2. SQL und relationale Algebra

Gegeben sei der folgende Ausschnitt aus dem Schema einer Schulverwaltung:

```
Person : {[
  ID : INTEGER,
  Name : VARCHAR(255),
  Wohnort : VARCHAR(255),
  Typ : CHAR(1)
]}

Unterricht : {[
  Klassenbezeichnung : VARCHAR(20),
  Schuljahr : INTEGER,
  Lehrer : INTEGER,
  Fach : VARCHAR(100)
]}

Klasse : {[
  Klassenbezeichnung : VARCHAR(20),
  Schuljahr : INTEGER,
  Klassenlehrer : INTEGER
]}

Klassenverband : {[
  Schüler : INTEGER,
  Klassenbezeichnung : VARCHAR(20),
  Schuljahr : INTEGER
]}
```

```sql
CREATE TABLE Person (
  ID INTEGER PRIMARY KEY,
  Name VARCHAR(255),
  Wohnort VARCHAR(255),
  Typ CHAR(1) CHECK(Typ in ('S', 'L'))
);

CREATE TABLE Klasse (
  Klassenbezeichnung VARCHAR(20),
  Schuljahr INTEGER,
  Klassenlehrer INTEGER REFERENCES Person(ID),
  PRIMARY KEY (Klassenbezeichnung, Schuljahr)
);

CREATE TABLE Klassenverband (
  Schüler INTEGER REFERENCES Person(ID),
  Klassenbezeichnung VARCHAR(20),
  Schuljahr INTEGER,
  PRIMARY KEY (Schüler, Schuljahr)
);

CREATE TABLE Unterricht (
  Klassenbezeichnung VARCHAR(20),
  Schuljahr INTEGER,
  Lehrer INTEGER REFERENCES Person(ID),
```

```
  Fach VARCHAR(100),
  CONSTRAINT Unterricht_PK
    PRIMARY KEY (Klassenbezeichnung, Schuljahr, Lehrer, Fach)
);

INSERT INTO Person VALUES
  (1, 'Lehrer Ludwig', 'München', 'L'),
  (2, 'Schüler Max', 'München', 'S'),
  (3, 'Schülerin Maria', 'München', 'S'),
  (4, 'Schülerin Eva', 'Starnberg', 'S'),
  (5, 'Lehrerin Walter', 'München', 'L');

INSERT INTO Klasse VALUES
  ('1a', 2015, 1),
  ('1a', 2014, 1),
  ('1b', 2015, 5);

INSERT INTO Klassenverband VALUES
  (2, '1a', 2015),
  (3, '1a', 2015),
  (4, '1b', 2015);
```

Hierbei enthält die Tabelle *Person* Informationen über Lehrer (Typ 'L') und Schüler (Typ 'S'); andere Werte für Typ sind nicht zulässig. *Klasse* beschreibt die Klassen, die in jedem Schuljahr gebildet wurden, zusammen mit ihrem Klassenlehrer. In *Unterricht* wird abgelegt, welcher Lehrer welches Fach in welcher Klasse unterrichtet; es ist möglich, dass derselbe Lehrer mehr als ein Fach in einer Klasse unterrichtet. *Klassenverband* beschreibt die Zuordnung der Schüler zu den Klassen.

(a) Schreiben Sie eine SQL-Anweisung, die die Tabelle *Unterricht* mit allen ihren Constraints (einschließlich Fremdschlüsselconstraints) anlegt.

```
CREATE TABLE IF NOT EXISTS Person(
  ID INTEGER PRIMARY KEY,
  Name VARCHAR(255),
  Wohnort VARCHAR(255),
  Typ CHAR(1) CHECK(Typ in ('S', 'L'))
);

CREATE TABLE IF NOT EXISTS Klasse(
  Klassenbezeichnung VARCHAR(20),
  Schuljahr INTEGER,
  Klassenlehrer INTEGER REFERENCES Person(ID),
  PRIMARY KEY (Klassenbezeichnung, Schuljahr)
);

CREATE TABLE IF NOT EXISTS Unterricht (
  Klassenbezeichnung VARCHAR(20) REFERENCES
  ↪ Klasse(Klassenbezeichnung),
  Schuljahr INTEGER REFERENCES Klasse(Schuljahr),
  Lehrer INTEGER REFERENCES Person(ID),
  Fach VARCHAR(100),
  CONSTRAINT Unterricht_PK
    PRIMARY KEY (Klassenbezeichnung, Schuljahr, Lehrer, Fach)
);
```

(b) Definieren Sie ein geeignetes Constraint, das sicherstellt, dass nur zulässige Werte im Attribut Typ der (bereits angelegten) Tabelle *Person* eingefügt werden können.

Ich habe `REFERENCES` bei Unterricht Schuljahr vergessen, die referenzierten Tabellen `Person` und `Klasse` wurden in der Musterlösung auch nicht angelegt.

```
ALTER TABLE Person
  ADD CONSTRAINT TypLS
    CHECK(Typ IN ('S', 'L'));
```

(c) Schreiben Sie eine SQL-Anweisung, die die Bezeichnung der Klassen bestimmt, die im Schuljahr 2015 die meisten Schüler haben.

Falsch: `ORDER BY Anzahl;`. `DESC` vergessen.

```
SELECT k.Klassenbezeichnung, COUNT(*) AS Anzahl
FROM Klasse k, Klassenverband v
WHERE
  k.Schuljahr = 2015 AND
  k.Klassenbezeichnung = v.Klassenbezeichnung
GROUP BY k.Klassenbezeichnung
ORDER BY COUNT(*) DESC;
```

(d) Schreiben Sie eine SQL-Anweisung, die die Namen aller Lehrer bestimmt, die nur Schüler aus ihrem Wohnort unterrichtet haben.

```
SELECT DISTINCT l.Name
FROM Person l
WHERE NOT EXISTS(
  SELECT DISTINCT *
  FROM Unterricht u, Klassenverband v, Person s
  WHERE
    u.Lehrer = l.ID AND
    u.Klassenbezeichnung = v.Klassenbezeichnung AND
    v.Schüler = s.ID AND
    l.Wohnort != s.Wohnort
);
```

(e) Schreiben Sie eine SQL-Anweisung, die die Namen aller Schüler bestimmt, die immer den gleichen Klassenlehrer hatten.

```
SELECT s.Name
FROM Person s, Klasse k, Klassenverband v
WHERE
  s.ID = v.Schueler AND
  v.Klassenbezeichnung = k.Klassenbezeichnung AND
  v.Schuler = s.ID
GROUP BY k.Klassenlehrer, v.Schueler
HAVING COUNT(*) = 1
```

(f) Schreiben Sie eine SQL-Anweisung, die alle Paare von Schülern bestimmt, die mindestens einmal in der gleichen Klasse waren. Es genügt dabei, wenn Sie die ID der Schüler bestimmen.

Ich habe Schuljahr zum joinen vergessen.

```
SELECT DISTINCT s1.ID, s2.ID
FROM Klassenverband s1, Klassenverband s2
WHERE
```

```
4     s1.Schuljahr = s2.Schuljahr AND
5     s1.Schueler <> s2.Schueler AND
6     s1.Klassenbezeichung = s2. Klassenbezeichnung;
```

(g) Formulieren Sie eine Anfrage in der relationalen Algebra, die die ID aller Schüler bestimmt, die mindestens einmal von „Ludwig Lehrer“ unterrichtet wurden.

$$\pi_{\text{Schüler}}\Big(\sigma_{\text{Name}='\text{Ludwig Lehrer}'}(\text{Person}) \bowtie_{\text{Person.ID}=\text{Unterricht.Lehrer}} \text{Unterricht} \bowtie_{\text{Unterricht.Klassenbezeichnung}=\text{Klassenverband.Klassenbezeichnung}} \text{Klassenverband}\Big)$$

(h) Formulieren Sie eine Anfrage in der relationalen Algebra, die Namen und ID der Schüler bestimmt, die von allen Lehrern unterrichtet wurden.

$$\pi_{\text{Name,ID}}\Big(\big(\pi_{\text{Lehrer,Schueler}}(\text{Unterricht} \bowtie \text{Klassenverband}) \div \pi_{\text{ID}}(\sigma_{\text{Typ}='\text{L}'}(\text{Person}))\big) \bowtie \text{Person}\Big)$$

Beachten Sie bei der Formulierung der SQL-Anfragen, dass die Ergebnisrelationen keine Duplikate enthalten dürfen. Sie dürfen geeignete Views definieren.