**(August 13th, 2019)**

If you know someone who would benefit from being an Insider, feel free to forward this PDF to them so they can sign up here.

Note: As an Insider, you can read all prior Insider newsletters here.

**Quick Tips for our Insider friends!**

Hey Insiders,

This newsletter is coming to you from Redmond, where we're basking in the summer sun and working on Pluralsight content.

**SQLskills News**

**Live, ONLINE classes:** we have three classes coming up in the remainder of the year:

- IETLB: Transactions, Locking, Blocking, Isolation, and Versioning – October 1-3
- IEVLT: Very Large Tables: Performance/Manageability through Partitioning – October 29-31
- IECS: Columnstore Indexes – November 12-14

The classes are US$595 each, with a US$495 early-bird price through the end of August, and there's an all-class-combo deal for US$1,350, which makes each class US$450. You can get all the details and registration information through the class schedule page here.

**Live, IN-PERSON classes:** we have two more in-person classes this year, in Chicago in October:

- IEPTO1: Performance Tuning and Optimization – Part 1 – October 7-11
- SOLD OUT: IEAzure: Azure SQL Database, Azure VMs, Azure Managed Instance – October 7-10

These will be our only, in-person Immersion Events running this Fall and you can get all the details and registration information through the class schedule page here.

**PASS workshops:** if you're planning to go to the PASS Summit in November, Erin and Jonathan both have pre-conference workshops:

- Erin: Performance Tuning with Query Store in SQL Server – see here
- Jonathan: Bigger Hardware or Better Code and Design? – see here

**Finally, even if you can't join us in person,** I've put out a call for **2019 remote user group sessions** and we've got 43 scheduled this year already! If you'd like one of us to present for your user group, check out my blog post here.

## Book Review

The most recent book I've read is Violet Moller's *The Map of Knowledge: A Thousand-Year History of How Classical Ideas Were Lost and Found*. This is a fascinating book that explains how ancient Greek knowledge survived the fall of the Roman Empire and suppression by Christianity (for being 'pagan') and was reintroduced in the Middle Ages to fuel the renaissance in Europe. It follows three key Greek texts – Euclid's Elements, Ptolemy's The Almagest, and Galen's writings on medicine – plus Arab discoveries as they move around the Mediterranean. Seven cities are highlighted, where the texts either originated, translated, copied or a combination – Alexandria, Baghdad, Cordoba, Toledo, Salerno, Palermo, and Venice – along with the eclectic people mainly responsible. Highly recommended!

## The Curious Case of…

*This section of the newsletter explains recent problems we've helped with on client systems; they might be something you're experiencing too.*

Do you know what the wait resource (0:0:0) in tempdb means?

Read the explanation in my blog post here.

## Ponderings…

*(The guest editorial this time is from Kimberly – re-running one she originally wrote many years ago but is still just as relevant today – enjoy!)*

Our two flagship 5-day classes are about performance tuning. Basically we discuss the common question: what's taking up all of the resources on my SQL Server and preventing my workload from running as fast as I want? As a part of the first class we discuss indexing in depth, and Kimberly explains a commonly overlooked cause of poor performance: having duplicate nonclustered indexes.

SQL Server will let you create up to 999 nonclustered indexes per table, and if you really want to, you can create them all to be exactly the same. *Exactly* the same, down to the *INCLUDE*d columns and key order. Why? The reason is backwards compatibility, so that SQL Server doesn't break poorly-written applications that use hard-coded index hints. In actuality, there's absolutely NO good reason to allow duplicate indexes and often people end up with duplicates that don't even look similar (because of how SQL Server internally changes the indexes based on the clustering key). The end result is that you might have duplicates and not even know it!

These indexes are not just wasted disk space however…

Every *INSERT*, *UPDATE*, and *DELETE* operation (which I'll just call 'DML' from now on) will affect every nonclustered index (except possibly filtered indexes), as each nonclustered index has to have a row for each row in the table. This means that each DML operation generates transaction log records for the change to each nonclustered index, and the index pages for each index have to be in the buffer pool to have the change made on them.

So each duplicate index is taking up space in the data files, taking up space in your precious buffer pool, taking up space in the transaction log, needing locks, needing latches, taking up processing time, and so on.

And then what about index fragmentation? Each of these duplicate nonclustered indexes will be getting fragmented (unless you've created them all to have the same index keys that match your insert pattern), so there will be a bunch of page splits happening, with all the extra log records those generate. Those fragmented indexes will likely be picked up by automated index maintenance, causing further resources to be used and transaction log to be generated. And then they may be picked up by automated statistics maintenance, and they will have to be checked by your consistency checking jobs, and so on.

You get the idea. Duplicate nonclustered indexes are really bad. And, all you have to do is remove them and you'll reduce overhead and increase performance.

That's the catch though – you have to know internals and how SQL Server stores data in the leaf and non-leaf levels of an index. Luckily Kimberly worked out how to find them and she wrote two posts:

- [How can you tell if an index is REALLY a duplicate?](#)
- [Removing duplicate indexes](#)

The second post has all the code to use to figure out which indexes are duplicates on 2008 onwards. Additionally one of our class attendees also ported the code to SQL Server 2000 – see [here](#) (still relevant for people with old systems they can't upgrade).

**Call to action:** Read the first post and then download the code from the second post and check out some of your databases to see whether you have any duplicate indexes. You'll be happy if you find them as every one of them is a leech sucking resources out of your SQL Server!

## Glenn's Tech Insights

*Recent news and views from the hardware and Windows worlds that we think will be interesting to SQL Server community members.*

In Glenn's regular roundup of tech news this time he talks about the following:

- AMD's release of the EPYC 7002 series processors
- The latest Cumulative Updates for SQL Server 2016 and 2017

Check it all out in his post here!

**#TBT**

*(Turn Back Time…) Blog posts we've published since the previous newsletter plus some older resources we've referred to recently that you may find useful.*

I've been dealing with a couple of corruption issues over the last week, so that's the theme for this TBT:

- My course: *SQL Server: Detecting and Correcting Database Corruption*
- My course: *SQL Server: Advanced Corruption Recovery Techniques*
- My blog post: *SQLskills SQL101: Why DBCC CHECKDB can miss memory corruption*
- My blog post: *SQLskills SQL101: Dealing with SQL Server corruption*
- My blog post: *Disaster recovery 101: fixing a broken boot page*
- My blog post: *Corruption recovery using DBCC WRITEPAGE*
- My *Corruption* blog post category

Here are the blog posts we've published since the last newsletter:

- Glenn: *SQLskills SQL101: Upgrading to a Different Edition of SQL Server*
- Glenn: *SQL Server Diagnostic Information Queries for August 2019*
- Erin: *SQLskills at the PASS Summit*
- Erin: *tempdb Enhancements in SQL Server 2019*
- Erin: *What columns are in that index?*

I hope you find these useful and interesting!

**Video Demo**

In the demo video this week, Jonathan shows how to enable the collection of execution statistics for natively-compiled stored procedures in memory-optimized workloads.

The video is about 8.5 minutes long and you can get it here.

The demo code zip is here.

Enjoy!

**Upcoming SQLskills Events**

Our set of 2019 live, in-person classes for Chicago in October are filling up fast and we've announced our Fall live, online events too.

You have multiple learning opportunities as every event has a different focus as well as different benefits – from deep-technical training in our Immersion Events to wide-ranging topics at SQLintersection where you can learn more effectively how to keep moving forward in both your environment and your career! And, of course, one benefit you'll always get from in-person events is networking; we hope to meet/see you at an event soon!

To help your boss understand the importance of focused, technical training, we've also added a few items to help you justify spending your training dollars with us:

- Letter to your boss explaining why SQLskills training is worthwhile
- So why do you want to come to our training? And the winners are…
- Community blog posts about our classes
- Immersion Event FAQ

**LIVE, IN-PERSON Immersion Events:**

Chicago, IL, October 2019

- **IEPTO1**: Immersion Event on Performance Tuning and Optimization – Part 1
    o October 7-11
- **IEAzure**: Immersion Event on Azure SQL Database, Azure VMs, and Azure Managed Instance
    o October 7-10 **SOLD OUT!!**

**LIVE, ONLINE Immersion Events:**

- **IETLB**: Immersion Event on Transactions, Locking, Blocking, Isolation, and Versioning
    o October 1-3
- **IEVLT**: Immersion Event on Very Large Tables: Performance/Manageability through Partitioning
    o October 29-31
- **IECS**: Immersion Event on Columnstore Indexes
    o November 12-14

Click here for the main Immersion Event Calendar page that allows you to drill through to each class for more details and registration links.

**Summary**

We hope you've enjoyed this issue - we really enjoy putting these together.

If there is anything else you're interested in, we'd love to hear from you - [drop us a line](drop us a line).

Thanks,
Paul and Kimberly

[Paul@SQLskills.com](Paul@SQLskills.com) and [Kimberly@SQLskills.com](Kimberly@SQLskills.com)