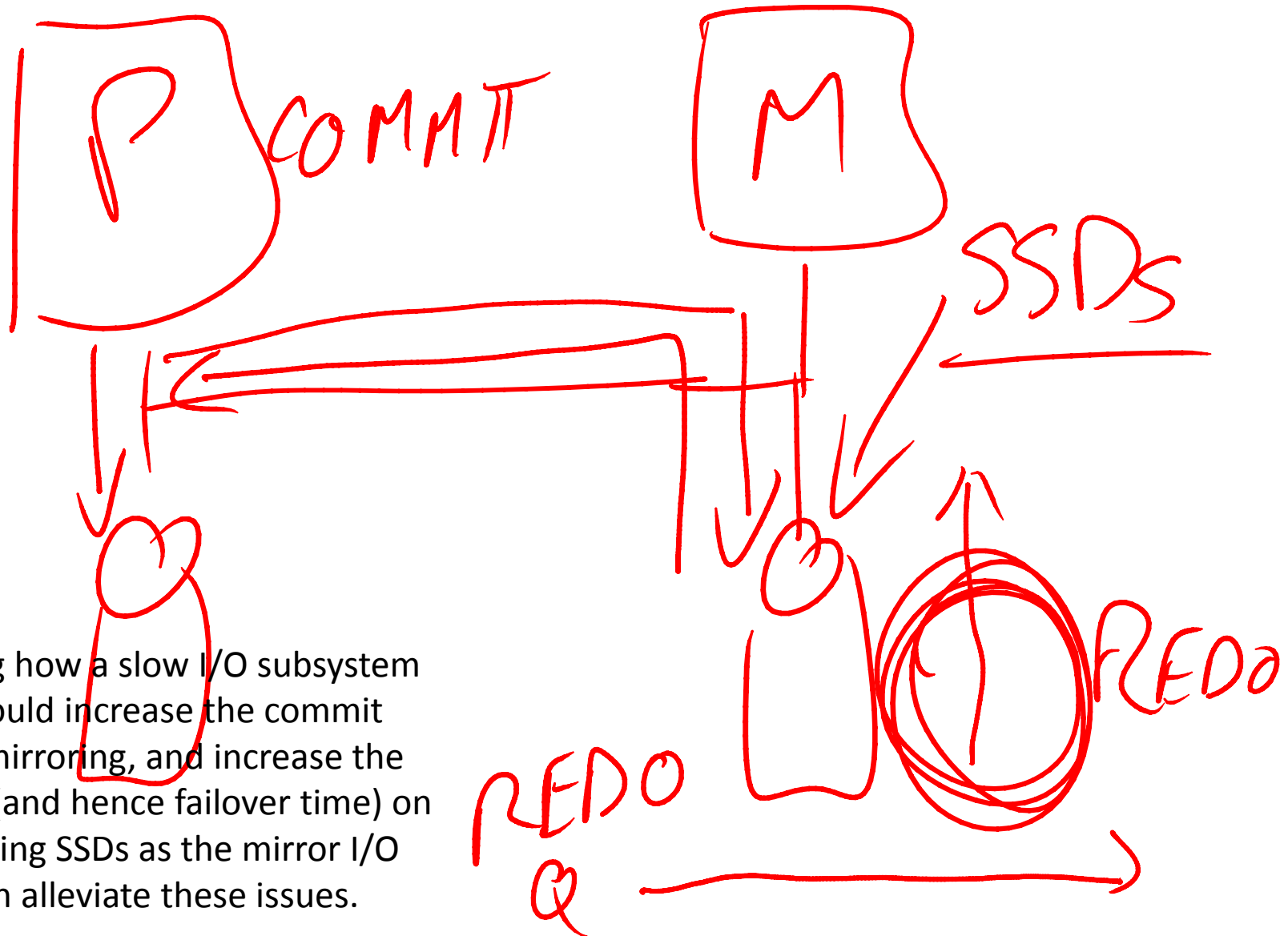


1 TB SALES

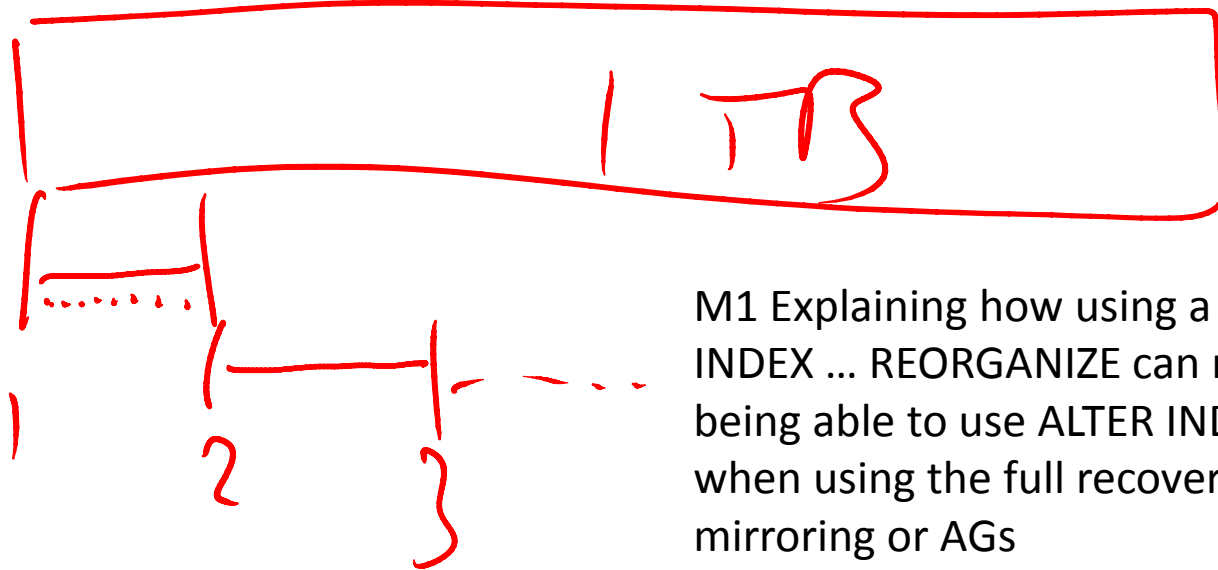
2013 - - - - - 2010

M1 Explaining how with a single data file, restore options are limited. With a database split into filegroups, partial restores are possible, as well as targeted piecemeal restores.

19 | 2013 | 2012 | | 2010



M1 explaining how a slow I/O subsystem on a mirror could increase the commit time in sync mirroring, and increase the REDO queue (and hence failover time) on the mirror. Using SSDs as the mirror I/O subsystem can alleviate these issues.

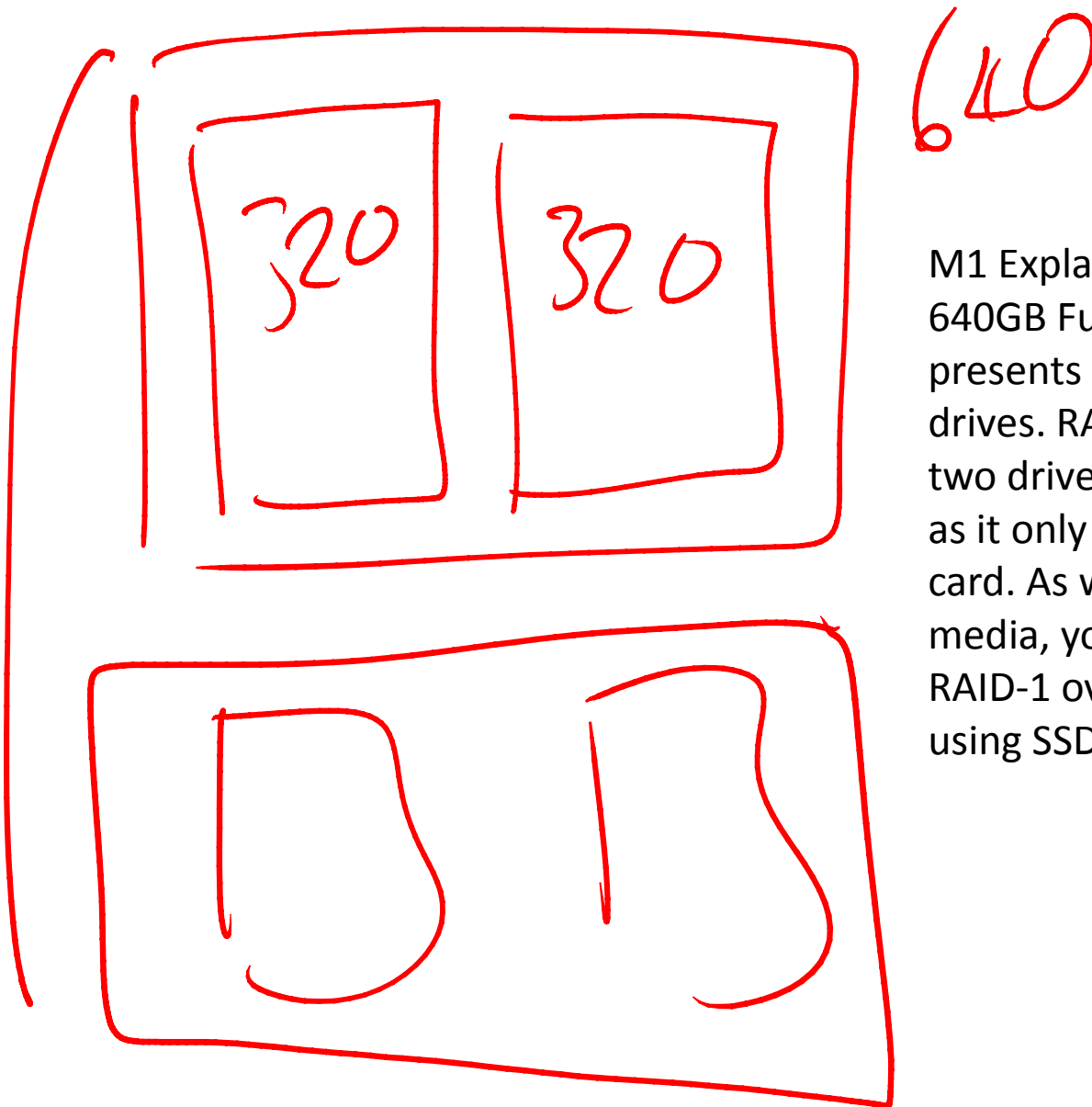


M1 Explaining how using a staggered ALTER INDEX ... REORGANIZE can replace not being able to use ALTER INDEX ... REBUILD when using the full recovery model with mirroring or AGs

ALTER INDEX
REORGANIZE

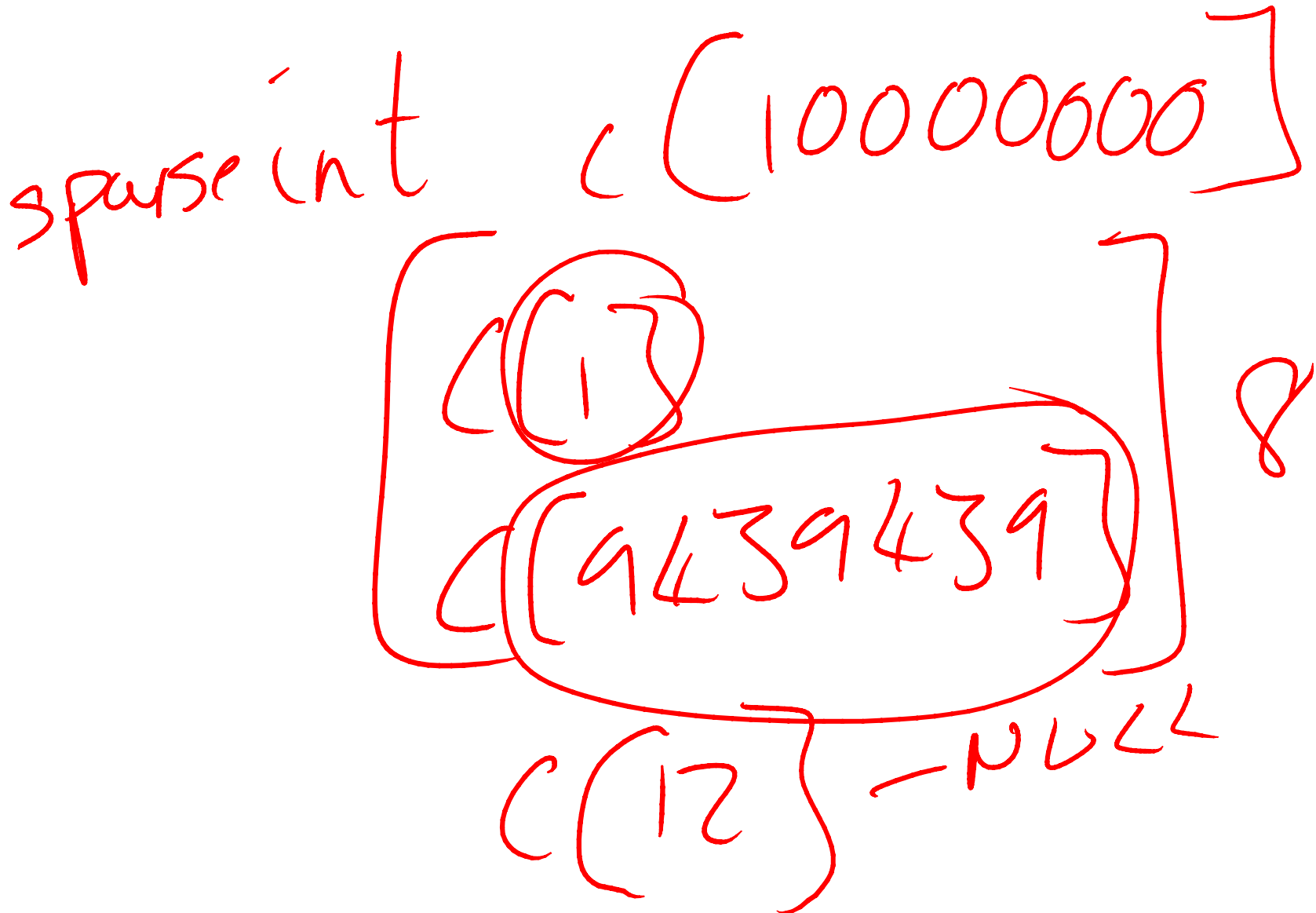
sys.dm_db_persisted_sku_features

M1 sys.dm_db_persisted_sku_features will let you know if you have any features in use that prevent the database being restored on non-Enterprise Editions. All such features except data compression require SA. Data compression only requires 'table owner'.



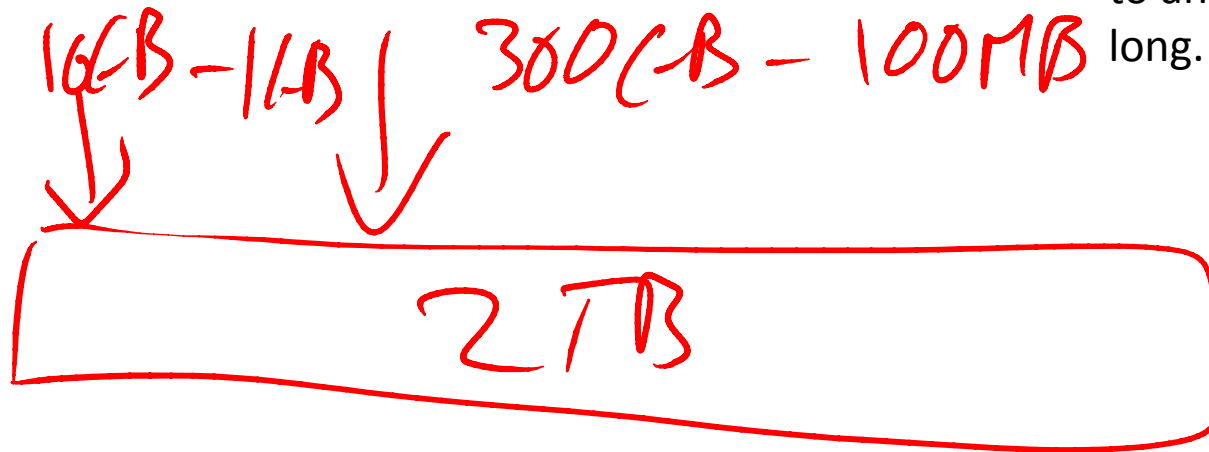
M1 Explaining how a 640GB Fusion-io drive presents itself as 2 x 320GB drives. RAID-1 over these two drives is really RAID-0, as it only uses one physical card. As with spinning media, you need to use RAID-1 over two cards if using SSDs.

M4 Drawing an analogy between sparse files in NTFS and sparse arrays in various programming languages. You can define an array of arbitrary size but only the populated elements take up memory. Any elements you ask for that haven't been populated result in an error.

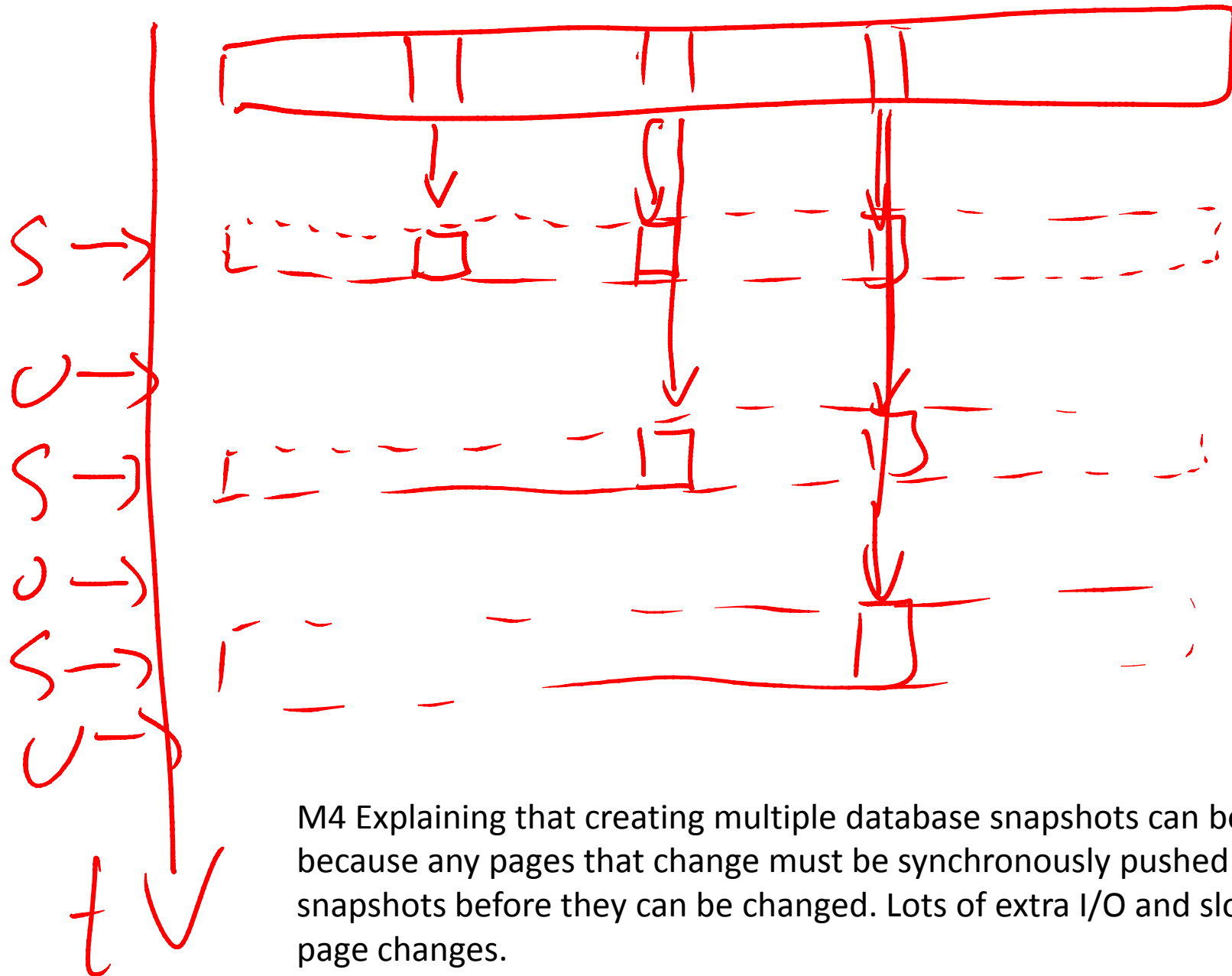


M4 Explaining that the NTFS sparse file keeps track of which file offsets have been written to and for how long.

SPARSE FILE

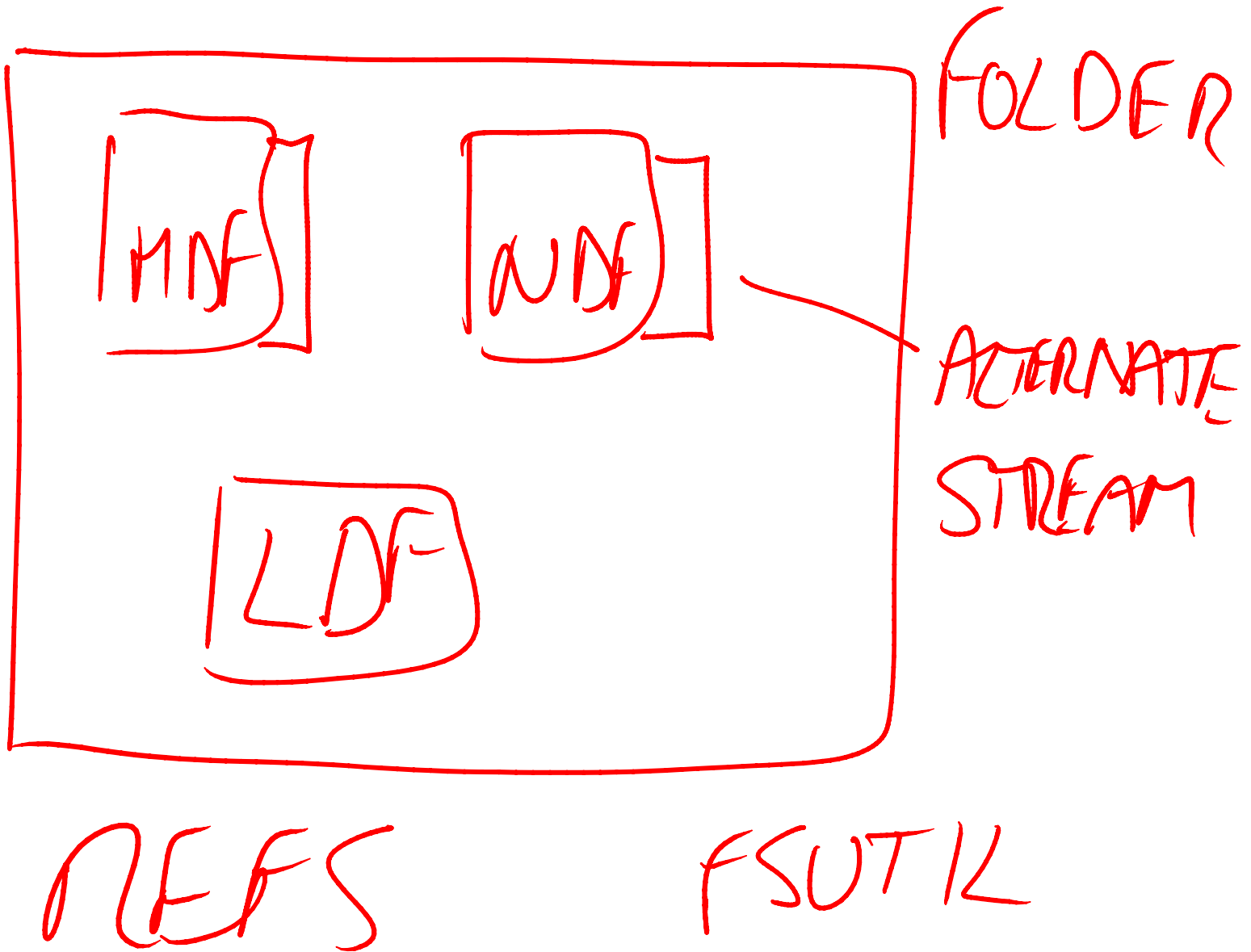


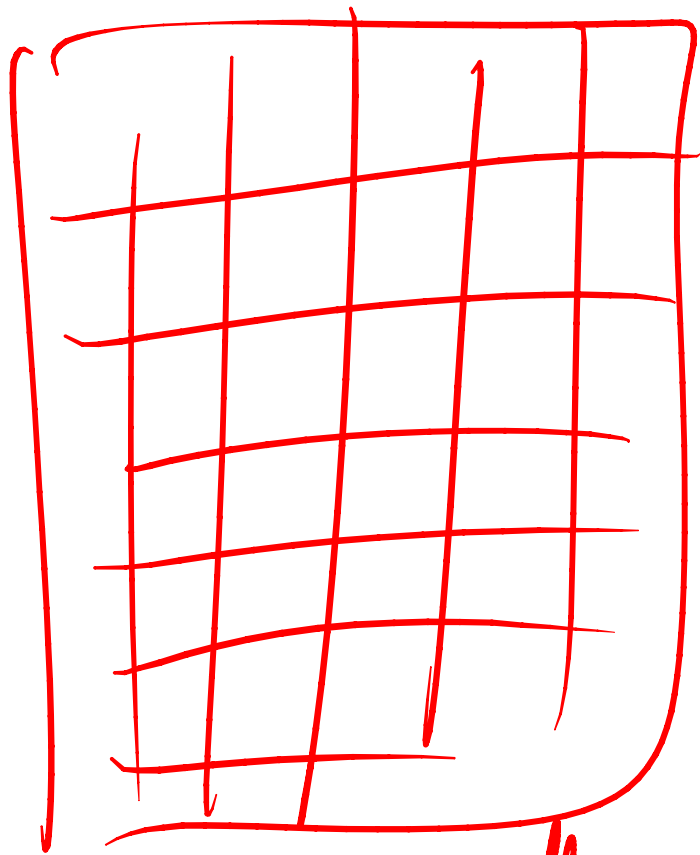
300GB 10GB
(64KB) (100MB) [1GB]



M4 Explaining that creating multiple database snapshots can be bad because any pages that change must be synchronously pushed into all snapshots before they can be changed. Lots of extra I/O and slows down page changes.

M10 and M5
Explaining that
NTFS alternate
streams that
make up the
hidden
snapshot are
stored in the
same NTFS
location as the
data files of
the database.
REFS does not
support sparse
files (yet). You
can see
alternate
streams using
fsutil.



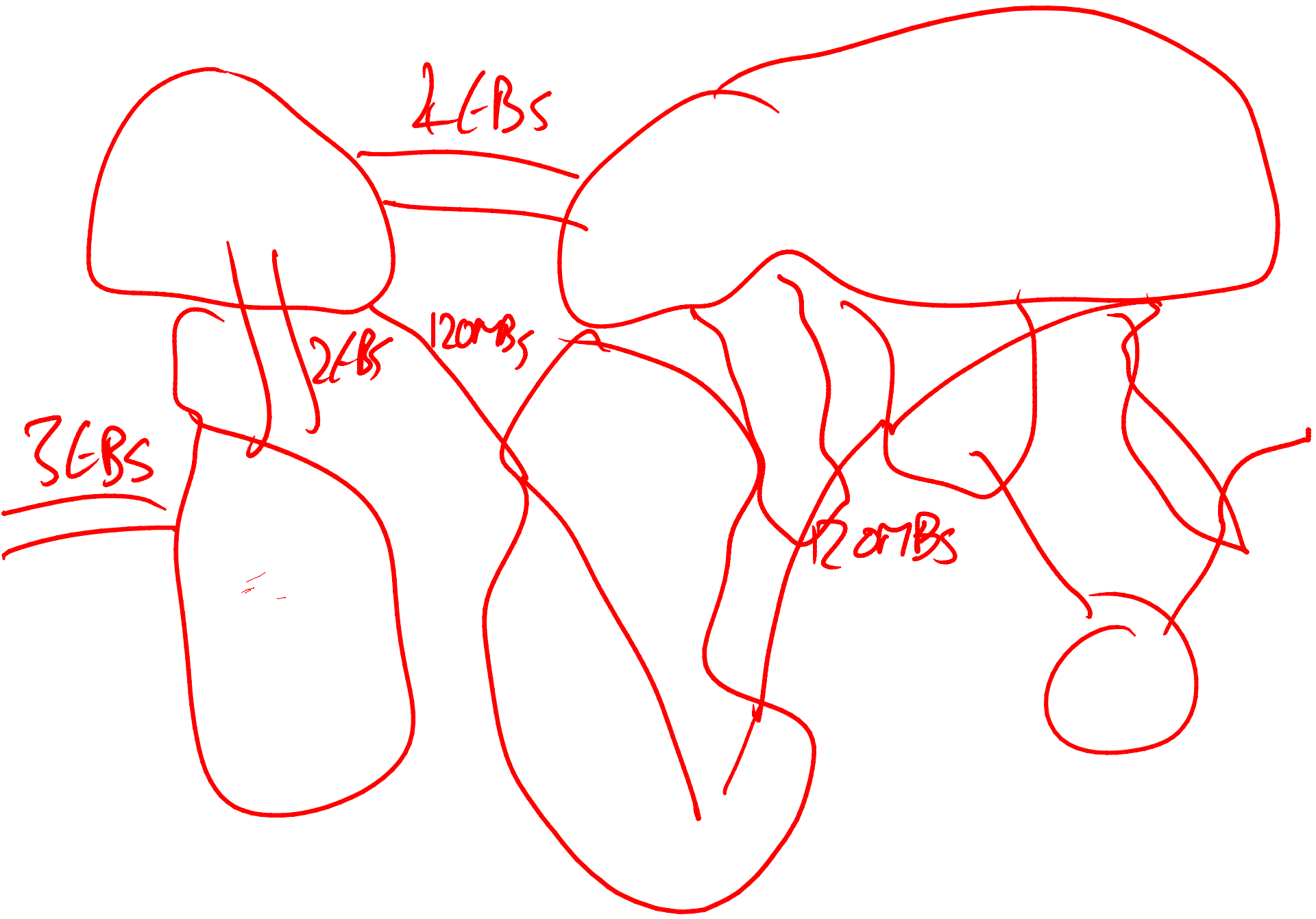


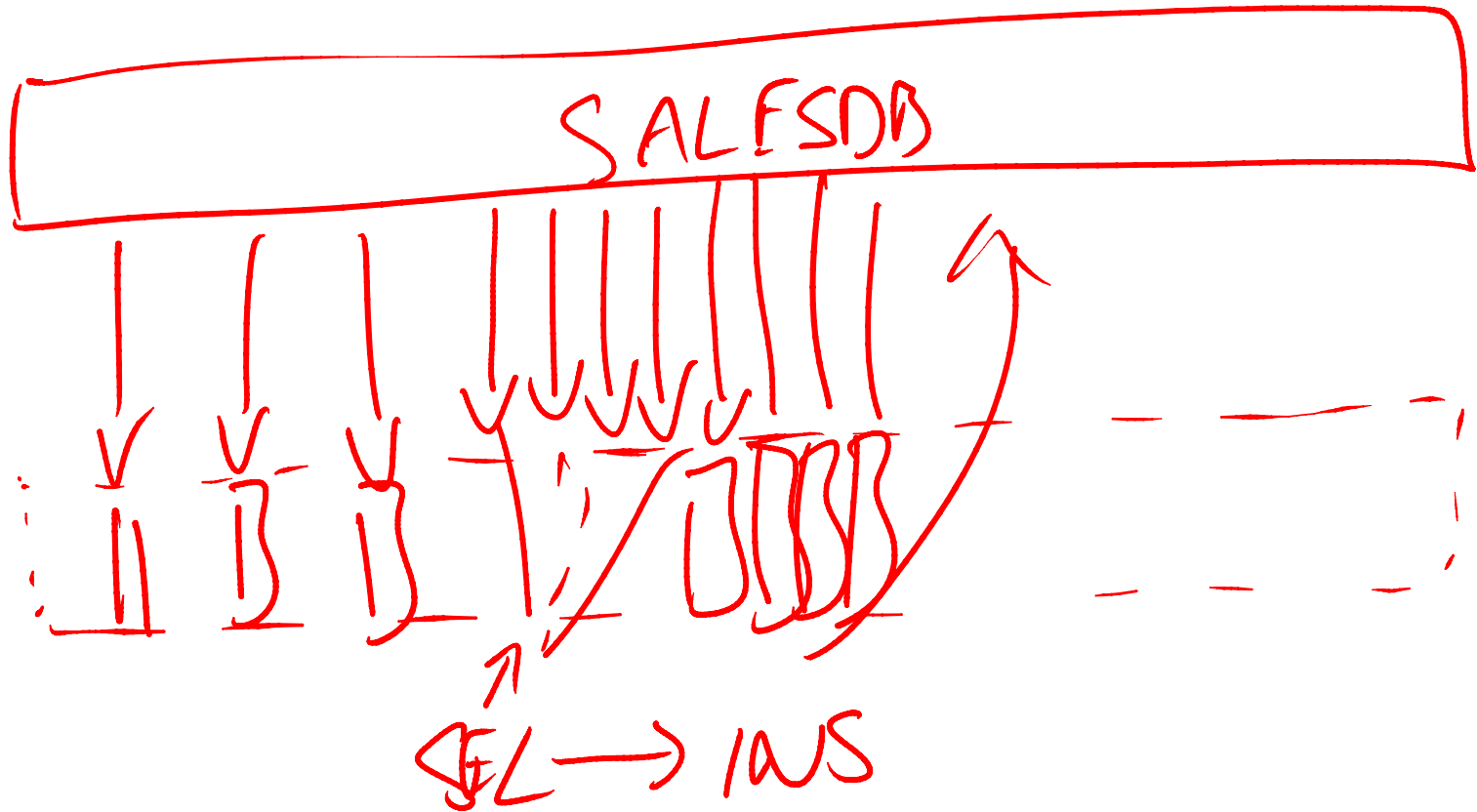
sys.dm_os_buffer_descriptors



M5 The buffer pool has hash lists per DB to track what's in memory. Use sys.dm_os_buffer_descriptors to see what's in memory

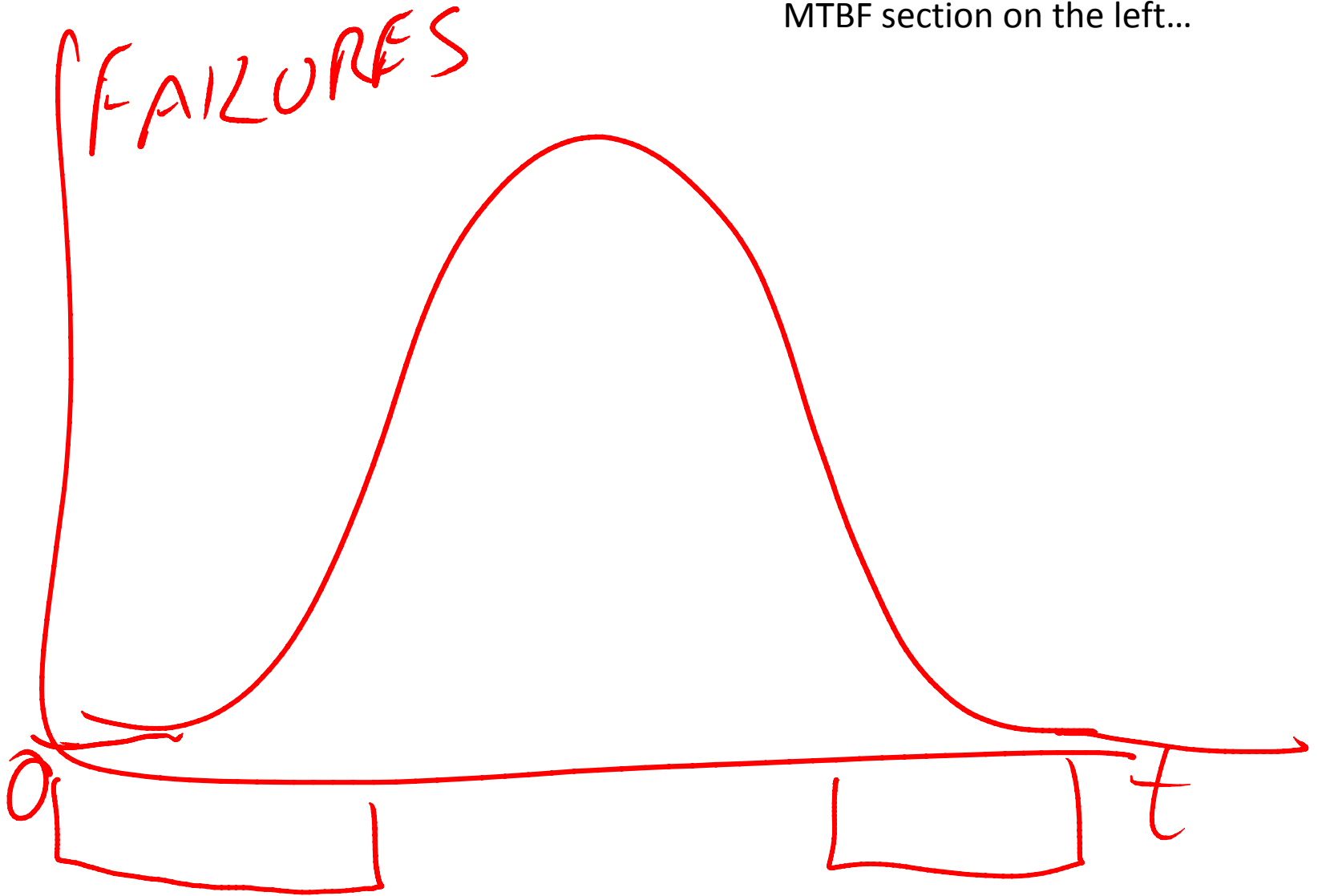
M5 My excellent drawing of the world and the data pipe bandwidths

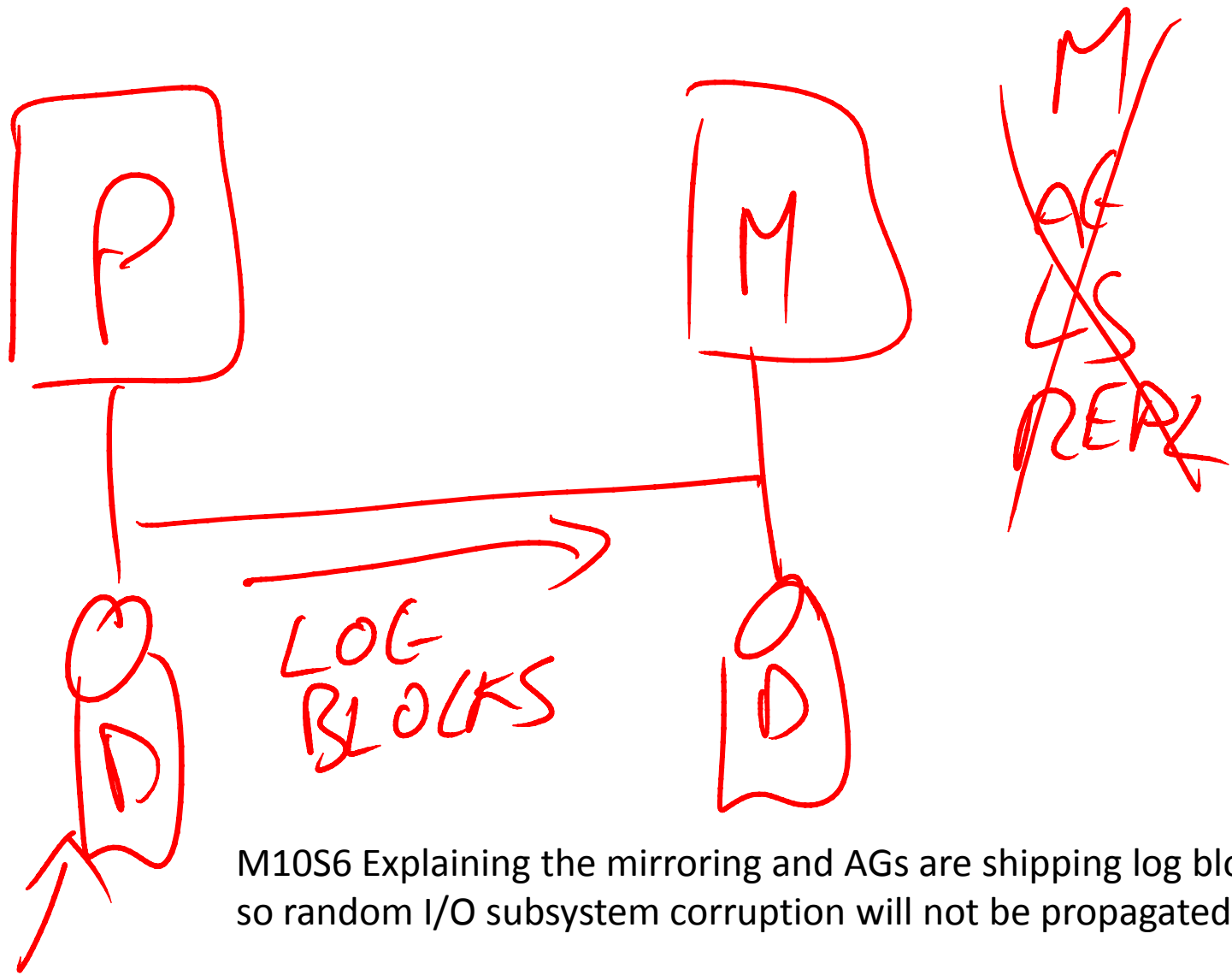


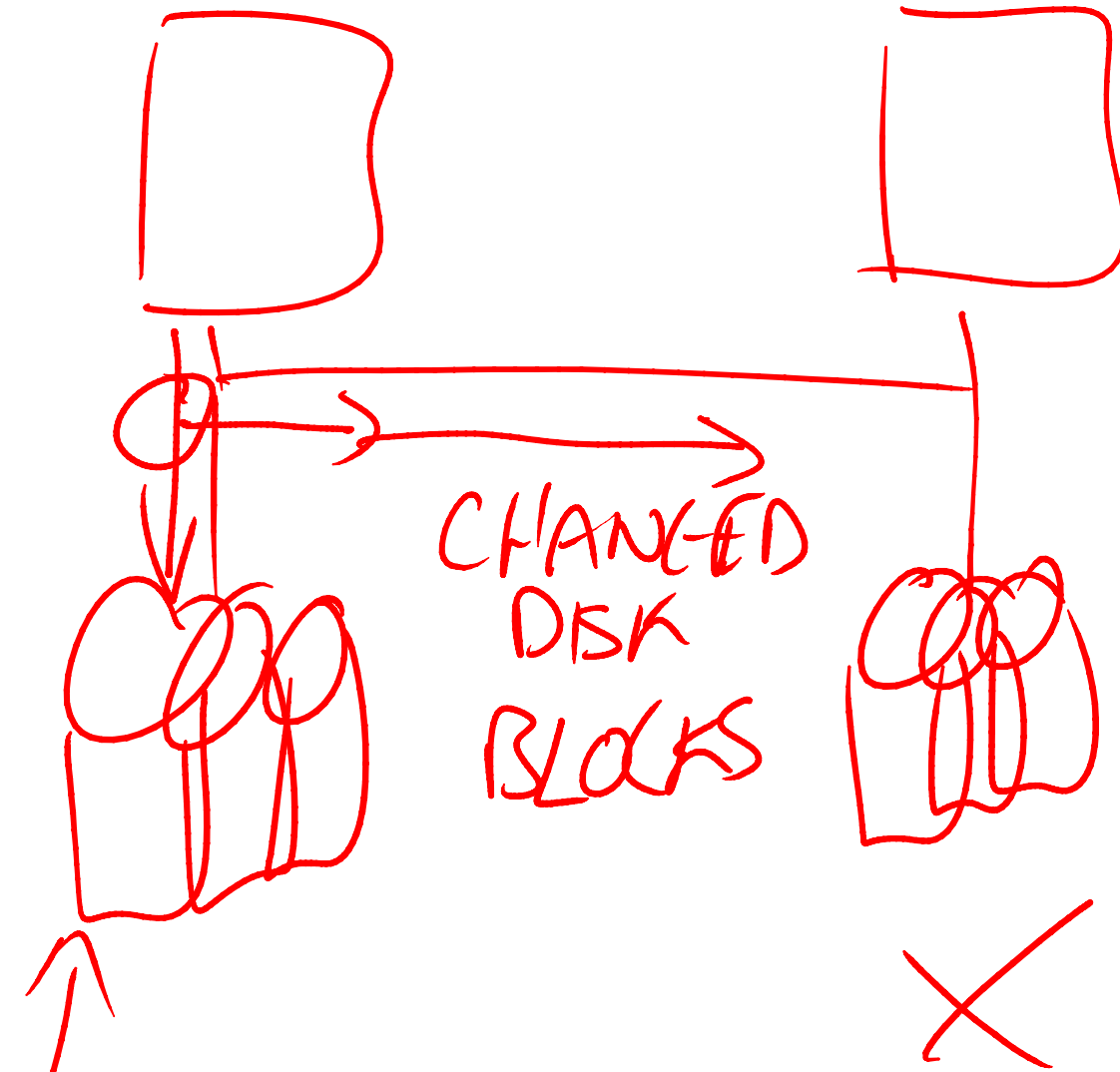


M5 Explaining the demo on recovering data from a snapshot. After dropping the table, the snapshot doesn't grow because the pages from the dropped table haven't been changed yet. As we repopulate the newly-created table though, it's pulling unchanged pages from the source database through the context of the snapshot to populate the new table, thus changing pages and pushing them into the snapshot. A bit of a mind-bender.

M10S6 The MTBF curve of drives.
Someone's going to be in the low
MTBF section on the left...







M10S6 Explaining how I/O subsystem-based replication on ships disk blocks changed by the host.



DROPCLEANBUFFERS (DBID)

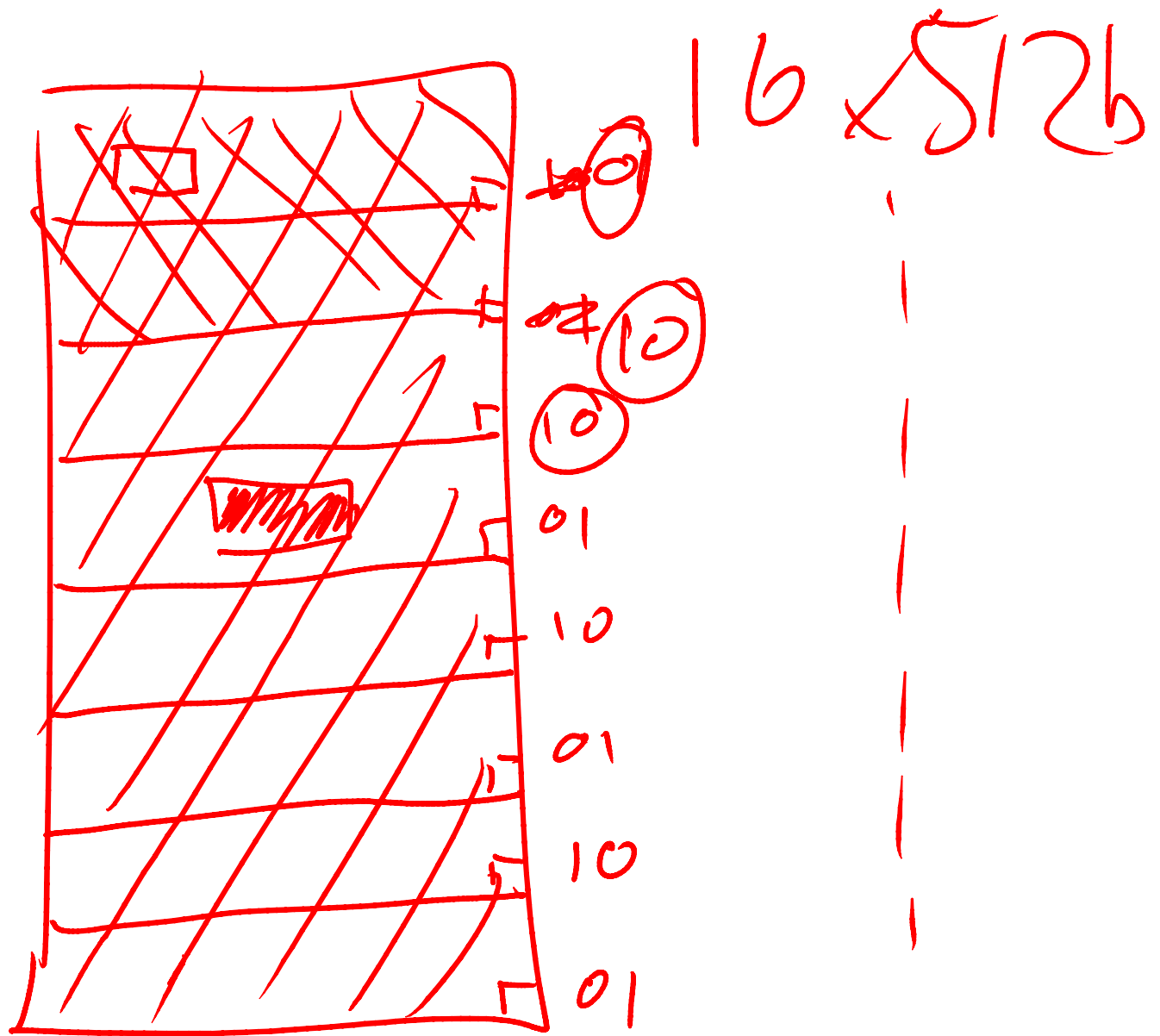
UNHOOKS FCB

CORRUPTION

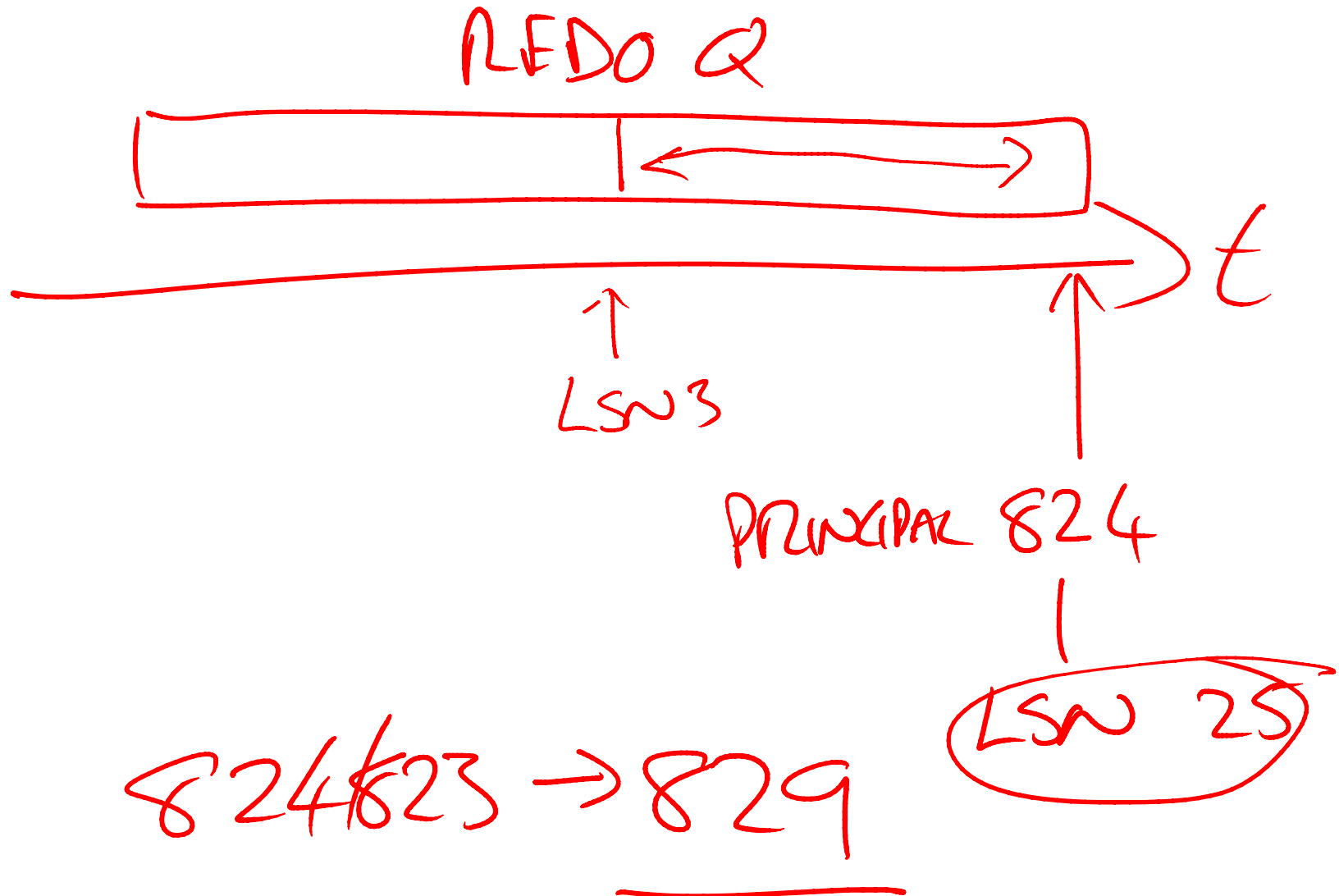
PUTS BACK FCB

M10 Explaining what the final parameter of DBCC
WRITEPAGE does. See my post for more details:

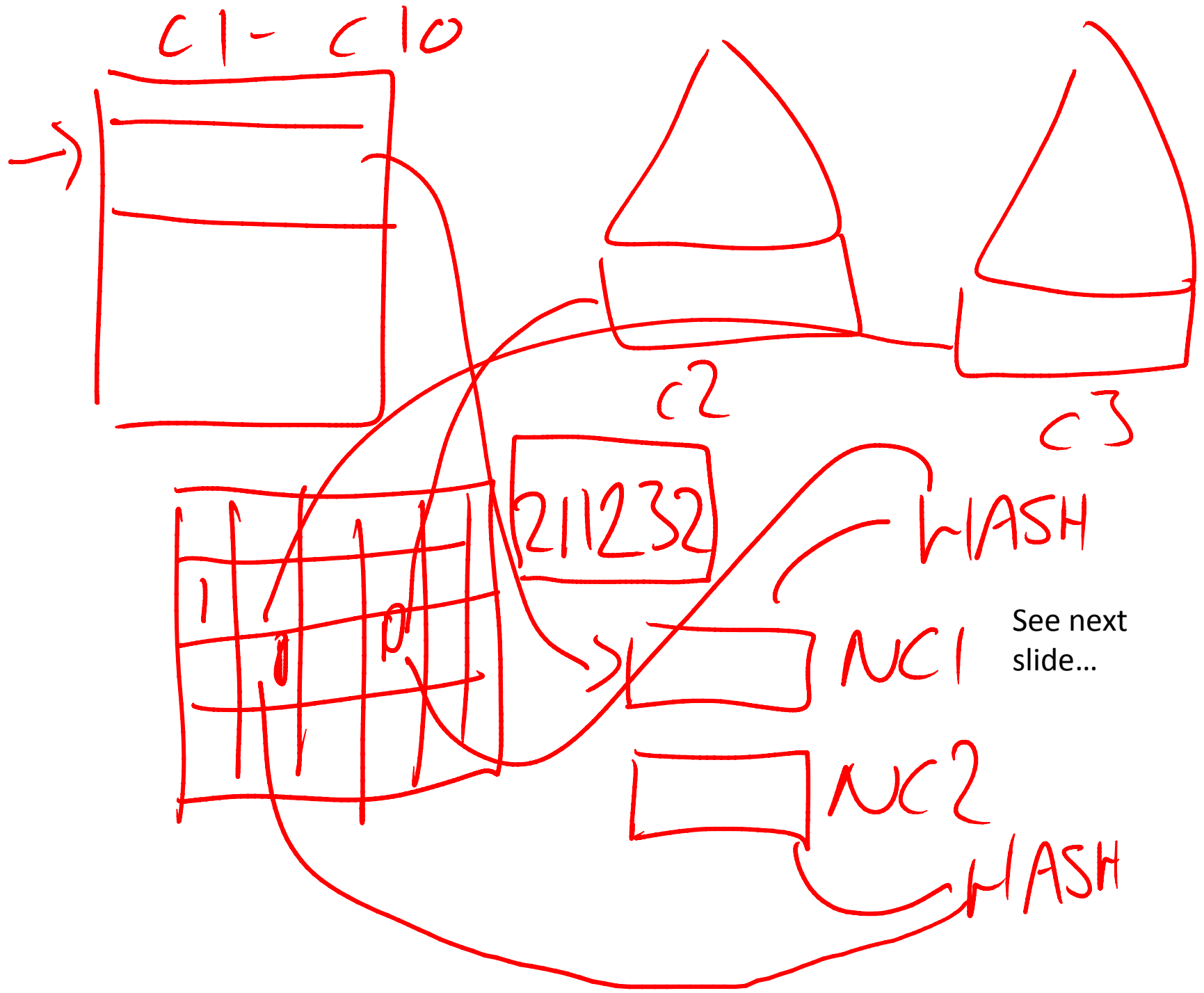
<http://www.sqlskills.com/blogs/paul/dbcc-writepage/>



M10S11 Explaining torn-page detection



M10S15 Explaining how automatic page repair in DBM may not be instant, because the mirror REDO queue needs to get to the LSN at which the principal asked for the damaged page, to avoid sending an older page image to the principal



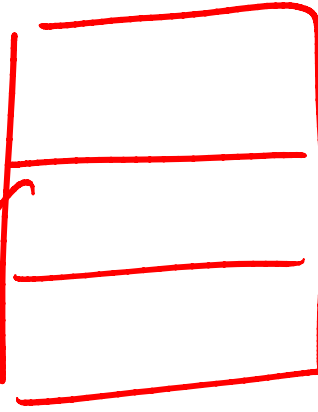
Previous slide:

M10 Explaining how the nonclustered index checking algorithm works. Each data record is used to construct all NC index records, then each is hashed to a value and a bit is flipped in a bitmap. Hashing the actual NC index records should map to the same bits, flipping the bits back again. Lossy algorithm, so any bits set at end mean a deep-dive to figure out which records are incorrect.

I AM
FACTS

See next
slide

P

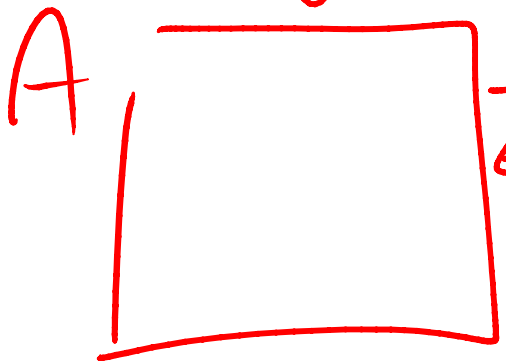


PACTUAL
PPARENT A
PPAR B
PPAR C

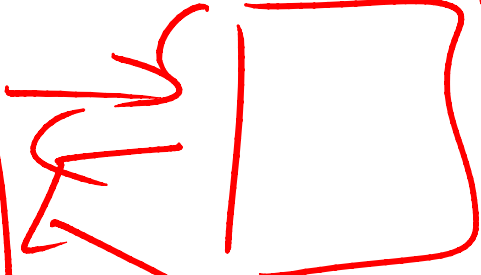
B



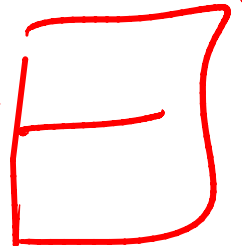
BACTUAL
BSIBLING C



BACTUAL
PPAR B
ASIBLB



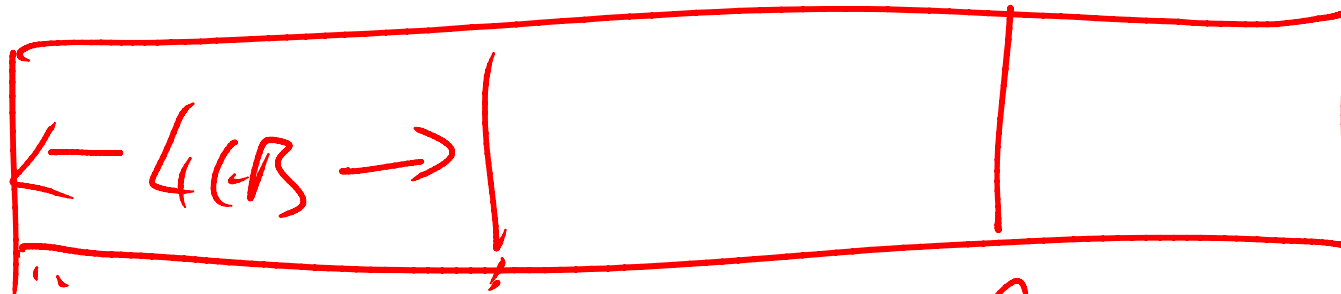
Q
EACT
QPAR



Previous slide:

M10 Explaining how fact generation works. More details
at <http://bit.ly/QnglMj>.

Current algorithm is $O(n \log(n))$.



GAM INTERVAL
4GB

T1 ... IAM

T2 ... IAM

M10 Allocation checks will XOR all the IAM pages for a single 4GB GAM interval together to make sure no two tables have the same extents allocated to them.

XOR = $\emptyset$

@@ERROR $\neq$ 0

M10S34 When CHECKDB completes, if it finds errors, the SQL Agent job or SSMS will reflect that. Programmatically, it will complete successfully (unless a nasty error forced it to stop) and so TRY ... CATCH will not work. If it found no errors, @@ERROR will be 0. If it found errors, @@ERROR will be non-zero.