OSA CON 24

AERODYNAMIC DATA MODELS: FLYING FAST AT SCALE WITH DUCKDB & CLICKHOUSE

Mike Driscoll
Co-Founder & CEO at Rill Data

November 19-21, 2024

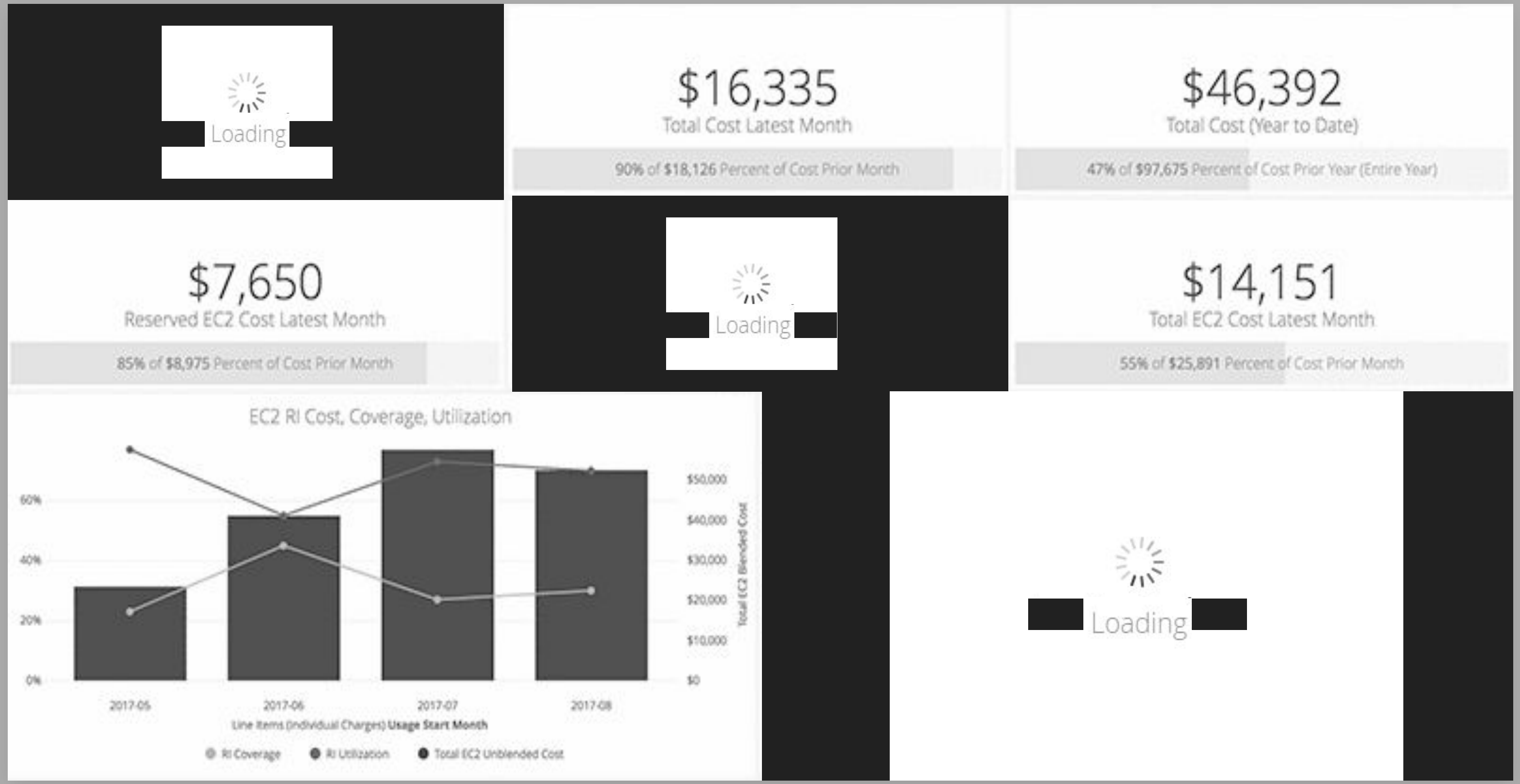
Rill is an operational BI platform

Proven tech: Spun core IP out of Snap (ex-Metamarkets)

Proven scale: Managing 100TB+ of data, 1000s of users, millions of queries

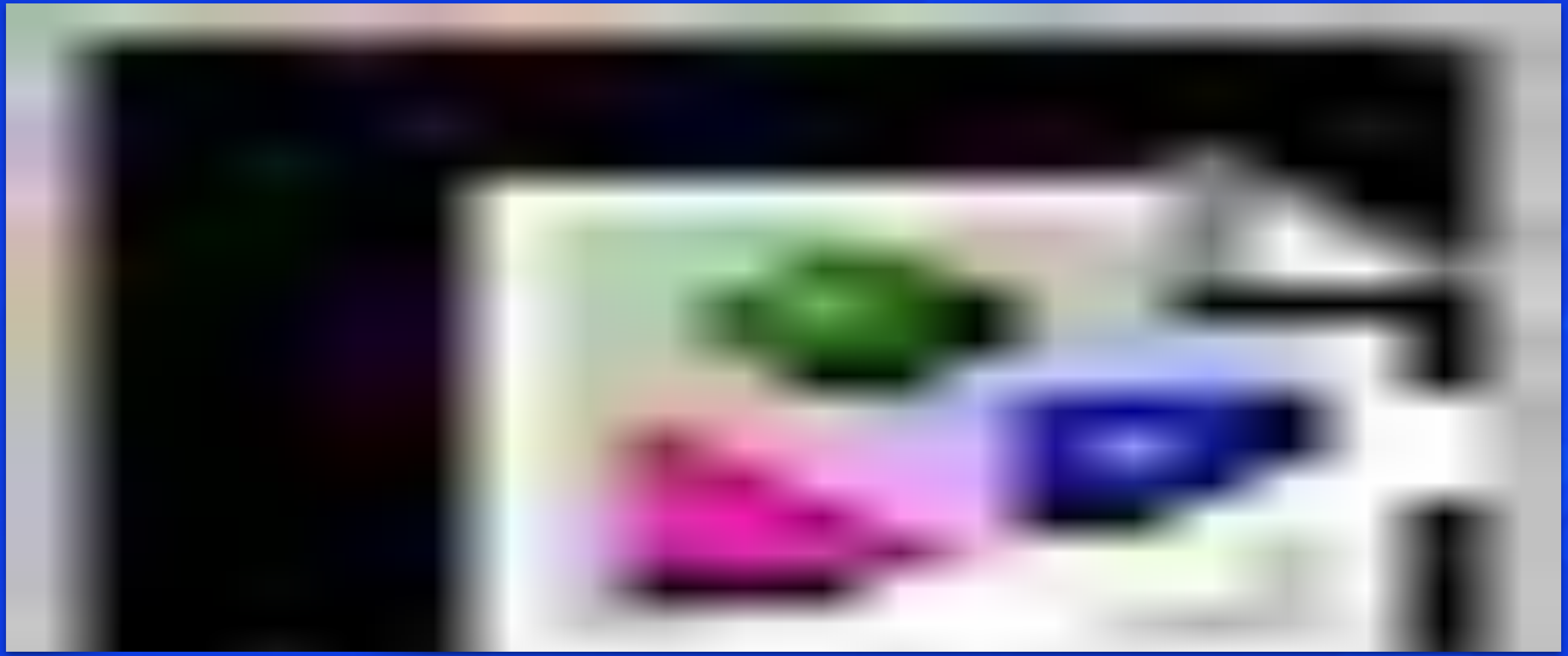
Proven impact: Trusted by leading media, ecommerce, and tech platforms

Traditional BI is *slow & rigid*


Loading
$16,335
Total Cost Latest Month
90% of $18,126 Percent of Cost Prior Month
$46,392
Total Cost (Year to Date)
47% of $97,675 Percent of Cost Prior Year (Entire Year)
$7,650
Reserved EC2 Cost Latest Month
85% of $8,975 Percent of Cost Prior Month
Loading
$14,151
Total EC2 Cost Latest Month
55% of $25,891 Percent of Cost Prior Month
EC2 RI Cost, Coverage, Utilization
60%
40%
20%
0%
$50,000
$40,000
$30,000
$20,000
$10,000
$0
Total EC2 Blended Cost
2017-05
2017-06
2017-07
2017-08
Line Items (Individual Charges) Usage Start Month
● RI Coverage
● RI Utilization
● Total EC2 Unblended Cost
Loading

- Canned reports
- Limited exploration
- Works with slow data warehouses

Rill is *fast & flexible*



- Ad hoc, exploratory analysis
- High dimensionality & cardinality
- Requires a high performance DB

Why is Rill's BI tool different?

10X Faster Dashboards

- DuckDB & ClickHouse-powered, so queries return instantly

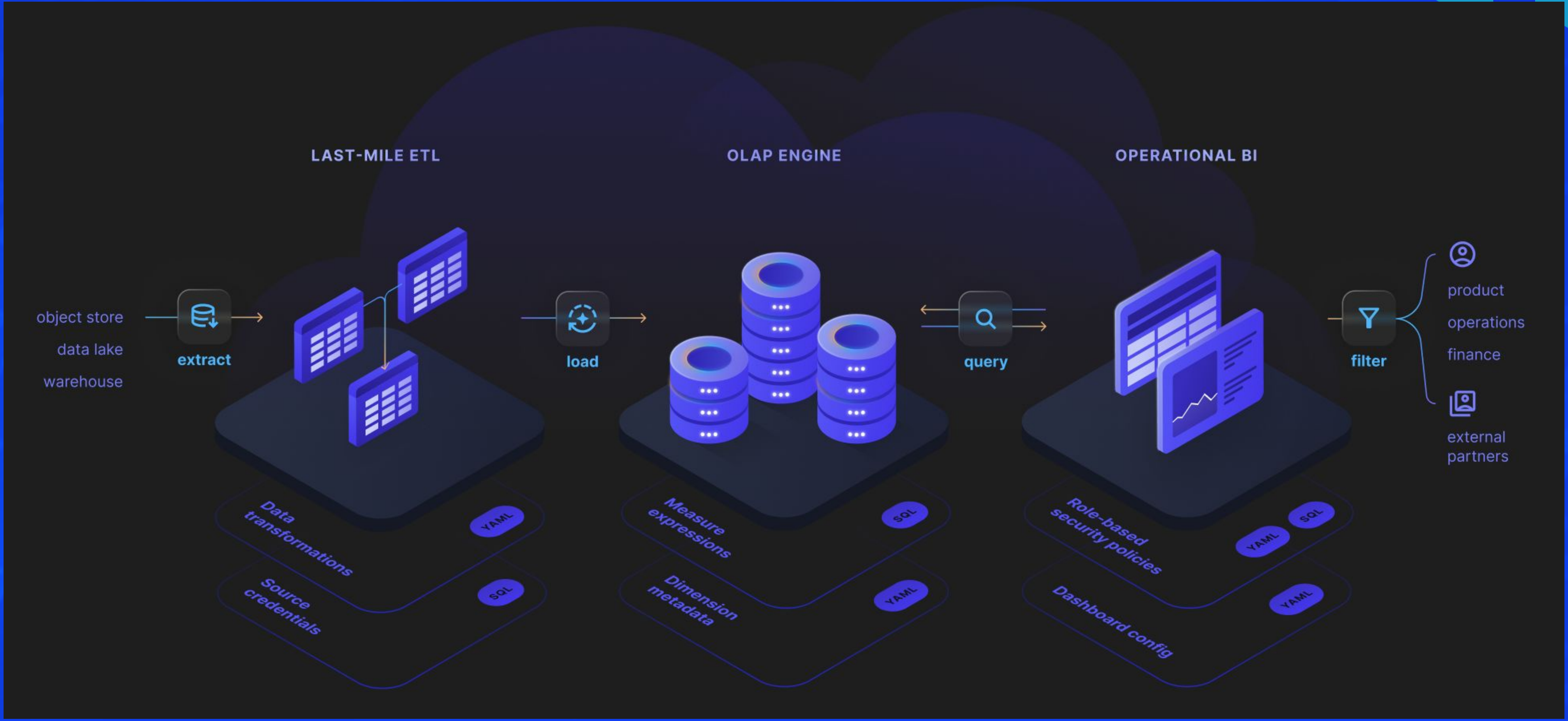
BI-as-Code

- Develop locally, deploy globally with benefits of Git workflows

Metrics-first Philosophy

- Users declare metrics with SQL expressions, Rill *auto-generates* dashboards

3-in-1 Architecture: ETL + OLAP + BI



Our criteria for choosing OLAP engines

Speed

- Profiling 10s of GBs of datasets with sub-second performance

Simplicity

- Lightweight, embeddable, low maintenance overhead

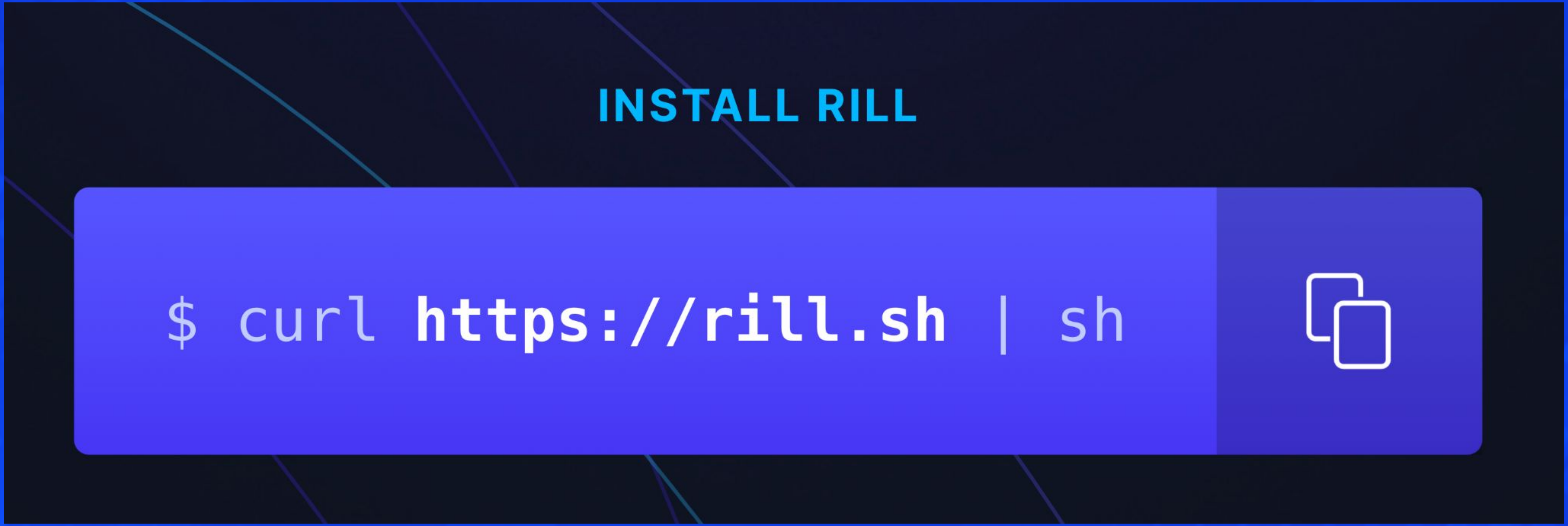
Scalability

- Serve 100s of concurrent queries and scales to 100s of GBs

Open source

- Permissive license with a vibrant developer community

Live Demo with the BlueSky Firehose


INSTALL RILL

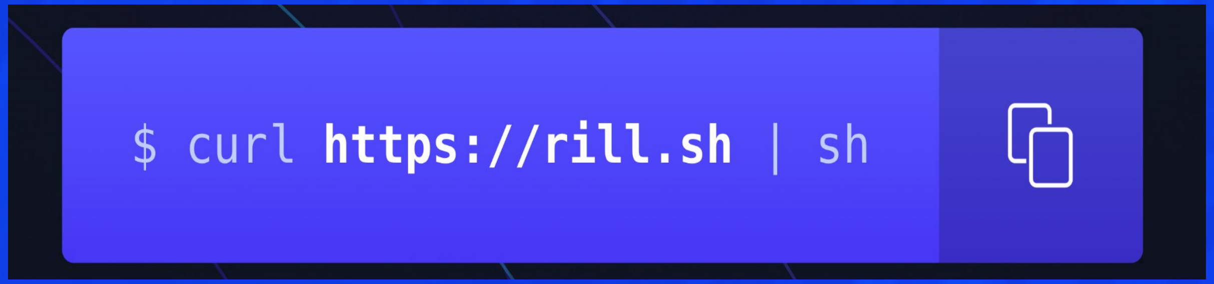
```
$ curl https://rill.sh | sh
```

Visit www.rilldata.com to copy command

Thank you!

mike@rilldata.com

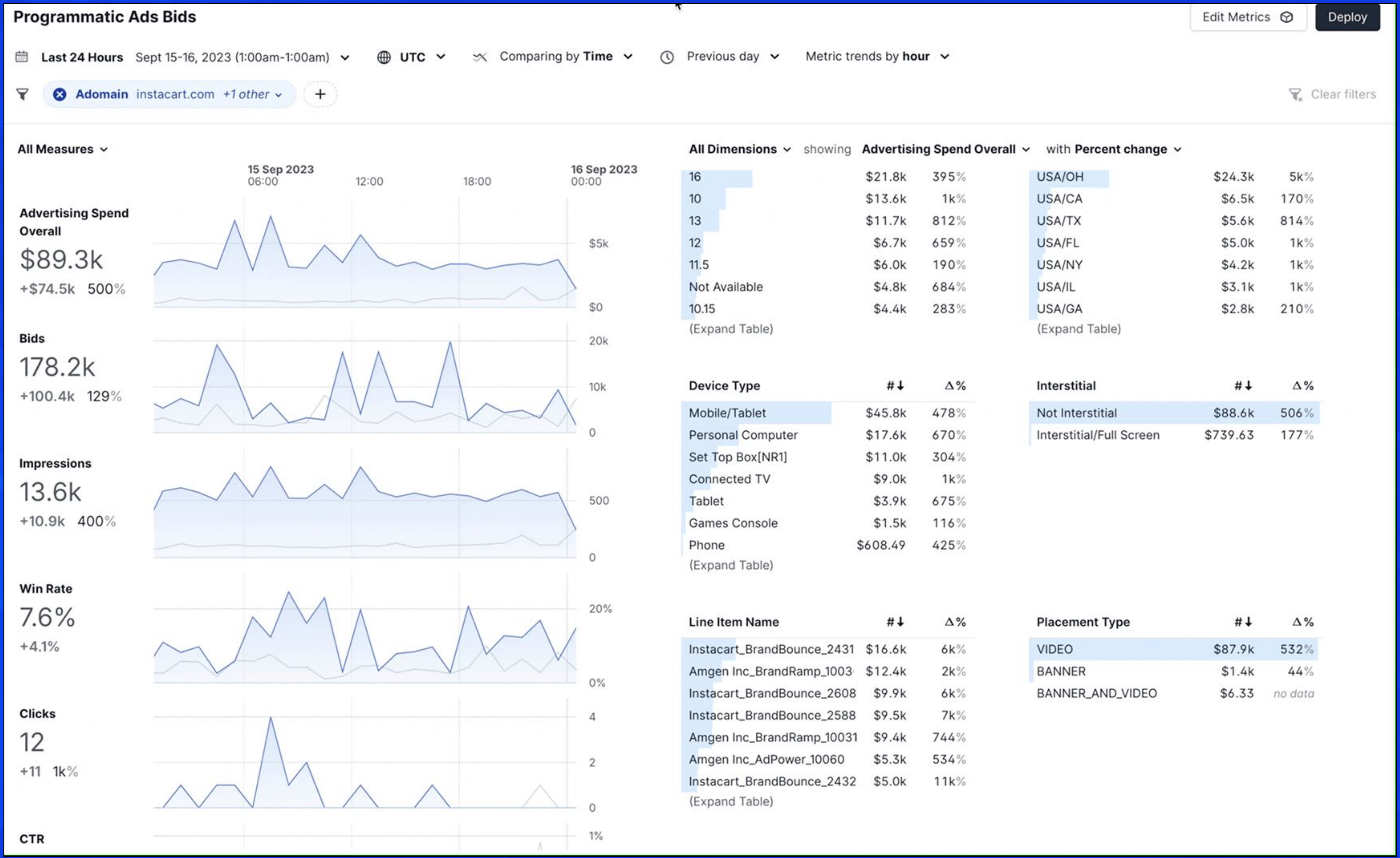
www.rilldata.com



```
$ curl https://rill.sh | sh
```

Follow us @RillData

Appendix



Programmatic Ads Bids

Edit Metrics | Deploy

Last 24 Hours Sept 15-16, 2023 (1:00am-1:00am) | UTC | Comparing by **Time** | Previous day | Metric trends by **hour**

Adomain instacart.com *+1 other* | + | Clear filters

All Measures

Advertising Spend Overall	$89.3k	+$74.5k 500%
Bids	178.2k	+100.4k 129%
Impressions	13.6k	+10.9k 400%
Win Rate	7.6%	+4.1%
Clicks	12	+11 1k%
CTR		

All Dimensions showing **Advertising Spend Overall** with **Percent change**

16	$21.8k	395%
10	$13.6k	1k%
13	$11.7k	812%
12	$6.7k	659%
11.5	$6.0k	190%
Not Available	$4.8k	684%
10.15	$4.4k	283%
(Expand Table)		

USA/OH	$24.3k	5k%
USA/CA	$6.5k	170%
USA/TX	$5.6k	814%
USA/FL	$5.0k	1k%
USA/NY	$4.2k	1k%
USA/IL	$3.1k	1k%
USA/GA	$2.8k	210%
(Expand Table)		

Device Type	# ↓	Δ %
Mobile/Tablet	$45.8k	478%
Personal Computer	$17.6k	670%
Set Top Box[NR1]	$11.0k	304%
Connected TV	$9.0k	1k%
Tablet	$3.9k	675%
Games Console	$1.5k	116%
Phone	$608.49	425%
(Expand Table)		

Interstitial	# ↓	Δ %
Not Interstitial	$88.6k	506%
Interstitial/Full Screen	$739.63	177%

Line Item Name	# ↓	Δ %
Instacart_BrandBounce_2431	$16.6k	6k%
Amgen Inc_BrandRamp_1003	$12.4k	2k%
Instacart_BrandBounce_2608	$9.9k	6k%
Instacart_BrandBounce_2588	$9.5k	7k%
Amgen Inc_BrandRamp_10031	$9.4k	744%
Amgen Inc_AdPower_10060	$5.3k	534%
Instacart_BrandBounce_2432	$5.0k	11k%
(Expand Table)		

Placement Type	# ↓	Δ %
VIDEO	$87.9k	532%
BANNER	$1.4k	44%
BANNER_AND_VIDEO	$6.33	*no data*

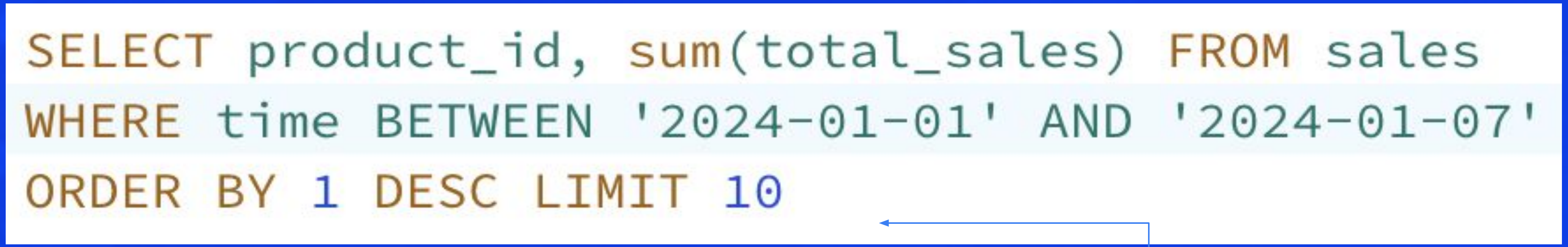
Speed at Scale Requires Coordinated Optimization Across the Stack

- Application layer
- Data model
- Data engine (DuckDB)

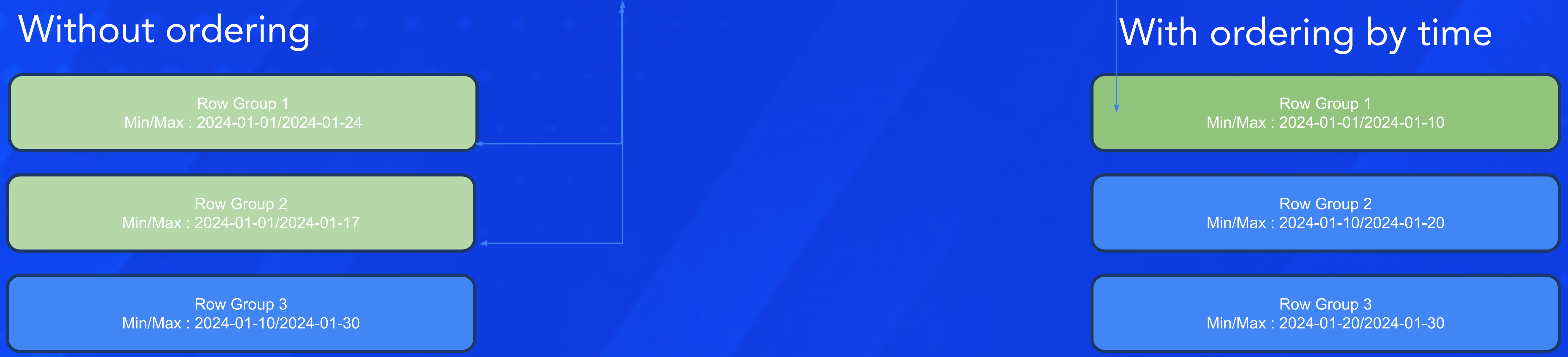
Data reordering speeds up min-max indexes

DuckDB's storage format stores the data in row groups, i.e., horizontal partitions of the data.

Sorting by time -> efficient use of min-max indexes



```sql
SELECT product_id, sum(total_sales) FROM sales
WHERE time BETWEEN '2024-01-01' AND '2024-01-07'
ORDER BY 1 DESC LIMIT 10
```

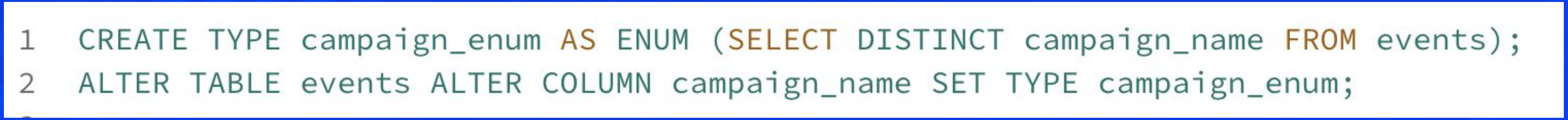


Without ordering
Row Group 1
Min/Max : 2024-01-01/2024-01-24
Row Group 2
Min/Max : 2024-01-01/2024-01-17
Row Group 3
Min/Max : 2024-01-10/2024-01-30
With ordering by time
Row Group 1
Min/Max : 2024-01-01/2024-01-10
Row Group 2
Min/Max : 2024-01-10/2024-01-20
Row Group 3
Min/Max : 2024-01-20/2024-01-30

Tip: For datasets already partitioned by time, preserving insertion order during ingestion is faster and leads to natural partitioning in row groups as well

Enums are faster than strings

Converting strings to enum types leads to lower RAM usage, improving speed at scale



```sql
CREATE TYPE campaign_enum AS ENUM (SELECT DISTINCT campaign_name FROM events);
ALTER TABLE events ALTER COLUMN campaign_name SET TYPE campaign_enum;
```

Trade-offs

- Higher data ingestion time

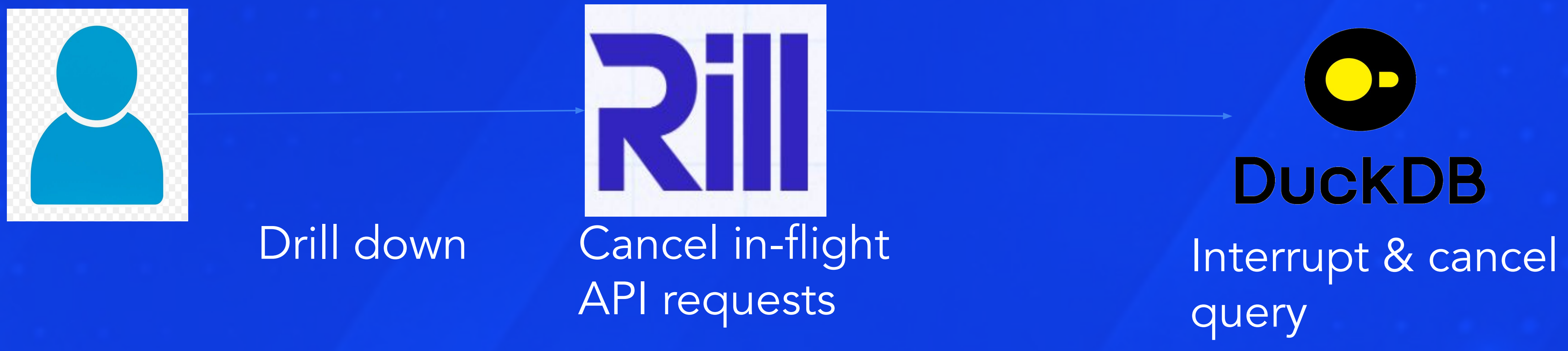
- Incremental ingestion becomes harder as you need to rewrite column types for new values in enum

More details - https://duckdb.org/2021/11/26/duck-enum.html

Query cancellation sheds unnecessary load

Exploratory dashboards can queue 100s of queries per view

When the user changes a view, pending queries are no longer needed



Priority queues keep applications snappy

Some application queries ought to be served faster than others:

- Interactive dashboard queries are the highest priority
- Scheduled reports and machine generated queries are lower priority

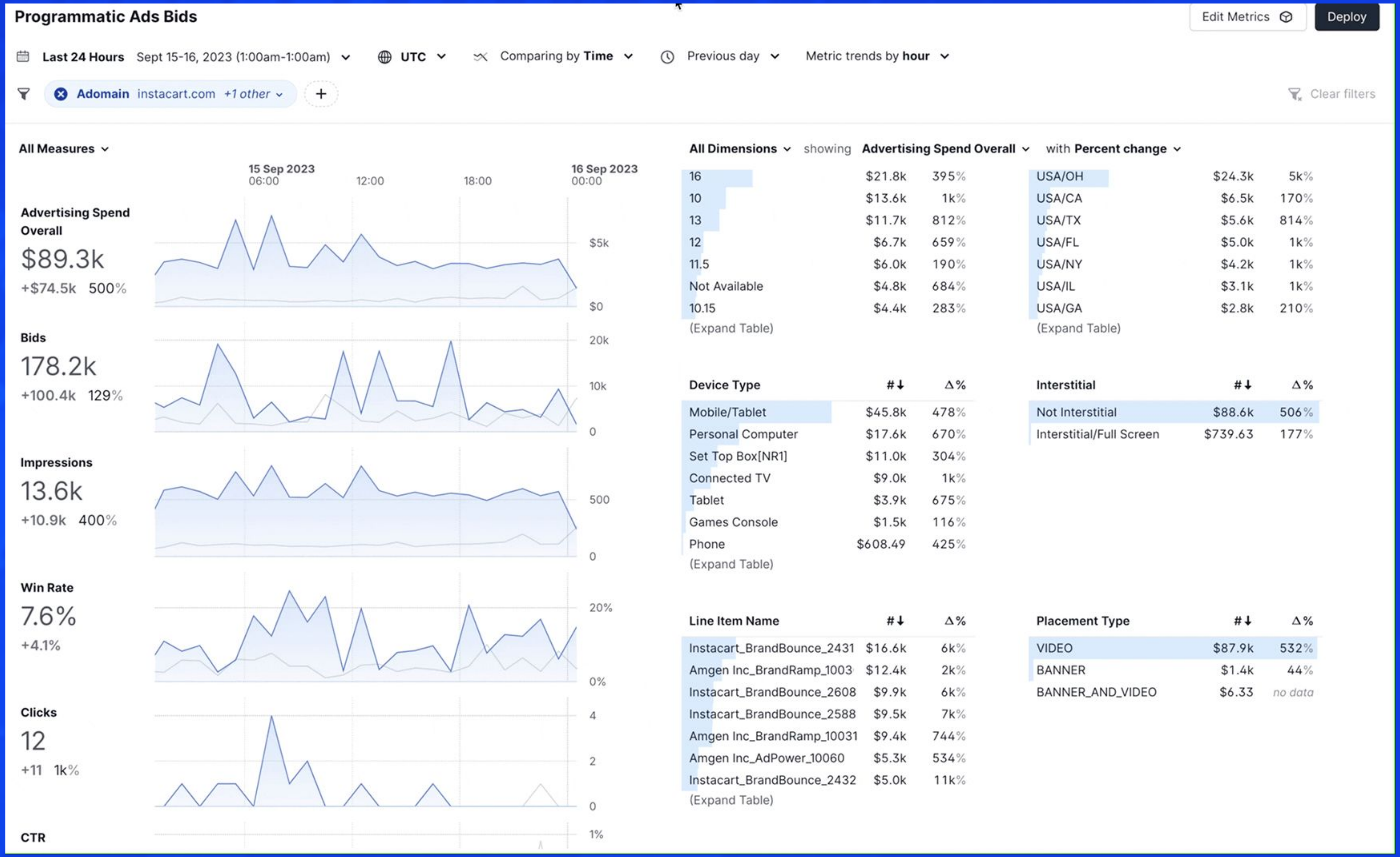
How

- Create a priority queue for database queries

Results

- Improved dashboard interactivity

WYSIWYF (What you see is what you fetch)
Delayed evaluation of queries



Programmatic Ads Bids

Last 24 Hours Sept 15-16, 2023 (1:00am-1:00am) · UTC · Comparing by Time · Previous day · Metric trends by hour

Adomain instacart.com +1 other · Clear filters

All Measures

Advertising Spend Overall	$89.3k	+$74.5k	500%
Bids	178.2k	+100.4k	129%
Impressions	13.6k	+10.9k	400%
Win Rate	7.6%	+4.1%	
Clicks	12	+11	1k%
CTR			

All Dimensions showing Advertising Spend Overall with Percent change

16	$21.8k	395%
10	$13.6k	1k%
13	$11.7k	812%
12	$6.7k	659%
11.5	$6.0k	190%
Not Available	$4.8k	684%
10.15	$4.4k	283%
(Expand Table)		

USA/OH	$24.3k	5k%
USA/CA	$6.5k	170%
USA/TX	$5.6k	814%
USA/FL	$5.0k	1k%
USA/NY	$4.2k	1k%
USA/IL	$3.1k	1k%
USA/GA	$2.8k	210%
(Expand Table)		

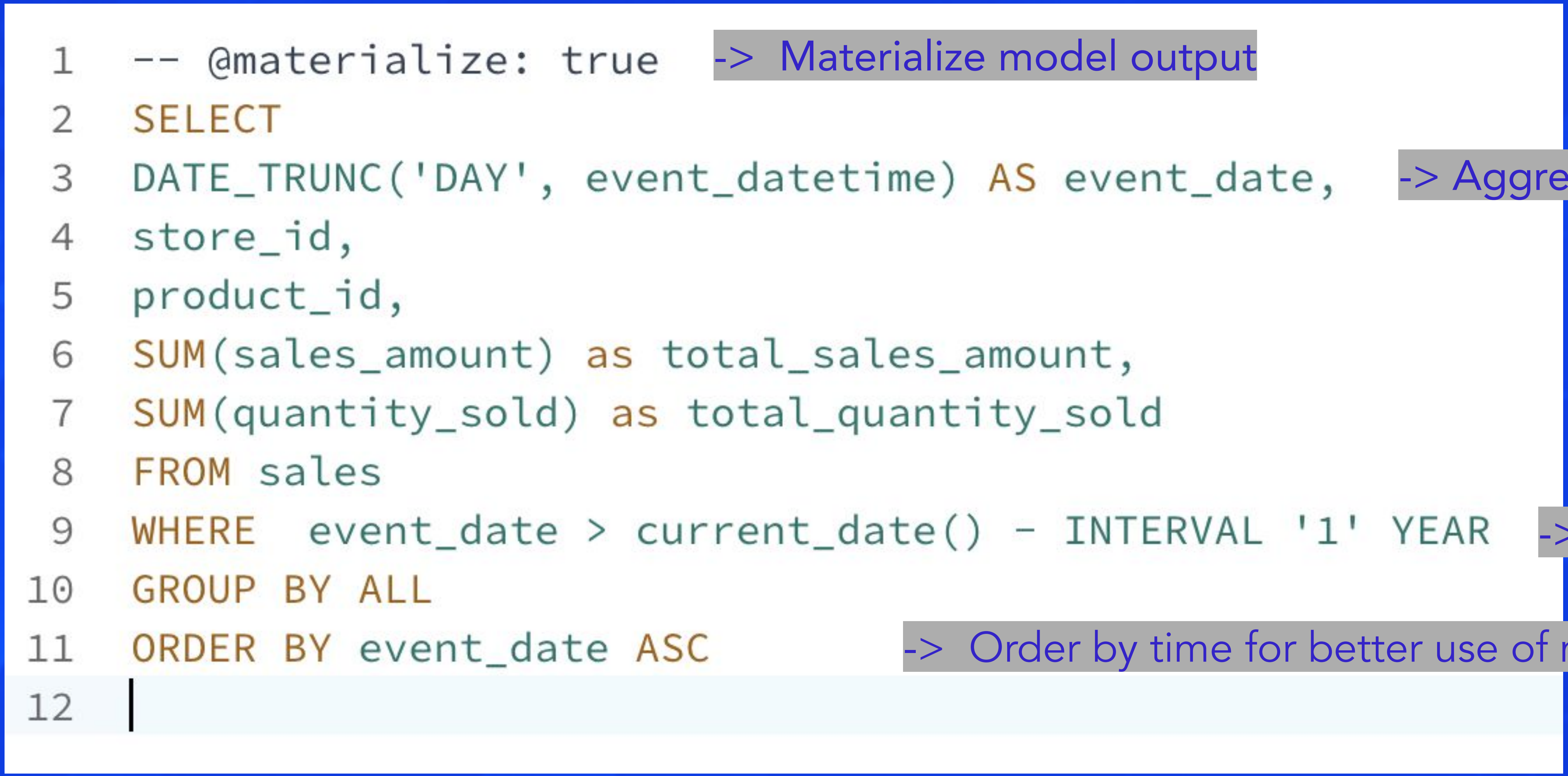
Device Type	# ↓	Δ%
Mobile/Tablet	$45.8k	478%
Personal Computer	$17.6k	670%
Set Top Box[NR1]	$11.0k	304%
Connected TV	$9.0k	1k%
Tablet	$3.9k	675%
Games Console	$1.5k	116%
Phone	$608.49	425%
(Expand Table)		

Interstitial	# ↓	Δ%
Not Interstitial	$88.6k	506%
Interstitial/Full Screen	$739.63	177%

Line Item Name	# ↓	Δ%
Instacart_BrandBounce_2431	$16.6k	6k%
Amgen Inc_BrandRamp_1003	$12.4k	2k%
Instacart_BrandBounce_2608	$9.9k	6k%
Instacart_BrandBounce_2588	$9.5k	7k%
Amgen Inc_BrandRamp_10031	$9.4k	744%
Amgen Inc_AdPower_10060	$5.3k	534%
Instacart_BrandBounce_2432	$5.0k	11k%
(Expand Table)		

Placement Type	# ↓	Δ%
VIDEO	$87.9k	532%
BANNER	$1.4k	44%
BANNER_AND_VIDEO	$6.33	*no data*

Data modeling: reduce data, retain insights



```sql
-- @materialize: true
SELECT
DATE_TRUNC('DAY', event_datetime) AS event_date,
store_id,
product_id,
SUM(sales_amount) as total_sales_amount,
SUM(quantity_sold) as total_quantity_sold
FROM sales
WHERE  event_date > current_date() - INTERVAL '1' YEAR
GROUP BY ALL
ORDER BY event_date ASC

```

-> Materialize model output

-> Aggregate to reduce data by 10X

-> Prune per business needs

-> Order by time for better use of min-max indexes