EECS 584: Advanced Database Systems

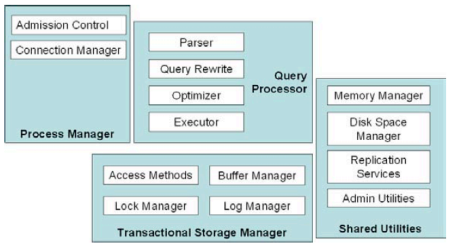
Prof. Michael Cafarella
michjc [at] umich.edu
4709 CSE

Some slides thanks to K. LeFevre

Today's topics

- Anatomy of a Database System
- Operating System Support for Database Management

DBMS Architecture



Process Management

- Process-per-connection
- Multi-threaded server process
- Server process + I/O processes
- Pros/cons of each?

Data Passing

- Threads/processes need to share data
- Disk I/O buffers
  - Buffer pool (dirtied and flushed to disk)
  - Log I/O requests
- Client communication buffers not a huge issue

Threads & Processes

- When DBMSes invented, OS did not offer threads or disk-buffering
  - DBMS invented a lot of its own machinery
  - DBMSes retain custom code, despite OS advances
- Some DBMSes actually *multiprocess*
  - One process per async component
  - Largely for historical reasons
  - Allocating per-thread probably better

Notes on External Storage

- *Typically, databases are assumed to be larger than main memory. Is this a good idea?*
- Disks: Can retrieve random page at fixed cost
  - But reading several consecutive pages is much cheaper than reading them in random order
- File organization: Method of arranging a file of records on external storage.
  - Record id (rid) is sufficient to physically locate record
  - Indexes are data structures that allow us to find the record ids of records with given values in index search key fields
- Architecture: Buffer manager stages pages from external storage to main memory buffer pool. File and index layers make calls to the buffer manager.

Notes on Buffer Management

- Data read into memory for processing, and written to disk for persistent storage by layer called the buffer manager
- Buffer manager maintains a pool of pages in memory
- Buffer manager implements its own page replacement (separate from OS's virtual memory, file system buffer)

Storage Basics

- Modern disks:
  - 5ms seek time
  - 50-120 MB/sec sustained reads
  - A lot of disk management is avoiding seeks
- Memory:
  - Bandwidth in 4-6 GB/sec range (max bandwidth rarely seen in practice)
  - Access latency high and not improving
  - Caches are critical

Storage

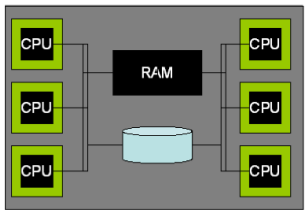
- Disk control
  - OS-handled
  - Raw interface
- Buffer Management
  - OS performs read-ahead or write-behind
  - Match to DBMS workloads?
  - Replacement policy?

Parallel DBMSes

- More on this in later lectures
- A few basic models…

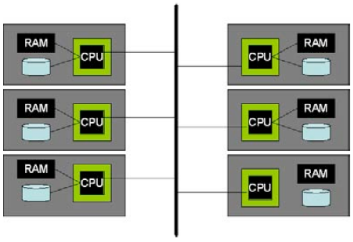
Parallel DBMSes

- Shared memory:



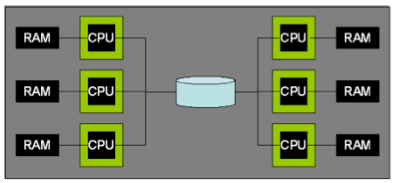
Parallel DBMSes

- Shared nothing:



Parallel DBMSes

- Shared disk:



Parallel DBMSes

- Non-Uniform Memory Access (NUMA)
  - As exotic as the paper claims?
- Trends since this paper was written?

Query Processing

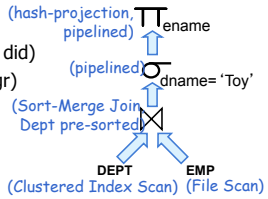
- Parsing and metadata checking
- Rewrite
  - View rewriting
  - Constant evaluation, logical rewriting
  - Semantic optimization (using integrity constraints)
  - Subquery flattening, heuristics
- Optimization
- Execution

Query Optimization

- Important component of a DBMS
  - Convert SQL query *blocks* to extended relational algebra expressions
  - Enumerate alternative plans
  - Choose a plan based on estimated cost
    - Cardinality & selectivity estimation
- Impact on parallel DBMSes?
- What about administrative overhead?

Query Plan Example

EMP (ssn, ename, addr, sal, did)
DEPT (did, dname, floor, mgr)



Annotated RA Tree

```
SELECT DISTINCT E.ename
FROM Emp E, Dept D
WHERE D.dname = 'Toy'
AND D.did = E.did
```

Query Execution

- Query plan is annotated RA tree
- Translate RA operators into iterator operators

```
class iterator {
  iterator &inputs[];
  void init();
  tuple get_next();
  void close();
}
```

- Benefits: pipeliness, w/temporary materialization where necessary

Alternative File Organizations

Many alternatives exist, *each ideal for some situations, and not so good in others:*

- Heap (random order) files: Suitable when typical access is a file scan retrieving all records.
- Sorted Files: Best if records must be retrieved in some order, or only a 'range' of records is needed.
- Indexes: Data structures to organize records via trees or hashing.
  - Like sorted files, they speed up searches for a subset of records, based on values in certain ("search key") fields
  - Updates are much faster than in sorted files.

Indexes

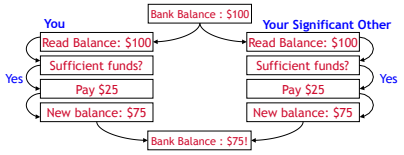
- An *index* on a file speeds up selections on the *search key fields* for the index.
  - Any subset of the fields of a relation can be the search key for an index on the relation.
- An index contains a collection of *data entries*, and supports efficient retrieval of all data entries **k*** with a given key value **k**.
- Index Types from 484
  - B+ Trees
  - Hash-based Indexes

Transactions

- Foundation for concurrent execution and recovery in DBMS
- Transaction is an atomic unit of work
  - E.g., Debit $500 from my bank account
- Transaction consists of multiple actions
- For performance, DBMS can interleave actions from different transactions
- Must guarantee same result as executing transactions serially

Example - Concurrent Execution

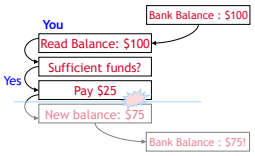
Read (A);
Check (A > $25);
Pay ($25);
A = A - 25;
Write (A);



- Interleaving actions of different transactions can cause inconsistency
- DBMS should provide users an illusion of a single-user system
- Could insist on admitting only one transaction at a time
  - Lower utilization: CPU / IO overlap
  - Long running queries starve other queries, reduce overall response time

Example - Crash Recovery

Read (A);
Check (A > $25);
Pay ($25);
A = A - 25;
Write (A);



- DBMS must also guarantee that changes made by partially completed transactions are not seen by other transactions

The ACID Properties

- Atomicity: All actions in the Xact happen, or none happen.
- Consistency: Consistent DB + consistent Xact ⇒ consistent DB
- Isolation: Execution of one Xact is isolated from that of other Xacts.
- Durability: If a Xact commits, its effects persist.

DBMS Support for Txs

- Touches every part of DBMS
- Lock Manager maintains tables of locks and transactions
- Different isolation levels:
  - READ UNCOMMITTED
  - READ COMMITTED
  - REPEATABLE READ
  - SERIALIZABLE
- WAL critical for recovery

OS Support for DBMS

- Buffer Pool Management
- Filesystem
- Scheduling, Processes
- Consistency
- Virtual Memory