



# Apache Arrow

Cross-Language Development  
Platform for In-Memory Analytics

**Wes McKinney**

SciPy 2018

# Wes McKinney



- Created Python **pandas** project (~2008), lead developer/maintainer until 2013
- PMC Apache Arrow, Apache Parquet, ASF Member
- Wrote ***Python for Data Analysis*** (1e 2012, 2e 2017)
- Formerly Co-founder / CEO of DataPad (acquired by Cloudera in 2014)
- Other OSS work: Ibis, Feather, Apache Kudu, statsmodels

# Open standards: why do they matter?

- Simplify system architectures
- Reduce ecosystem fragmentation
- Improve interoperability
- Reuse more libraries and algorithms

# Example open standards

- Human-readable semi-structured data: XML, JSON
- Structured data query language: SQL
- Binary storage formats (with metadata)
  - NetCDF
  - HDF5
  - Apache Parquet, ORC
- Serialization / RPC protocols
  - Apache Avro
  - Protocol buffers
- Not an open standard: Excel, CSV (grrrr)

# Standardizing in-memory data

- Best example: strided ndarray / tensor memory (NumPy / Fortran-compatible)
- Why?
  - Zero-overhead memory sharing between libraries in-memory and processes via shared memory
  - Reuse algorithms
  - Reuse IO / storage code

# Tables and data frames

- Notoriously not based on open standards
- Vary widely in supported data types (e.g. nested data)
- Where are they found?
  - Internals of SQL databases
  - Big data systems (Apache Spark, Apache Hive)
  - In-memory data frame libraries: Python (pandas), R (base, data.table), Julia (DataFrames.jl)
- We say “data frame” but the byte-level RAM layout varies greatly from system-to-system

# Columnar tables: why?

- Analytical query memory access patterns
  - “Access all elements in column X in succession”
  - “Select columns A, B, C from table with 1000 columns”
- Optimal vectorized
  - Minimize CPU/GPU cache misses
  - Enable SIMD-based algorithms (SSE4, AVX)
  - Vectorized (no-branching) algorithms
  - Leverage columnar compression techniques

# Apache Arrow

- An open, language-independent standard for in-memory columnar data (tables, data frames)
- A development platform to build analytical data processing systems
- <https://github.com/apache/arrow>



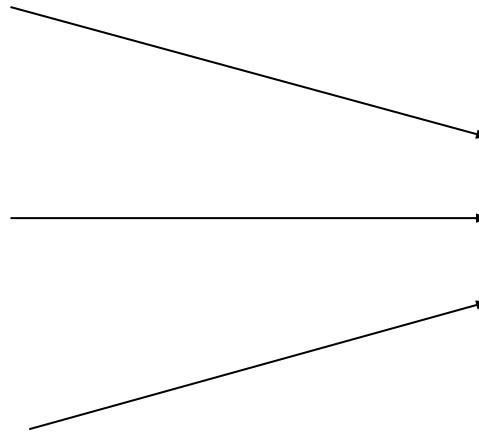
# “Portable” Data Frames

## Non-Portable Data Frames

pandas

R

JVM

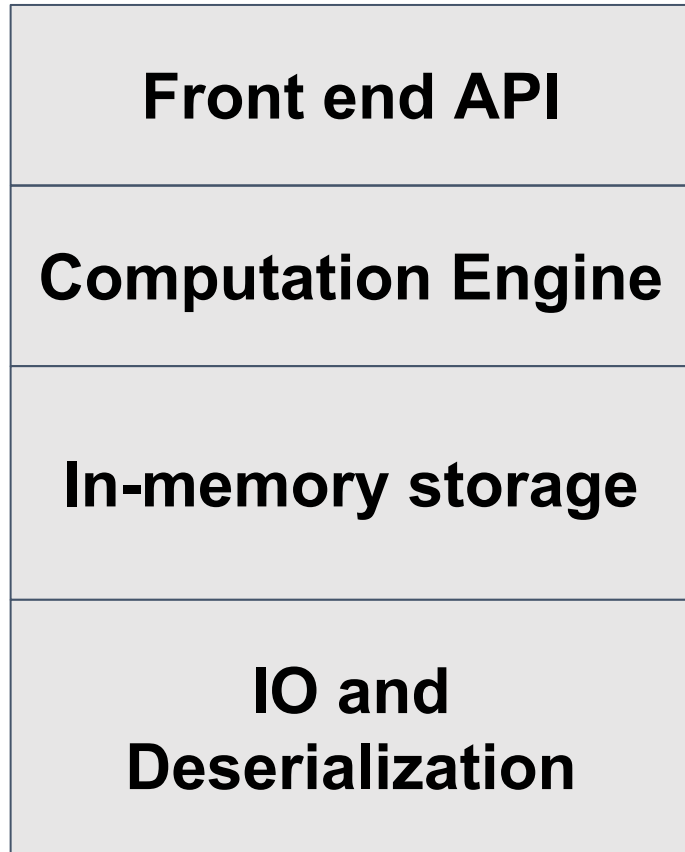


## Portable Data Frames

Arrow

Share **data** and **algorithms** at ~zero cost

# Analytic database architecture



- Vertically integrated / “Black Box”
- Internal components do not have a public API
- Users interact with front end

# Analytic database, deconstructed

**Front end API**

**Computation Engine**

**In-memory storage**

**IO and  
Deserialization**

- Components have public APIs
- Use what you need
- Different front ends can be developed

# Analytic database, deconstructed



**Front end API**

**Computation Engine**

**In-memory storage**

**IO and  
Deserialization**

**Arrow is front end agnostic**



shutterstock

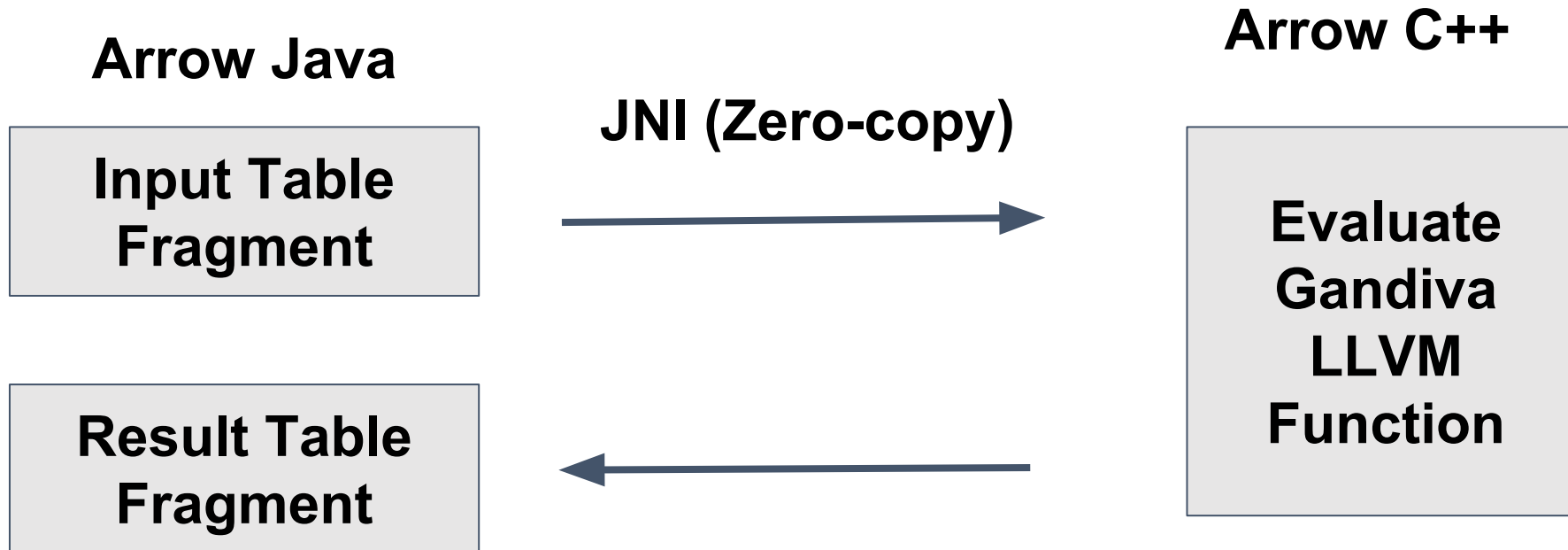
IMAGE ID: 552124468  
www.shutterstock.com

# Arrow: History and Status

- Community initiative started in 2016, initially backed by leading developers of ~13 major OSS data processing projects
- Project development status
  - Codebase 2.5 years old
  - > 170 distinct contributors
  - 9 major releases
  - Some level of support in 8 programming languages (C, C++, Go, Java, JavaScript, Python, Ruby, Rust)
  - Over 100K monthly installs in Python alone

# Example: Gandiva, Arrow-LLVM compiler

```
SELECT year(timestamp), month(timestamp), ...  
FROM table  
...
```

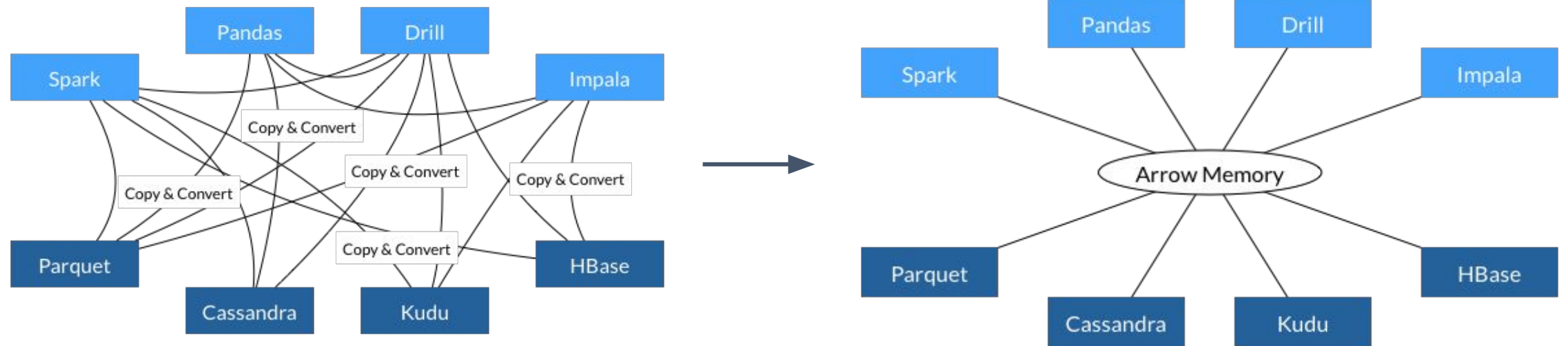


# Some Arrow Use Cases

- Runtime in-memory format for analytical query engines
- Zero-copy (no deserialization) interchange via shared memory
- Low-overhead streaming messaging / RPC
- Serialization format implementation
  - Zero-copy random access to on-disk data
  - Example: Feather files
- Data ingest / data access



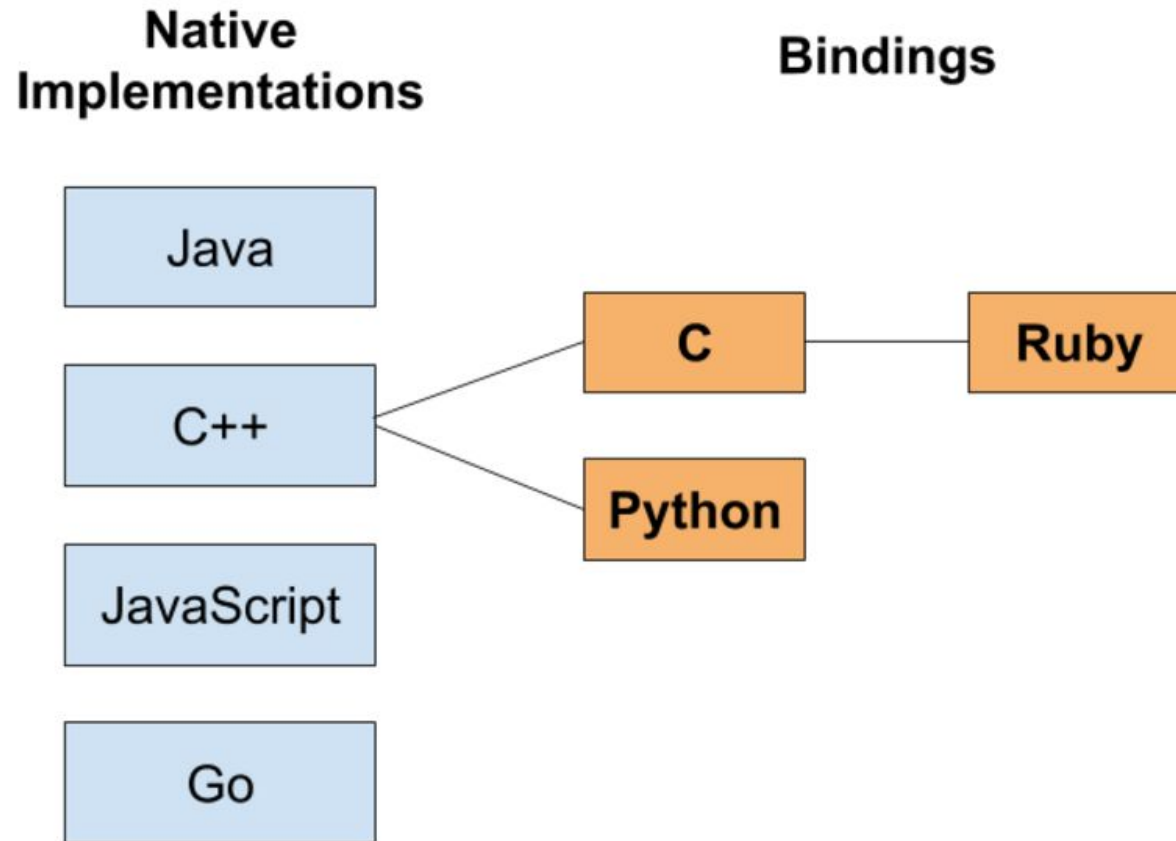
# Defragmenting Data Access



# Arrow's Columnar Memory Format

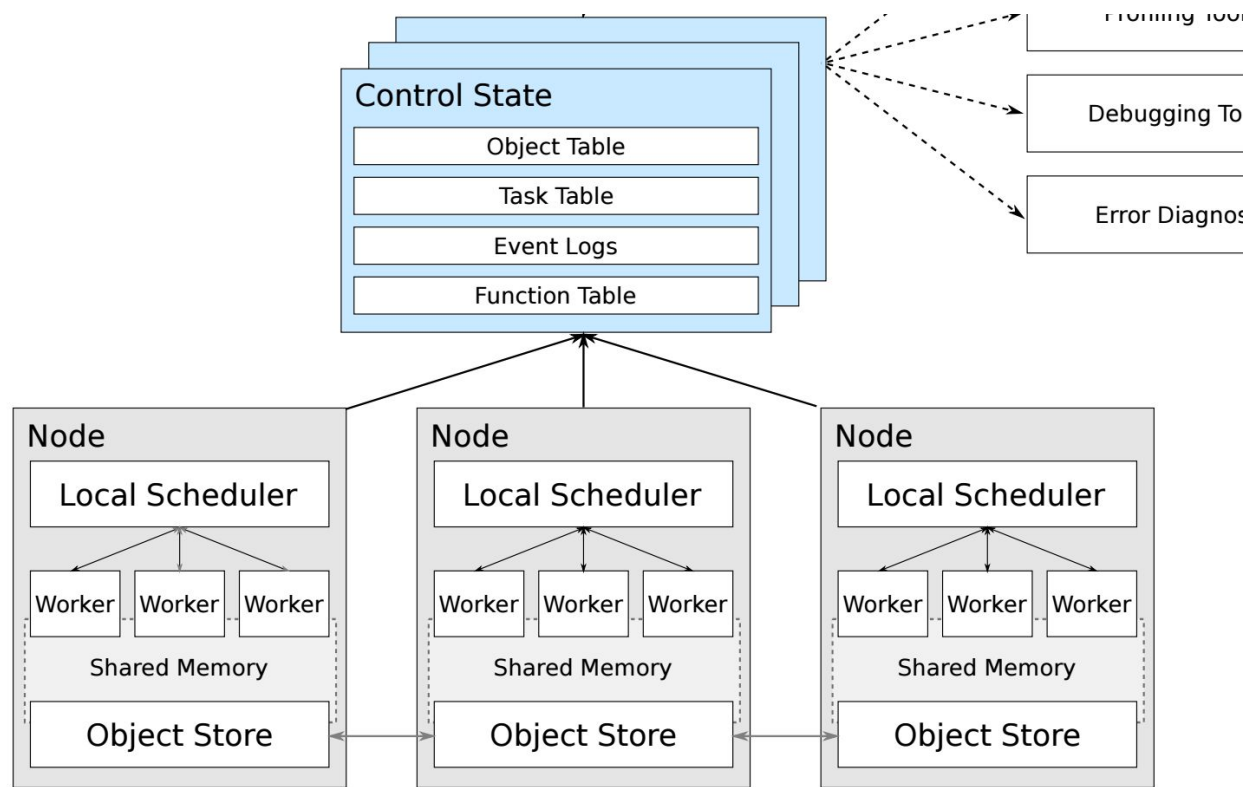
- Runtime memory format for analytical query processing
  - Companion to serialization tech like Apache {Parquet, ORC}
- “Fully shredded” columnar, supports flat and nested schemas
- Organized for cache-efficient access on CPUs/GPUs
- Optimized for data locality, SIMD, parallel processing
- Accommodates both random access and scan workloads

# Arrow Implementations and Bindings



**Upcoming:** Rust (native), R (binding), Julia (native)

# Example use: Ray ML framework from Berkeley RISELab



- **Uses Plasma, shared memory-based object store originally developed for Ray**
- **Zero-copy reads of tensor collections**

Source: <https://arxiv.org/abs/1703.03924>

# Arrow on the GPU

- NVIDIA-led GPU Open Analytics Initiative  
(<http://gpuopenanalytics.com>)
- “GPU DataFrame”: Arrow on the GPU
- Example: Execute Numba-compiled code on SQL results from MapD shared via CUDA IPC
- Plasma also supports GPU shared memory

# Some Industry Contributors to Apache Arrow



ClearCode

UBER



BlueYonder



cloudera



# Upcoming Roadmap

- Software development lifecycle improvements
- Data ingest / access / export
- Computational libraries (CPU + GPU)
- Expanded language support
- Richer RPC / messaging
- More system integrations

# Computational libraries

- “Kernel functions” performing vectorized analytics on Arrow memory format
  - Select CPU or GPU variant based on data location
- Operator graphs (compose multiple operators)
- Subgraph compiler (using LLVM -- see Gandiva)
- Runtime engine: execute operator graphs



# Data Access / Ingest

- Apache Avro
- Apache Parquet nested data support
- Apache ORC
- CSV
- JSON
- ODBC / JDBC
- ... and likely other data access points

# Arrow-powered Data Science Systems

- Portable runtime libraries, usable from multiple programming languages
- Decoupled front ends
- Companion to distributed systems like Dask, Ray



<https://ursalabs.org>

- Raise money to support full-time open source developers
- Grow **Apache Arrow** ecosystem
- Build cross-language, portable computational libraries for data science
- Build relationships across industry

# People

---

## Leadership



Wes McKinney

*Director*

Wes created the pandas project in 2008 and wrote the book Python for Data Analysis, helping popularize the use of Python for data science. He is a Member of The Apache Software Foundation, and is a PMC member for Apache Arrow and Apache Parquet. He was formerly the CEO and co-founder of DataPad. He is the managing director of Ursa Labs.



Hadley Wickham

*Advisor*

Hadley is the creator of many of the most widely-used R packages for data science, such as ggplot2, dplyr, and many others. He has written several books about R, such as R for Data Science and Advanced R. Hadley is the Chief Scientist at RStudio. He is a technical advisor for Ursa Labs on R language support and general API design and usability.

---

# Initial Sponsors and Partners



**TWO SIGMA**

**Prospective sponsors / partners,  
please reach out: [info@ursalabs.org](mailto:info@ursalabs.org)**

# Getting involved

- Join [dev@arrow.apache.org](mailto:dev@arrow.apache.org)
- PRs to <https://github.com/apache/arrow>
- Learn more about the Ursa Labs vision for Arrow-powered data science: <https://ursalabs.org/tech/>