GDB [HA] zum 12. 12. 2013

Tim Dobert, Kai Sonnenwald, Arne Struck

12. Dezember 2013

1.: a)

$$\pi_{Sorte}(\sigma_{Vorname=''Horst''}(Person) \underset{PNR=Entdecker}{\bowtie} Obst)$$

b)

$$\pi_{Vorname,Nachname}(\sigma_{Symptom=''Halskratzen''}(Allergie) \underset{Person=PNR}{\bowtie} Person)$$

c)

$$\pi_{Sorte,Nachname}(\sigma_{Symptom=''Würgereiz''}(Allergie) \underset{Person=PNR}{\bowtie} Person \underset{PNR=Entdecker}{\bowtie} Obst)$$

2.: a)

Erstellen der Rennort-Tabelle:

```
CREATE TABLE Rennort (
        OID             int,
        Name            varchar(30) NOT NULL,
        Strecke         varchar(50) NOT NULL,
        CONSTRAINT      pk_rennort PRIMARY KEY (OID));
```

Erstellen der Rennstall-Tabelle:

```
CREATE TABLE Rennstall (
        RSID            int,
        Name            varchar(15) NOT NULL,
        Teamchef        varchar(50),
        Budget          int NOT NULL UNIQUE,
        CONSTRAINT pk_rennstall PRIMARY KEY (RSID),
        CONSTRAINT ck_budget CHECK(500>Budget>0));
```

Erstellen der Rennfahrer-Tabelle:

```
CREATE TABLE Rennfahrer (
        RID             int,
        Vorname         varchar(20) NOT NULL,
        Nachname        varchar(30) NOT NULL,
        Geburt          date NOT NULL,
        Wohnort         varchar(50),
        Rennstall       int NOT NULL,
        CONSTRAINT pk_rennfahrer PRIMARY KEY (RID),
        CONSTRAINT fk_rennstall FOREIGN KEY (Rennstall) REFERENCES Rennst
```

Erstellen der Platzierung-Tabelle:

```
CREATE TABLE Platzierung (
        RID             int NOT NULL,
        OID             int NOT NULL,
        Platz           int NOT NULL,
        CONSTRAINT pk_platzierung PRIMARY KEY (RID, OID),
        CONSTRAINT fk_rid FOREIGN KEY (RID) REFERENCES Rennfahrer (RID),
        CONSTRAINT fk_oid FOREIGN KEY (OID) REFERENCES Rennort (OID));
```

b)

Die sofortige Prüfung der referentiellen Integrität bedeutet, dass zwingend die richtige Reihenfolge eingehalten werden muss um Fehler bei Verweisen auf Fremdschlüsseln zu vermeiden. Mit dem zusätzlichen Attribut Star refrenzieren sich die Tabellen Rennfahrer und Rennstall gegenseitig. Damit muss man beim Anlegen der Tabellen schrittweise vorgehen und einen der Fremdschlüssel nachträglich hinzufügen. Bei der Manipulation der Daten per DML kann es dazukommen, dass sich Zeilen aufgrund gegenseitiger Verweise nicht entfernen lassen, es sei denn man setzt erst Star auf null.

c)

Füllen der Rennort-Tabelle:

```
INSERT INTO Rennort
VALUES (4, "Australien GP", "Albert Park Circuit");
INSERT INTO Rennort
VALUES (15, "Malaysia GP", "Sepang International Circuit");
INSERT INTO Rennort
VALUES (21, "China GP", "Shanghai International Circuit");
```

Füllen der Rennstall-Tabelle:

```
INSERT INTO Rennstall
VALUES (2,"Red Bull","Christian Horner",370);
INSERT INTO Rennstall
VALUES (5,"Ferrari","Stefano Domenicali",350);
INSERT INTO Rennstall
VALUES (31,"McLaren","Martin Whitmarsh",220);
INSERT INTO Rennstall
VALUES (34,"Lotus F1","Eric Boullier",100);
```

Füllen der Rennfahrer-Tabelle:

```
INSERT INTO Rennfahrer
VALUES (4,"Sebastian","Vettel","1987-07-03","Kemmental(Schweiz)",2);
INSERT INTO Rennfahrer
VALUES (6,"Fernando","Alonso","1981-07-29","Lugano(Schweiz)",5);
INSERT INTO Rennfahrer
VALUES (8,"Marc","Webber","1976-08-27","Aston Clinton(UK)",2);
INSERT INTO Rennfahrer
VALUES (9,"Lewis","Hamilton","1985-01-07","Genf(Schweiz)",31);
INSERT INTO Rennfahrer
VALUES (20,"Jenson","Button","1980-01-19","Monte Carlo(Monaco)",31);
```

```
INSERT INTO Rennfahrer
VALUES (21,"Felipe","Massa","1982-04-25","Sao Paulo(Brasilien)",5);
INSERT INTO Rennfahrer
VALUES (44,"Kimi",,"1979-10-17","Espoo(Finnland)",34);
```

Füllen der Platzierung-Tabelle:

```
INSERT INTO Platzierung
VALUES (8,4,6);
INSERT INTO Platzierung
VALUES (4,15,1);
INSERT INTO Platzierung
VALUES (20,15,17);
INSERT INTO Platzierung
VALUES (4,4,3);
INSERT INTO Platzierung
VALUES (6,4,2);
INSERT INTO Platzierung
VALUES (8,15,2);
INSERT INTO Platzierung
VALUES (6,21,1);
INSERT INTO Platzierung
VALUES (9,4,5);
INSERT INTO Platzierung
VALUES (21,15,5);
INSERT INTO Platzierung
VALUES (20,4,9);
INSERT INTO Platzierung
VALUES (21,4,4);
```

d)

Rennfahrer löschen die mit "F"beginnen:
Erster Ansatz:

```
DELETE FROM Rennfahrer WHERE Vorname LIKE "F%";
```

Funktioniert allerdings nicht einfach so, da es Foreign Key Verweise aus der Platzierungstabelle auf diese Rennfahrer gibt. Diese müssen zuerst gelöscht werden.

```
DELETE FROM Platzierung
WHERE RID IN     (SELECT RID
                  FROM Rennfahrer
                  WHERE Vorname LIKE "F%");

DELETE FROM Rennfahrer WHERE Vorname LIKE "F%";
```

Alle Tabellen löschen:
Dabei muss besonders auf die korrekte Reihenfolge geachtet werden, da es sonst zu Verletzungen der referentiellen Integrität kommt.

```
DROP TABLE Platzierung;
```

```
DROP TABLE Rennfahrer;
DROP TABLE Rennstall;
DROP TABLE Rennort;
```

3.: Annahme, dass die Tabellen entsprechend des Relationenschemas aus Aufgabe 1 vorliegen.

a)

Alle Obstsorten, gegen die ein Peter Meyer allergisch ist, ohne Duplikate in absteigender Sortierung:

```
SELECT  DISTINCT o.Sorte
FROM    Obst o, Person p, Allergie a
WHERE   a.Obst = o.ONR
AND     a.Person = p.PNR
AND     p.Vorname = "Peter"
AND     p.Nachname = "Meyer"
ORDER BY o.Sorte DESC;
```

b)

Die PNR, den Nachnamen und die Anzahl der Allergien jedes Allergikers:

```
SELECT  p.PNR, p.Nachname, COUNT(a.PNR)
FROM    Person p, Allergie a
WHERE   p.PNR = a.PNR;
```

c)

Die PNR aller Personen, die mehr als 6 Obstsorten entdeckt haben:

```
SELECT  p.PNR
FROM    Person p, Obst o
WHERE   p.PNR = o.PNR AND (COUNT(o.Entdecker)>6);
```

d)

Vorname und Nachname aller Personen, deren Lieblingsobst von einer Person gleichen Vornamens entdeckt wurde:

```
SELECT  p.Vorname, p.Nachname
FROM    Person p, Obst o
WHERE   p.Lieblingsobst = o.ONR
AND     p.Vorname IN    (SELECT p2.Vorname
                         FROM Person p2, Obst o2
                         WHERE o2.Entdecker = p2.PNR
                         AND p.Lieblingsobst = o2.ONR);
```

e)

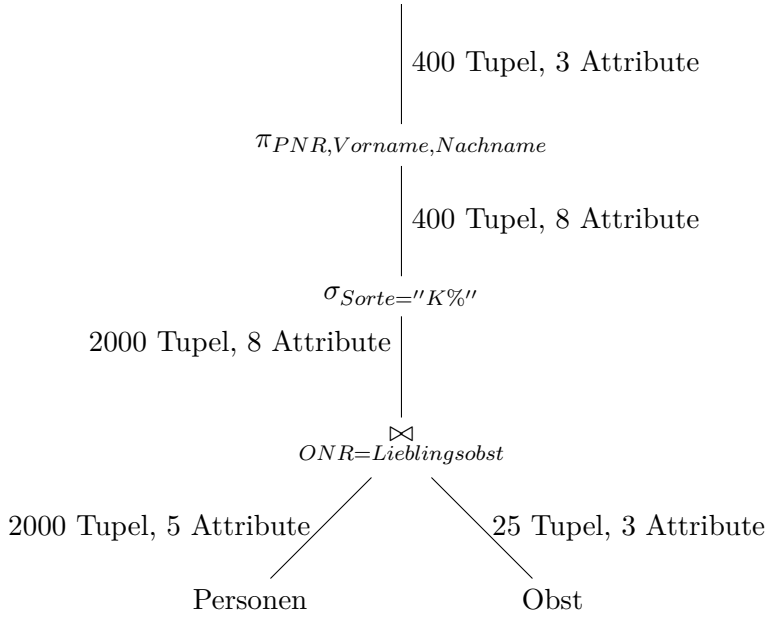
Die PNR, Vorname und Nachname aller Personen, die noch keine Obstsorte entdeckt haben:

```
SELECT  p.PNR, p.Vorname, p.Nachname
FROM    Person p, Obst o
WHERE   p.PNR NOT IN    (SELECT p2.PNR
                         FROM Person p2, Obst o2
                         WHERE p2.PNR = o.Entdecker);
```

4.:

$$\pi_{PNR,Vorname,Nachname}(\sigma_{Sorte=''K\%''}(Obst) \underset{ONR=Lieblingsobst}{\bowtie} Personen)$$

Naiver Operatorbaum:



Optimierter Operatorbaum: