MIGRATING A DATA STACK FROM AWS TO AZURE VIA RASPBERRY PI

@SOOBROSA @WUNDERLIST @MICROSOFT

THE VOICE OF GOD


I am the Metatron.

THE TEAM

TOPICS

1. ORIGINS
2. PLANNING
3. IN-FLIGHT REFACTOR
4. FIXUP
5. BUZZWORDS

DISCLAIMER

ALL OPINIONS SHARED ARE MY OWN

I MIGHT BE STATISTICALLY MEAN

SCALE AND COMPLEXITY

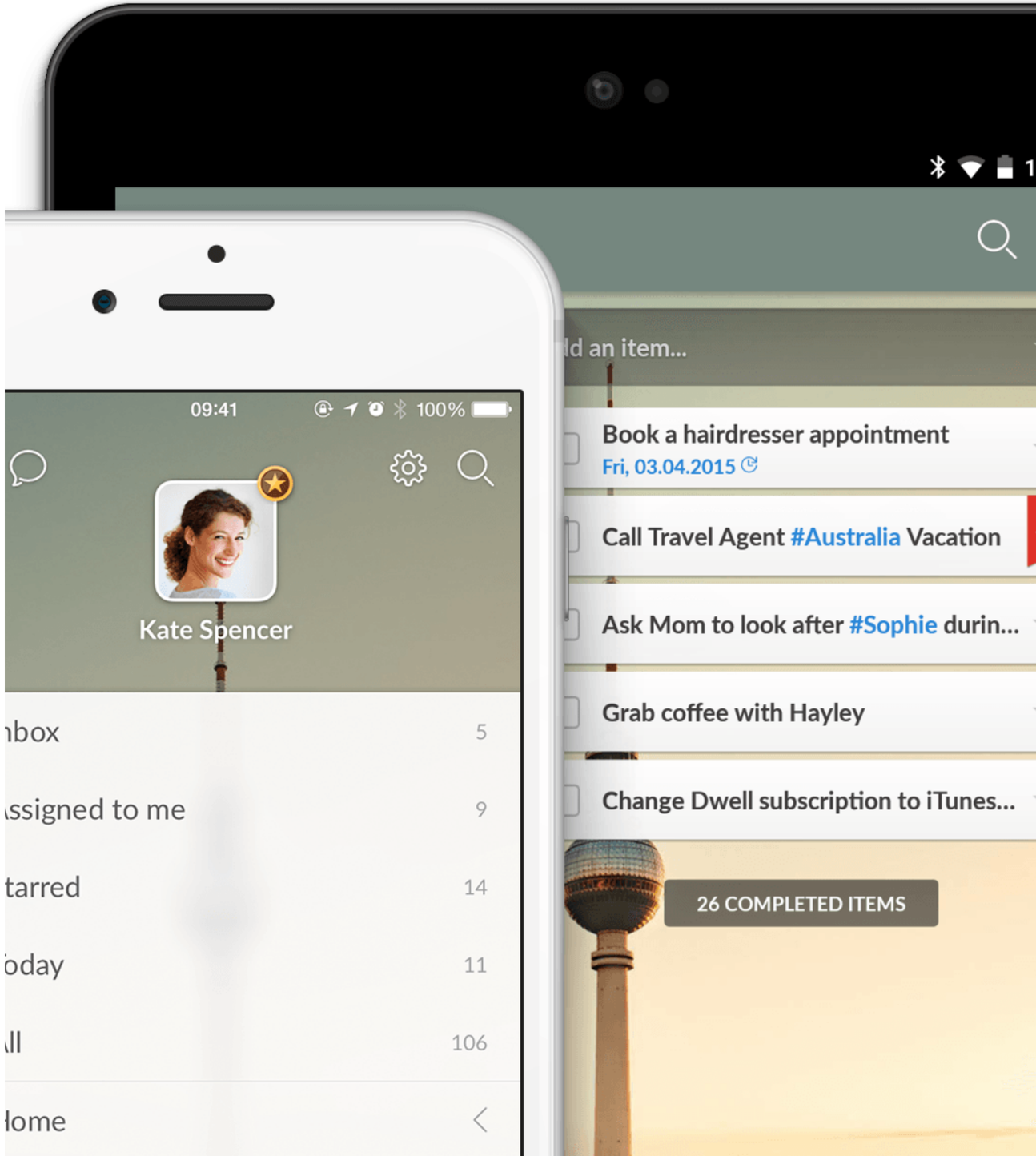
WUNDERLIST

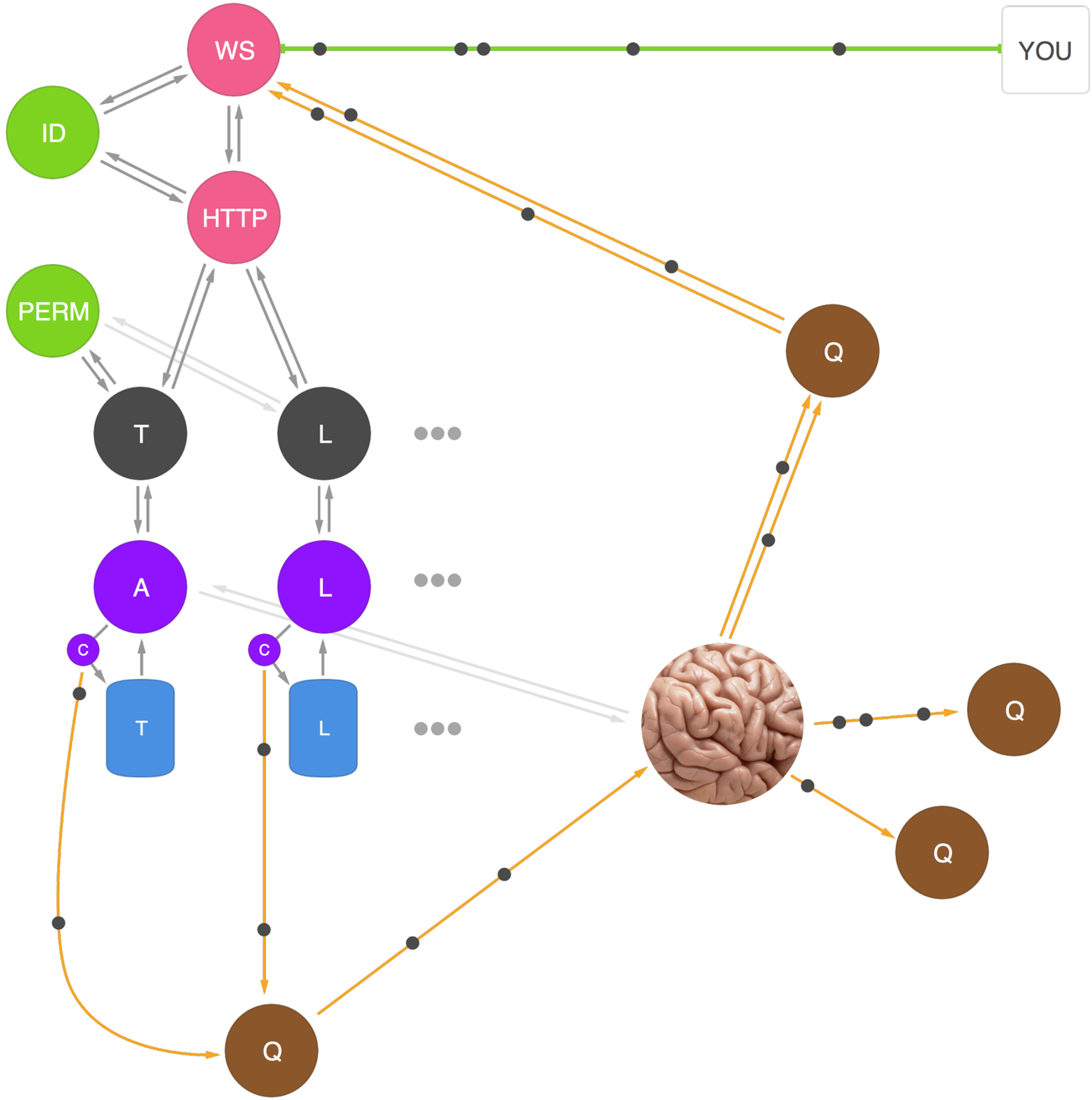
PRODUCTIVITY APP ON IPHONE. IPAD. MAC. ANDROID. WINDOWS. KINDLE FIRE AND THE WEB

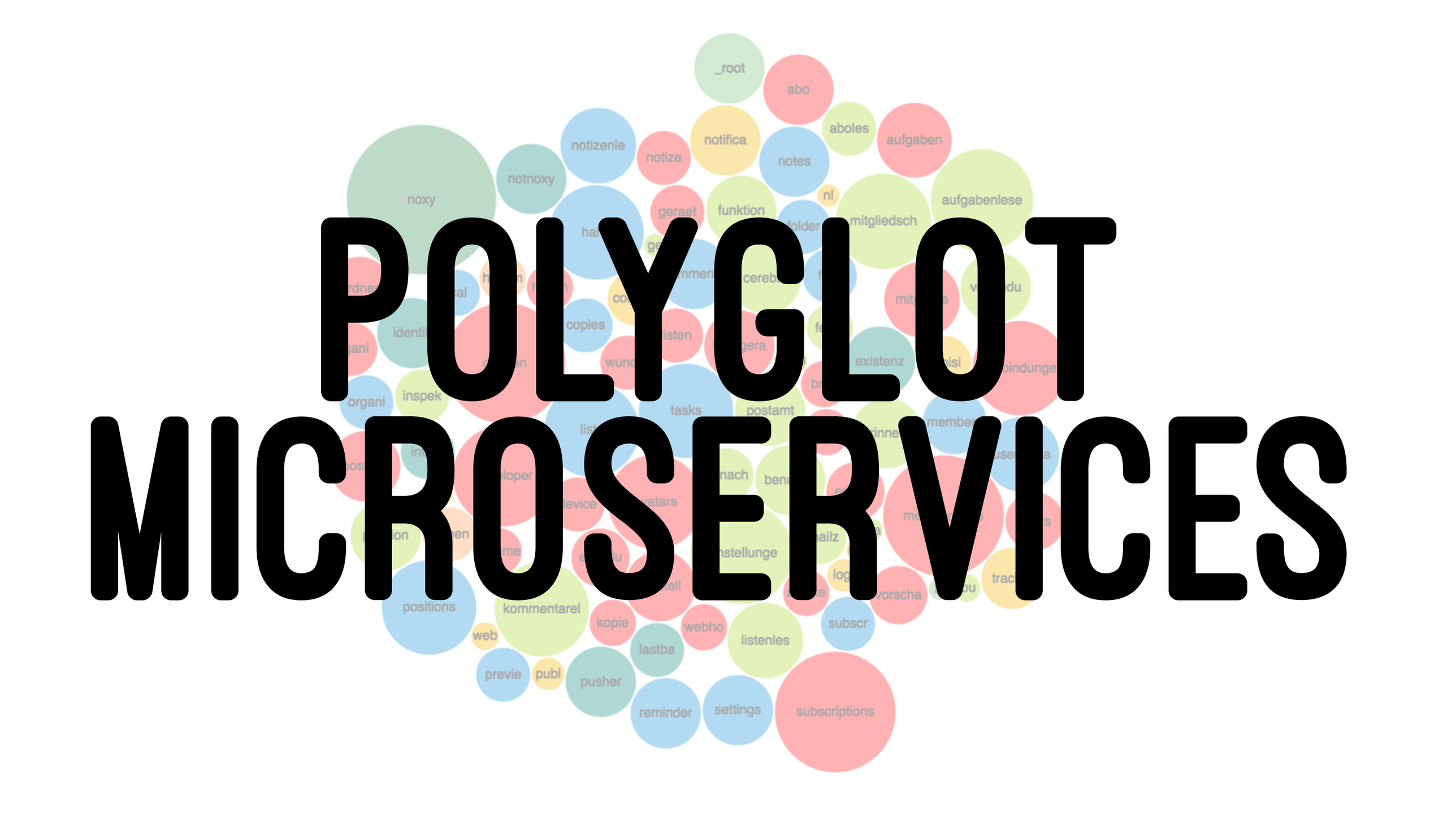
21+ MILLION USERS. 6 YEARS. HEADCOUNT OF 67

FROM MONOLITHIC RAILS TO POLYGLOT MICROSERVICES

SCALA. CLOJURE. GO ON AWS




WS
YOU
ID
HTTP
PERM
Q
T
L
A
L
C
C
T
L
Q
Q
Q


POLYGLOT
MICROSERVICES
_root
abo
notifica
aboles
aufgaben
notizenle
notize
notes
notnoxy
noxy
nl
aufgabenlese
geraet
funktion
folder
mitgliedsch
cereb
copies
existenz
inspek
tasks
postamt
member
nach
kommentarel
kopie
webho
listenles
subscr
lastba
web
previe
publ
pusher
reminder
settings
subscriptions
positions

Honest Status Page
@honest_update

Követés

We replaced our monolith with micro services so that every outage could be more like a murder mystery.

Fordítás megtekintése

RETWEET 2 652 KEDVELÉS 1 929

16:10 - 2015. okt. 7.

DATA MOSTLY IN POSTGRESQL

- HOSTED ON AWS
  - ~33 DATABASES
    - ~120 CONCURRENT CONNECTIONS/DATABASE
      - USUALLY 2-3 TABLES PER DATABASE
- `tasks` TABLE CONTAINS 1 BILLION RECORDS.

DATA SIZING

- **COLLECT** EVERY EVENT **FROM CLIENTS** 125M/DAY
- **PARSE & FILTER** COMPRESSED LOGS' 375GB/DAY
- **MIRROR EVERY** PRODUCTION DATABASE 35GB INC./DAY
- **LOAD** EXTERNAL SOURCES **(E.G.: APP STORE, PAYMENTS)**
- **CALCULATE** KPIS, AGGREGATES, BUSINESS LOGIC - 200+ QUERIES
- **SELF SERVICE DATA FOR EVERYBODY**

INGREDIENTS

UNIX

BASH

MAKE

CRONTAB

SQL

WHY MAKE?

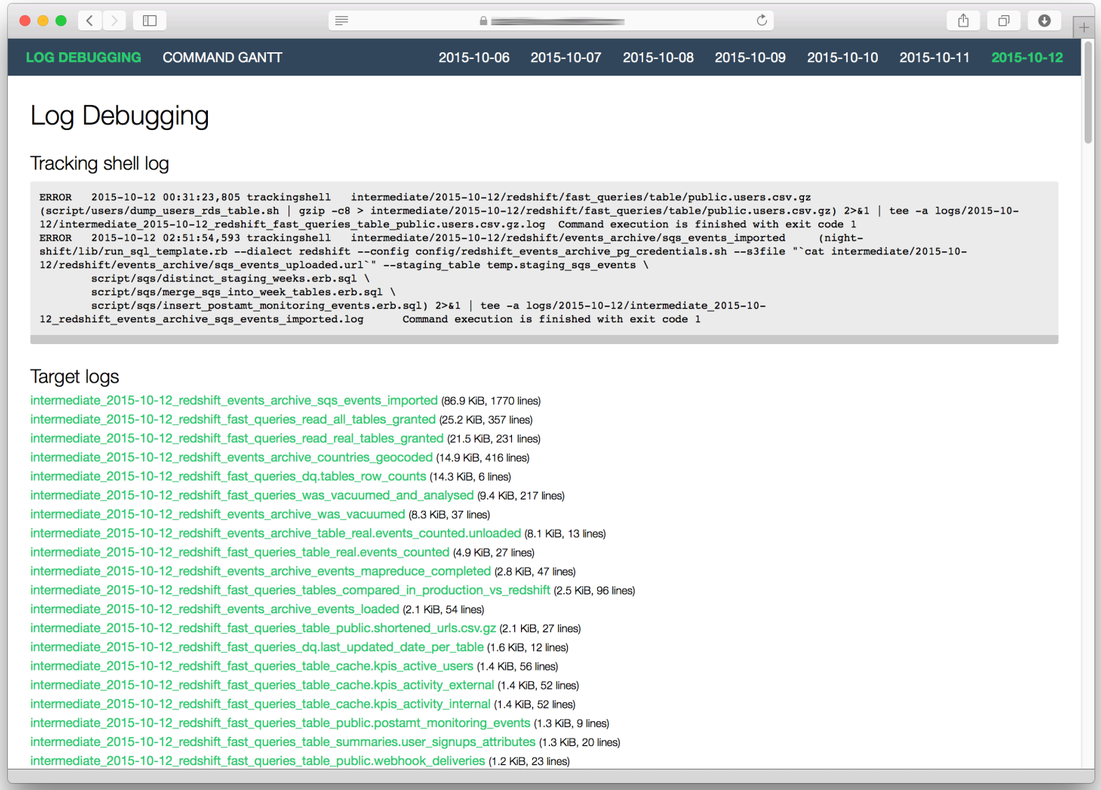
- BLAME JEFF HAMMERBACHER
- IT'S A MACHINE-READABLE DOCUMENTATION
- SUPPORTS DEPENDENCIES, RETRIES
- EASY TO TEST. EVEN LOCALLY ALL TARGET
- EXECUTES MULTIPLE TARGETS IN PARALLEL
- CODING IS NECESSARY TO MODIFY -> CHANGELOG IN GIT

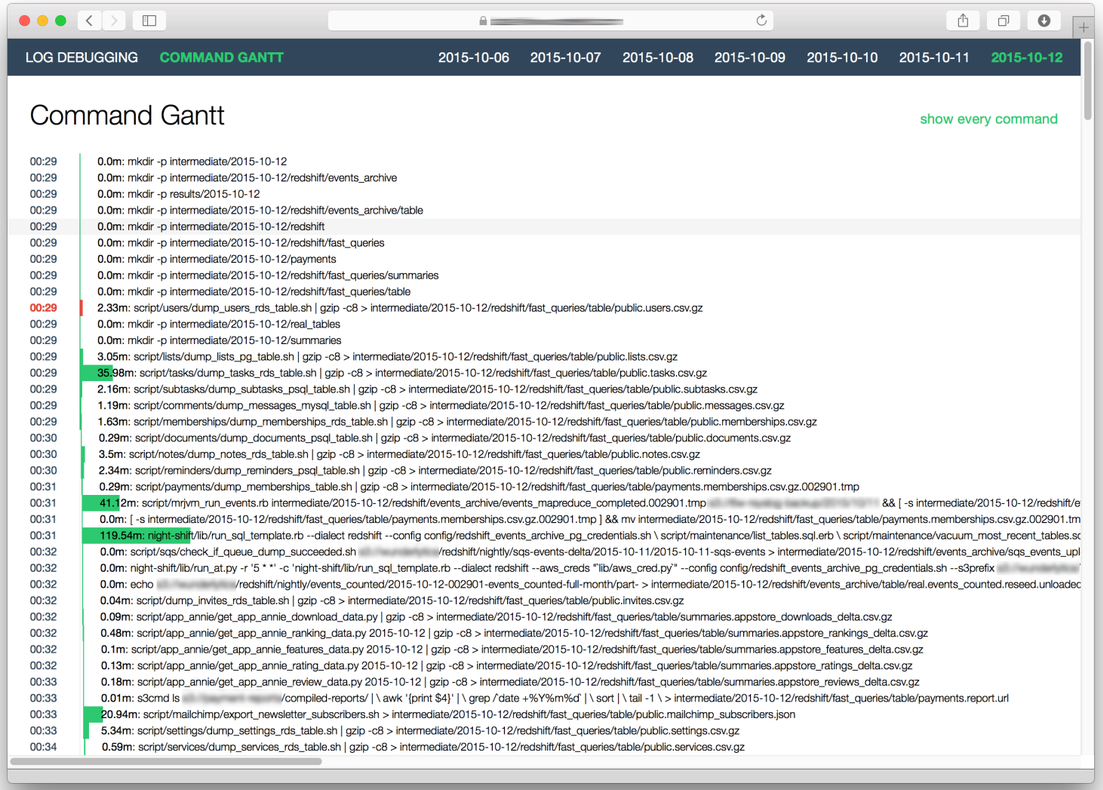
```
# Dumps users table from production.
public.users.csv.gz:
  script/users/dump_users_rds_table.sh | gzip -c8 > $@

# Upload the compressed dump into S3.
users_s3_url:=s3://wunderlytics/.../users-delta-$(TODAY)-$(TMP_TOKEN).csv.gz
public.users.csv.gz.uploaded: public.users.csv.gz
  script/move_to_s3.sh $< $(users_s3_url) > $@

# Load users into Redshift.
public.users: | public.users.csv.gz.uploaded
  night-shift/lib/run_sql_template.rb \
    --dialect redshift \
    --aws_creds "`lib/aws_cred.py`" \
    --config config/redshift_fast_queries_pg_credentials.sh \
    --s3file "`cat $(firstword $|)`" \
    script/users/schema.sql.erb \
    script/users/replace_users_table.sql.erb
  touch $@
```

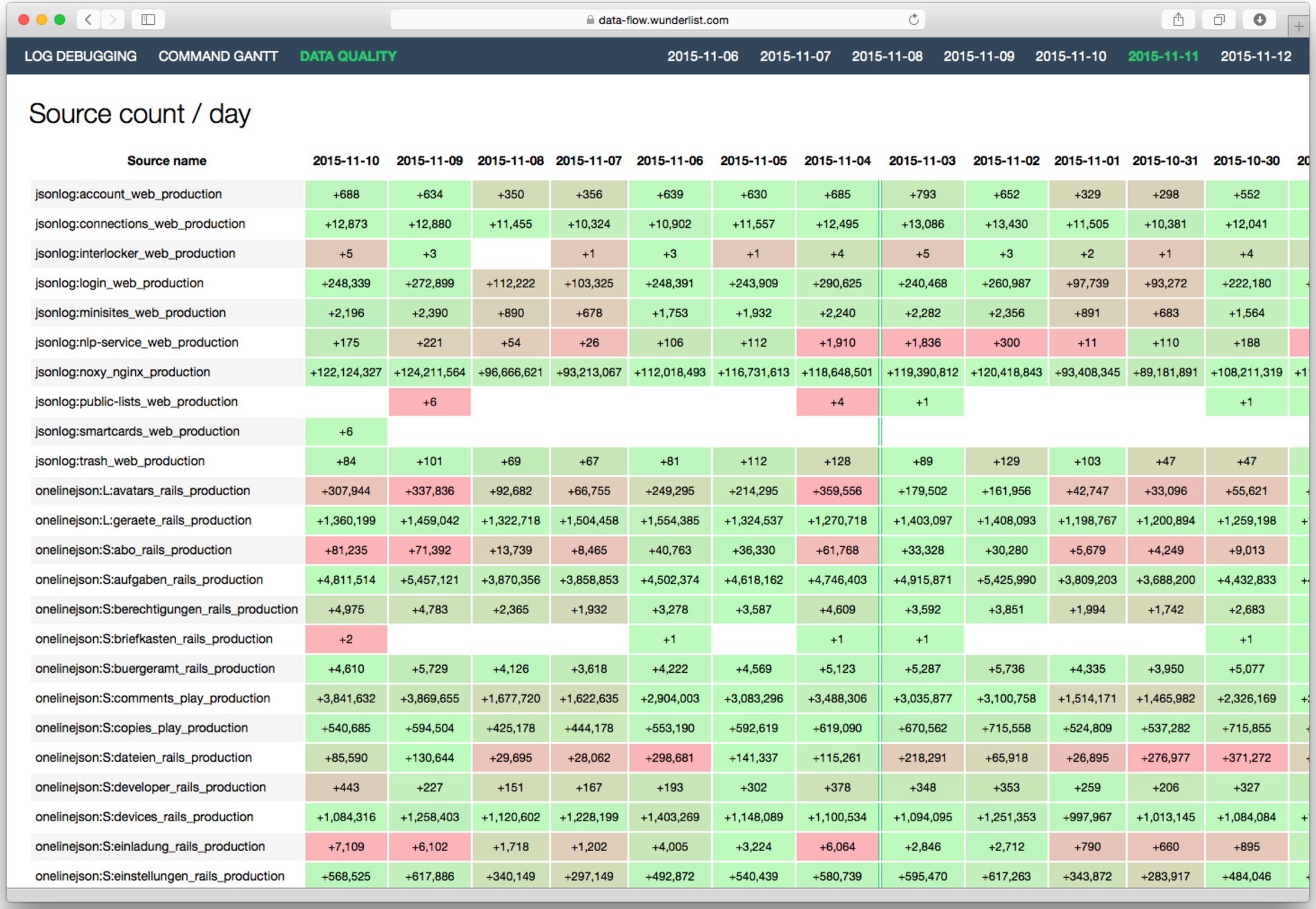
NIGHT-SHIFT AS ETL

- CRON FOR SCHEDULING
- MAKE FOR DEPENDENCIES, PARTIAL RESULTS, RETRIES
- GLUE WITH BASH
- INJECT VARIABLES AND LOGIC INTO SQL WITH RUBY'S ERB
- RUNS IN A TRACKING SHELL, SO TIMING, OUTPUT AND ERRORS ARE LOGGED
- MONITORING INTERFACE IN FLASK
- LOCALLY TESTABLE
- OPEN SOURCE

```
# Create a temporary table
CREATE TABLE #notes_staging (
  <%= specs.map {|col, type| "#{col} #{type}"}.join(", ") %>
) SORTKEY(id);

# Load data into the temporary table from S3
COPY #notes_staging ( <%= columns.join "," %> )
FROM '<%= s3file %>'
WITH CREDENTIALS <%= aws_creds %>
GZIP TRUNCATECOLUMNS DELIMITER '\001' ESCAPE REMOVEQUOTES;

# Updating the changed values
UPDATE notes SET <%= updates.join "," %>
FROM #notes_staging u
WHERE ( u.deleted_at IS NOT NULL OR u.updated_at > notes.updated_at )
  AND notes.id = u.id;

# Inserting the new rows
INSERT INTO notes ( <%= columns.join "," %> ) (
  SELECT <%= columns.join "," %>
  FROM #notes_staging u
  WHERE u.id NOT IN (SELECT id FROM notes) );
```


LOG DEBUGGING
COMMAND GANTT
2015-10-06
2015-10-07
2015-10-08
2015-10-09
2015-10-10
2015-10-11
2015-10-12
Log Debugging
Tracking shell log
ERROR 2015-10-12 00:31:23,805 trackingshell intermediate/2015-10-12/redshift/fast_queries/table/public.users.csv.gz
(script/users/dump_users_rds_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.users.csv.gz) 2>&1 | tee -a logs/2015-10-
12/intermediate_2015-10-12_redshift_fast_queries_table_public.users.csv.gz.log Command execution is finished with exit code 1
ERROR 2015-10-12 02:51:54,593 trackingshell intermediate/2015-10-12/redshift/events_archive/sqs_events_imported (night-
shift/lib/run_sql_template.rb --dialect redshift --config config/redshift_events_archive_pg_credentials.sh --s3file "`cat intermediate/2015-10-
12/redshift/events_archive/sqs_events_uploaded.url`" --staging_table temp.staging_sqs_events \
script/sqs/distinct_staging_weeks.erb.sql \
script/sqs/merge_sqs_into_week_tables.erb.sql \
script/sqs/insert_postamt_monitoring_events.erb.sql) 2>&1 | tee -a logs/2015-10-12/intermediate_2015-10-
12_redshift_events_archive_sqs_events_imported.log Command execution is finished with exit code 1
Target logs
intermediate_2015-10-12_redshift_events_archive_sqs_events_imported (86.9 KiB, 1770 lines)
intermediate_2015-10-12_redshift_fast_queries_read_all_tables_granted (25.2 KiB, 357 lines)
intermediate_2015-10-12_redshift_fast_queries_read_real_tables_granted (21.5 KiB, 231 lines)
intermediate_2015-10-12_redshift_events_archive_countries_geocoded (14.9 KiB, 416 lines)
intermediate_2015-10-12_redshift_fast_queries_dq.tables_row_counts (14.3 KiB, 6 lines)
intermediate_2015-10-12_redshift_fast_queries_was_vacuumed_and_analysed (9.4 KiB, 217 lines)
intermediate_2015-10-12_redshift_events_archive_was_vacuumed (8.3 KiB, 37 lines)
intermediate_2015-10-12_redshift_events_archive_table_real.events_counted.unloaded (8.1 KiB, 13 lines)
intermediate_2015-10-12_redshift_fast_queries_table_real.events_counted (4.9 KiB, 27 lines)
intermediate_2015-10-12_redshift_events_archive_events_mapreduce_completed (2.8 KiB, 47 lines)
intermediate_2015-10-12_redshift_fast_queries_tables_compared_in_production_vs_redshift (2.5 KiB, 96 lines)
intermediate_2015-10-12_redshift_events_archive_events_loaded (2.1 KiB, 54 lines)
intermediate_2015-10-12_redshift_fast_queries_table_public.shortened_urls.csv.gz (2.1 KiB, 27 lines)
intermediate_2015-10-12_redshift_fast_queries_dq.last_updated_date_per_table (1.6 KiB, 12 lines)
intermediate_2015-10-12_redshift_fast_queries_table_cache.kpis_active_users (1.4 KiB, 56 lines)
intermediate_2015-10-12_redshift_fast_queries_table_cache.kpis_activity_external (1.4 KiB, 52 lines)
intermediate_2015-10-12_redshift_fast_queries_table_cache.kpis_activity_internal (1.4 KiB, 52 lines)
intermediate_2015-10-12_redshift_fast_queries_table_public.postamt_monitoring_events (1.3 KiB, 9 lines)
intermediate_2015-10-12_redshift_fast_queries_table_summaries.user_signups_attributes (1.3 KiB, 20 lines)
intermediate_2015-10-12_redshift_fast_queries_table_public.webhook_deliveries (1.2 KiB, 23 lines)


LOG DEBUGGING
COMMAND GANTT
2015-10-06
2015-10-07
2015-10-08
2015-10-09
2015-10-10
2015-10-11
2015-10-12
Command Gantt
show every command
00:29 0.0m: mkdir -p intermediate/2015-10-12
00:29 0.0m: mkdir -p intermediate/2015-10-12/redshift/events_archive
00:29 0.0m: mkdir -p results/2015-10-12
00:29 0.0m: mkdir -p intermediate/2015-10-12/redshift/events_archive/table
00:29 0.0m: mkdir -p intermediate/2015-10-12/redshift
00:29 0.0m: mkdir -p intermediate/2015-10-12/redshift/fast_queries
00:29 0.0m: mkdir -p intermediate/2015-10-12/payments
00:29 0.0m: mkdir -p intermediate/2015-10-12/redshift/fast_queries/summaries
00:29 0.0m: mkdir -p intermediate/2015-10-12/redshift/fast_queries/table
00:29 2.33m: script/users/dump_users_rds_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.users.csv.gz
00:29 0.0m: mkdir -p intermediate/2015-10-12/real_tables
00:29 0.0m: mkdir -p intermediate/2015-10-12/summaries
00:29 3.05m: script/lists/dump_lists_pg_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.lists.csv.gz
00:29 35.98m: script/tasks/dump_tasks_rds_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.tasks.csv.gz
00:29 2.16m: script/subtasks/dump_subtasks_psql_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.subtasks.csv.gz
00:29 1.19m: script/comments/dump_messages_mysql_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.messages.csv.gz
00:29 1.63m: script/memberships/dump_memberships_rds_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.memberships.csv.gz
00:30 0.29m: script/documents/dump_documents_psql_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.documents.csv.gz
00:30 3.5m: script/notes/dump_notes_rds_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.notes.csv.gz
00:30 2.34m: script/reminders/dump_reminders_psql_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.reminders.csv.gz
00:31 0.29m: script/payments/dump_memberships_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/payments.memberships.csv.gz.002901.tmp
00:31 41.12m: script/mrjvm_run_events.rb intermediate/2015-10-12/redshift/events_archive/events_mapreduce_completed.002901.tmp && [-s intermediate/2015-10-12/redshift/e
00:31 0.0m: [-s intermediate/2015-10-12/redshift/fast_queries/table/payments.memberships.csv.gz.002901.tmp] && mv intermediate/2015-10-12/redshift/fast_queries/table/payments.memberships.csv.gz.002901.tm
00:31 119.54m: night-shift/lib/run_sql_template.rb --dialect redshift --config config/redshift_events_archive_pg_credentials.sh \ script/maintenance/list_tables.sql.erb \ script/maintenance/vacuum_most_recent_tables.sc
00:32 0.0m: script/sqs/check_if_queue_dump_succeeded.sh /redshift/nightly/sqs-events-delta/2015-10-11/2015-10-11-sqs-events > intermediate/2015-10-12/redshift/events_archive/sqs_events_upl
00:32 0.0m: night-shift/lib/run_at.py -r '5 * *' -c 'night-shift/lib/run_sql_template.rb --dialect redshift --aws_creds "lib/aws_cred.py" --config config/redshift_events_archive_pg_credentials.sh --s3prefix
00:32 0.0m: echo /redshift/nightly/events_counted/2015-10-12-002901-events_counted-full-month/part- > intermediate/2015-10-12/redshift/events_archive/table/real.events_counted.reseed.unloadec
00:32 0.04m: script/dump_invites_rds_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.invites.csv.gz
00:32 0.09m: script/app_annie/get_app_annie_download_data.py | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/summaries.appstore_downloads_delta.csv.gz
00:32 0.48m: script/app_annie/get_app_annie_ranking_data.py 2015-10-12 | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/summaries.appstore_rankings_delta.csv.gz
00:32 0.1m: script/app_annie/get_app_annie_features_data.py 2015-10-12 | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/summaries.appstore_features_delta.csv.gz
00:32 0.13m: script/app_annie/get_app_annie_rating_data.py 2015-10-12 | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/summaries.appstore_ratings_delta.csv.gz
00:33 0.18m: script/app_annie/get_app_annie_review_data.py 2015-10-12 | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/summaries.appstore_reviews_delta.csv.gz
00:33 0.01m: s3cmd ls /compiled-reports/ | \ awk '{print $4}' | \ grep /`date +%Y%m%d` | \ sort | \ tail -1 \ > intermediate/2015-10-12/redshift/fast_queries/table/payments.report.url
00:33 20.94m: script/mailchimp/export_newsletter_subscribers.sh > intermediate/2015-10-12/redshift/fast_queries/table/public.mailchimp_subscribers.json
00:33 5.34m: script/settings/dump_settings_rds_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.settings.csv.gz
00:34 0.59m: script/services/dump_services_rds_table.sh | gzip -c8 > intermediate/2015-10-12/redshift/fast_queries/table/public.services.csv.gz



Source count / day

Source name	2015-11-10	2015-11-09	2015-11-08	2015-11-07	2015-11-06	2015-11-05	2015-11-04	2015-11-03	2015-11-02	2015-11-01	2015-10-31	2015-10-30	20
jsonlog:account_web_production	+688	+634	+350	+356	+639	+630	+685	+793	+652	+329	+298	+552	
jsonlog:connections_web_production	+12,873	+12,880	+11,455	+10,324	+10,902	+11,557	+12,495	+13,086	+13,430	+11,505	+10,381	+12,041	
jsonlog:interlocker_web_production	+5	+3		+1	+3	+1	+4	+5	+3	+2	+1	+4	
jsonlog:login_web_production	+248,339	+272,899	+112,222	+103,325	+248,391	+243,909	+290,625	+240,468	+260,987	+97,739	+93,272	+222,180	
jsonlog:minisites_web_production	+2,196	+2,390	+890	+678	+1,753	+1,932	+2,240	+2,282	+2,356	+891	+683	+1,564	
jsonlog:nlp-service_web_production	+175	+221	+54	+26	+106	+112	+1,910	+1,836	+300	+11	+110	+188	
jsonlog:noxy_nginx_production	+122,124,327	+124,211,564	+96,666,621	+93,213,067	+112,018,493	+116,731,613	+118,648,501	+119,390,812	+120,418,843	+93,408,345	+89,181,891	+108,211,319	+1
jsonlog:public-lists_web_production		+6					+4	+1				+1	
jsonlog:smartcards_web_production	+6												
jsonlog:trash_web_production	+84	+101	+69	+67	+81	+112	+128	+89	+129	+103	+47	+47	
onelinejson:L:avatars_rails_production	+307,944	+337,836	+92,682	+66,755	+249,295	+214,295	+359,556	+179,502	+161,956	+42,747	+33,096	+55,621	
onelinejson:L:geraete_rails_production	+1,360,199	+1,459,042	+1,322,718	+1,504,458	+1,554,385	+1,324,537	+1,270,718	+1,403,097	+1,408,093	+1,198,767	+1,200,894	+1,259,198	+
onelinejson:S:abo_rails_production	+81,235	+71,392	+13,739	+8,465	+40,763	+36,330	+61,768	+33,328	+30,280	+5,679	+4,249	+9,013	
onelinejson:S:aufgaben_rails_production	+4,811,514	+5,457,121	+3,870,356	+3,858,853	+4,502,374	+4,618,162	+4,746,403	+4,915,871	+5,425,990	+3,809,203	+3,688,200	+4,432,833	+
onelinejson:S:berechtigungen_rails_production	+4,975	+4,783	+2,365	+1,932	+3,278	+3,587	+4,609	+3,592	+3,851	+1,994	+1,742	+2,683	
onelinejson:S:briefkasten_rails_production	+2				+1		+1	+1				+1	
onelinejson:S:buergeramt_rails_production	+4,610	+5,729	+4,126	+3,618	+4,222	+4,569	+5,123	+5,287	+5,736	+4,335	+3,950	+5,077	
onelinejson:S:comments_play_production	+3,841,632	+3,869,655	+1,677,720	+1,622,635	+2,904,003	+3,083,296	+3,488,306	+3,035,877	+3,100,758	+1,514,171	+1,465,982	+2,326,169	+
onelinejson:S:copies_play_production	+540,685	+594,504	+425,178	+444,178	+553,190	+592,619	+619,090	+670,562	+715,558	+524,809	+537,282	+715,855	
onelinejson:S:dateien_rails_production	+85,590	+130,644	+29,695	+28,062	+298,681	+141,337	+115,261	+218,291	+65,918	+26,895	+276,977	+371,272	
onelinejson:S:developer_rails_production	+443	+227	+151	+167	+193	+302	+378	+348	+353	+259	+206	+327	
onelinejson:S:devices_rails_production	+1,084,316	+1,258,403	+1,120,602	+1,228,199	+1,403,269	+1,148,089	+1,100,534	+1,094,095	+1,251,353	+997,967	+1,013,145	+1,084,084	+
onelinejson:S:einladung_rails_production	+7,109	+6,102	+1,718	+1,202	+4,005	+3,224	+6,064	+2,846	+2,712	+790	+660	+895	
onelinejson:S:einstellungen_rails_production	+568,525	+617,886	+340,149	+297,149	+492,872	+540,439	+580,739	+595,470	+617,263	+343,872	+283,917	+484,046	

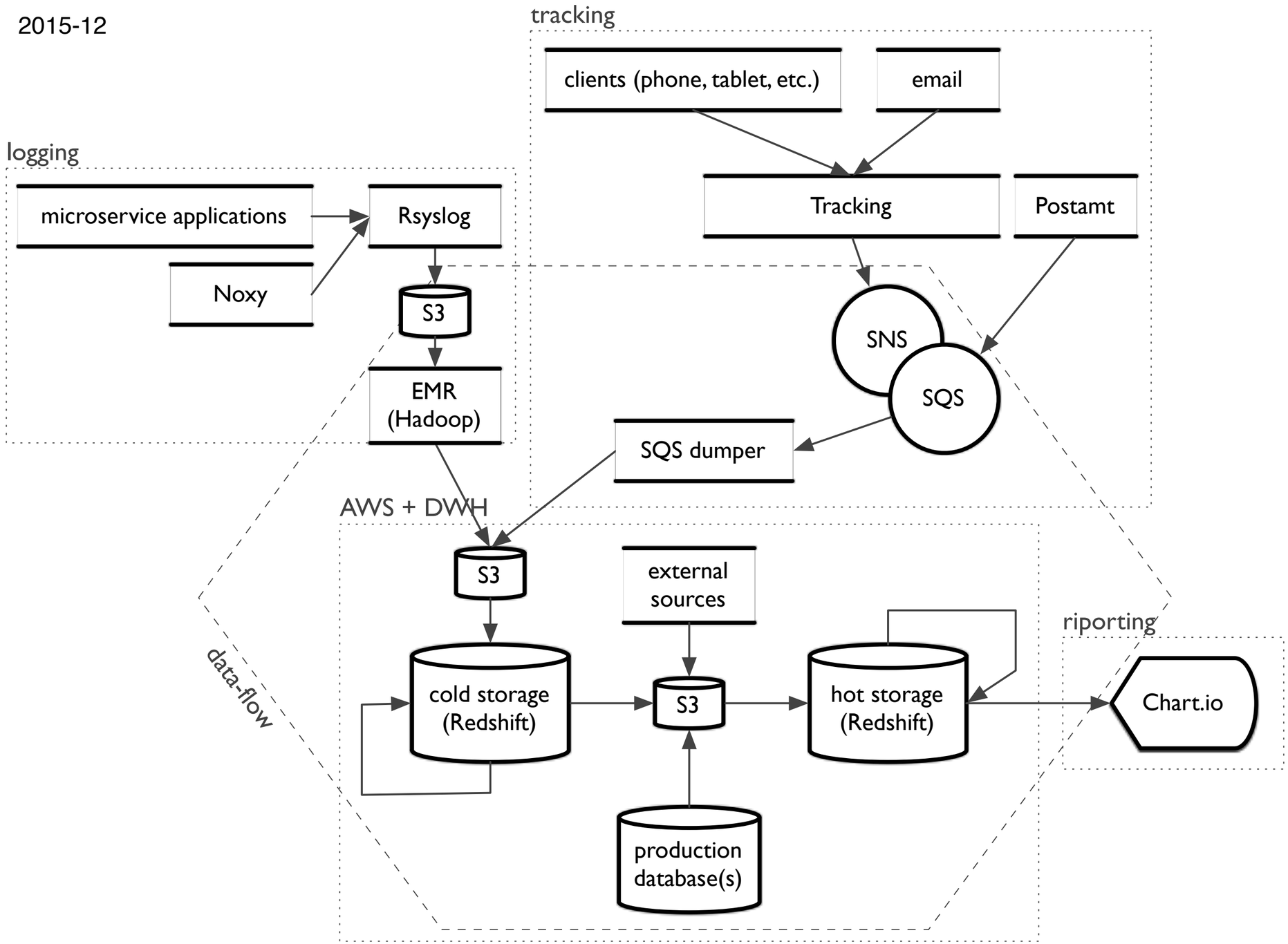
ANALYTICS IN REDSHIFT

10 TB COMPRESSED AGGREGATION

TWO CLUSTERS:

- HOT: 22 X DC1.LARGE
(2 VCPU, 15GB RAM, 160GB SSD)
- COLD: 6 X DS2.XLARGE
(4 VCPU, 31GB RAM, 2TB HDD)

2015-12


tracking
clients (phone, tablet, etc.)
email
logging
Tracking
Postamt
microservice applications
Rsyslog
Noxy
S3
SNS
SQS
EMR
(Hadoop)
SQS dumper
AWS + DWH
S3
external
sources
riporting
data-flow
cold storage
(Redshift)
S3
hot storage
(Redshift)
Chart.io
production
database(s)

PLANNING

LET'S MOVE A DATA ARCHITECTURE FROM AWS TO AZURE

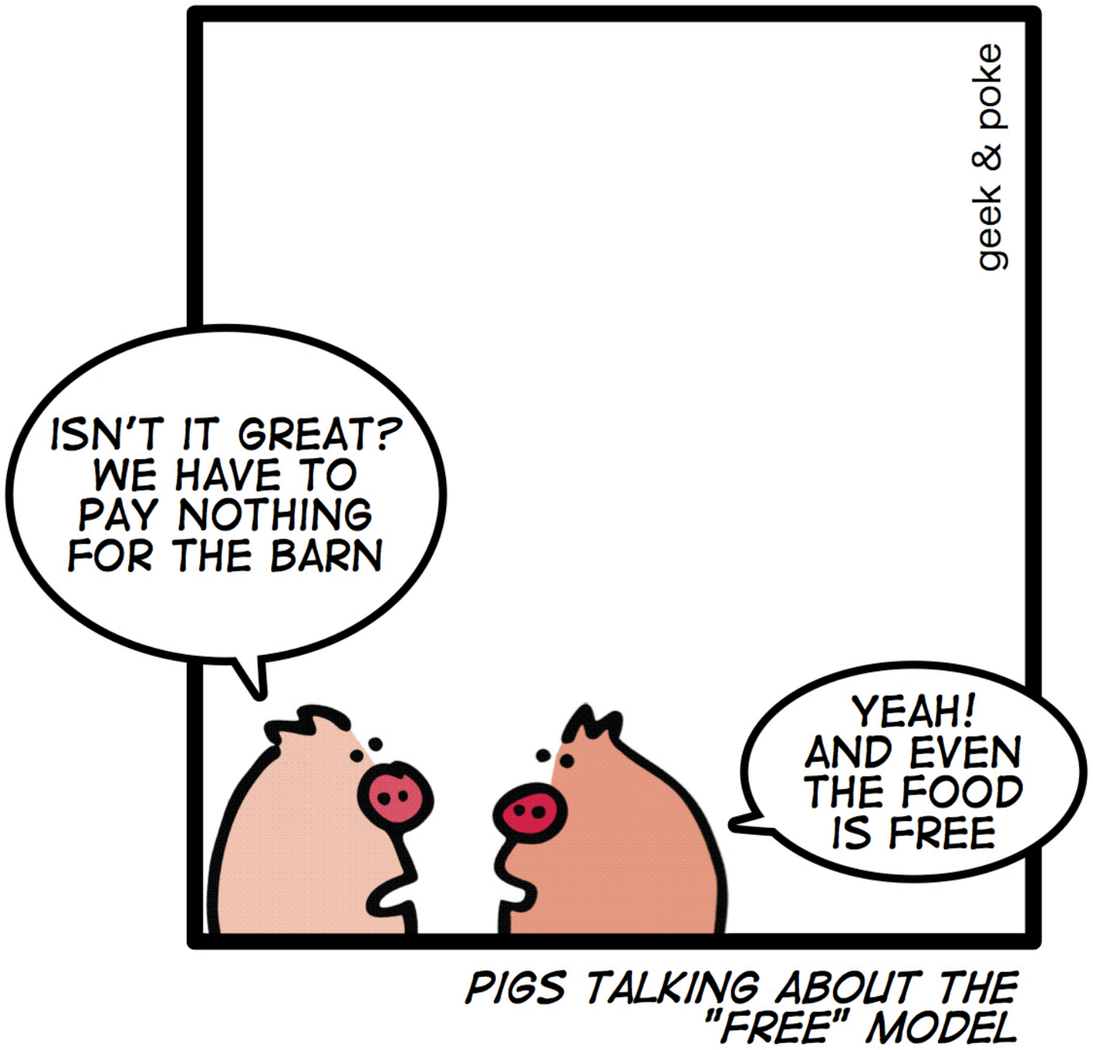
WITH AN AVERAGE OF

1.5 ENGINEERS

AT HAND IN ANY GIVEN MOMENT.

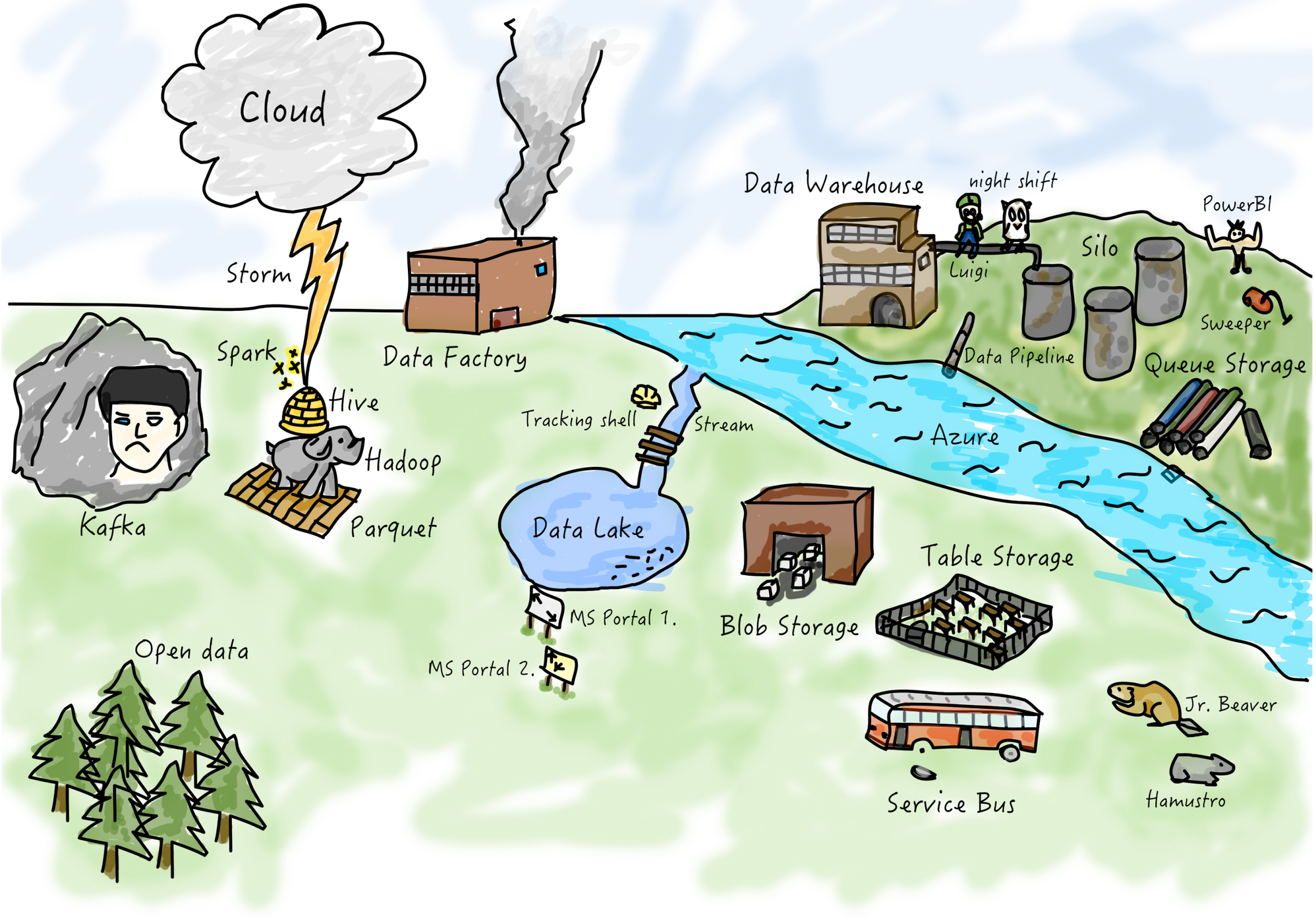
TRANSLATED TO BUSINESS

- TOTAL COST OF OWNERSHIP IS DEAD SERIOUS
- CAN'T DO 24/7 SUPPORT ON DATA
- FORENSIC ANALYSIS IS NOT OUR SCOPE
- REMOVE IF YOU CAN



THE BUCOLIC DATA LANDSCAPE

(MACIEJ CEGŁOWSKI)


Cloud
Storm
Spark
Hive
Hadoop
Parquet
Kafka
Data Factory
Data Warehouse
night shift
Luigi
PowerBI
Silo
Sweeper
Data Pipeline
Queue Storage
Azure
Tracking shell
Stream
Data Lake
MS Portal 1.
MS Portal 2.
Blob Storage
Table Storage
Open data
Service Bus
Jr. Beaver
Hamustro

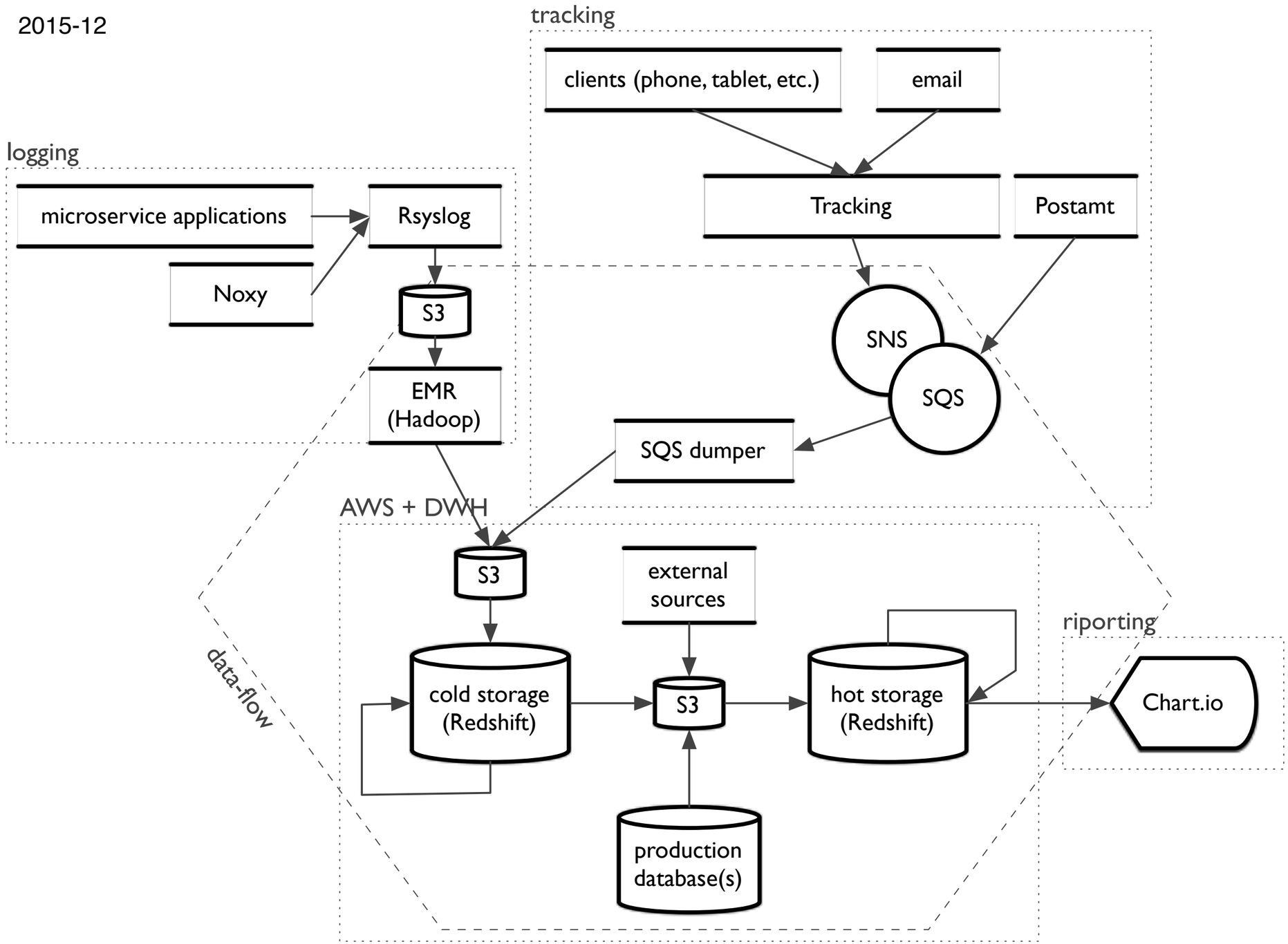
PRAY OUR LORD
JAMES MICKENS
AND LET'S GO!

IN-FLIGHT REFACTOR

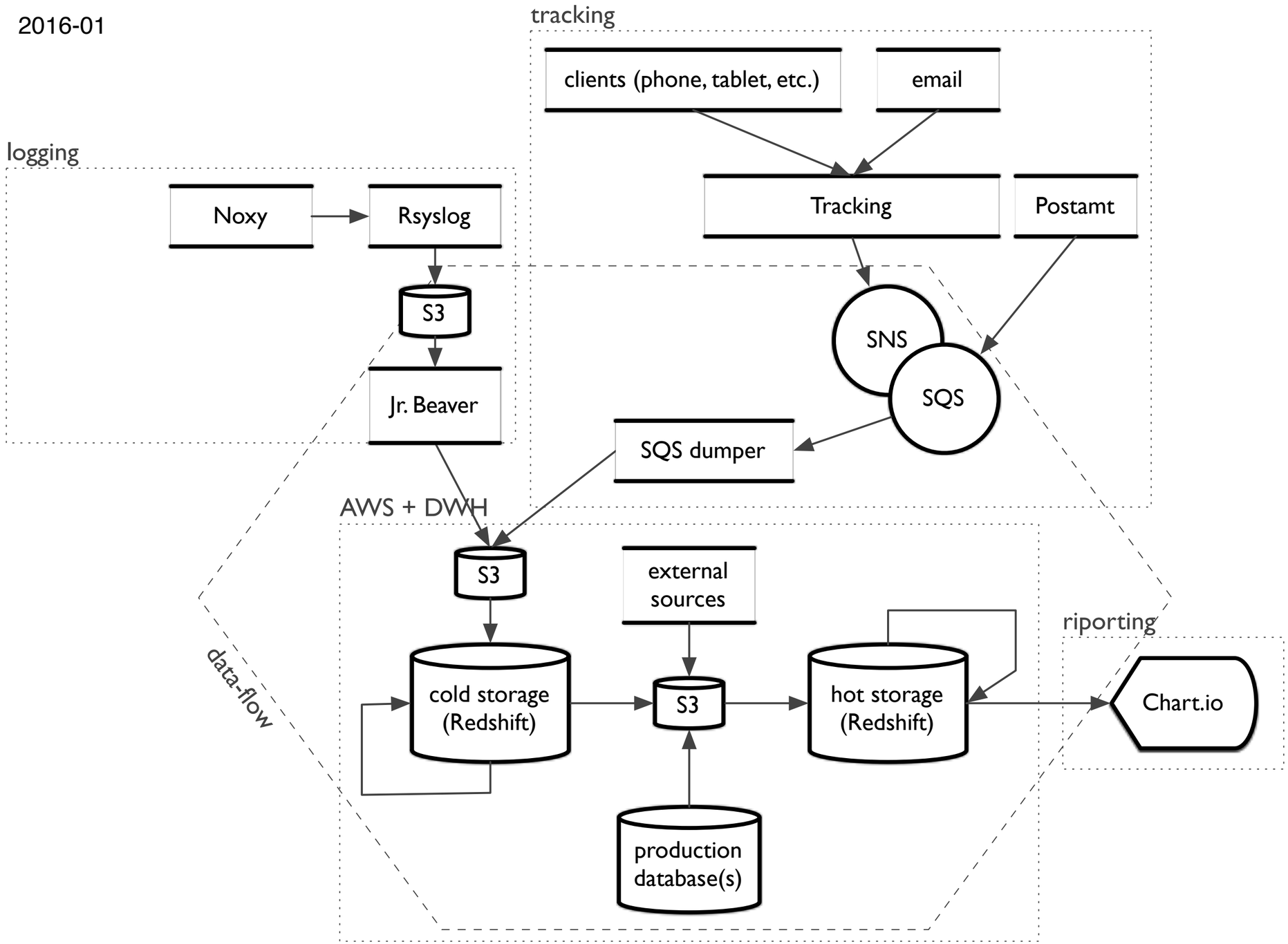
GOALS

- SIMPLIFY
- ABSTRACT AWAY AWS SPECIFIC PARTS
- REMOVE UNNECESSARY COMPLICATIONS LIKE HADOOP
- ADD AZURE SUPPORT FOR THE COMPONENTS
- REFACTOR AND MAKE THE CODE REUSABLE

2015-12


tracking
clients (phone, tablet, etc.)
email
logging
Tracking
Postamt
microservice applications
Rsyslog
Noxy
S3
SNS
SQS
EMR
(Hadoop)
SQS dumper
AWS + DWH
S3
external
sources
riporting
data-flow
cold storage
(Redshift)
S3
hot storage
(Redshift)
Chart.io
production
database(s)

2016-01


tracking
clients (phone, tablet, etc.)
email
logging
Noxy
Rsyslog
Tracking
Postamt
S3
SNS
SQS
Jr. Beaver
SQS dumper
AWS + DWH
S3
external sources
data-flow
cold storage (Redshift)
S3
hot storage (Redshift)
riporting
Chart.io
production database(s)

EMR TO JR. BEAVER

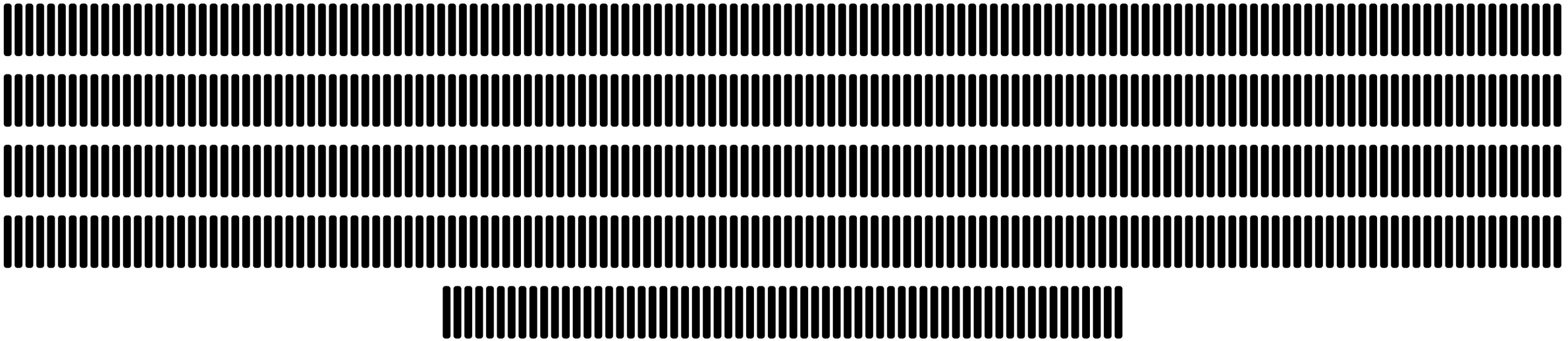
- DETECTS THE FORMAT OF EVERY LOG LINE
- LOG CRUNCHER THAT STANDARDIZES MICROSERVICES' LOGS
- CLASSIFIES EVENTS' NAMES BASED ON API'S URL
- FILTERS THE ANALYTICALLY INTERESTING ROWS
- MAP/REDUCE FUNCTIONALITY.
- HADOOP+SCALA TO MAKE+PYPY

JR. BEAVER

- CONFIGURABLE WITH YAML FILES
- WRITTEN IN PYPY INSTEAD OF GO
- USING NIGHT-SHIFT'S `make` FOR PARALLELISM
  - "BIG RAM KILLS BIG DATA"
- NO HADOOP+SCALA HEADACHE ANYMORE
  - GIVES MONITORING

VCPU COUNT

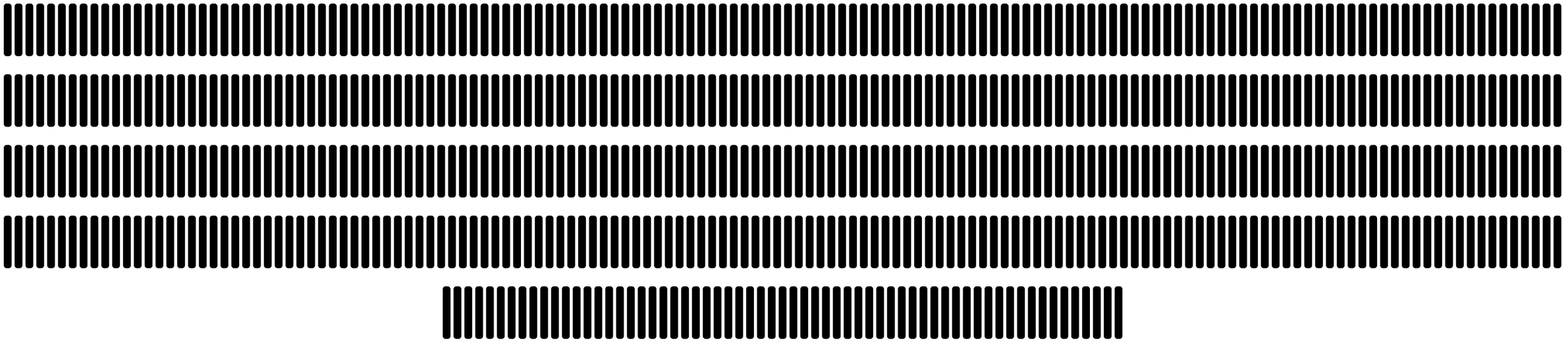
EMR (600+ IN 20 COMPUTERS):



JR. BEAVER (8 IN 1 COMPUTER):

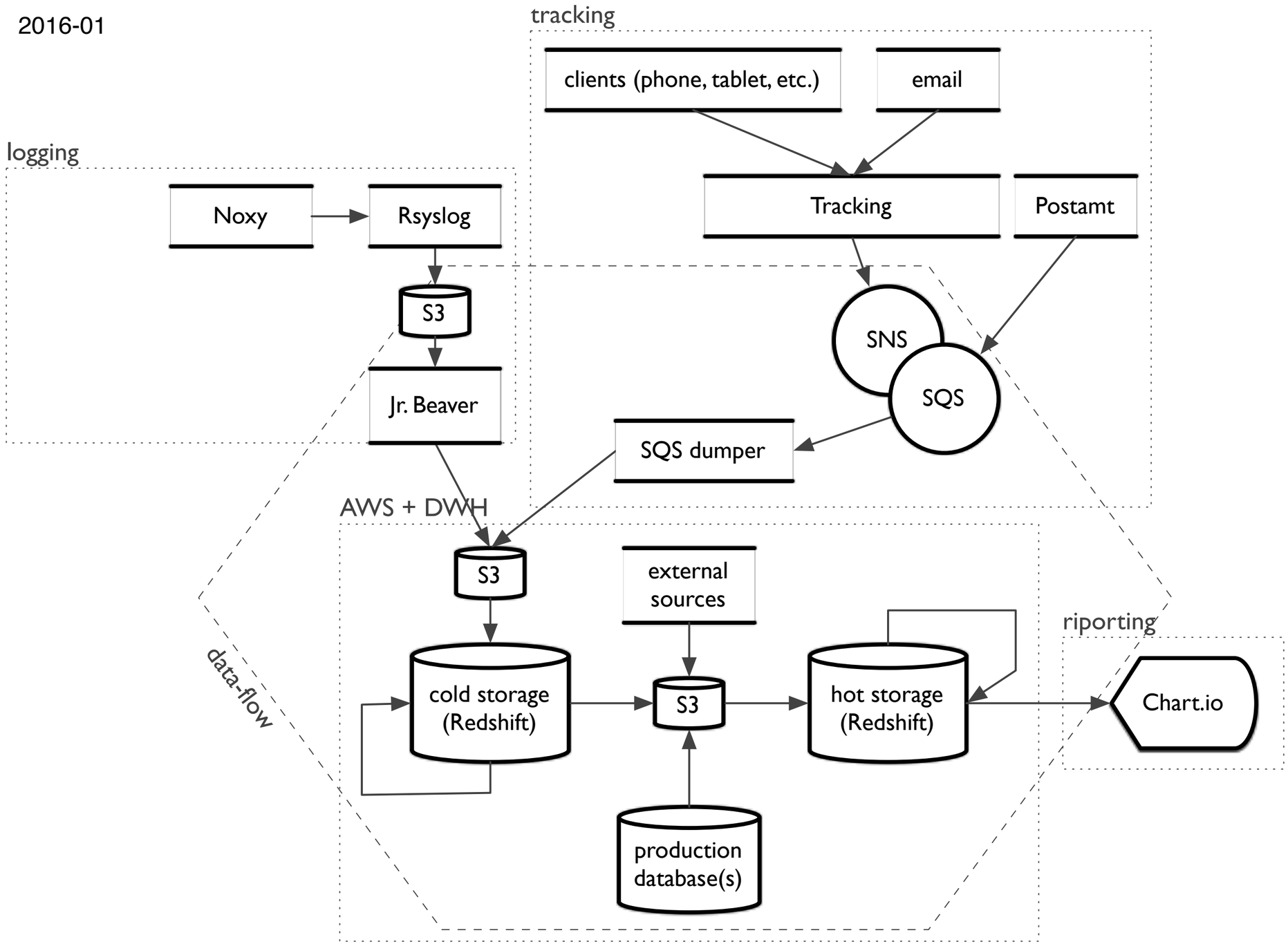
VCPU * WORKING HOURS COMPARISON

EMR (600HRS):

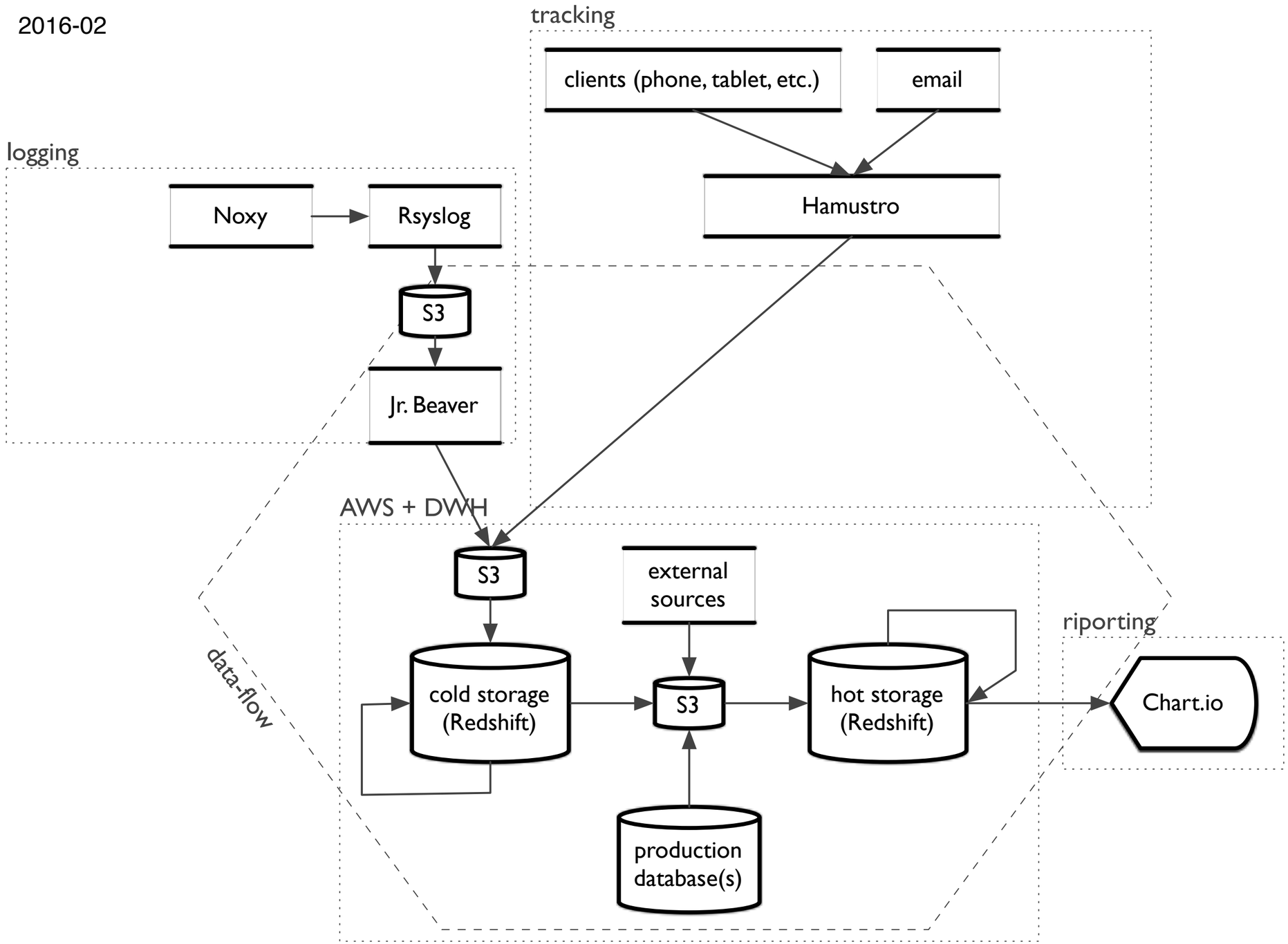


JR. BEAVER (64HRS):

2016-01


tracking
clients (phone, tablet, etc.)
email
logging
Noxy
Rsyslog
Tracking
Postamt
S3
SNS
SQS
Jr. Beaver
SQS dumper
AWS + DWH
S3
external sources
data-flow
cold storage (Redshift)
S3
hot storage (Redshift)
riporting
Chart.io
production database(s)

2016-02


tracking
clients (phone, tablet, etc.)
email
Hamustro
logging
Noxy
Rsyslog
S3
Jr. Beaver
AWS + DWH
S3
external sources
data-flow
cold storage (Redshift)
S3
hot storage (Redshift)
riporting
Chart.io
production database(s)

HOMEBREW TRACKING TO HAMUSTRO

- TRACKS CLIENT DEVICE EVENTS
  - SAVES TO CLOUD TARGETS
- HANDLES SESSIONS AND STRICT ORDER OF EVENTS
  - REWRITTEN FROM NODEJS TO GO
  - USES S3 DIRECTLY INSTEAD OF SNS/SQS
    (INSPIRED BY MARCIO CASTILHO)

HAMUSTRO

- SUPPORTS AMAZON SNS/SQS, AZURE QUEUE STORAGE
- SUPPORTS AMAZON S3, AZURE BLOB STORAGE
- TRACKS UP TO 6M EVENTS/MIN ON A SINGLE 4VCPU SERVER
- USING PROTOBUF/JSON FOR EVENTS SENDING
- WRITTEN IN GO
- OPEN SOURCE

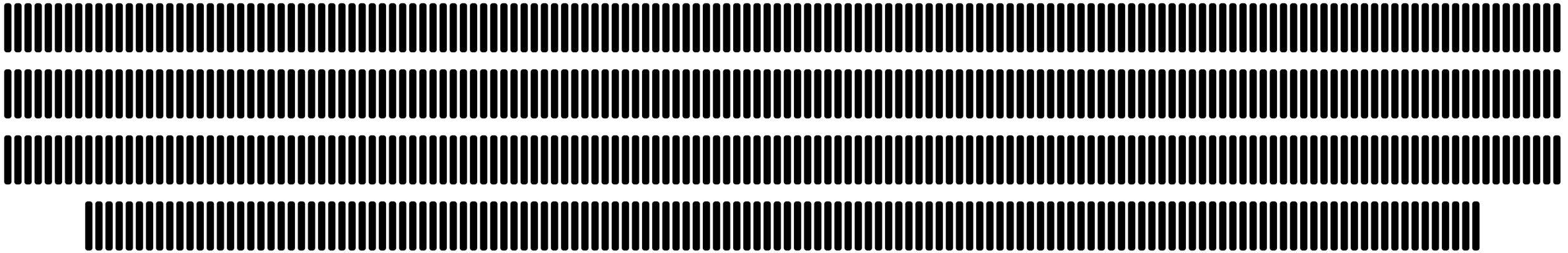
VCPU COUNT

HOMEBREW TRACKING (12X1):

HAMUSTRO (2X2):

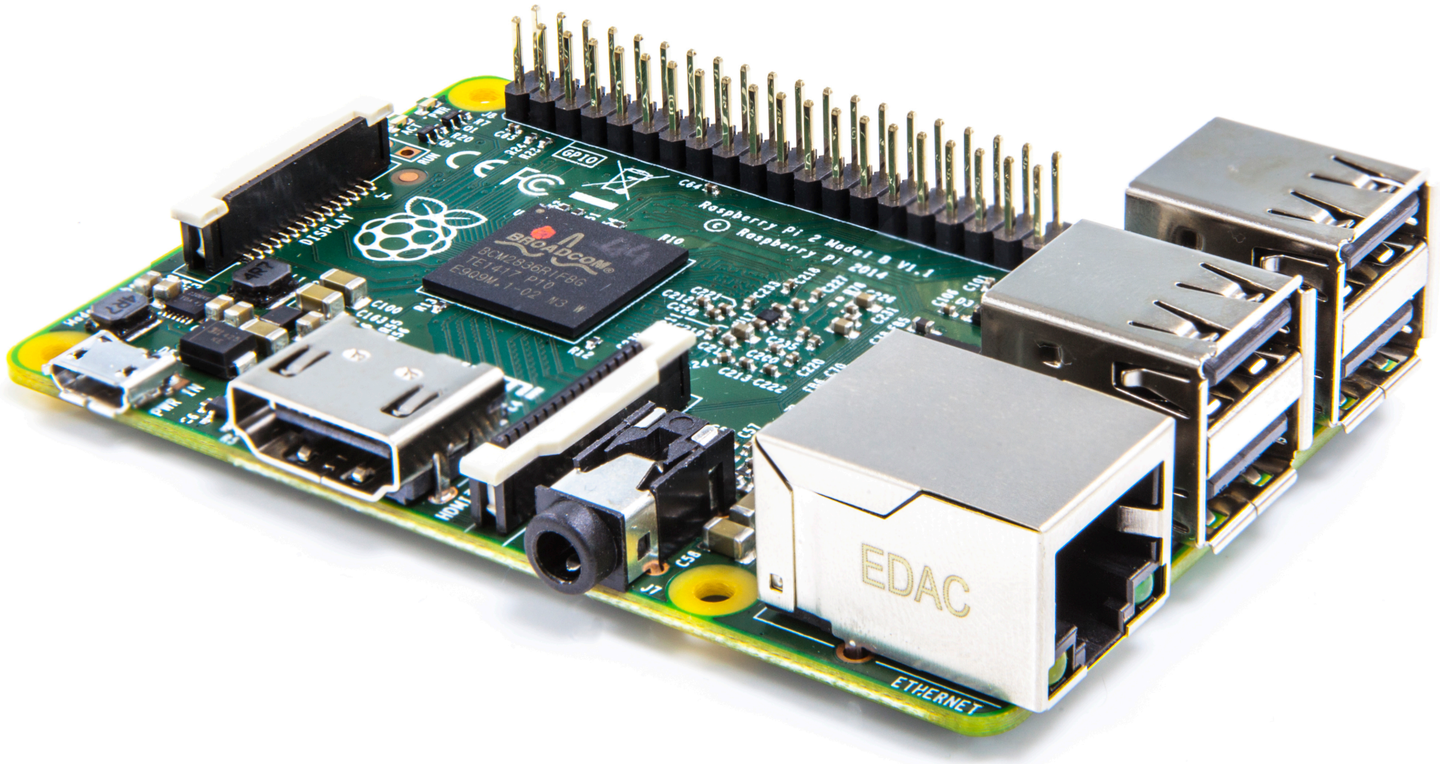
S3 VS. SNS IN A SINGLE 4VCPU COMPUTER

HAMUSTRO'S S3 DIALECT (~6M/MIN):



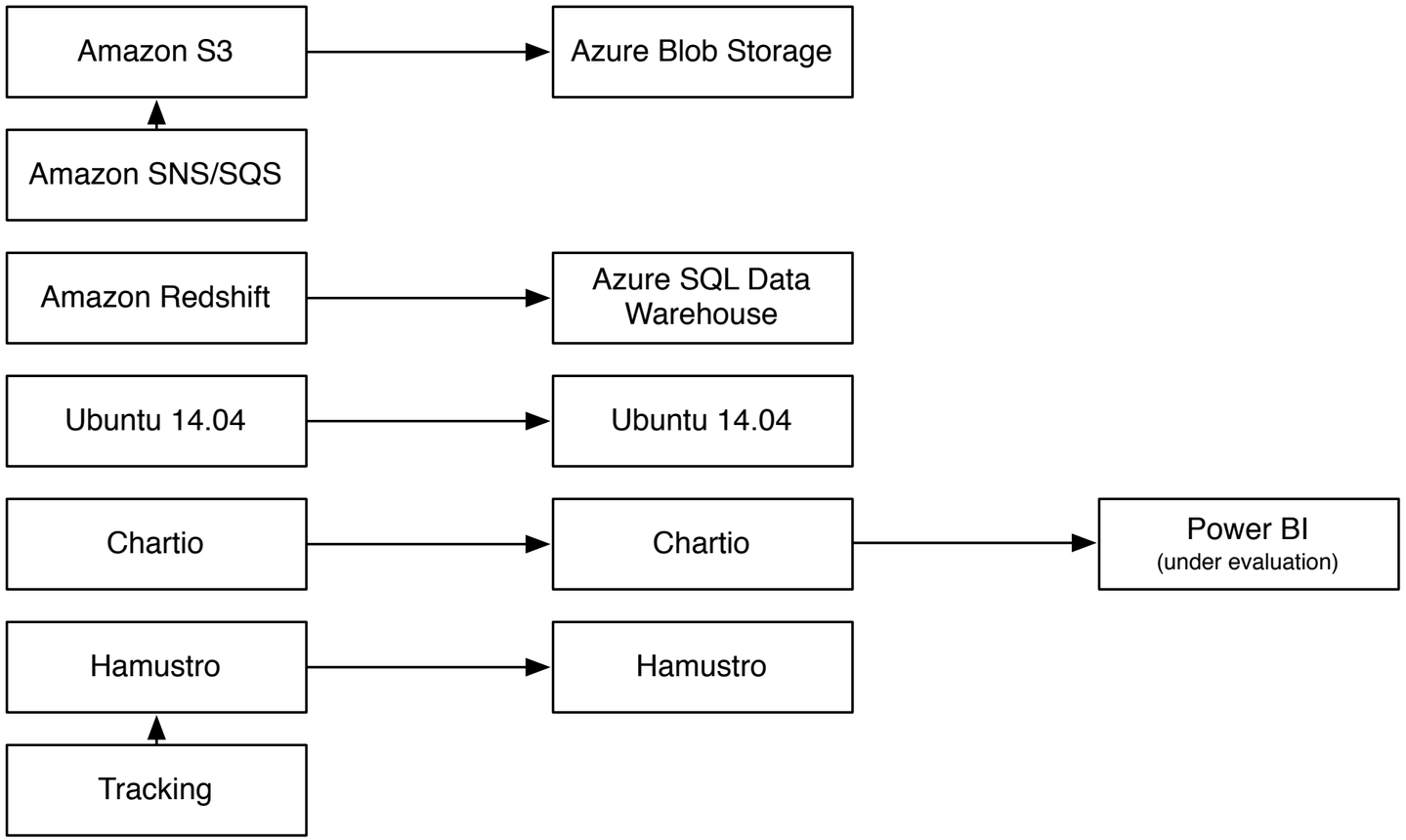
HAMUSTRO'S SNS DIALECT (~60K/MIN):

EVEN A SINGLE RASBERRYPI IS OVERKILL FOR OUR 25K EVENTS/MIN



FIXUP

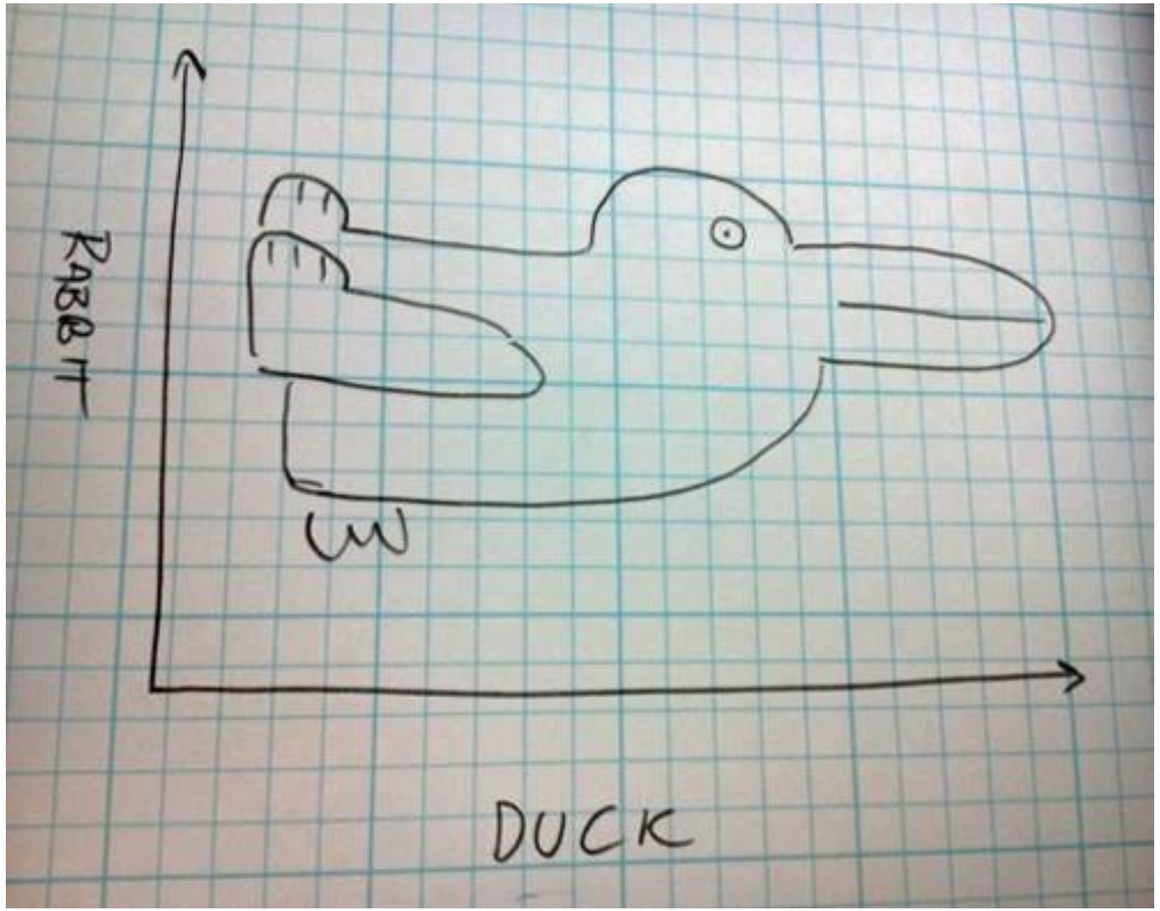
MAPPING AND BENCHMARKING



AMAZON S3 = AZURE BLOB STORAGE

AMAZON REDSHIFT ~ AZURE SQL DATA WAREHOUSE

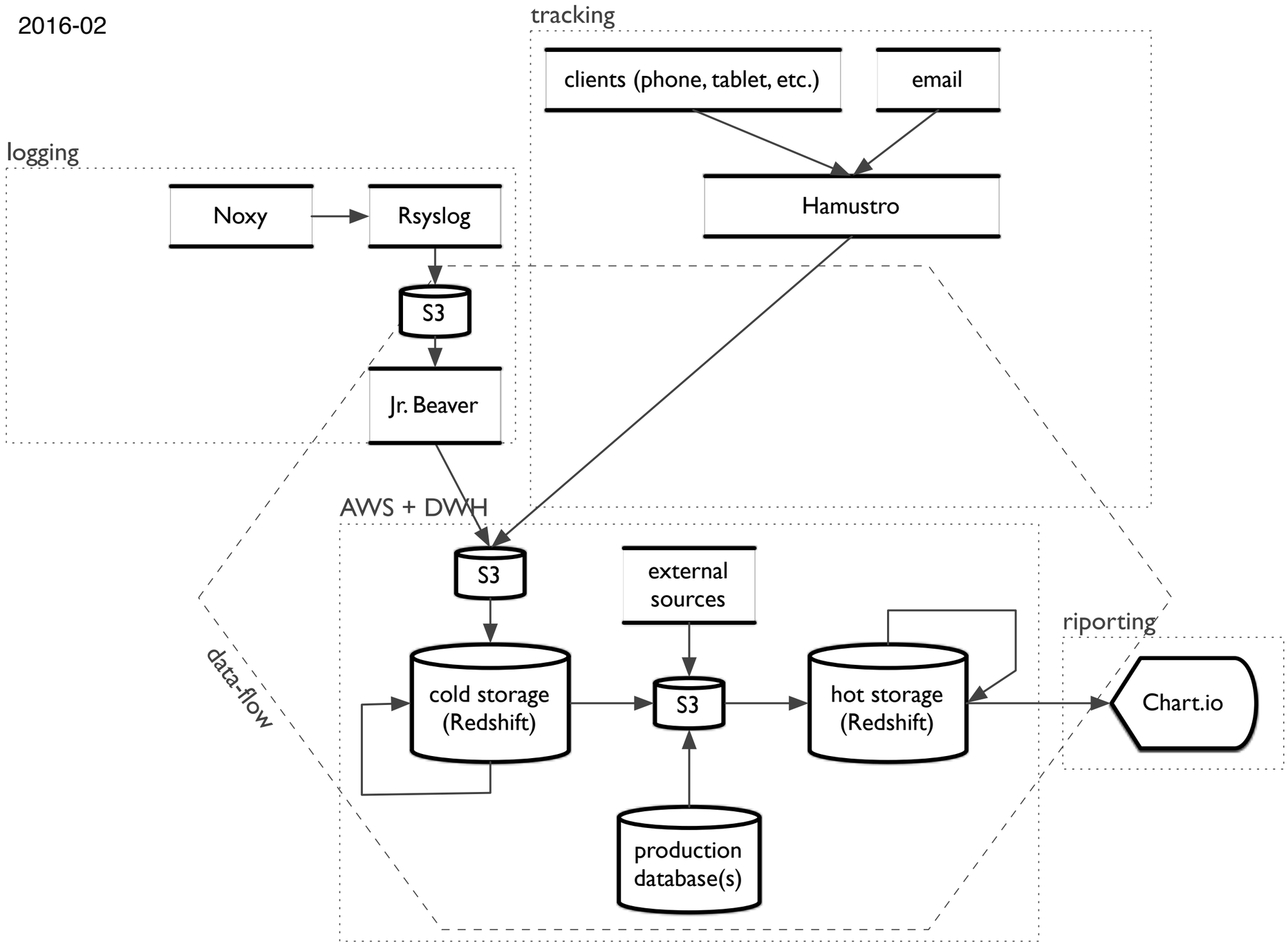
IT DEPENDS ON THE PERSPECTIVE



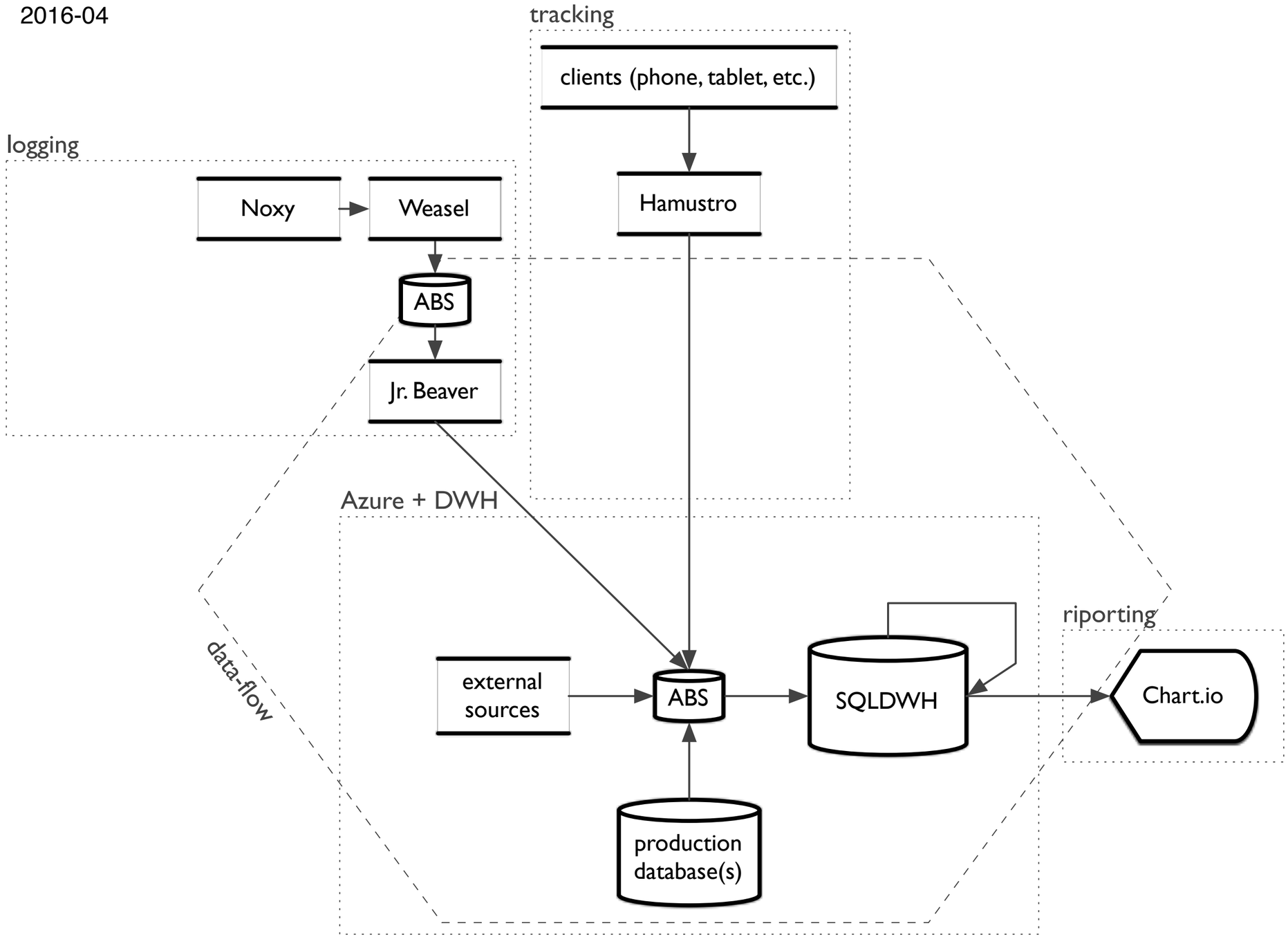
TOOLS IN UNIX FOR PRODUCTION

- AZRCMD: CLI TO DOWNLOAD AND UPLOAD FILES TO AZURE BLOB STORAGE. PROVIDES `s3cmd` LIKE FUNCTIONALITY
- CHEETAH: CLI FOR MSSQL THAT WORKS IN OSX AND LINUX AND ALSO SUPPORTS AZURE SQL DATA WAREHOUSE. SIMILAR TO `psql` AND SUPERIOR TO `sql-cli` AND MICROSOFT'S `sqlcmd`

2016-02


tracking
clients (phone, tablet, etc.)
email
Hamustro
logging
Noxy
Rsyslog
S3
Jr. Beaver
AWS + DWH
S3
external sources
data-flow
cold storage (Redshift)
S3
hot storage (Redshift)
riporting
Chart.io
production database(s)

2016-04


tracking
clients (phone, tablet, etc.)
logging
Noxy
Weasel
Hamustro
ABS
Jr. Beaver
Azure + DWH
data-flow
external sources
ABS
SQLDWH
riporting
Chart.io
production database(s)

ADAPT SQL APPROACH

- DIFFERENT LOADING STRATEGIES
- SCALE UP WHILE THE DATA PIPELINE IS RUNNING
- SET UP THE RIGHT RESOURCE GROUPS FOR EVERY USER
- DEFINE DISTRIBUTIONS AND USE PARTITIONS
- USE FULL FEATURED SQL
- FIND THE PERFECT BALANCE BETWEEN CONCURRENCY AND SPEED

BUZZWORDS

HYBRID, CLOUD AGNOSTIC DATA STACK

*

POST-CLOUD DATA INFRASTRUCTURE

AKA A DOZEN RPI POWERTAPED TOGETHER

*

REDNECK DATA

AS OPPOSING DATA SCIENCE

THIS IS A FERRY

#MAHLZEIT

@SOOBROSA