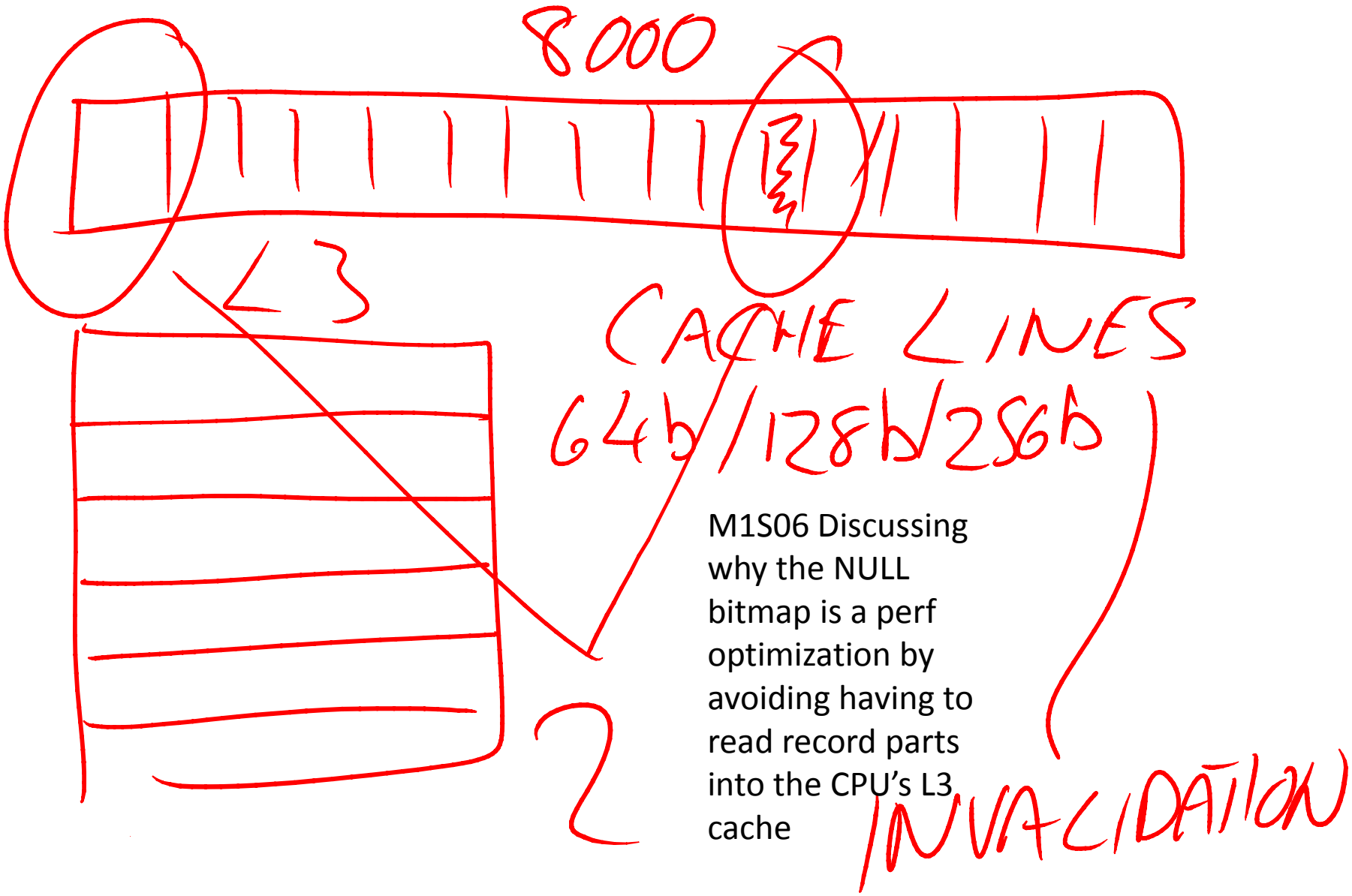
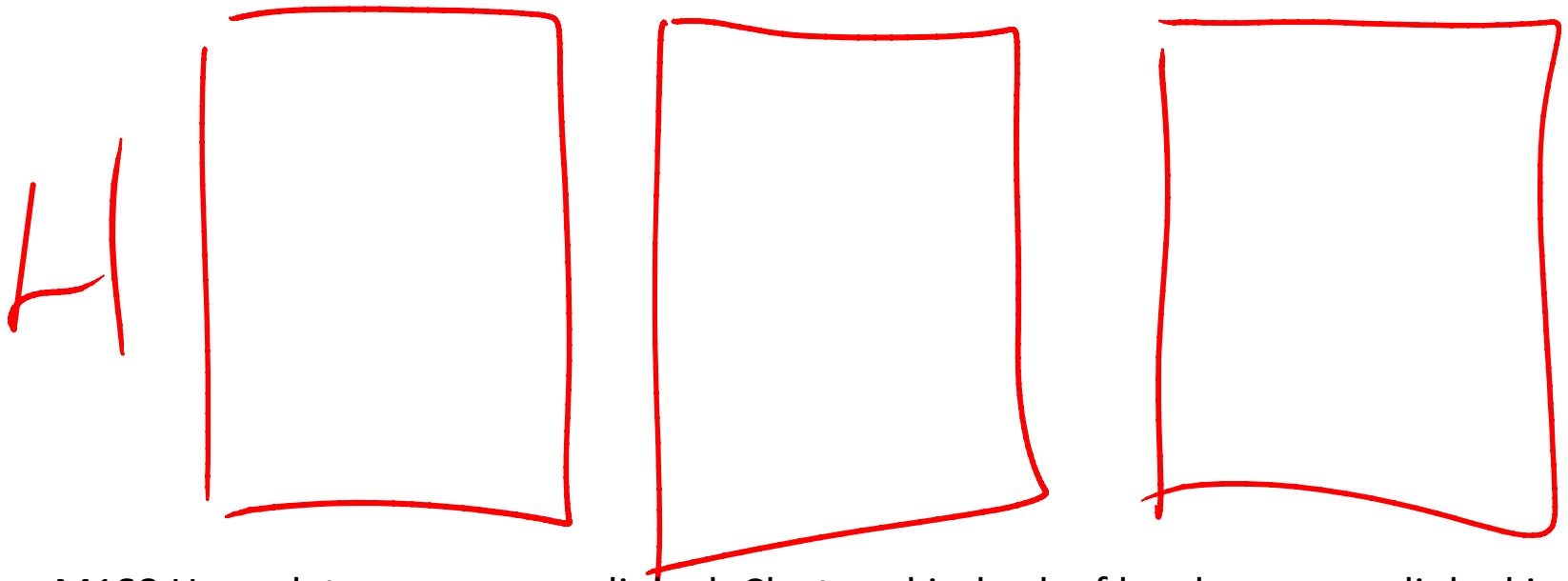
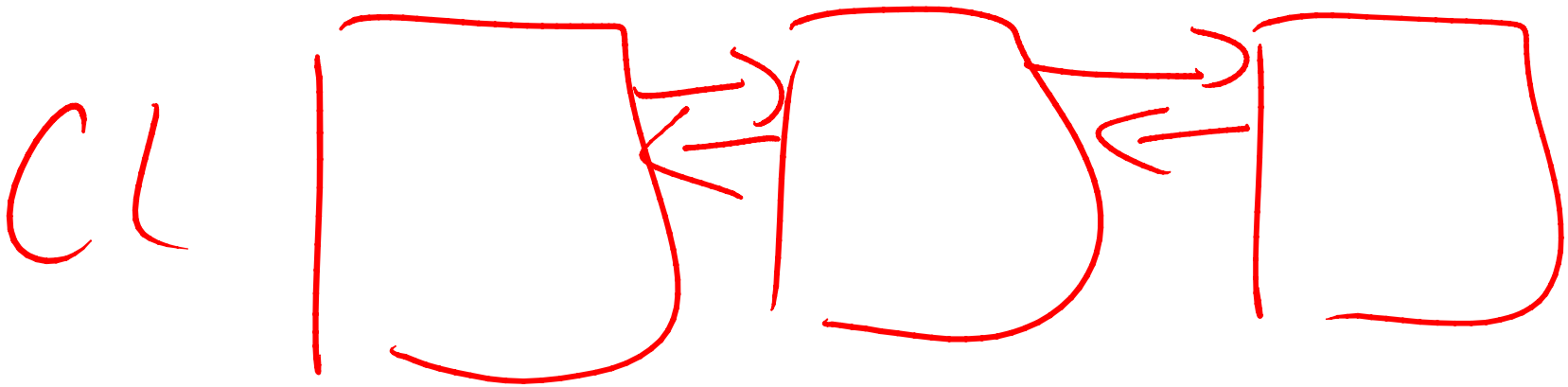
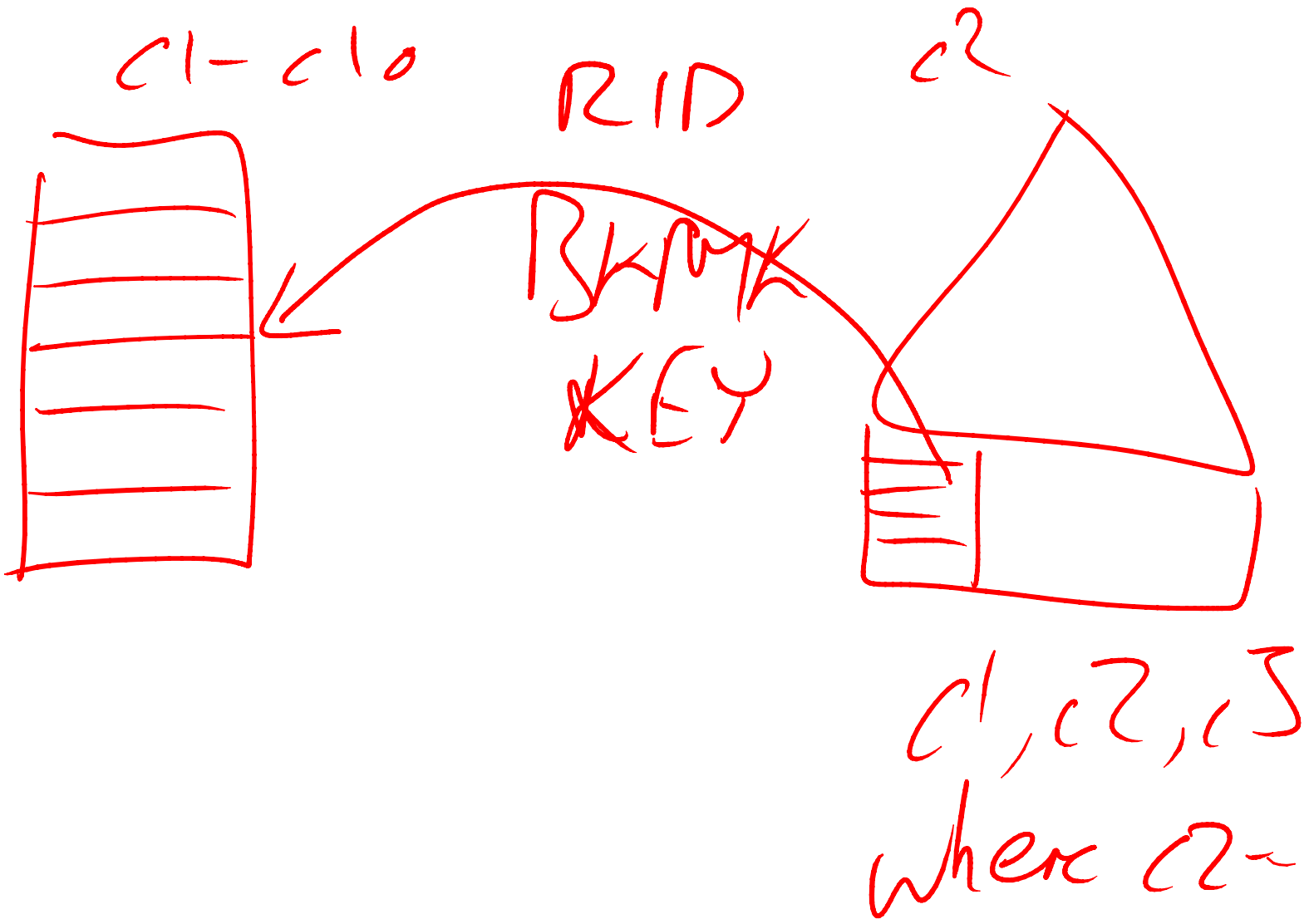


M1S06 Discussing why the NULL bitmap is a perf optimization by avoiding having to read record parts into the CPU's L3 cache

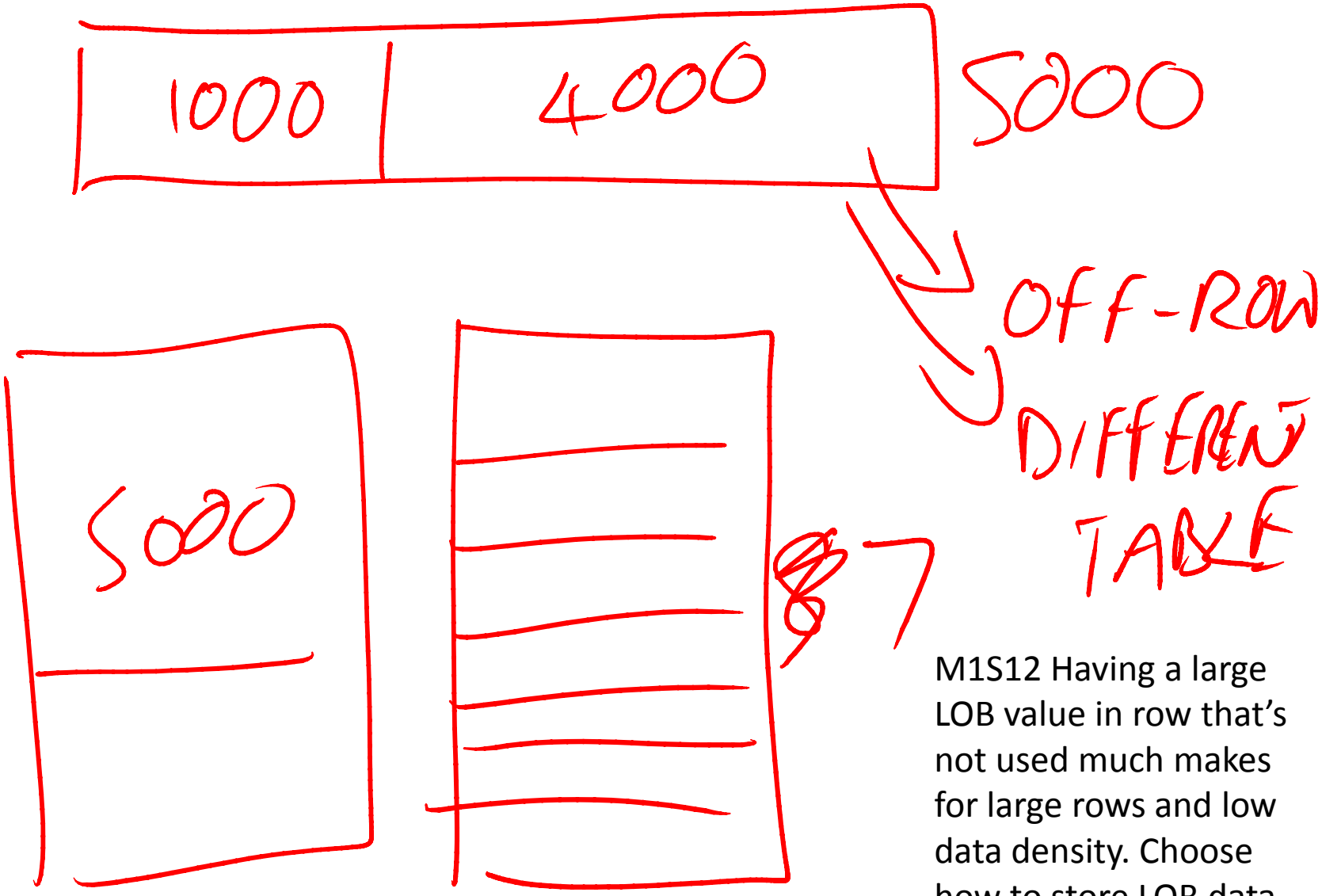


M1S8 Heap data pages are unlinked. Clustered index leaf level pages are linked in a doubly-linked list.

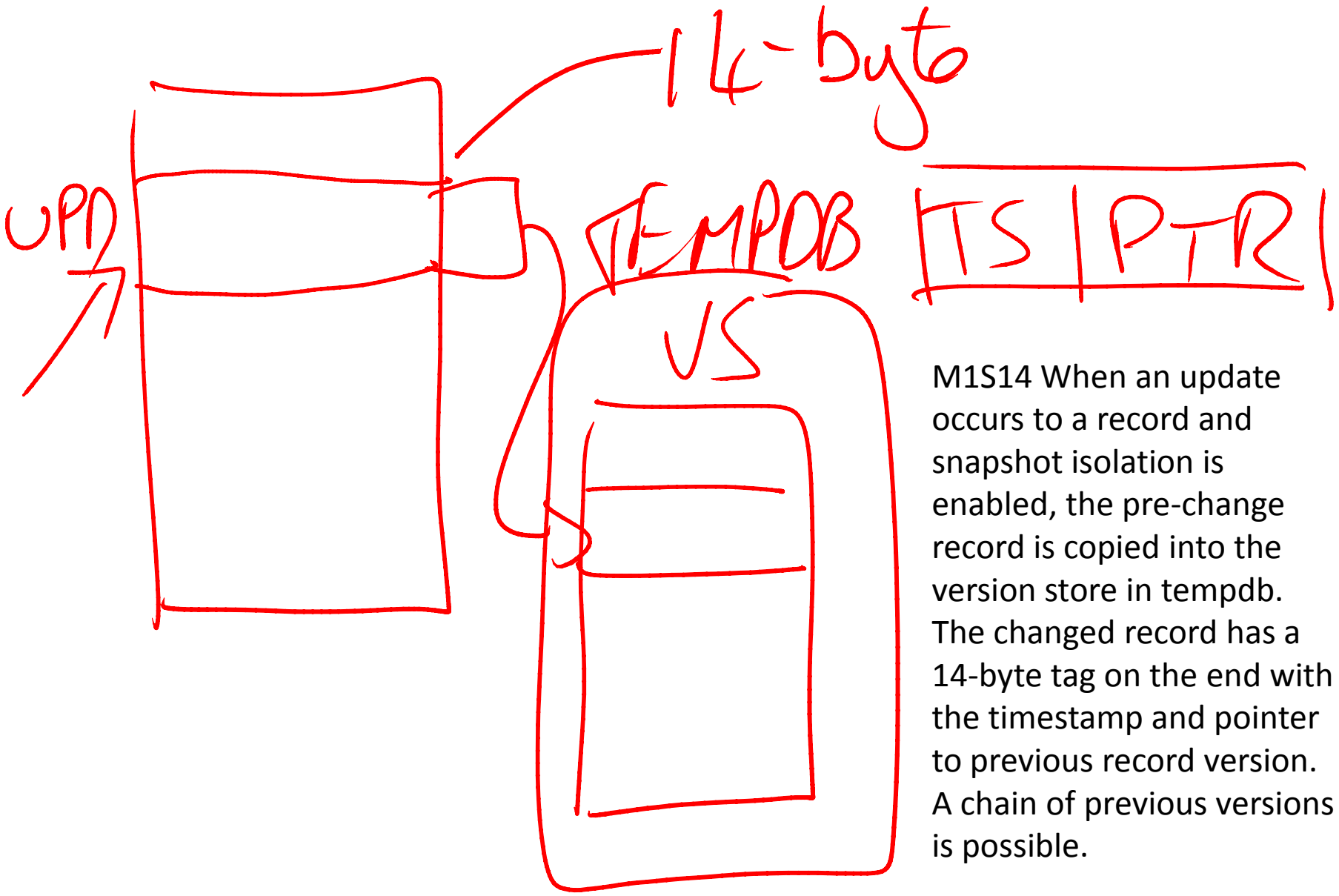




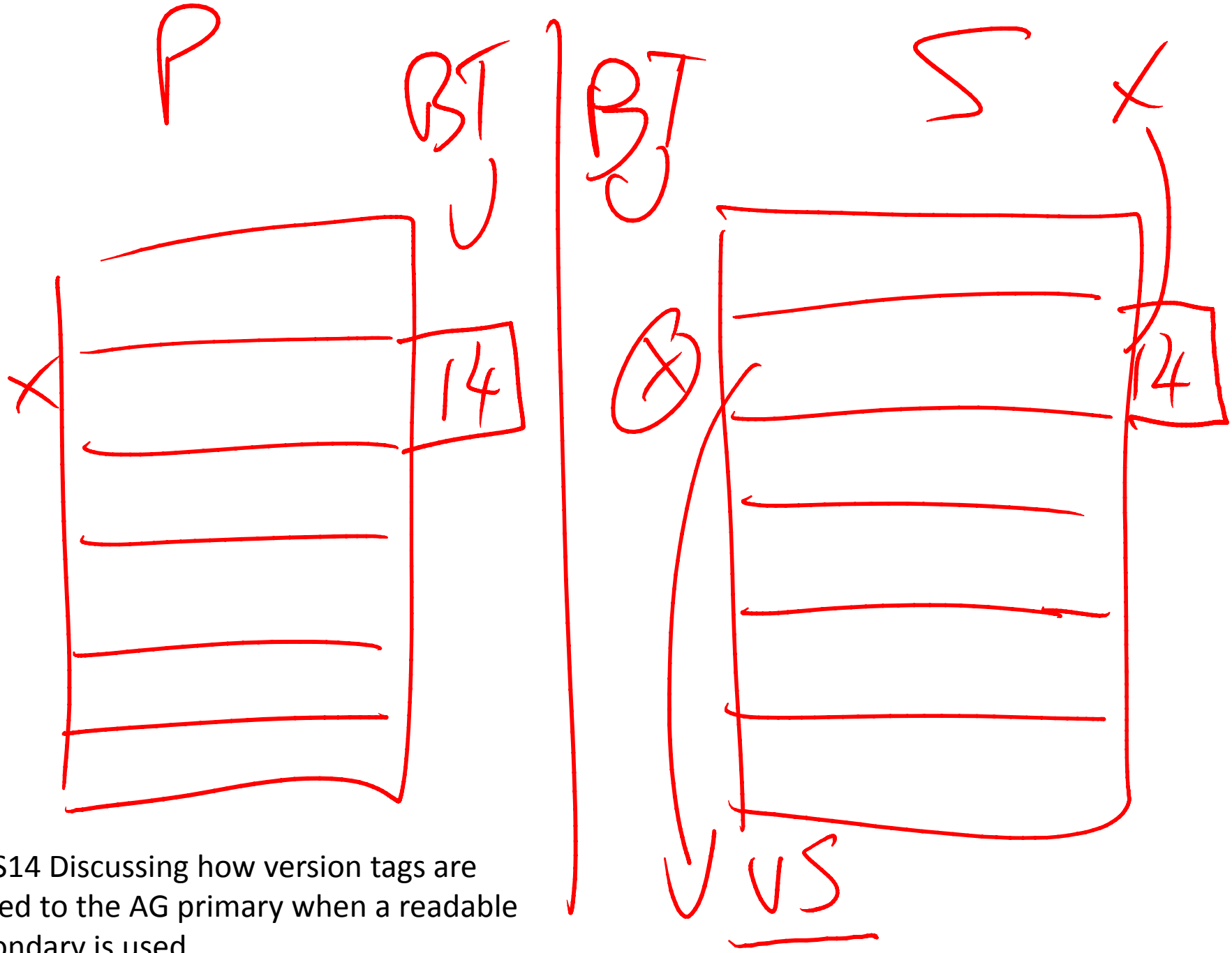
M1S09 Explaining NC index back-links to the data records as part of describing why forwarding/forwarded records exist. Without them, moving a heap row would entail changing all NC index records with a back-link to that row.



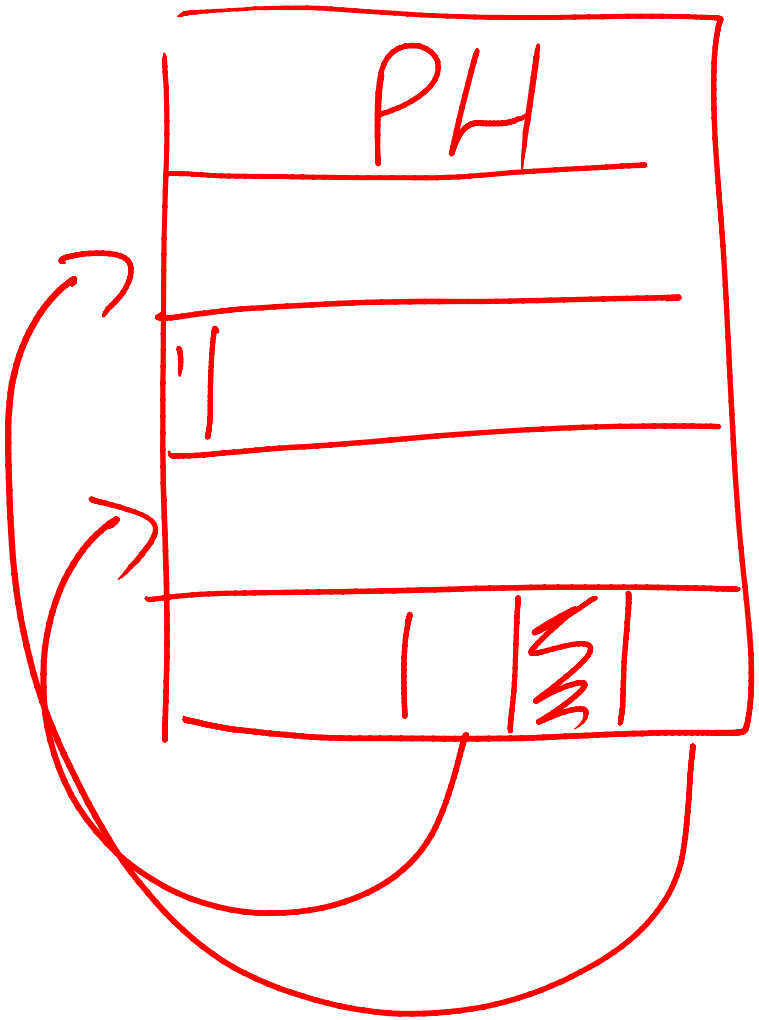
M1S12 Having a large LOB value in row that's not used much makes for large rows and low data density. Choose how to store LOB data wisely.



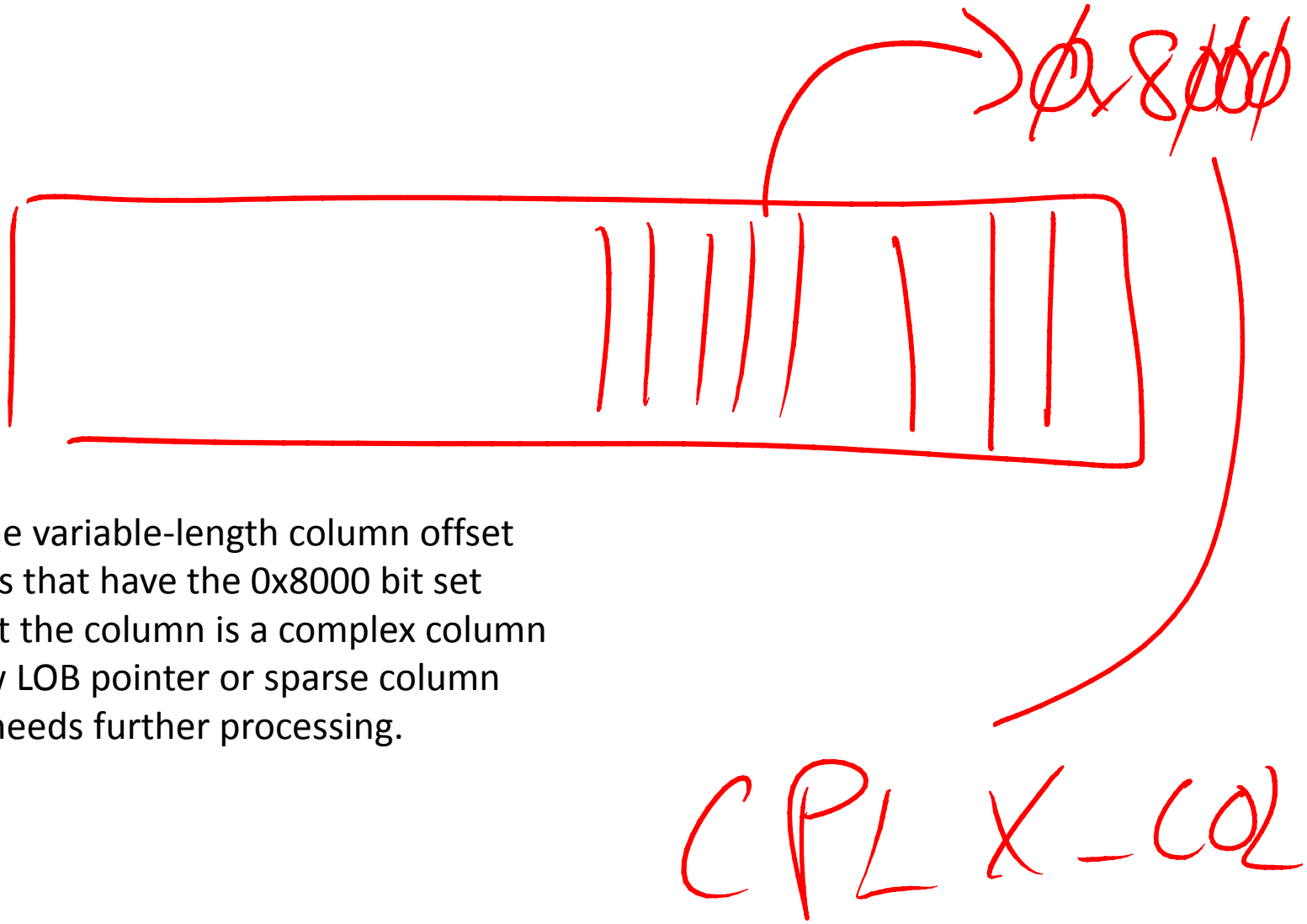
M1S14 When an update occurs to a record and snapshot isolation is enabled, the pre-change record is copied into the version store in tempdb. The changed record has a 14-byte tag on the end with the timestamp and pointer to previous record version. A chain of previous versions is possible.



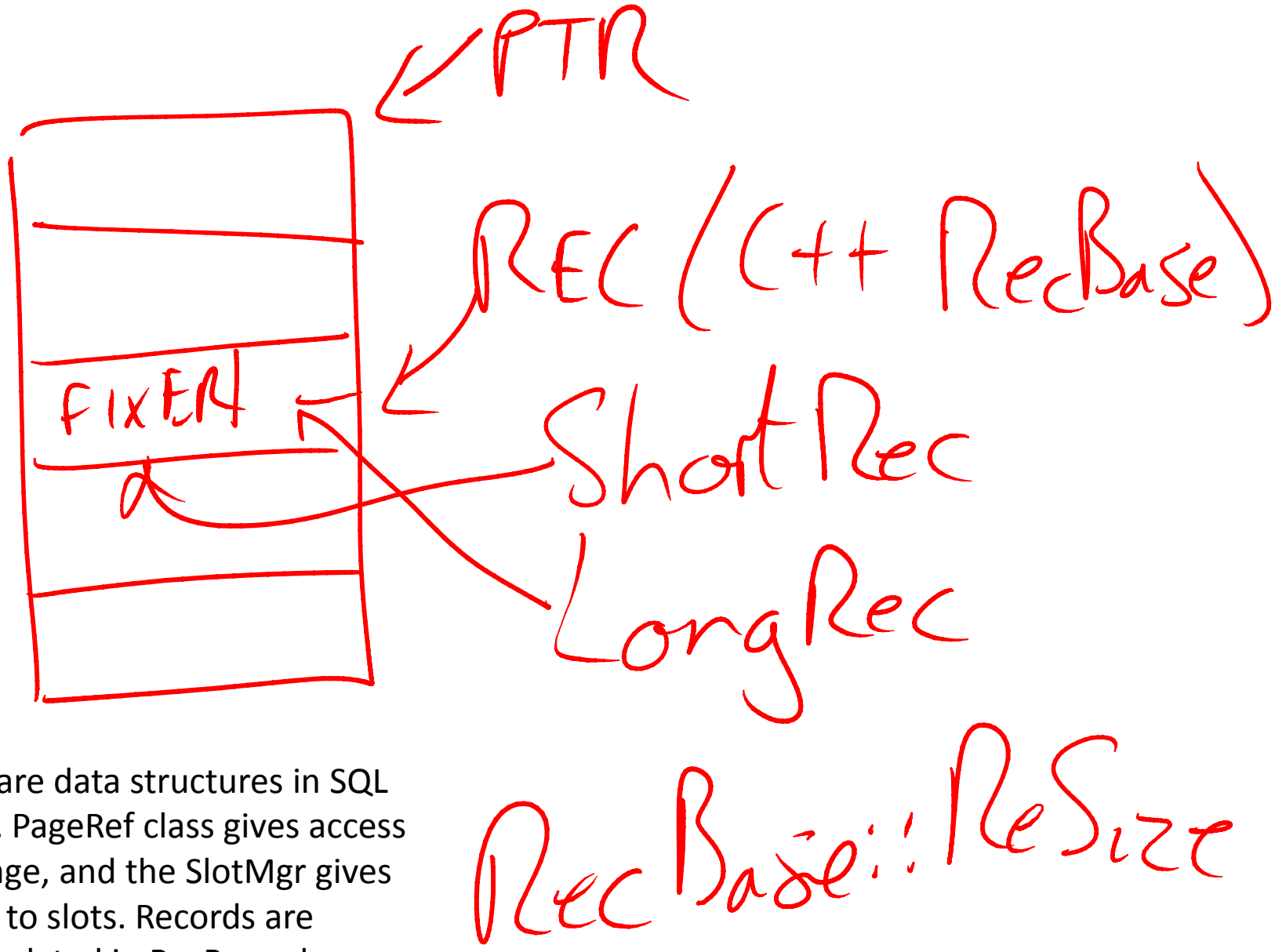
M1S14 Discussing how version tags are added to the AG primary when a readable secondary is used.



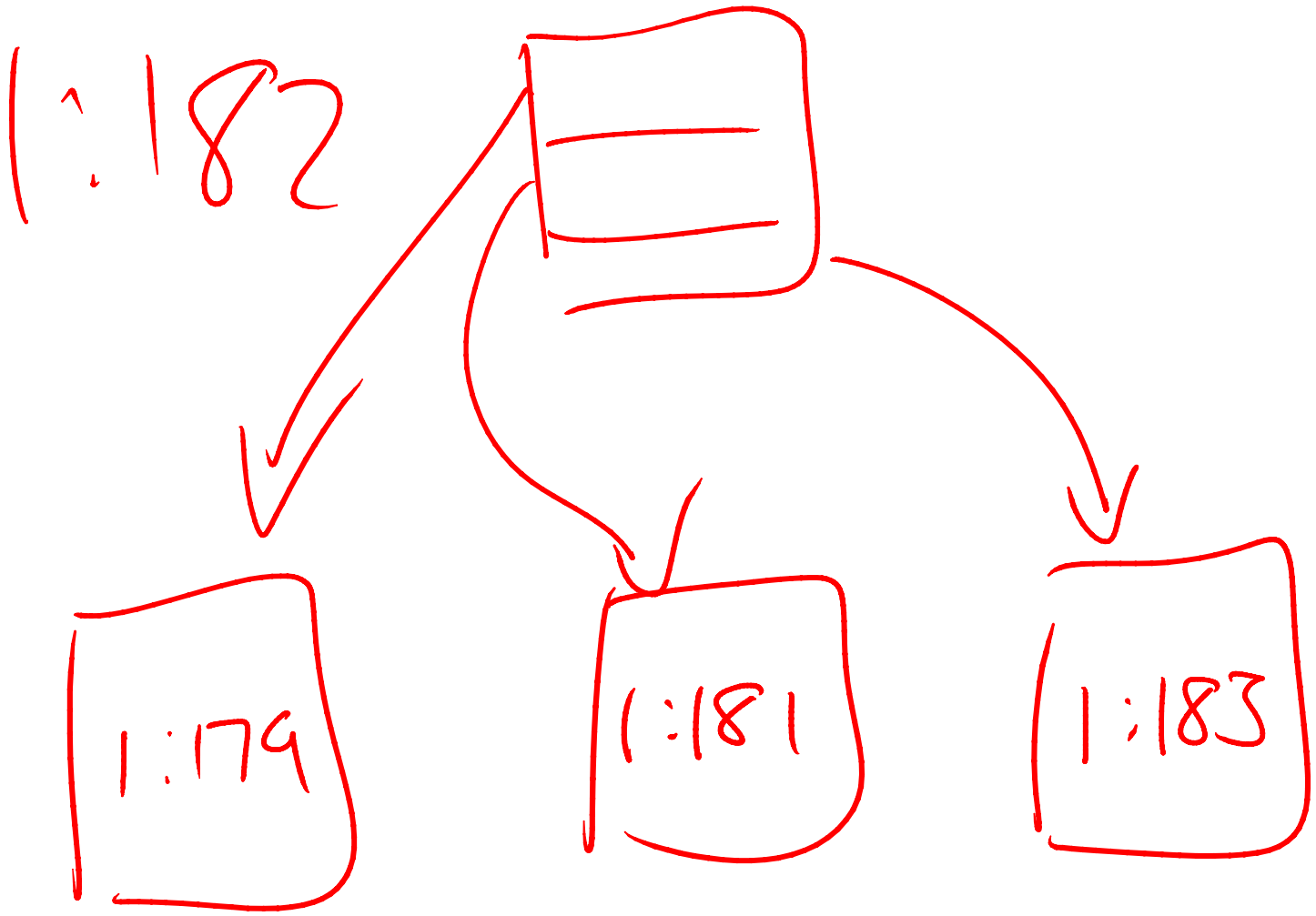
M1S15 When a record is deleted, it's marked as a ghost record. The ghost cleanup task does not physically delete the records – it removes the slot array entry that points to the record on the page. I.e. the record is unmarked as being a record and the area on the page is now free space – that just happens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



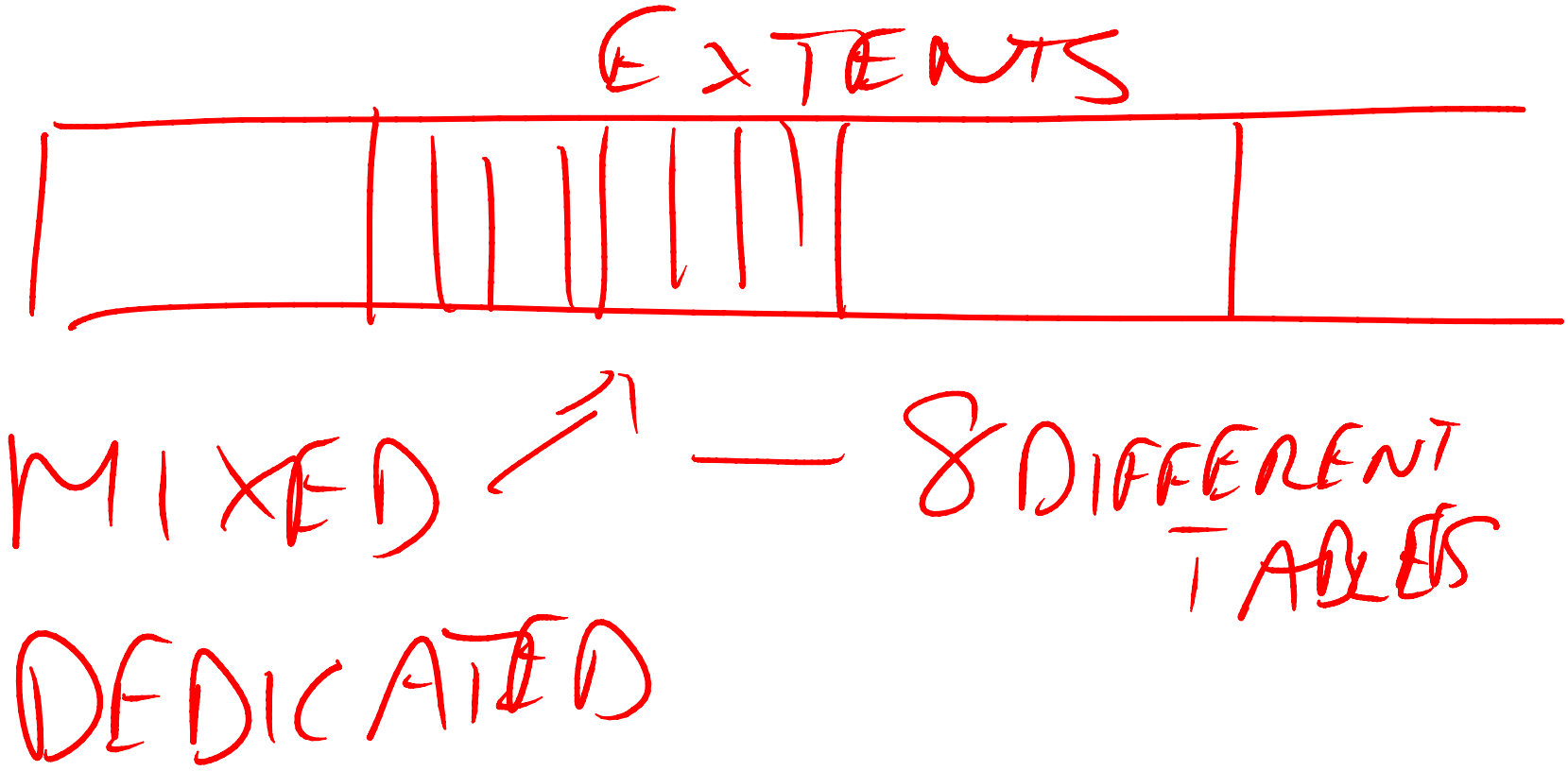
M1S11 In the variable-length column offset array, offsets that have the 0x8000 bit set indicate that the column is a complex column (e.g. off-row LOB pointer or sparse column array) that needs further processing.



Pages are data structures in SQL Server. PageRef class gives access to a page, and the SlotMgr gives access to slots. Records are encapsulated in RecBase classes.

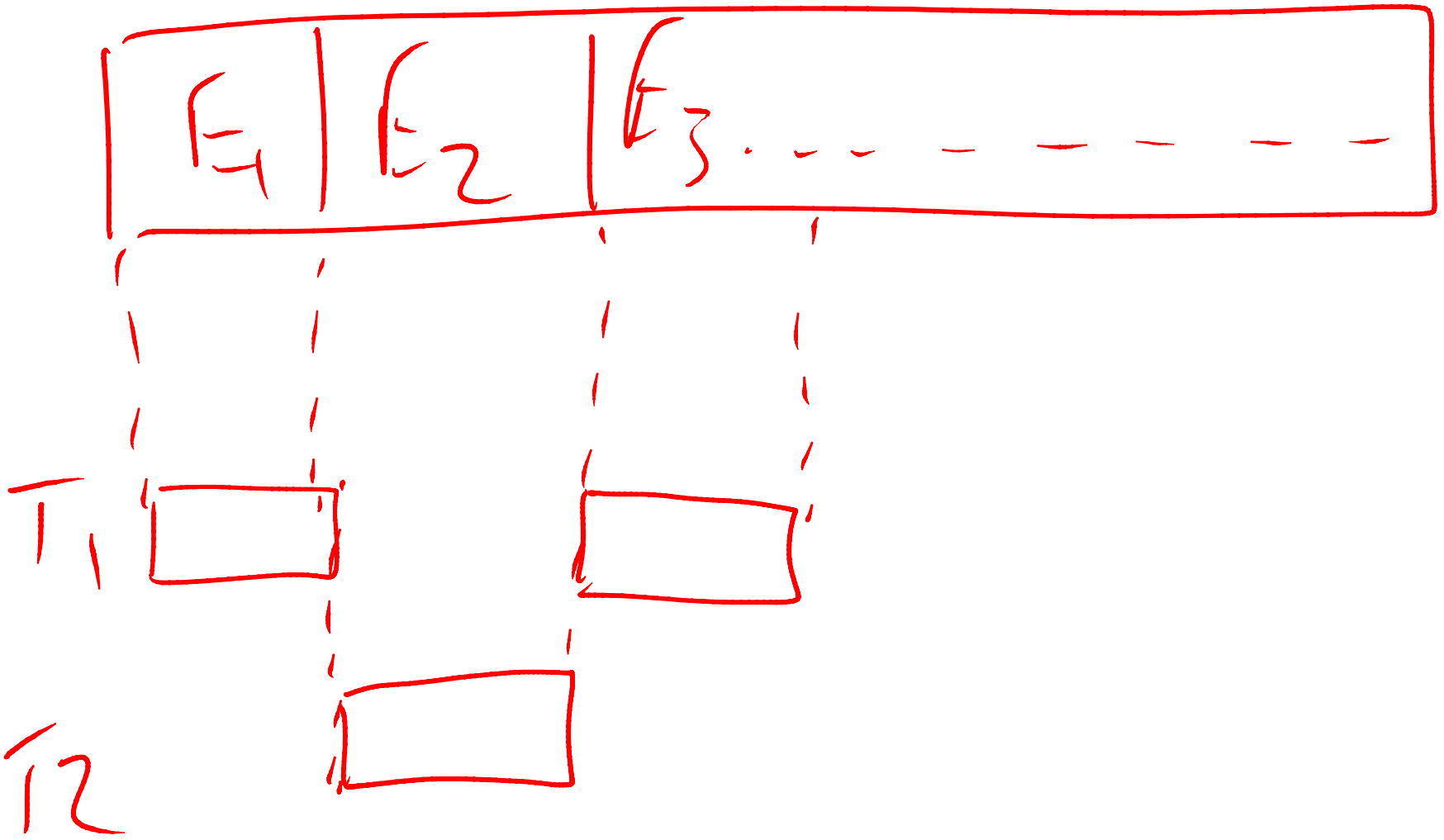


M1S22 Showing the clustered index structure generated by the page/record demo.



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.

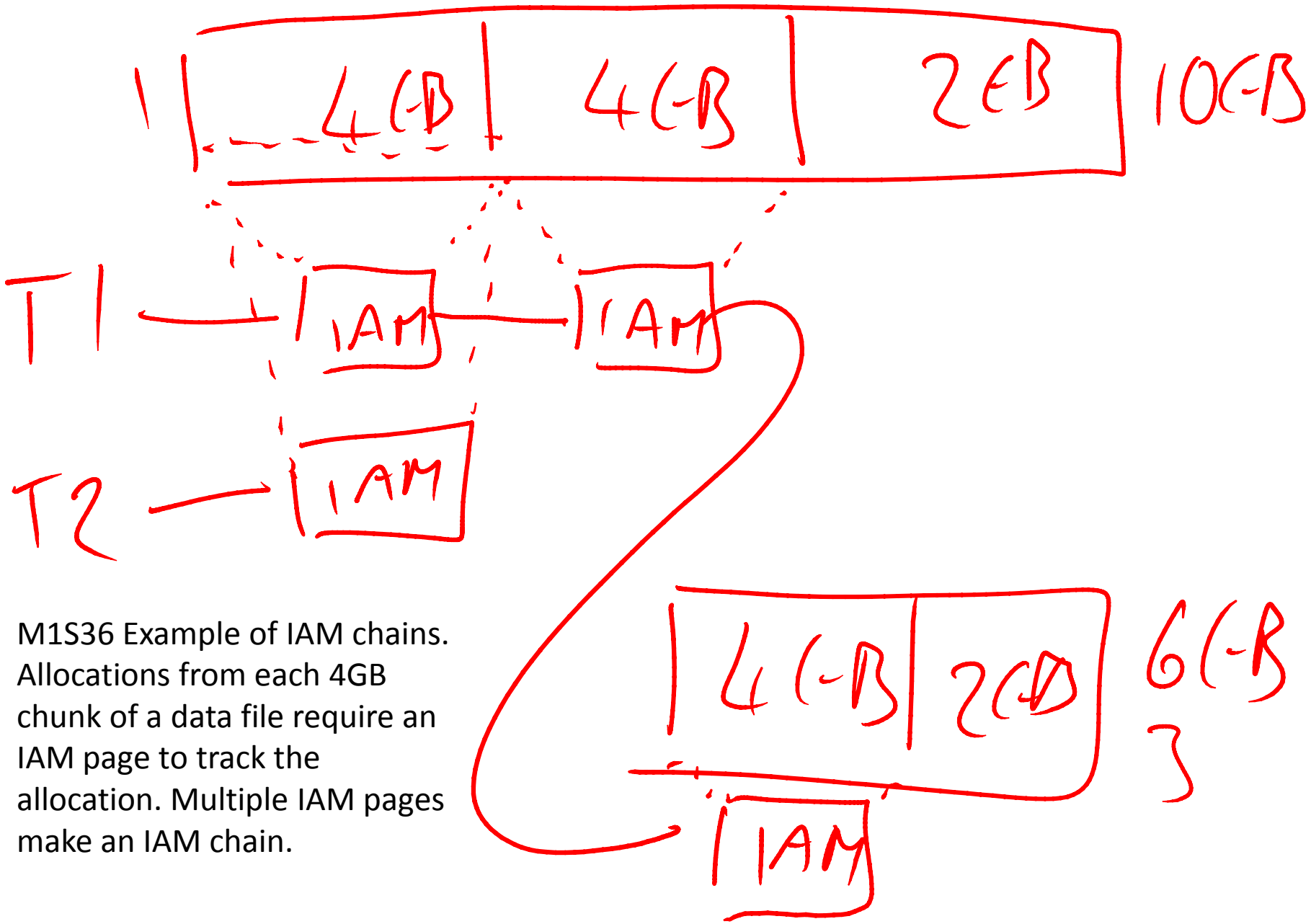
Later in the deck I refine ‘object’ to really be ‘allocation unit’.



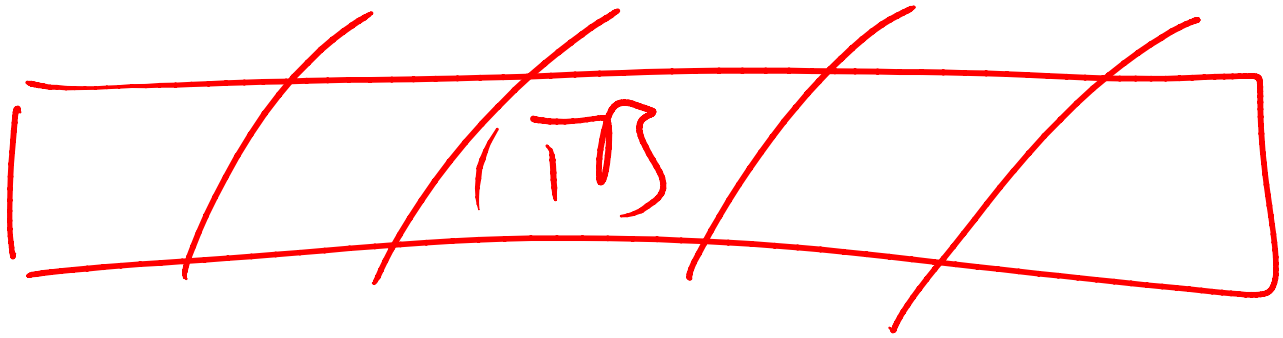
M1S24 Explaining about dedicated extents and how the allocation of a dedicated extent to an object really means ‘reserve these 8 pages for exclusive use’ of that object. T2 needs an extent before T1 has finished allocating the 8 pages in it’s first extent.

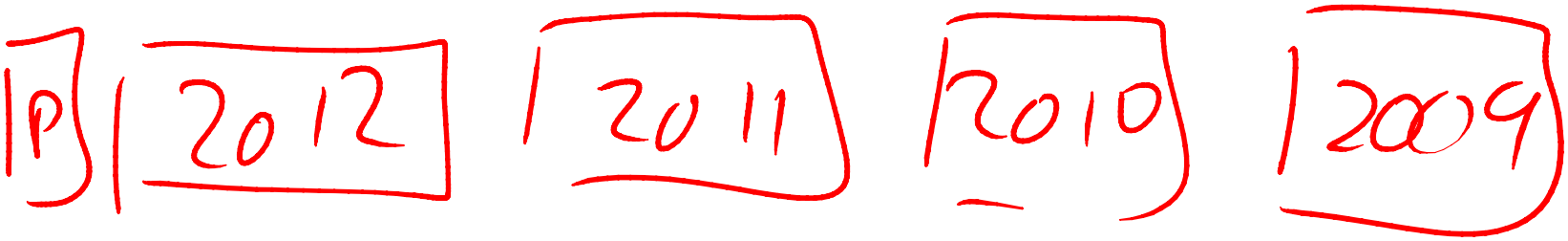
0 – FH
1 – PFS
2 – GAM
3 – SGAM
4 – UNUSED
5 – UNUSED
6 – DIFF
7 – ML

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

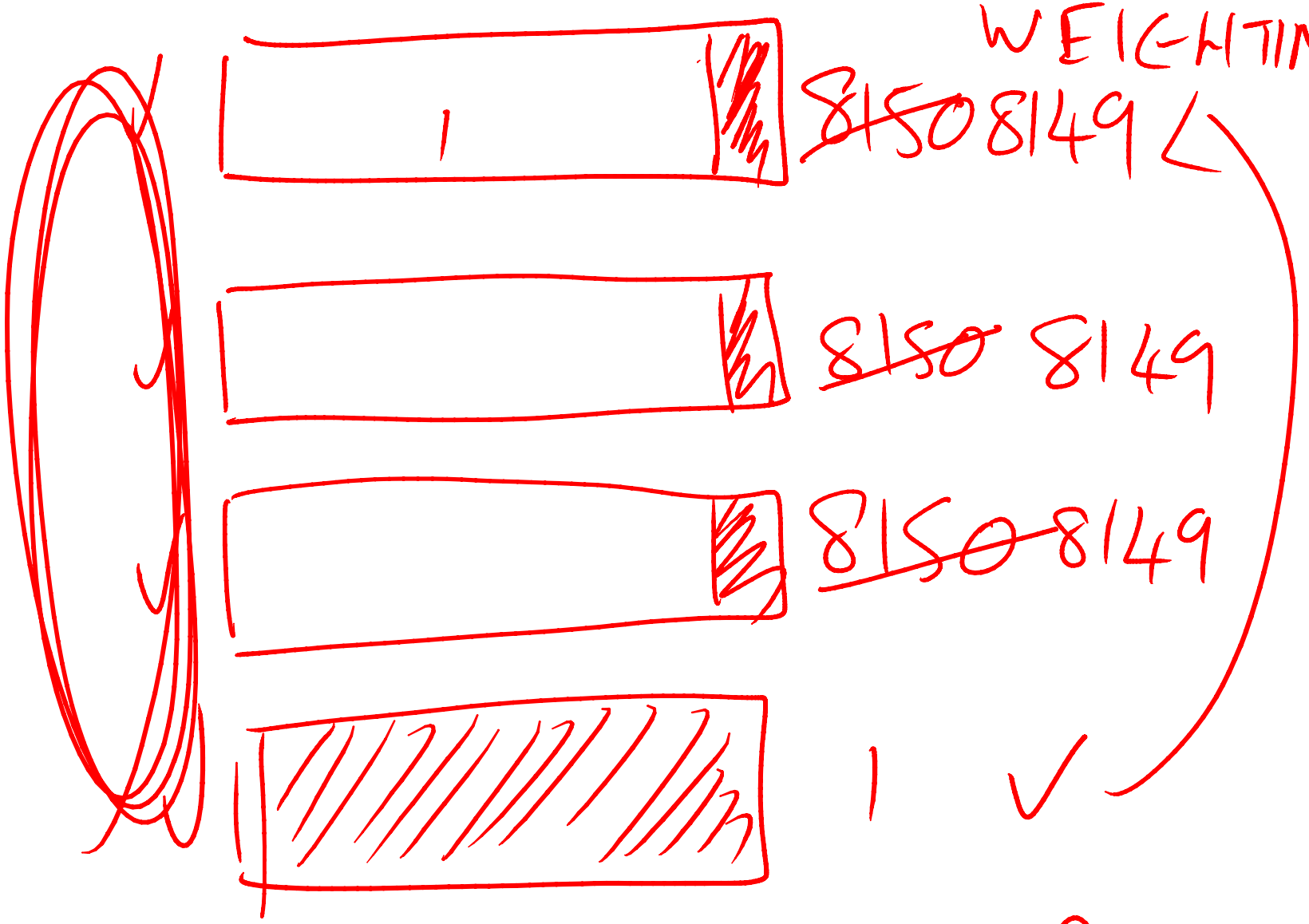


M1S36 Example of IAM chains. Allocations from each 4GB chunk of a data file require an IAM page to track the allocation. Multiple IAM pages make an IAM chain.

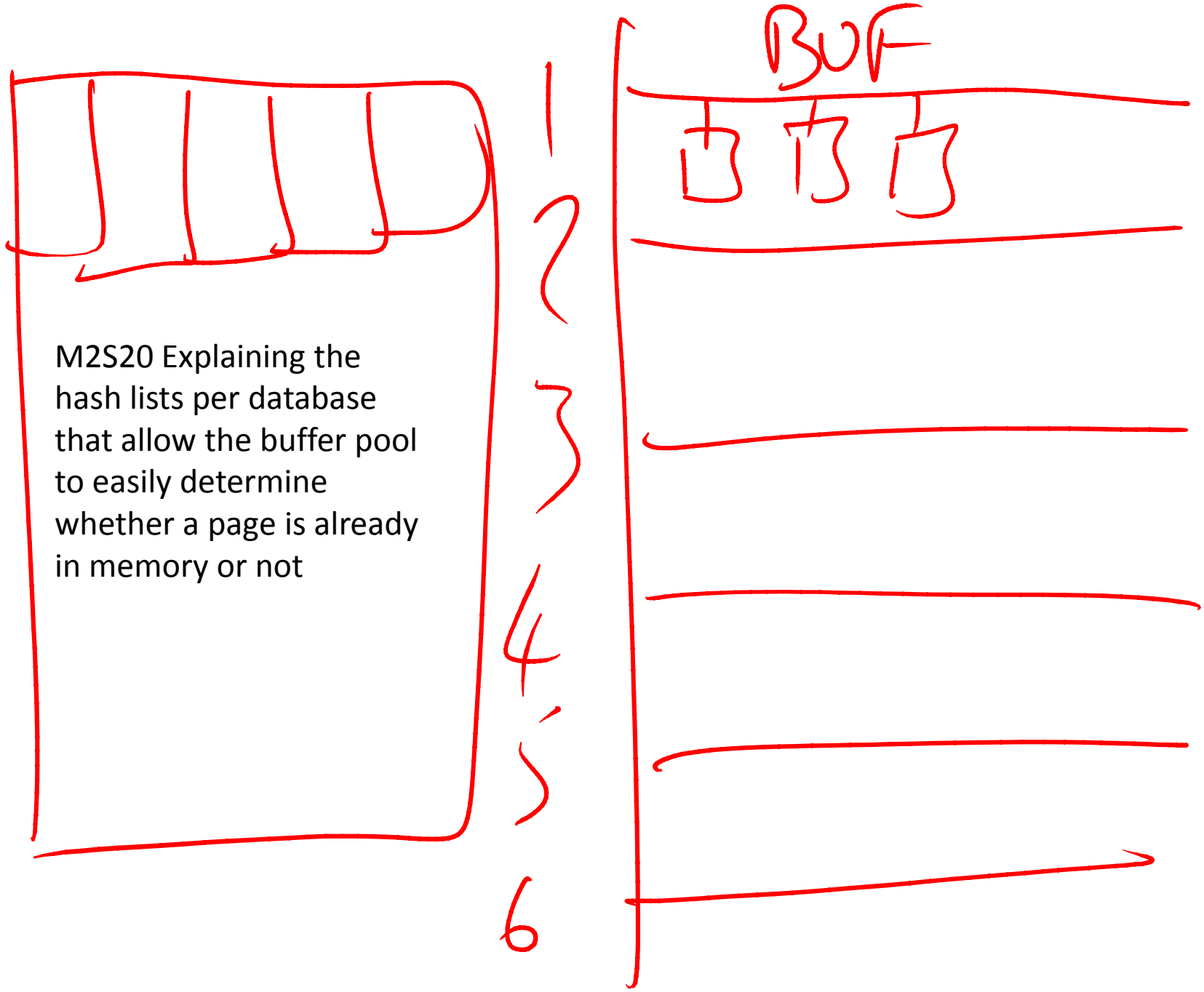




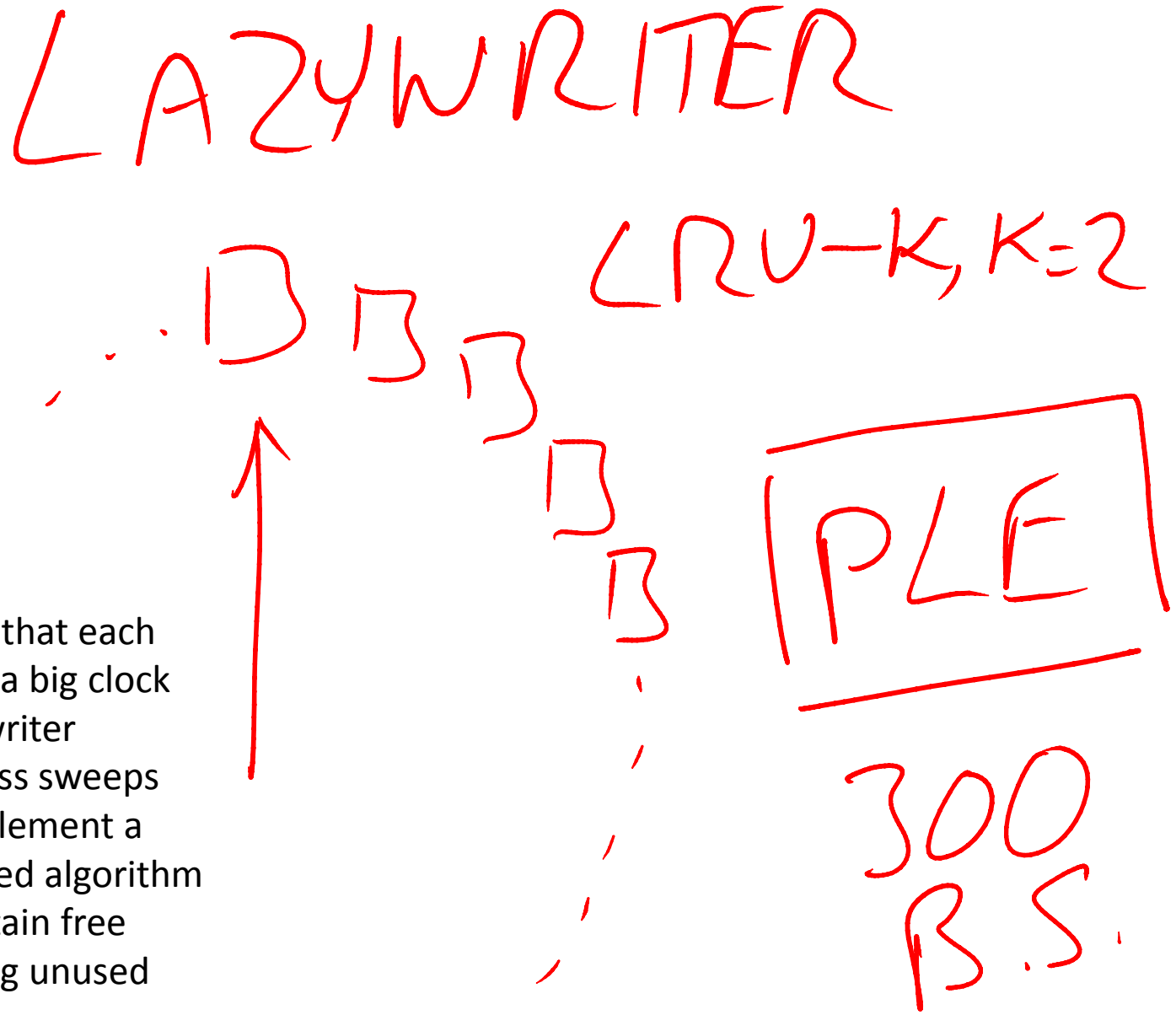
M2S7 Splitting a large database up into multiple filegroups can enable you to do small, targeted restores or even a partial restore from scratch – in the event that the database is damaged or destroyed.



M2S19 Explaining round-robin allocation and proportional fill

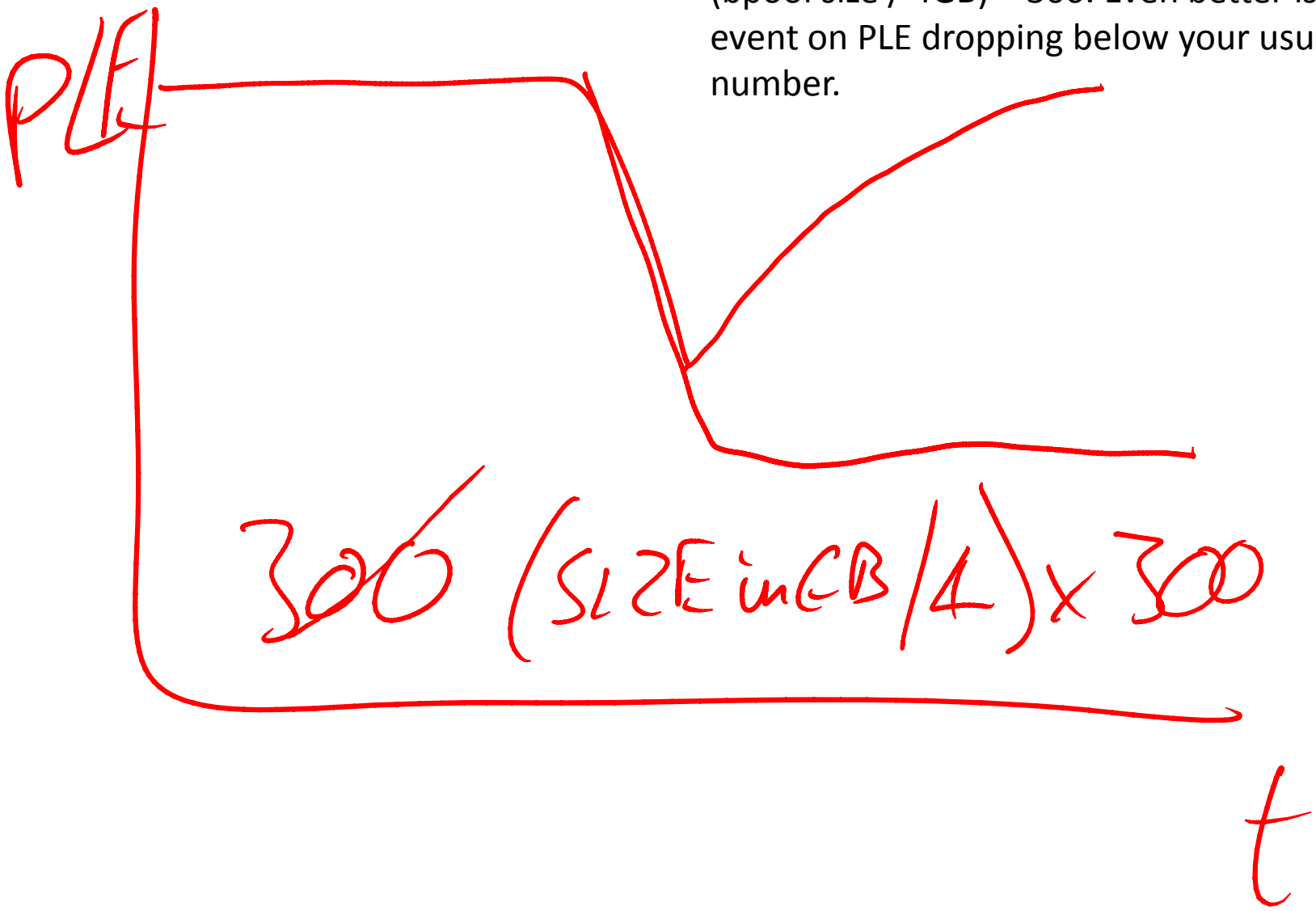


M2S20 Explaining the hash lists per database that allow the buffer pool to easily determine whether a page is already in memory or not

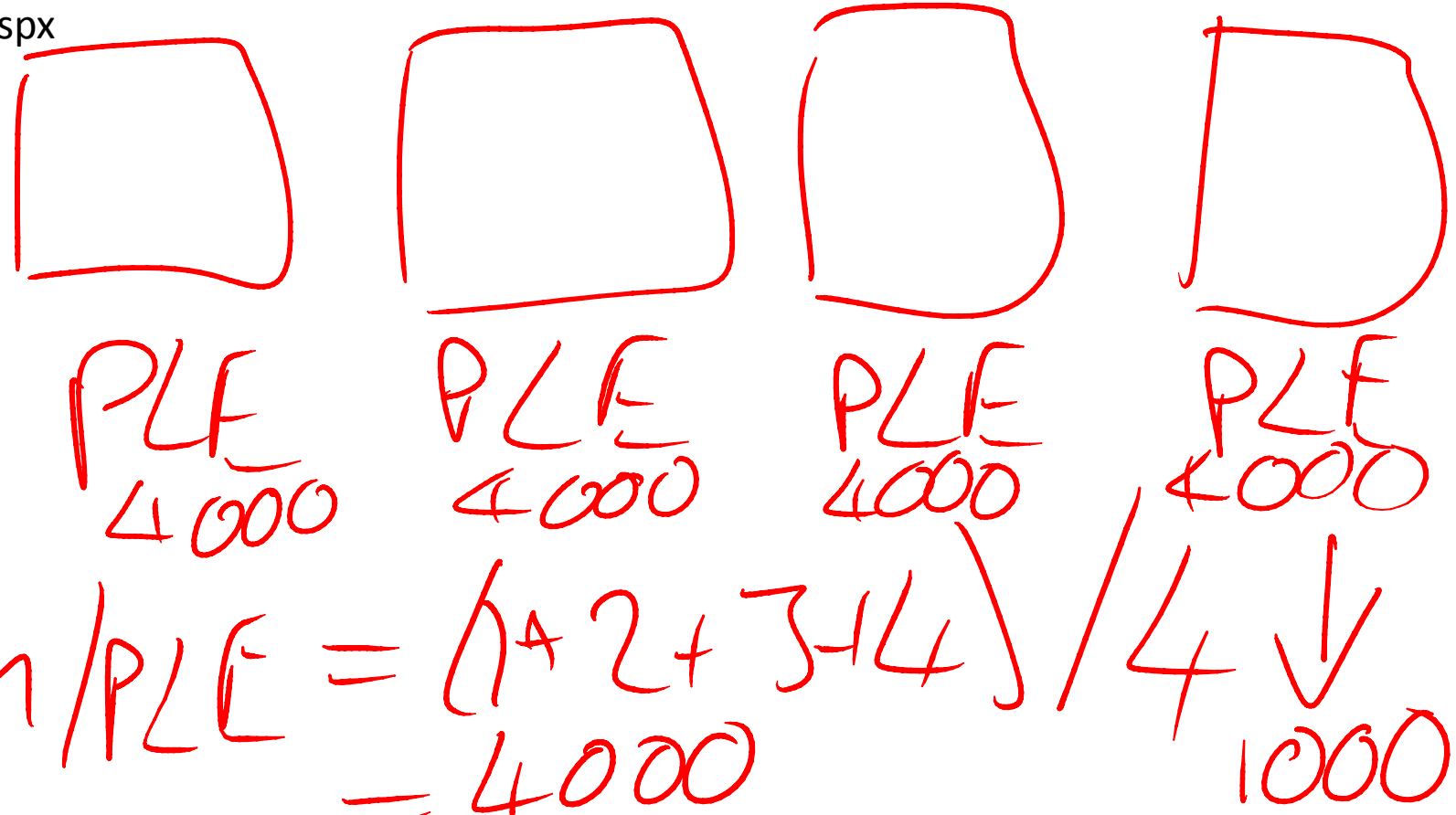


M2S20 Explaining that each page is a point on a big clock face and the lazywriter background process sweeps the clock face implement a Least-Recently-Used algorithm that helps it maintain free space by discarding unused pages.
300 is not a good PLE value as a threshold.

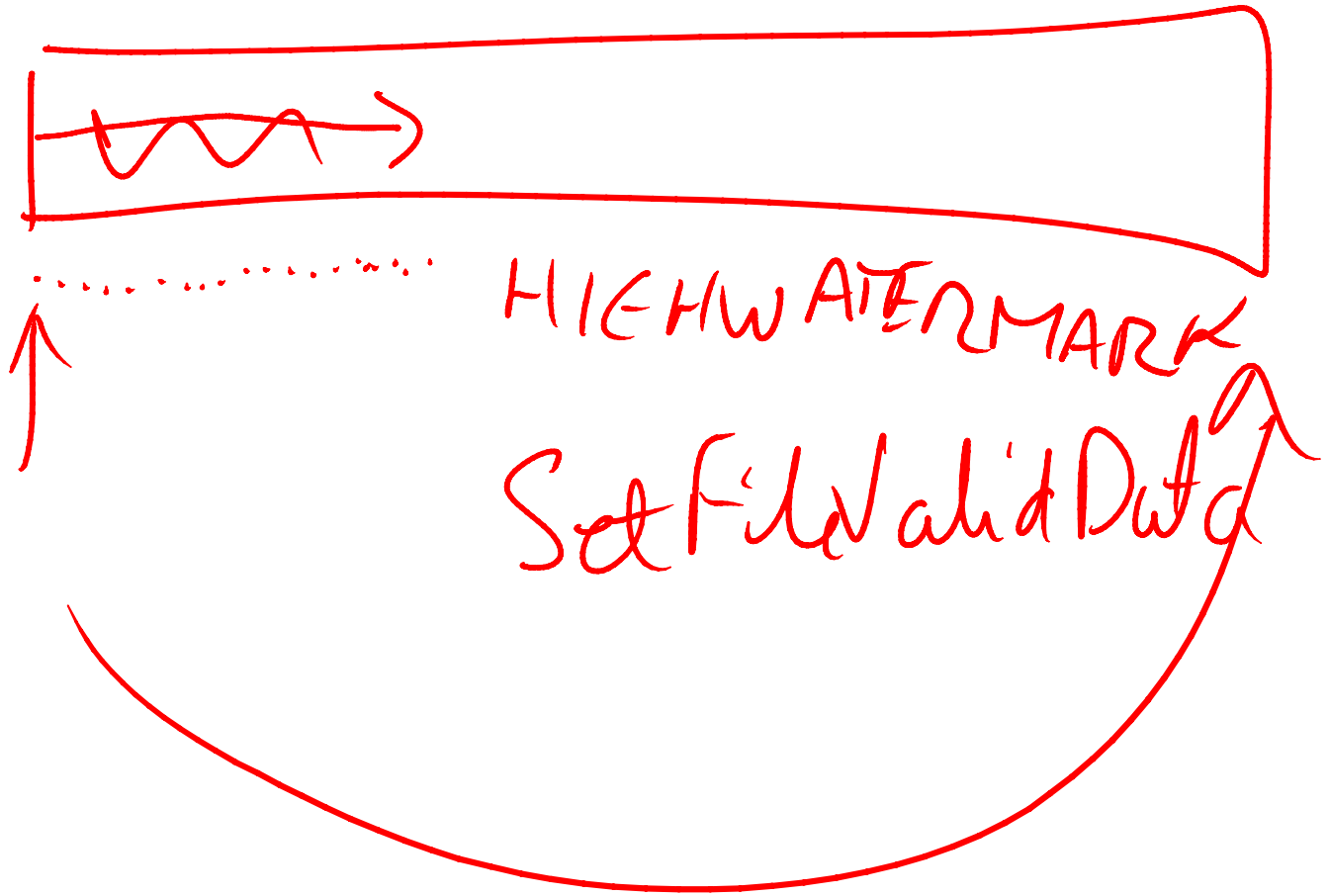
Jonathan's better formula for PLE is (bpool size / 4GB) * 300. Even better is to event on PLE dropping below your usual number.



See http://www.sqlskills.com/BLOGS/PAUL/post/Page-Life-Expectancy-isnt-what-you-think.aspx



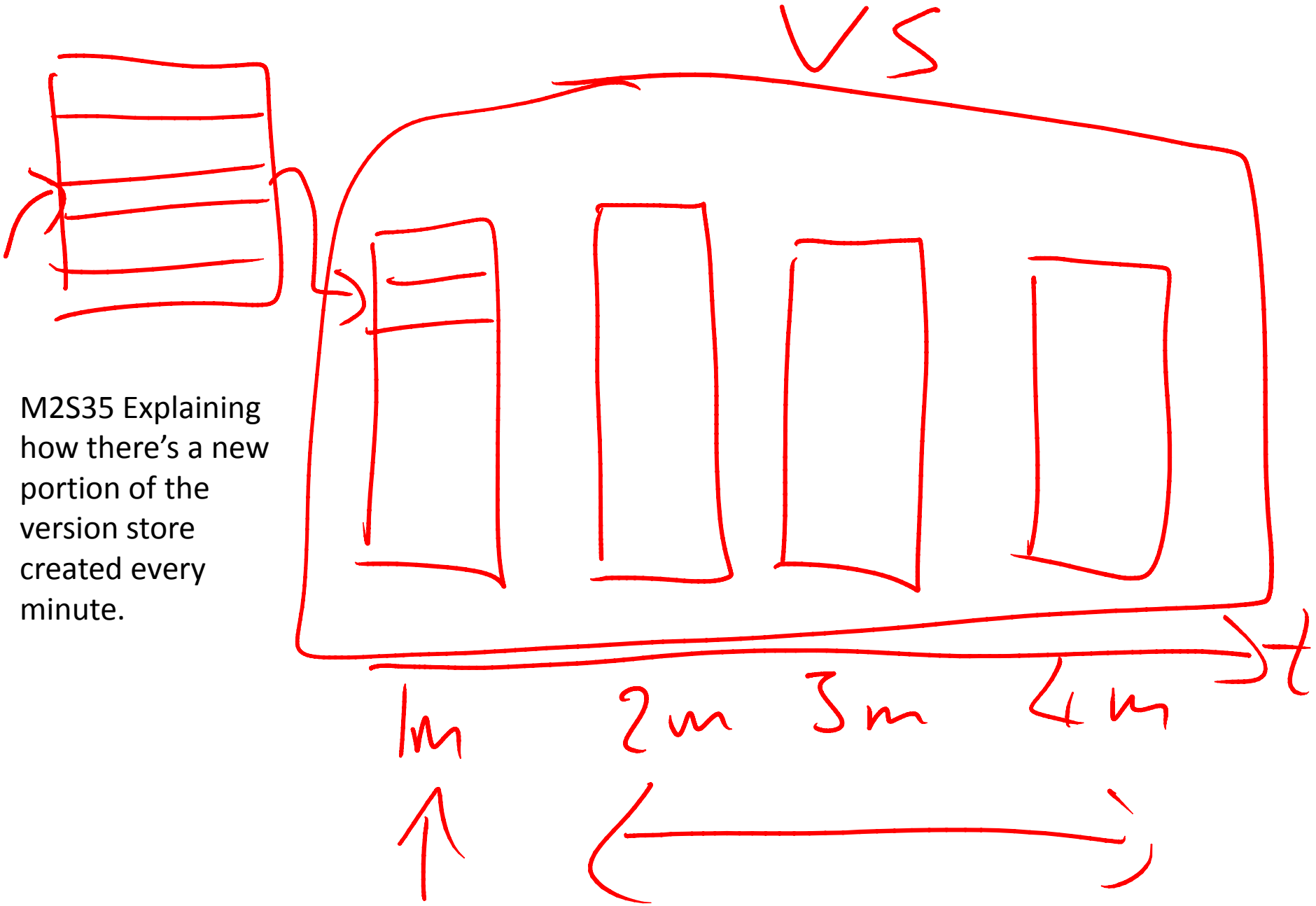
This should be Buffer Node instead of Buffer Partition – my mistake.



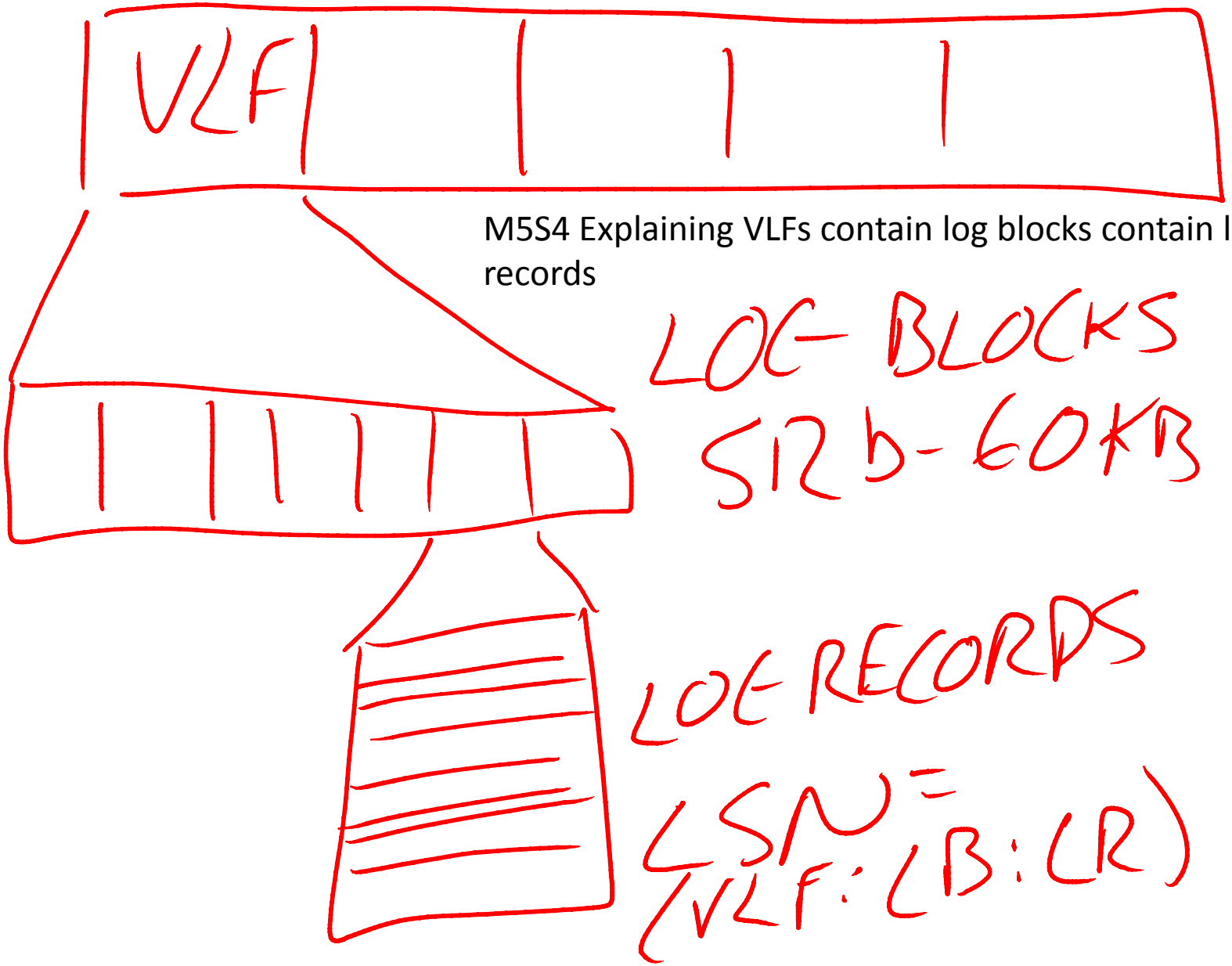
M2S22 Explaining zero initialization and how NTFS does not trust newly allocated space until the highwatermark of the file has increased, either through consecutive writes, or setting the HWM through a called to SetFileValidData.

M2S24 Explaining how the security flaw with instant initialization works. Secure files are deleted, SQL files expand and pick up the disk space without zeroing it out. SA using DBCC PAGE can look at the raw bytes of the disk space now included in the data file, potentially leading to information leakage.

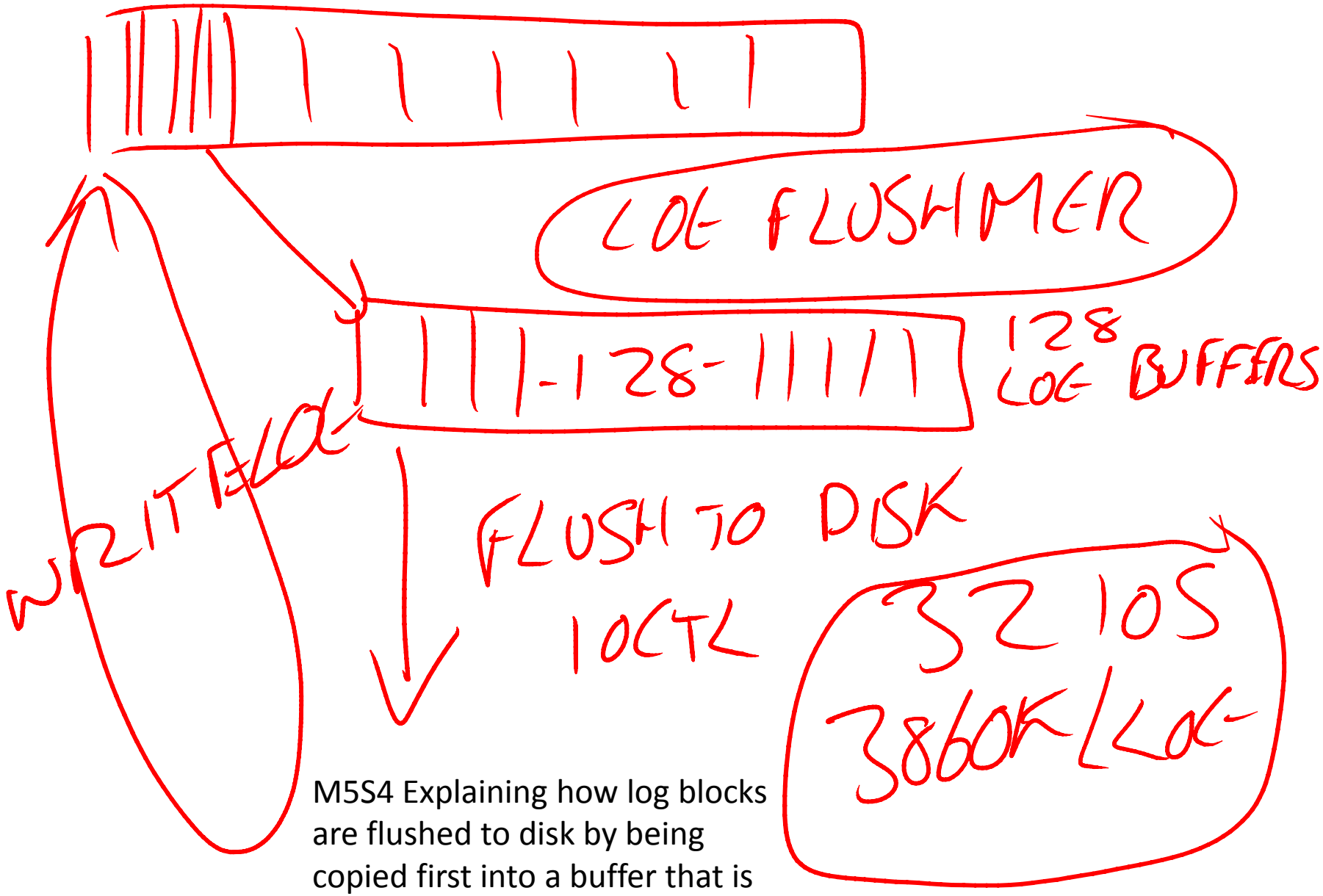
M2S35 Explaining how there's a new portion of the version store created every minute.



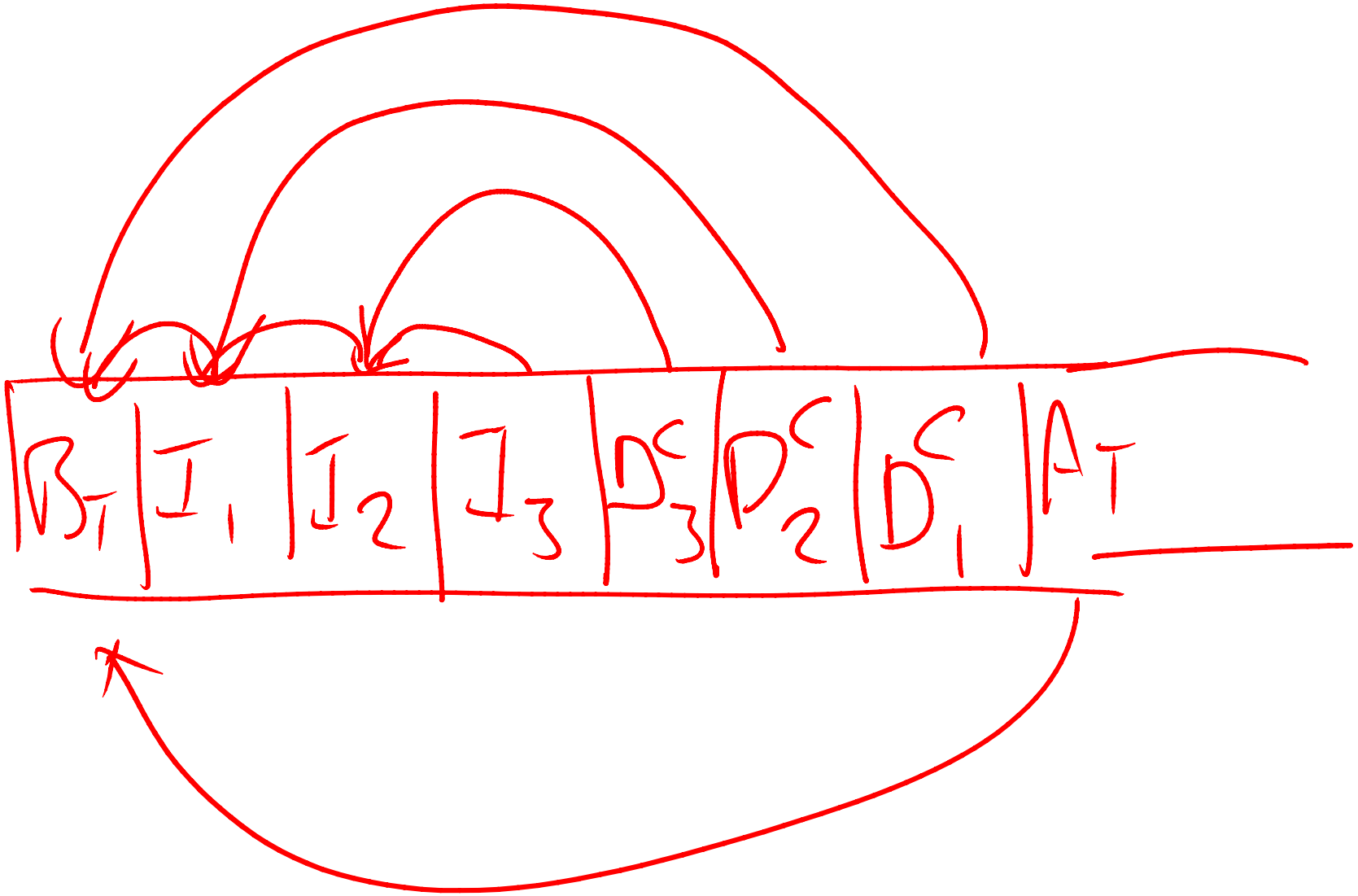
M2S40 Explaining PFS and SGAM contention in tempdb and how adding multiple files spreads the contention over multiple files, thus reducing it.



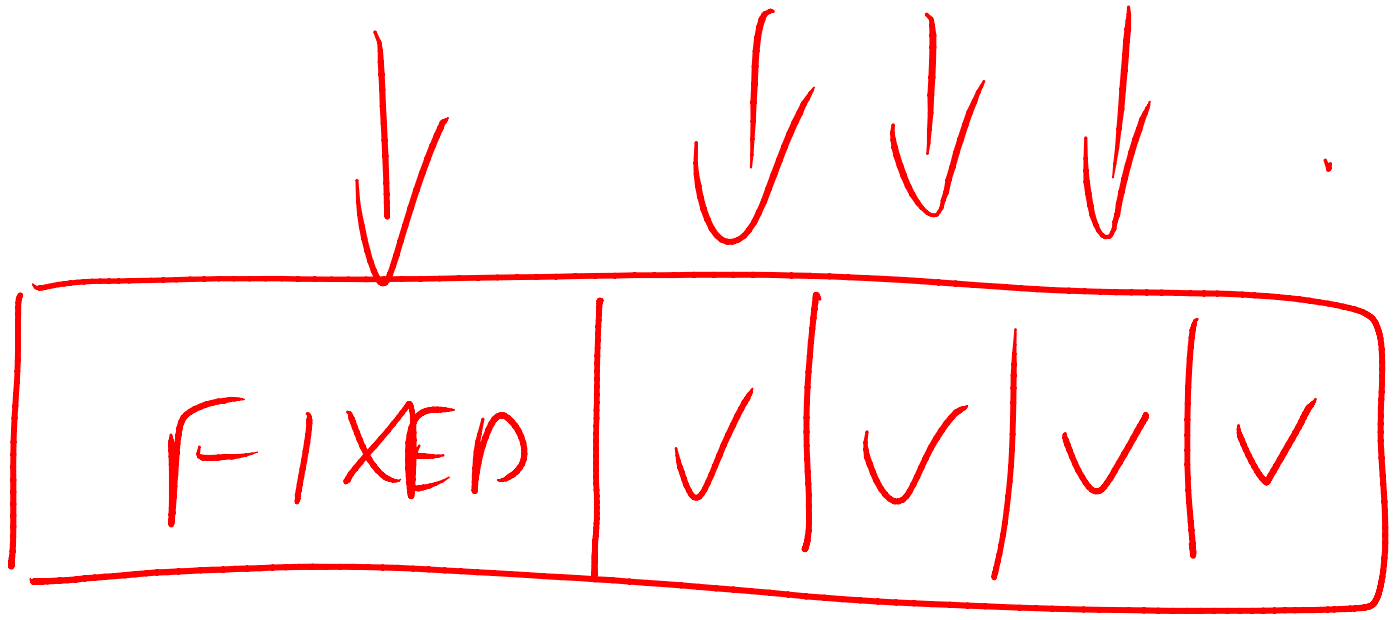
M5S4 Explaining VLFs contain log blocks contain log records



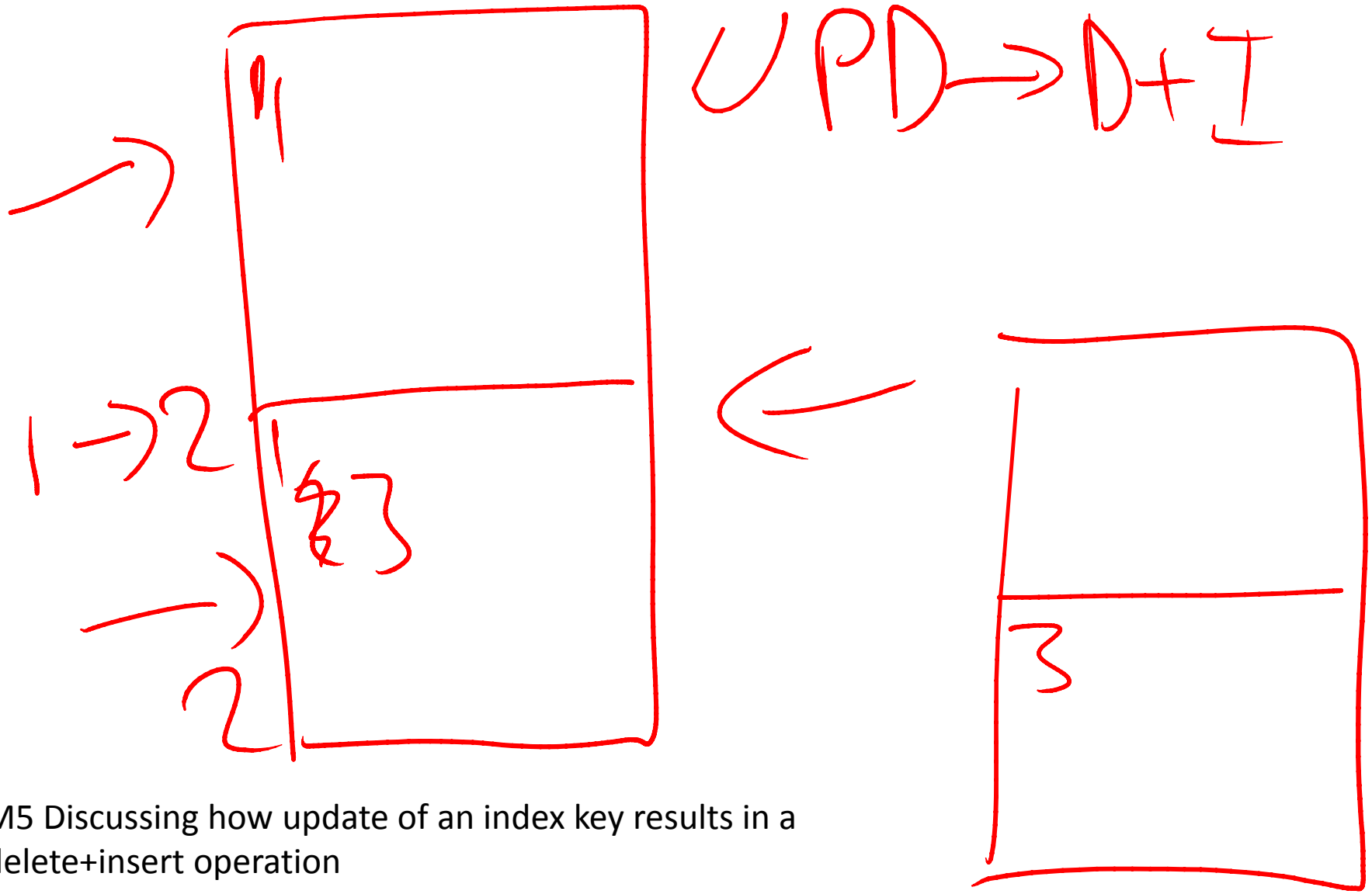
M5S4 Explaining how log blocks are flushed to disk by being copied first into a buffer that is then async written to disk.



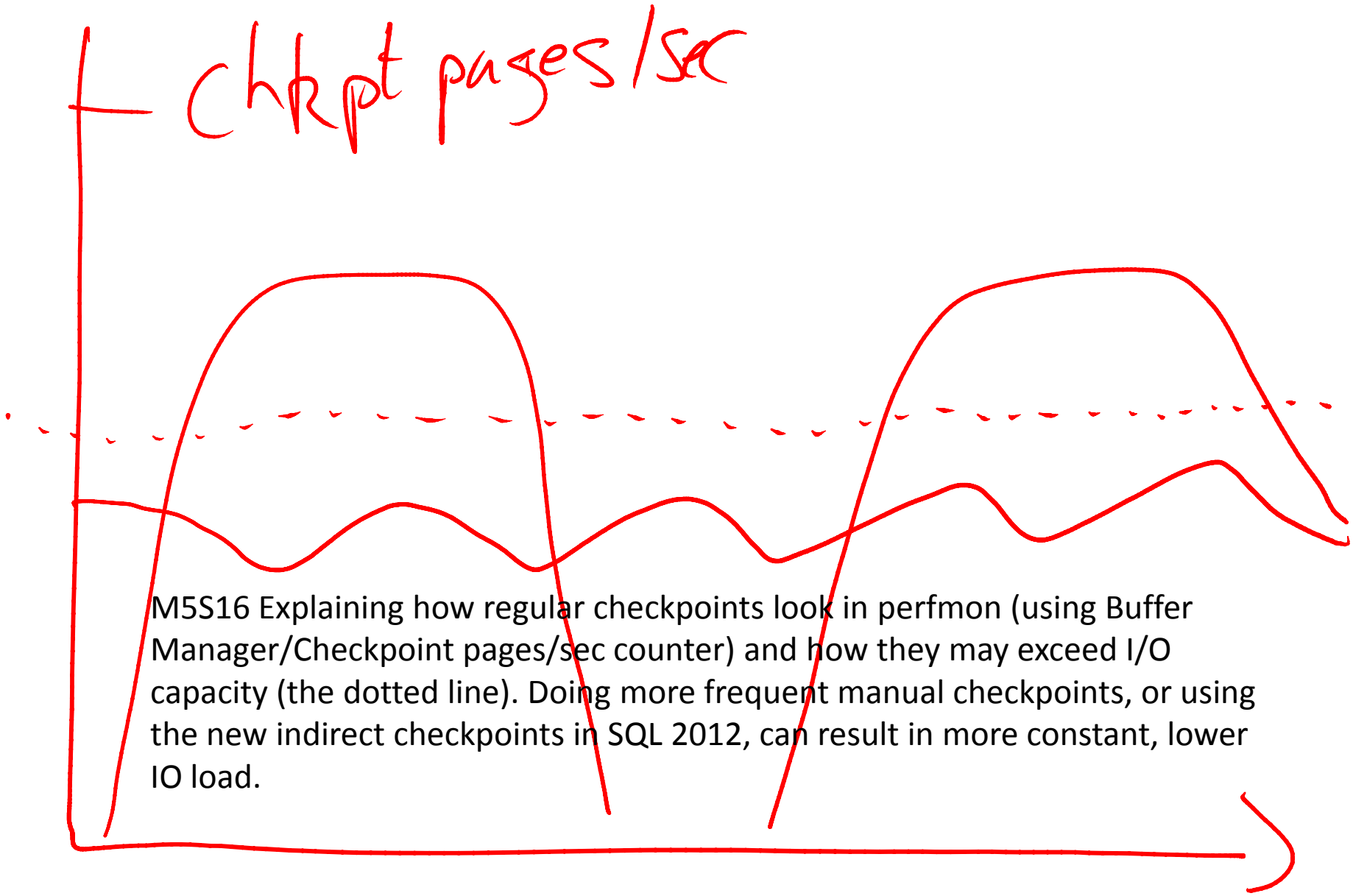
M5S11 Explaining how log records link back to the previous log record in the transaction to allow the transaction to be rolled back by generating compensating actions in reverse order. A compensation log record will link over the log record it is compensating for.



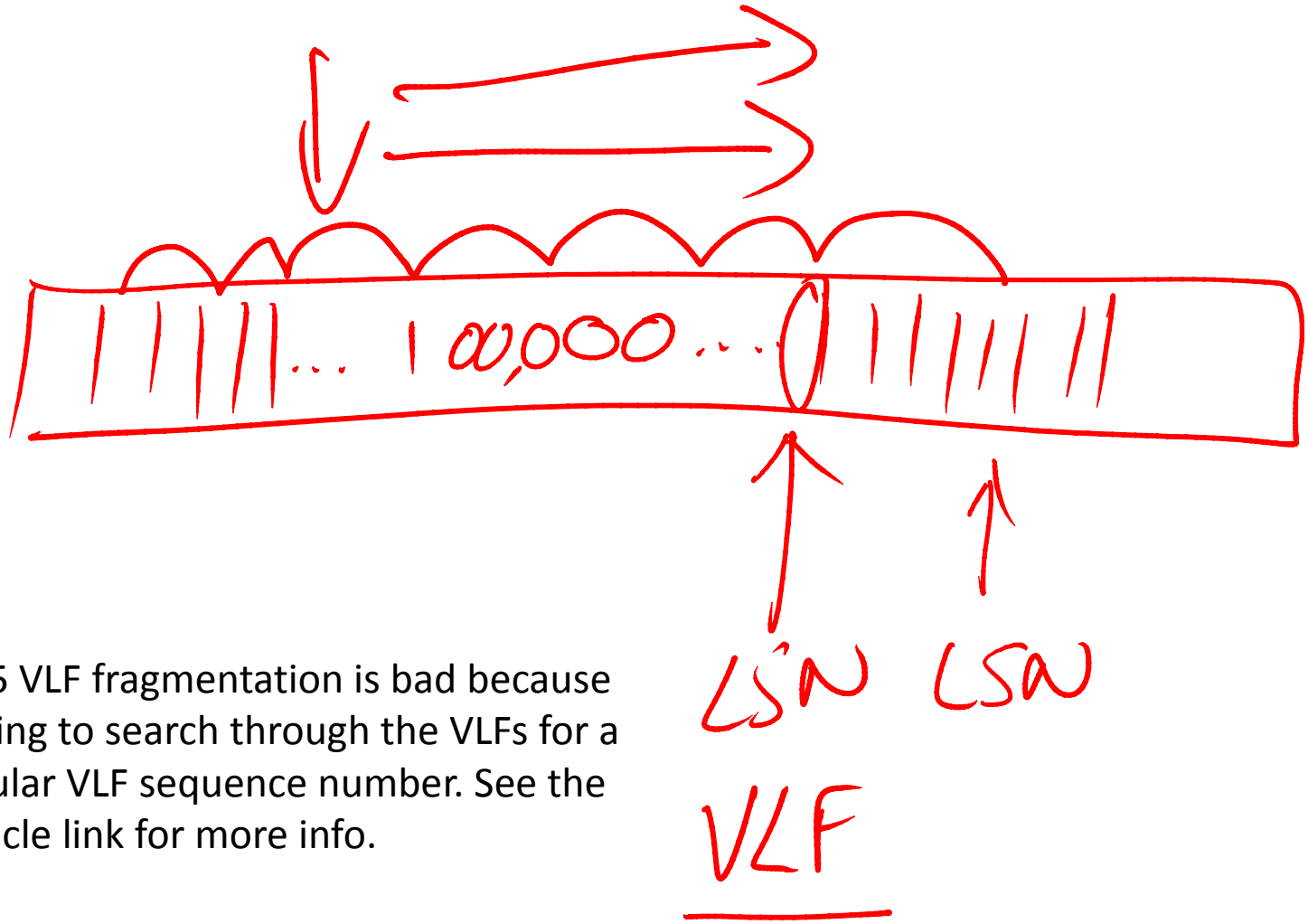
M5S11 Explaining how the Access Methods determines how to log UPDATEs efficiently by splitting the logging by portion of the record. Each variable length column is a portion, as is the fixed-width portion (up to 16 bytes in each log record)



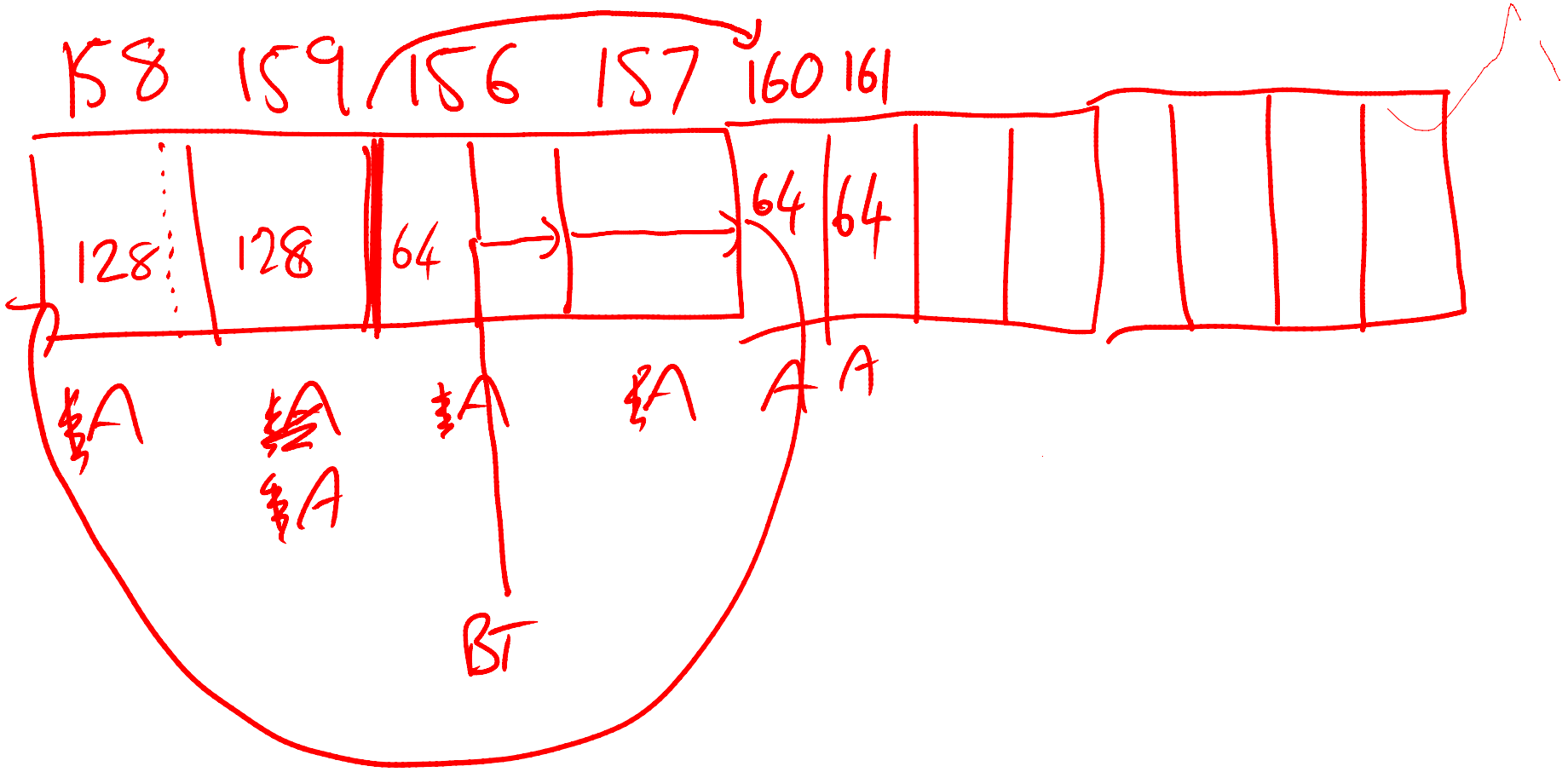
M5 Discussing how update of an index key results in a delete+insert operation



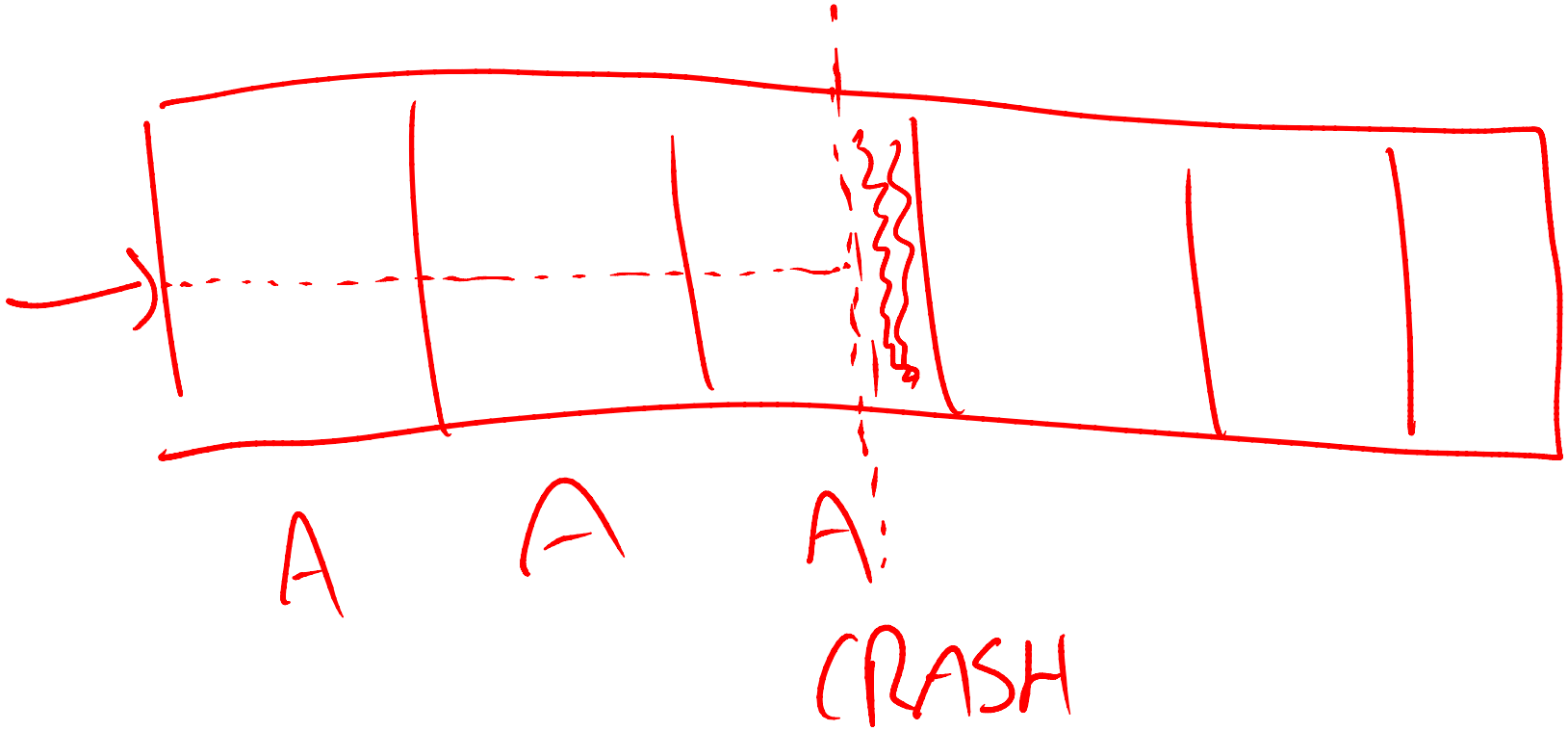
M5S16 Explaining how regular checkpoints look in perfmon (using Buffer Manager/Checkpoint pages/sec counter) and how they may exceed I/O capacity (the dotted line). Doing more frequent manual checkpoints, or using the new indirect checkpoints in SQL 2012, can result in more constant, lower IO load.



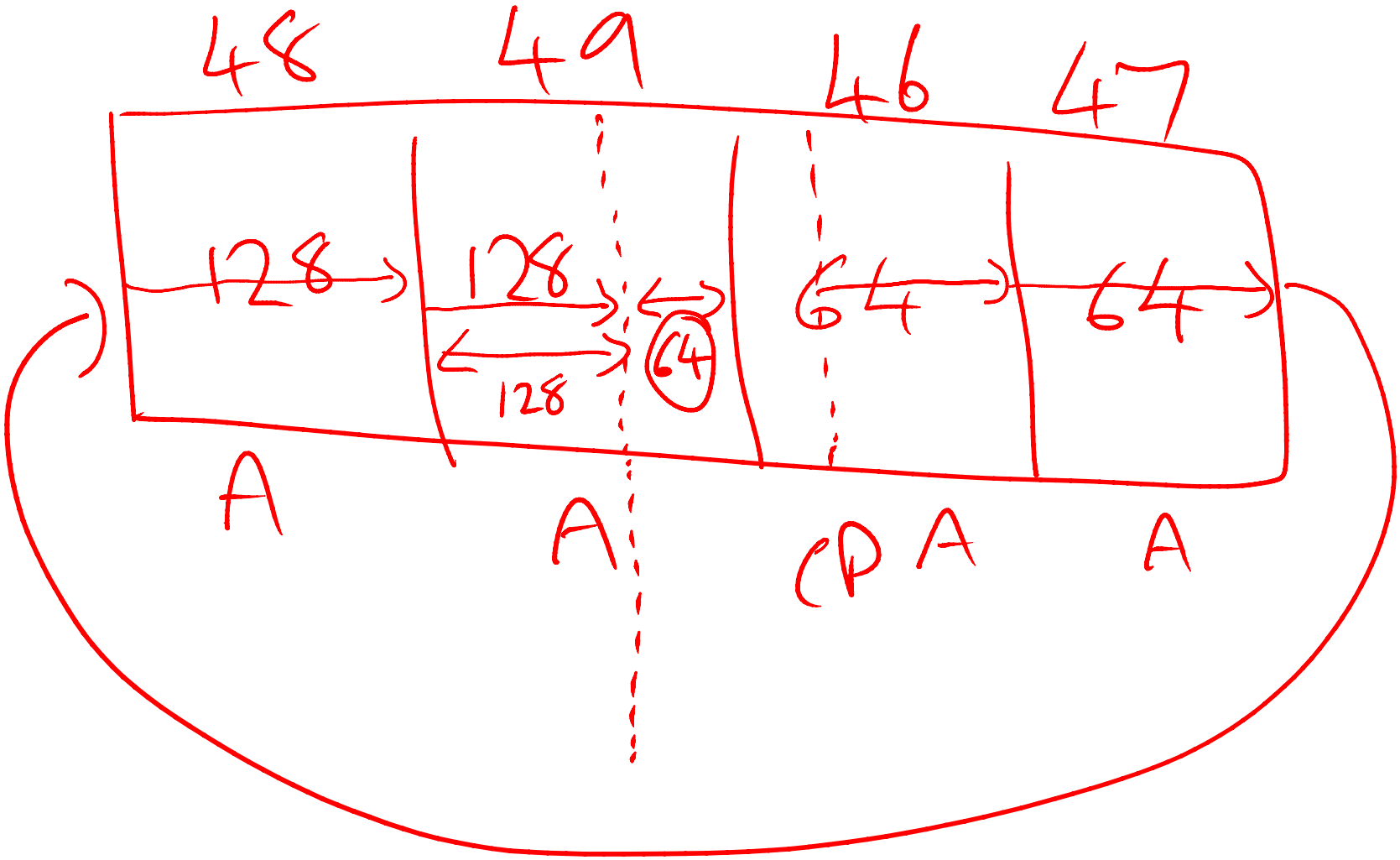
M5S25 VLF fragmentation is bad because of having to search through the VLFs for a particular VLF sequence number. See the KB article link for more info.



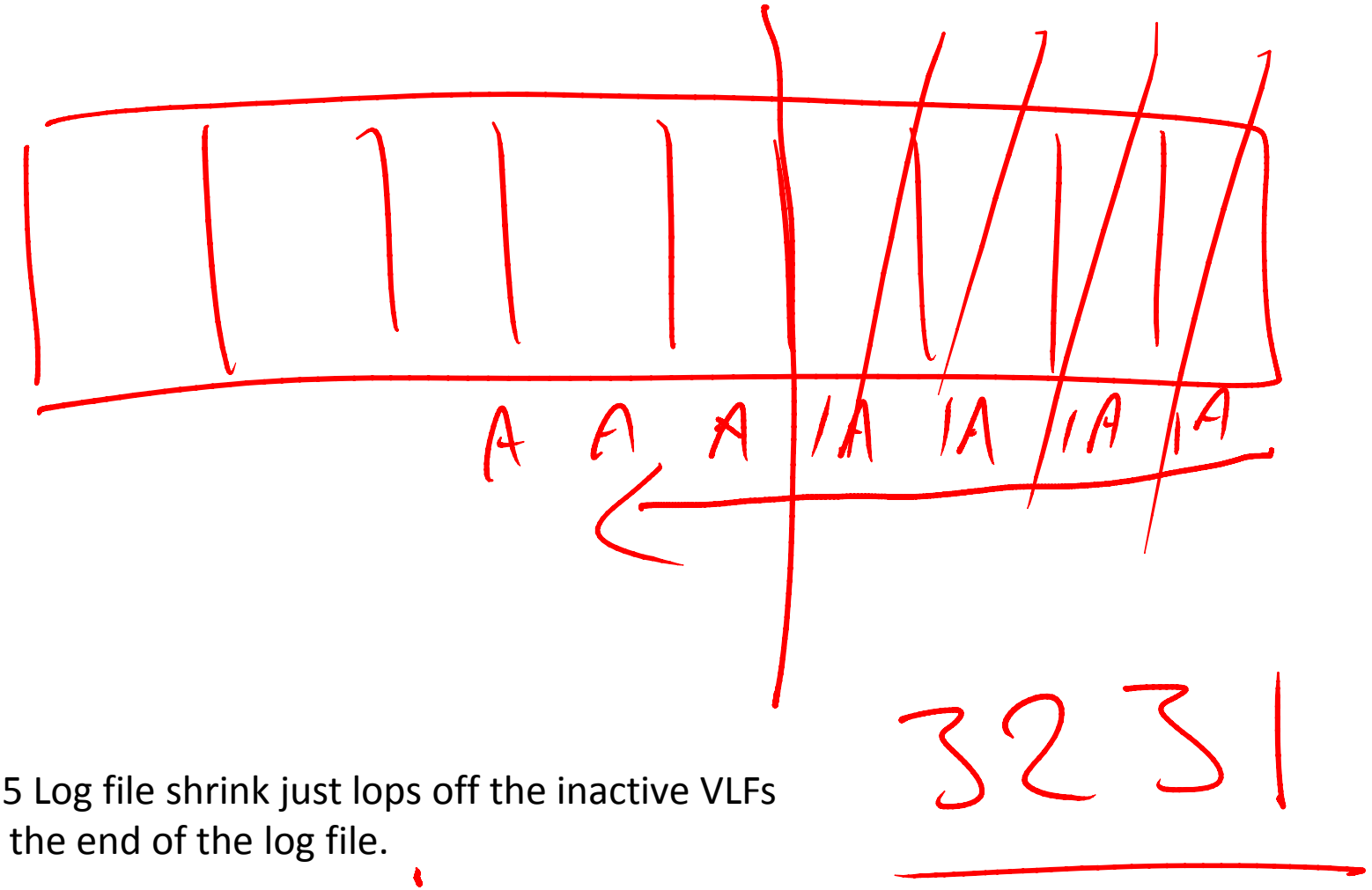
M5S28 Explaining the circular log demo. We discussed log space reservation that happens to ensure the active transactions can roll back without having to grow the log – this accounts for extra log growths. Space reserved in the log does not make a VLF become active – as there are no log records written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



M5S29 Explaining how the log must be zero initialized when first allocated so that crash recovery can figure out where to stop if the log has never wrapped around.

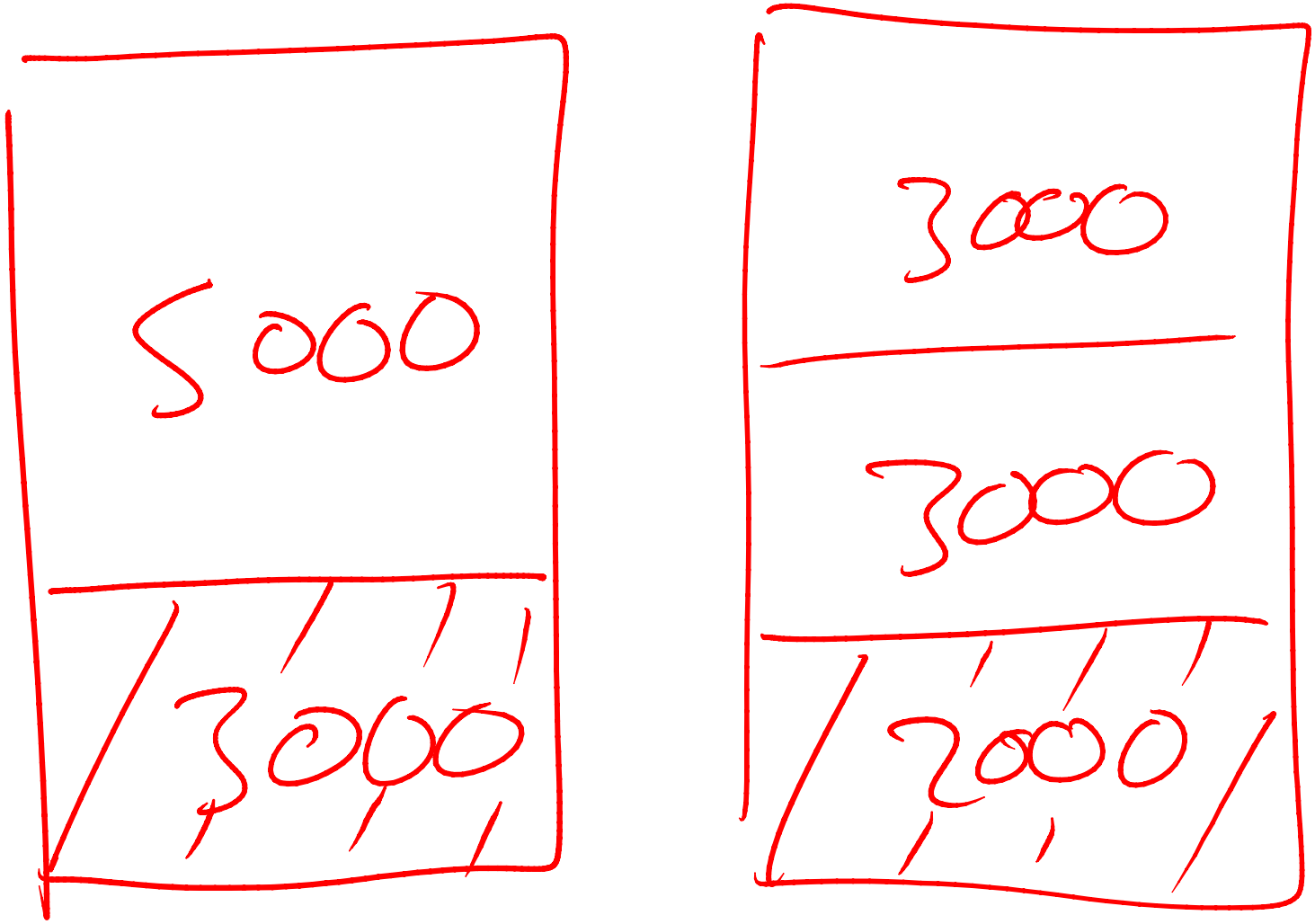


M5S29 Explaining the use of parity bits to help SQL Server determine where the end of the log is during crash recovery.

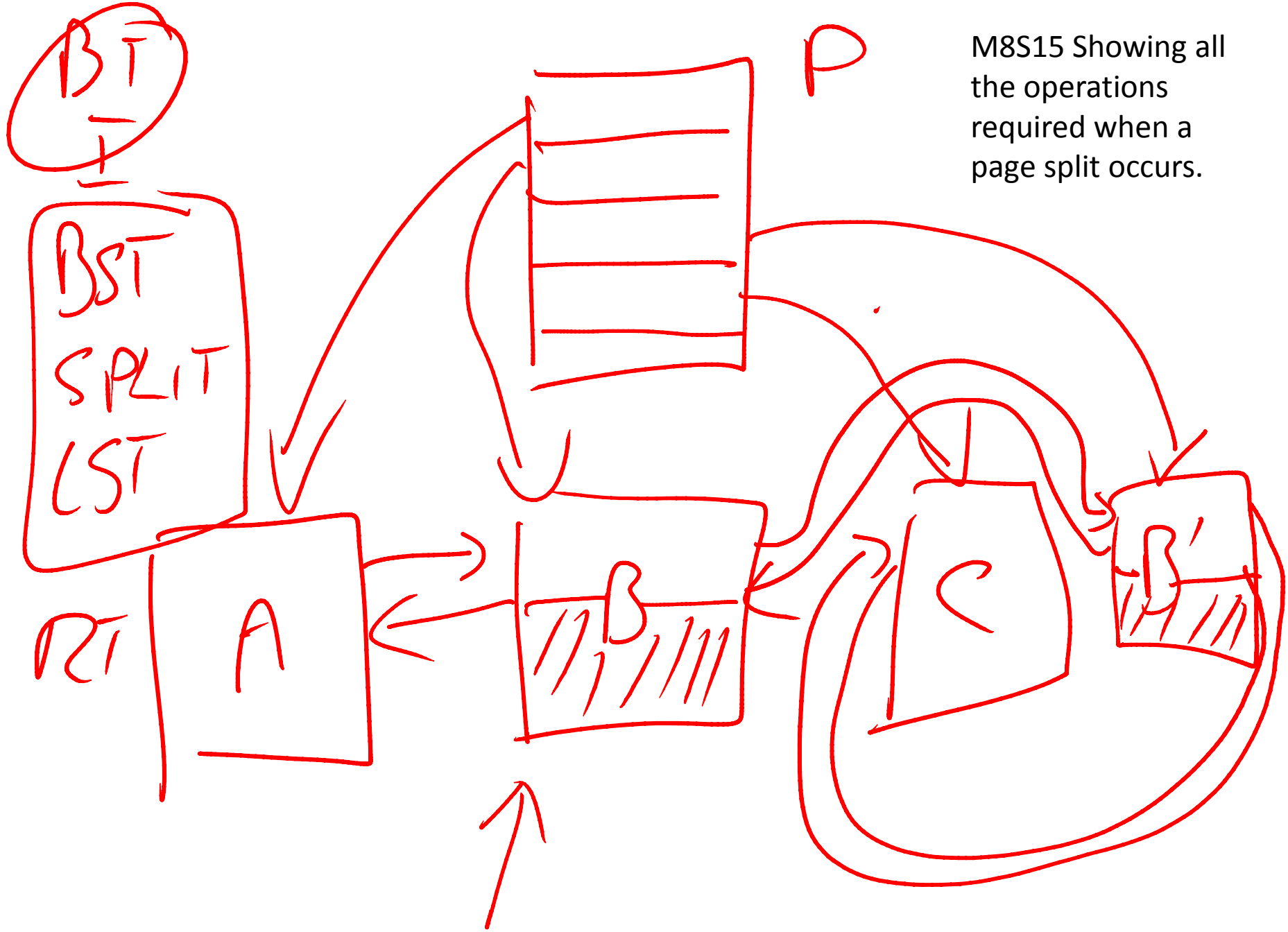


M5 Log file shrink just lops off the inactive VLFs at the end of the log file.

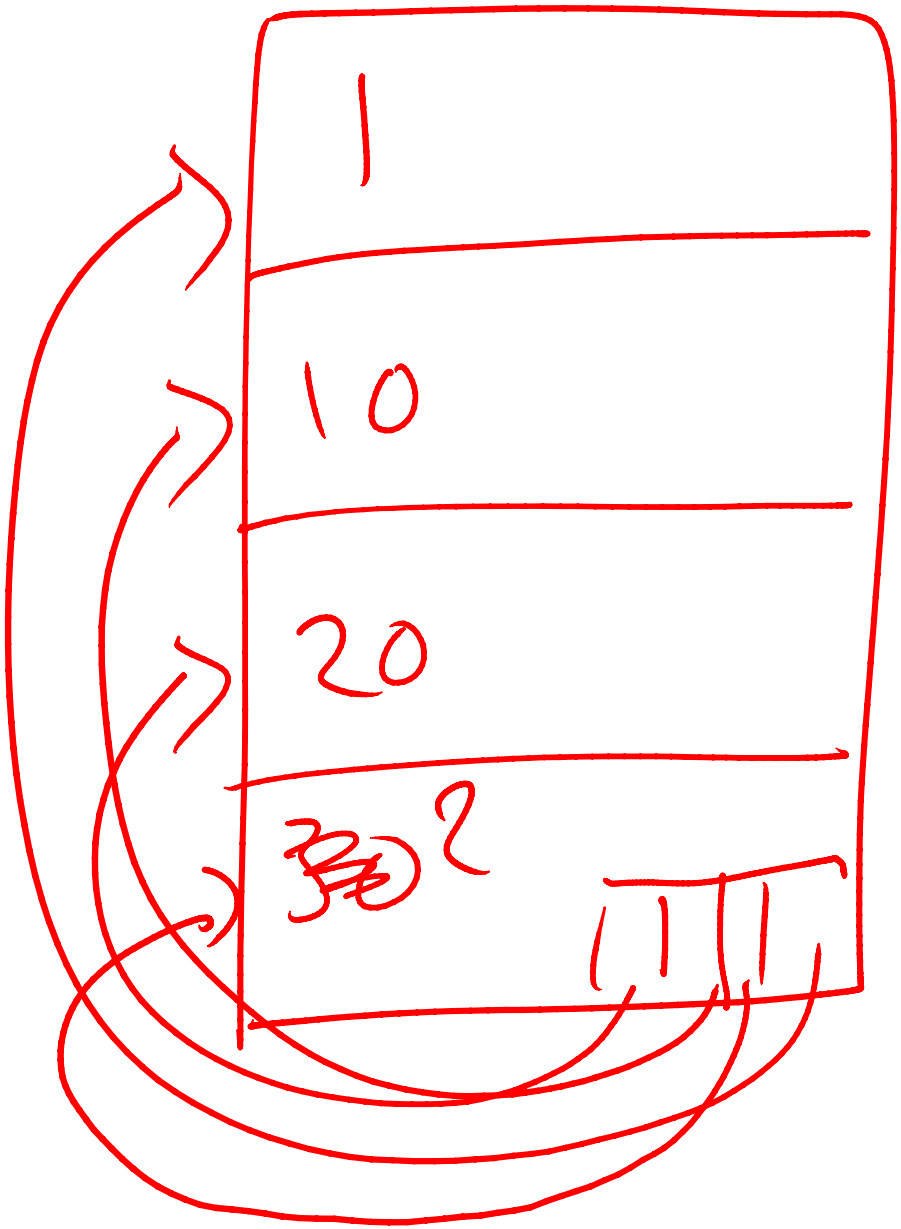
BACKUP WITH NO_LOG can be turned into an operation that does nothing using TF3231 in SQL 2005 and before.



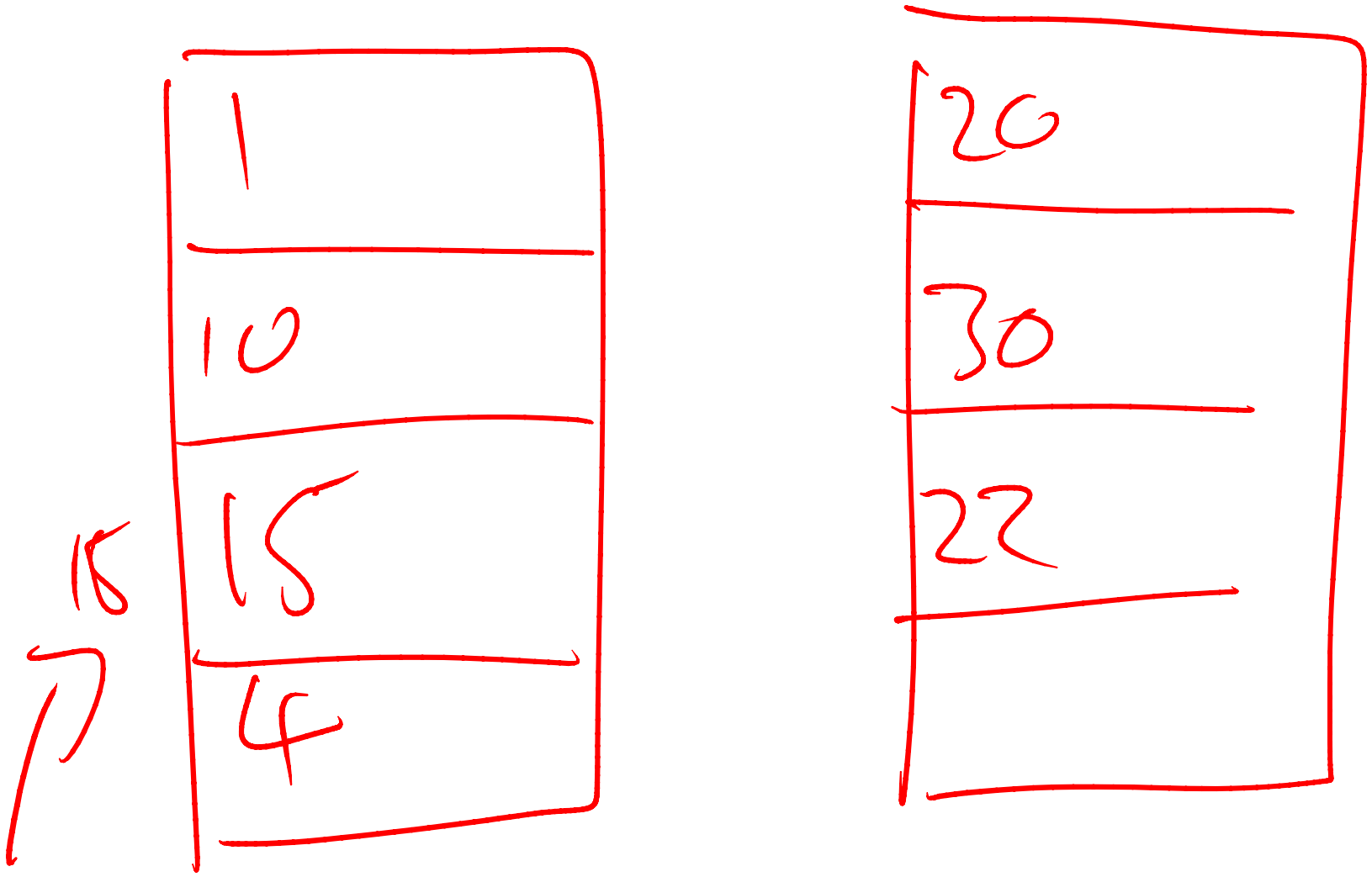
M8S13 Explaining how fixed-size large rows leads to low page density and wasted space.



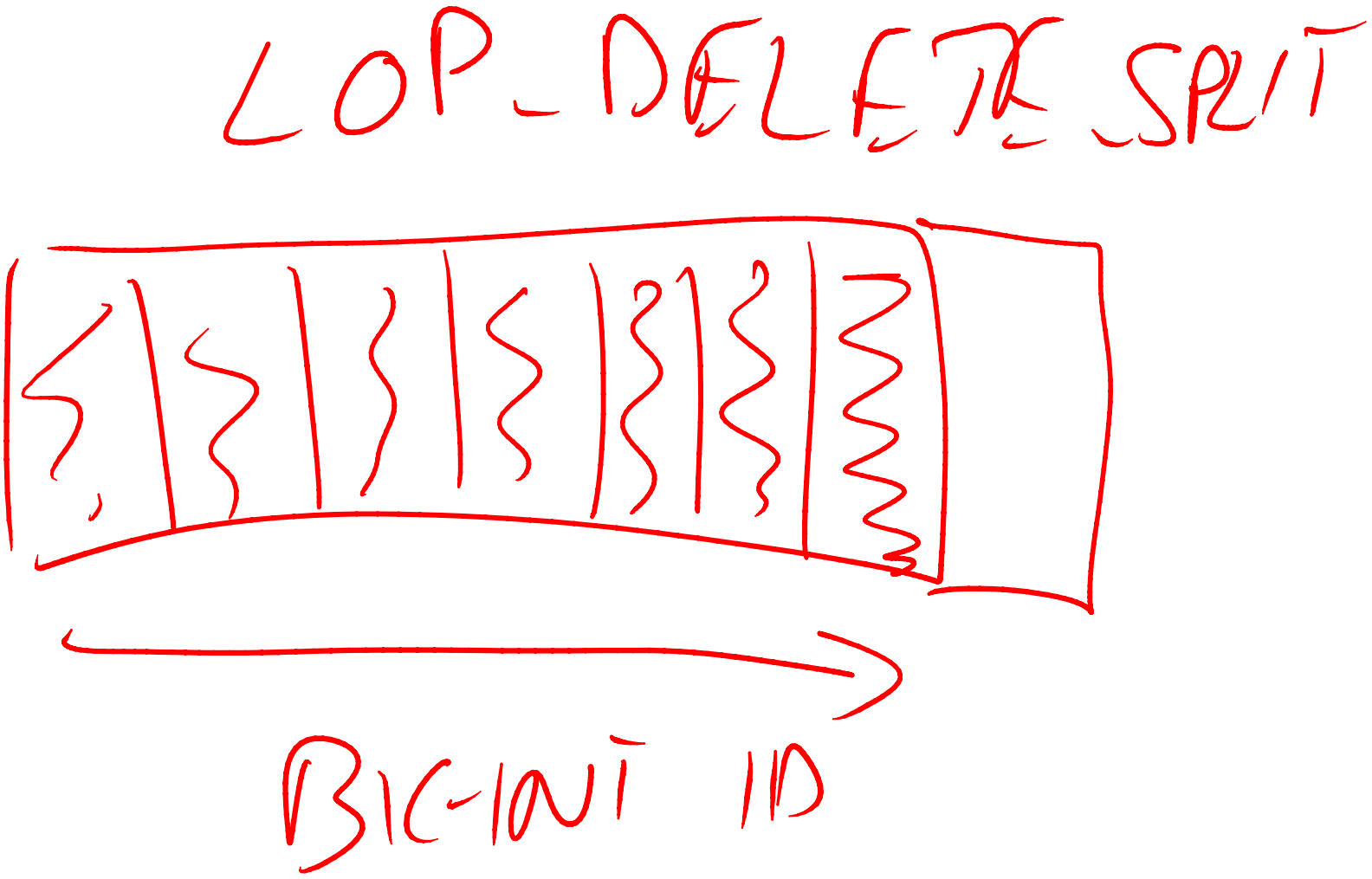
M8S15 Showing all the operations required when a page split occurs.



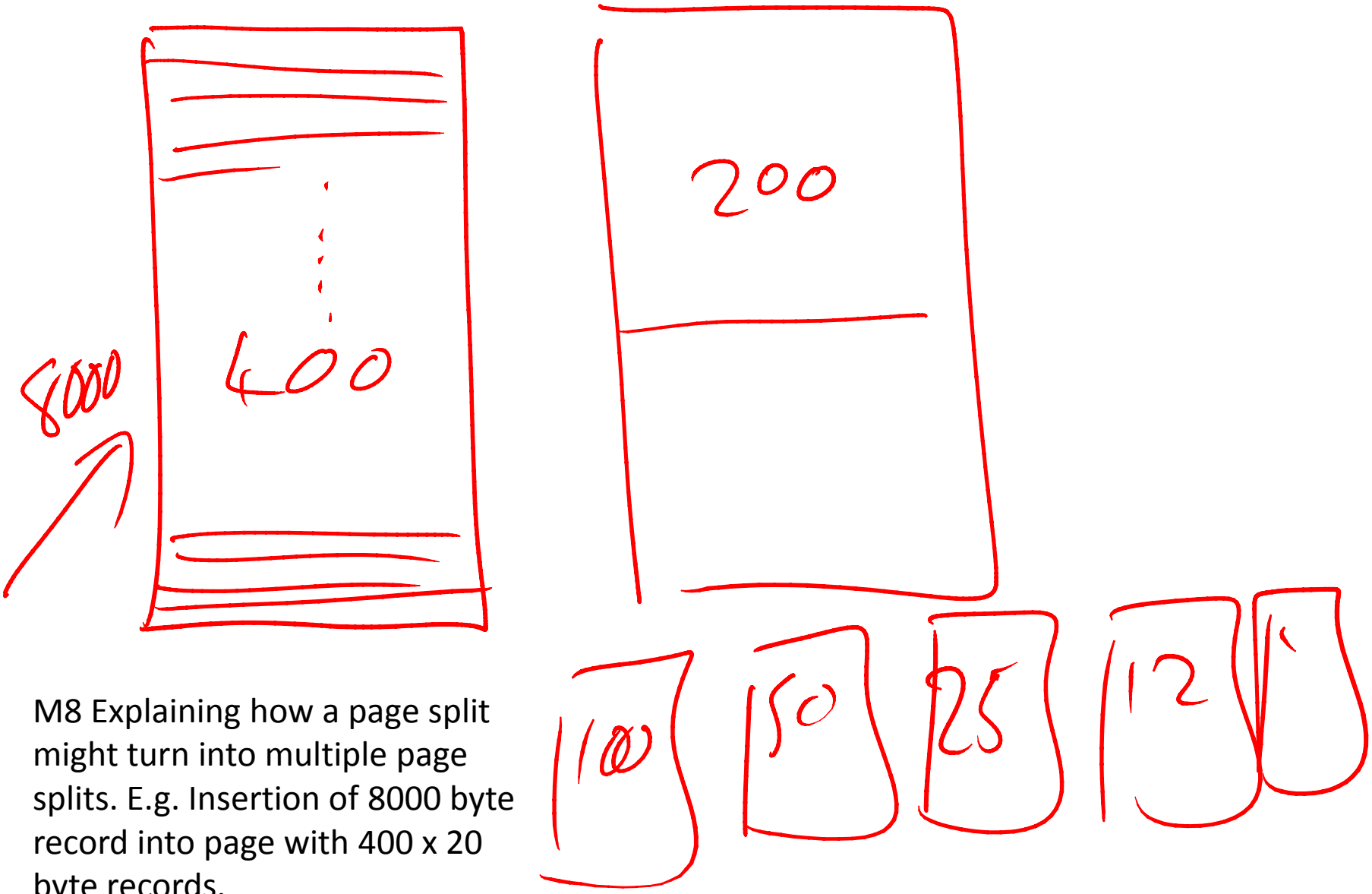
M8 The slot array can become ‘unordered’ if records with keys 1, 10, 20, 30 are entered. If 30 is deleted and then 2 inserted, 2 is the last physical record on the page, but the second logical record on the page (and the second entry in the slot array).



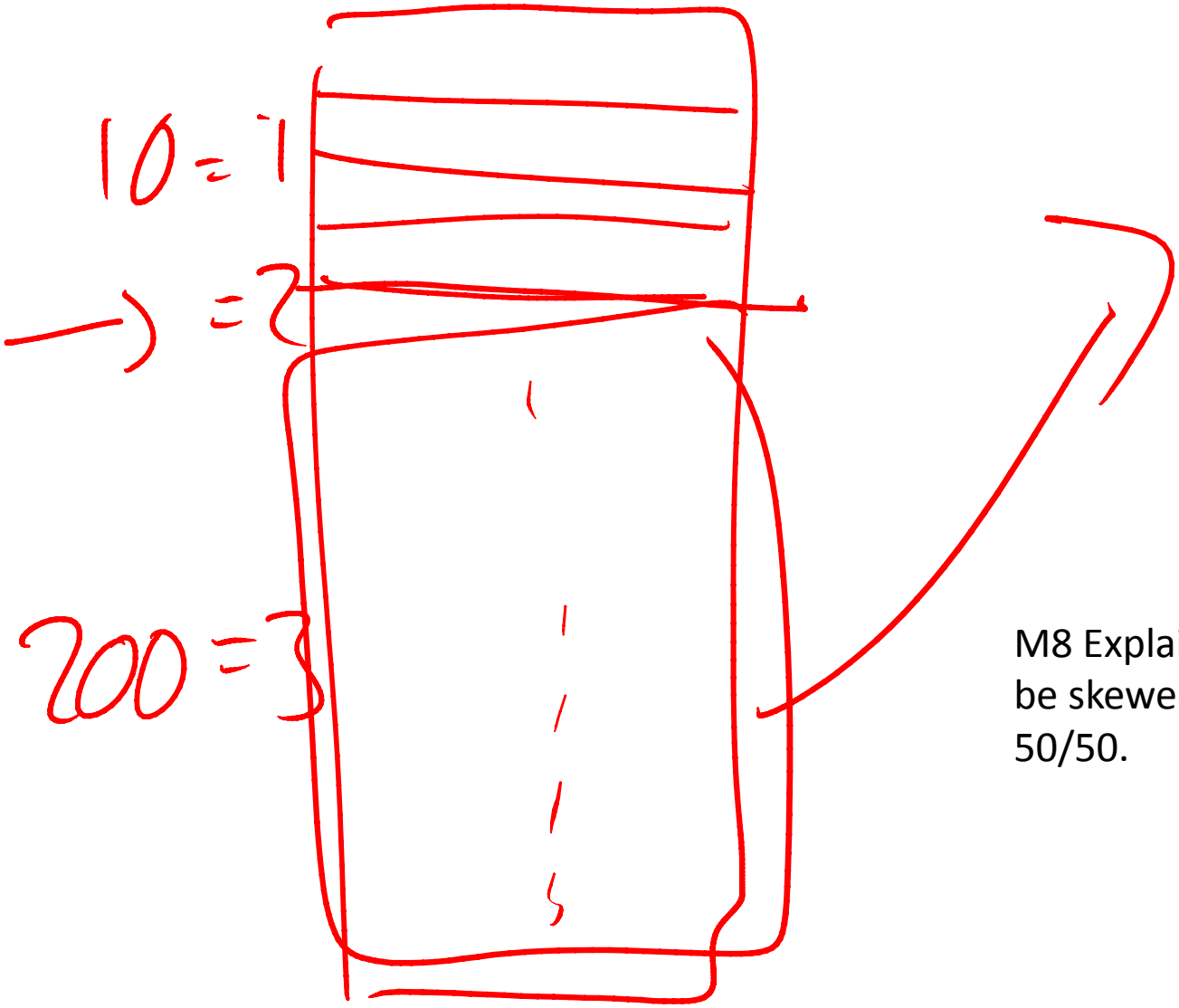
M8 After a page split, where 1, 10, 20, 30 was split into 1, 10 and 20, 30; subsequent inserts into either page may result in ‘unordered’ slot array entries.



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.



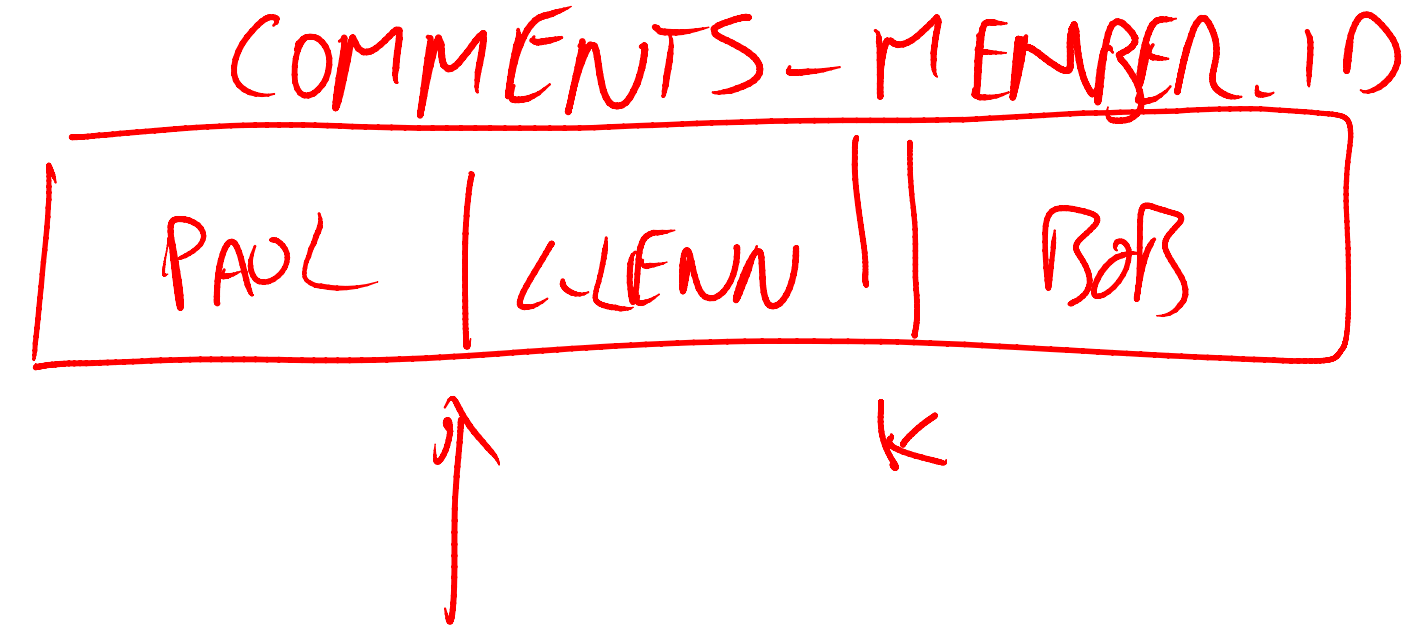
M8 Explaining how a page split might turn into multiple page splits. E.g. Insertion of 8000 byte record into page with 400 x 20 byte records.



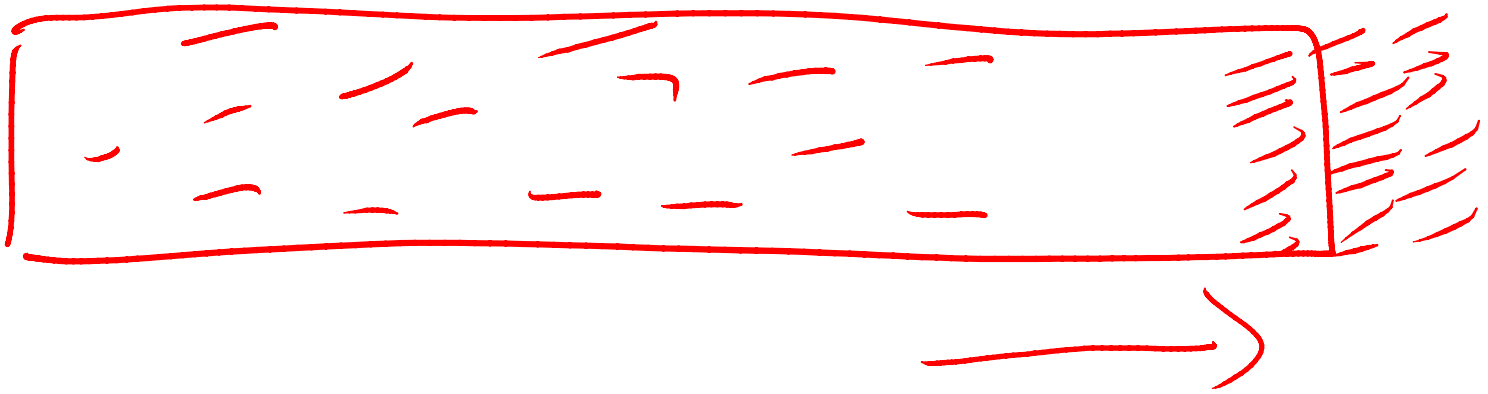
M8 Explaining that page splits can be skewed sometimes and not 50/50.

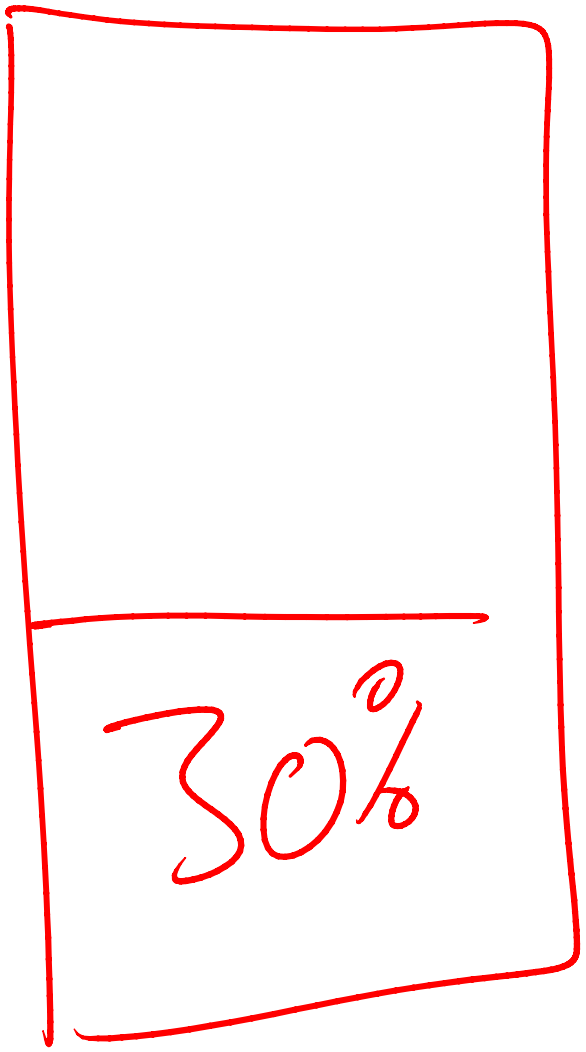
SIZE	NO SPLIT	SPLIT	X
1000	1244	6772	x5
100	344	5964	x18
10	256	10364	x45

M8S22 The numbers from the page split logging cost demo.



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





FF = 70

M8S25 To leave 30% free space, set the fillfactor to 70.