(Week-2)

fact Data - every event a user can do

huge - Petabyte per day at netflix

Fact Data Modelling

<Fact> can't change it

(postgres)

└ that happened or occurred

Ex. User logs in to an app

A transaction is made

You run a mile w/ fitbit - not atomic

can take to steps level - mile is agg level

- 10 - 100x of dimension data

FB - 2B Active user ~ 25 notifications per day

= ~ 50 B noti per day

- Fact need context for effective analysis - funnel of what happened after noti.
- Duplicates in facts - clicking multiple times

they bought somthing

<Dimensions> Slowly changing - Easier to Model

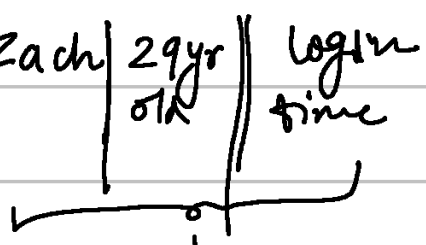
Fact Modelling

Normalized

user_id / fact data col~

- No Dups
- Small scale better
- Smaller than raw logs
- Parse out Json

Denormalized



Bring in dimensions into fact data but redundancy if

<Raw Logs> - owned by SWEs
Ugly schemas for online system
can have duplicates
Shorter retention -

<Fact>
Who → pushed as IDs (this user clicked the button)
Where → where in webpage they clicked.
How → used an iphone to make the click
What → generated, click, sent, Delivered.
When → event timestamp

<Fact Data> Nice col^n name → trusted Dataset.

2 Petabytes of data every new delta
Every network request Netflix recives every day -

Ex - Getting data of how Microservices are interacting w/ eachother.

(Apps)

Security issue if one app gets hacked then all other it talks to. - Network traffic

I.P. ↔ APP Mapping.

IP Address Data for all Netflix Microservices (small dataset > 5-6 GiB)

Worked for IPv4 But not for IPv6

Broadcast Join in Spark

Shuffle Join will not work.

figured out what Application it is after Join

Added a "Sidecar Proxy" in each microservice to log the APP they were coming from in each network request. Introduced Denormalisation as I.P. is the identifier. - Remove need for Join

Talked to so many people App owners to install this Sidecar proxy & new Apps. ~ 3000 App owners convos.

Logging in fact data → contract or shared schema while logging. Logging in Ruby but data in Scala so middle layer Thrift used at Airbnb.

Shared Schema → for specification + Language Agnostic of schema description for all teams

Potential options when working w/ high volume fact data.

1. Sampling → works for metric driven use cases cases where imprecision isn't an issue

like S3 cost will increase if we do this

Law of large numbers → normal distribution to know direction of data

not where you want to catch one specific thing.

2. Bucketting → Bucket Join to reduce shuffles ↑ faster

joined on imp dimension

done on who id - user id.

SMB - No Shuffles

3. Retention → cost to hold data
→ legal risk ; anonymize it.

Any fact table > 100TBs, retention < 14 days

Deduplication of Fact data

- Notification dataset dedup at Fb is 9hr/day in Hive

Do you really care if a user clicks on notification after 2 yrs.?

Pick right window for deduplication

A day? An hour? A week?

Look at distribution of duplicates.

Deduping options

1. Streaming - lower
2. Micro batch - hourly

↓

capture in small window

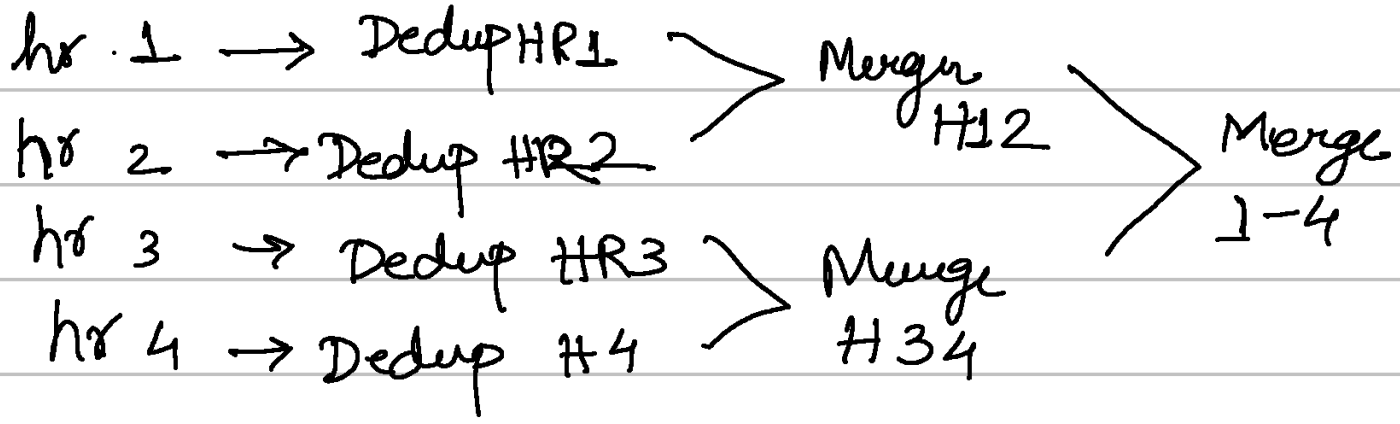
Entire day duplicates can be harder for streaming because it needs to hold onto such big window of memory.

Hourly Microbatch Dedup

50 Bill records/day

1. All data for an hour and remove all duplicates using group by — for all hours.

2. full outer Join b/w 0 & 1 to remove dups from 2 hours 0 & 1.



Batch - low latency - Scale ↑

→ Grain of the data - count(*) group by grain having count > 1.

filter to get rid of data.