The Multiwell Prosper Scanner application

In our previous tutorial (link), we saw the use of Python notebooks to run multiwell scanning of multiple models in one shot. This time we will use a graphical user interface or GUI to run the same model scanning but even in a more efficient way.

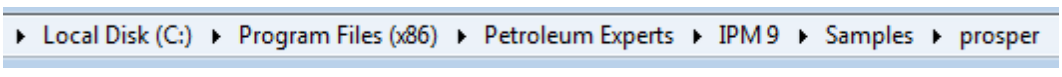
The beauty of using these digital aids is that they allow us to run the data exploratory analysis for one, a couple of well, dozens of wells of hundreds of wells. By getting familiar with Python scripting you will be building more reliable well models where the quality verification of input data is performed not only by you but by the computer and some basic statistics.

How does this work?

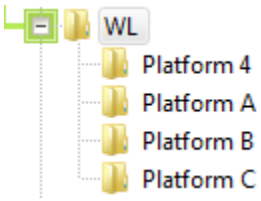
The workflow for multiwell scanning is very simple. Start by selecting the folder containing the well models to analyze. Then, select the option "*Scan*". The script will start Prosper and will open each model and retrieve the data you indicated. At the end of the process you obtain a tabular form in an Excel worksheet where each row is a well –or string-, and each column is an OpenServer variable. In essence, the output will look like this:

	A	E	Q	W	AB
1	modelTailname	PROSPER.SIN.SUM.Field	PROSPER.SIN.GLF.Gravity	PROSPER.SIN.GLF.ValveDepth	PROSPER.SIN.GLF.Depth[$]
2	T00_Integrated_Oil_Well.Out	Logie	0.699999988	0	0|0|0|0|0|0|0|0|0|0|
3	T01_Basic_Oil_Well.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
4	T02_Basic_Gas_Well.Out	Powderhall	0	0	0|0|0|0|0|0|0|0|0|0|
5	T03a_Water_Injector.Out		0	0	0|0|0|0|0|0|0|0|0|0|
26	T23_33a_GasLift_Quicklook.Out	Logie	0.699999988	0	3791.65|5761.76|6763.89|7092.66|0|0|0|0|0|0|
27	T23_Continuous_GasLift_Design.Out	Logie	0.699999988	0	3731.44|5671.63|6669.94|7027.49|0|0|0|0|0|0|
28	T24_Intermittent_GasLift_Design.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
29	T25_Coiled_Tubing_GasLift_Design.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
30	T26_ESP_Design.Out	Logie	0.699999988	5000	0|0|0|0|0|0|0|0|0|0|
31	T27_HSP_Design.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
32	T28_Sucker_Rod_Design.Out		0	0	0|0|0|0|0|0|0|0|0|0|
33	T29_PCP_Design.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
34	T30_Diluent_Injection_With_GasLift.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
35	T31_Modelling_Surface_Multiphase_Pumps.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
36	T32_Designing_A_Jet_Pump.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
37	T33a_GasLift_Quicklook.Out	Logie	0.699999988	0	3791.65|5761.76|6763.89|7092.66|0|0|0|0|0|0|
38	T33b_GasLift_Quicklook.Out	Logie	0.699999988	0	3731.44|5671.63|6669.94|7027.49|0|0|0|0|0|0|
39	T34_Troubleshooting_an_ESP_System.Out	Logie	0.699999988	5000	0|0|0|0|0|0|0|0|0|0|
40	T35_HSP_Quicklook.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
41	T37_Compositional_Condensate_Well.Out	Spectrum	0	0	0|0|0|0|0|0|0|0|0|0|
42	T38_Validation_of_the_Black_Oil_Retrograde_C	Spectrum	0	0	0|0|0|0|0|0|0|0|0|0|
43	T39_Compositional_CO2_Injector.Out		0	0	0|0|0|0|0|0|0|0|0|0|
44	T40_Enthalpy_Balance.Out	Spectrum	0	0	0|0|0|0|0|0|0|0|0|0|
45	T41_Improved_Approximation.Out	Spectrum	0	0	0|0|0|0|0|0|0|0|0|0|
46	T42_Steam_Injector.Out	Logie	0	0	0|0|0|0|0|0|0|0|0|0|
47	T43_Flow_Assurance.Out	Spectrum	0	0	0|0|0|0|0|0|0|0|0|0|
48	T44_Modelling_NonNewtonian_Fluids.Out	Mill	0	0	0|0|0|0|0|0|0|0|0|0|
49	T45_CBM_Dewatering_Well.Out		0	0	0|0|0|0|0|0|0|0|0|0|
50	T46_Modelling_An_Emulsion.Out	Spectrum	0	0	0|0|0|0|0|0|0|0|0|0|
51	T47_Gas_Lift_For_Gas_wells.Out	Spectrum	0.699999988	10000	0|0|0|0|0|0|0|0|0|0|

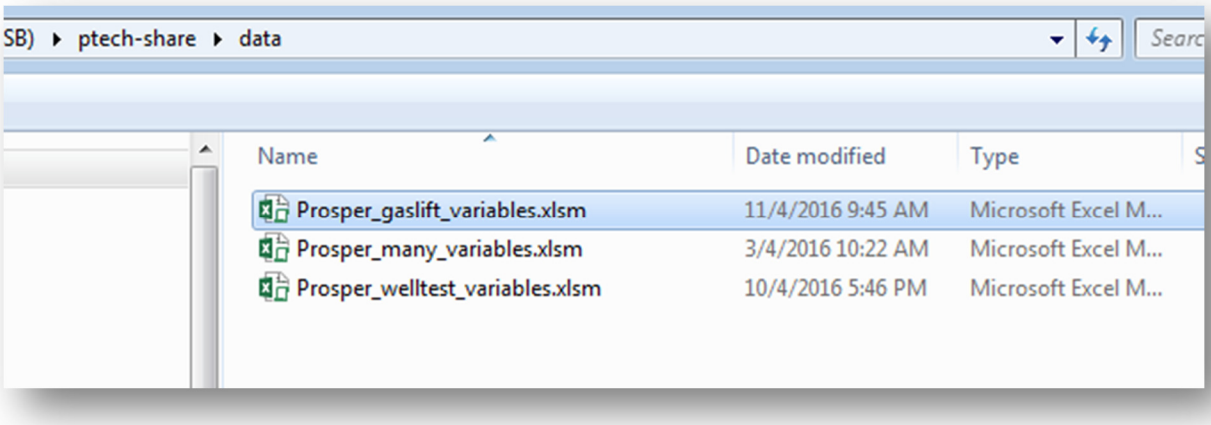
This is your well known Excel containing a dataframe structure, rows by columns. Note that the columns are OpenServer variables and the rows are well file names. In this example I used the Prosper samples provided by Petex under the installation directory shown here:



Of course this directory can be changed to whatever you want, just be sure you put all the well models you want to analyze under the same folder. They can grouped as sub-directories but all of them should be living under the same parent folder. Something like this:



One of the most inconvenient and time consuming tasks is getting a good summary of the well model variables forcing us production technologists to open a well model one by one. That is really so 20th century! Fear not. Today we can have the computer opening all the well models at once and make a summary for us in few minutes; for five wells or 100 wells. We can even configure our table of well variables in an Excel file for use and reuse. We can have a set of variable for gas lift, another for well test analysis, another one for PVT, etc. The files can be saved and look like this:



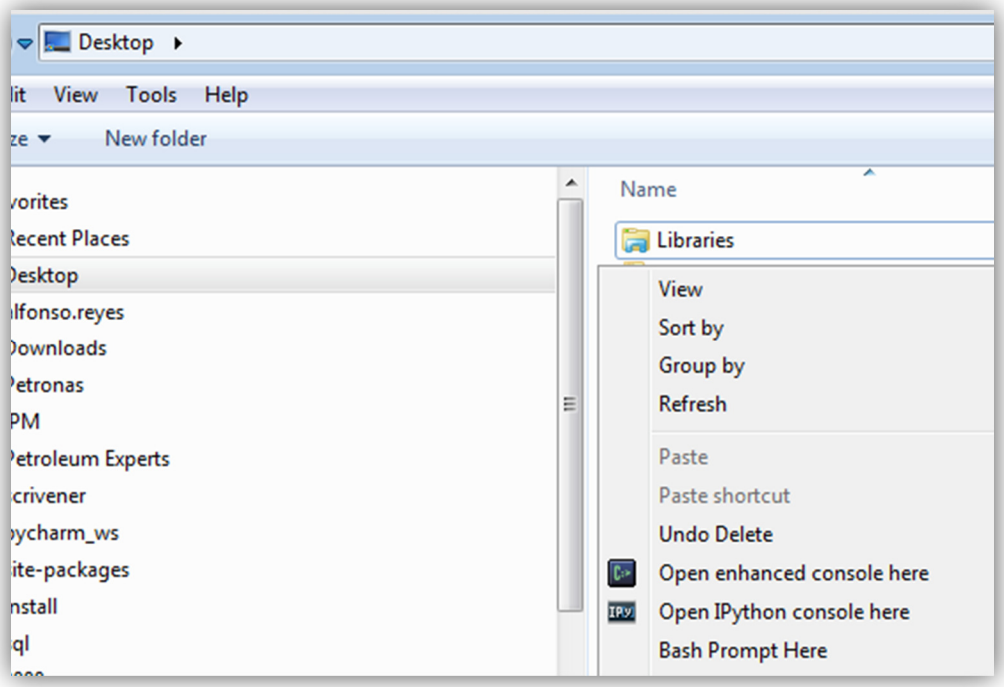
The variables file is very flexible. It can contain 10 variables, 40 or 100 variables, and is very customizable. You can add OpenServer variable, remove, or move up or down. Also, you can activate or deactivate a particular variable for a session and reactivate for the next by double-clicking the mouse on the variable. This how the variables file look like:

	A	B	C	E	F	G	H	J	M	O
1	app	top	section	scope	array_index	variable	alias	description	var_typ	openserver_command
2	PROSPER					FileName	FileName	Name of the file	str	PROSPER.FileName
3	PROSPER					FileVersion	FileVersion	Version of the file	str	PROSPER.FileVersion
4	PROSPER					FileCreationVersion	FileCreationVersion	File creation version of the file	str	PROSPER.FileCreationVersion
5	PROSPER	SIN	SUM			Well	Wellname	Well name in summary screen	str	PROSPER.SIN.SUM.Well
6	PROSPER	SIN	SUM			Company	Company_name	Company name	str	PROSPER.SIN.SUM.Company
7	PROSPER	SIN	SUM			Analyst	Analyst	Who made this model	str	PROSPER.SIN.SUM.Analyst
8	PROSPER	SIN	SUM			Field	Field	Field name in summary screen	str	PROSPER.SIN.SUM.Field
9	PROSPER	SIN	SUM			Location	Location	Location	str	PROSPER.SIN.SUM.Location
10	PROSPER	SIN	SUM			Platform	Platform	Platform	str	PROSPER.SIN.SUM.Platform
11	PROSPER	PVT	INPUT			Solgor	GOR	Gas Oil Ration @ PVT section	float	PROSPER.PVT.INPUT.Solgor
12	PROSPER	PVT	INPUT			Api	API	API	float	PROSPER.PVT.INPUT.Api
13	PROSPER	PVT	INPUT			Grvgas	SG_gas	Specific Gravity of Gas	float	PROSPER.PVT.INPUT.Grvgas
14	PROSPER	PVT	INPUT			Watsal	WaterSalinity	Water Salinity	float	PROSPER.PVT.INPUT.Watsal
15	PROSPER	PVT	INPUT			H2s	H2S	H2S	float	PROSPER.PVT.INPUT.H2s
16	PROSPER	PVT	INPUT			Co2	CO2	CO2	float	PROSPER.PVT.INPUT.Co2
17	PROSPER	PVT	INPUT			PBcorr	PB_CORR	Pb, Rs, Bo Correlation.	int	PROSPER.PVT.INPUT.PBcorr
18	PROSPER	PVT	INPUT			UOcorr	VISC_CORR	Oil Viscosity Correlation.	int	PROSPER.PVT.INPUT.UOcorr

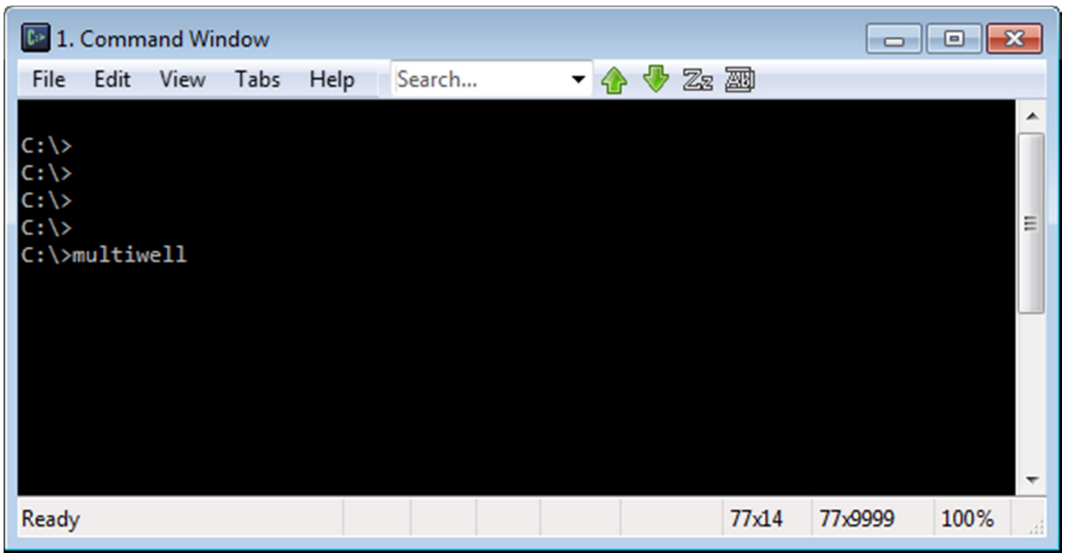
Step by Step

Start the multiwell model scanner script

If you have installed Python(x,y) in full, then you should be able to open a console by going to Windows Explorer and right-clicking on a blank area of the screen. The click on "*Open enhanced console here*" to open a console or terminal. See the installation instruction at this link.

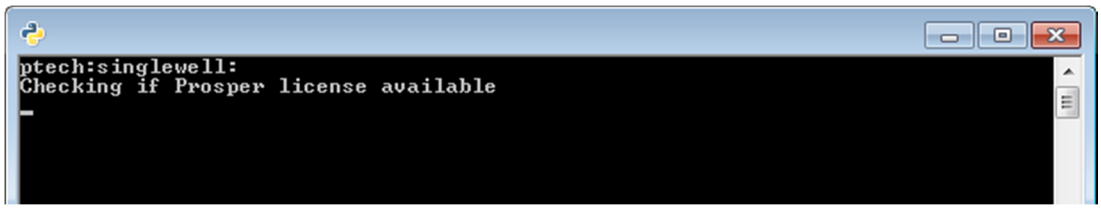


Once the terminal is opened, type the word "**multiwell**" and press Enter. See the example below:

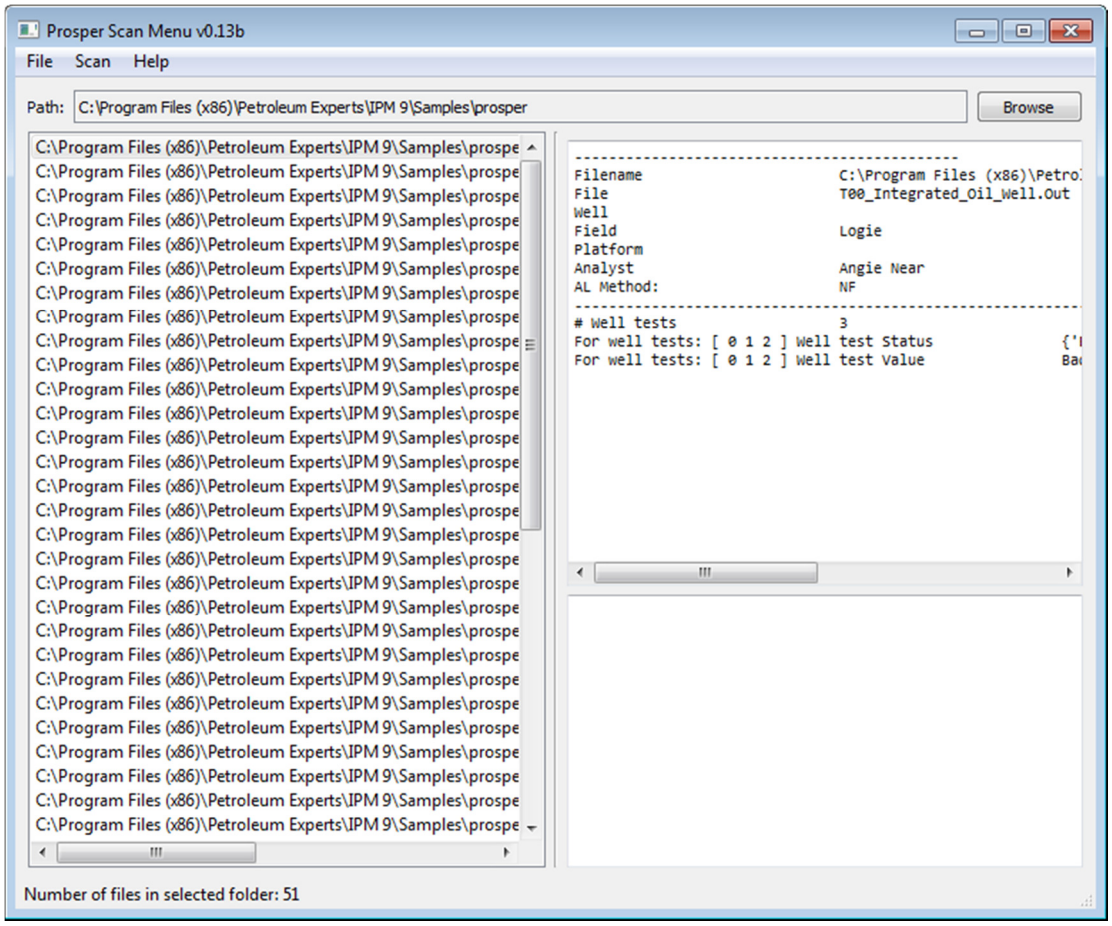


This will launch the multiwell scanner.

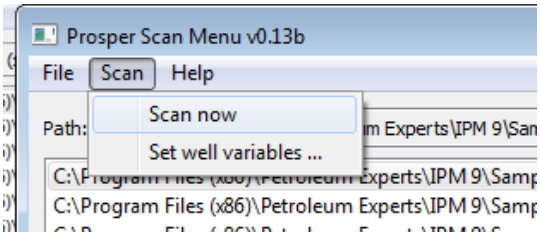
First, the script will check if a Prosper license is available, otherwise it will not launch.



If there are licenses available, the application will open. This is the main screen:



The default directory is the samples for the Prosper folder. Change it by clicking on the **Browse** button on the top right. Then select the folder for your well model. Once you selected the folder, then go to the **Scan** menu option and click on **Scan now**.



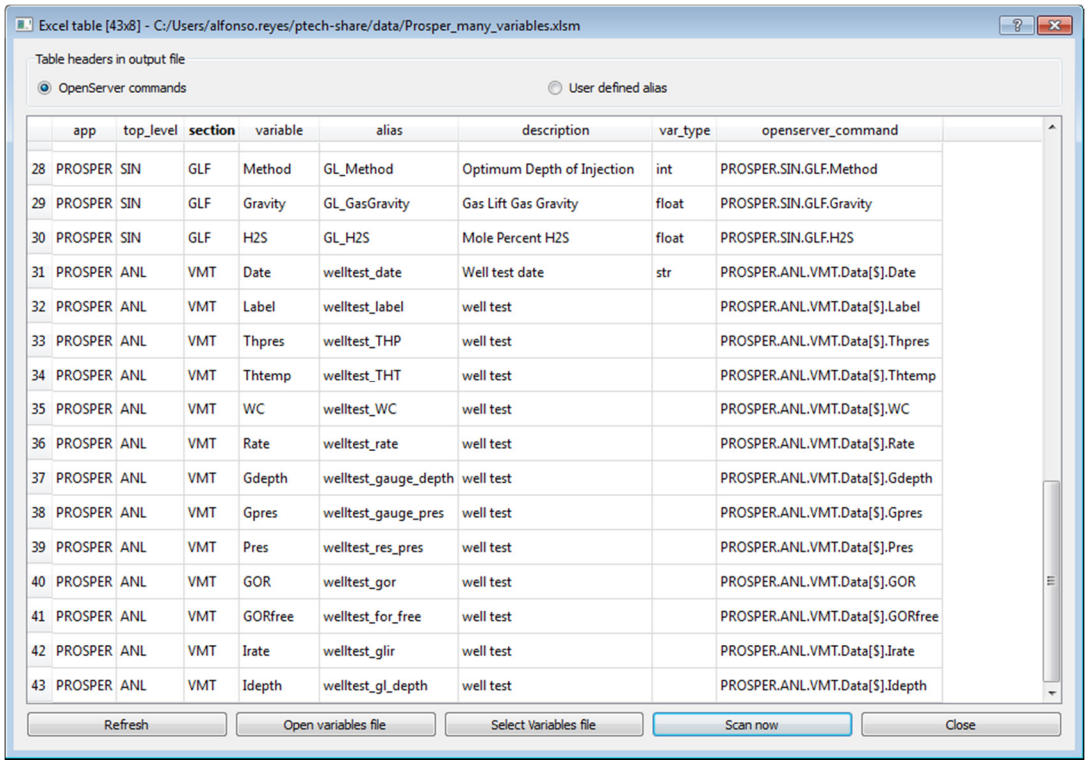
You will see that the script start scanning all your well models. If you chose just the default samples folder, then at the end of the scanning an Excel workbook will open showing you a well summary. A screenshot would look like this:

	A	P	Q	R	S	U	X	AB	AF	
1	modelTailname	T.INPUT.Solgor	T.INPUT.Api	PVT.INPUT.Grvgas	PVT.INPUT.Watsal	PVT.INPUT.Co2	SIN.IPR.Single.Wc	SIN.EQP.GEO.Htc	PROSPER.ANL.VMT.Data[$].Date	
2	T00_Integrated_Oil_Well.Out	800	37	0.75999999	23000	0	0	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
22	T19_Matching_Well_Test_For_Oil_Well.Out	800	37	0.75999999	23000	0	0	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
23	T20_Matching_Well_Test_For_Gas_Well.Out	0	50	0.579999983	100000	0.5	0	3.120496035	18/02/2007|20/03/2007|14/04/2007|	Te
24	T21_Matching_Water_Injection_Well_Test.Out	800	37	0.699999988	200000	0	100	8.88293457	17/08/2011|23/09/2011|07/12/2011|	te
25	T22_Matching_Gas_Injection_Well_Tests.Out	0	67	0.768999994	100000	0.368000001	0	2.880125999	12/07/2011|19/08/2011|29/11/2011|	te
26	T23_33a_GasLift_Quicklook.Out	800	37	0.75999999	23000	0	80	8.703613281	16/03/2011|21/05/2011|07/10/2011|15/10/2015|	Te
27	T23_Continuous_GasLift_Design.Out	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
28	T24_Intermittent_GasLift_Design.Out	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
29	T25_Coiled_Tubing_GasLift_Design.Out	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
30	T26_ESP_Design.Out	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
31	T27_HSP_Design.Out	800	37	0.75999999	23000	0	40	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
32	T28_Sucker_Rod_Design.Out	160	25	0.680000007	80000	0	80	8		
33	T29_PCP_Design.Out	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
34	T30_Diluent_Injection_With_GasLift.Out	800	37	0.75999999	180000	0	54	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
35	T31_Modelling_Surface_Multiphase_Pumps.Ou	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
36	T32_Designing_A_Jet_Pump.Out	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
37	T33a_GasLift_Quicklook.Out	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
38	T33b_GasLift_Quicklook.Out	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
39	T34_Troubleshooting_an_ESP_System.Out	800	37	0.75999999	23000	0	80	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
40	T35_HSP_Quicklook.Out	800	37	0.75999999	23000	0	40	8.641601563	16/03/2011|21/05/2011|07/10/2011|	Te
41	T37_Compositional_Condensate_Well.Out	0	50.794956	0.809115291	20000	0	0	5		
42	T38_Validation_of_the_Black_Oil_Retrograde_(	0	43.560001	0.809115291	20000	2.329999924	0	5		
43	T39_Compositional_CO2_Injector.Out	0	40	1.519157767	20000	0	100	3		
44	T40_Enthalpy_Balance.Out	0	43.560001	0.809115291	20000	2.329999924	0	5		
45	T41_Improved_Approximation.Out	0	43.566799	0.809115291	20000	2.329999924	0	5		
46	T42_Steam_Injector.Out	0	37	0.699999988	200000	0	100	8		

Again, the rows are the wells and the columns the variables.

What's next?

Once you run your first multiwell scan you may start playing with other options such as selecting the variables or selecting variables inside the file.



In a next tutorial I will explore details and uses of the script.