

Relationale Datenbank zur Verwaltung eines Onlineshops

Verfasser:

Lorenz Schwob
Himmelrainweg 36
4450 Sissach

Referent:

Prof. Andreas Meier

Betreuer:

Darius Zumstein

Freiburg, 28.10.2010

Inhalt

Abbildungsverzeichnis	3
1. Einleitung.....	4
1.1 Problemstellung	4
1.2 Zielsetzung.....	4
1.3 Vorgehensweise	4
2. Datenanalyse	5
3. Das Entitäten-Beziehungsmodell (ERM)	7
3.1 Assoziationstypen.....	9
3.2 Beziehungen	10
3.3 Schema	12
3.4 Relationales Datenbankschema	13
4. Implementierung mit Access 2007	16
4.1 Tabellen	16
4.2 Beziehungen	17
4.3 Abfragen	19
4.4 Aktualisierungsabfragen.....	23
4.5 Formulare	24
4.6 Berichte.....	25
4.7 Makros	27
4.8 Code-Generator	28
4.9 Datensicherheit	29
5. Schlusswort	30
Literaturverzeichnis	31

Abbildungsverzeichnis

Abb.1:	Verwendete Entitätsmengen	Seite 8
Abb.2:	Übersicht über die Mächtigkeit von Beziehungen	Seite 9
Abb.3:	Beziehung zwischen den Entitätsmengen "Mitarbeiter" und "Kundenbestellung" mit Assoziationstypen	Seite 10
Abb.4:	Beziehung zwischen den Entitätsmengen "Kunde" und "Kundenbestellung" mit Assoziationstypen	Seite 10
Abb.5:	Beziehung zwischen den Entitätsmengen "Produkt" und "Kundenbestellung" mit Assoziationstypen	Seite 10
Abb.6:	Beziehung zwischen den Entitätsmengen "Produkt" und "Produktkategorie" mit Assoziationstypen	Seite 10
Abb.7:	Beziehung zwischen den Entitätsmengen "Produkt" und "Lieferantenbestellung" mit Assoziationstypen	Seite 11
Abb.8:	Beziehung zwischen den Entitätsmengen "Lieferant" und "Lieferantenbestellung" mit Assoziationstypen	Seite 11
Abb.9:	Beziehung zwischen den Entitätsmengen "Produkt" und "Lieferant" mit Assoziationstypen	Seite 11
Abb.10:	Beziehung zwischen den Entitätsmengen "Mitarbeiter" und "Lieferant" mit Assoziationstypen	Seite 11
Abb. 11:	Entitäten-Beziehungsmodell	Seite 12
Abb. 12:	Entwurfsansicht der Tabelle Kunden_Bestellungen	Seite 16
Abb. 13:	Beziehungen bearbeiten	Seite 17
Abb. 14:	Beziehungen	Seite 18
Abb. 15:	Entwurfsansicht der Abfrage „Lagerbestand zu klein“	Seite 19
Abb. 16:	SQL-Ansicht der Abfrage „Lagerbestand zu klein“	Seite 20
Abb. 17:	Resultate der Abfrage „Unbezahlte Rechnungen“	Seite 21
Abb. 18:	Resultate der Abfrage „Monatsumsatz“	Seite 21
Abb. 19:	Resultate der Abfrage „Monatsverkäufe/-umsätze“	Seite 22
Abb. 20:	Entwurfsansicht der Aktualisierungsabfrage „Warenausgang“	Seite 23
Abb. 21:	Entwurfsansicht des Formulars „Bestellformular“	Seite 24
Abb. 22:	Berichtsansicht des Berichtes „Rechnung“ basierend auf der gleichnamigen Abfrage	Seite 26
Abb. 23:	Entwurfansicht eines Makros (Automatischer Start der Aktualisierungsabfrage „Warenausgang“ beim Schliessen des Formulars „Bestellformular“)	Seite 27
Abb. 24:	Code-Generator mit Ausdruck, der die SQL-Abfrage des Produktauswahlfeldes durchführt	Seite 29
Abb. 25:	Auswahl der Option „Mit Kennwort verschlüsseln“	Seite 29

1. Einleitung

1.1 Problemstellung

Die Verwaltung eines Onlineshops ist eine anspruchsvolle Aufgabe, welche ohne ein geeignetes Informationssystem kaum zu bewerkstelligen wäre. Je mehr Produkte, Kunden und Lieferanten am gesamten System beteiligt sind, desto unübersichtlicher wird das gesamte Konstrukt. Deshalb ist es notwendig, beim Betreiben eines Onlineshops ein richtig implementiertes und angepasstes Informationssystem zu verwenden, um einen optimalen Betrieb der Verwaltung und Distribution garantieren zu können.

1.2 Zielsetzung

In dieser Arbeit wird aufgezeigt, welche Anforderungen ein Onlineshop an eine Datenbank hat und wie man mit Hilfe von Microsoft Access 2007 eine solche erstellen kann, welche für das Betreiben eines Onlineshops eingesetzt werden könnte. Die Datenbank muss in der Lage sein, sämtliche Mitarbeiter, Lieferanten, Bestellungen und Kunden zu erfassen und deren Verbindungen untereinander zu Verwalten. Das Ziel dieser Arbeit ist es, eine voll funktionsfähige Datenbank zu generieren, die sämtliche Anforderungen erfüllt und sofort für einen Onlineshop benutzt werden könnte.

1.3 Vorgehensweise

Gemäss [Faeskorn-Woyke et al. 2007] existieren bei einer Datenbankentwicklung fünf Phasen. Die Analyse-, die Entwurfs-, die Implementierungs-, die Abnahme-/Einführungs- sowie die Wartungs-/Pflegephase. In dieser Arbeit werden nur die ersten drei Phasen berücksichtigt, da die Datenbank nicht tatsächlich zum Einsatz kommt.

Der erste Teil der Arbeit befasst sich mit dem Kreieren eines sinnvollen Datenbank-Modells. Dies beinhaltet als erstes eine Datenanalyse, in welcher die Anforderungen ersichtlich und die einzelnen Elemente, welche die Datenbank beinhalten soll, zusammengetragen werden. In einem nächsten Schritt wird ein Entitäten-Beziehungsmodell erstellt, welches dann in ein relationales Datenbankschema überführt wird. Im zweiten Teil dieser Arbeit wird aufgezeigt, wie aus diesem Schema eine Datenbank implementiert werden kann und es wird näher auf die Hilfsmittel eingegangen, welche das Programm Access 2007 bereitstellt.

2. Datenanalyse

Um mit der Implementierung einer Datenbank beginnen zu können ist es notwendig, in einem ersten Schritt, der Datenanalyse, die Anforderungen zusammenzutragen. Es stellt sich die Frage, welche Informationen in der Datenbank gespeichert werden und wie man damit arbeiten möchte.

Produkte:

Im Vordergrund stehen die Produkte, welche der Onlineshop verkaufen möchte. Will ein Kunde ein Produkt bestellen, interessiert er sich in erster Linie für das Aussehen und den Preis. Für die Mitarbeiter des Onlineshops ist es wichtig zu wissen, wie viele Einheiten eines Produkts sich noch an Lager befinden, ab welchem Lagerbestand man diese nachzubestellen hat und bei welchem Lieferanten es sich bei Bedarf zu welchem Preis anfordern lässt. Um die Produkte leicht und eindeutig voneinander unterscheiden zu können ist es ratsam, Produktnummern einzuführen. Die Attribute, welche in der Tabelle gespeichert werden sollten sind: Produktnummer, Produktname, Produktbeschreibung, Bild des Produkts, Kategoriennummer, Lieferantennummer, Einkaufspreis, Verkaufspreis, Menge an Lager, Mindestbestand.

Kategorien:

Um die Übersicht (sowohl seitens der Kunden als auch der Mitarbeiter) zu wahren ist es ratsam, die Produkte in verschiedene Kategorien zu unterteilen, um ähnliche Produkte zusammenzufassen. Diese werden in einer eigenen Tabelle festgehalten und ebenfalls mit einer Nummer versehen. Eine kurze Beschreibung erleichtert die Einteilung der Produkte. Folgende Attribute sollten also über eine Kategorie gespeichert werden: Kategoriennummer, Kategoriename, Kategorienbeschreibung.

Kunden:

Die Tabelle, welche die Daten der Kunden enthält ist wichtig für einen funktionierenden Onlineshop. Entschliesst sich ein Kunde, etwas zu bestellen, so muss er, falls er dies zum ersten Mal tut, eine Auswahl an Daten angeben, welche gespeichert werden. Zu Marktforschungszwecken könnte man hier viele Daten über den Kunden speichern. In der Beispieldatenbank sind dies jedoch lediglich die Angaben, welche für eine Lieferung notwendig sind: Kundennummer, Anrede, Name, Vorname, Adresse, PLZ, Wohnort, Telefonnummer, Email-Adresse.

Mitarbeiter:

Damit ein Geschäft funktioniert benötigt es Mitarbeiter, die die Bestellungen ausführen, welche die Kunden eingeben. Diese werden am besten ebenfalls in der Datenbank erfasst, um so die einzelnen Bestellungen auf die verschiedenen Mitarbeiter aufteilen zu können. Da die Datenbank komplett sein sollte und es nicht nur Mitarbeiter im Lager gibt, sollte auch ihre Funktion angegeben werden. Beispielsweise gibt es ProduktmanagerInnen, die die Auswahl der Produkte, die vertrieben werden, treffen oder Personen

die eigens dafür angestellt sind, Lieferantenbeziehungen aufzubauen und zu pflegen. Die Attribute, welche über einen Mitarbeiter gespeichert werden sollten sind: Mitarbeiternummer, Funktion, Anrede, Name, Vorname, Adresse, PLZ, Wohnort, Telefonnummer, Email-Adresse.

Lieferanten:

Um die Produkte überhaupt vertreiben zu können, ist ein Onlineshop auf Lieferanten angewiesen, welche die Waren, die er verkauft, überhaupt erst verfügbar machen. Auch diese sollten in einer Datenbank festgehalten werden, um die verschiedenen Lieferantenbeziehungen ständig im Überblick behalten zu können. Es sollte auch vermerkt werden, welcher Mitarbeiter sich um welche Lieferantenbeziehungen kümmert. Dies wird mit Hilfe der Mitarbeiternummer angezeigt. Sie weisen nahezu identische Attribute wie die Mitarbeiter auf: Lieferantenummer, Lieferantename, Adresse, PLZ, Ort, Telefonnummer, Email-Adresse, Mitarbeiternummer.

Kundenbestellungen:

Die einzelnen Bestellungen der jeweiligen Kunden werden ebenfalls in einer Tabelle abgelegt. Der Kunde gibt die verschiedenen Produkte sowie die Menge an, welche er bestellen möchte. Je nach Auslastung eines Onlineshops ist es ebenfalls ratsam, den eingehenden Bestellungen sofort einen Mitarbeiter zuzuweisen, welcher die Verantwortung für deren Erfüllung trägt. Ebenfalls sollte der aktuelle Status der Bestellung festgehalten werden. Die gesamte Bestellung wird dann gespeichert und zwar mit folgenden Attributen: Auftragsnummer, Kundennummer, Produktmenge, Produktnummer, Auftragsdatum, Mitarbeiternummer, Status.

Lieferantenbestellungen:

Zur Verwaltung des Onlineshops sollten auch die Bestellungen, welche ausgehend an die Lieferanten gesendet werden, in einer Tabelle festgehalten werden. Diese folgen wiederum dem gleichen Schema wie die eingehenden Aufträge, mit Ausnahme der Angabe des Mitarbeiters, welcher die Bestellung aufgegeben hat: Auftragsnummer, Lieferantenummer, Produktmenge, Produktnummer, Auftragsdatum, Mitarbeiternummer, Status.

Anforderungen:

Die Mitarbeiter des Onlineshops müssen mit der Datenbank arbeiten können. Versteht man den Aufbau einer Datenbank nicht, so ist es schwierig ohne voreingestellte Formulare damit zu arbeiten. Deshalb müssen die Anforderungen, welche die Mitarbeiter an die Datenbank haben, durch das Erstellen der einzelnen Lösungen erfüllt werden.

Folgende Listen sollten gedruckt oder angesehen werden können:

- Kundenliste
- Produktliste
- Mitarbeiterliste
- Lieferantenliste
- Liste der Bestellungen
- Liste der nachzubestellenden Produkte

Die Mitarbeiter sollten die Möglichkeit haben, mit der Datenbank Mitarbeiter-, Kunden-, Produkt-, und Lieferantenprofile neu zu erstellen oder bestehende zu bearbeiten. Desweiteren müssen die (Nach-)Bestellungen erfasst und ihre Status im Nachhinein verändert werden können. Zusätzlich wäre es wünschenswert, wenn die entsprechenden Rechnungen oder Mahnungen automatisch generiert werden, damit die Mitarbeiter keine Zeit verlieren, indem sie diese von Hand erstellen müssen.

Diese Datenbank ist nur für die Verwendung durch die Mitarbeiter des Onlineshops vorgesehen. Die Kunden haben keinen Zugriff darauf. Wird eine Bestellung aufgegeben, so erhalten die Mitarbeiter diese und müssen sie danach manuell in die Datenbank eingeben. In Realität werden die Daten automatisch eingefügt. Solche Abläufe zu programmieren würde jedoch den Rahmen dieser Seminararbeit sprengen.

3. Das Entitäten-Beziehungsmodell (ERM)

Nach dem Zusammentragen dieser Daten wird in einem nächsten Schritt das Entitäten-Beziehungsmodell geschaffen. In diesem Modell werden Entitäts- und Beziehungsmengen angegeben.

Eine Entität ist ein Objekt, welches von anderen gut zu unterscheiden ist. Beispielsweise ist ein Mitarbeiter eine Entität, eine Kundenbestellung ist wiederum eine Entität eines anderen Typs. Entitäten des gleichen Typs werden in Entitätsmengen zusammengefasst, welche im Entitäten-Beziehungsmodell durch ein Rechteck dargestellt werden. Zudem werden die Entitätsmengen durch Attribute charakterisiert die hier bereits in der Datenanalyse zusammengetragen wurden. So wird zum Beispiel ein Kunde mit einer Nummer versehen und neben seinem Namen werden Merkmale wie Adresse, Wohnort, etc. festgehalten. Für eine Entitätsmenge muss ebenfalls ein Identifikationsschlüssel festgelegt werden, welcher eine einzelne Entität in einer Entitätsmenge eindeutig identifiziert. Zu diesem Zweck wird oftmals ein künstlicher Schlüssel verwendet, indem die einzelnen Entitäten mit einer Nummer versehen werden. So können beispielsweise auch Kunden, die dieselben Vor- und Nachnamen besitzen eindeutig voneinander unterschieden werden.

In dieser Datenbank werden sieben Entitätsmengen verwendet:

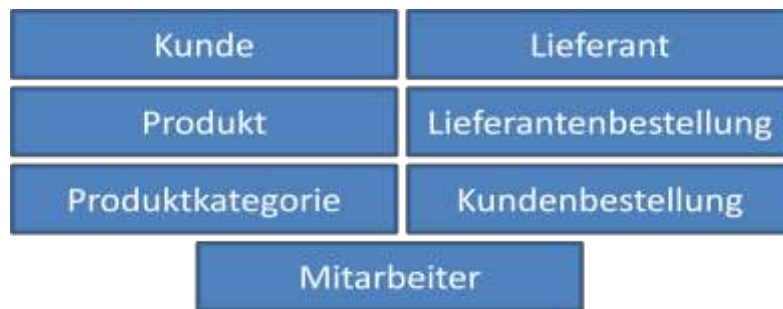


Abb. 1: Verwendete Entitätsmengen

Zusätzlich zu den Entitätsmengen sind auch ihre Beziehungen zueinander von grosser Bedeutung. Diese bilden Beziehungsmengen, welche im Entitäten-Beziehungsmodell durch einen Rhombus dargestellt werden. Wie bei den Entitätsmengen ist es möglich, Beziehungsmengen mit Attributen zu versehen.

Durch dieses Modell kann man überprüfen, ob sich die in der Datenanalyse gesammelten Anforderungen überhaupt durch ein relationales Datenbankschema verbinden lassen und die geplante Datenbank machbar ist. Mit Hilfe des Entitäten-Beziehungsmodell lassen sich so die in der Datenanalyse gesammelten Fakten übersichtlich strukturieren und darstellen.

3.1 Assoziationstypen




Eine Assoziation ist die Bedeutung der Beziehung in eine Richtung. Der Assoziationstyp von einer Entitätsmenge zu einer anderen zeigt auf, wie viele Entitäten aus der ersten Entitätsmenge einer bestimmten Entität aus der zweiten Entitätsmenge zugeordnet werden können. Sie zeigen die Mächtigkeit einer Beziehung auf. Es existieren nach [Meier 2010 S.23] folgende Assoziationstypen:

- **Einfache Assoziation (1):** Jeder Entität aus der Entitätsmenge EM_1 ist „*genau eine*“ Entität aus der EM_2 zugeordnet.
- **Konditionelle Assoziation (c):** Jeder Entität aus der EM_1 ist „*keine oder eine*“ (höchstens eine) Entität aus der EM_2 zugeordnet.
- **Mehrfache Assoziation (m):** Jeder Entität aus der EM_1 sind „*eine oder mehrere*“ (mindestens eine) Entität aus der EM_2 zugeordnet.
- **Mehrfach-konditionelle Assoziation (mc):** Jeder Entität aus der EM_1 sind „*keine, eine oder mehrere*“ Entität aus der EM_2 zugeordnet.

Die Beziehungen laufen immer in beide Richtungen, daraus folgt, dass eine Beziehung immer zwei Assoziationstypen umfasst. Betrachtet man diese beiden Assoziationstypen lässt sich daraus die Mächtigkeit einer Beziehung ermitteln.

	1	c	M	Mc
1	(1,1)	(1,c)	(1,m)	(1,mc)
c	(c,1)	(c,c)	(c,m)	(c,mc)
m	(m,1)	(m,c)	(m,m)	(m,mc)
mc	(mc,1)	(mc,c)	(mc,m)	(mc,mc)

Abb. 2: Übersicht über die Mächtigkeiten von Beziehungen (nach [Meier 2010 S.31])

-  = einfach-einfache Beziehungen
-  = einfach-komplexe Beziehungen
-  = komplex-komplexe Beziehungen

3.2 Beziehungen

Auf Basis der eben erläuterten Regeln für Beziehungen und Assoziationstypen erstellt man nun als ersten Schritt die Beziehungen zwischen den Entitätsmengen und versteht diese mit den passenden Assoziationstypen.

Für diese Datenbank können folgende Beziehungen betrachtet werden:



Abb. 3: Beziehung zwischen den Entitätsmengen "Mitarbeiter" und "Kundenbestellung" mit Assoziationstypen.

Eine Bestellung wird von genau einem Mitarbeiter bearbeitet, während ein Mitarbeiter keine, eine oder mehrere Bestellungen bearbeiten kann.



Abb. 4: Beziehung zwischen den Entitätsmengen "Kunde" und "Kundenbestellung" mit Assoziationstypen.

Ein Kunde kann keine, eine oder mehrere Bestellungen aufgeben, während eine Bestellung selbst nur von genau einem Kunden in Auftrag gegeben werden kann.



Abb. 5: Beziehung zwischen den Entitätsmengen "Produkt" und "Kundenbestellung" mit Assoziationstypen.

Ein Produkt kann in keiner, einer oder mehreren Bestellungen vorkommen. Eine Bestellung beinhaltet ein oder mehr Produkte.



Abb. 6: Beziehung zwischen den Entitätsmengen "Produkt" und "Produktkategorie" mit Assoziationstypen.

Ein Produkt gehört genau einer Kategorie an, während eine Kategorie ein oder mehr Produkte beinhaltet.



Abb. 7: Beziehung zwischen den Entitätsmengen "Produkt" und "Lieferantenbestellung" mit Assoziationstypen.

Ein Produkt kann in einer oder mehreren Lieferantenbestellungen vorkommen, während eine Lieferantenbestellung ein oder mehrere Produkte beinhalten kann.



Abb. 8: Beziehung zwischen den Entitätsmengen "Lieferant" und "Lieferantenbestellung" mit Assoziationstypen.

Ein Lieferant kann keine, eine oder mehrere Bestellungen erhalten, während sich eine Bestellung an genau einen Lieferanten richtet.



Abb.9: Beziehung zwischen den Entitätsmengen "Produkt" und "Lieferant" mit Assoziationstypen.

Ein Produkt wird bei genau einem Lieferanten bezogen. Ein Lieferant kann jedoch ein oder mehr Produkte anbieten.



Abb. 10: Beziehung zwischen den Entitätsmengen "Mitarbeiter" und "Lieferant" mit Assoziationstypen.

Ein Mitarbeiter betreut keinen, einen oder mehrere Lieferanten, während ein Lieferant von genau einem Mitarbeiter betreut wird.

3.3 Schema

Nach der Erstellung sämtlicher Beziehungen werden die einzelnen Entitätsmengen und ihre Beziehungen zum Entitäten-Beziehungsmodell zusammengeführt:

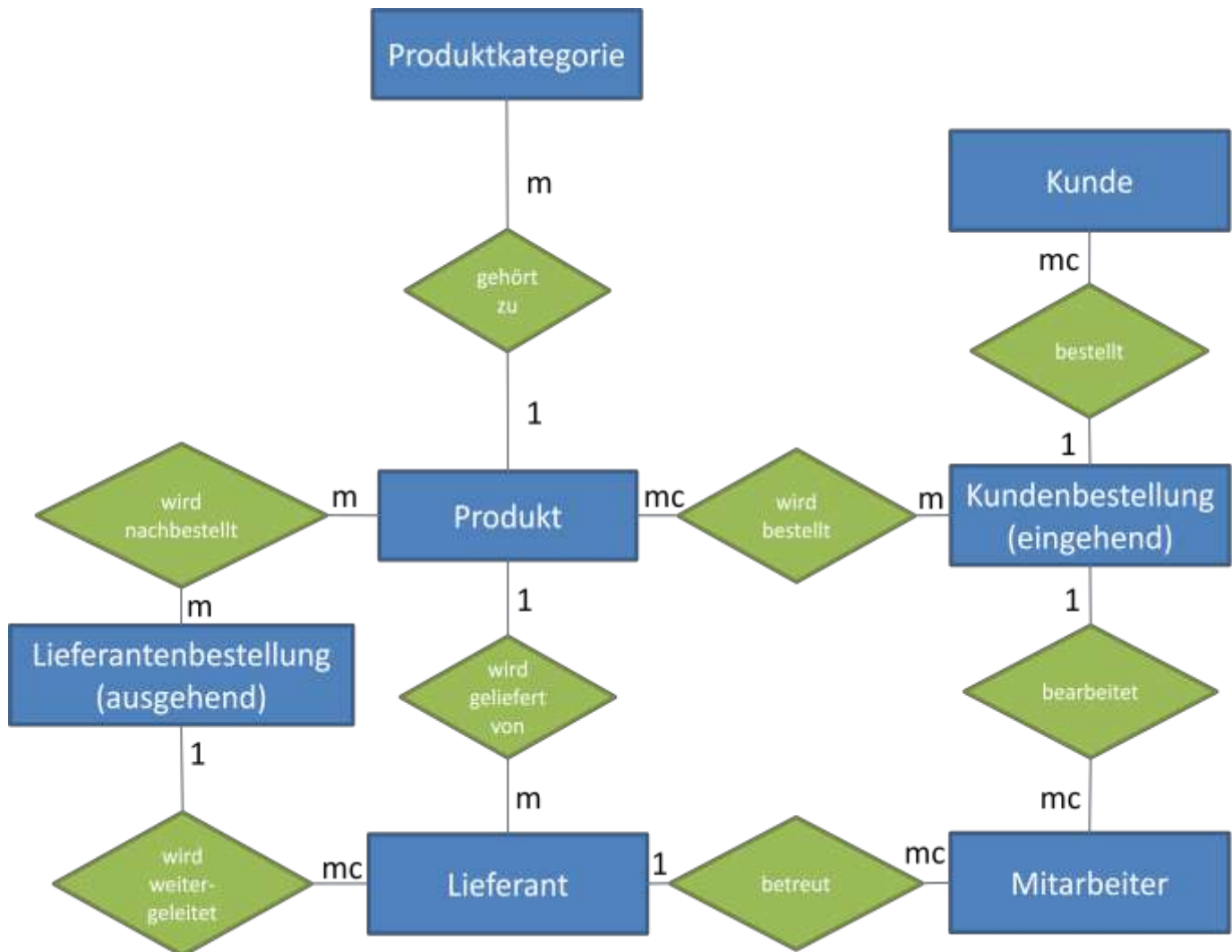


Abb.11: Entitäten-Beziehungsmodell

3.4 Relationales Datenbankschema

Mit Hilfe des Entitäten-Beziehungsmodells kann nun im nächsten Schritt das relationale Datenbankschema erstellt werden. Dabei sind nach [Meier 2010 S.30-34] gewisse Regeln einzuhalten, die sogenannten Abbildungsregeln.

Regel 1 (Entitätsmengen): Jede Entitätsmenge *muss* als eigenständige Tabelle mit einem eigenständigen Primärschlüssel definiert werden. Als Primärschlüssel der Tabelle dient entweder der entsprechende Schlüssel der Entitätsmenge oder ein Schlüsselkandidat. Die übrigen Merkmale der Entitätsmengen gehen in die entsprechenden Attribute der Tabellen über.

Regel 2 (Beziehungsmengen): Jede Beziehungsmenge *kann* als eigenständige Tabelle definiert werden, wobei die Identifikationsschlüssel der zugehörigen Entitätsmengen als Fremdschlüssel in dieser Tabelle auftreten müssen. Der Primärschlüssel kann dabei aus den Fremdschlüsseln zusammengesetzt sein oder es wird ein künstlicher Schlüssel erstellt.

Regel 3 (komplex-komplexe Beziehungen): Jede komplex-komplexe Beziehung *muss* als eigenständige Tabelle angegeben werden. Der Primärschlüssel der Beziehungsmengentabelle kann dabei aus den Fremdschlüsseln zusammengesetzt sein oder ein künstlicher Schlüssel wird erstellt.

Regel 4 (einfach-komplexe Beziehungen): Eine einfach-komplexe Beziehung kann ohne eine eigene Beziehungsmengentabelle durch die beiden Tabellen der zugehörigen Entitätsmengen ausgedrückt werden. Dazu wird in der Tabelle mit der einfachen Assoziation (1 oder c) ein Fremdschlüssel auf die referenzierte Tabelle mit eventuell weiteren Merkmalen der Beziehungsmenge geführt.

Regel 5 (einfach-einfache Beziehungen): Eine einfach-einfache Beziehungsmenge kann ohne eigenständige Tabelle durch die beiden Tabellen der zugehörigen Entitätsmengen ausgedrückt werden, indem einer der Identifikationsschlüssel der referenzierten Tabelle als Fremdschlüssel in die andere Tabelle eingebracht wird.

Die Umsetzung unseres Entitäten-Beziehungsmodells benötigt also folgende Tabellen:

Tabelle 1: Produkt

Produkt									
Produkt_ID	PName	PBeschreibung	Bild	Kategorie_ID	Lieferant_ID	Einkaufspreis	Verkaufspreis	Menge an Lager	Mindestbestand

Der Primärschlüssel Produkt_ID identifiziert jedes Produkt eindeutig, da jedem neu erfassten Artikel sofort eine Nummer zugewiesen wird. Die Fremdschlüssel Kategorie_ID und Lieferant_ID zeigen die jeweilige Zuordnung welcher Kategorie das Produkt zuzuordnen ist und bei welchem Lieferanten es bezogen werden kann.

Tabelle 2: Mitarbeiter

Mitarbeiter

Mitarbeiter_ID	Funktion	Anrede	Name	Vorname	Adresse	PLZ	Wohnort	Telefon	Email

Auch den Mitarbeitern wird jeweils eine Nummer zugeteilt, sobald sie eingestellt bzw. in der Tabelle erfasst werden, um sie eindeutig zu identifizieren. Die restlichen Attribute enthalten die jeweiligen Informationen über die Mitarbeiter. Dieselben Informationen werden auch in den zwei folgenden Tabellen „Lieferant“ und „Kunde“ gespeichert, nur das den Lieferanten jeweils zusätzlich ein Mitarbeiter, der sich um die Beziehungen kümmert, zugeordnet wird.

Tabelle 3: Lieferant

Lieferant

Lieferant_ID	Lieferant	Adresse	PLZ	Ort	Telefon	Email	Mitarbeiter_ID

Tabelle 4: Kunde

Kunde

Kunde_ID	Anrede	Name	Vorname	Adresse	PLZ	Wohnort	Telefon	Email

Tabelle 5: Produktkategorie

Produktkategorie

Kategorie_ID	Kategorie	KBeschreibung

Der Primärschlüssel Kategorie_ID weist jeder Kategorie eine Nummer zu. Die Kategorien wären zwar auch über ihren Namen eindeutig zu identifizieren, da ein Primärschlüssel jedoch möglichst kurz sein sollte empfiehlt es sich, auch hier einen künstlichen zu verwenden. Die Beschreibung der Kategorien sollte erstellt werden um den Mitarbeitern die Zuteilung neuer Produkte in die Kategorien zu vereinfachen.

Tabelle 6: Kundenbestellung

Kundenbestellung

Auftrag_ID	Kunde_ID	Auftragsdatum	Mitarbeiter_ID	Ist erledigt	Bestand aktualisiert	Ist bezahlt

Die Bestellungen der Kunden erhalten ebenfalls eine automatisch erstellte Nummer, um sie voneinander unterscheiden zu können. Der Fremdschlüssel Kunde_ID zeigt die Nummer des bestellenden Kunden und der Fremdschlüssel Mitarbeiter_ID gibt an, welcher Mitarbeiter sich um die Ausführung des Auftrags kümmert. Die Attribute „Ist Erledigt“ und „Ist Bezahlt“ sind jeweils Ja/Nein-Formatiert und dienen der Bewahrung der Übersicht, welche Bestellungen wie weit fortgeschritten sind. Das Attribut „Bestand aktualisiert“ ist ebenfalls im Ja/Nein-Format vorhanden und dient auch der Übersicht, jedoch nur für die Datenbank und nicht für den Benutzer. Näheres dazu findet sich im Kapitel „Aktualisierungsabfragen“.

Tabelle 7: Lieferantenbestellung

Lieferantenbestellung

LAuftrag_ID	Lieferant_ID	Auftragsdatum	Mitarbeiter_ID	Ist erledigt	Bestand aktualisiert	Ist bezahlt

Die ausgehenden Bestellungen werden gleich verwaltet, mit dem Unterschied, dass der Fremdschlüssel Lieferant_ID verwendet wird, um den mit der Bestellung beauftragten Lieferanten aufzuzeigen.

Tabelle 8: Wird bestellt

Wird bestellt

Auftrag_ID	Produkt_ID	Anzahl

Tabelle 9: Wird nachbestellt

Wird nachbestellt

LAuftrag_ID	Produkt_ID	Anzahl

Die Beziehungsmengen „wird bestellt“ und „wird nachbestellt“ müssen als eigenständige Tabellen erstellt werden, da sie komplex-komplexe Beziehungen sind. Sie werden jeweils mit dem Attribut „Anzahl“ ergänzt, welches anzeigt, in welcher Menge jedes einzelne Produkt eines Auftrages bestellt wurde.

4. Implementierung mit Access 2007

Im letzten Kapitel wurde das relationale Datenbankschema erstellt, welches Software-unabhängig ist. Nun kann mit der Implementierung begonnen werden. Die Datenbank dieser Seminararbeit wird mit Microsoft Access 2007 erstellt.

4.1 Tabellen

Als erstes müssen die Tabellen, welche oben beschrieben sind 1:1 übernommen werden. Hat man das Programm gestartet, muss zuerst eine neue Datenbank angelegt werden, welche zu Beginn komplett leer ist. Durch Auswählen von „Erstellen → Tabelle“ wird eine neue Tabelle angelegt, in welche wir nun unsere Attribute einfügen können.

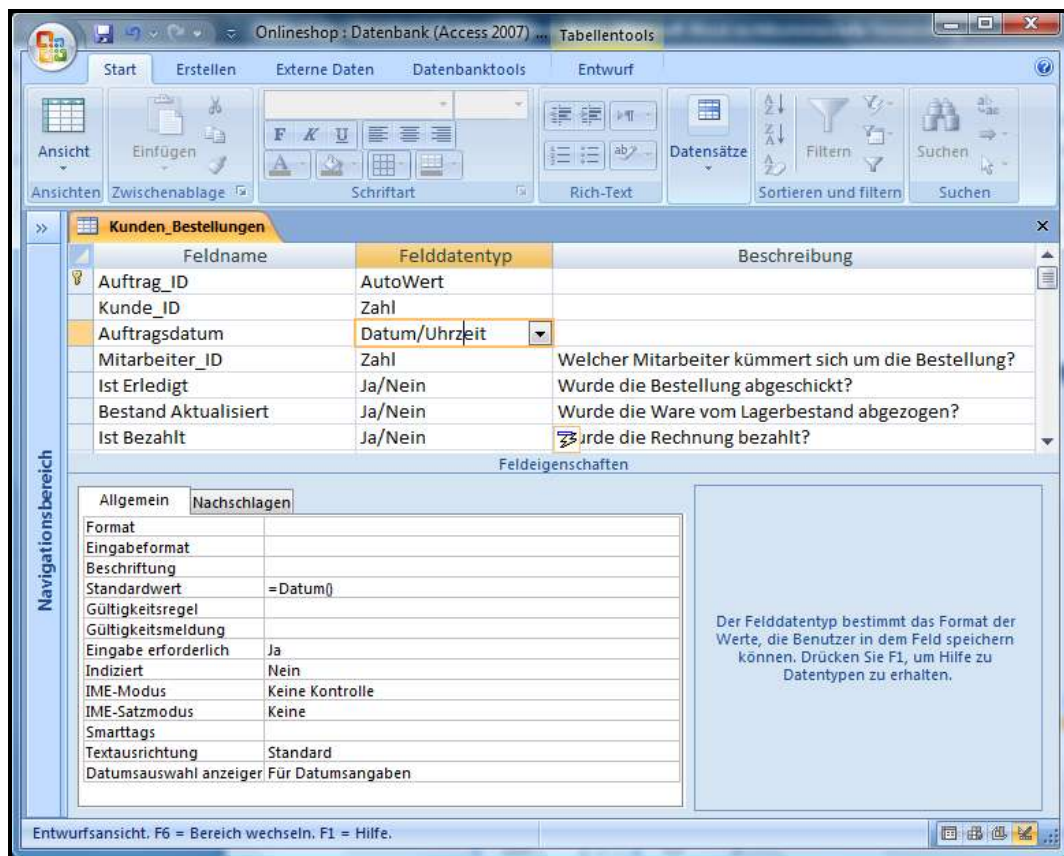


Abb. 12: Entwurfsansicht der Tabelle Kunden_Bestellungen.

Die verschiedenen Attribute werden als Feldname eingegeben. Das Attribut „Auftrag_ID“ ist hier als Primärschlüssel definiert, wie der kleine Schlüssel vor dem Feldnamen anzeigt. Pro Tabelle existiert nur ein Primärschlüssel, dieser könnte jedoch auch aus mehreren Attributen zusammengesetzt werden. Wurde ein Attribut eingegeben, so muss noch der Felddatentyp festgelegt werden. Der Typ „AutoWert“ eignet sich für Primärschlüssel. Bei jedem neuen Datensatz, welches man eingibt, wird automatisch die nächstgrössere Zahl eingefügt. So ist sichergestellt, dass keine Nummer doppelt vorkommt und die Eindeutigkeit weiter gewährleistet ist. Weitere Typen sind: „Text“, „Zahl“ für normale Zahlen, „Währung“ für Währungsangaben, „OLE-Objekt“ für Bilder, Videos, Musik, etc., „Memo“

für lange Beschreibungen, „Datum/Uhrzeit“, „Ja/Nein“ für Wahrheitsangaben und „Hyperlink“ um auf Internetadressen zu verweisen.

Es können Standardwerte eingestellt werden wie auf Abb. 12 der Befehl „=Datum()“, welcher das aktuelle Datum einfügt, sobald ein neuer Datensatz erstellt wird. Ebenfalls möglich sind Gültigkeitsregeln, so wird beispielsweise beim Attribut „Anrede“ in den Tabellen *Mitarbeiter* und *Kunde* nur Herr oder Frau als Feldinhalt geduldet alle anderen Eingaben werden mit einer Fehlermeldung abgelehnt, deren Text im Feld *Gültigkeitsmeldung* eingegeben werden kann.

4.2 Beziehungen

Hat man sämtliche Tabellen nach diesem Schema erstellt, müssen die Beziehungen zwischen ihnen definiert werden. Hierfür muss die Funktion „Datenbanktools → Beziehungen“ angewählt werden. Mit einem weiteren Klick auf „Tabelle anzeigen“ können die Tabellen ausgewählt werden, welche man in das Beziehungsgeflecht einbeziehen möchte. Es können nur Primärschlüssel einer Tabelle mit Fremdschlüsseln anderer Tabellen verknüpft werden.



Abb. 13: Beziehungen bearbeiten

Im Fenster „Beziehungen bearbeiten“ können nun die Einzelheiten der Beziehung eingestellt werden. Durch Auswahl von „mit referentieller Integrität“ wird gewährleistet, dass ein entsprechender Fremdschlüssel nur Werte annehmen kann, die bereits im zugeordneten Primärschlüssel vorhanden sind. So kann z. B. in der Tabelle *Produkt* keine Kategorie eingegeben werden, die nicht bereits in der Tabelle *Produktkategorie* festgelegt worden ist. Wurde „Aktualisierungsweitergabe an verwandte Felder“ aktiv, so müssen Änderungen nicht in allen Tabellen nachgetragen werden, da die Änderung in einer Tabelle auf alle anderen übertragen wird. „Löschweitergabe an verwandte Datensätze“ wird aktiviert um beispielsweise beim löschen eines Lieferanten auch dessen Produkte aus der Datenbank zu streichen. Diese Option ist heikel. Möchte man ein Bestellungsarchiv anle-

gen, so sollte man es vermeiden, die Löschoption zu aktivieren, denn in diesem Falle würden auch die Bestellungen, welche die jeweiligen Produkte enthalten gelöscht oder sie würden gelöscht, sobald der Kunde, welcher sie in Auftrag gegeben hat, aus dem System gestrichen wird.

Sind alle Beziehungen definiert, sollte das Bild das sich ergibt dem Entitäten-Beziehungsmodell entsprechen. Hier zeigt sich auch die Bedeutung der Fremdschlüssel, über die sich die verschiedenen Tabellen miteinander in Beziehung setzen lassen.

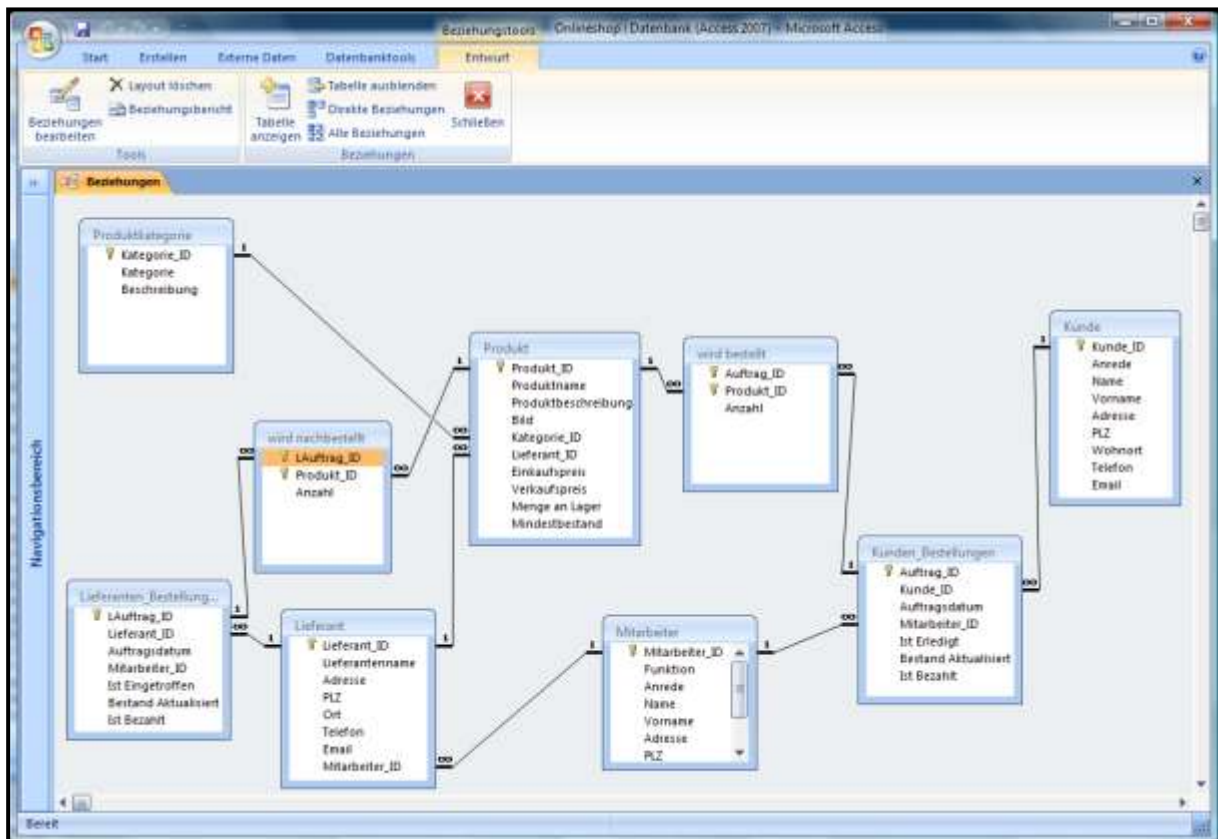


Abb. 14: Beziehungen

4.3 Abfragen

Abfragen sind die Schnittstellen zwischen den Daten, welche in den Tabellen gespeichert sind, und den entsprechenden Formularen, welche die Daten anzeigen und deren Bearbeitung zulassen. Sie haben eine Art Filterfunktion um gewisse Attribute nicht anzeigen zu müssen und können gleichzeitig mit Berechnungen ausgestattet werden, welche nicht in der Tabelle erfasst werden. So ist es z.B. möglich, den Gesamtbetrag einer Bestellung innerhalb einer Abfrage berechnen zu lassen und für den Mitarbeiter nicht relevante Informationen wie die Produktkategorie herauszufiltern. Abfragen können einmal erstellt und dann gespeichert werden. So können sie auch dann wieder benutzt werden, wenn sich die Daten innerhalb der Datenbank verändert haben.

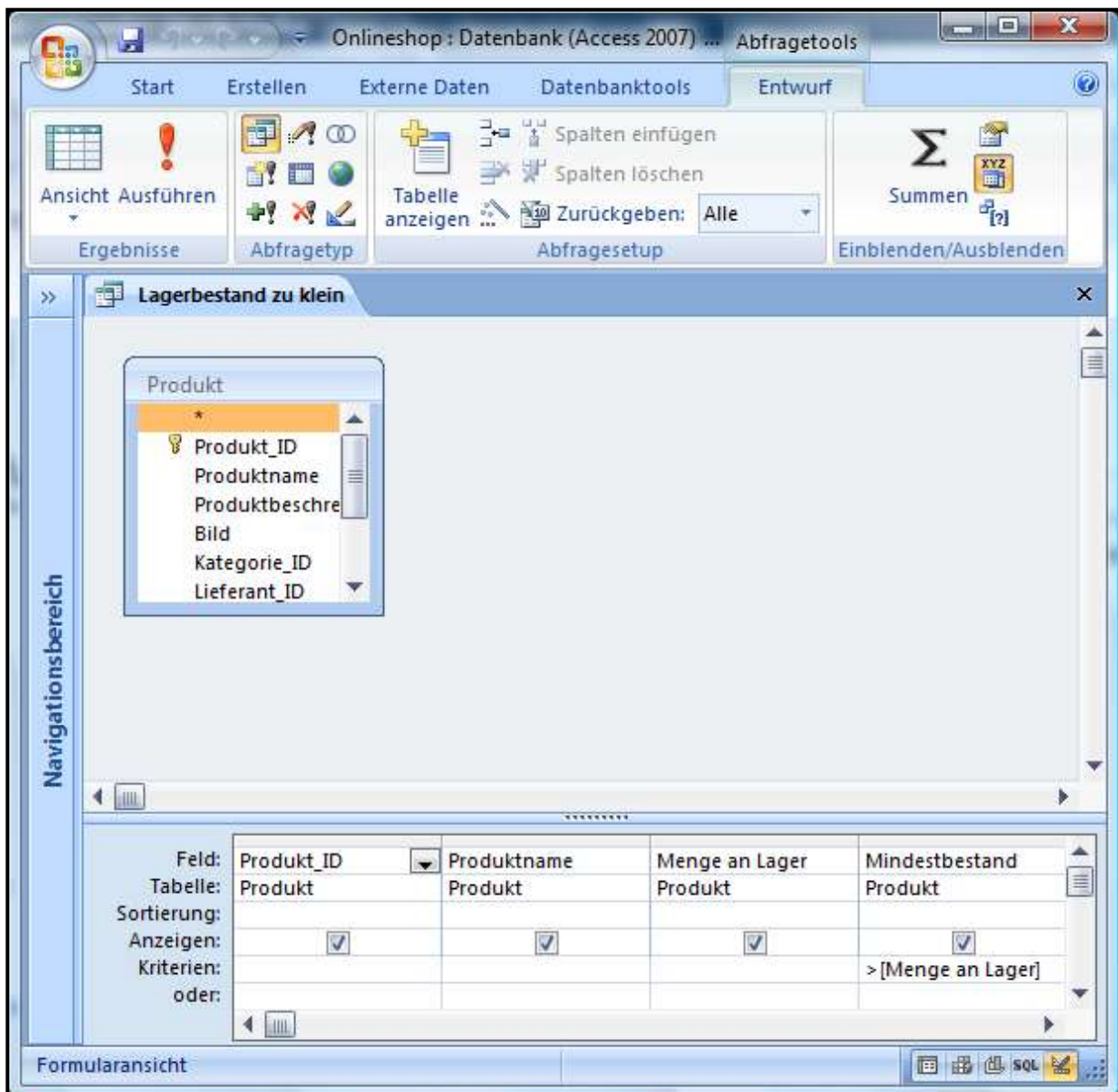


Abb. 15: Entwurfsansicht der Abfrage „Lagerbestand zu klein“

Abfragen können entweder in der Entwurfsansicht oder aber mittels SQL-Eingabe erstellt werden. Für simple Abfragen stellt Access 2007 einen Assistenten zur Verfügung, welcher den grössten Teil der Arbeit übernimmt. Über „Erstellen → Abfrageentwurf“ bzw. über den Abfrageassistenten wird eine Abfrage kreiert. Die Entwurfsansicht lässt sich in die SQL-Ansicht umstellen, welche erfahrenen Programmierern die Direkteingabe ermöglicht.



Abb. 16: SQL-Ansicht der Abfrage „Lagerbestand zu klein“

Um sämtliche Bedürfnisse zu befriedigen werden viele Abfragen benötigt. Die Folgenden drei Beispielabfragen sind in SQL geschrieben und dienen der Veranschaulichung, welche Art von Abfragen in der Datenbank zum Einsatz kommen :

Unbezahlte Rechnungen:

```
SELECT Kunden_Bestellungen.Auftrag_ID,
Kunden_Bestellungen.Auftragsdatum, Kunde.Kunde_ID, Kunde.Name,
Kunde.Vorname, Kunden_Bestellungen.[Ist Erledigt],
Kunden_Bestellungen.[Ist Beahlt],
Sum([Produkt].[Verkaufspreis]*[wird bestellt].[Anzahl]) AS Preis

FROM Produkt INNER JOIN ((Kunde INNER JOIN Kunden_Bestellungen ON
Kunde.Kunde_ID = Kunden_Bestellungen.Kunde_ID) INNER JOIN [wird
bestellt] ON Kunden_Bestellungen.Auftrag_ID = [wird
bestellt].Auftrag_ID) ON Produkt.Produkt_ID = [wird
bestellt].Produkt_ID

GROUP BY Kunden_Bestellungen.Auftrag_ID,
Kunden_Bestellungen.Auftragsdatum, Kunde.Kunde_ID, Kunde.Name,
Kunde.Vorname, Kunden_Bestellungen.[Ist Erledigt],
Kunden_Bestellungen.[Ist Beahlt]

HAVING (((Kunden_Bestellungen.[Ist Erledigt])=True) AND
((Kunden_Bestellungen.[Ist Beahlt])=False));
```

Diese Abfrage zeigt sämtliche Aufträge, deren Rechnungen noch nicht bezahlt wurden.

Auftrag_ID	Auftragsdatum	Kunde_ID	Name	Vorname	Ist Erledigt	Ist Beahlt	Preis
2	09.07.2010	5	Linkert	Simone	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Fr. 90.70
3	11.08.2010	4	Linkert	Horst	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Fr. 81.40

Abb 17: Resultate der Abfrage „Unbezahlte Rechnungen“

Monatsumsätze:

```

SELECT DISTINCTROW
Format$([Kunden_Bestellungen].[Auftragsdatum], 'mmm yy') AS Monat,
Sum([wird bestellt].[Anzahl]*[Produkt].[Verkaufspreis]) AS
Monatumsatz

FROM Produkt INNER JOIN (Kunden_Bestellungen INNER JOIN [wird
bestellt] ON Kunden_Bestellungen.Auftrag_ID = [wird
bestellt].Auftrag_ID) ON Produkt.Produkt_ID = [wird
bestellt].Produkt_ID

GROUP BY Format$([Kunden_Bestellungen].[Auftragsdatum], 'mmm yy'),
Year([Kunden_Bestellungen].[Auftragsdatum])*12+DatePart('m', [Kunden_
Bestellungen].[Auftragsdatum])-1

ORDER BY
Year([Kunden_Bestellungen].[Auftragsdatum])*12+DatePart('m', [Kunden_
Bestellungen].[Auftragsdatum])-1;

```

Die Abfrage addiert sämtliche Einnahmen, welche innerhalb eines Monats generiert wurden zu einer Gesamtsumme und sortiert diese nach Datum.

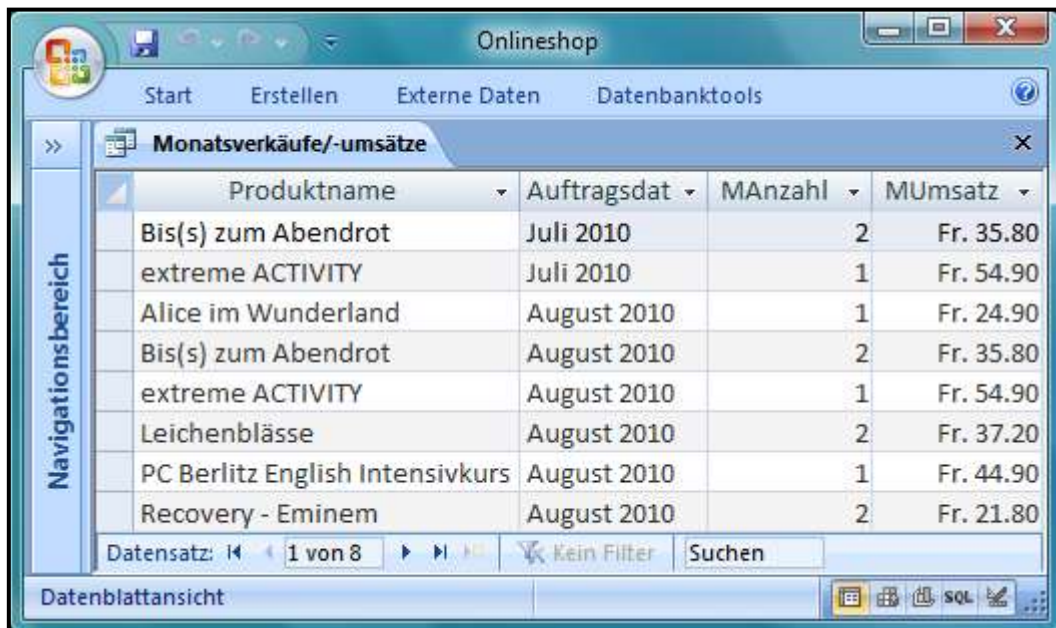
Monat	Monatumsatz
Juli 2010	Fr. 90.70
August 2010	Fr. 219.50

Abb. 18: Resultate der Abfrage „Monatsumsatz“

Monatsverkäufe/-umsätze je Produkt:

```
SELECT DISTINCTROW Produkt.Produktname,  
Format$([Kunden_Bestellungen].[Auftragsdatum], 'mmm yy') AS  
Auftragsdatum, Sum([wird bestellt].Anzahl) AS MAnzahl, Sum([wird  
bestellt].Anzahl*Produkt.Verkaufspreis) AS MUMsatz  
  
FROM Produkt INNER JOIN (Kunden_Bestellungen INNER JOIN [wird  
bestellt] ON Kunden_Bestellungen.Auftrag_ID=[wird  
bestellt].Auftrag_ID) ON Produkt.Produkt_ID=[wird  
bestellt].Produkt_ID  
  
GROUP BY Produkt.Produktname,  
Format$([Kunden_Bestellungen].[Auftragsdatum], 'mmm yy'),  
Year([Kunden_Bestellungen].[Auftragsdatum])*12+DatePart('m', [Kunden_  
Bestellungen].[Auftragsdatum])-1  
  
ORDER BY  
Year([Kunden_Bestellungen].[Auftragsdatum])*12+DatePart('m', [Kunden_  
Bestellungen].[Auftragsdatum])-1;
```

Hier werden sämtliche Produkte aufgelistet, die bestellt wurden, zeigt wie viele davon verkauft wurden und welchen Umsatz jedes Produkt generiert hat. Danach werden die einzelnen Datensätze nach Datum geordnet



Produktname	Auftragsdat	MAnzahl	MUMsatz
Bis(s) zum Abendrot	Juli 2010	2	Fr. 35.80
extreme ACTIVITY	Juli 2010	1	Fr. 54.90
Alice im Wunderland	August 2010	1	Fr. 24.90
Bis(s) zum Abendrot	August 2010	2	Fr. 35.80
extreme ACTIVITY	August 2010	1	Fr. 54.90
Leichenblässe	August 2010	2	Fr. 37.20
PC Berlitz English Intensivkurs	August 2010	1	Fr. 44.90
Recovery - Eminem	August 2010	2	Fr. 21.80

Abb. 19: Resultate der Abfrage „Monatsverkäufe/-umsätze“

4.4 Aktualisierungsabfragen

Aktualisierungsabfragen sind eine Unterform normaler Abfragen, mit deren Hilfe sich Daten in Tabellen ändern lassen. In der Datenbank werden sie benutzt, um den Lagerbestand der verschiedenen Produkte nach einer Bestellung bzw. Nachbestellung auf den neusten Stand zu bringen.

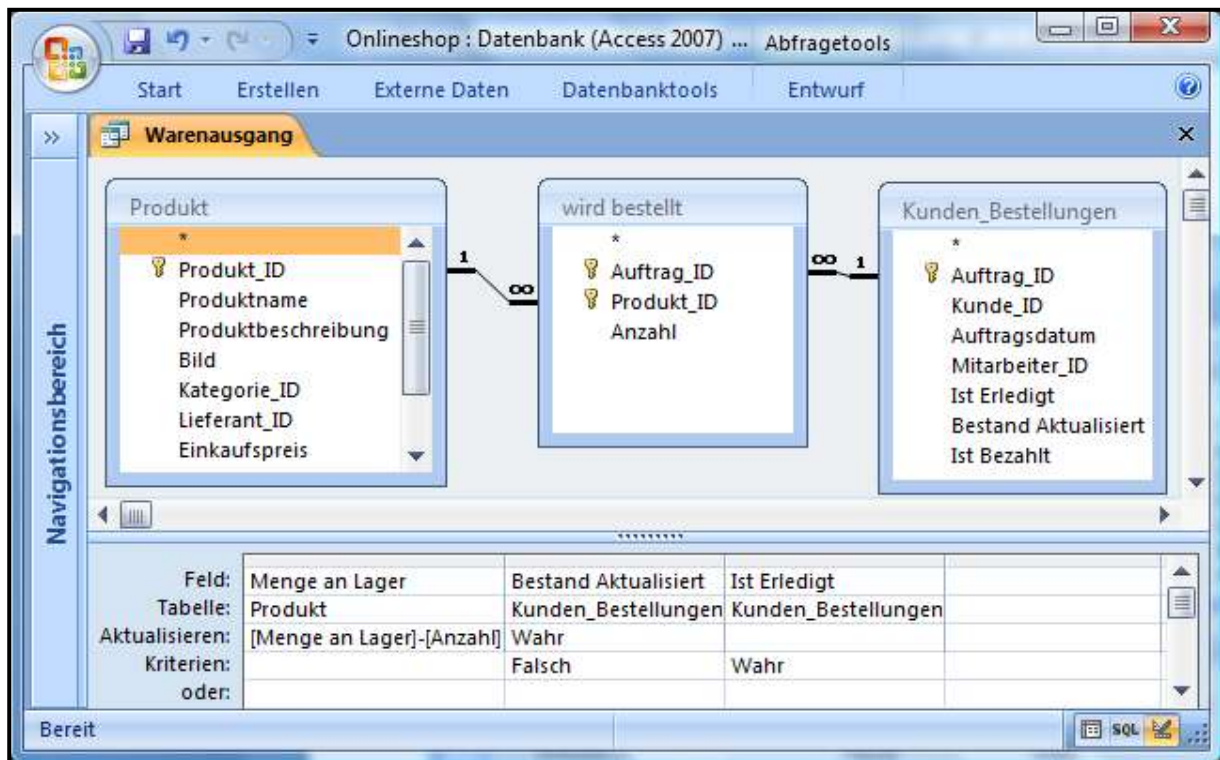


Abb. 20: Entwurfsansicht der Aktualisierungsabfrage „Warenausgang“

In der Entwurfsansicht können relativ simpel die Kriterien eingegeben werden, wann eine Aktualisierung einzelner Daten überhaupt vorgenommen wird und unter dem Punkt „Aktualisieren“ werden die Aktionen eingetragen, welche im Falle einer Aktualisierung durchgeführt werden. So wird z.Bsp. in der Abfrage *Warenausgang* die Werte der Attribute *Bestand Aktualisiert* und *Ist Erledigt* überprüft. Wurde eine Bestellung als erledigt gekennzeichnet und der Bestand noch nicht aktualisiert, so reduziert die Aktualisierungsabfrage den Wert *Menge an Lager* der bestellten Produkte um die jeweils bestellte Anzahl und ändert den Wert *Bestand Aktualisiert* in „Wahr“ um ein mehrmaliges Abziehen der bestellten Mengen zu verhindern. Das Attribut *Bestand Aktualisiert* wird zu keinem Zeitpunkt von den Benutzern der Datenbank gesehen, es dient einzig den Aktualisierungsabfragen *Wareneingang* und *Warenausgang* die dank eines Makros (werden im Kapitel Makros näher erläutert) automatisch ausgeführt werden, sobald die jeweiligen (Nach-)Bestellformulare geschlossen werden.

4.5 Formulare

Ein Formular ist die Schnittstelle zwischen der Datenbank und dem Benutzer. Es ist zwar möglich, Werte auch ohne Formular in die Datenbank einzugeben bzw. anzusehen, jedoch ist dies deutlich komplizierter, als wenn man extra angepasste Formulare dafür erstellt. Für die Führung des Onlineshops ist dies ein Muss, da nicht davon ausgegangen werden kann, dass jeder Mitarbeiter mit der Materie der Datenbanken vertraut ist und durch die Formulare können auch Menschen ohne Grundkenntnisse einfach neue Werte in die Datenbanken einfügen oder sich alte Daten ansehen.

Access 2007 bietet verschiedene Möglichkeiten Formulare zu erstellen. Ein Formular kann von Grund auf neu in der Entwurfsansicht erstellt werden, was jedoch einiges an Arbeit und Wissen erfordert. Das Generieren mit einem Assistenten ist eine weitere Möglichkeit, dieser erstellt aufgrund einer Tabelle oder einer Abfrage ein Formular mit sämtlichen beinhalteten Attributen. In Access 2007 gibt es neu auch Assistenten für geteilte Formulare, welche auf mehr als einer Tabelle oder Abfrage basieren. Formulare die mit einem Assistenten erzeugt wurden können jedoch im Nachhinein ebenfalls in der Entwurfsansicht modifiziert werden, weshalb die Erstellung eines „Grundgerüsts“ mit Hilfe des Assistenten empfehlenswert ist.

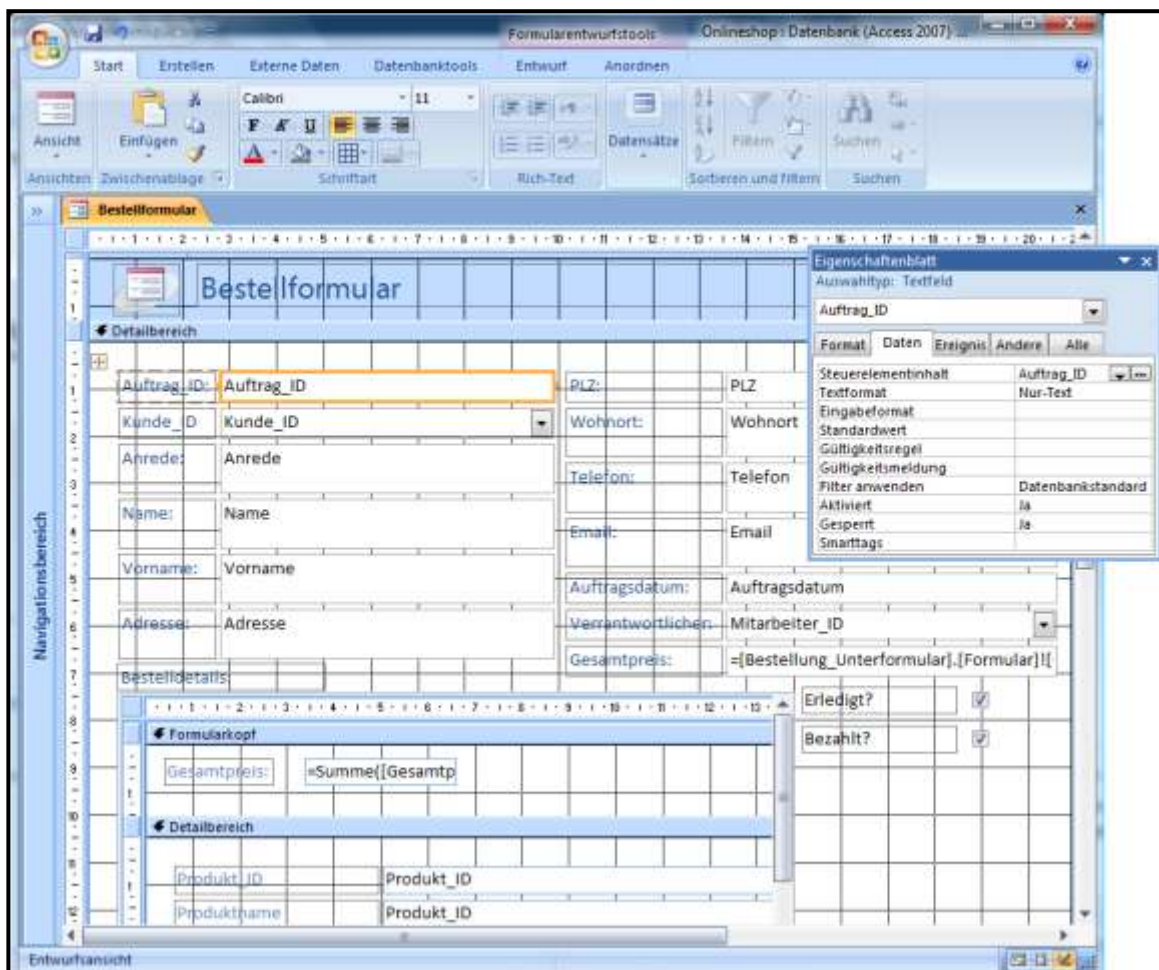


Abb. 21: Entwurfsansicht des Formulars „Bestellformular“

Desweiteren können auch ungebundene Formulare erstellt werden, die keine Daten aus Tabellen oder Abfragen erhalten sondern beispielsweise als Startbildschirm angezeigt werden, damit der Benutzer über bestimmte Funktionen Bescheid weiss oder sich innerhalb der Datenbank leichter zurechtfindet.

Über „Erstellen → Formular“ wird sofort ein Formular kreiert, welches die Daten der aktuell im Navigationsmenü ausgewählten Tabelle bzw. Abfrage enthält. Bearbeitet man dieses in der Entwurfsansicht, so ist das Eigenschaftenblatt unerlässlich. Es beinhaltet nahezu sämtliche Informationen über das Formular. Über das Eigenschaftenblatt lässt sich die Datenherkunft des Formulars und der einzelnen Felder festlegen. Es kann eingestellt werden, in welche Felder der Benutzer einen Inhalt einfügen darf und welche für ihn gesperrt bleiben, welche Standardwerte die Felder aufweisen, welche Einträge (un-)zulässig sind und welches Format die einzelnen Felder aufweisen. Desweiteren können fortgeschrittene Benutzer Ereignisprozeduren programmieren, welche beim Klick auf gewisse Schaltflächen oder bei bestimmten Eingaben abgespielt werden. Die Möglichkeiten sind zu vielfältig, als dass hier alle erwähnt werden könnten.

4.6 Berichte

Berichte werden verwendet um Daten übersichtlich darzustellen. Rechnungen und Mitarbeiter-, Kunden- oder Lieferantenlisten können mit ein paar wenigen Klicks generiert werden. Die Daten lassen sich in beinahe beliebiger Weise darstellen. Auch für Berichte stellt Access einen Assistenten zur Verfügung, am einfachsten ist es jedoch, Berichte auf Basis von Abfragen zu erstellen. Um die verschiedenen Listen darstellen bzw. drucken zu können wird auf die Möglichkeit der Berichtserstellung zugegriffen. Als Grundlage werden dafür entweder Tabellen oder Abfragen verwendet. Um beispielsweise eine Liste aller Kunden zu generieren, kann die Tabelle „Kunde“ mit ein paar wenigen Mausklicks verwendet werden. Sämtliche Attribute, welche in dieser Tabelle abgespeichert sind können so in eine übersichtliche und meist schönere Form gebracht werden.

Die Rechnungen werden ebenfalls mit Hilfe von Berichten erstellt. Diese basieren auf einer Abfrage. Um nicht jedes Mal sämtliche Rechnungen von längst vergangenen Bestellungen durchsehen zu müssen, um die richtige zu finden, werden nur die Rechnungen der Bestellungen angezeigt, deren „Ist Erledigt“-Attribut einen *Nein*-Wert enthält. Sobald die Lieferung zusammen mit der Rechnung verschickt wird, ändert der Mitarbeiter diesen Wert in „Ja“ und die entsprechende Rechnung wird ab sofort nicht mehr im Bericht angezeigt. Dies ist dank der Abfragen möglich, in welchen man die Kriterien einstellt, wann eine Rechnung im entsprechenden Bericht angezeigt wird und wann nicht.

Auch die Mahnungen werden mit Hilfe einer Abfrage automatisch generiert. Ist eine Bestellung länger als 14 Tage her und wurde ihr „Ist Bezahlt“-Attribut in dieser Zeit nicht auf „Ja“ geändert, so erscheint sie im Bericht der Mahnungen. Die Mitarbeiter verlieren so keine Zeit mit der Überprüfung der einzelnen Aufträge und der Besorgung der nötigen Informationen.

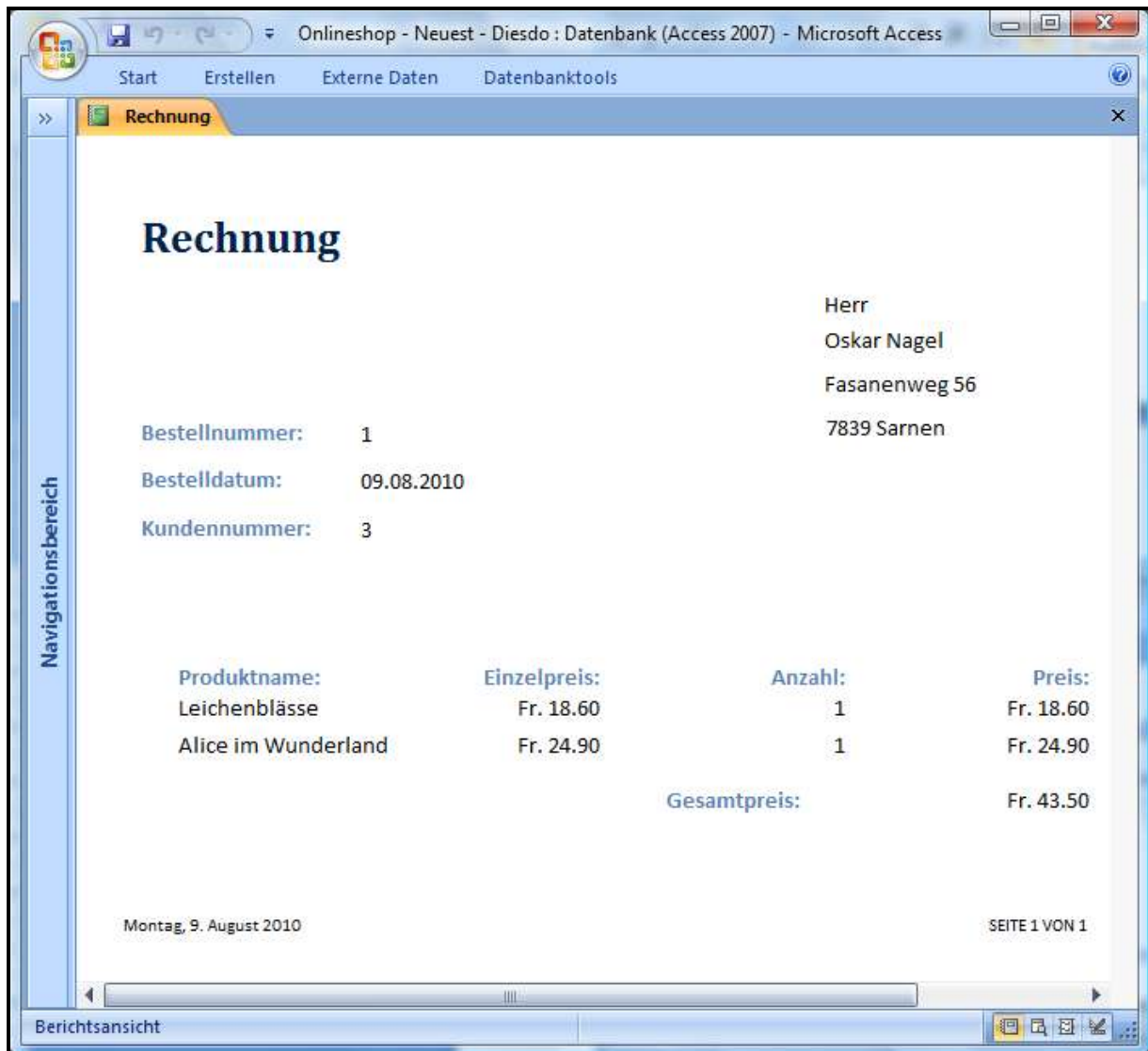


Abb. 22: Berichtsansicht des Berichtes „Rechnung“ basierend auf der gleichnamigen Abfrage

4.7 Makros

Makros sind sehr simple Programme, welche eine fest vorgegebene Abfolge von Befehlen ausführen. Mit Hilfe von Makros lassen sich gewisse Abläufe in der Datenbank selbstständig ausführen, so dass der Benutzer nicht jeden Schritt, den die Datenbank ausführt starten muss.

So wird beispielsweise das Makro „Autoexec“ dazu verwendet, das Startformular der Datenbank unverzüglich nach dem Start von Access anzuzeigen. Dieser Name ist von Access 2007 festgelegt und beschreibt das Makro, welches sofort nach dem Programmstart ausgeführt wird.

Desweiteren sind sämtliche Schaltflächen auf dem Startformular mit Makros versehen. Jeder Klick auf eine dieser Schaltflächen startet das entsprechende Makro und somit das entsprechende Formular bzw. den entsprechenden Bericht.

In der Datenbank wurden ebenfalls Makros verwendet, welche nach Markierung einer Bestellung oder Nachbestellung als „Ist Erledigt“ bzw. „Ist Eingetroffen“ automatisch mit dem Schliessen der jeweiligen Formulare ausgeführt werden. Die Makros starten dann die Aktualisierungsabfrage „Wareneingang“ bzw. „Warenausgang“, welche den Bestand der Waren automatisch aktualisieren.

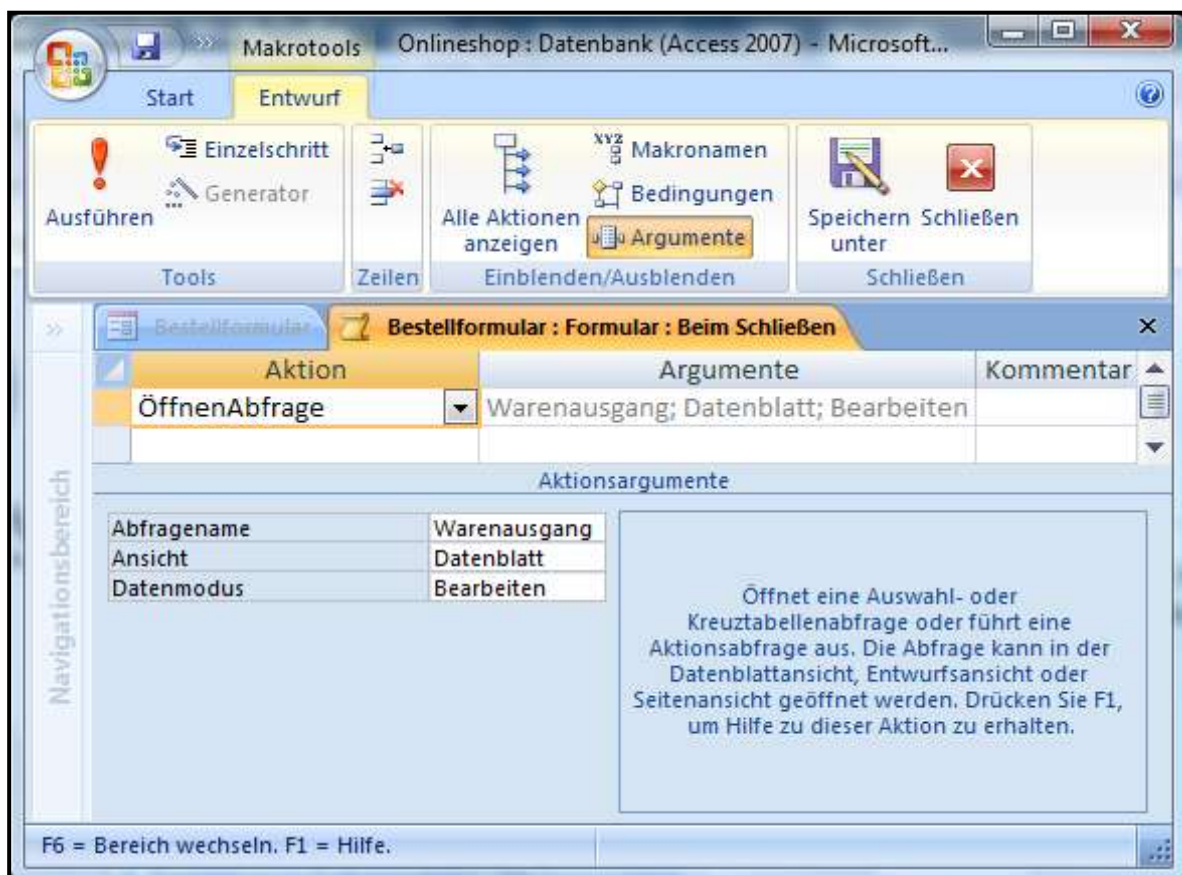


Abb. 23: Entwurfansicht eines Makros (Automatischer Start der Aktualisierungsabfrage „Warenausgang“ beim Schliessen des Formulars „Bestellformular“)

Die Erstellung von Makros ist denkbar simpel. So wird beispielsweise das Makro, welches den Warenbestand nach einer erledigten Bestellung aktualisiert folgendermassen programmiert: Da es sich um ein eingebettetes Makro handelt, wird im Eigenschaftenblatt des Bestellformulars unter *Ereignis* die Spalte *beim Schliessen* gesucht. Hier kann man nun ein neues Makro erstellen. Nun muss lediglich die Aktion ausgewählt werden, die das Makro ausführen soll (Es können auch gleich mehrere Aktionen einprogrammiert werden). Anschliessend müssen noch die Aktionsargumente eingestellt werden. Es muss konkretisiert werden, welche Abfrage nun geöffnet werden soll, in welcher Ansicht dies getan wird und wie der Datenmodus aussieht. Da die Abfrage „Warenausgang“ Daten aktualisiert, muss der Datenmodus hier „bearbeiten“ sein. Möchte man jedoch ein Formular anzeigen, mit dessen Hilfe neue Kunden erfasst werden können, so wird der Datenmodus auf „hinzufügen“ gesetzt. Mit dieser Einstellung werden keine bereits vorhandenen Daten angezeigt, es können lediglich neue Daten hinzugefügt werden. Ebenfalls möglich wäre noch die Einstellung „nur lesen“ die wie der Name schon sagt eine Veränderung an den Daten nicht zulässt, sondern diese nur anzeigt.

4.8 Code-Generator

Für fortgeschrittene Programmierer hält Access 2007 einen Code-Generator bereit, mit dessen Hilfe Ereignisse individuell geschrieben und angepasst werden können. Dieser Generator arbeitet mit der Programmiersprache Visual Basic for Applications, welche Einsteigern schnell gewisse Erfolge beschert. Nichtsdestotrotz ist es eine langwierige Angelegenheit eine Programmiersprache zu erlernen und die simplen Ereignisse in der Datenbank für den Onlineshop sind fast alle mit Makros zu lösen. Nur ein einziger Code war von Nöten, um die Funktionalität zu gewährleisten: Um im Formular „Nachbestellung“ nur die Produkte anzuzeigen, welche der ausgewählte Lieferant auch wirklich liefern kann, arbeitet das Unterformular mit einer SQL-Abfrage. Leider wird diese nicht automatisch erneuert, weshalb dies mit einem Code gemacht werden muss. Sobald die Lieferant_ID geändert wird, aktualisiert die SQL-Abfrage und filtert die Produkte des entsprechenden Lieferanten heraus. So wird vermieden, dass bei einem Lieferanten Produkte bestellt werden, welche er gar nicht anbietet.

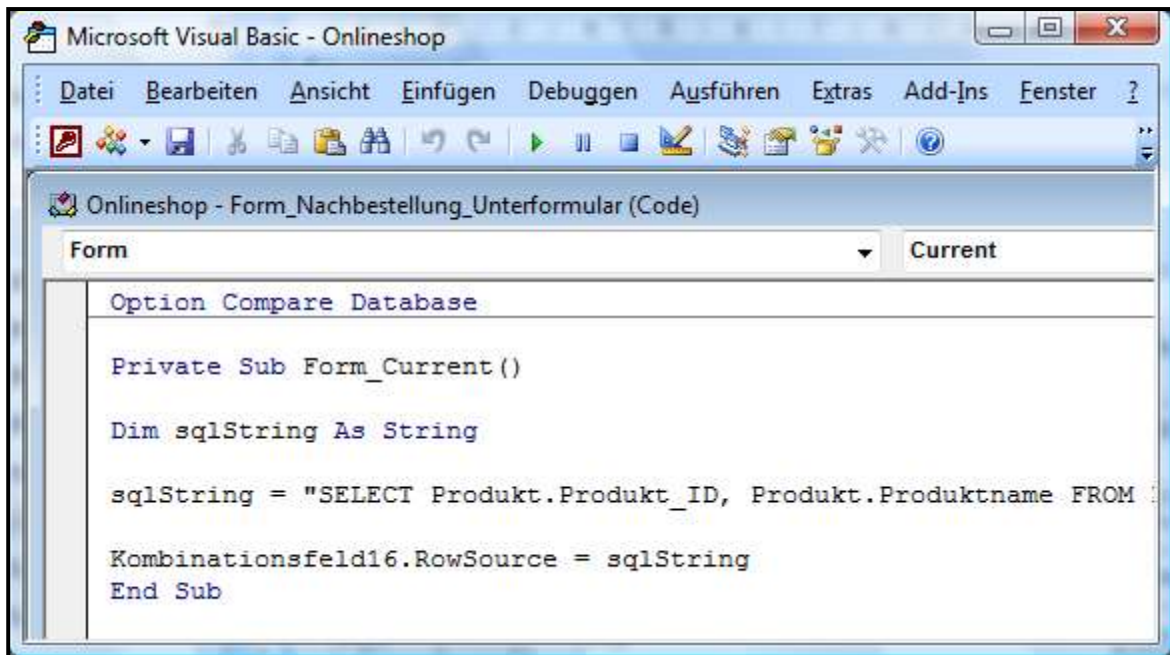


Abb. 24: Code-Generator mit Ausdruck, der die SQL-Abfrage des Produktauswahlfeldes durchführt

In dieser Arbeit wird jedoch nicht näher auf den Code-Generator eingegangen, da dessen Programmiersprache und Funktionen selbst eine Arbeit füllen könnten.

4.9 Datensicherheit

Die Daten, welche in der Datenbank gespeichert werden sollten vor Missbrauch geschützt werden. Kein Kunde oder Mitarbeiter möchte, dass seine Daten geraubt und/oder missbraucht werden können. In früheren Versionen von Access wurden verschiedene Funktionen angeboten um eine Datenbank vor unerwünschten Einsichten zu schützen. Es gab ein Verschlüsselungstool, welches den Zugriff auf die Datenbank mit anderen Programmen als Access verhinderte sowie die Möglichkeit, die Datenbank mit einem Kennwort zu schützen. Diese beiden Funktionen sind in Access 2007 sinnvollerweise zu einer einzigen zusammengefasst. Sobald die Datenbank über den entsprechenden Befehl mit einem Kennwort geschützt wird, wird sie automatisch verschlüsselt und so gegenüber Zugriffen mit anderen Programmen geschützt. Laut Angaben von Microsoft wurden auch die Algorithmen, welche für die Verschlüsselung eingesetzt werden, stark verbessert.



Abb. 25: Auswahl der Option „Mit Kennwort verschlüsseln“

5. Schlusswort

Datenbanken haben den Betrieb von Geschäften deutlich erleichtert. Während früher von Hand sämtliche Informationen festgehalten wurden, ist es heutzutage möglich, alle Bestände, Bestellungen, Mitarbeiter, etc. in ein und dem selben Konstrukt festzuhalten und jederzeit massgeschneiderte Informationen abzurufen. Viele Prozesse können automatisiert werden, so dass keine menschlichen Arbeitsressourcen dafür verbraucht werden. Die Erstellung einer solchen Datenbank und auch das Einfügen der Daten sowie das modellieren geeigneter Berichte und Abfragen sind zwar sehr zeitaufwändig. Dafür wird, sobald diese ersten Schritte abgeschlossen sind, eine Menge an Zeit und Mühe gespart, um einen reibungslosen Betrieb zu gewährleisten.

Durch die verschiedenen Hilfestellungen wie das Entitäten-Beziehungsmodell ist es sehr simpel, ein erstes Grundgerüst zu erstellen und mit dessen Hilfe dann die vollständige Implementierung durchzuführen. Auch Microsoft Access 2007 unterstützt den Benutzer mit seinen zahlreichen (Hilfe-) Funktionen und erleichtert den Einstieg in die Materie. Trotzdem ist das Erweitern der Kenntnisse unabdingbar für die Erstellung von Datenbanken mit mehr Inhalt als einer Namens- und Adressliste, da man sehr schnell an die Grenzen stösst.

Die Beispieldatenbank, welche aus dieser Arbeit resultiert, umfasst alle Möglichkeiten, die als Anforderungen gestellt wurden. Sämtliche Mitarbeiter, Lieferanten, Kunden und Bestellungen sowie deren Beziehungen untereinander können gespeichert, editiert und gelöscht werden. Somit könnte die Datenbank nun in eine tatsächliche Webseite eingebunden und benutzt werden. Es wären noch Anpassungen und Erweiterungen notwendig: In einem ersten Schritt müssten noch sämtliche Operationen zur Bestimmung von Steuern, Porto sowie evtl. Mengenrabatte modelliert werden. Zudem ist das Design nur zweckmässig und schlicht.

Zusammenfassend lässt sich sagen, dass die Datenbank bereits in ihrer jetzigen Form für ein Startup einsatzfähig wäre, jedoch längst nicht alle Funktionen enthält, die für einen unkomplizierten Betrieb mit automatischer Erfassung sämtlicher Transaktionen notwendig wären. Es sind noch viele Arbeitsstunden an der Datenbank nötig, um sie so zu modellieren, dass ein reibungsloser Ablauf sämtlicher Aktion gewährleistet ist.

Literaturverzeichnis

- [Meier 2010] Meier Andreas: Relationale und postrelationale Datenbanken, 7. Auflage, Springer Verlag, Berlin Heidelberg, 2010.
- [Preiss 2007] Preiss Nikolai: Entwurf und Verarbeitung relationaler Datenbanken, 1. Auflage, R.Oldenburg Verlag, München Wien 2007
- [Minhorst 2007] Minhorst André: Access 2007 – Das Grundlagenbuch für Entwickler 1. Auflage, Addison-Wesley Verlag, München, 2007
- [Schicker 2000] Schicker Edwin: Datenbanken und SQL 3. Auflage, Teubner Verlag, Stuttgart/Leipzig/Wiesbaden, 2000
- [Faeskorn-Woyke et al. 2007]
Faeskorn-Woyke Heide, Bertelsmeier Birgit, Riemer Petra, Bauer Elena: Datenbanksysteme 1. Auflage, Pearson Studium Verlag, München, 2007
- [Elmasri/Navathe 2009]
Ramez Elmasri, Shamkant B. Navathe: Grundlagen von Datenbanksystemen 3. Auflage, Addison-Wesley Verlag, München, 2009
- [Mathiessen/Unterstein 1998]
Günter Matthiessen, Michael Unterstein: Relationale Datenbanken und Standard-SQL 4. Auflage, Addison-Wesley Verlag, München, 2007
- [Kemper/Eickler 2009]
Alfons Kemper, André Eickler: Datenbanksysteme – Eine Einführung 7. Auflage, Oldenbourg-Verlag, München, 2009