Large Data Methods:
Keeping it Simple on the Path to Big Data

Big Data Business Forum,

San Francisco, November 13, 2012

Jim Porzak, Sr. Dir. Business Intelligence, Minted

What we will cover:

1. Who is Minted?
2. Our large data challenges.
3. Large data solutions.
4. Migration to “big data.”
5. Discussion

About Minted

- A social commerce site.
- Crowd-sourcing graphic designs and art from a global design community.
- Selling those as printed paper products.
- Initially focused on the $10 billion stationery and $48 billion art markets.
- Combining community with commerce.
- Built on stellar technology, operations, and customer service.

Minted.com Architecture

~ Classic LAMP

Integrated back office: “MBO”

MySQL holds site & MBO data

Minted BI Architecture

On Amazon EC2:

- Replicated MySQL site DB
- PostgreSQL BI DB
- Tableau Server

In support of marketing:

Our job is to understand…

the customer,

the whole customer, and

nothing but the customer.

CIA

Customer

Initiated

Actions

CIA's are:

- Ordering
- Other Minted.com actions
- Responding to Minted outreach
  - Email opens, reads, clicks
- Contacting Minted
- Social Behavior
- And more!

Ordering:
“How do you sell bread?”

1. “You tell me what you want.”
2. “I give you the bread.”
3. “I tell you how much it costs.”
4. “You give me the money.”


minted.
Refer a Friend | 1-888-828-6468 | Help | Feedback | My Saved Designs | Jim
SHOP | VOTE | EYE CANDY | DESIGN | BLOG
Search Minted
Holiday | Gifts | Weddings | Baby & Kids | Paperie | NEW | Art Prints | Party Decor
Ends Thursday! 10% off + free ground shipping on all holiday cards. Use code: CHEERS.
Home > Holiday Photo Cards > Merry Modern
merry modern holiday photo cards
DESIGN CHALLENGE WINNER
by sarah curry -
santa cruz, ca
happy holidays FROM THE ASHBYS
available in other designs too!
SHAPES
COLORS
Special shapes not available for folded cards or TripleThick™ paper.
MORE VIEWS
Front
Interior Options
Backer Options
(Flat cards only)
Liner and Label
Assembled
Quantity and Options:
1.63 ea. 100 @ $163
Choose Format:
about our unique formats
Flat Card Photo and other backer options + $0
Folded Card Blank or with photo + text inside + $45
Folded Card Story or Yearline™ interior + $59
Choose Paper:
about our paper
Signature Paper A thick and luxurious... more + $0
Premium 100% Recycled Paper Very thick paper... more + $17
Pearlescent Paper Adds a subtle shimmer... more + $33
TripleThick™ Paper - NEW! Our thickest paper... more + $137
Subtotal for 100 Holiday Photo Cards 1.63 ea. $163.00
1 free digital proof per item ordered.
personalize
Not ready yet?
mark as favorite
add sample to cart
email to a friend
Customer Reviews:
0.0
Be the first to Write a Review

Other Site CIA's

1. Google Analytics
   - Not for individual visitor!
   - f(tagging(t))
2. "App-request" logs
   - CIA's & some 2nd level
   - Visitor (cookie) & user ID's
   - 12 months of history

Responses to email:

1. Sends
   - Transactional: order ack, ship, …
   - Marketing: retention, offers
     - Targeted & personalized
2. CIA's
   - Bounce, open, click, buy
   - Opt-outs, opt-ins

Other data sources:

- Convertro
- Survey Tools
- Demographic Appends

Roadmap to “Big Data”

- Large Data:
  - PostgreSQL
- Big Data:
  - Some columnar DB
- Bigger Data:
  - Some map-reduce platform

Why PostgreSQL?

- Open source (~free)
- SQL for analytics
  - Window functions, etc.
- Known to scale
- Foundation for many of the columnar DB products

BI in PostgreSQL philosophy

- Data structures as if columnar
  - Big & wide; not star
- Focus on high impact first:
  - Orders
  - Order Detail
  - Customers
  - Site Sessions

Primary wide tables

Features: Demographics, order & site behavior rollups, derived segments. Used for targeting.

Features: $ & # totals this order & to date, SKU's & product category, order sequence #, prior order, first order,

Features: all the product options for this order line/SKU - merchandising details.

Features: 30 minute inactivity rule rollup, session sequence #, basic counts & duration, sales funnel flags, SKU & site area visit strings

Features: spiders, bots, & load testing events removed. Parsed to basic CIA parameters. Sequenced by visitor_id.

Table Details

- Customer: 35 columns
  - ID, source, acquisition date/source, purchase profiles, site profiles, demo profiles
- Order Summary: 72 columns
  - ID’s, date, order seq #, gap, $’s, #’s, flags, top, prior, first product code/group/class, to-date $’s by class, geo, source.
- Order Product Detail: 27 columns
  - ID’s, timestamp, SKU, $, #, promo, details of options
- Site Sessions: 27 columns
  - ID’s, seq #, # events, duration, timestamps, gap, funnel flags, products, actions, sources, media, campaigns, …
- Site Clean Events: 21 columns
  - ID’s, timestamp, ip, seq #, interval to prior, entry/exit actions, source/medium/campaign, sku, …

Performance

- Queries off of these tables very fast; typically sub-minute for even the most complex.
- Tableau server has internal columnar engine for interactive analytics performance.
- Nightly refresh & updates in under three hours with no attempt at tuning.

Next Steps:

- Finalize logical design in PostgreSQL based on needs of our business partners over next few months.
- Tune PostgreSQL platform test limits of scale. Estimate when we will need to move to next level. In meantime:
  - POC's on a couple of columnar DBs.
- Migrate to final columnar DB.

Questions? Comments?

Now would be the time!

APPENDIX

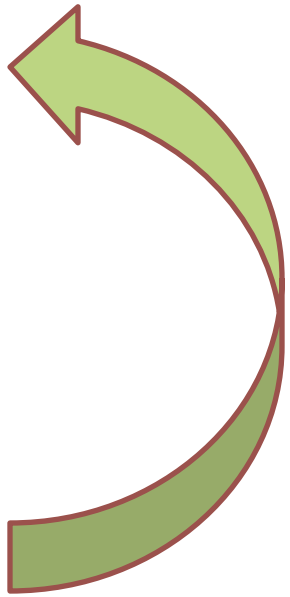
Application request logs – deep dive

What’s an app-request log?

{"time_start":1313620339.85,"time_end":1313620340.01,"request":{"headers":[["Host","localhost:8888"],["Connection","keep-alive"],["Cache-Control","max-age=0"],["User-Agent","Mozilla\/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit\/534.30 (KHTML, like Gecko) Chrome\/12.0.742.112 Safari\/534.30"],["Accept","text\/html,application\/xhtml+xml,application\/xml;q=0.9,*\/*;q=0.8"],["Accept-Encoding","gzip,deflate,sdch"],["Accept-Language","en-US,en;q=0.8"],["Accept-Charset","ISO-8859-1,utf-8;q=0.7,*;q=0.3"]],"method":"GET","remote_addr":"127.0.0.1","protocol":"HTTP\/1.1","uri":"\/register"},"response":{"status":null,"headers":[["Expires","Sun, 19 Nov 1978 05:00:00 GMT"],["Last-Modified","Wed, 17 Aug 2011 22:32:19 GMT"],["Cache-Control","store, no-cache, must-revalidate"],["Cache-Control","post-check=0, pre-check=0"],["Content-Type","text\/html; charset=utf-8"],["X-Powered-By","PHP\/5.3.2; Qcodo\/0.3.24 (Qcodo Beta 3)"],["Set-Cookie","minted_tr=UQ%BDn%830%10%7E%17%EF%04%8C%9D41S%D4%A1%EA%90%8C%5D%91%03%26%B5%0Aq%E5%3B%90h%94w%EF%9D%0BR3%FA%BE%DF%3B%5B%23%B5%B9%83QF4%16EeMa%EE%8F%F4%9E%5C%14%957%B2%02%B3Oh%0D%7D%40%1E%95%5BEC%29%8D%E8%7C%04%AC%27%0F%3E%01%B2%D4%3B%BD%D7%07%A9%19%2F%8C%E8%ED%13%AC%A4%DA%95%85%D2%05%C3z%95G%D7%B9%189%0D%8C%A6%E0%8BC%AB6%83%BF%A1k7M%18%F2f%04%0C%83%FFq1%87%AF1%3F%BD%9F%B3%E3%DBkv%FA8%B2D%AA%25%E7%BF%0Fe%27%AC%5CC+%8C%B1q%A2%3A%2F%CD%93%E2iH%D4%D44%C4%1A%E7o%27%96%BDS%F6%9F%29%27%AD%94%86%0C%80%EFU%F2%F9%B6%04%04d%D6e%3DX%EB%C0_o%EB%88NJ%FEC%CF%0A%C9%8A%17ftv%EC%D3e%0A%26%1C%D6%AE%E9%27H%40.C%C0O%17%21k%ED%9C%5D%7D%87%90%01Z%F4%81%8C%E7%A5a%DAd%91pC%B0%93%CBHE%1A%A4-%1E%BF; expires=Mon, 13-Feb-2012 22:32:20 GMT; path=\/"]]}}

Making app-requests useful

1. Parse to .csv (Python)
2. Load to BI PostgreSQL DB
3. Clean & more parsing
4. Sessionize
5. Analyze



Sessionization in PostgreSQL Part 1

```
-- get event sequence #s & seconds after prior event
CREATE TABLE v_sessions1 AS
SELECT *,
       ROW_NUMBER() OVER(Members) AS event_seq_number,
       event_at - LAG(event_at) OVER(Members)
           AS interval_to_prior
  FROM v_events
WINDOW Members
    AS (PARTITION BY member_id   -- unique member ID
             ORDER BY event_at   -- timestamp of event
       )
;
```

Sessionization in PostgreSQL Part 2

```
-- update with session sequence #
CREATE TABLE v_sessions2 AS
SELECT *,
       COUNT(CASE WHEN interval_to_prior IS NULL OR
                       interval_to_prior > '30 minutes'
                  THEN 1 ELSE NULL END) OVER(Members)
           AS session_seq_number
  FROM v_sessions1
WINDOW Members
    AS (PARTITION BY member_id      -- unique member ID
             ORDER BY event_seq_number  -- Session #
       )
;
```

Sessionization in PostgreSQL Part 3

```
-- now roll up into sessions getting session start, total time in session,
-- site areas explored, other site specific rollups

CREATE TABLE v_sessions AS
SELECT member_id,
       session_seq_number,
       MIN(event_at) AS session_start_at,
       COUNT(*) AS num_events_in_session,
       SUM(CASE WHEN interval_to_prior > '30 minutes'
                THEN NULL ELSE interval_to_prior END) AS session_duration,
       STRING_AGG(DISCRETE site_area, ', ') AS site_areas_visited,
       <other site specific aggregations>
  FROM v_sessions2
 GROUP BY member_id,
          session_seq_number
 ORDER BY member_id,
          session_seq_number
;
```

How “mining” app-request logs helps us understand the customer:

1. Sales funnel analysis by product & YOY.
2. Customer’s individual interests, preferences, …
3. Customer’s evolution in relation w/ Minted
4. Usage based customer segments
5. And we will discover more!