

Mathematics of MITS Drill-Down Paths

Dimensioned Paths

This document describes the impact of the number of identifiers and maximum indentation depth as established in an application's configuration. This applies to the use of dimensioned paths (fixed number of maximum drill-down combinations) versus the use of designated paths as established by using MitsTree or specifying the specific drill-down paths in MitsMaker.

The number of accumulators that are updated for each transaction is computed as the sum of the *combinations* of identifiers going from zero identifiers (the root) to the maximum indentation depth (MAX.INDENTATION), i.e.:

$$\text{SUM}(\text{IDENTIFIERS!}/\text{DEPTH!}(\text{IDENTIFIERS}-\text{DEPTH})!, \text{DEPTH}=0..\text{MAX.INDENTIONS})$$

The following table shows the results of this computation from one through twelve identifiers (going down) for each maximum indentation depth (going across):

Number of Drill-down Combinations													
Maximum Indention (Drill-down) Level													
		1	2	3	4	5	6	7	8	9	10	11	12
# of Drill-down Identifiers	1	2											
	2	3	4										
	3	4	7	8									
	4	5	11	15	16								
	5	6	16	26	31	32							
	6	7	22	42	57	63	64						
	7	8	29	64	99	120	127	128					
	8	9	37	93	163	219	247	255	256				
	9	10	46	130	256	382	466	502	511	512			
	10	11	56	176	386	638	848	968	1,013	1,023	1,024		
	11	12	67	232	562	1,024	1,486	1,816	1,981	2,036	2,047	2,048	
	12	13	79	299	794	1,586	2,510	3,302	3,797	4,017	4,083	4,095	4,096

As show above, an application with 7 identifiers and a maximum indentation depth of 4 will cause the updating of 99 accumulators for each transaction. These are the actual number of combinations updated. The number of drill-down paths are always more than the combinations paths are counted using every “direction” within combinations. Note that an application with no limit on indentation depth will happen to end up with $2^{\text{identifiers}}$ combinations.

In contrast with the above, the number of unique identifier *paths* available within an application is computed as the sum of the *permutations* of identifiers going from zero identifiers (the root) to the maximum indentation depth (MAX.INDENTATION), i.e.:

$$\text{SUM}(\text{IDENTIFIERS!}/(\text{IDENTIFIERS-DEPTH})!, \text{DEPTH}=0..\text{MAX.INDENTIONS})$$

The following table shows the results of this computation from one through ten identifiers (going down) for each maximum indentation depth (going across):

Number of Possible Drill-down Paths									
Maximum Indention (Drill-down) Level									
		1	2	3	4	5	6	7	8
# of Drill-down Identifiers	1	2							
	2	3	5						
	3	4	10	16					
	4	5	17	41	65				
	5	6	26	86	206	326			
	6	7	37	157	517	1,237	1,957		
	7	8	50	260	1,100	3,620	8,660	13,700	
	8	9	65	401	2,081	8,801	28,961	69,281	109,601
	9	10	82	586	3,610	18,730	79,210	260,650	623,530
	10	11	101	821	5,861	36,101	187,301	792,101	2,606,501

These numbers get large very quickly. For example, a 16-identifier application with no maximum indention has over 50 trillion possible paths.

Using dimensioned paths as shown in the above charts has the most simple setup process, (just stating the number of maximum drill-down levels), but is the most inefficient in terms of use of disk space and build time.

Designated Paths

Choosing the “designated” paths method of setting up drill-down parameters requires someone to specifically designate each path to be available in the datacube along with specifically stating what paths allow detail display. By stating just the paths needed for the specific cube, there are usually far less combinations to be created and this also eliminates any unneeded paths that would otherwise be created with dimensioned paths.

Setup of Designated paths is performed in either the stand-alone MitsTree windows-based program or within MitsMaker.