Anfragebearbeitung (Literatur: divers)

Gegeben: Anfrage (Read + Update) am Knoten i

Ziele:

1. richtiges Ergebnis am Knoten i
2. Ergebis möglichst effizient berechnen; d.h. mit geringen Kosten und/oder geringer Antwortzeit

Berücksichtige:

- Partitionierung
- Allokation (inkl. Redundanz)
- viele unterschiedliche Möglichkeiten z.B. einen Join auszuführen. (Alle "zentralen" Möglichkeiten plus einige mehr im verteilten Fall.)

Zur Erinnerung: Wieso brauchen Sie das?

Kein System macht das alles richtig. Deswegen müssen Sie, wo immer Sie auch sind, fehlendes Zeug in Ihre Anwendungen einbauen, damit die Anwendungen einigermaßen effizient laufen.

Roadmap:

1. Einführung
2. Query Rewrite
3. Anfrageoptimierer
4. Kostenmodelle
5. Anfrageauswertung

Query Rewrite

- algebraische Umformungen auf logischer Ebene
- nutzt "semanitisches" Wissen aus
  - Constraints
  - Partitionierungsschema
  - (nicht Allokation; kein Kostenmodell)
- Ziel – Umformungen, die die spätere kostenbasierte Optimierung ermöglichen
- Beispiele:
  - Vereinfachung von algebraischen Ausdrücken,
    Elimination redundanter Ausdrücke,
    Erkennung gemeinsamer Teilausdrücke,
    Erkennung von zusätzlichen Prädikaten
  - Auflösen von Subqueries und Views
  - Transformation von globalen Anfragen in lokal ausführbare Anfragen

Beispiel 1: Hinzunahme von Joinprädikaten

SELECT *
FROM A, B, C
WHERE A.a = B.b AND B.b = C.c

wird durch Query Rewrite zu

SELECT *
FROM A, B, C
WHERE A.a = B.b AND B.b = C.c
AND A.a = C.c

Hierdurch wird der folgende Plan möglich:

$$(A \bowtie C) \bowtie B$$

(Ohne Query Rewrite wäre $A \bowtie C$ ein karthesisches Produkt.)

Beispiel 2: Auflösen von Views und Subqueries

```
SELECT distinct A.a
FROM A
WHERE A.a IN (SELECT B.b FROM B)
```

wird zu

```
SELECT distinct A.a
FROM A, B
WHERE A.a = B.b
```

Hierdurch sind alle Joinmethoden anwendbar.

Beispiel 3: Tranformation von globalen Anfragen in lokale Anfragen

Teile ist in 3 horizontale Partitionen geteilt:

SELECT *
FROM Teile
WHERE $25 \leq$ TeileNr ≤ 350

wird zu

SELECT *
FROM Teile_1
WHERE $25 \leq$ TeileNr ≤ 350
$\cup$
SELECT *
FROM Teile_2
WHERE $25 \leq$ TeileNr ≤ 350
$\cup$
SELECT *
FROM Teile_3
WHERE $25 \leq$ TeileNr ≤ 350

Regel: Ersetze globale Relation durch ihre Rekonstruktionsvorschrift.

Beispiel 3 (Fortsetzung)
(Weitere Optimierungen, die sich ergeben können)

$\text{Teile}_1 = \sigma(\text{Teile}, \text{TeileNr} \leq 300)$
$\text{Teile}_2 = \sigma(\text{Teile}, 300 < \text{TeileNr} \leq 600)$
$\text{Teile}_3 = \sigma(\text{Teile}, \text{TeileNr} > 600)$

SELECT *
FROM Teile_3
WHERE $25 \leq \text{TeileNr} \leq 350$

ist äquivalent zu

SELECT *
FROM Teile_3
WHERE $25 \leq \text{TeileNr} \leq 350$
AND $\underline{\text{TeileNr} > 600}$

ist äquivalent zu

$\emptyset$

Anwendung weiterer Regeln bringt weitere Vereinfachung. (Insbesondere deshalb, weil Partitionierung darauf ausgelegt ist, dass man einige Partitionen bei vielen Anfragen nicht betrachten muss.)

Allgemein

- Ersetze jede globale Relation durch ihre "Rekonstruktionsvorschrift"
  $\rightarrow$ d.h. betrachte jede globale Relation als ein View.
- dann wende alle bekannten Techniken für Query Rewrite in zentralen Datenbanksystemen an, um Views aufzulösen, Prädikate zu vereinfachen und nicht erfüllbare Ausdrücke zu eliminieren.
- Implementierung: am besten durch ein Regelsystem (Vorsicht! Das ist nicht ganz trivial.)
  - evtl. mehrfache Anwendung derselben Regel
  - evtl. können Regeln erst nach der Anwendung von anderen Regeln angewendet werden.
  - Konfluenz ???
- Literatur
  - Ceri, Pelagatti (Kapitel 5) für spezielle Fälle in verteilten Systemen
  - Pirahesh, Hellerstein, Hasan (SIGMOD 1992) für Fälle in zentralen Systemen

Und noch ein Beispiel: Aggregation

SELECT AVG(Preis)
FROM Teile

wird zu

SELECT AVG(Preis)
FROM $\text{Teile}_1 \cup \text{Teile}_2 \cup \text{Teile}_3$

wird zu

SELECT $\frac{v_1.S+v_2.S+v_3.S}{v_1.C+v_2.C+v_3.C}$
FROM V_1 v_1, V_2 v_2, V_3 v_3
mit
CREATE VIEW V_i IS
SELECT COUNT as C, SUM(Preis) as S
FROM Teile_i

Vorteil: Es müssen lediglich Count und Sum Werte anstatt kompletter Partitionen durch die Gegend geshippert werden.