Prof. Dr.-Ing. Dr. h. c. T. Härder
Fachbereich Informatik
Arbeitsgruppe Datenbanken und Informationssysteme
Universität Kaiserslautern
www.haerder.de

Übungsblatt 8 – Lösungsvorschläge

Unterlagen zur Vorlesung:
„http://wwwlgis.informatik.uni-kl.de/cms/index.php?id=29“

Aufgabe 1: Normalisierung, Vereinfachung, Restrukturierung

Gegeben seien folgende Tabellen:

Personal (Pnr, Pname, Beruf, Gehalt)
Projekt (Pronr, Proname, Probudget)
PMitarbeit (Pnr, Pronr, Dauer)

1. Bestimmen Sie für die Qualifikationsbedingung der folgenden Anfragen die konjunktive und disjunktive Normalform:

```
Select * From Personal
Where (Pname Like 'M%' And Beruf = 'Techniker') Or
((Pnr > 550 Or Beruf = 'Programmierer') And Gehalt < 80000)
```

Konjunktive Normalform:

(Pname Like ‘M%’ ∨ Pnr > 550 ∨ Beruf = ‘Programmierer’) ∧
(Pname Like ‘M%’ ∨ Pnr > 550 ∨ Gehalt < 80000) ∧
(Pname Like ‘M%’ ∨ Beruf = ‘Programmierer’ ∨ Gehalt < 80000) ∧
(Pname Like ‘M%’ ∨ Gehalt < 80000) ∧
(Beruf = ‘Techniker’ ∨ Pnr > 550 ∨ Beruf = ‘Programmierer’) ∧
(Beruf = ‘Techniker’ ∨ Pnr > 550 ∨ Gehalt < 8000) ∧
(Beruf = ‘Techniker’ ∨ Beruf = ‘Programmierer’ ∨ Gehalt < 80000) ∧
(Beruf = ‘Techniker’ ∨ Gehalt < 80000).

Disjunktive Normalform:

(Pname Like ‘M%’ ∧ Beruf = ‘Techniker’) ∨
(Pnr > 550 ∧ Gehalt < 80000) ∨
(Beruf = ‘Programmierer’ ∧ Gehalt < 80000).

2. Vereinfachen Sie die Qualifikationsbedingungen der folgenden Anfrage durch Anwendung der Idempotenzregeln:

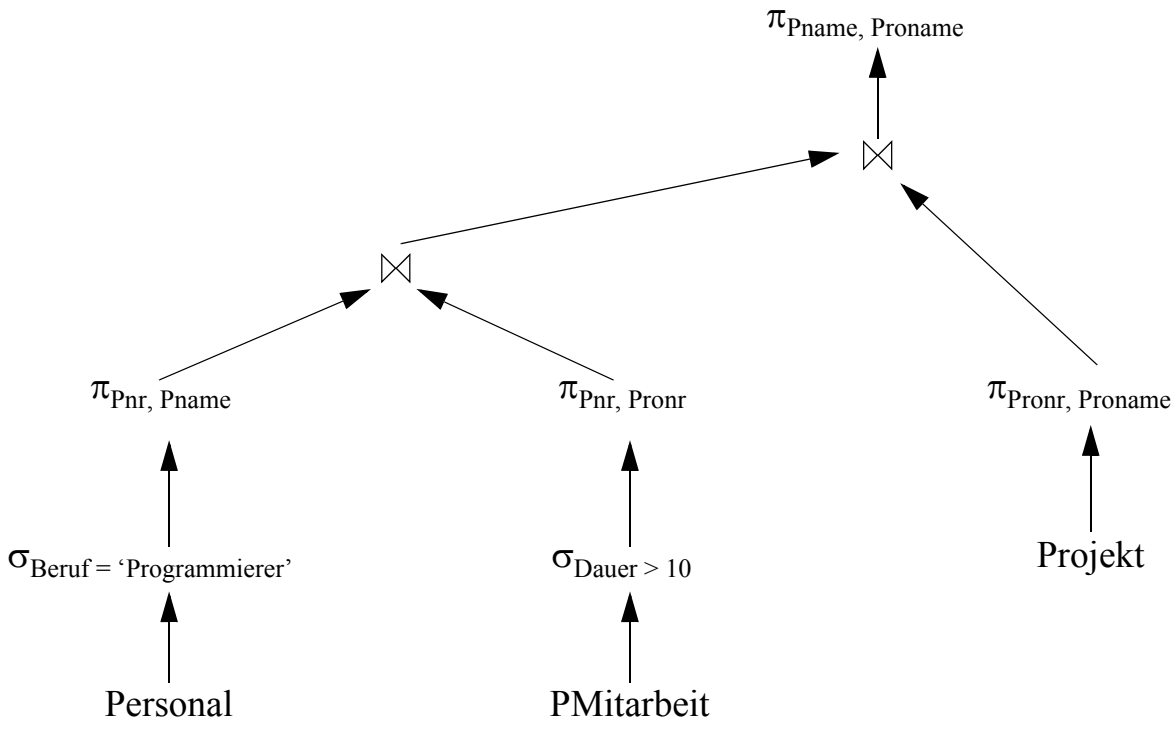
```
Select * From Personal
Where Pnr > 456 And
Not (Beruf = 'Techniker' Or Gehalt < 50000) And
Beruf ≠ 'Techniker' And Gehalt < 50000
```

Es ergibt sich eine leere Ergebnismenge wegen den gegensätzlichen Prädikaten bzgl. Gehalt.

3. Führen Sie die Anfragetransformation für folgende Query durch.

```
Select Pname, Proname
From Personal P, Projekt Pt, PMitarbeit Pm
Where Dauer > 10 and P.Pnr = Pm.Pnr and
Beruf = 'Programmierer' and Pt.Pronr = Pm.Pronr
```

Bestimmen Sie den Operatorbaum und führen darauf Vereinfachungen und Restrukturierungen zur algebraischen Optimierung durch.



Neben Selektionen wurden auch Projektionen vorgezogen. Es ist dabei darauf zu achten, daß die für die Join-Bearbeitung sowie für die Ausgabe benötigten Attribute erhalten bleiben. Die Reihenfolge der Join-Operationen wurde nicht optimiert, da über die Kardinalitäten keine Aussagen gemacht wurden.

Aufgabe 2: Einfache Join-Strategien

Sei Card (R) = 10000, Card (S) = 1000, JSF (R ⋈ S) = 0.001. Jede Tabelle soll 5 Attribute umfassen. Welche Kommunikationskosten ergeben sich für 'Ship Whole' bzw. 'Fetch as needed' bei Join-Ausführung an K_R bzw. an K_S?

S1 (Ship Whole, Join-Berechnung am R-Knoten):
#Nachrichten = 2; #AW = 1000*5 = 5 000.

S2 (Ship Whole, Join-Berechnung am S-Knoten):
#Nachrichten = 2; #AW = 10 000*5 = 50 000.

F1 (Fetch As Needed; Join-Berechnung am R-Knoten):
#Nachrichten = 10 000*2 = 20 000; #AW = 10 000 * 0.001 * 1000 * 5 = 50 000.

F2 (Fetch As Needed; Join-Berechnung am S-Knoten):
#Nachrichten = 1000*2 = 2000; #AW = 1000 * 0.001 * 10000 * 5 = 50 000.

Aufgabe 3: Ship-Whole vs. Semi-Join vs. Bitvektor- Join

Auf den Tabellen Personal und PMitarbeit seit folgende Join-Query zu bearbeiten:

```
Select P.Pnr, Pname, Beruf, Pronr, Dauer
From Personal.P, PMitarbeit Pm
Where P.Pnr = Pm.Pnr And P.Gehalt > 60000
```

Es gelte Card (Personal) = 1000, Card (PMitarbeit) = 1500; beide Tabellen seien an verschiedenen Knoten gespeichert. Die Anfrage soll an einem dritten Knoten K initiiert werden; das Ergebnis ist dort auch auszugeben. Die Gehaltsbedingung soll von 20 % der Angestellten erfüllt werden (SF = 0.2); 25% der Angestellten sollen in keinem Projekt mitarbeiten.

Bestimmten Sie die Kommunikationskosten (#Nachrichten, #AW (Anzahl zu übertragender Attributwerte)) für folgende Join-Strategien:

- Ship-Whole; Join-Berechnung an Knoten $K_{Pmitarbeit}$
- Ship-Join; Join-Berechnung an Knoten K
- Semi-Join; Join-Bestimmung an Knoten $K_{Personal}$
- Semi-Join; Join-Berechnung an Knoten K
- Bitvektor-Join; Join-Berechnung an Knoten K

Vor der Übertragung sollen alle anwendbaren Selektionen und Projektionen durchgeführt werden. Die Länge des Bitvektors soll 5 Attributwerten entsprechen; durch Anwendung des Bitvektors soll sich die zurückzuliefernde Tupelanzahl um 5% erhöhen.

Ship-Whole; Join-Berechnung an Knoten $K_{PMitarbeit}$

- $K \rightarrow K_{PMitarbeit}$: Nachricht zum Query-Start
- $K_{PMitarbeit} \rightarrow K_{Personal}$: Anfordern der Personal-Daten
- $K_{Personal} \rightarrow K_{Pmitarbeit}$: 1000*0.2*3 = 600 AW
- Join-Berechnung an $K_{PMitarbeit}$
- $K_{PMitarbeit} \rightarrow K$: Join-Ergebnis (200*0.75*2*5 = 1500 AW)
  Jeder Angestellte mit Projekten arbeitet im Mittel in zwei Projekten mit.

➥ #Nachrichten = 4; #AW = 2100.

Ship-Whole; Join-Berechnung an Knoten K

- $K \rightarrow K_{Pmitarbeit}$: Anfordern der PMitarbeit-Daten
- $K \rightarrow K_{Personal}$: Anfordern der Personal-Daten
- $K_{PMitarbeit} \rightarrow K$: 1500*3 = 4500 AW
- $K_{Personal} \rightarrow K$: 1000*0.2*3 = 600 AW
- Join-Berechnung an K

➥ #Nachrichten = 4; #AW = 5100.

Semi-Join; Join-Berechnung an Knoten $K_{Personal}$

- $K \rightarrow K_{Personal}$: Nachricht zum Query-Start
- $K_{Personal} \rightarrow K_{PMitarbeit}$: 1000*0.2 = 200 AW Verbundattributwerte
- $K_{PMitarbeit} \rightarrow K_{Personal}$: 200*0.75*2*3 = 900 AW (Verbundpartner)
- Join-Berechnung an $K_{Personal}$
- $K_{Personal} \rightarrow K$: Join-Ergebnis (200*0.75*2*5 = 1500 AW)

➥ #Nachrichten = 4; #AW = 2600.

Semi-Join; Join-Berechnung an Knoten K

- $K \rightarrow K_{PMitarbeit}$: Anfordern der reduzierten Daten
- $K \rightarrow K_{Personal}$: Anfordern der reduzierten Daten
- $K_{PMitarbeit} \rightarrow K_{Personal}$: 750 AW Verbundattributwerte
- $K_{Personal} \rightarrow K$: 750*0.2*3 = 450 AW (Verbundpartner)
- $K_{Personal} \rightarrow K_{PMitarbeit}$: 1000*0.2 = 200 AW Verbundattributwerte
- $K_{PMitarbeit} \rightarrow K$: 200*0.75*2*3 = 900 AW (Verbundpartner)
- Join-Berechnung an K

➥ #Nachrichten = 6; #AW = 2300.

Bitvektor-Join; Join-Berechnung an Knoten K

wie vorher, jedoch wird in Schritten 3 und 5 jeweils der Bitvektor übertragen ($\rightarrow$ 10 AW); in Schritt 4 (6) erhöht sich der Übertragungsumfang auf ca. 475 (945) AW.

➥ #Nachrichten = 6; #AW = 1430.

Aufgabe 4: Optimierung von mengenorientierten Anfragen mit Hilfe von Cluster-Bildung

Gegeben sei eine Tabelle Personal mit Attributen für die Personalnummer, den Namen des Angestellten, das Gehalt sowie den Manager bei dem der Mitarbeiter beschäftigt ist. Daneben ist noch weitere, hier nicht relevante Information enthalten. Insgesamt ergibt sich Personal (Pnr, Name, Gehalt, Mnr, ...), wobei die Datensätze zwischen 512 und 1024 Bytes groß sind. Die genannten Attribute haben die Größen: Pnr = 2 Byte, Name = 30 Byte, Gehalt = 4 Byte. Im Durchschnitt hat jeder Manager ca. 10 Mitarbeiter. Die Seitengröße des DBVS sowie die Slotgröße der Dateien auf dem Externspeicher betragen 4kB.Weiterhin existiere eine Indexstruktur $I_{Personal}(Pnr)$.

Folgende Anfrage soll unterstützt werden:

- Select P1.Name, P1.Gehalt P2.Gehalt
  From Personal P1, Personal P2
  Where P1.Mnr = P2.Pnr AND P1.Gehalt > P2.Gehalt

Untersuchen Sie die Frage, ob Cluster-Bildung der Tabelle Personal bezüglich der Mnr sinnvoll ist. Wenden Sie dabei für den Verbund-Algorithmus Nested Loops an und diskutieren Sie den Einfluß des DB-Puffers auf das E/A-Verhalten, wenn minimal nur 5 und maximal 200 Pufferrahmen zur Verfügung gestellt werden. Berücksichtigen Sie dabei auch die Größe N der Tabelle (z. B. $N=10^3$ und $N=10^5$ Tupel in Personal).

Datensätze sind im Mittel 768 Bytes groß, damit sind im Durchschnitt 5 Datensätze in einer Seite.

Für die Anfrage gilt:

mit Cluster-Bildung:

N/5 Zugriffe auf Datenseiten
N/10 Zugriffe auf Seiten mit Managern über $I_{Personal}(Pnr)$
(es gibt N/10 Manager,
und Datensätze sind mit Cluster-Bildung nach Mnr abgespeichert)

==> min. Puffer: 200 + 100 = 300 Zugriffe bei 10^3 Datensätzen bzw.
==> max. Puffer: 200 Zugriffe bei 10^3 Datensätzen bzw.

==> ~ 20 000 + 10 000 = 30 000 Zugriffe bei 10^5 Datensätzen
bei min. und max. Pufferrahmen

ohne Cluster-Bildung, jedoch kompakt auf N/5 Seiten gespeichert
:

N/5 Zugriffe auf Datenseiten

N Zugriffe auf Seiten mit Manager

==> min. Puffer: 200 + 1000 = 1 200 Zugriffe bei 10^3 Datensätzen
bzw.
==> max. Puffer: 200 + 500 = 700 Zugriffe bei 10^3 Datensätzen
unter der Annahme, daß $I_{Personal}$(Pnr) im Puffer resident ist und im Mittel jeder 2. Zugriff über Pnr den Satz im Puffer vorfindet
bzw.
==> ~ 20 000 + 100 000 = 120 000 Zugriffe bei 10^5 Datensätzen, da die Wahrscheinlichkeit, einen gesuchten Personal-Satz in Puffer mit 200 Rahmen zu finden, sehr gering ist.

Die Verbesserung durch die Cluster-Bildung variiert also um den Faktor 3.5-4.

Welchen Einfluß auf die Aktualisierungskosten hat eine Cluster-Bildung über Mnr (Anpassungen von $I_{Personal}$(Pnr) werden nicht betrachtet), wenn

- die Abteilung einen neuen Manager bekommt? Hier muß die Mnr in allen Sätzen eines Clusters auf die Pnr des neuen Chefs geändert werden.

  1. Änderung beim Manager (Änderung eines ganzen Clusters)

     ==> Änderung eines Clusters (2 Seiten) (bei Cluster-Bildung)
     ==> Änderung von 10 Sätzen (10 Seiten) (keine Cluster-Bildung)

- ein Mitarbeiter in eine neue Abteilung wechselt?

  2. Änderung eines Mitarbeiters, Abteilungswechsel (Änderung eines Satzes)

ohne Cluster-Bildung: immer 3 Zugriffe

(1 Index lesen, 1 Datensatz lesen, 1 Datensatz schreiben: also 2 Seiten lesen und 1 Seite schreiben)

mit Cluster-Bildung:

Bemerkung: Es ist unsinnig, hier anzunehmen, daß notwendige Verschiebungen einen Domino-Effekt über den gesamten betroffenen Bereich von Tabelle Personal auslösen.
Deshalb sind die Kosten nur geringfügig höher als die ohne Cluster-Bildung, wobei ggf. noch das Splitting einer Seite (beim neuen Cluster) dazukommt.
(1 Index lesen, 1 Datensatz lesen und löschen und 1 Datensatz schreiben: also 2 Seiten lesen und 2 (+ Delta) Seiten schreiben)