excel yourself

Autoren: Ralf Sowa, Christian Hapke

Excel - Thema: Datenbank-Funktionen

DBANZAHL - DBANZAHL2 - DBAUSZUG - DBMAX - DBMIN - DBMITTELWERT - DBPRODUKT - DBSUMME

Einleitung zu Datenbank-Funktionen

Ein Leben im Abseits

Wofür gibt es die Datenbank-Funktionen? – Die Frage drängt sich geradezu auf, denn: Es gibt kaum eine Aufgabenstellung, die sich nicht auch mit anderen Funktionen lösen ließe. Und in den meisten Fällen sind diese anderen Lösungen durchaus vorzuziehen.

Mit wenig *mathematisch anmutendem Erscheinungsbild* wirken Datenbank-Funktionen auf den geübten Excel-Anwender (vor allem auf Profis) wenig *attraktiv*; geradezu *befremdlich* ihre Eigenarten:

- Die Auswertung der Datenbank ist nur für Spalten möglich. → Die auszuwertende Datenbank muss vertikal verlaufen.
- Jede Spalte muss eine eindeutige Überschrift tragen – sie wird für die auszuwertende Spalte sowie für die Definition der Selektionskriterien genutzt.
- Die Selektionskriterien werden in Hilfszellen ausgelagert und sollten sorgfältig (in vorgegebener Weise) erfasst werden.

  (Natürlich wissen wir, dass es in Datenbanken nun einmal so ist – man denke an Access. Aber von Excel *hätten wir das nicht erwartet*.)

Alles Gründe dafür, dass DB-Funktionen ein *Leben* im Abseits führen. Nicht immer zu Recht!

Verwandtschaftsverhältnisse

Zu den *größten Wettbewerbern* der Datenbank-Funktionen zählen SUMMENPRODUKT und individuell erstellte Matrixformeln sowie auch die PIVOTTABELLE. – Vermutlich haben wir mit dieser Aufzählung der *Wettbewerber* auch die Eingangsfrage *(„Wofür gibt es die Datenbank-Funktionen?“)* beantwortet: Komplexe SUMMENPRODUKT-, individuelle Matrixformeln und die PIVOTTABELLE sind nicht *Jedermanns Sache...*

DBAUSZUG liefert dem SVERWEIS bzw. INDEX mit VERGLEICH entsprechende Ergebnisse. DBSUMME, DBANZAHL und DBANZAHL2 erfüllen Aufgaben, die wir sonst den Funktionen ZÄHLENWENN, SUMMEWENN und SUMMENPRODUKT anvertrauen. DBMIN entspricht etwa einer individuellen Matrixfunktion wie {=MIN(WENN(…))}…

Die Erfassung der Selektionskriterien für Datenbank-Funktionen erinnern an den SPEZIALFILTER.

Existiert je Datensatz ein eindeutiges Kriterium (ggf. in einer Hilfsspalte), kann DBAUSZUG Datensätze anzeigen, wie wir es von AUTOFILTER, SPEZIALFILTER und PIVOTTABELLE kennen. Überhaupt können die Datenbank-Funktionen fast alles, was auch die PIVOTTABELLE kann – bei weitem nicht so *komfortabel* (geübten Umgang mit Pivottabellen vorausgesetzt), dafür ohne manuelles „Aktualisieren“ bzw. ohne für das automatische Aktualisieren einen VBA-Code zu nutzen.

Vorteile der Datenbank-Funktionen

Ecxel-Funktionen, die mit ganzen Spalten arbeiten können (z. B. SUMMEWENN, ANZAHL), verfügen scheinbar über eine gewisse *Intelligenz,* jene Bereiche zu erkennen, die genutzt werden. – Sind in einer Spalte A nur 10 Zellen genutzt, werden SUMME(A:A) oder SUMMEWENN(A:A;“>0“) das Ergebnis schneller liefern als eine Matrixformel wie SUMMENPRODUKT((A1:A65536>=0)*1). Letztere hätte hier – trotzdem nur 10 Zellen genutzt sind – 130.000 Berechnungen anzustellen!

Auch die Datenbank-Funktionen können mit **ganzen Spalten** *umgehen*. Und auch deshalb bieten auch sie diese **gute Performance**. Eine geeignete Datenbank vorausgesetzt können Sie ohne Sorgen um die Performance ganze Spalten, ja sogar **ein ganzes Tabellenblatt als Datenbank** definieren. Damit empfehlen sich die Datenbank-Funktionen insbesondere für die Anwendung in **großen Datenbanken.**

> Der Kritiker wird einwenden, dass insbesondere mit der Funktion BEREICH.VERSCHIEBEN die Möglichkeit besteht, den zu berechnenden Bereich für Funktionen wie SUMMENPRODUKT und individuelle Matrixformeln auf das nötige Maß zu beschränken. Dem wollen wir gern zustimmen. Ein *Plus* für Datenbank-Funktionen bleibt es dennoch – für sie sind derartige Maßnahmen gar nicht nötig.
>
> Zur Performance:
>
> Schon bei nur einer Bedingung lieferte DBSUMME in unseren Tests das Ergebnis etwa **3,6-fach schneller** als SUMMENPRODUKT. Und es gilt: Je mehr Bedingungen, desto größer wird der Geschwindigkeitsvorteil. Bei 30 Bedingungen kann eine **bis zu 100-fach bessere Performance** realisierbar sein. – Natürlich gibt es auch noch VBA, aber was die Excel-Bordmittel angeht, dürften die Datenbank-Funktionen die schnellsten sein.

In (zugegebenen: seltenen) Einzelfällen, stellen die Datenbank-Funktionen durchaus den besten Lösungsweg dar.

Die Fähigkeit der DB-Funktionen mit ganzen Spalten arbeiten zu können, macht die Definition des Bereichs, in dem sich die Datenbank befindet, leicht überschaubar. Befindet sich die Datenbank in einem separaten Tabellenblatt, kann ohne Furcht vor Performance-Nachteilen – wie bereits geschrieben – das gesamte Tabellenblatt als Datenbank definiert werden. Jede Spalte der Datenbank wird über den Begriff in der ersten Zeile des Bereichs genutzt. – Nötige Nachsicht für die Eigenwilligkeiten der Datenbank-Funktionen vorausgesetzt, kann das durchaus als **anwenderfreundlich** gelten.

Weil die Datenbank für die Anwendung der DB-Funktionen stets mit einer Überschrift beginnen muss, sind die Auswertungen **unempfindlich gegen Zeilenlöschungen und Zeilenergänzungen**. Werden ganze Spalten (oder das ganze Tabellenblatt) als Datenbank definiert, kann ohne Eingriff in die Formeln eine neue Zeile 2 *eingeschoben* oder jede beliebige Zeile gelöscht werden. Auch eine **vollständige Aktualisierung der Datenbank** (neu einlesen / überschreiben) ist problemlos – lediglich die Spalten-Überschriften müssen erhalten bleiben bzw. neu mit eingelesen werden. In all diesen Fällen bleiben die Formeln so *simpel* wie sonst auch.

Wer sich auf die Eigenwilligkeiten der Datenbank-Funktionen einlässt, kann **selbst komplexe Datenbank-Auswertungen auch mit mittlerem Excel- und Mathematik-Verständnis beherrschen**. Die Alternative zu den Datenbank-Funktionen stellen recht komplexe Matrixformeln, oft mit dynamisierten Bereichsangaben (zwecks Performanceerhalt), dar – für den *nur gelegentlichen Excel-Anwender* mit mittleren Kenntnissen kaum selbst zu entwickeln oder anzupassen. Das macht die Datenbank-Funktionen vor allem für diese gelegentlichen Excel-Anwender besonders interessant.

Auf den ersten Blick erscheinen die Datenbank-Funktionen *übersichtlicher*, weil die Bedingungen sorgfältig in einen Zellenbereich ausgelagert sind. Als Vorteil kann dies nicht *durchgehen*, weil andere Funktionen derart ausgelagerte Bedingungen ebenso nutzen *könnten*. Nur – diese anderen *brauchen es nicht*.

Nachteile der Datenbank-Funktionen

Ein häufiges Problem, geradezu ein Ärgernis sind die **redundanten Überschriften** für die Kriterienangaben. Sie sind nicht *loszuwerden*. – In vielen Fällen wird dies Grund sein, die DB-Funktionen nicht zu verwenden.

Je nach Anwendung kann die Problematik allerdings gemildert werden: durch schlichtes Ausblenden der Bedingungen oder Auslagern in ein anderes Arbeitsblatt. Zuweilen kann eine Fülle von Bedingungen – mit ein paar Tricks – auch deutlich reduziert werden.

Anwendung der Datenbank-Funktionen

Die Funktionsweise der Datenbank-Funktionen erläutern wir am Beispiel von DBSUMME. Die übrigen Datenbank-Funktionen arbeiten nach den gleichen Grundprinzipien; auf sie gehen wir unten ebenfalls ein.

Definition der DB-Funktionen

Die Datenbank-Funktionen erwarten drei Eingaben:

(1.) Datenbereich – das sind die Felder, in denen sich die Datenbank befindet.

DBSUMME(**A:C**;“Umsatz“;E3:E4)

(2.) Die auszuwertende Spalte (als Bezug auf eine Zelle, die den Text der Überschrift enthält oder der in Anführungszeichen gesetzte Text der Überschrift)

DBSUMME(A:C;**“Umsatz“**;E3:E4)

alternativ:

DBSUMME(A:C;**C1**;E3:E4)
DBSUMME(A:C;**F1**;E3:E4)
DBSUMME(A:C;**2**;E3:E4)

In der letzten Alternative steht die 2 für die *zweite* Spaltenüberschrift des Bereichs.

(3.) Bereich, der die Selektionskriterien enthält: Dies sind die Bezeichnungen (Überschriften) der Spalten, die die Selektionskriterien enthalten sowie die Werte der Selektionskriterien. UND-Bedingungen stehen nebeneinander, ODER-Bedingungen untereinander.

DBSUMME(A:C;“Umsatz“;**E3:E4**)

	A	B	C	D	E	F	G
1	Artikel	ReNr	Umsatz				
2	1	1	1.000				
3	1	2	500		Artikel	Umsatz	
4	2	2	250		3	1.350	=DBSUMME(A:C;"Umsatz";E3:E4)
5	3	2	750				
6	2	3	5.000				
7	3	4	400				
8	4	4	600				
9	1	5	300				
10	3	5	200				
11	1	6	100				

Anforderungen an die Datenbank

Die Datenbank muss vertikal verlaufen und die erste Zeile der Datenbank muss für jede Spalte eine eindeutige Überschrift enthalten. (Mittels der Überschriften werden der auszuwertende Bereich sowie die Bereiche der Selektionskriterien gesteuert.)

Grundsätzlich kann sich die Datenbank *irgendwo* in einem Tabellenblatt befinden – der Bereich A1:C20 ist beispielsweise ebenso möglich wie C10:Z27.

DBSUMME(**A1:C20**;"Umsatz";E3:E4)
DBSUMME(**C10:Z27**;"Umsatz";A3:A4)

Empfehlenswert aber ist, die Datenbank stets in Zeile 1 beginnen zu lassen. Bleiben die Zellen unterhalb der Datenbank frei von sonstigen Daten, kann die Datenbank als ganze Spalten definiert werden: A:C.

DBSUMME(**A:C**;"Umsatz";E3:E4)

Oft bietet sich an, die Datenbank in ein separates Tabellenblatt auszulagern, die Auswertungen also in einem anderen Blatt vorzunehmen. Als Datenbank kann dann auch das gesamte Arbeitsblatt definiert werden: 1:65536

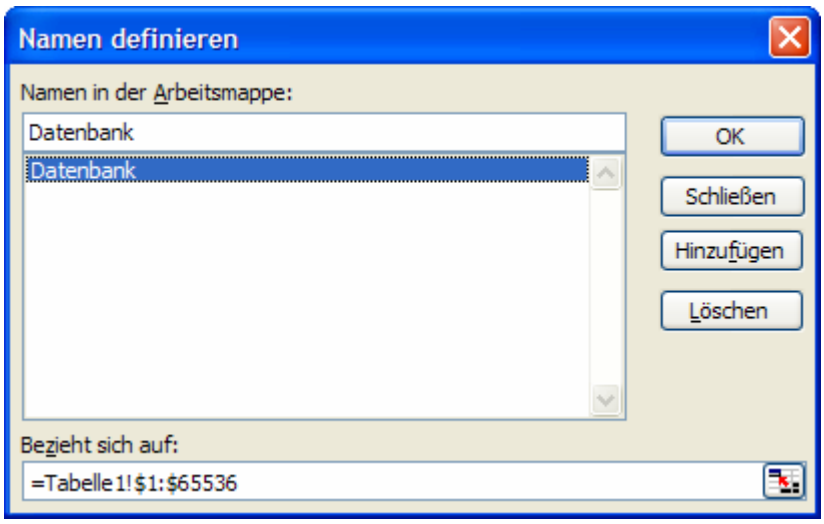
DBSUMME(**Tabelle1!$1:$65536**;"Umsatz";E3:E4)

Datenbank als Name definieren

Selbstverständlich können Sie für die Datenbank auch einen *Namen* definieren:

→ Einfügen → Namen → Definieren.

In folgendem Beispiel vergeben wir den Namen „Datenbank“ für den Bereich (→ bezieht sich auf): das gesamte Tabellenblatt „Tabelle1“



In der Formel verwenden Sie den Namen anschließend so:

DBSUMME(**Datenbank**;"Umsatz";E3:E4)

DBSUMME

Das folgende Beispiel zeigt eine Liste von Rechnungspositionen: Darin die Artikel-, Rechnungs- und Kundennummer sowie Umsatz und Rohertrag.

	A	B	C	D	E
1	Artikel	ReNr	KundeNr	Umsatz	Rohertrag
2	1	1	10001	1.000	200
3	1	2	10005	500	100
4	2	2	10005	250	50
5	3	2	10005	750	150
6	2	3	10006	5.000	1.000
7	3	4	10002	400	80
8	4	4	10002	600	120
9	1	5	10004	300	60
10	3	5	10004	200	40
11	1	6	10001	100	20
12	4	6	10001	200	40
13	5	6	10001	200	40
14	5	7	10004	1.250	250
15					

Interessieren sollen uns: → der Umsatz mit Artikel 2, → der Rohertrag aus Rechnung 4 und → der Umsatz mit Kunde 10001

	A	B	C	D	E	F	G	H	I
1	Artikel	ReNr	KundeNr	Umsatz	Rohertrag				
2	1	1	10001	1.000	200				
3	1	2	10005	500	100		Artikel	Umsatz	
4	2	2	10005	250	50		2	5.250	=DBSUMME(A:E;H3;G3:G4)
5	3	2	10005	750	150				
6	2	3	10006	5.000	1.000		ReNr	Rohertrag	
7	3	4	10002	400	80		4	200	=DBSUMME(A:E;H6;G6:G7)
8	4	4	10002	600	120				
9	1	5	10004	300	60		KundeNr	Umsatz	
10	3	5	10004	200	40		10001	1.500	=DBSUMME(A:E;H9;G9:G10)
11	1	6	10001	100	20				
12	4	6	10001	200	40				
13	5	6	10001	200	40				
14	5	7	10004	1.250	250				

Zu oben verwendeter Funktion:

=DBSUMME(A:E;H3;G3:G4)

In den Spalten A bis E befindet sich die Datenbank.

=DBSUMME(A:E;H3;G3:G4)

Im Feld H3 befindet sich die Bezeichnung (Überschrift) der auszuwertenden Spalte. In diesem Fall ist es diejenige Spalte der Datenbank, die die Bezeichnung „Umsatz“ trägt. In der Formel hätte ebenso C3 oder „Umsatz“ verwendet werden können:

=DBSUMME(A:E;C3;G3:G4)
=DBSUMME(A:E;“Umsatz“;G3:G4))

=DBSUMME(A:E;H3;G3:G4)

In diesen Feldern sind die Kriterien hinterlegt: In Feld G3 ist das zunächst die Überschrift „Artikel“ und in G4 der ihr zugeordnete Wert „10001“, nach dem zu selektieren ist.

Änderung von Suchkriterien und auszuwertender Spalte

Ohne Änderung der hinterlegten Formeln können Sie andere Berechnungen anstellen, in dem Sie die definierten Selektionskriterien ändern:

	A	B	C	D	E	F	G	H	I
1	Artikel	ReNr	KundeNr	Umsatz	Rohertrag				
2	1	1	10001	1.000	200				
3	1	2	10005	500	100		Artikel	Umsatz	
4	2	2	10005	250	50		3	1.350	=DBSUMME(A:E;H3;G3:G4)
5	3	2	10005	750	150				
6	2	3	10006	5.000	1.000		ReNr	Umsatz	
7	3	4	10002	400	80		4	1.000	=DBSUMME(A:E;H6;G6:G7)
8	4	4	10002	600	120				
9	1	5	10004	300	60		KundeNr	Rohertrag	
10	3	5	10004	200	40		10001	300	=DBSUMME(A:E;H9;G9:G10)
11	1	6	10001	100	20				
12	4	6	10001	200	40				
13	5	6	10001	200	40				
14	5	7	10004	1.250	250				

In G4 wurde die Artikelnummer geändert – Ergebnis ist nun der Umsatz für diesen Artikel.

In H6 wurde die zu berechnende Spalte geändert – Ergebnis ist nun nicht mehr der „Rohertrag“, sondern der „Umsatz“.

Ebenso wurde in H9 die zu berechnende Spalte gewechselt.

Wollten Sie dies mit anderen Funktionen nachbilden, könnte Ihre Lösung z. B. so aussehen:
=SUMMEWENN(INDEX(A:E;;VERGLEICH(G3;1:1;));G4;INDEX(A:E;;VERGLEICH(H3;1:1;)))

Mehrere Suchkriterien

Im folgenden Beispiel wird der Umsatz für Artikel 1 des Kunden 10001 ermittelt.

	A	B	C	D	E	F	G	H	I	J
1	Artikel	ReNr	KundeNr	Umsatz	Rohertrag					
2	1	1	10001	1.000	200					
3	1	2	10005	500	100		Artikel	KundeNr	Umsatz	
4	2	2	10005	250	50		1	10001	1.100	=DBSUMME(A:E;I3;G3:H4)
5	3	2	10005	750	150					
6	2	3	10006	5.000	1.000					
7	3	4	10002	400	80					
8	4	4	10002	600	120					
9	1	5	10004	300	60					
10	3	5	10004	200	40					
11	1	6	10001	100	20					
12	4	6	10001	200	40					
13	5	6	10001	200	40					
14	5	7	10004	1.250	250					

Die Suchkriterien der Funktion wurden erweitert: Die Bedingungen stehen nun in den Zellen G3 bis H4

=DBSUMME(A:E;I3;G3:H4)

Vielleicht interessieren Sie sich nun für die Summe aller Rechnungspositionen, die größer als 1000 Euro sind? Bitte sehr:

	A	B	C	D	E	F	G	H	I	J
1	Artikel	ReNr	KundeNr	Umsatz	Rohertrag					
2	1	1	10001	1.000	200					
3	1	2	10005	500	100		Artikel	Umsatz	Umsatz	
4	2	2	10005	250	50			>=1000	7.250	=DBSUMME(A:E;I3;G3:H4)
5	3	2	10005	750	150					
6	2	3	10006	5.000	1.000					
7	3	4	10002	400	80					
8	4	4	10002	600	120					
9	1	5	10004	300	60					
10	3	5	10004	200	40					
11	1	6	10001	100	20					
12	4	6	10001	200	40					
13	5	6	10001	200	40					
14	5	7	10004	1.250	250					

Die Formel in I4 bleibt unangetastet. Verändert wurden die Suchkriterien:

- Für das Suchkriterium „Artikel“ gibt es keine Kriterien. Der Eintrag in G4 ist gelöscht, dort steht nun nichts mehr.
- Wir benötigen nun aber ein weiteres Selektionskriterium: die Spalte „Umsatz“. In Zelle H3 steht nun der Text „Umsatz“.
- Das Selektionskriterium für „Umsatz“ lautet: alles größer oder gleich 1000. Das steht in Zelle H4:

  =“>=1000“

Erfassung der Suchkriterien

Die Suchkriterien sind wie oben dargestellt einzutragen:

1. Gleichheitszeichen
2. Anführungszeichen
3. Bedingung
4. Anführungszeichen

In unserem Beispiel oben könnte noch genügen, die Kriterien ohne Anführungszeichen einzutragen. Oft aber führt es zu fehlerhaften Auswertungen. Besser, Sie sich gleich an, die Kriterien stets so zu erfassen.

Es kann sich anbieten, *alle denkbaren* Selektionskriterien vorzudefinieren – und fallweise diejenigen zu nutzen, die – für welchen Zweck auch immer – benötigt werden.

	A	B	C	D	E
1	Artikel	ReNr	KundeNr	Umsatz	Rohertrag
2	1	1	10001	1.000	200
3	1	2	10005	500	100
4	2	2	10005	250	50
5	3	2	10005	750	150
6	2	3	10006	5.000	1.000
7	3	4	10002	400	80
8	4	4	10002	600	120
9	1	5	10004	300	60
10	3	5	10004	200	40
11	1	6	10001	100	20
12	4	6	10001	200	40
13	5	6	10001	200	40
14	5	7	10004	1.250	250

	G	H	I	J	K	L	M
3	Artikel	ReNr	KundeNr	Umsatz	Rohertrag	Umsatz	
4	<=4	>2	>10002	<=1000		500	=DBSUMME(A:E;L3;G3:K4)

Und- / Oder-Bedingungen

Bisher (oben) wurden nur *Und*-Bedingungen verwendet – der selektierte Bereich hatte *sowohl* die eine Bedingung zu erfüllen (z. B. ArtikelNr 4) *als auch* die zweite (z. B. RechnungsNr. 2) *und* eine dritte (z. B. Umsatz größer 1.000) usw. Summiert wurden nur solche Zeilenpositionen, die alle Bedingung erfüllten.

UND-Bedingungen stehen *nebeneinander*.

Es gibt auch die *Oder's* –

ODER-Bedingungen werden *untereinander* aufgeführt:

	A	B	C	D	E	F	G	H	I	J
1	Artikel	ReNr	KundeNr	Umsatz	Rohertrag					
2	1	1	10001	1.000	200					
3	1	2	10005	500	100		Artikel	KundeNr	Umsatz	
4	2	2	10005	250	50		1	10001	2.550	=DBSUMME(A:E;I3;G3:H5)
5	3	2	10005	750	150		5			
6	2	3	10006	5.000	1.000					
7	3	4	10002	400	80		Artikel	KundeNr	Umsatz	
8	4	4	10002	600	120		1	10001	1.300	=DBSUMME(A:E;I7;G7:H9)
9	1	5	10004	300	60		5	10001		
10	3	5	10004	200	40					
11	1	6	10001	100	20					
12	4	6	10001	200	40					
13	5	6	10001	200	40					
14	5	7	10004	1.250	250					

Blicken wir zunächst auf Zelle I8 – den unteren Block rechts:

Die Und/Oder-Bedingung besagt hier:

- *entweder* in Spalte „Artikel" eine 1 UND in Spalte „KundeNr" eine 10001
- *oder* in Spalte „Artikel" eine 5 UND in Spalte „KundeNr" eine 10001.

Im Ergebnis erhalten wir die Umsätze für Kunde 10001 mit den Artikeln 1 und 5.

Dass wir mit der Definition der Bedingungen sorgfältig umgehen müssen sehen wir in Zelle I4.

Die Und/Oder-Bedingung besagt hier:

- *entweder* in Spalte „Artikel" eine 1 UND in Spalte „KundeNr" eine 10001
- *oder* in Spalte „Artikel" eine 5 UND in Spalte „KundeNr" beliebiges (keine Bedingung).

Im Gegensatz zum Ergebnis in Zelle I8 werden hier alle Umsätze mit Artikel 5 aufsummiert, nicht nur jene mit Kunde 10001. (Was durchaus gewollt sein kann!)

Dynamischer Bereich für *ODER*-Suchkriterien

	A	B	C	D	E	F	G
1	Artikel	KundeNr	Umsatz				
2	1	10001	1.000				
3	1	10005	500		Artikel	Umsatz	
4	2	10005	250		1	10.750	=DBSUMME(A:C;F3;E3:E10)
5	3	10005	750				
6	2	10006	5.000				
7	3	10002	400				
8	4	10002	600				
9	1	10004	300				
10	3	10004	200				
11	1	10001	100				
12	4	10001	200				
13	5	10001	200				
14	5	10004	1.250				

Oft ist die Angabe eines Bereichs für Suchkriterien gewünscht (in diesem Beispiel sei es der Bereich E3:E10), um damit – ohne die Formel anpassen zu müssen – selektive Auswertungen zu erstellen. Oben gelingt dies nicht, weil die ODER-Bedingungen in Zellen E4 bis E10 keine Definitionen enthalten – folglich wird in F4 die Summe aller vorhandenen Umsätze gebildet.

Eine *wenig elegante* Problemlösung:

	E	F	G
3	Artikel	Umsatz	
4	1	3.350	=DBSUMME(A:C;F3;E3:E10)
5	5		
6	###		
7	###		
8	###		
9	###		
10	###		

Der Bereich mit den Bedingungen wird um in der Datenbank nicht vorhandene Zeichen *aufgefüllt* – die Bedingung lautet: addiere die „Umsätze“, wenn „Artikel“ eine 1 oder eine 5 oder ein „###“ enthält.

Eleganter ist folgende Lösung:

	E	F	G
3	Artikel	Umsatz	
4	1	3.350	=DBSUMME(A:C;F3;BEREICH.VERSCHIEBEN(E3;0;0;ANZAHL2(E:E)))
5	5		
6			
7			
8			
9			
10			

Oben wird der Bereich der Suchkriterien auf jene Zellen der Spalte E begrenzt, die auch Daten enthalten: in Spalte E sind es hier insgesamt 3 Zellen. Der bei E3 beginnende Bereich für die Suchkriterien ist damit auf 3 Zeilen definiert – hier gilt also der Bereich E3:E5.

Voraussetzung ist hier, dass in Spalte E keine weiteren Zellen belegt sind! Alternativ hätte hier die Zählung belegter Zellen innerhalb der Funktion BEREICH.VERSCHIEBEN mit ANZAHL2(E3:E10) gewählt werden können.

Zur Anordnung der Bedingungen

Datenbank (Ausschnitt)

	A	B	C
1	Monat	WGr	Umsatz
2	1	A	1.000
3	1	B	100
4	2	A	2.000
5	2	B	200
6	3	A	3.000
7	3	B	300
8	4	A	4.000
9	4	B	400

Die *klassische* Anordnung der Bedingungen:

	E	F	G	H	I	J	K	L
20	WGr	Monat	WGr	Monat	WGr	Monat	WGr	Monat
21	A	=1	A	=2	A	=3	A	=4
22								
23								
24	Monat	1	2	3	4			
25	A	1.000	2.000	3.000	4.000			

Formeln:
F25 =DBSUMME($A:$C;"Umsatz";E20:F21)
G25 =DBSUMME($A:$C;"Umsatz";G20:H21)
usw.

Eine etwas *platzsparende* Alternative:

	E	F	G	H	I	J
12	Monat	WGr	Monat	Monat	WGr	Monat
13	=1	A	=2	=3	A	=4
14						
15						
16	Monat	1	2	3	4	
17	A	1.000	2.000	3.000	4.000	

Formeln:
F17 =DBSUMME($A:$C;"Umsatz";E12:F13)
G17 =DBSUMME($A:$C;"Umsatz";F12:G13)
usw.

Eine weitere (kompakte) Alternative:

	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	WGr	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat
2	A	<=12	<=11	<=10	<=9	<=8	<=7	<=6	<=5	<=4	<=3	<=2	<=1
3	WGr	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat	Monat
4	B	<=12	<=11	<=10	<=9	<=8	<=7	<=6	<=5	<=4	<=3	<=2	<=1
5													
6													
7	Monat	1	2	3	4	5	6	7	8	9	10	11	12
8	A	1.000	2.000	3.000	4.000	5.000	6.000	7.000	8.000	9.000	10.000	11.000	12.000
9	B	100	200	300	400	500	600	700	800	900	1.000	1.100	1.200

Formel in F8 und nach rechts kopieren:

= DBSUMME($A:$C;"Umsatz";INDIREKT("Z1S5:Z2S"&18-SPALTE(A1);0))-SUMME($E8:E8)

DBANZAHL – DBANZAHL2

Mit der Funktion DBANZAHL lässt sich die Anzahl der selektierten Werte zählen (gezählt werden nur Zahlen, nicht sonstige Feldeinträge wie Text).

	A	B	C	D	E	F	G	H	I	J
1	Artikel	ReNr	KundeNr	Umsatz	Rohertrag					
2	1	1	10001	1.000	200					
3	1	2	10005	500	100		Artikel		Artikel	
4	2	2	10005	250	50		4		2	=DBANZAHL(A:E;I3;G3:G4)
5	3	2	10005	750	150					
6	2	3	10006	5.000	1.000		Artikel		Artikel	
7	3	4	10002	400	80		2		7	=DBANZAHL(A:E;I6;G6:G9)
8	4	4	10002	600	120		3			
9	1	5	10004	300	60		4			
10	3	5	10004	200	40					
11	1	6	10001	100	20		Artikel	KundeNr	Artikel	
12	4	6	10001	200	40		1	10001	2	=DBANZAHL(A:E;I11;G11:H12)
13	5	6	10001	200	40					
14	5	7	10004	1.250	250					

In Zelle I4 wird die Anzahl der Artikel mit der Zahl 4 gezählt. Dies entspricht hier etwa der Formel =ZÄHLENWENN(A:A;4)

In Zelle I7 wird die Anzahl der Artikel mit den Zahlen 2, 3 und 4. Dies entspricht z. B. den Formeln
=ZÄHLENWENN(A:A;">=2")-ZÄHLENWENN(A:A;">4")
=SUMME(ZÄHLENWENN(A:A;{2.3.4}))

In Zelle I12 wird die Anzahl jener Artikel gezählt, die auf KundeNr 10001 entfallen. Entsprechend
=SUMMENPRODUKT((C2:C65536=10001)*(A2:A65536=1))

Auf die Funktion DBANZAHL2 gehen wir hier nicht näher ein. Nur so viel: DBANZAHL2 unterscheidet sich von DBANZAHL so, wie sich die Funktionen ANZAHL2 und ANZAHL voneinander unterscheiden - nämlich in der Beachtung von Zahlen und Texten.

DBMAX – DBMIN – DBMITTELWERT

Die Funktion DBMAX ermittelt den größten, DBMIN den kleinsten und DBMITTELWERT den mittleren (durchschnittlichen) Wert für die vorgegebenen Suchkriterien in der Datenbank.

	A	B	C	D	E	F	G	H	I
1	Artikel	ReNr	KundeNr	Umsatz	Rohertrag				
2	1	1	10001	1.000	200				
3	1	2	10005	500	100		ReNr	Umsatz	
4	2	2	10005	250	50		5	300	=DBMAX(A:E;H3;G3:G4)
5	3	2	10005	750	150				
6	2	3	10006	5.000	1.000		Artikel	Umsatz	
7	3	4	10002	400	80		2	250	=DBMIN(A:E;H6;G6:G7)
8	4	4	10002	600	120				
9	1	5	10004	300	60		KundeNr	Rohertrag	
10	3	5	10004	200	40		10002	100	=DBMITTELWERT(A:E;H9;G9:G10)
11	1	6	10001	100	20				
12	4	6	10001	200	40				
13	5	6	10001	200	40				
14	5	7	10004	1.250	250				

In H4 wird der größte Umsatzposten aus Rechnung Nr. 5 angezeigt. Den kleinsten Umsatz mit Artikel 2 zeigt Zelle H7. Beide Funktionen zeigen nichts anderes als den größten bzw. kleinsten Wert; ein mehrfaches Vorkommen des größten/kleinsten Wertes hat dafür keine Bedeutung.

In H10 wird der durchschnittliche Rohertrag je Belegposten für Kunde 10002 gebildet.

DBAUSZUG

Die Funktion DBAUSZUG liefert den gesuchten Zellinhalt für die Bedingung. DBAUSZUG liefert nur dann ein Ergebnis, wenn die Bedingungen ausschließlich von *einem* Datensatz erfüllt sind – die Bedingungen müssen also so exakt gewählt sein, dass DBAUSZUG nur diesen einen Datensatz findet.

DBAUSZUG entspricht etwa der Funktion INDEX mit VERGLEICH oder auch dem SVERWEIS. Die Spalte wird mit der Überschrift ausgewählt (bei INDEX machte das die Funktion VERGLEICH), die Zeilennummer wird durch die Kriterienangabe bestimmt (bei INDEX machte das ebenfalls die Funktion VERGLEICH – mit dem Vergleichswert „0“ für eine exakte Übereinstimmung).

	A	B	C	D	E	F	G	H	I
1	KdNr	Kunde	Straße	Ort			KdNr	KdNr	
2	10004	Antons AG	D-Str. 4	44444 DStadt			10001	10002	
3	10005	Bertas OHG	E-Str. 5	55555 EStadt		Kunde	Egon & Fritz GbR	Connys GmbH	=DBAUSZUG($A:$D;$F3;H$1:H$2)
4	10002	Connys GmbH	B-Str. 2	22222 BStadt		Straße	A-Str. 1	B-Str. 2	=DBAUSZUG($A:$D;$F4;H$1:H$2)
5	10003	Dieters KG	C-Str. 3	33333 COrt		Ort	11111 ADorf	22222 BStadt	=DBAUSZUG($A:$D;$F5;H$1:H$2)
6	10001	Egon & Fritz GbR	A-Str. 1	11111 ADorf					
7	10006	Huberts Laden	F-Str. 6	66666 FBurg			Kunde		
8							egon*		
9						Kunde	Egon & Fritz GbR	=DBAUSZUG($A:$D;$F9;G7:G8)	
10									
11							Straße		
12							*Str. 6		
13						Kunde	Huberts Laden	=DBAUSZUG($A:$D;$F13;G11:G12)	
14									
15							Straße		
16							*Str.		
17						Kunde	#ZAHL!	=DBAUSZUG($A:$D;$F17;G15:G16)	

Zellen G1:H5 zeigen die Möglichkeiten der Funktion auf: Es können einzelne Daten ausgelesen werden, auch die Erstellung einer Matrix ist möglich.

Die Verwendung von Platzhaltern (* Stern/Multiplikationszeichen und ? Fragezeichen) für Texte ist möglich. G9 / G13: Da es nur einen „Egon“ in Spalte „Kunde“ gibt, und nur eine auf „Str. 6“ endende Straßenangabe in Spalte „Straße“, liefert DBAUSZUG in den Zellen G9 und G13 zutreffende Daten.

In Zelle G17 wird DBAUSZUG nicht mehr fündig, weil es für das Suchkriterien mehr als einen Treffer gibt. Deshalb wird hier ein Fehler gemeldet.

Übrigens: Der Platzhalter in Zelle G8 „egon*“ wäre nicht nötig gewesen, die Funktion liefert dasselbe Ergebnis ohne den Platzhalter für ein beliebig folgendes Zeichen.

Verwenden der Platzhalter

Wie wir oben bereits gesehen haben, können die Platzhalter * und ? für DB-Funktionen verwendet werden. Beachte: Platzhalter können ausschließlich für Text verwendet werden – nicht für Zahlen!

	A	B	C	D	E	F	G
1	KdNr	Kunde	Umsatz		KdNr		
2	10004	Antons AG	100		100*	0	=DBANZAHL2(A:C;A1;E1:E2)
3	10005	Bertas OHG	200				
4	10002	Connys GmbH	300		KdNr		
5	10003	Dieters KG	400		10*	0	=DBSUMME(A:C;"Umsatz";E4:E5)
6	10001	Egon & Fritz GbR	500				
7	10006	Huberts Laden	600		Kunde		
8					*	2100	=DBSUMME(A:C;"Umsatz";E7:E8)

E2 / E5: Die KdNr liegt als Zahl vor. DBANZAHL2 und DBSUMME finden im Bereich „KdNr“ keine Texte, die mit „100“ oder „10“ beginnen.

E8: Der Bereich „Kunde“ enthält Texte. Hier wird beliebiger Text gesucht, folglich kann DBSUMME eine Summe bilden.

DBPRODUKT

Folgendes Beispiel zeigt die Funktionsweise der Funktion DBPRODUKT: Es werden alle den Suchkriterien entsprechenden Elemente miteinander multipliziert.

	A	B	C	D	E	F	G	H	I
1	KdNr	Kunde	Mix	Inhalt			KdNr		
2	10001	A_GmbH	1	Menge			10004	160	=DBPRODUKT((A:C);"Mix";G1:G2)
3	10002	B_AG	2	Menge					
4	10003	C_Ltd.	3	Menge			KdNr		
5	10004	D_Ltd.	4	Menge			10002	3600	=DBPRODUKT((A:C);"Mix";G4:G6)
6	10005	E_Sarl	5	Menge			10003		
7	10001	A_GmbH	10,00 €	Preis					
8	10002	B_AG	20,00 €	Preis		KdNr	Inhalt		
9	10003	C_Ltd.	30,00 €	Preis		10005	Menge	250	=DBPRODUKT(A:D;C1;F8:G10)
10	10004	D_Ltd.	40,00 €	Preis		10005	Preis		
11	10005	E_Sarl	50,00 €	Preis					
12	10005	E_Sarl	2	Paul					

Auf den ersten Blick sieht das interessant aus. Doch auf den zweien Blick erschließt sich der tiefere Sinn der Funktion DBPRODUKT nicht – vor allem wirft dies einige Zweifel an einer sorgfältigen Gestaltung der Datenbank auf.

Nichtsdestotrotz mag es Datenbanken geben, die für DBPRODUKT geeignet aufgebaut sind und Datenbank-Besitzer, die derartige Berechnung anzustellen begehren. Womöglich gelingt einem mathematischen Genie mit dieser Funktion die Anzahl der Eier zu berechnen, die ein braunes Huhn im Vollrausch in Vollmondnächten zu legen vermag...

Beispielanwendungen

Beispiel 1

Datenbank

	A	B	C	D
1	KoArt	KoSt	KoStelle	Wert
2	4110	300	VA	1.000
3	4110	301	VA_1	800
4	4110	302	VA_2	750
5	4110	303	VA_3	850
6	4110	304	VA_4	700
7	4120	300	VA	100
8	4120	301	VA_1	80
9	4120	302	VA_2	75
10	4120	303	VA_3	85
11	4120	304	VA_4	70
12	4150	300	VA	250
13	4150	301	VA_1	200
14	4150	302	VA_2	188
15	4150	303	VA_3	213
16	4150	304	VA_4	175
17	4110	400	VI	1.250
18	4120	400	VI	125
19	4150	400	VI	313

Auswertung nach Kategorien *trotz* Oder-Bedingungen

	H	I	J
1	KoArt	Wert	
2	4110	5.350	=DBSUMME(A:D;$I$1;H$1:H2)-SUMME(I$1:I1)
3	4120	535	=DBSUMME(A:D;$I$1;H$1:H3)-SUMME(I$1:I2)
4	4150	1.339	=DBSUMME(A:D;$I$1;H$1:H4)-SUMME(I$1:I3)

Weitere Auswertungsbeispiele

	F	G	H	I	J
6		KoArt	KoArt	Wert	
7		>=4100	<4200	7.224	=DBSUMME(A:D;"Wert";G6:H7)
8					
9			KoStelle	Wert	
10			VA	5.536	=DBSUMME(A:D;"Wert";H$9:H10)-SUMME(I$9:I9)
11			VI	1.688	=DBSUMME(A:D;"Wert";H$9:H11)-SUMME(I$9:I10)
12					
13			KoStelle	Wert	
14			*_4	945	=DBSUMME(A:D;"Wert";H$13:H14)-SUMME(I$13:I13)
15					
16	KoArt	KoSt	KoSt	Wert	
17	4110	>=300	<400	4.100	=DBSUMME(A:D;"Wert";F$16:H17)-SUMME(I$16:I16)
18	4110	>=400	<500	1.250	=DBSUMME(A:D;"Wert";F$16:H18)-SUMME(I$16:I17)

Beispiel 2

Datenbank (hier in Tabellenblatt *Datenbank2* ausgelagert)

	A	B	C	D	E
1	KdNr	Kunde	Staat	Ort	Umsatz
2	10001	A_GmbH	BRD	Wuppertal	1.000,00 €
3	10002	B_AG	BRD	Düsseldorf	
4	10003	C_Ltd.	GB	London	100,00 €
5	10004	D_Ltd.	GB	Edinburgh	500,00 €
6	10005	E_Sarl	F	Paris	10.000,00 €
7	10006	F_OHG	BRD	München	5.000,00 €
8	10007	G_Sarl	F	Metz	750,00 €
9	10008	H_KG	BRD	Hamburg	300,00 €
10	10009	J_SA	BRD	Berlin	
11	10010	K & L	BRD	Berlin	250,00 €

Datenbank-Auszug aufgrund der Kundennummer

	A	B	C
1	KdNr		
2	10007		
3			
4	Kunde	G_Sarl	=DBAUSZUG(Datenbank2!$1:$65536;A4;$A$1:$A$2)
5			
6	Staat	F	=DBAUSZUG(Datenbank2!$1:$65536;A6;$A$1:$A$2)
7	Ort	Metz	=DBAUSZUG(Datenbank2!$1:$65536;A7;$A$1:$A$2)
8			
9	Umsatz	750,00 €	=DBAUSZUG(Datenbank2!$1:$65536;A9;$A$1:$A$2)
10	Umsatz	750,00 €	=DBSUMME(Datenbank2!$1:$65536;A10;$A$1:$A$2)

Weitere Auswertungsbeispiele

	A	B	C	D
1	Staat	Anzahl Kunden	Summe Umsatz	
2	BRD	6	6.550 €	
3	GB	2	600 €	
4	F	2	10.750 €	
5				
6	Staat	Anzahl Kunden		
7	BRD	6		=DBANZAHL2(Datenbank2!$1:$65536;"Staat";$A$6:$A7)-SUMME(B$6:B6)
8	GB	2		=DBANZAHL2(Datenbank2!$1:$65536;"Staat";$A$6:$A8)-SUMME(B$6:B7)
9	F	2		=DBANZAHL2(Datenbank2!$1:$65536;"Staat";$A$6:$A9)-SUMME(B$6:B8)
10				
11	Staat	Summe Umsatz		
12	BRD	6.550 €		=DBSUMME(Datenbank2!$1:$65536;"Umsatz";$A$11:$A12)-SUMME(B$11:B11)
13	GB	600 €		=DBSUMME(Datenbank2!$1:$65536;"Umsatz";$A$11:$A13)-SUMME(B$11:B12)
14	F	10.750 €		=DBSUMME(Datenbank2!$1:$65536;"Umsatz";$A$11:$A14)-SUMME(B$11:B13)