Announcements

- Homework #4 due in 2 days (April 23)
- Recitation session this Friday or next Monday?
  - Homework #4 and final review
  - Will announce on newsgroup
- Project demos start next week (April 28)
  - Sign-up sheet will be circulated
- Final exam in 10 days (D243; Thursday, May 1, 2-5pm)
  - Comprehensive (everything up to today’s lecture, with emphasis on the second half of the course)
  - Open book, open notes
  - No time pressure!
- Please fill out course evaluations at the end of this lecture

Review: basics

- Relational model/algebra → physical data independence
- Design theory (FD’s, BCNF) → help eliminate redundancy
- SQL
  - NULL and three-value logic → nifty feature, big mess
  - Bag versus set semantics
  - Subqueries, grouping and aggregation
  - Views → logical data independence
    - Materialized views → reintroduce redundancy to improve performance
  - Constraints → the more you know the better you can do
  - Triggers (ECA) → “active” data
  - Transactions and isolation levels

Review: physical data organization

- Storage hierarchy (DC vs. Pluto) → count I/O’s
- Data layout
  - Record layout (handling variable-length fields, NULL’s)
  - Block layout (NSM, DSM, PAX) → inter-/intra-record locality
- Access paths
  - Primary versus secondary indexes → again, reintroduce redundancy to improve performance
  - Tree-based indexes: ISAM, B⁺-tree, R-tree, GIST
  - Hash-based indexes: extensible, linear
  - Text indexes: inverted lists, signature files
  - Variant indexes: bitmap, projection, bit-sliced indexes
  - Main-memory indexes: T-index, CSS and CSB⁺-trees
    → Fundamental trade-off: query versus update cost

Review: query processing

- Buffer management
  - Per-query, per-table policy is ideal
    → The more you know the better you can do
- Sort- and hash-based algorithms (and their duality)
- Index-based algorithms
- Pipelined execution with iterators

Review: query optimization or “goodification”?

- Heuristics: push selections down; smaller joins first
  → Reduce the size of intermediate results
- Cost-based
  - Query rewrite: merge blocks to get a bigger search space
  - Cost estimation: use statistics (e.g., histograms)
  - Search algorithm: dynamic programming (+ interesting orders), randomized search, generic programming, etc.
- Online query processing: change in cost metric—throughput ≠ user satisfaction
  → Trade-off: throughput versus responsiveness
- Adaptive query processing: adapt on a per-tuple basis
  → The sooner you know the better you can do
  → Trade-off: overhead versus adaptivity

Review: transaction processing

- ACID properties
- Concurrency control
  - Locking-based: strict 2PL; handling deadlocks; multiple-granularity locking; predicate locking and tree locking
  - Validation-based, timestamp-based, multi-version
    → Trade-off: blocking versus aborts and restarts
- Recovery
  - Steal: requires undo logging
  - No force: requires redo logging
  - WAL (log holds the truth)