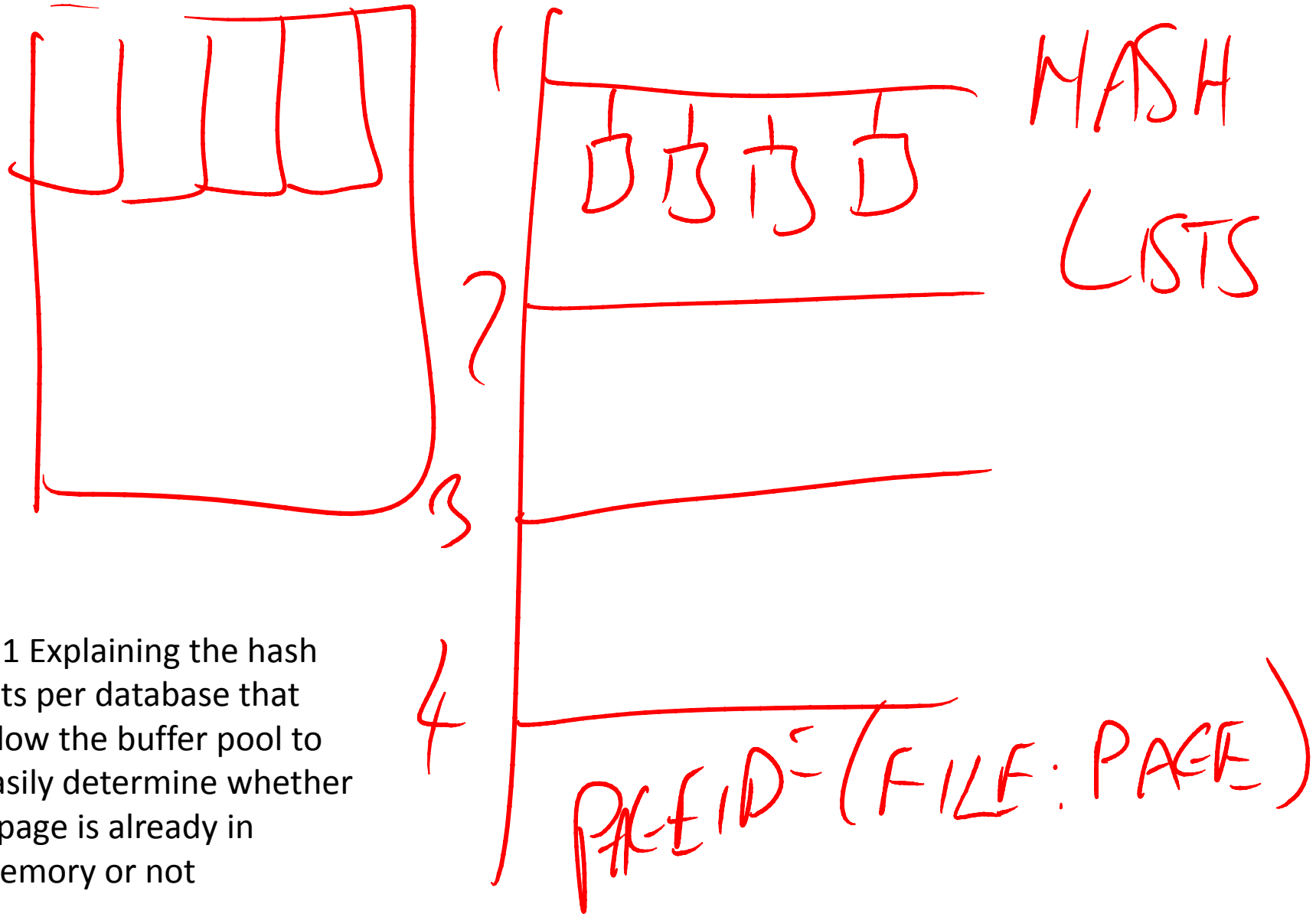
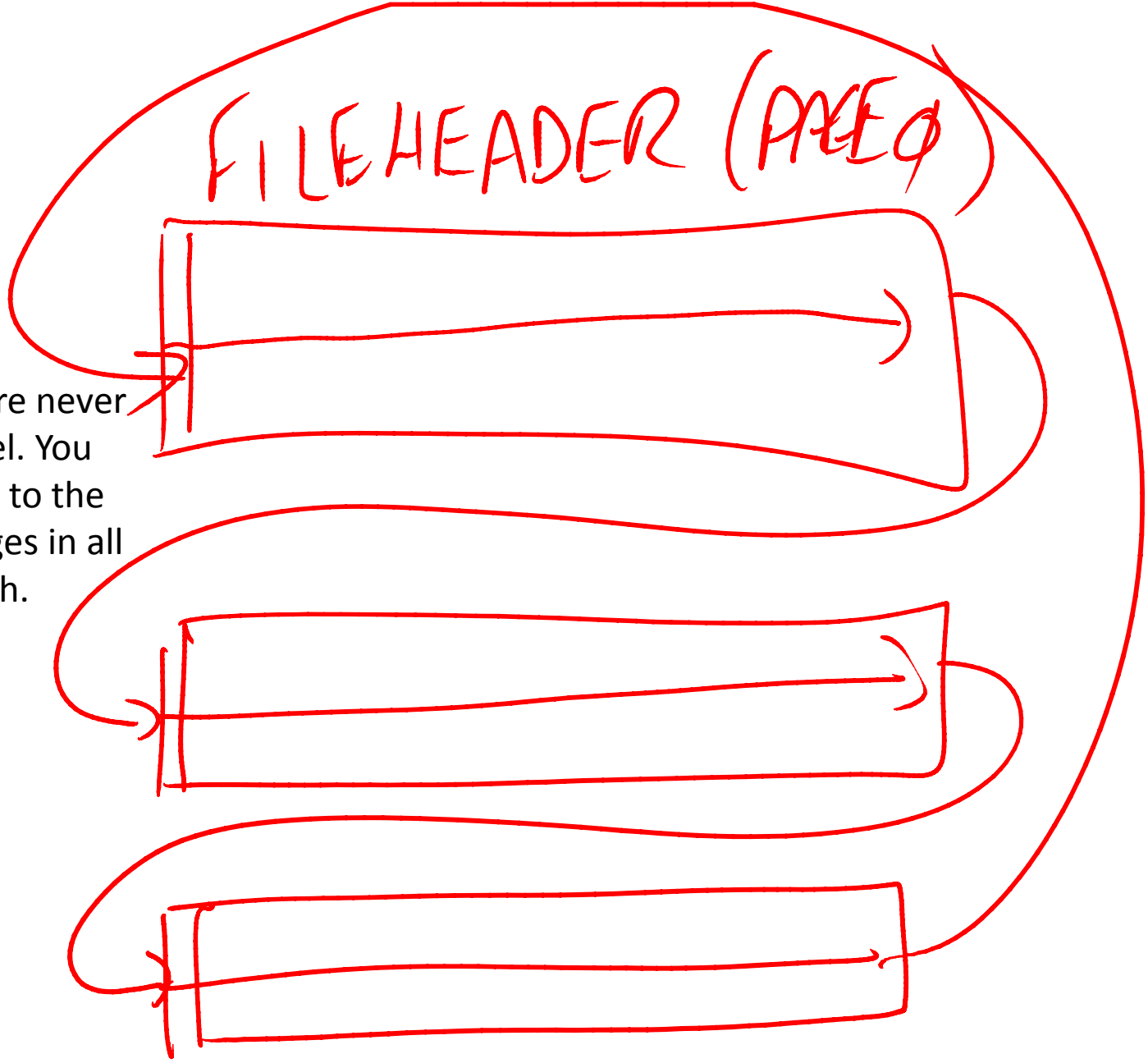
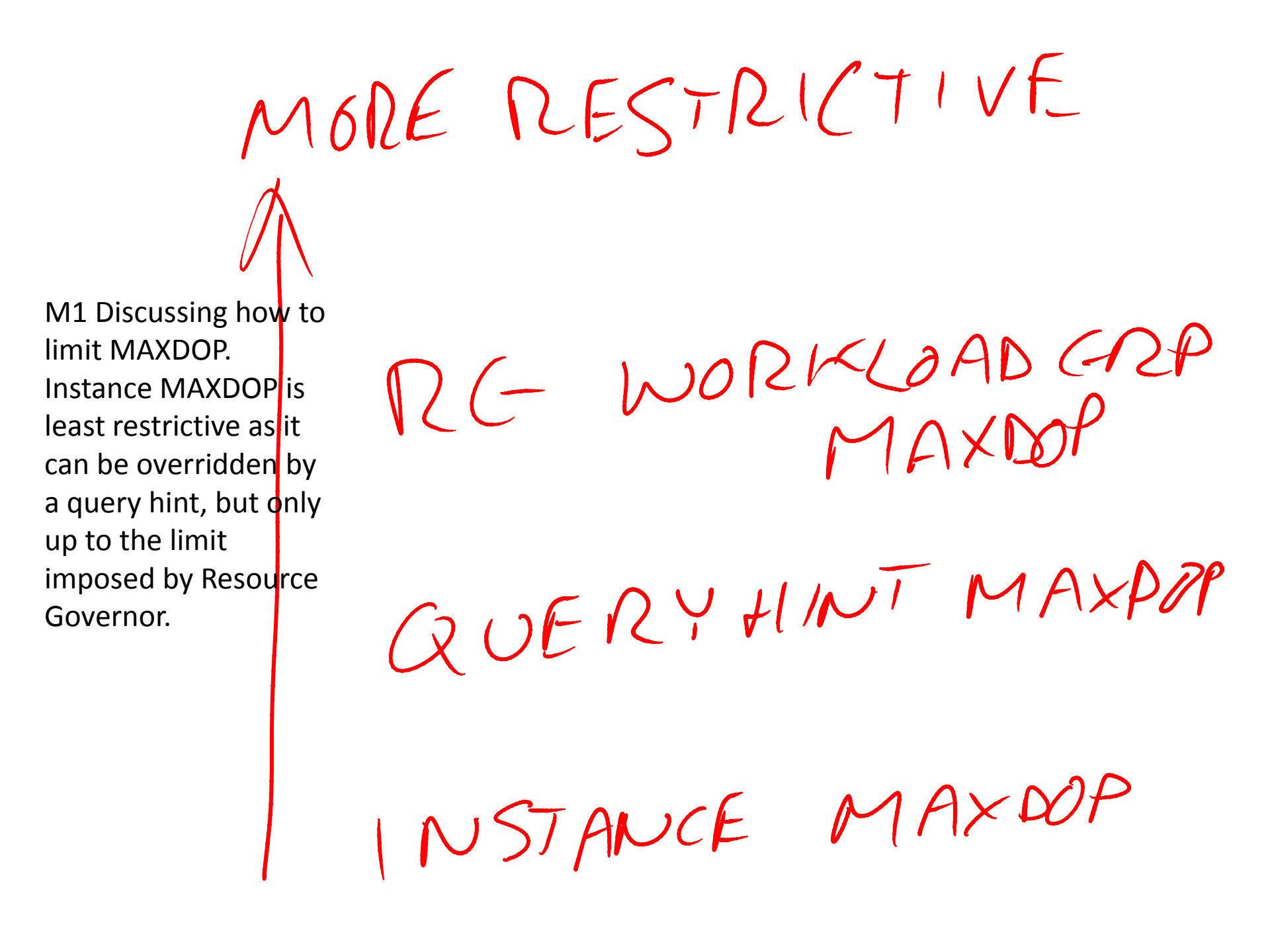


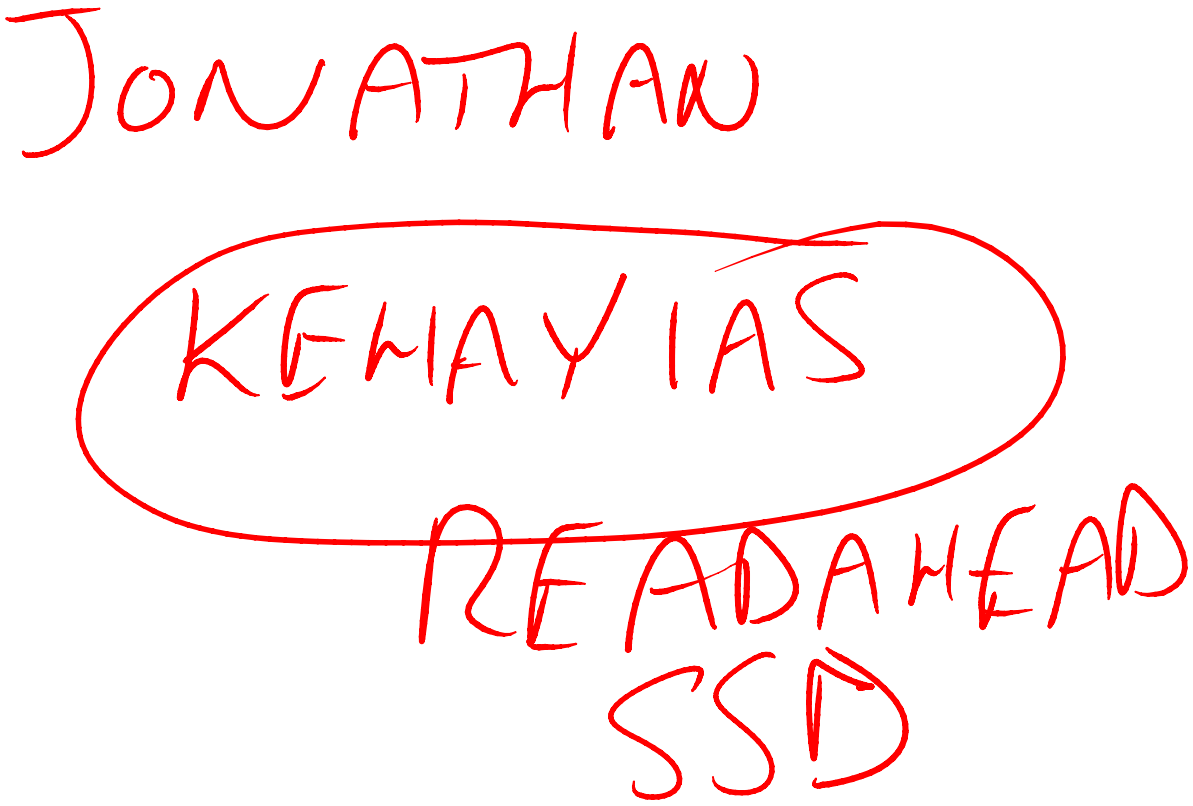
M1 Explaining the hash lists per database that allow the buffer pool to easily determine whether a page is already in memory or not

M1 Log files are never used in parallel. You will see writes to the fileheader pages in all log files though.

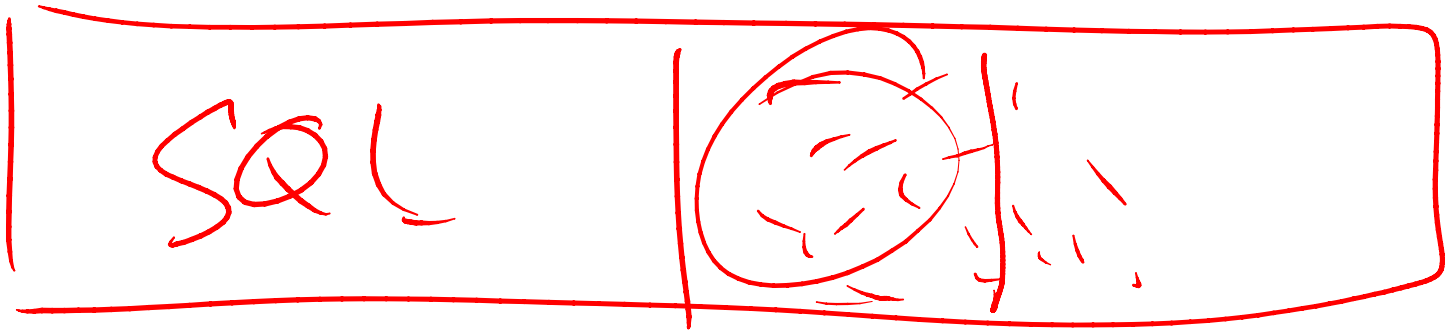




M1 Discussing how to limit MAXDOP. Instance MAXDOP is least restrictive as it can be overridden by a query hint, but only up to the limit imposed by Resource Governor.



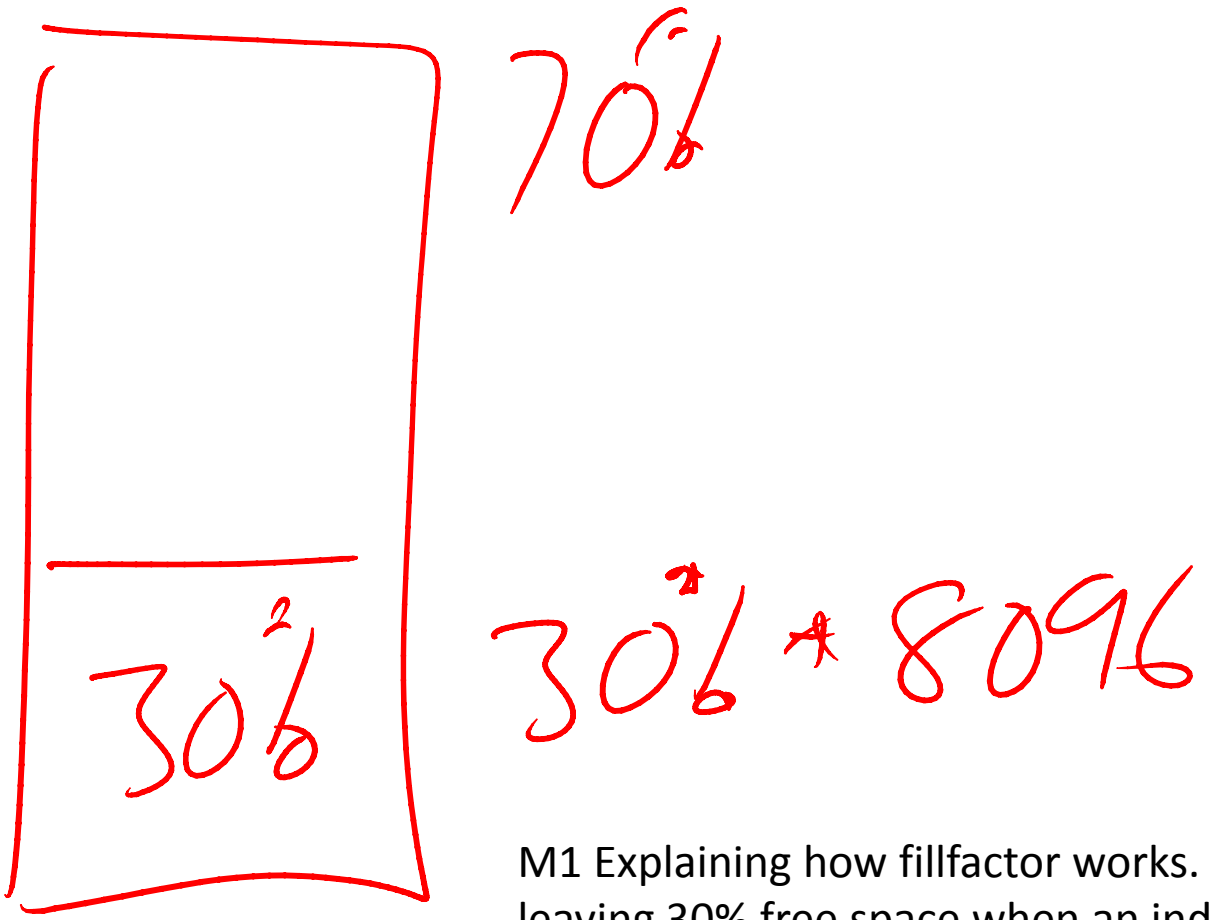
M1S13 Jonathan has a blog post that explains why you still need to care about index fragmentation when using SSDs. See http://bit.ly/klq1JF



SA

DBCC PAGE

M1S14 Explaining why instant file initialization is off by default. Sharing a volume between SQL Server and ‘secure files’ can lead to information leakage if the secure files are deleted, then SQL grows a data file, and an SA can use DBCC PAGE to get a hex dump of the bits/bytes that comprised the deleted secure file. If you don’t have this situation, turn instant file initialization on.

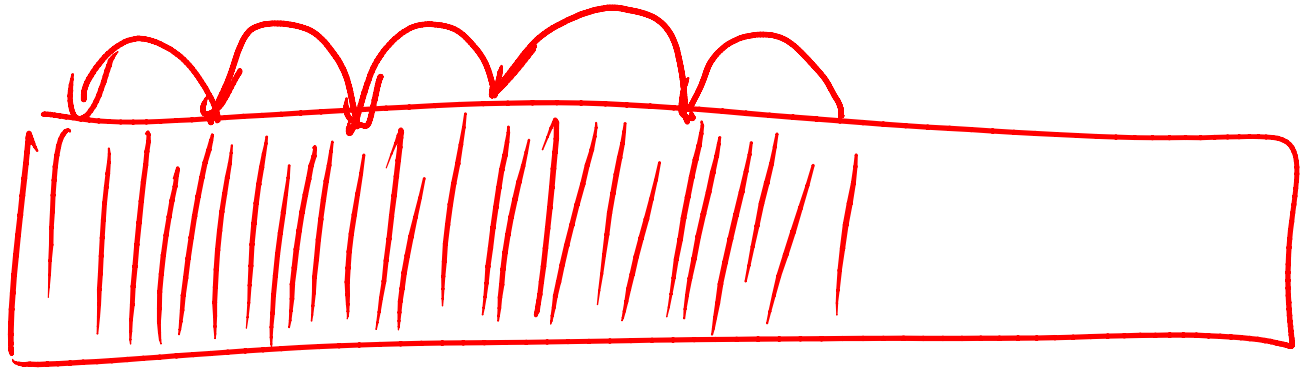


M1 Explaining how fillfactor works. Fillfactor = 70 means leaving 30% free space when an index is created or rebuilt. Trade-off of fewer page splits (i.e. more logging) against extra space.

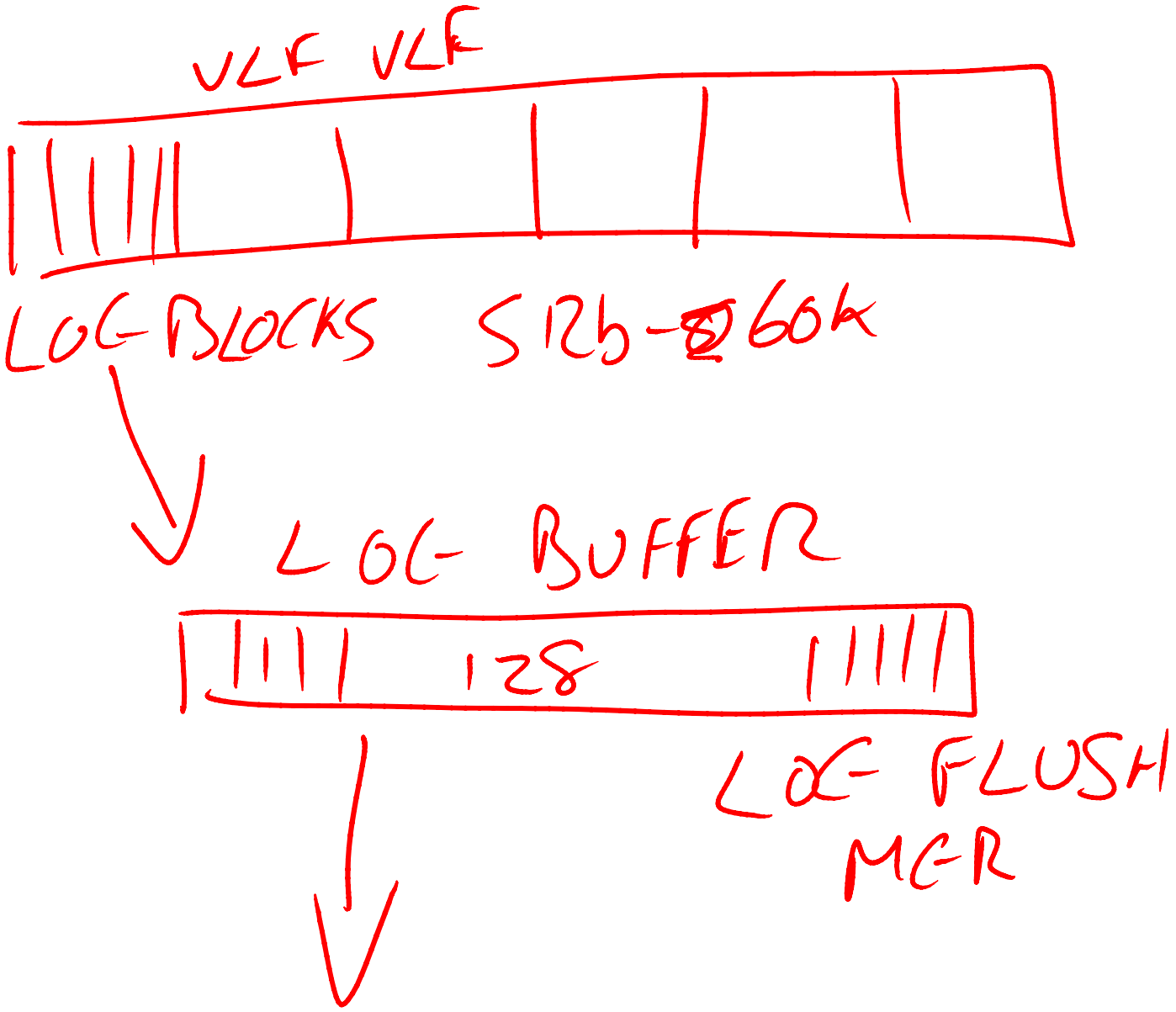
TF 3605

TF 3004

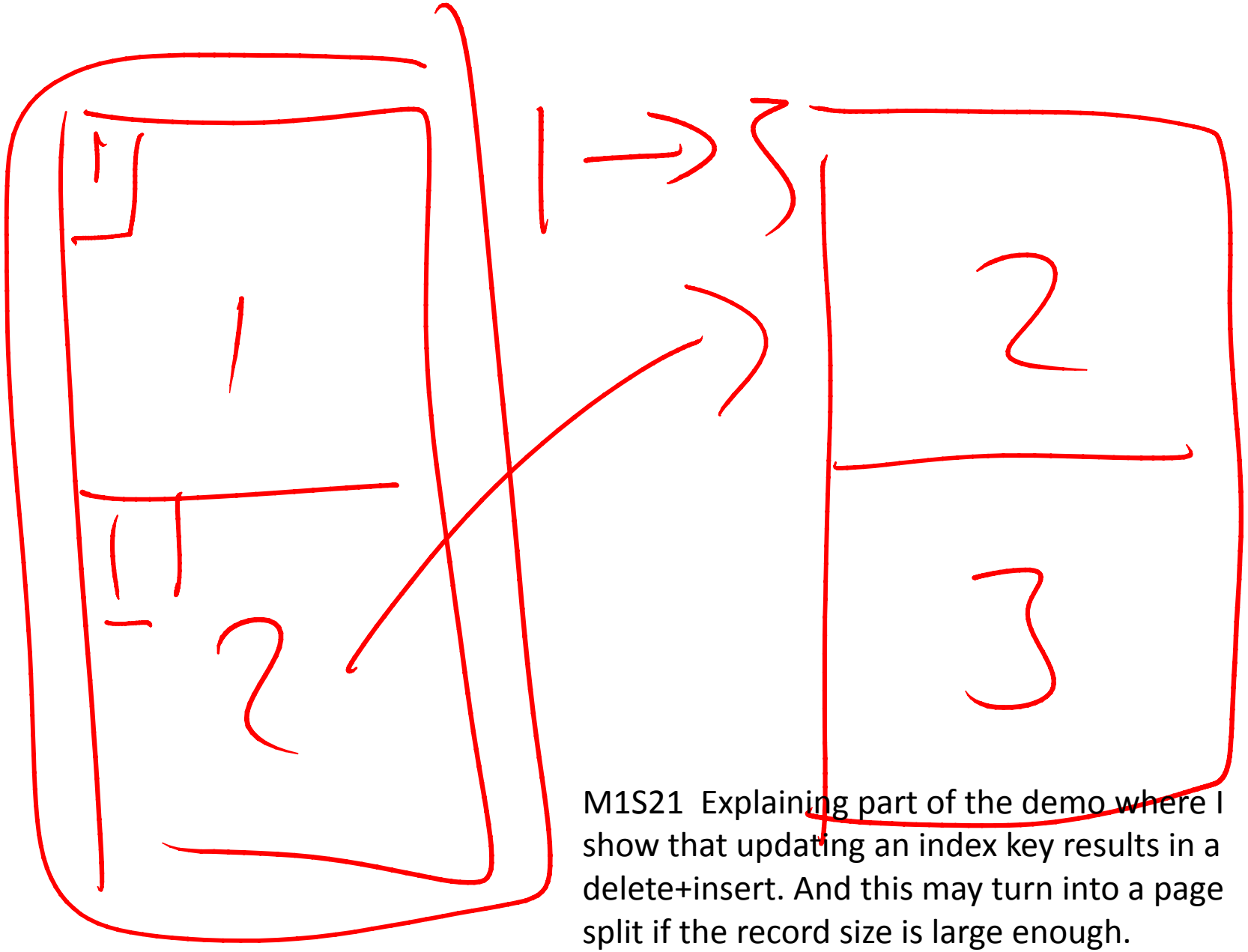
M1S14 How to check if you have instant file initialization enabled if you don't have box access to check for the service account having Perform Volume Maintenance Tasks priv. Turn on these trace flags and create a dummy database. Look in the error log. With instant file initialization you'll see one line – 'zeroing out the log file'. Without it, you'll also see 'zeroing out the data file'.



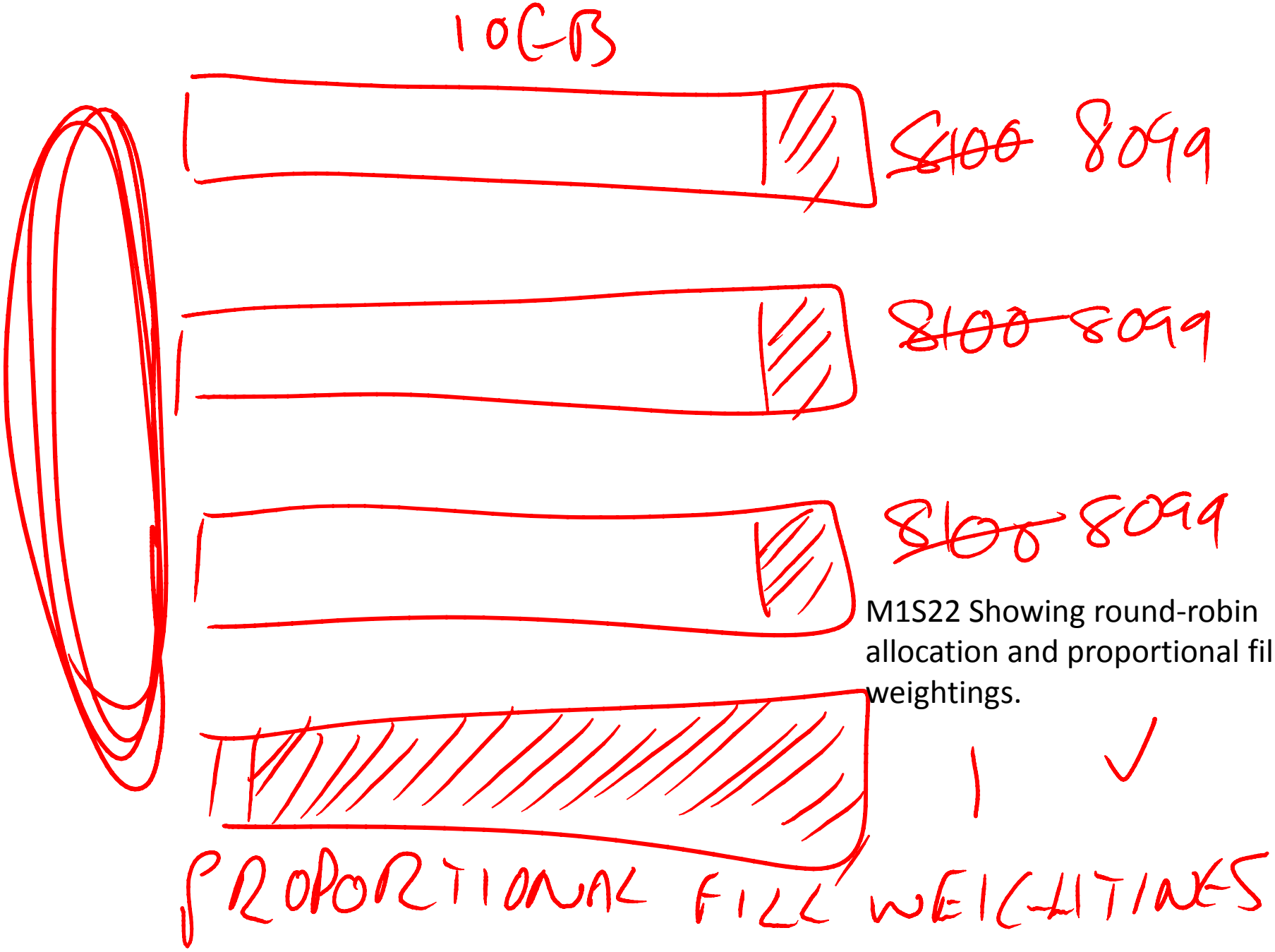
M1S18 Explaining how log records link back to the previous log record in the transaction to allow the transaction to be rolled back by generating compensating actions in reverse order. A compensation log record will link over the log record it is compensating for. This is what accounts for any random reads of the log.



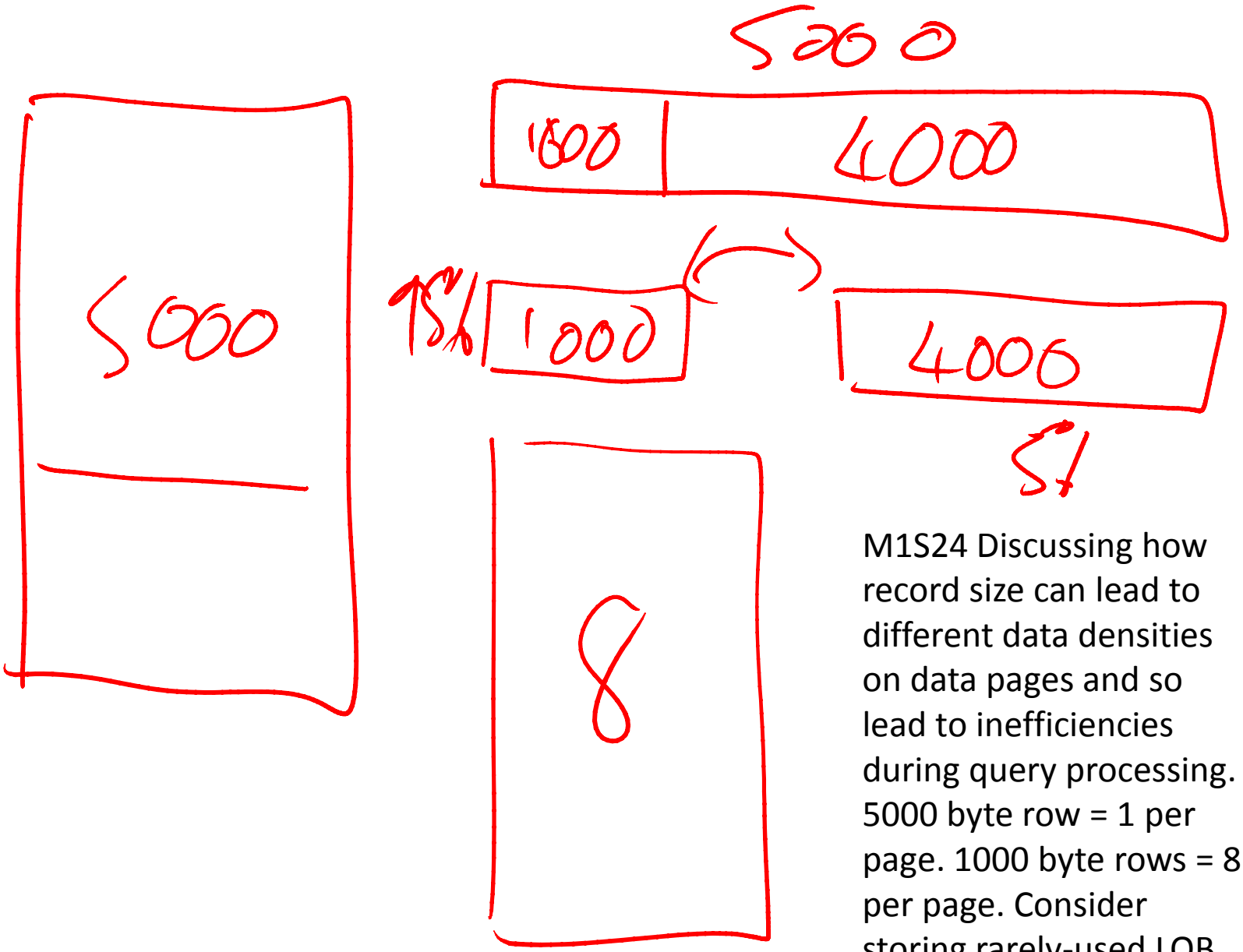
M1S18-19 Discussing how log records are flushed to disk by first being copied into a log buffer and then that buffer is flushed to disk.



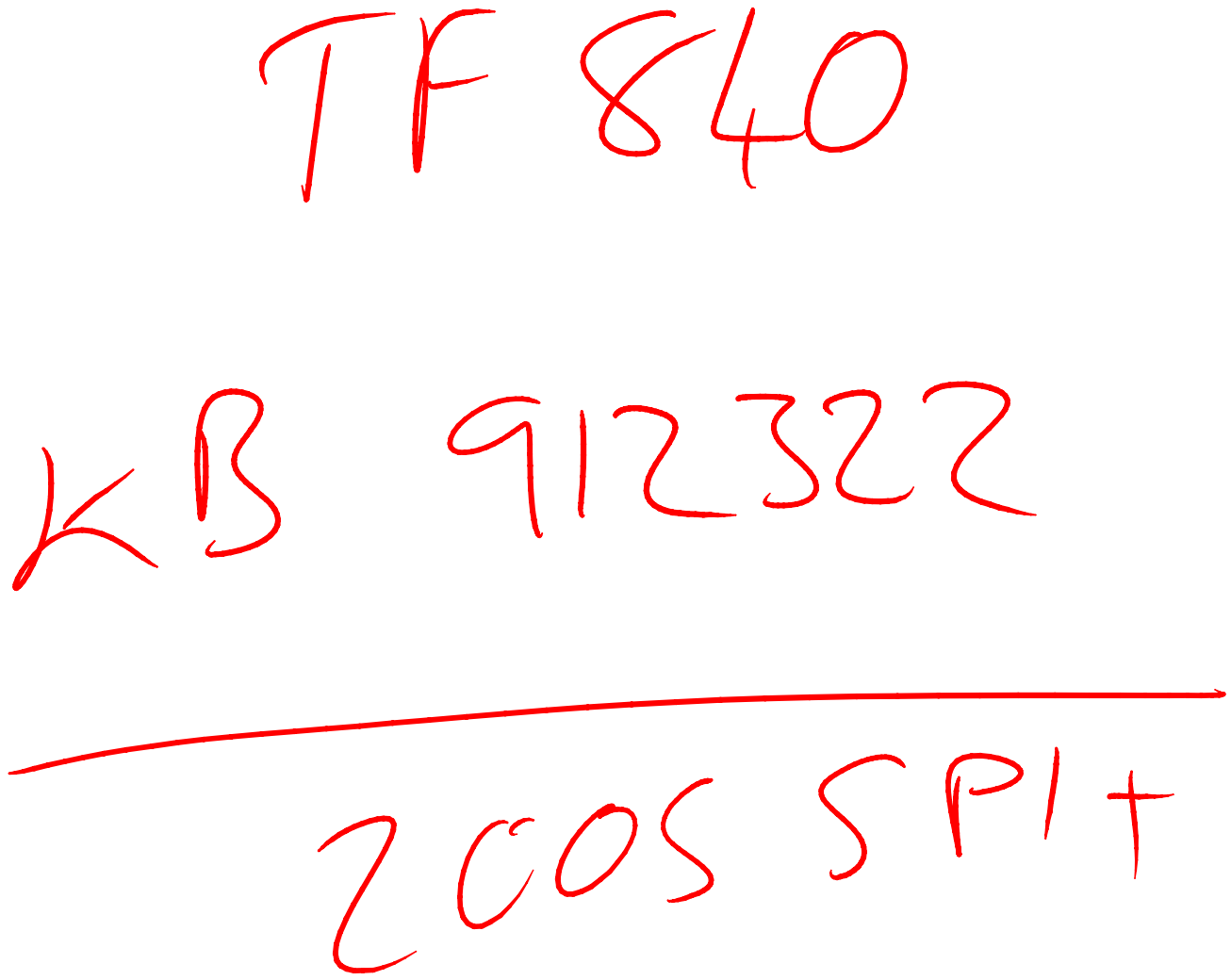
M1S21 Explaining part of the demo where I show that updating an index key results in a delete+insert. And this may turn into a page split if the record size is large enough.



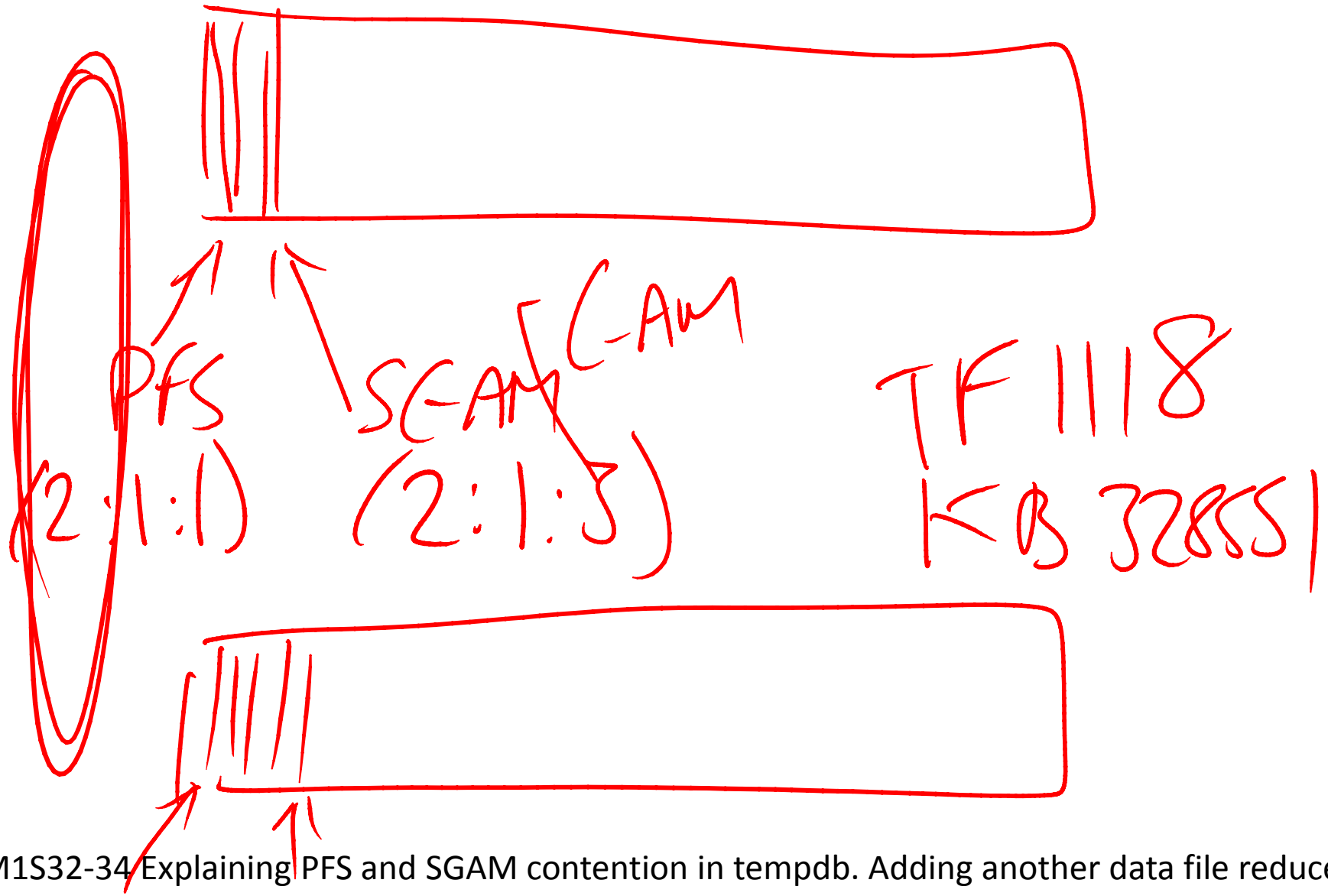
M1S22 Showing round-robin allocation and proportional fill weightings.



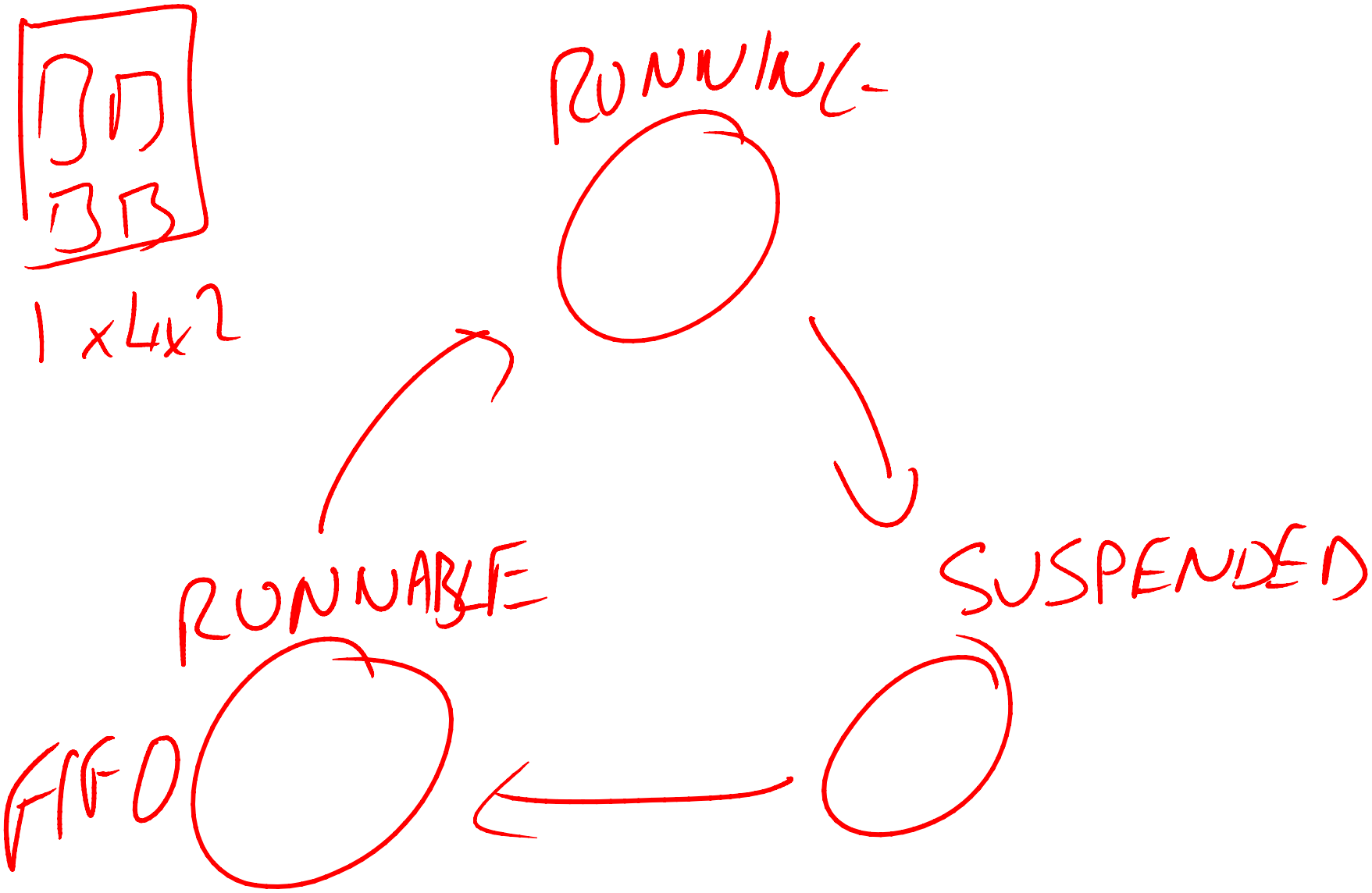
M1S24 Discussing how record size can lead to different data densities on data pages and so lead to inefficiencies during query processing. 5000 byte row = 1 per page. 1000 byte rows = 8 per page. Consider storing rarely-used LOB data off-row to increase data density.



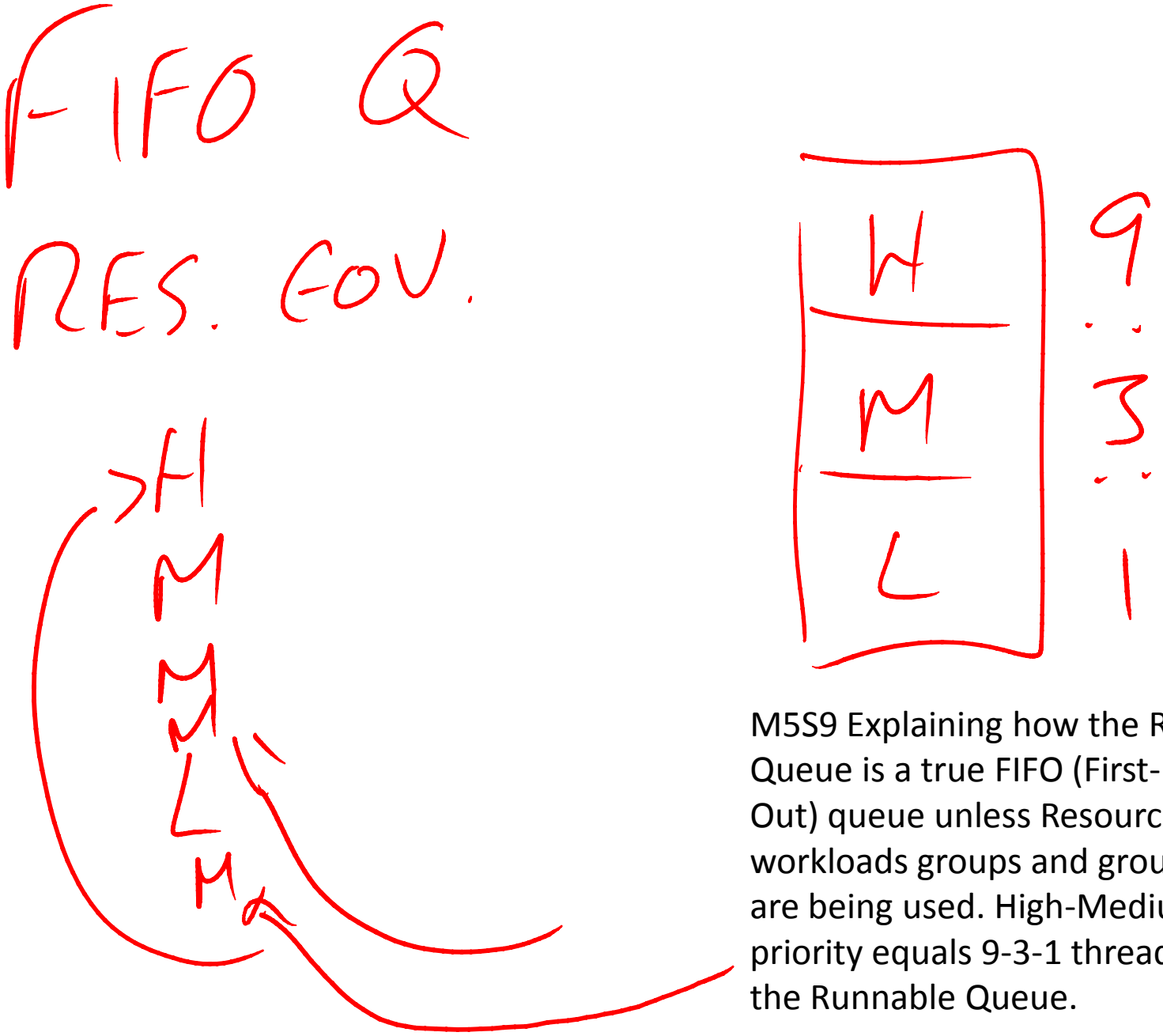
M1 Forcing buffer pool fast ramp-up after restart on non-Enterprise versions.



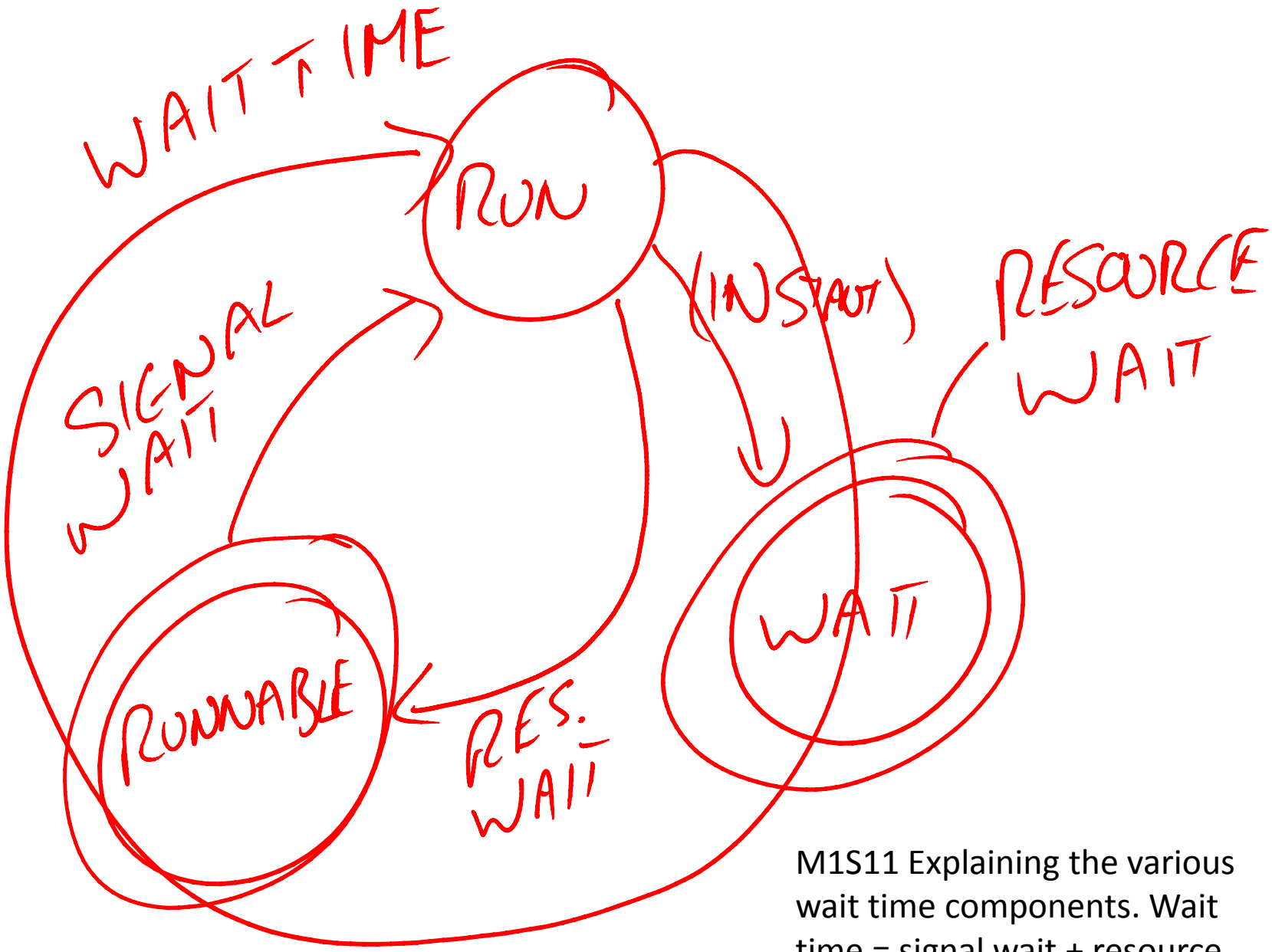
M1S32-34 Explaining PFS and SGAM contention in tempdb. Adding another data file reduces contention because of round-robin allocation.



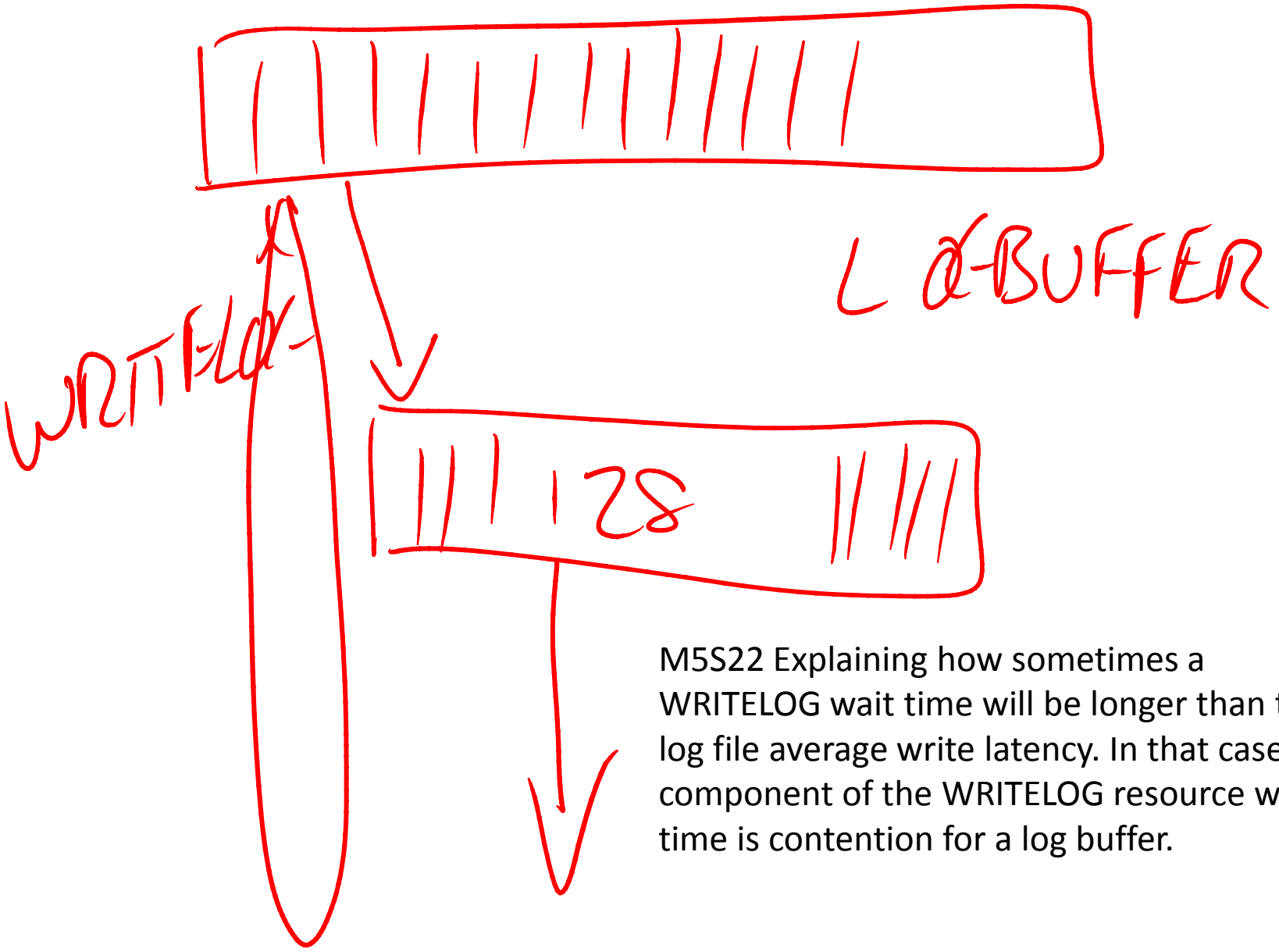
M5S6 Explaining the thread execution cycle



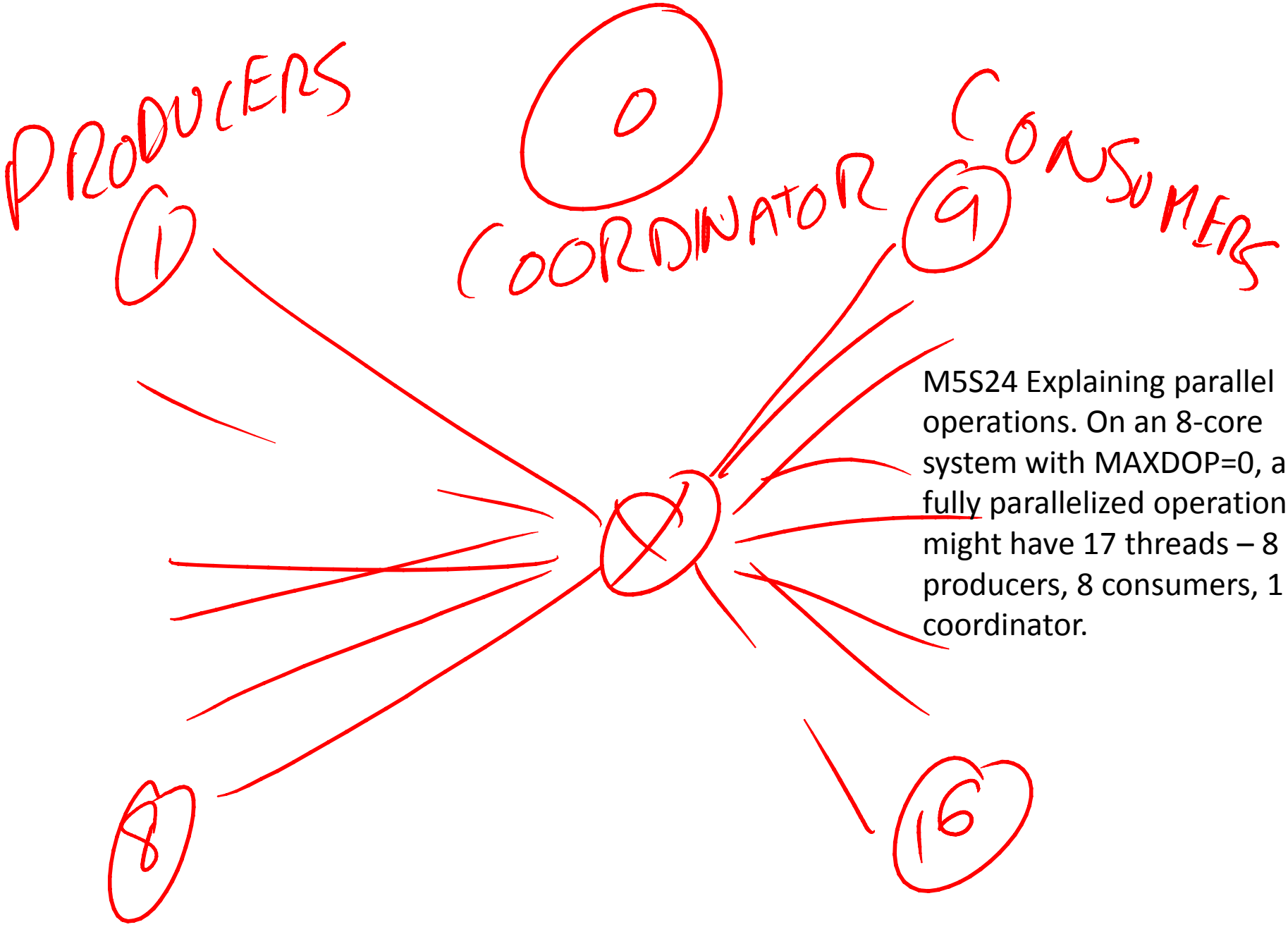
M5S9 Explaining how the Runnable Queue is a true FIFO (First-In, First-Out) queue unless Resource Governor workloads groups and group priorities are being used. High-Medium-Low priority equals 9-3-1 threads through the Runnable Queue.



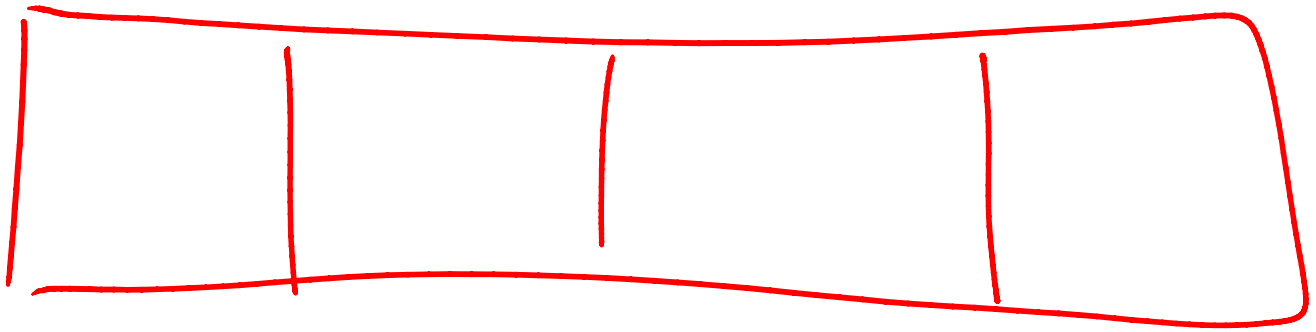
M1S11 Explaining the various wait time components. Wait time = signal wait + resource wait



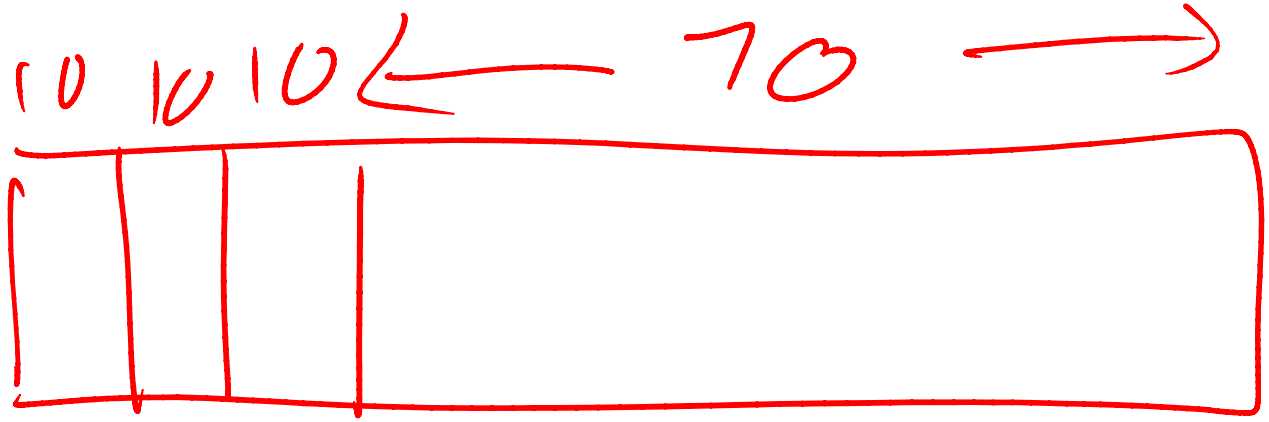
M5S22 Explaining how sometimes a WRITELOG wait time will be longer than the log file average write latency. In that case, a component of the WRITELOG resource wait time is contention for a log buffer.



M5S24 Explaining parallel operations. On an 8-core system with MAXDOP=0, a fully parallelized operation might have 17 threads – 8 producers, 8 consumers, 1 coordinator.



M5S24 Explaining how CXPACKET waits occur. In a parallel operation, the coordinator thread always incurs a CXPACKET wait. If the threads are not given an equal amount of work to do (the bottom illustration), some will finish early and incur CXPACKET waits until the longest-running thread(s) complete.



ACCESS_METHODS_
DATASET_PARENT

M5S24 In parallel scans, it's common to see CXPACKET waits plus LATCH_EX waits on the ACCESS_METHODS_DATASET_PARENT latch, which is used for parallel scan coordination.

CXPACKET

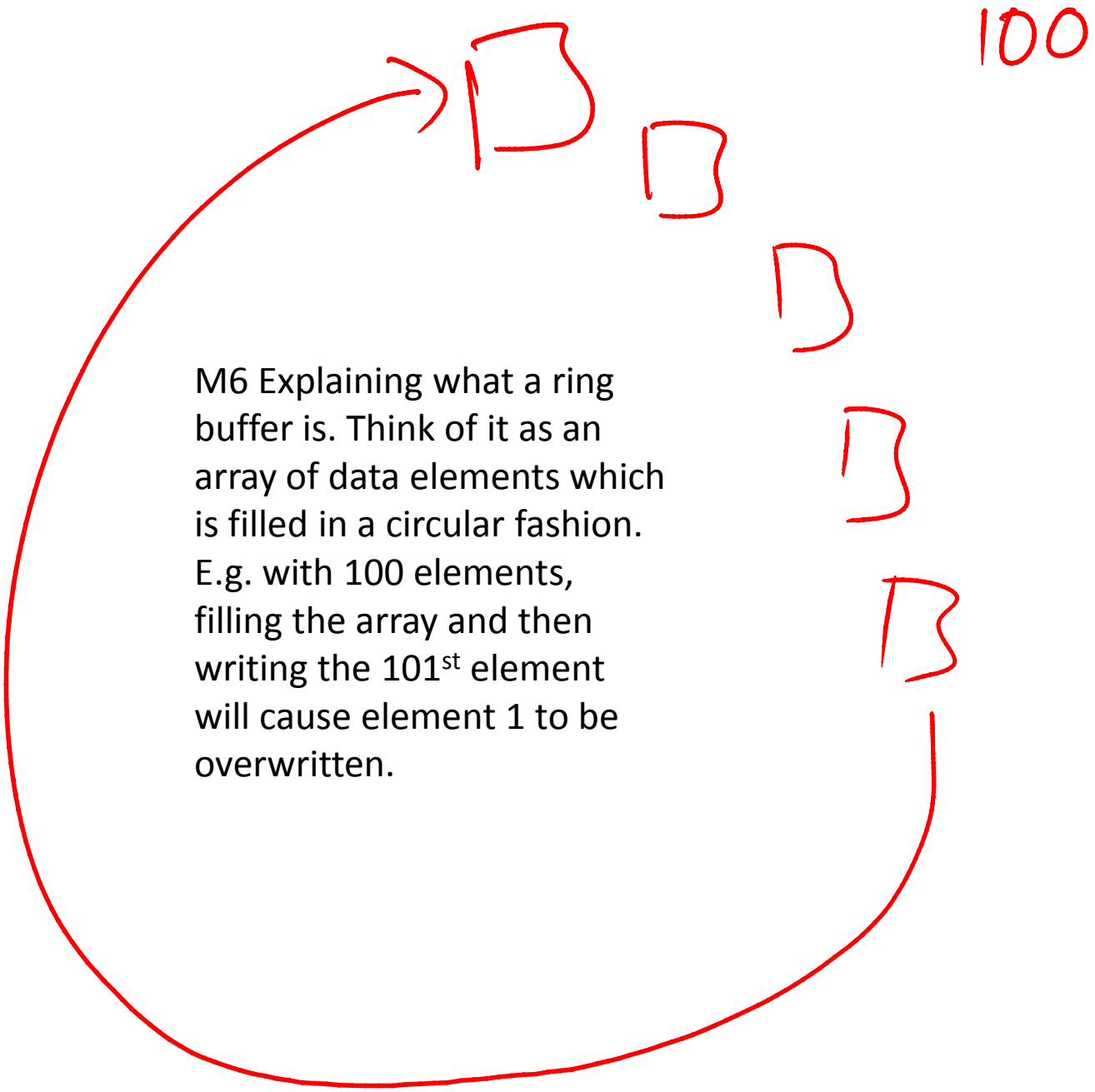
+ PAGEIOLATCH_SH

M5S24 Common pattern for parallel scans is CXPACKET plus PAGEIOLATCH_SH. Look for plans not using nonclustered indexes.

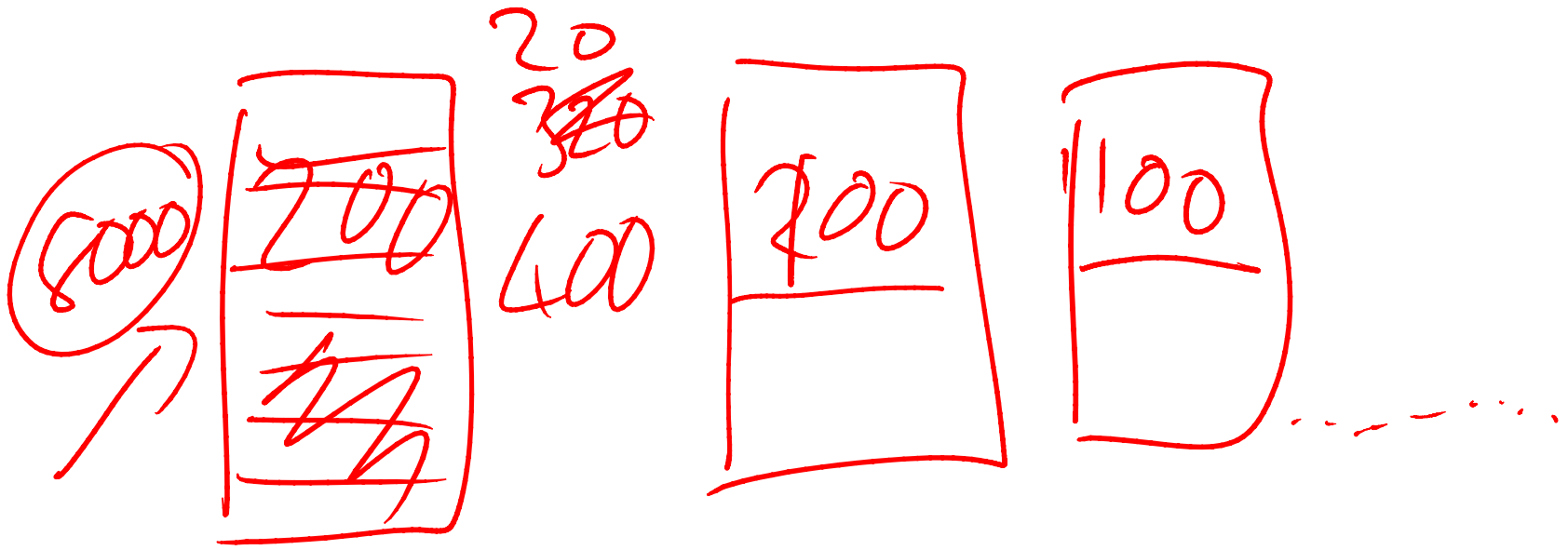
Pass@word1

SHIFT-2

Password for the VPC on your DVD



M6 Explaining what a ring buffer is. Think of it as an array of data elements which is filled in a circular fashion. E.g. with 100 elements, filling the array and then writing the 101st element will cause element 1 to be overwritten.



M6 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. Extended events can let you see this using the TRACK_CAUSALITY option. See the Page Splits demo for an example.