mguchiQ

Model Building Tutorial 2

Derived Variables, Constants, and MaxT

Contents

Overview

The objective of this document is to build on the model we created in the tutorial described in the document **021.mguchiQ – Model Building Tutorial 1** by introducing some more features of mguchiQ.

Specifically we will be introducing **Derived Variables**, **Constants** and an intelligent **MaxT**.

Derived Variables

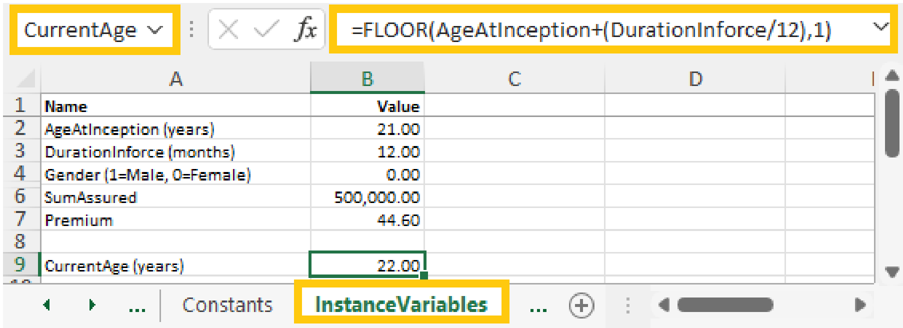
Derived variables allow us to take computational functionality out of **function** formula. This is a good idea for 2 reasons:

- It reduces the complexity and aids in the readability of **function** formula.
- It greatly enhances performance by only evaluating the derived variable once per run rather than once per product instance of the run.

Derived Instance Variables

A **Derived Instance Variable** is a variable that is derived, via a formula, from other instance variables.

We are going to create a Derived Instance Variable called **CurrentAge** that uses the Instance Variables **AgeAtInception** and **DurationInforce**. **CurrentAge** is defined in the **InstanceVariables** sheet as below:



Derived Single Variables

Our example model does not require any **Derived Single Variables**, but these can be specified, via a formula, in a similar way to how **Derived Instance Variables** are specified.

Derived Series Variables

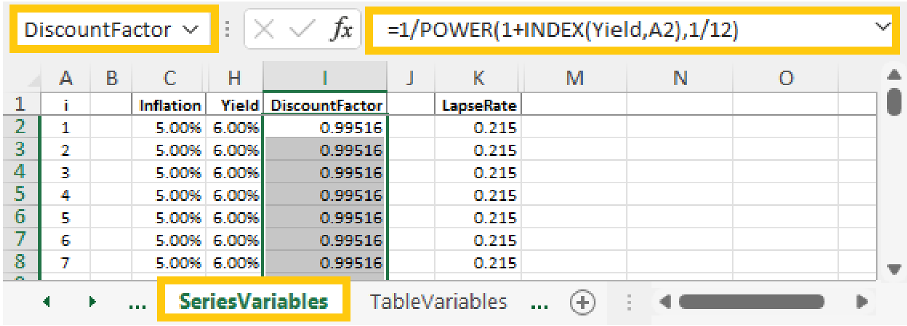
A Derived Series Variable is a variable that is derived, via a formula, from other Series Variables.

If we have a look at our Function **BEL** which is defined as follows:

BEL(t) = IF(t=MaxT, NetCF(t), NetCF(t) + (BEL(t+1) * (1/((1+Yield(t))^(1/12))))

The underlined part of the formula is a conversion of the yield to a discount factor. Note that this part of the formula does not rely on any **Instance Variables** and is therefore a candidate for extraction out of the function formula as a Derived Series Variable.

This is achieved by created a Derived Series Variable called **DiscountFactor** in the **SeriesVariables** sheet as follows:



	A	B	C	H	I	J	K
1	i		Inflation	Yield	DiscountFactor		LapseRate
2	1		5.00%	6.00%	0.99516		0.215
3	2		5.00%	6.00%	0.99516		0.215
4	3		5.00%	6.00%	0.99516		0.215
5	4		5.00%	6.00%	0.99516		0.215
6	5		5.00%	6.00%	0.99516		0.215
7	6		5.00%	6.00%	0.99516		0.215
8	7		5.00%	6.00%	0.99516		0.215

Referencing other **Series Variables** must always be done via the **INDEX** function using the name of the referenced **Series Variable** and must not be referenced directly. So, the formula noted above must be written:

= 1 / POWER(1+INDEX(Yield,A2),1/12)

And not:

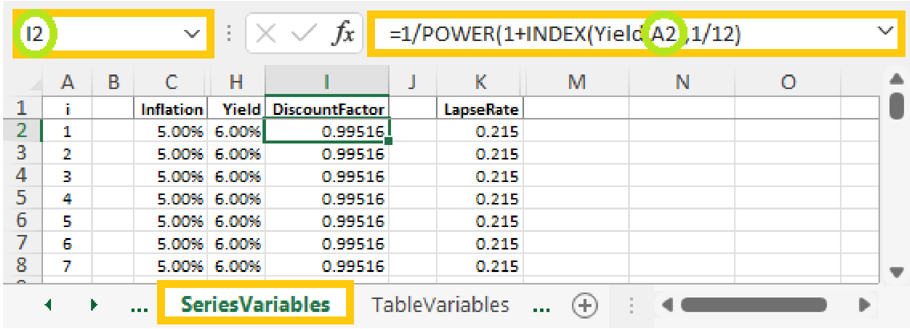
= 1 / POWER(1+H2,1/12)

We can then update our BEL formula as follows:

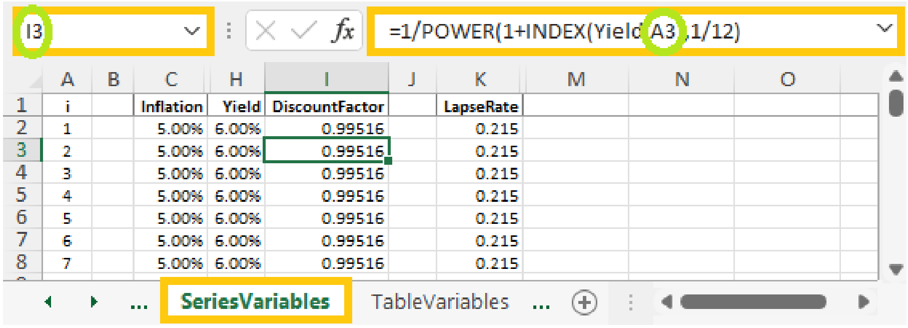
BEL(t) = IF(t=MaxT, NetCF(t), NetCF(t) + (BEL(t+1) * DiscountFactor[t])

Not only does this reduce the complexity of the BEL formula and make it more readable but it will also make all Runs using this model more efficient. This is because all Derived Variables are only calculated once at the start of a Run whereas the previous formula would have calculated the portion **(1/((1+Yield(t))^(1/12)))** for every product instance of the Run.

All formulae in the derived series range must have a similar formula, the only thing that must change between formulae on different rows is the reference to column A (the series index), as below:

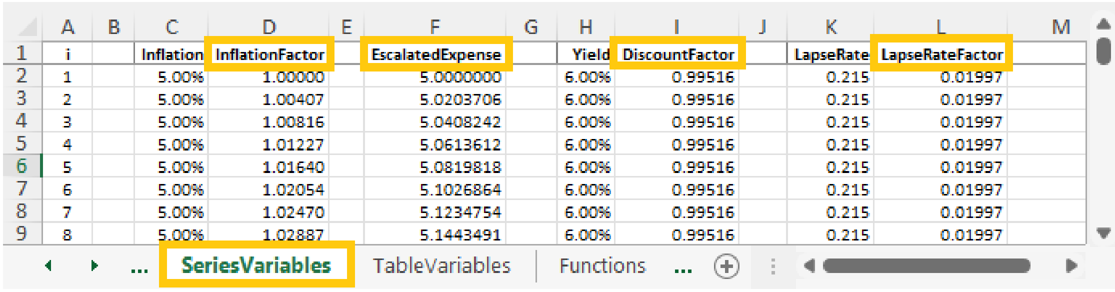


	A	B	C	H	I	J	K	M	N	O
1	i		Inflation	Yield	DiscountFactor		LapseRate			
2	1		5.00%	6.00%	0.99516		0.215			
3	2		5.00%	6.00%	0.99516		0.215			
4	3		5.00%	6.00%	0.99516		0.215			
5	4		5.00%	6.00%	0.99516		0.215			
6	5		5.00%	6.00%	0.99516		0.215			
7	6		5.00%	6.00%	0.99516		0.215			
8	7		5.00%	6.00%	0.99516		0.215			



	A	B	C	H	I	J	K	M	N	O
1	i		Inflation	Yield	DiscountFactor		LapseRate			
2	1		5.00%	6.00%	0.99516		0.215			
3	2		5.00%	6.00%	0.99516		0.215			
4	3		5.00%	6.00%	0.99516		0.215			
5	4		5.00%	6.00%	0.99516		0.215			
6	5		5.00%	6.00%	0.99516		0.215			
7	6		5.00%	6.00%	0.99516		0.215			
8	7		5.00%	6.00%	0.99516		0.215			

We will update our **SeriesVariable** sheet with a few more Derived Series Variables as follows:



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	i		Inflation	InflationFactor		EscalatedExpense		Yield	DiscountFactor		LapseRate	LapseRateFactor	
2	1		5.00%	1.00000		5.0000000		6.00%	0.99516		0.215	0.01997	
3	2		5.00%	1.00407		5.0203706		6.00%	0.99516		0.215	0.01997	
4	3		5.00%	1.00816		5.0408242		6.00%	0.99516		0.215	0.01997	
5	4		5.00%	1.01227		5.0613612		6.00%	0.99516		0.215	0.01997	
6	5		5.00%	1.01640		5.0819818		6.00%	0.99516		0.215	0.01997	
7	6		5.00%	1.02054		5.1026864		6.00%	0.99516		0.215	0.01997	
8	7		5.00%	1.02470		5.1234754		6.00%	0.99516		0.215	0.01997	
9	8		5.00%	1.02887		5.1443491		6.00%	0.99516		0.215	0.01997	

InflationFactor = IF(A2=1,1,POWER(1+INDEX(Inflation,A2-1),(A2-1)/12))

EscalatedExpense = Expense*INDEX(InflationFactor,A2)

DiscountFactor = 1 / POWER(1+INDEX(Yield,A2),1/12)

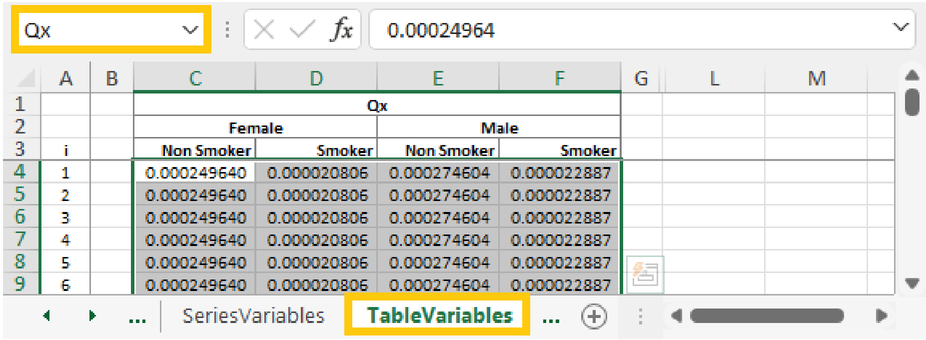
LapseRateFactor = 1-POWER(1-INDEX(LapseRate,A2),1/12)

Note: in the formulae above the only **direct** reference to a cell is to cells in column **A** – the index of the **series**. All other cell references must be by their **name**.

Derived Table Variables

A **Derived Table Variable** is a variable that is derived, via a formula, from other **Table Variables**.

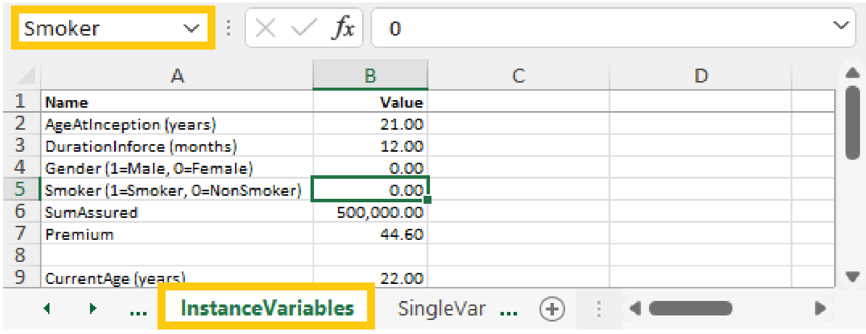
Firstly we are going to flesh out our Table Variable **Qx** a bit. Previously we had a 2 column table representing males and females. We will now have a 4 column table representing a combination of **Gender** (male / female) and **Smoker s**tatus (smoker / non-smoker) as below:


Qx | 0.00024964

	Qx			
	Female		Male	
i	Non Smoker	Smoker	Non Smoker	Smoker
1	0.000249640	0.000020806	0.000274604	0.000022887
2	0.000249640	0.000020806	0.000274604	0.000022887
3	0.000249640	0.000020806	0.000274604	0.000022887
4	0.000249640	0.000020806	0.000274604	0.000022887
5	0.000249640	0.000020806	0.000274604	0.000022887
6	0.000249640	0.000020806	0.000274604	0.000022887

SeriesVariables | TableVariables

We are now going to jump back to our **InstanceVariables** sheet and add an extra Instance Variable **Smoker**:


Smoker | 0

Name	Value
AgeAtInception (years)	21.00
DurationInforce (months)	12.00
Gender (1=Male, 0=Female)	0.00
Smoker (1=Smoker, 0=NonSmoker)	0.00
SumAssured	500,000.00
Premium	44.60
CurrentAge (years)	22.00

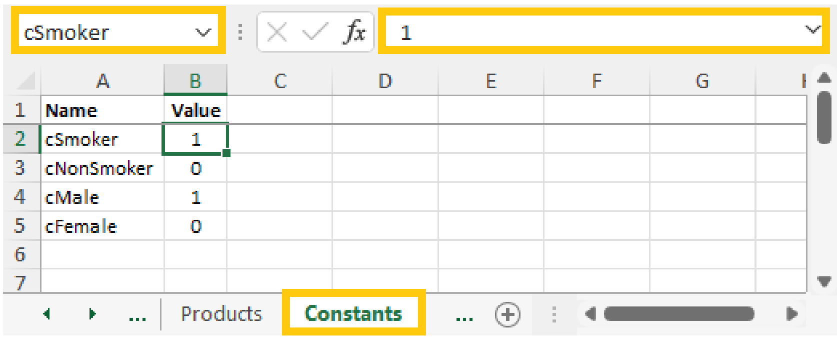
InstanceVariables | SingleVar

We are now going to take a slight diversion to introduce **Constants**…

Constants

Constants are entered in the **Constants** sheet. Constants provide no other function than making formulae more readable.

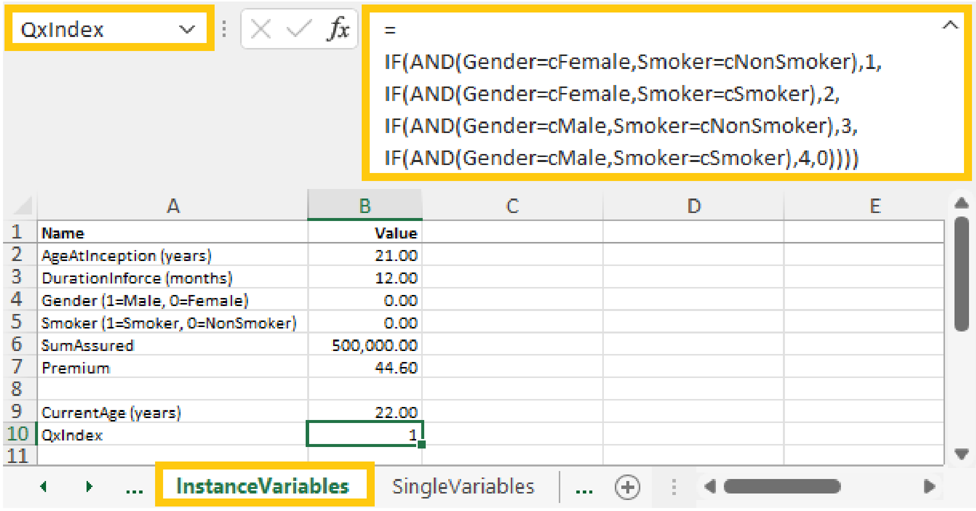
We will create 4 constants as below - simply name each cell to the name of the **Constant** required:


cSmoker | 1

Name	Value
cSmoker	1
cNonSmoker	0
cMale	1
cFemale	0

Products | Constants

Getting back to our InstanceVariables sheet, we are going to add another Derived Instance Variable called **QxIndex**. **QxIndex** is going to be our index into the **Qx** table based on a combination of **Gender** and **Smoker** status. Note the use of the Constants defined above in the formula:


QxIndex

```
=
IF(AND(Gender=cFemale,Smoker=cNonSmoker),1,
IF(AND(Gender=cFemale,Smoker=cSmoker),2,
IF(AND(Gender=cMale,Smoker=cNonSmoker),3,
IF(AND(Gender=cMale,Smoker=cSmoker),4,0))))
```

	A	B
1	Name	Value
2	AgeAtInception (years)	21.00
3	DurationInforce (months)	12.00
4	Gender (1=Male, 0=Female)	0.00
5	Smoker (1=Smoker, 0=NonSmoker)	0.00
6	SumAssured	500,000.00
7	Premium	44.60
8		
9	CurrentAge (years)	22.00
10	QxIndex	1
11		

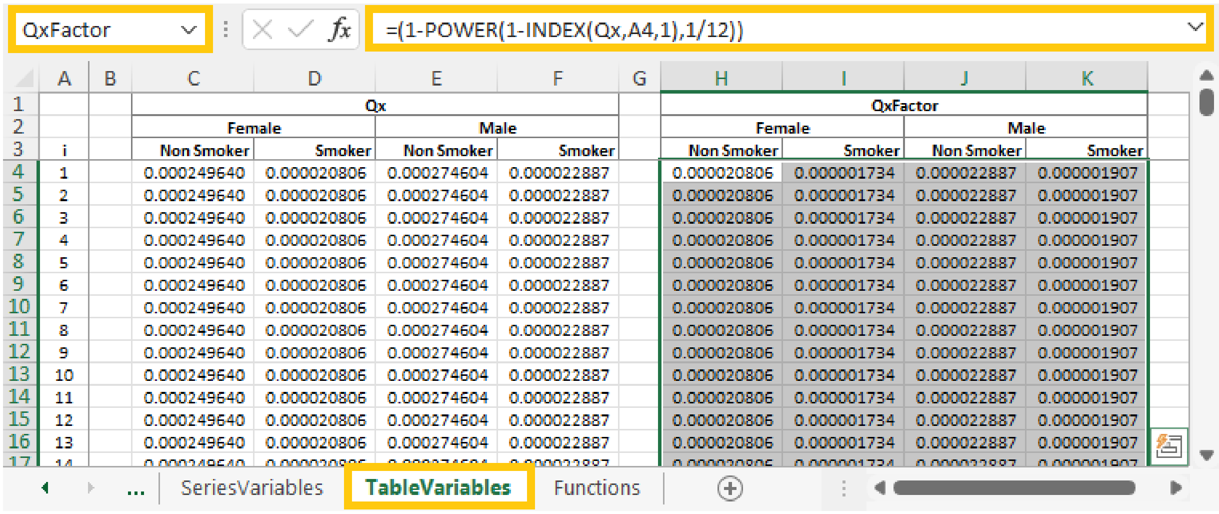
Derives Table Variables continued...

If we have a look at our Function **Deaths** which is defined as follows:

Deaths(t) = PolicyHoldersInforce(t) * <u>(1-(1-Qx[Age(t),Gender+1])^(1/12))</u>

The underlined part of the formula does not rely on any Instance Variables and is therefore a candidate for extraction out of the function formula as a Derived Table Variable.

This is achieved by created a Derived Table Variable called **QxFactor** in the **TableVariables** sheet as follows:


QxFactor

```
=(1-POWER(1-INDEX(Qx,A4,1),1/12))
```

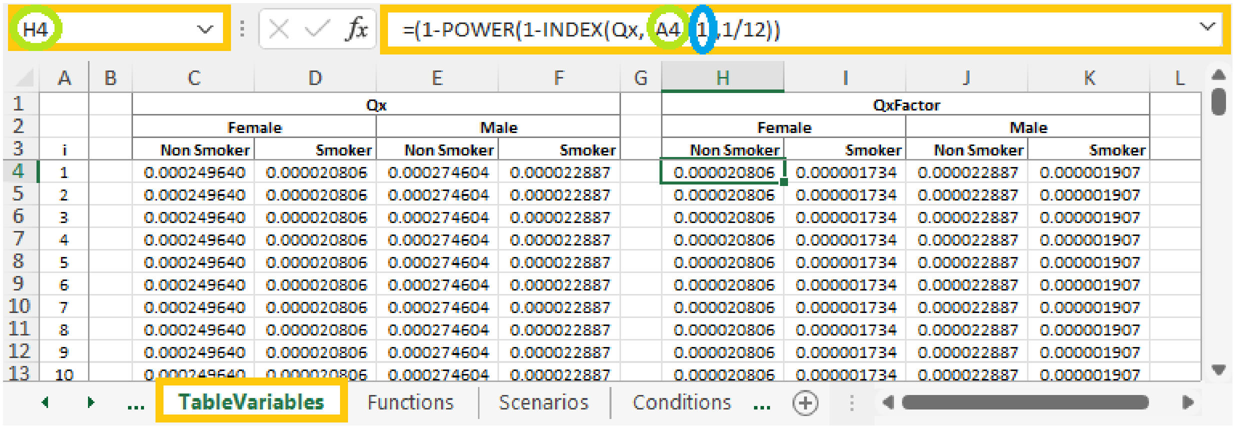
	A	B	C	D	E	F	G	H	I	J	K
1			Qx					QxFactor			
2			Female		Male			Female		Male	
3	i		Non Smoker	Smoker	Non Smoker	Smoker		Non Smoker	Smoker	Non Smoker	Smoker
4	1		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
5	2		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
6	3		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
7	4		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
8	5		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
9	6		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
10	7		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
11	8		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
12	9		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
13	10		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
14	11		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
15	12		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
16	13		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907
17	14		[illegible]	[illegible]	[illegible]	[illegible]		[illegible]	[illegible]	[illegible]	[illegible]

We can then update our Deaths formula as follows:

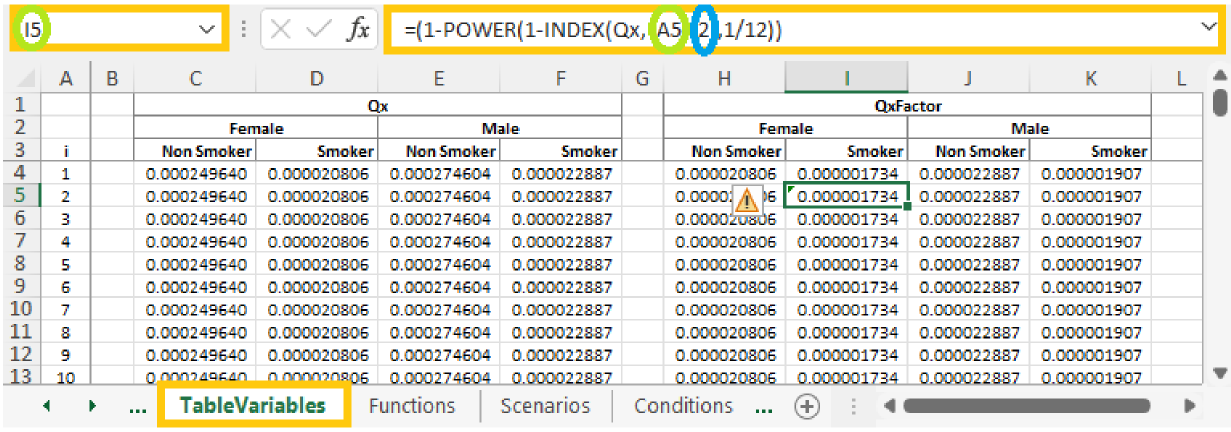
Deaths(t) = PolicyHoldersInforce(t) * <u>QxFactor[Age(t),QxIndex]</u>

Again this is more readable and also more efficient.

All formulae in the table range must have a similar formula, the only thing that must change between formulae on different rows is the reference to column A, the second parameter to the INDEX function (the table x-index), and the third parameter to the INDEX function (the table y-index) as below:



	A	B	C	D	E	F	G	H	I	J	K	L
1			Qx					QxFactor				
2			Female		Male			Female		Male		
3	i		Non Smoker	Smoker	Non Smoker	Smoker		Non Smoker	Smoker	Non Smoker	Smoker	
4	1		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
5	2		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
6	3		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
7	4		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
8	5		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
9	6		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
10	7		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
11	8		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
12	9		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
13	10		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	



	A	B	C	D	E	F	G	H	I	J	K	L
1			Qx					QxFactor				
2			Female		Male			Female		Male		
3	i		Non Smoker	Smoker	Non Smoker	Smoker		Non Smoker	Smoker	Non Smoker	Smoker	
4	1		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
5	2		0.000249640	0.000020806	0.000274604	0.000022887		[illegible]	0.000001734	0.000022887	0.000001907	
6	3		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
7	4		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
8	5		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
9	6		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
10	7		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
11	8		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
12	9		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	
13	10		0.000249640	0.000020806	0.000274604	0.000022887		0.000020806	0.000001734	0.000022887	0.000001907	

Updated Functions

Our original set of functions looked as follows:

Function	Formula
Age(t)	FLOOR(AgeAtInception+((DurationInforce+t)/12))
Deaths(t)	PolicyHoldersInforce(t) * (1-(1-Qx[Age(t),Gender+1)^(1/12)))
Lapses(t)	(PolicyHoldersInforce(t) - Deaths(t)) * (1-((1-LapseRate[DurationInforce+t])^(1/12)))
PolicyHoldersInforce(t)	IF(t=1, 1, PolicyHoldersInforce(t-1) - Deaths(t-1) - Lapses(t-1))
PremiumsReceived(t)	PolicyHoldersInforce(t) * Premium
DeathBenefitsPaid(t)	Deaths(t) * SumAssured
ExpensesPaid(t)	PolicyHoldersInforce(t) * (Expense * IF(t=1,1,(1+Inflation(t-1))^((t-1)/12)))
NetCF(t)	PremiumsReceived(t) - ExpensesPaid(t) - DeathBenefitsPaid(t)
BEL(t)	IF(t=MaxT, NetCF(t), NetCF(t) + (BEL(t+1) * (1/((1+Yield(t))^(1/12)))

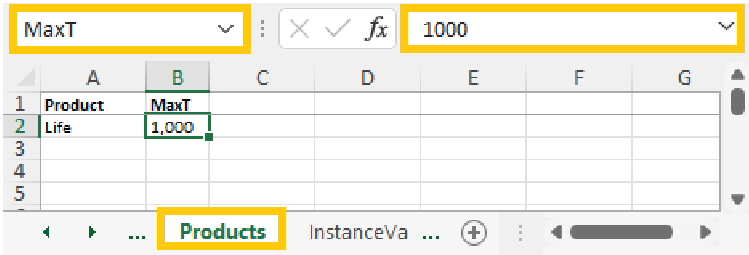
With the Derived Variables we have introduced we can rewrite our functions as follows (changed portions highlighted in *italics*):

Function	Formula
Age(t)	FLOOR(AgeAtInception+((DurationInforce+t)/12))
Deaths(t)	PolicyHoldersInforce(t) * *QxFactor[Age(t),QxIndex]*
Lapses(t)	(PolicyHoldersInforce(t) - Deaths(t)) * *LapseRateFactor[DurationInforce+t]*
PolicyHoldersInforce(t)	IF(t=1, 1, PolicyHoldersInforce(t-1) - Deaths(t-1) - Lapses(t-1))
PremiumsReceived(t)	PolicyHoldersInforce(t) * Premium
DeathBenefitsPaid(t)	Deaths(t) * SumAssured
ExpensesPaid(t)	PolicyHoldersInforce(t) * *EscalatedExpense[t]*
NetCF(t)	PremiumsReceived(t) - ExpensesPaid(t) - DeathBenefitsPaid(t)
BEL(t)	IF(t=MaxT, NetCF(t), NetCF(t) + (BEL(t+1) * *DiscountFactor[t])*

This new set of function definitions is easier to understand and will run more efficiently.

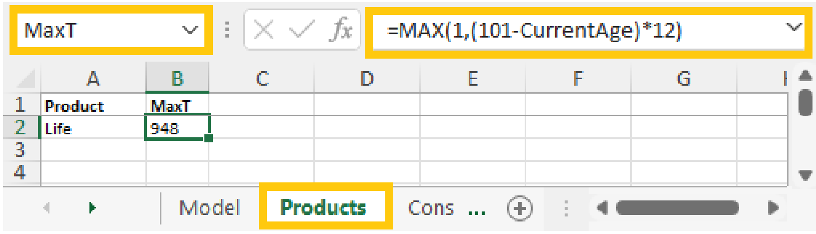
MaxT

In the first version of our model we simply specified **MaxT** as 1,000:



What a **MaxT** specified in this way will do is run each product instance 1,000 time periods into the future. This will not be the most efficient way to specify **MaxT** as it will run, for example, a policy holder who is 20 years old the same number of time periods into the future as a policy holder who is 80 years old.

Instead we are going to specify **MaxT** via a formula using our **CurrentAge** Instance Variable we created earlier, as follow:



A **MaxT** specified in this way would run all policies up to each policy holder being 101 years of age.