

Indexes & Performance Tuning

Chapter 8.5

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 1

Physical Database Design

- One important job of DBA
- Choice of physical schema, indexes have a big effect on system performance
- Important to choose indexes based on expected *workload*
 - Queries
 - Updates

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 2

Choice of Indexes

- What indexes should we create?
 - Which relations should have indexes?
 - What field(s) should be the search key?
 - Should we build several indexes?
- For each index, what kind of an index should it be?
 - Clustered?
 - Hash index or B+ tree?

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 3

Choice of Indexes

- What are the tradeoffs?
 - Indexes (often) make queries go faster
 - Make updates slower because we must maintain the index
 - Also, consume disk space

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 4

One Approach

- Consider most important queries.
- Consider the best *plan* using the current indexes.
- See if a better plan is possible with additional index.
 - Need to understand how a DBMS evaluates queries and creates *query evaluation plans*!
 - For now, consider 1-table queries.

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 5

Index Selection Guidelines

Query #1:
SELECT * FROM Students
WHERE Sid = 12345

Query #2:
SELECT Name FROM Students
WHERE GPA > 3.5

Query #3:
SELECT Name FROM Students
WHERE GPA > 1.5

- Attributes in WHERE clause are candidates for index keys.
 - Range query suggests tree index
 - Equality suggests hash or tree index
 - Clustering especially useful for range queries

Students			
Sid	Name	Age	GPA

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 6

Indexes with Composite Keys

- Consider a B+ tree with a composite search key <Age, GPA>
 - Order of attributes important for range queries!

Data entries w/ Age = 20, sorted by GPA

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 7

Indexes with Composite Keys

- B+ Tree on <Age, GPA>
- Query #4
 SELECT * FROM Students
 WHERE Age = 20 AND GPA = 4.0
 Index on <Age, GPA> better than index on <Age> or index on <GPA>
- Query #5
 SELECT * FROM Students
 WHERE Age = 20 AND GPA > 3.5
 Index on <Age, GPA> is also good
- Query #6
 SELECT * FROM Students
 WHERE GPA > 3.5
 Index on <Age, GPA> Not useful! (Index on <GPA> would be better.)

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 8

Index Selection Guidelines

- Consider composite key indexes when WHERE clause contains multiple conditions
 - Index must "match" the query
 - Attribute order important for range queries
- Composite key indexes can potentially provide *index-only plans*

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 9

Composite Key Design Examples

- Query #8


```
SELECT S.sid
FROM Students S
WHERE S.age BETWEEN 17 AND 19
      AND S.gpa BETWEEN 3.0 AND 3.5
```
- Query #9


```
SELECT S.sid
FROM Students S
WHERE S.age BETWEEN 17 AND 23
      AND S.gpa BETWEEN 3.99 AND 4.0
```

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 10

Index-Only Plans

- A number of queries can be answered just using the index (without retrieving any full tuples)

<p>$\langle S.age \rangle$</p> <p>$\langle S.age, S.gpa \rangle$ <i>Tree index</i></p> <p>$\langle S.age, S.gpa \rangle$ or $\langle S.gpa, S.age \rangle$ <i>Tree index</i></p>	<pre>SELECT S.age, COUNT(*) FROM Students S GROUP BY S.age</pre>
	<pre>SELECT S.age, MIN(S.gpa) FROM Students S GROUP BY S.age</pre>
	<pre>SELECT AVG(S.gpa) FROM Students S WHERE S.age=22 AND S.gpa BETWEEN 3.0 AND 4.0</pre>

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 11

Index Specification in SQL

- SQL:99 has no standard statement
- However the syntax is very similar across systems
- For example:


```
CREATE BTREE INDEX IdxGpaYear ON
Student (gpa ASC, year DESC)
```

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 12

Automatic Index Selection

- Many modern commercial systems include index tuning “wizards”
 - MS SQL Server Index Tuning Wizard
 - IBM DB2 Index Advisor
- Tools for automatically selecting indexes based on workload

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 13

Summary

- Many alternative file organizations exist, each appropriate in some situation.
- If selection queries are frequent, building an *index* is usually important.
 - Hash-based indexes only good for equality search.
 - B+ Tree indexes best for range search; also good for equality search.

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 14

Summary (Contd.)

- Understanding the nature of the *workload* is essential to developing a good design.
 - What are the important queries and updates?
- Indexes should be chosen to speed up important queries.
 - Index maintenance overhead on updates to key fields.
 - Choose indexes that can help many queries, if possible.
 - Build indexes to support index-only strategies.
 - Clustering is an important decision.
 - Order of fields in composite index key can be important.

3/3/09 EECS 484: Database Management Systems, Kristen LeFevre 15



Announcement

- Minirel Project #1 (Buffer Manager) is posted
 - Project description on class website
 - Skeleton source code on Ctools
 - Same partners as last project
