Succeeding with Apache Druid®
and Clickstream Data

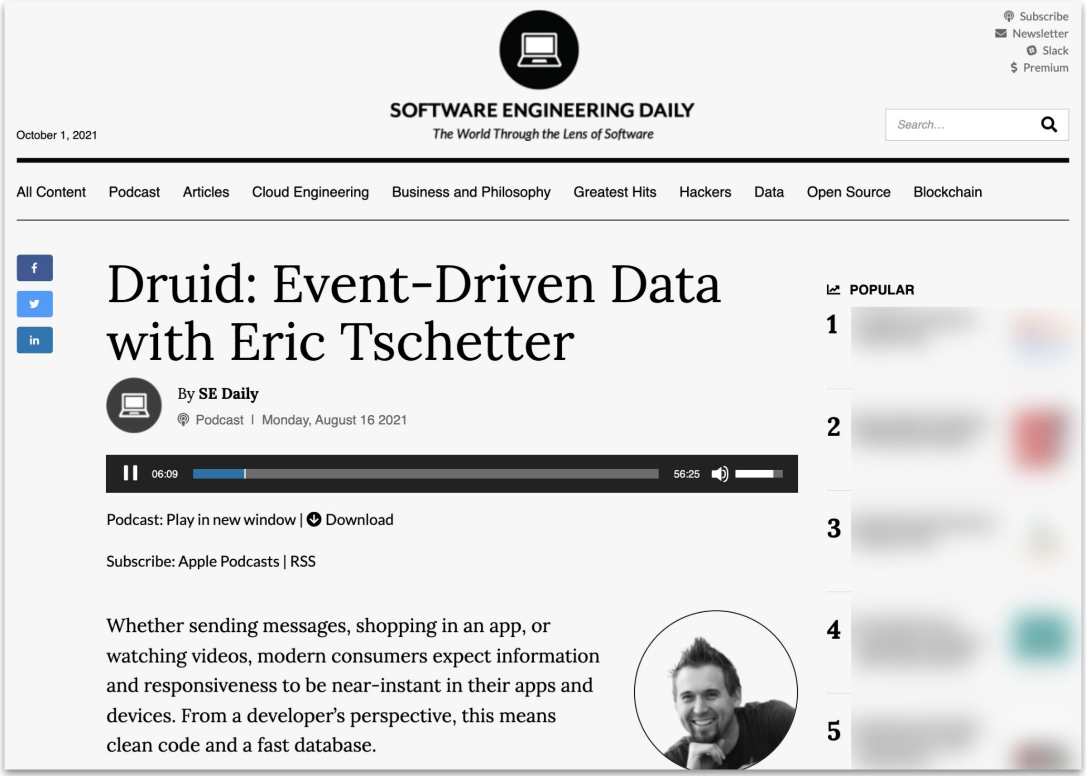
Peter Marshall

What is Apache Druid®? Who are Imply?

Apache Druid is a **high performance, real-time analytics** database … where **fast** ad-hoc analytics, **instant** data visibility, or supporting high **concurrency** is important … where an **interactive**, consistent user **experience** is desired.

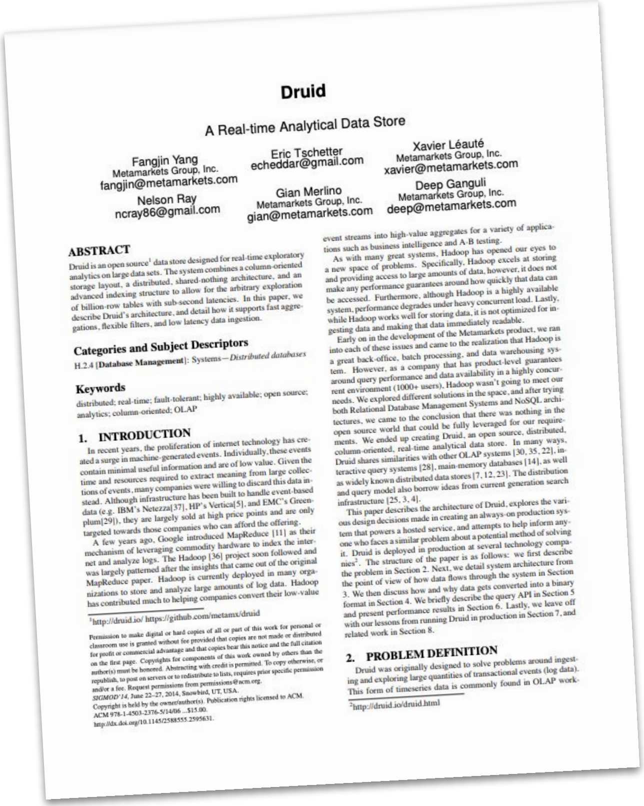
We will devote our energy to making it as easy as possible for people to **use Druid** and **build awesome data applications** on top of it.

Fangjin Yang, October 2015



Software Engineering Daily: an interview with Eric Tschetter

https://softwareengineeringdaily.com/2021/08/16/druid-event-driven-data-with-eric-tschetter/



Druid

A Real-time Analytical Data Store

Fangjin Yang
Metamarkets Group, Inc.
fangjin@metamarkets.com

Eric Tschetter
echeddar@gmail.com

Xavier Léauté
Metamarkets Group, Inc.
xavier@metamarkets.com

Nelson Ray
ncray86@gmail.com

Gian Merlino
Metamarkets Group, Inc.
gian@metamarkets.com

Deep Ganguli
Metamarkets Group, Inc.
deep@metamarkets.com

ABSTRACT

Druid is an open source[1] data store designed for real-time exploratory analytics on large data sets. The system combines a column-oriented storage layout, a distributed, shared-nothing architecture, and an advanced indexing structure to allow for the arbitrary exploration of billion-row tables with sub-second latencies. In this paper, we describe Druid's architecture, and detail how it supports fast aggregations, flexible filters, and low latency data ingestion.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems—*Distributed databases*

Keywords

distributed; real-time; fault-tolerant; highly available; open source; analytics; column-oriented; OLAP

1. INTRODUCTION

In recent years, the proliferation of internet technology has created a surge in machine-generated events. Individually, these events contain minimal useful information and are of low value. Given the time and resources required to extract meaning from large collections of events, many companies were willing to discard this data instead. Although infrastructure has been built to handle event-based data (e.g. IBM's Netezza[37], HP's Vertica[5], and EMC's Greenplum[29]), they are largely sold at high price points and are only targeted towards those companies who can afford the offering.

A few years ago, Google introduced MapReduce [11] as their mechanism of leveraging commodity hardware to index the internet and analyze logs. The Hadoop [36] project soon followed and was largely patterned after the insights that came out of the original MapReduce paper. Hadoop is currently deployed in many organizations to store and analyze large amounts of log data. Hadoop has contributed much to helping companies convert their low-value event streams into high-value aggregates for a variety of applications such as business intelligence and A-B testing.

As with many great systems, Hadoop has opened our eyes to a new space of problems. Specifically, Hadoop excels at storing and providing access to large amounts of data, however, it does not make any performance guarantees around how quickly that data can be accessed. Furthermore, although Hadoop is a highly available system, performance degrades under heavy concurrent load. Lastly, while Hadoop works well for storing data, it is not optimized for ingesting data and making that data immediately readable.

Early on in the development of the Metamarkets product, we ran into each of these issues and came to the realization that Hadoop is a great back-office, batch processing, and data warehousing system. However, as a company that has product-level guarantees around query performance and data availability in a highly concurrent environment (1000+ users), Hadoop wasn't going to meet our needs. We explored different solutions in the space, and after trying both Relational Database Management Systems and NoSQL architectures, we came to the conclusion that there was nothing in the open source world that could be fully leveraged for our requirements. We ended up creating Druid, an open source, distributed, column-oriented, real-time analytical data store. In many ways, Druid shares similarities with other OLAP systems [30, 35, 22], interactive query systems [28], main-memory databases [14], as well as widely known distributed data stores [7, 12, 23]. The distribution and query model also borrow ideas from current generation search infrastructure [25, 3, 4].

This paper describes the architecture of Druid, explores the various design decisions made in creating an always-on production system that powers a hosted service, and attempts to help inform anyone who faces a similar problem about a potential method of solving it. Druid is deployed in production at several technology companies[2]. The structure of the paper is as follows: we first describe the problem in Section 2. Next, we detail system architecture from the point of view of how data flows through the system in Section 3. We then discuss how and why data gets converted into a binary format in Section 4. We briefly describe the query API in Section 5 and present performance results in Section 6. Lastly, we leave off with our lessons from running Druid in production in Section 7, and related work in Section 8.

2. PROBLEM DEFINITION

Druid was originally designed to solve problems around ingesting and exploring large quantities of transactional events (log data). This form of timeseries data is commonly found in OLAP work-

[1] http://druid.io/ https://github.com/metamx/druid

[2] http://druid.io/druid.html

SIGMOD'14, June 22–27, 2014, Snowbird, UT, USA.

ACM 978-1-4503-2376-5/14/06 ...$15.00.
http://dx.doi.org/10.1145/2588555.2595631.

Druid: A real-time analytical data store

http://static.druid.io/docs/druid.pdf

NETFLIX

druid®

NETFLIX

ebay®

druid®

YAHOO!®

PayPal

cisco

airbnb
NETFLIX
CONDÉ NAST
dripstat
Alibaba.com
WIKIMEDIA FOUNDATION
ebay
druid®
YAHOO!
hulu
SK telecom
Time Warner Cable®
criteo
PayPal
cisco
triplelift

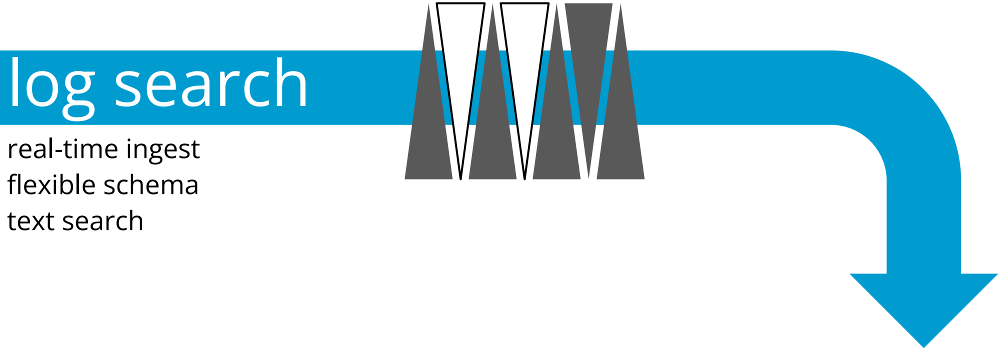
druid

Fully scalable

Batch and real-time data

Ad-hoc statistical queries

Low latency delivery



real-time ingest
flexible schema
text search

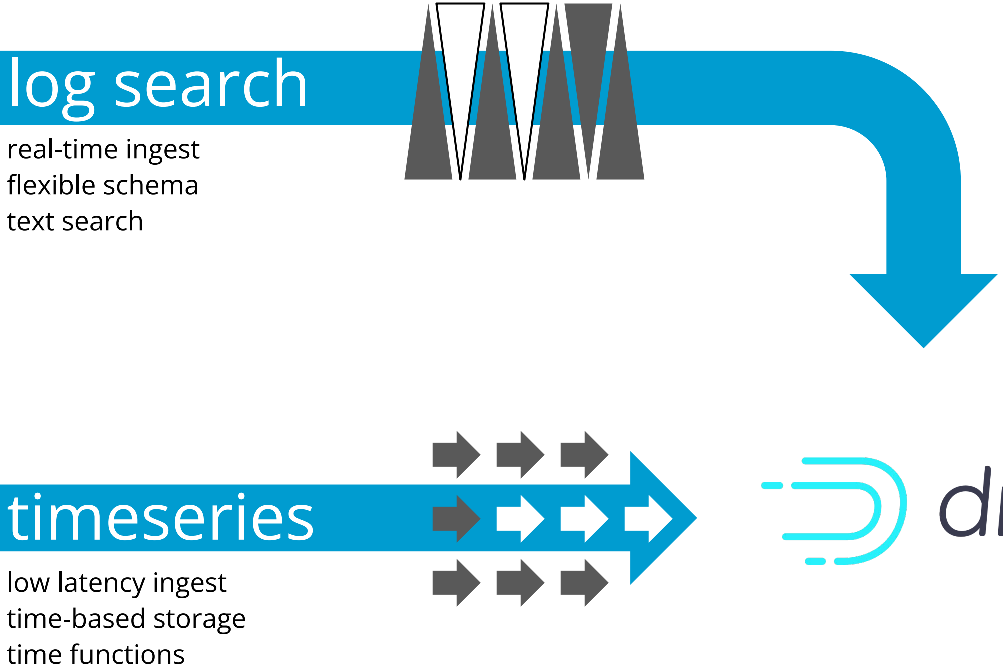
druid

Fully scalable

Batch and real-time data

Ad-hoc statistical queries

Low latency delivery



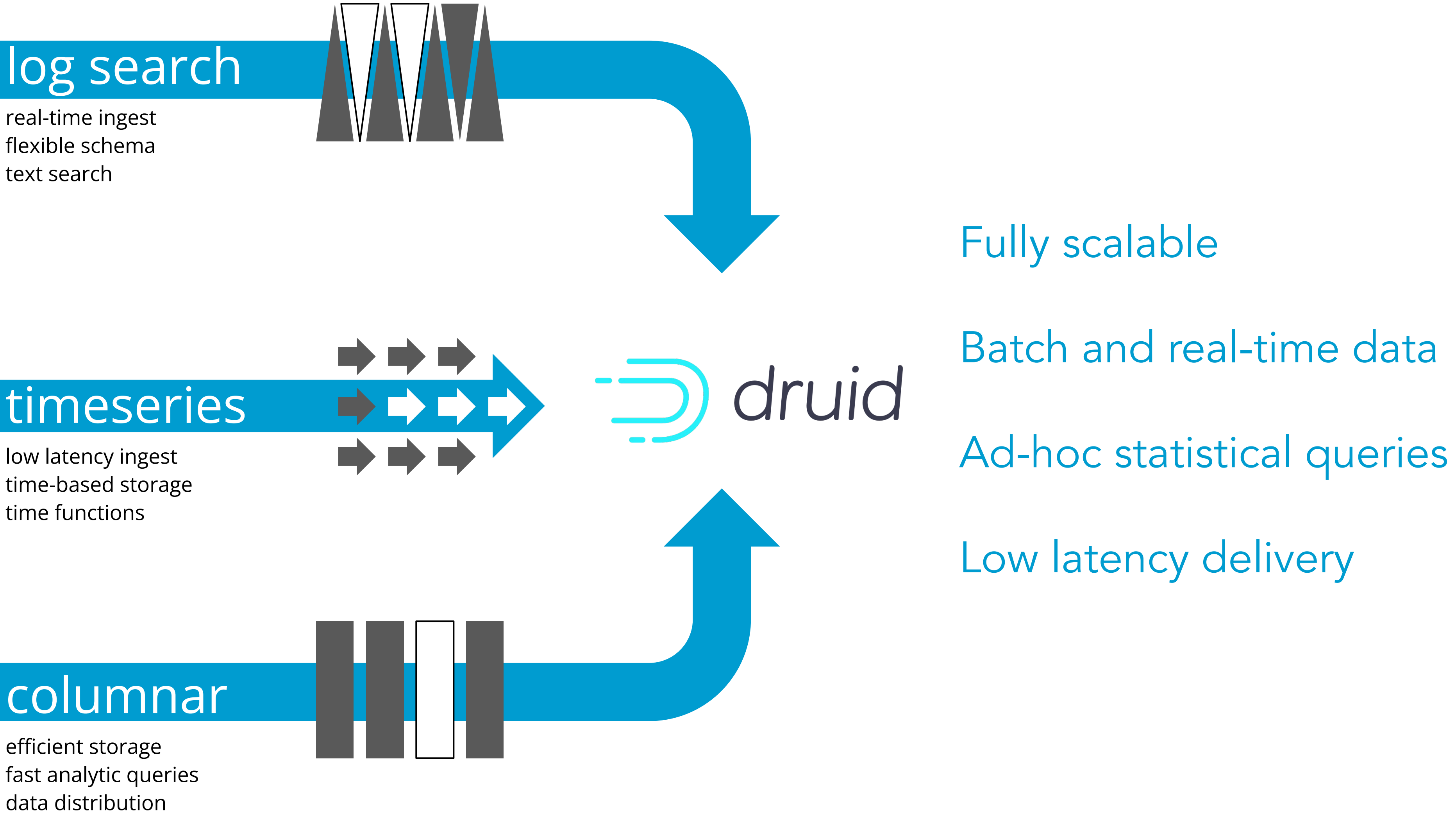
druid

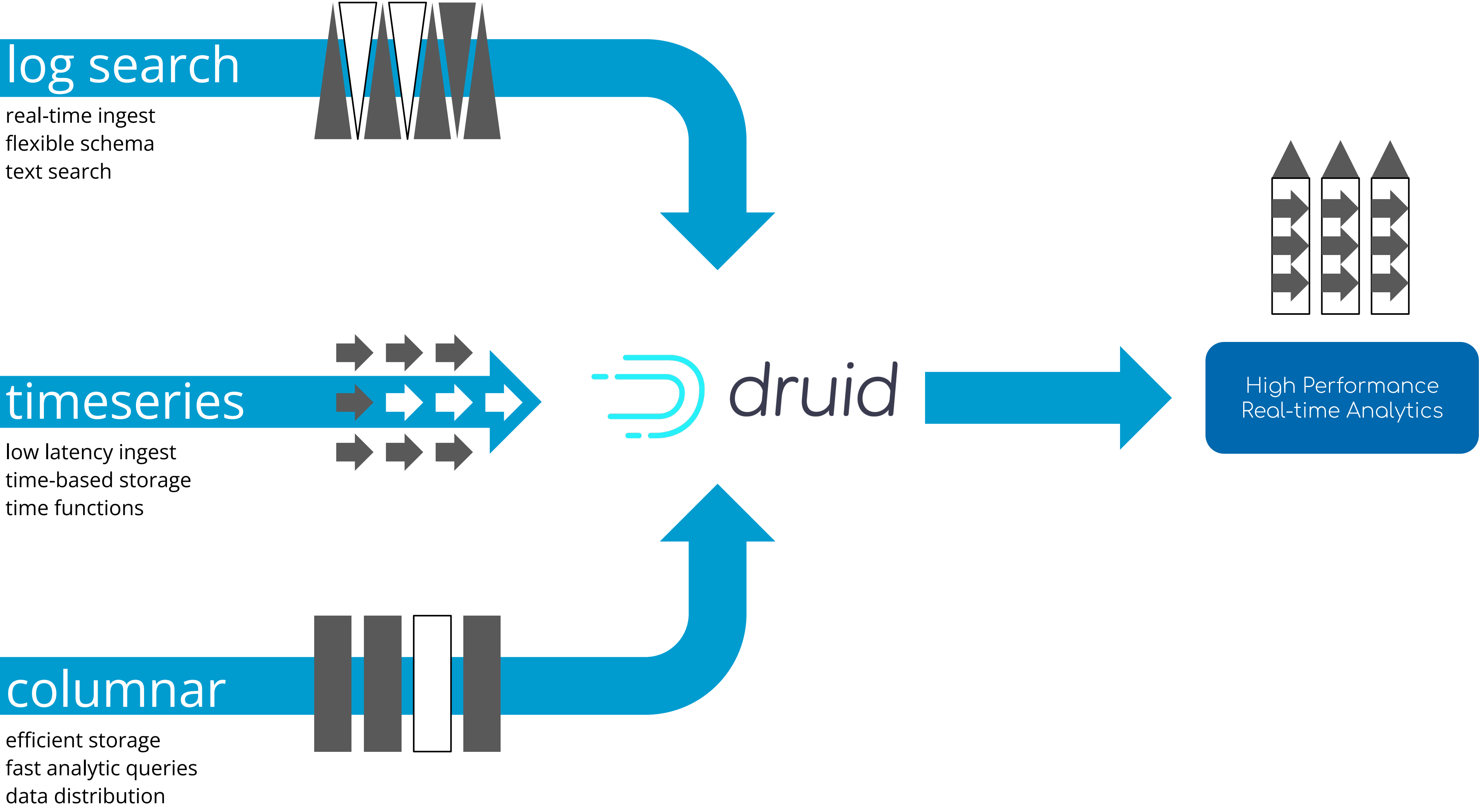
Fully scalable

Batch and real-time data

Ad-hoc statistical queries

Low latency delivery


log search
real-time ingest
flexible schema
text search
timeseries
low latency ingest
time-based storage
time functions
columnar
efficient storage
fast analytic queries
data distribution
druid
Fully scalable
Batch and real-time data
Ad-hoc statistical queries
Low latency delivery


log search
real-time ingest
flexible schema
text search
timeseries
low latency ingest
time-based storage
time functions
columnar
efficient storage
fast analytic queries
data distribution
druid
High Performance
Real-time Analytics

Why Clickstream Analytics?

“It puts us closer to our users
and if you know what your users want,
you’re better able to serve them.”

BBC | Sign in | Home | News | Sport | Weather | iPlayer | Sounds | CBBC | CBeebies | More | Search

Welcome to the BBC

LIVE Lifting curbs on 21 June risks U-turn - scientist

UK

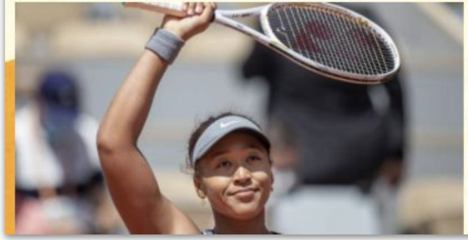
House prices soar 10.9% in 'race for space'

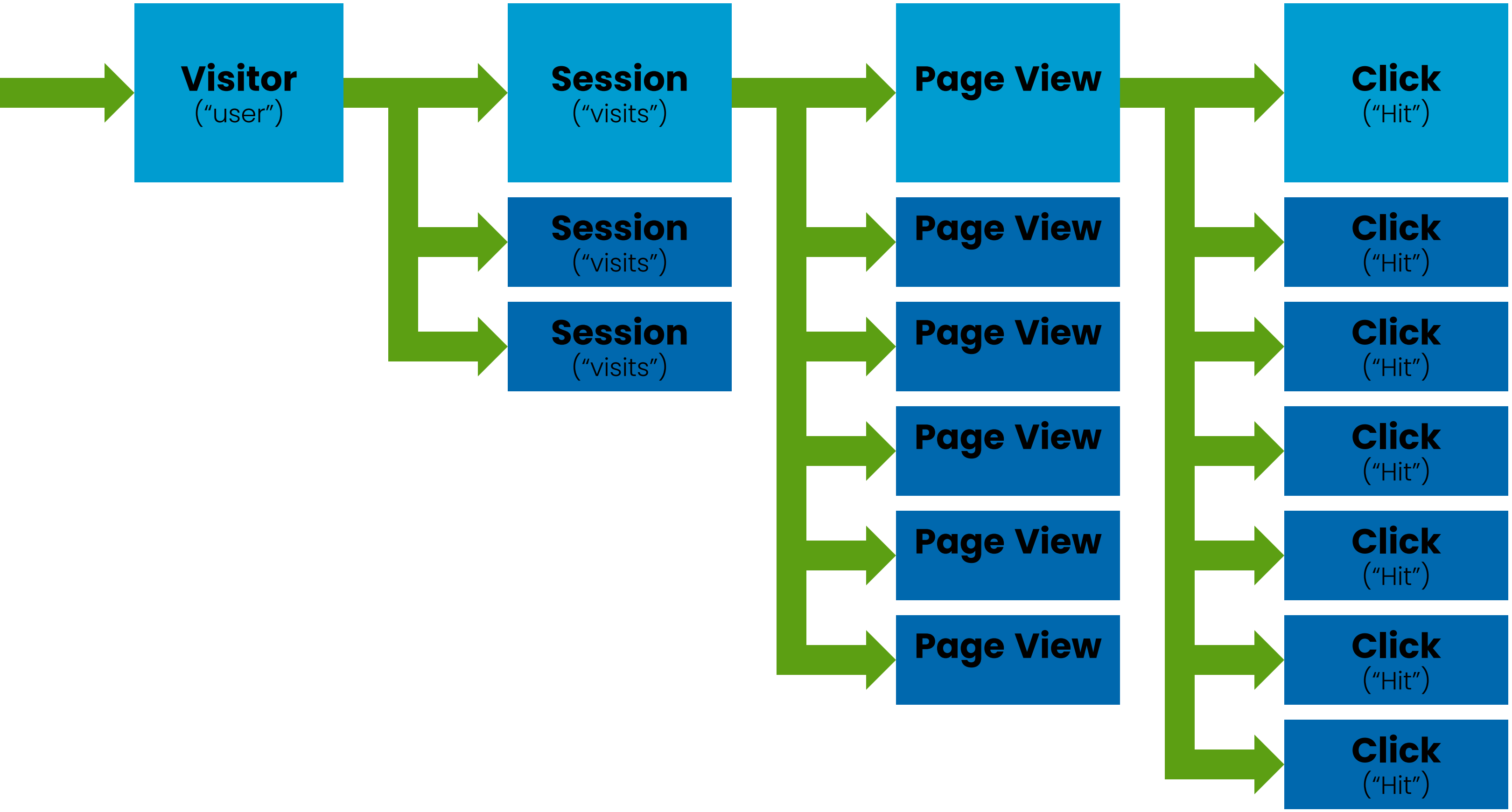
Business

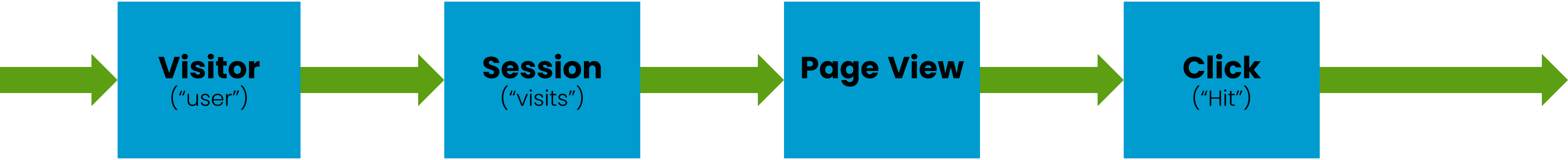
Greenwood pulls out of England squad

Football

More top stories




Visitor
("user")
Session
("visits")
Session
("visits")
Session
("visits")
Page View
Page View
Page View
Page View
Page View
Page View
Click
("Hit")
Click
("Hit")
Click
("Hit")
Click
("Hit")
Click
("Hit")
Click
("Hit")
Click
("Hit")



Drive **loyalty** with promotions

Change **advertising** and alter **promotion** strategies

Test tactics to reduce **churn** of loyal visitors

Personalise **experiences** based on your "usual" purchasing pattern (graze or hunt?)

Get rid of site content and **navigation** that doesn't produce results

Create "first visit, first buy" **behaviour** against competitors

Drive conversion right from the **entry point**

Change what **products** they're showing to you by understanding purchase causation / correlation

Clickstream Pipelines

Production	Sources	Processing	Storage	Analysis	Presentation

Javascript Code

(pixels, Divolte…)

Embedded Component

Log Files

Packet Sniffing

Click
("Hit")

Production	Sources	Processing	Storage	Analysis	Presentation
Specialised collectors Applications & APIs Machine & Human Data Environmental Sensors Systems of Record	Buses & Queues Bulk Repositories Distributed File Systems	Consolidation & Enrichment Transformation & Stripping Verification & Validation Filtering & Sorting		Feature & Structure Discovery Segmentation & Classification Recommendation Prediction & Anomaly Detection Statistical Calculations	Real-time Analytics BI Reporting & Dashboards Search & Filtering UIs Applications & APIs

Production	Sources	Processing	Storage	Analysis	Presentation

Javascript Code

(pixels, Divolte…)

Embedded Component

Log Files

Packet Sniffing

Click
("Hit")

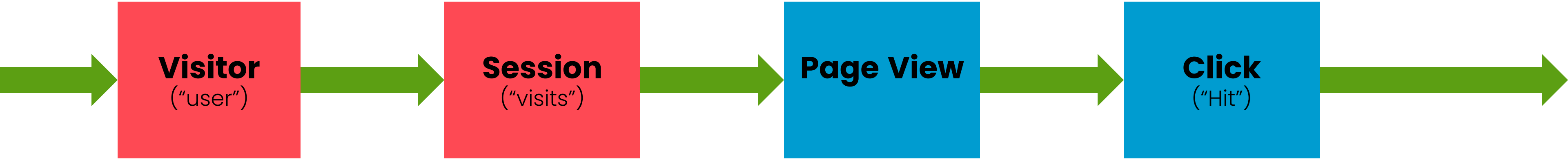
Specialised collectors
Applications & APIs
Machine & Human Data
Environmental Sensors
Systems of Record

Buses & Queues
Bulk Repositories
Distributed File Systems

Consolidation & Enrichment
Transformation & Stripping
Verification & Validation
Filtering & Sorting

Feature & Structure Discovery
Segmentation & Classification
Recommendation
Prediction & Anomaly Detection
Statistical Calculations

Real-time Analytics
BI Reporting & Dashboards
Search & Filtering UIs
Applications & APIs



Clickstream is stateless
We don't know when a session ends

Clickstream is anonymous
We don't know who the visitor is

Production	Sources	Processing	Storage	Analysis	Presentation

Javascript Code

(pixels, Divolte…)

Embedded Component

Log Files

Packet Sniffing

APACHE kafka

PULSAR

APACHE Spark

Visitor
("user")

Click
("Hit")

Session
("visits")

Specialised collectors
Applications & APIs
Machine & Human Data
Environmental Sensors
Systems of Record

Buses & Queues
Bulk Repositories
Distributed File Systems

Consolidation & Enrichment
Transformation & Stripping
Verification & Validation
Filtering & Sorting

Feature & Structure Discovery
Segmentation & Classification
Recommendation
Prediction & Anomaly Detection
Statistical Calculations

Real-time Analytics
BI Reporting & Dashboards
Search & Filtering UIs
Applications & APIs

Production	Sources	Processing	Storage	Analysis	Presentation

Production

Javascript Code

(pixels, Divolte…)

Embedded Component

Log Files

Packet Sniffing

Click
("Hit")

Sources

APACHE kafka

PULSAR

hadoop

Processing

APACHE Spark

Visitor
("user")

Session
("visits")

Specialised collectors
Applications & APIs
Machine & Human Data
Environmental Sensors
Systems of Record

Buses & Queues
Bulk Repositories
Distributed File Systems

Consolidation & Enrichment
Transformation & Stripping
Verification & Validation
Filtering & Sorting

Feature & Structure Discovery
Segmentation & Classification
Recommendation
Prediction & Anomaly Detection
Statistical Calculations

Real-time Analytics
BI Reporting & Dashboards
Search & Filtering UIs
Applications & APIs

What problems do they face?

- Capturing data is hard
  - Data aggregation
  - Data scale

What problems do they face?

- Capturing data is hard
  - Data aggregation
  - Data scale
- The volume is scary
  - Filtering for the right stuff
  - Doing statistics ad-hoc
  - Solving the COUNT DISTINCT problem

Production	Sources	Processing	Storage	Analysis	Presentation
Javascript Code (pixels, Divolte…) Embedded Component Log Files Packet Sniffing	Apache Kafka PULSAR hadoop	Apache Spark	druid®		React ANGULAR Metabase tableau Superset looker R python Scala jupyter
Specialised collectors Applications & APIs Machine & Human Data Environmental Sensors Systems of Record	Buses & Queues Bulk Repositories Distributed File Systems	Consolidation & Enrichment Transformation & Stripping Verification & Validation Filtering & Sorting	Feature & Structure Discovery Segmentation & Classification Recommendation Prediction & Anomaly Detection Statistical Calculations		Real-time Analytics BI Reporting & Dashboards Search & Filtering UIs Applications & APIs

Click
(“Hit”)

Visitor
(“user”)

Session
(“visits”)

What analytics are we talking about?

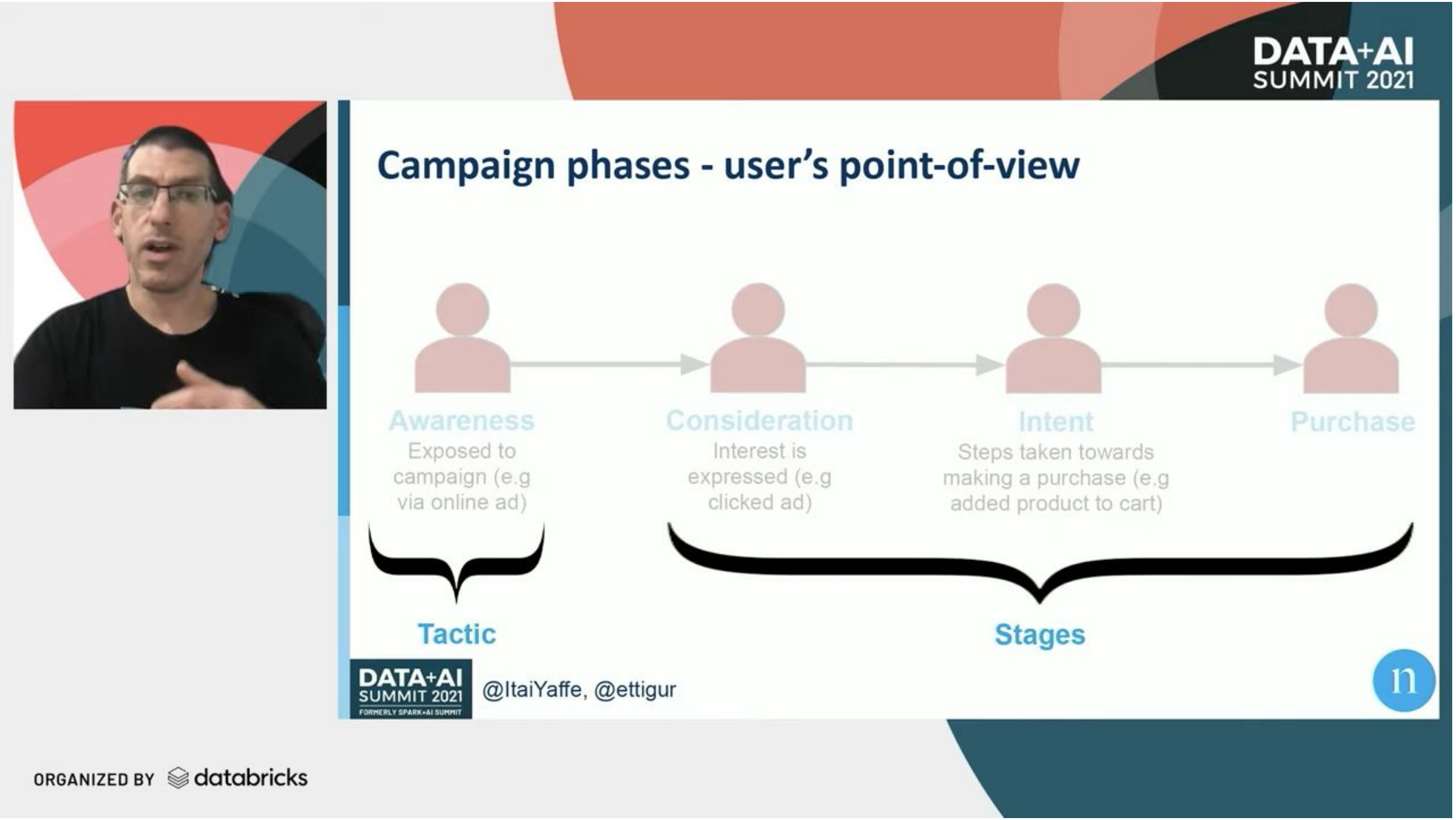
- Web Analytics
- Mobile App Analytics
- Advertising
- Streaming Video
- Process Mining

Critical Techniques

Druid functionality

- **Sub-partitioning:** enhance pruning of data that's picked up (multitenant, market segmentation, end-user segmentation)
- **Enrichment:** ingestion time and upstream (e.g. visitor demographics)
- **Rollup:** match end user display requirements – the pixels on the screen!
- **Approximation:** Use HyperLogLog and Thetasketches for DISTINCT COUNT and for set analysis (funnels)
- **Streaming:** get data in FAST!
- **Compaction:** Dealing with late arriving and out of order data
- **Changeable Schemas:** Adapting to changes in upstream data
- **Expressions:** Ingestion-time or upstream calculation (e.g. RegEx)
- **Hyperlean tables:** Filtering ahead of time at ingestion or upstream

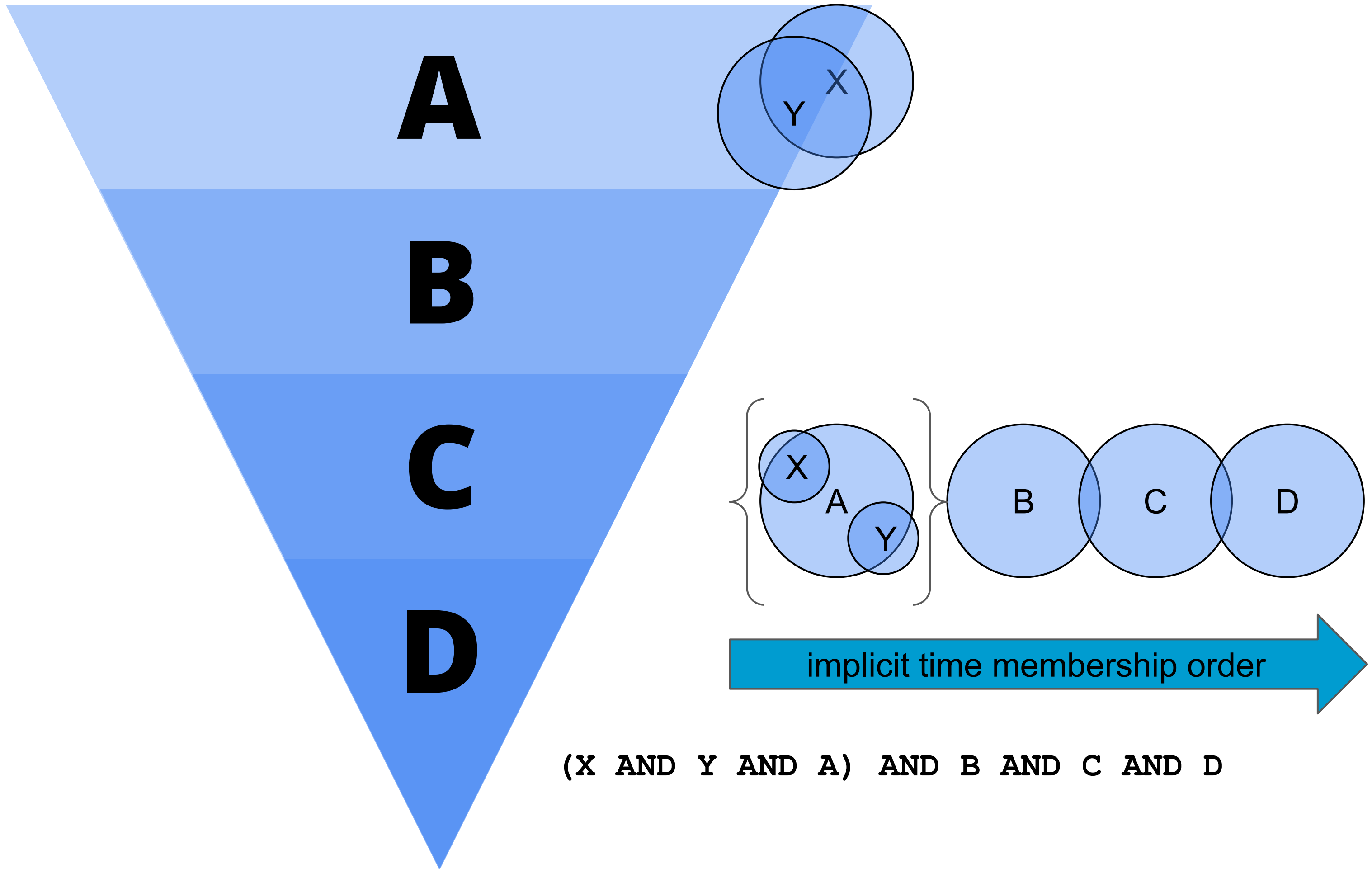
Critical Techniques: Funnel Analysis



Funnel Analysis with Apache Spark and Druid

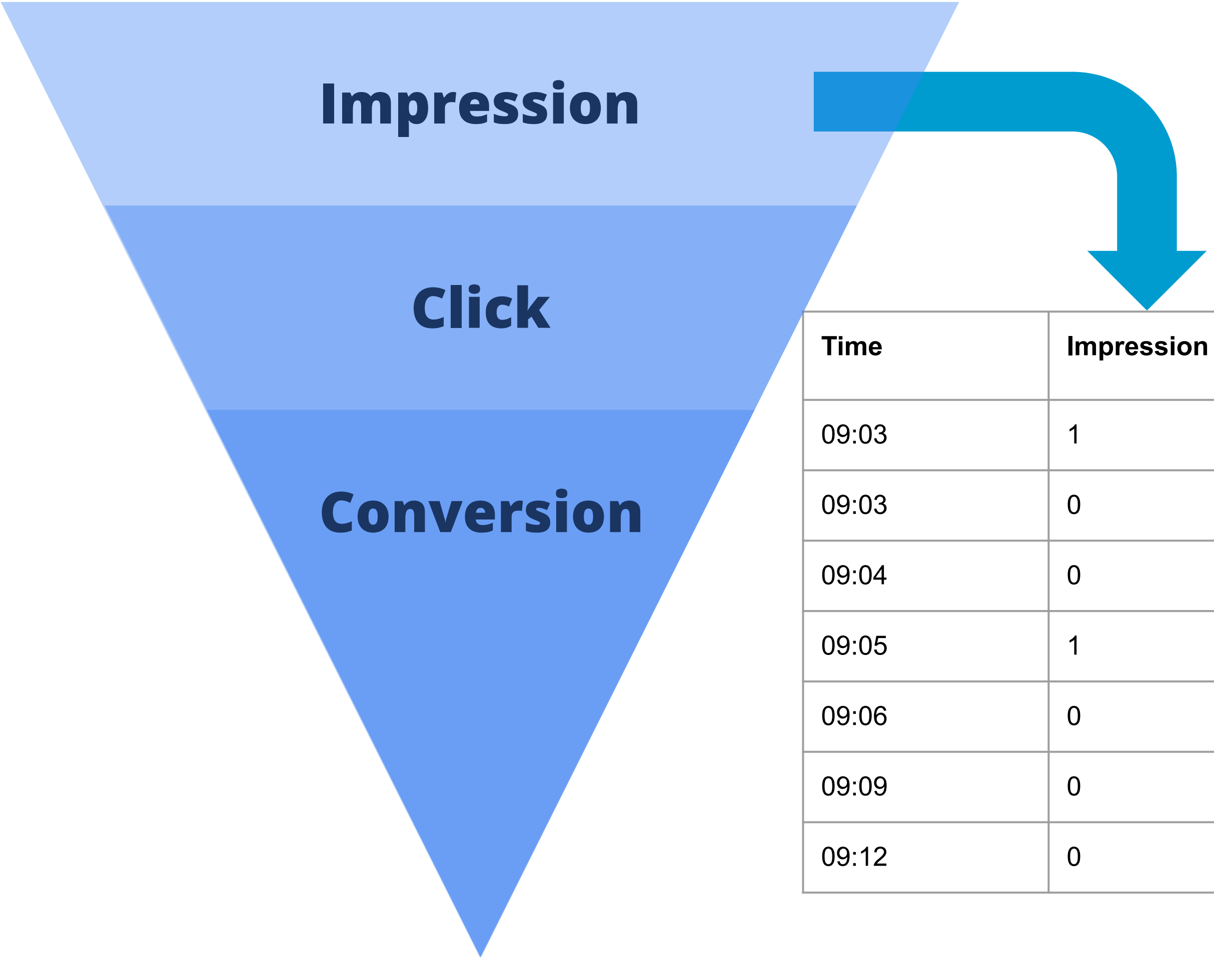
Funnel Patterns

- **Approximation with Set Analysis**
- Enriched Click data
- Enriched Session data

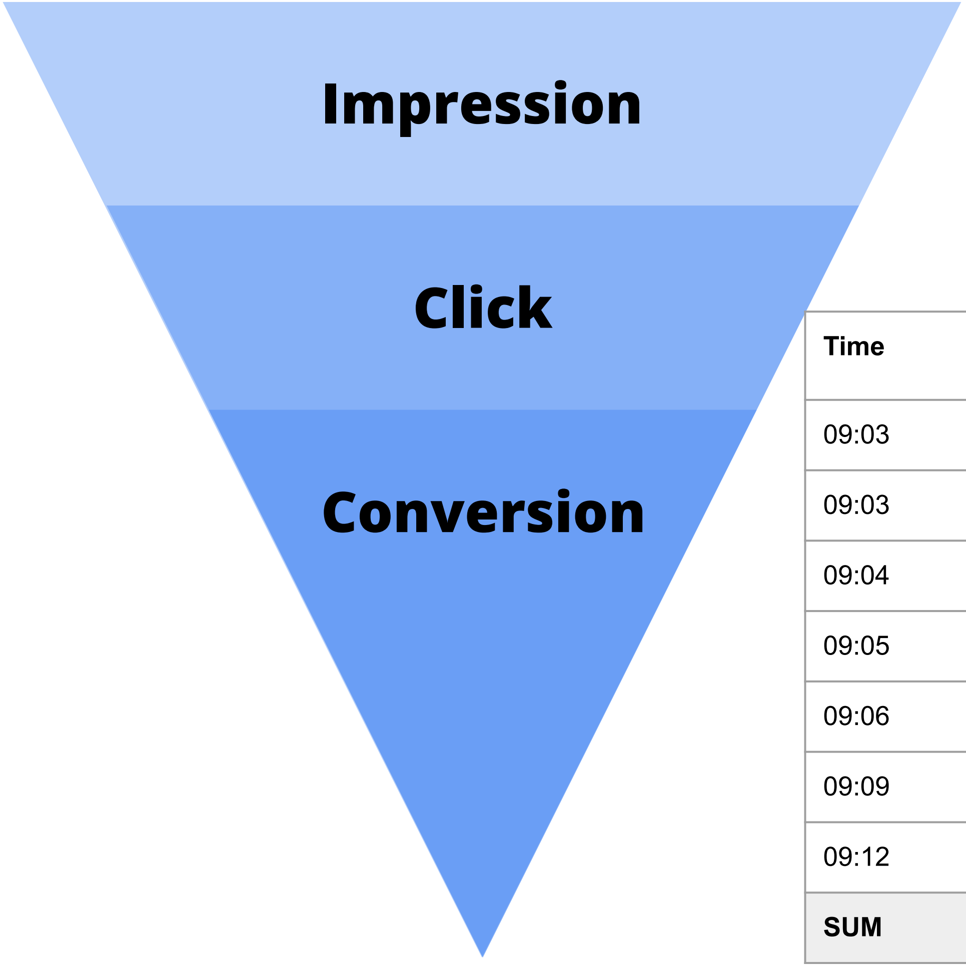

A
B
C
D
X
Y
X
A
Y
B
C
D
implicit time membership order
(X AND Y AND A) AND B AND C AND D

Funnel Patterns

- Approximation with Set Analysis
- **Enriched Click data**
- Enriched Session data



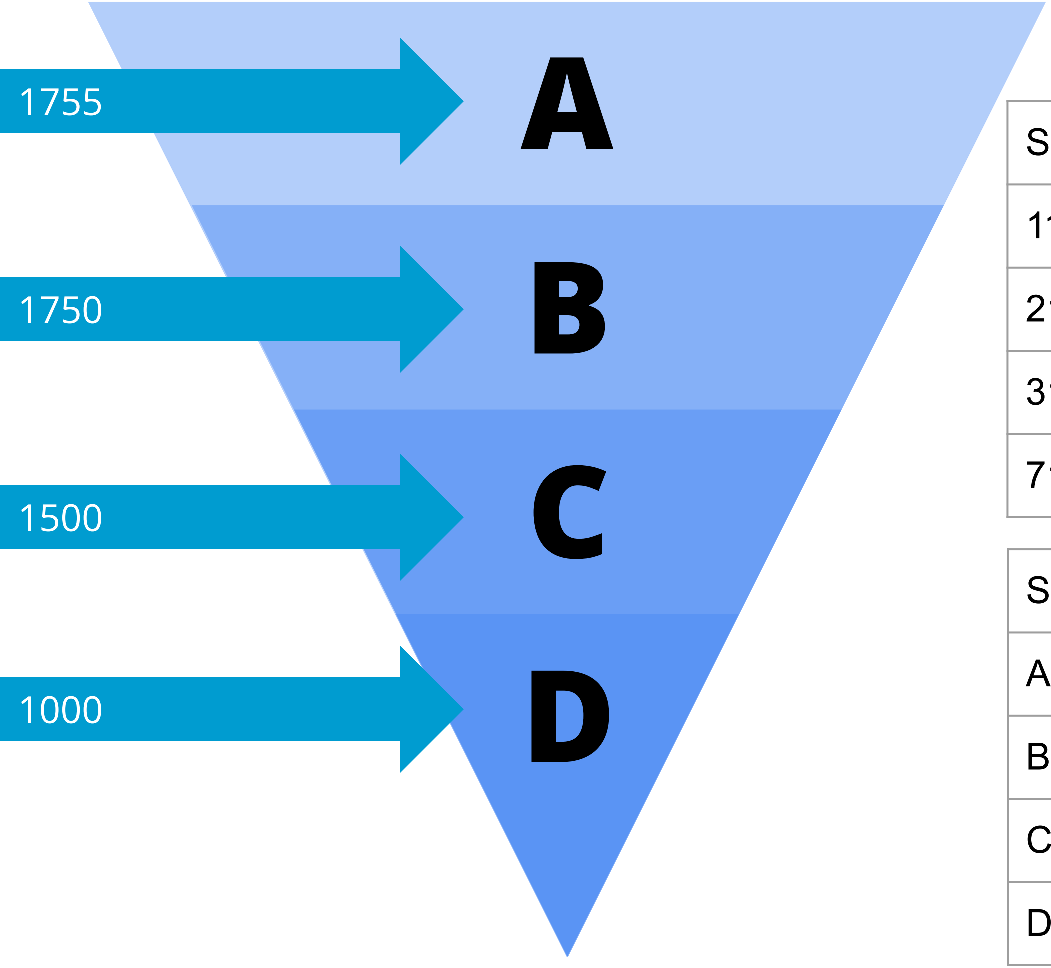
Time	Impression	Click	Conversion
09:03	1	0	0
09:03	0	0	1
09:04	0	0	0
09:05	1	0	0
09:06	0	1	0
09:09	0	1	1
09:12	0	0	1



Time	Impression	Click	Conversion
09:03	1	0	0
09:03	0	0	1
09:04	0	0	0
09:05	1	0	0
09:06	0	1	0
09:09	0	1	1
09:12	0	0	1
SUM	**2**	**2**	**3**

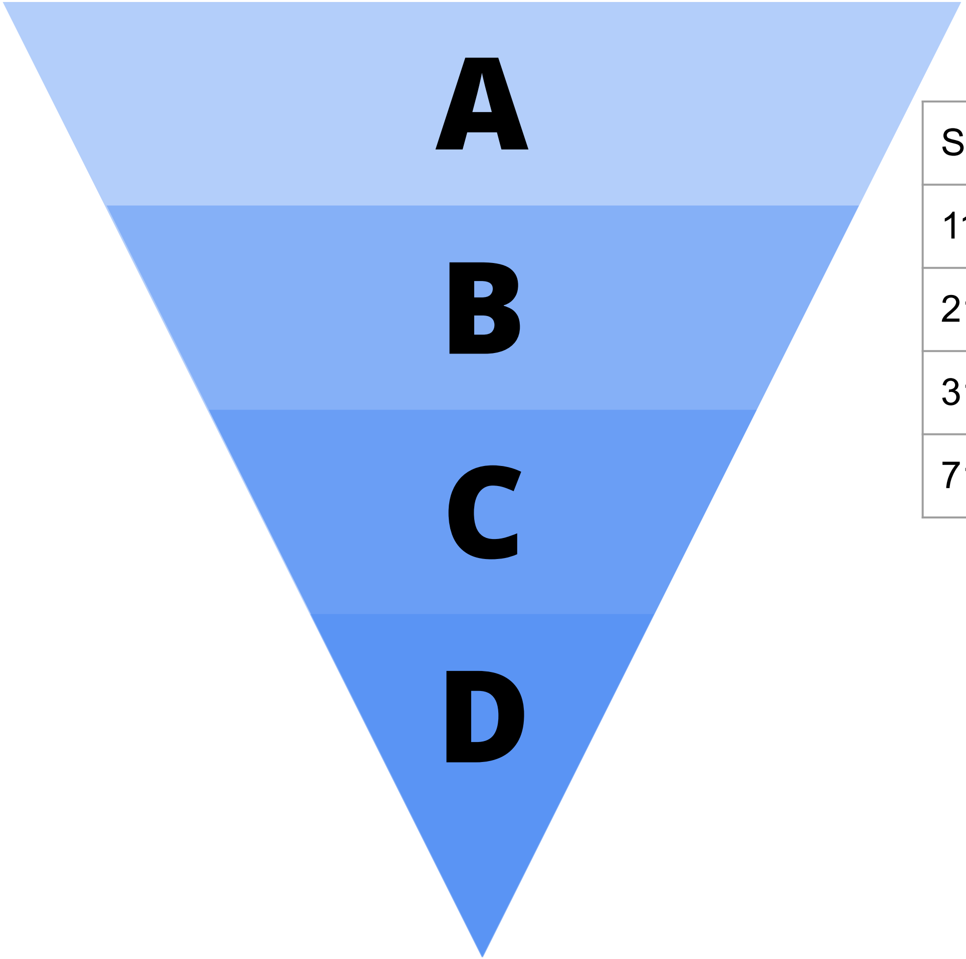
Funnel Patterns

- Approximation with Set Analysis
- Enriched Click data
- **Enriched Session data**



Session	Time	Last Stage
1142	09:04	A
2131	10:03	A
3112	11:43	B
7126	12:51	D

Stage	COUNT
A	5
B	250
C	500
D	1000



Session	Time	F1	F2	F3
1142	09:04	A	Z4	pl
2131	10:03	A	Z6	pl
3112	11:43	B	Z21	cart
7126	12:51	D	Z7	ad

Critical Techniques: Sessionisation

Sessions are long-lasting
Sessions are vague things
Some values *cannot* be known at 0s

Session Analytics

Difficulty
How much **effort** was required to move along?

Progress
What was the **schedule** from A to B?

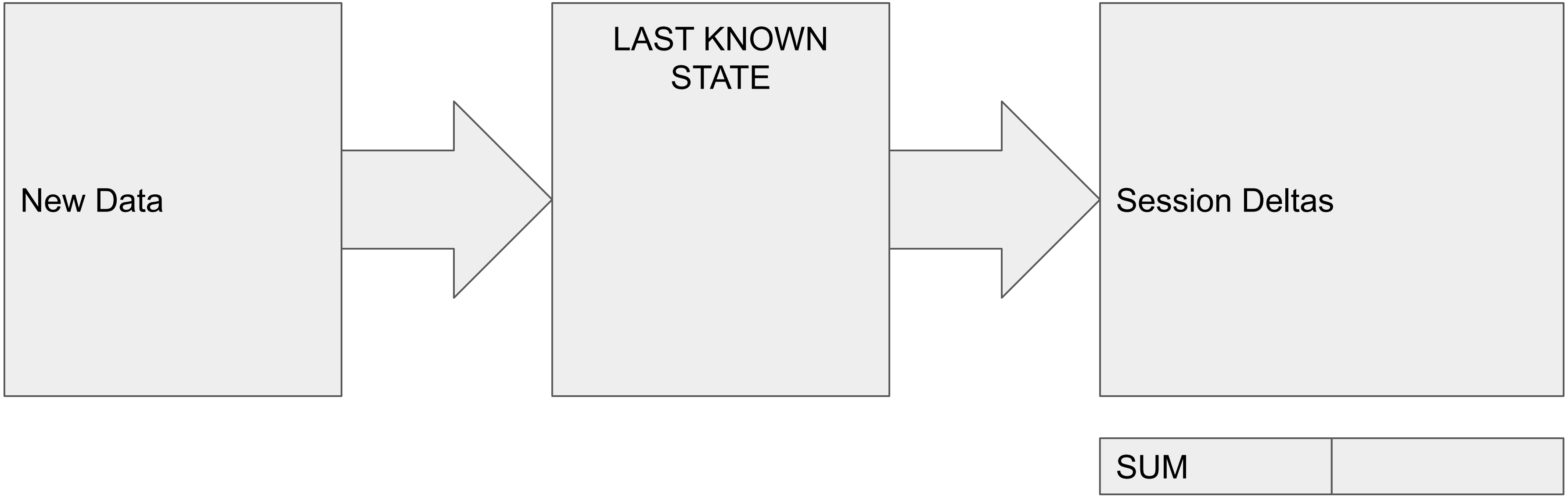
Goals
Did they attain any **achievements or rewards**?

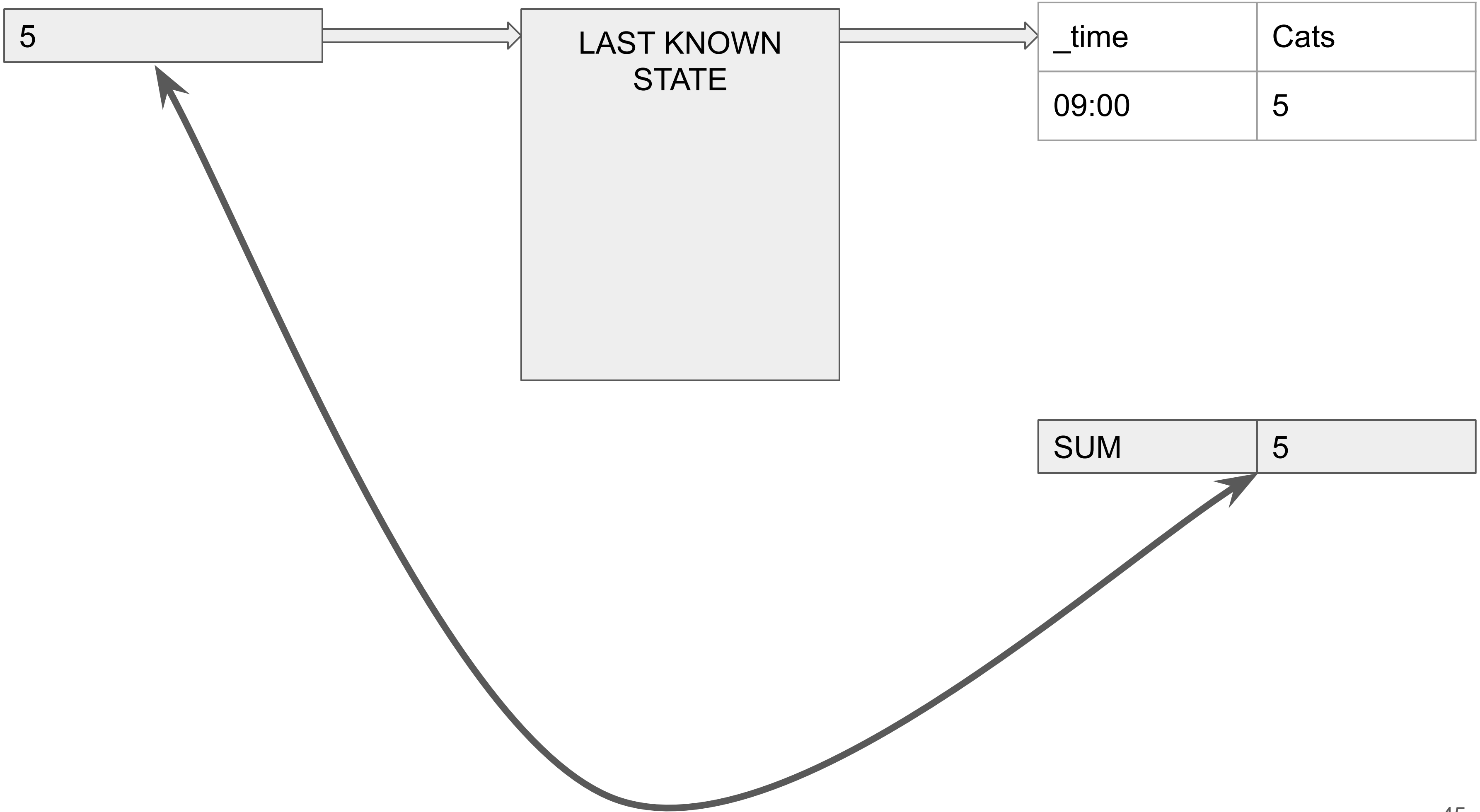
Cause
What kinds of things force a **change** in state?

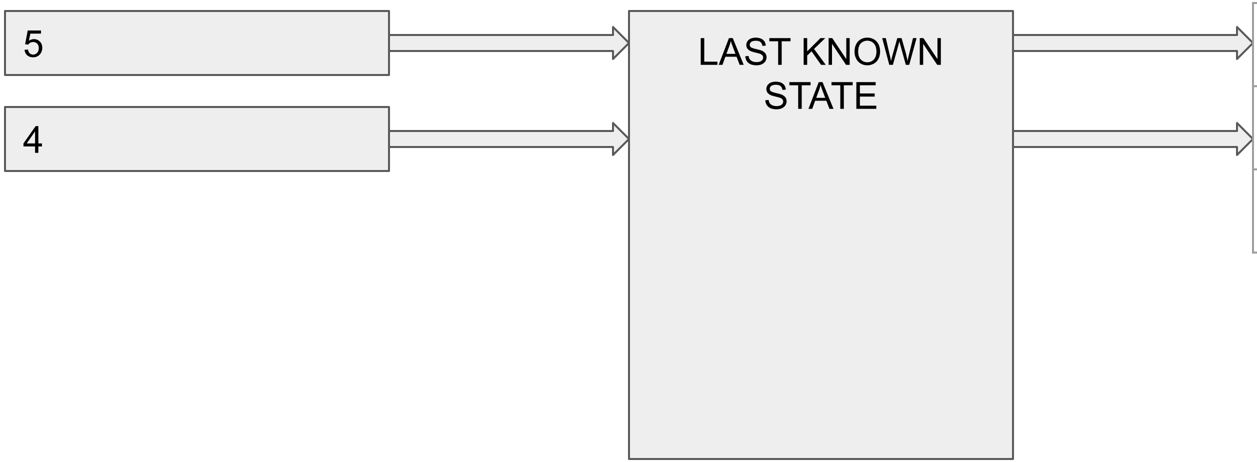
Purpose
What external factors **influence** things towards their eventual state?

Plan
What **route** did the actor take to getting finished?

Acts
What **actions** did the actor take?

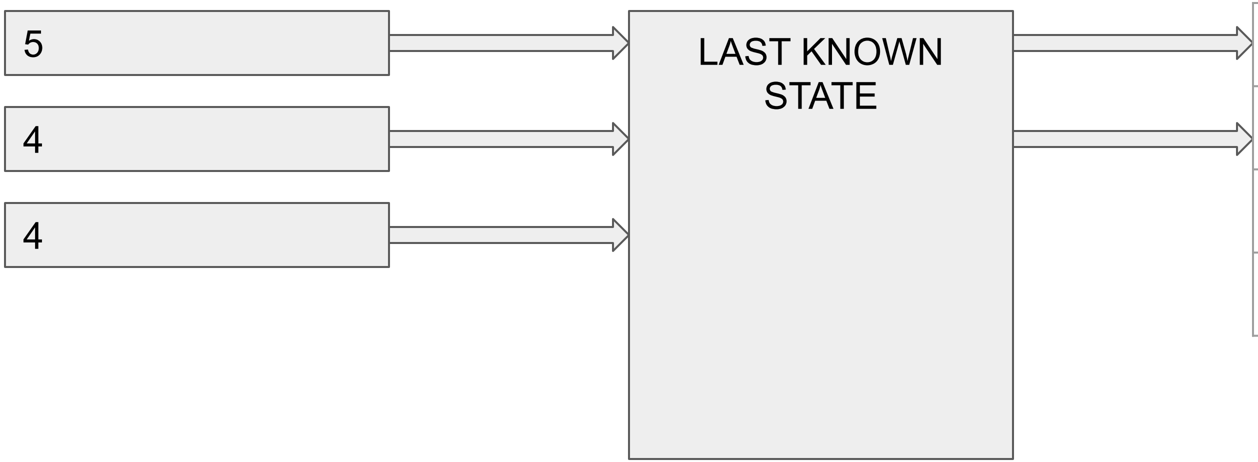

New Data
LAST KNOWN STATE
Session Deltas
SUM


5
LAST KNOWN STATE
_time
Cats
09:00
5
SUM
5



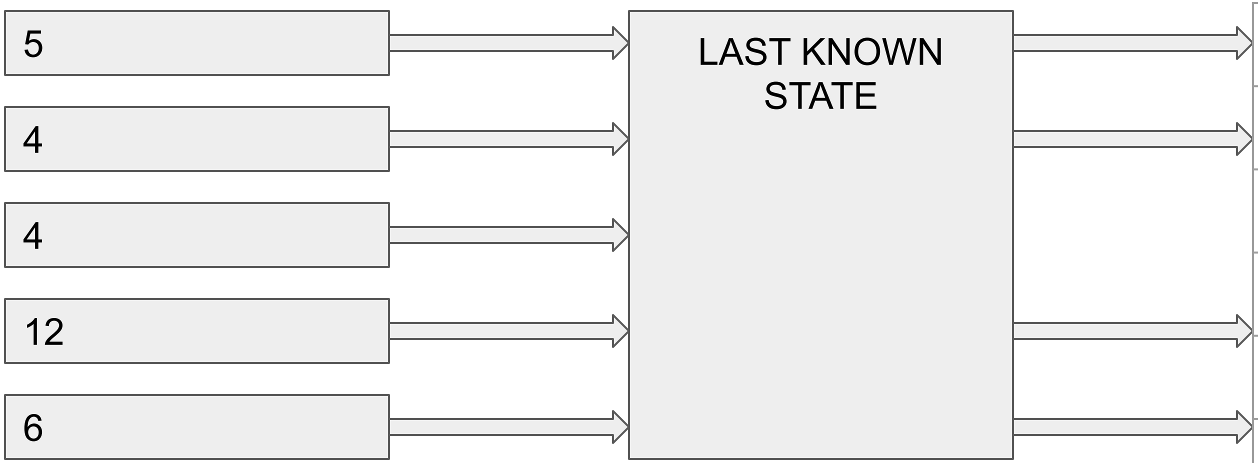
_time	Cats
09:00	5
09:13	-1

SUM	4



_time	Cats
09:00	5
09:13	-1
09:34	0

SUM	4



_time	Cats
09:00	5
09:13	-1
09:34	0
09:54	8
10:02	-6
SUM	6

- Ingestion scalability
- On-demand aggregation
- Filtering efficiency
- Time-based comparison
- Approximation

Get it to the desk!

Thank You!