# DB101 – Course overview

Goetz Graefe – Madison, Wis.

# Why use databases and database software?

Sharing structured data

1. Reliable storage – protection against data loss
2. Privacy protection, security, and retention
3. Concurrency control – protection of data consistency
4. Schema – common understanding of the bits and bytes
5. Physical data independence – changing storage formats, automatic mapping from tables to indexes, files, etc.
6. Controlled redundancy – consistency
7. Query processing – simple data transformations from storage to application

# Why touches database research & development?

- Programming languages, parsing, data abstraction
- Planning (under uncertainty), statistics, sampling
- Algorithms & data structures
- Storage, storage structures, file systems
- Operating systems, scheduling, resource management
- Networking, distributed systems
- Parallelism, high-performance computing
- Fault-tolerant computing, redundancy, failures & recovery
- Security, privacy, randomization
- Theory (normal forms, constraints, concurrency control)

# Course agenda

1. Sorting
2. Transactions
3. Distributed commit
4. Storage formats
5. Consistency checking
6. Query processing
7. Robust query performance
8. Streaming

# Topics omitted

1. Application design, deployment, testing, debugging, tuning, maintenance, regression testing
2. NoSQL databases (i.e., “not” SQL & “not only” SQL)
3. Business intelligence, OLAP, cubes, analytics
4. Security, privacy, compliance
5. Testing & deployment & monitoring at scale
6. Cloud deployments, virtual storage and processing
7. Self-management & auto-tuning, automatic indexes & constraints
8. Performance metrics, benchmarks, regression testing
9. Data cleaning, entity matching, etc.
10. Machine learning: ML for DB, DB for ML
11. Database theory, database design, serializability
12. Database machines, hardware support
13. Disaster preparedness & recovery & testing

# CS 764 entry quiz

1. How do you spell SQL?
2. What is the most central concept in relational databases?
3. What is an integrity constraint?
4. What is a normal form?
5. What is a b-tree?
6. What is physical data independence?
7. What is physical database design?
8. What is a join of two tables?
9. What algorithms can compute a join?

# CS764 fall 2023: implement an external merge sort

- 1 M × 50 B = 50 MB, 2.5 M × 50 B = 125 MB
  12 M × 1 KB = 12 GB, 120 M × 1 KB = 120 GB
- 1 CPU core, 1 MB cache, 100 MB DRAM
  SSD: 10 GB capacity, 0.1 ms latency, 100 MB/s bandwidth
  HDD: ∞ capacity, 10 ms latency, 100 MB/s bandwidth
  Emulate SSD + HDD, report total latency & transfer time
- Extra credit: logic & performance evaluation for
  - in-stream (after-sort) 'distinct', 'group by', or 'top'
  - in-sort 'distinct', 'group by', or 'top'
- Provided: iterator template & logic

# Techniques to consider in external merge sort

1. Quicksort?
2. Tournament trees
3. Replacement selection
4. Run size > memory size?
5. Offset-value coding
6. Variable-size records?
7. Compression?
8. Prefix truncation?
9. Minimum count of row & column comparisons
10. Cache-size mini runs
11. Device-optimized page sizes
12. Spilling memory-to-SSD
13. Spilling from SSD to disk
14. Graceful degradation
    a. into merging
    b. beyond one merge step
15. Optimized merge patterns
16. Verifying
    a. sets of rows & values
    b. sort order