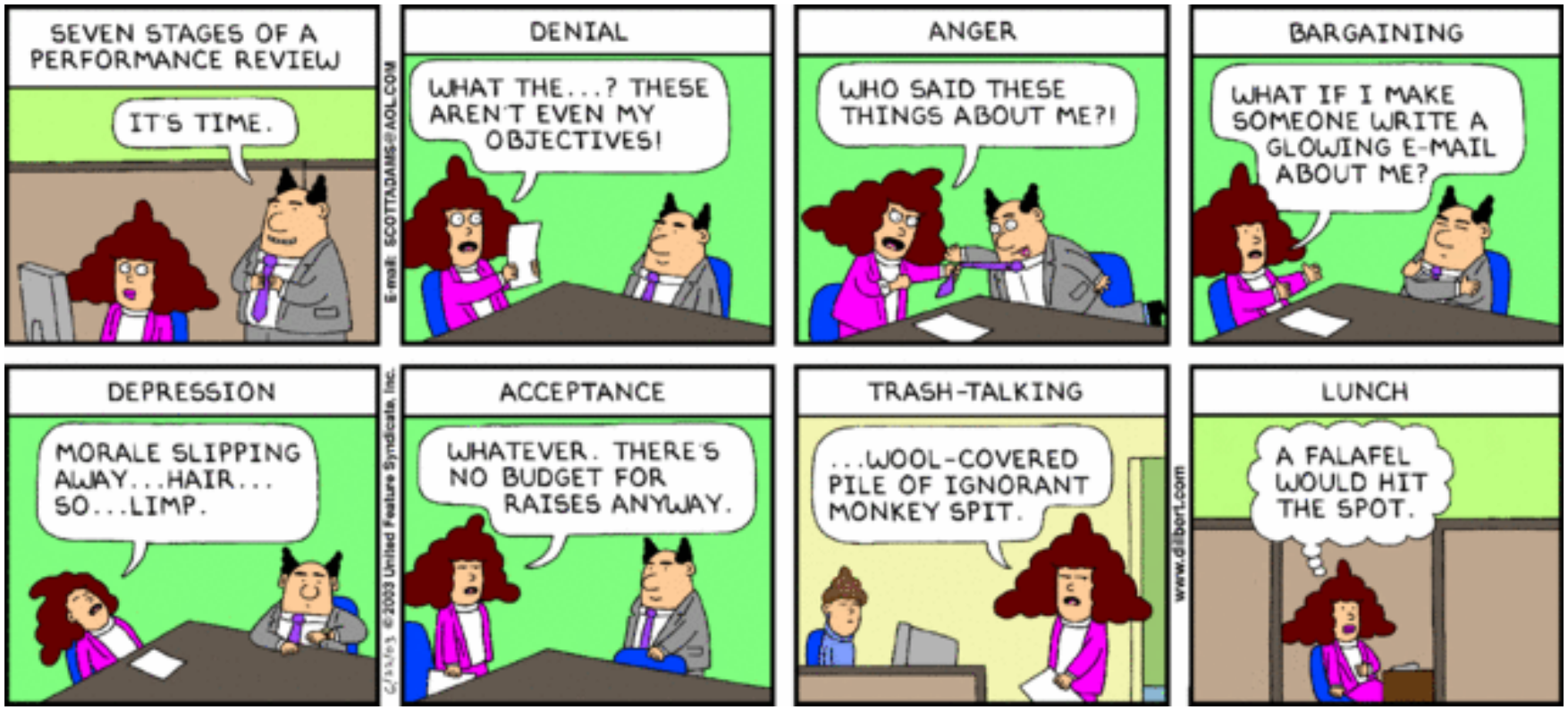
Please fill out our course eval on ACES!



Final Review

Introduction to Databases

CompSci 316 Fall 2015

DUKE
COMPUTER SCIENCE

Announcements (Tue., Dec. 1)

- Remember course evals on ACES!
- Homework #4 due today
  - Problem 4 (Gradiance) due tomorrow
  - Sample solutions to be posted by Thursday
- Project demos start Thursday
  - Schedule to be finalized tonight
  - Submit your report and code by your demo slot
- Final exam Wednesday 7-10pm
  - Open-book, open-notes
  - Focuses on the second half of the course
  - No communication or Internet use (besides accessing materials on the course website)
  - Last year's exam and sample solution posted

Relational basics

- Relational model + query languages: physical data independence
- Relation algebra (set semantics)
- SQL (bag semantics by default)
- Schema design
  - Entity-relationship design
  - Theory (FD's, MVD's, BNCF, 4NF): help eliminate redundancy

More about SQL

- `NULL` and three-valued logic: nifty but messy
- Bag vs. set: beware of broken equivalences
- `SELECT-FROM-WHERE` (SPJ)
- Grouping, aggregation, ordering
- Subqueries (including correlated ones)
- Modifications
- Constraints: the more you know the better
- Triggers (ECA): "active" data
- Index: reintroduce redundancy for performance
- Transactions and isolation levels

XML

- Data model: well-formed vs. DTD vs. XML Schema
- Query languages:
  - XPath: (branching) path expressions (with conditions)
    - Be careful about the semantics of overloaded operators on sets
  - XQuery: FLWOR, subqueries in return (restructuring output), quantified expressions, aggregation, ordering
  - XSLT: structural recursion with templates
- Programming: SAX (streaming) vs. DOM (in-memory)
- Relational vs. XML
  - Tables vs. hierarchies
  - Highly structured/typed vs. less
  - Join vs. path traversals
  - Storing XML as relations: various mapping methods

Physical data organization

- Storage hierarchy (DC vs. Pluto): so count I/Os!
- Hard drives: geometry → three components of access cost; random vs. sequential I/O
- Solid state drives: faster, but still far slower than memory; also block-oriented access
- Data layout
- Access paths (indexing)
  - Primary vs. secondary; sparse vs. dense
  - Tree-based indexes: ISAM, B^+-tree
    - Big fan-out: do as much as you can with one I/O
  - Again, reintroduce redundancy to improve performance, but keep in mind the query vs. update cost trade-off

Query processing & optimization

- Processing
  - Scan-, sort-, hash-, and index-based algorithms
    - Do as much as you can with each I/O
    - Manage memory very carefully
  - Pipelined execution vs. materialization
- Optimization (or "goodification")
  - Heuristics: push selections down; smaller joins first
    - Reduce the size of intermediate results
  - Cost-based
    - Query rewrite: de-correlate and merge query blocks to expand search space
    - Cost estimation: comes down to estimating size of intermediate results; statistics + assumptions
    - Search algorithms: greedy vs. dynamic programming (with interesting orders)

Transaction processing

- ACID
- Concurrency control
  - Serial and conflict-serializable scheduled
  - Locking-based: 2PL and strict 2PL
- Recovery with logging
  - Steal: requires undo logging
  - No force: requires redo logging
  - WAL: log holds the truth
  - Fuzzy checkpointing