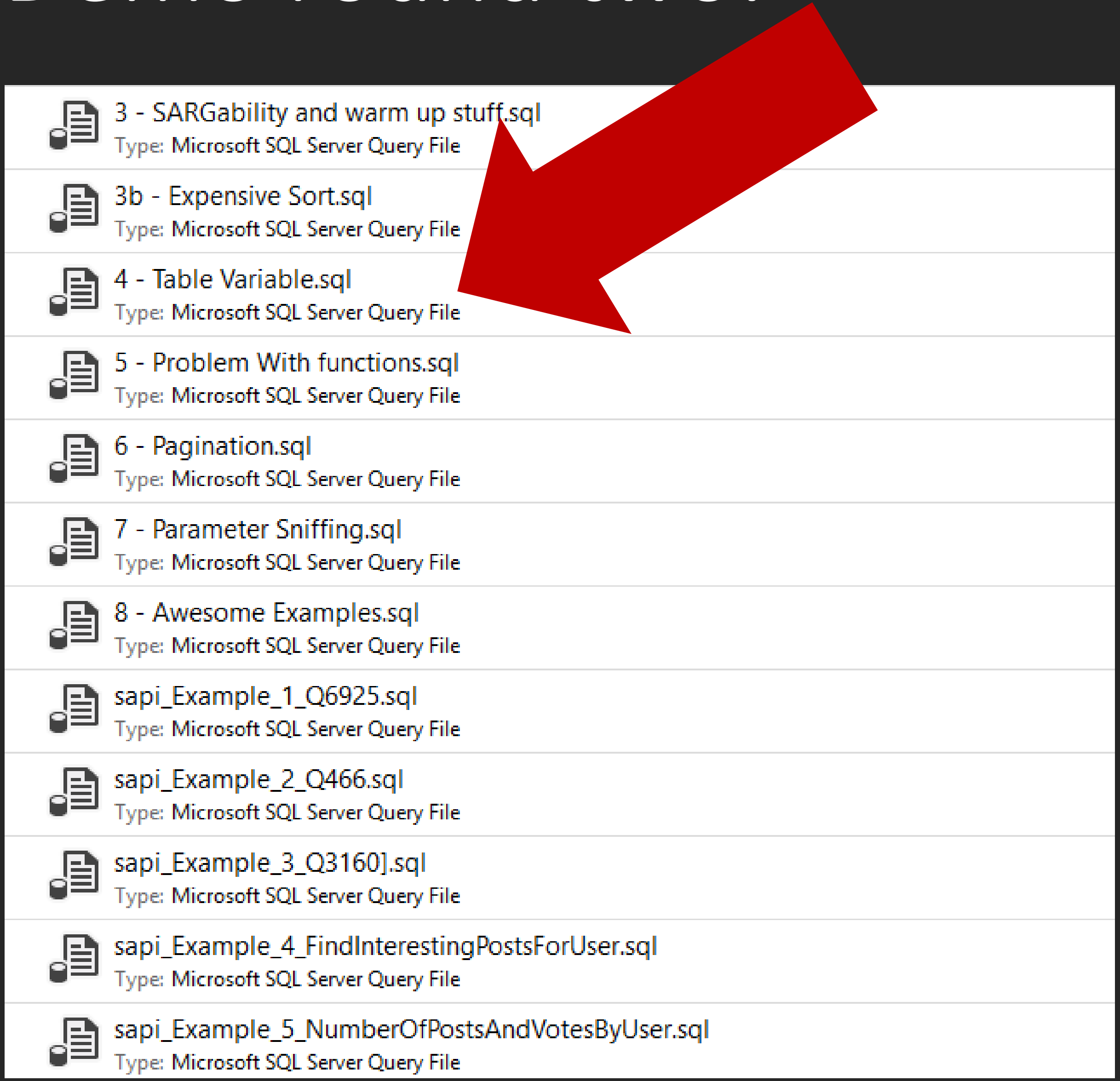
VERDICT round one:

- Real problems when a sort operator spills to TEMP DB
- Tricks may help you but try to use the correct data type for your need
- Indexes are also helping
- Most important to get to know your data

Query Issues: Those Which I learnt in the hard way - or still learning :P

- Table Variable
- Forced Serialization
- Problem with the UDFs (User Defined Functions)
- Parameter Sniffing
- Deadlock => Deadlock Graph
- Blocking

Demo round two:



3 - SARGability and warm up stuff.sql
Type: Microsoft SQL Server Query File

3b - Expensive Sort.sql
Type: Microsoft SQL Server Query File

4 - Table Variable.sql
Type: Microsoft SQL Server Query File

5 - Problem With functions.sql
Type: Microsoft SQL Server Query File

6 - Pagination.sql
Type: Microsoft SQL Server Query File

7 - Parameter Sniffing.sql
Type: Microsoft SQL Server Query File

8 - Awesome Examples.sql
Type: Microsoft SQL Server Query File

sapi_Example_1_Q6925.sql
Type: Microsoft SQL Server Query File

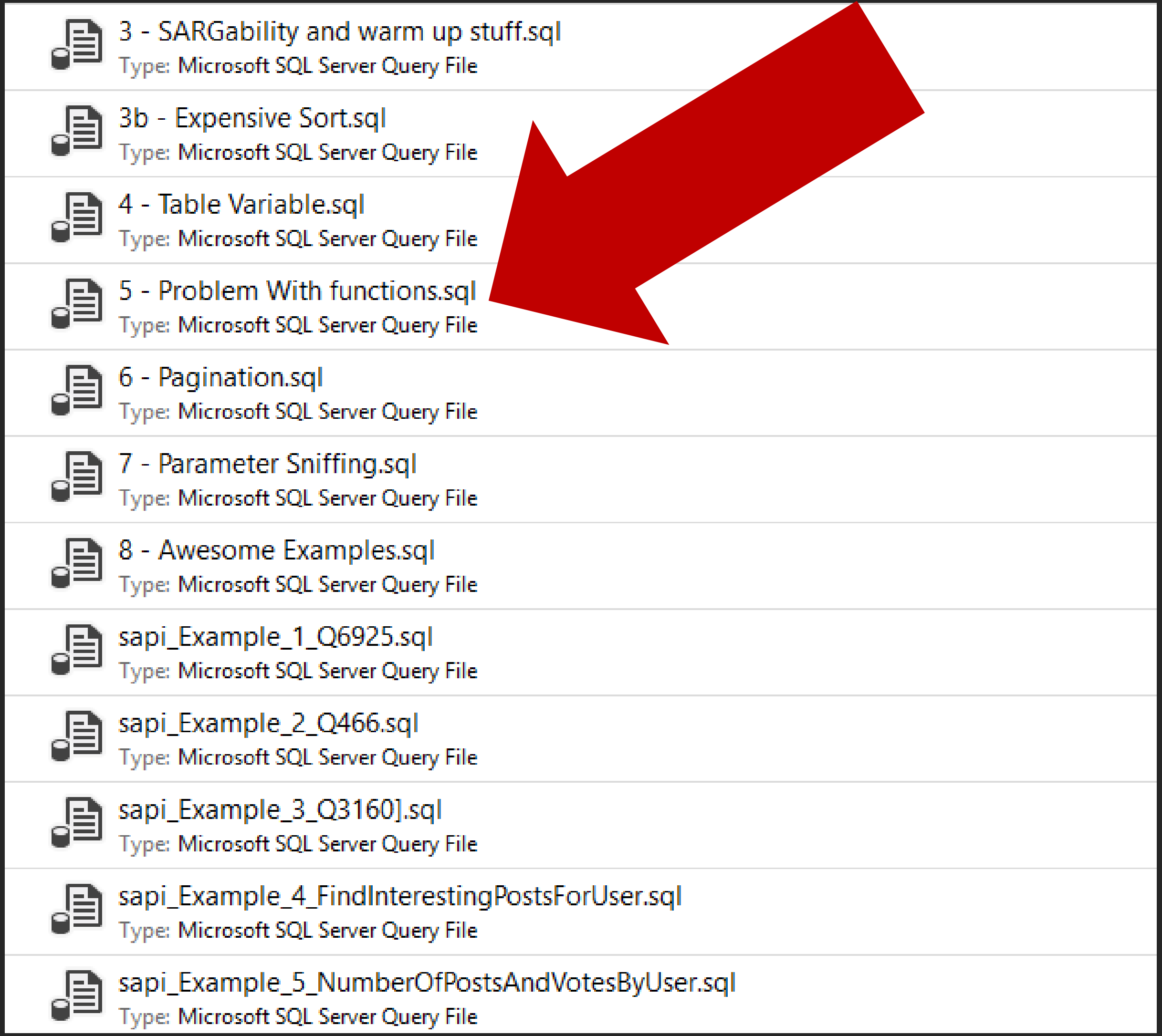
sapi_Example_2_Q466.sql
Type: Microsoft SQL Server Query File

sapi_Example_3_Q3160].sql
Type: Microsoft SQL Server Query File

sapi_Example_4_FindInterestingPostsForUser.sql
Type: Microsoft SQL Server Query File

sapi_Example_5_NumberOfPostsAndVotesByUser.sql
Type: Microsoft SQL Server Query File

Demo round two:



3 - SARGability and warm up stuff.sql
Type: Microsoft SQL Server Query File

3b - Expensive Sort.sql
Type: Microsoft SQL Server Query File

4 - Table Variable.sql
Type: Microsoft SQL Server Query File

5 - Problem With functions.sql
Type: Microsoft SQL Server Query File

6 - Pagination.sql
Type: Microsoft SQL Server Query File

7 - Parameter Sniffing.sql
Type: Microsoft SQL Server Query File

8 - Awesome Examples.sql
Type: Microsoft SQL Server Query File

sapi_Example_1_Q6925.sql
Type: Microsoft SQL Server Query File

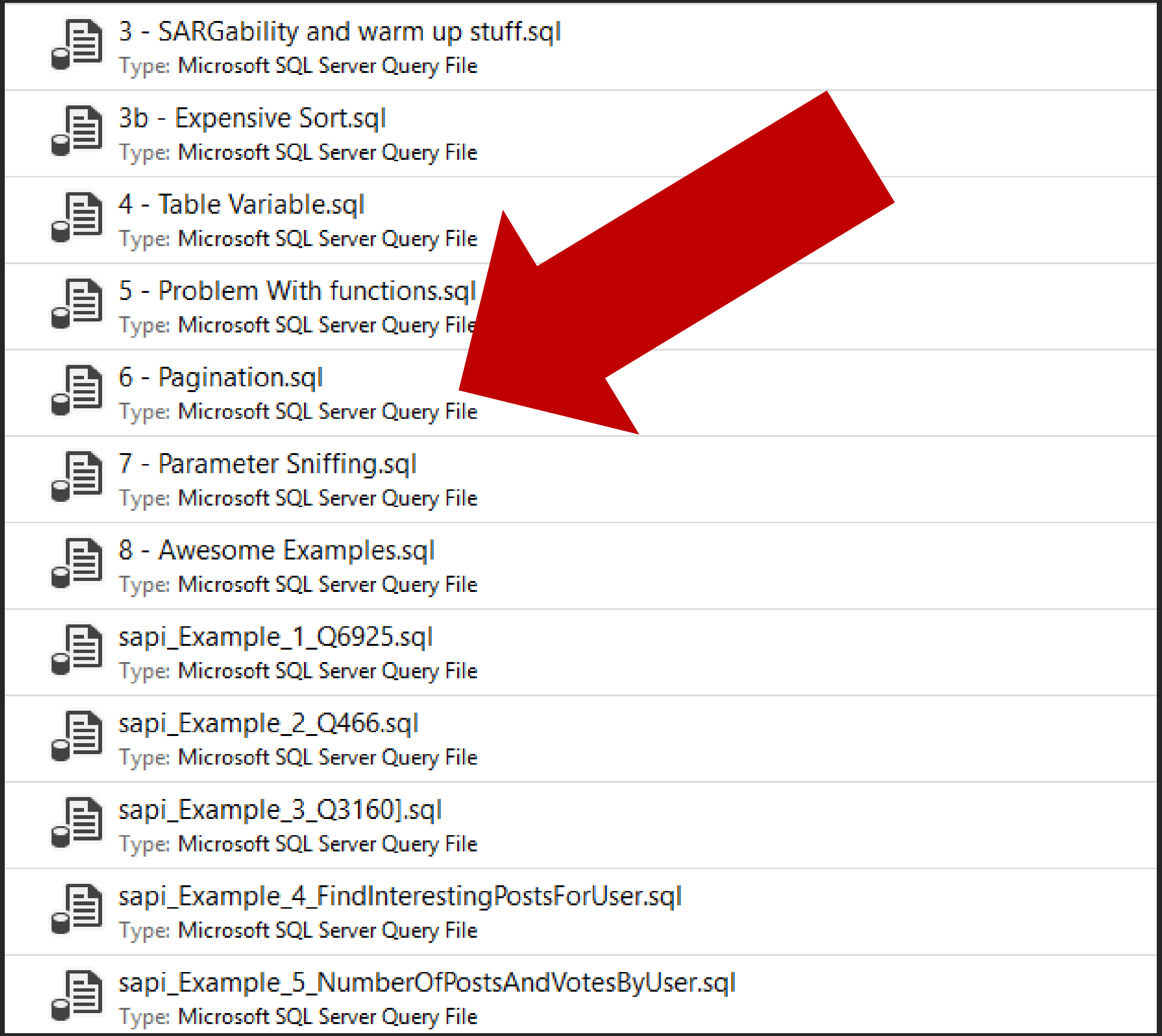
sapi_Example_2_Q466.sql
Type: Microsoft SQL Server Query File

sapi_Example_3_Q3160].sql
Type: Microsoft SQL Server Query File

sapi_Example_4_FindInterestingPostsForUser.sql
Type: Microsoft SQL Server Query File

sapi_Example_5_NumberOfPostsAndVotesByUser.sql
Type: Microsoft SQL Server Query File

Demo round two:



3 - SARGability and warm up stuff.sql
Type: Microsoft SQL Server Query File

3b - Expensive Sort.sql
Type: Microsoft SQL Server Query File

4 - Table Variable.sql
Type: Microsoft SQL Server Query File

5 - Problem With functions.sql
Type: Microsoft SQL Server Query File

6 - Pagination.sql
Type: Microsoft SQL Server Query File

7 - Parameter Sniffing.sql
Type: Microsoft SQL Server Query File

8 - Awesome Examples.sql
Type: Microsoft SQL Server Query File

sapi_Example_1_Q6925.sql
Type: Microsoft SQL Server Query File

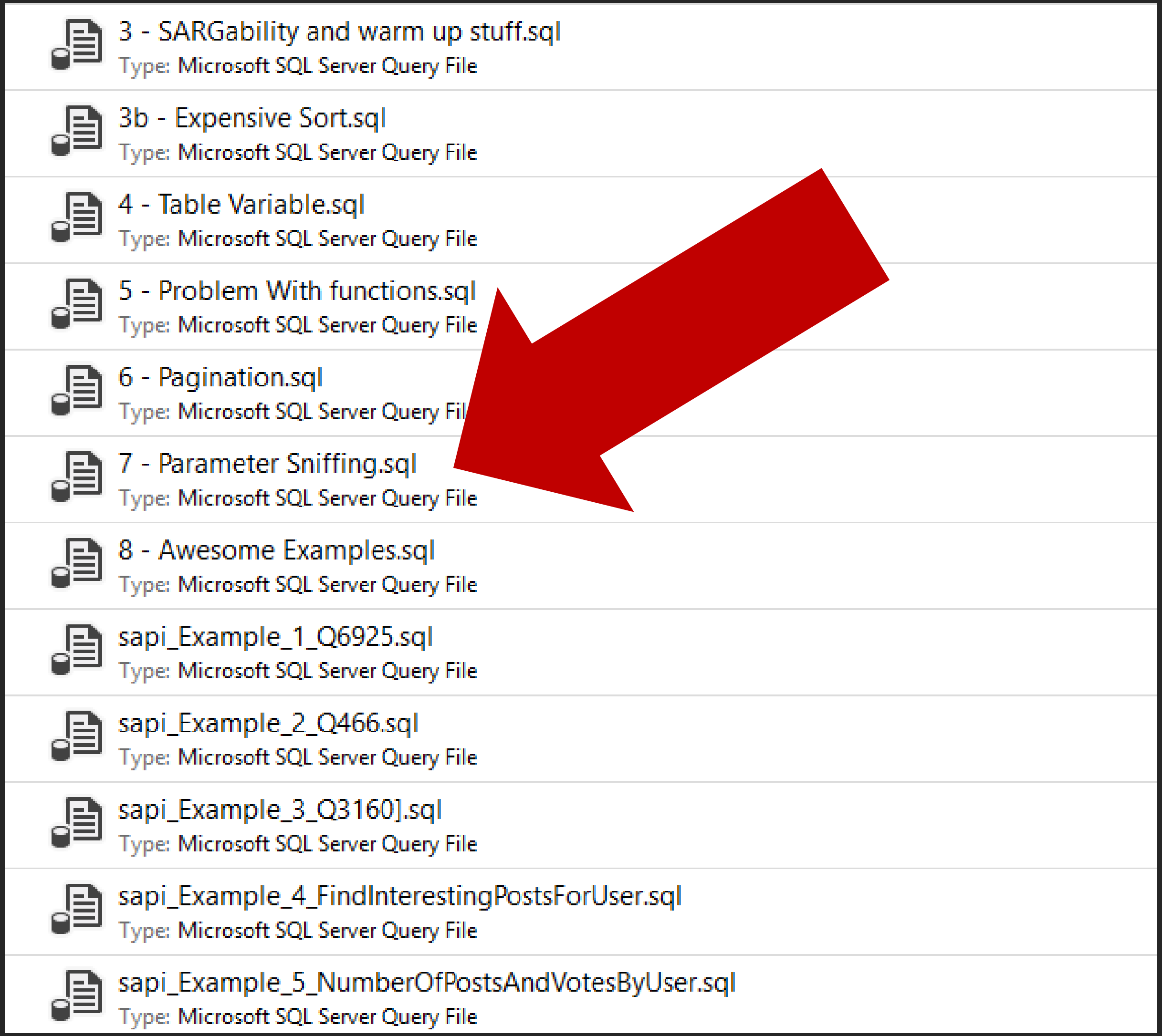
sapi_Example_2_Q466.sql
Type: Microsoft SQL Server Query File

sapi_Example_3_Q3160].sql
Type: Microsoft SQL Server Query File

sapi_Example_4_FindInterestingPostsForUser.sql
Type: Microsoft SQL Server Query File

sapi_Example_5_NumberOfPostsAndVotesByUser.sql
Type: Microsoft SQL Server Query File

Demo round two:



3 - SARGability and warm up stuff.sql
Type: Microsoft SQL Server Query File

3b - Expensive Sort.sql
Type: Microsoft SQL Server Query File

4 - Table Variable.sql
Type: Microsoft SQL Server Query File

5 - Problem With functions.sql
Type: Microsoft SQL Server Query File

6 - Pagination.sql
Type: Microsoft SQL Server Query Fil

7 - Parameter Sniffing.sql
Type: Microsoft SQL Server Query File

8 - Awesome Examples.sql
Type: Microsoft SQL Server Query File

sapi_Example_1_Q6925.sql
Type: Microsoft SQL Server Query File

sapi_Example_2_Q466.sql
Type: Microsoft SQL Server Query File

sapi_Example_3_Q3160].sql
Type: Microsoft SQL Server Query File

sapi_Example_4_FindInterestingPostsForUser.sql
Type: Microsoft SQL Server Query File

sapi_Example_5_NumberOfPostsAndVotesByUser.sql
Type: Microsoft SQL Server Query File

VERDICT round two:

- Simply forget about the existence of Table variable
- SQL is not for reusing logic so avoid procedural/object oriented thinking
- Think in sets and avoid functions use CTE, APPLY instead of them
- Limit the data a user/application can receive at once it will be much better for your SQL server
- Parameter sniffing is a tough nut to crack but I leave this to you dear reader (Dynamic SQL, SP branching, OPTIMIZE FOR UNKWOWN or to a SPECIFIC VALUE)

Real life queries at the EMERGENCY today.

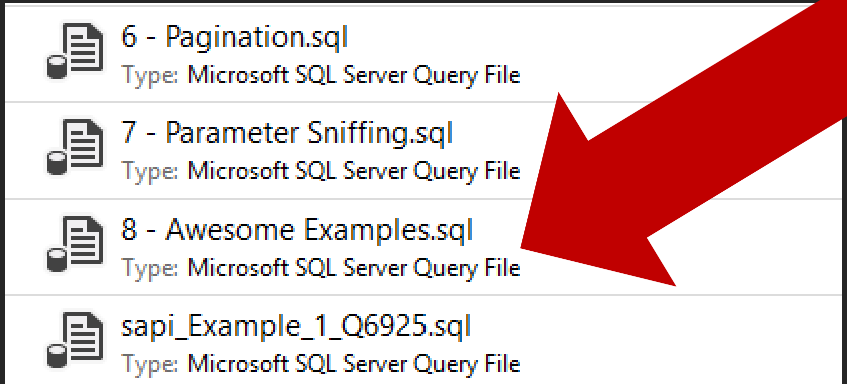
sapi_Example_3_Q3160].sql
Type: Microsoft SQL Server Query File

sapi_Example_4_FindInterestingPostsForUser.sql
Type: Microsoft SQL Server Query File

sapi_Example_5_NumberOfPostsAndVotesByUser.sql
Type: Microsoft SQL Server Query File

Awesome features I really like in T-SQL

- CTE
- Pivot
- Windowing Functions
- Dynamic SQL
- Hekaton (Columnstore Index)



6 - Pagination.sql
Type: Microsoft SQL Server Query File

7 - Parameter Sniffing.sql
Type: Microsoft SQL Server Query File

8 - Awesome Examples.sql
Type: Microsoft SQL Server Query File

sapi_Example_1_Q6925.sql
Type: Microsoft SQL Server Query File

Thank You!

Source Code : https://goo.gl/BYqqKY

Questions?

Source Code : https://goo.gl/BYqqKY