

# Access/ACE und große Datenmengen

Konzept für einen Beitrag auf der AEK 2015

Theodor Weidmann, 06.09.2015

## Inhalt

1	Vorstellung und Einordnung des Themas	2
1.1	Vorstellung	2
1.2	Mythen zu Access im LAN	2
1.3	Warum keine große Datenbank?	2
1.4	Access kann mehr als allgemein bekannt ist	3
2	2 GB-Grenze	3
3	Aspekte zum Datenvolumen von Access-Datenbanken	4
3.1	ACE vergrößert das Datenvolumen	4
3.2	Platzbedarf für Feldtypen	6
4	Strategien, mit der 2-GB-Grenze umzugehen	7
4.1	1 Tabelle – 1 ACCDB	7
4.2	Striktes Datenmodell Star bzw. Snowflake	7
4.3	Gleichartige Daten portionieren	7
5	Alternative Datenmodelle	8
5.1	Anfänge, Irrtümer	8
5.2	Beschäftigung mit der Data Warehouse Welt	8
5.3	Erfahrungen mit SAS, Data Warehouse, Cubes	8
5.4	Star-Schema	8
5.5	Snowflake-Schema	9
5.6	Kennzahlen-Typ als Dimension	10
5.7	Möglichkeit, dieses Kennzahlen-Modell für OLTP-Systeme anzuwenden	11
5.8	Resümee zur Datenstruktur	12
6	Mit Access lokale Ressourcen nutzen	13
7	Typisches Vorgehen beim Einlesen	13
7.1	Tool zum Teilen großer Textdateien	15
8	Einlesen von Text-Daten	16
8.1	Bispiele für verschiedene Szenarien	16
8.2	Anforderungen für Einlesen von Daten	16
8.3	Importspezifikationen	16
8.4	Alternative: Einlesen mit TextStream	18
8.5	Fazit zu Import-Spezifikationen	19
9	Verwendung von großen Datenmengen	20
9.1	Beispiel einer Abfragemaske	20
9.2	Access-Modul myViews.accdb als Mittler zwischen Daten und Auswertungen	20
10	Access als Datenlieferant für PowerPivot	21
10.1	Beschränkungen von PowerPivot beim Einlesen von Daten	21
10.2	Ausgleich dieser PowerPivot-Schwächen mit Access	21
11	Access versus PowerPivot zum Auswerten von Daten	25
11.1	PowerPivot hat auch seine Grenzen	25
11.2	Vorteile von PowerPivot	26
11.3	Nachteile von PowerPivot	26
12	Fazit	26

# 1 Vorstellung und Einordnung des Themas

## 1.1 Vorstellung

In der IT seit 1985

selbständig seit 1990

Arbeiten mit Access seit 2000

2005 – 2009 Data Ware House Projekt mit SAS für Telekom

Mit Access habe ich mehrere Projekte zum Thema Daten einlesen, aufbereiten, laden gemacht und mich mit Tools zum Auswerten beschäftigt.

Aus diesen Projekten habe ich meine Erfahrungen zu Access und großen Daten. Die Branchen sind Musikindustrie, Pharma-Großhandel und Telekommunikation. Daraus stammen auch meine

Beispiele:

- aus der Musikindustrie  
z.B. Sendeminuten auf allen deutschen Hörfunksendern für Songs, die mein Auftraggeber unter Vertrag hat. Die Daten stammen von der GEMA (CSV)
- vom Pharma-Großhandel  
Verkäufe für ca. 250.000 Artikel, monatliche Summen, Gesamtmarkt und Anteil meines Auftraggebers  
Datensammlung ab 2013 bis 2015.07, monatlich

Es geht hier hauptsächlich um die Fähigkeiten der Access Database Engine (ACE) zur Datenhaltung und zum Datenzugriff. Im Folgenden wird oft „ACE“ und „Access“ synonym verwendet.

## 1.2 Mythen zu Access im LAN

Wenn man im Zusammenhang mit Access von mehr als 100.000 Sätzen spricht, bekommt man diese typischen Einwände:

- Access kann doch nur kleine Datenmengen verarbeiten
- Access ist für große Datenmengen zu langsam
- Da gibt es doch diese 2-GB-Grenze für alle Daten

Für Auswertungen werden alle Sätze einer Tabelle übers Netz übertragen.

„da ja bei jedem Zugriff eines Clients über Frontend die gesamte Tabelle übers Netzwerk übertragen wird, wenn ich Access richtig verstehe, und erst lokal ausgewertet wird.“

## 1.3 Warum keine große Datenbank?

Warum Access als Backend und nicht SQL-Server oder Oracle?

Mit Access als Backend ist man in der Entwicklungsphase viel flexibler.

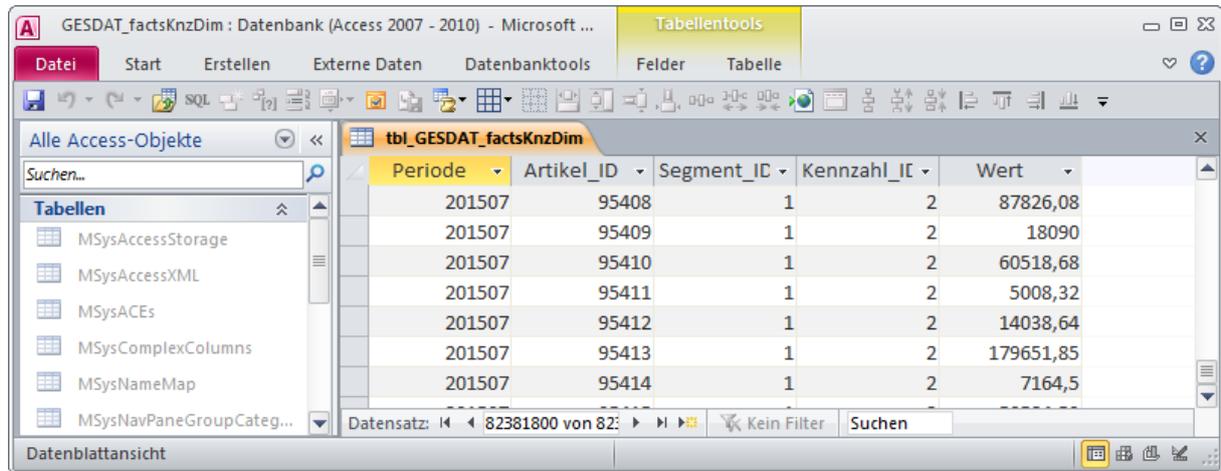
Ich würde Projekte verlieren, weil meine Auftraggeber andere Dienstleister für diese Datenbanken haben. (Ich stehe allerdings jetzt auch vor einer Portierung nach Oracle.)

Zum Teil mache ich auch Prototypen und übergebe dann, wenn es mit Access nicht mehr weitergeht.

Bitte kein Aufstöhnen: „SQL-Server macht das automatisch und viel besser“ „Mit Oracle ist das alles kein Problem“. Das ist mir alles bewusst.

## 1.4 Access kann mehr als allgemein bekannt ist

Demonstration einer großen Tabelle mit 80 Mio. Sätzen (und mehr)



Periode	Artikel_ID	Segment_ID	Kennzahl_IC	Wert
201507	95408	1	2	87826,08
201507	95409	1	2	18090
201507	95410	1	2	60518,68
201507	95411	1	2	5008,32
201507	95412	1	2	14038,64
201507	95413	1	2	179651,85
201507	95414	1	2	7164,5

Die ACCDB enthält nur diese eine Tabelle. Sie ist 1,86 GB groß.

Demonstration des Verhaltens von Access:

Sofortige Anzeige der ersten Daten, Nachladen der restlichen Daten

Große Tabellen zeigen auch im LAN einen relativ schnellen Zugriff, wenn Indices vorhanden sind.

Im WAN ist die Arbeit mit großen Datenmengen zu langsam.

## 2 2 GB-Grenze

2 GB ist die Grenze der physischen Größe einer x.accdb

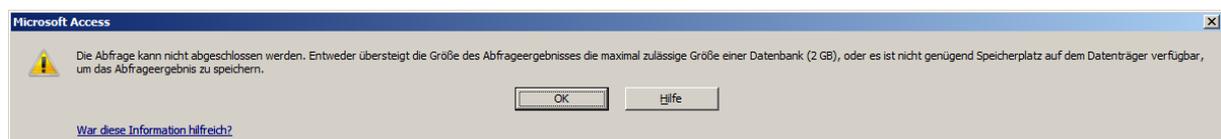
Die 2-GB-Grenze tritt aber auch bei folgendem Setting auf:

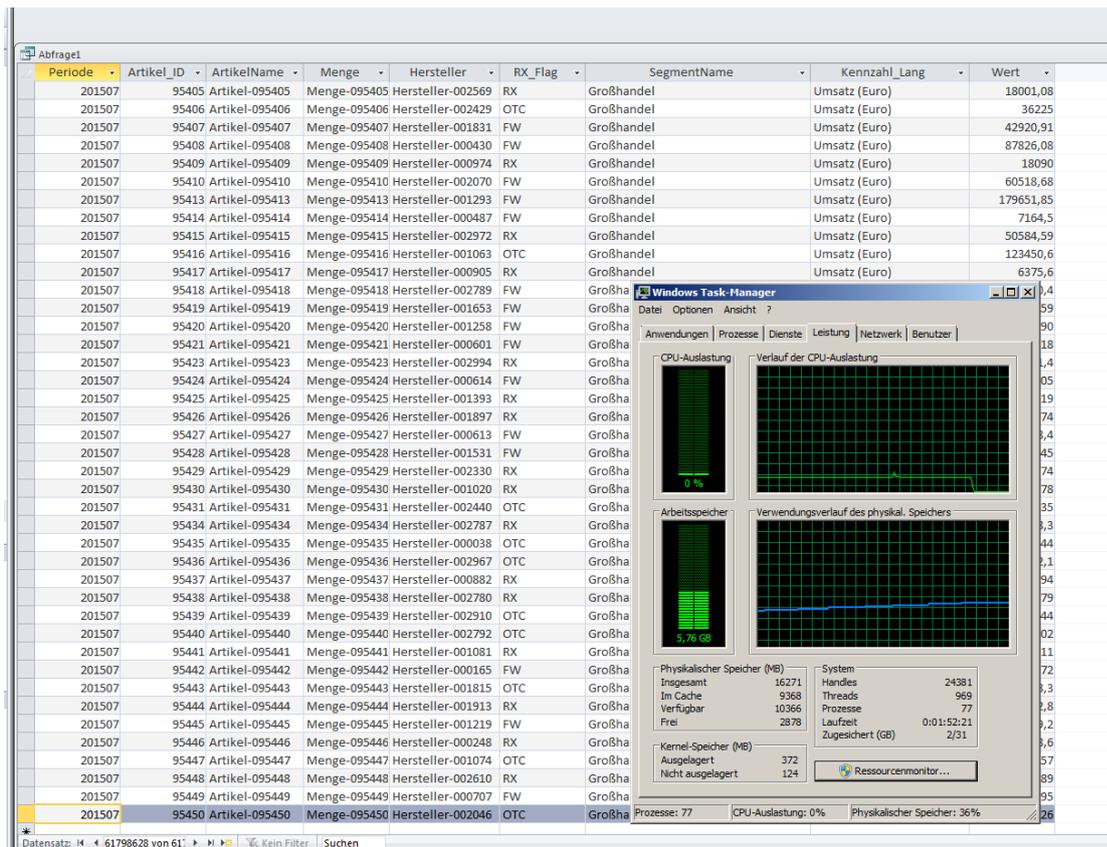
[Link auf eine CSV-Datei](#)

Eine sehr große CSV-Datei wird per Import-Spezifikation eingebunden. Die Datei wird in Access geöffnet. Daten werden angezeigt. Beim Nachladen im Hintergrund erscheint die Fehlermeldung zu der 2-GB-Grenze.

[Ergebnismenge einer Abfrage:](#)

Die 2-GB-Grenze trifft auch auf die Ergebnismenge einer Abfrage zu:





Die 2-GB-Grenze ist wohl verankert in der Größe des Arbeitsspeicers, den Access handhaben kann.

Es fällt auf, wie schwer Hintergrund-Informationen zur 2-GB-Grenze im Netz zu bekommen sind.

Die 2-GB-Grenze hat viele Nachteile.

Ein Vorteil ist, dass man sich intensiv mit der Speicherung von Daten beschäftigen muss und zu sauberem Arbeiten gezwungen ist.

(Hinter den „breiten Schultern“ des SQL-Servers verbirgt sich oft viel Mist an Datenstrukturen.)

### 3 Aspekte zum Datenvolumen von Access-Datenbanken

#### 3.1 ACE vergrößert das Datenvolumen

##### 3.1.1 Daten-Prozesse

Update-Abfragen

Löschprozesse

usw.

Es wird in interne, nicht sichtbare Objekte geschrieben. Dies hat oft mit Indices zu tun. Deshalb:

Vor Massen-Operationen Indices entfernen

Hinweise auf nicht geschlossene Recordsets oder QueryDefs

<https://support.microsoft.com/en-us/kb/289562#/en-us/kb/289562>

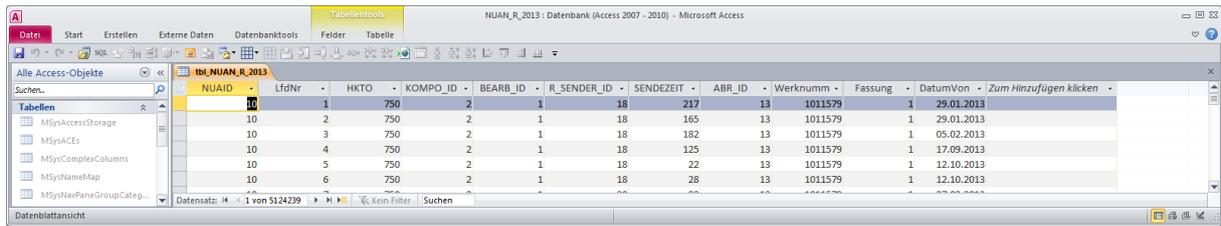
*„If you do not release a recordset's memory each time that you loop through the recordset code, DAO may recompile, using more memory and increasing the size of the database.“*

Strategien, um hier Fehler zu vermeiden: Schreiben in Tabellen mit SQLs statt mit Recordsets

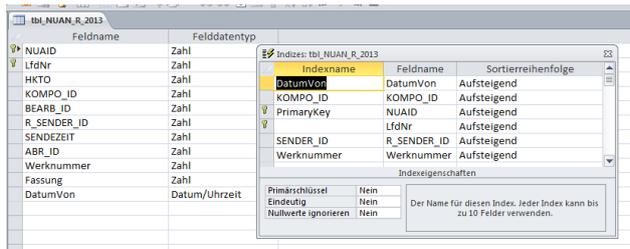
Datenbank komprimieren

### 3.1.2 Indices

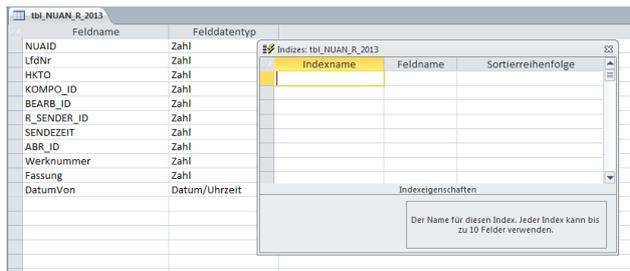
Tabelle mit 5.124.239 Sätzen



Indices:



ohne Indices:



Name	Änderungsdatum	Typ	Größe
NUAN_R_2013.accdb	25.07.2015 15:18	Microsoft Access Da...	390.112 KB
NUAN_R_2013_ohne_Indices.accdb	25.07.2015 15:24	Microsoft Access Da...	277.408 KB
NUAN_R_2013_ohne_Indices_1_LongDurchByte_ersetzt.accdb	25.07.2015 15:27	Microsoft Access Da...	263.240 KB

113 MB für Indices, ca. 30 % des Datenvolumens

Allerdings sind diese Indices hier unverzichtbar.

Indices werden oft „automatisch“ angelegt: Datenbankentwurf kontrollieren  
Indices sparsam verwenden, sind aber andererseits für Auswertung unverzichtbar.

Anmerkung: vor Massentransfers Indices abschalten, danach wieder einsetzen

### 3.1.3 Referenzielle Integrität

Screenshot mit Dateigrößen  
Platzbedarf wie Primary Key

## 3.2 Platzbedarf für Feldtypen

places, and Double allows 15. Use the Double setting when you need many decimal places or very large numbers.

Numeric Field Size Properties

Setting	Description	Decimal Precision	Storage Size
Byte	Stores numbers from 0 to 255 (no fractions).	None	1 byte
Integer	Stores numbers from -32,768 to 32,767 (no fractions).	None	2 bytes
Long Integer	(Default) Stores numbers from -2,147,483,648 to 2,147,483,647 (no fractions).	None	4 bytes
Decimal	Stores numbers from $-10^{28}$ -1 through $10^{28}$ -1	28	12 bytes
Single	Stores numbers from -3.402823E38 to -1.401298E-45 for negative values and from 1.401298E-45 to 3.402823E38 for positive values.	7	4 bytes
Double	Stores numbers from -1.79769313486231E308 to -4.94065645841247E-324 for negative values and from 1.79769313486231E308 to 4.94065645841247E-324 for positive values.	15	8 bytes
Replication ID	Globally unique identifier (GUID)	N/A	16 bytes

Caution: If you convert a large FieldSize setting to a smaller one in a field that already contains data, you might lose data. For example, if you change the FieldSize setting for a Text data type field from

DATETIME 8 Byte

Beispiel:

Tabelle mit 5.124.239 Sätzen

ABR\_ID LONG ersetzt durch Byte). Ersparnis je Satz 3 Byte:

NUAN_R_2013_ohne_Indices.accdb	25.07.2015 15:24	Microsoft Access Da...	277.408 KB
NUAN_R_2013_ohne_Indices_1_LongDurchByte_ersetzt.accdb	25.07.2015 15:27	Microsoft Access Da...	263.240 KB

Errechnet: 15.373 KB

Angezeigt: 14.168 KB

Feldtyp-Änderungen bringen weniger als man erwartet.

Leere numerische Felder belegen ihren Platz. Kein Unterschied in der Größe der DB nach Füllen mit Werten.

Leere TEXT-Felder: Nach Füllen ist die Datenbank größer.

## 4 Strategien, mit der 2-GB-Grenze umzugehen

### 4.1 1 Tabelle - 1 ACCDB

### 4.2 Striktes Datenmodell Star bzw. Snowflake

dadurch sparsamer Umgang mit Speicherplatz  
siehe unten Datenmodelle und Einlesen von Daten

### 4.3 Gleichartige Daten portionieren

Beispiel: Daten je Jahr in einer separaten Tabelle

Dies ist nur möglich, wenn Auswertungen hauptsächlich in Jahresgrenzen ausgeführt werden.

Beispiel für Abfrage aus tbl\_R\_2013 und tbl\_R\_2014 (SourceCode)

Prinzip:

Eine lokale Tabelle mit der gleichen Struktur wie die Faktentabellen wird verwendet.

In diese Tabelle werden die Ergebnisse geschrieben.

Die Ergebnistabelle wird mit den Dimensionen gejoint und ausgegeben.

```
gCurrentDB.Execute ("DELETE FROM tblTemp_R_Erg")

gstrSQL = _
"INSERT INTO    tblTemp_R_Erg " & _
"SELECT        * FROM tbl_NUAN_R_2013 WHERE " & strWHERE
gCurrentDB.Execute (gstrSQL)

gstrSQL = _
"INSERT        INTO tblTemp_R_Erg " & _
"SELECT        * FROM tbl_NUAN_R_2014 WHERE " & strWHERE
gCurrentDB.Execute (gstrSQL)
```

Dieser Weg mit Datenportionen ist suboptimal, weil man jedes Jahr Programme ändern muss.

Es können sich aber auch erstaunliche Performance-Gewinne ergeben, mehrmalige Zugriffe auf kleinere Datenvolumen; Abfrage mit Nullmenge als Ergebnis läuft sehr schnell.

## 5 Alternative Datenmodelle

### 5.1 Anfänge, Irrtümer

Ich habe begonnen, CSV-Daten so in Access einzulesen, wie sie geliefert werden.

Ich glaubte anfangs, denormalisierte Tabellen mit Auswertungen ohne jegliche JOINS würden schneller laufen. Das hat sich schnell als Irrweg herausgestellt: Das Datenvolumen wird zu groß und Access wird mit breiten Tabellen viel langsamer.

### 5.2 Beschäftigung mit der Data Warehouse Welt

Dimensionen und Kennzahlen in Faktentabelle

Dimensionstabellen als Stammdaten

Dimensionen			Kennzahlen					
Key1	Key2	...						

Literatur

Klassische Bücher zur Data Warehouse-Technik gibt es von Inmon (2005) und Kimball et al. (2008).

### 5.3 Erfahrungen mit SAS, Data Warehouse, Cubes

In einem Projekt bei der Telekom mit SAS konnte ich praktische Erfahrung mit der klassischen Data Warehouse-Technik und mit der Erstellung von vorbereiteten Auswertungen in Cubes sammeln.

Dabei wurde auch Access als Datensammler eingesetzt, um heterogene Daten wie Excel und CSV dem eigentlichen DWH zuzuführen.

Immer wieder kamen weitere Kennzahlen dazu, Tabellen wurden um Spalten erweitert, Programme mussten an vielen Stellen umgeschrieben werden. Dadurch kam ich auf das Modell der Kennzahlen-Typs als eigene Dimension. Das werde ich später näher erläutern.

Ich habe dann auch mitbekommen, dass das System SAS Financial Management ebenfalls dieses Modell der Kennzahlen-Dimension benutzt, dass es also nicht nur eine „Spinnerei“ von mir ist.

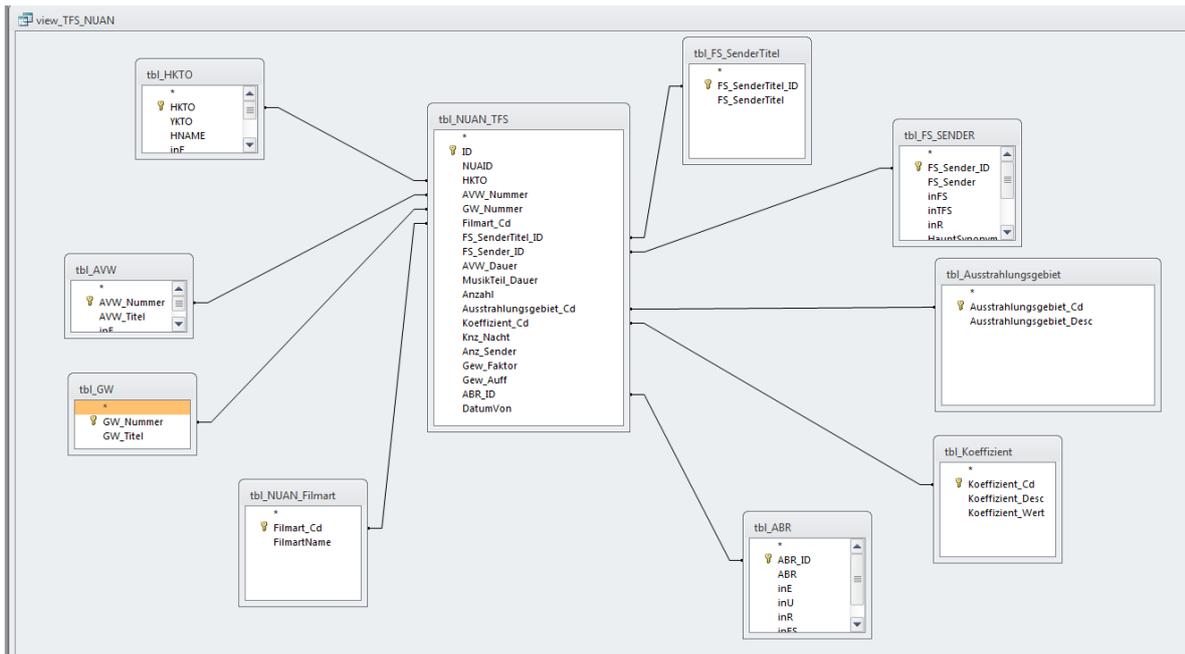
In der Data Warehouse-Welt sind 2 Datenmodelle gängig: Star und Snowflake:

### 5.4 Star-Schema

Beispiele

NUAN\_R

GESDAT



Name Star, weil es wie ein Stern aussieht

1 Faktentabelle mit Schlüsseln in die Dimensionstabellen

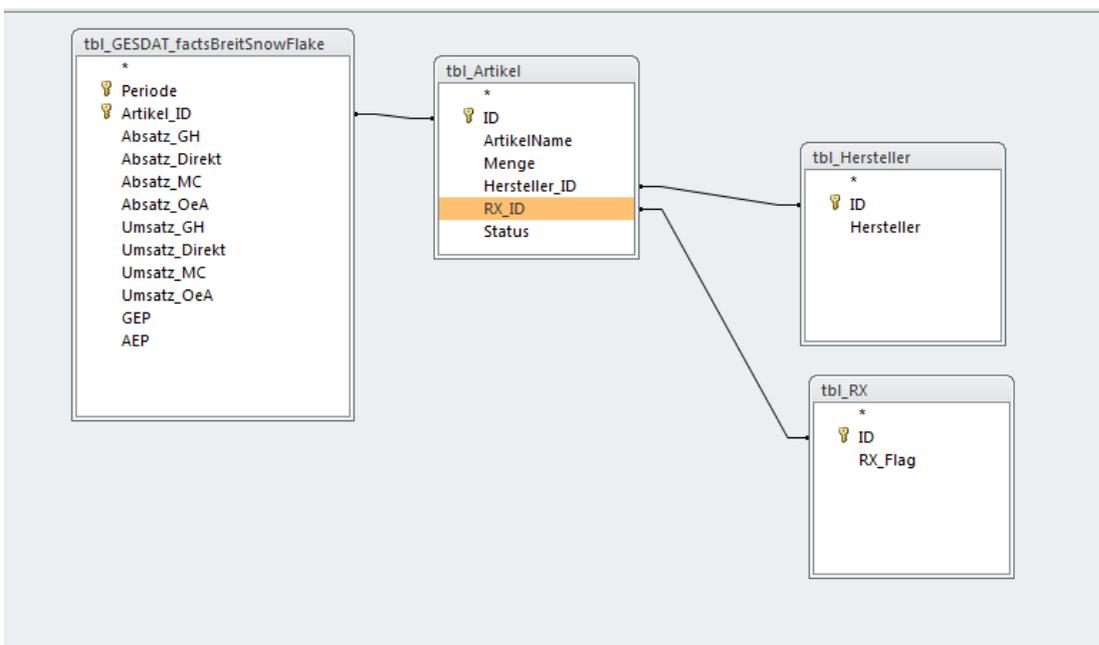
Vorteile:

leicht zu verstehen und zu handhaben  
weniger Joins, keine verketteten Joins

Nachteile:

größerer Speicherbedarf  
teilweise denormalisiert in den Dimensionen

## 5.5 Snowflake-Schema



Name Snowflake, weil es wie eine Schneeflocke aussieht

Vorteile:

geringerer Speicherbedarf, da der Teil der Dimensionen in der Faktentabelle kleiner wird, Dimensionen werden „nach hinten“ verlagert

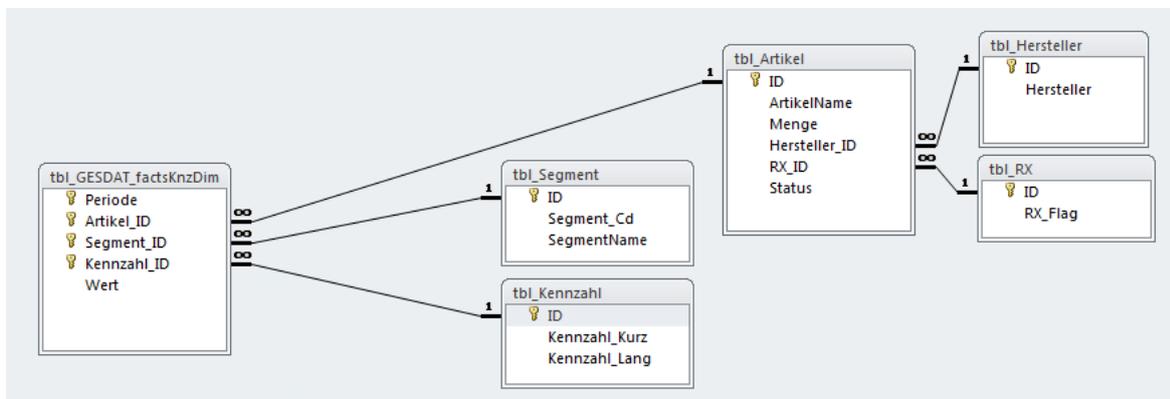
logischer, da vollständig normalisiert

weniger Joins, keine verketteten Joins

Nachteile: schwerer zu verstehen und zu handhaben

## 5.6 Kennzahlen-Typ als Dimension

Der Kennzahlen-Typ wird auch als Dimension behandelt. Es gibt nur 1 Messzahl. Für jede Messzahl wird ein eigener Satz geschrieben.



**Vorteile:**

- Variabilität in der Entwicklung: neue Kennzahlen können fast ohne Programmänderung angefügt werden
- Bestimmte Auswertungen in Pivot werden schlüssiger: leichte Auswahl von Kennzahlen
- In vielen Fällen weniger Code nötig

**Nachteile:**

- größerer Speicherbedarf:  
1,9 GB bei 20,6 Mio Sätzen und 10 Spalten mit Zahlenwerten.  
1 Kennzahl benötigt 465 MB. 10 Kennzahlen benötigen 4,65 GB.
- berechnete Kennzahlen sind schwieriger zu handhaben. Man muss mit Kreuztabellen-Abfragen arbeiten.

Periode	Artikel_ID	Segment_ID	Kennzahl_ID	Wert
200897	1	1	3	5,21
200897	2	1	3	7,4
200897	3	0	3	3,04
200897	4	0	3	3,17
200897	5	0	3	3,63
200897	6	0	3	6,28
200897	7	0	3	2,9
200897	8	0	3	1,36
200897	9	0	3	9,49

Wegen des größeren Speicherplatzbedarfs durch die Wiederholung von Dimensionen ist hier die Kombination mit einem Snowflake-Schema ratsam.

## 5.7 Möglichkeit, dieses Kennzahlen-Modell für OLTP-Systeme anzuwenden

Version 7 Weidmann EDV-Beratung Fact-Control

allgemeine Eingabe Stammdaten Auswertungen Tools Info

Sortierung:  Datum absteigend  ID absteigend  Prj/Datum absteigend  
 Datum aufsteigend  ID aufsteigend  Prj/Datum aufsteigend

Filtern Export nach Excel Filter entfernen  summieren neuer Satz Satz laden Satz löschen

ID	Projekt	AP	Jahr	Mon	Tag	StdVon	StdBis	Ort	Berater	DokumentNr	Kennzahl	Wert	Beschreibung
1017	Prj-29	Prj_AP-22	2015	7	13	Mo		Stierstr.	TW		Std ist MA	4,25	FormEdit
1018	Prj-34	Prj_AP-31	2015	7	13	Mo		Remote	TW		Std ist MA	1,00	Besprechung
1015	Prj-34	Prj_AP-34	2015	7	13	Mo		Remote	TW		Std ist MA	1,00	Besprechung
1014	Prj-34	Prj_AP-31	2015	7	13	Mo		Remote	TW		FahrStd	2,00	Berlin, zu DG, Smetanastr. Hin und zurück
1013	Prj-34	Prj_AP-35	2015	7	10	Fr		Remote	TW		FahrStd	0,50	Badensche Str.
1012	Prj-34	Prj_AP-35	2015	7	10	Fr		Remote	TW		Std ist MA	2,00	Besprechung mit Revision zu Dokumentation und Testplanung in Badenscher Str.
1011	Prj-15	Prj_AP-5	2015	7	10	Fr		Stierstr.	TW		Std ist MA	1,00	Eva AWK
1010	Prj-10	Prj_AP-30	2015	7	9	Do		UMP	TW		Std ist MA	4,50	Nutzungsdaten Rundfunk 2014 eingelesen
1007	Prj-15	Prj_AP-5	2015	7	8	Mi		Stierstr.	TW		Std ist MA	2,00	
1006	Prj-14	Prj_AP-4	2015	7	8	Mi		Stierstr.	TW		Std ist MA	5,00	
999	Prj-29	Prj_AP-22	2015	7	8	Mi		<kein>	--	60-C-151312-C	Std beauftragt	160,00	Juni - September 2015
998	Prj-29	Prj_AP-22	2015	7	8	Mi		<kein>	--	60-C-151312-C	Auftragswert	9.280,00	160 Std, Juni - September 2015
1084	Prj-34	Prj_AP-34	2015	7	7	Di		HomeOffice	DG		Std ist MA	2,00	
1005	Prj-32	Prj_AP-27	2015	7	7	Di		München	TW		Std ist MA	8,00	Besprechungen mit Anwendern, Betreuung der Projekte

Das Zuschalten von weiteren Kennzahlen ist möglich, ohne Programmänderungen vorzunehmen.

Gute Auswertungsmöglichkeiten mit Pivot, weil die Kennzahlen als Dimensionen verfügbar sind.

tbl\_Kennzahl

Kennzahl

Knz_ID	Kurz	Lang	Dim. Kurz	Dimension lang	Faktor	PrjCtrl Geld	PrjCtrl Zeit	Plan-Ist	Berech. Modus	kumuliert aus Knz_ID
	<keine Knz>	keine Kennzahl	-	-		<input type="checkbox"/>	<input type="checkbox"/>			
2	Std Plan	Stunden geplant	Std	Std		<input type="checkbox"/>	<input checked="" type="checkbox"/>	P		
3	Std ist MA	Stunden Ist Berater eingabe	Std	Std	-1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	I		
6	Re Ausgang	Ausgangsrechnung Projektarbeit	€	Euro	1	<input checked="" type="checkbox"/>	<input type="checkbox"/>			
7	Z Eingang	Zahlungseingang	€	Euro	1	<input type="checkbox"/>	<input type="checkbox"/>			
10	AT MA in Re	von MA in Rechnung gestellte AT	AT	AT	1	<input type="checkbox"/>	<input type="checkbox"/>			
12	AT beauftragt	beauftragte Arbeitstage	AT	AT	1	<input type="checkbox"/>	<input type="checkbox"/>			
13	AT abgerechnet	abgerechnete Projektstage	AT	AT	1	<input type="checkbox"/>	<input type="checkbox"/>			
16	Prj Spesen	Reisekosten	€	Euro	-1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	I		
19	AP geplant	geplante AP	AT	AT	1	<input type="checkbox"/>	<input type="checkbox"/>			
21	ATistMA	von MA eingereichte AT	AT	AT	1	<input type="checkbox"/>	<input type="checkbox"/>			
22	Std Reisezeiten	Reisezeiten Stunden	Std	Std	1	<input type="checkbox"/>	<input type="checkbox"/>	I		

ID	PrjAP_ID	PrjPh_ID	Datum	Jahr	Monat	Tag	StdVon	StdBis	Ort_ID	Berater_ID	Knz_ID	Doc_Cd	Wert	Beschreibung
42	7	1	01.04.2014	2014	4	1			1	2	3		1,00	ASP.NET
43	3	1	01.04.2014	2014	4	1			1	2	3		0,50	
44	9	1	01.04.2014	2014	4	1			1	2	3		3,50	FORCE-Berichte, Gli
45	4	1	02.04.2014	2014	4	2			1	2	3		8,00	Buchführung, Umba
46	4	1	03.04.2014	2014	4	3			1	2	3		0,50	Umbau Büro
47	3	1	03.04.2014	2014	4	3			1	2	3		1,00	Windscheidstr
48	10	1	03.04.2014	2014	4	3			1	2	3		2,00	Valovis Datentrenn
49	2	1	03.04.2014	2014	4	3			1	2	3		5,00	Datenmodell
50	3	1	04.04.2014	2014	4	4			1	2	3		1,00	Windscheidstr
51	10	1	04.04.2014	2014	4	4			1	2	3		1,00	Telco
52	2	1	04.04.2014	2014	4	4			1	2	3		3,00	mit Patricia zebra-S
53	6	1	07.04.2014	2014	4	7			1	2	3		1,00	Stunden März erfas
54	3	1	07.04.2014	2014	4	7			1	2	3		1,50	UMP
55	13	1	07.04.2014	2014	4	7			1	2	3		1,75	Wartung
56	19	1	07.04.2014	2014	4	7			1	2	3		8,00	GEMA-Nutzungsauf
57	8	1	08.04.2014	2014	4	8			1	2	3		4,00	Anpassungen an Ac
58	3	1	08.04.2014	2014	4	8			1	2	3		0,25	Badensche Str.
59	3	1	08.04.2014	2014	4	8			1	2	3		1,50	UMP
60	19	1	08.04.2014	2014	4	8			1	2	3		7,50	GEMA-Nutzungsauf
62	3	1	09.04.2014	2014	4	9			1	2	3		0,25	Badensche Str.
68	9	1	09.04.2014	2014	4	9			1	2	3		0,50	Modul war zerstört.
69	8	1	09.04.2014	2014	4	9			1	2	3		1,00	Reports angepasst;
70	3	1		2014	4	9			1	2	3		1,50	UMP
71	13	1		2014	4	9			1	2	3		7,50	Verarbeitung GEMA
72	3	1		2014	4	10			1	2	3		1,00	Windscheidstr
73	3	1		2014	4	10			1	2	3		1,50	UMP
74	2	1		2014	4	10			1	2	3		4,50	Konzept, Besprechu
77	3	1		2014	4	11			1	2	3		1,00	Windscheidstr.
78	10	1		2014	4	11			1	2	3		1,00	Telco
79	2	1		2014	4	11			1	2	3		4,00	Konzept, Datenstru
80	11	1		2014	4	14			1	2	3		8,00	
81	11	1		2014	4	15			1	2	3		8,00	
82	11	1		2014	4	16			1	2	3		8,00	
83	11	1		2014	4	17			1	2	3		8,00	
84	5	1		2014	4	22			1	2	3		6,00	
85	4	1		2014	4	22			1	2	3		2,00	

## 5.8 Resümee zur Datenstruktur

Durch die 2-GB-Grenze wird man in Access gezwungen, sich intensiv mit Strukturen und Speichervorgängen zu beschäftigen und so sauber als möglich zu arbeiten. Eine große Maschine und Datenbank steckt viele Unsauberkeiten weg.

## 6 Mit Access lokale Ressourcen nutzen

Die Performance von SQL-Server u.a. beruht zum größten Teil darauf, dass die Daten-intensiven Prozesse lokal erfolgen und nur die Ergebnisse übers Netz gehen.

Außerdem sind die Datenbank-Server normalerweise mit starker Hardware ausgestattet.

ACE ist stark in der Handhabung lokaler Daten.

Deshalb muss die Verarbeitung von großen Datenmengen möglichst weitgehend auf der lokalen Platte erfolgen.

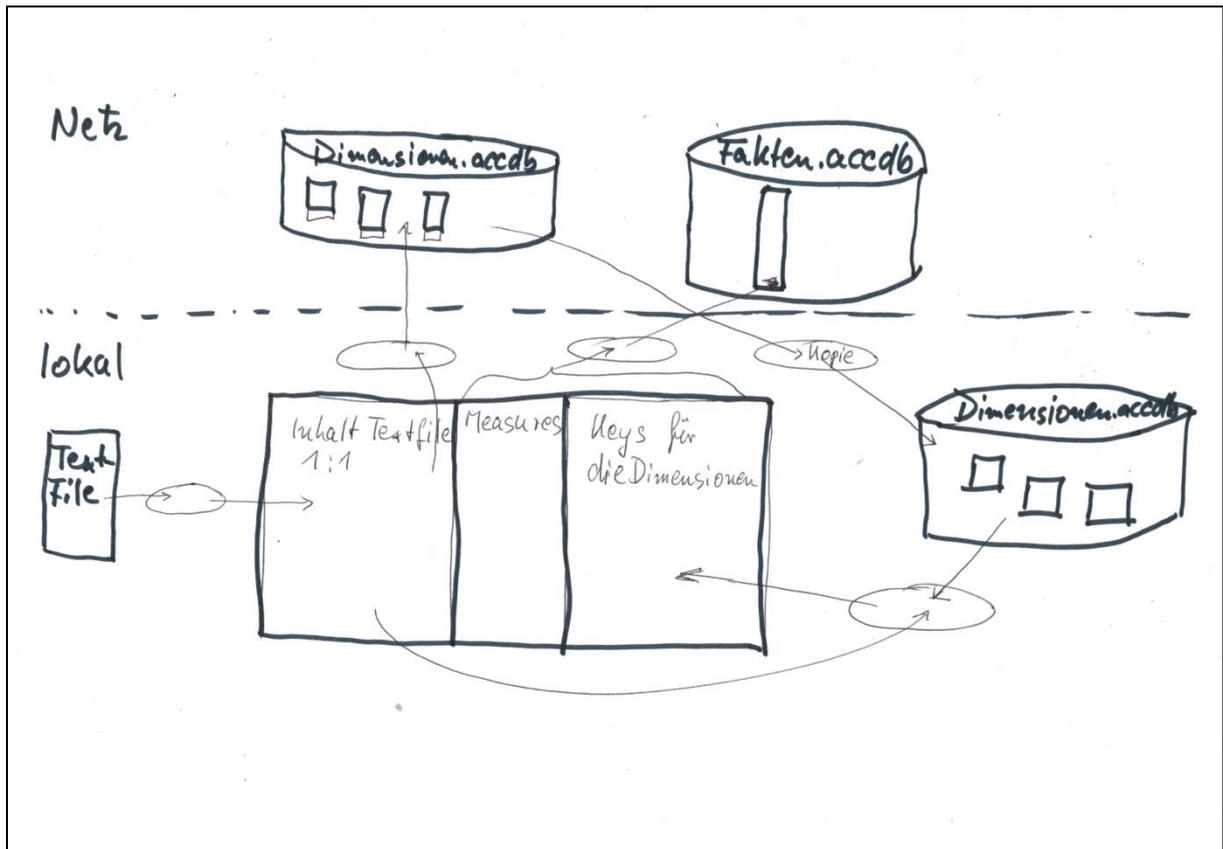
Die Strategie der Datenbank-Server kann man mit ACE weitestgehend nachbilden, indem man mit lokalen Tabellen arbeitet. Bei Schreibvorgängen bremsen z.B. keine Lock-Vorgänge.

Der Unterschied ist, dass man mit Access meist auf normal ausgestatteten Maschinen arbeitet, im Gegensatz zu starken Datenbank-Servern. Allerdings muss man die Arbeitsstation nicht mit anderen Benutzern teilen wie beim Server.

## 7 Typisches Vorgehen beim Einlesen

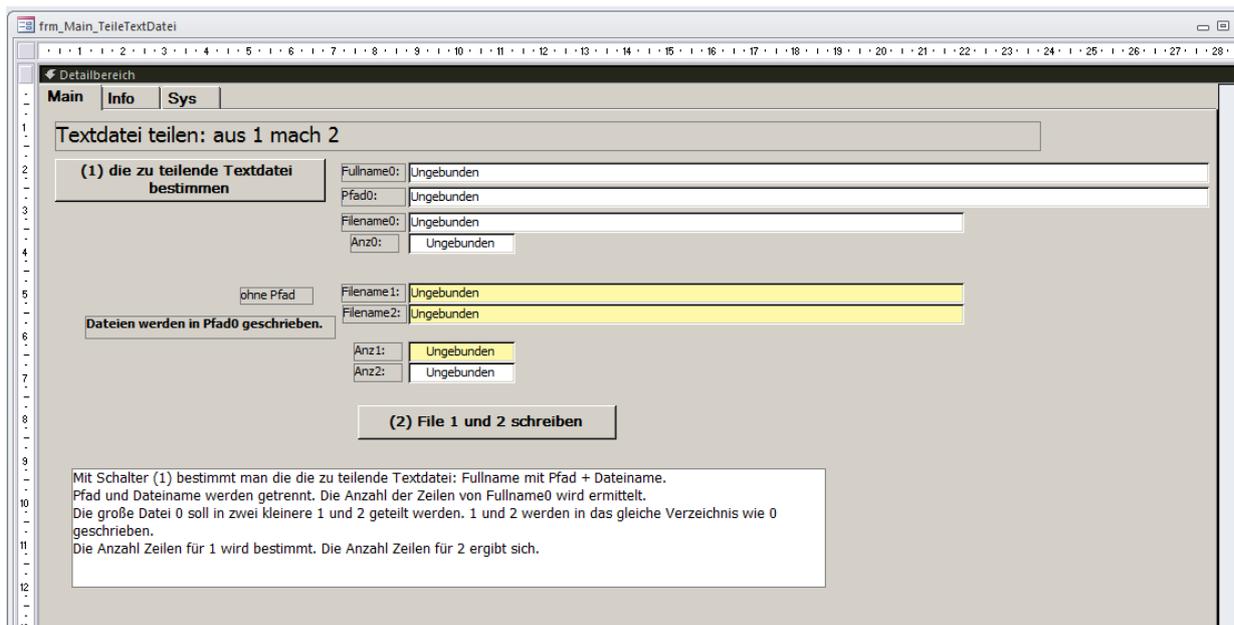
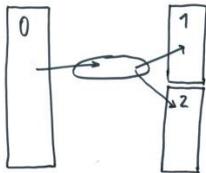
Nr	lokal	Netz
1	Erzeugen einer Datenbank	
2	Erstellen einer Einlesetabelle tblTemp_Work mit 3 Bereichen : <ul style="list-style-type: none"><li>- Inhalt der Textdatei 1:1</li><li>- Measures</li><li>- Keys für die Dimensionen</li></ul>	
3	Einlesen der Textdatei in die tblTemp_Work	
4		Eintrag neuer Dimensionen in die Dimensionstabellen
5	Kopie der Dimensionstabellen in eine lokale Datenbank	
6	Eintrag der Keys für die Dimensionen in tblTemp_Work	
7	Eintrag der Measures in tblTemp_Work	
8		Eintrag in die Fakten-Tabelle im Netz

Alle datenintensiven Prozesse finden lokal statt.



## 7.1 Tool zum Teilen großer Textdateien

Die lokale Arbeitsdatenbank kann die 2-GB-Grenze übersteigen. In solchen Fällen wird die einzulesende Textdatei geteilt.



```
Dim fso As Scripting.FileSystemObject
Dim txtStreamIn As Scripting.TextStream
Dim txtStreamOut As Scripting.TextStream
Dim strLine As String
Dim lngZeileNr As Long
```

```
On Error GoTo ErrorHandler
```

```
Me.mytxtAnz2 = Me.mytxtAnz0 - Me.mytxtAnz1
```

```
Set fso = CreateObject("Scripting.FileSystemObject")
Set txtStreamIn = fso.OpenTextFile(Filename:=Me.mytxtFullname0, IOMode:=ForReading)
Set txtStreamOut = fso.CreateTextFile(Filename:=Me.mytxtPfad0 & Me.mytxtFilename1), IOMode:=ForAppending)
```

```
strLine = ""
lngZeileNr = 0
```

```
Do While lngZeileNr < Me.mytxtAnz1
    lngZeileNr = lngZeileNr + 1
    strLine = txtStreamIn.ReadLine
    strLine = Trim(strLine)
    txtStreamOut.WriteLine strLine
Loop
txtStreamOut.Close
```

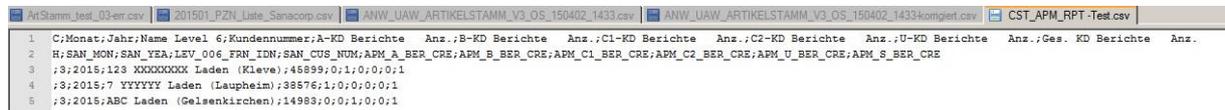
```
Set txtStreamOut = fso.CreateTextFile(Filename:=Me.mytxtPfad0 & Me.mytxtFilename2), IOMode:=ForAppending)
Do While Not txtStreamIn.AtEndOfStream
    lngZeileNr = lngZeileNr + 1
    strLine = txtStreamIn.ReadLine
    strLine = Trim(strLine)
    txtStreamOut.WriteLine strLine
Loop
```

## 8 Einlesen von Text-Daten

### 8.1 Beispiele für verschiedene Szenarien

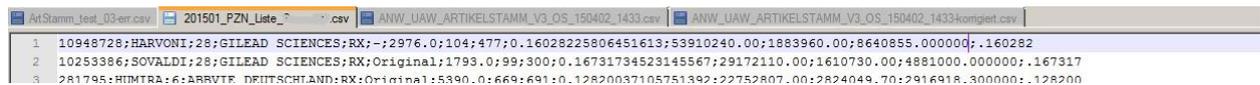
Die Kreativität der Programmierer, die Text-Dateien erzeugen, ist erstaunlich groß. Hier einige Beispiele:

- 2 Headerzeilen



```
1 C;Monat;Jahr;Name Level 6;Kundennummer;A-KD Berichte Anz.;B-KD Berichte Anz.;C1-KD Berichte Anz.;C2-KD Berichte Anz.;U-KD Berichte Anz.;Ges. KD Berichte Anz.
2 H;SAM_MON;SAM_YEA;LEV_006;FRN_IDN;SAM_CUS_NUM;APM_A_BER_CRE;APM_B_BER_CRE;APM_C1_BER_CRE;APM_C2_BER_CRE;APM_U_BER_CRE;APM_S_BER_CRE
3 ;8;2015;128 XXXXXXXX Laden (Kleve);45899;0;1;0;0;0;1
4 ;8;2015;7 YYYYYY Laden (Laupheim);98876;1;0;0;0;1
5 ;8;2015;ABC Laden (Gelsenkirchen);14989;0;0;1;0;0;1
```

- keine Headerzeile



```
1 10948728;HARVONI;28;GILEAD SCIENCES;RX;-;2976.0;104;477;0.16028225806451613;53910240.00;1883960.00;8640855.000000;.160282
2 10253386;SOVALDI;28;GILEAD SCIENCES;RX;Original;1793.0;99;300;0.16731734523145567;29172110.00;1610730.00;4881000.000000;.167317
3 281795;HIMTR1;6;ARRVTE DEUTSCHLAND;RX;Original;5390.0;669;691;0.12820037105751392;22752807.00;2824049.70;2916918.300000;.128200
```

- Fehler in den Daten z.B. durch eine Zeile als Seitenüberschrift

### 8.2 Anforderungen für Einlesen von Daten

- zyklisches Einlesen und Bereitstellen (täglich, monatlich ...)
- Änderungen in der Datenstruktur erkennen
- Fehler in den Daten erkennen

Access bietet das Instrumentarium der Import/Export-Spezifikationen.

Ist es besser, mit Importspezifikationen oder individuellen Prozeduren auf Basis von Textstreams zu arbeiten?

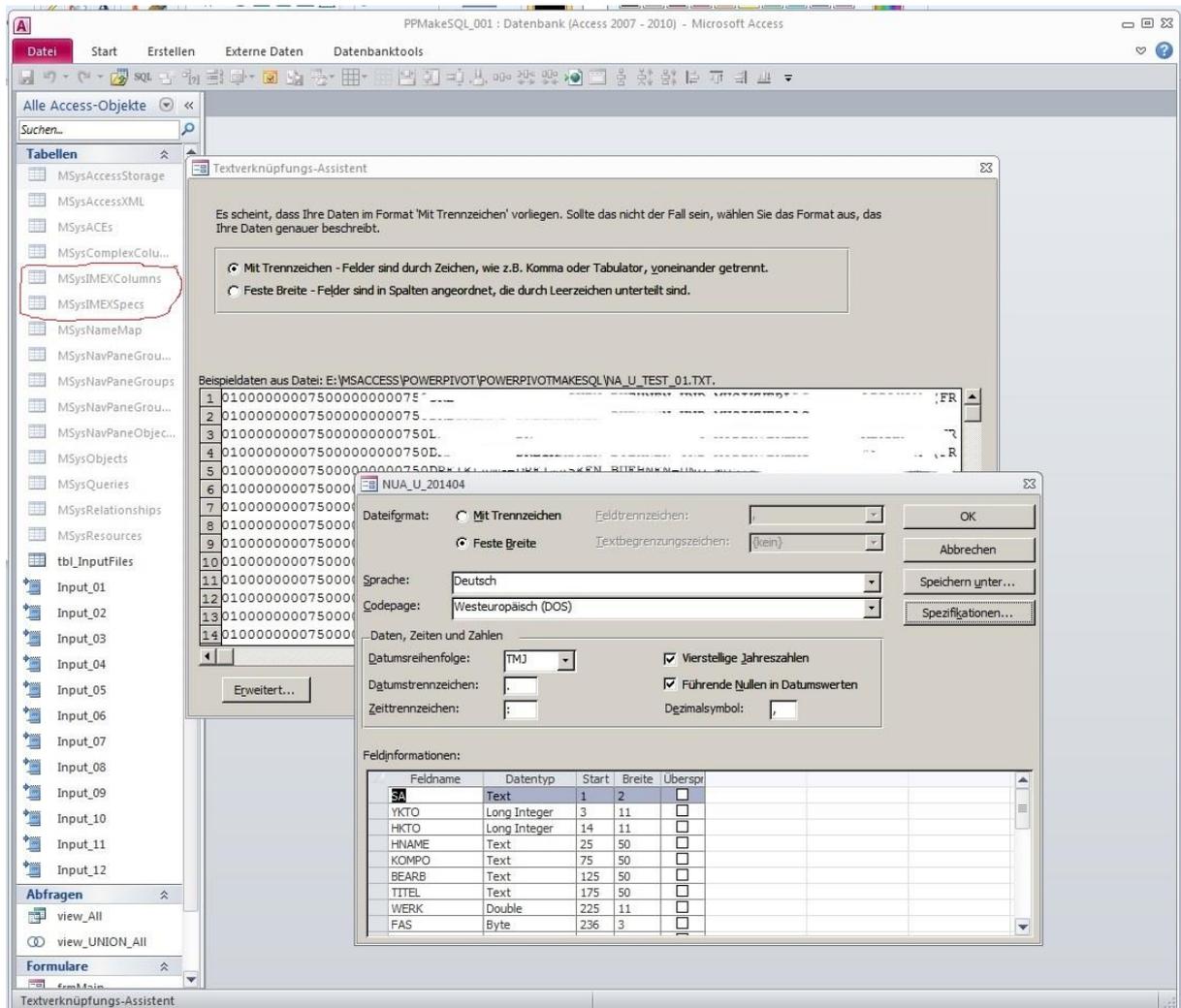
### 8.3 Importspezifikationen

Für das Einlesen von Daten bietet Access das Instrument der gespeicherten Spezifikationen an. Sind diese Spezifikationen geeignet für den produktiven Dauerbetrieb?

Hier die Benutzeroberfläche zum interaktiven Erstellen von Importspezifikationen.

Die Daten werden in den beiden Tabellen MSysIMEXSpecs und MSysIMEXColumns gespeichert.

In diese Tabellen kann man auch direkt schreiben oder aus ihnen per VBA lesen.



### 8.3.1 Vorteile

geringerer Programmier-Aufwand

Daten können eingelinkt und wie Access-Tabellen behandelt werden.

### 8.3.2 Nachteile

keine Fehlerbehandlung bei Link auf Text-Dateien

Die Importspezifikationen sind in ihrer Funktionsweise schwer zu durchschauen und enthalten viele Unklarheiten.

### 8.3.3 Beobachtungen mit Importspezifikationen

Import von CSV, Excel usw.

Die Tatsache, ob Feldnamen in den Import-Daten vorhanden sind, wird nicht gespeichert. Dies wird erst beim eigentlichen Import per VBA mitgegeben:

```
DoCmd.TransferText acLinkDelim, str_ImportSpec, str_ImportTable, Me.mytxt_GESDAT_File, True ' (bolHasFieldNames)
```

Wenn keine Feldnamen vorhanden sind (False), dann vergibt Access Standardnamen (Feld01, Feld02), oder man hat in der Spezifikation Namen angegeben.

Wenn Feldnamen vorhanden sind, dann setzt Access diese beim Speichern der Spezifikation in die Tabelle MsysIMEXColumns ein. Oft sind dies "krude" Usernamen mit Klammern und anderem

Gestrüpp. Diese Feldnamen kann man dann wieder durch VBA-nahe, technische Namen ersetzen. Diese Namen erscheinen dann auch als Feldnamen der eingebundenen Daten.

### 8.3.4 Fehlersituationen und Import-Spezifikationen

Falsche Datentypen:

```

1 H;SAN_MON;SAN_YEA;LEV_006_FRN_IDN;SAN_CUS_NUM;APM_A_BER_CRE;APM_B_BER_CRE;APM_C1_BER_CRE;APM_C2_BER_CRE;APM_U_BER_CRE;APM_S_BER_CRE
2 ;3;2015;123 XXXXXXXX Laden (Kleve);45899;0;1;0;0;0;1
3 ;3;2015;7 YYYYYY Laden (Laupheim);38576;1;0;0;0;0;1
4 ;3;2015;ABC Laden (Gelsenkirchen);14983;0;0;1;0;0;1
5 ;3;2015;ABF-Apo. Breitscheidstr. (Fürth);32773;1;0;0;0;0;1
6 ;3;2015;ABF-Apo. Königswarterstr. (Fürth);32768;0;1;0;0;0;1

```

H	SAN_MON	SAN_YEA	LEV_006_FR	SAN_CUS_N	APM_A_BER	APM_B_BER	APM_C1_BE	APM_C2_BE	APM_U_BER	APM_S_BER
	3	2015	123 XXXXXXXX	45899	0	1	0	0	0	1
	3	2015	7 YYYYYY Laden	38576	1	0	0	0	0	1
	#Zahl!	2015	ABC Laden (Ge	14983	0	0	1	0	0	1
	3	2015	ABF-Apo. Breit	32773	1	0	0	0	0	1
	3	2015	ABF-Apo. Köni	32768	0	1	0	0	0	1
	3	2015	APOLAND Lad	39744	1	0	0	0	0	1
	3	2015	AVIE Forst Lad	13535	1	0	0	0	0	1
	3	2015	Abtei Laden (N	800035	1	0	0	0	0	1
	3	2015	Abtei Laden (D	14627	0	1	0	0	0	1

Fehler in Zeilenverlauf:

z.B. in Zeile 112.014 kommt eine Zwischenzeile mit Seitenkopf.

Fazit:

Fehler werden von Importspezifikationen – vor allem beim Einbinden – nicht vernünftig abgefangen.

## 8.4 Alternative: Einlesen mit TextStream

Demonstration einer typischen Programmstruktur

```

Dim fso As Scripting.FileSystemObject
Dim txtStream As Scripting.TextStream
Dim strLine As String
Dim rstTempU As DAO.Recordset
Dim lngZeileNr As Long
Dim arrTemp() As String

On Error GoTo ErrorHandler

Set rstTempU = gCurrentDB.OpenRecordset("tempU")

Set fso = CreateObject("Scripting.FileSystemObject")
Set txtStream = fso.OpenTextFile(Filename:=pstrFullname,
IOMode:=ForReading)

strLine = ""
lngZeileNr = 0

Do While Not txtStream.AtEndOfStream

    strLine = txtStream.ReadLine
    strLine = Trim(strLine)
    arrTemp = Split(strLine, "|")

```

```
lngZeileNr = lngZeileNr + 1

rstTempU.AddNew

rstTempU!LfdNr = lngZeileNr
rstTempU!YKTO = Nz(arrTemp(1), "")
rstTempU!HKTO = Nz(arrTemp(2), "")
usw.
```

Loop

## 8.5 Fazit zu Import-Spezifikationen

Wir arbeiten z.Z. an einer Kombination von gespeicherten Importspezifikationen und vorgeschaltetem Prüfen der Daten mithilfe von Textstreams. Die Feldnamen können in der verwendeten ImpSpec gespeichert sein. Weitere Details wie das Vorhandensein von Spaltenüberschriften müssen separat vorgehalten werden. Das Prüfen auf den Typ der Feldinhalte (Zahlen, Datum) muss durch VBA gemacht werden.

## 9 Verwendung von großen Datenmengen

- Über vordefinierte Formulare stellen Anwender Anfragen, die per VBA in SQLs umgewandelt und ausgeführt werden. Die Anwender können in den Ergebnissen weiter sortieren und filtern.
- Für Excel-Pivot und PowerPivot stehen Abfragen bereit.

### 9.1 Beispiel einer Abfragemaske

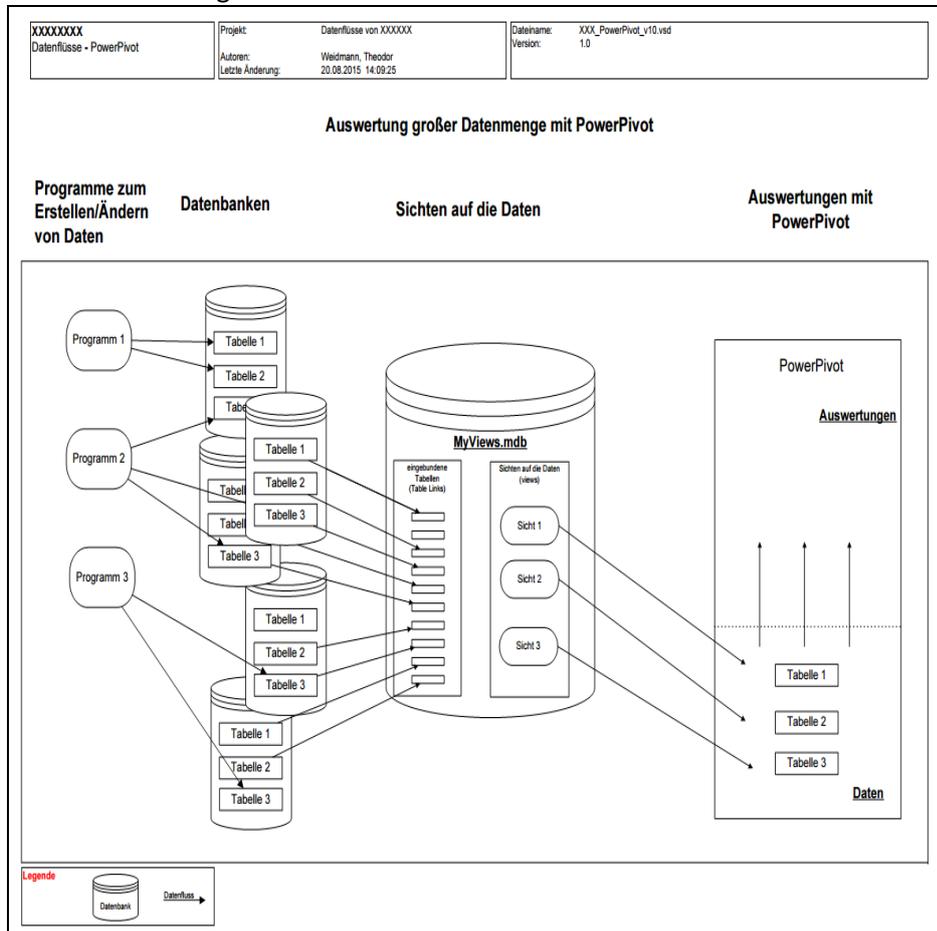


### 9.2 Access-Modul myViews.accdb als Mittler zwischen Daten und Auswertungen

Vorteile:

Keine produktive ACCDB wird blockiert beim Verwenden der Abfragen

Zusammenfassung an einer zentralen Stelle



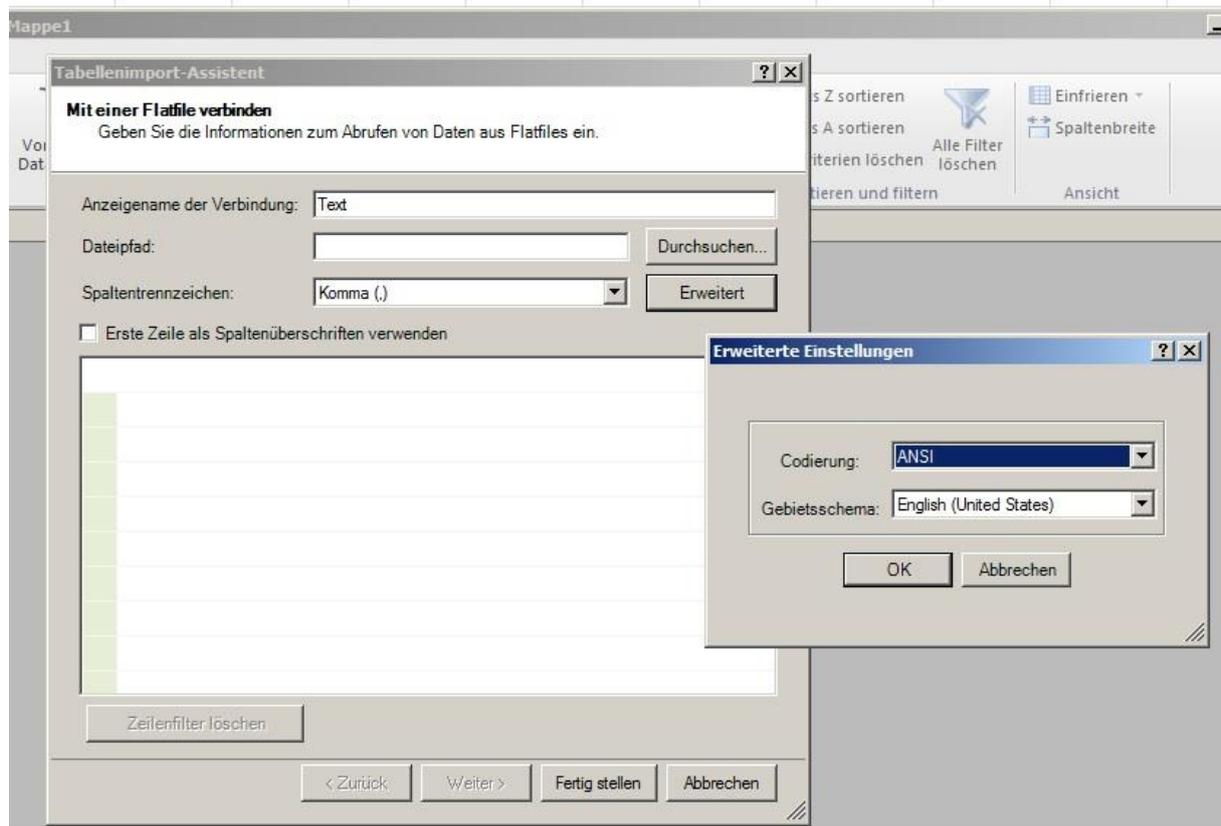
## 10 Access als Datenlieferant für PowerPivot

Access 2010 und Excel 2010

### 10.1 Beschränkungen von PowerPivot beim Einlesen von Daten

#### 10.1.1 Wenige Konfigurationsmöglichkeiten

Die Möglichkeiten, beim Einlesen in PowerPivot zu konfigurieren, sind dürftig, vor allem bei Formaten von Zahlen und Datum



#### 10.1.2 Daten anfügen unmöglich/umständlich

#### 10.1.3 Beschränkungen bei Access-Objekten

- Keine eingelinkten Tabellen
- Keine UNION-Views
- Keine Views mit Funktionen

WorkAround:

Views auf diese Daten-Objekte

## 10.2 Ausgleich dieser PowerPivot-Schwächen mit Access

Diese Schwächen von PowerPivot kann Access als „Mittler“ zu den Textdaten ausgleichen.

Dies beruht auf folgenden Funktionen von Access:

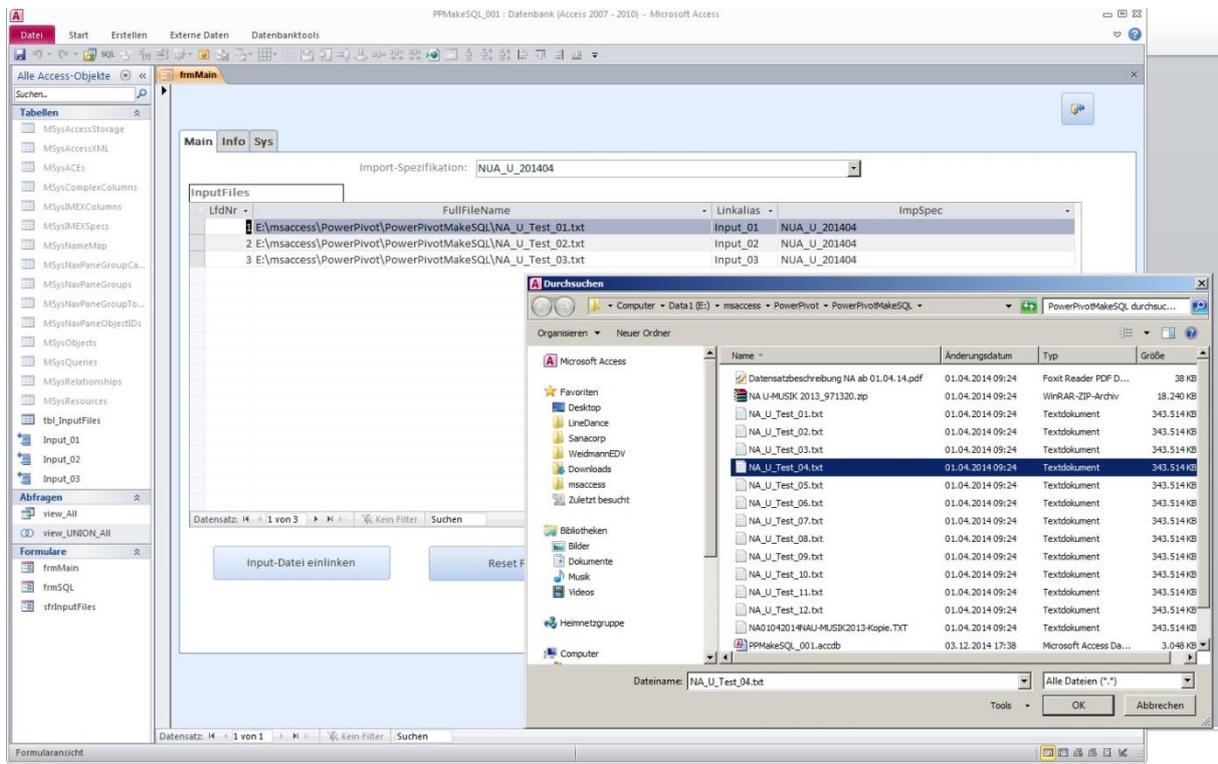
- Importspezifikationen zur feinteiligen Konfiguration des Text-Stroms
- Einlinken von Text-Dateien in Access wie Tabellen
- Verwenden von eingelinkten Tabellen in UNION-Abfragen

#### 10.2.1 Beispiel: 12 Textfiles (Monate) in PowerPivot auswerten

Die 12 Textfiles haben alle die gleiche Struktur.



### 10.2.1.3 Programm zum Einlinken der Textfiles



```
Public Function myfctEinlinken(ByVal pstrFullname As String, _
                             ByVal pstrSpecName As String, _
                             ByVal plngSpecType As Long, _
                             ByVal plngStartRow As Long) As Long
' *****
' Ausdruck .TransferText(TransferType, SpecificationName, TableName, FileName, HasFieldNames,
HTMLTableName, CodePage)
.....

Select Case plngSpecType
Case 1:
DoCmd.TransferText acLinkDelim, pstrSpecName, strLinkAlias, strFullname, bolHasFieldNames
Case 2:
DoCmd.TransferText acLinkFixed, pstrSpecName, strLinkAlias, strFullname, bolHasFieldNames
Case Else:
MsgBox "Nicht vorgesehener SpecType: " & plngSpecType, vbCritical, "Dateien einlinken"
myfctEinlinken = 0
End Select

If myfctEinlinken > 0 Then
rstLinks.AddNew
rstLinks!LfdNr = lngNewLfdNr
rstLinks!FullFileName = pstrFullname
rstLinks!LinkAlias = strLinkAlias
rstLinks!ImpSpec = pstrSpecName
rstLinks.Update
End If

Exit_Function:
On Error Resume Next
Exit Function

ErrorHandler:
.....
End Function
```

### 10.2.1.4 Views für das Einlesen

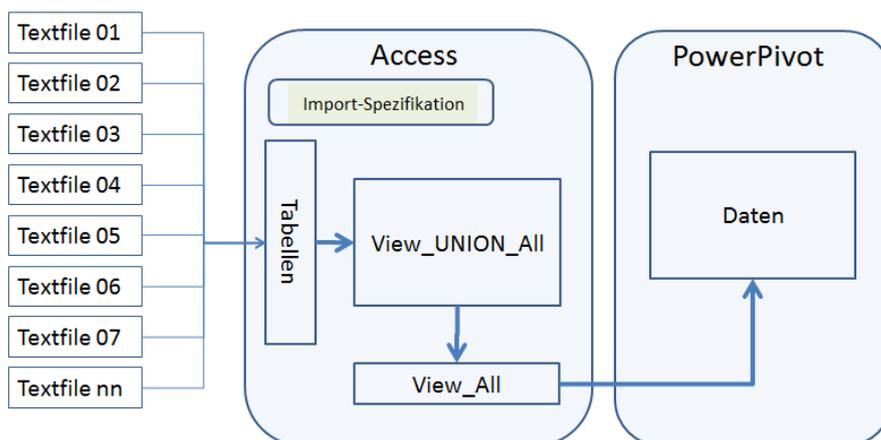
mit UNION ALL die eingebundenen Textfiles zusammenfügen

Aus PowerPivot diese UNION-Daten einlesen  
(PowerPivot kann keine UNION-Abfragen. Deshalb nochmals in Access eine weitere Abfrage vorschalten.)  
oder in PowerPivot die UNION-Abfrage von Hand schreiben

```
SELECT * FROM Input_01
UNION ALL
SELECT * FROM Input_02
UNION ALL
SELECT * FROM Input_03
UNION ALL
SELECT * FROM Input_04
UNION ALL
SELECT * FROM Input_05
UNION ALL
SELECT * FROM Input_06
UNION ALL
SELECT * FROM Input_07
UNION ALL
SELECT * FROM Input_08
UNION ALL
SELECT * FROM Input_09
UNION ALL
SELECT * FROM Input_10;
```

Die Daten werden nach und nach eingelesen. Die 2-GB-Grenze von Access greift hier nicht!

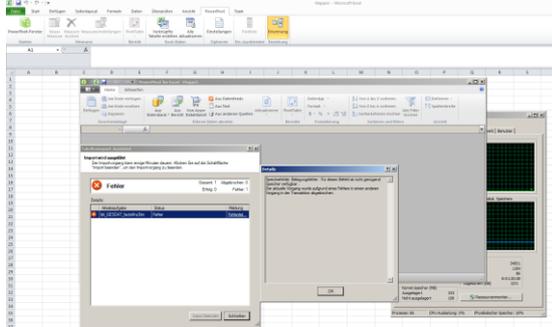
### 10.2.1.5 Systemskizze



# 11 Access versus PowerPivot zum Auswerten von Daten

## 11.1 PowerPivot hat auch seine Grenzen

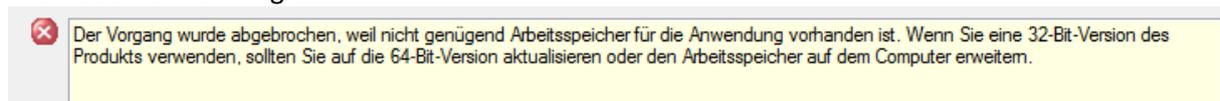
Versuch, die 80 Mio Sätze in PowerPivot einzulesen



Bei 40 Mio war Schluss.

Das Problem ist, dass PowerPivot alles erst mal in den Speicher einliest und dann erst auf Platte speichert.

Andere Fehlermeldung:



(Das Verhalten unter 64-Bit Office wurde nicht getestet.)

Erfahrung auf „normalen“ Arbeitsstationen (4 GB RAM): Bei ca. 10 Mio Sätzen wird die Luft für PowerPivot dünn.

FileID	PaketID	ID	JAHR	Sparte	AbrechNr
3066	468	615388	2009	11	1328
3066	468	615453	2009	11	1328
3066	468	615624	2009	11	1328
3066	468	615773	2009	11	1328
3066	468	616011	2009	11	1328
3066	468	616087	2009	11	1328
3066	468	616254	2009	11	1328
3066	468	617025	2009	11	1328
3066	468	617320	2009	11	1328
3066	468	617329	2009	11	1328
3066	468	617331	2009	11	1328
3066	468	617332	2009	11	1328
3066	468	617333	2009	11	1328
3066	468	617334	2009	11	1328
3066	468	617335	2009	11	1328
3066	468	617655	2009	11	1328
3066	468	618027	2009	11	1328
3066	468	618493	2009	11	1328
3066	468	618521	2009	11	1328
3066	468	618522	2009	11	1328
3066	468	618675	2009	11	1328
3066	468	618676	2009	11	1328
3066	468	618677	2009	11	1328
3066	468	618866	2009	11	1328
3066	468	619010	2009	11	1328
3066	468	619046	2009	11	1328
3066	468	619196	2009	11	1328
3066	468	619381	2009	11	1328

## 11.2 Vorteile von PowerPivot

- enorme Komprimierung der Daten
  - Geschwindigkeit bei Auswahllisten
- PowerPivot bildet intern die Dimensionstabellen.  
Die DropDown-Listen kommen sofort:



- CalculatedColumns
- kann Beziehungen zwischen eingelesenen Tabellen aufbauen
- Geschwindigkeit im Pivot-Prozess schneller als bei normalen Pivots

## 11.3 Nachteile von PowerPivot

Daten müssen bei jeder Änderung neu eingelesen werden.

## 12 Fazit

Ich hoffe, ich konnte einen Einblick geben über die Möglichkeiten und Grenzen der ACE mit größeren Datenmengen.

Als Access-Entwickler können wir durchaus selbstbewusst über die Möglichkeiten unseres Tools sein.