Dokumentation Onlineshop

Modul 153 Datenmodelle entwickeln

18.04.2019

Alexander Siegenthaler

Inhaltsverzeichnis

[1. Zusammenfassung des Projektauftrags 3](#_Toc6444543)

[2. Zeitplan 3](#_Toc6444544)

[3. Anforderungen Datenmodell Onlineshop 4](#_Toc6444545)

[4. Konzeptionelles Datenmodell 5](#_Toc6444546)

[5. Physisches Datenmodell 6](#_Toc6444547)

[5.1. Tabellen definieren, Attribute festlegen 6](#_Toc6444548)

[5.2. Grafische Darstellung 7](#_Toc6444549)

[5.3. Normalisierung 7](#_Toc6444550)

[5.4. Prognose über Mengen und Häufigkeiten, Indexe festlegen 10](#_Toc6444551)

[5.5. SQL-Script 11](#_Toc6444552)

[6. Anforderungen überprüfen, Prototyp 12](#_Toc6444553)

1. Zusammenfassung des Projektauftrags

Ziel der Projektarbeit ist die Realisierung einer Datenbank, für einen Onlineshop, gemäss folgenden Anforderungen:

Externe Geschäftsfälle

* Angebote im Onlineshop ansehen
* Erfassen eines Warenkorbs, Produkte in den Warenkorb legen
* Produkte im Warenkorb kaufen
* Bestellte Produkte verfolgen
* Benutzerkonto pflegen

Interne Geschäftsfälle

* Anlegen und Pflegen der Produktekategorien
* Anlegen und Pflegen des Produktestamms
* Anlegen und Pflegen der Angaben zum Shop Betreiber
* Anlegen und Pflegen der Kundeninformationen
* Anlegen und Pflegen der Profildaten der Kunden
* Rechnung erstellen
* Verfügbarkeit der Produkte ermitteln

Im ersten Schritt sollen wir alle benötigten Entitäten identifizieren und beschreiben, um so ein konzeptionelles Datenmodell zu erstellen. Danach sollen wir die Tabellen definieren und das physische Datenmodell erstellen. Anschliessend wird das physische Datenmodell normalisiert, die Indizes festgelegt und das SQL-Script vom physischen Datenmodell erstellt. Zum Schluss sollen wir einen Prototyp der Datenbankanwendung erstellen (Access) und mit möglichen Datensätzen testen. Für die gesamte Projektarbeit wird eine Dokumentation inkl. Zeitplan verlangt.

1. Zeitplan

|  |  |
| --- | --- |
| **Datum** | **Arbeitsschritte** |
| 28.03.2019 | - Beschaffung der benötigten Informationen (Anforderungen)  - Zeitplan erstellen  - Zusammenfassung des Projektauftrags  - Entitäten identifizieren  - Konzeptionelles Datenmodell erstellen  - Dokumentation ergänzen |
| 04.04.2019 | - Evtl. Konzeptionelles Datenmodell ergänzen  - Tabellen definieren, Attribute festlegen  - Physisches Datenmodell grafisch darstellen  - Physisches Datenmodell normalisieren  - Dokumentation ergänzen |
| 17.04.2019 | - Indexe festlegen  - SQL-Script vom Physischen Datenmodell erstellen  - Dokumentation ergänzen |
| 18.04.2019 | - Prototyp (Access) der Datenbankanwendung erstellen  - Plausible Datensätze erfassen  - Projektarbeit überprüfen  - Dokumentation ergänzen  - Abgabe Projektarbeit |

1. Anforderungen Datenmodell Onlineshop

Ort

Die Entität "Ort" enthält eine Liste von Ortschaften und wird dem Kunden zugewiesen.

*Ort-Name, PLZ*

Kunde

Die Entität "Produkte" enthält Informationen über den Kunden und wird benötigt um den Kunden zu Identifizieren und um die Bestellung an die richtige Person zu liefern.

*Name, E-Mail, Alter, Adresse, Login-Name, Passwort*

Bestellung

Die Entität "Bestellung" verbindet den Warenkorb mit dem Kunden. Sie gibt ausserdem noch Auskunft über die Lieferart (Liefern/Abholen), das Bestell-/Übergabedatum und, falls das Produkt bezahlt wurde, über das Bezahldatum.

*KundeID, Bestelldatum, Lieferart, Übergabedatum, Bezahldatum*

Warenkorb

Die Entität "Warenkorb" verbindet die gewünschten Produkte, und deren Anzahl, mit der Bestellung.

*ProduktID, BestellungID, Anzahl*

Produkte

Die Entität "Produkte" enthält Informationen bezüglich der Produkte, die verkauft werden und wird benötigt, um zu wissen, welche Produkte der Kunde will, ob das Produkt noch vorhanden ist, was der Preis ist und zu welcher Kategorie das genannte Produkt gehört.

*Name, Preis, Anzahl, Kategorie*

Kategorie

Die Entität "Kategorie" enthält den Kategorienamen (z.B. Monitor, Maus) und eine Beschreibung der Kategorie.

*Name, Beschreibung*

Sonstige Überlegungen

Die Login-Daten könnte man noch in einer eigenen Entität festhalten, damit nicht alle Datenbank-Benutzer darauf Zugriff haben und um gleiche Login-Namen zu verhindern. Da ich aber von einem einzelnen Datenbank-Benutzer ausgehe, gebe ich einfach dem Attribut für den Login-Namen noch die Eigenschaft Unique mit.

1. Konzeptionelles Datenmodell



1. Physisches Datenmodell

Tabellen definieren, Attribute festlegen

Tabelle Ort

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribut** | **Datentyp** | **erforderlich** | **Primary Key** |
| ort\_id | Integer | X | X |
| ort | Varchar(30) | X |  |
| plz | Integer | X |  |

Tabelle Kunde

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribut** | **Datentyp** | **erforderlich** | **Primary Key** |
| kunde\_id | Integer | X | X |
| name | Varchar(20) | X |  |
| vorname | Varchar(20) | X |  |
| email | Varchar(40) | X |  |
| geburtstag | Date | X |  |
| strasse | Varchar(40) | X |  |
| ort\_id | Integer | X |  |
| login | Varchar(20) | X |  |
| passwort | Varchar(20) | X |  |

Tabelle Bestellung

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribut** | **Datentyp** | **erforderlich** | **Primary Key** |
| bestellung\_id | Integer | X | X |
| kunde\_id | Integer | X |  |
| bestelldatum | Datetime | X |  |
| lieferart | Enum(Lieferung,Abholen) | X |  |
| übergabedatum | Datetime |  |  |
| bezahldatum | Datetime |  |  |

Tabelle Warenkorb

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribut** | **Datentyp** | **erforderlich** | **Primary Key** |
| warenkorb\_id | Integer | X | X |
| produkt\_id | Integer | X |  |
| anzahl | Integer | X |  |
| bestellung\_id | Integer | X |  |

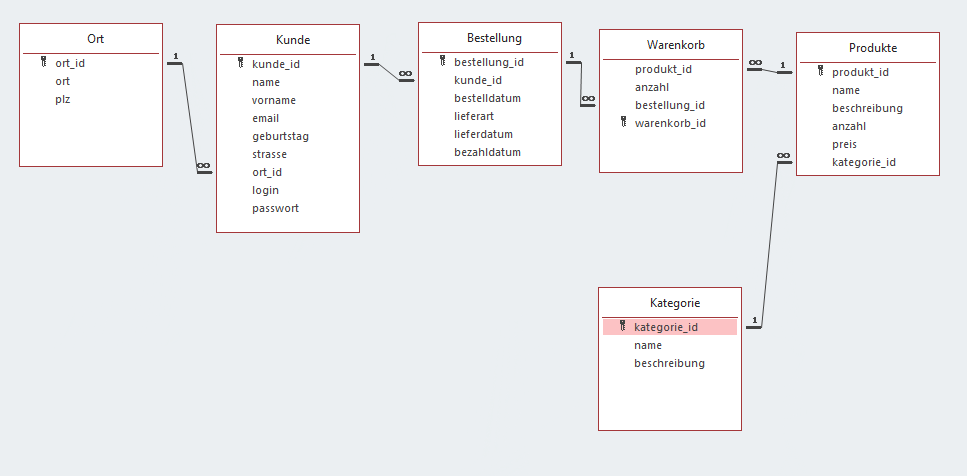
Tabelle Produkte

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribut** | **Datentyp** | **erforderlich** | **Primary Key** |
| produkt\_id | Integer | X | X |
| name | Varchar(30) | X |  |
| beschreibung | Text | X |  |
| anzahl | Integer | X |  |
| preis | Decimal(8,2) | X |  |
| kategorie\_id | Integer | X |  |

Tabelle Kategorie

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribut** | **Datentyp** | **erforderlich** | **Primary Key** |
| kategorie\_id | Integer | X | X |
| name | Varchar(30) | X |  |
| beschreibung | Text | X |  |

Grafische Darstellung



Normalisierung

**Ort**

**Erste Normalform**  
Die Tabelle Ort enthält den Ortsnamen, die Postleitzahl und die ID. Alle Attribute weisen einfache Attributwerte auf. Die erste Normalform ist somit gegeben.

**Zweite Normalform**  
Die Zweite Normalform ist erfüllt, da kein zusammengesetzter Primärschlüssel existiert.

**Dritte Normalform**  
Der Name(ort) ist transitiv von ort\_id abhängig. Dies könnte ich umgehen, wenn ich die Postleitzahl(plz) als Primärschlüssel definieren würde. Würden jedoch die Postleitzahlen geändert werden, müsste ich meine Datenbank dementsprechend anpassen. Daher habe ich mich gegen eine Änderung entschieden und deswegen erfüllt diese Tabelle nicht die dritte Normalform.

**Kunde**

**Erste Normalform**  
Die Tabelle Kunde enthält Informationen über den Kunden und seine Login-Daten. Der Name besteht aus Vornamen und Nachnamen, Adresse besteht aus Strasse und Ortschaft und der Login aus Login-Namen und Passwort. Bei der Adresse enthält die Strasse den Namen und die Nummer, da sich durch die Trennung in unterschiedliche Attribute jedoch keine Vor-/Nachteile ergeben benütze ich dafür ein einzelnes Attribut. Alle Attribute weisen einfache Attributwerte auf. Die erste Normalform ist somit gegeben.

**Zweite Normalform**  
Die Zweite Normalform ist erfüllt, da kein zusammengesetzter Primärschlüssel existiert.

**Dritte Normalform**Alle Attribute sind funktional vom Primärschlüssel abhängig. Die Tabelle befindet sich in der zweiten Normalform. Die dritte Normalform ist somit gegeben.

**Bestellung**

**Erste Normalform**  
Die Tabelle Bestellung enthält die Verbindung zum Kunden und zum Warenkorb und Informationen zum Bestell-/Bezahlvorgang. Das Bestell-/Bezahl- und Lieferdatum sind in unterschiedlichen Attributen festgehalten. Alle Attribute weisen einfache Attributwerte auf. Die erste Normalform ist somit gegeben.

**Zweite Normalform**  
Die Zweite Normalform ist erfüllt, da kein zusammengesetzter Primärschlüssel existiert.

**Dritte Normalform**Alle Attribute sind funktional vom Primärschlüssel abhängig. Die Tabelle befindet sich in der zweiten Normalform. Die dritte Normalform ist somit gegeben.

Warenkorb

**Erste Normalform**  
Die Tabelle Warenkorb enthält die Verbindung zur Bestellung und zu den Produkten und die Anzahl der Produkte. Alle Attribute weisen einfache Attributwerte auf. Die erste Normalform ist somit gegeben.

**Zweite Normalform**  
Die Zweite Normalform ist erfüllt, da kein zusammengesetzter Primärschlüssel existiert.

**Dritte Normalform**Alle Attribute sind funktional vom Primärschlüssel abhängig. Die Tabelle befindet sich in der zweiten Normalform. Die dritte Normalform ist somit gegeben.

Produkte

**Erste Normalform**  
Die Tabelle Produkte enthält Namen, Preis und Anzahl vom Produkt und die Verbindung zu der Kategorie. Der Name besteht aus dem Produktnamen und der Beschreibung. Die Anzahl und der Preis sind in unterschiedlichen Attributen. Alle Attribute weisen einfache Attributwerte auf. Die erste Normalform ist somit gegeben.

**Zweite Normalform**  
Die Zweite Normalform ist erfüllt, da kein zusammengesetzter Primärschlüssel existiert.

**Dritte Normalform**Alle Attribute sind funktional vom Primärschlüssel abhängig. Die Tabelle befindet sich in der zweiten Normalform. Die dritte Normalform ist somit gegeben.

Kategorie

**Erste Normalform**  
Die Tabelle Kategorie enthält den Namen der Kategorie. Der Name besteht aus dem Kategorienamen und der Beschreibung. Alle Attribute weisen einfache Attributwerte auf. Die erste Normalform ist somit gegeben.

**Zweite Normalform**  
Die Zweite Normalform ist erfüllt, da kein zusammengesetzter Primärschlüssel existiert.

**Dritte Normalform**Alle Attribute sind funktional vom Primärschlüssel abhängig. Die Tabelle befindet sich in der zweiten Normalform. Die dritte Normalform ist somit gegeben.

Prognose über Mengen und Häufigkeiten, Indexe festlegen

**Kundenanforderungen und Zugriffe auf Tabellen**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Kundenanforderungen** | **Ort** | **Kunde** | **Bestellung** | **Warenkorb** | **Produkte** | **Kategorie** |
| Kunde: Angebote im Onlineshop ansehen | - | - | - | - | X | X |
| Kunde: Erfassen eines Warenkorbs | X | X | X | X | X | X |
| Kunde: Bestellung | X | X | X | X | X | X |
| Kunde: Bestellungsvorgang verfolgen | - | X | X | X | X | X |
| Kunde: Benutzerkonto pflegen | X | X | - | - | - | - |
| Intern: Anlegen/Pflegen Kategorien | - | - | - | - | - | X |
| Intern: Anlegen/Pflegen Produkte | - | - | - | - | X | X |
| Intern: Anlegen/Pflegen Kundeninformationen | X | X | - | - | - | - |
| Intern: Anlegen/Pflegen Profildaten(Login) | - | X | - | - | - | - |
| Rechnung erstellen | - | X | X | X | X | - |
| Verfügbarkeit der Produkte | - | - | - | - | X | - |

**Ort**Auf den Ortsnamen und die Plz gibt es vergleichsweise wenig Zugriffe, daher keine Indizes.

**Kunde**Auf den Kunden gibt es relativ viele Zugriffe, daher indexier ich das Attribut name und vorname.

**Bestellung**Auf die Bestellung gibt es vergleichsweise wenig Zugriffe, daher keine Indizes.

**Warenkorb**Auf den Warenkorb gibt es vergleichsweise wenig Zugriffe, daher keine Indizes.

**Produkte**Auf die Produkte gibt es relativ viele Zugriffe, daher indexier ich das Attribut name, anzahl und preis.

**Kategorie**Auf die Kategorie gibt es relativ viele Zugriffe, daher indexier ich das Attribut name.

**Sonstige Überlegungen**Da die meisten Datenbanktools, mit denen ich gearbeitet habe, prinzipiell alle Fremdschlüssel indexieren, sind diese in meiner Datenbank ebenfalls indexiert. Ausserdem macht es Sinn diese zu indexieren da auf diese vergleichsweise oft zugegriffen wird.

SQL-Script

drop database if exists onlineshop;

create database onlineshop;

use onlineshop;

create table ort(

ort\_id integer not null auto\_increment,

ort varchar(30) not null,

plz integer not null,

primary key(ort\_id)

);

create table kunde(

kunde\_id integer not null auto\_increment,

name varchar(20) not null,

vorname varchar(20) not null,

email varchar(40) not null,

geburtstag date not null,

strasse varchar(40) not null,

ort\_id integer not null,

login varchar(20) not null unique,

passwort varchar(20) not null,

primary key(kunde\_id),

foreign key(ort\_id) references ort(ort\_id) on update cascade on delete cascade

);

create table kategorie(

kategorie\_id integer not null auto\_increment,

name varchar(30) not null,

beschreibung text not null,

primary key(kategorie\_id)

);

create table produkte(

produkt\_id integer not null auto\_increment,

name varchar(30) not null,

beschreibung text not null,

anzahl integer not null,

preis decimal(8,2) not null,

kategorie\_id integer not null,

primary key(produkt\_id),

foreign key(kategorie\_id) references kategorie(kategorie\_id) on update cascade on delete cascade

);

create table bestellung(

bestellung\_id integer not null auto\_increment,

kunde\_id integer not null,

bestelldatum datetime not null default current\_timestamp,

lieferart enum('lieferung','abholen') not null default 'abholen',

uebergabedatum datetime,

bezahldatum datetime,

primary key(bestellung\_id),

foreign key(kunde\_id) references kunde(kunde\_id) on update cascade on delete cascade

);

create table warenkorb(

warenkorb\_id integer not null auto\_increment,

produkt\_id integer not null,

bestellung\_id integer not null,

anzahl integer not null,

primary key(warenkorb\_id),

foreign key(produkt\_id) references produkte(produkt\_id) on update cascade on delete cascade,

foreign key(bestellung\_id) references bestellung(bestellung\_id) on update cascade on delete cascade

);

#Indexe definieren

CREATE INDEX ort\_id\_index

ON kunde (ort\_id);

CREATE INDEX kategorie\_id\_index

ON produkte (kategorie\_id);

CREATE INDEX kunde\_id\_index

ON bestellung (kunde\_id);

CREATE INDEX produkt\_id\_index

ON warenkorb (produkt\_id);

CREATE INDEX bestellung\_id\_index

ON warenkorb (bestellung\_id);

CREATE INDEX name\_kunde\_index

ON kunde (name);

CREATE INDEX vorname\_kunde\_index

ON kunde (vorname);

CREATE INDEX name\_produkte\_index

ON produkte (name);

CREATE INDEX anzahl\_produkte\_index

ON produkte (anzahl);

CREATE INDEX preis\_produkte\_index

ON produkte (preis);

CREATE INDEX name\_kategorie\_index

ON kategorie (name);

1. Anforderungen überprüfen, Prototyp

Für die Überprüfung ist eine Access Anwendung mit entsprechenden Formularen erstellt worden. Diverse Datenfeldtypen wurden geändert, da das Sql-Script für MariaDB erstellt wurde und diese teilweise in Access nicht vorhanden sind. Gewisse Attributnamen wurden ebenfalls leicht abgeändert, da Access reservierte Namen hat und es sonst zu Fehlern kommen kann z.B. name -> namen. Anschliessend wurde für jede Tabelle plausible Datensätze erfasst.