

Musterlösung zur Hausarbeit

January 22, 2007

1 Aufgabe 1

1.1 Anforderungsanalyse

Es soll eine Datenbank-Anwendung für eine Öffentliche Leihbibliothek erstellt werden. Zu dieser gehört die Verwaltung des Buch- und Zeitschriftenbestandes, die Verwaltung von Benutzer- und Personaldaten, sowie natürlich das Ausleihen und die Rückgabe von Leihartikeln (inklusive der Berechnung von Säumnisgebühren).

Beteiligte Akteure sind Bibliotheksbenutzer, die über eine Bibliotheksausweisnummer (Benutzer-ID) identifiziert werden sollen, und Bibliothekare, die über eine Personalnummer identifiziert werden. Zu beiden Personengruppen müssen Adresse und Kontaktdaten, für die Angestellten eventuell auch Angaben wie Gehalt oder verbleibende Urlaubstage gespeichert werden.

Von der Datenbank-Anwendung sollen insbesondere folgende Aufgaben erledigt werden:

1. Erfassung bibliographischer Daten zu Büchern und Zeitschriften, sowie die Suche in diesen Daten.
2. Verwaltung von Medien (Exemplare von Büchern und Ausgaben von Zeitschriften), inklusive der Neueinstellung, Zuordnung und Ausmusterung solcher Medien.
3. Aufnahme und Änderung von Personendaten, sowie Ausstellen und Adressierung von Rechnungen und Gehaltsabrechnungen.

4. Ausleihe und Rückgabe von Medien, unter Berücksichtigung von Leihfristen. Bei Überziehung sollen Säumnisgebühren fällig sein, die vierteljährlich zu einer Rechnung zusammengefasst werden.

Zur Erfassung der bibliographischen Daten werden Angaben über das **Buch** (wie z.B. ISBN, Titel, Verlag) bzw. die Zeitschrift (wie z.B. ISSN, Titel, Herausgeber) benötigt. Zu Büchern sollen auch Autoren Daten aufgenommen werden. Mehrere Autoren können gemeinsam ein Buch verfasst haben, jeder gespeicherte Autor hat mindestens ein Buch geschrieben, und kann beliebig viele Bücher mitverfasst haben. Zudem sollen zu Büchern Deskriptoren (Schlagworte) aufgenommen werden, nach denen gesucht werden kann. Ein Buch kann mehrere Deskriptoren besitzen.

Zur Verwaltung der Medien ist es nötig, die einzelnen Exemplare eines Buches durch eine eindeutige Nummer zu unterscheiden. Buchexemplare können entweder ausleihbar sein oder nur zum Präsenzbestand gehören. Ein Exemplar ist ausserdem in einem definierten Zustand (entweder „neuwertig“, „mit Gebrauchsspuren“, „stark beschädigt“ oder „auszumustern“). Jedes Buchexemplar im Bestand soll ausserdem einer bestimmten Sektion zugeordnet sein, und in dieser in genau einem Regal stehen. Ein einzelnes Regal gehört immer nur zu einer Sektion, aber eine Sektion kann sich über mehrere Regale erstrecken. Für Zeitschriften soll nur die Ausgabe festgehalten werden. Es wird angenommen, dass jede Ausgabe nur einmal vorrätig ist und nicht entliehen werden, sowie dass Zeitschriften separat von den Büchern geführt werden, und daher auch keine Sektion bzw. Regalnummer haben.

Zur Führung der Personendaten sollen Adressen und Telefonnummern zu allen Personen aufgenommen werden. Bibliothekare besitzen ausserdem ein monatliches Gehalt, eine Kontonummer und einen Eintrag für den verbleibenden Resturlaub. Identifiziert werden Personen über eine künstliche ID (Personalnummer oder Benutzernummer). Es müssen ausserdem alle benutzerspezifischen Buchungsvorgänge (fällige Gebühren bzw. eingegangene Zahlungen) mit Betrag und Vorgangsdatum festgehalten werden.

Zur Verwaltung der Leihvorgänge soll jeder Vorgang mit einer fortlaufenden, identifizierenden Nummer versehen werden. An einem Leihvorgang sind immer ein Benutzer und ein Bibliothekar beteiligt. Um unterschiedliche Rückgabetermine für einzelne Bücher einer Ausleihe zu erlauben, sollen die Posten der Ausleihe eine separate Entität sein, die einem Leihvorgang eindeutig zugeordnet sind. Jeder Posten verweist auf ein bestimmtes Medium, hat ein Fälligkeitsdatum und das Datum der Rückgabe,

das zunächst NULL sein soll. Wird der Artikel zurückgegeben, so wird das Datum gesetzt.

Um festzustellen, ob ein Exemplar eines Buches verliehen ist, darf es von keinem Ausleihposten mit leerem Rückgabedatum referenziert werden.

Andere Nebenbedingungen:

- Ein bestimmtes Exemplar kann immer nur von einem Benutzer zur gleichen Zeit ausgeliehen sein.
- Der selbe Benutzer kann ein Exemplar mehrmals zu unterschiedlichen Zeiten ausleihen.
- Zur Erhaltung der Leihhistorie sollen keine Leihvorgänge gelöscht werden.
- Zur Erhaltung der Rechnungshistorie sollen keine Rechnungsposten gelöscht werden. Ein Konto ist ausgeglichen, wenn sich Zahlungen und Säumnigkeiten für einen Benutzer genau aufheben.

1.2 ER-Diagramm

Siehe separate Datei.

2 Aufgabe 2

2.1 Relationales Schema

Buch: {[ISBN: String, Titel: String, Erscheinungsjahr: Date, Verlag: String]}

Exemplar: {[BibNr: Integer, Zustand: String, ausleihbar: Boolean]}

hatExemplar: {[BibNr: Integer, ISBN: String]}

Zeitschrift: {[ISSN: String, Titel: String, Herausgeber: String, Verlag: String]}

Ausgabe: {[BibNr: Integer, Nummer: Integer, Erscheinungsjahr: Date]}

hatAusgabe: {[BibNr: Integer, ISSN: String]}

Standplatz: {[Sektion: String, Regal: Integer]}

hatStandplatz: {[BibNr: Integer, Regal: Integer]}

hatDeskriptor: {[ISBN: String, DID: Integer]}

Deskriptor: {[DID: Integer, Schlagwort: String]}

Autor: {[AutorID: String, Nachname: String, Vorname: String]}
 hatGeschrieben: { AutorID: String, ISBN: String }
 Bibliothekar: {[PersID: Integer, Nachname: String, Vorname: String, Telefon: String, StrasseHNr: String, PLZ: Integer, Ort: String, Gehalt: Decimal, Resturlaub: Integer]}
 Benutzer: {[BenID: Integer, Nachname: String, Vorname: String, Telefon: String, StrasseHNr: String, PLZ: Integer, Ort: String]}
 Kontenvorgang: {[VorgangsID: Integer, VorgangsDatum: Date, Betrag: Decimal]}
 gehörtZu: {[VorgangsID: Integer, BenID: Integer]}
 ausleihen: {[BibNr: Integer, LeihDatum: Date, BenID: Integer, PersID: Integer, FälligDatum: Date, RückDatum: Date]}

2.2 Verfeinerung

- Exemplar und hatExemplar werden zu:
Exemplar: {[BibNr: Integer, ISBN: String, Zustand: String, ausleihbar: Boolean]}
- Ausgabe und hatAusgabe werden zu:
Ausgabe: {[BibNr: Integer, ISSN: String, Nummer: Integer, Erscheinungsjahr: Date]}
- Kontenvorgang und gehörtZu werden zu:
Kontenvorgang: {[VorgangsID: Integer, BenID: Integer, VorgangsDatum: Date, Betrag: Decimal]}
- Exemplar und hatStandplatz werden zu:
Exemplar: {[BibNr: Integer, ISBN: String, Regal: Integer, Zustand: String, ausleihbar: Boolean]}

2.3 Normalformen

Die Relationen liegen bereits in der 2. Normalform vor. Insbesondere haben nur die Relation `hatGeschrieben`, `hatDeskriptor` und die Relation `ausleihen` einen zusammengesetzten Schlüssel (aber `hatGeschrieben` und `hatDeskriptor` besitzen keine Nicht-Schlüsselattribute). Alle NSA von `ausleihen` sind voll funktional abhängig vom Kandidatenschlüssel.

Durch Betrachtung der funktionalen Abhängigkeiten ist für diese Relationen leicht zu erkennen, dass keine transitiven Abhängigkeiten vorliegen. Somit sind die Relationen auch bereits in der 3NF.

Für `ausleihen` führen wir zur Vereinfachung noch einen künstlichen Schlüssel `LeihID` ein, der `LeihDatum`, `PersID` und `BenID` eindeutig festlegt. Um die 3NF zu wahren, wird die Relation zerlegt:

`ausleihen`: {[`LeihID`: Integer, `LeihDatum`: Date, `BenID`: Integer, `PersID`: Integer]} `Ausleihposten`: {[`LeihID`: Integer, `BibNr`: Integer, `FälligDatum`: Date, `RückDatum`: Date]}

In den beiden neu entstandenen Relationen sind alle Nicht-Schlüsselattribute voll dem jeweils einzigen Kandidatenschlüssel abhängig, und es existieren keine transitiven Abhängigkeiten.

3 Aufgabe 3: Tabellendefinitionen

```
CREATE DATABASE Bibliothek;  
CONNECT Bibliothek;
```

```
CREATE TABLE Benutzer (  
    BenID      BIGINT NOT NULL AUTO_INCREMENT,  
    Nachname   VARCHAR(30) NOT NULL,  
    Vorname    VARCHAR(30) NOT NULL,  
    Telefon    VARCHAR(15),  
    StrasseHNr VARCHAR(50) NOT NULL,  
    Plz        INT NOT NULL  
              CHECK Plz BETWEEN 1000 AND 99999,  
    Ort        VARCHAR(30) NOT NULL DEFAULT 'Duisburg',  
    PRIMARY KEY (BenID)  
);
```

```
CREATE TABLE Bibliothekar (  
    PersID     BIGINT NOT NULL AUTO_INCREMENT,  
    Nachname   VARCHAR(30) NOT NULL,  
    Vorname    VARCHAR(30) NOT NULL,  
    Telefon    VARCHAR(15) NOT NULL,  
    StrasseHNr VARCHAR(50) NOT NULL,  
    Plz        INT NOT NULL CHECK Plz BETWEEN 1000 AND 99999,
```

```

    Ort          VARCHAR(30) NOT NULL DEFAULT 'Duisburg',
    Gehalt       DECIMAL(6,2) NOT NULL
                CHECK Gehalt BETWEEN 2000.00 AND 2500.00,
    Resturlaub   SMALLINT CHECK Resturlaub < 45,
    PRIMARY KEY (PersID)
);

```

```

CREATE TABLE Autor (
    AutorID     BIGINT NOT NULL AUTO_INCREMENT,
    Nachname    VARCHAR(30) NOT NULL,
    Vorname     VARCHAR(30) NOT NULL,
    PRIMARY KEY (AutorID)
);

```

```

CREATE TABLE Buch (
    Isbn        CHAR(13) NOT NULL PRIMARY KEY,
    Titel       VARCHAR(40) NOT NULL,
    ErschJahr   INT NOT NULL
                CHECK ErschJahr BETWEEN 1200 AND 2007,
    Verlag      VARCHAR(20) NOT NULL
);

```

```

CREATE TABLE HatGeschrieben (
    AutorID     BIGINT NOT NULL REFERENCES Autor,
    Isbn        CHAR(13) NOT NULL REFERENCES Buch,
    PRIMARY KEY (AutorID, Isbn)
);

```

```

CREATE TABLE Deskriptor (
    DID         INT NOT NULL PRIMARY KEY,
    Schlagwort  VARCHAR(30)
);

```

```

CREATE TABLE HatDeskriptor (
    Isbn        CHAR(13) NOT NULL REFERENCES Buch,
    DID         INT NOT NULL REFERENCES Deskriptor,
    PRIMARY KEY (Isbn, DID)
);

```

```

CREATE TABLE Standplatz (
    Regal          INT NOT NULL PRIMARY KEY,
    Sektion       VARCHAR(20) NOT NULL
);

CREATE TABLE Exemplar (
    BibNr         BIGINT NOT NULL AUTO_INCREMENT,
    Isbn          CHAR(13) NOT NULL REFERENCES Buch,
    Regal         INT NOT NULL REFERENCES Standplatz,
    Zustand       CHAR(15) NOT NULL DEFAULT 'neuwertig'
                CHECK Zustand IN ('neuwertig','Gebrauchsspuren',
                                'beschädigt','auszumustern'),
    ausleihbar    TINYINT(1) NOT NULL DEFAULT 1,
    PRIMARY KEY (BibNr)
);

CREATE TABLE Zeitschrift (
    Issn          VARCHAR(15) NOT NULL PRIMARY KEY,
    Titel         VARCHAR(40) NOT NULL,
    Herausgeber   VARCHAR(30),
    Verlag        VARCHAR(30)
);

CREATE TABLE Ausgabe (
    Issn          VARCHAR(15) NOT NULL REFERENCES Zeitschrift,
    Nummer        INT NOT NULL,
    ErschJahr     INT NOT NULL
                CHECK ErschJahr BETWEEN 1900 AND 2007,
    PRIMARY KEY (Issn, Nummer)
);

CREATE TABLE KontenVorgang (
    VorgangsID    BIGINT NOT NULL AUTO_INCREMENT,
    BenID         BIGINT NOT NULL REFERENCES Benutzer,
    VDatum        DATE NOT NULL,
    Betrag        DECIMAL(5,2) NOT NULL,
    PRIMARY KEY (VorgangsID)
);

```

```
);
```

```
CREATE TABLE Ausleihen (  
    LeihID      BIGINT NOT NULL AUTO_INCREMENT,  
    LeihDatum   DATE NOT NULL,  
    BenID       BIGINT NOT NULL REFERENCES Benutzer,  
    PersID      BIGINT NOT NULL REFERENCES Bibliothekar,  
    PRIMARY KEY (LeihID)  
);
```

```
CREATE TABLE Ausleihposten (  
    LeihID      BIGINT NOT NULL REFERENCES Ausleihen,  
    BibNr       BIGINT NOT NULL REFERENCES Exemplar,  
    FalligDatum DATE NOT NULL,  
    RuckDatum   DATE DEFAULT NULL,  
    PRIMARY KEY (LeihID, BibNr)  
);
```

4 Aufgabe 4: Anfragen

Da wo systemspezifische Funktionen benutzt werden, orientiert sich die Syntax an MySQL.

Zunächst einmal soll die Datenbank mit einigen Testeinträgen gefüllt werden:

```
INSERT INTO Bibliothekar  
    (PersID, Nachname, Vorname, Telefon, StrasseHNr, Plz, Ort,  
     Gehalt)  
VALUES (NULL, 'Kriewel', 'Sascha',  
        '0203-XXXXXXX',  
        'Eine Strasse 1', 47048, 'Duisburg',  
        2500);  
INSERT INTO Autor  
    VALUES (NULL, 'Follett', 'Ken'),  
            (NULL, 'Kay', 'Guy Gavriel');  
INSERT INTO Buch  
    VALUES ('340412815X', 'Die Säulen der Erde', 'Lübbe', 1998),
```

```

        ('3453108418', 'Ein Lied für Arbonne', 'Heyne', 1996),
        ('3453097386', 'Die Löwen von Al-Rassan', 'Heyne', 1996);
INSERT INTO HatGeschrieben
    SELECT AutorID, '340412815X' FROM Autor WHERE Nachname='Follett';
INSERT INTO HatGeschrieben
    SELECT AutorID, '3453108418' FROM Autor WHERE Nachname='Kay';
INSERT INTO HatGeschrieben
    SELECT AutorID, '3453097386' FROM Autor WHERE Nachname='Kay';
INSERT INTO Standplatz
    VALUES (100, 'Belletristik');
INSERT INTO Exemplar (BibNr, Isbn, Regal)
    VALUES (NULL, '340412815X', 100),
    (NULL, '340412815X', 100),
    (NULL, '340412815X', 100),
    (NULL, '3453108418', 100),
    (NULL, '3453108418', 100),
    (NULL, '3453097386', 100);

```

4.1 Neuer Nutzer und Ausleihe

Alle IDs werden per AUTO_INCREMENT automatisch vergeben (bei der Einfügung wird das auto-inkrementierte Attribut mit NULL versehen).

Hier wird erste verfügbare Exemplar des Buches mit dem angegebenen Titel ausgeliehen.

```

INSERT INTO Benutzer
    VALUES (NULL, 'Müller', 'Heike',
            '0203-XXXXXXX',
            'Irgendwo 45', 99999, 'Nirgendwo');

```

Variante 1: Hier wird erste verfügbare Exemplar des Buches mit dem angegebenen Titel vom gerade eingetragenen Benutzer ausgeliehen.

```

INSERT INTO Ausleihen (LeihID, LeihDatum, BenID)
    VALUES (NULL, '2006-12-04', LAST_INSERT_ID());
INSERT INTO Ausleihposten (LeihID, BibNr, FalligDatum, RuckDatum)
    SELECT LAST_INSERT_ID(), e.BibNr, '2006-12-25', NULL
    FROM Buch b, Exemplar e

```

```

WHERE b.Titel = 'Die Säulen der Erde'
      AND b.Isbn = e.Isbn
      AND e.BibNr NOT IN (
          SELECT BibNr
          FROM Ausleihposten
          WHERE RuckDatum IS NULL
      )
LIMIT 1;

```

Variante 2: Die Exemplar-Nummer (987654) ist durch Einscannen des Buches, die Benutzer-Nummer (123456789) ist durch Einscannen des Benutzerausweis bekannt. Statt LAST_INSERT_ID könnte im zweiten Befehl auch die tatsächliche ID des Ausleihvorgangs eingesetzt werden.

```

INSERT INTO Ausleihe (LeihID, LeihDatum, BenID)
VALUES (NULL, '2006-12-04', 123456789);
INSERT INTO Ausleihposten (Leih, BibID, FalligDatum, RuckDatum)
VALUES (LAST_INSERT_ID(), 987654, '2006-12-25', NULL);

```

4.2 Verfügbarkeit überprüfen

```

SELECT COUNT(*)
FROM Buch b, Exemplar e
WHERE b.Titel = 'Die Säulen der Erde'
      AND b.Isbn = e.Isbn
      AND e.BibNr NOT IN (
          SELECT BibNr
          FROM Ausleihposten
          WHERE RuckDatum IS NULL
      )

```

4.3 Rückgabe

Die ID des Ausleihvorgangs offenen Ausleihvorgangs zu dem zurückgegebenen sei 314159265.

```

UPDATE Ausleihposten
SET RuckDatum = '2007-01-02'

```

```

WHERE LeihID = 314159265
      AND BibNr = 987654;
INSERT INTO KontenVorgang (VorgangsID, BenID, VDatum, Betrag)
VALUES (NULL, 123456789, CURRENT_DATE(), 2.00);

```

4.4 Rechnungsstellung

```

SELECT b.Nachname, b.Vorname, b.StrasseHNr, b.PLZ, b.Ort, SUM(k.Betrag)
FROM Benutzer b, KontenVorgang k
WHERE b.BenID = k.BenID
GROUP BY b.BenID
HAVING SUM(k.Betrag) > 0;

```

4.5 Suche nach Autor

```

SELECT DISTINCT b.Titel
FROM Buch b, Autor a, hatGeschrieben h
WHERE b.ISBN = h.ISBN
      AND a.AutorID = h.AutorID
      AND a.Vorname = 'Guy Gavriel'
      AND a.Nachname = 'Kay';

```

```

SELECT DISTINCT b.Titel
FROM Buch b, Exemplar e, Autor a, hatGeschrieben h
WHERE b.ISBN = e.ISBN
      AND e.ISBN = h.ISBN
      AND a.AutorID = h.AutorID
      AND a.Vorname = 'Guy Gavriel'
      AND a.Nachname = 'Kay'
      AND e.BibNr NOT IN (
        SELECT BibNr
        FROM Ausleihposten
        WHERE RuckDatum IS NULL
      )
)

```

4.6 Suche nach Schlagworten

```

SELECT b.Titel, b.ISBN

```

```

FROM Buch b, Deskriptor d1, Deskriptor d2,
     hatDeskriptor h1, hatDeskriptor h2
WHERE b.ISBN = h1.ISBN
     AND b.ISBN = h2.ISBN
     AND h1.DID = d1.DID
     AND h2.DID = d2.DID
     AND d1.DeskName = 'Wissenschaftsgeschichte'
     AND d2.DeskName = 'Evolutionsbiologie'

```

4.7 Neue Bücher und Exemplare

```

INSERT INTO Buch (ISBN, Titel, ErschJahr, Verlag)
VALUES ( '3486273922',
        'Datenbanksysteme - Eine Einführung, 5. Auflage',
        2004, 'Oldenbourg');
INSERT INTO Autor VALUES (NULL, 'Kemper', 'Alfons');
INSERT INTO HatGeschrieben VALUES (LAST_INSERT_ID(), '3486273922');
INSERT INTO Autor VALUES (NULL, 'Eickler', 'André');
INSERT INTO HatGeschrieben VALUES (LAST_INSERT_ID(), '3486273922');
INSERT INTO Exemplar (BibNr, ISBN, ausleihbar)
VALUES ( NULL, '3486273922', 1 ), ( NULL, '3486273922', 1 );

```

4.8 Ausmustern zerstörter Exemplares

Es werden zunächst alle Exemplare mit dem Zustand „auszumustern“ ausgegeben und dann aus der Datenbank gelöscht.

```

SELECT BibNr
FROM Exemplar
WHERE Zustand='auszumustern';
DELETE FROM Exemplar
WHERE Zustand='auszumustern';

```

4.9 Übersicht über Urlaubstage

Ausgabe aller Bibliothekare, die noch über Urlaubstage verfügen.

```

SELECT PersID, Nachname, Vorname

```

```
FROM Bibliothekar  
WHERE Resturlaub NOT NULL  
    AND Resturlaub > 0;
```