runway

# Building a Data Foundation for Multimodal Foundation Models

Ethan Rosenthal

Data Council 2025


runway
Gen-4

# About

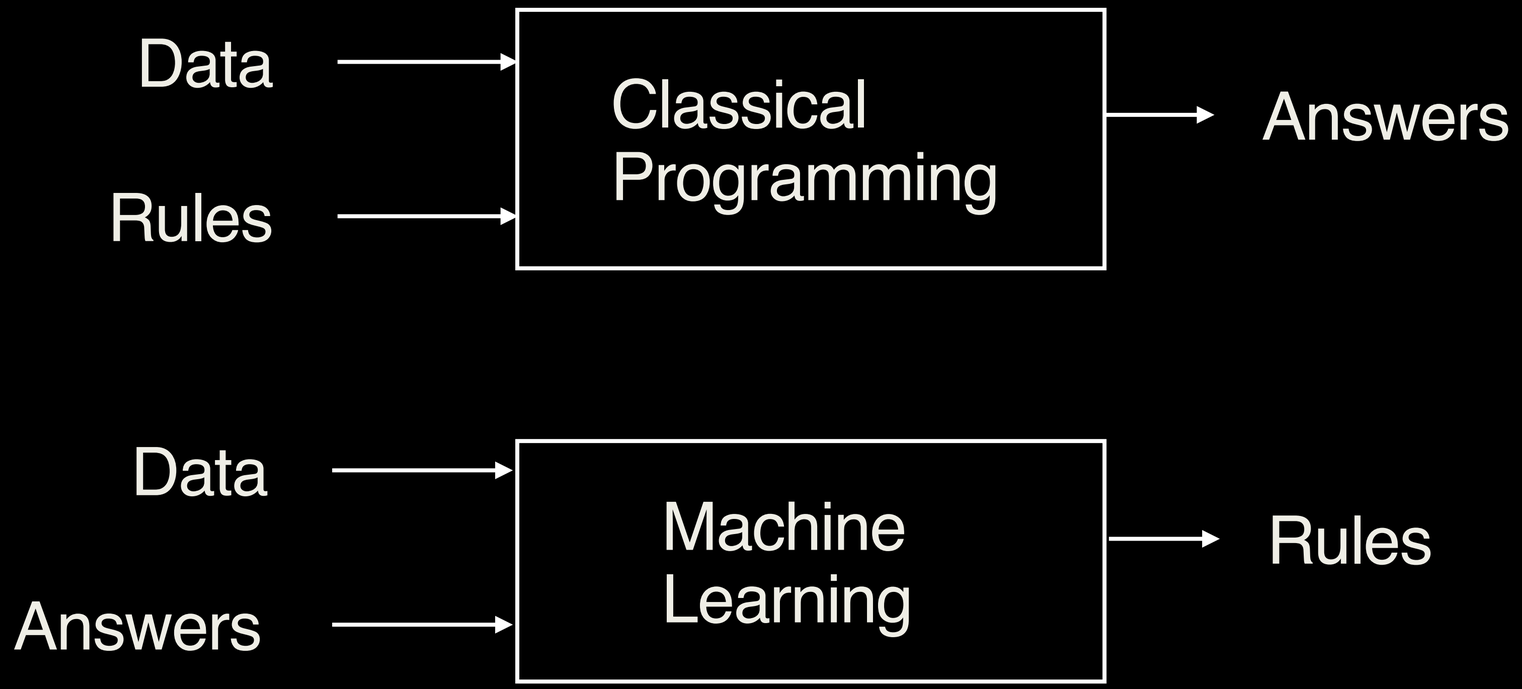
Currently: ML @ Runway
Formerly: LLMs @ Square; startups; physics PhD
Also: Adjunct Professor @ NYU Tandon

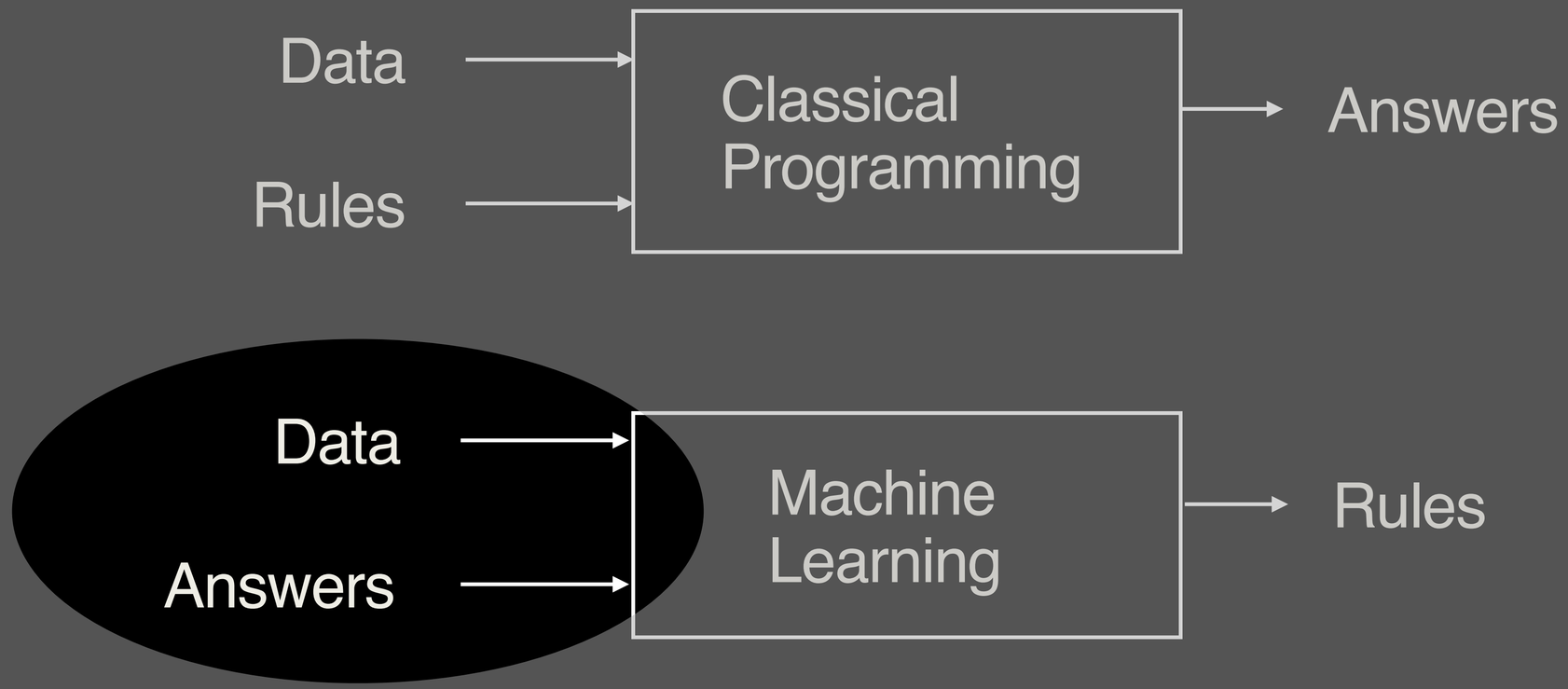
Find me


ethanrosenthal.com

@ethanrosenthal.com

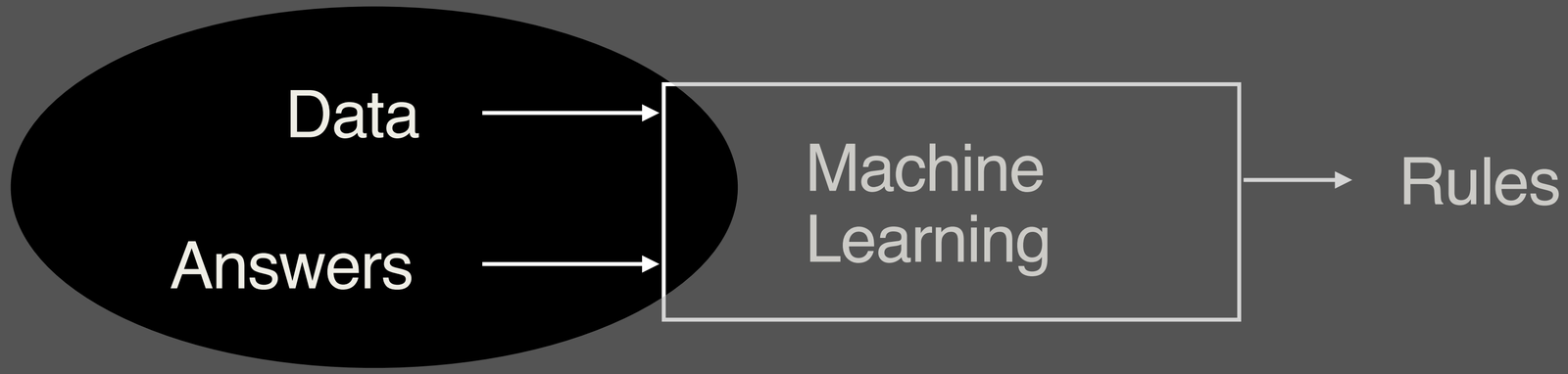
@EthanRosenthal

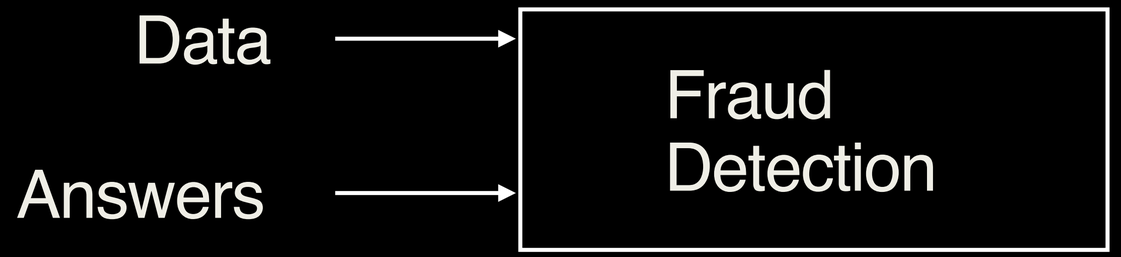

Data
Rules
Classical Programming
Answers
Data
Answers
Machine Learning
Rules


Data
Rules
Classical Programming
Answers
Data
Answers
Machine Learning
Rules

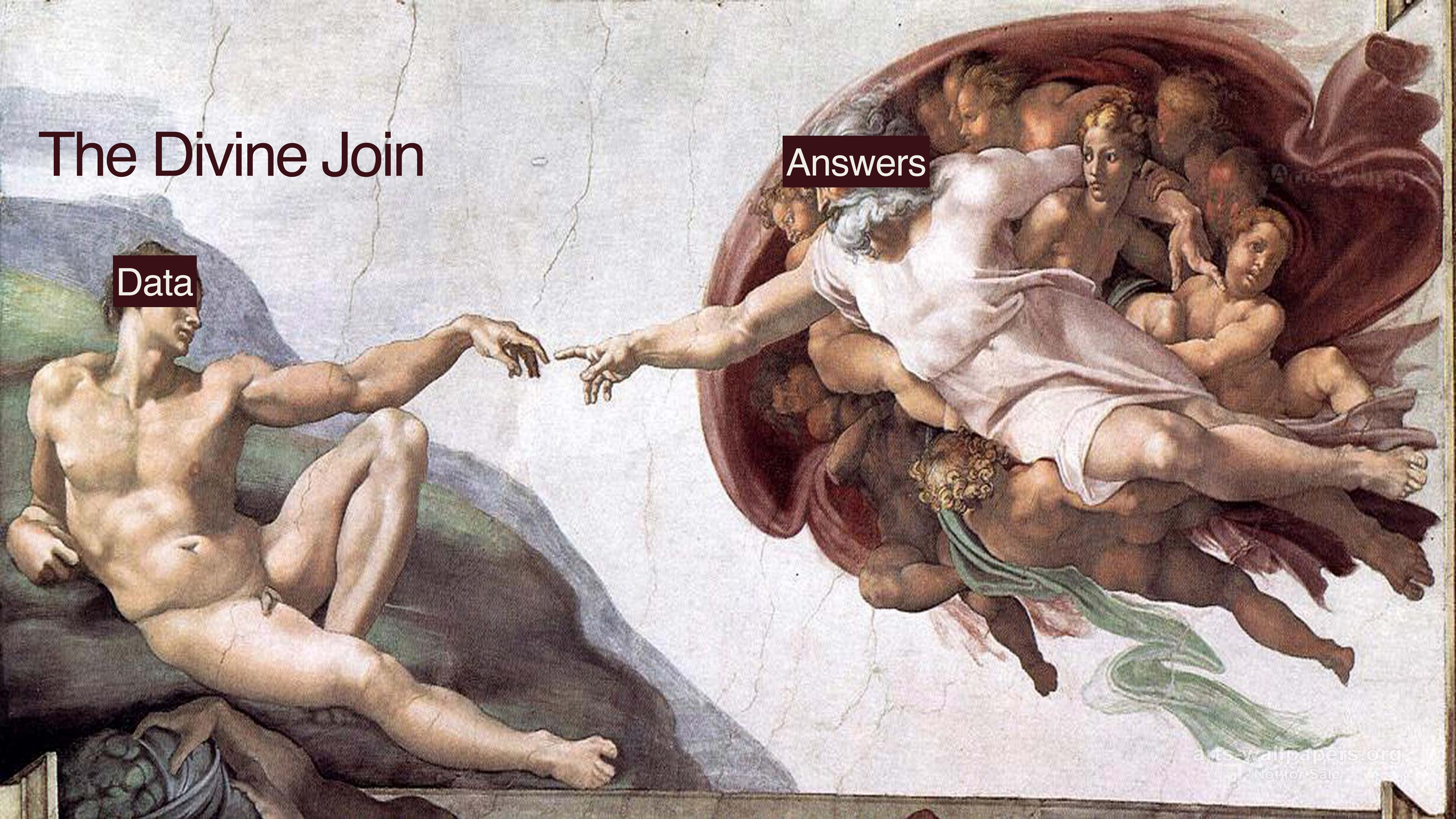
# “*Everyone wants to do the model work, not the data work*”: Data Cascades in High-Stakes AI

Nithya Sambasivan, Shivani Kapania, Hannah Highfill, Diana Akrong, Praveen Paritosh, Lora Aroyo
[nithyasamba,kapania,hhighfill,dakrong,pkp,loraa]@google.com
Google Research
Mountain View, CA

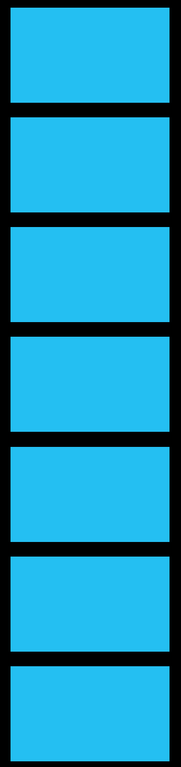



Data
Answers
Fraud
Detection

# The Divine Join



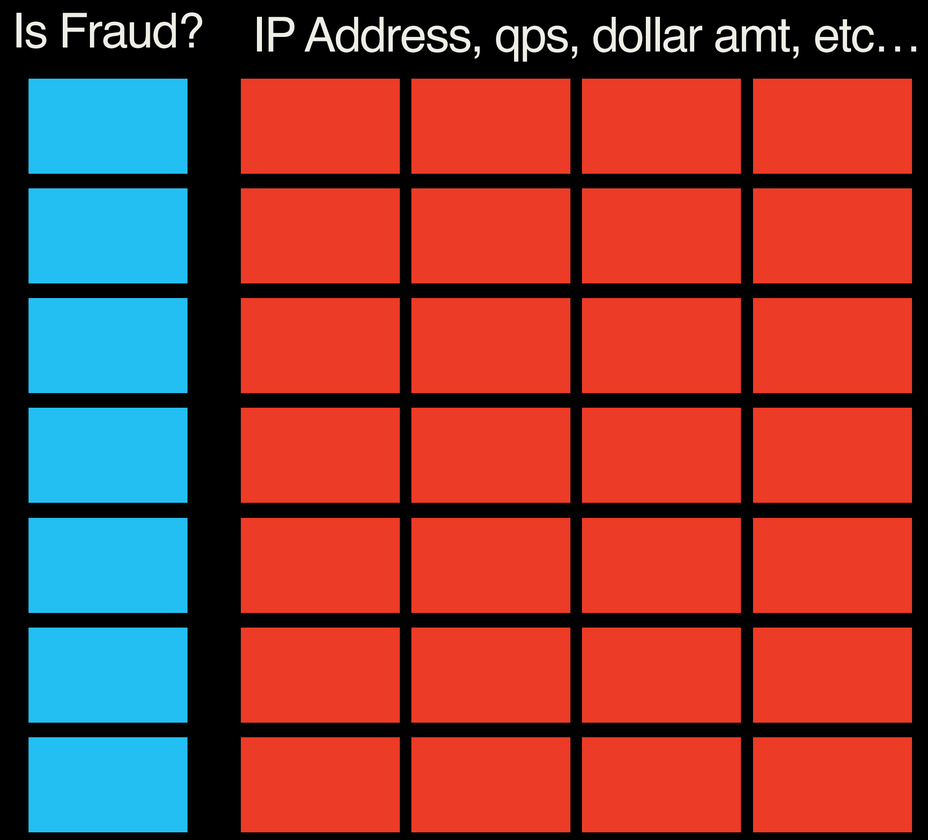
Is Fraud?



Data

Answers

Fraud Detection

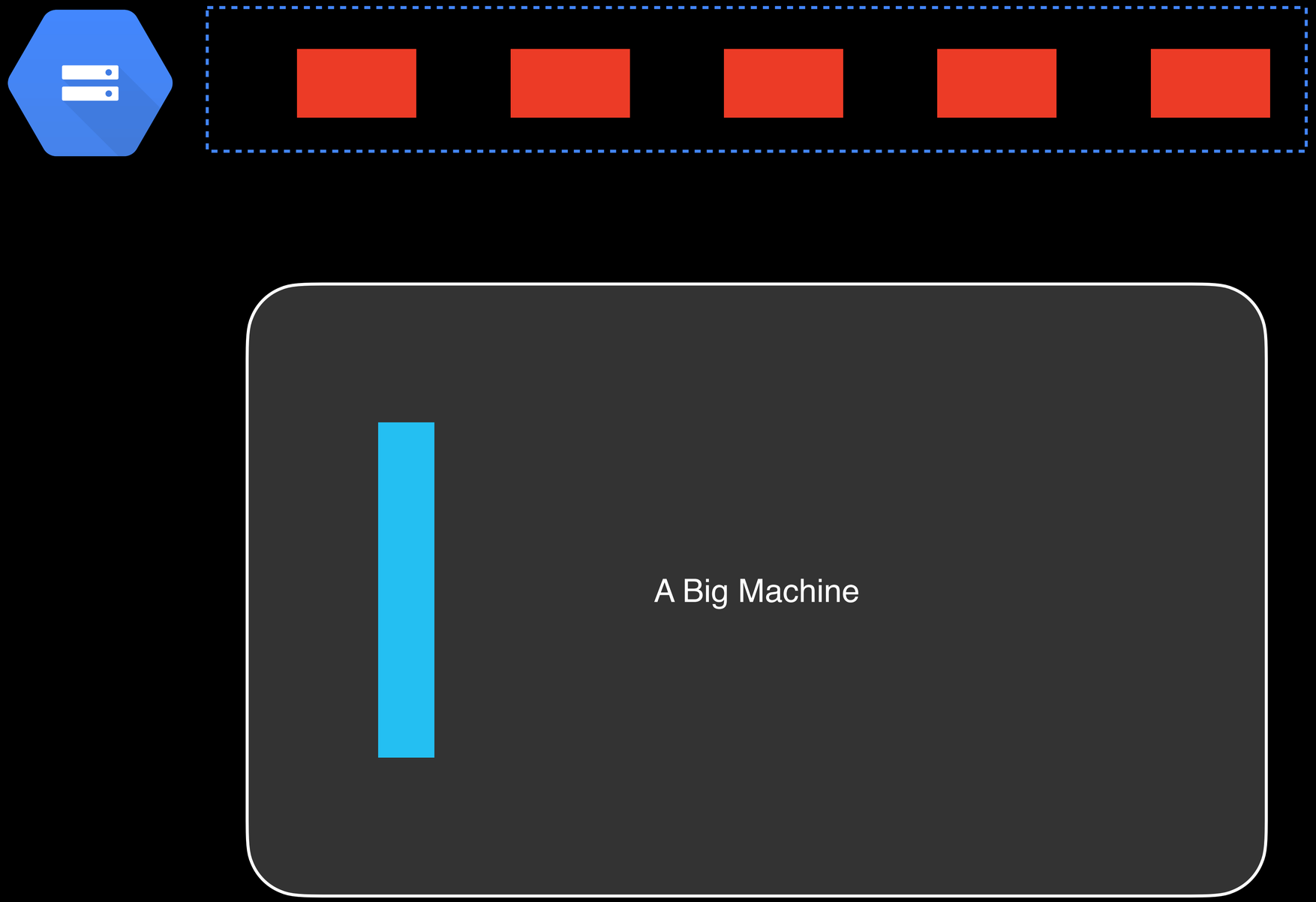


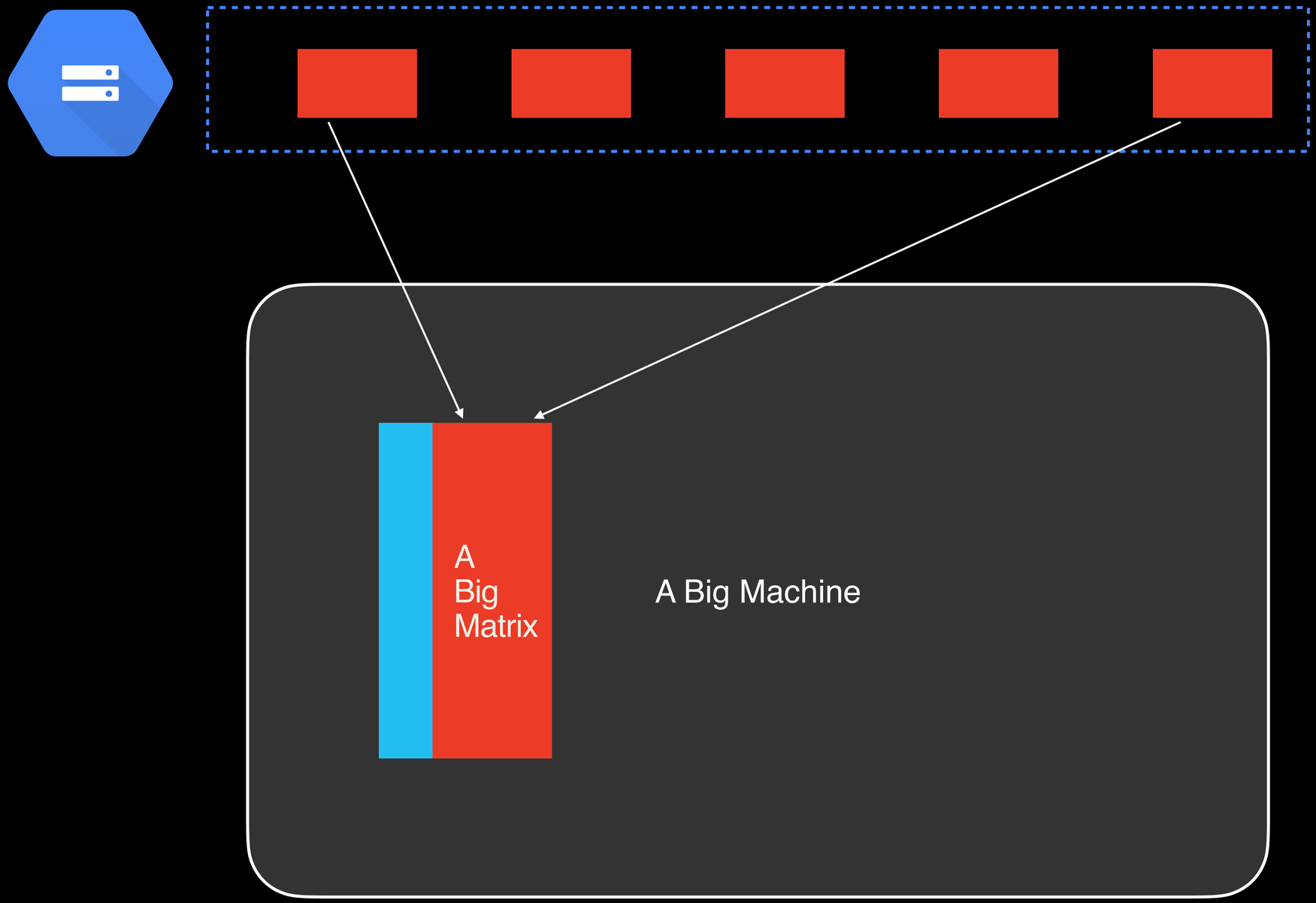


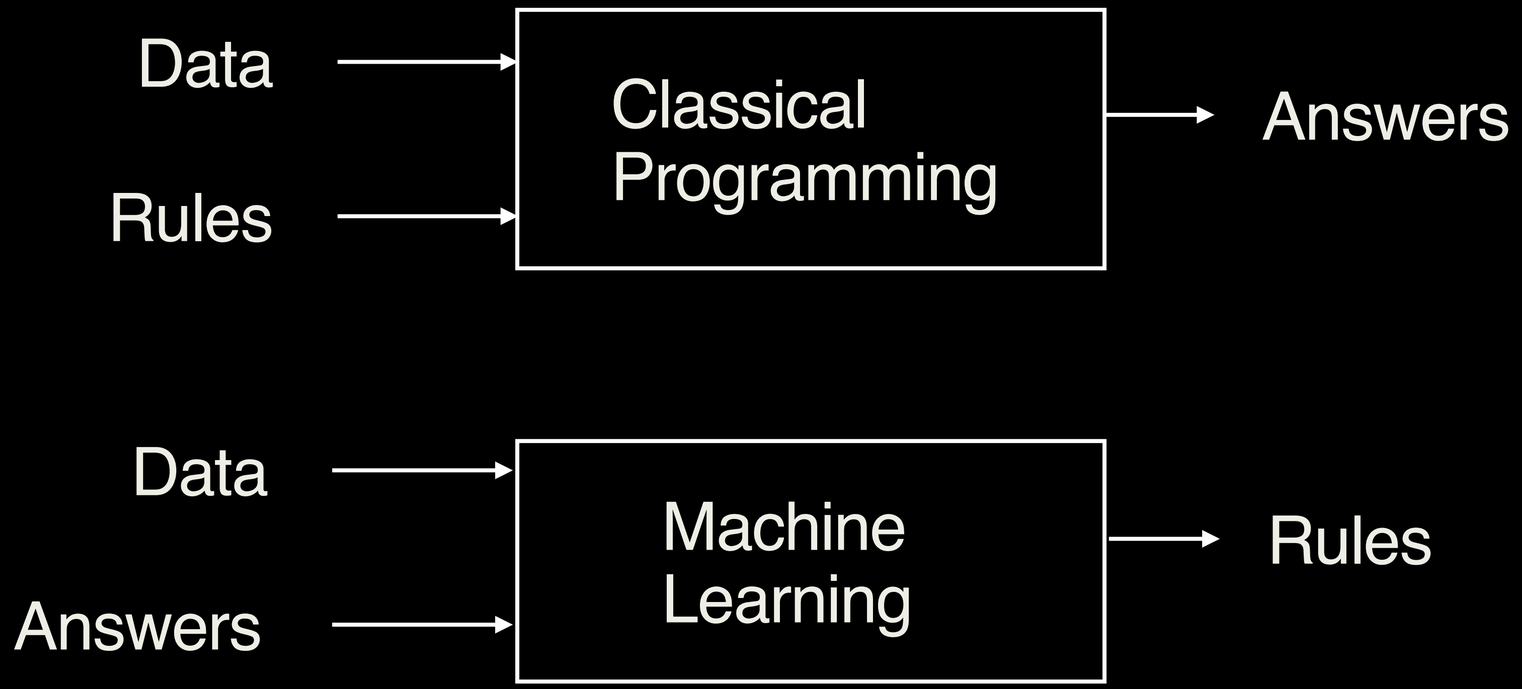
Data

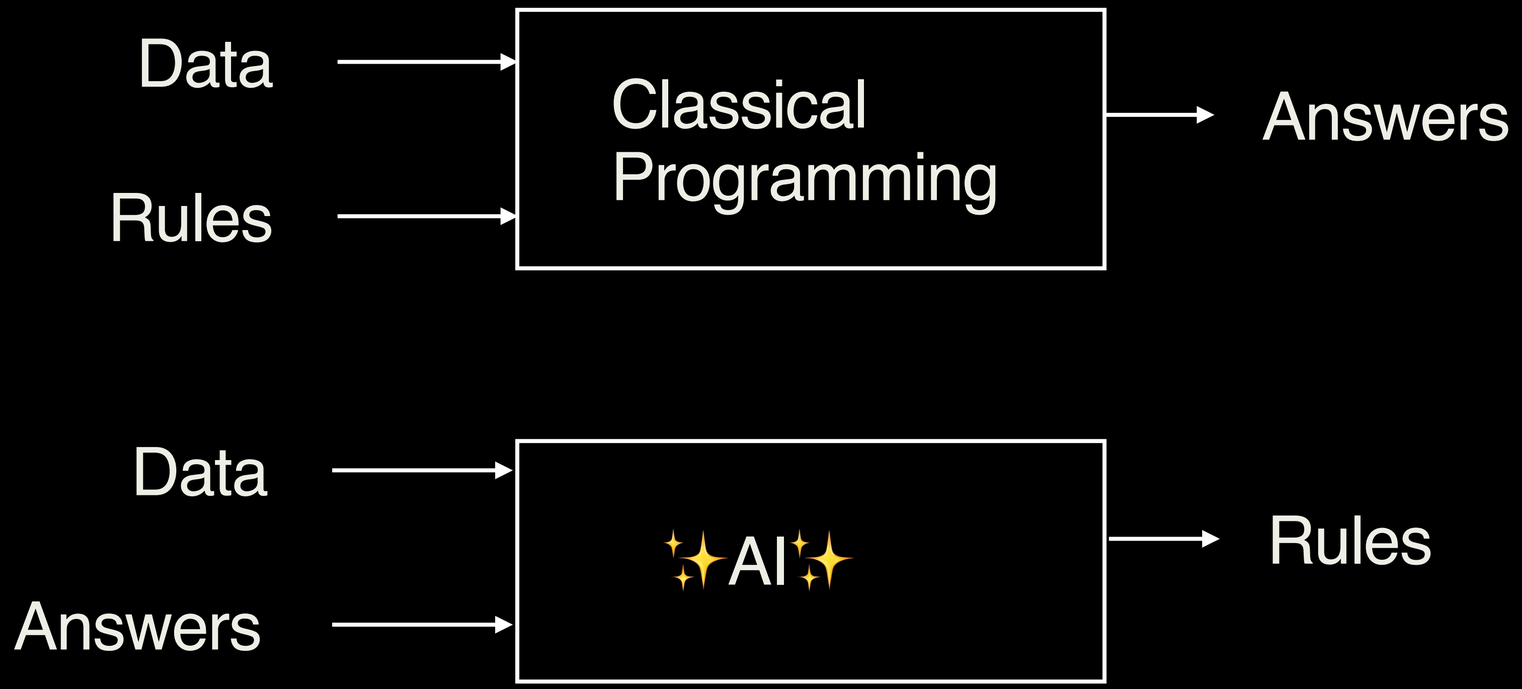
Answers

Fraud Detection


A Big Machine

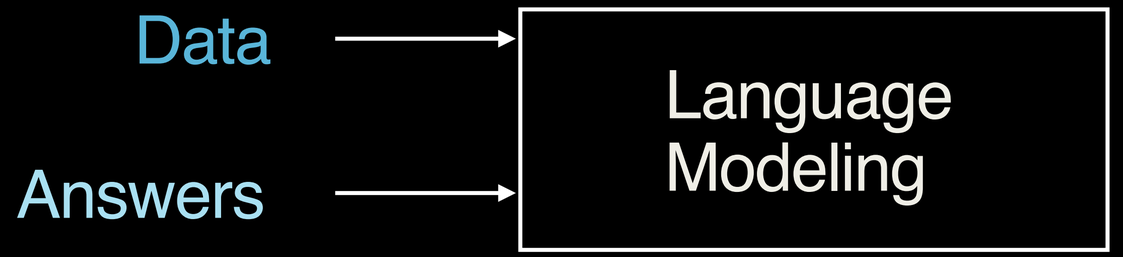

A
Big
Matrix
A Big Machine

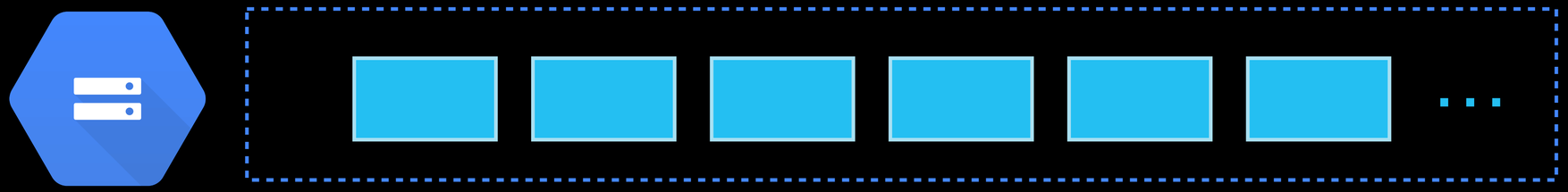

Data
Rules
Classical Programming
Answers
Data
Answers
Machine Learning
Rules


Data
Rules
Classical Programming
Answers
Data
Answers
✨AI✨
Rules

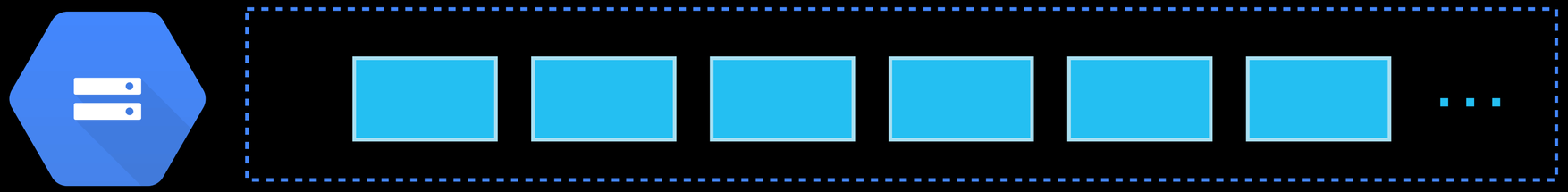
# The Corpus Dump

Customer: Do you all do kids birthdays?
Merchant: Of course!





A Big Machine



GPU Process 1

GPU Process 2

GPU Process 3

GPU Process 4

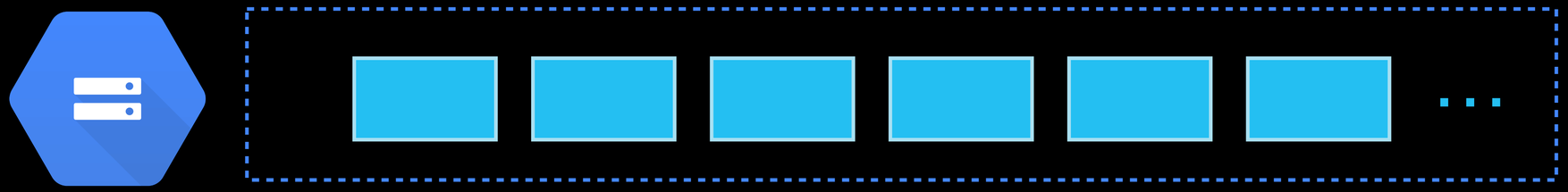
A Big Machine

GPU Process 5

GPU Process 6

GPU Process 7

GPU Process 8



A Big Machine

GPU Process 1
Worker 1
Worker 2
Worker 3
Worker 4

GPU Process 2
Worker 1
Worker 2
Worker 3
Worker 4

GPU Process 3
Worker 1
Worker 2
Worker 3
Worker 4

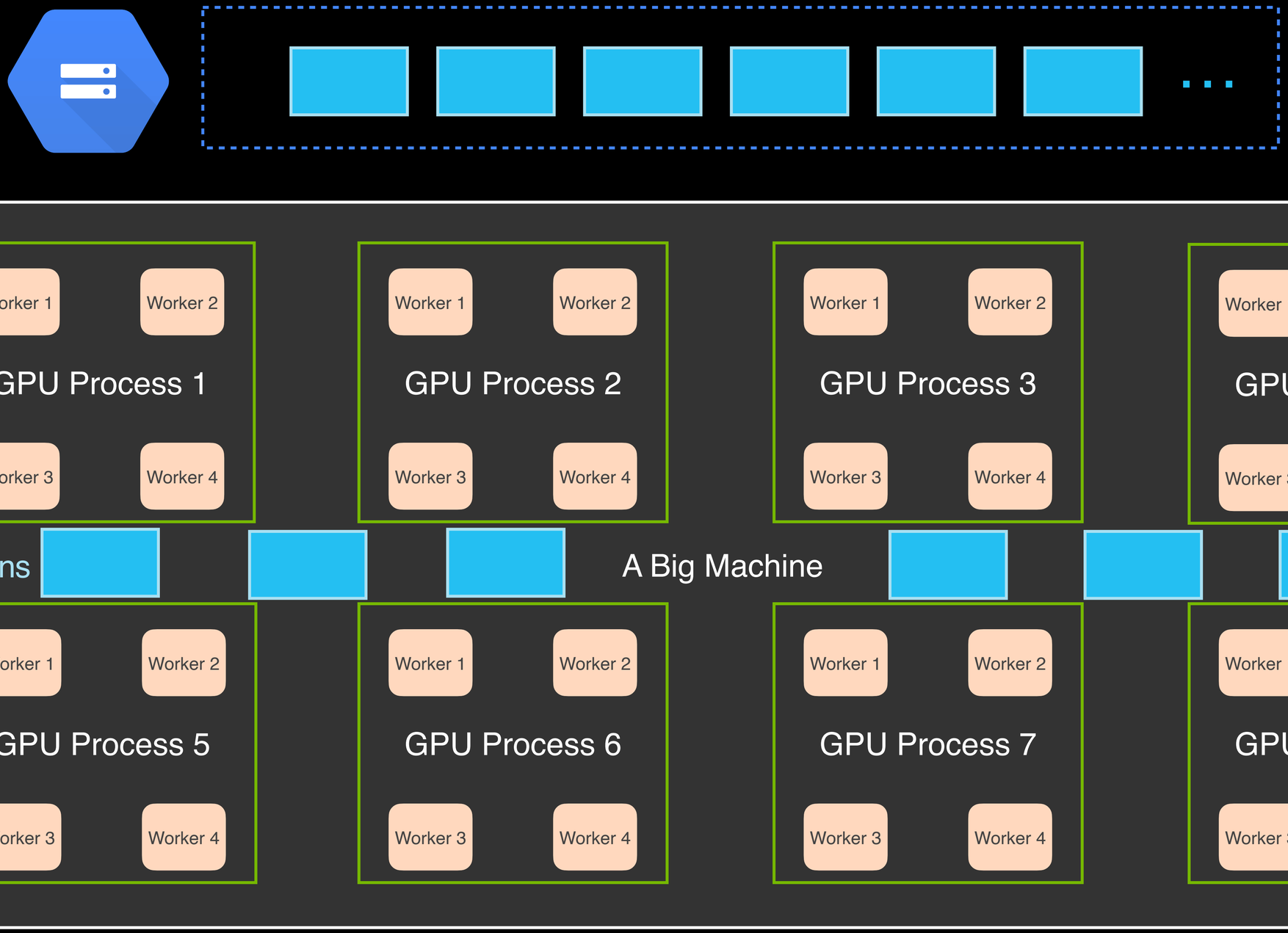
GPU Process 4
Worker 1
Worker 2
Worker 3
Worker 4

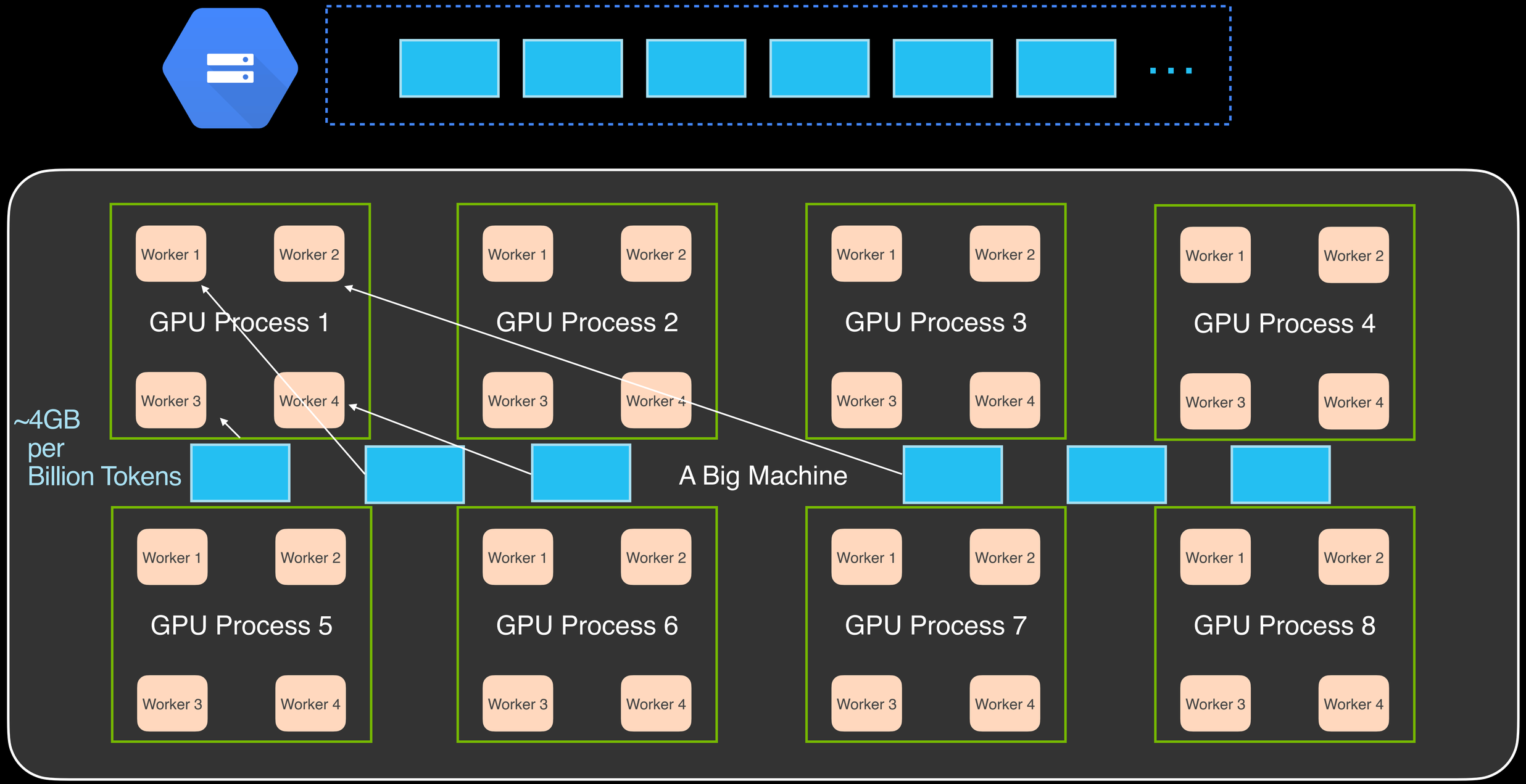
GPU Process 5
Worker 1
Worker 2
Worker 3
Worker 4

GPU Process 6
Worker 1
Worker 2
Worker 3
Worker 4

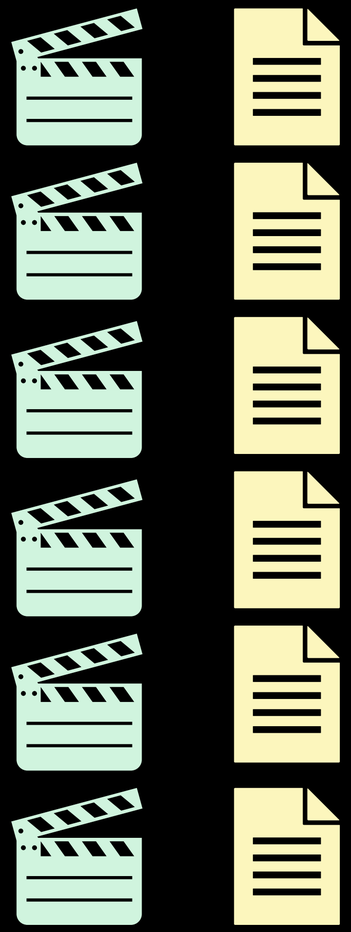
GPU Process 7
Worker 1
Worker 2
Worker 3
Worker 4

GPU Process 8
Worker 1
Worker 2
Worker 3
Worker 4


GPU Process 1
Worker 1
Worker 2
Worker 3
Worker 4
GPU Process 2
Worker 1
Worker 2
Worker 3
Worker 4
GPU Process 3
Worker 1
Worker 2
Worker 3
Worker 4
GPU Process 4
Worker 1
Worker 2
Worker 3
Worker 4
~4GB per Billion Tokens
A Big Machine
GPU Process 5
Worker 1
Worker 2
Worker 3
Worker 4
GPU Process 6
Worker 1
Worker 2
Worker 3
Worker 4
GPU Process 7
Worker 1
Worker 2
Worker 3
Worker 4
GPU Process 8
Worker 1
Worker 2
Worker 3
Worker 4


GPU Process 1
GPU Process 2
GPU Process 3
GPU Process 4
GPU Process 5
GPU Process 6
GPU Process 7
GPU Process 8
Worker 1
Worker 2
Worker 3
Worker 4
~4GB per Billion Tokens
A Big Machine

runway



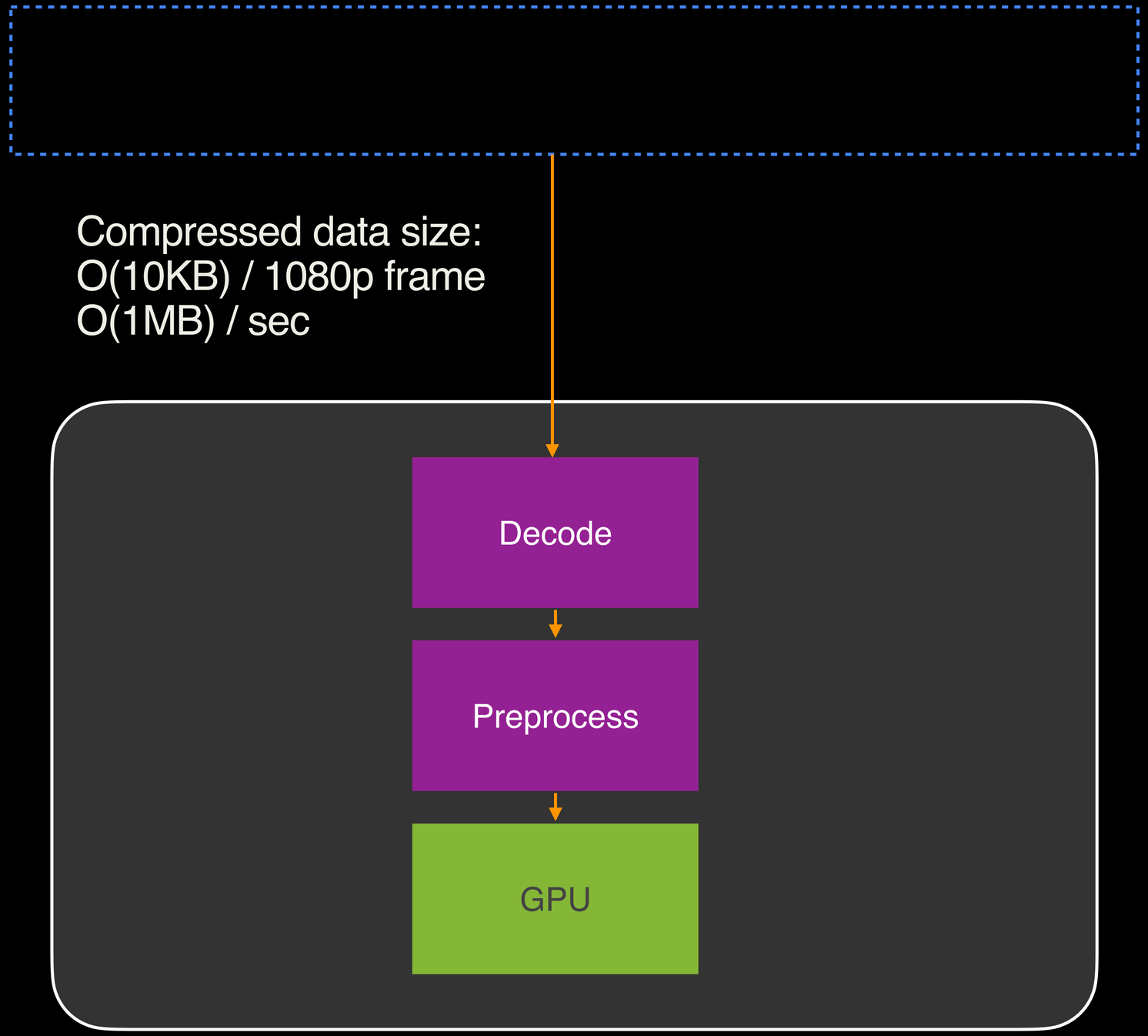
Data

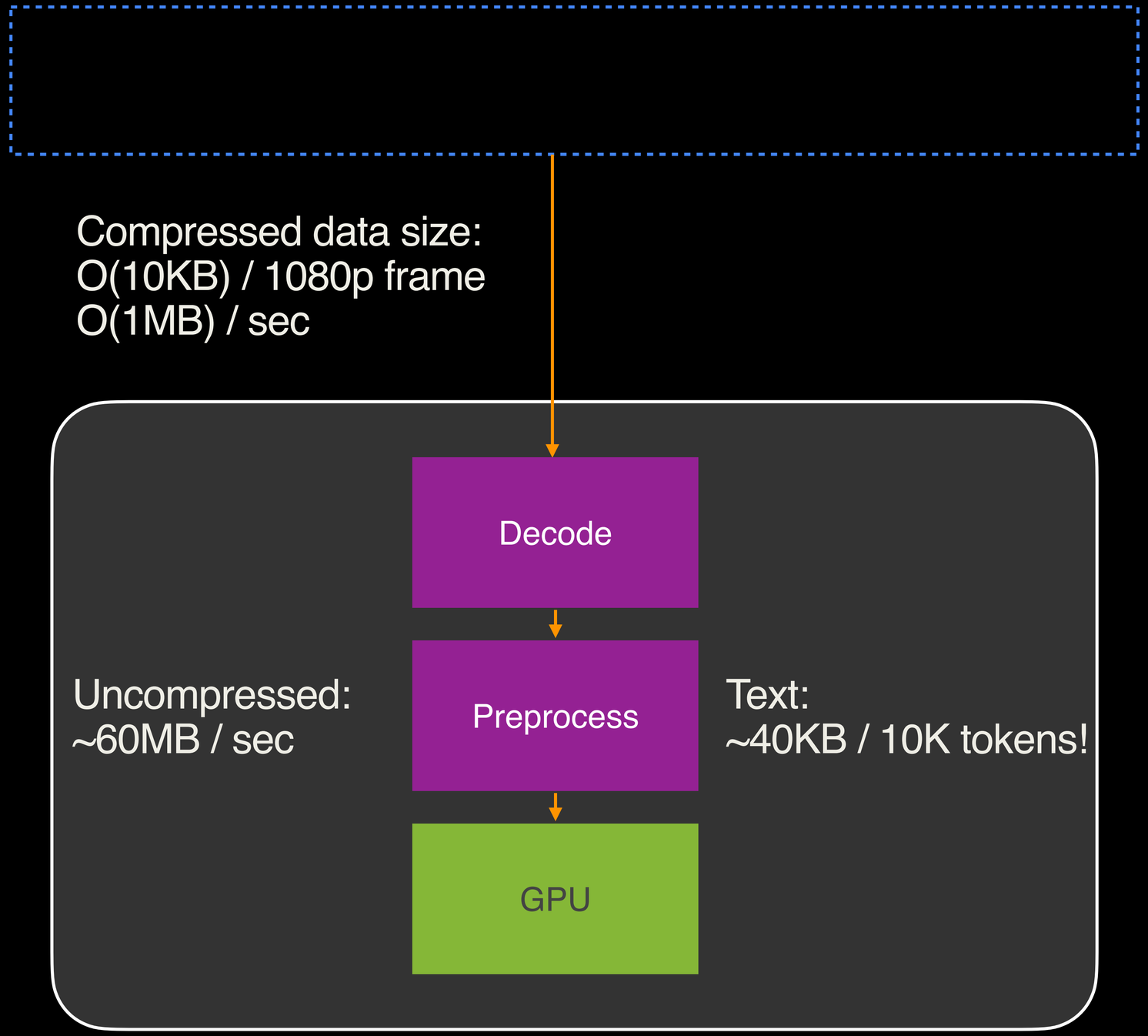
Answers

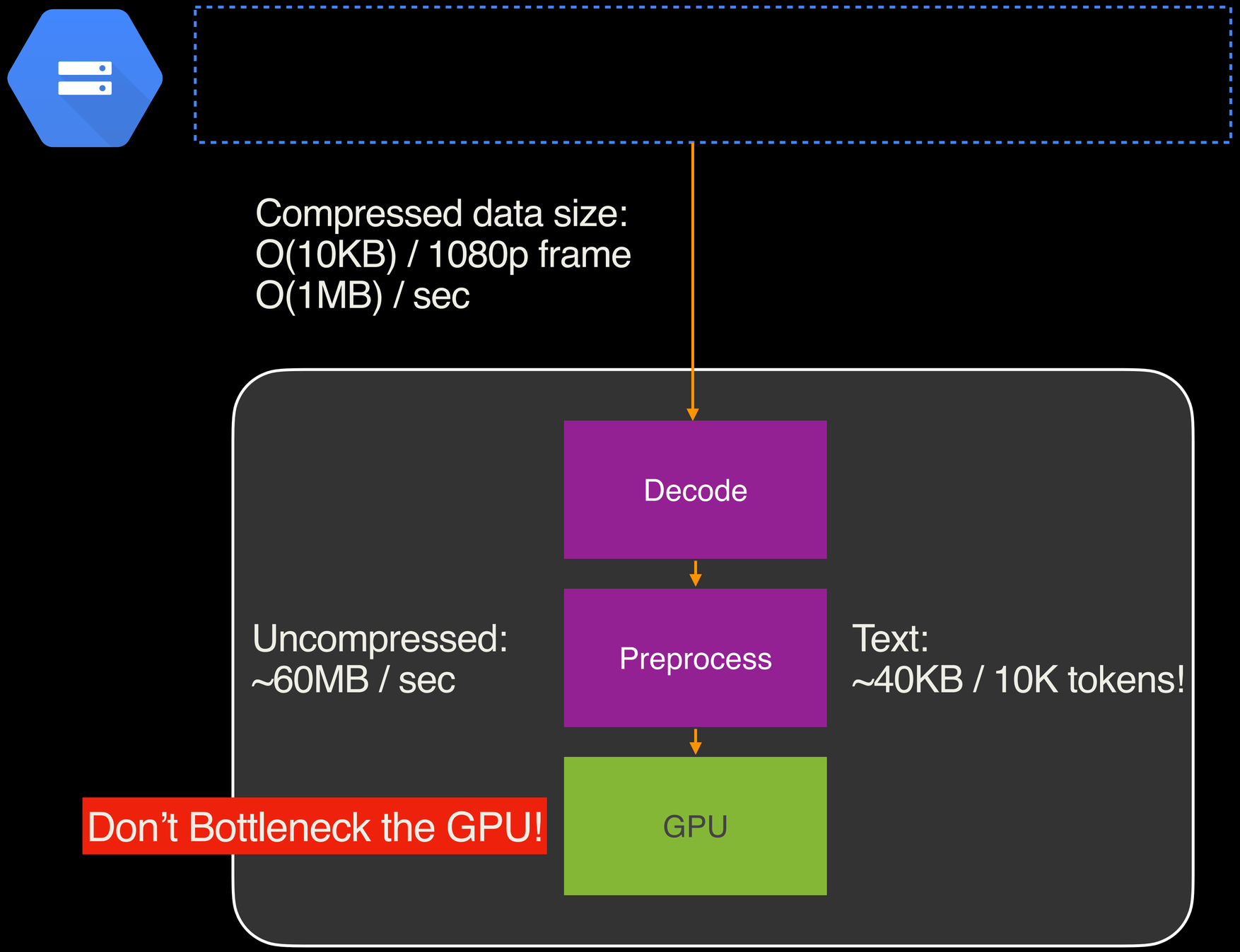
Generative Video

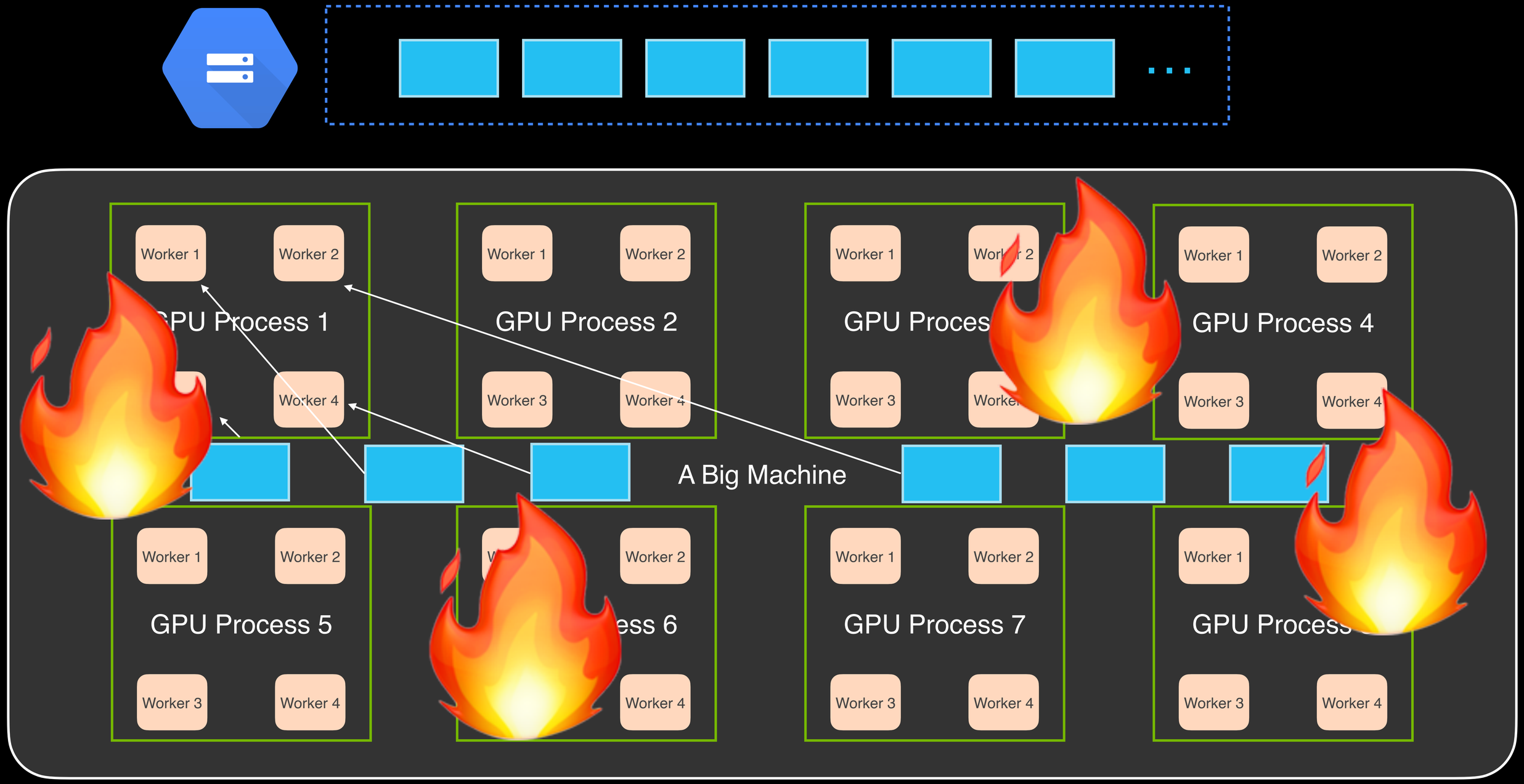

Generative Video

# Videos >> Text

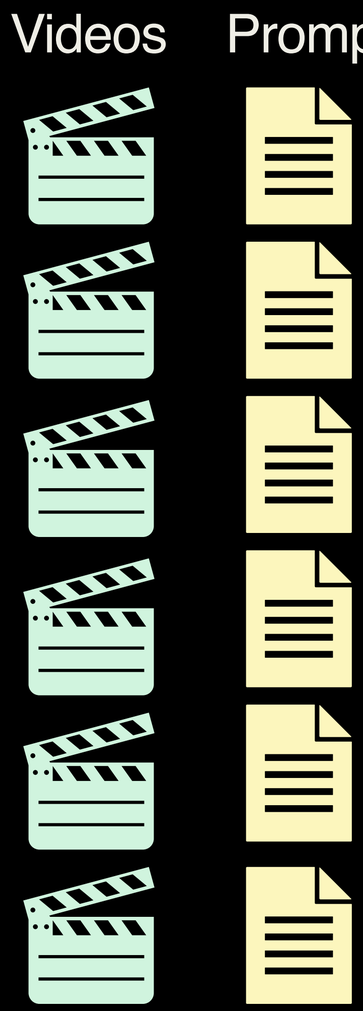

Compressed data size:
O(10KB) / 1080p frame
O(1MB) / sec
Decode
Preprocess
GPU


Compressed data size:
O(10KB) / 1080p frame
O(1MB) / sec
Decode
Preprocess
GPU
Uncompressed:
~60MB / sec
Text:
~40KB / 10K tokens!


Compressed data size:
O(10KB) / 1080p frame
O(1MB) / sec
Decode
Preprocess
GPU
Uncompressed:
~60MB / sec
Text:
~40KB / 10K tokens!
Don’t Bottleneck the GPU!


Worker 1
Worker 2
Worker 3
Worker 4
GPU Process 1
GPU Process 2
GPU Process 4
GPU Process 5
GPU Process 7
A Big Machine

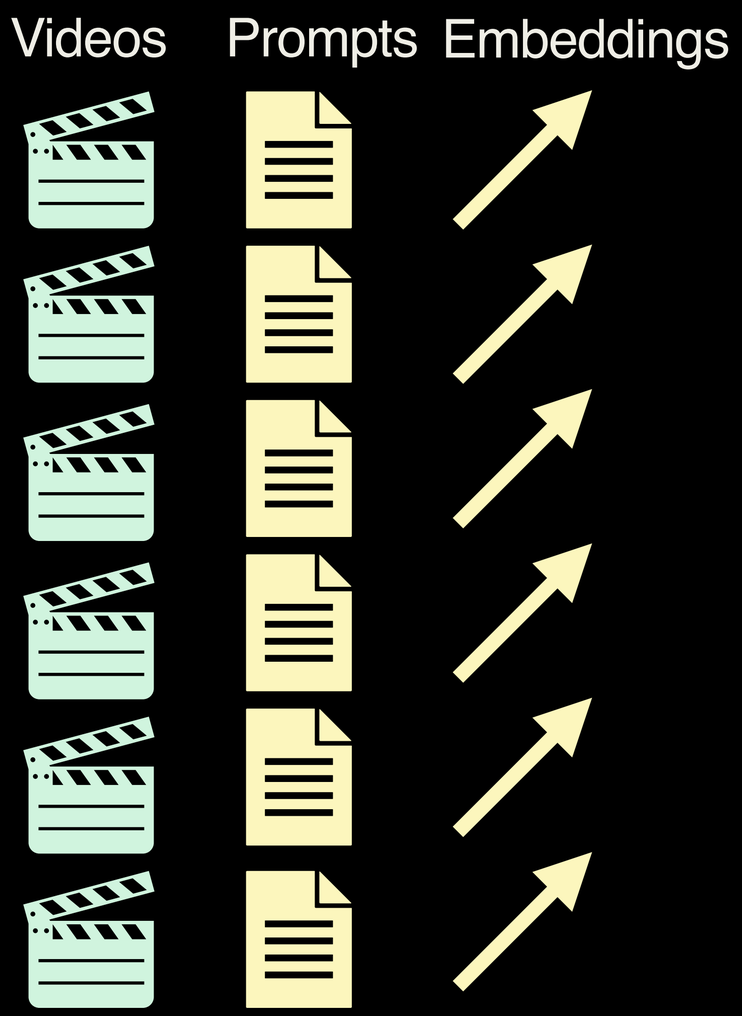
# Data Refining




Data

Answers

Generative Video




Data

Answers

Generative Video

# Structure and Content-Guided Video Synthesis with Diffusion Models

Patrick Esser Johnathan Chiu Parmida Atighehchian
Jonathan Granskog Anastasis Germanidis
Runway
https://research.runwayml.com/gen1

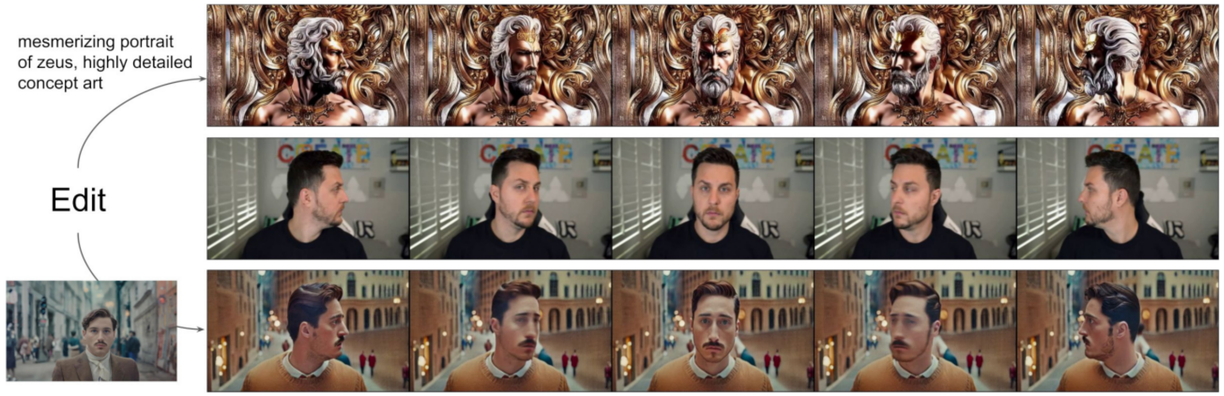



Figure 1. **Guided Video Synthesis** We present an approach based on latent video diffusion models that synthesizes videos (top and bottom) guided by content described through text (top) or images (bottom) while keeping the structure of an input video (middle).

## Structure and Content-Guided Video Synthesis with Diffusion Models

Patrick Esser Johnathan Chiu Parmida Atighehchian
Jonathan Granskog Anastasis Germanidis
Runway
https://research.runwayml.com/gen1

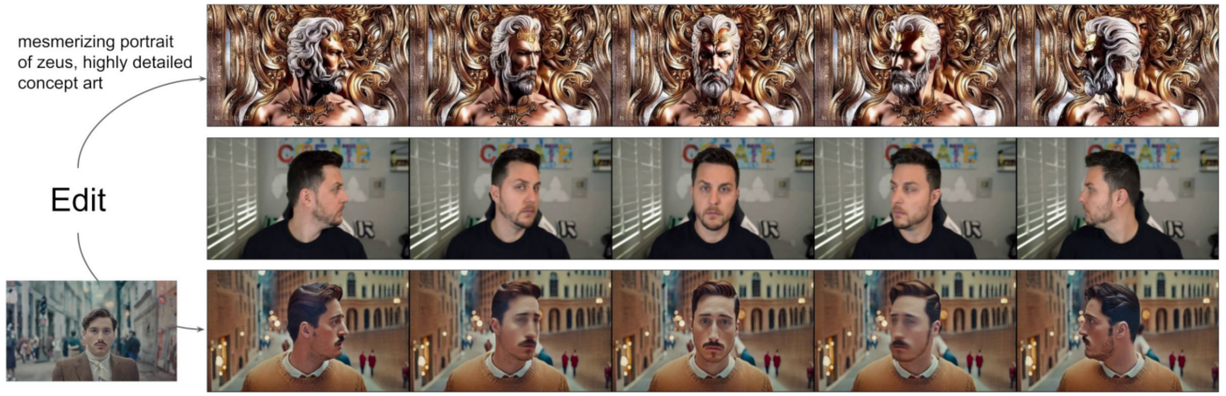
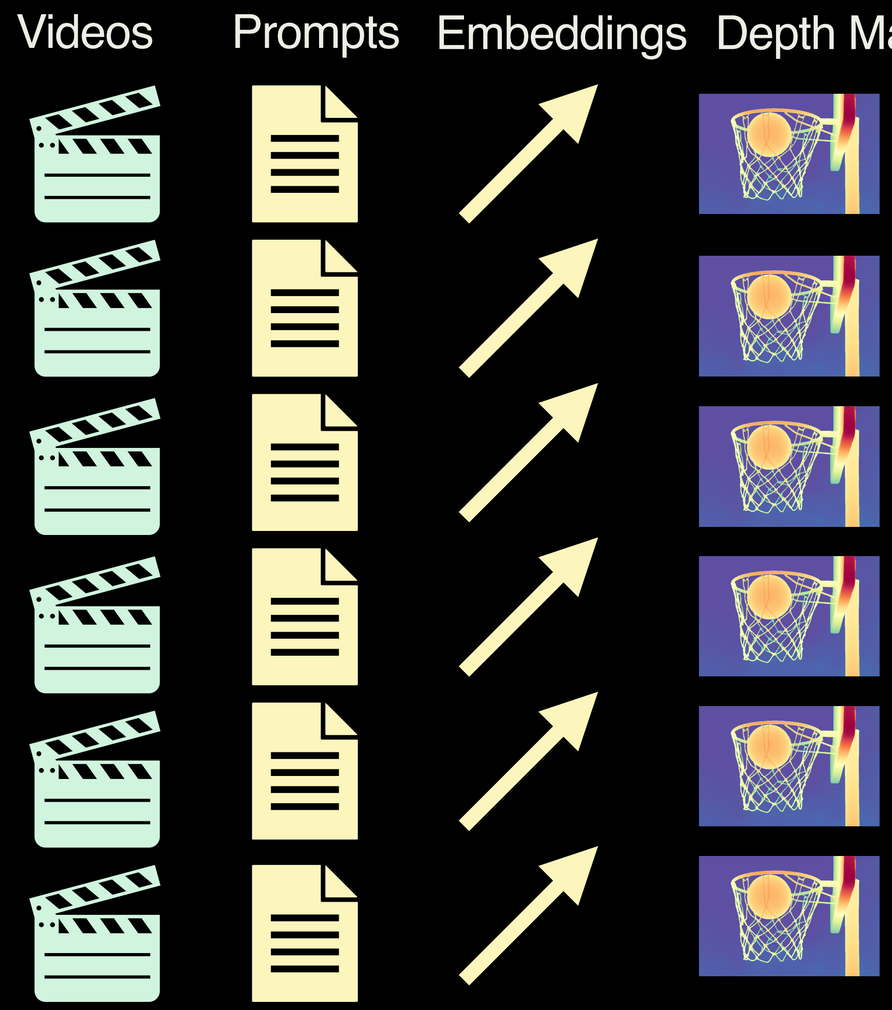



Figure 1. **Guided Video Synthesis** We present an approach based on latent video diffusion models that synthesizes videos (top and bottom) guided by content described through text (top) or images (bottom) while keeping the structure of an input video (middle).

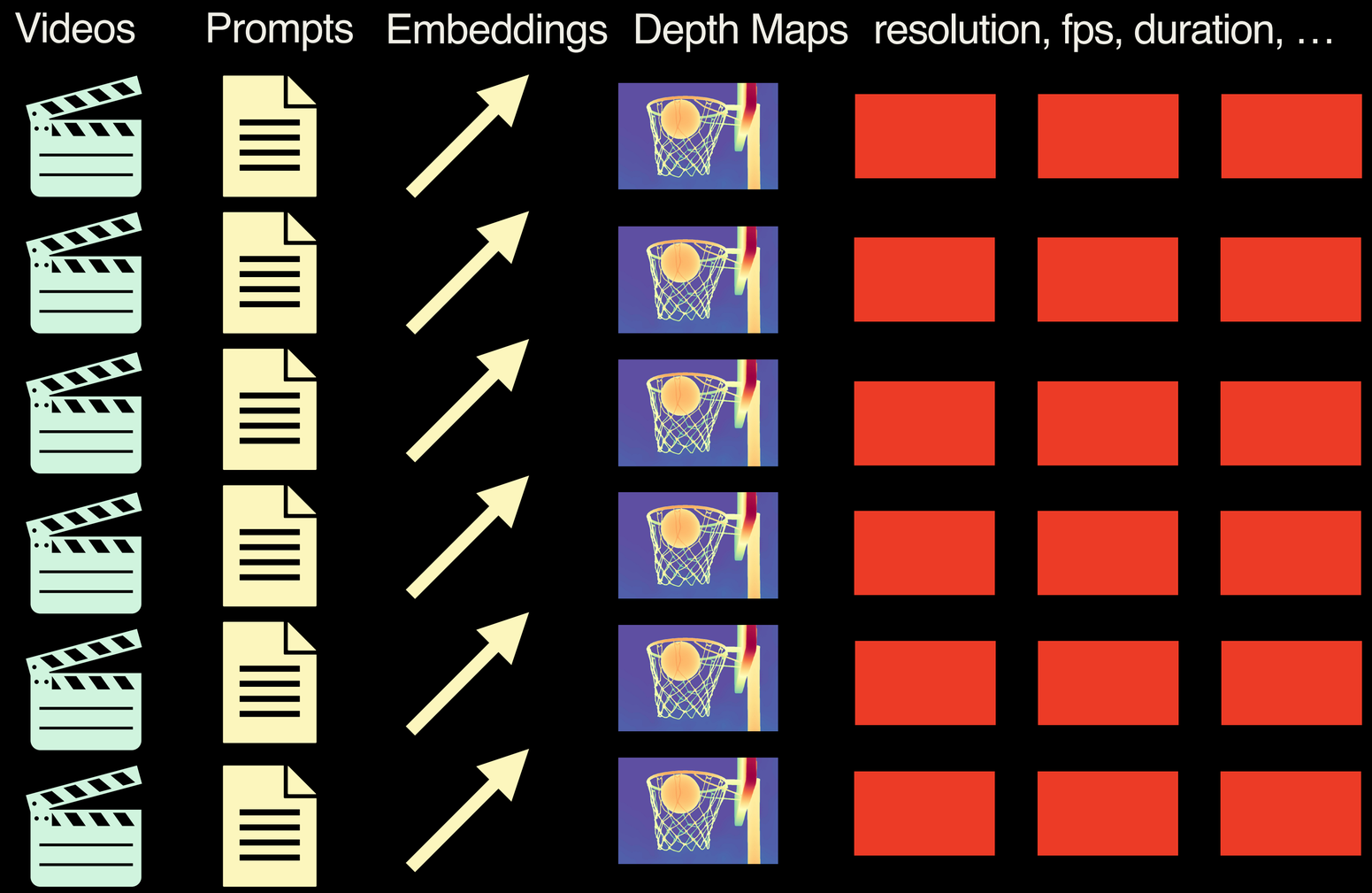

Depth Anything V2, Neurips ‘24




Data

Answers

Generative Video




Generative Video

Videos Prompts Embeddings Depth Maps resolution, fps, duration, …

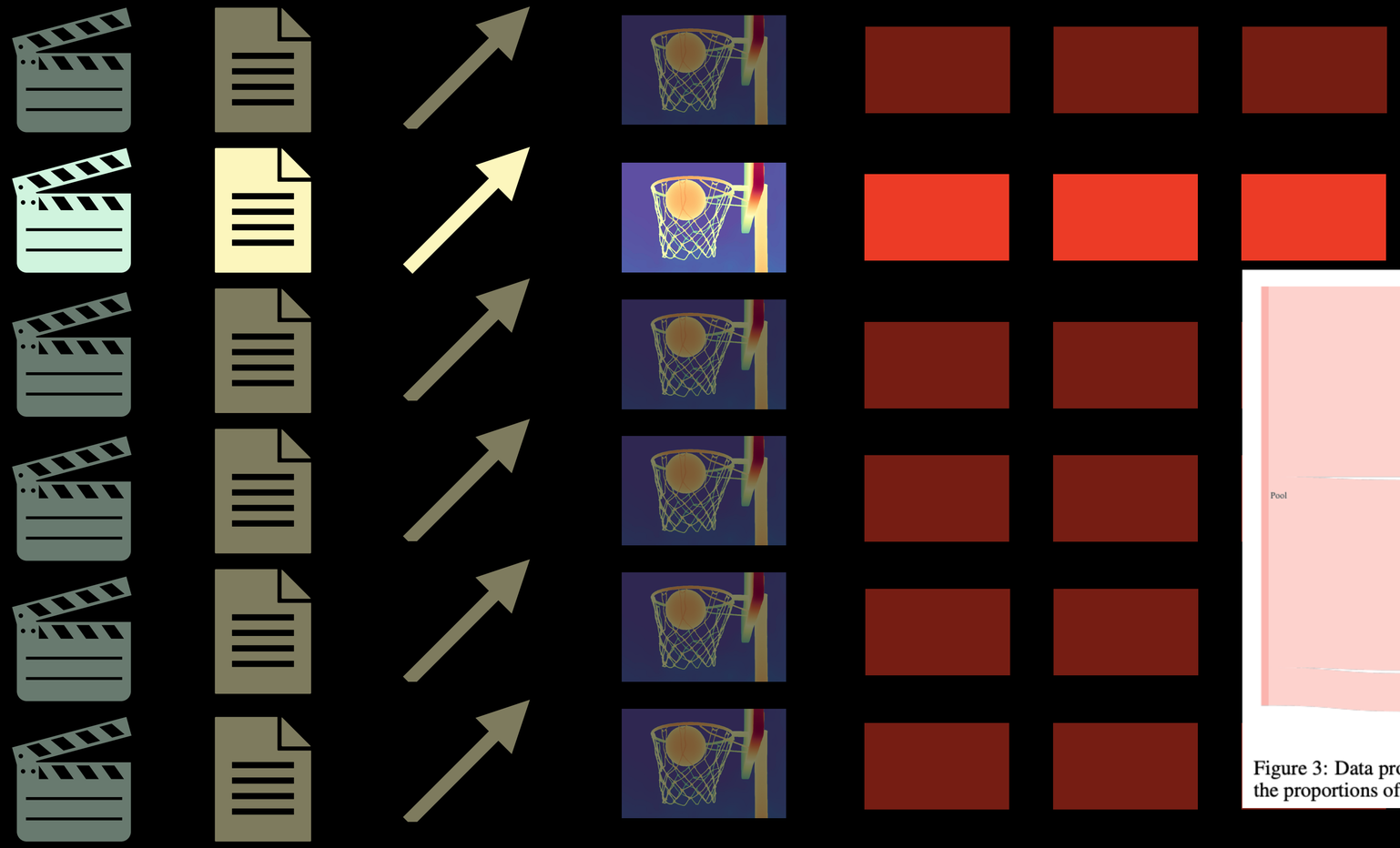



Figure 3: Data provisioning across different training phases. For each stage, we dynamically adjust the proportions of data related to motion, quality, and category based on data throughput.

Wan 2.1, Alibaba ‘25

# Taking Stock

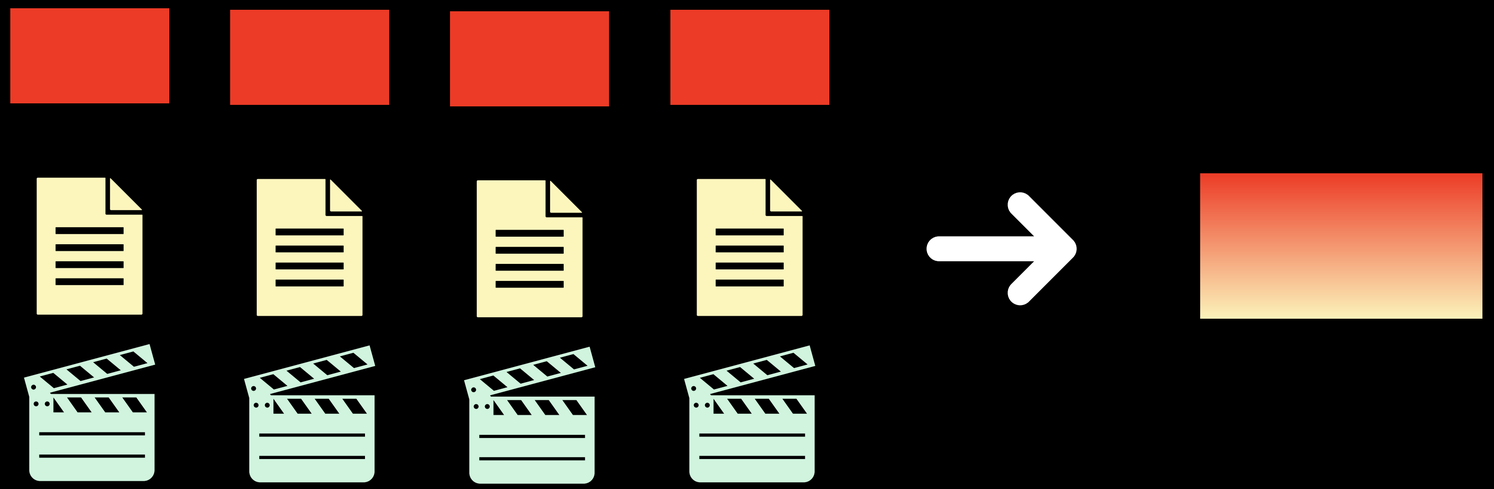
1. Massive unstructured data
2. Structured data
3. Large column appends w/ backfills
4. Dynamic partitioning across workers

# One Off vs. Continual Iteration

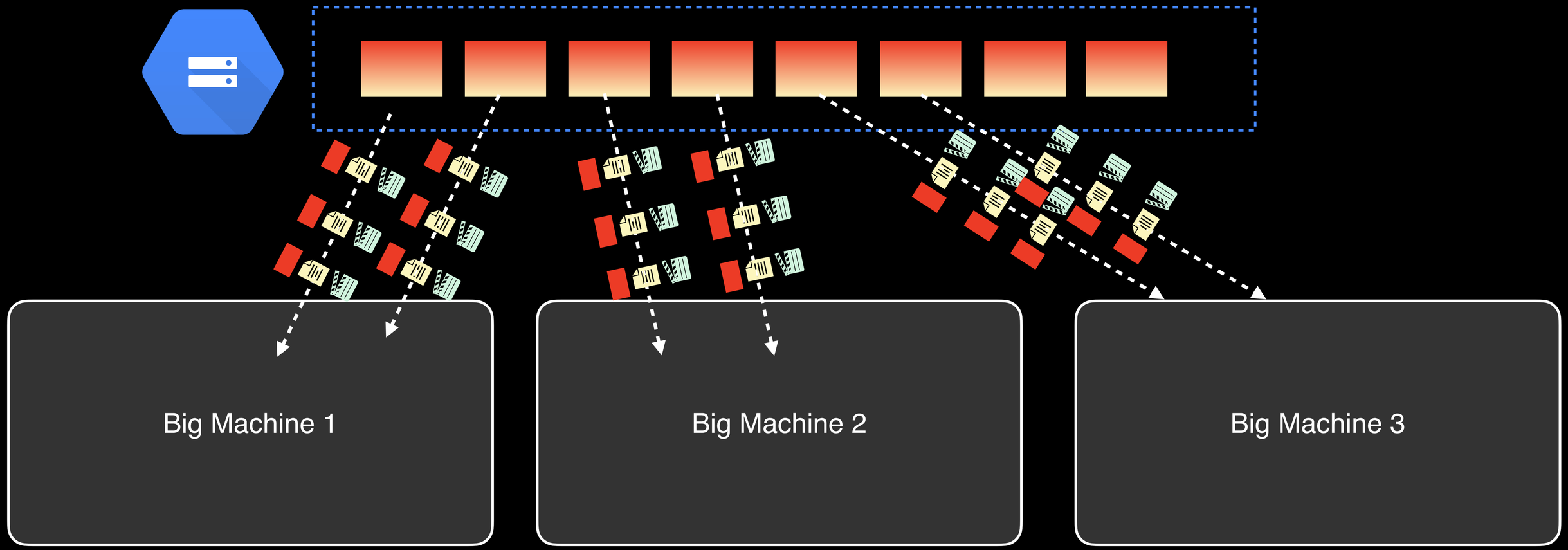
# Taking Stock

1. Massive unstructured data
2. Structured data
3. Large column appends w/ backfills
4. Dynamic partitioning across workers
5. EDA

# Webdataset (aka tarballs + API)



# Webdataset (aka tarballs + API)



# Taking Stock — Webdataset

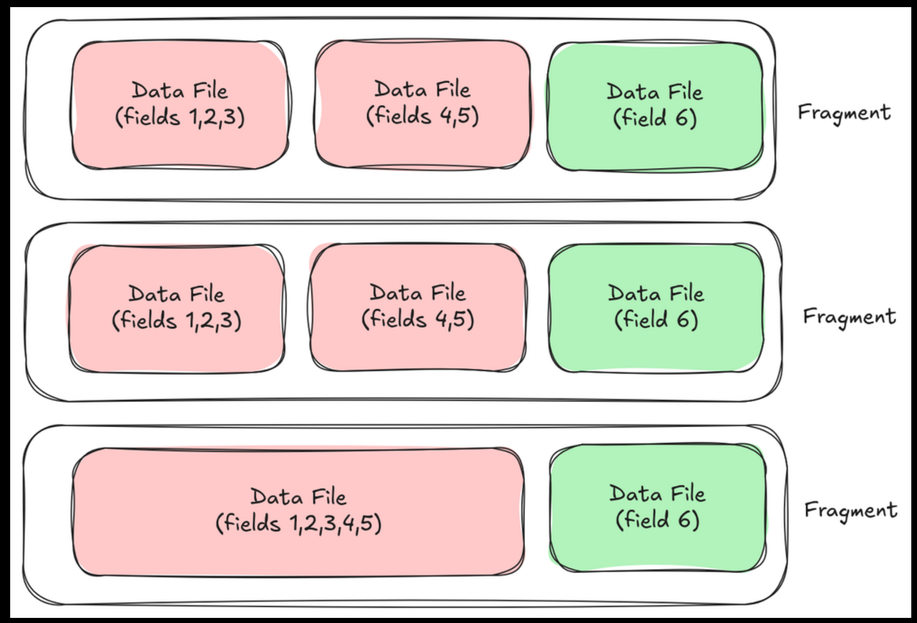
✅ 1. Massive unstructured data
✅ 2. Structured data
❌ 3. Large column appends w/ backfills
😐 4. Dynamic partitioning across workers
❌ 5. EDA

Object storage-native
Columnar
Heterogeneous types
Fast random access


LanceDB

# LanceDB

- File + table format
- Column appends w/o rewriting data
- Fast random access
- Multimodal support
- Lots of other bells and whistles (versioning, arrow integration, vector search, etc…)



# Taking Stock — Lance

✅ 1. Massive unstructured data
✅ 2. Structured data
✅ 3. Column appends
4. Dynamic partitioning across workers
5. EDA

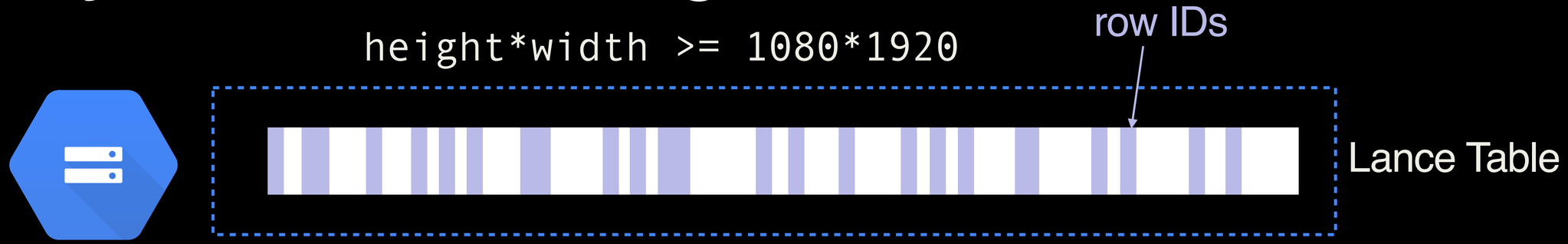
# Lance Dynamic Partitioning


Big Machine 1

Big Machine 2

Big Machine 3

# Lance Dynamic Partitioning

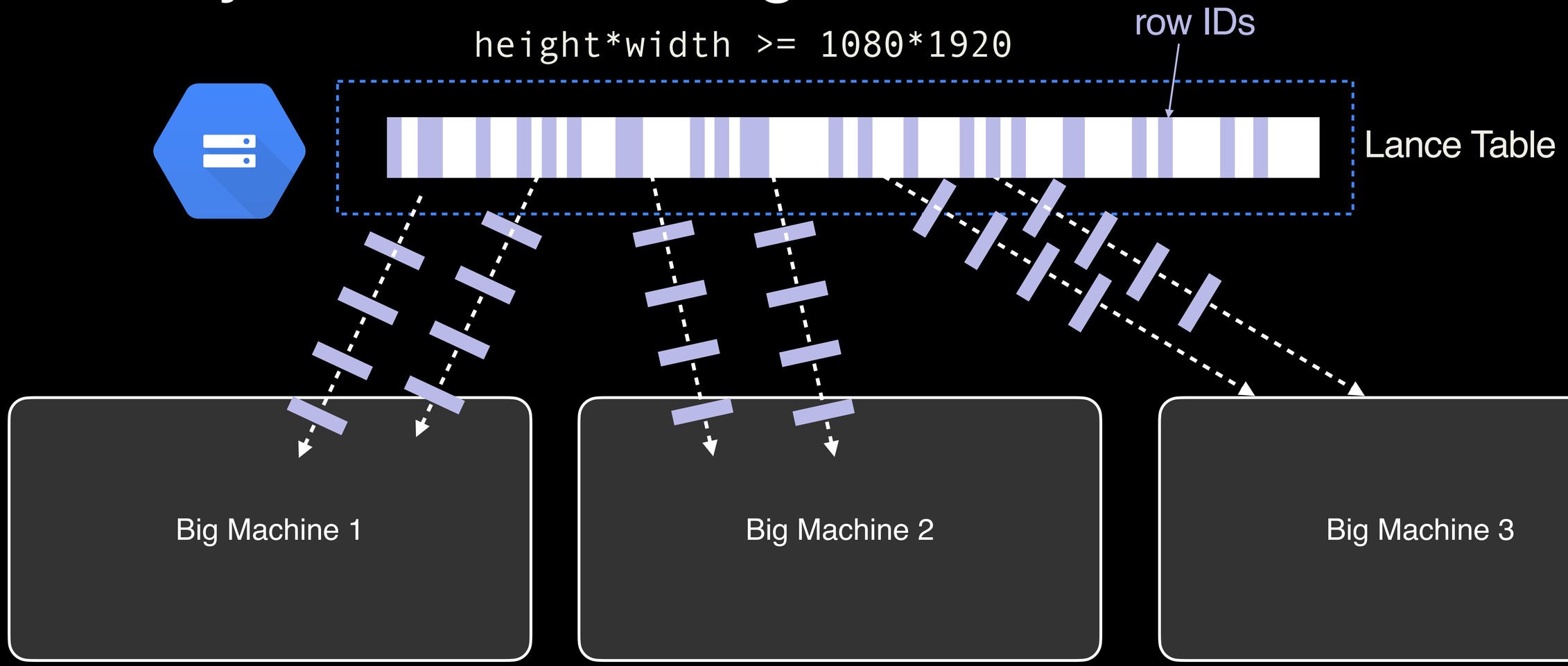



Big Machine 1

Big Machine 2

Big Machine 3

# Lance Dynamic Partitioning



# Taking Stock — Lance

✅ 1. Massive unstructured data
✅ 2. Structured data
✅ 3. Column appends
✅ 4. Dynamic partitioning across workers
5. EDA

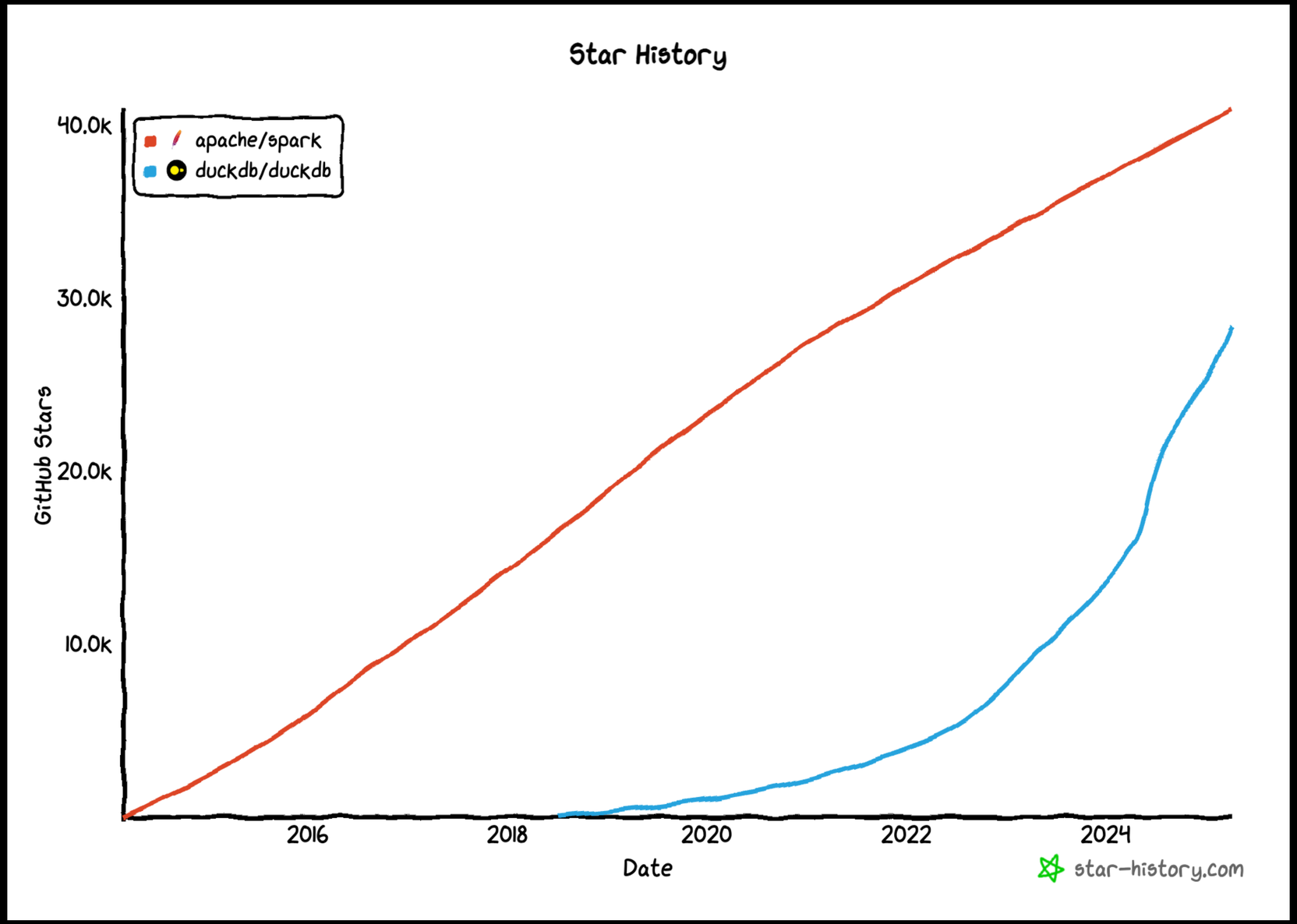
# Taking Stock — Lance

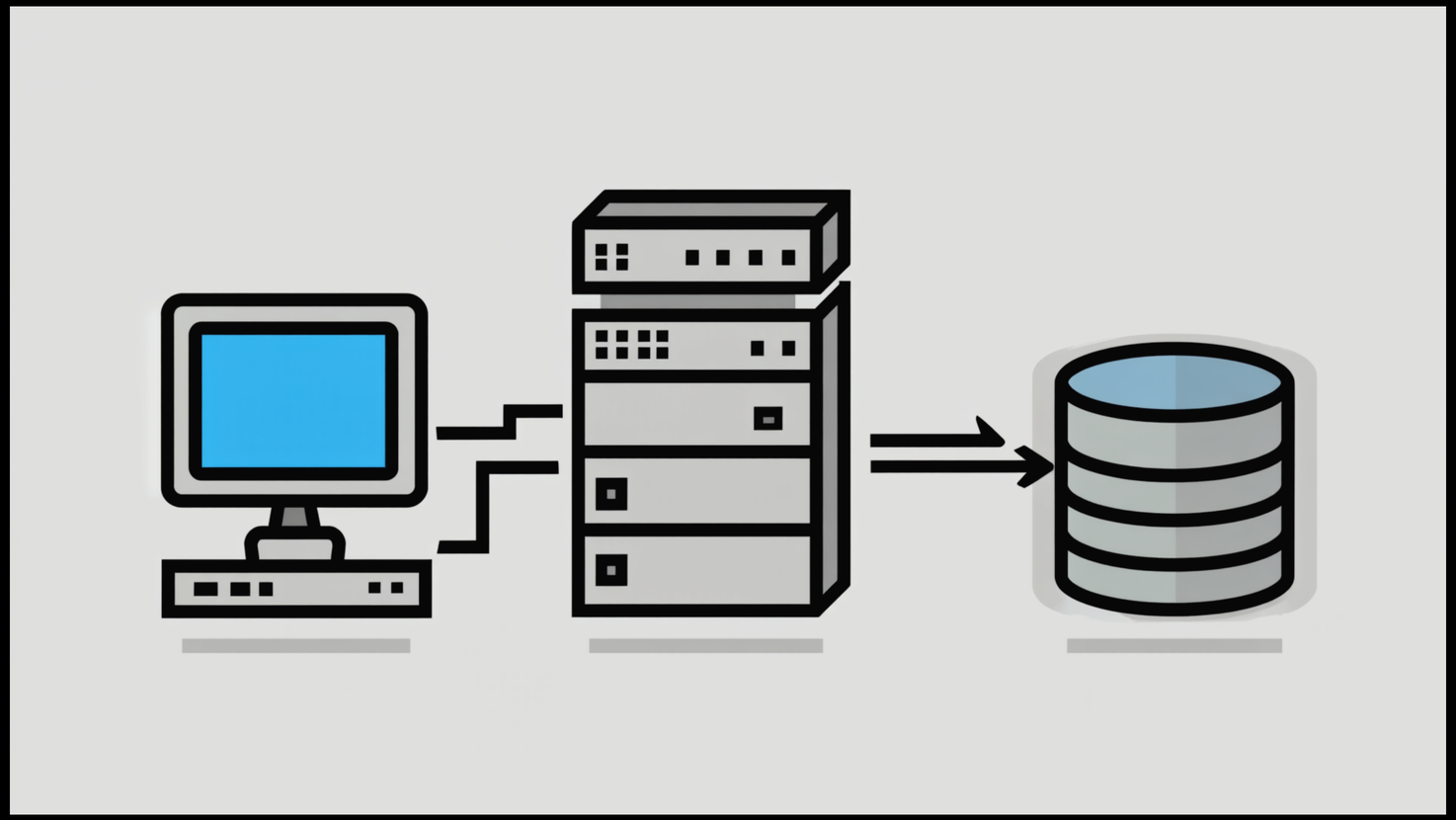


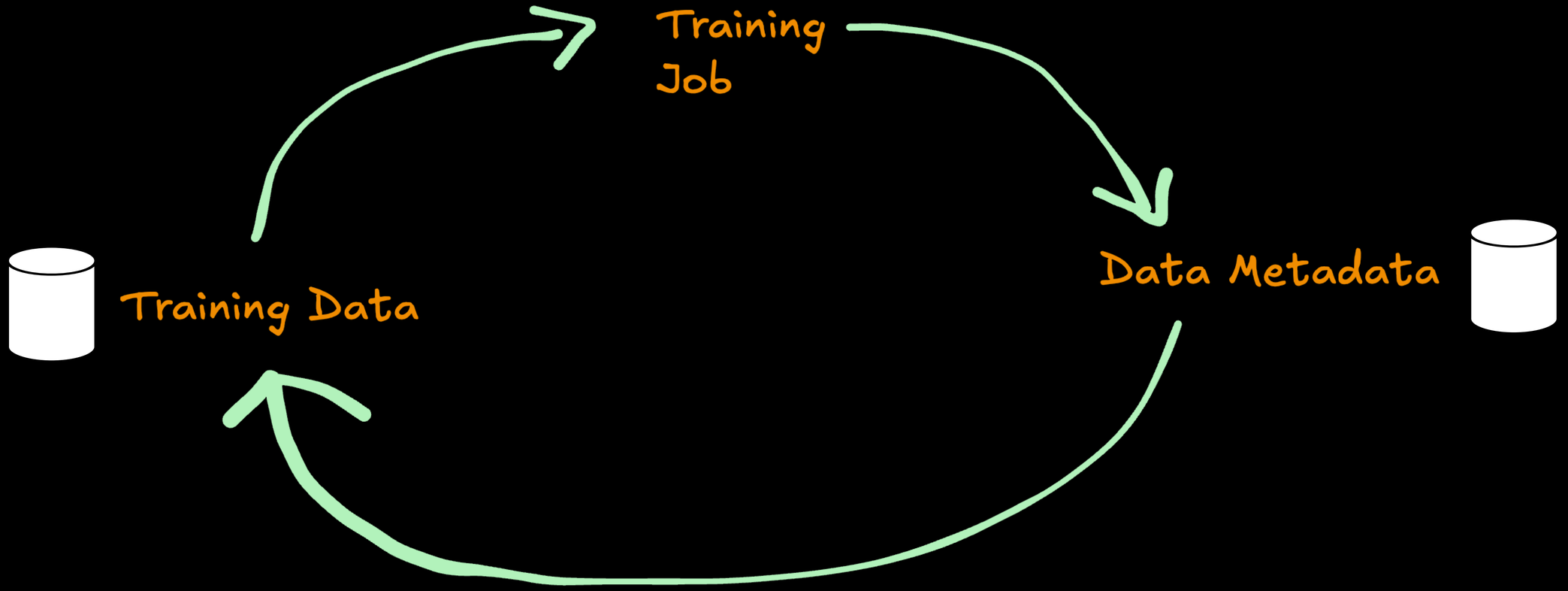
1. ✅ Massive unstructured data
2. ✅ Structured data
3. ✅ Column appends
4. ✅ Dynamic partitioning across workers
5. ✅ EDA

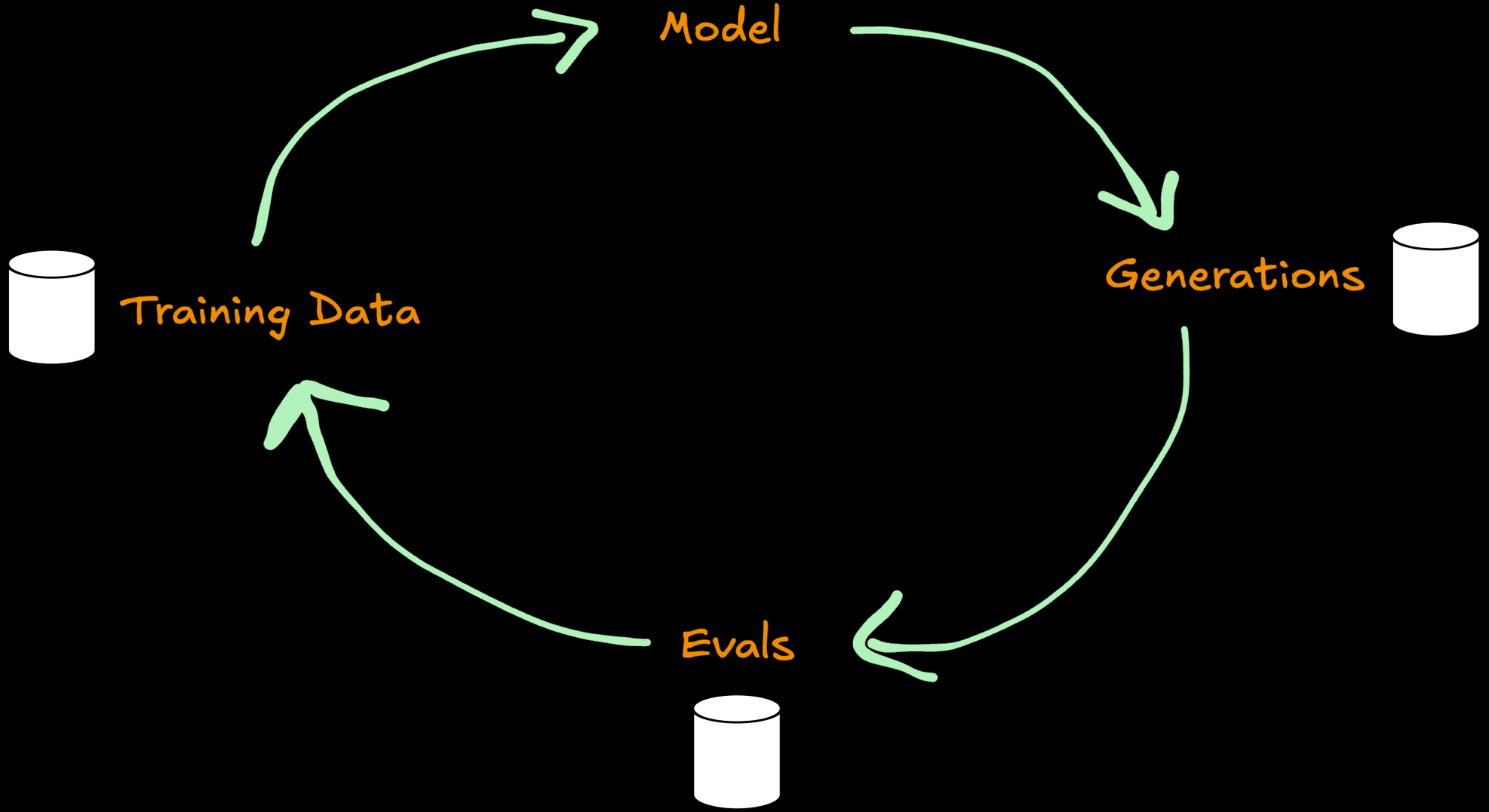
```sql
SELECT
  COUNT(1) AS num_hd
FROM my_videos
WHERE
  height * width >= 1080 * 1920
```

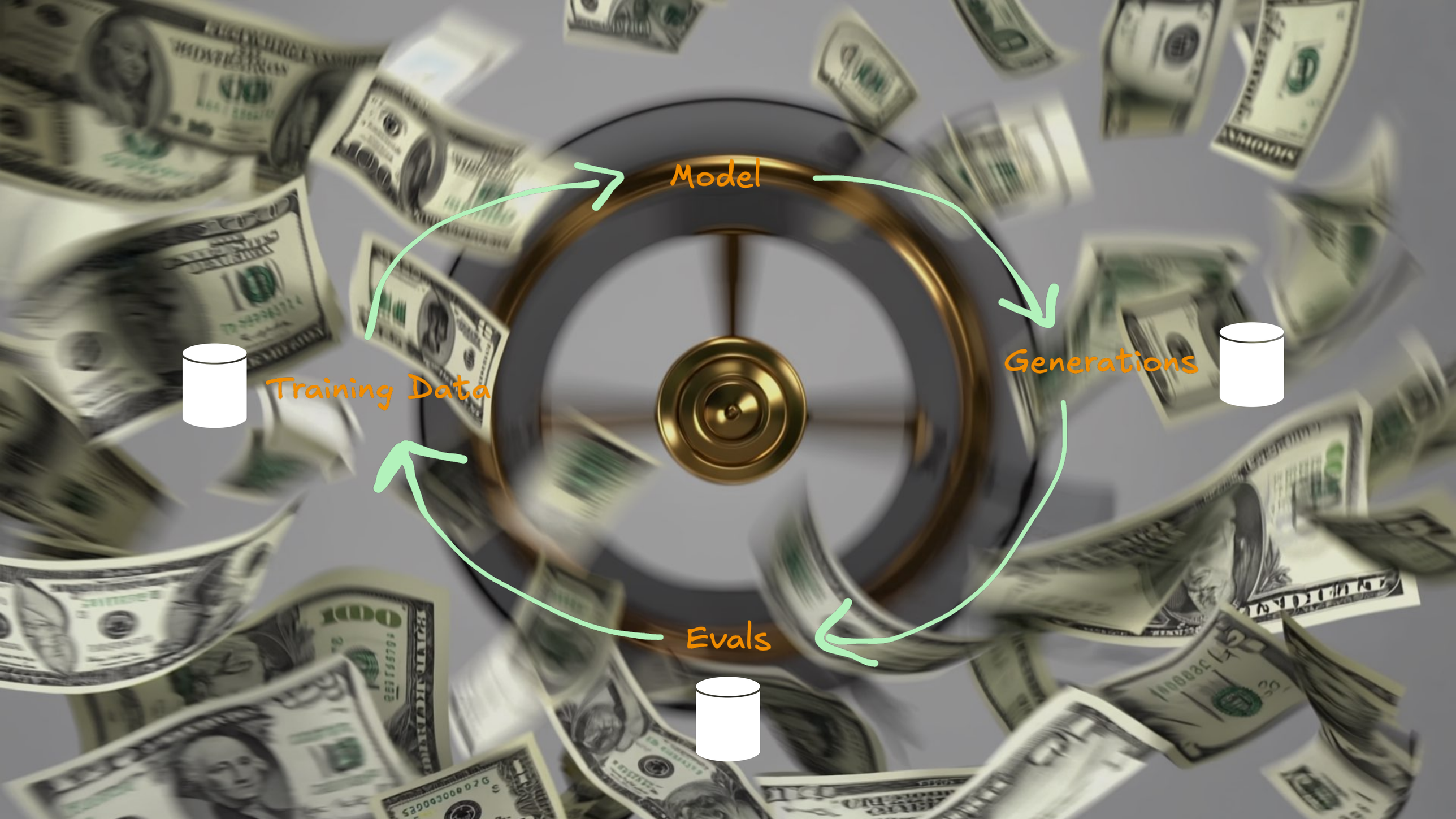

THE WORLD AFTER YOUR DATA IS IN A DB
imgflip.com


Star History
apache/spark
duckdb/duckdb
40.0k
30.0k
20.0k
10.0k
GitHub Stars
2016
2018
2020
2022
2024
Date
star-history.com




Training
Job
Training Data
Data Metadata


Model
Generations
Evals
Training Data

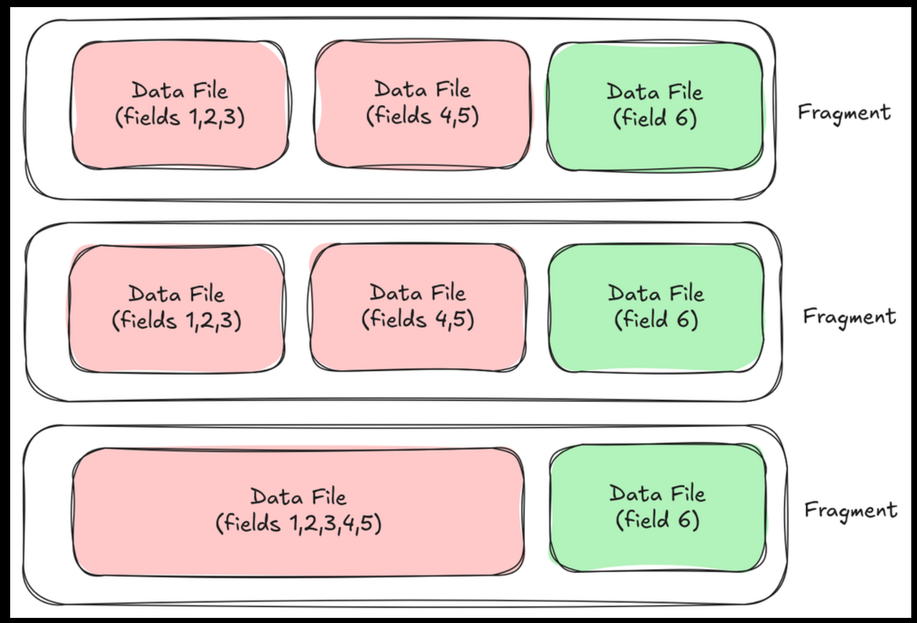

Model
Generations
Evals
Training Data

# That easy, huh?

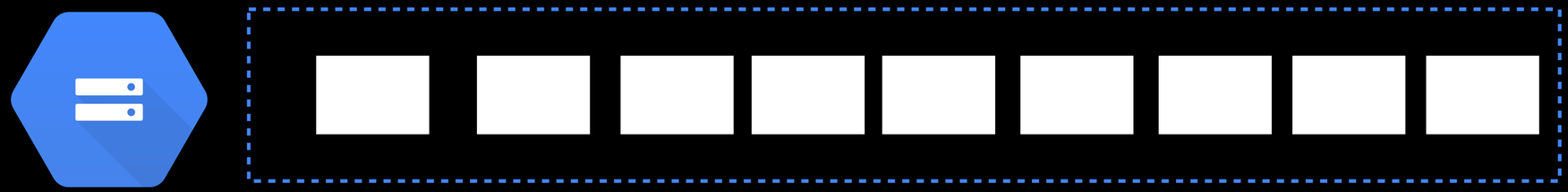
# Lessons Learned


STOP TRYING TO MAKE
PARQUET HAPPEN
imgflip.com


STOP TRYING TO MAKE
PARQUET HAPPEN
imgflip.com


Data File (fields 1,2,3)
Data File (fields 4,5)
Data File (field 6)
Fragment
Data File (fields 1,2,3)
Data File (fields 4,5)
Data File (field 6)
Fragment
Data File (fields 1,2,3,4,5)
Data File (field 6)
Fragment

# Lance Fragment Partitioning

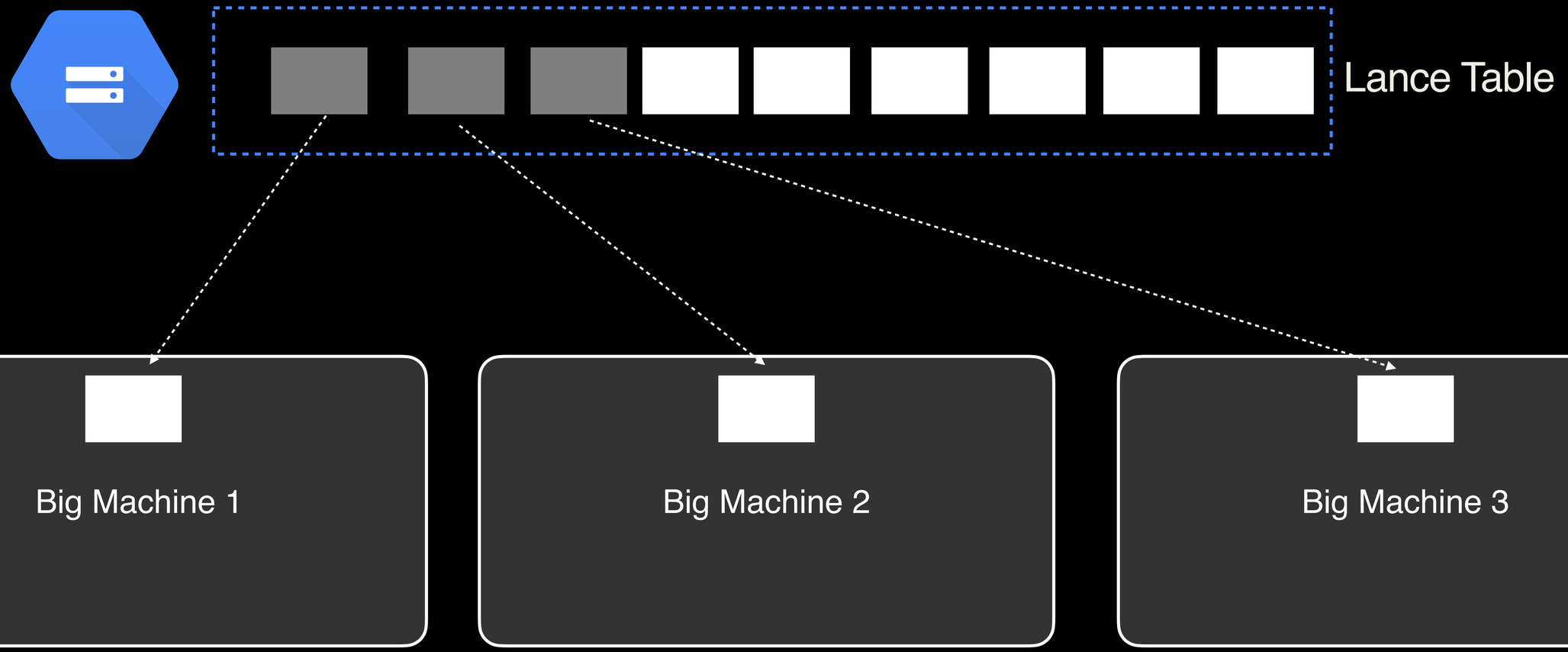



Big Machine 1

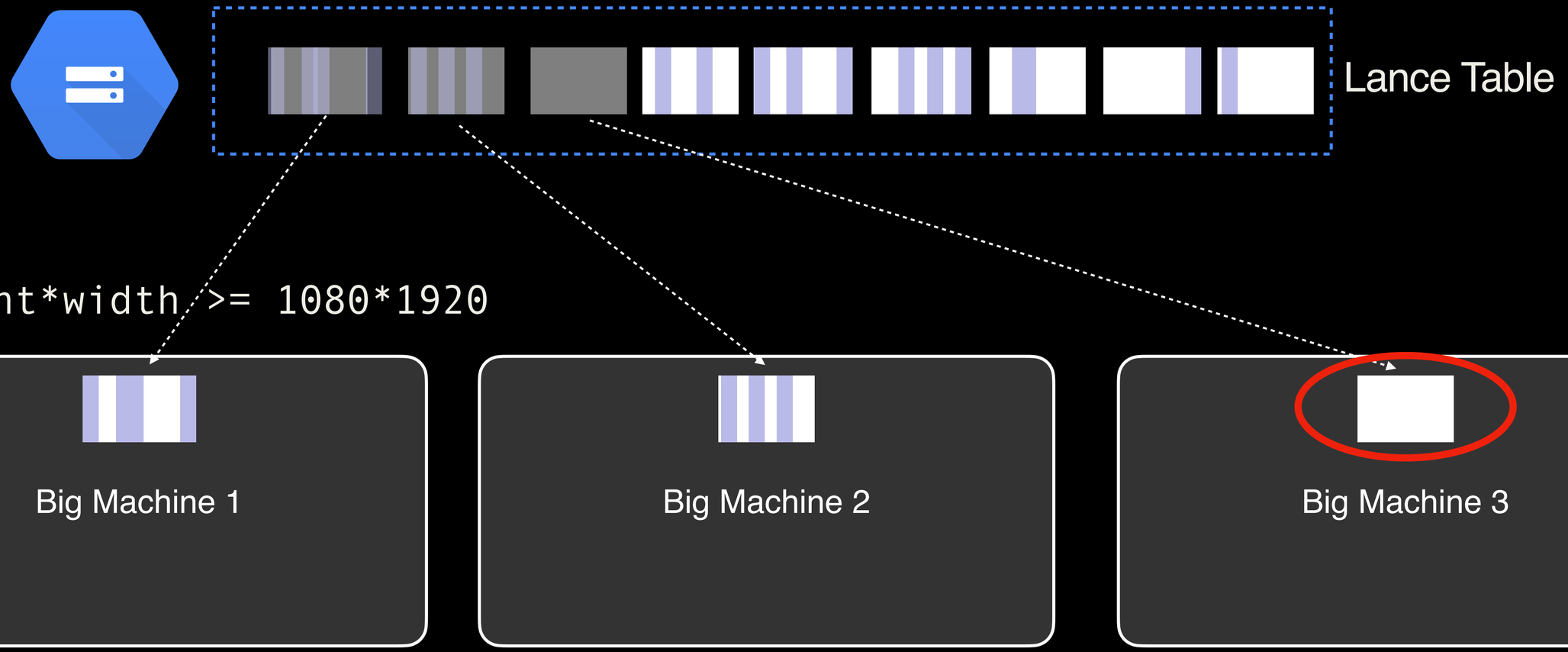
Big Machine 2

Big Machine 3

# Lance Fragment Partitioning



# Lance Fragment Partitioning



# Choose your pointers wisely

POINTERS

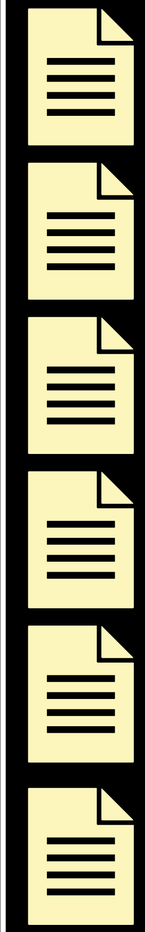
Videos

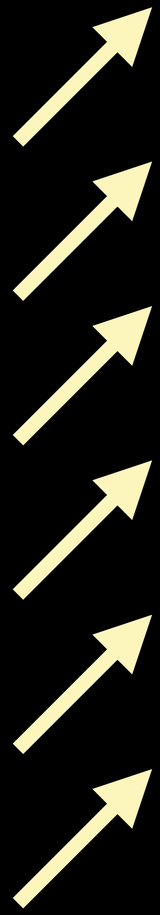
Prompts

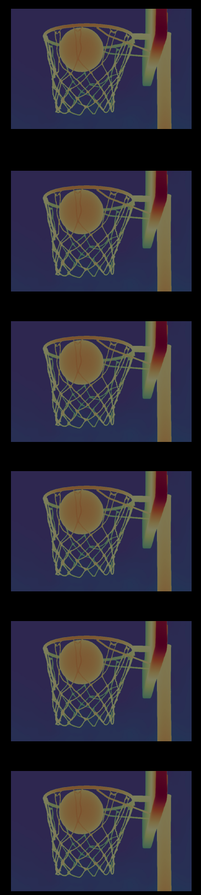
Embeddings

Depth Maps

resolution, fps, duration, …

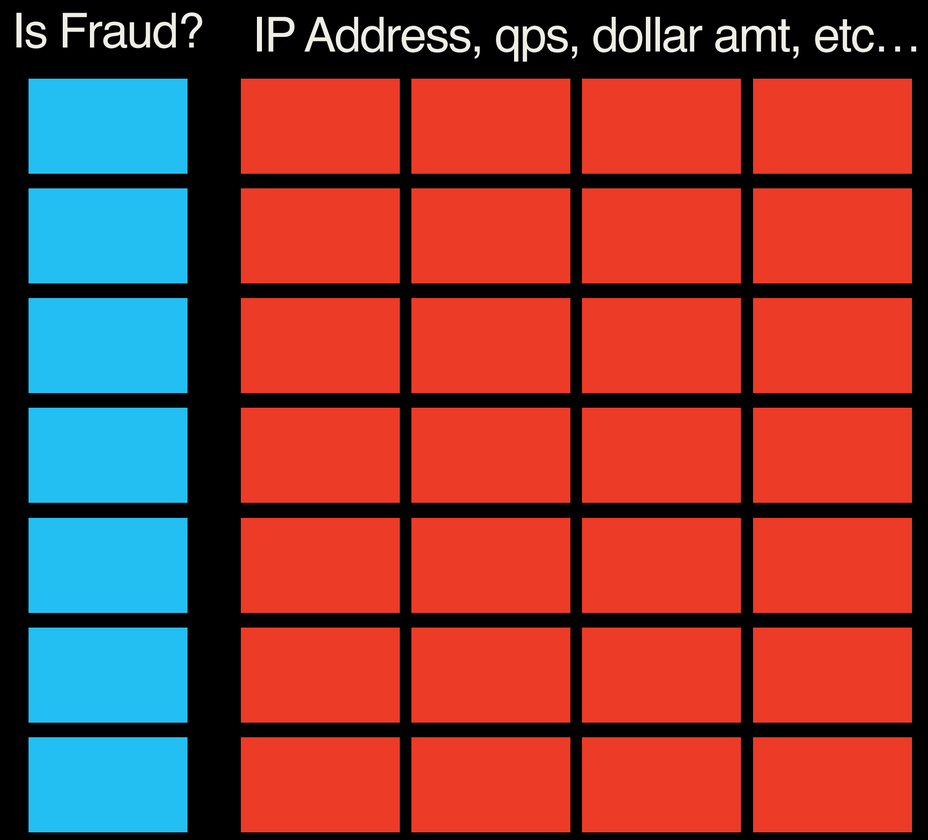






Generative Video


Maximize Read Flexibility

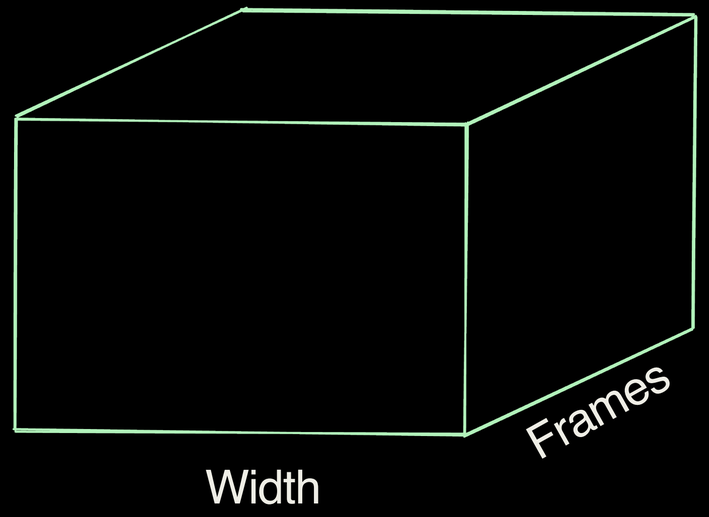




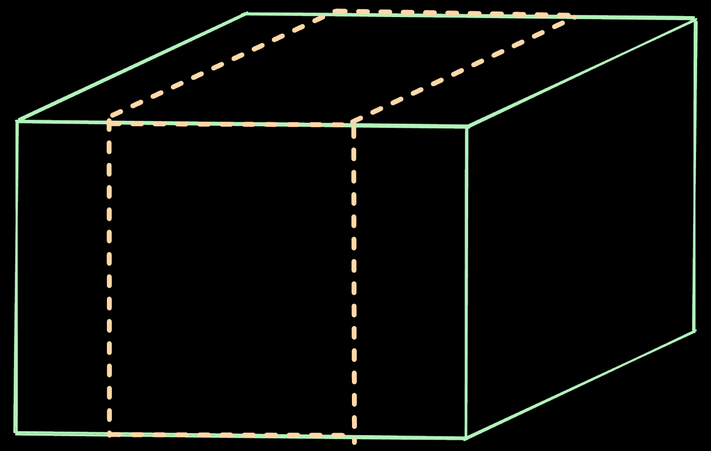
Data

Answers

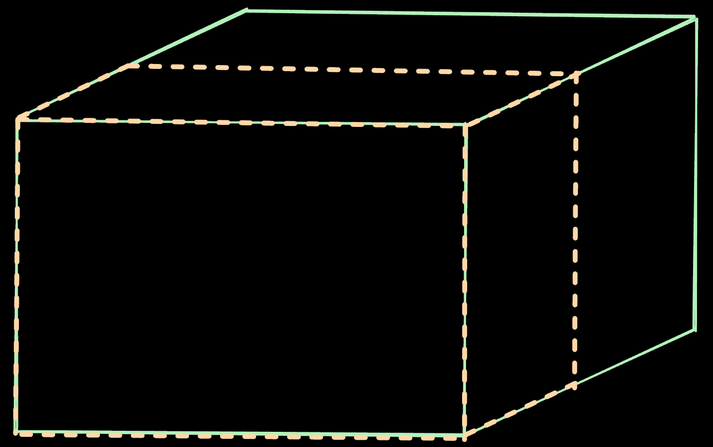
Fraud Detection



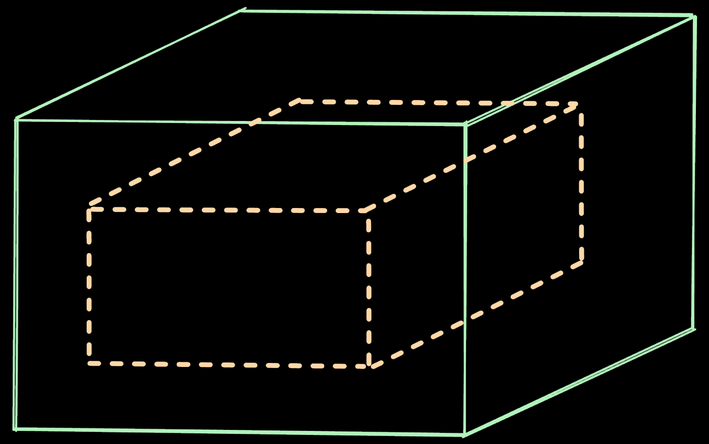

Crop

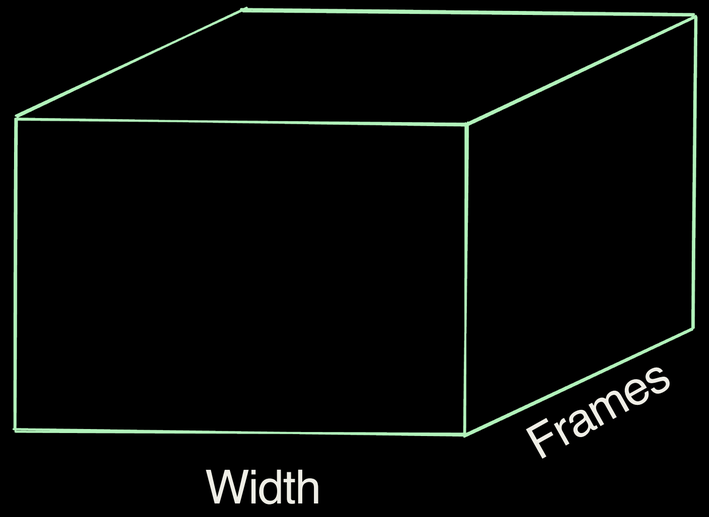


Clip



Resize

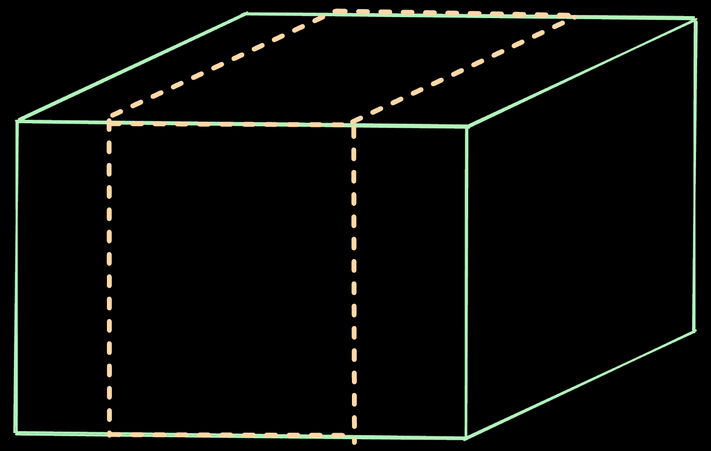


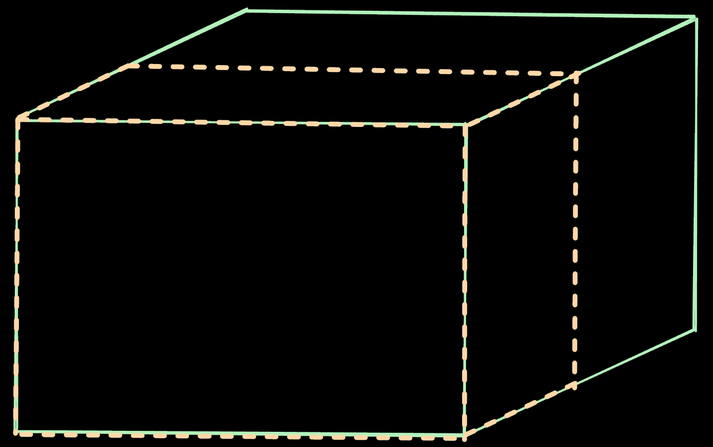


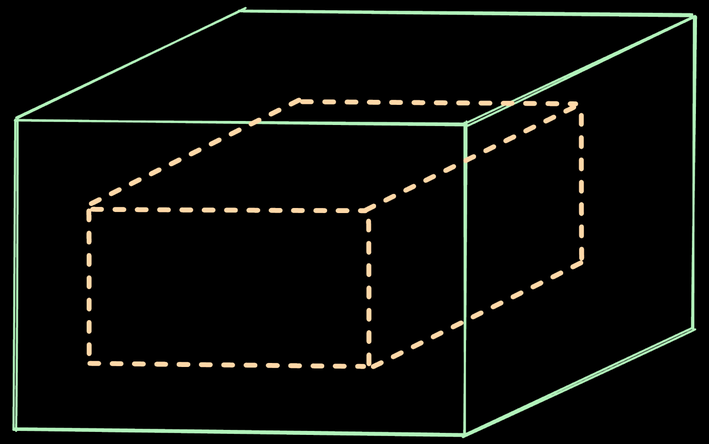

Crop

Clip

Resize







On the Fly

Precompute

rows >> batches

```
for row in rows:
    for processor in processors:
        row = processor(row)

    batcher.consume(row)
    if batcher.full:
        yield batcher.emit_batch()
```

runway

(we’re hiring)



ethanrosenthal.com
@ethanrosenthal.com
@EthanRosenthal