6.5830/6.5831
Introduction to Databases

6.5830 Lecture 1- 9/13/2022

Sam Madden
madden@csail.mit.edu

http://dsg.csail.mit.edu/6.5830

Mike Cafarella

Geoffrey Yu

Sam Madden

Anna Zeng

Course Staff

Administrivia

http://dsg.csail.mit.edu/6.5830
Email: 6.5830-staff@mit.edu
Ask questions on Piazza! Use sign up link on website.

Lecturers:

Sam Madden (madden@csail.mit.edu)
Michael Cafarella (michjc@csail.mit.edu)

TAs:

Geoffrey Yu (geoffxy@mit.edu)
Anna Zeng (annazeng@mit.edu)

Office hours: TBD (weekly)

Textbooks

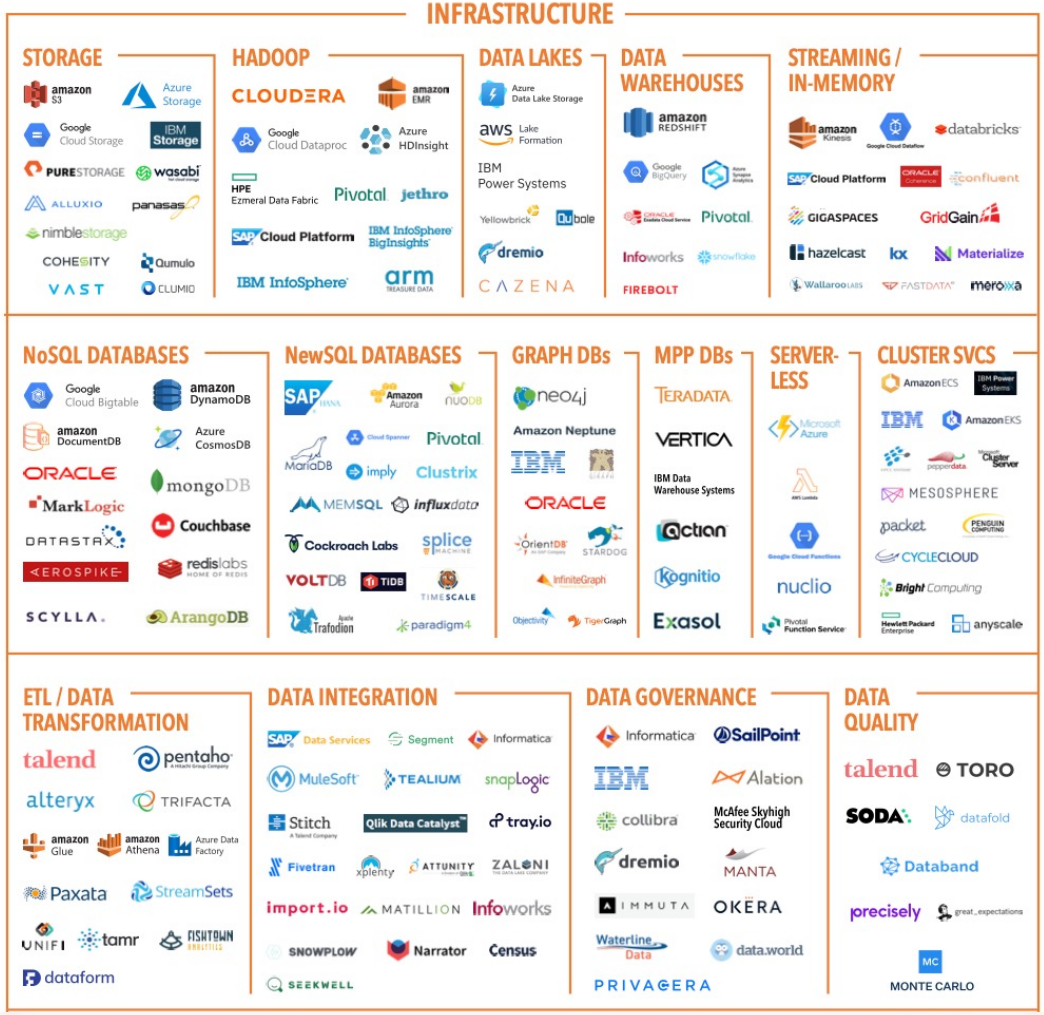
- Readings in Database Systems
  - http://www.redbook.io

- Rest of readings will be drawn from literature (research papers and web pages)

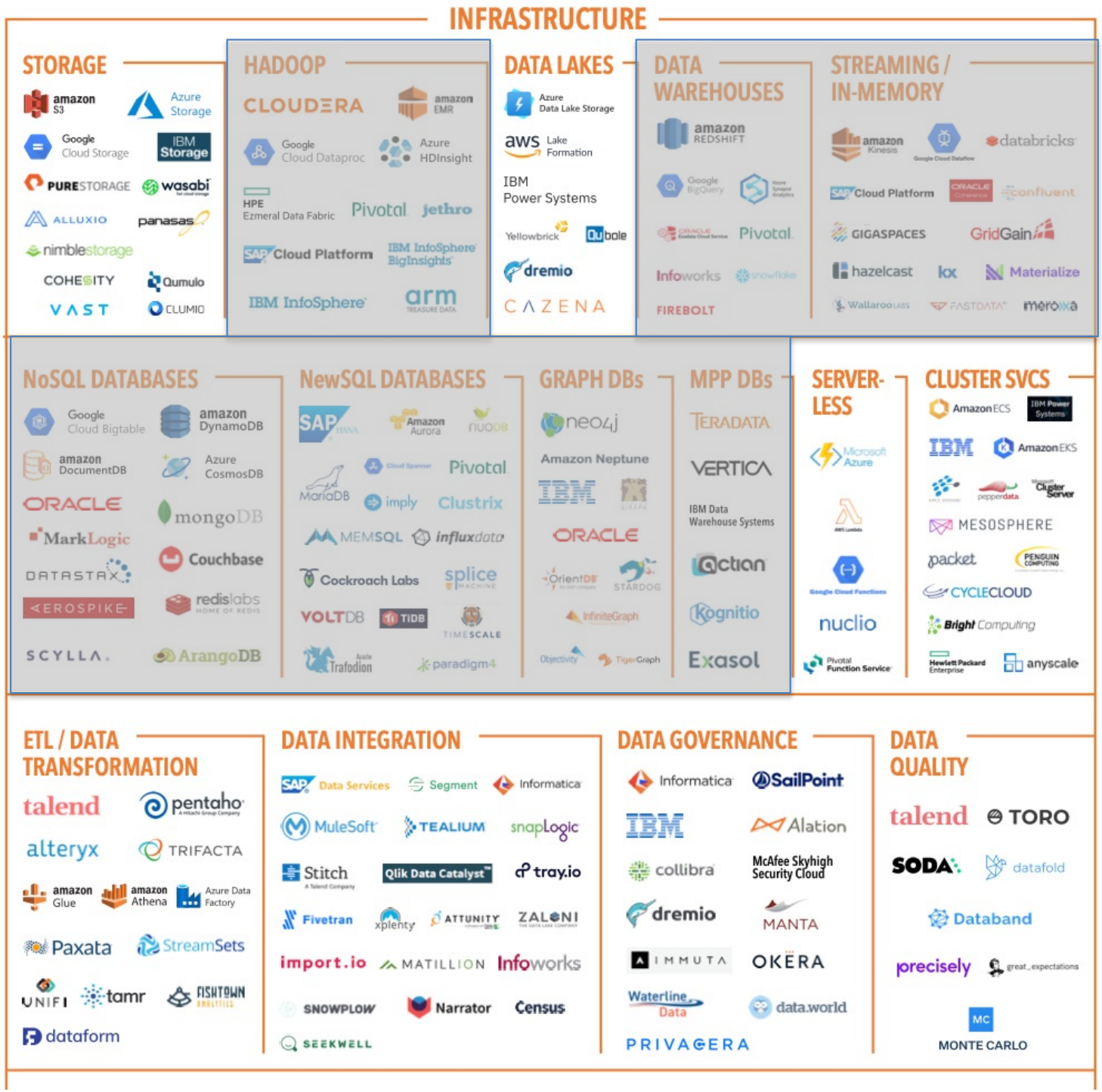
What is A Database?

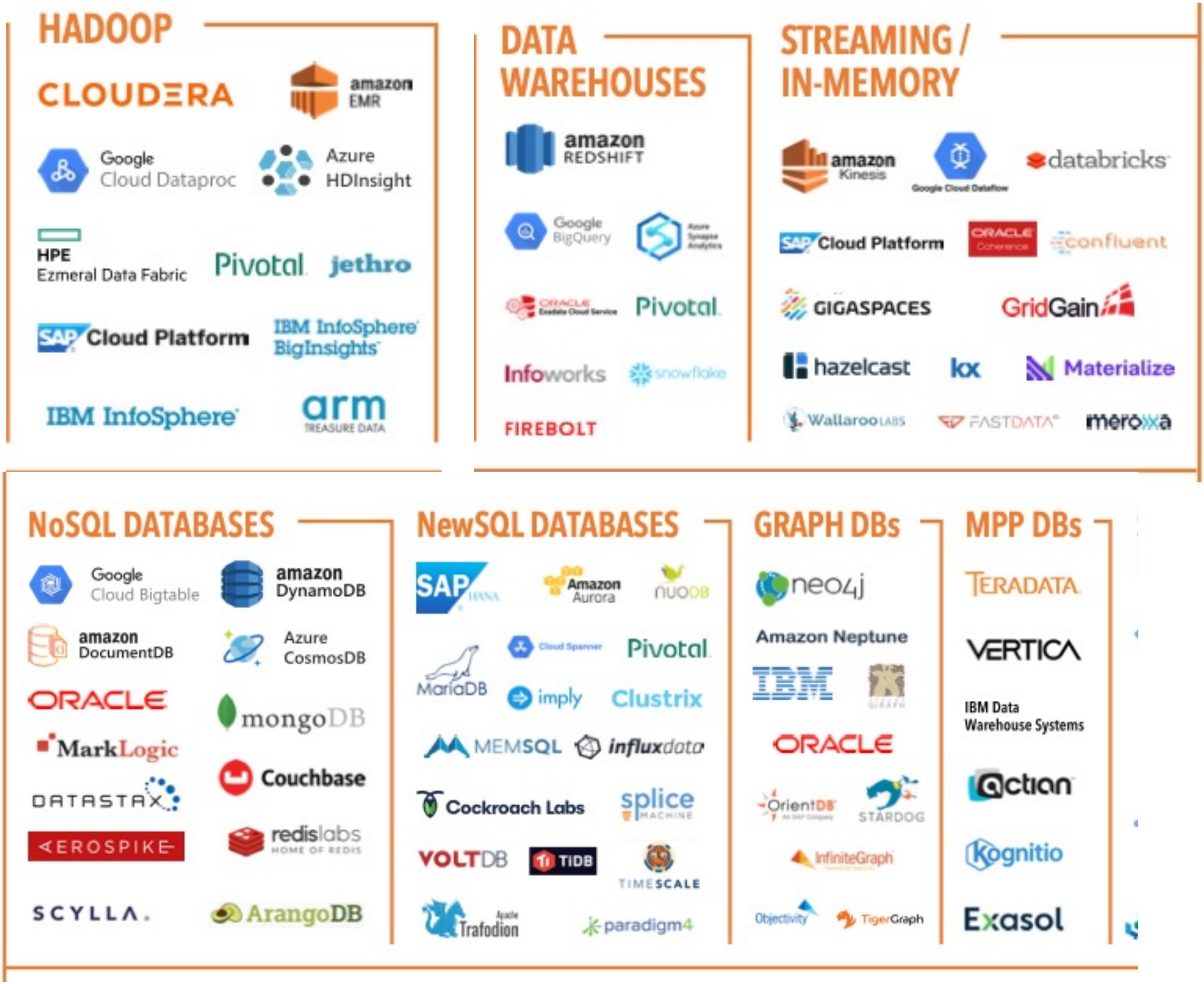
- Structured Data Collection
  - Records
  - Relationships

- This class: Database Management Systems (DBMSs)
  - Software systems for storing and querying databases



Source: mattturck.com


INFRASTRUCTURE
STORAGE
amazon S3
Azure Storage
Google Cloud Storage
IBM Storage
PURESTORAGE
wasabi
ALLUXIO
panasas
nimblestorage
COHESITY
Qumulo
VAST
CLUMIO
HADOOP
CLOUDERA
amazon EMR
Google Cloud Dataproc
Azure HDInsight
HPE Ezmeral Data Fabric
Pivotal
jethro
SAP Cloud Platform
IBM InfoSphere BigInsights
IBM InfoSphere
arm TREASURE DATA
DATA LAKES
Azure Data Lake Storage
aws Lake Formation
IBM Power Systems
Yellowbrick
bole
dremio
CAZENA
DATA WAREHOUSES
amazon REDSHIFT
Google BigQuery
Azure Synapse Analytics
Pivotal
Infoworks
snowflake
FIREBOLT
STREAMING / IN-MEMORY
amazon Kinesis
Google Cloud Dataflow
databricks
SAP Cloud Platform
confluent
GIGASPACES
GridGain
hazelcast
kx
Materialize
Wallaroo LABS
FASTDATA
meroxa
NoSQL DATABASES
Google Cloud Bigtable
amazon DynamoDB
amazon DocumentDB
Azure CosmosDB
ORACLE
mongoDB
MarkLogic
Couchbase
DATASTAX
redislabs
AEROSPIKE
SCYLLA.
ArangoDB
NewSQL DATABASES
SAP HANA
Amazon Aurora
NUODB
MariaDB
Cloud Spanner
Pivotal
imply
Clustrix
MEMSQL
influxdata
Cockroach Labs
splice MACHINE
VOLTDB
TiDB
TIMESCALE
Apache Trafodion
paradigm4
GRAPH DBs
neo4j
Amazon Neptune
IBM
ORACLE
OrientDB
STARDOG
InfiniteGraph
Objectivity
TigerGraph
MPP DBs
TERADATA
VERTICA
IBM Data Warehouse Systems
Actian
Kognitio
Exasol
SERVER-LESS
Microsoft Azure
AWS Lambda
Google Cloud Functions
nuclio
Pivotal Function Service
CLUSTER SVCS
Amazon ECS
IBM Power Systems
IBM
Amazon EKS
pepperdata
Microsoft Cluster Server
MESOSPHERE
packet
PENGUIN COMPUTING
CYCLECLOUD
Bright Computing
Hewlett Packard Enterprise
anyscale
ETL / DATA TRANSFORMATION
talend
pentaho
alteryx
TRIFACTA
amazon Glue
amazon Athena
Azure Data Factory
Paxata
StreamSets
UNIFI
tamr
FISHTOWN ANALYTICS
dataform
DATA INTEGRATION
SAP Data Services
Segment
Informatica
MuleSoft
TEALIUM
snapLogic
Stitch
Qlik Data Catalyst
tray.io
Fivetran
xplenty
ATTUNITY
ZALONI
import.io
MATILLION
Infoworks
SNOWPLOW
Narrator
Census
SEEKWELL
DATA GOVERNANCE
Informatica
SailPoint
IBM
Alation
collibra
McAfee Skyhigh Security Cloud
dremio
MANTA
IMMUTA
OKERA
Waterline Data
data.world
PRIVACERA
DATA QUALITY
talend
TORO
SODA
datafold
Databand
precisely
great_expectations
MC
MONTE CARLO


HADOOP
CLOUDERA
amazon EMR
Google Cloud Dataproc
Azure HDInsight
HPE Ezmeral Data Fabric
Pivotal
jethro
SAP Cloud Platform
IBM InfoSphere BigInsights
IBM InfoSphere
arm TREASURE DATA
DATA WAREHOUSES
amazon REDSHIFT
Google BigQuery
Azure Synapse Analytics
ORACLE Exadata Cloud Service
Pivotal
Infoworks
snowflake
FIREBOLT
STREAMING / IN-MEMORY
amazon Kinesis
Google Cloud Dataflow
databricks
SAP Cloud Platform
ORACLE Coherence
confluent
GIGASPACES
GridGain
hazelcast
kx
Materialize
WallarooLABS
FASTDATA
meroxa
NoSQL DATABASES
Google Cloud Bigtable
amazon DynamoDB
amazon DocumentDB
Azure CosmosDB
ORACLE
mongoDB
MarkLogic
Couchbase
DATASTAX
redislabs HOME OF REDIS
AEROSPIKE
SCYLLA.
ArangoDB
NewSQL DATABASES
SAP HANA
Amazon Aurora
NUODB
MariaDB
Cloud Spanner
Pivotal
imply
Clustrix
MEMSQL
influxdata
Cockroach Labs
splice MACHINE
VOLTDB
TiDB
TIMESCALE
Apache Trafodion
paradigm4
GRAPH DBs
neo4j
Amazon Neptune
IBM
ORACLE
OrientDB
STARDOG
InfiniteGraph
Objectivity
TigerGraph
MPP DBs
TERADATA
VERTICA
IBM Data Warehouse Systems
actian
Kognitio
Exasol

6.5830 Concepts

- Data modeling / layout
- Declarative querying
  - Query processing
  - Algorithms for accessing and manipulating data
- Consistency / Transactions (“ACID”)
- “Big Data” – scaling to massive volumes, many machines

MAPD (MASSIVELY PARALLEL DATABASE)

USING GPUS FOR REAL-TIME QUERYING AND VISUALIZATION OF BIG DATA

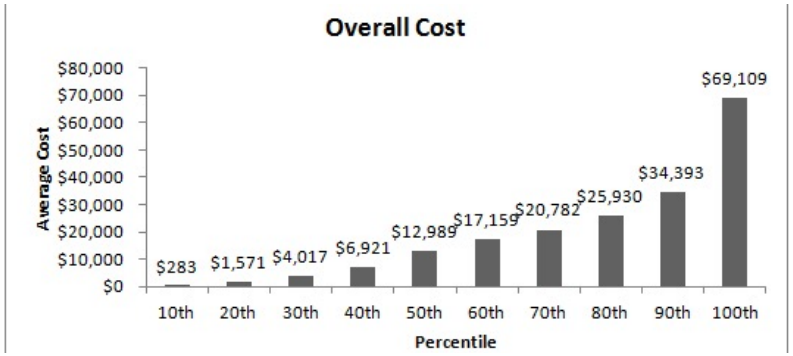
Interactive Large-Scale Visualization using a GPU Database

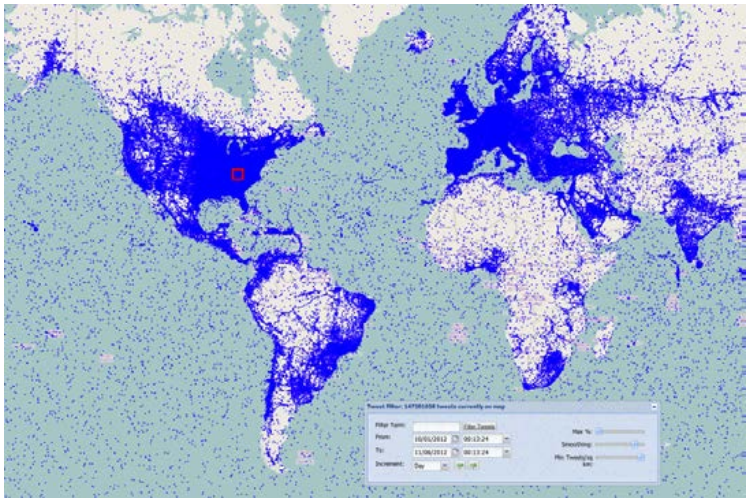
Todd Mostak

The Need for Interactive Analytics

- Idea: often need to browse massive data sets
- Browsing is best supported through visualization

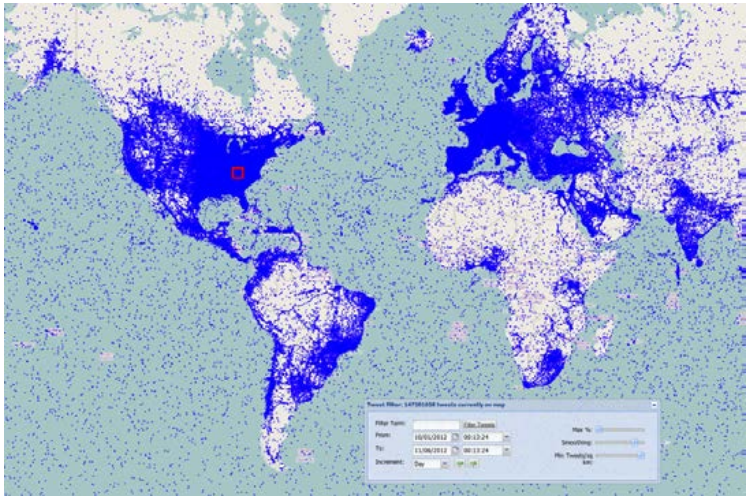
➔ ad-hoc analytics, with millisecond response times



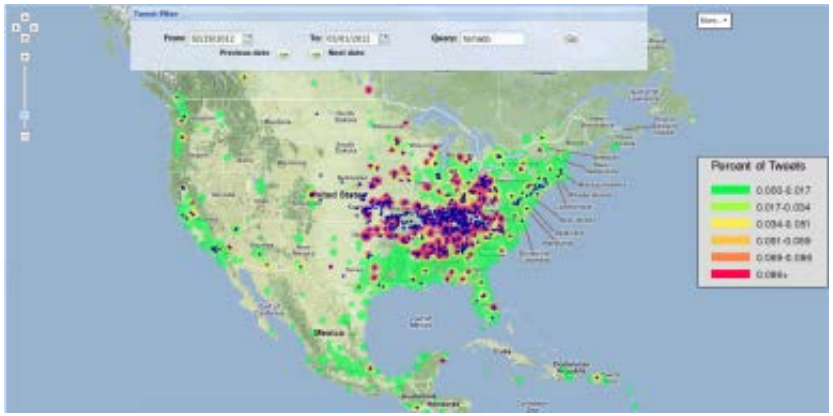


MapD: GPU Accelerated SQL Database

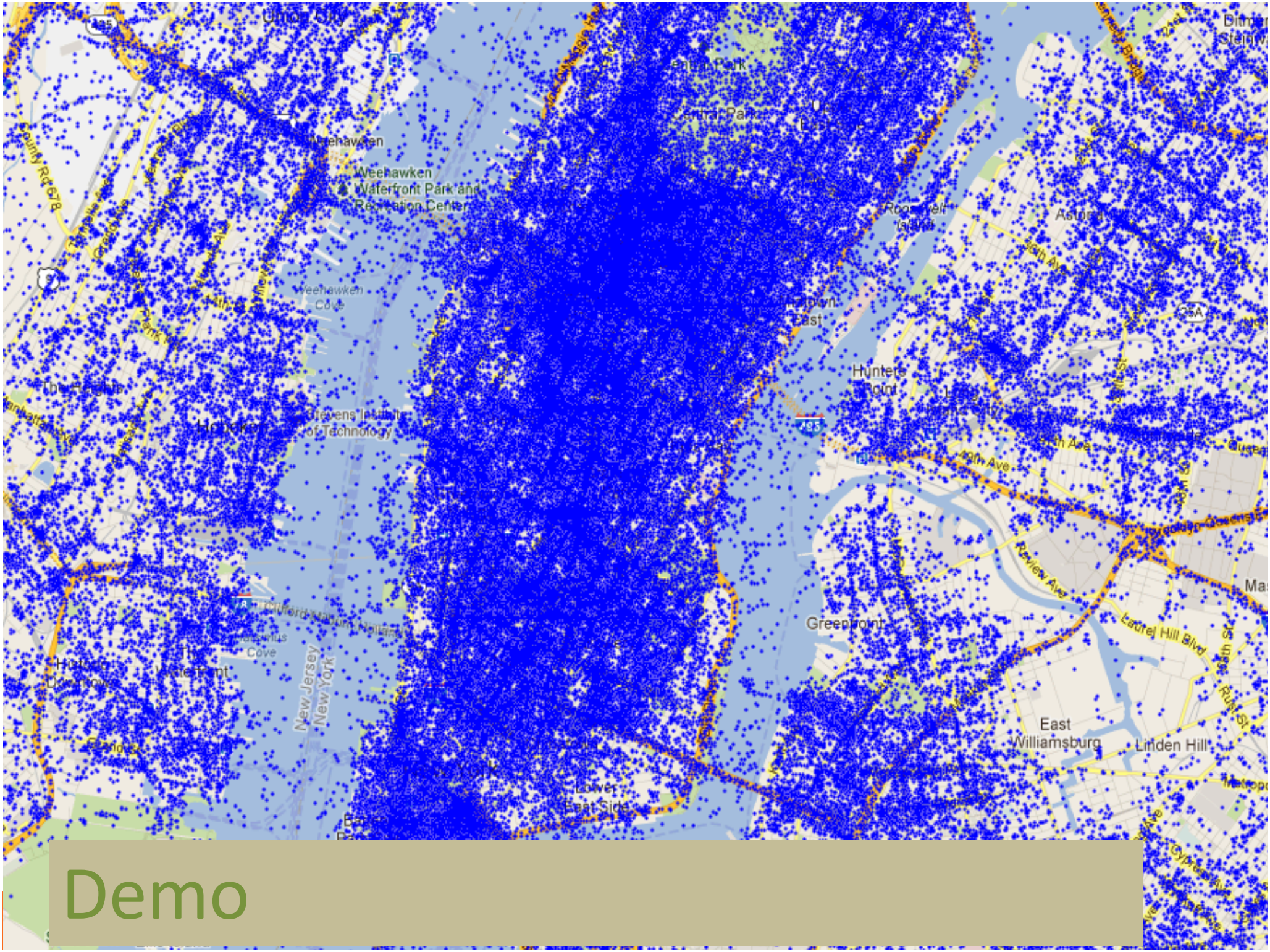
- *Key insight:* GPUs have enough memory that a cluster of them can store substantial amounts of data
- Not an accelerator, but a full blown query processor!
- Massive parallelism enables interactive browsing interfaces
  - 4x GPUs can provide > 1 TB/sec of bandwidth
  - 12 Tflops compute
  - Order of magnitude speedups over CPUs, when data is on GPU
- "Shared nothing" arrangement



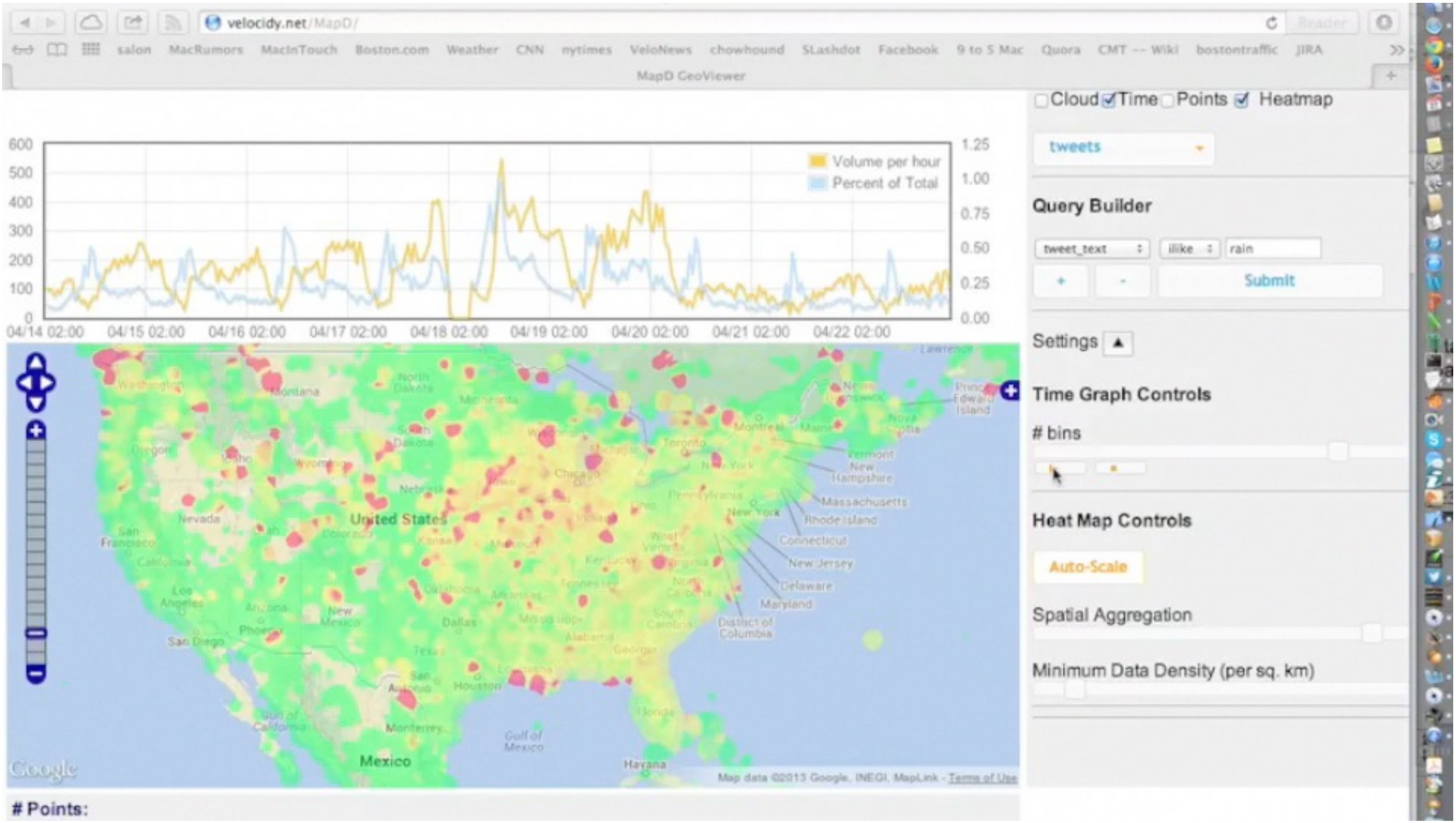
147,201,658 tweets from Oct 1, 2012 to Nov 6, 2012



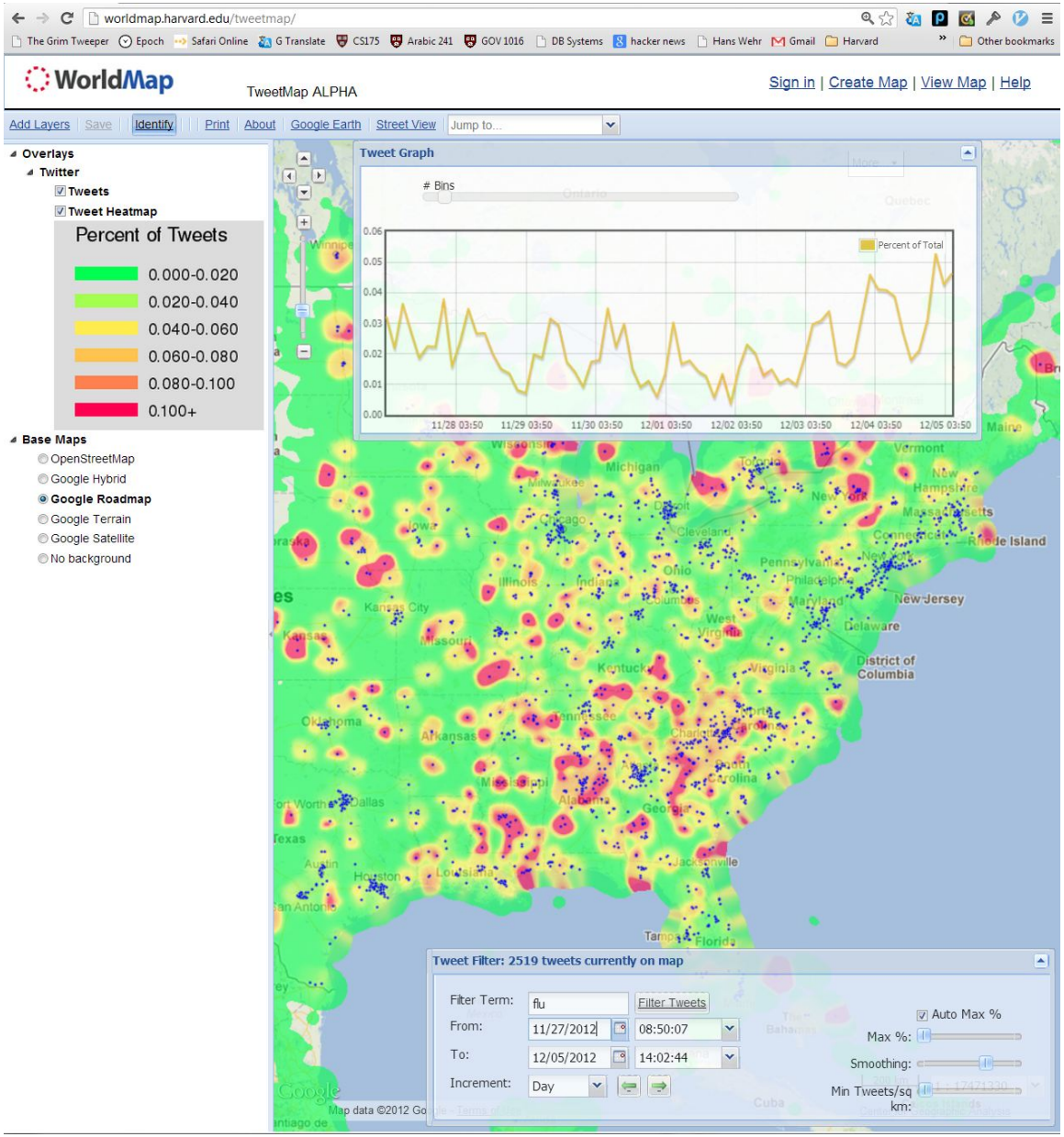
Relative intensity of "tornado" on Twitter (with point overlay) from Febuary 29, 2012 to March 1, 2012


Weehawken
Waterfront Park and
Recreation Center
Weehawken
Cove
Stevens Institute
of Technology
Hunters
Point
Greenpoint
East
Williamsburg
Linden Hill
East Side
New Jersey
New York
Laurel Hill Blvd
Review Ave
Newtown Creek
495

Demo


velocidy.net/MapD/
MapD GeoViewer
Cloud
Time
Points
Heatmap
tweets
Query Builder
tweet_text
ilike
rain
+
-
Submit
Volume per hour
Percent of Total
04/14 02:00
04/15 02:00
04/16 02:00
04/17 02:00
04/18 02:00
04/19 02:00
04/20 02:00
04/21 02:00
04/22 02:00
Settings
Time Graph Controls
bins
Heat Map Controls
Auto-Scale
Spatial Aggregation
Minimum Data Density (per sq. km)
United States
Mexico
Map data ©2013 Google, INEGI, MapLink - Terms of Use
Points:


Tweet Filter: 2082 tweets currently on map
Filter Term:
central park
Filter Tweets
From:
10/14/2012
20:17:29
To:
11/04/2012
20:17:29
Increment:
Day
Max %:
Smoothing:
Min Tweets/sq km:
Manhattan
Central Park
Upper West Side
Upper East Side
Midtown East
Theater District
Times Square
Hell's Kitchen
Chelsea
Greenwich Village
West Village
Union Square
Flatiron District
Gramercy Park
Kips Bay
Chinatown
New York
East Harlem
Randalls Island
Port Morris
Marcus Garvey
Riverside Park
Guttenberg
West New York
Bulls Ferry & Jacob's Ferry
Ditmars Steinway
LaGuardia Airport
East Elmhurst
Astoria
Jackson Heights
Maspeth
Greenpoint
Rego Park
Middle Village
Linden Hill



Search for “flu” showing outbreak over Southeastern U.S.

Today

- Why database systems?

- User's perspective:
  - Modeling data
  - Querying data

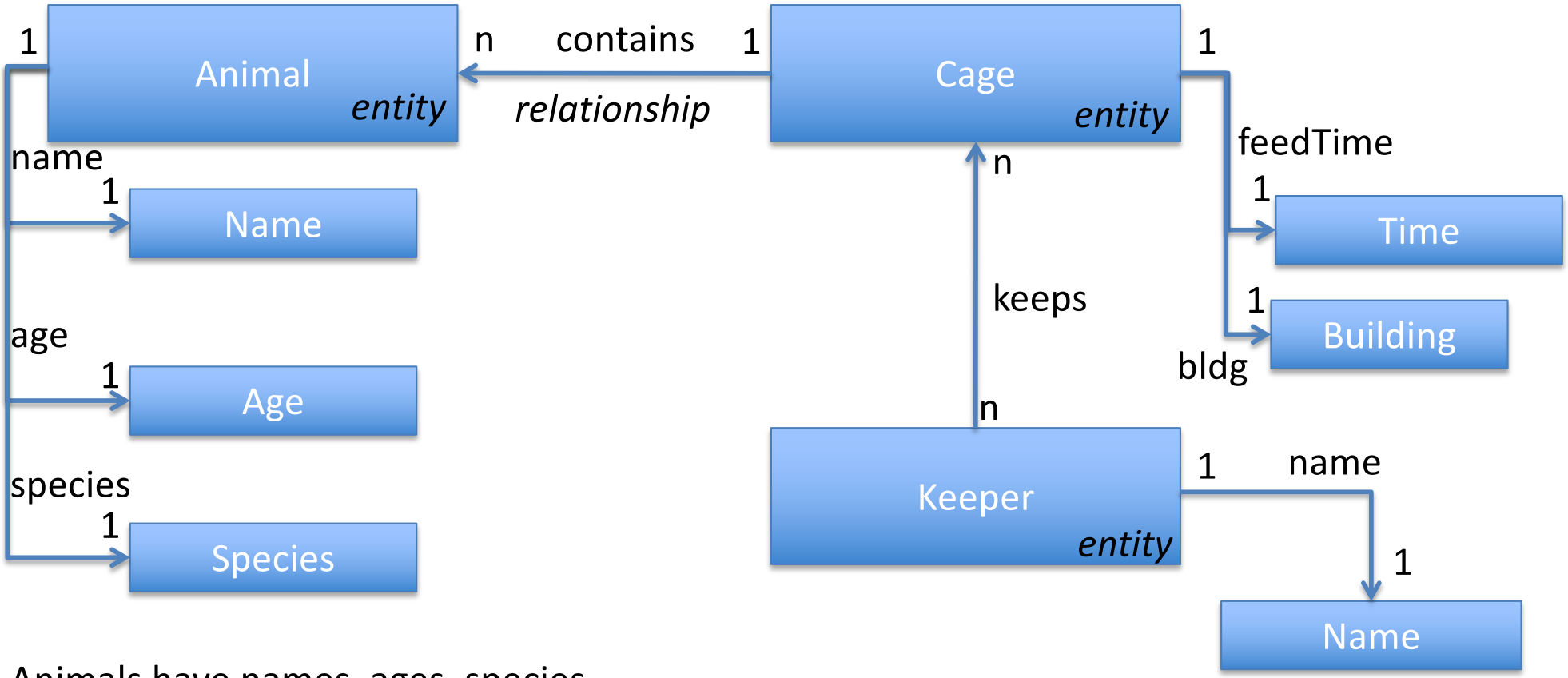
Zoo Website Features

- Admin interface
  - Edit
  - Add an animal
- Public
  - Pictures & Maps
- Zookeeper
  - Feed times

- 1K animals, 5K URLs, 10 admins, 200 keepers

Zoo Data Model

Entity Relationship Diagram



Animals have names, ages, species
Keepers have names
Cages have cleaning times, buildings
Animals are in 1 cage; cages have multiple animals
Keepers keep multiple cages, cages kept by multiple keepers

Our Zoo

Sam the Salamander

Slimy

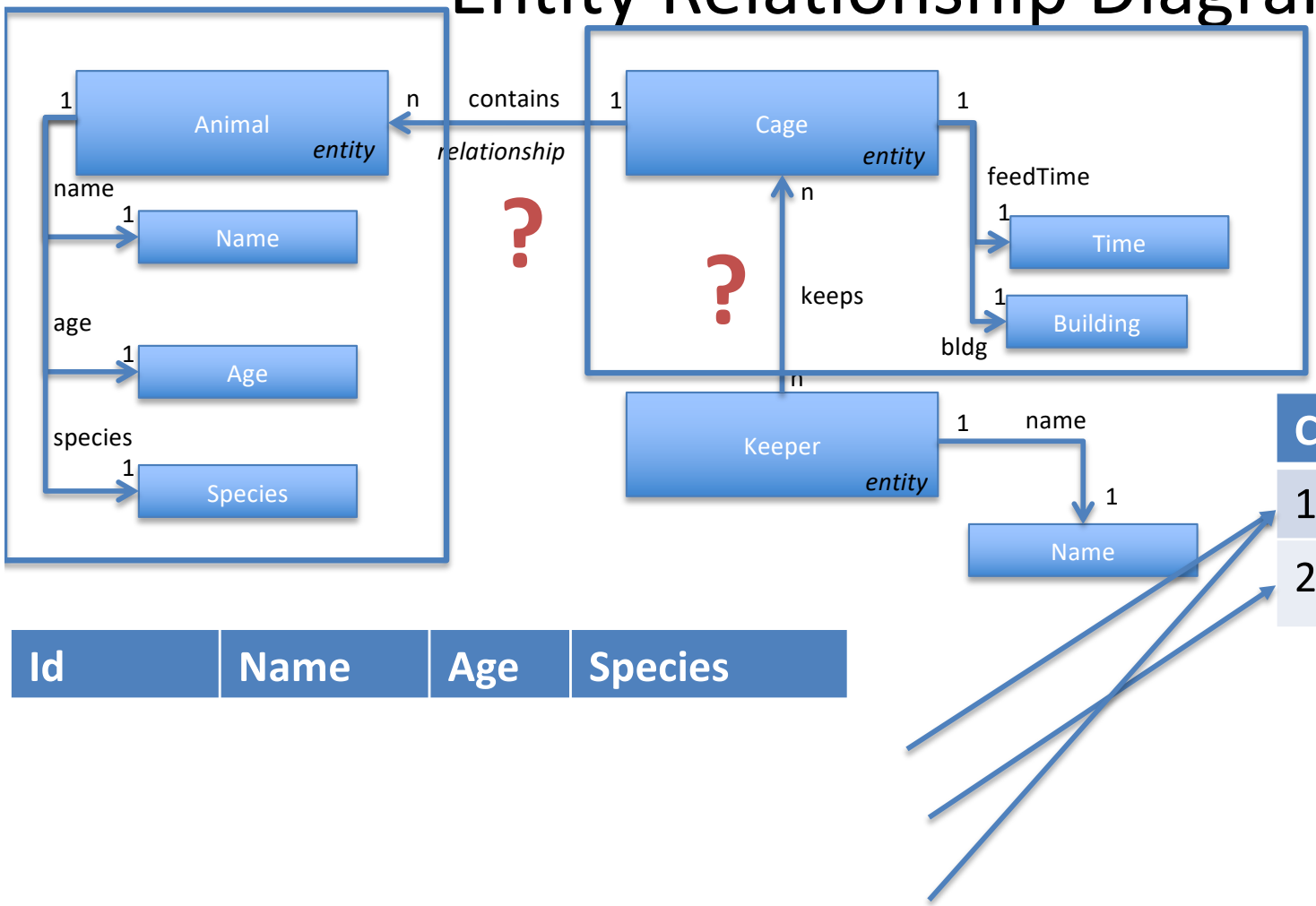
Mike the Giraffe

Lanky

Sally the Student

Zoo Data Model

Entity Relationship Diagram

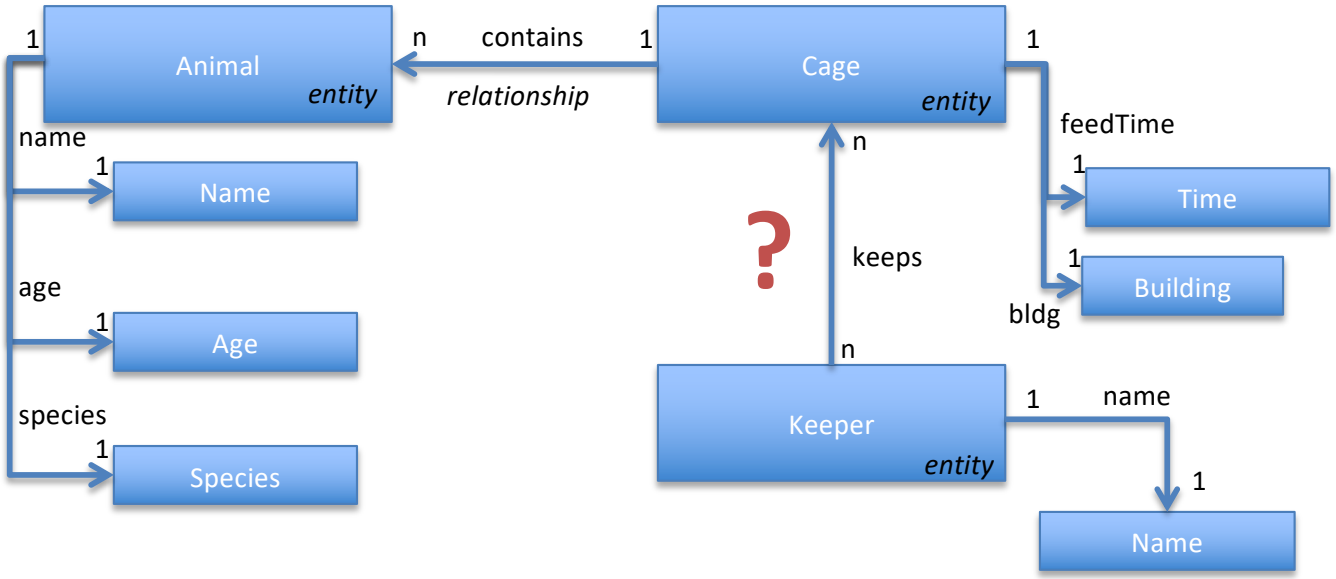


Id	Name	Age	Species

Cageid	Feedtime	Bldg
1	1:30	1
2	2:30	2

Zoo Data Model

Entity Relationship Diagram



Cageid	Feedtime	Bldg
1	1:30	1
2	2:30	2

keeperid	name
1	jenny
2	joe

keeperid	cageid
1	1
1	2
2	1

Break

- Questions
  - Are there other ways to represent this zoo data than a collection of tables?
  - What are tradeoffs in different representations?

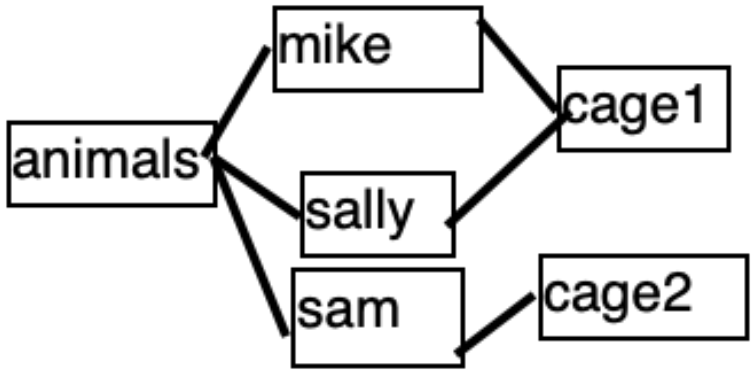
Alternatives to Relations

Hierarchy

cage 1
- mike
  - giraffe
  - 13 yrs
- sally
  - student
  - 1 yr

cage 2
- sam
  - salamander
  - 3 yrs

Graph



Multiple Tabular Representations Are Possible

name	age	species	cageno	feedtime	bldg
mike	13	giraffe	1	1:30	1
sam	3	salamander	2	2:30	2
sally	1	student	1	1:30	1

Is this a good representation? Why or why not?

Not “Normalized” – repeats data. More in later lectures!

SQL – Structured Query Language

```
SELECT  field1, …, fieldM
FROM table1, …
WHERE condition1, …


INSERT INTO table VALUES (field1, …)


UPDATE table SET field1 = X, …
WHERE condition1,…
```

Names of Giraffes

- Imperative

```
for each row r in animals
   if r.species = ‘giraffe’
      output r.name
```

- Declarative

```
SELECT r.name FROM animals
WHERE r.species = ‘giraffe’
```

Cages in Building 32

- Imperative

NESTED LOOPS

```
for each row a in animals
   for each row c in cages
      if a.cageno = c.no and c.bldg = 32
         output a
```

- Declarative

JOIN

```
SELECT a.name FROM animals AS a, cages AS c
WHERE a.cageno = c.no AND c.bldg = 32
```

Average Age of Bears

- Declarative

```
SELECT AVG(age) FROM animals
WHERE species = 'bear'
```

Complex Queries

Find pairs of animals of the same species and different genders older than 1 year:

```
SELECT a1.name,a2.name
FROM animals as a1, animals as a2
WHERE a1.gender = M and a2.gender = F
AND a1.species = a2.species
AND a1.age > 1 and a2.age > 1
```

"self join"

Find cages with salamanders fed later than the average feedtime of any cage:

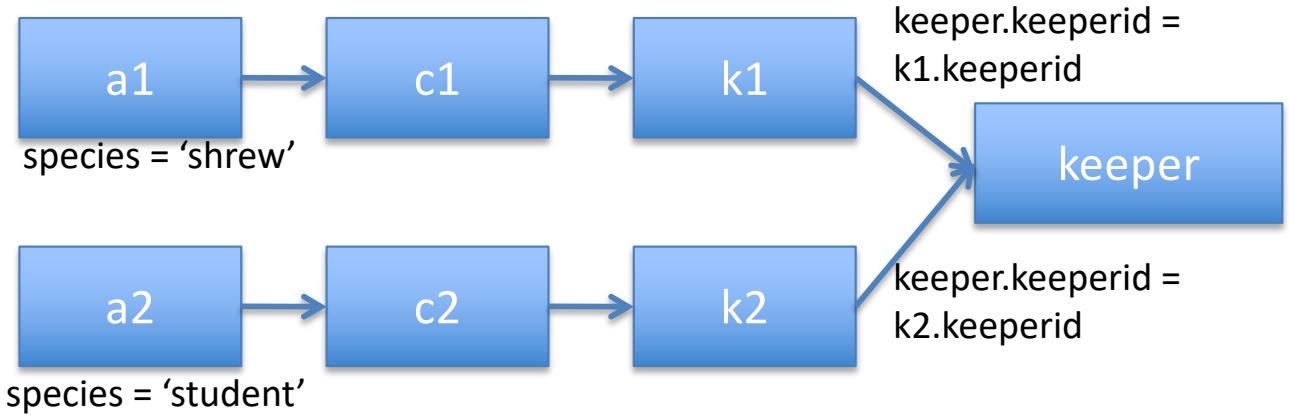
```
SELECT cages.cageid FROM cages, animals
WHERE animals.species = 'salamander'
AND animals.cageid = cages.cageid
AND cages.feedtime >
      (SELECT AVG(feedtime) FROM cages )
```

"nested queries"

Complex Queries 2

Find keepers who keep both students and salamanders:

```
SELECT keeper.name
FROM keeper, cages as c1, cages as c2,
        keeps as k1, keeps as k2, animals as a1, animals as a2
WHERE c1.cageid = k1.cageid AND keeper.keeperid = k1.keeperid
AND c2.cageid = k2.cageid AND keeper.keeperid = k2.keeperid
AND a1.species = 'student' AND a2.species = 'salamander'
AND c1.cageid = a1.cageid AND c2.cageid = a2.cageid
```

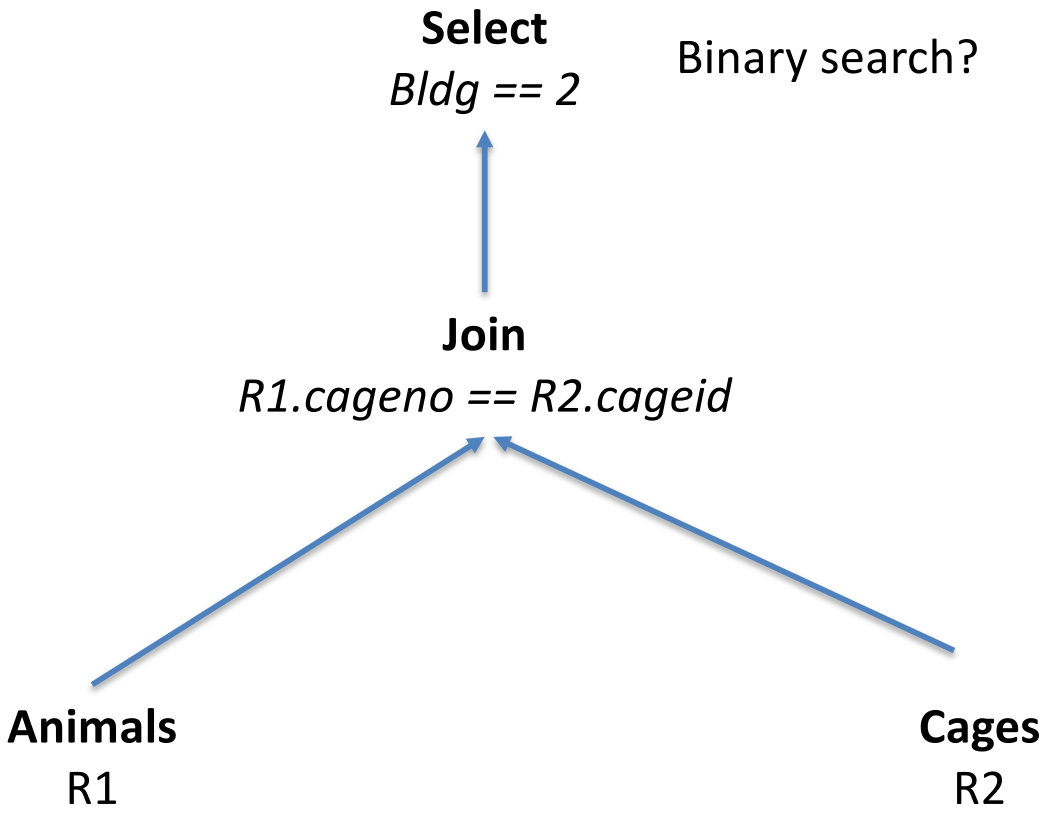


Declarative Queries: What not How!

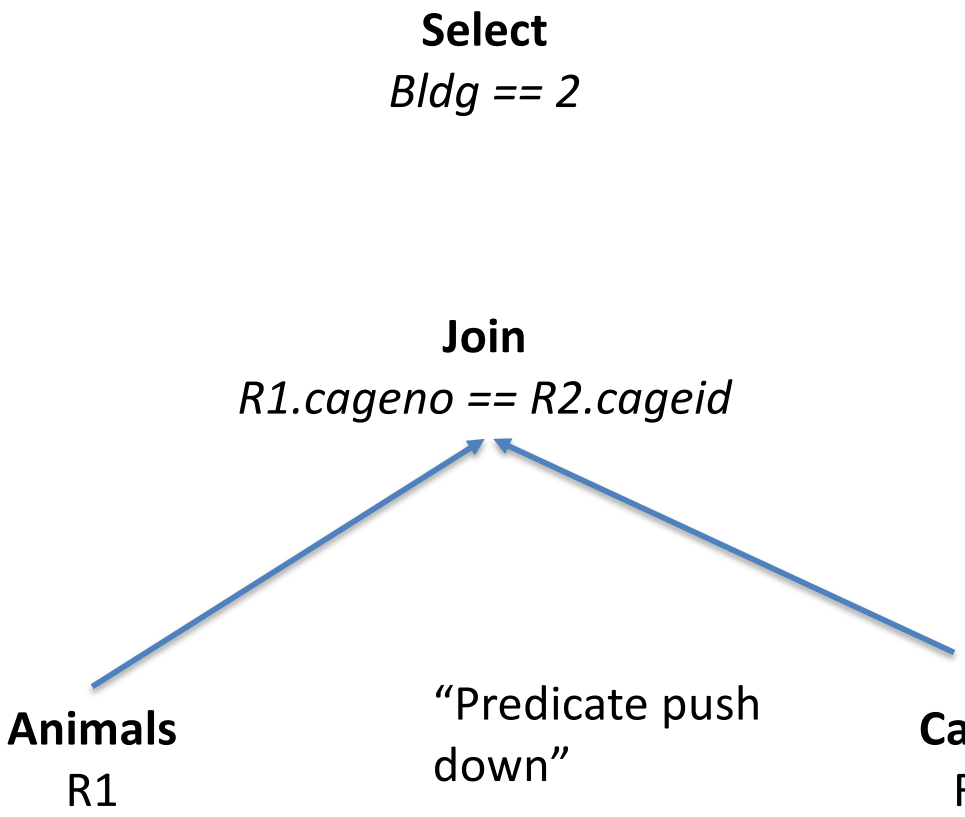
- Many possible procedural plans for a given SQL query

- What else could we do?
  - Sort animals on type
    - + good for “bears” query
    - - Inserts are slower
  - Store animals table in a hash table or tree (“index”)

SQL → Procedural Plan → Optimized Plan → Compiled Program



SQL → Procedural Plan → Optimized Plan → Compiled Program



Programmer just thinks in terms of table operations, not the order or implementation!

Summary: Database Systems

- Relational Model + Schema Design
- Declarative Queries
- Query Optimization
- Efficient access and updates to data
  - Recoverability
  - Consistency