Introduction to Database Systems
CSE 444

Lecture 17: Database Tuning

Magda Balazinska - CSE 444, Fall 2010

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

Levels of Abstraction in a DBMS

External Schema
External Schema
External Schema

Conceptual Schema
a.k.a logical schema
describes stored data
in terms of data model

Physical Schema
includes storage details
file organization
indexes

Disk

views
access control

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

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

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
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

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

The Index Selection Problem 3

V(M, N, P);

Your workload is this
100000 queries: 100000 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

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

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

A: V(N, P)
The Index Selection Problem 4

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?

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...

Index Selection: Multi-attribute Keys

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

To Cluster or Not

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

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

Tuning the Conceptual Schema

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

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?
Denormalization

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

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

\[
\begin{align*}
\text{INSERT INTO ProductCompany} \\
& \quad \text{SELECT } x.pid, x.pname, x.price, y.cid, y.cname, y.city \\
& \quad \text{FROM Product } x, \text{ Company } y \\
& \quad \text{WHERE } x.cid = y.cid
\end{align*}
\]

Issues with Denormalization

• It is no longer in BCNF
  – We have the hidden FD: cid \(\rightarrow\) 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?

Denormalization Using Views

\[
\begin{align*}
\text{INSERT INTO ProductCompany} \\
& \quad \text{SELECT } x.pid, x.pname, x.price, y.cid, y.cname, y.city \\
& \quad \text{FROM Product } x, \text{ Company } y \\
& \quad \text{WHERE } x.cid = y.cid; \\
\text{DROP Product; DROP Company;}
\end{align*}
\]

\[
\begin{align*}
\text{CREATE VIEW Product AS} \\
& \quad \text{SELECT pid, pname, price, cid FROM ProductCompany}
\end{align*}
\]

\[
\begin{align*}
\text{CREATE VIEW Company AS} \\
& \quad \text{SELECT DISTINCT cid, cname, city FROM ProductCompany}
\end{align*}
\]