AsterixDB
more engine less trunk

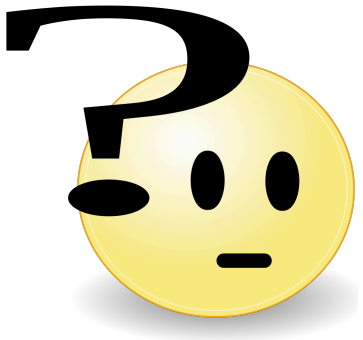
LSMs, DFSs, and Other Acronyms: Revisiting Storage and Layering for Big Data Management

Michael Carey
Information Systems Group
CS Department
UC Irvine

http://isg.ics.uci.edu

Rough Topical Plan

- Background and motivation (quick!)
- Big Data storage landscape (satellite view ☺)
  - Two points of view (plus *cloudy* skies)
- AsterixDB: our next-generation ***BDMS***
  - What it does (in a nutshell)
  - What we do about storage
- Storage research plans and Q&A

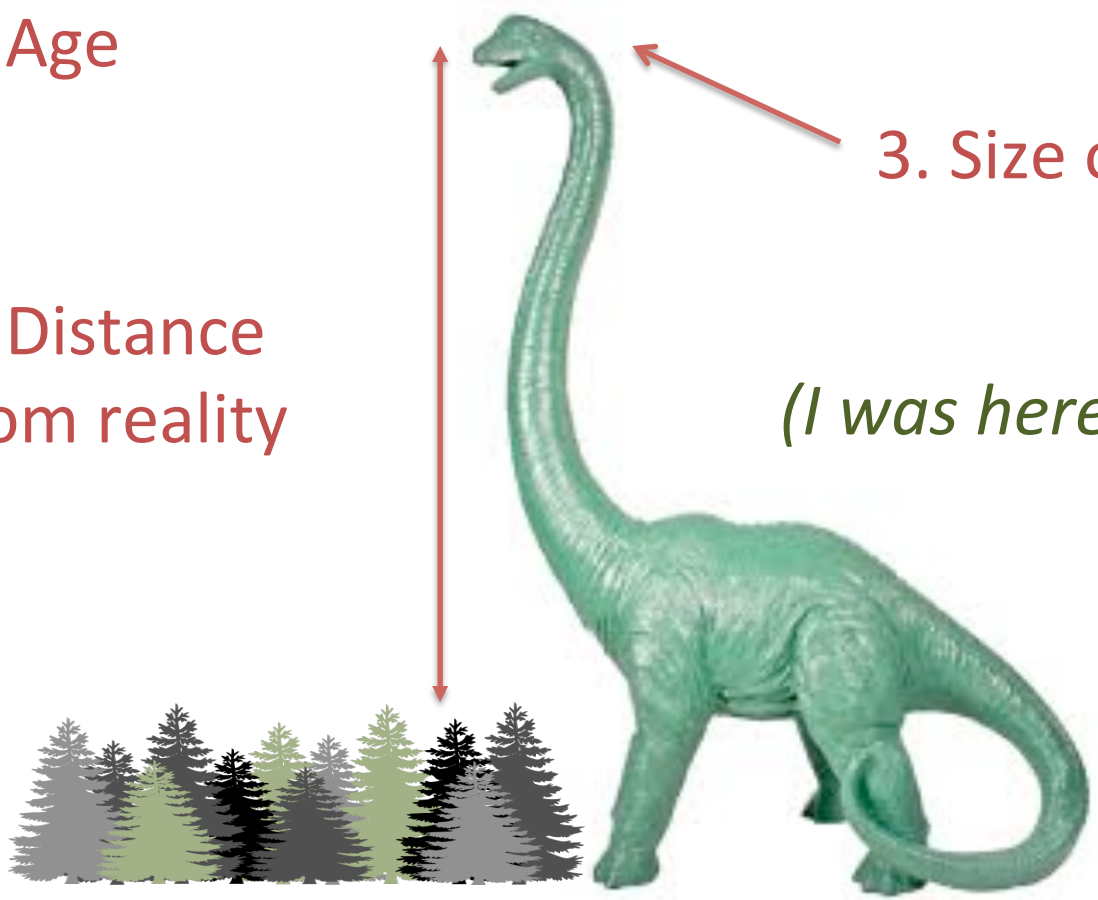


UCIrvine
http://isg.ics.uci.edu

A Few Presenter Cautions

1. Age

2. Distance from reality



3. Size of brain

(I was here in 1985...)

UCIrvine
http://isg.ics.uci.edu

HPTS Has Some Great Debates…

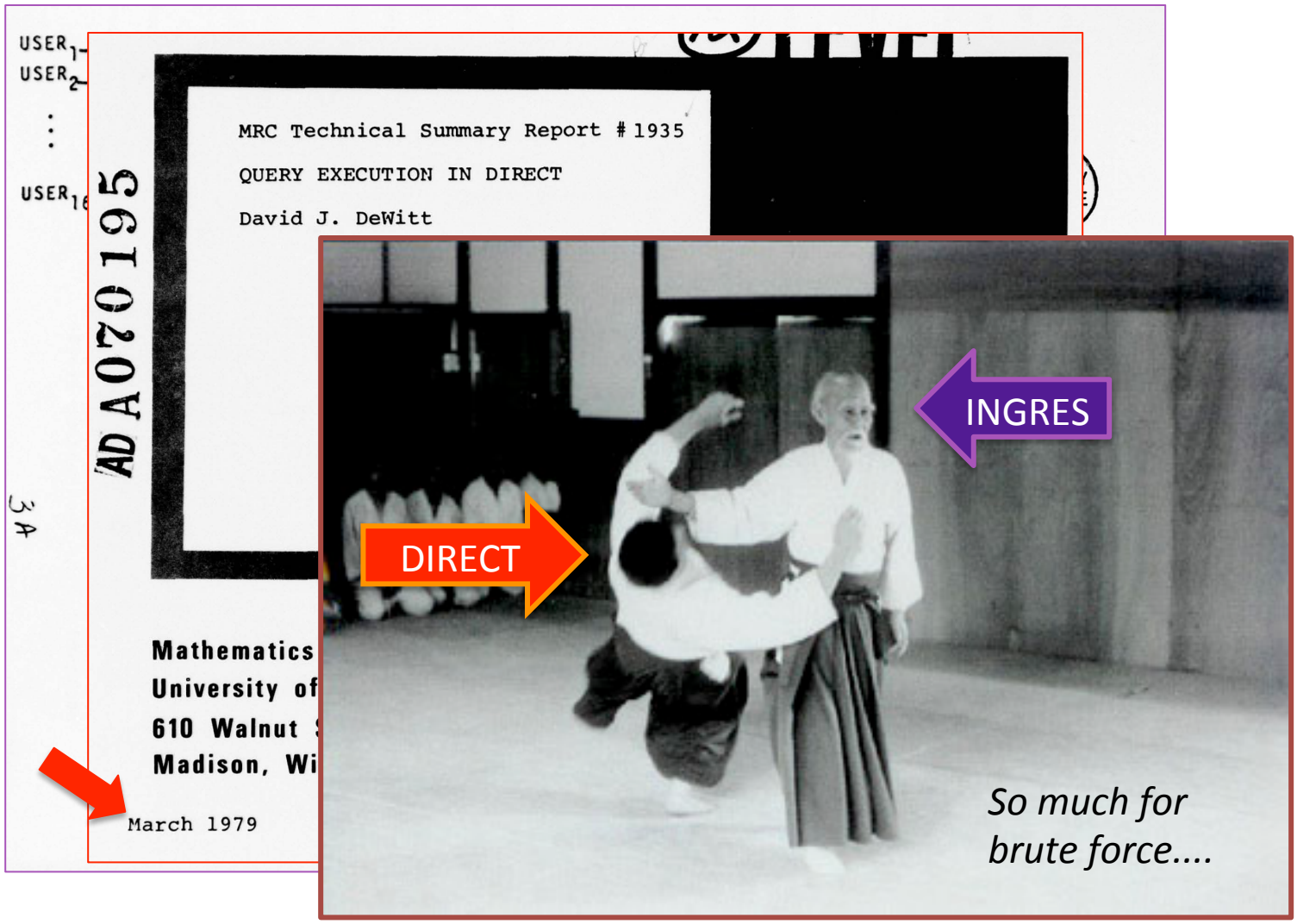
- *Debate #1:* The ***TP Architecture Wars*** in the early days of HPTS (late 1980's?)
  - *TP Heavy:* Transaction monitors (middleware)
  - *TP Lite:* Stored procedures (→ one less tier)

TP Heavy is the right way to go!

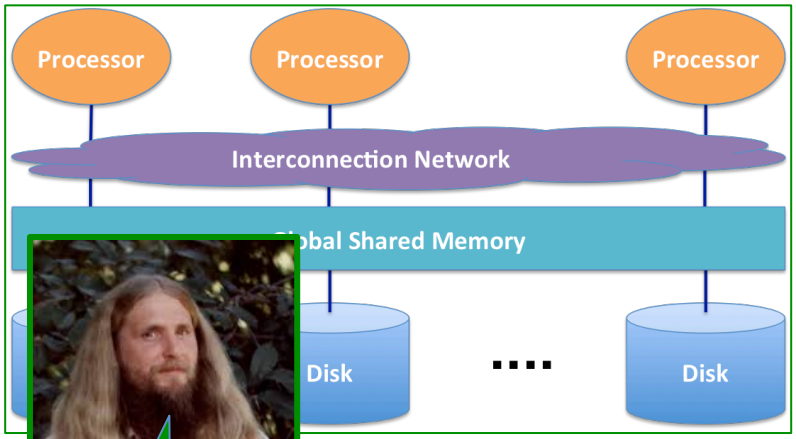
No way – TP Lite makes way more sense, you bozo!

UCIrvine
http://isg.ics.uci.edu

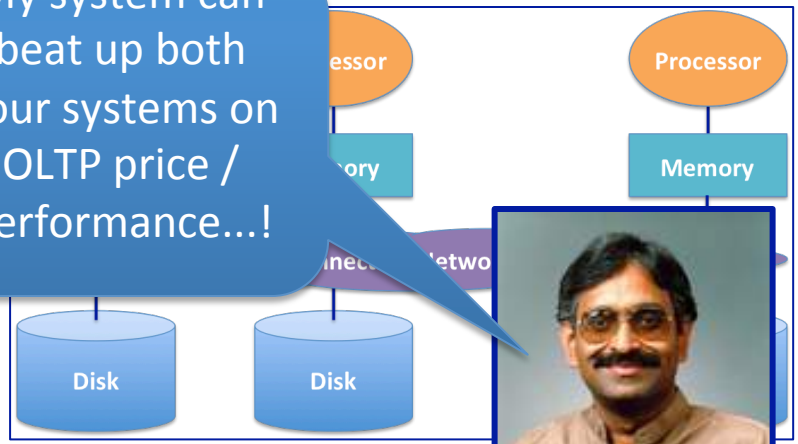
A DB History Lesson: ***DIRECT***



Debate #2: The ***Shared What? Wars***



Shared-everything



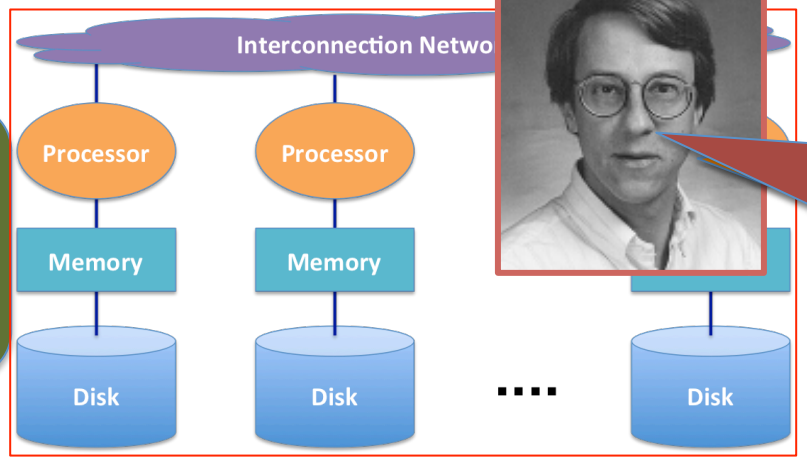
Shared-disk

My system can beat up both your systems on OLTP price / performance...!

My system can load balance OLTP *and* parallelize queries...!

last night...!

Shared-nothing

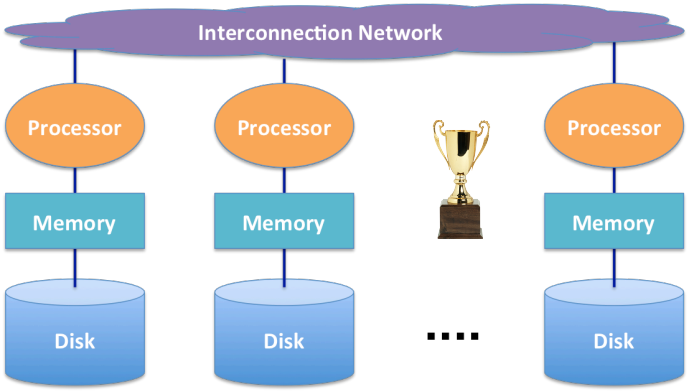


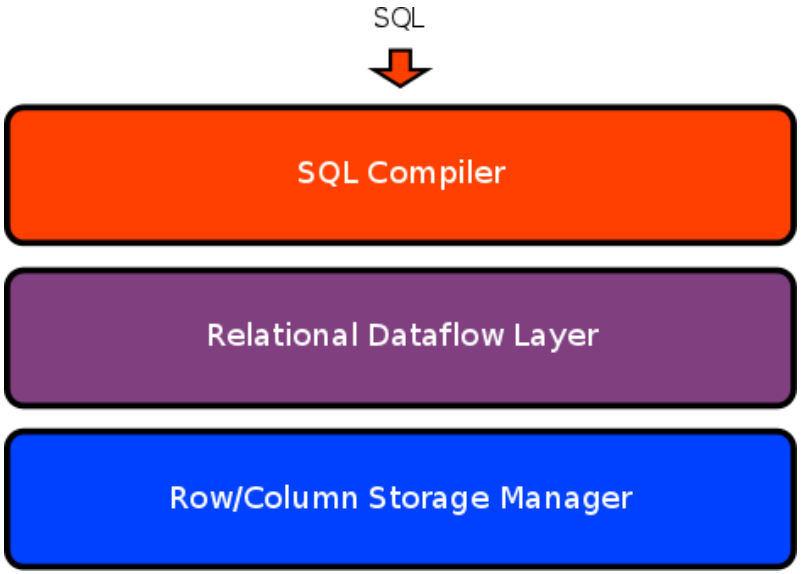
What part of "shared-nothing is the answer" do you IBM bozos not understand...?!?

UCIrvine
http://isg.ics.uci.edu

Big Data in the ***Database*** World

- Enterprise data warehouses
  - 1980's: Shared-nothing parallel DBMSs
  - 2000's: Enter new players (Netezza, Aster Data, DATAllegro, Greenplum, Vertica, ParAccel, ...)
- Scalable OLTP
  - 1980's: Tandem's NonStop SQL





Notes:

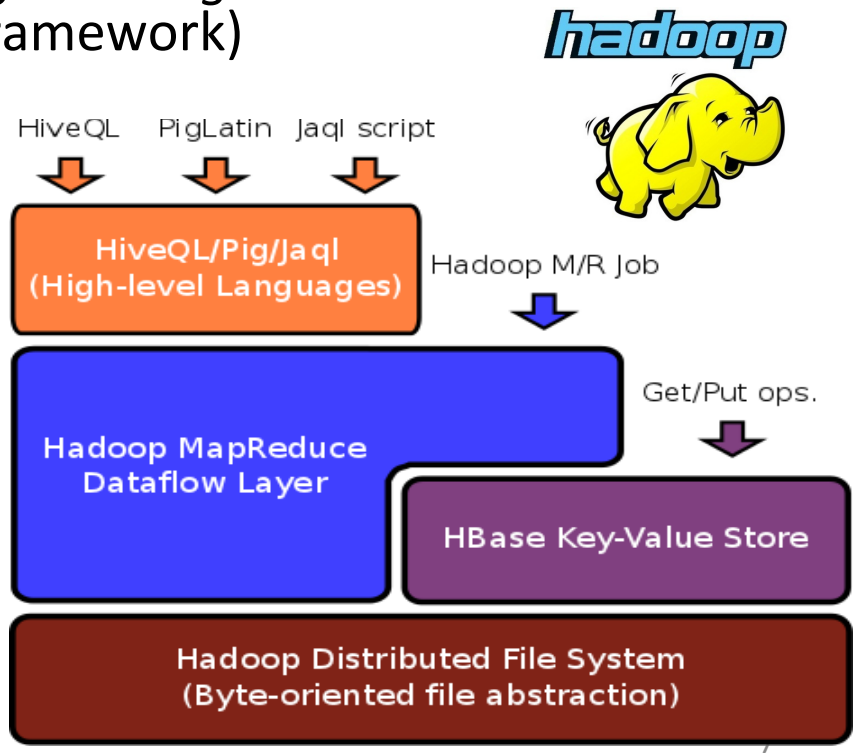
- One storage manager per machine
- Upper layers orchestrate query execution
- ***One way in/out:*** through the SQL door

UCIrvine
http://isg.ics.uci.edu

Later: Big Data in the ***Systems*** World

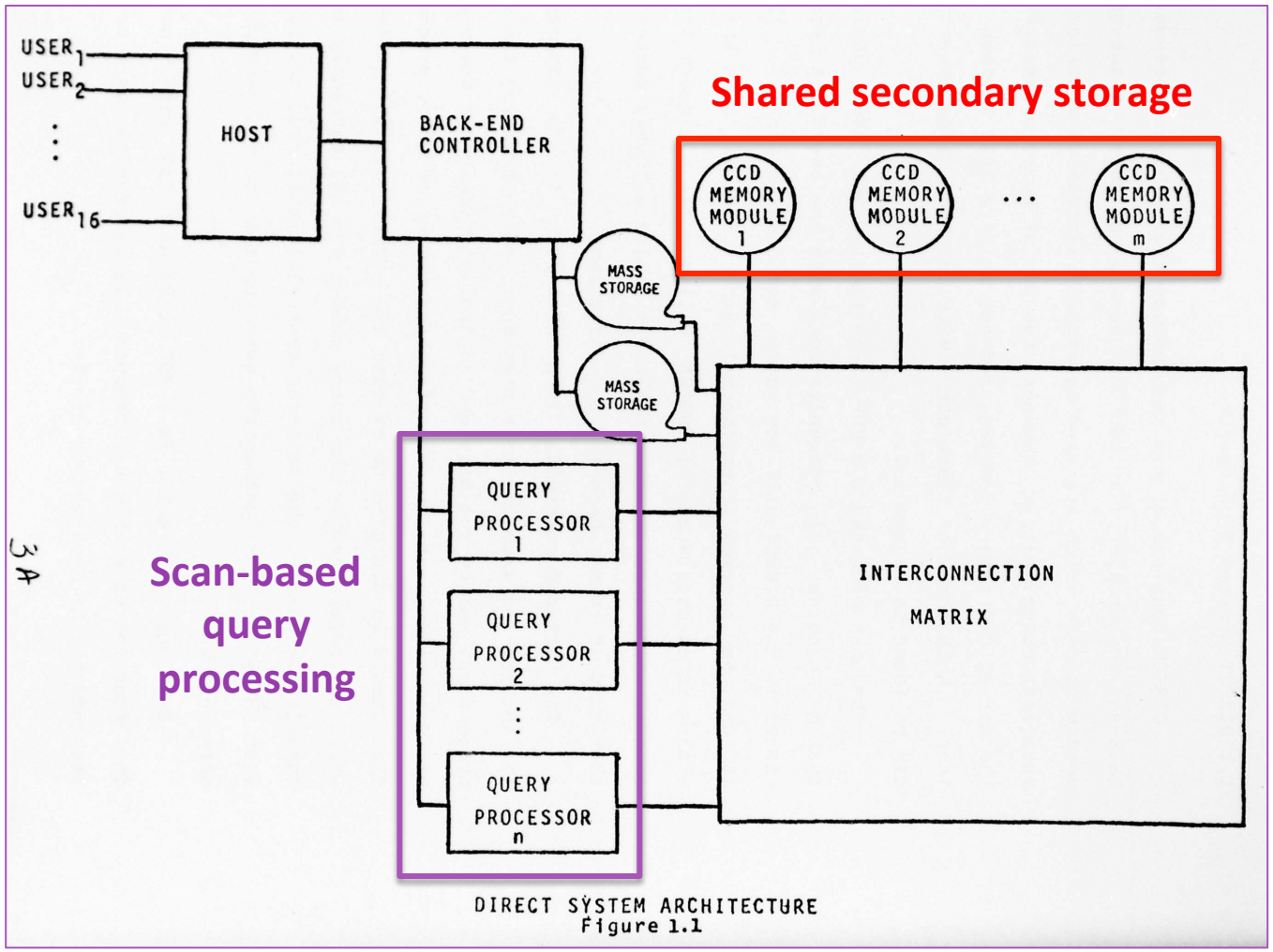
- Out to index and query the Web, Google laid a new foundation in the early 2000's
  - Google File System (GFS): Files spanning many machines with 3-way replication
  - MapReduce (MR): *"Parallel programming for dummies"* (UDFs + parallel framework)

- Yahoo!, FB, et al read the papers
  - HDFS and Hadoop MapReduce
  - Declarative HLLs: Pig, Hive, …
  - HLLs now heavily preferred to MR
- Also key-value stores ("NoSQL")
  - Social sites, online games, …
  - BigTable/HBase, Dynamo/Cassandra, MongoDB, …



UCIrvine
http://isg.ics.uci.edu

Remember History...? *(DIRECT)*



UCIrvine
http://isg.ics.uci.edu

One More Bit of History

Operating System Support for Database Management

Michael Stonebraker
University of California, Berkeley

Communications
of
the ACM

July 1981
Volume 24
Number 7

ting system services are examined
pplicability to support of database
hese services include buffer pool
em; scheduling, process manage-
communication; and consistency

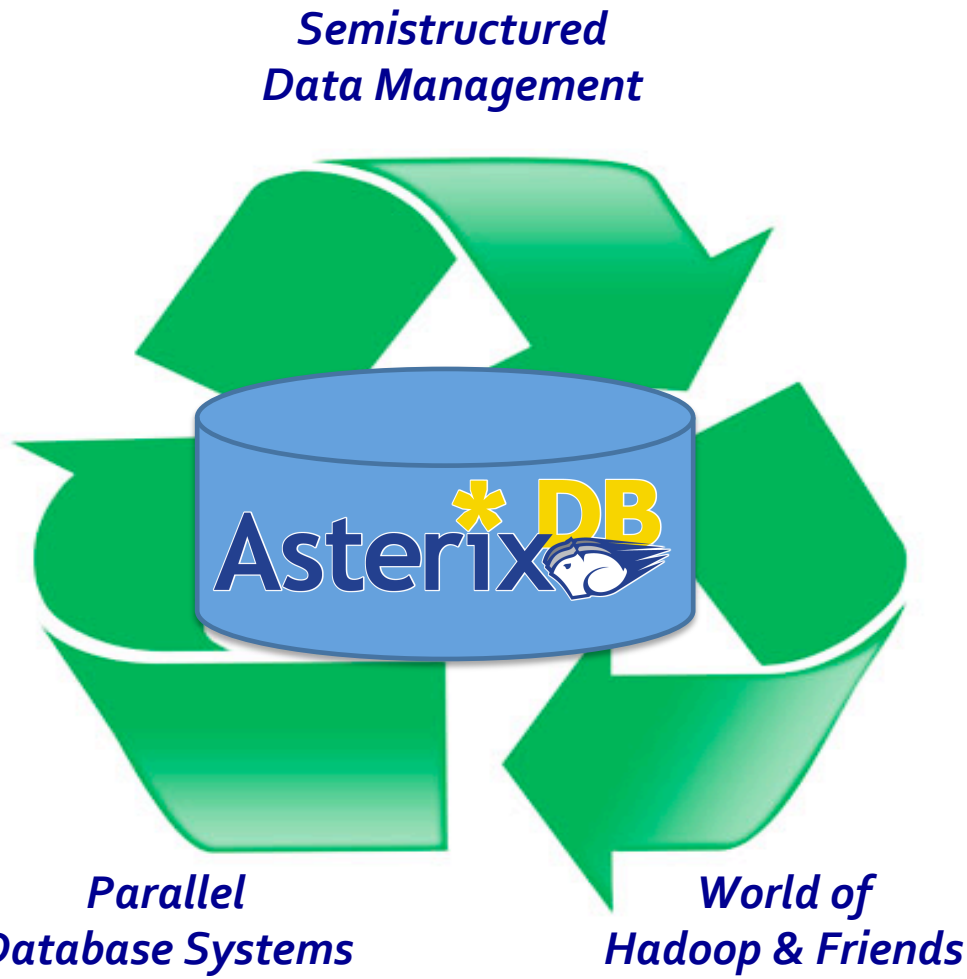
7. Conclusions

The bottom line is that operating system services in many existing systems are either too slow or inappropriate. Current DBMSs usually provide their own and make little or no use of those offered by the operating system. It is important that future operating system designers become more sensitive to DBMS needs.

UCIrvine
http://isg.ics.uci.edu

Plus: Today's Big Data Tangle

AsterixDB: "One Size Fits a Bunch"



BDMS Desiderata:

- Flexible data model
- Efficient runtime
- Full query capability
- Cost proportional to task at hand (!)
- Designed for continuous data ingestion
- Support today's "Big Data data types"

•
•
•

(***Note:*** *This work began in 2009.)*

ASTERIX Data Model (ADM)

```
create dataverse TinySocial;
use dataverse TinySocial;

create type MugshotUserType as {
   id: int32,
   alias: string,
   name: string,
   user-since: datetime,
   address: {
      street: string,
      city: string,
      state: string,
      zip: string,
      country: string
   },
   friend-ids: {{ int32 }},
   employment: [EmploymentType]
}
```

```
create type EmploymentType as open {
   organization-name: string,
   start-date: date,
   end-date: date?
}

create dataset MugshotUsers(MugshotUserType)
      primary key id;
```

Note: We store and manage datasets...

Highlights include:

- JSON++ based data model
- Rich type support (spatial, temporal, …)
- Records, lists, bags
- ***Open*** vs. ***closed*** types

UCIrvine
http://isg.ics.uci.edu

ASTERIX Data Model (ADM)

```
create dataverse TinySocial;
use dataverse TinySocial;

create type MugshotUserType as {
   id: int32
}
```

... and schemas are supported, but optional.

```
create type EmploymentType as open {
   organization-name: string,
   start-date: date,
   end-date: date?
}

create dataset MugshotUsers(MugshotUserType)
        primary key id;
```

Note: We store and manage datasets ...

Highlights include:

- JSON++ based data model
- Rich type support (spatial, temporal, …)
- Records, lists, bags
- ***Open*** vs. ***closed*** types

UCIrvine
http://isg.ics.uci.edu

Other Data Management Features

```
create index msUserSinceIdx on MugshotUsers(user-since);
create index msTimestampIdx on MugshotMessages(timestamp);
create index msAuthorIdx  on MugshotMessages(author-id) type btree;
create index msSenderLocIndex on MugshotMessages(sender-location) type rtree;
create index msMessageIdx on MugshotMessages(message) type keyword;

create type AccessLogType as closed
   { ip: string, time: string, user: string, verb: string, path: string,  stat: int32, size: int32 };
create external dataset AccessLog(AccessLogType) using localfs
    (("path"="{hostname}://{path}"), ("format"="delimited-text"), ("delimiter"="|"));

create feed socket_feed using socket_adaptor
    (("sockets"="{address}:{port}"), ("addressType"="IP"),
     ("type-name"="MugshotMessageType"), ("format"="adm"));
connect feed socket_feed to dataset MugshotMessages;
```

UCIrvine
http://isg.ics.uci.edu

ASTERIX Query Language (AQL)

- *Ex:* Identify active users and group/count them by country:

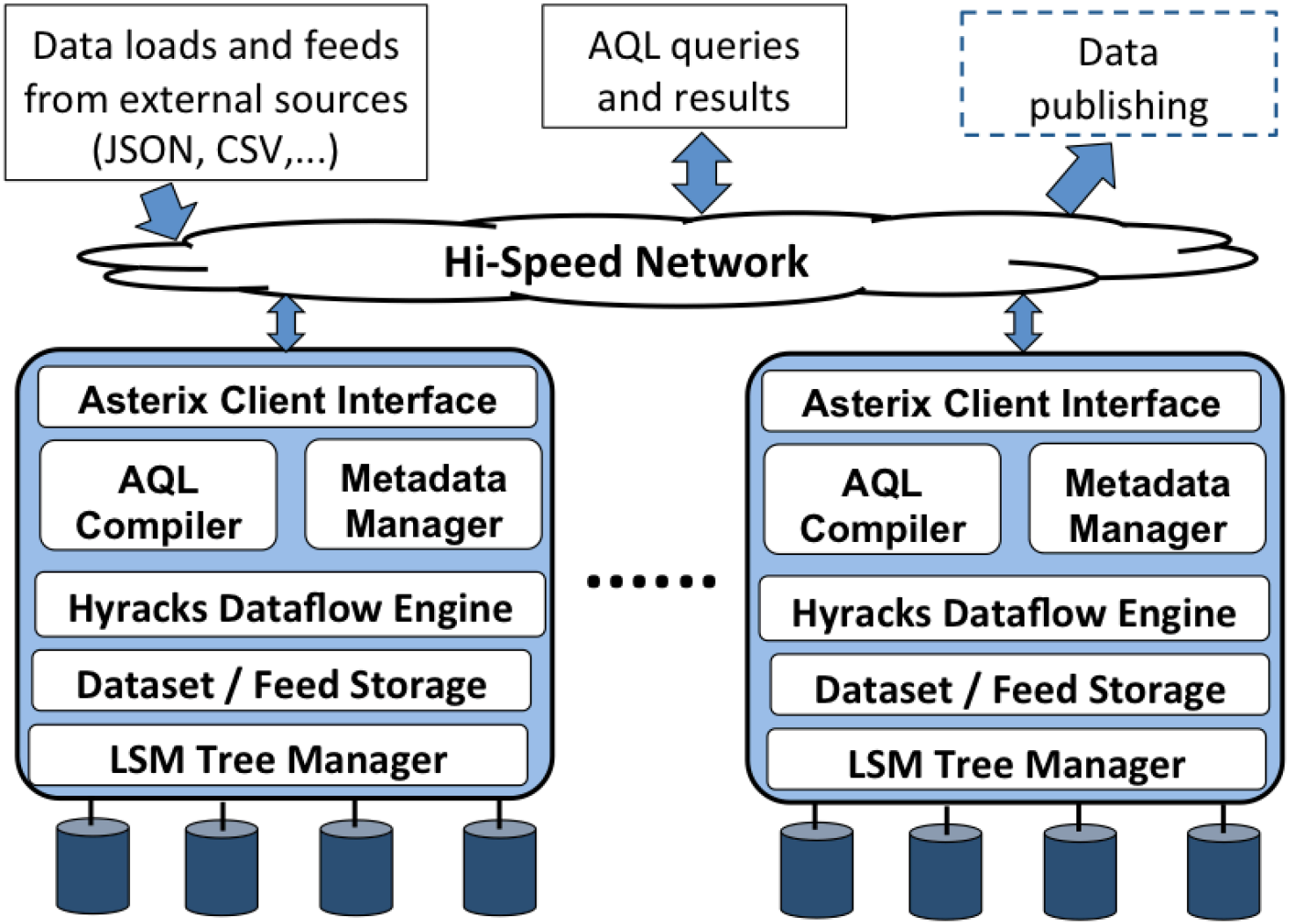
```
with $end := current-datetime()
with $start := $end - duration("P30D")
from $user in dataset MugshotUsers
where some $logrecord in dataset AccessLog
  satisfies $user.alias = $logrecord.user
  and datetime($logrecord.time) >= $start
  and datetime($logrecord.time) <= $end
group by $country := $user.address.country keeping $user
select {
  "country" : $country,
  "active users" : count($user)
}
```

***AQL highlights*:**

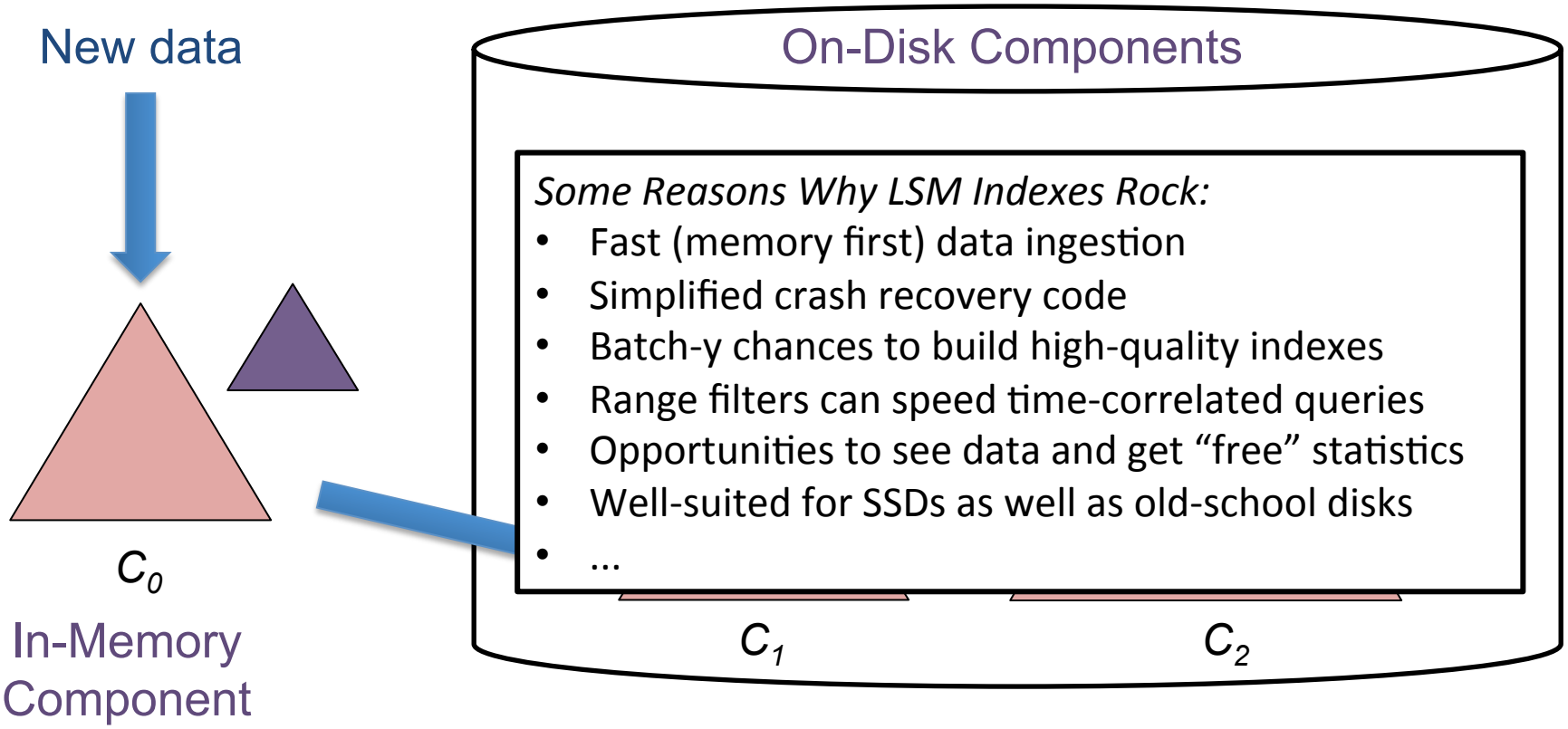
- Lots of other features (see website!)
- Spatial predicates and aggregation
- Set-similarity matching
- And plans for more…

UCIrvine
http://isg.ics.uci.edu

AsterixDB System Overview



Local LSM-Based Storage & Indexes



LSM-ified Indexes:

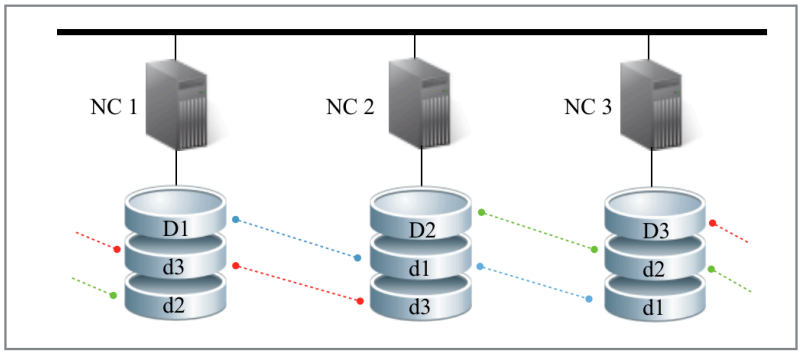
- B+ trees
- R trees (secondary)
- Inverted (secondary)

UCIrvine
http://isg.ics.uci.edu

Distributed Storage in AsterixDB

- Hash-partitioned, shared-nothing, local drives
  - Partitioning based on primary key (hashing)
  - Secondary indexes local to, and consistent with, corresponding primary partitions (all LSM-based)
- Also offer external dataset feature (for HDFS)
  - Multiple (Hive) formats, secondary index support
  - Index partitions co-located with data (if possible)
  - Developed for space and "IT comfort" reasons

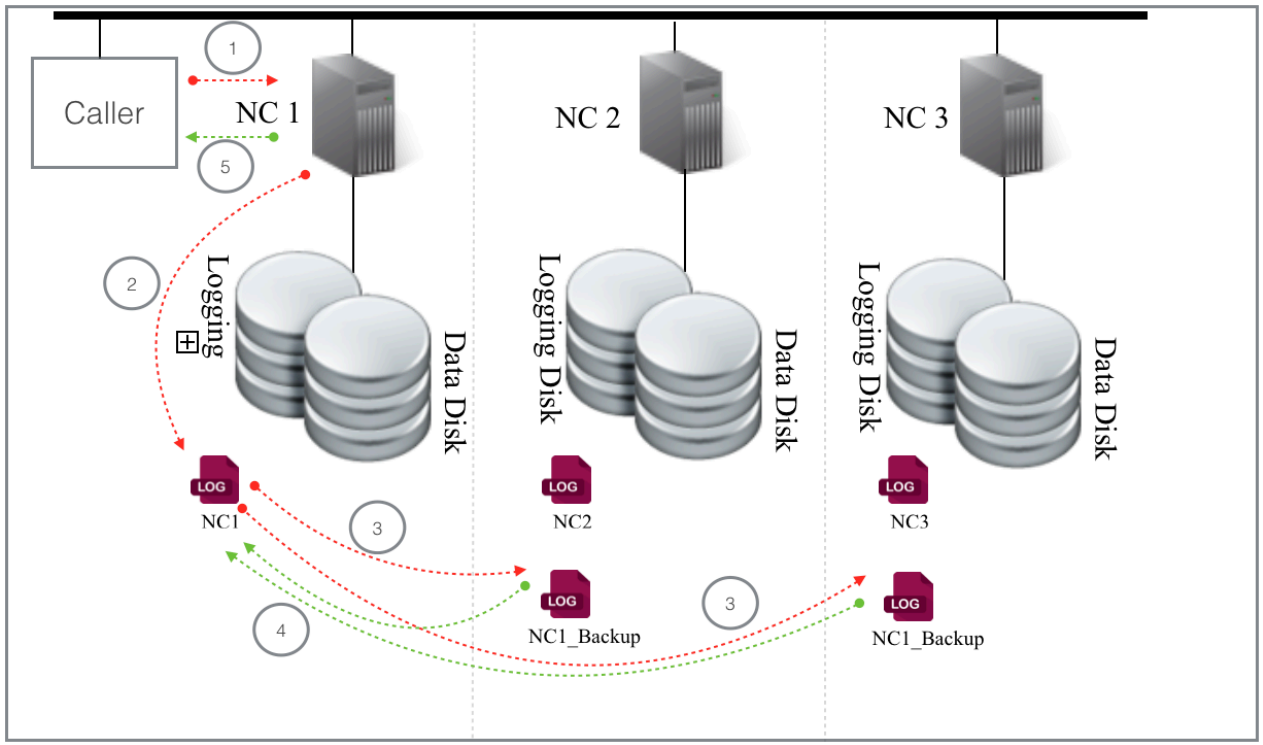
Data Replication in AsterixDB *(WIP)*



Chained Declustering

Log-Based Replication

(synchronous, recovery-only copies kept)



UCIrvine
http://isg.ics.uci.edu

Hedging Our Bets

- We're currently porting our LSM-based storage system to also work on *top* of HDFS (and YARN)
  - Might somehow feel more "comforting" (and/or "environmentally friendly") to Big Data IT shops
  - Another path to replication and high availability
- Interesting experiments lie ahead!
  - Revisit Stonebraker-like OS issues (modern version)
  - Bake-off: Distributed record management *vs.* DFS
  - Just how well does HDFS do *w.r.t.* locality of writes?

What About the Cloud?

- Computing may be elastic, but data is not...!
  - Native storage → hard to expand & contract
  - Seems to demand a shared-disk-like approach based on cloud storage facilities?
- Experimentation is needed
  - E.g., AWS storage or Google persistent disks
  - Performance implications seem pretty interesting to explore...

For More AsterixDB Info...

NSF project page (UC Irvine and UC Riverside):

- http://asterixdb.ics.uci.edu

Apache AsterixDB (Incubating) project page:

- https://asterixdb.incubator.apache.org/

I Asked the Questions, So....

- Got any answers?