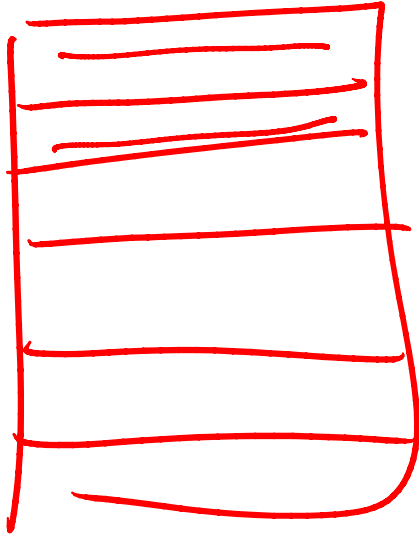


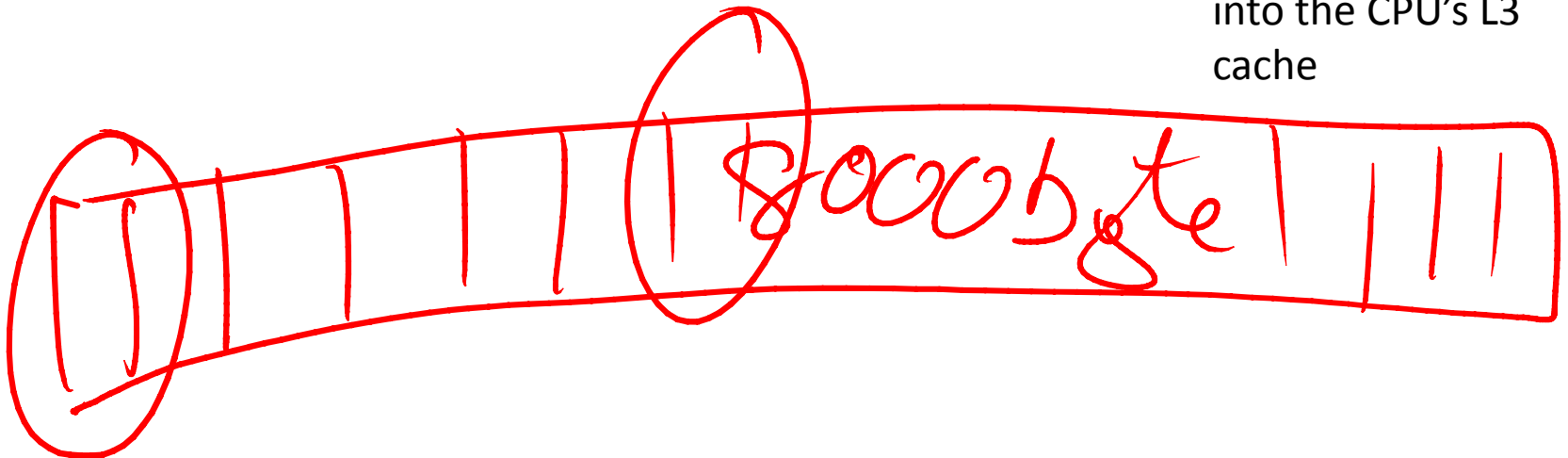
#Sqlhelp

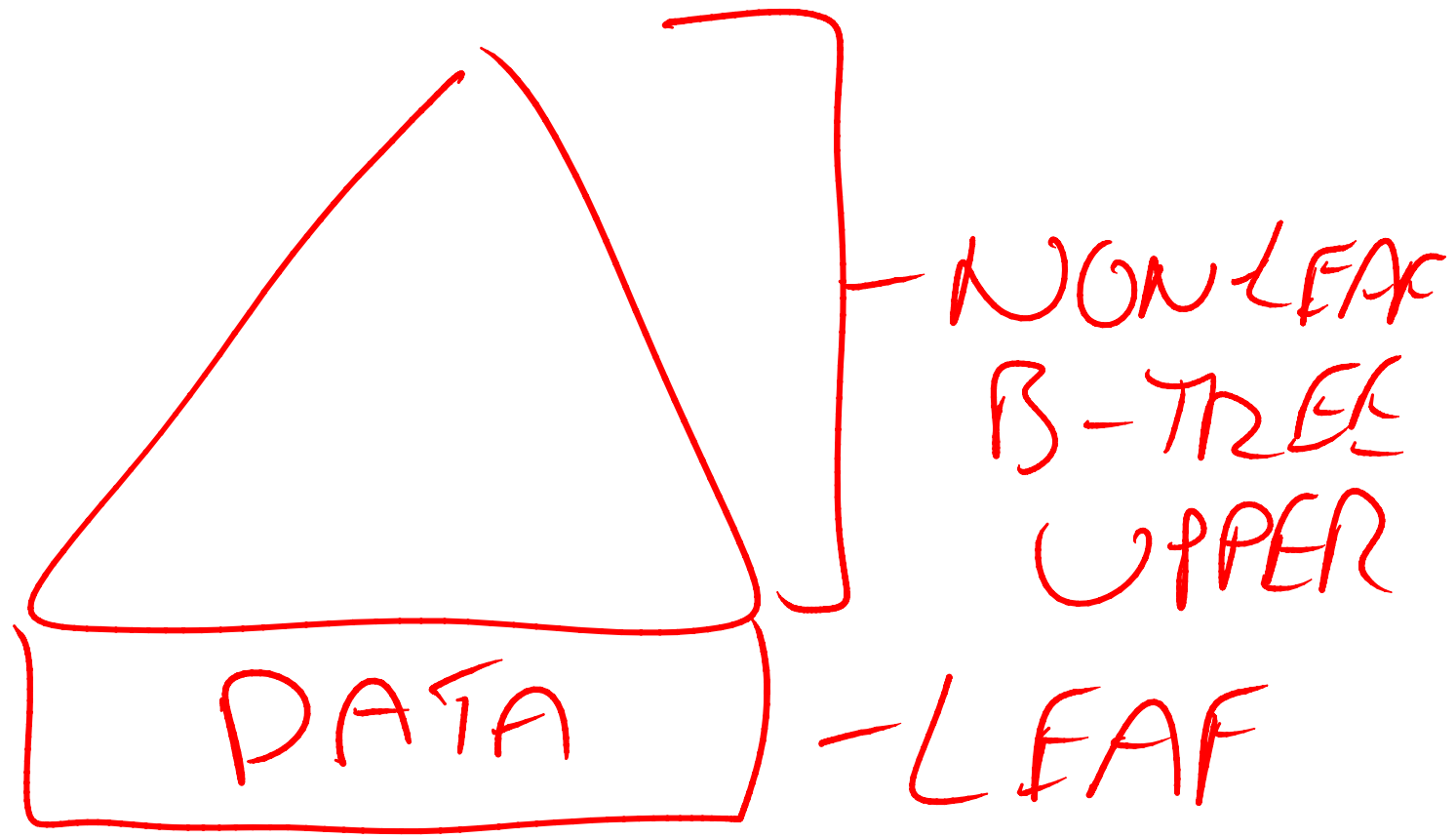
CACHE LINE INVALIDATION

CACHE LINES
64b / 128b



M1S06 Discussing
why the NULL
bitmap is a perf
optimization by
avoiding having to
read record parts
into the CPU's L3
cache





M1S8 Explaining the division of data records/pages and index records/pages in a clustered index.

$c_1, c_2, \dots, c_{10}$



HL: PHYSICAL
RID

CL: LOGICAL
RID

Set c_3, c_4
Where $c_2 = X$

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.

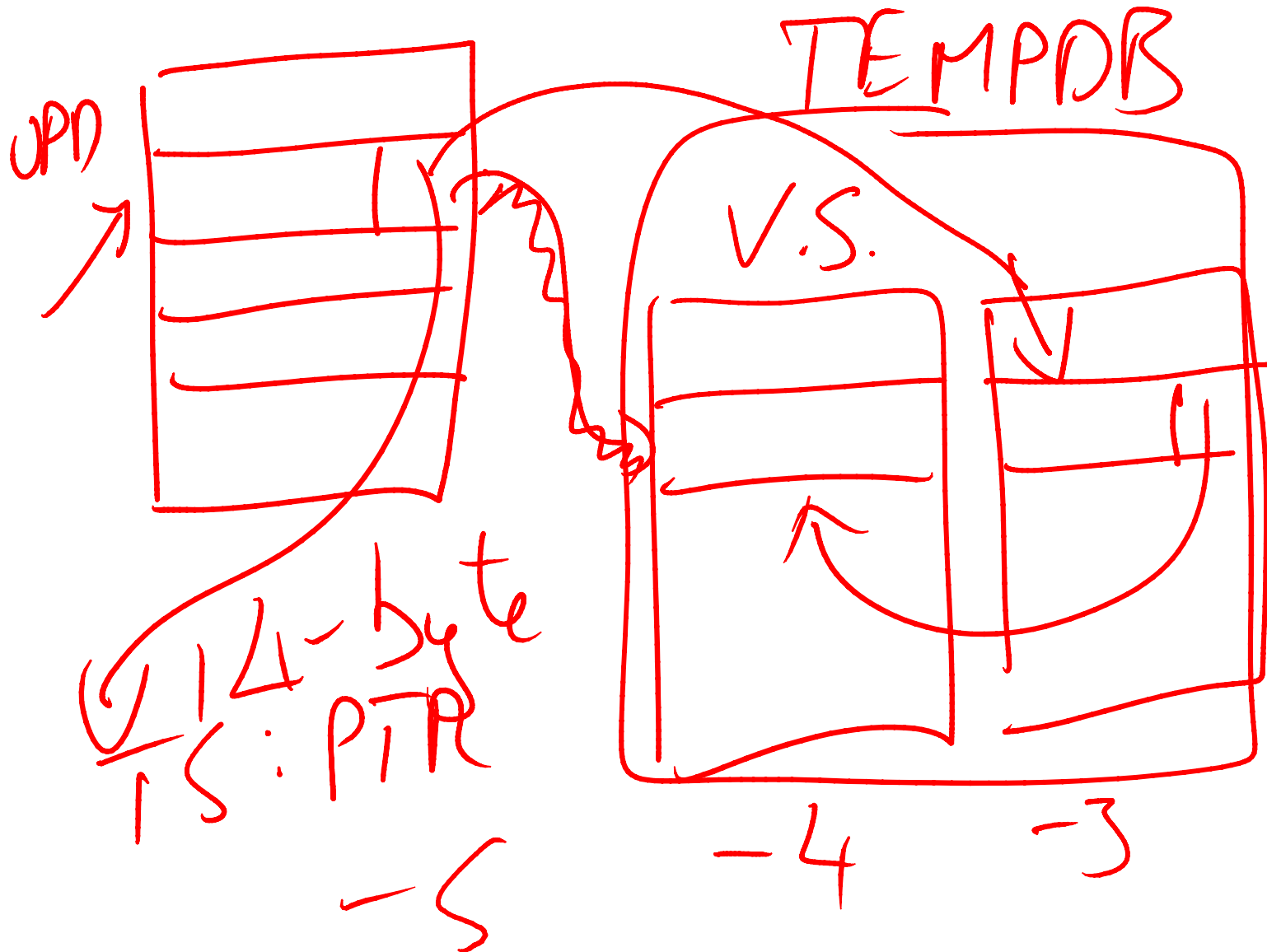
100	1900 byte
-----	-----------

[illegible]

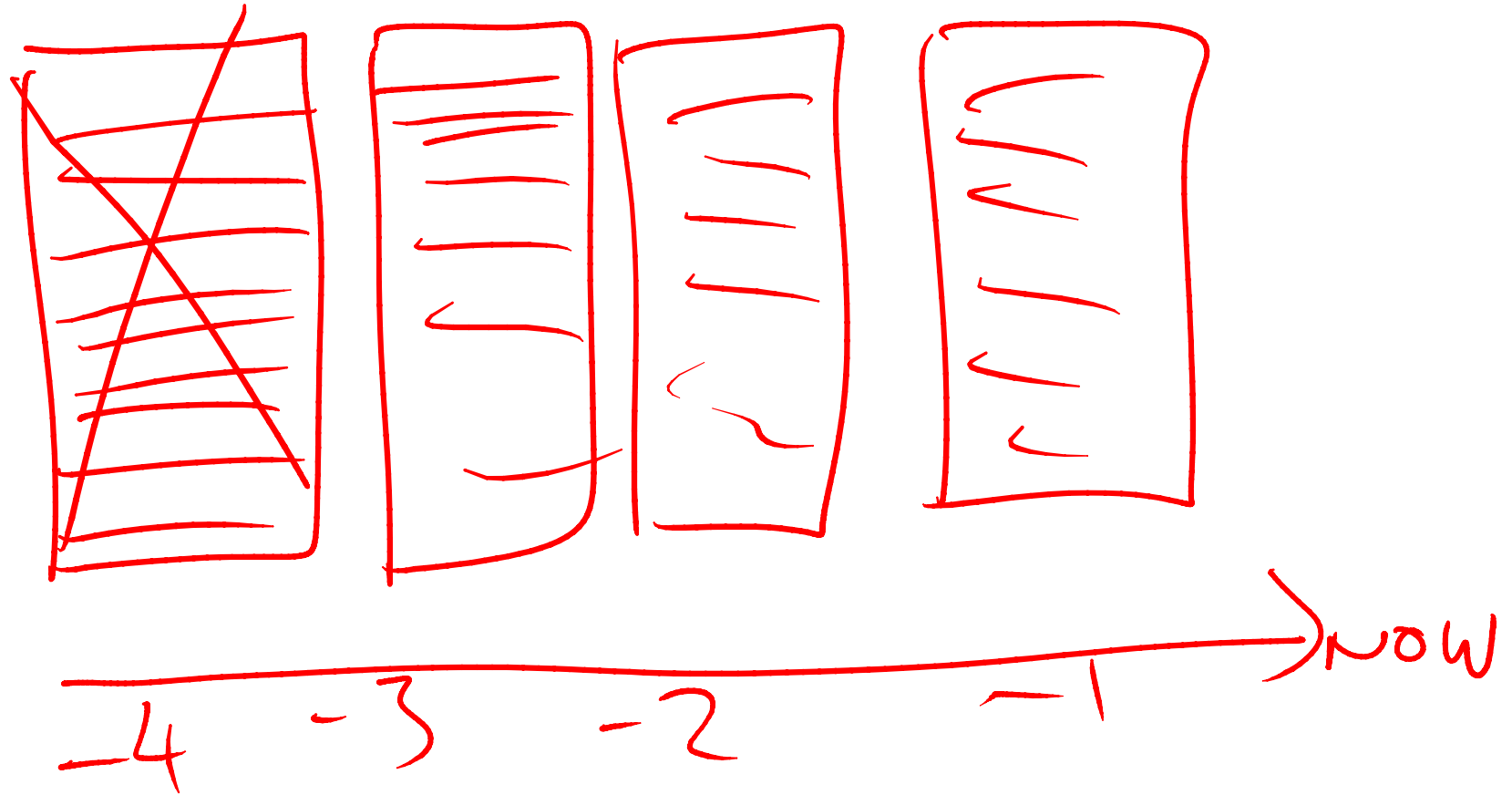
M1S12 Having a large LOB value in row that's not used much makes for large rows and low data density. Choose how to store LOB data wisely.

A hand-drawn sketch of a spiral-bound notebook. The notebook is oriented vertically. The top page is labeled '128' and the bottom page is labeled '64'. The pages are represented by horizontal lines, and the spiral binding is shown on the right side. The drawing is done in a simple, sketchy style with red lines.

M1S14 & M2S35 Explaining how there's a new portion of the version store created every minute.



M1S14 & M2S35 Explaining how there's a new portion of the version store created every minute.



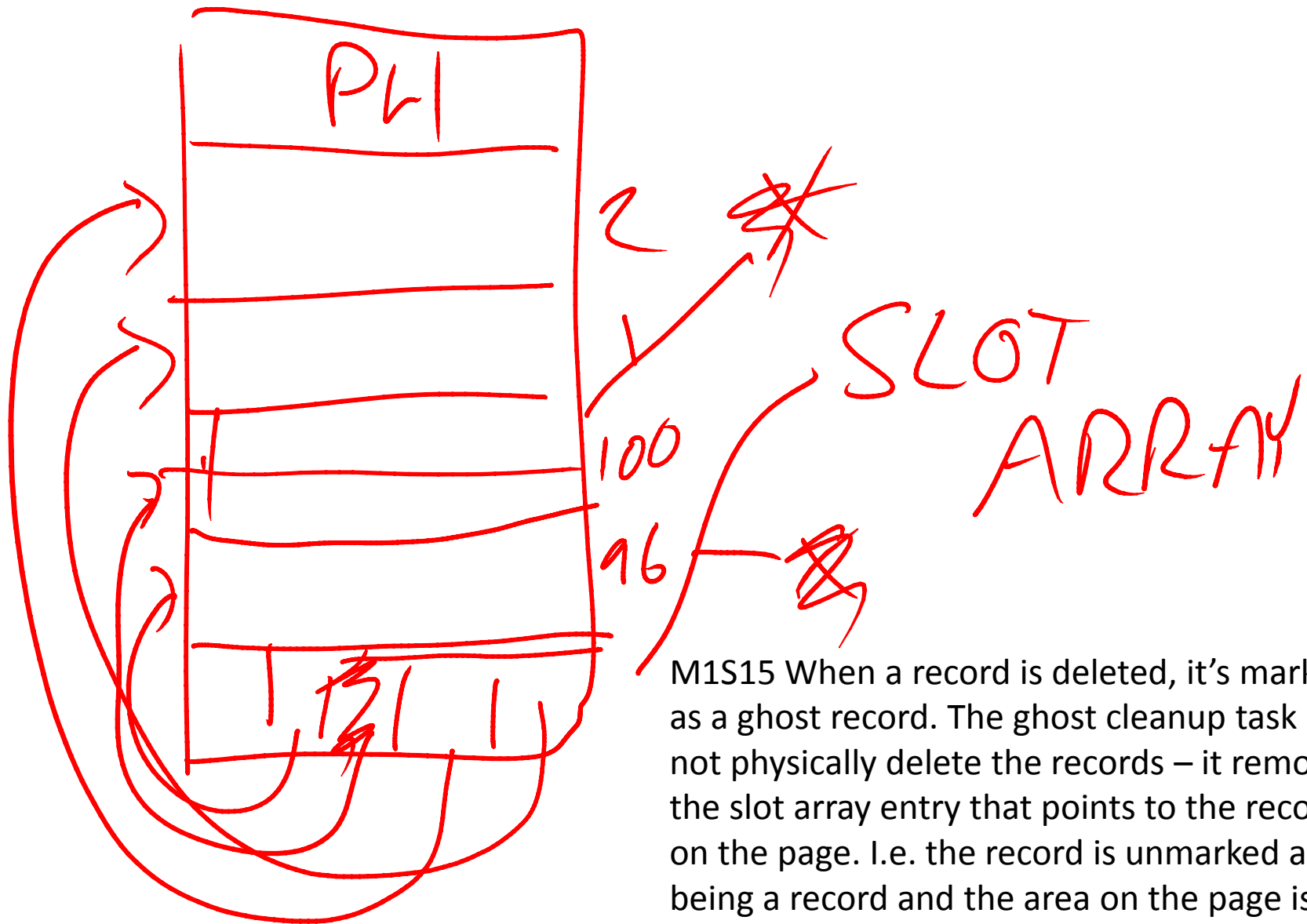
- TF 662

DBCC TRACEON(662,-1)

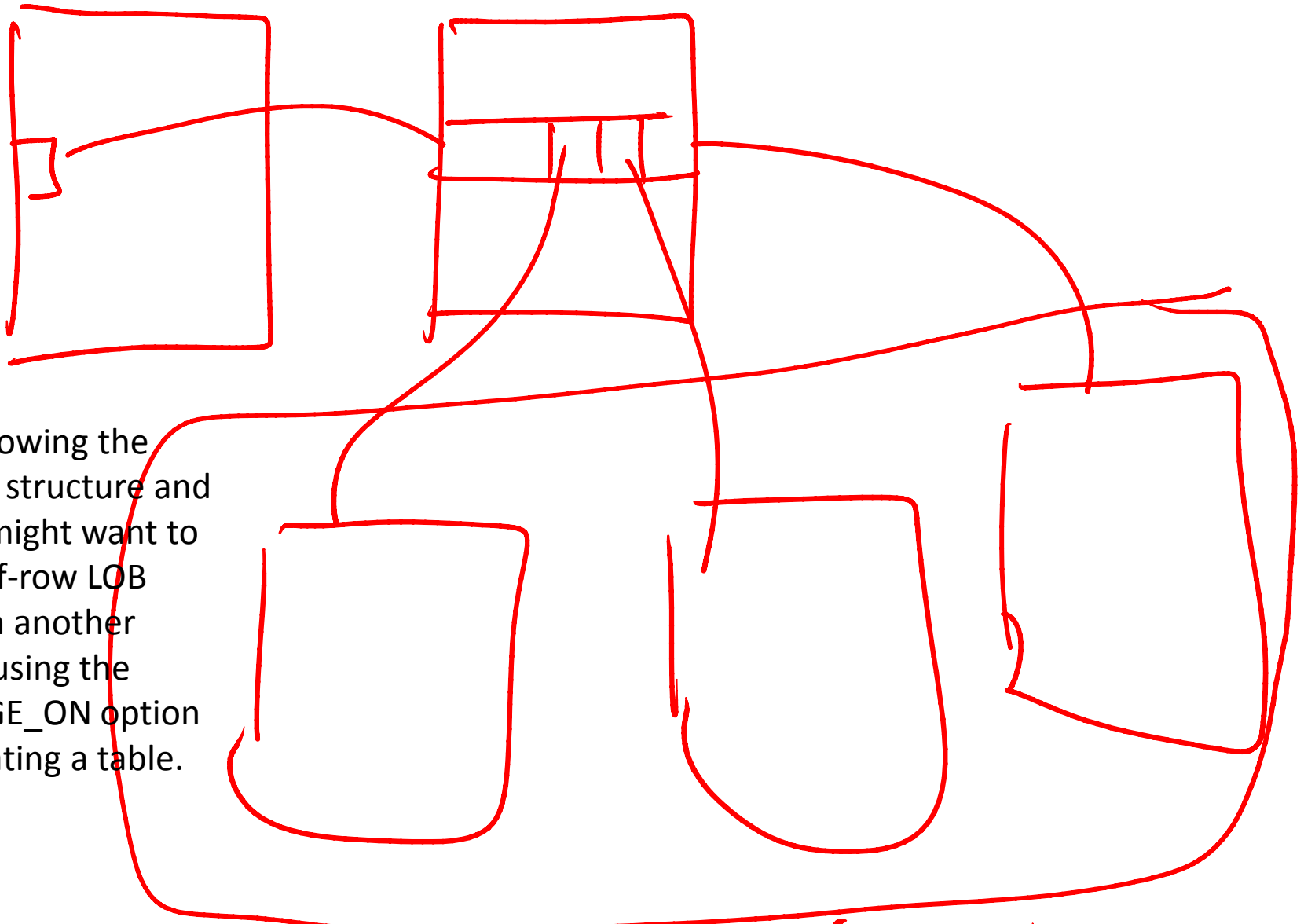
- TF 2588

DBCC HELP('FEC')

M156 Use TF662 to trace what's happening with the ghost cleanup task. Use TF2588 to allow DBCC HELP to show info for undocumented commands.

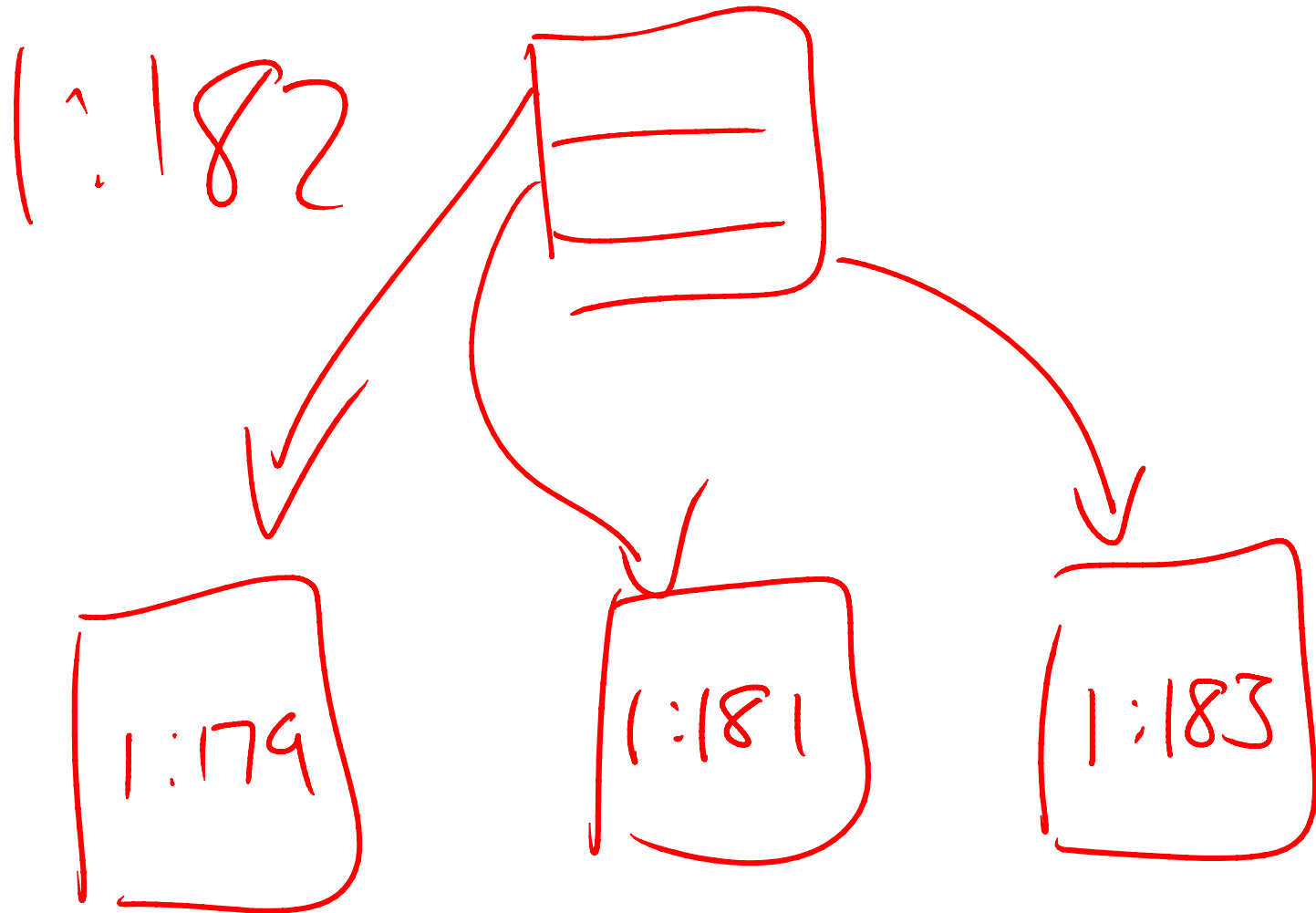


M1S15 When a record is deleted, it's marked as a ghost record. The ghost cleanup task does not physically delete the records – it removes the slot array entry that points to the record on the page. I.e. the record is unmarked as being a record and the area on the page is now free space – that just happens to have bits and bytes that used to be a valid record.

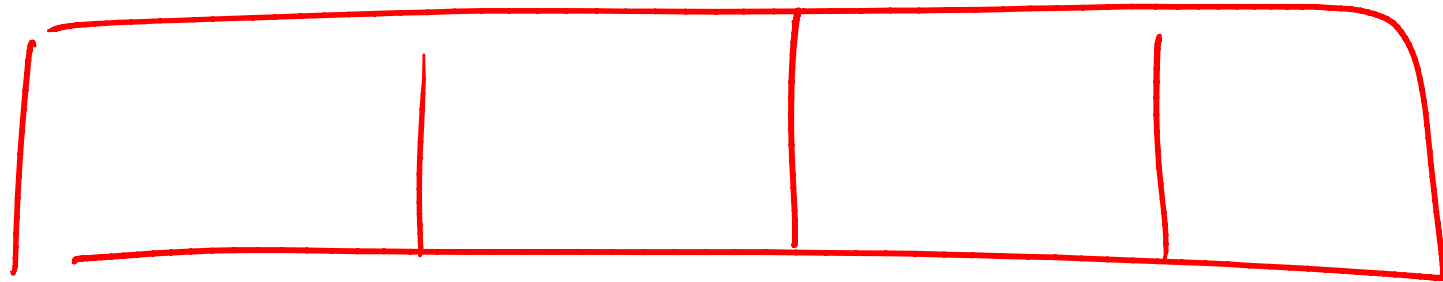


M1S11 Showing the loose tree structure and how you might want to put the off-row LOB storage on another filegroup using the TEXTIMAGE_ON option when creating a table.

TEXTIMAGE_ON



M1S22 Showing the clustered index structure generated by the page/record demo.



0 — 7 8 — 15 16 — 23

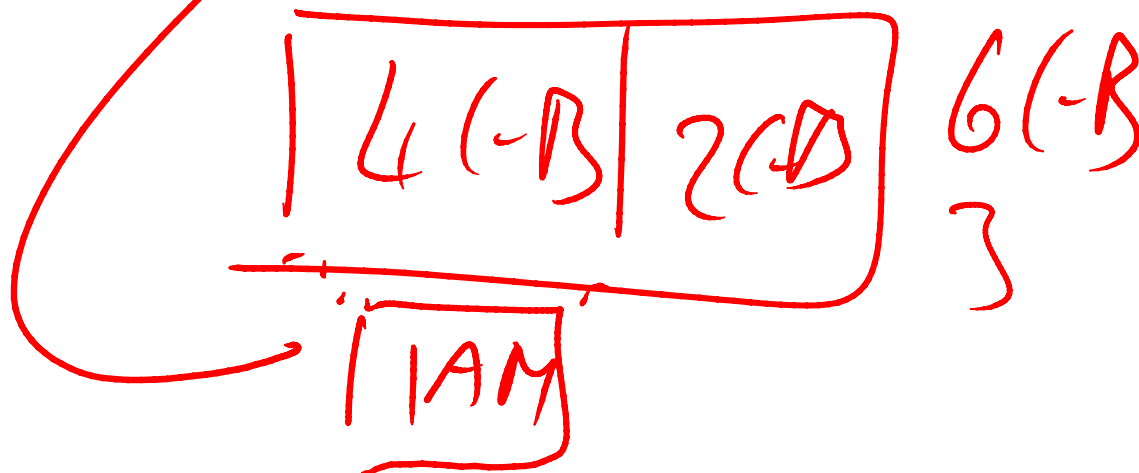


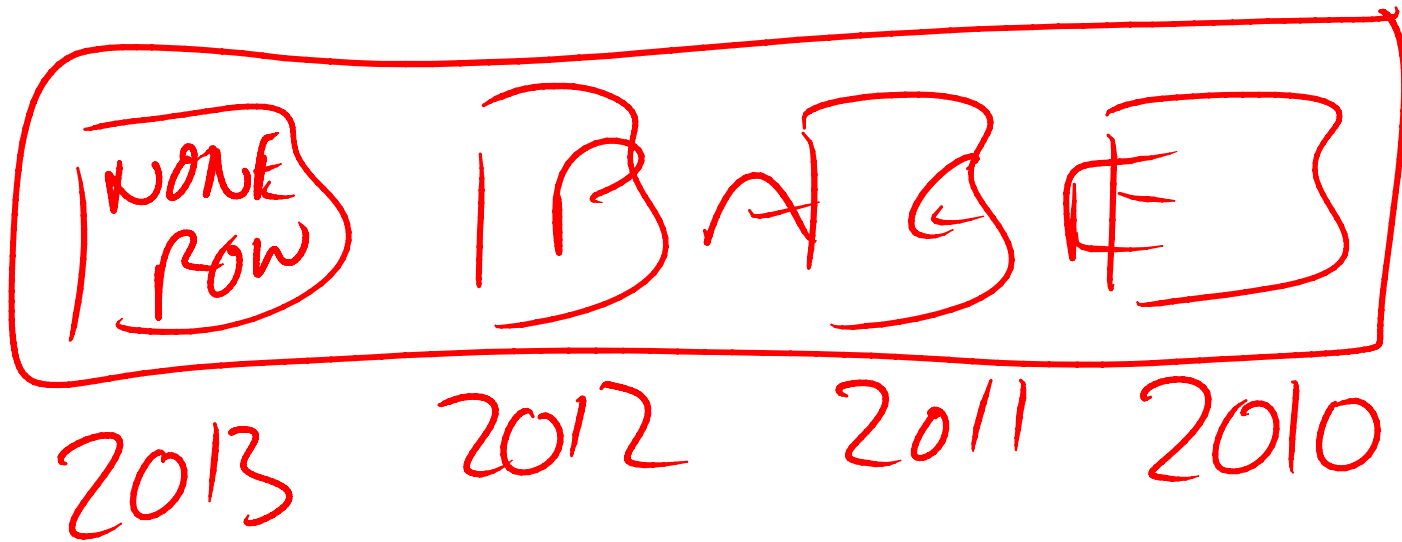
M1S24 Explaining the difference between mixed and dedicated extents. Mixed extent is one in which the 8 pages could be allocated to 8 different objects. Dedicated extent is one where all 8 pages are reserved for exclusive use of a single object.

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

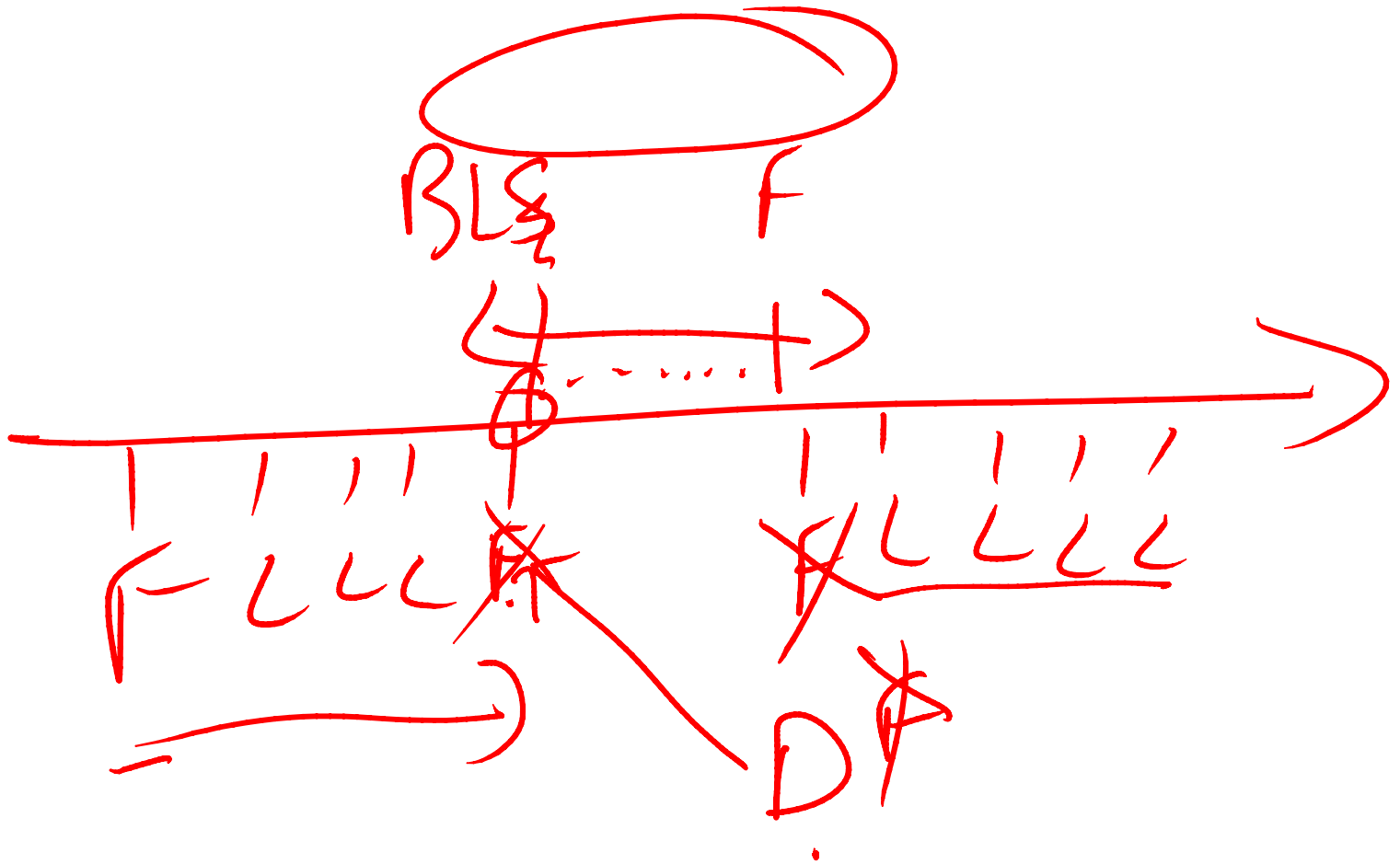


M1S36 Example of IAM chains.
Allocations from each 4GB
chunk of a data file require an
IAM page to track the
allocation. Multiple IAM pages
make an IAM chain.

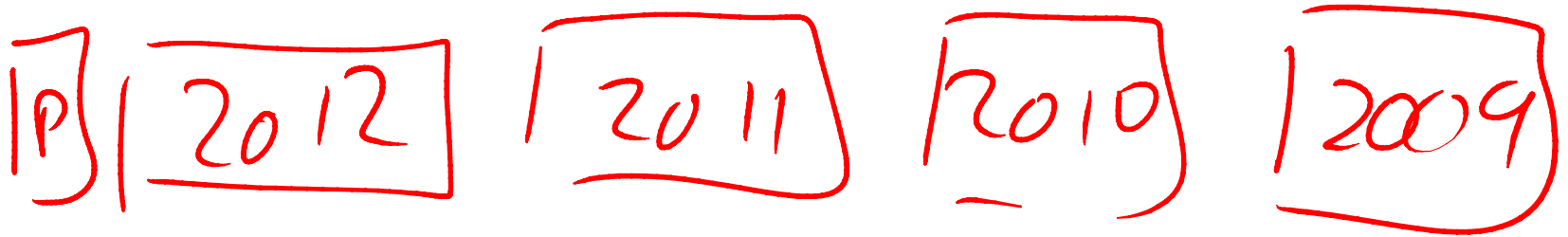
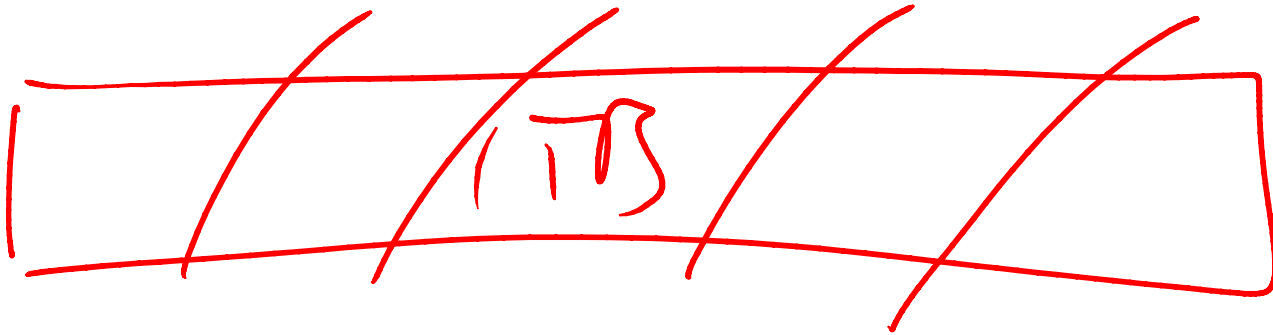




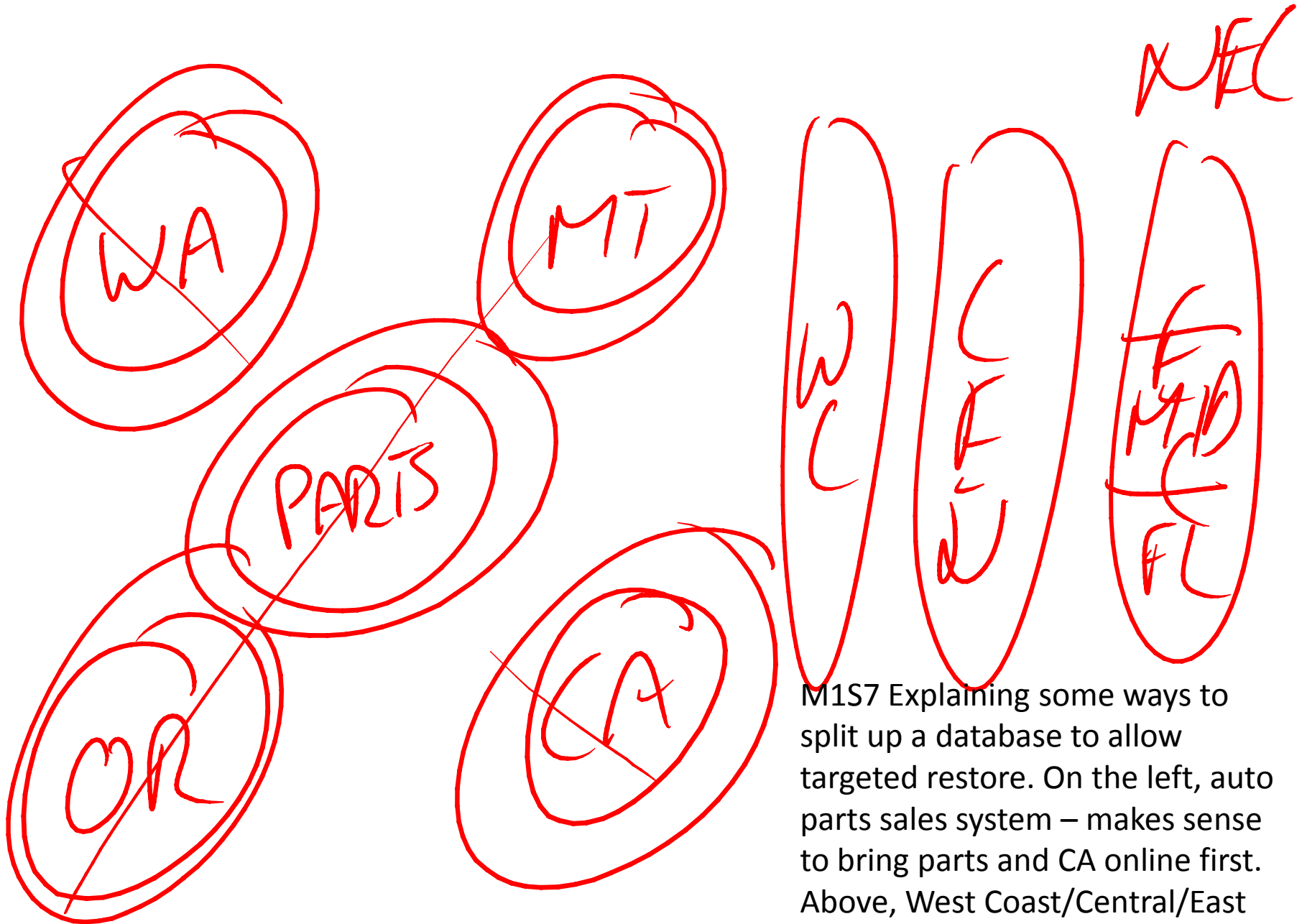
M1S50 Discussing how different partitions in a table can have different compression settings.



M2 If you switch to the Simple recovery model, you break the log backup chain. You can restart it by a full or differential backup.

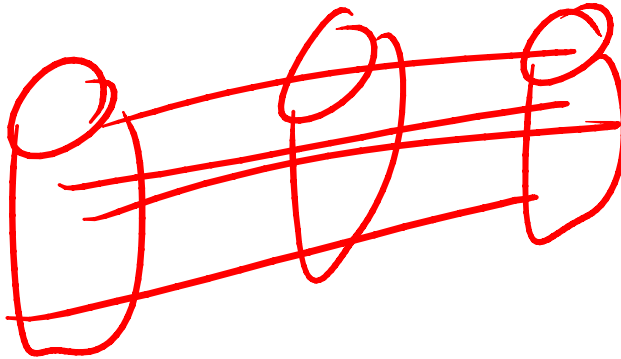


M2S7 Splitting a large database up into multiple filegroups can enable you to do small, targeted restores or even a partial restore from scratch – in the event that the database is damaged or destroyed.



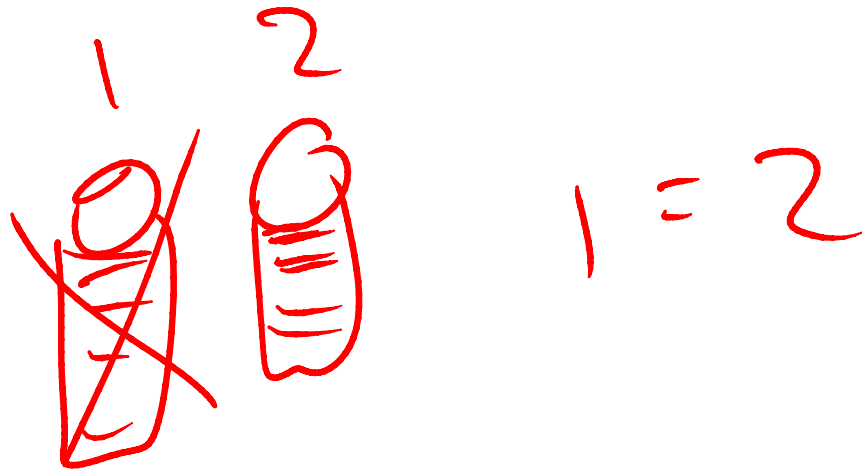
M1S7 Explaining some ways to split up a database to allow targeted restore. On the left, auto parts sales system – makes sense to bring parts and CA online first. Above, West Coast/Central/East Coast as filegroups.

R-0 STRIPING



Raid-0 – striping across
disks

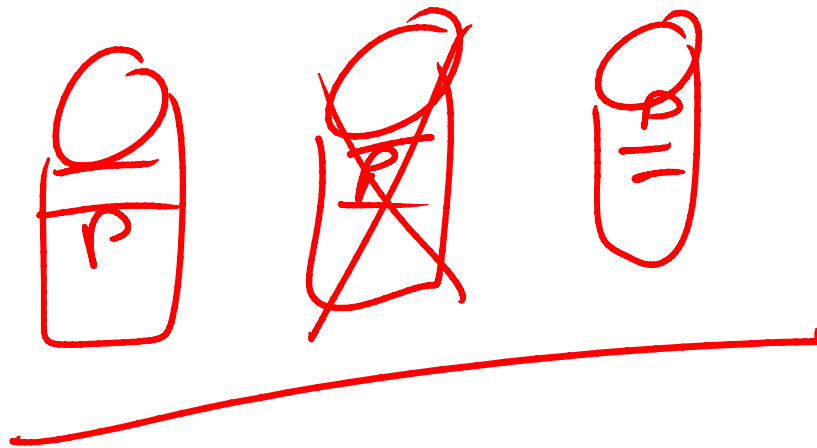
R1 MIRRORING



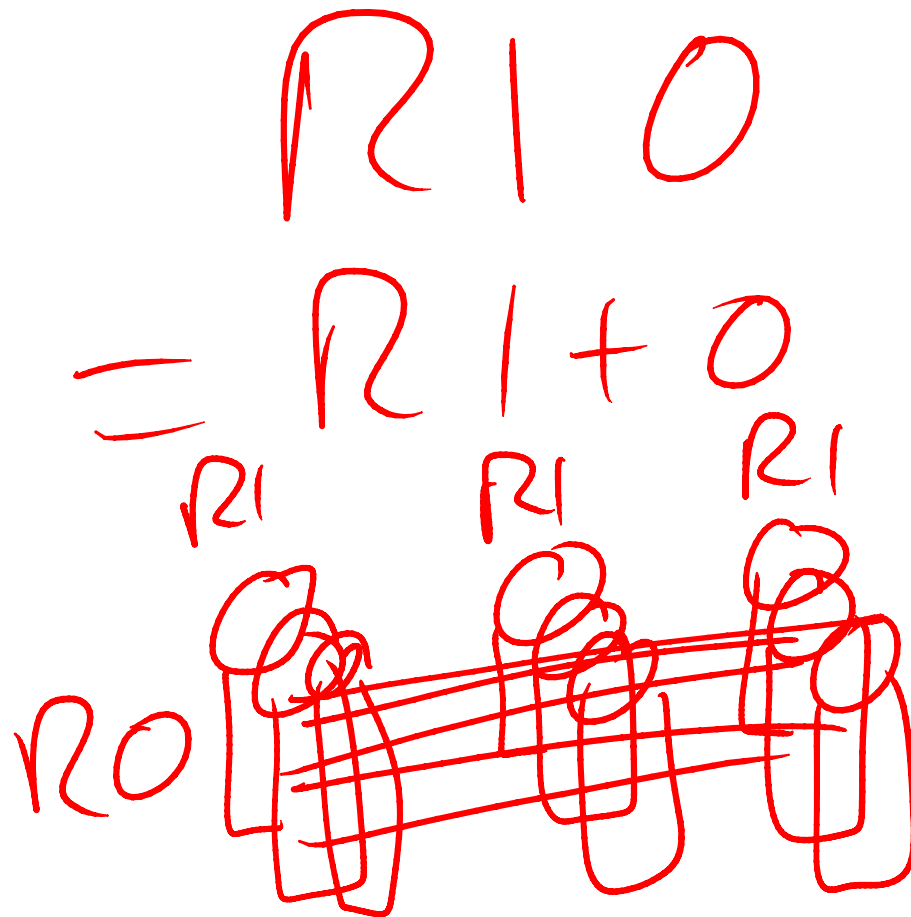
Raid-1 Disk mirroring

RS

ROTATING PARITY



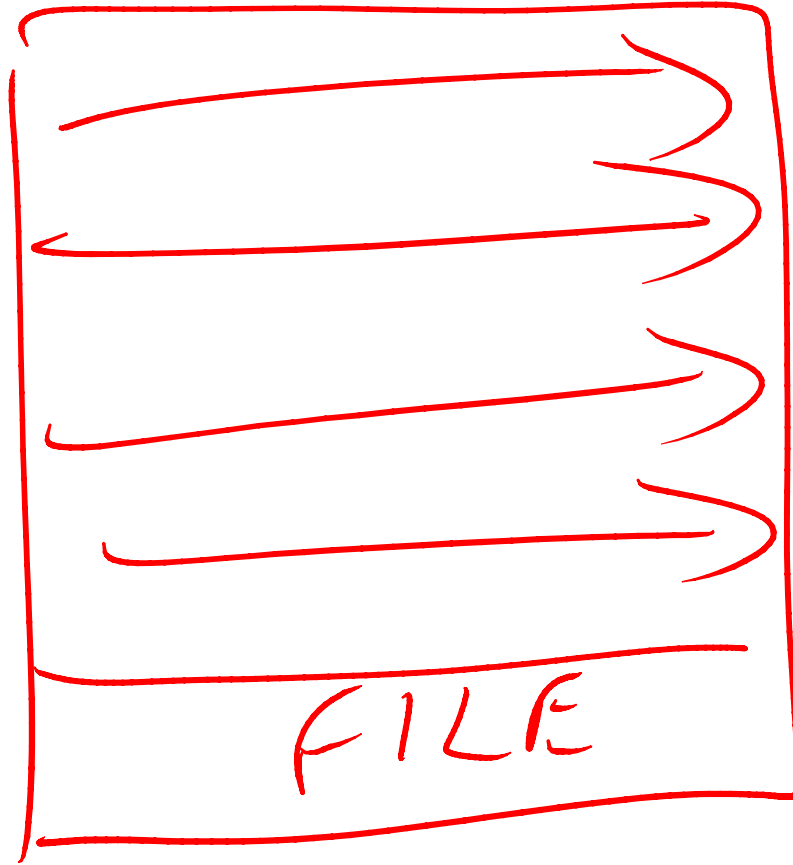
Raid-5 – rotating parity stripe



Raid-10 – striping across sets of mirrored disks

sys.dm_io_virtual_file_stats

The DMV to use to examine I/O latencies from inside SQL Server –
sys.dm_io_virtual_file_stats. See <http://www.sqlskills.com/blogs/paul/how-to-examine-io-subsystem-latencies-from-within-sql-server/>



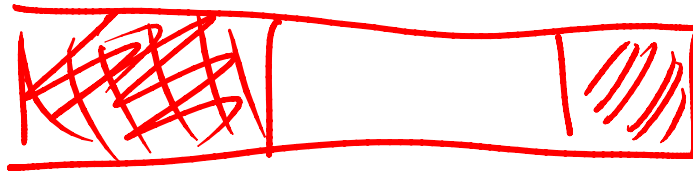
NAND

NOT-AND



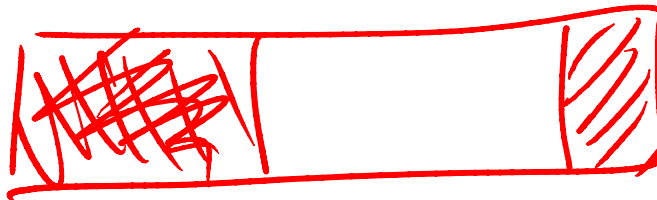
M2S17 Explaining how SSDs do wear-leveling – spreading writes across the whole surface of the disk to avoid NAND cells wearing out.

RR

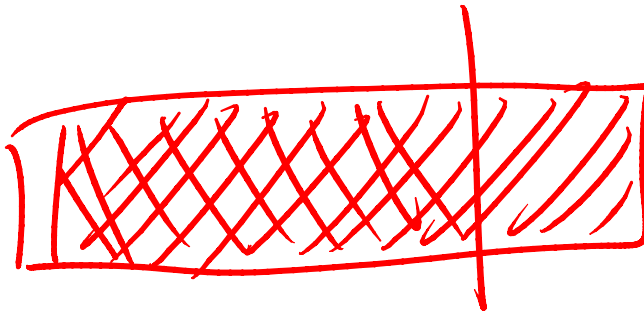


PROP FILL
WEIGHTING

~~8100~~ ←
SO (8099)

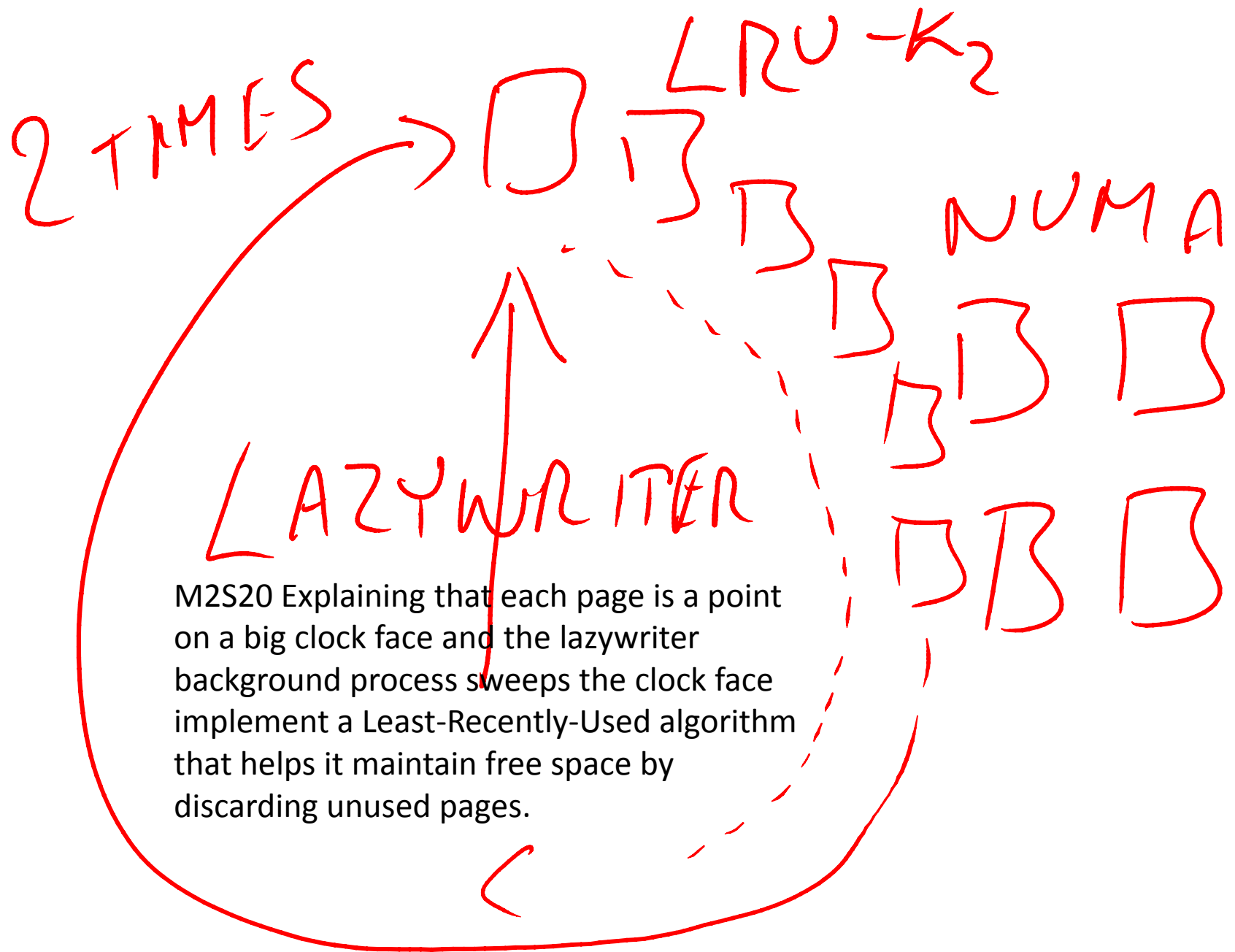


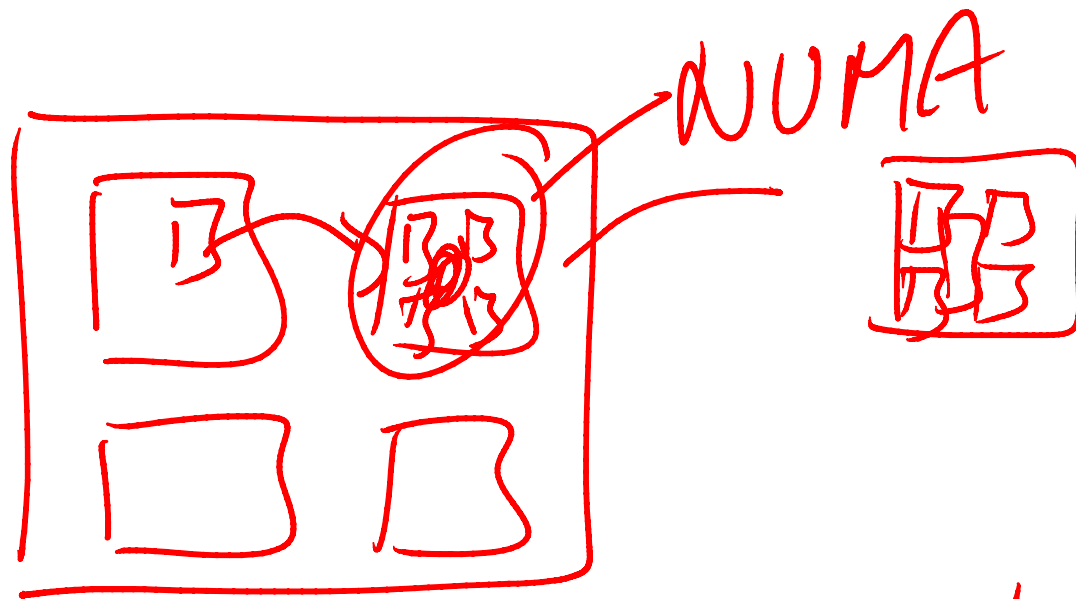
~~8100~~ ←
SO (8099)



1 ← ✓
8100

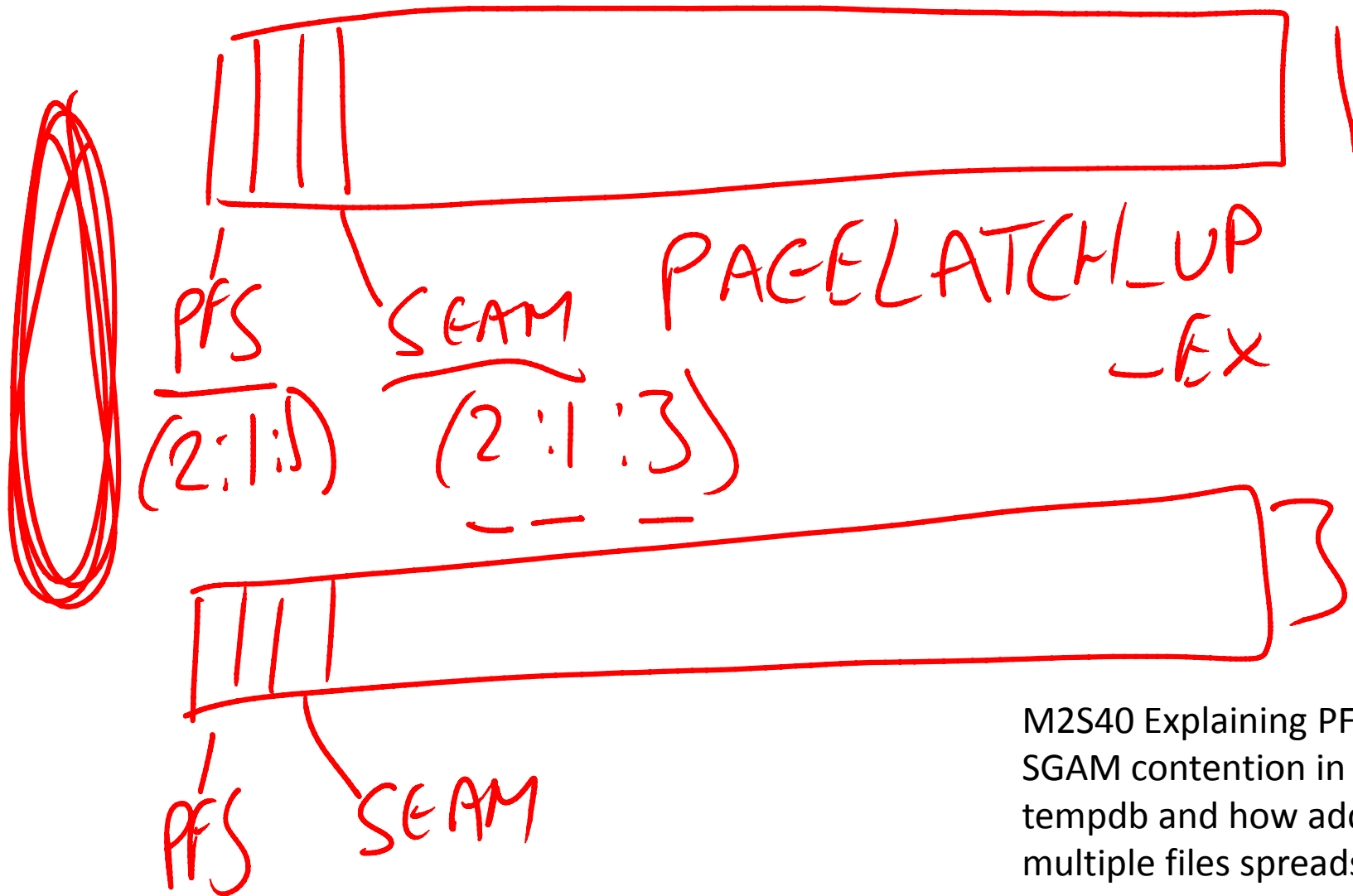
M2S19 Explaining round-robin
allocation and proportional fill





`sys.dm_exec_requests`

M2S20 Explaining how each NUMA node in a NUMA system has its own partition of the buffer pool and will have a separate lazywriter thread. You can watch the lazywriter threads using `sys.dm_exec_requests`.



M2S40 Explaining PFS and SGAM contention in tempdb and how adding multiple files spreads the contention over multiple files, thus reducing it.

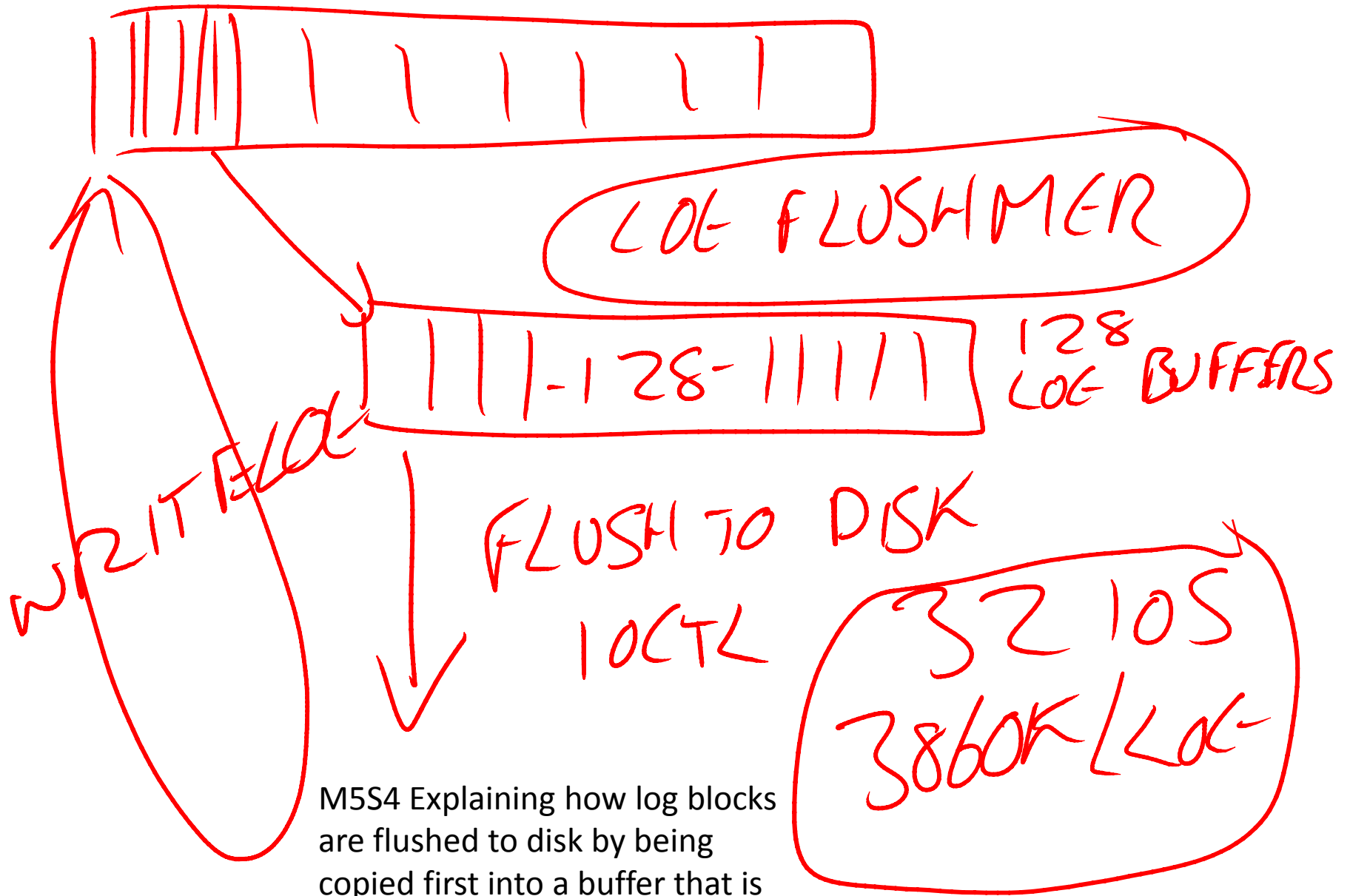
2 CPU CPUs

12-CORES PHYSICAL CORES

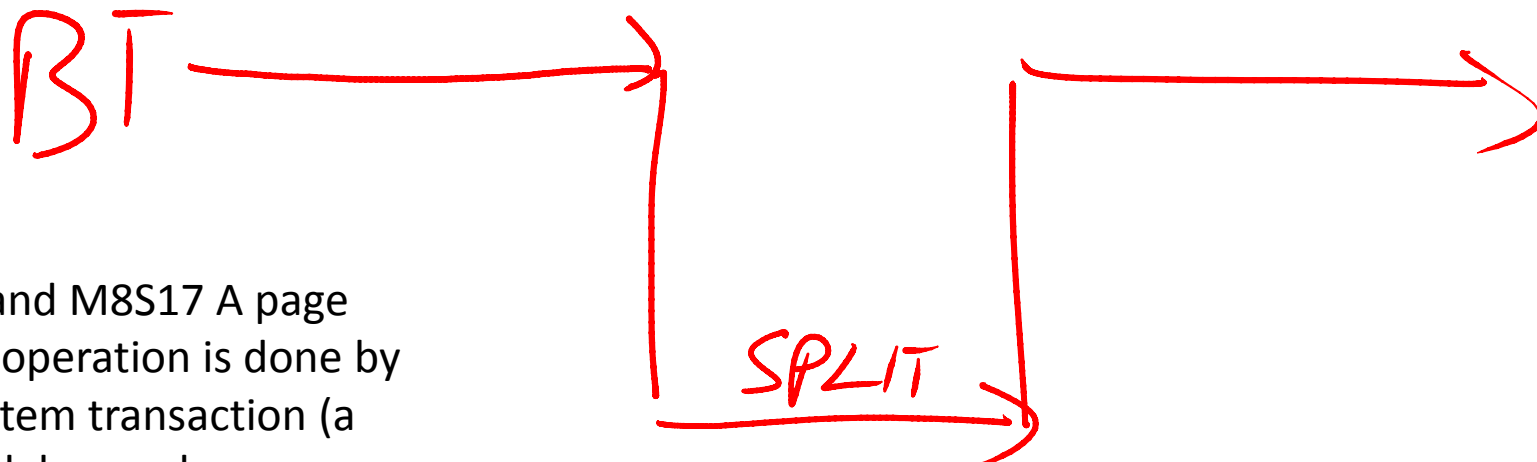
Explaining how a CPU contains physical cores. A server with two CPUs, each with 12 cores, will have 24 physical cores. If hyperthreading is enabled, that becomes 48 logical cores.

24-CORES

WIP → 48-CORES LOGICAL CORES



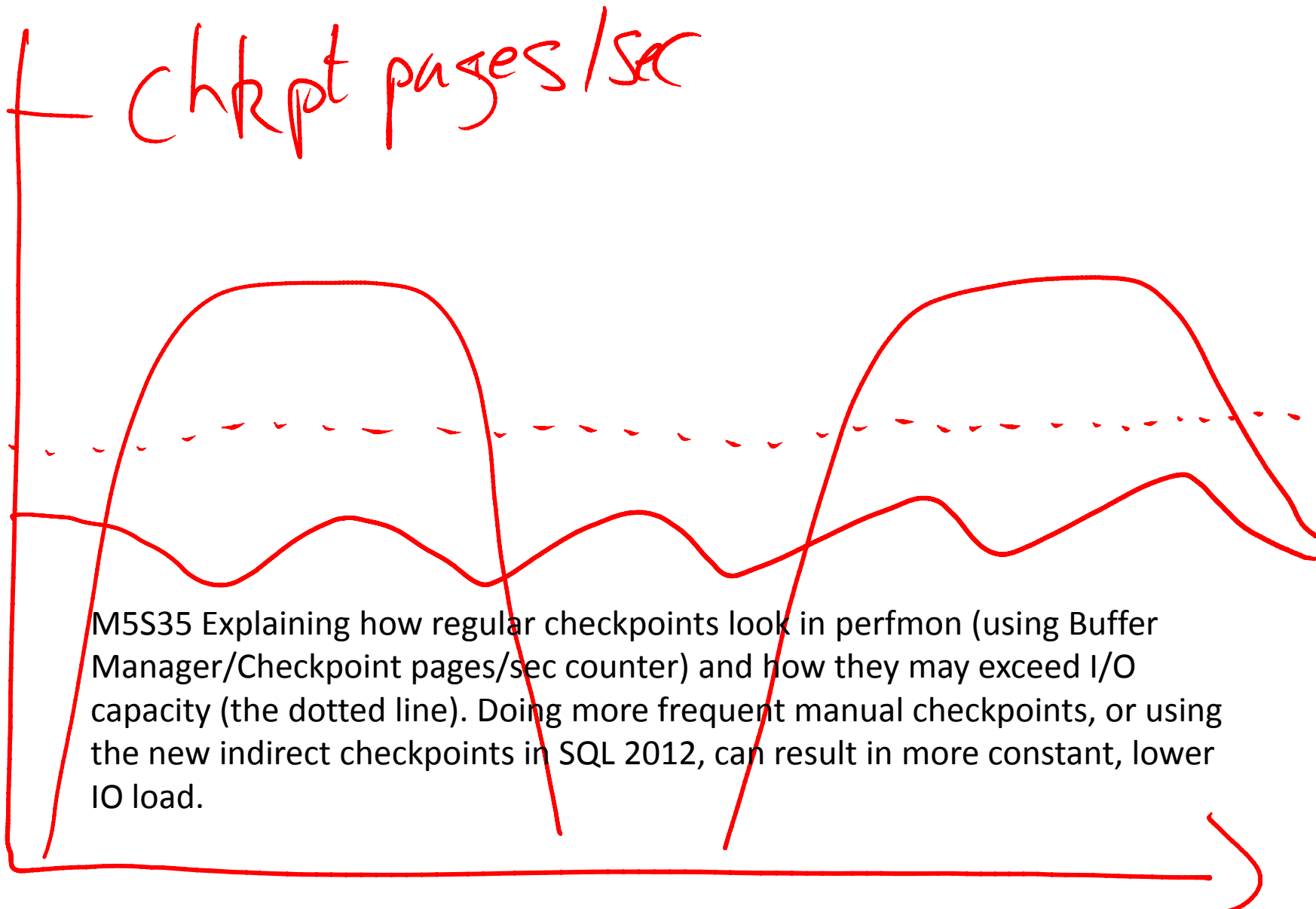
M5S4 Explaining how log blocks are flushed to disk by being copied first into a buffer that is then async written to disk.



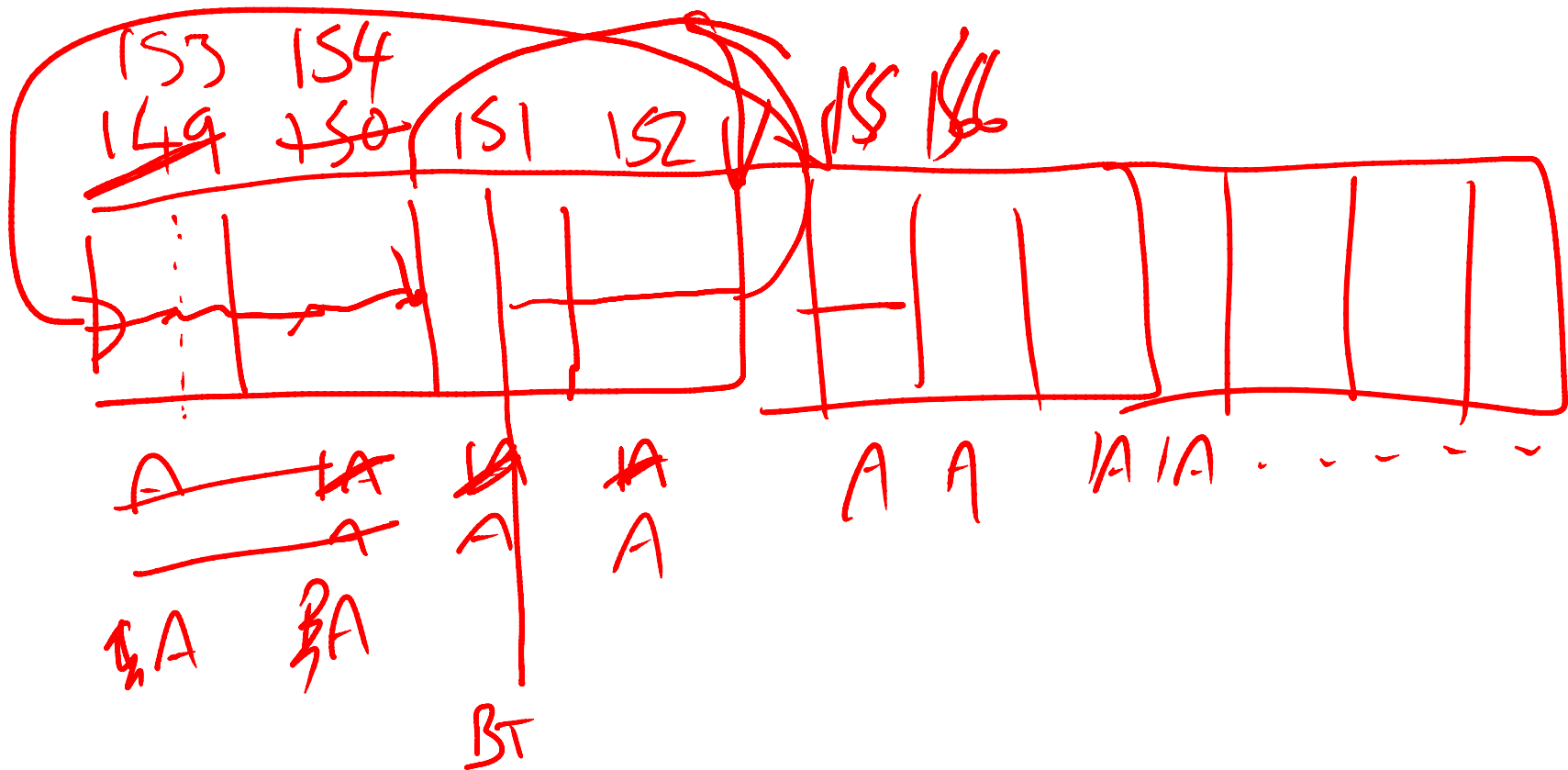
M5 and M8S17 A page split operation is done by a system transaction (a standalone sub-transaction from your transaction). It is committed automatically and will not roll back if your transaction rolls back.

BT
↑
SYSTEM
XACT

chkpt pages/sec



M5S35 Explaining how regular checkpoints look in perfmon (using Buffer Manager/Checkpoint pages/sec counter) and how they may exceed I/O capacity (the dotted line). Doing more frequent manual checkpoints, or using the new indirect checkpoints in SQL 2012, can result in more constant, lower IO load.



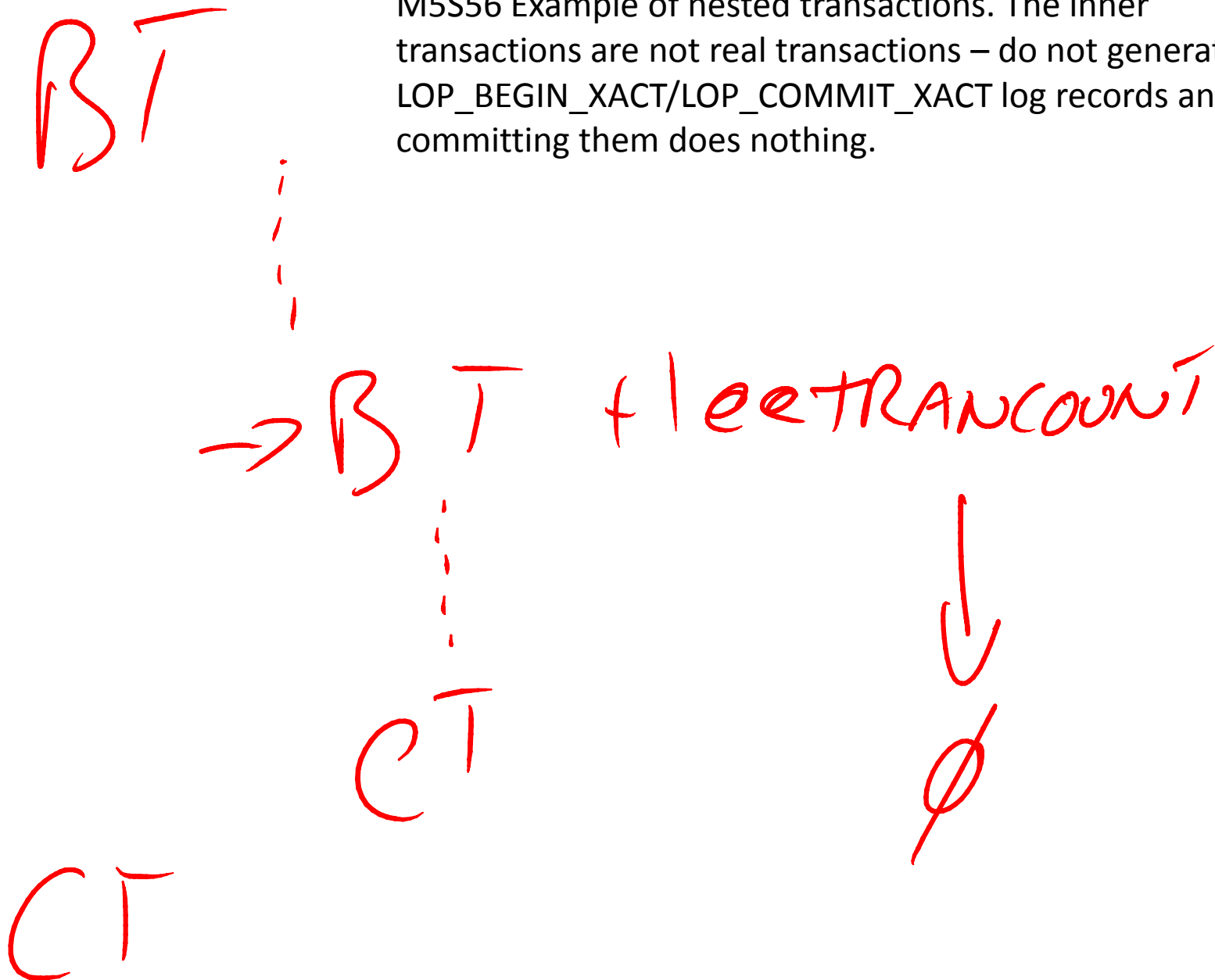
M5S28 Explaining the circular log demo. We discussed log space reservation that happens to ensure the active transactions can roll back without having to grow the log – this accounts for extra log growths. Space reserved in the log does not make a VLF become active – as there are no log records written out, just empty space reserved.

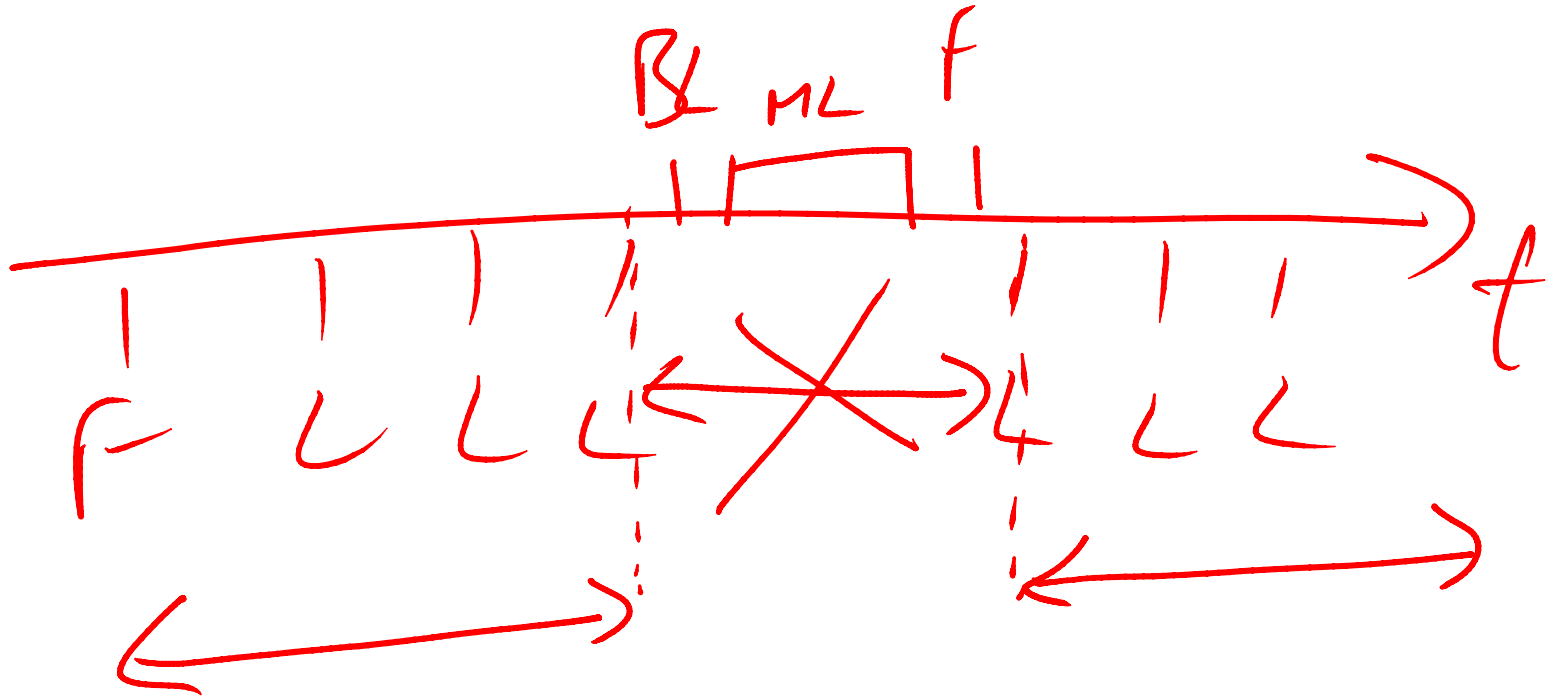
RECOVERY_PENDING

SUSPECT

M5 Discussing database states. RECOVERY_PENDING means recovery needs to be run but could not start. SUSPECT means recovery started but could not finish.

M5556 Example of nested transactions. The inner transactions are not real transactions – do not generate LOP_BEGIN_XACT/LOP_COMMIT_XACT log records and committing them does nothing.



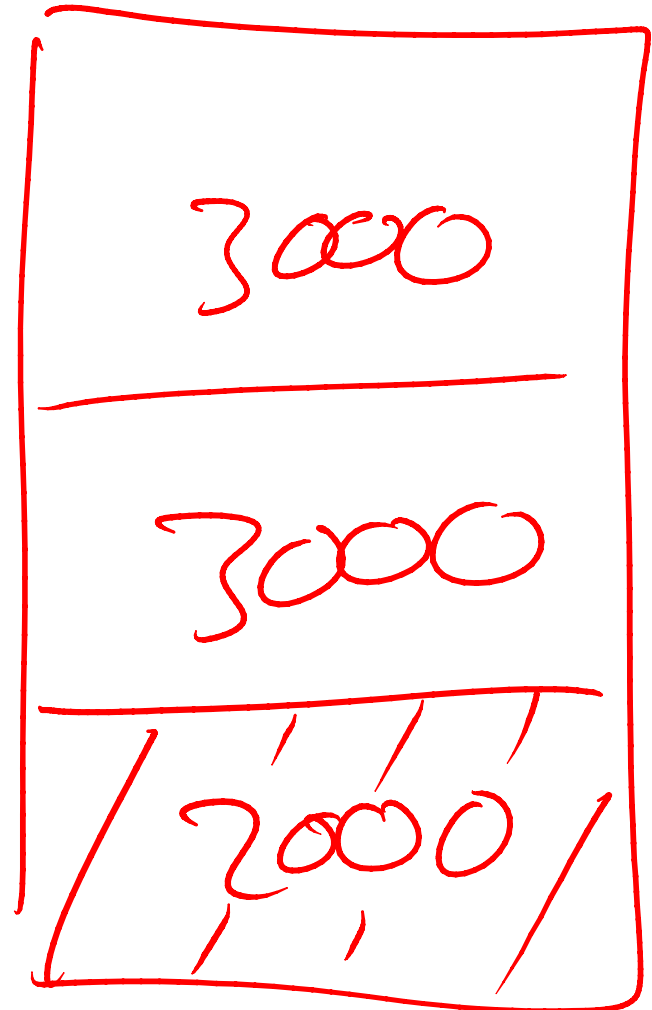
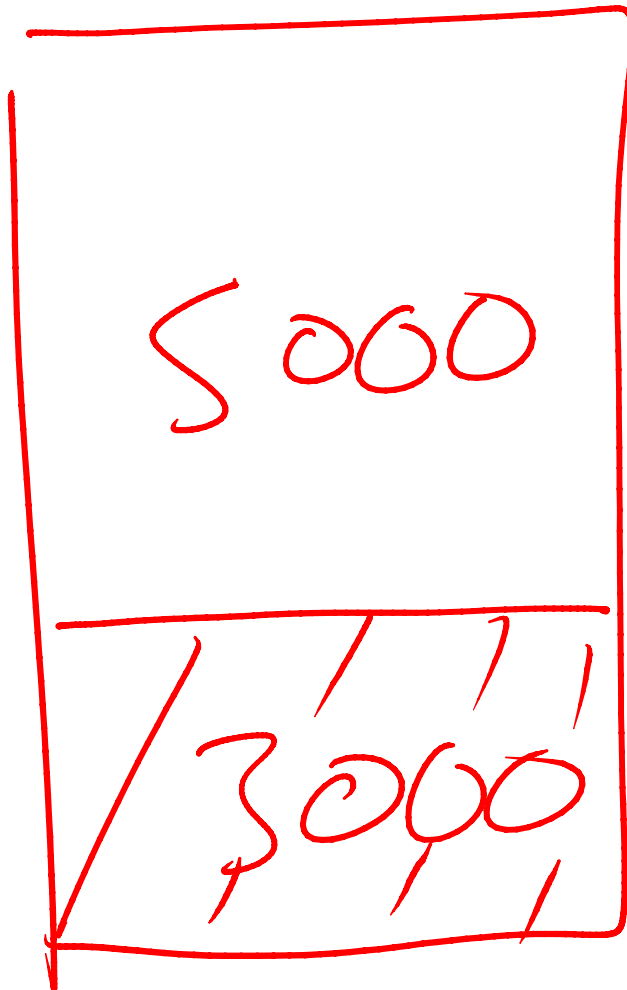


M5S60 If a log backup contains a minimally-logged operation, it cannot be used to do a point-in-time restore using STOPAT for any point in time between the beginning and end of the backup. It can be used as part of a restore sequence as normal though.



M5S25 VLF fragmentation is bad because of having to search through the VLFs for a particular VLF sequence number. See the KB article link for more info.

LSN LSN
VLF

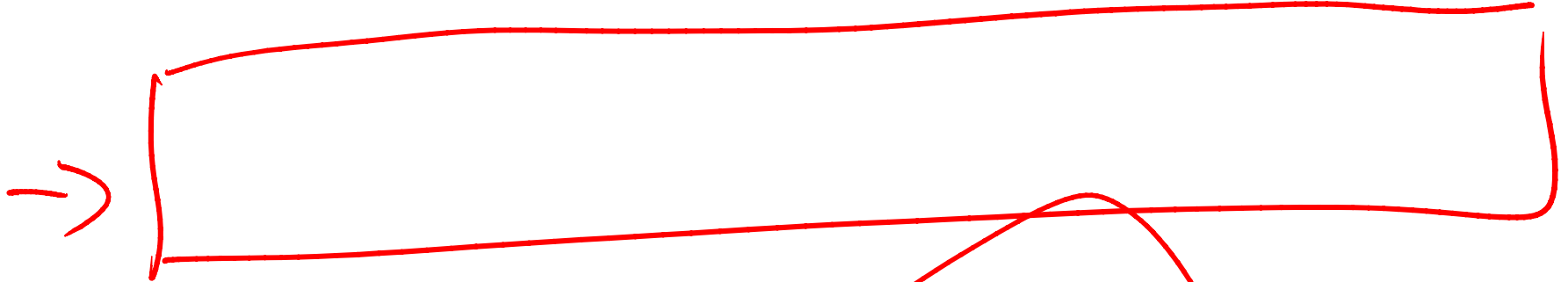
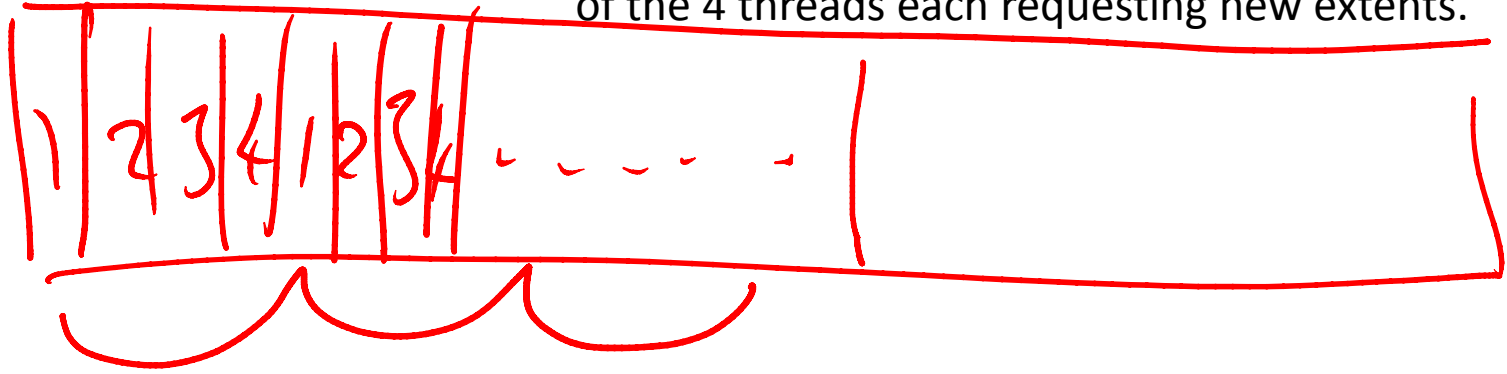


M8S13 Explaining how fixed-size large rows leads to low page density and wasted space.

-E

64

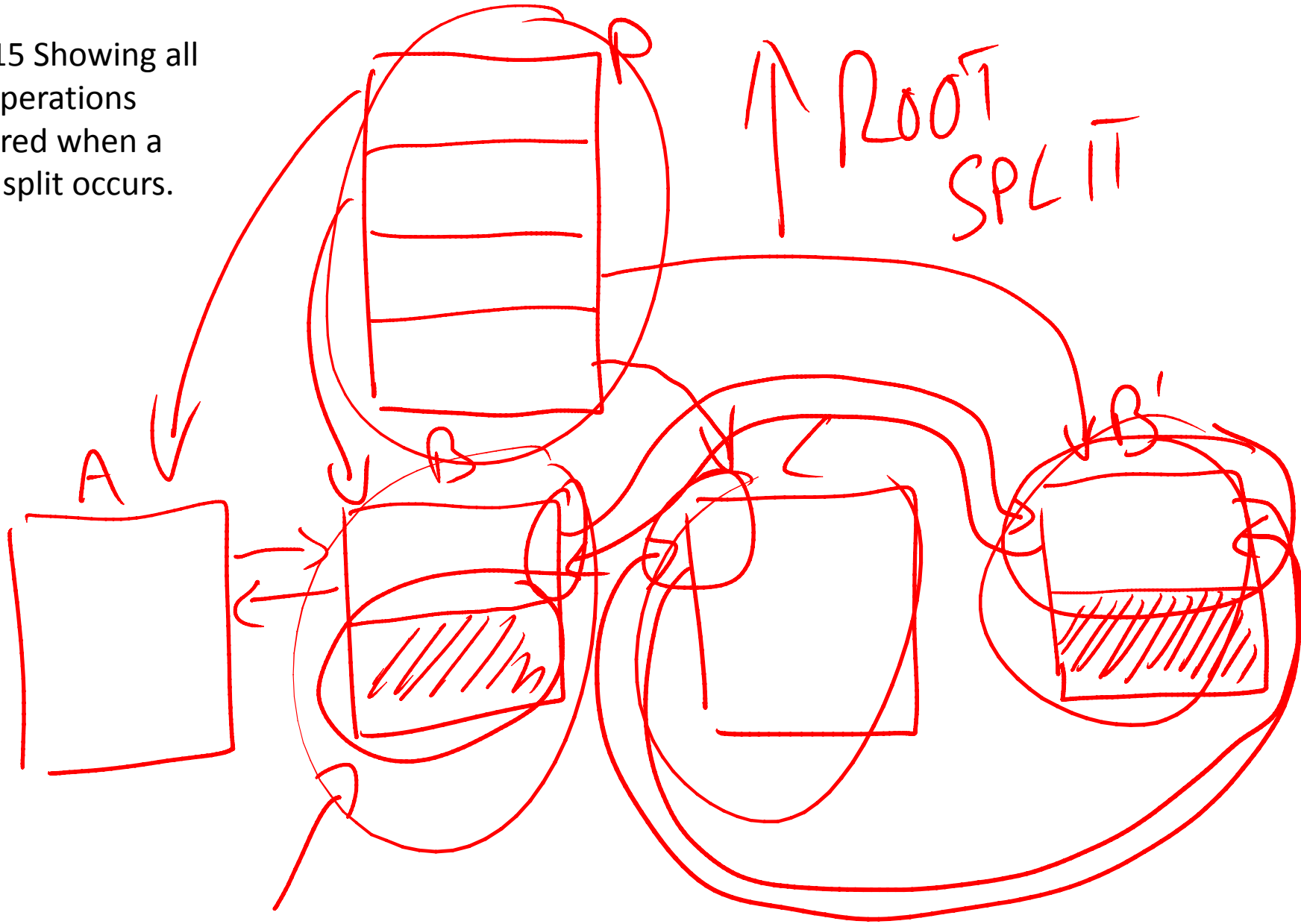
M8S14 Explaining index rebuild DOP 4 after using -E will not lead to large contiguous index ranges because of the 4 threads each requesting new extents.

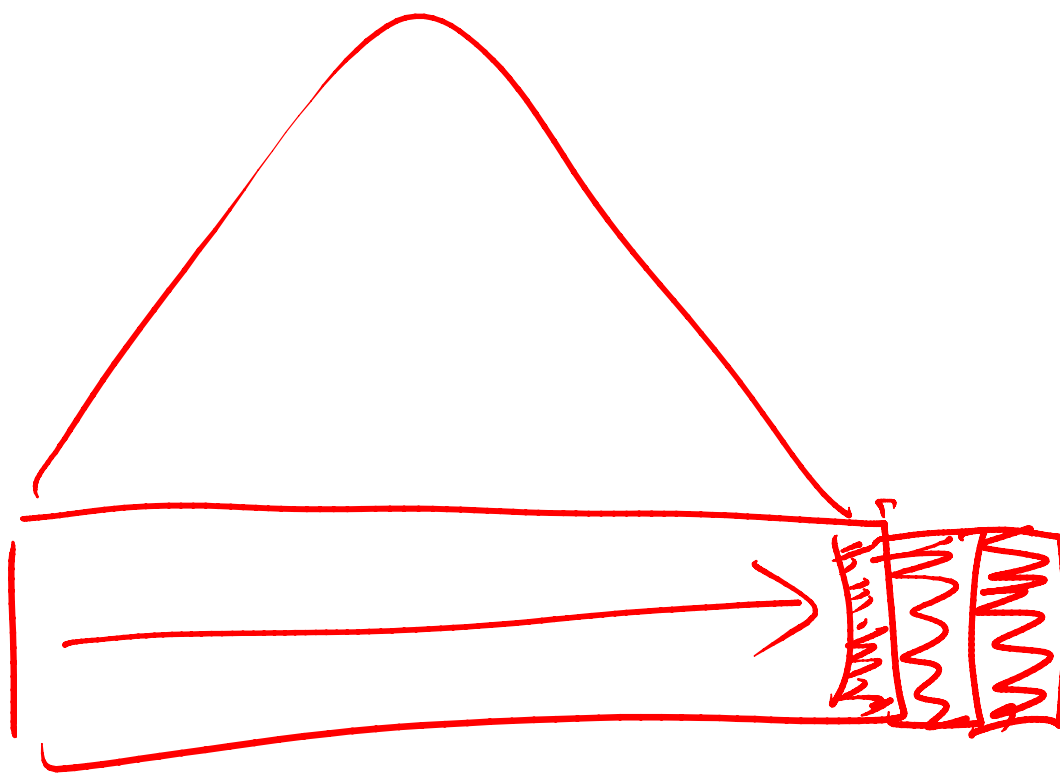


DOP = 4

A sequence of four overlapping circles. The first circle is labeled 4, and the subsequent three circles are crossed out with diagonal lines, indicating a sequence of four threads.

M8S15 Showing all
the operations
required when a
page split occurs.





LOP_DELETE_SPLIT

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.

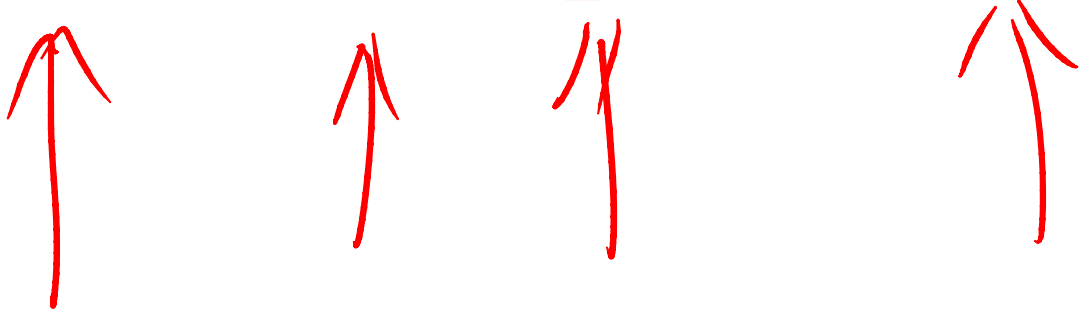
M8S22 The numbers from the page split logging cost demo.

	NO SPLIT	SPLIT	
1000	1244	6772	<u>x5</u>
100	344	5964	<u><u>x18</u></u>
10	256	10364	<u><u>x45</u></u>

COMMENTS

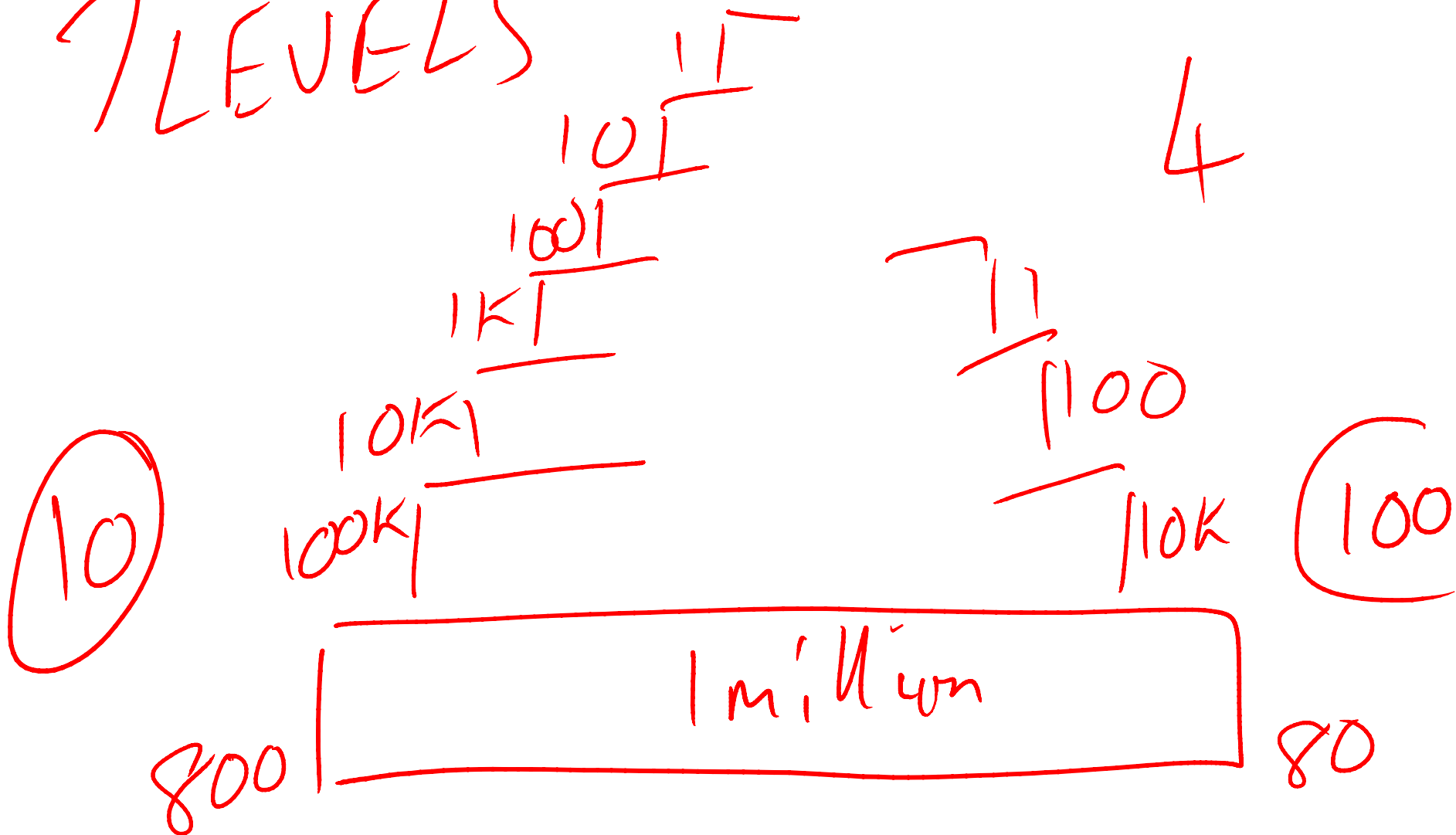
MEMBER_ID

PAUL	LYNDI	BILL	K	PHIL
------	-------	------	---	------



M8S23 The MySpace story. The 'K' is the tiny portion of comments on Kimberly's page because she wasn't very popular back then... 😊

7 LEVELS



M8 Explaining 'fanout'. Larger index key size means less rows per page on each index level, meaning more index levels and more CPU for index traversal from root to leaf.

ITB



M8S37 Explaining staggered index maintenance with database mirroring. See <http://bit.ly/IS04W5> for more explanation