

4. ÜZ Musterlösung

Einführung in Datenbanksysteme
Datenbanken für die Bioinformatik

Heinz Schweppe, Jürgen Broß, Manuel Scholz

Übungsaufgaben

1. Aufgabe (Relationaler Algebra)

Gegeben sei folgendes Datenbankschema:

(Beachten Sie: Autor und Spieler sind von Person abgeleitet)

Rollenspiel-DB(Personen (ID : integer, Vorname : string(20), Nachname : string(20), Geburtstag : date),
Spieler (ID : integer, Telefonnummer : integer),
Autoren (ID : integer, e-mail : string(20)),
Abenteuer (ID : integer, Name : string(2), Welt : integer, Spieleranzahl : integer, Autor : integer),
Welten (ID : integer, Name : string(20), Genre : string(20)),
Spielfigur (ID : integer, Vorname : string(20), Nachname : string(20), Alter : integer, Rasse: string, Spieler : integer, Welt : integer)
)

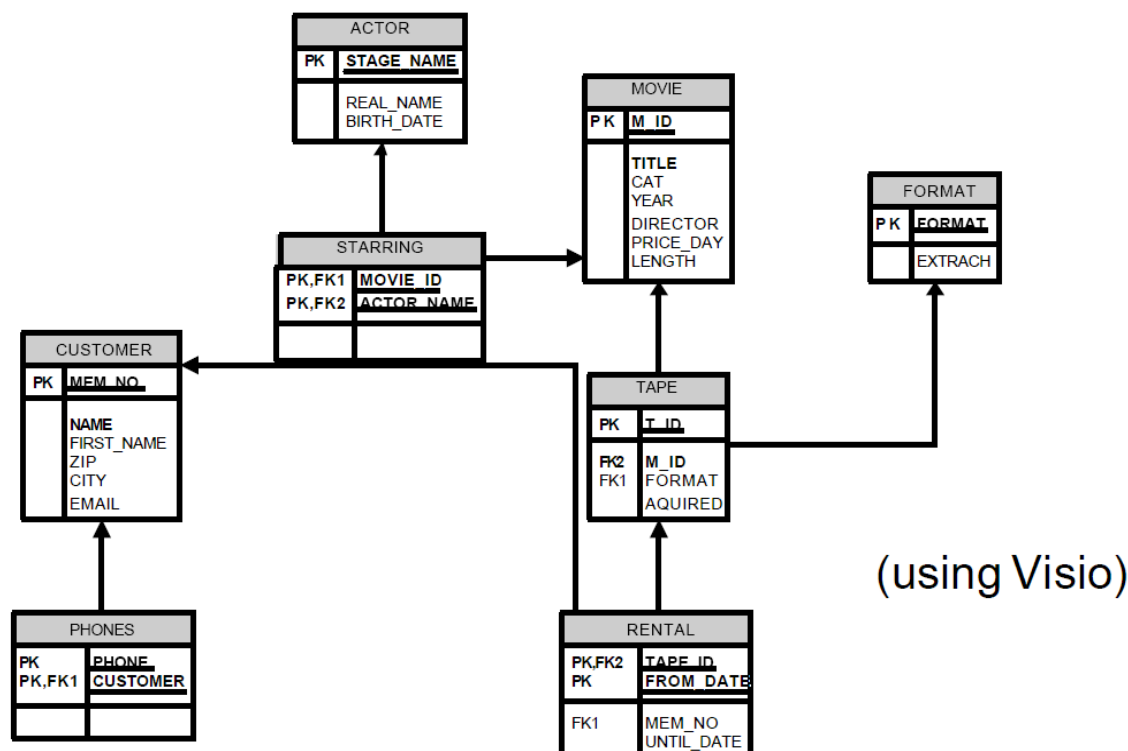
- Interpretieren Sie die folgenden Ausdrücke der relationalen Algebra in natürlicher Sprache.

$$\begin{aligned}
 & \bullet \quad \Pi_{Name, Vorname} \left(\sigma_{Geburtsdag < 12.03.1983} (Person) \right) \\
 & \bullet \quad \Pi_{Name, Vorname} \left(\left(\sigma_{Alter > 120} (Spielfigur) \right) \bowtie_{Spielfigur.Welt = Welten.id} \left(\sigma_{Name = 'DSA'} (Welten) \right) \right) \\
 & \bullet \quad \Pi_{Vorname, Genre} \left(Spielfigur \bowtie_{Spielfigur.Welt = Welten.id} Welten \right. \\
 & \quad \left. \bowtie_{Abenteuer.Welt = Welten.id} \sigma_{Name = 'EinGott greift ein'} (Abenteuer) \right)
 \end{aligned}$$

- Name und Vorname aller Personen die vor dem 12.03.1983 geboren sind.
- Name und Vorname aller Spielfiguren die in der Welt 'DSA' beheimatet sind und älter als 120 Jahre sind.
- Vorname der Spielfiguren, die in einer Welt leben, für die es das Abenteuer 'Ein Gott greift ein' gibt und das Genre dieser Welt.

2. Aufgabe (Relationale Algebra)

Gegeben sei die Videodatenbank aus der Vorlesung.



- Formulieren Sie in relationaler Algebra und SQL folgende Anfragen:
 - Alle Filme (Titel), von denen derzeit alle Exemplare ausgeliehen sind.

c) $\Pi_{title}(Movie) - \Pi_{title}(Movie \mid \times (\Pi_{m_id}(Tape - (Tape \mid \times_{Tape.t_id=Rental.tape_id} (\sigma_{from_date < now < until_date}(Rental))))))$

- Filme die in den letzten 6 Monaten nicht ausgeliehen wurden.

$$\Pi_{title}(Movie) - \Pi_{title}(Movie \mid \times (\Pi_{m_id}(Tape \mid \times_{Tape.t_id=Rental.tape_id} (\sigma_{from_date > now - 180days}(Rental))))))$$

- Namen und Städte der Kunden, die keine Telefonnummer haben.

$$\Pi_{FirstName, Name, City}(Customer - (Customer \mid \times_{Customer.mem_no=Phones.customer} Phones))$$

- Namen aller Filme von denen es keine Tapes mehr gibt.

$$\Pi_{title}(Movie - (Movie \mid \times Tape))$$

- Namen aller Kunden, die aus Berlin kommen.

$$\Pi_{first_name, name}(\sigma_{Customer.city='Berlin'}(Customer))$$

3. Aufgabe (DDL + Constraints)

Die Tabellen *Sportkurs* und *Sportler* werden mit folgenden SQL Befehlen erzeugt:

```
Create table Sportkurs(  
    Titel varchar2(20),  
    Trainer varchar2(10),  
    Dauer_in_min number(3),  
    CONSTRAINT Titel_pk PRIMARY KEY (Titel),  
    CONSTRAINT Dauer_in_min_ck  
    CHECK (Dauer_in_min <= 90)  
);  
  
Create table Sportler(  
    Vorname varchar2(10),  
    Nachname varchar2(10),  
    Geburtsdatum date,  
    Haarfarbe varchar2(10),  
    Sportler_ID number(3),  
    Sportkurs varchar2(20),  
    CONSTRAINT Sportler_ID_pk PRIMARY KEY (Sportler_ID),  
    CONSTRAINT Haarfarbe_ck CHECK (Haarfarbe != 'blau'),  
    CONSTRAINT Sportkurs_fk  
    FOREIGN KEY (Sportkurs) REFERENCES Sportkurs(Titel)  
);
```

a) Welche der folgenden Datensätze können in die (anfänglich leeren) Tabellen eingefügt werden? Einfügbare Datensätze werden beim nächsten Datensatz als in der Datenbank vorhanden betrachtet. Begründen Sie jeweils Ihre Entscheidung, wenn ein Datensatz nicht eingefügt werden kann.

- („Hüpfen für Anfänger“, „Flip“, 45) → Sportkurs
Geht
- („Peter“, „Müller“, „30-JUN-75“, „blond“, 23, „Springen für Doofe“) → Sportler
Sportkurs fehlt
- („Anne“, „Mücke“, „26-MAY-72“, „braun“, 25, „Hüpfen für Anfänger“) → Sportler
Geht
- („Hans“, „Tropopoffof“, „12-AUG-67“, „blond“, 33, „Hüpfen für Anfänger“) → Sportler
Name zu lang
- („Springen für Doofe“, „Hansi“, „90“) → Sportkurs
Geht
- („Michael“, „Dingsbums“, „25-FEB-64“, „blond“, 23, „Hüpfen für Anfänger“) → Sportler
Geht
- („Peter“, „Maier“, „28-SEP-75“, „blond“, 45, „Hüpfen für Anfänger“) → Sportler
Geht
- („Fitness“, „Popeye“, 70) → Sportkurs
Geht
- („Frauke“, „Nitzlaff“, „23-SEP-78“, „blau“, 97, „Fitness“) → Sportler
Haarfarbe verletzt CHECK Constraint

b) Auf welche Kardinalitäten lassen die Tabellendefinitionen schließen? Begründen Sie ihre Ergebnisse.

Ein Sportler kann höchstens einen Sportkurs belegen. Ein Sportkurs kann von beliebig vielen Sportlern belegt werden.

c) Wie müssten die Tabellen umgeformt werden, wenn man folgende Forderungen umsetzen müsste: *Jeder Sportkurs hat genau einen Teilnehmer und kein Sportler darf an zwei Kursen teilnehmen. Für die Sportler_ID sind nur Werte zwischen 1 und 100 erlaubt.*

Geben Sie die resultierenden Tabellen mit allen Constraints an beschreiben Sie ihre Vorgehensweise.

Hinzufügen eines UNIQUE Constraints zu Sportkurs in der Tabelle Sportler oder hinzufügen eines FOREIGN KEY Constraints (NOT NULL) von Sportkurs auf Sportler (genauere Lösung).

- *Hinzufügen eine CHECK CONSTRAINTS für Sportler ID.*
-