BOSTON UNIVERSITY

CS 591: Data Systems Architectures

class 2

Data Systems 101

Prof. Manos Athanassoulis

https://midas.bu.edu/classes/CS591A1/

some reminders

no smartphones

no laptop

class summary

2 classes per week / OH 5 days per week

each student

1 presentation/discussion lead + 2 reviews/questions per week

systems or research project + proposal + mid-semester report

BOSTON UNIVERSITY

systems project

implementation-heavy C/C++ project

groups of 2

research project

groups of 3

pick a subject (list will be available)

design & analysis

experimentation

class timeline

now

Week 2 – register
for presentations by 1/30
first presentation on 2/4

Week 3 – form
groups by 2/7

Week 4 – find project
by 2/14

Week 5 – submit project
proposal on 2/21

Week 9 – submit mid-
semester project
report on 3/21

Week 15 – Project presentations
submit all material by 4/26

discussions
interaction in OH
questions

discussions
interaction in OH
questions

discussions
interaction in OH
questions

Piazza

all discussions & announcements

http://piazza.com/bu/spring2020/cs591a1/

also available on class website

10 already registered!

register so we can reach you easily

BOSTON UNIVERSITY

size (volume)

rate (velocity)

sources (variety)

big data

(it’s not only about size)

The 3 V’s

+ our ability to collect ***machine-generated*** data

scientific experiments

sensors

social

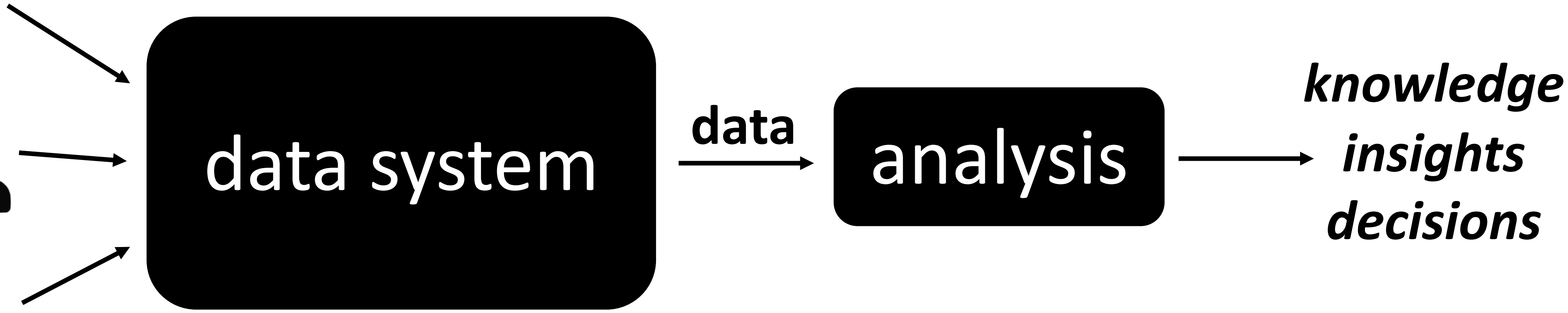
Internet-of-Things

BOSTON UNIVERSITY

a **data system** is a large software system

that **stores data**, and provides the **interface** to

update and **access** them **efficiently**

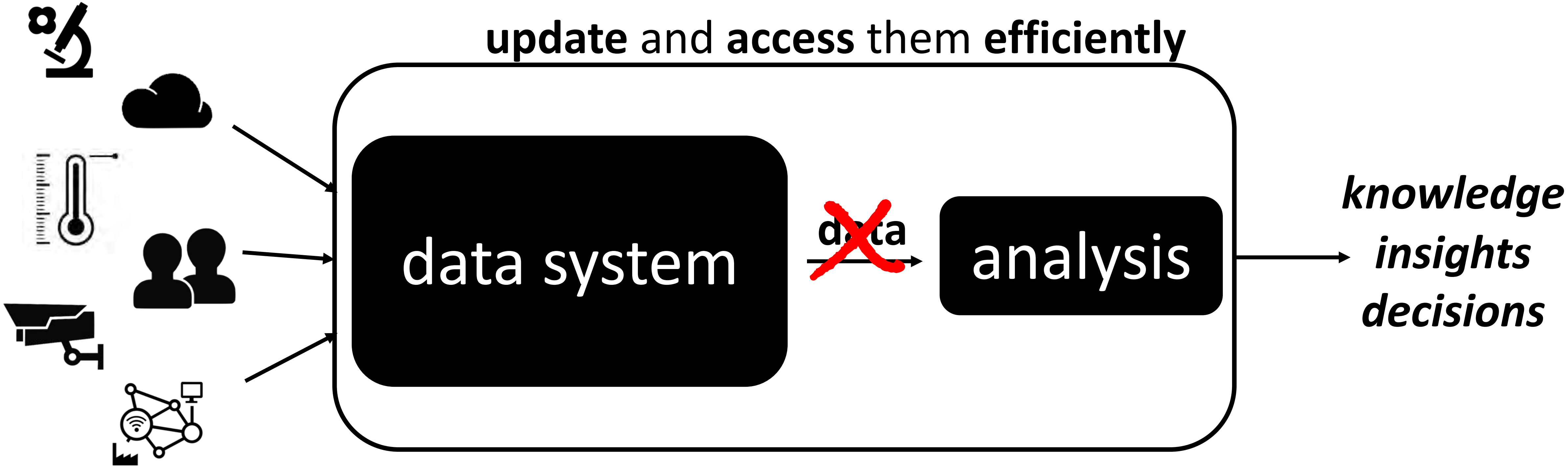


BOSTON UNIVERSITY

a **data system** is a large software system

that **stores data**, and provides the **interface** to

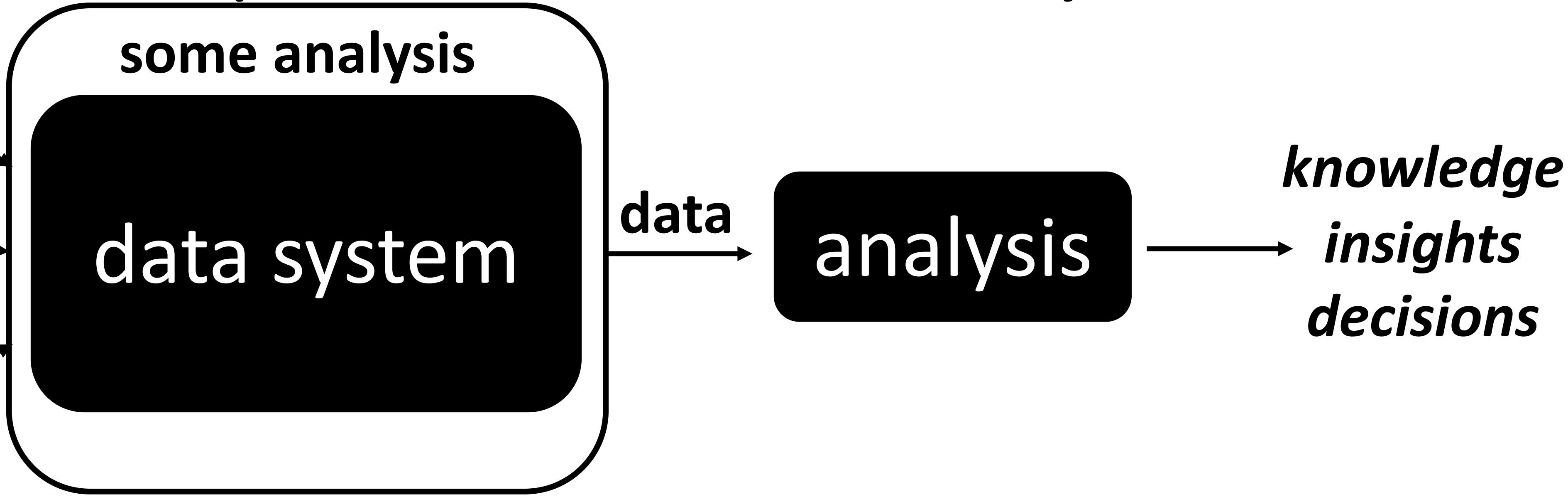
update and **access** them **efficiently**



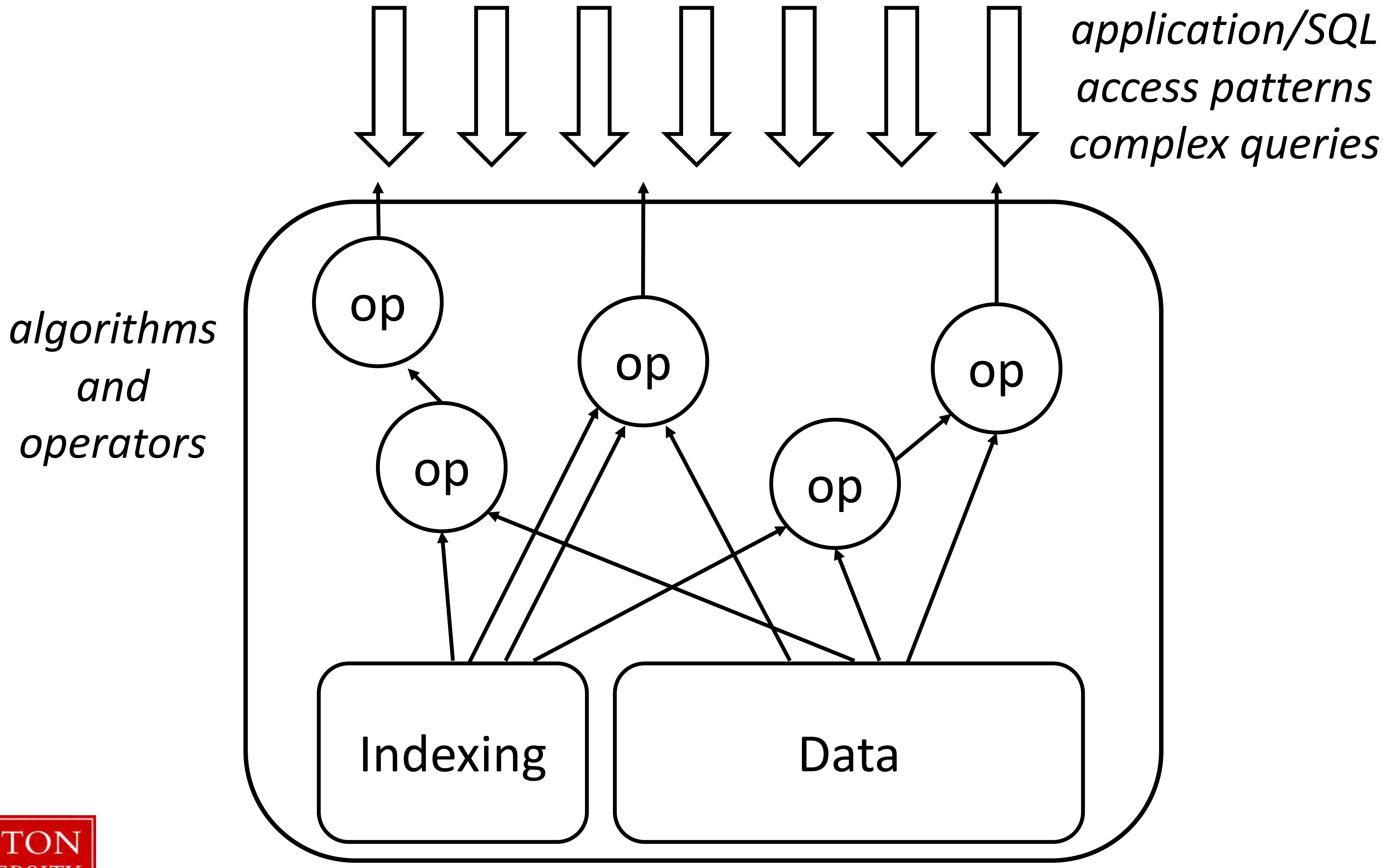
a **data system** is a large software system

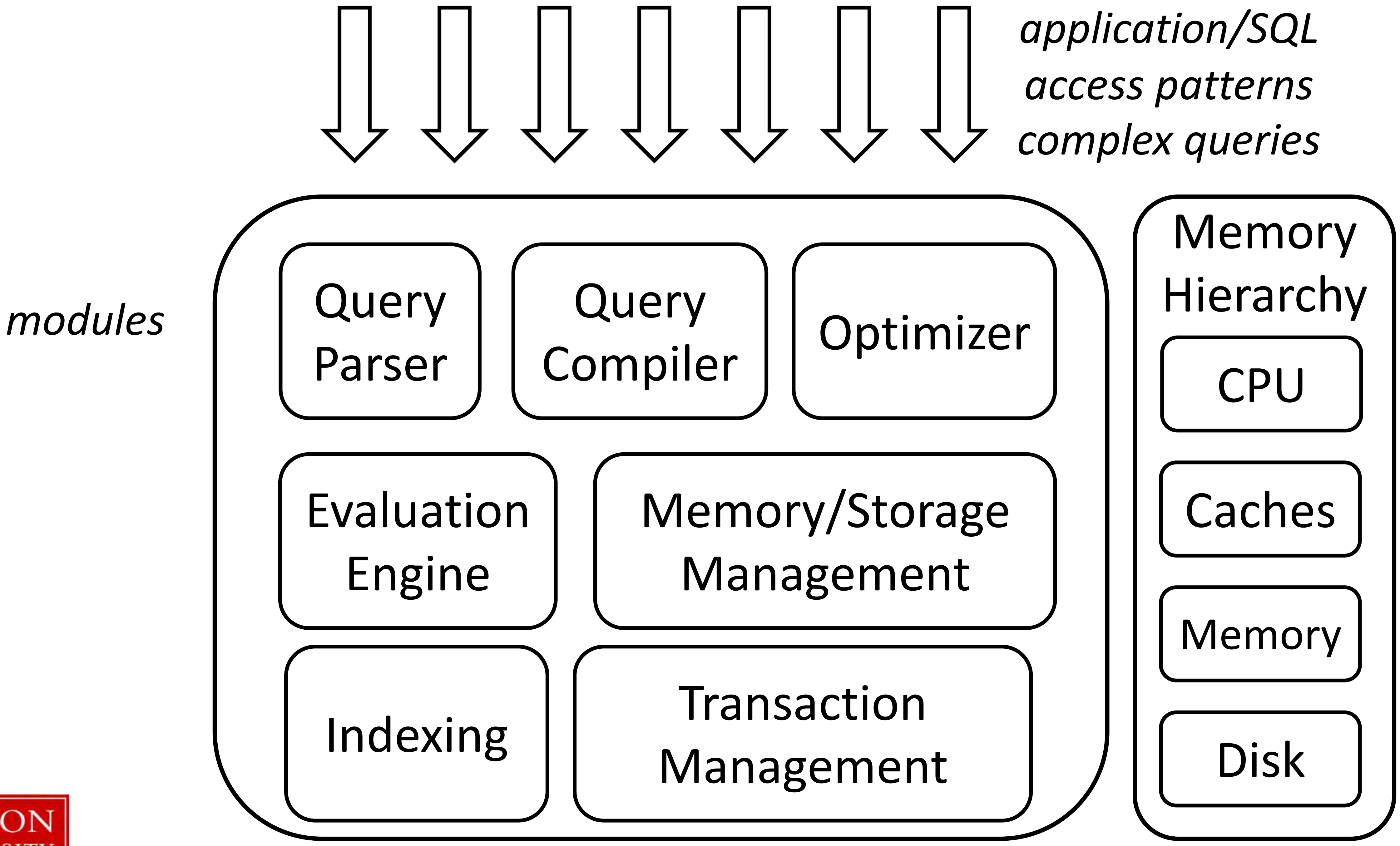
that **stores data**, and provides the **interface** to

update and **access** them **efficiently**



data system, what’s inside?


application/SQL
access patterns
complex queries
algorithms
and
operators
op
op
op
op
op
Indexing
Data



BOSTON UNIVERSITY

growing environment

db

large systems
complex
lots of tuning
legacy

noSQL

simple, clean
"just enough"

more **complex** applications

need for **scalability**

newSQL

>$200B by 2020, growing at 11.7% every year

[The Forbes, 2016]

[noSQL]

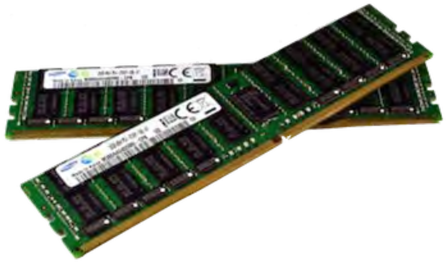
$3B by 2020, growing at 20% every year

[Forrester, 2016]

growing need for tailored systems

new applications

new hardware

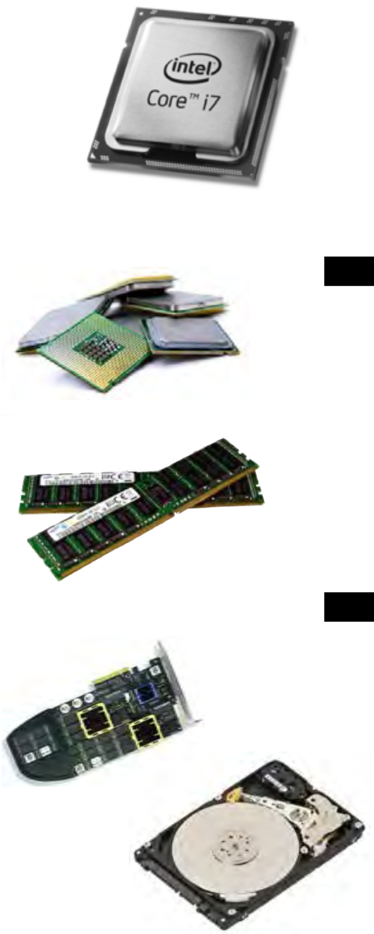


more data

BOSTON UNIVERSITY

data system, what’s underneath?

memory hierarchy



CPU

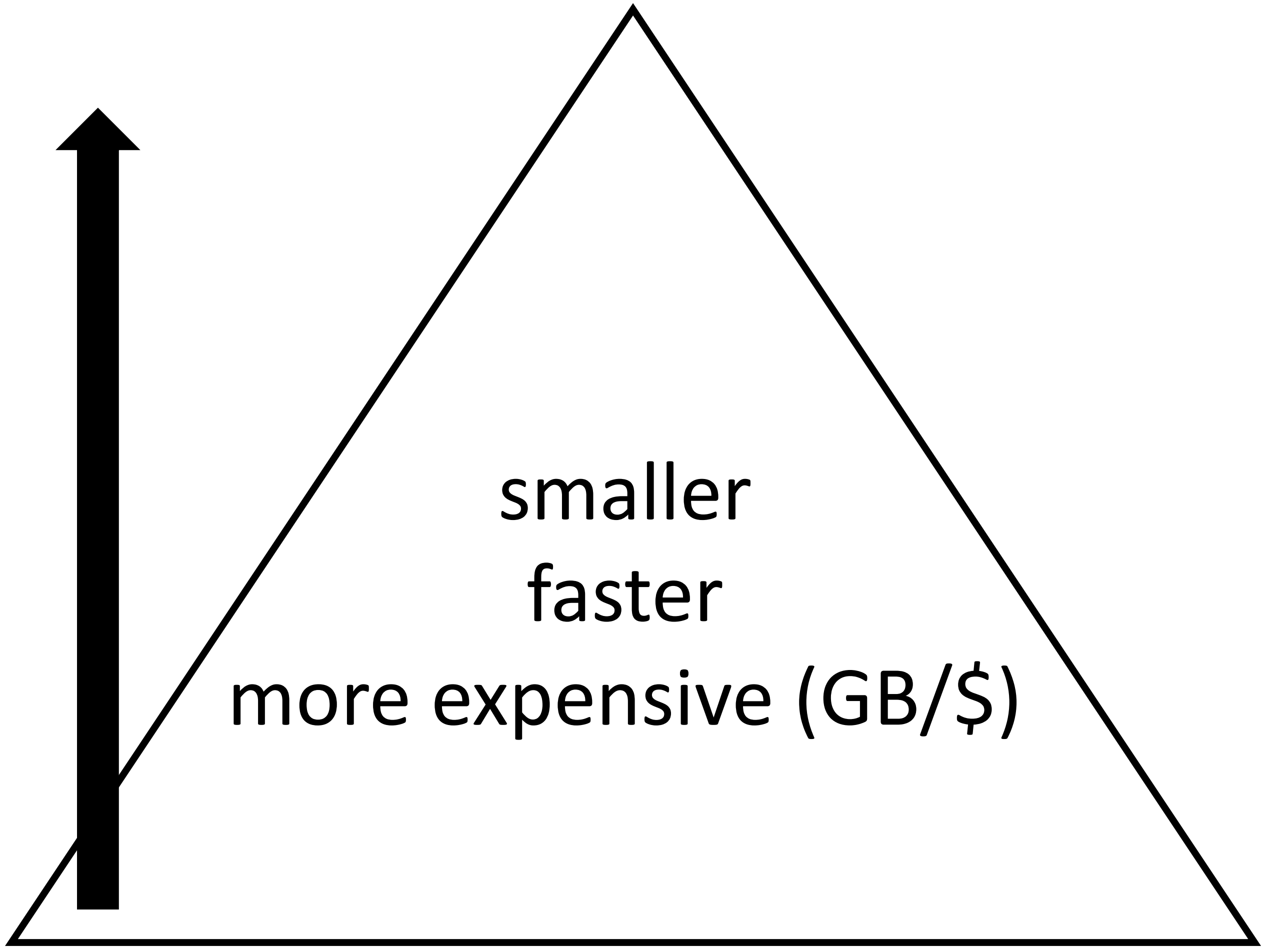
on-chip cache

on-board cache

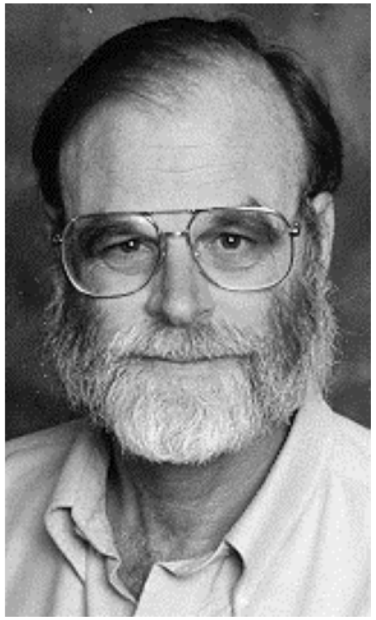
main memory

flash storage

disks | flash



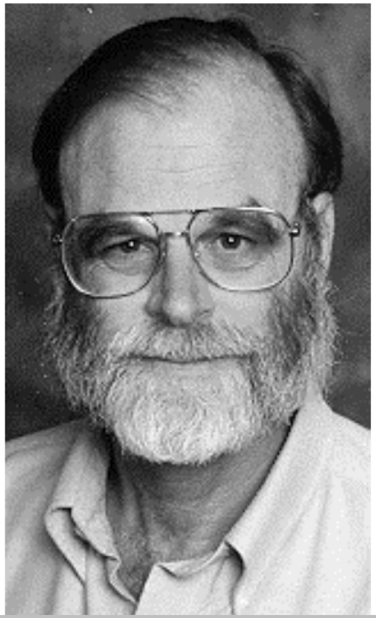
memory hierarchy (by Jim Gray)



Jim Gray, IBM, Tandem, Microsoft, DEC
“The Fourth Paradigm” is based on his vision
ACM Turing Award 1998
ACM SIGMOD Edgar F. Codd Innovations award 1993

	registers/CPU	my head ~0
2x	**on chip cache**	this room 1min
10x	**on board cache**	this building 10min
100x	**memory**	Washington, DC 5 hours
10^6x	**disk**	Pluto 2 years
10^9x	**tape**	Andromeda 2000 years

memory hierarchy (by Jim Gray)



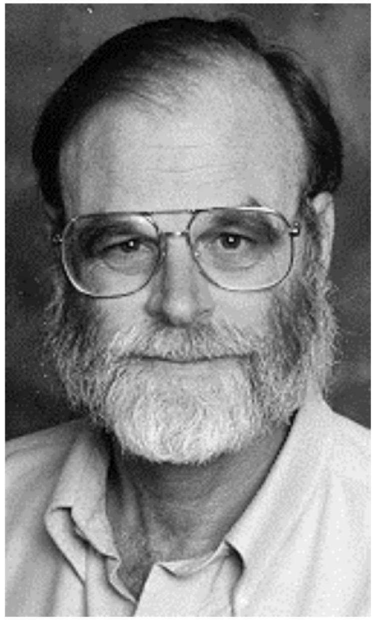
	registers/CPU	my head ~0
2x	**on chip cache**	this room 1min
10x	**on board cache**	this building 10min

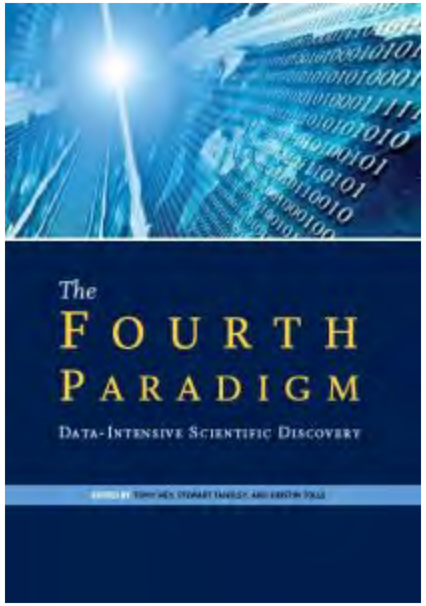
tape?

sequential-only magnetic storage

still a multi-billion industry

Jim Gray (a great scientist and engineer)



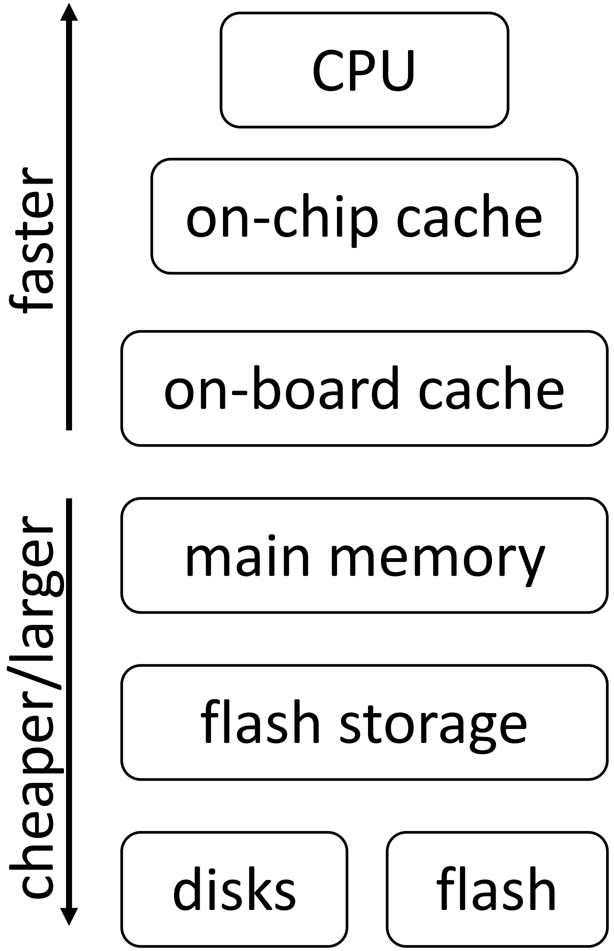


the first collection of
technical visionary research on
a data-intensive scientific discovery

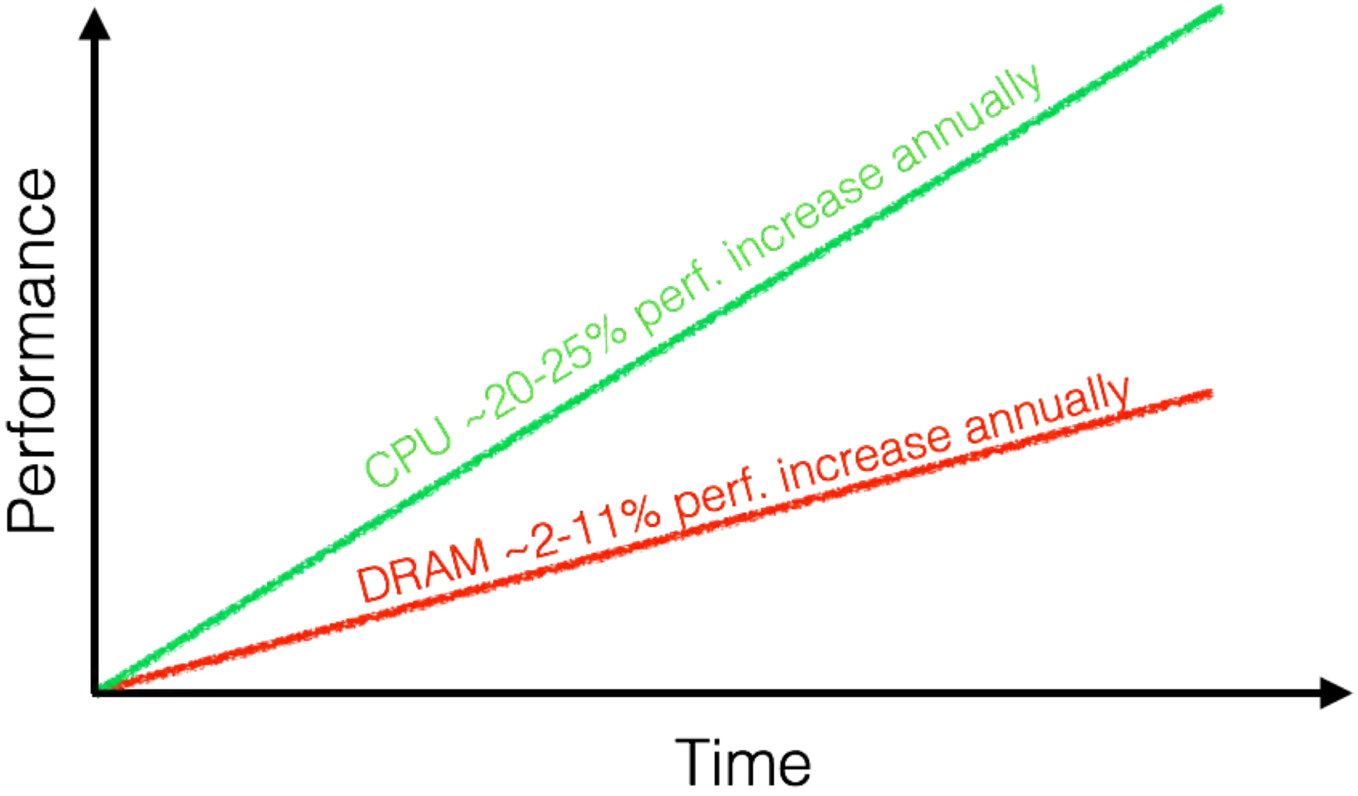
Jim Gray, IBM, Tandem, Microsoft, DEC
“The Fourth Paradigm” is based on his vision
ACM Turing Award 1998
ACM SIGMOD Edgar F. Codd Innovations award 1993

BOSTON UNIVERSITY

memory wall

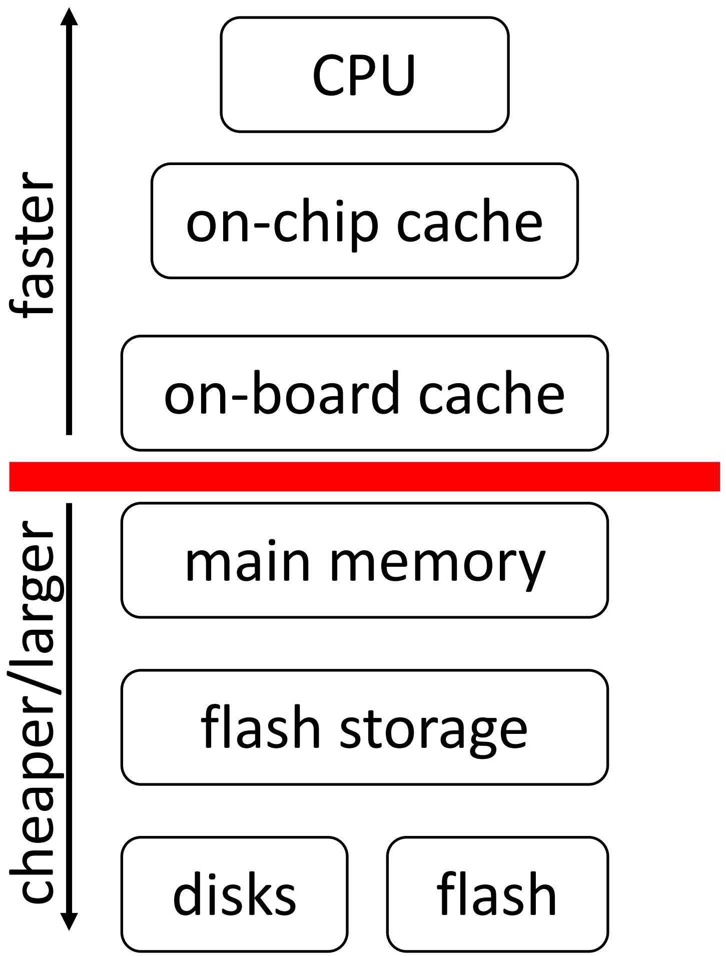


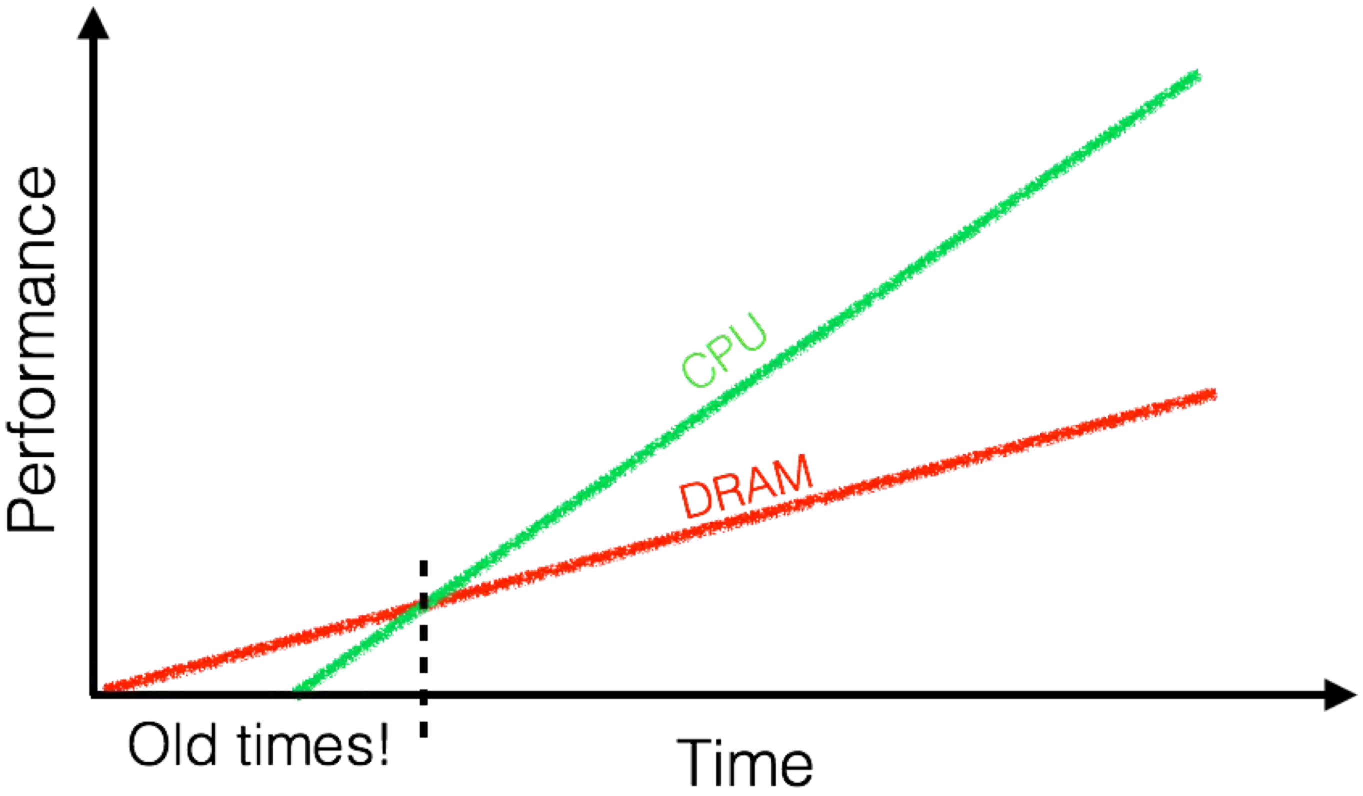
faster
cheaper/larger
CPU
on-chip cache
on-board cache
main memory
flash storage
disks
flash



Performance
Time
CPU ~20-25% perf. increase annually
DRAM ~2-11% perf. increase annually

memory wall


faster
cheaper/larger
CPU
on-chip cache
on-board cache
main memory
flash storage
disks
flash


Performance
CPU
DRAM
Old times!
Time

cache/memory misses

CPU

on-chip cache

on-board cache

main memory

flash storage

disks | flash

cache miss: looking for something that is not in the cache

memory miss: looking for something that is not in memory

what happens if I miss?

BOSTON UNIVERSITY

data movement

CPU

on-chip cache

on-board cache

main memory

flash storage

disks | flash

Photo by Gary Dineen/NBAE via Getty Images

data go through
all necessary levels

need to read only X
read the whole page

also read
unnecessary data

BOSTON UNIVERSITY

data movement

CPU

on-chip cache

on-board cache

main memory

flash storage

data go through
all necessary levels

also read
unnecessary data

Photo by Gary Dineen/NBAE via Getty Images

need to read only X
read the whole page

X page

remember!
disk is millions (mem, hundreds) times slower than CPU

page-based access & random access

query x<7

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23	2, 7, 13, 9, 8	10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

$ 40 bytes

page-based access & random access

query x<7

scan

output

1, 5, 12, 24, 23

1, 5

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

BOSTON UNIVERSITY

page-based access & random access

$ 40 bytes

query x<7

scan → output

1, 5, 12, 24, 23	2, 7, 13, 9, 8	1, 5

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23	2, 7, 13, 9, 8	10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

BOSTON UNIVERSITY

page-based access & random access

$ 40 bytes

query x<7

scan →

output

1, 5, 12, 24, 23	2, 7, 13, 9, 8	1, 5, 2

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23	2, 7, 13, 9, 8	10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

page-based access & random access

$ 80 bytes

query x<7

scan →

output

1, 5, 12, 24, 23

2, 7, 13, 9, 8

1, 5, 2

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

page-based access & random access

$ 80 bytes

query x<7

scan →

output

10, 11, 6, 14, 15

2, 7, 13, 9, 8

1, 5, 2

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

BOSTON UNIVERSITY

page-based access & random access

$ 80 bytes

query x<7

scan

output

10, 11, 6, 14, 15

2, 7, 13, 9, 8

1, 5, 2, 6

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

BOSTON UNIVERSITY

$ 120 bytes

page-based access & random access

query x<7

scan →

output

10, 11, 6, 14, 15

2, 7, 13, 9, 8

1, 5, 2, 6

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

BOSTON UNIVERSITY

what if we had an oracle (perfect index)?

page-based access & random access

query x<7

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

page-based access & random access

$ 40 bytes

query x<7

oracle

1, 5, 12, 24, 23

output

1, 5

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

page-based access & random access

40 bytes

query x<7

oracle

output

1, 5, 12, 24, 23 | 2, 7, 13, 9, 8 | 1, 5

size=120 bytes

memory (memory level N)

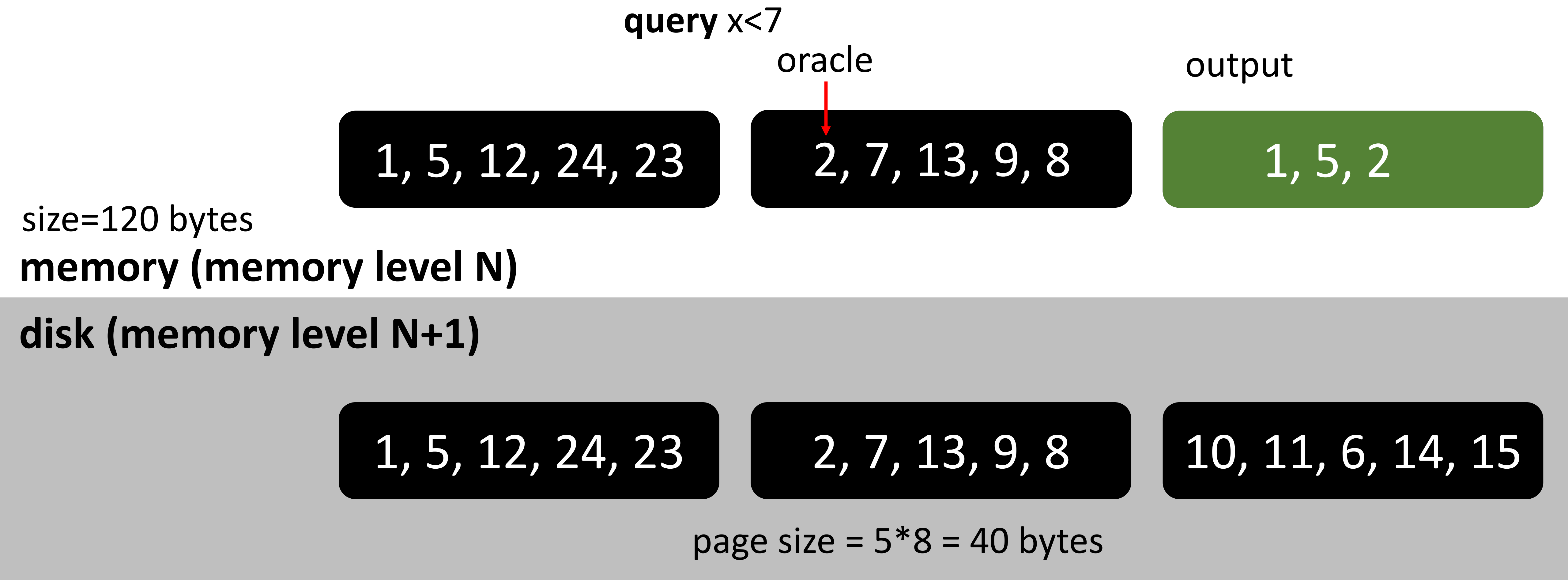
disk (memory level N+1)

1, 5, 12, 24, 23 | 2, 7, 13, 9, 8 | 10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

page-based access & random access

$ 40 bytes



page-based access & random access

$ 80 bytes

query x<7

oracle

output

1, 5, 12, 24, 23

2, 7, 13, 9, 8

1, 5, 2

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

BOSTON UNIVERSITY

page-based access & random access

$ 80 bytes

query x<7

oracle

output

10, 11, 6, 14, 15

2, 7, 13, 9, 8

1, 5, 2

size=120 bytes

memory (memory level N)

disk (memory level N+1)

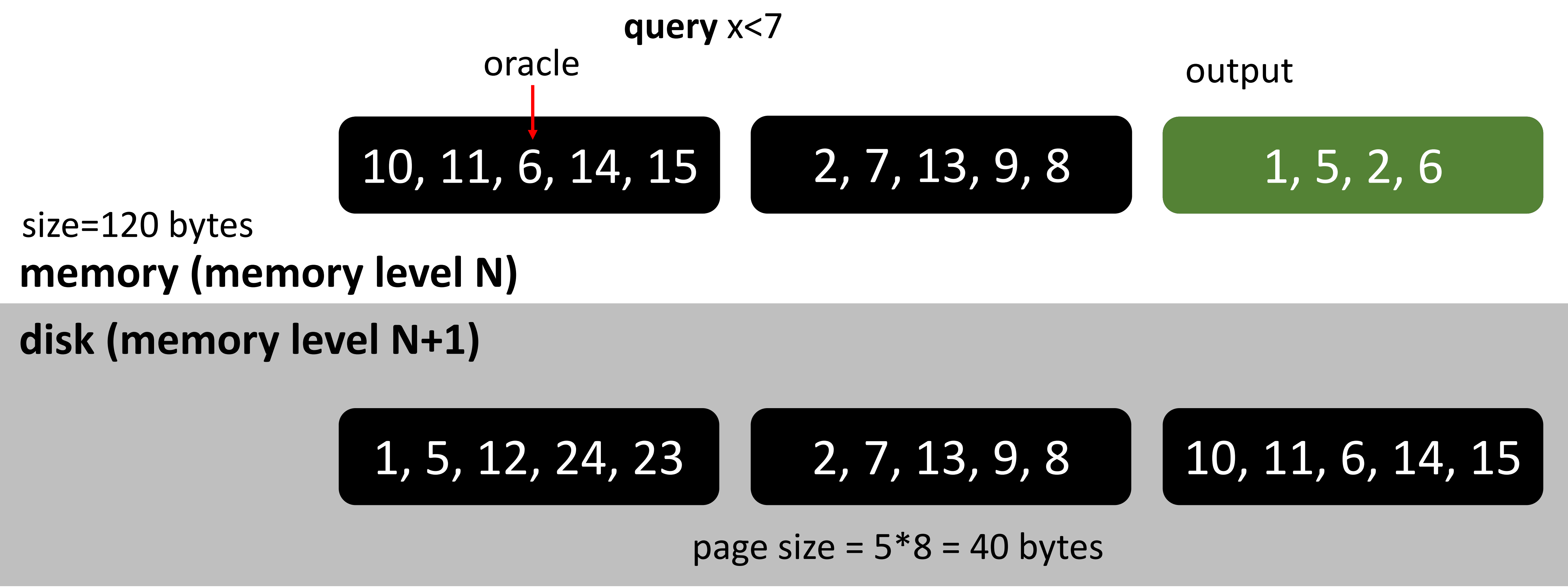
1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

page-based access & random access



BOSTON UNIVERSITY

$ 120 bytes

page-based access & random access

query x<7

was the oracle helpful?

oracle

output

10, 11, 6, 14, 15	2, 7, 13, 9, 8	1, 5, 2, 6

size=120 bytes

memory (memory level N)

disk (memory level N+1)

1, 5, 12, 24, 23	2, 7, 13, 9, 8	10, 11, 6, 14, 15

page size = 5*8 = 40 bytes

BOSTON UNIVERSITY

when is the oracle helpful?

for which query would an oracle help us?

how to decide whether to use the oracle?

1, 5, 12, 24, 23

2, 7, 13, 9, 8

10, 11, 6, 14, 15

how we store data

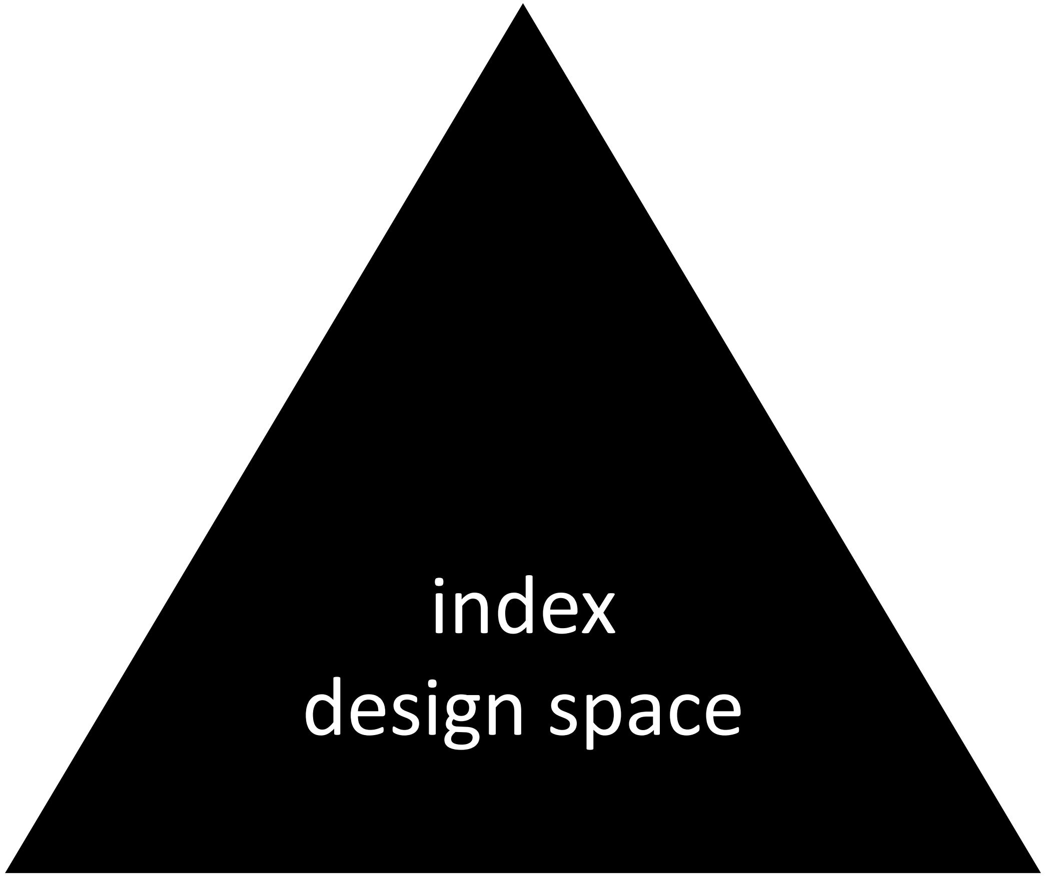
layouts, indexes

every **byte** counts

overheads and tradeoffs

know the **query**

access path selection



BOSTON UNIVERSITY

rules of thumb

sequential access

read one block; consume it completely; discard it; read next;

hardware can predict and start prefetching

prefetching can exploit full memory/disk bandwidth

random access

read one block; consume it partially; discard it; (may re-use);

read random next;

ideal random access?

the one that helps us **avoid a large number of accesses** (random or sequential)

the language of efficient systems: C/C++

why?

low-level control over hardware

make decisions about physical data placement and consumptions

fewer assumptions

the language of efficient systems: C/C++

why?

low-level control over hardware

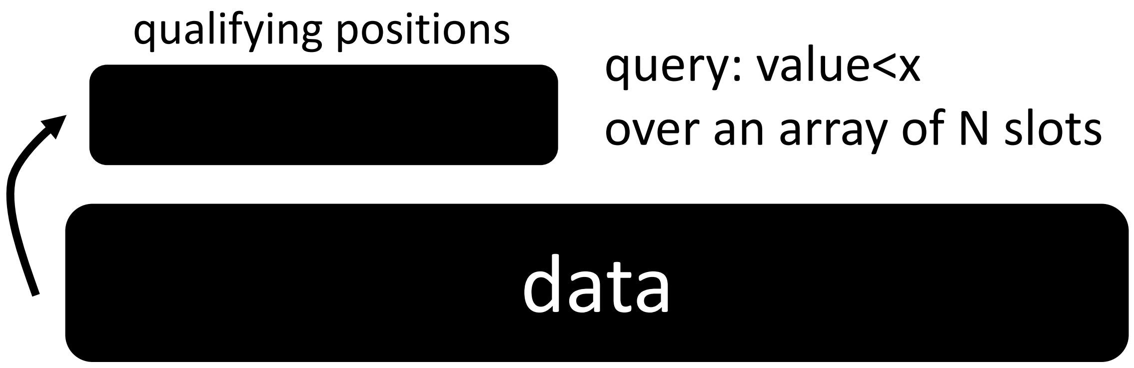
we want you in the project to make low-level decisions

a “simple” database operator

select operator (scan)

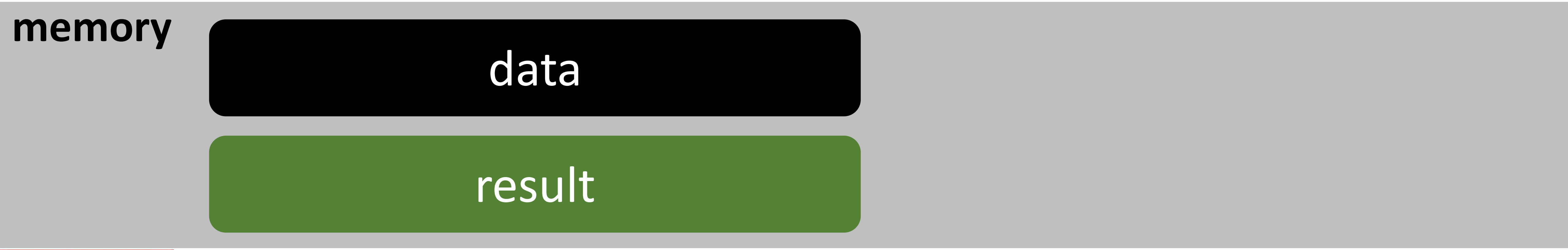
BOSTON UNIVERSITY

how to implement it?



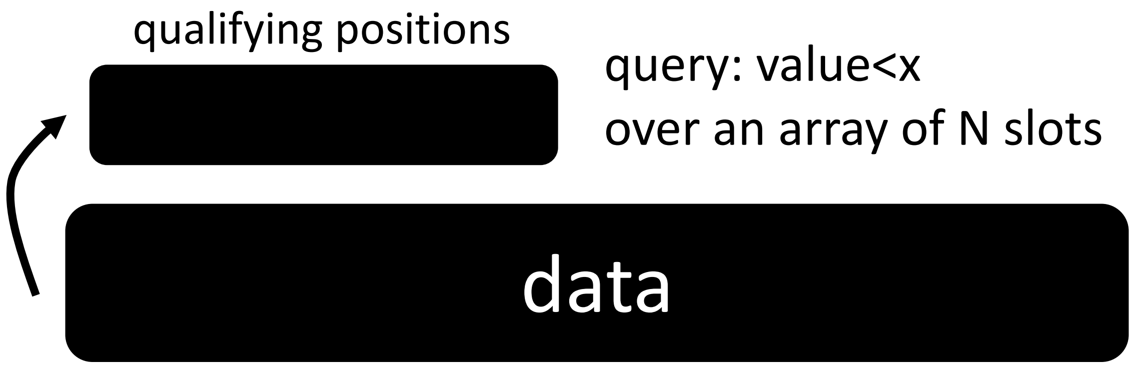

```
result = new array[data.size];
j=0;
for (i=0; i<data.size; i++)
    if (data[i]<x)
        result[j++]=i;
```

what if only 0.1% qualifies?



BOSTON UNIVERSITY

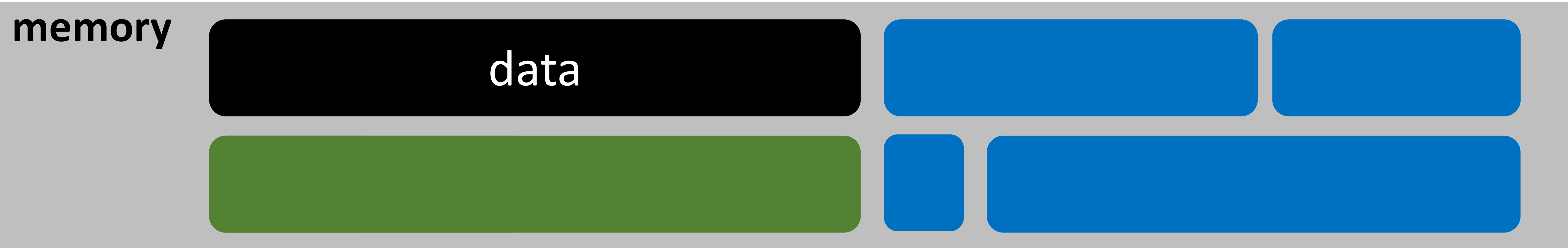
how to implement it?



query: value<x
over an array of N slots

```
result = new array[data.size];
j=0;
for (i=0; i<data.size; i++)
    if (data[i]<x)
        result[j++]=i;
```

what if only 0.1% qualifies?



BOSTON UNIVERSITY

qualifying positions

query: value<x
over an array of N slots

data

how to implement it?

```
result = new array[data.size];
j=0;
for (i=0; i<data.size; i++)
    if (data[i]<x)
        result[j++]=i;
```

what if 99% qualifies?

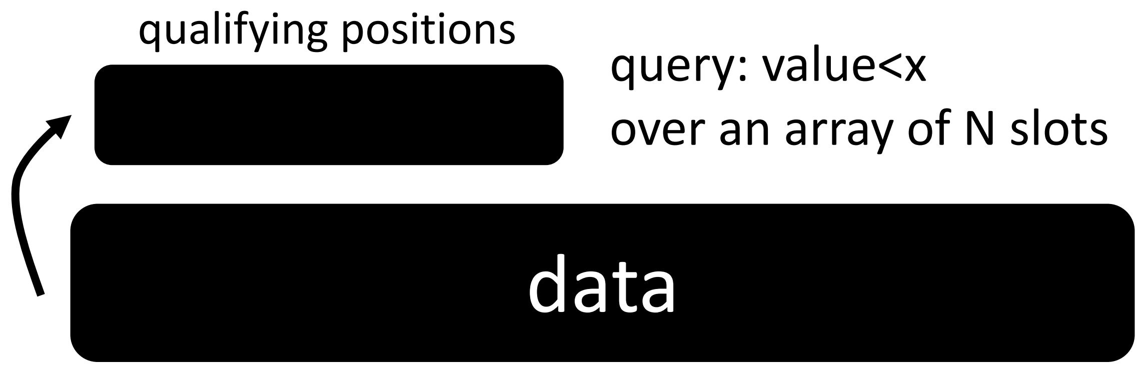
how can we know?

branches (if statements)
are bad for the processors,
can we avoid them?

```
result = new array[data.size];
j=0;
for (i=0; i<data.size; i++)
    result[j+=(data[i]<x)]=i;
```

how to bring the values?
(remember we have the positions)

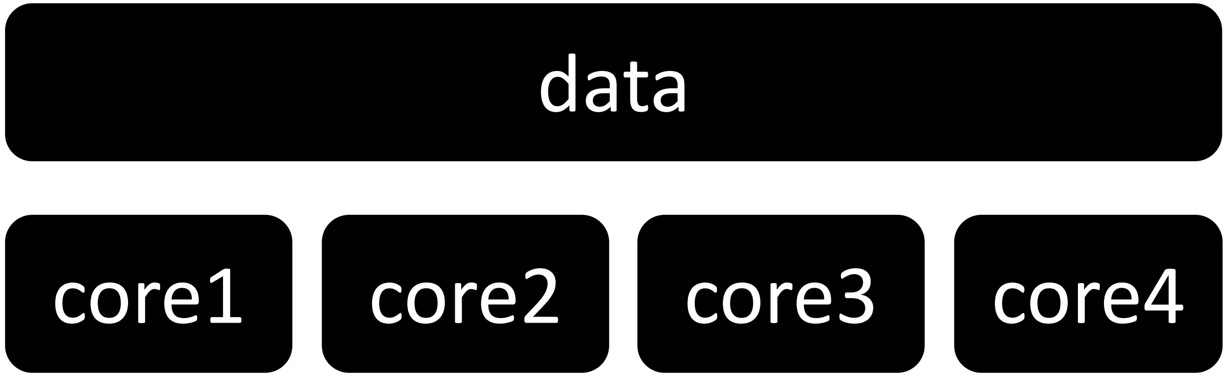
BOSTON UNIVERSITY



query: value<x
over an array of N slots

```
result = new array[data.size];
j=0;
for (i=0; i<data.size; i++)
    if (data[i]<x)
        result[j++]=i;
```

what about multi-core?
NUMA? SIMD? GPU?



needs coordination!
what about result writing?

BOSTON UNIVERSITY

what about having multiple queries?

query1: value<x1
query2: value<x2 …

```
result = new array[data.size];
j=0;
for (i=0; i<data.size; i++)
    if (data[i]<x)
        result[j++]=i;
```

BOSTON UNIVERSITY

query: value<x
over an array of N slots

data

should I scan?

should I probe an index?

how to decide which one is best?

total data movement

&

computation

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

BOSTON UNIVERSITY

what to do now?

A) read the syllabus and the website
B) register to piazza
C) register to gradescope
D) register for the presentation (**early next week!**)
E) start submitting paper reviews (week 3)
F) go over the project (next week will be available)
G) start working on the proposal (week 3)

BOSTON UNIVERSITY

survival guide

class website: https://midas.bu.edu/classes/CS591A1/

piazza website: http://piazza.com/bu/spring2020/cs591a1/

presentation registration: https://tinyurl.com/S2020-CS591-presentations

gradescope entry-code: 9568G3

office hours: Manos (Tu/Th, before class)
Andy (M/W 3-4pm), Ju Hyoung (M 11am-noon / F 3-4pm)

material: papers available from BU network

BOSTON UNIVERSITY

CS 591: Data Systems Architectures

class 2

Data Systems 101

next week: modern main-memory data systems & semester project