



# Data as software as data

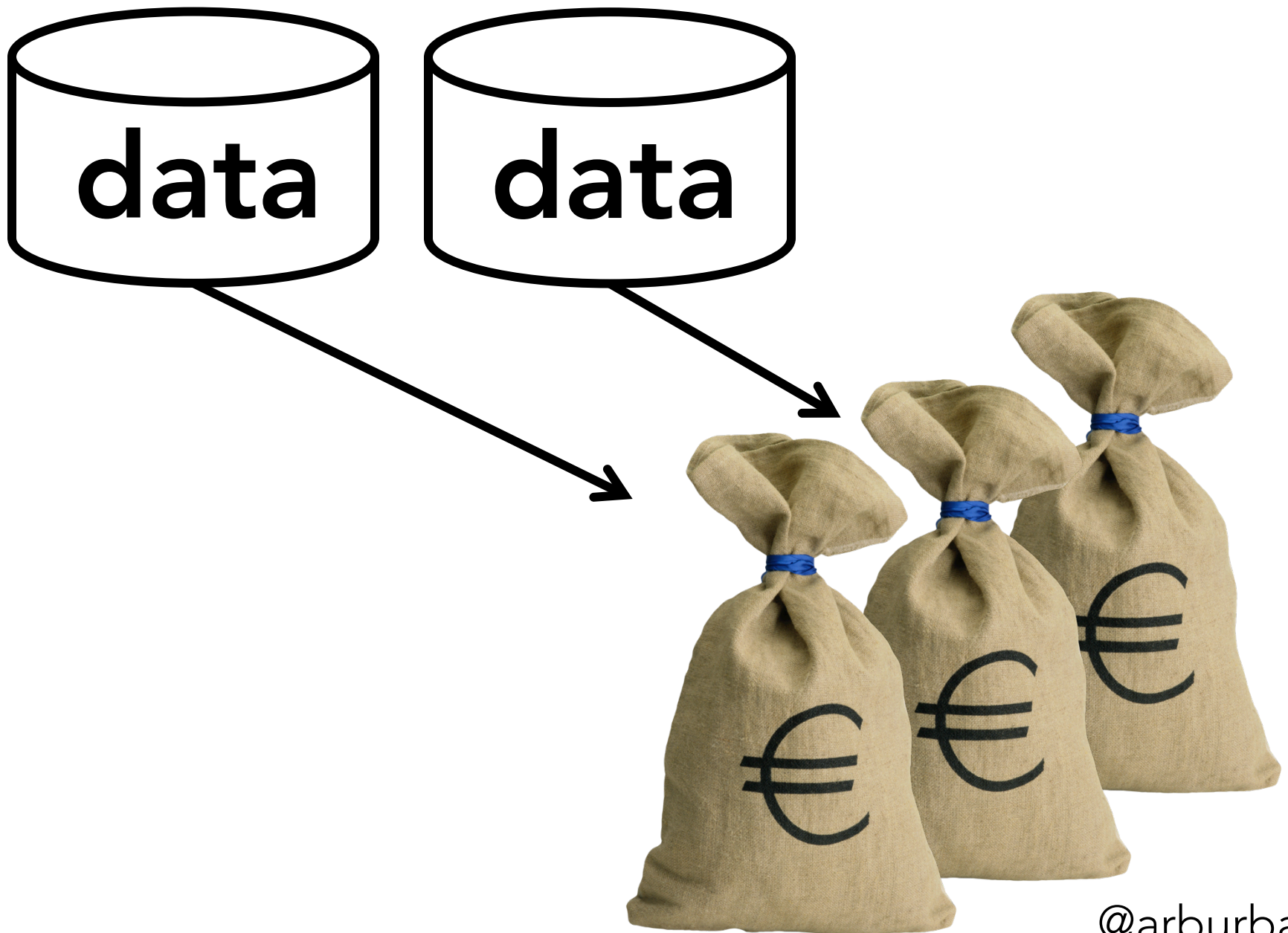
*scaling data science at Pinterest*

*Andrea Burbank*

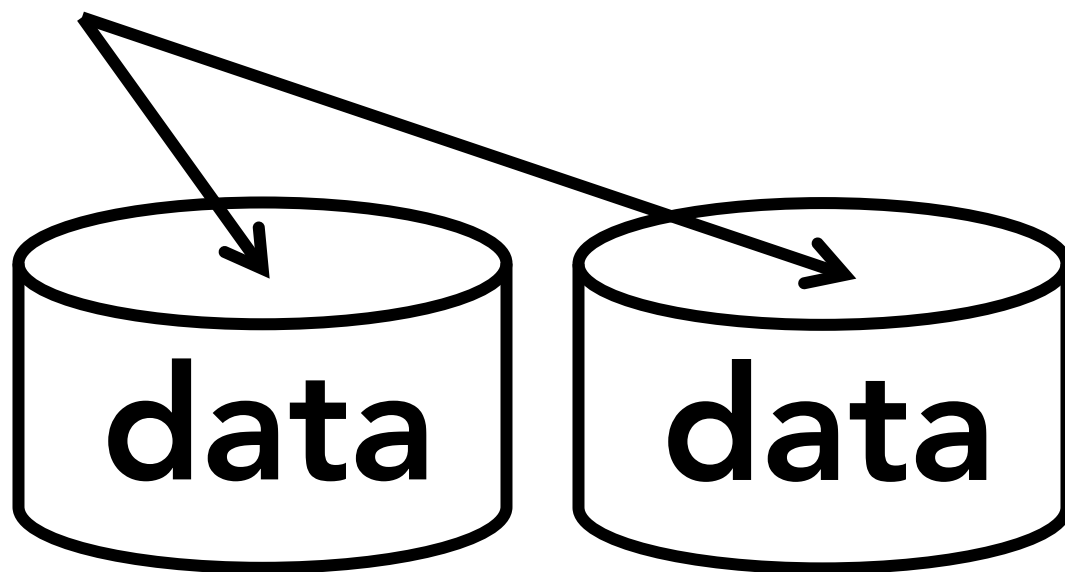
# Pinterest

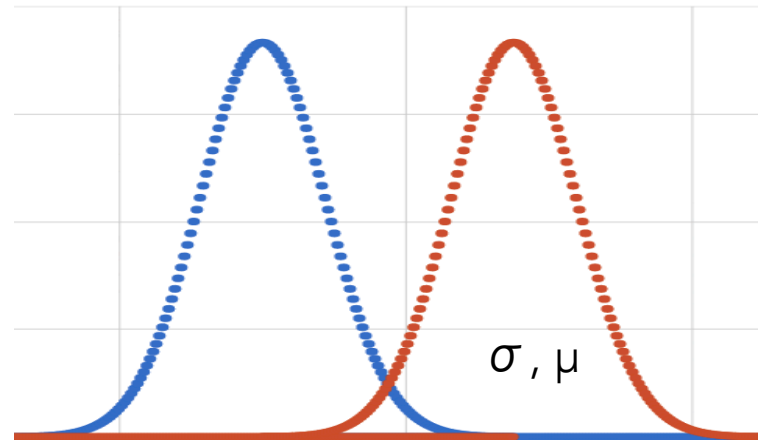


@arburbank



**data is not raw material**







# **Data as software as data**

*better software for better data*

*better data for better software*

# Outline

1. No logging at all
2. Logs, but no insights
3. Incorrect data due to external factors
4. Incorrect data due to internal factors
5. Data gets messy
6. Correlation and causation with data
7. Software can make data easier





# Part I: who is coming?

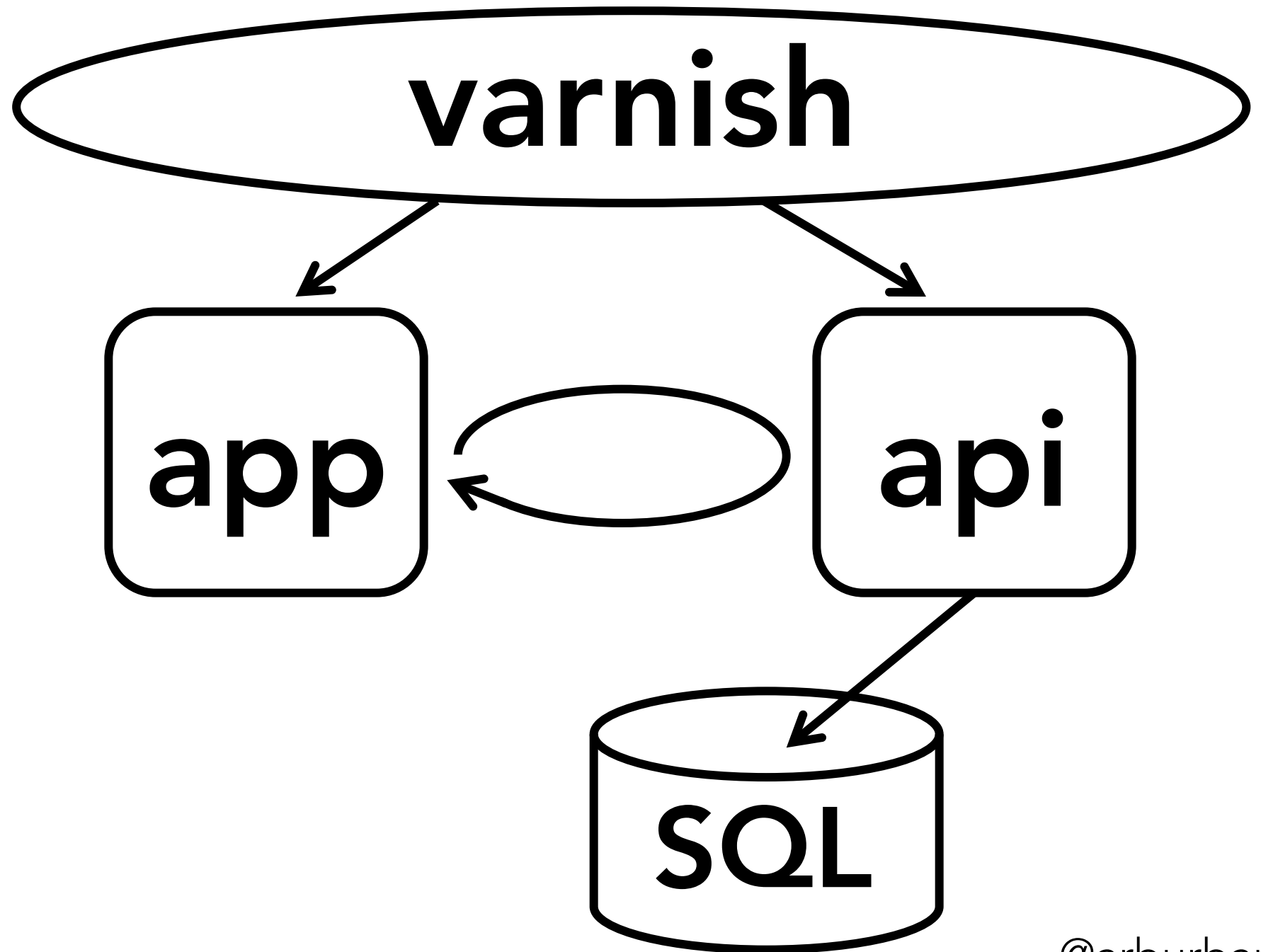
*counting is hard*

```
graph TD; Varnish([varnish]) --> app[app]; Varnish --> api[api]; app --> api; api --> app;
```

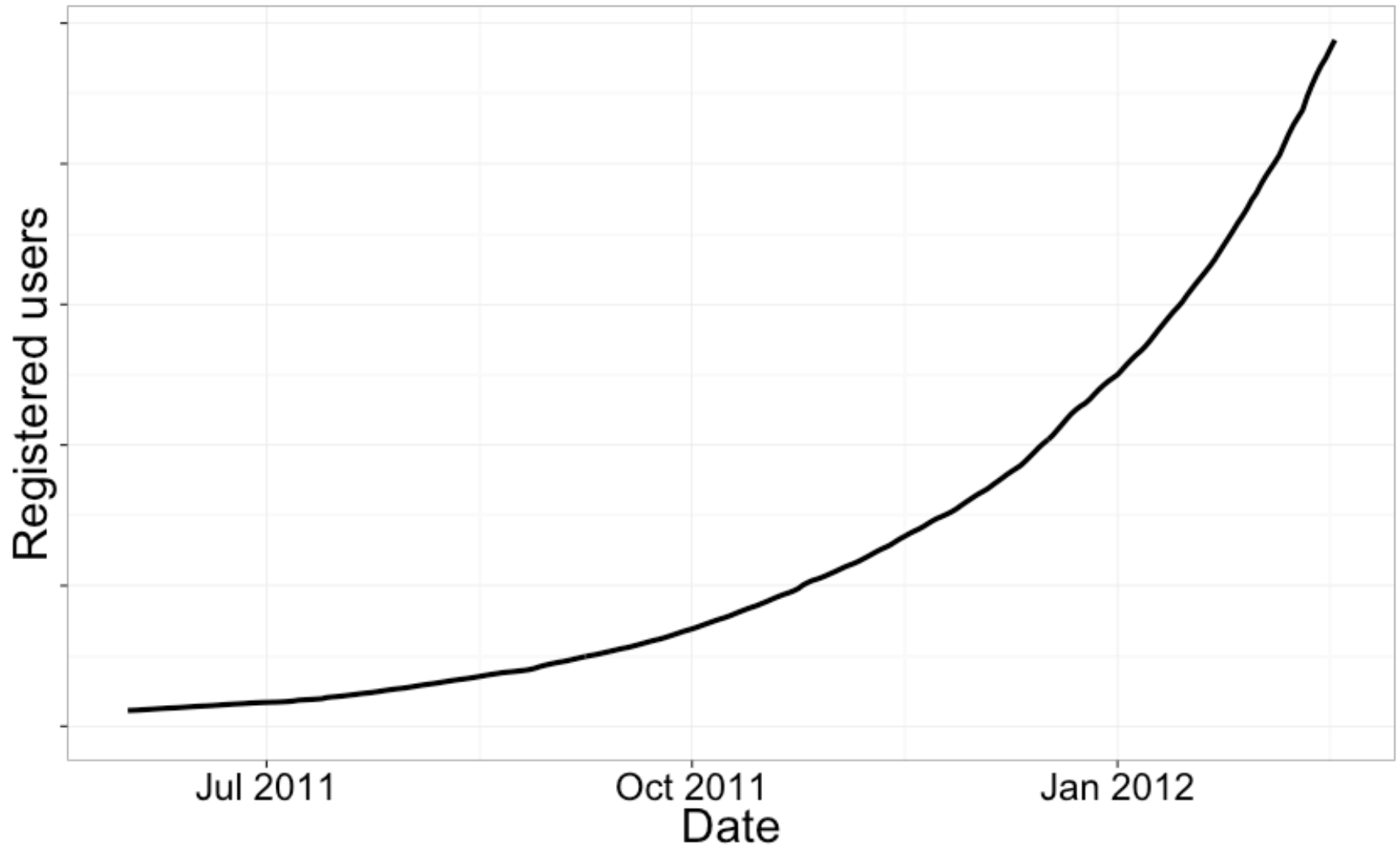
**varnish**

**app**

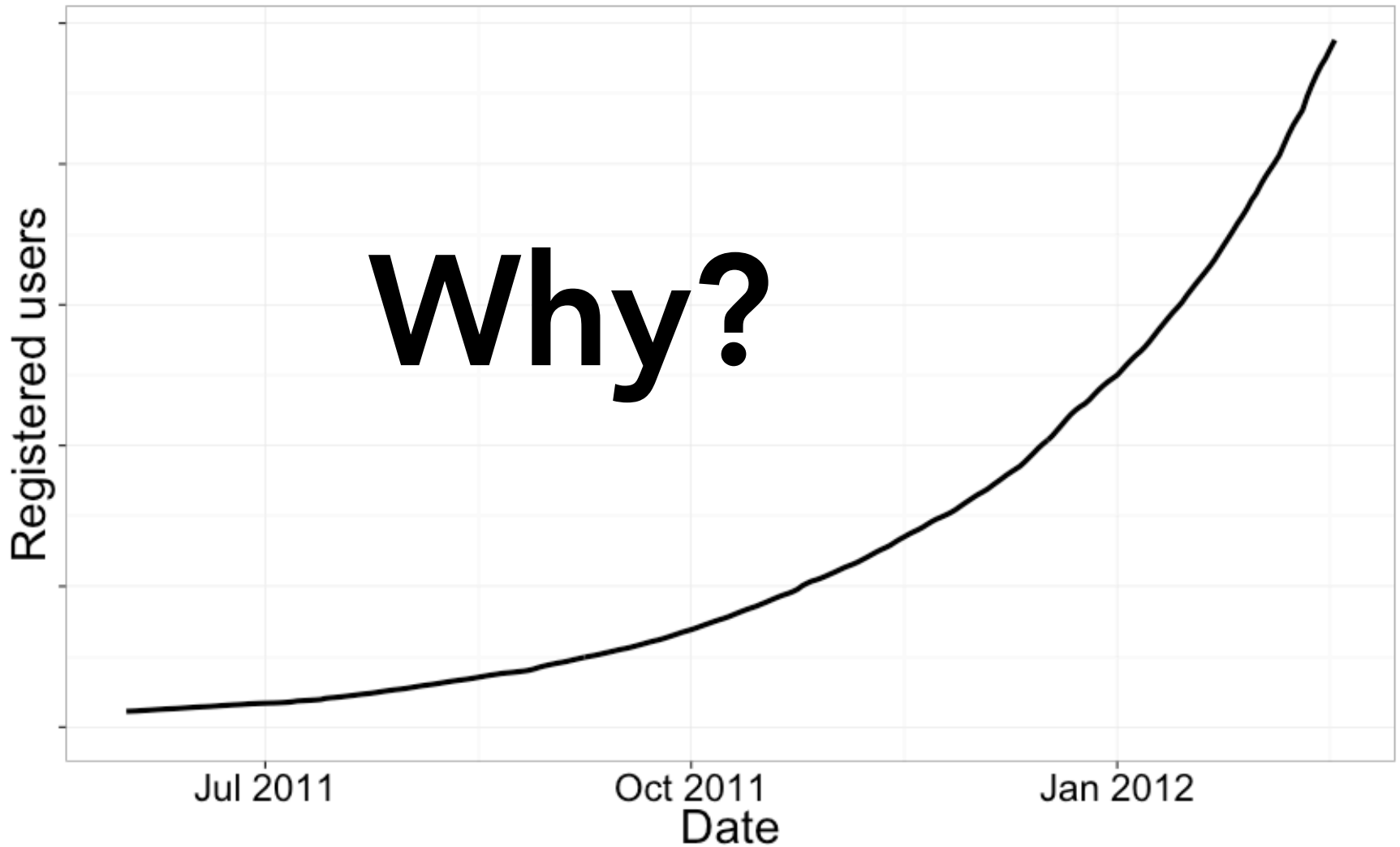
**api**

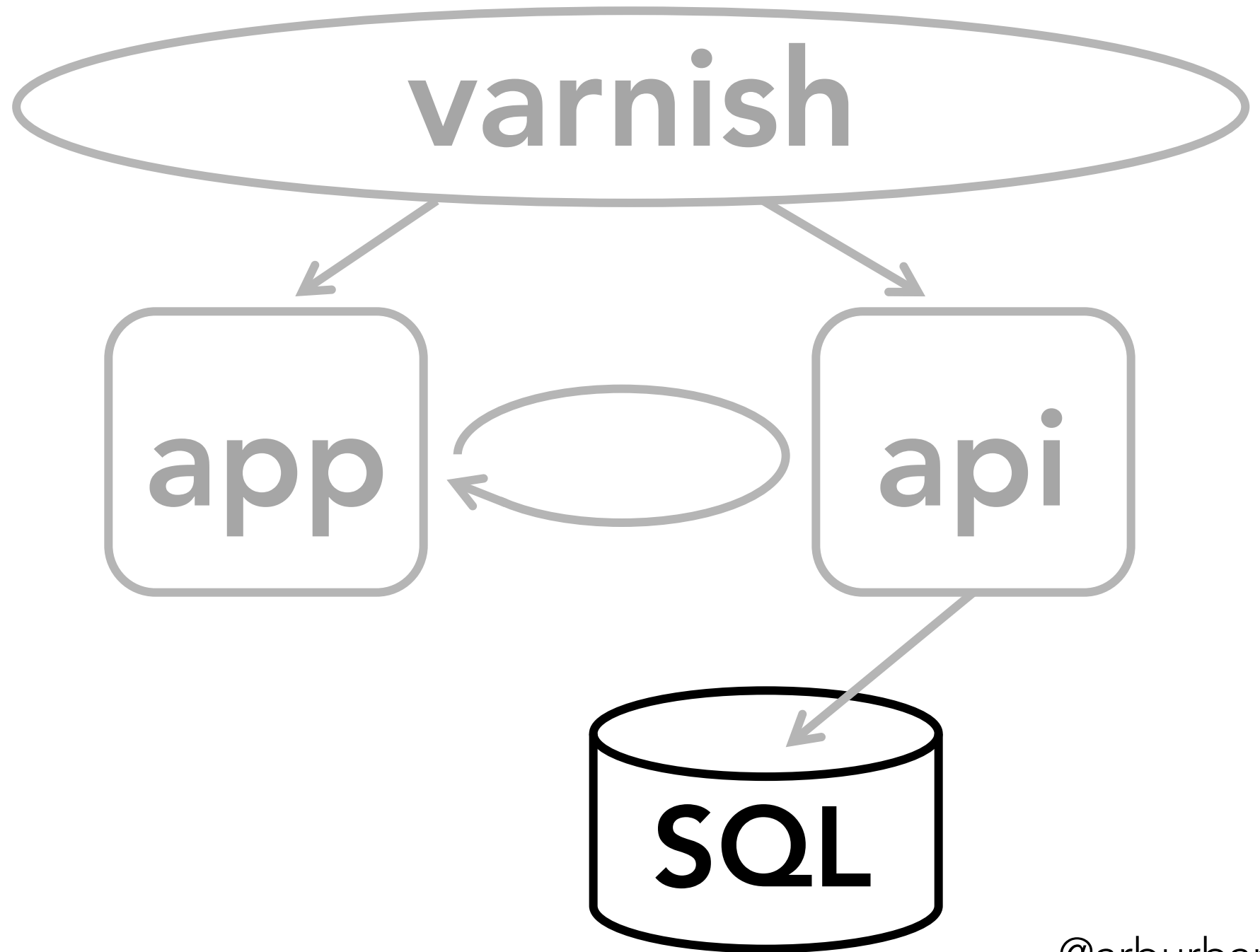


# Huge growth in 2011-2012



# Huge growth in 2011-2012







**first name**

**last name**

**email address**

**signup date & time**

**username**

**gender**

**email settings**



**first name**  
**last name**  
**email address**  
**signup date & time**  
**username**  
**gender**  
**email settings**

referrer  
landing page  
signup funnel  
pages viewed

?



# Why such growth?

- invite-only → more desirable?
- invite-only → more homogeneous?
- viral spread through a small community?
- lots of traffic from Facebook?
- extensive press coverage?

# Daily Active Users

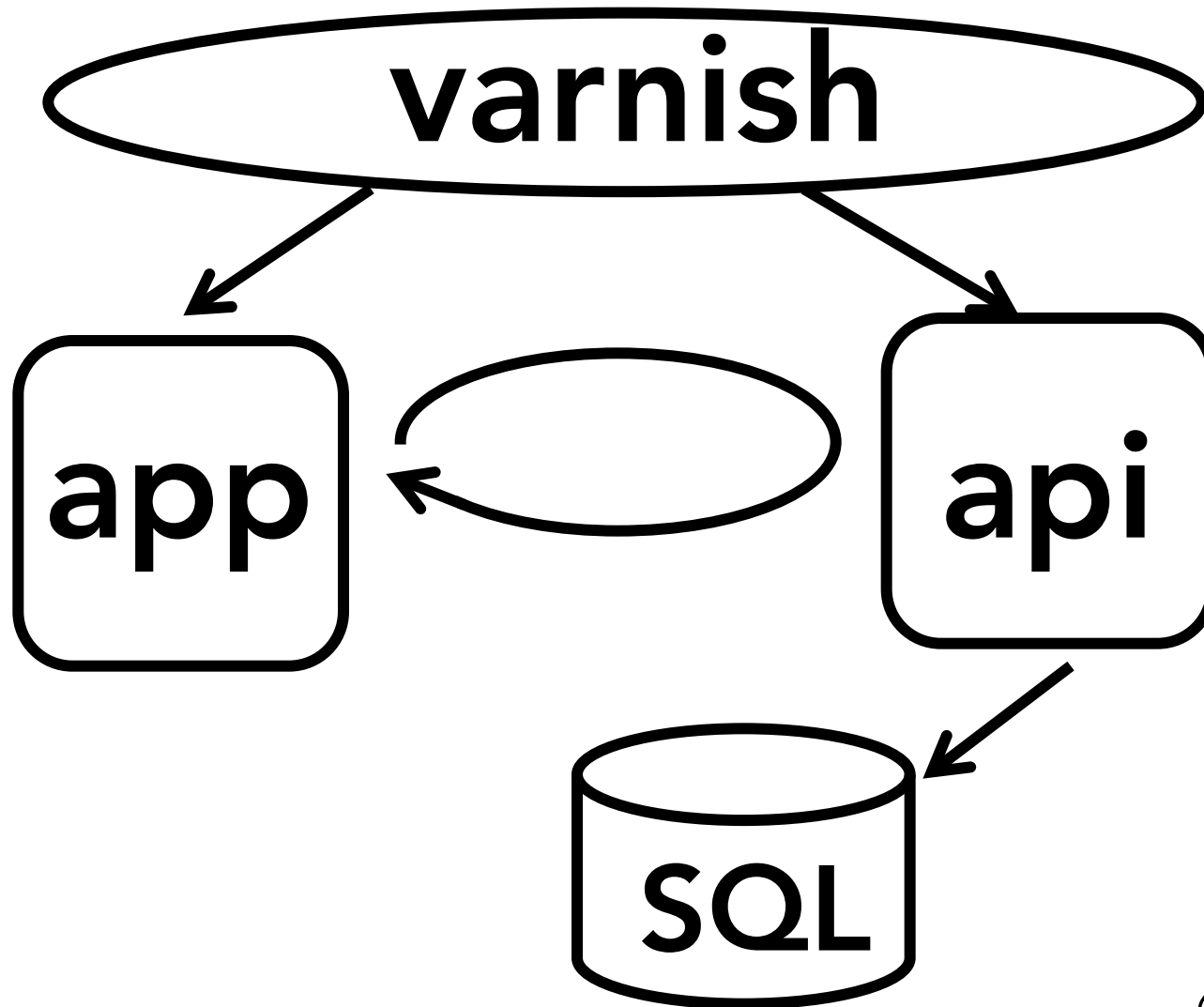
## Daily active users

---

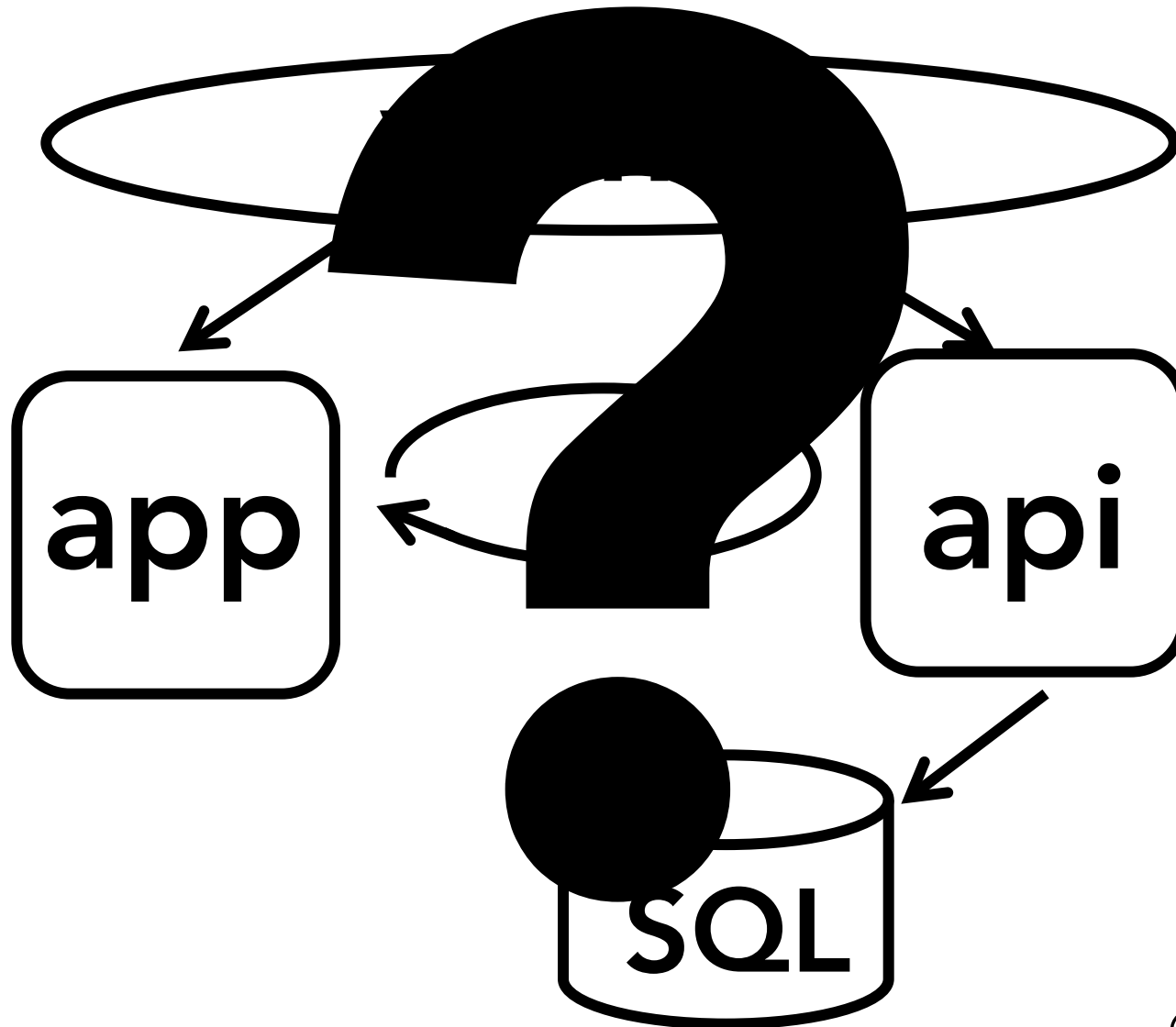
From Wikipedia, the free encyclopedia

**Daily active users** (DAU) is one of the ways used for [measuring](#) success of an [internet](#) product, e.g., [online social games](#).<sup>[1]</sup> DAU measures the "stickiness" of an online product by answering the question, "How many [unique users](#) visit the site daily?" Usually the only requirement for a user to be considered "active" is that they somehow view or engage with the product. Examples of this type of usage would range from visiting the [splash page](#) of a game or commenting on a post by the game's Facebook page, to actually playing the game itself.<sup>[2]</sup> The monthly aggregate of active users is [monthly active users](#) (MAU).

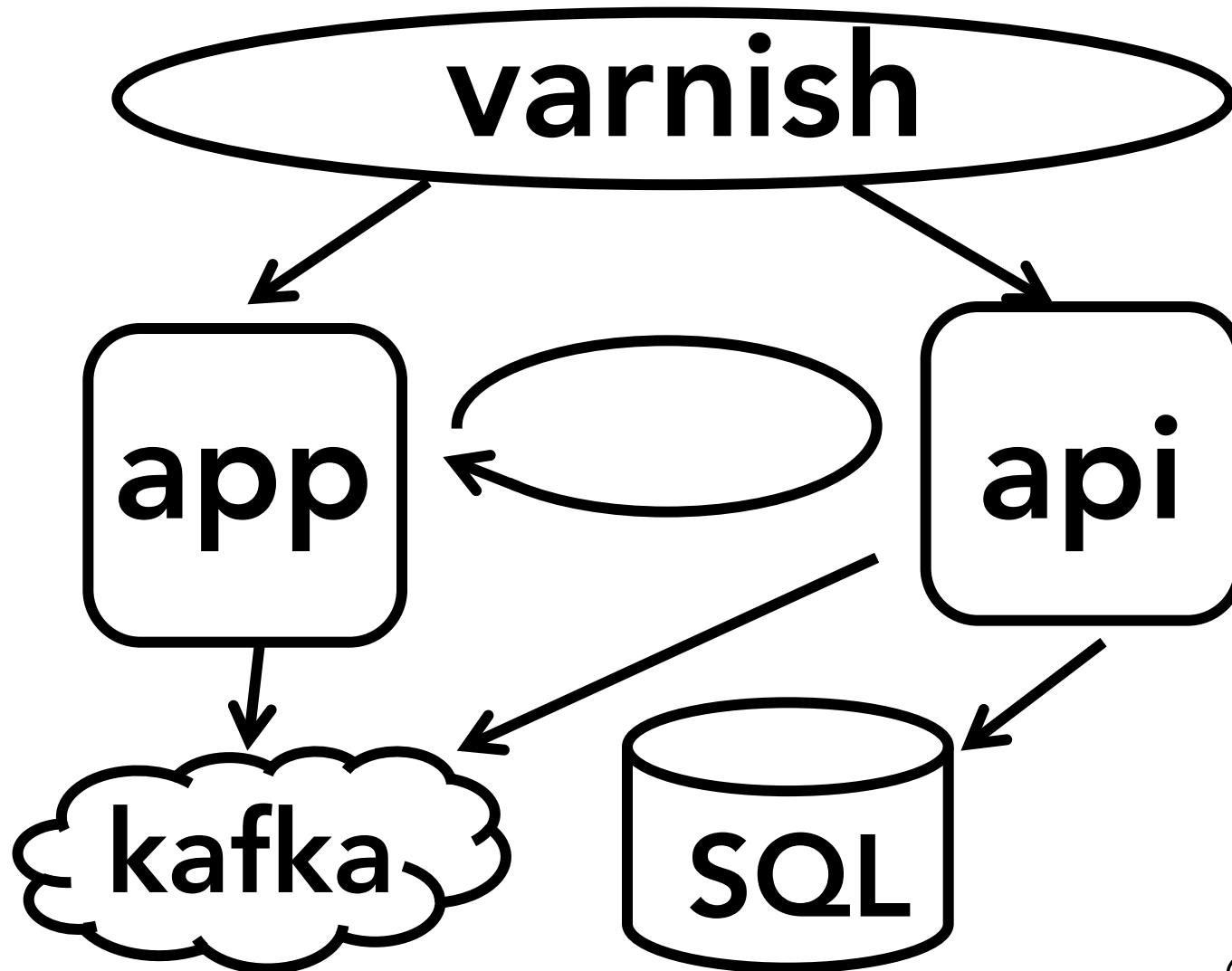
# Daily Active Users



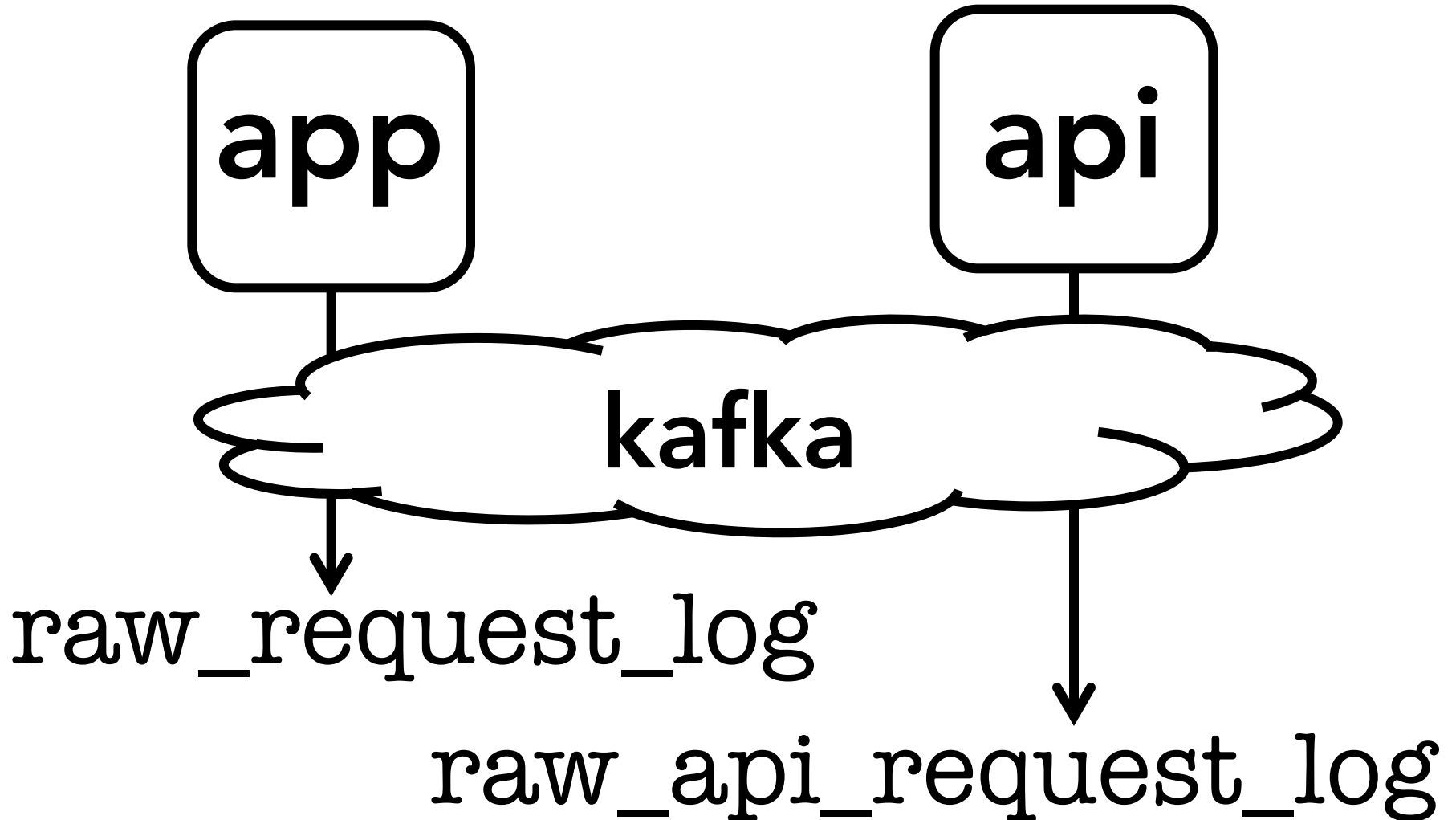
# Daily Active Users



# Daily Active Users



# Daily Active Users



# Daily Active Users

## raw\_request\_log

userid	ts	path	status	dt
3789823	08:37:03.41	/pin/123/	200	2012-06-01
3789823	08:37:07.22	/pin/repin/	200	2012-06-01
1724468	08:37:10.39	/user/8en/	200	2012-06-01
8779233	08:37:11.97	/category/art/	200	2012-06-01

# Daily Active Users

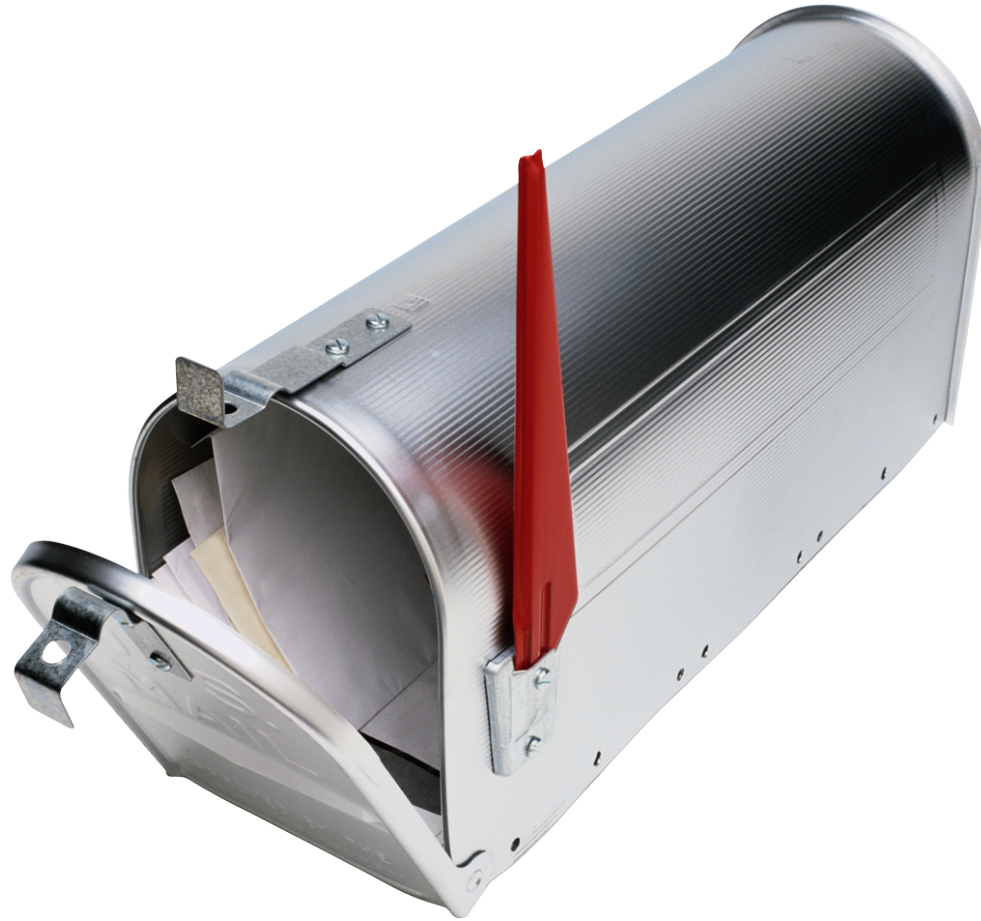
## raw\_request\_log

userid	ts	path	status	dt
3789823	08:37:03.41	/pin/123/	200	2012-06-01
3789823	08:37:07.22	/pin/repin/	200	2012-06-01
1724468	08:37:10.39	/user/8en/	200	2012-06-01
8779233	08:37:11.97	/category/art/	200	2012-06-01

**DAU: users with a request**



# Daily Active Users



# Daily Active Users

## raw\_request\_log

userid	ts	path	status	dt
3789823	08:37:03.41	/pin/123/	200	2012-06-01
3789823	08:37:07.22	/pin/repin/	200	2012-06-01
1724468	08:37:10.39	/user/8en/	200	2012-06-01
8779233	08:37:11.97	/category/art/	200	2012-06-01
4980307	08:37:12.38	/email/tracking.gif	200	2012-06-01

# Daily Active Users

## raw\_request\_log

userid	ts	path	status	dt
3789823	08:37:03.41	/pin/123/	200	2012-06-01
3789823	08:37:07.22	/pin/repin/	200	2012-06-01
1724468	08:37:10.39	/user/8en/	200	2012-06-01
8779233	08:37:11.97	/category/art/	200	2012-06-01
4980307	08:37:12.38	/email/tracking.gif	200	2012-06-01

# Daily Active Users

## raw\_request\_log

userid	ts	path	status	dt
3789823	08:37:03.41	/pin/123/	200	2012-06-01
3789823	08:37:07.22	/pin/repin/	200	2012-06-01
1724468	08:37:10.39	/user/8en/	200	2012-06-01
8779233	08:37:11.97	/category/art/	200	2012-06-01
<del>4980307</del>	<del>08:37:12.38</del>	<del>/email/tracking.gif</del>	<del>200</del>	<del>2012-06-01</del>

**DAU: users with certain requests**

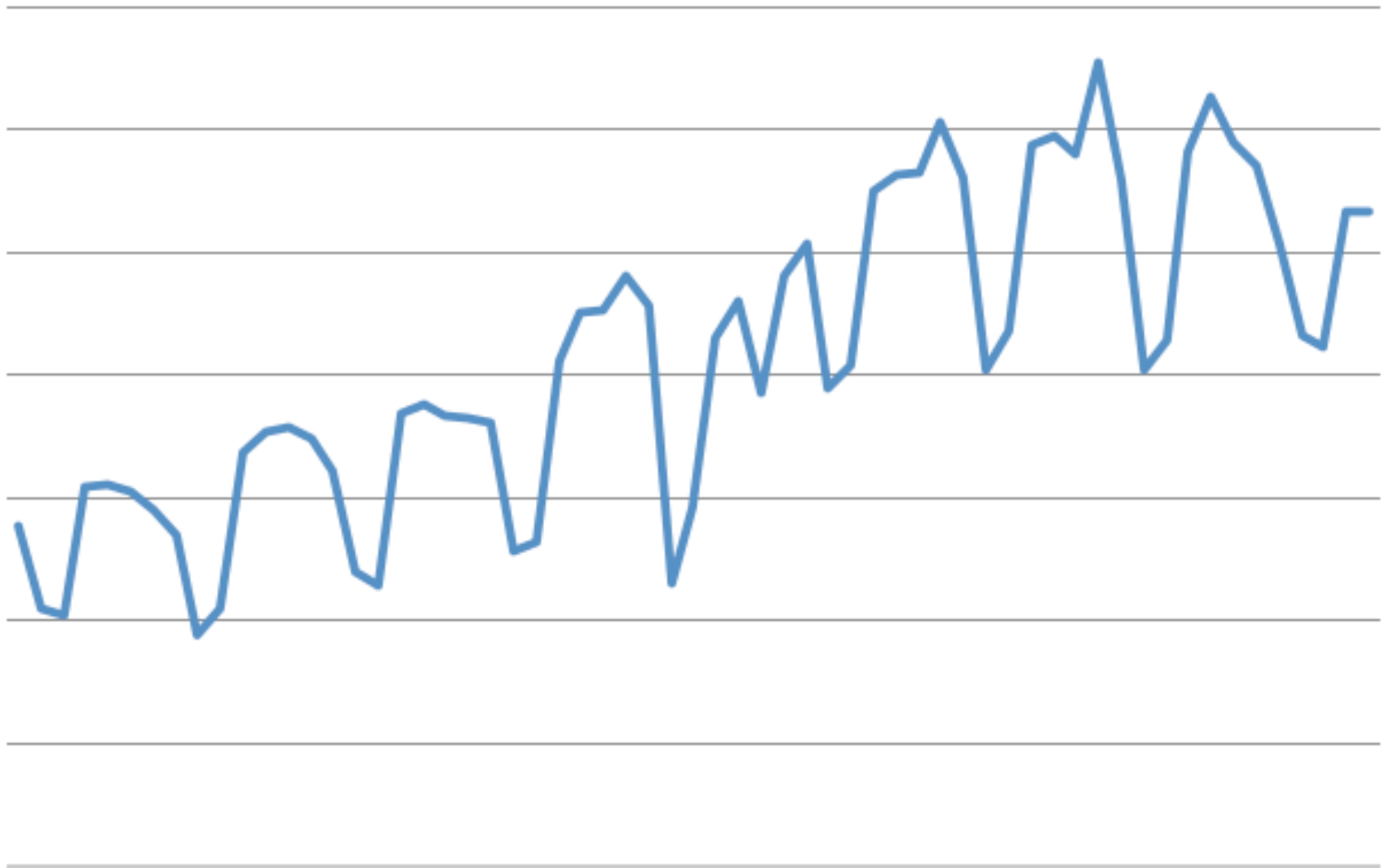
@arburbank

# Daily Active Users

```
SELECT userid,  
       count(*)  
FROM auth_compact_view  
WHERE dt="%(end_date)s"  
AND path not like "/email/tracking.gif%%"  
GROUP BY userid, dt;
```

**DAU: users with certain requests**

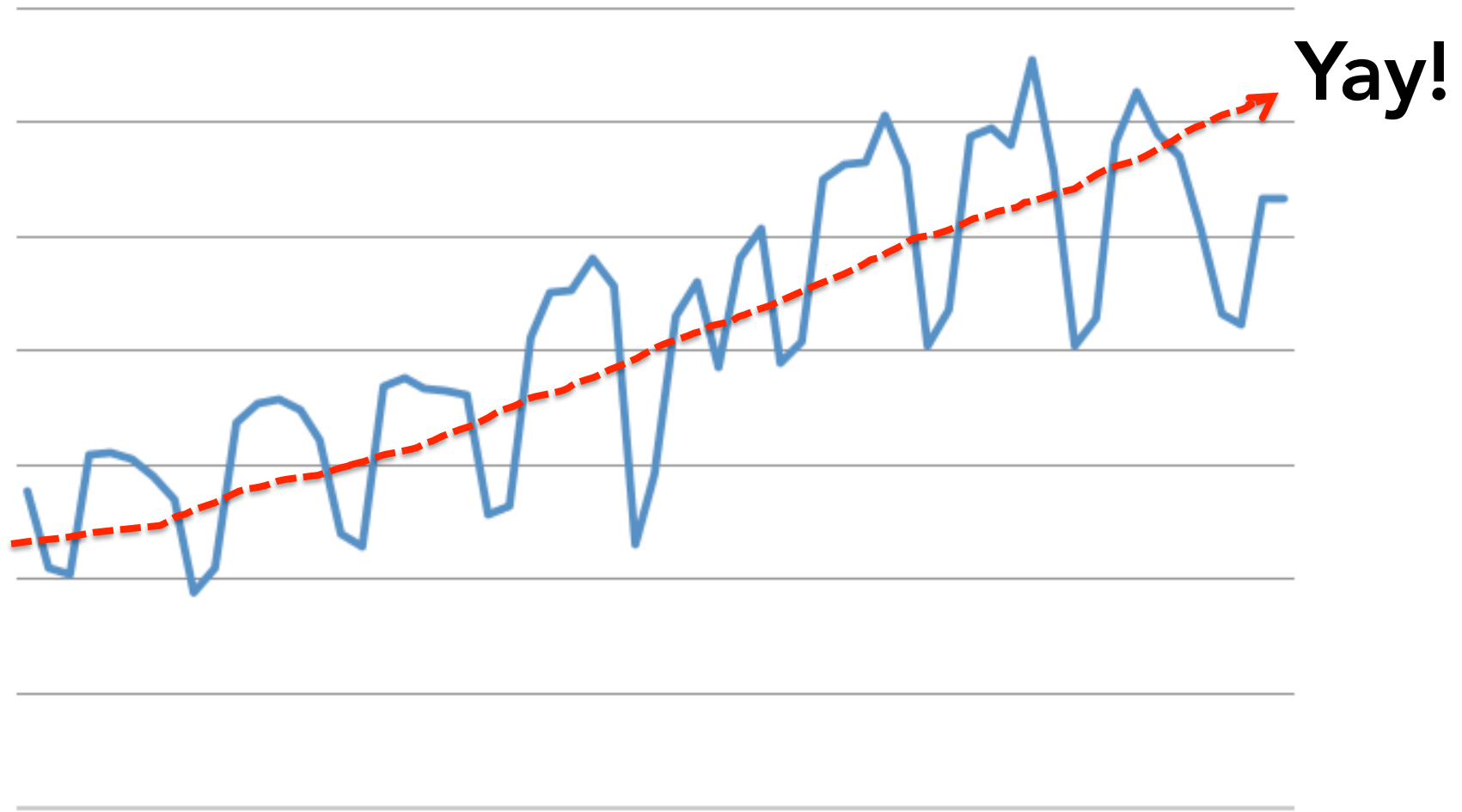
# Daily Active Users



# Daily Active Users



# Daily Active Users





# Who are those users?

**Who are those users?**

**Is it the same users every day?**

**Who are those users?**

**Is it the same users every day?**

**Or new users trying us out?**

**Who are those users?**

**Is it the same users every day?**

**Or new users trying us out?**

**Do people come back often?**

# Weekly Active Users

# Weekly Active Users

```
SELECT userid,  
       count(*)  
FROM auth_compact_view  
WHERE dt > DATE_SUB("%(end_date)s", 7)  
AND dt <= "%(end_date)s"  
AND path not like "/email/tracking.gif%%"  
GROUP BY userid, dt;
```

# Monthly Active Users

```
SELECT userid,  
       count(*)  
FROM auth_compact_view  
WHERE dt >= DATE_SUB("%(end_date)s", 28)  
AND dt <= "%(end_date)s"  
AND path not like "/email/tracking.gif%%"  
GROUP BY userid, dt;
```

# Yearly Active Users

```
SELECT userid,  
       count(*)  
FROM auth_compact_view  
WHERE dt >= DATE_SUB("%(end_date)s", 365)  
AND dt <= "%(end_date)s"  
AND path not like "/email/tracking.gif%%"  
GROUP BY userid, dt;
```



# Yearly Active Users

```
SELECT userid,  
       count(*)  
FROM auth_com view  
WHERE dt >= DATE_SUB("%(end_date)s", 365)  
AND dt <= "%(end_date)s"  
AND path not like 'mail/tracking.gif%%'  
GROUP BY userid, dt;
```

# Derived tables: DAU

```
INSERT OVERWRITE TABLE unique_actions  
  partition(dt= "%(end_date)s", action_type=0)
```

```
SELECT userid,  
       count(*) num_actions  
FROM auth_compact_view  
WHERE dt="%(end_date)s"  
AND path not like "/email/tracking.gif%%"  
GROUP BY userid, dt;
```

# Derived tables: DAU

## unique\_actions

userid	num_actions	action_type	dt
3789823	27	0	2012-06-01
1724468	135	0	2012-06-01
8779233	1	0	2012-06-01
2987345	234	0	2012-06-01
9873001	12	0	2012-06-01
8244108	87	0	2012-06-01
4027394	43	0	2012-06-01

**Who are those users?**

**Is it the same users every day?**

**Or new users trying us out?**

**Do people come back often?**

# Derived tables: WAU

```
INSERT OVERWRITE TABLE xd7_users
  partition(dt= "%(end_date)s", action_type=0)

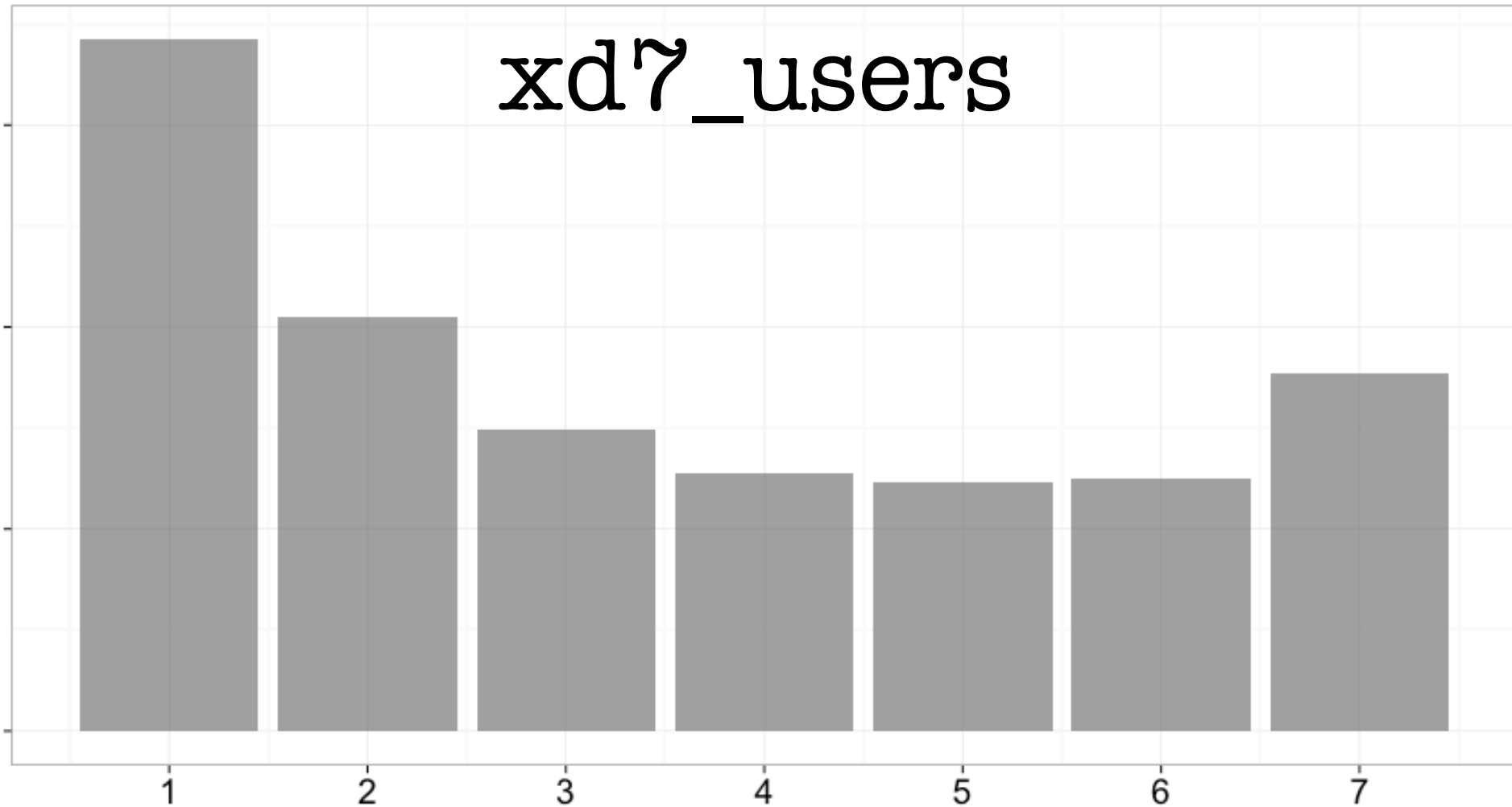
SELECT userid,
       count(*) num_days_active,
       sum(num_actions) num_actions
FROM unique_actions
WHERE dt > DATE_SUB("%(end_date)s", 7)
AND dt <= "%(end_date)s"
GROUP BY userid;
```

# Derived tables: MAU

```
INSERT OVERWRITE TABLE xd28_users
  partition(dt= "%(end_date)s", action_type=0)

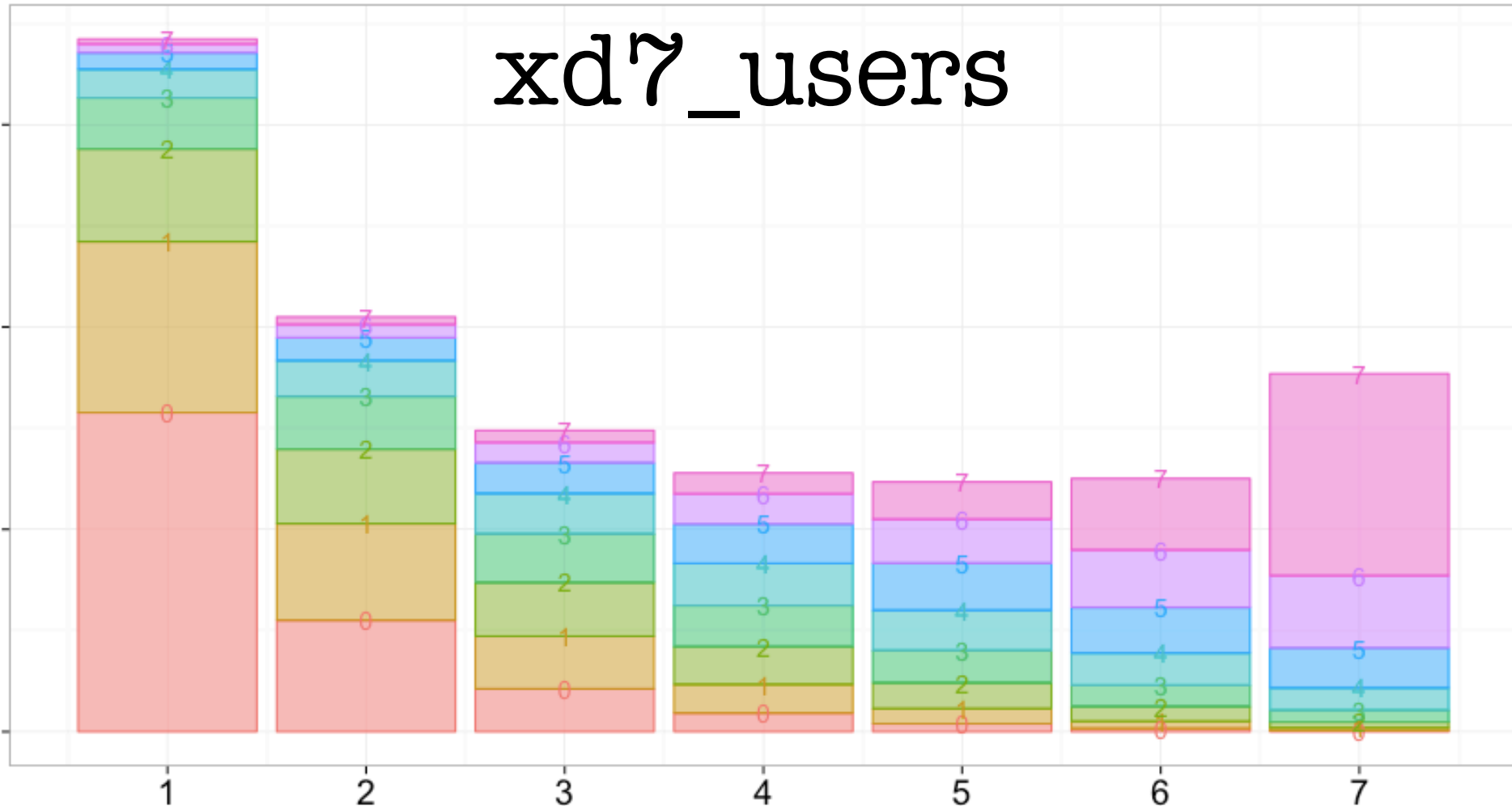
SELECT userid,
       count(*) num_days_active,
       sum(num_actions) num_actions
FROM unique_actions
WHERE dt > DATE_SUB("%(end_date)s", 28)
AND dt <= "%(end_date)s"
GROUP BY userid;
```

# How often do users visit?



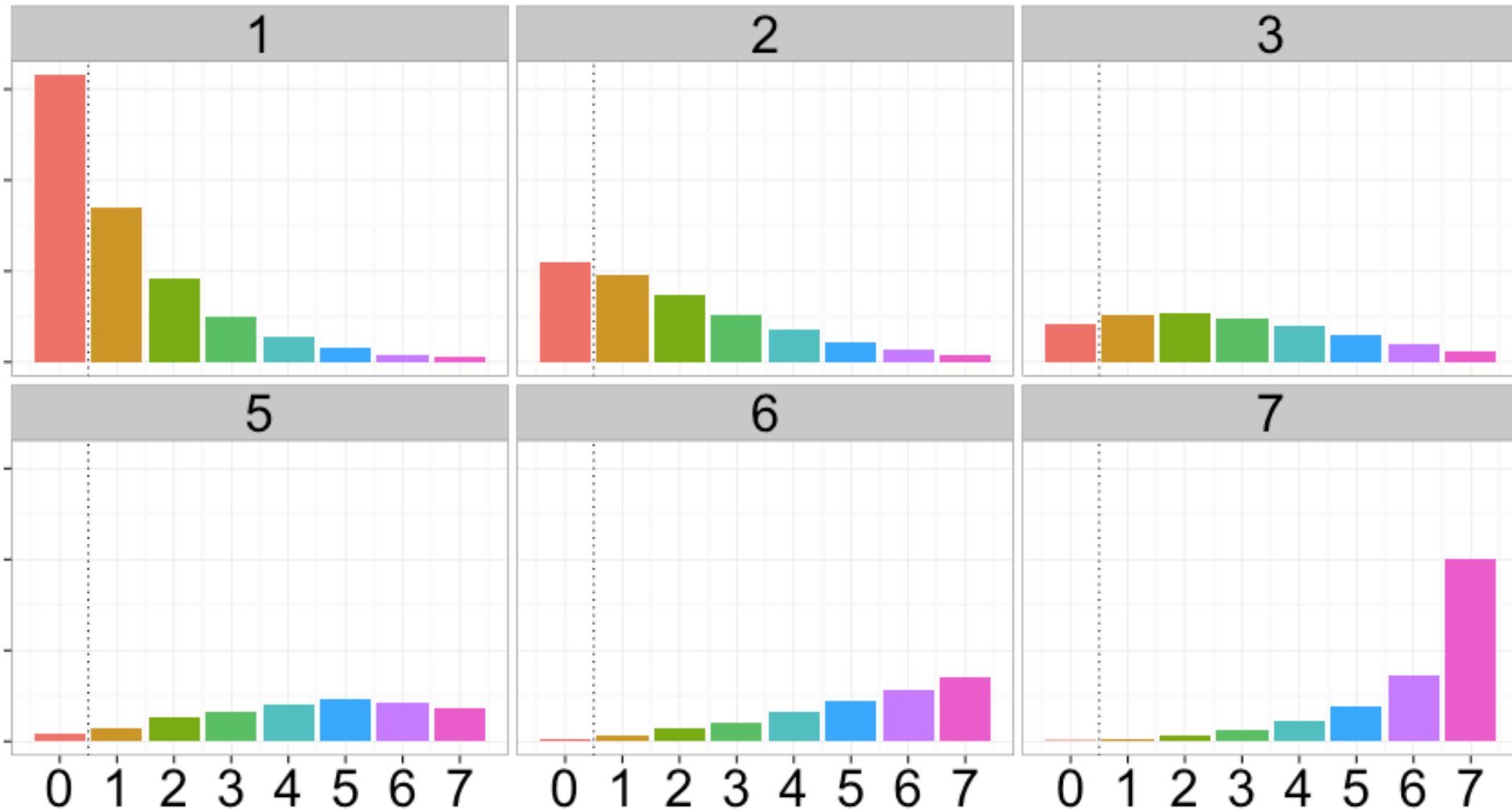
# How is next week different?

xd7\_users





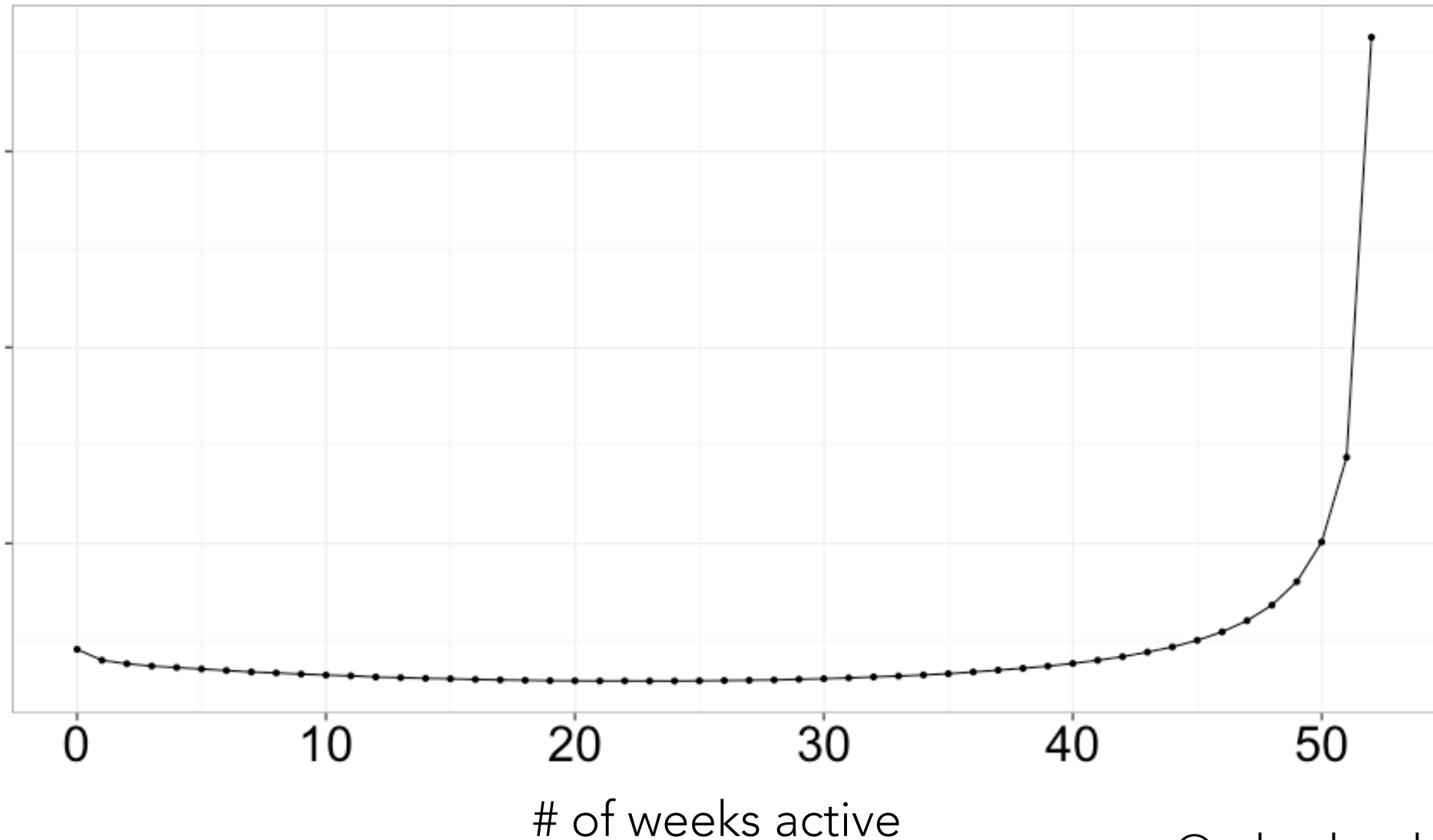
# How is next week different?



# What about next year?

```
select num_weeks_active, count(*) num_users
from
(select x.userid, count(*) num_weeks_active
 from
  (select dt, userid from xd7_users
   where dt='2015-04-15' and event_type=0) x
 join xd7_users y
 on x.userid = y.userid
 where y.dt >= '2015-04-15'
       and y.dt <= '2016-04-16'
       and (y.dt - x.dt) / 7 * 7 = y.dt - x.dt
       and y.event_type = 0
 group by x.userid) w
group by 1;
```

# What about next year?





**We can count users!**

**Pinterest is sticky**

AMSTERDAM

INTERNATIONAL  
SOFTWARE DEVELOPMENT  
CONFERENCE 2016

goto;  
conference

follow us on @GOTOamst

Workshop: June 13 / Conference: June 14-15

# data as software

@arburbank

# request logging



**which requests count?**

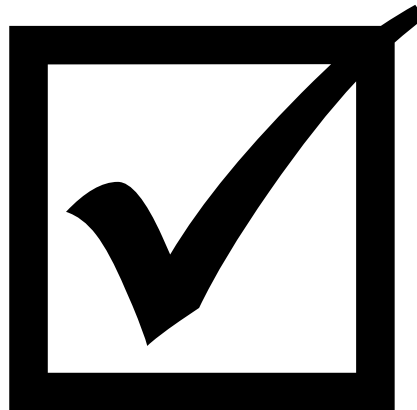
# derived tables

**xd7\_users / xd28\_users**



# Part I: who is coming?

*counting is hard*

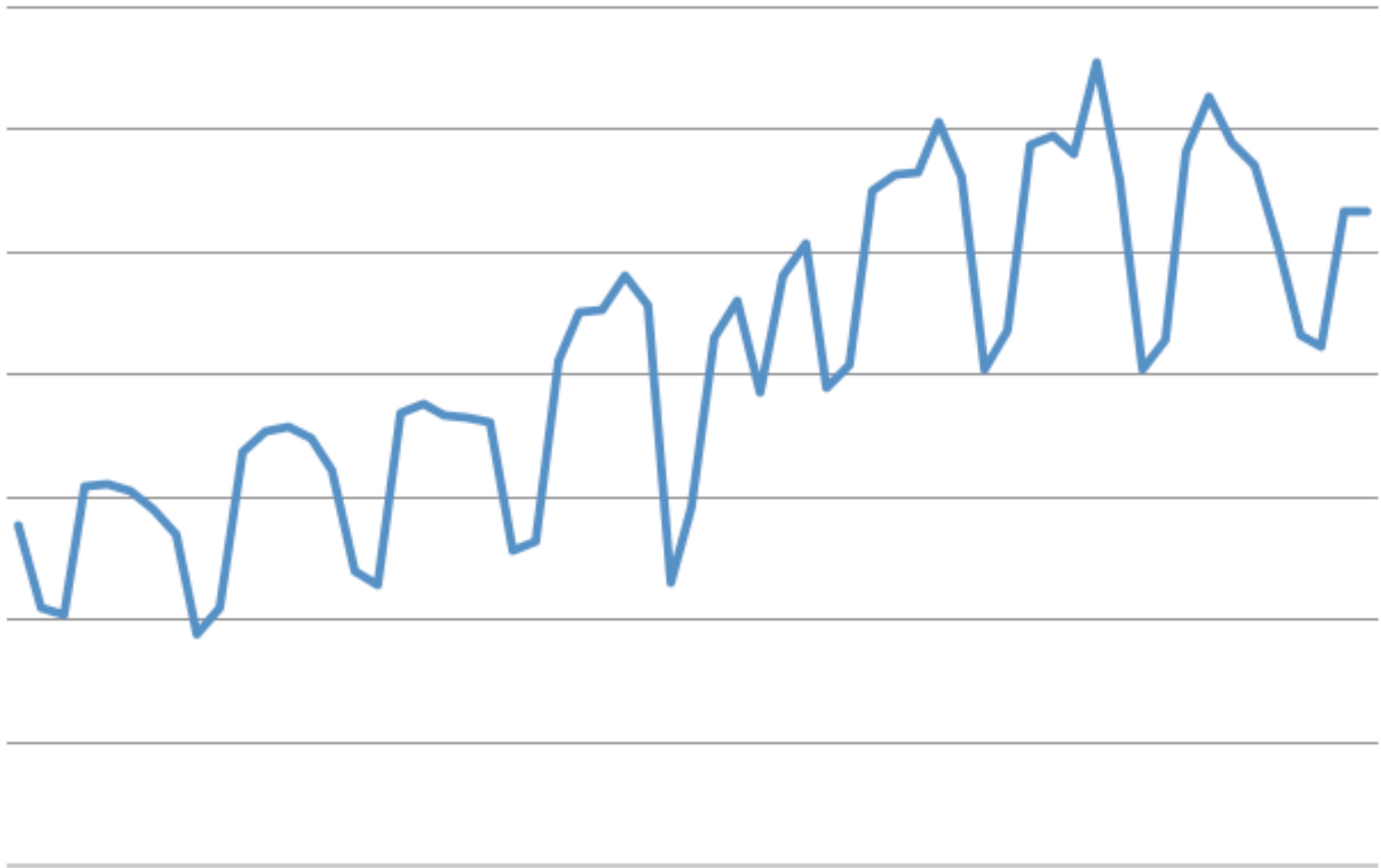




# Part II: how are we doing?

*counting becomes useful*

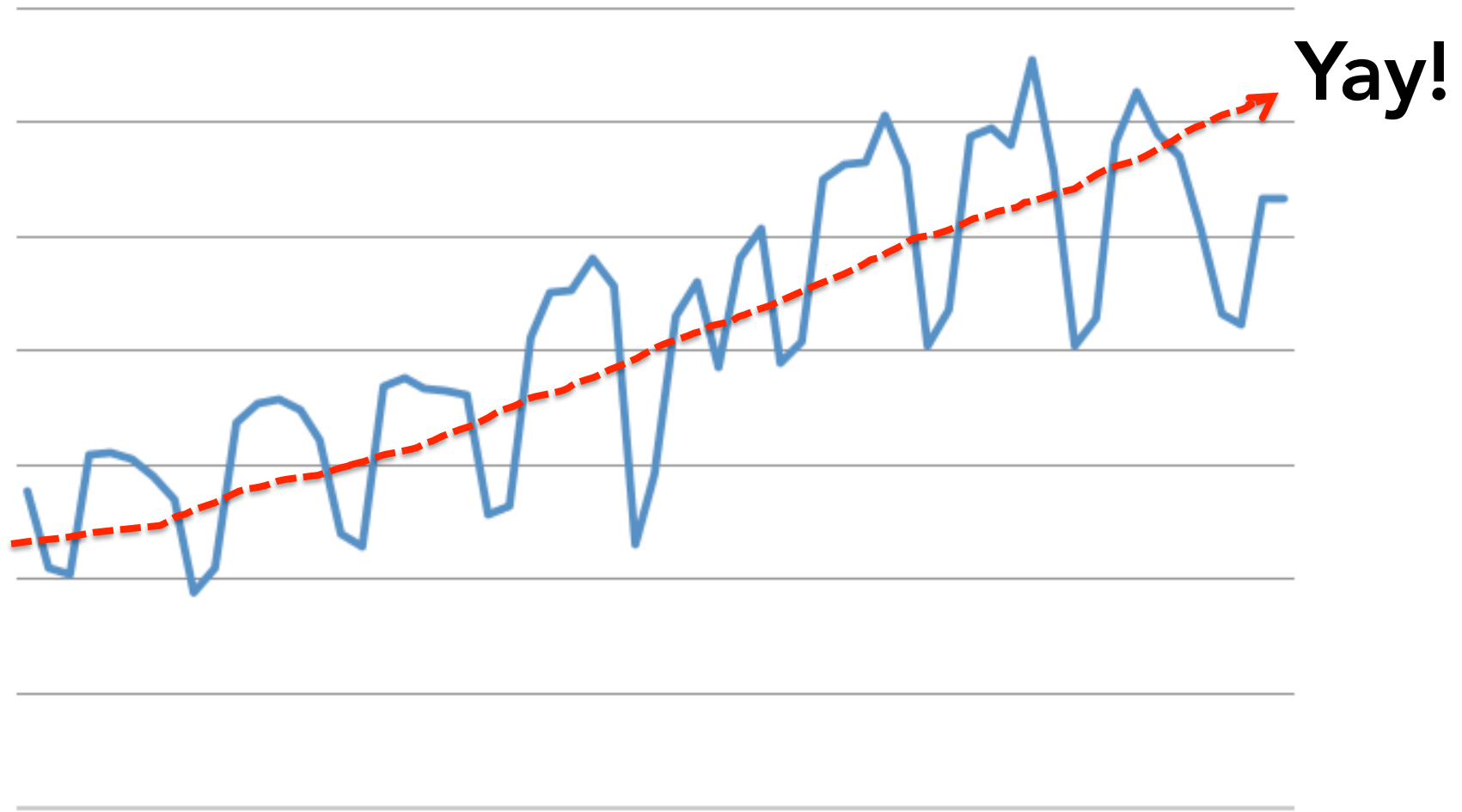
# Daily Active Users



# Daily Active Users



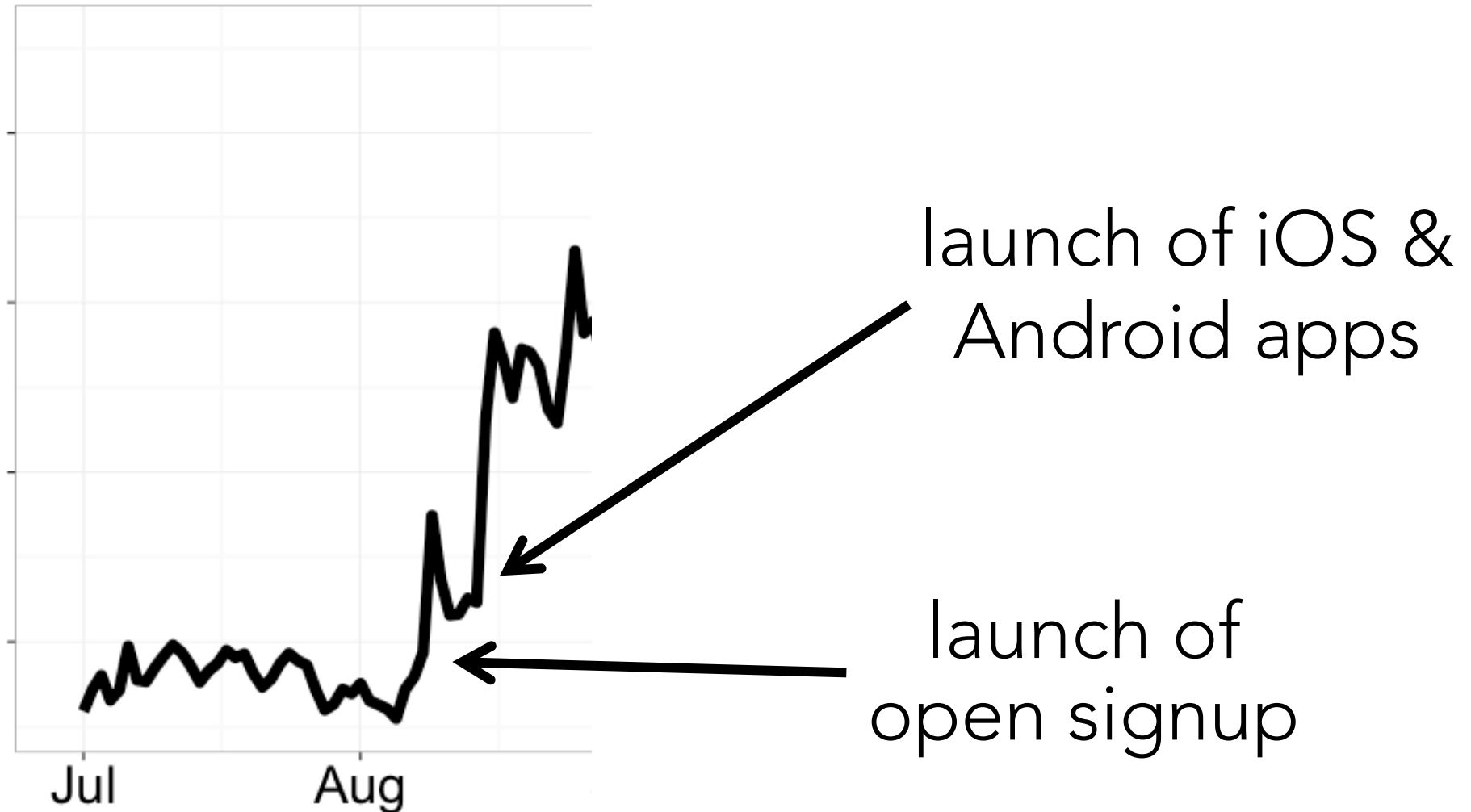
# Daily Active Users





# Signups, fall 2012

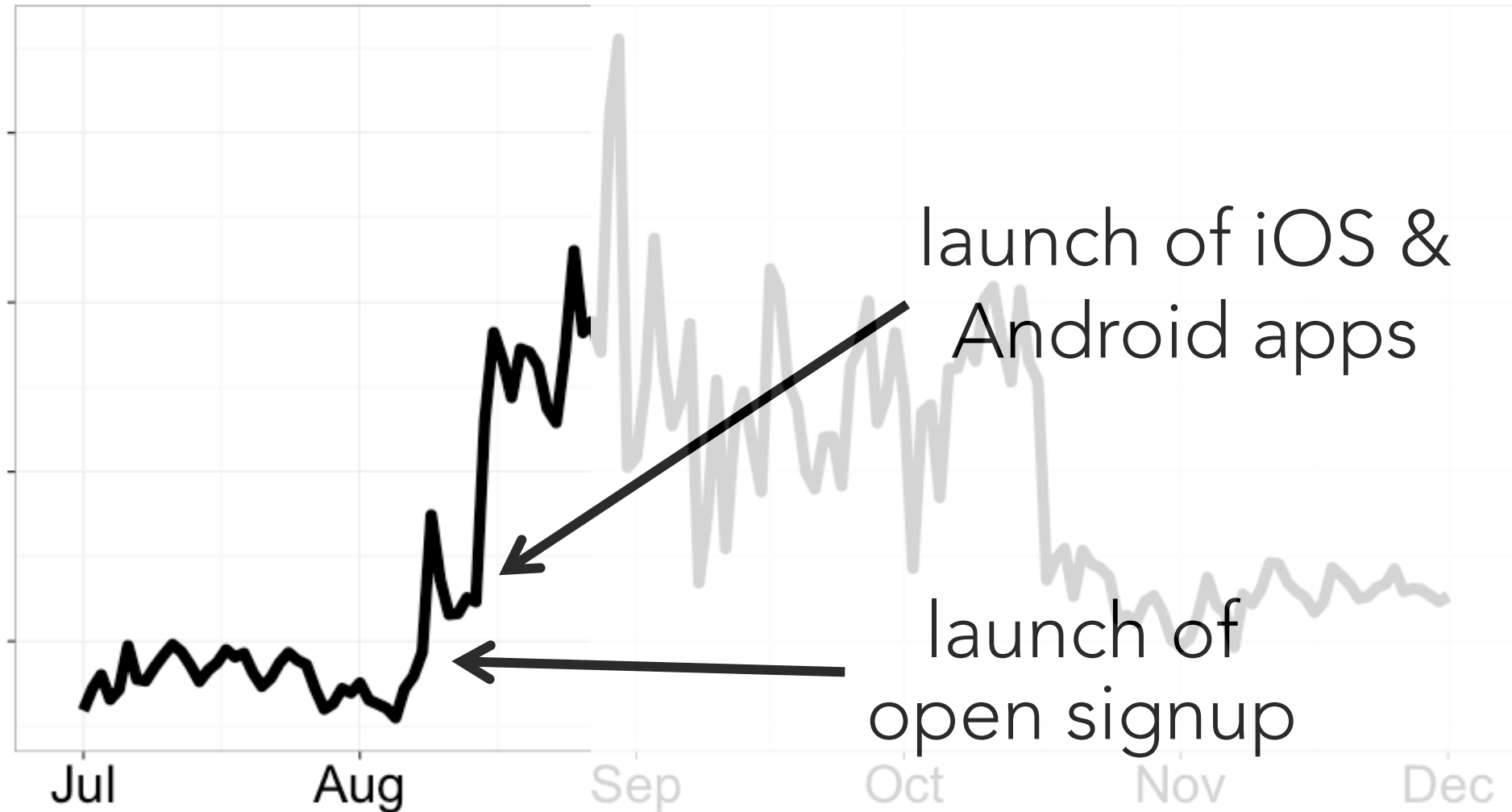
# Signups, fall 2012



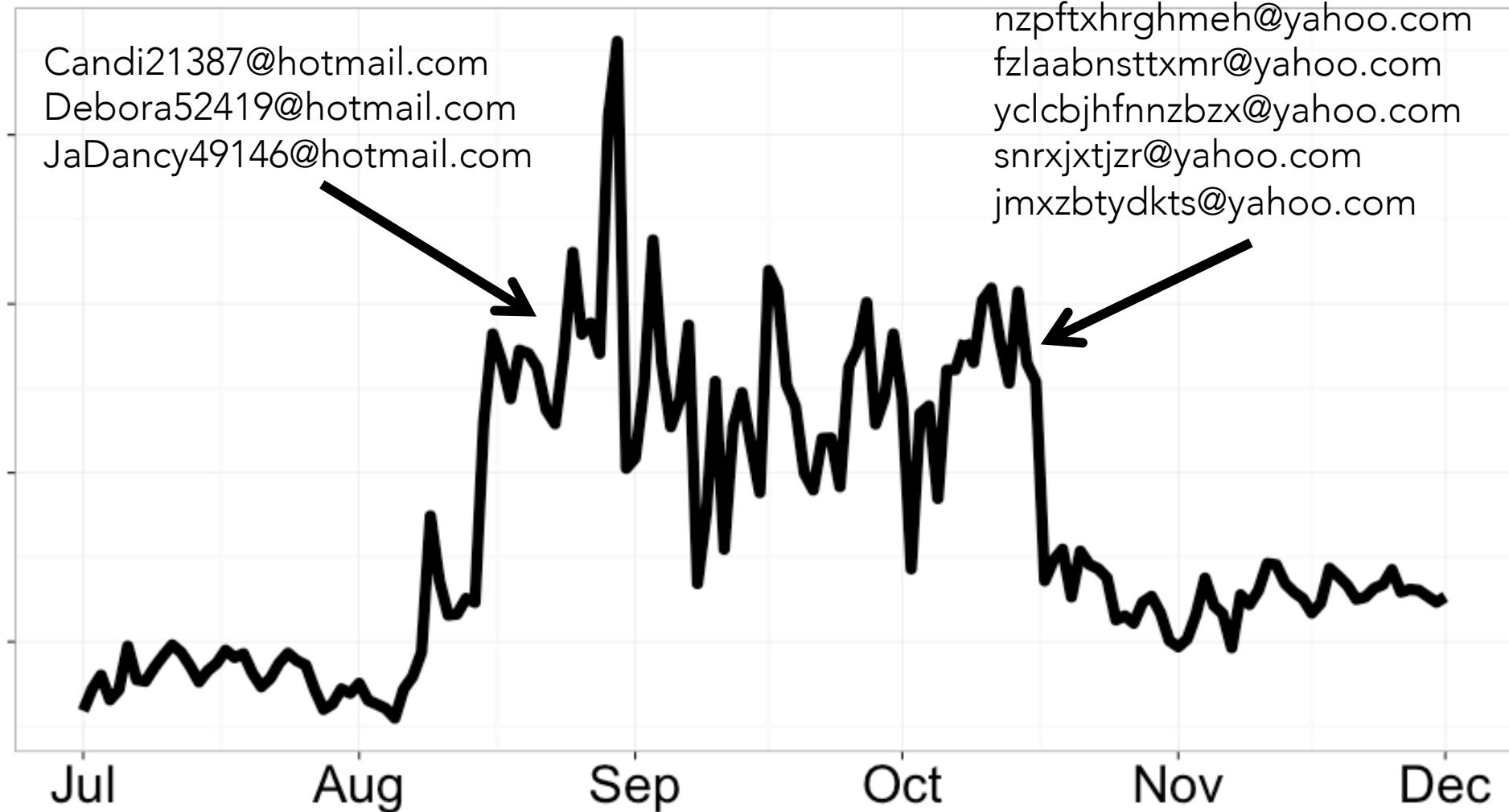
# Signups, fall 2012



# Signups, fall 2012

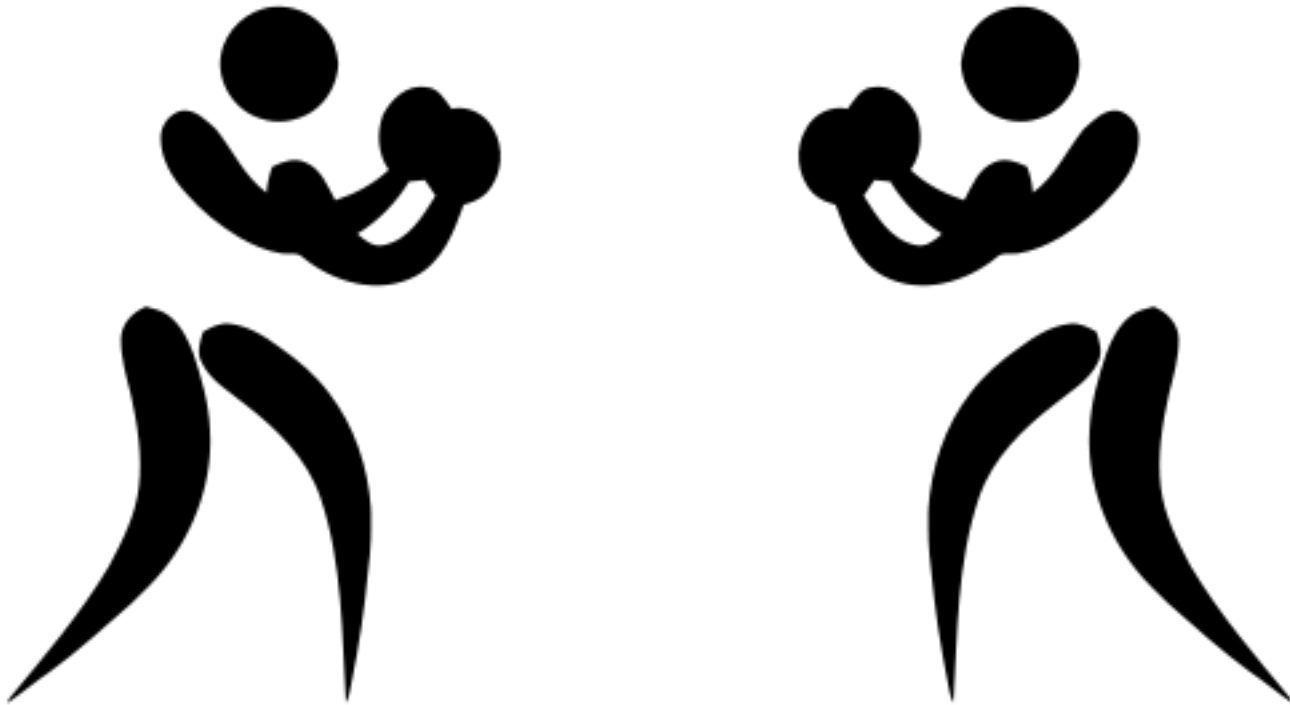


# Signups, fall 2012

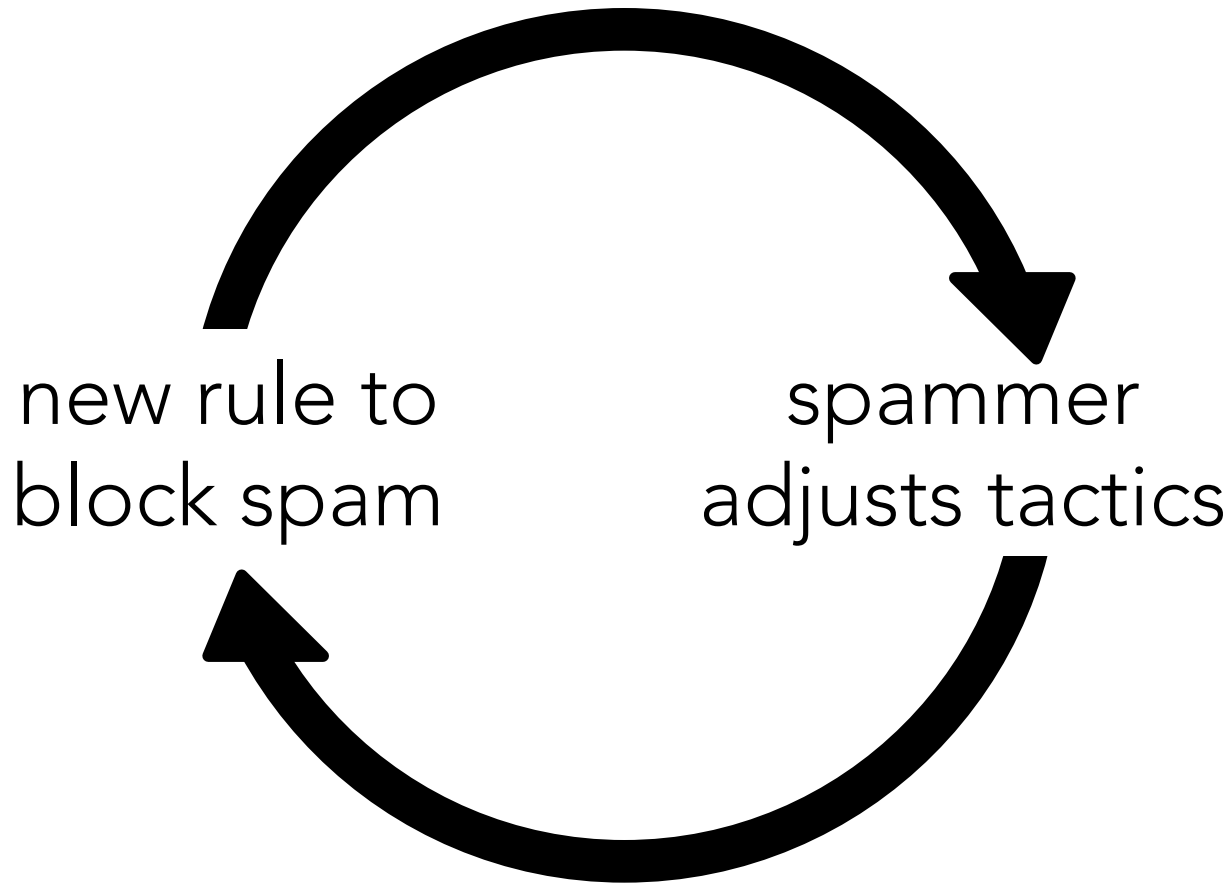


# SPAM

# Spam filtering



# Spam filtering







# Spam filtering

TiaKnauf7543@yahoo.com

CherrieBottin23120@hotmail.com

Delana64732@hotmail.com

DeaneFerrill45683@hotmail.com

ApoloniaChmela84292@hotmail.com

GlennieBalay99738@hotmail.com

DeettaGaraventa24915@hotmail.com

Nita42659@hotmail.com

Vernetta47911@hotmail.com

Soon93012@hotmail.com

BabetteEcheverri85476@hotmail.com

NichelleZych24319@hotmail.com

JeanetteKelton56943@hotmail.com

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

-I was born botanical, with the soul of an animal.

@arburbank

# Spam filtering

qhrwxfbzbyz@aol.com

szogfdvuyn@aol.com

gxqxdaoos@aol.com

qkjmzps9b0@aol.com

aeaagstwea@aol.com

wqlfboijaov@aol.com

dvdld3n@aol.com

znklexxtuhu@aol.com

ftsexhsekn@aol.com

hvhuoymdp@aol.com

rxijwkqrpw@aol.com

amy0lr

amy1pv

amy2gr

amy2oj

amy2za

amy3ac

amy3dd

amy3hr

amy3or

amy4bj

amy4hn

hrprgg0c0@aol.com

mwxdkmgfz@aol.com

llafduqq@aol.com

gbxjphkp@aol.com

eagufszmkml@aol.com

h6ln5fnn9wm@aol.com

llcrrhqedt@aol.com

ljoeyanoaw@aol.com

rfdklhqj@aol.com

pwhjknjlxzq@aol.com

nbq4xh4s@aol.com

anna0is

anna5uh

anna5vc

anna5yn

anna6jn

anna6uq

anna9pb


annabf2

annajz6

annamw8

annamy1

# Spam filtering

 找小姐吗

Follow



50  
Boards

0  
Pins

0  
Likes

0  
Followers

0  
Following

## 无锡滨湖华庄找小姐服务I

无锡滨湖华庄找小姐服务I / 无锡滨湖华庄找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

## 无锡滨湖滨湖找小姐服务B

无锡滨湖滨湖找小姐服务B / 无锡滨湖滨湖找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

## 无锡滨湖马山镇找小姐服务S

无锡滨湖马山镇找小姐服务S / 无锡滨湖马山镇找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

## 无锡滨湖河埭找小姐服务e

无锡滨湖河埭找小姐服务e / 无锡滨湖河埭找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

## 无锡滨湖湖塘找小姐服务v

无锡滨湖湖塘找小姐服务v / 无锡滨湖湖塘找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

## 无锡滨湖雪浪找小姐服务P

无锡滨湖雪浪找小姐服务P / 无锡滨湖雪浪找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

## 无锡滨湖江溪找小姐服务p

无锡滨湖江溪找小姐服务p / 无锡滨湖江溪找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

## 无锡滨湖新安找小姐服务W

无锡滨湖新安找小姐服务W / 无锡滨湖新安找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

## 无锡滨湖梅村找小姐服务p

无锡滨湖梅村找小姐服务p / 无锡滨湖梅村找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

## 无锡新区旺庄找小姐服务P

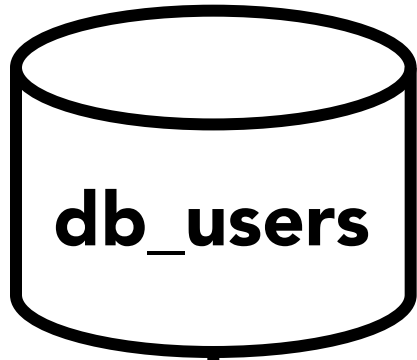
无锡新区旺庄找小姐服务P / 无锡新区旺庄找小姐上门服务185-10205881哪里有找 电话 185-10205881(内附相册) 娜娜-小姐服务态度好, 兼职学生妹 妓女 小姐 美女 小妹 少妇 援交 鸡婆 包夜联系电话 185-10205881 娜娜 哪里有外国小姐服务-兼职学生妹服务=南北佳丽荟萃, 让你感受帝王般的尊崇!品质服务, 精彩无限!



Follow

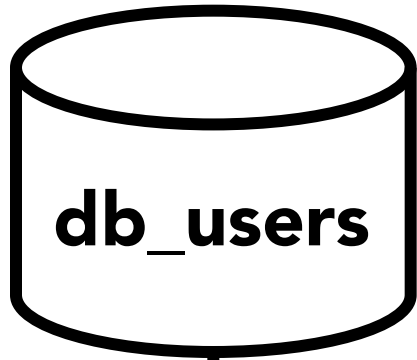
@arburbank

# Spam filtering



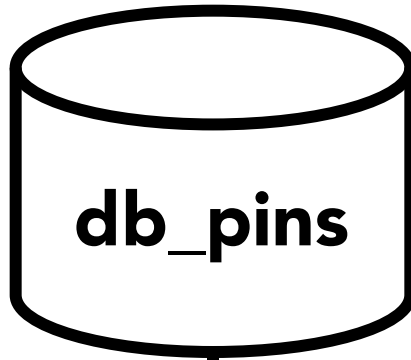
same "about me"  
email regexes  
same name  
**rules**

# Spam filtering



same "about me"  
email regexes  
same name

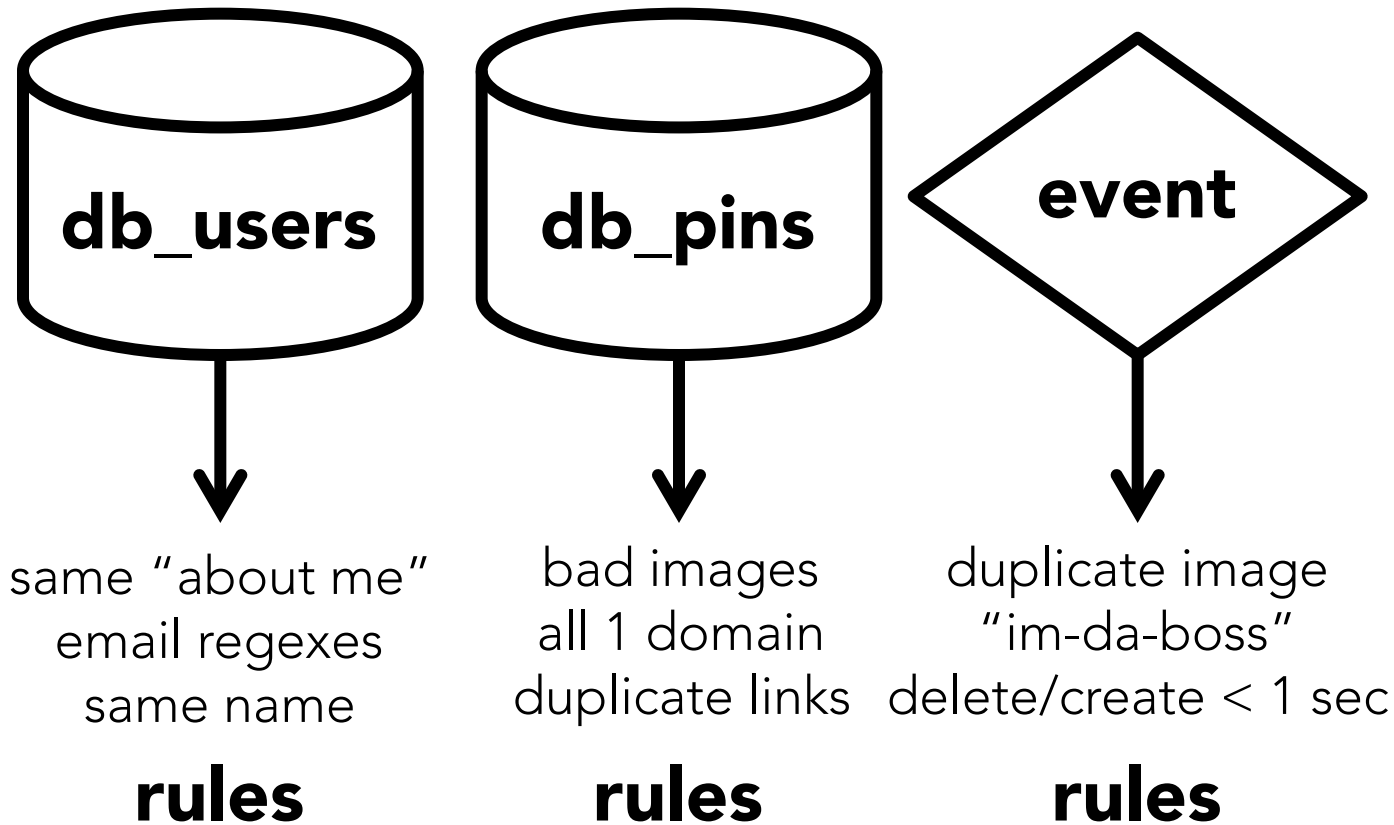
**rules**



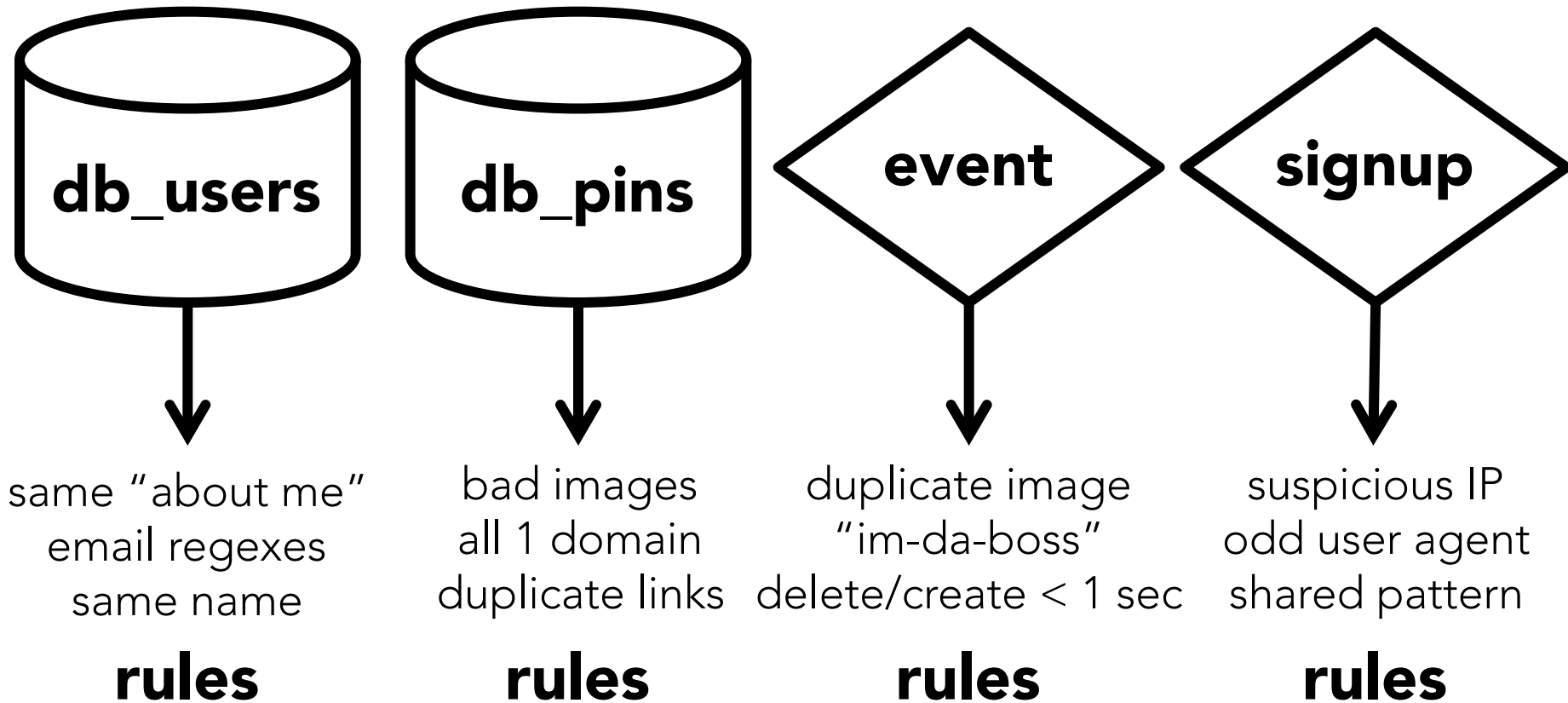
bad images  
all 1 domain  
duplicate links

**rules**

# Spam filtering



# Spam filtering





# Spam filtering

File Edit Options Buffers Tools Python Help

```
class SignupGjtfhtSpammersJob(PossibleSpammerHiveJob):
    """There are many accounts with the name gjtfht fthtr."""
    _SIGNAL = 'signup_gjtfht_fthtr'
    _MULTIDATE_EXECUTE = True
    _QUERY_TEMPLATE = """
SET mapred.reduce.tasks=1;

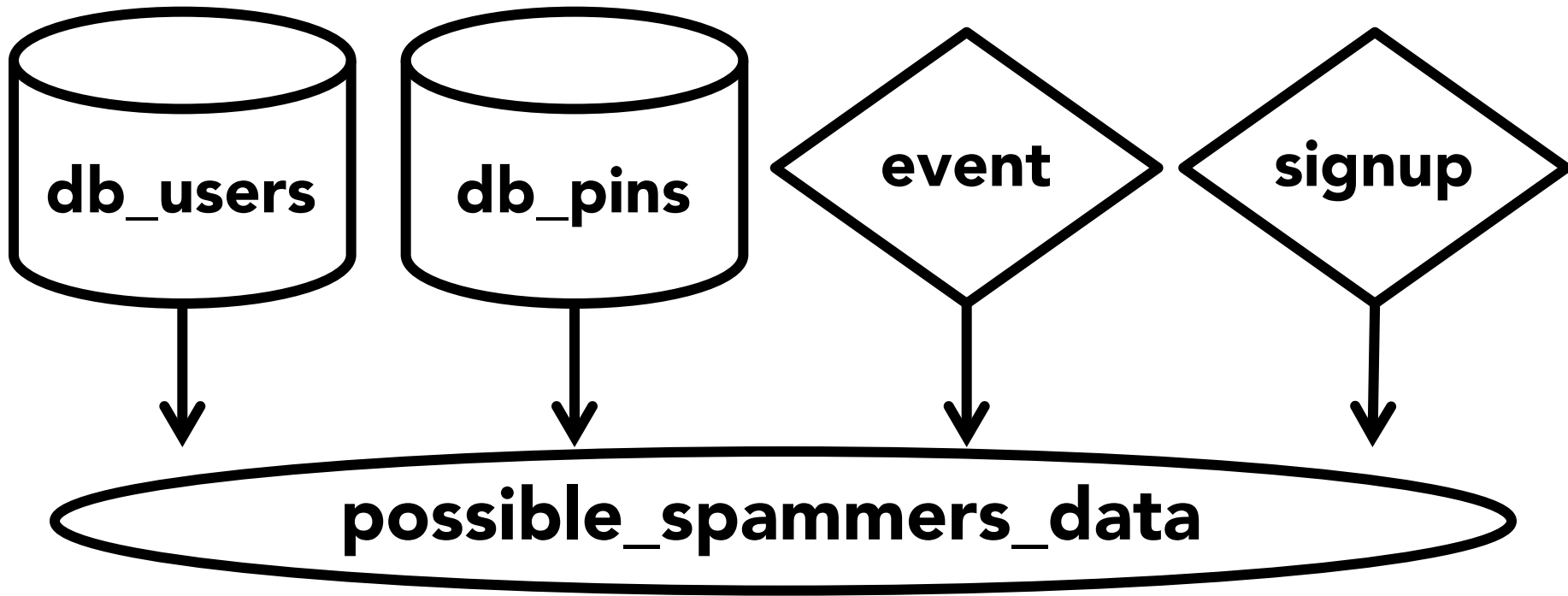
INSERT OVERWRITE TABLE pinalytics.possible_spammers_data PARTITION(signal='signup_gjtfht_fthtr', dt)

SELECT id as userid, get_json(json, 'email') as info, to_date(created_at) as dt
FROM
db_users
WHERE get_json(json, 'first_name') = 'gjtfht'
AND get_json(json, 'last_name') = 'fthtr'
AND to_date(created_at) >= '%(start_date)s'
AND to_date(created_at) <= '%(end_date)s'
;
"""

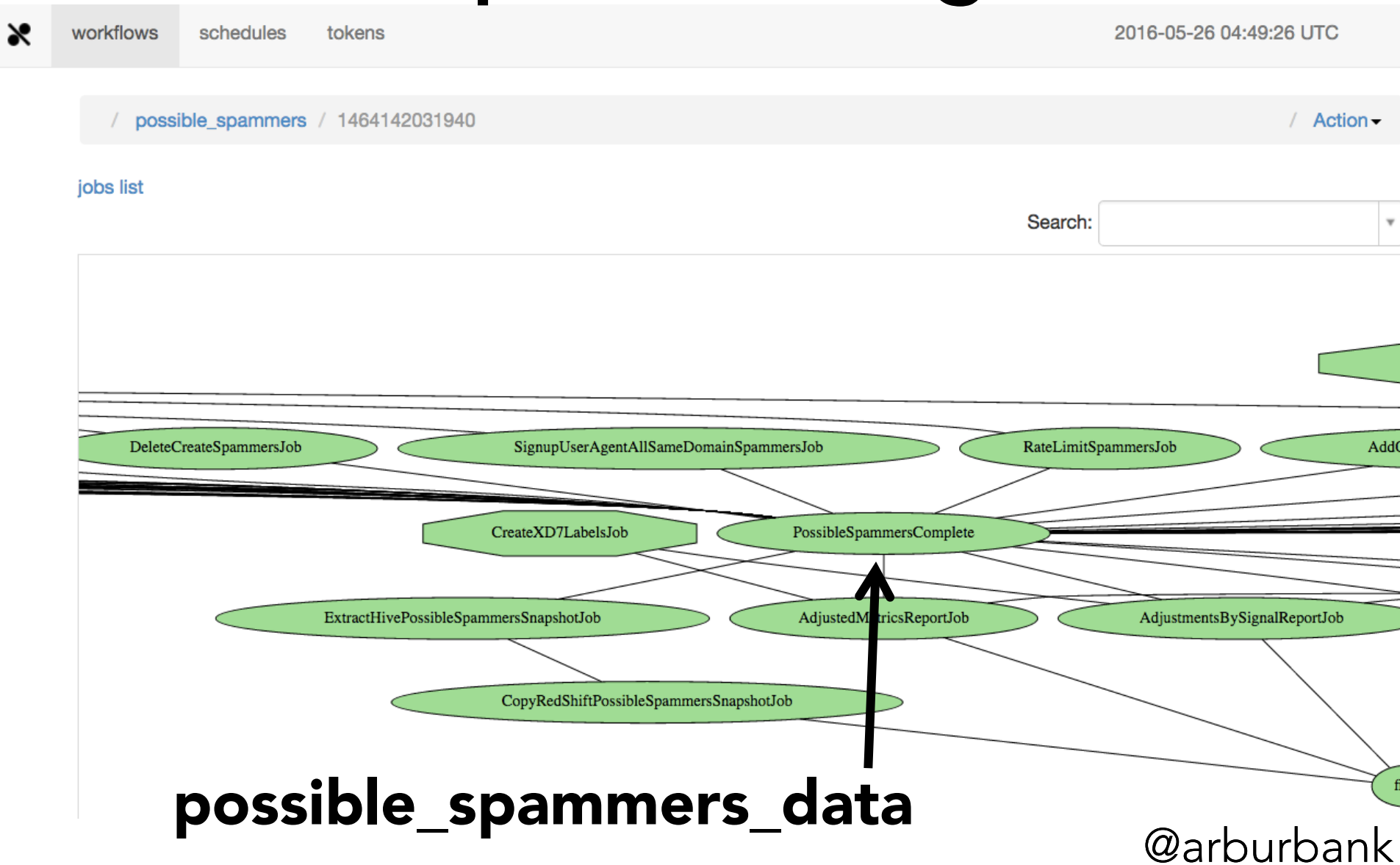
class JerseyBagSignupSpammersJob(PossibleSpammerHiveJob):
    """In October 2014, thousands of accounts for fake jerseys and bags."""
    _SIGNAL = "signup_jersey_bag"
    _MULTIDATE_EXECUTE = True
    _QUERY_TEMPLATE = """
SET mapred.reduce.tasks=1;

-UU-:----F1 possible_spammers_data.py 19% L398 Git-master (Python)-----
```

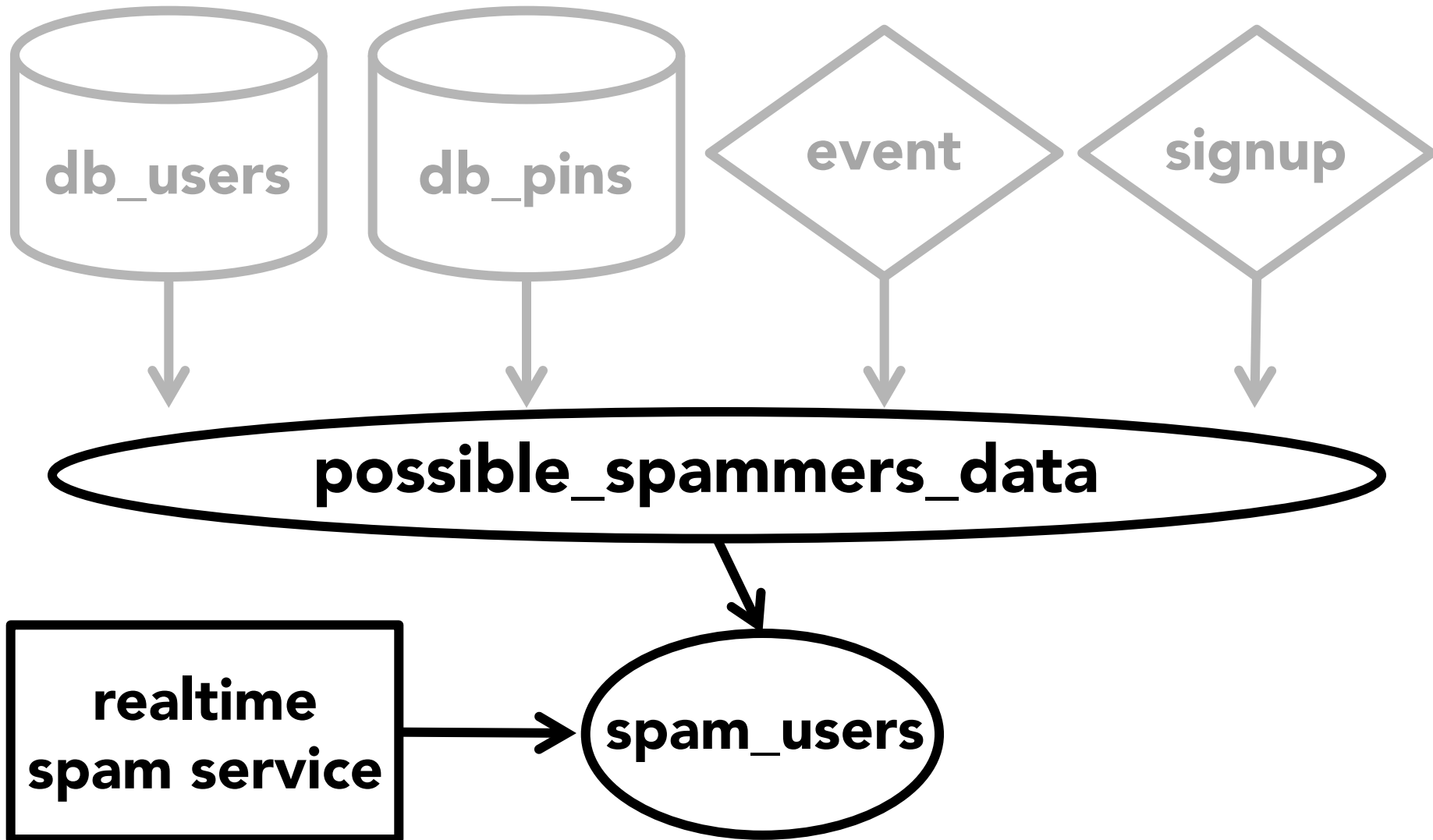
# Spam filtering



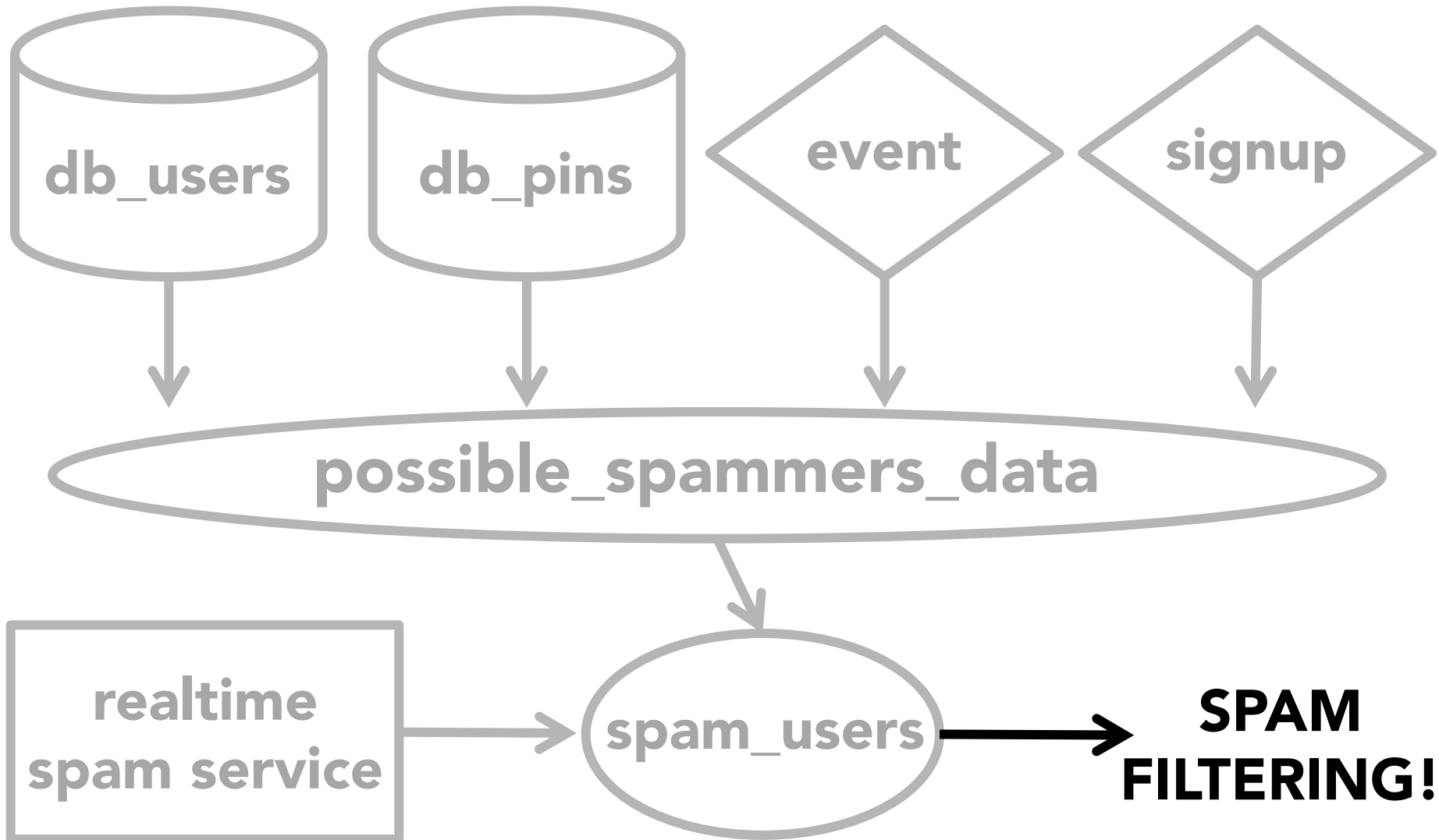
# Spam filtering



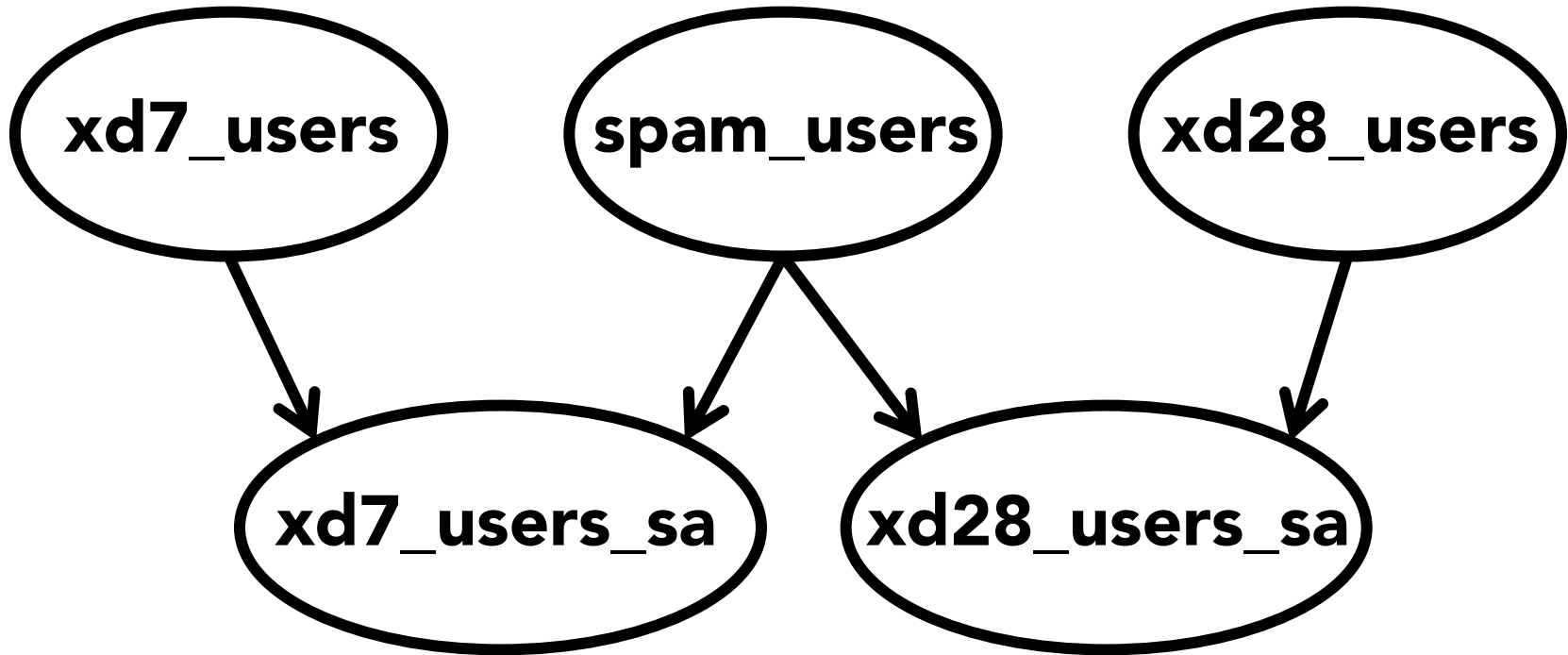
# Spam filtering



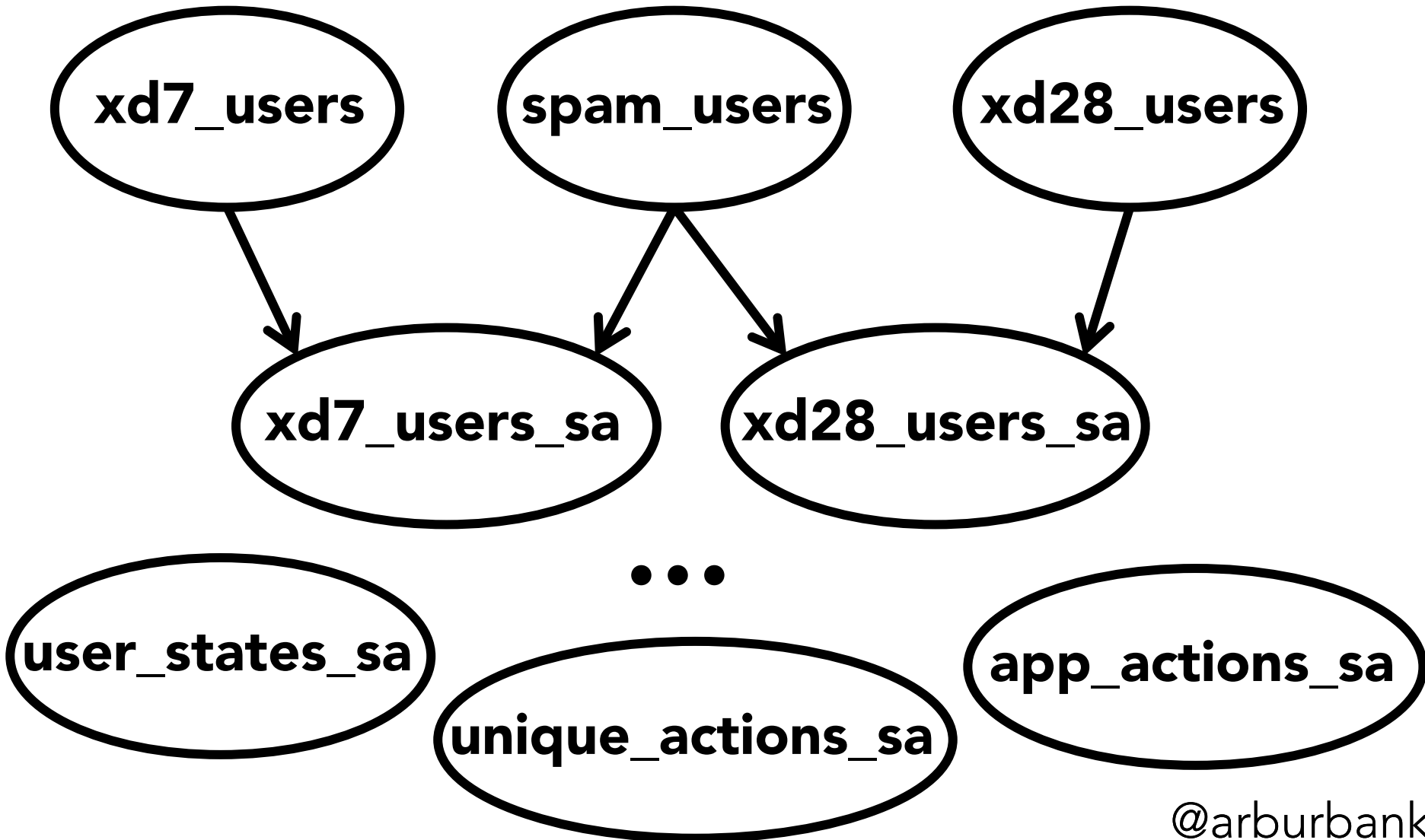
# Spam filtering



# Spam filtering



# Spam filtering



# Spam filtering

workflows schedules tokens

2016-05-26 06:26:55 UTC

/ spam\_adjusted / 1464145629038

/ Action ▾

jobs list

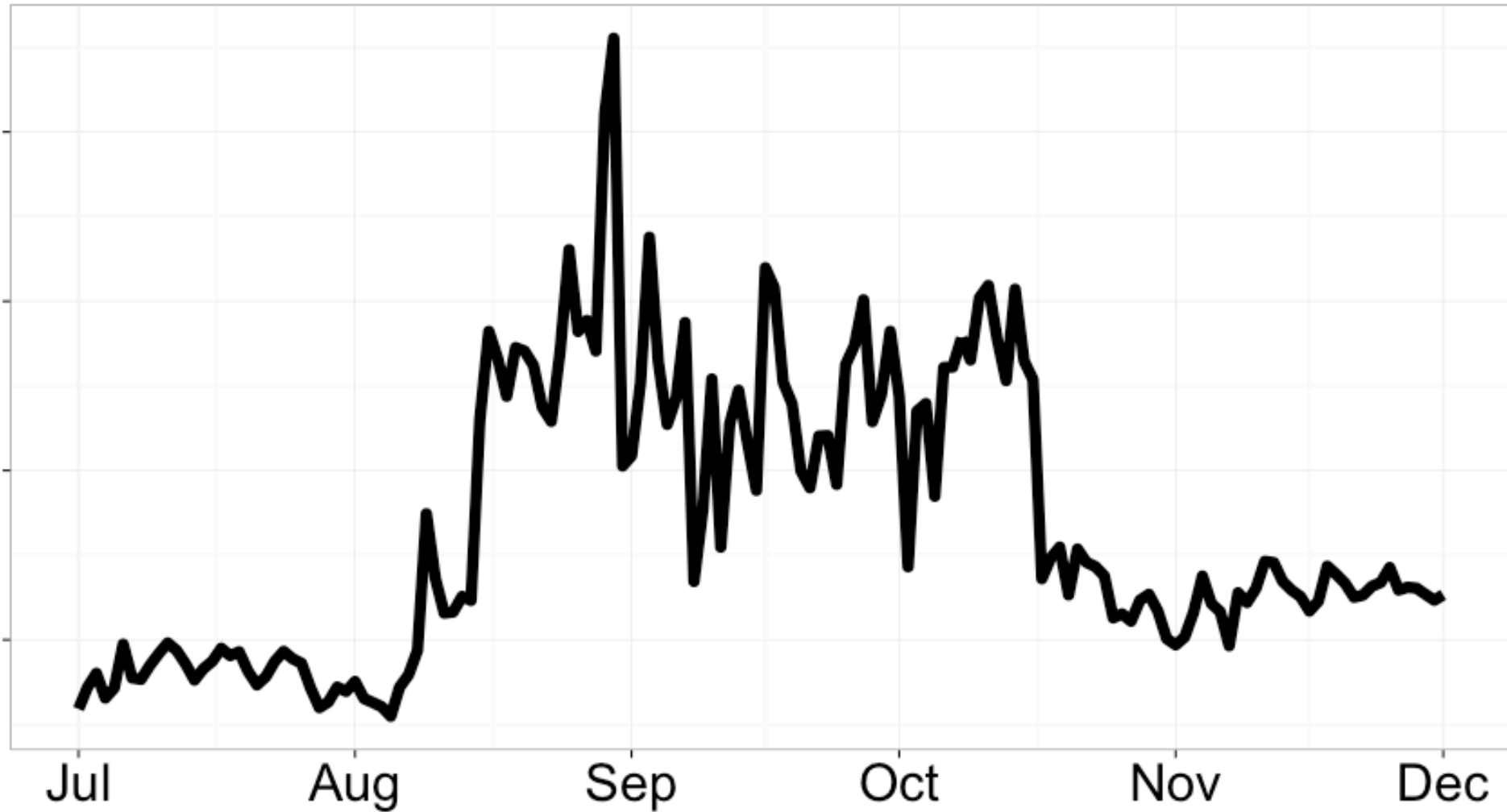
Search:



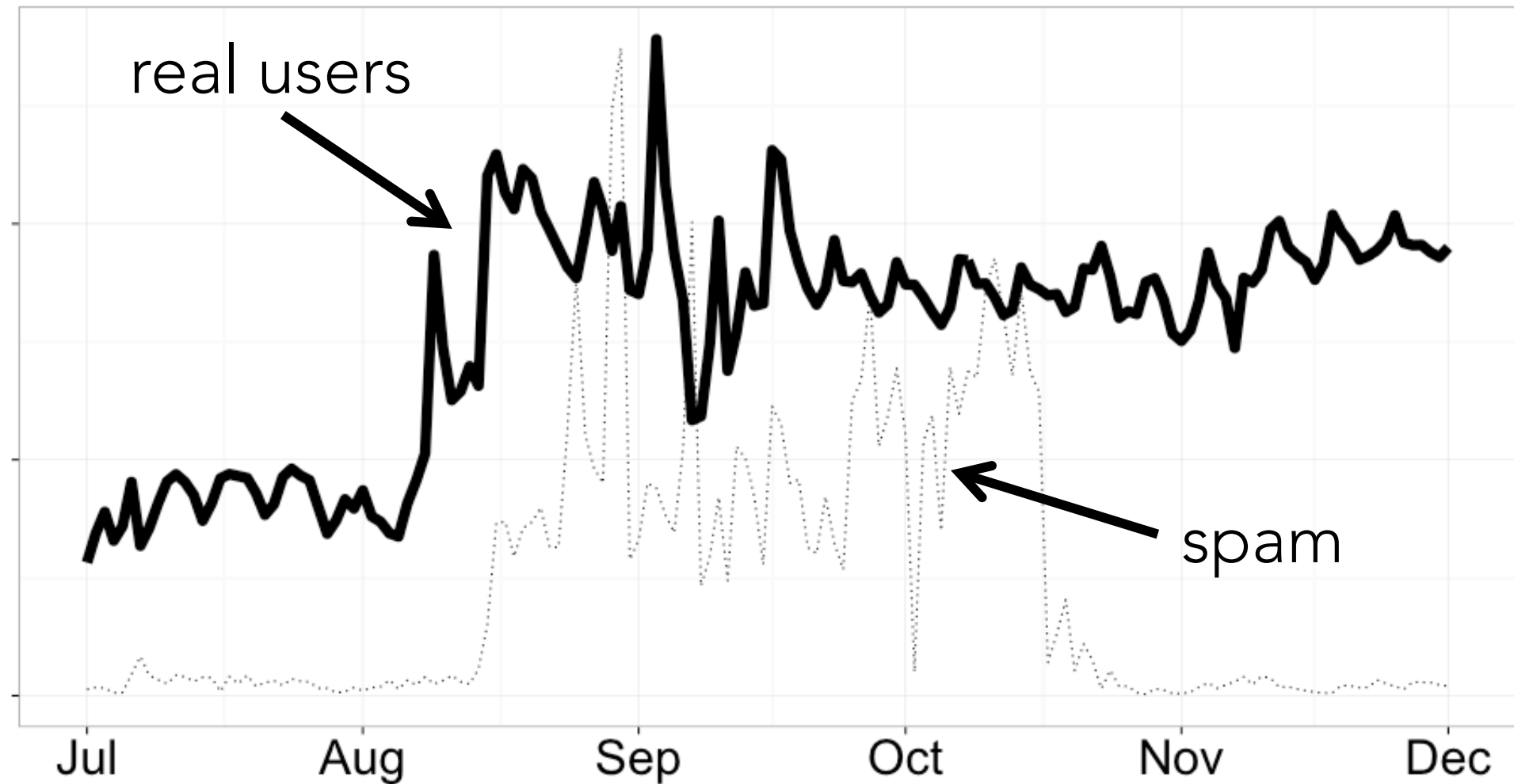
@arburbank



# Signups, fall 2012



# Signups, fall 2012



AMSTERDAM

INTERNATIONAL  
SOFTWARE DEVELOPMENT  
CONFERENCE 2016

goto;  
conference

follow us on @GOTOamst

Workshop: June 13 / Conference: June 14-15

# data as software

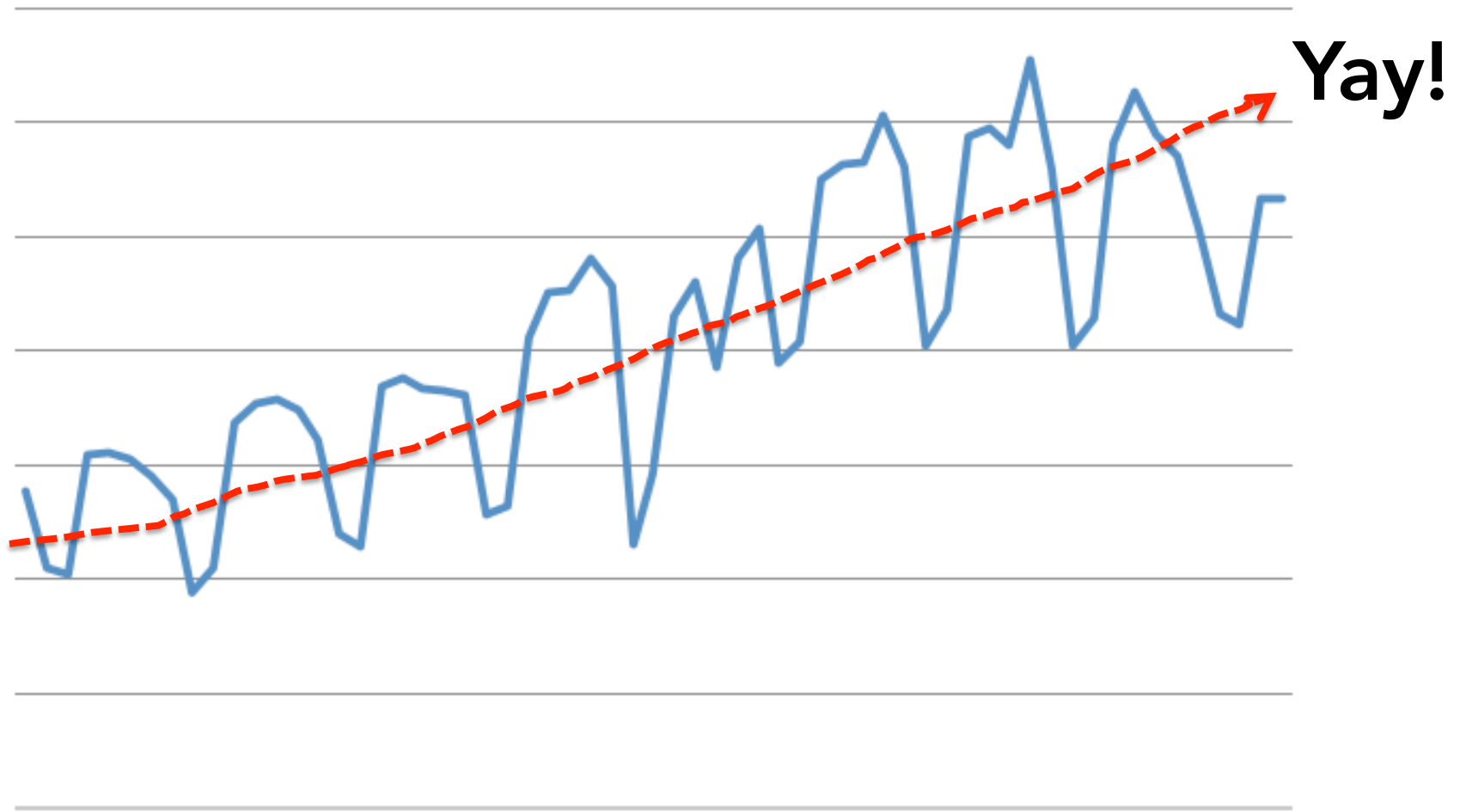
@arburbank

# data aggregation

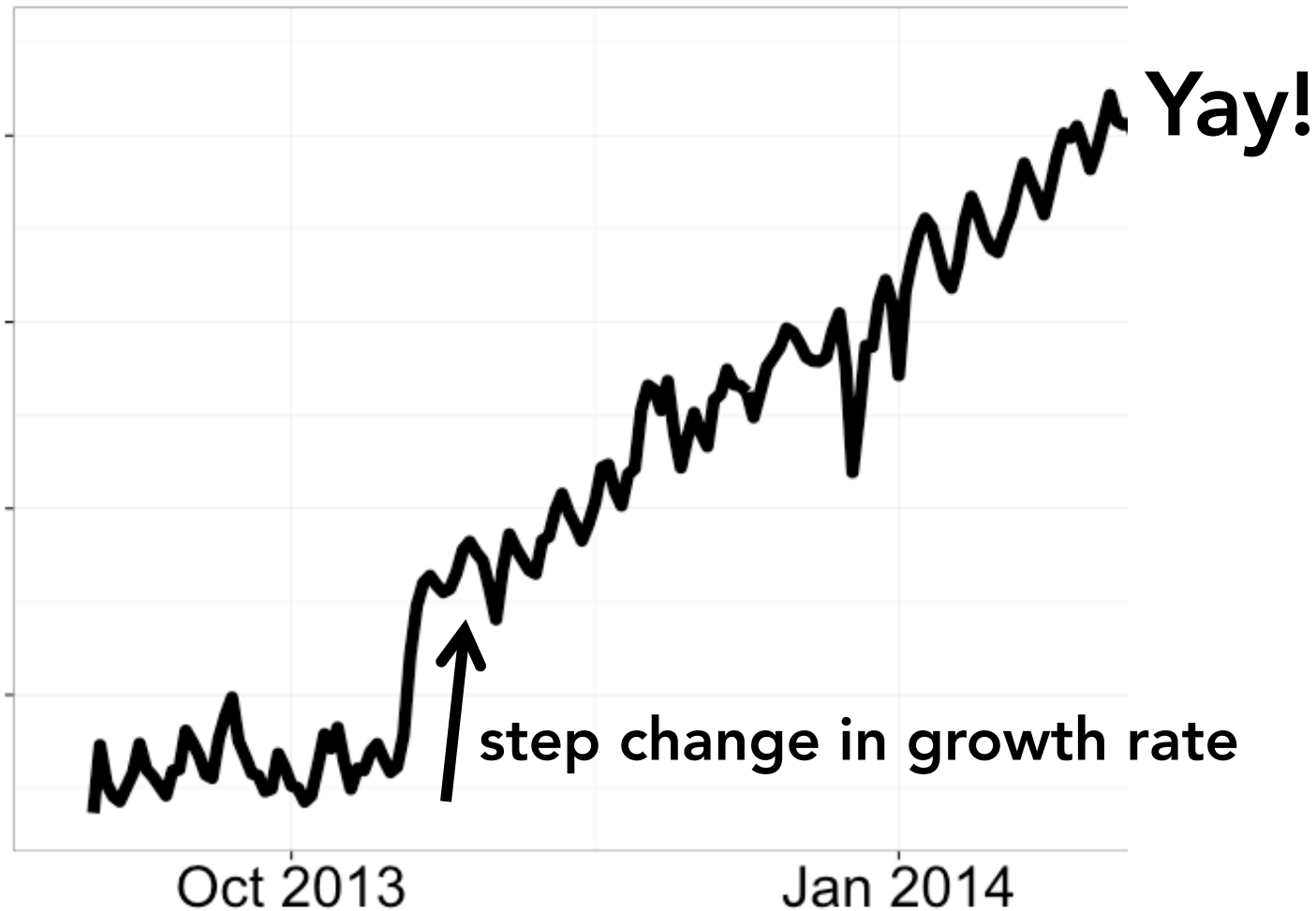
# **workflows**

# **software engineering**

# We can count again!

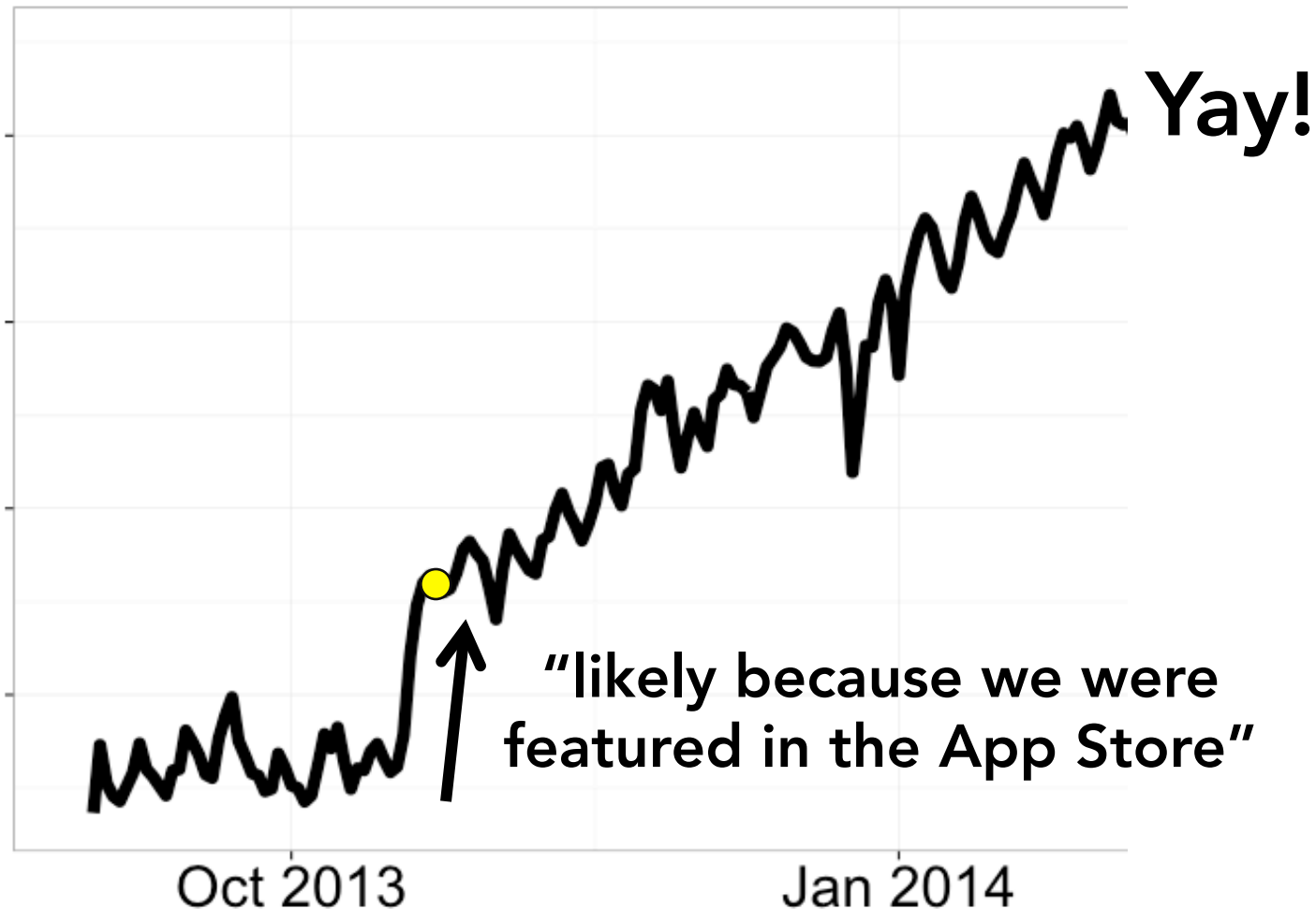


# Daily Active iPhone Users

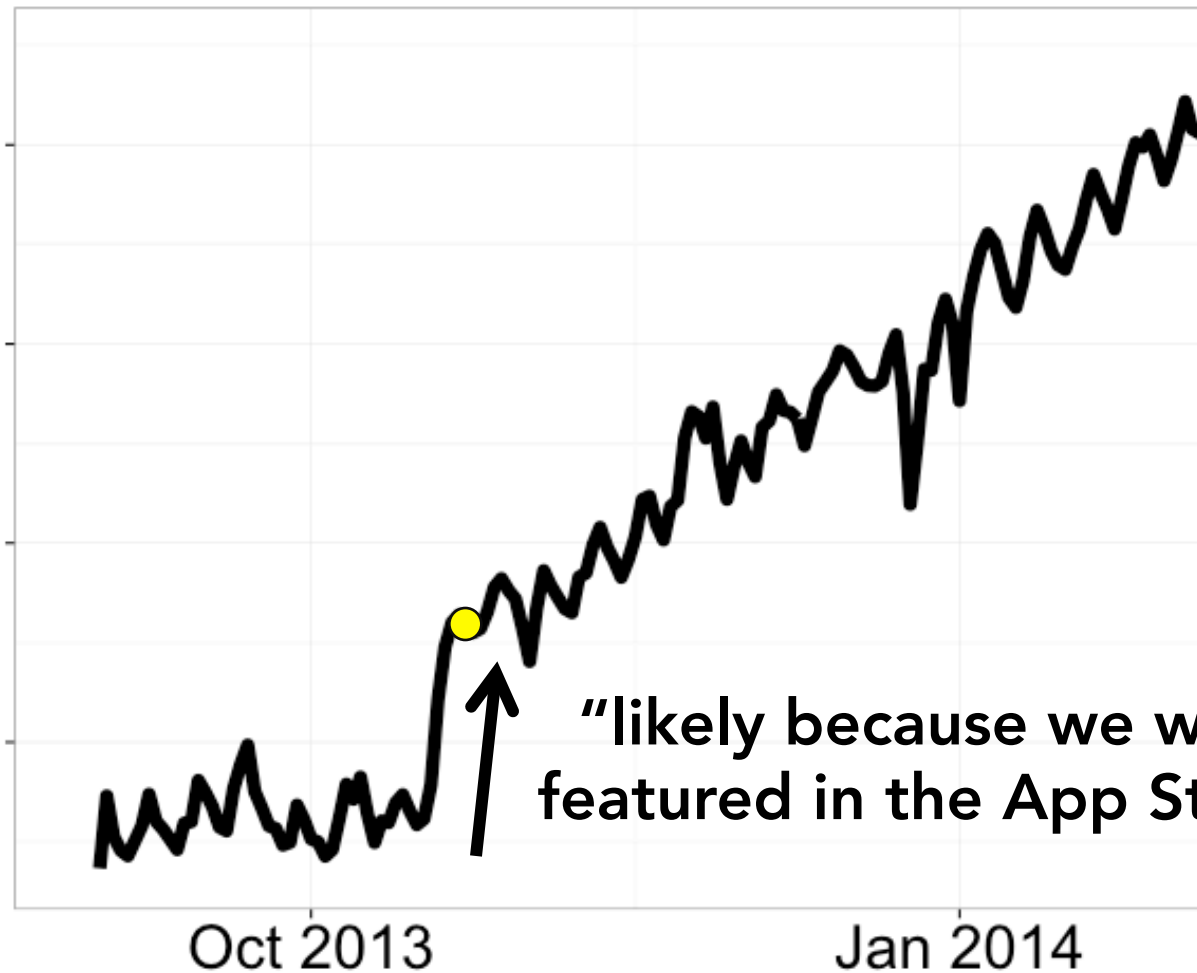




# Daily Active iPhone Users

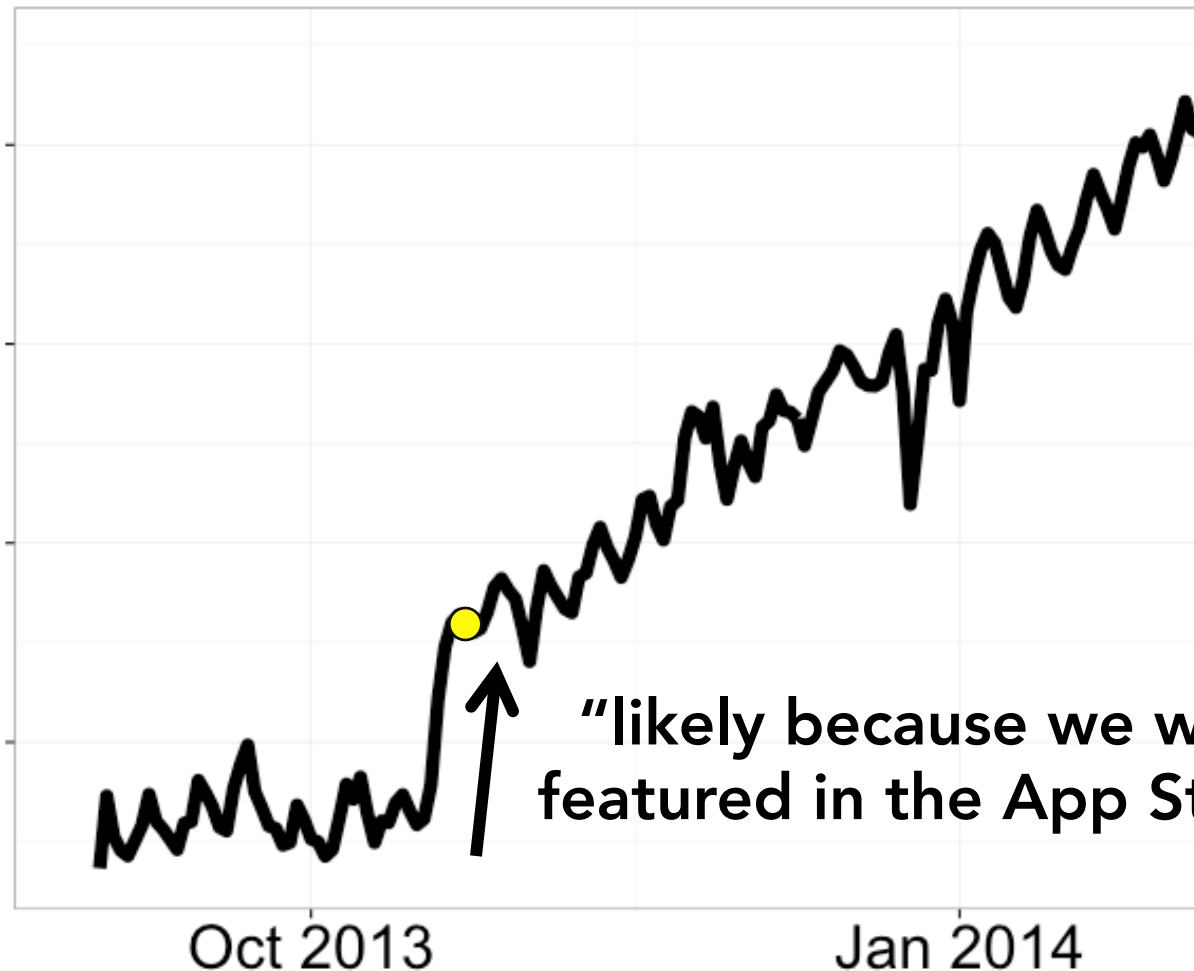


# Daily Active iPhone Users



Really?

# Daily Active iPhone Users



**What else happened?**

**"likely because we were featured in the App Store"**

# Daily Active iPhone Users

**From: mobile PM**

**To: team**

**Subject: Have an iOS device? Check out “Featured” in the App Store ...**

We launched v3.0 of the iOS Pinterest App today, which represents a *huge* amount of work from the mobile team. ... an inspired combination of iOS7's and Pinterest's visual language, the interaction model has been carefully designed ... Transitions, gestures, and information architecture were painstakingly thought through.

ios



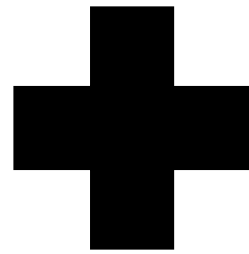
# Daily Active iPhone Users

**MacWorld: "What changed in iOS 7?"**

Not that long ago, iOS apps running in the background on your iPhone or iPad were essentially stuck in suspended animation, unable to do anything (besides trigger alerts via Apple's "push notification" system) until you re-launched them and put them back on your display.

All of that changed [in October 2013] thanks to "Background App Refresh," a feature introduced with iOS 7.

@arburbank



ios



@arburbank

# Daily Active iPhone Users

We used iOS 7's background fetch feature to get users' homefeeds ready for them before they launched the app!

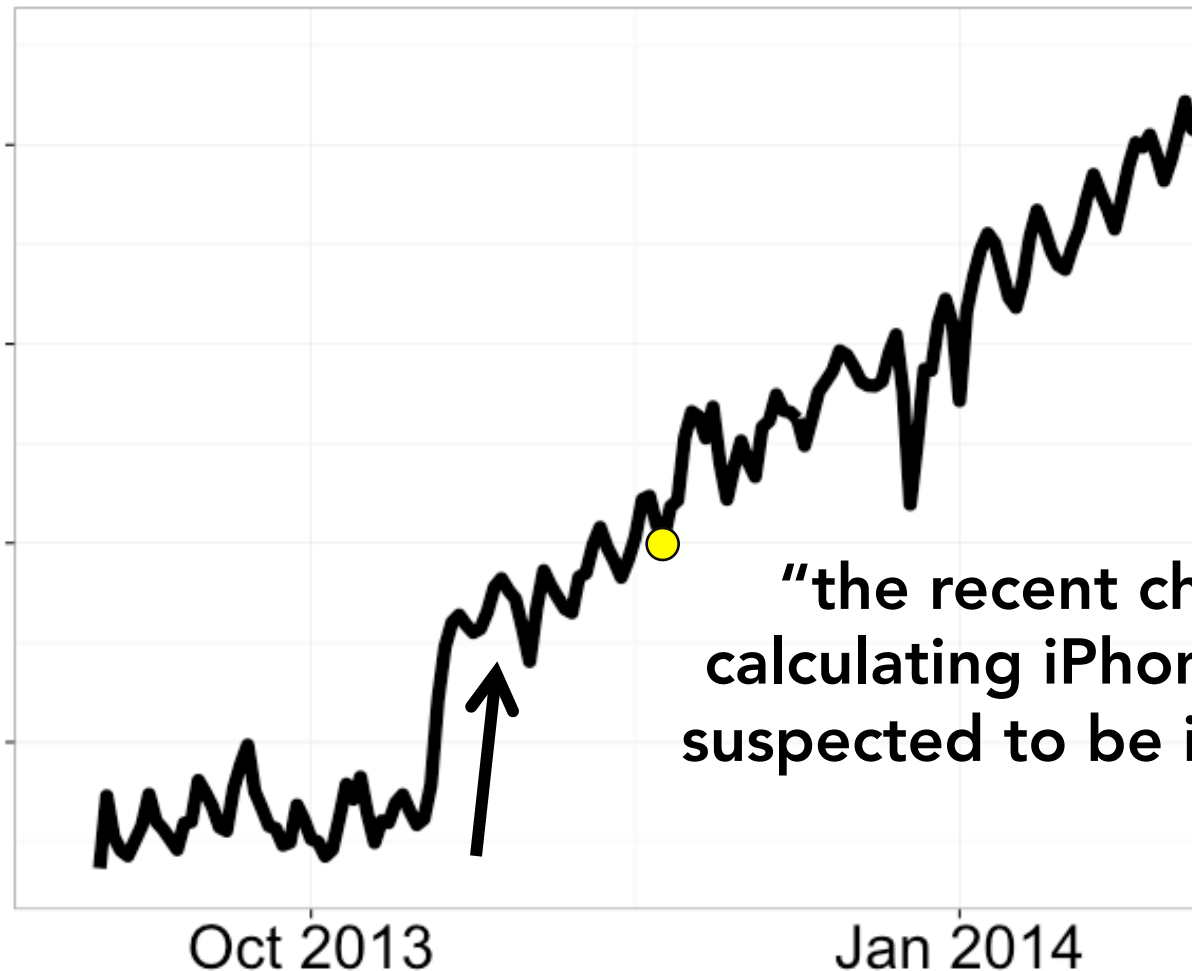


# Daily Active iPhone Users



@arburbank

# Daily Active iPhone Users



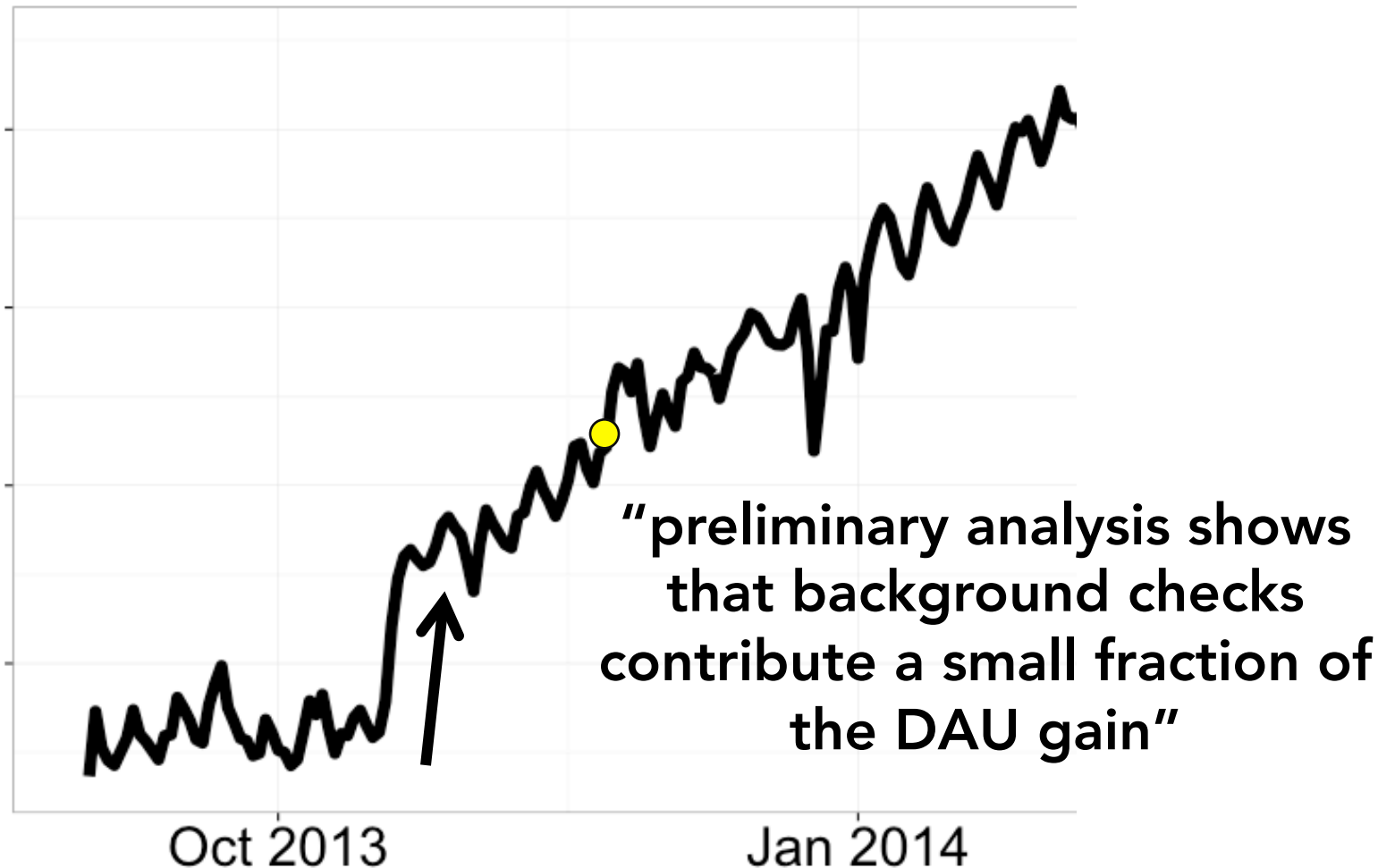
**But how  
much?**

# Daily Active iPhone Users

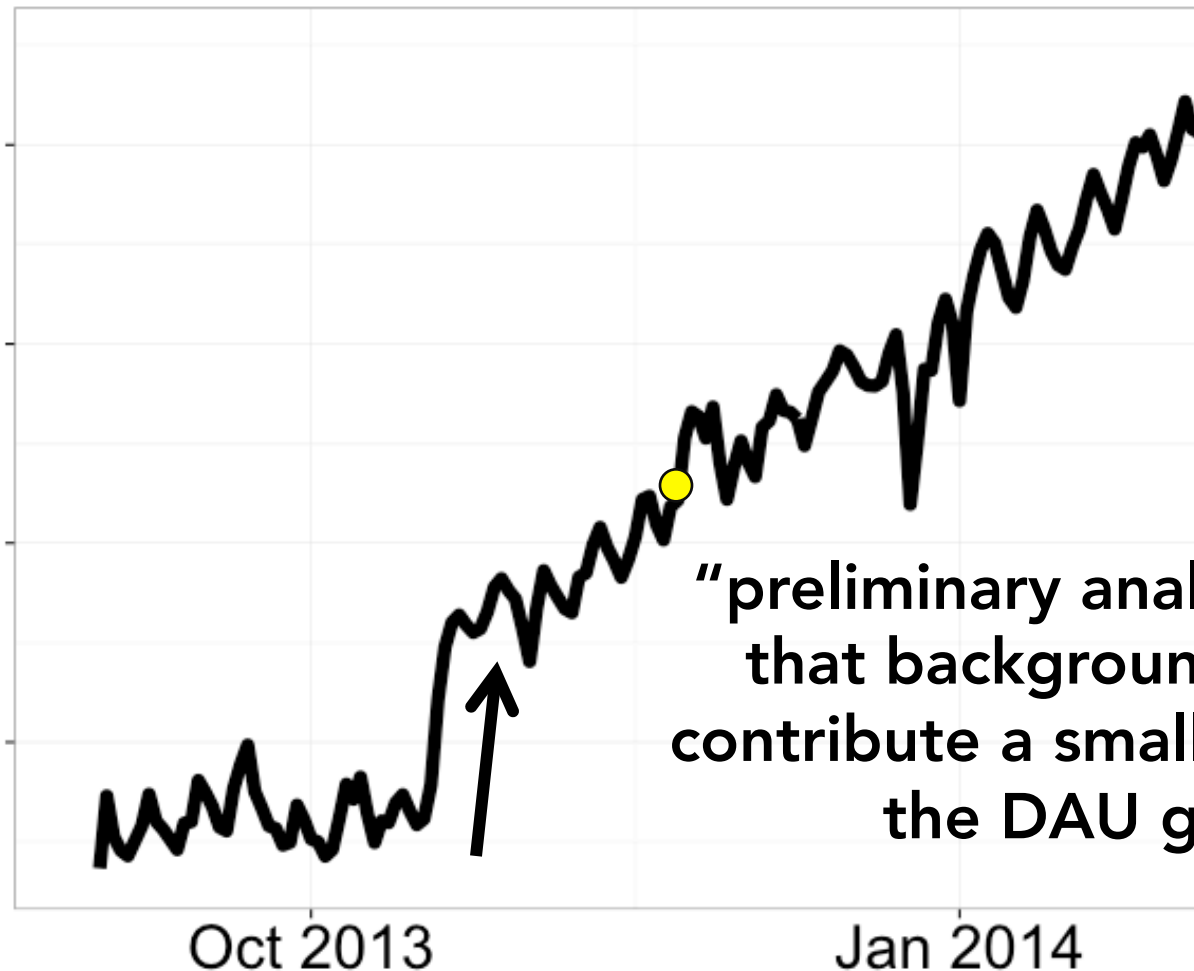
“Only the /v3/feeds/home/ handler is being called in the background.”

– iOS developer

# Daily Active iPhone Users



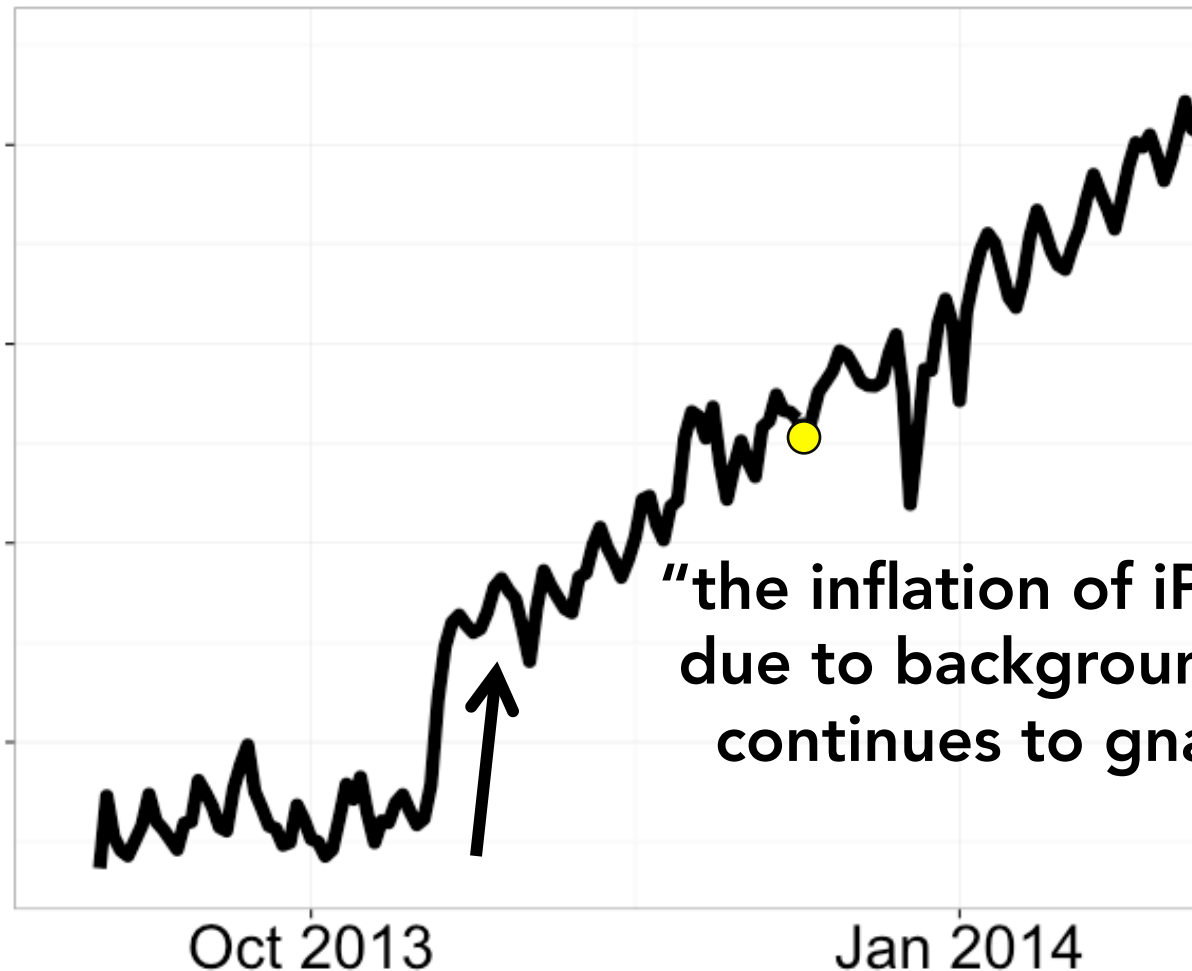
# Daily Active iPhone Users



**Really?**

**"preliminary analysis shows  
that background checks  
contribute a small fraction of  
the DAU gain"**

# Daily Active iPhone Users

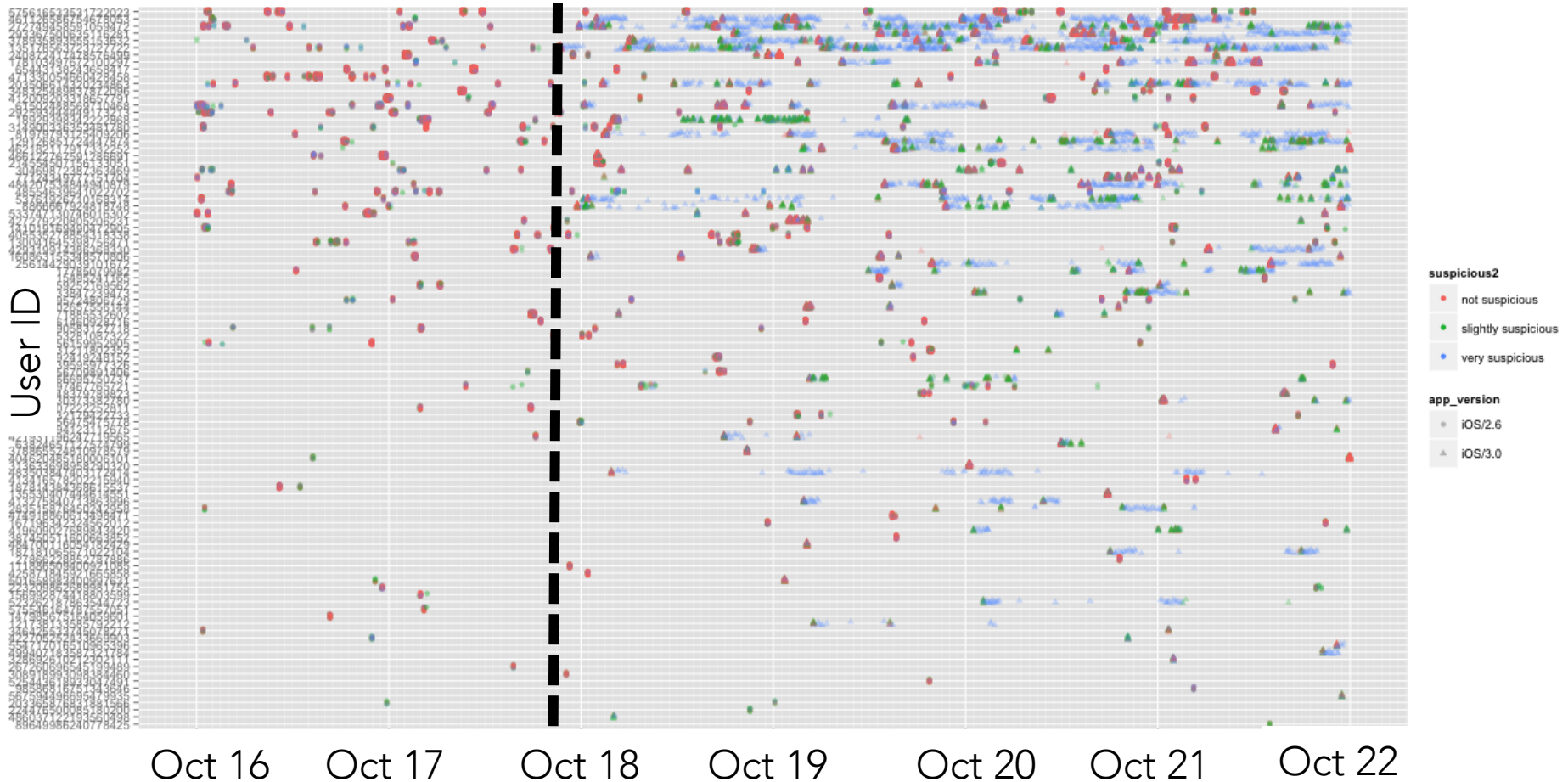


# Daily Active iPhone Users

userid	ts	path	status
2975021	08:36:49.32	/notifications/	200
2975021	08:37:03.41	/pin/123/	200
2975021	08:37:07.22	/pin/repin/	200
2975021	11:32:10.39	/v3/home/	200
2975021	11:32:11.97	/v3/callback/	200
2975021	11:32:13.23	/v3/user/	200
2975021	14:19:03.41	/pin/347/	200
2975021	14:19:07.22	/pin/repin/	200

# Daily Active iPhone Users

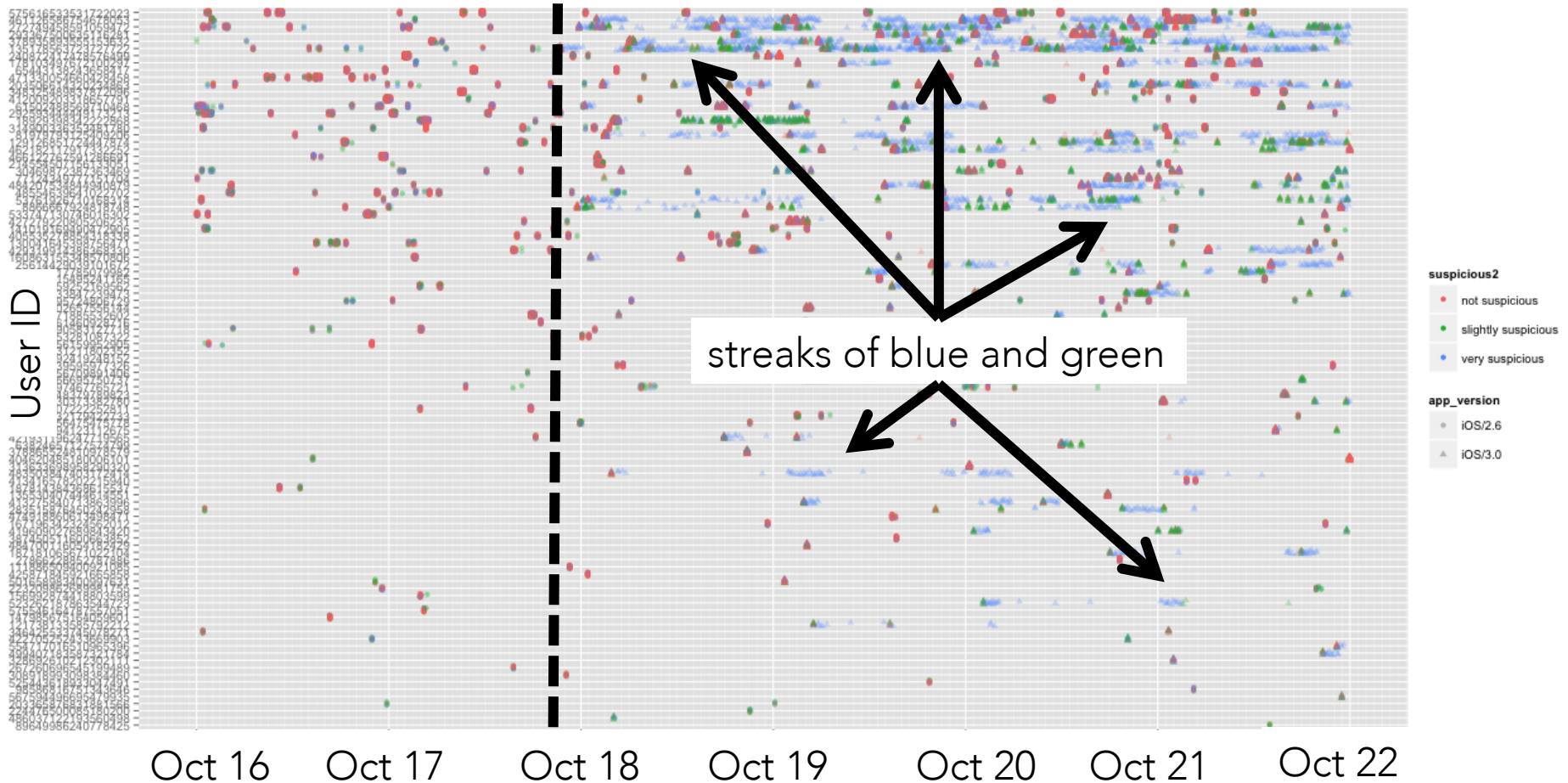
not suspicious      slightly suspicious      very suspicious





# Daily Active iPhone Users

not suspicious      slightly suspicious      very suspicious



“You can easily see long smears of blue, which are background hits to (mostly) v3\_home\_feed.

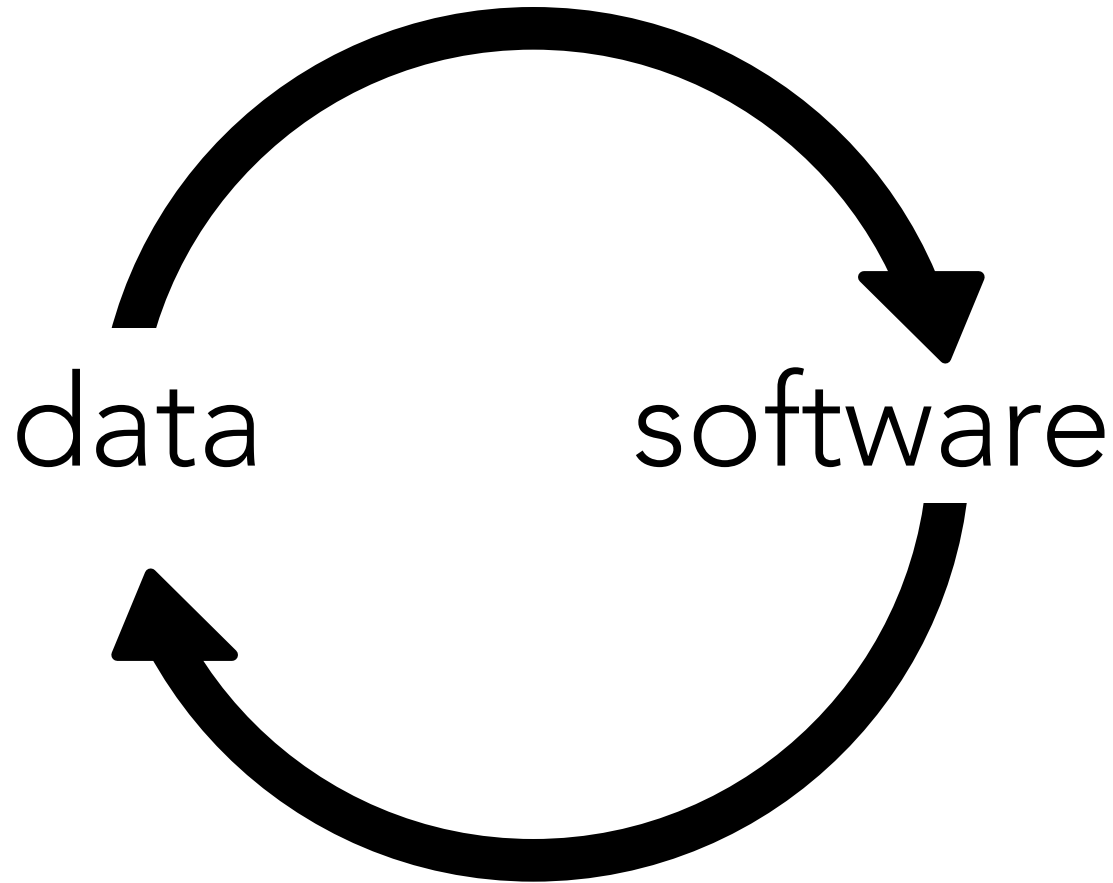
“You can easily see long smears of blue, which are background hits to (mostly) v3\_home\_feed. These we knew about already.

“You can easily see long smears of blue, which are background hits to (mostly) v3\_home\_feed. These we knew about already. But after the 3.0 app is launched, you can also see long smears of green triangles ...

“You can easily see long smears of blue, which are background hits to (mostly) v3\_home\_feed. These we knew about already. But after the 3.0 app is launched, you can also see long smears of green triangles ... and there are no corresponding smears of green circles before October 18.

“You can easily see long smears of blue, which are background hits to (mostly) v3\_home\_feed. These we knew about already. But after the 3.0 app is launched, you can also see long smears of green triangles ... and there are no corresponding smears of green circles before October 18. This makes me believe that when in the background, the app can do more than just fetch v3\_home\_feed.

# Daily Active iPhone Users



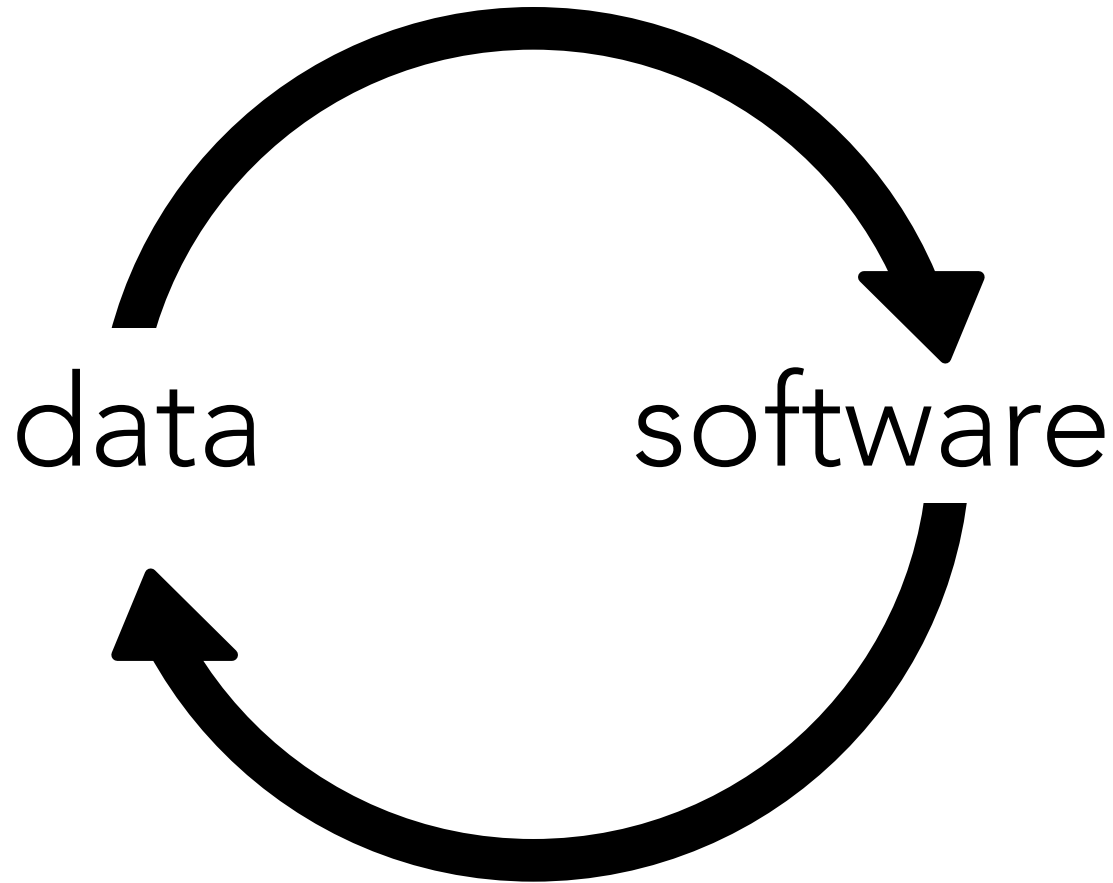
# Daily Active iPhone Users

As per our discussions yesterday, I actually confirmed iOS applications **can be launched due to a background fetch even when they are not visibly running** (not appearing in multitask tray).

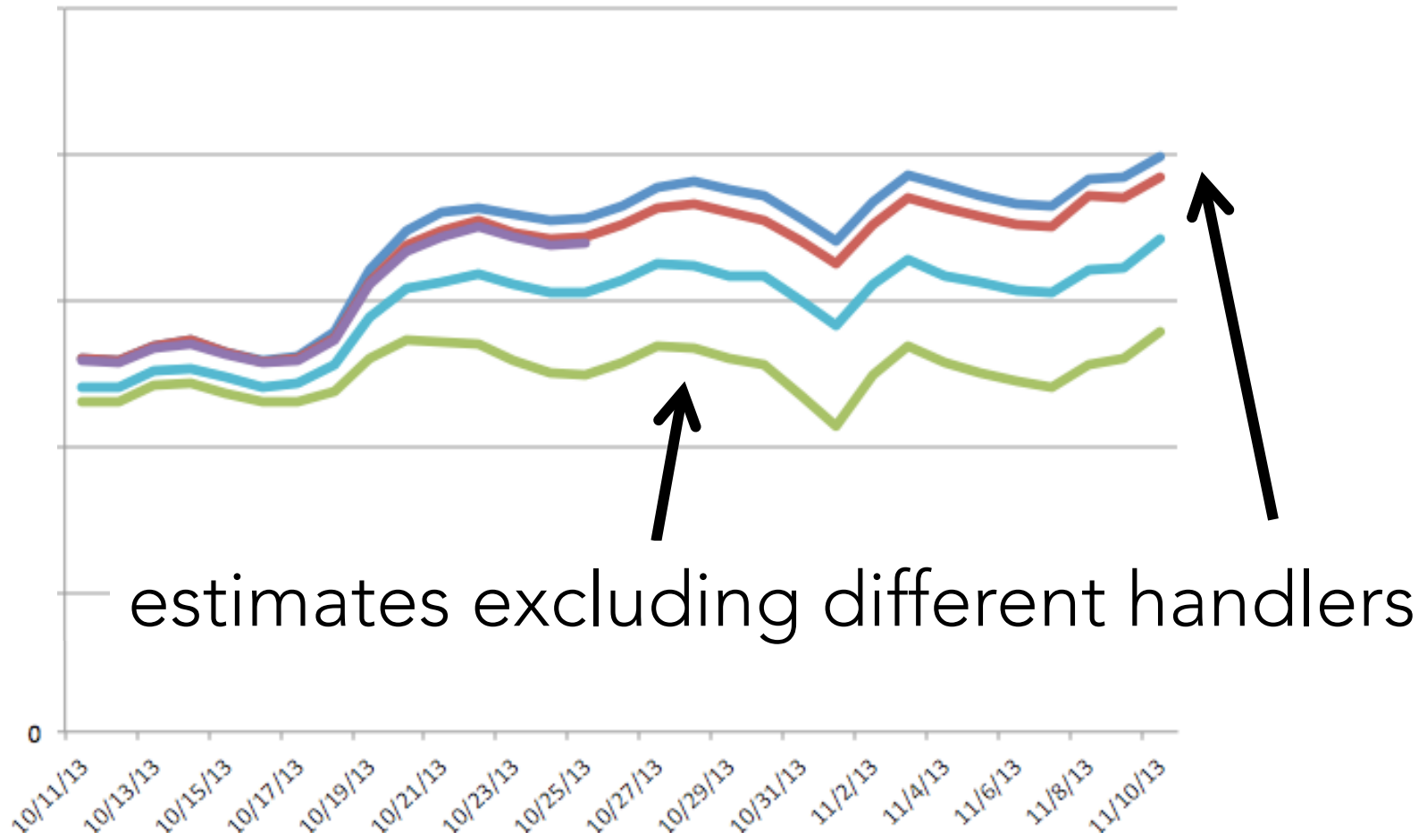
... **(this could explain the green streaks** we were seeing).



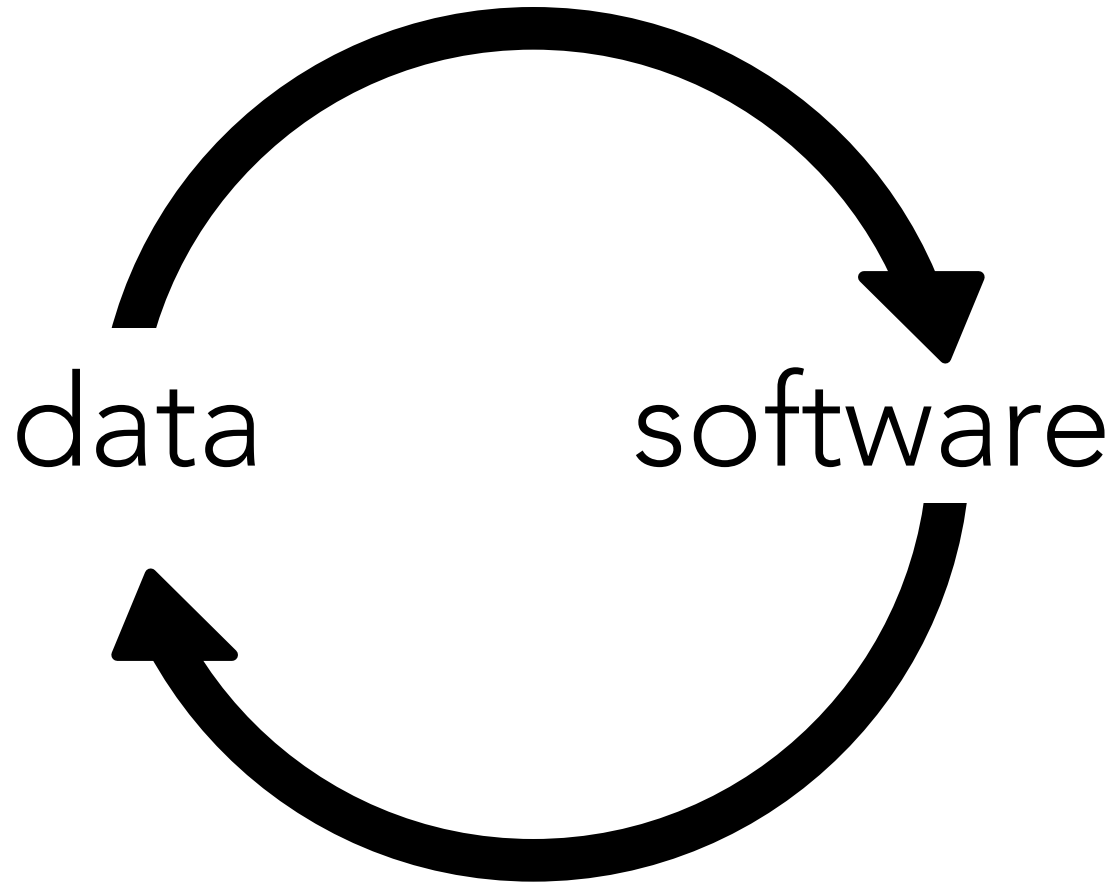
# Daily Active iPhone Users



# Daily Active iPhone Users



# Daily Active iPhone Users



# Daily Active iPhone Users

userid	ts	path	status
2975021	08:36:49.32	/notifications/	200
3789823	08:37:03.41	/pin/123/	200
3789823	08:37:07.22	/pin/repin/	200
1724468	08:37:10.39	/v3/home/	200
8779233	08:37:11.97	/category/art/	200

# Daily Active iPhone Users

userid	ts	path	status	app_state
2975021	08:36:49.32	/notifications/	200	background
3789823	08:37:03.41	/pin/123/	200	active
3789823	08:37:07.22	/pin/repin/	200	active
1724468	08:37:10.39	/v3/home/	200	background
8779233	08:37:11.97	/category/art/	200	active

# Daily Active iPhone Users

**DAU**

userid	ts	path	status	app_state
2975021	08:36:49.32	/notifications/	200	background
3789823	08:37:03.41	/pin/123/	200	<b>active</b>
3789823	08:37:07.22	/pin/repin/	200	<b>active</b>
1724468	08:37:10.39	/v3/home/	200	background
8779233	08:37:11.97	/category/art/	200	<b>active</b>

# Daily Active iPhone Users

**not DAU**

userid	ts	path	status	app_state
2975021	08:36:49.32	/notifications/	200	<b>background</b>
3789823	08:37:03.41	/pin/123/	200	active
3789823	08:37:07.22	/pin/repin/	200	active
1724468	08:37:10.39	/v3/home/	200	<b>background</b>
8779233	08:37:11.97	/category/art/	200	active

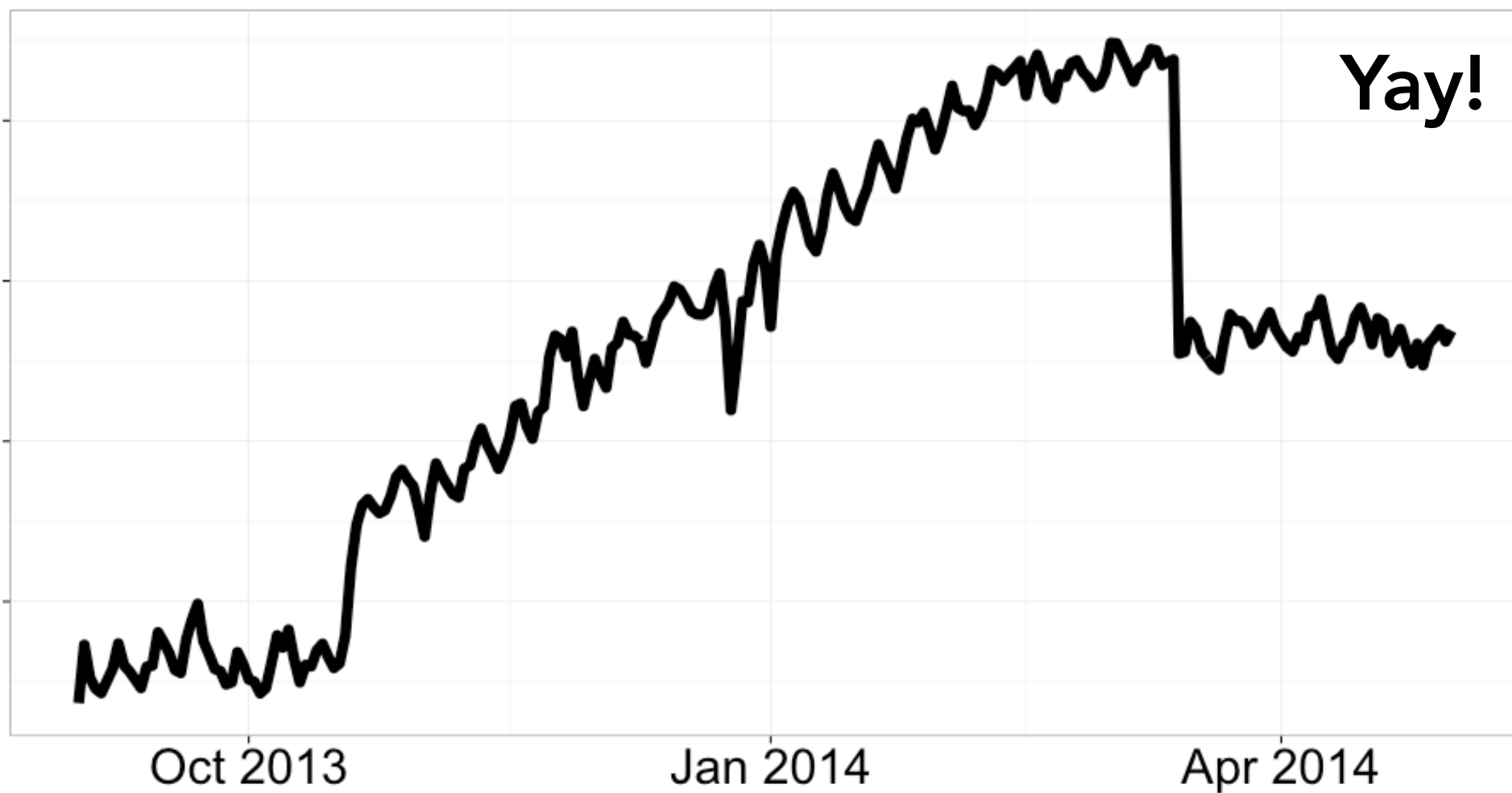
# Daily Active iPhone Users

```
class CreateUniqueAPIRequestsJob(data_job.HiveJob):  
    _QUERY_TEMPLATE = """
```

```
INSERT OVERWRITE TABLE unique_actions  
PARTITION(dt="% (end_date)s", action_type=100)  
    SELECT userid,  
           count(*)  
    FROM auth_compact_api  
    WHERE dt="% (end_date)s"  
    -- exclude background requests on iOS  
    AND (app_state not in ('background','inactive'))  
    OR app_state is NULL)  
GROUP BY userid, dt;  
    """
```



# Daily Active Users



AMSTERDAM

INTERNATIONAL  
SOFTWARE DEVELOPMENT  
CONFERENCE 2016

goto;  
conference

follow us on @GOTOamst

Workshop: June 13 / Conference: June 14-15

# data as software

@arburbank

# app behavior

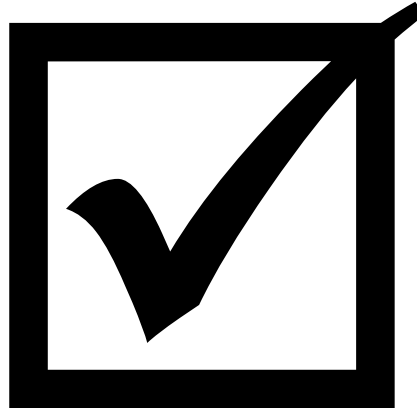
**changes in activity**

# explicit logging



# **Part II: how are we doing?**

*counting is even harder than we thought*

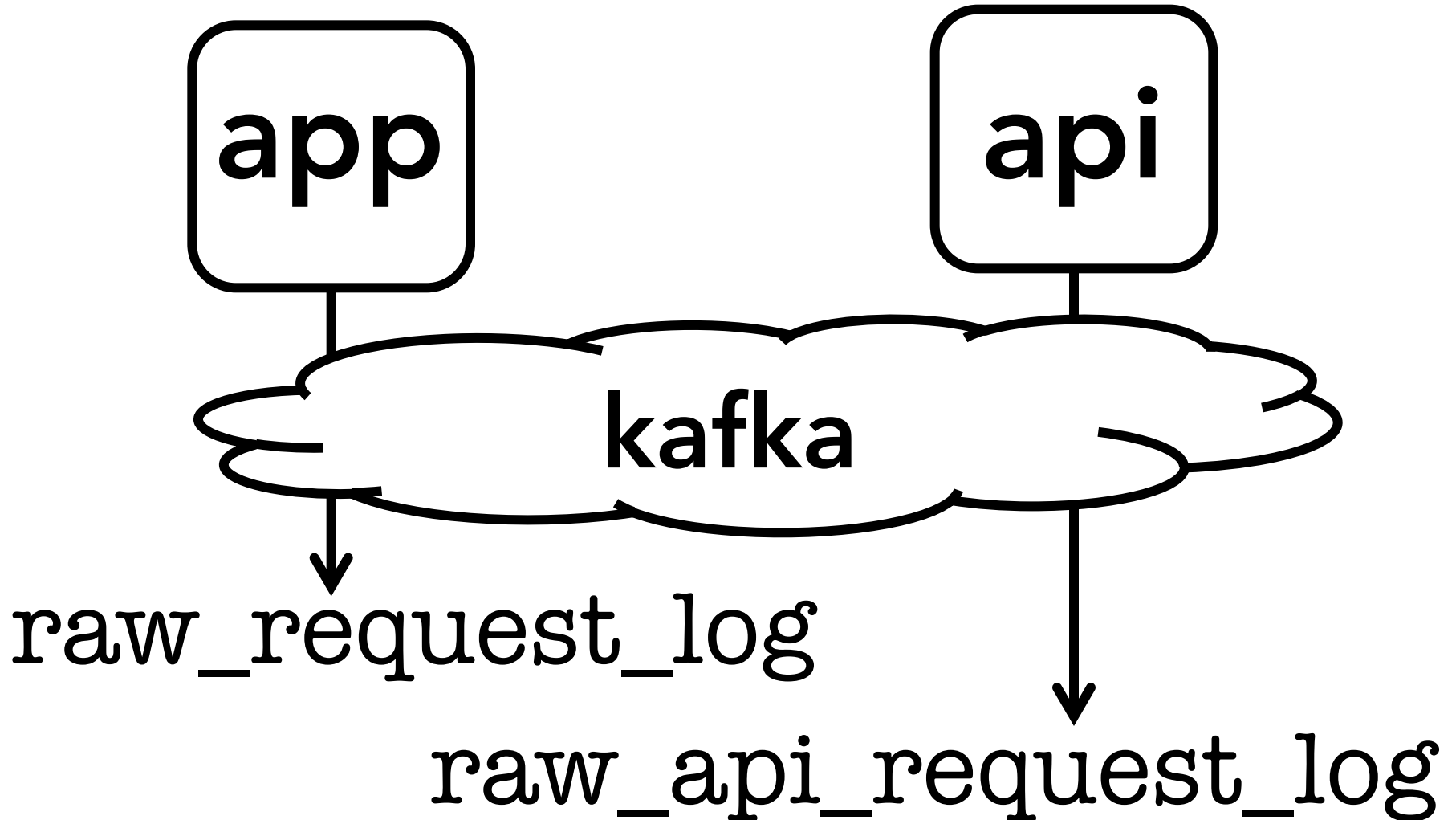




# Part III: avoiding data chaos

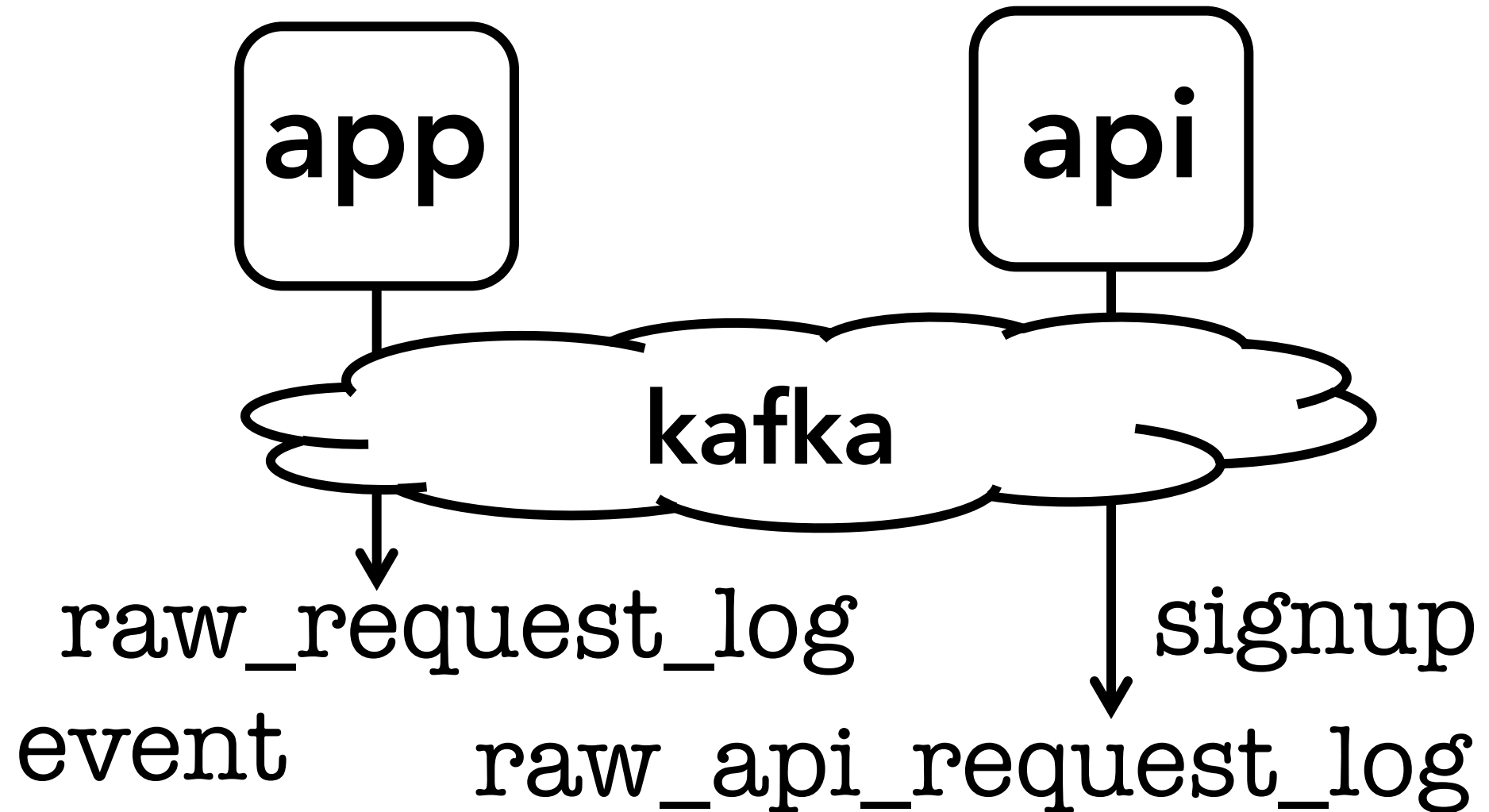
*data, data everywhere and not a drop to drink!*

# Daily Active Users

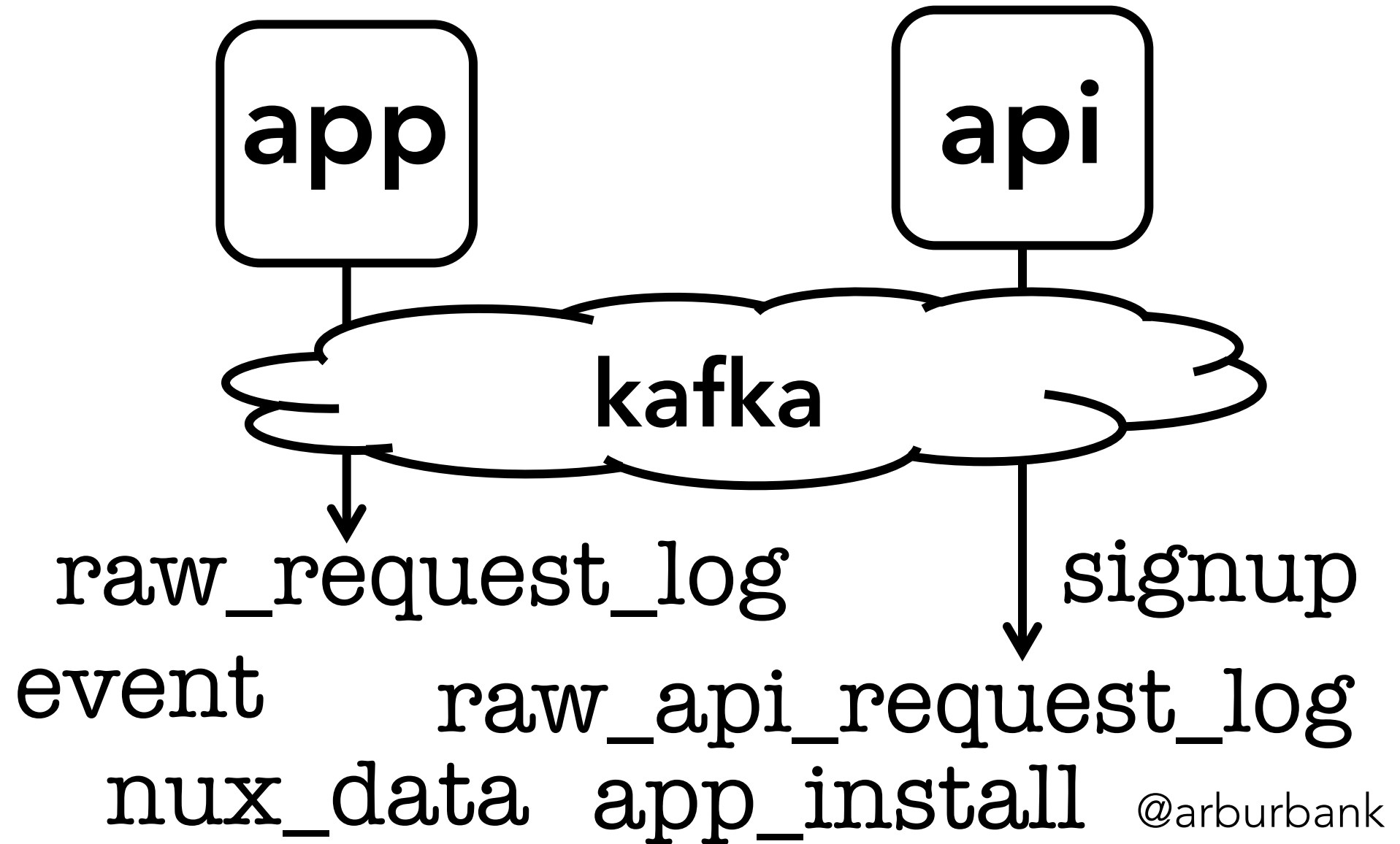


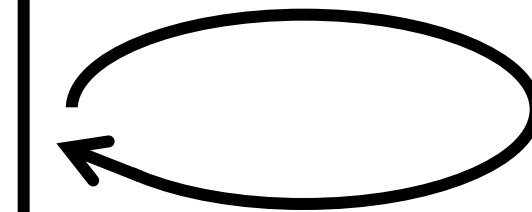
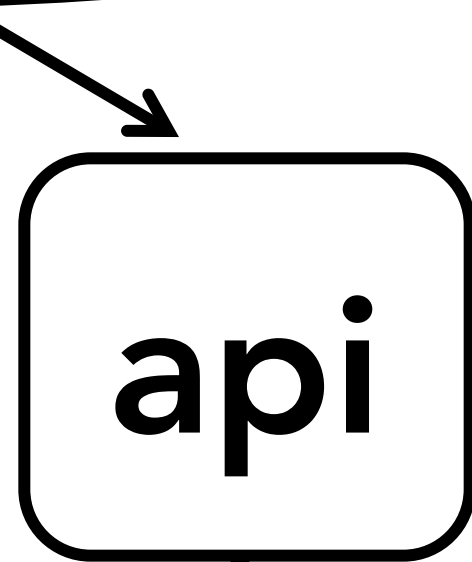
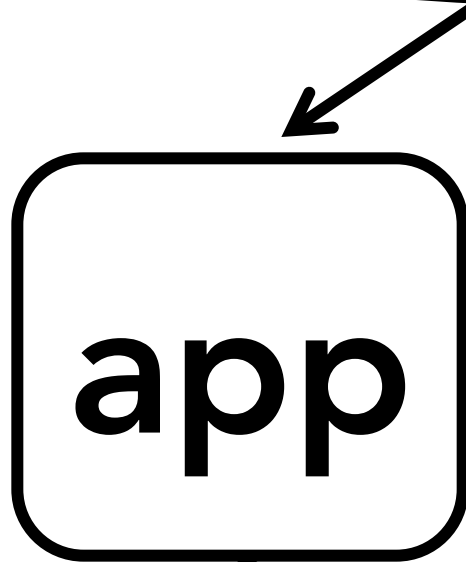


# Daily Active Users



# Daily Active Users





log  
log

log  
log

log  
log

log  
log

# kafka

log  
log

log  
log

log  
log

log  
log



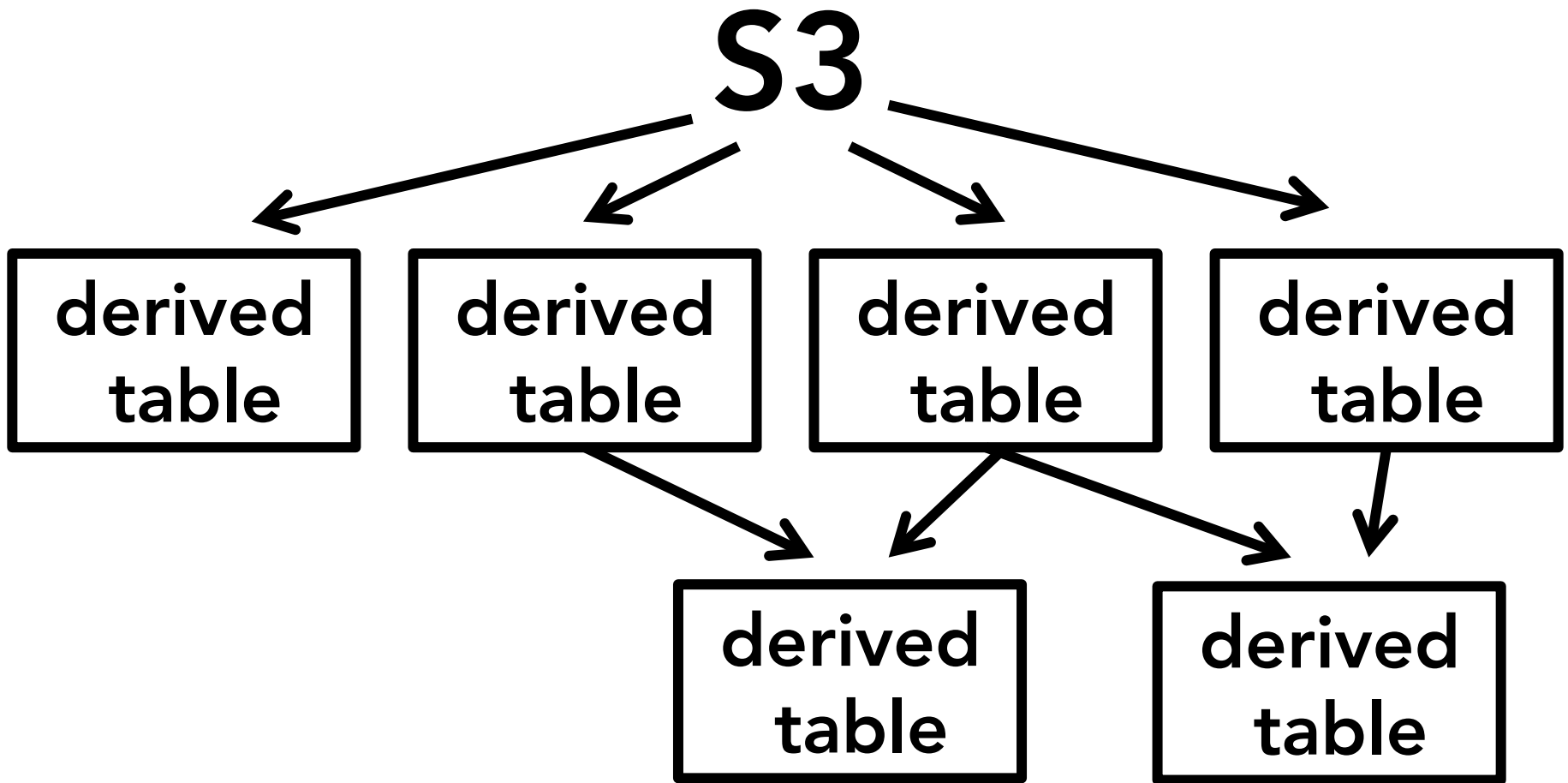
## S3

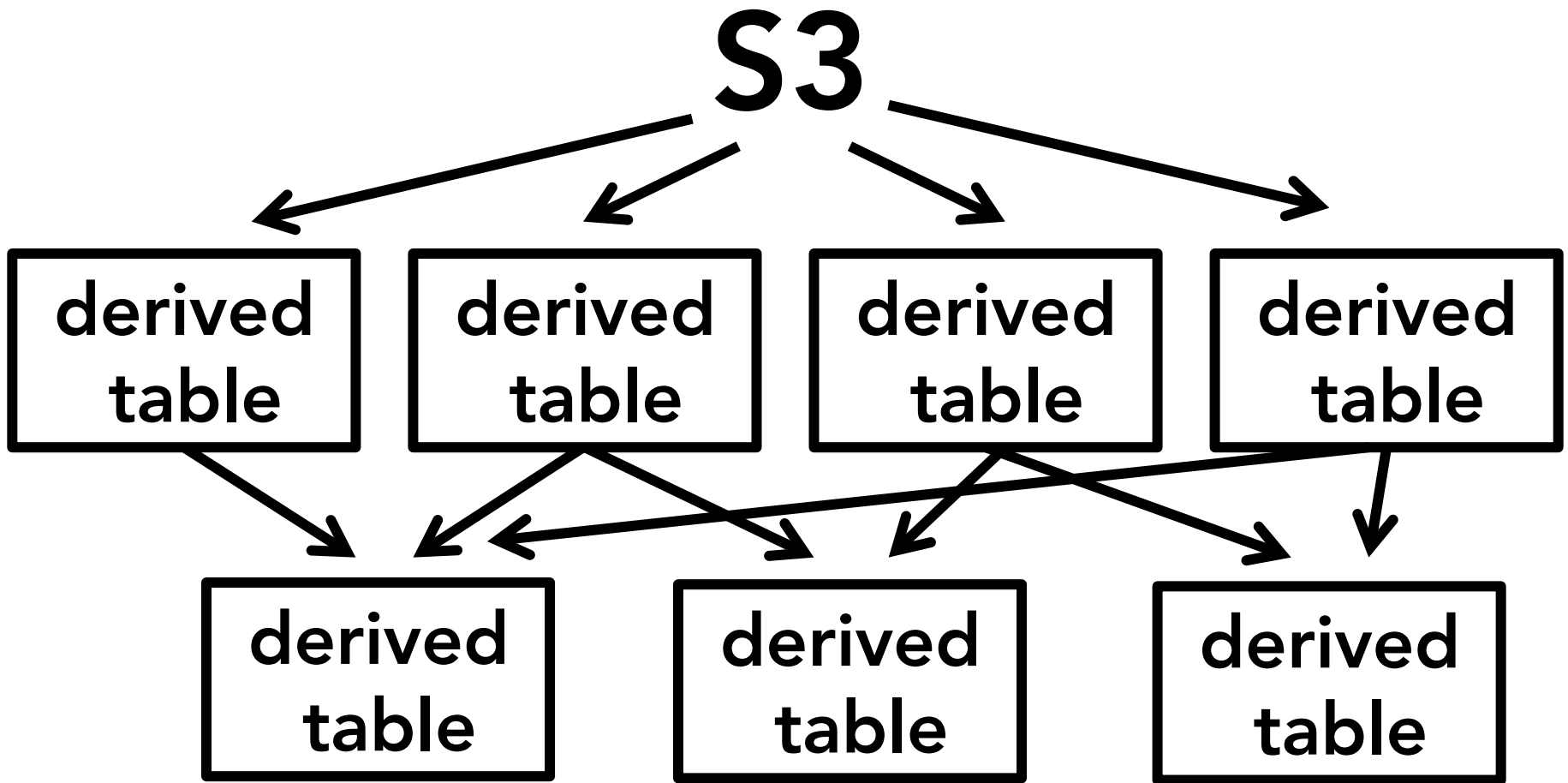
derived  
table

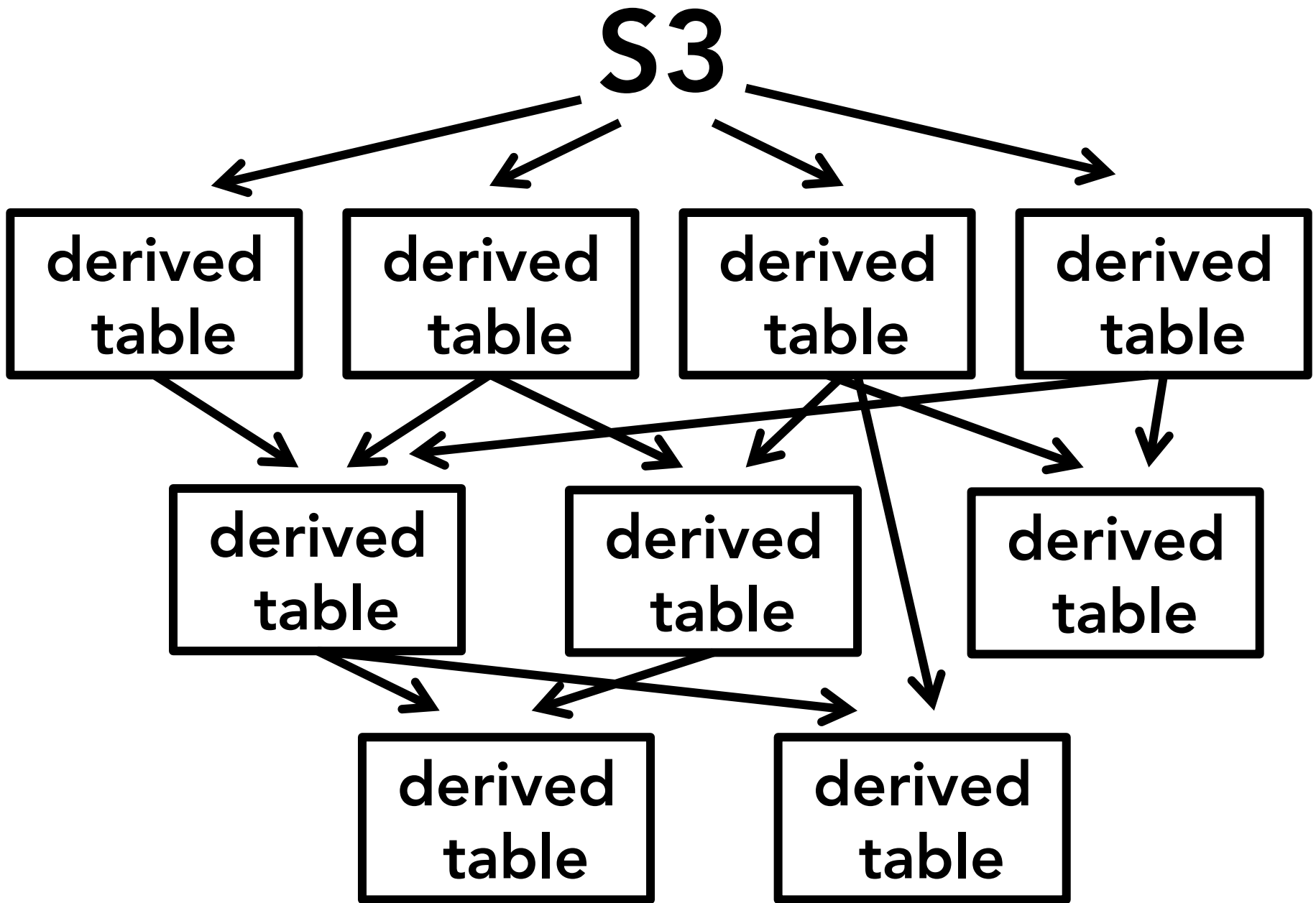
derived  
table

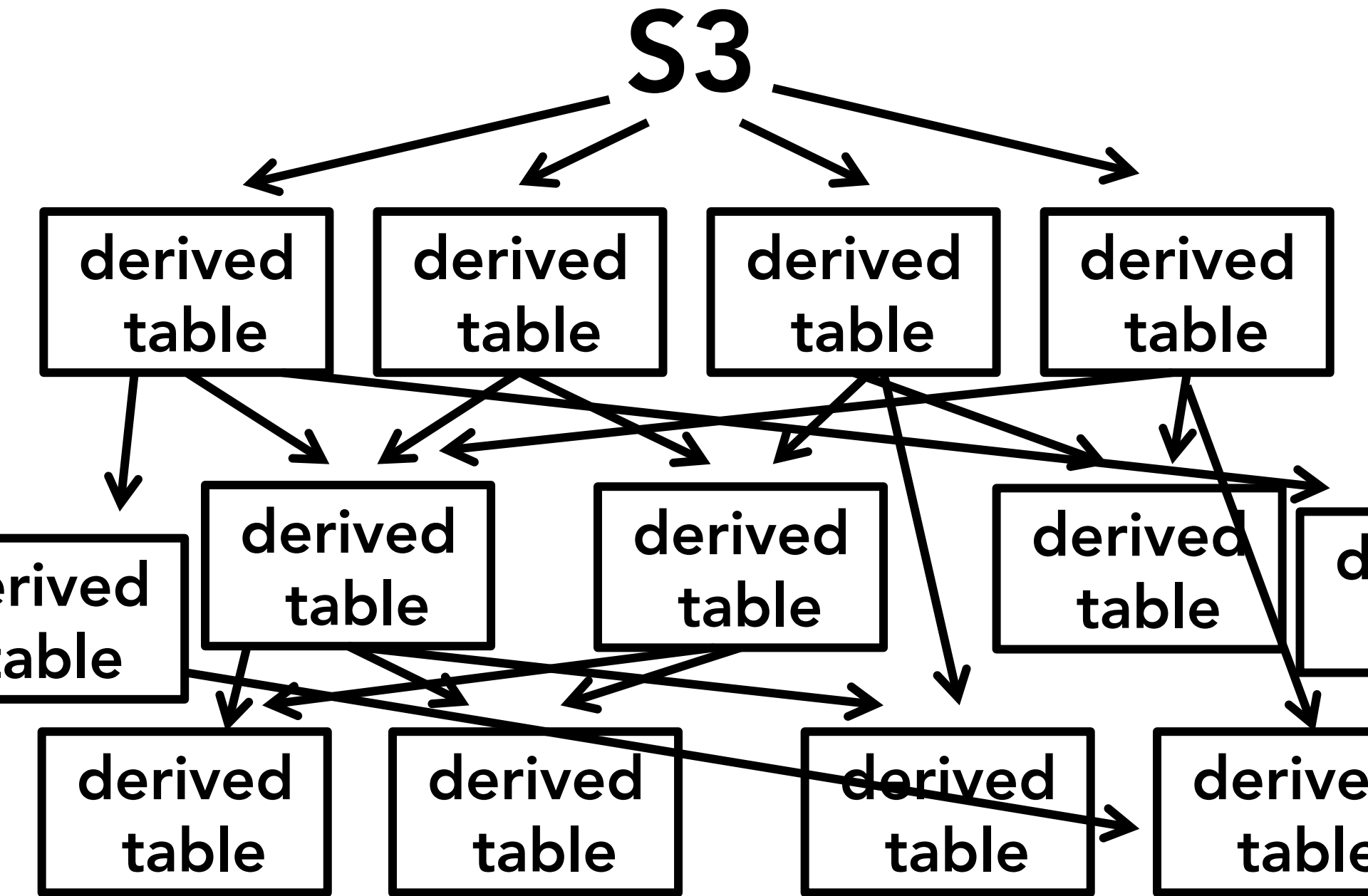
derived  
table

derived  
table



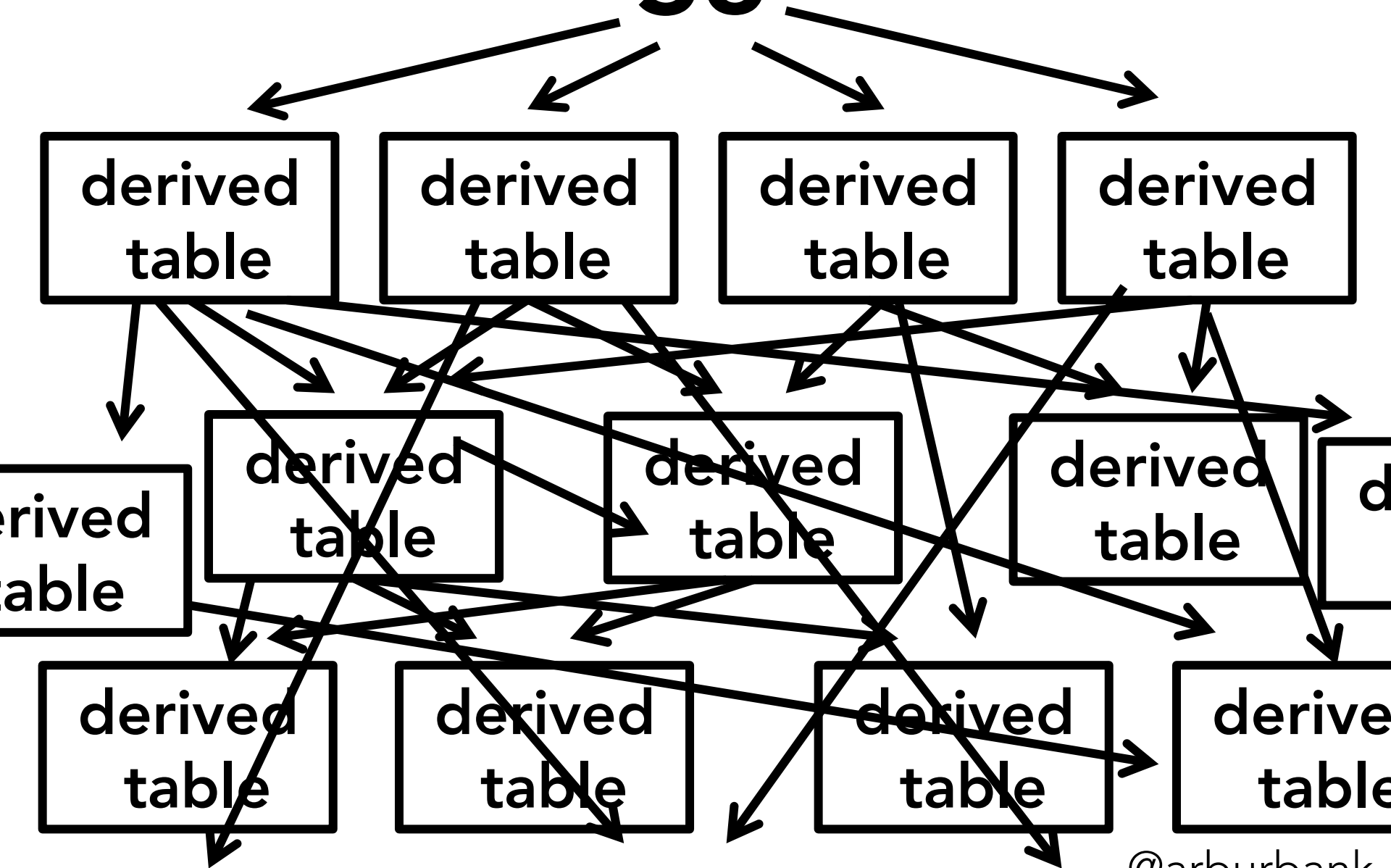




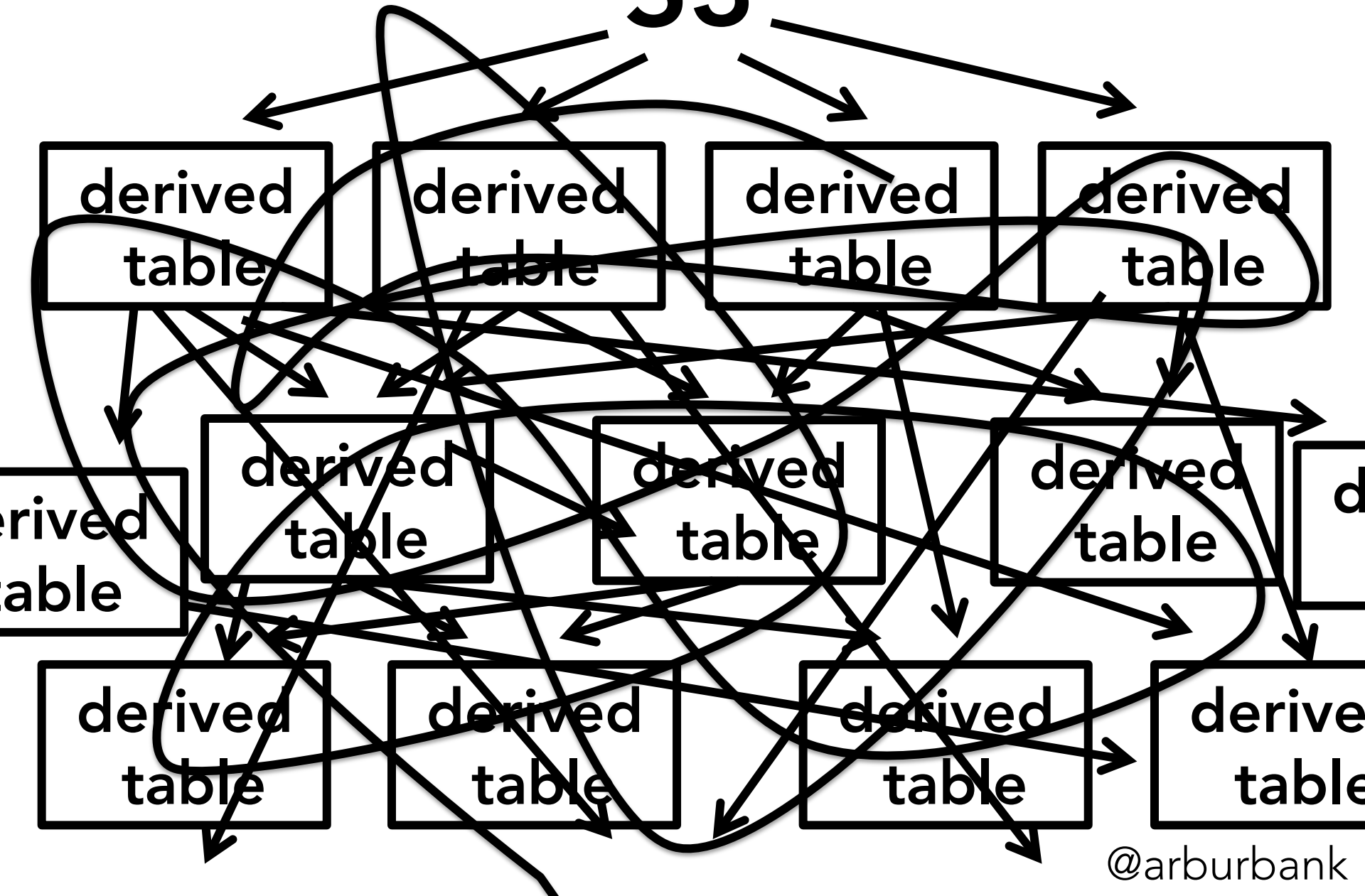




# S3



# S3



# Data chaos: table proliferation

# Data chaos: JSON

```
{'user_id': 23987345,  
  'details': 'chocolate cake',  
  'link': 'allrecipes.com/devils-food-cake/',  
  'image_url': '/378340.png',  
  'parent_id': 109327598,  
  'private': false,  
}
```

# Data chaos: JSON

```
{'user_id': 23987345,  
  'details': 'chocolate cake',  
  'link': 'allrecipes.com/devils-food-cake/',  
  'image_url': '/378340.png',  
  'parent_id': 109327598,  
  'private': false,  
  'any_key_we_want': 'foobar',  
}
```

# Data chaos: JSON

```
{'user_id': 23987345,  
  'details': 'chocolate cake',  
  'link': 'allrecipes.com/devils-food-cake/',  
  'image_url': '/378340.png', deprecated  
  'parent_id': 109327598,  
  'private': false,  
  'any_key_we_want': 'foobar',  
}
```

# Data chaos: JSON

```
{'user_id': 23987345,  
  'details': 'chocolate cake',  
  'link': 'allrecipes.com/devils-food-cake/',  
  'image_url': '/378340.png',  
  'parent_id': 109327598,  
  'private': NULL, added later; not backfilled  
  'any_key_we_want': 'foobar',  
}
```

# Data chaos: JSON

`get_json(p.json, 'user_id')`

*very slow*



# Data chaos: JSON

`get_json(p.json, 'user_id')`

*very slow*



do it once & reuse derived table

userid	ts	path	status	dt
3789823	08:37:03.41	/pin/123/	200	2012-06-01
3789823	08:37:07.22	/pin/repin/	200	2012-06-01
1724468	08:37:10.39	/user/8en/	200	2012-06-01
8779233	08:37:11.97	/category/art/	200	2012-06-01

# Data chaos: JSON

`get_json(p.json, 'user_id')`

*very slow*



do it once & reuse derived table

userid	ts	path	status	dt
3789823	08:37:03.41	/pin/123/	200	2012-06-01
3789823	08:37:07.22	/pin/repin/	200	2012-06-01
1724468	08:37:10.39	/user/8en/	200	2012-06-01
8779233	08:37:11.97	/category/art/	200	2012-06-01

`{'my_new_key': 'important_data'}`



*update schema;  
write new query*

@arburbank

# Data clarity: thrift

```
struct PinPromotion {  
    // next id: 10  
    1: optional i64 pinId,  
    2: optional i64 position,  
    3: optional binary insertionId,  
    4: optional i64 pinPromotionId,  
    5: optional i64 promoterId,  
    6: optional PinPromotionSource source,  
    7: optional i64 userId,  
    8: optional i64 advertiserPinterestId,  
    9: optional i64 gPinPromotionId,  
}
```

# Data clarity: thrift

```
struct PinPromotionsActionEvent {  
    // next id: 10  
    1: required i64 timestamp,  
    2: required binary actionId,  
    3: optional ActionType actionType,  
    4: optional i64 userId,  
    5: optional PinPromotion pinPromotion,  
    6: optional bool isFirstOrder,  
    7: optional event_if.Event event,  
    8: optional event_if.PinImpression impression,  
    9: optional string host,  
}
```

# Data clarity: thrift

```
struct PinPromotionsActionEvent {  
    // next id: 10  
    1: required i64 timestamp,  
    2: required binary actionId,  
    3: optional ActionType actionType,  
    4: optional i64 userId,  
    5: optional PinPromotion pinPromotion,  
    6: optional bool isFirstOrder,  
    7: optional event_if.Event event,  
    8: optional event_if.PinImpression impression,  
    9: optional string host,  
}
```

# JSON vs. thrift

untyped

strongly typed

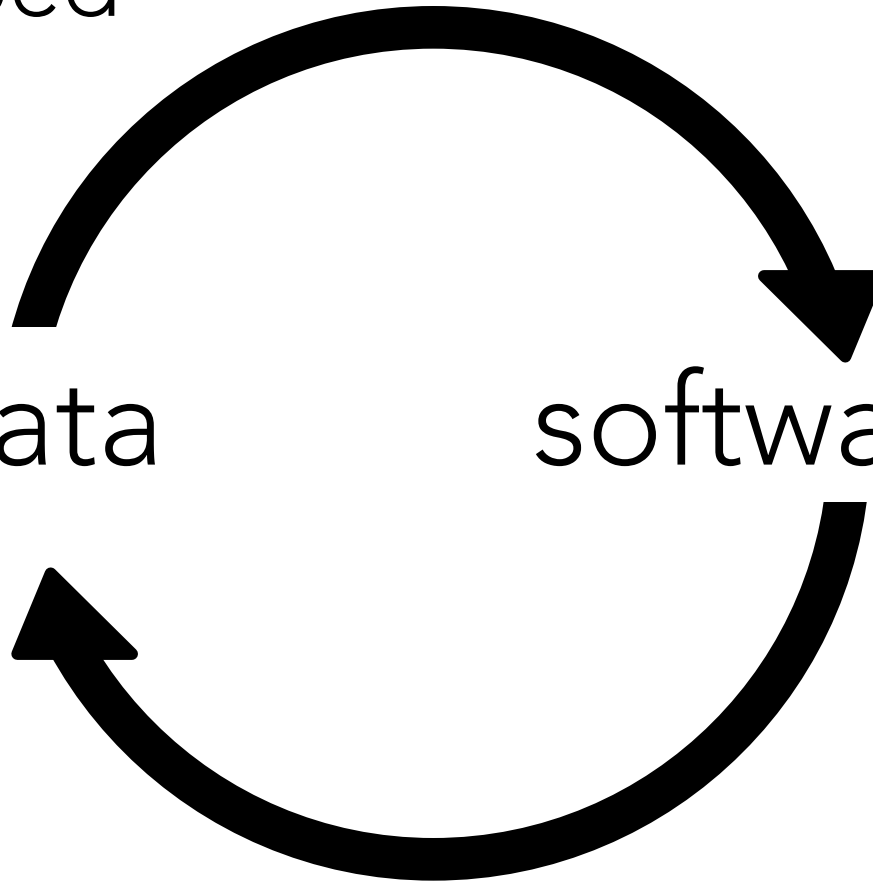
# JSON vs. thrift

untyped

strongly typed

data

software



# Data chaos

proliferation of tables

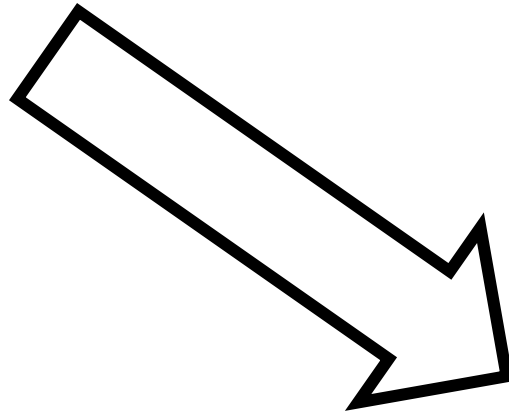
no clear data ownership

duplicated workstreams

lack of schema



**cHaos**



**order**

# data governance

- Central data repository
- Key shared tables across the org
- Deprecate unused tables
- Tests for data accuracy
- Self-documenting tables

# ~~data~~ governance

- Central ~~data~~ repository
- Key shared ~~tables~~ across the org
- Deprecate unused ~~tables~~
- Tests for ~~data~~ accuracy
- Self-documenting ~~tables~~

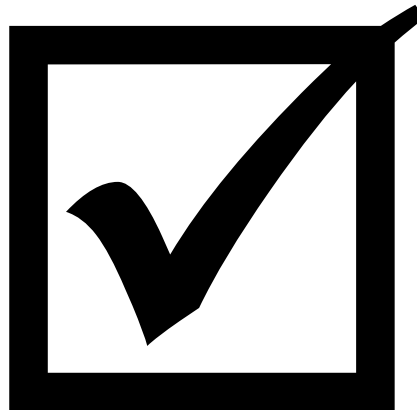
# ~~software~~ ~~data~~ governance

- Central ~~data~~ repository *code*
- Key shared ~~tables~~ across the org *functions*
- Deprecate unused ~~tables~~ *code*
- Tests for ~~data~~ accuracy *code*
- Self-documenting ~~tables~~ *code*



# Part III: avoiding data chaos

*well-structured data, well understood*



**what have we done so far?**

**added kafka logging**

**cloned our databases**



**built workflows**

**defined derived tables**

**removed unimportant requests**

**squashed spammers**

**understood our app**

**ignored phantom requests**

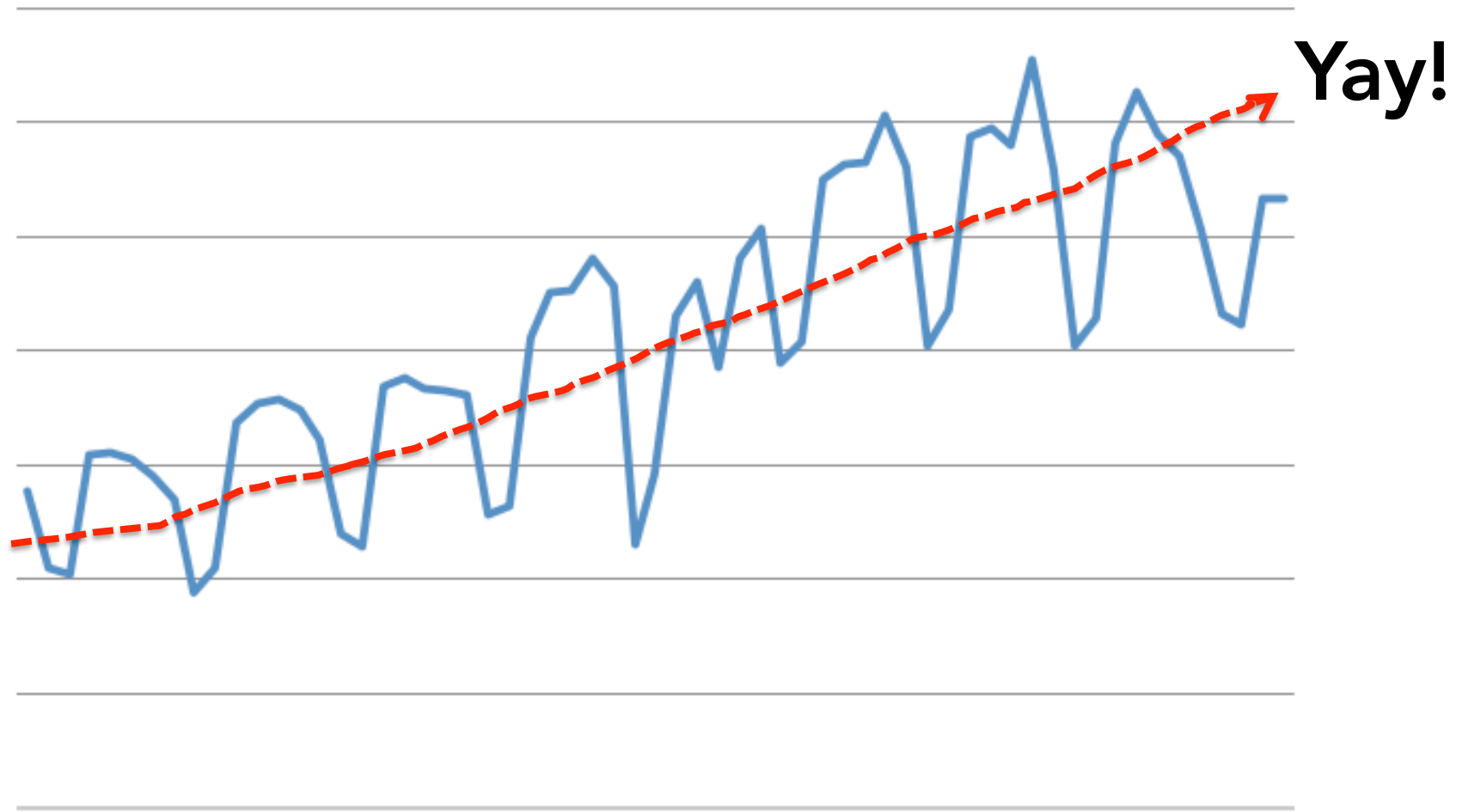
**created data schemas**

**added tests for data accuracy**



**And where did that get us?**

# Daily Active Users

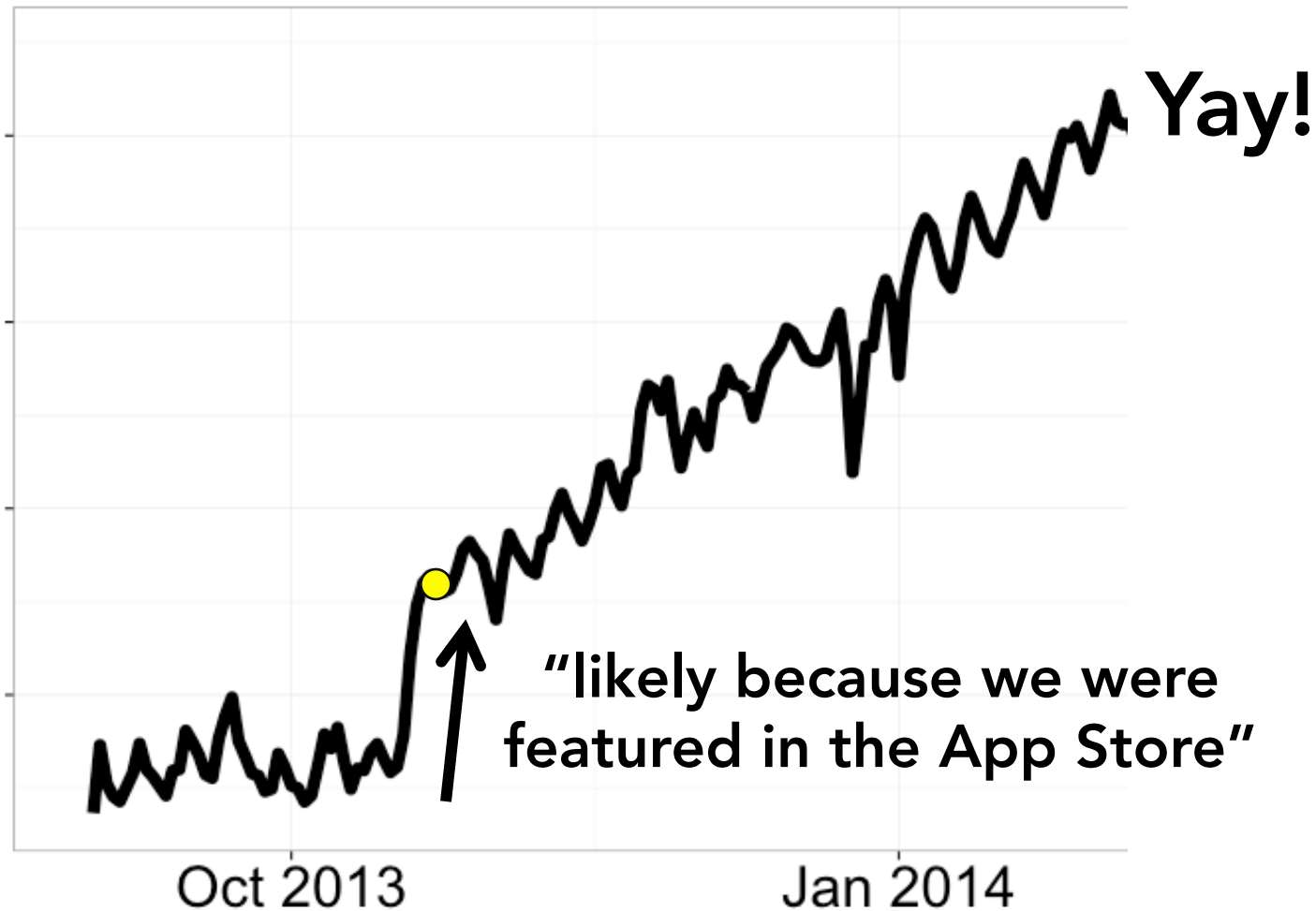




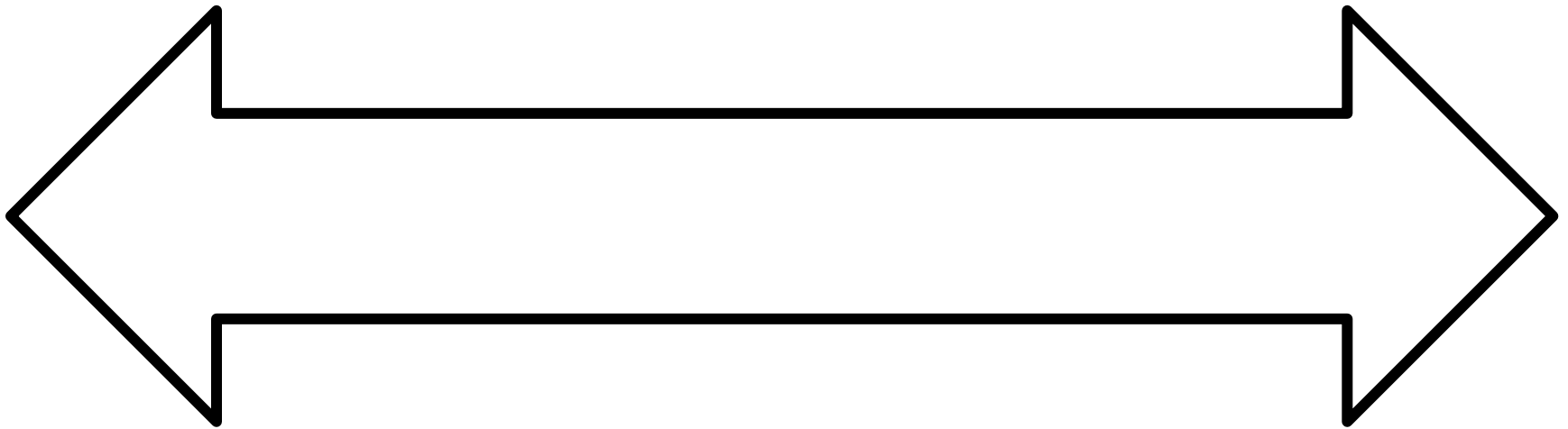
# Part IV: how do we grow?

*counting at internet scale*

# Remember this?



# Spectrum of certainty

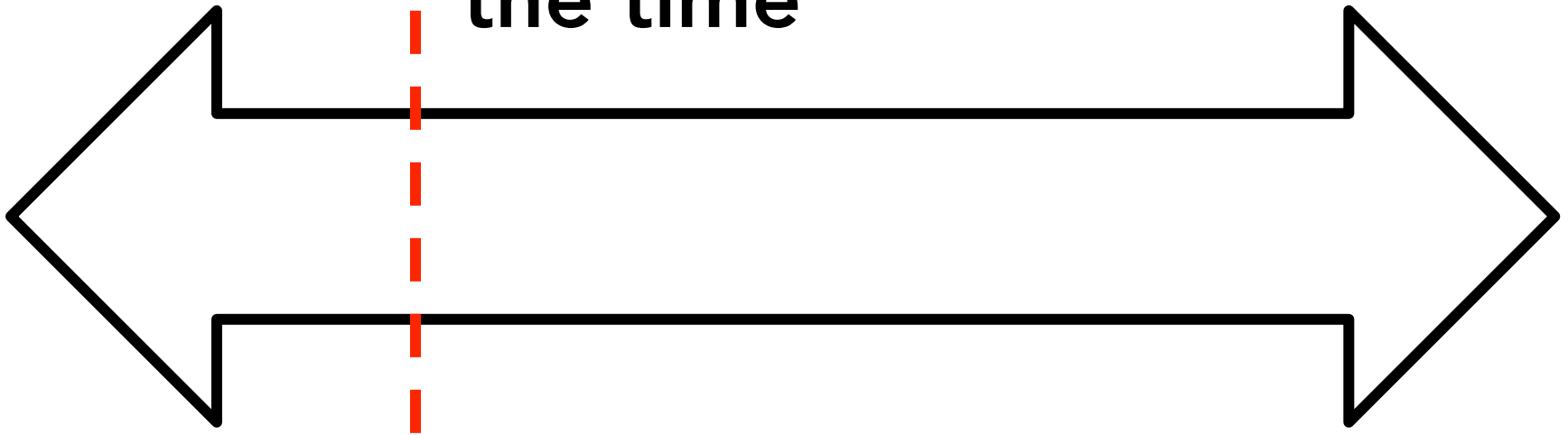


**correlation**

**causation**

# Spectrum of certainty

where we  
are most of  
the time

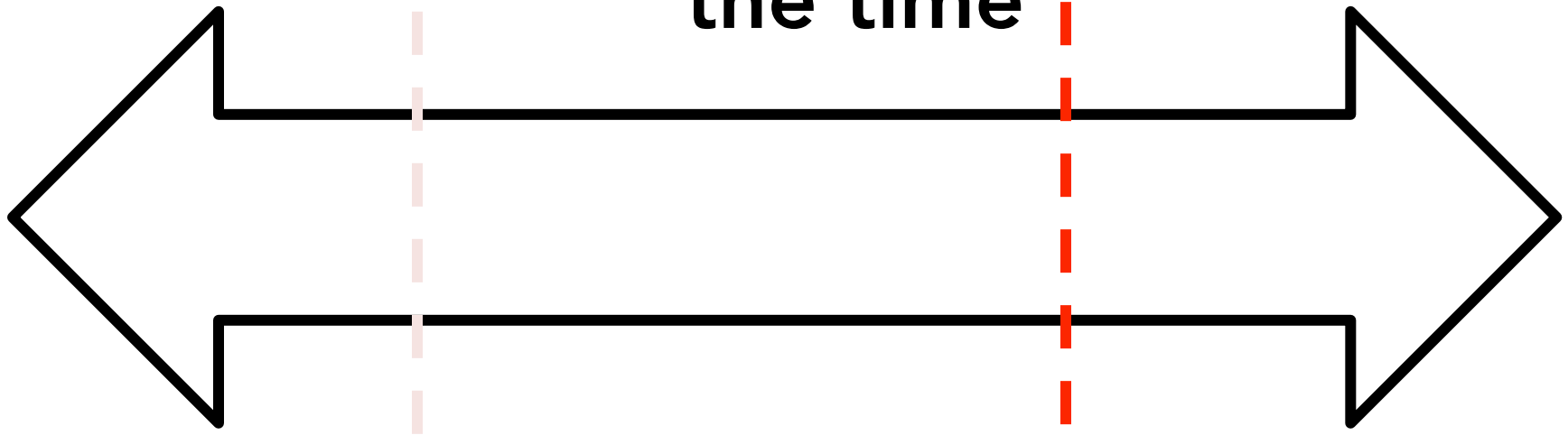


**correlation**

**causation**

# Spectrum of certainty

where we *think*  
we are most of  
the time

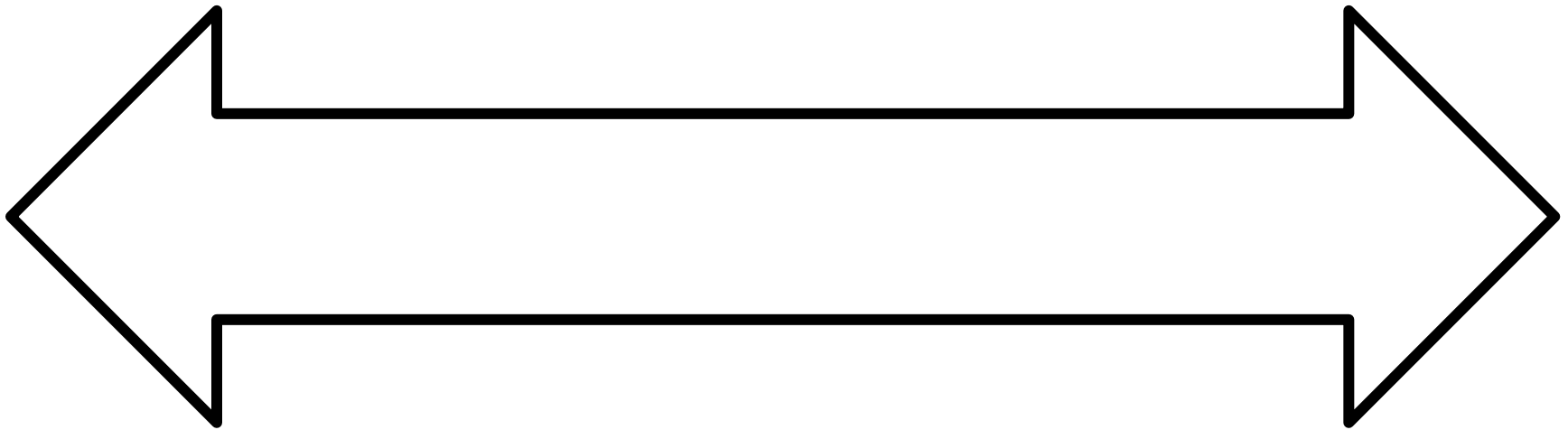


correlation

causation

# Spectrum of certainty

**GOAL:**  
move toward causal inference



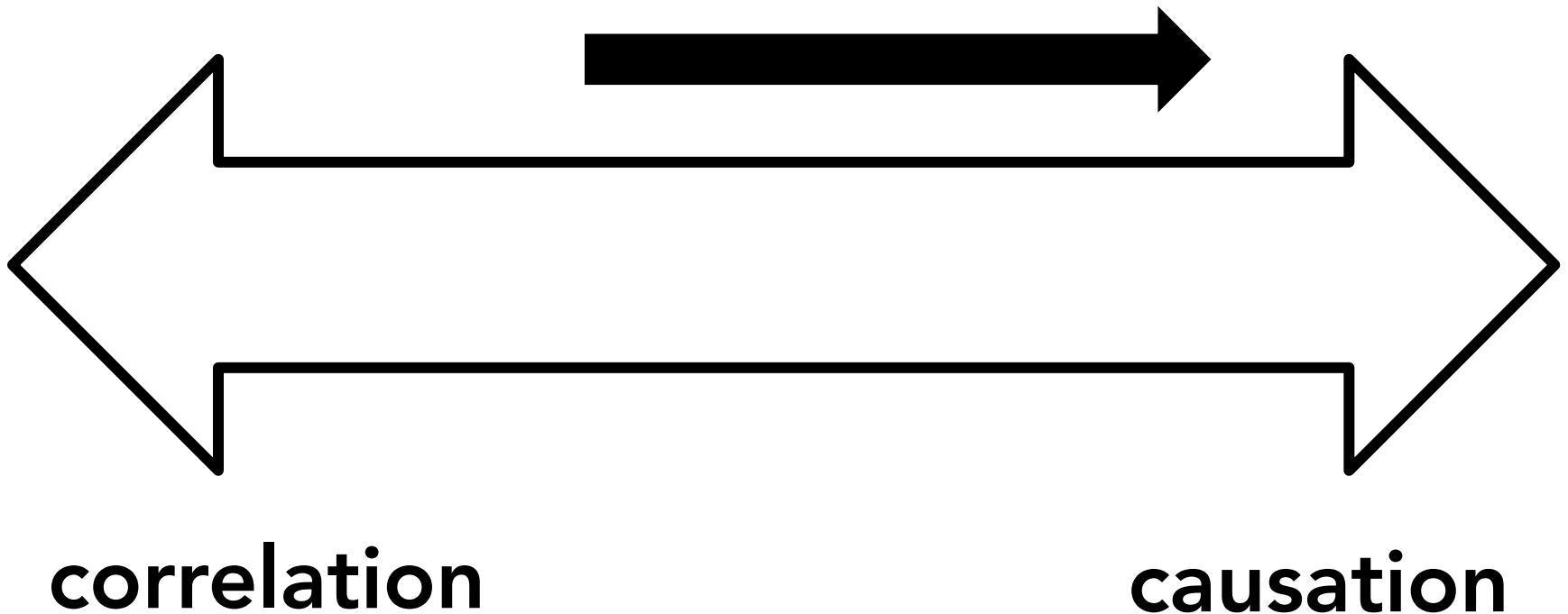
**correlation**

**causation**



# Spectrum of certainty

**GOAL:**  
move toward causal inference

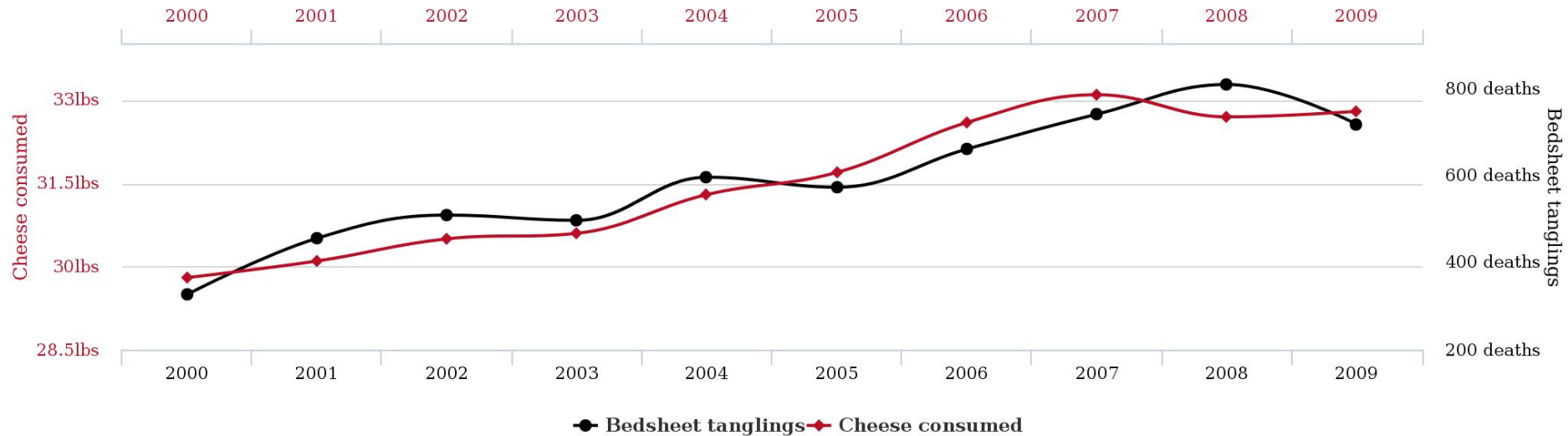


# Spectrum of certainty

**Per capita cheese consumption**

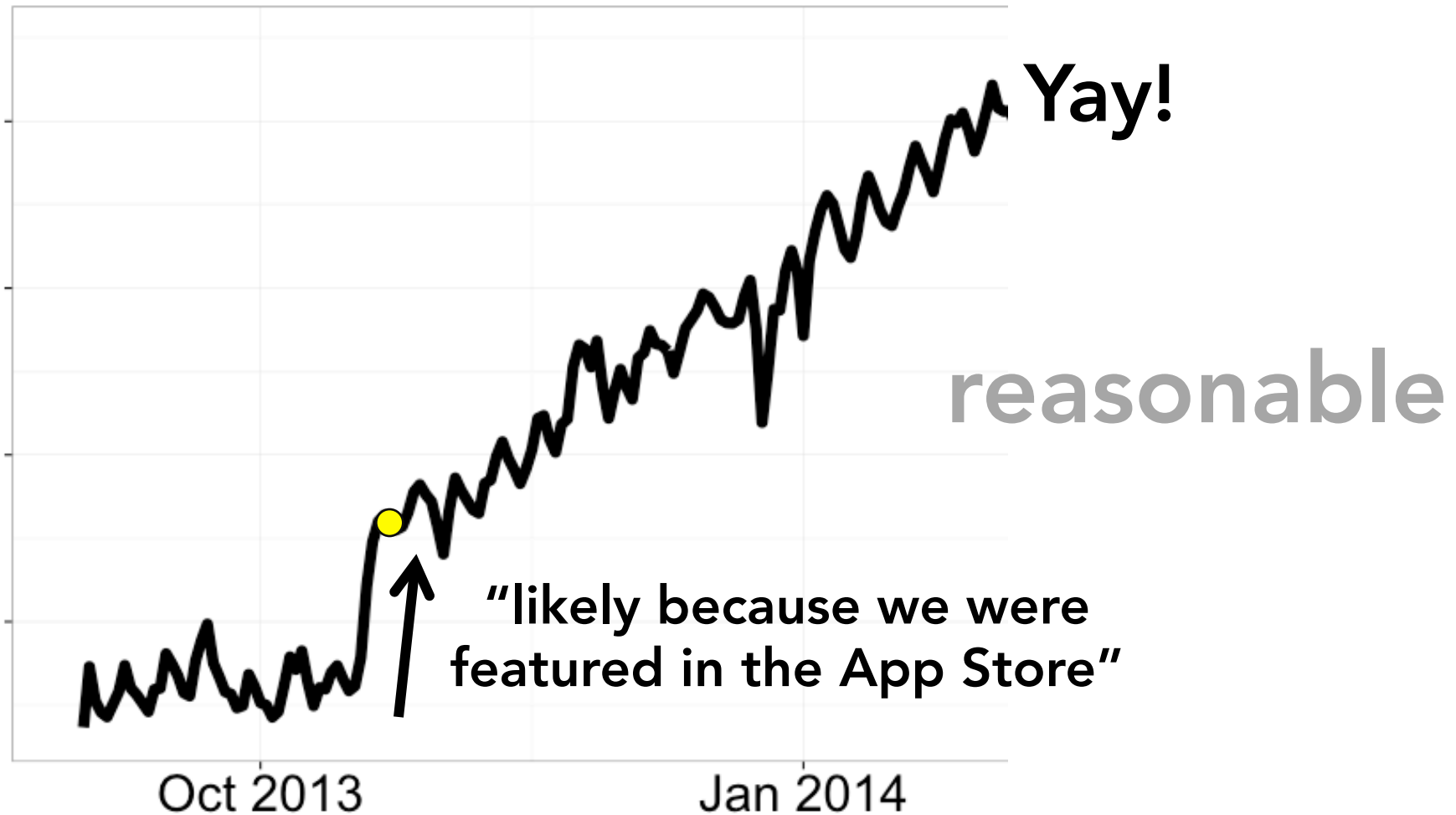
correlates with

**Number of people who died by becoming tangled in their bedsheets**

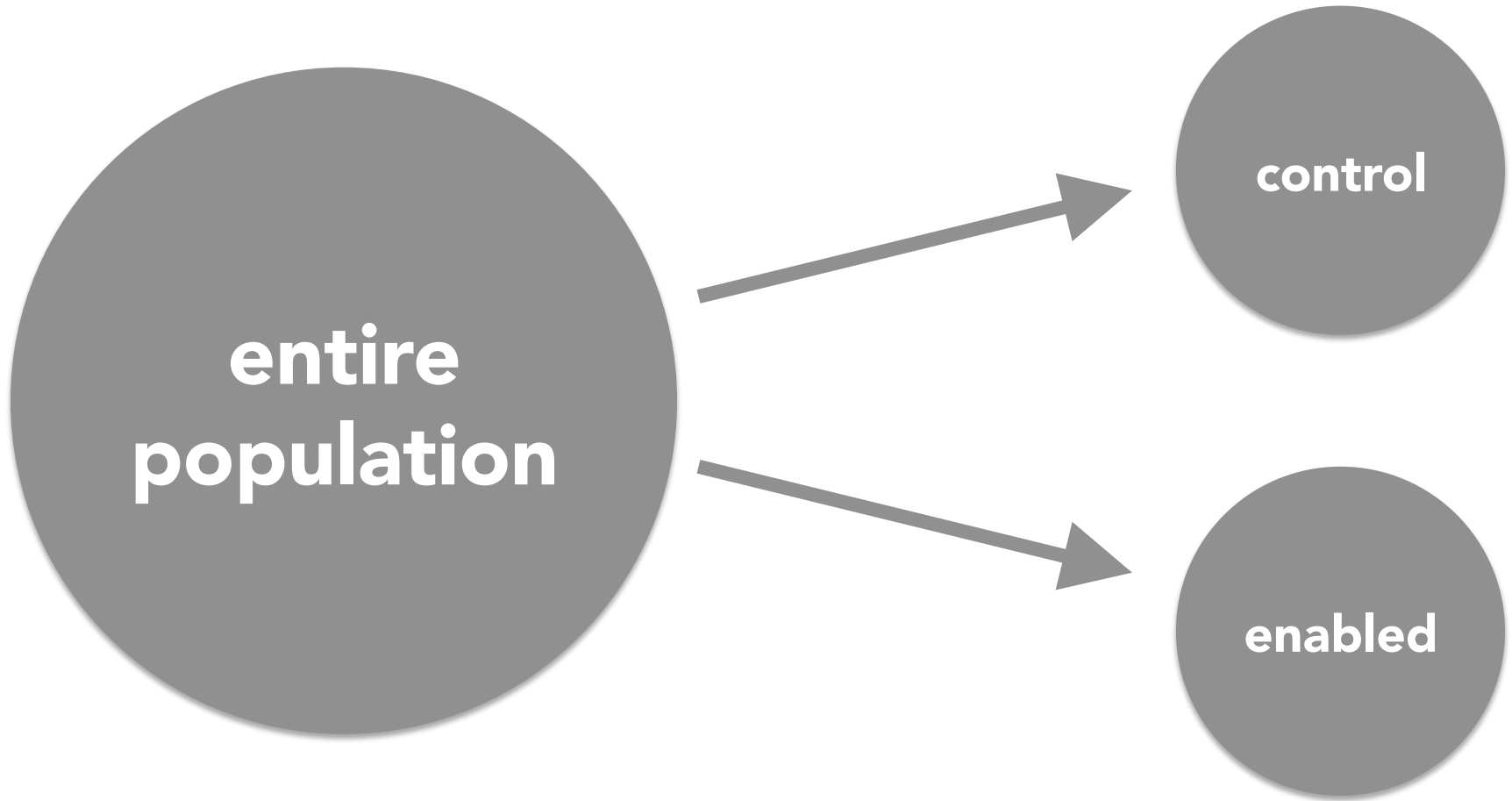


tylervigen.com

# Spectrum of certainty

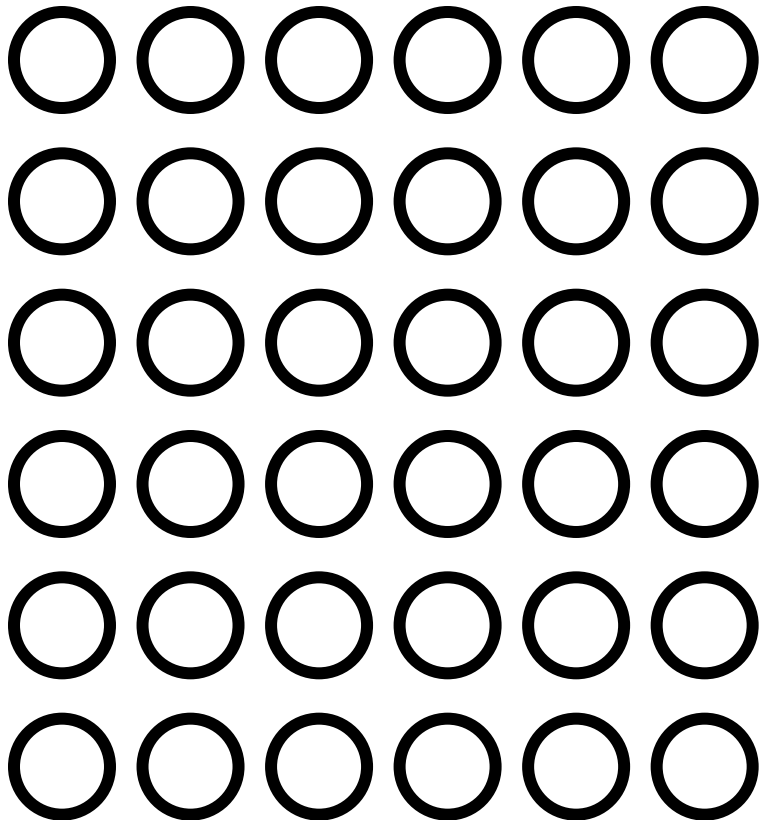


# A/B testing

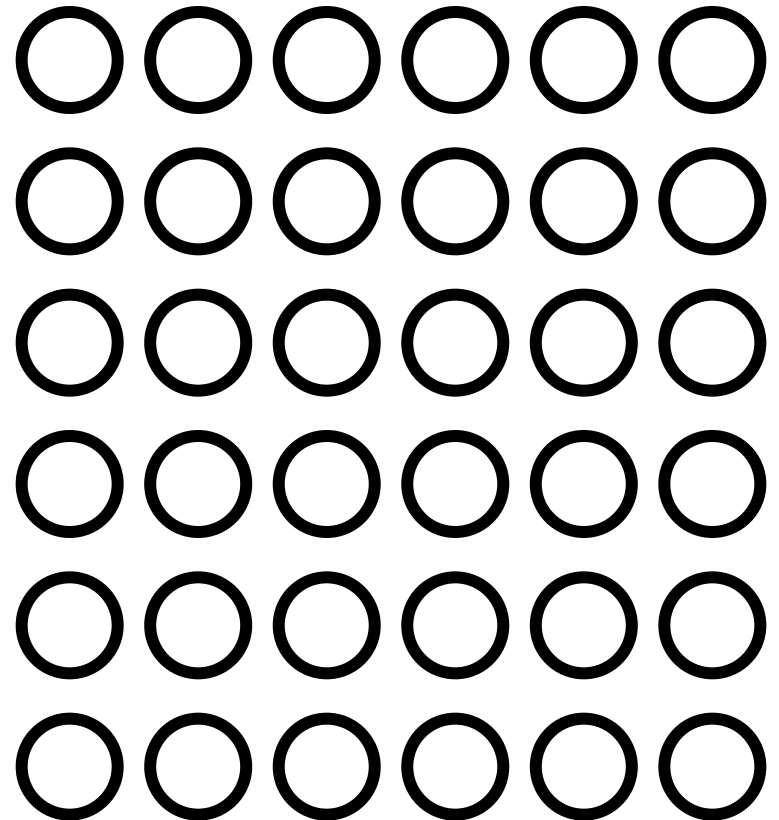


# A/B testing

control

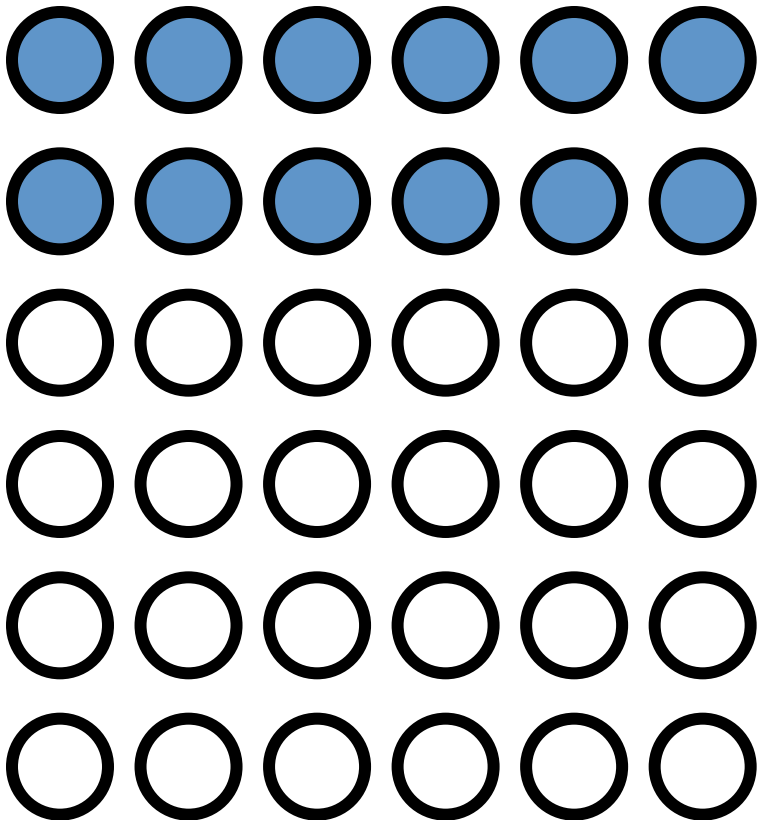


enabled

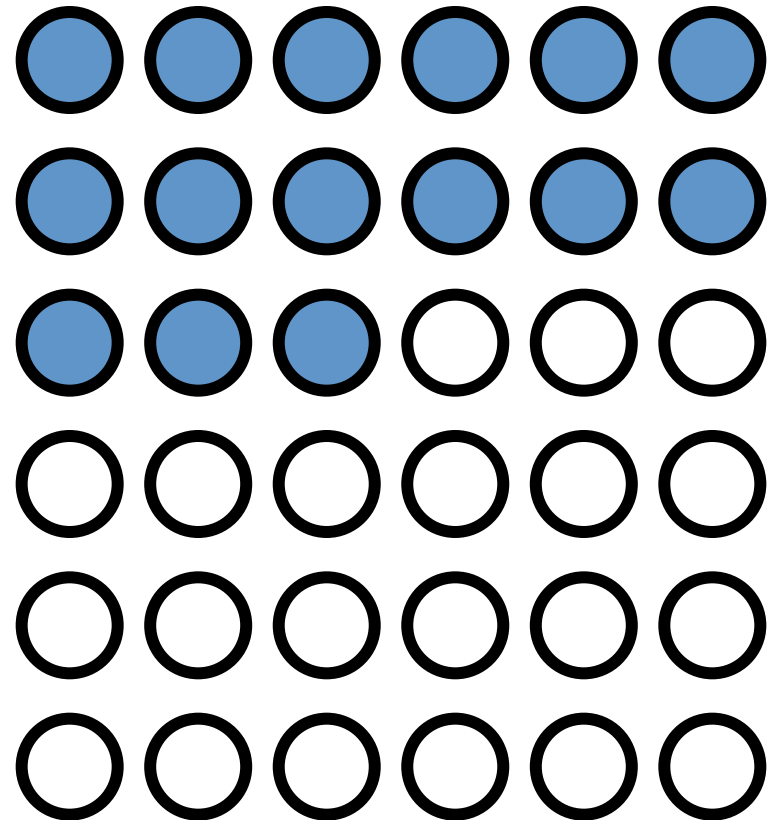


# A/B testing

control

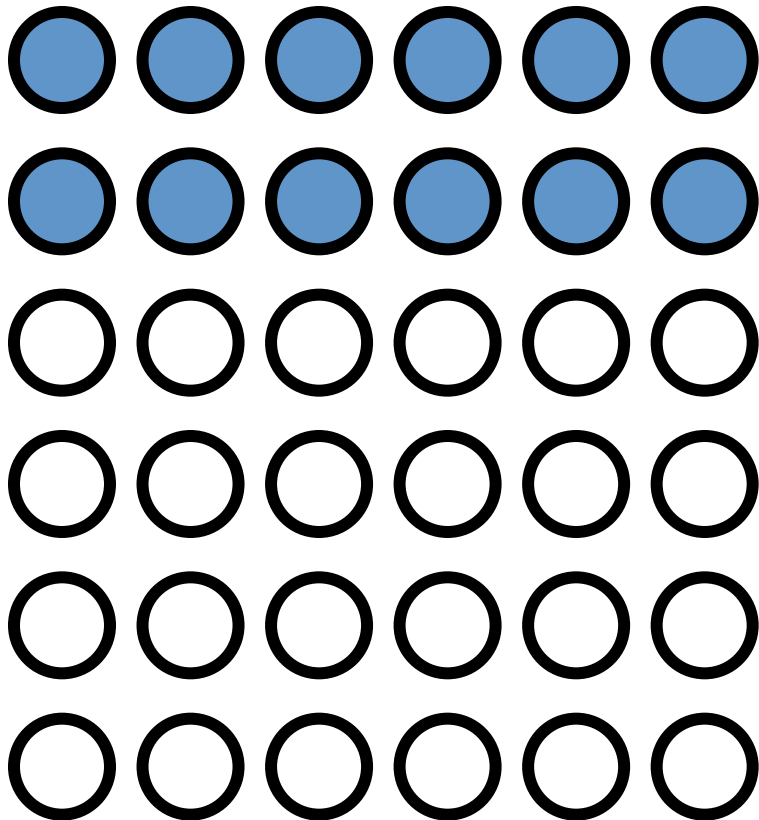


enabled

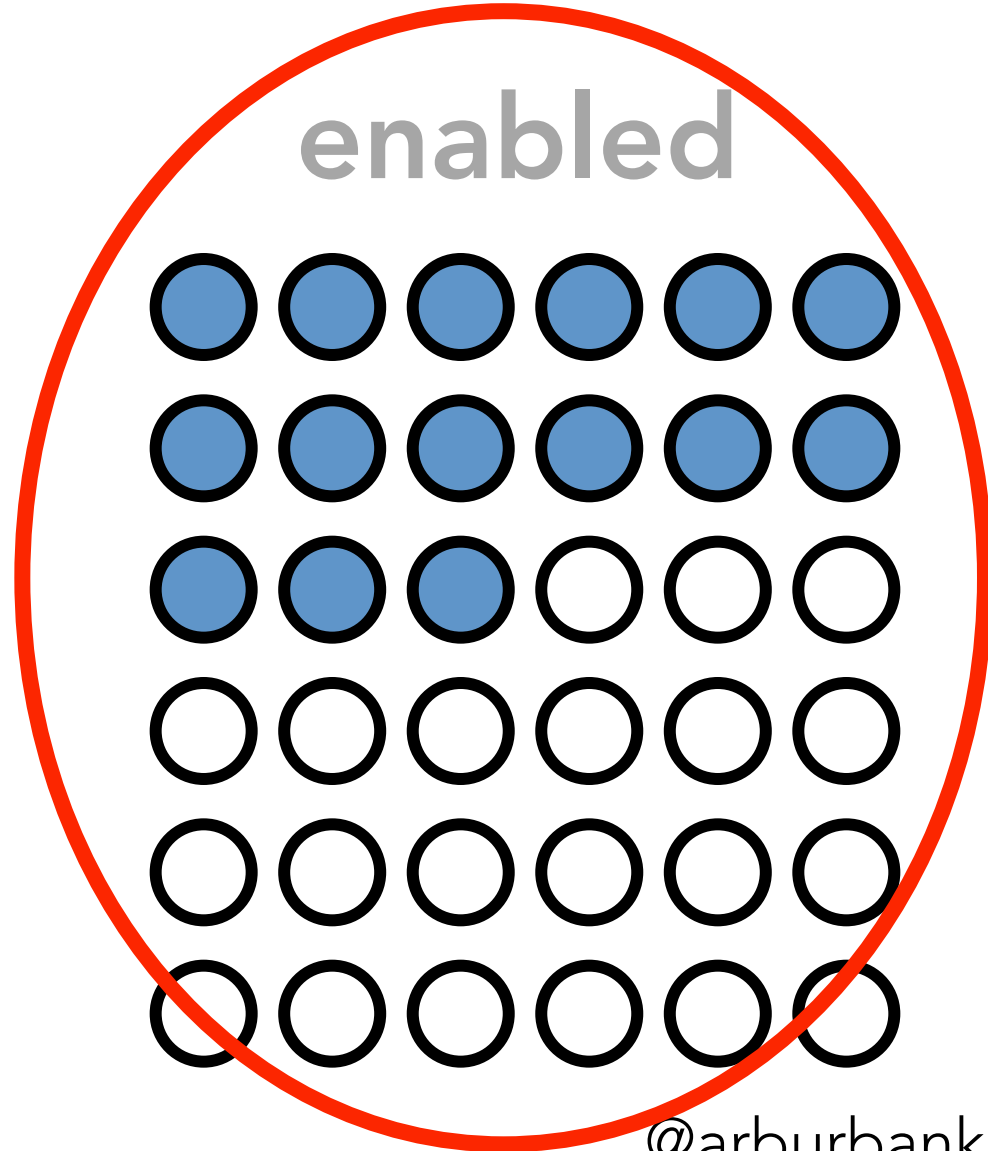


# A/B testing

control

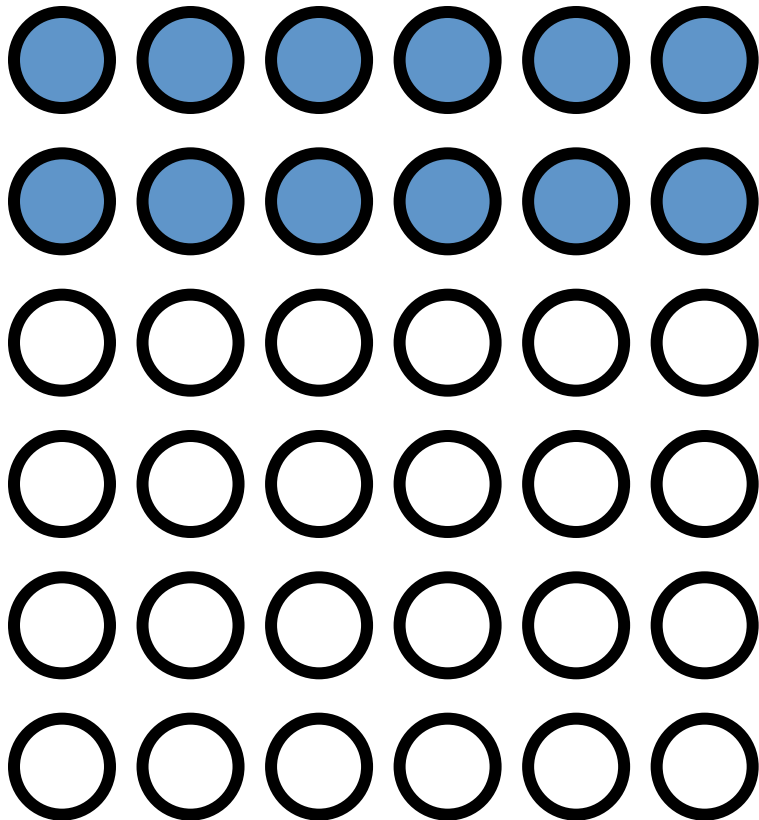


enabled

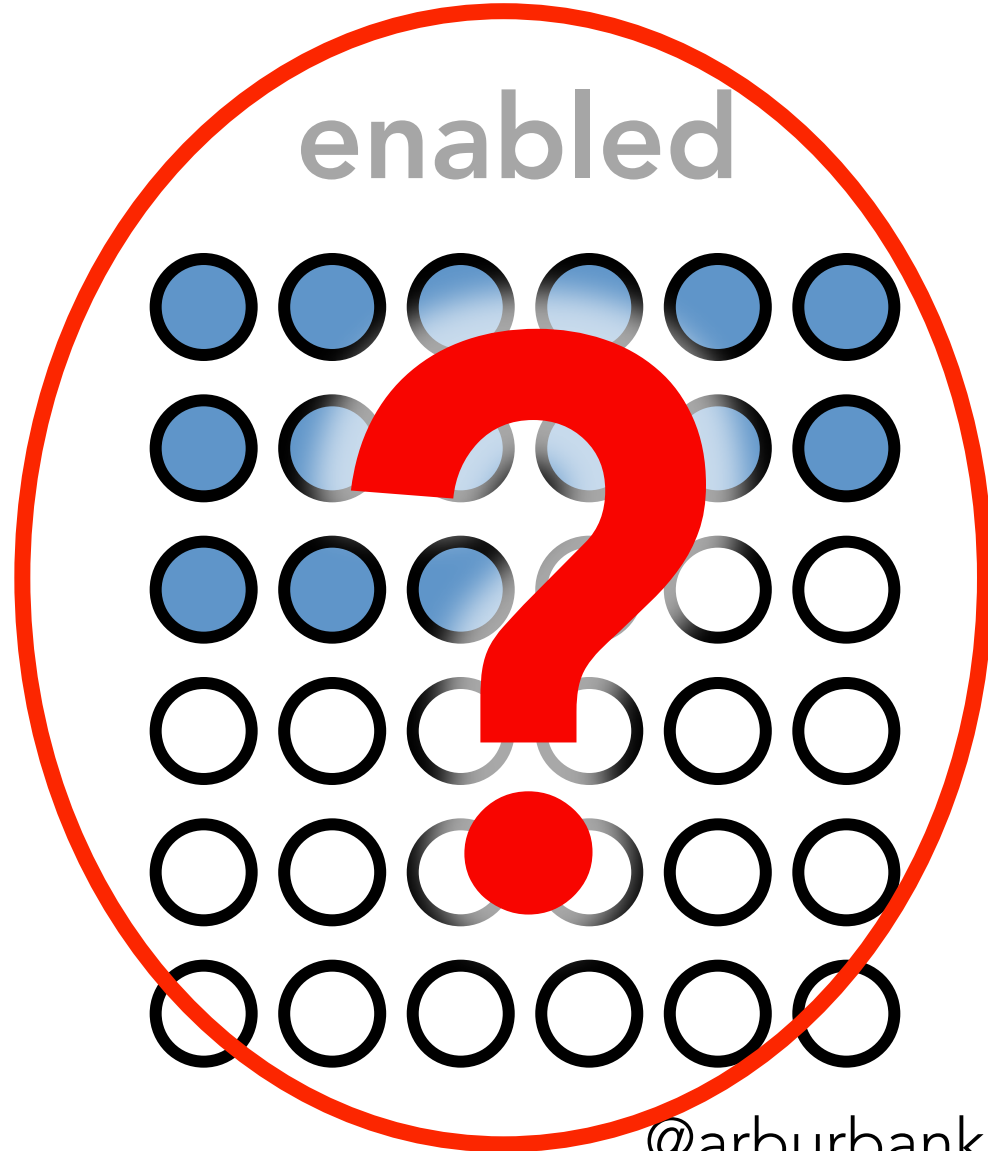


# A/B testing

control



enabled





# A/B testing

control

enabled

**statistical significance**

# A/B testing



**"Statistical  
significance" is  
mumbo jumbo**

A thought bubble with a large cloud-like body and three small circles leading down to the left.

**Insignificant  
effects MIGHT  
NOT BE REAL!**

A thought bubble with a large cloud-like body and three small circles leading down to the right.

# A/B testing

control

enabled

**novelty effects**

# A/B testing



This is new! I wonder what it will do.

A thought bubble with a cloud-like top and three small circles leading down to the left.

Short-term effects may not last ...

A thought bubble with a cloud-like top and three small circles leading down to the right.

# A/B testing

control

enabled

**user segmentation**

# A/B testing



Are some users  
affected more  
than others?

A thought bubble with a scalloped border and three small circles leading down to the main bubble.

It looks negative  
for men in  
Kyrgyzstan ...

A thought bubble with a scalloped border and three small circles leading down to the main bubble.

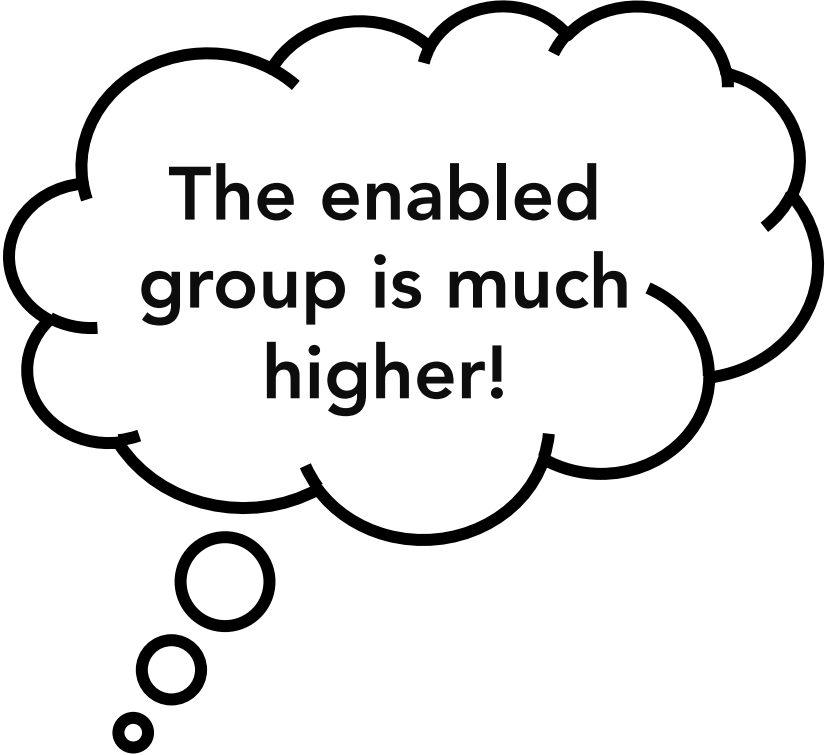
# A/B testing

control

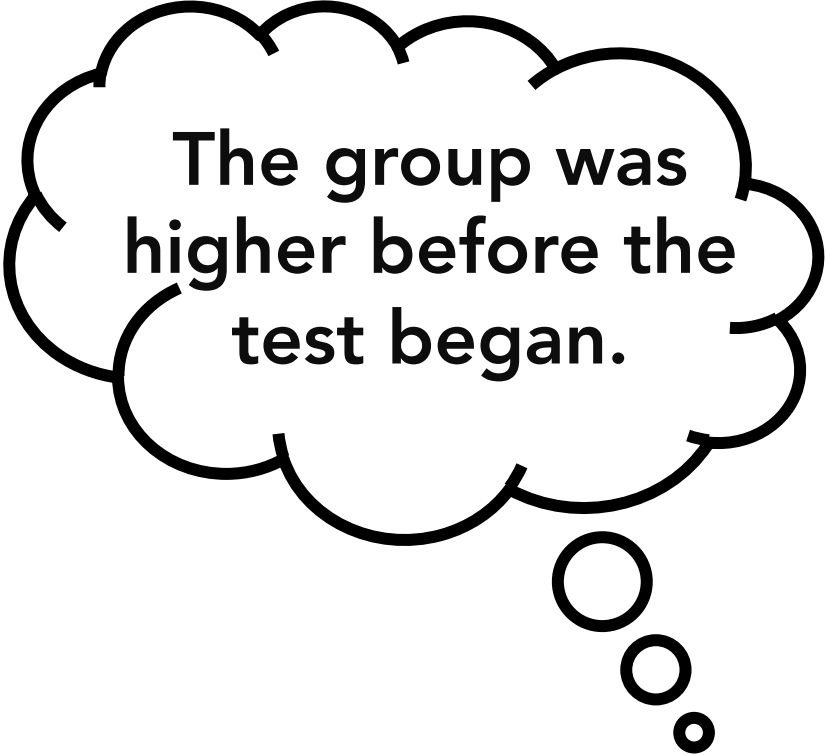
enabled

**randomization errors**

# A/B testing



The enabled group is much higher!



The group was higher before the test began.



# Software tools

Doing the **right** thing should be **easy**

Doing the **wrong** thing should be **hard**

**Build tools that make doing  
the right thing easy**

# Build tools that make doing the right thing easy

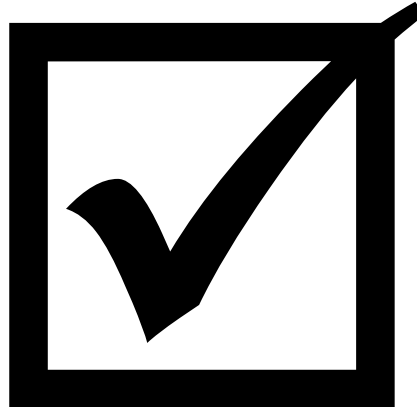
By showing:

- only significant differences
- novelty vs. long-term effects
- important user segments
- randomization errors
- and more ...



# Part IV: how do we grow?

*counting at internet scale*



AMSTERDAM

INTERNATIONAL  
SOFTWARE DEVELOPMENT  
CONFERENCE 2016

goto;  
conference

follow us on @GOTOamst

Workshop: June 13 / Conference: June 14-15

# data as software

@arburbank

# leverage AB testing

**scale insights with tools**

**Doing the right thing  
should be easy**



**Doing the wrong thing  
should be hard**

AMSTERDAM

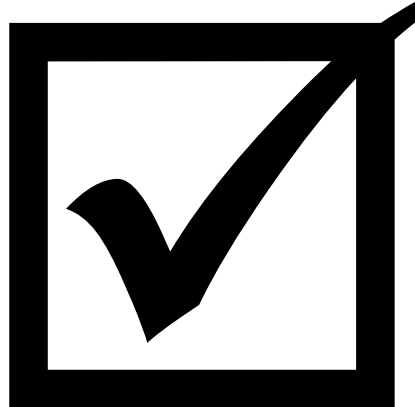
INTERNATIONAL  
SOFTWARE DEVELOPMENT  
CONFERENCE 2016

goto;  
conference

follow us on @GOTOamst

Workshop: June 13 / Conference: June 14-15

# data as software



@arburbank

# Counting is hard

How we got everything wrong

- No logging at all!
- Log every request as DAU
- Time-consuming request parsing
- Spam interfering with metrics
- Background fetches
  - Open browser tabs? Screensavers? Widgets?
- App identification, country attribution, ...

# Data is unruly

Data governance (or lack thereof)

- ad hoc creation of kafka topics
- ad hoc creation of derived tables
- no schema enforcement (arbitrary JSON)
- ever-growing data = \$\$\$\$\$\$

Solutions:

- use thrift to enforce schema
- tools to manage & discover tables
- data retention policies

# Data for scaled decisions

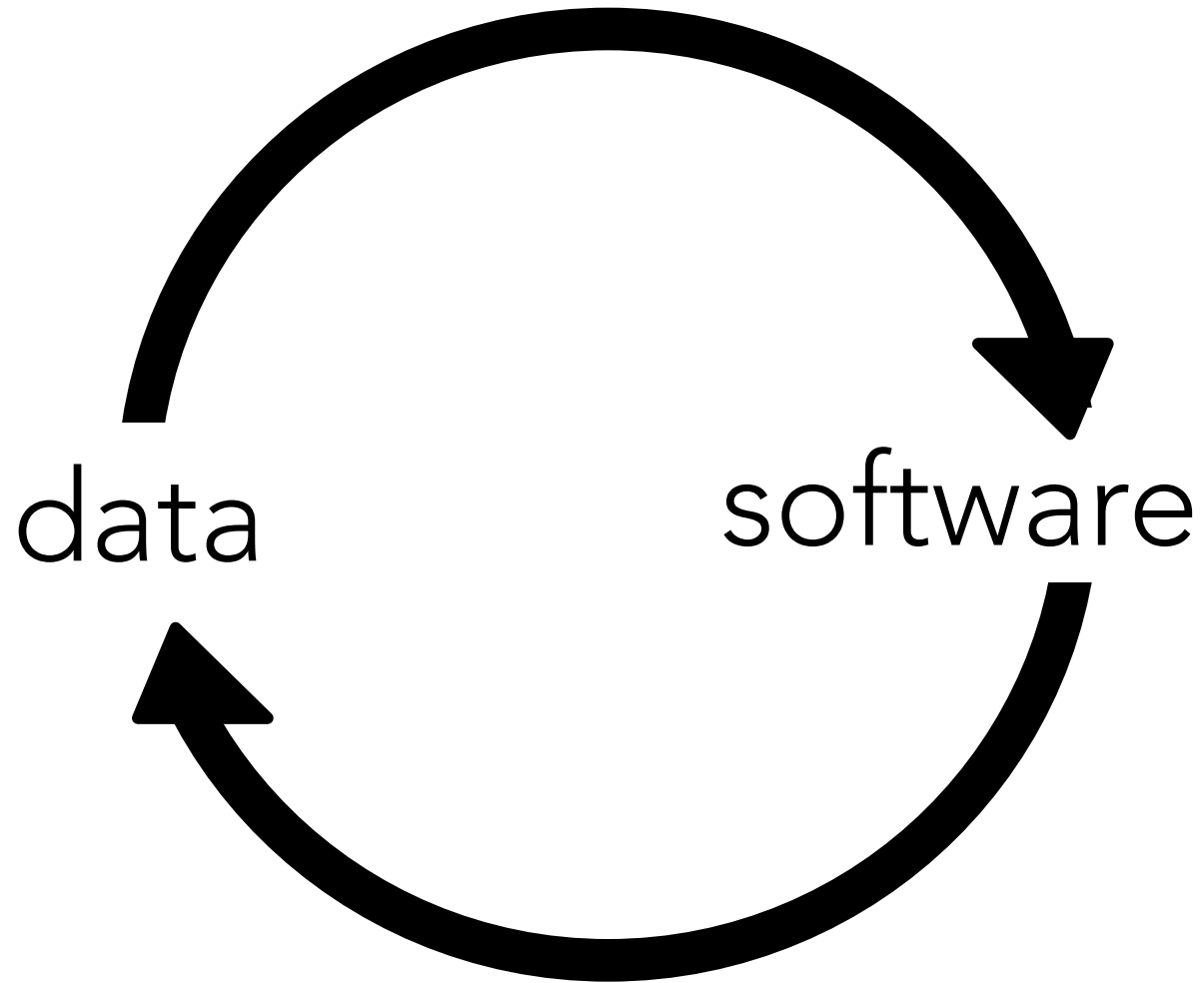
## A/B testing

- Automate metrics collection
  - Make it easy to do the right thing
    - Show which results are statistically significant
    - Estimate population needed to measure effects
    - Show novelty and long-term effects separately
  - Make it hard to do things wrong
    - Show warnings when groups are imbalanced
    - Alert on anomalous metrics

# Data for scaled decisions

## A/B testing

- Culture of experimentation
  - Checklists
  - Test even when you know you'll ship
  - Have discipline:
    - State hypotheses ahead of time
    - Wait several weeks before shipping
    - Write down what you did and why, plus results
- Much more!



Thanks!