Comp 115: Databases

Database System Architectures

Instructor: Manos Athanassoulis

http://www.cs.tufts.edu/comp/115/
Today

logistics, goals, admin

database systems architectures

project details

when you see this, I want you to speak up! [and you can always interrupt me]

no smartphones

no laptop
Course Scope

A detailed look “under the hood” of a DBMS

why?

applications writers, data scientists
database researchers, db admins

they all *understand* the internals

there is a huge need for database experts
data-intensive applications
big data workflows
Course Scope: Practical Side

use
benchmark
develop
database systems!

More details when discussing the project!
Readings

“Cowbook”
by Ramakrishnan & Gehrke

Additional Readings

Architecture of a Database System, by J. Hellerstein, M. Stonebraker and J. Hamilton


Modern B-Tree Techniques, by Goetz Graefe, Foundations and Trends in Databases, 2011

+research papers
Guest Lectures

We will have a couple guest lectures

Make sure to attend!

Will be notified ahead of time.
Evaluation

Class Participation: 5%

In-class discussion

Collaborative Notes
1-2 students take notes on shared gdoc
2 days after the class anybody can augment it
http://tinyurl.com/Comp115-2017s-Notes
[top part of website as well]

Enroll right after class!
Evaluation

Class Participation: 5%
Homeworks: 15%

First half of the semester
[tentatively] on:
ER model & Relational Model
Normalization
Relational Algebra
SQL
Evaluation

Class Participation: 5%
Homeworks: 15%
Project 0: 5%

First quarter of the semester
due week 4-5
[more details later today]
Evaluation

Class Participation: 5%
Homeworks: 15%
Project 0: 5%
Project: 30%

Mostly on second half of the semester
due end of the semester
[more details later today]
Evaluation

Class Participation: 5%
Homeworks: 15%
  Project 0: 5%
  Project: 30%
Midterm 1: 20%
Midterm 2: 25%

both exams during the semester
Evaluation

Class Participation: 5%
Homeworks: 15%
   Project 0: 5%
   Project: 30%
   Midterm 1: 20%
   Midterm 2: 25%

SQL Hands-On Test (bonus): 5%

Yes! you will use your laptop in class (this once)
Office Hours

Manos (after class)
M/W Halligan Hall 228B 6-7:15pm

TAs (will announce in Piazza soon)
Database Systems

I want “blah”

there you go

a declarative box

why having a declarative box is useful?
Database Systems

I want “blah”

there you go

a declarative box

application and backend development are independent
collection of algorithms & data structures

multiple ways to do the same thing

**optimization**: dynamically decide which to use

how?
collection of algorithms & data structures

multiple ways to do the same thing

**optimization**: dynamically decide which to use

how? understand & model alternatives
data management goals
data management goals

Application

monetary cost

performance

DBMS

energy

DATA

hardware
“three things are important in the database world: performance, performance, and performance”

Bruce Lindsay, IBM Research
ACM SIGMOD Edgar F. Codd Innovations award 2012
but

datacenterknowledge.com, 2016
but

Savings: 620 billion kWh

datacenterknowledge.com, 2016
but

new hardware in the last 20 years

multi-core processors
multi-level cache memories
flash drives
SIMD instructions

...
Comp115
What is inside?
How it works?

performance on a declarative box
Database Systems Architecture

Physical storage very important for performance!

DBMS: a set of cooperating software modules
Some questions for today

how can we physically store our (relational) data?

how to efficiently access the data?

does that affect the way we *ask* queries?

does that affect the way we *evaluate* queries?

does that affect the way we apply *updates*?
how to physically store data?

what is a relation?

a table with **rows** & **columns**!

how to physically store it?
how to physically store data?

one row at a time
how to efficiently access data?

how to retrieve rows:

if I am interested in the average GPA of all students?

if I am interested in the GPA of student A?
how to efficiently access data?

Scan the whole table

if I am interested in most of the data
how to efficiently access data?

how to retrieve rows:

if I am interested in the average GPA of all students?

if I am interested in the GPA of student A?
how to efficiently access data?

Ask an *oracle* to tell me where is my data

if I am interested in a single row
how to efficiently access data?

what is an *oracle* or *index*?

a data structure that given a value (e.g., student id) returns location (e.g., row id or a pointer) with less than $O(n)$ cost  

ideally $O(1)$!

e.g., B Tree, bitmap, hash index
how to efficiently access data?

Scan vs. Index

How to choose?
Model!

What are the parameters?
- data size
- index traversal cost
- access cost (random vs. sequential)
- result set size (“selectivity”)
how to efficiently access data?

Scan vs. Index

Scan: many rows

Index: few rows
how to physically store data?

is there another way?

one row at a time

columns first
how to efficiently access data?

if I want to read an entire single row?
  if I want to find the name of the younger student?
    if I want to calculate the average GPA?
      if I want the average GPA of all students with CS Major?
how to efficiently access data?

Rows vs. Columns

Rows: many attributes + few rows

Columns: few attributes + lots of rows
does that affect the way we *ask* queries?

I want “blah”

there you go

a declarative box

No!
does that affect the way we *evaluate* queries?

Query Engine *is* different

row-oriented systems ("row-stores")
move around rows

column-oriented systems ("column-stores")
move around columns
does that affect the way we *evaluate* queries?

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>DOB</th>
<th>Tel</th>
<th>email</th>
<th>GPA</th>
</tr>
</thead>
</table>

easy mapping from SQL to evaluation strategy

few basic operators: select, project, join, aggregate

simple logic for “query plan”
does that affect the way we *evaluate* queries?

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>DOB</th>
<th>Tel</th>
<th>email</th>
<th>GPA</th>
</tr>
</thead>
</table>

simpler basic operators

complicated query logic (more operators to connect)
does that affect the way we apply *updates*?

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>DOB</th>
<th>Tel</th>
<th>email</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Name</td>
<td>DOB</td>
<td>Tel</td>
<td>email</td>
<td>GPA</td>
</tr>
<tr>
<td>ID</td>
<td>Name</td>
<td>DOB</td>
<td>Tel</td>
<td>email</td>
<td>GPA</td>
</tr>
<tr>
<td>ID</td>
<td>Name</td>
<td>DOB</td>
<td>Tel</td>
<td>email</td>
<td>GPA</td>
</tr>
<tr>
<td>ID</td>
<td>Name</td>
<td>DOB</td>
<td>Tel</td>
<td>email</td>
<td>GPA</td>
</tr>
</tbody>
</table>

how to insert a new row?

how to delete a row?

how to change the GPA of a student?

how to update the email format of all students?
DBMS timeline

- **60s**
  - hand coded
  - first DB systems

- **80s**
  - OS
  - war OS vs. DBs
  - key concepts

- **00s**
  - declarative, abstractions, recovery, consistency
  - object-oriented DBs
  - perf & HW

- **today**
  - column stores, cache memories
  - multi-core flash
  - NoSQL vs. DBs
  - hybrid stores
  - NoSQL

- **today**
  - hybrid stores

- **today**
  - NoSQL vs. DBs
Row-Stores vs. Column-Stores

physical data layout

simple query plan vs. simple operators

“transactions” vs. ”analytics”
Other Architectures?

Key-Value Stores (NoSQL)
no transactions
data model: keys & values
row: a key and an arbitrarily complex value

Graph Stores
natural representation of graph links
data model: nodes & relationships
also maybe: weights, labels, properties
Project 0: column-stores vs. row-stores

**PostgreSQL**: open-source, widely used  
**MonetDB**: open-source, first academic system

*Groups of 2*
A will install & benchmark a column store (monetDB)  
B will install & benchmark a row store (postgres)

eexecute 8 queries (at your discretion) in each present performance (average, std. dev)  

discuss behavior of each system  

due date: week 4-5
Project

Implementation with C/C++

Two important components of a real system:

(1) Buffer Management
(2) B+ Tree

[more details soon]
Piazza

Announcements & Discussions in Piazza

https://piazza.com/tufts/spring2017/comp115
Remember & Next Time

database systems: performance (energy, HW)

physical storage (row-oriented vs. col-oriented) affects query engine/big design space

Project 0: benchmark row- and col-stores
Project: build internals [more soon]

Next: Modeling Data