Spotify®

ClickHouse for Experimentation

Gleb Kanterov
@kanterov
2018-07-03

Quick Facts

170M Monthly Active Users
75M Subscribers
35M Tracks
65 Markets

[1] https://investors.spotify.com

Organization

1 Company

10 Organizations

30+ Tribes

150+ Squads

Organization

1 Company

10 Organizations

30+ Tribes

150+ Squads

Move fast, break things

Organization

1 Company

10 Organizations

30+ Tribes

150+ Squads

Move fast, break things
Ask for forgiveness, not for permission

Organization

1 Company

10 Organizations

30+ Tribes

150+ Squads

Move fast, break things
Ask for forgiveness, not for permission

AUTONOMY

Hadoop@Spotify

Hadoop@Spotify

- On-Premise
- 2,500 nodes
- 100 PB Disk
- 100 TB RAM
- 100B+ events per day
- 20K+ jobs per day

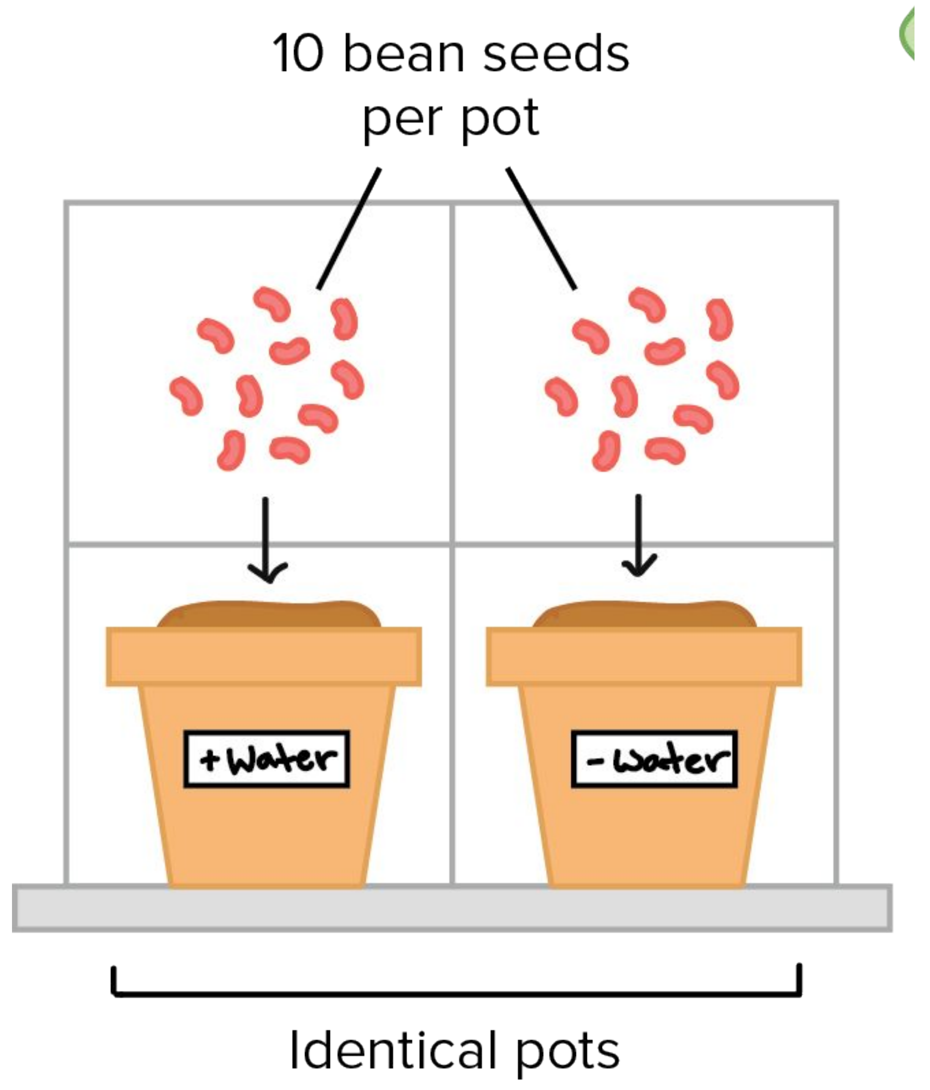
Hadoop@Spotify

- Migration from On-Premise to GCP
- Moved 100 PB of data
- Our Hadoop cluster is dead

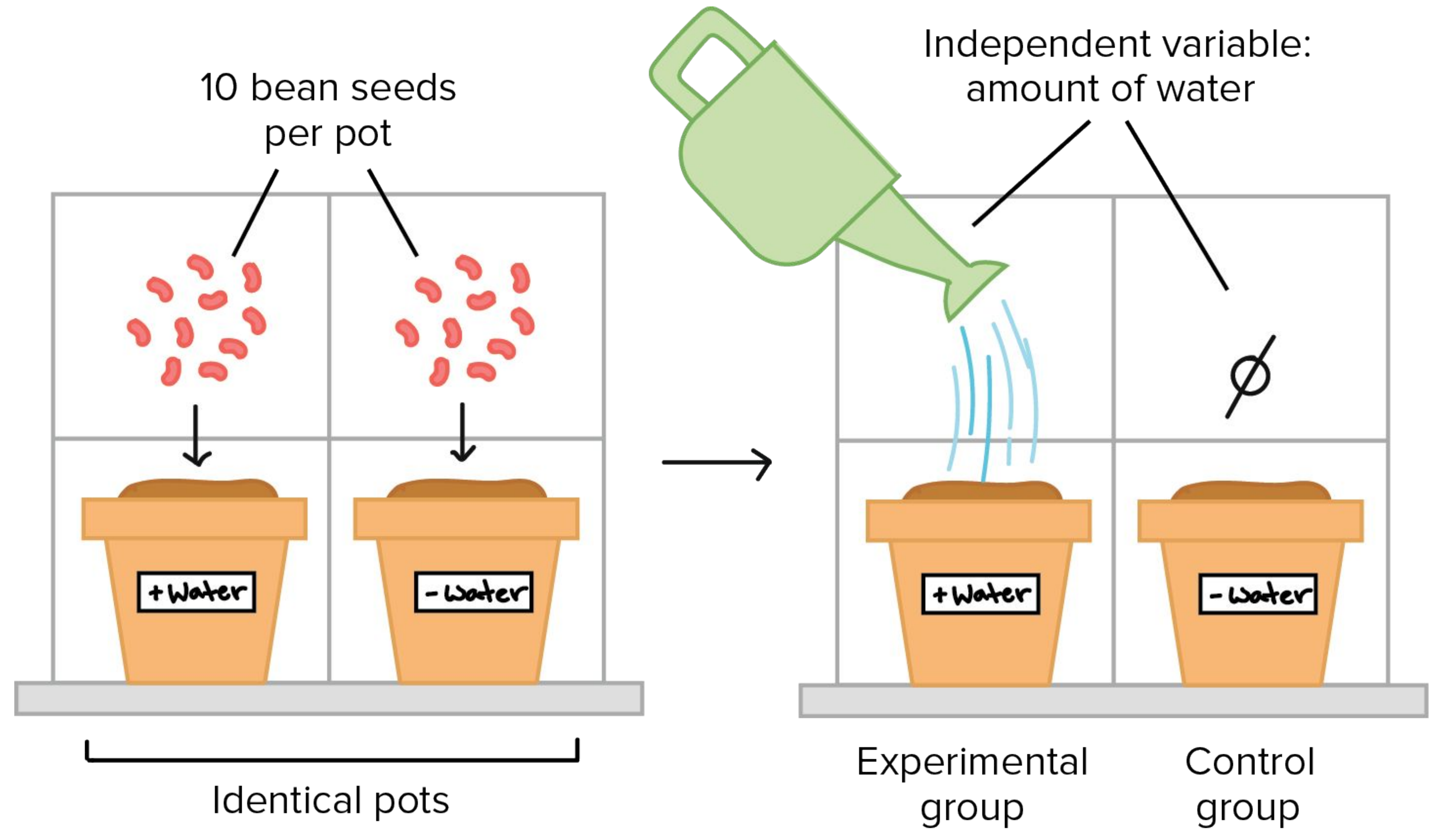
What are experiments, and why ClickHouse?

Randomized Controlled Experiment

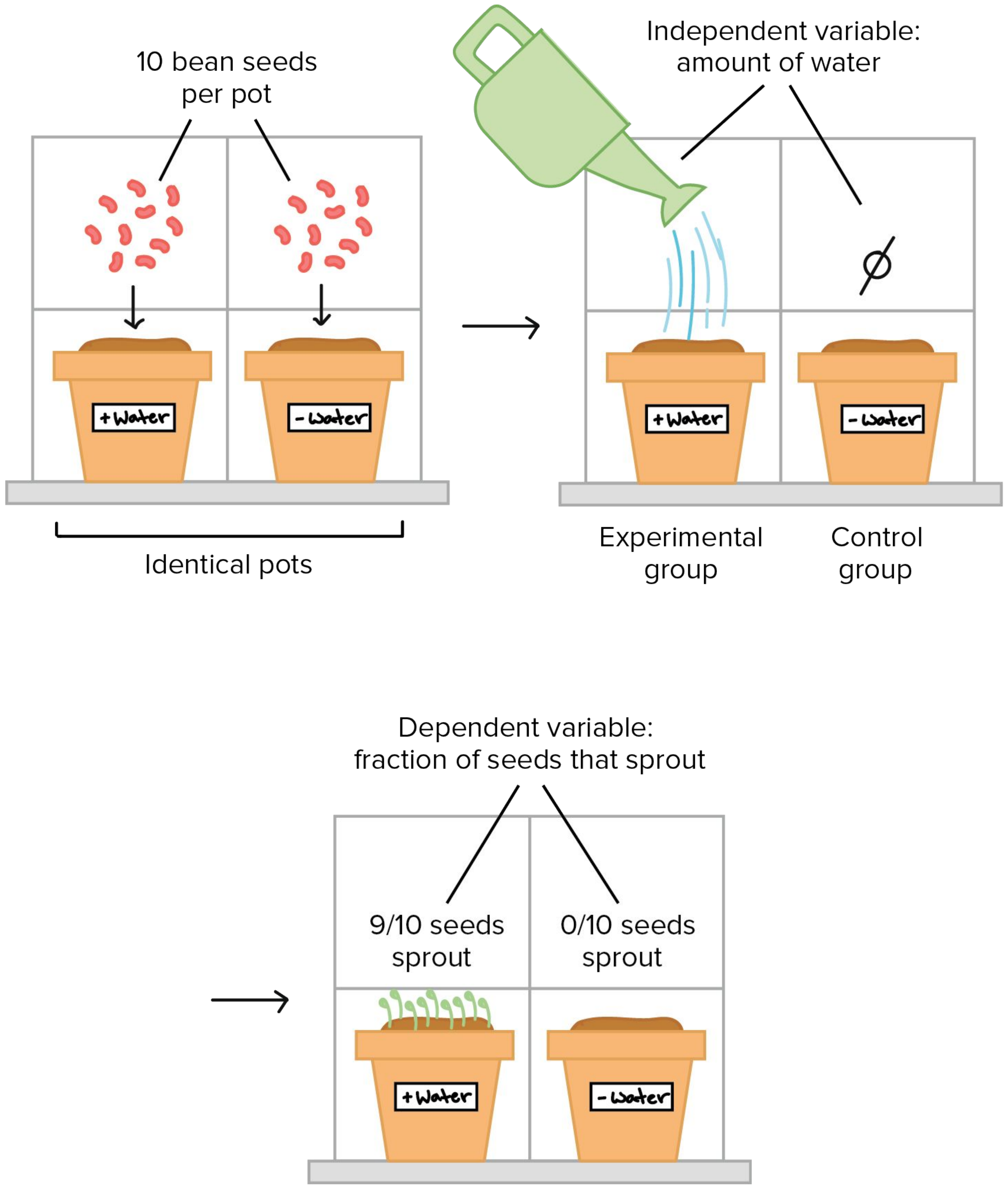
Randomized Controlled Experiment



Randomized Controlled Experiment



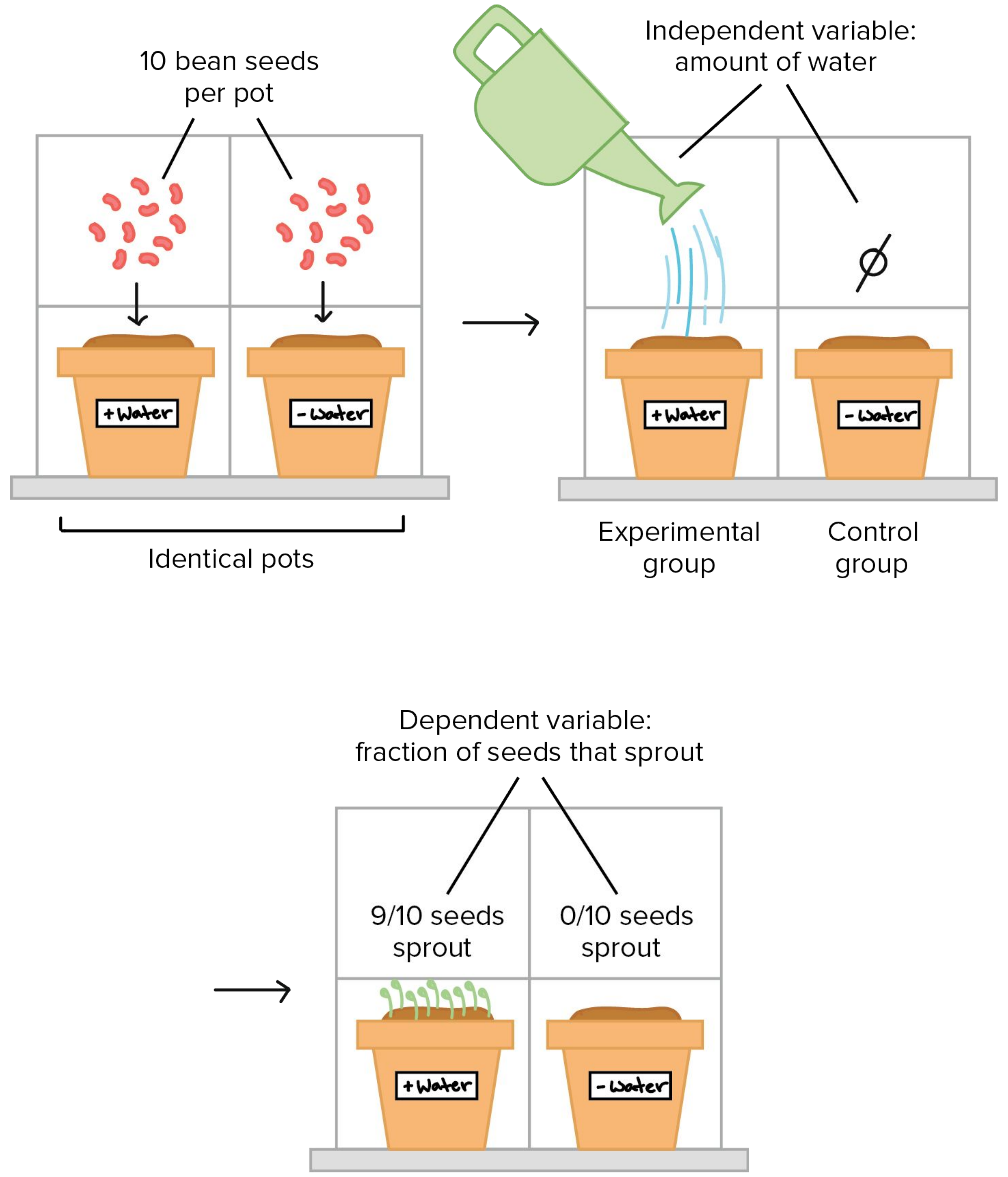
Randomized Controlled Experiment



Randomized Controlled Experiment

An experiment where all subjects involved in the experiment are treated the same except for one deviation.

One variable is changed in order to isolate the results.



A/B Testing

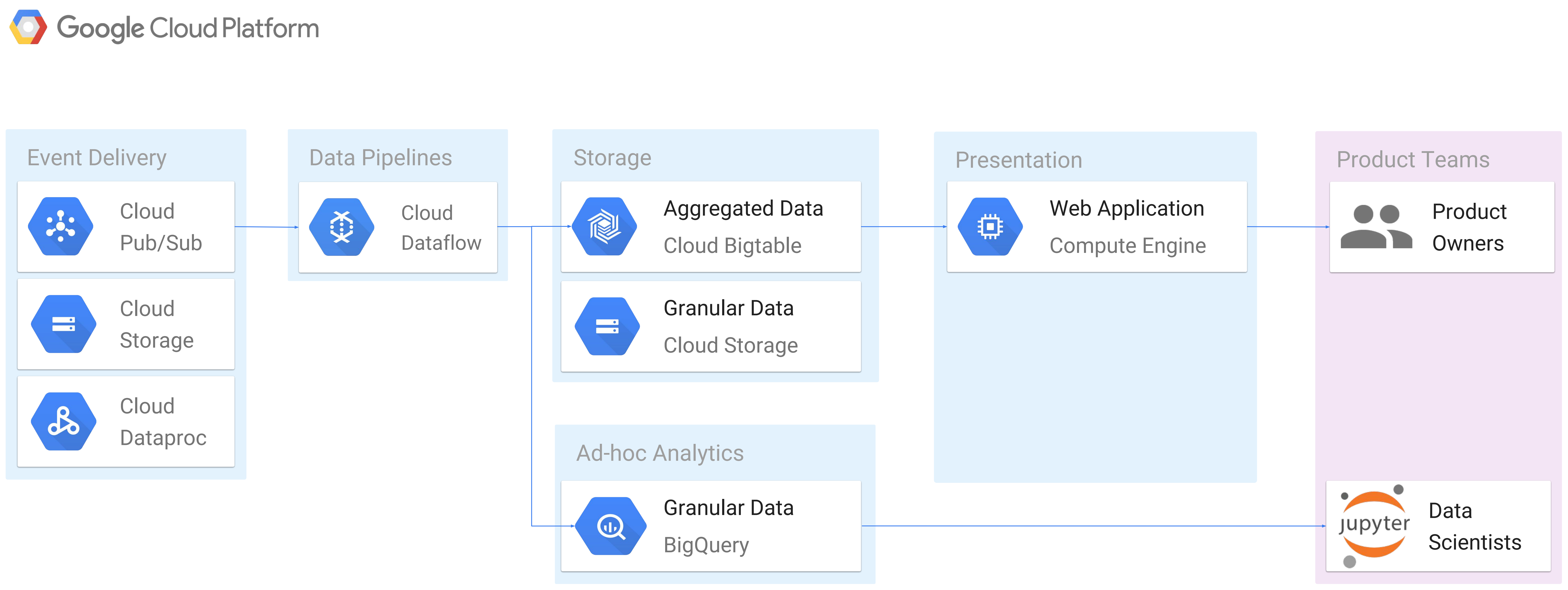
A/B Testing is a randomized controlled experiment where one variable is tested.

E.g., hypothesis *Our new recommendation algorithm increases content consumption.*

How to verify?

1. Formulate hypothesis
2. Run A/B test
3. See if there is a statistically significant increase in consumption.

Metrics for Experimentation Platform v1



Metrics for Experimentation Platform v1

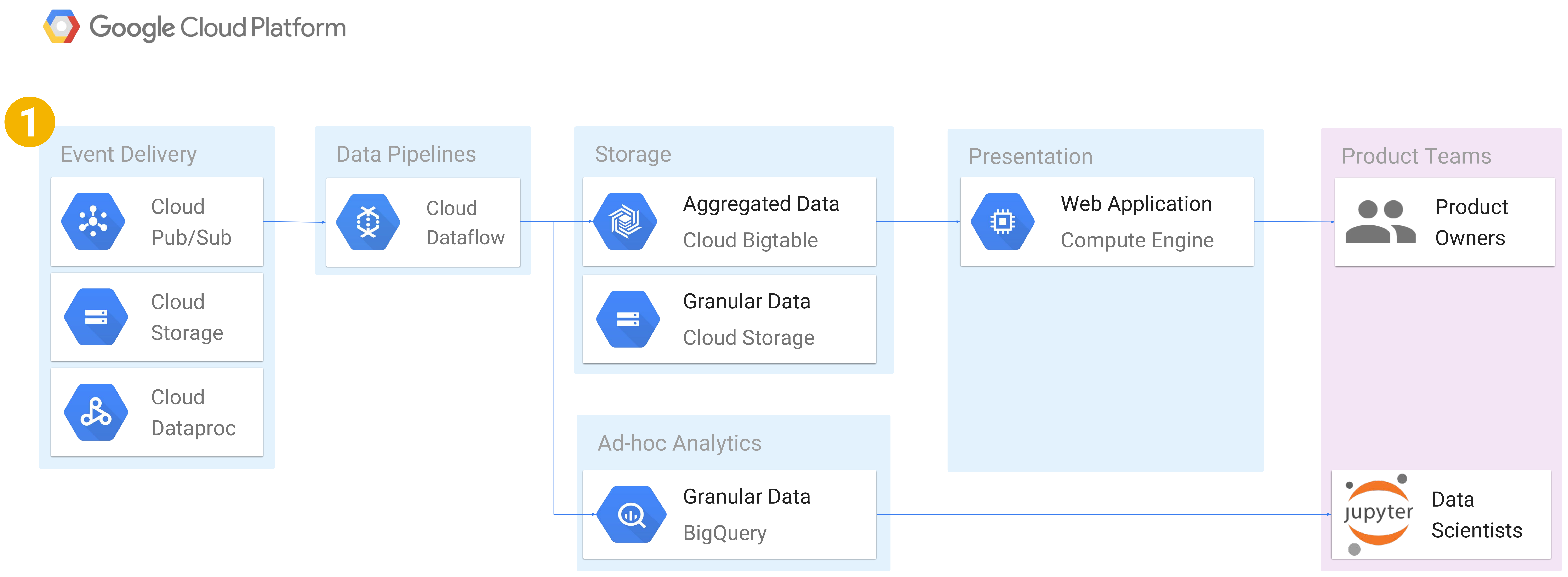
1. Event Delivery

Developers instrument applications and services using SDK.

Events are collected and published to Pub/Sub.

Batch jobs read data from Pub/Sub, deduplicate and anonymize, and then store in hourly partitions on GCS.

Exposing users to experiments, and configuring A/B variations on clients is done by dedicates services.



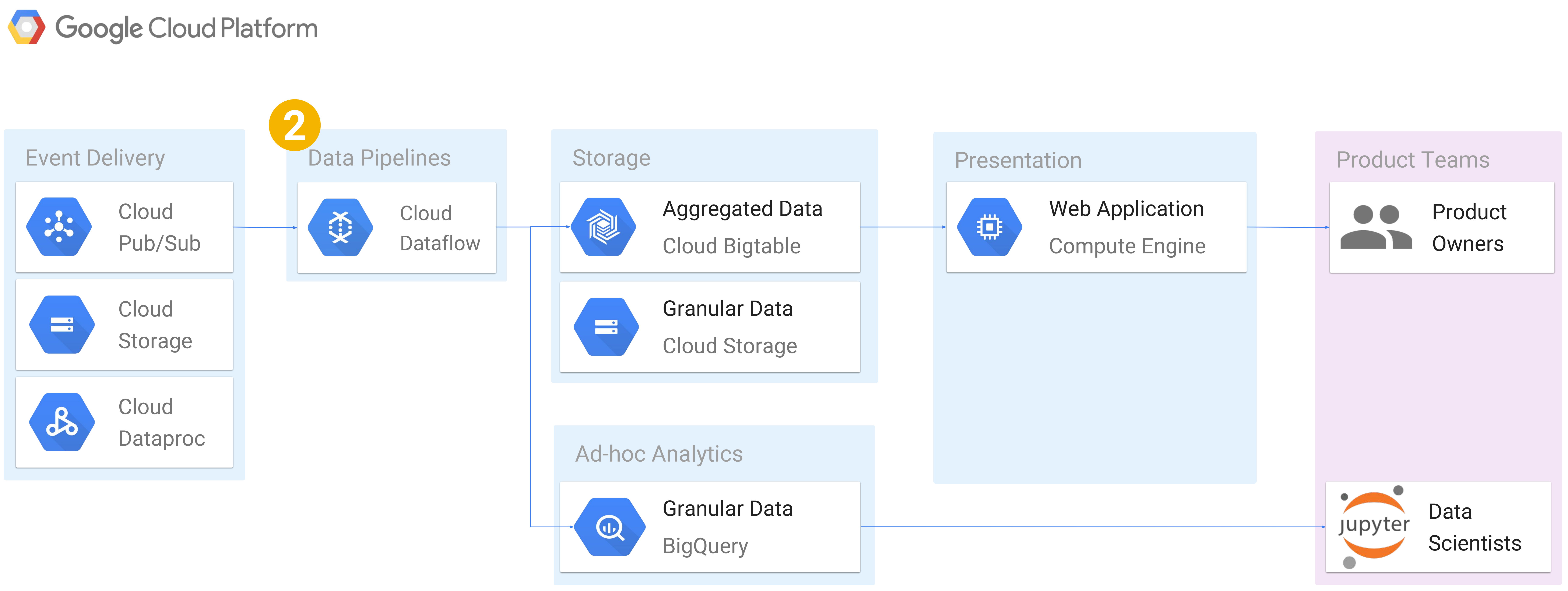
Metrics for Experimentation Platform v1

2. Data Pipelines and Storage

Data gets transformed and aggregated using Dataflow batch jobs, and stored in Bigtable, GCS and BigQuery.

Bigtable contains pre-computed aggregated experiment results.

BigQuery has granular data used in ad-hoc analysis.

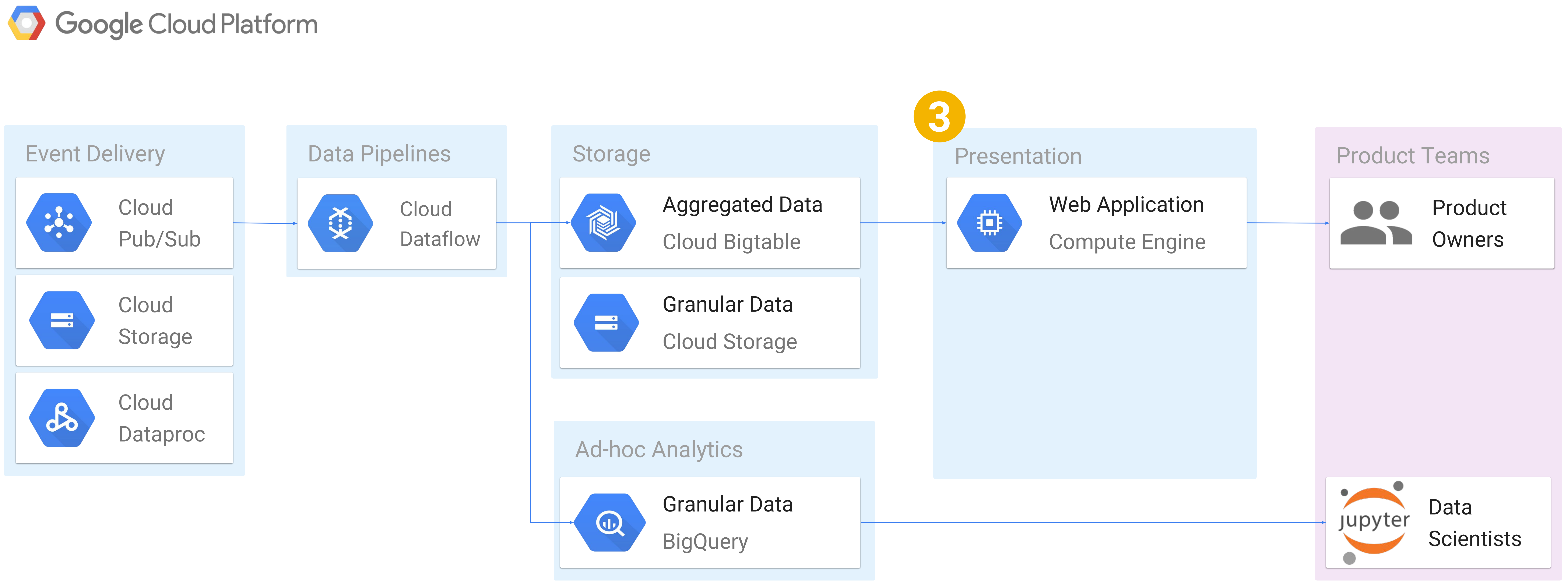


Metrics for Experimentation Platform v1

3. Presentation

Users of Experimentation platform see their experiment results in web application.

Statistical tests and health checks are performed automatically.

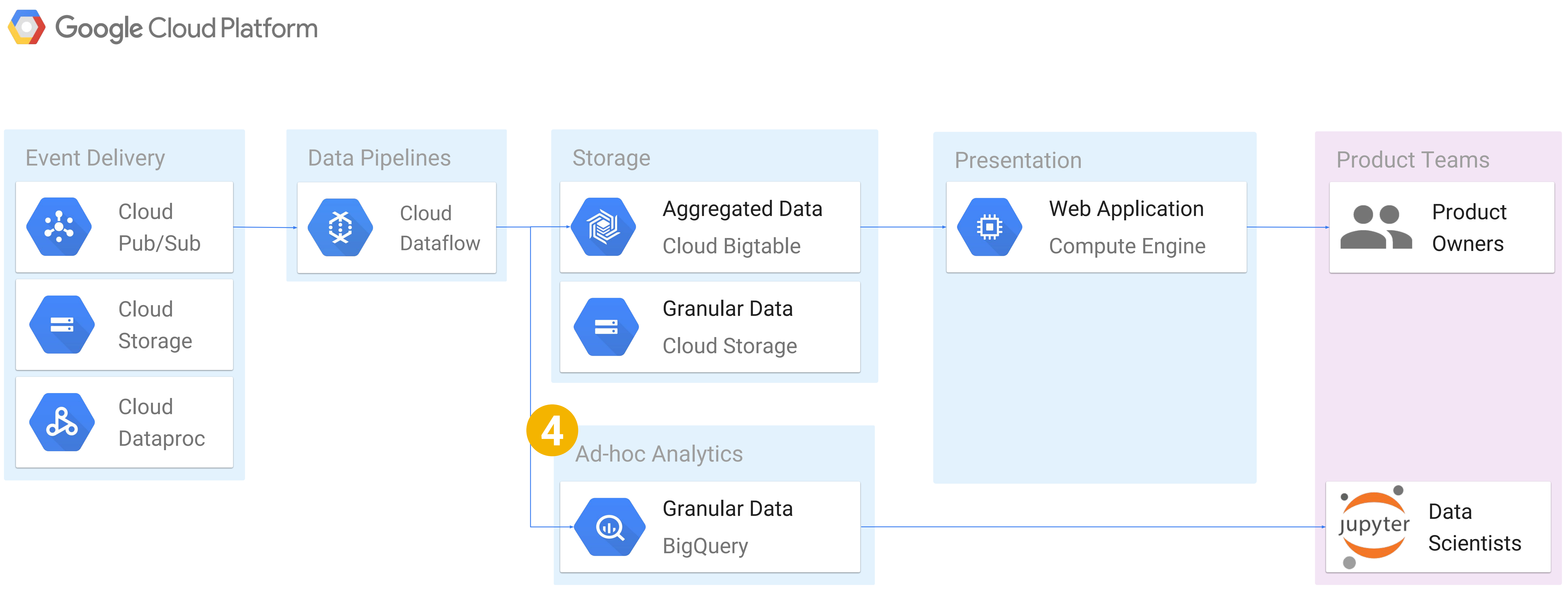


Metrics for Experimentation Platform v1

4. Ad-hoc Analytics

Data scientists do ad-hoc exploration in Jupyter notebooks using BigQuery.

Here they answer experiment specific-questions, not automatically supported by experimentation system.



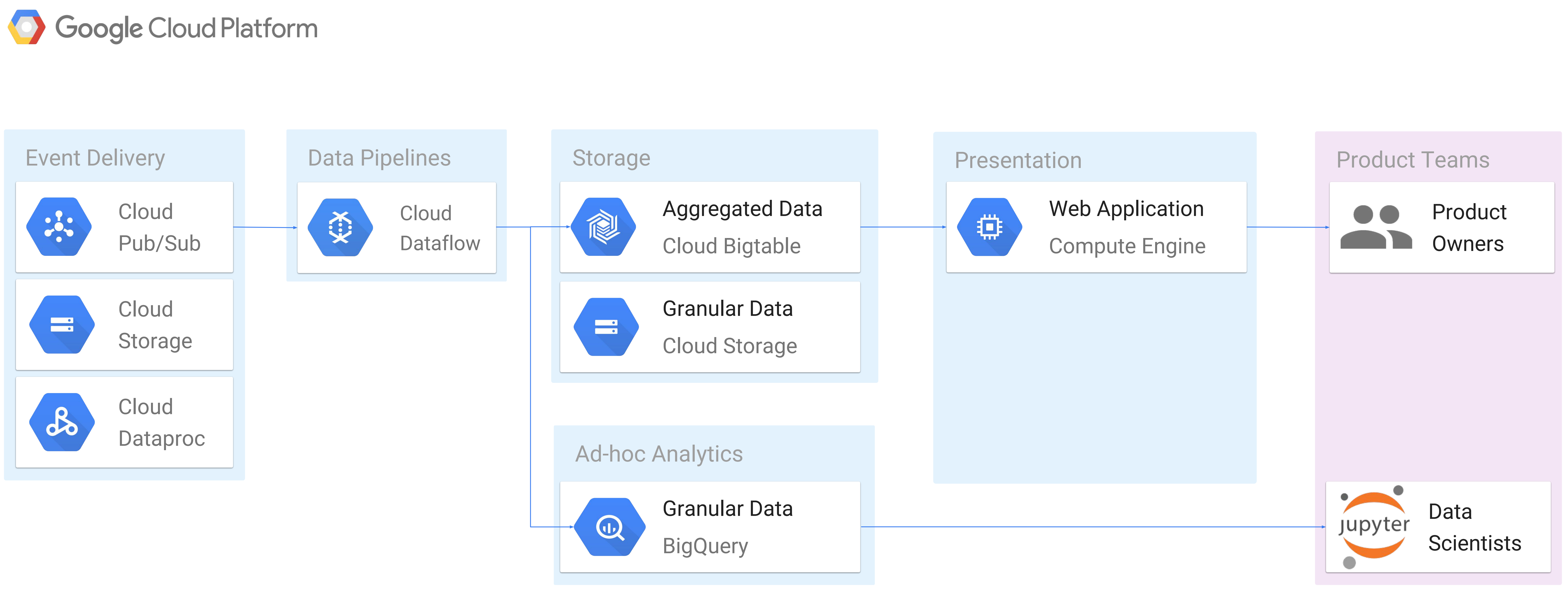
Metrics for Experimentation Platform v1

What works well

Centralized team owning 100-s of core metrics.

Automatic experiment analysis and planning.

Allows to conclude experiments without manual analysis. Autonomous feature teams can move fast and iterate on their product.



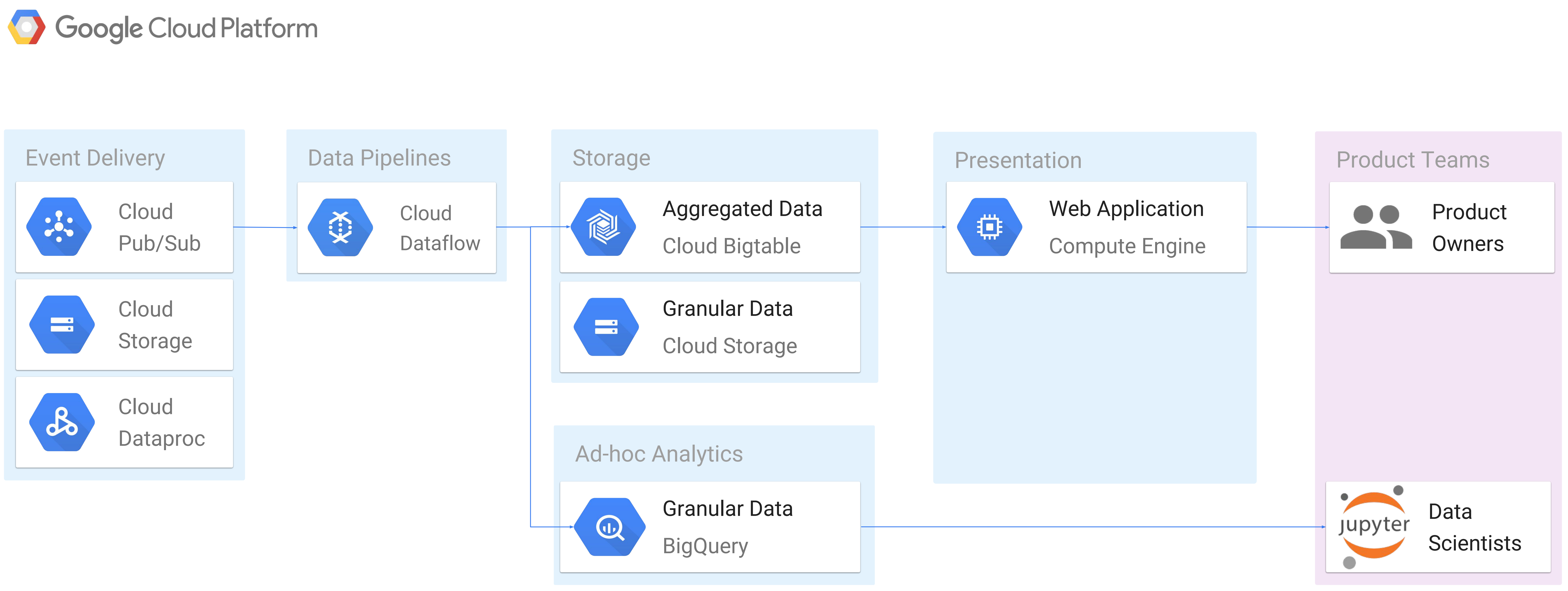
Metrics for Experimentation Platform v1

Problems

Not every metric worths centralization.

Centralized team became a bottleneck for Feature features.

As a result, too much repetitive work goes into notebooks and ad-hoc queries.



Metrics for Experimentation Platform v1

Reasons

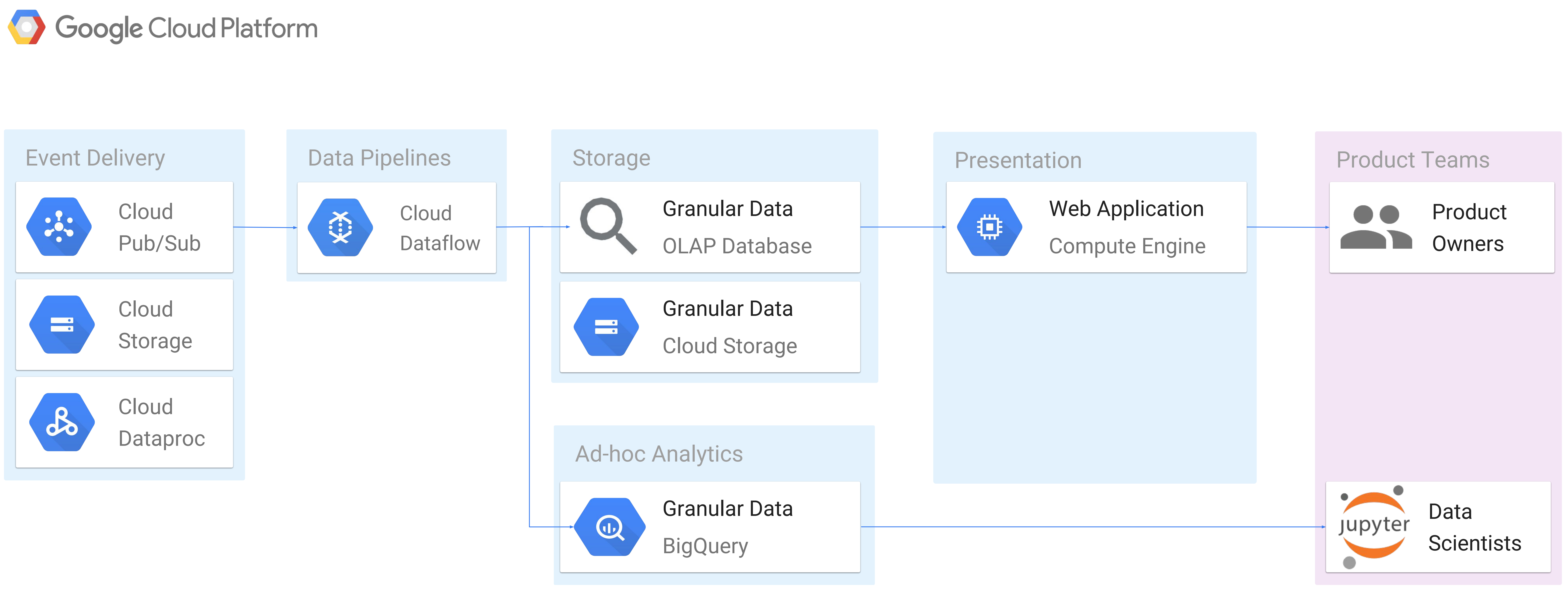
1. Experimentation isn't only about hypothesis testing, but learning from experiments.

Aggregated data in Bigtable wasn't granular enough, and didn't have enough dimensions.

2. Can't add a new metric without involving a central team.

What we want

Provide teams more granular data out of the box, and give a way to define a new metric.



Requirements

- Serve 100-s of QPS with sub-second latency
- We know in advance what are queries and data
- Maintain 10x metrics with the same cost
- Thousands of metrics
- Billions of rows per day in each of 100-s of tables
- Ready to be used out of the box
- Leverage existing infrastructure as much as feasible
- Hide unnecessary complexity from internal users

What about BigQuery?

- Supports Standard SQL
- Don’t have to optimize datasets in advance
- Works great for heavy queries with joins among multiple datasets
- Doesn’t need operations and machines running
- Good for interactive ad-hoc queries (~ minutes)
- Isn’t best for a high amount of low-latency queries you are aware in advance

Why ClickHouse?

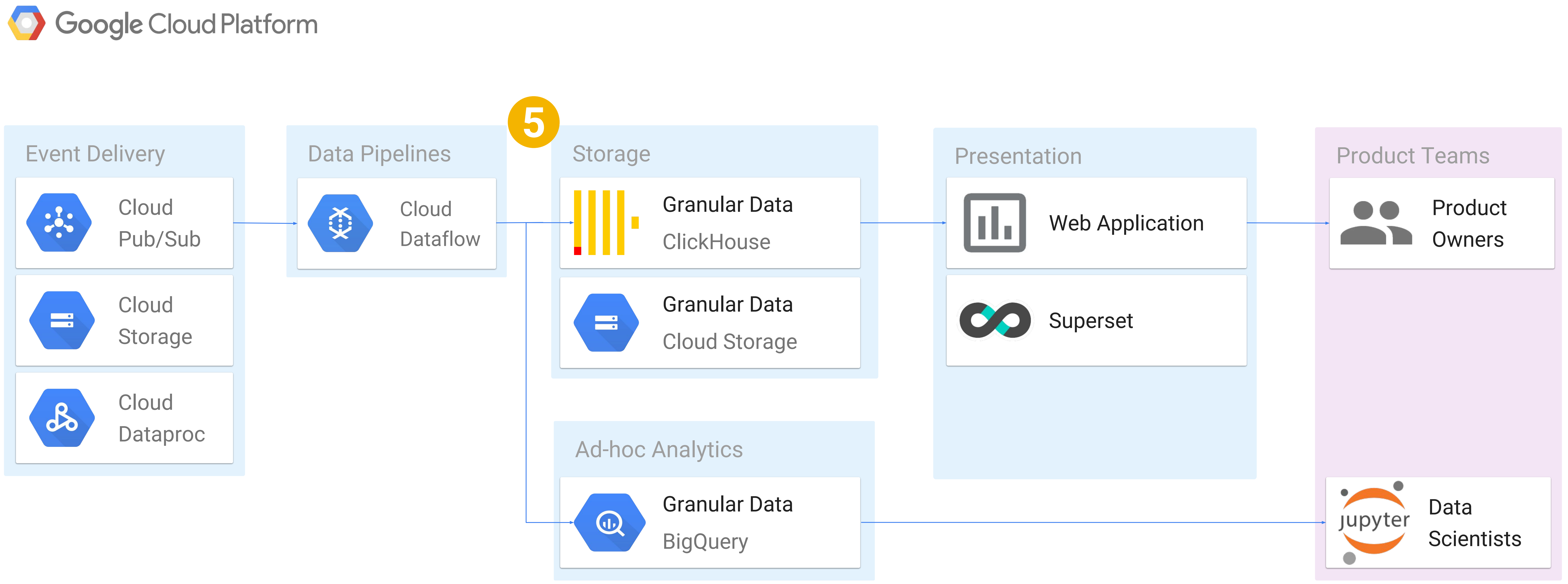
- Build proof of concept using various OLAP storages (ClickHouse, Druid, Pinot, ...)
- ClickHouse has the most simple architecture
- Powerful SQL dialect close to Standard SQL
- A comprehensive set of built-in functions and aggregators
- Was ready to be used out of the box
- Superset integration is great
- Easy to query using clickhouse-jdbc and jooq

Metrics for Experimentation Platform v2

5. ClickHouse

Interactive queries on granular data.

Reduce demand in notebooks and BigQuery with dashboards and exploration in Superset.

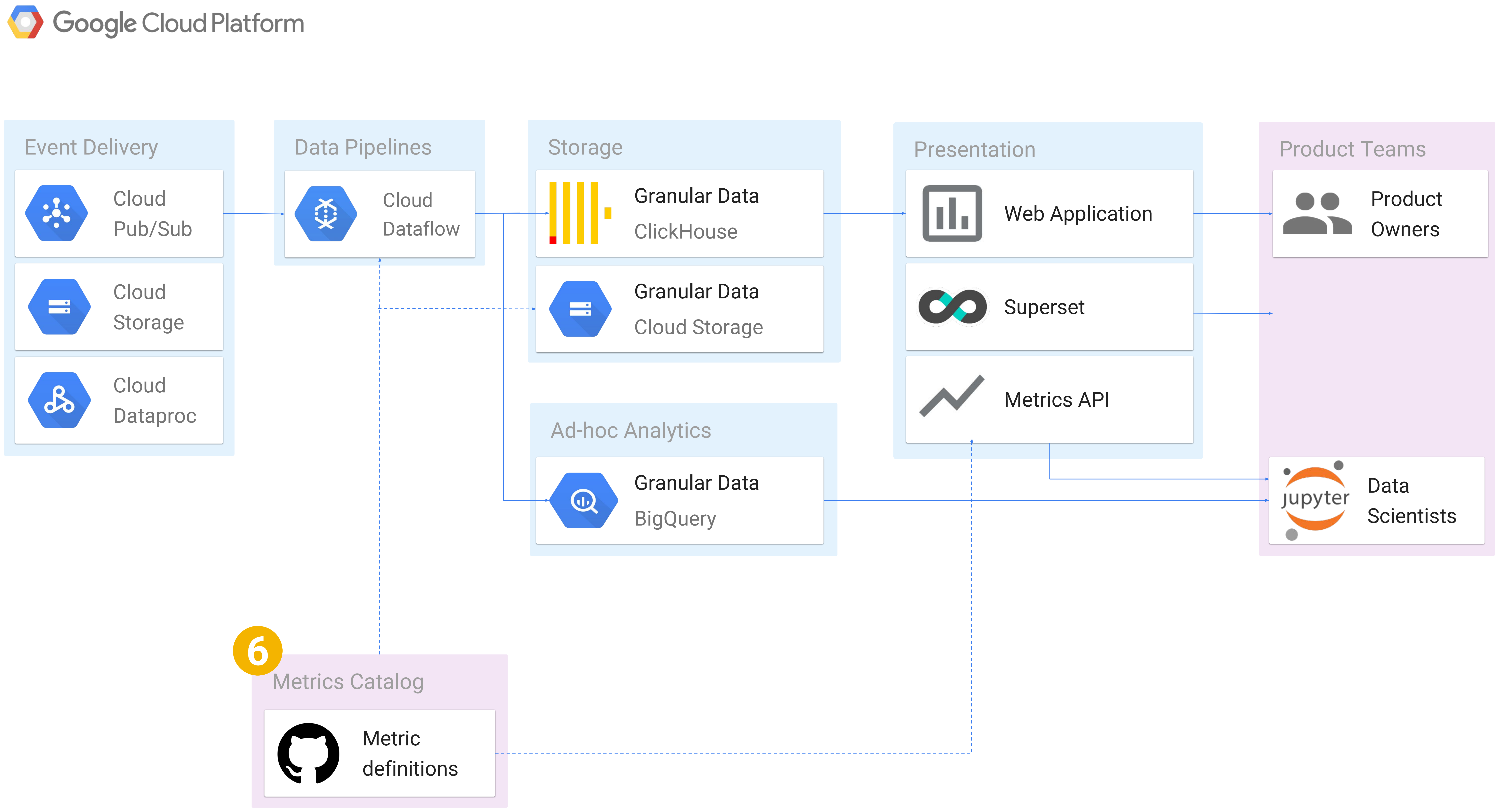


Metrics for Experimentation Platform v2

6. Metrics Catalog

Centralized place for teams to define their own metrics.

20 minutes to define a metric.



What we have built

- Own DSL to define metrics, and centralized metrics catalog
- Expressive and simple model that we can efficiently scale to 1000-s of metrics

- Generalize existing components to work with Metrics DSL
  - data preparation and ingestion into ClickHouse
  - denormalization with conformed dimensions
  - create dashboards, tables and charts in Superset
  - do statistical tests, and expose results through API
  - define ownership, tiering, and other attributes
  - integrates with the rest of infrastructure for alerting, monitoring, data quality, anomaly detection, access control & etc

- Users don’t work with ClickHouse SQL, or need to know how it works
- API to query metrics and metadata

Ingestion to ClickHouse

- Move data from GCS to ClickHouse
- Use clickhouse-jdbc, custom code and RowBinary format
- Use daily partitioning, and ingest once a day
- 1 hour to ingest 5 TiB on test cluster using 9 n1-standard-32 with 8 NVMe SSD RAID0
- Don’t use materialized views in ClickHouse
- Offload most of computations to batch data pipelines due to scalability, experience and tooling
- **TODO** try ClickHouse-Native-JDBC
- **TODO** pre-sort in data pipelines before ingesting

What is next

- Do lambda-style ingestion for subset of metrics with low-latency requirements
- Add more aggregations to DSL (e.g. 5 statistical moments)
- Add custom chart types to Superset
- Try ClickHouse for similar use cases within Spotify

Spotify®