BRENT OZAR
UNLIMITED®

Why Defragmenting Your Indexes Isn't Helping

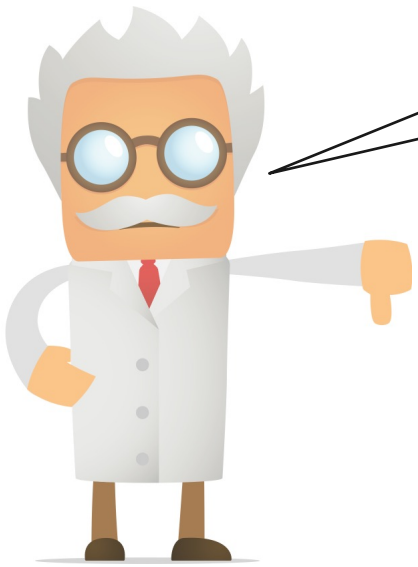
BRENT OZAR
UNLIMITED

99-05: dev, architect, DBA
05-08: DBA, VM, SAN admin
08-10: MCM, Quest Software
Since: consulting DBA

www.BrentOzar.com
Help@BrentOzar.com

You're here because performance sucks.

Success means a performance boost that end users notice.



“Fragmentation sucks.”

The two kinds of fragmentation

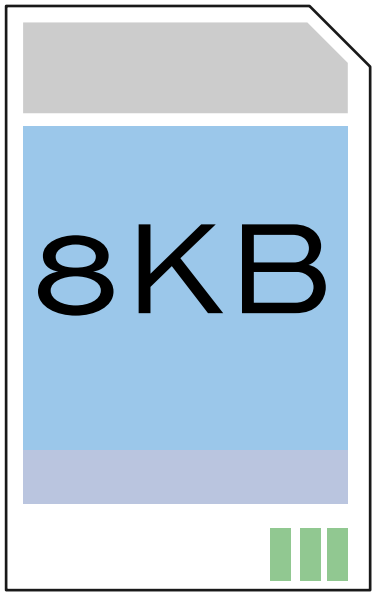
Why the fix makes things worse

The better performance fix

When people insist on rebuilds

What to do overnight instead

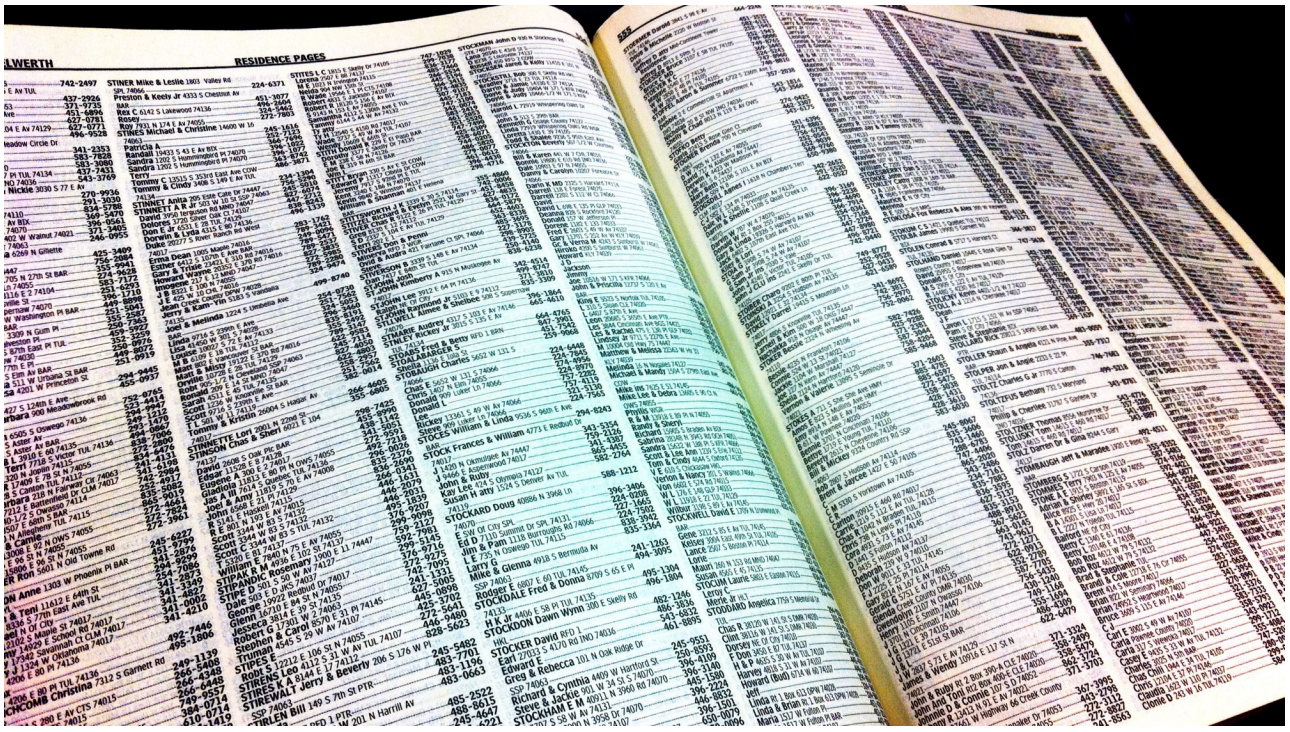
Data is stored in 8KB pages.



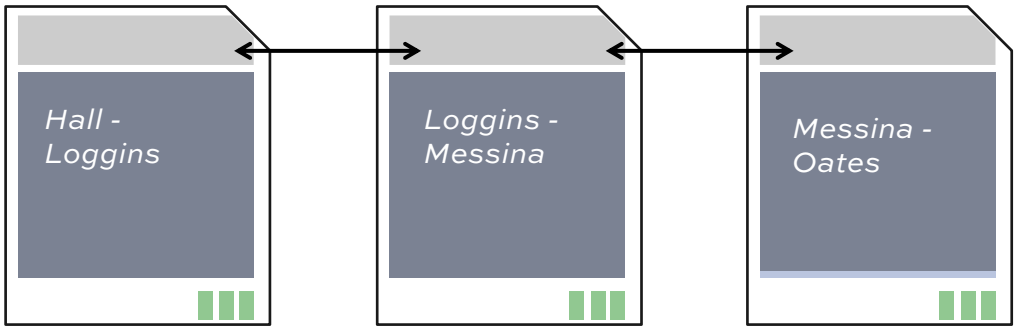
Page Header

Index *OR* Data Rows

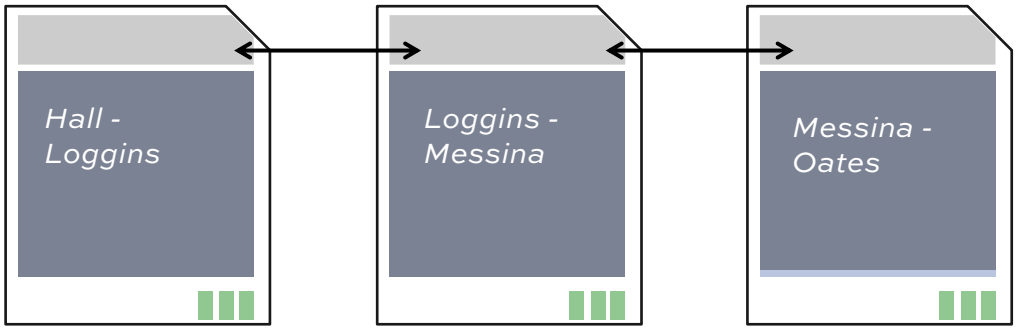
Slot Array



In a freshly rebuilt index, the 8K pages are full, and let's say they're all in order on disk.



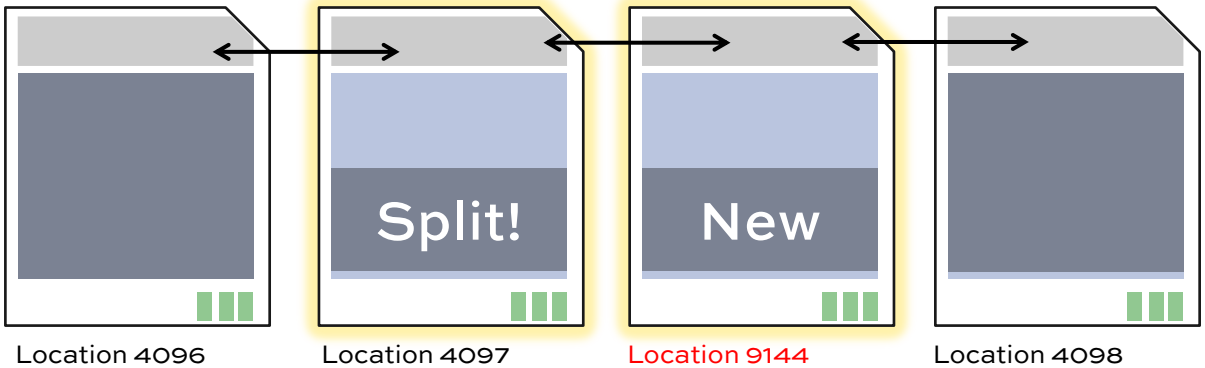
McDonald, Michael moves to our town.



He needs to go here, but there's no empty space.

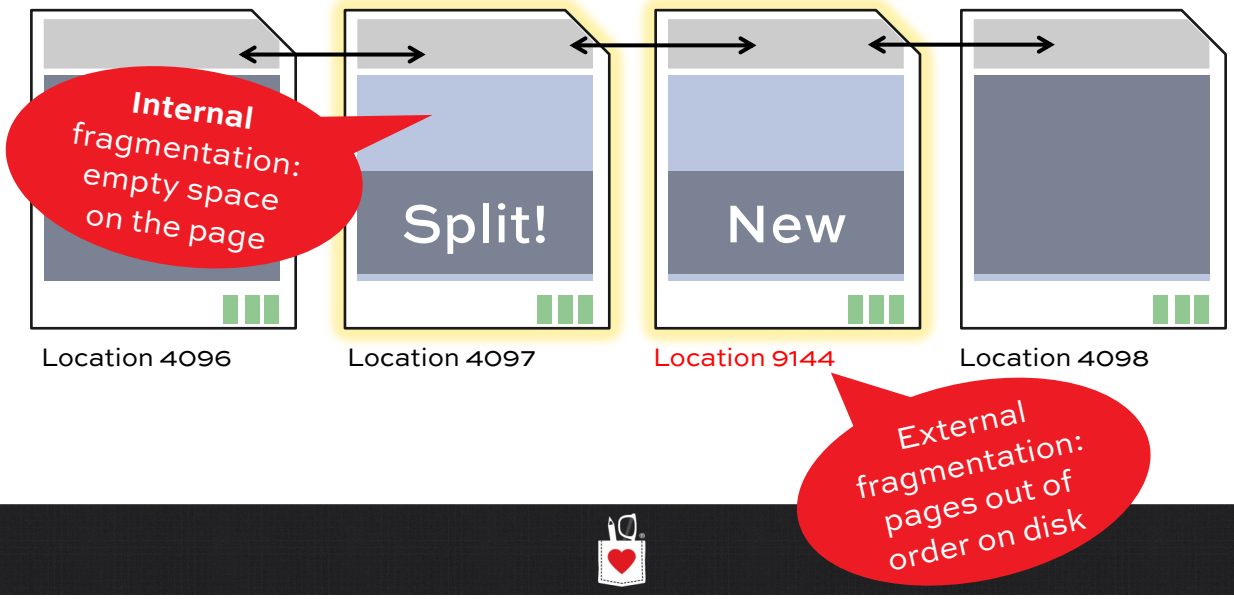
Page split!

SQL allocates another page.



But it won't be in physical order.

There are two kinds of fragmentation here.



Monitoring this stuff

To track page splits:
Perfmon counter SQL Server:Access Methods – Page Splits/sec

To track external fragmentation:
sys.dm_db_index_physical_stats – avg_fragmentation_in_percent

To track internal fragmentation:
sys.dm_db_index_physical_stats – avg_page_space_used_in_percent

What does that actually mean though?

To find out, let's:

1. Create a table
2. Check page splits
3. Insert one row
4. Check page splits again

```
1  DROP TABLE IF EXISTS dbo.Customers;
2  GO
3  CREATE TABLE dbo.Customers
4      (Id INT IDENTITY(1,1) PRIMARY KEY CLUSTERED,
5       CustomerName VARCHAR(100));
6  GO
7  SELECT * FROM sys.dm_os_performance_counters
8  WHERE counter_name LIKE '%Splits%';
9  GO
10 INSERT INTO dbo.Customers(CustomerName)
11     VALUES ('Brent Ozar')
12 GO
13 SELECT * FROM sys.dm_os_performance_counters
14 WHERE counter_name LIKE '%Splits%';
```

```
7  SELECT * FROM sys.dm_os_performance_counters
8  WHERE counter_name LIKE '%Splits%';
9  GO
10 INSERT INTO dbo.Customers(CustomerName)
11     VALUES ('Brent Ozar')
12 GO
13 SELECT * FROM sys.dm_os_performance_counters
14 WHERE counter_name LIKE '%Splits%';
```

Results | Messages

object_name	counter_name	instance_name	cntr_value	cntr_type
SQLServer:Access Methods	Page Splits/sec		30	272696576

Went up by 1.

object_name	counter_name	instance_name	cntr_val	
SQLServer:Access Methods	Page Splits/sec		31	272696576

Adding a new page is also called a **page split**.

Monitoring this stuff

Lies

To track page splits:
Perfmon counter SQL Server:Access Methods – Page Splits/sec
There are other more accurate ways to monitor page splits, but if this lying thing came as a surprise to you, buckle up. It's about to get worse.

To track external fragmentation:
sys.dm_db_index_physical_stats – avg_fragmentation_in_percent

To track internal fragmentation:
sys.dm_db_index_physical_stats – avg_page_space_used_in_percent

Is external fragmentation bad?

Pop quiz: if your pages are out of order

Will your maintenance tasks take longer?
(Backups, index & stats maintenance, CHECKDB)

Will queries reading from RAM be slower?

Will queries reading from disk be slower?

- What if you're seeking?
- What if you're scanning?

How do you fix it?

Pop quiz: if your pages are out of order

Will your maintenance tasks take longer? No
(Backups, index & stats maintenance, CHECKDB)

Will queries reading from RAM be slower? No

Will queries reading from disk be slower?

- What if you're seeking? No
- What if you're scanning? Maybe, but...

How do you fix it?

Systems Performance by Brendan Gregg

1 CPU cycle	0.3 ns
Level 1 cache access	0.9 ns
Level 2 cache access	2.8 ns
Level 3 cache access	12.9 ns
Main memory access	120 ns
Solid-state disk I/O	50-150 µs
Rotational disk I/O	1-10 ms
Internet: SF to NYC	40 ms
Internet: SF to UK	81 ms
Internet: SF to Australia	183 ms

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

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	µs	2-6 days
Rotational disk I/O		1-12 months
Internet: SF to NYC	ms	4 years
Internet: SF to UK	81 ms	8 years
Internet: SF to Australia	183 ms	19 years

Users want this.

They don't notice the difference down here.

Pop quiz: if your pages are out of order

Will your maintenance tasks take longer? No
(Backups, index & stats maintenance, CHECKDB)

Will queries reading from RAM be slower? No

Will queries reading from disk be slower?

- What if you're seeking? No
- What if you're scanning? Maybe, but...

How do you fix it? This sucks.

Fixing out-of-order pages

	Reorganize	Rebuild
What does it do?	Shuffles individual pages around, one at a time	Builds a whole new copy of the index
Are all pages in physical order when we're done?	No	Maybe, but only if you use MAXDOP 1
Can it be done online?	Yes	Enterprise Edition only
Can it be interrupted?	Yes, and your progress is kept	No
Is it a logged operation?	Yes	Yes
Does it slow down mirrors, AGs, log shipping?	Yes	Yes

Users hate your "fix."

Pop quiz: if your pages are out of order

Will your maintenance tasks take longer? (Backups, index & stats maintenance, CHECKDB)

Defragmenting makes this WORSE

Will queries reading from RAM be slower?

Doesn't help

Will queries reading from disk be slower?

- What if you're seeking?

Doesn't help

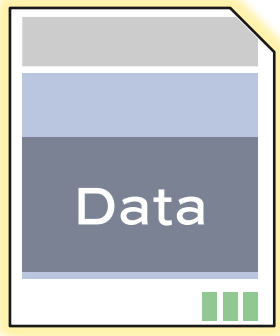
- What if you're scanning?

Helps – but not much

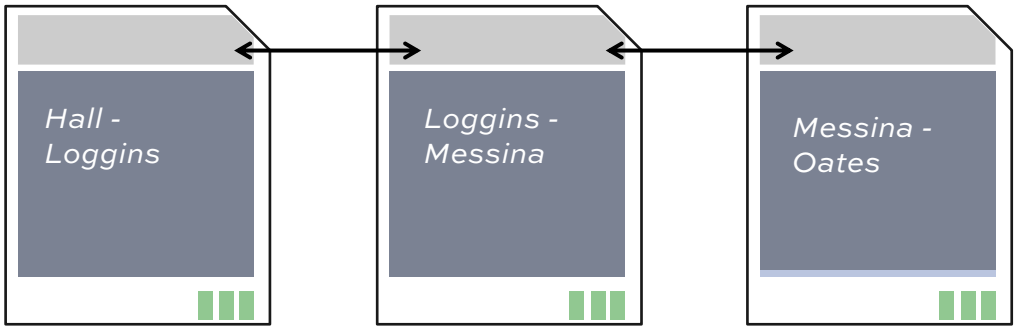
How do you fix it?

I'm not so sure we should, but if we do, it shouldn't be with constant index shuffling.

Wait! I'll set FILL FACTOR!



McDonald, Michael moves to our town.



He needs to go here, but there's no empty space.

But what if there was?

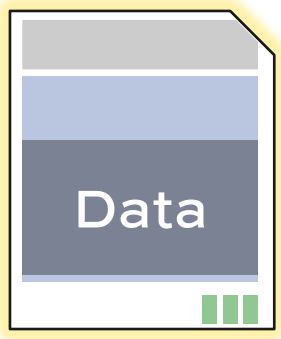
Fill factor leaves empty space on the page.

Default set at the server level: 100% (and 0 means the same thing)

So the idea is:

- Set fill factor to something lower (like 70-80)
- Leave empty space on every page
- When new records are added, we'll have space so we don't need to split pages

Let's see a couple examples.



dbo.Users Clustered index on ID

Users are ordered by ID, an identity field

Do you want empty space on this page?

- Inserts?
- Updates?
- Deletes?

What's the right fill factor?

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
557	1	7/15/09 9:45 AM	Beto	2/13/10 11:01 AM	NULL	NULL	NULL
558	953	7/15/09 9:46 AM	marco.ragogna	4/1/10 7:41 AM	IT	29	<h2>Software Developer</
559	593	7/15/09 9:46 AM	moobaa	2/18/10 7:14 AM	AU	39	Hey, at least I know 6502!
560	83	7/15/09 9:46 AM	rjstelling	3/4/10 6:47 PM	United Kingdom	30	<h3>Tech geek and hopeles
561	716	7/15/09 9:47 AM	balpha	4/1/10 4:00 PM	NULL	NULL	NULL
562	43	7/15/09 9:47 AM	Mike McClelland	3/16/10 8:45 AM	London	NULL	NULL
563	101	7/15/09 9:48 AM	arturh	10/29/09 2:11 PM	Bremen, Germany	24	NULL
564	960	7/15/09 9:49 AM	Sliff	4/1/10 6:59 AM	UK	30	<p>I'm a Software Enginee
565	121	7/15/09 9:49 AM	Franci Penov	3/5/10 12:13 AM	Kirkland, WA	37	<p>Father, SDE@Microsoft

From http://StackOverflow.com's Creative Commons Data Dump

dbo.Users non-clustered on LastAccessDate

Users are ordered by the last date they visited Stack Overflow

Do you want empty space on this page?

- Inserts?
- Updates?
- Deletes?

What's the right fill factor?

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
7/15/09 7:13 AM	65	7/15/09 10:21 AM	618	7/20/09 1:06 PM	849	8/18/09 8:06 AM	511
7/15/09 7:14 AM	68	7/15/09 10:25 AM	347	7/21/09 7:22 AM	881	8/18/09 9:00 AM	262
7/15/09 7:15 AM	73	7/15/09 10:26 AM	623	7/23/09 11:53 AM	503	8/18/09 9:43 AM	210
7/15/09 7:17 AM	87	7/15/09 10:28 AM	629	7/23/09 12:56 PM	446	8/18/09 10:22 AM	673
7/15/09 7:18 AM	92	7/15/09 10:32 AM	638	7/24/09 12:15 AM	407	8/18/09 1:05 PM	959
7/15/09 7:20 AM	110	7/15/09 10:36 AM	642	7/24/09 12:08 PM	488	8/18/09 1:14 PM	643
7/15/09 7:26 AM	139	7/15/09 10:46 AM	661	7/24/09 3:17 PM	144	8/18/09 1:32 PM	525
7/15/09 7:36 AM	173	7/15/09 10:59 AM	686	7/26/09 8:48 AM	615	8/18/09 1:38 PM	159
7/15/09 7:38 AM	179	7/15/09 11:14 AM	722	7/27/09 2:27 PM	337	8/18/09 2:28 PM	690
7/15/09 7:46 AM	223	7/15/09 11:26 AM	756	7/28/09 3:17 AM	174	8/19/09 9:22 AM	750
7/15/09 7:57 AM	258	7/15/09 11:48 AM	803	7/28/09 12:34 PM	118	8/19/09 2:46 PM	887
7/15/09 7:58 AM	259	7/15/09 11:50 AM	681	7/28/09 9:28 PM	201	8/20/09 3:45 AM	749
7/15/09 8:03 AM	279	7/15/09 11:51 AM	808	7/28/09 10:53 PM	656	8/20/09 8:29 AM	268
7/15/09 8:06 AM	286	7/15/09 12:00 PM	822	7/29/09 12:06 PM	751	8/20/09 10:32 AM	901
7/15/09 8:12 AM	305	7/15/09 12:07 PM	841	7/30/09 12:26 PM	754	8/20/09 12:05 PM	752
7/15/09 8:13 AM	313	7/15/09 12:07 PM	842	7/31/09 6:36 PM	11	8/20/09 3:03 PM	272
7/15/09 8:14 AM	320	7/15/09 12:18 PM	878	8/1/09 12:09 PM	775	8/21/09 7:29 PM	487
7/15/09 8:16 AM	323	7/15/09 12:18 PM	779	8/1/09 7:29 PM	670	8/22/09 11:58 AM	580
7/15/09 8:25 AM	349	7/15/09 12:47 PM	973	8/1/09 11:33 PM	85	8/24/09 11:22 AM	56
7/15/09 8:26 AM	354	7/15/09 12:48 PM	819	8/3/09 10:13 PM	300	8/24/09 11:33 PM	889
7/15/09 8:27 AM	97	7/15/09 12:52 PM	987	8/5/09 12:06 PM	918	8/25/09 8:57 PM	150
7/15/09 8:27 AM	363	7/15/09 12:54 PM	992	8/6/09 5:46 AM	726	8/26/09 2:43 AM	851
7/15/09 8:33 AM	384	7/15/09 1:28 PM	931	8/6/09 10:42 AM	370	8/26/09 2:53 AM	117
7/15/09 8:37 AM	402	7/15/09 2:19 PM	728	8/6/09 12:37 PM	398	8/26/09 9:59 AM	546
7/15/09 8:47 AM	431	7/15/09 2:20 PM	217	8/9/09 5:28 AM	45	8/27/09 3:54 AM	239
7/15/09 8:52 AM	440	7/15/09 4:23 PM	975	8/10/09 12:25 AM	389	8/27/09 2:10 PM	271

From http://StackOverflow.com's Creative Commons Data Dump

The default fill factor of 100% keeps your database small.

When you set fill factor <> 100%, guess what you're really doing.

When you set fill factor <> 100%, guess what you're really doing.

That's right: setting fill factor <> 100% actually causes...

Internal fragmentation

Pop quiz: if your pages have empty space

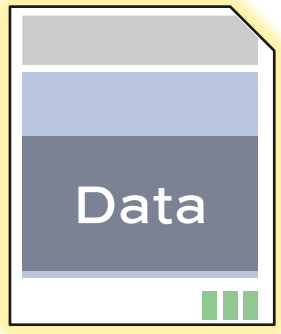
Will your maintenance tasks take longer?
(Backups, index & stats maintenance, CHECKDB)

Will queries reading from RAM be slower?

Will queries reading from disk be slower?

- What if you're seeking?
- What if you're scanning?

How do you fix it?



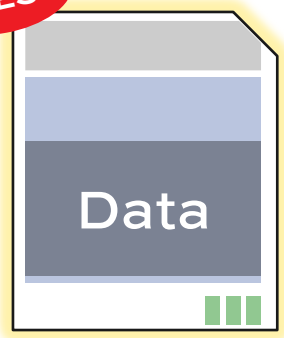
Pop quiz: if your pages have empty space

Will your maintenance tasks take longer? YES
(Backups, index & stats maintenance, CHECKDB)

Will queries reading from RAM be slower? YES

Will queries reading from disk be slower?

- What if you're seeking? No
- What if you're scanning? YES

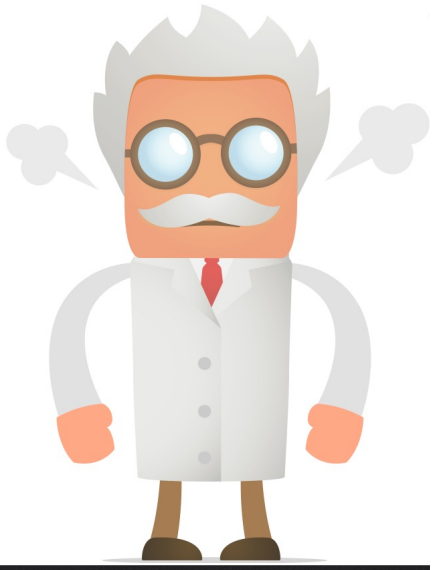


THE PROBLEM IS US.

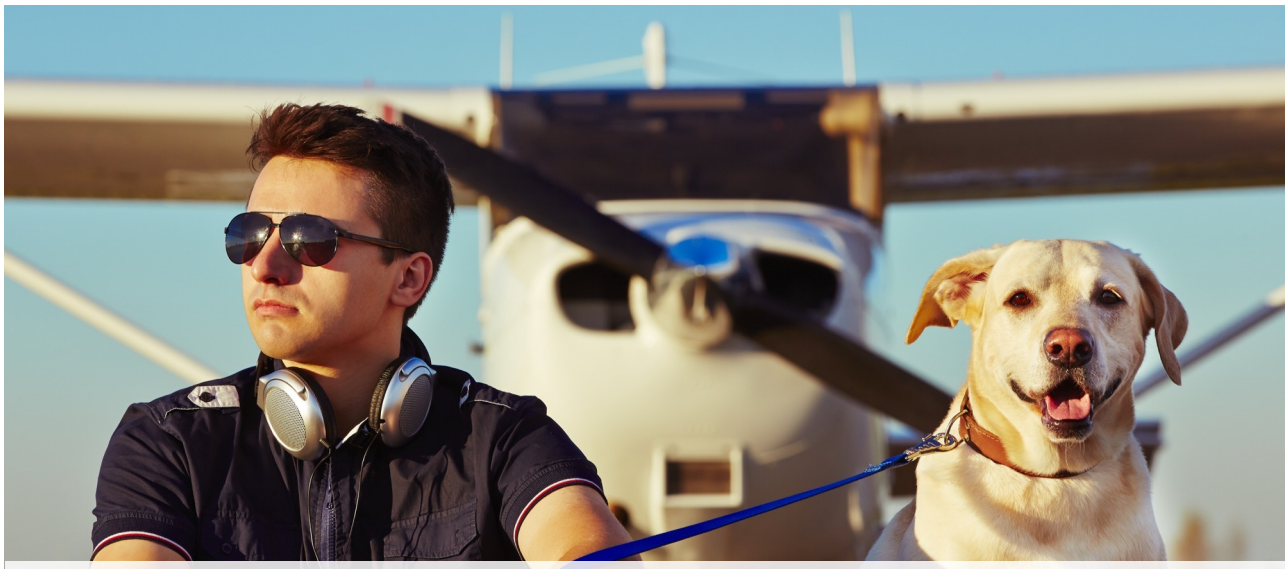
External fragmentation doesn't really matter

But to "fix" it, we've been:

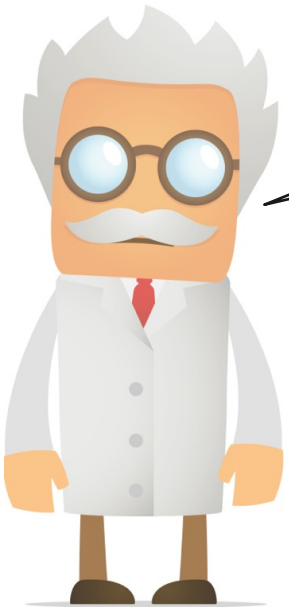
- Rebuilding/reorging our indexes daily, which actually **makes some things worse**
- Setting fill factor, which **causes internal fragmentation**



And internal fragmentation DOES matter
(and we've been the ones causing it all along!)



SQL Server needs a dog.



I'm so confused.

~~The two kinds of fragmentation~~

~~Why the fix makes things worse~~

The better performance fix

When people insist on rebuilds

What to do overnight instead

The short answer

1. Set fill factor at the server level back to 100%.
2. Use sp_Blitz and sp_BlitzIndex to check for index fill factors <80%, and set those back up to 80% or higher (or 100%).
3. Rebuild your indexes once to atone for those past sins, getting them back up to 100% fill factor.
4. Monitor the right metric, and use the scientific method to improve it.

How SQL Server Schedules CPU

What's Running Now **What's Waiting (Queue)**

How SQL Server Schedules CPU

What's Running Now

SELECT * FROM
dbo.Restaurants
(By Brent)

What's Waiting (Queue)

How SQL Server Schedules CPU

What's Running Now

SELECT * FROM
dbo.Restaurants
(By Brent)

What's Waiting (Queue)

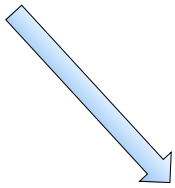
SELECT * FROM
dbo.Tattoos
(By Jeremiah)

SELECT * FROM
dbo.Rabbits
(By Kendra)

How SQL Server Schedules CPU

What's Running Now

SELECT * FROM
dbo.Restaurants
(By Brent)



What's Waiting (Queue)

SELECT * FROM
dbo.Tattoos
(By Jeremiah)

SELECT * FROM
dbo.Rabbits
(By Kendra)

How SQL Server Schedules CPU

What's Running Now

What's Waiting (Queue)

SELECT * FROM
dbo.Tattoos
(By Jeremiah)

SELECT * FROM
dbo.Rabbits
(By Kendra)

SELECT * FROM
dbo.Restaurants
(By Brent)

SQL Server Waits For:

Resources:
CPU, memory, storage, network, latches, locks

Stuff outside of SQL Server (Preemptive):
COM, OLEDB, CLR

System Tasks:
Lazywriter, trace, full text search

Is this SQL Server working hard?

What's Running Now	What's Waiting (Queue)
SELECT * FROM dbo.Restaurants *(By Brent)*	

What about now?

What's Running Now

SELECT * FROM dbo.Restaurants
(By Brent)

What's Waiting (Queue)

SELECT * FROM dbo.Tattoos
(By Jeremiah)

SELECT * FROM dbo.Rabbits
(By Kendra)

Or now?

What's Running Now

SELECT * FROM dbo.Restaurants
(By Brent)

What's Waiting (Queue)

SELECT * FROM dbo.Tattoos
(By Jeremiah)

SELECT * FROM dbo.Rabbits
(By Kendra)

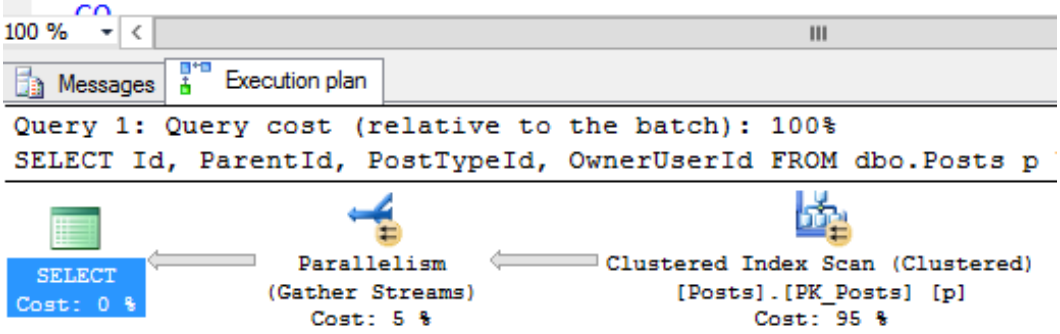
EXEC sp_WhoIsActive
(By Adam Machanic)

BACKUP DATABASE SQLbits
(By Ola Hallengren)

DBCC CHECKDB
(By Paul Randal)

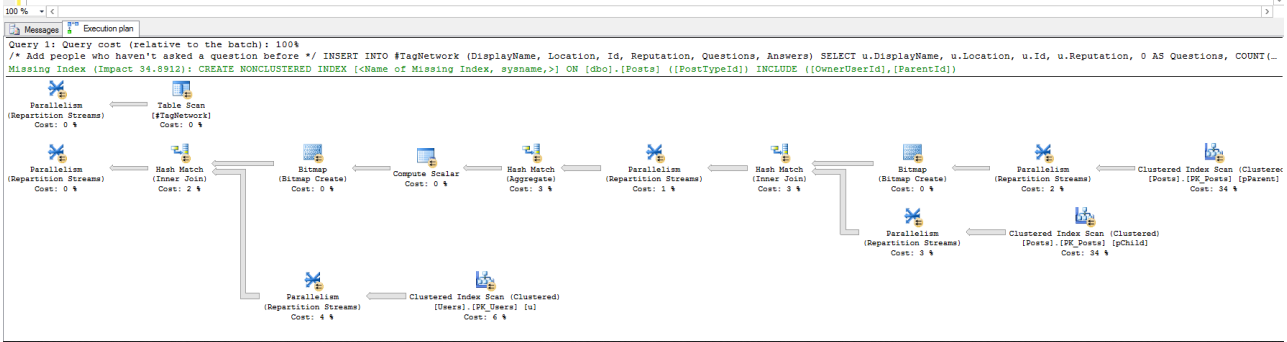
Some queries are simple

```
SELECT Id, ParentId, PostTypeId, OwnerUserId
FROM dbo.Posts p
WHERE p.Tags LIKE '%<sql-server>%';
GO
```



Some queries have a lot going on

```
INSERT INTO #TagNetwork (DisplayName, Location, Id, Reputation, Questions, Answers)
SELECT u.DisplayName, u.Location, u.Id, u.Reputation, 0 AS Questions, COUNT(*) AS Answers
  FROM dbo.Posts pChild
  INNER JOIN dbo.Posts pParent ON pParent.Id = pChild.ParentId
  INNER JOIN dbo.Users u ON pChild.OwnerUserId = u.Id
  WHERE pParent.Tags LIKE '%<sql-server>%' AND pChild.PostTypeId = 2
    AND NOT EXISTS (SELECT * FROM #TagNetwork WHERE Id = u.Id)
GROUP BY u.DisplayName, u.Location, u.Id, u.Reputation
ORDER BY COUNT(*) DESC;
```



These were simple little queries.

What's Running Now
SELECT * FROM dbo.Restaurants
(By Brent)

What's Waiting (Queue)
SELECT * FROM dbo.Tattoos
(By Jeremiah)

SELECT * FROM dbo.Rabbits
(By Kendra)

But one query looks more like this:

What's Running Now
Index scan of Table1

What's Waiting (Queue)
Index scan of Table2

Index scan of Table3

Index scan of Table4

So for each core, you'll likely see:

What's Running Now

One task running, using CPU

What's Waiting (Queue)

Several (or several DOZEN) tasks stacked up waiting on stuff

If we do this for one second, how many seconds of waits?

What's Running Now

One task running, using CPU

What's Waiting (Queue)

15 tasks waiting on storage

3 tasks waiting on locks

1 task waiting on CPU

Hours of wait time per hour

How hard is your server working?

Dynamic Management View (DMV)
sys.dm_os_wait_stats

Tracked cumulatively over time

Trend on an hourly basis and break it out by:

- Weekday vs weekend
- Business hours vs after hours
- Maintenance windows
  (backups, DB maintenance)

Wait Time per core per hour

Near zero – Your server isn't doing anything.
(Although that doesn't mean every query is fast.)

1 hour of waits – You're still not doing much.

Multiple hours per core – now we're working!
And we should probably be tuning.

Free script: sp_BlitzFirst

Totally free open source diagnostic tool.

Installs in the master by default, but can go anywhere.

Runs in 5 seconds ideally, but can be more under load.

By default, shows waits for a 5-second sample now.

@SinceStartup = 1 shows waits since, uh, startup.

Solve the emergencies, then focus on waits, in order.

Some waits are like poison:
any occurrences of them have to be fixed first.

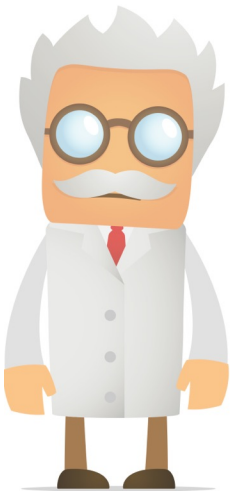
After that, work through the list.

You'll never make them go away,
but you have to focus on them in order.

	Priority	FindingsGroup	Finding	URL	Details	How
1	0	sp_BlitzFirst 2017-03-0...	From Your Community Volunteers	http://FirstResponderKit.org/	<?ClickToSeeDetails -- We hope you found this t...	NUL
2	10	Server Performance	Poison Wait Detected: THREADPOOL	http://www.brentozar.com/go/poi...	<?ClickToSeeDetails -- For 0 seconds over the l...	<?Cl
3	50	Server Performance	High CPU Utilization	http://www.BrentOzar.com/go/cpu	<?ClickToSeeDetails -- 98%. Ring buffer details: ...	NUL
4	200	Wait Stats	CXPACKET	http://www.brentozar.com/sql/wai...	<?ClickToSeeDetails -- For 25750 seconds over ...	<?Cl
5	200	Wait Stats	SOS_SCHEDULER_YIELD	http://www.brentozar.com/sql/wai...	<?ClickToSeeDetails -- For 237 seconds over th...	<?Cl
6	200	Wait Stats	MEMORY_ALLOCATION_EXT	http://www.brentozar.com/sql/wai...	<?ClickToSeeDetails -- For 55 seconds over the ...	<?Cl

Wait stats decoder ring

Wait type	Means	sp_BlitzCache @SortOrder =
CXPACKET, CXCONSUMER	Parallelism	Reads, CPU
LCK*	Blocking locks	Duration, reads
PAGEIOLATCH	Reading data from data files	Reads
SOS_SCHEDULER_YIELD	Waiting for CPU time	CPU
RESOURCE_SEMAPHORE	Query needs memory to start	Unused grant
WRITELOG	Writing to the transaction log	Writes
HADR_SYNC_COMMIT	Writing to AG sync secondaries	Writes



What if someone wants to play doctor?

~~The two kinds of fragmentation~~

~~Why the fix makes things worse~~

~~The better performance fix~~

When people insist on rebuilds

What to do overnight instead

How to react

"The problem is high fragmentation."

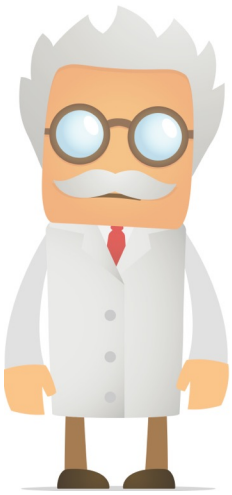
"So you guarantee that rebuilding the indexes will fix it?"

"Uhhhh..."

"What metric shall we monitor to know that the rebuild fixed it?"

"Ummmm...query runtime?"

"Great. Give me a query, and the current average runtime, and what you expect it'll be afterwards."



So what do I do daily?

~~The two kinds of fragmentation~~

~~Why the fix makes things worse~~

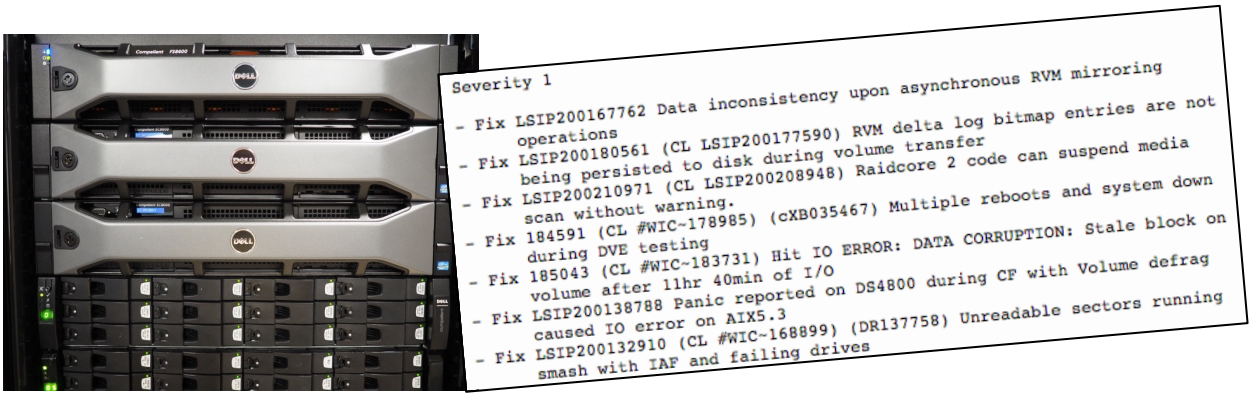
~~The better performance fix~~

~~When people insist on rebuilds~~

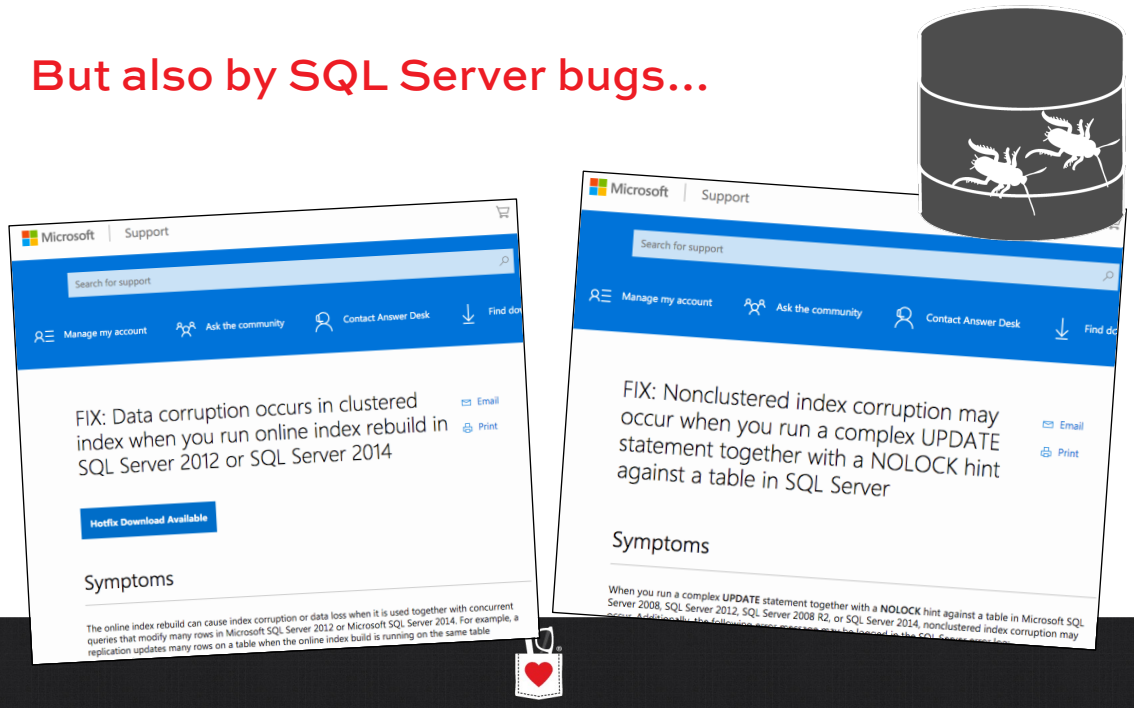
What to do overnight instead

People get fired for losing data.

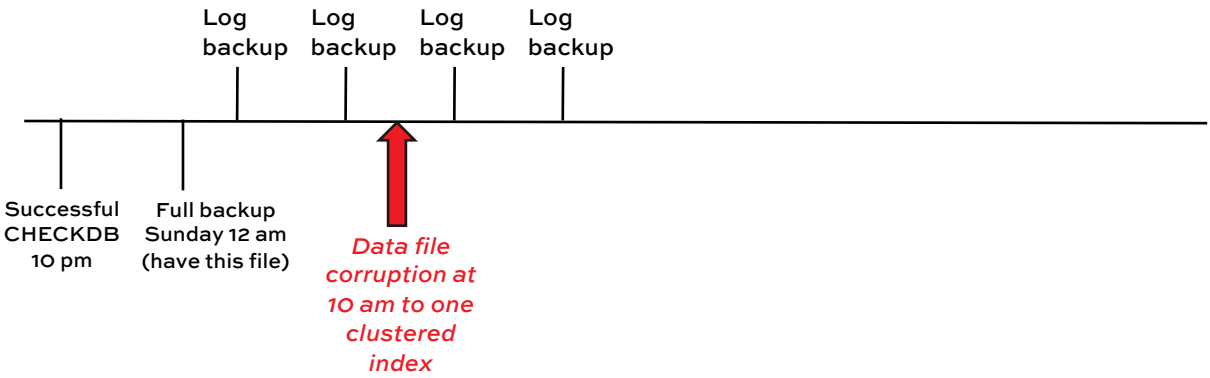
Data corruption is sometimes caused by storage failures and bugs...



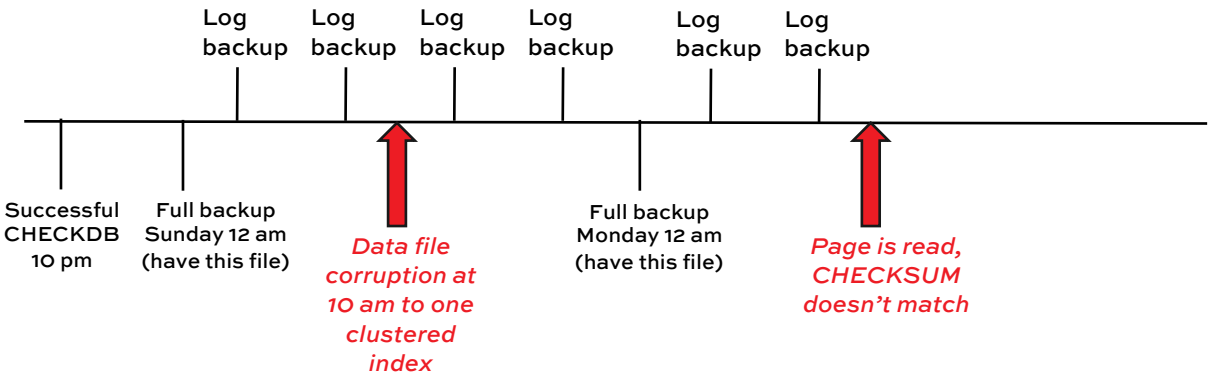
But also by SQL Server bugs...



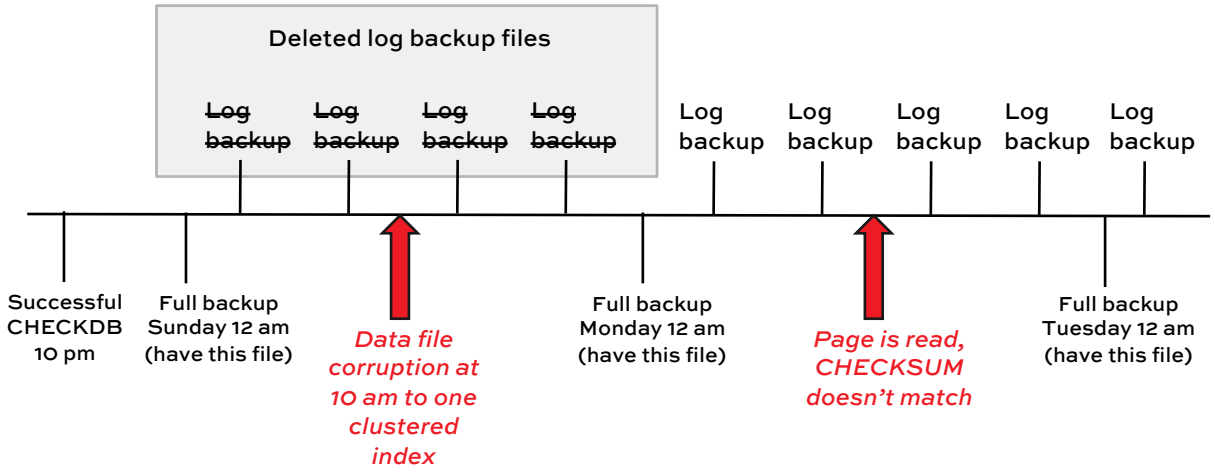
We get corruption sometime on Sunday



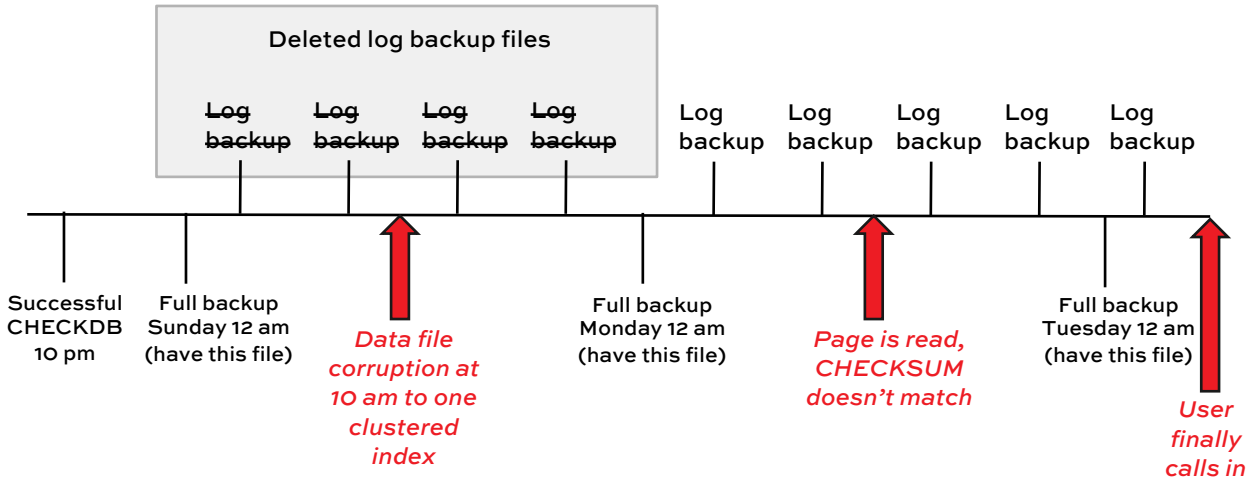
Corruption is detected – but not acted on



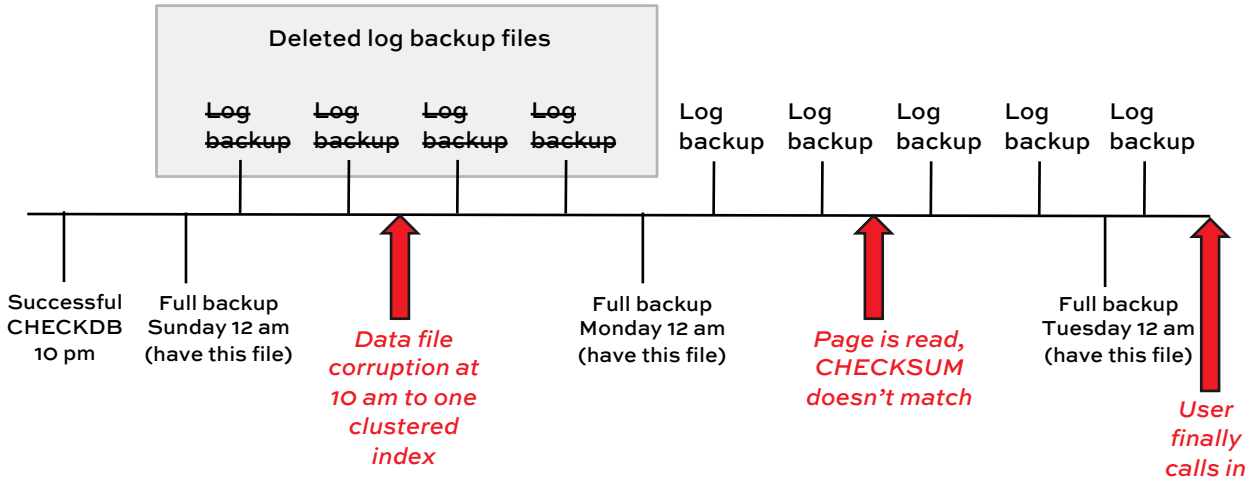
We only keep 24 hours of log backup files



Will you lose data? How much?



What could you do to lose less data?



Ways to lose less data

Set up alerts for corruption errors: https://BrentOzar.com/go/alerts

Keep transaction log backup files longer
(dictated by your RPO, RTO, and CHECKDB frequency)

Run CHECKDB more often (and act fast on the outputs)

The more of these you do, the better your chances are to keep your job.

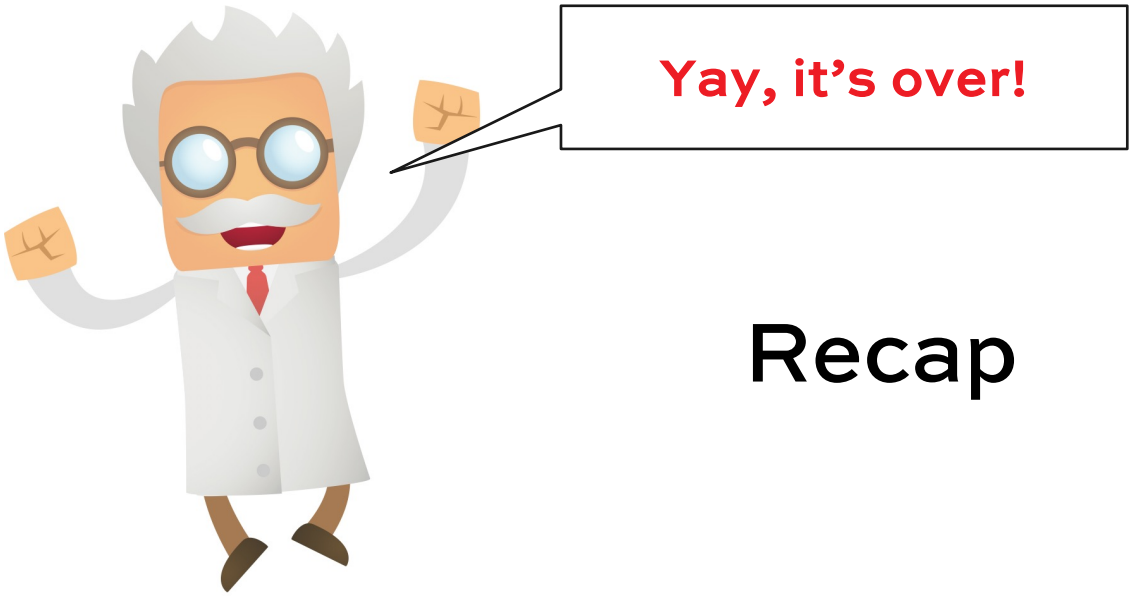
DBAs don't get fired
for slow performance.

DBAs get fired
for losing data.

Run CHECKDB as frequently as practical.

Forget shuffling indexes around every night.

It's time to run CHECKDB in that time window instead.



Recap

What you learned

External fragmentation = pages out of order on disk
(Only matters when you're reading from disk)

Internal fragmentation = empty space on pages
(Matters a lot, and fix this with judicious rebuilds)

Fill factor = internal fragmentation
(Set this, and you're making internal fragmentation worse)

Index maintenance is fine,
but it doesn't solve most wait types.
Find your top wait type with sp_BlitzFirst and focus on it.

What you'll do tomorrow

1. Set fill factor at the server level back to 100%.
2. Use sp_Blitz and sp_BlitzIndex to check for index fill factors <80%, and set those back up to 80% or higher (or 100%).
3. Rebuild your indexes once to atone for those past sins, getting them back up to 100% fill factor.
4. Monitor wait time per hour, and use the scientific method to improve it.