class 17

updates

prof. Stratos Idreos

HTTP://DASLAB.SEAS.HARVARD.EDU/CLASSES/CS165/
so far

early/late tuple reconstruction, tuple-at-a-time, vectorized or bulk processing, intermediates format, pushing selects down, etc.

scan, binary search, tuple reconstruction, min, max, search/update b-tree, join, etc.

arrays, columns, matrixes, rows, trees
**UPDATE** table_name
**SET** column1=value1,column2=value2,...
**WHERE** some_column=some_value

**INSERT INTO** table_name
**VALUES** (value1,value2,value3,...)
traditional applications
e.g., banking

how many times per day do you send
update queries to your bank account
the world has changed a little bit by now…

updates
still we spy Facebook more than the # of photos we upload or # of our twitter posts, etc…
so systems should be tuned for more reads…

yet we have way more writes than before

and a variable read/write ratio
which kind of update is more common
update, insert, delete
which kind of update is more common
update, insert, delete

so our new challenge is:
reads and inserts + variable read/write ratio
not just about user data: everything is data!
monitor CPU utilization

monitor memory hierarchy utilization

monitor clicks
(frequency, locations, specific links, sequences)

what & how
Today, data systems are nearly everywhere…

Continuous need for new and tailored data systems
data systems are nearly everywhere...

today

data grows
tomorrow

continuous need for new and tailored data systems
more applications

more data

more h/w
analyze data as it arrives and react (standing queries)
merge incoming data with already archived data
see the correct up-to-date values
do not lose any updates (software/hardware failures)
>>1 updates concurrently
conflicting goals
(hardware and requirements change continuously and rapidly)

moving target

application requirements

budget

performance

hardware

energy profile
“Three things are important in the database world: performance, performance, and performance”

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

true for both reads & writes
how to do fast (& correct) updates?
(more or less same way we do fast reads)

transactions
logging
locking

lazy vs eager updates
in-place or not

fractured mirrors

layout vs scheduling
student{name, age, address, telephone, GPA, …}
insert new entry \((a,b,c,d,\ldots)\) on table \(x\)
update \(N\) columns, \(K\) trees, statistics, \(\ldots\)
Table x

To index or not to index

What info do we need

How do we make decisions

When to do create indices

Secondary index on D
data (array)

value
data (array)

value

insertion
data (array)

value

delete
data (array)

value

delete
data (array)

value

update
data (array)

value

inserts, deletes, updates = deletes followed by inserts
Jim Gray, IBM, Tandem, DEC, Microsoft
ACM Turing award
ACM SIGMOD Edgar F. Codd Innovations award

100Kx disk
100x memory
10x on board cache
2x on chip cache
registers

Pluto
2 years
New York
1.5 hours
this building
10 min
this room
1 min
my head
~0
random access & page-based access

same for writes!

need to only read $x$... but have to read all of page 1
1. read input into stream buffer, hash and write to respective partition buffer
2. when input buffer is consumed, bring the next one
3. when a partition buffer is full, write to L2
update value x to y in page p of array z

Level 1

Level 2

page to update

update

cost

what if >1 updates
(no locking for now)
buffer $\gg 1$ updates to this page before pushing to L2

page to update

update
e.g., from disk to flash

ideal write granularity is different

what do you think changed in update algorithms?
content vs structure update

insert tuple(a1, b1, c1, …) → insert(A,a1), insert(B, b1), …

say there is a secondary index on A

1) append a1 anywhere to index (any node/buffer)
2) reorganize index to maintain structure

…
row-store

A B C D

column-store

A B C D

vs

update row7 = (A=a, B=b, C=c, D=d)
A case for fractured mirrors
Ravishankar Ramamurthy, David J. DeWitt, Qi Su
Very Large Databases Journal (VLDBJ), 2003
select(A,v1,v2)
update all rows
where $A=v_1$ & $B=v_2$
to $(a=a/2,b=b/4,c=c-3,d=d+2)$

**how to perform updates efficiently and correctly?**

*correctly=all or nothing*

**problems to worry about (?):**
what if user/applications aborts?
what if power goes down?
what if there is an earthquake in our city?
what if aliens come to earth?

(assume simplified memory hierarchy)
all data fit in L2, not all data fit in L1
L2 is non-volatile, L1 is volatile
update all rows where \( A = v_1 \) \& \( B = v_2 \) to \((a=a/2, b=b/4, c=c-3, d=d+2)\)
update all rows where $A=v_1$ & $B=v_2$
to $(a=a/2, b=b/4, c=c-3, d=d+2)$

list of rowIDs (positions)

search (scan/index) to find row to update
select+project actions
update all rows where $A=v_1$ & $B=v_2$

to $(a=a/2, b=b/4, c=c-3, d=d+2)$

list of rowIDs (positions)

search (scan/index) to find row to update

select+project actions

we know what to update but nothing happened yet
read page in L1
update
persist to L2

if problem (power/abort)
before we write all pages
we are left with an inconsistent state

WAL: keep persistent notes as we go so we can resume or undo
when is the log or an update persistent?

disk

persistent memory, e.g., disk?
when is the log or an update persistent?

- disk
- persistent memory, e.g., disk?

machine 1
machine 2
machine 3

replicate to multiple machines?
when is the log or an update persistent?

persistent memory, e.g., disk?

machine 1
machine 2
machine 3

replicate to multiple machines?

city 1
city 2
city 3

replicate to multiple machines
>1 clusters in >1 cities?
more details about all these

**next class:** transactions, ACID

what if >>1 update queries
at the same time

WAL & replication
Aurora: a new model and architecture for data stream management
Daniel J. Abadi, Donald Carney, Ugur Çetintemel, Mitch Cherniack, Christian Convey, Sangdon Lee, Michael Stonebraker, Nesime Tatbul, Stanley B. Zdonik
Very Large Databases Journal (VLDBJ), 2003

Enhanced stream processing in a DBMS kernel
Erietta Liarou, Stratos Idreos, Stefan Manegold, Martin Kersten
In Proc. of the International Conf. on Extending Database Technology (EDBT), 2013
Positional update handling in column stores
Sándor Héman, Marcin Zukowski, Niels J. Nes, Lefteris Sidirourgos, Peter A. Boncz
In Proc. of the ACM SIGMOD Inter. Conference on Management of Data, 2010

Updating a cracked database
Stratos Idreos, Martin Kersten, Stefan Manegold

(textbook: chapters 16, 17, 18)

(also for next class)
updates
DATA SYSTEMS
prof. Stratos Idreos