

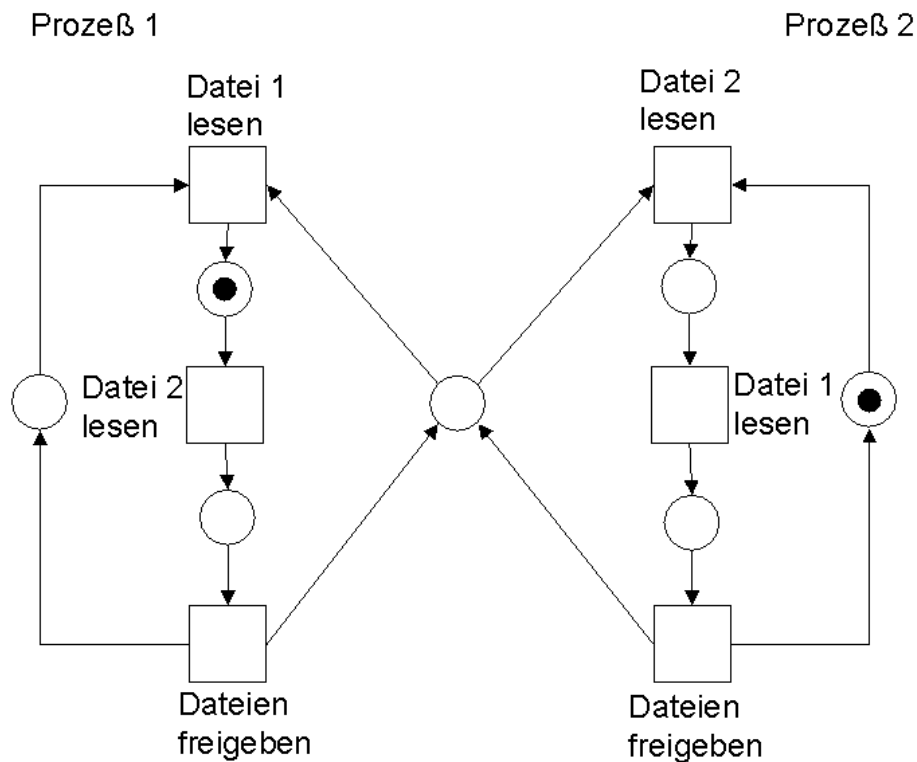


Musterlösungen zum Übungsblatt 6

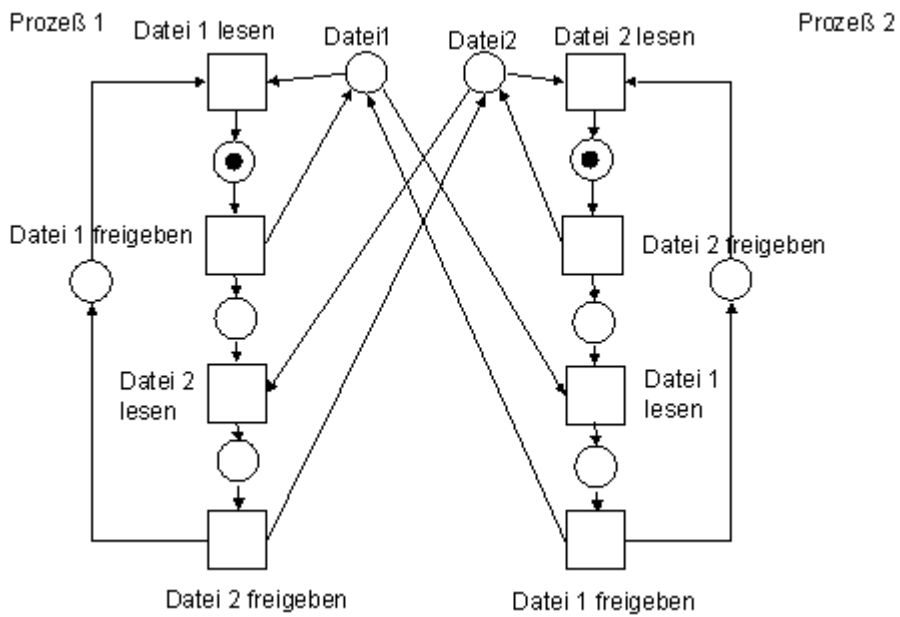
Lösung

1. Petrinetz - Auflösung von Verklemmungen (1,5 Punkte)

a)

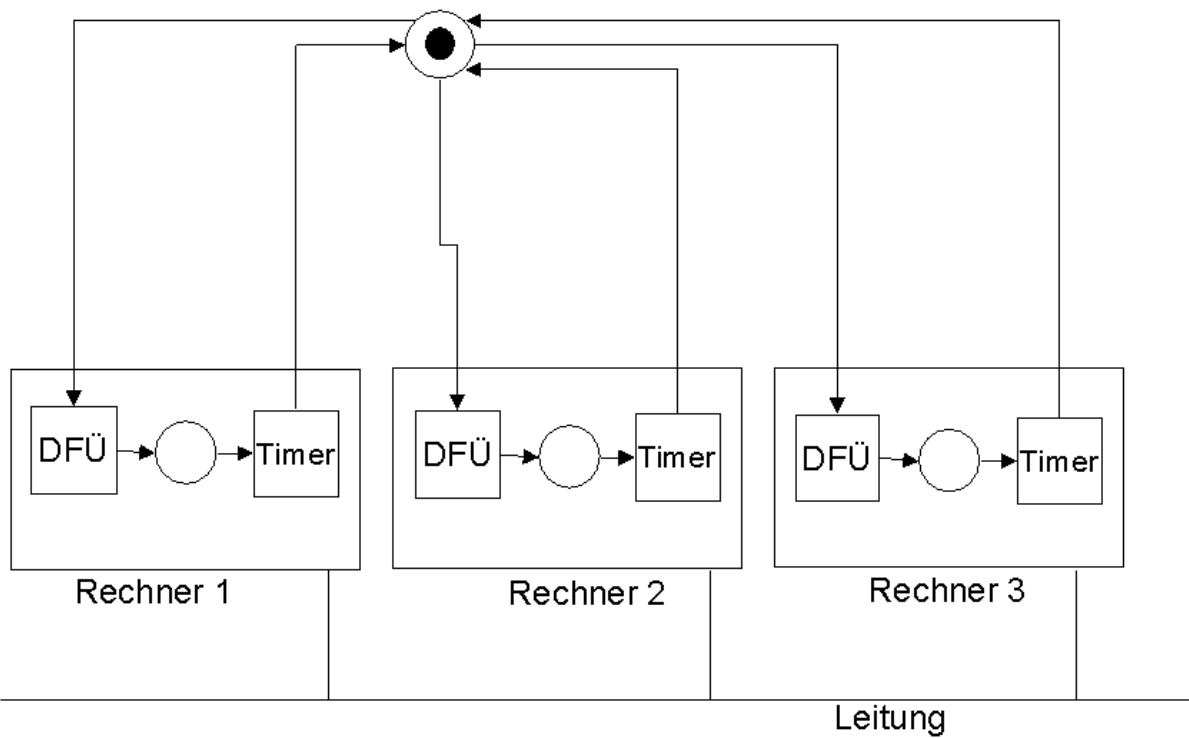


b)

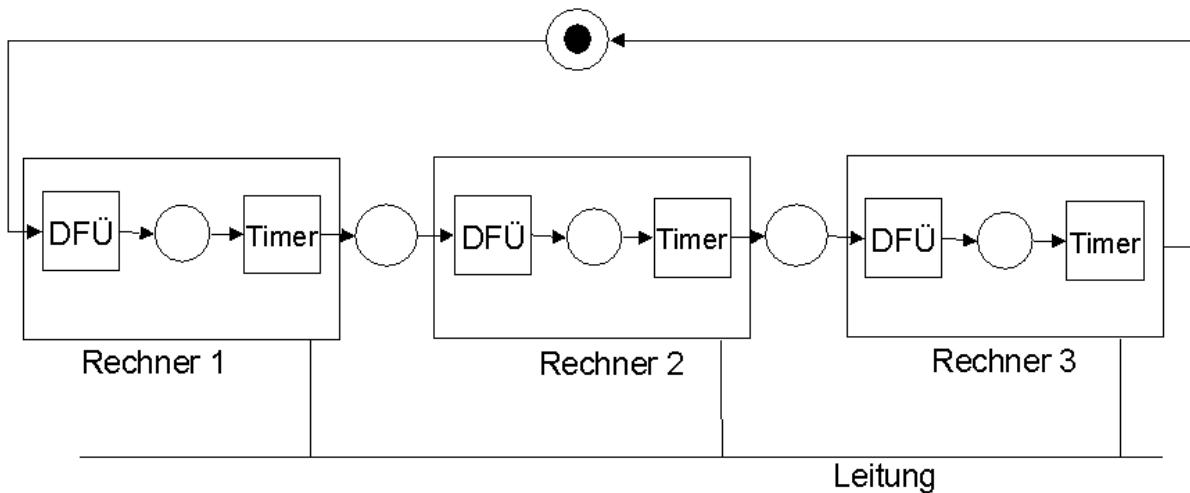


Lösung

2. Petrinetz – Rechnersteuerung (1 Punkt)



b)



Lösung

3. Relationale Algebra - Studierenden-Datenbestand (6,5 Punkte)

a) Im folgenden seien die Schemata und wie folgt definiert:

$R_1 = \{\text{Name:CHARACTER}(20), \text{Vorname:CHARACTER}(10), \text{Matr_nr:INTEGER}, \text{Fach:CHARACTER}(3), \text{Semester:INTEGER}, \text{Tutor:CHARACTER}(4), \text{Tutorium:INTEGER}\}$

$R_2 = \{\text{Name:CHARACTER}(20), \text{Vorname:CHARACTER}(10), \text{Matr_nr:INTEGER}, \text{Fach:CHARACTER}(3), \text{Semester:INTEGER}\}$

b) Die benötigte Datenbankabfrage ist eine Projektion, bei der für jeden Datensatz lediglich die Semesterzahl ausgegeben wird.

Datenbankoperation:

Es sei π_{Semester} Unterschema zum Schema der Relation Informatikstudierende mit

$\pi_{\text{Semester}} = \{\text{Semester:INTEGER}\}$

Dann erzeugt die Projektion P (Informatikstudierende) die gewünschte Relation.

SQL-Abfrage:

SELECT Semester FROM Informatikstudierende;

Hier wird also nur die Spalte "Semester" der Relation "Informatikstudierende" ausgegeben.

Semester
1
1

2
3
3
1
1
4
3
5
7
9

c) Hier ist ein natürlicher Verbund durchzuführen, der aus den Relationen Informatikstudierende und Biologiestudierende eine neue Relation erzeugt, in der alle Studierenden enthalten sind, die auch in *beiden* Ausgangsrelationen enthalten sind.

Datenbankoperation:

Achtung: Obwohl in den Schemata und jeweils ein Attribut mit dem gleichen Namen "Fach" vorkommt, handelt es sich hier dennoch nicht um ein und das selbe Attribut. Es darf deshalb auch nicht in das gemeinsame Unterschema ϕ der Schemata und aufgenommen werden, da sonst eine leere Relation das Ergebnis wäre. Bei dem im Rahmen des Joins durchzuführenden Kreuzprodukt existieren dann zwei Attribute für Fach (beispielsweise bio.Fach und info.Fach).

Es sei ϕ Unterschema zu und mit

$\phi = \{ \text{Name:CHARACTER(20), Vorname:CHARACTER(10), Matr_nr:INTEGER, Semester:INTEGER} \}$

Die durchzuführende Datenbankoperation ist dann:
 Doppel := Informatikstudierende \bowtie_{ϕ} Biologiestudierende

Das Schema von Doppel ist dann:

={ Name:CHARACTER(20), Vorname:CHARACTER(10), Matr_nr:INTEGER, Informatikstudierende.Fach:CHARACTER(3), Biologiestudierende.Fach:CHARACTER(3), Semester:INTEGER, Tutor:CHARACTER(4), Tutorium:INTEGER }

Damit die Relation Doppel nicht zu unübersichtlich wird, wird sie anschließend noch auf ϕ projiziert.

Das endgültige Ergebnis ist dann also:

$P_{\phi}(\text{Doppel})$.

SQL-Abfrage:

SELECT info.Name, info.Vorname, info.Matr_nr, info.Semester FROM Informatikstudierende AS info, Biologiestudierende AS bio WHERE info.Matr_nr = bio.Matr_nr;

Aus der Relation "Informatikstudierende" werden jene Studierende herausgesucht, deren Matrikelnummer auch in der Relation "Biologiestudierende" auftaucht. Von ihnen wird Name, Vorname, Matrikelnummer und Semester ausgegeben.

Tabelle:

Name	Vorname	Matr_nr	Semester
Meier	Hans	123456	1
Mülle	Susanne	123452	1

Bemerkung: Die Studentin Karin Schulz absolviert kein Doppelstudium. Es gibt zwei Personen mit diesem Namen, die verschiedene Matrikelnummern haben.

d) Die Aufgabenstellung kann durch Vereinigung gelöst werden.

Datenbankoperation:

De beiden Relationen Informatikstudierende und Biologiestudierende lassen sich nicht direkt miteinander vereinigen, da sie nicht kompatibel zueinander sind.

Projiziert man die Relation Informatikstudierende jedoch auf das Schema der Biologiestudierenden, um Kompatibilität zu erreichen, so stehen im Ergebnis der Vereinigung jene Studenten doppelt, die ein Doppelstudium absolvieren, da sich ihre Datensätze durch das Attribut "Fach" in beiden Relationen unterscheiden.

Also werden beide Relationen auf das Schema ϕ aus Teilaufgabe b) projiziert und dann vereinigt.

Es sei

Infos = P (Informatikstudierende)

Bios = P (Biologiestudierende).

Dann erzeugt die Operation

Alle := Infos \cup Bios

die gewünschte Relation "Alle" in, der alle Informatik- und Biologiestudierenden enthalten sind.

SQL - Abfrage:

SELECT Name, Vorname, Matr_nr, Semester FROM Informatikstudierende UNION SELECT Name, Vorname, Matr_nr, Semester FROM Biologiestudierende;

Die SQL-Abfrage gibt aus beiden Tabellen zu jedem Studierenden Name, Vorname, Matrikelnummer und Semester aus. Studierende die in beiden Tabellen vorkommen werden nur einmal ausgegeben.

Name	Vorname	Matr_nr	Semester
Schulz	Karin	123451	1

Mülle	Susanne	123452	1
Schmidt	Dieter	123453	2
Maier	Klaus	123454	3
Kehl	Maria	123455	3
Meier	Hans	123456	1
Hofmann	Peter	123457	1
Kerner	Dirk	123458	4
Bachmann	Jan	123459	3
Schwarz	Thomas	123464	5
Wolf	Kirsten	123465	7
Mung	Manfred	123466	9
Müller	Peter	654321	3
Schulz	Karin	554321	1
Hoffmann	Doris	454321	2
Peter	Andreas	354321	1
Wehner	Christof	254321	3
Wallner	Peter	154321	1

e) Zur Ausblendung der Matrikelnummer muss die Relation Biologiestudierende auf ein Schema projiziert werden, welches das Attribut Mat_nr nicht enthält.

Datenbankoperation:

= {Name:CHARACTER(20), Vorname:CHARACTER(10), Semester:INTEGER }

Sinnvoller Weise enthält auch das Attribut "Fach" nicht, da dies bei allen Biologiestudierenden gleich ist. P (Biologiestudierende) liefert dann Informationen über jeden Biologiestudenten, ohne dabei seine Matrikelnummer und sein Studienfach auszugeben.

SQL - Abfrage:

SELECT Name, Vorname, Semester FROM Biologiestudierende;

Aus der Relation "Biologiestudierende" werden gerade die Spalten Name, Vorname und Semester ausgewählt.

Tabelle:

Name	Vorname	Semester
------	---------	----------

Meier	Hans	1
Müller	Peter	3
Schulz	Karin	1
Mülle	Susanne	1
Hoffmann	Doris	2
Peter	Andreas	1
Wehner	Christof	3
Wallner	Peter	1

f) Die gestellte Frage lässt sich mit Hilfe einer Selektion beantworten.

Datenbankoperation:

$sel_{\text{Tutorium} = 2 \wedge \text{Tutor} = \text{"nein"}}$ (Informatikstudierende)

SQL - Abfrage:

SELECT * FROM Informatikstudierende WHERE Tutor = 'Nein' and Tutorium = 2;

Bei dieser SQL-Datenbankoperation wird nach jenen Datensätzen in der Relation gesucht, die das Kriterium $\text{Tutorium} = 2 \text{ AND Tutor} = \text{"nein"}$ erfüllen.

Name	Vorname	Matr_nr	Fach	Semester	Info1-Tutor	Info1-Tutorium
Schulz	Karin	123451	Inf	1	nein	2
Kehl	Maria	123455	Inf	3	nein	2

g) Zur Lösung dieser Teilaufgabe kann die Relation "Doppel" aus Teilaufgabe b) verwendet werden. Auf sie muss eine Selektion ausgeführt werden.

Datenbankoperation:

$sel_{\text{Semester} = 1}$ (doppel)

liefert die Studierenden zurück, die ein Doppelstudium absolvieren und im ersten Semester sind.

SQL - Abfrage:

SELECT info.Name, info.Vorname, info.Matr_nr, info.Semester FROM Informatikstudierende AS info, Biologiestudierende AS bio WHERE info.Matr_nr = bio.Matr_nr AND info.Semester=1;

Tabelle:

Name	Vorname	Matr_nr	Semester
------	---------	---------	----------

Meier	Hans	123456	1
Mülle	Susanne	123452	1

Lösung

4. Java-Programm: Kleinstes Element (1 Punkt)

```
import infol.*

class KElem{
    public static void main(String args[]){
        int q=0; // Kleinste Zahl
        int z=4; // Länge d. Eingabe
        int[] a = new int[z]; // Lege Array mit z Elementen an
                                                // Zahlen einle:
        System.out.println("Geben Sie "+ z +" Zahlen ein:");
        for (int i = 1; i < z; i++){
            a[i]=Console.in.readInt();
        }
                                                // Setze erstes
        q = a[0];
                                                // Vergleiche q
        int j=0;
        while (j<z){
            if(a[j] < q) //Falls ein Element kleiner ist als q,
                q=a[j]; // wähle es als neues q.
            j++;
        }
                                                // Ausgabe
        System.out.println("Die kleinste Zahl ist "+ q + ".");
    }
}
```
