

# **Building InfluxDB 3.0**

With Apache Arrow, DataFusion, Flight, Parquet

Andrew Lamb | Staff Engineer, InfluxData



#### 🇳 influxdata®



#### **Andrew Lamb**

Staff Engineer InfluxData > <del>20</del> 21 years in enterprise software development

Oracle: Database (2 years)

DataPower: XSLT compiler (2 years)

Vertica: DB / Query Optimizer (6 years)

Nutonian/DataRobot: ML Startups (7 years)

InfluxData: InfluxDB 3.0, Arrow, DataFusion (4 years)

## Goals

Convince you, via example, that:

- 1. Databases of the future will be assembled from reusable components
- 2. This is not a terrible idea

Talk outline:

- Database Implementation Trends
- Time Series Databases, need for (yet another) one
- FDAP: Flight, DataFusion, Arrow, Parquet: Rationale + Use InfluxDB 3.0

Blog: Flight, DataFusion, Arrow, and Parquet: Using the FDAP Architecture to build InfluxDB 3.0



# Thesis: Long Term Trends in Databases





#### "One Size Fits All"

#### An Idea Whose Time Has Come and Gone

Stonebraker & Çetintemel (2005)

Case Study: In which InfluxData decides to build a **new time series database**.

# What / Why of Time Series Databases

Specialized for storing data with times (obviously).

Key Properties:

- 1. High volume, denormalized ingest
  - eg. host=myhost123.example.com repeated over and over
- 2. Low latency query after load: milliseconds between ingest and query
- 3. Schema on write: new columns can appear at any time, backfills
- 4. Rapid data value decay: newest data is super important, falls off drastically

**Examples:** 

Open → InfluxDB, Timescale, Graphite Whisper, VictoriaMetrics Closed → Facebook Gorilla, Google Monarch, AWS Timestream, DataDog Husky



# InfluxDB 3.0 Requirements

- Need: No series cardinality limits:
  - Tech: TSM (LSM Tree / KV Store) → Column Store
- Need: 'Infinite' Retention
  - Tech: TSM on locally attached disks → object store (cheap!)
- Need: Elastic Scalability
  - Tech: Shared Nothing (local disk)→ Disaggregated Storage (S3)
- Need: Ecosystem Compatibility
  - Tech: InfluxQL / custom APIs → SQL + JDBC/ODBC



## So, let's build a new Database (How hard could that be, really?)

## It is hard (expensive) to build a new databases

Database company money raised

- Snowflake: \$2B
- Databricks (Spark) \$3.5B
- MongoDB: \$311M
- SingleStore: \$464.1M
- CockroachLabs (CockroachDB): \$633.1M
- Pingcap (TiDB): \$341.6M
- Elastic: \$162M
- TimescaleDB: \$181M
- DuckDB \$? / MotherDuck: <del>\$47.5M</del> \$100M

l did this at Vertica too



Source: https://www.crunchbase.com

#### "We can do it with the Apache Arrow Ecosystem"





Sinfluxdata Products v Developers v Customers v Resources v

opers V Customers V Resources V

#### Apache Arrow, Parquet, Flight and Their Ecosystem are a Game Changer for OLAP

By Paul Dix / Apr 16, 2020 / Community, Developer

Apache Arrow, a specification for an in-memory columnar data format, and associated projects: Parquet for compressed on disk data, Flight for highly efficient RPC, and other projects for in-memory query processing will likely shape the future of OLAP and data warehousing systems. This will mostly be driven by the promise of interoperability between projects, paired with massive performance gains for pushing and pulling data in and out of big data systems. With object storage like S3 as the common data lake, OLAP projects need a common data API, which Parquet represents. For data science and query workloads, they need a common RPC that is optimized for pulling many millions of records to do more complex analytical and machine learning tasks.

In this post, I'll cover each of these areas and why I think the Apache Arrow unbrella of projects represents the common API around which current and future big data, OLAP, and data warehousing projects will collaborate and innovate. I'll conclude with some thoughts on where these projects are and where things might be going.

#### **Apache Arrow**

Apache Arrow is an in-memory columnar data format. It is designed to take advantage of modern CPU architectures (like SIMD) to achieve fast performance on columnar data. It is ideal for vectorized analytical queries. The Arrow specification gives a standard memory layout for columnar and nested data that can be shared between processes and query processing libraries. It's the base level building block for working with in-memory (or MMAP'0) data that ties everything together. It's designed for zero-copy semantics to make moving data around as fast and efficient as possible.

#### Persistence, bulk data and Parquet

Data is the API. I'm sure others said it before me, but I said it myself back in 2010 when talking about service-oriented design and building loosely coupled systems that interacted through message buses like RabbitMQ and later Kafka. What I meant the was that the data you passed through message queues served as the API for services that integrated with each other through those queues. And like APIs, that data needs to have a common serialization format and its schema should be versioned. More broadly, I was trying to highlight the importance of data interchange, which is as important for data processing and analytic systems as it is for services. Categories About Company Community Developer ~ Community Flux InfluxData InfluxDB Templates InfluxDB Partners v Kapacitor **Release Notes** Tech Tips Telegraf Chronograf General InfluxDB Cloud InfluxDB Enterprise InfluxDays Newsroom ~ Trust Tutorial Use Case v

Get InfluxDB

Pricing Contact Us Log In

in

#### Try InfluxDB Cloud

The most powerful time caries database as a service i *#*Join Fou

#### https://www.influxdata.com/blog/apache-arrow-parquet-flight-and-their-ecosystem-are-a-game-changer-for-olap/





# Benefits of building on FDAP foundation

⇒ Innovation on Time Series, FDAP provides standard (lengthy to build) pieces

```
match tool_needed_for_database {
  File format (persistence) => Parquet,
  Columnar memory representation => Arrow Arrays,
  Operations (e.g. multiply, avg) => Compute Kernels,
  SQL + extensible query engine => Arrow DataFusion,
  Network transfer => Arrow Flight,
  JDBC/ODBC driver => Arrow FlightSQL,
```



# Let's do it

# InfluxDB 3.0 Architecture





## InfluxDB 3.0: Fast, Real Time & Cost Effective







#### Non profit governance of open source communities

"Community over Code" - The Apache Way



# Apache: Benefits for InfluxDB 3.0

- $\Rightarrow$  Predictable Foundation
- Stable License: (ASL 20 years old) low risk of changes, (ahem OpenTofu)
- **Communication**: Predictable and open (if slow)
- Multi-Vendor Participation: Shared investment reduces individual risk
- Long Term Maintenance: Hedged against life changes, corporate strategy shifts, VC funding cycles





Columnar file format from 2013 (originally part of Hadoop ecosystem)

https://parquet.apache.org/



18 | © Copyright 2024, InfluxData

# Apache Parquet: Benefits for InfluxDB

⇒ Avoid Engineering overhead of building a custom file format + ecosystem

**Compression:** Works well with wide data variety (including time series) **Performance:** Techniques such as projection, filter pushdown, late materialization

**Interoperability:** "Defacto" interchange format for analytics, immediate compatibility with 1000s of tools (Apache Iceberg makes it even easier)



## TSM (time series) vs Parquet Compression

Format	File Count	Size (GB)
TSM	483	591
TSM (gzip)	483	97
Parquet	246,140 (4K - 2.2GB)	118

Entire dataset (Parquet/TSM: 20%)

Format	File Count	Size (GB)
TSM	1	.110
Parquet	54 (4K - 7M)	.020

File 1 (Parquet/TSM: 18%)

Format	File Count	Size (GB)
TSM	1	2
Parquet	974 (12K - 53M)	.226

File 2 (Parquet/TSM 11%)





# Parquet Organization



("PAX" in DB literature)



## Parquet Structure + Metadata



Footer contains location of pages, and statistics such as min/max/count/nullcount.

#### Source:

https://www.influxdata.com/blog/guerying-parguet-millisecond-latency/

("Zone Maps", "Small Materialized Aggregates" in DB literature)



# Parquet Projection + Filter Pushdown



#### Source: https://www.influxdata.com/blog/guerving-parguet-millisecond-latency/



# How does InfluxDB 3.0 use Parquet?

Durable Store: All data durably persisted to object store as parquet; Query: Read from parquet files + latest unpersisted ingester data



# APACHE ARROW

In memory format for fast vectorized processing

https://arrow.apache.org/



# Arrow: Benefits for InfluxDB

⇒ Best practice for storing columnar data in memory

Type System: Full type support (integers, strings, timestamps, etc)
Null Support: Standard null bitmask representation + semantics
Efficient Encodings: Dictionary encoding for Strings
Natural Integration: DataFusion, parquet libraries, Arrow Flight
Optimized Compute Kernels: Fast vectorized kernels are well understood, but time consuming to implement, test and maintain

Note: Arrow has (many) more features than this



#### Arrow Array: Int64Array



Pretty much what you will find in every vectorized column store engine

**Arrow Array** 







```
let output = gt(
   &left,
   &right
);
```

The gt (greater than) kernel computes an output BooleanArray where each element is left > right

Kernels handle nulls (validity masks), optimizations for different data sizes, etc.

~50 different kernels, full list: <u>docs.rs page</u>



## How does InfluxDB 3.0 use Arrow: Ingester Buffers

Indirectly: via DataFusion and Flight.

**Directly:** Ingest path, which parses input line protocol - Arrow Arrays







Highly customizable, fast query engine that uses Arrow

https://github.com/apache/arrow-datafusion

30 | © Copyright 2024, InfluxData



# Arrow DataFusion: Benefits for InfluxDB

⇒ Fast full featured, extensible query engine

All the OLAP buzzwords: vectorized, columnar, multi-core, streaming, out of core, ...

**"Full" SQL:** JOINs, date/time/timestamp functions, structured data, ...

**Customizable:** Easily extend time series specific functions (e.g. date\_bin gapfill, InfluxQL, ...)



# DataFusion: Input / Output





#### DataFusion: Totally Customizable Architecture



Query FrontEnds

Plan Representations (DataFlow Graphs) Optimized Execution Operators (Arrow Based)



# How InfluxDB 3.0 uses DataFusion





Efficiently send columnar data (as Arrow Arrays) over the network

https://arrow.apache.org/docs/format/Flight.html



# Arrow Flight: Benefits for InfluxDB

⇒ Efficient network transfer; wide client support

Network Efficient: columnar format, high bandwidth transfer

**CPU Efficient:** "zero copy" Serialization / Deserialization

Pre Existing Clients: Minimal effort to make InfluxDB 3.0 clients

(just use Arrow Flight clients)



## How does InfluxDB 3.0 use Flight? Native Query Protocol

1. Client ←→ Querier





## How does InfluxDB 3.0 use Flight? Ingester ←→ Querier

1. Ingester and Querier's internal protocol is built on Flight\*



🌒 influxdata®

# APACHE ARROW **FlightSQL**

*Execute SQL queries and return results with a standard (non custom) client* 

https://arrow.apache.org/docs/format/FlightSql.html



# Arrow FlightSQL: Benefits for InfluxDB

⇒ Access to SQL ecosystem, without implementing our own

drivers/connectors

Prepackaged Client Libraries: JDBC/ODBC/Ecosystem drivers

Pre-packaged Integrations: Many systems already read from FlightSQL

Community Leverage: E.g. FlightSQL Grafana plugin, others contribute



# Apache Arrow FlightSQL

- Send SQL queries, receive Responses as Arrow Arrays
- Has clients in many languages / APIs (JDBC, python DB API, etc)

#### Introducing Apache Arrow Flight SQL: Accelerating Database Access

#### PUBLISHED 16 Feb 2022

José Almeida, James Duong, Vinicius Fraga, Juscelino Junior, David Li, Kyle Porter, Rafael Telles

We would like to introduce Flight SQL, a new client-server protocol developed by the Apache Arrow community for interacting with SQL databases that makes use of the Arrow in-memory columnar format and the Flight RPC framework.

Flight SQL aims to provide broadly similar functionality to existing APIs like JDBC and ODBC, including executing queries; creating prepared statements; and fetching metadata about the supported SQL dialect, available types, defined tables, and so on. By building on Apache Arrow, however, Flight SQL makes it easy for clients to talk to Arrow-native databases without converting data. And by using Flight, it provides an efficient implementation of a wire format that supports features like encryption and authentication out of the box, while allowing for further optimizations like parallel data access.

While it can be directly used for database access, it is not a direct replacement for JDBC/ODBC. Instead, Flight SQL serves as a concrete wire protocol/driver implementation that can support a JDBC/ODBC driver and reduces implementation burden on databases.



Read More:Expanding Arrow's Reach with a JDBC Driver for Arrow Flight SQL

https://arrow.apache.org/blog/2022/02/16/introducing-arrow-flight-sgl/



## How InfluxDB 3.0 uses FlightSQL







# Before Flight SQL



43 | © Copyright 2024, InfluxData



## Alternate Strategy: Use Posgres FEBE

Server: RowDescription fields:[{name:"a", type:"int", format:"text"}, ...]

Server: DataRow fields:[{data:"321"}, {data: "Charles River Watershed"} ...]

Server: DataRow fields:[{data:"42"}, {data: "Hunter Valley"} ...]

Server: DataRow fields: [{data: "12"}, {data: "Merrimack Valley"} ...]



It also turns out clients using the postgres driver tend to try and query the postgres metadata tables

#### ... (steps elided) ...

... (lots ros for each data)



# Conclusion

# Conclusion

- Building Databases from scratch is hard (and expensive \$\$)
- You don't have to anymore 🎉
- We built InfluxDB 3.0 using Apache Flight, DataFusion,
   Arrow, and Parquet, and it was awesome
- + + + + + +: Highly recommended for your next projects



## **Related Work**



The Compo	sable Data Mar	agement System	n Manifesto
Pedro Pedreira Meta Platforma Inc. pedroerpilimeta.com	Orni Erling Meta Platforma Inc. serlingälmeta.com	Konstantinos Karanasos Mata Piatieme Inc. Matarenasosijiencia con	Scott Schneider Heta Pletform Inc. scottanijeneta.com
Wes McKinney Voltron Data west[two]tronalata.com	Satya R Valluri Datahtida Inc atya valluri@datahtida.com	Mohamed Zait Databricks Inc.	Jacques Nadeau Sendock n. jacquesijimandeck in
AUTION CONTRACT AND A A A A A A A A A A A A A A A A A A	while a second system of the second system. The first system is the second system is the second system of the seco	<text><text><text></text></text></text>	In prophetic section, and a section of the section
"Die werk in forsend under the Consten Date Unsen. Viel brige Unseiterungenen zug bie die konnen Par angruns begraht Anne erternet emailing anleigheich zug Engerigtet is belefigte konnend to the VLBB Ensistement.	new HYRCNO LUbicensional matche au all 43 to star popp of loy flat branc, detait permanent by the consecution(c). Publication rights	AP's and encapedating respon could more easily be adapted, for and accelerators, or the underly a mobile stack that more rec-	delities, data management softwar e example, to levenage novel device ing hardware evolves, By relying o ution runies and laneware levales

Velox: Meta's Unifie	d Execution Engine
Pedro Pedreira, Orri Erläng, Masha Ba	amaneva, Kevin Wilforg, Laith Sakka,
Krishna Pai, Wei He, Bir	wapesh Chattopadhyny
herberen ortiket missenatura kryingilit	ne bakka keni webe himaned-hitib.com
INTRACT	This maketer has more the deal that success and
The ad-has development of hew specialized computation engines targened to very peelfs dens workshoch has consold a sized data landwape. Community, there enginess abare like is confising with such stilter and are band in matisfati, evelow, and enginese, and utilizated percells an increasive engenisms to data users. In or- to is address there haves, Moda are experises to data users. In or- to is address there haves, Moda are unrel Volum, a served pers sames D - disabase anothering hiltony. Wene provides remailing ensuable. high percentance, Moda and works sparsely data percessing ensuable. high percentance, and data within sparsely data percessing	down of specialized regimes that are built using allow works and linearized and due life is analogic with re- axe written in different hugangars, and are maintained and sequence planes. Manower, enviring and optimid regions as hardware and are cases readers, is not post- do on an power specialism. The complex constants of the down on a power plane hash. For example, constants of the two powers sured hardware advancements. The author moderates and WRAM, experiming features the 'two specialisms' data with the two planes.
components for building execution englishs, and exhibiting data management systems. The Blowy hereity relevants we verivalisation and adaptivity, and in-designed from the ground up to support effi- cient comparations are coupled data types due to their adaptive modern workdamic. Writes a surveyed in singulation in building and ph- wilds more than a down-data system at Mata, including and ph- ol specer engines such as Proven and Data, including and ph-	types for ML werkloads, and loversping forzer incorrectly by the source-boundary are importantial and invariable requests with dispatch arise of explorationian and feature importantly, this Dispatchetical additionally impacts the pro- of data unsets, who are commonly required to interact v and different regimes to finish a particular task. The axial types, functions, and aggregation very axian taken spin
platform, smorage bears and data warehouse inparties industrio- tions, machine bearsing exotons for hardner sequences gas and data proprocessing OP/Sweb2, and noor. It provides benefits in stems of the efficiency with seq descent station generational periodes only found in individual regimes, (b) increased consistency for data says, and ici engineering efficiency by promoting resulting.	The ORDER'S of Moore Tableon, the Indexing, and cambre workly inconduct names engines. For instance, so, in follows conducted al 30-bits information of the Index on Index Opposed for string of engines, the Index on Index Opposed forest in prevane was and the Index on Index Opposed forest in prevane was and the Index on Index Opposed forest in prevane was and the Index Opposed forest in prevane was and the Index Opposed forest in the Index Opposed for Index Opposed Forest in the Index Opposed for Index Opposed Forest Index Opposed for Index Opposed for Index Opposed Forest Index Opposed Forest Index Opposed for Index Opposed Forest Index Opposed Forest Index Opposed Forest Opposed Forest
PTERM References Format Productions of Eding Marka Remainsm, Kevin Willing, Lath Halas, Ecological Re, Wei He, Breugeth Chairspellery, Holes, Herics Esabel Enversion Epigen, PTERM, 1022, PTEV - 1886, SEE, doi:10.1073/9710421.0104009	mostly in the language fronteed (VQC, dataframes, and othe the optimizer; the way india are distributed among works (also referred to a stary india, and the ID layer. The exer- gines at the so as of these reprints are all other similar. All need a type system to represent acidar and complex data to
1 INTRODUCTION	coprotion evaluation system, spectrace (such as joint, again
The increasing workload/sensity is madern that are manorophile with exponential advances proved here in loss the possibility and or specialized query and comparison megines, such trapped to a specialized query and comparison megines, such trapped to a special distribution of the special special special special and special special special special special special special and special special special special special special special methods to satisfy Q2 (see same including data propriors indicating and forum specialized specialized specialized specialized specialized special forum special specialized specialized specialized specialized special forum specialized	and notice in addition to strateging and a stravely architecture. The device of the strateging of the
This was a formation and a fee function formation 1000 ADA to be intermediated bases. Note they considered an annual fee formation of the term of the term for the term of term of term (term of the term of the term of the term onling a help-the term of term of the term of term of the term of the term of the TLB factorement, the LLBs of the term of the term of the term of the term of the term of the term of the data of the term of the term of the term of the term of the data of the term of the term of the term of the term of the data of the term of the data of term of the term of the data of term of the term of term of term of term of the term of t	In common usage scenarios, Ndox taless a fully sprintle plan as topest and performs the described computations memory enversion exceeding on the standard for the solution vide a RCS passes, a databases layer, silver DELs, or a glob optimizer, and it is usually not meant to be used directly same.



Modular Analytic Query Engine				
Andrew Lamb InfanData alambijinfundata.com	Tijie Shen Spare and Tour sila.hten/hypocard/free.io		Daniël Hores Daniejis danid herselpconlogiccom	
Jayjeet Chaksaborty IafaaData Johaksabortyäjäinfloodata.com	Melamet O Sym attanijko	tsan Kabak nata rends.ai	Chan Sun Apple Inc. surchasting ple com	
	Liang-C Appl Temphi@	hi Hsieh k In: ispjecom		
Domain Specific Language Operators	Multiple SQL Dialects	Data Plew Analysis	Application Logic Tile System Interface	
FUSIKN	FUS	KN .	FUSIKN	
igure 1. Apache Arrow DataFasien is otres designers implement demain ape arr matine technology (cores). Overv	a fast, embedded gu effie fealwara via er le rer fe manor is simile	ery engine that uses name APh Dilact, rat is to best of based, to	Apache Arrow as its in-memory former, her than se implementing similard CLAP ddfs internated extreme.	
Journet paths for the United National National paths for any section without its Read [11 prov[15] as its memory model. Many one wave deatheres, marking bearing specific models optimized with a widel formers net with monostrone that a rich formers net with deatwares is attached with a model and a statement of the statem	abeddable, wod ev (that case Apache mannish and open- ore, and other data absor, Dataf value do state-of-the-art ar and extendable	design based on o wave Foundation accountibility and competitive performance specific counts by CCS Concepts - sgement system Database geory for matters	per trebated intendeds and Agoshe Self- (21) generation: We activity that the second researching of Defaultions, slong with its researching of Defaultions, and with the researching of the second second second composing modeline components. Information systems — Batchase statu- tegistres: BMOS engine architectures; processing, Codine analytical process-	
minimum to calle digital or hard report of all e- minimum devices on a 2 grants where the distribution to par- ticities of a distribution of the distribution of the dis- tribution of the distribution of the distribution of the distribution of the distribution of the distribution distribution of the distribution of the distribution of the distribution distribution of the distribution of the distribution of the distribution distribution of the distribution of the distribution of the distribution distribution of the distribution of the distribution of the distribution of the distribution distribution of the distribution of the distribution of the distribution of the distribution distribution of the distribution of the dist	r gapt of this work for sided from replex worker are and their replex bear register for responses based, thereasting with the part an arriver or to mainlife a the Nepard	ing english. Krywerds: datab ACM Reference F Andere Lanh, Tijo met Onan Eabak, C Arrow Databaton English. In Proceedings and et Managonia USA, 14 pages, hitj	are spillenta umati Bon, David Horos, Jappet Chalendury, Mich Jan Han, et al. (Sing Chi Hinde, 2008, Agarbe A Fun, Enderhälde, Hindein Jonayler, Query of ACM SCORED/POCS International Confer- or (State SCORED/POCS Survey), New York, NY, without ang 200000000, 20000000.	

The Modern Data Architecture: The Deconstructed Database (USENIX ;login: Winter 2018)

The Composable Data Management System Manifesto (VLDB 2023)

Velox: Meta's Unified Execution Engine (VLDB 2022)

A Deep Dive into Common Open Formats for Analytical DBMSs (VLDB 2023) Apache Arrow DataFusion: A Fast, Embeddable, Modular Analytic Query Engine (To Appear SIGMOD 2024)



Try It Yourself



https://www.influxdata.com



https://github.com/InfluxCommunity





#### ΤΗΑΝΚΥΟυ

# **Backup Slides**



# Thank you!

**Questions / Discussion** 

Find out more:

- <u>Flight, DataFusion, Arrow, and Parquet: Using the FDAP</u> <u>Architecture to build InfluxDB 3.0</u>
- Apache Arrow: <u>https://arrow.apache.org/</u>
- Apache DataFusion: <u>https://arrow.apache.org/datafusion/</u>
- Apache Parquet: <u>https://parquet.apache.org/</u>

#### Defragmenting Data Access Across Systems





## Integration: Arrow Language Implementations





## Why Arrow Internally (and not just at interface)?



Agg ? Filter ? Scan ?

**Option 1:** Use Arrow Internally (DataFusion, pola.rs, Acero)

**Pros**: Fast interchange, reuse Arrow libraries

Cons: Constrained(\*) to Arrow

Option 2: Use specialized structures internally, convert to Arrow at edges (Velox, DuckDB)

Pros: Can use specialized structures

Cons: Maintain specialized code



## Why Arrow Internally (and not just at interface)?

**Theory:** Using Arrow is "good enough" compared to specialized structures

Pooled open source development → invest heavily in optimized Arrow kernels

So far results are encouraging

**Good:** Sorting, Filtering, Projection, Parquet

Could Improve: Grouping, Joining





https://github.com/tustvold/access-log-bench

