Real-World Datomic: An Experience Report

cognitect

Me

- Craig Andera
- @craigandera
- craig@cognitect.com
- http://cognitect.com/podcast

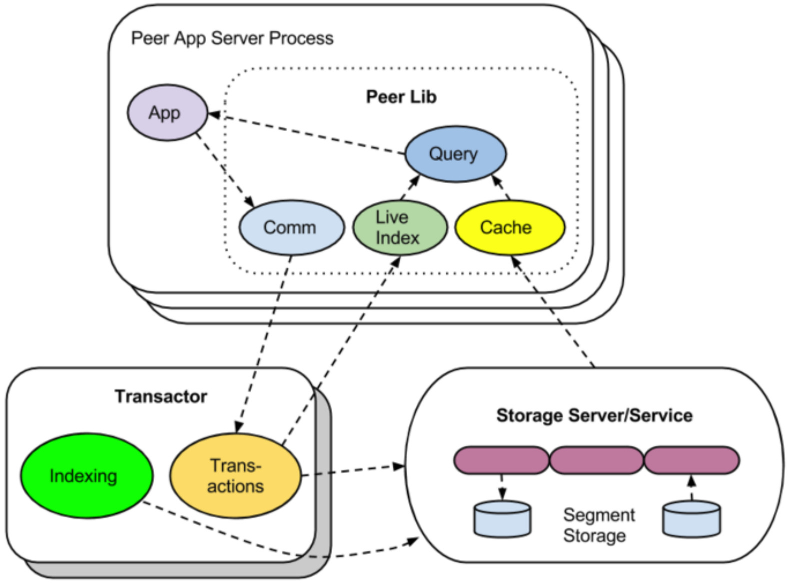
Room Key

- http://roomkey.com
- Employer of Clojure Programmers
- http://is.gd/rkpodcast

The Problem

- Provide a summary of sales funnel events for Room Key
- Users move through the website
  - impression -> hotel details -> lead -> booking
- Want per-day event counts by hotel & event type
- 30 days * ~20 events/sec * ~5 hotels/event
  - => ~250M hotel-events

Datomic's Architecture



Datomic's Data Model

- All facts stored as **`datoms`**
- A datom is EAVT tuple
  - **`Entity`** - The thing we're modeling
  - **`Attribute`** - Which aspect we're describing
  - **`Value`** - The quantity or quality
  - **`Time/Tx`** - When we learned this fact
- *Yesterday, as part of a transaction that happened at 3:02PM, I learned that Craig likes pizza*

Attributes

- Have a type, an ident, and a cardinality
    - Optionally other stuff, e.g. uniqueness
- Types include the usual scalars: strings, numbers, etc.
- Also include **`refs`**
    - Value is an entity ID
    - Allows datoms to form graphs
- Set of defined attributes constitutes database schema

Capturing Events – Schema

Attribute	Type	Qualifiers
`:roomkey.hotel/id`	string	unique
`:roomkey.event/type`	ref	single
`:roomkey.event/hotels`	ref	many
`:roomkey.event/time`	inst	single

Capturing Events - Example

Impression (EID 123)	
`:db/ident`	`:impression`

Hotel details (EID 234)	
`:db/ident`	`:details`

Lead (EID 345)	
`:db/ident`	`:lead`

Booking (EID 456)	
`:db/ident`	`:booking`

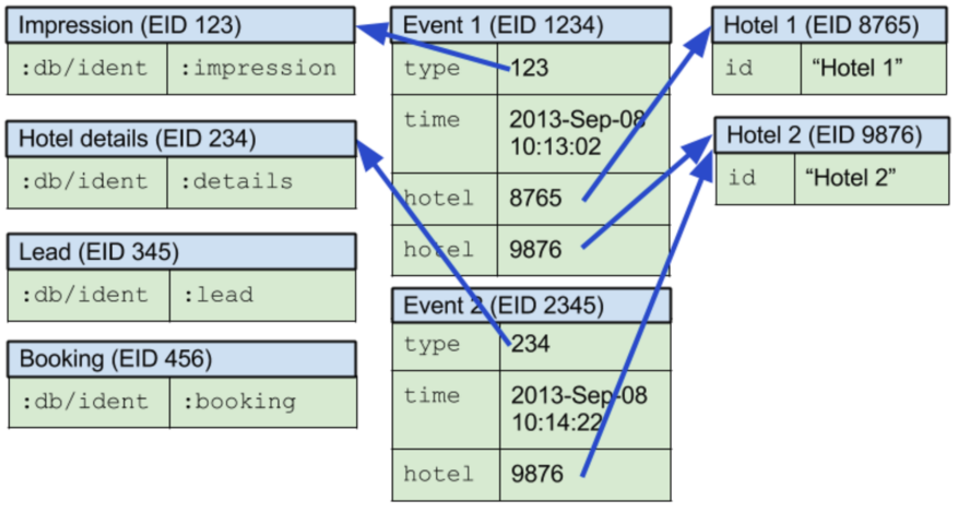
Event 1 (EID 1234)	
`type`	123
`time`	2013-Sep-08 10:13:02
`hotel`	8765
`hotel`	9876

Event 2 (EID 2345)	
`type`	234
`time`	2013-Sep-08 10:14:22
`hotel`	9876

Hotel 1 (EID 8765)	
`id`	“Hotel 1”

Hotel 2 (EID 9876)	
`id`	“Hotel 2”

Capturing Events - Refs



Impression (EID 123)	
:db/ident	:impression

Hotel details (EID 234)	
:db/ident	:details

Lead (EID 345)	
:db/ident	:lead

Booking (EID 456)	
:db/ident	:booking

Event 1 (EID 1234)	
type	123
time	2013-Sep-08 10:13:02
hotel	8765
hotel	9876

Event 2 (EID 2345)	
type	234
time	2013-Sep-08 10:14:22
hotel	9876

Hotel 1 (EID 8765)	
id	"Hotel 1"

Hotel 2 (EID 9876)	
id	"Hotel 2"

Storing and Retrieving Data in Datomic

- One or more EAV tuples submitted to the transactor
- All are given the same **`t`** value
- Datoms written transactionally to the log
- All datoms also stored redundantly in the **`indexes`**

Datomic Indexes

- Every datom stored in two or more **`indexes`**
- Index is (logically) a sorted set of datoms
- Indexes named by their sort order
- EAVT, AEVT, AVET, and VAET

EAVT and AEVT

- All datoms stored in these
- EAVT
  - Efficient access to all attributes of an entity
- AEVT
  - Efficient access to all values of an attribute

Capturing Events - EAVT

E	A	V	T
<Event 1>	type	*<impression>*	1111
<Event 1>	time	10:13:02	1111
<Event 1>	hotel	*<Hotel 1>*	1111
<Event 1>	hotel	*<Hotel 2>*	1111
<Event 2>	type	*<details>*	2222
<Event 2>	time	10:14:22	2222
<Event 2>	hotel	*<Hotel 2>*	2222

Capturing Events - EAVT

E	A	V	T
1234	26	123	1111
1234	27	10:13:02	1111
1234	28	8765	1111
1234	28	9876	1111
2345	26	234	2222
2345	27	10:14:22	2222
2345	28	9876	2222

Capturing Events - AEVT

E	A	V	T
<Event 1>	type	*<impression>*	1111
<Event 2>	type	*<details>*	2222
<Event 1>	time	10:13:02	1111
<Event 2>	time	10:14:22	2222
<Event 2>	hotel	*<Hotel 1>*	1111
<Event 1>	hotel	*<Hotel 2>*	1111
<Event 2>	hotel	*<Hotel 2>*	2222

AVET and VAET

- AVET
  - Only stores datoms with attribute marked **`:db/index`**
  - Efficient lookup of entities by attribute/value pairing
- VAET
  - Only stores datoms with attribute of type **`ref`**
  - Allows efficient navigation of relationships in reverse
  - Also called the **`reverse index`**

Capturing Events - AVET

E	A	V	T
<Hotel 1>	id	"Hotel 1"	888
<Hotel 2>	id	"Hotel 2"	999

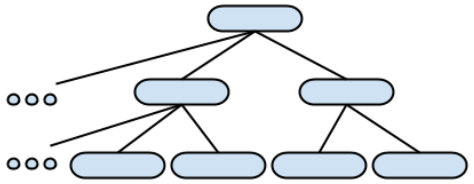
Capturing Events - VAET

E	A	V	T
<Event 1>	type	*<impression>*	1111
<Event 2>	type	*<details>*	2222
<Event 1>	hotel	*<Hotel 1>*	1111
<Event 1>	hotel	*<Hotel 2>*	1111
<Event 2>	hotel	*<Hotel 2>*	2222

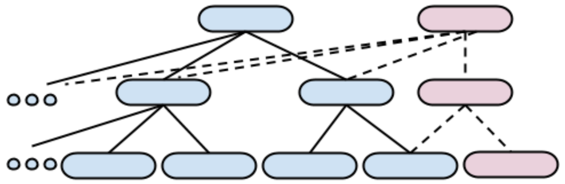
Index Storage and Retrieval

- Datoms are stored in compressed chunks called **`segments`**
- Segments are stored as a tree
- Segments are immutable once written
- Each transaction conceptually creates a new root
- Persistent indexes built less frequently
- Peers pull segments from the appropriate index as needed

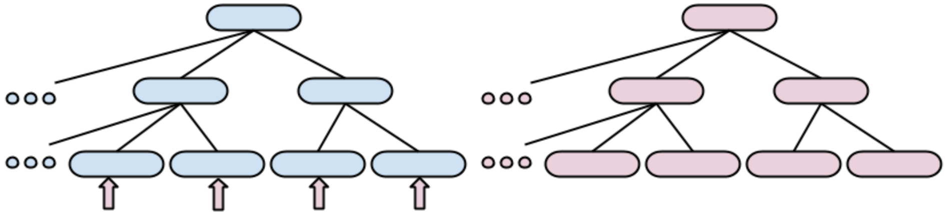
Index Trees



Indexing - Append



Indexing – Highly Nonlocal



Trying to Solve the Problem - Query

```
[:find ?event-type ?hotel-id ?day (count ?event)
 :in $ [?hotel-id ...] ?since
 :where
 [?hotel-eid :roomkey.hotel/id ?hotel-id]
 [?event :roomkey.event/hotel ?hotel-eid]
 [?event :roomkey.event/time ?time]
 [(<= ?since ?time)]
 [(roomkey.time/midnight ?time) ?day]
 [?event :roomkey.event/type ?event-type]]
```

Why This Didn't Work

- Too much data, not enough time: ~250M datoms, 100ms
- No query order eliminates enough data

Anatomy of an Entity ID

- Entity IDs have two parts
  - High bits are the **`partition`** ID
  - Low bits are a timelike component
- Partition ID user-assigned at entity creation

A Change In Approach

- A big, giant cache
- Stop recording time explicitly
- Record events using a dedicated partition
- Use **`seek-datoms`**
    - Returns **`java.lang.Iterable`** over raw index data
    - Fast, lazy, reduceable
- **`entid-at`**
    - Given a time and a partition, get an EID

The Code

```clojure
(defn datoms-between
  "Returns a reducible collection of datoms created
  between the start and end dates in a single partition."
  [db partition start end]
  (let [start-e (d/entid-at db
                            partition
                            start)
        end-e   (d/entid-at db
                            partition
                            end)]
    (->> (d/seek-datoms db :eavt start-e)
         (r/take-while #(< (:e %) end-e)))))
```

Why You Shouldn't Always Do This

- Datalog is a big deal
    - Joins!
    - Query as data
    - Future improvements to query
- Segment caching: hot data often in-memory

Colophon

- Appreciations
    - Room Key
    - Tim Ewald
    - Rich Hickey
    - Bobby Calderwood
    - Stuart Halloway
- Typography
    - Carrois Gothic

Questions?

Thanks!