August 8, 2012

Cassandra at eBay

Time left: **29m 59s**

Buy It Now

Jay Patel

Architect, Platform Systems

@pateljay3001

eBay Marketplaces

- 97 million active buyers and sellers
- 200+ million items
- 2 billion page views each day
- 80 billion database calls each day
- 5+ petabytes of site storage capacity
- 80+ petabytes of analytics storage capacity

How do we scale databases?

- Shard
  - Patterns: Modulus, lookup-based, range, etc.
  - Application sees only logical shard/database
- Replicate
  - Disaster recovery, read availability/scalability
- Big NOs
  - No transactions
  - No joins
  - No referential integrity constraints

We like Cassandra

- Multi-datacenter (active-active)
- Availability - No SPOF
- Scalability
- Write performance
- Distributed counters
- Hadoop support

We also utilize MongoDB & HBase

Are we replacing RDBMS with NoSQL?

Not at all! But, complementing.

- Some use cases don’t fit well - sparse data, big data, schema optional, real-time analytics, …
- Many use cases don’t need top-tier set-ups - logging, tracking, …

A glimpse on our Cassandra deployment

- Dozens of nodes across multiple clusters
- 200 TB+ storage provisioned
- 400M+ writes & 100M+ reads per day, and growing
- QA, LnP, and multiple Production clusters

Use Cases on Cassandra

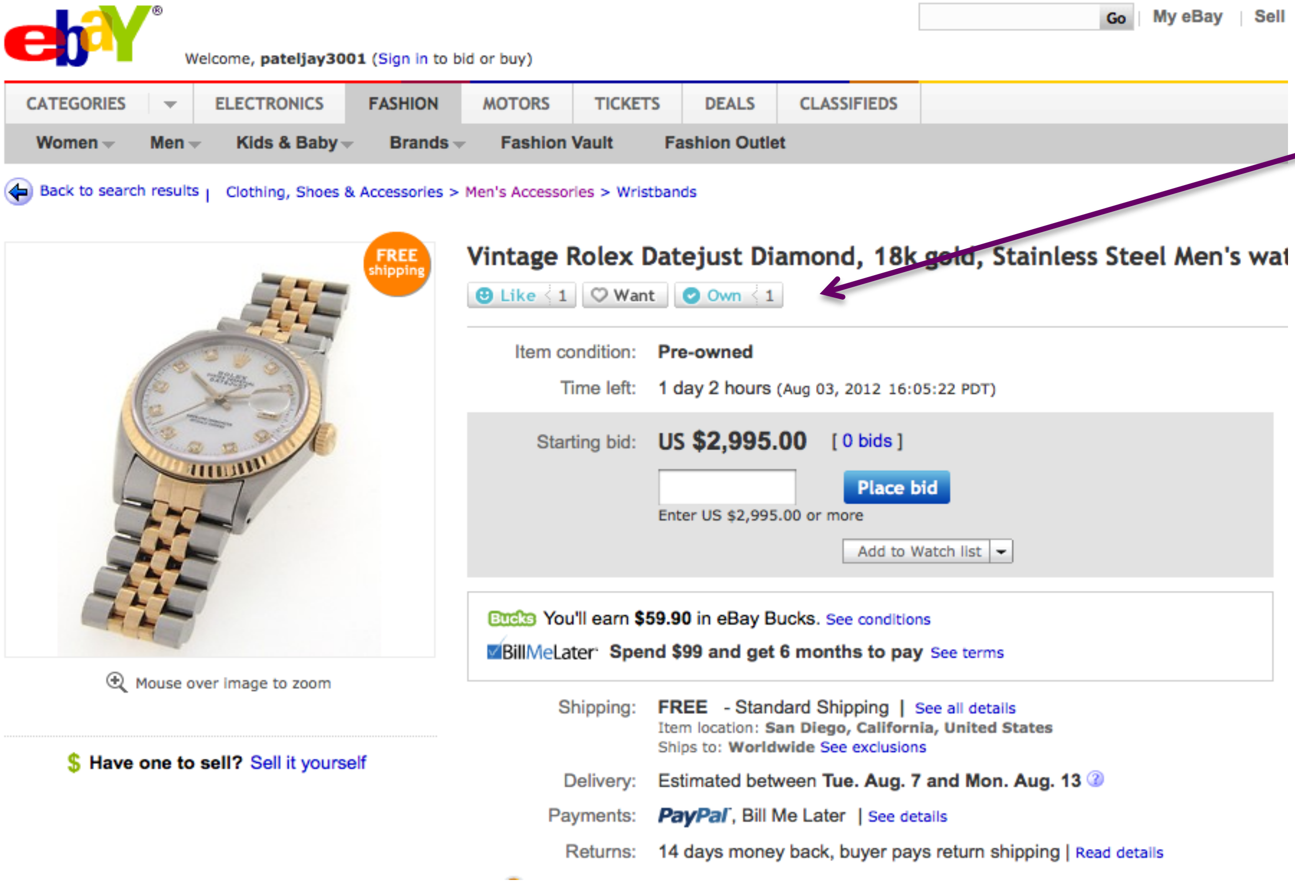
# 1: Social Signals on eBay product & item pages

# 2: Hunch taste graph for eBay users & items

# 3: Time series use cases (many):

- Mobile notification logging and tracking
- Tracking for fraud detection
- SOA request/response payload logging
- RedLaser server logs and analytics

USE CASE #1: SOCIAL SIGNALS



Welcome, **pateljay3001** (Sign in to bid or buy)

Back to search results | Clothing, Shoes & Accessories > Men's Accessories > Wristbands

Vintage Rolex Datejust Diamond, 18k gold, Stainless Steel Men's wat

Item condition: **Pre-owned**

Time left: 1 day 2 hours (Aug 03, 2012 16:05:22 PDT)

Starting bid: **US $2,995.00** [0 bids]

Enter US $2,995.00 or more

Bucks You'll earn **$59.90** in eBay Bucks. See conditions

BillMeLater **Spend $99 and get 6 months to pay** See terms

Shipping: **FREE** - Standard Shipping | See all details

Item location: **San Diego, California, United States**

Ships to: **Worldwide** See exclusions

Delivery: Estimated between **Tue. Aug. 7 and Mon. Aug. 13**

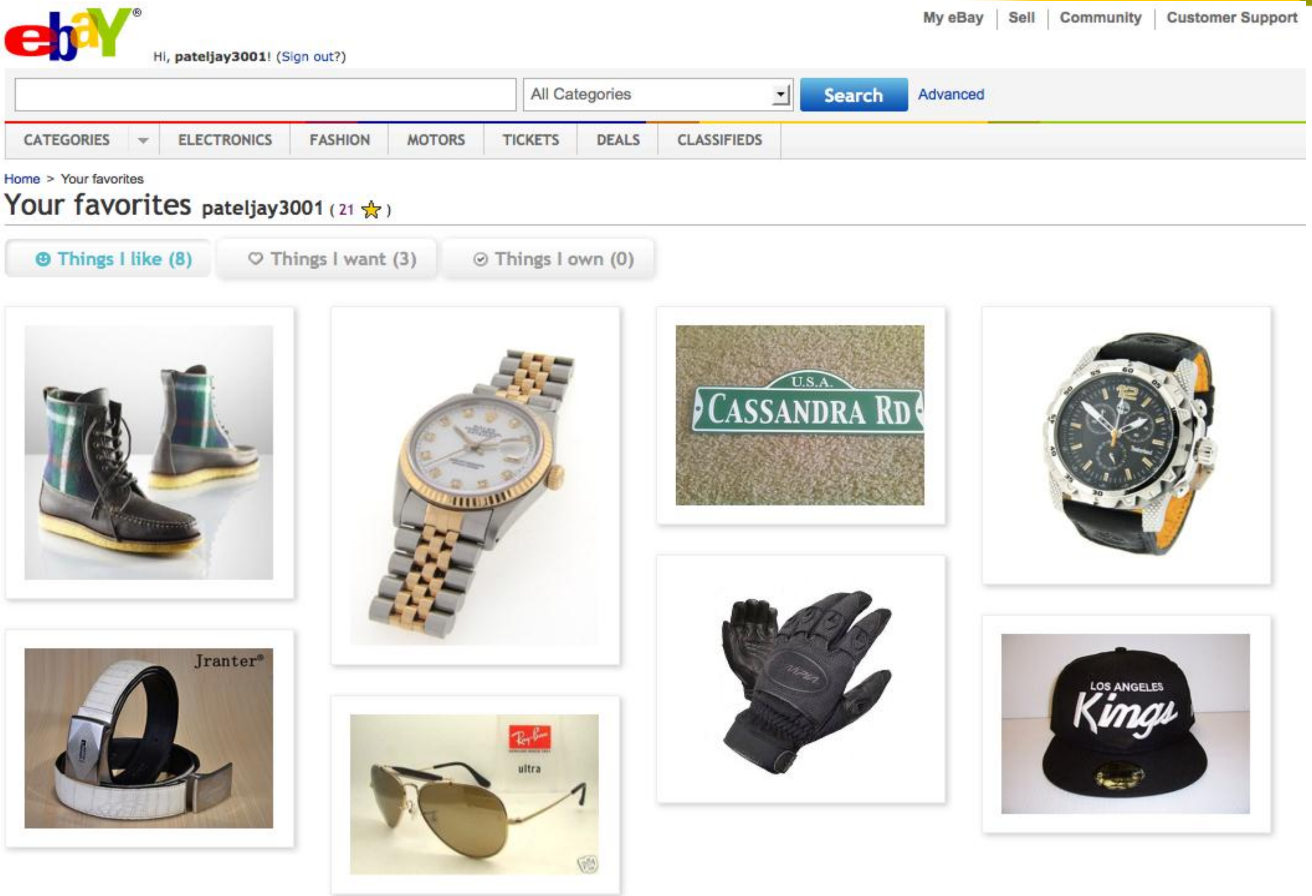
Payments: PayPal, Bill Me Later | See details

Returns: 14 days money back, buyer pays return shipping | Read details

$ Have one to sell? Sell it yourself

Served by Cassandra

Manage signals via "Your Favorites"

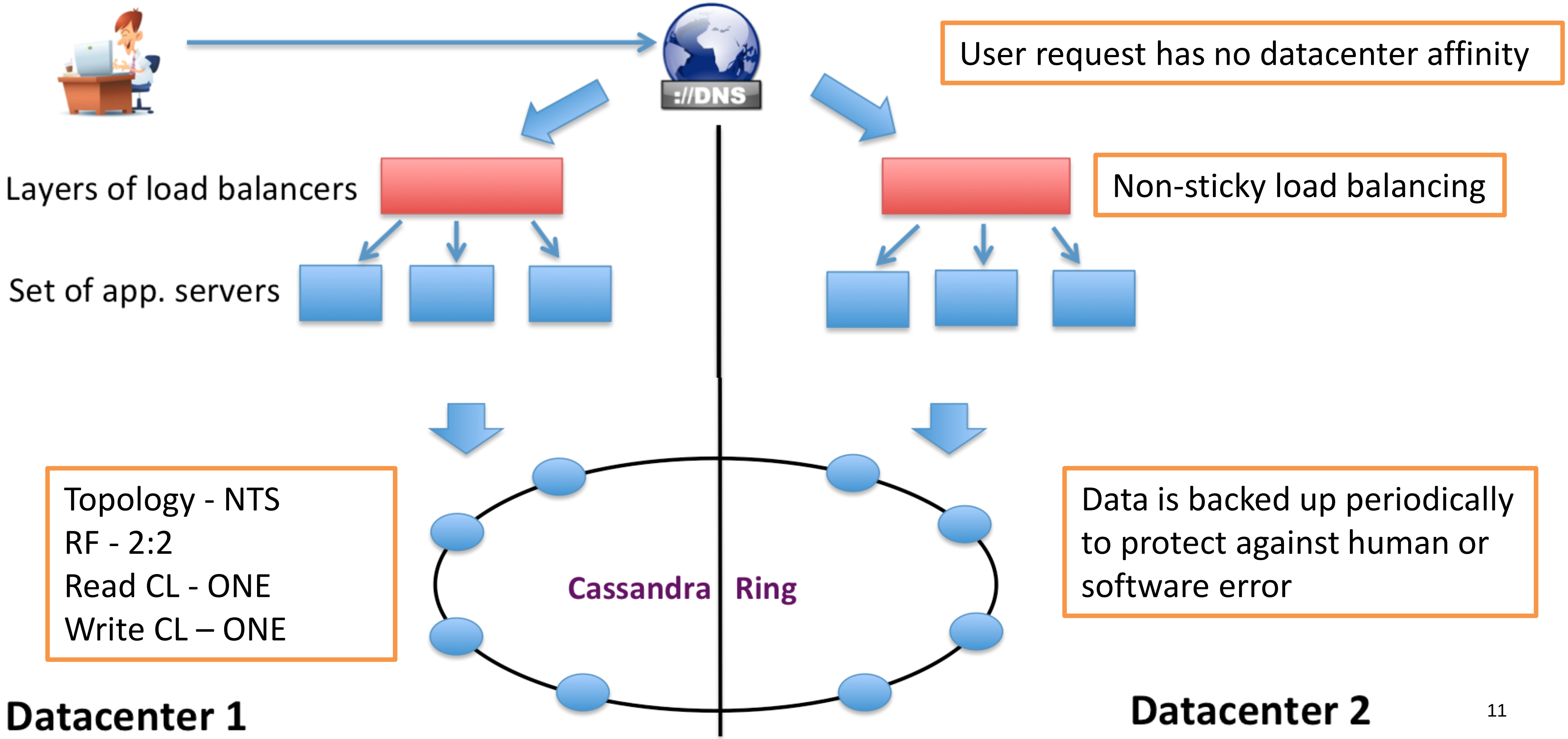


Whole page is served by Cassandra

Why Cassandra for Social Signals?

- Need scalable counters
- Need real (or near) time analytics on collected social data
- Need good write performance
- Reads are not latency sensitive

Deployment



User request has no datacenter affinity

Non-sticky load balancing

Topology - NTS
RF - 2:2
Read CL - ONE
Write CL – ONE

Data is backed up periodically to protect against human or software error

Data Model

depends on query patterns

Data Model (simplified)

ItemCount

itemid1	“likeCount”	“wantCount”	“ownCount”
	2000	5000	1000
⋮			

- Get signal count for a item

UserCount

userid1	“likeCount”	“wantCount”	“ownCount”
	20	100	50
⋮			

- Get signal count for a user

ItemLike

itemid1	userid1	userid2	userid3	…
	timeuuid1	timeuuid2	timeuuid3	
⋮				

- Check if user has already liked a item or not.

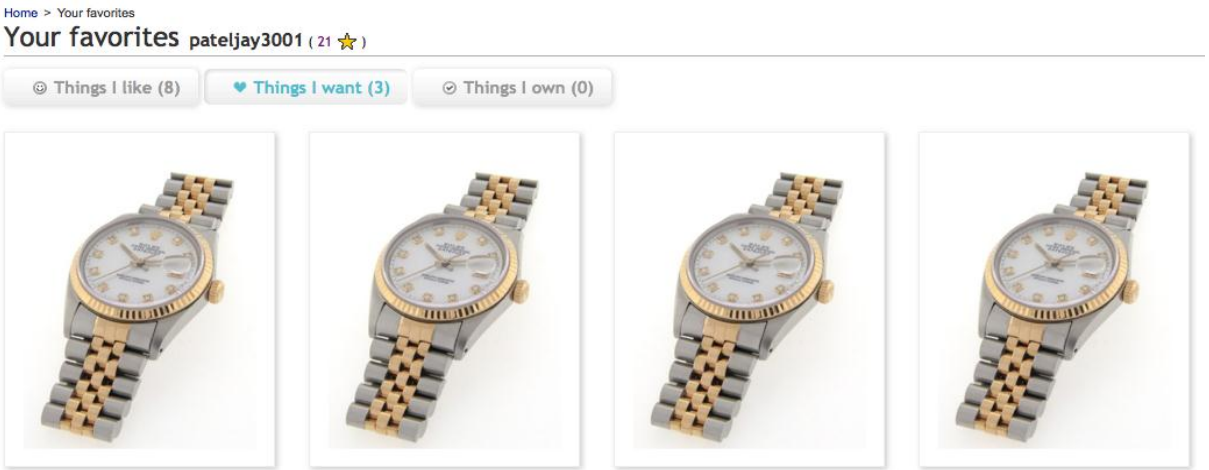
UserLike

Composite column name/key

userid1	timeuuid1 | itemid1	timeuuid2 | itemid2	…
	-null-	-null-	
⋮			

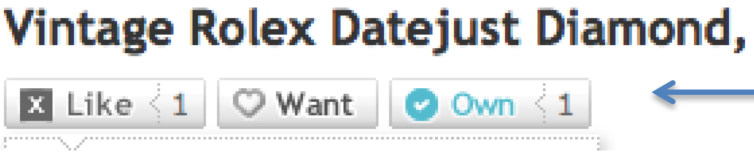
- Get user’s liked items in chronological order.

Wait…



Your favorites pateljay3001 (21 ★)

Duplicates!


Vintage Rolex Datejust Diamond,

Oh, toggle button!

Signal --> De-signal --> Signal…

Yes, eventual consistency!

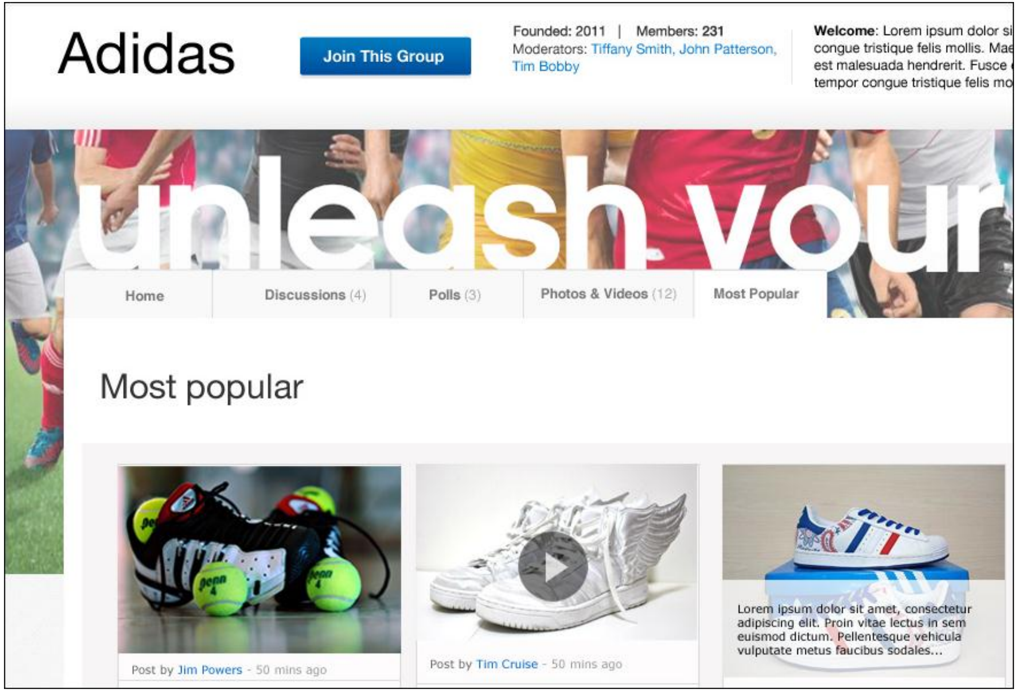
One scenario that produces duplicate signals in UserLike CF:

1. Signal
2. De-signal (1st operation is not propagated to all replica)
3. Signal, again (1st operation is not propagated yet!)

So, what's the solution? Later…

Social Signals, next phase: Real-time Analytics

- Most signaled or popular items per affinity groups (category, etc.)
- Aggregated item count per affinity group



Adidas

Founded: 2011 | Members: 231
Moderators: Tiffany Smith, John Patterson, Tim Bobby

Welcome: Lorem ipsum dolor si
congue tristique felis mollis. Mae
est malesuada hendrerit. Fusce
tempor congue tristique felis mo

Most popular

Post by Jim Powers - 50 mins ago

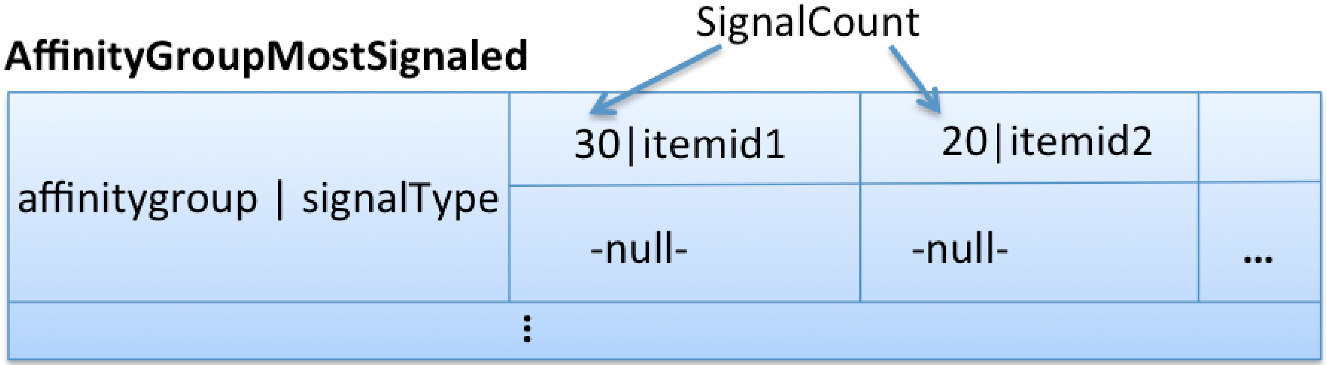
Post by Tim Cruise - 50 mins ago

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin vitae lectus in sem euismod dictum. Pellentesque vehicula vulputate metus faucibus sodales...

Example affinity group

Initial Data Model for real-time analytics

AffinityGroupMostSignaled

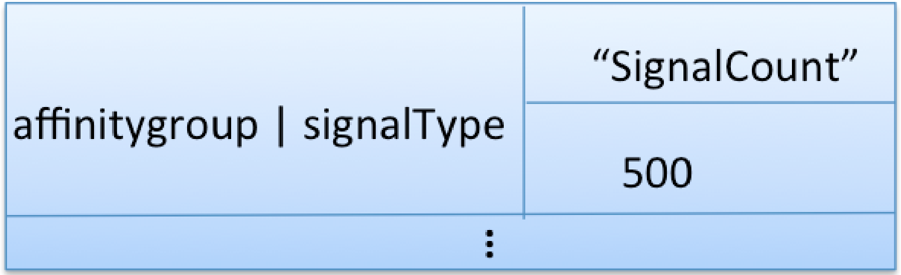


SignalCount

affinitygroup | signalType	30|itemid1	20|itemid2	
	-null-	-null-	...
⋮			

Items in an affinitygroup is physically stored sorted by their signal count

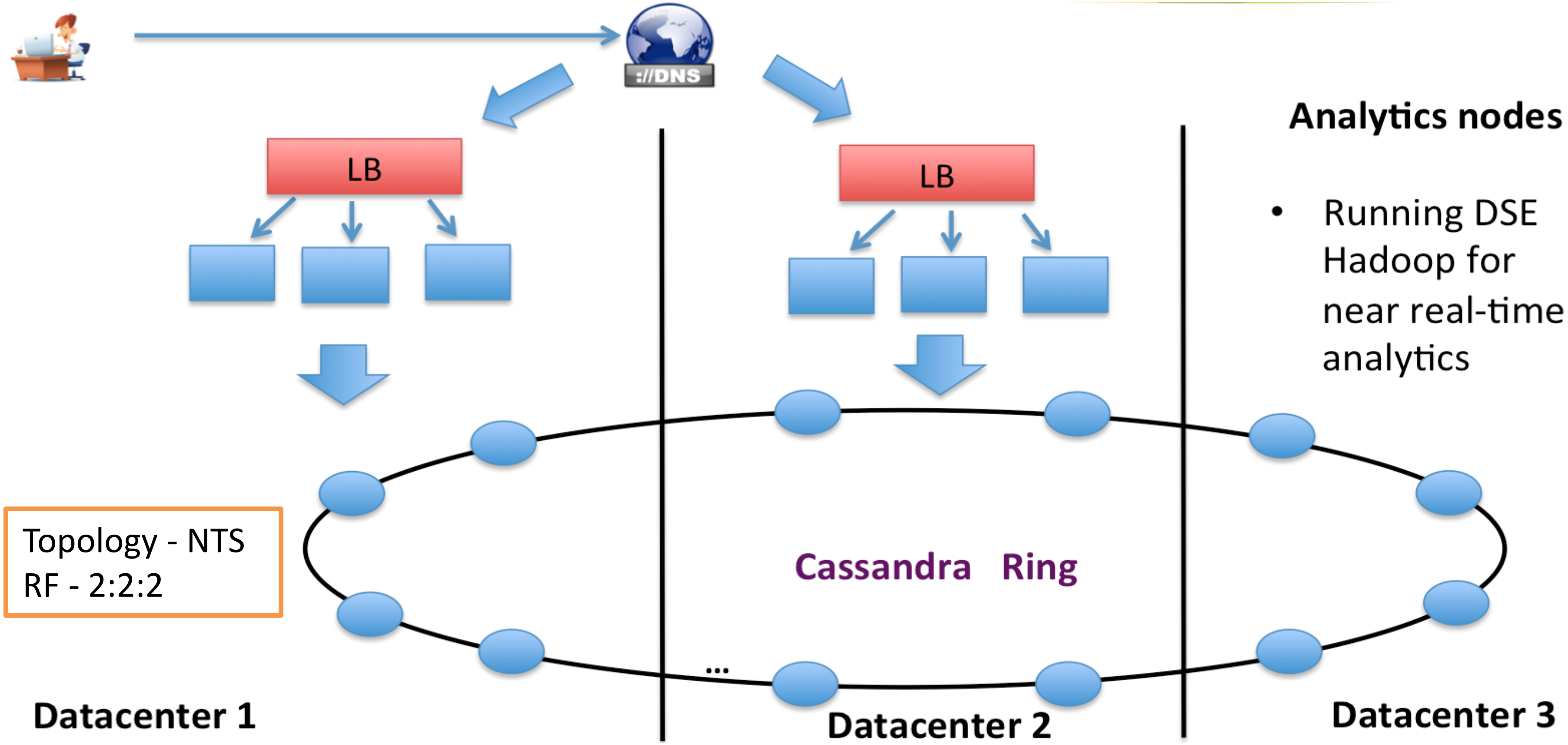
AffinityGroupCounter



affinitygroup | signalType	“SignalCount”
	500
⋮	

Update counters for both individual item and all the affinity groups that item belongs to

Deployment, next phase

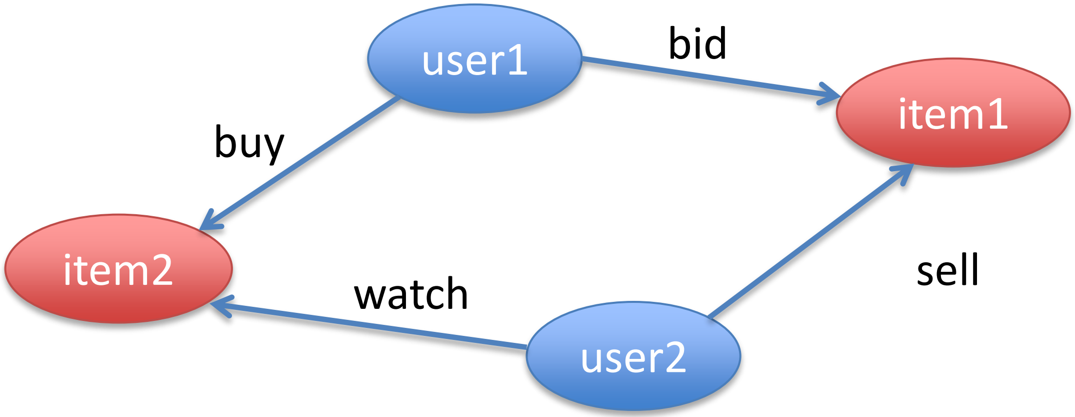


Analytics nodes

- Running DSE Hadoop for near real-time analytics

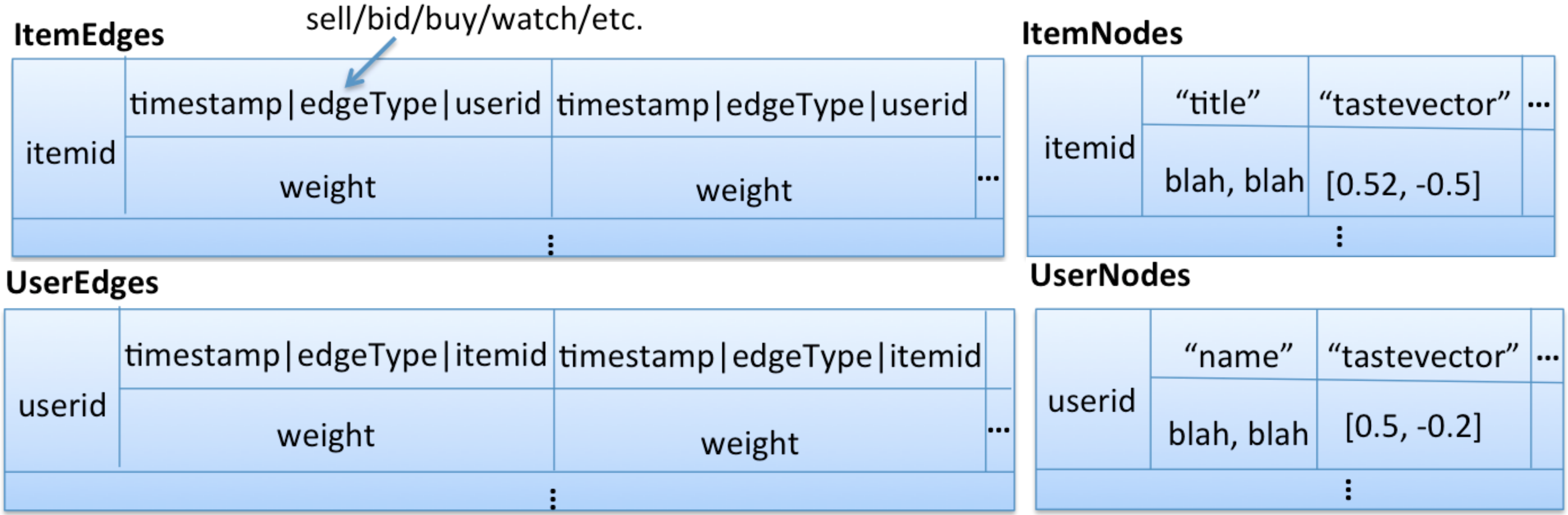
Topology - NTS
RF - 2:2:2

USE CASE #2: TASTE GRAPH FOR EBAY USERS & ITEMS



Graph in Cassandra

Event consumers listen for site events (sell/bid/buy/watch) & populate graph in Cassandra



ItemEdges

sell/bid/buy/watch/etc.

itemid

timestamp|edgeType|userid — weight

timestamp|edgeType|userid — weight

…

⋮

ItemNodes

itemid

“title” — blah, blah

“tastevector” — [0.52, -0.5]

…

⋮

UserEdges

userid

timestamp|edgeType|itemid — weight

timestamp|edgeType|itemid — weight

…

⋮

UserNodes

userid

“name” — blah, blah

“tastevector” — [0.5, -0.2]

…

⋮

- 30 million+ writes daily
- 14 billion+ edges already

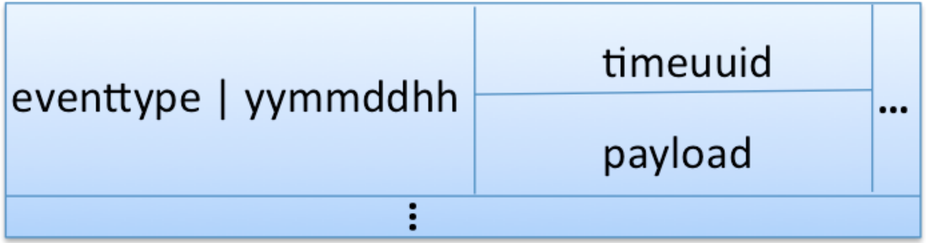
- Batch-oriented reads (for taste vector updates)

USE CASE #3: TIME SERIES DATA

- Mobile notification logging and tracking
- Tracking for fraud detection
- SOA request/response payload logging
- RedLaser server logs and analytics

A glimpse on Data Model

Events



eventtype | yymmddhh

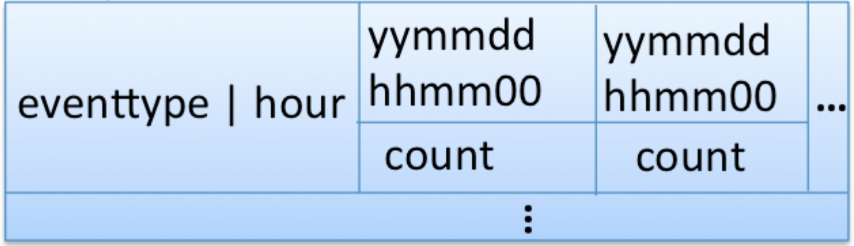
timeuuid

payload

…

⋮

Rollups-minute



eventtype | hour

yymmdd hhmm00

count

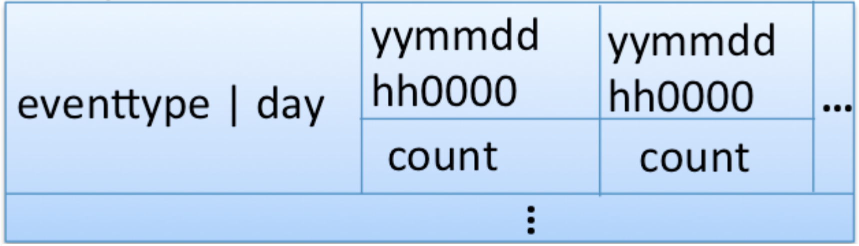
yymmdd hhmm00

count

…

⋮

Rollups-hour



eventtype | day

yymmdd hh0000

count

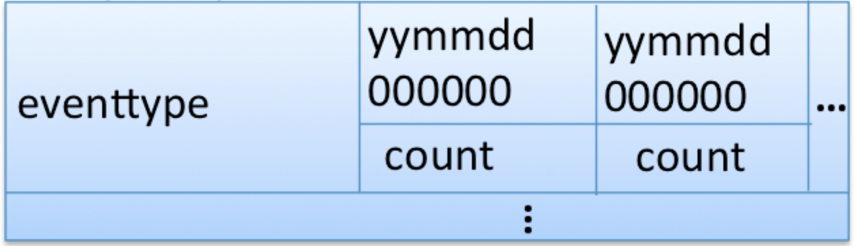
yymmdd hh0000

count

…

⋮

Rollups-day



eventtype

yymmdd 000000

count

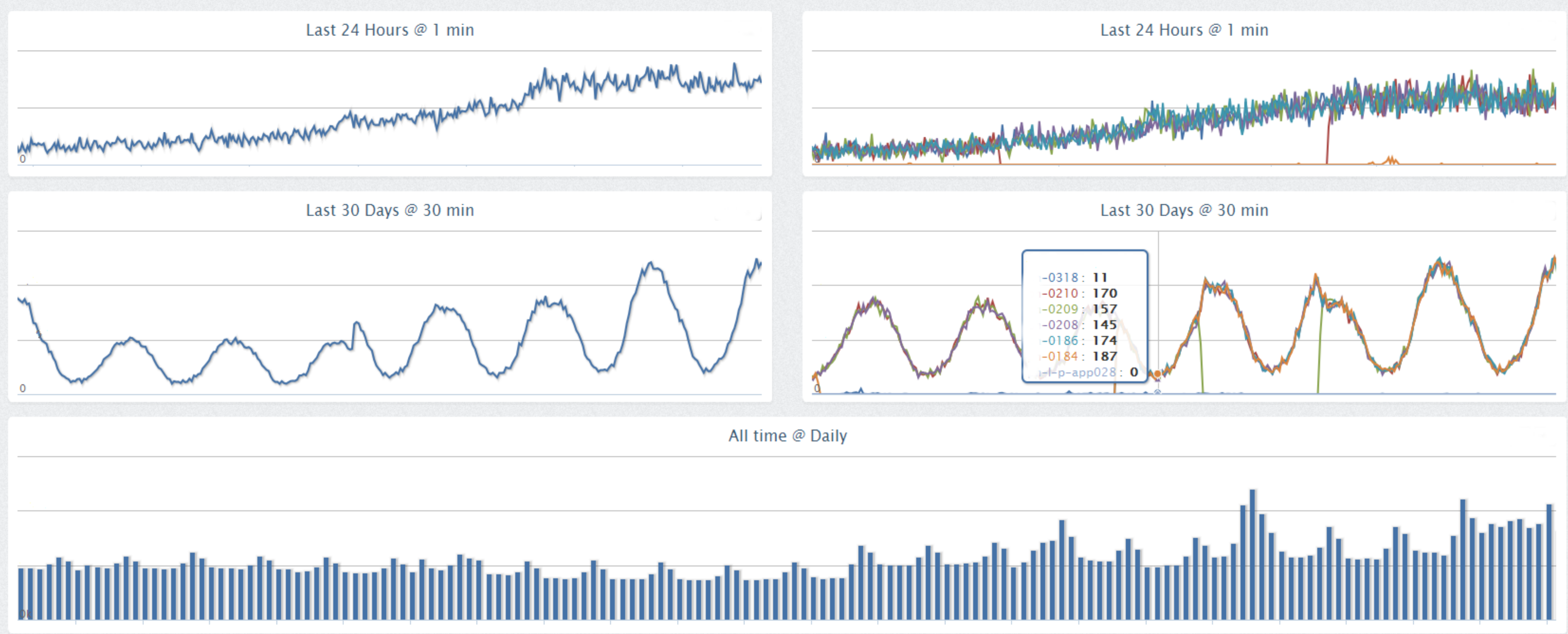
yymmdd 000000

count

…

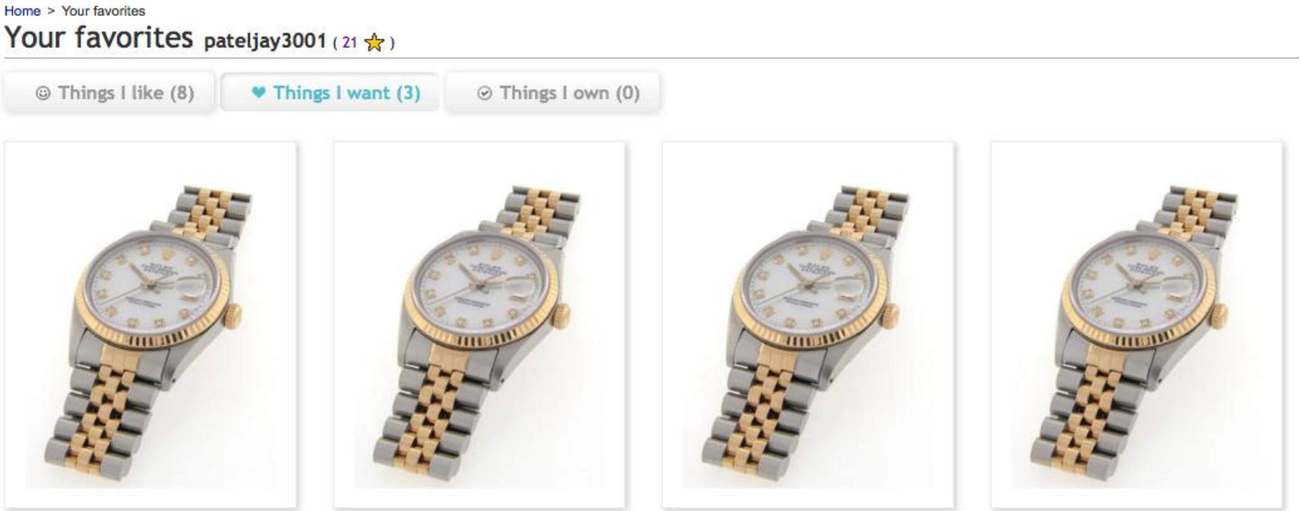
⋮

RedLaser tracking & monitoring console



That’s all about the use cases..

Remember the duplicate problem in Use Case #1?



Home > Your favorites

Your favorites pateljay3001 (21 ☆)

☺ Things I like (8) | ♥ Things I want (3) | ⊘ Things I own (0)

Let’s see some options we considered to solve this…

Option 1 – Make ‘Like’ idempotent for UserLike

- Remove time (timeuuid) from the composite column name:
  - Multiple signal operations are now Idempotent
  - No need to read before de-signaling (deleting)

UserLike *Old*

userid1	timeuuid1 | itemid1	timeuuid2 | itemid2	…
	-null-	-null-	
⋮			

UserLike *New*

userid1	itemid1	itemid2	…
	-null-	-null-	
⋮			

X Need timeuuid for ordering!

Already have a user with more than 1300 signals

Option 2 – Use strong consistency

- Local Quorum
  - Won't help us. User requests are not geo-load balanced (no DC affinity).
- Quorum
  - Won't survive during partition between DCs (or, one of the DC is down). Also, adds additional latency.

X Need to survive!

Option 3 – Adapt to eventual consistency

If desire survival!

http://www.strangecosmos.com/content/item/101254.html

Adjustments to eventual consistency

De-signal steps:

- Don’t check whether item is already signaled by a user, or not
- Read all (duplicate) signals from UserLike_unordered (new CF to avoid reading whole row from UserLike)
- Delete those signals from UserLike_unordered and UserLike

UserLike

<table>
<tr><td rowspan="2">userid1</td><td>timeuuid1 | itemid1</td><td>timeuuid2 | itemid1</td><td rowspan="2">...</td></tr>
<tr><td>-null-</td><td>-null-</td></tr>
<tr><td colspan="4">⋮</td></tr>
</table>

UserLike_unordered

<table>
<tr><td rowspan="2">userid1</td><td>Itemid1|timeuuid1</td><td>Itemid1|timeuuid2</td><td rowspan="2">...</td></tr>
<tr><td>-null-</td><td>-null-</td></tr>
<tr><td colspan="4">⋮</td></tr>
</table>

Still, can get duplicate signals or false positives as there is a ‘read before delete’.

To shield further, do ‘repair on read’.

Not a full story!

Lessons & Best Practices

- Choose proper Replication Factor and Consistency Level.
  - They alter latency, availability, durability, consistency and cost.
  - Cassandra supports tunable consistency, but remember strong consistency is not free.
- Consider all overheads in capacity planning.
  - Replicas, compaction, secondary indexes, etc.
- De-normalize and duplicate for read performance.
  - But don’t de-normalize if you don’t need to.
- Many ways to model data in Cassandra.
  - The best way depends on your use case and query patterns.

More on http://ebaytechblog.com?p=1308

Thank You

@pateljay3001

#cassandra12