Principles of Information and Database Management
198:336
Week 13 – May 2

Matthew Stone

Project Revision

Email just **links** to mdstone@cs

- Link to code (on the web)
- Link to writeup (on the web)
- Link to project site itself

Project Revision

Email just **links** to mdstone@cs

- Link to code (on the web)
- Link to writeup (on the web)
- Link to project site itself

Kill me.

Project Revision

Email just **links** to mdstone@cs

- Link to code (on the web)
- Link to writeup (on the web)
- Link to project site itself

Easy enough to add to e.g., tomcat dirs
(or wherever you keep web stuff)

OK to send tomorrow by 5pm.

Outline

Under the Hood: How Databases Work

- Files and indexes
- Join algorithms and query plans
- Locking and transactions

Final Review

Under the Hood

How do you handle all this data anyway?

- Data storage
- Data access
- Data update

Storage

Data is broken up into blocks on disk

- Basic units of input/output
- Few kilobytes
- Stores a few records or a little indexing informaiton

Storage

Tables are represented as files

- Arbitrarily large collections of blocks
- Organized so that system can work by reading in one block at a time
- Linked together so that the system can find a particular record

Example File Structures

Heap Files

- Records are added in order to the collection of blocks
- Fast to insert an item
- Slow to search for an item

Example File Structures

Clustered file

- Tree indexes the underlying data by key
- Leaves of this tree are blocks directly containing many data records
- Fast to find data with keys in a specific range
- Fast to insert/delete
- Fast to search entire DB in order

Example File Structures

Index into heap file

- Tree indexes underlying data by key
- Leaf nodes point to records stored in another block order
- Becomes expensive to scan data in order, because each record needs its own disk I/O

Leads to Lots of Interesting Qs

How do you sort more records than can fit into memory?

- Mergesort on disk!

How do you decide what is the best structure?

- Depends on what you do with the DB!

Plus issues for DB Management

You can say what kinds of index your DB should have

- Adds space
- Adds time to insert
- May save time in queries
- Part of DB administrator's expertise

Join Algorithms and Query Plans

What's the best way to answer queries?

- Given the particular way the data is stored
- Given what you can fit into memory at once

Join Algorithms

Nested loops:

– Foreach tuple r in R

Foreach tuple s in S do

if ri = sj then add <r,s> to result

Can be carefully optimized to minimize disk I/O (by looking at blocks together) and minimize CPU using hashing

Join Algorithms

Can exploit index on one relation to avoid an inner loop,

– though looking tuples up separately with the index may be slower than reading through them all at once.

Join Algorithms

Can just sort based on the join key and merge the resulting lists

- Works really well if tuples in the relation correspond one-to-one.

Join Algorithms

What's the right strategy?

- It depends
- Optimizers have many different join algorithms at their disposal
- They pick the one they think will make the overall query take the least time.

Leads to lots of interesting issues

Modeling performance

Searching for the best query plans

Making Transactions Work

Defining protocols for access

- 2 phase locking
- To read an object, T requests a shared lock
- To write an object, T requests an exclusive lock on it.
- T's locks are released when it is completed.

Making Transactions Work

Tradeoff between safety and throughput

- 2PL asks for many more locks than “strictly” needed in many applications
- Concurrency may be too important

Interesting Questions

Isolation levels:

- Modeling kinds of transaction errors you’re willing to tolerate
- The need for speed!

Putting locks into our crazy indexed file reps.

The Point

Lots of cool hairy stuff
- If you like that sort of thing.

Whirlwind tour of Chapters 8, 14, 17.

Final Review

Topics for exam questions

Principles and techniques

Big Picture

These lists are not exhaustive – but they do prioritize what's most important.

Topics for Exam Questions

Problems in representation

- Draw E-R diagrams
- Formalize networks of concepts
- Design or use relational schemas
- Design semi-structured data representations
- Give information in relational tables
- Represent information in XML, XHTML
- Formulate SQL queries
- Describe XML nodes and paths

Topics for Exam Questions

Problems working with algorithms

- Describe what happens where on the web
- Calculate results of SQL statements
- Trace execution of concurrent DB actions and fix bugs with transactions
- Construct HTML to encode interaction
- Describe results of HTML interaction
- Know what makes documents similar in IR
- Know how web pages get high rank
- Find frequent itemsets, verify CART rules

Topics for Exam Questions

Problems in applying course principles

- Characterize system as information manager
- Identify information needed for a problem
- Choose the right technology
- Relate some data, a model and reality
- Fit information into an application/business
- Design a web interaction

What This Course Taught

Interacting with information as CS

- Getting information from the world
- Storing information
- Querying information
- Visualizing and using the results

What This Course Taught

Kinds of data

- Relational Model
  Tabular data, SQL
- Semi-structured Model
  Tree data, XML
- Text and graph data
  Vector space model + similarity
  Link analysis + page rank

What This Course Taught

How to build data-intensive applications

- Software engineering methodology
- Web architecture and design
- Hands-on experience with code, platforms

Big Picture

Useful practical skills

- I see flyers everywhere recruiting ACCESS/PHP/XHTML hackers

Empowerment