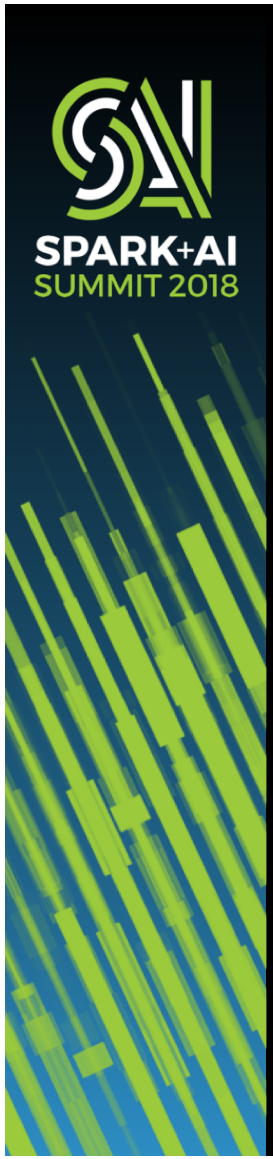
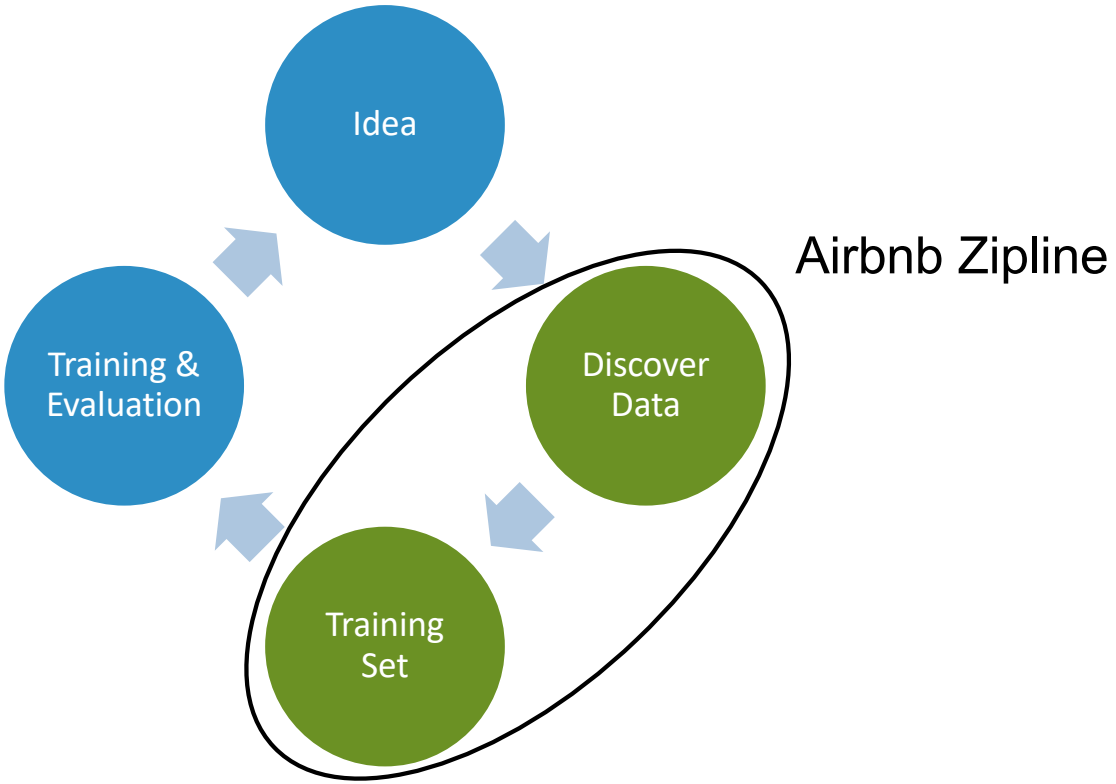


Zipline – Airbnb's ML Data Management Framework

Nikhil Simha, Airbnb
Andrew Hoh, Airbnb

#ML3SAIS

Feature Engineering



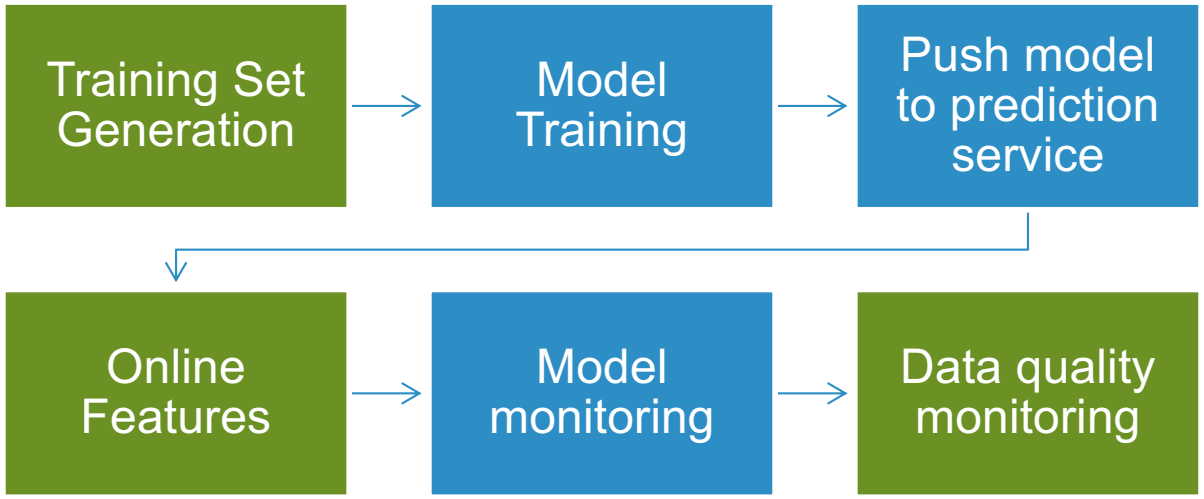
Discovering data

- Where is it?
- Is it good?
- Will it continue to be good?
- Has it already been done before?

Training set

- What processing engine to use?
- Backfilling
- Point-in-time correct windows

Production


Training Set Generation
Model Training
Push model to prediction service
Online Features
Model monitoring
Data quality monitoring

What is Zipline?

- Part of Bighead – Airbnb's e2e ml platform
- Training Sets
- Feature Store
- Data quality

Concepts

- Data Source
- Feature-set
- Training-set
- Client

Feature-Set

- Source queries
  - Hive tables
  - Event streams
  - Mutation streams
- Primary Keys
- Time Stamp

Feature-Set – Sources

- Multiple sources
  - Backfills
  - Migrations

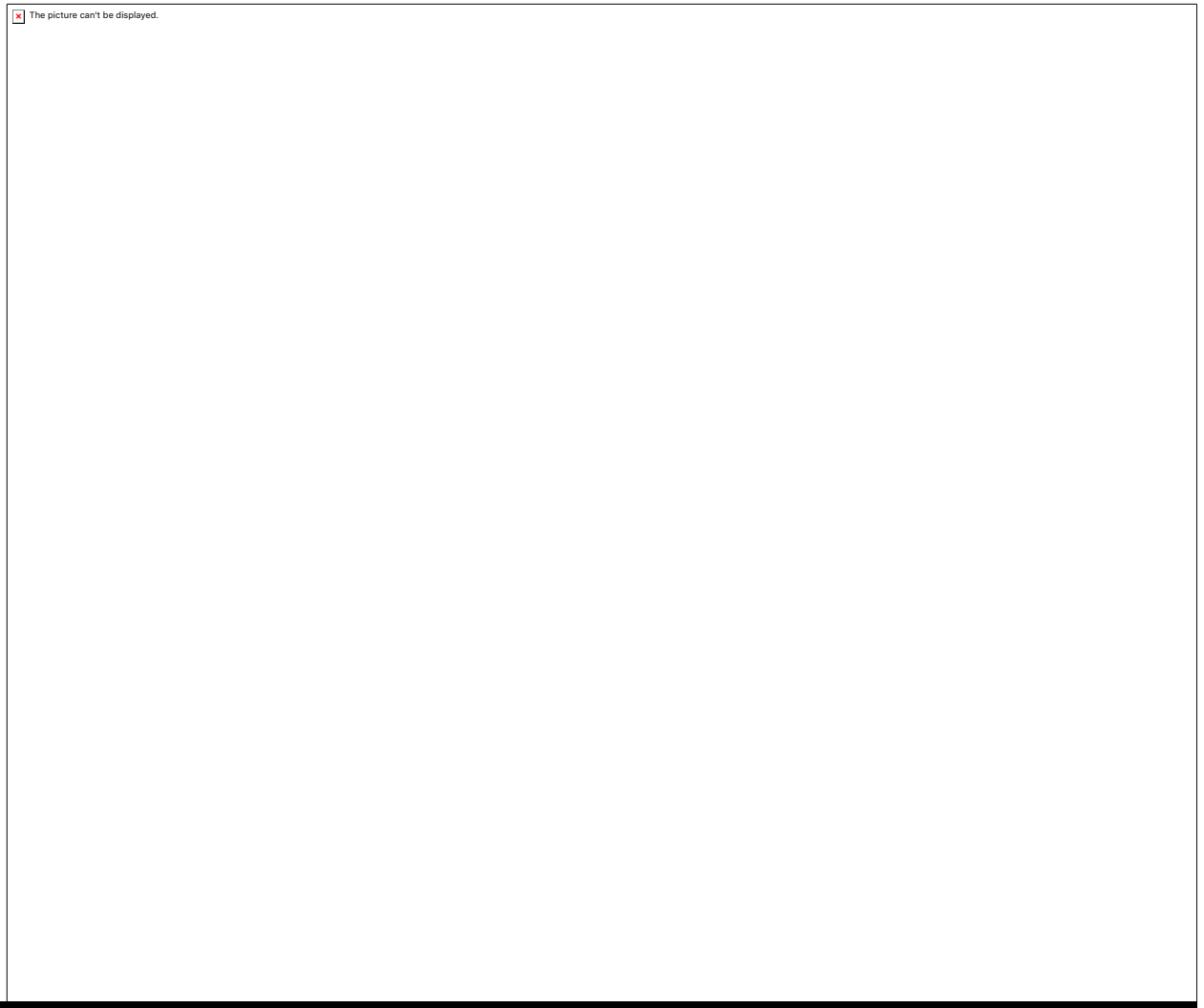
Example/Operations

```
owner: guest_growth

source: {
  type: hive
  query: """
    SELECT
        1                                 AS bookings
      , IF(dim_instant_booked = 1, 1, 0)  AS ib_bookings
      , id_guest                          AS user
      , ts
    FROM core_data.fct_bookings
    WHERE
      ds BETWEEN '{{ start_date }}' AND '{{ end_date }}'
  """
  dependencies: [core_data.fct_bookings]
}

features: {
  bookings: {
    doc: "Total bookings made by guest in various time windows"
    operation: sum
    windows: ["7d", "14d", "30d", "90d", "180d", "1y"]
  }
```

Data Quality



Training set

- Driver query
  - primary keys
  - timestamps
  - Point-in-time correct

Training set – example

```
from zipline.training_set import TrainingSet

features = [ "user_email_clicks_by_user_v1_count_7d",
             "user_totals_v1_total_nights_cancelled_guest",
             "user_v2_has_profile_photo" ]

query = """
    SELECT
          id_user as user,
          updated_at AS ts,
    FROM
        user_activity du
    WHERE du.ds BETWEEN '{{ start_date }}' AND '{{ end_date }}';
"""

query_dependencies = [ "user_activity" ]
team = 'host_growth'
name = 'host_referral'
start_date = '2018-01-01'

training_set = TrainingSet(name, query, query_dependencies, features, team, start_date)
```

Training set – example

FeatureSet 1

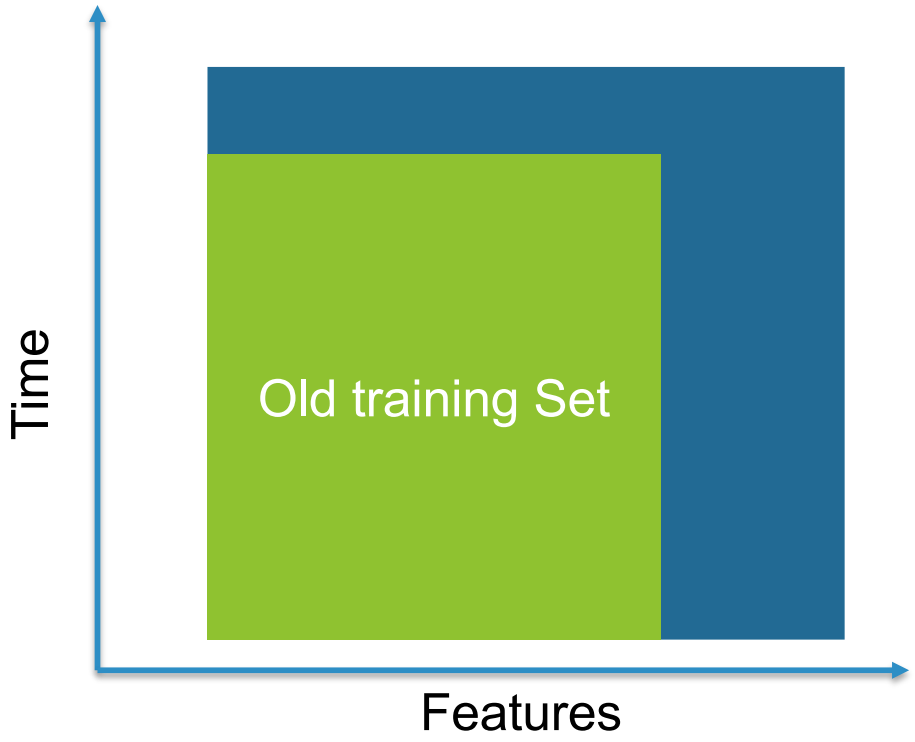
FeatureSet 2

PK1 = User ID	PK2 = Listing ID	Timestamp	bookings_by_user	bookings_by_listing
123	456	2018-01-01 23...	0	4
234	567	2018-01-04 01...	2	8
456	789	2018-01-02 08...	1	0

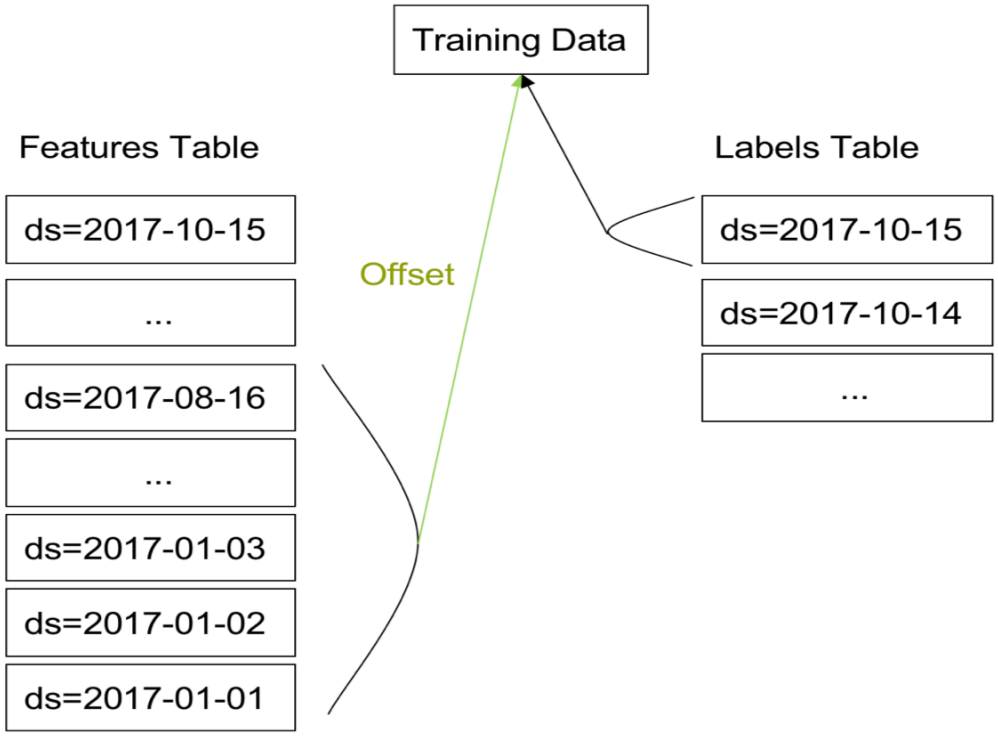
User provides: Primary keys and timestamps

Zipline computes feature values point-in-time correct for those PKs and those timestamps.

Backfills



Label offsets

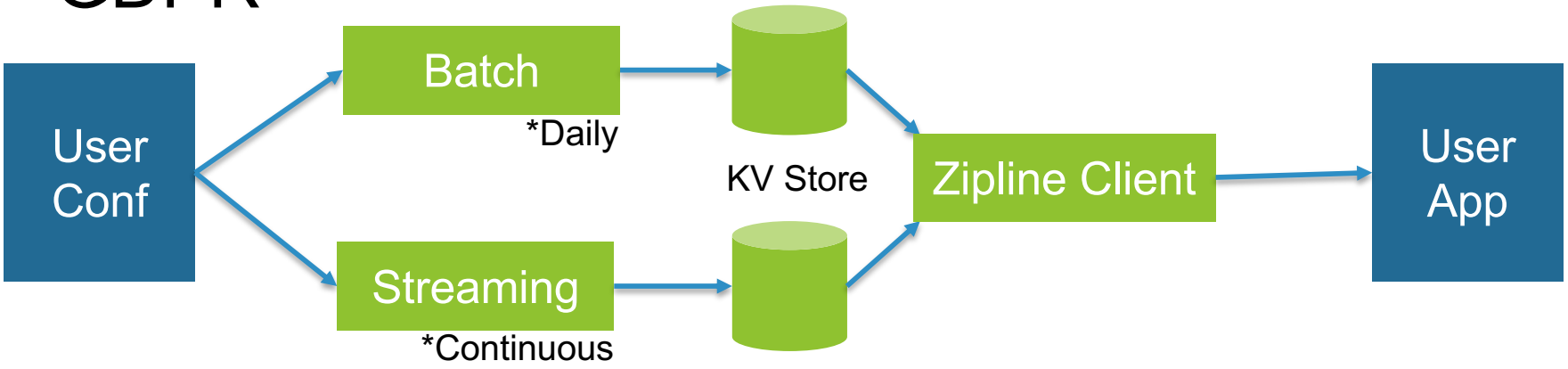


Online scoring

- Backfills
- Low latency
- DB Mutations
  - Batch correction

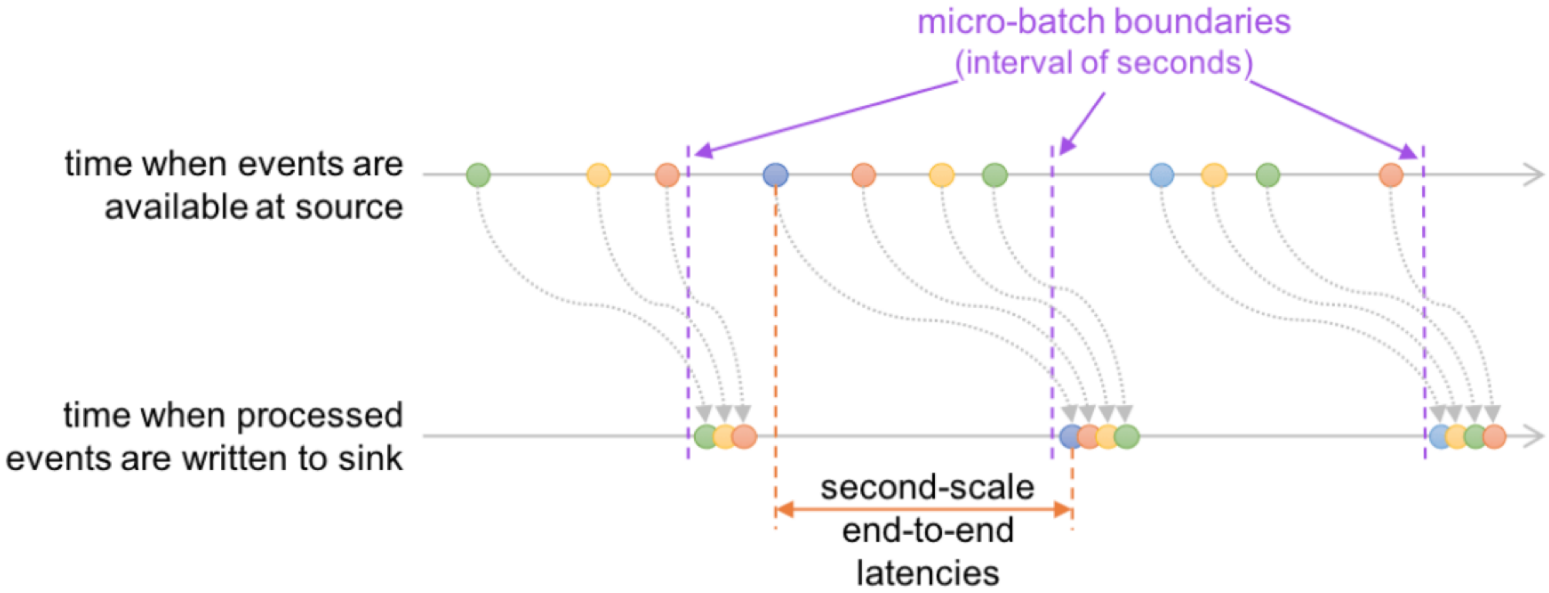
Lambda

- Batch – Spark
- Streaming – Flink
- GDPR



Why Flink?

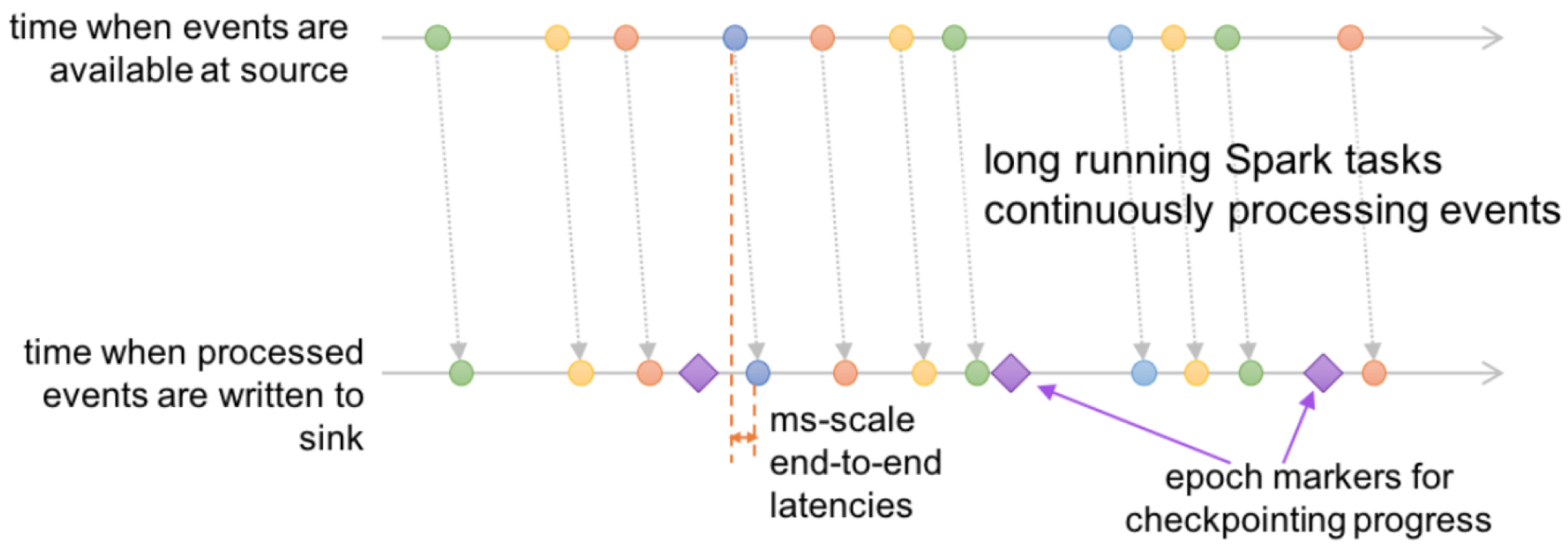
*Image from spark blog for continuous processing



Second-scale end-to-end latencies with Micro-batch Processing

Why Flink?

*Image from spark blog for continuous processing



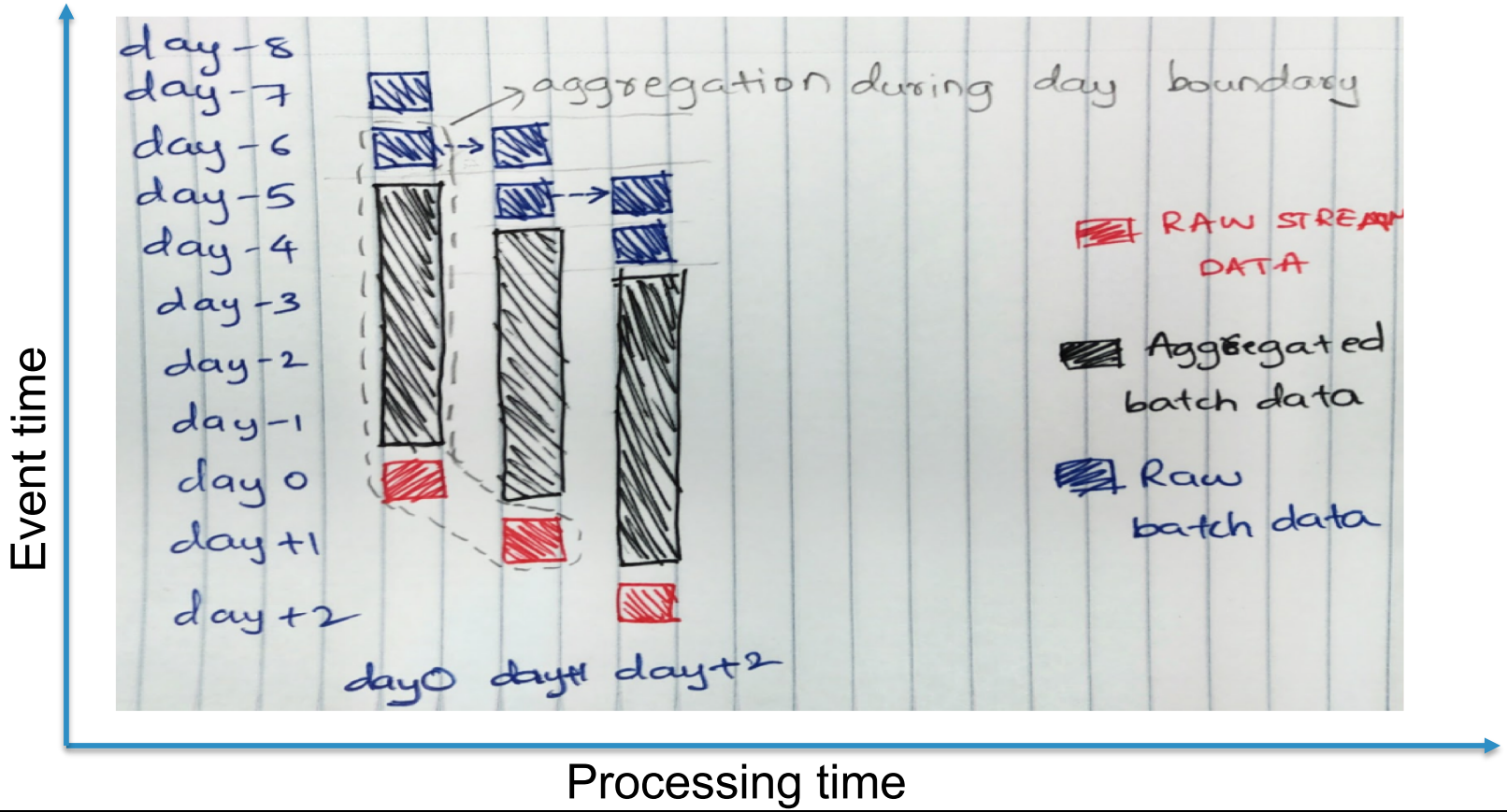
Millisecond-scale end-to-end latencies with Continuous Processing

Aggregations

- Across batch and streaming
- Fixed length sliding windows
- Raw Events at the tail and head

7 day sliding window

Availability



Client

- Aggregation
- Logic to handle availability
- Java
  - The API is (List of Feature Names) => Map of feature values – as objects.

Mutations

- Organizationally easy
- Consistent
- Not flexible
- More complex
  - Inverse

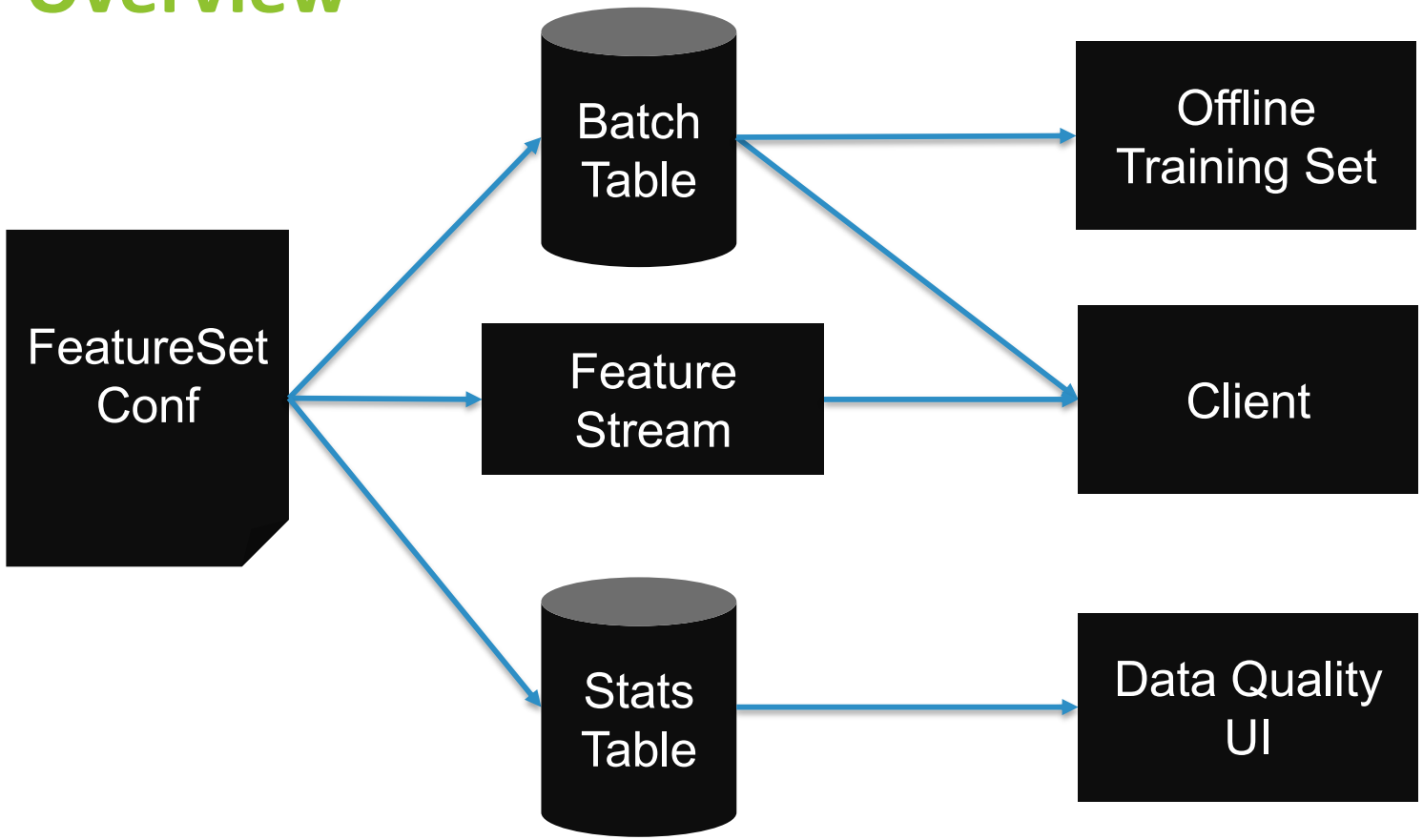
Mutations - Aggregation

- Monoids
  - Associative: (a + b) + c = a + (b + c)
  - Distribute aggregation
  - Sum, Avg, Count etc..
- Group
  - Invertible
  - Min, Max, Median, Ntile
  - Not possible without memory
  - Compromise

Mutations - Aggregation

- Deletes
  = Inverse(before)
- Updates
  = Inverse(before) + after

Overview


Batch Table
Offline Training Set
FeatureSet Conf
Feature Stream
Client
Stats Table
Data Quality UI

When?

- Plan to open source in Q3 2018.
- Will be part of Bighead
  - Talk tomorrow.

Questions?