DATA+AI SUMMIT 2022

Photon under the hood

Radical Speed on the Lakehouse

Justin Breese
PM for Photon and SQL, Databricks

Sriram Krishnamurthy
Sr Eng Mgr for Photon, Databricks

ORGANIZED BY databricks

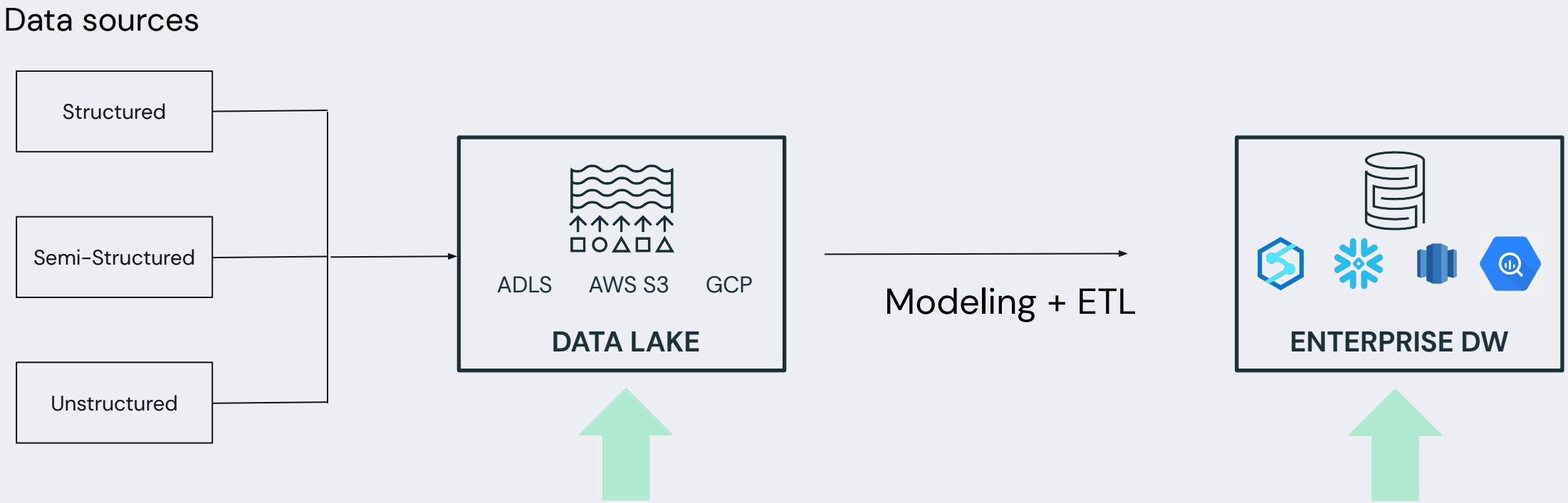
Product Safe Harbor Statement

This information is provided to outline Databricks' general product direction and is for informational purposes only. Customers who purchase Databricks services should make their purchase decisions relying solely upon services, features, and functions that are currently available. Unreleased features or functionality described in forward-looking statements are subject to change at Databricks discretion and may not be delivered as planned or at all.

Motivation

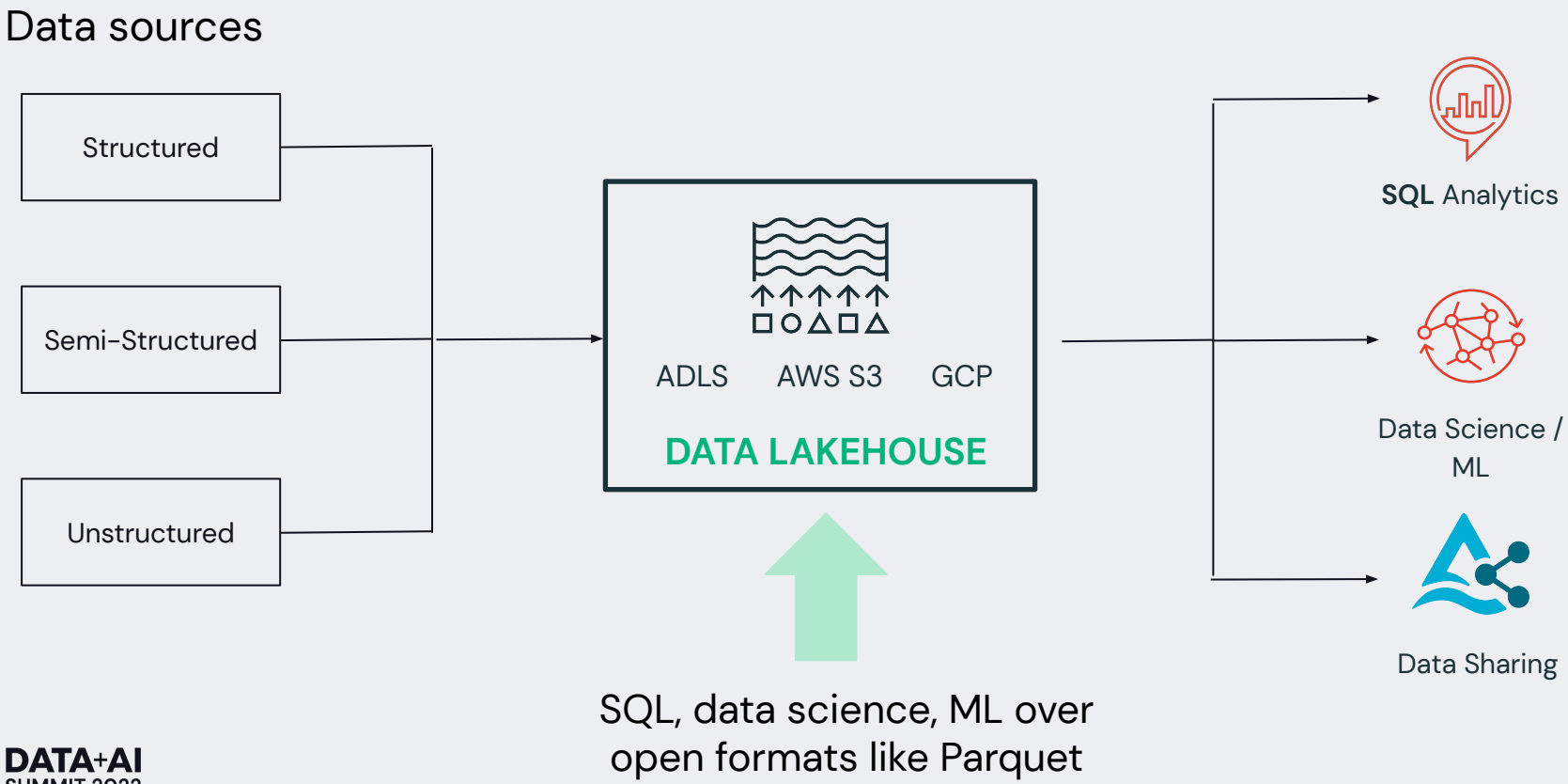
Workload and Storage Trends

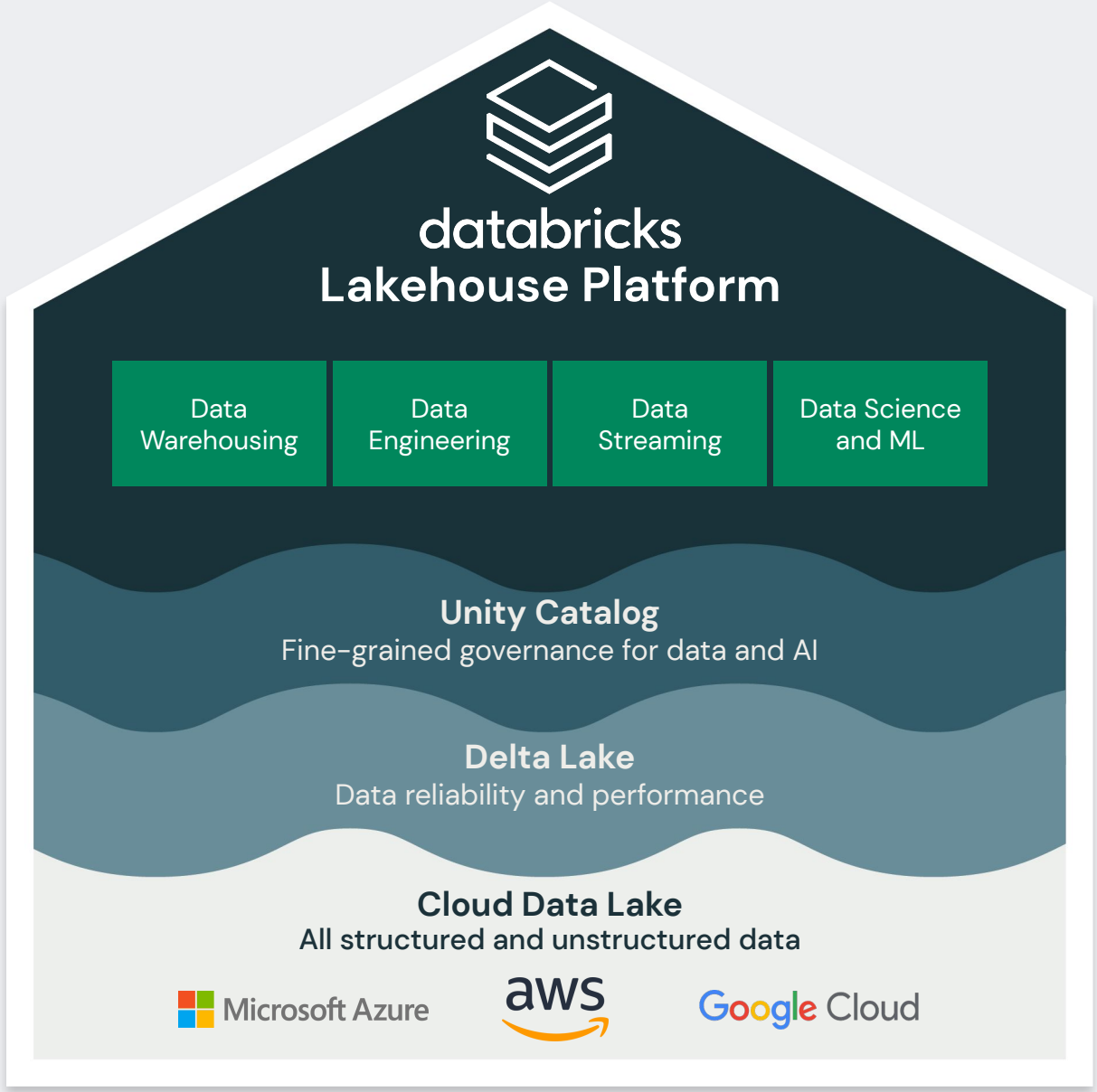
Businesses are moving faster, organizations want to **spend less time in data modeling** and more time querying



Unifying Data Lake and Warehouse

Lakehouse can provide **same or better performance** as proprietary warehouses over **open formats** and **cheap, elastic cloud storage**





Databricks Lakehouse Platform

Simple

Unify your data warehousing and AI use cases on a single platform

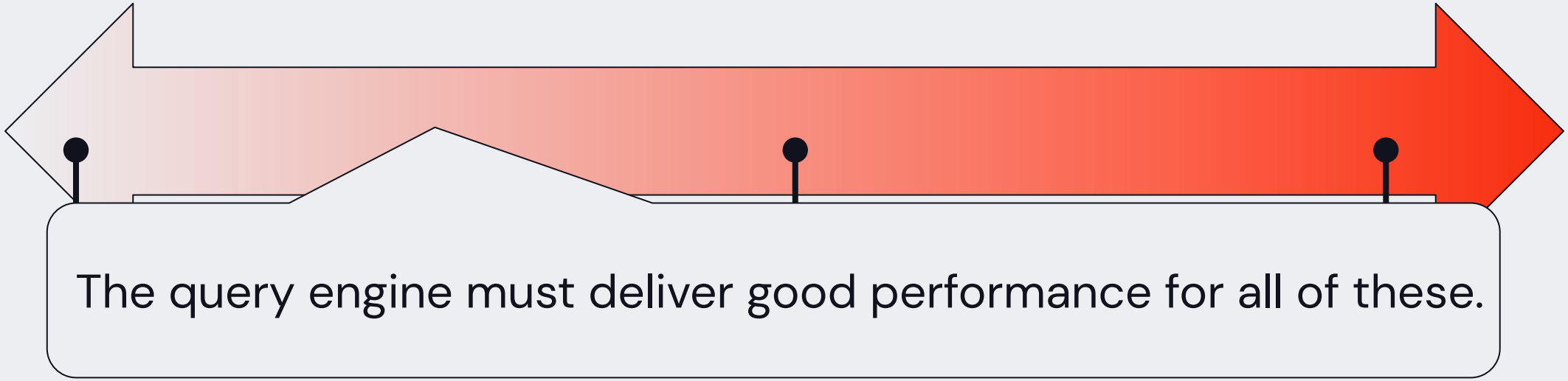
Open

Built on open source and open standards

Multicloud

One consistent data platform across clouds

Query engine challenge: data in many stages



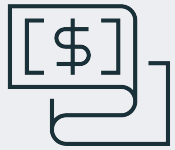
Modeled, structured, cleaned, and ETL'd

Optimized data layout (e.g., large files with clustering), but no normalization

Raw data with small files, no clustering or normalization

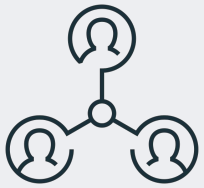
What is Photon?

The next-generation engine for the Lakehouse



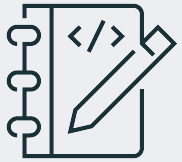
Cheaper and faster

Built from the ground-up for the fastest performance at lower costs, Photon provides up to **80% TCO savings** while accelerating data and analytics workloads - **up to 12x faster**



Built for all use cases

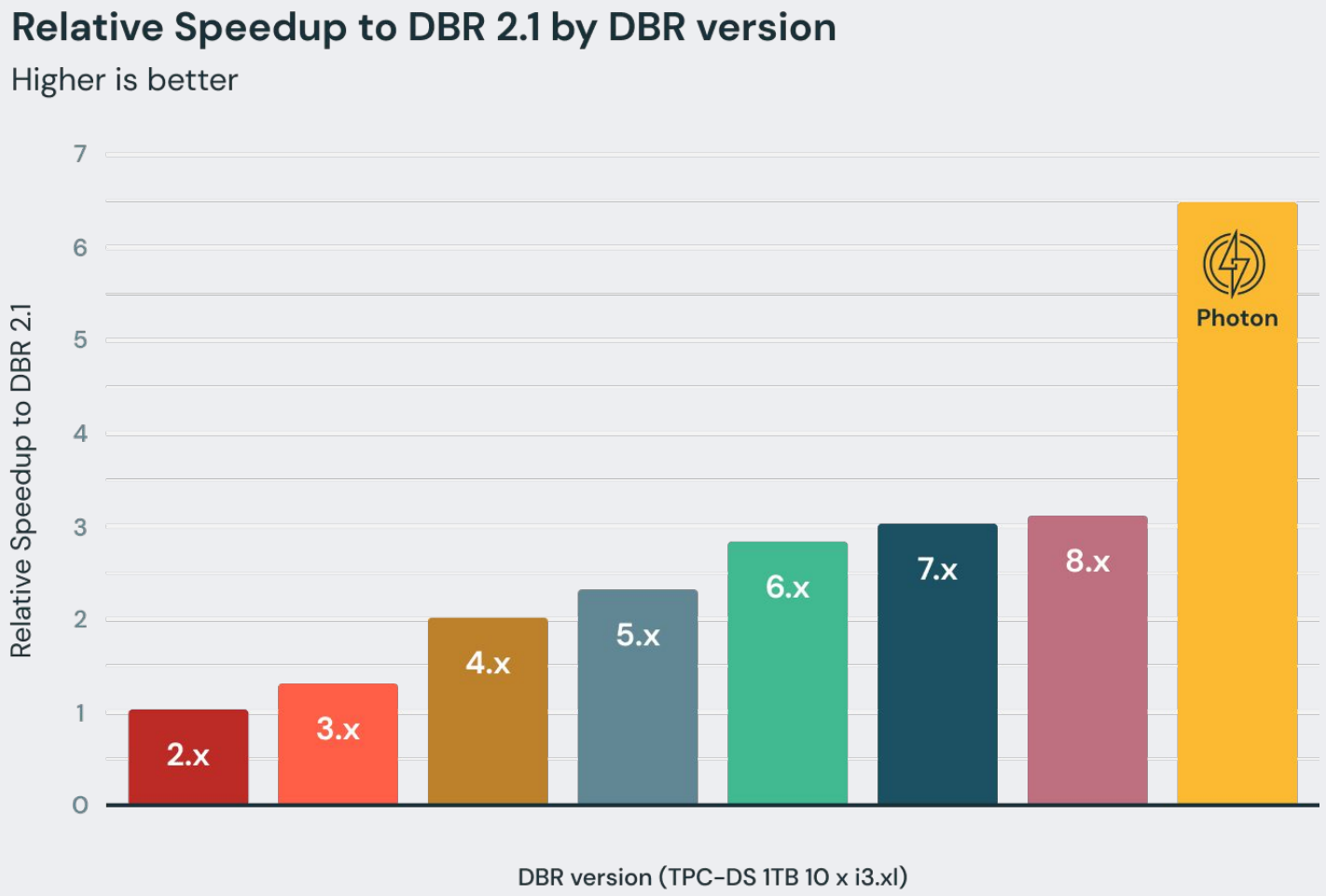
Photon is the first engine that enables data teams to standardize on **one set of APIs for all workloads** - ETL, analytics, and data science - in batch or streaming



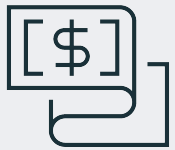
No code changes

Photon is an **ANSI SQL compliant engine** designed to be **compatible with modern Apache Spark™ APIs**, and just works with your existing code - SQL, Python, R, Scala, and Java - **no rewrite required**

Why build a new MPP execution engine?



What are customers saying?



Insurance Company

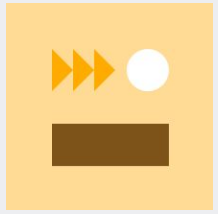
5 minute query on Redshift took 35 seconds on Photon

Retailer

Leveraged Photon with DLT and reduced their latency by 5x

Manufacturer

13x speed-up and a 74% TCO reduction



Media Service

Saved 30% of their total EC2 compute hours in their first night after enabling Photon for all of their ETL

FinTech

5x perf improvement for their streaming workloads after they turned on Photon!

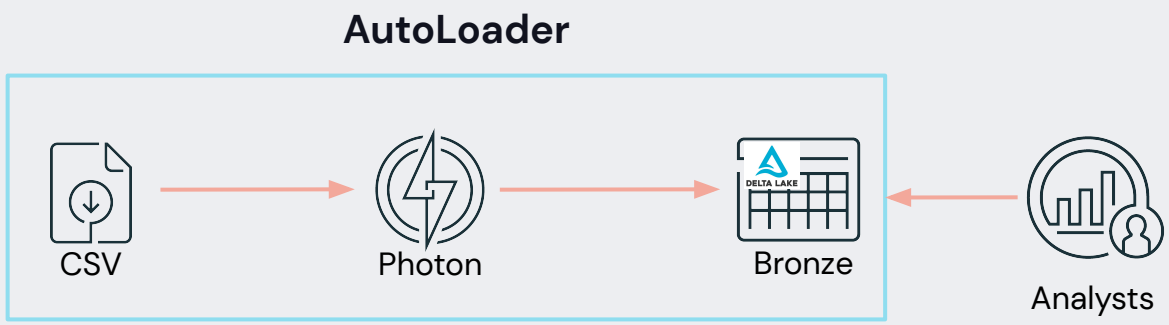
Automaker

Saw a 3x speedup; then enabled Photon for a bunch of jobs

Streaming Ingestion and Photon

AutoLoader is making ingestion easier and faster at a media company

- Marketing analyst needs to generate reports from clickstream data provided by partners
- ETL comprises of 700GB of CSV files, lightweight transformation, and write to a Delta table
- **Result: 19% TCO reduction** and **2.7x latency improvement**

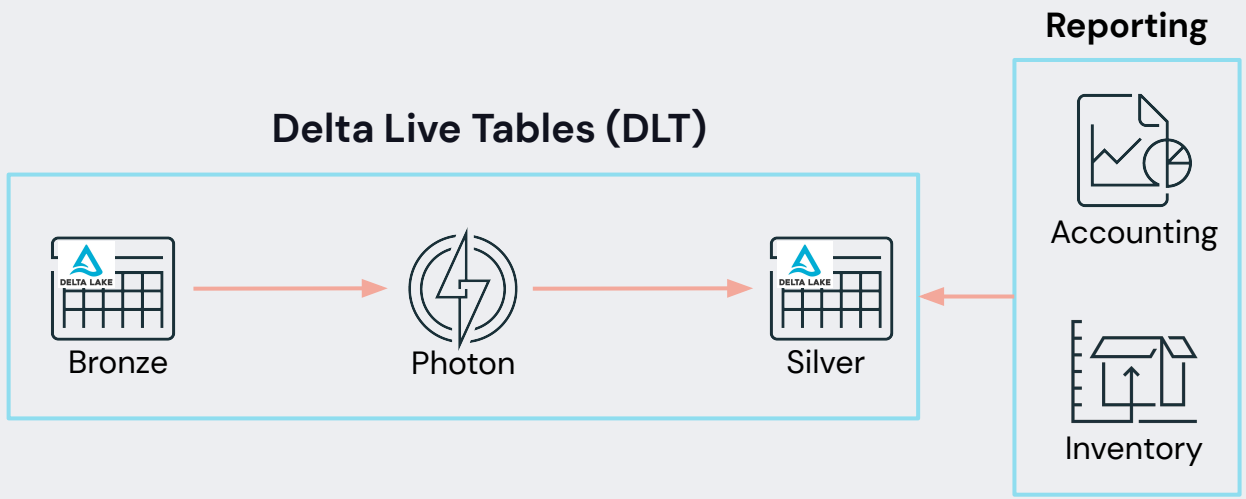


DLT Runtime	Latency (min)	Total Cost
Databricks Runtime (DBR)	7:18	$19.28
Databricks Photon	2:42	$15.60

ETL and Photon

Delta Live Tables (DLT) at a large warehouse retailer

- Business wanted faster access to datasets for inventory and accounting reports
- Delta Lake Merge between bronze and silver tables; LowShuffleMerge and Change Data Feed (CDF)
- **Result: 55% TCO reduction** and **5x latency improvement**

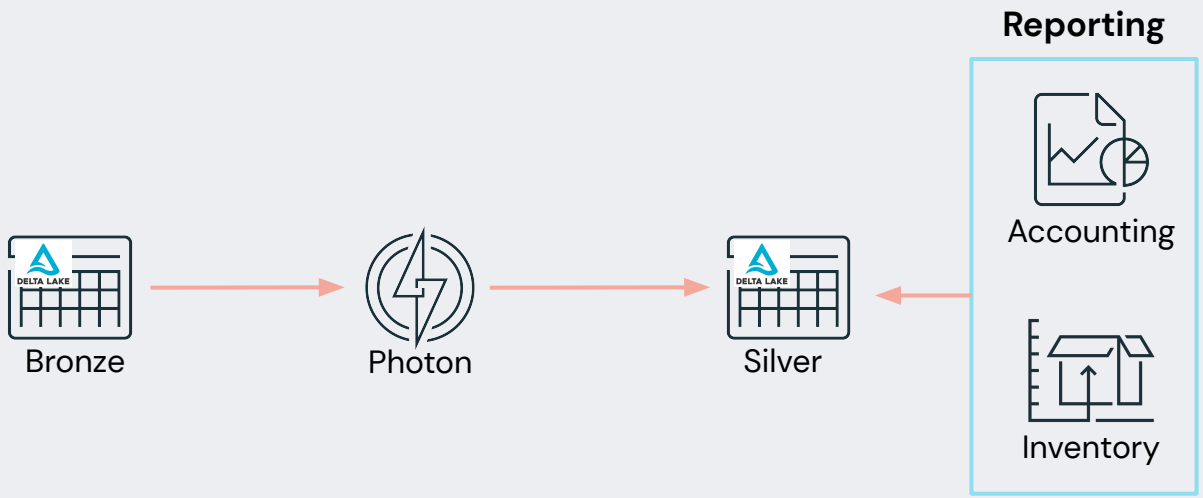


Runtime	Latency (min)	Total Cost
Databricks Runtime (DBR)	15:32	$14.55
Databricks Photon	3:32	$6.51

ETL and Photon

Saving on cost, latency, and accelerating time to Decision

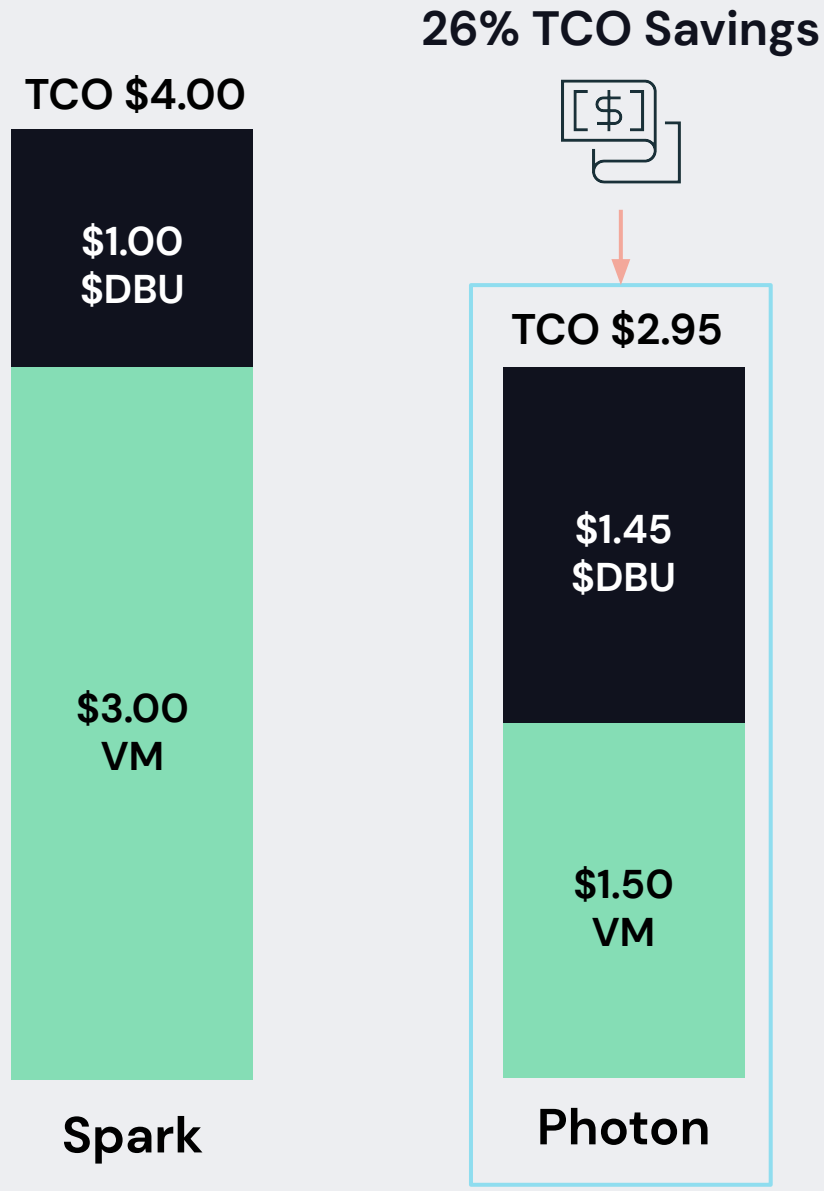
- Business wanted more reliable and fresher data from production sites around the world
- Complicated Architecture with nearly 120 tables in the workflow.
- **Result: 67% TCO reduction** and **11x latency improvement**



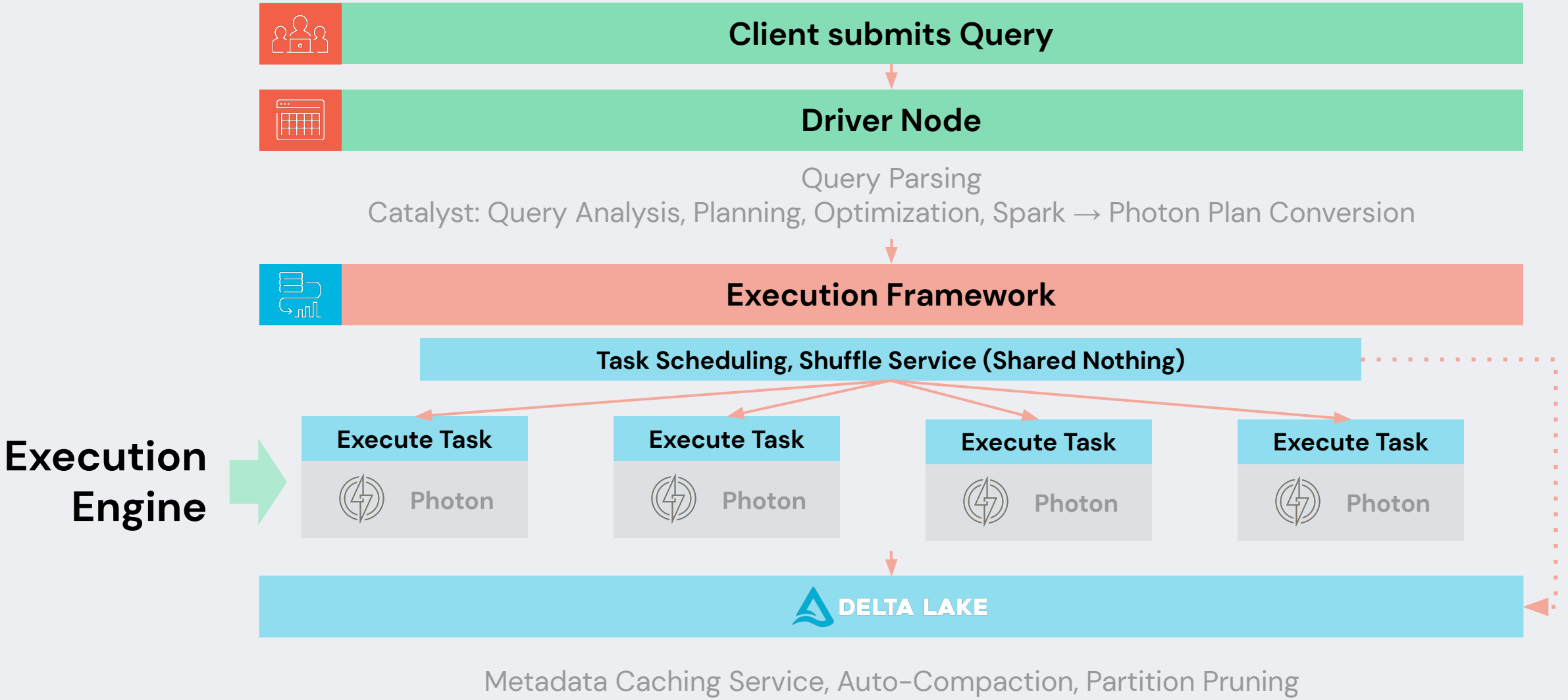
Runtime	Latency (min)	Total Cost
Databricks Runtime (DBR)	330	$265
Databricks Photon	30	$85

Sample TCO breakdown

- Since Photon finishes queries faster, it consumes less compute resources
  - Making it a lower TCO per job
- 2x faster means that you spend half on the infrastructure costs
- 26% TCO reduction due to cloud infrastructure costs



Inner workings


Client submits Query
Driver Node
Query Parsing
Catalyst: Query Analysis, Planning, Optimization, Spark → Photon Plan Conversion
Execution Framework
Task Scheduling, Shuffle Service (Shared Nothing)
Execution Engine
Execute Task
Photon
Execute Task
Photon
Execute Task
Photon
Execute Task
Photon
DELTA LAKE
Metadata Caching Service, Auto-Compaction, Partition Pruning

Key Photon Characteristics

- **Hybrid Photon/Spark Plans**
  - Use Photon when possible, fall back to Spark for unsupported operations
  - Completely transparent to users
- **Native code using off-heap memory**
  - Natural access to memory and intrinsics (no fiddling with Java Unsafe)
  - No JVM GC, large heaps ok
  - No JVM JIT performance cliffs / limitations
- **Fully integrated with Spark's memory manager**
- **Rich per-operator performance metrics**

Design choices to improve performance

Old Engine

Implemented in the JVM

Code generation for performance

Row-oriented execution

New Engine (Photon)

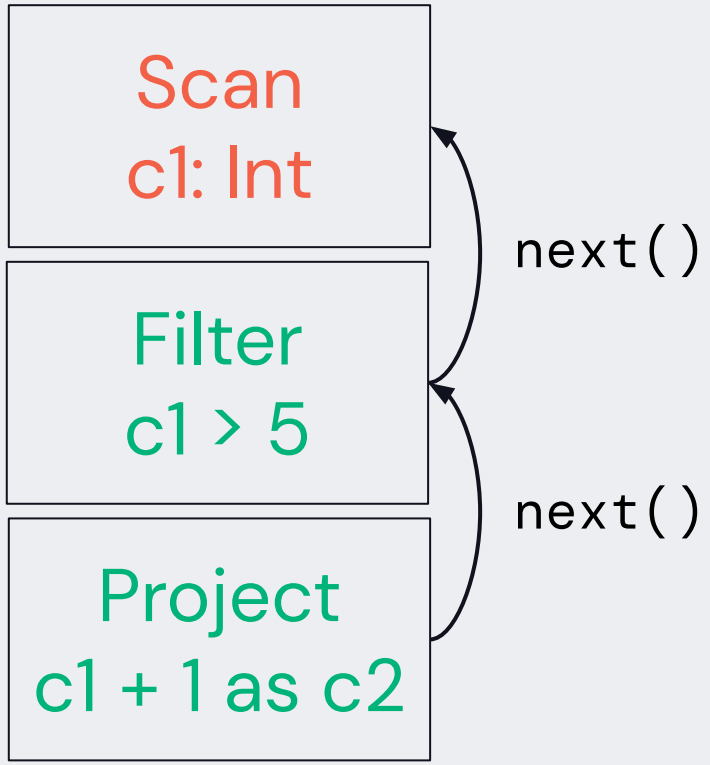
Implemented in Native Code

Interpreted-vectorized execution

Column-oriented execution

Apache Spark today: code generation

Spark 2.x: Code generated query execution model

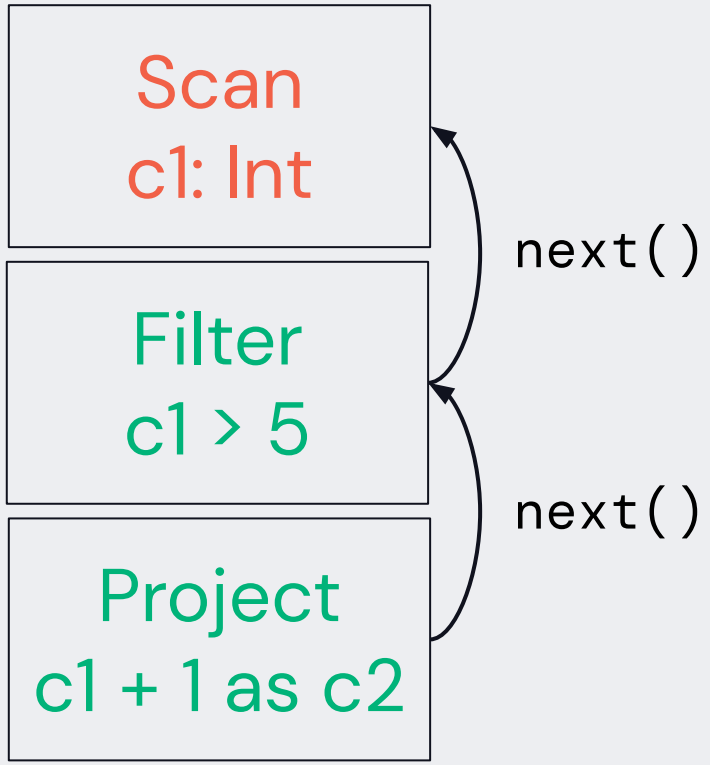



```
def query(): Iterator[Row] = {
  Row r = scanner.next();
  if (r.get('c1') > 5) {
    yield new Row(r.get('c1') + 1);
  }
}
```

Problem: fast, but **very complex to implement and debug!**

Interpreted vectorization: Simpler, but still fast

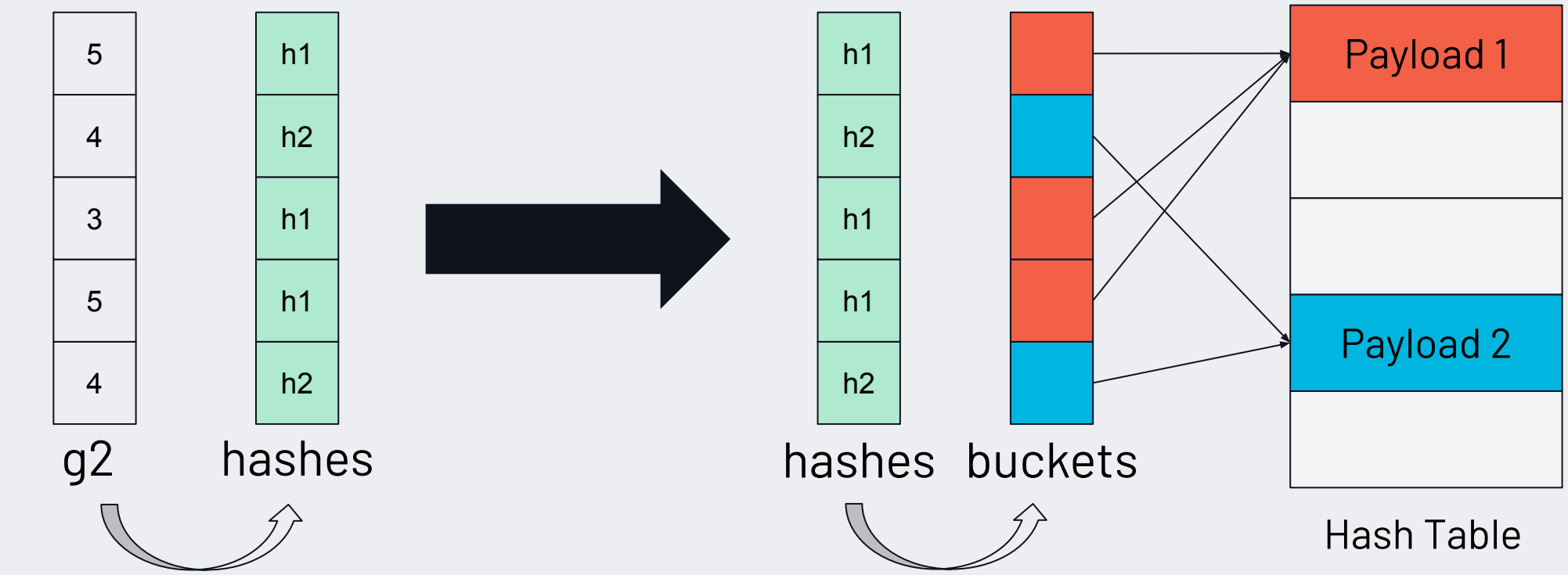
Photon: interpreted vectorized model




```
def next(): ColumnBatch = {
  ColumnBatch b = child.next()
  ColumnBatch out = new ColumnBatch(b)
  for (int i = 0; i < b.num_rows; ++i) {
    out['c1'][i] = b['c1'][i] + 1;
  }
  return out;
}
```

Simplicity of interpretation, speed of code gen (highly optimizable loops)

Example benefit: Grouping Hash Aggregation

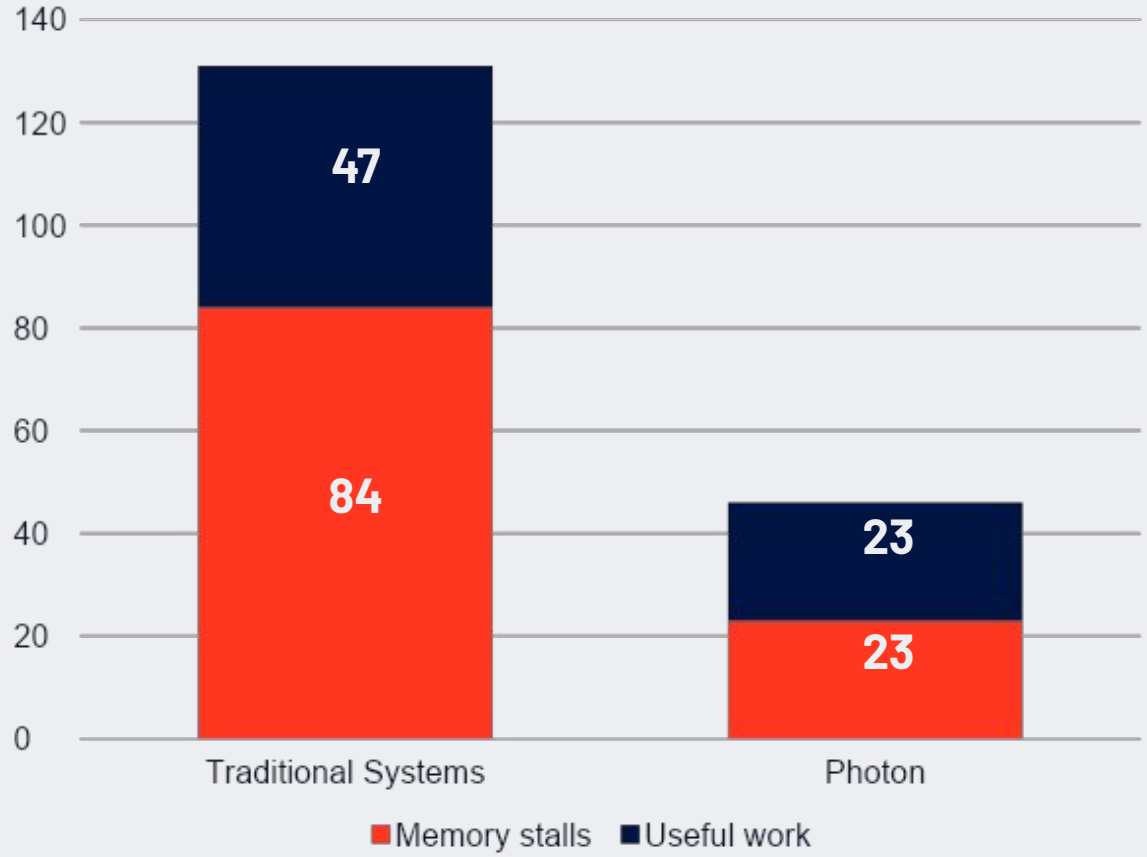


Vectorized Hashing
(SIMD)

Vectorized Probing
(Instruction/Memory Parallelism)

Exploiting Instruction Level Parallelism

- Tight kernels with independent loads → HW loads multiple memory addresses in parallel
- Minimize TLB misses with huge pages



Benefits of vectorized execution vs. code-gen

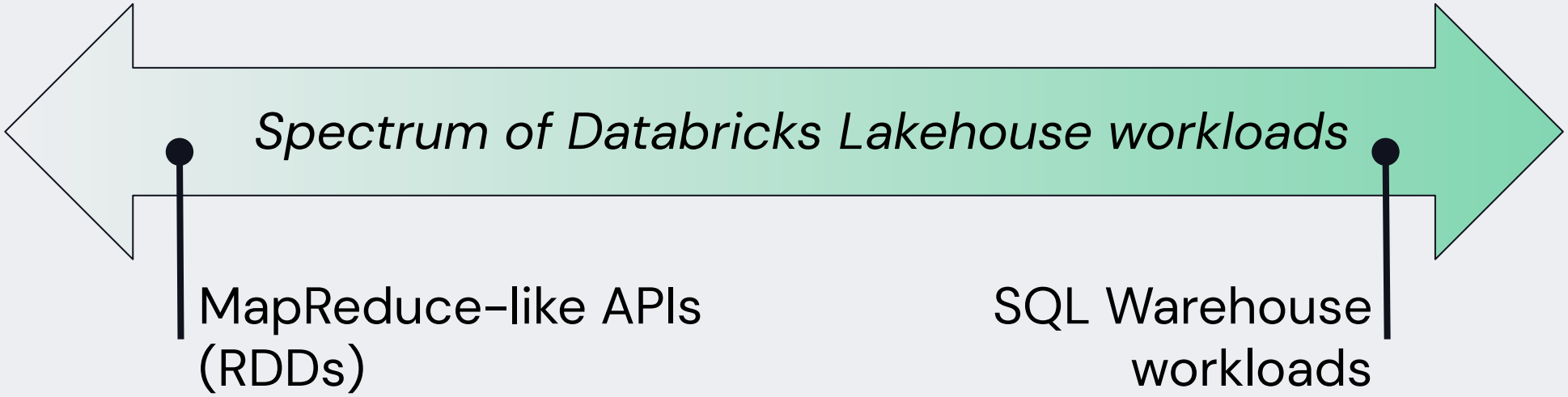
Advantage 1: Human Factor. Code is “just C++” with templating: easy to read, optimize, debug (code generation: code that generates code)

Advantage 2: Operators maintain abstraction boundaries → richer metrics (code generation: everything “squished” together)

Advantage 3: Easy to adapt to diverse Lakehouse data by choosing executed code on the fly (code generation: need to recompile everything)

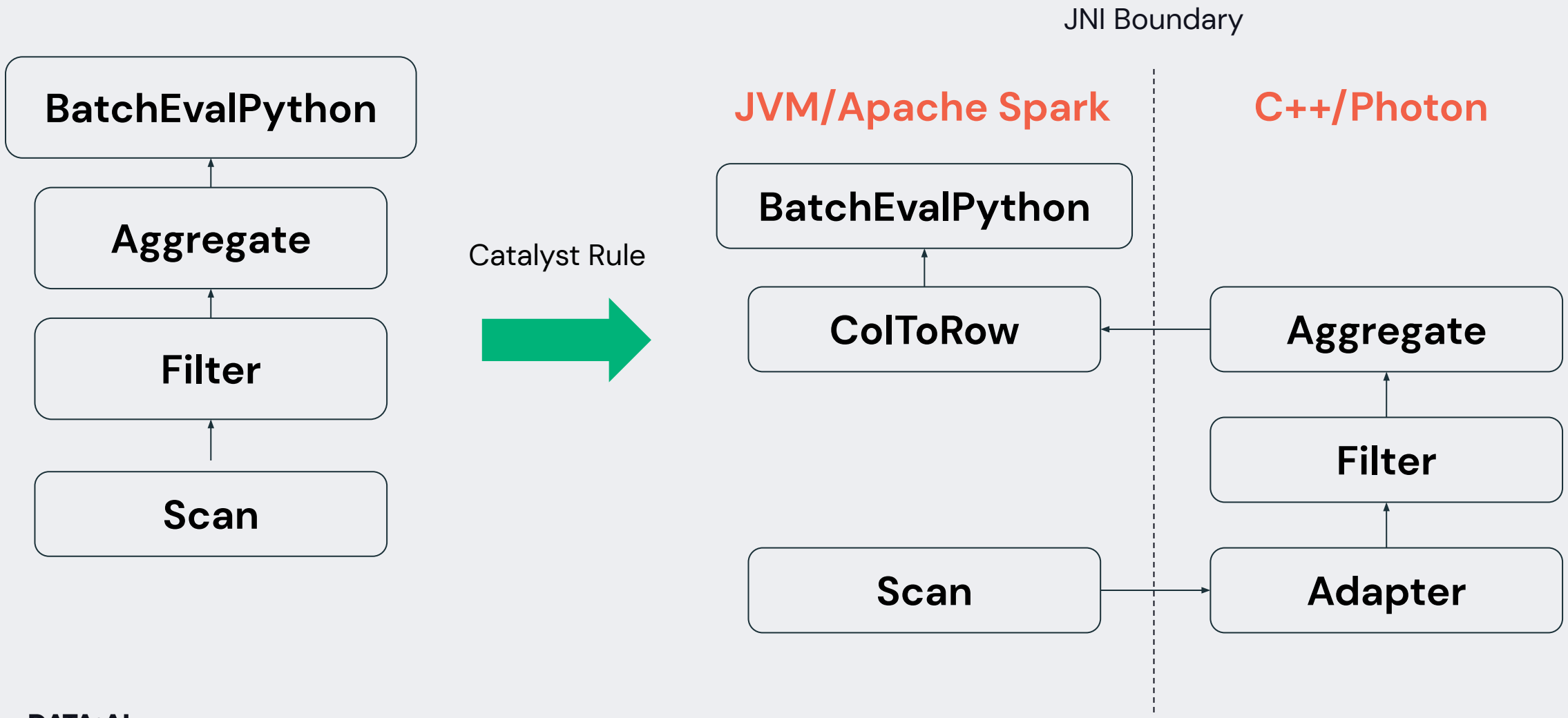
Why integrate with the existing engine at all?

- Support **existing Lakehouse workloads** built using DataFrame APIs



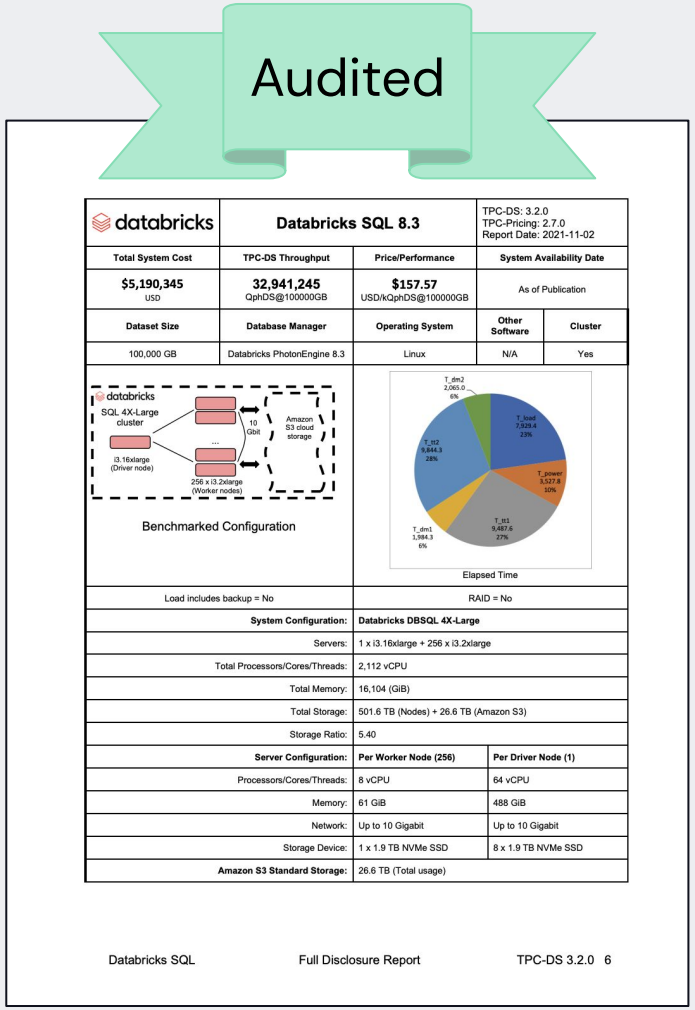
- **Incremental rollout** to get real-world experience ASAP

Hybrid Photon/Apache Spark plans



Result: TPC-DS World Record

- Performance metric uses mixed workload
  - Data loading
  - Power run (run queries back-to-back)
  - Concurrent query run
  - Data maintenance run (INSERT/DELETE)
- First audited cloud-storage based run
- Databricks + Photon **beat previous performance record with 10% less TCO**



Current and the Future

Photon recent releases

Dec 2021

- Photon on GCP (Public Preview)
- Additional AWS instance types (r5d/n, i3en, m5d/n)

Jan 2022

- Better query plan display

Feb 2022

- Additional GCP instances (n2-standard)
- More math and string functions (GA)
- Array/Map data types with basic function support

Mar 2022

- Sort/Window (Private Preview)
- Structured Streaming (Private Preview)
- Adapter for CSV, AVRO, and JSON (Private Preview)

Apr 2022

- Low Shuffle Merge (GA)
- Delta/Parquet scan performance (Private Preview)
- Additional array aggregate functions (GA)
- AWS Graviton instances

Battle tested: processed **Billions of queries** and **exabytes of data** from **large set of customers,** providing up to order-of-magnitude performance improvements

General Availability of Photon for Workspaces in the next few weeks.

Vectorized Sort

Accelerated Window Functions

Data Source Expansion

Photon coverage (DBR 11.1)

Data types

- ✅ Byte/Short/Int/Long
- ✅ Boolean
- ✅ String/Binary
- ✅ Decimal
- ✅ Float/Double
- ✅ Date/Timestamp
- ✅ Struct
- ✅ Array
- ✅ Map

Operators

- ✅ Scan, Filter, Project
- ✅ Hash Aggregate/Join/Shuffle
- ✅ Nested-Loop Join
- ✅ Null-Aware Anti Join
- ✅ Union, Expand, ScalarSubquery
- ✅ Delta/Parquet Write Sink
- ✅ Sort
- ✅ Window Function

- 🚫 RDD Scan

Expressions

- ✅ Comparison / Logic
- ✅ Arithmetic / Math (most)
- ✅ Conditional (IF, CASE, etc.)
- ✅ String (common ones)
- ✅ Casts
- ✅ Aggregates(most common ones)
- ✅ Date/Timestamp

- 🚫 Scala UDFs

Photon—what's coming next?

Best in class analytics

- Low Latency improvements for BI workloads
- Faster window functions and Top K queries
- Python and Pandas UDF support

Easier and faster ingestion

- Query from Kafka, Kinesis, EventHub with enhanced deduplication support
- Native JSON Scan

Continued Price/Performance Focus

- Native Vectorized Scans for Delta/Parquet
- Shuffle Improvements

How to enable Photon?

Use 11.0 and select the ‘Use Photon Acceleration’ box

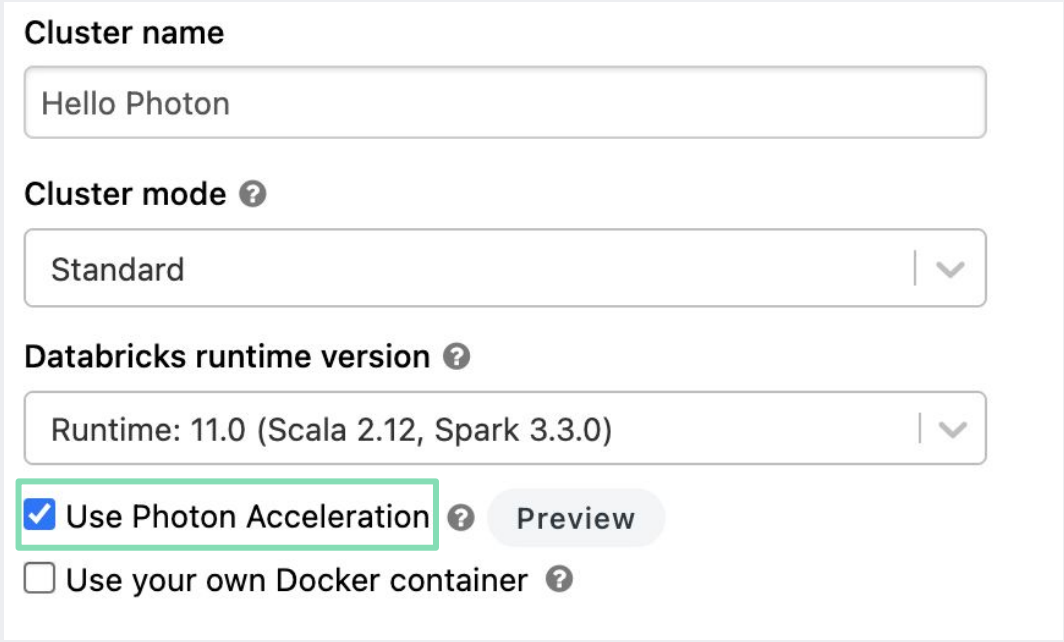
Terraform

```
data "databricks_node_type" "photon" {
  photon_worker_capable = true
}

data "databricks_spark_version" "photon_lts"
{
  long_term_support = true
  photon            = true
}

resource "databricks_cluster" "photon" {
  cluster_name            = "Photon Cluster"
  spark_version           =
data.databricks_spark_version.photon_lts.id
  node_type_id            =
data.databricks_node_type.photon.id
  autotermination_minutes = 20
  num_workers             = 3
 }
```

Cluster manager



Key Takeaways

Next generation execution engine for the Lakehouse

- Cheaper and faster; save up to 80% and up to 12x better Price/Perf
- Built for all use cases; BI, Stream, Ingestion and ETL
- No code changes necessary

Check the “Use Photon Accelerate” box today and speed up your workloads

DATA+AI SUMMIT 2022

Thank you

Justin Breese
PM for Photon and SQL, Databricks

Sriram Krishnamurthy
Sr Eng Mgr for Photon, Databricks