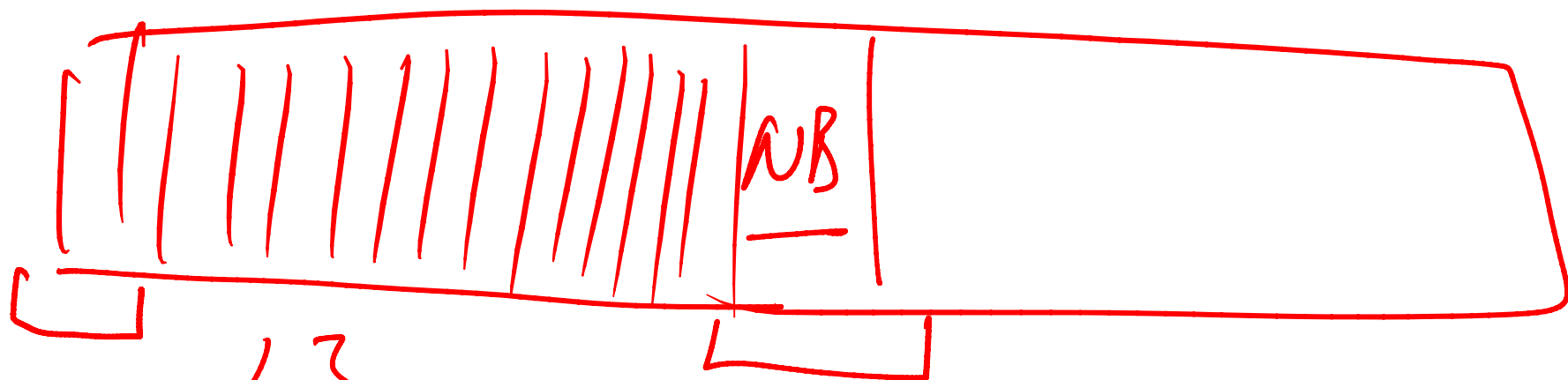




L3

CACHE LINES

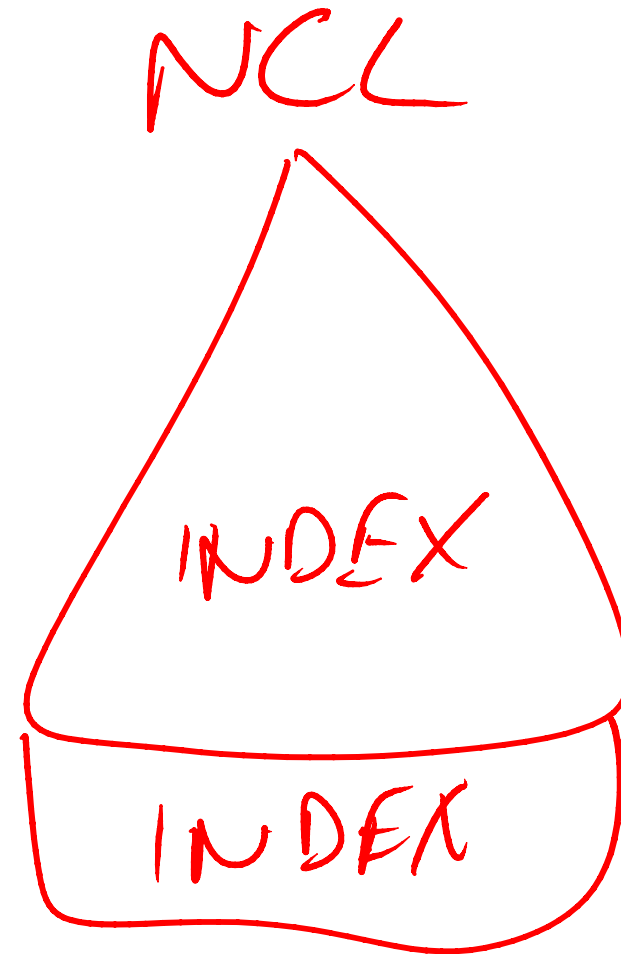
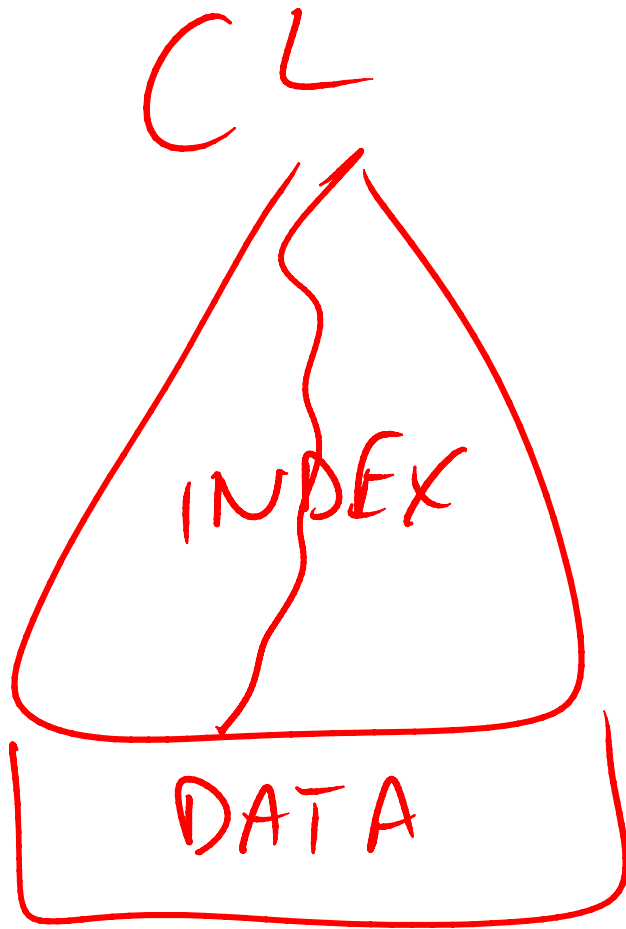
64b
128b

INVALIDATION

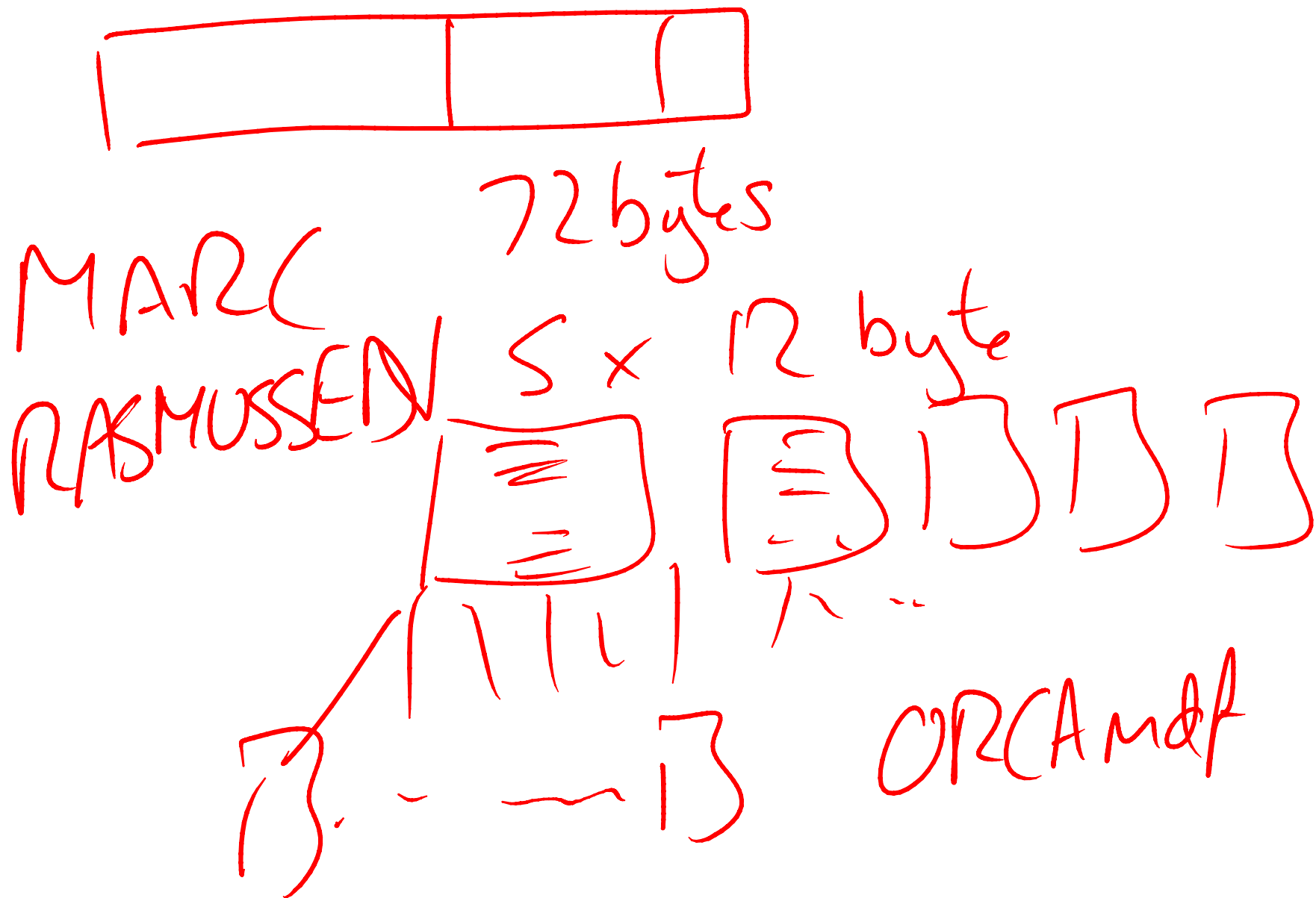
M1S06 Discussing
why the NULL
bitmap is a perf
optimization by
avoiding having to
read record parts
into the CPU's L3
cache

RM 1 LOP_COUNT_DELTA
2
~~3~~

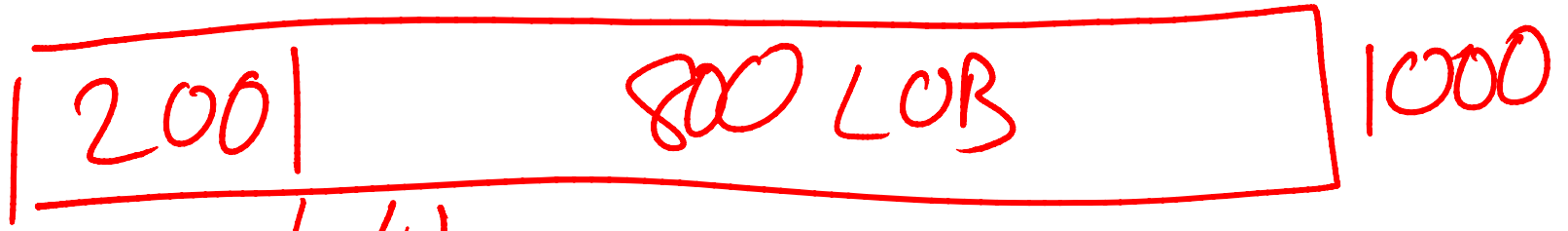
SEM 1 CL NCL
2 2
3D RED ~~3D~~ RED-~~3D~~



M1S8 Index rows in the upper levels of all indexes. Data records at the leaf of a clustered index, index records at the leaf of a nonclustered index. Index records in the nonleaf are used to navigate through the index. See Kimberly's later modules for details.

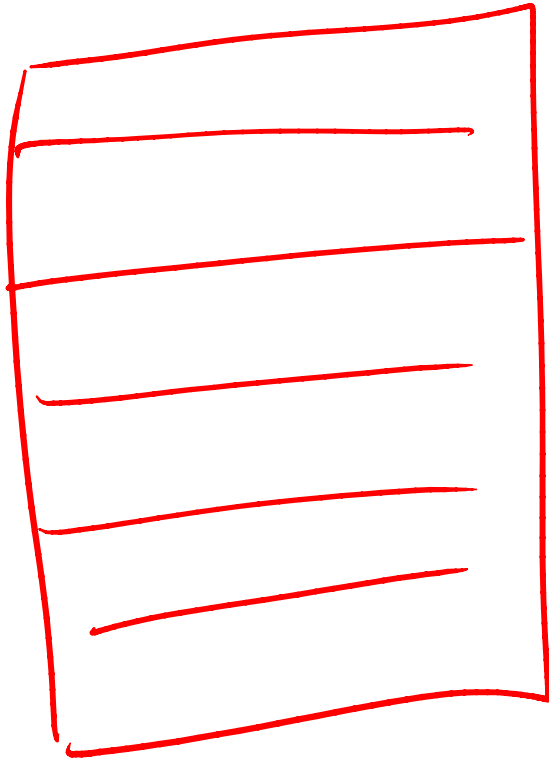


Google Marc Rasmussen for his blog. He's blogged text tree structures.

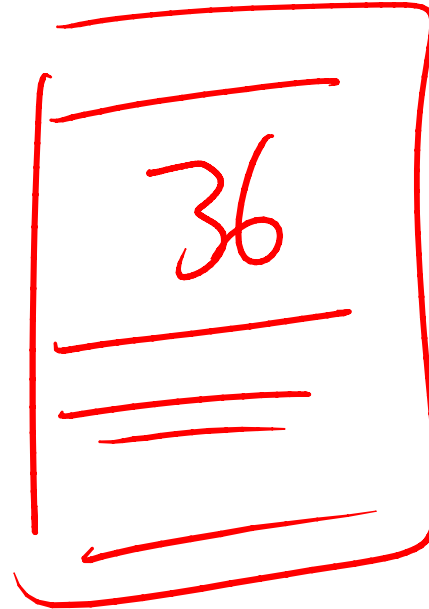


(24)

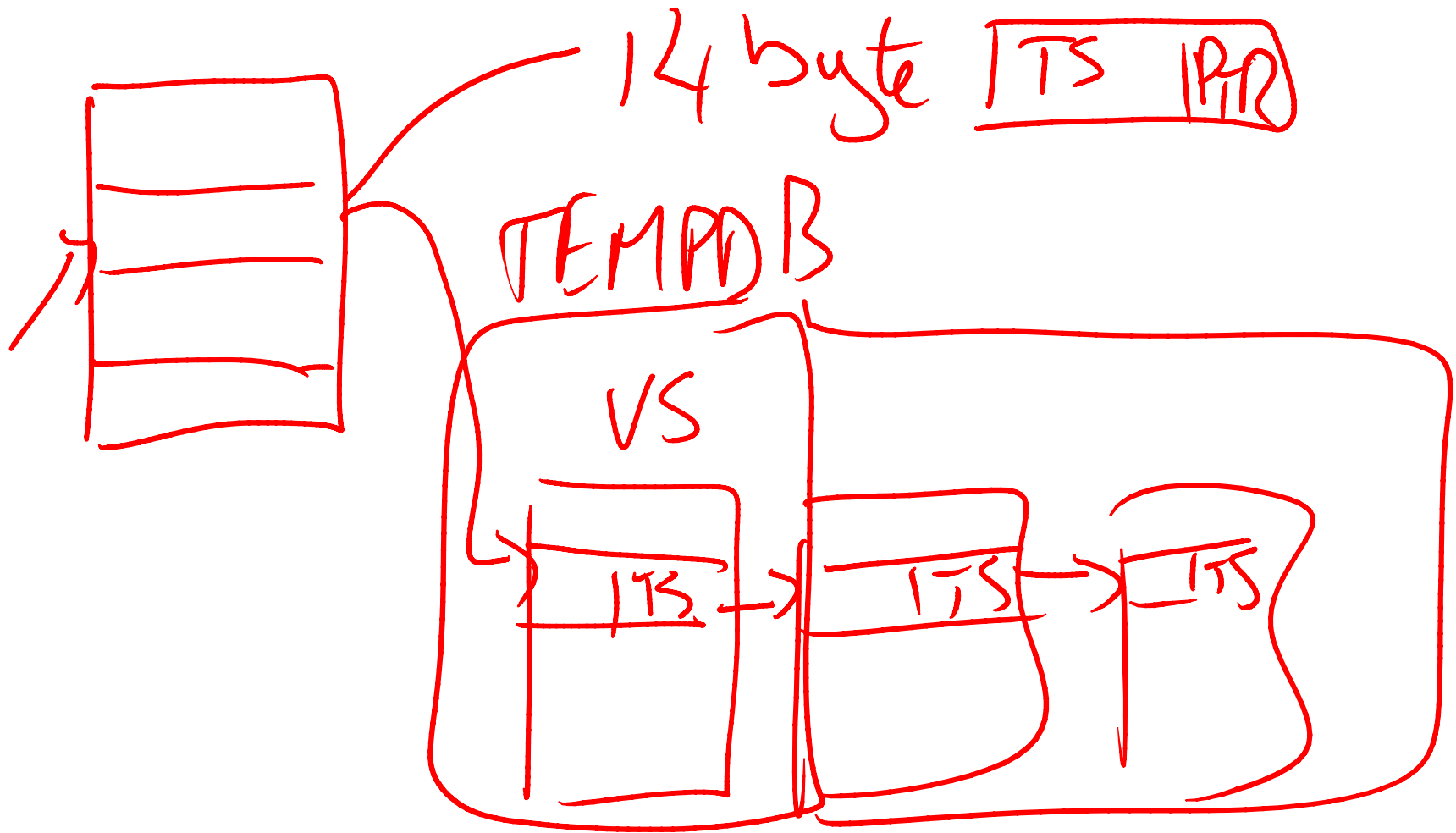
OFF-ROW LOB
ANOTHER LOB



8

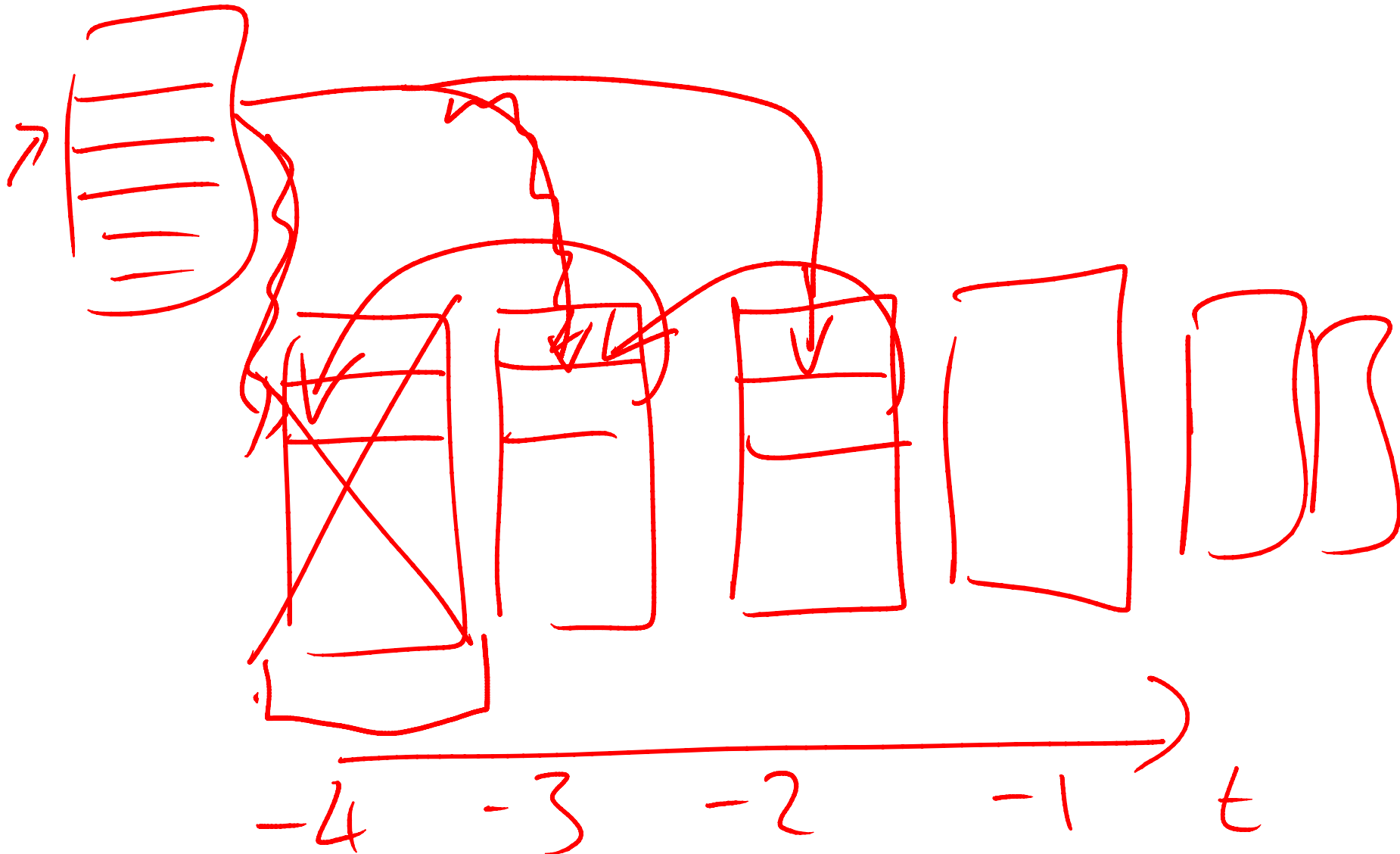


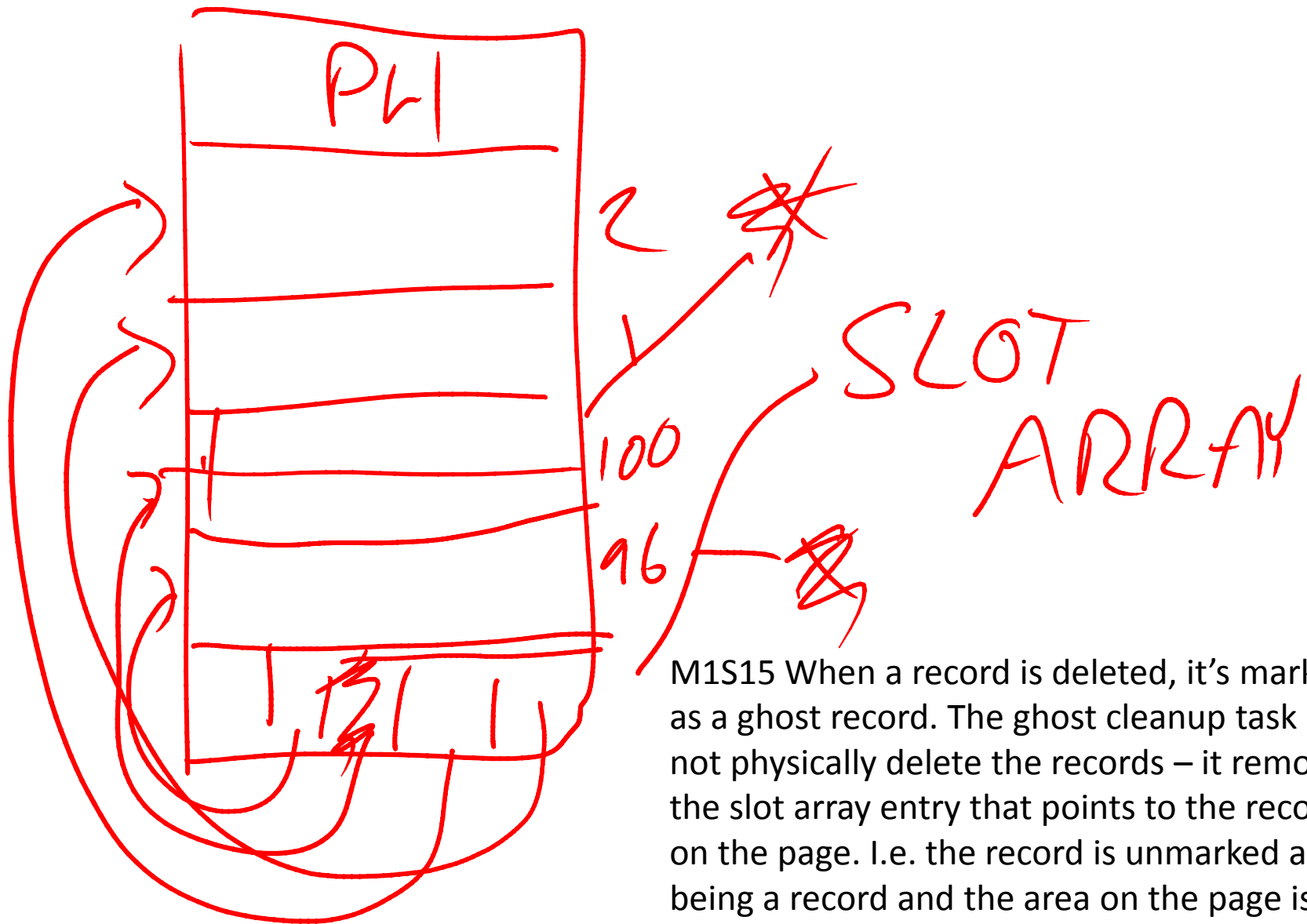
M1S12 Having a large LOB value in row that's not used much makes for large rows and low data density. Choose how to store LOB data wisely.



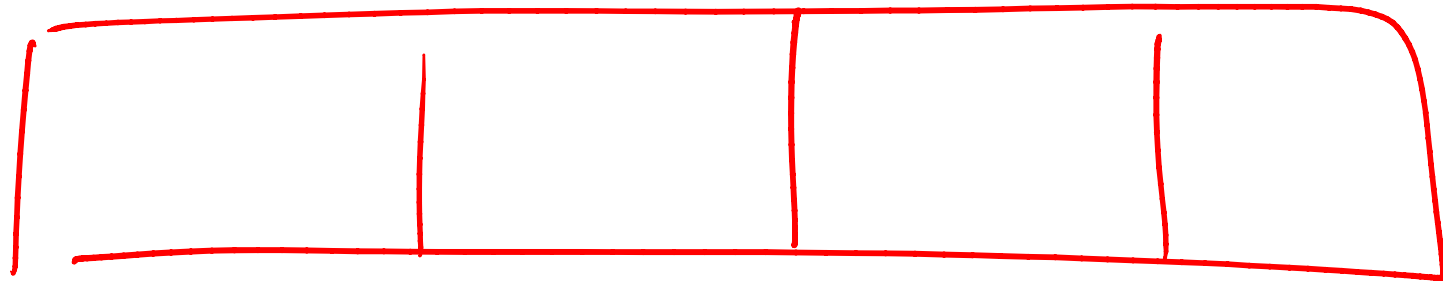
M1S14 When an update occurs to a record and snapshot isolation is enabled, the pre-change record is copied into the version store in tempdb. The changed record has a 14-byte tag on the end with the timestamp and pointer to previous record version. A chain of previous versions is possible.

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





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.

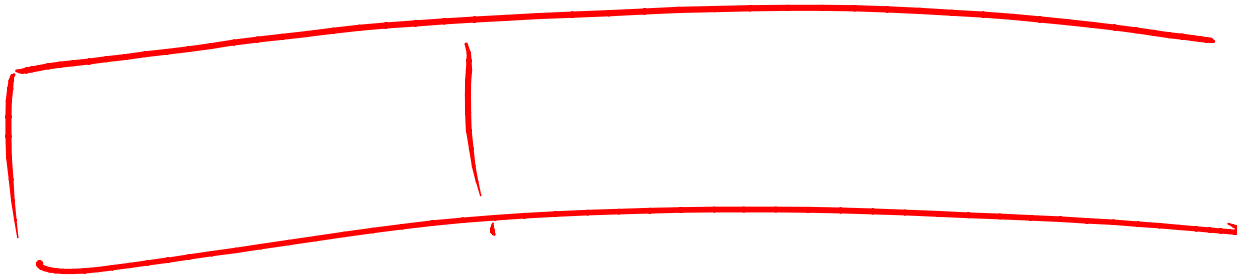


0 — 7 8 — 15 16 — 23



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'.



0 - FH

1 - PFS

2 - GAM

3 - SGAM

4 - UNUSED

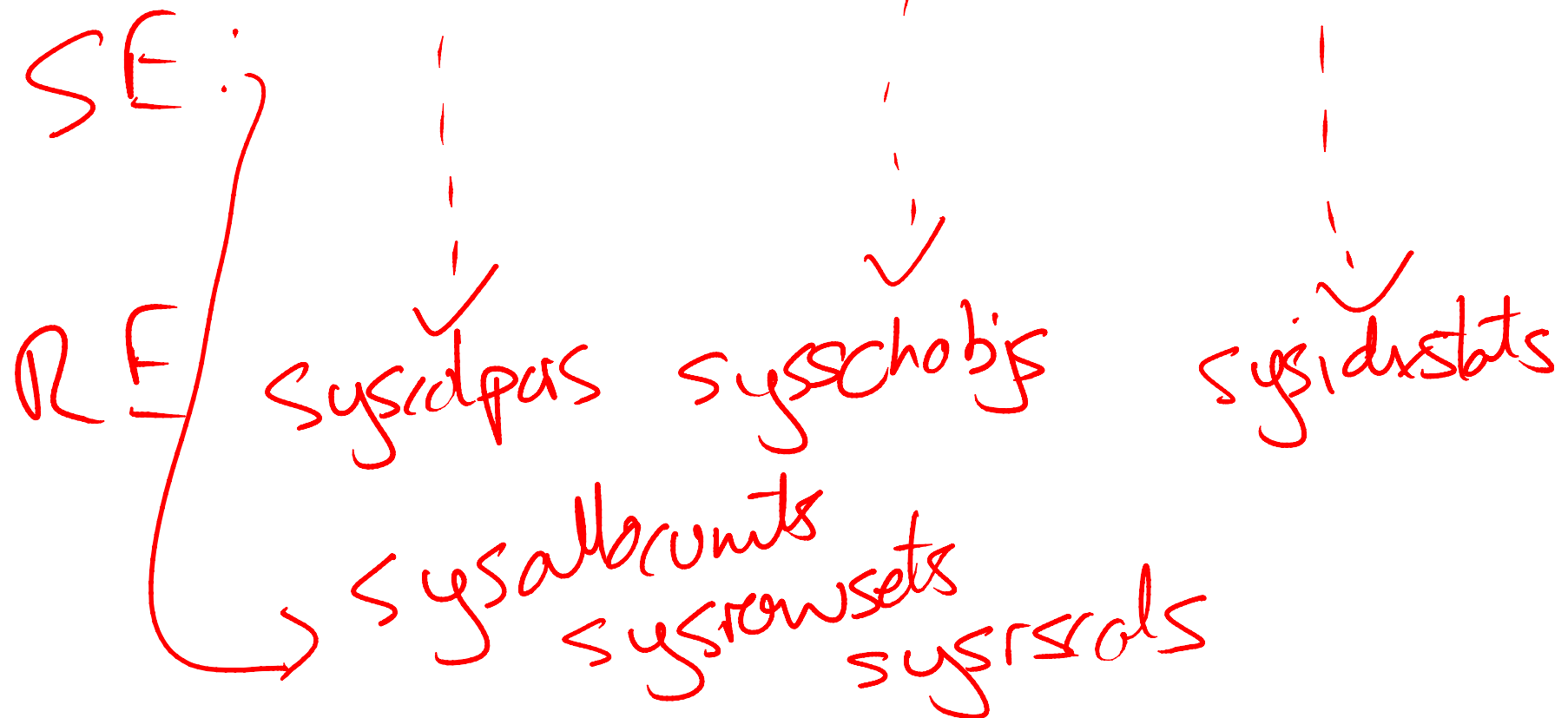
5 - UNUSED

6 - DIFF

7 - ML

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

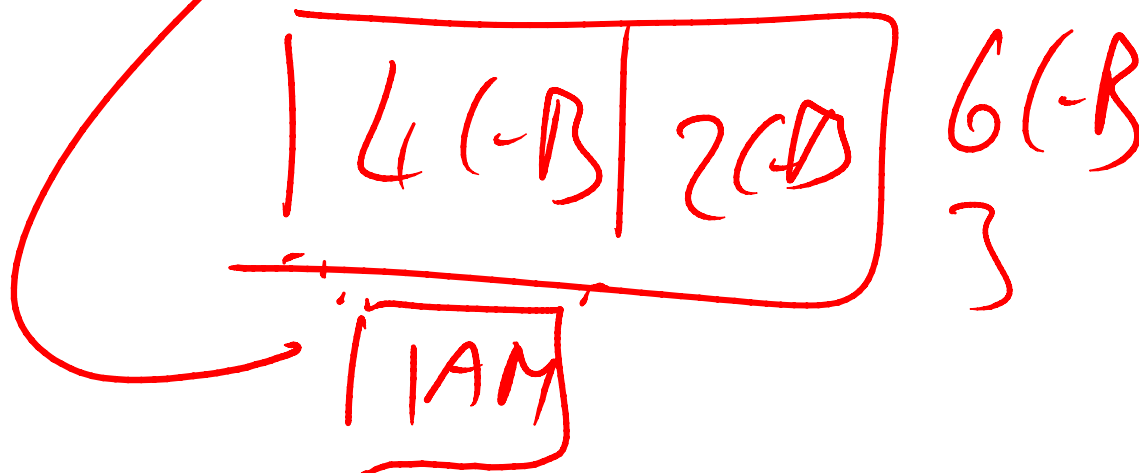
2000: syscolumns, sysobjects, sysindexes



M1 Explaining the names of the hidden system tables that store column metadata.



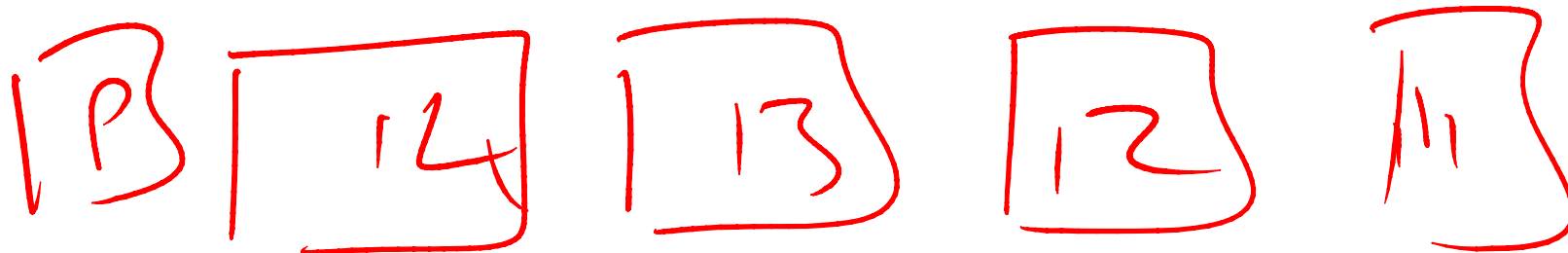
M1S36 Example of IAM chains.
Allocations from each 4GB
chunk of a data file require an
IAM page to track the
allocation. Multiple IAM pages
make an IAM chain.



SALES 1 TB

2014 2013 2012 2011

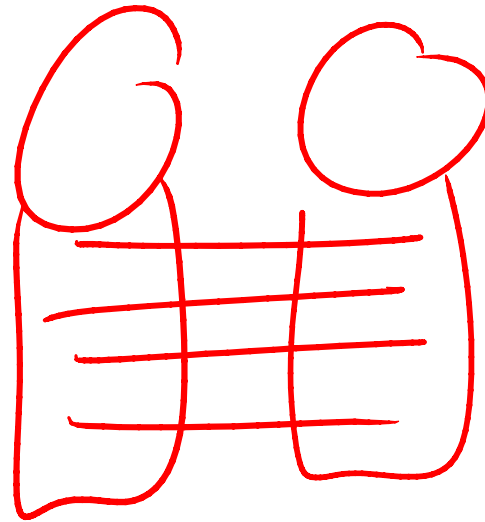
M2S7 Splitting a large database up into multiple filegroups can enable you to do small, targeted restores or even a partial restore from scratch – in the event that the database is damaged or destroyed.





M2S7 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 NSW online first.

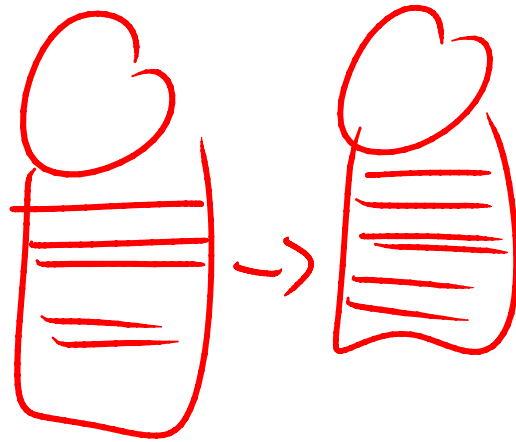
STRIPING



R0

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

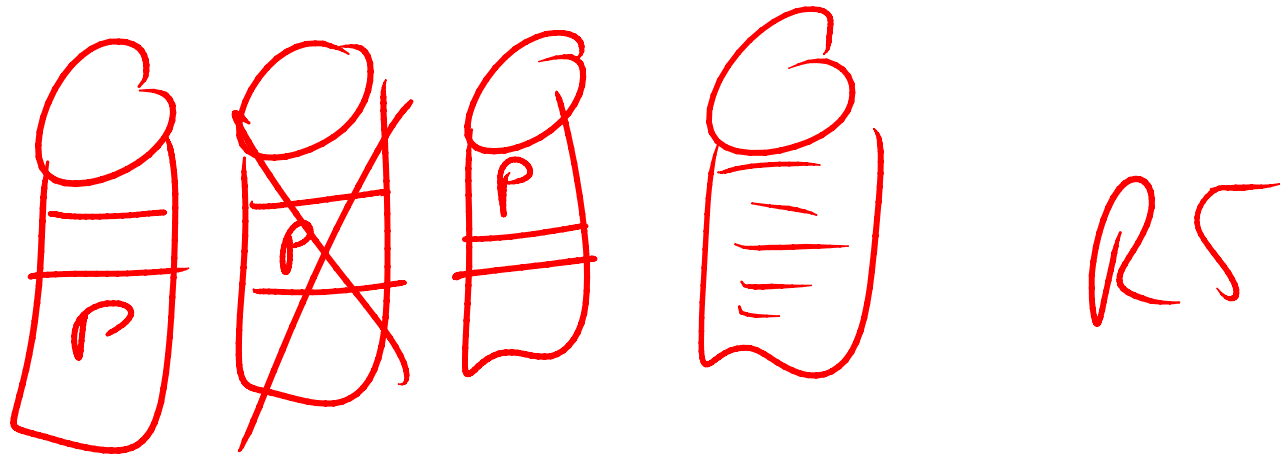
MIRRORING



RAID

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

NOTATING PARITY

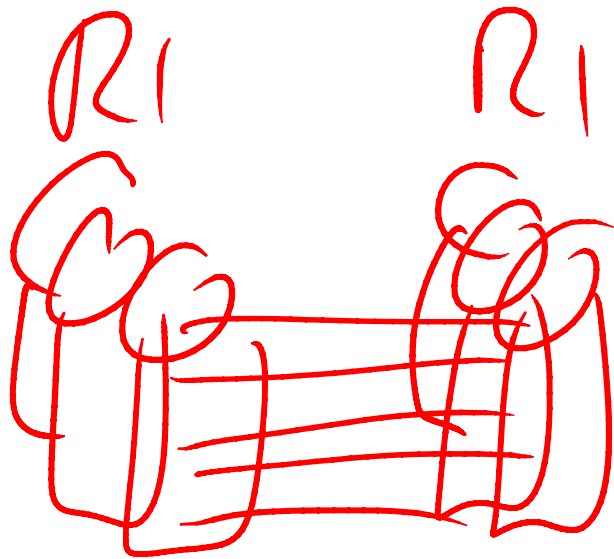


http://en.wikipedia.org/wiki/Standard_RAID_levels

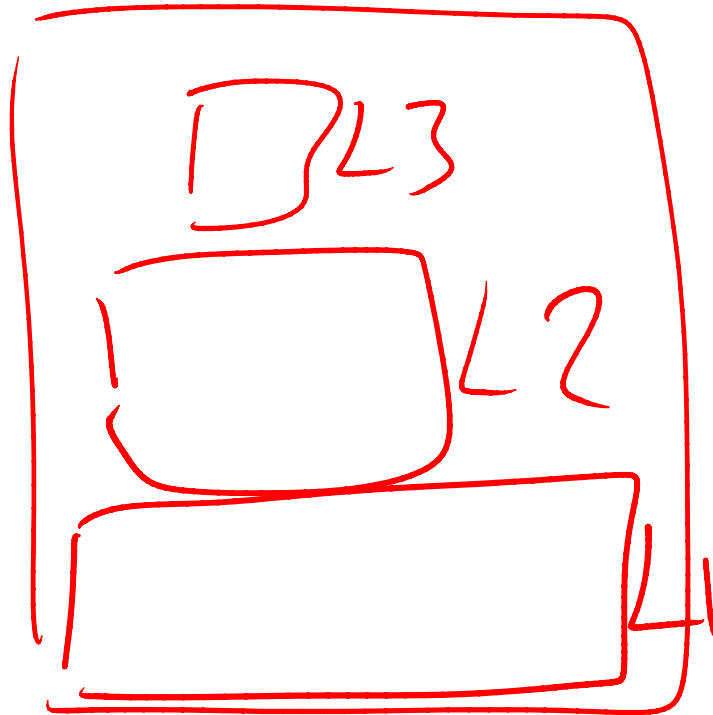
$$R10 = R1 + G$$

= MIRRORING

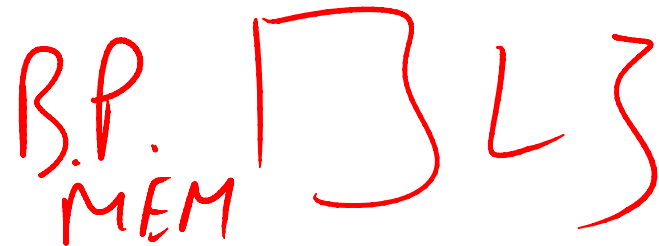
+ STRIPING



CPU

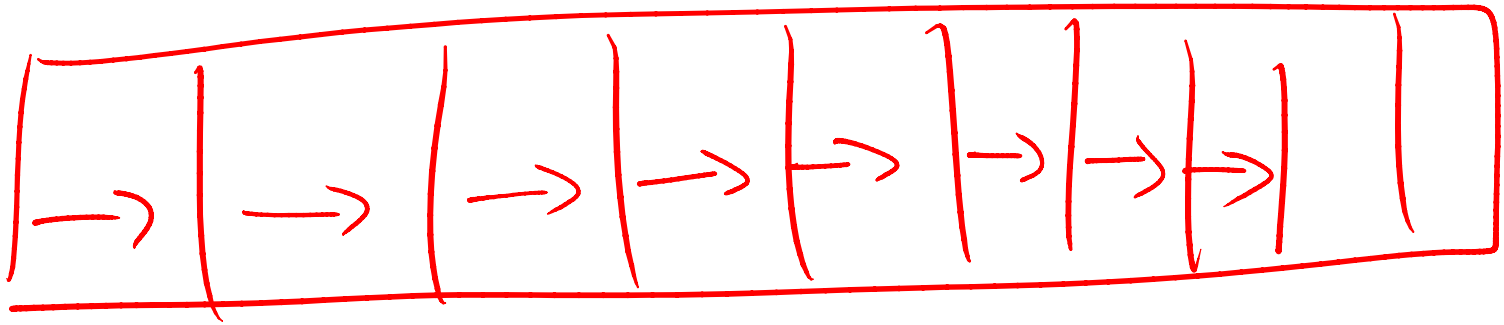


I/O LATENCY



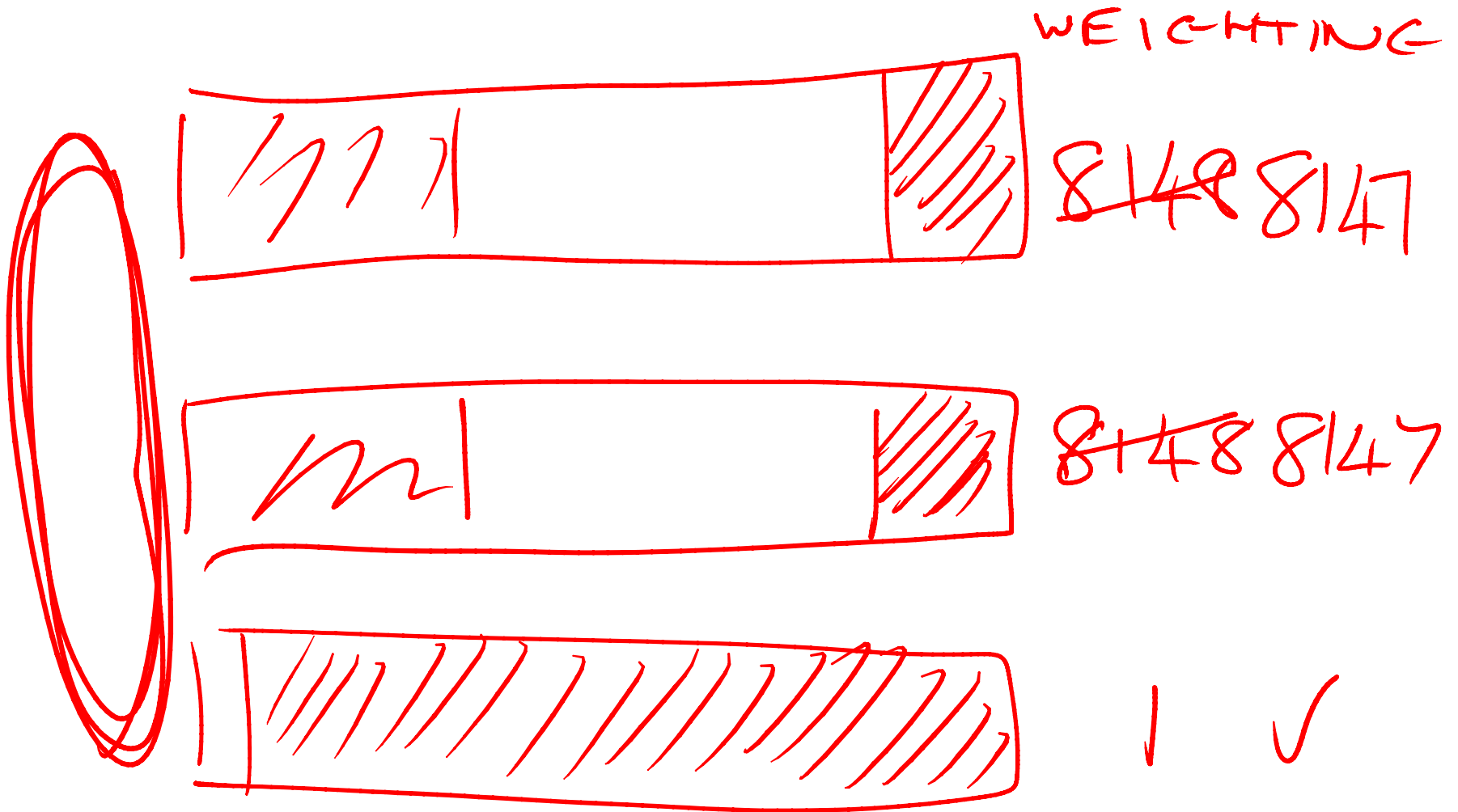
I/O SUBSYSTEM

M2S17 Explaining 2014 Buffer Pool Extension. It's like have L1-L3 caches for your data in memory. BPE sits in between memory and the I/O subsystem.

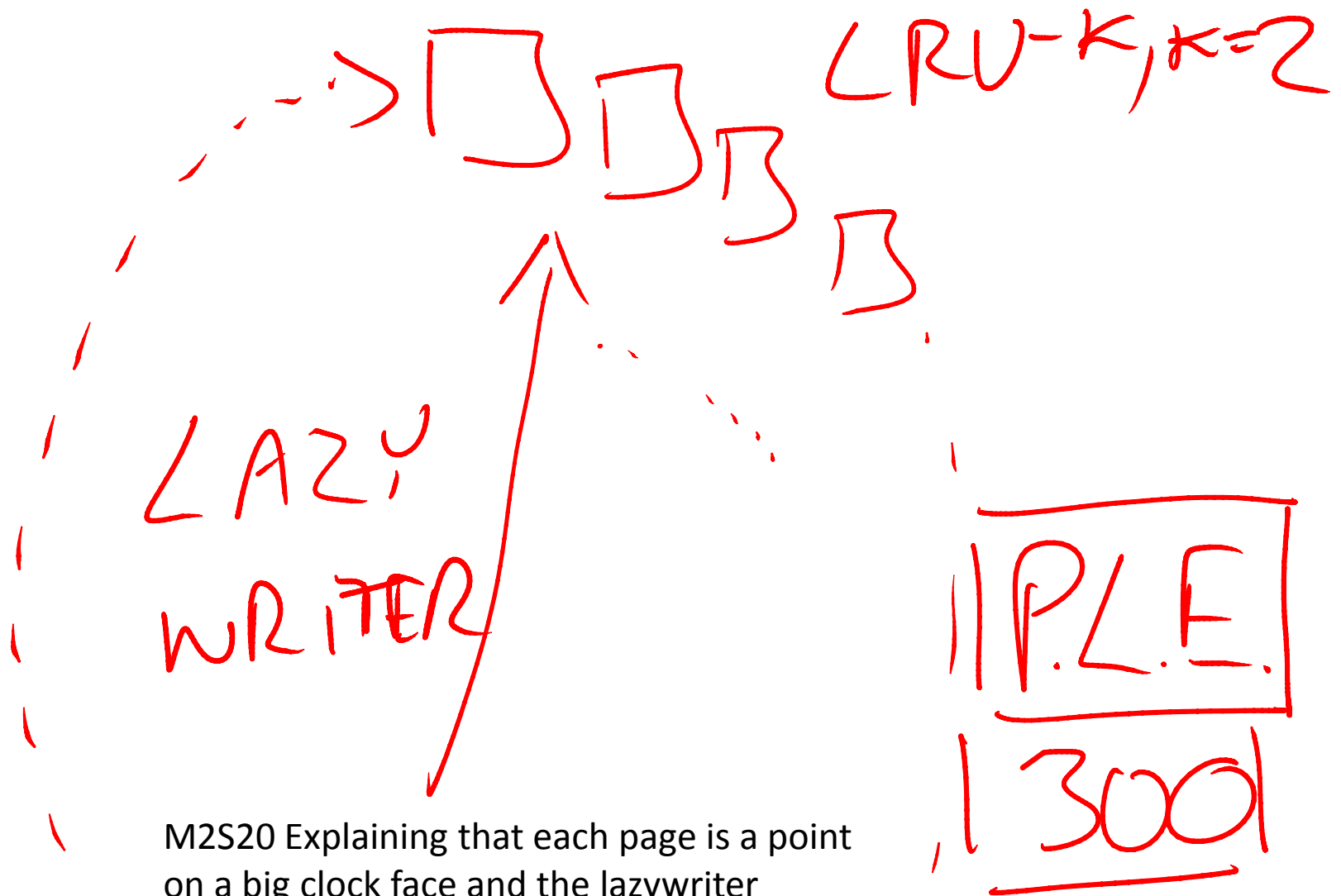


△:

M2S14 If you put all your log files on one volume, its I/O characteristics become random



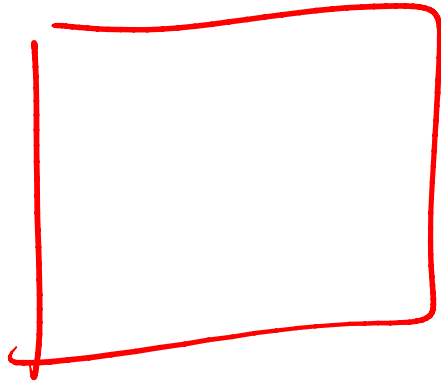
M2S19 Explaining round-robin allocation and proportional fill. Each file has a proportional fill weighting. Allocations happen proportionally more frequently from files with more free space.



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.

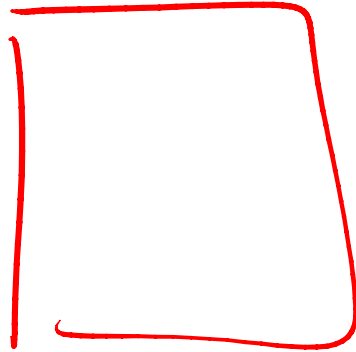
PLE = 300 is a nonsense threshold for tuning. See next slide.

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

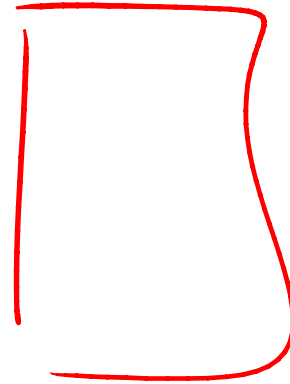


PLE
4000

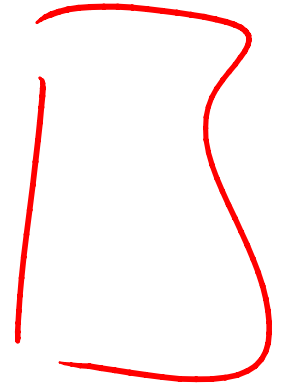
4000



PLE
4000



PLE
4000

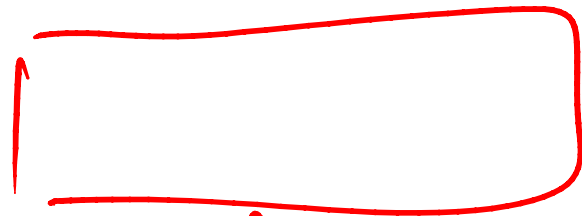


PLE
1000

3625

sys.dm_io_virtual_file_stats
dm_io_pending_io_requests

M2 Tracking I/O latencies: see <http://www.sqlskills.com/blogs/paul/how-to-examine-io-subsystem-latencies-from-within-sql-server/>



ASYNC

< 2012

32 logblocks

48-16

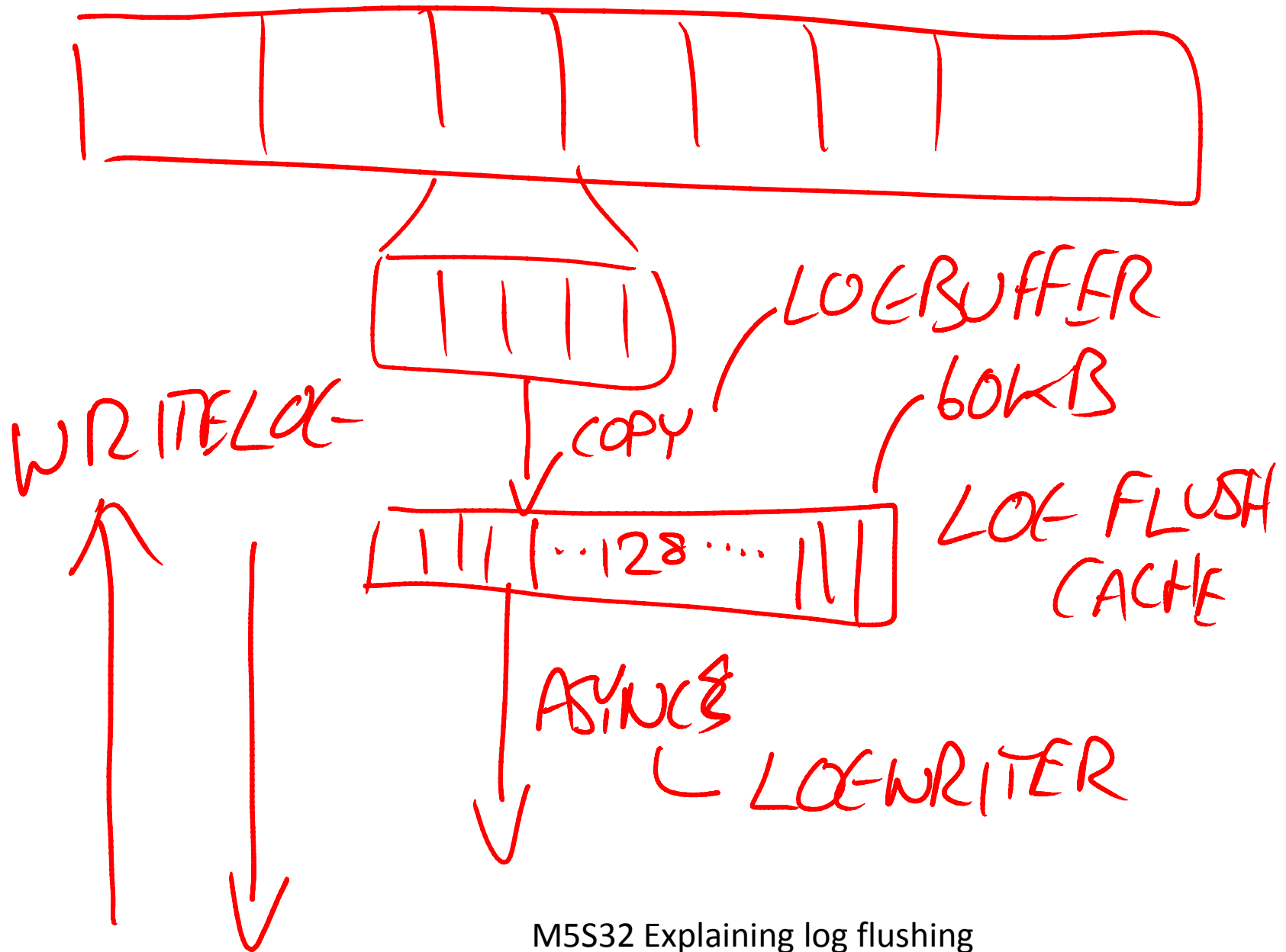
2012 >

128-16

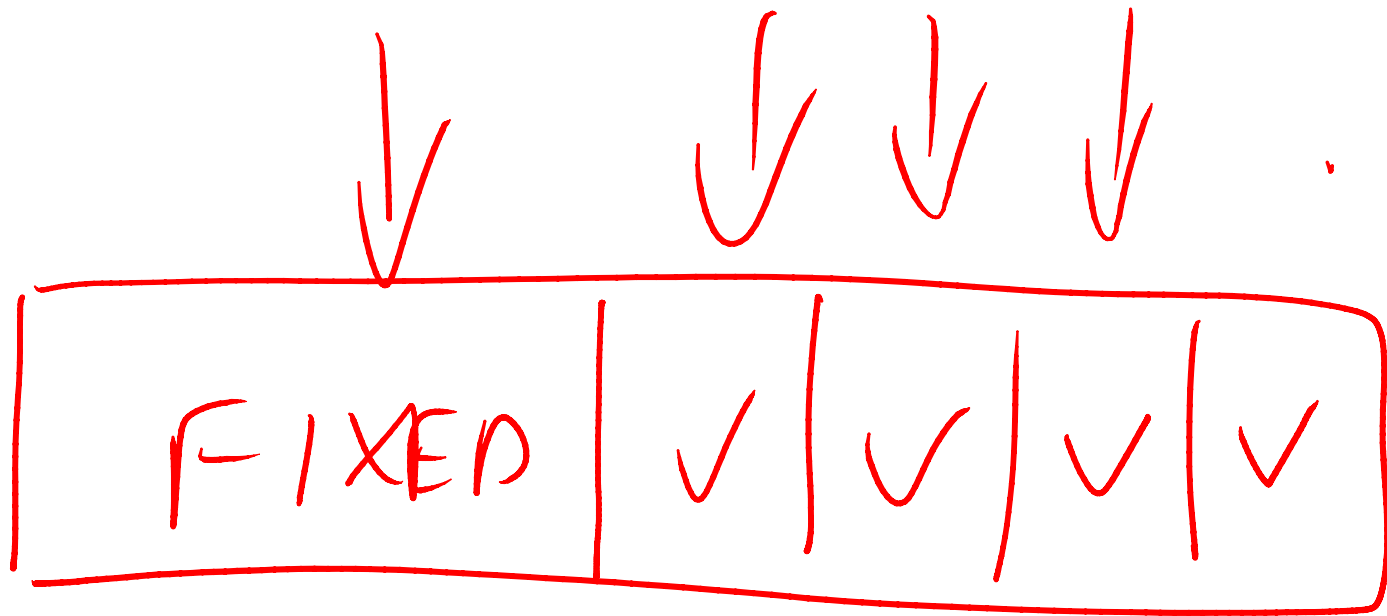
112 logblocks

sys.dm-io-pending-io-requests

M5S46 Explaining log flushing. There is a limit of 32 outstanding async log block I/Os pre-2012. The limit is 112 in 2012.



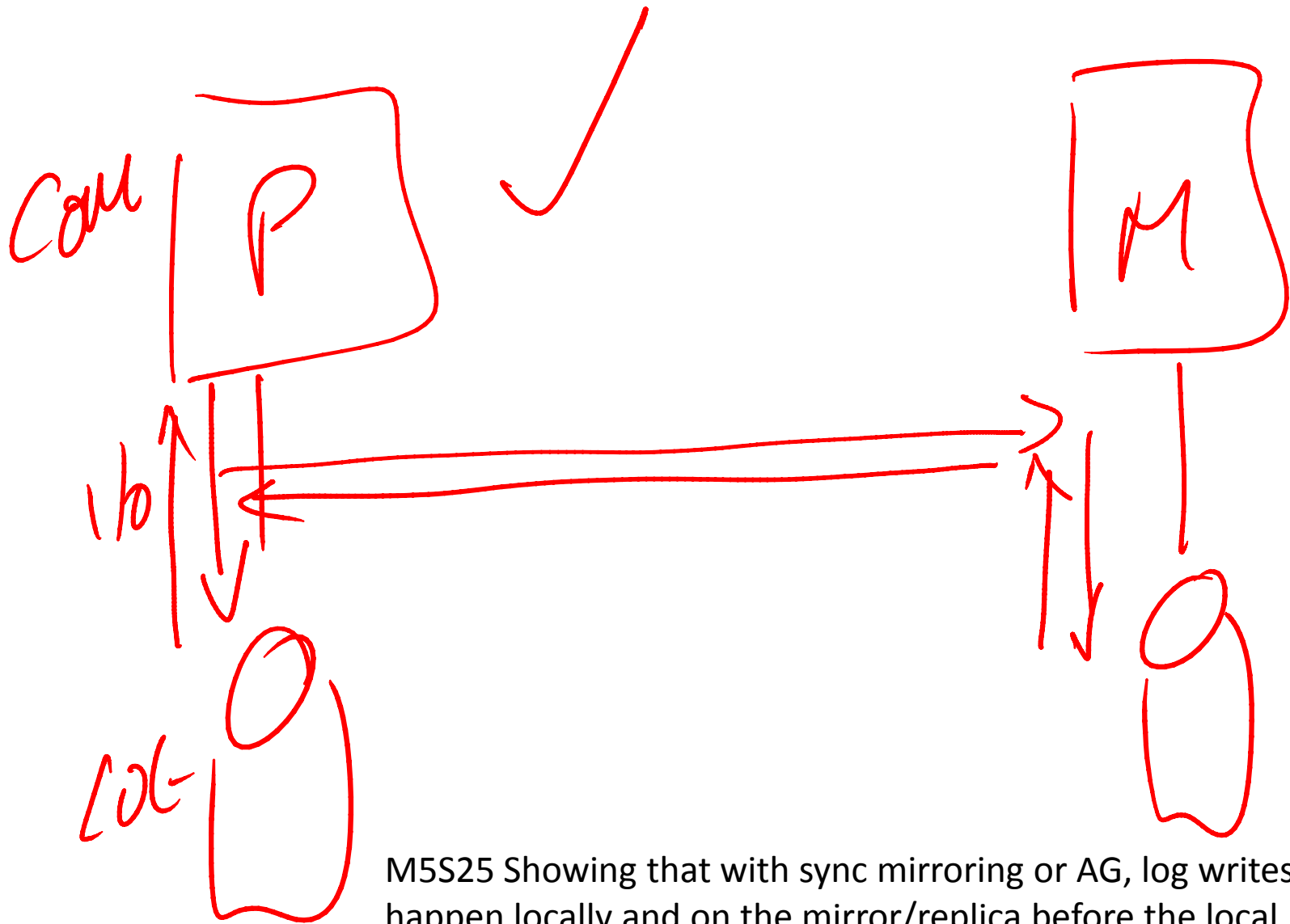
M5S32 Explaining log flushing



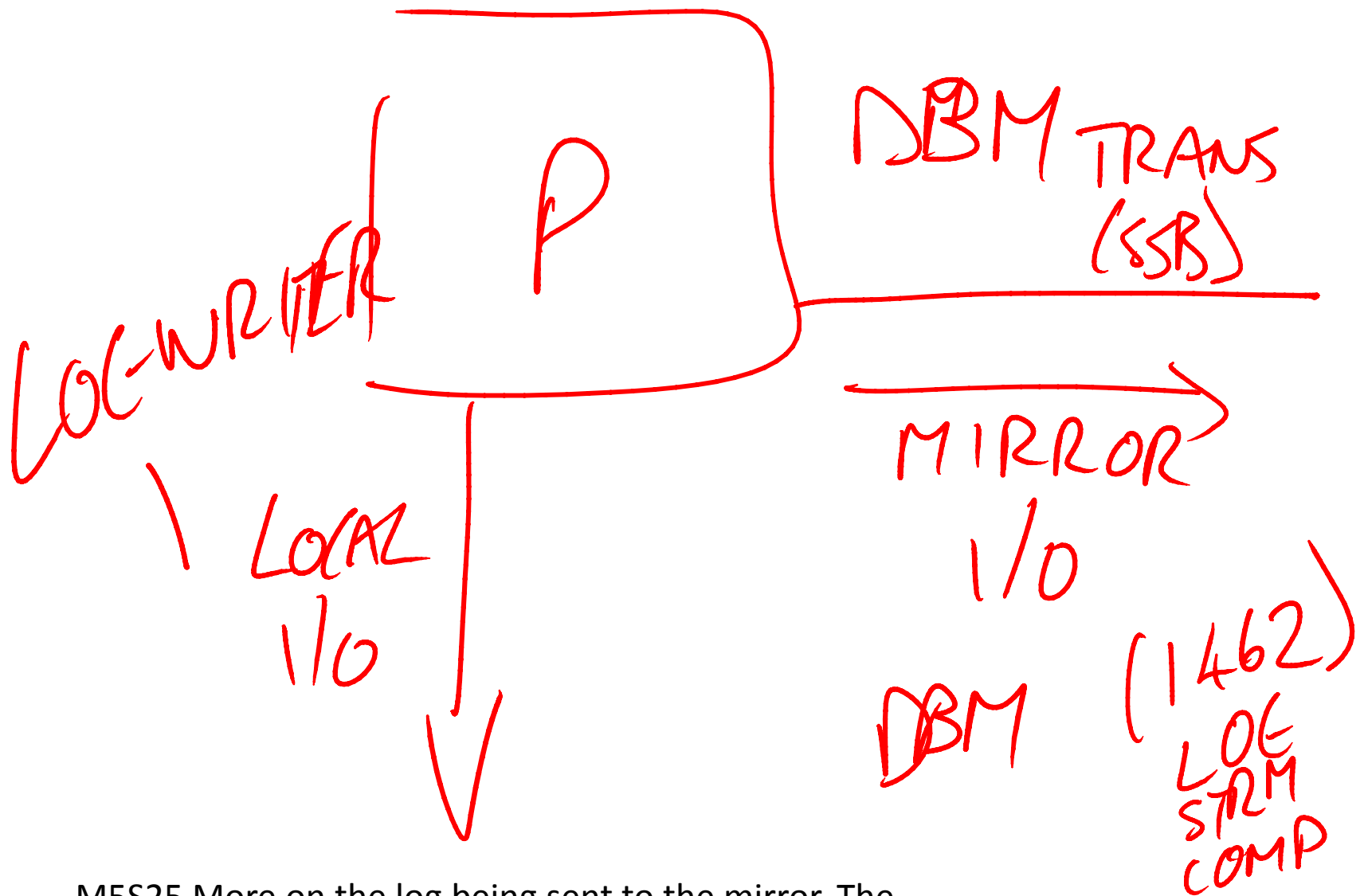
LOP_MODIFY_ROW

LOP_MODIFY_COLUMN

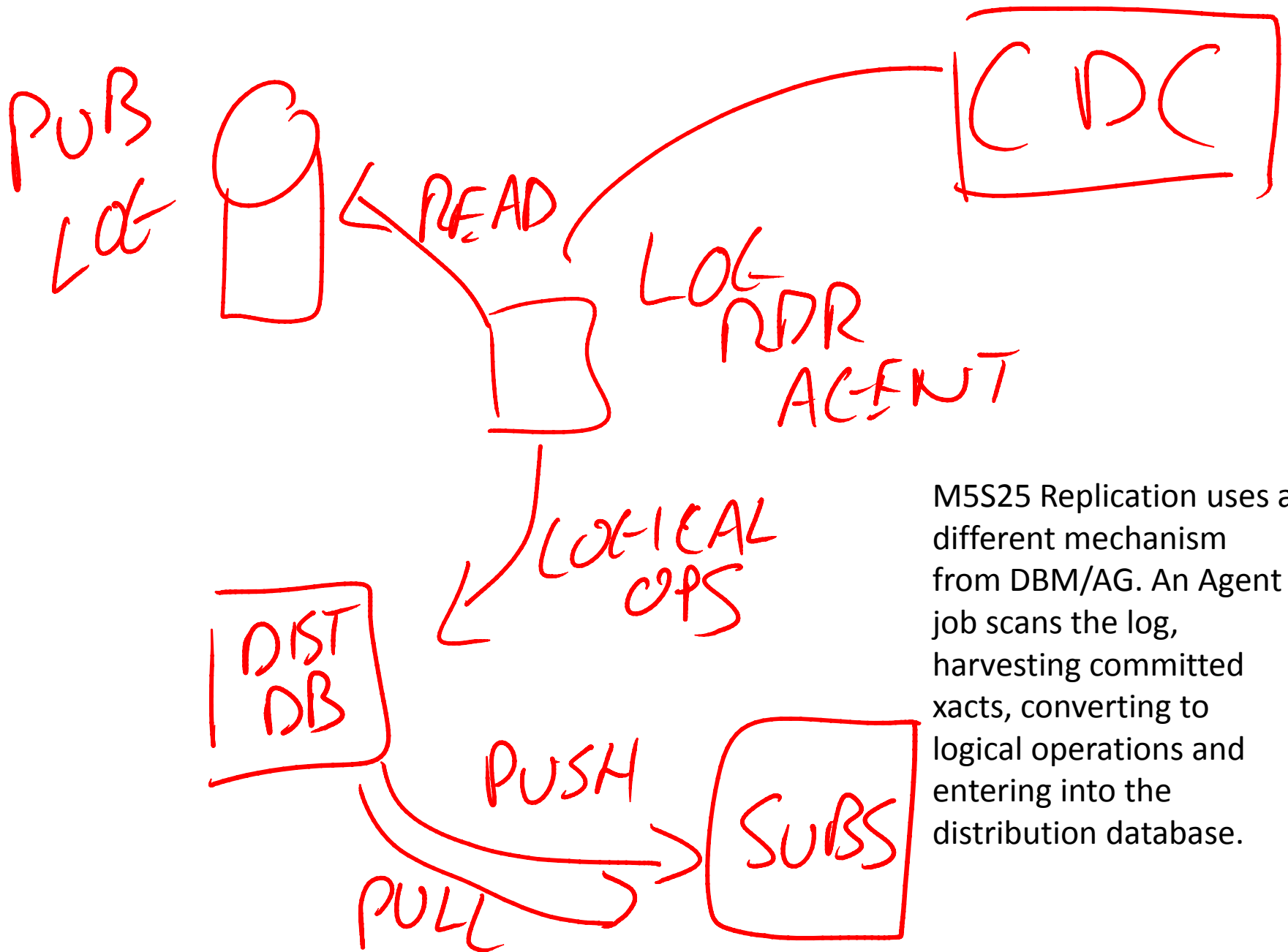
M5S20 Explaining how the Access Methods determines how to log UPDATES efficiently by splitting the logging by portion of the record. Each variable length column is a portion, as is the fixed-width portion (up to 16 bytes in each log record)



M5S25 Showing that with sync mirroring or AG, log writes have to happen locally and on the mirror/replica before the local transaction can commit. The I/Os are issued in parallel, not local, complete local, remote.

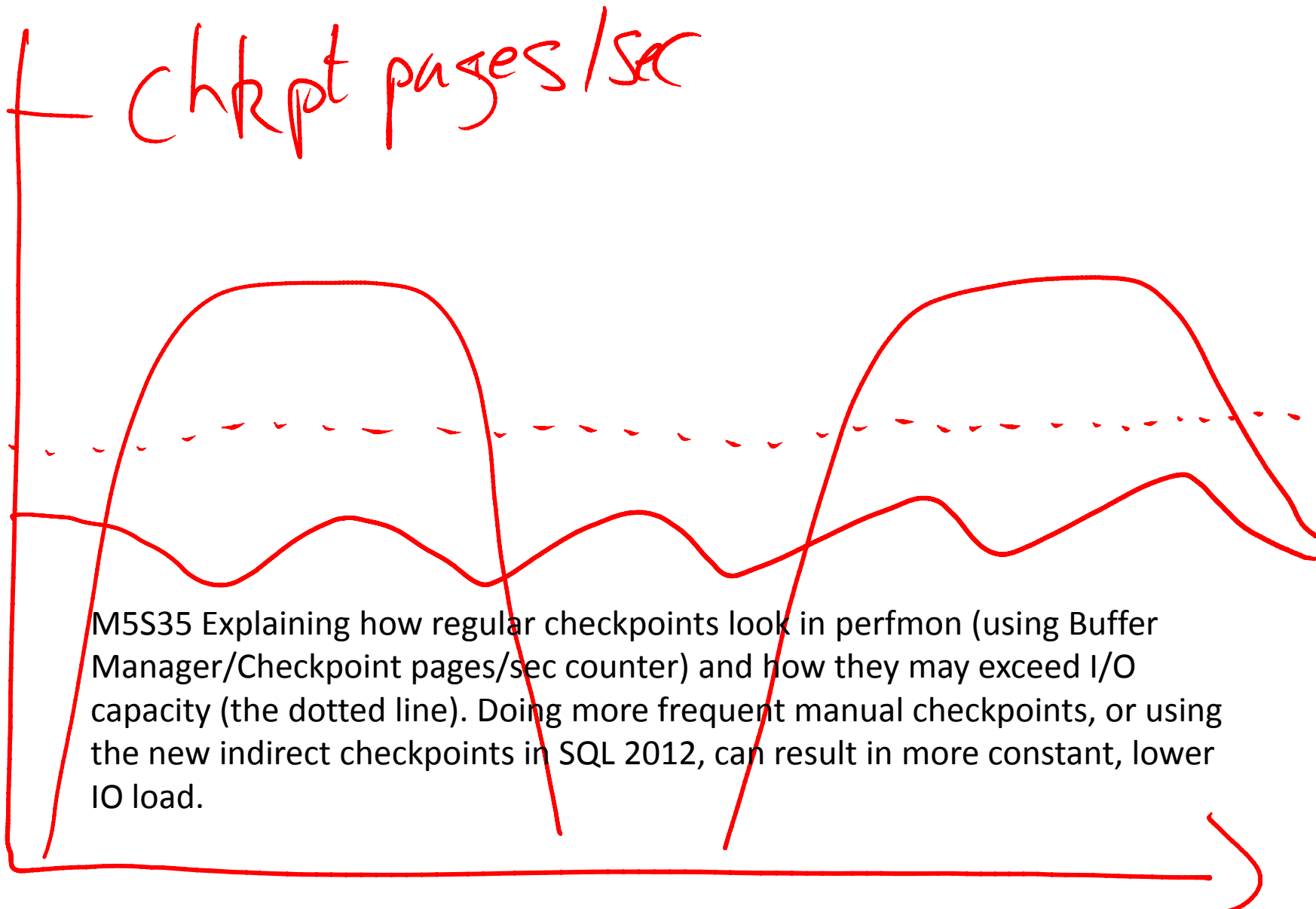


M5S25 More on the log being sent to the mirror. The local and remote I/Os are two separate operations.

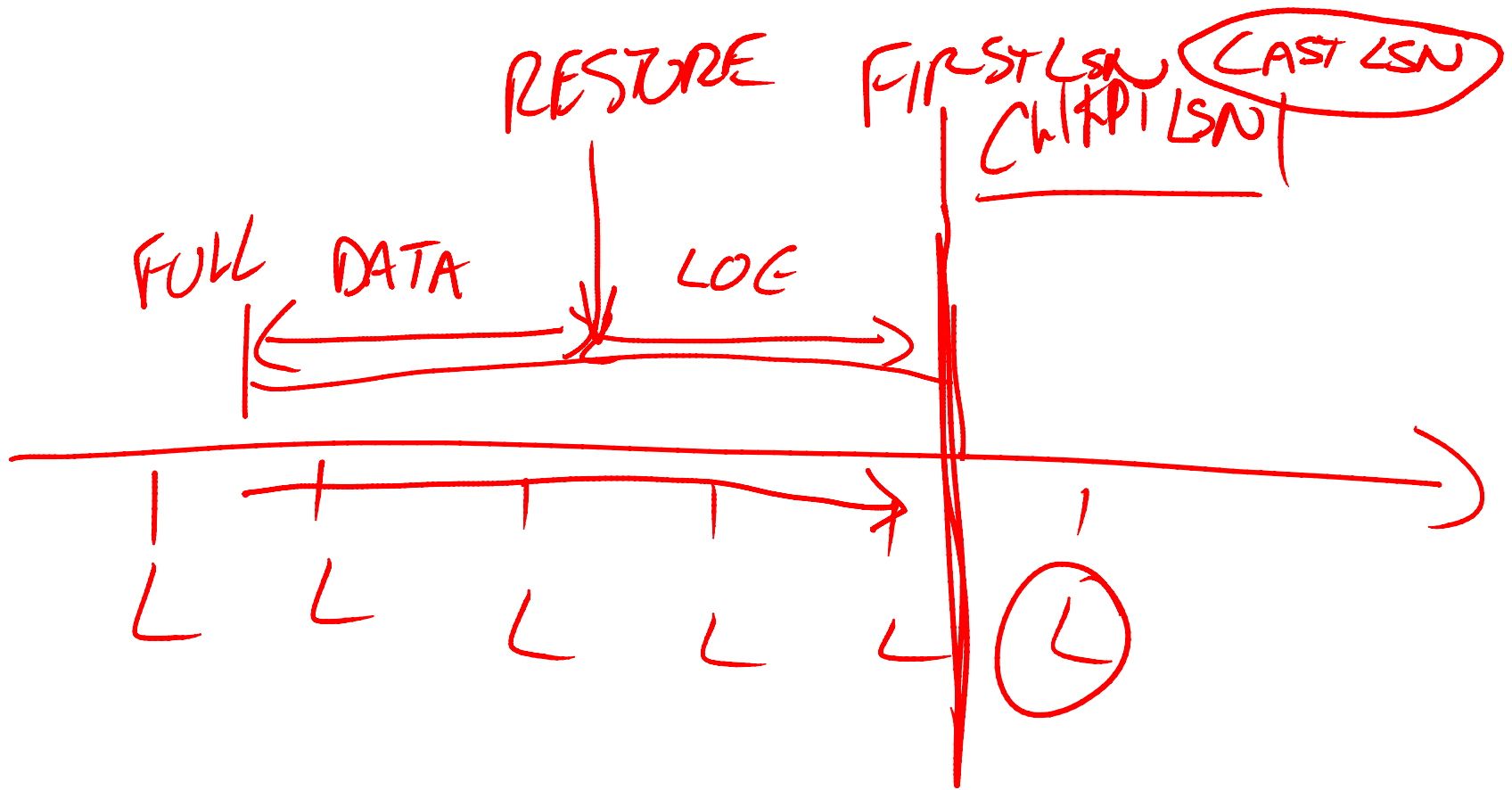


M5S25 Replication uses a different mechanism from DBM/AG. An Agent job scans the log, harvesting committed xacts, converting to logical operations and entering into the distribution database.

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.



M5S43 It's possible for concurrent log backups to occur with a data backup (full or differential). Log clearing cannot occur though, as the data backup will back up the log. When the data backup finishes, deferred log clearing takes place.

The point in time you get when you restore a data backup (and don't restore any more) is the point when the data reading portion of the backup ended (marked RESTORE in the diagram).

100,000

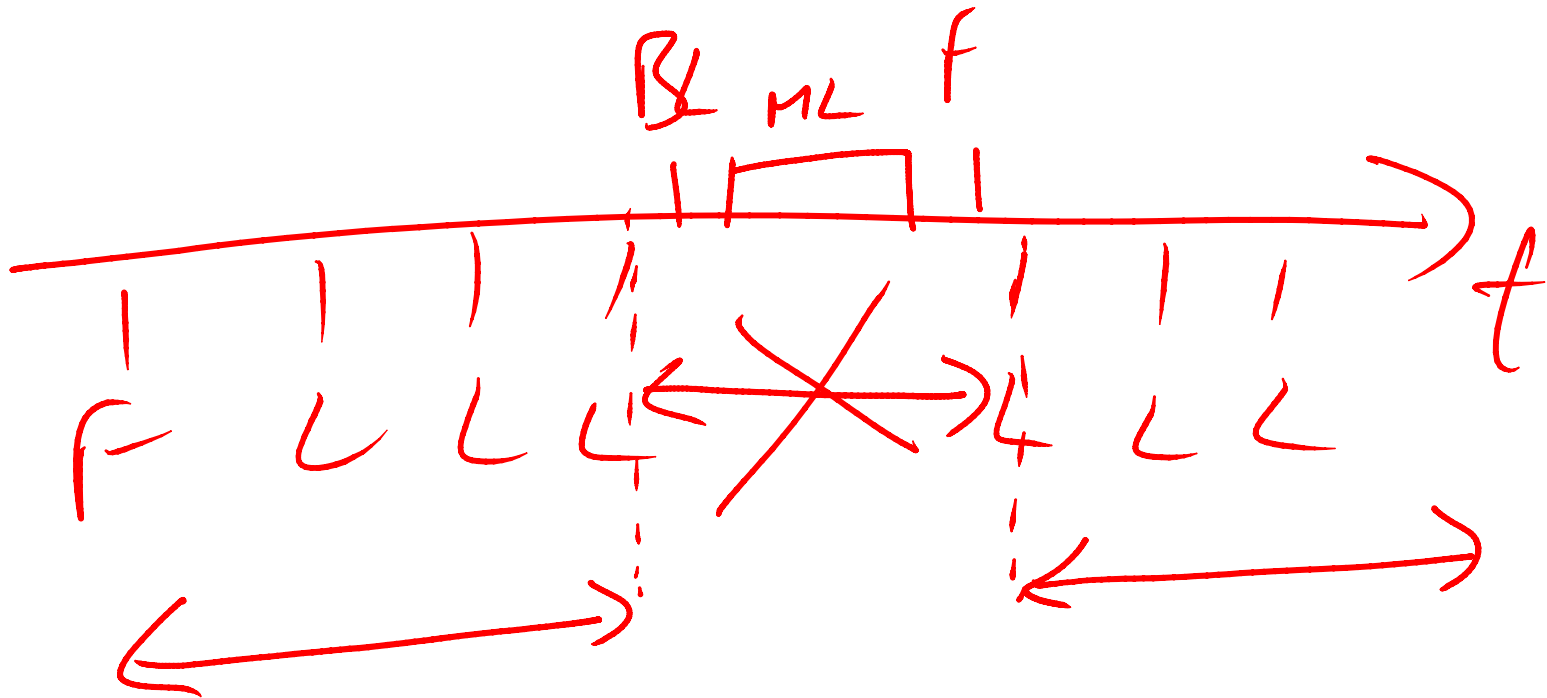
10

M5S59 Explaining that full recovery doesn't mean there's always one log record per operation. An index rebuild in full recovery will log full LOP_FORMAT_PAGE log records instead of multiple LOP_INSERT_ROWS log records. It's more efficient, but is still fully logged.

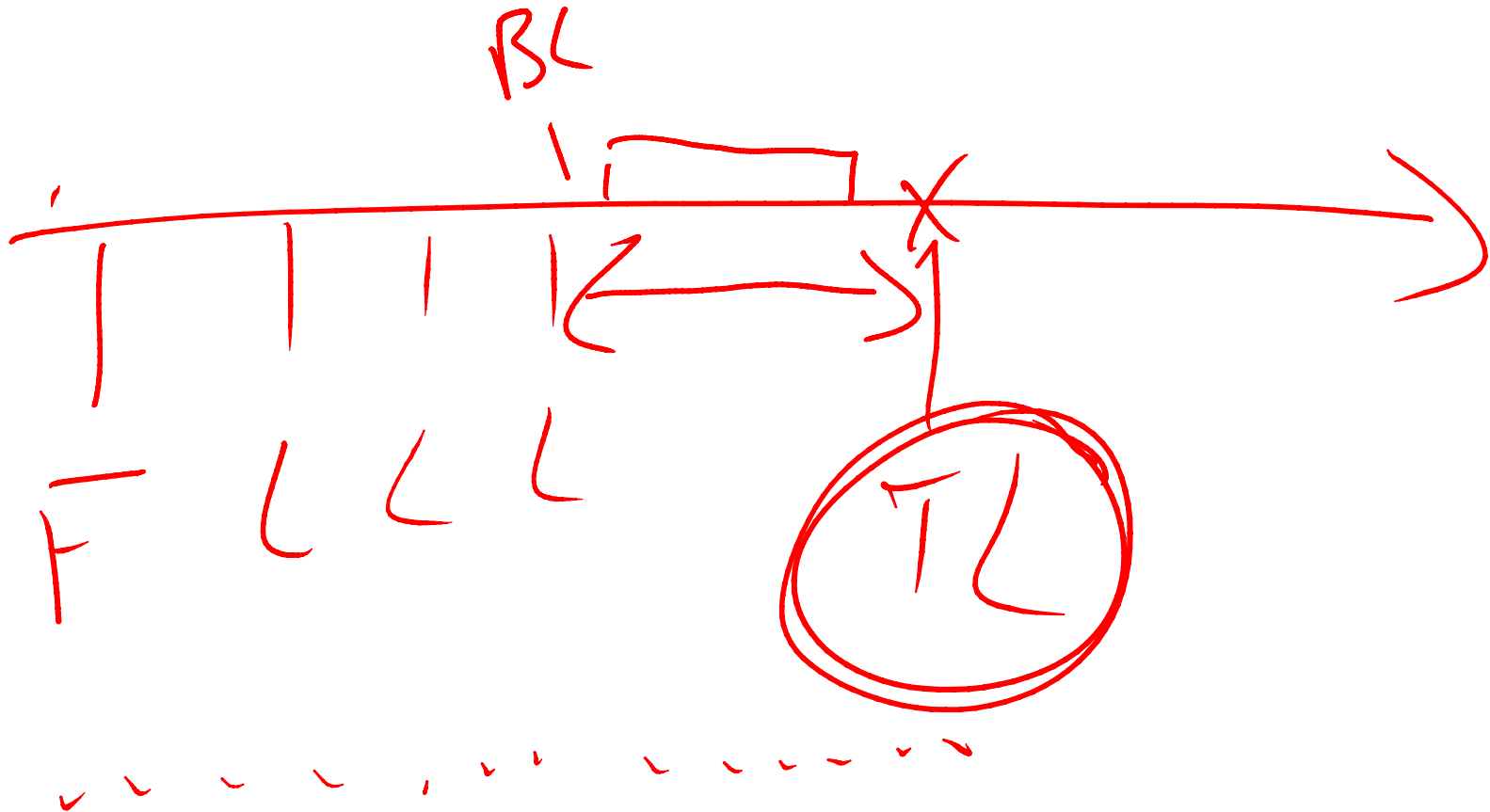
~~INS
INS
INS~~

F - PAGE
F - PAGE

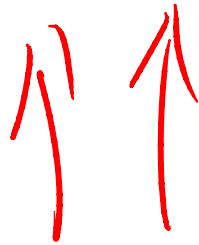
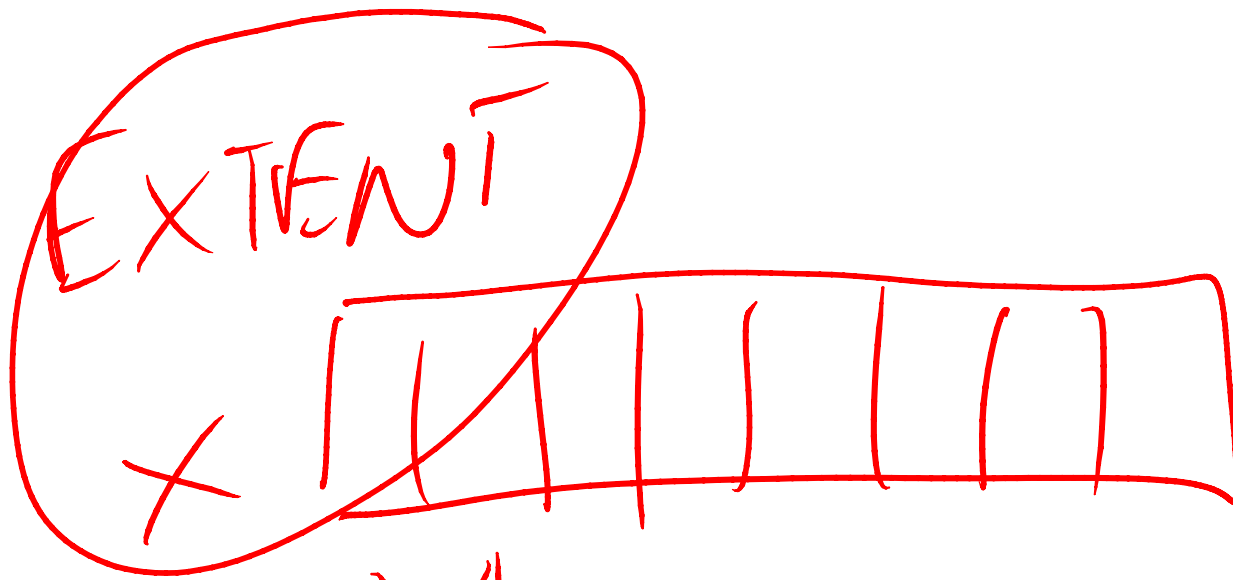
10



M5S61 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.



M5S61 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). You also 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.



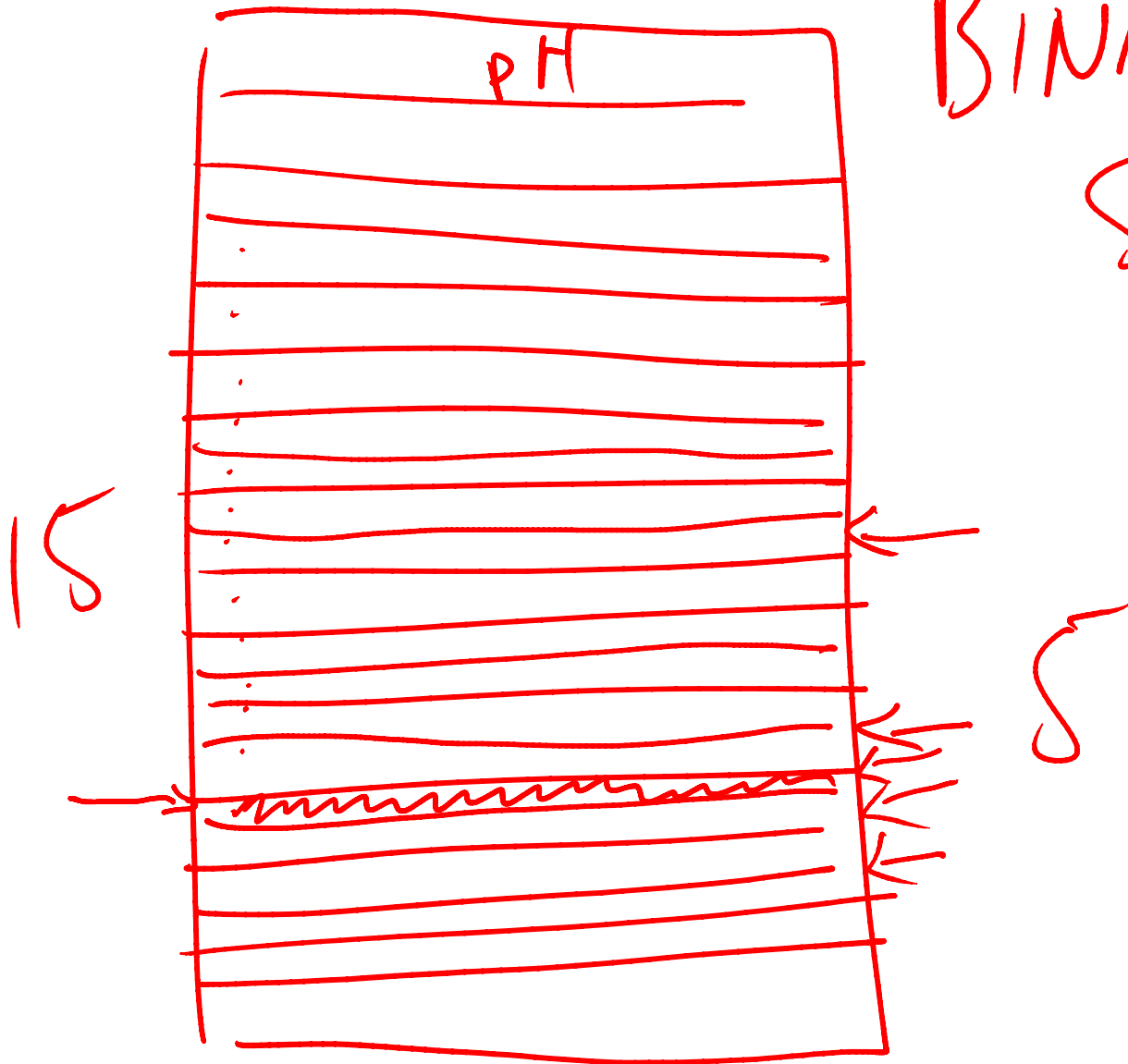
PROBE

M5S63 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.

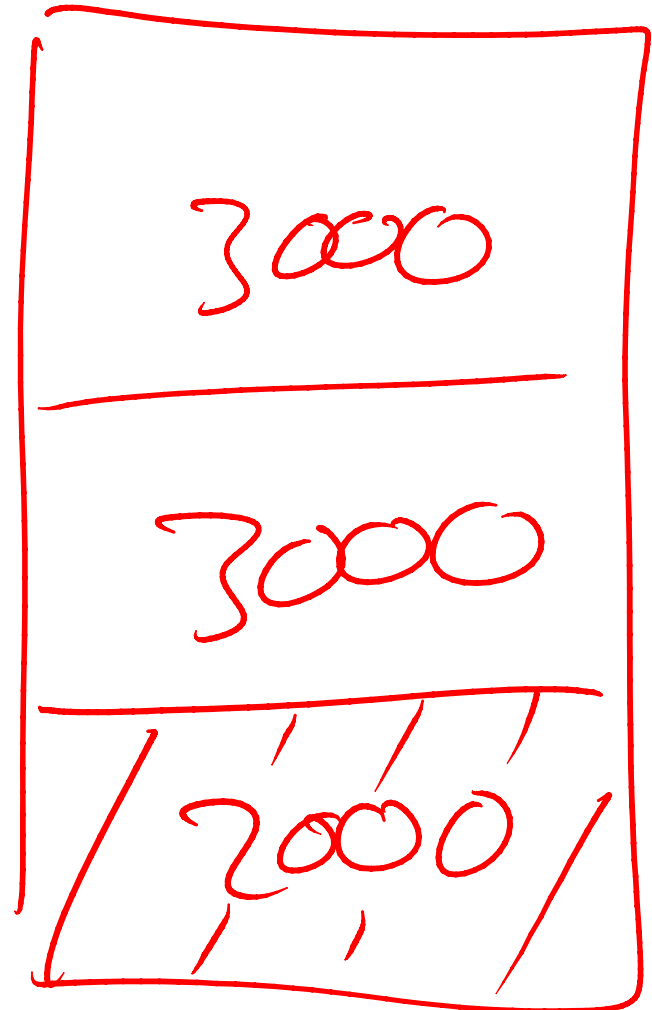
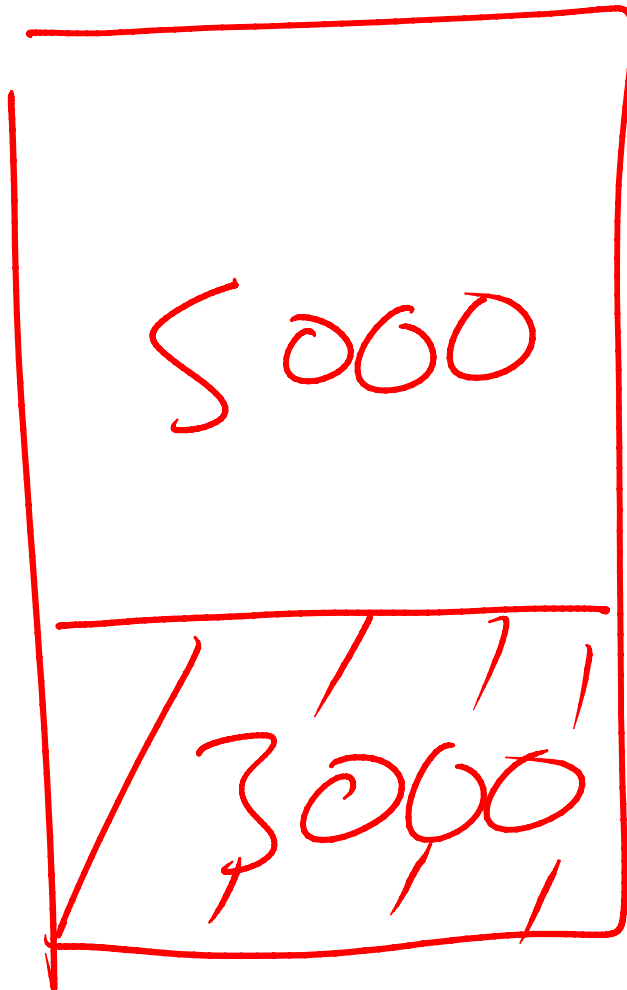


MSS64 VLF fragmentation is bad because of having to search through the VLFs for a particular VLF sequence number. See the KB article link for more info.

BINARY SEARCH

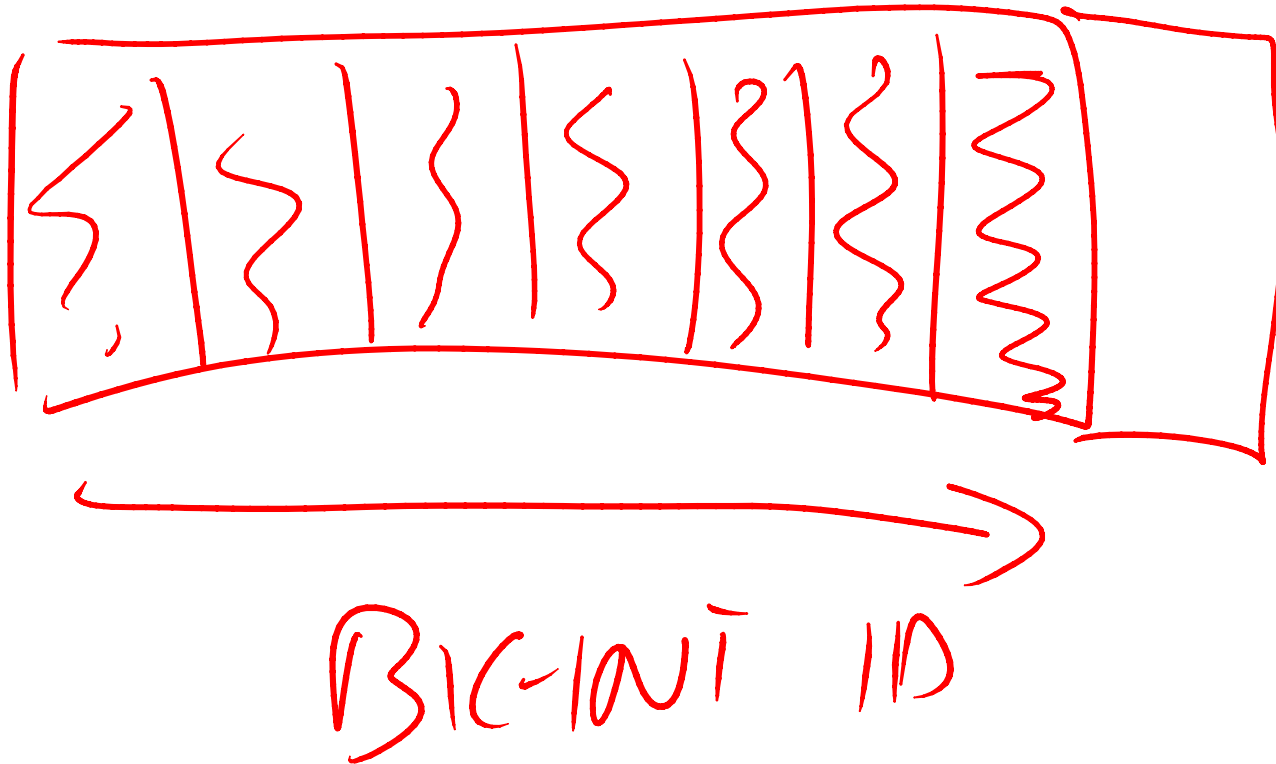


M8S5 Explaining how a record is found on a page using binary search rather than brute force search.

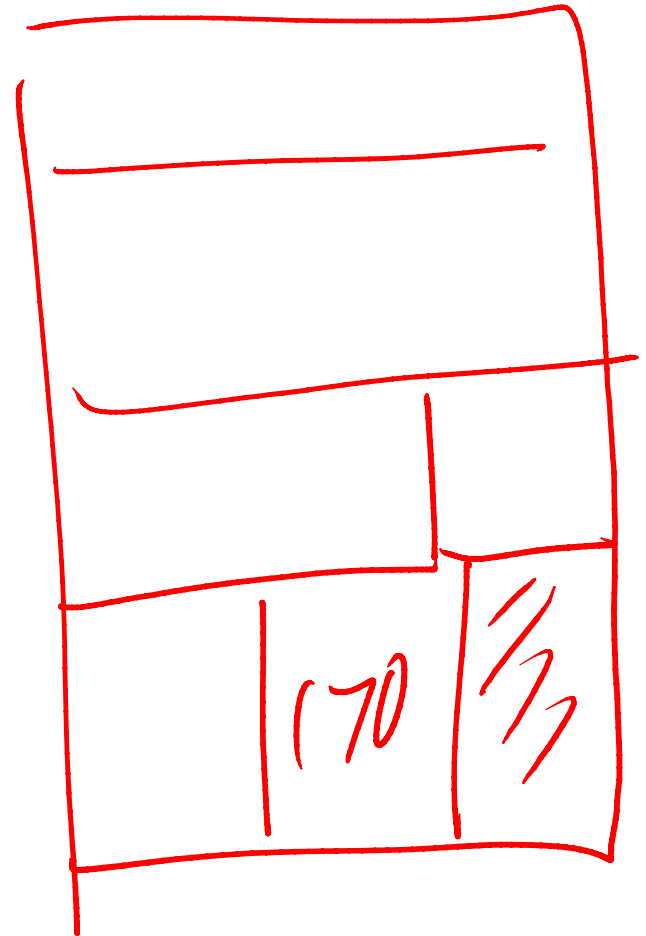
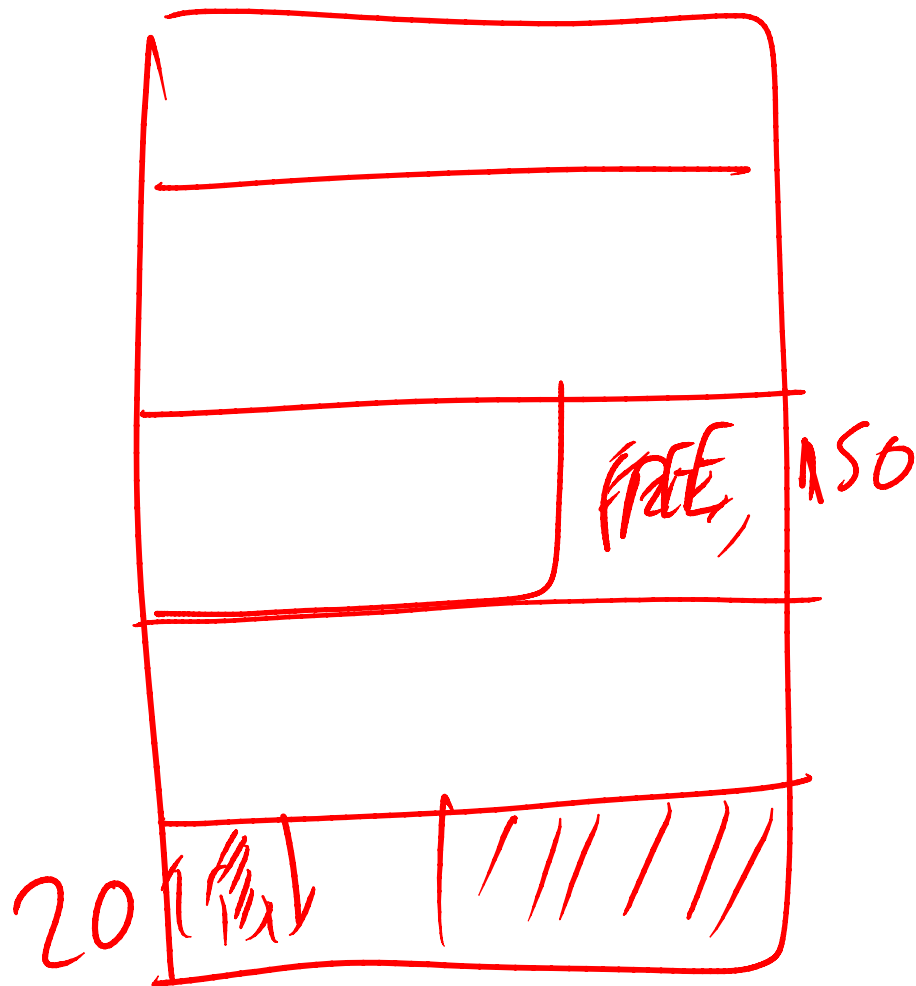


M7S14 Explaining how fixed-size large rows leads to low page density and wasted space.

LOP_DELETE_SPLIT

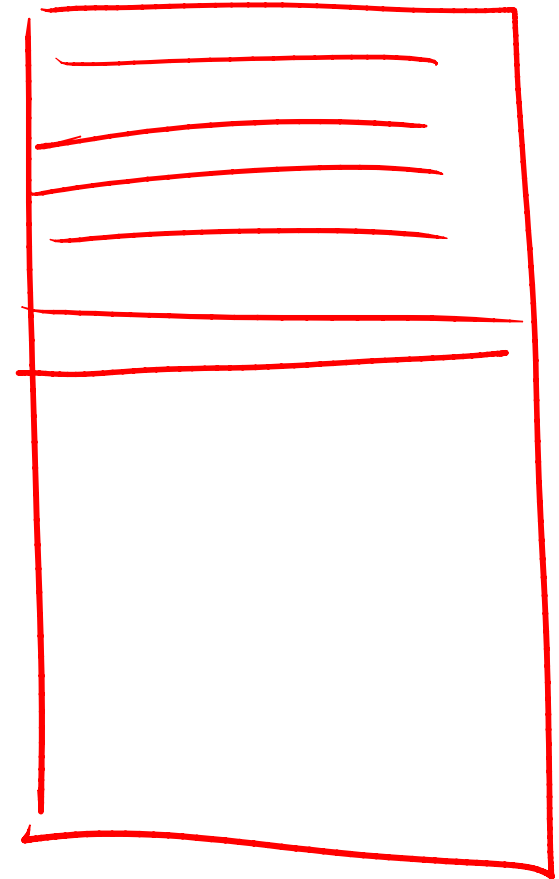
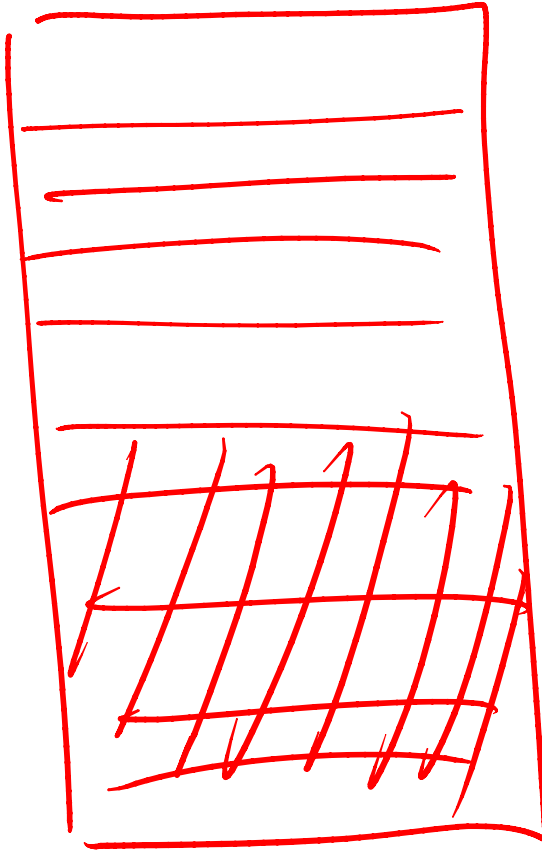


M8S18 Append-only insert pattern fills pages on right-hand side of index. New page allocations in this case are called page splits too – making all methods of tracking page splits largely useless (until 2012 with Xevents). You can look for LOP_DELETE_SPLIT log records to see splits occurring.

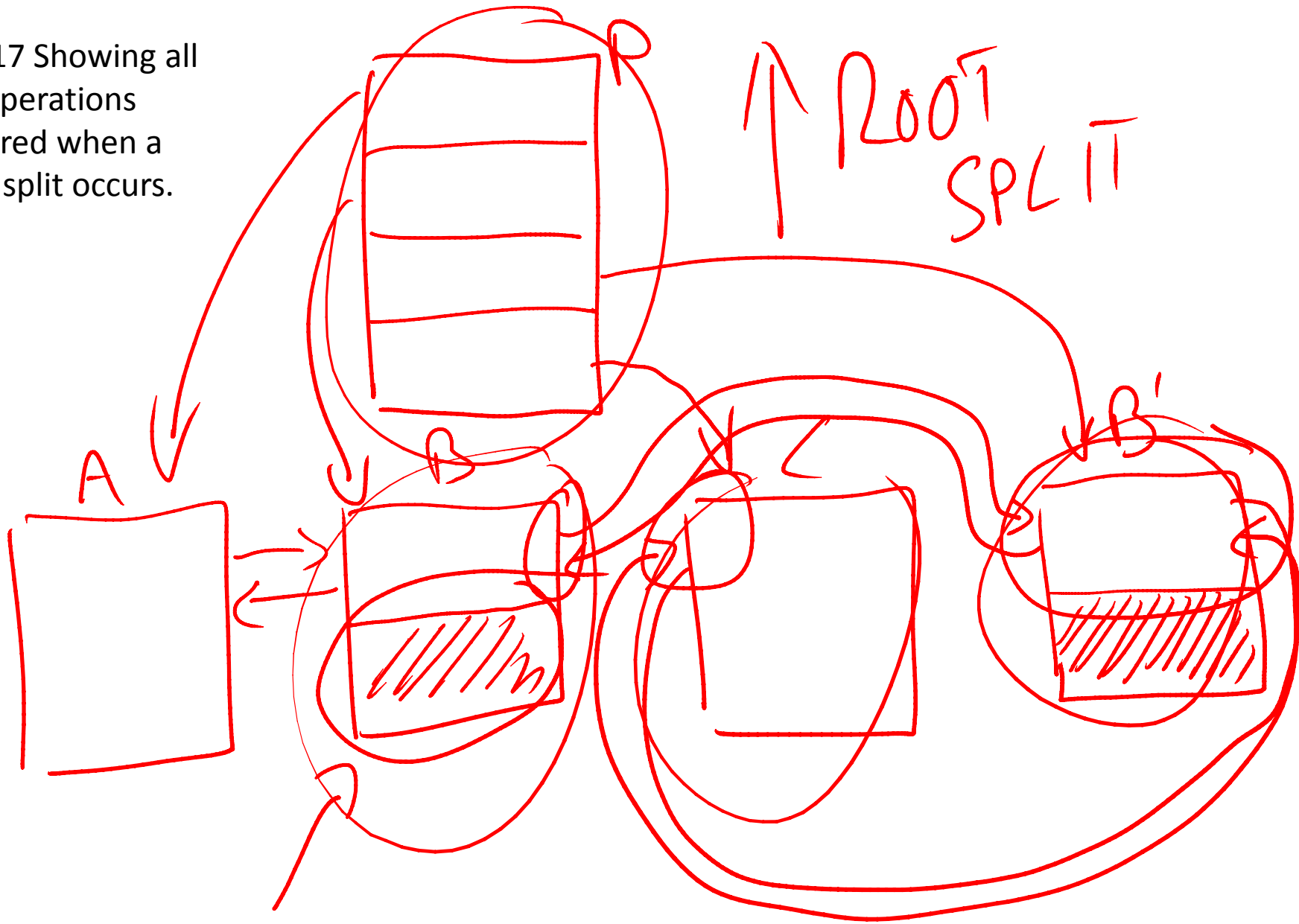


M7S16 If the Access Methods knows there is enough free space on the page, but it's not contiguous, it will do an in-memory page compaction rather than a page split.

M7S16 When a page split occurs, it's usually 50/50 and creates 50% free space on each page.



M7S17 Showing all the operations required when a page split occurs.



	NO SPLIT	SPLIT	
1000	1256	7572	x5
100	356	6764	x19
10	268	11164	x45

M7S23 The numbers from the page split logging cost demo.

COMMENTS

PAUL	JON	ERIN	ELENA
MEMBER_ID			K

M7S24 The MySpace story. The 'K' is the tiny portion of comments on Kimberly's page because she wasn't very popular back then...



ITB



M7S33 Explaining staggered index maintenance with database mirroring. See <http://bit.ly/IS04W5> for more explanation