INDIAN DATA CLUB | CODE BASICS

Performance Optimization in Databricks

The Day 10 Challenge: From Query Analysis to Sub-Second Latency

Objective: Master the optimization stack—Partitioning, Z-Ordering, and Benchmarking—to transform sluggish queries into high-performance operations.

The Optimization Roadmap

1. Diagnose

Analyze query plans to understand execution costs.

2. Structure

Partition large tables to minimize file scanning.

3. Refine

Apply Z-ORDER clustering to colocate related data.

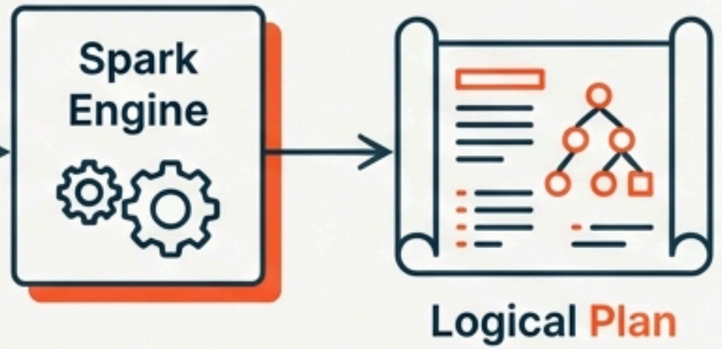
4. Validate

Benchmark improvements to quantify speed gains.

Step 1: Diagnosing the Query

Before optimizing, we must understand how Spark interprets the request.

```
# Explain query
spark.sql('SELECT * FROM silver.events WHERE
    event_type=\'purchase\'').explain(True)
```



Reveals Parsed and Analyzed Logical Plans

Reading the Logical Plan

```
== Parsed Logical Plan ==
'Project [*]
 +- 'Filter ('event_type = purchase)
    +- 'UnresolvedRelation [silver, events], [], false

== Analyzed Logical Plan ==
Project [event_time#19820, event_type#19821, product_id#19822...]
+- Filter (event_type#19821 = purchase)
    +- SubqueryAlias workspace.silver.events
```

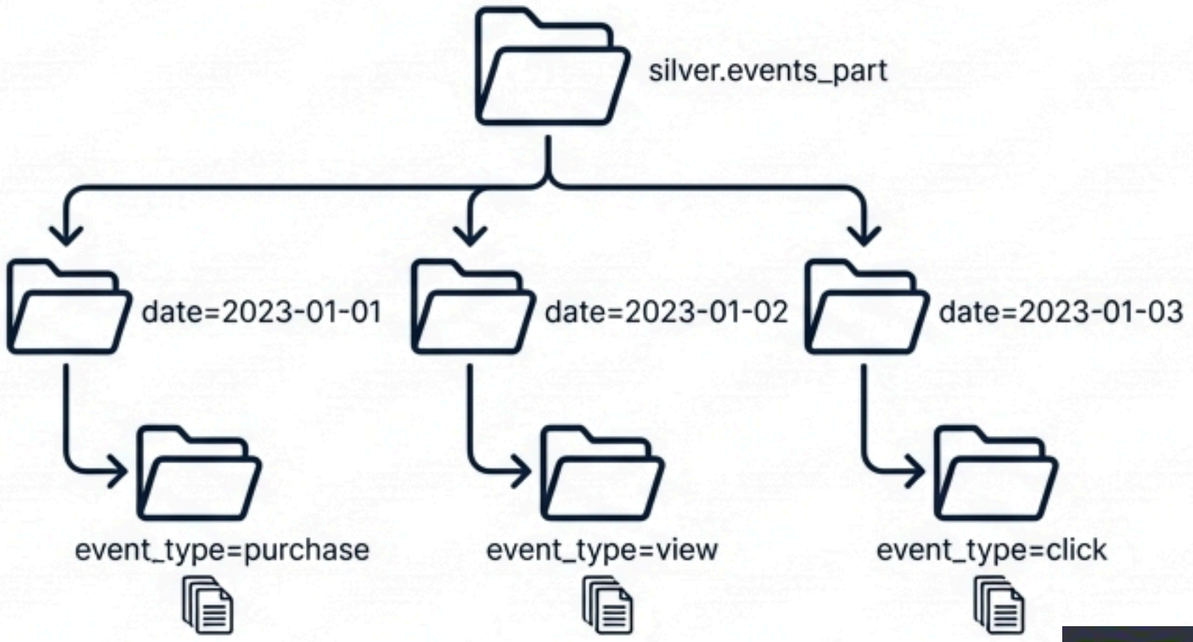
Spark recognizes the filter intent, but without physical optimization, this may still result in a full table scan.

Step 2: Structural Optimization via Partitioning

Coarse-Grained Data Skipping

Partitioning physically separates files into directories based on column values. This allows the engine to to completely ignore folders that don't match the query predicate.

```
CREATE TABLE IF NOT EXISTS silver.events_part
 USING DELTA
 PARTITIONED BY (event_date, event_type)
AS SELECT * FROM silver.events
```

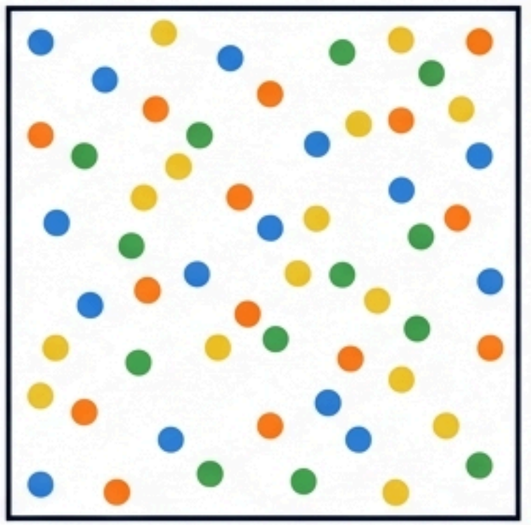


Step 3: High-Fidelity Indexing with Z-Order

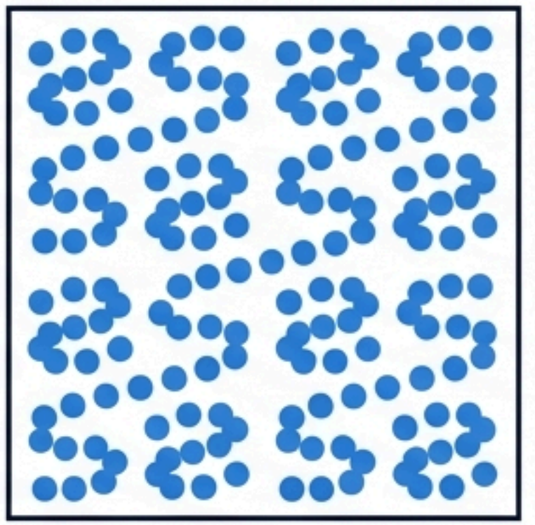
Fine-Grained Data Skipping

Z-Ordering maps multidimensional data points to a single dimension, preserving locality. It ensures similar data (like specific User IDs) is stored close together within the same file.

```
%sql
OPTIMIZE silver.events_part
ZORDER BY (user_id, product_id)
```



Random Distribution



Z-Ordered Clustering

Technical Editorial

Verifying the Optimization History

version	timestamp	operation	operationParameters
1	[Time]	OPTIMIZE	{"predicate":"[]", "zOrderBy":"["user_id", "product_id"]"...}
0	[Time]	CREATE TABLE AS SELECT	{"partitionBy": "["event_date", "event_type"]"...}

The Delta Table History structure in the tables usually dewrinted in **JetBrains Mono**.
The transaction log confirms the table was rewritten with Z-Order clustering on `user_id` and `product_id`.

Step 4: Benchmarking the Improvements

Using a point-lookup test to validate Z-Order efficiency.

```
# Benchmark
import time
start = time.time()
spark.sql("SELECT * FROM silver.events WHERE user_id=12345").count()
print(f"Time: {time.time()-start:.2f}s")
```

This specific lookup benefits heavily from Z-Ordering, as the engine can skip files that don't contain this user.

Technical Editorial

The Result: Sub-Second Latency

0.93s

Execution Time

```
Time: 0.93s
```

Note: Iterative queries on a warmed cache would yield even faster results. This benchmark reflects the optimized physical layout performance.

Technical Editorial

Quality Assurance & Data Integrity

silver.events (Original)

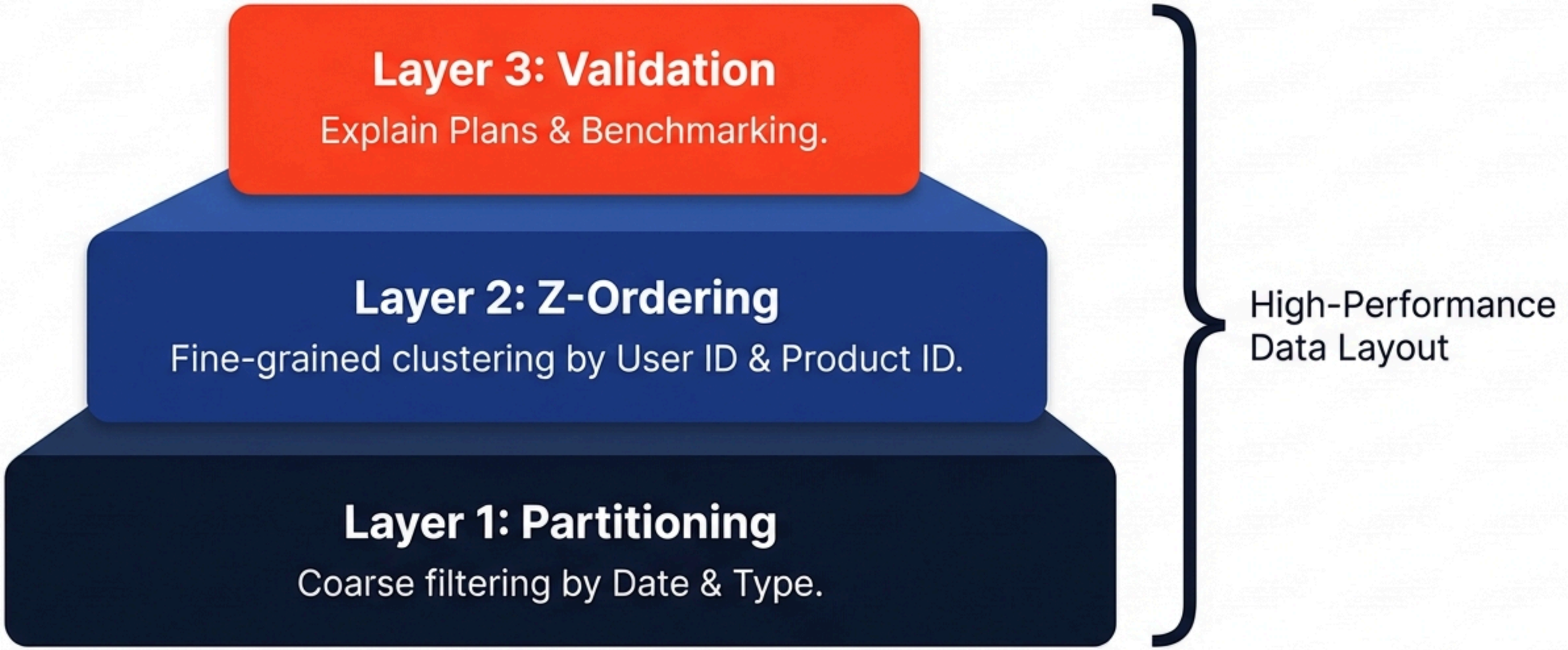
category_id	category_code	brand	price	user_id	user_session
5556318826…	electronics.smartphone	apple	1363.95	543296136	b6c1c551-d7cb-406c-a943-5c58dc0db10d
5610928667…	computers.desktop	hp	967.82	546350875	cd00b163-df39-4d2b-b9a5-d97185772e05
5652528027…	appael.shoes	hayaki	57.4	572425877	a420226-ea3f-54bc-8212-80779208aae1

silver.events_part (Optimized)

event_time	event_type	product_id	category_id	...	brand	price	user_id
2019-11-07T11:46:32.000…	cart	1004856	20530135556318826…	...	samsung	131.66	563251536
2019-11-07T11:55:08.000…	cart	1004856	20530135556318826…	...	samsung	131.66	538796194
2019-11-07T11:17:11.000…	cart	1004856	20530135556318826…	...	samsung	131.66	562495527

While the physical storage layout has changed completely,
the logical data schema and values remain 100% intact.

The Optimization Stack Summary



INDIAN DATA CLUB

CODE BASICS

Mission Accomplished

Databricks Performance Optimization Challenge.

We successfully diagnosed, restructured, and accelerated the silver.events dataset.