

CS 561: Data Systems Architectures

Dr. Subhadeep Sarkar

ssarkar1@bu.edu

<https://bu-disc.github.io/CS561/>

no 
smartphones

no 
laptop

Why?

there is enough evidence that laptops and phones slow you down

Today

big data

data-driven world

data systems

which are the driving trends?

why do we need new designs?

CS 561 goals & logistics



I want you to speak up!
[and you can always interrupt me]

CS 561 philosophy

cutting-edge research

question everything (to understand it better!)

There are no stupid questions!

interactive & collaborative

projects, presentations, labs, OH



Understanding a design/system/algorithm ...

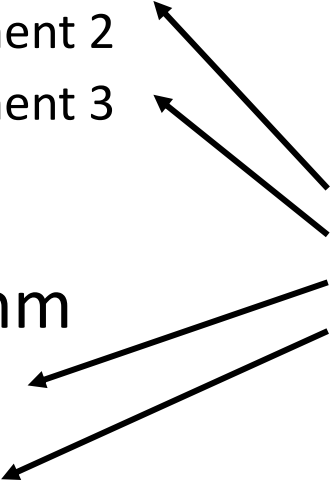
system

- component 1
- component 2
- component 3

algorithm

- step 1
- step 2
- step 3

why?
why not?



understanding all steps and all decisions
helps us see the ***big picture***
and do **good research!**

(otherwise, we make ad hoc choices!)



Ask Questions!

... and answer my questions!

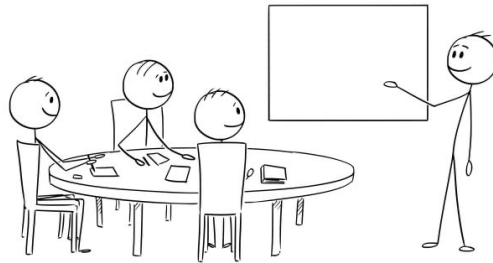
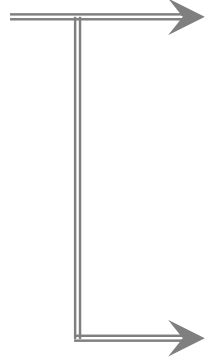
our **main goal** is to have **interesting discussions** that will help to gradually understand what the material discusses

(it's ok if not everything is clear, as long as you have questions!)

What do we do in this class?



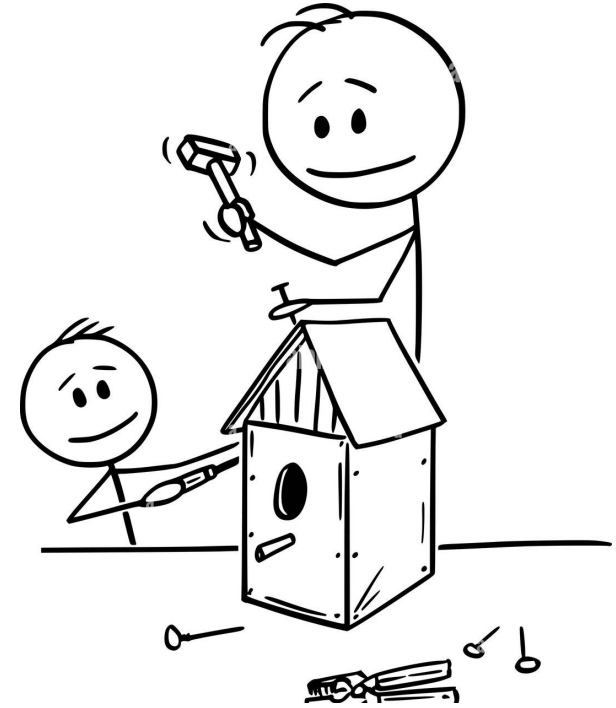
reading papers



presentations



reviews



projects

Reading Papers



every class **1-2 papers to discuss** in detail

in some classes the discussion will be led by a group of students

so that, each student will present one paper during the semester

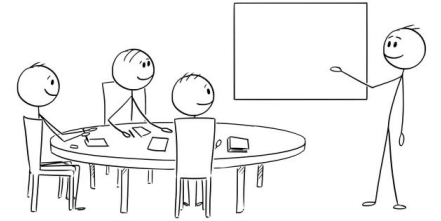
(background papers also available to provide more details)

read all of them!

write 4 reviews

answer one technical question per week (for a subset of the papers)

Presenting Papers



2-4 students will be responsible for presenting the paper
(discussing all main points of a review – see next slide)

during the presentation **anyone can ask questions** (including me!)
and each question is **addressed to all** (including me!)

prepare slides at least a **week before your presentation**



Writing Reviews

4 reviews and the 6 technical questions

(some weeks will be “free”!)

review (up to one page)

what is the problem & why it is important?

why is it hard & why older approaches are not enough?

what is the key idea and why it works?

what is missing and how can we improve this idea?

does the paper support its claims?

possible next steps of the work presented in the paper?

single technical question

to make sure the heart of the paper is clearly understood

learn

critic

remember, this will help us do **good research!**

Projects

project 0

A small implementation project
to sharpen dev skills

independent project



Due on Feb 2, 2023

AND

project 1

A medium project to give you a flavor of
large-scale production system

groups of 3



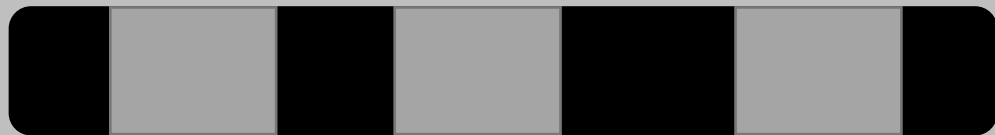
Projects

AND

project 0

A small implementation project
to sharpen dev skills

independent project



Due on Feb 2, 2023

project 1

A medium project to give you a flavor of
large-scale production system

groups of 3



Projects

systems project

groups of 3

implementation-heavy C/C++ project

```

01:0 cout<<endl<<endl<<"iterations #<< " <<< X1)<<X(2)<< " X(3);
0:0 cout<<endl<< " 0<<sw(1)<<j<<j<<sw(15)<<j2<<sw(14)<<j3;
float temp[3]; long float j1,j2,j3;
1:001 1:0101 0101 1001001 000111 0 001101 11011101 100111010
00001for(int s=1;s<=20;s++) 10 01 01100111 1100011000 0111010011010
cout<<endl<<"Formatting"<<endl;
0 0110 1 1 01 0 0 011 1011 0010111010 011000 0 0110111001 001 001
cout<<endl<<X(1) =";
1 0 0 0 1 temp[0]=1;temp[1]=2;temp[2]=3;01 0 000 0 1 00000101 010111
cin>>j1; cout<<endl<<X(2) =";
0110 1100 j1=(a[3]-a[1])*temp[1]-a[2]*temp[2];a[0]; 11011100110 11101010101
cin>>j2; cout<<endl<<X(3) =";
01110 0 0 j2=(b[3]-b[0])*temp[0]-b[2]*temp[2];b[1]; 001 01101101011 0001011
cin>>j3;
10011 110 j3=(c[3]-c[0])*temp[0]-c[1]*temp[1];c[2]; 1011 1011 0 011 1 0
000110100 cout<< " <<<sw(17)<<j1<<sw(15)<<j2<<sw(14)<<j3<<endl;
110 01 001 (j1==temp[0]&&j2==temp[1]&&j3==temp[2])
110101110 break;
100101101 0 0010111 0100 10111 0 011 01110100 0
for(int i=0;i<=3;i++)
0 }
0 00101110011011010
110//////////Function Of
011 Transcoding
0 void swap(float a[],float b[])
1 float temp[4];
00 11010111000010110111011101000101101011
00 100111010 01 01100110110000 0 000 cout<<"b1" =";
00 110101100000101110111011011100010001011010
011 cout<<"---PROCESSING---"<<endl;
011 cout<<"Preparing Encode X-Y con."<<endl;
1001 cout<<"Procedure starting"<<endl;
011 for(int i=0;i<=4;i++)
1:0 j temp[i]=a[i];
110 j temp[i]=b[i];
1:000 b[i]=temp[i];
1100 j }
0100 cout<<"---Parsing---"<<endl<<endl;
0100 cout<<"X-Y transcode"<<[0]<<X(1) + "<< [1]<<X(2) + 01110 cout<<" (3*<<k+1<<?)";
010 <<< [2]<<X(3) = "<< [3]<<endl;
11 0 110100 1011 101101 100
1 <<< [0]V transcode<< [0]<<X(1) + "<< [1]<<X(2)
10 << [3]<<endl;
20 <<< [3]<<endl;
10 <<< [3]<<endl;
10 <<< [3];
011 cout<<endl<<endl;
011 }

```

OR

research project

groups of 3

pick a subject (list will be available)

design & analysis

experimentation



Project theme: NoSQL key-value stores

... are everywhere



work on a *state-of-the-art* design

Research Project: open questions

quickly delete and free-up resources

tuning based on workload

exploit *data being sorted*

robust designs and tunings

come up with your **own topic!**

more on the website (soon)



A good project

- (1) has a clear plan by project proposal by **mid-February** (5%)
- (2) has significant preliminary work done by **mid-March** (5%)

evaluation at the **end of the semester** (30%)

- (i) present the key ideas of the implementation/new approach
- (ii) present a set of experiments supporting your claims

come to OH!

(more details for the projects in Class 4)

The ultimate reward!



Santiago, Chile





Class Goal

understand the internals of
data systems for data science

tune data systems through **adaptation** and **automation**

get acquainted with research in the area

Can I take this class?



background

programming
data structures
algorithms
comp. architecture

pre-req

CS460/660 & CS210
contact Subhadeep if not sure

how to be sure?

if familiar with most, then maybe!
if familiar with **none**, then no!

Next classes

Class 1-2

logistics, big data, data systems, trends and outlook

Class 3

more basics on data systems, systems classification, graph, cloud

Class 4

intro to class project

Class 5 and beyond

present and **discuss** research papers from Manos + students + guest lectures



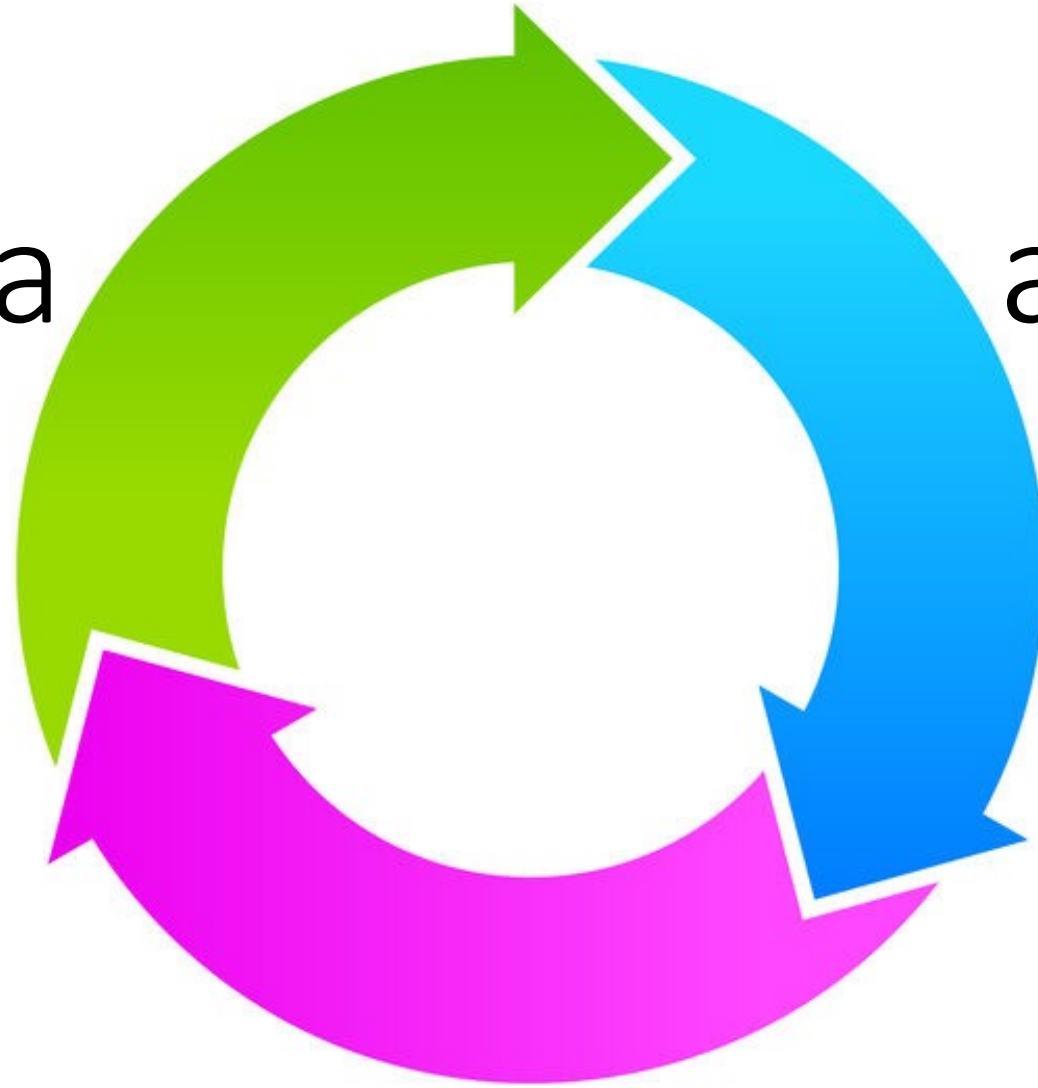
big data?

who doesn't have a lot of data?

So what do we do with this data?

data

analysis



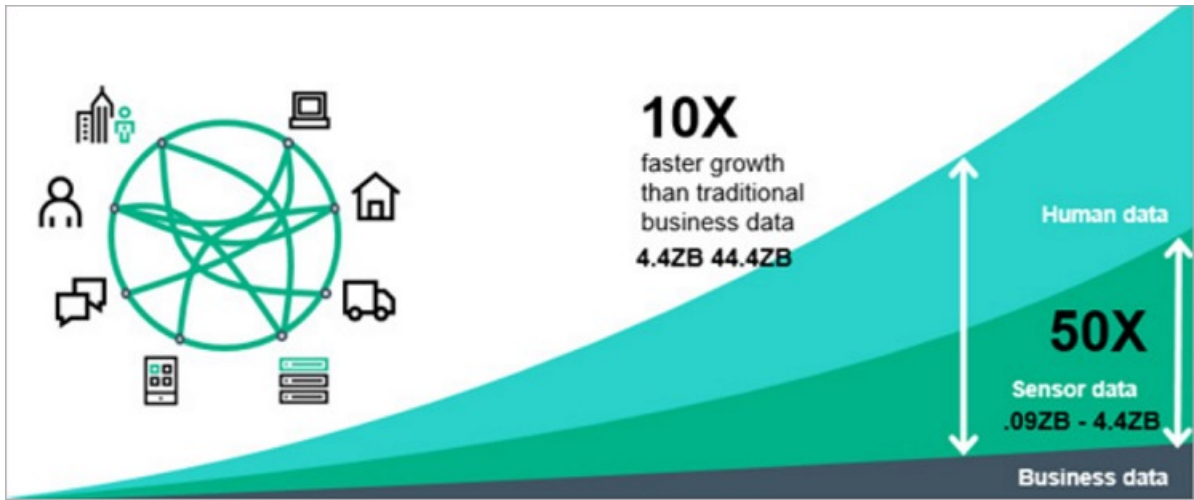
knowledge

is data
analysis new?



what is
really new?





Every day, we create 2.5 exabytes* of data — 90% of the data in the world today has been created in the last two years alone.

[Understanding Big Data, IBM]

*exabyte = 10^9 GB

DOMO

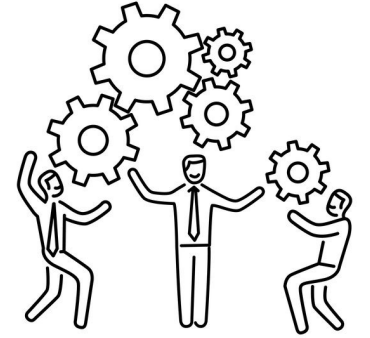
Data Never Sleeps 9.0

How much data is generated every minute?

The 2020 pandemic upended everything, from how we engage with each other to how we engage with brands and the digital world. At the same time, it transformed how we eat, how we work and how we entertain ourselves. Data never sleeps and it shows no signs of slowing down. In our 9th edition of the "Data Never Sleeps" infographic, we bring you a glimpse of how much data is created every digital minute in our increasingly data-driven world.



data management skills needed



100s of entries

pen & paper

10^3 - 10^6 of entries

UNIX tools and excel

10^9 of entries

custom solutions, programming

10^{12+} of entries

data systems

size (volume)

rate (velocity)

sources (variety)




big data

(it's not only about size)

all of the above plus ...


our ability to collect *machine-generated* data

 scientific experiments

 sensors

social 

monitoring 

 micro-payments

Internet-of-Things 

cloud 

data analysis

*know what we
are looking for*



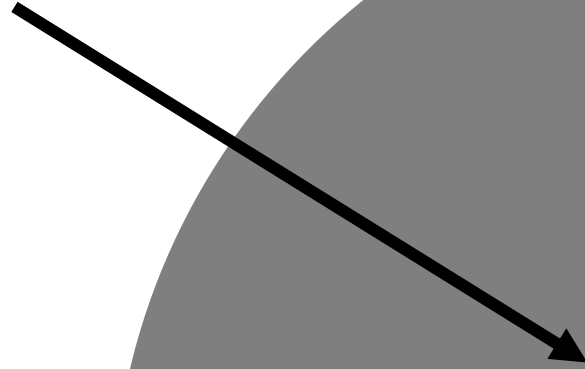
data exploration

*not sure what we
are looking for*



big data

data systems are
in the middle of this!



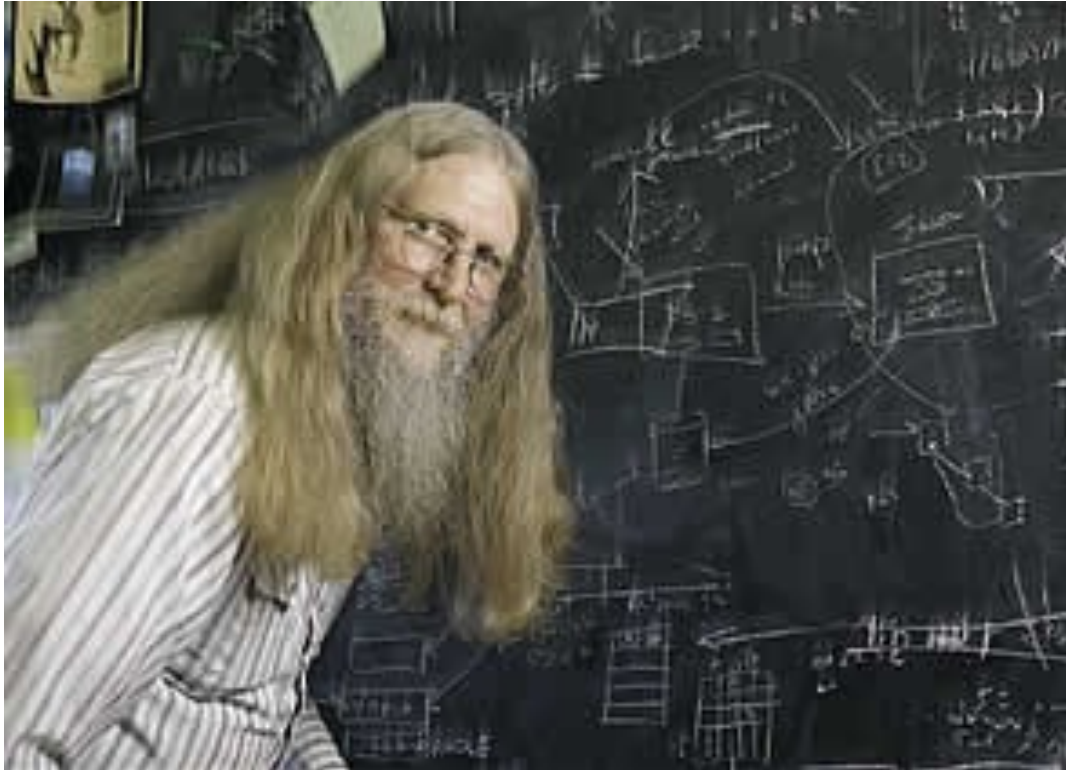
big data

**data
systems**

what is a data system?

a **data system** is a large software system
(a collection of algorithms and data structures)
that **stores data**, and provides the **interface** to
update and **access** them **efficiently**

the end goal is to make **data analysis** easy



*“relational databases
are the foundation of
western civilization”*

Bruce Lindsay, IBM Research

ACM SIGMOD Edgar F. Codd Innovations award 2012

Big
Tech Era

2019



+ growing need for tailored systems

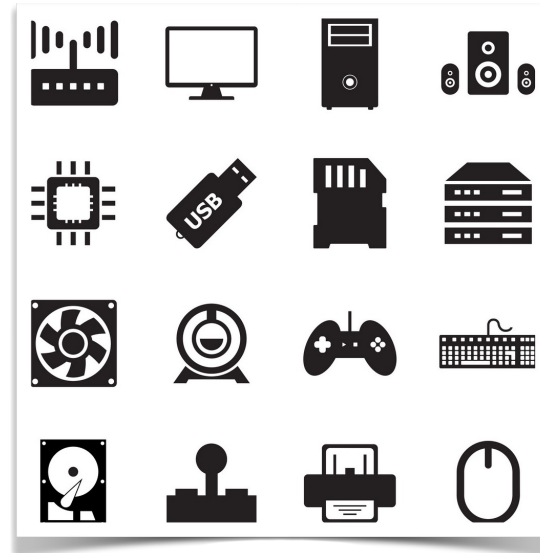
Why?



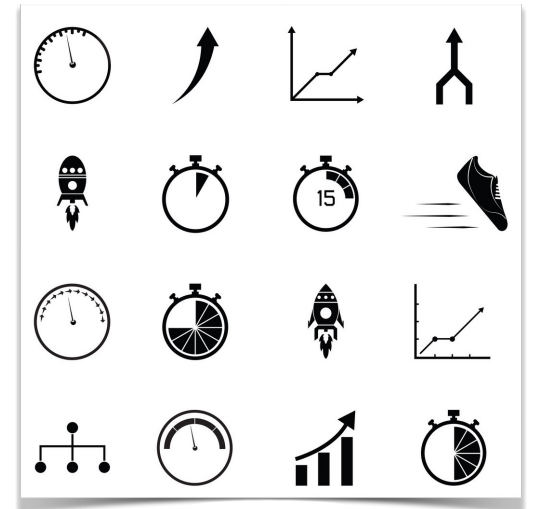
more data



new hardware



new applications



new performance
goals

ORACLE®

facebook.



SAP®



IBM

Google

The big success of 5 decades of research

a declarative interface!

“ask and thou shall receive”

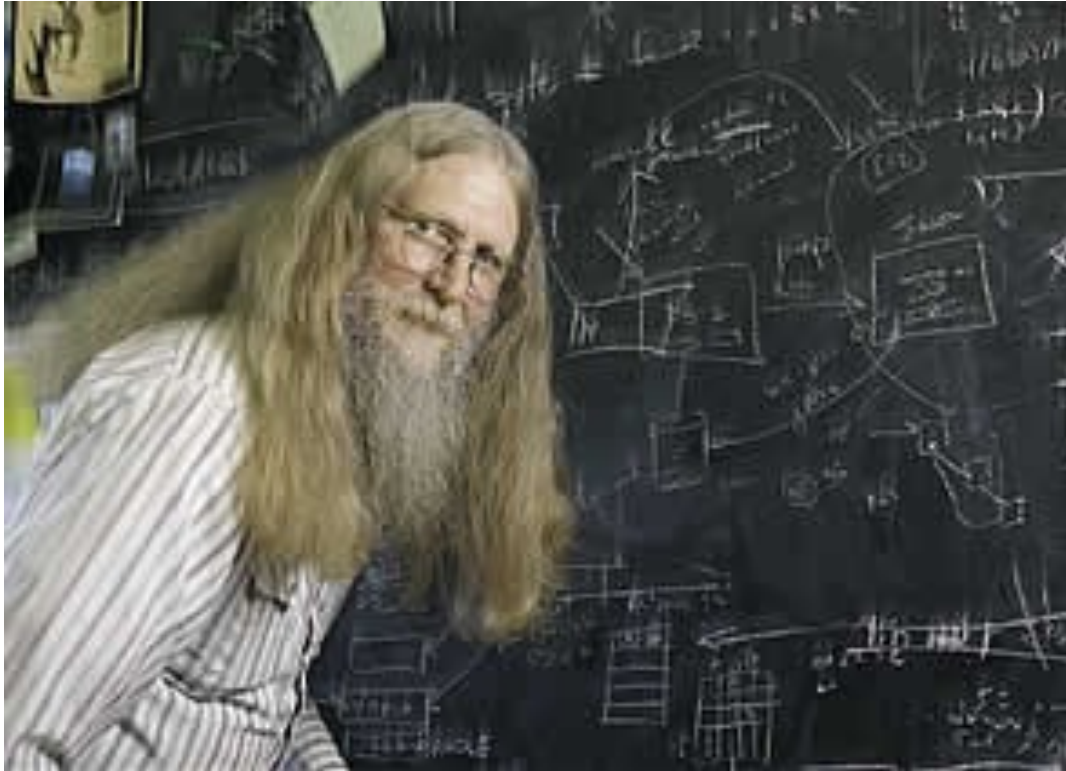
ask *what* you want

data system

system decides *how*
to store & access



is this good?

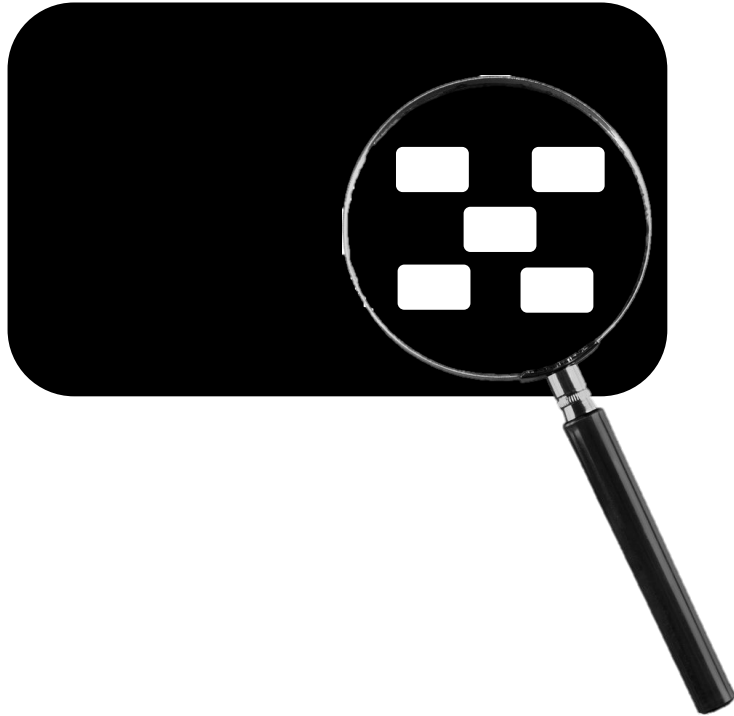


*“three things are important
in the database world:
**performance, performance,
and performance”***

Bruce Lindsay, IBM Research

ACM SIGMOD Edgar F. Codd Innovations award 2012

CS561: data systems **kernel** under the looking glass



this is where we will spend our time!

system architecture (row/column/hybrid)

indexing

relational/graph/key-value

scale-up/scale-out

goal: learn to design and implement a DB kernel

how to design a data system kernel?

what are its basic components?

algorithms/data structures/caching policies

what decisions should we make?

how to combine? how to optimize for hardware?

*designing a DB kernel is **complex***

data system design complexity



application



performance



budget

thousands of options
millions of decisions
billions of combinations

let's think together: a simple DB kernel

a key-value system, each entry is a {key,value} pair

main operations: *put, get, scan, range scan, count*

workload has both reads (*get, scan, range scan*) *and writes (put)*

data

how to store and how to access data?

how to efficiently delete?



designing a simple key-value system

what is the key/value?

are they stored together?

can read/write ratio change over time?

what to use? b-tree, hash-table, scans, skip-lists, zonemaps?

how to handle concurrent queries? million concurrent queries?

what happens if data does not fit in memory?

how to compress data?

what about privacy and security?

how to offer robustness guarantees?

what happens when we move to the cloud?

hardware at massive scale

performance tradeoffs different

10GB app: 1% less memory in your machine

10GB app: 1% less memory in 1M instances

$1M * 10GB * 1\% = 100TB!$

~800k\$ in today's price

so what?



class key goal

understand **system design tradeoffs**

design and **prototype** a system

with other **side-effects**:

sharpening your systems skills

(C/C++, profiling, debugging, linux tools)

data system designer & researcher

any business, any startup, any scientific domain

CS 561: more logistics

topics

storage layouts, solid-state storage, multi-cores, indexing, access path selection, HTAP systems, data skipping, adaptive indexing, time-series, scientific data management, map/reduce, data systems and ML, learned indexes

past but still relevant topics

relational systems, row-stores, query optimization, concurrency control, SQL

no textbook – only research papers

grading



class participation: 5%

project 0: 10%

project 1: 15%

reviews: 5%

technical questions: 10%

paper presentation: 15%

project proposal: 5%

mid-semester project report: 5%

project: 30%

Piazza

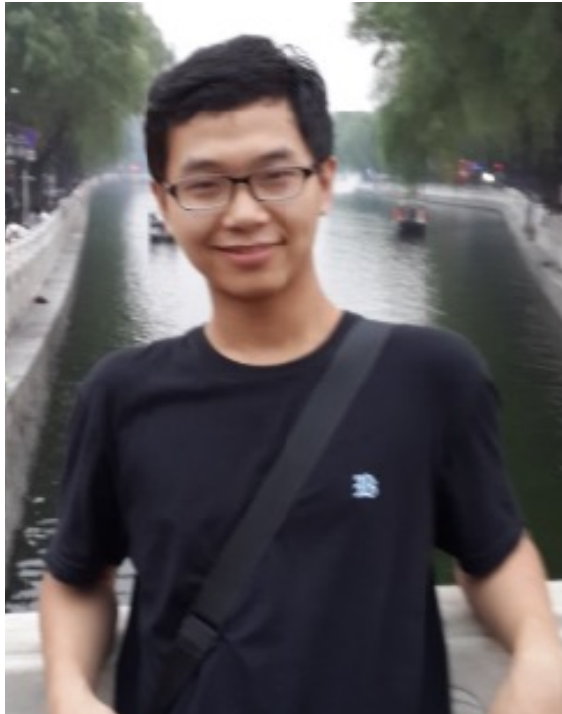


all discussions & announcements

<https://piazza.com/bu/spring2023/cs561>

also available on class website

Your awesome TA!



Zichen Zhu

How can I prepare?

1) Read background research material

- **Architecture of a Database System.** By J. Hellerstein, M. Stonebraker and J. Hamilton. Foundations and Trends in Databases, 2007
- **The Design and Implementation of Modern Column-store Database Systems.** By D. Abadi, P. Boncz, S. Harizopoulos, S. Idreos, S. Madden. Foundations and Trends in Databases, 2013
- **Massively Parallel Databases and MapReduce Systems.** By Shivnath Babu and Herodotos Herodotou. Foundations and Trends in Databases, 2013

2) Start going over the papers

class summary

2 classes per week / OH + Labs 5 days per week

each student

1 presentation/discussion lead + 1 review/question per week

project 0 + project 1 + systems or research project

proposal + mid-semester report + final report/presentation

what to do now?

- A) read the syllabus and the website**
- B) register to Piazza + Gradescope**
- C) start working on project 0**
- D) register for the presentation (week 2)
- E) start submitting paper reviews/answering tech. questions (week 3)
- F) go over the project (end of next week will be available)
- G) start working on the proposal (week 3)

survival guide

class website: <https://bu-disc.github.io/CS561/>

piazza website: <https://piazza.com/bu/spring2023/cs561>

presentation registration: <https://tinyurl.com/S23-CS561-presentations>

gradescope: <https://www.gradescope.com/courses/470999> (**code in Piazza**)

office hours: Subhadeep (Tu/Th 2-3pm)
Zichen (Mo/We 2-3pm)

material: papers available from the BU network

Welcome to CS 561: Data Systems Architectures!

Dr. Subhadeep Sarkar

ssarkar1@bu.edu

next time: more detailed logistics and start with data systems design