

Datenmodellierungs -Assistenten auch in Access 97 für professio- nelle Anwendungen ungeeignet

Der Markterfolg von Access hat erheblich zur Popularisierung relationaler Datenbanken auf dem PC beigetragen. Viele Anwender machen z.B. die Erfahrung, daß ihre Excel-Lösungen von der Datenorganisation her an die Grenzen einer Tabellenkalkulation stoßen. Warum sollte einem solchen "Upsizer" nicht mit Hilfe von Datenmodellierungsassistenten der Umstieg auf eine RDBMS-gestützte Datenhaltung erleichtert werden? Die Antwort ist einfach: weil gerade der Datenbankneuling in der Illusion bestärkt wird, eine Datenbank entwerfe sich quasi von selbst! "Wissen um Relationen ist nicht erforderlich" (Computerwoche 1995, Nr. 42).

Nichts gegen Assistenten. Aber sie sind nur dann sinnvoll, wenn der Anwender seine Aufgabe mit ihnen schneller erledigen kann als ohne sie. Insbesondere den nur gelegentlichen Benutzer können sie von nicht jederzeit präsentem Bedienungswissen entlasten, ohne die überzogene Erwartung zu wecken, das erforderliche konzeptionelle Know-how sei verzichtbar. Sofern eine vom Assistenten produzierte Lösung noch individuell angepaßt werden muß, darf der für dieses Customizing erforderliche Zeitaufwand zudem nicht die ursprünglich durch den Einsatz des Assistenten erzielte Zeitersparnis überkompensieren.

An relativ anspruchslosen Beispielen - die Beispieldateien finden Sie unter <http://www.nsl.de> - soll gezeigt werden, daß die Datenmodellierungsassistenten von Access 7.0 und unverändert auch von Access 97 diesen Anforderungen nicht gerecht werden. Während der Datenbankassistent, der Tabellenassistent und der Datentypassistent für den Entwurf einer kompletten Datenbank, einzelner Tabellen bzw. einzelner Felder gedacht sind, dient der Tabellenanalyseassistent der Normalisierung eines vorhandenen Tabellenbestandes. Einem CASE-Tool-gestützten Datenbankdesign können sie jedoch nicht das Wasser reichen.

Die drei erstgenannten Assistenten werden darauf überprüft, ob sie die Access-Demodatenbank NORDWIND "nachzubauen" helfen, die hier quasi als Fachkonzept fungiert. Beginnen wir mit dem **Datenbankassistenten**, der 22 mitgelieferte Datenbankvorlagen anbietet, 11 davon laut Microsoft für geschäftliche Anwendungen. Da sich die NORDWIND-Datenbank im Kern mit dem Bestell- und Beschaffungswesen befaßt, liegt es nahe, die Datenbankvorlage "Bestellabwicklung" heranzuziehen.

Der Datenbankassistent erzeugt eine Basislösung CW_01.MDB. Sechs der neun Nutdatentabellen scheinen für das angestrebte NORDWIND-Datenmodell geeignet: Kunden, Bestellungen, Bestelldetails, Artikel, Versandarten und Personal. Es fehlen Tabellen für die Lieferanten und Artikelkategorien. Das Entfernen der überflüssigen Tabellen Zahlungen, Zahlungsweisen und Firmeninformationen führt beim Öffnen von CW_02.MDB zu einem wenn auch nicht kritischen Laufzeitfehler, weil die Tabelle Firmeninformationen nicht mehr

zur Verfügung steht, aus der die Daten für das beim Applikationsstart automatisch geöffnete Formular Hauptübersicht ausgelesen werden.

Generell ist anzumerken, daß Abfragen, Formulare und Berichte als externe Sichten, die auf eliminierte Tabellen zugreifen, nunmehr funktionslos werden und schlimmstenfalls die Dialogführung lahmlegen. Andere Informationen zum Personal, zu Artikeln und zu Versandarten sowie die Berichte sind wegen der gelöschten Tabellen überhaupt nicht mehr erreichbar. **Fazit:** um die Funktionsfähigkeit der vom automatisch Datenbankassistenten miterzeugten Dialogführung zu erhalten, muß nach dem Entfernen überflüssiger Tabellen ein erheblicher Customizing-Aufwand betrieben werden.

Im nächsten Schritt soll die fehlende Tabelle "Lieferanten" mit dem **Tabellenassistenten** erstellt werden. Wir haben insofern Glück, als sich unter den 25 geschäftlichen Beispieltabellen auch eine Lieferantentabelle befindet (CW_03.MDB). **Fazit:** der Tabellenassistent dürfte nur in Ausnahmefällen genau die zu der gewünschten Entität passende Tabellenvorlage bereithalten. Da fehlende Felder ohnehin ebenso wie die Regeln der referentiellen Integrität (RI) für hinzukommende Beziehungen manuell nachzutragen sind, erweist sich der Tabellenassistent als nur sehr begrenzt nützlich.

Stellt der Tabellenassistent keine geeignete Vorlage bereit, was eher die Regel als die Ausnahme sein dürfte, bleibt noch der **Datenblattassistent**, der ein leeres Datenblatt öffnet, in das der Entwickler direkt Daten eingeben kann (CW_04.MDB). Seine Leistungsfähigkeit reduziert sich auf die automatische Auswahl eines Datentyps mit begrenzter Intelligenz. Obwohl wir z.B. für die Kategoriennummer fortlaufende Zahlen von 1 bis 8 eingegeben haben, kommt der Assistent nicht auf die Idee, daß es sich hier um den Datentyp AutoWert handeln könnte. **Fazit:** Der Datenblattassistent leistet nicht mehr, als aus den vom Anwender eingegebenen Beispieldaten die Felddatentypen Text, ganze Zahlen, gebrochene Zahlen, Währung, Datum/Zeit und Ja/Nein herauszulesen. Da die Unterscheidung dieser Datentypen selbst den unbedarftesten Anwender nicht überfordern dürfte, kann der Datenblattassistent als schlicht und einfach überflüssig bezeichnet werden.

Geradezu abenteuerlich ist, was der Datenbankassistent an **Beziehungen und referentiellen Integritätsregeln** produziert (CW_05.MDB). Dort, wo der Datenbankassistent die Beziehungen mit referentieller Integrität belegt, greift er gleich wagemutig zur Aktualisierungs- und Löschweitergabe (Tab. 1). Wegen der mehrstufigen Löschweitergabe von der Kundentabelle auf die Bestellungstabelle und von dieser wiederum auf die Bestelldetailtabelle zieht das automatische Löschen von Kundendatensätzen gravierende Datenverluste in den beiden anderen Tabellen nach sich!

Master-Tabelle	Detail-Tabelle	Datenbankassistent	NORDWIND
Artikel	Bestelldetails	RI, LW, AW	RI
Bestellungen	Bestelldetails	RI, LW, AW	RI
Kunden	Bestellungen	RI, LW, AW	RI, AW
Personal (Detail !)	Bestellungen (Master !)	Outer Join ohne RI (!!!)	RI
Versandfirmen (Detail !)	Bestellungen (Master !)	Outer Join ohne RI (!!!)	RI

*Erläuterung: RI = mit referentieller Integrität, LW = Löschweitergabe, AW = Aktualisierungsweitergabe
 Tab. 1: Mängel der vom Datenbankassistenten definierten Regeln zur referentiellen Integrität*

Es geht hier nicht darum, ob in der NORDWIND-Datenbank nicht z.B. eine Löschweitergabe von der Tabelle "Bestellungen" auf die Tabelle "Bestelldetails" sinnvoller wäre, sondern ausschließlich darum, mit welcher RI-Regel der Entwickler zunächst auf der "sicheren Seite" ist. Unter diesem Gesichtspunkt spricht viel dafür, daß der Datenbankassistent Beziehungen prototypisch grundsätzlich mit referentieller Integrität und Aktualisierungsweitergabe, aber ohne Löschweitergabe erstellen sollte. Noch merkwürdiger sind die Beziehungen, die der Datenbankassistent zwischen den Tabellen Personal und Versandfirmen einerseits und der Tabelle Bestellungen andererseits definiert: er vertauscht nicht nur die Master- und Detailseite, sondern entscheidet sich statt für Gleichheits- für Inklusionsverknüpfungen - diese zudem noch in der falschen Richtung und ohne referentielle Integrität! Es kann also Bestellungen geben, die von nicht vorhandenen Mitarbeitern bearbeitet und von nicht vorhandenen Spediteuren ausgeliefert werden! Um diesem offensichtlichen Unfug gegenzusteuern, richtet der Datenbankassistent die Fremdschlüsselfelder in der Bestellungstabelle als Nachschlagfelder mit Listenfeldbeschränkungen ein. Diese Lösung darf man wohl als "von hinten durch die Brust ins Auge geschossen" bezeichnen. **Fazit:** Der Datenbankassistent produziert befremdliche RI-Regeln und Verknüpfungseigenschaften, die dringend der Nacharbeit bedürfen, was aber ohne relationales Know-how nicht zu bewerkstelligen ist.

"Der sogenannte 'Tabellenanalyseassistent' ist ein Kind der Erkenntnis, daß viele Anwender mit dem Entwurf relationaler Datenbanken überfordert sind. Er hilft 'flache' Dateistrukturen zu normalisieren" (c't 1996, H. 2). Diese rosige Aussicht provoziert die Gegenprobe, ob der Tabellenanalyseassistent nicht vielleicht selbst mit dem Normalisieren überfordert ist. Wir wählen das einfache Beispiel einer fiktiven Autovermietung (Tab. 2).

Mieter-ID	Name	Branche	Wagen-ID	Fahrzeugtyp	Baujahr	Versicherung	Mietsatz	Mietdauer
M1	AEG	B2	W12	Corsa	93	Allianz	185	31
M1	AEG	B2	W84	Clio	95	Lloyd	185	160
M2	Bolle	B1	W45	Vectra	94	Allianz	235	215
M3	IBM	B3	W52	DB S320	94	Lloyd	539	8
M3	IBM	B3	W64	Clio	94	Iduna	185	324
M3	IBM	B3	W99	DB E200	95	Iduna	449	87
M4	KaDeWe	B1	W75	Golf	94	Allianz	199	106
M5	Siemens	B2	W12	Corsa	93	Allianz	185	289
M5	Siemens	B2	W99	DB E200	95	Iduna	449	149
M6	Zuntz	B1	W33	Golf	93	Lloyd	199	57

Tab. 2: Unnormalisierte Tabelle einer Autovermietung (Quelle: P. Stahlknecht: Einführung in die Wirtschaftsinformatik)

Zwischen den Mietern (links) und den Mietwagen (rechts) besteht eine N:M-Beziehung, deren Abbildung in einer einzigen Tabelle Redundanzen verursacht, die eine Quelle potentieller Datenanomalien bilden, was man durch Normalisieren vermeiden möchte. (Auf gezieltes Denormalisieren zwecks besserer Performanz einer Datenbank braucht hier nicht eingegangen zu werden.) Bei dem Unterfangen, diesen Verstoß gegen die 1. Normalform (1NF) zu erkennen und durch Auslagerung einer der beiden Wiederholungsgruppen zu beseitigen, versagt der Tabellenanalyseassistent seinen Dienst. Wenn man ihm die Entscheidung überläßt (wozu sollte er sonst gut sein?), dann überrascht er mit

der Empfehlung, die Tabelle nicht aufzuteilen (Abb. 1). Der Verweis des Assistenten auf die manuelle Aufteilbarkeit ist gut gemeint, dürfte jedoch den von der Normalformenlehre unbeleckten Anwender hoffnungslos überfordern.

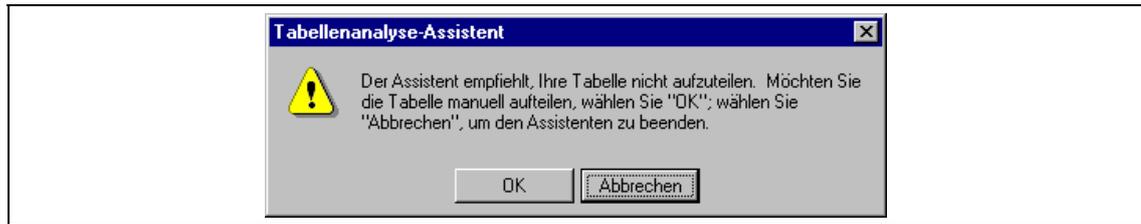


Abb. 1: Irreführende Empfehlung des Tabellenanalyseassistenten

Nach Auslagerung der Mieterattribute {Mieter-ID, Name, Branche} bleibt eine immer noch redundanzhaltige "Resttabelle" {Wagen-ID, Mieter-ID, Fahrzeugtyp, Baujahr, Versicherung, Mietsatz, Mietdauer} übrig. Da die Attribute Fahrzeugtyp, Baujahr, Versicherung und Mietsatz nur von der Wagen-ID und nicht vom gesamten, sondern nur von einem Teil des Primärschlüssels funktional abhängig sind, liegt hier ein Verstoß gegen die 2. Normalform (2NF) vor, der durch Auslagerung dieser Nichtschlüsselattribute in eine eigenständige Tabelle zu heilen ist. Auch bei diesen Normalisierungsschritt läßt der Tabellenanalyseassistent den Anwender mit der aus Abb. 1 bekannten Meldung im Stich, der wiederum auf "Handarbeit" angewiesen ist.

Damit ist die Redundanzbeseitigung aber immer noch nicht ganz am Ende. In der neu entstandenen Tabelle {Wagen-ID, Fahrzeugtyp, Baujahr, Versicherung, Mietsatz} ist das Nichtschlüsselattribut Mietsatz durch das Nichtschlüsselattribut Fahrzeugtyp determiniert, was die 3. Normalform (3NF) verletzt. Wenn es sich hierbei nicht um eine Zufälligkeit der wenigen, vorhandenen Ist-Daten handelt, sondern die Preispolitik korrekt widerspiegelt (business rule), dann muß eine neue Tabelle {Fahrzeugtyp, Mietsatz} gebildet werden. Sie werden es bereits geahnt haben: auch zu einer Empfehlung für die 3NF-Tabellenaufteilung mag sich der Tabellenanalyseassistent nicht aufraffen.

Fazit: Der Tabellenanalyseassistent ist mit der automatischen Erkennung von Verstößen gegen eine der drei üblicherweise einzuhaltenden Normalformen u.U. hoffnungslos überfordert. Für eine manuelle Normalisierung kann er als technisches Hilfsmittel fungieren, obwohl die hierfür notwendigen Auswahl- und Aktionsabfragen dank des vorbildlich einfachen graphischen Abfrageeditors von Access leicht zu erstellen sind. Die Kenntnis der Normalformenlehre kann der Tabellenanalyseassistent aber in keinem Fall ersetzen.

Während uns der Tabellenanalyseassistent bei der Normalisierung bisher komplett im Stich gelassen hat, wären wir bei der Formulierung eines **Entity-Relationship-Modells (ERM)** fast automatisch - sozusagen aus der Logik der Sache selbst - zumindest bei der 2NF angelangt. Die Daten in Tab. 2 legen nämlich folgende Erkenntnisse nahe: es gibt einen Entitätstyp Mieter, der über die assoziative Entität Vermietung in einer N:M-Beziehung zum Entitätstyp Mietwagen steht. Zieht man noch die hier unterstellte Abhängigkeit des Mietpreises vom Fahrzeugtyp in Betracht, dann kommt man sehr schnell zum ERM der Abb. 2, das exakt die vier Tabellen widerspiegelt, die in einem RDBMS wie Access zu implementieren wären.

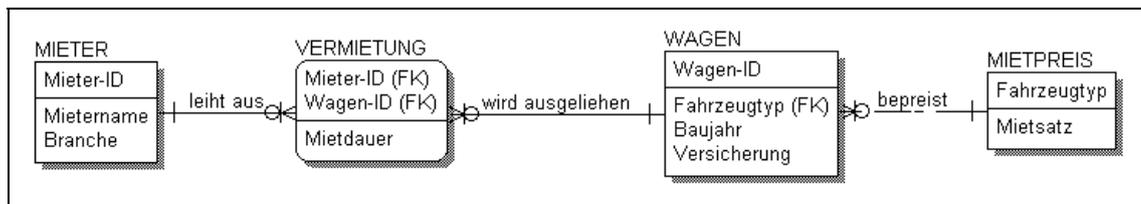


Abb. 2: ERwin-Modell zur Autovermietung BERENT in der 3. Normalform

Die ganze in die Verwendung der Datenmodellierungsassistenten zu steckende und in ihrem Ergebnis dennoch unbefriedigende Arbeit konzentriert man besser auf die Erstellung eines gehaltvollen ERM's mit einem Access-unterstützten CASE-Tool wie z.B. ERwin (Logic Works) oder S-Designer (PowerSoft), das die Entwurfsinformationen zum logischen und physischen Datenmodell gebündelt dokumentiert und beim forward engineering an eine Access-Datenbank weiterreicht. Diese Vorgehensweise bietet gegenüber den Datenmodellierungsassistenten erhebliche Vorteile:

- Das semantisch am Fachkonzept und der Endbenutzerterminologie orientierte ERM ist für die Kommunikation mit den prospektiven Anwendern besser geeignet als das Access-Beziehungsfenster.
- Alle Spezifikationen des logischen und physikalischen Datenmodells werden zentral und leicht zugänglich verwaltet.
- Die Dokumentation ist mit CASE-Tool-Reports flexibler gestaltbar und weiterverwendbar als mit dem Access-Datenbankdokumentierer.
- Ein Wechsel der Datenhaltung von der JET-Engine zu einem der gängigen SQL-Server ist via forward engineering kontrollierter möglich.
- Last not least: der Einsatz eines Datenmodellierungswerkzeugs nötigt dazu, vor dem (Quick-And-Dirty)-Entwurf von Tabellen in Access zunächst einmal gründlich über das Datenmodell als eine der wichtigsten Säulen einer datenbankgestützten Applikation nachzudenken.

Dr. Manfred Sommer, Professor für Wirtschaftsinformatik in Hamburg und Mitgesellschafter der NSL EDV-Training & Consulting GmbH in Bielefeld, Autor der Buches "Datenbankdesign für Access 7.0/2.0"