ASCEND

Intelligent Orchestration: Data's missing link

Sean Knapp,
Founder @ Ascend.io

Topics

- Quick Intro
- The State of Data Architectures
- Why Pipelines Suck
- Building a new Control Plane
- Making it Scale
- SaaS-ifying
- Future Topics

Quick Intro

About Ascend

- Founded 2015
- Team of 30
- We <3 Data Pipelines

About Me (Sean Knapp) 👋🏻

ASCEND.IO OOYALA Google

- 15 years building data platforms & teams
- Search Frontend TL @Google:
  first MapReduce in 2004
- Founder & CTO @Ooyala: 4B+ events/day
- Founder & CEO @Ascend: 1T+ events/day

Smarter Pipelines.
Less Code.

DATA LAKE

Amazon S3 · Azure Blob · Google Cloud Storage

DATA WAREHOUSE

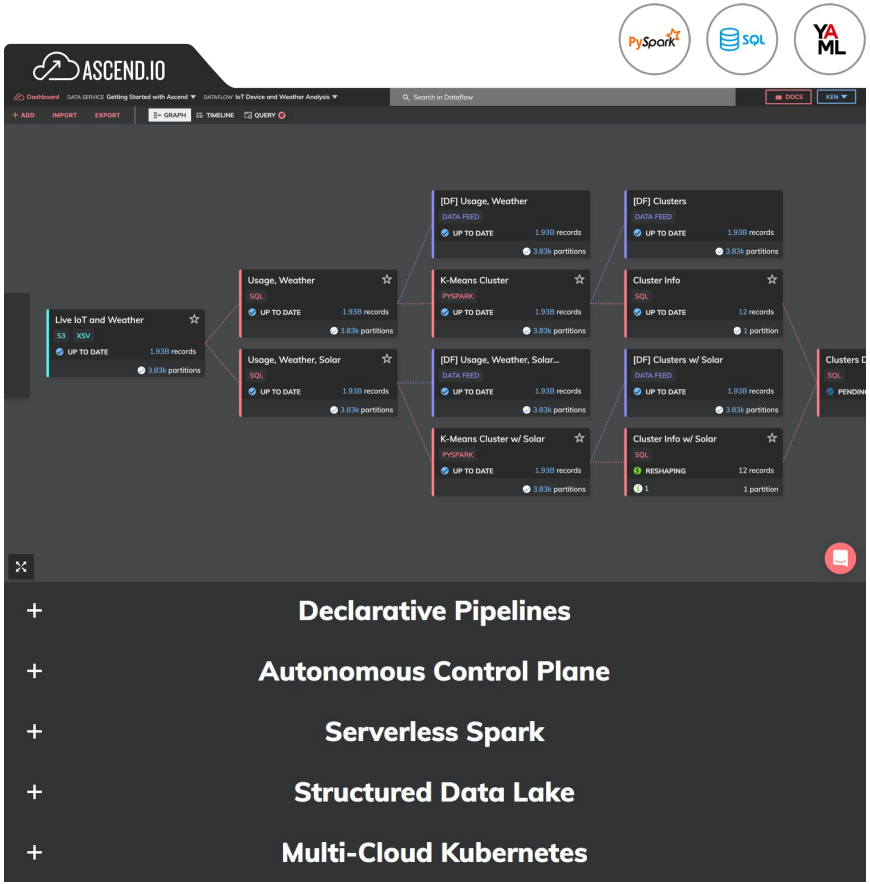
Amazon Redshift · Google BigQuery · Snowflake

DATABASE

MySQL · Postgres · SQL Server

STREAMING

Amazon Kinesis · Apache Kafka



NOTEBOOKS

Apache Zeppelin · Jupyter Notebook

MACHINE LEARNING

DataRobot · H2O.ai

BI & VIZ

PowerBI · Tableau

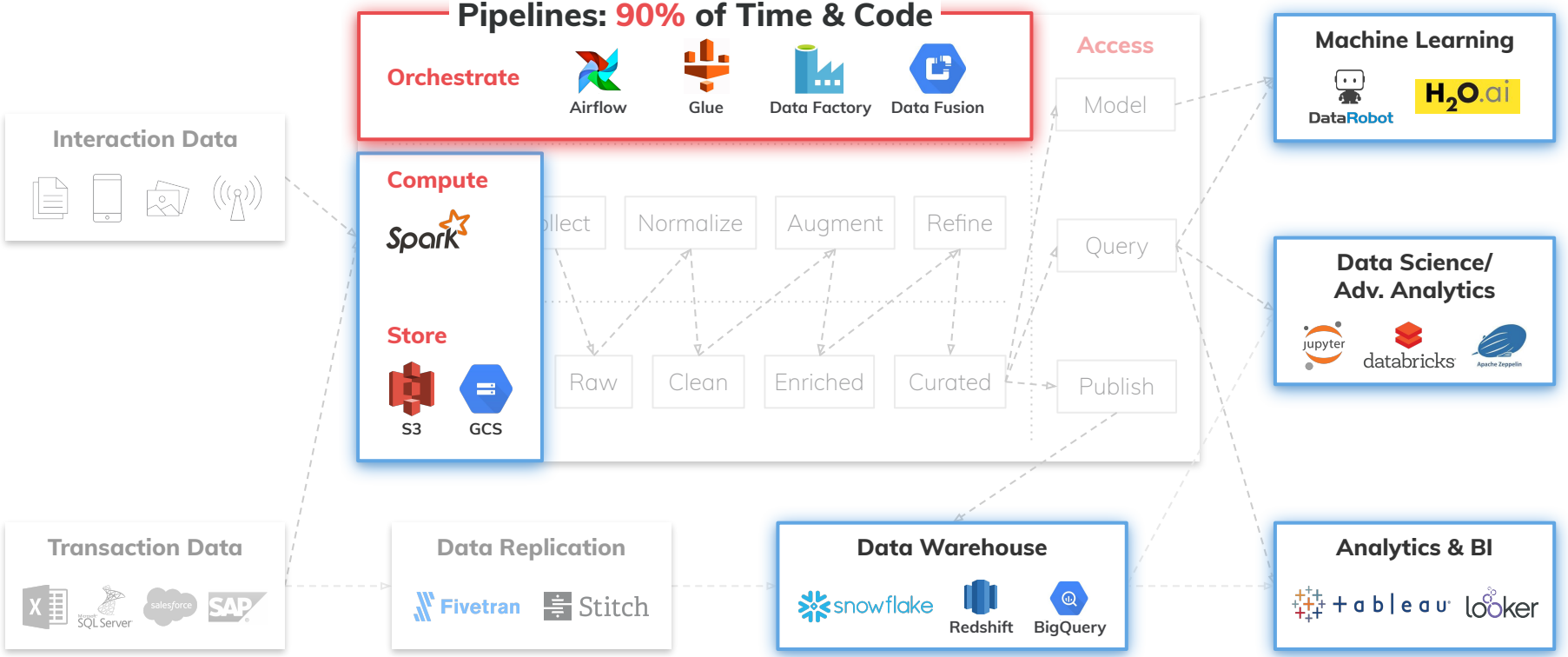
APPS

Appsflyer · Facebook · Salesforce

Despite advancements in
every other part
of the data lifecycle...

Building. Pipelines. SUCKS.

The Current State of Data Architectures



Why Pipelines Suck

Why Pipelines Suck

Databases & Warehouses

Where this...

```
1   SELECT date, country, gender, SUM(clicks)
2   FROM user_events
3   WHERE date >= DATE_SUB(NOW(), INTERVAL 30 DAY)
```

Database Engine

MySQL (1.5M lines of code)

ORACLE (25M lines of code)

Results

Pipelines

...becomes 1,000s of lines of this...

- Monitoring For New Data
- Ingest & Reformat Data
- Profile & Partition Data
- Analyze Downstream Dependencies
- Incremental Processing & Updates
- Intermediate Persistence
- Data & Task Deduplication
- Spark Parameterization & Tuning
- Data Consistency & Integrity Checks
- Error-Handling, Classification & Recovery
- Data Lineage & Privacy Compliance
- Garbage Collection & Lifecycle Management

We are ***manually creating*** a query plan, for ***every*** stage, of ***every*** pipeline

Evolution of Pipeline Orchestration

	1.0	2.0
Hosting Model	Roll-Your-Own	SaaS
Code Generation	Manual	Templatized
Interaction Model	Code	Code + GUI
Control System	Scheduler	Scheduler
Programming Model	Imperative	Imperative
Examples	cron Airflow OOZIE Luigi	AWS Glue, Azure Data Factory, Google Data Fusion

We looked for ideas in adjacent spaces…

Who here has used a **Database**?

Who here has heard of
React?

Who here uses
Kubernetes?

What do they all
have in common?

They're **Declarative**.

> *Declarative programming is a programming paradigm that expresses the logic of a computation without describing its control flow...*
>
> *... [in an] attempt to* ***minimize or eliminate side effects by describing what the program must accomplish****...*
>
> *... rather than describe how to accomplish it.*

Declarative vs. Imperative

The **desired** outcome

What **vs** How

Declarative

Imperative

The **usual** outcome

Declarative

Imperative

Declarative vs. Imperative

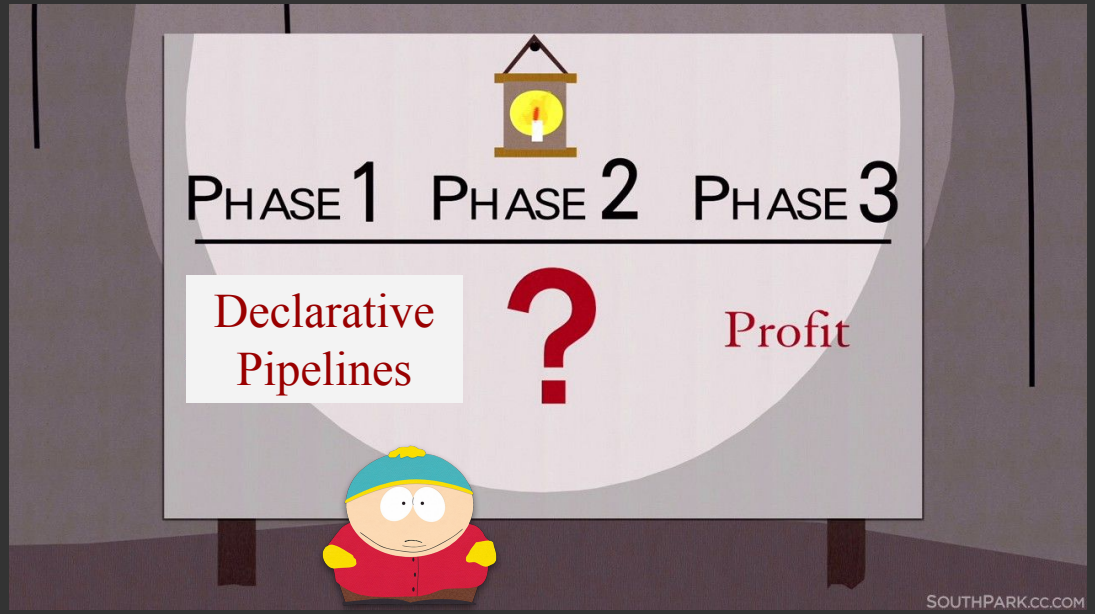
	Declarative	Imperative
Pros	✓ **Less Code** (like.. A LOT less) ✓ **Faster Dev Cycles** ✓ **Adaptive to Changes** ✓ **Less Maintenance**	✓ **Flexible** ✓ **High Levels of Control**
Cons	✗ Domain specific ✗ Difficult to manually optimize ✗ Require annotations to override automated behaviors	✗ State Management ✗ Stale assumptions in code ✗ Manual Optimizations ✗ Integrity Checks ✗ Failure Management

> *Imperative gives you the ability to do anything, and* ***the responsibility to do everything****.*

— *Steven Parkes, CTO @ Ascend*

Building a Control System for Declarative Pipelines

Our master plan...



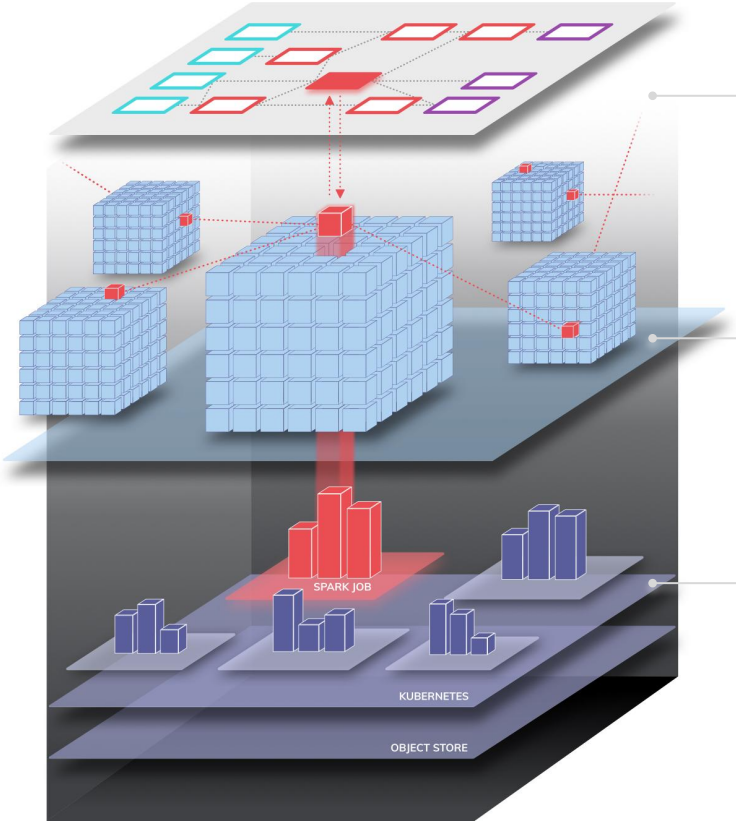
What should a
good control system
do?


Dynamic Partitioning
Data Backfill
Metadata Centric
Data-aware/Profiling
Dynamic Scheduling (time + trigger + auto)
Global optimization + local namespacing / sandboxing
Multi-Cloud + Region + Zone + etc.
Static Type Checking
SLA-driven Scheduling
Graph Algorithms
Aggregate Data Types
Lineage Tracking
Provenance
Data Consistency & Integrity Checks --> Data Repair
Inter-transform Optimization
SQL + Other Languages (Python & Pyspark)
Separation of Logic & Execution
unctional (Immutable of Blocks)
Garbage Collection
Declarative
Functional Decomposition
Hierarchical Components
Asynchronous Scheduling
Configuration Management
Pipeline Rewriting
Dataflow Scheduling
Storage Optimization
Error Classification & Intelligent Retries
Semantically Transparent
Graph Rewriting
Pipeline Evolution
Data & Task Deduplication
Static Type Influence
Time series Optimizations

Then we spent ~~1~~, ~~2~~, 3 years building it!

So... **how does it work**?

Separation of Logic & Control



Logic Plane

User defined logic

Control Plane

Dynamic task generation to achieve desired state

Data Plane

Fully managed, portable cloud services

The Control System Answers

1. Is there anything I need to do?
2. What is the current state of my world?
3. What should it be?
4. What doesn't match?
5. How do I "fix" it?

1) Is there anything I need to do?

A Simple Analytics Pipeline

2) What is the current state of my world?

A Simple Analytics Pipeline

Logic Plane

Source: files in S3
- Location
- Creds
- Schema

→ **Transform (SQL)**
- Filter
- Join / Enrich

→ **Transform (SQL)**
- Daily Aggregation

→ **Write to DB**
- Hostname
- Creds
- Update Rules

Control Plane

Fragments
- Stored `[bucket]/[uuid]/`
- Metadata
  - **P-SHA**
  - **UUID**
  - Data SHA
  - Data Profile
  - Job Statistics

P-SHA
- Partition-level SHA

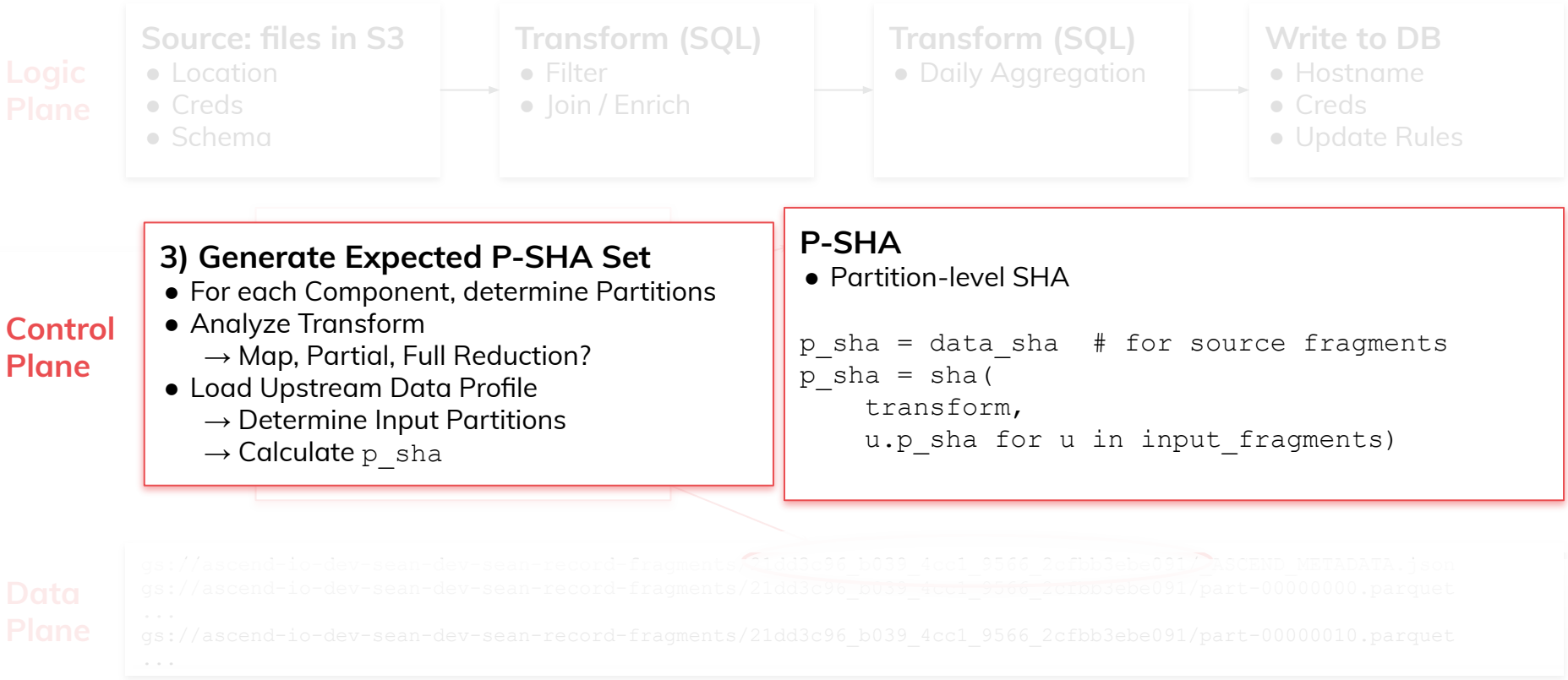
```
p_sha = data_sha  # for source fragments
p_sha = sha(
    transform,
    u.p_sha for u in input_fragments)
```

Data Plane

```
gs://ascend-io-dev-sean-dev-sean-record-fragments/21dd3c96_b039_4cc1_9566_2cfbb3ebe091/_ASCEND_METADATA.json
gs://ascend-io-dev-sean-dev-sean-record-fragments/21dd3c96_b039_4cc1_9566_2cfbb3ebe091/part-00000000.parquet
...
gs://ascend-io-dev-sean-dev-sean-record-fragments/21dd3c96_b039_4cc1_9566_2cfbb3ebe091/part-00000010.parquet
...
```

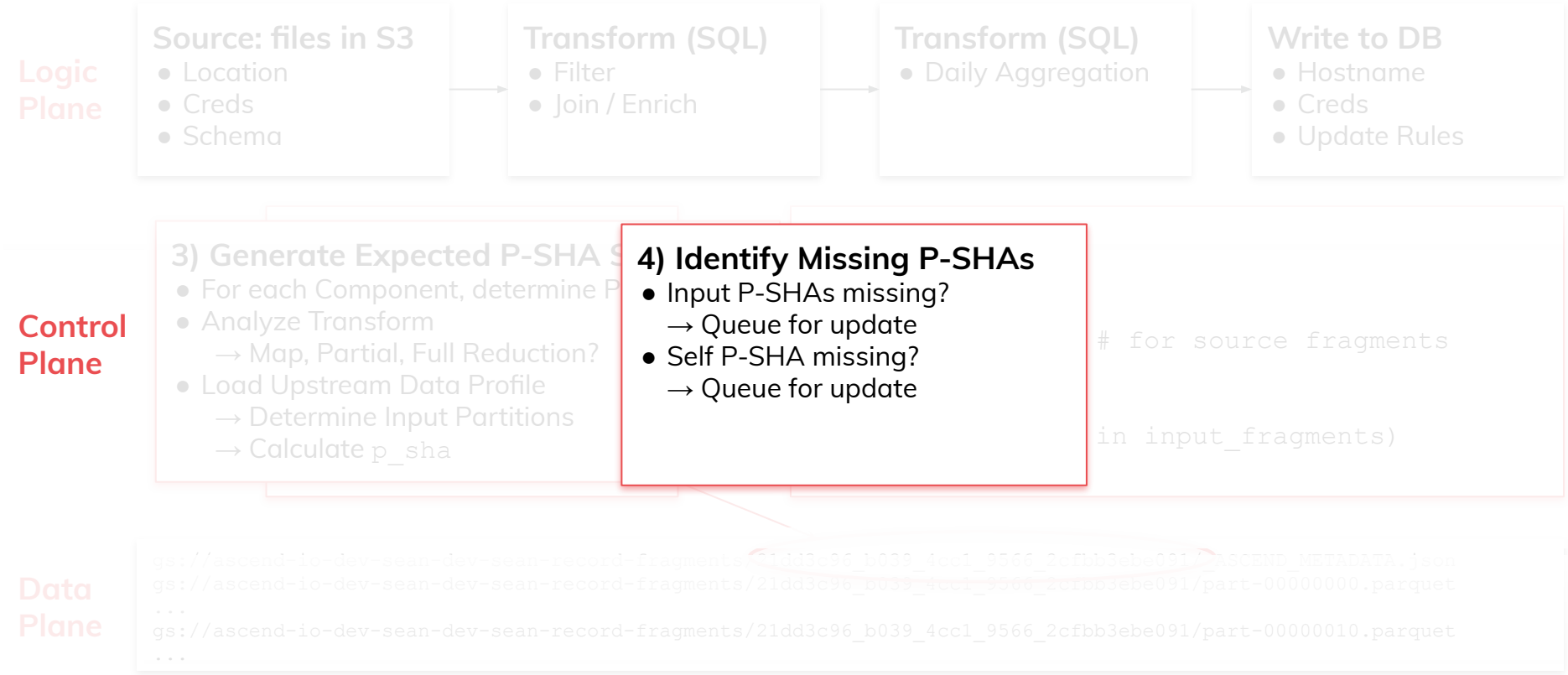
3) What should it be?

A Simple Analytics Pipeline



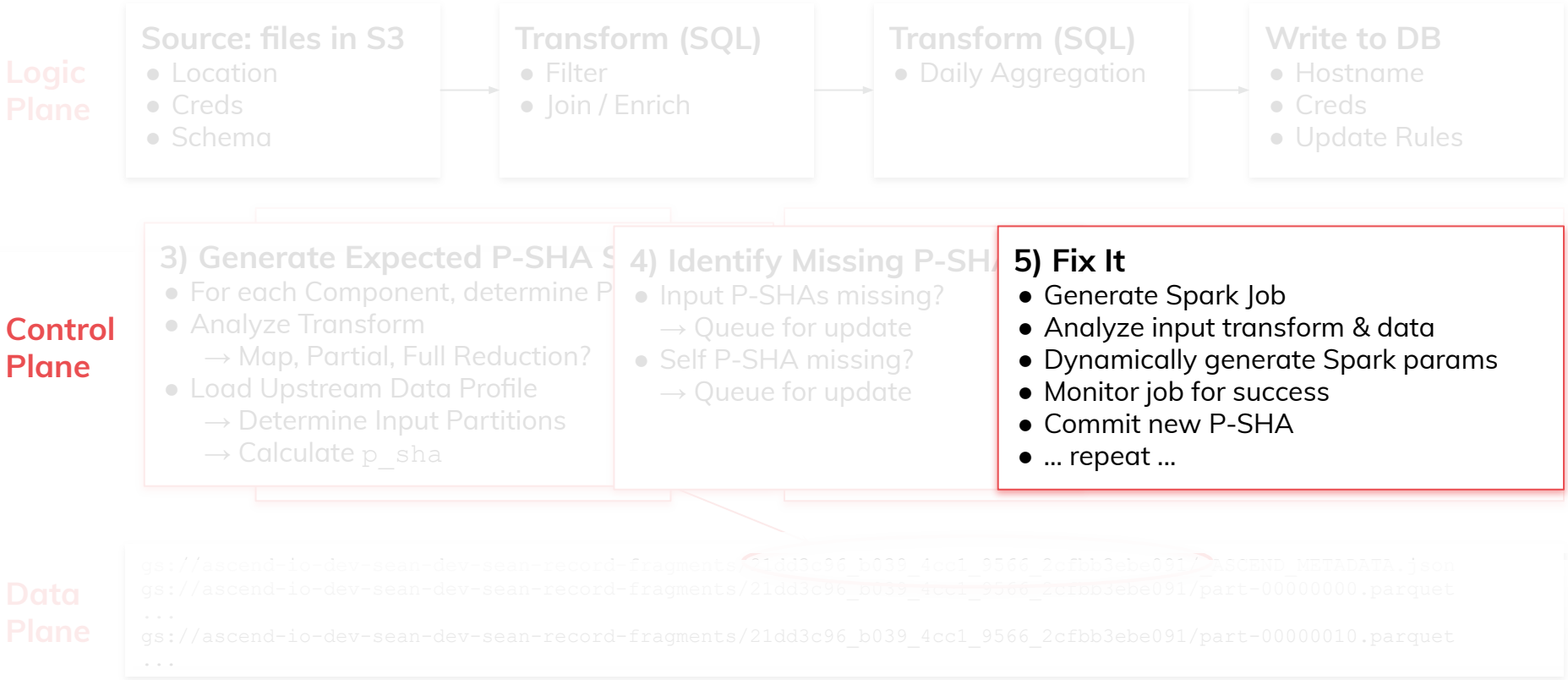
4) What doesn't match?

A Simple Analytics Pipeline



5) How do I “fix” it?

A Simple Analytics Pipeline



Scaling to **1B Partitions**
and **1T+ records per day**

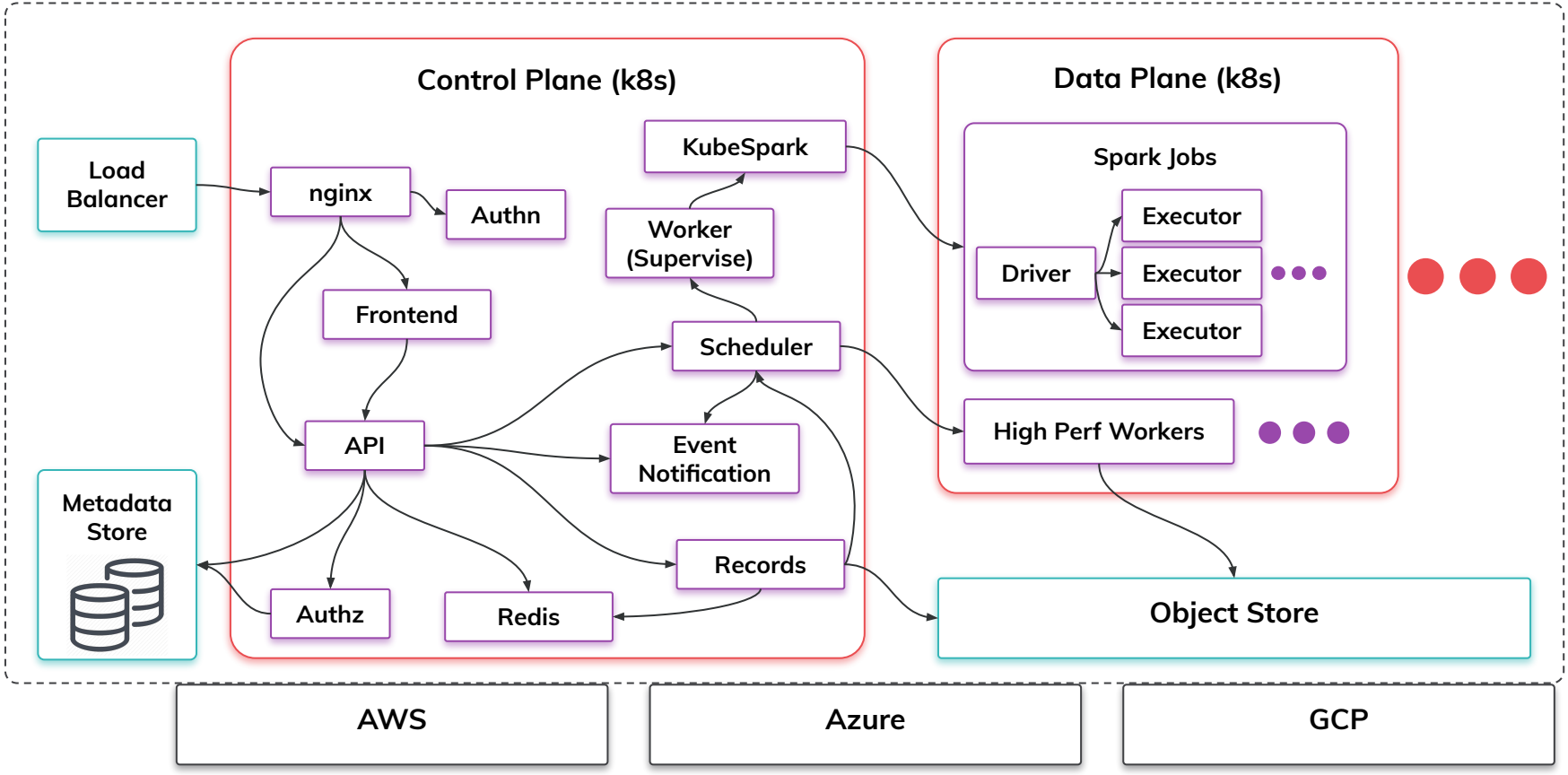
Scaling the Control Plane

- **SHAs**: lots and lots of SHAs (and SHAs of SHAs)
- **Caching**: lots and lots of caching
- **Trees, not lists**
  - Leverage time-series partitions
  - Aggregate metadata
  - SHAs at each node, not just leaves
- **Be Lazy**: Only do as much work as is useful right now

Scaling the Data Plane

- **Do less work**
  - Intermediate Data Persistence
  - Data & Task De-duplication
- **Do the right kind of work**
  - Small file aggregation
  - Small job optimizations (local mode)
  - Specialized compute pools for different tasks
- **Do it efficiently**
  - Auto-Scaling Spark on Kubernetes w/ Spot/Preemptible Instances
  - Single-zone clusters (reduce network costs)

SaaS-ifying the Control Plane



What we didn't discuss...

- **Garbage Collection:** background task
- **Multi-cloud abstractions:** k8s, MinIO
- **Data repair**: failure to retrieve data → delete p-sha → self-heal
- **Part files**: similarities & differences with other fragments
- **Resource Management**: capacity-aware Control Plane
- **SLA Driven Scheduling & Job Prioritization**:
  per-component priority + upstream inheritance
- **Scaling Spark on Kubernetes**
- **Scaling to 100+ environments**: terraform, templates, monitoring, & automation

Come ask us @ Office Hours!!!
(4th floor in 476a)

> *Declarative programming is a paradigm that expresses the logic of a computation without describing its control flow...*
>
> *... [in an] attempt to* ***minimize or eliminate side effects by describing what the program must accomplish****...*
>
> *... rather than describe how to accomplish it.*

tl;dr

Declarative

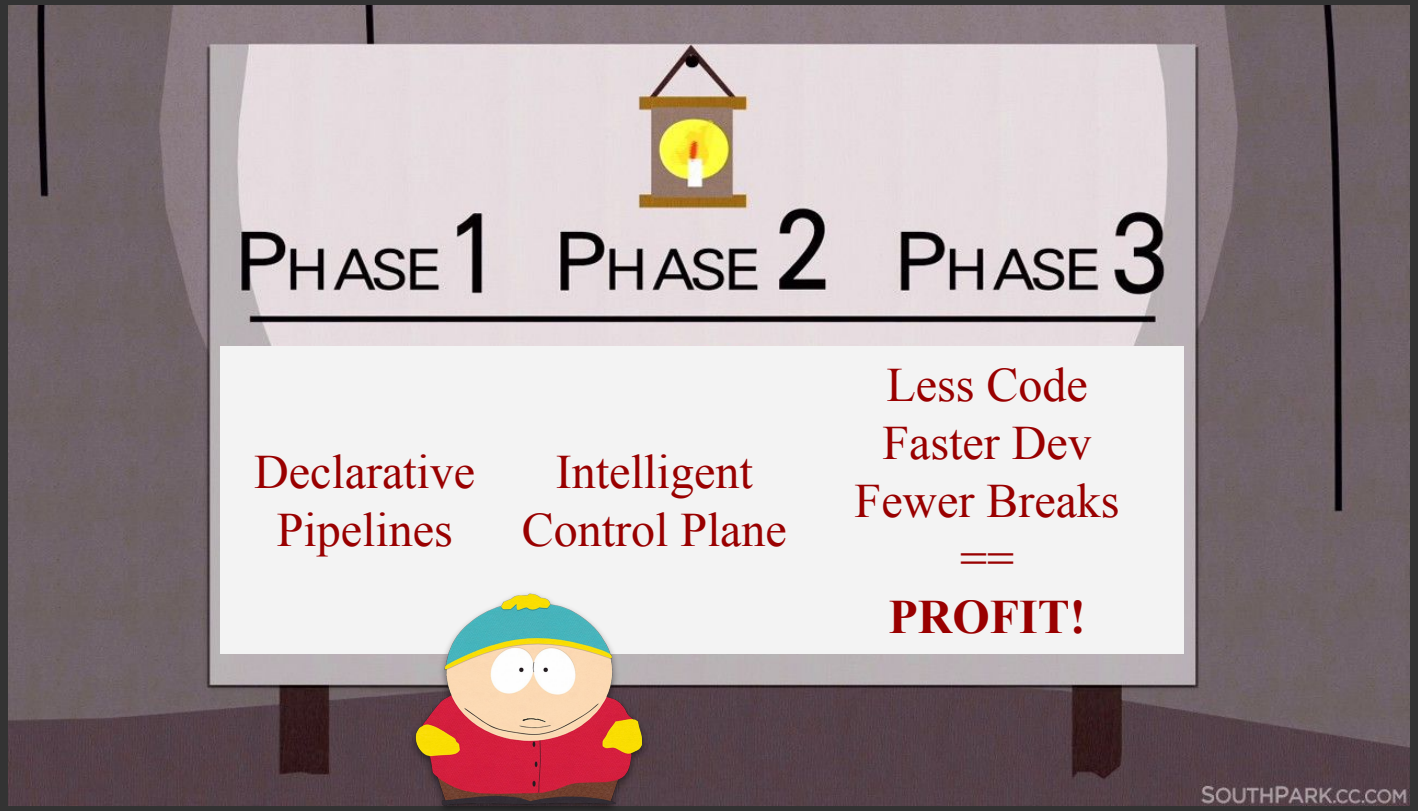
The What

Logic + Data → Tasks

Imperative

The How

State + Tasks → Data


PHASE 1
PHASE 2
PHASE 3
Declarative Pipelines
Intelligent Control Plane
Less Code
Faster Dev
Fewer Breaks
==
PROFIT!
SOUTHPARK.CC.COM

Smarter Pipelines.
Less Code.

DATA LAKE

Amazon S3 · Azure Blob · Google Cloud Storage

DATA WAREHOUSE

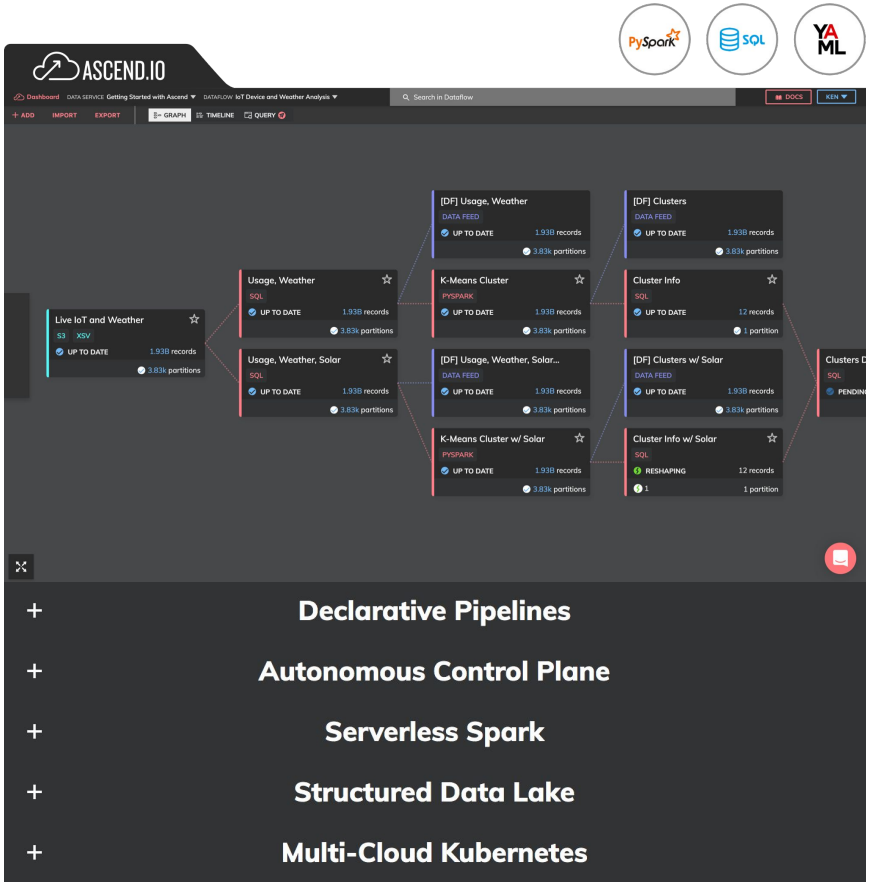
Amazon Redshift · Google BigQuery · Snowflake

DATABASE

MySQL · Postgres · SQL Server

STREAMING

Amazon Kinesis · Apache Kafka



NOTEBOOKS

Apache Zeppelin · Jupyter Notebook

MACHINE LEARNING

DataRobot · H2O.ai

BI & VIZ

PowerBI · Tableau

APPS

Appsflyer · Facebook · Salesforce

See You in Office Hours

- Right after this, 4th floor, 476a
- Ask me anything
- Meet our CTO, Steven Parkes
- Visit our booth at the **Partner Spotlight** gallery for a live demo & free swag
- Visit us @ **www.ascend.io**