

#TC18

Real-Time Insights for IoT and Applications

Max Zuckerman

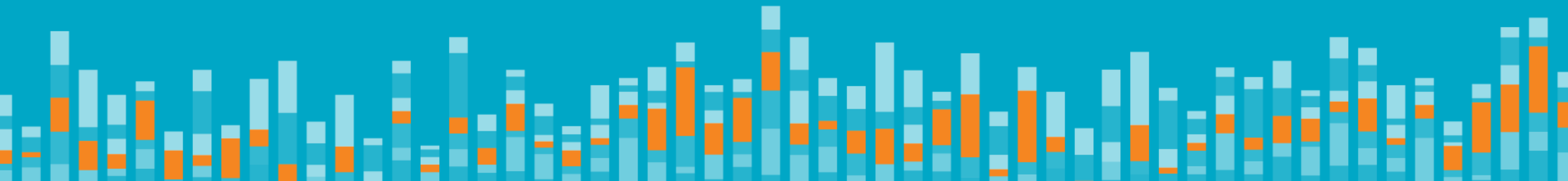
Sr. Director, Solutions Sales

Alooma

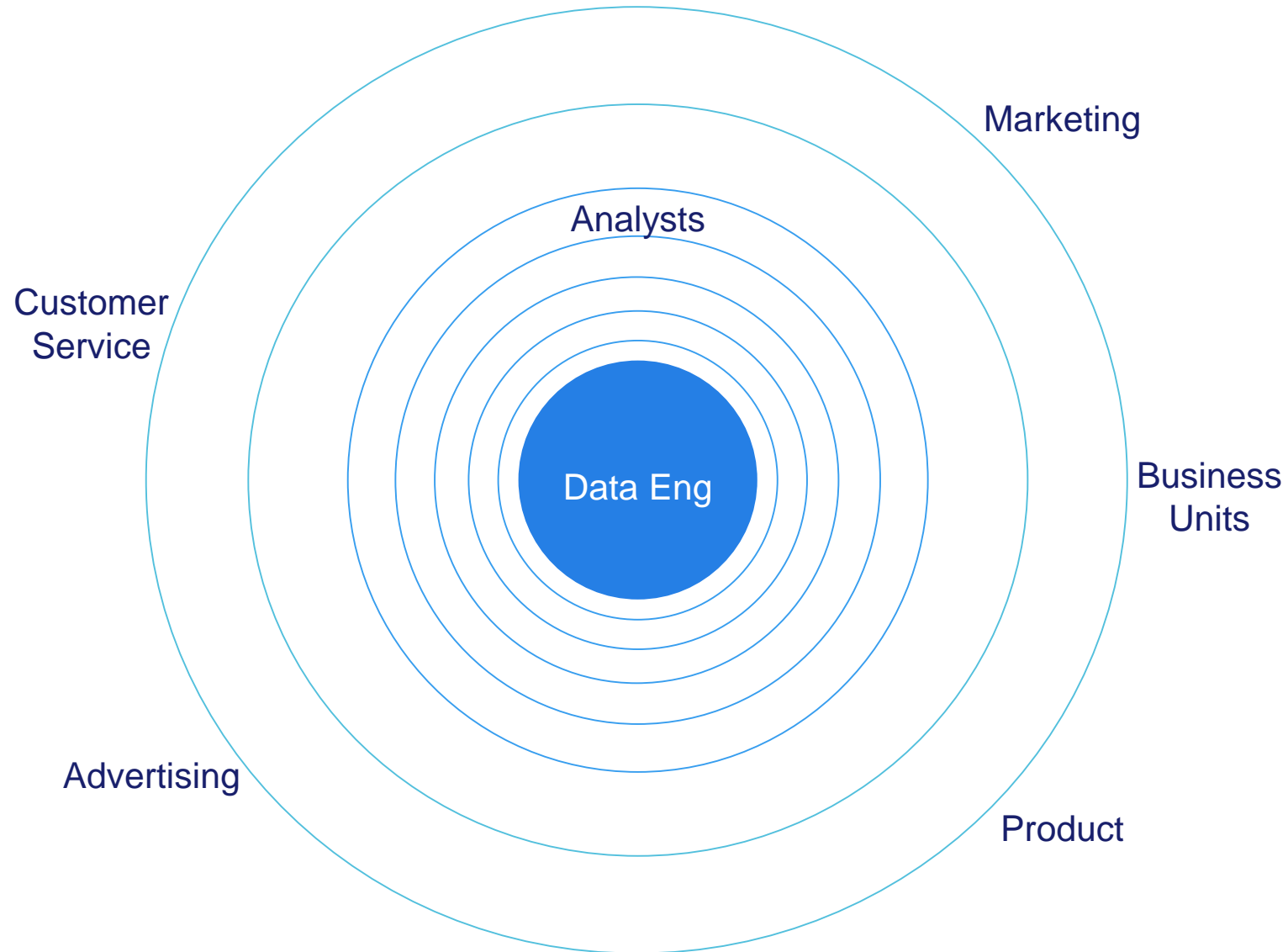


First of all, let's talk cloud.

But I already use the cloud...

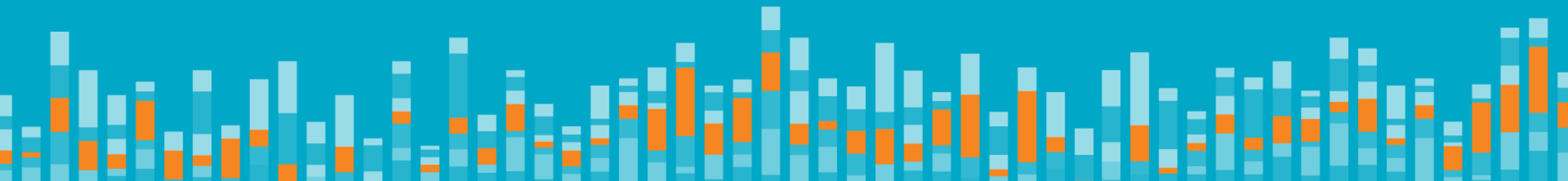


Expand Who Can Use the Data

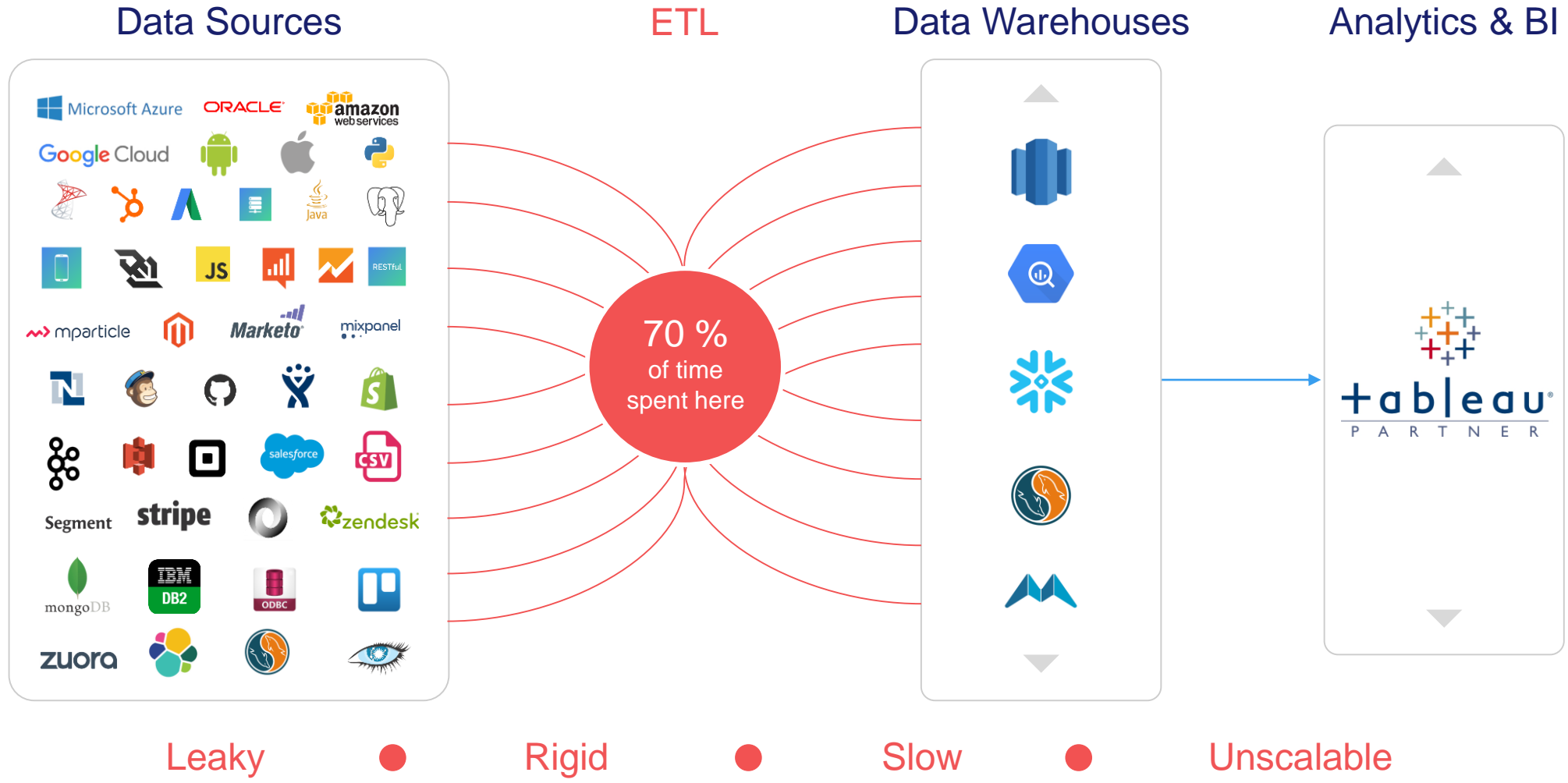


OK, so how do I get my data there?

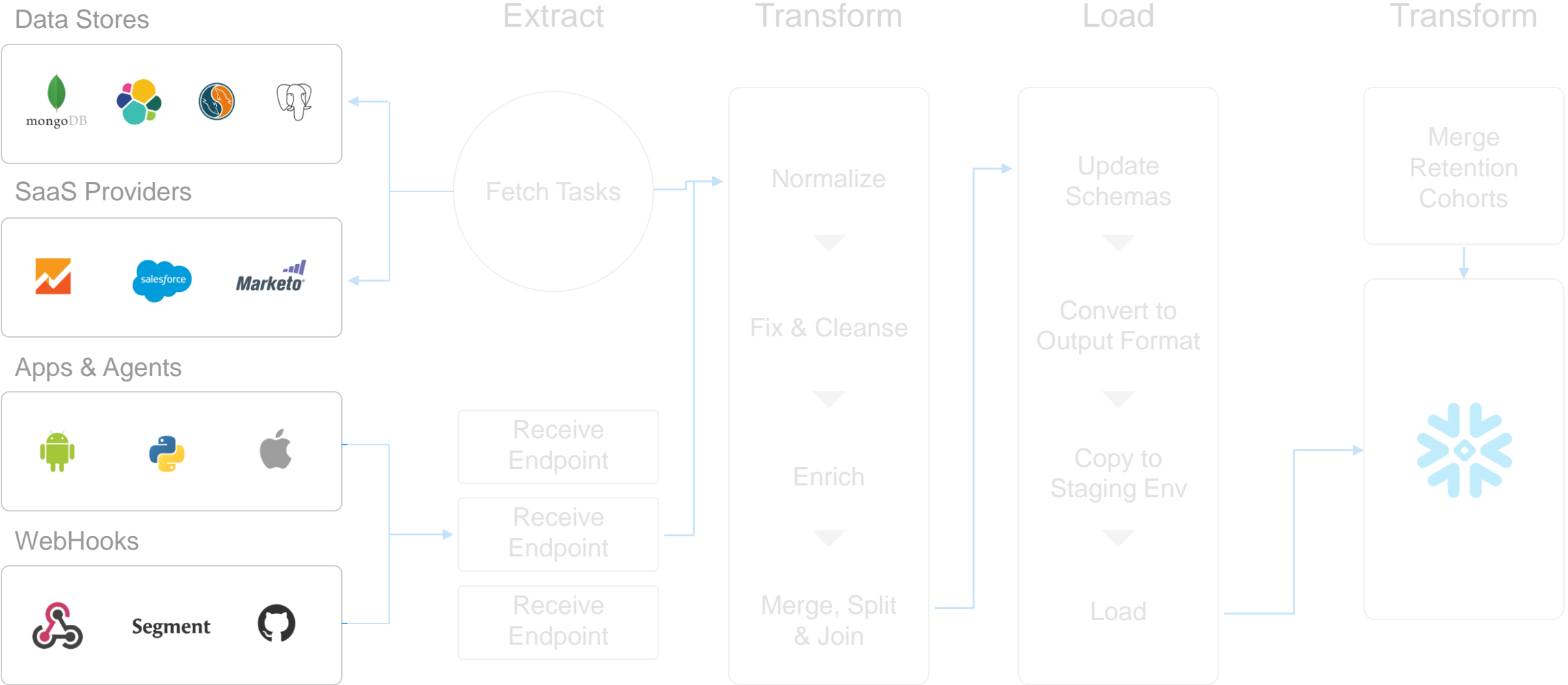
I'll just write a simple script...



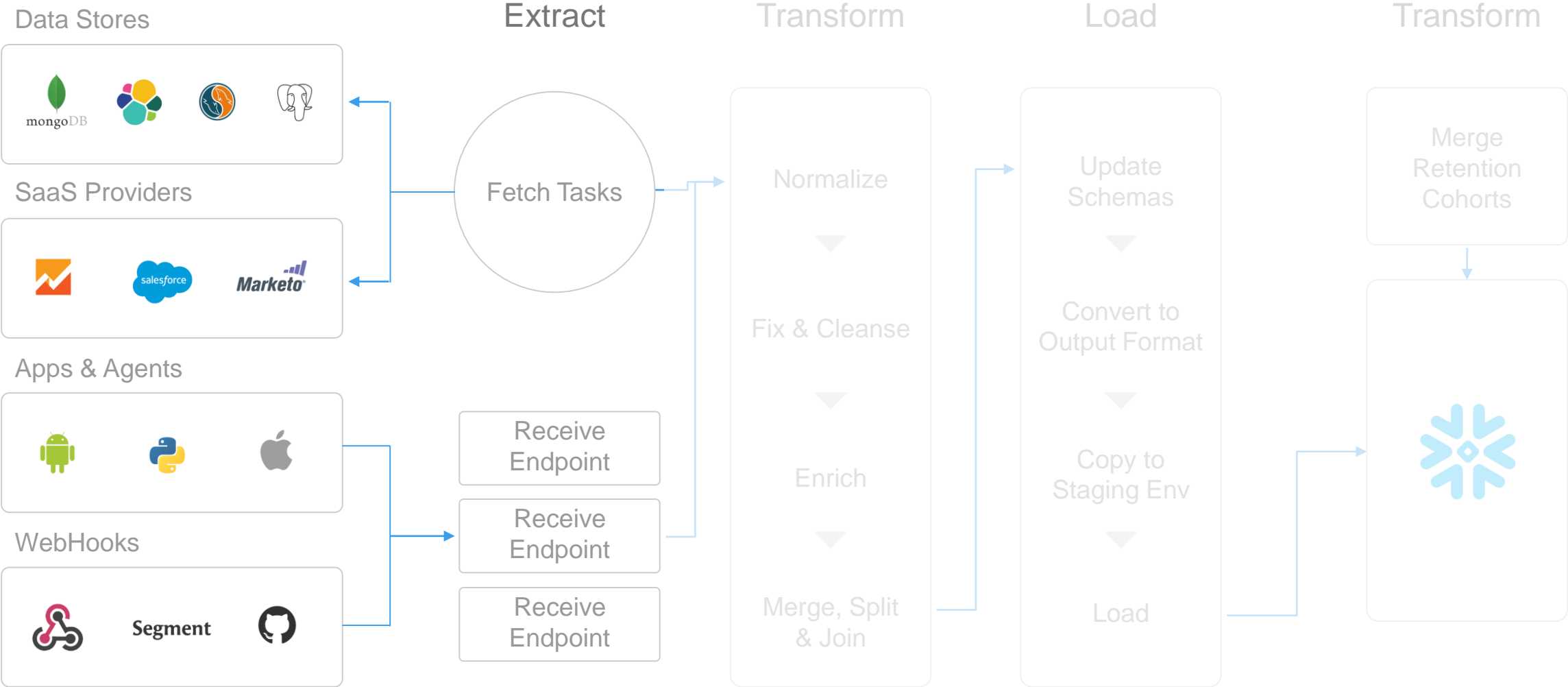
Getting Data to the Cloud



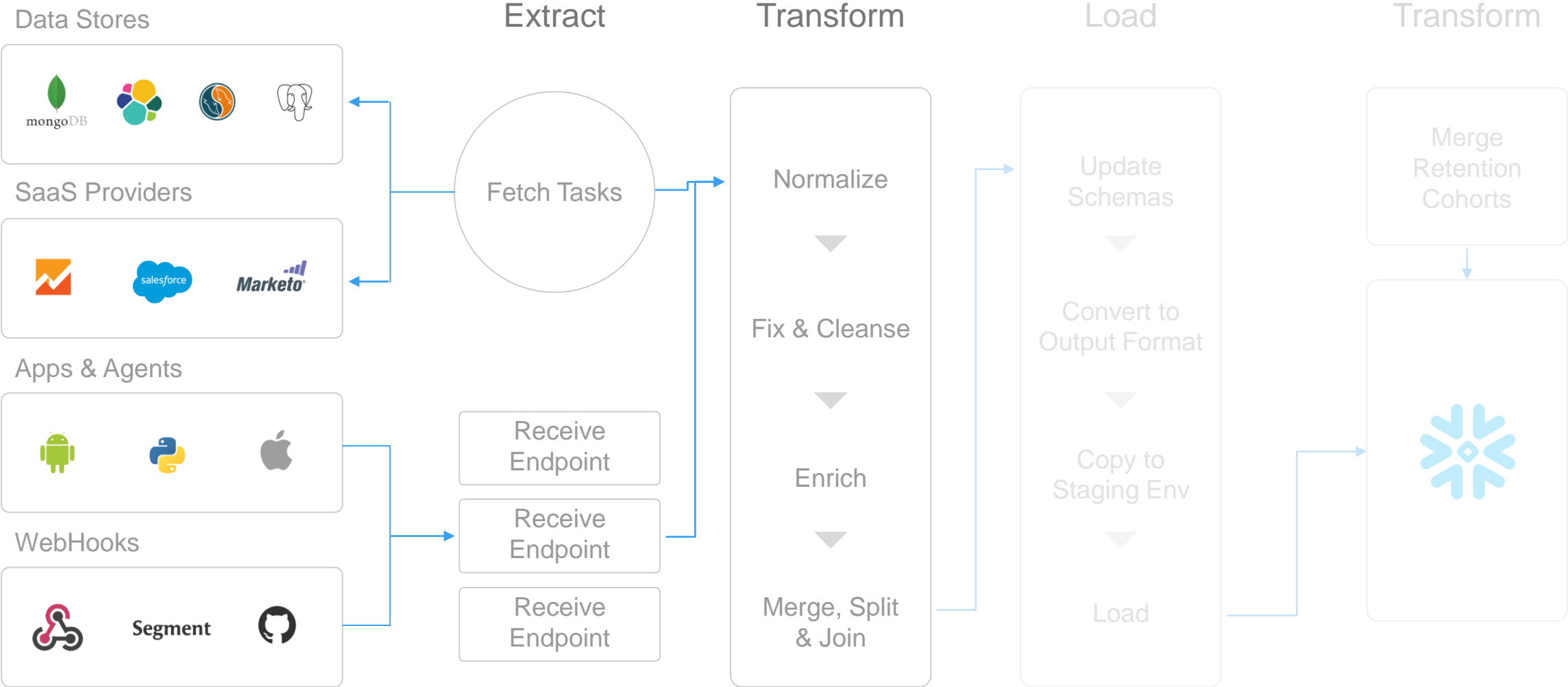
Basic Architecture of a Unified Data Pipeline



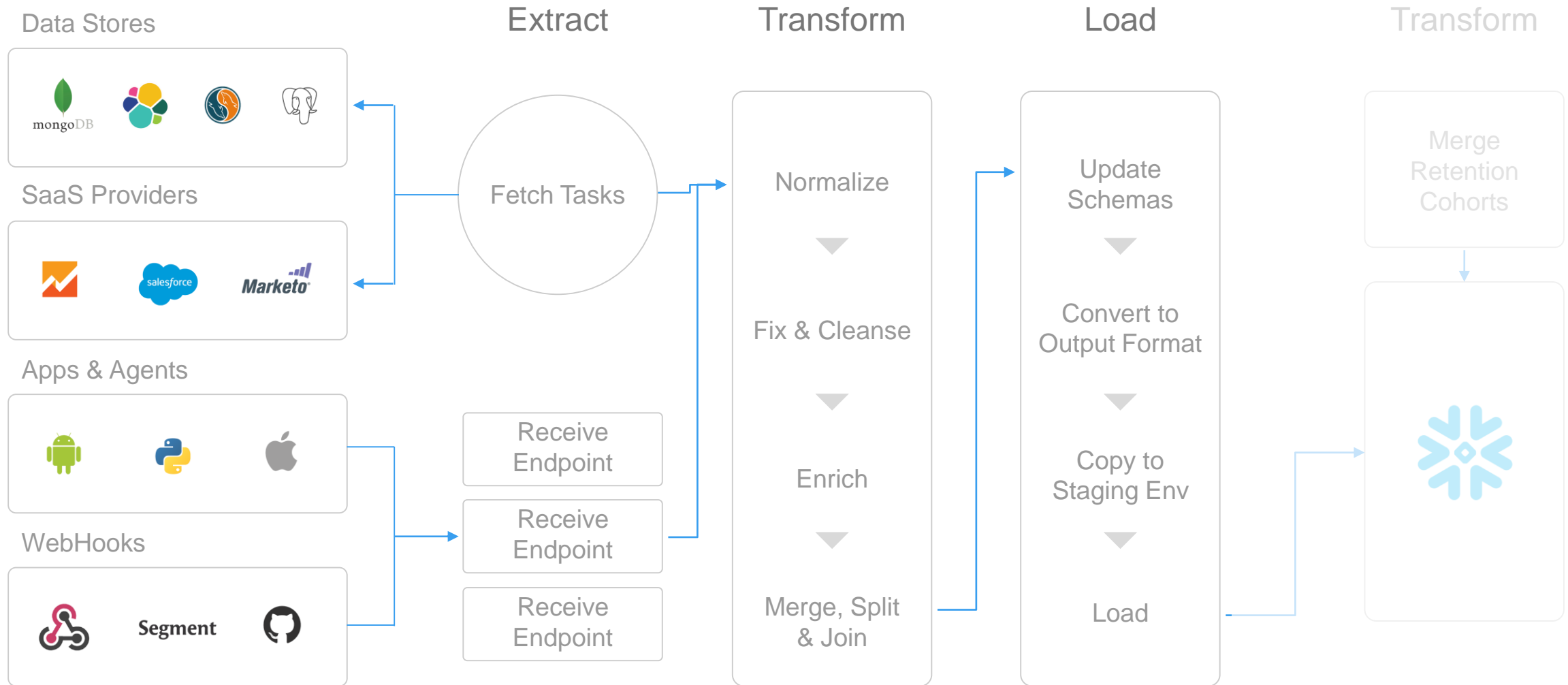
Basic Architecture of a Unified Data Pipeline



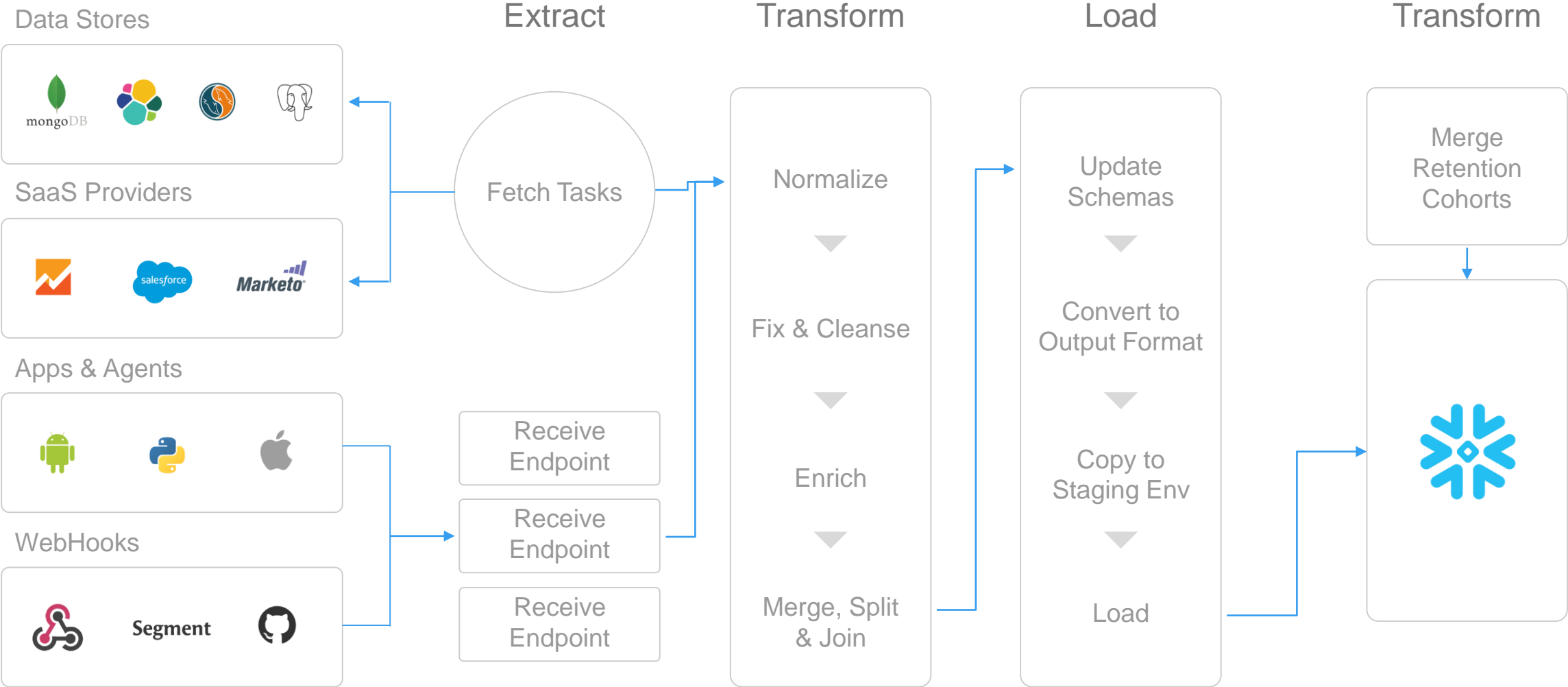
Basic Architecture of a Unified Data Pipeline



Basic Architecture of a Unified Data Pipeline



Basic Architecture of a Unified Data Pipeline



Main Challenge of a Data Pipeline is Integrity



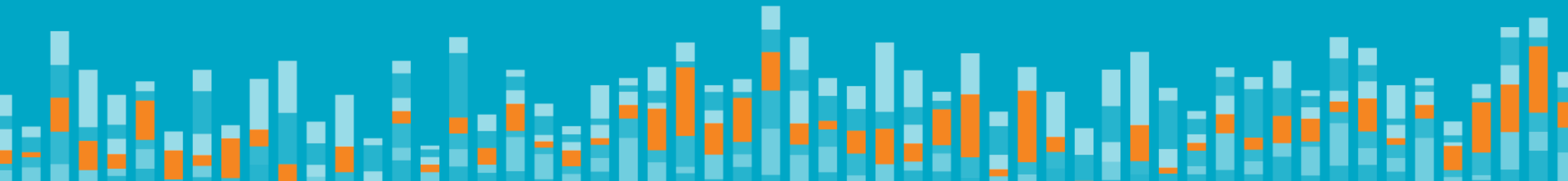
Managing
Dynamic
Schemas

Handling
Errors

Transmitting
Data
Exactly Once

Let's tackle dynamic schemas

Because the only constant, is change.



Challenge: Schemas Always Change

New tables,
New columns

NoSQL ==
No schema

New features,
New events

Backend apps require
monitoring

Weekly
A/B tests

New Marketing
tools

Salesforce
data

Partial Solution: “Guesstimate”

- Very early on we built our *Automapper*:
 - Periodic process: Detect Schema Changes
→ Modify Schemas on data warehouse
→ Update schema mappings
- Detecting schema changes was based on sampling the events, and guessing the data types of the different values

The screenshot shows the Aloomo Workspace Mapper interface. The browser address bar displays `https://app.alooma.com/alooma/#/mapper/AWS-billing`. The interface is titled "Mapper" and shows a loading status for "AWS-billing" from the source "public.aws_billing".

The main area displays a table of field mappings:

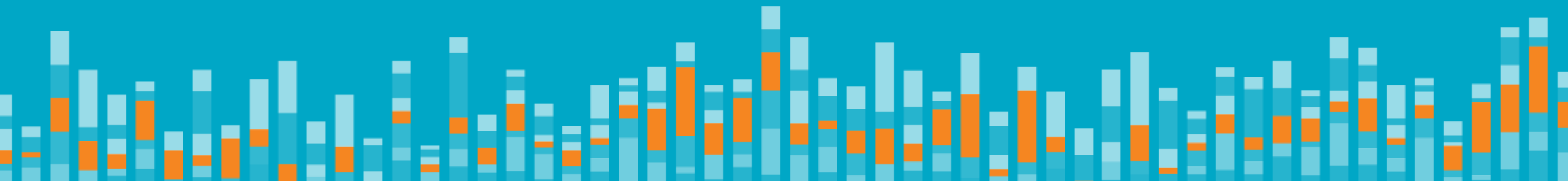
Field Name	Source Field	Target Data Type
RateId	rate_id	integer
ProductName	product_name	varchar(64)
Operation	operation	varchar(64)
UsageType	usage_type	varchar(64)
ItemDescription	item_description	varchar(128)
Cost	cost	double precision
UnBlendedRate	un_blended_rate	double precision
user:Client	user_client	varchar(1024)
SubscriptionId	subscription_id	varchar(64)
RecordType	record_type	varchar(64)
PayerAccountId	payer_account_id	double precision
UnBlendedCost	un_blended_cost	double precision
RecordId	record_id	double precision
user:Name	user_name	varchar(1024)

On the right side of the interface, a large number "166.1M" is displayed, with "Events" written below it.

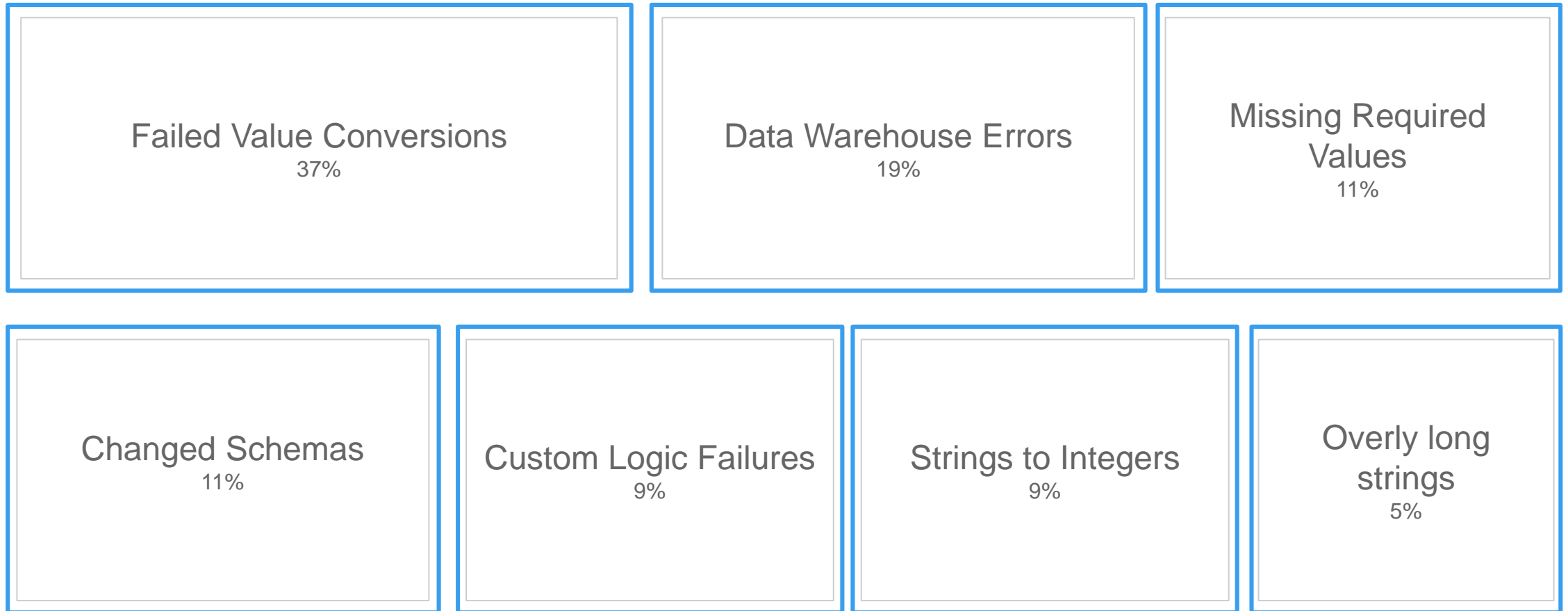
More Complete Solution: “Guesstimate” + Schema Import

- Import schemas from any data source that supports it
 - Added an `import_schema(schema_id)` function to our **inputs**:
MySQL, Postgres, MSSQL, Oracle, Mongo, Salesforce, Google Adwords, ...
 - Added a `schema_id` metadata field to all **events**
 - New Automapper:
 - Detect Schema Change
 - Try to extract `schema_id` and import the updated schema
 - Translate source schema to data warehouse schema
 - Modify Schemas on data warehouse
 - Update schema mappings

OK, but what about errors?



Challenge: The Pipeline May Fail at Any Stage



* Statistics taken from 1B events in our Restream Queues

Solution: Store All Failed Events and Failure Details

- Build a central, “catch-all,” event store that can also save error details
- Program all pipeline components to write to it in case of error
- Always store the original, unmodified event to allow simple re-processing
- Design a querying interface to enable easy classifying, grouping, and prioritizing of errors

- Beware of:
 - pipeline loops
 - multiple pipeline destinations
 - complex processing (event splitting / joining / enrichment)

Implementation V1

- Kafka queue to store all failed events (in original format)
- Metadata on each failed event:
 - Failure reason
 - # of failures (to avoid loops)
- Notification service which aggregates failure notifications by error type, similar parameters
- Redis to store samples of failed events

Implementation V2

The screenshot shows the Aloomo Restream Queue interface. At the top, it displays the URL `https://app.alooma.com/alooma/#/restream?page=1&count=10&sorting=%7B%7D` and a sidebar with navigation options like Plumbing, Dashboard, Live, Restream, Code Engine, Mapper, Settings, and Log Out. The main content area features a large '851 Events in queue' counter with a 'Restream' button. Below this is a horizontal bar chart showing the distribution of event types: Job_Listin... (741), app_alooma... (93), discrepan... (7), backend_Jo... (4), Salesforce (3), alooma_log... (2), and tst (1). To the right is a line chart showing event counts over time from 2017-01-11 to 2017-03-17, with a prominent spike on 2017-02-06. A filter bar allows filtering events by input label, event type, error type, and date. The main table lists event details:

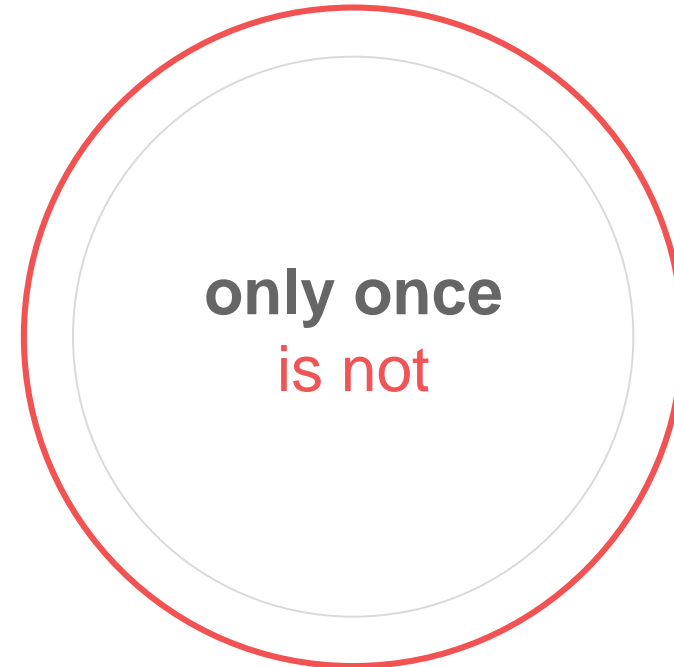
Input Label	Event Type	Error Type	Timestamp	Event
app_alooma_logs	No event type	Partial mapping Missing mapping for the following field S...	Mar 28, 2017 8:08:33 AM	<code>{"_metadata": {"input_label": "app_alooma_logs", "input_type": "REST API", "client_ip": "52.42.131.65", "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IWR5bG93IiwiaWF0IjoiMjAxNzAzMjg0MCJ9"}}</code>
app_alooma_logs	No event type	Partial mapping Missing mapping for the following field S...	Sep 16, 2015 3:26:06 AM	<code>{"_metadata": {"input_label": "app_alooma_logs", "input_type": "REST API", "client_ip": "54.69.140.154", "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IWR5bG93IiwiaWF0IjoiMjAxNSA5MTYwNiJ9"}}</code>
app_alooma_logs	No event type	Partial mapping Missing mapping for the following field S...	Sep 16, 2015 3:26:06 AM	<code>{"_metadata": {"input_label": "app_alooma_logs", "input_type": "REST API", "client_ip": "52.42.131.65", "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IWR5bG93IiwiaWF0IjoiMjAxNSA5MTYwNiJ9"}}</code>
app_alooma_logs	No event type	Varchar too long The value of "queryParams.metrics" is l o...	Sep 16, 2015 3:26:06 AM	<code>{"_metadata": {"input_label": "app_alooma_logs", "input_type": "REST API", "client_ip": "52.43.222.15", "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IWR5bG93IiwiaWF0IjoiMjAxNSA5MTYwNiJ9"}}</code>
app_alooma_logs	No event type	Partial mapping Missing mapping for the following field S...	Sep 16, 2015 3:26:06 AM	<code>{"_metadata": {"input_label": "app_alooma_logs", "input_type": "REST API", "client_ip": "52.42.131.65", "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IWR5bG93IiwiaWF0IjoiMjAxNSA5MTYwNiJ9"}}</code>

Uh oh, won't this mean duplicates?

Or accidentally overwriting newer data with old data?



Challenge: Transmit All Events Once, and *Only* Once



Solution: Idempotency

idempotent | ,īdem'pōt(ə)nt 'ēdem,pōt(ə)nt | *Mathematics*

adjective

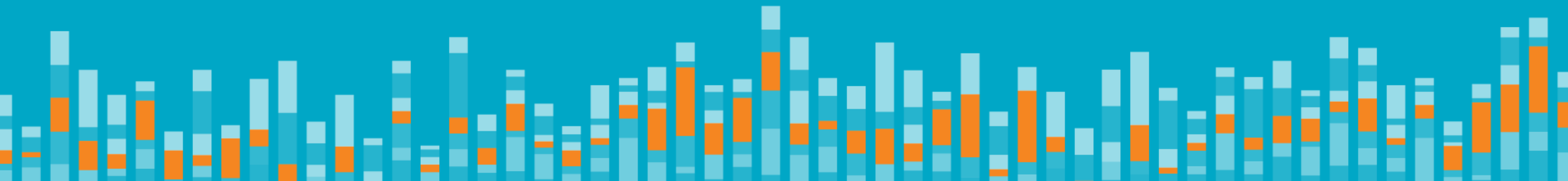
denoting an element of a set that is unchanged in value when multiplied or otherwise operated on by itself.

- In simple words: a duplicate event will overwrite itself
- In practice, this is still difficult:
 - What identifies an event? Its content ? Its ID ?
 - Not all data warehouses support idempotency
 - Every component in the pipeline may fail and resend its last few events upon resuming

Implementation: Track IDs and Batches End-to-End

- Identifying events & duplicates
 - Generate an event ID & batch ID as early as possible (at the data source)
 - Where possible, make the IDs sequential
- Idempotency with cloud data warehouses
 - Primary key constraints are not validated on cloud data warehouses, so overwriting is not an option
 - We must keep track of loaded events (or batches) in a separate table and:
 - Verify writing to both tables is atomic (using a transaction)
 - Check whether events have been written before writing every new batch
 - Expand this to encompass your whole pipeline, either end-to-end or step-by-step

So does all of this work?





Alooma just works. We're able to leverage the platform's real-time architecture to **iterate on solutions in minutes rather than days**, giving us the agility our business demands.”



WARGAMING



Alooma makes it easy for us to join data from across all marketing channels **without draining our internal engineering resources** on untested solutions.

SONY



Alooma gives our marketing organization **real-time data analytics pulled from across all app and ad sources** without draining our internal engineering resources or sacrificing our security practices.

STRAVA



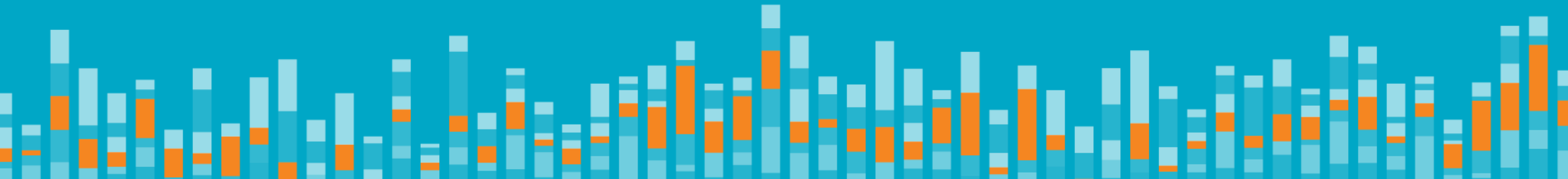
Alooma provides us a scalable, robust and flexible infrastructure needed to **accelerate our real-time analytics across the entire company.**



Can I see this in action?



Closing Thoughts



Closing Thoughts

- By leveraging a cloud based data warehouse and ETL platform, you dramatically improve data usage across the organization with more reliable data
- 99.999% data integrity requires design and tested implementation
- When you plan your next analytics platform:
 - Plan for a highly scalable, simple to manage data warehouse
 - Include a schema repository & automated mapping tasks
 - Integrate sources with end-to-end idempotency in mind
 - Design a catch-all error queue & an iterative manual fixing process
- Come say hi and **catch a live demo at our booth: #417**

Questions?

max@alooma.com – booth #417

