February 13, 2024

tinybird

From Kafka to Analytics hands-on workshop

90-minutes to help you start building on top of your Kafka streams

Jim Moffitt - Developer Advocate

Workshop goals

- Hands on experience with Tinybird.
- Explore and Understand the basics.
- Gain familiarity with core Tinybird objects: Data Sources, Pipes, Nodes, Materialized Views, and API Endpoints

Sections

- Creating Data Sources
- Building data analysis pipelines
- Introduction to Materialized Views
- Introduction to the Tinybird CLI
- Using version control with data projects

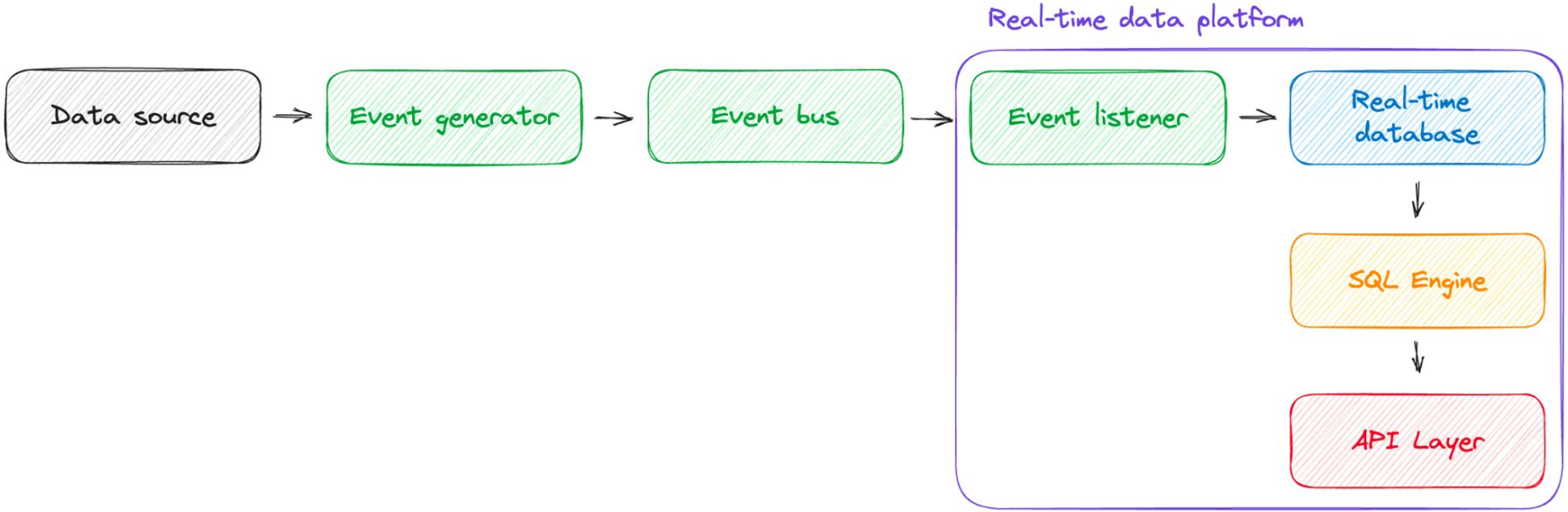
Tinybird core characteristics

What is Tinybird?

- An OLAP database designed to manage high data volumes and velocities. Tinybird integrates ClickHouse as its data engine
- A data platform for unifying data sources. Blending batch and real-time data. Enriching data streams with related metadata.
- Data analysis platform: Filtering, aggregating, joining, and transforming that data with SQL
- A place to design and deploy API-based data products.

What is Tinybird?

Tinybird is a real-time data platform



What will we build *with* today?

- A ‘dimensional’ table containing company metadata
- Kafka stream of company stock price data
- A Python script generating mock data
- Confluent Cloud
- Tinybird
- API data consumers

Kafka to Analytics workshop

symb	timestamp	price
ALG	2024-02-06 07:16:05.000	202.98
SYS	2024-02-06 07:16:05.000	113.08
BLP	2024-02-06 07:16:05.000	0.01
CCS	2024-02-06 07:16:05.000	163.97
CNS	2024-02-06 07:16:05.000	76.08
COM	2024-02-06 07:16:05.000	78.18
CPC	2024-02-06 07:16:05.000	270.51
CTC	2024-02-06 07:16:05.000	103.03
DYO	2024-02-06 07:16:05.000	3.88
ECC	2024-02-06 07:16:05.000	528.30
EGD	2024-02-06 07:16:05.000	90.77
EGI	2024-02-06 07:16:05.000	218.58
EGT	2024-02-06 07:16:05.000	117.57
EGY	2024-02-06 07:16:05.000	37.87
ELE	2024-02-06 07:16:05.000	236.75
ENE	2024-02-06 07:16:05.000	172.58
ENI	2024-02-06 07:16:05.000	243.21
EVE	2024-02-06 07:16:05.000	38.94
EEE	2024-02-06 07:16:05.000	171.3

What will we build today?

- Endpoints to return stock price data
- Provide endpoint query parameters for selecting companies and time periods of interest
- Build an endpoint that serves hourly stats

What will we build today?

Kafka

Kafka-to-Analytics / **filter**

Save

GET https://api.us-east.tinybird.co/v0/pipes/filter.json?company=ALG&max_results=100

GET hourly_stats

	Key	Value	Description	Bulk Edit
☑	company	ALG		
☑	max_results	100		
	Key	Value	Description	

Body

200 OK 393 ms 2.02 KB Save as example

Pretty Raw Preview Visualize JSON


```
18    "data":
19    [
20        {
21            "timestamp": "2024-02-09 23:25:04",
22            "symbol": "ALG",
23            "price": 210.9
24        },
25        {
26            "timestamp": "2024-02-09 23:24:50",
```

Workshop resources

- Getting started document
- Main GitHub Repository
  GitHub - tinybirdco/zero-to-tinybird
- Reference project GitHub Repository
  GitHub - jimmoffitt/k2a-workshop

Sections

- **Creating Data Sources**
- Building data analysis pipelines
- Introduction to Materialized Views
- Introduction to the Tinybird CLI
- Using version control with data projects

z2t_workshop | Connect to Git | Search anything ⌘K

- Overview
- Data Flow
- Playground
- Time Series
- Tokens

DATA PROJECT

- Pipes (0)
- Data Sources (0)

Metrics

Last update a few seconds ago | Last 5 minutes

REQUESTS	PROCESSED DATA	APIs AVG. LATENCY	APIs ERROR RATE
0	0.0B	0.00ms	0.00%

No jobs in the queue

ROWS INGESTED

0

Consumption from

NAME	ERRORS	REQUESTS	AVG. LATENCY	PROCESSED	AVG. PROCESSED
Query API (/sql)	-	-	-	-	-

Your Pipes

Create empty Pipe

Create your first Pipe

Use Pipes to work with data and create Endpoints. You can chain SQL queries to filter data, redefine your schema, join Data Sources or Pipes and shape the output of your API endpoints. Read more about Pipes in our docs.

Data Sources log

Add Data Source

Sections

- Creating Data Sources
- **Building data analysis pipelines**
- Introduction to Materialized Views
- Introduction to the Tinybird CLI
- Using version control with data projects

Building data analysis pipelines

- Filtering objects
- Aggregating objects
- Joining objects

Tinybird templating language

- Used to build in dynamic query parameters into Node SQL.
- Used to affect SELECT, WHERE, ORDER BY, GROUP BY, and LIMIT statements.
- In its most basic form:

  SELECT * FROM event_stream LIMIT **{{ Int32(max_results, 100) }}**

Tinybird templating language

- {{ Int32(max_results) }}
- {{ Int32(max_results, 100) }}
- {{ Int32(max_results, 100, description=”A popular option.”) }}
- {{ Int32(max_results, 100, description=””, required=True) }}

Tinybird templating language

```
%
SELECT *
FROM event_stream
WHERE 1 = 1
{% if defined(company) %}
  AND symbol = {{ String(company) }})
{% end %}
```

Sections

- Creating Data Sources
- Building data analysis pipelines
- Introduction to Materialized Views
- Introduction to the Tinybird CLI
- Using version control with data projects

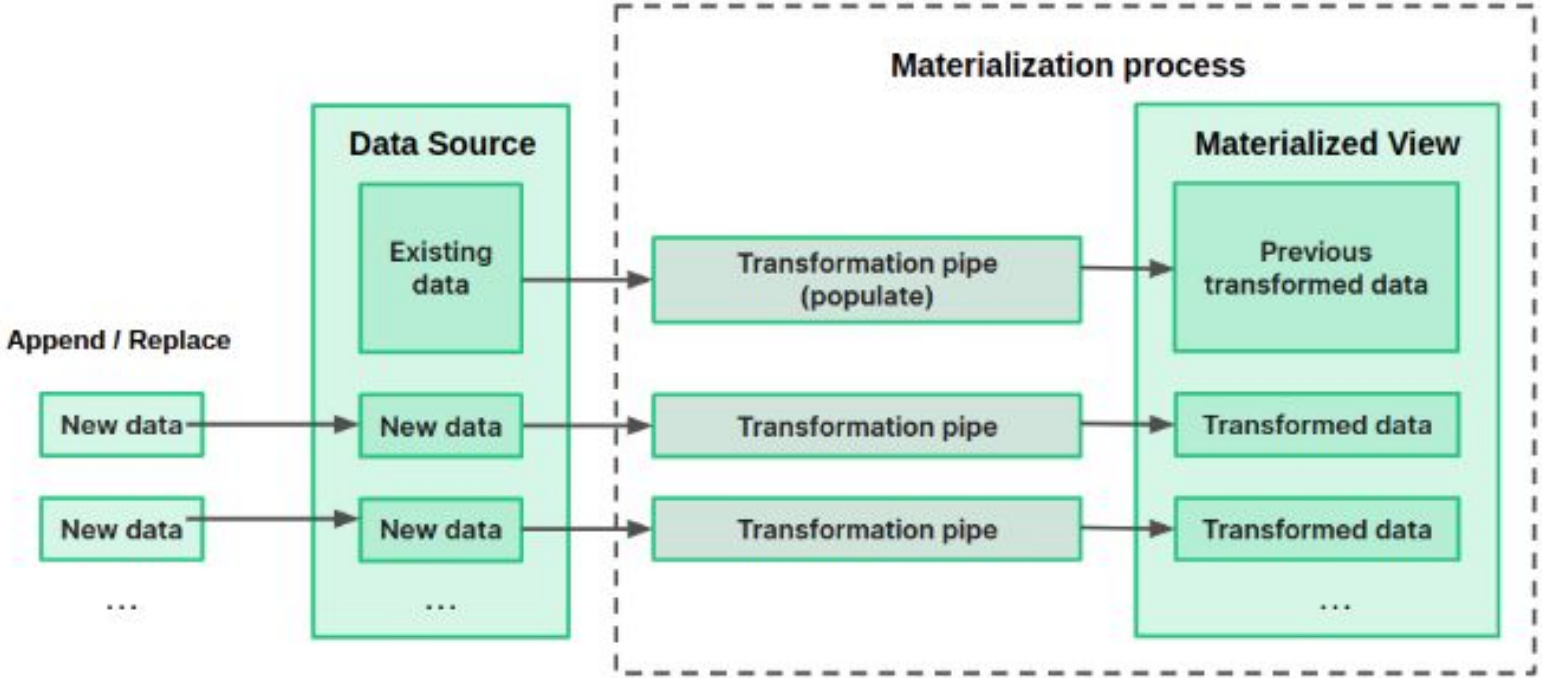
Materialized Views

- Processing data as it arrives, not when it is queried.
- Building aggregations, summaries, and views as data comes in.
- As Data Sources grow in size, MVs become more important for performance.

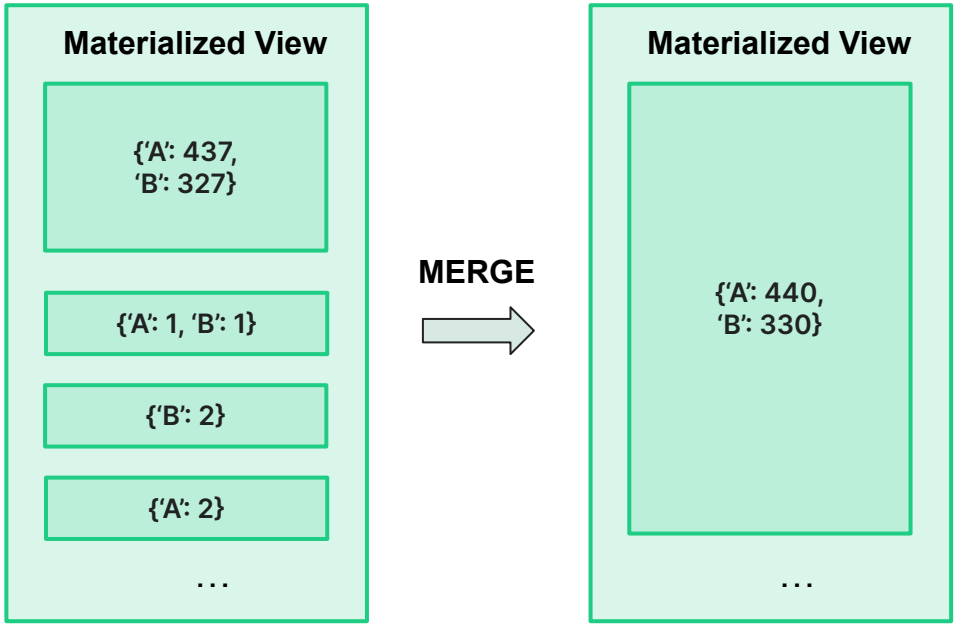
Materialized Views

- Materialized Views consist of:
  - Transformation pipe that writes to a new Data Source using -State operations.
  - A Data Source that maintains intermediate states for new data and existing data.
  - A second Pipe that reads from the Data Source and triggers -Merge operations.

What's an MV?


Materialization process
Data Source
Existing data
Transformation pipe (populate)
Materialized View
Previous transformed data
Append / Replace
New data
New data
Transformation pipe
Transformed data
New data
New data
Transformation pipe
Transformed data
...
...
...

Materialized Views



Sections

- Creating Data Sources
- Building data analysis pipelines
- Introduction to Materialized Views
- Introduction to the Tinybird CLI
- Using version control with data projects

Installing the CLI

- (Optional) Create a virtual environment:
  - `python3 -m venv .temp-project`
  - `source .temp-project/bin/activate`
- Install Tinybird-CLI:
  - `pip install tinybird-cli`
- Authenticate

Sections

- Creating Data Sources
- Building data analysis pipelines
- Introduction to Materialized Views
- Introduction to the Tinybird CLI
- Using version control with data projects

Thank you.

https://jimmoffitt.grafana.net/d/aadc39bb-e74b-44f4-89b1-e81f6f1047dd/weather-api?orgId=1&refresh=30s&var-City=Minneapolis

The 5 Rules of Fast Queries

- Rule № 1 ⟶ The best data is the one you don’t write.
- Rule № 2 ⟶ The second best data is the one you don’t read. (The less data you read, the better.)
- Rule № 3 ⟶ Sequential reads are 100x faster.
- Rule № 4 ⟶ The less data you process (after read), the better.
- Rule № 5 ⟶ Complex operations later in the processing pipeline.

Data types: rules of thumb

We should use **the least possible space:**

- Avoid NULL: coalesce is your friend.
- Strings and arrays are more expensive than fixed size types.
- String columns with less than 2000 different elements $\longrightarrow$ LowCardinality(String)
- Number better than String.
- Integer better than Float.
- The less bits, the better.