Introduction to
Data and Knowledge Engineering
Tutorium: 6 - SQL

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Aufgabe 6.3 - Multiple Choice: SQL und Relationensprachen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Welche der folgenden Operationen wird hier ausgeführt ?
  - Kartesisches Produkt

Begründung:
'SELECT * FROM A, B' kann auch geschrieben werden als
'SELECT A.*, B.* FROM A, B'
Dabei werden also alle Spalten der Tabellen A und B in
die Ergebnistabelle projeziert und keine weiteren Selektions-
Bedingungen angewendet.
Um ein Equi-Join zu erhalten muss zusätzlich in der WHERE-Klausel
bestimmt werden, welche Spalten gleich sein müssen.
Bsp.: 'SELECT * FROM A, B WHERE A.id = B.id'

Aufgabe 6.4 - SQL als DDL

TECHNISCHE UNIVERSITÄT DARMSTADT

- Legen Sie mittels SQL Tabellen für diese vier Schemata an. Überlegen Sie selbst welche Wertebereiche die Attribute jeweils erhalten.
  - CREATE TABLE Hersteller (
    Name varchar(20) NOT NULL,
    Land varchar(20) NOT NULL,
    Stadt varchar(20) NOT NULL,
    PRIMARY KEY (Name)
    );
  - CREATE TABLE Produkt (
    Name varchar(20) NOT NULL,
    Hersteller varchar(20) NOT NULL,
    Preis decimal(6,2) NOT NULL,
    PRIMARY KEY (Name),
    FOREIGN KEY (Hersteller) REFERENCES Hersteller (Name)
    );

Aufgabe 6.4 - SQL als DDL

TECHNISCHE UNIVERSITÄT DARMSTADT

- Legen Sie mittels SQL Tabellen für diese vier Schemata an. Überlegen Sie selbst welche Wertebereiche die Attribute jeweils erhalten.
  - CREATE TABLE Lager (
    ProduktName varchar(20) NOT NULL,
    Groesse int NOT NULL, Anzahl int NOT NULL,
    PRIMARY KEY (ProduktName,Groesse),
    FOREIGN KEY (ProduktName) REFERENCES Produkt(Name)
    );
  - CREATE TABLE Bestellposten (
    Bestellung int NOT NULL, Name varchar(20) NOT NULL,
    Groesse int NOT NULL, Anzahl int NOT NULL,
    PRIMARY KEY (Bestellung,Name,Groesse),
    CONSTRAINT fk_ProduktName FOREIGN KEY (Name) REFERENCES
    Produkt(Name)
    );

Aufgabe 6.4 - SQL als DDL

TECHNISCHE UNIVERSITÄT DARMSTADT

- Einschub: Foreign Keys & MySQL
  - CREATE TABLE Bestellposten (
    Bestellung INT NOT NULL , Name VARCHAR(20) NOT NULL ,
    Groesse INT NOT NULL , Anzahl INT NOT NULL ,
    PRIMARY KEY (Bestellung , Name , Groesse) ,
    CONSTRAINT fk_ProduktName FOREIGN KEY (Name) REFERENCES
    Produkt(Name)
    );
  - Hier muss beachtet werden, dass die Anweisung
    'CONSTRAINT fk_ProduktName' noch zusaetzlich geschrieben werden muss, weil fuer die DROP-Operation fuer FK's immer der interne Symbolname des FK's angegeben werden muss. Dies laesst sich dadurch erklaeren, dass mehrere FK's pro Tabelle definiert werden koennen.

Aufgabe 6.4 - SQL als DDL

TECHNISCHE UNIVERSITÄT DARMSTADT

- Einschub: Foreign Keys & MySQL
  - Bei MySQL wird im DBMS intern die Engine ’MyISAM’ als Standardeinstellung fuer ’CREATE TABLE’ verwendet. Dabei koennen FK’s ohne weiteres angelegt werden. Versucht man allerdings die Tabelle ’Bestellposten’ nun vor der Tabelle ’Hersteller’ zu fuellen, gibt es entgegen der Integritaetsbedingung des FK’s keinen Fehler, da die FK’s in MyISAM nicht verarbeitet werden. Um also tatsaechlich diese Integritaetsbedingung verwenden zu koennen, muss in MySQL die Engine ’InnoDB’ explizit zusaetzlich angeben werden:
  - CREATE TABLE Bestellposten (
    Bestellung INT(10) NOT NULL , Name VARCHAR(20) NOT NULL ,
    Groesse INT(2) NOT NULL , Anzahl INT(10) NOT NULL ,
    PRIMARY KEY (Bestellung , Name , Groesse) ,
    CONSTRAINT fk_ProduktName FOREIGN KEY (Name) REFERENCES
    Produkt(Name)
) ENGINE = INNODB;

Aufgabe 6.4 - SQL als DDL

TECHNISCHE UNIVERSITÄT DARMSTADT

- Entfernen Sie die Attribute des Primary-Key der Tabelle ’Bestellposten’ und legen Sie sie anschliessend wieder an.
  - FK entfernen, PK entfernen, Attribute entfernen:
    ALTER TABLE Bestellposten DROP FOREIGN KEY fk_ProduktName
    ALTER TABLE Bestellposten DROP PRIMARY KEY
    ALTER TABLE Bestellposten DROP COLUMN Bestellung
    ALTER TABLE Bestellposten DROP COLUMN Name
    ALTER TABLE Bestellposten DROP COLUMN Groesse

Aufgabe 6.4 - SQL als DDL

TECHNISCHE UNIVERSITÄT DARMSTADT

- Entfernen Sie die Attribute des Primary-Key der Tabelle ’Bestellposten’ und legen Sie sie anschliessend wieder an.
  - Attribute, PK, FK wieder anlegen:
    ALTER TABLE Bestellposten ADD Bestellung INT NOT NULL
    ALTER TABLE Bestellposten ADD Name VARCHAR(20) NOT NULL
    ALTER TABLE Bestellposten ADD Groesse INT NOT NULL
    ALTER TABLE Bestellposten ADD PRIMARY KEY (Bestellung , Name , Groesse)
    ALTER TABLE Bestellposten ADD CONSTRAINT fk_ProduktName FOREIGN KEY (Name) REFERENCES Produkt(Name)

Aufgabe 6.5 - SQL als DML

TECHNISCHE UNIVERSITÄT DARMSTADT

1. Alle Produkte des Herstellers ’Reebok’ mit der Schuhgrösse 42
   - SELECT P.*, L.Groessse FROM Produkt P, Lager L
     WHERE L.ProduktName = P.Name AND P.Hersteller = ’Reebok’
     AND L.Groesse = 42
2. Schuhgrössen aller Produkte, von denen weniger als 12 Exemplare im Lager vorhanden sind.
   - SELECT ProduktName, Groesse FROM Lager
     WHERE Anzahl < 12
3. Namen aller Hersteller, die nur Produkte unter 125 EUR anbieten (alle Preise werden in EUR angegeben)
   - SELECT P.Hersteller FROM Produkt P
     GROUP BY P.Hersteller HAVING MAX(P.Preis) < 125

Aufgabe 6.6 - SQL als DML

TECHNISCHE UNIVERSITÄT DARMSTADT

1. SELECT * FROM Lager L
   WHERE L.Anzahl > 30 AND L.ProduktName <> 'Evaluate Trainer'
2. SELECT * FROM Produkt P
   WHERE P.Preis < 80
3. SELECT P.Name, P.Preis FROM Produkt P
   WHERE P.Preis > 100

Aufgabe 6.7 - SQL als DML: Aggregation und Sortieren

TECHNISCHE UNIVERSITÄT DARMSTADT

1. Namen aller Produkte, gruppiert nach Hersteller
   - SELECT P.Name, P.Hersteller
     FROM Produkt P ORDER BY P.Hersteller
2. Gesamtwert aller verfügbaren Waren im Lager
   - SELECT SUM(L.Anzahl * P.Preis) AS LagerSumme
     FROM Lager L, Produkt P WHERE L.ProduktName = P.Name