

Übungen Informationssysteme WS 1998/99

Anmerkungen zu Übungsblatt 5: Anfragebearbeitung

Jan Stohner, stohner@informatik.uni-bonn.de
Universität Bonn, Institut für Informatik III, Römerstr. 164, 53117 Bonn

26. November 1998

Aufgabe 1

Die Auswertung der SQL-Anfragen ergibt folgende Bearbeitungszeiten (Abweichungen durch Rechnerauslastung oder Meßungenauigkeiten können das Ergebnis beeinflussen!):

| Anfrage | Zeit |
|---|-------------|
| • SELECT COUNT(*) FROM Big b1, Big b2 WHERE b1.A1=b2.A1 AND b1.A1 BETWEEN 5000 AND 5004; | 00:00:00.02 |
| SELECT COUNT(*) FROM Big b1, Big b2 WHERE b1.A1=b2.A1 AND b1.A2 BETWEEN 5000 AND 5004; | 00:02:02.80 |
| • SELECT COUNT(A2) FROM Big WHERE A3 BETWEEN 5000 AND 5004; | 00:00:00.02 |
| SELECT COUNT(A3) FROM Big WHERE A2 BETWEEN 5000 AND 5004; | 00:00:05.84 |
| • SELECT AVG(b2.A2) FROM Big b1, Big b2 WHERE b1.A3>490000 AND b1.A2=b2.A3; | 00:00:02.45 |
| SELECT AVG(b2.A2) FROM Big b1, Big b2 WHERE b1.A1>490000 AND b1.A2=b2.A1; | 00:00:03.65 |

Dem menschlichen Betrachter fällt auf, daß jeweils die zwei Anfragen der 3 Aufzählungspunkte das gleiche Ergebnis liefern, da die Relation *Big* in den Attributen A1, A2 und A3 identisch ist. Trotzdem benötigt Auswertung der ersten Anfrage 2 Hundertstelsekunden, die Auswertung der zweiten Anfrage 2 Minuten und 2,8 Sekunden.

Über dieses Wissen verfügt das Datenbanksystem nicht, weshalb es auf die ihm vorgegebenen Index-Strukturen zur Auswertung angewiesen ist. Dadurch ergeben sich die teilweise drastischen Laufzeitunterschiede:

- In der ersten Anfrage kann zur Auswertung von `b1.A1 BETWEEN 5000 AND 5004` der Index auf A1 genutzt werden. Bei der zweiten Anfrage, wo A2 zwischen 5000 und 5004 liegen soll, kann kein Index genutzt werden. Deshalb wird die Relation vollständig sequentiell gelesen, um die qualifizierenden Tupel zu finden. Es gibt zwar einen zusammengesetzten Index (A3,A2), dieser kann aber nicht genutzt werden, da nach A2 nur an zweiter Stelle der Sortierreihenfolge steht (Bsp. Telefonbuch: Das Telefonbuch ist nach (Nachname, Vorname) sortiert, man kann also nicht alle Personen mit dem gleichen Vornamen unmittelbar nachschlagen.).
- Ebenso wie beim ersten Anfragepaar kann in der ersten Anfrage für A3 ein Index (jetzt der zusammengesetzte) verwendet werden, für A2 ist bei der zweiten Anfrage aber ein vollständiges Durchlaufen der Tabelle nötig.
- Während beim dritten Anfragepaar die Anfrage A1/A3 größer 490000 durch den Index vollzogen werden kann, spart die erste Anfrage einige Tabellenzugriffe, weil aus dem zusammengesetzten Index (A3,A2) nach der Suche nach A3 auch sofort der Wert von A2 bekannt ist, ebenso ist

nach Suche von b2.A3 auch b2.A2 bekannt, über das ja der Durchschnitt gebildet wird. Bei der zweiten Anfrage sind Zugriffe auf die Tabelle erforderlich, da man, nachdem b1.A1 gefunden wurde, noch den Wert b1.A2 benötigt. Der gleiche Zugriff ist auch für b2.A1 und b2.A2 nötig.

Oracle 8 gibt folgende Auswertungspläne für das erste Anfragepaar aus:

- Anfrage:

```
SELECT COUNT(*) FROM Big b1, Big b2
WHERE b1.A1=b2.A1 AND b1.A1 BETWEEN 5000 AND 5004;
```

Plan:

| OPERATION | OPTIONS | OBJECT_NAME | POSITION |
|------------------|-------------|-------------|----------|
| SELECT STATEMENT | | | |
| SORT | AGGREGATE | | 1 |
| NESTED LOOPS | | | 1 |
| INDEX | RANGE SCAN | PK_BIG | 1 |
| INDEX | UNIQUE SCAN | PK_BIG | 2 |

- Anfrage:

```
SELECT COUNT(*) FROM Big b1, Big b2
WHERE b1.A1=b2.A1 AND b1.A2 BETWEEN 5000 AND 5004;
```

Plan:

| OPERATION | OPTIONS | OBJECT_NAME | POSITION |
|------------------|----------------|-------------|----------|
| SELECT STATEMENT | | | |
| SORT | AGGREGATE | | 1 |
| NESTED LOOPS | | | 1 |
| TABLE ACCESS | FULL | IS_BIG | 1 |
| TABLE ACCESS | BY INDEX ROWID | IS_BIG | 2 |
| INDEX | UNIQUE SCAN | PK_BIG | 1 |

Abbildung 1 zeigt eine graphische Darstellung der Anfragepläne. In beiden Plänen setzt Oracle das Nested-Loops Verfahren ein. Die äußere Schleife des ersten Planes besteht aus einem *Range Scan*, Oracle sucht hier also einen Indexbereich ab. Mit dem Range Scan werden alle Indexeinträge b1 gesucht, für die A1 zwischen 5000 und 5004 liegt. Mit dem *Unique Scan* der inneren Schleife werden passende Einträge b2 mit b2.A1=b1.A1 gesucht.

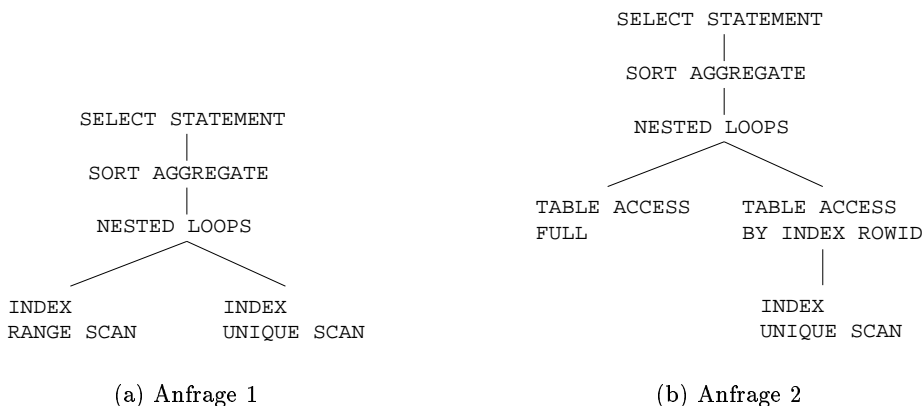


Abbildung 1: Graphische Darstellung der Anfragepläne

Bei der zweiten Anfrage bleibt für den Bereichstest von b1.A2 nur ein vollständiges durchlaufen der Tabelle übrig. Oracle bezeichnet dies als *Full Table Access*. Anschließend wird in der inneren Schleife nach einem passenden b2.A1 gesucht. Dies kann mit Hilfe des Indexes auf A1 geschehen. Mit *Table Access by Index RowId* ist bei Oracle der Zugriff auf das gesamte Tupel gemeint. Im Gegensatz zum

index-Zugriff nicht Grace also alle Attribute des Tupels ein (Dies ist an dieser Stelle unnötig, da das Tupel lediglich gezählt, aber nicht ausgewertet werden soll.).

Aufgabe 2

Betrachten Sie nochmal das Beispiel zur logischen Optimierung aus der Vorlesung (Folien 209–213) und das nicht besprochene Beispiel „Flugreservierung“ auf den Folien 215–225. Anschließend sollte die Lösung dieser Aufgabe kein Problem mehr darstellen.

Aufgabe 3

Lesen Sie hierzu bitte den — auch auf der Vorlesungs-Homepage erreichbaren — Text „Anmerkungen zum verfeinerten Join“.