mguchiQ

Run Spreadsheet Tutorial

Deterministic Runs with no external data sources

Contents

Overview

In order to initiate a **Run** in mguchiQ it is necessary to provide the data necessary for the **Run** via an Excel spreadsheet. This spreadsheet is then uploaded to the mguchiQ website where it is verified and, if successful, the **Run** is processed. This document describes how to create an Excel spreadsheet that provides the necessary data for a **Run** that can be uploaded to the mguchiQ website.

It is not necessary to create a spreadsheet for a **Run** from scratch as you can download a template from the mguchiQ website specific to a **Model** and **Calculation Type**.

This document only addresses the case where you specify all the data requirements for the **Run** in the spreadsheet – it is possible to specify that some or all of the data resides in external databases – this case is covered in a later document.

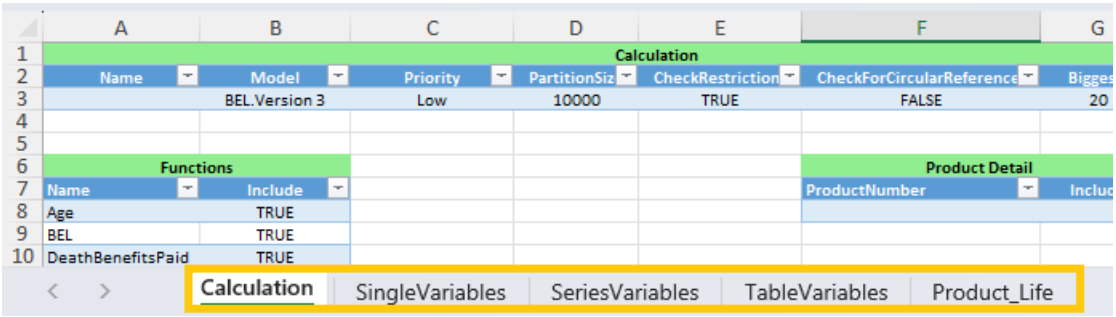
We will progress through the different sheets of the Run Template spreadsheets …

Run Spreadsheet Layout

A spreadsheet used to instruct mguchiQ to execute a Run is usually made up of a number of sheets within the Spreadsheet as follows:

Sheet	Description
Calculation	Specifies what to calculate and how to go about doing the calculations. The **Calculation** sheet is different for each different **Calculation Type**.
SingleVariables	Where you input the **Single Variable** values of the Model for this Run.
SeriesVariables	Where you input the **Series Variable** values of the Model for this Run.
TableVariables	Where you input the **Table Variable** values of the Model for this Run.
Product_xxx	Where you input the actual product items for the Run.

Below is a typical Excel spreadsheet with the required sheets defined:

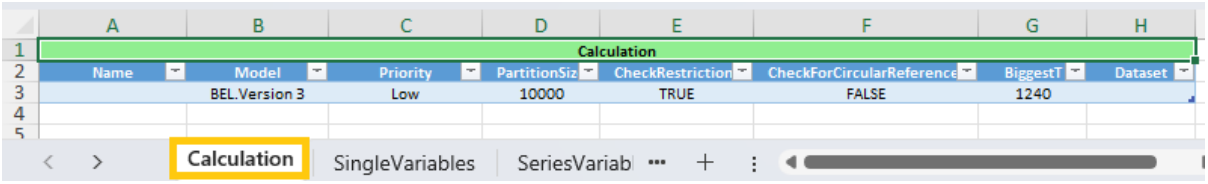


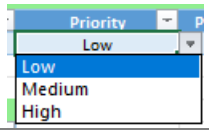
Calculation Sheet

The **Calculation** sheet specifies some generic parameters for your **Run**. All **Calculation Types** have their own **Calculation** sheets although they share many items in common:

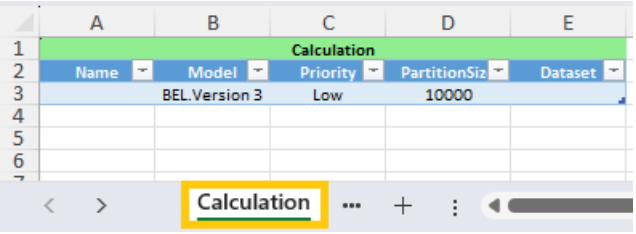
Calculation Sheet – Common Items

The **Calculation** table at the top of most **Calculations** sheets will have all or some of the following items:



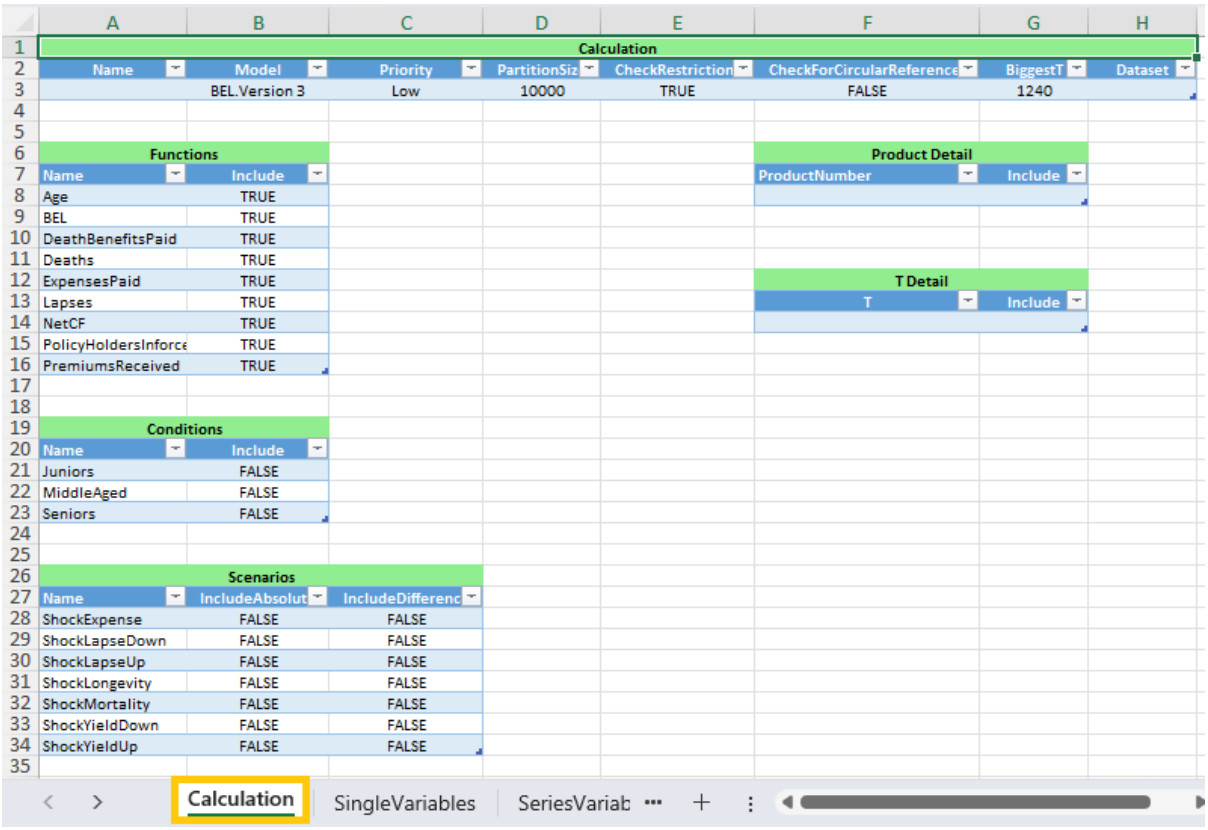
Item	Description
Name	A unique **Name** by which to identify this Run.
Model	The **Model** this Run will use.
Priority	What **Priority** must be given to this calculation. You can choose from the available items in the list box 
PartitionSize	Tells mguchiQ how to partition the data so that the calculations can be split over multiple machines. A value of 10,000 whould mean that each partition would contain 10,000 product items.
CheckRestrictions	Instructs mguchiQ to check the **Restrictions** of the Model for each product item. Note that checking **Restrictions** increase the processing time so it may well be worth doing an initial Run to first check **Restrictions** and then remove any offending product items. Initiating a Run without checking **Restrictions** that has product items that fail any of the model **Restrictions** may well cause the Run to abort – if you are unsure of the integrity of your product data then always check **Restrictions** in your Run.
CheckForCircularReferences	Instructs mguchiQ to check for circular references in the **Model**. Note that checking for circular references increase the processing time. It is usually only necessary to check for circular references once with some sample data while testing your model.
BiggestT	The absolute maximum time periods into the future. This is used for sizing of internal arrays and should not need to be changed.
Dataset	The **Dataset** name to use to find the product items for this Run. If this is left blank it is means the product items will be specified in the this Run input spreadsheet.

Calculation Sheet – Check



The **Check Calculation** Sheet has a reduced set of items from the **Common Items** described above.

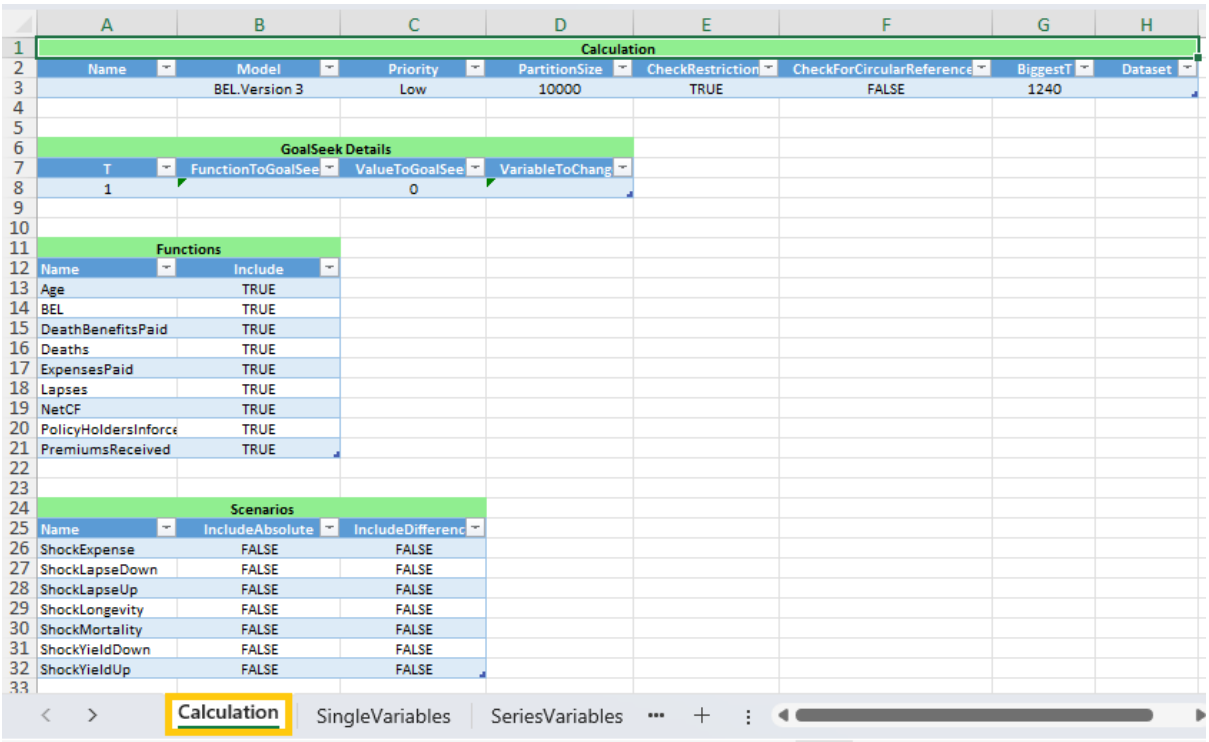
Calculation Sheet – Deterministic Aggregate



The **Deterministic Aggregate** Calculation Sheet has the following additional tables:

Item	Description
Functions	This table is pre-populated with the **Functions** of the Model and allows you to specify which **Functions** must be calculated in this Run.
Conditions	This table is pre-populated with the **Conditions** of the Model and allows you to specify which **Conditions** must be calculated in the Run.
Scenarios	This table is pre-populated with the **Scenarios** of the Model and allows you to specify which **Scenarios** must be calculated in the Run. When specifying which Scenarios must be calculated you indicate whether **Absolute** and / or **Difference** results must be calculated.
Product Detail	This table allows you to specify for which product item a **Product Calculation** must be performed. Expand the table downwards to include more product items.
T Detail	This table allows you to specify for which values of T a **T Calculation** must be performed. Expand the table downwards to include more T's.

Calculation Sheet – Goal Seek



Calculation							
Name	Model	Priority	PartitionSize	CheckRestriction	CheckForCircularReference	BiggestT	Dataset
	BEL.Version 3	Low	10000	TRUE	FALSE	1240	

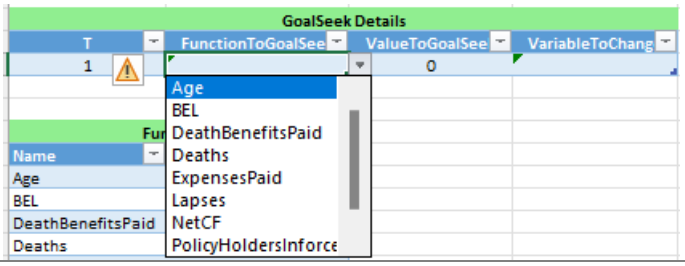
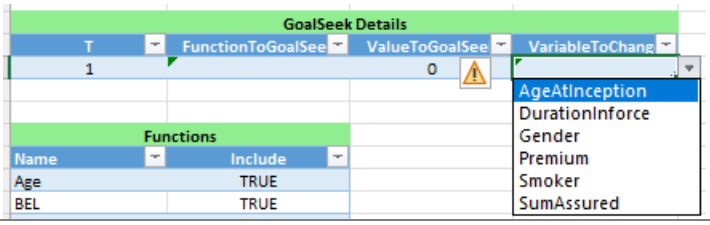
GoalSeek Details			
T	FunctionToGoalSee	ValueToGoalSee	VariableToChang
1		0	

Functions	
Name	Include
Age	TRUE
BEL	TRUE
DeathBenefitsPaid	TRUE
Deaths	TRUE
ExpensesPaid	TRUE
Lapses	TRUE
NetCF	TRUE
PolicyHoldersInforce	TRUE
PremiumsReceived	TRUE

Scenarios		
Name	IncludeAbsolute	IncludeDifferenc
ShockExpense	FALSE	FALSE
ShockLapseDown	FALSE	FALSE
ShockLapseUp	FALSE	FALSE
ShockLongevity	FALSE	FALSE
ShockMortality	FALSE	FALSE
ShockYieldDown	FALSE	FALSE
ShockYieldUp	FALSE	FALSE

Calculation | SingleVariables | SeriesVariables

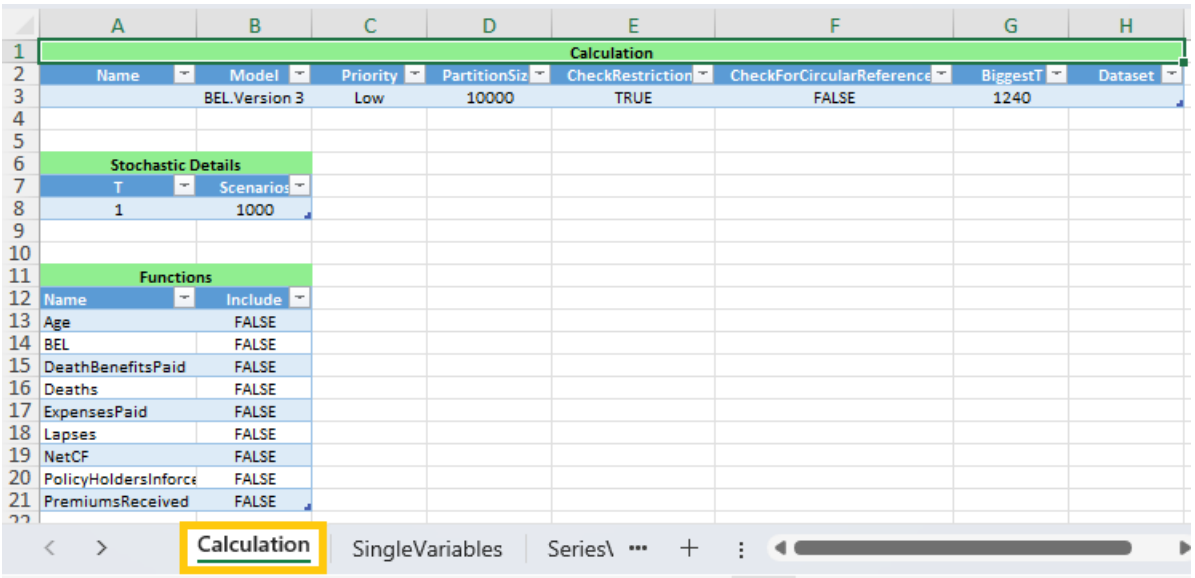
The **Goal Seek** Calculation Sheet has a **Goal Seek Details** table with the following items:

Item	Description
T	Which time interval to goal seek for.
FunctionToGoalSeek	The particular **Function** of the **Model** that is to be goal-seeked to a particular **Value**. You can choose from the pre-populated list box which includes all the **Functions** of the **Model.** 
ValueToGoalSeek	The particular numeric value that we are attempting to get the **Function** chosen above to converge to. This must be an actual number, such as 0 or 100.
VariableToChange	The **Variable** we will change in order for the chosen **Function** to converge to the chosen **Value**. You can choose from the pre-populated list box which includes all the **Variables** of the **Model.** 

The **Goal Seek** Calculation Sheet also has the following tables:

Item	Description
Functions	This table is pre-populated with the **Functions** of the Model and allows you to specify which **Functions** must be calculated in this Run.
Scenarios	This table is pre-populated with the **Scenarios** of the Model and allows you to specify which **Scenarios** must be calculated in the Run. When specifying which Scenarios must be calculated you indicate whether **Absolute** and / or **Difference** results must be calculated.

Calculation Sheet – Stochastic Aggregate



The **Stochastic Aggregate** Calculation Sheet has a **Stochastic Details** table with the following items:

Item	Description
T	Which time interval to calculate the functions for.
Scenarios	The number of scenarios to use in the stochastic calculation.

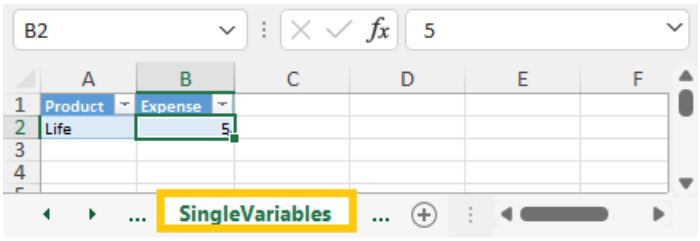
The **Stochastic Aggregate** Calculation Sheet has the following additional tables:

Item	Description
Functions	This table is pre-populated with the **Functions** of the Model and allows you to specify which **Functions** must be calculated in this Run.

Single Variables Sheet

The SingleVariables sheet allows you to provide values for **Single Variables** of the Model for this Run.

Our tutorial **Model** only has one **Single Variable** called **Expense** and it can simply be provided as follows:



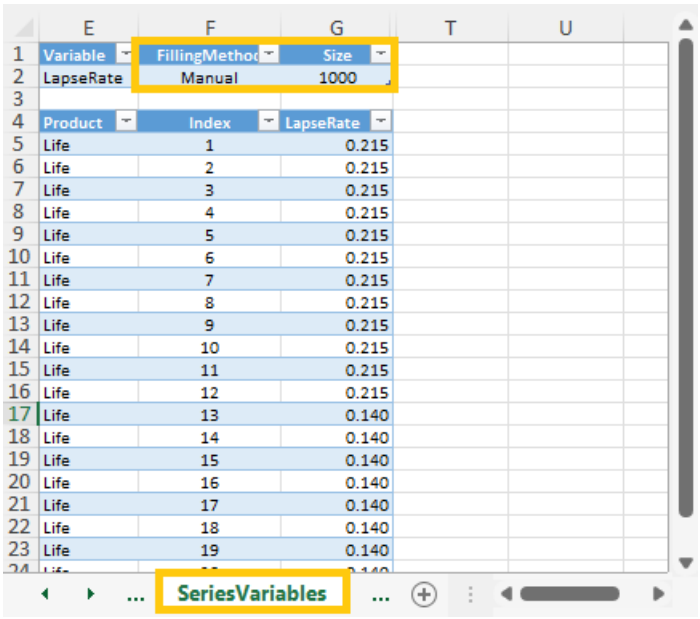
Series Variables Sheet

The **SeriesVariables** sheet allows us to provide values for **Series Variables** and optionally specify how they must be interpreted.

For this tutorial we will work with one of the **Series Variables** of our **Model**, namely **LapseRate**.

For **Series Variables** we need to provide a value for each element of the series, up to a series size that makes sense for our data.

The simplest way of providing data is to provide all the data **manually**, as below:



Variable	FillingMethod	Size
LapseRate	Manual	1000

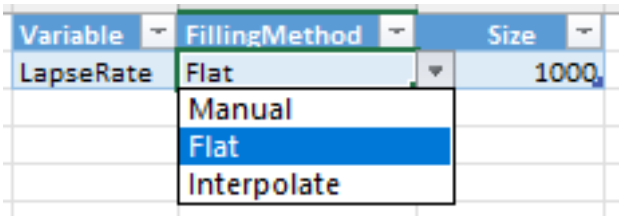
Product	Index	LapseRate
Life	1	0.215
Life	2	0.215
Life	3	0.215
Life	4	0.215
Life	5	0.215
Life	6	0.215
Life	7	0.215
Life	8	0.215
Life	9	0.215
Life	10	0.215
Life	11	0.215
Life	12	0.215
Life	13	0.140
Life	14	0.140
Life	15	0.140
Life	16	0.140
Life	17	0.140
Life	18	0.140
Life	19	0.140

This series would need to extend to cover all the possible values of **T (Index)** that may be used in our **Function** formulae.

mguchiQ provides some shortcut methods, called **Filling** methods, for specifying **Series** data:

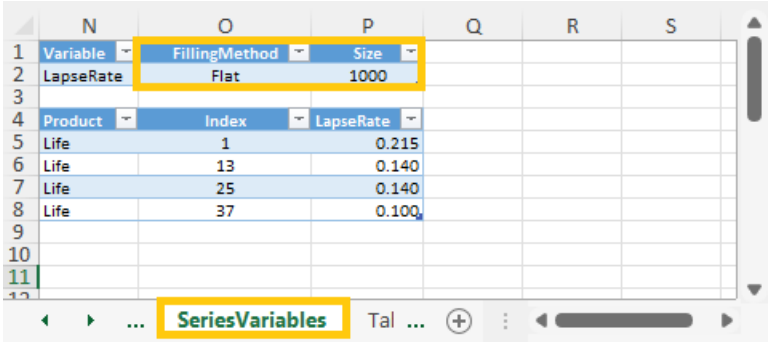
Filling Method	Description
Manual	All values are specified manually, as in the example above.
Flat	Any gaps in the Series are filled with the previous value.
Interpolate	Any gaps in the series are filled by linear interpolating between the next and previous values.

A specification table tells mguchiQ how to fill in the series:



Flat

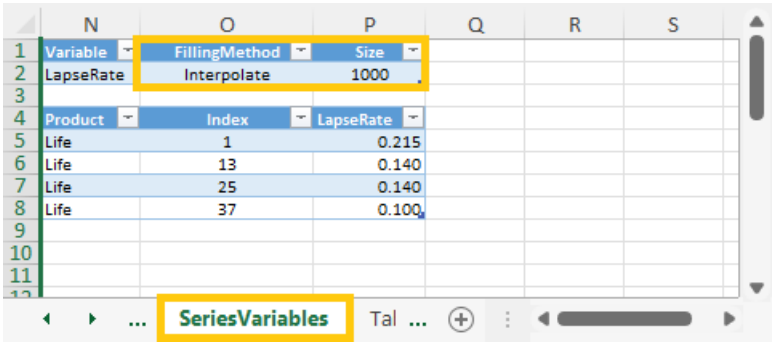
Missing values in the series are filled with the previous value in the series.



This will produce a series of 1000 values.

Interpolate

Missing values in the series are filled by linear interpolating between the previous and next values in the series.



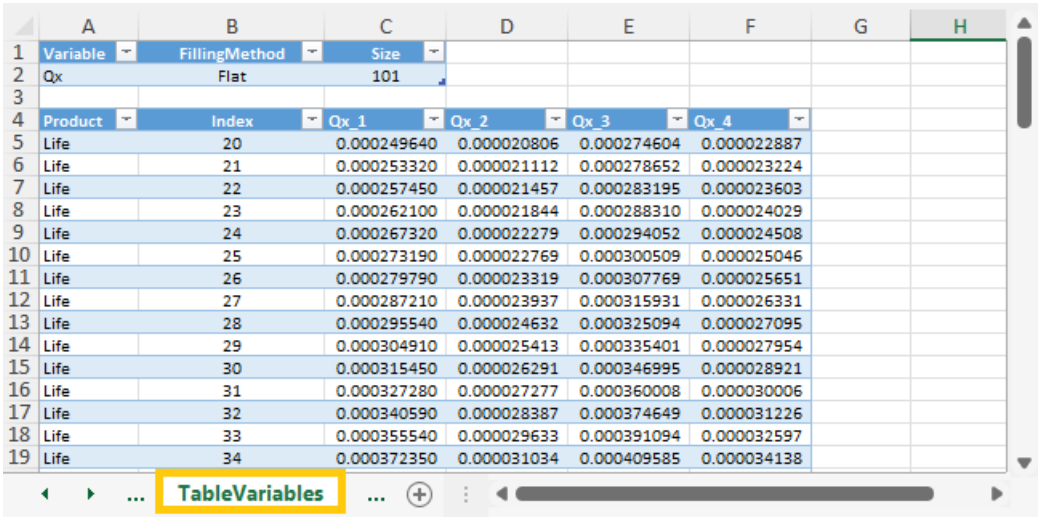
This will also produce a series of 1000 values.

Table Variables Sheet

The **TableVariables** sheet allows us to provide values for **Table Variables** and optionally specify how they must be interpreted.

Supplying Table variables is similar to supplying Series variables, with each column of the table representing a series.

An example for our **Qx** table is as follows – note the column headings **Qx_1** to **Qx_4**:



	A	B	C	D	E	F	G	H
1	Variable	FillingMethod	Size					
2	Qx	Flat	101					
3								
4	Product	Index	Qx_1	Qx_2	Qx_3	Qx_4		
5	Life	20	0.000249640	0.000020806	0.000274604	0.000022887		
6	Life	21	0.000253320	0.000021112	0.000278652	0.000023224		
7	Life	22	0.000257450	0.000021457	0.000283195	0.000023603		
8	Life	23	0.000262100	0.000021844	0.000288310	0.000024029		
9	Life	24	0.000267320	0.000022279	0.000294052	0.000024508		
10	Life	25	0.000273190	0.000022769	0.000300509	0.000025046		
11	Life	26	0.000279790	0.000023319	0.000307769	0.000025651		
12	Life	27	0.000287210	0.000023937	0.000315931	0.000026331		
13	Life	28	0.000295540	0.000024632	0.000325094	0.000027095		
14	Life	29	0.000304910	0.000025413	0.000335401	0.000027954		
15	Life	30	0.000315450	0.000026291	0.000346995	0.000028921		
16	Life	31	0.000327280	0.000027277	0.000360008	0.000030006		
17	Life	32	0.000340590	0.000028387	0.000374649	0.000031226		
18	Life	33	0.000355540	0.000029633	0.000391094	0.000032597		
19	Life	34	0.000372350	0.000031034	0.000409585	0.000034138		

... TableVariables ...

The specification table at the top (where we specify a **flat filling method** and a **size of 101**) applies to all columns of the table.

Product Sheets

Product data relates to the values for **Instance Variables** we need to supply for each of the **Product** item that will constitute our run.

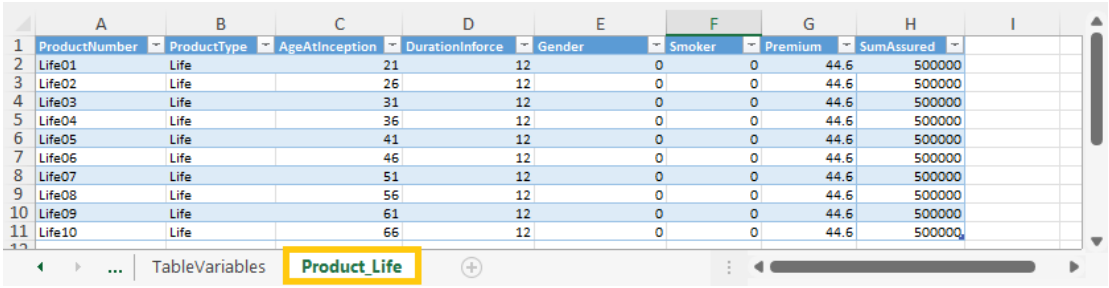
A sheet is created for each of the **Products** in our model. Each sheet's name is a combination of "**Product**" and the **Product Name** itself – in our example we have one product called **Life**, giving the sheet name **Product_Life**.

The tables for each of these products must include columns for all of the **Instance Variables** of the **Product** as defined in the **Model**. 2 extra columns need to be provided as well:

Column Name	Description
ProductNumber	A unique id to different each Product instance
ProductType	In our example Life

The table for our example is as follows:

Product_Life



	A	B	C	D	E	F	G	H	I
1	ProductNumber	ProductType	AgeAtInception	DurationInforce	Gender	Smoker	Premium	SumAssured	
2	Life01	Life	21	12	0	0	44.6	500000	
3	Life02	Life	26	12	0	0	44.6	500000	
4	Life03	Life	31	12	0	0	44.6	500000	
5	Life04	Life	36	12	0	0	44.6	500000	
6	Life05	Life	41	12	0	0	44.6	500000	
7	Life06	Life	46	12	0	0	44.6	500000	
8	Life07	Life	51	12	0	0	44.6	500000	
9	Life08	Life	56	12	0	0	44.6	500000	
10	Life09	Life	61	12	0	0	44.6	500000	
11	Life10	Life	66	12	0	0	44.6	500000	

... TableVariables Product_Life

Conclusion

This tutorial explains the case where you want to supply all of the data to your Run via an Excel spreadsheet. This is very useful for testing and proof of concept purpose. In reality your data will probably be stored in one or more data sources – a later tutorial explains this scenario.