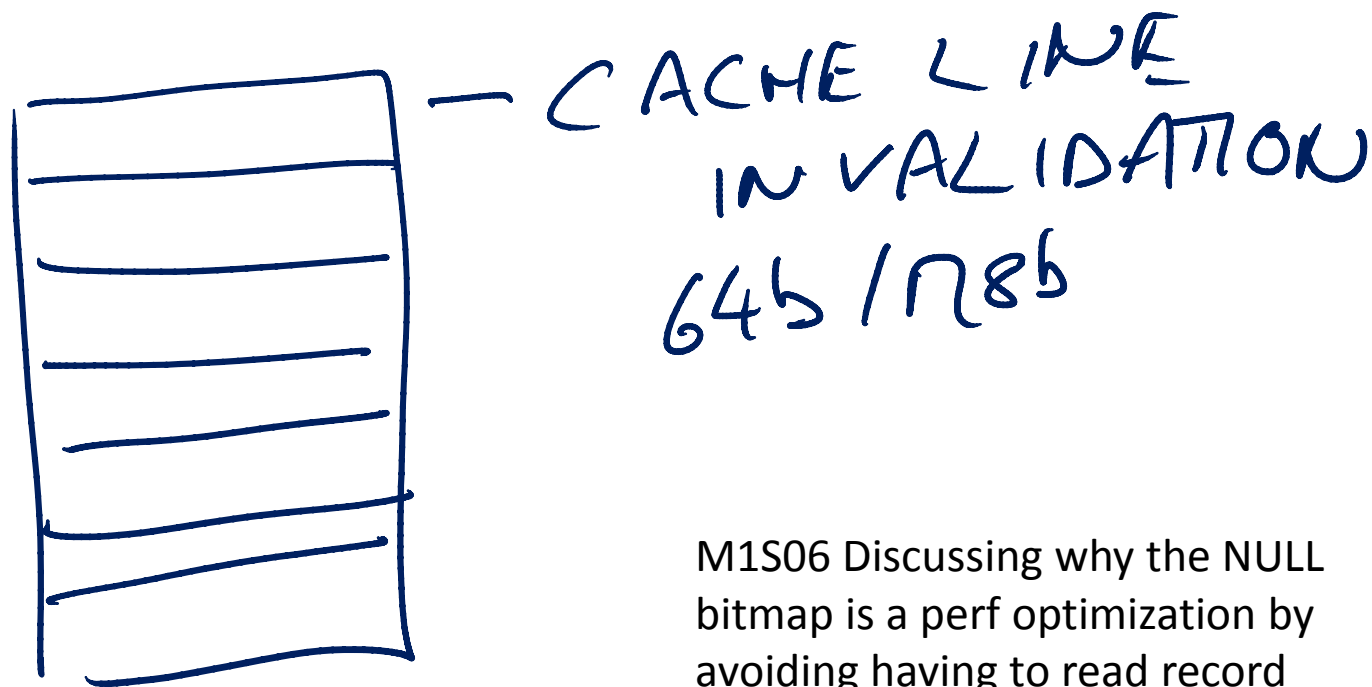
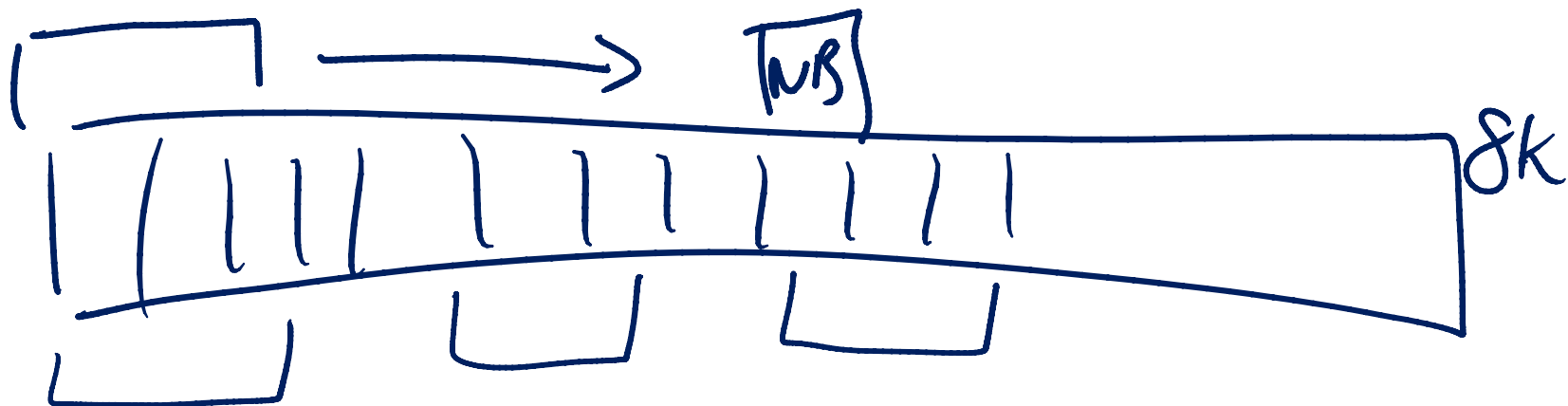
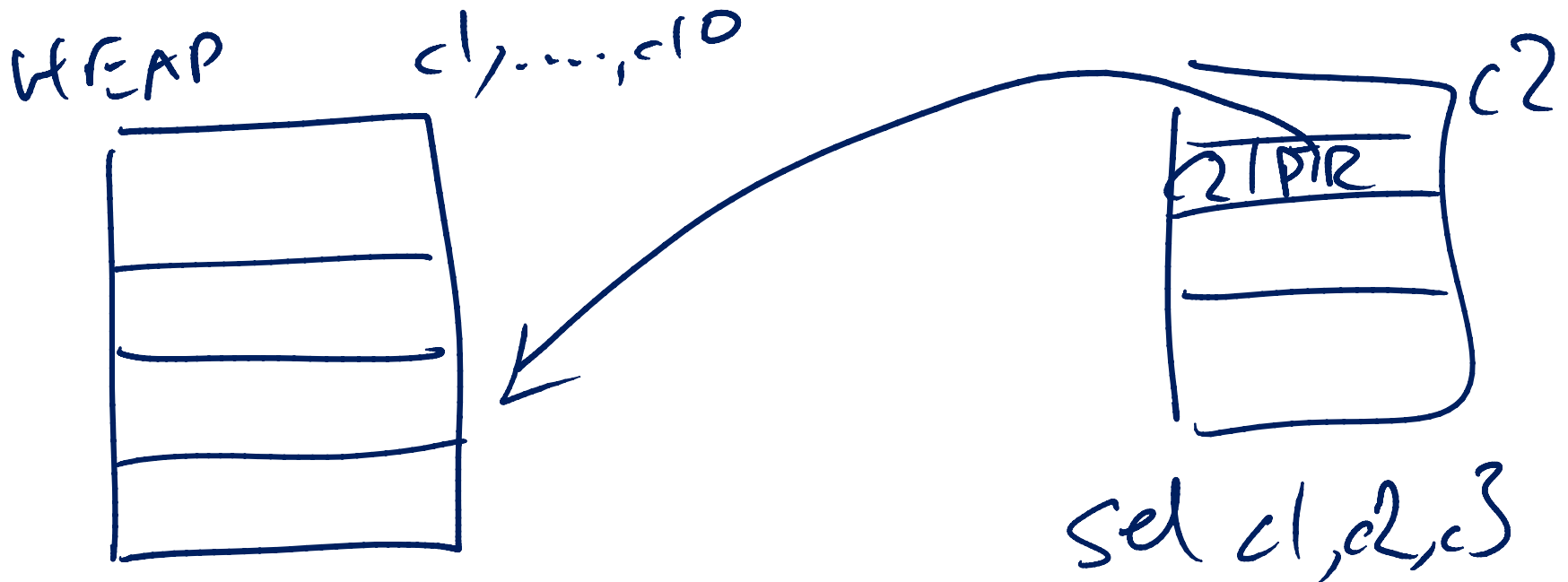




M1S06 Discussing why the NULL
bitmap is a perf optimization by
avoiding having to read record
parts into the CPU's L1 cache



PTR	
HEAP	PHYSICAL RID (FILE:PAGE:SLOT)
c_2	LOGICAL RID (CLUSTER KEY)

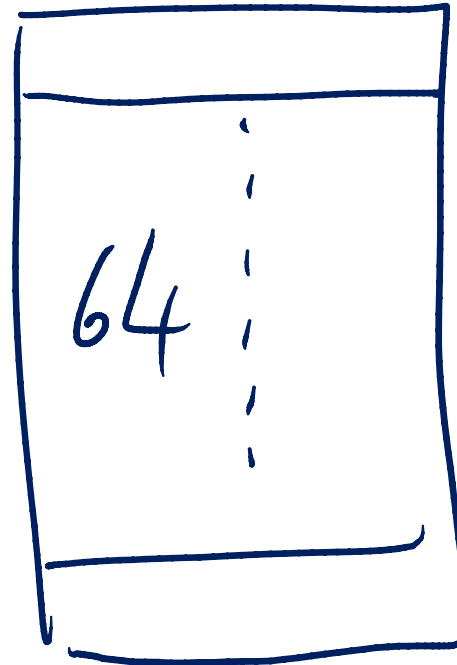
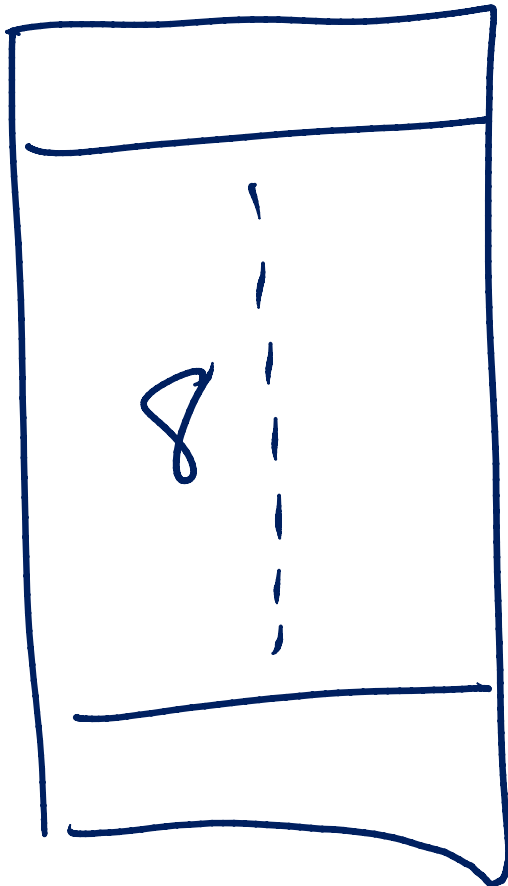
M1S09 Explaining NC index back-links to the data records as part of describing why forwarding/forwarded records exist. Without them, moving a heap row would entail changing all NC index records with a back-link to that row.

1K

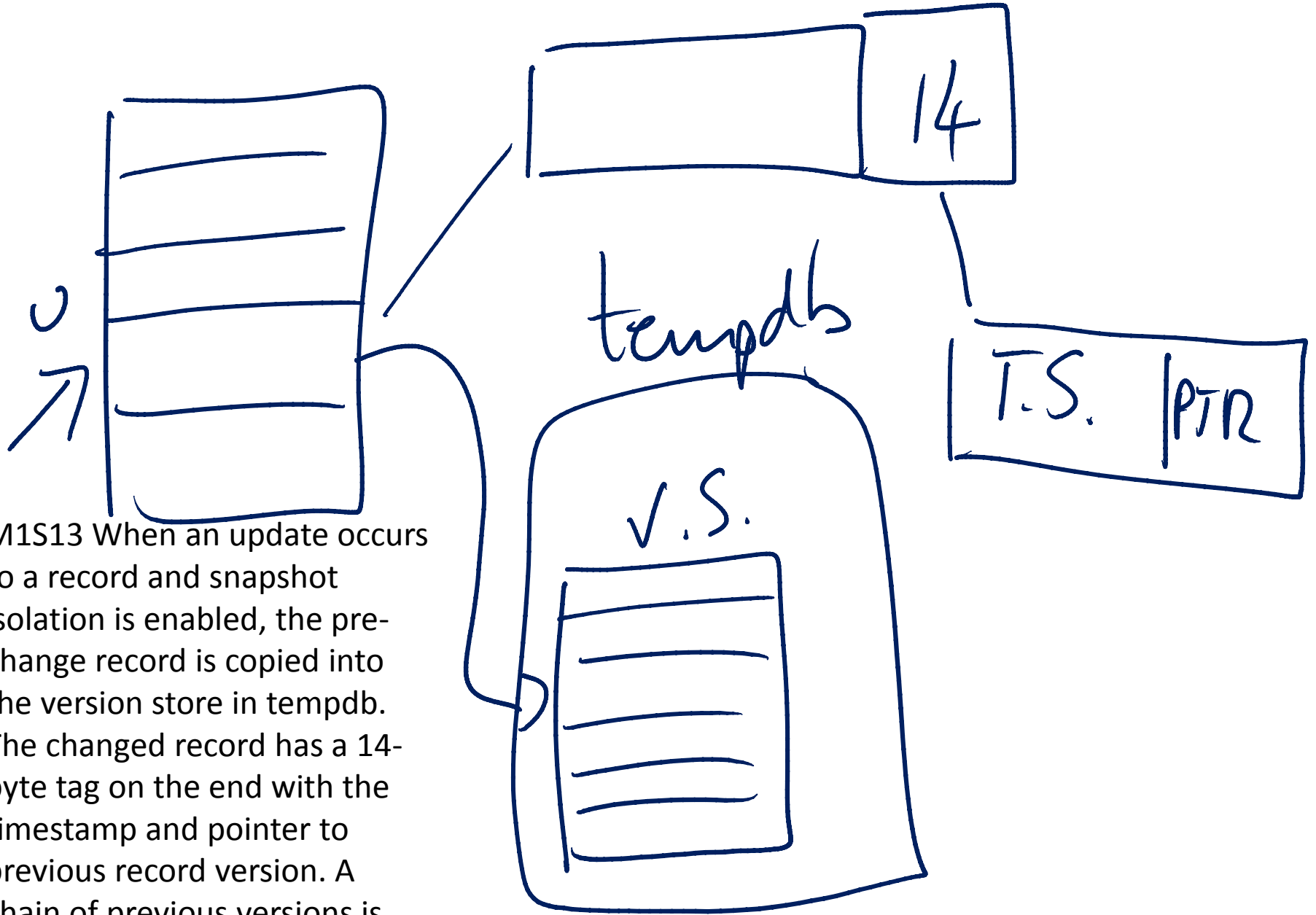
100	900	1%
-----	-----	----

100	24
-----	----

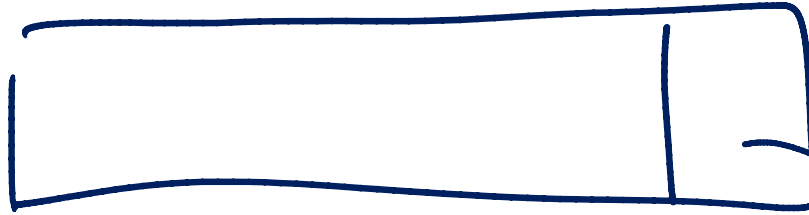
900



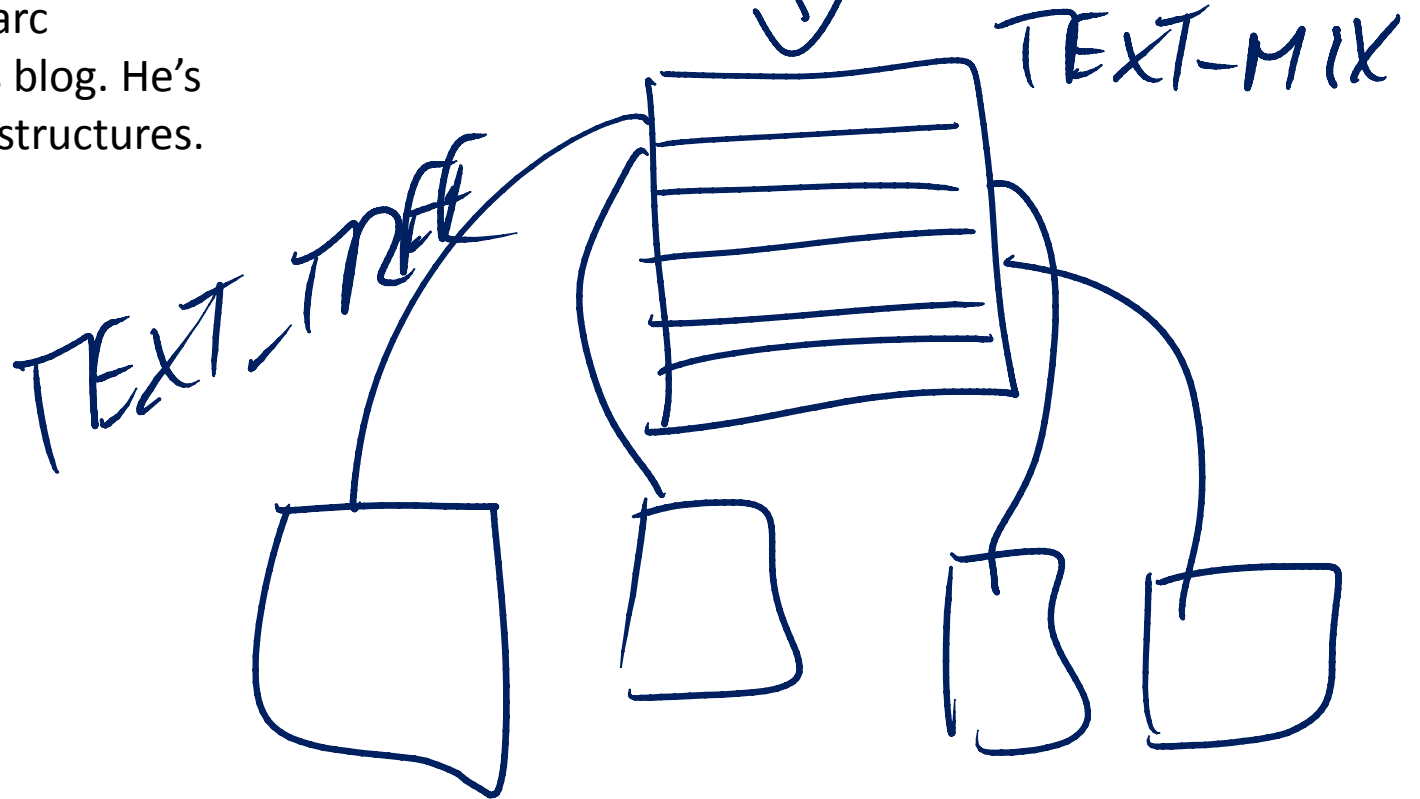
M1S12 Having a large LOB value in row that's not used much makes for large rows and low data density. Choose how to store LOB data wisely, either in-row, off-row, or in another table with a join.



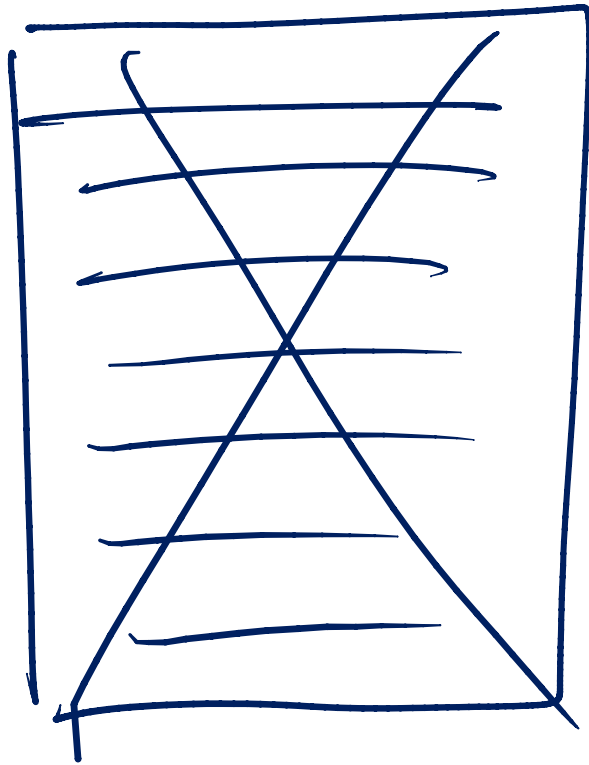
M1S13 When an update occurs to a record and snapshot isolation is enabled, the pre-change record is copied into the version store in tempdb. The changed record has a 14-byte tag on the end with the timestamp and pointer to previous record version. A chain of previous versions is possible.



M1S11 Google Marc
Rasmussen for his blog. He's
blogged text tree structures.



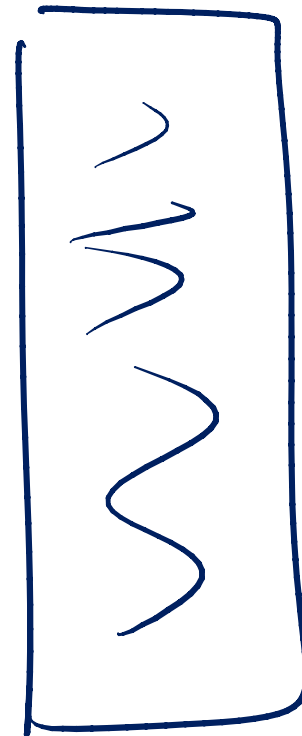
APPEND-ONLY-ALLOCATION-UNIT



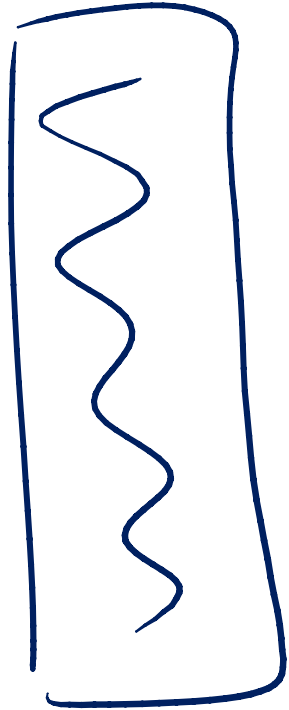
-4m



-3m



-2m

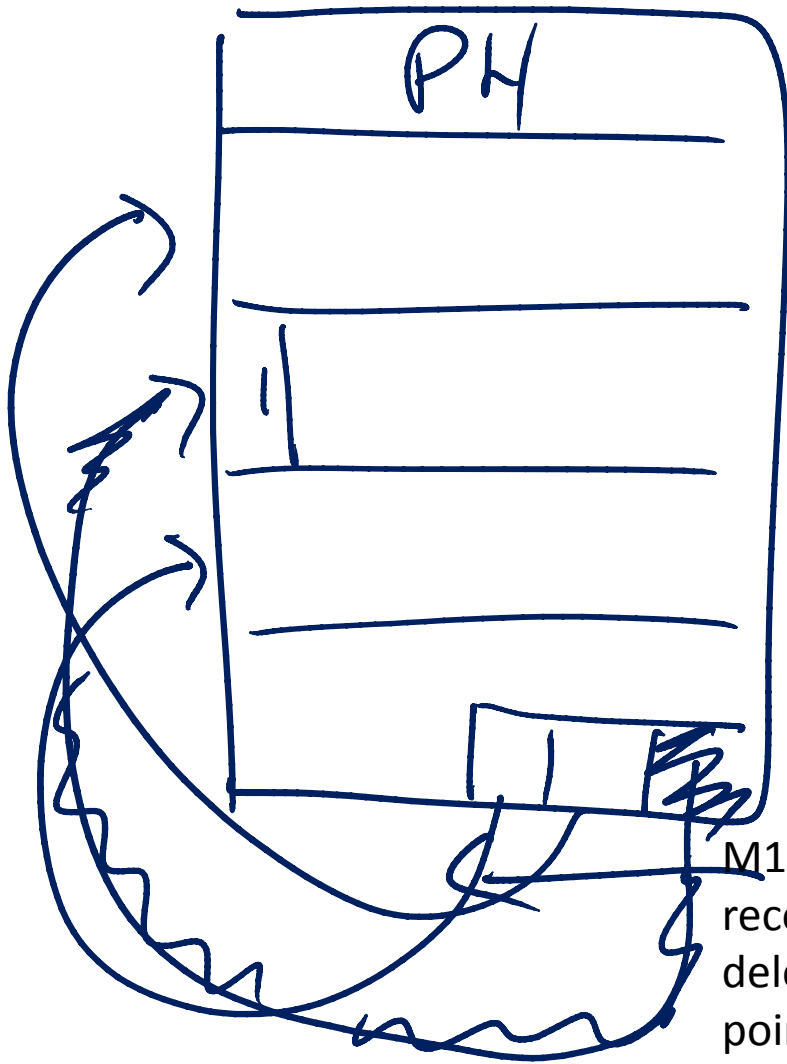


-1m

M1S13 and M2 S54 When an update occurs to a record and snapshot isolation is enabled, the ~~pre-change record~~ is copied into the version store in tempdb. The changed record has a 14-byte tag on the end with the timestamp and pointer to previous record version. A chain of previous versions is possible.



LOP_EXPUNGE
- GHOST



SLOT
ARRAY

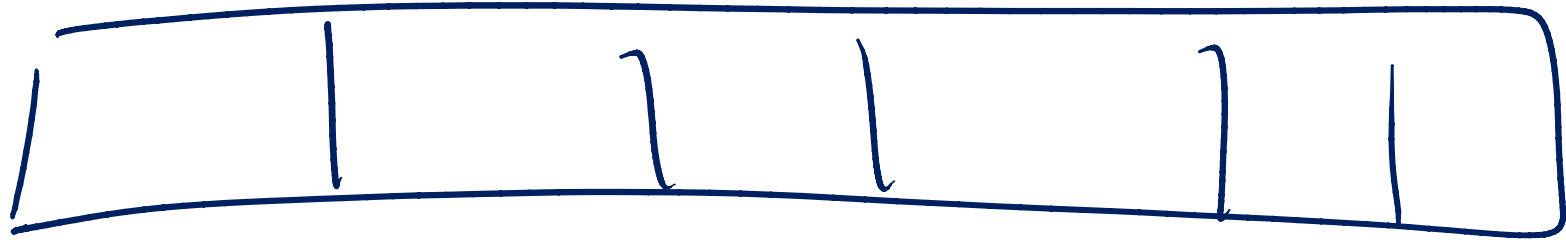
M1S14 When a record is deleted, it's marked as a ghost record. The ghost cleanup task does not physically delete the records – it removes the slot array entry that points to the record on the page. I.e. the record is unmarked as being a record and the area on the page is now free space – that just happens to have bits and bytes that used to be a valid record.

DBCC TRACEON

(662, -1)

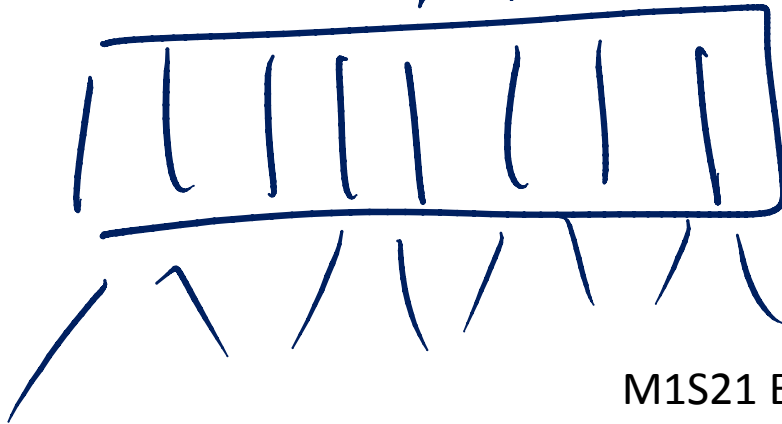


M1S14 For any trace flag that you want to effect background tasks, you must use the -1 in DBCC TRACEON to enable the trace flag globally instead of just for your connection.



0 - 7 8-15 16-23

MIXED



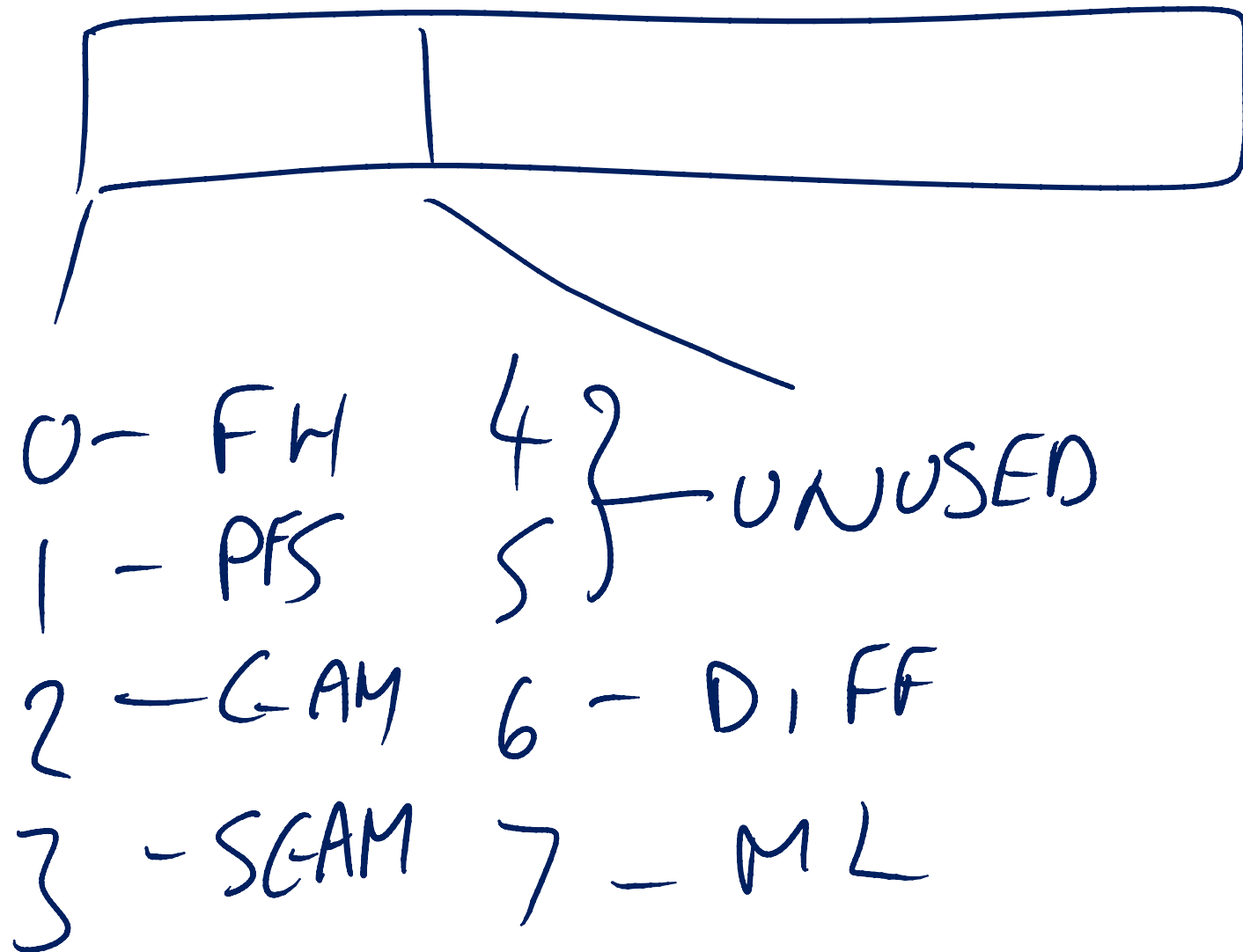
DEDICATED



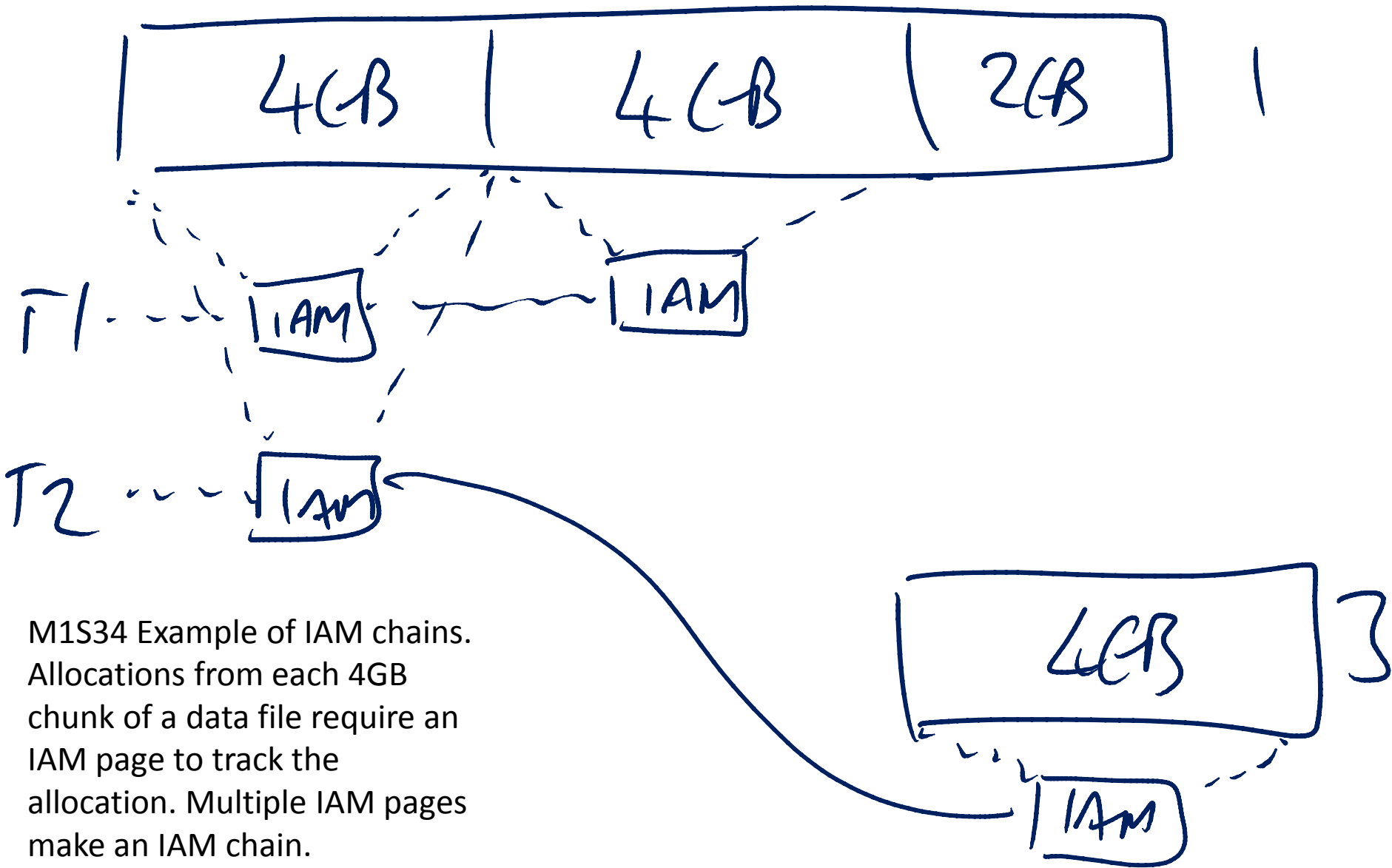
M1S21 Explaining the difference between mixed and dedicated extents. Mixed extent is one in which the 8 pages could be allocated to 8 different objects.

Dedicated extent is one where all 8 pages are reserved for exclusive use of a single object.

Later in the deck I refine 'object' to really be 'allocation unit'.



M1 The page types in the first extent in a data file



M1S34 Example of IAM chains.
Allocations from each 4GB
chunk of a data file require an
IAM page to track the
allocation. Multiple IAM pages
make an IAM chain.

SALES DB

1 TB SALES

2012 - - - - - 2016

M2S6 Splitting a large database up into multiple filegroups can enable you to do small, targeted restores or even a partial restore from scratch – in the event that the database is damaged or destroyed.

PARTIAL
RESTORE

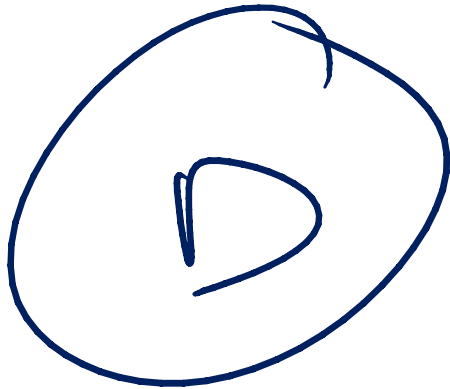
TABLE - PARTITIONED VIEW

OR TABLE PARTITIONING

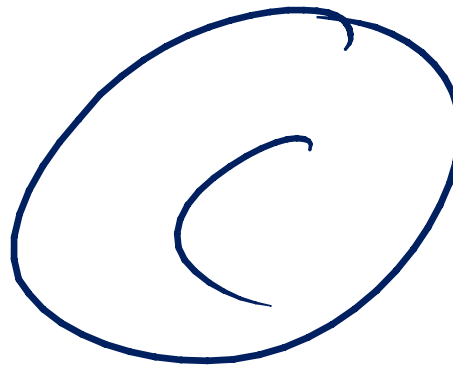
P

2016

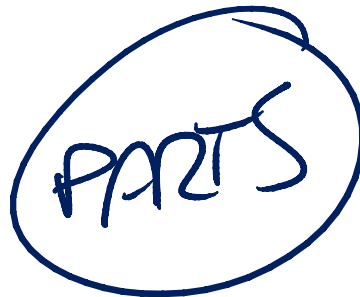
- - - - - 2012



D

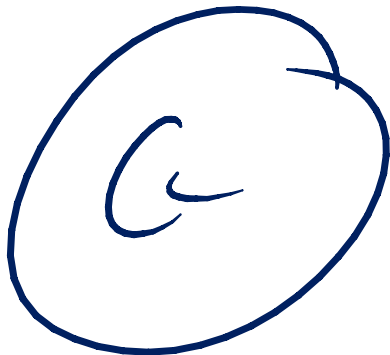


C

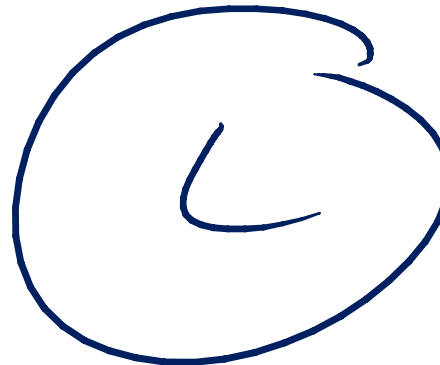


PARTS

M2S6 Explaining some ways to split up a database to allow targeted restore. Consider time zone, population density, or other demographics.



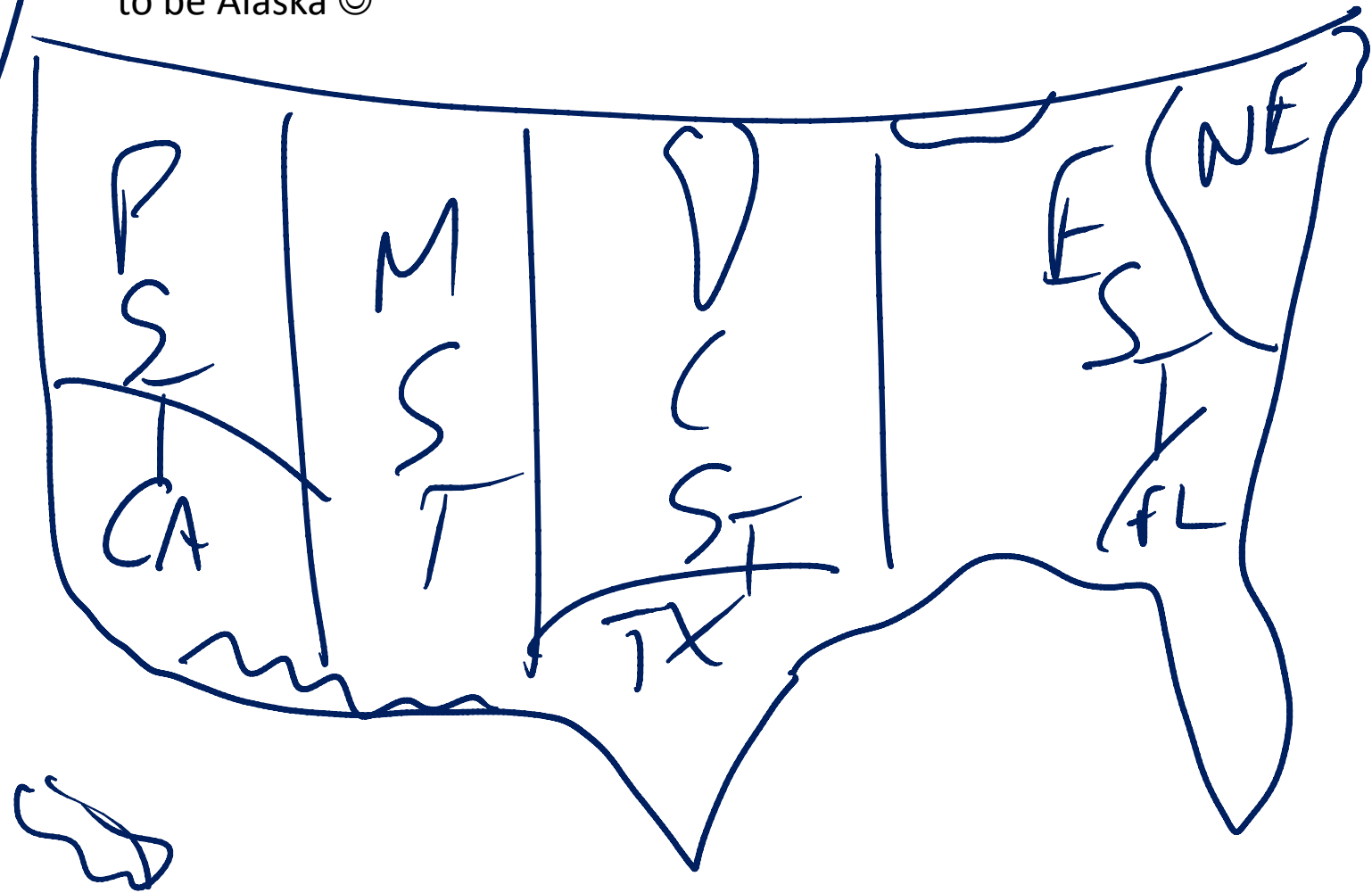
G



L

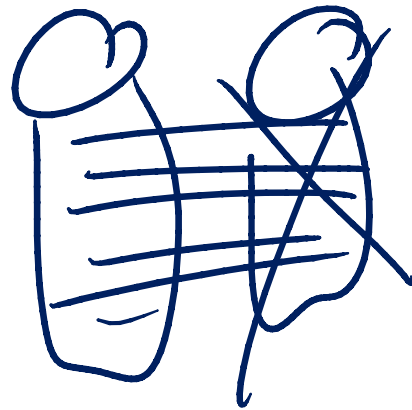
← This is
meant
to be Alaska 😊

7AM - 6PM



M2S6 Explaining some ways to split up a database to allow targeted restore. Consider time zone, population density, or other demographics.

RO STRIPING



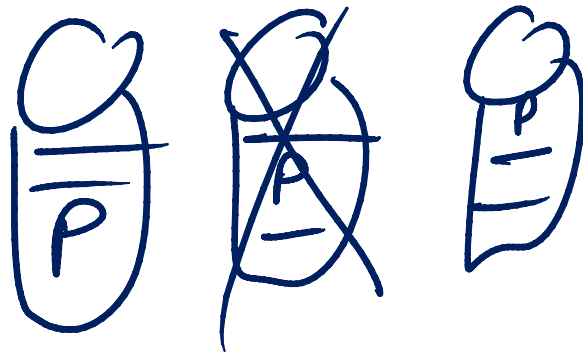
http://en.wikipedia.org/wiki/Standard_RAID_levels

RAID 1 - MIRRORING



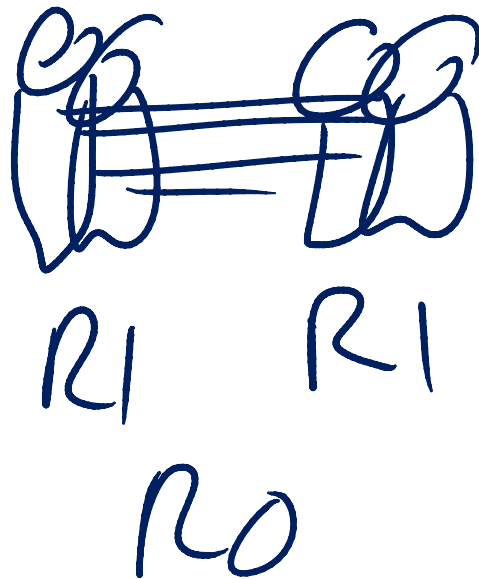
http://en.wikipedia.org/wiki/Standard_RAID_levels

RS - ROTATING PARITY



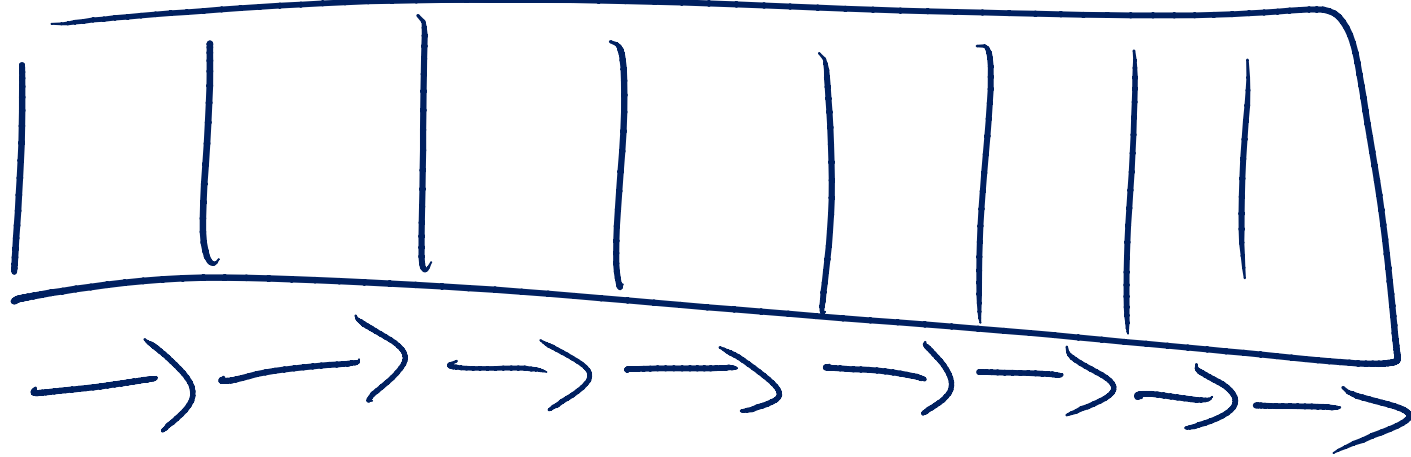
http://en.wikipedia.org/wiki/Standard_RAID_levels

RAID 0 ~~STR~~ MIRRORING
+ STRIPING

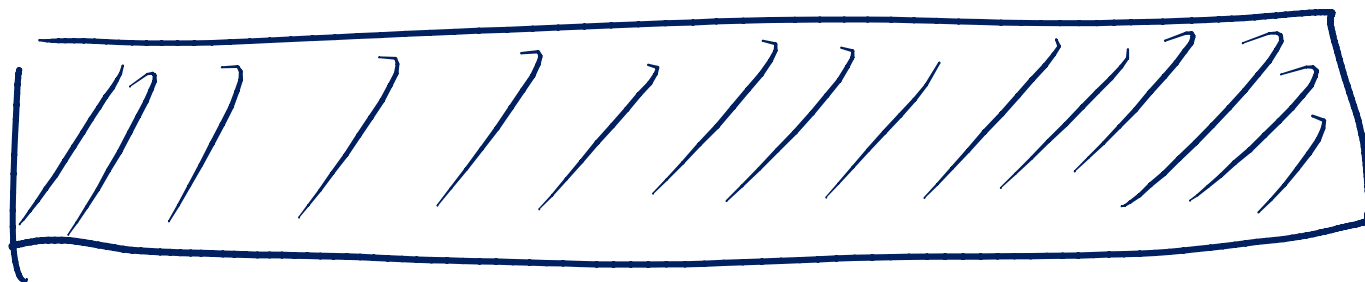
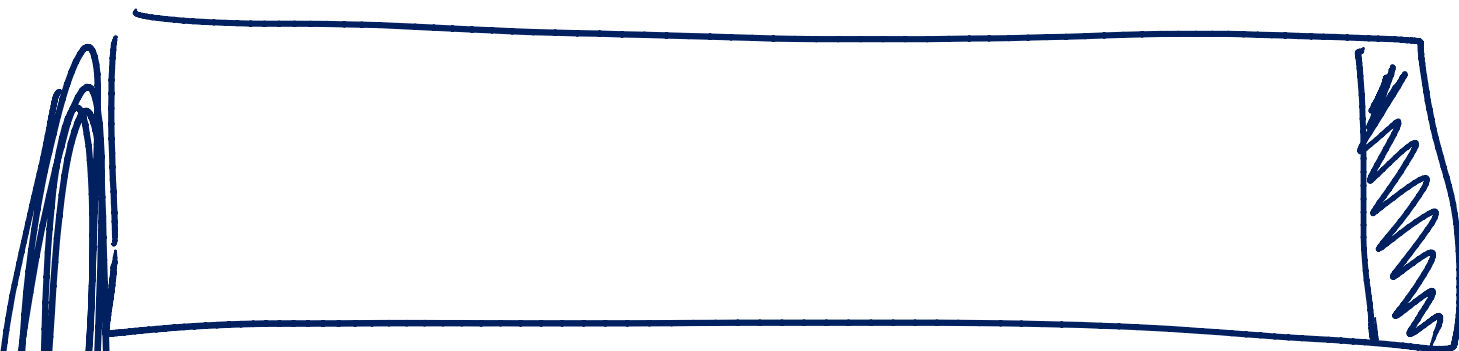


http://en.wikipedia.org/wiki/Standard_RAID_levels

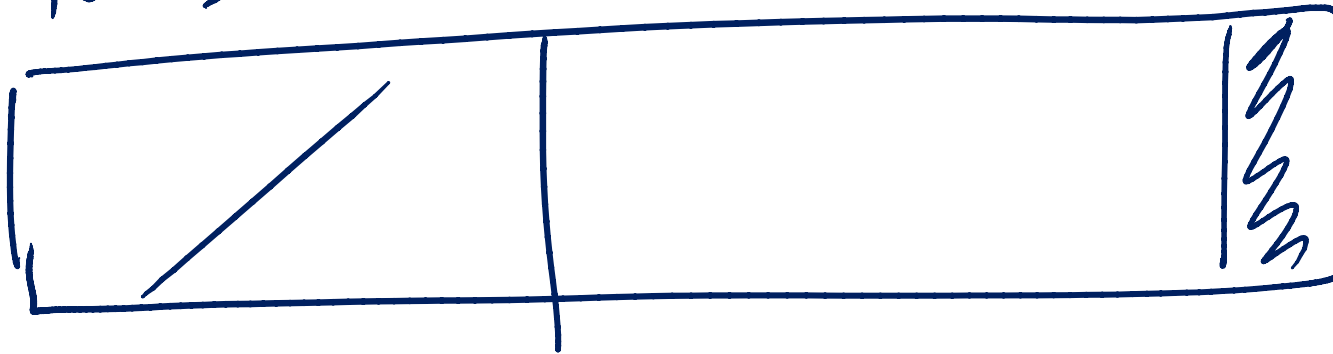
L:



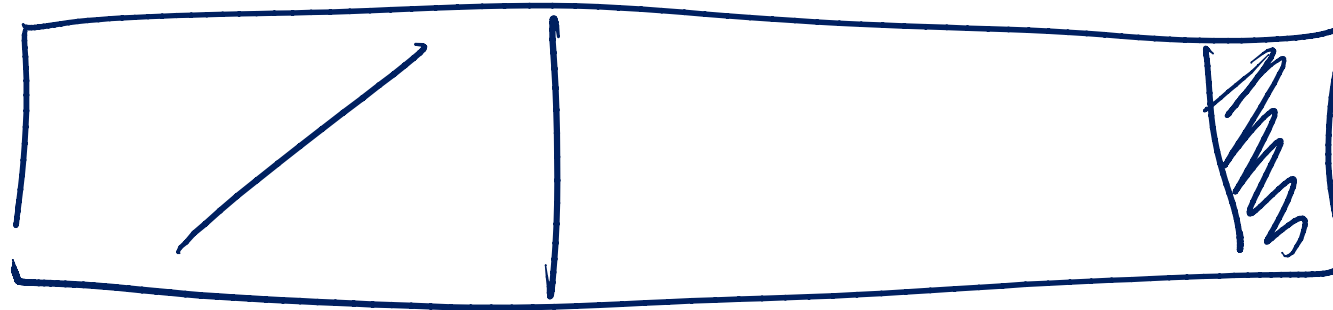
M2S10 If you put all your log files on one volume, its I/O characteristics become random. This may or may not become a performance problem, depending on the capabilities of the I/O subsystem.



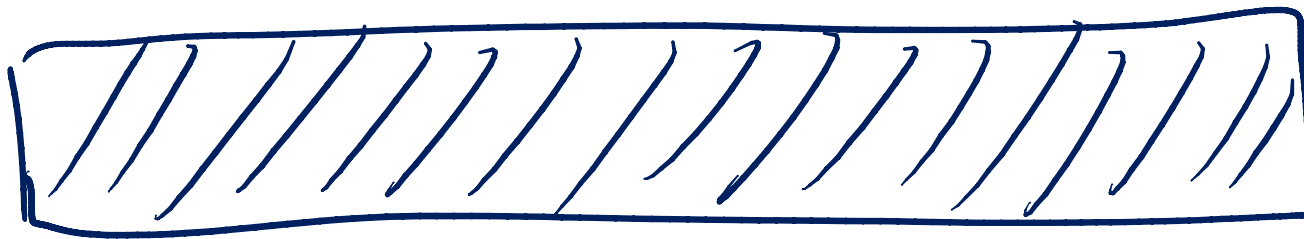
10GB



8/60 8/9 11



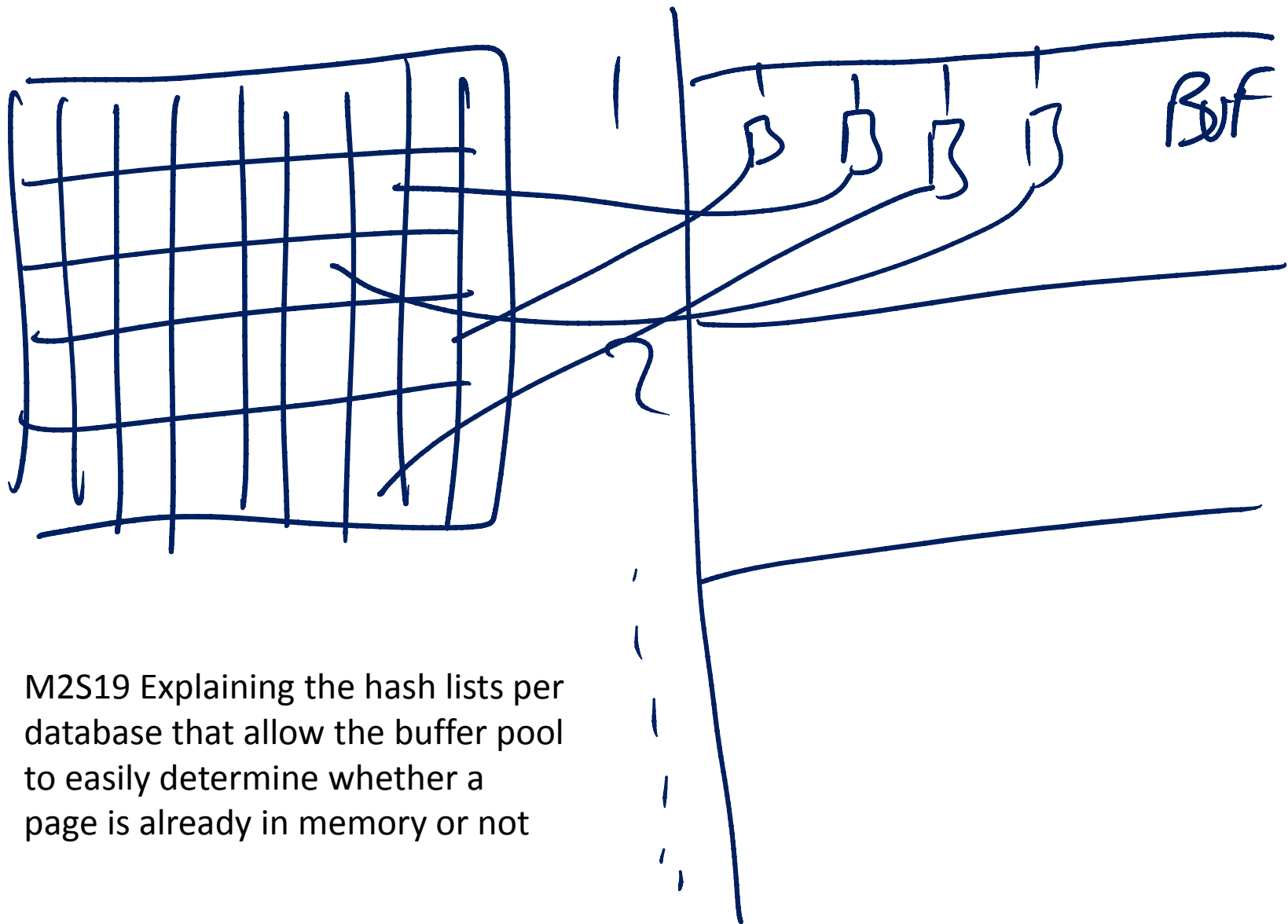
8/60 8/59 11



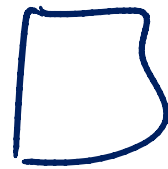
1 8/70

M2S18 Explaining round-robin allocation and proportional fill. Each file has a proportional fill weighting. Allocations happen proportionally more frequently from files with more free space.

PROP FILL
WEIGHTING



M2S19 Explaining the hash lists per database that allow the buffer pool to easily determine whether a page is already in memory or not



LRU-K, K=2



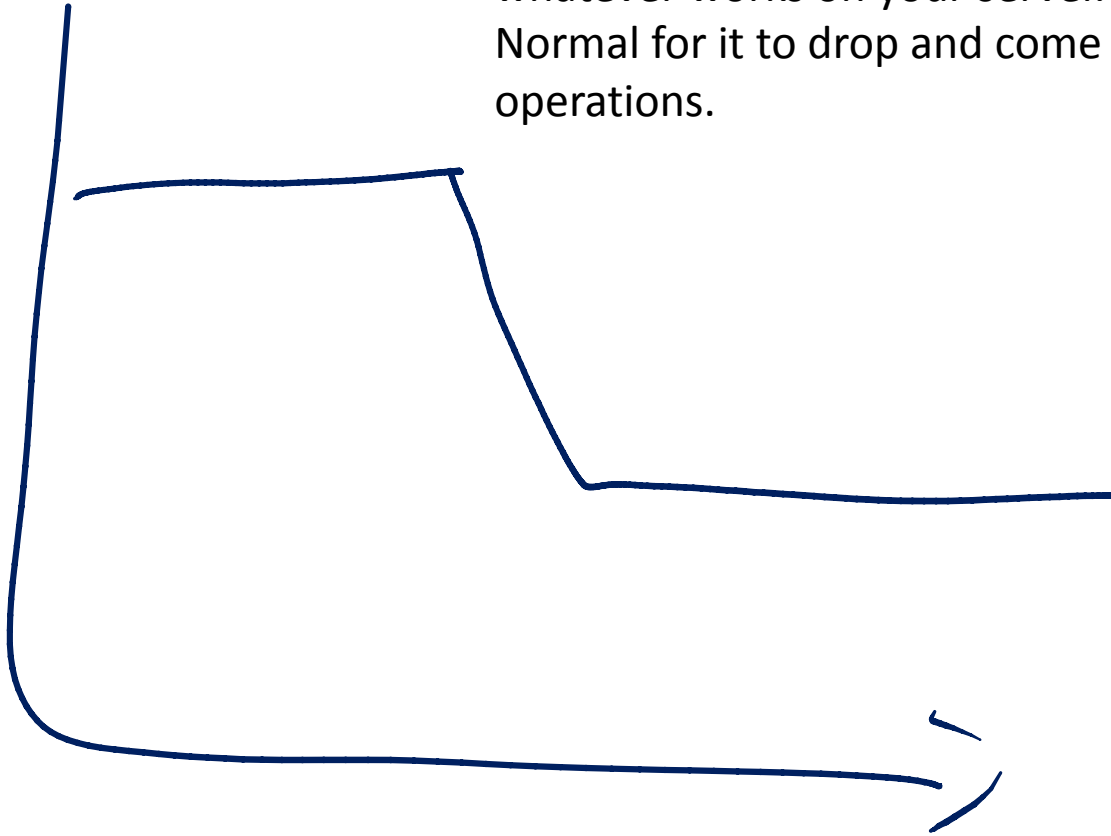
P.L.E.

LAZYWRITER

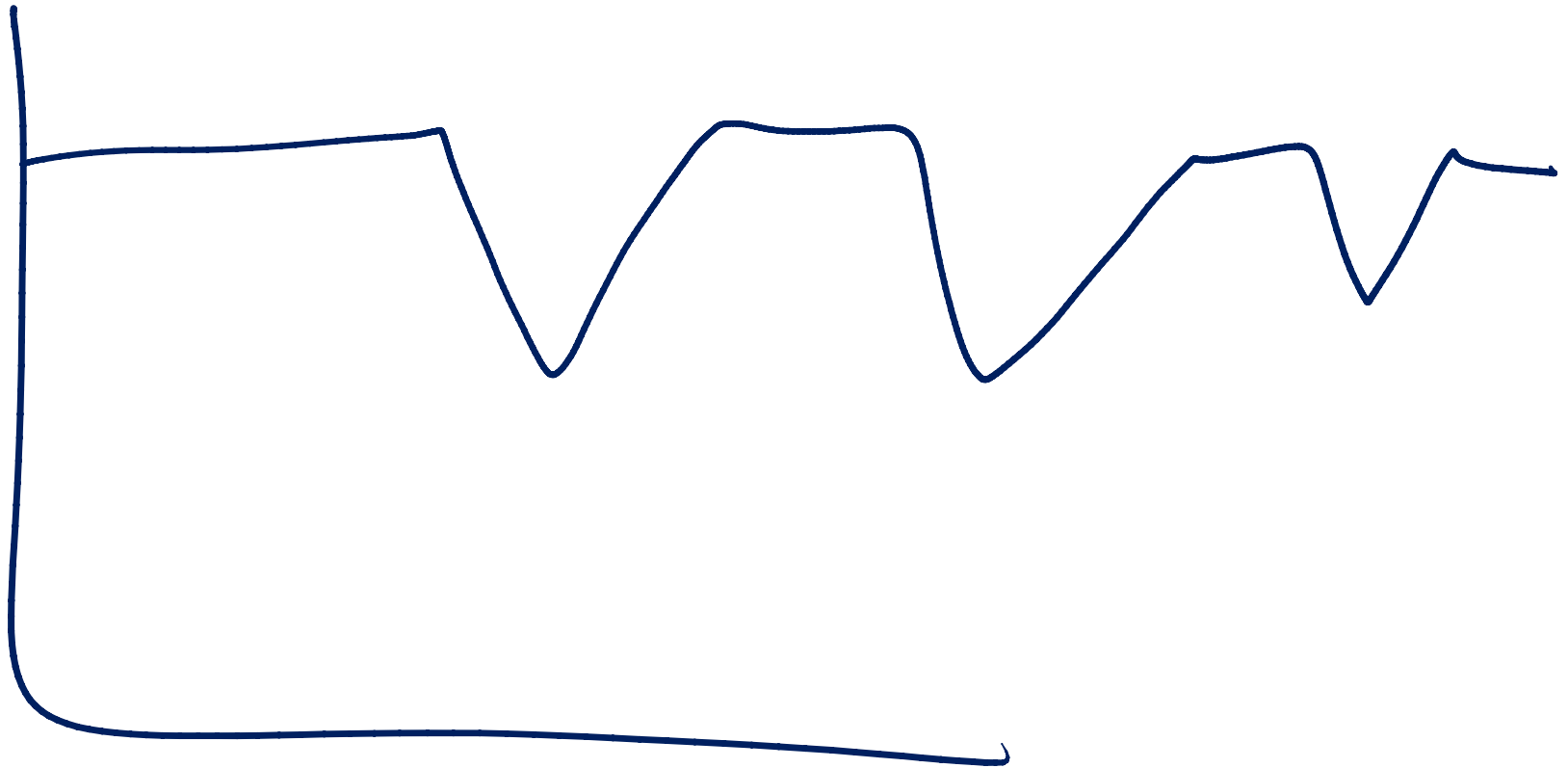
M2S19: The buffer pool implements a Least Recently Used algorithm, based on last two times the page was accessed. Done by the Lazy Writer background task.

P.L.E. ≤ 300

M2S19: PLE of 300 is an old, old threshold. PLE should be whatever works on your server. Problem if it drops and stays low. Normal for it to drop and come back up again after some operations.

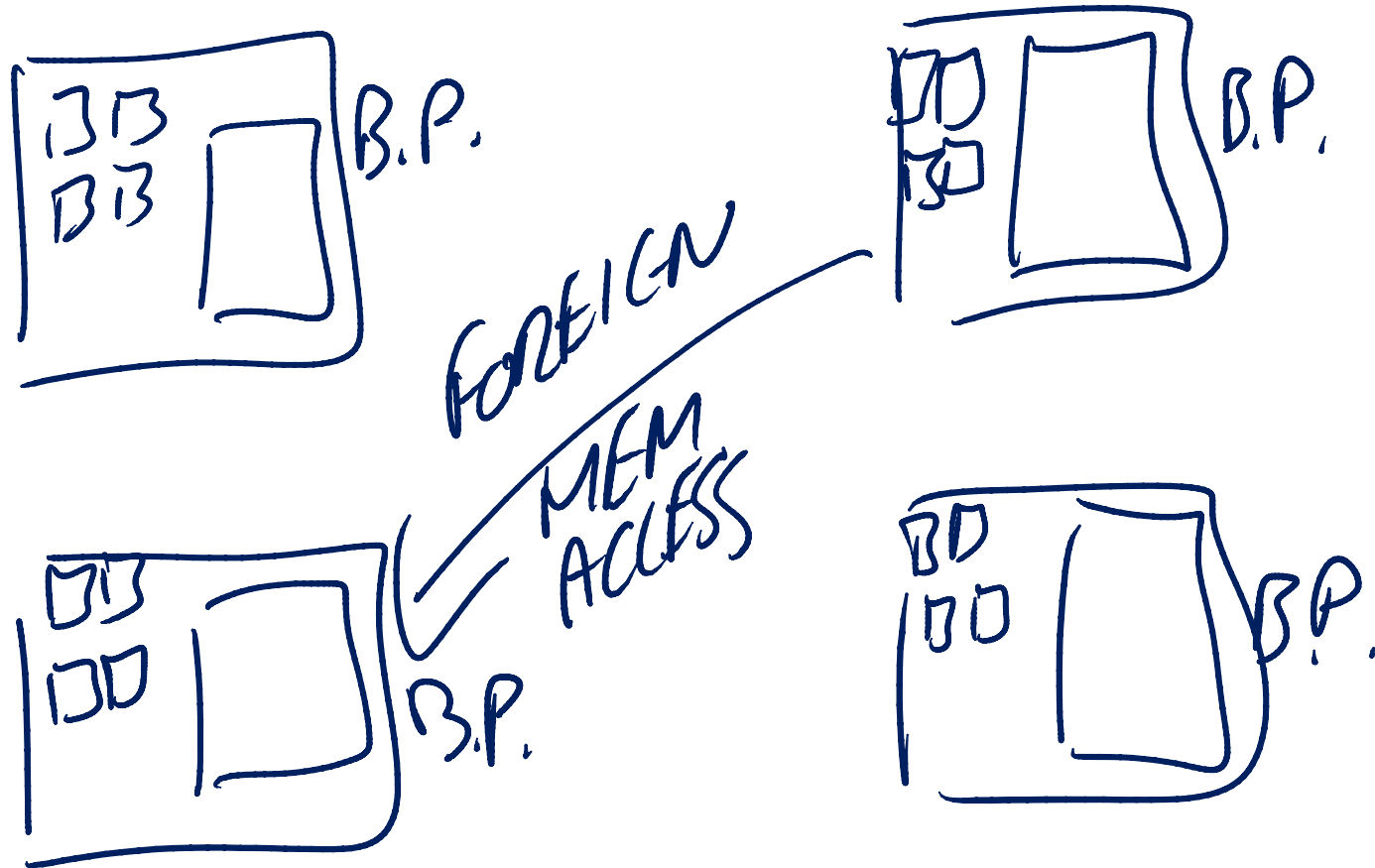


M2S19: PLE of 300 is an old, old threshold. PLE should be whatever works on your server. Problem if it drops and stays low. Normal for it to drop and come back up again after some operations.



NUMA

BUFFER NODE: PLE



M2S19: NUMA – Non Uniform Memory Access. Each processor (or group) has local memory, containing a partition of the buffer pool. 4 NUMA nodes = 4 equal-size partitions of the buffer pool.

NUMA - Sys. km.os-nodes

BUFFER MGR PLE.

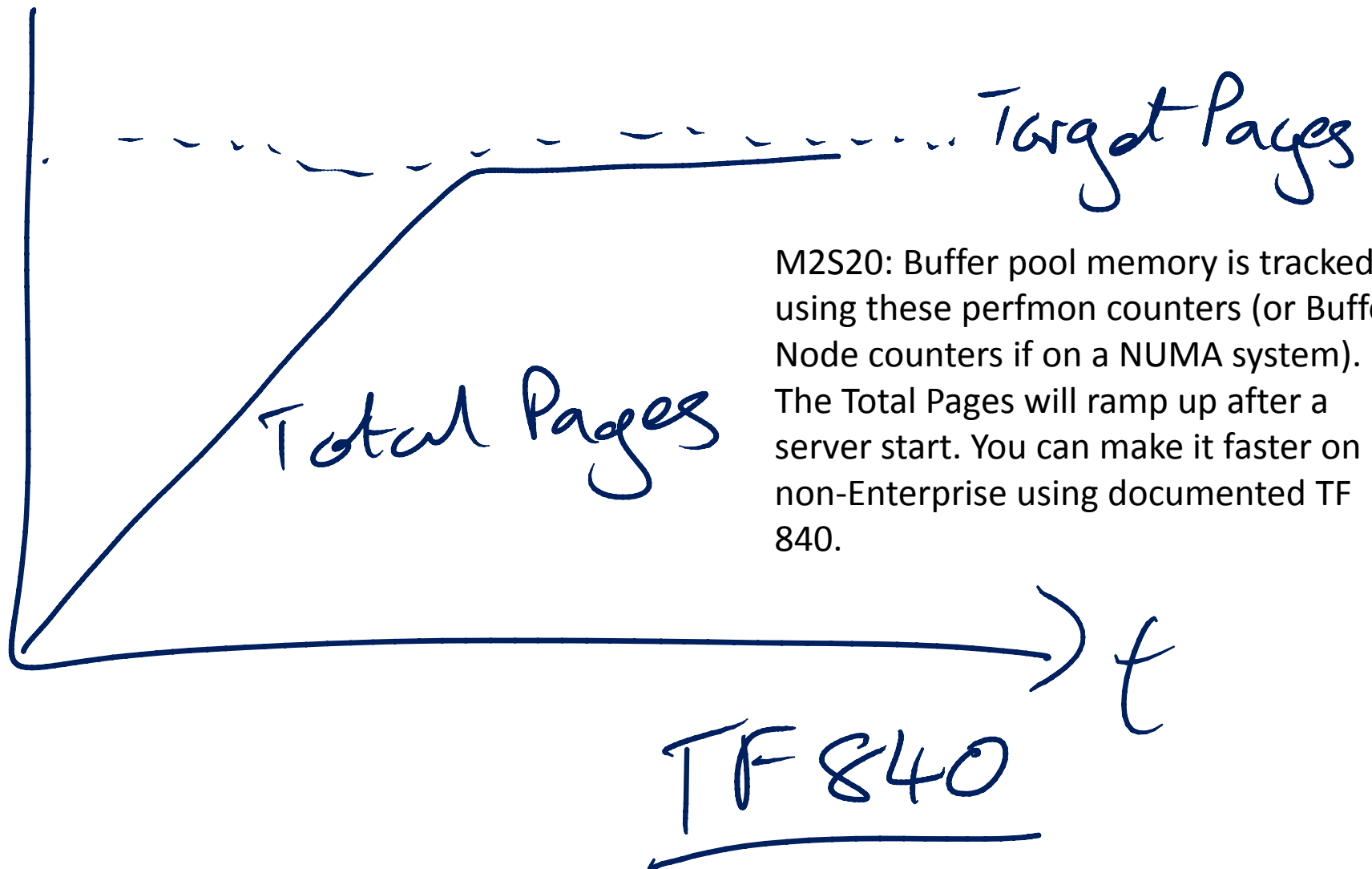
X = HARMONIC MEAN

BUFFER NODE PLE

✓ M2S19: If you have NUMA, you need to monitor the PLE in each Buffer Node perfmon object, not the Buffer Manager PLE.

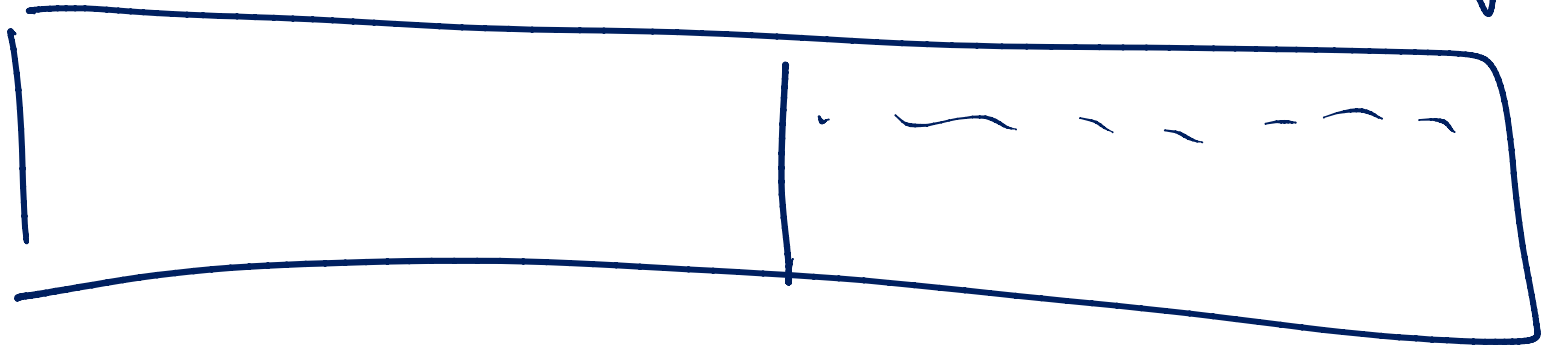
||
||
||

See <http://www.sqlskills.com/blogs/paul/page-life-expectancy-isnt-what-you-think/>



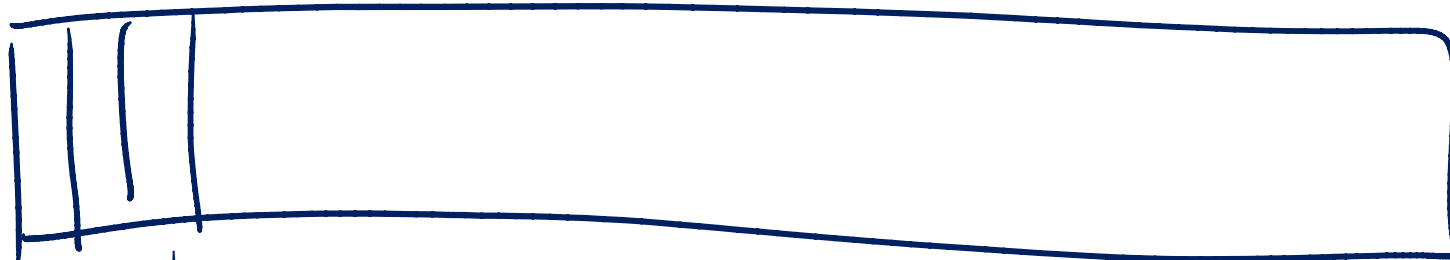
M2S20: Buffer pool memory is tracked using these perfmon counters (or Buffer Node counters if on a NUMA system). The Total Pages will ramp up after a server start. You can make it faster on non-Enterprise using documented TF 840.

SETFILEVALIDDATA



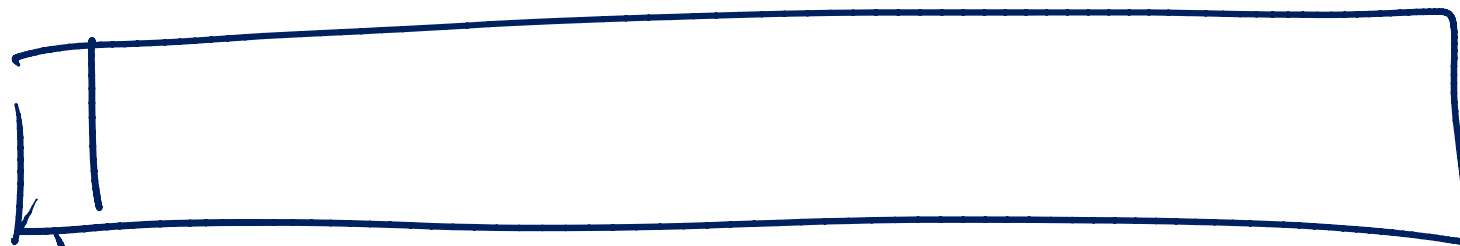
HIGH
WATER
MARK

M2S22 NTFS will not 'trust' a newly added portion of the file until the 'high water mark' of the file is advanced to include that section. This is usually done by writing zeroes to the file, but Instant File Initialization does it by calling the SetFileValidData API to do it 'instantaneously'.

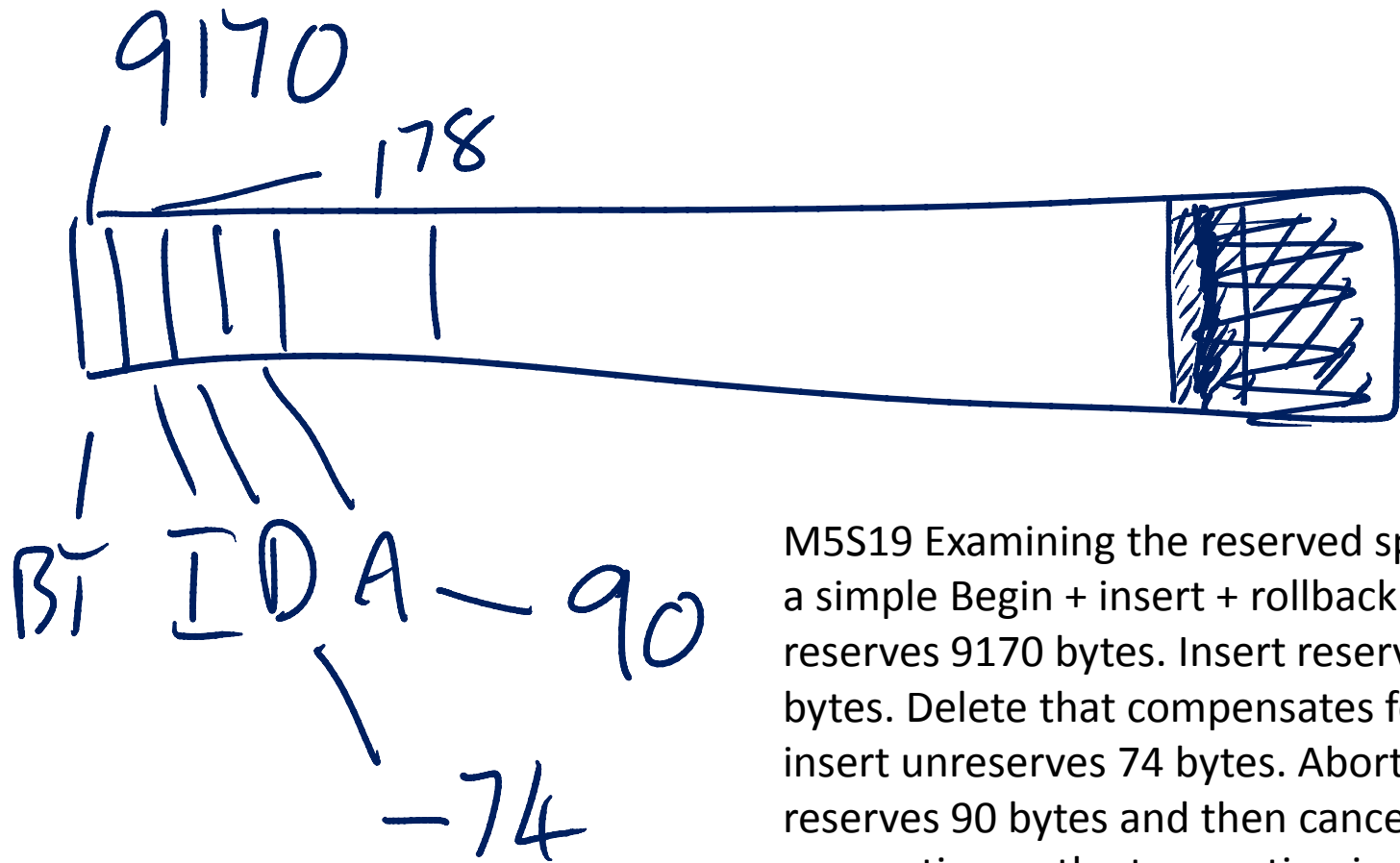


PFS
~~SGAM~~ T1118
2:1:1 ~~2:1:3~~

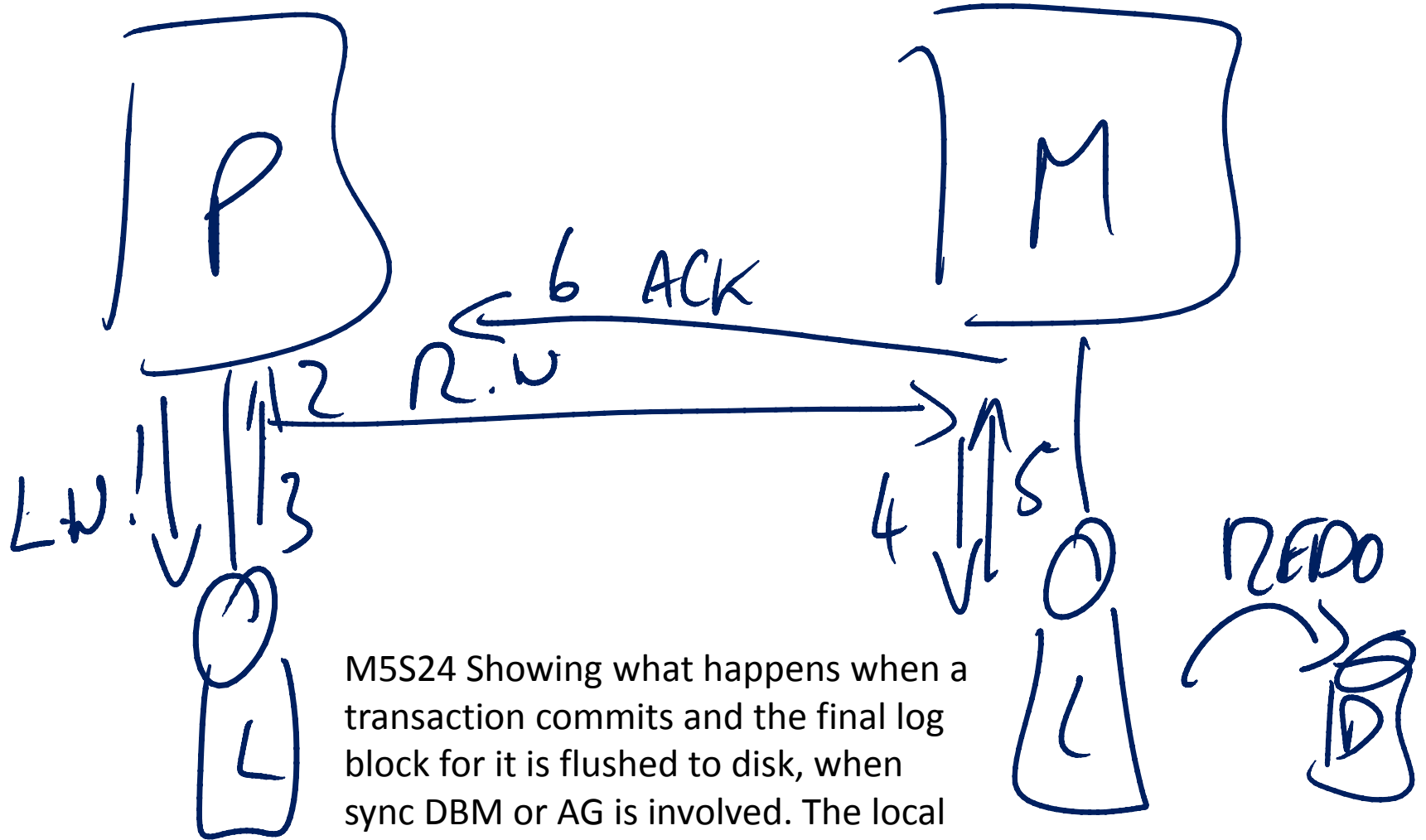
M2S59 Explaining PFS and SGAM contention in tempdb and how adding multiple files spreads the contention over multiple files, thus reducing it.



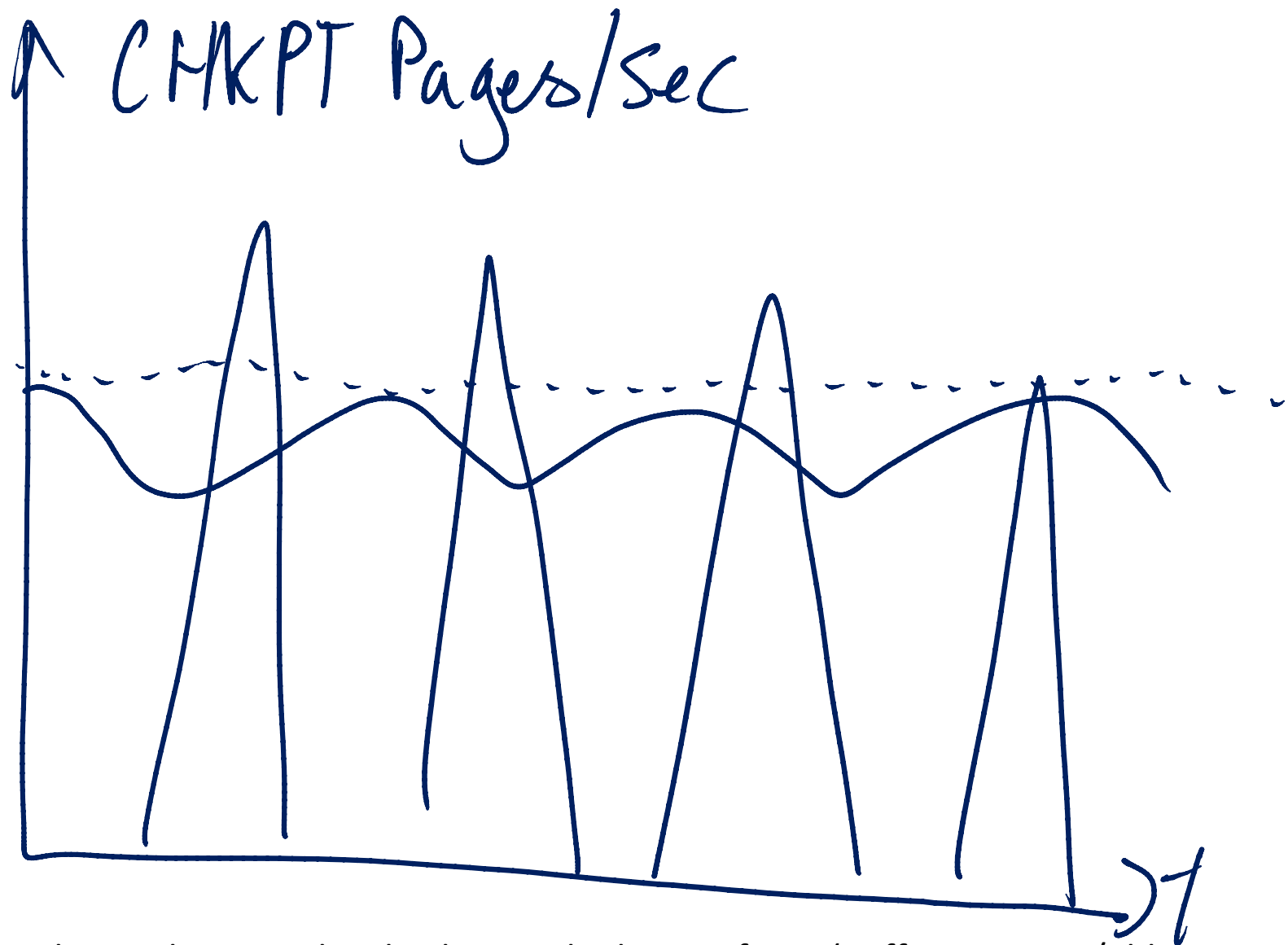
PFS 2:3:1



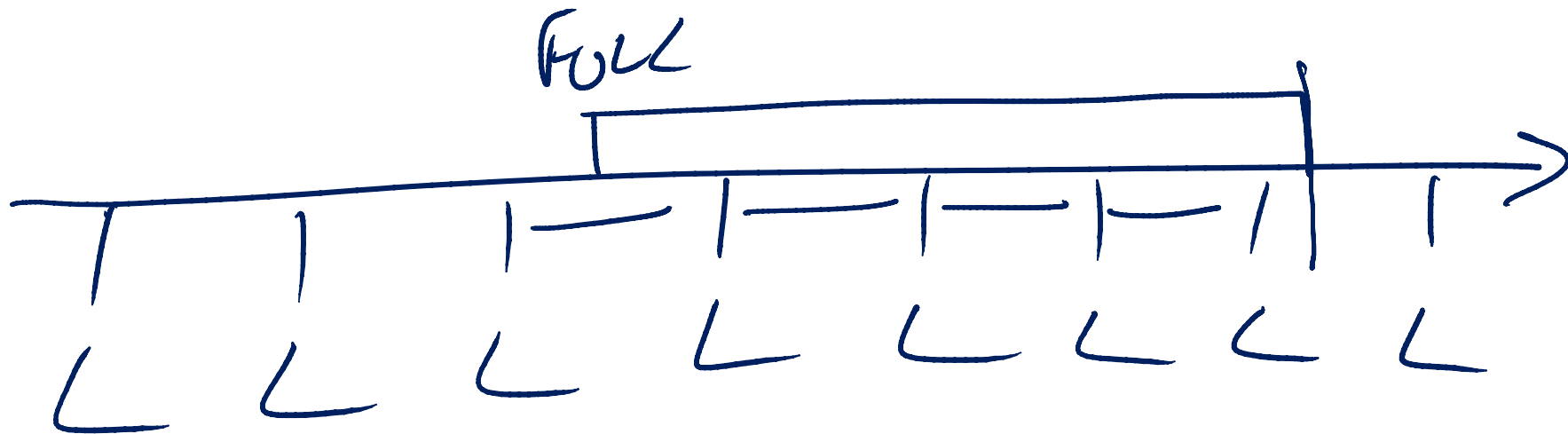
M5S19 Examining the reserved space for a simple Begin + insert + rollback. Begin reserves 9170 bytes. Insert reserves 178 bytes. Delete that compensates for the insert unreserves 74 bytes. Abort reserves 90 bytes and then cancels entire reservation as the transaction is done. The reservation comes from the unused space at the 'end' of the log.



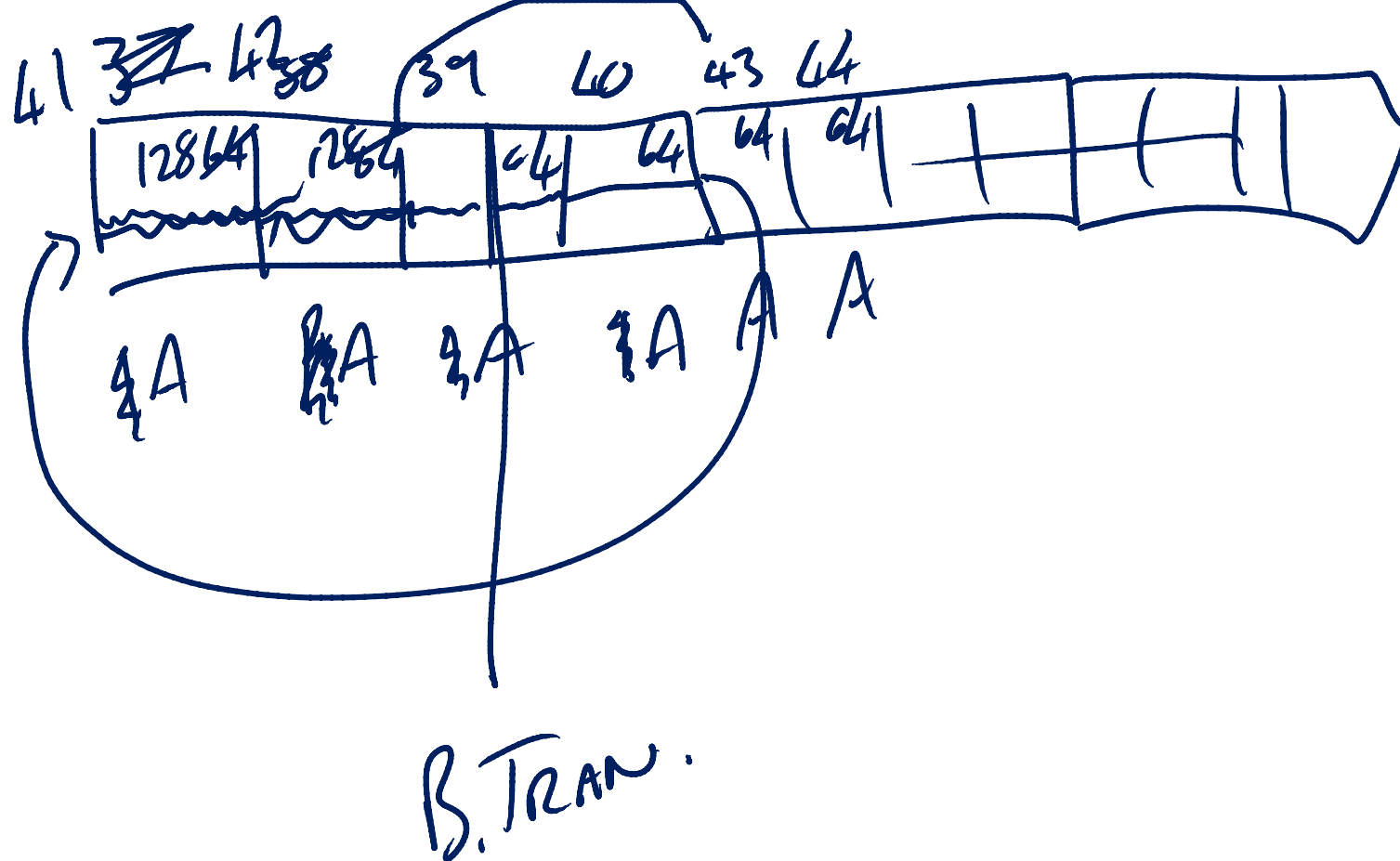
M5S24 Showing what happens when a transaction commits and the final log block for it is flushed to disk, when sync DBM or AG is involved. The local I/O will complete first (3) but the transaction commit cannot complete until the mirror acknowledges the remote I/O (6).



M5S33 Explaining how regular checkpoints look in perfmon (Buffer Manager/Chkpt pages/sec counter) and how they may exceed I/O capacity (dotted line). More frequent manual checkpoints, or using indirect checkpoints in SQL 2012+, can mean more constant, lower IO load.



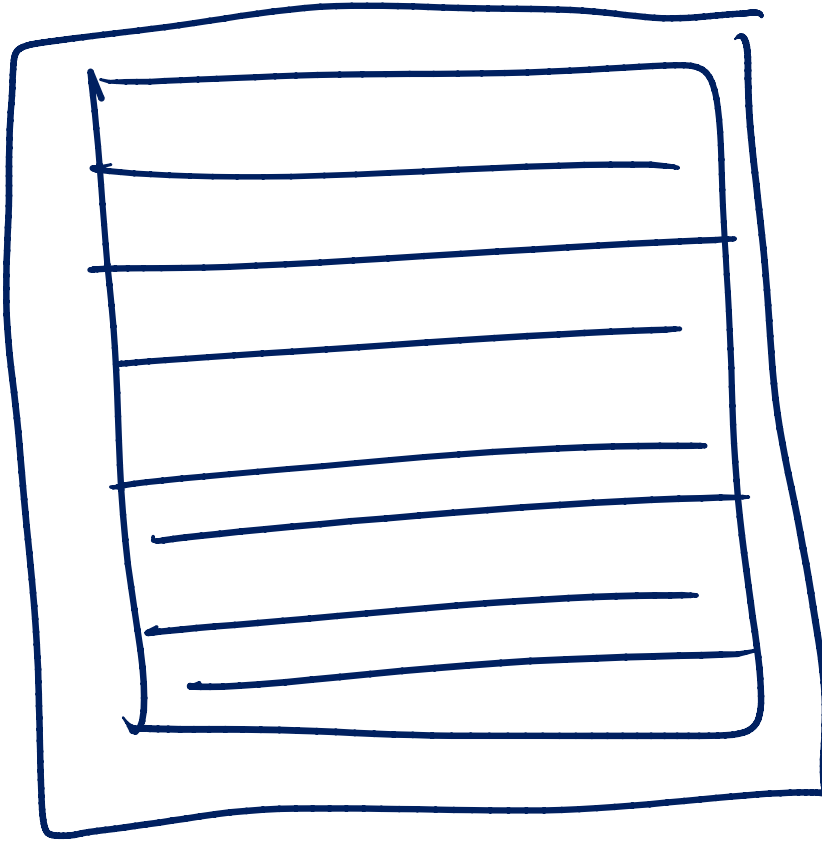
M5S45 Explaining how concurrent log backups while a data backup is running cannot clear the log. When the data backup completes, deferred log clearing takes place. This gives the impression that a data backup clears the log.



M5S51 Explaining the circular log demo. We discussed log space reservation that happens to ensure the active transactions can roll back without having to grow the log – this accounts for extra log growths. Space reserved in the log does not make a VLF become active – as there are no log records written out, just empty space reserved.

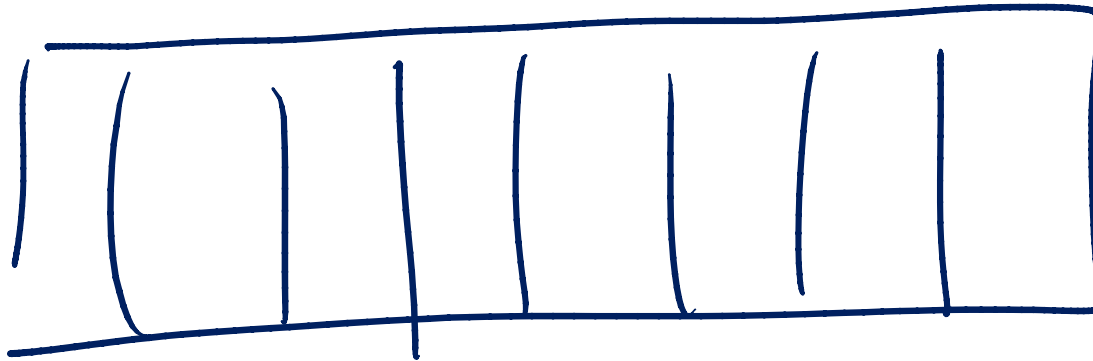
100,000

10 rows/page



M5S64 Explaining how an index rebuild in the full recovery model is efficiently logged. There will not be a log record per insert into the new index – instead there will be one log record for each complete page, with the net effect of the inserts. This is less log than for all the inserts.

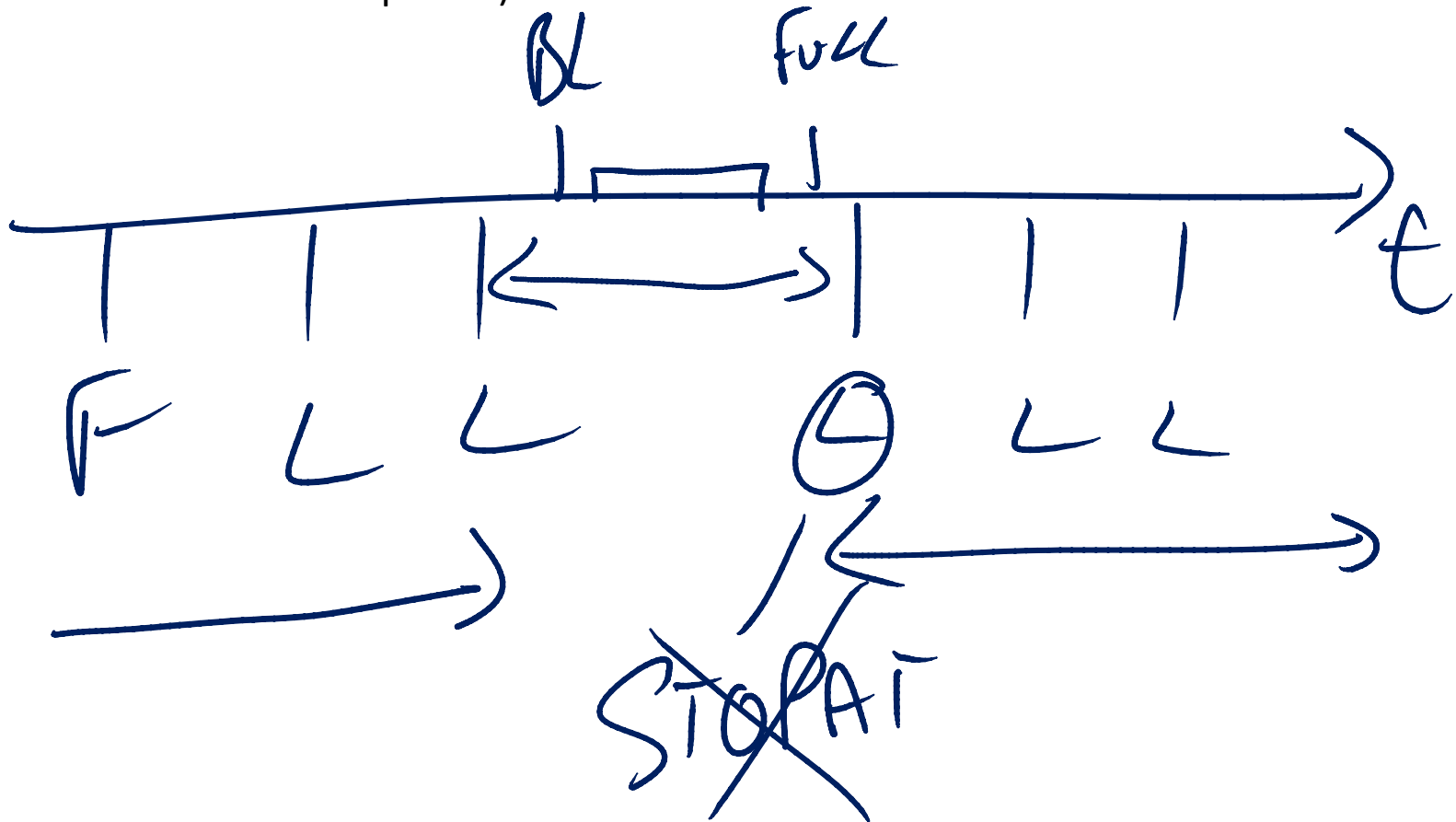
EXTENT X



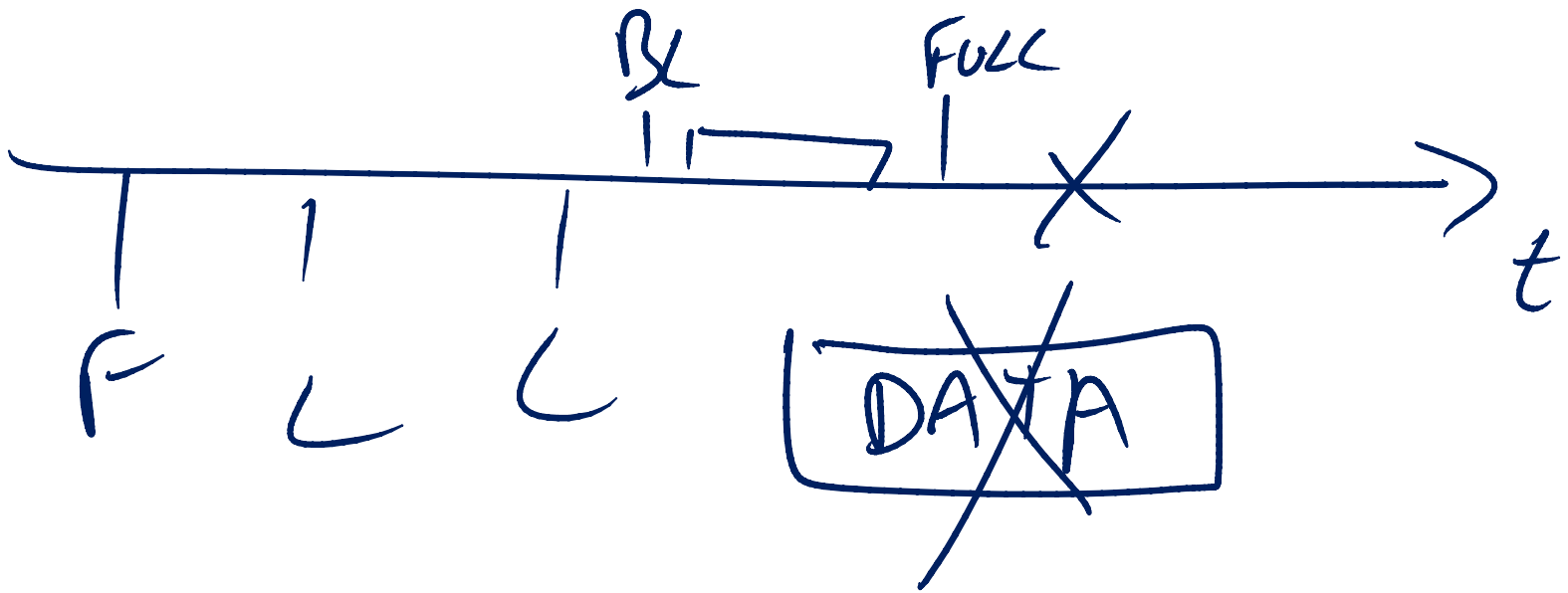
PROBE

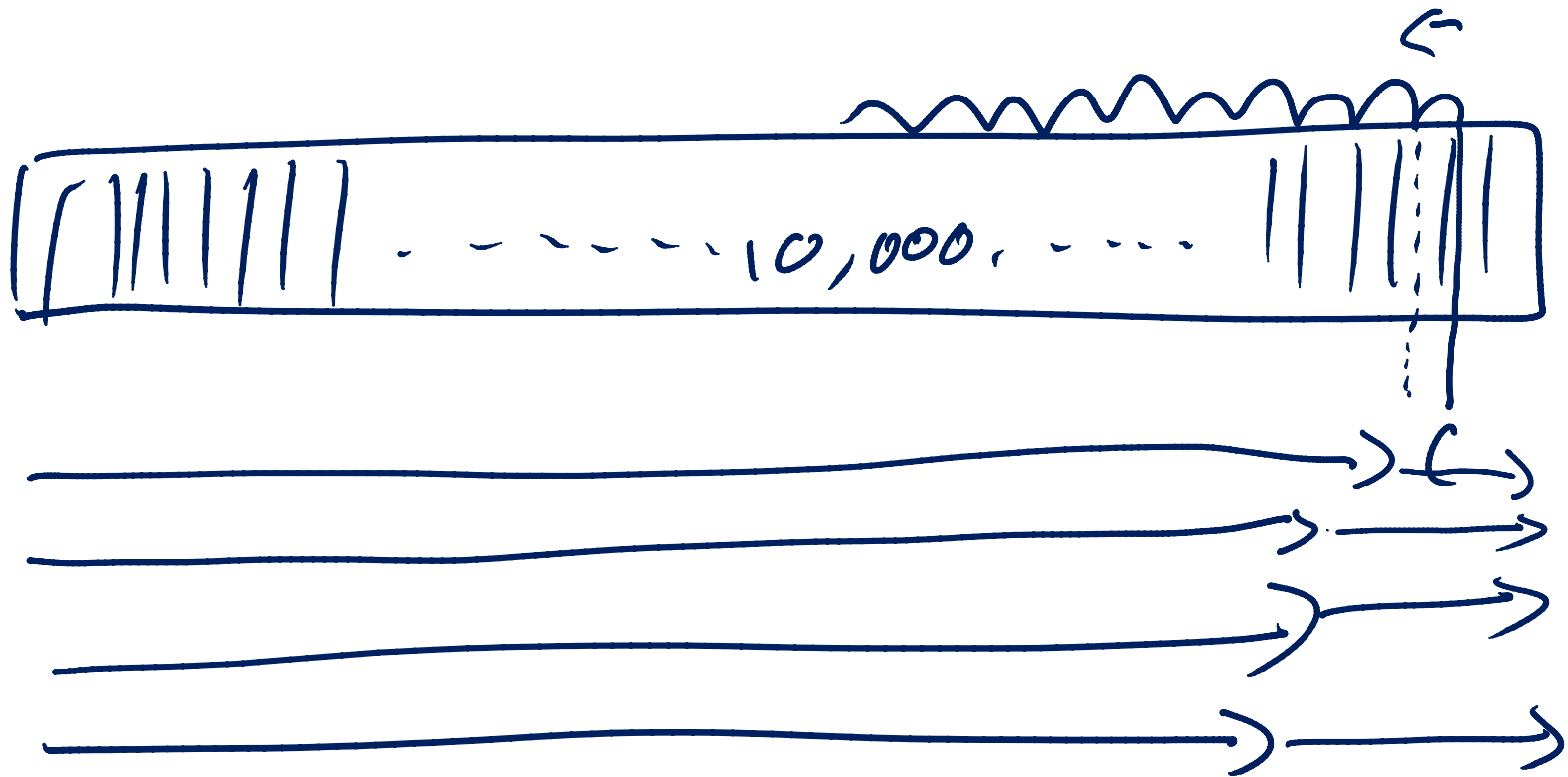
M5S65 Explaining part of the deferred-drop functionality. Before an extent can be deallocated, an X lock is acquired on it and all 8 page locks are probed (instant acquire X lock and release) to make sure nothing else has them locked.

M5S66 Explaining that if a minimally-logged operation is present in a log backup, you cannot do a point-in-time restore to any time covered by that backup (except the start or end of that time period).

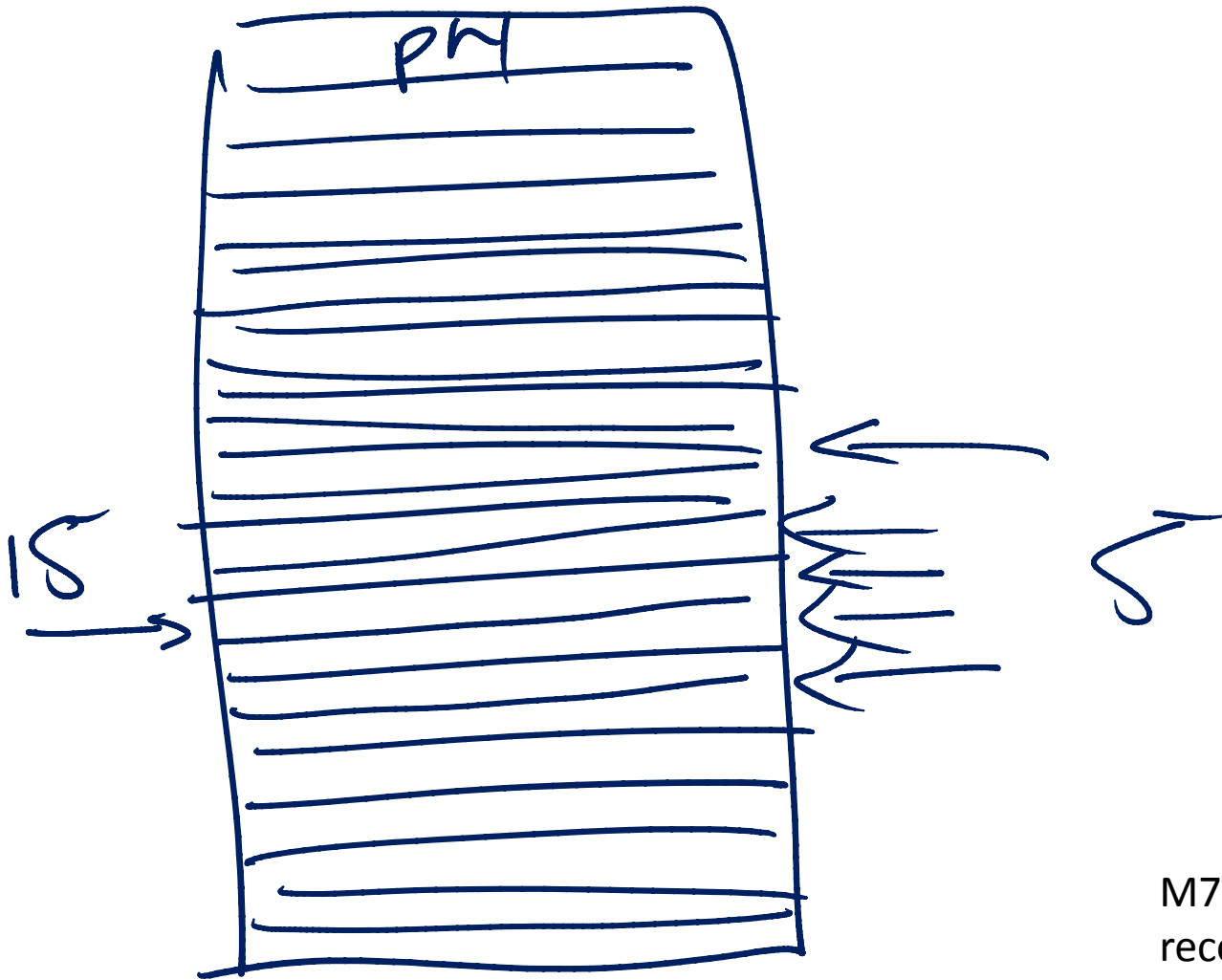


M5S66 You cannot do a tail-of-the-log backup after a crash that destroys the data files if there has been a minimally-logged operation since the last log backup. Well in 2008 R2/2012, you can, but the log backup is invalid.





M5S68 VLF fragmentation is bad because of having to search through the VLFs for a particular VLF sequence number. See the KB article link on the slide for more info.



M7S5 Explaining how a record is found on a page using binary search rather than brute force search.

COMMENTS MEMBERID

PAUL	ERIK	GLENN	TIM
------	------	-------	-----

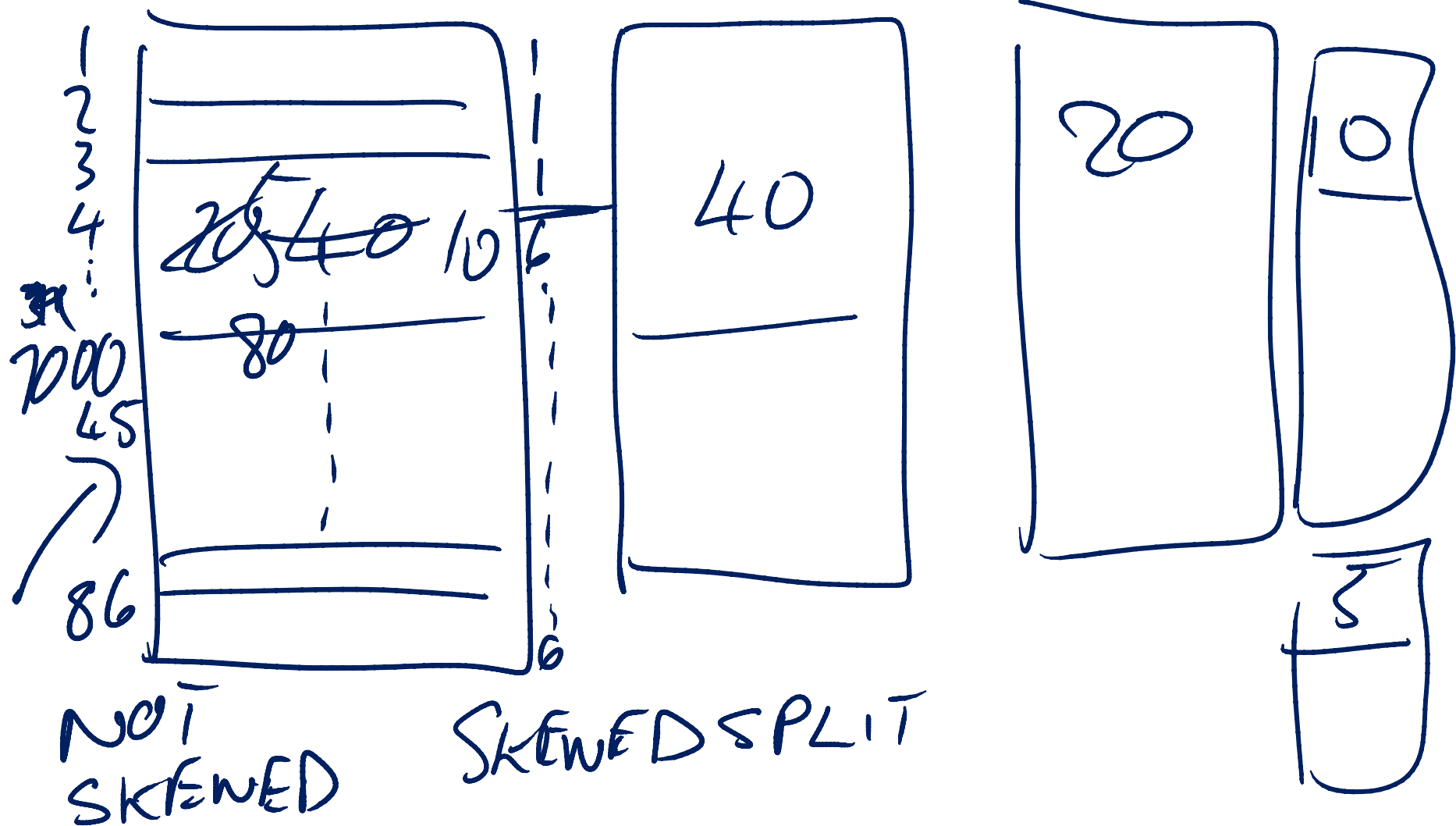
KIMBERLY

M7S19 The MySpace story. The tiny portion is for the comments on Kimberly's page because she wasn't very popular back then... 😊

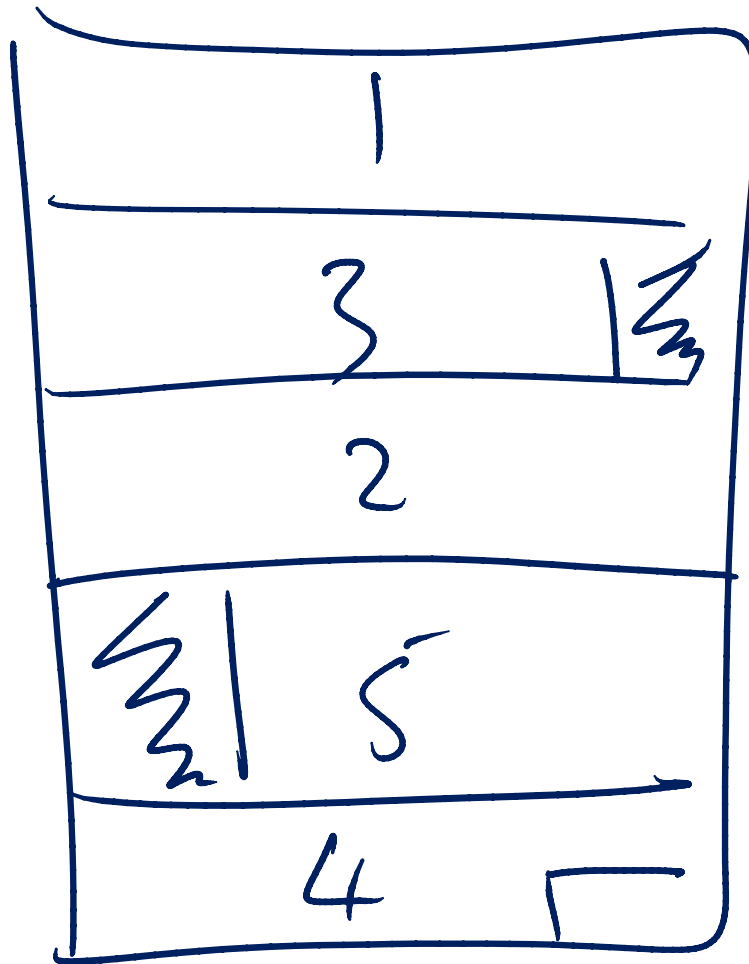


IDENTITY

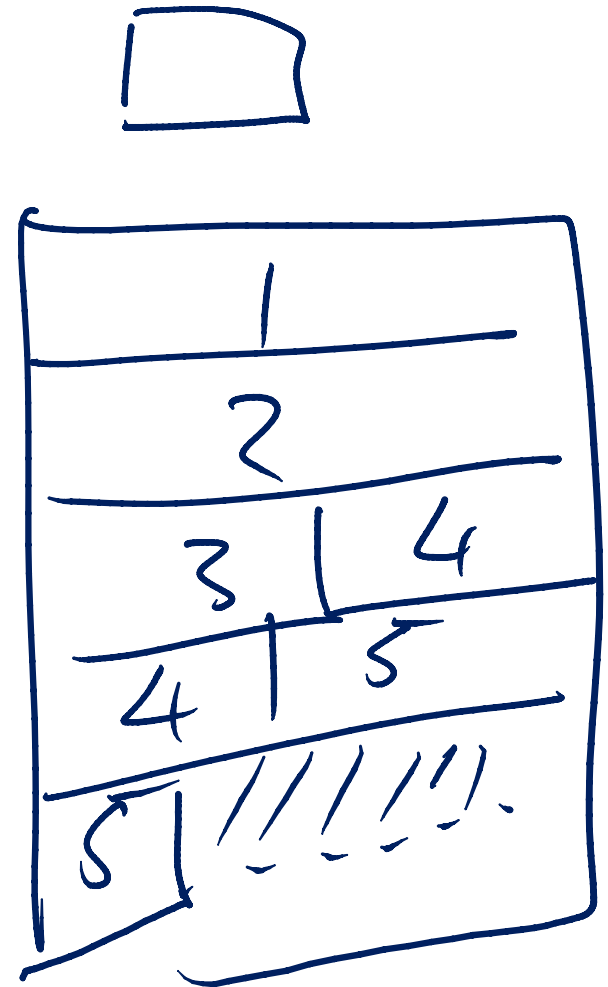
M8S18 Append-only insert pattern fills pages on right-hand side of index. New page allocations in this case are called page splits too – making all methods of tracking page splits largely useless (until 2012 with Xevents). You can look for LOP_DELETE_SPLIT log records to see splits occurring.



M7S22 Page splits aren't always smart. Sometimes it will split and there still isn't enough space, so it splits again, and again, and again....



M7S22 Access Methods will do an in-memory page compaction instead of a page split, if it knows there is enough space on the page, but the space just isn't contiguous.



BEGIN TRAN
INSERT → SYSTEM TRAN

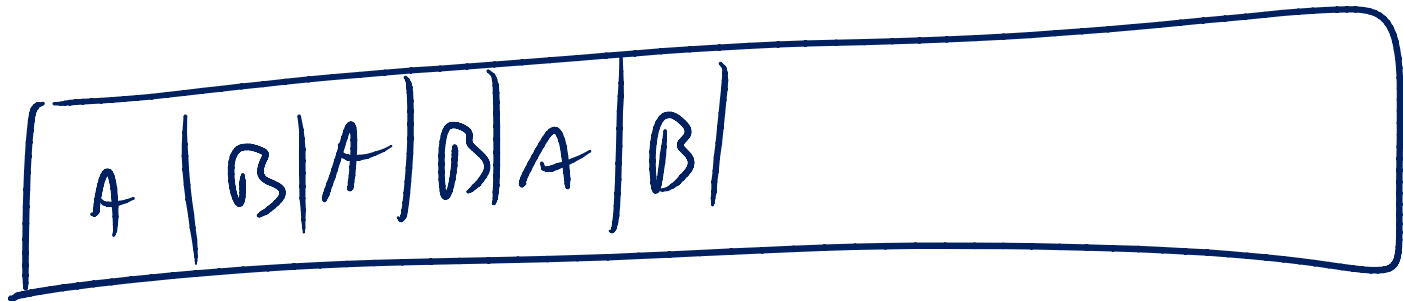
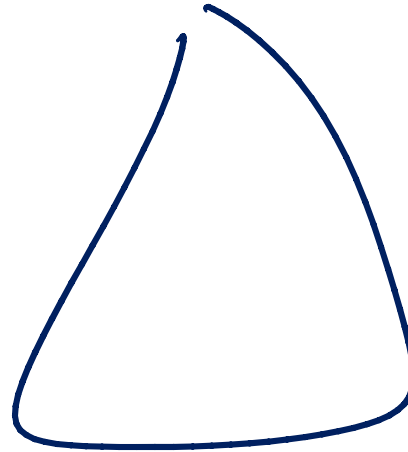
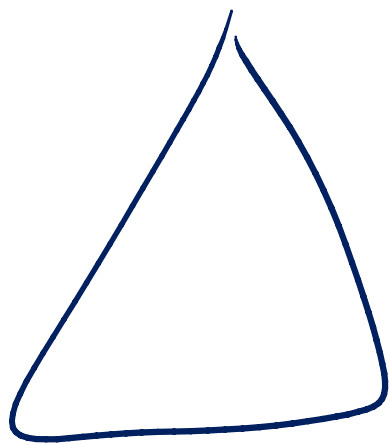
M7S22 A page split is done by a system transaction that commits and does not ever roll back.

|
SPLIT

|
COMMIT
FINISH ———
COMMIT TRAN

M7S28 The numbers from the page split logging cost demo.

	NO SPLIT	SPLIT	?
1000	1256	7580	$\times 5$
100	356	6772	$\times 18$
10	268	11172	$\times 45$



M7S38 Discussing how DOP affects contiguity. With DOP 2, 2 threads are building portions of the new index in parallel. They will all be allocating extents, leading to extent fragmentation (and hence logical fragmentation).



M7S44 Explaining staggered index maintenance with database mirroring. See <http://bit.ly/IS04W5> for more explanation