

Geschwindigkeit in Power Query

Excelstammtisch vom 22.04.2024

1. Chandeeep Chhabra

Auf dem Blog von Chandeeep finden sich interessante Beispiele zu Power Query (aber auch zu Excel und DAX). Man findet seinen Blog unter

<https://goodly.co.in/>

Beispielsweise folgendes Problem:

The screenshot shows a file explorer window with the following files:

Name	Änderungsdatum	Typ	Größe
Altona.xlsx	22.04.2024 10:39	Microsoft Excel-A...	11 KB
Bad Segeberg.xlsx	22.04.2024 10:39	Microsoft Excel-A...	11 KB
Harburg.xlsx	18.11.2018 17:22	Microsoft Excel-A...	12 KB
Norderstedt.xlsx	22.04.2024 10:39	Microsoft Excel-A...	11 KB

Below the file explorer, two Excel spreadsheets are shown. The first spreadsheet, 'Altona', has the following data:

Position	Plan	Ist	Ist-Plan
Roherttrag	766,2	790,7	24,5
LKW-Löhne	164,7	144,0	-20,7
LKW-Kosten c	87,8	122,1	34,3
Spediteure	20,3	35,1	14,8
Leistungskost	272,8	301,2	28,4
Deckungsbeitr	493,4	489,5	-3,9
Gehälter	213,3	221,7	8,4
Zinsen	16,9	24,3	7,4
Gehälter + Zin	230,2	246,0	15,8
Raumkosten	29,1	27,1	-2,0
EDV-Kosten	16,2	22,9	6,7
Büromaterial-	2,7	8,6	5,9
Beratungskost	2,7	13,3	10,6
Hermes	0,0	2,7	2,7
Versicherung	7,5	3,8	-3,7
Instandhaltung	23,7	16,7	-7,0
Zinsen	11,6	14,0	2,4

The second spreadsheet, 'Bad Segeberg', has the following data:

Position	Plan	Ist	Ist-Plan
Roherttrag	420,0	415,0	-5,0
LKW-Löhne	92,1	75,0	-17,1
LKW-Kosten ohne AfA	45,6	60,0	14,4
Spediteure	10,9	19,5	8,6
Leistungskosten	148,6	154,5	5,9
Deckungsbeitrag I	271,4	260,5	-10,9
Gehälter	120,0	121,0	1,0
Zinsen	10,6	13,0	2,4
Gehälter + Zinsen	130,6	134,0	3,4
Raumkosten	16,1	13,9	-2,2
EDV-Kosten	8,0	11,8	3,8
Büromaterial- und Leasing	2,0	5,0	3,0
Beratungskosten	1,5	7,5	6,0
Hermes	0,6	1,4	0,8

In einem Ordner befinden sich mehrere Dateien, die jeweils einen unterschiedlichen Header haben. Das heißt: eine unterschiedliche Anzahl leere Zeilen vor den Daten, die ausgewertet werden sollen. Um die Daten aus allen Dateien zusammen zufassen wäre ich wie folgt vorgegangen:

1.1. Lösung I

Im ersten Schritt wird auf den Ordner zugegriffen:

```
= Folder.Files("D:\Pfad\25_Plan_IstAusZweiDateien")
```

In einer neuen Spalte wird mit Excel.Workbook der (Datei-)Inhalt aus der Spalte Content herausgelöst:

```
= Table.AddColumn(Quelle, "Benutzerdefiniert",  
    each Excel.Workbook([Content]))
```

Die „Tabelle“ wird „geöffnet“:

Table.AddColumn(Quelle, "Benutzerdefiniert", each Excel.Workbook([Content]))						
	Extension	Date accessed	Date modified	Date created	Attributes	Folder Path
1	.xlsx	24.04.2024 22:28:30	18.11.2018 17:21:50	21.03.2023 20:56:06	Record	D:\Eigene Dateien\Übungsdateien\Excel\PowerQuery\PowerQuery\2...
2	.xlsx	24.04.2024 22:28:30	18.11.2018 17:22:10	21.03.2023 20:56:07	Record	D:\Eigene Dateien\Übungsdateien\Excel\PowerQuery\PowerQuery\2...
3	.xlsx	24.04.2024 22:28:30	18.11.2018 17:22:38	21.03.2023 20:56:07	Record	D:\Eigene Dateien\Übungsdateien\Excel\PowerQuery\PowerQuery\2...
4	.xlsx	24.04.2024 22:28:30	18.11.2018 17:22:44	21.03.2023 20:56:07	Record	D:\Eigene Dateien\Übungsdateien\Excel\PowerQuery\PowerQuery\2...

Name	Data	Item	Kind	Hidden
Altona	Table	Altona	Sheet	FALSE
Deckungsbeitrag_Altona	Table	Deckungsbeitrag_Altona	DefinedName	FALSE

Da sich in den Dateien auch Namen befinden wird aus der Spalte „Kind“ das „Sheet“ herausgefiltert:

ABC 123	Kind	ABC 123
	Sheet	
	DefinedName	
	Sheet	
erg	DefinedName	
	Sheet	
	DefinedName	
	Sheet	
edt	DefinedName	

Anschließend kann man die anderen Spalten – außer der Spalte „Data“ löschen:

= Table.SelectColumns("#Gefilterte Zeilen",{"Data"})			
ABC 123	Data		
1	Table		
2	Table		
3	Table		
4	Table		

Column1	Column2	Column3	Column4
Altona	null	null	null
Position	Plan	Ist	Ist-Plan
Rohrertrag	766,2	790,7	24,5
LKW-Löhne	164,7	144	-20,7
LKW-Kosten ohne AfA	87,8	122,1	34,3
Spediteure	20,3	35,1	14,8
Leistungskosten	272,8	301,2	28,4
Deckungsbeitrag I	493,4	489,5	-3,9
Gehälter	213,3	221,7	8,4
Zinsen	16,9	24,3	7,4
Gehälter + Zinsen	230,2	246	15,8
Raumkosten	29,1	27,1	-2
EDV-Kosten	16,2	22,9	6,7
Büromaterial- und Leasing	2,7	8,6	5,9
Beratungskosten	2,7	13,3	10,6
Hermes	0	2,7	2,7
Versicherung und Beiträge	7,5	3,8	-3,7
Instandhaltung und Werbung	23,7	16,7	-7
Porto und Telefon	11,5	14	2,5
sonstige Kosten	14,2	27,6	13,4

Und diese Spalte „öffnen“:

Table.ExpandTableColumn("#Andere entfernte Spalten", "Data", {"Column1", "Column2", "Column3", "Column4"}, {"Column1", "Column2", "Column3", "Column4"})

Column1	Column2	Column3	Column4
1	Altona	null	null
2	Position	Plan	Ist
3	Rohrertrag	766,2	790,7
4	LKW-Löhne	164,7	144
5	LKW-Kosten ohne AfA	87,8	122,1
6	Spediteure	20,3	35,1
7	Leistungskosten	272,8	301,2
8	Deckungsbeitrag I	493,4	489,5
9	Gehälter	213,3	221,7
10	Zinsen	16,9	24,3
11	Gehälter + Zinsen	230,2	246
12	Raumkosten	29,1	27,1
13	EDV-Kosten	16,2	22,9
14	Büromaterial- und Leasing	2,7	8,6
15	Beratungskosten	2,7	13,3
16	Hermes	0	2,7
17	Versicherung und Beiträge	7,5	3,8
18	Instandhaltung und Werbung	23,7	16,7
19	Porto und Telefon	11,5	14
20	sonstige Kosten	14,2	27,6
21	Bereitschaftskosten I	107,6	136,7
22	Deckungsbeitrag II	155,6	106,8
23	AfA	108	127,7
24	LKW-Leasing	50,7	48,3
25	Gewerbesteuer	0,7	0,6
26	Bereitschaftskosten II	159,4	176,6
27	Deckungsbeitrag III	-3,8	-69,8
28	Bad Segeberg	null	null
29	Position	Plan	Ist
30	Rohrertrag	420	415
31	LKW-Löhne	92,1	75
32	LKW-Kosten ohne AfA	45,6	60
33	Spediteure	10,9	19,5
34	Leistungskosten	148,6	154,5
35	Deckungsbeitrag I	271,4	260,5
36	Gehälter	120	121
37	Zinsen	10,6	13
38	Gehälter + Zinsen	130,6	134
39	Raumkosten	16,1	13,9
40	EDV-Kosten	8	11,8
41	Büromaterial- und Leasing	2	5
42	Beratungskosten	1,5	7,5
43	Harmer	0,6	1,4

Mit einem geschickten Filter können die überflüssigen Zeilen gelöscht werden:

Table.SelectRows("#Erweiterte Data", each ([Column2] <> null) and ([Column1] <> "Position"))

Column1	Column2	Column3	Column4
1	Rohrertrag	766,2	790,7
2	LKW-Löhne	164,7	144
3	LKW-Kosten ohne AfA	87,8	122,1
4	Spediteure	20,3	35,1
5	Leistungskosten	272,8	301,2
6	Deckungsbeitrag I	493,4	489,5
7	Gehälter	213,3	221,7
8	Zinsen	16,9	24,3
9	Gehälter + Zinsen	230,2	246
10	Raumkosten	29,1	27,1

1.2. Lösung II

Chandeep geht einen anderen Weg.

Man kann auf eine Tabelle einen Drilldown ausführen und sich so eine Tabelle genauer „ansehen“:

Data	Item
Table	Sh
Table	Sh
Table	Sh
Table	Sh

Fügt man eine neue Spalte hinzu mit dem Befehl

`Record.ToList(_)`

So erhält man eine Liste, welche sämtliche Zelleninformationen der aktuellen Zeile enthält:

24	LKW-Leasing	50,7	48,3	-2,4	List
25	Gewerbesteuer	0,7	0,6	-0,1	List
26	Bereitschaftskosten II	159,4	176,6	17,2	List
27	Deckungsbeitrag III	-3,8	-69,8	-66	List

List
Bereitschaftskosten II
159,4
176,6
17,2

Erweitert man den Befehl auf:

`Record.ToList(_) = {"Position", "Plan", "Ist", "Ist-Plan"}`

Erhält man True oder False – je nachdem ob diese vier Texte in der Zeile auftauchen:

`= Table.AddColumn(Data, "Benutzerdefiniert", each Record.ToList(_) = {"Position", "Plan", "Ist", "Ist-Plan"})`

	Column1	Column2	Column3	Column4	Benutzerdefiniert
1	Altona	null	null	null	FALSE
2	Position	Plan	Ist	Ist-Plan	TRUE
3	Rohrertrag	766,2	790,7	24,5	FALSE
4	LKW-Löhne	164,7	144	-20,7	FALSE
5	LKW-Kosten ohne AfA	87,8	122,1	34,3	FALSE
6	Spediteure	20,3	35,1	14,8	FALSE
7	Leistungskosten	272,8	301,2	28,4	FALSE
8	Deckungsbeitrag I	493,4	489,5	-3,9	FALSE

Da alle leeren Zeilen (bis zur ersten False-Zelle) gelöscht werden sollen, muss die Bedingung umgekehrt werden:

```
= Table.AddColumn(Data, "Benutzerdefiniert",
    each Record.ToList(_) <>
    {"Position", "Plan", "Ist", "Ist-Plan"}
)
```

Der Befehl „erste Zeile entfernen“ lautet:

```
= Table.Skip("#Hinzugefügte benutzerdefinierte Spalte1",1)
```

Der letzte Parameter kann eine Zahl oder einen Wahrheitswert darstellen:

`= Table.Skip("#Hinzugefügte benutzerdefinierte Spalte1",`

`Table.Skip(table as table, countOrCondition as any)`

any

Gibt eine Tabelle zurück, in der die ersten x Zeilen übersprungen wurden.

	Column1	Column2	Column3	Column4	Benutzerdefiniert
1	Position	Plan	Ist		
2	Rohrertrag	766,2			
3	LKW-Löhne	164,7			
4	LKW-Kosten ohne AfA	87,8	122,1	34,3	TRUE

Verwendet man dort die Benutzerdefinierte Spalte:

```
= Table.Skip("#Hinzugefügte benutzerdefinierte Spalte1",
    each [Benutzerdefiniert])
```

Werden alle Zeilen bis zur Überschriftszeile gelöscht:

fx = Table.Skip("#Hinzugefügte benutzerdefinierte Spalte1", each [Benutzerdefiniert])

ABC 123	Column1	ABC 123	Column2	ABC 123	Column3	ABC 123	Column4	ABC 123	Benutzerdefiniert
1	Position		Plan		Ist		Ist-Plan		FALSE
2	Rohrertrag			766,2		790,7		24,5	TRUE
3	LKW-Löhne			164,7		144		-20,7	TRUE
4	LKW-Kosten ohne AfA			87,8		122,1		34,3	TRUE
5	Spediteure			20,3		35,1		14,8	TRUE
6	Leistungskosten			272,8		301,2		28,4	TRUE
7	Restwert			402,4		402,4		0,0	TRUE

Die Hilfsspalte kann gelöscht werden; die erste Zeile wird zur Überschrift der Tabelle:

```
= Table.PromoteHeaders("#Entfernte Spalten",  
[PromoteAllScalars=true])
```

fx = Table.PromoteHeaders("#Entfernte Spalten", [PromoteAllScalars=true])

ABC 123	Position	ABC 123	Plan	ABC 123	Ist	ABC 123	Ist-Plan
1	Rohrertrag			766,2		790,7	24,5
2	LKW-Löhne			164,7		144	-20,7

Diese Befehle

- each Record.ToList(_) <>
 {"Position", "Plan", "Ist", "Ist-Plan"}
- Table.Skip
- Table.PromoteHeaders

Können zu einem Befehl zusammengefügt werden und auf die ganze Spalte angewendet werden:

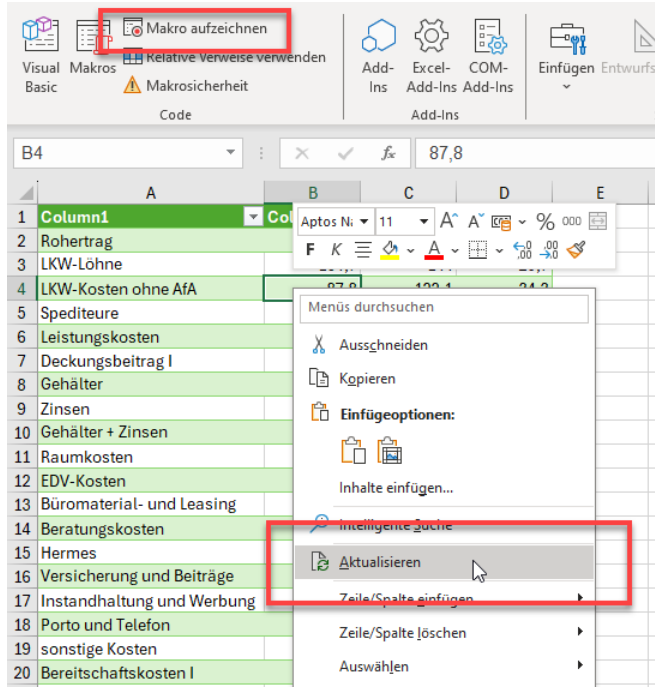
```
= Table.AddColumn(  
    #"Gefilterte Zeilen",  
    "Benutzerdefiniert",  
    each Table.PromoteHeaders(  
        Table.Skip([Data],  
            each  
                (Record.ToList(_) <>  
                    {"Position", "Plan", "Ist", "Ist-Plan"})  
            )  
        )  
    )  
)
```

So haben alle Tabellen die korrekte Form:

1.4. Ein Messwerkzeug für die Geschwindigkeit?

Zeichnet man mit dem Makrorekorder den Befehl des Aktualisierens auf, erhält man folgenden VBA-Befehl:

```
Selection.ListObject.QueryTable.Refresh BackgroundQuery:=False
```



Und damit kann leicht ein kleiner Timer gebaut werden:

```
Sub TabelleAktualisieren()
```

```
    Dim dblBeginn As Double
```

```
    Dim dblEnde As Double
```

```
    Dim i As Integer
```

```
    Dim strAusgabe As String
```

```
    For i = 1 To ThisWorkbook.Worksheets.Count
```

```
        dblBeginn = Now
```

```
        ThisWorkbook.Worksheets(i).Range("A4").ListObject.  
            QueryTable.Refresh BackgroundQuery:=False
```

```
        dblEnde = Now
```

```
strAusgabe = strAusgabe & vbCr & _  
    ThisWorkbook.Worksheets(i).Name & ": " & _  
    Format(dblEnde - dblBeginn, "hh:mm:ss.000")  
  
Next i  
  
MsgBox strAusgabe
```

End Sub

1.5. Große Datenmengen

Über die Seite <https://www.fakenamegenerator.com/> kann man Namen generieren lassen. Sie werden per Mail zugeschickt:

The screenshot shows the 'FAKE NAME GENERATOR™' website. The main navigation bar includes links for 'Name Generator', 'Free Tools', 'Order in Bulk', 'Smiley Generator', and 'FAQ'. The 'Order Bulk Identities' section is active, displaying a form with the following steps:

- Step 1 - Read and agree to terms of service**: A checkbox for 'I agree to the terms of service and understand that all generated information is fake.' is present.
- Step 2 - Choose output format and compression**: 'Output Format' is set to 'Comma separated (.csv)' and 'Compression' is set to '.zip'.
- Step 3 - Choose name sets, countries, gender, and age**:
 - Name set**: A list box containing 'French', 'German', 'Greenland', 'Hispanic', 'Hobbit', and 'Hungarian'.
 - Country**: A list box containing 'Australia', 'Austria', 'Belgium', 'Brazil', and 'Canada'.
 - Gender**: A slider set to 50% Male and 50% Female.
 - Age**: A slider set to 19 - 85 years old.
- Step 4 - Choose fields to include**: A section with instructions on how to select fields for the output.

Damit lassen sich (schnell) große Datenmengen erzeugen. Wie beispielsweise hier die Datei, die 300.000 Datensätze (und 41 Spalten) beinhaltet.

1.6. Kein Switch?

Chandeep bemängelt, dass es in Power Query keinen Switch (oder Select Case oder WENNS) Befehl gibt. Also selbst schreiben.

Statt verschachtelter Wenn-Funktionen, die beispielsweise so aussehen könnten:


```
if Date.Year([Geburtsdatum]) > 2020 then "Kategorie A"
else if Date.Year([Geburtsdatum]) > 2000 then "Kategorie B"
else if Date.Year([Geburtsdatum]) > 1980 then "Kategorie C"
else if Date.Year([Geburtsdatum]) > 1960 then "Kategorie D"
else if Date.Year([Geburtsdatum]) > 1940 then "Kategorie E"
else if Date.Year([Geburtsdatum]) > 1920 then "Kategorie F"
else "Kategorie X"
```

verwendet Chandeeep eine an Switch angelehnte selbst geschriebene Funktion, die auch ausgelagert werden kann:

let

```
Bedingungen = {
    Date.Year([Geburtsdatum]) > 2020,
    Date.Year([Geburtsdatum]) > 2000 and
        Date.Year([Geburtsdatum]) <= 2020,
    Date.Year([Geburtsdatum]) > 1980 and
        Date.Year([Geburtsdatum]) <= 2000,
    Date.Year([Geburtsdatum]) > 1960 and
        Date.Year([Geburtsdatum]) <= 1980,
    Date.Year([Geburtsdatum]) > 1940 and
        Date.Year([Geburtsdatum]) <= 1960,
    Date.Year([Geburtsdatum]) > 1920 and
        Date.Year([Geburtsdatum]) <= 1940,
    Date.Year([Geburtsdatum]) <= 1920
},
Ergebnisse = {"Kategorie A", "Kategorie B", "Kategorie C",
    "Kategorie D", "Kategorie E", "Kategorie F", "Kategorie X"}
in
Ergebnisse{List.PositionOf(Bedingungen, true)}
```

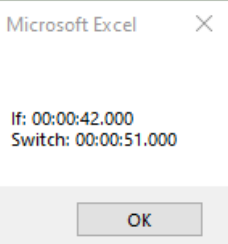
Das Ergebnis sieht folgendermaßen aus:

Jahrgang	
Kategorie D	
Kategorie F	
Kategorie F	
Kategorie F	
Kategorie F	
Kategorie F	
Kategorie C	
Kategorie C	
Kategorie E	
Kategorie E	
Kategorie C	
Kategorie C	
Kategorie C	
Kategorie C	
Kategorie E	
Kategorie D	
Kategorie C	
Kategorie C	
Kategorie D	
Kategorie F	
Kategorie F	
Kategorie F	
Kategorie D	
Kategorie C	
Kategorie F	

1.7. Wer ist schneller?

Und nun gilt es ein Wettrennen zu starten: Wer ist schneller – die selbstgebaute Funktion oder die verschachtelten If-Anweisungen?

	A	B	C	D	E
1					
2		Timer			
3					
4	Nummer	Geschlecht	Sprache	Anrede	Titel
5	3088		10 deutsch	Frau	
6	13906		20 d		
7	17561		20 d		
8	20219		20 d		
9	21040		20 d		
10	22953		10 d		
11	25951		20 d		
12	27088		20 d		
13	28828		20 d		
14	31967		10 deutsch	Frau	



Die Antwort: Mehrere IFs sind schneller als eine Switch-Funktion!