Datenbanken
Blatt 4

Gruppe 26

Markus Vieth Christian Stricker

24. Mai 2016

1 Aufgabe

1.1

1.1.1

```
{[a.Firstname, a.Lastname, d.Firstname, d.Lastname] | a, d ∈ person
    ∧ ∃ x ∈ directs (d.PersonId = x.PersonId
        ∧ ∃ y ∈ acts (a.PersonId = y.PersonId
            ∧ x.MovieId = y.MovieId
        )∨ ∃ x2 ∈ acts (d.PersonId = x2.PersonId
            ∧ ∃ y ∈ acts (a.PersonId = y.PersonId
                ∧ x2.MovieId = y.MovieId
        )
    ) ∧ ¬ (
        ¬(d.Firstname = 'Lars' ∧ d.Lastname = 'von Trier')
        ∧ ¬(d.Firstname = 'Michael' ∧ d.Lastname = 'Haneke')
    )
}
```

1.1.2

```
{[c.name, c.location, s2.time, s2.price, m.title] | c ∈ cinema
    ∧ ∃ r ∈ screeningRoom (r.inCinema = c.CinemaId
        ∧ ∃ s1, s2 ∈ screening (s1.RoomId = r.RoomId
            ∧ ∃ m ∈ movie (s1.MovieId = m.MovieId ∧ s2.MovieId = m.MovieId
                ∧ s1.date = Today ∧ s1.time = '9:00'
            )
        )
    )
}
```

1.1.3

```
{[a.Firstname, a.Lastname] | a, d ∈ person
    ∧ ∀ m ∈ movies (d.Firstname = 'Lars' ∧ d.Lastname = 'von Trier'
        ∧ ∃ x ∈ directs (m.MovieId = x.MovieId
            ∧ d.PersonId = x.PersonId
        ) ⇒ (
            ∃ y ∈ acts (m.MovieId = y.MovieId
                ∧ a.PersonId = y.PersonId
            )
        )
    ) ∧ ∃ t ∈ acts (t.PersonId = a.PersonId)
}
```

1.2

1.2.1

```
{ [afn, aln, dfn, dln] |
    ∃ dPid ( [dPid, dfn, dln] ∈ Person
        ∧ ∃ aPid ([aPid, afn, aln] ∈ Person
            ∧ ∃ aMid, r ([aMid, aPid, r] ∈ acts
                ∧ ∃ dMid ([dMid, dPid] ∈ directs
                    ∨ ∃ role ([dMid, dPid, role] ∈ acts
                        ∧ ¬(
                            ¬(dfn = '' ∧ dln='' )
                            ∧ ¬(dfn = '' ∧ dln='' )
                        )
                    )
                )
            )
        )
    )
}
```

1.2.2

```
{ [n, l, time, p, t] |
    ∃ cId ( [cId, n ,l] ∈ Cinema
        ∧ ∃ rId, num ([rId, num, cId] ∈ screeningRoom
            ∧ ∃ mId, d, len ([mId, rId, time, d, len, p] ∈ screening
                ∧ ∃ time2, d2, len2, p2, rId2 ([mId, rId2, time2, d2, len2, p2] ∈ screening
                    ∧ ∃ lang, fsk, y ([mId, t, lang, fsk, y] ∈ Movie
                        ∧ time2 = (21*60)
                        ∧ d2 = today
                    )
                )
            )
        )
    )
}
```

1.2.3

```
{ [afn, aln] |
     ∃ aPid ( [aPid, afn, aln] ∈ Person
        ∧∀ mId (
            ∃ dPid, dfn, dln ( [dPid, dfn, dln] ∈ Person
                ∧ [dPid, mId] ∈ directs
                ∧ dfn = 'Vorname'
                ∧ dln = 'Nachname'
            ) ⇒ (
                ∃ r (
                    [aPid, mId, r] ∈ acts
                )
            )
        ) ∧ ∃ mId2, r2 (
            [aPid, mId2, r2] ∈ acts
        )
    )
}
```

2 Aufgabe

2.1 Beispiel

$$\{n|\neg(n \in \text{Professoren})\}$$

2.2 Erklärung

Man definiert die Domäne der Formel des Ausdrucks. Dies ist eine Menge, welche folgende Elemente beinhaltet:

- alle in der Formel vorkommende Konstanten
- alle Attributwerte von Relationen, welche in der Formel referenziert werden

Das Ergebnis des Ausdrucks muss nun Teilmenge dieser Domäne sein, um sicher zu stellen, dass das Ergebnis der Anfrage endlich ist.

3 Aufgabe

$$\{n|\neg(n \in \text{Professoren})\}$$

Eine unsichere Anfrage wie in Aufgabe 2, kann in der relationalen Algebra nicht konstruiert werden. Wie der Name schon sagt, beschreibt die relationale Algebra Rechenoperationen auf Relationen, welche an sich bereits endlich sind. Alle Operatoren der relationalen Algebra ermöglichen höchstens das vervielfachen der Anzahl der betrachteten Elemente, aber nie das Wachstum ins unendliche.

4 Aufgabe

4.1

```
select m.Title
from movie m, directs d, person p
where m.MovieId = d.MovieId
    and d.PersonId = p.PersonId
    and p.Lastname = "Murnau"
    and p.Firstname = "Friedrich Wilhelm"
```

4.2

```
select m.Title
from movie m, directs d, person p
where m.MovieId = d.MovieId
    and d.PersonId = p.PersonId
    and p.Lastname = 'Burton'
    and p.Firstname = 'Tim'
    and m.MovieId not in (
        select a.MovieId
        from acts a, person pjd
        where a.PersonId = pjd.PersonId
            and pjd.Lastname = 'Depp'
            and pjd.Firstname = 'Johnny')
```

4.3

```
select m.Title, c.Name, c.Location, s.Time, s.Price
from movie m, cinema c, screening s, genre g, hasGenre hg, screeningRoom sr
where s.Date = Today()
    and s.MovieId = m.MovieId
    and m.MovieId = hg.MovieId
    and hg.GenreId = g.GenreId
    and g.Name = 'Documentary'
    and s.RoomId = sr.RoomId
    and sr.inCinema = c.CinemaId
```

4.4

```
Select a.Firstname, a.Lastname, d.Firstname, d.Lastname
from person a, person d
where
    exists(
        select *
        from  directs x
        where d.PersonId = x.PersonId
        and exists(
            select *
            from acts y
            where a.PersonId = y.PersonId
                and  x.MovieId = y.MovieId
            )or  exists(
                select *
                from acts x2
                where d.PersonId = x2.PersonId
                and exists(
                    select *
                    from acts y
                    where a.PersonId = y.PersonId
                        and x2.MovieId = y.MovieId
```

```
22                )   and not (
23                not(d.Firstname = "Lars" and d.Lastname = "von Trier")
24                and not(d.Firstname = "Michael" and d.Lastname = "Haneke")
25            )
26        )
27    )
```

4.5

```
1   Select c.Name, c.Location, s.Price, m.Title
2   from cinema c, screening s, screeningRoom r, movie m
3   where m.MovieId = s.MovieId
4       and s.RoomId = r.RoomId
5       and c.Cinemaid = r.inCinema
6       and exists (
7           Select m.MovieId = s.MovieId
8           and s.RoomId = r.RoomId
9           and c.Cinemaid = r.inCinema
10          and s.Date = Today()
11          and s.Time = 2100
12  )
```