Goals for Today

- Analytics queries and systems
- Brief overview of course
- Course evaluations

Data Analytics and Decision Support

- **Idea**: current and historical data to identify useful patterns and support business strategies
- Complex, interactive, exploratory analysis of data
  - Large datasets
  - Data integrated from across all parts of an enterprise
  - Data is fairly static
- **OLAP**: on-line analytical processing

OLAP vs. OLTP

- **OLTP** (on-line transactional processing)
  - Update-heavy
  - Short, simple transactions
  - Goal: transaction throughput
- **OLAP**
  - Mostly reads
  - Longer, complex queries for analysis and decision-making
  - Goal: fast queries
Data Integration

- Data may reside in many distributed, heterogeneous OLTP sources
  - Sales, inventory, customer, ...
  - NC branch, NY branch, CA branch, ...

- Need to support OLAP over integrated view of the data

- Possible approaches to integration
  - **Eager**: integrate in advance and store the integrated data in a data warehouse
  - **Lazy**: integrate on demand; process queries over distributed sources—mediated or federated systems

Example: Car Sales Schema

- Example car sales schema:
  - Cars(serialNo, make, model, color)
  - Dealers(name, city, state, phone)
  - Date(date, day, week, quarter, month, year)
  - Sales(serialNo, date, dealer, price)

Star Schema in Relational OLAP (ROLAP) System

- Fact table **BCNF**; dimension tables possibly denormalized
  - Dimension tables are small; updates/inserts/deletes are rare.... anomalies less important than performance

- **Star Schema**

A Multidimensional View

- Example car sales schema:
  - Cars(serialNo, model, color)
  - Dealers(name, city, state, phone)
  - Date(date, day, week, month, year)
  - Sales(serialNo, date, dealer, price)
Dicing the Cube

- Can think of partitioning the raw data cube along each dimension at some level of granularity

- A choice of partition for each dimension “dices” the cube

Slicing the Cube

- Idea: want info about a fixed slice of the data

- In general, in SQL:
  - Dice: GROUP BY
  - Slice: WHERE

Example: Data Analysis

- Suppose Mazda3 model is not selling as well as anticipated

- Query: which colors not doing well?

```sql
SELECT color, SUM(price)
FROM Sales NATURAL JOIN Cars
WHERE model = "Mazda3"
GROUP BY color;
```

OLAP Queries

- A common operation is to aggregate a measure over one or more dimensions.

- Roll-up: Aggregating at coarser granularity, e.g., higher level in dimension hierarchy.

- Drill-down: The inverse of roll-up

Specialized MOLAP and ROLAP systems may store pre-aggregated data (materialized views)
Analyzing Big Data: Current Trends

• Motivation
  – Expensive ROLAP and MOLAP systems not for everyone
  – Desire to analyze semi-structured or unstructured data

• Big Data rampant!
  – E.g., data sets generated by some of the applications backed by NoSQL systems
  – Sensor data, tweets, etc.

• Trend: many people using MapReduce/Hadoop for Big Data Analysis
  – Scalability and commodity hardware

Open-source version of Google's MapReduce

Example: Word Count

```python
def mapper(line):
    for word in line.split():
        output(word, 1)

def reducer(key, values):
    output(key, sum(values))
```

MapReduce Programming Model

• Data type: key-value records

• Map function:
  \[(K_{in}, V_{in}) \rightarrow \text{list}(K_{intermediate}, V_{intermediate})\]

• Reduce function:
  \[(K_{intermediate}, \text{list}(V_{intermediate})) \rightarrow \text{list}(K_{out}, V_{out})\]

Word Count Execution

Input → Map → Shuffle & Sort → Reduce → Output

Thanks Mike Franklin & Matei Zaharia
Final Exam: Logistics

- **Take-home exam**
  - Available to you Friday December 9th, 4pm (printed, Olin hallway cubbies)
  - Due to my office (Olin 1267) at or before: Thursday, December 15th, 12pm (noon)

- **Two 8.5x11, double-sided note sheets**
  - You can use your note sheet from the midterm as one of the two
  - No other resources

- **3-hour timed exam**

Why Use a DBMS?

- **Data independence and efficient access**
  - Declarative language: say what you want, not how to get it!
  - Hides complexity from user
  - Accommodates changes in database without requiring applications to be recompiled

- **Data integrity and security**

- **Concurrent access, recovery from system crashes.**

Simplified RDBMS Architecture

```
```

Optimize the Memory Hierarchy

- **DBMS worries about Disk vs. Memory**
  - Can spend a lot of CPU cycles thinking about how to best fetch data from disk
    - e.g., query optimization, buffer replacement strategies
  - I/O cost “hides” the CPU think time

“Bottom-up” order of topics to show role of abstraction and algorithms for efficiency/optimization
Practical Algorithm Analysis

- Due to need for query cost estimation, DBMS developers understand the real costs of their main algorithms
  - e.g., external sorting

- In many applications, the bottlenecks determine the cost model
  - E.g., I/O is mostly what matters in DBs
  - Affects the practical analysis of the algorithm

Query Operators & Optimization

- Query operators are actually all similar:
  - Iterator model approach to building query plans

- Query Optimization steps
  - Define a plan space
  - Estimate costs for plans
  - Algorithm to search in the plan space for cheapest

ACID Transactions

- Concurrency and reliability
  - Transactions and 2-Phase Locking
  - Write-ahead-logging to ensure consistency if system crashes

Database Design

- Components
  - Conceptual design
  - Schema refinement
  - Physical design

- Complex!
  - Modeling real-life application
  - Tools like BCNF normalization to help avoid anomalies
  - Heuristics and tradeoffs

Tradeoffs in distributed setting!
Course Objectives

- Provide a solid background in database management system design principles
- Promote understanding of these principles through hands-on exercises implementing the internals of a relational database management system
- Further develop students' ability to reason about algorithm and software design, optimization, and tradeoffs generally applicable in computer science

Possible Topics on Final

- Cumulative-ish
  - Topics we covered earlier still relevant (e.g., hash & tree indexes, estimating cost in I/Os)
  - Won’t focus on nitty gritty from before midterm (e.g., linear vs extendible hashing)
- Query Optimization
- Transactions and ACID
- Database design
- ORDBMS, Distributed DBMS and NoSQL, OLAP (high-level)
- General themes
  - Reasoning about cost and tradeoffs
  - Consistency and correctness with concurrent access and failures

Query Optimization

- Query
  - relational algebra tree
  - logical plan
  - physical plan
- Unit of optimization: query block
- Logical plan
  - Relational algebra equivalences
  - Outer vs. inner relation in joins
  - Query plan tree shape: bushy, linear, deep

Query Optimization

- Choosing physical plan
  - Enumerate plan space
    - Join permutations and orders
    - System R choices
  - Estimate cost of plan
  - Picking cheapest
    - Dynamic programming algorithm (idea)
    - Interesting orders
- Cost estimation
  - Operator algorithm cost
    - Estimating cost of different join algorithms
  - Operator result size estimation
    - Selectivity/Reduction Factor, statistics, histograms
    - Using indexes
ACID Transactions

• Transactions, how to achieve ACID

• Isolation (I)
  — Schedules: serializable, conflict-serializable, etc.
  — Anomalies from interleaved actions, conflicting actions
  — Locking, lock granularity and compatibility, deadlock detection and prevention
  — 2PL vs Strict 2PL, cascading aborts
  — Optimistic concurrency control, backwards validation algorithm

• Recovery (A and D)
  — Steal vs. force and implications on UNDO/REDO
  — Write-Ahead-Logging
  — ARIES recovery algorithm

Database design

• E/R modeling (general idea)
  — Entities, relationships, weak entities
  — Capturing key and participation constraints

• Functional dependencies
  — Attribute closure, Armstrong’s axioms
  — Determining candidate keys
  — Role in detecting data redundancy

• Schema refinement
  — Normalization
  — BCNF normalization process

• Capturing integrity constraints in relational schema

• General motivation and ideas from ORDBMS

Special Topics

• Distributed DBMS
  — Goals of data partitioning and data replication
    • Types of partitioning: range vs hash
  — Replication
    • Synchronous vs asynchronous
    • Strong vs. eventual/weak consistency
  — Challenges with distributed xacts (generally)

• NoSQL
  — CAP theorem
  — Query restrictions for performance (generally)

• Analytics
  — Generally what OLAP is, vs. OLTP, and what kinds of queries run