Finding Change in the CouchDB

Dr. James E. Marca

January 13, 2013

Big Data

We’ve always had big data!

- We’ve been handling big data for years in transportation
- What’s wrong with sticking to the old tools?

Old Way:

- PostgreSQL
- Simple queries
- Canned aggregates

"Is the current reading of volume and occupancy from some detector more or less than what should be expected?"

But we had a new server...

- Lots of CPU
- Big Disks
- Raw data already loaded
- in PostgreSQL...

- ☐ The index was too big
- ☐ Extensive disk swapping
- ☐ Super-duper slow!

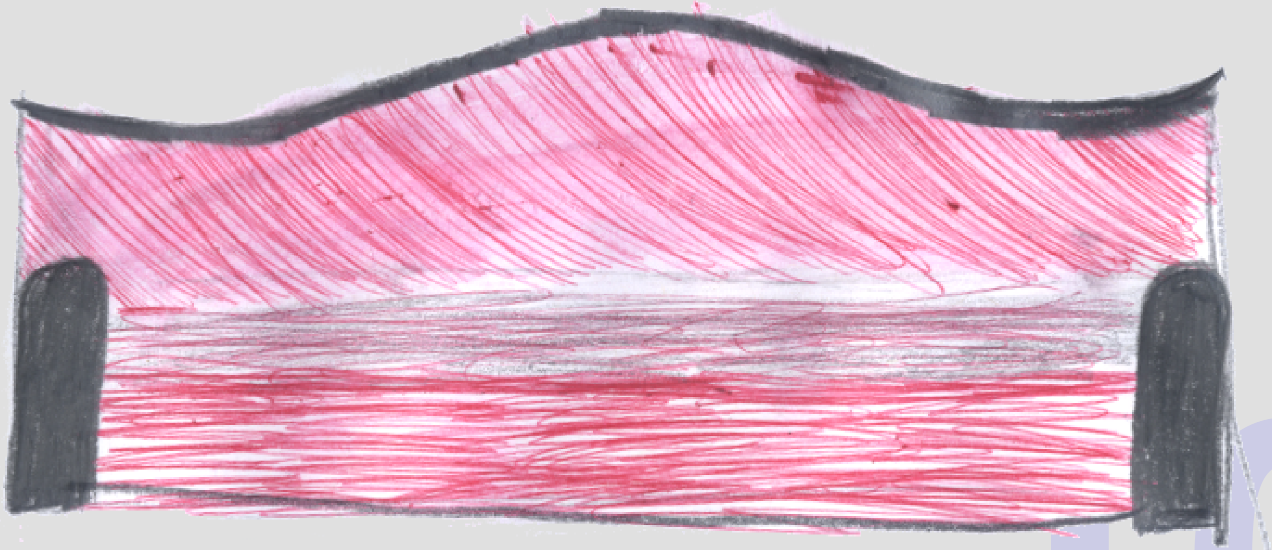
Origins of Big Data and NoSQL

- Google (Big Table)
- Amazon (Dynamo)

Some options available now

- Hadoop
- MongoDB
- Riak
- CouchDB
- BigCouch
- Couchbase
- Cassandra
- Voldemort
- RethinkDB
- Datalog
- and more all the time...

CouchDB is great for transportation data

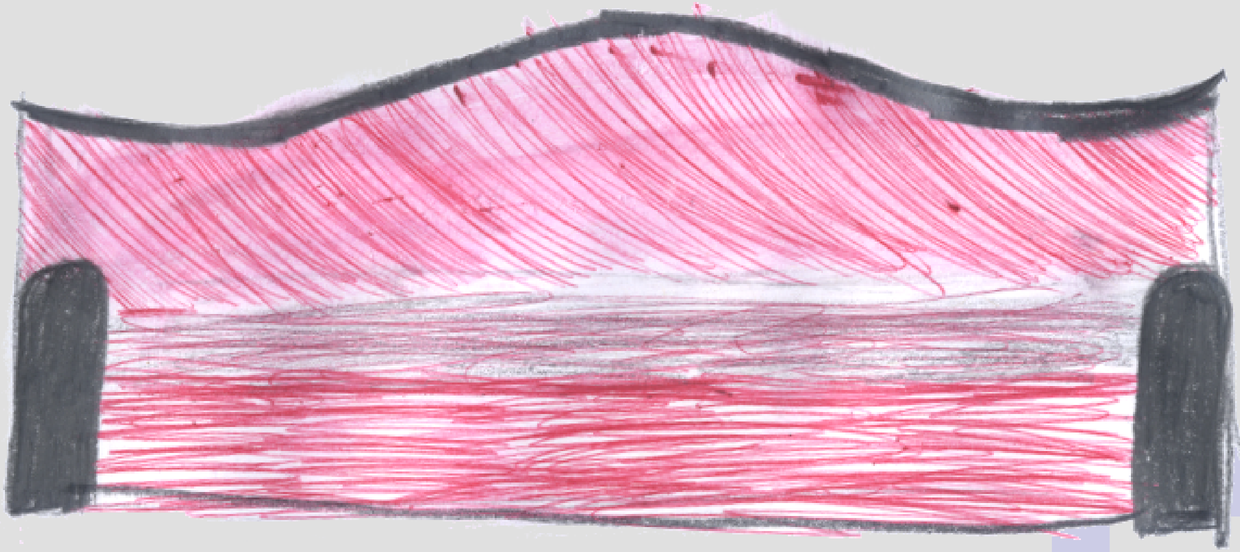


Because transportation data is

- largely observations and measurements
- write once
- read raw data for short term applications
- process raw data into summary stats

Traffic events

What is CouchDB?



CouchDB is a document oriented database

- no schema
- JSON documents
- Views (map and reduce steps) are analogous to queries
- Eventual consistency

The CAP Theorem

The CAP theorem describes a fundamental limitation of databases

- ☐ **Consistency** All database clients see the same data, even with concurrent updates.
- ☐ **Availability** All database clients are able to access some version of the data.
- ☐ **Partition Tolerance** The database can be split over multiple servers.

Choose any two

Traditional RDBMS

- ☐ Consistency
- ☐ Availability

CouchDB

- **A**vailability
- **P**artition Tolerance

  "Eventually Consistent"

Consistency isn't that big a deal for traffic data

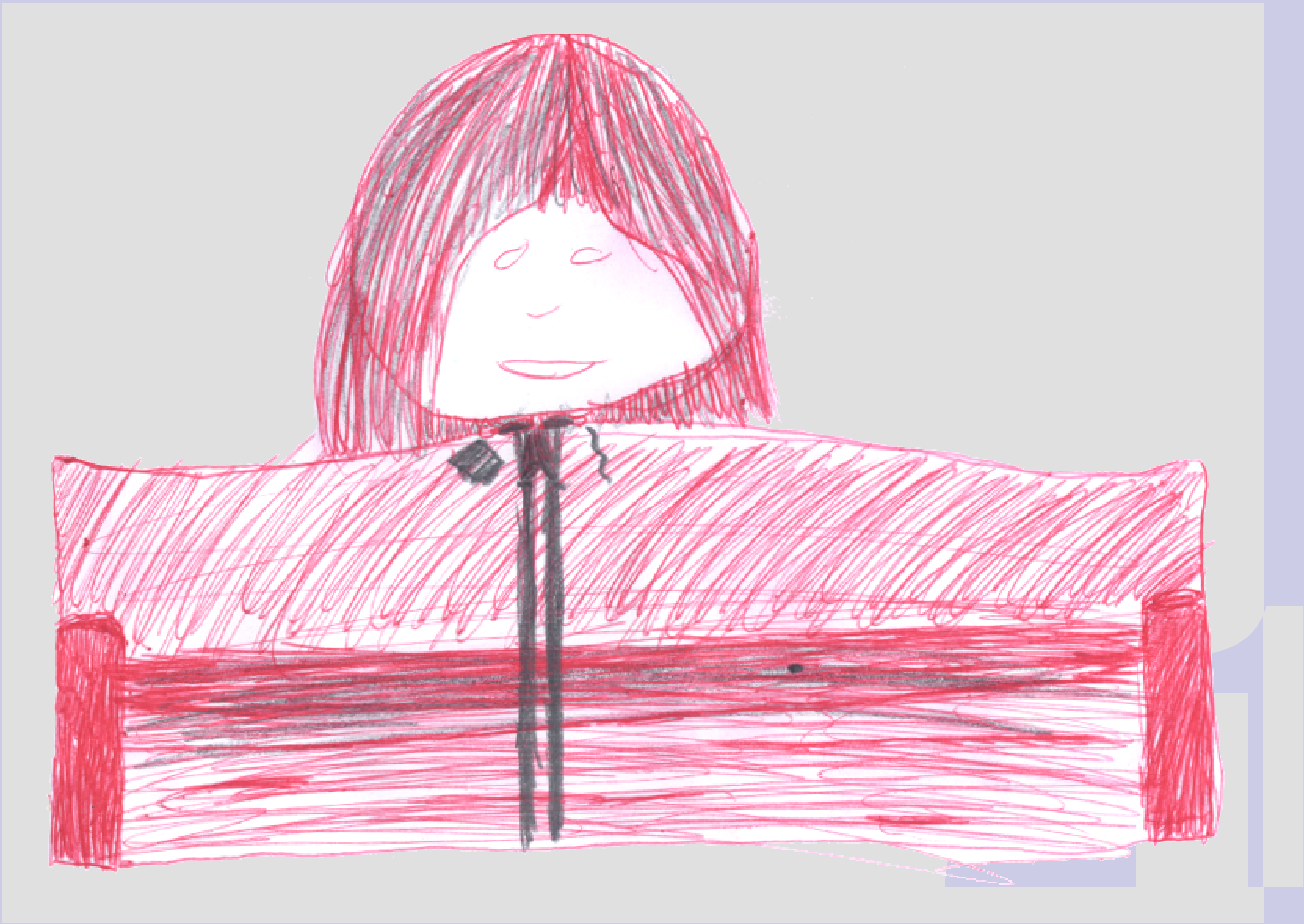
- This isn't stock trading or e-commerce
- It's okay if:
  - you are 30s behind reading a loop
  - Controller A has slightly more current info than Controller B

  A universally consistent view isn't mission critical

Bonus: CouchDB has master-master replication

Replication is super awesome!

Replication is the *change* I was looking for



We can have distributed databases!!

- Put the database *at the detector*
- Move data around by using replication
- A TMC can pull from *all* detector databases
- A Local TMC can *limit replication* to relevant detectors
- A traveler can replicate traffic DBs along common routes

Relax

Practical experience

1. Processing, storing raw loop detector data
2. Imputing missing detector data
3. A single stash for storing detector metadata

1: Processing Raw Loop Detector Data

- Orange County, California (CalTrans District 12)
- about 900 mainline detectors
- Process in R
  - compute 27 different measures per location
  - for 20 minute running time window
  - (vol, occ per lane + 27) per 30s period
  - run models estimating relative risk of accident types
- 280GB of data (three years)
- 590GB of generated views

2: Storing the results of imputation runs

- 400GB of data
- 700GB of generated views
- Databases spread over three machines
- Uses per-district collation databases
- Analysis step used CouchDB to coordinate multiple processes
  - Local “state” database on each machine
  - State databases replicated with each other
  - No overlapping runs were observed

3: Convenient stash for detector metadata

- 20GB of data
- 222MB of generated views
- Uses GeoCouch extensions, stores location of each detector
- Stashes all known metadata about each detector in a single place
- Uses binary attachments to save R analysis output (plots, files)
- Small enough to replicate to my laptop too

Questions?

James E. Marca
UCI ITS
jmarca@translab.its.uci.edu
Research funded by Caltrans, California Air Resources Board

Database design:

- One CouchDB database per detector
- One document per day

Document per day reasons:

- Based on ~~painful~~ ~~experience~~ informal testing
- One document per year is too big to process
- One document per timestamp would work okay,
- But the web *application* uses daily data
- CouchDB sorts by document id, id is based on timestamp
- HTTP GET:
  /vdsdata/d12/2007/1202248/1202248 2007-01-03 00:00:00

Database per detector per year reasons:

- One big database is possible, but
- Impossible to *split* or *shard* over different machines (now can use BigCouch)
- makes better use of multi-core machines when generating views

Use views to run models and summarize data

- Views are CouchDB's version of map/reduce
- Write JavaScript code for the map function that is run on each document (to apply models, run summaries, etc)
- ProTip© Only use embedded Erlang reduce functions like `_count`, `_sum`, and `_stats`

Difficulty: Need to use another database to collate model output

- Pipe summaries of the per-detector views to a single database for all detectors in the district
- Requires external programming
- Difficult to automate