

Introduction to Database Systems CSE 444

Lecture 17: Database Tuning

Magda Balazinska - CSE 444, Fall 2010

1

Database Tuning Overview

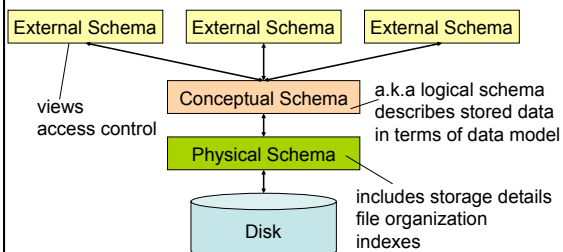
- The database tuning problem
- Index selection (discuss in detail)
- Horizontal/vertical partitioning (see lecture 4)
- Denormalization (discuss briefly)

This material is partially based on the book: "Database Management Systems" by Ramakrishnan and Gehrke, Ch. 20

Magda Balazinska - CSE 444, Fall 2010

2

Levels of Abstraction in a DBMS



Magda Balazinska - CSE 444, Fall 2010

3

The Database Tuning Problem

- We are given a workload description
 - List of queries and their frequencies
 - List of updates and their frequencies
 - Performance goals for each type of query
- Perform *physical database design*
 - Choice of indexes
 - Tuning the conceptual schema
 - Denormalization, vertical and horizontal partition
 - Query and transaction tuning

Magda Balazinska - CSE 444, Fall 2010

4

The Index Selection Problem

- Given a database schema (tables, attributes)
- Given a "query workload":
 - Workload = a set of (query, frequency) pairs
 - The queries may be both SELECT and updates
 - Frequency = either a count, or a percentage
- Select a set of indexes that optimizes the workload

In general this is a very hard problem

Magda Balazinska - CSE 444, Fall 2010

5

Index Selection: Which Search Key

- Make some attribute K a search key if the WHERE clause contains:
 - An exact match on K
 - A range predicate on K
 - A join on K

Magda Balazinska - CSE 444, Fall 2010

6

The Index Selection Problem 1

V(M, N, P);

Your workload is this

100000 queries: 100 queries:

```
SELECT *
FROM V
WHERE N=?
```

```
SELECT *
FROM V
WHERE P=?
```

What indexes ?

Magda Balazinska - CSE 444, Fall 2010 7

The Index Selection Problem 1

V(M, N, P);

Your workload is this

100000 queries: 100 queries:

```
SELECT *
FROM V
WHERE N=?
```

```
SELECT *
FROM V
WHERE P=?
```

A: V(N) and V(P) (hash tables or B-trees)

Magda Balazinska - CSE 444, Fall 2010 8

The Index Selection Problem 2

V(M, N, P);

Your workload is this

100000 queries: 100 queries: 100000 queries:

```
SELECT *
FROM V
WHERE N>? and N<?
```

```
SELECT *
FROM V
WHERE P=?
```

```
INSERT INTO V
VALUES (?, ?, ?)
```

What indexes ?

Magda Balazinska - CSE 444, Fall 2010 9

The Index Selection Problem 2

V(M, N, P);

Your workload is this

100000 queries: 100 queries: 100000 queries:

```
SELECT *
FROM V
WHERE N>? and N<?
```

```
SELECT *
FROM V
WHERE P=?
```

```
INSERT INTO V
VALUES (?, ?, ?)
```

A: definitely V(N) (must B-tree); unsure about V(P)

Magda Balazinska - CSE 444, Fall 2010 10

The Index Selection Problem 3

V(M, N, P);

Your workload is this

100000 queries: 1000000 queries: 100000 queries:

```
SELECT *
FROM V
WHERE N=?
```

```
SELECT *
FROM V
WHERE N=? and P>?
```

```
INSERT INTO V
VALUES (?, ?, ?)
```

What indexes ?

Magda Balazinska - CSE 444, Fall 2010 11

The Index Selection Problem 3

V(M, N, P);

Your workload is this

100000 queries: 1000000 queries: 100000 queries:

```
SELECT *
FROM V
WHERE N=?
```

```
SELECT *
FROM V
WHERE N=? and P>?
```

```
INSERT INTO V
VALUES (?, ?, ?)
```

A: V(N, P)

Magda Balazinska - CSE 444, Fall 2010 12

The Index Selection Problem 4

V(M, N, P);

Your workload is this
 1000 queries: 100000 queries:

```
SELECT *
FROM V
WHERE N>? and N<?
```

```
SELECT *
FROM V
WHERE P>? and P<?
```

What indexes ?

Magda Balazinska - CSE 444, Fall 2010 13

The Index Selection Problem 4

V(M, N, P);

Your workload is this
 1000 queries: 100000 queries:

```
SELECT *
FROM V
WHERE N>? and N<?
```

```
SELECT *
FROM V
WHERE P>? and P<?
```

A: V(N) secondary, V(P) primary index

Magda Balazinska - CSE 444, Fall 2010 14

The Index Selection Problem

- **SQL Server**
 - Automatically, thanks to *AutoAdmin* project
 - Much acclaimed successful research project from mid 90's, similar ideas adopted by the other major vendors
- **PostgreSQL**
 - You will do it manually, part of project 3
 - But tuning wizards also exist

Magda Balazinska - CSE 444, Fall 2010 15

Basic Index Selection Guidelines

- Consider queries in workload in order of importance
- Consider relations accessed by query
 - No point indexing other relations
- Look at WHERE clause for possible search key
- Try to choose indexes that speed-up multiple queries
- And then consider the following...

Magda Balazinska - CSE 444, Fall 2010 16

Index Selection: Multi-attribute Keys

Consider creating a multi-attribute key on K1, K2, ... if

- WHERE clause has matches on K1, K2, ...
 - But also consider separate indexes
- SELECT clause contains only K1, K2, ..
 - A *covering index* is one that can be used exclusively to answer a query, e.g. index R(K1,K2) covers the query:

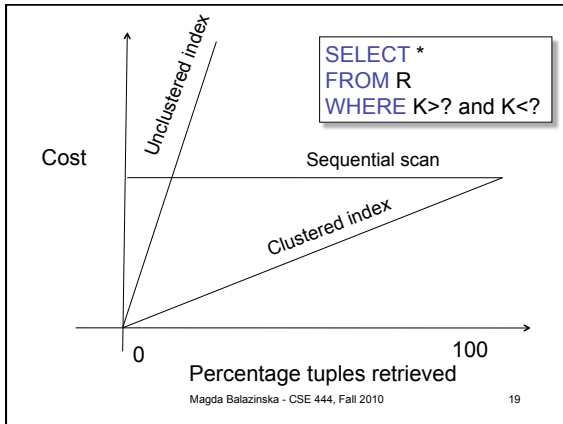
```
SELECT K2 FROM R WHERE K1=55
```

Magda Balazinska - CSE 444, Fall 2010 17

To Cluster or Not

- Range queries benefit mostly from clustering
- Covering indexes do *not* need to be clustered: they work equally well unclustered

Magda Balazinska - CSE 444, Fall 2010 18



Hash Table v.s. B+ tree

- Rule 1: always use a B+ tree ☺
- Rule 2: use a Hash table on K when:
 - There is a very important selection query on equality (WHERE K=?), and no range queries
 - You know that the optimizer uses a nested loop join where K is the join attribute of the inner relation (you will understand that in a few lectures)

Magda Balazinska - CSE 444, Fall 2010 20

Balance Queries v.s. Updates

- Indexes speed up queries
 - SELECT FROM WHERE
- But they usually slow down updates:
 - INSERT, DELETE, UPDATE
 - However some updates benefit from indexes

UPDATE R
SET A = 7
WHERE K=55

Magda Balazinska - CSE 444, Fall 2010 21

Tools for Index Selection

- SQL Server 2000 Index Tuning Wizard
- DB2 Index Advisor
- How they work:
 - They walk through a large number of configurations, compute their costs, and choose the configuration with minimum cost

Magda Balazinska - CSE 444, Fall 2010 22

Tuning the Conceptual Schema

- Index selection
- Horizontal/vertical partitioning (see lecture 4)
- Denormalization

Magda Balazinska - CSE 444, Fall 2010 23

Denormalization

Product(pid, pname, price, cid)
Company(cid, cname, city)

A very frequent query:

SELECT x.pid, x.pname
FROM Product x, Company y
WHERE x.cid = y.cid and x.price < ? and y.city = ?

How can we speed up this query workload ?

Magda Balazinska - CSE 444, Fall 2010 24

Denormalization

Product(pid, pname, price, cid)
Company(cid, cname, city)

Denormalize:

ProductCompany(pid, pname, price, cname, city)

```
INSERT INTO ProductCompany
SELECT x.pid, x.pname, x.price, y.cname, y.city
FROM Product x, Company y
WHERE x.cid = y.cid
```

Magda Balazinska - CSE 444, Fall 2010

25

Denormalization

Next, replace the query

```
SELECT x.pid, x.pname
FROM Product x, Company y
WHERE x.cid = y.cid and x.price < ? and y.city = ?
```



```
SELECT pid, pname
FROM ProductCompany
WHERE price < ? and city = ?
```

Magda Balazinska - CSE 444, Fall 2010

26

Issues with Denormalization

- It is no longer in BCNF
 - We have the hidden FD: cid → cname, city
- When Product or Company are updated, we need to propagate updates to ProductCompany
 - Use RULE in PostgreSQL (see PostgreSQL doc.)
 - Or use a trigger on a different RDBMS
- Sometimes cannot modify the query
 - What do we do then ?

Magda Balazinska - CSE 444, Fall 2010

27

Denormalization Using Views

```
INSERT INTO ProductCompany
SELECT x.pid, x.pname, price, y.cid, y.cname, y.city
FROM Product x, Company y
WHERE x.cid = y.cid;
```

DROP Product; DROP Company;

```
CREATE VIEW Product AS
SELECT pid, pname, price, cid FROM ProductCompany
```

```
CREATE VIEW Company AS
SELECT DISTINCT cid, cname, city FROM ProductCompany
```