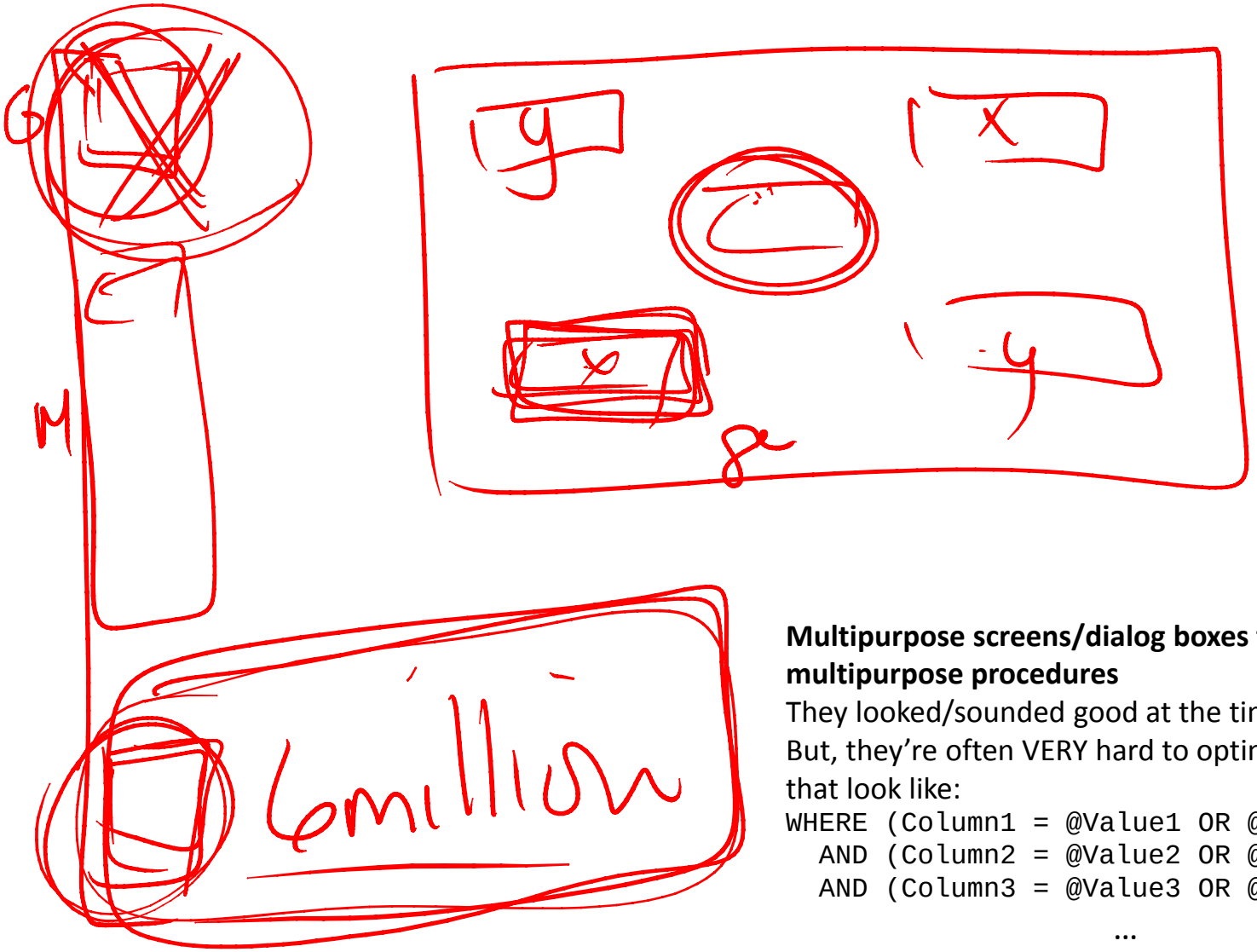

Module 9
Plan Cache &
Index Analysis
SQL
skills



Multipurpose screens/dialog boxes that lead to multipurpose procedures

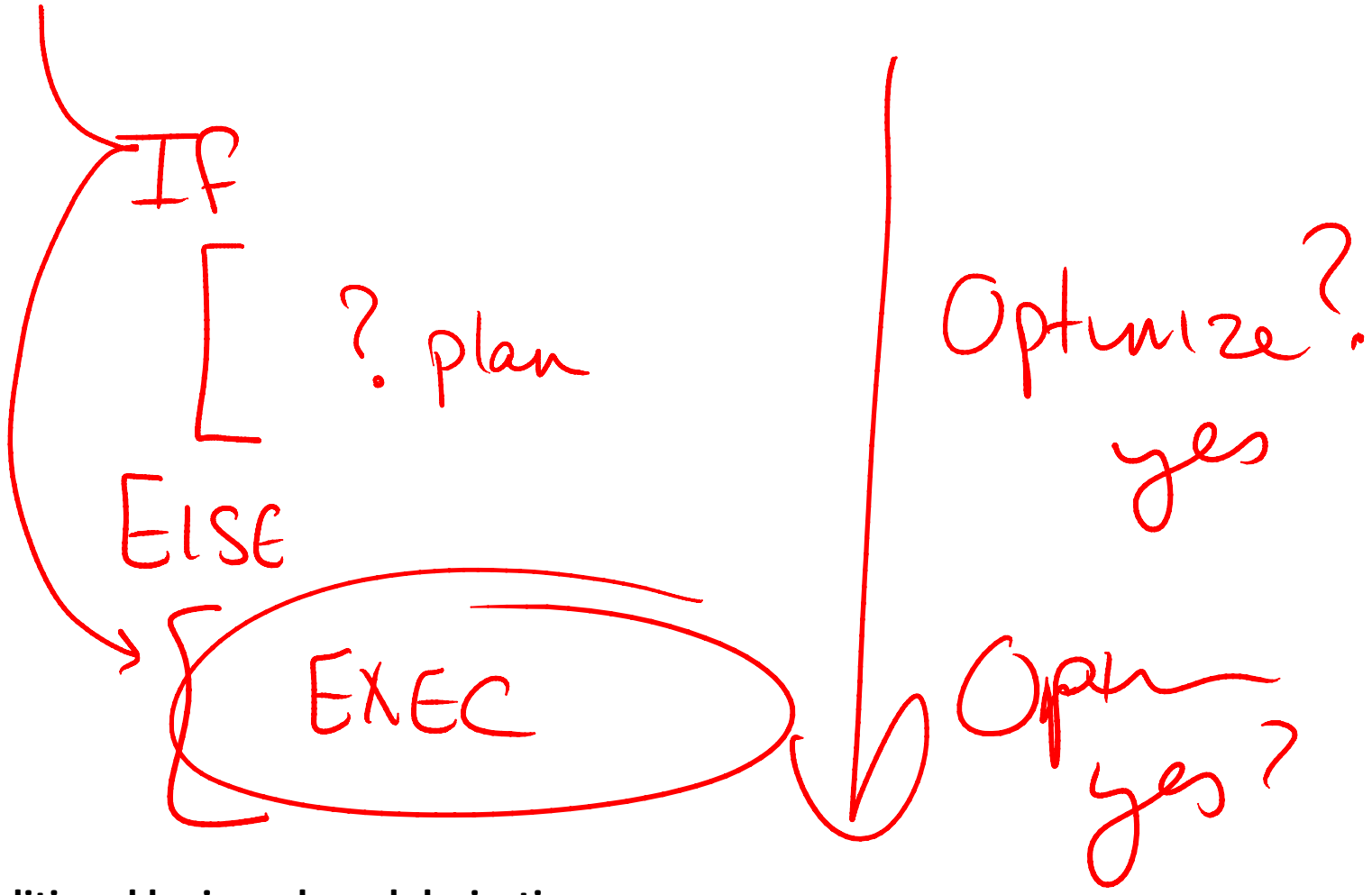
They looked/sounded good at the time?

But, they're often VERY hard to optimize with where clauses that look like:

```
WHERE (Column1 = @Value1 OR @Value1 IS NULL)
  AND (Column2 = @Value2 OR @Value2 IS NULL)
  AND (Column3 = @Value3 OR @Value3 IS NULL)
```

…

Check out the demo scripts in order to fully see how to better optimize this – using sp_executesql (for combinations that are stable) and EXEC (@String) for combinations that need to be recompiled. Or, use sp_executesql with strings that include OPTION (RECOMPILE) for the parameters that might not generate consistent plans.



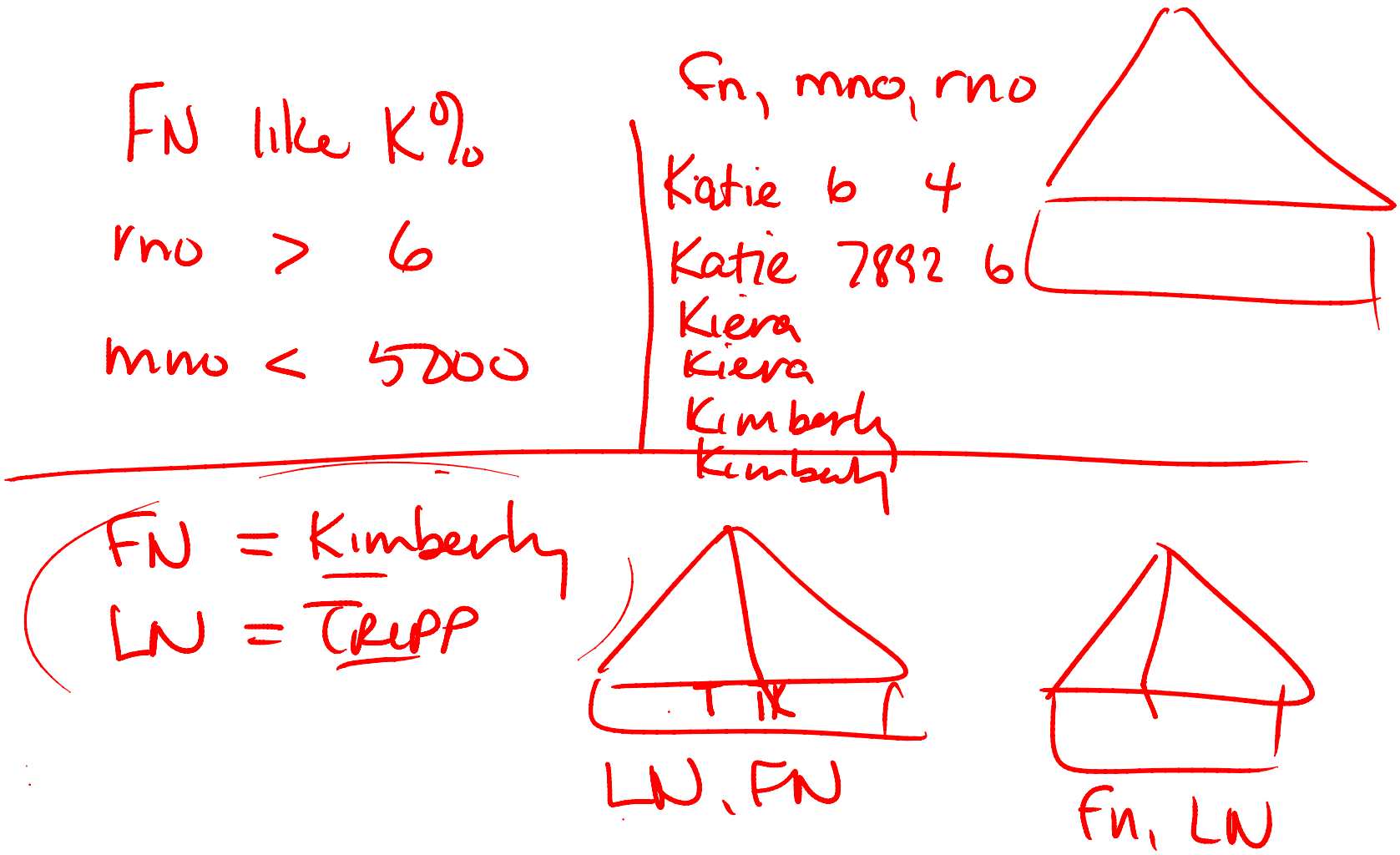
Conditional logic and modularization

Creating branching logic for different types of parameters seems logical but because SQL Server optimizes the process of optimization (where they try to optimize anything that's optimizable), you can often end up with a plan that's not ideal.

You are much better off breaking that code into smaller subprocedures – SQL Server will never step into a subprocedure unless it's executed and in this case it will only be executed with exactly the right parameters.

Index Health – Checking for fragmentation

Here I was drawing how sys.dm_db_index_physical_stats uses a scan of the first intermediate level (index level 1) to analyze for the value: avg_fragmentation_in_percent. If your analysis/rebuild scripts use this to determine fragmentation than you'd be better off just doing a limited scan. What SAMPLED and DETAILED offer are page-specific details (page density, forwarding pointers, etc.) and if you're NOT using those (most people aren't!) then you're wasting time with a SAMPLED or DETAILED can during your maintenance window!



Adding more indexes

Here we were discussing the order of columns. If there are range-based predicates than it doesn't usually matter which column is second. You tend to want to put the most selective first (in terms of the PREDICATES supplied). If all of the predicates are equality-based then it doesn't really matter. You're best ordering them by the LEFT-based combination that's used most commonly.